# SAM D21 Family

## Low-Power, 32-bit Cortex-M0+ MCU with Advanced Analog and PWM

## Features

- Processor
  - ARM® Cortex®-M0+ CPU running at up to 48 MHz
    - Single-cycle hardware multiplier
    - Micro Trace Buffer (MTB)
- Memories
  - 32/64/128/256 KB in-system self-programmable Flash
  - 4/8/16/32 KB SRAM Memory
- System
  - Power-on Reset (POR) and Brown-out Detection (BOD)
  - Internal and external clock options with 48 MHz Digital Frequency-Locked Loop (DFLL48M) and 48 MHz to 96 MHz Fractional Digital Phase-Locked Loop (FDPLL96M)
  - External Interrupt Controller (EIC)
  - 16 external interrupts
  - One Non-maskable Interrupt (NMI)
  - Two-pin Serial Wire Debug (SWD) programming, test and debugging interface
- Low Power
  - Idle and Stand-by Sleep modes
  - SleepWalking peripherals
- Peripherals
  - 12-channel Direct Memory Access Controller (DMAC)
  - 12-channel Event System
  - Up to five 16-bit Timer/Counters (TC), configurable as either:
    - One 16-bit TC with two compare/capture channels
    - One 8-bit TC with two compare/capture channels
    - One 32-bit TC with two compare/capture channels, by using two TCs
  - Three 24-bit Timer/Counters for Control (TCC), with extended functions:
    - Up to four compare channels with optional complementary output
    - Generation of synchronized pulse width modulation (PWM) pattern across port pins
    - Deterministic fault protection, fast decay and configurable dead-time between complementary output
    - Dithering that increase resolution with up to 5 bit and reduce quantization error
  - 32-bit Real Time Counter (RTC) with clock/calendar function
  - Watchdog Timer (WDT)

© 2018 Microchip Technology Inc.

  - CRC-32 generator
  - One full-speed (12 Mbps) Universal Serial Bus (USB) 2.0 interface
    - Embedded host and device function
    - Eight endpoints
  - Up to six Serial Communication Interfaces (SERCOM), each configurable to operate as either:
    - USART with full-duplex and single-wire half-duplex configuration
    - $I^2C$ up to 3.4 MHz
    - SPI
    - LIN slave
  - One two-channel Inter-IC Sound ($I^2S$) interface
  - One 12-bit, 350ksps Analog-to-Digital Converter (ADC) with up to 20 channels
    - Differential and single-ended input
    - 1/2x to 16x programmable gain stage
    - Automatic offset and gain error compensation
    - Oversampling and decimation in hardware to support 13-, 14-, 15- or 16-bit resolution
  - 10-bit, 350 ksps Digital-to-Analog Converter (DAC)
  - Up to four Analog Comparators (AC) with Window Compare function
  - Peripheral Touch Controller (PTC)
    - 256-Channel capacitive touch and proximity sensing
- I/O
  - Up to 52 programmable I/O pins
- Drop-in compatible with SAM D20
- Packages
  - 64-pin TQFP, QFN, UFBGA
  - 48-pin TQFP, QFN, WLCSP
  - 32-pin TQFP, QFN, WLCSP
- Operating Voltage
  - 1.62V – 3.63V

# Table of Contents


Features ..... 1

1. Description ..... 12

2. Configuration Summary ..... 13

3. Ordering Information[1] ..... 14

4. Block Diagram ..... 15

5. Pinout ..... 17
   - 5.1. SAM D21J ..... 17
   - 5.2. SAM D21G ..... 19
   - 5.3. SAM D21GxL ..... 21
   - 5.4. SAM D21ExA/B/C ..... 22
   - 5.5. SAM D21ExL ..... 24

6. Signal Descriptions List ..... 25

7. I/O Multiplexing and Considerations ..... 27
   - 7.1. Multiplexed Signals ..... 27
   - 7.2. Other Functions ..... 30

8. Power Supply and Start-Up Considerations ..... 34
   - 8.1. Power Domain Overview ..... 34
   - 8.2. Power Supply Considerations ..... 34
   - 8.3. Power-Up ..... 36
   - 8.4. Power-On Reset and Brown-Out Detector ..... 36

9. Product Mapping ..... 38

10. Memories ..... 40
    - 10.1. Embedded Memories ..... 40
    - 10.2. Physical Memory Map ..... 40
    - 10.3. NVM Calibration and Auxiliary Space ..... 41

11. Processor And Architecture ..... 45
    - 11.1. Cortex M0+ Processor ..... 45
    - 11.2. Nested Vector Interrupt Controller ..... 46
    - 11.3. Micro Trace Buffer ..... 48
    - 11.4. High-Speed Bus System ..... 49
    - 11.5. AHB-APB Bridge ..... 51
    - 11.6. PAC - Peripheral Access Controller ..... 52

12. Peripherals Configuration Summary ..... 68

13. DSU - Device Service Unit..... 70
13.1. Overview..... 70
13.2. Features..... 70
13.3. Block Diagram..... 71
13.4. Signal Description..... 71
13.5. Product Dependencies..... 71
13.6. Debug Operation..... 72
13.7. Chip Erase..... 74
13.8. Programming..... 75
13.9. Intellectual Property Protection..... 75
13.10. Device Identification..... 77
13.11. Functional Description..... 78
13.12. Register Summary..... 83
13.13. Register Description..... 85

14. Clock System..... 108
14.1. Clock Distribution..... 108
14.2. Synchronous and Asynchronous Clocks..... 109
14.3. Register Synchronization..... 109
14.4. Enabling a Peripheral..... 114
14.5. Disabling a Peripheral..... 114
14.6. On-demand, Clock Requests..... 114
14.7. Power Consumption vs. Speed..... 115
14.8. Clocks after Reset..... 115

15. GCLK - Generic Clock Controller..... 116
15.1. Overview..... 116
15.2. Features..... 116
15.3. Block Diagram..... 116
15.4. Signal Description..... 117
15.5. Product Dependencies..... 117
15.6. Functional Description..... 118
15.7. Register Summary..... 124
15.8. Register Description..... 124

16. PM – Power Manager..... 138
16.1. Overview..... 138
16.2. Features..... 138
16.3. Block Diagram..... 139
16.4. Signal Description..... 139
16.5. Product Dependencies..... 139
16.6. Functional Description..... 141
16.7. Register Summary..... 149
16.8. Register Description..... 149

17. SYSCTRL – System Controller..... 170
17.1. Overview..... 170

17.2. Features....170
17.3. Block Diagram....172
17.4. Signal Description....172
17.5. Product Dependencies....172
17.6. Functional Description....174
17.7. Register Summary....190
17.8. Register Description....192

18. WDT – Watchdog Timer....235
18.1. Overview....235
18.2. Features....235
18.3. Block Diagram....236
18.4. Signal Description....236
18.5. Product Dependencies....236
18.6. Functional Description....237
18.7. Register Summary....243
18.8. Register Description....243

19. RTC – Real-Time Counter....254
19.1. Overview....254
19.2. Features....254
19.3. Block Diagram....255
19.4. Signal Description....255
19.5. Product Dependencies....255
19.6. Functional Description....257
19.7. Register Summary....263
19.8. Register Description....265

20. DMAC – Direct Memory Access Controller....298
20.1. Overview....298
20.2. Features....298
20.3. Block Diagram....300
20.4. Signal Description....300
20.5. Product Dependencies....300
20.6. Functional Description....301
20.7. Register Summary....322
20.8. Register Description....323
20.9. Register Summary - SRAM....354
20.10. Register Description - SRAM....354

21. EIC – External Interrupt Controller....362
21.1. Overview....362
21.2. Features....362
21.3. Block Diagram....362
21.4. Signal Description....363
21.5. Product Dependencies....363
21.6. Functional Description....364
21.7. Register Summary....369

21.8. Register Description....369

22. NVMCTRL – Nonvolatile Memory Controller....380
22.1. Overview....380
22.2. Features....380
22.3. Block Diagram....380
22.4. Signal Description....381
22.5. Product Dependencies....381
22.6. Functional Description....382
22.7. Register Summary....389
22.8. Register Description....389

23. PORT - I/O Pin Controller....403
23.1. Overview....403
23.2. Features....403
23.3. Block Diagram....404
23.4. Signal Description....404
23.5. Product Dependencies....404
23.6. Functional Description....406
23.7. Register Summary....412
23.8. Register Description....414

24. EVSYS – Event System....435
24.1. Overview....435
24.2. Features....435
24.3. Block Diagram....435
24.4. Signal Description....436
24.5. Product Dependencies....436
24.6. Functional Description....437
24.7. Register Summary....442
24.8. Register Description....443

25. SERCOM – Serial Communication Interface....456
25.1. Overview....456
25.2. Features....456
25.3. Block Diagram....457
25.4. Signal Description....457
25.5. Product Dependencies....457
25.6. Functional Description....459

26. SERCOM USART....465
26.1. Overview....465
26.2. USART Features....465
26.3. Block Diagram....466
26.4. Signal Description....466
26.5. Product Dependencies....466
26.6. Functional Description....468
26.7. Register Summary....482

26.8. Register Description....483

27. SERCOM SPI – SERCOM Serial Peripheral Interface....507
27.1. Overview....507
27.2. Features....507
27.3. Block Diagram....508
27.4. Signal Description....508
27.5. Product Dependencies....508
27.6. Functional Description....510
27.7. Register Summary....522
27.8. Register Description....523

28. SERCOM I2C – Inter-Integrated Circuit....545
28.1. Overview....545
28.2. Features....545
28.3. Block Diagram....546
28.4. Signal Description....546
28.5. Product Dependencies....546
28.6. Functional Description....548
28.7. Register Summary - I2C Slave....572
28.8. Register Description - I2C Slave....573
28.9. Register Summary - I2C Master....595
28.10. Register Description - I2C Master....596

29. I2S - Inter-IC Sound Controller....622
29.1. Overview....622
29.2. Features....622
29.3. Block Diagram....623
29.4. Signal Description....624
29.5. Product Dependencies....624
29.6. Functional Description....626
29.7. I2S Application Examples....637
29.8. Register Summary....640
29.9. Register Description....641

30. TC – Timer/Counter....659
30.1. Overview....659
30.2. Features....659
30.3. Block Diagram....660
30.4. Signal Description....660
30.5. Product Dependencies....661
30.6. Functional Description....662
30.7. Register Summary....674
30.8. Register Description....677

31. TCC – Timer/Counter for Control Applications....701
31.1. Overview....701
31.2. Features....701

31.3. Block Diagram....702
31.4. Signal Description....703
31.5. Product Dependencies....703
31.6. Functional Description....705
31.7. Register Summary....739
31.8. Register Description....741

32. USB – Universal Serial Bus....788
32.1. Overview....788
32.2. Features....788
32.3. USB Block Diagram....789
32.4. Signal Description....789
32.5. Product Dependencies....789
32.6. Functional Description....791
32.7. Register Summary....811
32.8. Register Description....815

33. ADC – Analog-to-Digital Converter....888
33.1. Overview....888
33.2. Features....888
33.3. Block Diagram....889
33.4. Signal Description....889
33.5. Product Dependencies....890
33.6. Functional Description....891
33.7. Register Summary....901
33.8. Register Description....902

34. AC – Analog Comparators....927
34.1. Overview....927
34.2. Features....927
34.3. Block Diagram....928
34.4. Signal Description....929
34.5. Product Dependencies....929
34.6. Functional Description....930
34.7. Register Summary....940
34.8. Register Description....940

35. DAC – Digital-to-Analog Converter....957
35.1. Overview....957
35.2. Features....957
35.3. Block Diagram....957
35.4. Signal Description....957
35.5. Product Dependencies....957
35.6. Functional Description....959
35.7. Register Summary....963
35.8. Register Description....963

36. Peripheral Touch Controller (PTC)....974

36.1. Overview....974
36.2. Features....974
36.3. Block Diagram....975
36.4. Signal Description....976
36.5. System Dependencies....976
36.6. Functional Description....978

37. Electrical Characteristics....979
37.1. Disclaimer....979
37.2. Absolute Maximum Ratings....979
37.3. General Operating Ratings....979
37.4. Supply Characteristics....980
37.5. Maximum Clock Frequencies....980
37.6. Power Consumption....982
37.7. Peripheral Power Consumption....987
37.8. I/O Pin Characteristics....990
37.9. Injection Current....992
37.10. Analog Characteristics....993
37.11. NVM Characteristics....1007
37.12. Oscillators Characteristics....1008
37.13. PTC Typical Characteristics....1016
37.14. USB Characteristics....1022
37.15. Timing Characteristics....1023

38. Electrical Characteristics at 105°C....1033
38.1. Disclaimer....1033
38.2. Absolute Maximum Ratings....1033
38.3. General Operating Ratings....1033
38.4. Maximum Clock Frequencies....1034
38.5. Power Consumption....1035
38.6. Analog Characteristics....1038
38.7. NVM Characteristics....1046
38.8. Oscillators Characteristics....1047
38.9. USB Characteristics....1053
38.10. Timing Characteristics....1053

39. Electrical Characteristics at 125°C....1054
39.1. Disclaimer....1054
39.2. Absolute Maximum Ratings....1054
39.3. General Operating Ratings....1054
39.4. Maximum Clock Frequencies....1055
39.5. Power Consumption....1058
39.6. Analog Characteristics....1061
39.7. NVM Characteristics....1073
39.8. Oscillators Characteristics....1074
39.9. Timing Characteristics....1084

40. Packaging Information....1089

40.1. Thermal Considerations ... 1089
40.2. Package Drawings ... 1090
40.3. Soldering Profile ... 1109

41. Schematic Checklist ... 1110
41.1. Introduction ... 1110
41.2. Power Supply ... 1110
41.3. External Analog Reference Connections ... 1111
41.4. External Reset Circuit ... 1112
41.5. Clocks and Crystal Oscillators ... 1113
41.6. Unused or Unconnected Pins ... 1117
41.7. Programming and Debug Ports ... 1117
41.8. USB Interface ... 1120

42. Conventions ... 1122
42.1. Numerical Notation ... 1122
42.2. Memory Size and Type ... 1122
42.3. Frequency and Time ... 1122
42.4. Registers and Bits ... 1123

43. Acronyms and Abbreviations ... 1124

44. Data Sheet Revision History ... 1127
44.1. Rev. B – 04/2018 ... 1127
44.2. Rev. A – 01/2017 ... 1127
44.3. Rev. O – 12/2016 ... 1128
44.4. Rev. N – 10/2016 ... 1128
44.5. Rev. M – 09/2016 ... 1129
44.6. Rev. L – 09/2016 ... 1129
44.7. Rev. K – 09/2016 ... 1130
44.8. Rev. J – 07/2016 ... 1131
44.9. Rev. I – 03/2016 ... 1131
44.10. Rev. H – 01/2016 ... 1132
44.11. Rev. G – 09/2015 ... 1133
44.12. Rev. F – 07/2015 ... 1134
44.13. Rev. E – 02/2015 ... 1135
44.14. Rev. D – 09/2014 ... 1136
44.15. Rev. C – 07/2014 ... 1137
44.16. Rev. B – 07/2014 ... 1137
44.17. Rev. A - 02/2014 ... 1142

The Microchip Web Site ... 1143

Customer Change Notification Service ... 1143

Customer Support ... 1143

Product Identification System ... 1144

Microchip Devices Code Protection Feature..... 1144

Legal Notice.....1145

Trademarks.....1145

Quality Management System Certified by DNV.....1146

Worldwide Sales and Service..... 1147

# 1. Description

The SAM D21 is a series of low-power microcontrollers using the 32-bit ARM® Cortex®-M0+ processor, and ranging from 32-pins to 64-pins with up to 256 KB Flash and 32 KB of SRAM. The SAM D21 operate at a maximum frequency of 48 MHz and reach 2.46 CoreMark/MHz. They are designed for simple and intuitive migration with identical peripheral modules, hex compatible code, identical linear address map, and pin compatible migration paths between all devices in the product series. All devices include intelligent and flexible peripherals, Event System for inter-peripheral signaling, and support for capacitive touch button, slider, and wheel user interfaces.

The SAM D21 provide the following features: In-system programmable Flash, 12-channel Direct Memory Access Controller (DMAC), 12-channel Event System, programmable Interrupt Controller, up to 52 programmable I/O pins, 32-bit Real-Time Clock and Calendar (RTC), up to five 16-bit Timer/Counters (TC) and three 24-bit Timer/Counters for Control (TCC), where each TC can be configured to perform frequency and waveform generation, accurate program execution timing or input capture with time and frequency measurement of digital signals. The TCs can operate in 8- or 16-bit mode, selected TCs can be cascaded to form a 32-bit TC, and three timer/counters have extended functions optimized for motor, lighting, and other control applications. The series provide one full-speed USB 2.0 embedded host and device interface; up to six Serial Communication Modules (SERCOM) that each can be configured to act as an USART, UART, SPI, $I^2C$ up to 3.4MHz, SMBus, PMBus, and LIN slave; two-channel $I^2S$ interface; up to twenty-channel 350 ksps 12-bit ADC with programmable gain and optional oversampling and decimation supporting up to 16-bit resolution, one 10-bit 350 ksps DAC, up to four analog comparators with Window mode, Peripheral Touch Controller (PTG) supporting up to 256 buttons, sliders, wheels, and proximity sensing; programmable Watchdog Timer (WDT), brown-out detector and power-on Reset and two-pin Serial Wire Debug (SWD) program and debug interface.

All devices have accurate and low-power external and internal oscillators. All oscillators can be used as a source for the system clock. Different clock domains can be independently configured to run at different frequencies, enabling power saving by running each peripheral at its optimal clock frequency, and thus maintaining a high CPU frequency while reducing power consumption.

The SAM D21 have two software-selectable sleep modes, Idle and Stand-by. In Idle mode, the CPU is stopped while all other functions can be kept running. In Stand-by mode, all clocks and functions are stopped, expect those selected to continue running. The device supports SleepWalking. This feature allows the peripheral to wake up from sleep based on predefined conditions, and thus allows the CPU to wake up only when needed, e.g., when a threshold is crossed or a result is ready. The Event System supports synchronous and asynchronous events, allowing peripherals to receive, react to and send events even in Stand-by mode.

The Flash program memory can be reprogrammed in-system through the SWD interface. The same interface can be used for non-intrusive on-chip debug of application code. A boot loader running in the device can use any communication interface to download and upgrade the application program in the Flash memory.

The SAM D21 microcontrollers are supported with a full suite of program and system development tools, including C compilers, macro assemblers, program debugger/simulators, programmers, and evaluation kits.

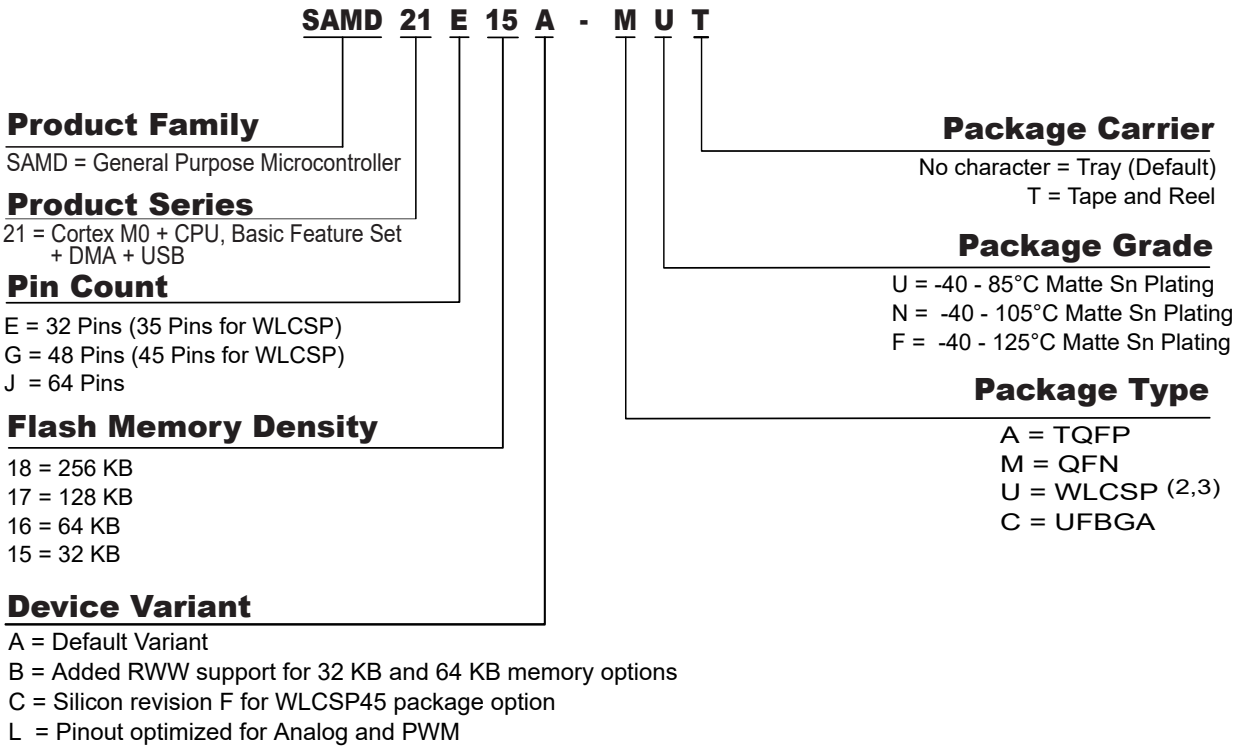
# 2. Configuration Summary

**Table 2-1. SAM D21 E/G/J and SAM D21 EL/GL Product Family Features**

| Device | Program Memory (KB) | Data Memory (KB) | Pins | Packages | Oscillators | | Peripherals | | | | | | | | | | | | Analog | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Internal | External | USB | SERCOM | TC/Waveform Output Channels per TCx Instance | TCC | Waveform Output Channels per TCCx Instance | I2S | DMA Channels | RTC | WDT | Event System (Channels) | External Interrupt Lines | I/O Pins | ADC Channels | Analog Comparator | DAC | PTC (Mutual/Self-capacitance Channels)" |
| ATSAMD21E15A | 32 | 4 | 32 | TQFP, QFN | OSC32K, OSCULP32K, OSC8M, DFLL48M, FDPLL96M | XOSC32K, XOSC, | Y | 4 | 3/2 | 3 | 6/4/2 | Y | 12 | Y | Y | 12 | 16 | 26 | 10 | 2 | Y | 60/6 |
| ATSAMD21E16A | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E17A | 128 | 16 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E18A | 256 | 32 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E15B | 32 | 4 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E16B | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E15C | 32 | 4 | 35 | WLCSP | | | | | | | | | | | | | | | | | | |
| ATSAMD21E16C | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G15A | 32 | 4 | 48 | TQFP, QFN | | | | 6 | | | 8/4/2 | | | | | | | 38 | 14 | | | 120/10 |
| ATSAMD21G16A | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G17A (see **Note 1**) | 128 | 16 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G18A (see **Note 1**) | 256 | 32 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G15B | 32 | 4 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G16B | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21J15A | 32 | 4 | 64 | TQFP, QFN | | | | | 5/2 | | | | | | | | | 52 | 20 | | | 256/16 |
| ATSAMD21J16A | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21J17A | 128 | 16 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21J18A | 256 | 32 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21J15B | 32 | 4 | | TQFP,QFN,UFBGA | | | | | | | | | | | | | | | | | | |
| ATSAMD21J16B | 64 | 8 | | | | | | | | | | | | | | | | | | | | |
| ATSAMD21E15L | 32 | 4 | 32 | TQFP, QFN | | XOSC | N | 4 | 3/2 | | 6/4/2 | N | | | | | | 26 | 14 | 4 | | N |
| ATSAMD21E16L | 64 | 8 | 32 | | | | | | | | | | | | | | | | | | | |
| ATSAMD21G16L | 64 | 8 | 48 | QFN | | | | 6 | 5/2 | | 8/4/2 | | | | | | | 38 | 18 | | | |

**Note:** This part number is also available in a 45-Ball WLCSP package with a total of five TC instances and 15 ADC Channels..

# 3. Ordering Information(1)

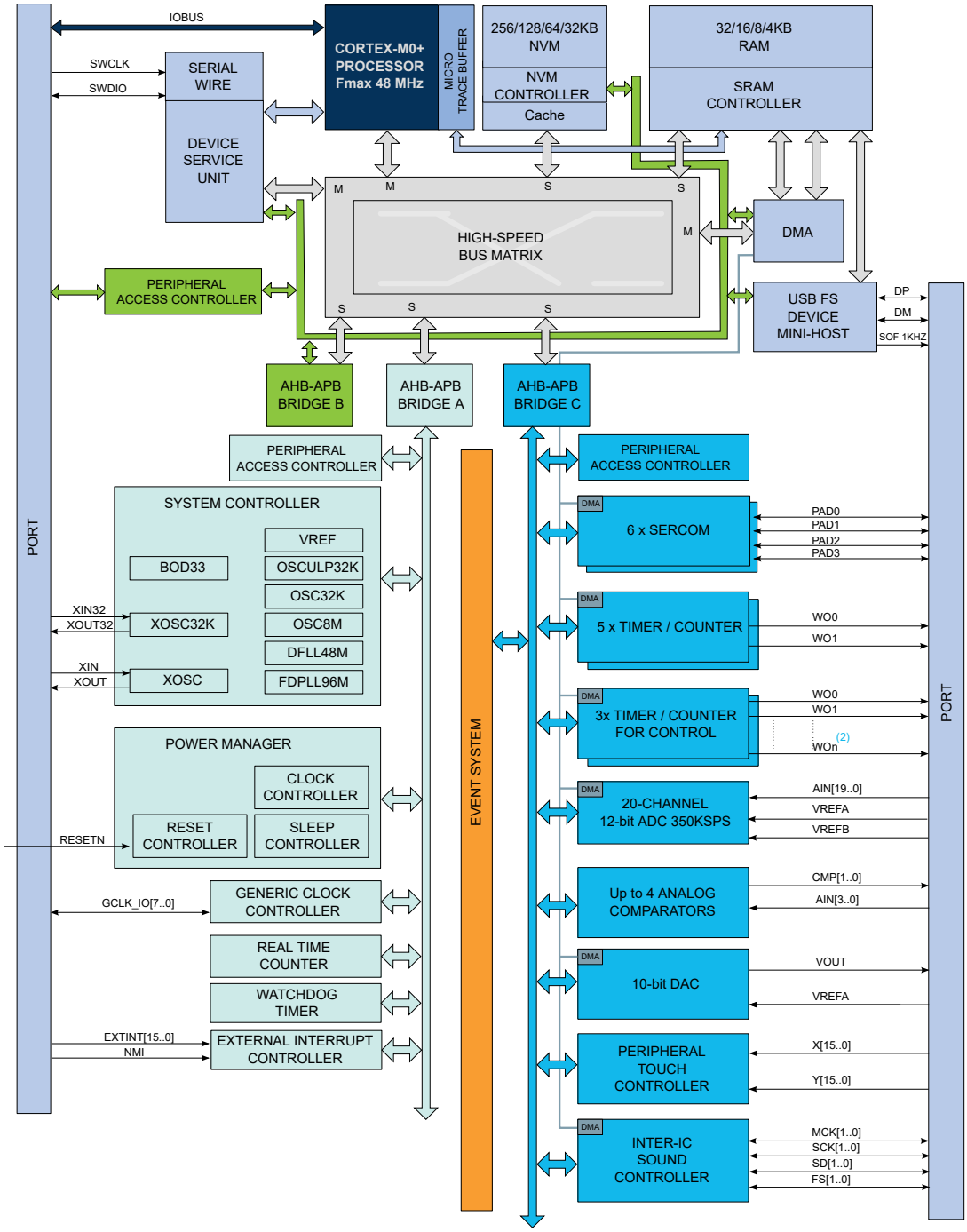



**Note:** 1. Not all combinations are valid. The available ordering numbers are listed in the Configuration Summary.

**Note:** 2. WLCSP package is available in -40C to 85C operating temperature range.

**Note:** 3. WLCSP parts are programmed with a specific SPI/$I^2C$ bootloader. Refer to the "Application Note AT09002" for additional information. Contact Microchip Sales office for additional information on availability.

# 4. Block Diagram




1. Some products have different number of SERCOM instances, Timer/Counter instances, PTC signals and ADC signals. Refer to the Configuration Summary for details.
2. The three TCC instances have different configurations, including the number of Waveform Output (WO) lines. Refer to the TCC Configuration for details.

**Related Links**

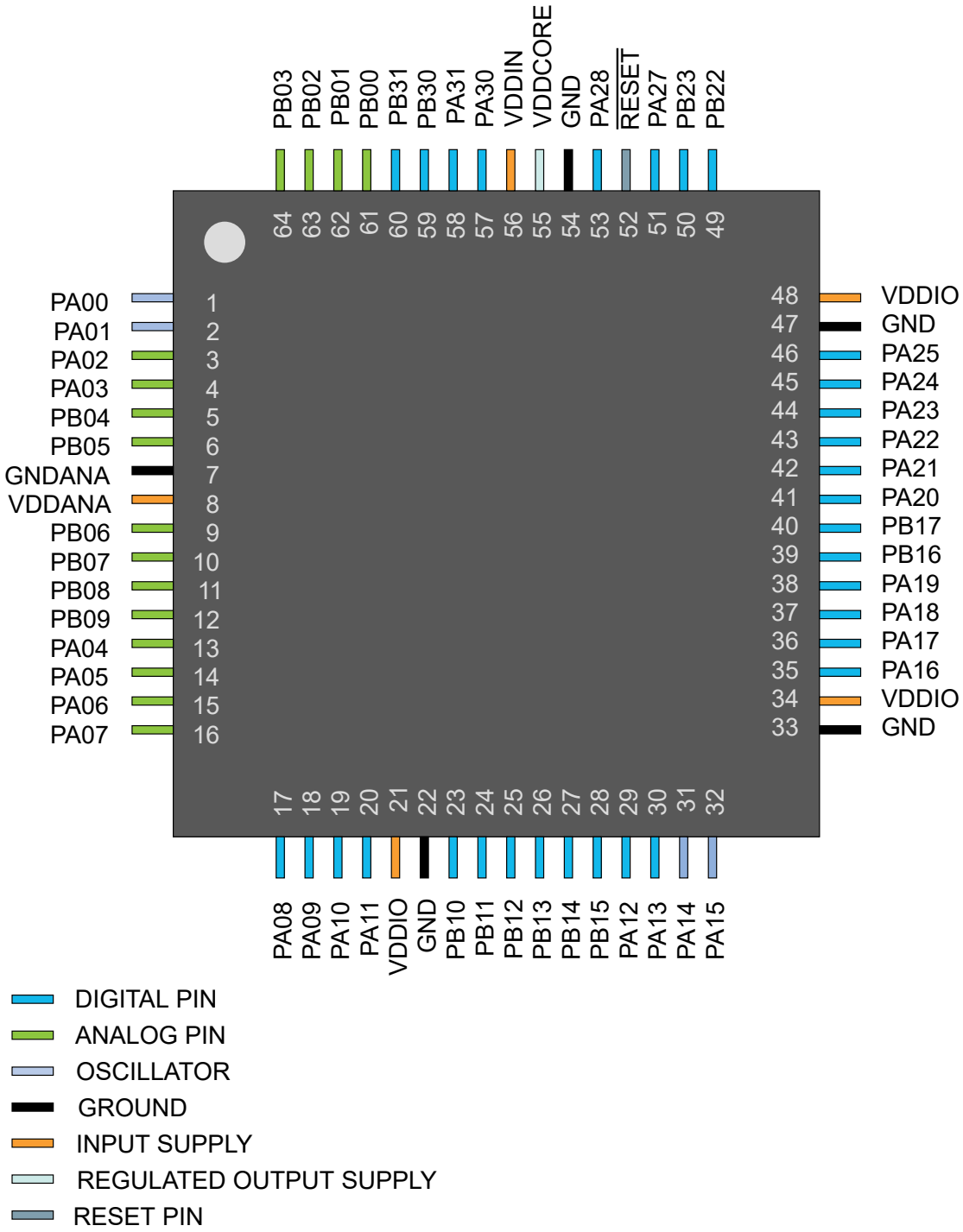
2. Configuration Summary

7.2.5 TCC Configurations

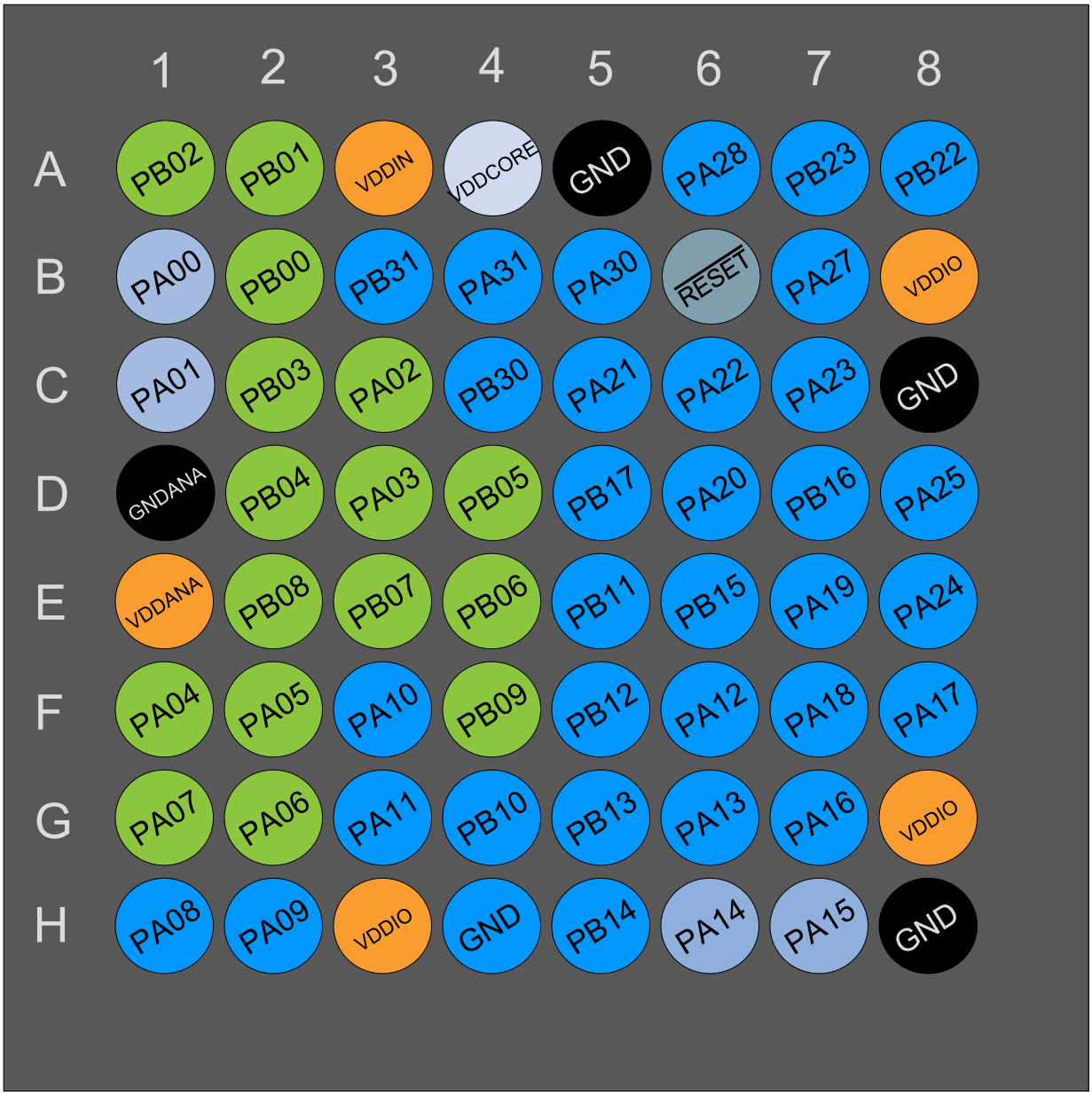
# 5. Pinout

## 5.1 SAM D21J

### 5.1.1 QFN64 / TQFP64



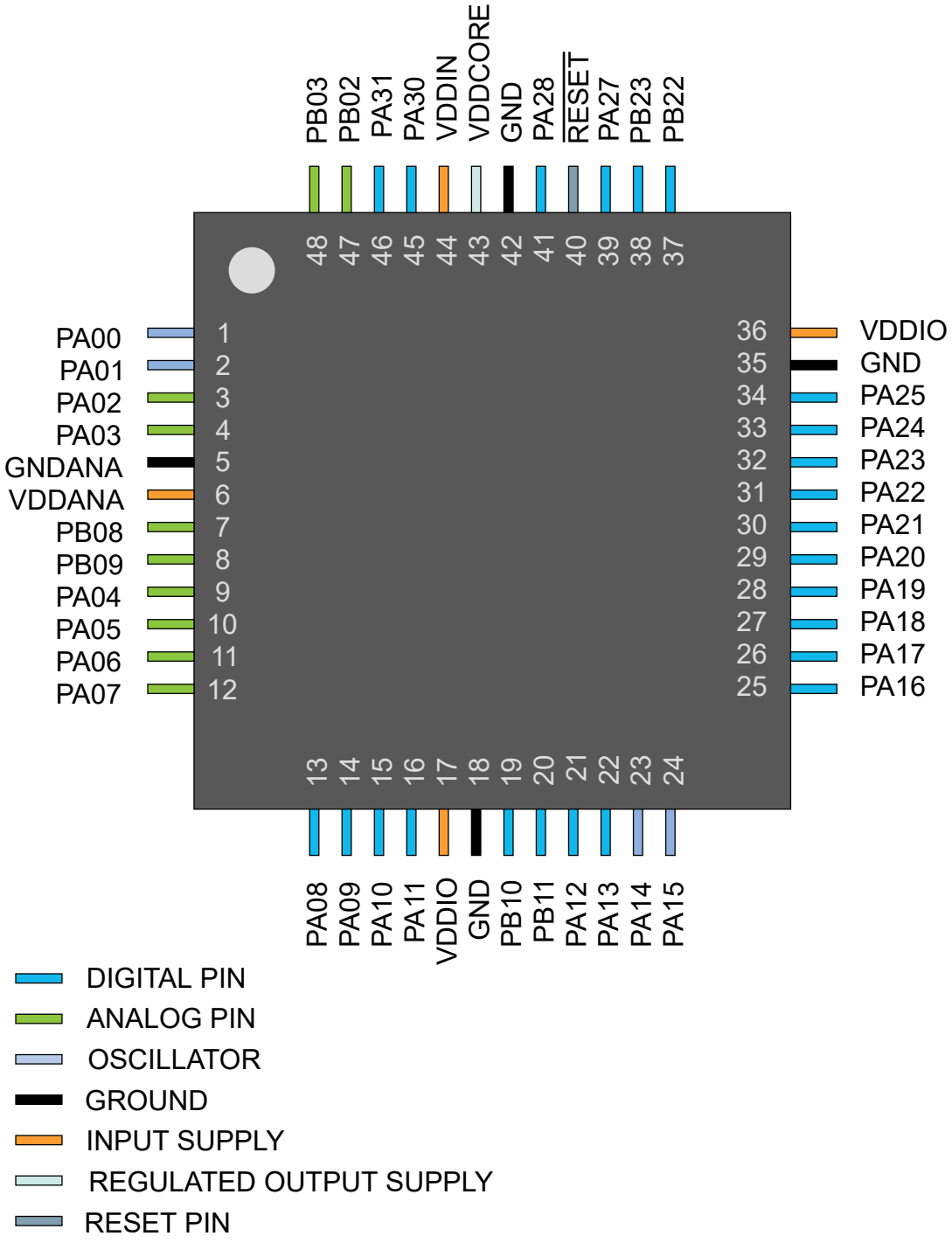
### 5.1.2 UFBGA64



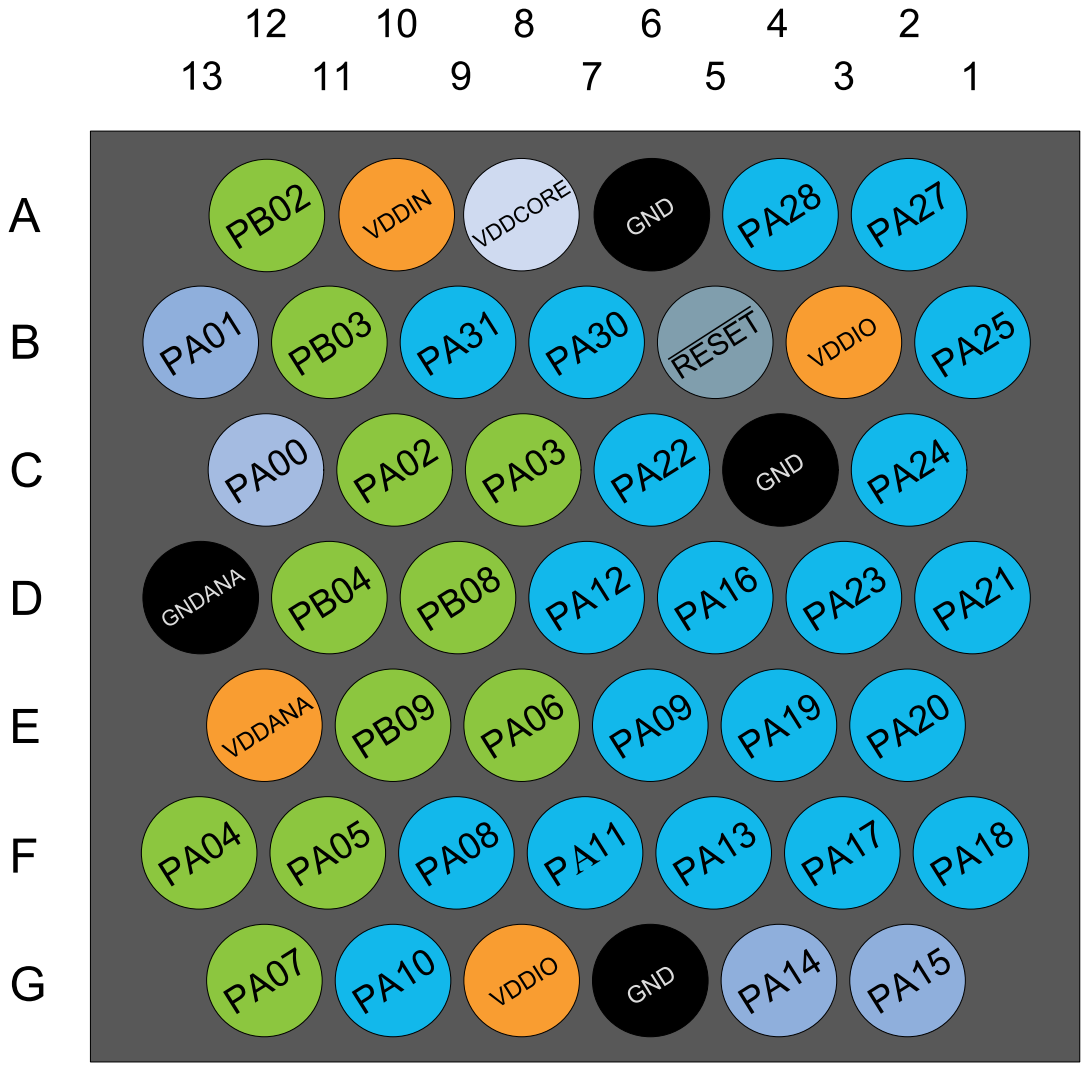

- DIGITAL PIN
- ANALOG PIN
- OSCILLATOR
- GROUND
- INPUT SUPPLY
- REGULATED OUTPUT SUPPLY
- RESET PIN

## 5.2 SAM D21G

### 5.2.1 QFN48 / TQFP48

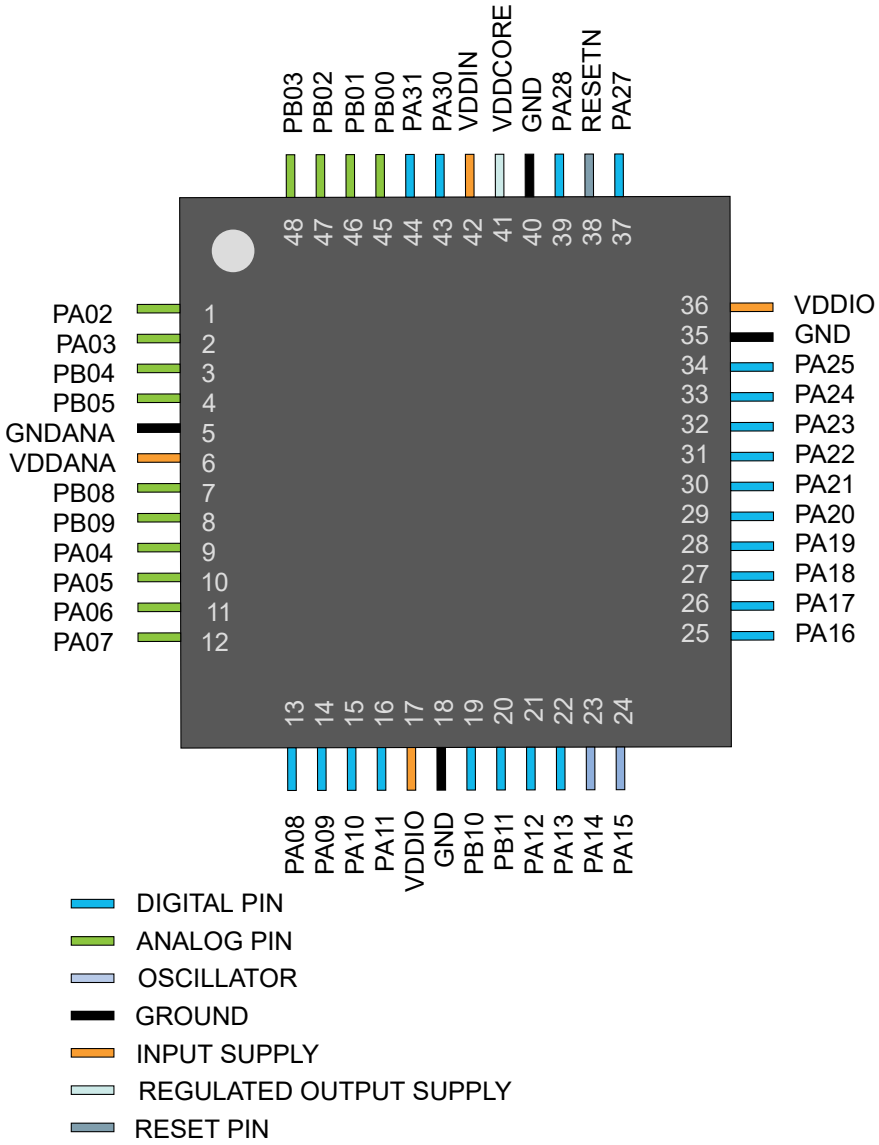


### 5.2.2 WLCSP45


12 10 8 6 4 2
13 11 9 7 5 3 1
A
B
C
D
E
F
G
PB02 VDDIN VDDCORE GND PA28 PA27
PA01 PB03 PA31 PA30 RESET VDDIO PA25
PA00 PA02 PA03 PA22 GND PA24
GNDANA PB04 PB08 PA12 PA16 PA23 PA21
VDDANA PB09 PA06 PA09 PA19 PA20
PA04 PA05 PA08 PA11 PA13 PA17 PA18
PA07 PA10 VDDIO GND PA14 PA15
DIGITAL PIN
ANALOG PIN
OSCILLATOR
GROUND
INPUT SUPPLY
REGULATED OUTPUT SUPPLY
RESET PIN

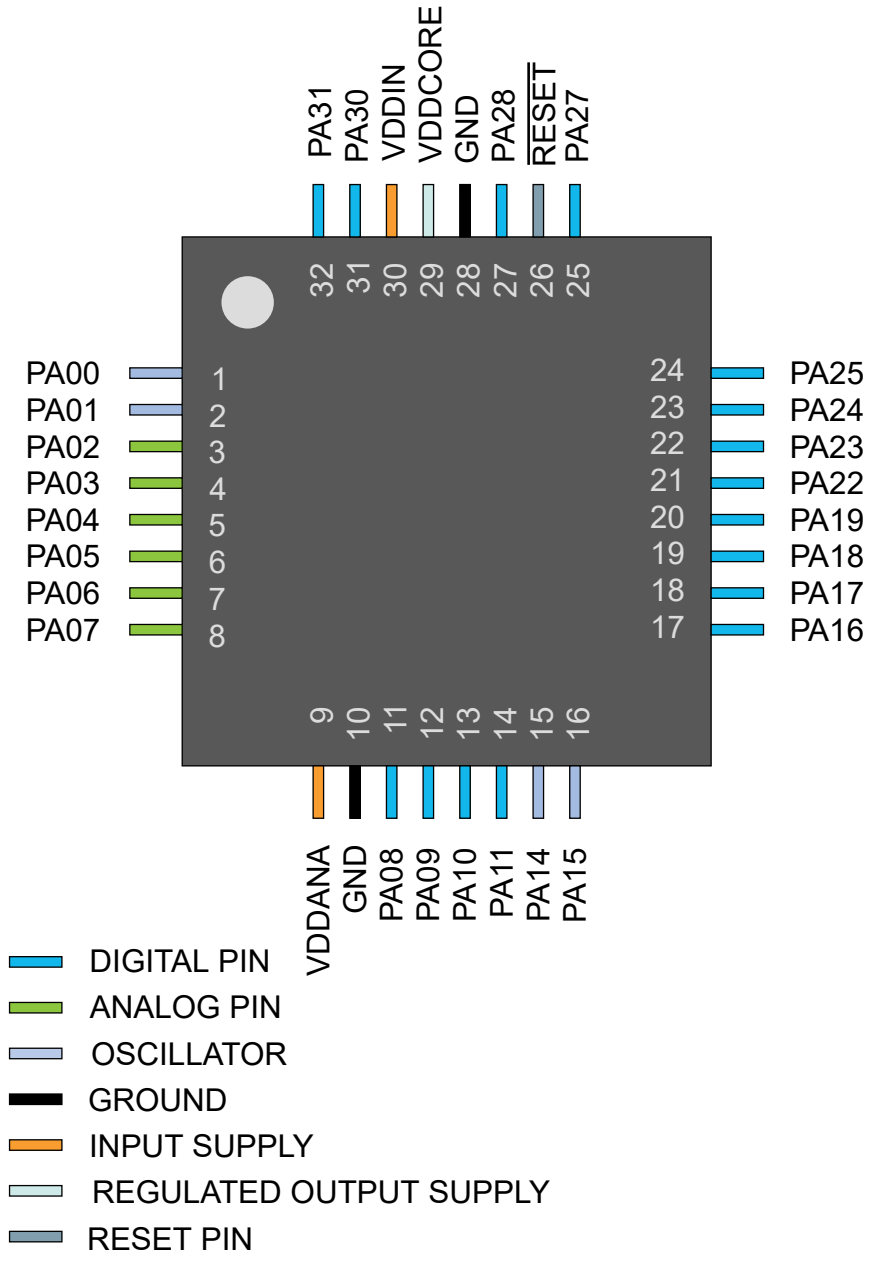
## 5.3 SAM D21GxL

### 5.3.1 QFN48

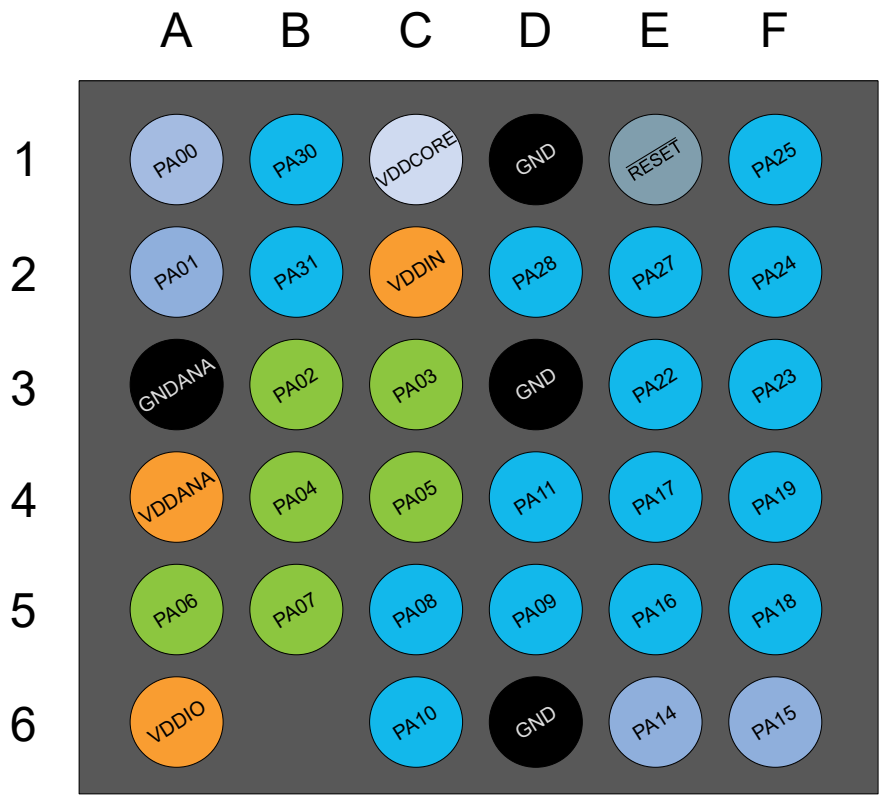


## 5.4 SAM D21ExA/B/C

### 5.4.1 QFN32 / TQFP32

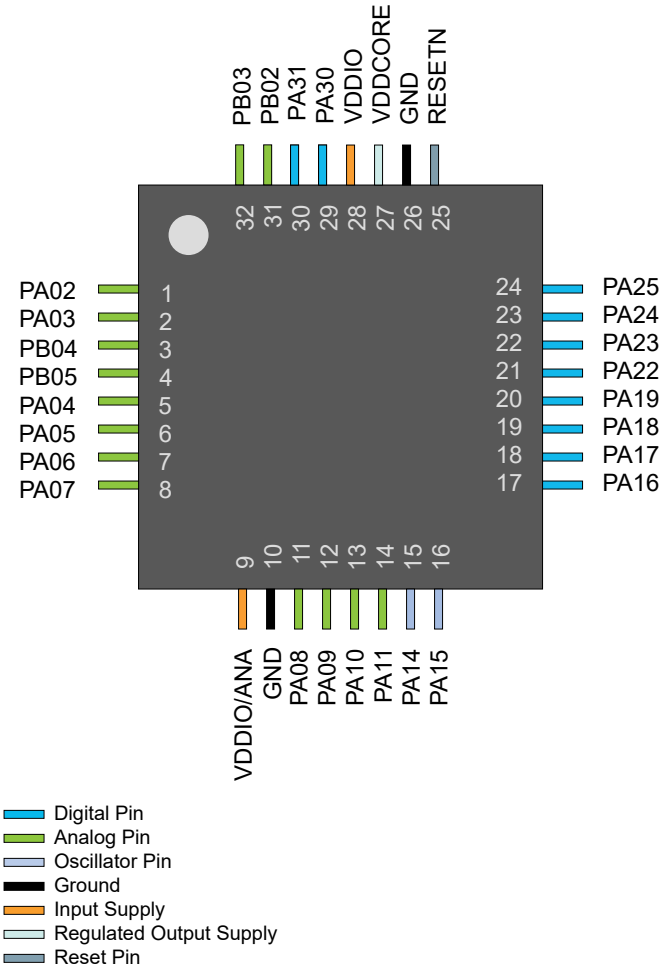


### 5.4.2 WLCSP35


A
B
C
D
E
F
1
2
3
4
5
6
PA00
PA30
VDDCORE
GND
RESET
PA25
PA01
PA31
VDDIN
PA28
PA27
PA24
GNDANA
PA02
PA03
GND
PA22
PA23
VDDANA
PA04
PA05
PA11
PA17
PA19
PA06
PA07
PA08
PA09
PA16
PA18
VDDIO
PA10
GND
PA14
PA15
DIGITAL PIN
ANALOG PIN
OSCILLATOR
GROUND
INPUT SUPPLY
REGULATED OUTPUT SUPPLY
RESET PIN

## 5.5 SAM D21ExL

### 5.5.1 QFN32 / TQFP32



# 6. Signal Descriptions List

The following table gives details on signal names classified by peripheral.

| Signal Name | Function | Type | Active Level |
|---|---|---|---|
| Analog Comparators - AC | | | |
| AIN[3:0] | AC Analog Inputs | Analog | |
| CMP[:0] | AC Comparator Outputs | Digital | |
| Analog Digital Converter - ADC | | | |
| AIN[19:0] | ADC Analog Inputs | Analog | |
| VREFA | ADC Voltage External Reference A | Analog | |
| VREFB | ADC Voltage External Reference B | Analog | |
| Digital Analog Converter - DAC | | | |
| VOUT | DAC Voltage output | Analog | |
| VREFA | DAC Voltage External Reference | Analog | |
| External Interrupt Controller | | | |
| EXTINT[15:0] | External Interrupts | Input | |
| NMI | External Non-Maskable Interrupt | Input | |
| Generic Clock Generator - GCLK | | | |
| GCLK_IO[7:0] | Generic Clock (source clock or generic clock generator output) | I/O | |
| Inter-IC Sound Controller - $I^2S$ | | | |
| MCK[1:0] | Master Clock | I/O | |
| SCK[1:0] | Serial Clock | I/O | |
| FS[1:0] | $I^2S$ Word Select or TDM Frame Sync | I/O | |
| SD[1:0] | Serial Data Input or Output | I/O | |
| Power Manager - PM | | | |
| RESETN | Reset | Input | Low |
| Serial Communication Interface - SERCOMx | | | |
| PAD[3:0] | SERCOM I/O Pads | I/O | |
| System Control - SYSCTRL | | | |
| XIN | Crystal Input | Analog/ Digital | |
| XIN32 | 32kHz Crystal Input | Analog/ Digital | |
| XOUT | Crystal Output | Analog | |

| Signal Name | Function | Type | Active Level |
|---|---|---|---|
| XOUT32 | 32kHz Crystal Output | Analog | |
| Timer Counter - TCx | | | |
| WO[1:0] | Waveform Outputs | Output | |
| Timer Counter - TCCx | | | |
| WO[1:0] | Waveform Outputs | Output | |
| Peripheral Touch Controller - PTC | | | |
| X[15:0] | PTC Input | Analog | |
| Y[15:0] | PTC Input | Analog | |
| General Purpose I/O - PORT | | | |
| PA25 - PA00 | Parallel I/O Controller I/O Port A | I/O | |
| PA28 - PA27 | Parallel I/O Controller I/O Port A | I/O | |
| PA31 - PA30 | Parallel I/O Controller I/O Port A | I/O | |
| PB17 - PB00 | Parallel I/O Controller I/O Port B | I/O | |
| PB23 - PB22 | Parallel I/O Controller I/O Port B | I/O | |
| PB31 - PB30 | Parallel I/O Controller I/O Port B | I/O | |
| Universal Serial Bus - USB | | | |
| DP | DP for USB | I/O | |
| DM | DM for USB | I/O | |
| SOF 1kHz | USB Start of Frame | I/O | |

# 7. I/O Multiplexing and Considerations

## 7.1 Multiplexed Signals

Each pin is by default controlled by the PORT as a general purpose I/O and alternatively it can be assigned to one of the peripheral functions A, B, C, D, E, F, G or H. To enable a peripheral function on a pin, the Peripheral Multiplexer Enable bit in the Pin Configuration register corresponding to that pin (PINCFGn.PMUXEN, n = 0-31) in the PORT must be written to one. The selection of peripheral function A to H is done by writing to the Peripheral Multiplexing Odd and Even bits in the Peripheral Multiplexing register (PMUXn.PMUXE/O) in the PORT.

This table describes the peripheral signals multiplexed to the PORT I/O pins.

**Table 7-1. PORT Function Multiplexing for A, B, and C Variant Devices**

| Pin(1) | | | I/O Pin | Supply | A | B(2)(3) | | | | | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SAMD21E | SAMD21G | SAMD21J | | | EIC | REF | ADC | AC | PTC | DAC | SERCOM(2)(3) | SERCOM-ALT | TC(4) /TCC | TCC | COM | AC/ GCLK |
| 1 | 1 | 1 | PA00 | VDDANA | EXTINT[0] | | | | | | | SERCOM1/ PAD[0] | TCC2/WO[0] | | | |
| 2 | 2 | 2 | PA01 | VDDANA | EXTINT[1] | | | | | | | SERCOM1/ PAD[1] | TCC2/WO[1] | | | |
| 3 | 3 | 3 | PA02 | VDDANA | EXTINT[2] | | AIN[0] | | Y[0] | VOUT | | | | | | |
| 4 | 4 | 4 | PA03 | VDDANA | EXTINT[3] | ADC/ VREFA DAC/ VREFA | AIN[1] | | Y[1] | | | | | | | |
| | | 5 | PB04 | VDDANA | EXTINT[4] | | AIN[12] | | Y[10] | | | | | | | |
| | | 6 | PB05 | VDDANA | EXTINT[5] | | AIN[13] | | Y[11] | | | | | | | |
| | | 9 | PB06 | VDDANA | EXTINT[6] | | AIN[14] | | Y[12] | | | | | | | |
| | | 10 | PB07 | VDDANA | EXTINT[7] | | AIN[15] | | Y[13] | | | | | | | |
| | 7 | 11 | PB08 | VDDANA | EXTINT[8] | | AIN[2] | | Y[14] | | | SERCOM4/ PAD[0] | TC4/WO[0] | | | |
| | 8 | 12 | PB09 | VDDANA | EXTINT[9] | | AIN[3] | | Y[15] | | | SERCOM4/ PAD[1] | TC4/WO[1] | | | |
| 5 | 9 | 13 | PA04 | VDDANA | EXTINT[4] | ADC/ VREFB | AIN[4] | AIN[0] | Y[2] | | | SERCOM0/ PAD[0] | TCC0/WO[0] | | | |
| 6 | 10 | 14 | PA05 | VDDANA | EXTINT[5] | | AIN[5] | AIN[1] | Y[3] | | | SERCOM0/ PAD[1] | TCC0/WO[1] | | | |
| 7 | 11 | 15 | PA06 | VDDANA | EXTINT[6] | | AIN[6] | AIN[2] | Y[4] | | | SERCOM0/ PAD[2] | TCC1/WO[0] | | | |
| 8 | 12 | 16 | PA07 | VDDANA | EXTINT[7] | | AIN[7] | AIN[3] | Y[5] | | | SERCOM0/ PAD[3] | TCC1/WO[1] | | I2S/SD[0] | |
| 11 | 13 | 17 | PA08 | VDDIO | NMI | | AIN[16] | | X[0] | | SERCOM0/ PAD[0] | SERCOM2/ PAD[0] | TCC0/WO[0] | TCC1/ WO[2] | I2S/SD[1] | |
| 12 | 14 | 18 | PA09 | VDDIO | EXTINT[9] | | AIN[17] | | X[1] | | SERCOM0/ PAD[1] | SERCOM2/ PAD[1] | TCC0/WO[1] | TCC1/ WO[3] | I2S/ MCK[0] | |
| 13 | 15 | 19 | PA10 | VDDIO | EXTINT[10] | | AIN[18] | | X[2] | | SERCOM0/ PAD[2] | SERCOM2/ PAD[2] | TCC1/WO[0] | TCC0/ WO[2] | I2S/ SCK[0] | GCLK_IO[4] |
| 14 | 16 | 20 | PA11 | VDDIO | EXTINT[11] | | AIN[19] | | X[3] | | SERCOM0/ PAD[3] | SERCOM2/ PAD[3] | TCC1/WO[1] | TCC0/ WO[3] | I2S/FS[0] | GCLK_IO[5] |
| | 19 | 23 | PB10 | VDDIO | EXTINT[10] | | | | | | | SERCOM4/ PAD[2] | TC5/WO[0] | TCC0/ WO[4] | I2S/ MCK[1] | GCLK_IO[4] |
| | 20 | 24 | PB11 | VDDIO | EXTINT[11] | | | | | | | SERCOM4/ PAD[3] | TC5/WO[1] | TCC0/ WO[5] | I2S/ SCK[1] | GCLK_IO[5] |
| | | 25 | PB12 | VDDIO | EXTINT[12] | | | | X[12] | | SERCOM4/ PAD[0] | | TC4/WO[0] | TCC0/ WO[6] | I2S/FS[1] | GCLK_IO[6] |
| | | 26 | PB13 | VDDIO | EXTINT[13] | | | | X[13] | | SERCOM4/ PAD[1] | | TC4/WO[1] | TCC0/ WO[7] | | GCLK_IO[7] |

| Pin[1] | | | I/O Pin | Supply | A | B[2][3] | | | | | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SAMD21E | SAMD21G | SAMD21J | | | EIC | REF | ADC | AC | PTC | DAC | SERCOM[2][3] | SERCOM-ALT | TC[4] /TCC | TCC | COM | AC/ GCLK |
| | | 27 | PB14 | VDDIO | EXTINT[14] | | | | X[14] | | SERCOM4/ PAD[2] | | TC5/WO[0] | | | GCLK_IO[0] |
| | | 28 | PB15 | VDDIO | EXTINT[15] | | | | X[15] | | SERCOM4/ PAD[3] | | TC5/WO[1] | | | GCLK_IO[1] |
| | 21 | 29 | PA12 | VDDIO | EXTINT[12] | | | | | | SERCOM2/ PAD[0] | SERCOM4/ PAD[0] | TCC2/WO[0] | TCC0/ WO[6] | | AC/CMP[0] |
| | 22 | 30 | PA13 | VDDIO | EXTINT[13] | | | | | | SERCOM2/ PAD[1] | SERCOM4/ PAD[1] | TCC2/WO[1] | TCC0/ WO[7] | | AC/CMP[1] |
| 15 | 23 | 31 | PA14 | VDDIO | EXTINT[14] | | | | | | SERCOM2/ PAD[2] | SERCOM4/ PAD[2] | TC3/WO[0] | TCC0/ WO[4] | | GCLK_IO[0] |
| 16 | 24 | 32 | PA15 | VDDIO | EXTINT[15] | | | | | | SERCOM2/ PAD[3] | SERCOM4/ PAD[3] | TC3/WO[1] | TCC0/ WO[5] | | GCLK_IO[1] |
| 17 | 25 | 35 | PA16 | VDDIO | EXTINT[0] | | | | X[4] | | SERCOM1/ PAD[0] | SERCOM3/ PAD[0] | TCC2/WO[0] | TCC0/ WO[6] | | GCLK_IO[2] |
| 18 | 26 | 36 | PA17 | VDDIO | EXTINT[1] | | | | X[5] | | SERCOM1/ PAD[1] | SERCOM3/ PAD[1] | TCC2/WO[1] | TCC0/ WO[7] | | GCLK_IO[3] |
| 19 | 27 | 37 | PA18 | VDDIO | EXTINT[2] | | | | X[6] | | SERCOM1/ PAD[2] | SERCOM3/ PAD[2] | TC3/WO[0] | TCC0/ WO[2] | | AC/CMP[0] |
| 20 | 28 | 38 | PA19 | VDDIO | EXTINT[3] | | | | X[7] | | SERCOM1/ PAD[3] | SERCOM3/ PAD[3] | TC3/WO[1] | TCC0/ WO[3] | I2S/SD[0] | AC/CMP[1] |
| | | 39 | PB16 | VDDIO | EXTINT[0] | | | | | | SERCOM5/ PAD[0] | | TC6/WO[0] | TCC0/ WO[4] | I2S/SD[1] | GCLK_IO[2] |
| | | 40 | PB17 | VDDIO | EXTINT[1] | | | | | | SERCOM5/ PAD[1] | | TC6/WO[1] | TCC0/ WO[5] | I2S/ MCK[0] | GCLK_IO[3] |
| | 29 | 41 | PA20 | VDDIO | EXTINT[4] | | | | X[8] | | SERCOM5/ PAD[2] | SERCOM3/ PAD[2] | TC7/WO[0] | TCC0/ WO[6] | I2S/ SCK[0] | GCLK_IO[4] |
| | 30 | 42 | PA21 | VDDIO | EXTINT[5] | | | | X[9] | | SERCOM5/ PAD[3] | SERCOM3/ PAD[3] | TC7/WO[1] | TCC0/ WO[7] | I2S/FS[0] | GCLK_IO[5] |
| 21 | 31 | 43 | PA22 | VDDIO | EXTINT[6] | | | | X[10] | | SERCOM3/ PAD[0] | SERCOM5/ PAD[0] | TC4/WO[0] | TCC0/ WO[4] | | GCLK_IO[6] |
| 22 | 32 | 44 | PA23 | VDDIO | EXTINT[7] | | | | X[11] | | SERCOM3/ PAD[1] | SERCOM5/ PAD[1] | TC4/WO[1] | TCC0/ WO[5] | USB/SOF 1kHz | GCLK_IO[7] |
| 23 | 33 | 45 | PA24[6] | VDDIO | EXTINT[12] | | | | | | SERCOM3/ PAD[2] | SERCOM5/ PAD[2] | TC5/WO[0] | TCC1/ WO[2] | USB/DM | |
| 24 | 34 | 46 | PA25[6] | VDDIO | EXTINT[13] | | | | | | SERCOM3/ PAD[3] | SERCOM5/ PAD[3] | TC5/WO[1] | TCC1/ WO[3] | USB/DP | |
| | 37 | 49 | PB22 | VDDIO | EXTINT[6] | | | | | | | SERCOM5/ PAD[2] | TC7/WO[0] | | | GCLK_IO[0] |
| | 38 | 50 | PB23 | VDDIO | EXTINT[7] | | | | | | | SERCOM5/ PAD[3] | TC7/WO[1] | | | GCLK_IO[1] |
| 25 | 39 | 51 | PA27 | VDDIO | EXTINT[15] | | | | | | | | | | | GCLK_IO[0] |
| 27 | 41 | 53 | PA28 | VDDIO | EXTINT[8] | | | | | | | | | | | GCLK_IO[0] |
| 31 | 45 | 57 | PA30 | VDDIO | EXTINT[10] | | | | | | | SERCOM1/ PAD[2] | TCC1/WO[0] | | SWCLK | GCLK_IO[0] |
| 32 | 46 | 58 | PA31 | VDDIO | EXTINT[11] | | | | | | | SERCOM1/ PAD[3] | TCC1/WO[1] | | SWDIO[5] | |
| | | 59 | PB30 | VDDIO | EXTINT[14] | | | | | | | SERCOM5/ PAD[0] | TCC0/WO[0] | TCC1/ WO[2] | | |
| | | 60 | PB31 | VDDIO | EXTINT[15] | | | | | | | SERCOM5/ PAD[1] | TCC0/WO[1] | TCC1/ WO[3] | | |
| | | 61 | PB00 | VDDANA | EXTINT[0] | | AIN[8] | | Y[6] | | | SERCOM5/ PAD[2] | TC7/WO[0] | | | |
| | | 62 | PB01 | VDDANA | EXTINT[1] | | AIN[9] | | Y[7] | | | SERCOM5/ PAD[3] | TC7/WO[1] | | | |
| | 47 | 63 | PB02 | VDDANA | EXTINT[2] | | AIN[10] | | Y[8] | | | SERCOM5/ PAD[0] | TC6/WO[0] | | | |
| | 48 | 64 | PB03 | VDDANA | EXTINT[3] | | AIN[11] | | Y[9] | | | SERCOM5/ PAD[1] | TC6/WO[1] | | | |

1. Use the SAMD21J pinout muxing for WLCSP45 package.

2. All analog pin functions are on peripheral function B. Peripheral function B must be selected to disable the digital control of the pin.
3. Only some pins can be used in SERCOM I2C mode. Refer to 7.2.3 SERCOM I2C Pins.
4. Note that TC6 and TC7 are not supported on the SAM D21E and G devices. Refer to 2. Configuration Summary for details.
5. This function is only activated in the presence of a debugger.
6. If the PA24 and PA25 pins are not connected, it is recommended to enable a pull-up on PA24 and PA25 through input GPIO mode. The aim is to avoid an eventually extract power consumption (<1mA) due to a not stable level on pad. The port PA24 and PA25 doesn't have Drive Strength option.

**Table 7-2. PORT Function Multiplexing for L Variant Devices**

| Pin | | I/O Pin | Supply | A | B(1)(2) | | | | | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SAMD21ExL | SAMD21GxL | | | EIC | REF | ADC | AC | AC1 | DAC | SERCOM(1)(2) | SERCOM-ALT(7) | TC(3) /TCC | TCC | COM | AC/ GCLK |
| 1 | 1 | PA02 | VDDANA | EXTINT[2] | | AIN[0] | | | VOUT | | | | | | |
| 2 | 2 | PA03 | VDDANA | EXTINT[3] | DAC/VREFA | AIN[1] | | | | | | | | | |
| 3 | 3 | PB04 | VDDANA | EXTINT[4] | | AIN[12] | | AIN[0] | | | | | | | |
| 4 | 4 | PB05 | VDDANA | EXTINT[5] | | AIN[13] | | AIN[1] | | | | | | | |
| | 7 | PB08 | VDDANA | EXTINT[8] | | AIN[2] | | | | | SERCOM4/ PAD[0] | TC4/WO[0] | | | |
| | 8 | PB09 | VDDANA | EXTINT[9] | | AIN[3] | | | | | SERCOM4/ PAD[1] | TC4/WO[1] | | | |
| 5 | 9 | PA04 | VDDANA | EXTINT[4] | ADC/VREFB | AIN[4] | AIN[0] | | | | SERCOM0/ PAD[0] | TCC0/WO[0] | | | |
| 6 | 10 | PA05 | VDDANA | EXTINT[5] | | AIN[5] | AIN[1] | | | | SERCOM0/ PAD[1] | TCC0/WO[1] | | | |
| 7 | 11 | PA06 | VDDANA | EXTINT[6] | | AIN[6] | AIN[2] | | | | SERCOM0/ PAD[2] | TCC1/WO[0] | | | |
| 8 | 12 | PA07 | VDDANA | EXTINT[7] | | AIN[7] | AIN[3] | | | | SERCOM0/ PAD[3] | TCC1/WO[1] | | | |
| 11 | 13 | PA08 | VDDIO | NMI | | AIN[16] | | | | SERCOM0/ PAD[0] | SERCOM2/ PAD[0] | TCC0/WO[0] | TCC1/ WO[2] | | |
| 12 | 14 | PA09 | VDDIO | EXTINT[9] | | AIN[17] | | | | SERCOM0/ PAD[1] | SERCOM2/ PAD[1] | TCC0/WO[1] | TCC1/ WO[3] | | |
| 13 | 15 | PA10 | VDDIO | EXTINT[10] | | AIN[18] | | | | SERCOM0/ PAD[2] | SERCOM2/ PAD[2] | TCC1/WO[0] | TCC0/ WO[2] | | GCLK_IO[4] |
| 14 | 16 | PA11 | VDDIO | EXTINT[11] | | AIN[19] | | | | SERCOM0/ PAD[3] | SERCOM2/ PAD[3] | TCC1/WO[1] | TCC0/ WO[3] | | GCLK_IO[5] |
| | 19 | PB10 | VDDIO | EXTINT[10] | | | | | | | SERCOM4/ PAD[2] | TC5/WO[0] | TCC0/ WO[4] | | GCLK_IO[4] |
| | 20 | PB11 | VDDIO | EXTINT[11] | | | | | | | SERCOM4/ PAD[3] | TC5/WO[1] | TCC0/ WO[5] | | GCLK_IO[5] |
| | 21 | PA12 | VDDIO | EXTINT[12] | | | | | | SERCOM2/ PAD[0] | SERCOM4/ PAD[0] | TCC2/WO[0] | TCC0/ WO[6] | | AC/CMP[0] |
| | 22 | PA13 | VDDIO | EXTINT[13] | | | | | | SERCOM2/ PAD[1] | SERCOM4/ PAD[1] | TCC2/WO[1] | TCC0/ WO[7] | | AC/CMP[1] |
| 15 | 23 | PA14 | VDDIO | EXTINT[14] | | | | | | SERCOM2/ PAD[2] | SERCOM4/ PAD[2] | TC3/WO[0] | TCC0/ WO[4] | | GCLK_IO[0] |
| 16 | 24 | PA15 | VDDIO | EXTINT[15] | | | | | | SERCOM2/ PAD[3] | SERCOM4/ PAD[3] | TC3/WO[1] | TCC0/ WO[5] | | GCLK_IO[1] |
| 17 | 25 | PA16 | VDDIO | EXTINT[0] | | | | | | SERCOM1/ PAD[0] | SERCOM3/ PAD[0] | TCC2/WO[0] | TCC0/WO[6] | | GCLK_IO[2] |
| 18 | 26 | PA17 | VDDIO | EXTINT[1] | | | | | | SERCOM1/ PAD[1] | SERCOM3/ PAD[1] | TCC2/WO[1] | TCC0/WO[7] | | GCLK_IO[3] |
| 19 | 27 | PA18 | VDDIO | EXTINT[2] | | | | | | SERCOM1/ PAD[2] | SERCOM3/ PAD[2] | TC3/WO[0] | TCC0/ WO[2] | | AC/CMP[0] |
| 20 | 28 | PA19 | VDDIO | EXTINT[3] | | | | | | SERCOM1/ PAD[3] | SERCOM3/ PAD[3] | TC3/WO[1] | TCC0/ WO[3] | | AC/CMP[1] |

| Pin | | I/O Pin | Supply | A | B[1][2] | | | | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SAMD21ExL | SAMD21GxL | | | EIC | REF | ADC | AC | AC1 | DAC | SERCOM[1][2] | SERCOM-ALT[7] | TC[3] /TCC | TCC | COM | AC/ GCLK |
| | 29 | PA20 | VDDIO | EXTINT[4] | | | | | | SERCOM5/ PAD[2] | SERCOM3/ PAD[2] | TC7/WO[0] | TCC0/ WO[6] | | GCLK_IO[4] |
| | 30 | PA21 | VDDIO | EXTINT[5] | | | | | | SERCOM5/ PAD[3] | SERCOM3/ PAD[3] | TC7/WO[1] | TCC0/ WO[7] | | GCLK_IO[5] |
| 21 | 31 | PA22 | VDDIO | EXTINT[6] | | | | | | SERCOM3/ PAD[0] | SERCOM5/ PAD[0] | TC4/WO[0] | TCC0/ WO[4] | | GCLK_IO[6] |
| 22 | 32 | PA23 | VDDIO | EXTINT[7] | | | | | | SERCOM3/ PAD[1] | SERCOM5/ PAD[1] | TC4/WO[1] | TCC0/ WO[5] | | GCLK_IO[7] |
| 23 | 33 | PA24[5] | VDDIO | EXTINT[12] | | | | | | SERCOM3/ PAD[2] | SERCOM5/ PAD[2] | TC5/WO[0] | TCC1/ WO[2] | | AC1/CMP[0] |
| 24 | 34 | PA25[5] | VDDIO | EXTINT[13] | | | | | | SERCOM3/ PAD[3] | SERCOM5/ PAD[3] | TC5/WO[1] | TCC1/ WO[3] | | AC1/CMP[1] |
| | 37 | PA27 | VDDIO | EXTINT[15] | | | | | | | | | | | GCLK_IO[0] |
| | 39 | PA28 | VDDIO | EXTINT[8] | | | | | | | | | | | GCLK_IO[0] |
| 29 | 43 | PA30 | VDDIO | EXTINT[10] | | | | | | | SERCOM1/ PAD[2] | TCC1/WO[0] | | SWCLK | GCLK_IO[0] |
| 30 | 44 | PA31 | VDDIO | EXTINT[11] | | | | | | | SERCOM1/ PAD[3] | TCC1/WO[1] | | SWDIO[5] | |
| | 45 | PB00 | | | | AIN[8] | | | | | | | | | |
| | 46 | PB01 | | | | AIN[9] | | | | | | | | | |
| 31 | 47 | PB02 | VDDANA | EXTINT[2] | | AIN[10] | | AIN[2] | | | SERCOM5/ PAD[0] | TC6/WO[0][6] | | | |
| 32 | 48 | PB03 | VDDANA | EXTINT[3] | | AIN[11] | | AIN[3] | | | SERCOM5/ PAD[1] | TC6/WO[1][6] | | | |

1. All analog pin functions are on peripheral function B. Peripheral function B must be selected to disable the digital control of the pin.
2. Only some pins can be used in SERCOM $I^2C$ mode.
3. Note that TC6 and TC7 are not supported on the SAM D21ExL and SAM D21GxL devices. Refer to 2. Configuration Summary for details.
4. This function is only activated in the presence of a debugger.
5. If the PA24 and PA25 pins are not connected, it is recommended to enable a pull-up on PA24 and PA25 through input GPIO mode. The aim is to avoid an eventually extract power consumption (<1mA) due to a not stable level on pad. The port PA24 and PA25 doesn't have Drive Strength option.
6. TC6 is only available on the SAM D21GxL and not the SAM D21ExL.
7. SERCOM4 and SERCOM5 are not supported on the SAM D21E devices. Refer to 2. Configuration Summary for details.

**Related Links**

37. Electrical Characteristics

## 7.2 Other Functions

### 7.2.1 Oscillator Pinout

The oscillators are not mapped to the normal PORT functions and their multiplexing are controlled by registers in the System Controller (SYSCTRL).

**Table 7-3. Oscillator Pinout**

| Oscillator | Supply | Signal | I/O pin |
|---|---|---|---|
| XOSC | VDDIO | XIN | PA14 |
| | | XOUT | PA15 |
| XOSC32K | VDDANA | XIN32 | PA00 |
| | | XOUT32 | PA01 |

### 7.2.2 Serial Wire Debug Interface Pinout

Only the SWCLK pin is mapped to the normal PORT functions. A debugger cold-plugging or hot-plugging detection will automatically switch the SWDIO port to the SWDIO function.

**Table 7-4. Serial Wire Debug Interface Pinout**

| Signal | Supply | I/O pin |
|---|---|---|
| SWCLK | VDDIO | PA30 |
| SWDIO | VDDIO | PA31 |

### 7.2.3 SERCOM $I^2C$ Pins

**Table 7-5. SERCOM Pins Supporting $I^2C$**

| Device | Pins Supporting $I^2C$ Hs mode |
|---|---|
| SAM D21E | PA08, PA09, PA16, PA17, PA22, PA23 |
| SAM D21G | PA08, PA09, PA12, PA13, PA16, PA17, PA22, PA23 |
| SAM D21J | PA08, PA09, PA12, PA13, PA16, PA17, PA22, PA23, PB12, PB13, PB16, PB17, PB30, PB31 |

### 7.2.4 GPIO Clusters

**Table 7-6. GPIO Clusters**

| PACKAGE | CLUSTER | GPIO | | | | | | | | | | | | | | | | SUPPLIES PINS CONNECTED TO THE CLUSTER |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 64pins | 1 | PB31 | PB30 | PA31 | PA30 | | | | | | | | | | | | | VDDIN pin56/GND pin54 |
| | 2 | PA28 | PA27 | PB23 | PB22 | | | | | | | | | | | | | VDDIN pin56/GND pin54 and VDDIO pin 48/GND pin47 |
| | 3 | PA25 | PA24 | PA23 | PA22 | PA21 | PA20 | PB17 | PB16 | PA19 | PA18 | PA17 | PA16 | | | | | VDDIO pin 48/GND pin47 and VDDIO pin34/GND pin33 |
| | 4 | PA15 | PA14 | PA13 | PA12 | PB15 | PB14 | PB13 | PB12 | PB11 | PB10 | | | | | | | VDDIO pin 34/GND pin33 |

| PACKAGE | CLUSTER | GPIO | | | | | | | | | | | | | | | | SUPPLIES PINS CONNECTED TO THE CLUSTER |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | | and VDDIO pin21/GND pin22 |
| | 5 | PA11 | PA10 | PA09 | PA08 | | | | | | | | | | | | | VDDIO pin21/GND pin22 |
| | 6 | PA07 | PA06 | PA05 | PA04 | PB09 | PB08 | PB07 | PB06 | | | | | | | | | VDDANA pin 8/GNDANA pin7 |
| | 7 | PB05 | PB04 | PA03 | PA02 | PA01 | PA00 | PB03 | PB02 | PB01 | PB00 | | | | | | | VDDANA pin 8/GNDANA pin7 |
| 48pins | 1 | PA31 | PA30 | | | | | | | | | | | | | | | VDDIN pin44/GND pin42 |
| | 2 | PA28 | PA27 | PB23 | PB22 | | | | | | | | | | | | | VDDIN pin44/GND pin42 and VDDIO pin36/GND pin35 |
| | 3 | PA25 | PA24 | PA23 | PA22 | PA21 | PA20 | PA19 | PA18 | PA17 | PA16 | PA15 | PA14 | PA13 | PA12 | PB11 | PB10 | VDDIO pin36/GND pin35 and VDDIO pin17/GND pin18 |
| | 4 | PA11 | PA10 | PA09 | PA08 | | | | | | | | | | | | | VDDIO pin17/GND pin18 |
| | 5 | PA07 | PA06 | PA05 | PA04 | PB09 | PB08 | | | | | | | | | | | VDDANA pin6/ GNDANA pin5 |
| | 6 | PA03 | PA02 | PA01 | PA00 | PB03 | PB02 | PB05 | PB04 | PB01 | PB00 | | | | | | | VDDANA pin6/ GNDANA pin5 |
| 32pins | 1 | PA31 | PA30 | | | | | | | | | | | | | | | VDDIN pin30/GND pin 28 |
| | 2 | PA28 | PA27 | PA25 | PA24 | PA23 | PA22 | PA19 | PA18 | PA17 | PA16 | PA15 | PA14 | PA11 | PA10 | PA09 | PA08 | VDDIN pin30/GND pin 28 and VDDANA pin9/GND pin10 |
| | 3 | PA07 | PA06 | PA05 | PA04 | PA03 | PA02 | PA01 | PA00 | PB05 | PB04 | PB03 | PB02 | | | | | VDDANA pin9/GND pin10 |

### 7.2.5 TCC Configurations

The SAM D21 has three instances of the Timer/Counter for Control applications (TCC) peripheral, , TCC[2:0]. The following table lists the features for each TCC instance.

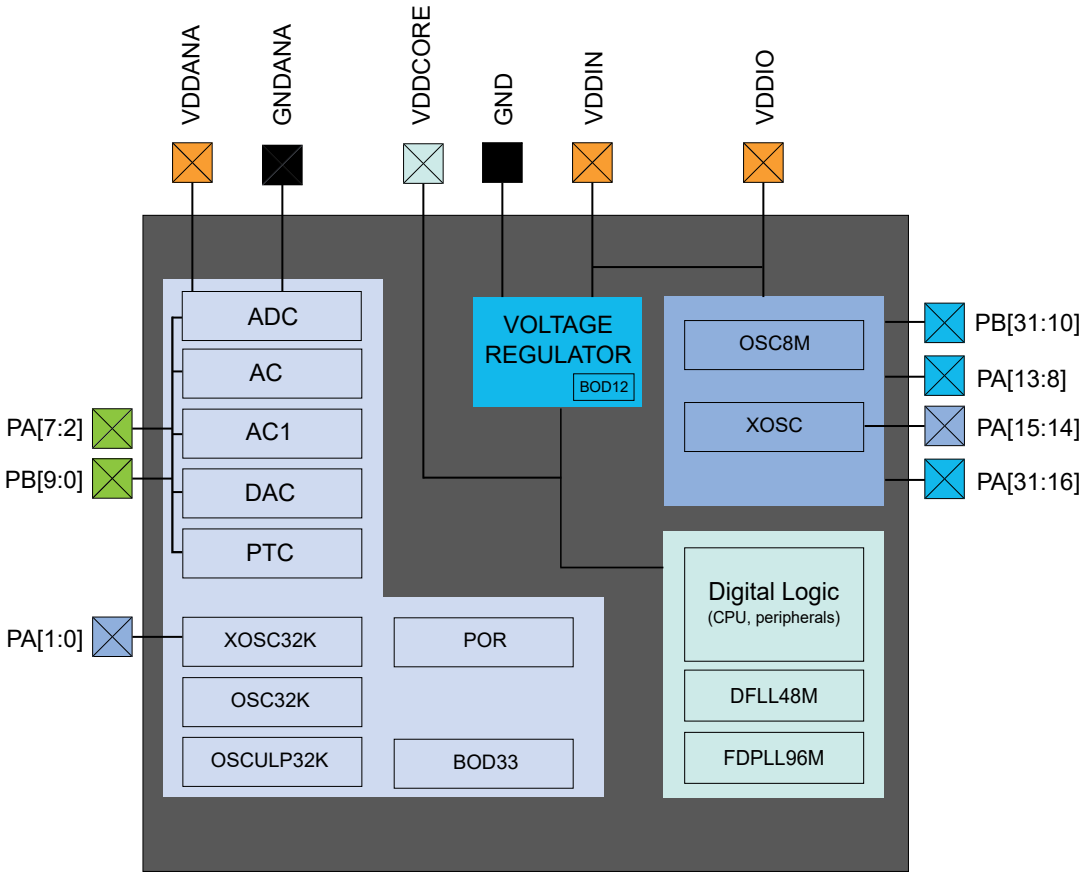
**Table 7-7. TCC Configuration Summary**

| TCC# | Channels (CC_NUM) | Waveform Output (WO_NUM) | Counter size | Fault | Dithering | Output matrix | Dead Time Insertion (DTI) | SWAP | Pattern generation |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 4 | 8 | 24-bit | Yes | Yes | Yes | Yes | Yes | Yes |
| 1 | 2 | 4 | 24-bit | Yes | Yes | | | | Yes |
| 2 | 2 | 2 | 16-bit | Yes | | | | | |

**Note:** The number of CC registers (CC_NUM) for each TCC corresponds to the number of compare/capture channels, so that a TCC can have more Waveform Outputs (WO_NUM) than CC registers.

# 8. Power Supply and Start-Up Considerations

## 8.1 Power Domain Overview




## 8.2 Power Supply Considerations

### 8.2.1 Power Supplies

The device has several different power supply pins:

- VDDIO: Powers I/O lines, OSC8M and XOSC. Voltage is 1.62V to 3.63V.
- VDDIN: Powers I/O lines and the internal regulator. Voltage is 1.62V to 3.63V.
- VDDANA: Powers I/O lines and the ADC, AC, DAC, PTC, OSCULP32K, OSC32K, XOSC32K. Voltage is 1.62V to 3.63V.
- VDDCORE: Internal regulated voltage output. Powers the core, memories, peripherals, FDPLL96M, and DFLL48M. Voltage is 1.2V.

The same voltage must be applied to both VDDIN, VDDIO and VDDANA. This common voltage is referred to as $V_{DD}$ in the datasheet.

The ground pins, GND, are common to VDDCORE, VDDIO and VDDIN. The ground pin for VDDANA is GNDANA.

For decoupling recommendations for the different power supplies. Refer to *Schematic Checklist* for details.

**Related Links**

41. Schematic Checklist

### 8.2.2 Voltage Regulator

The voltage regulator has two different modes:

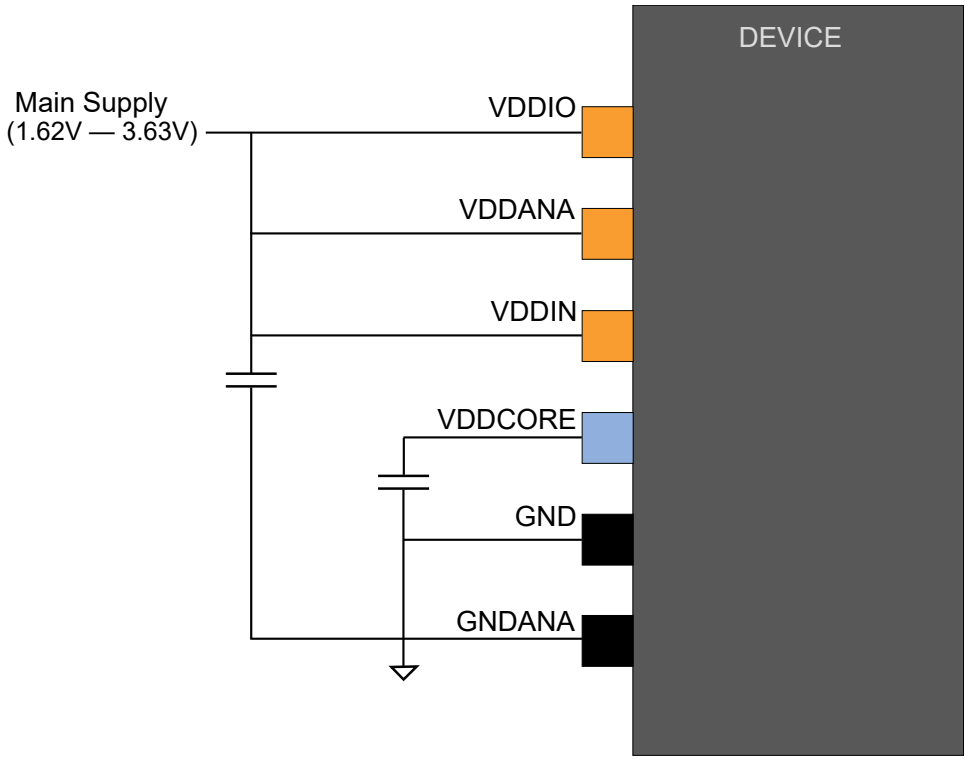
- Normal mode: To be used when the CPU and peripherals are running
- Low Power (LP) mode: To be used when the regulator draws small static current. It can be used in standby mode

### 8.2.3 Typical Powering Schematics

The device uses a single main supply with a range of 1.62V - 3.63V.

The following figure shows the recommended power supply connection.

**Figure 8-1. Power Supply Connection**




### 8.2.4 Power-Up Sequence

#### 8.2.4.1 Minimum Rise Rate

The integrated Power-on Reset (POR) circuitry monitoring the VDDANA power supply requires a minimum rise rate. Refer to the *Electrical Characteristics* for details.

**Related Links**

37. Electrical Characteristics

#### 8.2.4.2 Maximum Rise Rate

The rise rate of the power supply must not exceed the values described in Electrical Characteristics. Refer to the *Electrical Characteristics* for details.

**Related Links**

37. Electrical Characteristics

## 8.3 Power-Up

This section summarizes the power-up sequence of the device. The behavior after power-up is controlled by the Power Manager. Refer to *PM – Power Manager* for details.

**Related Links**

16. PM – Power Manager

### 8.3.1 Starting of Clocks

After power-up, the device is set to its initial state and kept in reset, until the power has stabilized throughout the device. Once the power has stabilized, the device will use a 1MHz clock. This clock is derived from the 8MHz Internal Oscillator (OSC8M), which is divided by eight and used as a clock source for generic clock generator 0. Generic clock generator 0 is the main clock for the Power Manager (PM).

Some synchronous system clocks are active, allowing software execution.

Refer to the “Clock Mask Register” section in *PM – Power Manager* for the list of default peripheral clocks running. Synchronous system clocks that are running are by default not divided and receive a 1MHz clock through generic clock generator 0. Other generic clocks are disabled except GCLK_WDT, which is used by the Watchdog Timer (WDT).

**Related Links**

16. PM – Power Manager

### 8.3.2 I/O Pins

After power-up, the I/O pins are tri-stated.

### 8.3.3 Fetching of Initial Instructions

After reset has been released, the CPU starts fetching PC and SP values from the reset address, which is 0x00000000. This address points to the first executable address in the internal Flash. The code read from the Internal Flash is free to configure the clock system and clock sources. Refer to *PM – Power Manager*, *GCLK – Generic Clock Controller* and *SYSCTRL – System Controller* for details. Refer to the ARM Architecture Reference Manual for more information on CPU startup (http://www.arm.com).

**Related Links**

16. PM – Power Manager
17. SYSCTRL – System Controller
14. Clock System

## 8.4 Power-On Reset and Brown-Out Detector

The SAM D21 embeds three features to monitor, warn and/or reset the device:

- POR: Power-On Reset on VDDANA

- BOD33: Brown-Out Detector on VDDANA
- BOD12: Voltage Regulator Internal Brown-Out Detector on VDDCORE. The Voltage Regulator Internal BOD is calibrated in production and its calibration configuration is stored in the NVM User Row. This configuration should not be changed if the user row is written to assure the correct behavior of the BOD12.

### 8.4.1 Power-On Reset on VDDANA

POR monitors VDDANA. It is always activated and monitors voltage at startup and also during all the sleep modes. If VDDANA goes below the threshold voltage, the entire chip is reset.

### 8.4.2 Brown-Out Detector on VDDANA

BOD33 monitors VDDANA. Refer to *SYSCTRL – System Controller* for details.

**Related Links**

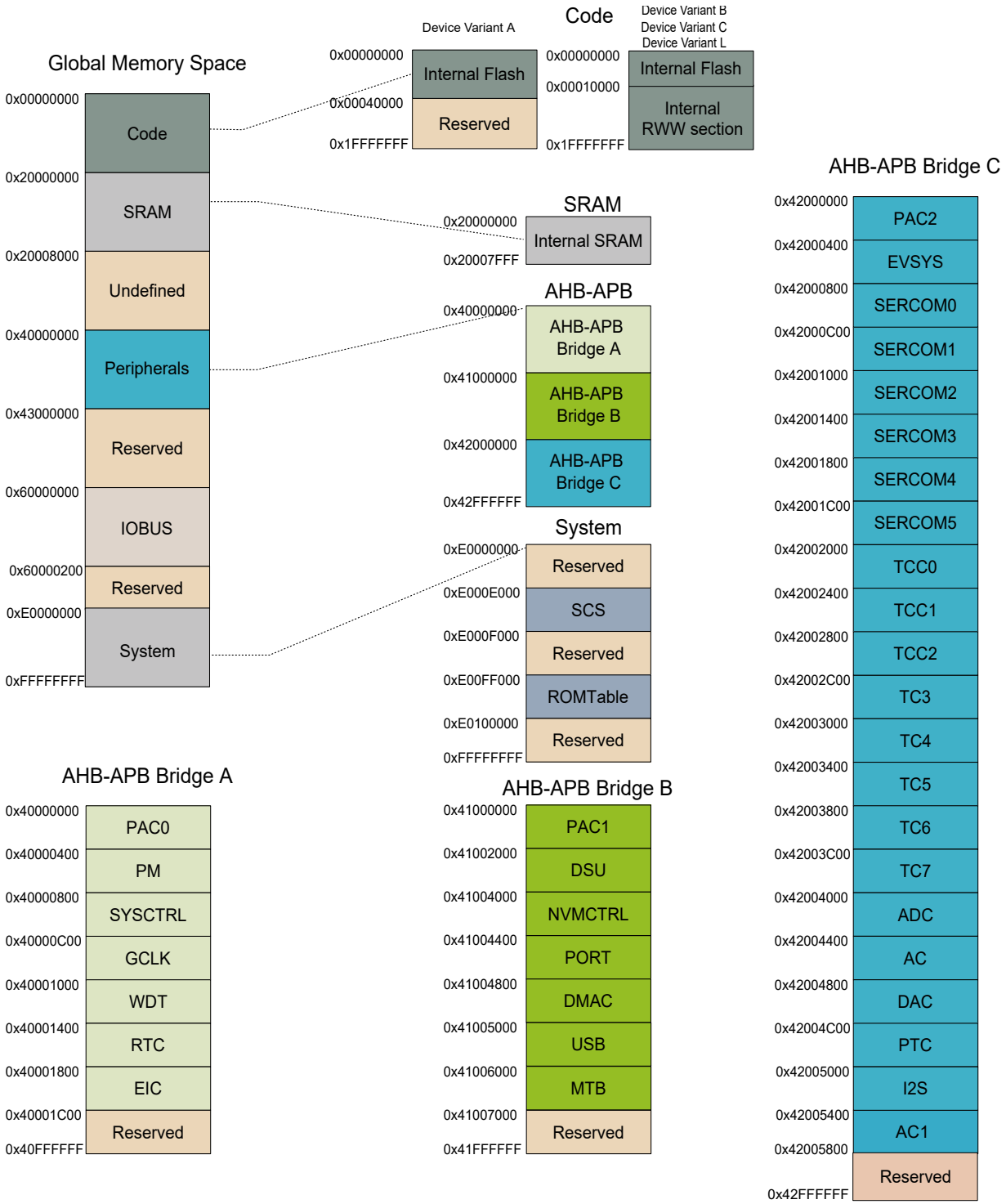
17. SYSCTRL – System Controller

### 8.4.3 Brown-Out Detector on VDDCORE

Once the device has started up, BOD12 monitors the internal VDDCORE.

# 9. Product Mapping

**Figure 9-1. SAM D21 Product Mapping**



This figure represents the full configuration of the SAM D21 with maximum Flash and SRAM capabilities and a full set of peripherals. Refer to the 2. Configuration Summary for details.

# 10. Memories

## 10.1 Embedded Memories

- Internal high-speed flash with Read-While-Write (RWW) capability on section of the array (Device Variant B, C and L).
- Internal high-speed RAM, single-cycle access at full speed

## 10.2 Physical Memory Map

The High-Speed bus is implemented as a bus matrix. All High-Speed bus addresses are fixed, and they are never remapped in any way, even during boot. The 32-bit physical address space is mapped as follow:

**Table 10-1. SAM D21 Physical Memory Map(1)**

| Memory | Start address | Size | | | | | |
|---|---|---|---|---|---|---|---|
| | | SAMD21x18 | SAMD21x17 | SAMD21x16 | SAMD21x15 | SAMD21x16L | SAMD21x15L |
| Internal Flash | 0x00000000 | 256 Kbytes | 128 Kbytes | 64 Kbytes | 32 Kbytes | 64 Kbytes | 32 Kbytes |
| Internal RWW section(2) | 0x00400000 | - | - | 2 Kbytes | 1 Kbytes | 2 Kbytes | 1 Kbytes |
| Internal SRAM | 0x20000000 | 32 Kbytes | 16 Kbytes | 8 Kbytes | 4 Kbytes | 8 Kbytes | 4 Kbytes |
| Peripheral Bridge A | 0x40000000 | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes |
| Peripheral Bridge B | 0x41000000 | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes |
| Peripheral Bridge C | 0x42000000 | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes | 64 Kbytes |
| IOBUS | 0x60000000 | 0.5 Kbytes | 0.5 Kbytes | 0.5 Kbytes | 0.5 Kbytes | 0.5 Kbytes | 0.5 Kbytes |

**Notes:**

1. x = G, J or E.

2. Only applicable for device variants B, C, and L.

**Table 10-2. Flash Memory Parameters(1,2)**

| Device | Flash size | Number of pages | Page size |
|---|---|---|---|
| SAMD21x18 | 256 Kbytes | 4096 | 64 bytes |
| SAMD21x17 | 128 Kbytes | 2048 | 64 bytes |
| SAMD21x16 | 64 Kbytes | 1024 | 64 bytes |
| SAMD21x15 | 32 Kbytes | 512 | 64 bytes |

**Notes:**

1. x = G, J or E.

2. The number of pages (NVMP) and page size (PSZ) can be read from the NVM Pages and Page Size bits in the NVM Parameter register in the NVMCTRL (PARAM.NVMP and PARAM.PSZ, respectively). Refer to *NVM Parameter* (PARAM) register for details.

**Table 10-3. RWW Section Parameters (Device Variants B, C, and L)**

| Device[1] | Flash size | Number of pages | Page size |
|---|---|---|---|
| SAMD21x16 | 2 Kbytes | 32 | 64 bytes |
| SAMD21x15 | 1 Kbytes | 16 | 64 bytes |

**Note:**

1. x = G, J, or E.

**Related Links**

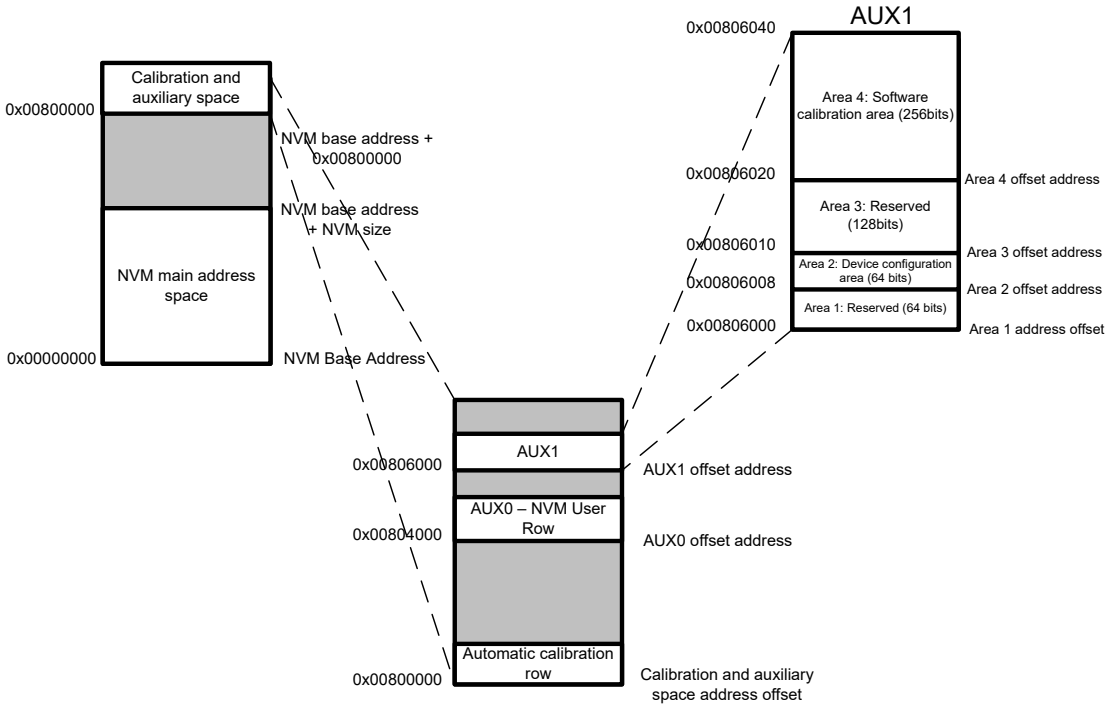
22.8.3 PARAM

3. Ordering Information(1)

## 10.3 NVM Calibration and Auxiliary Space

The device calibration data are stored in different sections of the NVM calibration and auxiliary space presented in the following figure.

**Figure 10-1. Calibration and Auxiliary Space**




The values from the automatic calibration row are loaded into their respective registers at startup.

### 10.3.1 NVM User Row Mapping

The first two 32-bit words of the NVM User Row contain calibration data that are automatically read at device power on.

The NVM User Row can be read at address 0x804000.

To write the NVM User Row refer to *NVMCTRL – Non-Volatile Memory Controller*.

Note that when writing to the user row the values do not get loaded by the other modules on the device until a device reset occurs.

**Table 10-4. NVM User Row Mapping**

| Bit Position | Name | Usage |
|---|---|---|
| 2:0 | BOOTPROT | Used to select one of eight different bootloader sizes. Refer to *NVMCTRL – Non-Volatile Memory Controller*. Default value = 7 except for WLCSP (Default value = 3). |
| 3 | Reserved | |
| 6:4 | EEPROM | Used to select one of eight different EEPROM sizes. Refer to *NVMCTRL – Non-Volatile Memory Controller*. Default value = 7. |
| 7 | Reserved | |
| 13:8 | BOD33 Level | BOD33 Threshold Level at power on. Refer to *SYSCTRL BOD33 register*.<br>Default value = 7. |
| 14 | BOD33 Enable | BOD33 Enable at power on . Refer to *SYSCTRL BOD33 register*.<br>Default value = 1. |
| 16:15 | BOD33 Action | BOD33 Action at power on. Refer to *SYSCTRL BOD33 register*.<br>Default value = 1. |
| 24:17 | Reserved | Voltage Regulator Internal BOD (BOD12) configuration. These bits are written in production and must not be changed. Default value = 0x70. |
| 25 | WDT Enable | WDT Enable at power on. Refer to *WDT CTRL register*.<br>Default value = 0. |
| 26 | WDT Always-On | WDT Always-On at power on. Refer to *WDT CTRL register*.<br>Default value = 0. |
| 30:27 | WDT Period | WDT Period at power on. Refer to *WDT CONFIG register*.<br>Default value = 0x0B. |
| 34:31 | WDT Window | WDT Window mode time-out at power on. Refer to *WDT CONFIG register*.<br>Default value = 0x05. |
| 38:35 | WDT EWOFFSET | WDT Early Warning Interrupt Time Offset at power on. Refer to *WDT EWCTRL register*. Default value = 0x0B. |
| 39 | WDT WEN | WDT Timer Window Mode Enable at power on. Refer to *WDT CTRL register*. Default value = 0. |
| 40 | BOD33 Hysteresis | BOD33 Hysteresis configuration at power on. Refer to *SYSCTRL BOD33 register*.<br>Default value = 0. |
| 41 | Reserved | Voltage Regulator Internal BOD(BOD12) configuration. This bit is written in production and must not be changed. Default value = 0. |

| Bit Position | Name | Usage |
|---|---|---|
| 47:42 | Reserved | |
| 63:48 | LOCK | NVM Region Lock Bits. Refer to *NVMCTRL – Non-Volatile Memory Controller*.<br>Default value = 0xFFFF. |

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

17.8.14 BOD33

18.8.1 CTRL

### 10.3.2 NVM Software Calibration Area Mapping

The NVM Software Calibration Area contains calibration data that are measured and written during production test. These calibration values should be read by the application software and written back to the corresponding register.

The NVM Software Calibration Area can be read at address 0x806020.

The NVM Software Calibration Area can not be written.

**Table 10-5. NVM Software Calibration Area Mapping**

| Bit Position | Name | Description |
|---|---|---|
| 2:0 | Reserved | |
| 14:3 | Reserved | |
| 26:15 | Reserved | |
| 34:27 | ADC LINEARITY | ADC Linearity Calibration. Should be written to ADC CALIB register. |
| 37:35 | ADC BIASCAL | ADC Bias Calibration. Should be written to ADC CALIB register. |
| 44:38 | OSC32K CAL | OSC32KCalibration. Should be written to SYSCTRL OSC32K register. |
| 49:45 | USB TRANSN | USB TRANSN calibration value. Should be written to USB PADCAL register. |
| 54:50 | USB TRANSP | USB TRANSP calibration value. Should be written to USB PADCAL register. |
| 57:55 | USB TRIM | USB TRIM calibration value. Should be written to the USB PADCAL register. |
| 63:58 | DFLL48M COARSE CAL | DFLL48M Coarse calibration value. Should be written to SYSCTRL DFLLVAL register. |
| 73:64 | Reserved | |
| 127:74 | Reserved | |

### 10.3.3 Serial Number

Each device has a unique 128-bit serial number which is a concatenation of four 32-bit words contained at the following addresses:

Word 0: 0x0080A00C

Word 1: 0x0080A040

Word 2: 0x0080A044

Word 3: 0x0080A048

The uniqueness of the serial number is guaranteed only when using all 128 bits.

# 11. Processor And Architecture

## 11.1 Cortex M0+ Processor

The SAM D21 implements the ARM® Cortex®-M0+ processor, based on the ARMv6 Architecture and Thumb®-2 ISA. The Cortex M0+ is 100% instruction set compatible with its predecessor, the Cortex-M0 core, and upward compatible to Cortex-M3 and M4 cores. The ARM Cortex-M0+ implemented is revision r0p1. For more information refer to http://www.arm.com.

### 11.1.1 Cortex M0+ Configuration

**Table 11-1. Cortex M0+ Configuration**

| Features | Configurable option | Device configuration |
|---|---|---|
| Interrupts | External interrupts 0-32 | 28 |
| Data endianness | Little-endian or big-endian | Little-endian |
| SysTick timer | Present or absent | Present |
| Number of watchpoint comparators | 0, 1, 2 | 2 |
| Number of breakpoint comparators | 0, 1, 2, 3, 4 | 4 |
| Halting debug support | Present or absent | Present |
| Multiplier | Fast or small | Fast (single cycle) |
| Single-cycle I/O port | Present or absent | Present |
| Wake-up interrupt controller | Supported or not supported | Not supported |
| Vector Table Offset Register | Present or absent | Present |
| Unprivileged/Privileged support | Present or absent | Absent(1) |
| Memory Protection Unit | Not present or 8-region | Not present |
| Reset all registers | Present or absent | Absent |
| Instruction fetch width | 16-bit only or mostly 32-bit | 32-bit |

**Note:**

1. All software run in Privileged mode only.

The ARM Cortex-M0+ core has the following two bus interfaces:

- Single 32-bit AMBA-3 AHB-Lite system interface that provides connections to peripherals and all system memory, which includes Flash and RAM.
- Single 32-bit I/O port bus interfacing to the PORT with 1-cycle loads and stores.

### 11.1.2 Cortex-M0+ Peripherals

- System Control Space (SCS)
  - The processor provides debug through registers in the SCS. Refer to the Cortex-M0+ Technical Reference Manual for details (www.arm.com).
- System Timer (SysTick)

  - The System Timer is a 24-bit timer clocked by CLK_CPU that extends the functionality of both the processor and the NVIC. Refer to the Cortex-M0+ Technical Reference Manual for details (www.arm.com).
- Nested Vectored Interrupt Controller (NVIC)
  - External interrupt signals connect to the NVIC, and the NVIC prioritizes the interrupts. Software can set the priority of each interrupt. The NVIC and the Cortex-M0+ processor core are closely coupled, providing low latency interrupt processing and efficient processing of late arriving interrupts. Refer to 11.2 Nested Vector Interrupt Controller and the Cortex-M0+ Technical Reference Manual for details (www.arm.com).
- System Control Block (SCB)
  - The System Control Block provides system implementation information, and system control. This includes configuration, control, and reporting of the system exceptions. Refer to the Cortex-M0+ Devices Generic User Guide for details (www.arm.com).
- Micro Trace Buffer (MTB)
  - The CoreSight MTB-M0+ (MTB) provides a simple execution trace capability to the Cortex-M0+ processor. Refer to section 11.3 Micro Trace Buffer and the CoreSight MTB-M0+ Technical Reference Manual for details (www.arm.com).

### 11.1.3 Cortex-M0+ Address Map

**Table 11-2. Cortex-M0+ Address Map**

| Address | Peripheral |
|---|---|
| 0xE000E000 | System Control Space (SCS) |
| 0xE000E010 | System Timer (SysTick) |
| 0xE000E100 | Nested Vectored Interrupt Controller (NVIC) |
| 0xE000ED00 | System Control Block (SCB) |
| 0x41006000 (see also Product Mapping) | Micro Trace Buffer (MTB) |

### 11.1.4 I/O Interface

#### 11.1.4.1 Overview

Because accesses to the AMBA® AHB-Lite™ and the single cycle I/O interface can be made concurrently, the Cortex-M0+ processor can fetch the next instructions while accessing the I/Os. This enables single cycle I/O accesses to be sustained for as long as needed. Refer to *CPU Local Bus* for more information.

**Related Links**

23.5.10 CPU Local Bus

#### 11.1.4.2 Description

Direct access to PORT registers.

## 11.2 Nested Vector Interrupt Controller

### 11.2.1 Overview

The Nested Vectored Interrupt Controller (NVIC) in the SAM D21 supports 32 interrupt lines with four different priority levels. For more details, refer to the Cortex-M0+ Technical Reference Manual (www.arm.com).

### 11.2.2 Interrupt Line Mapping

Each of the 28 interrupt lines is connected to one peripheral instance, as shown in the table below. Each peripheral can have one or more interrupt flags, located in the peripheral's Interrupt Flag Status and Clear (INTFLAG) register. The interrupt flag is set when the interrupt condition occurs. Each interrupt in the peripheral can be individually enabled by writing a one to the corresponding bit in the peripheral's Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the peripheral's Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated from the peripheral when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt requests for one peripheral are ORed together on system level, generating one interrupt request for each peripheral. An interrupt request will set the corresponding interrupt pending bit in the NVIC interrupt pending registers (SETPEND/CLRPEND bits in ISPR/ICPR). For the NVIC to activate the interrupt, it must be enabled in the NVIC interrupt enable register (SETENA/CLRENA bits in ISER/ICER). The NVIC interrupt priority registers IPR0-IPR7 provide a priority field for each interrupt.

**Table 11-3. Interrupt Line Mapping**

| Peripheral Source | NVIC Line |
|---|---|
| EIC NMI – External Interrupt Controller | NMI |
| PM – Power Manager | 0 |
| SYSCTRL – System Control | 1 |
| WDT – Watchdog Timer | 2 |
| RTC – Real Time Counter | 3 |
| EIC – External Interrupt Controller | 4 |
| NVMCTRL – Non-Volatile Memory Controller | 5 |
| DMAC - Direct Memory Access Controller | 6 |
| USB - Universal Serial Bus | 7 |
| EVSYS – Event System | 8 |
| SERCOM0 – Serial Communication Interface 0 | 9 |
| SERCOM1 – Serial Communication Interface 1 | 10 |
| SERCOM2 – Serial Communication Interface 2 | 11 |
| SERCOM3 – Serial Communication Interface 3 | 12 |
| SERCOM4 – Serial Communication Interface 4 | 13 |
| SERCOM5 – Serial Communication Interface 5 | 14 |
| TCC0 – Timer Counter for Control 0 | 15 |
| TCC1 – Timer Counter for Control 1 | 16 |
| TCC2 – Timer Counter for Control 2 | 17 |
| TC3 – Timer Counter 3 | 18 |
| TC4 – Timer Counter 4 | 19 |
| TC5 – Timer Counter 5 | 20 |

| Peripheral Source | NVIC Line |
|---|---|
| TC6 – Timer Counter 6 | 21 |
| TC7 – Timer Counter 7 | 22 |
| ADC – Analog-to-Digital Converter | 23 |
| AC – Analog Comparator | 24 |
| DAC – Digital-to-Analog Converter | 25 |
| PTC – Peripheral Touch Controller | 26 |
| I2S - Inter IC Sound | 27 |
| AC1 - Analog Comparator 1 | 28 |

## 11.3 Micro Trace Buffer

### 11.3.1 Features

- Program flow tracing for the Cortex-M0+ processor
- MTB SRAM can be used for both trace and general purpose storage by the processor
- The position and size of the trace buffer in SRAM is configurable by software
- CoreSight compliant

### 11.3.2 Overview

When enabled, the MTB records changes in program flow, reported by the Cortex-M0+ processor over the execution trace interface shared between the Cortex-M0+ processor and the CoreSight MTB-M0+. This information is stored as trace packets in the SRAM by the MTB. An off-chip debugger can extract the trace information using the Debug Access Port to read the trace information from the SRAM. The debugger can then reconstruct the program flow from this information.

The MTB simultaneously stores trace information into the SRAM, and gives the processor access to the SRAM. The MTB ensures that trace write accesses have priority over processor accesses.

The execution trace packet consists of a pair of 32-bit words that the MTB generates when it detects the processor PC value changes non-sequentially. A non-sequential PC change can occur during branch instructions or during exception entry. See the CoreSight MTB-M0+ Technical Reference Manual for more details on the MTB execution trace packet format.

Tracing is enabled when the MASTER.EN bit in the Master Trace Control Register is 1. There are various ways to set the bit to 1 to start tracing, or to 0 to stop tracing. See the CoreSight Cortex-M0+ Technical Reference Manual for more details on the Trace start and stop and for a detailed description of the MTB's MASTER register. The MTB can be programmed to stop tracing automatically when the memory fills to a specified watermark level or to start or stop tracing by writing directly to the MASTER.EN bit. If the watermark mechanism is not being used and the trace buffer overflows, then the buffer wraps around overwriting previous trace packets.

The base address of the MTB registers is 0x41006000; this address is also written in the CoreSight ROM Table. The offset of each register from the base address is fixed and as defined by the CoreSight MTB-M0+ Technical Reference Manual. The MTB has 4 programmable registers to control the behavior of the trace features:

- POSITION: Contains the trace write pointer and the wrap bit,
- MASTER: Contains the main trace enable bit and other trace control fields,
- FLOW: Contains the WATERMARK address and the AUTOSTOP and AUTOHALT control bits,
- BASE: Indicates where the SRAM is located in the processor memory map. This register is provided to enable auto discovery of the MTB SRAM location, by a debug agent.

See the CoreSight MTB-M0+ Technical Reference Manual for a detailed description of these registers.

## 11.4 High-Speed Bus System

### 11.4.1 Features

High-Speed Bus Matrix has the following features:

- Symmetric crossbar bus switch implementation
- Allows concurrent accesses from different masters to different slaves
- 32-bit data bus
- Operation at a one-to-one clock frequency with the bus masters

### 11.4.2 Configuration

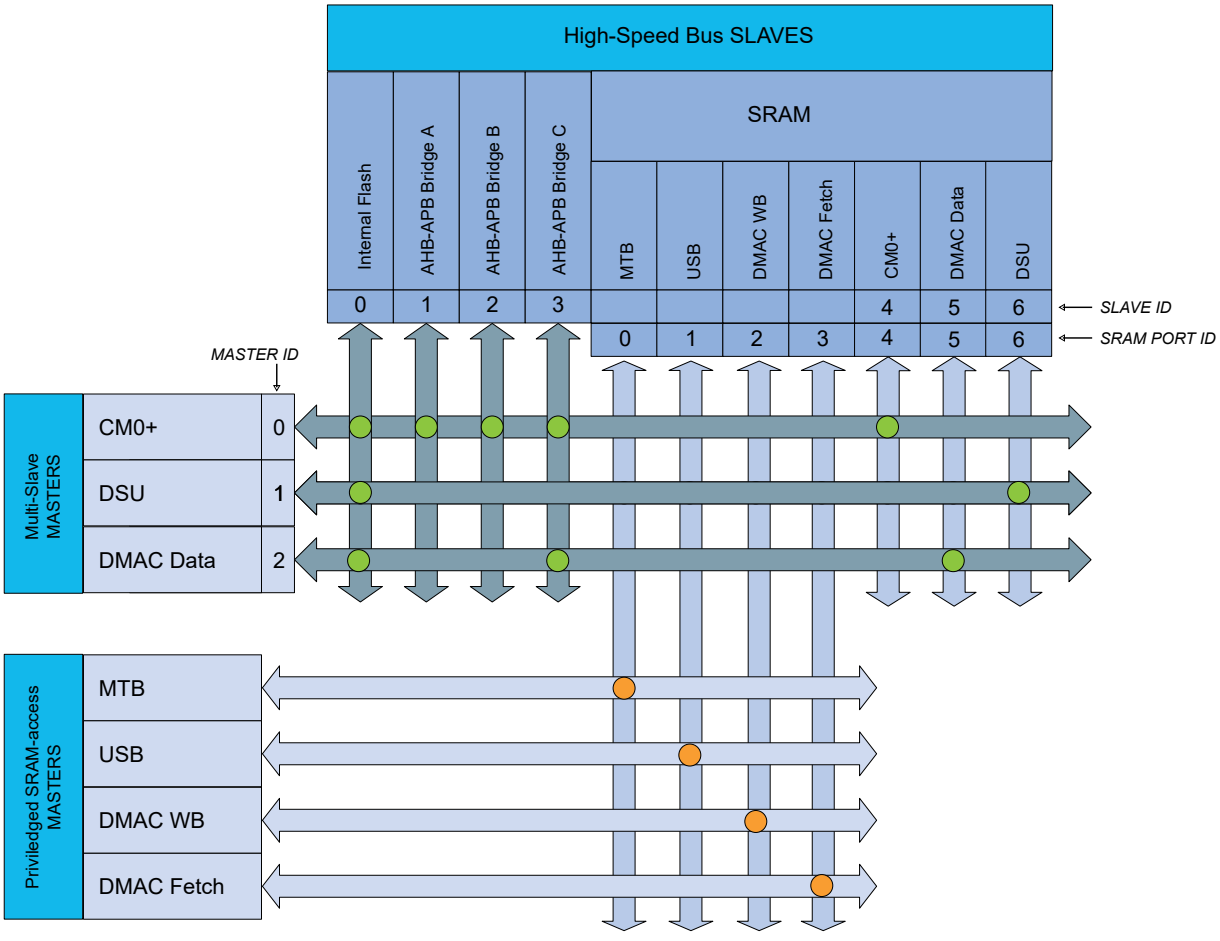


**Table 11-4. Bus Matrix Masters**

| Bus Matrix Masters | Master ID |
|---|---|
| CM0+ - Cortex M0+ Processor | 0 |
| DSU - Device Service Unit | 1 |
| DMAC - Direct Memory Access Controller / Data Access | 2 |

**Table 11-5. Bus Matrix Slaves**

| Bus Matrix Slaves | Slave ID |
|---|---|
| Internal Flash Memory | 0 |
| AHB-APB Bridge A | 1 |
| AHB-APB Bridge B | 2 |
| AHB-APB Bridge C | 3 |
| SRAM Port 4 - CM0+ Access | 4 |
| SRAM Port 5 - DMAC Data Access | 5 |
| SRAM Port 6 - DSU Access | 6 |

**Table 11-6. SRAM Port Connection**

| SRAM Port Connection | Port ID | Connection Type |
|---|---|---|
| MTB - Micro Trace Buffer | 0 | Direct |
| USB - Universal Serial Bus | 1 | Direct |
| DMAC - Direct Memory Access Controller - Write-Back Access | 2 | Direct |
| DMAC - Direct Memory Access Controller - Fetch Access | 3 | Direct |
| CM0+ - Cortex M0+ Processor | 4 | Bus Matrix |
| DMAC - Direct Memory Access Controller - Data Access | 5 | Bus Matrix |
| DSU - Device Service Unit | 6 | Bus Matrix |

### 11.4.3 SRAM Quality of Service

To ensure that masters with latency requirements get sufficient priority when accessing RAM, the different masters can be configured to have a given priority for different type of access.

The Quality of Service (QoS) level is independently selected for each master accessing the RAM. For any access to the RAM the RAM also receives the QoS level. The QoS levels and their corresponding bit values for the QoS level configuration is shown in Table. Quality of Service.

**Table 11-7. Quality of Service**

| Value | Name | Description |
|---|---|---|
| 00 | DISABLE | Background (no sensitive operation) |
| 01 | LOW | Sensitive Bandwidth |

| Value | Name | Description |
|---|---|---|
| 10 | MEDIUM | Sensitive Latency |
| 11 | HIGH | Critical Latency |

If a master is configured with QoS level 0x00 or 0x01 there will be minimum one cycle latency for the RAM access.

The priority order for concurrent accesses are decided by two factors. First the QoS level for the master and then a static priority given by table nn-mm (table: SRAM port connection) where the lowest port ID has the highest static priority.

The MTB has fixed QoS level 3 and the DSU has fixed QoS level 1.

The CPU QoS level can be written/read at address 0x41007110, bits [1:0]. Its reset value is 0x0.

Refer to different master QOSCTRL registers for configuring QoS for the other masters (USB, DMAC).

## 11.5 AHB-APB Bridge

The AHB-APB bridge is an AHB slave, providing an interface between the high-speed AHB domain and the low-power APB domain. It is used to provide access to the programmable control registers of peripherals.

AHB-APB bridge is based on AMBA APB Protocol Specification V2.0 (ref. as APB4) including:

- Wait state support
- Error reporting
- Transaction protection
- Sparse data transfer (byte, half-word and word)

Additional enhancements:

- Address and data cycles merged into a single cycle
- Sparse data transfer also apply to read access

to operate the AHB-APB bridge, the clock (CLK_HPBx_AHB) must be enabled. See *PM – Power Manager* for details.

**Figure 11-1. APB Write Access.**

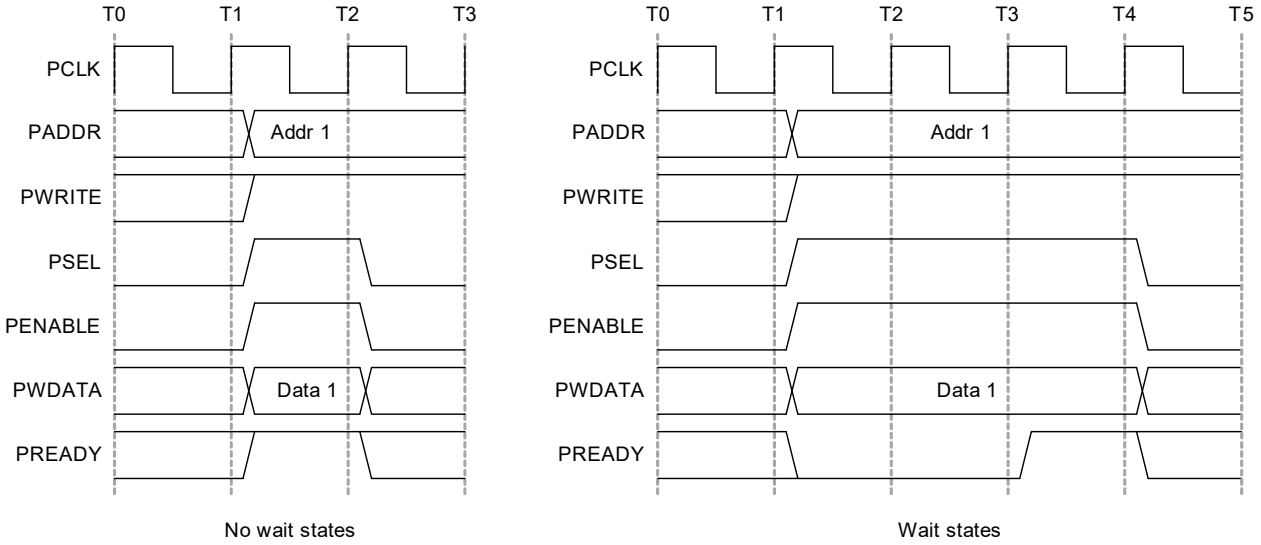
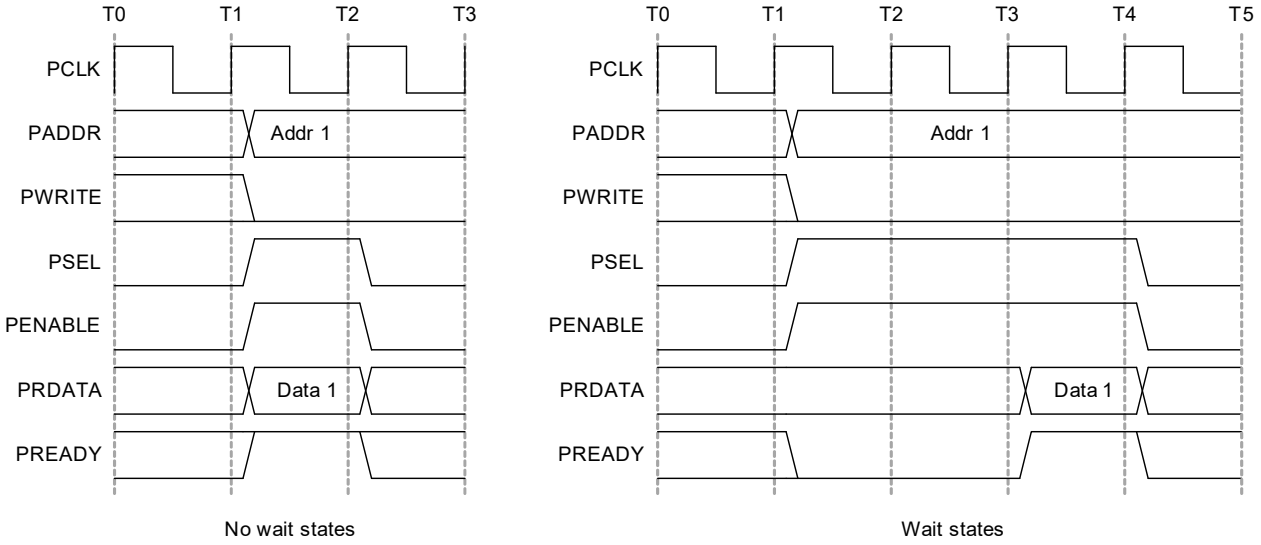



**Figure 11-2. APB Read Access.**




**Related Links**

16. PM – Power Manager

9. Product Mapping

## 11.6 PAC - Peripheral Access Controller

### 11.6.1 Overview

One PAC is associated with each AHB-APB bridge and the PAC can provide write protection for registers of each peripheral connected on the same bridge.

The PAC peripheral bus clock (CLK_PACx_APB) can be enabled and disabled in the Power Manager. CLK_PAC0_APB and CLK_PAC1_APB are enabled are reset. CLK_PAC2_APB is disabled at reset. Refer to *PM – Power Manager* for details. The PAC will continue to operate in any Sleep mode where the selected clock source is running. Write-protection does not apply for debugger access. When the

debugger makes an access to a peripheral, write-protection is ignored so that the debugger can update the register.

Write-protect registers allow the user to disable a selected peripheral's write-protection without doing a read-modify-write operation. These registers are mapped into two I/O memory locations, one for clearing and one for setting the register bits. Writing a one to a bit in the Write Protect Clear register (WPCLR) will clear the corresponding bit in both registers (WPCLR and WPSET) and disable the write-protection for the corresponding peripheral, while writing a one to a bit in the Write Protect Set (WPSET) register will set the corresponding bit in both registers (WPCLR and WPSET) and enable the write-protection for the corresponding peripheral. Both registers (WPCLR and WPSET) will return the same value when read.

If a peripheral is write-protected, and if a write access is performed, data will not be written, and the peripheral will return an access error (CPU exception).

The PAC also offers a safety feature for correct program execution, with a CPU exception generated on double write-protection or double unprotection of a peripheral. If a peripheral n is write-protected and a write to one in WPSET[n] is detected, the PAC returns an error. This can be used to ensure that the application follows the intended program flow by always following a write-protect with an unprotect, and vice versa. However, in applications where a write-protected peripheral is used in several contexts, for example, interrupts, care should be taken so that either the interrupt can not happen while the main application or other interrupt levels manipulate the write-protection status, or when the interrupt handler needs to unprotect the peripheral, based on the current protection status, by reading WPSET.

**Related Links**

16. PM – Power Manager

### 11.6.2 Register Description

Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly. Refer to the Product Mapping for PAC locations.

#### 11.6.2.1 PAC0 Register Description

#### 11.6.2.1.1 Write Protect Clear

**Name:** WPCLR
**Offset:** 0x00
**Reset:** 0x000000
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | EIC | RTC | WDT | GCLK | SYSCTRL | PM | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 6 – EIC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 5 – RTC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 4 – WDT**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 3 – GCLK**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 2 – SYSCTRL**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – PM**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

#### 11.6.2.1.2 Write Protect Set

**Name:** WPSET
**Offset:** 0x04
**Reset:** 0x000000
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | EIC | RTC | WDT | GCLK | SYSCTRL | PM | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 6 – EIC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 5 – RTC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 4 – WDT**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 3 – GCLK**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 2 – SYSCTRL**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – PM**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

#### 11.6.2.2 PAC1 Register Description

#### 11.6.2.2.1 Write Protect Clear

**Name:** WPCLR
**Offset:** 0x00
**Reset:** 0x000002
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | MTB | USB | DMAC | PORT | NVMCTRL | DSU | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | |

**Bit 6 – MTB**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 5 – USB**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 4 – DMAC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 3 – PORT**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 2 – NVMCTRL**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – DSU**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

#### 11.6.2.2.2 Write Protect Set

**Name:** WPSET
**Offset:** 0x04
**Reset:** 0x000002
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | MTB | USB | DMAC | PORT | NVMCTRL | DSU | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | |

**Bit 6 – MTB**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 5 – USB**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 4 – DMAC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 3 – PORT**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 2 – NVMCTRL**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – DSU**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

#### 11.6.2.3 PAC2 Register Description

#### 11.6.2.3.1 Write Protect Clear

**Name:** WPCLR
**Offset:** 0x00
**Reset:** 0x00800000
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | AC1 | I2S | PTC | DAC | AC | ADC |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TC7 | TC6 | TC5 | TC4 | TC3 | TCC2 | TCC1 | TCC0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SERCOM[5:0] | | | | | | EVSYS | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 21 – AC1**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 20 – I2S**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 19 – PTC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 18 – DAC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 17 – AC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 16 – ADC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 11, 12, 13, 14, 15 – TC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 8, 9, 10 – TCC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 7:2 – SERCOM[5:0]**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – EVSYS**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

#### 11.6.2.3.2 Write Protect Set

**Name:** WPSET
**Offset:** 0x04
**Reset:** 0x00800000
**Property:** –

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | AC1 | I2S | PTC | DAC | AC | ADC |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TC7 | TC6 | TC5 | TC4 | TC3 | TCC2 | TCC1 | TCC0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SERCOM5 | SERCOM4 | SERCOM3 | SERCOM2 | SERCOM1 | SERCOM0 | EVSYS | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 21 – AC1**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 20 – I2S**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 19 – PTC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 18 – DAC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 17 – AC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 16 – ADC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 11, 12, 13, 14, 15 – TC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 8, 9, 10 – TCC**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bits 2, 3, 4, 5, 6, 7 – SERCOM**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

**Bit 1 – EVSYS**
Writing a zero to these bits has no effect.

Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

| Value | Description |
|---|---|
| 0 | Write-protection is disabled. |
| 1 | Write-protection is enabled. |

# 12. Peripherals Configuration Summary

**Table 12-1. Peripherals Configuration Summary**

| Periph. Name | Base Address | IRQ Line | AHB Clock Index | AHB Clock Enabled at Reset | APB Clock Index | APB Clock Enabled at Reset | Generic Clock Index | PAC Index | PAC Prot. at Reset | Events User | Events Generator | DMA Index | Sleep Walking |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AHB-APB Bridge A | 0x40000000 | | 0 | Y | | | | | | | | | |
| PAC0 | 0x40000000 | | | | 0 | Y | | | | | | | |
| PM | 0x40000400 | 0 | | | 1 | Y | | 1 | N | | | | Y |
| SYSCTRL | 0x40000800 | 1 | | | 2 | Y | 0: DFLL48M reference<br>1: FDPLL96M clk source<br>2: FDPLL96M 32kHz | 2 | N | | | | Y |
| GCLK | 0x40000C00 | | | | 3 | Y | | 3 | N | | | | Y |
| WDT | 0x40001000 | 2 | | | 4 | Y | 3 | 4 | N | | | | |
| RTC | 0x40001400 | 3 | | | 5 | Y | 4 | 5 | N | | 1: CMP0/ALARM0<br>2: CMP1<br>3: OVF<br>4-11: PER0-7 | | Y |
| EIC | 0x40001800 | NMI, 4 | | | 6 | Y | 5 | 6 | N | | 12-27: EXTINT0-15 | | Y |
| AHB-APB Bridge B | 0x41000000 | | 1 | Y | | | | | | | | | |
| PAC1 | 0x41000000 | | | | 0 | Y | | | | | | | |
| DSU | 0x41002000 | | 3 | Y | 1 | Y | | 1 | Y | | | | |
| NVMCTRL | 0x41004000 | 5 | 4 | Y | 2 | Y | | 2 | N | | | | |
| PORT | 0x41004400 | | | | 3 | Y | | 3 | N | | | | |
| DMAC | 0x41004800 | 6 | 5 | Y | 4 | Y | | 4 | N | 0-3: CH0-3 | 30-33: CH0-3 | | |
| USB | 0x41005000 | 7 | 6 | Y | 5 | Y | 6 | 5 | N | | | | Y |
| MTB | 0x41006000 | | | | | | | 6 | N | | | | |
| AHB-APB Bridge C | 0x42000000 | | 2 | Y | | | | | | | | | |
| PAC2 | 0x42000000 | | | | 0 | N | | | | | | | |
| EVSYS | 0x42000400 | 8 | | | 1 | N | 7-18: one per CHANNEL | 1 | N | | | | Y |
| SERCOM0 | 0x42000800 | 9 | | | 2 | N | 20: CORE<br>19: SLOW | 2 | N | | | 1: RX<br>2: TX | Y |
| SERCOM1 | 0x42000C00 | 10 | | | 3 | N | 21: CORE<br>19: SLOW | 3 | N | | | 3: RX<br>4: TX | Y |
| SERCOM2 | 0x42001000 | 11 | | | 4 | N | 22: CORE<br>19: SLOW | 4 | N | | | 5: RX<br>6: TX | Y |
| SERCOM3 | 0x42001400 | 12 | | | 5 | N | 23: CORE<br>19: SLOW | 5 | N | | | 7: RX<br>8: TX | Y |
| SERCOM4 | 0x42001800 | 13 | | | 6 | N | 24: CORE<br>19: SLOW | 6 | N | | | 9: RX<br>10: TX | Y |
| SERCOM5 | 0x42001C00 | 14 | | | 7 | N | 25: CORE<br>19: SLOW | 7 | N | | | 11: RX<br>12: TX | Y |
| TCC0 | 0x42002000 | 15 | | | 8 | N | 26 | 8 | N | 4-5: EV0-1<br>6-9: MC0-3 | 34: OVF<br>35: TRG<br>36: CNT<br>37-40: MC0-3 | 13: OVF<br>14-17: MC0-3 | Y |
| TCC1 | 0x42002400 | 16 | | | 9 | N | 26 | 9 | N | 10-11: EV0-1<br>12-13: MC0-1 | 41: OVF<br>42: TRG<br>43: CNT<br>44-45: MC0-1 | 18: OVF<br>19-20: MC0-1 | Y |
| TCC2 | 0x42002800 | 17 | | | 10 | N | 27 | 10 | N | 14-15: EV0-1<br>16-17: MC0-1 | 46: OVF<br>47: TRG<br>48: CNT<br>49-50: MC0-1 | 21: OVF<br>22-23: MC0-1 | Y |
| TC3 | 0x42002C00 | 18 | | | 11 | N | 27 | 11 | N | 18: EV | 51: OVF<br>52-53: MC0-1 | 24: OVF<br>25-26: MC0-1 | Y |

| Periph. Name | Base Address | IRQ Line | AHB Clock | | APB Clock | | Generic Clock | PAC | | Events | | DMA | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Index | Enabled at Reset | Index | Enabled at Reset | Index | Index | Prot. at Reset | User | Generator | Index | Sleep Walking |
| TC4 | 0x42003000 | 19 | | | 12 | N | 28 | 12 | N | 19: EV | 54: OVF<br>55-56: MCX0-1 | 27: OVF<br>28-29: MC0-1 | Y |
| TC5 | 0x42003400 | 20 | | | 13 | N | 28 | 13 | N | 20: EV | 57: OVF<br>58-59: MC0-1 | 30: OVF<br>31-32: MC0-1 | Y |
| TC6 | 0x42003800 | 21 | | | 14 | N | 29 | 14 | N | 21: EV | 60: OVF<br>61-62: MC0-1 | 33: OVF<br>34-35: MC0-1 | Y |
| TC7 | 0x42003C00 | 22 | | | 15 | N | 29 | 15 | N | 22: EV | 63: OVF<br>64-65: MC0-1 | 36: OVF<br>37-38: MC0-1 | Y |
| ADC | 0x42004000 | 23 | | | 16 | Y | 30 | 16 | N | 23: START<br>24: SYNC | 66: RESRDY<br>67: WINMON | 39: RESRDY | Y |
| AC | 0x42004400 | 24 | | | 17 | N | 31: DIG<br>32: ANA | 17 | N | 25-26: SOC0-1 | 68-69: COMP0-1<br>70: WIN0 | | Y |
| DAC | 0x42004800 | 25 | | | 18 | N | 33 | 18 | N | 27: START | 71: EMPTY | 40: EMPTY | Y |
| PTC | 0x42004C00 | 26 | | | 19 | N | 34 | 19 | N | 28: STCONV | 72: EOC<br>73: WCOMP | | |
| I2S | 0x42005000 | 27 | | | 20 | N | 35-36 | 20 | N | | | 41:42: RX<br>43:44: TX | Y |
| AC1 | 0x42005400 | 28 | | | 19 | N | 31: DIG<br>32: ANA | 19 | N | 25-26: SOC0-1 | 68-69: COMP0-1<br>70: WIN0 | | Y |

# 13. DSU - Device Service Unit

## 13.1 Overview

The Device Service Unit (DSU) provides a means of detecting debugger probes. It enables the ARM Debug Access Port (DAP) to have control over multiplexed debug pads and CPU reset. The DSU also provides system-level services to debug adapters in an ARM debug system. It implements a CoreSight Debug ROM that provides device identification as well as identification of other debug components within the system. Hence, it complies with the ARM Peripheral Identification specification. The DSU also provides system services to applications that need memory testing, as required for IEC60730 Class B compliance, for example. The DSU can be accessed simultaneously by a debugger and the CPU, as it is connected on the High-Speed Bus Matrix. For security reasons, some of the DSU features will be limited or unavailable when the device is protected by the NVMCTRL security bit.

**Related Links**

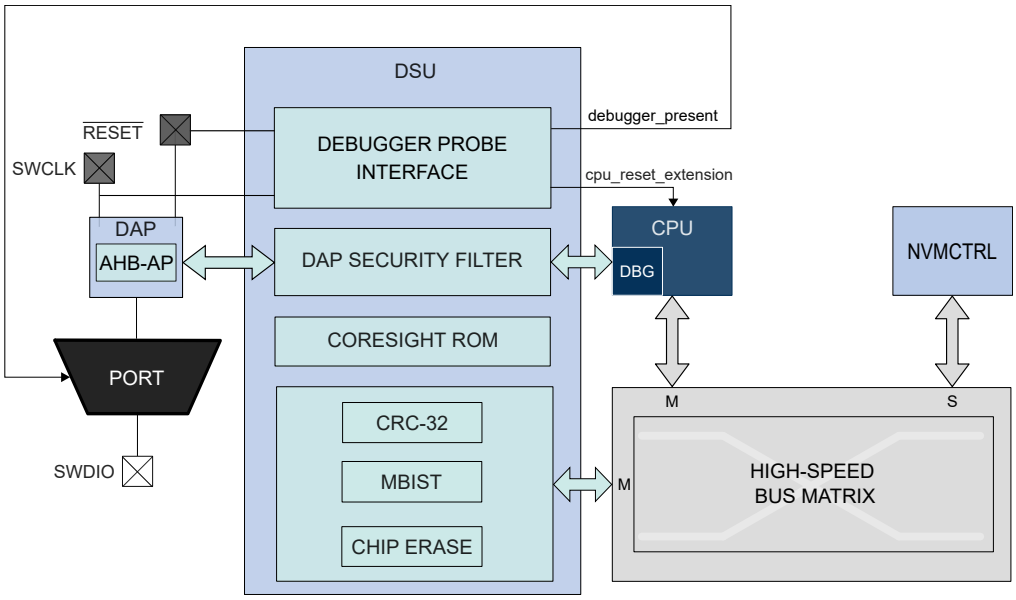
13.11.6 System Services Availability when Accessed Externally and Device is Protected
22. NVMCTRL – Nonvolatile Memory Controller
22.6.6 Security Bit

## 13.2 Features

- CPU reset extension
- Debugger probe detection (Cold- and Hot-Plugging)
- Chip-Erase command and status
- 32-bit cyclic redundancy check (CRC32) of any memory accessible through the bus matrix
- ARM® CoreSight™ compliant device identification
- Two debug communications channels
- Debug access port security filter
- Onboard memory built-in self-test (MBIST)

## 13.3 Block Diagram

**Figure 13-1. DSU Block Diagram**




## 13.4 Signal Description

The DSU uses three signals to function.

| Signal Name | Type | Description |
|---|---|---|
| $\overline{\text{RESET}}$ | Digital Input | External reset |
| SWCLK | Digital Input | SW clock |
| SWDIO | Digital I/O | SW bidirectional data pin |

**Related Links**

7. I/O Multiplexing and Considerations

## 13.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 13.5.1 I/O Lines

The SWCLK pin is by default assigned to the DSU module to allow debugger probe detection and to stretch the CPU reset phase. For more information, refer to 13.6.3 Debugger Probe Detection. The Hot-Plugging feature depends on the PORT configuration. If the SWCLK pin function is changed in the PORT or if the PORT_MUX is disabled, the Hot-Plugging feature is disabled until a power-reset or an external reset is performed.

### 13.5.2 Power Management

The DSU will continue to operate in Idle mode.

**Related Links**

16. PM – Power Manager

### 13.5.3 Clocks

The DSU bus clocks (CLK_DSU_APB and CLK_DSU_AHB) can be enabled and disabled by the Power Manager. Refer to *PM – Power Manager*

**Related Links**

16. PM – Power Manager

### 13.5.4 DMA

Not applicable.

### 13.5.5 Interrupts

Not applicable.

### 13.5.6 Events

Not applicable.

### 13.5.7 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Debug Communication Channel 0 register (DCC0)
- Debug Communication Channel 1 register (DCC1)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 13.5.8 Analog Connections

Not applicable.

## 13.6 Debug Operation

### 13.6.1 Principle of Operation

The DSU provides basic services to allow on-chip debug using the ARM Debug Access Port and the ARM processor debug resources:

- CPU reset extension
- Debugger probe detection

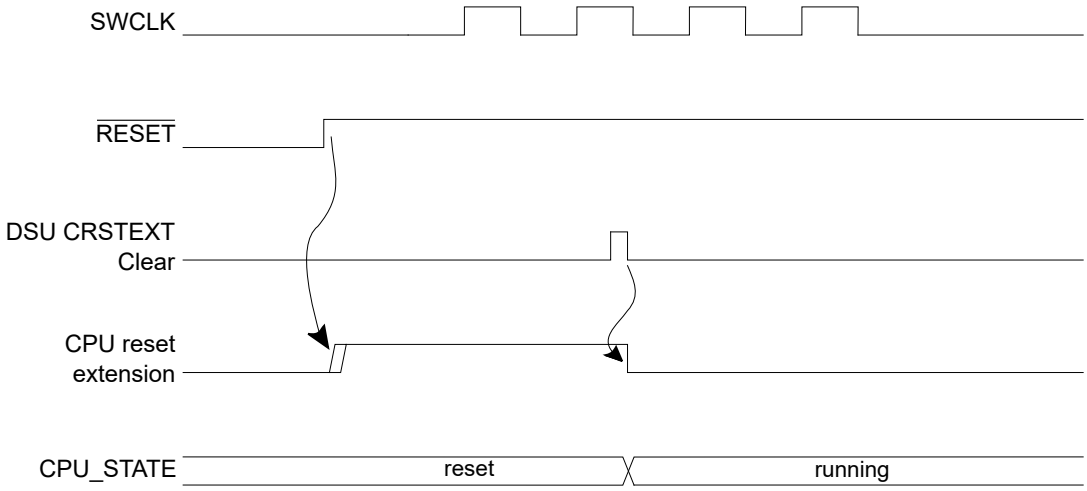
For more details on the ARM debug components, refer to the ARM Debug Interface v5 Architecture Specification.

### 13.6.2 CPU Reset Extension

“CPU reset extension” refers to the extension of the reset phase of the CPU core after the external reset is released. This ensures that the CPU is not executing code at startup while a debugger is connects to

the system. The debugger is detected on a $\overline{\text{RESET}}$ release event when SWCLK is low. At startup, SWCLK is internally pulled up to avoid false detection of a debugger if the SWCLK pin is left unconnected. When the CPU is held in the reset extension phase, the CPU Reset Extension bit of the Status A register (STATUSA.CRSTEXT) is set. To release the CPU, write a '1' to STATUSA.CRSTEXT. STATUSA.CRSTEXT will then be set to '0'. Writing a '0' to STATUSA.CRSTEXT has no effect. For security reasons, it is not possible to release the CPU reset extension when the device is protected by the NVMCTRL security bit. Trying to do so sets the Protection Error bit (PERR) of the Status A register (STATUSA.PERR).

**Figure 13-2. Typical CPU Reset Extension Set and Clear Timing Diagram**




**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

22.6.6 Security Bit

### 13.6.3 Debugger Probe Detection

#### 13.6.3.1 Cold Plugging

Cold-Plugging is the detection of a debugger when the system is in reset. Cold-Plugging is detected when the CPU reset extension is requested, as described above.

#### 13.6.3.2 Hot Plugging

Hot-Plugging is the detection of a debugger probe when the system is not in reset. Hot-Plugging is not possible under reset because the detector is reset when POR or $\overline{\text{RESET}}$ are asserted. Hot-Plugging is active when a SWCLK falling edge is detected. The SWCLK pad is multiplexed with other functions and the user must ensure that its default function is assigned to the debug system. If the SWCLK function is changed, the Hot-Plugging feature is disabled until a power-reset or external reset occurs. Availability of the Hot-Plugging feature can be read from the Hot-Plugging Enable bit of the Status B register (STATUSB.HPE).

**Figure 13-3. Hot-Plugging Detection Timing Diagram**

The presence of a debugger probe is detected when either Hot-Plugging or Cold-Plugging is detected. Once detected, the Debugger Present bit of the Status B register (STATUSB.DBGPRES) is set. For security reasons, Hot-Plugging is not available when the device is protected by the NVMCTRL security bit.

This detection requires that pads are correctly powered. Thus, at cold startup, this detection cannot be done until POR is released. If the device is protected, Cold-Plugging is the only way to detect a debugger probe, and so the external reset timing must be longer than the POR timing. If external reset is deasserted before POR release, the user must retry the procedure above until it gets connected to the device.

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

22.6.6 Security Bit

## 13.7 Chip Erase

Chip-Erase consists of removing all sensitive information stored in the chip and clearing the NVMCTRL security bit. Therefore, all volatile memories and the Flash memory (including the EEPROM emulation area) will be erased. The Flash auxiliary rows, including the user row, will not be erased.

When the device is protected, the debugger must first reset the device in order to be detected. This ensures that internal registers are reset after the protected state is removed. The Chip-Erase operation is triggered by writing a '1' to the Chip-Erase bit in the Control register (CTRL.CE). This command will be discarded if the DSU is protected by the Peripheral Access Controller (PAC). Once issued, the module clears volatile memories prior to erasing the Flash array. To ensure that the Chip-Erase operation is completed, check the Done bit of the Status A register (STATUSA.DONE).

The Chip-Erase operation depends on clocks and power management features that can be altered by the CPU. For that reason, it is recommended to issue a Chip- Erase after a Cold-Plugging procedure to ensure that the device is in a known and safe state.

The recommended sequence is as follows:

1. Issue the Cold-Plugging procedure (refer to 13.6.3.1 Cold Plugging). The device then:
   1.1. Detects the debugger probe.
   1.2. Holds the CPU in reset.
2. Issue the Chip-Erase command by writing a '1' to CTRL.CE. The device then:
   2.1. Clears the system volatile memories.

2.2. Erases the whole Flash array (including the EEPROM emulation area, not including auxiliary rows).
2.3. Erases the lock row, removing the NVMCTRL security bit protection.
3. Check for completion by polling STATUSA.DONE (read as '1' when completed).
4. Reset the device to let the NVMCTRL update the fuses.

## 13.8 Programming

Programming the Flash or RAM memories is only possible when the device is not protected by the NVMCTRL security bit. The programming procedure is as follows:

1. At power up, $\overline{\text{RESET}}$ is driven low by a debugger. The on-chip regulator holds the system in a POR state until the input supply is above the POR threshold (refer to Powe-On Reset (POR) characteristics). The system continues to be held in this static state until the internally regulated supplies have reached a safe operating state.
2. The PM starts, clocks are switched to the slow clock (Core Clock, System Clock, Flash Clock and any Bus Clocks that do not have clock gate control). Internal resets are maintained due to the external reset.
3. The debugger maintains a low level on SWCLK. $\overline{\text{RESET}}$ is released, resulting in a debugger Cold-Plugging procedure.
4. The debugger generates a clock signal on the SWCLK pin, the Debug Access Port (DAP) receives a clock.
5. The CPU remains in Reset due to the Cold-Plugging procedure; meanwhile, the rest of the system is released.
6. A Chip-Erase is issued to ensure that the Flash is fully erased prior to programming.
7. Programming is available through the AHB-AP.
8. After the operation is completed, the chip can be restarted either by asserting $\overline{\text{RESET}}$ or toggling power. Make sure that the SWCLK pin is high when releasing $\overline{\text{RESET}}$ to prevent extending the CPU reset.

**Related Links**

37. Electrical Characteristics
22. NVMCTRL – Nonvolatile Memory Controller
22.6.6 Security Bit

## 13.9 Intellectual Property Protection

Intellectual property protection consists of restricting access to internal memories from external tools when the device is protected, and this is accomplished by setting the NVMCTRL security bit. This protected state can be removed by issuing a Chip-Erase (refer to 13.7 Chip Erase). When the device is protected, read/write accesses using the AHB-AP are limited to the DSU address range and DSU commands are restricted. When issuing a Chip-Erase, sensitive information is erased from volatile memory and Flash.

The DSU implements a security filter that monitors the AHB transactions inside the DAP. If the device is protected, then AHB-AP read/write accesses outside the DSU external address range are discarded, causing an error response that sets the ARM AHB-AP sticky error bits (refer to the ARM Debug Interface v5 Architecture Specification on http://www.arm.com).

The DSU is intended to be accessed either:

- Internally from the CPU, without any limitation, even when the device is protected
- Externally from a debug adapter, with some restrictions when the device is protected

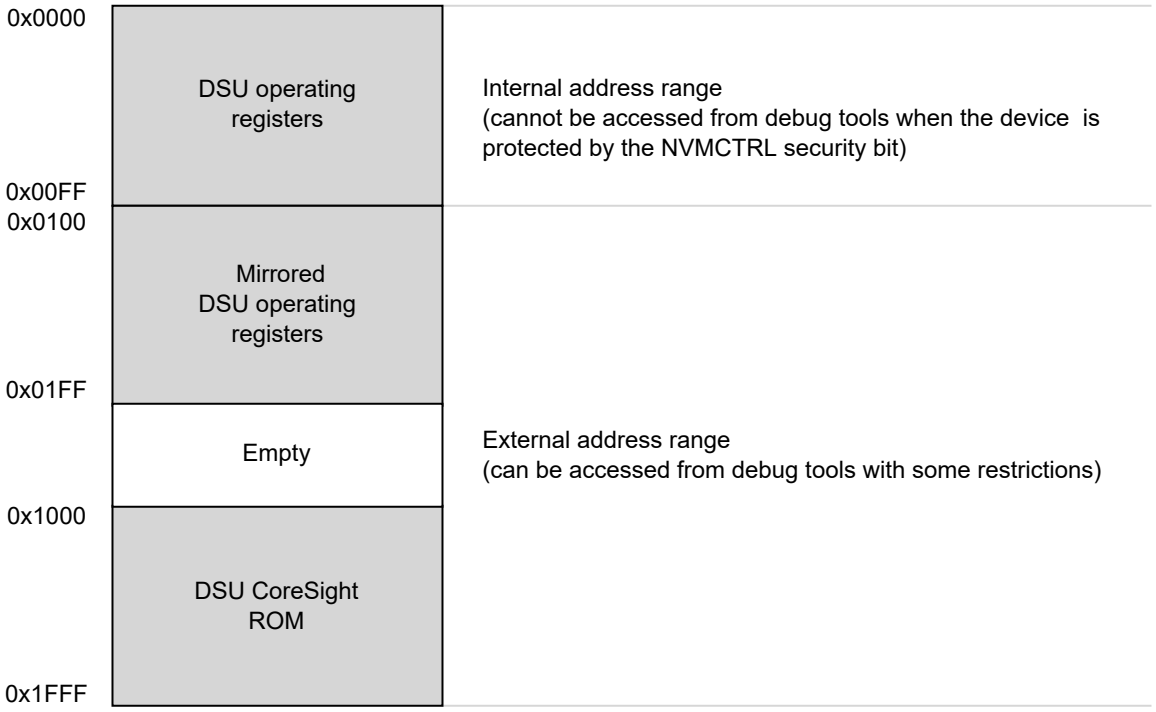
For security reasons, DSU features have limitations when used from a debug adapter. To differentiate external accesses from internal ones, the first 0x100 bytes of the DSU register map has been mirrored at offset 0x100:

- The first 0x100 bytes form the internal address range
- The next 0x100 bytes form the external address range

When the device is protected, the DAP can only issue MEM-AP accesses in the DSU range 0x0100-0x2000.

The DSU operating registers are located in the 0x0000-0x00FF area and remapped in 0x0100-0x01FF to differentiate accesses coming from a debugger and the CPU. If the device is protected and an access is issued in the region 0x0100-0x01FF, it is subject to security restrictions. For more information, refer to the Table 13-1.

**Figure 13-4. APB Memory Mapping**




Some features not activated by APB transactions are not available when the device is protected:

**Table 13-1. Feature Availability Under Protection**

| Features | Availability when the device is protected |
|---|---|
| CPU Reset Extension | Yes |
| Clear CPU Reset Extension | No |
| Debugger Cold-Plugging | Yes |
| Debugger Hot-Plugging | No |

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller
22.6.6 Security Bit

## 13.10 Device Identification

Device identification relies on the ARM CoreSight component identification scheme, which allows the chip to be identified as a SAM device implementing a DSU. The DSU contains identification registers to differentiate the device.

### 13.10.1 CoreSight Identification

A system-level ARM® CoreSight™ ROM table is present in the device to identify the vendor and the chip identification method. Its address is provided in the MEM-AP BASE register inside the ARM Debug Access Port. The CoreSight ROM implements a 64-bit conceptual ID composed as follows from the PID0 to PID7 CoreSight ROM Table registers:

**Figure 13-5. Conceptual 64-bit Peripheral ID**

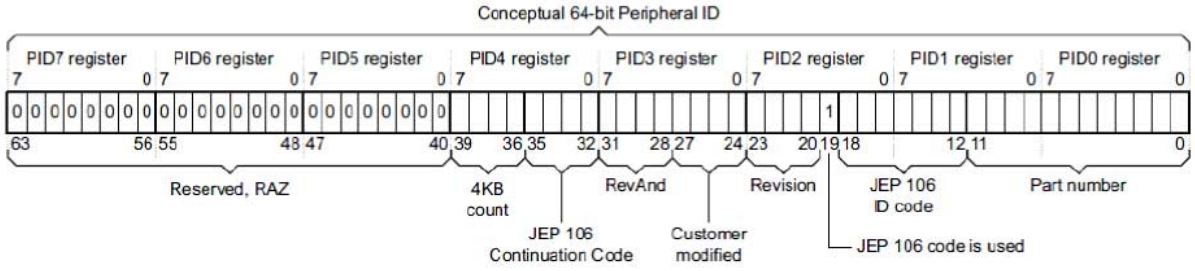



**Table 13-2. Conceptual 64-Bit Peripheral ID Bit Descriptions**

| Field | Size | Description | Location |
|---|---|---|---|
| JEP-106 CC code | 4 | Continuation code: 0x0 | PID4 |
| JEP-106 ID code | 7 | Device ID: 0x1F | PID1+PID2 |
| 4KB count | 4 | Indicates that the CoreSight component is a ROM: 0x0 | PID4 |
| RevAnd | 4 | Not used; read as 0 | PID3 |
| CUSMOD | 4 | Not used; read as 0 | PID3 |
| PARTNUM | 12 | Contains 0xCD0 to indicate that DSU is present | PID0+PID1 |
| REVISION | 4 | DSU revision (starts at 0x0 and increments by 1 at both major and minor revisions). Identifies DSU identification method variants. If 0x0, this indicates that device identification can be completed by reading the Device Identification register (DID) | PID2 |

For more information, refer to the ARM Debug Interface Version 5 Architecture Specification.

### 13.10.2 Chip Identification Method

The DSU DID register identifies the device by implementing the following information:

- Processor identification
- Product family identification

- Product series identification
- Device select

## 13.11 Functional Description

### 13.11.1 Principle of Operation

The DSU provides memory services, such as CRC32 or MBIST that require almost the same interface. Hence, the Address, Length and Data registers (ADDR, LENGTH, DATA) are shared. These shared registers must be configured first; then a command can be issued by writing the Control register. When a command is ongoing, other commands are discarded until the current operation is completed. Hence, the user must wait for the STATUSA.DONE bit to be set prior to issuing another one.

### 13.11.2 Basic Operation

#### 13.11.2.1 Initialization

The module is enabled by enabling its clocks. For more details, refer to 13.5.3 Clocks. The DSU registers can be PAC write-protected.

**Related Links**

11.6 PAC - Peripheral Access Controller

#### 13.11.2.2 Operation From a Debug Adapter

Debug adapters should access the DSU registers in the external address range 0x100 – 0x2000. If the device is protected by the NVMCTRL security bit, accessing the first 0x100 bytes causes the system to return an error. Refer to 13.9 Intellectual Property Protection.

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

22.6.6 Security Bit

#### 13.11.2.3 Operation From the CPU

There are no restrictions when accessing DSU registers from the CPU. However, the user should access DSU registers in the internal address range (0x0 – 0x100) to avoid external security restrictions. Refer to 13.9 Intellectual Property Protection.

### 13.11.3 32-bit Cyclic Redundancy Check CRC32

The DSU unit provides support for calculating a cyclic redundancy check (CRC32) value for a memory area (including Flash and AHB RAM).

When the CRC32 command is issued from:

- The internal range, the CRC32 can be operated at any memory location
- The external range, the CRC32 operation is restricted; DATA, ADDR, and LENGTH values are forced (see below)

**Table 13-3. AMOD Bit Descriptions when Operating CRC32**

| AMOD[1:0] | Short name | External range restrictions |
|---|---|---|
| 0 | ARRAY | CRC32 is restricted to the full Flash array area (EEPROM emulation area not included) DATA forced to 0xFFFFFFFF before calculation (no seed) |
| 1 | EEPROM | CRC32 of the whole EEPROM emulation area DATA forced to 0xFFFFFFFF before calculation (no seed) |
| 2-3 | Reserved | |

The algorithm employed is the industry standard CRC32 algorithm using the generator polynomial 0xEDB88320 (reversed representation).

#### 13.11.3.1 Starting CRC32 Calculation

CRC32 calculation for a memory range is started after writing the start address into the Address register (ADDR) and the size of the memory range into the Length register (LENGTH). Both must be word-aligned.

The initial value used for the CRC32 calculation must be written to the Data register (DATA). This value will usually be 0xFFFFFFFF, but can be, for example, the result of a previous CRC32 calculation if generating a common CRC32 of separate memory blocks.

Once completed, the calculated CRC32 value can be read out of the Data register. The read value must be complemented to match standard CRC32 implementations or kept non-inverted if used as starting point for subsequent CRC32 calculations.

The actual test is started by writing a '1' in the 32-bit Cyclic Redundancy Check bit of the Control register (CTRL.CRC). A running CRC32 operation can be canceled by resetting the module (writing '1' to CTRL.SWRST).

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

22.6.6 Security Bit

#### 13.11.3.2 Interpreting the Results

The user should monitor the Status A register. When the operation is completed, STATUSA.DONE is set. Then the Bus Error bit of the Status A register (STATUSA.BERR) must be read to ensure that no bus error occurred.

### 13.11.4 Debug Communication Channels

The Debug Communication Channels (DCCO and DCC1) consist of a pair of registers with associated handshake logic, accessible by both CPU and debugger even if the device is protected by the NVMCTRL security bit. The registers can be used to exchange data between the CPU and the debugger, during run time as well as in debug mode. This enables the user to build a custom debug protocol using only these registers.

The DCC0 and DCC1 registers are accessible when the protected state is active. When the device is protected, however, it is not possible to connect a debugger while the CPU is running (STATUSA.CRSTEXT is not writable and the CPU is held under Reset).

Two Debug Communication Channel status bits in the Status B registers (STATUS.DCCDx) indicate whether a new value has been written in DCC0 or DCC1. These bits, DCC0D and DCC1D, are located in the STATUSB registers. They are automatically set on write and cleared on read.

**Note:** The DCC0 and DCC1 registers are shared with the on-board memory testing logic (MBIST). Accordingly, DCC0 and DCC1 must not be used while performing MBIST operations.

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller

22.6.6 Security Bit

### 13.11.5 Testing of On-Board Memories MBIST

The DSU implements a feature for automatic testing of memory, also known as MBIST (memory built-in self test). This is primarily intended for production test of on-board memories. MBIST cannot be operated from the external address range when the device is protected by the NVMCTRL security bit. If an MBIST command is issued when the device is protected, a protection error is reported in the Protection Error bit in the Status A register (STATUSA.PERR).

1. Algorithm
   The algorithm used for testing is a type of March algorithm called "March LR". This algorithm is able to detect a wide range of memory defects, while still keeping a linear run time. The algorithm is:

   1.1. Write entire memory to '0', in any order.
   1.2. Bit by bit read '0', write '1', in descending order.
   1.3. Bit by bit read '1', write '0', read '0', write '1', in ascending order.
   1.4. Bit by bit read '1', write '0', in ascending order.
   1.5. Bit by bit read '0', write '1', read '1', write '0', in ascending order.
   1.6. Read '0' from entire memory, in ascending order.

   The specific implementation used as a run time which depends on the CPU clock frequency and the number of bytes tested in the RAM. The detected faults are:

   - Address decoder faults
   - Stuck-at faults
   - Transition faults
   - Coupling faults
   - Linked Coupling faults

2. Starting MBIST
   To test a memory, you need to write the start address of the memory to the ADDR.ADDR bit field, and the size of the memory into the Length register.

   For best test coverage, an entire physical memory block should be tested at once. It is possible to test only a subset of a memory, but the test coverage will then be somewhat lower.

   The actual test is started by writing a '1' to CTRL.MBIST. A running MBIST operation can be canceled by writing a '1' to CTRL.SWRST.

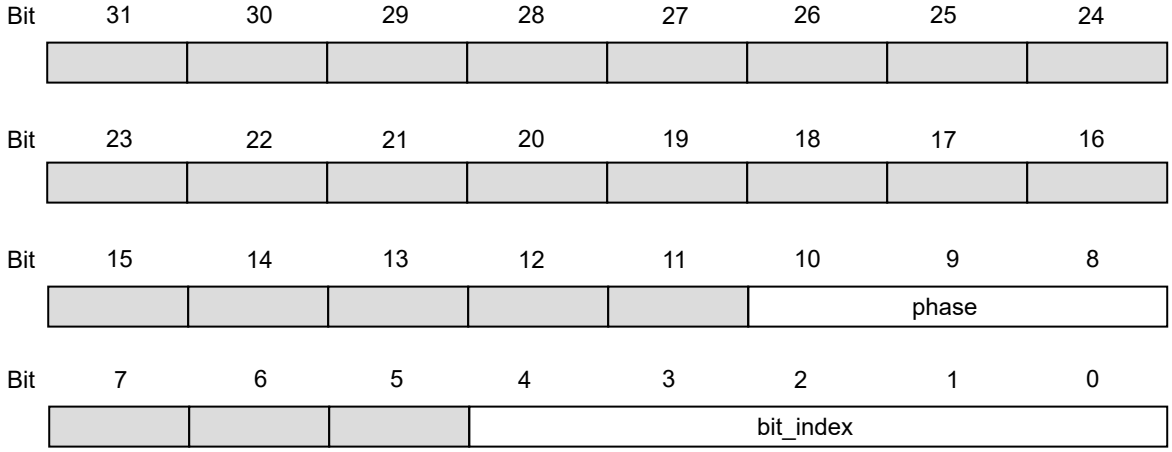
3. Interpreting the Results
   The tester should monitor the STATUSA register. When the operation is completed, STATUSA.DONE is set. There are two different modes:

   - ADDR.AMOD=0: exit-on-error (default)
     In this mode, the algorithm terminates either when a fault is detected or on successful completion. In both cases, STATUSA.DONE is set. If an error was detected, STATUSA.FAIL will be set. User then can read the DATA and ADDR registers to locate the fault.
   - ADDR.AMOD=1: pause-on-error

In this mode, the MBIST algorithm is paused when an error is detected. In such a situation, only STATUSA.FAIL is asserted. The state machine waits for user to clear STATUSA.FAIL by writing a '1' in STATUSA.FAIL to resume. Prior to resuming, user can read the DATA and ADDR registers to locate the fault.

4. Locating Faults
   If the test stops with STATUSA.FAIL set, one or more bits failed the test. The test stops at the first detected error. The position of the failing bit can be found by reading the following registers:
   - ADDR: Address of the word containing the failing bit
   - DATA: contains data to identify which bit failed, and during which phase of the test it failed. The DATA register will in this case contains the following bit groups:

**Figure 13-6. DATA bits Description When MBIST Operation Returns an Error**




- bit_index: contains the bit number of the failing bit
- phase: indicates which phase of the test failed and the cause of the error, as listed in the following table.

**Table 13-4. MBIST Operation Phases**

| Phase | Test actions |
|---|---|
| 0 | Write all bits to zero. This phase cannot fail. |
| 1 | Read '0', write '1', increment address |
| 2 | Read '1', write '0' |
| 3 | Read '0', write '1', decrement address |
| 4 | Read '1', write '0', decrement address |
| 5 | Read '0', write '1' |
| 6 | Read '1', write '0', decrement address |
| 7 | Read all zeros. bit_index is not used |

**Table 13-5. AMOD Bit Descriptions for MBIST**

| AMOD[1:0] | Description |
|---|---|
| 0x0 | Exit on Error |
| 0x1 | Pause on Error |
| 0x2, 0x3 | Reserved |

**Related Links**

22. NVMCTRL – Nonvolatile Memory Controller
22.6.6 Security Bit
9. Product Mapping

### 13.11.6 System Services Availability when Accessed Externally and Device is Protected

External access: Access performed in the DSU address offset 0x200-0x1FFF range.

Internal access: Access performed in the DSU address offset 0x000-0x100 range.

**Table 13-6. Available Features when Operated From The External Address Range and Device is Protected**

| Features | Availability From The External Address Range and Device is Protected |
|---|---|
| Chip-Erase command and status | Yes |
| CRC32 | Yes, only full array or full EEPROM |
| CoreSight Compliant Device identification | Yes |
| Debug communication channels | Yes |
| Testing of onboard memories (MBIST) | No |
| STATUSA.CRSTEXT clearing | No (STATUSA.PERR is set when attempting to do so) |

## 13.12 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | CE | MBIST | CRC | | SWRST |
| 0x01 | STATUSA | 7:0 | | | | PERR | FAIL | BERR | CRSTEXT | DONE |
| 0x02 | STATUSB | 7:0 | | | | HPE | DCCD1 | DCCD0 | DBGPRES | PROT |
| 0x03 | Reserved | | | | | | | | | |
| 0x04 | ADDR | 7:0 | ADDR[5:0] | | | | | | AMOD[1:0] | |
| | | 15:8 | ADDR[13:6] | | | | | | | |
| | | 23:16 | ADDR[21:14] | | | | | | | |
| | | 31:24 | ADDR[29:22] | | | | | | | |
| 0x08 | LENGTH | 7:0 | LENGTH[5:0] | | | | | | | |
| | | 15:8 | LENGTH[13:6] | | | | | | | |
| | | 23:16 | LENGTH[21:14] | | | | | | | |
| | | 31:24 | LENGTH[29:22] | | | | | | | |
| 0x0C | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x10 | DCC0 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x14 | DCC1 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x18 | DID | 7:0 | DEVSEL[7:0] | | | | | | | |
| | | 15:8 | DIE[3:0] | | | | REVISION[3:0] | | | |
| | | 23:16 | FAMILY[0:0] | | SERIES[5:0] | | | | | |
| | | 31:24 | PROCESSOR[3:0] | | | | FAMILY[4:1] | | | |
| 0x1C ... 0x0FFF | Reserved | | | | | | | | | |
| 0x1000 | ENTRY0 | 7:0 | | | | | | | FMT | EPRES |
| | | 15:8 | ADDOFF[3:0] | | | | | | | |
| | | 23:16 | ADDOFF[11:4] | | | | | | | |
| | | 31:24 | ADDOFF[19:12] | | | | | | | |
| 0x1004 | ENTRY1 | 7:0 | | | | | | | FMT | EPRES |
| | | 15:8 | ADDOFF[3:0] | | | | | | | |
| | | 23:16 | ADDOFF[11:4] | | | | | | | |
| | | 31:24 | ADDOFF[19:12] | | | | | | | |
| 0x1008 | END | 7:0 | END[7:0] | | | | | | | |
| | | 15:8 | END[15:8] | | | | | | | |
| | | 23:16 | END[23:16] | | | | | | | |
| | | 31:24 | END[31:24] | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x100C ... 0x1FCB | Reserved | | | | | | | | | |
| 0x1FCC | MEMTYPE | 7:0 | | | | | | | | SMEMP |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FD0 | PID4 | 7:0 | FKBC[3:0] | | | | JEPCC[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FD4 ... 0x1FDF | Reserved | | | | | | | | | |
| 0x1FE0 | PID0 | 7:0 | PARTNBL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FE4 | PID1 | 7:0 | JEPIDCL[3:0] | | | | PARTNBH[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FE8 | PID2 | 7:0 | REVISION[3:0] | | | | JEPU | JEPIDCH[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FEC | PID3 | 7:0 | REVAND[3:0] | | | | CUSMOD[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FF0 | CID0 | 7:0 | PREAMBLEB0[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FF4 | CID1 | 7:0 | CCLASS[3:0] | | | | PREAMBLE[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FF8 | CID2 | 7:0 | PREAMBLEB2[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1FFC | CID3 | 7:0 | PREAMBLEB3[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

## 13.13 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 13.5.7 Register Access Protection.

### 13.13.1 Control

**Name:** CTRL
**Offset:** 0x0000
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | CE | MBIST | CRC | | SWRST |
| Access | | | | W | W | W | | W |
| Reset | | | | 0 | 0 | 0 | | 0 |

**Bit 4 – CE** Chip-Erase
Writing a '0' to this bit has no effect.

Writing a '1' to this bit starts the Chip-Erase operation.

**Bit 3 – MBIST** Memory Built-In Self-Test
Writing a '0' to this bit has no effect.

Writing a '1' to this bit starts the memory BIST algorithm.

**Bit 2 – CRC** 32-bit Cyclic Redundancy Check
Writing a '0' to this bit has no effect.

Writing a '1' to this bit starts the cyclic redundancy check algorithm.

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets the module.

### 13.13.2 Status A

**Name:** STATUSA
**Offset:** 0x0001
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | PERR | FAIL | BERR | CRSTEXT | DONE |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 4 – PERR** Protection Error
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Protection Error bit.

This bit is set when a command that is not allowed in protected state is issued.

**Bit 3 – FAIL** Failure
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Failure bit.

This bit is set when a DSU operation failure is detected.

**Bit 2 – BERR** Bus Error
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Bus Error bit.

This bit is set when a bus error is detected.

**Bit 1 – CRSTEXT** CPU Reset Phase Extension
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the CPU Reset Phase Extension bit.

This bit is set when a debug adapter Cold-Plugging is detected, which extends the CPU reset phase.

**Bit 0 – DONE** Done
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Done bit.

This bit is set when a DSU operation is completed.

### 13.13.3 Status B

**Name:** STATUSB
**Offset:** 0x0002
**Reset:** 0x1X
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | HPE | DCCD1 | DCCD0 | DBGPRES | PROT |
| Access | | | | R | R | R | R | R |
| Reset | | | | 1 | 0 | 0 | 0 | 0 |

**Bit 4 – HPE** Hot-Plugging Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit has no effect.

This bit is set when Hot-Plugging is enabled.

This bit is cleared when Hot-Plugging is disabled. This is the case when the SWCLK function is changed. Only a power-reset or a external reset can set it again.

**Bits 2, 3 – DCCDx** Debug Communication Channel x Dirty [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit has no effect.

This bit is set when DCCx is written.

This bit is cleared when DCCx is read.

**Bit 1 – DBGPRES** Debugger Present
Writing a '0' to this bit has no effect.

Writing a '1' to this bit has no effect.

This bit is set when a debugger probe is detected.

This bit is never cleared.

**Bit 0 – PROT** Protected
Writing a '0' to this bit has no effect.

Writing a '1' to this bit has no effect.

This bit is set at power-up when the device is protected.

This bit is never cleared.

### 13.13.4 Address

**Name:** ADDR
**Offset:** 0x0004
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[5:0] | | | | | | AMOD[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:2 – ADDR[29:0]** Address
Initial word start address needed for memory operations.

**Bits 1:0 – AMOD[1:0]** Access Mode
The functionality of these bits is dependent on the operation mode.

Bit description when operating CRC32: refer to 13.11.3 32-bit Cyclic Redundancy Check CRC32

Bit description when testing onboard memories (MBIST): refer to 13.11.5 Testing of On-Board Memories MBIST

### 13.13.5 Length

**Name:** LENGTH
**Offset:** 0x0008
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LENGTH[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LENGTH[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LENGTH[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LENGTH[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – LENGTH[29:0]** Length
Length in words needed for memory operations.

### 13.13.6 Data

**Name:** DATA
**Offset:** 0x000C
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data
Memory operation initial value or result value.

### 13.13.7 Debug Communication Channel 0

**Name:** DCC0
**Offset:** 0x0010
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data
Data register.

### 13.13.8 Debug Communication Channel 1

**Name:** DCC1
**Offset:** 0x0014
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data
Data register.

### 13.13.9 Device Identification

**Name:** DID
**Offset:** 0x0018
**Property:** PAC Write-Protection

The information in this register is related to the *Ordering Information*.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PROCESSOR[3:0] | | | | FAMILY[4:1] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | p | p | p | p | f | f | f | f |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FAMILY[0:0] | | SERIES[5:0] | | | | | |
| Access | R | | R | R | R | R | R | R |
| Reset | f | | s | s | s | s | s | s |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DIE[3:0] | | | | REVISION[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | d | d | d | d | r | r | r | r |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DEVSEL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

**Bits 31:28 – PROCESSOR[3:0]** Processor
The value of this field defines the processor used on the device.

**Bits 27:23 – FAMILY[4:0]** Product Family
The value of this field corresponds to the product family part of the ordering code.

**Bits 21:16 – SERIES[5:0]** Product Series
The value of this field corresponds to the product series part of the ordering code.

**Bits 15:12 – DIE[3:0]** Die Number
Identifies the die family.

**Bits 11:8 – REVISION[3:0]** Revision Number
Identifies the die revision number. 0x0=rev.A, 0x1=rev.B etc.

**Note:** The device variant (last letter of the ordering number) is independent of the die revision (DSU.DID.REVISION): The device variant denotes functional differences, whereas the die revision marks evolution of the die.

**Bits 7:0 – DEVSEL[7:0]** Device Selection
This bit field identifies a device within a product family and product series.

### 13.13.10 CoreSight ROM Table Entry 0

**Name:** ENTRY0
**Offset:** 0x1000
**Reset:** 0xXXXXX00X
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[19:12] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[3:0] | | | | | | | |
| Access | R | R | R | R | | | | |
| Reset | x | x | x | x | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FMT | EPRES |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | x |

**Bits 31:12 – ADDOFF[19:0]** Address Offset
The base address of the component, relative to the base address of this ROM table.

**Bit 1 – FMT** Format
Always reads as '1', indicating a 32-bit ROM table.

**Bit 0 – EPRES** Entry Present
This bit indicates whether an entry is present at this location in the ROM table.

This bit is set at power-up if the device is not protected indicating that the entry is not present.

This bit is cleared at power-up if the device is not protected indicating that the entry is present.

### 13.13.11 CoreSight ROM Table Entry 1

**Name:** ENTRY1
**Offset:** 0x1004
**Reset:** 0xXXXXX00X
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[19:12] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDOFF[3:0] | | | | | | | |
| Access | R | R | R | R | | | | |
| Reset | x | x | x | x | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FMT | EPRES |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | x |

**Bits 31:12 – ADDOFF[19:0]** Address Offset
The base address of the component, relative to the base address of this ROM table.

**Bit 1 – FMT** Format
Always read as '1', indicating a 32-bit ROM table.

**Bit 0 – EPRES** Entry Present
This bit indicates whether an entry is present at this location in the ROM table.

This bit is set at power-up if the device is not protected indicating that the entry is not present.

This bit is cleared at power-up if the device is not protected indicating that the entry is present.

### 13.13.12 CoreSight ROM Table End

**Name:** END
**Offset:** 0x1008
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | END[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | END[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | END[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | END[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – END[31:0]** End Marker
Indicates the end of the CoreSight ROM table entries.

### 13.13.13 CoreSight ROM Table Memory Type

**Name:** MEMTYPE
**Offset:** 0x1FCC
**Reset:** 0x0000000x
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SMEMP |
| Access | | | | | | | | R |
| Reset | | | | | | | | x |

**Bit 0 – SMEMP** System Memory Present
This bit indicates whether system memory is present on the bus that connects to the ROM table.

This bit is set at power-up if the device is not protected, indicating that the system memory is accessible from a debug adapter.

This bit is cleared at power-up if the device is protected, indicating that the system memory is not accessible from a debug adapter.

### 13.13.14 Peripheral Identification 4

**Name:** PID4
**Offset:** 0x1FD0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FKBC[3:0] | | | | JEPCC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:4 – FKBC[3:0]** 4KB Count
These bits will always return zero when read, indicating that this debug component occupies one 4KB block.

**Bits 3:0 – JEPCC[3:0]** JEP-106 Continuation Code
These bits will always return zero when read.

### 13.13.15 Peripheral Identification 0

**Name:** PID0
**Offset:** 0x1FE0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARTNBL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – PARTNBL[7:0]** Part Number Low
These bits will always return 0xD0 when read, indicating that this device implements a DSU module instance.

### 13.13.16 Peripheral Identification 1

**Name:** PID1
**Offset:** 0x1FE4
**Reset:** 0x000000FC
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | JEPIDCL[3:0] | | | | PARTNBH[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |

**Bits 7:4 – JEPIDCL[3:0]** Low part of the JEP-106 Identity Code
These bits will always return 0xF when read (JEP-106 identity code is 0x1F).

**Bits 3:0 – PARTNBH[3:0]** Part Number High
These bits will always return 0xC when read, indicating that this device implements a DSU module instance.

### 13.13.17 Peripheral Identification 2

**Name:** PID2
**Offset:** 0x1FE8
**Reset:** 0x00000009
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REVISION[3:0] | | | | JEPU | JEPIDCH[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bits 7:4 – REVISION[3:0]** Revision Number
Revision of the peripheral. Starts at 0x0 and increments by one at both major and minor revisions.

**Bit 3 – JEPU** JEP-106 Identity Code is used
This bit will always return one when read, indicating that JEP-106 code is used.

**Bits 2:0 – JEPIDCH[2:0]** JEP-106 Identity Code High
These bits will always return 0x1 when read, (JEP-106 identity code is 0x1F).

### 13.13.18 Peripheral Identification 3

**Name:** PID3
**Offset:** 0x1FEC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REVAND[3:0] | | | | CUSMOD[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:4 – REVAND[3:0]** Revision Number
These bits will always return 0x0 when read.

**Bits 3:0 – CUSMOD[3:0]** ARM CUSMOD
These bits will always return 0x0 when read.

### 13.13.19 Component Identification 0

**Name:** CID0
**Offset:** 0x1FF0
**Reset:** 0x0000000D
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PREAMBLEB0[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |

**Bits 7:0 – PREAMBLEB0[7:0]** Preamble Byte 0
These bits will always return 0x0000000D when read.

### 13.13.20 Component Identification 1

**Name:** CID1
**Offset:** 0x1FF4
**Reset:** 0x00000010
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CCLASS[3:0] | | | | PREAMBLE[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

**Bits 7:4 – CCLASS[3:0]** Component Class
These bits will always return 0x1 when read indicating that this ARM CoreSight component is ROM table (refer to the ARM Debug Interface v5 Architecture Specification at http://www.arm.com).

**Bits 3:0 – PREAMBLE[3:0]** Preamble
These bits will always return 0x00 when read.

### 13.13.21 Component Identification 2

**Name:** CID2
**Offset:** 0x1FF8
**Reset:** 0x00000005
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PREAMBLEB2[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

**Bits 7:0 – PREAMBLEB2[7:0]** Preamble Byte 2
These bits will always return 0x00000005 when read.

### 13.13.22 Component Identification 3

**Name:** CID3
**Offset:** 0x1FFC
**Reset:** 0x000000B1
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PREAMBLEB3[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |

**Bits 7:0 – PREAMBLEB3[7:0]** Preamble Byte 3
These bits will always return 0x000000B1 when read.

# 14. Clock System

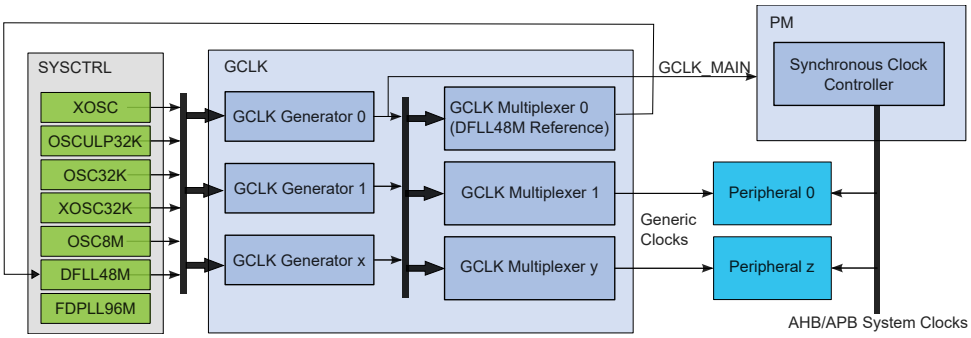
This chapter summarizes the clock distribution and terminology in the SAM D21 device. It will not explain every detail of its configuration. For in-depth documentation, see the respective peripherals descriptions and the *Generic Clock* documentation.

**Related Links**

15. GCLK - Generic Clock Controller

## 14.1 Clock Distribution

**Figure 14-1. Clock distribution**



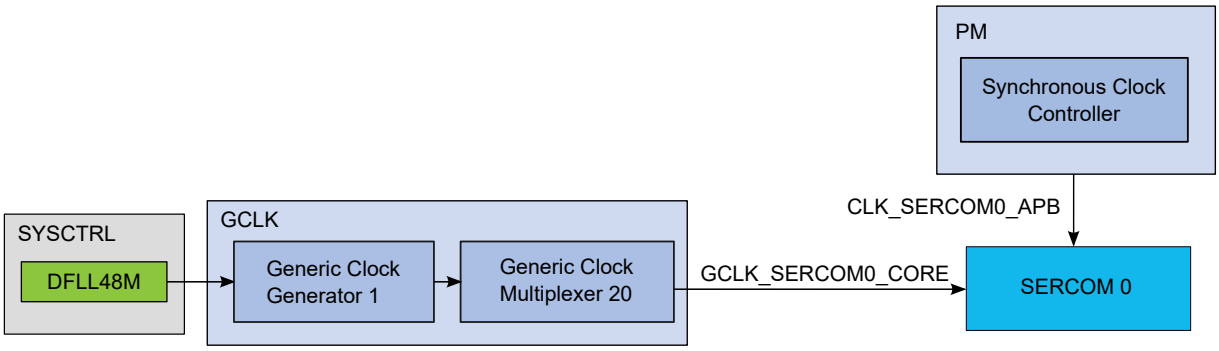

The clock system on the SAM D21 consists of:

- *Clock sources*, controlled by SYSCTRL
  - A clock source provides a time base that is used by other components, such as Generic Clock Generators. Example clock sources are the internal 8MHz oscillator (OSC8M), External crystal oscillator (XOSC) and the Digital frequency locked loop (DFLL48M).
- *Generic Clock Controller (GCLK)* which controls the clock distribution system, made up of:
  - *Generic Clock Generators:* These are programmable prescalers that can use any of the system clock sources as a time base. The Generic Clock Generator 0 generates the clock signal GCLK_MAIN, which is used by the Power Manager, which in turn generates synchronous clocks.
  - *Generic Clocks:* These are clock signals generated by Generic Clock Generators and output by the Generic Clock Multiplexer, and serve as clocks for the peripherals of the system. Multiple instances of a peripheral will typically have a separate Generic Clock for each instance. Generic Clock 0 serves as the clock source for the DFLL48M clock input (when multiplying another clock source).
- Power Manager (PM)
  - The PM generates and controls the synchronous clocks on the system. This includes the CPU, bus clocks (APB, AHB) as well as the synchronous (to the CPU) user interfaces of the peripherals. It contains clock masks that can turn on/off the user interface of a peripheral as well as prescalers for the CPU and bus clocks.

The next figure shows an example where SERCOM0 is clocked by the DFLL48M in open loop mode. The DFLL48M is enabled, the Generic Clock Generator 1 uses the DFLL48M as its clock source and feeds into Peripheral Channel 20. The Generic Clock 20, also called GCLK_SERCOM0_CORE, is connected to

SERCOM0. The SERCOM0 interface, clocked by CLK_SERCOM0_APB, has been unmasked in the APBC Mask register in the PM.

**Figure 14-2. Example of SERCOM clock**




## 14.2 Synchronous and Asynchronous Clocks

As the CPU and the peripherals can be in different clock domains, i.e. they are clocked from different clock sources and/or with different clock speeds, some peripheral accesses by the CPU need to be synchronized. In this case the peripheral includes a SYNCBUSY status register that can be used to check if a sync operation is in progress.

For a general description, see 14.3 Register Synchronization. Some peripherals have specific properties described in their individual sub-chapter “Synchronization”.

In the datasheet, references to Synchronous Clocks are referring to the CPU and bus clocks, while asynchronous clocks are generated by the Generic Clock Controller (GCLK).

## 14.3 Register Synchronization

There are two different register synchronization schemes implemented on this device: *common synchronizer register synchronization* and *distributed synchronizer register synchronization*.

The modules using a common synchronizer register synchronization are: GCLK, WDT, RTC, EIC, TC, ADC, AC and DAC.

The modules adopting a distributed synchronizer register synchronization are: SERCOM USART, SERCOM SPI, SERCOM I2C, I2S, TCC, USB.

### 14.3.1 Common Synchronizer Register Synchronization

#### 14.3.1.1 Overview

All peripherals are composed of one digital bus interface connected to the APB or AHB bus and running from a corresponding clock in the Main Clock domain, and one peripheral core running from the peripheral Generic Clock (GCLK).

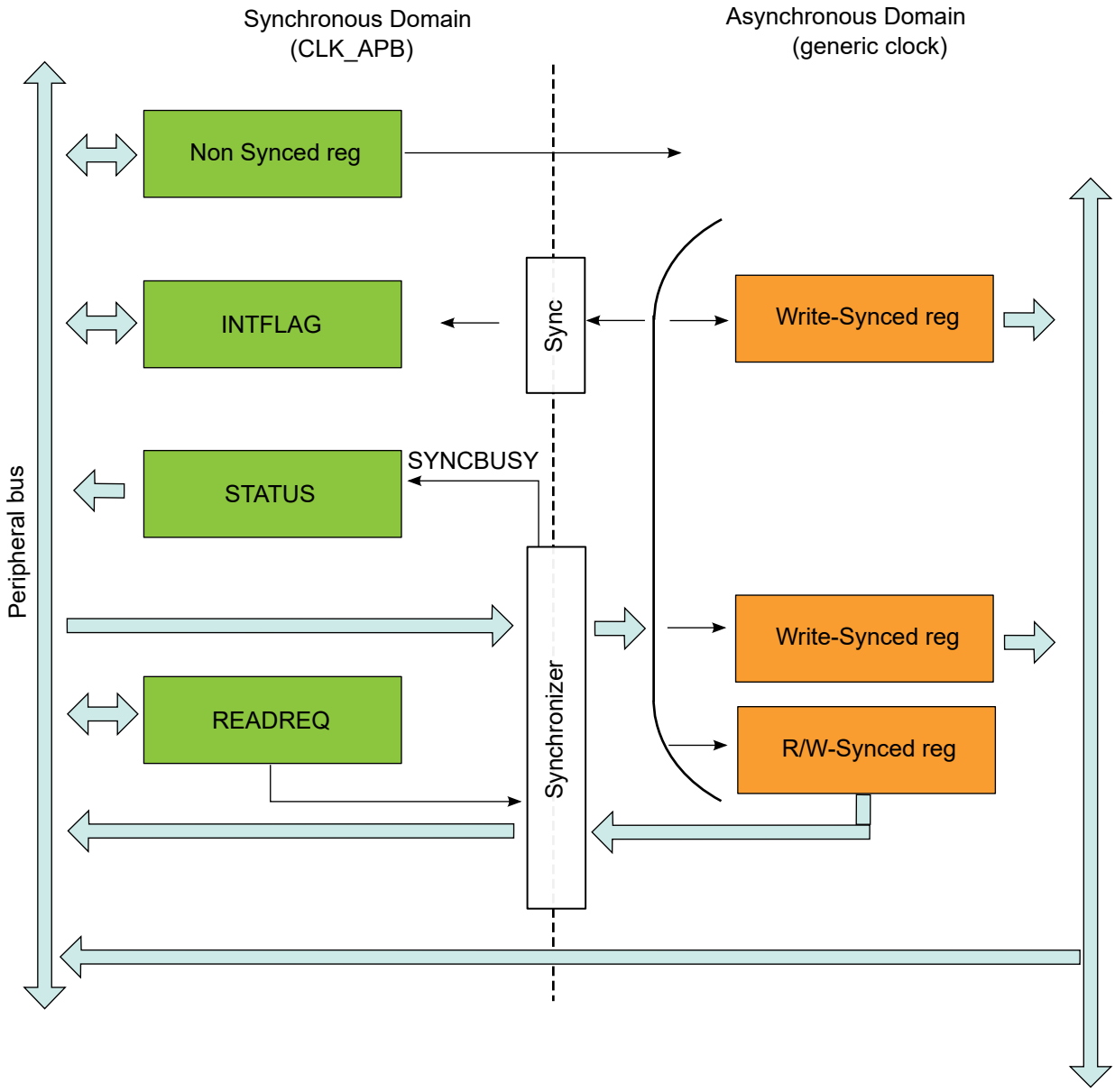
Communication between these clock domains must be synchronized. This mechanism is implemented in hardware, so the synchronization process takes place even if the peripheral generic clock is running from the same clock source and on the same frequency as the bus interface.

All registers in the bus interface are accessible without synchronization. All registers in the peripheral core are synchronized when written. Some registers in the peripheral core are synchronized when read.

Each individual register description will have the properties "Read-Synchronized" and/or "Write-Synchronized" if a register is synchronized.

As shown in the figure below, the common synchronizer is used for all registers in one peripheral. Therefore, status register (STATUS) of each peripheral can be synchronized at a time.

**Figure 14-3. Synchronization**




### 14.3.1.2 Write-Synchronization

Write-Synchronization is triggered by writing to a register in the peripheral clock domain. The Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set when the write-synchronization starts and cleared when the write-synchronization is complete. Refer to 14.3.1.8 Synchronization Delay for details on the synchronization delay.

When the write-synchronization is ongoing (STATUS.SYNCBUSY is one), any of the following actions will cause the peripheral bus to stall until the synchronization is complete:

- Writing a generic clock peripheral core register
- Reading a read-synchronized peripheral core register
- Reading the register that is being written (and thus triggered the synchronization)

Peripheral core registers without read-synchronization will remain static once they have been written and synchronized, and can be read while the synchronization is ongoing without causing the peripheral bus to stall. APB registers can also be read while the synchronization is ongoing without causing the peripheral bus to stall.

#### 14.3.1.3 Read-Synchronization

Reading a read-synchronized peripheral core register will cause the peripheral bus to stall immediately until the read-synchronization is complete. STATUS.SYNCBUSY will not be set. Refer to 14.3.1.8 Synchronization Delay for details on the synchronization delay. Note that reading a read-synchronized peripheral core register while STATUS.SYNCBUSY is one will cause the peripheral bus to stall twice; first because of the ongoing synchronization, and then again because reading a read-synchronized core register will cause the peripheral bus to stall immediately.

#### 14.3.1.4 Completion of synchronization

The user can either poll STATUS.SYNCBUSY or use the Synchronisation Ready interrupt (if available) to check when the synchronization is complete. It is also possible to perform the next read/write operation and wait, as this next operation will be started once the previous write/read operation is synchronized and/or complete.

#### 14.3.1.5 Read Request

The read request functionality is only available to peripherals that have the Read Request register (READREQ) implemented. Refer to the register description of individual peripheral chapters for details.

To avoid forcing the peripheral bus to stall when reading read-synchronized peripheral core registers, the read request mechanism can be used.

**Basic Read Request**

Writing a '1' to the Read Request bit in the Read Request register (READREQ.RREQ) will request read-synchronization of the register specified in the Address bits in READREQ (READREQ.ADDR) and set STATUS.SYNCBUSY. When read-synchronization is complete, STATUS.SYNCBUSY is cleared. The read-synchronized value is then available for reading without delay until READREQ.RREQ is written to '1' again.

The address to use is the offset to the peripheral's base address of the register that should be synchronized.

**Continuous Read Request**

Writing a '1' to the Read Continuously bit in READREQ (READREQ.RCONT) will force continuous read-synchronization of the register specified in READREQ.ADDR. The latest value is always available for reading without stalling the bus, as the synchronization mechanism is continuously synchronizing the given value.

SYNCBUSY is set for the first synchronization, but not for the subsequent synchronizations. If another synchronization is attempted, i.e. by executing a write-operation of a write-synchronized register, the read request will be stopped, and will have to be manually restarted.

**Note:**
The continuous read-synchronization is paused in sleep modes where the generic clock is not running. This means that a new read request is required if the value is needed immediately after exiting sleep.

#### 14.3.1.6 Enable Write-Synchronization

Writing to the Enable bit in the Control register (CTRL.ENABLE) will also trigger write-synchronization and set STATUS.SYNCBUSY. CTRL.ENABLE will read its new value immediately after being written. The Synchronisation Ready interrupt (if available) cannot be used for Enable write-synchronization.

When the enable write-synchronization is ongoing (STATUS.SYNCBUSY is one), attempt to do any of the following will cause the peripheral bus to stall until the enable synchronization is complete:

- Writing a peripheral core register
- Writing an APB register
- Reading a read-synchronized peripheral core register

APB registers can be read while the enable write-synchronization is ongoing without causing the peripheral bus to stall.

#### 14.3.1.7 Software Reset Write-Synchronization

Writing a '1' to the Software Reset bit in CTRL (CTRL.SWRST) will also trigger write-synchronization and set STATUS.SYNCBUSY. When writing a '1' to the CTRL.SWRST bit it will immediately read as '1'. CTRL.SWRST and STATUS.SYNCBUSY will be cleared by hardware when the peripheral has been reset. Writing a zero to the CTRL.SWRST bit has no effect. The Synchronisation Ready interrupt (if available) cannot be used for Software Reset write-synchronization.

When the software reset is in progress (STATUS.SYNCBUSY and CTRL.SWRST are '1'), attempt to do any of the following will cause the peripheral bus to stall until the Software Reset synchronization and the reset is complete:

- Writing a peripheral core register
- Writing an APB register
- Reading a read-synchronized register

APB registers can be read while the software reset is being write-synchronized without causing the peripheral bus to stall.

#### 14.3.1.8 Synchronization Delay

The synchronization will delay write and read accesses by a certain amount. This delay *D* is within the range of:

$$5 \times P_{\mathrm{GCLK}} + 2 \times P_{\mathrm{APB}} < D < 6 \times P_{\mathrm{GCLK}} + 3 \times P_{\mathrm{APB}}$$

Where $P_{\mathrm{GCLK}}$ is the period of the generic clock and $P_{\mathrm{APB}}$ is the period of the peripheral bus clock. A normal peripheral bus register access duration is $2 \times P_{\mathrm{APB}}$.

### 14.3.2 Distributed Synchronizer Register Synchronization

#### 14.3.2.1 Overview

All peripherals are composed of one digital bus interface connected to the APB or AHB bus and running from a corresponding clock in the Main Clock domain, and one peripheral core running from the peripheral Generic Clock (GCLK).

Communication between these clock domains must be synchronized. This mechanism is implemented in hardware, so the synchronization process takes place even if the peripheral generic clock is running from the same clock source and on the same frequency as the bus interface.

All registers in the bus interface are accessible without synchronization. All registers in the peripheral core are synchronized when written. Some registers in the peripheral core are synchronized when read. Registers that need synchronization has this denoted in each individual register description.

#### 14.3.2.2 General Write synchronization

Write-Synchronization is triggered by writing to a register in the peripheral clock domain. The respective bit in the Synchronization Busy register (SYNCBUSY) will be set when the write-synchronization starts and cleared when the write-synchronization is complete. Refer to 14.3.2.7 Synchronization Delay for details on the synchronization delay.

When write-synchronization is ongoing for a register, any subsequent write attempts to this register will be discarded, and an error will be reported.

Example:

REGA, REGB are 8-bit peripheral core registers. REGC is 16-bit peripheral core register.

| Offset | Register |
|---|---|
| 0x00 | REGA |
| 0x01 | REGB |
| 0x02 | REGC |
| 0x03 | |

Synchronization is per register, so multiple registers can be synchronized in parallel. Consequently, after REGA (8-bit access) was written, REGB (8-bit access) can be written immediately without error.

REGC (16-bit access) can be written without affecting REGA or REGB. If REGC is written to in two consecutive 8-bit accesses without waiting for synchronization, the second write attempt will be discarded and an error is generated.

A 32-bit access to offset 0x00 will write all three registers. Note that REGA, REGB and REGC can be updated at different times because of independent write synchronization.

#### 14.3.2.3 General read synchronization

Read-synchronized registers are synchronized when the register value is updated. During synchronization the corresponding bit in SYNCBUSY will be set. Reading a read-synchronized register will return its value immediately and the corresponding bit in SYNCBUSY will not be set.

#### 14.3.2.4 Completion of synchronization

In order to check if synchronization is complete, the user can either poll the relevant bits in SYNCBUSY or use the Synchronisation Ready interrupt (if available). The Synchronization Ready interrupt flag will be set when all ongoing synchronizations are complete, i.e. when all bits in SYNCBUSY are '0'.

#### 14.3.2.5 Enable Write-Synchronization

Setting the Enable bit in a module's Control register (CTRL.ENABLE) will also trigger write-synchronization and set SYNCBUSY.ENABLE. CTRL.ENABLE will read its new value immediately after being written. SYNCBUSY.ENABLE will be cleared by hardware when the operation is complete. The Synchronisation Ready interrupt (if available) cannot be used for Enable write-synchronization.

#### 14.3.2.6 Software Reset Write-Synchronization

Setting the Software Reset bit in CTRLA (CTRLA.SWRST=1) will trigger write-synchronization and set SYNCBUSY.SWRST. When writing a ‘1’ to the CTRLA.SWRST bit it will immediately read as ‘1’. CTRL.SWRST and SYNCBUSY.SWRST will be cleared by hardware when the peripheral has been reset. Writing a '0' to the CTRL.SWRST bit has no effect. The Ready interrupt (if available) cannot be used for Software Reset write-synchronization.

#### 14.3.2.7 Synchronization Delay

The synchronization will delay write and read accesses by a certain amount. This delay *D* is within the range of:

$$5 \times P_{GCLK} + 2 \times P_{APB} < D < 6 \times P_{GCLK} + 3 \times P_{APB}$$

Where $P_{GCLK}$ is the period of the generic clock and $P_{APB}$ is the period of the peripheral bus clock. A normal peripheral bus register access duration is $2 \times P_{APB}$.

## 14.4 Enabling a Peripheral

In order to enable a peripheral that is clocked by a Generic Clock, the following parts of the system needs to be configured:

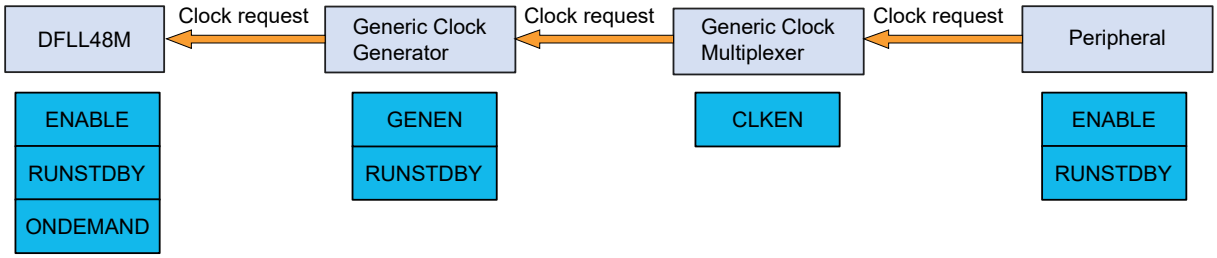
- A running Clock Source.
- A clock from the Generic Clock Generator must be configured to use one of the running Clock Sources, and the Generator must be enabled.
- The Generic Clock Multiplexer that provides the Generic Clock signal to the peripheral must be configured to use a running Generic Clock Generator, and the Generic Clock must be enabled.
- The user interface of the peripheral needs to be unmasked in the PM. If this is not done the peripheral registers will read all 0's and any writing attempts to the peripheral will be discarded.

## 14.5 Disabling a Peripheral

When disabling a peripheral and if a pin change interrupt is enabled on pins driven by the respective peripheral, a wake condition may be generated. If this happen the interrupt flag will not be set. As a consequence the system will not be able to identify the wake source. To avoid this, the interrupt enable register of the peripheral must be cleared (or the Nested Vectored Interrupt Controller (NVIC) Enable for the peripheral must be cleared) before disabling the peripheral.

## 14.6 On-demand, Clock Requests

**Figure 14-4. Clock request routing**




All clock sources in the system can be run in an on-demand mode: the clock source is in a stopped state unless a peripheral is requesting the clock source. Clock requests propagate from the peripheral, via the GCLK, to the clock source. If one or more peripheral is using a clock source, the clock source will be started/kept running. As soon as the clock source is no longer needed and no peripheral has an active request, the clock source will be stopped until requested again.

The clock request can reach the clock source only if the peripheral, the generic clock and the clock from the Generic Clock Generator in-between are enabled. The time taken from a clock request being asserted to the clock source being ready is dependent on the clock source startup time, clock source

frequency as well as the divider used in the Generic Clock Generator. The total startup time Tstart from a clock request until the clock is available for the peripheral is between:

$T_{start_max}$ = Clock source startup time + 2 × clock source periods + 2 × divided clock source periods

$T_{start_min}$ = Clock source startup time + 1 × clock source period + 1 × divided clock source period

The time between the last active clock request stopped and the clock is shut down, $T_{stop}$, is between:

$T_{stop_min}$ = 1 × divided clock source period + 1 × clock source period

$T_{stop_max}$ = 2 × divided clock source periods + 2 × clock source periods

The On-Demand function can be disabled individually for each clock source by clearing the ONDEMAND bit located in each clock source controller. Consequently, the clock will always run whatever the clock request status is. This has the effect of removing the clock source startup time at the cost of power consumption.

The clock request mechanism can be configured to work in standby mode by setting the RUNSDTBY bits of the modules, see Figure 14-4.

## 14.7 Power Consumption vs. Speed

When targeting for either a low-power or a fast acting system, some considerations have to be taken into account due to the nature of the asynchronous clocking of the peripherals:

If clocking a peripheral with a very low clock, the active power consumption of the peripheral will be lower. At the same time the synchronization to the synchronous (CPU) clock domain is dependent on the peripheral clock speed, and will take longer with a slower peripheral clock. This will cause worse response times and longer synchronization delays.

## 14.8 Clocks after Reset

On any reset the synchronous clocks start to their initial state:

- OSC8M is enabled and divided by 8
- Generic Generator 0 uses OSC8M as source and generates GCLK_MAIN
- CPU and BUS clocks are undivided

On a Power Reset, the GCLK module starts to its initial state:

- All Generic Clock Generators are disabled except
  - Generator 0 is using OSC8M as source without division and generates GCLK_MAIN
  - Generator 2 uses OSCULP32K as source without division
- All Generic Clocks are disabled except:
  - WDT Generic Clock uses the Generator 2 as source

On a User Reset the GCLK module starts to its initial state, except for:

- Generic Clocks that are write-locked , i.e., the according WRTLOCK is set to 1 prior to Reset or WDT Generic Clock if the WDT Always-On at power on bit set in the NVM User Row
- Generic Clock is dedicated to the RTC if the RTC Generic Clock is enabled

On any reset the clock sources are reset to their initial state except the 32KHz clock sources which are reset only by a power reset.

# 15. GCLK - Generic Clock Controller

## 15.1 Overview

Depending on the application, peripherals may require specific clock frequencies to operate correctly. The Generic Clock controller GCLK provides nine Generic Clock Generators that can provide a wide range of clock frequencies.

Generators can be set to use different external and internal oscillators as source. The clock of each Generator can be divided. The outputs from the Generators are used as sources for the Generic Clock Multiplexers, which provide the Generic Clock (GCLK_PERIPHERAL) to the peripheral modules, as shown in Generic Clock Controller Block Diagram. The number of Peripheral Clocks depends on how many peripherals the device has.

**Note:** The Generator 0 is always the direct source of the GCLK_MAIN signal.

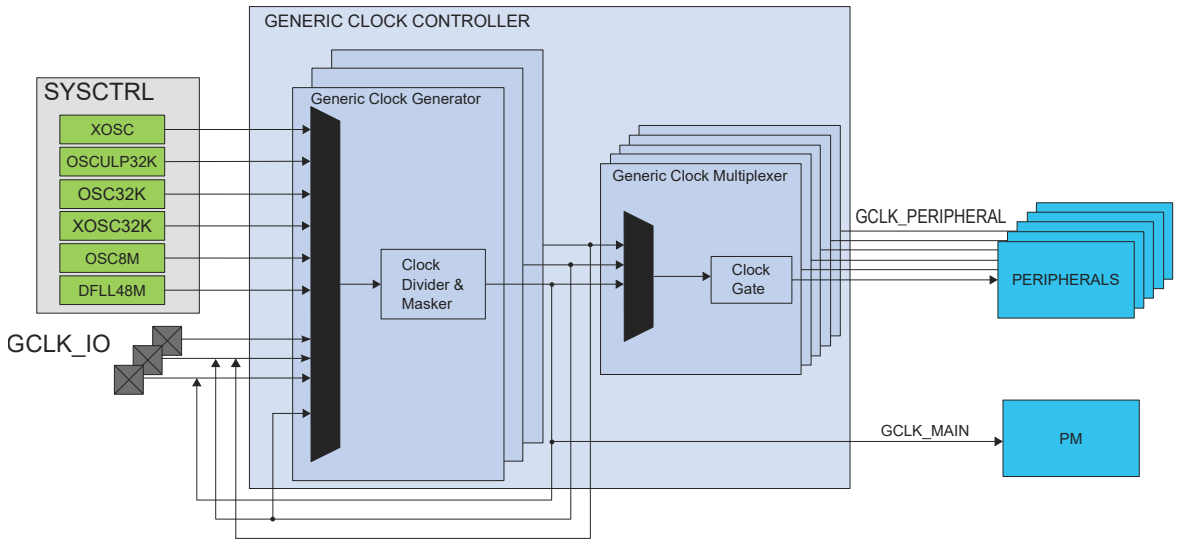
## 15.2 Features

- Provides Generic Clocks
- Wide frequency range
- Clock source for the generator can be changed on the fly

## 15.3 Block Diagram

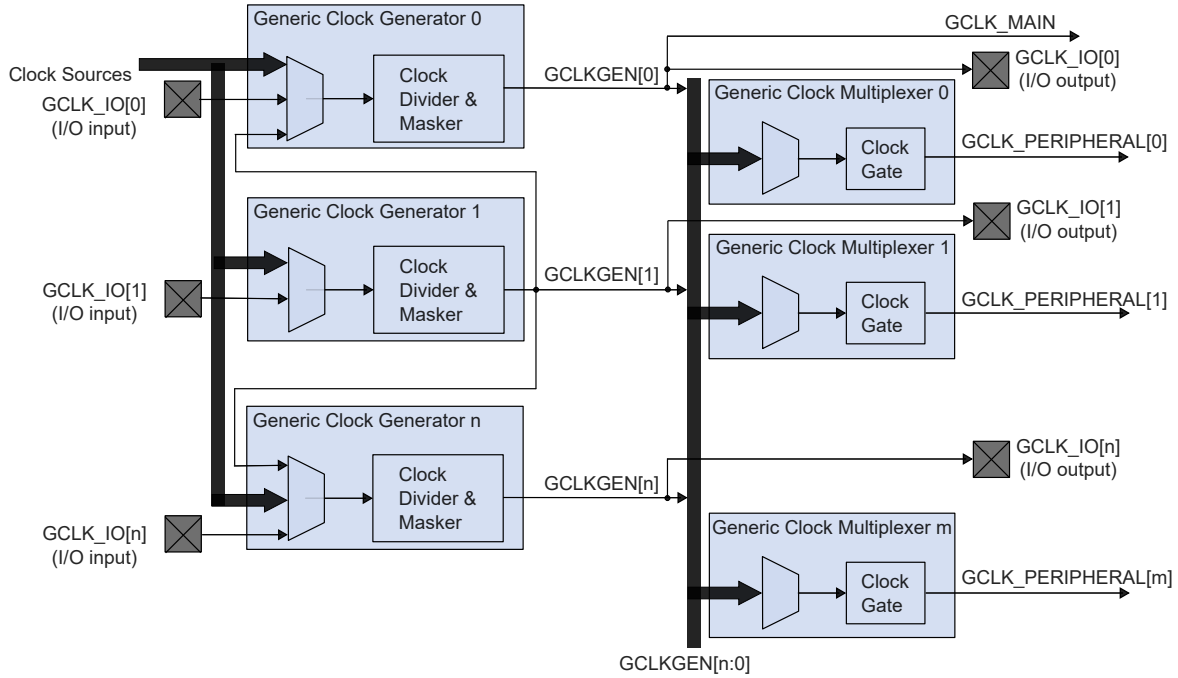
The generation of Peripheral Clock signals (GCLK_PERIPHERAL) and the Main Clock (GCLK_MAIN) can be seen in the figure below.

**Figure 15-1. Device Clocking Diagram**




The GCLK block diagram is shown in the next figure.

**Figure 15-2. Generic Clock Controller Block Diagram(1)**




**Note:** 1. If GENCTRL.SRC=0x01(GCLKIN), the GCLK_IO is set as an input.

## 15.4 Signal Description

**Table 15-1. Signal Description**

| Signal Name | Type | Description |
|---|---|---|
| GCLK_IO[7:0] | Digital I/O | Clock source for Generators when input<br>Generic Clock signal when output |

Refer to PORT Function Multiplexing table in I/O Multiplexing and Considerations for details on the pin mapping for this peripheral.

**Note:** One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 15.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 15.5.1 I/O Lines

Using the GCLK I/O lines requires the I/O pins to be configured.

**Related Links**

23. PORT - I/O Pin Controller

### 15.5.2 Power Management

The GCLK can operate in sleep modes, if required. Refer to the sleep mode description in the Power Manager (PM) section.

**Related Links**

16. PM – Power Manager

### 15.5.3 Clocks

The GCLK bus clock (CLK_GCLK_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_GCLK_APB can be found in the Peripheral Clock Masking section of PM – Power Manager.

**Related Links**

16. PM – Power Manager

### 15.5.4 DMA

Not applicable.

### 15.5.5 Interrupts

Not applicable.

### 15.5.6 Events

Not applicable.

### 15.5.7 Debug Operation

Not applicable.

### 15.5.8 Register Access Protection

All registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC).

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 15.5.9 Analog Connections

Not applicable.

## 15.6 Functional Description

### 15.6.1 Principle of Operation

The GCLK module is comprised of eight Generic Clock Generators (Generators) sourcing m Generic Clock Multiplexers.

A clock source selected as input to a Generator can either be used directly, or it can be prescaled in the Generator. A generator output is used as input to one or more the Generic Clock Multiplexers to provide a peripheral (GCLK_PERIPHERAL). A generic clock can act as the clock to one or several of peripherals.

### 15.6.2 Basic Operation

#### 15.6.2.1 Initialization

Before a Generator is enabled, the corresponding clock source should be enabled. The Peripheral clock must be configured as outlined by the following steps:

1. The Generic Clock Generator division factor must be set by performing a single 32-bit write to the Generic Clock Generator Division register (GENDIV):
   - The Generic Clock Generator that will be selected as the source of the generic clock by setting the ID bit group (GENDIV.ID).
   - The division factor must be selected by the DIV bit group (GENDIV.DIV)
     **Note:** Refer to *Generic Clock Generator Division register (GENDIV)* for details.
2. The generic clock generator must be enabled by performing a single 32-bit write to the Generic Clock Generator Control register (GENCTRL):
   - The Generic Clock Generator will be selected as the source of the generic clock by the ID bit group (GENCTRL.ID)
   - The Generic Clock generator must be enabled (GENCTRL.GENEN=1)
     **Note:** Refer to *Generic Clock Generator Control register (GENCTRL)* for details.
3. The generic clock must be configured by performing a single 16-bit write to the Generic Clock Control register (CLKCTRL):
   - The Generic Clock that will be configured via the ID bit group (CLKCTRL.ID)
   - The Generic Clock Generator used as the source of the generic clock by writing the GEN bit group (CLKCTRL.GEN)
     **Note:** Refer to *Generic Clock Control register (CLKCTRL)* for details.

**Related Links**

15.8.5 GENDIV
15.8.4 GENCTRL
15.8.3 CLKCTRL

#### 15.6.2.2 Enabling, Disabling and Resetting

The GCLK module has no enable/disable bit to enable or disable the whole module.

The GCLK is reset by setting the Software Reset bit in the Control register (CTRL.SWRST) to 1. All registers in the GCLK will be reset to their initial state, except for Generic Clocks Multiplexer and associated Generators that have their Write Lock bit set to 1 (CLKCTRL.WRTLOCK). For further details, refer to 15.6.3.4 Configuration Lock.

#### 15.6.2.3 Generic Clock Generator

Each Generator (GCLK_GEN) can be set to run from one of eight different clock sources except GCLKGEN[1], which can be set to run from one of seven sources. GCLKGEN[1] is the only Generator that can be selected as source to other Generators but can not act as source to itself.

Each generator GCLKGEN[x] can be connected to one specific pin GCLK_IO[x]. The GCLK_IO[x] can be set to act as source to GCLKGEN[x] or GCLK_IO[x] can be set up to output the clock generated by GCLKGEN[x].

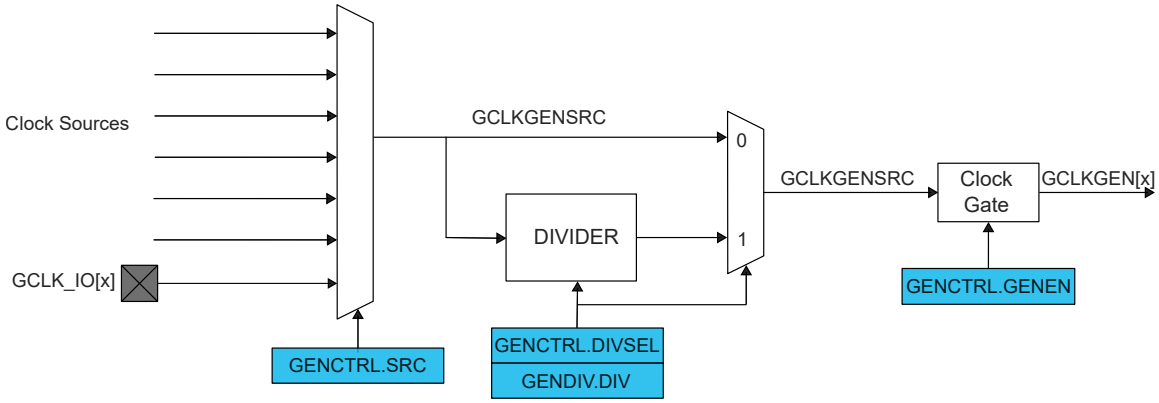
The selected source can be divided. Each Generator can be enabled or disabled independently.

Each GCLKGEN clock signal can then be used as clock source for Generic Clock Multiplexers. Each Generator output is allocated to one or several Peripherals.

GCLKGEN[0], is used as GCLK_MAIN for the synchronous clock controller inside the Power Manager.

Refer to *PM-Power Manager* for details on the synchronous clock generation.

**Figure 15-3. Generic Clock Generator**




**Related Links**

16. PM – Power Manager

#### 15.6.2.4 Enabling a Generic Clock Generator

A Generator is enabled by setting the Generic Clock Generator Enable bit in the Generic Clock Generator Control register (GENCTRL.GENEN=1).

#### 15.6.2.5 Disabling a Generic Clock Generator

A Generator is disabled by clearing GENCTRL.GENEN. When GENCTRL.GENEN=0, the GCLKGEN clock is disabled and clock gated.

#### 15.6.2.6 Selecting a Clock Source for the Generic Clock Generator

Each Generator can individually select a clock source by setting the Source Select bit group in GENCTRL (GENCTRL.SRC).

Changing from one clock source, for example A, to another clock source, B, can be done on the fly: If clock source B is not ready, the Generator will continue running with clock source A. As soon as clock source B is ready, however, the generic clock generator will switch to it. During the switching operation, the Generator holds clock requests to clock sources A and B and then releases the clock source A request when the switch is done.

The available clock sources are device dependent (usually the crystal oscillators, RC oscillators, PLL and DFLL). Only GCLKGEN[1] can be used as a common source for all other generators except Generator 1.

**Note:** Before switching the Generic Clock Generator 0 (GCLKGEN0) from a clock source A to another clock source B, enable the "ONDEMAND" feature of the clock source A to ensure a proper transition from clock source A to clock source B.

#### 15.6.2.7 Changing Clock Frequency

The selected source (GENCLKSRC) for a Generator can be divided by writing a division value in the Division Factor bit group in the Generic Clock Generator Division register (GENDIV.DIV). How the actual division factor is calculated is depending on the Divide Selection bit in GENCTRL (GENCTRL.DIVSEL), it can be interpreted in two ways by the integer divider.

**Note:** The number of DIV bits for each Generator is device dependent.

#### 15.6.2.8 Duty Cycle

When dividing a clock with an odd division factor, the duty-cycle will not be 50/50. Writing the Improve Duty Cycle bit in GENCTRL (GENCTRL.IDC=1) will result in a 50/50 duty cycle.

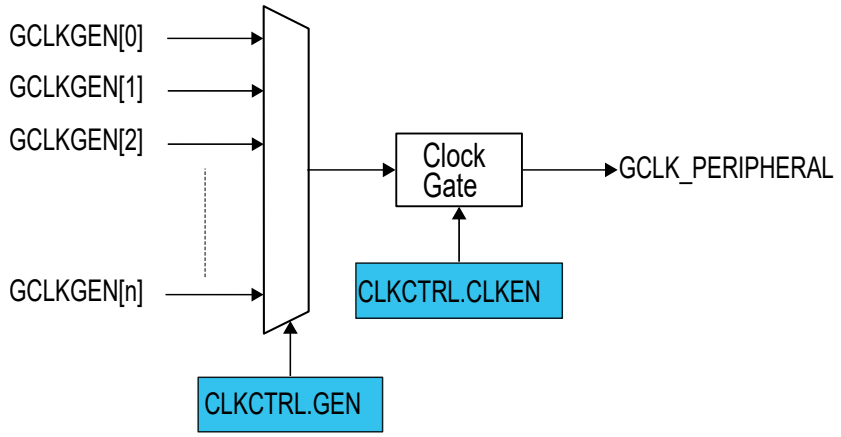
#### 15.6.2.9 Generic Clock Output on I/O Pins

Each Generator's output can be directed to a GCLK_IO pin. If the Output Enable bit in GENCTRL is '1' (GENCTRL.OE=1) and the Generator is enabled (GENCTRL.GENEN=1), the Generator requests its clock source and the GCLKGEN clock is output to a GCLK_IO pin. If GENCTRL.OE=0, GCLK_IO is set according to the Output Off Value bit. If the Output Off Value bit in GENCTRL (GENCTRL.OOV) is zero, the output clock will be low when generic clock generator is turned off. If GENCTRL.OOV=1, the output clock will be high when Generator is turned off.

In standby mode, if the clock is output (GENCTRL.OE=1), the clock on the GCLK_IO pin is frozen to the OOV value if the Run In Standby bit in GENCTRL (GENCTRL.RUNSTDBY) is zero. If GENCTRL.RUNSTDBY=1, the GCLKGEN clock is kept running and output to GCLK_IO.

### 15.6.3 Generic Clock

**Figure 15-4. Generic Clock Multiplexer**




#### 15.6.3.1 Enabling a Generic Clock

Before a generic clock is enabled, one of the Generators must be selected as the source for the generic clock by writing to CLKCTRL.GEN. The clock source selection is individually set for each generic clock.

When a Generator has been selected, the generic clock is enabled by setting the Clock Enable bit in CLKCTRL (CLKCTRL.CLKEN=1). The CLKCTRL.CLKEN bit must be synchronized to the generic clock domain. CLKCTRL.CLKEN will continue to read as its previous state until the synchronization is complete.

#### 15.6.3.2 Disabling a Generic Clock

A generic clock is disabled by writing CLKCTRL.CLKEN=0. The SYNCBUSY bit will be cleared when this write-synchronization is complete. CLKCTRL.CLKEN will stay in its previous state until the synchronization is complete. The generic clock is gated when disabled.

#### 15.6.3.3 Selecting a Clock Source for the Generic Clock

When changing a generic clock source by writing to CLKCTRL.GEN, the generic clock must be disabled before being re-enabled with the new clock source setting. This prevents glitches during the transition:

1. Write CLKCTRL.CLKEN=0
2. Assert that CLKCTRL.CLKEN reads '0'
3. Change the source of the generic clock by writing CLKCTRL.GEN
4. Re-enable the generic clock by writing CLKCTRL.CLKEN=1

#### 15.6.3.4 Configuration Lock

The generic clock configuration can be locked for further write accesses by setting the Write Lock bit in the CLKCTRL register (CLKCTRL.WRTLOCK). All writes to the CLKCTRL register will be ignored. It can only be unlocked by a Power Reset.

The Generator source of a locked generic clock are also locked, too: The corresponding GENCTRL and GENDIV are locked, and can be unlocked only by a Power Reset.

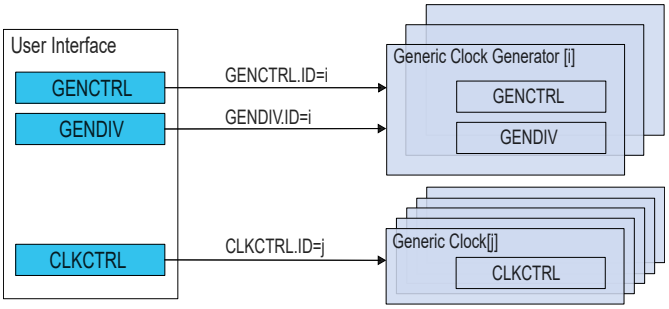
There is one exception concerning the GCLKGEN[0]. As it is used as GCLK_MAIN, it can not be locked. It is reset by any Reset and will start up in a known configuration. The software reset (CTRL.SWRST) can not unlock the registers.

### 15.6.4 Additional Features

#### 15.6.4.1 Indirect Access

The Generic Clock Generator Control and Division registers (GENCTRL and GENDIV) and the Generic Clock Control register (CLKCTRL) are indirectly addressed as shown in the next figure.

**Figure 15-5. GCLK Indirect Access**




Writing these registers is done by setting the corresponding ID bit group. To read a register, the user must write the ID of the channel, i, in the corresponding register. The value of the register for the corresponding ID is available in the user interface by a read access.

For example, the sequence to read the GENCTRL register of generic clock generator i is:

1. Do an 8-bit write of the i value to GENCTRL.ID
2. Read the value of GENCTRL

#### 15.6.4.2 Generic Clock Enable after Reset

The Generic Clock Controller must be able to provide a generic clock to some specific peripherals after a reset. That means that the configuration of the Generators and generic clocks after Reset is device-dependent.

Refer to *GENCTRL.ID* for details on GENCTRL reset.

Refer to *GENDIV.ID* for details on GENDIV reset.

Refer to *CLKCTRL.ID* for details on CLKCTRL reset.

**Related Links**

15.8.5 GENDIV
15.8.4 GENCTRL
15.8.3 CLKCTRL

### 15.6.5 Sleep Mode Operation

#### 15.6.5.1 Sleep Walking

The GCLK module supports the Sleep Walking feature. If the system is in a sleep mode where the Generic Clocks are stopped, a peripheral that needs its clock in order to execute a process must request it from the Generic Clock Controller.

The Generic Clock Controller receives this request, determines which Generic Clock Generator is involved and which clock source needs to be awakened. It then wakes up the respective clock source, enables the Generator and generic clock stages successively, and delivers the clock to the peripheral.

#### 15.6.5.2 Run in Standby Mode

In standby mode, the GCLK can continuously output the generator output to GCLK_IO.

When set, the GCLK can continuously output the generator output to GCLK_IO.

Refer to 15.6.2.9 Generic Clock Output on I/O Pins for details.

### 15.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY=1, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following registers are synchronized when written:

- Generic Clock Generator Control register (GENCTRL)
- Generic Clock Generator Division register (GENDIV)
- Control register (CTRL)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 15.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | | | | | SWRST |
| 0x01 | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x02 | CLKCTRL | 7:0 | | | ID[5:0] | | | | | |
| | | 15:8 | WRTLOCK | CLKEN | | | GEN[3:0] | | | |
| 0x04 | GENCTRL | 7:0 | | | | | ID[3:0] | | | |
| | | 15:8 | | | | SRC[4:0] | | | | |
| | | 23:16 | | | RUNSTDBY | DIVSEL | OE | OOV | IDC | GENEN |
| | | 31:24 | | | | | | | | |
| 0x08 | GENDIV | 7:0 | | | | | ID[3:0] | | | |
| | | 15:8 | DIV[7:0] | | | | | | | |
| | | 23:16 | DIV[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

## 15.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Refer to 15.5.8 Register Access Protection for details.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Refer to 15.6.6 Synchronization for details.

### 15.8.1 Control

**Name:** CTRL
**Offset:** 0x0
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the GCLK to their initial state after a power reset, except for generic clocks and associated generators that have their WRTLOCK bit in CLKCTRL read as one.

Refer to *GENCTRL.ID* for details on GENCTRL reset.

Refer to *GENDIV.ID* for details on GENDIV reset.

Refer to *CLKCTRL.ID* for details on CLKCTRL reset.

Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | There is a reset operation ongoing. |

### 15.8.2 Status

**Name:** STATUS
**Offset:** 0x1
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy Status
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 15.8.3 Generic Clock Control

**Name:** CLKCTRL
**Offset:** 0x2
**Reset:** 0x0000
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WRTLOCK | CLKEN | | | GEN[3:0] | | | |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ID[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – WRTLOCK** Write Lock
When this bit is written, it will lock from further writes the generic clock pointed to by CLKCTRL.ID, the generic clock generator pointed to in CLKCTRL.GEN and the division factor used in the generic clock generator. It can only be unlocked by a power reset.

One exception to this is generic clock generator 0, which cannot be locked.

| Value | Description |
|---|---|
| 0 | The generic clock and the associated generic clock generator and division factor are not locked. |
| 1 | The generic clock and the associated generic clock generator and division factor are locked. |

**Bit 14 – CLKEN** Clock Enable
This bit is used to enable and disable a generic clock.

| Value | Description |
|---|---|
| 0 | The generic clock is disabled. |
| 1 | The generic clock is enabled. |

**Bits 11:8 – GEN[3:0]** Generic Clock Generator

**Table 15-2. Generic Clock Generator**

| GEN[3:0] | Name | Description |
|---|---|---|
| 0x0 | GCLKGEN0 | Generic clock generator 0 |
| 0x1 | GCLKGEN1 | Generic clock generator 1 |
| 0x2 | GCLKGEN2 | Generic clock generator 2 |
| 0x3 | GCLKGEN3 | Generic clock generator 3 |
| 0x4 | GCLKGEN4 | Generic clock generator 4 |
| 0x5 | GCLKGEN5 | Generic clock generator 5 |

| GEN[3:0] | Name | Description |
|---|---|---|
| 0x6 | GCLKGEN6 | Generic clock generator 6 |
| 0x7 | GCLKGEN7 | Generic clock generator 7 |
| 0x8 | GCLKGEN8 | Generic clock generator 8 |
| 0x9-0xF | - | Reserved |

**Bits 5:0 – ID[5:0]** Generic Clock Selection ID
These bits select the generic clock that will be configured. The value of the ID bit group versus module instance is shown in the table below.

A power reset will reset the CLKCTRL register for all IDs, including the RTC. If the WRTLOCK bit of the corresponding ID is zero and the ID is not the RTC, a user reset will reset the CLKCTRL register for this ID.

After a power reset, the reset value of the CLKCTRL register versus module instance is as shown in the next table.

**Table 15-3. Generic Clock Selection ID and CLKCTRL value after Power Reset**

| Module Instance | Reset Value after Power Reset | | |
|---|---|---|---|
| | CLKCTRL.GEN | CLKCTRL.CLKEN | CLKCTRL.WRTLOCK |
| RTC | 0x00 | 0x00 | 0x00 |
| WDT | 0x02 | 0x01 if WDT Enable bit in NVM User Row written to one<br>0x00 if WDT Enable bit in NVM User Row written to zero | 0x01 if WDT Always-On bit in NVM User Row written to one<br>0x00 if WDT Always-On bit in NVM User Row written to zero |
| Others | 0x00 | 0x00 | 0x00 |

After a user reset, the reset value of the CLKCTRL register versus module instance is as shown in the table below.

**Table 15-4. Generic Clock Selection ID and CLKCTRL Value after User Reset**

| Module Instance | Reset Value after a User Reset | | |
|---|---|---|---|
| | CLKCTRL.GEN | CLCTRL.CLKEN | CLKCTRL.WRTLOCK |
| RTC | 0x00 if WRTLOCK=0 and CLKEN=0<br>No change if WRTLOCK=1 or CLKEN=1 | 0x00 if WRTLOCK=0 and CLKEN=0<br>No change if WRTLOCK=1 or CLKEN=1 | No change |
| WDT | 0x02 if WRTLOCK=0<br>No change if WRTLOCK=1 | If WRTLOCK=0<br>0x01 if WDT Enable bit in NVM User Row written to one<br>0x00 if WDT Enable bit in NVM User | No change |

| Module Instance | Reset Value after a User Reset | | |
|---|---|---|---|
| | CLKCTRL.GEN | CLCTRL.CLKEN | CLKCTRL.WRTLOCK |
| | | Row written to zero<br>If WRTLOCK=1 no change | |
| Others | 0x00 if WRTLOCK=0<br>No change if WRTLOCK=1 | 0x00 if WRTLOCK=0<br>No change if WRTLOCK=1 | No change |

| Value | Name | Description |
|---|---|---|
| 0x00 | GCLK_DFLL48M_REF | DFLL48M Reference |
| 0x01 | GCLK_DPLL | FDPLL96M input clock source for reference |
| 0x02 | GCLK_DPLL_32K | FDPLL96M 32kHz clock for FDPLL96M internal lock timer |
| 0x03 | GCLK_WDT | WDT |
| 0x04 | GCLK_RTC | RTC |
| 0x05 | GCLK_EIC | EIC |
| 0x06 | GCLK_USB | USB |
| 0x07 | GCLK_EVSYS_CHANNEL_0 | EVSYS_CHANNEL_0 |
| 0x08 | GCLK_EVSYS_CHANNEL_1 | EVSYS_CHANNEL_1 |
| 0x09 | GCLK_EVSYS_CHANNEL_2 | EVSYS_CHANNEL_2 |
| 0x0A | GCLK_EVSYS_CHANNEL_3 | EVSYS_CHANNEL_3 |
| 0x0B | GCLK_EVSYS_CHANNEL_4 | EVSYS_CHANNEL_4 |
| 0x0C | GCLK_EVSYS_CHANNEL_5 | EVSYS_CHANNEL_5 |
| 0x0D | GCLK_EVSYS_CHANNEL_6 | EVSYS_CHANNEL_6 |
| 0x0E | GCLK_EVSYS_CHANNEL_7 | EVSYS_CHANNEL_7 |
| 0x0F | GCLK_EVSYS_CHANNEL_8 | EVSYS_CHANNEL_8 |
| 0x10 | GCLK_EVSYS_CHANNEL_9 | EVSYS_CHANNEL_9 |
| 0x11 | GCLK_EVSYS_CHANNEL_10 | EVSYS_CHANNEL_10 |
| 0x12 | GCLK_EVSYS_CHANNEL_11 | EVSYS_CHANNEL_11 |
| 0x13 | GCLK_SERCOMx_SLOW | SERCOMx_SLOW |
| 0x14 | GCLK_SERCOM0_CORE | SERCOM0_CORE |
| 0x15 | GCLK_SERCOM1_CORE | SERCOM1_CORE |
| 0x16 | GCLK_SERCOM2_CORE | SERCOM2_CORE |
| 0x17 | GCLK_SERCOM3_CORE | SERCOM3_CORE |
| 0x18 | GCLK_SERCOM4_CORE | SERCOM4_CORE |
| 0x19 | GCLK_SERCOM5_CORE | SERCOM5_CORE |
| 0x1A | GCLK_TCC0, GCLK_TCC1 | TCC0,TCC1 |
| 0x1B | GCLK_TCC2, GCLK_TC3 | TCC2,TC3 |
| 0x1C | GCLK_TC4, GCLK_TC5 | TC4,TC5 |
| 0x1D | GCLK_TC6, GCLK_TC7 | TC6,TC7 |
| 0x1E | GCLK_ADC | ADC |
| 0x1F | GCLK_AC_DIG, GCLK_AC1_DIG | AC_DIG, AC1_DIG |
| 0x20 | GCLK_AC_ANA, GCLK_AC1_ANA | AC_ANA, AC1_ANA |
| 0x21 | GCLK_DAC | DAC |
| 0x22 | GCLK_PTC | PTC |
| 0x23 | GCLK_I2S_0 | I2S_0 |

| Value | Name | Description |
|---|---|---|
| 0x24 | GCLK_I2S_1 | I2S_1 |
| 0x25-0x3F | - | Reserved |

### 15.8.4 Generic Clock Generator Control

**Name:** GENCTRL
**Offset:** 0x4
**Reset:** 0x00000000
**Property:** Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | RUNSTDBY | DIVSEL | OE | OOV | IDC | GENEN |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | SRC[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ID[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 21 – RUNSTDBY** Run in Standby
This bit is used to keep the generic clock generator running when it is configured to be output to its dedicated GCLK_IO pin. If GENCTRL.OE is zero, this bit has no effect and the generic clock generator will only be running if a peripheral requires the clock.

| Value | Description |
|---|---|
| 0 | The generic clock generator is stopped in standby and the GCLK_IO pin state (one or zero) will be dependent on the setting in GENCTRL.OOV. |
| 1 | The generic clock generator is kept running and output to its dedicated GCLK_IO pin during standby mode. |

**Bit 20 – DIVSEL** Divide Selection
This bit is used to decide how the clock source used by the generic clock generator will be divided. If the clock source should not be divided, the DIVSEL bit must be zero and the GENDIV.DIV value for the corresponding generic clock generator must be zero or one.

| Value | Description |
|---|---|
| 0 | The generic clock generator equals the clock source divided by GENDIV.DIV. |
| 1 | The generic clock generator equals the clock source divided by 2^(GENDIV.DIV+1). |

**Bit 19 – OE** Output Enable
This bit is used to enable output of the generated clock to GCLK_IO when GCLK_IO is not selected as a source in the GENCLK.SRC bit group.

| Value | Description |
|---|---|
| 0 | The generic clock generator is not output. |
| 1 | The generic clock generator is output to the corresponding GCLK_IO, unless the corresponding GCLK_IO is selected as a source in the GENCLK.SRC bit group. |

**Bit 18 – OOV** Output Off Value
This bit is used to control the value of GCLK_IO when GCLK_IO is not selected as a source in the GENCLK.SRC bit group.

| Value | Description |
|---|---|
| 0 | The GCLK_IO will be zero when the generic clock generator is turned off or when the OE bit is zero. |
| 1 | The GCLK_IO will be one when the generic clock generator is turned off or when the OE bit is zero. |

**Bit 17 – IDC** Improve Duty Cycle
This bit is used to improve the duty cycle of the generic clock generator when odd division factors are used.

| Value | Description |
|---|---|
| 0 | The generic clock generator duty cycle is not 50/50 for odd division factors. |
| 1 | The generic clock generator duty cycle is 50/50. |

**Bit 16 – GENEN** Generic Clock Generator Enable
This bit is used to enable and disable the generic clock generator.

| Value | Description |
|---|---|
| 0 | The generic clock generator is disabled. |
| 1 | The generic clock generator is enabled. |

**Bits 12:8 – SRC[4:0]** Source Select
These bits define the clock source to be used as the source for the generic clock generator, as shown in the table below.

| Value | Name | Description |
|---|---|---|
| 0x00 | XOSC | XOSC oscillator output |
| 0x01 | GCLKIN | Generator input pad |
| 0x02 | GCLKGEN1 | Generic clock generator 1 output |
| 0x03 | OSCULP32K | OSCULP32K oscillator output |
| 0x04 | OSC32K | OSC32K oscillator output |
| 0x05 | XOSC32K | XOSC32K oscillator output |
| 0x06 | OSC8M | OSC8M oscillator output |
| 0x07 | DFLL48M | DFLL48M output |
| 0x08 | FDPLL96M | FDPLL96M output |
| 0x09-0x1F | Reserved | Reserved for future use |

**Bits 3:0 – ID[3:0]** Generic Clock Generator Selection
These bits select the generic clock generator that will be configured or read. The value of the ID bit group versus which generic clock generator is configured is shown in the next table.

A power reset will reset the GENCTRL register for all IDs, including the generic clock generator used by the RTC. If a generic clock generator ID other than generic clock generator 0 is not a source of a “locked” generic clock or a source of the RTC generic clock, a user reset will reset the GENCTRL for this ID.

After a power reset, the reset value of the GENCTRL register is as shown in the next table.

| GCLK Generator ID | Reset Value after a Power Reset |
|---|---|
| 0x00 | 0x00010600 |
| 0x01 | 0x00000001 |
| 0x02 | 0x00010302 |
| 0x03 | 0x00000003 |
| 0x04 | 0x00000004 |
| 0x05 | 0x00000005 |
| 0x06 | 0x00000006 |
| 0x07 | 0x00000007 |
| 0x08 | 0x00000008 |

After a user reset, the reset value of the GENCTRL register is as shown in the table below.

| GCLK Generator ID | Reset Value after a User Reset |
|---|---|
| 0x00 | 0x00010600 |
| 0x01 | 0x00000001 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x02 | 0x00010302 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x03 | 0x00000003 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x04 | 0x00000004 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |

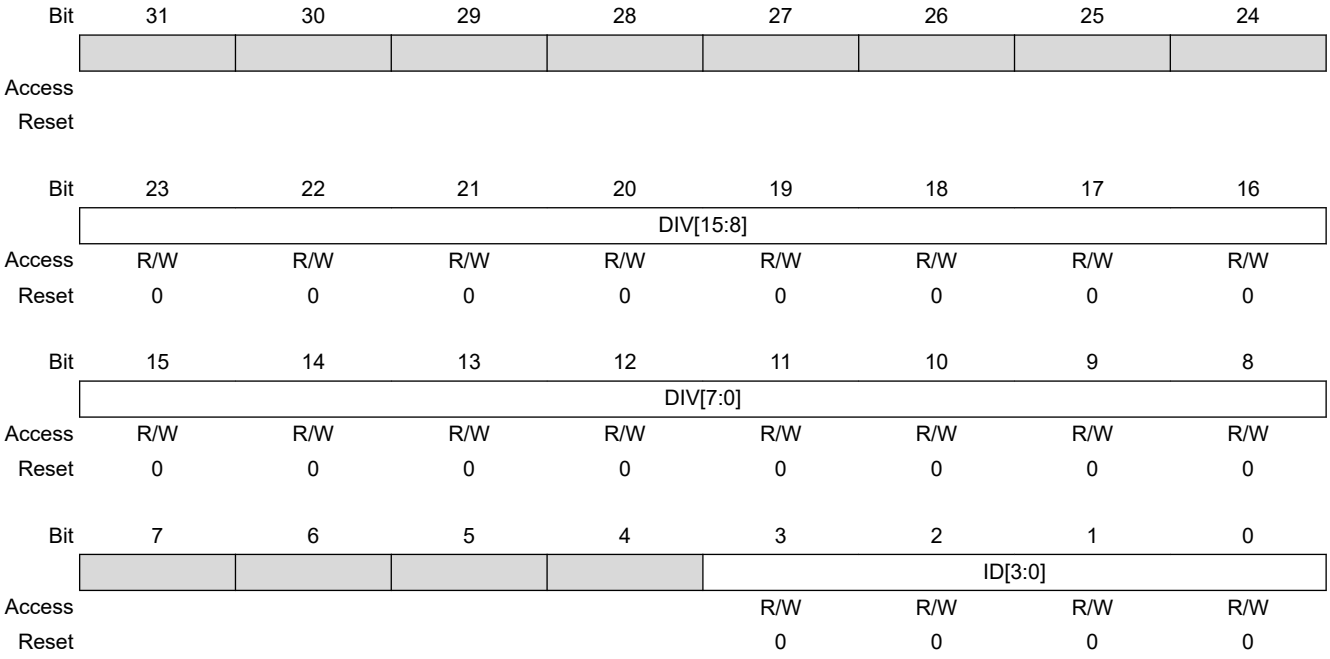
© 2018 Microchip Technology Inc.

| GCLK Generator ID | Reset Value after a User Reset |
|---|---|
| 0x05 | 0x00000005 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x06 | 0x00000006 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x07 | 0x00000007 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x08 | 0x00000008 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |

| Value | Name | Description |
|---|---|---|
| 0x0 | GCLKGEN0 | Generic clock generator 0 |
| 0x1 | GCLKGEN1 | Generic clock generator 1 |
| 0x2 | GCLKGEN2 | Generic clock generator 2 |
| 0x3 | GCLKGEN3 | Generic clock generator 3 |
| 0x4 | GCLKGEN4 | Generic clock generator 4 |
| 0x5 | GCLKGEN5 | Generic clock generator 5 |
| 0x6 | GCLKGEN6 | Generic clock generator 6 |
| 0x7 | GCLKGEN7 | Generic clock generator 7 |
| 0x8 | GCLKGEN8 | Generic clock generator 8 |
| 0x9-0xF | Reserved | |

### 15.8.5 Generic Clock Generator Division

**Name:** GENDIV
**Offset:** 0x8
**Reset:** 0x00000000
**Property:** -




**Bits 23:8 – DIV[15:0]** Division Factor
These bits apply a division on each selected generic clock generator. The number of DIV bits each generator has can be seen in the next table. Writes to bits above the specified number will be ignored.

| Generator | Division Factor Bits |
|---|---|
| Generic clock generator 0 | 8 division factor bits - DIV[7:0] |
| Generic clock generator 1 | 16 division factor bits - DIV[15:0] |
| Generic clock generators 2 | 5 division factor bits - DIV[4:0] |
| Generic clock generators 3 - 8 | 8 division factor bits - DIV[7:0] |

**Bits 3:0 – ID[3:0]** Generic Clock Generator Selection
These bits select the generic clock generator on which the division factor will be applied, as shown in the table below.

| Values | Description |
|---|---|
| 0x0 | Generic clock generator 0 |
| 0x1 | Generic clock generator 1 |
| 0x2 | Generic clock generator 2 |

| Values | Description |
|---|---|
| 0x3 | Generic clock generator 3 |
| 0x4 | Generic clock generator 4 |
| 0x5 | Generic clock generator 5 |
| 0x6 | Generic clock generator 6 |
| 0x7 | Generic clock generator 7 |
| 0x8 | Generic clock generator 8 |
| 0x9-0xF | Reserved |

A power reset will reset the GENDIV register for all IDs, including the generic clock generator used by the RTC. If a generic clock generator ID other than generic clock generator 0 is not a source of a ,“locked” generic clock or a source of the RTC generic clock, a user reset will reset the GENDIV for this ID.

After a power reset, the reset value of the GENDIV register is as shown in the next table.

| GCLK Generator ID | Reset Value after a Power Reset |
|---|---|
| 0x00 | 0x00000000 |
| 0x01 | 0x00000001 |
| 0x02 | 0x00000002 |
| 0x03 | 0x00000003 |
| 0x04 | 0x00000004 |
| 0x05 | 0x00000005 |
| 0x06 | 0x00000006 |
| 0x07 | 0x00000007 |
| 0x08 | 0x00000008 |

After a user reset, the reset value of the GENDIV register is as shown in next table.

| GCLK Generator ID | Reset Value after a User Reset |
|---|---|
| 0x00 | 0x00000000 |
| 0x01 | 0x00000001 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x02 | 0x00000002 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |

| GCLK Generator ID | Reset Value after a User Reset |
|---|---|
| 0x03 | 0x00000003 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x04 | 0x00000004 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x05 | 0x00000005 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x06 | 0x00000006 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x07 | 0x00000007 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |
| 0x08 | 0x00000008 if the generator is not used by the RTC and not a source of a 'locked' generic clock<br>No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one |

# 16. PM – Power Manager

## 16.1 Overview

The Power Manager (PM) controls the reset, clock generation and sleep modes of the device.

Utilizing a main clock chosen from a large number of clock sources from the GCLK, the clock controller provides synchronous system clocks to the CPU and the modules connected to the AHB and the APBx bus. The synchronous system clocks are divided into a number of clock domains; one for the CPU and AHB and one for each APBx. Any synchronous system clock can be changed at run-time during normal operation. The clock domains can run at different speeds, enabling the user to save power by running peripherals at a relatively low clock frequency, while maintaining high CPU performance. In addition, the clock can be masked for individual modules, enabling the user to minimize power consumption.

Before entering the Stand-by Sleep mode the user must make sure that a significant amount of clocks and peripherals are disabled, so that the voltage regulator is not overloaded. This is because during Stand-by Sleep mode the internal voltage regulator will be in Low-Power mode.

Various sleep modes are provided in order to fit power consumption requirements. This enables the PM to stop unused modules in order to save power. In active mode, the CPU is executing application code. When the device enters a Sleep mode, program execution is stopped and some modules and clock domains are automatically switched off by the PM according to the Sleep mode. The application code decides which Sleep mode to enter and when. Interrupts from enabled peripherals and all enabled reset sources can restore the device from a Sleep mode to Active mode.

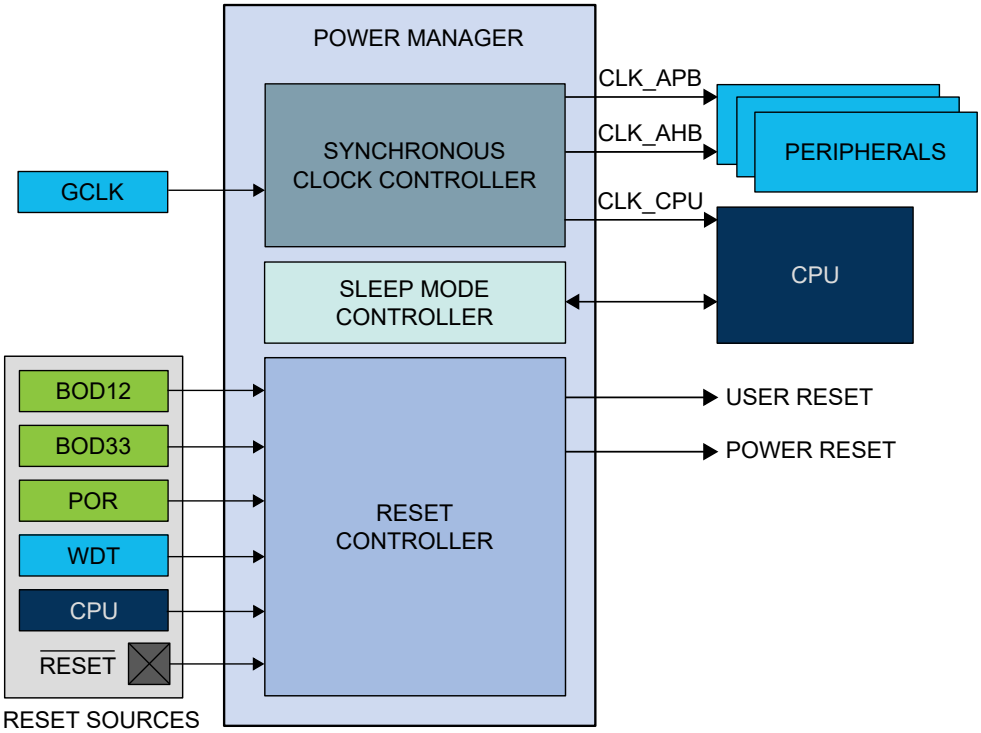
The PM also contains a reset controller to collect all possible reset sources. It issues a device reset and sets the device to its initial state, and allows the reset source to be identified by software.

## 16.2 Features

- Reset control
  - Reset the microcontroller and set it to an initial state according to the reset source
  - Multiple reset sources
    - Power reset sources: POR, BOD12, BOD33
    - User reset sources: External reset ($\overline{\text{RESET}}$), Watchdog Timer reset, software reset
  - Reset status register for reading the reset source from the application code
- Clock control
  - Controls CPU, AHB and APB system clocks
    - Multiple clock sources and division factor from GCLK
    - Clock prescaler with 1x to 128x division
  - Safe run-time clock switching from GCLK
  - Module-level clock gating through maskable peripheral clocks
- Power management control
  - Sleep modes: IDLE, STANDBY
  - SleepWalking support on GCLK clocks

## 16.3 Block Diagram

**Figure 16-1. PM Block Diagram**




## 16.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| $\overline{\text{RESET}}$ | Digital input | External reset |

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 16.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 16.5.1 I/O Lines

Not applicable.

### 16.5.2 Power Management

Not applicable.

### 16.5.3 Clocks

The PM bus clock (CLK_PM_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_PM_APB can be found in *Peripheral Clock Default State* table in the *Peripheral Clock Masking* section. If this clock is disabled in the Power Manager, it can only be re-enabled by a reset.

A generic clock (GCLK_MAIN) is required to generate the main clock. The clock source for GCLK_MAIN is configured by default in the Generic Clock Controller, and can be reconfigured by the user if needed. Refer to *GCLK – Generic Clock Controller* for details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

15. GCLK - Generic Clock Controller

#### 16.5.3.1 Main Clock

The main clock (CLK_MAIN) is the common source for the synchronous clocks. This is fed into the common 8-bit prescaler that is used to generate synchronous clocks to the CPU, AHB and APBx modules.

#### 16.5.3.2 CPU Clock

The CPU clock (CLK_CPU) is routed to the CPU. Halting the CPU clock inhibits the CPU from executing instructions.

#### 16.5.3.3 AHB Clock

The AHB clock (CLK_AHB) is the root clock source used by peripherals requiring an AHB clock. The AHB clock is always synchronous to the CPU clock and has the same frequency, but may run even when the CPU clock is turned off. A clock gate is inserted from the common AHB clock to any AHB clock of a peripheral.

#### 16.5.3.4 APBx Clocks

The APBx clock (CLK_APBX) is the root clock source used by modules requiring a clock on the APBx bus. The APBx clock is always synchronous to the CPU clock, but can be divided by a prescaler, and will run even when the CPU clock is turned off. A clock gater is inserted from the common APB clock to any APBx clock of a module on APBx bus.

### 16.5.4 DMA

Not applicable.

### 16.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the PM interrupt requires the Interrupt Controller to be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 16.5.6 Events

Not applicable.

### 16.5.7 Debug Operation

When the CPU is halted in debug mode, the PM continues normal operation. In sleep mode, the clocks generated from the PM are kept running to allow the debugger accessing any modules. As a consequence, power measurements are not possible in debug mode.

### 16.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag register (INTFLAG).
- Reset Cause register (RCAUSE).

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger. Refer to *PAC – Peripheral Access Controller* for details.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 16.5.9 Analog Connections

Not applicable.

## 16.6 Functional Description

### 16.6.1 Principle of Operation

#### 16.6.1.1 Synchronous Clocks

The GCLK_MAIN clock from GCLK module provides the source for the main clock, which is the common root for the synchronous clocks for the CPU and APBx modules. The main clock is divided by an 8-bit prescaler, and each of the derived clocks can run from any tapping off this prescaler or the undivided main clock, as long as $f_{CPU} \geq f_{APBx}$. The synchronous clock source can be changed on the fly to respond to varying load in the application. The clocks for each module in each synchronous clock domain can be individually masked to avoid power consumption in inactive modules. Depending on the sleep mode, some clock domains can be turned off (see Table 16-4).

#### 16.6.1.2 Reset Controller

The Reset Controller collects the various reset sources and generates reset for the device. The device contains a Power-on-Reset (POR) detector, which keeps the system reset until power is stable. This eliminates the need for external reset circuitry to guarantee stable operation when powering up the device.

#### 16.6.1.3 Sleep Mode Controller

In Active mode, all clock domains are active, allowing software execution and peripheral operation. The PM Sleep Mode Controller allows the user to choose between different sleep modes depending on application requirements, to save power (see Table 16-4).

### 16.6.2 Basic Operation

#### 16.6.2.1 Initialization

After a Power-on Reset (POR), the PM is enabled and the Reset Cause register indicates the POR source (RCAUSE.POR). The default clock source of the GCLK_MAIN clock is started and calibrated before the CPU starts running. The GCLK_MAIN clock is selected as the main clock without any division on the prescaler. The device is in the Active mode.

By default, only the necessary clocks are enabled (see Table 16-1).

#### 16.6.2.2 Enabling, Disabling and Resetting

The PM module is always enabled and can not be reset.

#### 16.6.2.3 Selecting the Main Clock Source

Refer to *GCLK – Generic Clock Controller* for details on how to configure the main clock source.

**Related Links**

15. GCLK - Generic Clock Controller

#### 16.6.2.4 Selecting the Synchronous Clock Division Ratio

The main clock feeds an 8-bit prescaler, which can be used to generate the synchronous clocks. By default, the synchronous clocks run on the undivided main clock. The user can select a prescaler division for the CPU clock by writing the CPU Prescaler Selection bits in the CPU Select register (CPUSEL.CPUDIV), resulting in a CPU clock frequency determined by this equation:

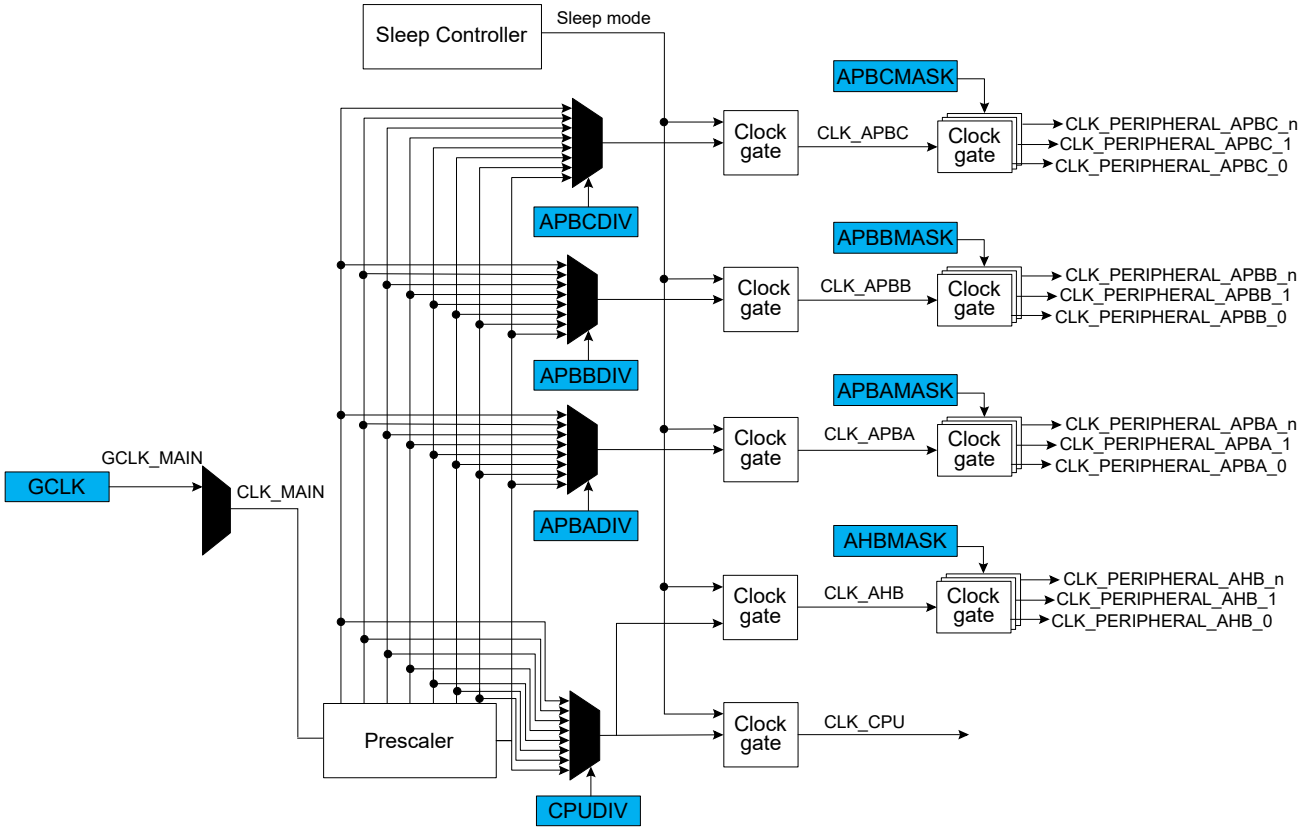
$$f_{\mathrm{CPU}} = \frac{f_{\mathrm{main}}}{2^{\mathrm{CPUDIV}}}$$

Similarly, the clock for the APBx can be divided by writing their respective registers (APBxSEL.APBxDIV). To ensure correct operation, frequencies must be selected so that $f_{CPU} \geq f_{APBx}$. Also, frequencies must never exceed the specified maximum frequency for each clock domain.

**Note:** The AHB clock is always equal to the CPU clock.

CPUSEL and APBxSEL can be written without halting or disabling peripheral modules. Writing CPUSEL and APBxSEL allows a new clock setting to be written to all synchronous clocks at the same time. It is possible to keep one or more clocks unchanged. This way, it is possible to, for example, scale the CPU speed according to the required performance, while keeping the APBx frequency constant.

**Figure 16-2. Synchronous Clock Selection and Prescaler**




#### 16.6.2.5 Clock Ready Flag

There is a slight delay from when CPUSEL and APBxSEL are written until the new clock setting becomes effective. During this interval, the Clock Ready flag in the Interrupt Flag Status and Clear register (INTFLAG.CKRDY) will read as zero. If CKRDY in the INTENSET register is written to one, the Power Manager interrupt can be triggered when the new clock setting is effective. CPUSEL must not be re-written while CKRDY is zero, or the system may become unstable or hang.

#### 16.6.2.6 Peripheral Clock Masking

It is possible to disable or enable the clock for a peripheral in the AHB or APBx clock domain by writing the corresponding bit in the Clock Mask register (APBxMASK) to zero or one. Refer to the table below for the default state of each of the peripheral clocks.

**Table 16-1. Peripheral Clock Default State**

| Peripheral Clock | Default State |
|---|---|
| CLK_PAC0_APB | Enabled |
| CLK_PM_APB | Enabled |
| CLK_SYSCTRL_APB | Enabled |
| CLK_GCLK_APB | Enabled |
| CLK_WDT_APB | Enabled |
| CLK_RTC_APB | Enabled |

| Peripheral Clock | Default State |
|---|---|
| CLK_EIC_APB | Enabled |
| CLK_PAC1_APB | Enabled |
| CLK_DSU_APB | Enabled |
| CLK_NVMCTRL_APB | Enabled |
| CLK_PORT_APB | Enabled |
| CLK_HMATRIX_APB | Enabled |
| CLK_PAC2_APB | Disabled |
| CLK_SERCOMx_APB | Disabled |
| CLK_TCx_APB | Disabled |
| CLK_ADC_APB | Enabled |
| CLK_ACx_APB | Disabled |
| CLK_DAC_APB | Disabled |
| CLK_PTC_APB | Disabled |
| CLK_USB_APB | Enabled |
| CLK_DMAC_APB | Enabled |
| CLK_TCCx_APB | Disabled |
| CLK_I2S_APB | Disabled |

When the APB clock for a module is not provided its registers cannot be read or written. The module can be re-enabled later by writing the corresponding mask bit to one.

A module may be connected to several clock domains (for instance, AHB and APB), in which case it will have several mask bits.

**Note:** Clocks should only be switched off if it is certain that the module will not be used. Switching off the clock for the NVM Controller (NVMCTRL) will cause a problem if the CPU needs to read from the flash memory. Switching off the clock to the Power Manager (PM), which contains the mask registers, or the corresponding APBx bridge, will make it impossible to write the mask registers again. In this case, they can only be re-enabled by a system reset.

#### 16.6.2.7 Reset Controller

The latest reset cause is available in RCAUSE, and can be read during the application boot sequence in order to determine proper action.

There are two groups of reset sources:

- Power Reset: Resets caused by an electrical issue.
- User Reset: Resets caused by the application.

The table below lists the parts of the device that are reset, depending on the reset type.

**Table 16-2. Effects of the Different Reset Events**

| | Power Reset | User Reset | |
|---|---|---|---|
| | POR, BOD12, BOD33 | External Reset | WDT Reset, SysResetReq |
| RTC<br>All the 32kHz sources<br>WDT with ALWAYSON feature<br>Generic Clock with WRTLOCK feature | Y | N | N |
| Debug logic | Y | Y | N |
| Others | Y | Y | Y |

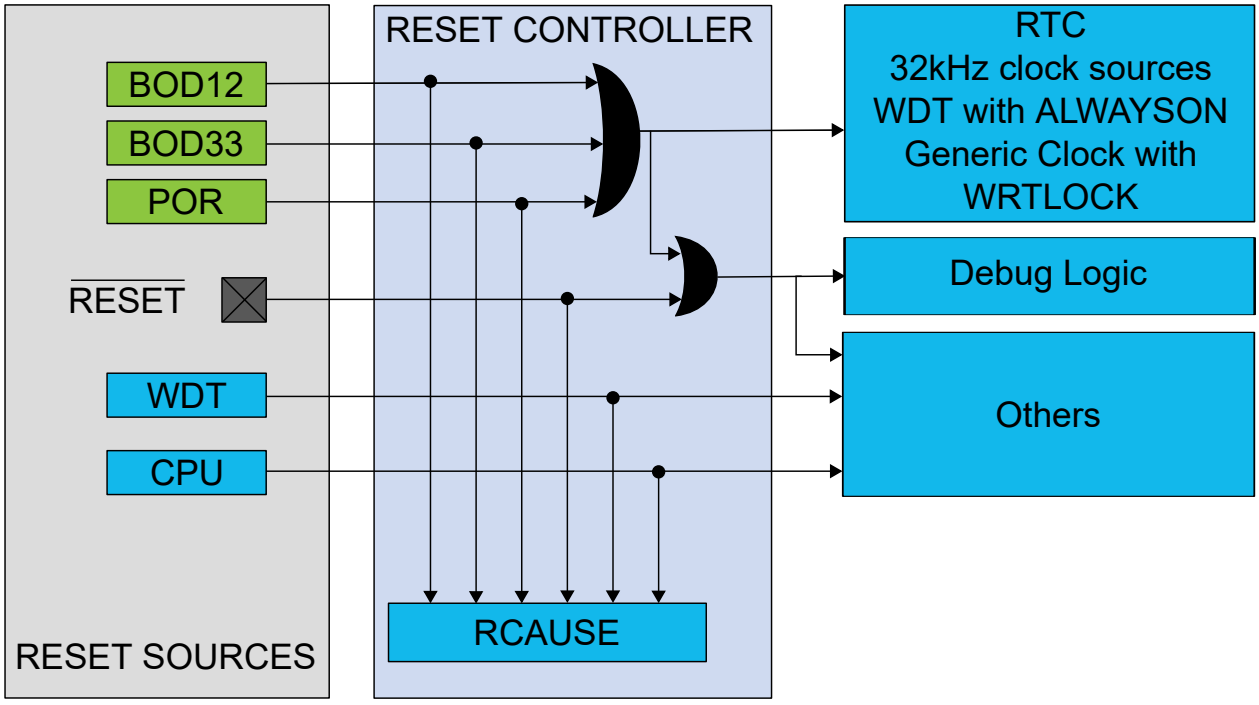
The external reset is generated when pulling the $\overline{\text{RESET}}$ pin low. This pin has an internal pull-up, and does not need to be driven externally during normal operation.

The POR, BOD12 and BOD33 reset sources are generated by their corresponding module in the System Controller Interface (SYSCTRL).

The WDT reset is generated by the Watchdog Timer.

The System Reset Request (SysResetReq) is a software reset generated by the CPU when asserting the SYSRESETREQ bit located in the Reset Control register of the CPU (See the ARM® Cortex® Technical Reference Manual on http://www.arm.com).

**Figure 16-3. Reset Controller**



#### 16.6.2.8 Sleep Mode Controller

Sleep mode is activated by the Wait For Interrupt instruction (WFI). The Idle bits in the Sleep Mode register (SLEEP.IDLE) and the SLEEPDEEP bit of the System Control register of the CPU should be used as argument to select the level of the sleep mode.

There are two main types of sleep mode:

- IDLE mode: The CPU is stopped. Optionally, some synchronous clock domains are stopped, depending on the IDLE argument. Regulator operates in normal mode.
- STANDBY mode: All clock sources are stopped, except those where the RUNSTDBY bit is set. Regulator operates in low-power mode. Before entering standby mode the user must make sure that a significant amount of clocks and peripherals are disabled, so that the voltage regulator is not overloaded.

**Table 16-3. Sleep Mode Entry and Exit Table**

| Mode | Level | Mode Entry | Wake-Up Sources |
|---|---|---|---|
| IDLE | 0 | SCR.SLEEPDEEP = 0<br>SLEEP.IDLE=Level<br>WFI | Synchronous(2) (APB, AHB), asynchronous(1) |
| | 1 | | Synchronous (APB), asynchronous |
| | 2 | | Asynchronous |
| STANDBY | | SCR.SLEEPDEEP = 1<br>WFI | Asynchronous |

**Note:**

1. Asynchronous: interrupt generated on generic clock or external clock or external event.
2. Synchronous: interrupt generated on the APB clock.

**Table 16-4. Sleep Mode Overview**

| Sleep Mode | CPU Clock | AHB Clock | APB Clock | Oscillators | | | | Main Clock | Regulator Mode | RAM Mode |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | ONDEMAND = 0 | | ONDEMAND = 1 | | | | |
| | | | | RUNSTDBY=0 | RUNSTDBY=1 | RUNSTDBY=0 | RUNSTDBY=1 | | | |
| Idle 0 | Stop | Run | Run | Run | Run | Run if requested | Run if requested | Run | Normal | Normal |
| Idle 1 | Stop | Stop | Run | Run | Run | Run if requested | Run if requested | Run | Normal | Normal |
| Idle 2 | Stop | Stop | Stop | Run | Run | Run if requested | Run if requested | Run | Normal | Normal |
| Standby | Stop | Stop | Stop | Stop | Run | Stop | Run if requested | Stop | Low power | Low power |

##### 16.6.2.8.1 IDLE Mode

The IDLE modes allow power optimization with the fastest wake-up time.

The CPU is stopped. To further reduce power consumption, the user can disable the clocking of modules and clock sources by configuring the SLEEP.IDLE bit group. The module will be halted regardless of the bit settings of the mask registers in the Power Manager (PM.AHBMASK, PM.APBxMASK).

Regulator operates in normal mode.

- Entering IDLE mode: The IDLE mode is entered by executing the WFI instruction. Additionally, if the SLEEPONEXIT bit in the ARM Cortex System Control register (SCR) is set, the IDLE mode will also be entered when the CPU exits the lowest priority ISR. This mechanism can be useful for applications that only require the processor to run when an interrupt occurs. Before entering the

IDLE mode, the user must configure the IDLE mode configuration bit group and must write a zero to the SCR.SLEEPDEEP bit.

- Exiting IDLE mode: The processor wakes the system up when it detects the occurrence of any interrupt that is not masked in the NVIC Controller with sufficient priority to cause exception entry. The system goes back to the ACTIVE mode. The CPU and affected modules are restarted.

#### 16.6.2.8.2 STANDBY Mode

The STANDBY mode allows achieving very low power consumption.

In this mode, all clocks are stopped except those which are kept running if requested by a running module or have the ONDEMAND bit set to zero. For example, the RTC can operate in STANDBY mode. In this case, its Generic Clock clock source will also be enabled.

The regulator and the RAM operate in low-power mode.

A SLEEPONEXIT feature is also available.

- Entering STANDBY mode: This mode is entered by executing the WFI instruction with the SCR.SLEEPDEEP bit of the CPU is written to 1.
- Exiting STANDBY mode: Any peripheral able to generate an asynchronous interrupt can wake up the system. For example, a module running on a Generic clock can trigger an interrupt. When the enabled asynchronous wake-up event occurs and the system is woken up, the device will either execute the interrupt service routine or continue the normal program execution according to the Priority Mask Register (PRIMASK) configuration of the CPU.

### 16.6.3 SleepWalking

SleepWalking is the capability for a device to temporarily wake-up clocks for the peripheral to perform a task without waking-up the CPU in STANDBY sleep mode. At the end of the sleepwalking task, the device can either be awakened by an interrupt (from a peripheral involved in SleepWalking) or enter into STANDBY sleep mode again.

In this device, SleepWalking is supported only on GCLK clocks by using the on-demand clock principle of the clock sources. Refer to *On-demand, Clock Requests* for more details.

**Related Links**

14.6 On-demand, Clock Requests

### 16.6.4 DMA Operation

Not applicable.

### 16.6.5 Interrupts

The peripheral has the following interrupt sources:

- Clock Ready flag

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the peripheral is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register. Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. Refer to *Nested Vector Interrupt Controller* for

details. If the peripheral has one common interrupt request line for all the interrupt sources, the user must read the INTFLAG register to determine which interrupt condition is present.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 16.6.6 Events

Not applicable.

### 16.6.7 Sleep Mode Operation

In all IDLE sleep modes, the power manager is still running on the selected main clock.

In STANDDBY sleep mode, the power manager is frozen and is able to go back to ACTIVE mode upon any asynchronous interrupt.

## 16.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | | | | | |
| 0x01 | SLEEP | 7:0 | | | | | | | IDLE[1:0] | |
| 0x02<br>...<br>0x07 | Reserved | | | | | | | | | |
| 0x08 | CPUSEL | 7:0 | | | | | | CPUDIV[2:0] | | |
| 0x09 | APBASEL | 7:0 | | | | | | APBADIV[2:0] | | |
| 0x0A | APBBSEL | 7:0 | | | | | | APBBDIV[2:0] | | |
| 0x0B | APBCSEL | 7:0 | | | | | | APBCDIV[2:0] | | |
| 0x0C<br>...<br>0x13 | Reserved | | | | | | | | | |
| 0x14 | AHBMASK | 7:0 | | USB | DMAC | NVMCTRL | DSU | HPB2 | HPB1 | HPB0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | APBAMASK | 7:0 | | EIC | RTC | WDT | GCLK | SYSCTRL | PM | PAC0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | APBBMASK | 7:0 | | | USB | DMAC | PORT | NVMCTRL | DSU | PAC1 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | APBCMASK | 7:0 | SERCOM5 | SERCOM4 | SERCOM3 | SERCOM2 | SERCOM1 | SERCOM0 | EVSYS | PAC2 |
| | | 15:8 | TC7 | TC6 | TC5 | TC4 | TC3 | TCC2 | TCC1 | TCC0 |
| | | 23:16 | | | AC1 | I2S | PTC | DAC | AC | ADC |
| | | 31:24 | | | | | | | | |
| 0x24<br>...<br>0x33 | Reserved | | | | | | | | | |
| 0x34 | INTENCLR | 7:0 | | | | | | | | CKRDY |
| 0x35 | INTENSET | 7:0 | | | | | | | | CKRDY |
| 0x36 | INTFLAG | 7:0 | | | | | | | | CKRDY |
| 0x37 | Reserved | | | | | | | | | |
| 0x38 | RCAUSE | 7:0 | | SYST | WDT | EXT | | BOD33 | BOD12 | POR |

## 16.8 Register Description

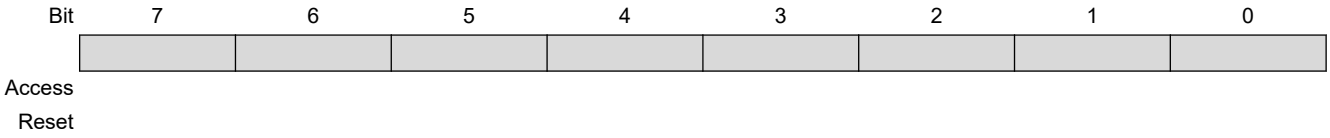
Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Exception for APBASEL, APBBSEL and APBCSEL: These registers must only be accessed with 8-bit access.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

### 16.8.1 Control

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x00
**Property:** Write-Protected



### 16.8.2 Sleep Mode

**Name:** SLEEP
**Offset:** 0x01
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | IDLE[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – IDLE[1:0]** Idle Mode Configuration
These bits select the Idle mode configuration after a WFI instruction.

| IDLE[1:0] | Name | Description |
|---|---|---|
| 0x0 | CPU | The CPU clock domain is stopped |
| 0x1 | AHB | The CPU and AHB clock domains are stopped |
| 0x2 | APB | The CPU, AHB and APB clock domains are stopped |
| 0x3 | | Reserved |

### 16.8.3 CPU Clock Select

**Name:** CPUSEL
**Offset:** 0x08
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CPUDIV[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – CPUDIV[2:0]** CPU Prescaler Selection
These bits define the division ratio of the main clock prescaler ($2^n$).

| CPUDIV[2:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Divide by 1 |
| 0x1 | DIV2 | Divide by 2 |
| 0x2 | DIV4 | Divide by 4 |
| 0x3 | DIV8 | Divide by 8 |
| 0x4 | DIV16 | Divide by 16 |
| 0x5 | DIV32 | Divide by 32 |
| 0x6 | DIV64 | Divide by 64 |
| 0x7 | DIV128 | Divide by 128 |

### 16.8.4 APBA Clock Select

**Name:** APBASEL
**Offset:** 0x09
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | APBADIV[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – APBADIV[2:0]** APBA Prescaler Selection
These bits define the division ratio of the APBA clock prescaler ($2^n$).

| APBADIV[2:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Divide by 1 |
| 0x1 | DIV2 | Divide by 2 |
| 0x2 | DIV4 | Divide by 4 |
| 0x3 | DIV8 | Divide by 8 |
| 0x4 | DIV16 | Divide by 16 |
| 0x5 | DIV32 | Divide by 32 |
| 0x6 | DIV64 | Divide by 64 |
| 0x7 | DIV128 | Divide by 128 |

### 16.8.5 APBB Clock Select

**Name:** APBBSEL
**Offset:** 0x0A
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | APBBDIV[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – APBBDIV[2:0]** APBB Prescaler Selection
These bits define the division ratio of the APBB clock prescaler ($2^n$).

| APBBDIV[2:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Divide by 1 |
| 0x1 | DIV2 | Divide by 2 |
| 0x2 | DIV4 | Divide by 4 |
| 0x3 | DIV8 | Divide by 8 |
| 0x4 | DIV16 | Divide by 16 |
| 0x5 | DIV32 | Divide by 32 |
| 0x6 | DIV64 | Divide by 64 |
| 0x7 | DIV128 | Divide by 128 |

### 16.8.6 APBC Clock Select

**Name:** APBCSEL
**Offset:** 0x0B
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | APBCDIV[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – APBCDIV[2:0]** APBC Prescaler Selection
These bits define the division ratio of the APBC clock prescaler ($2^n$).

| APBCDIV[2:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Divide by 1 |
| 0x1 | DIV2 | Divide by 2 |
| 0x2 | DIV4 | Divide by 4 |
| 0x3 | DIV8 | Divide by 8 |
| 0x4 | DIV16 | Divide by 16 |
| 0x5 | DIV32 | Divide by 32 |
| 0x6 | DIV64 | Divide by 64 |
| 0x7 | DIV128 | Divide by 128 |

### 16.8.7 AHB Mask

**Name:** AHBMASK
**Offset:** 0x14
**Reset:** 0x0000007F
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | USB | DMAC | NVMCTRL | DSU | HPB2 | HPB1 | HPB0 |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 6 – USB** USB AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the USB is stopped. |
| 1 | The AHB clock for the USB is enabled. |

**Bit 5 – DMAC** DMAC AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the DMAC is stopped. |
| 1 | The AHB clock for the DMAC is enabled. |

**Bit 4 – NVMCTRL** NVMCTRL AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the NVMCTRL is stopped. |
| 1 | The AHB clock for the NVMCTRL is enabled. |

**Bit 3 – DSU** DSU AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the DSU is stopped. |
| 1 | The AHB clock for the DSU is enabled. |

**Bit 2 – HPB2** HPB2 AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the HPB2 is stopped. |
| 1 | The AHB clock for the HPB2 is enabled. |

**Bit 1 – HPB1** HPB1 AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the HPB1 is stopped. |
| 1 | The AHB clock for the HPB1 is enabled. |

**Bit 0 – HPB0** HPB0 AHB Clock Mask

| Value | Description |
|---|---|
| 0 | The AHB clock for the HPB0 is stopped. |
| 1 | The AHB clock for the HPB0 is enabled. |

### 16.8.8 APBA Mask

**Name:** APBAMASK
**Offset:** 0x18
**Reset:** 0x0000007F
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | EIC | RTC | WDT | GCLK | SYSCTRL | PM | PAC0 |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 6 – EIC** EIC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the EIC is stopped. |
| 1 | The APBA clock for the EIC is enabled. |

**Bit 5 – RTC** RTC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the RTC is stopped. |
| 1 | The APBA clock for the RTC is enabled. |

**Bit 4 – WDT** WDT APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the WDT is stopped. |
| 1 | The APBA clock for the WDT is enabled. |

**Bit 3 – GCLK** GCLK APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the GCLK is stopped. |
| 1 | The APBA clock for the GCLK is enabled. |

**Bit 2 – SYSCTRL** SYSCTRL APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the SYSCTRL is stopped. |
| 1 | The APBA clock for the SYSCTRL is enabled. |

**Bit 1 – PM** PM APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the PM is stopped. |
| 1 | The APBA clock for the PM is enabled. |

**Bit 0 – PAC0** PAC0 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBA clock for the PAC0 is stopped. |
| 1 | The APBA clock for the PAC0 is enabled. |

### 16.8.9 APBB Mask

**Name:** APBBMASK
**Offset:** 0x1C
**Reset:** 0x0000007F
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | USB | DMAC | PORT | NVMCTRL | DSU | PAC1 |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 5 – USB** USB APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the USB is stopped. |
| 1 | The APBB clock for the USB is enabled. |

**Bit 4 – DMAC** DMAC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the DMAC is stopped. |
| 1 | The APBB clock for the DMAC is enabled. |

**Bit 3 – PORT** PORT APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the PORT is stopped. |
| 1 | The APBB clock for the PORT is enabled. |

**Bit 2 – NVMCTRL** NVMCTRL APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the NVMCTRL is stopped. |
| 1 | The APBB clock for the NVMCTRL is enabled. |

**Bit 1 – DSU** DSU APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the DSU is stopped. |
| 1 | The APBB clock for the DSU is enabled. |

**Bit 0 – PAC1** PAC1 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBB clock for the PAC1 is stopped. |
| 1 | The APBB clock for the PAC1 is enabled. |

### 16.8.10 APBC Mask

**Name:** APBCMASK
**Offset:** 0x20
**Reset:** 0x00010000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | AC1 | I2S | PTC | DAC | AC | ADC |
| Access | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TC7 | TC6 | TC5 | TC4 | TC3 | TCC2 | TCC1 | TCC0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SERCOM5 | SERCOM4 | SERCOM3 | SERCOM2 | SERCOM1 | SERCOM0 | EVSYS | PAC2 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 21 – AC1** AC1 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the AC1 is stopped. |
| 1 | The APBC clock for the AC1 is enabled. |

**Bit 20 – I2S** I2S APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the I2S is stopped. |
| 1 | The APBC clock for the I2S is enabled. |

**Bit 19 – PTC** PTC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the PTC is stopped. |
| 1 | The APBC clock for the PTC is enabled. |

**Bit 18 – DAC** DAC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the DAC is stopped. |
| 1 | The APBC clock for the DAC is enabled. |

**Bit 17 – AC** AC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the AC is stopped. |
| 1 | The APBC clock for the AC is enabled. |

**Bit 16 – ADC** ADC APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the ADC is stopped. |
| 1 | The APBC clock for the ADC is enabled. |

**Bit 15 – TC7** TC7 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TC7 is stopped. |
| 1 | The APBC clock for the TC7 is enabled. |

**Bit 14 – TC6** TC6 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TC6 is stopped. |
| 1 | The APBC clock for the TC6 is enabled. |

**Bit 13 – TC5** TC5 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TC5 is stopped. |
| 1 | The APBC clock for the TC5 is enabled. |

**Bit 12 – TC4** TC4 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TC4 is stopped. |
| 1 | The APBC clock for the TC4 is enabled. |

**Bit 11 – TC3** TC3 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TC3 is stopped. |
| 1 | The APBC clock for the TC3 is enabled. |

**Bit 10 – TCC2** TCC2 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TCC2 is stopped. |
| 1 | The APBC clock for the TCC2 is enabled. |

**Bit 9 – TCC1** TCC1 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TCC1 is stopped. |
| 1 | The APBC clock for the TCC1 is enabled. |

**Bit 8 – TCC0** TCC0 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the TCC0 is stopped. |
| 1 | The APBC clock for the TCC0 is enabled. |

**Bit 7 – SERCOM5** SERCOM5 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM5 is stopped. |
| 1 | The APBC clock for the SERCOM5 is enabled. |

**Bit 6 – SERCOM4** SERCOM4 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM4 is stopped. |
| 1 | The APBC clock for the SERCOM4 is enabled. |

**Bit 5 – SERCOM3** SERCOM3 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM3 is stopped. |
| 1 | The APBC clock for the SERCOM3 is enabled. |

**Bit 4 – SERCOM2** SERCOM2 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM2 is stopped. |
| 1 | The APBC clock for the SERCOM2 is enabled. |

**Bit 3 – SERCOM1** SERCOM1 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM1 is stopped. |
| 1 | The APBC clock for the SERCOM1 is enabled. |

**Bit 2 – SERCOM0** SERCOM0 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the SERCOM0 is stopped. |
| 1 | The APBC clock for the SERCOM0 is enabled. |

**Bit 1 – EVSYS** EVSYS APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the EVSYS is stopped. |
| 1 | The APBC clock for the EVSYS is enabled. |

**Bit 0 – PAC2** PAC2 APB Clock Enable

| Value | Description |
|---|---|
| 0 | The APBC clock for the PAC2 is stopped. |
| 1 | The APBC clock for the PAC2 is enabled. |

### 16.8.11 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x34
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CKRDY |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CKRDY** Clock Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Clock Ready Interrupt Enable bit and the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Clock Ready interrupt is disabled. |
| 1 | The Clock Ready interrupt is enabled and will generate an interrupt request when the Clock Ready Interrupt flag is set. |

### 16.8.12 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x35
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CKRDY |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CKRDY** Clock Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Clock Ready Interrupt Enable bit and enable the Clock Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Clock Ready interrupt is disabled. |
| 1 | The Clock Ready interrupt is enabled. |

### 16.8.13 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x36
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CKRDY |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CKRDY** Clock Ready
This flag is cleared by writing a one to the flag.

This flag is set when the synchronous CPU and APBx clocks have frequencies as indicated in the CPUSEL and APBxSEL registers, and will generate an interrupt if INTENCLR/SET.CKRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Clock Ready Interrupt flag.

### 16.8.14 Reset Cause

**Name:** RCAUSE
**Offset:** 0x38
**Reset:** 0x01
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SYST | WDT | EXT | | BOD33 | BOD12 | POR |
| Access | | R | R | R | | R | R | R |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 1 |

**Bit 6 – SYST** System Reset Request
This bit is set if a system reset request has been performed. Refer to the Cortex processor documentation for more details.

**Bit 5 – WDT** Watchdog Reset
This flag is set if a Watchdog Timer reset occurs.

**Bit 4 – EXT** External Reset
This flag is set if an external reset occurs.

**Bit 2 – BOD33** Brown Out 33 Detector Reset
This flag is set if a BOD33 reset occurs.

**Bit 1 – BOD12** Brown Out 12 Detector Reset
This flag is set if a BOD12 reset occurs.

**Bit 0 – POR** Power On Reset
This flag is set if a POR occurs.

# 17. SYSCTRL – System Controller

## 17.1 Overview

The System Controller (SYSCTRL) provides a user interface to the clock sources, brown out detectors, on-chip voltage regulator and voltage reference of the device.

Through the interface registers, it is possible to enable, disable, calibrate and monitor the SYSCTRL sub-peripherals.

All sub-peripheral statuses are collected in the Power and Clocks Status register (PCLKSR). They can additionally trigger interrupts upon status changes through the INTENSET (INTENSET), INTENCLR (INTENCLR) and INTFLAG (INTFLAG) registers.

Additionally, BOD33 and BOD12 interrupts can be used to wake up the device from Stand-by mode upon a programmed brown-out detection.

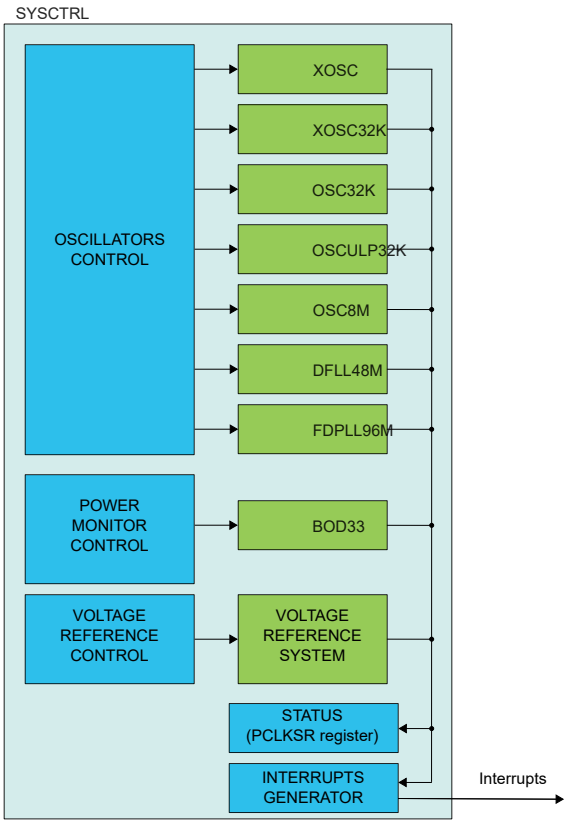
## 17.2 Features

- 0.4-32MHz Crystal Oscillator (XOSC)
  - Tunable gain control
  - Programmable start-up time
  - Crystal or external input clock on XIN I/O
- 32.768kHz Crystal Oscillator (XOSC32K)
  - Automatic or manual gain control
  - Programmable start-up time
  - Crystal or external input clock on XIN32 I/O
- 32.768kHz High Accuracy Internal Oscillator (OSC32K)
  - Frequency fine tuning
  - Programmable start-up time
- 32.768kHz Ultra Low-Power Internal Oscillator (OSCULP32K)
  - Ultra low-power, always-on oscillator
  - Frequency fine tuning
  - Calibration value loaded from Flash Factory Calibration at reset
- 8MHz Internal Oscillator (OSC8M)
  - Fast startup
  - Output frequency fine tuning
  - 4/2/1MHz divided output frequencies available
  - Calibration value loaded from Flash Factory Calibration at reset
- Digital Frequency Locked Loop (DFLL48M)
  - Internal oscillator with no external components
  - 48MHz output frequency
  - Operates standalone as a high-frequency programmable oscillator in Open-Loop mode
  - Operates as an accurate frequency multiplier against a known frequency in Closed-Loop mode

- Fractional Digital Phase Locked Loop (FDPLL96M)
  - 48MHz to 96MHz output clock frequency
  - 32KHz to 2MHz input reference clock frequency range
  - Three possible sources for the reference clock
  - Adjustable proportional integral controller
  - Fractional part used to achieve 1/16th of reference clock step
- 3.3V Brown-Out Detector (BOD33)
  - Programmable threshold
  - Threshold value loaded from Flash User Calibration at startup
  - Triggers resets or interrupts
  - Operating modes:
    - Continuous mode
    - Sampled mode for low power applications (programmable refresh frequency)
  - Hysteresis
- Internal Voltage Regulator system (VREG)
  - Operating modes:
    - Normal mode
    - Low-power mode
  - With an internal non-configurable Brown-out detector (BOD12)
- 1.2V Brown-Out Detector (BOD12)
  - Programmable threshold
  - Threshold value loaded from Flash User Calibration at start-up
  - Triggers resets or interrupts
  - Operating modes:
    - Continuous mode
    - Sampled mode for low power applications (programmable refresh frequency)
  - Hysteresis
- Voltage Reference System (VREF)
  - Bandgap voltage generator with programmable calibration value
  - Temperature sensor
  - Bandgap calibration value loaded from Flash Factory Calibration at start-up

## 17.3 Block Diagram

**Figure 17-1. SYSCTRL Block Diagram**




## 17.4 Signal Description

| Signal Name | Types | Description |
|---|---|---|
| XIN | Analog Input | Multipurpose Crystal Oscillator or external clock generator input |
| XOUT | Analog Output | External Multipurpose Crystal Oscillator output |
| XIN32 | Analog Input | 32kHz Crystal Oscillator or external clock generator input |
| XOUT32 | Analog Output | 32kHz Crystal Oscillator output |

The I/O lines are automatically selected when XOSC or XOSC32K are enabled. Refer to *Oscillator Pinout*.

**Related Links**

7. I/O Multiplexing and Considerations

## 17.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 17.5.1 I/O Lines

I/O lines are configured by SYSCTRL when either XOSC or XOSC32K are enabled, and need no user configuration.

### 17.5.2 Power Management

The SYSCTRL can continue to operate in any sleep mode where the selected source clock is running. The SYSCTRL interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to *PM – Power Manager* on the different sleep modes.

**Related Links**

16. PM – Power Manager

### 17.5.3 Clocks

The SYSCTRL gathers controls for all device oscillators and provides clock sources to the Generic Clock Controller (GCLK). The available clock sources are: XOSC, XOSC32K, OSC32K, OSCULP32K, OSC8M, DFLL48M and FDPLL96M.

The SYSCTRL bus clock (CLK_SYSCTRL_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_SYSCTRL_APB can be found in the Peripheral Clock Masking section in the PM – Power Manager.

The clock used by BOD33 and BOD12 in sampled mode is asynchronous to the user interface clock (CLK_SYSCTRL_APB). Likewise, the DFLL48M control logic uses the DFLL oscillator output, which is also asynchronous to the user interface clock (CLK_SYSCTRL_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to 17.6.14 Synchronization for further details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

### 17.5.4 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the SYSCTRL interrupts requires the Interrupt Controller to be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 17.5.5 Debug Operation

When the CPU is halted in debug mode, the SYSCTRL continues normal operation. If the SYSCTRL is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

If debugger cold-plugging is detected by the system, BOD33 and BOD12 resets will be masked. The BOD resets keep running under hot-plugging. This allows to correct a BOD33 user level too high for the available supply.

### 17.5.6 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag Status and Clear register (INTFLAG)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 17.5.7 Analog Connections

When used, the 32.768kHz crystal must be connected between the XIN32 and XOUT32 pins, and the 0.4-32MHz crystal must be connected between the XIN and XOUT pins, along with any required load capacitors. For details on recommended oscillator characteristics and capacitor load, refer to the *Electrical Characteristics* for details.

**Related Links**

37. Electrical Characteristics

## 17.6 Functional Description

### 17.6.1 Principle of Operation

XOSC, XOSC32K, OSC32K, OSCULP32K, OSC8M, DFLL48M, FDPLL96M, BOD33, BOD12, and VREF are configured through SYSCTRL control registers. Through this interface, the sub-peripherals are enabled, disabled or have their calibration values updated.

The Power and Clocks Status register gathers different status signals coming from the sub-peripherals controlled by the SYSCTRL. The status signals can be used to generate system interrupts, and in some cases wake up the system from Standby mode, provided the corresponding interrupt is enabled.

The oscillator must be enabled to run. The oscillator is enabled by writing a one to the ENABLE bit in the respective oscillator control register, and disabled by writing a zero to the oscillator control register. In Idle mode, the default operation of the oscillator is to run only when requested by a peripheral. In Standby mode, the default operation of the oscillator is to stop. This behavior can be changed by the user, see below for details.

The behavior of the oscillators in the different sleep modes is shown in the table below.

**Table 17-1. Behavior of the Oscillators**

| Oscillator | Idle 0, 1, 2 | Standby |
|---|---|---|
| XOSC | Run on request | Stop |
| XOSC32K | Run on request | Stop |
| OSC32K | Run on request | Stop |
| OSCULP32K | Run | Run |
| OSC8M | Run on request | Stop |
| DFLL48M | Run on request | Stop |
| FDPLL96M | Run on request | Stop |

To force an oscillator to always run in Idle mode, and not only when requested by a peripheral, the oscillator ONDEMAND bit must be written to zero. The default value of this bit is one, and thus the default operation in Idle mode is to run only when requested by a peripheral.

To force the oscillator to run in Standby mode, the RUNSTDBY bit must be written to one. The oscillator will then run in Standby mode when requested by a peripheral (ONDEMAND is one). To force an oscillator to always run in Standby mode, and not only when requested by a peripheral, the ONDEMAND bit must be written to zero and RUNSTDBY must be written to one.

The next table shows the behavior in the different sleep modes, depending on the settings of ONDEMAND and RUNSTDBY.

**Table 17-2. Behavior in the different sleep modes**

| Sleep mode | ONDEMAND | RUNSTDBY | Behavior |
|---|---|---|---|
| Idle 0, 1, 2 | 0 | X | Run |
| Idle 0, 1, 2 | 1 | X | Run when requested by a peripheral |
| Standby | 0 | 0 | Stop |
| Standby | 0 | 1 | Run |
| Standby | 1 | 0 | Stop |
| Standby | 1 | 1 | Run when requested by a peripheral |

**Note:** This does not apply to the OSCULP32K oscillator, which is always running and cannot be disabled.

### 17.6.2 External Multipurpose Crystal Oscillator (XOSC) Operation

The XOSC can operate in two different modes:

- External clock, with an external clock signal connected to the XIN pin
- Crystal oscillator, with an external 0.4-32MHz crystal

The XOSC can be used as a clock source for generic clock generators, as described in the *GCLK – Generic Clock Controller*.

At reset, the XOSC is disabled, and the XIN/XOUT pins can be used as General Purpose I/O (GPIO) pins or by other peripherals in the system. When XOSC is enabled, the operating mode determines the GPIO usage. When in crystal oscillator mode, the XIN and XOUT pins are controlled by the SYSCTRL, and GPIO functions are overridden on both pins. When in external clock mode, only the XIN pin will be overridden and controlled by the SYSCTRL, while the XOUT pin can still be used as a GPIO pin.

The XOSC is enabled by writing a one to the Enable bit in the External Multipurpose Crystal Oscillator Control register (XOSC.ENABLE). To enable the XOSC as a crystal oscillator, a one must be written to the XTAL Enable bit (XOSC.XTALEN). If XOSC.XTALEN is zero, external clock input will be enabled.

When in crystal oscillator mode (XOSC.XTALEN is one), the External Multipurpose Crystal Oscillator Gain (XOSC.GAIN) must be set to match the external crystal oscillator frequency. If the External Multipurpose Crystal Oscillator Automatic Amplitude Gain Control (XOSC.AMPGC) is one, the oscillator amplitude will be automatically adjusted, and in most cases result in a lower power consumption.

The XOSC will behave differently in different sleep modes based on the settings of XOSC.RUNSTDBY, XOSC.ONDEMAND and XOSC.ENABLE:

| XOSC.RUNSTDBY | XOSC.ONDEMAND | XOSC.ENABLE | Sleep Behavior |
|---|---|---|---|
| - | - | 0 | Disabled |
| 0 | 0 | 1 | Always run in IDLE sleep modes. Disabled in STANDBY sleep mode. |
| 0 | 1 | 1 | Only run in IDLE sleep modes if requested by a peripheral. Disabled in STANDBY sleep mode. |
| 1 | 0 | 1 | Always run in IDLE and STANDBY sleep modes. |
| 1 | 1 | 1 | Only run in IDLE or STANDBY sleep modes if requested by a peripheral. |

After a hard reset, or when waking up from a sleep mode where the XOSC was disabled, the XOSC will need a certain amount of time to stabilize on the correct frequency. This start-up time can be configured by changing the Oscillator Start-Up Time bit group (XOSC.STARTUP) in the External Multipurpose Crystal Oscillator Control register. During the start-up time, the oscillator output is masked to ensure that no unstable clock propagates to the digital logic. The External Multipurpose Crystal Oscillator Ready bit in the Power and Clock Status register (PCLKSR.XOSCRDY) is set when the user-selected start-up time is over. An interrupt is generated on a zero-to-one transition on PCLKSR.XOSCRDY if the External Multipurpose Crystal Oscillator Ready bit in the Interrupt Enable Set register (INTENSET.XOSCRDY) is set.

**Note:** Do not enter standby mode when an oscillator is in start-up:
Wait for the OSCxRDY bit in SYSCTRL.PCLKSR register to be set before going into standby mode.

**Related Links**

15. GCLK - Generic Clock Controller

### 17.6.3 32kHz External Crystal Oscillator (XOSC32K) Operation

The XOSC32K can operate in two different modes:

- External clock, with an external clock signal connected to XIN32
- Crystal oscillator, with an external 32.768kHz crystal connected between XIN32 and XOUT32

The XOSC32K can be used as a source for generic clock generators, as described in the *GCLK – Generic Clock Controller*.

At Power-on Reset (POR) the XOSC32K is disabled, and the XIN32/XOUT32 pins can be used as General Purpose I/O (GPIO) pins or by other peripherals in the system. When XOSC32K is enabled, the operating mode determines the GPIO usage. When in crystal oscillator mode, XIN32 and XOUT32 are controlled by the SYSCTRL, and GPIO functions are overridden on both pins. When in external clock mode, only the XIN32 pin will be overridden and controlled by the SYSCTRL, while the XOUT32 pin can still be used as a GPIO pin.

The external clock or crystal oscillator is enabled by writing a one to the Enable bit (XOSC32K.ENABLE) in the 32kHz External Crystal Oscillator Control register. To enable the XOSC32K as a crystal oscillator, a one must be written to the XTAL Enable bit (XOSC32K.XTALEN). If XOSC32K.XTALEN is zero, external clock input will be enabled.

The oscillator is disabled by writing a zero to the Enable bit (XOSC32K.ENABLE) in the 32kHz External Crystal Oscillator Control register while keeping the other bits unchanged. Writing to the XOSC32K.ENABLE bit while writing to other bits may result in unpredictable behavior. The oscillator remains enabled in all sleep modes if it has been enabled beforehand. The start-up time of the 32kHz External Crystal Oscillator is selected by writing to the Oscillator Start-Up Time bit group (XOSC32K.STARTUP) in the in the 32kHz External Crystal Oscillator Control register. The SYSCTRL masks the oscillator output during the start-up time to ensure that no unstable clock propagates to the digital logic. The 32kHz External Crystal Oscillator Ready bit (PCLKSR.XOSC32KRDY) in the Power and Clock Status register is set when the user-selected startup time is over. An interrupt is generated on a zero-to-one transition of PCLKSR.XOSC32KRDY if the 32kHz External Crystal Oscillator Ready bit (INTENSET.XOSC32KRDY) in the Interrupt Enable Set Register is set.

As a crystal oscillator usually requires a very long start-up time (up to one second), the 32kHz External Crystal Oscillator will keep running across resets, except for power-on reset (POR).

XOSC32K can provide two clock outputs when connected to a crystal. The XOSC32K has a 32.768kHz output enabled by writing a one to the 32kHz External Crystal Oscillator 32kHz Output Enable bit (XOSC32K.EN32K) in the 32kHz External Crystal Oscillator Control register. XOSC32K.EN32K is only usable when XIN32 is connected to a crystal, and not when an external digital clock is applied on XIN32.

**Note:** Do not enter Standby mode when an oscillator is in start-up:
Wait for the OSCxRDY bit in SYSCTRL.PCLKSR register to be set before going into Standby mode.

**Related Links**

15. GCLK - Generic Clock Controller

### 17.6.4 32kHz Internal Oscillator (OSC32K) Operation

The OSC32K provides a tunable, low-speed and low-power clock source.

The OSC32K can be used as a source for the generic clock generators, as described in the *GCLK – Generic Clock Controller*.

The OSC32K is disabled by default. The OSC32K is enabled by writing a one to the 32kHz Internal Oscillator Enable bit (OSC32K.ENABLE) in the 32kHz Internal Oscillator Control register. It is disabled by writing a zero to OSC32K.ENABLE. The OSC32K has a 32.768kHz output enabled by writing a one to the 32kHz Internal Oscillator 32kHz Output Enable bit (OSC32K.EN32K).

The frequency of the OSC32K oscillator is controlled by the value in the 32kHz Internal Oscillator Calibration bits (OSC32K.CALIB) in the 32kHz Internal Oscillator Control register. The OSC32K.CALIB value must be written by the user. Flash Factory Calibration values are stored in the NVM Software Calibration Area (refer to *NVM Software Calibration Area Mapping*). When writing to the Calibration bits, the user must wait for the PCLKSR.OSC32KRDY bit to go high before the value is committed to the oscillator.

**Related Links**

15. GCLK - Generic Clock Controller
10.3.2 NVM Software Calibration Area Mapping

### 17.6.5 32kHz Ultra Low Power Internal Oscillator (OSCULP32K) Operation

The OSCULP32K provides a tunable, low-speed and ultra-low-power clock source. The OSCULP32K is factory-calibrated under typical voltage and temperature conditions. The OSCULP32K should be preferred to the OSC32K whenever the power requirements are prevalent over frequency stability and accuracy.

The OSCULP32K can be used as a source for the generic clock generators, as described in the *GCLK – Generic Clock Controller*.

The OSCULP32K is enabled by default after a power-on reset (POR) and will always run except during POR. The OSCULP32K has a 32.768kHz output and a 1.024kHz output that are always running.

The frequency of the OSCULP32K oscillator is controlled by the value in the 32kHz Ultra Low Power Internal Oscillator Calibration bits (OSCULP32K.CALIB) in the 32kHz Ultra Low Power Internal Oscillator Control register. OSCULP32K.CALIB is automatically loaded from Flash Factory Calibration during startup, and is used to compensate for process variation, as described in the *Electrical Characteristics*. The calibration value can be overridden by the user by writing to OSCULP32K.CALIB.

**Related Links**

37. Electrical Characteristics
15. GCLK - Generic Clock Controller

### 17.6.6 8MHz Internal Oscillator (OSC8M) Operation

OSC8M is an internal oscillator operating in open-loop mode and generating an 8MHz frequency. The OSC8M is factory-calibrated under typical voltage and temperature conditions.

OSC8M is the default clock source that is used after a power-on reset (POR). The OSC8M can be used as a source for the generic clock generators, as described in the *GCLK – Generic Clock Controller*.

In order to enable OSC8M, the Oscillator Enable bit in the OSC8M Control register (OSC8M.ENABLE) must be written to one. OSC8M will not be enabled until OSC8M.ENABLE is set. In order to disable OSC8M, OSC8M.ENABLE must be written to zero. OSC8M will not be disabled until OSC8M is cleared.

The frequency of the OSC8M oscillator is controlled by the value in the calibration bits (OSC8M.CALIB) in the OSC8M Control register. CALIB is automatically loaded from Flash Factory Calibration during start-up, and is used to compensate for process variation, as described in the *Electrical Characteristics*.

The user can control the oscillation frequency by writing to the Frequency Range (FRANGE) and Calibration (CALIB) bit groups in the 8MHz RC Oscillator Control register (OSC8M). It is not recommended to update the FRANGE and CALIB bits when the OSC8M is enabled. As this is in open-loop mode, the frequency will be voltage, temperature and process dependent. Refer to the *Electrical Characteristics* for details.

OSC8M is automatically switched off in certain sleep modes to reduce power consumption, as described in the *PM – Power Manager*.

**Related Links**

16. PM – Power Manager
37. Electrical Characteristics
15. GCLK - Generic Clock Controller

### 17.6.7 Digital Frequency Locked Loop (DFLL48M) Operation

The DFLL48M can operate in both open-loop mode and closed-loop mode. In closed-loop mode, a low-frequency clock with high accuracy can be used as the reference clock to get high accuracy on the output clock (CLK_DFLL48M).

The DFLL48M can be used as a source for the generic clock generators, as described in the *GCLK – Generic Clock Controller*.

**Related Links**

15. GCLK - Generic Clock Controller

#### 17.6.7.1 Basic Operation

##### 17.6.7.1.1 Open-Loop Operation

After any reset, the open-loop mode is selected. When operating in open-loop mode, the output frequency of the DFLL48M will be determined by the values written to the DFLL Coarse Value bit group and the DFLL Fine Value bit group (DFLLVAL.COARSE and DFLLVAL.FINE) in the DFLL Value register. Using "DFLL48M COARSE CAL" value from *NVM Software Calibration Area Mapping* in DFLL.COARSE helps to output a frequency close to 48 MHz.

It is possible to change the values of DFLLVAL.COARSE and DFLLVAL.FINE and thereby the output frequency of the DFLL48M output clock, CLK_DFLL48M, while the DFLL48M is enabled and in use. CLK_DFLL48M is ready to be used when PCLKSR.DFLLRDY is set after enabling the DFLL48M.

**Related Links**

10.3.2 NVM Software Calibration Area Mapping

##### 17.6.7.1.2 Closed-Loop Operation

In closed-loop operation, the output frequency is continuously regulated against a reference clock. Once the multiplication factor is set, the oscillator fine tuning is automatically adjusted. The DFLL48M must be correctly configured before closed-loop operation can be enabled. After enabling the DFLL48M, it must be configured in the following way:

1. Enable and select a reference clock (CLK_DFLL48M_REF). CLK_DFLL48M_REF is Generic Clock Channel 0 (GCLK_DFLL48M_REF). Refer to *GCLK – Generic Clock Controller* for details.
2. Select the maximum step size allowed in finding the Coarse and Fine values by writing the appropriate values to the DFLL Coarse Maximum Step and DFLL Fine Maximum Step bit groups (DFLLMUL.CSTEP and DFLLMUL.FSTEP) in the DFLL Multiplier register. A small step size will ensure low overshoot on the output frequency, but will typically result in longer lock times. A high value might give a large overshoot, but will typically provide faster locking. DFLLMUL.CSTEP and DFLLMUL.FSTEP should not be higher than 50% of the maximum value of DFLLVAL.COARSE and DFLLVAL.FINE, respectively.
3. Select the multiplication factor in the DFLL Multiply Factor bit group (DFLLMUL.MUL) in the DFLL Multiplier register. Care must be taken when choosing DFLLMUL.MUL so that the output frequency does not exceed the maximum frequency of the DFLL.
4. Start the closed loop mode by writing a one to the DFLL Mode Selection bit (DFLLCTRL.MODE) in the DFLL Control register.

The frequency of CLK_DFLL48M ($F_{clkdfll48m}$) is given by:

$$F_{\mathrm{clkdfll48}m} = \mathrm{DFLLMUL} \cdot \mathrm{MUL} \times F_{\mathrm{clkdfll48mref}}$$

where $F_{clkdfll48mref}$ is the frequency of the reference clock (CLK_DFLL48M_REF). DFLLVAL.COARSE and DFLLVAL.FINE are read-only in closed-loop mode, and are controlled by the frequency tuner to meet user specified frequency. In closed-loop mode, the value in DFLLVAL.COARSE is used by the frequency tuner as a starting point for Coarse. Writing DFLLVAL.COARSE to a value close to the final value before entering closed-loop mode will reduce the time needed to get a lock on Coarse.

Using "DFLL48M COARSE CAL" from *NVM Software Calibration Area Mapping* for DFLL.COARSE will start DFLL with a frequency close to 48 MHz.

Following Software sequence should be followed while using the same.

1. load "DFLL48M COARSE CAL" from *NVM User Row Mapping* in DFLL.COARSE register
2. Set DFLLCTRL.BPLCKC bit

3. Start DFLL close loop

This procedure will reduce DFLL Lock time to DFLL Fine lock time.

**Related Links**

15. GCLK - Generic Clock Controller

10.3.2 NVM Software Calibration Area Mapping

#### 17.6.7.1.3 Frequency Locking

The locking of the frequency in closed-loop mode is divided into two stages. In the first, coarse stage, the control logic quickly finds the correct value for DFLLVAL.COARSE and sets the output frequency to a value close to the correct frequency. On coarse lock, the DFLL Locked on Coarse Value bit (PCLKSR.DFLLLOCKC) in the Power and Clocks Status register will be set.

In the second, fine stage, the control logic tunes the value in DFLLVAL.FINE so that the output frequency is very close to the desired frequency. On fine lock, the DFLL Locked on Fine Value bit (PCLKSR.DFLLLOCKF) in the Power and Clocks Status register will be set.

Interrupts are generated by both PCLKSR.DFLLLOCKC and PCLKSR.DFLLLOCKF if INTENSET.DFLLOCKC or INTENSET.DFLLOCKF are written to one.

CLK_DFLL48M is ready to be used when the DFLL Ready bit (PCLKSR.DFLLRDY) in the Power and Clocks Status register is set, but the accuracy of the output frequency depends on which locks are set. For lock times, refer to the *Electrical Characteristics*.

**Related Links**

37. Electrical Characteristics

#### 17.6.7.1.4 Frequency Error Measurement

The ratio between CLK_DFLL48M_REF and CLK48M_DFLL is measured automatically when the DFLL48M is in closed-loop mode. The difference between this ratio and the value in DFLLMUL.MUL is stored in the DFLL Multiplication Ratio Difference bit group(DFLLVAL.DIFF) in the DFLL Value register. The relative error on CLK_DFLL48M compared to the target frequency is calculated as follows:

$$\mathrm{ERROR} = \frac{\mathrm{DIFF}}{\mathrm{MUL}}$$

#### 17.6.7.1.5 Drift Compensation

If the Stable DFLL Frequency bit (DFLLCTRL.STABLE) in the DFLL Control register is zero, the frequency tuner will automatically compensate for drift in the CLK_DFLL48M without losing either of the locks. This means that DFLLVAL.FINE can change after every measurement of CLK_DFLL48M.

The DFLLVAL.FINE value overflows or underflows can occur in close loop mode when the clock source reference drifts or is unstable. This will set the DFLL Out Of Bounds bit (PCLKSR.DFLLOOB) in the Power and Clocks Status register.

To avoid this error, the reference clock in close loop mode must be stable, an external oscillator is recommended and internal oscillator forbidden. The better choice is to use an XOSC32K.

#### 17.6.7.1.6 Reference Clock Stop Detection

If CLK_DFLL48M_REF stops or is running at a very low frequency (slower than CLK_DFLL48M/(2 * $MUL_{MAX}$)), the DFLL Reference Clock Stopped bit (PCLKSR.DFLLRCS) in the Power and Clocks Status register will be set. Detecting a stopped reference clock can take a long time, on the order of 217 CLK_DFLL48M cycles. When the reference clock is stopped, the DFLL48M will operate as if in open-loop mode. Closed-loop mode operation will automatically resume if the CLK_DFLL48M_REF is restarted. An

interrupt is generated on a zero-to-one transition on PCLKSR.DFLLRCS if the DFLL Reference Clock Stopped bit (INTENSET.DFLLRCS) in the Interrupt Enable Set register is set.

#### 17.6.7.2 Additional Features

##### 17.6.7.2.1 Dealing with Delay in the DFLL in Closed-Loop Mode

The time from selecting a new CLK_DFLL48M frequency until this frequency is output by the DFLL48M can be up to several microseconds. If the value in DFLLMUL.MUL is small, this can lead to instability in the DFLL48M locking mechanism, which can prevent the DFLL48M from achieving locks. To avoid this, a chill cycle, during which the CLK_DFLL48M frequency is not measured, can be enabled. The chill cycle is enabled by default, but can be disabled by writing a one to the DFLL Chill Cycle Disable bit (DFLLCTRL.CCDIS) in the DFLL Control register. Enabling chill cycles might double the lock time.

Another solution to this problem consists of using less strict lock requirements. This is called Quick Lock (QL), which is also enabled by default, but it can be disabled by writing a one to the Quick Lock Disable bit (DFLLCTRL.QLDIS) in the DFLL Control register. The Quick Lock might lead to a larger spread in the output frequency than chill cycles, but the average output frequency is the same.

##### 17.6.7.2.2 USB Clock Recovery Mode

USB Clock Recovery mode can be used to create the 48MHz USB clock from the USB Start Of Frame (SOF). This mode is enabled by writing a '1' to both the USB Clock Recovery Mode bit and the Mode bit in DFLL Control register (DFLLCTRL.USBCRM and DFLLCTRL.MODE).

**Note:** In USB Clock Recovery mode, the status bits of the DFLL in OSCCTRL.STATUS are determined by the USB bus activity, and have no valid meaning.

The SOF signal from USB device will be used as reference clock (CLK_DFLL_REF), ignoring the selected generic clock reference. When the USB device is connected, a SOF will be sent every 1ms, thus DFLLVAL.MUX bits should be written to 0xBB80 to obtain a 48MHz clock.

In USB clock recovery mode, the DFLLCTRL.BPLCKC bit state is ignored, and the value stored in the DFLLVAL.COARSE will be used as final Coarse Value. The COARSE calibration value can be loaded from NVM OTP row by software. The locking procedure will also go instantaneously to the fine lock search.

The DFLLCTRL.QLDIS bit must be cleared and DFLLCTRL.CCDIS should be set to speed up the lock phase. The DFLLCTRL.STABLE bit state is ignored, an auto jitter reduction mechanism is used instead.

##### 17.6.7.2.3 Wake from Sleep Modes

DFLL48M can optionally reset its lock bits when it is disabled. This is configured by the Lose Lock After Wake bit (DFLLCTRL.LLAW) in the DFLL Control register. If DFLLCTRL.LLAW is zero, the DFLL48M will be re-enabled and start running with the same configuration as before being disabled, even if the reference clock is not available. The locks will not be lost. When the reference clock has restarted, the Fine tracking will quickly compensate for any frequency drift during sleep if DFLLCTRL.STABLE is zero. If DFLLCTRL.LLAW is one when the DFLL is turned off, the DFLL48M will lose all its locks, and needs to regain these through the full lock sequence.

##### 17.6.7.2.4 Accuracy

There are three main factors that determine the accuracy of $F_{clkdfll48m}$. These can be tuned to obtain maximum accuracy when fine lock is achieved.

- Fine resolution: The frequency step between two Fine values. This is relatively smaller for high output frequencies.
- Resolution of the measurement: If the resolution of the measured $F_{clkdfll48m}$ is low, i.e., the ratio between the CLK_DFLL48M frequency and the CLK_DFLL48M_REF frequency is small, then the

DFLL48M might lock at a frequency that is lower than the targeted frequency. It is recommended to use a reference clock frequency of 32kHz or lower to avoid this issue for low target frequencies.

- The accuracy of the reference clock.

### 17.6.8 FDPLL96M – Fractional Digital Phase-Locked Loop Controller (DFLL96M)

#### 17.6.8.1 Overview

The FDPLL96M controller allows flexible interface to the core digital function of the Digital Phase Locked Loop (DPLL). The FDPLL96M integrates a digital filter with a proportional integral controller, a Time-to-Digital Converter (TDC), a test mode controller, a Digitally Controlled Oscillator (DCO) and a PLL controller. It also provides a fractional multiplier of frequency N between the input and output frequency.

The CLK_FDPLL96M_REF is the DPLL input clock reference. The selectable sources for the reference clock are XOSC32K, XOSC and GCLK_DPLL. The path between XOSC and input multiplexer integrates a clock divider. The selected clock must be configured and enabled before using the FDPLL96M. If the GCLK is selected as reference clock, it must be configured and enabled in the Generic Clock Controller before using the FDPLL96M. Refer to *GCLK – Generic Clock Controller* for details. If the GCLK_DPLL is selected as the source for the CLK_FDPLL96M_REF, care must be taken to make sure the source for this GCLK is within the valid frequency range for the FDPLL96M.

The XOSC source can be divided inside the FDPLL96M. The user must make sure that the programmable clock divider and XOSC frequency provides a valid CLK_FDPLL96M_REF clock frequency that meets the FDPLL96M input frequency range.

The output clock of the FDPLL96M is CLK_FDPLL96M. The state of the CLK_FDPLL96M clock only depends on the FDPLL96M internal control of the final clock gater CG.

The FDPLL96M requires a 32kHz clock from the GCLK when the FDPLL96M internal lock timer is used. This clock must be configured and enabled in the Generic Clock Controller before using the FDPLL96M. Refer to *GCLK – Generic Clock Controller* for details.

**Table 17-3. Generic Clock Input for FDPLL96M**

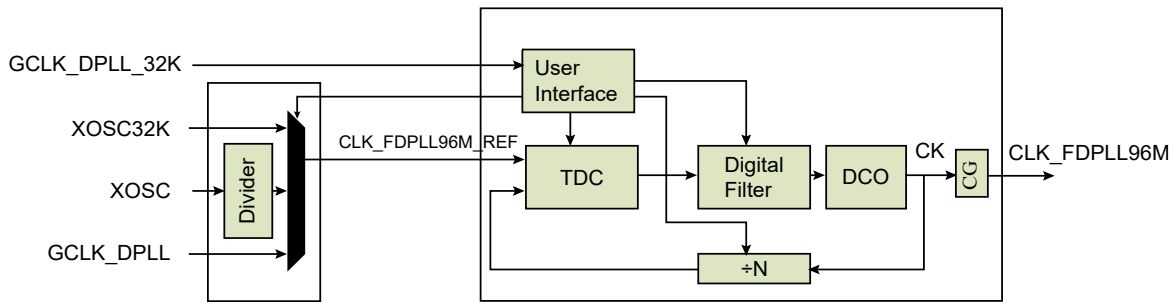
| Generic Clock | FDPLL96M |
|---|---|
| FDPLL96M 32kHz clock | GCLK_DPLL_32K for internal lock timer |
| FDPLL96M | GCLK_DPLL for CLK_FDPLL96M_REF |

**Related Links**

15. GCLK - Generic Clock Controller

#### 17.6.8.2 Block Diagram

**Figure 17-2. FDPLL96M Block Diagram**



#### 17.6.8.3 Principle of Operation

The task of the FDPLL96M is to maintain coherence between the input reference clock signal (CLK_FDPLL96M_REF) and the respective output frequency CK via phase comparison. The FDPLL96M supports three independent sources of clocks; XOSC32K, XOSC and GCLK_DPLL. When the FDPLL96M is enabled, the relationship between the reference clock (CLK_FDPLL96M_REF) frequency and the output clock (CLK_FDPLL96M) frequency is defined below.

$$f_{clk_fdpll96m} = f_{clk_fdpll96m_ref} \times \left(LDR + 1 + \frac{LDRFRAC}{16}\right)$$

Where LDR is the loop divider ratio integer part, LDRFRAC is the loop divider ratio fractional part, $f_{ckrx}$ is the frequency of the selected reference clock and $f_{ck}$ is the frequency of the FDPLL96M output clock. As previously stated a clock divider exist between XOSC and CLK_FDPLL96M_REF. The frequency between the two clocks is defined below.

$$f_{clk_fdpll96m_ref} = f_{xosc} \times \left(\frac{1}{2 \times (DIV + 1)}\right)$$

When the FDPLL96M is disabled, the output clock is reset. If the loop divider ratio fractional part (DPLLRATIO.LDRFRAC) field is reset, the FDPLL96M works in integer mode, otherwise the fractional mode is activated. It shall be noted that fractional part has a negative impact on the jitter of the FDPLL96M.

Example (integer mode only): assuming $f_{ckr}$ = 32kHz and $f_{ck}$ = 48MHz, the multiplication ratio is 1500. It means that LDR shall be set to 1499.

Example (fractional mode): assuming $f_{ckr}$ = 32kHz and $f_{ck}$ = 48.006MHz, the multiplication ratio is 1500.1875 (1500 + 3/16). Thus LDR is set to 1499 and LDRFRAC to 3.

#### 17.6.8.4 Initialization, Enabling, Disabling and Resetting

The FDPLL96M is enabled by writing a one to the Enable bit in the DPLL Control A register (DPLLCTRLA.ENABLE). The FDPLL96M is disabled by writing a zero to DPLLCTRLA.ENABLE. The frequency of the FDPLL96M output clock CK is stable when the module is enabled and when the DPLL Lock Status bit in the DPLL Status register (DPLLSTATUS.LOCK) bit is set. When DPLLCTRLB.LTIME is different from 0, a user defined lock time is used to validate the lock operation. In this case the lock time is constant. If DPLLCTRLB.LTIME is reset, the lock signal is linked with the status bit of the DPLL, the lock time vary depending on the filter selection and final target frequency.

When DPLLCTRLB.WUF is set, the wake up fast mode is activated. In that mode, the clock gating cell is enabled at the end of the startup time. At that time, the final frequency is not stable as it is still in the acquisition period, but it allows saving several milliseconds. After first acquisition, DPLLCTRLB.LBYPASS indicates if the Lock signal is discarded from the control of the clock gater generating the output clock CLK_FDPLL96M.

**Table 17-4. CLK_FDPLL96M behavior from start-up to first edge detection.**

| WUF | LTIME | CLK_FDPLL96M Behavior |
|---|---|---|
| 0 | 0 | Normal Mode: First Edge when lock is asserted |
| 0 | Not Equal To Zero | Lock Timer Timeout mode: First Edge when the timer downcounts to 0. |
| 1 | X | Wake Up Fast Mode: First Edge when CK is active (start-up time) |

**Table 17-5. CLK_FDPLL96M behavior after First Edge detection.**

| LBYPASS | CLK_FDPLL96M Behavior |
|---|---|
| 0 | Normal Mode: the CLK_FDPLL96M is turned off when lock signal is low. |
| 1 | Lock Bypass Mode: the CLK_FDPLL96M is always running, lock is irrelevant. |

**Figure 17-3. CK and CLK_FDPLL96M Off Mode to Running Mode**

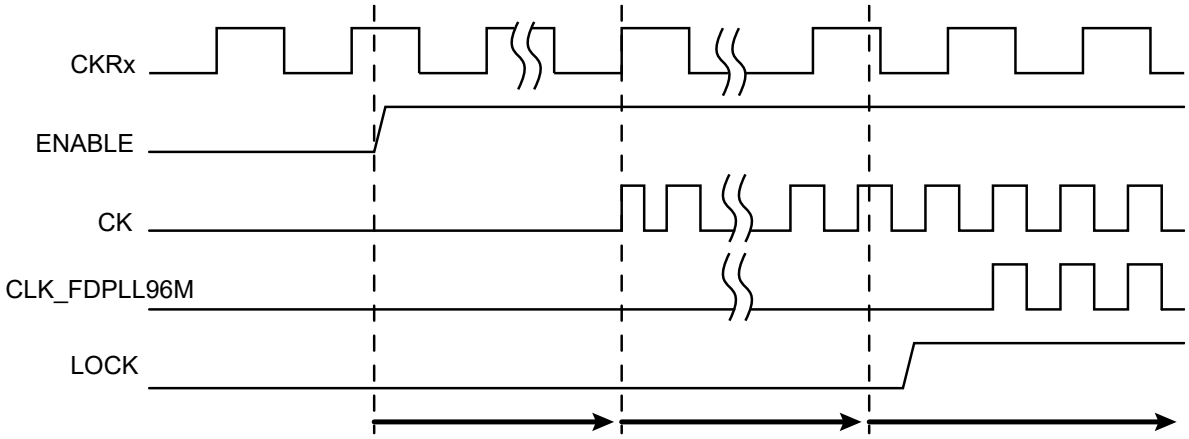



**Figure 17-4. CK and CLK_FDPLL96M Off Mode to Running Mode when Wake-Up Fast is Activated**

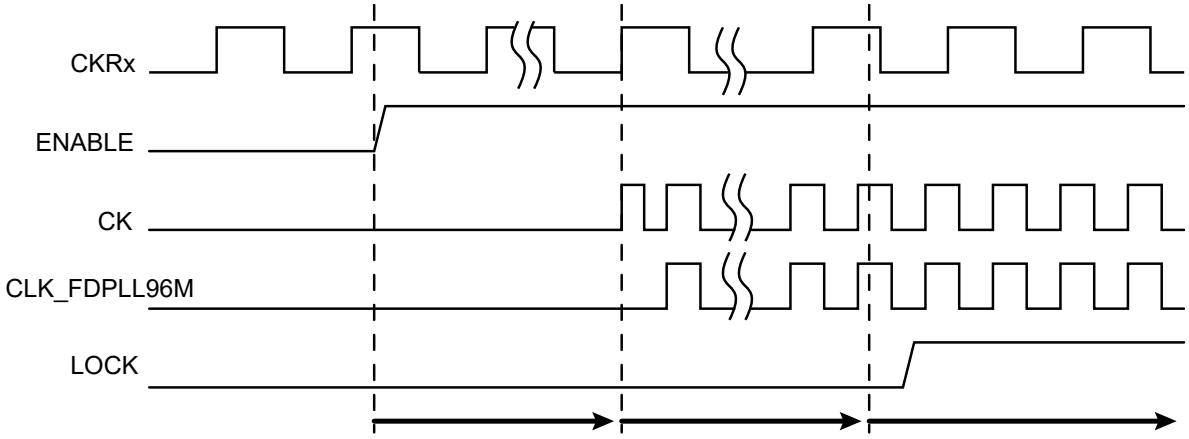
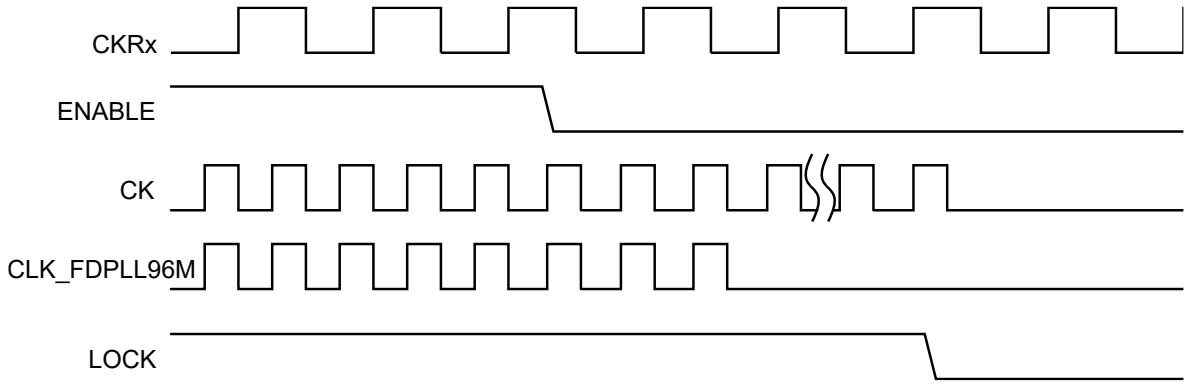



**Figure 17-5. CK and CLK_FDPLL96M Running Mode to Off Mode**



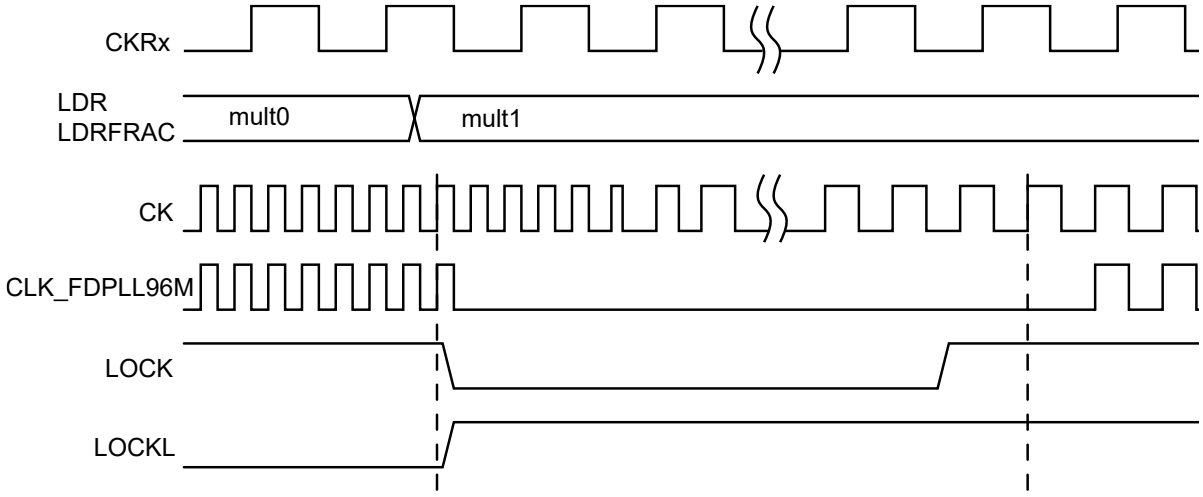
#### 17.6.8.5 Reference Clock Switching

When a software operation requires reference clock switching, the normal operation is to disable the FDPLL96M, modify the DPLLCTRLB.REFCLK to select the desired reference source and activate the FDPLL96M again.

#### 17.6.8.6 Loop Divider Ratio updates

The FDPLL96M supports on-the-fly update of the DPLLRATIO register, so it is allowed to modify the loop divider ratio and the loop divider ratio fractional part when the FDPLL96M is enabled. At that time, the DPLLSTATUS.LOCK bit is cleared and set again by hardware when the output frequency reached a stable state. The DPLL Lock Fail bit in the Interrupt Flag Status and Clear register (INTFLAG.DPLLLCK) is set when a falling edge has been detected. The flag is cleared when the software write a one to the interrupt flag bit location.

**Figure 17-6. RATIOCTRL Register Update Operation**




#### 17.6.8.7 Digital Filter Selection

The PLL digital filter (PI controller) is automatically adjusted in order to provide a good compromise between stability and jitter. Nevertheless a software operation can override the filter setting using the DPLLCTRLB.FILTER field. The DPLLCTRLB.LPEN field can be use to bypass the TDC module.

### 17.6.9 3.3V Brown-Out Detector Operation

The 3.3V BOD monitors the 3.3V VDDANA supply (BOD33). It supports continuous or sampling modes.

The threshold value action (reset the device or generate an interrupt), the Hysteresis configuration, as well as the enable/disable settings are loaded from Flash User Calibration at startup, and can be overridden by writing to the corresponding BOD33 register bit groups.

#### 17.6.9.1 3.3V Brown-Out Detector (BOD33)

The 3.3V Brown-Out Detector (BOD33) monitors the VDDANA supply and compares the voltage with the brown-out threshold level set in the BOD33 Level bit group (BOD33.LEVEL) in the BOD33 register. The BOD33 can generate either an interrupt or a reset when VDDANA crosses below the brown-out threshold level. The BOD33 detection status can be read from the BOD33 Detection bit (PCLKSR.BOD33DET) in the Power and Clocks Status register.

At start-up or at power-on reset (POR), the BOD33 register values are loaded from the Flash User Row. Refer to *NVM User Row Mapping* for more details.

**Related Links**

10.3.1 NVM User Row Mapping

#### 17.6.9.2 Continuous Mode

When the BOD33 Mode bit (BOD33.MODE) in the BOD33 register is written to zero and the BOD33 is enabled, the BOD33 operates in continuous mode. In this mode, the BOD33 is continuously monitoring the VDDANA supply voltage.

Continuous mode is the default mode for BOD33.

#### 17.6.9.3 Sampling Mode

The sampling mode is a low-power mode where the BOD33 or BOD12 is being repeatedly enabled on a sampling clock's ticks. The BOD33 or BOD12 will monitor the supply voltage for a short period of time and then go to a low-power disabled state until the next sampling clock tick.

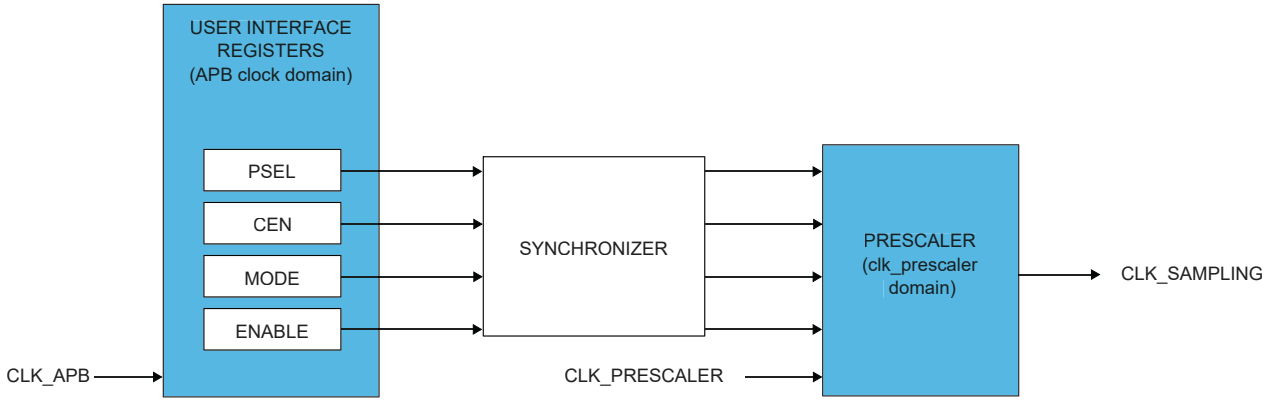
Sampling mode is enabled by writing one to BOD33.MODE for BOD33, and by writing one to BOD12.MODE for BOD12. The frequency of the clock ticks ($F_{clksampling}$) is controlled by the BOD33 Prescaler Select bit group (BOD33.PSEL) in the BOD33 register and Prescaler Select bit group(BOD12.PSEL) in the BOD12 BOD12 register for BOD33 and BOD12, respectively.

$$F_{\text{clksampling}} = \frac{F_{\text{clkprescaler}}}{2^{(\text{PSEL}+1)}}$$

The prescaler signal ($F_{clkprescaler}$) is a 1kHz clock, output from the32kHz Ultra Low Power Oscillator, OSCULP32K.

As the sampling mode clock is different from the APB clock domain, synchronization among the clocks is necessary. The next figure shows a block diagram of the sampling mode. The BOD33 and BOD12 Synchronization Ready bits (PCLKSR.B33SRDY and PCLKSR.B12SRDY, respectively) in the Power and Clocks Status register show the synchronization ready status of the synchronizer. Writing attempts to the BOD33 register are ignored while PCLKSR.B33SRDY is zero. Writing attempts to the BOD12 register are ignored while PCLKSR.B12SRDY is zero.

**Figure 17-7. Sampling Mode Block diagram**




The BOD33 Clock Enable bit (BOD33.CEN) in the BOD33 register and the BOD12 Clock Enable bit (BOD12.CEN) in the BOD12 register should always be disabled before changing the prescaler value. To change the prescaler value for the BOD33 or BOD12 during sampling mode, the following steps need to be taken:

1. Wait until the PCLKSR.B33SRDY bit or the PCLKSR.B12SRDY bit is set.
2. Write the selected value to the BOD33.PSEL or BOD12.PSEL bit group.

#### 17.6.9.4 Hysteresis

The hysteresis functionality can be used in both continuous and sampling mode. Writing a one to the BOD33 Hysteresis bit (BOD33.HYST) in the BOD33 register will add hysteresis to the BOD33 threshold

level. Writing a one to the BOD12 Hysteresis bit (BOD12.HYST) in the BOD12 register will add hysteresis to the BOD12 threshold level.

### 17.6.10 Voltage Reference System Operation

The Voltage Reference System (VREF) consists of a Bandgap Reference Voltage Generator and a temperature sensor.

The Bandgap Reference Voltage Generator is factory-calibrated under typical voltage and temperature conditions.

At reset, the VREF.CAL register value is loaded from Flash Factory Calibration.

The temperature sensor can be used to get an absolute temperature in the temperature range of CMIN to CMAX degrees Celsius. The sensor will output a linear voltage proportional to the temperature. The output voltage and temperature range are located in the *Electrical Characteristics*. To calculate the temperature from a measured voltage, the following formula can be used:

$$C_{\mathrm{MIN}} + (\mathrm{Vmes+} - \mathrm{Vout}_{\mathrm{MAX}})\frac{\Delta\mathrm{temperature}}{\Delta\mathrm{voltage}}$$

**Related Links**

37. Electrical Characteristics

#### 17.6.10.1 User Control of the Voltage Reference System

To enable the temperature sensor, write a one the Temperature Sensor Enable bit (VREF.TSEN) in the VREF register.

The temperature sensor can be redirected to the ADC for conversion.

The Bandgap Reference Voltage Generator output can also be routed to the ADC if the Bandgap Output Enable bit (VREF.BGOUTEN) in the VREF register is set.

The Bandgap Reference Voltage Generator output level is determined by the CALIB bit group (VREF.CALIB) value in the VREF register.The default calibration value can be overridden by the user by writing to the CALIB bit group.

### 17.6.11 Voltage Regulator System Operation

The embedded Voltage Regulator (VREG) is an internal voltage regulator that provides the core logic supply (VDDCORE).

### 17.6.12 DMA Operation

Not applicable.

### 17.6.13 Interrupts

The SYSCTRL has the following interrupt sources:

- XOSCRDY - Multipurpose Crystal Oscillator Ready: A “0-to-1” transition on the PCLKSR.XOSCRDY bit is detected
- XOSC32KRDY - 32kHz Crystal Oscillator Ready: A “0-to-1” transition on the PCLKSR.XOSC32KRDY bit is detected
- OSC32KRDY - 32kHz Internal Oscillator Ready: A “0-to-1” transition on the PCLKSR.OSC32KRDY bit is detected
- OSC8MRDY - 8MHz Internal Oscillator Ready: A “0-to-1” transition on the PCLKSR.OSC8MRDY bit is detected

- DFLLRDY - DFLL48M Ready: A “0-to-1” transition on the PCLKSR.DFLLRDY bit is detected
- DFLLOOB - DFLL48M Out Of Boundaries: A “0-to-1” transition on the PCLKSR.DFLLOOB bit is detected
- DFLLLOCKF - DFLL48M Fine Lock: A “0-to-1” transition on the PCLKSR.DFLLLOCKF bit is detected
- DFLLLOCKC - DFLL48M Coarse Lock: A “0-to-1” transition on the PCLKSR.DFLLLOCKC bit is detected
- DFLLRCS - DFLL48M Reference Clock has Stopped: A “0-to-1” transition on the PCLKSR.DFLLRCS bit is detected
- BOD33RDY - BOD33 Ready: A “0-to-1” transition on the PCLKSR.BOD33RDY bit is detected
- BOD33DET - BOD33 Detection: A “0-to-1” transition on the PCLKSR.BOD33DET bit is detected. This is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.
- B33SRDY - BOD33 Synchronization Ready: A “0-to-1” transition on the PCLKSR.B33SRDY bit is detected
- BOD12RDY - BOD12 Ready: A “0-to-1” transition on the PCLKSR.BOD12RDY bit is detected
- BOD12DET - BOD12 Detection: A “0-to-1” transition on the PCLKSR.BOD12DET bit is detected
- B12SRDY - BOD12 Synchronization Ready: A “0-to-1” transition on the PCLKSR.B12SRDY bit is detected
- PLL Lock (LOCK): Indicates that the DPLL Lock bit is asserted.
- PLL Lock Lost (LOCKL): Indicates that a falling edge has been detected on the Lock bit during normal operation mode.
- PLL Lock Timer Timeout (LTTO): This interrupt flag indicates that the software defined time DPLLCTRLB.LTIME has elapsed since the start of the FDPLL96M.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the SYSCTRL is reset. See Interrupt Flag Status and Clear (INTFLAG) register for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. Refer to *Nested Vector Interrupt Controller* for details. The user must read the INTFLAG register to determine which interrupt condition is present.

**Note:** Interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 17.6.14 Synchronization

Due to the multiple clock domains, values in the DFLL48M control registers need to be synchronized to other clock domains. The status of this synchronization can be read from the Power and Clocks Status register (PCLKSR). Before writing to any of the DFLL48M control registers, the user must check that the DFLL Ready bit (PCLKSR.DFLLRDY) in PCLKSR is set to one. When this bit is set, the DFLL48M can be configured and CLK_DFLL48M is ready to be used. Any write to any of the DFLL48M control registers

while DFLLRDY is zero will be ignored. An interrupt is generated on a zero-to-one transition of DFLLRDY if the DFLLRDY bit (INTENSET.DFLLDY) in the Interrupt Enable Set register is set.

In order to read from any of the DFLL48M configuration registers, the user must request a read synchronization by writing a one to DFLLSYNC.READREQ. The registers can be read only when PCLKSR.DFLLRDY is set. If DFLLSYNC.READREQ is not written before a read, a synchronization will be started, and the bus will be halted until the synchronization is complete. Reading the DFLL48M registers when the DFLL48M is disabled will not halt the bus.

The prescaler counter used to trigger one-shot brown-out detections also operates asynchronously from the peripheral bus. As a consequence, the prescaler registers require synchronization when written or read. The synchronization results in a delay from when the initialization of the write or read operation begins until the operation is complete.

The write-synchronization is triggered by a write to the BOD12 or BOD33 control register. The Synchronization Ready bit (PCLKSR.B12SRDY or PCLKSR.B33SRDY) in the PCLKSR register will be cleared when the write-synchronization starts and set when the write-synchronization is complete. When the write-synchronization is ongoing (PCLKSR.B33SRDY or PCLKSR.B12SRDY is zero), an attempt to do any of the following will cause the peripheral bus to stall until the synchronization is complete:

- Writing to the BOD33 or BOD12 control register
- Reading the BOD33 or BOD12 control register that was written

The user can either poll PCLKSR.B12SRDY or PCLKSR.B33SRDY or use the INTENSET.B12SRDY or INTENSET.B33SRDY interrupts to check when the synchronization is complete. It is also possible to perform the next read/write operation and wait, as this next operation will be completed after the ongoing read/write operation is synchronized.

## 17.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | INTENCLR | 7:0 | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| | | 15:8 | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| | | 23:16 | | | | | | | DPLLLTO | DPLLLCKF |
| | | 31:24 | | | | | | | | |
| 0x04 | INTENSET | 7:0 | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| | | 15:8 | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| | | 23:16 | | | | | | | DPLLLTO | DPLLLCKF |
| | | 31:24 | | | | | | | | |
| 0x08 | INTFLAG | 7:0 | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| | | 15:8 | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| | | 23:16 | | | | | | | DPLLLTO | DPLLLCKF |
| | | 31:24 | | | | | | | | |
| 0x0C | PCLKSR | 7:0 | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| | | 15:8 | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| | | 23:16 | | | | | | | DPLLLTO | DPLLLCKF |
| | | 31:24 | | | | | | | | |
| 0x10 | XOSC | 7:0 | ONDEMAND | RUNSTDBY | | | | XTALEN | ENABLE | |
| | | 15:8 | STARTUP[3:0] | | | | AMPGC | GAIN[2:0] | | |
| 0x12 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | XOSC32K | 7:0 | ONDEMAND | RUNSTDBY | AAMPEN | | EN32K | XTALEN | ENABLE | |
| | | 15:8 | | | | WRTLOCK | | STARTUP[2:0] | | |
| 0x16 ... 0x17 | Reserved | | | | | | | | | |
| 0x18 | OSC32K | 7:0 | ONDEMAND | RUNSTDBY | | | | EN32K | ENABLE | |
| | | 15:8 | | | | WRTLOCK | | STARTUP[2:0] | | |
| | | 23:16 | | CALIB[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | OSCULP32K | 7:0 | WRTLOCK | | | CALIB[4:0] | | | | |
| 0x1D ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | OSC8M | 7:0 | ONDEMAND | RUNSTDBY | | | | | ENABLE | |
| | | 15:8 | | | | | | | PRESC[1:0] | |
| | | 23:16 | CALIB[7:0] | | | | | | | |
| | | 31:24 | FRANGE[1:0] | | | | CALIB[11:8] | | | |
| 0x24 | DFLLCTRL | 7:0 | ONDEMAND | RUNSTDBY | USBCRM | LLAW | STABLE | MODE | ENABLE | |
| | | 15:8 | | | | | WAITLOCK | BPLCKC | QLDIS | CCDIS |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x26<br>...<br>0x27 | Reserved | | | | | | | | | |
| 0x28 | DFLLVAL | 7:0 | FINE[7:0] | | | | | | | |
| | | 15:8 | COARSE[5:0] | | | | | | FINE[9:8] | |
| | | 23:16 | DIFF[7:0] | | | | | | | |
| | | 31:24 | DIFF[15:8] | | | | | | | |
| 0x2C | DFLLMUL | 7:0 | MUL[7:0] | | | | | | | |
| | | 15:8 | MUL[15:8] | | | | | | | |
| | | 23:16 | FSTEP[7:0] | | | | | | | |
| | | 31:24 | CSTEP[5:0] | | | | | | FSTEP[9:8] | |
| 0x30 | DFLLSYNC | 7:0 | READREQ | | | | | | | |
| 0x31<br>...<br>0x33 | Reserved | | | | | | | | | |
| 0x34 | BOD33 | 7:0 | | RUNSTDBY | | ACTION[1:0] | | HYST | ENABLE | |
| | | 15:8 | PSEL[3:0] | | | | | | CEN | MODE |
| | | 23:16 | | | LEVEL[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38<br>...<br>0x3B | Reserved | | | | | | | | | |
| 0x3C | VREG | 7:0 | | RUNSTDBY | | | | | | |
| | | 15:8 | | | FORCELDO | | | | | |
| 0x3E<br>...<br>0x3F | Reserved | | | | | | | | | |
| 0x40 | VREF | 7:0 | | | | | | BGOUTEN | TSEN | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | CALIB[7:0] | | | | | | | |
| | | 31:24 | | | | | | CALIB[10:8] | | |
| 0x44 | DPLLCTRLA | 7:0 | ONDEMAND | RUNSTDBY | | | | | ENABLE | |
| 0x45<br>...<br>0x47 | Reserved | | | | | | | | | |
| 0x48 | DPLLRATIO | 7:0 | LDR[7:0] | | | | | | | |
| | | 15:8 | | | | | LDR[11:8] | | | |
| | | 23:16 | | | | | LDRFRAC[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | DPLLCTRLB | 7:0 | | | REFCLK[1:0] | | WUF | LPEN | FILTER[1:0] | |
| | | 15:8 | | | | LBYPASS | | LTIME[2:0] | | |
| | | 23:16 | DIV[7:0] | | | | | | | |
| | | 31:24 | | | | | | DIV[10:8] | | |
| 0x50 | DPLLSTATUS | 7:0 | | | | | DIV | ENABLE | CLKRDY | LOCK |

## 17.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

### 17.8.1 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DPLLLTO | DPLLLCKF |
| Access | R | R | R | R | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| Access | R/W | R | R | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – DPLLLTO** DPLL Lock Timeout Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DPLL Lock Timeout Interrupt Enable bit, which disables the DPLL Lock Timeout interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Timeout interrupt is disabled. |
| 1 | The DPLL Lock Timeout interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Timeout Interrupt flag is set. |

**Bit 16 – DPLLLCKF** DPLL Lock Fall Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DPLL Lock Fall Interrupt Enable bit, which disables the DPLL Lock Fall interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Fall interrupt is disabled. |
| 1 | The DPLL Lock Fall interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Fall Interrupt flag is set. |

**Bit 15 – DPLLLCKR** DPLL Lock Rise Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DPLL Lock Rise Interrupt Enable bit, which disables the DPLL Lock Rise interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Rise interrupt is disabled. |
| 1 | The DPLL Lock Rise interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Rise Interrupt flag is set. |

**Bit 11 – B33SRDY** BOD33 Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the BOD33 Synchronization Ready Interrupt Enable bit, which disables the BOD33 Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Synchronization Ready interrupt is disabled. |
| 1 | The BOD33 Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Synchronization Ready Interrupt flag is set. |

**Bit 10 – BOD33DET** BOD33 Detection Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the BOD33 Detection Interrupt Enable bit, which disables the BOD33 Detection interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Detection interrupt is disabled. |
| 1 | The BOD33 Detection interrupt is enabled, and an interrupt request will be generated when the BOD33 Detection Interrupt flag is set. |

**Bit 9 – BOD33RDY** BOD33 Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the BOD33 Ready Interrupt Enable bit, which disables the BOD33 Ready interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Ready interrupt is disabled. |
| 1 | The BOD33 Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Ready Interrupt flag is set. |

**Bit 8 – DFLLRCS** DFLL Reference Clock Stopped Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DFLL Reference Clock Stopped Interrupt Enable bit, which disables the DFLL Reference Clock Stopped interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Reference Clock Stopped interrupt is disabled. |
| 1 | The DFLL Reference Clock Stopped interrupt is enabled, and an interrupt request will be generated when the DFLL Reference Clock Stopped Interrupt flag is set. |

**Bit 7 – DFLLLCKC** DFLL Lock Coarse Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DFLL Lock Coarse Interrupt Enable bit, which disables the DFLL Lock Coarse interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Lock Coarse interrupt is disabled. |
| 1 | The DFLL Lock Coarse interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Coarse Interrupt flag is set. |

**Bit 6 – DFLLLCKF** DFLL Lock Fine Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DFLL Lock Fine Interrupt Enable bit, which disables the DFLL Lock Fine interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Lock Fine interrupt is disabled. |
| 1 | The DFLL Lock Fine interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Fine Interrupt flag is set. |

**Bit 5 – DFLLOOB** DFLL Out Of Bounds Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DFLL Out Of Bounds Interrupt Enable bit, which disables the DFLL Out Of Bounds interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Out Of Bounds interrupt is disabled. |
| 1 | The DFLL Out Of Bounds interrupt is enabled, and an interrupt request will be generated when the DFLL Out Of Bounds Interrupt flag is set. |

**Bit 4 – DFLLRDY** DFLL Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the DFLL Ready Interrupt Enable bit, which disables the DFLL Ready interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Ready interrupt is disabled. |
| 1 | The DFLL Ready interrupt is enabled, and an interrupt request will be generated when the DFLL Ready Interrupt flag is set. |

**Bit 3 – OSC8MRDY** OSC8M Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the OSC8M Ready Interrupt Enable bit, which disables the OSC8M Ready interrupt.

| Value | Description |
|---|---|
| 0 | The OSC8M Ready interrupt is disabled. |
| 1 | The OSC8M Ready interrupt is enabled, and an interrupt request will be generated when the OSC8M Ready Interrupt flag is set. |

**Bit 2 – OSC32KRDY** OSC32K Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the OSC32K Ready Interrupt Enable bit, which disables the OSC32K Ready interrupt.

| Value | Description |
|---|---|
| 0 | The OSC32K Ready interrupt is disabled. |
| 1 | The OSC32K Ready interrupt is enabled, and an interrupt request will be generated when the OSC32K Ready Interrupt flag is set. |

**Bit 1 – XOSC32KRDY** XOSC32K Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the XOSC32K Ready Interrupt Enable bit, which disables the XOSC32K Ready interrupt.

| Value | Description |
|---|---|
| 0 | The XOSC32K Ready interrupt is disabled. |
| 1 | The XOSC32K Ready interrupt is enabled, and an interrupt request will be generated when the XOSC32K Ready Interrupt flag is set. |

**Bit 0 – XOSCRDY** XOSC Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the XOSC Ready Interrupt Enable bit, which disables the XOSC Ready interrupt.

| Value | Description |
|---|---|
| 0 | The XOSC Ready interrupt is disabled. |
| 1 | The XOSC Ready interrupt is enabled, and an interrupt request will be generated when the XOSC Ready Interrupt flag is set. |

### 17.8.2 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DPLLLTO | DPLLLCKF |
| Access | R | R | R | R | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| Access | R/W | R | R | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – DPLLLTO** DPLL Lock Timeout Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DPLL Lock Timeout Interrupt Enable bit, which enables the DPLL Lock Timeout interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Timeout interrupt is disabled. |
| 1 | The DPLL Lock Timeout interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Timeout Interrupt flag is set. |

**Bit 16 – DPLLLCKF** DPLL Lock Fall Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DPLL Lock Fall Interrupt Enable bit, which enables the DPLL Lock Fall interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Fall interrupt is disabled. |
| 1 | The DPLL Lock Fall interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Fall Interrupt flag is set. |

**Bit 15 – DPLLLCKR** DPLL Lock Rise Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DPLL Lock Rise Interrupt Enable bit, which enables the DPLL Lock Rise interrupt.

| Value | Description |
|---|---|
| 0 | The DPLL Lock Rise interrupt is disabled. |
| 1 | The DPLL Lock Rise interrupt is enabled, and an interrupt request will be generated when the DPLL Lock Rise Interrupt flag is set. |

**Bit 11 – B33SRDY** BOD33 Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the BOD33 Synchronization Ready Interrupt Enable bit, which enables the BOD33 Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Synchronization Ready interrupt is disabled. |
| 1 | The BOD33 Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Synchronization Ready Interrupt flag is set. |

**Bit 10 – BOD33DET** BOD33 Detection Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the BOD33 Detection Interrupt Enable bit, which enables the BOD33 Detection interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Detection interrupt is disabled. |
| 1 | The BOD33 Detection interrupt is enabled, and an interrupt request will be generated when the BOD33 Detection Interrupt flag is set. |

**Bit 9 – BOD33RDY** BOD33 Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the BOD33 Ready Interrupt Enable bit, which enables the BOD33 Ready interrupt.

| Value | Description |
|---|---|
| 0 | The BOD33 Ready interrupt is disabled. |
| 1 | The BOD33 Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Ready Interrupt flag is set. |

**Bit 8 – DFLLRCS** DFLL Reference Clock Stopped Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DFLL Reference Clock Stopped Interrupt Enable bit, which enables the DFLL Reference Clock Stopped interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Reference Clock Stopped interrupt is disabled. |
| 1 | The DFLL Reference Clock Stopped interrupt is enabled, and an interrupt request will be generated when the DFLL Reference Clock Stopped Interrupt flag is set. |

**Bit 7 – DFLLLCKC** DFLL Lock Coarse Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DFLL Lock Coarse Interrupt Enable bit, which enables the DFLL Lock Coarse interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Lock Coarse interrupt is disabled. |
| 1 | The DFLL Lock Coarse interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Coarse Interrupt flag is set. |

**Bit 6 – DFLLLCKF** DFLL Lock Fine Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DFLL Lock Fine Interrupt Disable/Enable bit, disable the DFLL Lock Fine interrupt and set the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The DFLL Lock Fine interrupt is disabled. |
| 1 | The DFLL Lock Fine interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Fine Interrupt flag is set. |

**Bit 5 – DFLLOOB** DFLL Out Of Bounds Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DFLL Out Of Bounds Interrupt Enable bit, which enables the DFLL Out Of Bounds interrupt.

| Value | Description |
|---|---|
| 0 | The DFLL Out Of Bounds interrupt is disabled. |
| 1 | The DFLL Out Of Bounds interrupt is enabled, and an interrupt request will be generated when the DFLL Out Of Bounds Interrupt flag is set. |

**Bit 4 – DFLLRDY** DFLL Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the DFLL Ready Interrupt Enable bit, which enables the DFLL Ready interrupt and set the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The DFLL Ready interrupt is disabled. |
| 1 | The DFLL Ready interrupt is enabled, and an interrupt request will be generated when the DFLL Ready Interrupt flag is set. |

**Bit 3 – OSC8MRDY** OSC8M Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the OSC8M Ready Interrupt Enable bit, which enables the OSC8M Ready interrupt.

| Value | Description |
|---|---|
| 0 | The OSC8M Ready interrupt is disabled. |
| 1 | The OSC8M Ready interrupt is enabled, and an interrupt request will be generated when the OSC8M Ready Interrupt flag is set. |

**Bit 2 – OSC32KRDY** OSC32K Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the OSC32K Ready Interrupt Enable bit, which enables the OSC32K Ready interrupt.

| Value | Description |
|---|---|
| 0 | The OSC32K Ready interrupt is disabled. |
| 1 | The OSC32K Ready interrupt is enabled, and an interrupt request will be generated when the OSC32K Ready Interrupt flag is set. |

**Bit 1 – XOSC32KRDY** XOSC32K Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the XOSC32K Ready Interrupt Enable bit, which enables the XOSC32K Ready interrupt.

| Value | Description |
|---|---|
| 0 | The XOSC32K Ready interrupt is disabled. |
| 1 | The XOSC32K Ready interrupt is enabled, and an interrupt request will be generated when the XOSC32K Ready Interrupt flag is set. |

**Bit 0 – XOSCRDY** XOSC Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the XOSC Ready Interrupt Enable bit, which enables the XOSC Ready interrupt.

| Value | Description |
|---|---|
| 0 | The XOSC Ready interrupt is disabled. |
| 1 | The XOSC Ready interrupt is enabled, and an interrupt request will be generated when the XOSC Ready Interrupt flag is set. |

### 17.8.3 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** -

**Note:** Depending on the fuse settings, various bits of the INTFLAG register can be set to one at startup. Therefore the user should clear those bits before using the corresponding interrupts.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DPLLLTO | DPLLLCKF |
| Access | R | R | R | R | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| Access | R/W | R | R | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – DPLLLTO** DPLL Lock Timeout
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DPLL Lock Timeout bit in the Status register (PCLKSR.DPLLLTO) and will generate an interrupt request if INTENSET.DPLLLTO is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DPLL Lock Timeout interrupt flag.

**Bit 16 – DPLLLCKF** DPLL Lock Fall
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DPLL Lock Fall bit in the Status register (PCLKSR.DPLLLCKF) and will generate an interrupt request if INTENSET.DPLLLCKF is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DPLL Lock Fall interrupt flag.

**Bit 15 – DPLLLCKR** DPLL Lock Rise
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DPLL Lock Rise bit in the Status register (PCLKSR.DPLLLCKR) and will generate an interrupt request if INTENSET.DPLLLCKR is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DPLL Lock Rise interrupt flag.

**Bit 11 – B33SRDY** BOD33 Synchronization Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the BOD33 Synchronization Ready bit in the Status register (PCLKSR.B33SRDY) and will generate an interrupt request if INTENSET.B33SRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the BOD33 Synchronization Ready interrupt flag

**Bit 10 – BOD33DET** BOD33 Detection
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the BOD33 Detection bit in the Status register (PCLKSR.BOD33DET) and will generate an interrupt request if INTENSET.BOD33DET is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the BOD33 Detection interrupt flag.

**Bit 9 – BOD33RDY** BOD33 Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the BOD33 Ready bit in the Status register (PCLKSR.BOD33RDY) and will generate an interrupt request if INTENSET.BOD33RDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the BOD33 Ready interrupt flag.

**Bit 8 – DFLLRCS** DFLL Reference Clock Stopped
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DFLL Reference Clock Stopped bit in the Status register (PCLKSR.DFLLRCS) and will generate an interrupt request if INTENSET.DFLLRCS is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DFLL Reference Clock Stopped interrupt flag.

**Bit 7 – DFLLLCKC** DFLL Lock Coarse
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DFLL Lock Coarse bit in the Status register (PCLKSR.DFLLLCKC) and will generate an interrupt request if INTENSET.DFLLLCKC is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DFLL Lock Coarse interrupt flag.

**Bit 6 – DFLLLCKF** DFLL Lock Fine
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DFLL Lock Fine bit in the Status register (PCLKSR.DFLLLCKF) and will generate an interrupt request if INTENSET.DFLLLCKF is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DFLL Lock Fine interrupt flag.

**Bit 5 – DFLLOOB** DFLL Out Of Bounds
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DFLL Out Of Bounds bit in the Status register (PCLKSR.DFLLOOB) and will generate an interrupt request if INTENSET.DFLLOOB is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DFLL Out Of Bounds interrupt flag.

**Bit 4 – DFLLRDY** DFLL Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the DFLL Ready bit in the Status register (PCLKSR.DFLLRDY) and will generate an interrupt request if INTENSET.DFLLRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DFLL Ready interrupt flag.

**Bit 3 – OSC8MRDY** OSC8M Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the OSC8M Ready bit in the Status register (PCLKSR.OSC8MRDY) and will generate an interrupt request if INTENSET.OSC8MRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the OSC8M Ready interrupt flag.

**Bit 2 – OSC32KRDY** OSC32K Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the OSC32K Ready bit in the Status register (PCLKSR.OSC32KRDY) and will generate an interrupt request if INTENSET.OSC32KRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the OSC32K Ready interrupt flag.

**Bit 1 – XOSC32KRDY** XOSC32K Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the XOSC32K Ready bit in the Status register (PCLKSR.XOSC32KRDY) and will generate an interrupt request if INTENSET.XOSC32KRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the XOSC32K Ready interrupt flag.

**Bit 0 – XOSCRDY** XOSC Ready
This flag is cleared by writing a one to it.

This flag is set on a zero-to-one transition of the XOSC Ready bit in the Status register (PCLKSR.XOSCRDY) and will generate an interrupt request if INTENSET.XOSCRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the XOSC Ready interrupt flag.

© 2018 Microchip Technology Inc.

### 17.8.4 Power and Clocks Status

**Name:** PCLKSR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DPLLLTO | DPLLLCKF |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DPLLLCKR | | | | B33SRDY | BOD33DET | BOD33RDY | DFLLRCS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DFLLLCKC | DFLLLCKF | DFLLOOB | DFLLRDY | OSC8MRDY | OSC32KRDY | XOSC32KRDY | XOSCRDY |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – DPLLLTO** DPLL Lock Timeout

| Value | Description |
|---|---|
| 0 | DPLL Lock time-out not detected. |
| 1 | DPLL Lock time-out detected. |

**Bit 16 – DPLLLCKF** DPLL Lock Fall

| Value | Description |
|---|---|
| 0 | DPLL Lock fall edge not detected. |
| 1 | DPLL Lock fall edge detected. |

**Bit 15 – DPLLLCKR** DPLL Lock Rise

| Value | Description |
|---|---|
| 0 | DPLL Lock rise edge not detected. |
| 1 | DPLL Lock fall edge detected. |

**Bit 11 – B33SRDY** BOD33 Synchronization Ready

| Value | Description |
|---|---|
| 0 | BOD33 synchronization is complete. |
| 1 | BOD33 synchronization is ongoing. |

**Bit 10 – BOD33DET** BOD33 Detection

| Value | Description |
|---|---|
| 0 | No BOD33 detection. |
| 1 | BOD33 has detected that the I/O power supply is going below the BOD33 reference value. |

**Bit 9 – BOD33RDY** BOD33 Ready

| Value | Description |
|---|---|
| 0 | BOD33 is not ready. |
| 1 | BOD33 is ready. |

**Bit 8 – DFLLRCS** DFLL Reference Clock Stopped

| Value | Description |
|---|---|
| 0 | DFLL reference clock is running. |
| 1 | DFLL reference clock has stopped. |

**Bit 7 – DFLLLCKC** DFLL Lock Coarse

| Value | Description |
|---|---|
| 0 | No DFLL coarse lock detected. |
| 1 | DFLL coarse lock detected. |

**Bit 6 – DFLLLCKF** DFLL Lock Fine

| Value | Description |
|---|---|
| 0 | No DFLL fine lock detected. |
| 1 | DFLL fine lock detected. |

**Bit 5 – DFLLOOB** DFLL Out Of Bounds

| Value | Description |
|---|---|
| 0 | No DFLL Out Of Bounds detected. |
| 1 | DFLL Out Of Bounds detected. |

**Bit 4 – DFLLRDY** DFLL Ready
This bit is cleared when the synchronization of registers between clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

| Value | Description |
|---|---|
| 0 | The Synchronization is ongoing. |
| 1 | The Synchronization is complete. |

**Bit 3 – OSC8MRDY** OSC8M Ready

| Value | Description |
|---|---|
| 0 | OSC8M is not ready. |
| 1 | OSC8M is stable and ready to be used as a clock source. |

**Bit 2 – OSC32KRDY** OSC32K Ready

| Value | Description |
|---|---|
| 0 | OSC32K is not ready. |
| 1 | OSC32K is stable and ready to be used as a clock source. |

**Bit 1 – XOSC32KRDY** XOSC32K Ready

| Value | Description |
|---|---|
| 0 | XOSC32K is not ready. |
| 1 | XOSC32K is stable and ready to be used as a clock source. |

**Bit 0 – XOSCRDY** XOSC Ready

| Value | Description |
|---|---|
| 0 | XOSC is not ready. |
| 1 | XOSC is stable and ready to be used as a clock source. |

### 17.8.5 External Multipurpose Crystal Oscillator (XOSC) Control

**Name:** XOSC
**Offset:** 0x10
**Reset:** 0x0080
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | STARTUP[3:0] | | | | AMPGC | GAIN[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | | | | XTALEN | ENABLE | |
| Access | R/W | R/W | | | | R/W | R/W | |
| Reset | 1 | 0 | | | | 0 | 0 | |

**Bits 15:12 – STARTUP[3:0]** Start-Up Time
These bits select start-up time for the oscillator according to the table below.

The OSCULP32K oscillator is used to clock the start-up counter.

| STARTUP[3:0] | Number of OSCULP32K Clock Cycles | Number of XOSC Clock Cycles | Approximate Equivalent Time(1)(2)(3) |
|---|---|---|---|
| 0x0 | 1 | 3 | 31µs |
| 0x1 | 2 | 3 | 61µs |
| 0x2 | 4 | 3 | 122µs |
| 0x3 | 8 | 3 | 244µs |
| 0x4 | 16 | 3 | 488µs |
| 0x5 | 32 | 3 | 977µs |
| 0x6 | 64 | 3 | 1953µs |
| 0x7 | 128 | 3 | 3906µs |
| 0x8 | 256 | 3 | 7813µs |
| 0x9 | 512 | 3 | 15625µs |
| 0xA | 1024 | 3 | 31250µs |
| 0xB | 2048 | 3 | 62500µs |
| 0xC | 4096 | 3 | 125000µs |
| 0xD | 8192 | 3 | 250000µs |
| 0xE | 16384 | 3 | 500000µs |
| 0xF | 32768 | 3 | 1000000µs |

**Note:**

1. Number of cycles for the start-up counter
2. Number of cycles for the synchronization delay, before PCLKSR.XOSCRDY is set.
3. Actual start-up time is n OSCULP32K cycles + 3 XOSC cycles, but given the time neglects the 3 XOSC cycles.

**Bit 11 – AMPGC** Automatic Amplitude Gain Control
This bit must be set only after the XOSC has settled, indicated by the XOSC Ready flag in the PCLKSR register (PCLKSR.XOSCRDY).

| Value | Description |
|---|---|
| 0 | The automatic amplitude gain control is disabled. |
| 1 | The automatic amplitude gain control is enabled. Amplitude gain will be automatically adjusted during Crystal Oscillator operation. |

**Bits 10:8 – GAIN[2:0]** Oscillator Gain
These bits select the gain for the oscillator. The listed maximum frequencies are recommendations, and might vary based on capacitive load and crystal characteristics. Setting this bit group has no effect when the Automatic Amplitude Gain Control is active.

| GAIN[2:0] | Recommended Max Frequency |
|---|---|
| 0x0 | 2MHz |
| 0x1 | 4MHz |
| 0x2 | 8MHz |
| 0x3 | 16MHz |
| 0x4 | 30MHz |
| 0x5-0x7 | Reserved |

**Bit 7 – ONDEMAND** On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled, depending on peripheral clock requests.

In On Demand operation mode, i.e., if the XOSC.ONDEMAND bit has been previously written to one, the oscillator will be running only when requested by a peripheral. If there is no peripheral requesting the oscillator s clock source, the oscillator will be in a disabled state.

If On Demand is disabled, the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the XOSC.RUNSTDBY bit is one. If XOSC.RUNSTDBY is zero, the oscillator is disabled.

| Value | Description |
|---|---|
| 0 | The oscillator is always on, if enabled. |
| 1 | The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source. |

**Bit 6 – RUNSTDBY** Run in Standby
This bit controls how the XOSC behaves during standby sleep mode:

| Value | Description |
|---|---|
| 0 | The oscillator is disabled in standby sleep mode. |
| 1 | The oscillator is not stopped in standby sleep mode. If XOSC.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If XOSC.ONDEMAND is zero, the clock source will always be running in standby sleep mode. |

**Bit 2 – XTALEN** Crystal Oscillator Enable
This bit controls the connections between the I/O pads and the external clock or crystal oscillator:

| Value | Description |
|---|---|
| 0 | External clock connected on XIN. XOUT can be used as general-purpose I/O. |
| 1 | Crystal connected to XIN/XOUT. |

**Bit 1 – ENABLE** Oscillator Enable

| Value | Description |
|---|---|
| 0 | The oscillator is disabled. |
| 1 | The oscillator is enabled. |

### 17.8.6 32kHz External Crystal Oscillator (XOSC32K) Control

**Name:** XOSC32K
**Offset:** 0x14
**Reset:** 0x0080
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | WRTLOCK | | STARTUP[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | AAMPEN | | EN32K | XTALEN | ENABLE | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | |
| Reset | 1 | 0 | 0 | | 0 | 0 | 0 | |

**Bit 12 – WRTLOCK** Write Lock
This bit locks the XOSC32K register for future writes to fix the XOSC32K configuration.

| Value | Description |
|---|---|
| 0 | The XOSC32K configuration is not locked. |
| 1 | The XOSC32K configuration is locked. |

**Bits 10:8 – STARTUP[2:0]** Oscillator Start-Up Time
These bits select the start-up time for the oscillator.

The OSCULP32K oscillator is used to clock the start-up counter.

**Table 17-6. Start-Up Time for 32kHz External Crystal Oscillator**

| STARTUP[2:0] | Number of OSCULP32K Clock Cycles | Number of XOSC32K Clock Cycles | Approximate Equivalent Time (OSCULP = 32kHz)(1)(2)(3) |
|---|---|---|---|
| 0x0 | 1 | 3 | 122µs |
| 0x1 | 32 | 3 | 1068µs |
| 0x2 | 2048 | 3 | 62592µs |
| 0x3 | 4096 | 3 | 125092µs |
| 0x4 | 16384 | 3 | 500092µs |
| 0x5 | 32768 | 3 | 1000092µs |
| 0x6 | 65536 | 3 | 2000092µs |
| 0x7 | 131072 | 3 | 4000092µs |

Notes: 1. Number of cycles for the start-up counter.

2. Number of cycles for the synchronization delay, before PCLKSR.XOSC32KRDY is set.

3. Start-up time is n OSCULP32K cycles + 3 XOSC32K cycles.

**Bit 7 – ONDEMAND** On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator s clock source, the oscillator will be in a disabled state.

If On Demand is disabled the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the XOSC32K.RUNSTDBY bit is one. If XOSC32K.RUNSTDBY is zero, the oscillator is disabled.

| Value | Description |
|---|---|
| 0 | The oscillator is always on, if enabled. |
| 1 | The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source. |

**Bit 6 – RUNSTDBY** Run in Standby
This bit controls how the XOSC32K behaves during standby sleep mode:

| Value | Description |
|---|---|
| 0 | The oscillator is disabled in standby sleep mode. |
| 1 | The oscillator is not stopped in standby sleep mode. If XOSC32K.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If XOSC32K.ONDEMAND is zero, the clock source will always be running in standby sleep mode. |

**Bit 5 – AAMPEN** Automatic Amplitude Control Enable

| Value | Description |
|---|---|
| 0 | The automatic amplitude control for the crystal oscillator is disabled. |
| 1 | The automatic amplitude control for the crystal oscillator is enabled. |

**Bit 3 – EN32K** 32kHz Output Enable
This bit controls the connections between the I/O pads and the external clock or crystal oscillator:

| Value | Description |
|---|---|
| 0 | The 32kHz output is disabled. |
| 1 | The 32kHz output is enabled. |

**Bit 2 – XTALEN** Crystal Oscillator Enable
This bit controls the connections between the I/O pads and the external clock or crystal oscillator:

| Value | Description |
|---|---|
| 0 | External clock connected on XIN32. XOUT32 can be used as general-purpose I/O. |
| 1 | Crystal connected to XIN32/XOUT32. |

**Bit 1 – ENABLE** Oscillator Enable

| Value | Description |
|---|---|
| 0 | The oscillator is disabled. |
| 1 | The oscillator is enabled. |

### 17.8.7 32kHz Internal Oscillator (OSC32K) Control

**Name:** OSC32K
**Offset:** 0x18
**Reset:** 0x003F0080
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CALIB[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | WRTLOCK | | STARTUP[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | | | | EN32K | ENABLE | |
| Access | R/W | R/W | | | | R/W | R/W | |
| Reset | 1 | 0 | | | | 0 | 0 | |

**Bits 22:16 – CALIB[6:0]** Oscillator Calibration
These bits control the oscillator calibration.

This value must be written by the user.

Factory calibration values can be loaded from the non-volatile memory.

**Bit 12 – WRTLOCK** Write Lock
This bit locks the OSC32K register for future writes to fix the OSC32K configuration.

| Value | Description |
|---|---|
| 0 | The OSC32K configuration is not locked. |
| 1 | The OSC32K configuration is locked. |

**Bits 10:8 – STARTUP[2:0]** Oscillator Start-Up Time
These bits select start-up time for the oscillator.

The OSCULP32K oscillator is used as input clock to the startup counter.

**Table 17-7. Start-Up Time for 32kHz Internal Oscillator**

| STARTUP[2:0] | Number of OSC32K clock cycles | Approximate Equivalent Time (OSCULP= 32 kHz)(1)(2)(3) |
|---|---|---|
| 0x0 | 3 | 92µs |
| 0x1 | 4 | 122µs |
| 0x2 | 6 | 183µs |
| 0x3 | 10 | 305µs |
| 0x4 | 18 | 549µs |
| 0x5 | 34 | 1038µs |
| 0x6 | 66 | 2014µs |
| 0x7 | 130 | 3967µs |

Notes: 1. Number of cycles for the start-up counter.

2. Number of cycles for the synchronization delay, before PCLKSR.OSC32KRDY is set.

3. Start-up time is n OSC32K cycles + 2 OSC32K cycles.

**Bit 7 – ONDEMAND** On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator s clock source, the oscillator will be in a disabled state.

If On Demand is disabled the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the OSC32K.RUNSTDBY bit is one. If OSC32K.RUNSTDBY is zero, the oscillator is disabled.

| Value | Description |
|---|---|
| 0 | The oscillator is always on, if enabled. |
| 1 | The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source. |

**Bit 6 – RUNSTDBY** Run in Standby
This bit controls how the OSC32K behaves during standby sleep mode:

| Value | Description |
|---|---|
| 0 | The oscillator is disabled in standby sleep mode. |
| 1 | The oscillator is not stopped in standby sleep mode. If OSC32K.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If OSC32K.ONDEMAND is zero, the clock source will always be running in standby sleep mode. |

**Bit 2 – EN32K** 32kHz Output Enable

| Value | Description |
|---|---|
| 0 | The 32kHz output is disabled. |
| 1 | The 32kHz output is enabled. |
| 0 | The oscillator is disabled. |
| 1 | The oscillator is enabled. |

**Bit 1 – ENABLE** Oscillator Enable

| Value | Description |
|---|---|
| 0 | The oscillator is disabled. |
| 1 | The oscillator is enabled. |

### 17.8.8 32kHz Ultra Low Power Internal Oscillator (OSCULP32K) Control

**Name:** OSCULP32K
**Offset:** 0x1C
**Reset:** 0xXX
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WRTLOCK | | | CALIB[4:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | x | x | x | x | x |

**Bit 7 – WRTLOCK** Write Lock
This bit locks the OSCULP32K register for future writes to fix the OSCULP32K configuration.

| Value | Description |
|---|---|
| 0 | The OSCULP32K configuration is not locked. |
| 1 | The OSCULP32K configuration is locked. |

**Bits 4:0 – CALIB[4:0]** Oscillator Calibration
These bits control the oscillator calibration.

These bits are loaded from Flash Calibration at startup.

### 17.8.9 8MHz Internal Oscillator (OSC8M) Control

**Name:** OSC8M
**Offset:** 0x20
**Reset:** 0xXXXX0382
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FRANGE[1:0] | | | | CALIB[11:8] | | | |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | x | x | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CALIB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PRESC[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | | | | | ENABLE | |
| Access | R/W | R/W | | | | | R/W | |
| Reset | 1 | 0 | | | | | 1 | |

**Bits 31:30 – FRANGE[1:0]** Oscillator Frequency Range
These bits control the oscillator frequency range according to the table below. These bits are loaded from Flash Calibration at startup.

| FRANGE[1:0] | Description |
|---|---|
| 0x0 | 4 to 6MHz |
| 0x1 | 6 to 8MHz |
| 0x2 | 8 to 11MHz |
| 0x3 | 11 to 15MHz |

**Bits 27:16 – CALIB[11:0]** Oscillator Calibration
These bits control the oscillator calibration. The calibration field is split in two:

CALIB[11:6] is for temperature calibration

CALIB[5:0] is for overall process calibration

These bits are loaded from Flash Calibration at startup.

**Bits 9:8 – PRESC[1:0]** Oscillator Prescaler
These bits select the oscillator prescaler factor setting according to the table below.

| PRESC[1:0] | Description |
|---|---|
| 0x0 | 1 |
| 0x1 | 2 |
| 0x2 | 4 |
| 0x3 | 8 |

**Bit 7 – ONDEMAND** On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.

If On Demand is disabled the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the OSC8M.RUNSTDBY bit is one. If OSC8M.RUNSTDBY is zero, the oscillator is disabled.

| Value | Description |
|---|---|
| 0 | The oscillator is always on, if enabled. |
| 1 | The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source. |

**Bit 6 – RUNSTDBY** Run in Standby
This bit controls how the OSC8M behaves during standby sleep mode:

| Value | Description |
|---|---|
| 0 | The oscillator is disabled in standby sleep mode. |
| 1 | The oscillator is not stopped in standby sleep mode. If OSC8M.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If OSC8M.ONDEMAND is zero, the clock source will always be running in standby sleep mode. |

**Bit 1 – ENABLE** Oscillator Enable
The user must ensure that the OSC8M is fully disabled before enabling it, and that the OSC8M is fully enabled before disabling it by reading OSC8M.ENABLE.

| Value | Description |
|---|---|
| 0 | The oscillator is disabled or being enabled. |
| 1 | The oscillator is enabled or being disabled. |

### 17.8.10 DFLL48M Control

**Name:** DFLLCTRL
**Offset:** 0x24
**Reset:** 0x0080
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | WAITLOCK | BPLCKC | QLDIS | CCDIS |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | USBCRM | LLAW | STABLE | MODE | ENABLE | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 11 – WAITLOCK** Wait Lock
This bit controls the DFLL output clock, depending on lock status:

| Value | Description |
|---|---|
| 0 | Output clock before the DFLL is locked. |
| 1 | Output clock when DFLL is locked. |

**Bit 10 – BPLCKC** Bypass Coarse Lock
This bit controls the coarse lock procedure:

| Value | Description |
|---|---|
| 0 | Bypass coarse lock is disabled. |
| 1 | Bypass coarse lock is enabled. |

**Bit 9 – QLDIS** Quick Lock Disable

| Value | Description |
|---|---|
| 0 | Quick Lock is enabled. |
| 1 | Quick Lock is disabled. |

**Bit 8 – CCDIS** Chill Cycle Disable

| Value | Description |
|---|---|
| 0 | Chill Cycle is enabled. |
| 1 | Chill Cycle is disabled. |

**Bit 7 – ONDEMAND** On Demand Control
The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator s clock source, the oscillator will be in a disabled state.

If On Demand is disabled the oscillator will always be running when enabled.

In standby sleep mode, the On Demand operation is still active if the DFLLCTRL.RUNSTDBY bit is one. If DFLLCTRL.RUNSTDBY is zero, the oscillator is disabled.

| Value | Description |
|---|---|
| 0 | The oscillator is always on, if enabled. |
| 1 | The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source. |

**Bit 6 – RUNSTDBY** Run in Standby
This bit controls how the DFLL behaves during standby sleep mode:

| Value | Description |
|---|---|
| 0 | The oscillator is disabled in standby sleep mode. |
| 1 | The oscillator is not stopped in standby sleep mode. If DFLLCTRL.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If DFLLCTRL.ONDEMAND is zero, the clock source will always be running in standby sleep mode. |

**Bit 5 – USBCRM** USB Clock Recovery Mode

| Value | Description |
|---|---|
| 0 | USB Clock Recovery Mode is disabled. |
| 1 | USB Clock Recovery Mode is enabled. |

**Bit 4 – LLAW** Lose Lock After Wake

| Value | Description |
|---|---|
| 0 | Locks will not be lost after waking up from sleep modes if the DFLL clock has been stopped. |
| 1 | Locks will be lost after waking up from sleep modes if the DFLL clock has been stopped. |

**Bit 3 – STABLE** Stable DFLL Frequency

| Value | Description |
|---|---|
| 0 | FINE calibration tracks changes in output frequency. |
| 1 | FINE calibration register value will be fixed after a fine lock. |

**Bit 2 – MODE** Operating Mode Selection

| Value | Description |
|---|---|
| 0 | The DFLL operates in open-loop operation. |
| 1 | The DFLL operates in closed-loop operation. |

**Bit 1 – ENABLE** DFLL Enable
Due to synchronization, there is delay from updating the register until the peripheral is enabled/disabled. The value written to DFLLCTRL.ENABLE will read back immediately after written.

| Value | Description |
|---|---|
| 0 | The DFLL oscillator is disabled. |
| 1 | The DFLL oscillator is enabled. |

### 17.8.11 DFLL48M Value

**Name:** DFLLVAL
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-Synchronized, Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DIFF[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIFF[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COARSE[5:0] | | | | | | FINE[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FINE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – DIFF[15:0]** Multiplication Ratio Difference
In closed-loop mode (DFLLCTRL.MODE is written to one), this bit group indicates the difference between the ideal number of DFLL cycles and the counted number of cycles. This value is not updated in open-loop mode, and should be considered invalid in that case.

**Bits 15:10 – COARSE[5:0]** Coarse Value
Set the value of the Coarse Calibration register. In closed-loop mode, this field is read-only.

**Bits 9:0 – FINE[9:0]** Fine Value
Set the value of the Fine Calibration register. In closed-loop mode, this field is read-only.

### 17.8.12 DFLL48M Multiplier

**Name:** DFLLMUL
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CSTEP[5:0] | | | | | | FSTEP[9:8] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FSTEP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MUL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MUL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:26 – CSTEP[5:0]** Coarse Maximum Step
This bit group indicates the maximum step size allowed during coarse adjustment in closed-loop mode. When adjusting to a new frequency, the expected output frequency overshoot depends on this step size.

**Bits 25:16 – FSTEP[9:0]** Fine Maximum Step
This bit group indicates the maximum step size allowed during fine adjustment in closed-loop mode. When adjusting to a new frequency, the expected output frequency overshoot depends on this step size.

**Bits 15:0 – MUL[15:0]** DFLL Multiply Factor
This field determines the ratio of the CLK_DFLL output frequency to the CLK_DFLL_REF input frequency. Writing to the MUL bits will cause locks to be lost and the fine calibration value to be reset to its midpoint.

### 17.8.13 DFLL48M Synchronization

**Name:** DFLLSYNC
**Offset:** 0x30
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | READREQ | | | | | | | |
| Access | W | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – READREQ** Read Request
To be able to read the current value of DFLLVAL in closed-loop mode, this bit should be written to one. The updated value is available in DFLLVAL when PCLKSR.DFLLRDY is set.

### 17.8.14 3.3V Brown-Out Detector (BOD33) Control

**Name:** BOD33
**Offset:** 0x34
**Reset:** 0x00XX00XX
**Property:** Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | LEVEL[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PSEL[3:0] | | | | | | CEN | MODE |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | RUNSTDBY | | ACTION[1:0] | | HYST | ENABLE | |
| Access | | R/W | | R/W | R/W | R/W | R/W | |
| Reset | | 0 | | x | x | x | x | |

**Bits 21:16 – LEVEL[5:0]** BOD33 Threshold Level
This field sets the triggering voltage threshold for the BOD33. See the *Electrical Characteristics* for actual voltage levels. Note that any change to the LEVEL field of the BOD33 register should be done when the BOD33 is disabled in order to avoid spurious resets or interrupts.

These bits are loaded from Flash User Row at start-up. Refer to *NVM User Row Mapping* for more details.

**Bits 15:12 – PSEL[3:0]** Prescaler Select
Selects the prescaler divide-by output for the BOD33 sampling mode according to the table below. The input clock comes from the OSCULP32K 1kHz output.

| PSEL[3:0] | Name | Description |
|---|---|---|
| 0x0 | DIV2 | Divide clock by 2 |
| 0x1 | DIV4 | Divide clock by 4 |
| 0x2 | DIV8 | Divide clock by 8 |
| 0x3 | DIV16 | Divide clock by 16 |
| 0x4 | DIV32 | Divide clock by 32 |
| 0x5 | DIV64 | Divide clock by 64 |

| PSEL[3:0] | Name | Description |
|---|---|---|
| 0x6 | DIV128 | Divide clock by 128 |
| 0x7 | DIV256 | Divide clock by 256 |
| 0x8 | DIV512 | Divide clock by 512 |
| 0x9 | DIV1K | Divide clock by 1024 |
| 0xA | DIV2K | Divide clock by 2048 |
| 0xB | DIV4K | Divide clock by 4096 |
| 0xC | DIV8K | Divide clock by 8192 |
| 0xD | DIV16K | Divide clock by 16384 |
| 0xE | DIV32K | Divide clock by 32768 |
| 0xF | DIV64K | Divide clock by 65536 |

**Bit 9 – CEN** Clock Enable
Writing a zero to this bit will stop the BOD33 sampling clock.

Writing a one to this bit will start the BOD33 sampling clock.

| Value | Description |
|---|---|
| 0 | The BOD33 sampling clock is either disabled and stopped, or enabled but not yet stable. |
| 1 | The BOD33 sampling clock is either enabled and stable, or disabled but not yet stopped. |

**Bit 8 – MODE** Operation Mode

| Value | Description |
|---|---|
| 0 | The BOD33 operates in continuous mode. |
| 1 | The BOD33 operates in sampling mode. |

**Bit 6 – RUNSTDBY** Run in Standby

| Value | Description |
|---|---|
| 0 | The BOD33 is disabled in standby sleep mode. |
| 1 | The BOD33 is enabled in standby sleep mode. |

**Bits 4:3 – ACTION[1:0]** BOD33 Action
These bits are used to select the BOD33 action when the supply voltage crosses below the BOD33 threshold.

These bits are loaded from Flash User Row at start-up.

| ACTION[1:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | RESET | The BOD33 generates a reset |
| 0x2 | INTERRUPT | The BOD33 generates an interrupt |
| 0x3 | | Reserved |

**Bit 2 – HYST** Hysteresis
This bit indicates whether hysteresis is enabled for the BOD33 threshold voltage:

This bit is loaded from Flash User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0 | No hysteresis. |
| 1 | Hysteresis enabled. |

**Bit 1 – ENABLE** Enable
This bit is loaded from Flash User Row at startup. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0 | BOD33 is disabled. |
| 1 | BOD33 is enabled. |

**Related Links**

37. Electrical Characteristics

10.3.1 NVM User Row Mapping

### 17.8.15 Voltage Regulator System (VREG) Control

**Name:** VREG
**Offset:** 0x3C
**Reset:** 0x0X00
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FORCELDO | | | | | |
| Access | | | R/W | | | | | |
| Reset | | | 0 | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | RUNSTDBY | | | | | | |
| Access | | R/W | | | | | | |
| Reset | | 0 | | | | | | |

**Bit 13 – FORCELDO** Force LDO Voltage Regulator

| Value | Description |
|---|---|
| 0 | The voltage regulator is in low power and low drive configuration in standby sleep mode. |
| 1 | The voltage regulator is in low power and high drive configuration in standby sleep mode. |

**Bit 6 – RUNSTDBY** Run in Standby

| Value | Description |
|---|---|
| 0 | The voltage regulator is in low power configuration in standby sleep mode. |
| 1 | The voltage regulator is in normal configuration in standby sleep mode. |

### 17.8.16 Voltage References System (VREF) Control

**Name:** VREF
**Offset:** 0x40
**Reset:** 0x0XXX0000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CALIB[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | x | x | x |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CALIB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BGOUTEN | TSEN | |
| Access | | | | | | R/W | R/W | |
| Reset | | | | | | 0 | 0 | |

**Bits 26:16 – CALIB[10:0]** Bandgap Voltage Generator Calibration
These bits are used to calibrate the output level of the bandgap voltage reference. These bits are loaded from Flash Calibration Row at startup.

**Bit 2 – BGOUTEN** Bandgap Output Enable

| Value | Description |
|---|---|
| 0 | The bandgap output is not available as an ADC input channel. |
| 1 | The bandgap output is routed to an ADC input channel. |

**Bit 1 – TSEN** Temperature Sensor Enable

| Value | Description |
|---|---|
| 0 | Temperature sensor is disabled. |
| 1 | Temperature sensor is enabled and routed to an ADC input channel. |

### 17.8.17 DPLL Control A

**Name:** DPLLCTRLA
**Offset:** 0x44
**Reset:** 0x80
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ONDEMAND | RUNSTDBY | | | | | ENABLE | |
| Access | R/W | R/W | | | | | R/W | |
| Reset | 1 | 0 | | | | | 0 | |

**Bit 7 – ONDEMAND** On Demand Clock Activation

| Value | Description |
|---|---|
| 0 | The DPLL is always on when enabled. |
| 1 | The DPLL is activated only when a peripheral request the DPLL as a source clock. The DPLLCTRLA.ENABLE bit must be one to validate that operation, otherwise the peripheral request has no effect. |

**Bit 6 – RUNSTDBY** Run in Standby

| Value | Description |
|---|---|
| 0 | The DPLL is disabled in standby sleep mode. |
| 1 | The DPLL is not stopped in standby sleep mode. |

**Bit 1 – ENABLE** DPLL Enable
The software operation of enabling or disabling the DPLL takes a few clock cycles, so check the DPLLSTATUS.ENABLE status bit to identify when the DPLL is successfully activated or disabled.

| Value | Description |
|---|---|
| 0 | The DPLL is disabled. |
| 1 | The DPLL is enabled. |

### 17.8.18 DPLL Ratio Control

**Name:** DPLLRATIO
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LDRFRAC[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LDR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:16 – LDRFRAC[3:0]** Loop Divider Ratio Fractional Part
Write this field with the fractional part of the frequency multiplier.

**Bits 11:0 – LDR[11:0]** Loop Divider Ratio
Write this field with the integer part of the frequency multiplier.

### 17.8.19 DPLL Control B

**Name:** DPLLCTRLB
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | DIV[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | LBYPASS | | LTIME[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | REFCLK[1:0] | | WUF | LPEN | FILTER[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:16 – DIV[10:0]** Clock Divider
These bits are used to set the XOSC clock source division factor. Refer to 17.6.8.3 Principle of Operation.

**Bit 12 – LBYPASS** Lock Bypass

| Value | Description |
|---|---|
| 0 | Normal Mode: the CLK_FDPLL96M is turned off when lock signal is low. |
| 1 | Lock Bypass Mode: the CLK_FDPLL96M is always running, lock is irrelevant. |

**Bits 10:8 – LTIME[2:0]** Lock Time
These bits select Lock Timeout.

| LTIME[2:0] | Name | Description |
|---|---|---|
| 0x0 | DEFAULT | No time-out |
| 0x1-0x3 | | Reserved |
| 0x4 | 8MS | Time-out if no lock within 8 ms |
| 0x5 | 9MS | Time-out if no lock within 9 ms |

| LTIME[2:0] | Name | Description |
|---|---|---|
| 0x6 | 10MS | Time-out if no lock within 10 ms |
| 0x7 | 11MS | Time-out if no lock within 11 ms |

**Bits 5:4 – REFCLK[1:0]** Reference Clock Selection
These bits select the CLK_FDPLL96M_REF source.

| REFCLK[1:0] | Name | Description |
|---|---|---|
| 0x0 | XOSC32 | XOSC32 clock reference |
| 0x1 | XOSC | XOSC clock reference |
| 0x2 | GCLK_DPLL | GCLK_DPLL clock reference |
| 0x3 | | Reserved |

**Bit 3 – WUF** Wake Up Fast

| Value | Description |
|---|---|
| 0 | DPLL CK output is gated until complete startup time and lock time. |
| 1 | DPLL CK output is gated until startup time only. |

**Bit 2 – LPEN** Low-Power Enable

| Value | Description |
|---|---|
| 0 | The time to digital converter is selected. |
| 1 | The time to digital converter is not selected, this will improve power consumption but increase the output jitter. |

**Bits 1:0 – FILTER[1:0]** Proportional Integral Filter Selection
These bits select the DPLL filter type.

| FILTER[1:0] | Name | Description |
|---|---|---|
| 0x0 | DEFAULT | Default filter mode |
| 0x1 | LBFILT | Low bandwidth filter |
| 0x2 | HBFILT | High bandwidth filter |
| 0x3 | HDFILT | High damping filter |

### 17.8.20 DPLL Status

**Name:** DPLLSTATUS
**Offset:** 0x50
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DIV | ENABLE | CLKRDY | LOCK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – DIV** Divider Enable

| Value | Description |
|---|---|
| 0 | The reference clock divider is disabled. |
| 1 | The reference clock divider is enabled. |

**Bit 2 – ENABLE** DPLL Enable

| Value | Description |
|---|---|
| 0 | The DPLL is disabled. |
| 1 | The DPLL is enabled. |

**Bit 1 – CLKRDY** Output Clock Ready

| Value | Description |
|---|---|
| 0 | The DPLL output clock is off |
| 1 | The DPLL output clock in on. |

**Bit 0 – LOCK** DPLL Lock Status

| Value | Description |
|---|---|
| 0 | The DPLL Lock signal is cleared. |
| 1 | The DPLL Lock signal is asserted. |

# 18. WDT – Watchdog Timer

## 18.1 Overview

The Watchdog Timer (WDT) is a system function for monitoring correct program operation. It makes it possible to recover from error situations such as runaway or deadlocked code. The WDT is configured to a predefined time-out period, and is constantly running when enabled. If the WDT is not cleared within the time-out period, it will issue a system reset. An early-warning interrupt is available to indicate an upcoming watchdog time-out condition.

The window mode makes it possible to define a time slot (or window) inside the total time-out period during which the WDT must be cleared. If the WDT is cleared outside this window, either too early or too late, a system reset will be issued. Compared to the normal mode, this can also catch situations where a code error causes the WDT to be cleared frequently.

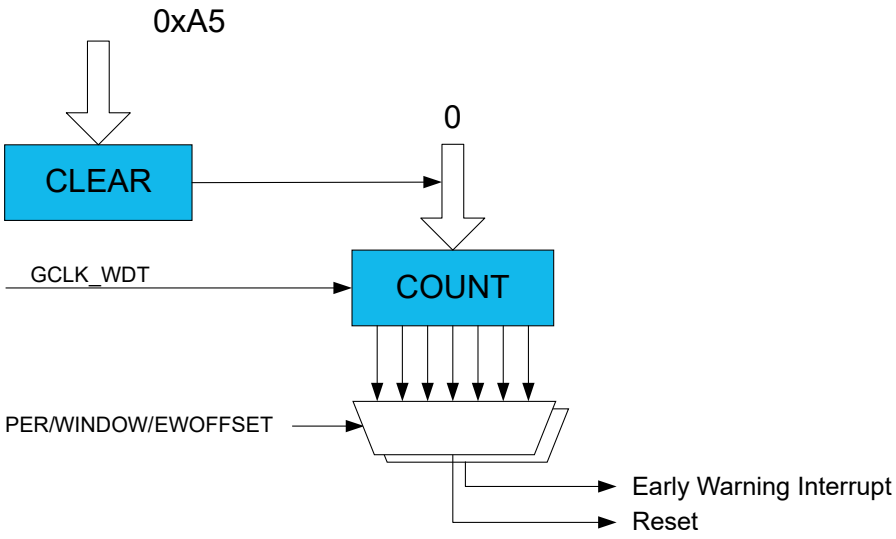
When enabled, the WDT will run in active mode and all sleep modes. It is asynchronous and runs from a CPU-independent clock source. The WDT will continue operation and issue a system reset or interrupt even if the main clocks fail.

## 18.2 Features

- Issues a system reset if the Watchdog Timer is not cleared before its time-out period
- Early Warning interrupt generation
- Asynchronous operation from dedicated oscillator
- Two types of operation:
  - Normal mode
  - Window mode
- Selectable time-out periods
  - From 8 cycles to 16,000 cycles in normal mode
  - From 16 cycles to 32,000 cycles in window mode
- Always-on capability

## 18.3 Block Diagram

**Figure 18-1. WDT Block Diagram**




## 18.4 Signal Description

Not applicable.

## 18.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 18.5.1 I/O Lines

Not applicable.

### 18.5.2 Power Management

The WDT can continue to operate in any sleep mode where the selected source clock is running. The WDT interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 18.5.3 Clocks

The WDT bus clock (CLK_WDT_APB) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to *PM – Power Manager* for details.

A generic clock (GCLK_WDT) is required to clock the WDT. This clock must be configured and enabled in the Generic Clock Controller before using the WDT. Refer to *GCLK – Generic Clock Controller* for details. This generic clock is asynchronous to the user interface clock (CLK_WDT_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to *Synchronization* for further details.

GCLK_WDT is intended to be sourced from the clock of the internal ultra-low-power (ULP) oscillator. Due to the ultralow- power design, the oscillator is not very accurate, and so the exact time-out period may

vary from device to device. This variation must be kept in mind when designing software that uses the WDT to ensure that the time-out periods used are valid for all devices. For more information on ULP oscillator accuracy, consult the *Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics*.

GCLK_WDT can also be clocked from other sources if a more accurate clock is needed, but at the cost of higher power consumption.

**Related Links**

16. PM – Power Manager
15. GCLK - Generic Clock Controller
17.6.5 32kHz Ultra Low Power Internal Oscillator (OSCULP32K) Operation
18.6.5 Synchronization

### 18.5.4 DMA

Not applicable.

### 18.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the WDT interrupt(s) requires the interrupt controller to be configured first.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 18.5.6 Events

Not applicable.

### 18.5.7 Debug Operation

When the CPU is halted in debug mode the WDT will halt normal operation.

### 18.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag Status and Clear register (INTFLAG)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 18.5.9 Analog Connections

Not applicable.

## 18.6 Functional Description

### 18.6.1 Principle of Operation

The Watchdog Timer (WDT) is a system for monitoring correct program operation, making it possible to recover from error situations such as runaway code, by issuing a Reset. When enabled, the WDT is a

constantly running timer that is configured to a predefined time-out period. Before the end of the time-out period, the WDT should be set back, or else, a system Reset is issued.

The WDT has two modes of operation, Normal mode and Window mode. Both modes offer the option of Early Warning interrupt generation. The description for each of the basic modes is given below. The settings in the Control register (CTRL) and the Interrupt Enable register (handled by INTENCLR/SET) determine the mode of operation:

**Table 18-1. WDT Operating Modes**

| CTRL.ENABLE | CTRL.WEN | INTENSET.EW | Mode |
|---|---|---|---|
| 0 | x | x | Stopped |
| 1 | 0 | 0 | Normal |
| 1 | 0 | 1 | Normal with Early Warning interrupt |
| 1 | 1 | 0 | Window |
| 1 | 1 | 1 | Window with Early Warning interrupt |

### 18.6.2 Basic Operation

#### 18.6.2.1 Initialization

The following bits are enable-protected:

- Window Mode Enable in the Control register (CTRL.WEN)
- Always-On in the Control register (CTRL-ALWAYSON)

The following registers are enable-protected:

- Configuration register (CONFIG)
- Early Warning Interrupt Control register (EWCTRL)

Any writes to these bits or registers when the WDT is enabled or is being enabled (CTRL.ENABLE=1) will be discarded. Writes to these registers while the WDT is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the Enable-Protected property in the register description.

Initialization of the WDT can be done only while the WDT is disabled. The WDT is configured by defining the required Time-Out Period bits in the Configuration register (CONFIG.PER). If window-mode operation is required, the Window Enable bit in the Control register (CTRL.WEN) must be written to one and the Window Period bits in the Configuration register (CONFIG.WINDOW) must be defined.

**Normal Mode**

- Defining the required Time-Out Period bits in the Configuration register (CONFIG.PER).

**Normal Mode with Early Warning interrupt**

- Defining the required Time-Out Period bits in the Configuration register (CONFIG.PER).
- Defining Early Warning Interrupt Time Offset bits in the Early Warning Interrupt Control register (EWCTRL. EWOFFSET).
- Setting Early Warning Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.EW).

**Window Mode**

- Defining Time-Out Period bits in the Configuration register (CONFIG.PER).
- Defining Window Mode Time-Out Period bits in the Configuration register (CONFIG.WINDOW).
- Setting Window Enable bit in the Control register (CTRL.WEN).

**Window Mode with Early Warning interrupt**

- Defining Time-Out Period bits in the Configuration register (CONFIG.PER).
- Defining Window Mode Time-Out Period bits in the Configuration register (CONFIG.WINDOW).
- Setting Window Enable bit in the Control register (CTRL.WEN).
- Defining Early Warning Interrupt Time Offset bits in the Early Warning Interrupt Control register (EWCTRL. EWOFFSET).
- Setting Early Warning Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.EW).

#### 18.6.2.2 Configurable Reset Values

After a Power-on Reset, some registers will be loaded with initial values from the NVM User Row. Refer to *NVM User Row Mapping* for more details.

This encompasses the following bits and bit groups:

- Enable bit in the Control register, CTRL.ENABLE
- Always-On bit in the Control register, CTRL.ALWAYSON
- Watchdog Timer Windows Mode Enable bit in the Control register, CTRL.WEN
- Watchdog Timer Windows Mode Time-Out Period bits in the Configuration register, CONFIG.WINDOW
- Time-Out Period in the Configuration register, CONFIG.PER
- Early Warning Interrupt Time Offset bits in the Early Warning Interrupt Control register, EWCTRL.EWOFFSET

For more information about fuse locations, see *NVM User Row Mapping*.

**Related Links**

10.3.1 NVM User Row Mapping

#### 18.6.2.3 Enabling and Disabling

The WDT is enabled by writing a '1' to the Enable bit in the Control register (CTRL.ENABLE). The WDT is disabled by writing a '0' to CTRL.ENABLE.

The WDT can be disabled only if the Always-On bit in the Control register (CTRL.ALWAYSON) is '0'.

#### 18.6.2.4 Normal Mode

In Normal mode operation, the length of a time-out period is configured in CONFIG.PER. The WDT is enabled by writing a '1' to the Enable bit in the Control register (CTRL.ENABLE). Once enabled, the WDT will issue a system reset if a time-out occurs. This can be prevented by clearing the WDT at any time during the time-out period.

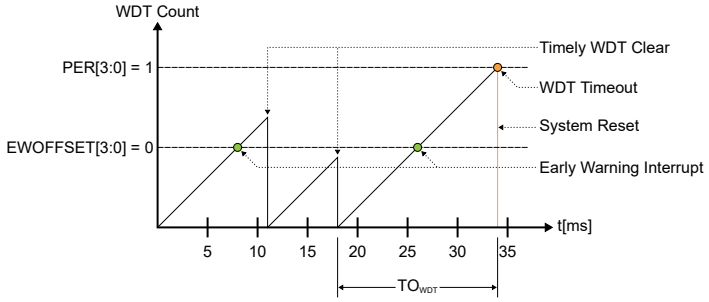
The WDT is cleared and a new WDT time-out period is started by writing 0xA5 to the Clear register (CLEAR). Writing any other value than 0xA5 to CLEAR will issue an immediate system reset.

There are 12 possible WDT time-out ($TO_{WDT}$) periods, selectable from 8ms to 16s.

By default, the early warning interrupt is disabled. If it is desired, the Early Warning Interrupt Enable bit in the Interrupt Enable register (INTENSET.EW) must be written to '1'. The Early Warning Interrupt is disabled again by writing a '1' to the Early Warning Interrupt bit in the Interrupt Enable Clear register

(INTENCLR.EW). If the Early Warning Interrupt is enabled, an interrupt is generated prior to a WDT time-out condition. In Normal mode, the Early Warning Offset bits in the Early Warning Interrupt Control register, EWCTRL.EWOFFSET, define the time when the early warning interrupt occurs. The Normal mode operation is illustrated in the figure Normal-Mode Operation.

**Figure 18-2. Normal-Mode Operation**

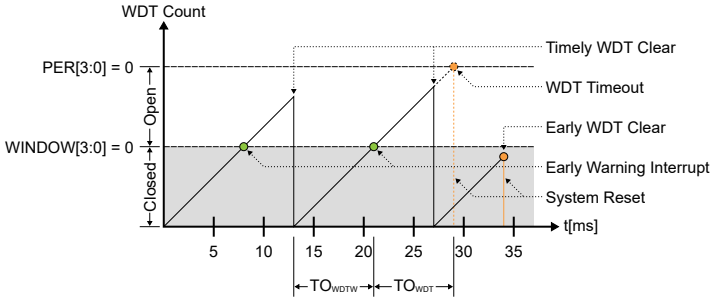



#### 18.6.2.5 Window Mode

In Window mode operation, the WDT uses two different time specifications: the WDT can only be cleared by writing 0xA5 to the CLEAR register *after* the closed window time-out period ($TO_{WDTW}$), during the subsequent Normal time-out period ($TO_{WDTW}$). If the WDT is cleared before the time window opens (before $TO_{WDTW}$ is over), the WDT will issue a system reset. Both parameters $TO_{WDTW}$ and $TO_{WDT}$ are periods in a range from 8ms to 16s, so the total duration of the WDT time-out period is the sum of the two parameters. The closed window period is defined by the Window Period bits in the Configuration register (CONFIG.WINDOW), and the open window period is defined by the Period bits in the Configuration register (CONFIG.PER).

By default, the Early Warning interrupt is disabled. If it is desired, the Early Warning Interrupt Enable bit in the Interrupt Enable register (INTENSET.EW) must be written to '1'. The Early Warning Interrupt is disabled again by writing a '1' to the Early Warning Interrupt bit in the Interrupt Enable Clear (INTENCLR.EW) register. If the Early Warning interrupt is enabled in Window mode, the interrupt is generated at the start of the open window period, i.e. after $TO_{WDTW}$. The Window mode operation is illustrated in figure Window-Mode Operation.

**Figure 18-3. Window-Mode Operation**




### 18.6.3 Additional Features

#### 18.6.3.1 Always-On Mode

The Always-On mode is enabled by setting the Always-On bit in the Control register (CTRLA.ALWAYSON=1). When the Always-On mode is enabled, the WDT runs continuously, regardless of the state of CTRL.ENABLE. Once written, the Always-On bit can only be cleared by a power-on reset.

The Configuration (CONFIG) and Early Warning Control (EWCTRL) registers are read-only registers while the CTRL.ALWAYSON bit is set. Thus, the time period configuration bits (CONFIG.PER, CONFIG.WINDOW, EWCTRL.EWOFFSET) of the WDT cannot be changed.

Enabling or disabling Window mode operation by writing the Window Enable bit (CTRLA.WEN) is allowed while in Always-On mode, but note that CONFIG.PER cannot be changed.

The CTRL.ALWAYSON bit must never be set to one by software if any of the following conditions is true:

1. The GCLK_WDT is disabled
2. The clock generator for the GCLK_WDT is disabled
3. The source clock of the clock generator for the GCLK_WDT is disabled or off

The Interrupt Clear and Interrupt Set registers are accessible in the Always-On mode. The Early Warning interrupt can still be enabled or disabled while in the Always-On mode, but note that EWCTRL.EWOFFSET cannot be changed.

**Table 18-2. WDT Operating Modes With Always-On**

| WEN | Interrupt enable | Mode |
|---|---|---|
| 0 | 0 | Always-on and normal mode |
| 0 | 1 | Always-on and normal mode with Early Warning interrupt |
| 1 | 0 | Always-on and window mode |
| 1 | 1 | Always-on and window mode with Early Warning interrupt |

### 18.6.4 Interrupts

The WDT has the following interrupt source:

- Early Warning (EW)

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register.

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the WDT is reset. See the INTFLAG register description for details on how to clear interrupt flags.

The WDT has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

**Note:** Interrupts must be globally enabled for interrupt requests to be generated.

The Early Warning interrupt behaves differently in normal mode and in window mode. In normal mode, the Early Warning interrupt generation is defined by the Early Warning Offset in the Early Warning Control register (EWCTRL.EWOFFSET). The Early Warning Offset bits define the number of GCLK_WDT clocks before the interrupt is generated, relative to the start of the watchdog time-out period. For example, if the WDT is operating in normal mode with CONFIG.PER = 0x2 and EWCTRL.EWOFFSET = 0x1, the Early Warning interrupt is generated 16 GCLK_WDT clock cycles from the start of the watchdog time-out period, and the watchdog time-out system reset is generated 32 GCLK_WDT clock cycles from the start of the watchdog time-out period. The user must take caution when programming the Early Warning Offset bits. If these bits define an Early Warning interrupt generation time greater than the watchdog time-out

© 2018 Microchip Technology Inc.

period, the watchdog time-out system reset is generated prior to the Early Warning interrupt. Thus, the Early Warning interrupt will never be generated.

In window mode, the Early Warning interrupt is generated at the start of the open window period. In a typical application where the system is in sleep mode, it can use this interrupt to wake up and clear the Watchdog Timer, after which the system can perform other tasks or return to sleep mode.

### 18.6.5 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY='1', the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following registers are synchronized when written:

- Control register (CTRL)
- Clear register (CLEAR)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 18.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | ALWAYSON | | | | | WEN | ENABLE | |
| 0x01 | CONFIG | 7:0 | WINDOW[3:0] | | | | PER[3:0] | | | |
| 0x02 | EWCTRL | 7:0 | | | | | EWOFFSET[3:0] | | | |
| 0x03 | Reserved | | | | | | | | | |
| 0x04 | INTENCLR | 7:0 | | | | | | | | EW |
| 0x05 | INTENSET | 7:0 | | | | | | | | EW |
| 0x06 | INTFLAG | 7:0 | | | | | | | | EW |
| 0x07 | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x08 | CLEAR | 7:0 | CLEAR[7:0] | | | | | | | |

## 18.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

### 18.8.1 Control

**Name:** CTRL
**Offset:** 0x0
**Reset:** N/A - Loaded from NVM User Row at start-up
**Property:** Write-Protected, Enable-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ALWAYSON | | | | | WEN | ENABLE | |
| Access | R/W | | | | | R/W | R/W | |
| Reset | x | | | | | x | x | |

**Bit 7 – ALWAYSON** Always-On
This bit allows the WDT to run continuously. After being written to one, this bit cannot be written to zero, and the WDT will remain enabled until a power-on reset is received. When this bit is one, the Control register (CTRL), the Configuration register (CONFIG) and the Early Warning Control register (EWCTRL) will be read-only, and any writes to these registers are not allowed. Writing a zero to this bit has no effect.

This bit is not enable-protected.

These bits are loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0 | The WDT is enabled and disabled through the ENABLE bit. |
| 1 | The WDT is enabled and can only be disabled by a power-on reset (POR). |

**Bit 2 – WEN** Watchdog Timer Window Mode Enable
The initial value of this bit is loaded from Flash Calibration.

This bit is loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0 | Window mode is disabled (normal operation). |
| 1 | Window mode is enabled. |

**Bit 1 – ENABLE** Enable
This bit enables or disables the WDT. Can only be written while CTRL.ALWAYSON is zero.

Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

This bit is not enable-protected.

This bit is loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0 | The WDT is disabled. |
| 1 | The WDT is enabled. |

**Related Links**

10.3.1 NVM User Row Mapping

### 18.8.2 Configuration

**Name:** CONFIG
**Offset:** 0x1
**Reset:** N/A - Loaded from NVM User Row at startup
**Property:** Write-Protected, Enable-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WINDOW[3:0] | | | | PER[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 7:4 – WINDOW[3:0]** Window Mode Time-Out Period
In window mode, these bits determine the watchdog closed window period as a number of oscillator cycles.

These bits are loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0x0 | 8 clock cycles |
| 0x1 | 16 clock cycles |
| 0x2 | 32 clock cycles |
| 0x3 | 64 clock cycles |
| 0x4 | 128 clock cycles |
| 0x5 | 256 clocks cycles |
| 0x6 | 512 clocks cycles |
| 0x7 | 1024 clock cycles |
| 0x8 | 2048 clock cycles |
| 0x9 | 4096 clock cycles |
| 0xA | 8192 clock cycles |
| 0xB | 16384 clock cycles |
| 0xC-0xF | Reserved |

**Bits 3:0 – PER[3:0]** Time-Out Period
These bits determine the watchdog time-out period as a number of GCLK_WDT clock cycles. In window mode operation, these bits define the open window period.

These bits are loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0x0 | 8 clock cycles |
| 0x1 | 16 clock cycles |
| 0x2 | 32 clock cycles |
| 0x3 | 64 clock cycles |
| 0x4 | 128 clock cycles |
| 0x5 | 256 clocks cycles |
| 0x6 | 512 clocks cycles |
| 0x7 | 1024 clock cycles |
| 0x8 | 2048 clock cycles |

| Value | Description |
|---|---|
| 0x9 | 4096 clock cycles |
| 0xA | 8192 clock cycles |
| 0xB | 16384 clock cycles |
| 0xC-0xF | Reserved |

**Related Links**

10.3.1 NVM User Row Mapping

### 18.8.3 Early Warning Interrupt Control

**Name:** EWCTRL
**Offset:** 0x2
**Reset:** N/A - Loaded from NVM User Row at start-up
**Property:** Write-Protected, Enable-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EWOFFSET[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | x | x | x | x |

**Bits 3:0 – EWOFFSET[3:0]** Early Warning Interrupt Time Offset
These bits determine the number of GCLK_WDT clocks in the offset from the start of the watchdog time-out period to when the Early Warning interrupt is generated. These bits are loaded from NVM User Row at start-up. Refer to *NVM User Row Mapping* for more details.

| Value | Description |
|---|---|
| 0x0 | 8 clock cycles |
| 0x1 | 16 clock cycles |
| 0x2 | 32 clock cycles |
| 0x3 | 64 clock cycles |
| 0x4 | 128 clock cycles |
| 0x5 | 256 clocks cycles |
| 0x6 | 512 clocks cycles |
| 0x7 | 1024 clock cycles |
| 0x8 | 2048 clock cycles |
| 0x9 | 4096 clock cycles |
| 0xA | 8192 clock cycles |
| 0xB | 16384 clock cycles |
| 0xC-0xF | Reserved |

**Related Links**
10.3.1 NVM User Row Mapping

### 18.8.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x4
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | EW |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – EW** Early Warning Interrupt Enable
Writing a zero to this bit has no effect.
Writing a one to this bit disables the Early Warning interrupt.

| Value | Description |
|---|---|
| 0 | The Early Warning interrupt is disabled. |
| 1 | The Early Warning interrupt is enabled. |

### 18.8.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x5
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | EW |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – EW** Early Warning Interrupt Enable
Writing a zero to this bit has no effect.
Writing a one to this bit enables the Early Warning interrupt.

| Value | Description |
|---|---|
| 0 | The Early Warning interrupt is disabled. |
| 1 | The Early Warning interrupt is enabled. |

### 18.8.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x6
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | EW |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – EW** Early Warning
This flag is set when an Early Warning interrupt occurs, as defined by the EWOFFSET bit group in EWCTRL.

Writing a zero to this bit has no effect.
Writing a one to this bit clears the Early Warning interrupt flag.

### 18.8.7 Status

**Name:** STATUS
**Offset:** 0x7
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 18.8.8 Clear

**Name:** CLEAR
**Offset:** 0x8
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLEAR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – CLEAR[7:0]** Watchdog Clear
Writing 0xA5 to this register will clear the Watchdog Timer and the watchdog time-out period is restarted. Writing any other value will issue an immediate system reset.

# 19. RTC – Real-Time Counter

## 19.1 Overview

The Real-Time Counter (RTC) is a 32-bit counter with a 10-bit programmable prescaler that typically runs continuously to keep track of time. The RTC can wake up the device from sleep modes using the alarm/compare wake up, periodic wake up, or overflow wake up mechanisms

The RTC is typically clocked by the 1.024kHz output from the 32.768kHz High-Accuracy Internal Crystal Oscillator(OSC32K) and this is the configuration optimized for the lowest power consumption. The faster 32.768kHz output can be selected if the RTC needs a resolution higher than 1ms. The RTC can also be clocked from other sources, selectable through the Generic Clock module (GCLK).

The RTC can generate periodic peripheral events from outputs of the prescaler, as well as alarm/compare interrupts and peripheral events, which can trigger at any counter value. Additionally, the timer can trigger an overflow interrupt and peripheral event, and can be reset on the occurrence of an alarm/compare match. This allows periodic interrupts and peripheral events at very long and accurate intervals.

The 10-bit programmable prescaler can scale down the clock source. By this, a wide range of resolutions and time-out periods can be configured. With a 32.768kHz clock source, the minimum counter tick interval is 30.5µs, and time-out periods can range up to 36 hours. For a counter tick interval of 1s, the maximum time-out period is more than 136 years.

## 19.2 Features

- 32-bit counter with 10-bit prescaler
- Multiple clock sources
- 32-bit or 16-bit Counter mode
  - One 32-bit or two 16-bit compare values
- Clock/Calendar mode
  - Time in seconds, minutes and hours (12/24)
  - Date in day of month, month and year
  - Leap year correction
- Digital prescaler correction/tuning for increased accuracy
- Overflow, alarm/compare match and prescaler interrupts and events
  - Optional clear on alarm/compare match

## 19.3 Block Diagram

**Figure 19-1. RTC Block Diagram (Mode 0 — 32-Bit Counter)**

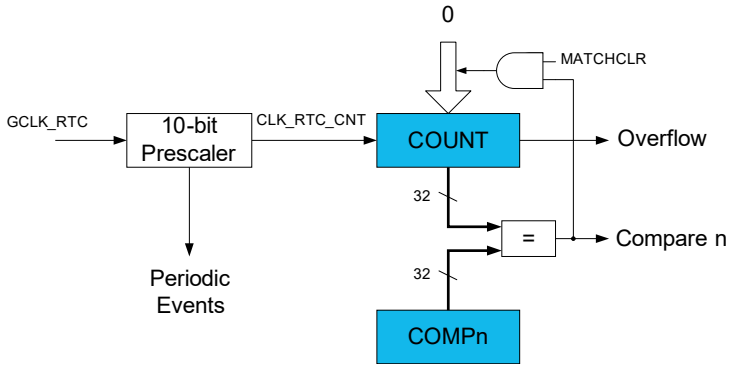



**Figure 19-2. RTC Block Diagram (Mode 1 — 16-Bit Counter)**

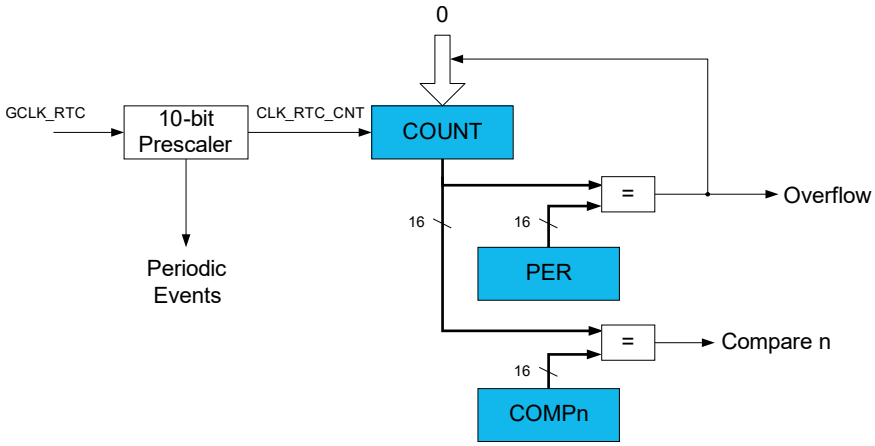
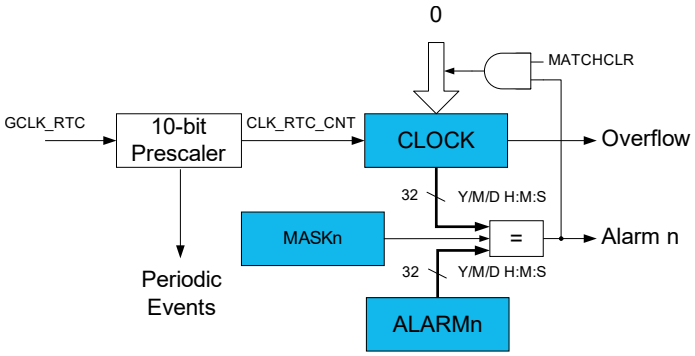



**Figure 19-3. RTC Block Diagram (Mode 2 — Clock/Calendar)**




## 19.4 Signal Description

Not applicable.

## 19.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 19.5.1 I/O Lines

Not applicable.

### 19.5.2 Power Management

The RTC will continue to operate in any sleep mode where the selected source clock is running. The RTC interrupts can be used to wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes. Refer to the *Power Manager* for details on the different sleep modes.

The RTC will be reset only at power-on (POR) or by setting the Software Reset bit in the Control register (CTRL.SWRST=1).

**Related Links**

16. PM – Power Manager

### 19.5.3 Clocks

The RTC bus clock (CLK_RTC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_RTC_APB can be found in the Peripheral Clock Masking section.

A generic clock (GCLK_RTC) is required to clock the RTC. This clock must be configured and enabled in the Generic Clock Controller before using the RTC. Refer to *GCLK – Generic Clock Controller* for details.

This generic clock is asynchronous to the user interface clock (CLK_RTC_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to 19.6.8 Synchronization for further details.

The RTC should not work with the Generic Clock Generator 0.

**Related Links**

16.6.2.6 Peripheral Clock Masking

15. GCLK - Generic Clock Controller

### 19.5.4 DMA

Not applicable.

### 19.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the RTC interrupts requires the Interrupt Controller to be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 19.5.6 Events

The events are connected to the *Event System*.

**Related Links**

24. EVSYS – Event System

### 19.5.7 Debug Operation

When the CPU is halted in debug mode the RTC will halt normal operation. The RTC can be forced to continue operation during debugging. Refer to the Debug Control (DBGCTRL) register for details.

### 19.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag Status and Clear register (INTFLAG)
- Read Request register (READREQ)
- Status register (STATUS)
- Debug register (DBGCTRL)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 19.5.9 Analog Connections

A 32.768kHz crystal can be connected to the XIN32 and XOUT32 pins, along with any required load capacitors. For details on recommended crystal characteristics and load capacitors, refer to *Electrical Characteristics* for details.

**Related Links**

37. Electrical Characteristics

## 19.6 Functional Description

### 19.6.1 Principle of Operation

The RTC keeps track of time in the system and enables periodic events, as well as interrupts and events at a specified time. The RTC consists of a 10-bit prescaler that feeds a 32-bit counter. The actual format of the 32-bit counter depends on the RTC operating mode.

The RTC can function in one of these modes:

- Mode 0 - COUNT32: RTC serves as 32-bit counter
- Mode 1 - COUNT16: RTC serves as 16-bit counter
- Mode 2 - CLOCK: RTC serves as clock/calendar with alarm functionality

### 19.6.2 Basic Operation

#### 19.6.2.1 Initialization

The following bits are enable-protected, meaning that they can only be written when the RTC is disabled (CTRL.ENABLE=0):

- Operating Mode bits in the Control register (CTRL.MODE)
- Prescaler bits in the Control register (CTRL.PRESCALER)
- Clear on Match bit in the Control register (CTRL.MATCHCLR)
- Clock Representation bit in the Control register (CTRL.CLKREP)

The following register is enable-protected:

- Event Control register (EVCTRL)

Any writes to these bits or registers when the RTC is enabled or being enabled (CTRL.ENABLE=1) will be discarded. Writes to these bits or registers while the RTC is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the "Enable-Protected" property in the register description.

Before the RTC is enabled, it must be configured, as outlined by the following steps:

1. RTC operation mode must be selected by writing the Operating Mode bit group in the Control register (CTRL.MODE)
2. Clock representation must be selected by writing the Clock Representation bit in the Control register (CTRL.CLKREP)
3. Prescaler value must be selected by writing the Prescaler bit group in the Control register (CTRL.PRESCALER)

The RTC prescaler divides the source clock for the RTC counter.
**Note:** In Clock/Calendar mode, the prescaler must be configured to provide a 1Hz clock to the counter for correct operation.

The frequency of the RTC clock (CLK_RTC_CNT) is given by the following formula:

$$f_{\mathrm{CLK_RTC_CNT}} = \frac{f_{\mathrm{GCLK_RTC}}}{2^{\mathrm{PRESCALER}}}$$

The frequency of the generic clock, GCLK_RTC, is given by $f_{GCLK_RTC}$, and $f_{CLK_RTC_CNT}$ is the frequency of the internal prescaled RTC clock, CLK_RTC_CNT.

#### 19.6.2.2 Enabling, Disabling and Resetting

The RTC is enabled by setting the Enable bit in the Control register (CTRL.ENABLE=1). The RTC is disabled by writing CTRL.ENABLE=0.

The RTC is reset by setting the Software Reset bit in the Control register (CTRL.SWRST=1). All registers in the RTC, except DEBUG, will be reset to their initial state, and the RTC will be disabled. The RTC must be disabled before resetting it.

### 19.6.3 Operating Modes

The RTC counter supports three RTC operating modes: 32-bit Counter, 16-bit Counter and Clock/Calendar. The operating mode is selected by writing to the Operating Mode bit group in the Control register (CTRL.MODE).

#### 19.6.3.1 32-Bit Counter (Mode 0)

When the RTC Operating Mode bits in the Control register are zero (CTRL.MODE=00), the counter operates in 32-bit Counter mode. The block diagram of this mode is shown in Figure 19-1. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The counter will increment until it reaches the top value of 0xFFFFFFFF, and then wrap to 0x00000000. This sets the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The RTC counter value can be read from or written to the Counter Value register (COUNT) in 32-bit format.

The counter value is continuously compared with the 32-bit Compare register (COMP0). When a compare match occurs, the Compare 0interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMP0) is set on the next 0-to-1 transition of CLK_RTC_CNT.

If the Clear on Match bit in the Control register (CTRL.MATCHCLR) is '1', the counter is cleared on the next counter cycle when a compare match with COMP0 occurs. This allows the RTC to generate periodic interrupts or events with longer periods than are possible with the prescaler events. Note that when CTRL.MATCHCLR is '1', INTFLAG.CMP0 and INTFLAG.OVF will both be set simultaneously on a compare match with COMP0.

#### 19.6.3.2 16-Bit Counter (Mode 1)

When the RTC Operating Mode bits in the Control register (CTRL.MODE) are 1, the counter operates in 16-bit Counter mode as shown in Figure 19-2. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. In 16-bit Counter mode, the 16-bit Period register (PER) holds the maximum value of the counter. The counter will increment until it reaches the PER value, and then wrap to 0x0000. This sets the Overflow interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The RTC counter value can be read from or written to the Counter Value register (COUNT) in 16-bit format.

The counter value is continuously compared with the 16-bit Compare registers (COMPn, n=0–1). When a compare match occurs, the Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn, n=0–1) is set on the next 0-to-1 transition of CLK_RTC_CNT.

#### 19.6.3.3 Clock/Calendar (Mode 2)

When CTRL.MODE is two, the counter operates in Clock/Calendar mode, as shown in Figure 19-3. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The selected clock source and RTC prescaler must be configured to provide a 1Hz clock to the counter for correct operation in this mode.

The time and date can be read from or written to the Clock Value register (CLOCK) in a 32-bit time/date format. Time is represented as:

- Seconds
- Minutes
- Hours

Hours can be represented in either 12- or 24-hour format, selected by the Clock Representation bit in the Control register (CTRL.CLKREP). This bit can be changed only while the RTC is disabled.

Date is represented as:

- Day as the numeric day of the month (starting at 1)
- Month as the numeric month of the year (1 = January, 2 = February, etc.)
- Year as a value counting the offset from a reference value that must be defined in software

The date is automatically adjusted for leap years, assuming every year divisible by 4 is a leap year. Therefore, the reference value must be a leap year, e.g. 2000. The RTC will increment until it reaches the top value of 23:59:59 December 31st of year 63, and then wrap to 00:00:00 January 1st of year 0. This will set the Overflow interrupt flag in the Interrupt Flag Status and Clear registers (INTFLAG.OVF).

The clock value is continuously compared with the 32-bit Alarm register (ALARM0). When an alarm match occurs, the Alarm 0 Interrupt flag in the Interrupt Flag Status and Clear registers (INTFLAG.ALARMn0) is set on the next 0-to-1 transition of CLK_RTC_CNT. E.g. For a 1Hz clock counter, it means the Alarm 0 Interrupt flag is set with a delay of 1s after the occurrence of alarm match. A valid alarm match depends on the setting of the Alarm Mask Selection bits in the Alarm

A valid alarm match depends on the setting of the Alarm Mask Selection bits in the Alarm 0 Mask register (MASK0.SEL). These bits determine which time/date fields of the clock and alarm values are valid for comparison and which are ignored.

If the Clear on Match bit in the Control register (CTRL.MATCHCLR) is one, the counter is cleared on the next counter cycle when an alarm match with ALARM0 occurs. This allows the RTC to generate periodic interrupts or events with longer periods than are possible with the prescaler events (see 19.6.9.1

Periodic Events). Note that when CTRL.MATCHCLR is '1', INTFLAG.ALARM0 and INTFLAG.OVF will both be set simultaneously on an alarm match with ALARM0.

### 19.6.4 DMA Operation

Not applicable.

### 19.6.5 Interrupts

The RTC has the following interrupt sources which are asynchronous interrupts and can wake-up the device from any sleep mode.:

- Overflow (INTFLAG.OVF): Indicates that the counter has reached its top value and wrapped to zero.
- Compare n (INTFLAG.CMPn): Indicates a match between the counter value and the compare register.
- Alarm n (INTFLAG.ALARMn): Indicates a match between the clock value and the alarm register.
- Synchronization Ready (INTFLAG.SYNCRDY): Indicates an operation requires synchronization.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by setting the corresponding bit in the Interrupt Enable Set register (INTENSET=1), and disabled by setting the corresponding bit in the Interrupt Enable Clear register (INTENCLR=1). An interrupt request is generated when the interrupt flag is raised and the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled or the RTC is reset. See the description of the INTFLAG registers for details on how to clear interrupt flags. All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. Refer to the Nested Vector Interrupt Controller for details. The user must read the INTFLAG register to determine which interrupt condition is present.

**Note:** Interrupts must be globally enabled for interrupt requests to be generated. Refer to the Nested Vector Interrupt Controller for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 19.6.6 Events

The RTC can generate the following output events, which are generated in the same way as the corresponding interrupts:

- Overflow (OVF): Indicates that the counter has reached its top value and wrapped to zero.
- Period n (PERn): The corresponding bit in the prescaler has toggled. Refer to 19.6.9.1 Periodic Events for details.
- Compare n (CMPn): Indicates a match between the counter value and the compare register.
- Alarm n (ALARMn): Indicates a match between the clock value and the alarm register.

Setting the Event Output bit in the Event Control Register (EVCTRL.xxxEO=1) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. Refer to the *EVSYS - Event System* for details on configuring the event system.

**Related Links**

24. EVSYS – Event System

### 19.6.7 Sleep Mode Operation

The RTC will continue to operate in any sleep mode where the source clock is active. The RTC *interrupts* can be used to wake up the device from a sleep mode. RTC *events* can trigger other operations in the system without exiting the sleep mode.

An interrupt request will be generated after the wake-up if the Interrupt Controller is configured accordingly. Otherwise the CPU will wake up directly, without triggering any interrupt. In this case, the CPU will continue executing right from the first instruction that followed the entry into sleep.

The periodic events can also wake up the CPU through the interrupt function of the Event System. In this case, the event must be enabled and connected to an event channel with its interrupt enabled. See *Event System* for more information.

**Related Links**

24. EVSYS – Event System

### 19.6.8 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The Synchronization Ready interrupt can be used to signal when synchronization is complete. This can be accessed via the Synchronization Ready Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.SYNCRDY). If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits are synchronized when written:

- Software Reset bit in the Control register (CTRL.SWRST)
- Enable bit in the Control register (CTRL.ENABLE)

The following registers are synchronized when written:

- Counter Value register (COUNT)
- Clock Value register (CLOCK)
- Counter Period register (PER)
- Compare n Value registers (COMPn)
- Alarm n Value registers (ALARMn)
- Frequency Correction register (FREQCORR)
- Alarm n Mask register (MASKn)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

The following registers are synchronized when read:

- The Counter Value register (COUNT)
- The Clock Value register (CLOCK)

Required read-synchronization is denoted by the "Read-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

### 19.6.9 Additional Features

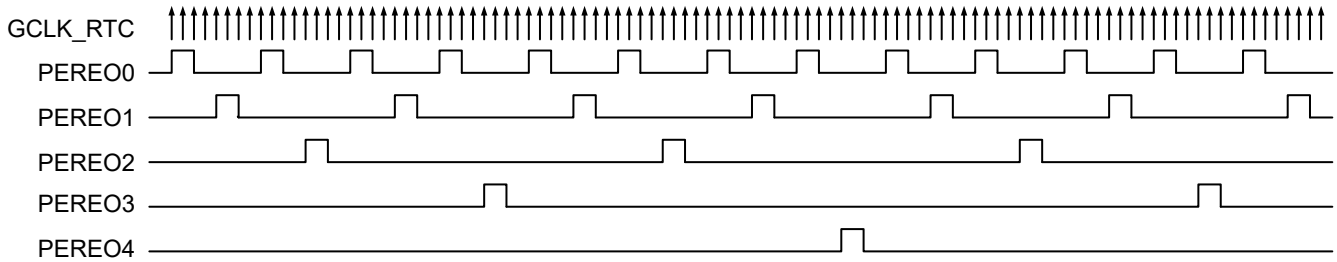
#### 19.6.9.1 Periodic Events

The RTC prescaler can generate events at periodic intervals, allowing flexible system tick creation. Any of the upper eight bits of the prescaler (bits 2 to 9) can be the source of an event. When one of the eight Periodic Event Output bits in the Event Control register (EVCTRL.PEREO[n=0..7]) is '1', an event is generated on the 0-to-1 transition of the related bit in the prescaler, resulting in a periodic event frequency of:

$$f_{PERIODIC} = \frac{f_{\mathrm{GCLK_RTC}}}{2^{n+3}}$$

$f_{GCLK_RTC}$ is the frequency of the internal prescaler clock, GCLK_RTC, and n is the position of the EVCTRL.PEREOn bit. For example, PER0 will generate an event every eight CLK_RTC_OSC cycles, PER1 every 16 cycles, etc. This is shown in the figure below. Periodic events are independent of the prescaler setting used by the RTC counter, except if CTRL.PRESCALER is zero. Then, no periodic events will be generated.

**Figure 19-4. Example Periodic Events**




#### 19.6.9.2 Frequency Correction

The RTC Frequency Correction module employs periodic counter corrections to compensate for a too-slow or too-fast oscillator. Frequency correction requires that CTRL.PRESCALER is greater than 1.

The digital correction circuit adds or subtracts cycles from the RTC prescaler to adjust the frequency in approximately 1 ppm steps. Digital correction is achieved by adding or skipping a single count in the prescaler once every 4096 GCLK_RTC_OSC cycles. The Value bit group in the Frequency Correction register (FREQCORR.VALUE) determines the number of times the adjustment is applied over 240 of these periods. The resulting correction is as follows:

*Correction in ppm = (FREQCORR.VALUE / 4096 * 240) * $10^{6ppm}$*

This results in a resolution of 1.017 PPM.

The Sign bit in the Frequency Correction register (FREQCORR.SIGN) determines the direction of the correction. A positive value will add counts and increase the period (reducing the frequency), and a negative value will reduce counts per period (speeding up the frequency). Digital correction also affects the generation of the periodic events from the prescaler. When the correction is applied at the end of the correction cycle period, the interval between the previous periodic event and the next occurrence may also be shortened or lengthened depending on the correction value.

© 2018 Microchip Technology Inc.

## 19.7 Register Summary

The register mapping depends on the Operating Mode bits in the Control register (CTRL.MODE). The register summary is presented for each of the three modes.

**Table 19-1. MODE0 - Mode Register Summary**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | MATCHCLR | | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | | | PRESCALER[3:0] | | | |
| 0x02 | READREQ | 7:0 | | | ADDR[5:0] | | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | EVCTRL | 7:0 | PEREO7 | PEREO6 | PEREO5 | PEREO4 | PEREO3 | PEREO2 | PEREO1 | PEREO0 |
| 0x05 | | 15:8 | OVFEO | | | | | | | CMPEO0 |
| 0x06 | INTENCLR | 7:0 | OVF | SYNCRDY | | | | | | CMP0 |
| 0x07 | INTENSET | 7:0 | OVF | SYNCRDY | | | | | | CMP0 |
| 0x08 | INTFLAG | 7:0 | OVF | SYNCRDY | | | | | | CMP0 |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x0B | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x0C | FREQCORR | 7:0 | SIGN | VALUE[6:0] | | | | | | |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| 0x11 | | 15:8 | COUNT[15:8] | | | | | | | |
| 0x12 | | 23:16 | COUNT[23:16] | | | | | | | |
| 0x13 | | 31:24 | COUNT[31:24] | | | | | | | |
| 0x14 ... 0x17 | Reserved | | | | | | | | | |
| 0x18 | COMP0 | 7:0 | COMP[7:0] | | | | | | | |
| 0x19 | | 15:8 | COMP[15:8] | | | | | | | |
| 0x1A | | 23:16 | COMP[23:16] | | | | | | | |
| 0x1B | | 31:24 | COMP[31:24] | | | | | | | |

**Table 19-2. MODE1 - Mode Register Summary**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | | | PRESCALER[3:0] | | | |
| 0x02 | READREQ | 7:0 | | | ADDR[5:0] | | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | EVCTRL | 7:0 | PEREO7 | PEREO6 | PEREO5 | PEREO4 | PEREO3 | PEREO2 | PEREO1 | PEREO0 |
| 0x05 | | 15:8 | OVFEO | | | | | | CMPEO1 | CMPEO0 |
| 0x06 | INTENCLR | 7:0 | OVF | SYNCRDY | | | | | CMP1 | CMP0 |
| 0x07 | INTENSET | 7:0 | OVF | SYNCRDY | | | | | CMP1 | CMP0 |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x08 | INTFLAG | 7:0 | OVF | SYNCRDY | | | | | CMP1 | CMP0 |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x0B | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x0C | FREQCORR | 7:0 | SIGN | VALUE[6:0] | | | | | | |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| 0x11 | | 15:8 | COUNT[15:8] | | | | | | | |
| 0x12 | Reserved | | | | | | | | | |
| 0x13 | Reserved | | | | | | | | | |
| 0x14 | PER | 7:0 | PER[7:0] | | | | | | | |
| 0x15 | | 15:8 | PER[15:8] | | | | | | | |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | COMP0 | 7:0 | COMP[7:0] | | | | | | | |
| 0x19 | | 15:8 | COMP[15:8] | | | | | | | |
| 0x1A | COMP1 | 7:0 | COMP[7:0] | | | | | | | |
| 0x1B | | 15:8 | COMP[15:8] | | | | | | | |

**Table 19-3. MODE2 - Mode Register Summary**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | MATCHCLR | CLKREP | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | | | PRESCALER[3:0] | | | |
| 0x02 | READREQ | 7:0 | | | ADDR[5:0] | | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | EVCTRL | 7:0 | PEREO7 | PEREO6 | PEREO5 | PEREO4 | PEREO3 | PEREO2 | PEREO1 | PEREO0 |
| 0x05 | | 15:8 | OVFEO | | | | | | | ALARMEO0 |
| 0x06 | INTENCLR | 7:0 | OVF | SYNCRDY | | | | | | ALARM0 |
| 0x07 | INTENSET | 7:0 | OVF | SYNCRDY | | | | | | ALARM0 |
| 0x08 | INTFLAG | 7:0 | OVF | SYNCRDY | | | | | | ALARM0 |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x0B | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x0C | FREQCORR | 7:0 | SIGN | VALUE[6:0] | | | | | | |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | CLOCK | 7:0 | MINUTE[1:0] | | SECOND[5:0] | | | | | |
| 0x11 | | 15:8 | HOUR[3:0] | | | | MINUTE[5:2] | | | |
| 0x12 | | 23:16 | MONTH[1:0] | | DAY[4:0] | | | | | HOUR[4] |
| 0x13 | | 31:24 | YEAR[5:0] | | | | | | MONTH[3:2] | |
| 0x14 | Reserved | | | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ...<br>0x17 | | | | | | | | | | |
| 0x18 | ALARM0 | 7:0 | MINUTE[1:0] | | SECOND[5:0] | | | | | |
| 0x19 | | 15:8 | HOUR[3:0] | | | | MINUTE[5:2] | | | |
| 0x1A | | 23:16 | MONTH[1:0] | | DAY[4:0] | | | | | HOUR[4] |
| 0x1B | | 31:24 | YEAR[5:0] | | | | | | MONTH[3:2] | |
| 0x1C | MASK | 7:0 | | | | | | SEL[2:0] | | |

## 19.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 19.8.1 Control - MODE0

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESCALER[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MATCHCLR | | | | MODE[1:0] | | ENABLE | SWRST |
| Access | R/W | | | | R/W | R/W | R/W | W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – PRESCALER[3:0]** Prescaler
These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).

These bits are not synchronized.

| PRESCALER[3:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | CLK_RTC_CNT = GCLK_RTC/1 |
| 0x1 | DIV2 | CLK_RTC_CNT = GCLK_RTC/2 |
| 0x2 | DIV4 | CLK_RTC_CNT = GCLK_RTC/4 |
| 0x3 | DIV8 | CLK_RTC_CNT = GCLK_RTC/8 |
| 0x4 | DIV16 | CLK_RTC_CNT = GCLK_RTC/16 |
| 0x5 | DIV32 | CLK_RTC_CNT = GCLK_RTC/32 |
| 0x6 | DIV64 | CLK_RTC_CNT = GCLK_RTC/64 |
| 0x7 | DIV128 | CLK_RTC_CNT = GCLK_RTC/128 |
| 0x8 | DIV256 | CLK_RTC_CNT = GCLK_RTC/256 |
| 0x9 | DIV512 | CLK_RTC_CNT = GCLK_RTC/512 |
| 0xA | DIV1024 | CLK_RTC_CNT = GCLK_RTC/1024 |
| 0xB-0xF | | Reserved |

**Bit 7 – MATCHCLR** Clear on Match
This bit is valid only in Mode 0 and Mode 2.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The counter is not cleared on a Compare/Alarm 0 match. |
| 1 | The counter is cleared on a Compare/Alarm 0 match. |

**Bits 3:2 – MODE[1:0]** Operating Mode
These bits define the operating mode of the RTC.

These bits are not synchronized.

| MODE[1:0] | Name | Description |
|---|---|---|
| 0x0 | COUNT32 | Mode 0: 32-bit Counter |
| 0x1 | COUNT16 | Mode 1: 16-bit Counter |
| 0x2 | CLOCK | Mode 2: Clock/Calendar |
| 0x3 | | Reserved |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 19.8.2 Control - MODE1

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESCALER[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MODE[1:0] | | ENABLE | SWRST |
| Access | | | | | R/W | R/W | R/W | W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – PRESCALER[3:0]** Prescaler
These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).

These bits are not synchronized.

| PRESCALER[3:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | CLK_RTC_CNT = GCLK_RTC/1 |
| 0x1 | DIV2 | CLK_RTC_CNT = GCLK_RTC/2 |
| 0x2 | DIV4 | CLK_RTC_CNT = GCLK_RTC/4 |
| 0x3 | DIV8 | CLK_RTC_CNT = GCLK_RTC/8 |
| 0x4 | DIV16 | CLK_RTC_CNT = GCLK_RTC/16 |
| 0x5 | DIV32 | CLK_RTC_CNT = GCLK_RTC/32 |
| 0x6 | DIV64 | CLK_RTC_CNT = GCLK_RTC/64 |
| 0x7 | DIV128 | CLK_RTC_CNT = GCLK_RTC/128 |
| 0x8 | DIV256 | CLK_RTC_CNT = GCLK_RTC/256 |
| 0x9 | DIV512 | CLK_RTC_CNT = GCLK_RTC/512 |
| 0xA | DIV1024 | CLK_RTC_CNT = GCLK_RTC/1024 |
| 0xB-0xF | | Reserved |

**Bits 3:2 – MODE[1:0]** Operating Mode
These bits define the operating mode of the RTC.

These bits are not synchronized.

| MODE[1:0] | Name | Description |
|---|---|---|
| 0x0 | COUNT32 | Mode 0: 32-bit Counter |
| 0x1 | COUNT16 | Mode 1: 16-bit Counter |
| 0x2 | CLOCK | Mode 2: Clock/Calendar |
| 0x3 | | Reserved |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 19.8.3 Control - MODE2

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESCALER[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MATCHCLR | CLKREP | | | MODE[1:0] | | ENABLE | SWRST |
| Access | R/W | R/W | | | R/W | R/W | R/W | W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – PRESCALER[3:0]** Prescaler
These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).

These bits are not synchronized.

| PRESCALER[3:0] | Name | Description |
|---|---|---|
| 0x0 | DIV1 | CLK_RTC_CNT = GCLK_RTC/1 |
| 0x1 | DIV2 | CLK_RTC_CNT = GCLK_RTC/2 |
| 0x2 | DIV4 | CLK_RTC_CNT = GCLK_RTC/4 |
| 0x3 | DIV8 | CLK_RTC_CNT = GCLK_RTC/8 |
| 0x4 | DIV16 | CLK_RTC_CNT = GCLK_RTC/16 |
| 0x5 | DIV32 | CLK_RTC_CNT = GCLK_RTC/32 |
| 0x6 | DIV64 | CLK_RTC_CNT = GCLK_RTC/64 |
| 0x7 | DIV128 | CLK_RTC_CNT = GCLK_RTC/128 |
| 0x8 | DIV256 | CLK_RTC_CNT = GCLK_RTC/256 |
| 0x9 | DIV512 | CLK_RTC_CNT = GCLK_RTC/512 |
| 0xA | DIV1024 | CLK_RTC_CNT = GCLK_RTC/1024 |
| 0xB-0xF | | Reserved |

**Bit 7 – MATCHCLR** Clear on Match
This bit is valid only in Mode 0 and Mode 2. This bit can be written only when the peripheral is disabled.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The counter is not cleared on a Compare/Alarm 0 match. |
| 1 | The counter is cleared on a Compare/Alarm 0 match. |

**Bit 6 – CLKREP** Clock Representation
This bit is valid only in Mode 2 and determines how the hours are represented in the Clock Value (CLOCK) register. This bit can be written only when the peripheral is disabled.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | 24 Hour |
| 1 | 12 Hour (AM/PM) |

**Bits 3:2 – MODE[1:0]** Operating Mode
These bits define the operating mode of the RTC.

These bits are not synchronized.

| MODE[1:0] | Name | Description |
|---|---|---|
| 0x0 | COUNT32 | Mode 0: 32-bit Counter |
| 0x1 | COUNT16 | Mode 1: 16-bit Counter |
| 0x2 | CLOCK | Mode 2: Clock/Calendar |
| 0x3 | | Reserved |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 19.8.4 Read Request

**Name:** READREQ
**Offset:** 0x02
**Reset:** 0x0010
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RREQ | RCONT | | | | | | |
| Access | W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ADDR[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 1 | 0 | 0 | 0 | 0 |

**Bit 15 – RREQ** Read Request
Writing a zero to this bit has no effect.

Writing a one to this bit requests synchronization of the register pointed to by the Address bit group (READREQ.ADDR) and sets the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY).

**Bit 14 – RCONT** Read Continuously
Writing a zero to this bit disables continuous synchronization.

Writing a one to this bit enables continuous synchronization of the register pointed to by READREQ.ADDR. The register value will be synchronized automatically every time the register is updated. READREQ.RCONT prevents READREQ.RREQ from clearing automatically.

This bit is cleared when an RTC register is written.

**Bits 5:0 – ADDR[5:0]** Address
These bits select the offset of the register that needs read synchronization. In the RTC only COUNT and CLOCK, which share the same address, are available for read synchronization. Therefore, ADDR is a read-only constant of 0x10.

### 19.8.5 Event Control - MODE0

**Name:** EVCTRL
**Offset:** 0x04
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OVFEO | | | | | | | CMPEO0 |
| Access | R/W | | | | | | | R/W |
| Reset | 0 | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – OVFEO** Overflow Event Output Enable

| Value | Description |
|---|---|
| 0 | Overflow event is disabled and will not be generated. |
| 1 | Overflow event is enabled and will be generated for every overflow. |

**Bit 8 – CMPEO0** Compare 0 Event Output Enable

| Value | Description |
|---|---|
| 0 | Compare 0 event is disabled and will not be generated. |
| 1 | Compare 0 event is enabled and will be generated for every compare match. |

**Bits 7,6,5,4,3,2,1,0 – PEREOx** Periodic Interval x Event Output Enable [x=7:0]

| Value | Description |
|---|---|
| 0 | Periodic Interval x event is disabled and will not be generated. |
| 1 | Periodic Interval x event is enabled and will be generated. |

### 19.8.6 Event Control - MODE1

**Name:** EVCTRL
**Offset:** 0x04
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OVFEO | | | | | | CMPEOx | CMPEOx |
| Access | R/W | | | | | | R/W | R/W |
| Reset | 0 | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – OVFEO** Overflow Event Output Enable

| Value | Description |
|---|---|
| 0 | Overflow event is disabled and will not be generated. |
| 1 | Overflow event is enabled and will be generated for every overflow. |

**Bits 9,8 – CMPEOx** Compare x Event Output Enable [x=1:0]

| Value | Description |
|---|---|
| 0 | Compare x event is disabled and will not be generated. |
| 1 | Compare x event is enabled and will be generated for every compare match. |

**Bits 7,6,5,4,3,2,1,0 – PEREOx** Periodic Interval x Event Output Enable [x=7:0]

| Value | Description |
|---|---|
| 0 | Periodic Interval x event is disabled and will not be generated. |
| 1 | Periodic Interval x event is enabled and will be generated. |

### 19.8.7 Event Control - MODE2

**Name:** EVCTRL
**Offset:** 0x04
**Reset:** 0x0000
**Property:** Enable-Protected, Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OVFEO | | | | | | | ALARMEO0 |
| Access | R/W | | | | | | | R/W |
| Reset | 0 | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx | PEREOx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – OVFEO** Overflow Event Output Enable

| Value | Description |
|---|---|
| 0 | Overflow event is disabled and will not be generated. |
| 1 | Overflow event is enabled and will be generated for every overflow. |

**Bit 8 – ALARMEO0** Alarm 0 Event Output Enable

| Value | Description |
|---|---|
| 0 | Alarm 0 event is disabled and will not be generated. |
| 1 | Alarm 0 event is enabled and will be generated for every alarm. |

**Bits 7,6,5,4,3,2,1,0 – PEREOx** Periodic Interval x Event Output Enable [x=7:0]

| Value | Description |
|---|---|
| 0 | Periodic Interval x event is disabled and will not be generated. |
| 1 | Periodic Interval x event is enabled and will be generated. |

### 19.8.8 Interrupt Enable Clear - MODE0

**Name:** INTENCLR
**Offset:** 0x06
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | CMP0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set. |

**Bit 0 – CMP0** Compare 0 Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Compare 0 Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Compare 0 interrupt is disabled. |
| 1 | The Compare 0 interrupt is enabled, and an interrupt request will be generated when the Compare x interrupt flag is set. |

### 19.8.9 Interrupt Enable Clear - MODE1

**Name:** INTENCLR
**Offset:** 0x06
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | CMPx | CMPx |
| Access | R/W | R/W | | | | | R/W | R/W |
| Reset | 0 | 0 | | | | | 0 | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set. |

**Bits 1,0 – CMPx** Compare x Interrupt Enable [x=1:0]
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Compare x Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Compare x interrupt is disabled. |
| 1 | The Compare x interrupt is enabled, and an interrupt request will be generated when the Compare x interrupt flag is set. |

### 19.8.10 Interrupt Enable Clear - MODE2

**Name:** INTENCLR
**Offset:** 0x06
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | ALARM0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

| Value | Description |
|---|---|
| 0 | The synchronization ready interrupt is disabled. |
| 1 | The synchronization ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set. |

**Bit 0 – ALARM0** Alarm 0 Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit disables the Alarm 0 interrupt.

| Value | Description |
|---|---|
| 0 | The Alarm 0 interrupt is disabled. |
| 1 | The Alarm 0 interrupt is enabled, and an interrupt request will be generated when the Alarm 0 interrupt flag is set. |

### 19.8.11 Interrupt Enable Set - MODE0

**Name:** INTENSET
**Offset:** 0x07
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | CMP0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Overflow Interrupt Enable bit and enable the Overflow interrupt.

| Value | Description |
|---|---|
| 0 | The overflow interrupt is disabled. |
| 1 | The overflow interrupt is enabled. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit and enable the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The synchronization ready interrupt is disabled. |
| 1 | The synchronization ready interrupt is enabled. |

**Bit 0 – CMP0** Compare 0 Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Compare 0 Interrupt Enable bit and enable the Compare 0 interrupt.

| Value | Description |
|---|---|
| 0 | The compare 0 interrupt is disabled. |
| 1 | The compare 0 interrupt is enabled. |

### 19.8.12 Interrupt Enable Set - MODE1

**Name:** INTENSET
**Offset:** 0x07
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | CMPx | CMPx |
| Access | R/W | R/W | | | | | R/W | R/W |
| Reset | 0 | 0 | | | | | 0 | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Overflow interrupt bit and enable the Overflow interrupt.

| Value | Description |
|---|---|
| 0 | The overflow interrupt is disabled. |
| 1 | The overflow interrupt is enabled. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit and enable the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The synchronization ready interrupt is disabled. |
| 1 | The synchronization ready interrupt is enabled. |

**Bits 1,0 – CMPx** Compare x Interrupt Enable [x=1:0]
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Compare x Interrupt Enable bit and enable the Compare x interrupt.

| Value | Description |
|---|---|
| 0 | The compare x interrupt is disabled. |
| 1 | The compare x interrupt is enabled. |

### 19.8.13 Interrupt Enable Set - MODE2

**Name:** INTENSET
**Offset:** 0x07
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | ALARM0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Overflow Interrupt Enable bit and enable the Overflow interrupt.

| Value | Description |
|---|---|
| 0 | The overflow interrupt is disabled. |
| 1 | The overflow interrupt is enabled. |

**Bit 6 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Synchronization Ready Interrupt bit and enable the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The synchronization ready interrupt is disabled. |
| 1 | The synchronization ready interrupt is enabled. |

**Bit 0 – ALARM0** Alarm 0 Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Alarm 0 Interrupt Enable bit and enable the Alarm 0 interrupt.

| Value | Description |
|---|---|
| 0 | The alarm 0 interrupt is disabled. |
| 1 | The alarm 0 interrupt is enabled. |

### 19.8.14 Interrupt Flag Status and Clear - MODE0

**Name:** INTFLAG
**Offset:** 0x08
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | CMP0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated if INTENCLR/SET.OVF is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Overflow interrupt flag.

**Bit 6 – SYNCRDY** Synchronization Ready
This flag is cleared by writing a one to the flag.

This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by enable or software reset, and an interrupt request will be generated if INTENCLR/SET.SYNCRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Synchronization Ready interrupt flag.

**Bit 0 – CMP0** Compare 0
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition, and an interrupt request will be generated if INTENCLR/SET.CMP0 is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Compare 0 interrupt flag.

### 19.8.15 Interrupt Flag Status and Clear - MODE1

**Name:** INTFLAG
**Offset:** 0x08
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | CMPx | CMPx |
| Access | R/W | R/W | | | | | R/W | R/W |
| Reset | 0 | 0 | | | | | 0 | 0 |

**Bit 7 – OVF** Overflow
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated if INTENCLR/SET.OVF is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Overflow interrupt flag.

**Bit 6 – SYNCRDY** Synchronization Ready
This flag is cleared by writing a one to the flag.

This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by enable or software reset, and an interrupt request will be generated if INTENCLR/SET.SYNCRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Synchronization Ready interrupt flag.

**Bits 1,0 – CMPx** Compare x [x=1:0]
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition and an interrupt request will be generated if INTENCLR/SET.CMPx is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Compare x interrupt flag.

### 19.8.16 Interrupt Flag Status and Clear - MODE2

**Name:** INTFLAG
**Offset:** 0x08
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | SYNCRDY | | | | | | ALARM0 |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 0 |

**Bit 7 – OVF** Overflow
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated if INTENCLR/SET.OVF is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Overflow interrupt flag.

**Bit 6 – SYNCRDY** Synchronization Ready
This flag is cleared by writing a one to the flag.

This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by enable or software reset, and an interrupt request will be generated if INTENCLR/SET.SYNCRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Synchronization Ready interrupt flag.

**Bit 0 – ALARM0** Alarm 0
This flag is cleared by writing a one to the flag.

This flag is set on the next CLK_RTC_CNT cycle after a match with ALARM0 condition occurs, and an interrupt request will be generated if INTENCLR/SET.ALARM0 is also one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Alarm 0 interrupt flag.

### 19.8.17 Status

**Name:** STATUS
**Offset:** 0x0A
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 19.8.18 Debug Control

**Name:** DBGCTRL
**Offset:** 0x0B
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGRUN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGRUN** Run During Debug
This bit is not reset by a software reset.

Writing a zero to this bit causes the RTC to halt during debug mode.

Writing a one to this bit allows the RTC to continue normal operation during debug mode.

### 19.8.19 Frequency Correction

**Name:** FREQCORR
**Offset:** 0x0C
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SIGN | VALUE[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SIGN** Correction Sign

| Value | Description |
|---|---|
| 0 | The correction value is positive, i.e., frequency will be decreased. |
| 1 | The correction value is negative, i.e., frequency will be increased. |

**Bits 6:0 – VALUE[6:0]** Correction Value
These bits define the amount of correction applied to the RTC prescaler.

1–127: The RTC frequency is adjusted according to the value.

| Value | Description |
|---|---|
| 0 | Correction is disabled and the RTC frequency is unchanged. |

### 19.8.20 Counter Value - MODE0

**Name:** COUNT
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-Synchronized, Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – COUNT[31:0]** Counter Value
These bits define the value of the 32-bit RTC counter.

### 19.8.21 Counter Value - MODE1

**Name:** COUNT
**Offset:** 0x10
**Reset:** 0x0000
**Property:** Read-Synchronized, Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COUNT[15:0]** Counter Value
These bits define the value of the 16-bit RTC counter.

### 19.8.22 Clock Value - MODE2

**Name:** CLOCK
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-Synchronized, Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | YEAR[5:0] | | | | | | MONTH[3:2] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MONTH[1:0] | | DAY[4:0] | | | | | HOUR[4:4] |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HOUR[3:0] | | | | MINUTE[5:2] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MINUTE[1:0] | | SECOND[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:26 – YEAR[5:0]** Year
The year offset with respect to the reference year (defined in software).

The year is considered a leap year if YEAR[1:0] is zero.

**Bits 25:22 – MONTH[3:0]** Month
1 – January

2 – February

...

12 – December

**Bits 21:17 – DAY[4:0]** Day
Day starts at 1 and ends at 28, 29, 30 or 31, depending on the month and year.

**Bits 16:12 – HOUR[4:0]** Hour
When CTRL.CLKREP is zero, the Hour bit group is in 24-hour format, with values 0-23. When CTRL.CLKREP is one, HOUR[3:0] has values 1-12 and HOUR[4] represents AM (0) or PM (1).

**Table 19-4. Hour**

| HOUR[4:0] | CLOCK.HOUR[4] | CLOCK.HOUR[3:0] | Description |
|---|---|---|---|
| 0 | 0x00 - 0x17 | | Hour (0 - 23) |
| | 0x18 - 0x1F | | Reserved |
| 1 | 0 | 0x0 | Reserved |
| | | 0x1 - 0xC | AM Hour (1 - 12) |
| | | 0xD - 0xF | Reserved |
| | 1 | 0x0 | Reserved |
| | | 0x1 - 0xC | PM Hour (1 - 12) |
| | | 0xF - 0xF | Reserved |

**Bits 11:6 – MINUTE[5:0]** Minute
0 – 59.

**Bits 5:0 – SECOND[5:0]** Second
0– 59.

### 19.8.23 Counter Period - MODE1

**Name:** PER
**Offset:** 0x14
**Reset:** 0x0000
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PER[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PER[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PER[15:0]** Counter Period
These bits define the value of the 16-bit RTC period.

### 19.8.24 Compare n Value - MODE0

**Name:** COMP
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COMP[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COMP[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COMP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – COMP[31:0]** Compare Value
The 32-bit value of COMPn is continuously compared with the 32-bit COUNT value. When a match occurs, the Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next counter cycle, and the counter value is cleared if CTRL.MATCHCLR is one.

### 19.8.25 Compare n Value - MODE1

**Name:** COMPn
**Offset:** 0x18+n*0x2 [n=0..1]
**Reset:** 0x0000
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COMP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COMP[15:0]** Compare Value
The 16-bit value of COMPn is continuously compared with the 16-bit COUNT value. When a match occurs, the Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next counter cycle.

### 19.8.26 Alarm 0 Value - MODE2

**Name:** ALARM0
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Write-Protected, Write-Synchronized

The 32-bit value of ALARM0 is continuously compared with the 32-bit CLOCK value, based on the masking set by MASKn.SEL. When a match occurs, the Alarm 0 interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.ALARMn) is set on the next counter cycle, and the counter is cleared if CTRL.MATCHCLR is one.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | YEAR[5:0] | | | | | | MONTH[3:2] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MONTH[1:0] | | DAY[4:0] | | | | | HOUR[4:4] |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HOUR[3:0] | | | | MINUTE[5:2] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MINUTE[1:0] | | SECOND[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:26 – YEAR[5:0]** Year
The alarm year. Years are only matched if MASKn.SEL is 6.

**Bits 25:22 – MONTH[3:0]** Month
The alarm month. Months are matched only if MASKn.SEL is greater than 4.

**Bits 21:17 – DAY[4:0]** Day
The alarm day. Days are matched only if MASKn.SEL is greater than 3.

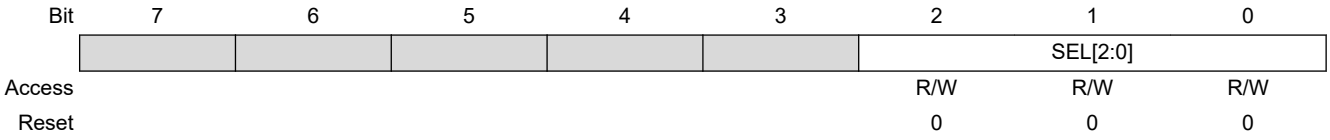
**Bits 16:12 – HOUR[4:0]** Hour
The alarm hour. Hours are matched only if MASKn.SEL is greater than 2.

**Bits 11:6 – MINUTE[5:0]** Minute
The alarm minute. Minutes are matched only if MASKn.SEL is greater than 1.

**Bits 5:0 – SECOND[5:0]** Second
The alarm second. Seconds are matched only if MASKn.SEL is greater than 0.

### 19.8.27 Alarm n Mask - MODE2

**Name:** MASK
**Offset:** 0x1C
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized




**Bits 2:0 – SEL[2:0]** Alarm Mask Selection
These bits define which bit groups of Alarm n are valid.

| SEL[2:0] | Name | Description |
|---|---|---|
| 0x0 | OFF | Alarm Disabled |
| 0x1 | SS | Match seconds only |
| 0x2 | MMSS | Match seconds and minutes only |
| 0x3 | HHMMSS | Match seconds, minutes, and hours only |
| 0x4 | DDHHMMSS | Match seconds, minutes, hours, and days only |
| 0x5 | MMDDHHMMSS | Match seconds, minutes, hours, days, and months only |
| 0x6 | YYMMDDHHMMSS | Match seconds, minutes, hours, days, months, and years |
| 0x7 | | Reserved |

# 20. DMAC – Direct Memory Access Controller

## 20.1 Overview

The Direct Memory Access Controller (DMAC) contains both a Direct Memory Access engine and a Cyclic Redundancy Check (CRC) engine. The DMAC can transfer data between memories and peripherals, and thus off-load these tasks from the CPU. It enables high data transfer rates with minimum CPU intervention, and frees up CPU time. With access to all peripherals, the DMAC can handle automatic transfer of data between communication modules.

The DMA part of the DMAC has several DMA channels which all can receive different types of transfer triggers to generate transfer requests from the DMA channels to the arbiter, see also the Block Diagram. The arbiter will grant one DMA channel at a time to act as the active channel. When an active channel has been granted, the fetch engine of the DMAC will fetch a transfer descriptor from the SRAM and store it in the internal memory of the active channel, which will execute the data transmission.

An ongoing data transfer of an active channel can be interrupted by a higher prioritized DMA channel. The DMAC will write back the updated transfer descriptor from the internal memory of the active channel to SRAM, and grant the higher prioritized channel to start transfer as the new active channel. Once a DMA channel is done with its transfer, interrupts and events can be generated optionally.

The DMAC has four bus interfaces:

- The *data transfer bus* is used for performing the actual DMA transfer.
- The *AHB/APB Bridge bus* is used when writing and reading the I/O registers of the DMAC.
- The *descriptor fetch bus* is used by the fetch engine to fetch transfer descriptors before data transfer can be started or continued.
- The *write-back bus* is used to write the transfer descriptor back to SRAM.

All buses are AHB master interfaces but the AHB/APB Bridge bus, which is an APB slave interface.

The CRC engine can be used by software to detect an accidental error in the transferred data and to take corrective action, such as requesting the data to be sent again or simply not using the incorrect data.

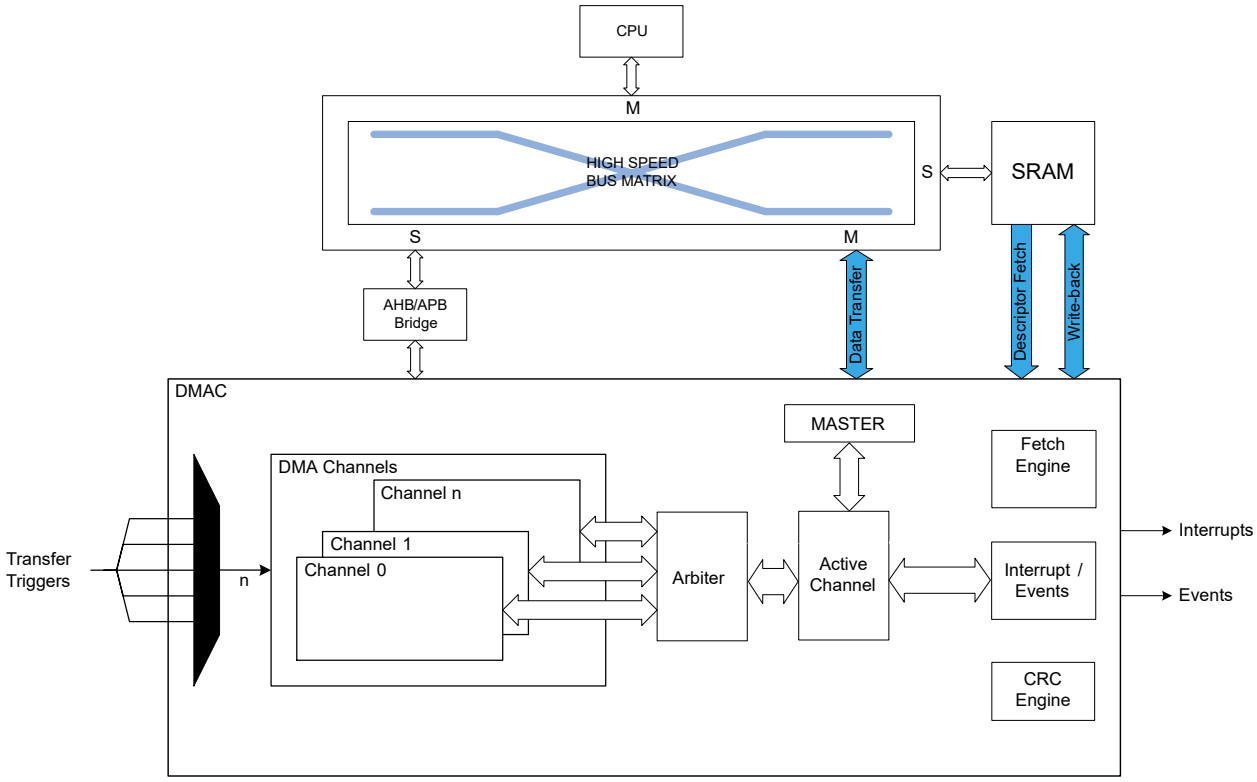
## 20.2 Features

- Data transfer from:
  - Peripheral to peripheral
  - Peripheral to memory
  - Memory to peripheral
  - Memory to memory
- Transfer trigger sources
  - Software
  - Events from Event System
  - Dedicated requests from peripherals
- SRAM based transfer descriptors
  - Single transfer using one descriptor
  - Multi-buffer or circular buffer modes by linking multiple descriptors

- Up to 12 channels
  - Enable 12 independent transfers
  - Automatic descriptor fetch for each channel
  - Suspend/resume operation support for each channel
- Flexible arbitration scheme
  - 4 configurable priority levels for each channel
  - Fixed or round-robin priority scheme within each priority level
- From 1 to 256KB data transfer in a single block transfer
- Multiple addressing modes
  - Static
  - Configurable increment scheme
- Optional interrupt generation
  - On block transfer complete
  - On error detection
  - On channel suspend
- 4 event inputs
  - One event input for each of the 4 least significant DMA channels
  - Can be selected to trigger normal transfers, periodic transfers or conditional transfers
  - Can be selected to suspend or resume channel operation
- 4 event outputs
  - One output event for each of the 4 least significant DMA channels
  - Selectable generation on AHB, block, or transaction transfer complete
- Error management supported by write-back function
  - Dedicated Write-Back memory section for each channel to store ongoing descriptor transfer
- CRC polynomial software selectable to
  - CRC-16 (CRC-CCITT)
  - CRC-32 (IEEE® 802.3)

## 20.3 Block Diagram

**Figure 20-1. DMAC Block Diagram**




## 20.4 Signal Description

Not applicable.

## 20.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 20.5.1 I/O Lines

Not applicable.

### 20.5.2 Power Management

The DMAC will continue to operate in any sleep mode where the selected source clock is running. The DMAC's interrupts can be used to wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes. On hardware or software reset, all registers are set to their reset value.

**Related Links**

16. PM – Power Manager

### 20.5.3 Clocks

The DMAC bus clock (CLK_DMAC_APB) must be configured and enabled in the Power Manager before using the DMAC.

An AHB clock (CLK_DMAC_AHB) is required to clock the DMAC. This clock must be configured and enabled in the power manager before using the DMAC, and the default state of CLK_DMAC_AHB can be found in *Peripheral Clock Masking*.

This bus clock (CLK_DMAC_APB) is always synchronous to the module clock (CLK_DMAC_AHB), but can be divided by a prescaler and may run even when the module clock is turned off.

**Related Links**

16.6.2.6 Peripheral Clock Masking

### 20.5.4 DMA

Not applicable.

### 20.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the DMAC interrupt requires the interrupt controller to be configured first.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 20.5.6 Events

The events are connected to the event system.

**Related Links**

24. EVSYS – Event System

### 20.5.7 Debug Operation

When the CPU is halted in debug mode the DMAC will halt normal operation. The DMAC can be forced to continue operation during debugging. Refer to 20.8.6 DBGCTRL for details.

### 20.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except for the following registers:

- Interrupt Pending register (INTPEND)
- Channel ID register (CHID)
- Channel Interrupt Flag Status and Clear register (CHINTFLAG)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 20.5.9 Analog Connections

Not applicable.

## 20.6 Functional Description

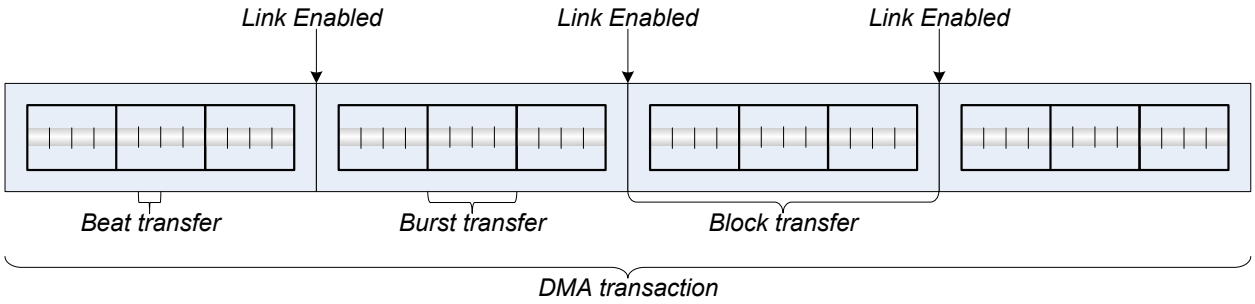
### 20.6.1 Principle of Operation

The DMAC consists of a DMA module and a CRC module.

### 20.6.1.1 DMA

The DMAC can transfer data between memories and peripherals without interaction from the CPU. The data transferred by the DMAC are called transactions, and these transactions can be split into smaller data transfers. The following figure shows the relationship between the different transfer sizes:

**Figure 20-2. DMA Transfer Sizes**




- Beat transfer: The size of one data transfer bus access, and the size is selected by writing the Beat Size bit group in the Block Transfer Control register (BTCTRL.BEATSIZE)
- Burst transfer: Defined as n beat transfers, where n will differ from one device family to another. A burst transfer is atomic, cannot be interrupted and the length of the burst is selected by writing the Burst Length bit group in each Channel n Control A register (CHCTRLA.BURSTLEN).
- Block transfer: The amount of data one transfer descriptor can transfer, and the amount can range from 1 to 64k beats. A block transfer can be interrupted, in contrast to the burst transfer.
- Transaction: The DMAC can link several transfer descriptors by having the first descriptor pointing to the second and so forth, as shown in the figure above. A DMA transaction is the complete transfer of all blocks within a linked list.

A transfer descriptor describes how a block transfer should be carried out by the DMAC, and it must remain in SRAM. For further details on the transfer descriptor refer to 20.6.2.3 Transfer Descriptors.

The figure above shows several block transfers linked together, which are called linked descriptors. For further information about linked descriptors, refer to 20.6.3.1 Linked Descriptors.

A DMA transfer is initiated by an incoming transfer trigger on one of the DMA channels. This trigger can be configured to be either a software trigger, an event trigger, or one of the dedicated peripheral triggers. The transfer trigger will result in a DMA transfer request from the specific channel to the arbiter. If there are several DMA channels with pending transfer requests, the arbiter chooses which channel is granted access to become the active channel. The DMA channel granted access as the active channel will carry out the transaction as configured in the transfer descriptor. A current transaction can be interrupted by a higher prioritized channel after each burst transfer, but will resume the block transfer when the according DMA channel is granted access as the active channel again.

For each beat transfer, an optional output event can be generated. For each block transfer, optional interrupts and an optional output event can be generated. When a transaction is completed, dependent of the configuration, the DMA channel will either be suspended or disabled.

### 20.6.1.2 CRC

The internal CRC engine supports two commonly used CRC polynomials: CRC-16 (CRC-CCITT) and CRC-32 (IEEE 802.3). It can be used on a selectable DMA channel, or on the I/O interface. Refer to 20.6.3.7 CRC Operation for details.

### 20.6.2 Basic Operation

#### 20.6.2.1 Initialization

The following DMAC registers are enable-protected, meaning that they can only be written when the DMAC is disabled (CTRL.DMAENABLE=0):

- Descriptor Base Memory Address register (BASEADDR)
- Write-Back Memory Base Address register (WRBADDR)

The following DMAC bit is enable-protected, meaning that it can only be written when both the DMAC and CRC are disabled (CTRL.DMAENABLE=0 and CTRL.CRCENABLE=0):

- Software Reset bit in Control register (CTRL.SWRST)

The following DMA channel register is enable-protected, meaning that it can only be written when the corresponding DMA channel is disabled (CHCTRLA.ENABLE=0):

- Channel Control B (CHCTRLB) register, except the Command bit (CHCTRLB.CMD) and the Channel Arbitration Level bit (CHCTRLB.LVL)

The following DMA channel bit is enable-protected, meaning that it can only be written when the corresponding DMA channel is disabled:

- Channel Software Reset bit in Channel Control A register (CHCTRLA.SWRST)

The following CRC registers are enable-protected, meaning that they can only be written when the CRC is disabled (CTRL.CRCENABLE=0):

- CRC Control register (CRCCTRL)
- CRC Checksum register (CRCCHKSUM)

Enable-protection is denoted by the "Enable-Protected" property in the register description.

Before the DMAC is enabled it must be configured, as outlined by the following steps:

- The SRAM address of where the descriptor memory section is located must be written to the Description Base Address (BASEADDR) register
- The SRAM address of where the write-back section should be located must be written to the Write-Back Memory Base Address (WRBADDR) register
- Priority level x of the arbiter can be enabled by setting the Priority Level x Enable bit in the Control register (CTRL.LVLENx=1)

Before a DMA channel is enabled, the DMA channel and the corresponding first transfer descriptor must be configured, as outlined by the following steps:

- DMA channel configurations
  - The channel number of the DMA channel to configure must be written to the Channel ID (CHID) register
  - Trigger action must be selected by writing the Trigger Action bit group in the Channel Control B register (CHCTRLB.TRIGACT)
  - Trigger source must be selected by writing the Trigger Source bit group in the Channel Control B register (CHCTRLB.TRIGSRC)
- Transfer Descriptor
  - The size of each access of the data transfer bus must be selected by writing the Beat Size bit group in the Block Transfer Control register (BTCTRL.BEATSIZE)

- The transfer descriptor must be made valid by writing a one to the Valid bit in the Block Transfer Control register (BTCTRL.VALID)
- Number of beats in the block transfer must be selected by writing the Block Transfer Count (BTCNT) register
- Source address for the block transfer must be selected by writing the Block Transfer Source Address (SRCADDR) register
- Destination address for the block transfer must be selected by writing the Block Transfer Destination Address (DSTADDR) register

If CRC calculation is needed, the CRC engine must be configured before it is enabled, as outlined by the following steps:

- The CRC input source must selected by writing the CRC Input Source bit group in the CRC Control register (CRCCTRL.CRCSRC)
- The type of CRC calculation must be selected by writing the CRC Polynomial Type bit group in the CRC Control register (CRCCTRL.CRCPOLY)
- If I/O is selected as input source, the beat size must be selected by writing the CRC Beat Size bit group in the CRC Control register (CRCCTRL.CRCBEATSIZE)

#### 20.6.2.2 Enabling, Disabling, and Resetting

The DMAC is enabled by writing the DMA Enable bit in the Control register (CTRL.DMAENABLE) to '1'. The DMAC is disabled by writing a '0' to CTRL.DMAENABLE.

A DMA channel is enabled by writing the Enable bit in the Channel Control A register (CHCTRLA.ENABLE) to '1', after writing the corresponding channel id to the Channel ID bit group in the Channel ID register (CHID.ID). A DMA channel is disabled by writing a '0' to CHCTRLA.ENABLE.

The CRC is enabled by writing a '1' to the CRC Enable bit in the Control register (CTRL.CRCENABLE). The CRC is disabled by writing a '0' to CTRL.CRCENABLE.

The DMAC is reset by writing a '1' to the Software Reset bit in the Control register (CTRL.SWRST) while the DMAC and CRC are disabled. All registers in the DMAC except DBGCTRL will be reset to their initial state.

A DMA channel is reset by writing a '1' to the Software Reset bit in the Channel Control A register (CHCTRLA.SWRST), after writing the corresponding channel id to the Channel ID bit group in the Channel ID register (CHID.ID). The channel registers will be reset to their initial state. The corresponding DMA channel must be disabled in order for the reset to take effect.

#### 20.6.2.3 Transfer Descriptors

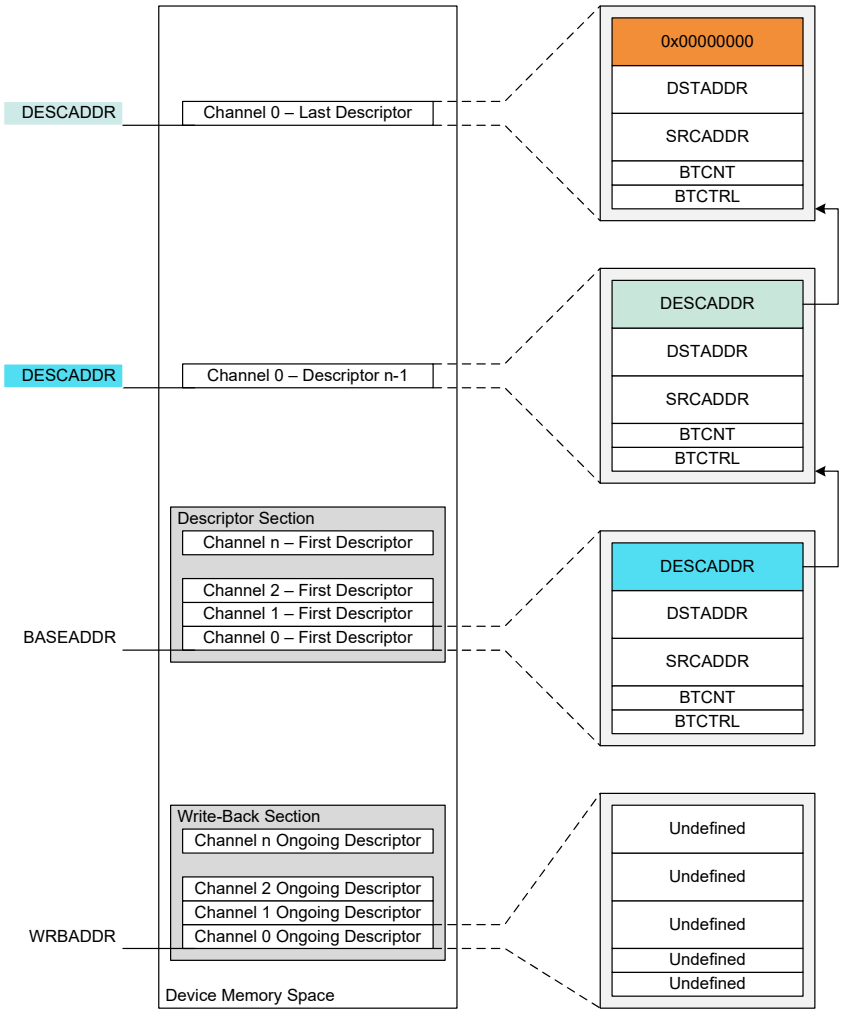
Together with the channel configurations the transfer descriptors decides how a block transfer should be executed. Before a DMA channel is enabled (CHCTRLA.ENABLE is written to one), and receives a transfer trigger, its first transfer descriptor has to be initialized and valid (BTCTRL.VALID). The first transfer descriptor describes the first block transfer of a transaction.

All transfer descriptors must reside in SRAM. The addresses stored in the Descriptor Memory Section Base Address (BASEADDR) and Write-Back Memory Section Base Address (WRBADDR) registers tell the DMAC where to find the descriptor memory section and the write-back memory section.

The descriptor memory section is where the DMAC expects to find the first transfer descriptors for all DMA channels. As BASEADDR points only to the first transfer descriptor of channel 0 (see figure below), all first transfer descriptors must be stored in a contiguous memory section, where the transfer descriptors must be ordered according to their channel number. For further details on linked descriptors, refer to 20.6.3.1 Linked Descriptors.

The write-back memory section is the section where the DMAC stores the transfer descriptors for the ongoing block transfers. WRBADDR points to the ongoing transfer descriptor of channel 0. All ongoing transfer descriptors will be stored in a contiguous memory section where the transfer descriptors are ordered according to their channel number. The figure below shows an example of linked descriptors on DMA channel 0. For further details on linked descriptors, refer to 20.6.3.1 Linked Descriptors.

**Figure 20-3. Memory Sections**




The size of the descriptor and write-back memory sections is dependent on the number of the most significant enabled DMA channel *m*, as shown below:

$$Size = 128\text{bits} \cdot (m + 1)$$

For memory optimization, it is recommended to always use the less significant DMA channels if not all channels are required.

The descriptor and write-back memory sections can either be two separate memory sections, or they can share memory section (BASEADDR=WRBADDR). The benefit of having them in two separate sections, is that the same transaction for a channel can be repeated without having to modify the first transfer descriptor. The benefit of having descriptor memory and write-back memory in the same section is that it requires less SRAM. In addition, the latency from fetching the first descriptor of a transaction to the first burst transfer is executed, is reduced.

### 20.6.2.4 Arbitration

If a DMA channel is enabled and not suspended when it receives a transfer trigger, it will send a transfer request to the arbiter. When the arbiter receives the transfer request it will include the DMA channel in the queue of channels having pending transfers, and the corresponding Pending Channel x bit in the Pending Channels registers (PENDCH.PENDCHx) will be set. Depending on the arbitration scheme, the arbiter will choose which DMA channel will be the next active channel. The active channel is the DMA channel being granted access to perform its next burst transfer. When the arbiter has granted a DMA channel access to the DMAC, the corresponding bit PENDCH.PENDCHx will be cleared. See also the following figure.

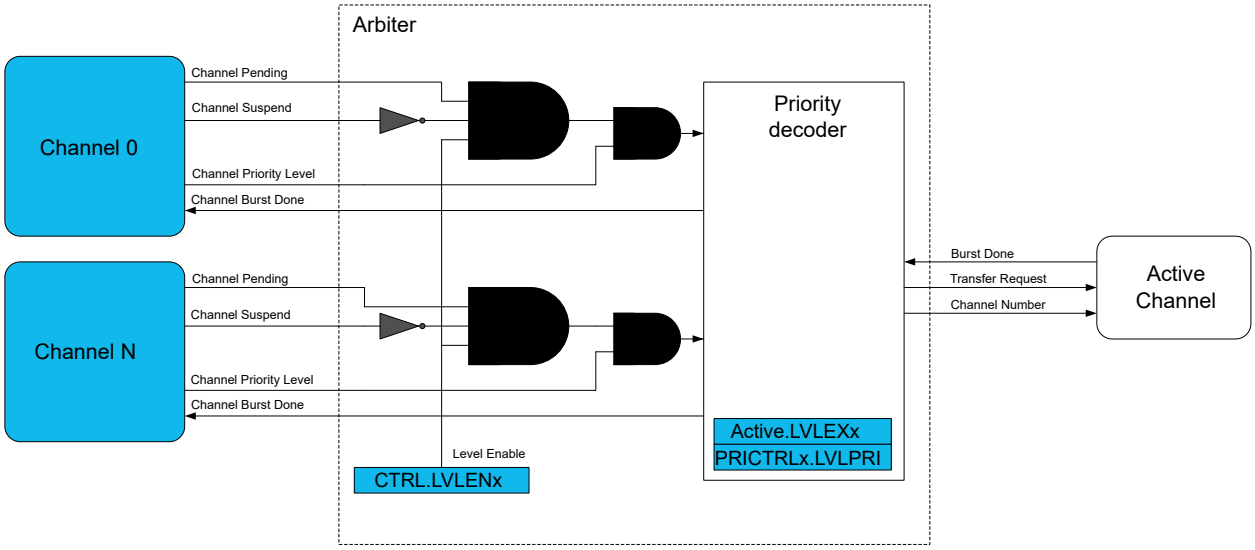
If the upcoming burst transfer is the first for the transfer request, the corresponding Busy Channel x bit in the Busy Channels register will be set (BUSYCH.BUSYCHx=1), and it will remain '1' for the subsequent granted burst transfers.

When the channel has performed its granted burst transfer(s) it will be either fed into the queue of channels with pending transfers, set to be waiting for a new transfer trigger, suspended, or disabled. This depends on the channel and block transfer configuration. If the DMA channel is fed into the queue of channels with pending transfers, the corresponding BUSYCH.BUSYCHx will remain '1'. If the DMA channel is set to wait for a new transfer trigger, suspended, or disabled, the corresponding BUSYCH.BUSYCHx will be cleared.

If a DMA channel is suspended while it has a pending transfer, it will be removed from the queue of pending channels, but the corresponding PENDCH.PENDCHx will remain set. When the same DMA channel is resumed, it will be added to the queue of pending channels again.

If a DMA channel gets disabled (CHCTRLA.ENABLE=0) while it has a pending transfer, it will be removed from the queue of pending channels, and the corresponding PENDCH.PENDCHx will be cleared.

**Figure 20-4. Arbiter Overview**




#### Priority Levels

When a channel level is pending or the channel is transferring data, the corresponding Level Executing bit is set in the Active Channel and Levels register (ACTIVE.LVLEXx).

Each DMA channel supports a 4-level priority scheme. The priority level for a channel is configured by writing to the Channel Arbitration Level bit group in the Channel Control B register (CHCTRLB.LVL). As long as all priority levels are enabled, a channel with a higher priority level number will have priority over

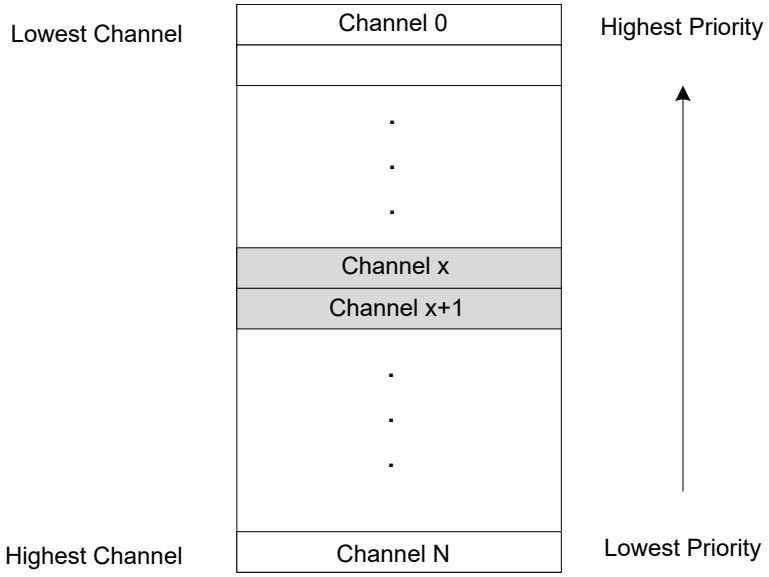
a channel with a lower priority level number. Each priority level x is enabled by setting the corresponding Priority Level x Enable bit in the Control register (CTRL.LVLENx=1).

Within each priority level the DMAC's arbiter can be configured to prioritize statically or dynamically:

*Static Arbitration* within a priority level is selected by writing a '0' to the Level x Round-Robin Scheduling Enable bit in the Priority Control 0 register (PRICTRL0.RRLVLENx).

When static arbitration is selected, the arbiter will prioritize a low channel number over a high channel number as shown in the figure below. When using the static arbitration there is a risk of high channel numbers never being granted access as the active channel. This can be avoided using a dynamic arbitration scheme.

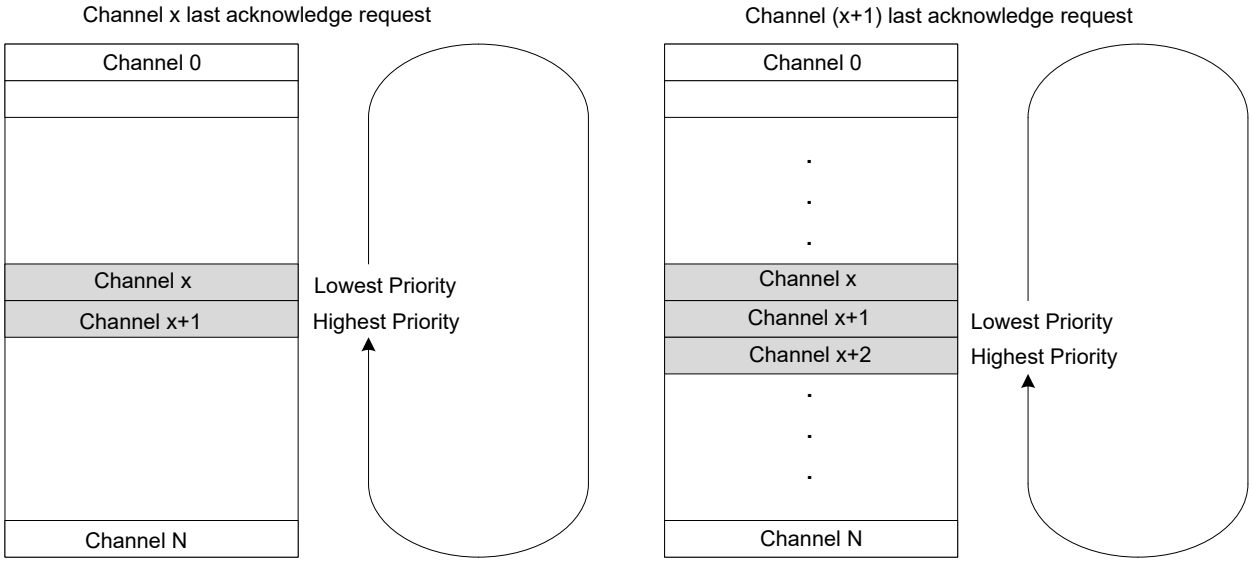
**Figure 20-5. Static Priority Scheduling**




*Dynamic Arbitration* within a priority level is selected by writing a '1' to PRICTRL0.RRLVLENx.

The dynamic arbitration scheme in the DMAC is round-robin. With the round-robin scheme, the channel number of the last channel being granted access will have the lowest priority the next time the arbiter has to grant access to a channel within the same priority level, as shown in Figure 20-6. The channel number of the last channel being granted access as the active channel is stored in the Level x Channel Priority Number bit group in the Priority Control 0 register (PRICTRL0.LVLPRIx) for the corresponding priority level.

**Figure 20-6. Dynamic (Round-Robin) Priority Scheduling**




### 20.6.2.5 Data Transmission

Before the DMAC can perform a data transmission, a DMA channel has to be configured and enabled, its corresponding transfer descriptor has to be initialized, and the arbiter has to grant the DMA channel access as the active channel.

Once the arbiter has granted a DMA channel access as the active channel (refer to DMA Block Diagram section) the transfer descriptor for the DMA channel will be fetched from SRAM using the fetch bus, and stored in the internal memory for the active channel. For a new block transfer, the transfer descriptor will be fetched from the descriptor memory section (BASEADDR); For an ongoing block transfer, the descriptor will be fetched from the write-back memory section (WRBADDR). By using the data transfer bus, the DMAC will read the data from the current source address and write it to the current destination address. For further details on how the current source and destination addresses are calculated, refer to the section on Addressing.

The arbitration procedure is performed after each burst transfer. If the current DMA channel is granted access again, the block transfer counter (BTCNT) of the internal transfer descriptor will be decremented by the number of beats in a burst transfer, the optional output event Beat will be generated if configured and enabled, and the active channel will perform a new burst transfer. If a different DMA channel than the current active channel is granted access, the block transfer counter value will be written to the write-back section before the transfer descriptor of the newly granted DMA channel is fetched into the internal memory of the active channel.

When a block transfer has come to its end (BTCNT is zero), the Valid bit in the Block Transfer Control register will be cleared (BTCTRL.VALID=0) before the entire transfer descriptor is written to the write-back memory. The optional interrupts, Channel Transfer Complete and Channel Suspend, and the optional output event Block, will be generated if configured and enabled. After the last block transfer in a transaction, the Next Descriptor Address register (DESCADDR) will hold the value 0x00000000, and the DMA channel will either be suspended or disabled, depending on the configuration in the Block Action bit group in the Block Transfer Control register (BTCTRL.BLOCKACT). If the transaction has further block transfers pending, DESCADDR will hold the SRAM address to the next transfer descriptor to be fetched. The DMAC will fetch the next descriptor into the internal memory of the active channel and write its content to the write-back section for the channel, before the arbiter gets to choose the next active channel.

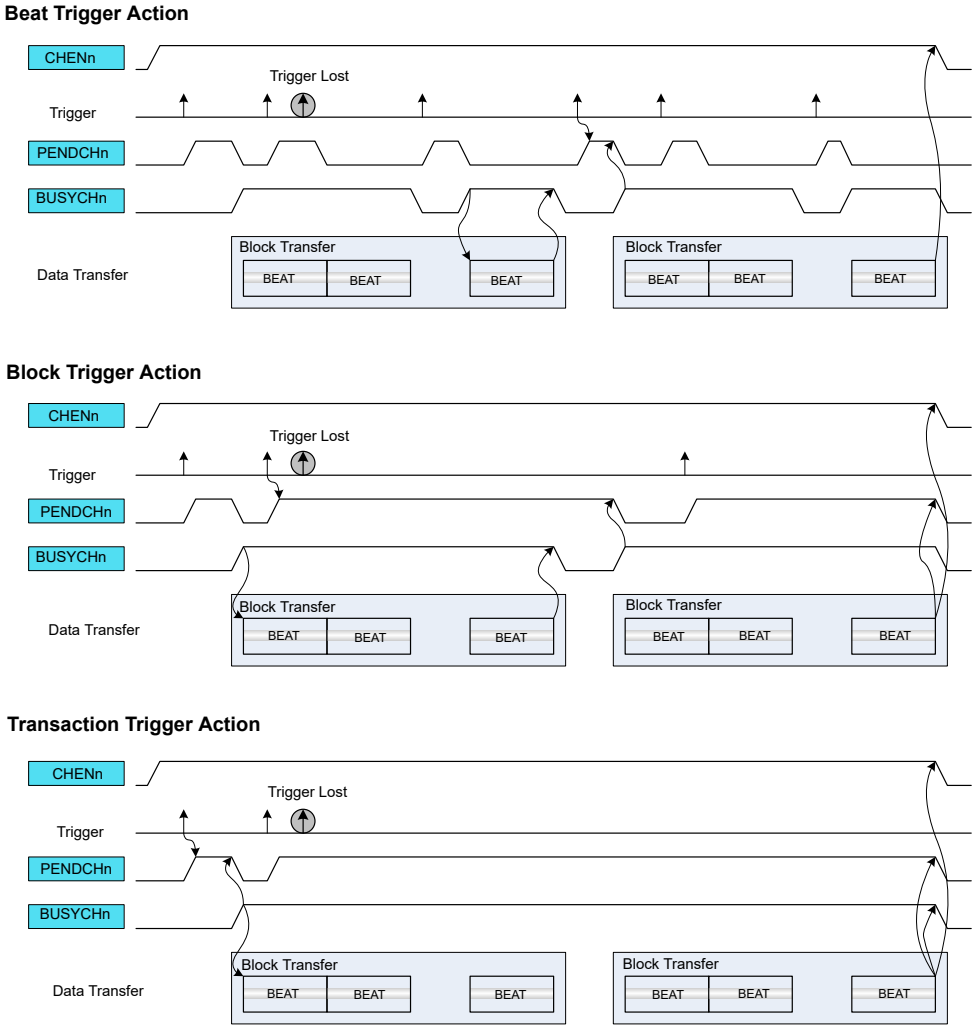
### 20.6.2.6 Transfer Triggers and Actions

A DMA transfer through a DMA channel can be started only when a DMA transfer request is detected, and the DMA channel has been granted access to the DMA. A transfer request can be triggered from software, from a peripheral, or from an event. There are dedicated Trigger Source selections for each DMA Channel Control B (CHCTRLB.TRIGSRC).

The trigger actions are available in the Trigger Action bit group in the Channel Control B register (CHCTRLB.TRIGACT). By default, a trigger generates a request for a block transfer operation. If a single descriptor is defined for a channel, the channel is automatically disabled when a block transfer has been completed. If a list of linked descriptors is defined for a channel, the channel is automatically disabled when the last descriptor in the list is executed. If the list still has descriptors to execute, the channel will be waiting for the next block transfer trigger. When enabled again, the channel will wait for the next block transfer trigger. The trigger actions can also be configured to generate a request for a beat transfer (CHCTRLB.TRIGACT=0x2) or transaction transfer (CHCTRLB.TRIGACT=0x3) instead of a block transfer (CHCTRLB.TRIGACT=0x0).

Figure 20-7 shows an example where triggers are used with two linked block descriptors.

**Figure 20-7. Trigger Action and Transfers**




If the trigger source generates a transfer request for a channel during an ongoing transfer, the new transfer request will be kept pending (CHSTATUS.PEND=1), and the new transfer can start after the

ongoing one is done. Only one pending transfer can be kept per channel. If the trigger source generates more transfer requests while one is already pending, the additional ones will be lost. All channels pending status flags are also available in the Pending Channels register (PENDCH).

When the transfer starts, the corresponding Channel Busy status flag is set in Channel Status register (CHSTATUS.BUSY). When the trigger action is complete, the Channel Busy status flag is cleared. All channel busy status flags are also available in the Busy Channels register (BUSYCH) in DMAC.

#### 20.6.2.7 Addressing

Each block transfer needs to have both a source address and a destination address defined. The source address is set by writing the Transfer Source Address (SRCADDR) register, the destination address is set by writing the Transfer Destination Address (SRCADDR) register.

The addressing of this DMAC module can be static or incremental, for either source or destination of a block transfer, or both.

Incrementation for the source address of a block transfer is enabled by writing the Source Address Incrementation Enable bit in the Block Transfer Control register (BTCTRL.SRCINC=1). The step size of the incrementation is configurable and can be chosen by writing the Step Selection bit in the Block Transfer Control register (BTCTRL.STEPSEL=1) and writing the desired step size in the Address Increment Step Size bit group in the Block Transfer Control register (BTCTRL.STEPSIZE). If BTCTRL.STEPSEL=0, the step size for the source incrementation will be the size of one beat.

When source address incrementation is configured (BTCTRL.SRCINC=1), SRCADDR is calculated as follows:

If BTCTRL.STEPSEL=1:

$$\mathrm{SRCADDR} = \mathrm{SRCADDR}_{START} + BTCNT \cdot \left(BEATSIZE + 1\right) \cdot 2^{\mathrm{STEPSIZE}}$$

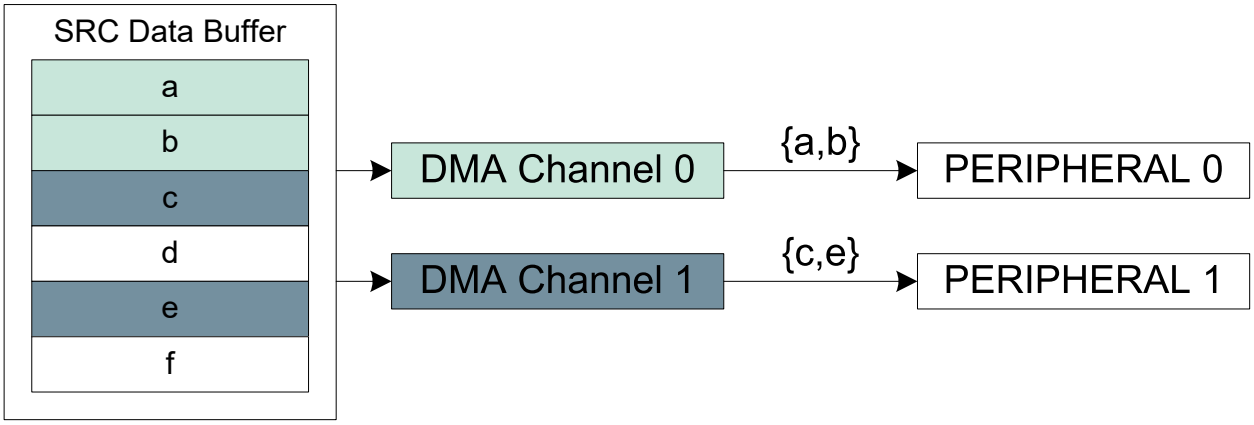
If BTCTRL.STEPSEL=0:

$$\mathrm{SRCADDR} = \mathrm{SRCADDR}_{START} + BTCNT \cdot \left(BEATSIZE + 1\right)$$

- $\mathrm{SRCADDR}_{START}$ is the source address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BEATSIZE is the configured number of bytes in a beat
- STEPSIZE is the configured number of beats for each incrementation

The following figure shows an example where DMA channel 0 is configured to increment the source address by one beat after each beat transfer (BTCTRL.SRCINC=1), and DMA channel 1 is configured to increment the source address by two beats (BTCTRL.SRCINC=1, BTCTRL.STEPSEL=1, and BTCTRL.STEPSIZE=0x1). As the destination address for both channels are peripherals, destination incrementation is disabled (BTCTRL.DSTINC=0).

**Figure 20-8. Source Address Increment**




Incrementation for the destination address of a block transfer is enabled by setting the Destination Address Incrementation Enable bit in the Block Transfer Control register (BTCTRL.DSTINC=1). The step size of the incrementation is configurable by clearing BTCTRL.STEPSEL=0 and writing BTCTRL.STEPSIZE to the desired step size. If BTCTRL.STEPSEL=1, the step size for the destination incrementation will be the size of one beat.

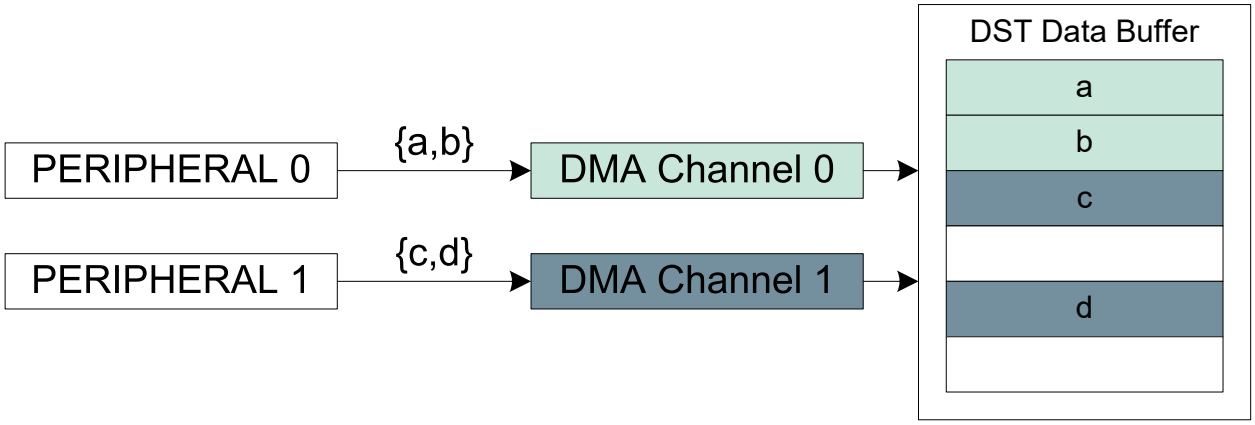
When the destination address incrementation is configured (BTCTRL.DSTINC=1), DSTADDR must be set and calculated as follows:

| | |
|---|---|
| $DSTADDR = DSTADDR_{START} + BTCNT \bullet \left(BEATSIZE + 1\right) \bullet 2^{STEPSIZE}$ | where BTCTRL.STEPSEL is zero |
| $DSTADDR = DSTADDR_{START} + BTCNT \bullet (BEATSIZE + 1)$ | where BTCTRL.STEPSEL is one |

- $DSTADDR_{START}$ is the destination address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BEATSIZE is the configured number of bytes in a beat
- STEPSIZE is the configured number of beats for each incrementation

The followiong figure shows an example where DMA channel 0 is configured to increment destination address by one beat (BTCTRL.DSTINC=1) and DMA channel 1 is configured to increment destination address by two beats (BTCTRL.DSTINC=1, BTCTRL.STEPSEL=0, and BTCTRL.STEPSIZE=0x1). As the source address for both channels are peripherals, source incrementation is disabled (BTCTRL.SRCINC=0).

**Figure 20-9. Destination Address Increment**




#### 20.6.2.8 Error Handling

If a bus error is received from an AHB slave during a DMA data transfer, the corresponding active channel is disabled and the corresponding Channel Transfer Error Interrupt flag in the Channel Interrupt Status and Clear register (CHINTFLAG.TERR) is set. If enabled, the optional transfer error interrupt is generated. The transfer counter will not be decremented and its current value is written-back in the write-back memory section before the channel is disabled.

When the DMAC fetches an invalid descriptor (BTCTRL.VALID=0) or when the channel is resumed and the DMA fetches the next descriptor with null address (DESCADDR=0x00000000), the corresponding channel operation is suspended, the Channel Suspend Interrupt Flag in the Channel Interrupt Flag Status and Clear register (CHINTFLAG.SUSP) is set, and the Channel Fetch Error bit in the Channel Status register (CHSTATUS.FERR) is set. If enabled, the optional suspend interrupt is generated.

### 20.6.3 Additional Features

#### 20.6.3.1 Linked Descriptors

A transaction can consist of either a single block transfer or of several block transfers. When a transaction consists of several block transfers it is done with the help of linked descriptors.

Figure 20-3 illustrates how linked descriptors work. When the first block transfer is completed on DMA channel 0, the DMAC fetches the next transfer descriptor, which is pointed to by the value stored in the Next Descriptor Address (DESCADDR) register of the first transfer descriptor. Fetching the next transfer descriptor (DESCADDR) is continued until the last transfer descriptor. When the block transfer for the last transfer descriptor is executed and DESCADDR=0x00000000, the transaction is terminated. For further details on how the next descriptor is fetched from SRAM, refer to section 20.6.2.5 Data Transmission.

##### 20.6.3.1.1 Adding Descriptor to the End of a List

To add a new descriptor at the end of the descriptor list, create the descriptor in SRAM, with DESCADDR=0x00000000 indicating that it is the new last descriptor in the list, and modify the DESCADDR value of the current last descriptor to the address of the newly created descriptor.

##### 20.6.3.1.2 Modifying a Descriptor in a List

In order to add descriptors to a linked list, the following actions must be performed:

1. Enable the Suspend interrupt for the DMA channel.
2. Enable the DMA channel.
3. Reserve memory space in SRAM to configure a new descriptor.
4. Configure the new descriptor:

- Set the next descriptor address (DESCADDR)
- Set the destination address (DSTADDR)
- Set the source address (SRCADDR)
- Configure the block transfer control (BTCTRL) including
  - Optionally enable the Suspend block action
  - Set the descriptor VALID bit

5. Clear the VALID bit for the existing list and for the descriptor which has to be updated.
6. Read DESCADDR from the Write-Back memory.
   - If the DMA has not already fetched the descriptor which requires changes (i.e., DESCADDR is wrong):
     - Update the DESCADDR location of the descriptor from the List
     - Optionally clear the Suspend block action
     - Set the descriptor VALID bit to '1'
     - Optionally enable the Resume software command
   - If the DMA is executing the same descriptor as the one which requires changes:
     - Set the Channel Suspend software command and wait for the Suspend interrupt
     - Update the next descriptor address (DESCRADDR) in the write-back memory
     - Clear the interrupt sources and set the Resume software command
     - Update the DESCADDR location of the descriptor from the List
     - Optionally clear the Suspend block action
     - Set the descriptor VALID bit to '1'
7. Go to step 4 if needed.

#### 20.6.3.1.3 Adding a Descriptor Between Existing Descriptors

To insert a new descriptor 'C' between two existing descriptors ('A' and 'B'), the descriptor currently executed by the DMA must be identified.

1. If DMA is executing descriptor B, descriptor C cannot be inserted.
2. If DMA has not started to execute descriptor A, follow the steps:
   2.1. Set the descriptor A VALID bit to '0'.
   2.2. Set the DESCADDR value of descriptor A to point to descriptor C instead of descriptor B.
   2.3. Set the DESCADDR value of descriptor C to point to descriptor B.
   2.4. Set the descriptor A VALID bit to '1'.
3. If DMA is executing descriptor A:
   3.1. Apply the software suspend command to the channel and
   3.2. Perform steps 2.1 through 2.4.
   3.3. Apply the software resume command to the channel.

### 20.6.3.2 Channel Suspend

The channel operation can be suspended at any time by software by writing a '1' to the Suspend command in the Command bit field of Channel Control B register (CHCTRLB.CMD). After the ongoing burst transfer is completed, the channel operation is suspended and the suspend command is automatically cleared.

When suspended, the Channel Suspend Interrupt flag in the Channel Interrupt Status and Clear register is set (CHINTFLAG.SUSP=1) and the optional suspend interrupt is generated.

By configuring the block action to suspend by writing Block Action bit group in the Block Transfer Control register (BTCTRL.BLOCKACT is 0x2 or 0x3), the DMA channel will be suspended after it has completed a block transfer. The DMA channel will be kept enabled and will be able to receive transfer triggers, but it will be removed from the arbitration scheme.

If an invalid transfer descriptor (BTCTRL.VALID=0) is fetched from SRAM, the DMA channel will be suspended, and the Channel Fetch Error bit in the Channel Status register(CHASTATUS.FERR) will be set.

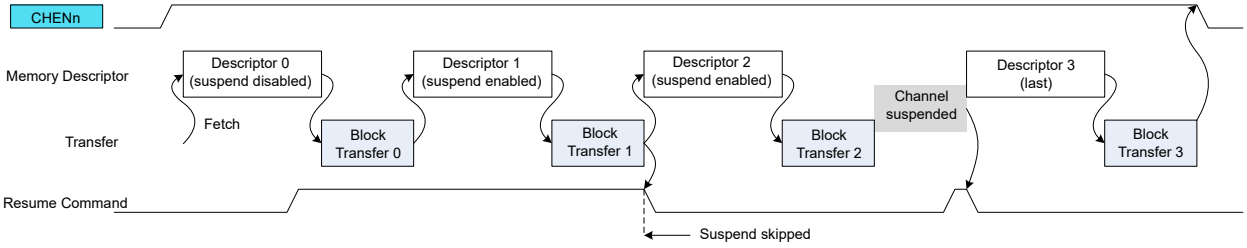
**Note:** Only enabled DMA channels can be suspended. If a channel is disabled when it is attempted to be suspended, the internal suspend command will be ignored.

For more details on transfer descriptors, refer to section 20.6.2.3 Transfer Descriptors.

### 20.6.3.3 Channel Resume and Next Suspend Skip

A channel operation can be resumed by software by setting the Resume command in the Command bit field of the Channel Control B register (CHCTRLB.CMD). If the channel is already suspended, the channel operation resumes from where it previously stopped when the Resume command is detected. When the Resume command is issued before the channel is suspended, the next suspend action is skipped and the channel continues the normal operation.

**Figure 20-10. Channel Suspend/Resume Operation**




### 20.6.3.4 Event Input Actions

The event input actions are available only on the least significant DMA channels. For details on channels with event input support, refer to the in the Event system documentation.

Before using event input actions, the event controller must be configured first according to the following table, and the Channel Event Input Enable bit in the Channel Control B register (CHCTRLB.EVIE) must be written to '1'. Refer also to 20.6.6 Events.

**Table 20-1. Event Input Action**

| Action | CHCTRLB.EVACT | CHCTRLB.TRGSRC |
|---|---|---|
| None | NOACT | - |
| Normal Transfer | TRIG | DISABLE |
| Conditional Transfer on Strobe | TRIG | any peripheral |
| Conditional Transfer | CTRIG | |
| Conditional Block Transfer | CBLOCK | |
| Channel Suspend | SUSPEND | |
| Channel Resume | RESUME | |
| Skip Next Block Suspend | SSKIP | |

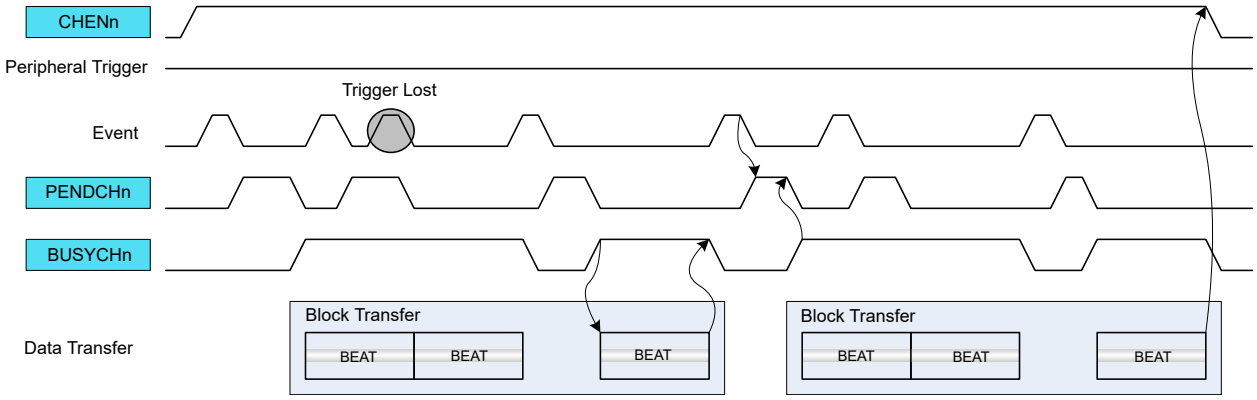
### Normal Transfer

The event input is used to trigger a beat or burst transfer on peripherals.

The event is acknowledged as soon as the event is received. When received, both the Channel Pending status bit in the Channel Status register (CHSTATUS.PEND) and the corresponding Channel n bit in the Pending Channels register (20.8.13 PENDCH.PENDCHn) are set. If the event is received while the channel is pending, the event trigger is lost.

The figure below shows an example where beat transfers are enabled by internal events.

**Figure 20-11. Beat Event Trigger Action**



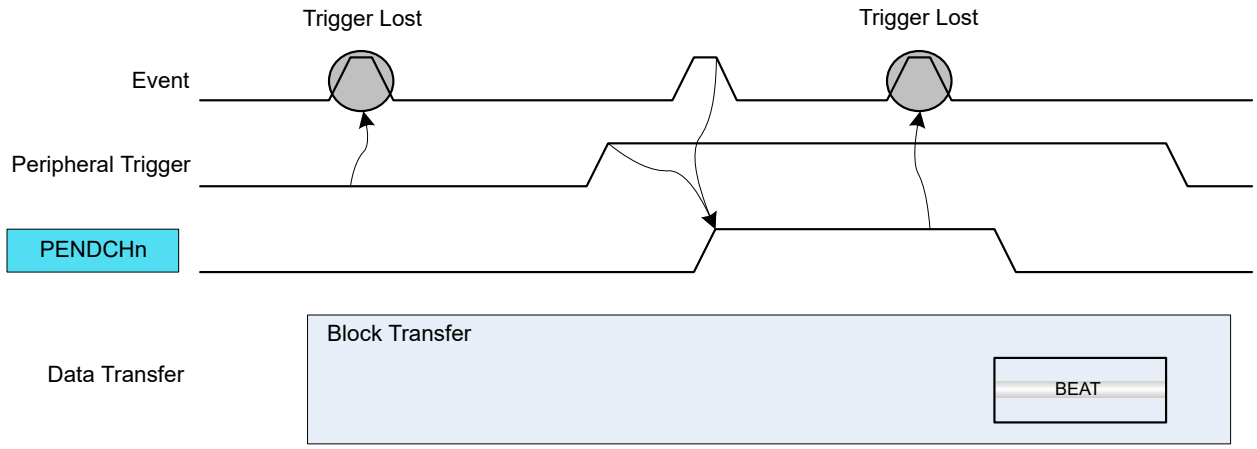

### Conditional Transfer on Strobe

The event input is used to trigger a transfer on peripherals with pending transfer requests. This event action is intended to be used with peripheral triggers, e.g. for timed communication protocols or periodic transfers between peripherals: only when the peripheral trigger coincides with the occurrence of a (possibly cyclic) event the transfer is issued.

The event is acknowledged as soon as the event is received. The peripheral trigger request is stored internally when the previous trigger action is completed (i.e. the channel is not pending) and when an active event is received. If the peripheral trigger is active, the DMA will wait for an event before the peripheral trigger is internally registered. When both event and peripheral transfer trigger are active, both CHSTATUS.PEND and 20.8.13 PENDCH.PENDCHn are set. A software trigger will now trigger a transfer.

The figure below shows an example where the peripheral beat transfer is started by a conditional strobe event action.

**Figure 20-12. Periodic Event with Beat Peripheral Triggers**



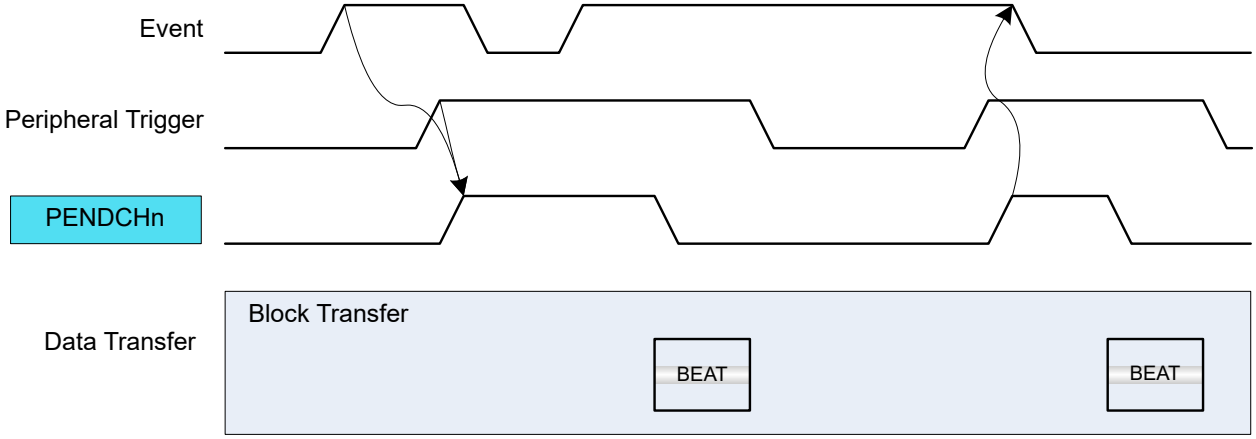

### Conditional Transfer

The event input is used to trigger a conditional transfer on peripherals with pending transfer requests. For example, this type of event can be used for peripheral-to-peripheral transfers, where one peripheral is the source of event and the second peripheral is the source of the trigger.

Each peripheral trigger is stored internally when the event is received. When the peripheral trigger is stored internally, the Channel Pending status bit is set (CHSTATUS.PEND), the respective Pending Channel n Bit in the Pending Channels register is set (20.8.13 PENDCH.PENDCHn), and the event is acknowledged. A software trigger will now trigger a transfer.

The figure below shows an example where conditional event is enabled with peripheral beat trigger requests.

**Figure 20-13. Conditional Event with Beat Peripheral Triggers**



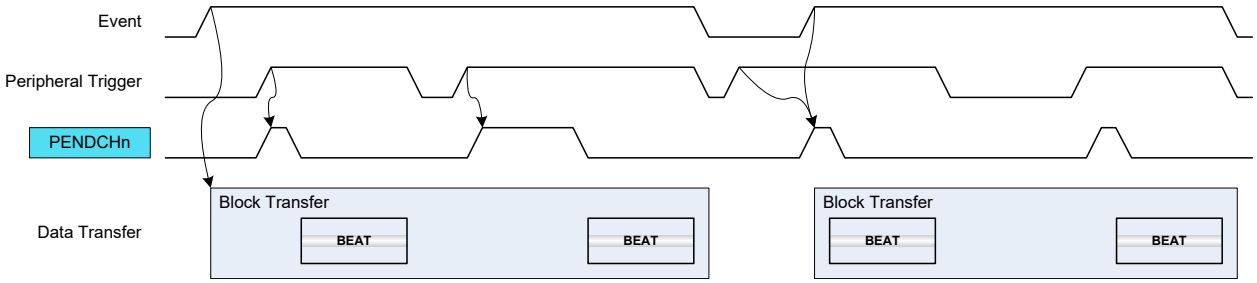

### Conditional Block Transfer

The event input is used to trigger a conditional block transfer on peripherals.

Before starting transfers within a block, an event must be received. When received, the event is acknowledged when the block transfer is completed. A software trigger will trigger a transfer.

The figure below shows an example where conditional event block transfer is started with peripheral beat trigger requests.

**Figure 20-14. Conditional Block Transfer with Beat Peripheral Triggers**




**Channel Suspend**

The event input is used to suspend an ongoing channel operation. The event is acknowledged when the current AHB access is completed. For further details on Channel Suspend, refer to 20.6.3.2 Channel Suspend.

**Channel Resume**

The event input is used to resume a suspended channel operation. The event is acknowledged as soon as the event is received and the Channel Suspend Interrupt Flag (CHINTFLAG.SUSP) is cleared. For further details refer to 20.6.3.2 Channel Suspend.

**Skip Next Block Suspend**

This event can be used to skip the next block suspend action. If the channel is suspended before the event rises, the channel operation is resumed and the event is acknowledged. If the event rises before a suspend block action is detected, the event is kept until the next block suspend detection. When the block transfer is completed, the channel continues the operation (not suspended) and the event is acknowledged.

**Related Links**

24.8.3 USER

### 20.6.3.5 Event Output Selection

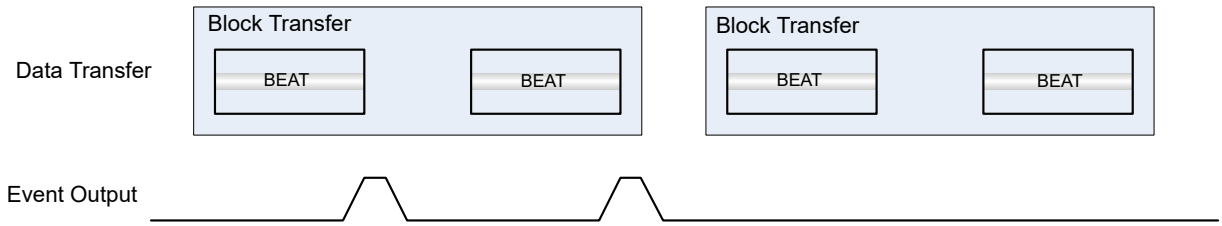
Event output selection is available only for the least significant DMA channels. The pulse width of an event output from a channel is one AHB clock cycle.

The output of channel events is enabled by writing a '1' to the Channel Event Output Enable bit in the Control B register (CHCTRLB.EVOE). The event output cause is selected by writing to the Event Output Selection bits in the Block Transfer Control register (BTCTRL.EVOSEL). It is possible to generate events after each block transfer (BTCTRL.EVOSEL=0x1) or beat transfer (BTCTRL.EVOSEL=0x3). To enable an event being generated when a transaction is complete, the block event selection must be set in the last transfer descriptor only.

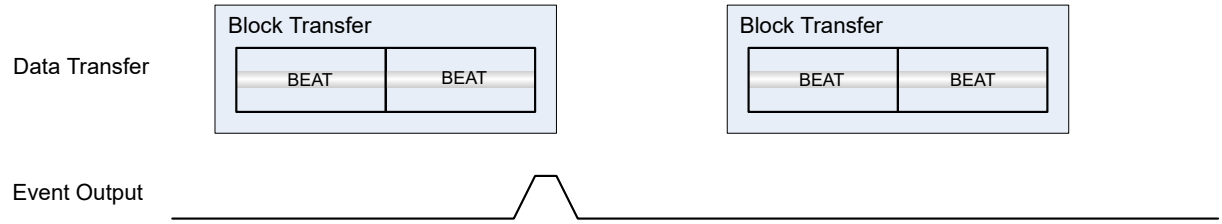
Figure 20-15 shows an example where the event output generation is enabled in the first block transfer, and disabled in the second block.

**Figure 20-15. Event Output Generation**

**Beat Event Output**




**Block Event Output**




#### 20.6.3.6 Aborting Transfers

Transfers on any channel can be aborted gracefully by software by disabling the corresponding DMA channel. It is also possible to abort all ongoing or pending transfers by disabling the DMAC.

When a DMA channel disable request or DMAC disable request is detected:

- Ongoing transfers of the active channel will be disabled when the ongoing beat transfer is completed and the write-back memory section is updated. This prevents transfer corruption before the channel is disabled.
- All other enabled channels will be disabled in the next clock cycle.

The corresponding Channel Enable bit in the Channel Control A register is cleared (CHCTRLA.ENABLE=0) when the channel is disabled.

The corresponding DMAC Enable bit in the Control register is cleared (CTRL.DMAENABLE=0) when the entire DMAC module is disabled.

#### 20.6.3.7 CRC Operation

A cyclic redundancy check (CRC) is an error detection technique used to find errors in data. It is commonly used to determine whether the data during a transmission, or data present in data and program memories has been corrupted or not. A CRC takes a data stream or a block of data as input and generates a 16- or 32-bit output that can be appended to the data and used as a checksum.

When the data is received, the device or application repeats the calculation: If the new CRC result does not match the one calculated earlier, the block contains a data error. The application will then detect this and may take a corrective action, such as requesting the data to be sent again or simply not using the incorrect data.

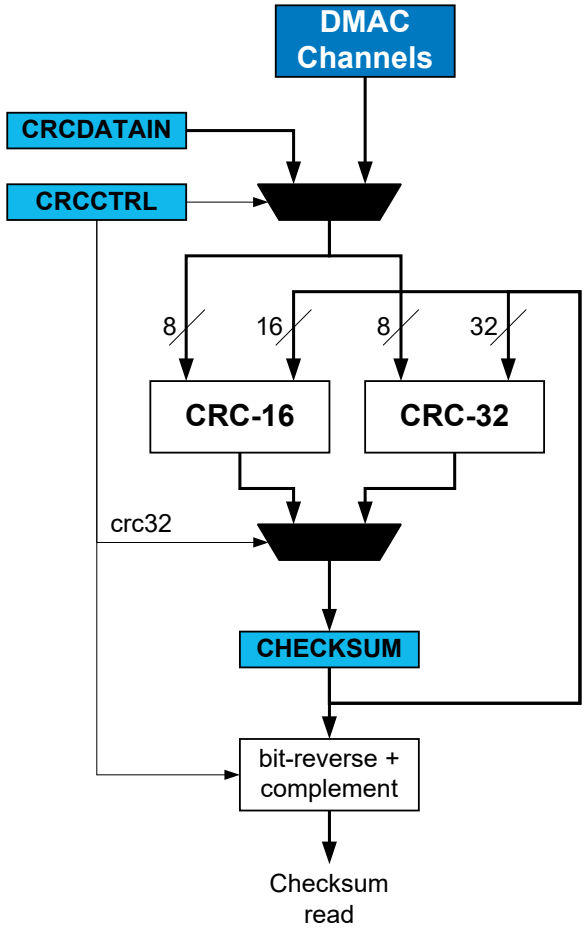
The CRC engine in DMAC supports two commonly used CRC polynomials: CRC-16 (CRC-CCITT) and CRC-32 (IEEE 802.3). Typically, applying CRC-n (CRC-16 or CRC-32) to a data block of arbitrary length will detect any single alteration that is ≤n bits in length, and will detect the fraction 1-2-n of all longer error bursts.

- CRC-16:
  - Polynomial: $x^{16}+ x^{12}+ x^5+ 1$
  - Hex value: 0x1021
- CRC-32:
  - Polynomial: $x^{32}+x^{26}+ x^{23}+ x^{22}+x^{16}+ x^{12}+ x^{11}+ x^{10}+ x^8+ x^7+ x^5+ x^4+ x^2+ x + 1$
  - Hex value: 0x04C11DB7

The data source for the CRC engine can either be one of the DMA channels or the APB bus interface, and must be selected by writing to the CRC Input Source bits in the CRC Control register (CRCCTRL.CRCSRC). The CRC engine then takes data input from the selected source and generates a checksum based on these data. The checksum is available in the CRC Checksum register (CRCCHKSUM). When CRC-32 polynomial is used, the final checksum read is bit reversed and complemented, as shown in Figure 20-16.

The CRC polynomial is selected by writing to the CRC Polynomial Type bit in the CRC Control register (CRCCTRL.CRCPOLY), the default is CRC-16. The CRC engine operates on byte only. When the DMA is used as data source for the CRC engine, the DMA channel beat size setting will be used. When used with APB bus interface, the application must select the CRC Beat Size bit field of CRC Control register (CRCCTRL.CRCBEATSIZE). 8-, 16-, or 32-bit bus transfer access type is supported. The corresponding number of bytes will be written in the CRCDATAIN register and the CRC engine will operate on the input data in a byte by byte manner.

**Figure 20-16. CRC Generator Block Diagram**



**CRC on DMA data** CRC-16 or CRC-32 calculations can be performed on data passing through any DMA channel. Once a DMA channel is selected as the source, the CRC engine will continuously generate the CRC on the data passing through the DMA channel. The checksum is available for readout once the DMA transaction is completed or aborted. A CRC can also be generated on SRAM, Flash, or I/O memory by passing these data through a DMA channel. If the latter is done, the destination register for the DMA data can be the data input (CRCDATAIN) register in the CRC engine.

**CRC using the I/O interface** Before using the CRC engine with the I/O interface, the application must set the CRC Beat Size bits in the CRC Control register (CRCCTRL.CRCBEATSIZE). 8/16/32-bit bus transfer type can be selected.

CRC can be performed on any data by loading them into the CRC engine using the CPU and writing the data to the CRCDATAIN register. Using this method, an arbitrary number of bytes can be written to the register by the CPU, and CRC is done continuously for each byte. This means if a 32-bit data is written to the CRCDATAIN register the CRC engine takes four cycles to calculate the CRC. The CRC complete is signaled by a set CRCBUSY bit in the CRCSTATUS register. New data can be written only when CRCBUSY flag is not set.

### 20.6.4 DMA Operation

Not applicable.

### 20.6.5 Interrupts

The DMAC channels have the following interrupt sources:

- Transfer Complete (TCMPL): Indicates that a block transfer is completed on the corresponding channel. Refer to 20.6.2.5 Data Transmission for details.
- Transfer Error (TERR): Indicates that a bus error has occurred during a burst transfer, or that an invalid descriptor has been fetched. Refer to 20.6.2.8 Error Handling for details.
- Channel Suspend (SUSP): Indicates that the corresponding channel has been suspended. Refer to 20.6.3.2 Channel Suspend and 20.6.2.5 Data Transmission for details.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Channel Interrupt Flag Status and Clear (CHINTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by setting the corresponding bit in the Channel Interrupt Enable Set register (CHINTENSET=1), and disabled by setting the corresponding bit in the Channel Interrupt Enable Clear register (CHINTENCLR=1).

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, the DMAC is reset or the corresponding DMA channel is reset. See CHINTFLAG for details on how to clear interrupt flags. All interrupt requests are ORed together on system level to generate one combined interrupt request to the NVIC.

The user must read the Channel Interrupt Status (INTSTATUS) register to identify the channels with pending interrupts and must read the Channel Interrupt Flag Status and Clear (CHINTFLAG) register to determine which interrupt condition is present for the corresponding channel. It is also possible to read the Interrupt Pending register (INTPEND), which provides the lowest channel number with pending interrupt and the respective interrupt flags.

**Note:** Interrupts must be globally enabled for interrupt requests to be generated.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 20.6.6 Events

The DMAC can generate the following output events:

- Channel (CH): Generated when a block transfer for a given channel has been completed, or when a beat transfer within a block transfer for a given channel has been completed. Refer to *Event Output Selection* for details.

Setting the Channel Event Output Enable bit (CHEVCTRLx.EVOE = 1) enables the corresponding output event configured in the Event Output Selection bit group in the Block Transfer Control register (BTCTRL.EVOSEL). Clearing CHEVCTRLx.EVOE = 0 disables the corresponding output event.

The DMAC can take the following actions on an input event:

- Transfer and Periodic Transfer Trigger (TRIG): normal transfer or periodic transfers on peripherals are enabled
- Conditional Transfer Trigger (CTRIG): conditional transfers on peripherals are enabled
- Conditional Block Transfer Trigger (CBLOCK): conditional block transfers on peripherals are enabled
- Channel Suspend Operation (SUSPEND): suspend a channel operation
- Channel Resume Operation (RESUME): resume a suspended channel operation
- Skip Next Block Suspend Action (SSKIP): skip the next block suspend transfer condition
- Increase Priority (INCPRI): increase channel priority

Setting the Channel Event Input Enable bit (CHEVCTRLx.EVIE = 1) enables the corresponding action on input event. Clearing this bit disables the corresponding action on input event. Note that several actions can be enabled for incoming events. If several events are connected to the peripheral, any enabled action will be taken for any of the incoming events. For further details on event input actions, refer to *Event Input Actions*.

**Note:** Event input and outputs are not available for every channel. Refer to the Features section for more information.

**Related Links**

24. EVSYS – Event System

### 20.6.7 Sleep Mode Operation

In Standby Sleep mode, the DMAC will be internally disabled, but maintains its current configuration.

### 20.6.8 Synchronization

Not applicable.

## 20.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | | | CRCENABLE | DMAENABLE | SWRST |
| | | 15:8 | | | | | LVLENx3 | LVLENx2 | LVLENx1 | LVLENx0 |
| 0x02 | CRCCTRL | 7:0 | | | | | CRCPOLY[1:0] | | CRCBEATSIZE[1:0] | |
| | | 15:8 | | | CRCSRC[5:0] | | | | | |
| 0x04 | CRCDATAIN | 7:0 | CRCDATAIN[7:0] | | | | | | | |
| | | 15:8 | CRCDATAIN[15:8] | | | | | | | |
| | | 23:16 | CRCDATAIN[23:16] | | | | | | | |
| | | 31:24 | CRCDATAIN[31:24] | | | | | | | |
| 0x08 | CRCCHKSUM | 7:0 | CRCCHKSUM[7:0] | | | | | | | |
| | | 15:8 | CRCCHKSUM[15:8] | | | | | | | |
| | | 23:16 | CRCCHKSUM[23:16] | | | | | | | |
| | | 31:24 | CRCCHKSUM[31:24] | | | | | | | |
| 0x0C | CRCSTATUS | 7:0 | | | | | | | CRCZERO | CRCBUSY |
| 0x0D | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x0E | QOSCTRL | 7:0 | | | DQOS[1:0] | | FQOS[1:0] | | WRBQOS[1:0] | |
| 0x0F | Reserved | | | | | | | | | |
| 0x10 | SWTRIGCTRL | 7:0 | SWTRIGn[7:0] | | | | | | | |
| | | 15:8 | | | | | SWTRIGn[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | PRICTRL0 | 7:0 | RRLVLEN0 | | | | LVLPRI0[3:0] | | | |
| | | 15:8 | RRLVLEN1 | | | | LVLPRI1[3:0] | | | |
| | | 23:16 | RRLVLEN2 | | | | LVLPRI2[3:0] | | | |
| | | 31:24 | RRLVLEN3 | | | | LVLPRI3[3:0] | | | |
| 0x18<br>...<br>0x1F | Reserved | | | | | | | | | |
| 0x20 | INTPEND | 7:0 | | | | | ID[3:0] | | | |
| | | 15:8 | PEND | BUSY | FERR | | | SUSP | TCMPL | TERR |
| 0x22<br>...<br>0x23 | Reserved | | | | | | | | | |
| 0x24 | INTSTATUS | 7:0 | CHINTn[7:0] | | | | | | | |
| | | 15:8 | | | | | CHINTn[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | BUSYCH | 7:0 | BUSYCHn[7:0] | | | | | | | |
| | | 15:8 | | | | | BUSYCHn[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | PENDCH | 7:0 | PENDCH7 | PENDCH6 | PENDCH5 | PENDCH4 | PENDCH3 | PENDCH2 | PENDCH1 | PENDCH0 |
| | | 15:8 | | | | | PENDCH11 | PENDCH10 | PENDCH9 | PENDCH8 |
| | | 23:16 | | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 31:24 | | | | | | | | |
| 0x30 | ACTIVE | 7:0 | | | | | LVLEXx | LVLEXx | LVLEXx | LVLEXx |
| | | 15:8 | ABUSY | | | ID[4:0] | | | | |
| | | 23:16 | BTCNT[7:0] | | | | | | | |
| | | 31:24 | BTCNT[15:8] | | | | | | | |
| 0x34 | BASEADDR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | WRBADDR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C ... 0x3E | Reserved | | | | | | | | | |
| 0x3F | CHID | 7:0 | | | | | ID[3:0] | | | |
| 0x40 | CHCTRLA | 7:0 | | | | | | | ENABLE | SWRST |
| 0x41 ... 0x43 | Reserved | | | | | | | | | |
| 0x44 | CHCTRLB | 7:0 | | LVL[1:0] | | EVOE | EVIE | EVACT[2:0] | | |
| | | 15:8 | | | TRIGSRC[5:0] | | | | | |
| | | 23:16 | TRIGACT[1:0] | | | | | | | |
| | | 31:24 | | | | | | | CMD[1:0] | |
| 0x48 ... 0x4B | Reserved | | | | | | | | | |
| 0x4C | CHINTENCLR | 7:0 | | | | | | SUSP | TCMPL | TERR |
| 0x4D | CHINTENSET | 7:0 | | | | | | SUSP | TCMPL | TERR |
| 0x4E | CHINTFLAG | 7:0 | | | | | | SUSP | TCMPL | TERR |
| 0x4F | CHSTATUS | 7:0 | | | | | | FERR | BUSY | PEND |

## 20.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 20.5.8 Register Access Protection.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 20.8.1 Control

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x00X0
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LVLENx3 | LVLENx2 | LVLENx1 | LVLENx0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CRCENABLE | DMAENABLE | SWRST |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 8, 9, 10, 11 – LVLENx** Priority Level x Enable
When this bit is set, all requests with the corresponding level will be fed into the arbiter block. When cleared, all requests with the corresponding level will be ignored.

For details on arbitration schemes, refer to the Arbitration section.

These bits are not enable-protected.

| Value | Description |
|---|---|
| 0 | Transfer requests for Priority level x will not be handled. |
| 1 | Transfer requests for Priority level x will be handled. |

**Bit 2 – CRCENABLE** CRC Enable
Writing a '0' to this bit will disable the CRC calculation when the CRC Status Busy flag is cleared (CRCSTATUS. CRCBUSY). The bit is zero when the CRC is disabled.

Writing a '1' to this bit will enable the CRC calculation.

| Value | Description |
|---|---|
| 0 | The CRC calculation is disabled. |
| 1 | The CRC calculation is enabled. |

**Bit 1 – DMAENABLE** DMA Enable
Setting this bit will enable the DMA module.

Writing a '0' to this bit will disable the DMA module. When writing a '0' during an ongoing transfer, the bit will not be cleared until the internal data transfer buffer is empty and the DMA transfer is aborted. The internal data transfer buffer will be empty once the ongoing burst transfer is completed.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit when both the DMAC and the CRC module are disabled (DMAENABLE and CRCENABLE are '0') resets all registers in the DMAC (except DBGCTRL) to their initial state. If either the DMAC or CRC module is enabled, the Reset request will be ignored and the DMAC will return an access error.

| Value | Description |
|---|---|
| 0 | There is no Reset operation ongoing. |
| 1 | A Reset operation is ongoing. |

### 20.8.2 CRC Control

**Name:** CRCCTRL
**Offset:** 0x02
**Reset:** 0x0000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CRCSRC[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CRCPOLY[1:0] | | CRCBEATSIZE[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 13:8 – CRCSRC[5:0]** CRC Input Source
These bits select the input source for generating the CRC, as shown in the table below. The selected source is locked until either the CRC generation is completed or the CRC module is disabled. This means the CRCSRC cannot be modified when the CRC operation is ongoing. The lock is signaled by the CRCBUSY status bit. CRC generation complete is generated and signaled from the selected source when used with the DMA channel.

| Value | Name | Description |
|---|---|---|
| 0x00 | NOACT | No action |
| 0x01 | IO | I/O interface |
| 0x02-0x1F | - | Reserved |
| 0x20 | CHN | DMA channel 0 |
| 0x21 | CHN | DMA channel 1 |
| 0x22 | CHN | DMA channel 2 |
| 0x23 | CHN | DMA channel 3 |
| 0x24 | CHN | DMA channel 4 |
| 0x25 | CHN | DMA channel 5 |
| 0x26 | CHN | DMA channel 6 |
| 0x27 | CHN | DMA channel 7 |
| 0x28 | CHN | DMA channel 8 |
| 0x29 | CHN | DMA channel 9 |
| 0x2A | CHN | DMA channel 10 |
| 0x2B | CHN | DMA channel 11 |
| 0x2C | CHN | DMA channel 12 |
| 0x2D | CHN | DMA channel 13 |
| 0x2E | CHN | DMA channel 14 |
| 0x2F | CHN | DMA channel 15 |
| 0x30 | CHN | DMA channel 16 |
| 0x31 | CHN | DMA channel 17 |
| 0x32 | CHN | DMA channel 18 |

| Value | Name | Description |
|---|---|---|
| 0x33 | CHN | DMA channel 19 |
| 0x34 | CHN | DMA channel 20 |
| 0x35 | CHN | DMA channel 21 |
| 0x36 | CHN | DMA channel 22 |
| 0x37 | CHN | DMA channel 23 |
| 0x38 | CHN | DMA channel 24 |
| 0x39 | CHN | DMA channel 25 |
| 0x3A | CHN | DMA channel 26 |
| 0x3B | CHN | DMA channel 27 |
| 0x3C | CHN | DMA channel 28 |
| 0x3D | CHN | DMA channel 29 |
| 0x3E | CHN | DMA channel 30 |
| 0x3F | CHN | DMA channel 31 |

**Bits 3:2 – CRCPOLY[1:0]** CRC Polynomial Type
These bits define the size of the data transfer for each bus access when the CRC is used with I/O interface, as shown in the table below.

| Value | Name | Description |
|---|---|---|
| 0x0 | CRC16 | CRC-16 (CRC-CCITT) |
| 0x1 | CRC32 | CRC32 (IEEE 802.3) |
| 0x2-0x3 | | Reserved |

**Bits 1:0 – CRCBEATSIZE[1:0]** CRC Beat Size
These bits define the size of the data transfer for each bus access when the CRC is used with I/O interface.

| Value | Name | Description |
|---|---|---|
| 0x0 | BYTE | 8-bit bus transfer |
| 0x1 | HWORD | 16-bit bus transfer |
| 0x2 | WORD | 32-bit bus transfer |
| 0x3 | | Reserved |

### 20.8.3 CRC Data Input

**Name:** CRCDATAIN
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAIN[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAIN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAIN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAIN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CRCDATAIN[31:0]** CRC Data Input
These bits store the data for which the CRC checksum is computed. A new CRC Checksum is ready (CRCBEAT+ 1) clock cycles after the CRCDATAIN register is written.

### 20.8.4 CRC Checksum

**Name:** CRCCHKSUM
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

The CRCCHKSUM represents the 16- or 32-bit checksum value and the generated CRC. The register is reset to zero by default, but it is possible to reset all bits to one by writing the CRCCHKSUM register directly. It is possible to write this register only when the CRC module is disabled. If CRC-32 is selected and the CRC Status Busy flag is cleared (i.e., CRC generation is completed or aborted), the bit reversed (bit 31 is swapped with bit 0, bit 30 with bit 1, etc.) and complemented result will be read from CRCCHKSUM. If CRC-16 is selected or the CRC Status Busy flag is set (i.e., CRC generation is ongoing), CRCCHKSUM will contain the actual content.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCCHKSUM[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCCHKSUM[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCCHKSUM[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCCHKSUM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CRCCHKSUM[31:0]** CRC Checksum
These bits store the generated CRC result. The 16 MSB bits are always read zero when CRC-16 is enabled.

### 20.8.5 CRC Status

**Name:** CRCSTATUS
**Offset:** 0x0C
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CRCZERO | CRCBUSY |
| Access | | | | | | | R | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – CRCZERO** CRC Zero
This bit is cleared when a new CRC source is selected.

This bit is set when the CRC generation is complete and the CRC Checksum is zero.

When running CRC-32 and appending the checksum at the end of the packet (as little endian), the final checksum should be 0x2144df1c, and not zero. However, if the checksum is complemented before it is appended (as little endian) to the data, the final result in the checksum register will be zero. See the description of CRCCHKSUM to read out different versions of the checksum.

**Bit 0 – CRCBUSY** CRC Module Busy
This flag is cleared by writing a one to it when used with I/O interface. When used with a DMA channel, the bit is set when the corresponding DMA channel is enabled, and cleared when the corresponding DMA channel is disabled. This register bit cannot be cleared by the application when the CRC is used with a DMA channel.

This bit is set when a source configuration is selected and as long as the source is using the CRC module.

### 20.8.6 Debug Control

**Name:** DBGCTRL
**Offset:** 0x0D
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGRUN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGRUN** Debug Run
This bit is not reset by a software reset.

This bit controls the functionality when the CPU is halted by an external debugger.

| Value | Description |
|---|---|
| 0 | The DMAC is halted when the CPU is halted by an external debugger. |
| 1 | The DMAC continues normal operation when the CPU is halted by an external debugger. |

### 20.8.7 Quality of Service Control

**Name:** QOSCTRL
**Offset:** 0x0E
**Reset:** 0x2A
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DQOS[1:0] | | FQOS[1:0] | | WRBQOS[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 0 | 1 | 0 | 1 | 0 |

**Bits 5:4 – DQOS[1:0]** Data Transfer Quality of Service
These bits define the memory priority access during the data transfer operation.

| DQOS[1:0] | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Background (no sensitive operation) |
| 0x1 | LOW | Sensitive Bandwidth |
| 0x2 | MEDIUM | Sensitive Latency |
| 0x3 | HIGH | Critical Latency |

**Bits 3:2 – FQOS[1:0]** Fetch Quality of Service
These bits define the memory priority access during the fetch operation.

| FQOS[1:0] | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Background (no sensitive operation) |
| 0x1 | LOW | Sensitive Bandwidth |
| 0x2 | MEDIUM | Sensitive Latency |
| 0x3 | HIGH | Critical Latency |

**Bits 1:0 – WRBQOS[1:0]** Write-Back Quality of Service
These bits define the memory priority access during the write-back operation.

| WRBQOS[1:0] | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Background (no sensitive operation) |
| 0x1 | LOW | Sensitive Bandwidth |
| 0x2 | MEDIUM | Sensitive Latency |
| 0x3 | HIGH | Critical Latency |

**Related Links**
11.4.3 SRAM Quality of Service

### 20.8.8 Software Trigger Control

**Name:** SWTRIGCTRL
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SWTRIGn[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWTRIGn[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – SWTRIGn[11:0]** Channel n Software Trigger [n = 11..0]
This bit is cleared when the Channel Pending bit in the Channel Status register (CHSTATUS.PEND) for the corresponding channel is either set, or by writing a '1' to it.

This bit is set if CHSTATUS.PEND is already '1' when writing a '1' to that bit.

Writing a '0' to this bit will clear the bit.

Writing a '1' to this bit will generate a DMA software trigger on channel x, if CHSTATUS.PEND=0 for channel x. CHSTATUS.PEND will be set and SWTRIGn will remain cleared.

### 20.8.9 Priority Control 0

**Name:** PRICTRL0
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RRLVLEN3 | | | | LVLPRI3[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RRLVLEN2 | | | | LVLPRI2[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RRLVLEN1 | | | | LVLPRI1[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RRLVLEN0 | | | | LVLPRI0[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 31 – RRLVLEN3** Level 3 Round-Robin Arbitration Enable
This bit controls which arbitration scheme is selected for DMA channels with priority level 3. For details on arbitration schemes, refer to 20.6.2.4 Arbitration.

| Value | Description |
|---|---|
| 0 | Static arbitration scheme for channels with level 3 priority. |
| 1 | Round-robin arbitration scheme for channels with level 3 priority. |

**Bits 27:24 – LVLPRI3[3:0]** Level 3 Channel Priority Number
When round-robin arbitration is enabled (PRICTRL0.RRLVLEN3=1) for priority level 3, this register holds the channel number of the last DMA channel being granted access as the active channel with priority level 3.

When static arbitration is enabled (PRICTRL0.RRLVLEN3=0) for priority level 3, and the value of this bit group is non-zero, it will not affect the static priority scheme.

This bit group is not reset when round-robin arbitration gets disabled (PRICTRL0.RRLVLEN3 written to '0').

**Bit 23 – RRLVLEN2** Level 2 Round-Robin Arbitration Enable
This bit controls which arbitration scheme is selected for DMA channels with priority level 2. For details on arbitration schemes, refer to 20.6.2.4 Arbitration.

| Value | Description |
|---|---|
| 0 | Static arbitration scheme for channels with level 2 priority. |
| 1 | Round-robin arbitration scheme for channels with level 2 priority. |

**Bits 19:16 – LVLPRI2[3:0]** Level 2 Channel Priority Number
When round-robin arbitration is enabled (PRICTRL0.RRLVLEN2=1) for priority level 2, this register holds the channel number of the last DMA channel being granted access as the active channel with priority level 2.

When static arbitration is enabled (PRICTRL0.RRLVLEN2=0) for priority level 2, and the value of this bit group is non-zero, it will not affect the static priority scheme.

This bit group is not reset when round-robin arbitration gets disabled (PRICTRL0.RRLVLEN2 written to '0').

**Bit 15 – RRLVLEN1** Level 1 Round-Robin Scheduling Enable
For details on arbitration schemes, refer to 20.6.2.4 Arbitration.

| Value | Description |
|---|---|
| 0 | Static arbitration scheme for channels with level 1 priority. |
| 1 | Round-robin arbitration scheme for channels with level 1 priority. |

**Bits 11:8 – LVLPRI1[3:0]** Level 1 Channel Priority Number
When round-robin arbitration is enabled (PRICTRL0.RRLVLEN1=1) for priority level 1, this register holds the channel number of the last DMA channel being granted access as the active channel with priority level 1.

When static arbitration is enabled (PRICTRL0.RRLVLEN1=0) for priority level 1, and the value of this bit group is non-zero, it will not affect the static priority scheme.

This bit group is not reset when round-robin arbitration gets disabled (PRICTRL0.RRLVLEN1 written to '0').

**Bit 7 – RRLVLEN0** Level 0 Round-Robin Scheduling Enable
For details on arbitration schemes, refer to 20.6.2.4 Arbitration.

| Value | Description |
|---|---|
| 0 | Static arbitration scheme for channels with level 0 priority. |
| 1 | Round-robin arbitration scheme for channels with level 0 priority. |

**Bits 3:0 – LVLPRI0[3:0]** Level 0 Channel Priority Number
When round-robin arbitration is enabled (PRICTRL0.RRLVLEN0=1) for priority level 0, this register holds the channel number of the last DMA channel being granted access as the active channel with priority level 0.

When static arbitration is enabled (PRICTRL0.RRLVLEN0=0) for priority level 0, and the value of this bit group is non-zero, it will not affect the static priority scheme.

This bit group is not reset when round-robin arbitration gets disabled (PRICTRL0.RRLVLEN0 written to '0').

### 20.8.10 Interrupt Pending

**Name:** INTPEND
**Offset:** 0x20
**Reset:** 0x0000
**Property:** -

This register allows the user to identify the lowest DMA channel with pending interrupt.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEND | BUSY | FERR | | | SUSP | TCMPL | TERR |
| Access | R | R | R | | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ID[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 15 – PEND** Pending
This bit will read '1' when the channel selected by Channel ID field (ID) is pending.

**Bit 14 – BUSY** Busy
This bit will read '1' when the channel selected by Channel ID field (ID) is busy.

**Bit 13 – FERR** Fetch Error
This bit will read '1' when the channel selected by Channel ID field (ID) fetched an invalid descriptor.

**Bit 10 – SUSP** Channel Suspend
This bit will read '1' when the channel selected by Channel ID field (ID) has pending Suspend interrupt.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel ID (ID) Suspend interrupt flag.

**Bit 9 – TCMPL** Transfer Complete
This bit will read '1' when the channel selected by Channel ID field (ID) has pending Transfer Complete interrupt.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel ID (ID) Transfer Complete interrupt flag.

**Bit 8 – TERR** Transfer Error
This bit is read one when the channel selected by Channel ID field (ID) has pending Transfer Error interrupt.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel ID (ID) Transfer Error interrupt flag.

**Bits 3:0 – ID[3:0]** Channel ID
These bits store the lowest channel number with pending interrupts. The number is valid if Suspend (SUSP), Transfer Complete (TCMPL) or Transfer Error (TERR) bits are set. The Channel ID field is refreshed when a new channel (with channel number less than the current one) with pending interrupts is detected, or when the application clears the corresponding channel interrupt sources. When no pending channels interrupts are available, these bits will always return zero value when read.

When the bits are written, indirect access to the corresponding Channel Interrupt Flag register is enabled.

### 20.8.11 Interrupt Status

**Name:** INTSTATUS
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHINTn[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHINTn[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – CHINTn[11:0]** Channel n Pending Interrupt [n=11..0]
This bit is set when Channel n has a pending interrupt/the interrupt request is received.

This bit is cleared when the corresponding Channel n interrupts are disabled or the interrupts sources are cleared.

### 20.8.12 Busy Channels

**Name:** BUSYCH
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BUSYCHn[11:8] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUSYCHn[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – BUSYCHn[11:0]** Busy Channel n [x=11..0]
This bit is cleared when the channel trigger action for DMA channel n is complete, when a bus error for DMA channel n is detected, or when DMA channel n is disabled.

This bit is set when DMA channel n starts a DMA transfer.

### 20.8.13 Pending Channels

**Name:** PENDCH
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PENDCH11 | PENDCH10 | PENDCH9 | PENDCH8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PENDCH7 | PENDCH6 | PENDCH5 | PENDCH4 | PENDCH3 | PENDCH2 | PENDCH1 | PENDCH0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – PENDCH** Pending Channel n [n=11..0]
This bit is cleared when trigger execution defined by channel trigger action settings for DMA channel n is started, when a bus error for DMA channel n is detected or when DMA channel n is disabled. For details on trigger action settings, refer to CHCTRLB.TRIGACT.

This bit is set when a transfer is pending on DMA channel n.

### 20.8.14 Active Channel and Levels

**Name:** ACTIVE
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BTCNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BTCNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ABUSY | | | ID[4:0] | | | | |
| Access | R | | | R | R | R | R | R |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LVLEXx | LVLEXx | LVLEXx | LVLEXx |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 31:16 – BTCNT[15:0]** Active Channel Block Transfer Count
These bits hold the 16-bit block transfer count of the ongoing transfer. This value is stored in the active channel and written back in the corresponding Write-Back channel memory location when the arbiter grants a new channel access. The value is valid only when the active channel active busy flag (ABUSY) is set.

**Bit 15 – ABUSY** Active Channel Busy
This bit is cleared when the active transfer count is written back in the write-back memory section.

This bit is set when the next descriptor transfer count is read from the write-back memory section.

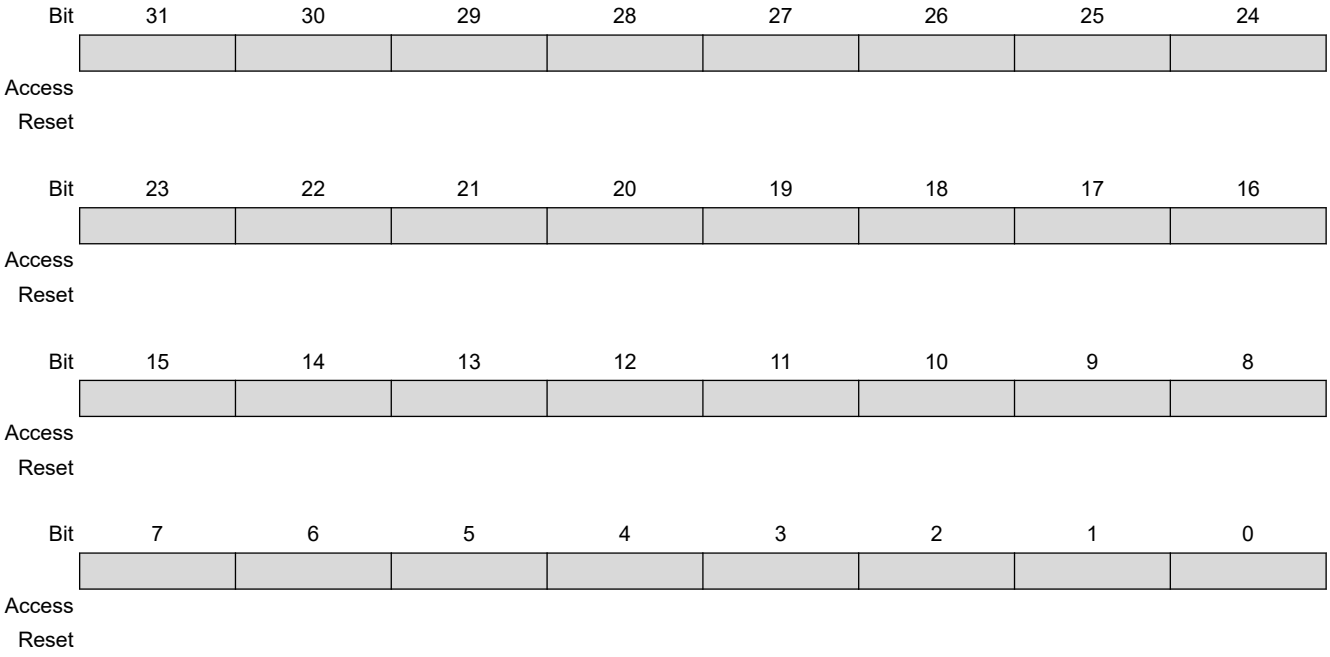
**Bits 12:8 – ID[4:0]** Active Channel ID
These bits hold the channel index currently stored in the active channel registers. The value is updated each time the arbiter grants a new channel transfer access request.

**Bits 3,2,1,0 – LVLEXx** Level x Channel Trigger Request Executing [x=3..0]
This bit is set when a level-x channel trigger request is executing or pending.

This bit is cleared when no request is pending or being executed.

### 20.8.15 Descriptor Memory Section Base Address

**Name:** BASEADDR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected



### 20.8.16 Write-Back Memory Section Base Address

**Name:** WRBADDR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

### 20.8.17 Channel ID

**Name:** CHID
**Offset:** 0x3F
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ID[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – ID[3:0]** Channel ID
These bits define the channel number that will be affected by the channel registers (CH*). Before reading or writing a channel register, the channel ID bit group must be written first.

### 20.8.18 Channel Control A

**Name:** CHCTRLA
**Offset:** 0x40
**Reset:** 0x00
**Property:** PAC Write-Protection, Enable-Protected

This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ENABLE | SWRST |
| Access | R | | R | R | R | R | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 1 – ENABLE** Channel Enable
Writing a '0' to this bit during an ongoing transfer, the bit will not be cleared until the internal data transfer buffer is empty and the DMA transfer is aborted. The internal data transfer buffer will be empty once the ongoing burst transfer is completed.

Writing a '1' to this bit will enable the DMA channel.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | DMA channel is disabled. |
| 1 | DMA channel is enabled. |

**Bit 0 – SWRST** Channel Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets the channel registers to their initial state. The bit can be set when the channel is disabled (ENABLE=0). Writing a '1' to this bit will be ignored as long as ENABLE=1. This bit is automatically cleared when the reset is completed.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 20.8.19 Channel Control B

**Name:** CHCTRLB
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CMD[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRIGACT[1:0] | | | | | | | |
| Access | R/W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TRIGSRC[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LVL[1:0] | | EVOE | EVIE | EVACT[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:24 – CMD[1:0]** Software Command
These bits define the software commands. Refer to 20.6.3.2 Channel Suspend and 20.6.3.3 Channel Resume and Next Suspend Skip.

These bits are not enable-protected.

| CMD[1:0] | Name | Description |
|---|---|---|
| 0x0 | NOACT | No action |
| 0x1 | SUSPEND | Channel suspend operation |
| 0x2 | RESUME | Channel resume operation |
| 0x3 | - | Reserved |

**Bits 23:22 – TRIGACT[1:0]** Trigger Action
These bits define the trigger action used for a transfer.

| TRIGACT[1:0] | Name | Description |
|---|---|---|
| 0x0 | BLOCK | One trigger required for each block transfer |
| 0x1 | - | Reserved |

| TRIGACT[1:0] | Name | Description |
|---|---|---|
| 0x2 | BEAT | One trigger required for each beat transfer |
| 0x3 | TRANSACTION | One trigger required for each transaction |

**Bits 13:8 – TRIGSRC[5:0]** Trigger Source
These bits define the peripheral trigger which is source of the transfer. For details on trigger selection and trigger modes, refer to Transfer Triggers and Actions and CHCTRLB.TRIGACT.

| Value | Name | Description |
|---|---|---|
| 0x00 | DISABLE | Only software/event triggers |
| 0x01 | SERCOM0 RX | SERCOM0 RX Trigger |
| 0x02 | SERCOM0 TX | SERCOM0 TX Trigger |
| 0x03 | SERCOM1 RX | SERCOM1 RX Trigger |
| 0x04 | SERCOM1 TX | SERCOM1 TX Trigger |
| 0x05 | SERCOM2 RX | SERCOM2 RX Trigger |
| 0x06 | SERCOM2 TX | SERCOM2 TX Trigger |
| 0x07 | SERCOM3 RX | SERCOM3 RX Trigger |
| 0x08 | SERCOM3 TX | SERCOM3 TX Trigger |
| 0x09 | SERCOM4 RX | SERCOM4 RX Trigger |
| 0x0A | SERCOM4 TX | SERCOM4 TX Trigger |
| 0x0B | SERCOM5 RX | SERCOM5 RX Trigger |
| 0x0C | SERCOM5 TX | SERCOM5 TX Trigger |
| 0x0D | TCC0 OVF | TCC0 Overflow Trigger |
| 0x0E | TCC0 MC0 | TCC0 Match/Compare 0 Trigger |
| 0x0F | TCC0 MC1 | TCC0 Match/Compare 1 Trigger |
| 0x10 | TCC0 MC2 | TCC0 Match/Compare 2 Trigger |
| 0x11 | TCC0 MC3 | TCC0 Match/Compare 3 Trigger |
| 0x12 | TCC1 OVF | TCC1 Overflow Trigger |
| 0x13 | TCC1 MC0 | TCC1 Match/Compare 0 Trigger |
| 0x14 | TCC1 MC1 | TCC1 Match/Compare 1 Trigger |
| 0x15 | TCC2 OVF | TCC2 Overflow Trigger |
| 0x16 | TCC2 MC0 | TCC2 Match/Compare 0 Trigger |
| 0x17 | TCC2 MC1 | TCC2 Match/Compare 1 Trigger |
| 0x18 | TC3 OVF | TC3 Overflow Trigger |
| 0x19 | TC3 MC0 | TC3 Match/Compare 0 Trigger |
| 0x1A | TC3 MC1 | TC3 Match/Compare 1 Trigger |
| 0x1B | TC4 OVF | TC4 Overflow Trigger |
| 0x1C | TC4 MC0 | TC4 Match/Compare 0 Trigger |
| 0x1D | TC4 MC1 | TC4 Match/Compare 1 Trigger |
| 0x1E | TC5 OVF | TC5 Overflow Trigger |
| 0x1F | TC5 MC0 | TC5 Match/Compare 0 Trigger |
| 0x20 | TC5 MC1 | TC5 Match/Compare 1 Trigger |
| 0x21 | TC6 OVF | TC6 Overflow Trigger |
| 0x22 | TC6 MC0 | TC6 Match/Compare 0 Trigger |
| 0x23 | TC6 MC1 | TC6 Match/Compare 1 Trigger |
| 0x24 | TC7 OVF | TC7 Overflow Trigger |
| 0x25 | TC7 MC0 | TC7 Match/Compare 0 Trigger |

© 2018 Microchip Technology Inc.

| Value | Name | Description |
|---|---|---|
| 0x26 | TC7 MC1 | TC7 Match/Compare 1 Trigger |
| 0x27 | ADC RESRDY | ADC Result Ready Trigger |
| 0x28 | DAC EMPTY | DAC Empty Trigger |
| 0x29 | I2S RX 0 | I2S RX 0 Trigger |
| 0x2A | I2S RX 1 | I2S RX 1 Trigger |
| 0x2B | I2S TX 0 | I2S TX 0 Trigger |
| 0x2C | I2S TX 0 | I2S TX 1 Trigger |

**Bits 6:5 – LVL[1:0]** Channel Arbitration Level
These bits define the arbitration level used for the DMA channel, where a high level has priority over a low level. For further details on arbitration schemes, refer to 20.6.2.4 Arbitration.

These bits are not enable-protected.

| TRIGACT[1:0] | Name | Description |
|---|---|---|
| 0x0 | LVL0 | Channel Priority Level 0 |
| 0x1 | LVL1 | Channel Priority Level 1 |
| 0x2 | LVL2 | Channel Priority Level 2 |
| 0x3 | LVL3 | Channel Priority Level 3 |

**Bit 4 – EVOE** Channel Event Output Enable
This bit indicates if the Channel event generation is enabled. The event will be generated for every condition defined in the descriptor Event Output Selection (BTCTRL.EVOSEL).

This bit is available only for the least significant DMA channels. Refer to table: *User Multiplexer Selection* and *Event Generator Selection* of the Event System for details.

| Value | Description |
|---|---|
| 0 | Channel event generation is disabled. |
| 1 | Channel event generation is enabled. |

**Bit 3 – EVIE** Channel Event Input Enable
This bit is available only for the least significant DMA channels. Refer to table: *User Multiplexer Selection* and *Event Generator Selection* of the Event System for details.

| Value | Description |
|---|---|
| 0 | Channel event action will not be executed on any incoming event. |
| 1 | Channel event action will be executed on any incoming event. |

**Bits 2:0 – EVACT[2:0]** Event Input Action
These bits define the event input action, as shown below. The action is executed only if the corresponding EVIE bit in CHCTRLB register of the channel is set.

These bits are available only for the least significant DMA channels. Refer to table: *User Multiplexer Selection* and *Event Generator Selection* of the Event System for details.

| EVACT[2:0] | Name | Description |
|---|---|---|
| 0x0 | NOACT | No action |
| 0x1 | TRIG | Normal Transfer and Conditional Transfer on Strobe trigger |
| 0x2 | CTRIG | Conditional transfer trigger |
| 0x3 | CBLOCK | Conditional block transfer |
| 0x4 | SUSPEND | Channel suspend operation |
| 0x5 | RESUME | Channel resume operation |
| 0x6 | SSKIP | Skip next block suspend action |
| 0x7 | - | Reserved |

**Related Links**

24.8.3 USER
24.8.2 CHANNEL

### 20.8.20 Channel Interrupt Enable Clear

**Name:** CHINTENCLR
**Offset:** 0x4C
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Channel Interrupt Enable Set (CHINTENSET) register.
This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SUSP | TCMPL | TERR |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SUSP** Channel Suspend Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel Suspend Interrupt Enable bit, which disables the Channel Suspend interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Suspend interrupt is disabled. |
| 1 | The Channel Suspend interrupt is enabled. |

**Bit 1 – TCMPL** Channel Transfer Complete Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel Transfer Complete Interrupt Enable bit, which disables the Channel Transfer Complete interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Transfer Complete interrupt is disabled. When block action is set to none, the TCMPL flag will not be set when a block transfer is completed. |
| 1 | The Channel Transfer Complete interrupt is enabled. |

**Bit 0 – TERR** Channel Transfer Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel Transfer Error Interrupt Enable bit, which disables the Channel Transfer Error interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Transfer Error interrupt is disabled. |
| 1 | The Channel Transfer Error interrupt is enabled. |

### 20.8.21 Channel Interrupt Enable Set

**Name:** CHINTENSET
**Offset:** 0x4D
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Channel Interrupt Enable Clear (CHINTENCLR) register.
This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SUSP | TCMPL | TERR |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SUSP** Channel Suspend Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Channel Suspend Interrupt Enable bit, which enables the Channel Suspend interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Suspend interrupt is disabled. |
| 1 | The Channel Suspend interrupt is enabled. |

**Bit 1 – TCMPL** Channel Transfer Complete Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Channel Transfer Complete Interrupt Enable bit, which enables the Channel Transfer Complete interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Transfer Complete interrupt is disabled. |
| 1 | The Channel Transfer Complete interrupt is enabled. |

**Bit 0 – TERR** Channel Transfer Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Channel Transfer Error Interrupt Enable bit, which enables the Channel Transfer Error interrupt.

| Value | Description |
|---|---|
| 0 | The Channel Transfer Error interrupt is disabled. |
| 1 | The Channel Transfer Error interrupt is enabled. |

### 20.8.22 Channel Interrupt Flag Status and Clear

**Name:** CHINTFLAG
**Offset:** 0x4E
**Reset:** 0x00
**Property:** -

This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SUSP | TCMPL | TERR |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SUSP** Channel Suspend
This flag is cleared by writing a '1' to it.

This flag is set when a block transfer with suspend block action is completed, when a software suspend command is executed, when a suspend event is received or when an invalid descriptor is fetched by the DMA.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Channel Suspend interrupt flag for the corresponding channel.

For details on available software commands, refer to CHCTRLB.CMD.

For details on available event input actions, refer to CHCTRLB.EVACT.

For details on available block actions, refer to BTCTRL.BLOCKACT.

**Bit 1 – TCMPL** Channel Transfer Complete
This flag is cleared by writing a '1' to it.

This flag is set when a block transfer is completed and the corresponding interrupt block action is enabled.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transfer Complete interrupt flag for the corresponding channel.

**Bit 0 – TERR** Channel Transfer Error
This flag is cleared by writing a '1' to it.

This flag is set when a bus error is detected during a beat transfer or when the DMAC fetches an invalid descriptor.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transfer Error interrupt flag for the corresponding channel.

### 20.8.23 Channel Status

**Name:** CHSTATUS
**Offset:** 0x4F
**Reset:** 0x00
**Property:** -

This register affects the DMA channel that is selected in the Channel ID register (CHID.ID).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FERR | BUSY | PEND |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – FERR** Channel Fetch Error
This bit is cleared when a software resume command is executed.

This bit is set when an invalid descriptor is fetched.

**Bit 1 – BUSY** Channel Busy
This bit is cleared when the channel trigger action is completed, when a bus error is detected or when the channel is disabled.

This bit is set when the DMA channel starts a DMA transfer.

**Bit 0 – PEND** Channel Pending
This bit is cleared when the channel trigger action is started, when a bus error is detected or when the channel is disabled. For details on trigger action settings, refer to CHCTRLB.TRIGACT.

This bit is set when a transfer is pending on the DMA channel, as soon as the transfer request is received.

## 20.9 Register Summary - SRAM

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | BTCTRL | 7:0 | | | | BLOCKACT[1:0] | | EVOSEL[1:0] | | VALID |
| | | 15:8 | STEPSIZE[2:0] | | | STEPSEL | DSTINC | SRCINC | BEATSIZE[1:0] | |
| 0x02 | BTCNT | 7:0 | BTCNT[7:0] | | | | | | | |
| | | 15:8 | BTCNT[15:8] | | | | | | | |
| 0x04 | SRCADDR | 7:0 | SRCADDR[7:0] | | | | | | | |
| | | 15:8 | SRCADDR[15:8] | | | | | | | |
| | | 23:16 | SRCADDR[23:16] | | | | | | | |
| | | 31:24 | SRCADDR[31:24] | | | | | | | |
| 0x08 | DSTADDR | 7:0 | DSTADDR[7:0] | | | | | | | |
| | | 15:8 | DSTADDR[15:8] | | | | | | | |
| | | 23:16 | DSTADDR[23:16] | | | | | | | |
| | | 31:24 | DSTADDR[31:24] | | | | | | | |
| 0x0C | DESCADDR | 7:0 | DESCADDR[7:0] | | | | | | | |
| | | 15:8 | DESCADDR[15:8] | | | | | | | |
| | | 23:16 | DESCADDR[23:16] | | | | | | | |
| | | 31:24 | DESCADDR[31:24] | | | | | | | |

## 20.10 Register Description - SRAM

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 20.5.8 Register Access Protection.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 20.10.1 Block Transfer Control

**Name:** BTCTRL
**Offset:** 0x00
**Property:** -

The BTCTRL register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | STEPSIZE[2:0] | | | STEPSEL | DSTINC | SRCINC | BEATSIZE[1:0] | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | BLOCKACT[1:0] | | EVOSEL[1:0] | | VALID |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 15:13 – STEPSIZE[2:0]** Address Increment Step Size
These bits select the address increment step size. The setting apply to source or destination address, depending on STEPSEL setting.

| Value | Name | Description |
|---|---|---|
| 0x0 | X1 | Next ADDR = ADDR + (Beat size in byte) * 1 |
| 0x1 | X2 | Next ADDR = ADDR + (Beat size in byte) * 2 |
| 0x2 | X4 | Next ADDR = ADDR + (Beat size in byte) * 4 |
| 0x3 | X8 | Next ADDR = ADDR + (Beat size in byte) * 8 |
| 0x4 | X16 | Next ADDR = ADDR + (Beat size in byte) * 16 |
| 0x5 | X32 | Next ADDR = ADDR + (Beat size in byte) * 32 |
| 0x6 | X64 | Next ADDR = ADDR + (Beat size in byte) * 64 |
| 0x7 | X128 | Next ADDR = ADDR + (Beat size in byte) * 128 |

**Bit 12 – STEPSEL** Step Selection
This bit selects if source or destination addresses are using the step size settings.

| Value | Name | Description |
|---|---|---|
| 0x0 | DST | Step size settings apply to the destination address |
| 0x1 | SRC | Step size settings apply to the source address |

**Bit 11 – DSTINC** Destination Address Increment Enable
Writing a '0' to this bit will disable the destination address incrementation. The address will be kept fixed during the data transfer.

Writing a '1' to this bit will enable the destination address incrementation. By default, the destination address is incremented by 1. If the STEPSEL bit is cleared, flexible step-size settings are available in the STEPSIZE register.

| Value | Description |
|---|---|
| 0 | The Destination Address Increment is disabled. |
| 1 | The Destination Address Increment is enabled. |

**Bit 10 – SRCINC** Source Address Increment Enable
Writing a '0' to this bit will disable the source address incrementation. The address will be kept fixed during the data transfer.

Writing a '1' to this bit will enable the source address incrementation. By default, the source address is incremented by 1. If the STEPSEL bit is set, flexible step-size settings are available in the STEPSIZE register.

| Value | Description |
|---|---|
| 0 | The Source Address Increment is disabled. |
| 1 | The Source Address Increment is enabled. |

**Bits 9:8 – BEATSIZE[1:0]** Beat Size
These bits define the size of one beat. A beat is the size of one data transfer bus access, and the setting apply to both read and write accesses.

| Value | Name | Description |
|---|---|---|
| 0x0 | BYTE | 8-bit bus transfer |
| 0x1 | HWORD | 16-bit bus transfer |
| 0x2 | WORD | 32-bit bus transfer |
| other | | Reserved |

**Bits 4:3 – BLOCKACT[1:0]** Block Action
These bits define what actions the DMAC should take after a block transfer has completed.

| BLOCKACT[1:0] | Name | Description |
|---|---|---|
| 0x0 | NOACT | Channel will be disabled if it is the last block transfer in the transaction |
| 0x1 | INT | Channel will be disabled if it is the last block transfer in the transaction and block interrupt |
| 0x2 | SUSPEND | Channel suspend operation is completed |
| 0x3 | BOTH | Both channel suspend operation and block interrupt |

**Bits 2:1 – EVOSEL[1:0]** Event Output Selection
These bits define the event output selection.

| EVOSEL[1:0] | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Event generation disabled |
| 0x1 | BLOCK | Event strobe when block transfer complete |
| 0x2 | | Reserved |
| 0x3 | BEAT | Event strobe when beat transfer complete |

**Bit 0 – VALID** Descriptor Valid
Writing a '0' to this bit in the Descriptor or Write-Back memory will suspend the DMA channel operation when fetching the corresponding descriptor.

The bit is automatically cleared in the Write-Back memory section when channel is aborted, when an error is detected during the block transfer, or when the block transfer is completed.

| Value | Description |
|---|---|
| 0 | The descriptor is not valid. |
| 1 | The descriptor is valid. |

### 20.10.2 Block Transfer Count

**Name:** BTCNT
**Offset:** 0x02
**Property:** -

The BTCNT register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BTCNT[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BTCNT[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 15:0 – BTCNT[15:0]** Block Transfer Count
This bit group holds the 16-bit block transfer count.

During a transfer, the internal counter value is decremented by one after each beat transfer. The internal counter is written to the corresponding write-back memory section for the DMA channel when the DMA channel loses priority, is suspended or gets disabled. The DMA channel can be disabled by a complete transfer, a transfer error or by software.

### 20.10.3 Block Transfer Source Address

**Name:** SRCADDR
**Offset:** 0x04
**Property:** -

The SRCADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SRCADDR[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SRCADDR[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SRCADDR[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SRCADDR[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:0 – SRCADDR[31:0]** Transfer Source Address
This bit group holds the source address corresponding to the last beat transfer address in the block transfer.

### 20.10.4 Block Transfer Destination Address

**Name:** DSTADDR
**Offset:** 0x08
**Property:** -

The DSTADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DSTADDR[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DSTADDR[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DSTADDR[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DSTADDR[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:0 – DSTADDR[31:0]** Transfer Destination Address
This bit group holds the destination address corresponding to the last beat transfer address in the block transfer.

### 20.10.5 Next Descriptor Address

**Name:** DESCADDR
**Offset:** 0x0C
**Property:** -

The DESCADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DESCADDR[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DESCADDR[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DESCADDR[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DESCADDR[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:0 – DESCADDR[31:0]** Next Descriptor Address
This bit group holds the SRAM address of the next descriptor. The value must be 128-bit aligned. If the value of this SRAM register is 0x00000000, the transaction will be terminated when the DMAC tries to load the next transfer descriptor.

# 21. EIC – External Interrupt Controller

## 21.1 Overview

The External Interrupt Controller (EIC) allows external pins to be configured as interrupt lines. Each interrupt line can be individually masked and can generate an interrupt on rising, falling, or both edges, or on high or low levels. Each external pin has a configurable filter to remove spikes. Each external pin can also be configured to be asynchronous in order to wake up the device from sleep modes where all clocks have been disabled. External pins can also generate an event.

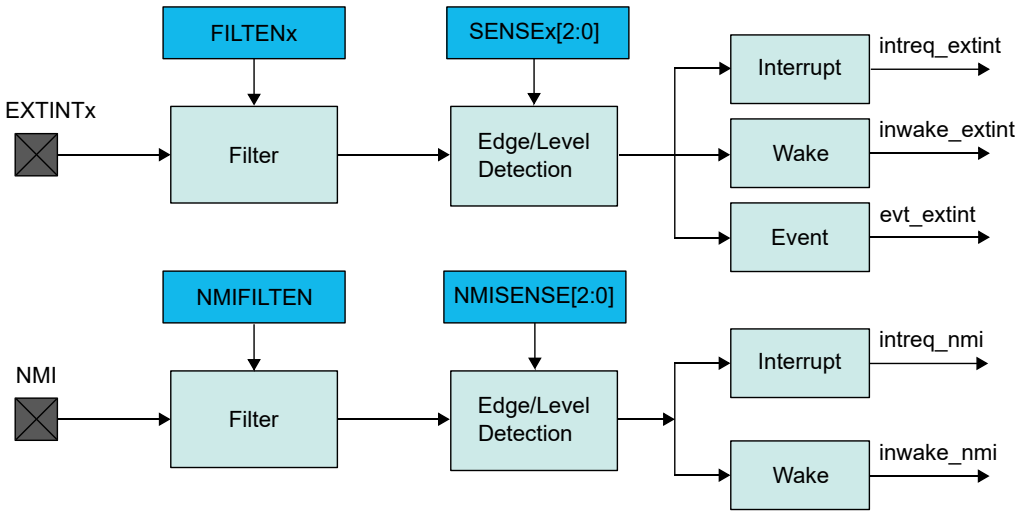
A separate non-maskable interrupt (NMI) is also supported. It has properties similar to the other external interrupts, but is connected to the NMI request of the CPU, enabling it to interrupt any other interrupt mode.

## 21.2 Features

- Up to 16 external pins (EXTINTx), plus one non-maskable pin (NMI)
- Dedicated, individually maskable interrupt for each pin
- Interrupt on rising, falling, or both edges
- Interrupt on high or low levels
- Asynchronous interrupts for sleep modes without clock
- Filtering of external pins
- Event generation from EXTINTx

## 21.3 Block Diagram

**Figure 21-1. EIC Block Diagram**



## 21.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| EXTINT[15..0] | Digital Input | External interrupt pin |
| NMI | Digital Input | Non-maskable interrupt pin |

One signal may be available on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 21.5 Product Dependencies

In order to use this EIC, other parts of the system must be configured correctly, as described below.

### 21.5.1 I/O Lines

Using the EIC's I/O lines requires the I/O pins to be configured.

**Related Links**

23. PORT - I/O Pin Controller

### 21.5.2 Power Management

All interrupts are available down to STANDBY sleep mode, but the EIC can be configured to automatically mask some interrupts in order to prevent device wake-up.

The EIC will continue to operate in any sleep mode where the selected source clock is running. The EIC's interrupts can be used to wake up the device from sleep modes. Events connected to the Event System can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 21.5.3 Clocks

The EIC bus clock (CLK_EIC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_EIC_APB can be found in the *Peripheral Clock Masking* section in PM – Power Manager.

A generic clock (GCLK_EIC) is required to clock the peripheral. This clock must be configured and enabled in the Generic Clock Controller before using the peripheral. Refer to *GCLK – Generic Clock Controller*.

This generic clock is asynchronous to the user interface clock (CLK_EIC_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to 21.6.9 Synchronization for further details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

15. GCLK - Generic Clock Controller

### 21.5.4 DMA

Not applicable.

### 21.5.5 Interrupts

There are several interrupt request lines, at least one for the external interrupts (EXTINT) and one for non-maskable interrupt (NMI).

The EXTINT interrupt request line is connected to the interrupt controller. Using the EIC interrupt requires the interrupt controller to be configured first.

The NMI interrupt request line is also connected to the interrupt controller, but does not require the interrupt to be configured.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 21.5.6 Events

The events are connected to the Event System. Using the events requires the Event System to be configured first.

**Related Links**

24. EVSYS – Event System

### 21.5.7 Debug Operation

When the CPU is halted in debug mode, the EIC continues normal operation. If the EIC is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

### 21.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except for the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Non-Maskable Interrupt Flag Status and Clear register (NMIFLAG)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 21.5.9 Analog Connections

Not applicable.

## 21.6 Functional Description

### 21.6.1 Principle of Operation

The EIC detects edge or level condition to generate interrupts to the CPU interrupt controller or events to the Event System. Each external interrupt pin (EXTINT) can be filtered using majority vote filtering, clocked by GCLK_EIC

### 21.6.2 Basic Operation

#### 21.6.2.1 Initialization

The EIC must be initialized in the following order:

1. Enable CLK_EIC_APB
2. If edge detection or filtering is required, GCLK_EIC must be enabled
3. Write the EIC configuration registers (EVCTRL, WAKEUP, CONFIGy)
4. Enable the EIC

To use NMI, GCLK_EIC must be enabled after EIC configuration (NMICTRL).

#### 21.6.2.2 Enabling, Disabling and Resetting

The EIC is enabled by writing a '1' the Enable bit in the Control register (CTRL.ENABLE). The EIC is disabled by writing CTRL.ENABLE to '0'.

The EIC is reset by setting the Software Reset bit in the Control register (CTRL.SWRST). All registers in the EIC will be reset to their initial state, and the EIC will be disabled.

Refer to the CTRL register description for details.

### 21.6.3 External Pin Processing

Each external pin can be configured to generate an interrupt/event on edge detection (rising, falling or both edges) or level detection (high or low). The sense of external interrupt pins is configured by writing the Input Sense x bits in the Config n register (CONFIGn.SENSEx). The corresponding interrupt flag (INTFLAG.EXTINT[x]) in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition is met.

When the interrupt flag has been cleared in edge-sensitive mode, INTFLAG.EXTINT[x] will only be set if a new interrupt condition is met. In level-sensitive mode, when interrupt has been cleared, INTFLAG.EXTINT[x] will be set immediately if the EXTINTx pin still matches the interrupt condition.

Each external pin can be filtered by a majority vote filtering, clocked by GCLK_EIC. Filtering is enabled if bit Filter Enable x in the Configuration n register (CONFIGn.FILTENx) is written to '1'. The majority vote filter samples the external pin three times with GCLK_EIC and outputs the value when two or more samples are equal.

**Table 21-1. Majority Vote Filter**

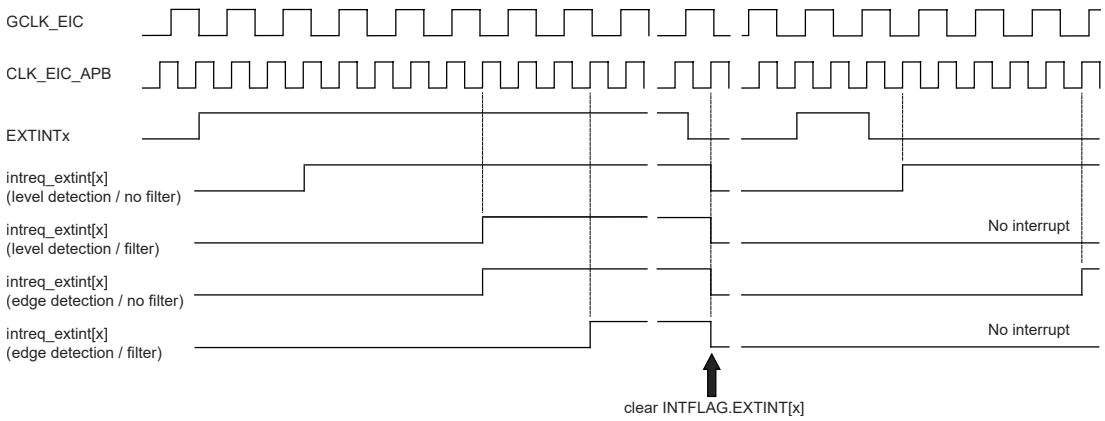
| Samples [0, 1, 2] | Filter Output |
|---|---|
| [0,0,0] | 0 |
| [0,0,1] | 0 |
| [0,1,0] | 0 |
| [0,1,1] | 1 |
| [1,0,0] | 0 |
| [1,0,1] | 1 |
| [1,1,0] | 1 |
| [1,1,1] | 1 |

When an external interrupt is configured for level detection, or if filtering is disabled, detection is made asynchronously, and GCLK_EIC is not required.

If filtering or edge detection is enabled, the EIC automatically requests the GCLK_EIC to operate (GCLK_EIC must be enabled in the GCLK module, see *GCLK – Generic Clock Controller* for details). If level detection is enabled, GCLK_EIC is not required, but interrupt and events can still be generated.

When an external interrupt is configured for level detection and when filtering is disabled, detection is done asynchronously. Asynchronuous detection does not require GCLK_EIC, but interrupt and events can still be generated. If filtering or edge detection is enabled, the EIC automatically requests GCLK_EIC to operate. GCLK_EIC must be enabled in the GCLK module.

**Figure 21-2. Interrupt Detections**




The detection delay depends on the detection mode.

**Table 21-2. Interrupt Latency**

| Detection mode | Latency (worst case) |
|---|---|
| Level without filter | Three CLK_EIC_APB periods |
| Level with filter | Four GCLK_EIC periods + Three CLK_EIC_APB periods |
| Edge without filter | Four GCLK_EIC periods + Three CLK_EIC_APB periods |
| Edge with filter | Six GCLK_EIC periods + Three CLK_EIC_APB periods |

**Related Links**

15. GCLK - Generic Clock Controller

### 21.6.4 Additional Features

#### 21.6.4.1 Non-Maskable Interrupt (NMI)

The non-maskable interrupt pin can also generate an interrupt on edge or level detection, but it is configured with the dedicated NMI Control register (NMICTRL). To select the sense for NMI, write to the NMISENSE bit group in the NMI Control register (NMICTRL.NMISENSE). NMI filtering is enabled by writing a '1' to the NMI Filter Enable bit (NMICTRL.NMIFILTEN).

If edge detection or filtering is required, enable GCLK_EIC or CLK_ULP32K.

NMI detection is enabled only by the NMICTRL.NMISENSE value, and the EIC is not required to be enabled.

When an NMI is detected, the non-maskable interrupt flag in the NMI Flag Status and Clear register is set (NMIFLAG.NMI). NMI interrupt generation is always enabled, and NMIFLAG.NMI generates an interrupt request when set.

### 21.6.5 DMA Operation

Not applicable.

### 21.6.6 Interrupts

The EIC has the following interrupt sources:

- External interrupt pins (EXTINTx). See 21.6.2 Basic Operation.
- Non-maskable interrupt pin (NMI). See 21.6.4 Additional Features.

Each interrupt source has an associated interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when an interrupt condition occurs (NMIFLAG for NMI). Each interrupt, except NMI, can be individually enabled by setting the corresponding bit in the Interrupt Enable Set register (INTENSET=1), and disabled by setting the corresponding bit in the Interrupt Enable Clear register (INTENCLR=1).

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the EIC is reset. See the INTFLAG register for details on how to clear interrupt flags. The EIC has one interrupt request line for each external interrupt (EXTINTx) and one line for NMI. The user must read the INTFLAG (or NMIFLAG) register to determine which interrupt condition is present.

**Note:**

1. Interrupts must be globally enabled for interrupt requests to be generated.
2. If an external interrupts (EXTINT) is common on two or more I/O pins, only one will be active (the first one programmed).

**Related Links**

7.1 Multiplexed Signals

11. Processor And Architecture

### 21.6.7 Events

The EIC can generate the following output events:

- External event from pin (EXTINTx).

Setting an Event Output Control register (EVCTRL.EXTINTEO) enables the corresponding output event. Clearing this bit disables the corresponding output event. Refer to *Event System* for details on configuring the Event System.

When the condition on pin EXTINTx matches the configuration in the CONFIGn register, the corresponding event is generated, if enabled.

**Related Links**

24. EVSYS – Event System

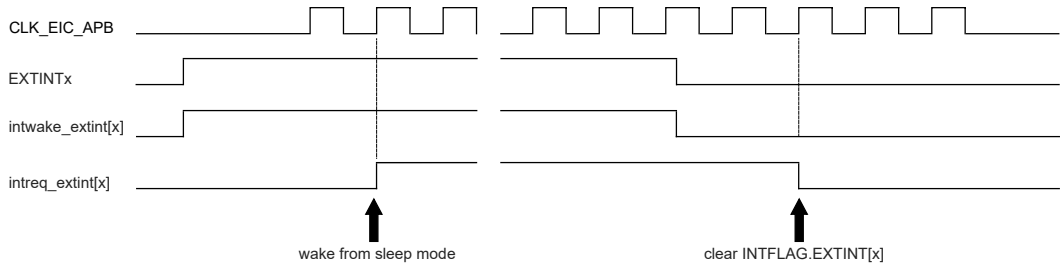
### 21.6.8 Sleep Mode Operation

In sleep modes, an EXTINTx pin can wake up the device if the corresponding condition matches the configuration in CONFIGy register. Writing a one to a Wake-Up Enable bit (WAKEUP.WAKEUPEN[x]) enables the wake-up from pin EXTINTx. Writing a zero to a Wake-Up Enable bit (WAKEUP.WAKEUPEN[x]) disables the wake-up from pin EXTINTx.

Using WAKEUPEN[x]=1 with INTENSET=0 is not recommended.

In sleep modes, an EXTINTx pin can wake up the device if the corresponding condition matches the configuration in CONFIGn register, and the corresponding bit in the Interrupt Enable Set register (INTENSET) is written to '1'. WAKEUP.WAKEUPEN[x]=1 can enable the wake-up from pin EXTINTx.

**Figure 21-3. Wake-Up Operation Example (High-Level Detection, No Filter, WAKEUPEN[x]=1)**




### 21.6.9 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled, and interrupts will be pending as long as the bus is stalled.

The following bits are synchronized when written:

- Software Reset bit in the Control register (CTRL.SWRST)
- Enable bit in the Control register (CTRL.ENABLE)

**Related Links**

14.3 Register Synchronization

## 21.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | | | | ENABLE | SWRST |
| 0x01 | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x02 | NMICTRL | 7:0 | | | | | NMIFILTEN | NMISENSE[2:0] | | |
| 0x03 | NMIFLAG | 7:0 | | | | | | | | NMI |
| 0x04 | EVCTRL | 7:0 | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx |
| | | 15:8 | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx |
| | | 23:16 | | | | | | | EXTINTEOx | EXTINTEOx |
| | | 31:24 | | | | | | | | |
| 0x08 | INTENCLR | 7:0 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 15:8 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 23:16 | | | | | | | EXTINTx | EXTINTx |
| | | 31:24 | | | | | | | | |
| 0x0C | INTENSET | 7:0 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 15:8 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 23:16 | | | | | | | EXTINTx | EXTINTx |
| | | 31:24 | | | | | | | | |
| 0x10 | INTFLAG | 7:0 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 15:8 | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| | | 23:16 | | | | | | | EXTINTx | EXTINTx |
| | | 31:24 | | | | | | | | |
| 0x14 | WAKEUP | 7:0 | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx |
| | | 15:8 | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx |
| | | 23:16 | | | | | | | WAKEUPENx | WAKEUPENx |
| | | 31:24 | | | | | | | | |
| 0x18 | CONFIG | 7:0 | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| | | 15:8 | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| | | 23:16 | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| | | 31:24 | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |

## 21.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 21.8.1 Control

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ENABLE | SWRST |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

| Value | Description |
|---|---|
| 0 | The EIC is disabled. |
| 1 | The EIC is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the EIC to their initial state, and the EIC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.

Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no ongoing reset operation. |
| 1 | The reset operation is ongoing. |

### 21.8.2 Status

**Name:** STATUS
**Offset:** 0x01
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 21.8.3 Non-Maskable Interrupt Control

**Name:** NMICTRL
**Offset:** 0x02
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NMIFILTEN | NMISENSE[2:0] | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – NMIFILTEN** Non-Maskable Interrupt Filter Enable

| Value | Description |
|---|---|
| 0 | NMI filter is disabled. |
| 1 | NMI filter is enabled. |

**Bits 2:0 – NMISENSE[2:0]** Non-Maskable Interrupt Sense
These bits define on which edge or level the NMI triggers.

| NMISENSE[2:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No detection |
| 0x1 | RISE | Rising-edge detection |
| 0x2 | FALL | Falling-edge detection |
| 0x3 | BOTH | Both-edges detection |
| 0x4 | HIGH | High-level detection |
| 0x5 | LOW | Low-level detection |
| 0x6-0x7 | | Reserved |

### 21.8.4 Non-Maskable Interrupt Flag Status and Clear

**Name:** NMIFLAG
**Offset:** 0x03
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NMI |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – NMI** Non-Maskable Interrupt
This flag is cleared by writing a one to it.

This flag is set when the NMI pin matches the NMI sense configuration, and will generate an interrupt request.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the non-maskable interrupt flag.

### 21.8.5 Event Control

**Name:** EVCTRL
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EXTINTEOx | EXTINTEOx |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx | EXTINTEOx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – EXTINTEOx** External Interrupt x Event Output Enable [x=17..0]
These bits indicate whether the event associated with the EXTINTx pin is enabled or not to generated for every detection.

| Value | Description |
|---|---|
| 0 | Event from pin EXTINTx is disabled. |
| 1 | Event from pin EXTINTx is enabled. |

### 21.8.6 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EXTINTx | EXTINTx |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – EXTINTx** External Interrupt x Enable [x=17..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the External Interrupt x Enable bit, which enables the external interrupt.

| Value | Description |
|---|---|
| 0 | The external interrupt x is disabled. |
| 1 | The external interrupt x is enabled. |

### 21.8.7 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EXTINTx | EXTINTx |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – EXTINTx** External Interrupt x Enable [x=17..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will set the External Interrupt x Enable bit, which enables the external interrupt.

| Value | Description |
|---|---|
| 0 | The external interrupt x is disabled. |
| 1 | The external interrupt x is enabled. |

### 21.8.8 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EXTINTx | EXTINTx |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx | EXTINTx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – EXTINTx** External Interrupt x [x=17..0]
This flag is cleared by writing a one to it.

This flag is set when EXTINTx pin matches the external interrupt sense configuration and will generate an interrupt request if INTENCLR/SET.EXTINT[x] is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the External Interrupt x flag.

### 21.8.9 Wake-Up Enable

**Name:** WAKEUP
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WAKEUPENx | WAKEUPENx |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx | WAKEUPENx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – WAKEUPENx** External Interrupt x Wake-up Enable [x=17..0]
This bit enables or disables wake-up from sleep modes when the EXTINTx pin matches the external interrupt sense configuration.

| Value | Description |
|---|---|
| 0 | Wake-up from the EXTINTx pin is disabled. |
| 1 | Wake-up from the EXTINTx pin is enabled. |

### 21.8.10 Configuration n

**Name:** CONFIG
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FILTENx | SENSEx[2:0] | | | FILTENx | SENSEx[2:0] | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31, 27, 23, 19, 15, 11, 7,3 – FILTENx** Filter 0 Enable [x=7..0]

| | |
|---|---|
| 0: | Filter is disabled for EXTINT[n*8+x] input. |
| 1: | Filter is enabled for EXTINT[n*8+x] input. |

**Bits 30:28, 26:24, 22:20, 18:16, 14:12, 10:8, 6:4, 2:0 – SENSEx** Input Sense 0 Configuration [x=7..0]

| SENSE0[2:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No detection |
| 0x1 | RISE | Rising-edge detection |
| 0x2 | FALL | Falling-edge detection |
| 0x3 | BOTH | Both-edges detection |
| 0x4 | HIGH | High-level detection |
| 0x5 | LOW | Low-level detection |
| 0x6-0x7 | | Reserved |

# 22. NVMCTRL – Nonvolatile Memory Controller

## 22.1 Overview

Nonvolatile Memory (NVM) is a reprogrammable Flash memory that retains program and data storage even with power off. It embeds a main array and a separate smaller array intended for EEPROM emulation (RWWEE) that can be programmed while reading the main array. The NVM Controller (NVMCTRL) connects to the AHB and APB bus interfaces for system access to the NVM block. The AHB interface is used for reads and writes to the NVM block, while the APB interface is used for commands and configuration.

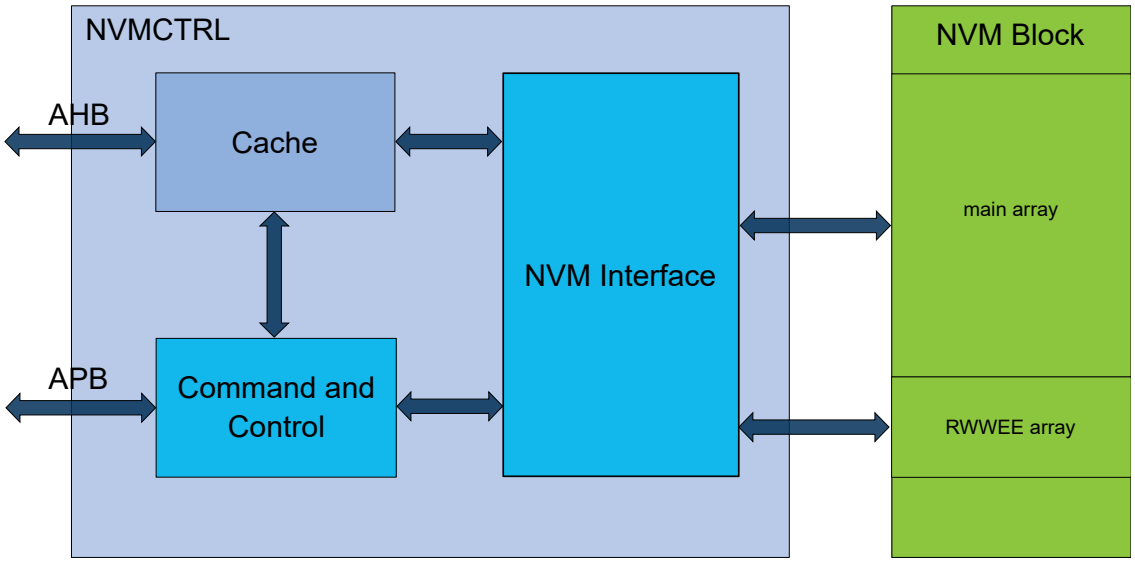
## 22.2 Features

- 32-bit AHB interface for reads and writes
- Read-While-Write DATA Flash
- All NVM sections are memory mapped to the AHB, including calibration and system configuration
- 32-bit APB interface for commands and control
- Programmable wait states for read optimization
- 16 regions can be individually protected or unprotected
- Additional protection for bootloader
- Supports device protection through a security bit
- Interface to Power Manager for power-down of Flash blocks in sleep modes
- Can optionally wake up on exit from sleep or on first access
- Direct-mapped cache

**Note:** A register with property "Enable-Protected" may contain bits that are *not* enable-protected.

## 22.3 Block Diagram

**Figure 22-1. Block Diagram**



## 22.4 Signal Description

Not applicable.

## 22.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described in the following sections.

### 22.5.1 Power Management

The NVMCTRL will continue to operate in any sleep mode where the selected source clock is running. The NVMCTRL interrupts can be used to wake up the device from sleep modes.

The Power Manager will automatically put the NVM block into a low-power state when entering sleep mode. This is based on the Control B register (CTRLB) SLEEPPRM bit setting. Refer to the 22.8.2 CTRLB. SLEEPPRM register description for more details. The NVM block goes into low-power mode automatically when the device enters STANDBY mode regardless of SLEEPPRM. The NVM Page Buffer is lost when the NVM goes into low power mode therefore a write command must be issued prior entering the NVM low power mode. NVMCTRL SLEEPPRM can be disabled to avoid such loss when the CPU goes into sleep except if the device goes into STANDBY mode for which there is no way to retain the Page Buffer.

**Related Links**

16. PM – Power Manager

### 22.5.2 Clocks

Two synchronous clocks are used by the NVMCTRL. One is provided by the AHB bus (CLK_NVMCTRL_AHB) and the other is provided by the APB bus (CLK_NVMCTRL_APB). For higher system frequencies, a programmable number of wait states can be used to optimize performance. When changing the AHB bus frequency, the user must ensure that the NVM Controller is configured with the proper number of wait states. Refer to the Electrical Characteristics for the exact number of wait states to be used for a particular frequency range.

**Related Links**

37. Electrical Characteristics

### 22.5.3 Interrupts

The NVM Controller interrupt request line is connected to the interrupt controller. Using the NVMCTRL interrupt requires the interrupt controller to be programmed first.

### 22.5.4 Debug Operation

When an external debugger forces the CPU into debug mode, the peripheral continues normal operation.

Access to the NVM block can be protected by the security bit. In this case, the NVM block will not be accessible. See the section on the NVMCTRL 22.6.6 Security Bit for details.

### 22.5.5 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)

- Status register (STATUS)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 22.5.6 Analog Connections

Not applicable.

## 22.6 Functional Description

### 22.6.1 Principle of Operation

The NVM Controller is a slave on the AHB and APB buses. It responds to commands, read requests and write requests, based on user configuration.

#### 22.6.1.1 Initialization

After power up, the NVM Controller goes through a power-up sequence. During this time, access to the NVM Controller from the AHB bus is halted. Upon power-up completion, the NVM Controller is operational without any need for user configuration.

### 22.6.2 Memory Organization

Refer to the Physical Memory Map for memory sizes and addresses for each device.

The NVM is organized into rows, where each row contains four pages, as shown in the NVM Row Organization figure. The NVM has a row-erase granularity, while the write granularity is by page. In other words, a single row erase will erase all four pages in the row, while four write operations are used to write the complete row.

**Figure 22-2. NVM Row Organization**

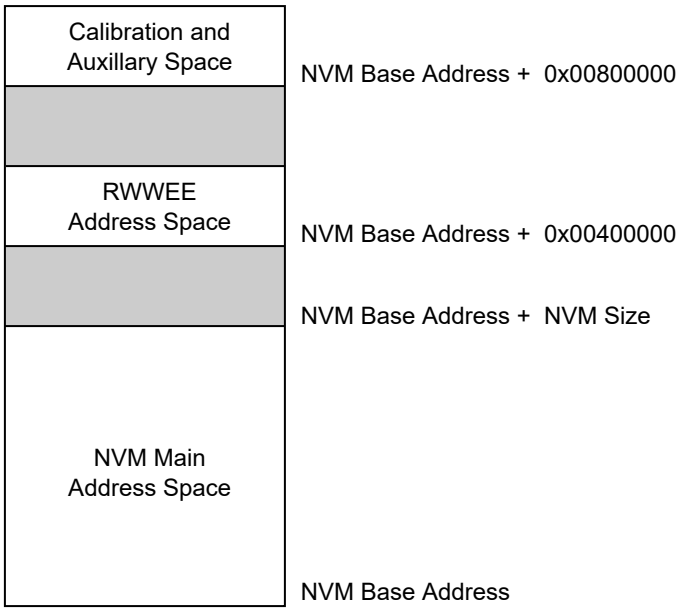
| | | | | |
|---|---|---|---|---|
| Row n | Page (n*4) + 3 | Page (n*4) + 2 | Page (n*4) + 1 | Page (n*4) + 0 |

The NVM block contains a calibration and auxiliary space plus a dedicated EEPROM emulation space that are memory mapped. Refer to the NVM Organization figure below for details.

The calibration and auxiliary space contains factory calibration and system configuration information. These spaces can be read from the AHB bus in the same way as the main NVM main address space.

In addition, a boot loader section can be allocated at the beginning of the main array, and an EEPROM section can be allocated at the end of the NVM main address space.

**Figure 22-3. NVM Memory Organization**

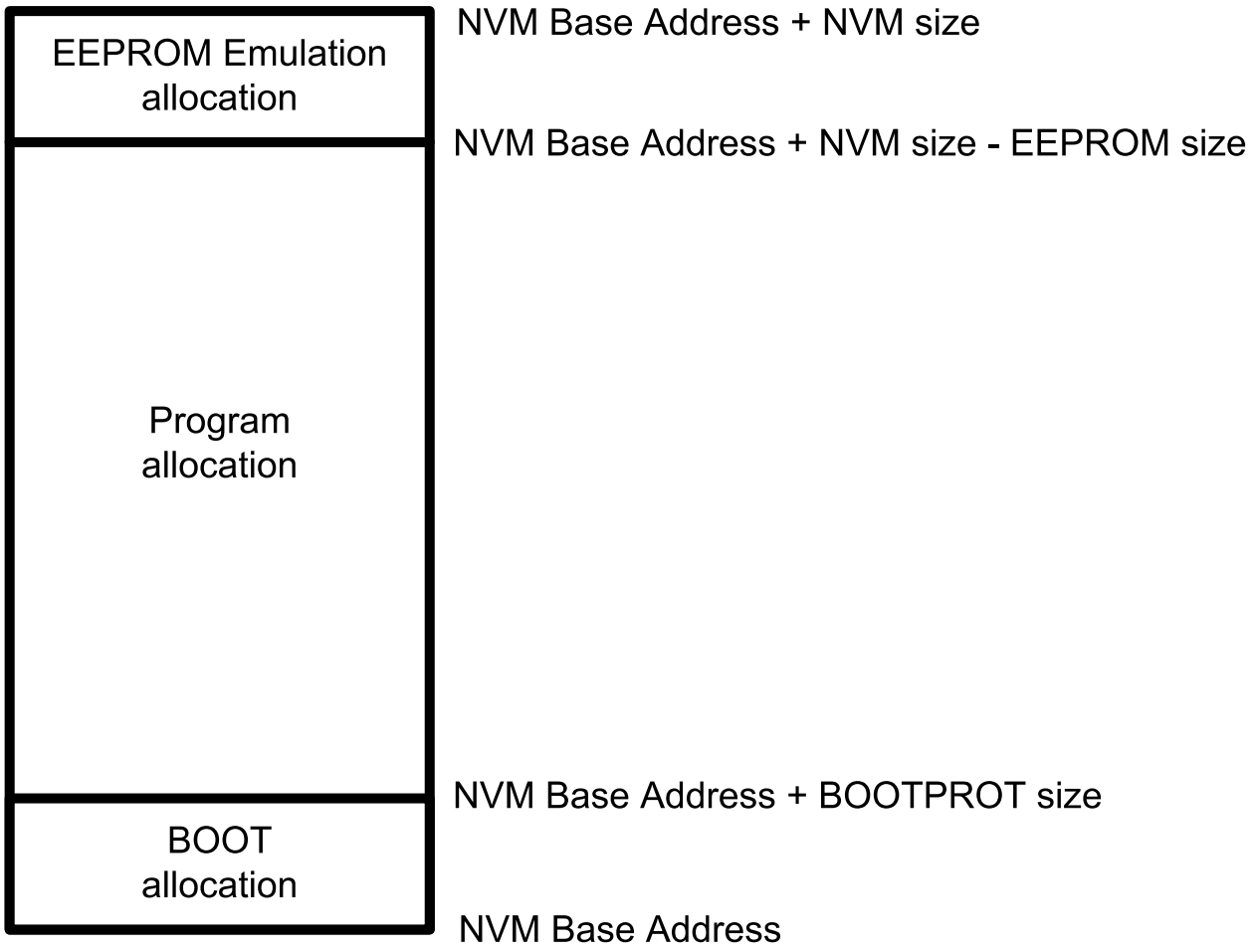



The lower rows in the NVM main address space can be allocated as a boot loader section by using the BOOTPROT fuses, and the upper rows can be allocated to EEPROM, as shown in the figure below.

The boot loader section is protected by the lock bit(s) corresponding to this address space and by the BOOTPROT[2:0] fuse. The EEPROM rows can be written regardless of the region lock status.

The number of rows protected by BOOTPROT is given in Boot Loader Size, the number of rows allocated to the EEPROM are given in EEPROM Size.

**Figure 22-4. EEPROM and Boot Loader Allocation**




**Related Links**

10.2 Physical Memory Map

### 22.6.3 Region Lock Bits

The NVM block is grouped into 16 equally sized regions. The region size is dependent on the Flash memory size, and is given in the table below. Each region has a dedicated lock bit preventing writing and erasing pages in the region. After production, all regions will be unlocked.

**Table 22-1. Region Size**

| Memory Size [KB] | Region Size [KB] |
|---|---|
| 256 | 16 |
| 128 | 8 |
| 64 | 4 |
| 32 | 2 |

To lock or unlock a region, the Lock Region and Unlock Region commands are provided. Writing one of these commands will temporarily lock/unlock the region containing the address loaded in the ADDR register. ADDR can be written by software, or the automatically loaded value from a write operation can be used. The new setting will stay in effect until the next Reset, or until the setting is changed again using

the Lock and Unlock commands. The current status of the lock can be determined by reading the LOCK register.

To change the default lock/unlock setting for a region, the user configuration section of the auxiliary space must be written using the Write Auxiliary Page command. Writing to the auxiliary space will take effect after the next Reset. Therefore, a boot of the device is needed for changes in the lock/unlock setting to take effect. Refer to the Physical Memory Map for calibration and auxiliary space address mapping.

**Related Links**

10.2 Physical Memory Map

### 22.6.4 Command and Data Interface

The NVM Controller is addressable from the APB bus, while the NVM main address space is addressable from the AHB bus. Read and automatic page write operations are performed by addressing the NVM main address space or the RWWEE address space directly, while other operations such as manual page writes and row erases must be performed by issuing commands through the NVM Controller.

To issue a command, the CTRLA.CMD bits must be written along with the CTRLA.CMDEX value. When a command is issued, INTFLAG.READY will be cleared until the command has completed. Any commands written while INTFLAG.READY is low will be ignored.

The CTRLB register must be used to control the power reduction mode, read wait states, and the write mode.

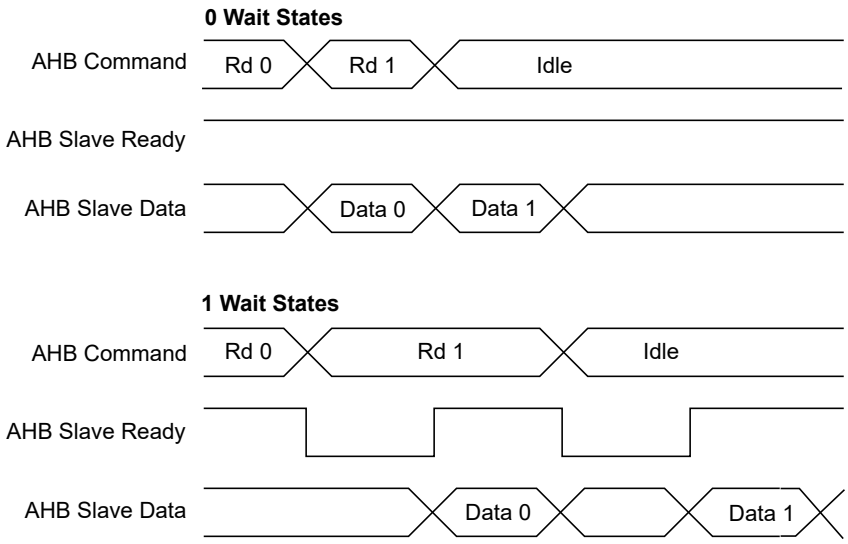
#### 22.6.4.1 NVM Read

Reading from the NVM main address space is performed via the AHB bus by addressing the NVM main address space or auxiliary address space directly. Read data is available after the configured number of read wait states (CTRLB.RWS) set in the NVM Controller.

The number of cycles data are delayed to the AHB bus is determined by the read wait states. Examples of using zero and one wait states are shown in the following figure.

Reading the NVM main address space while a programming or erase operation is ongoing on the NVM main array results in an AHB bus stall until the end of the operation. Reading the NVM main array does not stall the bus when the RWWEE array is being programmed or erased.

**Figure 22-5. Read Wait State Examples**



#### 22.6.4.2 RWWEE Read

Reading from the RWW EEPROM address space is performed via the AHB bus by addressing the RWWEE address space directly.

Read timings are similar to regular NVM read timings when access size is Byte or half-Word. The AHB data phase is twice as long in case of full-Word-size access.

It is not possible to read the RWWEE area while the NVM main array is being written or erased, whereas the RWWEE area can be written or erased while the main array is being read.

The RWWEE address space is not cached, therefore it is recommended to limit access to this area for performance and power consumption considerations.

#### 22.6.4.3 NVM Write

The NVM Controller requires that an erase must be done before programming. The entire NVM main address space and the RWWEE address space can be erased by a debugger Chip Erase command. Alternatively, rows can be individually erased by the Erase Row command or the RWWEE Erase Row command to erase the NVM main address space or the RWWEE address space, respectively.

After programming the NVM main array, the region that the page resides in can be locked to prevent spurious write or erase sequences. Locking is performed on a per-region basis, and so, locking a region will lock all pages inside the region.

Data to be written to the NVM block are first written to and stored in an internal buffer called the *page buffer*. The page buffer contains the same number of bytes as an NVM page. Writes to the page buffer must be 16 or 32 bits. 8-bit writes to the page buffer are not allowed and will cause a system exception.

Both the NVM main array and the RWWEE array share the same page buffer. Writing to the NVM block via the AHB bus is performed by a load operation to the page buffer. For each AHB bus write, the address is stored in the ADDR register. After the page buffer has been loaded with the required number of bytes, the page can be written to the NVM main array or the RWWEE array by setting CTRLA.CMD to 'Write Page' or 'RWWEE Write Page', respectively, and setting the key value to CMDEX. The LOAD bit in the STATUS register indicates whether the page buffer has been loaded or not. Before writing the page to memory, the accessed row must be erased.

Automatic page writes are enabled by writing the manual write bit to zero (CTRLB.MANW=0). This will trigger a write operation to the page addressed by ADDR when the last location of the page is written.

Because the address is automatically stored in ADDR during the I/O bus write operation, the last given address will be present in the ADDR register. There is no need to load the ADDR register manually, unless a different page in memory is to be written.

##### 22.6.4.3.1 Procedure for Manual Page Writes (CTRLB.MANW=1)

The row to be written to must be erased before the write command is given.

- Write to the page buffer by addressing the NVM main address space directly
- Write the page buffer to memory: CTRL.CMD='Write Page' and CMDEX
- The READY bit in the INTFLAG register will be low while programming is in progress, and access through the AHB will be stalled

##### 22.6.4.3.2 Procedure for Automatic Page Writes (CTRLB.MANW=0)

The row to be written to must be erased before the last write to the page buffer is performed.

Note that partially written pages must be written with a manual write.

- Write to the page buffer by addressing the NVM main address space directly.

When the last location in the page buffer is written, the page is automatically written to NVM main address space.

- INTFLAG.READY will be zero while programming is in progress and access through the AHB will be stalled.

#### 22.6.4.4 Page Buffer Clear

The page buffer is automatically set to all '1' after a page write is performed. If a partial page has been written and it is desired to clear the contents of the page buffer, the Page Buffer Clear command can be used.

#### 22.6.4.5 Erase Row

Before a page can be written, the row containing that page must be erased. The Erase Row command can be used to erase the desired row in the NVM main address space. The RWWEE Erase Row can be used to erase the desired row in the RWWEE array. Erasing the row sets all bits to '1'. If the row resides in a region that is locked, the erase will not be performed and the Lock Error bit in the Status register (STATUS.LOCKE) will be set.

##### 22.6.4.5.1 Procedure for Erase Row

- Write the address of the row to erase to ADDR. Any address within the row can be used.
- Issue an Erase Row command.

**Note:** The NVM Address bit field in the Address register (ADDR.ADDR) uses 16-bit addressing.

#### 22.6.4.6 Lock and Unlock Region

These commands are used to lock and unlock regions as detailed in section 22.6.3 Region Lock Bits.

#### 22.6.4.7 Set and Clear Power Reduction Mode

The NVM Controller and block can be taken in and out of power reduction mode through the Set and Clear Power Reduction Mode commands. When the NVM Controller and block are in power reduction mode, the Power Reduction Mode bit in the Status register (STATUS.PRM) is set.

### 22.6.5 NVM User Configuration

The NVM user configuration resides in the auxiliary space. Refer to the Physical Memory Map of the device for calibration and auxiliary space address mapping.

The bootloader resides in the main array starting at offset zero. The allocated boot loader section is write-protected.

**Table 22-2. Boot Loader Size**

| BOOTPROT [2:0] | Rows Protected by BOOTPROT | Boot Loader Size in Bytes |
|---|---|---|
| 0x7(1) | None | 0 |
| 0x6 | 2 | 512 |
| 0x5 | 4 | 1024 |
| 0x4 | 8 | 2048 |
| 0x3 | 16 | 4096 |
| 0x2 | 32 | 8192 |
| 0x1 | 64 | 16384 |
| 0x0 | 128 | 32768 |

**Note:** 1) Default value is 0x7.

The EEPROM[2:0] bits indicate the EEPROM size, see the table below. The EEPROM resides in the upper rows of the NVM main address space and is writable, regardless of the region lock status.

**Table 22-3. EEPROM Size**

| EEPROM[2:0] | Rows Allocated to EEPROM | EEPROM Size in Bytes |
|---|---|---|
| 7 | None | 0 |
| 6 | 1 | 256 |
| 5 | 2 | 512 |
| 4 | 4 | 1024 |
| 3 | 8 | 2048 |
| 2 | 16 | 4096 |
| 1 | 32 | 8192 |
| 0 | 64 | 16384 |

**Related Links**

10.2 Physical Memory Map

### 22.6.6 Security Bit

The security bit allows the entire chip to be locked from external access for code security. The security bit can be written by a dedicated command, Set Security Bit (SSB). Once set, the only way to clear the security bit is through a debugger Chip Erase command. After issuing the SSB command, the PROGE error bit can be checked.

In order to increase the security level it is recommended to enable the internal BOD33 when the security bit is set.

**Related Links**

13. DSU - Device Service Unit

### 22.6.7 Cache

The NVM Controller cache reduces the device power consumption and improves system performance when wait states are required. Only the NVM main array address space is cached. It is a direct-mapped cache that implements 8 lines of 64 bits (i.e., 64 Bytes). NVM Controller cache can be enabled by writing a '0' to the Cache Disable bit in the Control B register (CTRLB.CACHEDIS).

The cache can be configured to three different modes using the Read Mode bit group in the Control B register (CTRLB.READMODE).

The INVALL command can be issued using the Command bits in the Control A register to invalidate all cache lines (CTRLA.CMD=INVALL). Commands affecting NVM content automatically invalidate cache lines.

## 22.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | | | CMD[6:0] | | | | |
| | | 15:8 | CMDEX[7:0] | | | | | | | |
| 0x02 ... 0x03 | Reserved | | | | | | | | | |
| 0x04 | CTRLB | 7:0 | MANW | | | RWS[3:0] | | | | |
| | | 15:8 | | | | | | | SLEEPPRM[1:0] | |
| | | 23:16 | | | | | | CACHEDIS | READMODE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x08 | PARAM | 7:0 | NVMP[7:0] | | | | | | | |
| | | 15:8 | NVMP[15:8] | | | | | | | |
| | | 23:16 | RWWEEP[3:0] | | | | | PSZ[2:0] | | |
| | | 31:24 | RWWEEP[11:4] | | | | | | | |
| 0x0C | INTENCLR | 7:0 | | | | | | | ERROR | READY |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | INTENSET | 7:0 | | | | | | | ERROR | READY |
| 0x11 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTFLAG | 7:0 | | | | | | | ERROR | READY |
| 0x15 ... 0x17 | Reserved | | | | | | | | | |
| 0x18 | STATUS | 7:0 | | | | NVME | LOCKE | PROGE | LOAD | PRM |
| | | 15:8 | | | | | | | | SB |
| 0x1A ... 0x1B | Reserved | | | | | | | | | |
| 0x1C | ADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | ADDR[21:16] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | LOCK | 7:0 | LOCK[7:0] | | | | | | | |
| | | 15:8 | LOCK[15:8] | | | | | | | |

## 22.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 22.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMDEX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CMD[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – CMDEX[7:0]** Command Execution
When this bit group is written to the key value 0xA5, the command written to CMD will be executed. If a value different from the key value is tried, the write will not be performed and the Programming Error bit in the Status register (STATUS.PROGE) will be set. PROGE is also set if a previously written command is not completed yet.

The key value must be written at the same time as CMD. If a command is issued through the APB bus on the same cycle as an AHB bus access, the AHB bus access will be given priority. The command will then be executed when the NVM block and the AHB bus are idle.

INTFLAG.READY must be '1' when the command is issued.

Bit 0 of the CMDEX bit group will read back as '1' until the command is issued.

**Note:** The NVM Address bit field in the Address register (ADDR.ADDR) uses 16-bit addressing.

**Bits 6:0 – CMD[6:0]** Command
These bits define the command to be executed when the CMDEX key is written.

| CMD[6:0] | Group Configuration | Description |
|---|---|---|
| 0x00-0x01 | - | Reserved |
| 0x02 | ER | Erase Row - Erases the row addressed by the ADDR register in the NVM main array. |
| 0x03 | - | Reserved |
| 0x04 | WP | Write Page - Writes the contents of the page buffer to the page addressed by the ADDR register. |
| 0x05 | EAR | Erase Auxiliary Row - Erases the auxiliary row addressed by the ADDR register. This command can be given only when the security bit is not set and only to the User Configuration Row. |

| CMD[6:0] | Group Configuration | Description |
|---|---|---|
| 0x06 | WAP | Write Auxiliary Page - Writes the contents of the page buffer to the page addressed by the ADDR register. This command can be given only when the security bit is not set and only to the User Configuration Row. |
| 0x07-0x0E | - | Reserved |
| 0x0F | WL | Write Lockbits- write the LOCK register |
| 0x1A-0x19 | - | Reserved |
| 0x1A | RWWEEER | RWWEE Erase Row - Erases the row addressed by the ADDR register in the RWWEE array. |
| 0x1B | - | Reserved |
| 0x1C | RWWEEWP | RWWEE Write Page - Writes the contents of the page buffer to the page addressed by the ADDR register in the RWWEE array. |
| 0x1D-0x3F | - | Reserved |
| 0x40 | LR | Lock Region - Locks the region containing the address location in the ADDR register. |
| 0x41 | UR | Unlock Region - Unlocks the region containing the address location in the ADDR register. |
| 0x42 | SPRM | Sets the Power Reduction Mode. |
| 0x43 | CPRM | Clears the Power Reduction Mode. |
| 0x44 | PBC | Page Buffer Clear - Clears the page buffer. |
| 0x45 | SSB | Set Security Bit - Sets the security bit by writing 0x00 to the first byte in the lockbit row. |
| 0x46 | INVALL | Invalidates all cache lines. |
| 0x47 | LDR | Lock Data Region - Locks the data region containing the address location in the ADDR register.<br>When the Security Extension is enabled, only secure access can lock secure regions. |
| 0x48 | UDR | Unlock Data Region - Unlocks the data region containing the address location in the ADDR register.<br>When the Security Extension is enabled, only secure access can unlock secure regions. |
| 0x47-0x7F | - | Reserved |

### 22.8.2 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000080
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CACHEDIS | READMODE[1:0] | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SLEEPPRM[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MANW | | | RWS[3:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | |
| Reset | 1 | | | 0 | 0 | 0 | 0 | |

**Bit 18 – CACHEDIS** Cache Disable
This bit is used to disable the cache.

| Value | Description |
|---|---|
| 0 | The cache is enabled |
| 1 | The cache is disabled |

**Bits 17:16 – READMODE[1:0]** NVMCTRL Read Mode

| Value | Name | Description |
|---|---|---|
| 0x0 | NO_MISS_PENALTY | The NVM Controller (cache system) does not insert wait states on a cache miss. Gives the best system performance. |
| 0x1 | LOW_POWER | Reduces power consumption of the cache system, but inserts a wait state each time there is a cache miss. This mode may not be relevant if CPU performance is required, as the application will be stalled and may lead to increased run time. |
| 0x2 | DETERMINISTIC | The cache system ensures that a cache hit or miss takes the same amount of time, determined by the number of programmed Flash wait states. This mode can be used for real-time applications that require deterministic execution timings. |
| 0x3 | Reserved | |

**Bits 9:8 – SLEEPPRM[1:0]** Power Reduction Mode during Sleep
Indicates the Power Reduction Mode during sleep.

| Value | Name | Description |
|---|---|---|
| 0x0 | WAKEUPACCESS | NVM block enters low-power mode when entering sleep.<br>NVM block exits low-power mode upon first access. |
| 0x1 | WAKEUPINSTANT | NVM block enters low-power mode when entering sleep.<br>NVM block exits low-power mode when exiting sleep. |
| 0x2 | Reserved | |
| 0x3 | DISABLED | Auto power reduction disabled. |

**Bit 7 – MANW** Manual Write
Note that reset value of this bit is '1'.

| Value | Description |
|---|---|
| 0 | Writing to the last word in the page buffer will initiate a write operation to the page addressed by the last write operation. This includes writes to memory and auxiliary rows. |
| 1 | Write commands must be issued through the CTRLA.CMD register. |

**Bits 4:1 – RWS[3:0]** NVM Read Wait States
These bits control the number of wait states for a read operation. '0' indicates zero wait states, '1' indicates one wait state, etc., up to 15 wait states.
This register is initialized to 0 wait states. Software can change this value based on the NVM access time and system frequency.

### 22.8.3 NVM Parameter

**Name:** PARAM
**Offset:** 0x08
**Reset:** 0x000XXXXX
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RWWEEP[11:4] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RWWEEP[3:0] | | | | | PSZ[2:0] | | |
| Access | R | R | R | R | | R | R | R |
| Reset | 0 | 0 | 0 | 0 | | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NVMP[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NVMP[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

**Bits 31:20 – RWWEEP[11:0]** Read While Write EEPROM emulation area Pages
Indicates the number of pages in the RWW EEPROM emulation address space.

**Bits 18:16 – PSZ[2:0]** Page Size
Indicates the page size. Not all devices of the device families will provide all the page sizes indicated in the table.

| Value | Name | Description |
|---|---|---|
| 0x0 | 8 | 8 bytes |
| 0x1 | 16 | 16 bytes |
| 0x2 | 32 | 32 bytes |
| 0x3 | 64 | 64 bytes |
| 0x4 | 128 | 128 bytes |
| 0x5 | 256 | 256 bytes |
| 0x6 | 512 | 512 bytes |
| 0x7 | 1024 | 1024 bytes |

**Bits 15:0 – NVMP[15:0]** NVM Pages
Indicates the number of pages in the NVM main address space.

### 22.8.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x0C
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ERROR | READY |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ERROR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the ERROR interrupt enable.

This bit will read as the current value of the ERROR interrupt enable.

**Bit 0 – READY** NVM Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the READY interrupt enable.

This bit will read as the current value of the READY interrupt enable.

### 22.8.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x10
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ERROR | READY |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ERROR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit sets the ERROR interrupt enable.

This bit will read as the current value of the ERROR interrupt enable.

**Bit 0 – READY** NVM Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit sets the READY interrupt enable.

This bit will read as the current value of the READY interrupt enable.

### 22.8.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x14
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ERROR | READY |
| Access | | | | | | | R/W | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ERROR** Error
This flag is set on the occurrence of an NVME, LOCKE or PROGE error.

This bit can be cleared by writing a '1' to its bit location.

| Value | Description |
|---|---|
| 0 | No errors have been received since the last clear. |
| 1 | At least one error has occurred since the last clear. |

**Bit 0 – READY** NVM Ready

| Value | Description |
|---|---|
| 0 | The NVM controller is busy programming or erasing. |
| 1 | The NVM controller is ready to accept a new command. |

### 22.8.7 Status

**Name:** STATUS
**Offset:** 0x18
**Reset:** 0x0X00
**Property:** –

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SB |
| Access | | | | | | | | R |
| Reset | | | | | | | | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NVME | LOCKE | PROGE | LOAD | PRM |
| Access | | | | R/W | R/W | R/W | R/W | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – SB** Security Bit Status

| Value | Description |
|---|---|
| 0 | The Security bit is inactive. |
| 1 | The Security bit is active. |

**Bit 4 – NVME** NVM Error
This bit can be cleared by writing a '1' to its bit location.

| Value | Description |
|---|---|
| 0 | No programming or erase errors have been received from the NVM controller since this bit was last cleared. |
| 1 | At least one error has been registered from the NVM Controller since this bit was last cleared. |

**Bit 3 – LOCKE** Lock Error Status
This bit can be cleared by writing a '1' to its bit location.

| Value | Description |
|---|---|
| 0 | No programming of any locked lock region has happened since this bit was last cleared. |
| 1 | Programming of at least one locked lock region has happened since this bit was last cleared. |

**Bit 2 – PROGE** Programming Error Status
This bit can be cleared by writing a '1' to its bit location.

| Value | Description |
|---|---|
| 0 | No invalid commands or bad keywords were written in the NVM Command register since this bit was last cleared. |
| 1 | An invalid command and/or a bad keyword was/were written in the NVM Command register since this bit was last cleared. |

**Bit 1 – LOAD** NVM Page Buffer Active Loading
This bit indicates that the NVM page buffer has been loaded with one or more words. Immediately after an NVM load has been performed, this flag is set. It remains set until a page write or a page buffer clear (PBCLR) command is given.

This bit can be cleared by writing a '1' to its bit location.

**Bit 0 – PRM** Power Reduction Mode
This bit indicates the current NVM power reduction state. The NVM block can be set in power reduction mode in two ways: through the command interface or automatically when entering sleep with SLEEPPRM set accordingly.

PRM can be cleared in three ways: through AHB access to the NVM block, through the command interface (SPRM and CPRM) or when exiting sleep with SLEEPPRM set accordingly.

| Value | Description |
|---|---|
| 0 | NVM is not in power reduction mode. |
| 1 | NVM is in power reduction mode. |

### 22.8.8 Address

**Name:** ADDR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | ADDR[21:16] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:0 – ADDR[21:0]** NVM Address
ADDR drives the hardware (16-bit) address to the NVM when a command is executed using CMDEX. This register is also automatically updated when writing to the page buffer.

### 22.8.9 Lock Section

**Name:** LOCK
**Offset:** 0x20
**Reset:** 0xXXXX
**Property:** –

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOCK[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LOCK[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

**Bits 15:0 – LOCK[15:0]** Region Lock Bits
To set or clear these bits, the CMD register must be used.

Default state after erase will be unlocked (0x0000).

| Value | Description |
|---|---|
| 0 | The corresponding lock region is locked. |
| 1 | The corresponding lock region is not locked. |

# 23. PORT - I/O Pin Controller

## 23.1 Overview

The IO Pin Controller (PORT) controls the I/O pins of the device. The I/O pins are organized in a series of groups, collectively referred to as a PORT group. Each PORT group can have up to 32 pins that can be configured and controlled individually or as a group. The number of PORT groups on a device may depend on the package/number of pins. Each pin may either be used for general-purpose I/O under direct application control or be assigned to an embedded device peripheral. When used for general-purpose I/O, each pin can be configured as input or output, with highly configurable driver and pull settings.

All I/O pins have true read-modify-write functionality when used for general-purpose I/O; the direction or the output value of one or more pins may be changed (set, reset or toggled) explicitly without unintentionally changing the state of any other pins in the same port group by a single, atomic 8-, 16- or 32-bit write.

The PORT is connected to the high-speed bus matrix through an AHB/APB bridge. The Pin Direction, Data Output Value and Data Input Value registers may also be accessed using the low-latency CPU local bus (IOBUS; ARM® single-cycle I/O port) .

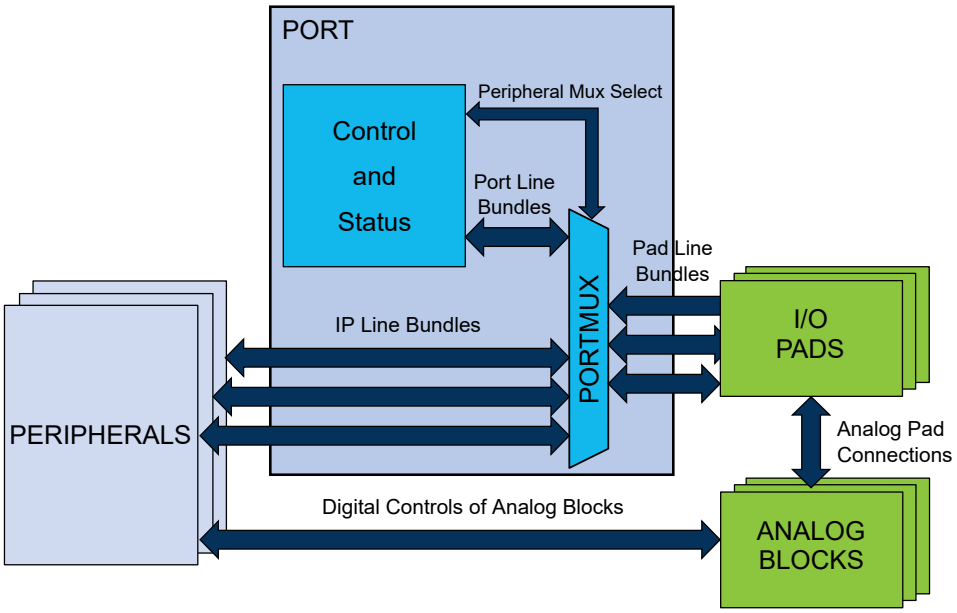
## 23.2 Features

- Selectable input and output configuration for each individual pin
- Software-controlled multiplexing of peripheral functions on I/O pins
- Flexible pin configuration through a dedicated Pin Configuration register
- Configurable output driver and pull settings:
  - Totem-pole (push-pull)
  - Pull configuration
  - Driver strength
- Configurable input buffer and pull settings:
  - Internal pull-up or pull-down
  - Input sampling criteria
  - Input buffer can be disabled if not needed for lower power consumption
  - Read-Modify-Write support for output value (OUTCLR/OUTSET/OUTGL) and pin direction (DIRCLR/DIRSET/DIRTGL)

© 2018 Microchip Technology Inc.

## 23.3 Block Diagram

**Figure 23-1. PORT Block Diagram**




## 23.4 Signal Description

**Table 23-1. Signal description for PORT**

| Signal name | Type | Description |
|---|---|---|
| Pxy | Digital I/O | General-purpose I/O pin y in group x |

Refer to the *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 23.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly as follows.

### 23.5.1 I/O Lines

The I/O lines of the PORT are mapped to pins of the physical device. The following naming scheme is used:

Each line bundle with up to 32 lines is assigned an identifier 'xy', with letter x=A, B, C… and two-digit number y=00, 01, …31. Examples: A24, C03.

PORT pins are labeled 'Pxy' accordingly, for example PA24, PC03. This identifies each pin in the device uniquely.

Each pin may be controlled by one or more peripheral multiplexer settings, which allow the pad to be routed internally to a dedicated peripheral function. When the setting is enabled, the selected peripheral

has control over the output state of the pad, as well as the ability to read the current physical pad state. Refer to *I/O Multiplexing and Considerations* for details.

Device-specific configurations may cause some lines (and the corresponding Pxy pin) not to be implemented.

**Related Links**

7. I/O Multiplexing and Considerations

### 23.5.2 Power Management

During Reset, all PORT lines are configured as inputs with input buffers, output buffers and pull disabled.

The PORT peripheral will continue operating in any sleep mode where its source clock is running.

### 23.5.3 Clocks

The PORT bus clock (CLK_PORT_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_PORT_APB can be found in the *Peripheral Clock Masking* section in *PM – Power Manager*.

The PORT requires an APB clock, which may be divided from the CPU main clock and allows the CPU to access the registers of PORT through the high-speed matrix and the AHB/APB bridge.

The PORT also requires an AHB clock for CPU IOBUS accesses to the PORT. That AHB clock is the internal PORT clock.

The priority of IOBUS accesses is higher than APB accesses. One clock cycle latency can be observed on the APB access in case of concurrent PORT accesses.

**Related Links**

16.6.2.6 Peripheral Clock Masking

### 23.5.4 DMA

Not applicable.

### 23.5.5 Interrupts

Not applicable.

### 23.5.6 Events

The events of this peripheral are connected to the Event System.

**Related Links**

24. EVSYS – Event System

### 23.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation.

### 23.5.8 Register Access Protection

All registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC).

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 23.5.9 Analog Connections

Analog functions are connected directly between the analog blocks and the I/O pads using analog buses. However, selecting an analog peripheral function for a given pin will disable the corresponding digital features of the pad.

### 23.5.10 CPU Local Bus

The CPU local bus (IOBUS) is an interface that connects the CPU directly to the PORT. It is a single-cycle bus interface, which does not support wait states. It supports 8-bit, 16-bit and 32-bit sizes.

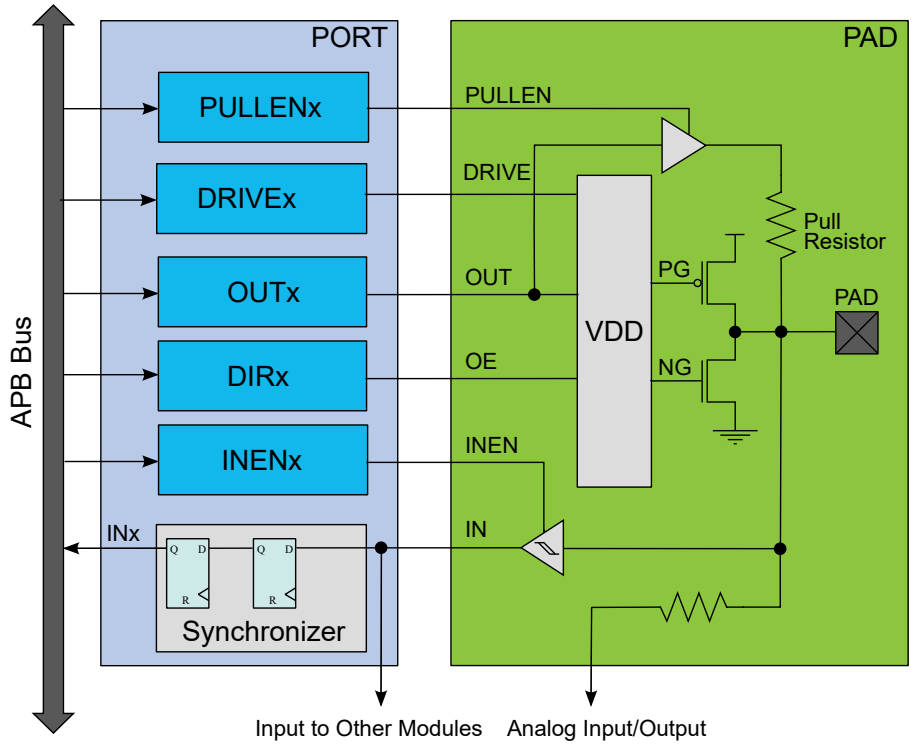
This bus is generally used for low latency operation. The Data Direction (DIR) and Data Output Value (OUT) registers can be read, written, set, cleared or be toggled using this bus, and the Data Input Value (IN) registers can be read.

Since the IOBUS cannot wait for IN register resynchronization, the Control register (CTRL) must be configured to continuous sampling of all pins that need to be read via the IOBUS in order to prevent stale data from being read.

**Note:** Refer to the *Product Mapping* chapter for the IOBUS address.

## 23.6 Functional Description

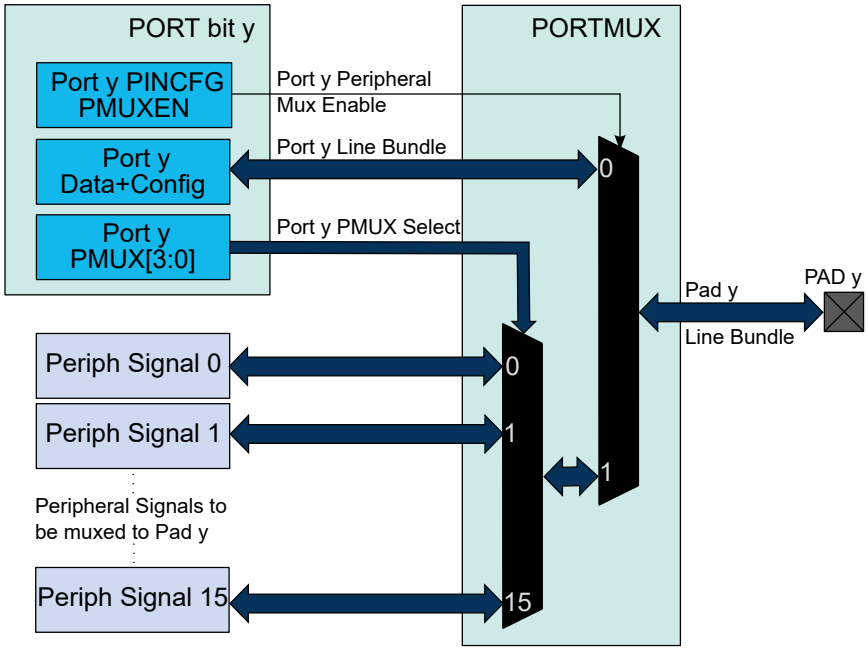
**Figure 23-2. Overview of the PORT**



### 23.6.1 Principle of Operation

Each PORT group of up to 32 pins is controlled by the registers in PORT, as described in the figure. These registers in PORT are duplicated for each PORT group, with increasing base addresses. The number of PORT groups may depend on the package/number of pins.

**Figure 23-3. Overview of the peripheral functions multiplexing**




The I/O pins of the device are controlled by PORT peripheral registers. Each port pin has a corresponding bit in the Data Direction (DIR) and Data Output Value (OUT) registers to enable that pin as an output and to define the output state.

The direction of each pin in a PORT group is configured by the DIR register. If a bit in DIR is set to '1', the corresponding pin is configured as an output pin. If a bit in DIR is set to '0', the corresponding pin is configured as an input pin.

When the direction is set as output, the corresponding bit in the OUT register will set the level of the pin. If bit y in OUT is written to '1', pin y is driven HIGH. If bit y in OUT is written to '0', pin y is driven LOW. Pin configuration can be set by Pin Configuration (PINCFGy) registers, with y=00, 01, ..31 representing the bit position.

The Data Input Value (IN) is set as the input value of a port pin with resynchronization to the PORT clock. To reduce power consumption, these input synchronizers can be clocked only when system requires reading the input value, as specified in the SAMPLING field of the Control register (CTRL). The value of the pin can always be read, whether the pin is configured as input or output. If the Input Enable bit in the Pin Configuration registers (PINCFGy.INEN) is '0', the input value will not be sampled.

In PORT, the Peripheral Multiplexer Enable bit in the PINCFGy register (PINCFGy.PMUXEN) can be written to '1' to enable the connection between peripheral functions and individual I/O pins. The Peripheral Multiplexing n (PMUXn) registers select the peripheral function for the corresponding pin. This will override the connection between the PORT and that I/O pin, and connect the selected peripheral signal to the particular I/O pin instead of the PORT line bundle.

### 23.6.2 Basic Operation

#### 23.6.2.1 Initialization

After reset, all standard function device I/O pads are connected to the PORT with outputs tri-stated and input buffers disabled, even if there is no clock running.

However, specific pins, such as those used for connection to a debugger, may be configured differently, as required by their special function.

#### 23.6.2.2 Operation

Each I/O pin Pxy can be controlled by the registers in PORT. Each PORT group x has its own set of PORT registers, with a base address at byte address (PORT + 0x80 * group index) (A corresponds to group index 0, B to 1, etc...). Within that set of registers, the pin index is y, from 0 to 31.

Refer to *I/O Multiplexing and Considerations* for details on available pin configuration and PORT groups.

**Configuring Pins as Output**

To use pin Pxy as an *output*, write bit y of the DIR register to '1'. This can also be done by writing bit y in the DIRSET register to '1' - this will avoid disturbing the configuration of other pins in that group. The y bit in the OUT register must be written to the desired output value.

Similarly, writing an OUTSET bit to '1' will set the corresponding bit in the OUT register to '1'. Writing a bit in OUTCLR to '1' will set that bit in OUT to zero. Writing a bit in OUTTGL to '1' will toggle that bit in OUT.

**Configuring Pins as Input**

To use pin Pxy as an *input*, bit y in the DIR register must be written to '0'. This can also be done by writing bit y in the DIRCLR register to '1' - this will avoid disturbing the configuration of other pins in that group. The input value can be read from bit y in register IN as soon as the INEN bit in the Pin Configuration register (PINCFGy.INEN) is written to '1'.

By default, the input synchronizer is clocked only when an input read is requested. This will delay the read operation by two cycles of the PORT clock. To remove the delay, the input synchronizers for each PORT group of eight pins can be configured to be always active, but this will increase power consumption. This is enabled by writing '1' to the corresponding SAMPLINGn bit field of the CTRL register, see CTRL.SAMPLING for details.

**Using Alternative Peripheral Functions**

To use pin Pxy as one of the available peripheral functions, the corresponding PMUXEN bit of the PINCFGy register must be '1'. The PINCFGy register for pin Pxy is at byte offset (PINCFG0 + y).

The peripheral function can be selected by setting the PMUXO or PMUXE in the PMUXn register. The PMUXO/PMUXE is at byte offset PMUX0 + (y/2). The chosen peripheral must also be configured and enabled.

**Related Links**

7. I/O Multiplexing and Considerations

### 23.6.3 I/O Pin Configuration

The Pin Configuration register (PINCFGy) is used for additional I/O pin configuration. A pin can be set in a totem-pole or pull configuration.

As pull configuration is done through the Pin Configuration register, all intermediate PORT states during switching of pin direction and pin values are avoided.

The I/O pin configurations are described further in this chapter, and summarized in Table 23-2.

#### 23.6.3.1 Pin Configurations Summary

**Table 23-2. Pin Configurations Summary**

| DIR | INEN | PULLEN | OUT | Configuration |
|---|---|---|---|---|
| 0 | 0 | 0 | X | Reset or analog I/O: all digital disabled |
| 0 | 0 | 1 | 0 | Pull-down; input disabled |
| 0 | 0 | 1 | 1 | Pull-up; input disabled |
| 0 | 1 | 0 | X | Input |
| 0 | 1 | 1 | 0 | Input with pull-down |
| 0 | 1 | 1 | 1 | Input with pull-up |
| 1 | 0 | X | X | Output; input disabled |
| 1 | 1 | X | X | Output; input enabled |

#### 23.6.3.2 Input Configuration

**Figure 23-4. I/O configuration - Standard Input**

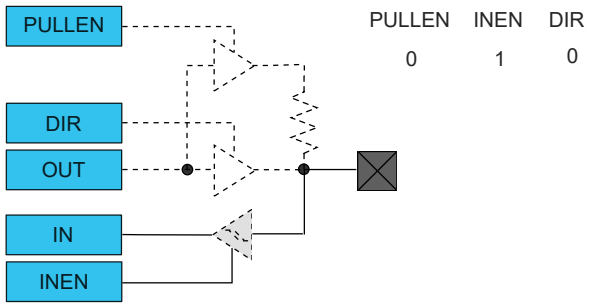
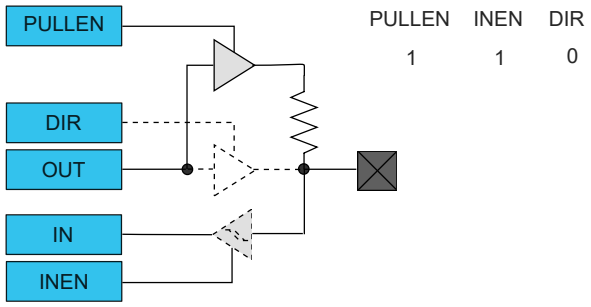



**Figure 23-5. I/O Configuration - Input with Pull**




**Note:** When pull is enabled, the pull value is defined by the OUT value.

#### 23.6.3.3 Totem-Pole Output

When configured for totem-pole (push-pull) output, the pin is driven low or high according to the corresponding bit setting in the OUT register. In this configuration there is no current limitation for sink or source other than what the pin is capable of. If the pin is configured for input, the pin will float if no external pull is connected.

**Note:** Enabling the output driver will automatically disable pull.

**Figure 23-6. I/O Configuration - Totem-Pole Output with Disabled Input**

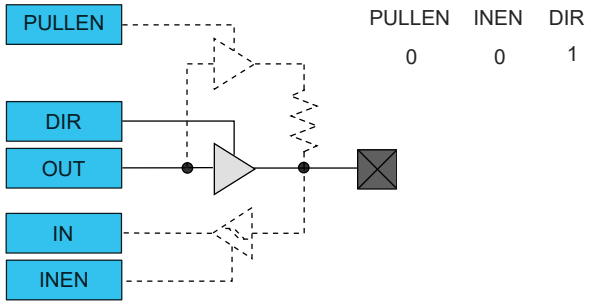



**Figure 23-7. I/O Configuration - Totem-Pole Output with Enabled Input**

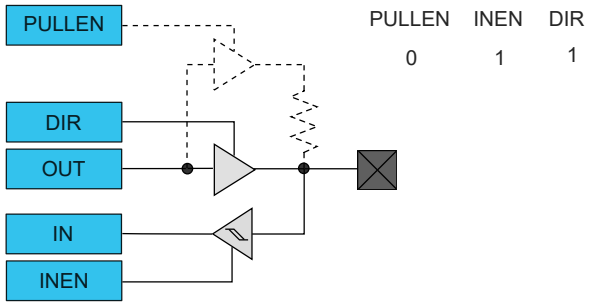
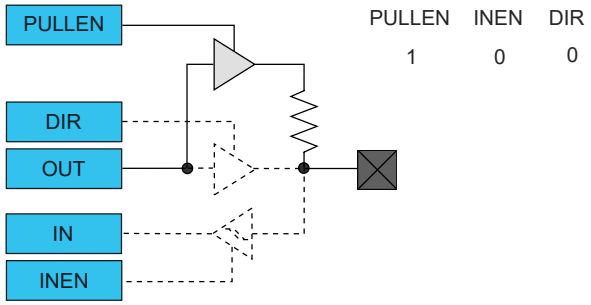


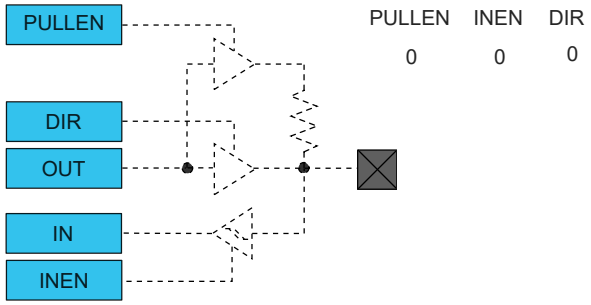

**Figure 23-8. I/O Configuration - Output with Pull**




#### 23.6.3.4 Digital Functionality Disabled

Neither Input nor Output functionality are enabled.

**Figure 23-9. I/O Configuration - Reset or Analog I/O: Digital Output, Input and Pull Disabled**




### 23.6.4 PORT Access Priority

The PORT is accessed by different systems:

- The ARM® CPU through the ARM® single-cycle I/O port (IOBUS)
- The ARM® CPU through the high-speed matrix and the AHB/APB bridge (APB)

The following priority is adopted:

1. ARM® CPU IOBUS (No wait tolerated)
2. APB

## 23.7 Register Summary

The I/O pins are assembled in pin groups with up to 32 pins. Group 0 consists of the PA pins, and group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | DIR | 7:0 | DIR[7:0] | | | | | | | |
| | | 15:8 | DIR[15:8] | | | | | | | |
| | | 23:16 | DIR[23:16] | | | | | | | |
| | | 31:24 | DIR[31:24] | | | | | | | |
| 0x04 | DIRCLR | 7:0 | DIRCLR[7:0] | | | | | | | |
| | | 15:8 | DIRCLR[15:8] | | | | | | | |
| | | 23:16 | DIRCLR[23:16] | | | | | | | |
| | | 31:24 | DIRCLR[31:24] | | | | | | | |
| 0x08 | DIRSET | 7:0 | DIRSET[7:0] | | | | | | | |
| | | 15:8 | DIRSET[15:8] | | | | | | | |
| | | 23:16 | DIRSET[23:16] | | | | | | | |
| | | 31:24 | DIRSET[31:24] | | | | | | | |
| 0x0C | DIRTGL | 7:0 | DIRTGL[7:0] | | | | | | | |
| | | 15:8 | DIRTGL[15:8] | | | | | | | |
| | | 23:16 | DIRTGL[23:16] | | | | | | | |
| | | 31:24 | DIRTGL[31:24] | | | | | | | |
| 0x10 | OUT | 7:0 | OUT[7:0] | | | | | | | |
| | | 15:8 | OUT[15:8] | | | | | | | |
| | | 23:16 | OUT[23:16] | | | | | | | |
| | | 31:24 | OUT[31:24] | | | | | | | |
| 0x14 | OUTCLR | 7:0 | OUTCLR[7:0] | | | | | | | |
| | | 15:8 | OUTCLR[15:8] | | | | | | | |
| | | 23:16 | OUTCLR[23:16] | | | | | | | |
| | | 31:24 | OUTCLR[31:24] | | | | | | | |
| 0x18 | OUTSET | 7:0 | OUTSET[7:0] | | | | | | | |
| | | 15:8 | OUTSET[15:8] | | | | | | | |
| | | 23:16 | OUTSET[23:16] | | | | | | | |
| | | 31:24 | OUTSET[31:24] | | | | | | | |
| 0x1C | OUTTGL | 7:0 | OUTTGL[7:0] | | | | | | | |
| | | 15:8 | OUTTGL[15:8] | | | | | | | |
| | | 23:16 | OUTTGL[23:16] | | | | | | | |
| | | 31:24 | OUTTGL[31:24] | | | | | | | |
| 0x20 | IN | 7:0 | IN[7:0] | | | | | | | |
| | | 15:8 | IN[15:8] | | | | | | | |
| | | 23:16 | IN[23:16] | | | | | | | |
| | | 31:24 | IN[31:24] | | | | | | | |
| 0x24 | CTRL | 7:0 | SAMPLING[7:0] | | | | | | | |
| | | 15:8 | SAMPLING[15:8] | | | | | | | |
| | | 23:16 | SAMPLING[23:16] | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 31:24 | SAMPLING[31:24] | | | | | | | |
| 0x28 | WRCONFIG | 7:0 | PINMASK[7:0] | | | | | | | |
| | | 15:8 | PINMASK[15:8] | | | | | | | |
| | | 23:16 | | | | | | PULLEN | INEN | PMUXEN |
| | | 31:24 | HWSEL | WRPINCFG | | WRPMUX | | PMUX[3:0] | | |
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | PMUX0 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x31 | PMUX1 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x32 | PMUX2 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x33 | PMUX3 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x34 | PMUX4 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x35 | PMUX5 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x36 | PMUX6 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x37 | PMUX7 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x38 | PMUX8 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x39 | PMUX9 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3A | PMUX10 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3B | PMUX11 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3C | PMUX12 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3D | PMUX13 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3E | PMUX14 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x3F | PMUX15 | 7:0 | PMUXO[3:0] | | | | | PMUXE[3:0] | | |
| 0x40 | PINCFG0 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x41 | PINCFG1 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x42 | PINCFG2 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x43 | PINCFG3 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x44 | PINCFG4 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x45 | PINCFG5 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x46 | PINCFG6 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x47 | PINCFG7 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x48 | PINCFG8 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x49 | PINCFG9 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4A | PINCFG10 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4B | PINCFG11 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4C | PINCFG12 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4D | PINCFG13 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4E | PINCFG14 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x4F | PINCFG15 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x50 | PINCFG16 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x51 | PINCFG17 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x52 | PINCFG18 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x53 | PINCFG19 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x54 | PINCFG20 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x55 | PINCFG21 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x56 | PINCFG22 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x57 | PINCFG23 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x58 | PINCFG24 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x59 | PINCFG25 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5A | PINCFG26 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5B | PINCFG27 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5C | PINCFG28 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5D | PINCFG29 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5E | PINCFG30 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |
| 0x5F | PINCFG31 | 7:0 | | | | | | PULLEN | INEN | PMUXEN |

## 23.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 23.5.8 Register Access Protection.

### 23.8.1 Data Direction

**Name:** DIR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to configure one or more I/O pins as an input or output. This register can be manipulated without doing a read-modify-write operation by using the Data Direction Toggle (DIRTGL), Data Direction Clear (DIRCLR) and Data Direction Set (DIRSET) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DIR[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIR[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DIR[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIR[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DIR[31:0]** Port Data Direction
These bits set the data direction for the individual I/O pins in the PORT group.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group is configured as an input. |
| 1 | The corresponding I/O pin in the PORT group is configured as an output. |

### 23.8.2 Data Direction Clear

**Name:** DIRCLR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to set one or more I/O pins as an input, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Toggle (DIRTGL) and Data Direction Set (DIRSET) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DIRCLR[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIRCLR[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DIRCLR[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIRCLR[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DIRCLR[31:0]** Port Data Direction Clear
Writing a '0' to a bit has no effect.

Writing a '1' to a bit will clear the corresponding bit in the DIR register, which configures the I/O pin as an input.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group will keep its configuration. |
| 1 | The corresponding I/O pin in the PORT group is configured as input. |

### 23.8.3 Data Direction Set

**Name:** DIRSET
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to set one or more I/O pins as an output, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Toggle (DIRTGL) and Data Direction Clear (DIRCLR) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DIRSET[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIRSET[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DIRSET[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIRSET[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DIRSET[31:0]** Port Data Direction Set
Writing '0' to a bit has no effect.

Writing '1' to a bit will set the corresponding bit in the DIR register, which configures the I/O pin as an output.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group will keep its configuration. |
| 1 | The corresponding I/O pin in the PORT group is configured as an output. |

### 23.8.4 Data Direction Toggle

**Name:** DIRTGL
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to toggle the direction of one or more I/O pins, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Set (DIRSET) and Data Direction Clear (DIRCLR) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DIRTGL[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIRTGL[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DIRTGL[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIRTGL[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DIRTGL[31:0]** Port Data Direction Toggle
Writing '0' to a bit has no effect.

Writing '1' to a bit will toggle the corresponding bit in the DIR register, which reverses the direction of the I/O pin.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group will keep its configuration. |
| 1 | The direction of the corresponding I/O pin is toggled. |

### 23.8.5 Data Output Value

**Name:** OUT
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register sets the data output drive value for the individual I/O pins in the PORT.

This register can be manipulated without doing a read-modify-write operation by using the Data Output Value Clear (OUTCLR), Data Output Value Set (OUTSET), and Data Output Value Toggle (OUTTGL) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | OUT[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OUT[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OUT[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OUT[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – OUT[31:0]** PORT Data Output Value
For pins configured as outputs via the Data Direction register (DIR), these bits set the logical output drive level.

For pins configured as inputs via the Data Direction register (DIR) and with pull enabled via the Pull Enable bit in the Pin Configuration register (PINCFG.PULLEN), these bits will set the input pull direction.

| Value | Description |
|---|---|
| 0 | The I/O pin output is driven low, or the input is connected to an internal pull-down. |
| 1 | The I/O pin output is driven high, or the input is connected to an internal pull-up. |

### 23.8.6 Data Output Value Clear

**Name:** OUTCLR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to set one or more output I/O pin drive levels low, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Toggle (OUTTGL) and Data Output Value Set (OUTSET) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | OUTCLR[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OUTCLR[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OUTCLR[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OUTCLR[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – OUTCLR[31:0]** PORT Data Output Value Clear
Writing '0' to a bit has no effect.

Writing '1' to a bit will clear the corresponding bit in the OUT register. Pins configured as outputs via the Data Direction register (DIR) will be set to low output drive level. Pins configured as inputs via DIR and with pull enabled via the Pull Enable bit in the Pin Configuration register (PINCFG.PULLEN) will set the input pull direction to an internal pull-down.

© 2018 Microchip Technology Inc.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group will keep its configuration. |
| 1 | The corresponding I/O pin output is driven low, or the input is connected to an internal pull-down. |

### 23.8.7 Data Output Value Set

**Name:** OUTSET
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to set one or more output I/O pin drive levels high, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Toggle (OUTTGL) and Data Output Value Clear (OUTCLR) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | OUTSET[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OUTSET[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OUTSET[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OUTSET[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – OUTSET[31:0]** PORT Data Output Value Set
Writing '0' to a bit has no effect.

Writing '1' to a bit will set the corresponding bit in the OUT register, which sets the output drive level high for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will set the input pull direction to an internal pull-up.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the group will keep its configuration. |
| 1 | The corresponding I/O pin output is driven high, or the input is connected to an internal pull-up. |

### 23.8.8 Data Output Value Toggle

**Name:** OUTTGL
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to toggle the drive level of one or more output I/O pins, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Set (OUTSET) and Data Output Value Clear (OUTCLR) registers.

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | OUTTGL[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OUTTGL[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OUTTGL[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OUTTGL[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – OUTTGL[31:0]** PORT Data Output Value Toggle
Writing '0' to a bit has no effect.

Writing '1' to a bit will toggle the corresponding bit in the OUT register, which inverts the output drive level for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will toggle the input pull direction.

| Value | Description |
|---|---|
| 0 | The corresponding I/O pin in the PORT group will keep its configuration. |
| 1 | The corresponding OUT bit value is toggled. |

### 23.8.9 Data Input Value

**Name:** IN
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** -

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IN[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IN[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IN[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IN[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IN[31:0]** PORT Data Input Value
These bits are cleared when the corresponding I/O pin input sampler detects a logical low level on the input pin.

These bits are set when the corresponding I/O pin input sampler detects a logical high level on the input pin.

### 23.8.10 Control

**Name:** CTRL
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** PAC Write-Protection

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SAMPLING[31:24] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SAMPLING[23:16] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SAMPLING[15:8] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SAMPLING[7:0] | | | | | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SAMPLING[31:0]** Input Sampling Mode
Configures the input sampling functionality of the I/O pin input samplers, for pins configured as inputs via the Data Direction register (DIR).

The input samplers are enabled and disabled in sub-groups of eight. Thus if any pins within a byte request continuous sampling, all pins in that eight pin sub-group will be continuously sampled.

| Value | Description |
|---|---|
| 0 | On demand sampling of I/O pin is enabled. |
| 1 | Continuous sampling of I/O pin is enabled. |

### 23.8.11 Write Configuration

**Name:** WRCONFIG
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** PAC Write-Protection

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

This write-only register is used to configure several pins simultaneously with the same configuration and/or peripheral multiplexing.

In order to avoid side effect of non-atomic access, 8-bit or 16-bit writes to this register will have no effect. Reading this register always returns zero.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HWSEL | WRPINCFG | | WRPMUX | PMUX[3:0] | | | |
| Access | W | W | | W | W | W | W | W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PULLEN | INEN | PMUXEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PINMASK[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PINMASK[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – HWSEL** Half-Word Select
This bit selects the half-word field of a 32-PORT group to be reconfigured in the atomic write operation.

This bit will always read as zero.

| Value | Description |
|---|---|
| 0 | The lower 16 pins of the PORT group will be configured. |
| 1 | The upper 16 pins of the PORT group will be configured. |

**Bit 30 – WRPINCFG** Write PINCFG
This bit determines whether the atomic write operation will update the Pin Configuration register (PINCFGy) or not for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.

Writing '0' to this bit has no effect.

Writing '1' to this bit updates the configuration of the selected pins with the written WRCONFIG.DRVSTR, WRCONFIG.PULLEN, WRCONFIG.INEN, WRCONFIG.PMUXEN, and WRCONFIG.PINMASK values.

This bit will always read as zero.

| Value | Description |
|---|---|
| 0 | The PINCFGy registers of the selected pins will not be updated. |
| 1 | The PINCFGy registers of the selected pins will be updated. |

**Bit 28 – WRPMUX** Write PMUX
This bit determines whether the atomic write operation will update the Peripheral Multiplexing register (PMUXn) or not for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.

Writing '0' to this bit has no effect.

Writing '1' to this bit updates the pin multiplexer configuration of the selected pins with the written WRCONFIG. PMUX value.

This bit will always read as zero.

| Value | Description |
|---|---|
| 0 | The PMUXn registers of the selected pins will not be updated. |
| 1 | The PMUXn registers of the selected pins will be updated. |

**Bits 27:24 – PMUX[3:0]** Peripheral Multiplexing
These bits determine the new value written to the Peripheral Multiplexing register (PMUXn) for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPMUX bit is set.

These bits will always read as zero.

This bit determines the new value written to PINCFGy.DRVSTR for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

This bit will always read as zero.

**Bit 18 – PULLEN** Pull Enable
This bit determines the new value written to PINCFGy.PULLEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

This bit will always read as zero.

**Bit 17 – INEN** Input Enable
This bit determines the new value written to PINCFGy.INEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

This bit will always read as zero.

**Bit 16 – PMUXEN** Peripheral Multiplexer Enable
This bit determines the new value written to PINCFGy.PMUXEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

This bit will always read as zero.

**Bits 15:0 – PINMASK[15:0]** Pin Mask for Multiple Pin Configuration
These bits select the pins to be configured within the half-word group selected by the WRCONFIG.HWSEL bit.

These bits will always read as zero.

| Value | Description |
|---|---|
| 0 | The configuration of the corresponding I/O pin in the half-word group will be left unchanged. |
| 1 | The configuration of the corresponding I/O pin in the half-word PORT group will be updated. |

### 23.8.12 Peripheral Multiplexing n

**Name:** PMUX
**Offset:** 0x30 + n*0x01 [n=0..15]
**Reset:** 0x00
**Property:** PAC Write-Protection

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

There are up to 16 Peripheral Multiplexing registers in each group, one for every set of two subsequent I/O lines. The n denotes the number of the set of I/O lines.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PMUXO[3:0] | | | | PMUXE[3:0] | | | |
| Access | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:4 – PMUXO[3:0]** Peripheral Multiplexing for Odd-Numbered Pin
These bits select the peripheral function for odd-numbered pins (2*n + 1) of a PORT group, if the corresponding PINCFGy.PMUXEN bit is '1'.

Not all possible values for this selection may be valid. For more details, refer to the *I/O Multiplexing and Considerations*.

| PMUXO[3:0] | Name | Description |
|---|---|---|
| 0x0 | A | Peripheral function A selected |
| 0x1 | B | Peripheral function B selected |
| 0x2 | C | Peripheral function C selected |
| 0x3 | D | Peripheral function D selected |
| 0x4 | E | Peripheral function E selected |
| 0x5 | F | Peripheral function F selected |
| 0x6 | G | Peripheral function G selected |
| 0x7 | H | Peripheral function H selected |
| 0x8 | I | Peripheral function I selected |
| 0x9-0xF | - | Reserved |

**Bits 3:0 – PMUXE[3:0]** Peripheral Multiplexing for Even-Numbered Pin
These bits select the peripheral function for even-numbered pins (2*n) of a PORT group, if the corresponding PINCFGy.PMUXEN bit is '1'.

Not all possible values for this selection may be valid. For more details, refer to the *I/O Multiplexing and Considerations.*

| PMUXE[3:0] | Name | Description |
|---|---|---|
| 0x0 | A | Peripheral function A selected |
| 0x1 | B | Peripheral function B selected |
| 0x2 | C | Peripheral function C selected |
| 0x3 | D | Peripheral function D selected |
| 0x4 | E | Peripheral function E selected |
| 0x5 | F | Peripheral function F selected |
| 0x6 | G | Peripheral function G selected |
| 0x7 | H | Peripheral function H selected |
| 0x8 | I | Peripheral function I selected |
| 0x9-0xF | - | Reserved |

**Related Links**

7. I/O Multiplexing and Considerations

### 23.8.13 Pin Configuration

**Name:** PINCFG
**Offset:** 0x40 + n*0x01 [n=0..31]
**Reset:** 0x00
**Property:** PAC Write-Protection

**Tip:** The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a 0x80 address spacing. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

There are up to 32 Pin Configuration registers in each PORT group, one for each I/O line.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PULLEN | INEN | PMUXEN |
| Access | | | | | | RW | RW | RW |
| Reset | | | | | | 0 | 0 | 0 |

This bit controls the output driver strength of an I/O pin configured as an output.

| Value | Description |
|---|---|
| 0 | Pin drive strength is set to normal drive strength. |
| 1 | Pin drive strength is set to stronger drive strength. |

**Bit 2 – PULLEN** Pull Enable
This bit enables the internal pull-up or pull-down resistor of an I/O pin configured as an input.

| Value | Description |
|---|---|
| 0 | Internal pull resistor is disabled, and the input is in a high-impedance configuration. |
| 1 | Internal pull resistor is enabled, and the input is driven to a defined logic level in the absence of external input. |

**Bit 1 – INEN** Input Enable
This bit controls the input buffer of an I/O pin configured as either an input or output.

Writing a zero to this bit disables the input buffer completely, preventing read-back of the physical pin state when the pin is configured as either an input or output.

| Value | Description |
|---|---|
| 0 | Input buffer for the I/O pin is disabled, and the input value will not be sampled. |
| 1 | Input buffer for the I/O pin is enabled, and the input value will be sampled when required. |

**Bit 0 – PMUXEN** Peripheral Multiplexer Enable
This bit enables or disables the peripheral multiplexer selection set in the Peripheral Multiplexing register (PMUXn) to enable or disable alternative peripheral control over an I/O pin direction and output drive value.

Writing a zero to this bit allows the PORT to control the pad direction via the Data Direction register (DIR) and output drive value via the Data Output Value register (OUT). The peripheral multiplexer value in PMUXn is ignored. Writing '1' to this bit enables the peripheral selection in PMUXn to control the pad. In this configuration, the physical pin state may still be read from the Data Input Value register (IN) if PINCFGn.INEN is set.

| Value | Description |
|---|---|
| 0 | The peripheral multiplexer selection is disabled, and the PORT registers control the direction and output drive value. |
| 1 | The peripheral multiplexer selection is enabled, and the selected peripheral function controls the direction and output drive value. |

# 24. EVSYS – Event System

## 24.1 Overview

The Event System (EVSYS) allows autonomous, low-latency and configurable communication between peripherals.

Several peripherals can be configured to generate and/or respond to signals known as events. The exact condition to generate an event, or the action taken upon receiving an event, is specific to each peripheral. Peripherals that respond to events are called event users. Peripherals that generate events are called event generators. A peripheral can have one or more event generators and can have one or more event users.

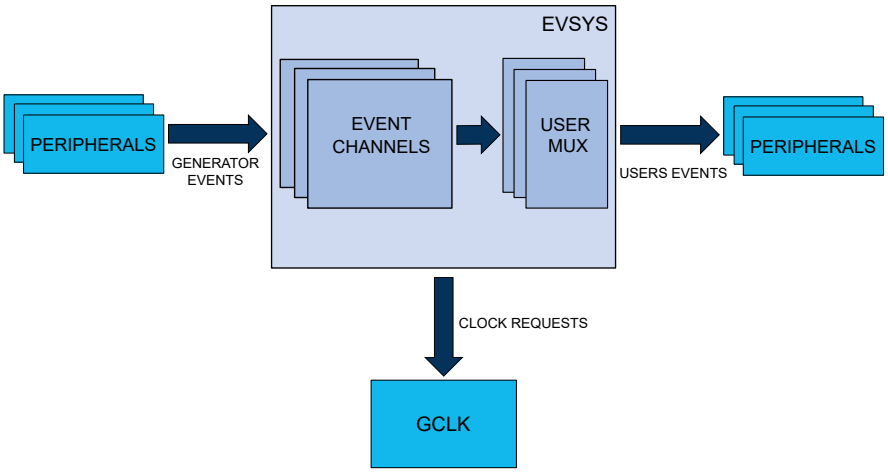
Communication is made without CPU intervention and without consuming system resources such as bus or RAM bandwidth. This reduces the load on the CPU and other system resources, compared to a traditional interrupt-based system.

## 24.2 Features

- 12 configurable event channels, where each channel can:
  - Be connected to any event generator.
  - Provide a pure asynchronous, resynchronized or synchronous path
- 74 event generators.
- 29 event users.
- Configurable edge detector.
- Peripherals can be event generators, event users, or both.
- SleepWalking and interrupt for operation in sleep modes.
- Software event generation.
- Each event user can choose which channel to respond to.

## 24.3 Block Diagram

**Figure 24-1. Event System Block Diagram**



## 24.4 Signal Description

Not applicable.

## 24.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 24.5.1 I/O Lines

Not applicable.

### 24.5.2 Power Management

The EVSYS can be used to wake up the CPU from all sleep modes, even if the clock used by the EVSYS channel and the EVSYS bus clock are disabled. Refer to the *PM – Power Manager* for details on the different sleep modes.

In all sleep modes, although the clock for the EVSYS is stopped, the device still can wake up the EVSYS clock. Some event generators can generate an event when their clocks are stopped.

**Related Links**

16. PM – Power Manager

### 24.5.3 Clocks

The EVSYS bus clock (CLK_EVSYS_APB) can be enabled and disabled in the Main Clock module, and the default state of CLK_EVSYS_APB can be found in *Peripheral Clock Masking*.

Each EVSYS channel has a dedicated generic clock (GCLK_EVSYS_CHANNEL_n). These are used for event detection and propagation for each channel. These clocks must be configured and enabled in the generic clock controller before using the EVSYS. Refer to *GCLK - Generic Clock Controller* for details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

15. GCLK - Generic Clock Controller

### 24.5.4 DMA

Not applicable.

### 24.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the EVSYS interrupts requires the interrupt controller to be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 24.5.6 Events

Not applicable.

### 24.5.7 Debug Operation

When the CPU is halted in Debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging.

### 24.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag Status and Clear register (INTFLAG)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 24.5.9 Analog Connections

Not applicable.

## 24.6 Functional Description

### 24.6.1 Principle of Operation

The Event System consists of several channels which route the internal events from peripherals (generators) to other internal peripherals or IO pins (users). Each event generator can be selected as source for multiple channels, but a channel cannot be set to use multiple event generators at the same time.

### 24.6.2 Basic Operation

#### 24.6.2.1 Initialization

Before enabling events routing within the system, the User Multiplexer (USER) and Channel (CHANNEL) register must be configured. The User Multiplexer (USER) must be configured first.

Configure the User Multiplexer (USER) register:

1. The channel to be connected to a user is written to the Channel bit group (USER.CHANNEL)
2. The user to connect the channel is written to the User bit group (USER.USER)

Configure the Channel (CHANNEL) register:

1. The channel to be configured is written to the Channel Selection bit group (CHANNEL.CHANNEL)
2. The path to be used is written to the Path Selection bit group (CHANNEL.PATH)
3. The type of edge detection to use on the channel is written to the Edge Selection bit group (CHANNEL.EDGSEL)
4. The event generator to be used is written to the Event Generator bit group (CHANNEL.EVGEN)

#### 24.6.2.2 Enabling, Disabling and Resetting

The EVSYS is always enabled.

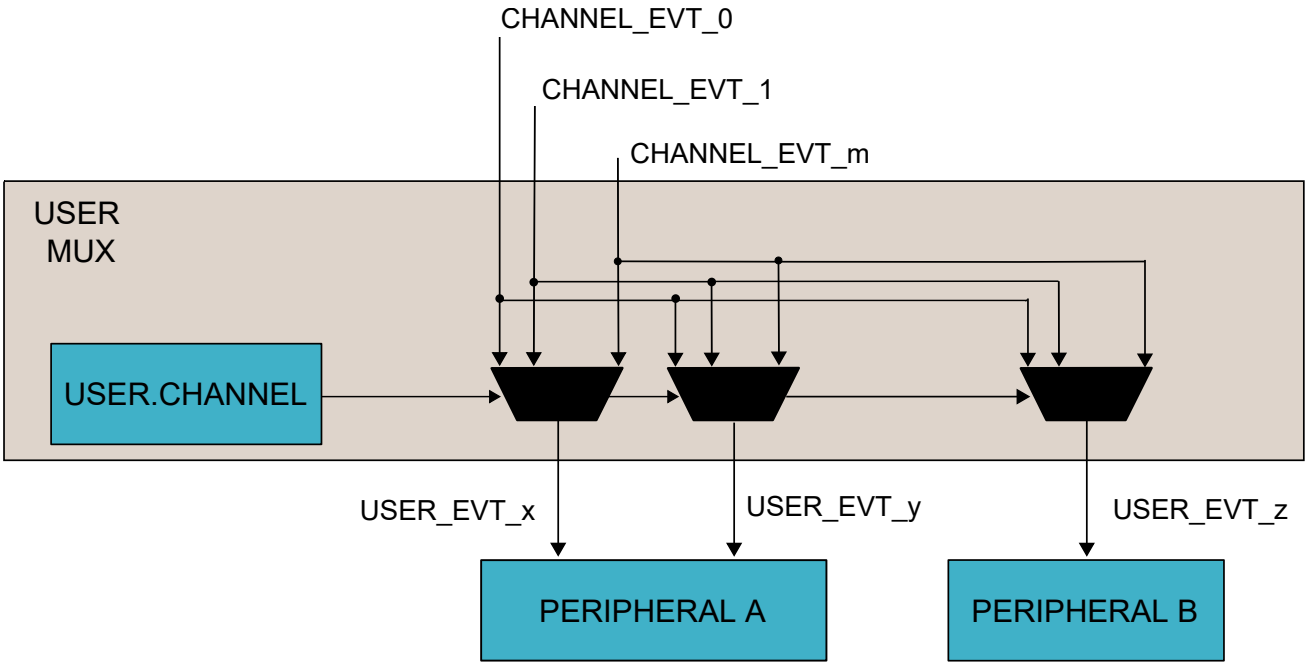
The EVSYS is reset by writing a ‘1’ to the Software Reset bit in the Control register (CTRL.SWRST). All registers in the EVSYS will be reset to their initial state and all ongoing events will be canceled. Refer to CTRL.SWRST register for details.

#### 24.6.2.3 User Multiplexer Setup

The user multiplexer defines the channel to be connected to which event user. Each user multiplexer is dedicated to one event user. A user multiplexer receives all event channels output and must be configured to select one of these channels, as shown in the next figure. The channel is selected with the Channel bit group in the USER register (USER.CHANNEL). The user multiplexer must always be configured before the channel. A full list of selectable users can be found in the User Multiplexer register (USER) description. Refer to UserList for details.

To configure a user multiplexer, the USER register must be written in a single 16-bit write. It is possible to read out the configuration of a user by first selecting the user by writing to USER.USER using an 8-bit write and then performing a read of the 16-bit USER register.

**Figure 24-2. User MUX**




#### 24.6.2.4 Channel Setup

An event channel can select one event from a list of event generators. Depending on configuration, the selected event could be synchronized, resynchronized or asynchronously sent to the users. When synchronization or resynchronization is required, the channel includes an internal edge detector, allowing the Event System to generate internal events when rising, falling or both edges are detected on the selected event generator. An event channel is able to generate internal events for the specific software commands. All these configurations are available in the Channel register (CHANNEL).

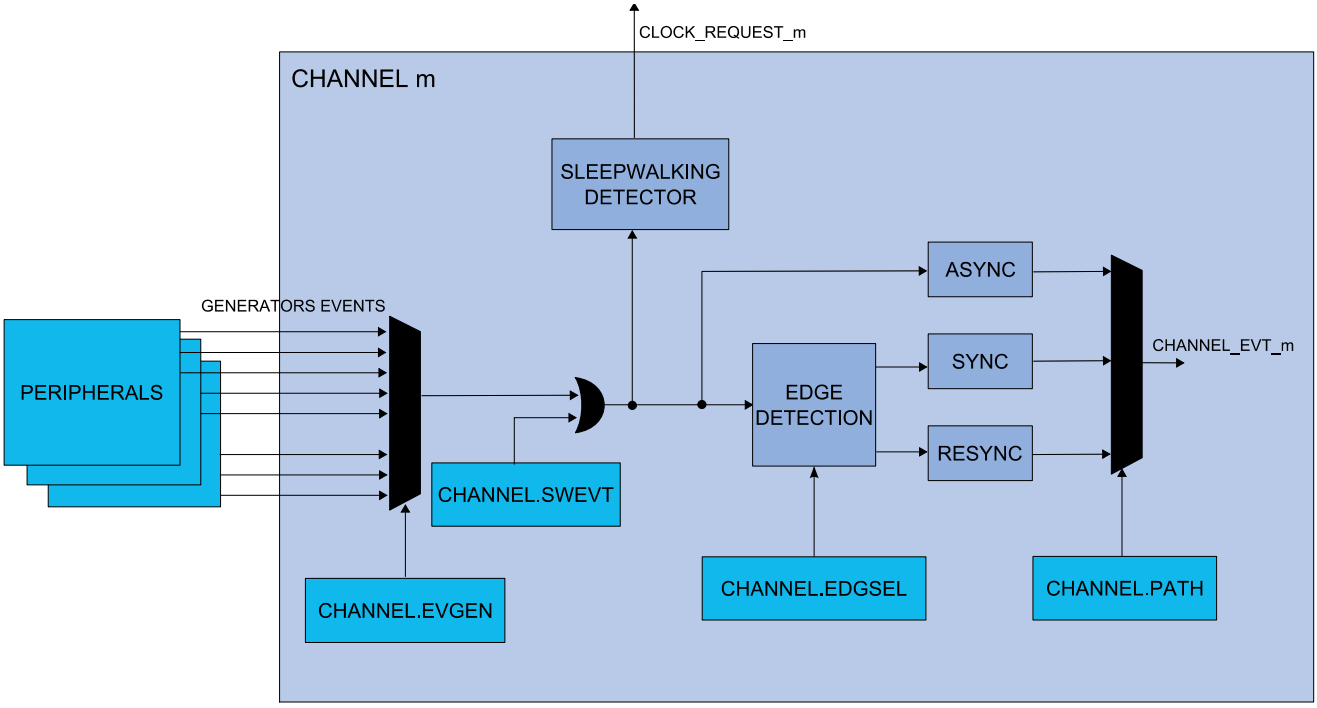
To configure a channel, the Channel register must be written in a single 32-bit write. It is possible to read out the configuration of a channel by first selecting the channel by writing to CHANNEL.CHANNEL using a, 8-bit write, and then performing a read of the CHANNEL register.

#### 24.6.2.5 Channel Path

There are three different ways to propagate the event provided by an event generator:

- Asynchronous path
- Synchronous path
- Resynchronized path

**Figure 24-3. Channel**




The path is selected by writing to the Path Selection bit group in the Channel register (CHANNEL.PATH).

#### 24.6.2.5.1 Asynchronous Path

When using the asynchronous path, the events are propagated from the event generator to the event user without intervention from the Event System. The GCLK for this channel (GCLK_EVSYS_CHANNEL_n) is not mandatory, meaning that an event will be propagated to the user without any clock latency.

When the asynchronous path is selected, the channel cannot generate any interrupts, and the Channel Status register (CHSTATUS) is always zero. No edge detection is available; this must be handled in the event user. When the event generator and the event user share the same generic clock, using the asynchronous path will propagate the event with the least amount of latency.

#### 24.6.2.5.2 Synchronous Path

The synchronous path should be used when the event generator and the event channel share the same generator for the generic clock and also if event user supports synchronous path. If event user doesn't support synchronous path, asynchronous path has to be selected. If they do not share the same clock, a logic change from the event generator to the event channel might not be detected in the channel, which means that the event will not be propagated to the event user. For details on generic clock generators, refer to *GCLK - Generic Clock Controller*.

When using the synchronous path, the channel is able to generate interrupts. The channel status bits in the Channel Status register (CHSTATUS) are also updated and available for use.

If the Generic Clocks Request bit in the Control register (CTRL.GCLKREQ) is zero, the channel operates in SleepWalking mode and request the configured generic clock only when an event is to be propagated through the channel. If CTRL.GCLKREQ is one, the generic clock will always be on for the configured channel.

**Related Links**

15. GCLK - Generic Clock Controller

#### 24.6.2.5.3 Resynchronized Path

The resynchronized path should be used when the event generator and the event channel do not share the same generic clock generator. When the resynchronized path is used, resynchronization of the event from the event generator is done in the channel. For details on generic clock generators, refer to *GCLK - Generic Clock Controller*.

When the resynchronized path is used, the channel is able to generate interrupts. The channel status bits in the Channel Status register (CHSTATUS) are also updated and available for use.

If the Generic Clocks Request bit in the Control register is zero (CTRL.GCLKREQ=0), the channel operates in SleepWalking mode and requests the configured generic clock only when an event is to be propagated through the channel. If CTRL.GCLKREQ=1 , the generic clock will always be on for the configured channel.

**Related Links**

15. GCLK - Generic Clock Controller

### 24.6.2.6 Edge Detection

When synchronous or resynchronized paths are used, edge detection must be used. The event system can perform edge detection in three different ways:

- Generate an event only on the rising edge
- Generate an event only on the falling edge
- Generate an event on rising and falling edges

Edge detection is selected by writing to the Edge Selection bit group in the Channel register (CHANNEL.EDGSEL).

If the generator event is a pulse, both edges cannot be selected. Use the rising edge or falling edge detection methods, depending on the generator event default level.

### 24.6.2.7 Event Generators

Each event channel can receive the events form all event generators. All event generators are listed in the statement of CHANNEL.EVGEN. For details on event generation, refer to the corresponding module chapter. The channel event generator is selected by the Event Generator bit group in the Channel register (CHANNEL.EVGEN). By default, the channels are not connected to any event generators (ie, CHANNEL.EVGEN = 0)

### 24.6.2.8 Channel Status

The Channel Status register (CHSTATUS) shows the status of the channels when using a synchronous or resynchronized path. There are two different status bits in CHSTATUS for each of the available channels:

- The CHSTATUS.CHBUSYn bit will be set when an event on the corresponding channel n has not been handled by all event users connected to that channel.
- The CHSTATUS.USRRDYn bit will be set when all event users connected to the corresponding channel are ready to handle incoming events on that channel.

### 24.6.2.9 Software Event

A software event can be initiated on a channel by setting the Software Event bit in the Channel register (CHANNEL.SWEVT) to ‘1’ at the same time as writing the Channel bits (CHANNEL.CHANNEL). This will generate a software event on the selected channel.

The software event can be used for application debugging, and functions like any event generator. To use the software event, the event path must be configured to either a synchronous path or resynchronized path (CHANNEL.PATH = 0x0 or 0x1), edge detection must be configured to rising-edge detection (CHANNEL.EDGSEL= 0x1) and the Generic Clock Request bit must be set to '1' (CTRL.GCLKREQ=0x1).

### 24.6.3 Interrupts

The EVSYS has the following interrupt sources:

- Overrun Channel n (OVRn): for details, refer to *The Overrun Channel n Interrupt* section.
- Event Detected Channel n (EVDn): for details, refer to *The Event Detected Channel n Interrupt* section.

These interrupts events are asynchronous wake-up sources. See *Sleep Mode Controller*.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt is issued. Each interrupt event can be individually enabled by setting a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by setting a '1' to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt event is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt event works until the interrupt flag is cleared, the interrupt is disabled, or the Event System is reset. See INTFLAG for details on how to clear interrupt flags.

All interrupt events from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. Refer to the *Nested Vector Interrupt Controller* for details. The event user must read the INTFLAG register to determine what the interrupt condition is.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

16.6.1.3 Sleep Mode Controller

#### 24.6.3.1 The Overrun Channel n Interrupt

The Overrun Channel n interrupt flag in the Interrupt Flag Status and Clear register (CHINTFLAGn.OVR) will be set, and the optional interrupt will be generated in the following cases:

- One or more event users on channel n is not ready when there is a new event.
- An event occurs when the previous event on channel m has not been handled by all event users connected to that channel.

The flag will only be set when using resynchronized paths. In the case of asynchronous path, the CHINTFLAGn.OVR is always read as zero.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 24.6.3.2 The Event Detected Channel n Interrupt

The Event Detected Channel n interrupt flag in the Interrupt Flag Status and Clear register (CHINTFLAGn.EVD) is set when an event coming from the event generator configured on channel n is detected.

The flag will only be set when using a resynchronized path. In the case of asynchronous path, the CHINTFLAGn.EVD is always zero.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 24.6.4 Sleep Mode Operation

The EVSYS can generate interrupts to wake up the device from any sleep mode.

Some event generators can generate an event when the system clock is stopped. The generic clock (GCLK_EVSYS_CHANNELx) for this channel will be restarted if the channel uses a synchronized path or a resynchronized path, without waking the system from sleep. The clock remains active only as long as necessary to handle the event. After the event has been handled, the clock will be turned off and the system will remain in the original sleep mode. This is known as SleepWalking. When an asynchronous path is used, there is no need for the clock to be activated for the event to be propagated to the user.

On a software reset, all registers are set to their reset values and any ongoing events are canceled.

## 24.7 Register Summary

**Table 24-1. Event System Register Summary**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRL | 7:0 | | | | GCLKREQ | | | | SWRST |
| 0x01<br>...<br>0x03 | Reserved | | | | | | | | | |
| 0x04 | CHANNEL | 7:0 | | | | | CHANNEL[3:0] | | | |
| 0x05 | | 15:8 | | | | | | | | SWEVT |
| 0x06 | | 23:16 | | EVGEN[6:0] | | | | | | |
| 0x07 | | 31:24 | | | | | EDGSEL[1:0] | | PATH[1:0] | |
| 0x08 | USER | 7:0 | | | | USER[4:0] | | | | |
| 0x09 | | 15:8 | | | | CHANNEL[4:0] | | | | |
| 0x0A | Reserved | | | | | | | | | |
| 0x0B | Reserved | | | | | | | | | |
| 0x0C | CHSTATUS | 7:0 | USRRDY7 | USRRDY6 | USRRDY5 | USRRDY4 | USRRDY3 | USRRDY2 | USRRDY1 | USRRDY0 |
| 0x0D | | 15:8 | CHBUSY7 | CHBUSY6 | CHBUSY5 | CHBUSY4 | CHBUSY3 | CHBUSY2 | CHBUSY1 | CHBUSY0 |
| 0x0E | | 23:16 | | | | | USRRDY11 | USRRDY10 | USRRDY9 | USRRDY8 |
| 0x0F | | 31:24 | | | | | CHBUSY11 | CHBUSY10 | CHBUSY9 | CHBUSY8 |
| 0x10 | INTENCLR | 7:0 | OVR7 | OVR6 | OVR5 | OVR4 | OVR3 | OVR2 | OVR1 | OVR0 |
| 0x11 | | 15:8 | EVD7 | EVD6 | EVD5 | EVD4 | EVD3 | EVD2 | EVD1 | EVD0 |
| 0x12 | | 23:16 | | | | | OVR11 | OVR10 | OVR9 | OVR8 |
| 0x13 | | 31:24 | | | | | EVD11 | EVD10 | EVD9 | EVD8 |
| 0x14 | INTENSET | 7:0 | OVR7 | OVR6 | OVR5 | OVR4 | OVR3 | OVR2 | OVR1 | OVR0 |
| 0x15 | | 15:8 | EVD7 | EVD6 | EVD5 | EVD4 | EVD3 | EVD2 | EVD1 | EVD0 |
| 0x16 | | 23:16 | | | | | OVR11 | OVR10 | OVR9 | OVR8 |
| 0x17 | | 31:24 | | | | | EVD11 | EVD10 | EVD9 | EVD8 |
| 0x18 | INTFLAG | 7:0 | OVR7 | OVR6 | OVR5 | OVR4 | OVR3 | OVR2 | OVR1 | OVR0 |
| 0x19 | | 15:8 | EVD7 | EVD6 | EVD5 | EVD4 | EVD3 | EVD2 | EVD1 | EVD0 |
| 0x1A | | 23:16 | | | | | OVR11 | OVR10 | OVR9 | OVR8 |
| 0x1B | | 31:24 | | | | | EVD11 | EVD10 | EVD9 | EVD8 |

## 24.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Refer to 24.5.8 Register Access Protection.

### 24.8.1 Control

**Name:** CTRL
**Offset:** 0x00
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | GCLKREQ | | | | SWRST |
| Access | | | | R/W | | | | W |
| Reset | | | | 0 | | | | 0 |

**Bit 4 – GCLKREQ** Generic Clock Requests
This bit is used to determine whether the generic clocks used for the different channels should be on all the time or only when an event needs the generic clock. Events propagated through asynchronous paths will not need a generic clock.

| Value | Description |
|---|---|
| 0 | Generic clock is requested and turned on only if an event is detected. |
| 1 | Generic clock for a channel is always on. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the EVSYS to their initial state.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

**Note:** Before applying a Software Reset it is recommended to disable the event generators.

### 24.8.2 Channel

**Name:** CHANNEL
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EDGSEL[1:0] | | PATH[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | EVGEN[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWEVT |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHANNEL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 27:26 – EDGSEL[1:0]** Edge Detection Selection
These bits set the type of edge detection to be used on the channel.

These bits must be written to zero when using the asynchronous path.

| EDGSEL[1:0] | Name | Description |
|---|---|---|
| 0x0 | NO_EVT_OUTPUT | No event output when using the resynchronized or synchronous path |
| 0x1 | RISING_EDGE | Event detection only on the rising edge of the signal from the event generator when using the resynchronized or synchronous path |
| 0x2 | FALLING_EDGE | Event detection only on the falling edge of the signal from the event generator when using the resynchronized or synchronous path |
| 0x3 | BOTH_EDGES | Event detection on rising and falling edges of the signal from the event generator when using the resynchronized or synchronous path |

**Bits 25:24 – PATH[1:0]** Path Selection
These bits are used to choose the path to be used by the selected channel.

The path choice can be limited by the channel source.

| PATH[1:0] | Name | Description |
|---|---|---|
| 0x0 | SYNCHRONOUS | Synchronous path |
| 0x1 | RESYNCHRONIZED | Resynchronized path |
| 0x2 | ASYNCHRONOUS | Asynchronous path |
| 0x3 | | Reserved |

**Bits 22:16 – EVGEN[6:0]** Event Generator Selection
These bits are used to choose which event generator to connect to the selected channel.

| Value | Event Generator | Description |
|---|---|---|
| 0x00 | NONE | No event generator selected |
| 0x01 | RTC CMP0 | Compare 0 (mode 0 and 1) or Alarm 0 (mode 2) |
| 0x02 | RTC CMP1 | Compare 1 |
| 0x03 | RTC OVF | Overflow |
| 0x04 | RTC PER0 | Period 0 |
| 0x05 | RTC PER1 | Period 1 |
| 0x06 | RTC PER2 | Period 2 |
| 0x07 | RTC PER3 | Period 3 |
| 0x08 | RTC PER4 | Period 4 |
| 0x09 | RTC PER5 | Period 5 |
| 0x0A | RTC PER6 | Period 6 |
| 0x0B | RTC PER7 | Period 7 |
| 0x0C | EIC EXTINT0 | External Interrupt 0 |
| 0x0D | EIC EXTINT1 | External Interrupt 1 |
| 0x0E | EIC EXTINT2 | External Interrupt 2 |
| 0x0F | EIC EXTINT3 | External Interrupt 3 |
| 0x10 | EIC EXTINT4 | External Interrupt 4 |
| 0x11 | EIC EXTINT5 | External Interrupt 5 |
| 0x12 | EIC EXTINT6 | External Interrupt 6 |
| 0x13 | EIC EXTINT7 | External Interrupt 7 |
| 0x14 | EIC EXTINT8 | External Interrupt 8 |
| 0x15 | EIC EXTINT9 | External Interrupt 9 |
| 0x16 | EIC EXTINT10 | External Interrupt 10 |
| 0x17 | EIC EXTINT11 | External Interrupt 11 |

© 2018 Microchip Technology Inc.

| Value | Event Generator | Description |
|---|---|---|
| 0x18 | EIC EXTINT12 | External Interrupt 12 |
| 0x19 | EIC EXTINT13 | External Interrupt 13 |
| 0x1A | EIC EXTINT14 | External Interrupt 14 |
| 0x1B | EIC EXTINT15 | External Interrupt 15 |
| 0x1C | Reserved | |
| 0x1D | Reserved | |
| 0x1E | DMAC CH0 | Channel 0 |
| 0x1F | DMAC CH1 | Channel 1 |
| 0x20 | DMAC CH2 | Channel 2 |
| 0x21 | DMAC CH3 | Channel 3 |
| 0x22 | TCC0 OVF | Overflow |
| 0x23 | TCC0 TRG | Trig |
| 0x24 | TCC0 CNT | Counter |
| 0x25 | TCC0_MCX0 | Match/Capture 0 |
| 0x26 | TCC0_MCX1 | Match/Capture 1 |
| 0x27 | TCC0_MCX2 | Match/Capture 2 |
| 0x28 | TCC0_MCX3 | Match/Capture 3 |
| 0x29 | TCC1 OVF | Overflow |
| 0x2A | TCC1 TRG | Trig |
| 0x2B | TCC1 CNT | Counter |
| 0x2C | TCC1_MCX0 | Match/Capture 0 |
| 0x2D | TCC1_MCX1 | Match/Capture 1 |
| 0x2E | TCC2 OVF | Overflow |
| 0x2F | TCC2 TRG | Trig |
| 0x30 | TCC2 CNT | Counter |
| 0x31 | TCC2_MCX0 | Match/Capture 0 |
| 0x32 | TCC2_MCX1 | Match/Capture 1 |
| 0x33 | TC3 OVF | Overflow/Underflow |
| 0x34 | TC3 MC0 | Match/Capture 0 |
| 0x35 | TC3 MC1 | Match/Capture 1 |
| 0x36 | TC4 OVF | Overflow/Underflow |
| 0x37 | TC4 MC0 | Match/Capture 0 |

© 2018 Microchip Technology Inc.

| Value | Event Generator | Description |
|---|---|---|
| 0x38 | TC4 MC1 | Match/Capture 1 |
| 0x39 | TC5 OVF | Overflow/Underflow |
| 0x3A | TC5 MC0 | Match/Capture 0 |
| 0x3B | TC5 MC1 | Match/Capture 1 |
| 0x3C | TC6 OVF | Overflow/Underflow |
| 0x3D | TC6 MC0 | Match/Capture 0 |
| 0x3E | TC6 MC1 | Match/Capture 1 |
| 0x3F | TC7 OVF | Overflow/Underflow |
| 0x40 | TC7 MC0 | Match/Capture 0 |
| 0x41 | TC7 MC1 | Match/Capture 1 |
| 0x42 | ADC RESRDY | Result Ready |
| 0x43 | ADC WINMON | Window Monitor |
| 0x44 | AC COMP0 | Comparator 0 |
| 0x45 | AC COMP1 | Comparator 1 |
| 0x46 | AC WIN0 | Window 0 |
| 0x47 | DAC EMPTY | Data Buffer Empty |
| 0x48 | PTC EOC | End of Conversion |
| 0x49 | PTC WCOMP | Window Comparator |
| 0x4A-0x7F | Reserved | |

**Bit 8 – SWEVT** Software Event
This bit is used to insert a software event on the channel selected by the CHANNEL.CHANNEL bit group.

This bit has the same behavior similar to an event.

This bit must be written together with CHANNEL.CHANNELusing a 16-bit write.

Writing a zero to this bit has no effect.

Writing a one to this bit will trigger a software event for the corresponding channel.

This bit will always return zero when read.

**Bits 3:0 – CHANNEL[3:0]** Channel Selection
These bits are used to select the channel to be set up or read from.

### 24.8.3 User Multiplexer

**Name:** USER
**Offset:** 0x08
**Reset:** 0x0000
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CHANNEL[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | USER[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 12:8 – CHANNEL[4:0]** Channel Event Selection
These bits are used to select the channel to connect to the event user.

Note that to select channel n, the value (n+1) must be written to the USER.CHANNEL bit group.

| CHANNEL[4:0] | Channel Number |
|---|---|
| 0x0 | No Channel Output Selected |
| 0x1-0xC | Channel n-1 selected |
| 0xD-0xFF | Reserved |

**Bits 4:0 – USER[4:0]** User Multiplexer Selection
These bits select the event user to be configured with a channel, or the event user to read the channel value from.

**Table 24-2. User Multiplexer Selection**

| USER[7:0] | User Multiplexer | Description | Path Type |
|---|---|---|---|
| 0x00 | DMAC CH0 | Channel 0 | Resynchronized path only |
| 0x01 | DMAC CH1 | Channel 1 | Resynchronized path only |
| 0x02 | DMAC CH2 | Channel 2 | Resynchronized path only |
| 0x03 | DMAC CH3 | Channel 3 | Resynchronized path only |
| 0x04 | TCC0 EV0 | | Asynchronous, synchronous and resynchronized paths |
| 0x05 | TCC0 EV1 | | Asynchronous, synchronous and resynchronized paths |
| 0x06 | TCC0 MC0 | Match/Capture 0 | Asynchronous, synchronous and resynchronized paths |

| USER[7:0] | User Multiplexer | Description | Path Type |
|---|---|---|---|
| 0x07 | TCC0 MC1 | Match/Capture 1 | Asynchronous, synchronous and resynchronized paths |
| 0x08 | TCC0 MC2 | Match/Capture 2 | Asynchronous, synchronous and resynchronized paths |
| 0x09 | TCC0 MC3 | Match/Capture 3 | Asynchronous, synchronous and resynchronized paths |
| 0x0A | TCC1 EV0 | | Asynchronous, synchronous and resynchronized paths |
| 0x0B | TCC1 EV1 | | Asynchronous, synchronous and resynchronized paths |
| 0x0C | TCC1 MC0 | Match/Capture 0 | Asynchronous, synchronous and resynchronized paths |
| 0x0D | TCC1 MC1 | Match/Capture 1 | Asynchronous, synchronous and resynchronized paths |
| 0x0E | TCC2 EV0 | | Asynchronous, synchronous and resynchronized paths |
| 0x0F | TCC2 EV1 | | Asynchronous, synchronous and resynchronized paths |
| 0x10 | TCC2 MC0 | Match/Capture 0 | Asynchronous, synchronous and resynchronized paths |
| 0x11 | TCC2 MC1 | Match/Capture 1 | Asynchronous, synchronous and resynchronized paths |
| 0x12 | TC3 | | Asynchronous, synchronous and resynchronized paths |
| 0x13 | TC4 | | Asynchronous, synchronous and resynchronized paths |
| 0x14 | TC5 | | Asynchronous, synchronous and resynchronized paths |
| 0x15 | TC6 | | Asynchronous, synchronous and resynchronized paths |
| 0x16 | TC7 | | Asynchronous, synchronous and resynchronized paths |
| 0x17 | ADC START | ADC start conversion | Asynchronous path only |
| 0x18 | ADC SYNC | Flush ADC | Asynchronous path only |
| 0x19 | AC COMP0 | Start comparator 0 | Asynchronous path only |
| 0x1A | AC COMP1 | Start comparator 1 | Asynchronous path only |
| 0x1B | DAC START | DAC start conversion | Asynchronous path only |

| USER[7:0] | User Multiplexer | Description | Path Type |
|---|---|---|---|
| 0x1C | PTC STCONV | PTC start conversion | Asynchronous path only |
| 0x1D-0x1F | Reserved | | Reserved |

### 24.8.4 Channel Status

**Name:** CHSTATUS
**Offset:** 0x0C
**Reset:** 0x000F00FF
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | USRRDYn | USRRDYn | USRRDYn | USRRDYn |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn | CHBUSYn |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USRRDYn | USRRDYn | USRRDYn | USRRDYn | USRRDYn | USRRDYn | USRRDYn | USRRDYn |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27,26,25,24,15,14,13,12,11,10,9,8 – CHBUSYn** Channel n Busy [n=11..0]
This bit is cleared when channel n is idle

This bit is set if an event on channel n has not been handled by all event users connected to channel n.

**Bits 19,18,17,16,7,6,5,4,3,2,1,0 – USRRDYn** Channel n User Ready [n=11..0]
This bit is cleared when at least one of the event users connected to the channel is not ready.

This bit is set when all event users connected to channel n are ready to handle incoming events on channel n.

### 24.8.5 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EVDn | EVDn | EVDn | EVDn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRn | OVRn | OVRn | OVRn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27,26,25,24,15,14,13,12,11,10,9,8 – EVDn** Channel n Event Detection Interrupt Enable [n=11..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Event Detected Channel n Interrupt Enable bit, which disables the Event Detected Channel n interrupt.

| Value | Description |
|---|---|
| 0 | The Event Detected Channel n interrupt is disabled. |
| 1 | The Event Detected Channel n interrupt is enabled. |

**Bits 19,18,17,16,7,6,5,4,3,2,1,0 – OVRn** Channel n Overrun Interrupt Enable [n=11..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overrun Channel n Interrupt Enable bit, which disables the Overrun Channel n interrupt.

| Value | Description |
|---|---|
| 0 | The Overrun Channel n interrupt is disabled. |
| 1 | The Overrun Channel n interrupt is enabled. |

### 24.8.6 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Write-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EVDn | EVDn | EVDn | EVDn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRn | OVRn | OVRn | OVRn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27,26,25,24,15,14,13,12,11,10,9,8 – EVDn** Channel n Event Detection Interrupt Enable [n=11..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Event Detected Channel n Interrupt Enable bit, which enables the Event Detected Channel n interrupt.

| Value | Description |
|---|---|
| 0 | The Event Detected Channel n interrupt is disabled. |
| 1 | The Event Detected Channel n interrupt is enabled. |

**Bits 19,18,17,16,7,6,5,4,3,2,1,0 – OVRn** Channel n Overrun Interrupt Enable [n=11..0]
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Overrun Channel n Interrupt Enable bit, which enables the Overrun Channel n interrupt.

| Value | Description |
|---|---|
| 0 | The Overrun Channel n interrupt is disabled. |
| 1 | The Overrun Channel n interrupt is enabled. |

### 24.8.7 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EVDn | EVDn | EVDn | EVDn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRn | OVRn | OVRn | OVRn |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn | EVDn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn | OVRn |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27,26,25,24,15,14,13,12,11,10,9,8 – EVDn** Channel n Event Detection [n=11..0]
This flag is set on the next CLK_EVSYS_APB cycle when an event is being propagated through the channel, and an interrupt request will be generated if INTENCLR/SET.EVDn is one.

When the event channel path is asynchronous, the EVDn interrupt flag will not be set.

Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Event Detected Channel n interrupt flag.

**Bits 19,18,17,16,7,6,5,4,3,2,1,0 – OVRn** Channel n Overrun [n=11..0]
This flag is set on the next CLK_EVSYS cycle after an overrun channel condition occurs, and an interrupt request will be generated if INTENCLR/SET.OVRn is one.

When the event channel path is asynchronous, the OVRn interrupt flag will not be set.

Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overrun Channel n interrupt flag.

# 25. SERCOM – Serial Communication Interface

## 25.1 Overview

There are up to six instances of the serial communication interface (SERCOM) peripheral.

A SERCOM can be configured to support a number of modes: $I^2C$, SPI, and USART. When an instance of SERCOM is configured and enabled, all of the resources of that SERCOM instance will be dedicated to the selected mode.

The SERCOM serial engine consists of a transmitter and receiver, baud-rate generator and address matching functionality. It can use the internal generic clock or an external clock. Using an external clock allows the SERCOM to be operated in all Sleep modes.

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit

## 25.2 Features

- Interface for configuring into one of the following:
  - Inter-Integrated Circuit ($I^2C$) Two-wire Serial Interface
  - System Management Bus (SMBus™) compatible
  - Serial Peripheral Interface (SPI)
  - Universal Synchronous/Asynchronous Receiver/Transmitter (USART)
- Single transmit buffer and double receive buffer
- Baud-rate generator
- Address match/mask logic
- Operational in all Sleep modes with an external clock source
- Can be used with DMA
- Up to 16-bytes internal FIFO

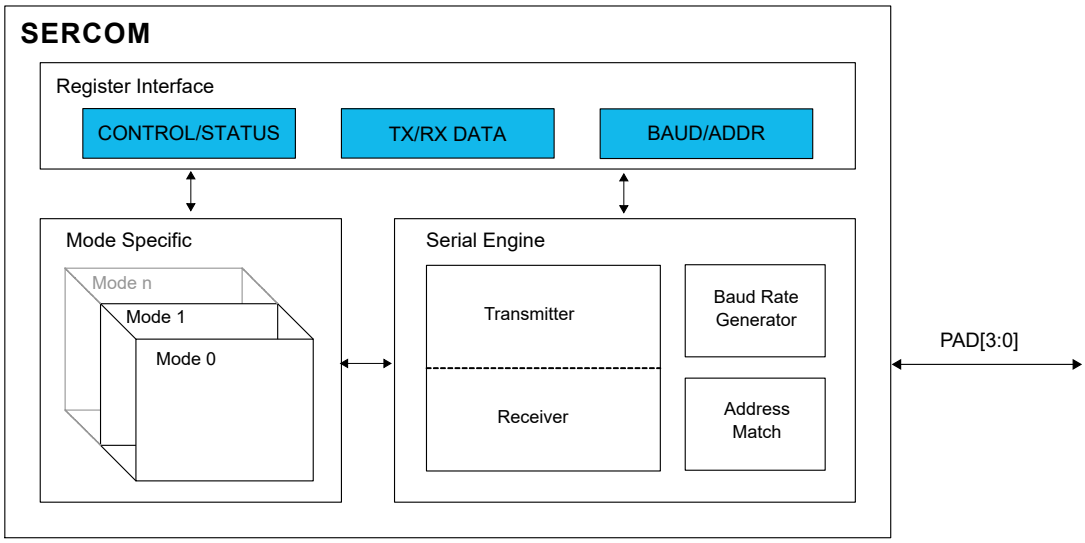
See the Related Links for full feature lists of the interface configurations.

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit

## 25.3 Block Diagram

**Figure 25-1. SERCOM Block Diagram**




## 25.4 Signal Description

See the respective SERCOM mode chapters for details.

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit

## 25.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 25.5.1 I/O Lines

Using the SERCOM I/O lines requires the I/O pins to be configured using port configuration (PORT).

The SERCOM has four internal pads, PAD[3:0], and the signals from I2C, SPI and USART are routed through these SERCOM pads via a multiplexer. The configuration of the multiplexer is available from the different SERCOM modes. Refer to the mode specific chapters for details.

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit
23. PORT - I/O Pin Controller
26.3 Block Diagram

### 25.5.2 Power Management

The SERCOM can operate in any Sleep mode provided the selected clock source is running. SERCOM interrupts can be configured to wake the device from Sleep modes.

**Related Links**

16. PM – Power Manager

### 25.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the Power Manager. Refer to *Peripheral Clock Masking* for details and default status of this clock.

The SERCOM uses two generic clocks: GCLK_SERCOMx_CORE and GCLK_SERCOMx_SLOW. The core clock (GCLK_SERCOMx_CORE) is required to clock the SERCOM while working as a master. The slow clock (GCLK_SERCOMx_SLOW) is only required for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the Generic Clock Controller (GCLK) before using the SERCOM.

The generic clocks are asynchronous to the user interface clock (CLK_SERCOMx_APB). Due to this asynchronicity, writing to certain registers will require synchronization between the clock domains. Refer to 25.6.8 Synchronization for details.

**Related Links**

15. GCLK - Generic Clock Controller

16.6.2.6 Peripheral Clock Masking

### 25.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before the SERCOM DMA requests are used.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 25.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller (NVIC). The NVIC must be configured before the SERCOM interrupts are used.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 25.5.6 Events

Not applicable.

### 25.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

### 25.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except for the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)

- Data register (DATA)
- Address register (ADDR)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 25.5.9 Analog Connections

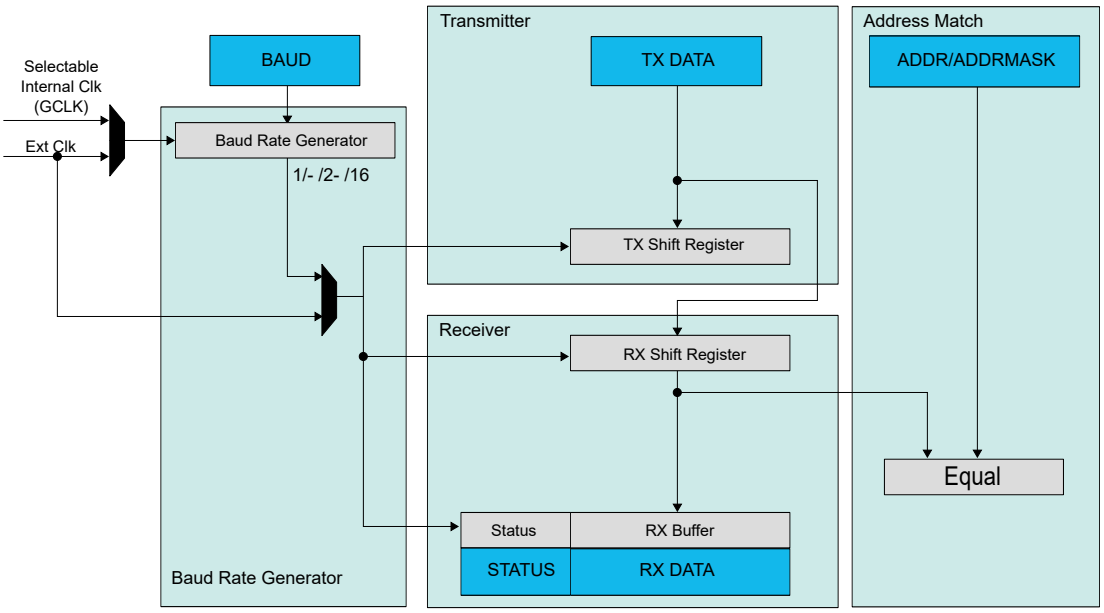
Not applicable.

## 25.6 Functional Description

### 25.6.1 Principle of Operation

The basic structure of the SERCOM serial engine is shown in Figure 25-2. Labels in capital letters are synchronous to the system clock and accessible by the CPU; labels in lowercase letters can be configured to run on the GCLK_SERCOMx_CORE clock or an external clock.

**Figure 25-2. SERCOM Serial Engine**




The transmitter consists of a single write buffer and a shift register.

The receiver consists of a one-level ($I^2C$), two-level (USART, SPI) receive buffer and a shift register.

The baud-rate generator is capable of running on the GCLK_SERCOMx_CORE clock or an external clock.

Address matching logic is included for SPI and $I^2C$ operation.

### 25.6.2 Basic Operation

#### 25.6.2.1 Initialization

The SERCOM must be configured to the desired mode by writing the Operating Mode bits in the Control A register (CTRLA.MODE). Refer to table SERCOM Modes for details.

**Table 25-1. SERCOM Modes**

| CTRLA.MODE | Description |
|---|---|
| 0x0 | USART with external clock |
| 0x1 | USART with internal clock |
| 0x2 | SPI in slave operation |
| 0x3 | SPI in master operation |
| 0x4 | $I^2C$ slave operation |
| 0x5 | $I^2C$ master operation |
| 0x6-0x7 | Reserved |

For further initialization information, see the respective SERCOM mode chapters:

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit

#### 25.6.2.2 Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

Refer to the CTRLA register description for details.

#### 25.6.2.3 Clock Generation – Baud-Rate Generator

The baud-rate generator, as shown in Figure 25-3, generates internal clocks for asynchronous and synchronous communication. The output frequency ($f_{BAUD}$) is determined by the Baud register (BAUD) setting and the baud reference frequency ($f_{ref}$). The baud reference clock is the serial engine clock, and it can be internal or external.

For asynchronous communication, the /16 (divide-by-16) output is used when transmitting, whereas the /1 (divide-by-1) output is used while receiving.

For synchronous communication, the /2 (divide-by-2) output is used.

This functionality is automatically configured, depending on the selected operating mode.

**Figure 25-3. Baud Rate Generator**

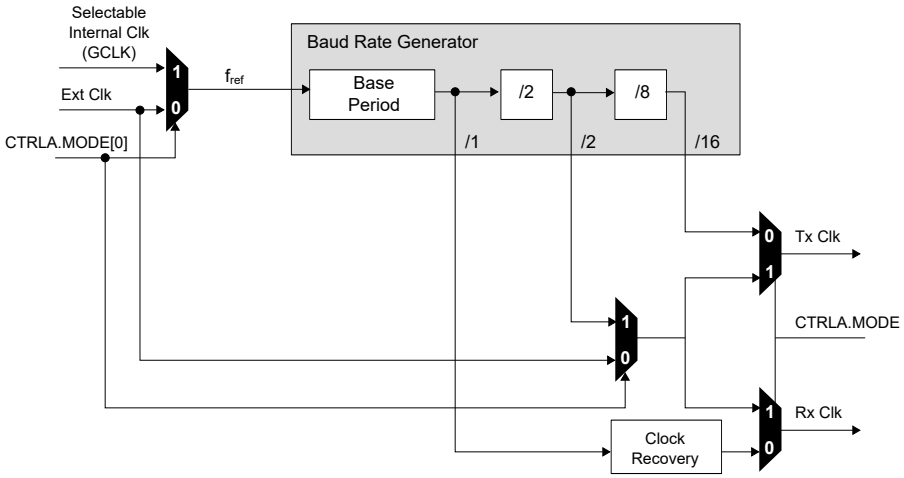



Table 25-2 contains equations for the baud rate (in bits per second) and the BAUD register value for each operating mode.

For asynchronous operation, the BAUD register value is 16 bits (0 to 65,535).

For synchronous operation, the BAUD register value is 8 bits (0 to 255).

**Table 25-2. Baud Rate Equations**

| Operating Mode | Condition | Baud Rate (Bits Per Second) | BAUD Register Value Calculation |
|---|---|---|---|
| Asynchronous Arithmetic | $f_{BAUD} \leq \frac{f_{ref}}{16}$ | $f_{BAUD} = \frac{f_{ref}}{16}\left(1 - \frac{BAUD}{65536}\right)$ | $BAUD = 65536 \cdot \left(1 - 16 \cdot \frac{f_{BAUD}}{f_{ref}}\right)$ |
| Synchronous | $f_{BAUD} \leq \frac{f_{ref}}{2}$ | $f_{BAUD} = \frac{f_{ref}}{2 \cdot (BAUD + 1)}$ | $BAUD = \frac{f_{ref}}{2 \cdot f_{BAUD}} - 1$ |

The baud rate error is represented by the following formula:

$$\text{Error} = 1 - \left(\frac{\text{ExpectedBaudRate}}{\text{ActualBaudRate}}\right)$$

#### 25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

The formula given for $f_{BAUD}$ calculates the average frequency over 65536 $f_{ref}$ cycles. Although the BAUD register can be set to any value between 0 and 65536, the actual average frequency of $f_{BAUD}$ over a single frame is more granular. The BAUD register values that will affect the average frequency over a single frame lead to an integer increase in the cycles per frame (CPF)

$$CPF = \frac{f_{ref}}{f_{BAUD}}(D + S)$$

where

- *D* represent the data bits per frame
- *S* represent the sum of start and first stop bits, if present.

Table 25-3 shows the BAUD register value versus baud frequency $f_{BAUD}$ at a serial engine frequency of 48MHz. This assumes a *D* value of 8 bits and an *S* value of 2 bits (10 bits, including start and stop bits).

**Table 25-3. BAUD Register Value vs. Baud Frequency**

| BAUD Register Value | Serial Engine CPF | $f_{BAUD}$ at 48MHz Serial Engine Frequency ($f_{REF}$) |
|---|---|---|
| 0 – 406 | 160 | 3MHz |
| 407 – 808 | 161 | 2.981MHz |
| 809 – 1205 | 162 | 2.963MHz |
| ... | ... | ... |
| 65206 | 31775 | 15.11kHz |
| 65207 | 31871 | 15.06kHz |
| 65208 | 31969 | 15.01kHz |

### 25.6.3 Additional Features

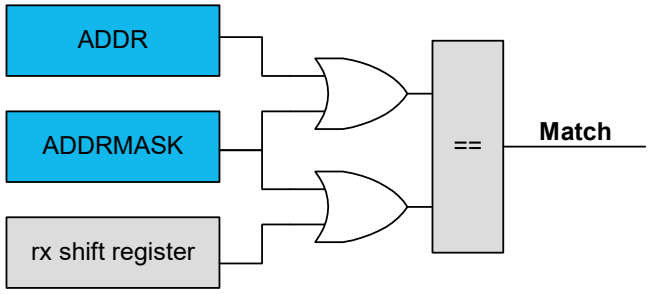
#### 25.6.3.1 Address Match and Mask

The SERCOM address match and mask feature is capable of matching either one address, two unique addresses, or a range of addresses with a mask, based on the mode selected. The match uses seven or eight bits, depending on the mode.

##### 25.6.3.1.1 Address With Mask

An address written to the Address bits in the Address register (ADDR.ADDR), and a mask written to the Address Mask bits in the Address register (ADDR.ADDRMASK) will yield an address match. All bits that are masked are not included in the match. Note that writing the ADDR.ADDRMASK to 'all zeros' will match a single unique address, while writing ADDR.ADDRMASK to 'all ones' will result in all addresses being accepted.

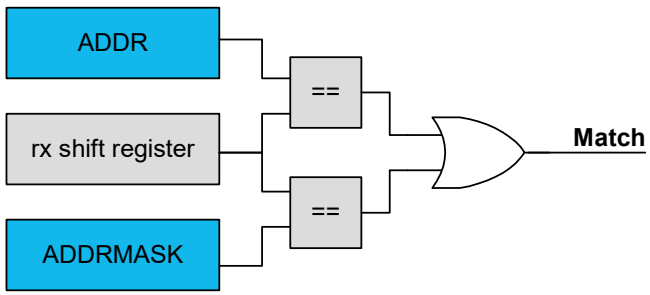
**Figure 25-4. Address With Mask**




##### 25.6.3.1.2 Two Unique Addresses

The two addresses written to ADDR and ADDRMASK will cause a match.

**Figure 25-5. Two Unique Addresses**



#### 25.6.3.1.3 Address Range

The range of addresses between and including ADDR.ADDR and ADDR.ADDRMASK will cause a match. ADDR.ADDR and ADDR.ADDRMASK can be set to any two addresses, with ADDR.ADDR acting as the upper limit and ADDR.ADDRMASK acting as the lower limit.

**Figure 25-6. Address Range**

### 25.6.4 DMA Operation

The available DMA interrupts and their depend on the operation mode of the SERCOM peripheral. Refer to the Functional Description sections of the respective SERCOM mode.

**Related Links**

26. SERCOM USART
27. SERCOM SPI – SERCOM Serial Peripheral Interface
28. SERCOM I2C – Inter-Integrated Circuit

### 25.6.5 Interrupts

Interrupt sources are mode-specific. See the respective SERCOM mode chapters for details.

Each interrupt source has its own interrupt flag.

The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is met.

Each interrupt can be individually enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled, or the SERCOM is reset. For details on clearing interrupt flags, refer to the INTFLAG register description.

The value of INTFLAG indicates which interrupt condition occurred. The user must read the INTFLAG register to determine which interrupt condition is present.

**Note:** Interrupts must be globally enabled for interrupt requests.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 25.6.6 Events

Not applicable.

### 25.6.7 Sleep Mode Operation

The peripheral can operate in any sleep mode where the selected serial clock is running. This clock can be external or generated by the internal baud-rate generator.

The SERCOM interrupts can be used to wake up the device from sleep modes. Refer to the different SERCOM mode chapters for details.

### 25.6.8 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

Required read-synchronization is denoted by the "Read-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

# 26. SERCOM USART

## 26.1 Overview

The Universal Synchronous and Asynchronous Receiver and Transmitter (USART) is one of the available modes in the Serial Communication Interface (SERCOM).

The USART uses the SERCOM transmitter and receiver, see 26.3 Block Diagram. Labels in uppercase letters are synchronous to CLK_SERCOMx_APB and accessible for CPU. Labels in lowercase letters can be programmed to run on the internal generic clock or an external clock.

The transmitter consists of a single write buffer, a shift register, and control logic for different frame formats. The write buffer support data transmission without any delay between frames. The receiver consists of a two-level receive buffer and a shift register. Status information of the received data is available for error checking. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception.

**Related Links**

25. SERCOM – Serial Communication Interface

## 26.2 USART Features

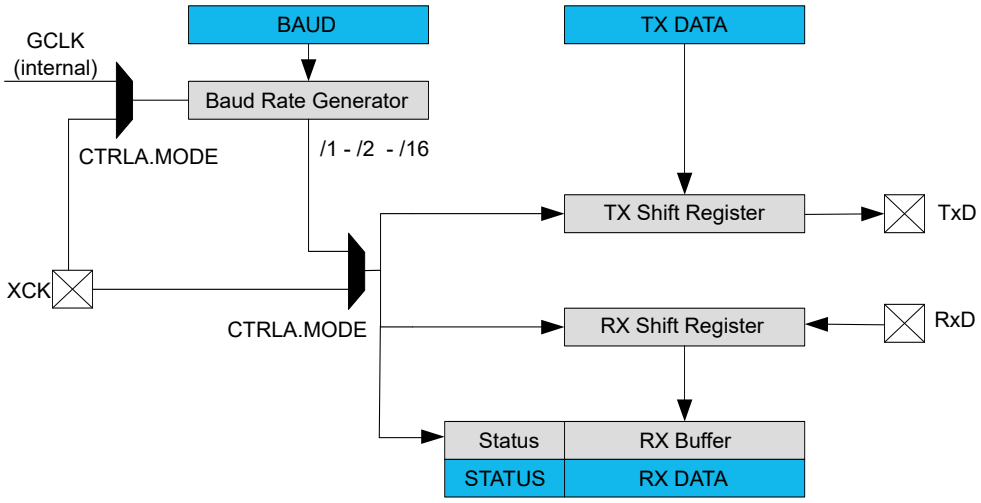
- Full-duplex operation
- Asynchronous (with clock reconstruction) or synchronous operation
- Internal or external clock source for asynchronous and synchronous operation
- Baud-rate generator
- Supports serial frames with 5, 6, 7, 8 or 9 data bits and 1 or 2 stop bits
- Odd or even parity generation and parity check
- Selectable LSB- or MSB-first data transfer
- Buffer overflow and frame error detection
- Noise filtering, including false start-bit detection and digital low-pass filter
- Collision detection
- Can operate in all sleep modes
- Operation at speeds up to half the system clock for internally generated clocks
- Operation at speeds up to the system clock for externally generated clocks
- RTS and CTS flow control
- IrDA modulation and demodulation up to 115.2kbps
- Start-of-frame detection
- Can work with DMA
- Up to 16-bytes internal FIFO

**Related Links**

25.2 Features

## 26.3 Block Diagram

**Figure 26-1. USART Block Diagram**




## 26.4 Signal Description

**Table 26-1. SERCOM USART Signals**

| Signal Name | Type | Description |
|---|---|---|
| PAD[3:0] | Digital I/O | General SERCOM pins |

One signal can be mapped to one of several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 26.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 26.5.1 I/O Lines

Using the USART's I/O lines requires the I/O pins to be configured using the I/O Pin Controller (PORT).

When the SERCOM is used in USART mode, the SERCOM controls the direction and value of the I/O pins according to the table below. Both PORT control bits PINCFGn.PULLEN and PINCFGn.DRVSTR are still effective. If the receiver or transmitter is disabled, these pins can be used for other purposes.

**Table 26-2. USART Pin Configuration**

| Pin | Pin Configuration |
|---|---|
| TxD | Output |
| RxD | Input |
| XCK | Output or input |

The combined configuration of PORT and the Transmit Data Pinout and Receive Data Pinout bit fields in the Control A register (CTRLA.TXPO and CTRLA.RXPO, respectively) will define the physical position of the USART signals in Table 26-2.

**Related Links**

23. PORT - I/O Pin Controller

### 26.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes.

**Related Links**

16. PM – Power Manager

### 26.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the Power Manager. Refer to *Peripheral Clock Masking* for details and default status of this clock.

A generic clock (GCLK_SERCOMx_CORE) is required to clock the SERCOMx_CORE. This clock must be configured and enabled in the Generic Clock Controller before using the SERCOMx_CORE. Refer to *GCLK - Generic Clock Controller* for details.

This generic clock is asynchronous to the bus clock (CLK_SERCOMx_APB). Therefore, writing to certain registers will require synchronization to the clock domains. Refer to *Synchronization* for further details.

**Related Links**

26.6.6 Synchronization

15. GCLK - Generic Clock Controller

16.6.2.6 Peripheral Clock Masking

### 26.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 26.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 26.5.6 Events

Not applicable.

### 26.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or

data loss may result during debugging. This peripheral can be forced to halt operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

**Related Links**

26.8.11 DBGCTRL

### 26.5.8 Register Access Protection

Registers with write-access can be write-protected optionally by the peripheral access controller (PAC).

PAC Write-Protection is not available for the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)

Optional PAC Write-Protection is denoted by the "PAC Write-Protection" property in each individual register description.

Write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 26.5.9 Analog Connections

Not applicable.

## 26.6 Functional Description

### 26.6.1 Principle of Operation

The USART uses the following lines for data transfer:

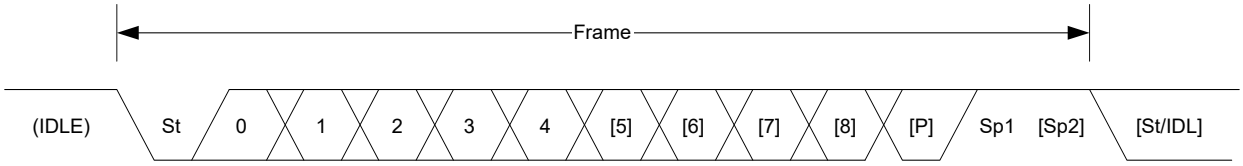
- RxD for receiving
- TxD for transmitting
- XCK for the transmission clock in synchronous operation

USART data transfer is frame based. A serial frame consists of:

- 1 start bit
- From 5 to 9 data bits (MSB or LSB first)
- No, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by one character of data bits. If enabled, the parity bit is inserted after the data bits and before the first stop bit. After the stop bit(s) of a frame, either the next frame can follow immediately, or the communication line can return to the idle (high) state. The figure below illustrates the possible frame formats. Brackets denote optional bits.

**Figure 26-2. Frame Formats**



**St** Start bit. Signal is always low.

**n, [n]** Data bits. 0 to [5..9]

**[P]** Parity bit. Either odd or even.

**Sp, [Sp]** Stop bit. Signal is always high.

**IDLE** No frame is transferred on the communication line. Signal is always high in this state.

### 26.6.2 Basic Operation

#### 26.6.2.1 Initialization

The following registers are enable-protected, meaning they can only be written when the USART is disabled (CTRL.ENABLE=0):

- Control A register (CTRLA), except the Enable (ENABLE) and Software Reset (SWRST) bits.
- Control B register (CTRLB), except the Receiver Enable (RXEN) and Transmitter Enable (TXEN) bits.
- Baud register (BAUD)

When the USART is enabled or is being enabled (CTRLA.ENABLE=1), any writing attempt to these registers will be discarded. If the peripheral is being disabled, writing to these registers will be executed after disabling is completed. Enable-protection is denoted by the "Enable-Protection" property in the register description.

Before the USART is enabled, it must be configured by these steps:

1. Select either external (0x0) or internal clock (0x1) by writing the Operating Mode value in the CTRLA register (CTRLA.MODE).
2. Select either asynchronous (0) or or synchronous (1) communication mode by writing the Communication Mode bit in the CTRLA register (CTRLA.CMODE).
3. Select pin for receive data by writing the Receive Data Pinout value in the CTRLA register (CTRLA.RXPO).
4. Select pads for the transmitter and external clock by writing the Transmit Data Pinout bit in the CTRLA register (CTRLA.TXPO).
5. Configure the Character Size field in the CTRLB register (CTRLB.CHSIZE) for character size.
6. Set the Data Order bit in the CTRLA register (CTRLA.DORD) to determine MSB- or LSB-first data transmission.
7. To use parity mode:
   7.1. Enable parity mode by writing 0x1 to the Frame Format field in the CTRLA register (CTRLA.FORM).
   7.2. Configure the Parity Mode bit in the CTRLB register (CTRLB.PMODE) for even or odd parity.
8. Configure the number of stop bits in the Stop Bit Mode bit in the CTRLB register (CTRLB.SBMODE).
9. When using an internal clock, write the Baud register (BAUD) to generate the desired baud rate.
10. Enable the transmitter and receiver by writing '1' to the Receiver Enable and Transmitter Enable bits in the CTRLB register (CTRLB.RXEN and CTRLB.TXEN).

#### 26.6.2.2 Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

Refer to the CTRLA register description for details.

#### 26.6.2.3 Clock Generation and Selection

For both synchronous and asynchronous modes, the clock used for shifting and sampling data can be generated internally by the SERCOM baud-rate generator or supplied externally through the XCK line.

The synchronous mode is selected by writing a '1' to the Communication Mode bit in the Control A register (CTRLA.CMODE), the asynchronous mode is selected by writing a zero to CTRLA.CMODE.

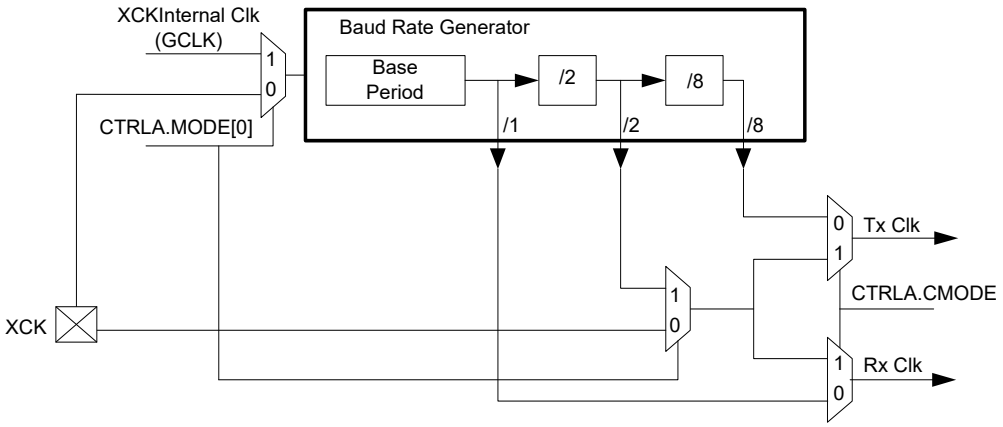
The internal clock source is selected by writing 0x1 to the Operation Mode bit field in the Control A register (CTRLA.MODE), the external clock source is selected by writing 0x0 to CTRLA.MODE.

The SERCOM baud-rate generator is configured as in the figure below.

In asynchronous mode (CTRLA.CMODE=0), the 16-bit Baud register value is used.

In synchronous mode (CTRLA.CMODE=1), the eight LSBs of the Baud register are used. Refer to *Clock Generation – Baud-Rate Generator* for details on configuring the baud rate.

**Figure 26-3. Clock Generation**




**Related Links**

25.6.2.3 Clock Generation – Baud-Rate Generator

25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

##### 26.6.2.3.1 Synchronous Clock Operation

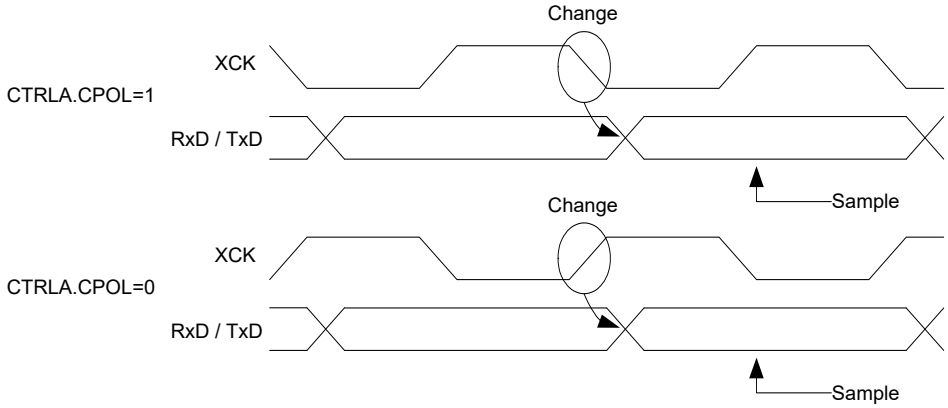
In synchronous mode, the CTRLA.MODE bit field determines whether the transmission clock line (XCK) serves either as input or output. The dependency between clock edges, data sampling, and data change is the same for internal and external clocks. Data input on the RxD pin is sampled at the opposite XCK clock edge when data is driven on the TxD pin.

The Clock Polarity bit in the Control A register (CTRLA.CPOL) selects which XCK clock edge is used for RxD sampling, and which is used for TxD change:

When CTRLA.CPOL is '0', the data will be changed on the rising edge of XCK, and sampled on the falling edge of XCK.

When CTRLA.CPOL is '1', the data will be changed on the falling edge of XCK, and sampled on the rising edge of XCK.

**Figure 26-4. Synchronous Mode XCK Timing**




When the clock is provided through XCK (CTRLA.MODE=0x0), the shift registers operate directly on the XCK clock. This means that XCK is not synchronized with the system clock and, therefore, can operate at frequencies up to the system frequency.

#### 26.6.2.4 Data Register

The USART Transmit Data register (TxDATA) and USART Receive Data register (RxDATA) share the same I/O address, referred to as the Data register (DATA). Writing the DATA register will update the TxDATA register. Reading the DATA register will return the contents of the RxDATA register.

#### 26.6.2.5 Data Transmission

Data transmission is initiated by writing the data to be sent into the DATA register. Then, the data in TxDATA will be moved to the shift register when the shift register is empty and ready to send a new frame. After the shift register is loaded with data, the data frame will be transmitted.

When the entire data frame including stop bit(s) has been transmitted and no new data was written to DATA, the Transmit Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set, and the optional interrupt will be generated.

The Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) indicates that the register is empty and ready for new data. The DATA register should only be written to when INTFLAG.DRE is set.

##### 26.6.2.5.1 Disabling the Transmitter

The transmitter is disabled by writing '0' to the Transmitter Enable bit in the CTRLB register (CTRLB.TXEN).

Disabling the transmitter will complete only after any ongoing and pending transmissions are completed, i.e., there is no data in the transmit shift register and TxDATA to transmit.

#### 26.6.2.6 Data Reception

The receiver accepts data when a valid start bit is detected. Each bit following the start bit will be sampled according to the baud rate or XCK clock, and shifted into the receive shift register until the first stop bit of a frame is received. The second stop bit will be ignored by the receiver.

When the first stop bit is received and a complete serial frame is present in the receive shift register, the contents of the shift register will be moved into the two-level receive buffer. Then, the Receive Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) will be set, and the optional interrupt will be generated.

The received data can be read from the DATA register when the Receive Complete interrupt flag is set.

#### 26.6.2.6.1 Disabling the Receiver

Writing '0' to the Receiver Enable bit in the CTRLB register (CTRLB.RXEN) will disable the receiver, flush the two-level receive buffer, and data from ongoing receptions will be lost.

#### 26.6.2.6.2 Error Bits

The USART receiver has three error bits in the Status (STATUS) register: Frame Error (FERR), Buffer Overflow (BUFOVF), and Parity Error (PERR). Once an error happens, the corresponding error bit will be set until it is cleared by writing ‘1’ to it. These bits are also cleared automatically when the receiver is disabled.

There are two methods for buffer overflow notification, selected by the Immediate Buffer Overflow Notification bit in the Control A register (CTRLA.IBON):

When CTRLA.IBON=1, STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA, until the receiver complete interrupt flag (INTFLAG.RXC) is cleared.

When CTRLA.IBON=0, the buffer overflow condition is attending data through the receive FIFO. After the received data is read, STATUS.BUFOVF will be set along with INTFLAG.RXC.

#### 26.6.2.6.3 Asynchronous Data Reception

The USART includes a clock recovery and data recovery unit for handling asynchronous data reception.

The clock recovery logic can synchronize the incoming asynchronous serial frames at the RxD pin to the internally generated baud-rate clock.

The data recovery logic samples and applies a low-pass filter to each incoming bit, thereby improving the noise immunity of the receiver.

#### 26.6.2.6.4 Asynchronous Operational Range

The operational range of the asynchronous reception depends on the accuracy of the internal baud-rate clock, the rate of the incoming frames, and the frame size (in number of bits). In addition, the operational range of the receiver is depending on the difference between the received bit rate and the internally generated baud rate. If the baud rate of an external transmitter is too high or too low compared to the internally generated baud rate, the receiver will not be able to synchronize the frames to the start bit.

There are two possible sources for a mismatch in baud rate: First, the reference clock will always have some minor instability. Second, the baud-rate generator cannot always do an exact division of the reference clock frequency to get the baud rate desired. In this case, the BAUD register value should be set to give the lowest possible error. Refer to *Clock Generation – Baud-Rate Generator* for details.

Recommended maximum receiver baud-rate errors for various character sizes are shown in the table below.

**Table 26-3. Asynchronous Receiver Error for 16-fold Oversampling**

| D (Data bits+Parity) | $R_{SLOW}$ [%] | $R_{FAST}$ [%] | Max. total error [%] | Recommended max. Rx error [%] |
|---|---|---|---|---|
| 5 | 94.12 | 107.69 | +5.88/-7.69 | ±2.5 |
| 6 | 94.92 | 106.67 | +5.08/-6.67 | ±2.0 |
| 7 | 95.52 | 105.88 | +4.48/-5.88 | ±2.0 |
| 8 | 96.00 | 105.26 | +4.00/-5.26 | ±2.0 |

| D (Data bits+Parity) | $R_{SLOW}$ [%] | $R_{FAST}$ [%] | Max. total error [%] | Recommended max. Rx error [%] |
|---|---|---|---|---|
| 9 | 96.39 | 104.76 | +3.61/-4.76 | ±1.5 |
| 10 | 96.70 | 104.35 | +3.30/-4.35 | ±1.5 |

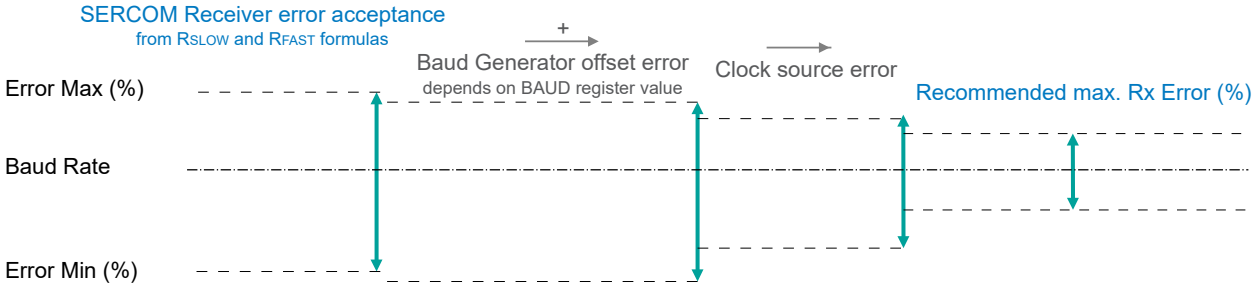
The following equations calculate the ratio of the incoming data rate and internal receiver baud rate:

$$R_{SLOW} = \frac{(D+1)S}{S-1+D\cdot S+S_F} \quad , \quad R_{FAST} = \frac{(D+2)S}{(D+1)S+S_M}$$

- $R_{SLOW}$ is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- $R_{FAST}$ is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate
- $D$ is the sum of character size and parity size ($D$ = 5 to 10 bits)
- $S$ is the number of samples per bit ($S$ = 16, 8 or 3)
- $S_F$ is the first sample number used for majority voting ($S_F$ = 7, 3, or 2) when CTRLA.SAMPA=0.
- $S_M$ is the middle sample number used for majority voting ($S_M$ = 8, 4, or 2) when CTRLA.SAMPA=0.

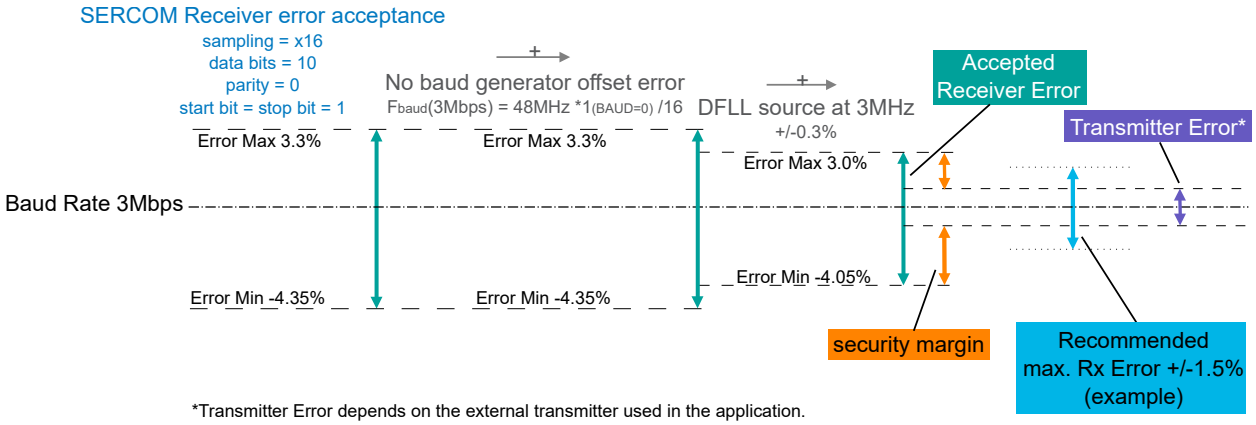
The recommended maximum Rx Error assumes that the receiver and transmitter equally divide the maximum total error. Its connection to the SERCOM Receiver error acceptance is depicted in this figure:

**Figure 26-5. USART Rx Error Calculation**




The recommendation values in the table above accommodate errors of the clock source and the baud generator. The following figure gives an example for a baud rate of 3Mbps:

**Figure 26-6. USART Rx Error Calculation Example**




*Transmitter Error depends on the external transmitter used in the application.
It is advised that it is within the Recommended max. Rx Error (+/-1.5% in this example).
Larger Transmitter Errors are acceptable but must lie within the Accepted Receiver Error.

**Related Links**

25.6.2.3 Clock Generation – Baud-Rate Generator
25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

### 26.6.3 Additional Features

#### 26.6.3.1 Parity

Even or odd parity can be selected for error checking by writing 0x1 to the Frame Format bit field in the Control A register (CTRLA.FORM).

If *even parity* is selected (CTRLB.PMODE=0), the parity bit of an outgoing frame is '1' if the data contains an odd number of bits that are '1', making the total number of '1' even.

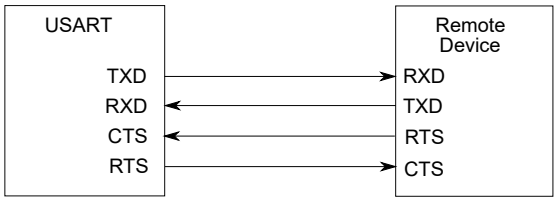
If *odd parity* is selected (CTRLB.PMODE=1), the parity bit of an outgoing frame is '1' if the data contains an even number of bits that are '0', making the total number of '1' odd.

When parity checking is enabled, the parity checker calculates the parity of the data bits in incoming frames and compares the result with the parity bit of the corresponding frame. If a parity error is detected, the Parity Error bit in the Status register (STATUS.PERR) is set.

#### 26.6.3.2 Hardware Handshaking

The USART features an out-of-band hardware handshaking flow control mechanism, implemented by connecting the RTS and CTS pins with the remote device, as shown in the figure below.

**Figure 26-7. Connection with a Remote Device for Hardware Handshaking**

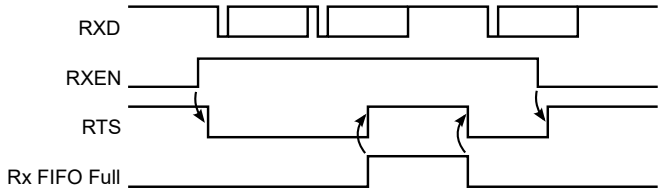



Hardware handshaking is only available in the following configuration:

- USART with internal clock (CTRLA.MODE=1),
- Asynchronous mode (CTRLA.CMODE=0),
- and Flow control pinout (CTRLA.TXPO=2).

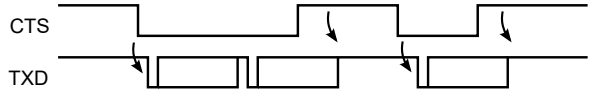
When the receiver is disabled or the receive FIFO is full, the receiver will drive the RTS pin high. This notifies the remote device to stop transfer after the ongoing transmission. Enabling and disabling the receiver by writing to CTRLB.RXEN will set/clear the RTS pin after a synchronization delay. When the receive FIFO goes full, RTS will be set immediately and the frame being received will be stored in the shift register until the receive FIFO is no longer full.

**Figure 26-8. Receiver Behavior when Operating with Hardware Handshaking**




The current CTS Status is in the STATUS register (STATUS.CTS). Character transmission will start only if STATUS.CTS=0. When CTS is set, the transmitter will complete the ongoing transmission and stop transmitting.

**Figure 26-9. Transmitter Behavior when Operating with Hardware Handshaking**



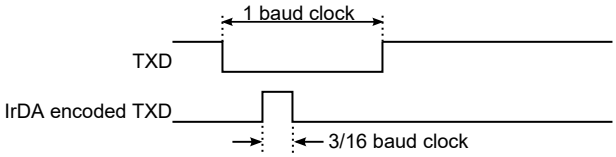

#### 26.6.3.3 IrDA Modulation and Demodulation

Transmission and reception can be encoded IrDA compliant up to 115.2 kb/s. IrDA modulation and demodulation work in the following configuration:

- IrDA encoding enabled (CTRLB.ENC=1),
- Asynchronous mode (CTRLA.CMODE=0),
- and 16x sample rate (CTRLA.SAMPR[0]=0).

During transmission, each low bit is transmitted as a high pulse. The pulse width is 3/16 of the baud rate period, as illustrated in the figure below.

**Figure 26-10. IrDA Transmit Encoding**




The reception decoder has two main functions.

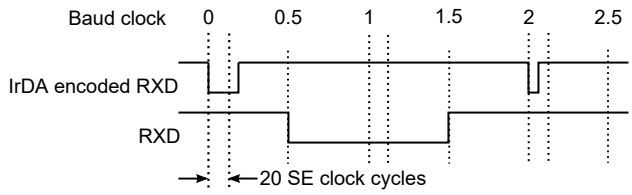
The first is to synchronize the incoming data to the IrDA baud rate counter. Synchronization is performed at the start of each zero pulse.

The second main function is to decode incoming Rx data. If a pulse width meets the minimum length set by configuration (RXPL.RXPL), it is accepted. When the baud rate counter reaches its middle value (1/2 bit length), it is transferred to the receiver.

**Note:** Note that the polarity of the transmitter and receiver are opposite: During transmission, a '0' bit is transmitted as a '1' pulse. During reception, an accepted '0' pulse is received as a '0' bit.

**Example:** The figure below illustrates reception where RXPL.RXPL is set to 19. This indicates that the pulse width should be at least 20 SE clock cycles. When using BAUD=0xE666 or 160 SE cycles per bit, this corresponds to 2/16 baud clock as minimum pulse width required. In this case the first bit is accepted as a '0', the second bit is a '1', and the third bit is also a '1'. A low pulse is rejected since it does not meet the minimum requirement of 2/16 baud clock.

**Figure 26-11. IrDA Receive Decoding**




#### 26.6.3.4 Break Character Detection and Auto-Baud

Break character detection and auto-baud are available in this configuration:

- Auto-baud frame format (CTRLA.FORM = 0x04 or 0x05),
- Asynchronous mode (CTRLA.CMODE = 0),

- and 16x sample rate using fractional baud rate generation (CTRLA.SAMPR = 1).

The USART uses a break detection threshold of greater than 11 nominal bit times at the configured baud rate. At any time, if more than 11 consecutive dominant bits are detected on the bus, the USART detects a Break Field. When a Break Field has been detected, the Receive Break interrupt flag (INTFLAG.RXBRK) is set and the USART expects the Sync Field character to be 0x55. This field is used to update the actual baud rate in order to stay synchronized. If the received Sync character is not 0x55, then the Inconsistent Sync Field error flag (STATUS.ISF) is set along with the Error interrupt flag (INTFLAG.ERROR), and the baud rate is unchanged.

After a break field is detected and the start bit of the Sync Field is detected, a counter is started. The counter is then incremented for the next 8 bit times of the Sync Field. At the end of these 8 bit times, the counter is stopped. At this moment, the 13 most significant bits of the counter (value divided by 8) give the new clock divider (BAUD.BAUD), and the 3 least significant bits of this value (the remainder) give the new Fractional Part (BAUD.FP).

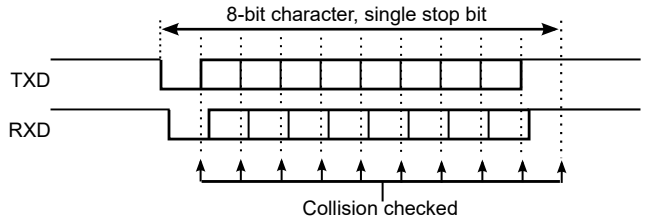
When the Sync Field has been received, the clock divider (BAUD.BAUD) and the Fractional Part (BAUD.FP) are updated after a synchronization delay. After the Break and Sync Fields are received, multiple characters of data can be received.

#### 26.6.3.5 Collision Detection

When the receiver and transmitter are connected either through pin configuration or externally, transmit collision can be detected after selecting the Collision Detection Enable bit in the CTRLB register (CTRLB.COLDEN=1). To detect collision, the receiver and transmitter must be enabled (CTRLB.RXEN=1 and CTRLB.TXEN=1).

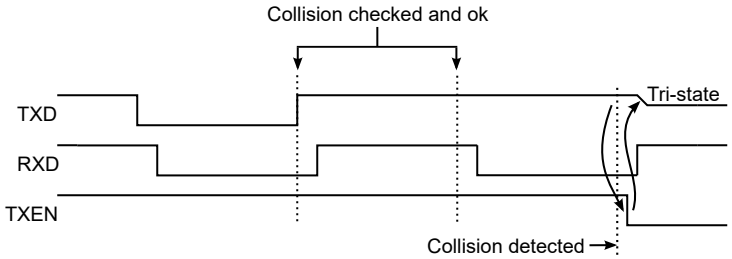
Collision detection is performed for each bit transmitted by comparing the received value with the transmit value, as shown in the figure below. While the transmitter is idle (no transmission in progress), characters can be received on RxD without triggering a collision.

**Figure 26-12. Collision Checking**




The next figure shows the conditions for a collision detection. In this case, the start bit and the first data bit are received with the same value as transmitted. The second received data bit is found to be different than the transmitted bit at the detection point, which indicates a collision.

**Figure 26-13. Collision Detected**




When a collision is detected, the USART follows this sequence:

1. Abort the current transfer.

2. Flush the transmit buffer.
3. Disable transmitter (CTRLB.TXEN=0)
   – This is done after a synchronization delay. The CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) will be set until this is complete.
   – After disabling, the TxD pin will be tri-stated.
4. Set the Collision Detected bit (STATUS.COLL) along with the Error interrupt flag (INTFLAG.ERROR).
5. Set the Transmit Complete interrupt flag (INTFLAG.TXC), since the transmit buffer no longer contains data.

After a collision, software must manually enable the transmitter again before continuing, after assuring that the CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) is not set.

#### 26.6.3.6 Loop-Back Mode

For loop-back mode, configure the Receive Data Pinout (CTRLA.RXPO) and Transmit Data Pinout (CTRLA.TXPO) to use the same data pins for transmit and receive. The loop-back is through the pad, so the signal is also available externally.

#### 26.6.3.7 Start-of-Frame Detection

The USART start-of-frame detector can wake up the CPU when it detects a start bit. In standby sleep mode, the internal fast startup oscillator must be selected as the GCLK_SERCOMx_CORE source.

When a 1-to-0 transition is detected on RxD, the 8MHz Internal Oscillator is powered up and the USART clock is enabled. After startup, the rest of the data frame can be received, provided that the baud rate is slow enough in relation to the fast startup internal oscillator start-up time. Refer to *Electrical Characteristics* for details. The start-up time of this oscillator varies with supply voltage and temperature.

The USART start-of-frame detection works both in asynchronous and synchronous modes. It is enabled by writing '1' to the Start of Frame Detection Enable bit in the Control B register (CTRLB.SFDE).

If the Receive Start Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.RXS) is set, the Receive Start interrupt is generated immediately when a start is detected.

When using start-of-frame detection without the Receive Start interrupt, start detection will force the 8MHz Internal Oscillator and USART clock active while the frame is being received. In this case, the CPU will not wake up until the Receive Complete interrupt is generated.

**Related Links**

37. Electrical Characteristics

#### 26.6.3.8 Sample Adjustment

In asynchronous mode (CTRLA.CMODE=0), three samples in the middle are used to determine the value based on majority voting. The three samples used for voting can be selected using the Sample Adjustment bit field in Control A register (CTRLA.SAMPA). When CTRLA.SAMPA=0, samples 7-8-9 are used for 16x oversampling, and samples 3-4-5 are used for 8x oversampling.

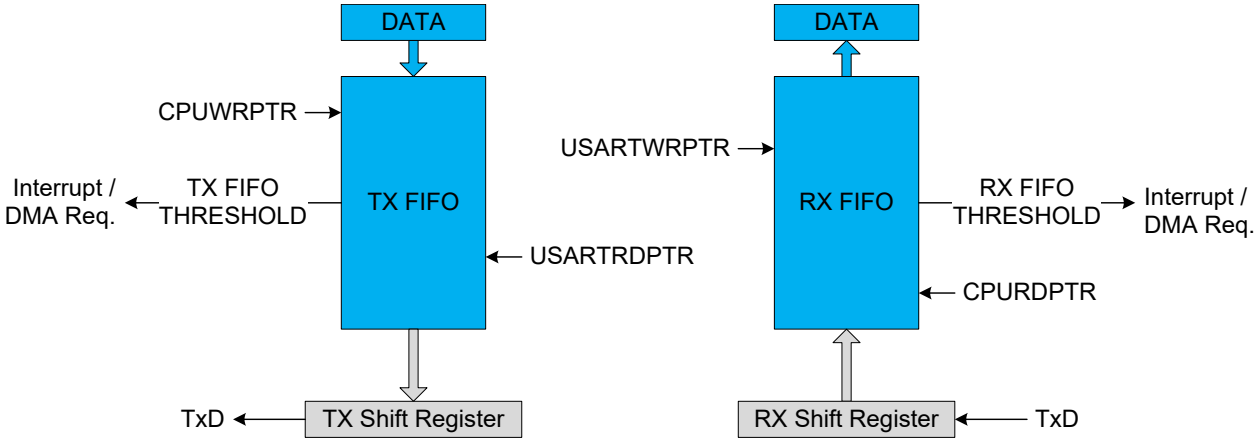
#### 26.6.3.9 FIFO Operation

The USART embeds up to 16-bytes FIFO capability. The receive / transmit buffer is considered to have the FIFO mode enabled when the FIFOEN bit in CTRLC register is set to a '1' (CTRLC.FIFOEN = 1). By default, the FIFO can act as a 16-by-8-bit array, or as a 4-by-32-bit array, depending on the setting of the CTRLC.DATA32B bit.

The hardware around this array implements four pointers, called the CPU Write Pointer (CPUWRPTR), the CPU Read Pointer (CPURDPTR), the USART Write pointer (USARTWRPTR) and the USART Read

pointer (USARTRDPTR). All of these pointers reset to '0'. The CPUWRPTR and CPURDPTR pointers are native to the CPU clock domain, while the USARTWRPTR and USARTRDPTR are native to the USART domain. The location pointed to by the CPUWRPTR is the current TX FIFO. The location pointed to by the CPURDPTR becomes the current RX FIFO. Writes to DATA register by the CPU will point to TX FIFO. Reads to DATA register by the CPU will point to RX FIFO. The location pointed to by the USARTWRPTR / USARTRDPTR is logically the current RX/TX shift registers.

**Figure 26-14. FIFO Overview**




The interrupts and DMA triggers are generated according to FIFO threshold settings in Control C register (CTRLC.TXTRHOLD, CTRLC.RXTRHOLD).

The Data Register Empty interrupt flag, and the DMA TX trigger respectivly, are generated when the available place in the TX FIFO is equal or higher than the threshold value defined by the CTRLC.TXTRHOLD settings. The Transfer complete interrupt is generated when the TX FIFO is empty and the entire data (including the stop bits) has been transmitted.

The Receive Complete interrupt flag, and the DMA RX trigger respectivly, are generated when the number of bytes present in the RX FIFO equals or is higher than the threshold value defined by the CTRLC.RXTRHOLD settings. The ERROR interrupt flag is generated when both RX shifter and the RX FIFO are full.

The FIFO is fully accessible if the SERCOM is halted, by writting the corresponding CPU FIFO pointer in the FIFOPTR register. The RX or TX FIFO can be individually cleared, by setting the respective FIFO Clear bit in the Control B register (CTRLB.FIFOCLR). The FIFO Clear must be written before data transfer begins. Writing the FIFO Clear bits while a frame is in progress will produce unpredicatable results.

#### 26.6.3.9.1 Pointer Operation when DATA Transmission

As in normal operation, data transmission is initiated by writing the data to be sent into the TX FIFO, by accessing the DATA register. CPUWRPTR is incremented by 1 every time the CPU writes a word to the memory array. Then, the data in TX FIFO will be moved to the shift register when the shift register is empty and ready to send a new frame, and the USARTRDPTR is incremented by 1. After the shift register is loaded with data, the data frame will be transmitted.

As long as data are present in TX FIFO (FIFOSPACE.TXSPACE != 0), a new data will be automatically loaded in the TX shift register when the previous data transmission is completed. All pointers increment to their maximum value, dictated by CTRLC.DATA32B bit, and then rolls over to '0'.

Depending the TX FIFO Threshold settings (CTRLC.TXTRHOLD), Interrupt Flag Status and Clear register (INTFLAG.DRE) indicates that the register is empty and ready for new data. DATA to TX FIFO should only be written to when INTFLAG.DRE is set.

If the USART is halted when debugging, the CPUWRPTR pointer can be accessed by writting the CPUWRPTR bits in FIFOPTR register (FIFOPTR.CPUWRPTR). These bits will not increment if a new data is written into the TX FIFO memory.

#### 26.6.3.9.2 Pointer Operation when DATA Reception

As in normal operation, when the first stop bit is received and a complete serial frame is present in the receive shift register, the contents of the shift register will be moved into the RX FIFO, and the USARTWRPTR is incremented by one. Depending the RX FIFO Threshold settings (CTRLC.RXTRHOLD), the Receive Complete interrupt flag (INTFLAG.RXC) is set, and the DATA can be read from RX FIFO. When a DATA is read, the CPURDPTR is incremented. As long as data are present in RX FIFO (FIFOSPACE.RXSPACE != 0), the CPU can read these data by accessing the DATA register. All pointers increment to their maximum value, dictated by CTRLC.DATA32B bit, and then rolls over to ‘0’.

When both RX shifter and RX FIFO are full, the Buffer Overflow status bit is set (STATUS.BUFOVF) and optional ERROR interrupt is generated. The data will not be stored while BUFOVF is ‘1’, effectively disabling the module until software reads RX FIFO.

If the USART is halted when debugging, the RX FIFO CPU read pointer can be accessed by writting the CPURDPTR bits in FIFOPTR register (FIFOPTR.CPURDPTR). These bits will not increment if a new data is read from the RX FIFO memory.

### 26.6.4 DMA, Interrupts and Events

**Table 26-4. Module Request for SERCOM USART**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Data Register Empty (DRE) | Yes<br>(request cleared when data is written) | Yes | NA |
| Receive Complete (RXC) | Yes<br>(request cleared when data is read) | Yes | |
| Transmit Complete (TXC) | NA | Yes | |
| Receive Start (RXS) | NA | Yes | |
| Clear to Send Input Change (CTSIC) | NA | Yes | |
| Receive Break (RXBRK) | NA | Yes | |
| Error (ERROR) | NA | Yes | |

**Table 26-5. Module Request for SERCOM USART**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Standard (DRE): Data Register Empty | Yes | Yes | NA |

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| FIFO (DRE): at least TXTRHOLD locations in TX FIFO are empty | (request cleared when data is written) | | |
| Standard (RXC): Receive Complete<br>FIFO (RXC): at least RXTRHOLD data available in RX FIFO, or a last word available and length frame reception completed. | Yes<br>(request cleared when data is read) | Yes | |
| Standard (TXC): Transmit Complete<br>FIFO (TXC): Transmit Complete and TX FIFO is empty | NA | Yes | |
| Receive Start (RXS) | NA | Yes | |
| Clear to Send Input Change (CTSIC) | NA | Yes | |
| Receive Break (RXBRK) | NA | Yes | |
| Error (ERROR) | NA | Yes | |

#### 26.6.4.1 DMA Operation

The USART generates the following DMA requests:

- Data received (RX): The request is set when data is available in the receive FIFO. The request is cleared when DATA is read.
- Data transmit (TX): The request is set when the transmit buffer (TX DATA) is empty. The request is cleared when DATA is written.
- Data received (RX): The request is set when data is available in the receive FIFO or if at least RXTRHOLD data are available in the RX FIFO when FIFO operation is enabled. The request is cleared when DATA is read.
- Data transmit (TX): The request is set when the transmit buffer (TX DATA) is empty or if at least TXTRHOLD data locations are empty in the TX FIFO, when FIFO operation is enabled. The request is cleared when DATA is written.

#### 26.6.4.2 Interrupts

The USART has the following interrupt sources. These are asynchronous interrupts, and can wake up the device from any sleep mode:

- Data Register Empty (DRE)
- Receive Complete (RXC)
- Transmit Complete (TXC)
- Receive Start (RXS)
- Clear to Send Input Change (CTSIC)
- Received Break (RXBRK)
- Error (ERROR)

Each interrupt source has its own interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is met. Each interrupt can be individually

enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and if the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled, or the USART is reset. For details on clearing interrupt flags, refer to the INTFLAG register description.

The value of INTFLAG indicates which interrupt is executed. Note that interrupts must be globally enabled for interrupt requests. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 26.6.4.3 Events

Not applicable.

### 26.6.5 Sleep Mode Operation

The behavior in sleep mode is depending on the clock source and the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY):

- Internal clocking, CTRLA.RUNSTDBY=1: GCLK_SERCOMx_CORE can be enabled in all sleep modes. Any interrupt can wake up the device.
- External clocking, CTRLA.RUNSTDBY=1: The Receive Start and the Receive Complete interrupt(s) can wake up the device.
- Internal clocking, CTRLA.RUNSTDBY=0: Internal clock will be disabled, after any ongoing transfer was completed. The Receive Start and the Receive Complete interrupt(s) can wake up the device.
- External clocking, CTRLA.RUNSTDBY=0: External clock will be disconnected, after any ongoing transfer was completed. All reception will be dropped.

### 26.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in the CTRLA register (CTRLA.SWRST)
- Enable bit in the CTRLA register (CTRLA.ENABLE)
- Receiver Enable bit in the CTRLB register (CTRLB.RXEN)
- Transmitter Enable bit in the Control B register (CTRLB.TXEN)

**Note:** CTRLB.RXEN is write-synchronized somewhat differently. See also 26.8.2 CTRLB for details.

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 26.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| | | 15:8 | SAMPR[2:0] | | | | | | | IBON |
| | | 23:16 | SAMPA[1:0] | | RXPO[1:0] | | | | TXPO[1:0] | |
| | | 31:24 | | DORD | CPOL | CMODE | FORM[3:0] | | | |
| 0x04 | CTRLB | 7:0 | | SBMODE | | | | CHSIZE[2:0] | | |
| | | 15:8 | | | PMODE | | | ENC | SFDE | COLDEN |
| | | 23:16 | FIFOCLR[1:0] | | | | | | RXEN | TXEN |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | BAUD | 7:0 | BAUD[7:0] | | | | | | | |
| | | 15:8 | BAUD[15:8] | | | | | | | |
| 0x0E | RXPL | 7:0 | RXPL[7:0] | | | | | | | |
| 0x0F ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENSET | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTFLAG | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x19 | Reserved | | | | | | | | | |
| 0x1A | STATUS | 7:0 | | TXE | COLL | ISF | CTS | BUFOVF | FERR | PERR |
| | | 15:8 | | | | | | | | |
| 0x1C | SYNCBUSY | 7:0 | | | | | | CTRLB | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x27 | Reserved | | | | | | | | | |
| 0x28 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | DATA[8:8] |
| 0x2A ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | DBGCTRL | 7:0 | | | | | | | | DBGSTOP |
| 0x31 ... 0x33 | Reserved | | | | | | | | | |
| 0x34 | FIFOSPACE | 7:0 | | | | TXSPACE[4:0] | | | | |
| | | 15:8 | | | | RXSPACE[4:0] | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x36 | FIFOPTR | 7:0 | | | | | CPUWRPTR[3:0] | | | |
| | | 15:8 | | | | | CPURDPTR[3:0] | | | |

## 26.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 26.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | DORD | CPOL | CMODE | FORM[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SAMPA[1:0] | | RXPO[1:0] | | | | TXPO[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SAMPR[2:0] | | | | | | | IBON |
| Access | R/W | R/W | R/W | | | | | R/W |
| Reset | 0 | 0 | 0 | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – DORD** Data Order
This bit selects the data order when a character is shifted out from the Data register.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | MSB is transmitted first. |
| 1 | LSB is transmitted first. |

**Bit 29 – CPOL** Clock Polarity
This bit selects the relationship between data output change and data input sampling in synchronous mode.

This bit is not synchronized.

| CPOL | TxD Change | RxD Sample |
|---|---|---|
| 0x0 | Rising XCK edge | Falling XCK edge |
| 0x1 | Falling XCK edge | Rising XCK edge |

**Bit 28 – CMODE** Communication Mode
This bit selects asynchronous or synchronous communication.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Asynchronous communication. |
| 1 | Synchronous communication. |

**Bits 27:24 – FORM[3:0]** Frame Format
These bits define the frame format.

These bits are not synchronized.

| FORM[3:0] | Description |
|---|---|
| 0x0 | USART frame |
| 0x1 | USART frame with parity |
| 0x2-0x3 | Reserved |
| 0x4 | Auto-baud - break detection and auto-baud. |
| 0x5 | Auto-baud - break detection and auto-baud with parity |
| 0x6-0xF | Reserved |

**Bits 23:22 – SAMPA[1:0]** Sample Adjustment
These bits define the sample adjustment.

These bits are not synchronized.

| SAMPA[1:0] | 16x Over-sampling (CTRLA.SAMPR=0 or 1) | 8x Over-sampling (CTRLA.SAMPR=2 or 3) |
|---|---|---|
| 0x0 | 7-8-9 | 3-4-5 |
| 0x1 | 9-10-11 | 4-5-6 |
| 0x2 | 11-12-13 | 5-6-7 |
| 0x3 | 13-14-15 | 6-7-8 |

**Bits 21:20 – RXPO[1:0]** Receive Data Pinout
These bits define the receive data (RxD) pin configuration.

These bits are not synchronized.

| RXPO[1:0] | Name | Description |
|---|---|---|
| 0x0 | PAD[0] | SERCOM PAD[0] is used for data reception |
| 0x1 | PAD[1] | SERCOM PAD[1] is used for data reception |
| 0x2 | PAD[2] | SERCOM PAD[2] is used for data reception |
| 0x3 | PAD[3] | SERCOM PAD[3] is used for data reception |

**Bits 17:16 – TXPO[1:0]** Transmit Data Pinout
These bits define the transmit data (TxD) and XCK pin configurations.

This bit is not synchronized.

| TXPO | TxD Pin Location | XCK Pin Location (When Applicable) | RTS | CTS |
|---|---|---|---|---|
| 0x0 | SERCOM PAD[0] | SERCOM PAD[1] | N/A | N/A |
| 0x1 | SERCOM PAD[2] | SERCOM PAD[3] | N/A | N/A |
| 0x2 | SERCOM PAD[0] | N/A | SERCOM PAD[2] | SERCOM PAD[3] |
| 0x3 | Reserved | | | |

**Bits 15:13 – SAMPR[2:0]** Sample Rate
These bits select the sample rate.

These bits are not synchronized.

| SAMPR[2:0] | Description |
|---|---|
| 0x0 | 16x over-sampling using arithmetic baud rate generation. |
| 0x1 | 16x over-sampling using fractional baud rate generation. |
| 0x2 | 8x over-sampling using arithmetic baud rate generation. |
| 0x3 | 8x over-sampling using fractional baud rate generation. |
| 0x4 | 3x over-sampling using arithmetic baud rate generation. |
| 0x5-0x7 | Reserved |

**Bit 8 – IBON** Immediate Buffer Overflow Notification
This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is asserted when a buffer overflow occurs.

| Value | Description |
|---|---|
| 0 | STATUS.BUFOVF is asserted when it occurs in the data stream. |
| 1 | STATUS.BUFOVF is asserted immediately upon buffer overflow. |

**Bit 7 – RUNSTDBY** Run In Standby
This bit defines the functionality in standby sleep mode.

This bit is not synchronized.

| RUNSTDBY | External Clock | Internal Clock |
|---|---|---|
| 0x0 | External clock is disconnected when ongoing transfer is finished. All reception is dropped. | Generic clock is disabled when ongoing transfer is finished. The device will not wake up on either Receive Start or Transfer Complete interrupt unless the appropriate ONDEMAND bits are set in the clocking chain. |
| 0x1 | Wake on Receive Start or Receive Complete interrupt. | Generic clock is enabled in all sleep modes. Any interrupt can wake up the device. |

**Bits 4:2 – MODE[2:0]** Operating Mode
These bits select the USART serial communication interface of the SERCOM.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0x0 | USART with external clock |
| 0x1 | USART with internal clock |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE is cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 26.8.2 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIFOCLR[1:0] | | | | | | RXEN | TXEN |
| Access | R/W | R/W | | | | | R/W | R/W |
| Reset | 0 | 0 | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PMODE | | | ENC | SFDE | COLDEN |
| Access | | | R/W | | | R/W | R/W | R/W |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SBMODE | | | | CHSIZE[2:0] | | |
| Access | | R/W | | | | R/W | R/W | R/W |
| Reset | | 0 | | | | 0 | 0 | 0 |

**Bits 23:22 – FIFOCLR[1:0]** FIFO Clear
When these bits are set, the corresponding FIFO will be cleared. The bits will automatically clear when SYNCBUSY.CTRLB = 0.

These bits are not enable-protected.

| FIFOCLR[1:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | TXFIFO | Clear TX FIFO |
| 0x2 | RXFIFO | Clear RX FIFO |
| 0x3 | BOTH | Clear both TX/RX FIFO |

**Bit 17 – RXEN** Receiver Enable
Writing '0' to this bit will disable the USART receiver. Disabling the receiver will flush the receive buffer and clear the FERR, PERR and BUFOVF bits in the STATUS register.

Writing '1' to CTRLB.RXEN when the USART is disabled will set CTRLB.RXEN immediately. When the USART is enabled, CTRLB.RXEN will be cleared, and SYNCBUSY.CTRLB will be set and remain set until the receiver is enabled. When the receiver is enabled, CTRLB.RXEN will read back as '1'.

Writing '1' to CTRLB.RXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as '1'.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The receiver is disabled or being enabled. |
| 1 | The receiver is enabled or will be enabled when the USART is enabled. |

**Bit 16 – TXEN** Transmitter Enable
Writing '0' to this bit will disable the USART transmitter. Disabling the transmitter will not become effective until ongoing and pending transmissions are completed.

Writing '1' to CTRLB.TXEN when the USART is disabled will set CTRLB.TXEN immediately. When the USART is enabled, CTRLB.TXEN will be cleared, and SYNCBUSY.CTRLB will be set and remain set until the transmitter is enabled. When the transmitter is enabled, CTRLB.TXEN will read back as '1'.

Writing '1' to CTRLB.TXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the transmitter is enabled, and CTRLB.TXEN will read back as '1'.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The transmitter is disabled or being enabled. |
| 1 | The transmitter is enabled or will be enabled when the USART is enabled. |

**Bit 13 – PMODE** Parity Mode
This bit selects the type of parity used when parity is enabled (CTRLA.FORM is '1'). The transmitter will automatically generate and send the parity of the transmitted data bits within each frame. The receiver will generate a parity value for the incoming data and parity bit, compare it to the parity mode and, if a mismatch is detected, STATUS.PERR will be set.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Even parity. |
| 1 | Odd parity. |

**Bit 10 – ENC** Encoding Format
This bit selects the data encoding format.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Data is not encoded. |
| 1 | Data is IrDA encoded. |

**Bit 9 – SFDE** Start of Frame Detection Enable
This bit controls whether the start-of-frame detector will wake up the device when a start bit is detected on the RxD line.

This bit is not synchronized.

| SFDE | INTENSET.RXS | INTENSET.RXC | Description |
|---|---|---|---|
| 0 | X | X | Start-of-frame detection disabled. |
| 1 | 0 | 0 | Reserved |
| 1 | 0 | 1 | Start-of-frame detection enabled. RXC wakes up the device from all sleep modes. |
| 1 | 1 | 0 | Start-of-frame detection enabled. RXS wakes up the device from all sleep modes. |
| 1 | 1 | 1 | Start-of-frame detection enabled. Both RXC and RXS wake up the device from all sleep modes. |

**Bit 8 – COLDEN** Collision Detection Enable
This bit enables collision detection.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Collision detection is not enabled. |
| 1 | Collision detection is enabled. |

**Bit 6 – SBMODE** Stop Bit Mode
This bit selects the number of stop bits transmitted.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | One stop bit. |
| 1 | Two stop bits. |

**Bits 2:0 – CHSIZE[2:0]** Character Size
These bits select the number of bits in a character.

These bits are not synchronized.

| CHSIZE[2:0] | Description |
|---|---|
| 0x0 | 8 bits |
| 0x1 | 9 bits |
| 0x2-0x4 | Reserved |
| 0x5 | 5 bits |
| 0x6 | 6 bits |
| 0x7 | 7 bits |

### 26.8.3 Baud

**Name:** BAUD
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** Enable-Protected, PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – BAUD[15:0]** Baud Value
Arithmetic Baud Rate Generation (`CTRLA.SAMPR[0]=0`):

These bits control the clock generation, as described in the SERCOM Baud Rate section.

If Fractional Baud Rate Generation (`CTRLA.SAMPR[0]=1`) bit positions 15 to 13 are replaced by FP[2:0] Fractional Part:

- **Bits 15:13 - FP[2:0]: Fractional Part**

  These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator* section.
- **Bits 12:0 - BAUD[12:0]: Baud Value**

  These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator* section.

**Related Links**

25.6.2.3 Clock Generation – Baud-Rate Generator
25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

### 26.8.4 Receive Pulse Length Register

**Name:** RXPL
**Offset:** 0x0E
**Reset:** 0x00
**Property:** Enable-Protected, PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXPL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXPL[7:0]** Receive Pulse Length
When the encoding format is set to IrDA (CTRLB.ENC=1), these bits control the minimum pulse length that is required for a pulse to be accepted by the IrDA receiver with regards to the serial engine clock period $SE_{per}$.

$PULSE \geq \left(\mathrm{RXPL} + 2\right) \cdot SE_{per}$

### 26.8.5 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 5 – RXBRK** Receive Break Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Break Interrupt Enable bit, which disables the Receive Break interrupt.

| Value | Description |
|---|---|
| 0 | Receive Break interrupt is disabled. |
| 1 | Receive Break interrupt is enabled. |

**Bit 4 – CTSIC** Clear to Send Input Change Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Clear To Send Input Change Interrupt Enable bit, which disables the Clear To Send Input Change interrupt.

| Value | Description |
|---|---|
| 0 | Clear To Send Input Change interrupt is disabled. |
| 1 | Clear To Send Input Change interrupt is enabled. |

**Bit 3 – RXS** Receive Start Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Start Interrupt Enable bit, which disables the Receive Start interrupt.

| Value | Description |
|---|---|
| 0 | Receive Start interrupt is disabled. |
| 1 | Receive Start interrupt is enabled. |

**Bit 2 – RXC** Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC** Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Transmit Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE** Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 26.8.6 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x16
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 5 – RXBRK** Receive Break Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Break Interrupt Enable bit, which enables the Receive Break interrupt.

| Value | Description |
|---|---|
| 0 | Receive Break interrupt is disabled. |
| 1 | Receive Break interrupt is enabled. |

**Bit 4 – CTSIC** Clear to Send Input Change Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Clear To Send Input Change Interrupt Enable bit, which enables the Clear To Send Input Change interrupt.

| Value | Description |
|---|---|
| 0 | Clear To Send Input Change interrupt is disabled. |
| 1 | Clear To Send Input Change interrupt is enabled. |

**Bit 3 – RXS** Receive Start Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Start Interrupt Enable bit, which enables the Receive Start interrupt.

| Value | Description |
|---|---|
| 0 | Receive Start interrupt is disabled. |
| 1 | Receive Start interrupt is enabled. |

**Bit 2 – RXC** Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC** Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE** Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 26.8.7 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R | R/W | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error
This flag is cleared by writing '1' to it.

This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. Errors that will set this flag are COLL, ISF, BUFOVF, FERR, and PERR.Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 5 – RXBRK** Receive Break
This flag is cleared by writing '1' to it.

This flag is set when auto-baud is enabled (CTRLA.FORM) and a break character is received.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 4 – CTSIC** Clear to Send Input Change
This flag is cleared by writing a '1' to it.

This flag is set when a change is detected on the CTS pin.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 3 – RXS** Receive Start
This flag is cleared by writing '1' to it.

This flag is set when a start condition is detected on the RxD line and start-of-frame detection is enabled (CTRLB.SFDE is '1').

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Start interrupt flag.

**Bit 2 – RXC** Receive Complete
This flag is cleared by reading the Data register (DATA) or by disabling the receiver.

This flag is set when there are unread data in DATA.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

**Bit 1 – TXC** Transmit Complete
This flag is cleared by writing '1' to it or by writing new data to DATA.

This flag is set when the entire frame in the transmit shift register has been shifted out and there are no new data in DATA.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 0 – DRE** Data Register Empty
This flag is cleared by writing new data to DATA.

This flag is set when DATA is empty and ready to be written.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

### 26.8.8 Status

**Name:** STATUS
**Offset:** 0x1A
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXE | COLL | ISF | CTS | BUFOVF | FERR | PERR |
| Access | | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – TXE** Transmitter Empty
This bit will always read back as zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 5 – COLL** Collision Detected
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when collision detection is enabled (CTRLB.COLDEN) and a collision is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 4 – ISF** Inconsistent Sync Field
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when the frame format is set to auto-baud (CTRLA.FORM) and a sync field not equal to 0x55 is received.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 3 – CTS** Clear to Send
This bit indicates the current level of the CTS pin when flow control is enabled (CTRLA.TXPO).

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

**Bit 2 – BUFOVF** Buffer Overflow
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when a buffer overflow condition is detected. A buffer overflow occurs when the receive buffer is full, there is a new character waiting in the receive shift register and a new start bit is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 1 – FERR** Frame Error
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set if the received character had a frame error, i.e., when the first stop bit is zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 0 – PERR** Parity Error
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set if parity checking is enabled (CTRLA.FORM is 0x1, 0x5) and a parity error is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

### 26.8.9 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CTRLB | ENABLE | SWRST |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – CTRLB** CTRLB Synchronization Busy
Writing to the CTRLB register when the SERCOM is enabled requires synchronization. When writing to CTRLB the SYNCBUSY.CTRLB bit will be set until synchronization is complete. If CTRLB is written while SYNCBUSY.CTRLB is asserted, an APB error will be generated.

| Value | Description |
|---|---|
| 0 | CTRLB synchronization is not busy. |
| 1 | CTRLB synchronization is busy. |

**Bit 1 – ENABLE** SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | Enable synchronization is not busy. |
| 1 | Enable synchronization is busy. |

**Bit 0 – SWRST** Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | SWRST synchronization is not busy. |
| 1 | SWRST synchronization is busy. |

### 26.8.10 Data

**Name:** DATA
**Offset:** 0x28
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATA[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – DATA[8:0]** Data
Reading these bits will return the contents of the Receive Data register. The register should be read only when the Receive Complete Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set. The status bits in STATUS should be read before reading the DATA value in order to get any corresponding error.

Writing these bits will write the Transmit Data register. This register should be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

### 26.8.11 Debug Control

**Name:** DBGCTRL
**Offset:** 0x30
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGSTOP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGSTOP** Debug Stop Mode
This bit controls the baud-rate generator functionality when the CPU is halted by an external debugger.

| Value | Description |
|---|---|
| 0 | The baud-rate generator continues normal operation when the CPU is halted by an external debugger. |
| 1 | The baud-rate generator is halted when the CPU is halted by an external debugger. |

### 26.8.12 FIFO Space

**Name:** FIFOSPACE
**Offset:** 0x34
**Reset:** 0x0000
**Property:** -

This register allows the user to identify the number of bytes present in each TX and RX FIFO.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | TXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 12:8 – RXSPACE[4:0]** RX FIFO Filled Space
These bits return the number filled locations in the RX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

**Bits 4:0 – TXSPACE[4:0]** TX FIFO Empty Space
These bits return the number of available locations in the TX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

### 26.8.13 FIFO CPU Pointers

**Name:** FIFOPTR
**Offset:** 0x36
**Reset:** 0x0000
**Property:** -

This register provides a copy of internal CPU TX and RX FIFO pointers.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPURDPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPUWRPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – CPURDPTR[3:0]** RX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. Reading DATA register, will return RXFIFO[CPURDPTR] location value.

**Bits 3:0 – CPUWRPTR[3:0]** TX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. When writting to DATA register, the DATA will be written to TXFIFO[CPUWRPTR] location.

# 27. SERCOM SPI – SERCOM Serial Peripheral Interface

## 27.1 Overview

The serial peripheral interface (SPI) is one of the available modes in the Serial Communication Interface (SERCOM).

The SPI uses the SERCOM transmitter and receiver configured as shown in 27.3 Block Diagram. Each side, master and slave, depicts a separate SPI containing a shift register, a transmit buffer and a two-level receive buffer. In addition, the SPI master uses the SERCOM baud-rate generator, while the SPI slave can use the SERCOM address match logic. Labels in capital letters are synchronous to CLK_SERCOMx_APB and accessible by the CPU, while labels in lowercase letters are synchronous to the SCK clock.

**Related Links**

25. SERCOM – Serial Communication Interface

## 27.2 Features

SERCOM SPI includes the following features:

- Full-duplex, four-wire interface (MISO, MOSI, SCK, $\overline{SS}$)
- One-level transmit buffer, two-level receive buffer
- Supports all four SPI modes of operation
- Single data direction operation allows alternate function on MISO or MOSI pin
- Selectable LSB- or MSB-first data transfer
- Can be used with DMA
- Up to 16-bytes internal FIFO
- Master operation:
  - Serial clock speed, $f_{SCK}=1/t_{SCK}$(1)
  - 8-bit clock generator
  - Hardware controlled $\overline{SS}$
- Slave Operation:
  - Serial clock speed, $f_{SCK}=1/t_{SSCK}$(1)
  - Optional 8-bit address match operation
  - Operation in all sleep modes
  - Wake on $\overline{SS}$ transition

1. For $t_{SCK}$ and $t_{SSCK}$ values, refer to SPI Timing Characteristics.

**Related Links**

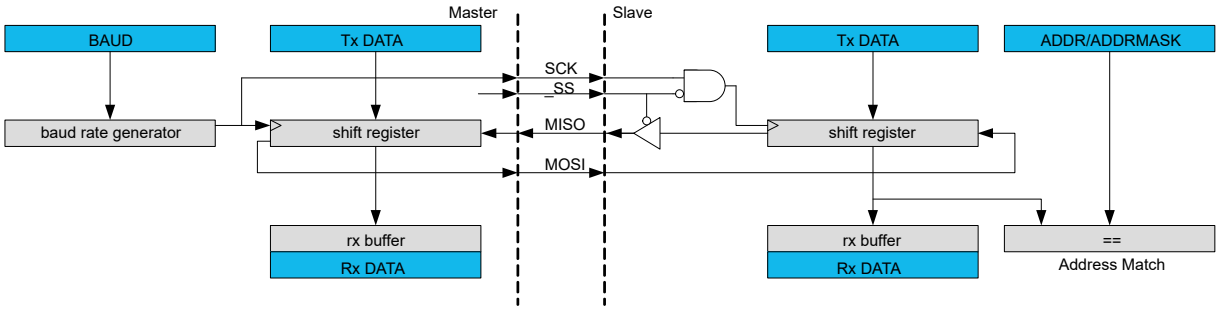
37.15.2 SERCOM in SPI Mode Timing

25. SERCOM – Serial Communication Interface

25.2 Features

## 27.3 Block Diagram

**Figure 27-1. Full-Duplex SPI Master Slave Interconnection**




## 27.4 Signal Description

**Table 27-1. SERCOM SPI Signals**

| Signal Name | Type | Description |
|---|---|---|
| PAD[3:0] | Digital I/O | General SERCOM pins |

One signal can be mapped to one of several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 27.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 27.5.1 I/O Lines

In order to use the SERCOM's I/O lines, the I/O pins must be configured using the IO Pin Controller (PORT).

When the SERCOM is configured for SPI operation, the SERCOM controls the direction and value of the I/O pins according to the table below. Both PORT control bits PINCFGn.PULLEN and PINCFGn.DRVSTR are still effective. If the receiver is disabled, the data input pin can be used for other purposes. In master mode, the slave select line ($\overline{SS}$) is hardware controlled when the Master Slave Select Enable bit in the Control B register (CTRLB.MSSEN) is '1'.

**Table 27-2. SPI Pin Configuration**

| Pin | Master SPI | Slave SPI |
|---|---|---|
| MOSI | Output | Input |
| MISO | Input | Output |
| SCK | Output | Input |
| $\overline{SS}$ | Output (CTRLB.MSSEN=1) | Input |

The combined configuration of PORT, the Data In Pinout and the Data Out Pinout bit groups in the Control A register (CTRLA.DIPO and CTRLA.DOPO) define the physical position of the SPI signals in the table above.

**Related Links**

23. PORT - I/O Pin Controller

### 27.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes.

**Related Links**

16. PM – Power Manager

### 27.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the Power Manager. Refer to *Peripheral Clock Masking* for details and default status of this clock.

A generic clock (GCLK_SERCOMx_CORE) is required to clock the SPI. This clock must be configured and enabled in the Generic Clock Controller before using the SPI.

This generic clock is asynchronous to the bus clock (CLK_SERCOMx_APB). Therefore, writes to certain registers will require synchronization to the clock domains.

**Related Links**

15. GCLK - Generic Clock Controller
16.6.2.6 Peripheral Clock Masking
27.6.6 Synchronization

### 27.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 27.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 27.5.6 Events

Not applicable.

### 27.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

### 27.5.8 Register Access Protection

Registers with write-access can be write-protected optionally by the peripheral access controller (PAC).

PAC Write-Protection is not available for the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)

Optional PAC Write-Protection is denoted by the "PAC Write-Protection" property in each individual register description.

Write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 27.5.9 Analog Connections

Not applicable.

## 27.6 Functional Description

### 27.6.1 Principle of Operation

The SPI is a high-speed synchronous data transfer interface. It allows high-speed communication between the device and peripheral devices.

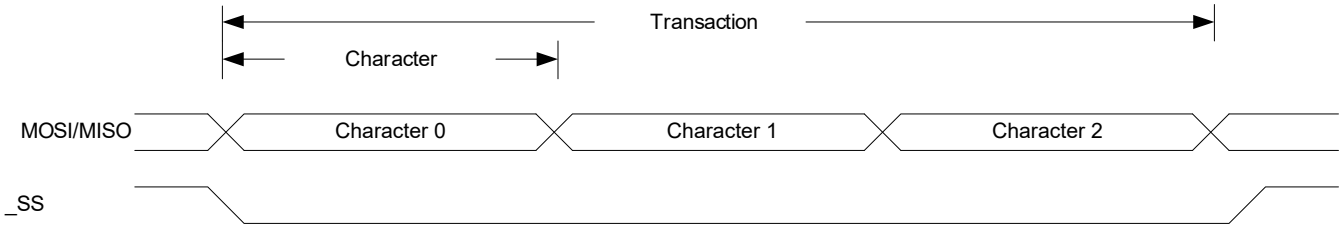
The SPI can operate as master or slave. As master, the SPI initiates and controls all data transactions. The SPI is single buffered for transmitting and double buffered for receiving.

When transmitting data, the Data register can be loaded with the next character to be transmitted during the current transmission.

When receiving, the data is transferred to the two-level receive buffer, and the receiver is ready for a new character.

The SPI transaction format is shown in SPI Transaction Format. Each transaction can contain one or more characters. The character size is configurable, and can be either 8 or 9 bits.

**Figure 27-2. SPI Transaction Format**




The SPI master must pull the slave select line ($\overline{SS}$) of the desired slave low to initiate a transaction. The master and slave prepare data to send via their respective shift registers, and the master generates the serial clock on the SCK line.

Data are always shifted from master to slave on the Master Output Slave Input line (MOSI); data is shifted from slave to master on the Master Input Slave Output line (MISO).

Each time character is shifted out from the master, a character will be shifted out from the slave simultaneously. To signal the end of a transaction, the master will pull the $\overline{SS}$ line high

### 27.6.2 Basic Operation

#### 27.6.2.1 Initialization

The following registers are enable-protected, meaning that they can only be written when the SPI is disabled (CTRL.ENABLE=0):

- Control A register (CTRLA), except Enable (CTRLA.ENABLE) and Software Reset (CTRLA.SWRST)
- Control B register (CTRLB), except Receiver Enable (CTRLB.RXEN)
- Baud register (BAUD)
- Address register (ADDR)

When the SPI is enabled or is being enabled (CTRLA.ENABLE=1), any writing to these registers will be discarded.

when the SPI is being disabled, writing to these registers will be completed after the disabling.

Enable-protection is denoted by the Enable-Protection property in the register description.

Initialize the SPI by following these steps:

1. Select SPI mode in master / slave operation in the Operating Mode bit group in the CTRLA register (CTRLA.MODE= 0x2 or 0x3 ).
2. Select transfer mode for the Clock Polarity bit and the Clock Phase bit in the CTRLA register (CTRLA.CPOL and CTRLA.CPHA) if desired.
3. Select the Frame Format value in the CTRLA register (CTRLA.FORM).
4. Configure the Data In Pinout field in the Control A register (CTRLA.DIPO) for SERCOM pads of the receiver.
5. Configure the Data Out Pinout bit group in the Control A register (CTRLA.DOPO) for SERCOM pads of the transmitter.
6. Select the Character Size value in the CTRLB register (CTRLB.CHSIZE).
7. Write the Data Order bit in the CTRLA register (CTRLA.DORD) for data direction.
8. If the SPI is used in master mode:
    8.1. Select the desired baud rate by writing to the Baud register (BAUD).
    8.2. If Hardware SS control is required, write '1' to the Master Slave Select Enable bit in CTRLB register (CTRLB.MSSEN).
9. Enable the receiver by writing the Receiver Enable bit in the CTRLB register (CTRLB.RXEN=1).

#### 27.6.2.2 Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

Refer to the CTRLA register description for details.

#### 27.6.2.3 Clock Generation

In SPI master operation (CTRLA.MODE=0x3), the serial clock (SCK) is generated internally by the SERCOM baud-rate generator.

In SPI mode, the baud-rate generator is set to synchronous mode. The 8-bit Baud register (BAUD) value is used for generating SCK and clocking the shift register. Refer to *Clock Generation – Baud-Rate Generator* for more details.

In SPI slave operation (CTRLA.MODE is 0x2), the clock is provided by an external master on the SCK pin. This clock is used to directly clock the SPI shift register.

**Related Links**

25.6.2.3 Clock Generation – Baud-Rate Generator

25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

#### 27.6.2.4 Data Register

The SPI Transmit Data register (TxDATA) and SPI Receive Data register (RxDATA) share the same I/O address, referred to as the SPI Data register (DATA). Writing DATA register will update the Transmit Data register. Reading the DATA register will return the contents of the Receive Data register.

#### 27.6.2.5 SPI Transfer Modes

There are four combinations of SCK phase and polarity to transfer serial data. The SPI data transfer modes are shown in SPI Transfer Modes (Table) and SPI Transfer Modes (Figure).

SCK phase is configured by the Clock Phase bit in the CTRLA register (CTRLA.CPHA). SCK polarity is programmed by the Clock Polarity bit in the CTRLA register (CTRLA.CPOL). Data bits are shifted out and latched in on opposite edges of the SCK signal. This ensures sufficient time for the data signals to stabilize.

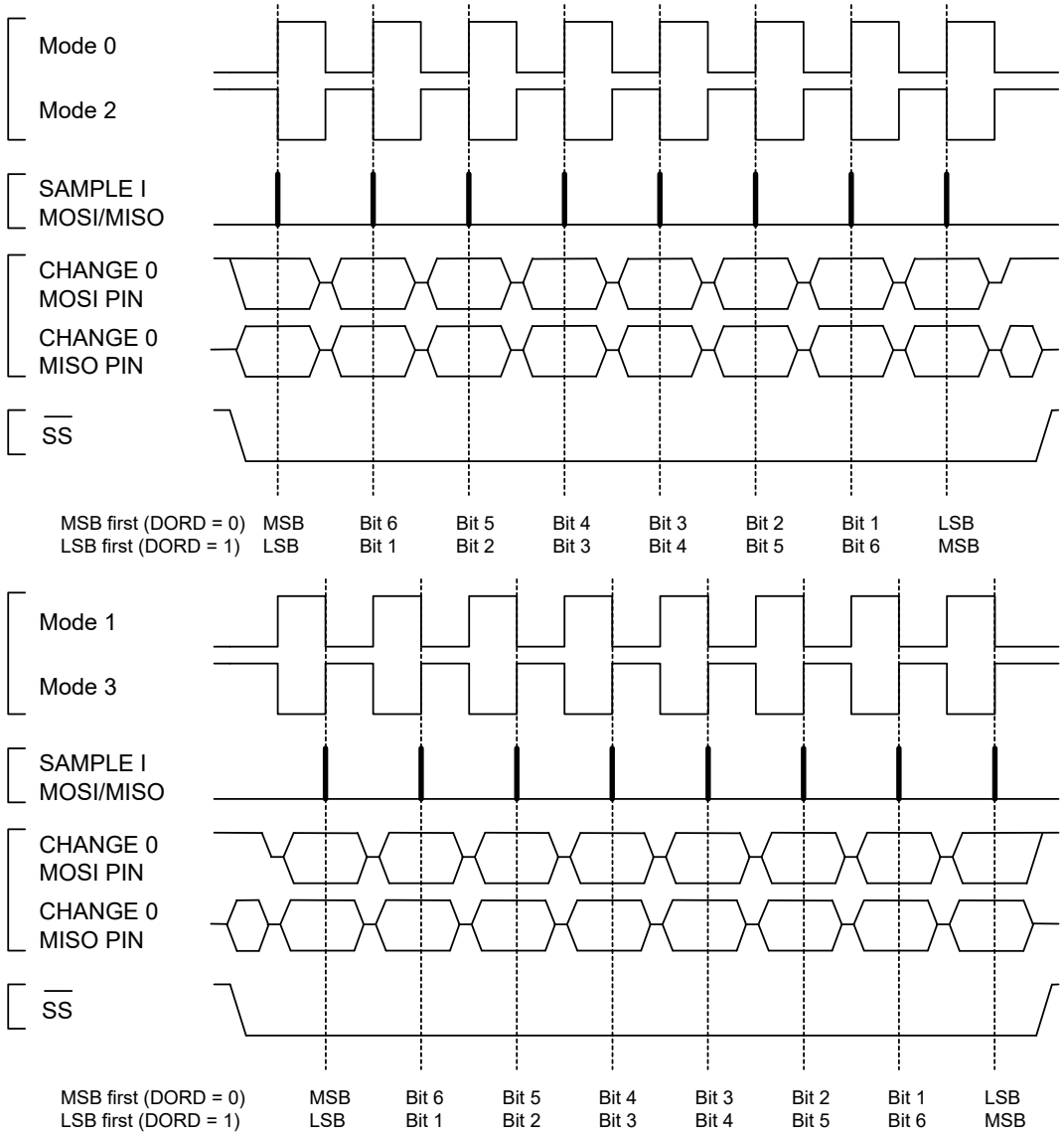
**Table 27-3. SPI Transfer Modes**

| Mode | CPOL | CPHA | Leading Edge | Trailing Edge |
|---|---|---|---|---|
| 0 | 0 | 0 | Rising, sample | Falling, setup |
| 1 | 0 | 1 | Rising, setup | Falling, sample |
| 2 | 1 | 0 | Falling, sample | Rising, setup |
| 3 | 1 | 1 | Falling, setup | Rising, sample |

**Note:**
Leading edge is the first clock edge in a clock cycle.

Trailing edge is the second clock edge in a clock cycle.

**Figure 27-3. SPI Transfer Modes**




### 27.6.2.6 Transferring Data

#### 27.6.2.6.1 Master

In master mode (CTRLA.MODE=0x3), when Master Slave Enable Select (CTRLB.MSSEN) is ‘1’, hardware will control the $\overline{SS}$ line.

When Master Slave Select Enable (CTRLB.MSSEN) is '0', the $\overline{SS}$ line must be configured as an output. $\overline{SS}$ can be assigned to any general purpose I/O pin. When the SPI is ready for a data transaction, software must pull the $\overline{SS}$ line low.

When writing a character to the Data register (DATA), the character will be transferred to the shift register. Once the content of TxDATA has been transferred to the shift register, the Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) will be set. And a new character can be written to DATA.

Each time one character is shifted out from the master, another character will be shifted in from the slave simultaneously. If the receiver is enabled (CTRLA.RXEN=1), the contents of the shift register will be

transferred to the two-level receive buffer. The transfer takes place in the same clock cycle as the last data bit is shifted in. And the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) will be set. The received data can be retrieved by reading DATA.

When the last character has been transmitted and there is no valid data in DATA, the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set. When the transaction is finished, the master must pull the $\overline{SS}$ line high to notify the slave. If Master Slave Select Enable (CTRLB.MSSEN) is set to '0', the software must pull the $\overline{SS}$ line high.

#### 27.6.2.6.2 Slave

In slave mode (CTRLA.MODE=0x2), the SPI interface will remain inactive with the MISO line tri-stated as long as the $\overline{SS}$ pin is pulled high. Software may update the contents of DATA at any time as long as the Data Register Empty flag in the Interrupt Status and Clear register (INTFLAG.DRE) is set.

When $\overline{SS}$ is pulled low and SCK is running, the slave will sample and shift out data according to the transaction mode set. When the content of TxDATA has been loaded into the shift register, INTFLAG.DRE will be set, and new data can be written to DATA.

Similar to the master, the slave will receive one character for each character transmitted. A character will be transferred into the two-level receive buffer within the same clock cycle its last data bit is received. The received character can be retrieved from DATA when the Receive Complete interrupt flag (INTFLAG.RXC) is set.

When the master pulls the $\overline{SS}$ line high, the transaction is done and the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set.

After DATA is written it takes up to three SCK clock cycles until the content of DATA is ready to be loaded into the shift register on the next character boundary. As a consequence, the first character transferred in a SPI transaction will not be the content of DATA. This can be avoided by using the preloading feature. Refer to 27.6.3.2 Preloading of the Slave Shift Register.

When transmitting several characters in one SPI transaction, the data has to be written into DATA register with at least three SCK clock cycles left in the current character transmission. If this criteria is not met, the previously received character will be transmitted.

Once the DATA register is empty, it takes three CLK_SERCOM_APB cycles for INTFLAG.DRE to be set.

### 27.6.2.7 Receiver Error Bit

The SPI receiver has one error bit: the Buffer Overflow bit (BUFOVF), which can be read from the Status register (STATUS). Once an error happens, the bit will stay set until it is cleared by writing '1' to it. The bit is also automatically cleared when the receiver is disabled.

There are two methods for buffer overflow notification, selected by the immediate buffer overflow notification bit in the Control A register (CTRLA.IBON):

If CTRLA.IBON=1, STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA until the receiver complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) goes low.

If CTRLA.IBON=0, the buffer overflow condition travels with data through the receive FIFO. After the received data is read, STATUS.BUFOVF and INTFLAG.ERROR will be set along with INTFLAG.RXC, and RxDATA will be zero.

### 27.6.3 Additional Features

#### 27.6.3.1 Address Recognition

When the SPI is configured for slave operation (CTRLA.MODE=0x2) with address recognition (CTRLA.FORM is 0x2), the SERCOM address recognition logic is enabled: the first character in a transaction is checked for an address match.

If there is a match, the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set, the MISO output is enabled, and the transaction is processed. If the device is in sleep mode, an address match can wake up the device in order to process the transaction.

If there is no match, the complete transaction is ignored.

If a 9-bit frame format is selected, only the lower 8 bits of the shift register are checked against the Address register (ADDR).

Preload must be disabled (CTRLB.PLOADEN=0) in order to use this mode.

**Related Links**

25.6.3.1 Address Match and Mask

#### 27.6.3.2 Preloading of the Slave Shift Register

When starting a transaction, the slave will first transmit the contents of the shift register before loading new data from DATA. The first character sent can be either the reset value of the shift register (if this is the first transmission since the last reset) or the last character in the previous transmission.

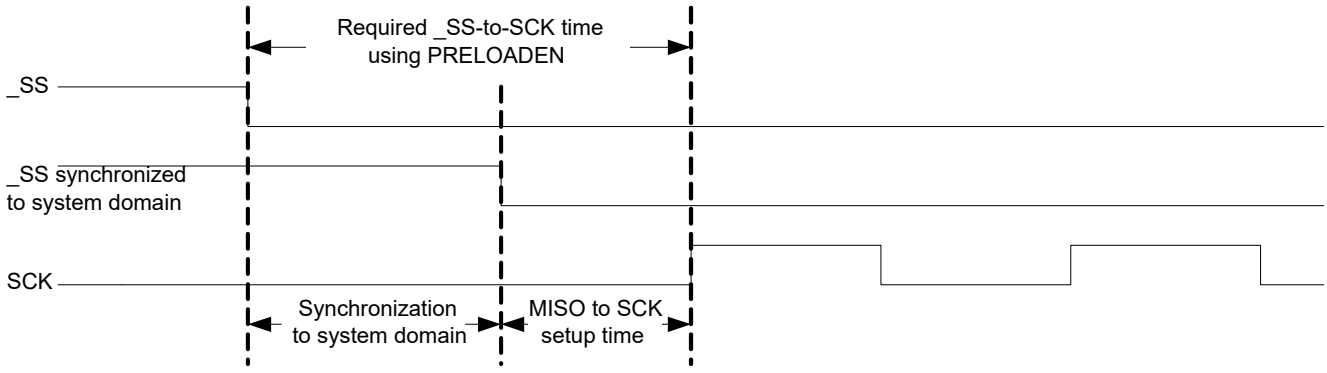
Preloading can be used to preload data into the shift register while $\overline{SS}$ is high: this eliminates sending a dummy character when starting a transaction. If the shift register is not preloaded, the current contents of the shift register will be shifted out.

Only one data character will be preloaded into the shift register while the synchronized $\overline{SS}$ signal is high. If the next character is written to DATA before $\overline{SS}$ is pulled low, the second character will be stored in DATA until transfer begins.

For proper preloading, sufficient time must elapse between $\overline{SS}$ going low and the first SCK sampling edge, as in Timing Using Preloading. See also *Electrical Characteristics* for timing details.

Preloading is enabled by writing '1' to the Slave Data Preload Enable bit in the CTRLB register (CTRLB.PLOADEN).

**Figure 27-4. Timing Using Preloading**



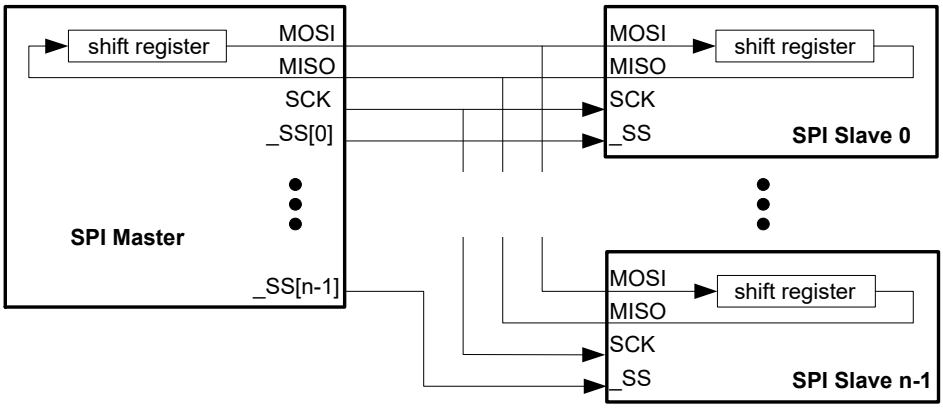

**Related Links**

37. Electrical Characteristics

#### 27.6.3.3 Master with Several Slaves

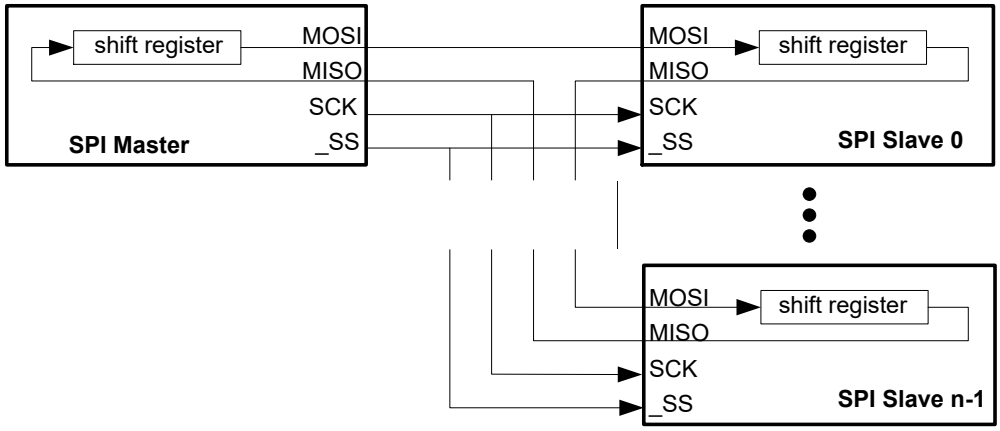
Master with multiple slaves in parallel is only available when Master Slave Select Enable (CTRLB.MSSEN) is set to zero and hardware $\overline{SS}$ control is disabled. If the bus consists of several SPI slaves, an SPI master can use general purpose I/O pins to control the $\overline{SS}$ line to each of the slaves on the bus, as shown in Multiple Slaves in Parallel. In this configuration, the single selected SPI slave will drive the tri-state MISO line.

**Figure 27-5. Multiple Slaves in Parallel**




Another configuration is multiple slaves in series, as in Multiple Slaves in Series. In this configuration, all n attached slaves are connected in series. A common $\overline{SS}$ line is provided to all slaves, enabling them simultaneously. The master must shift n characters for a complete transaction. Depending on the Master Slave Select Enable bit (CTRLB.MSSEN), the $\overline{SS}$ line can be controlled either by hardware or user software and normal GPIO.

**Figure 27-6. Multiple Slaves in Series**




#### 27.6.3.4 Loop-Back Mode

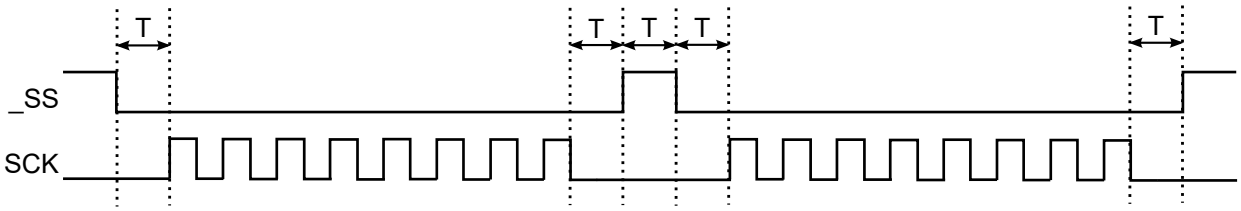
For loop-back mode, configure the Data In Pinout (CTRLA.DIPO) and Data Out Pinout (CTRLA.DOPO) to use the same data pins for transmit and receive. The loop-back is through the pad, so the signal is also available externally.

#### 27.6.3.5 Hardware Controlled $\overline{SS}$

In master mode, a single $\overline{SS}$ chip select can be controlled by hardware by writing the Master Slave Select Enable (CTRLB.MSSEN) bit to '1'. In this mode, the $\overline{SS}$ pin is driven low for a minimum of one baud cycle before transmission begins, and stays low for a minimum of one baud cycle after transmission completes. If back-to-back frames are transmitted, the $\overline{SS}$ pin will always be driven high for a minimum of one baud cycle between frames.

In Hardware Controlled $\overline{SS}$, the time T is between one and two baud cycles depending on the SPI transfer mode.

**Figure 27-7. Hardware Controlled $\overline{SS}$**




T = 1 to 2 baud cycles

When CTRLB.MSSEN=0, the $\overline{SS}$ pin(s) is/are controlled by user software and normal GPIO.

#### 27.6.3.6 Slave Select Low Detection

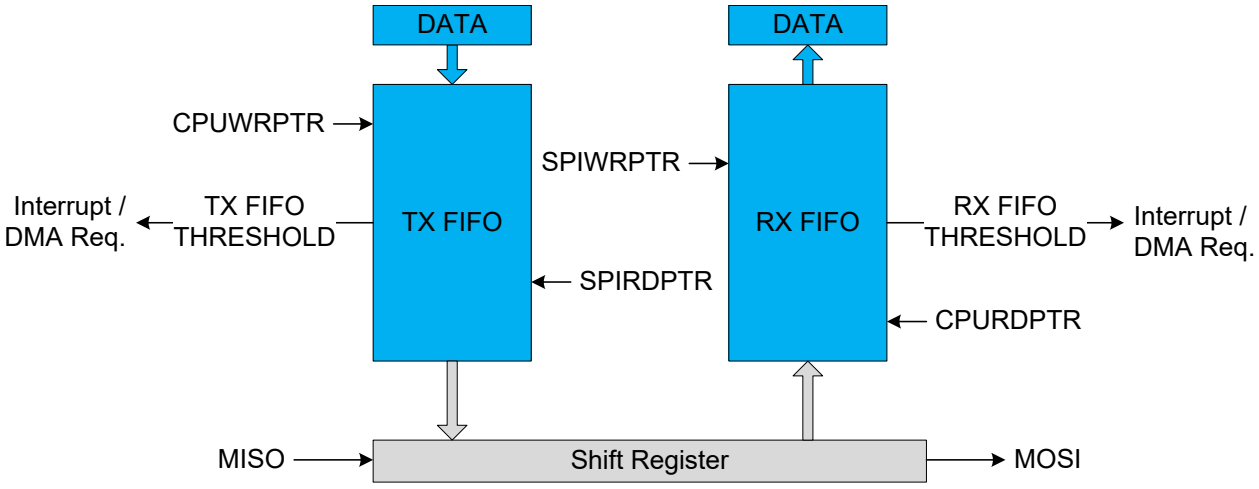
In slave mode, the SPI can wake the CPU when the slave select ($\overline{SS}$) goes low. When the Slave Select Low Detect is enabled (CTRLB.SSDE=1), a high-to-low transition will set the Slave Select Low interrupt flag (INTFLAG.SSL) and the device will wake up if applicable.

#### 27.6.3.7 FIFO Operation

The SPI embeds up to 16-bytes FIFO capability. The receive / transmit buffer is considered to have the FIFO mode enabled when the FIFOEN bit in CTRLC register is set to a '1' (CTRLC.FIFOEN = 1). By default, the FIFO can act as a 16-by-8-bit array, or as a 4-by-32-bit array, depending on the setting of the CTRLC.DATA32B bit.

The hardware around this array implements four pointers, called the CPU Write Pointer (CPUWRPTR), the CPU Read Pointer (CPURDPTR), the SPI Write pointer (SPIWRPTR) and the SPI Read pointer (SPIRDPTR). All of these pointers reset to '0'. The CPUWRPTR and CPURDPTR pointers are native to the CPU clock domain, while the SPIWRPTR and SPIRDPTR are native to the SPI domain. The location pointed to by the CPUWRPTR is the current TX FIFO. The location pointed to by the CPURDPTR becomes the current RX FIFO. Writes to DATA register by the CPU will point to TX FIFO. Reads to DATA register by the CPU will point to RX FIFO. The location pointed to by the SPIWRPTR / SPIRDPTR is logically the current shift register. Physically, the receive (shift-in) portion of the shift register is a single register located in the SCK clock domain, and the transmit (shift-out) portion is the buffer location pointed to by the SPIRDPTR. When a full word / byte is clocked into the SPI shift register, it is copied into the location pointed by SPIWRPTR.

**Figure 27-8. FIFO Overview**



The interrupts and DMA triggers are generated according to FIFO threshold settings in Control C register (CTRLC.TXTRHOLD, CTRLC.RXTRHOLD).

The Data Register Empty interrupt flag, and the DMA TX trigger respectivly, are generated when the available place in the TX FIFO is equal or higher than the threshold value defined by the CTRLC.TXTRHOLD settings. The Transfer complete interrupt is generated when the TX FIFO is empty and the last bit is shifted out.

The Receive Complete interrupt flag, and the DMA RX trigger respectivly, are generated when the number of bytes present in the RX FIFO equals or is higher than the threshold value defined by the CTRLC.RXTRHOLD settings. The ERROR interrupt flag is generated when both RX shifter and the RX FIFO are full.

The FIFO is fully accessible if the SERCOM is halted, by writting the corresponding CPU FIFO pointer in the FIFOPTR register. The RX or TX FIFO can be individually cleared, by setting the respective FIFO Clear bit in the Control B register (CTRLB.FIFOCLR). The FIFO Clear must be written before data transfer begins. Writing the FIFO Clear bits while a frame is in progress will produce unpredicatable results.

#### 27.6.3.7.1 Pointer Operation in Master Mode

In Master mode, the transmit / receive sequence is started by the CPU writing one or more transmit words into the TX FIFO. The CPUWRPTR is incremented by 1 every time the CPU writes a word to the memory array. As soon as the CPUWRPTR becomes not-equal to SPIRDPTR (FIFOSPACE.TXSPACE != 0), the SPI transmits the data pointed to by the SPIRDPTR through MOSI. When a complete word is shifted in, the SPIRDPTR is compared to CPUWRPTR. If they are not equal, SPIRDPTR is incremented, another byte / word is shifted in/out, and so on. When the CPU completes a read from the RX FIFO location (FIFOSPACE.RXSPACE != 0), the CPURDPTR pointer is incremented. When both RX shifter and RX FIFO are full, the Buffer Overflow status bit is set (STATUS.BUFOVF) and optional ERROR interrupt is generated. The module will not respond to SCK transitions while BUFOVF is ‘1’, effectively disabling the module until software reads DATA register.

All pointers increment to their maximum value, dictated by CTRLC.DATA32B bit, and then rolls over to ‘0’. CPURDPTR will not increment past SPIWRPTR. In other words, if CPURDPTR = SPIWRPTR, and the CPU attempts another read, the pointer will stay at the value of SPIWRPTR.

#### 27.6.3.7.2 Pointer Operation in Slave Mode

In Slave mode, the transmit / receive sequence is started by the SPI receiving an SCK clock pulse. As soon as an SCK pulse is received, the SPI transmits the data pointed to by the SPIRDPTR. When a complete data is shifted in, the SPIRDPTR / SPIWRPTR are incremented, another data are shifted in/out, and so on. The newly received data is written to the RX FIFO location pointed to by SPIWRPTR. The CPUWRPTR is incremented by one every time the CPU writes a new data to the TX FIFO memory array. If the CPUWRPTR is pointing to location n, and SPIWRPTR is pointing to location n, and a wrap is not detected, the CPUWRPTR will auto-increment to SPIWRPTR, to keep up with SPIWRPTR. This is so that the next data to be transmitted will be placed in the correct position in the storage element. The CPU can read data from RX FIFO as long as FIFOSPACE.RXSPACE != 0. When both RX shifter and RX FIFO are full, the Buffer Overflow status bit is set (STATUS.BUFOVF) and optional ERROR interrupt is generated. The module will not respond to SCK transitions while BUFOVF is ‘1’, effectively disabling the module until software reads RX FIFO. All pointers increment to their maximum value, dictated by CTRLC.DATA32B bit, and then roll over to ‘0’. CPURDPTR will not increment past SPIWRPTR. In other words, if CPURDPTR = SPIWRPTR, and the CPU attempts another read, the pointer will stay at the value of SPIWRPTR.

### 27.6.4 DMA, Interrupts, and Events

**Table 27-4. Module Request for SERCOM SPI**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Data Register Empty (DRE) | Yes<br>(request cleared when data is written) | Yes | NA |
| Receive Complete (RXC) | Yes<br>(request cleared when data is read) | Yes | |
| Transmit Complete (TXC) | NA | Yes | |
| Slave Select low (SSL) | NA | Yes | |
| Error (ERROR) | NA | Yes | |

**Table 27-5. Module Request for SERCOM SPI**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Standard (DRE): Data Register Empty<br>FIFO (DRE): at least TXTRHOLD locations in TX FIFO are empty | Yes<br>(request cleared when data is written) | Yes | NA |
| Standard (RXC): Receive Complete<br>FIFO (RXC): at least RXTRHOLD data available in RX FIFO, or a last word available and length frame reception completed. | Yes<br>(request cleared when data is read) | Yes | |
| Standard (TXC): Transmit Complete<br>FIFO (TXC): Transmit Complete and TX FIFO is empty | NA | Yes | |
| Slave Select low (SSL) | NA | Yes | |
| Error (ERROR) | NA | Yes | |

#### 27.6.4.1 DMA Operation

The SPI generates the following DMA requests:

- Data received (RX): The request is set when data is available in the receive FIFO. The request is cleared when DATA is read.
- Data transmit (TX): The request is set when the transmit buffer (TX DATA) is empty. The request is cleared when DATA is written.
- Data received (RX): The request is set when data is available in the receive FIFO or if at least RXTRHOLD data are available in the RX FIFO when FIFO operation is enabled. The request is cleared when DATA is read.

- Data transmit (TX): The request is set when the transmit buffer (TX DATA) is empty or if at least TXTRHOLD data locations are empty in the TX FIFO, when FIFO operation is enabled. The request is cleared when DATA is written.

#### 27.6.4.2 Interrupts

The SPI has the following interrupt sources. These are asynchronous interrupts, and can wake up the device from any sleep mode:

- Data Register Empty (DRE)
- Receive Complete (RXC)
- Transmit Complete (TXC)
- Slave Select Low (SSL)
- Error (ERROR)

Each interrupt source has its own interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is met. Each interrupt can be individually enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and if the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled, or the SPI is reset. For details on clearing interrupt flags, refer to the INTFLAG register description.

The value of INTFLAG indicates which interrupt is executed. Note that interrupts must be globally enabled for interrupt requests. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 27.6.4.3 Events

Not applicable.

### 27.6.5 Sleep Mode Operation

The behavior in sleep mode is depending on the master/slave configuration and the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY):

- Master operation, CTRLA.RUNSTDBY=1: The peripheral clock GCLK_SERCOM_CORE will continue to run in idle sleep mode and in standby sleep mode. Any interrupt can wake up the device.
- Master operation, CTRLA.RUNSTDBY=0: GLK_SERCOMx_CORE will be disabled after the ongoing transaction is finished. Any interrupt can wake up the device.
- Slave operation, CTRLA.RUNSTDBY=1: The Receive Complete interrupt can wake up the device.
- Slave operation, CTRLA.RUNSTDBY=0: All reception will be dropped, including the ongoing transaction.

### 27.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in the CTRLA register (CTRLA.SWRST)

- Enable bit in the CTRLA register (CTRLA.ENABLE)
- Receiver Enable bit in the CTRLB register (CTRLB.RXEN)

**Note:** CTRLB.RXEN is write-synchronized somewhat differently. See also *CTRLB* register for details.

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 27.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| | | 15:8 | | | | | | | | IBON |
| | | 23:16 | | | DIPO[1:0] | | | | DOPO[1:0] | |
| | | 31:24 | | DORD | CPOL | CPHA | FORM[3:0] | | | |
| 0x04 | CTRLB | 7:0 | | PLOADEN | | | | CHSIZE[2:0] | | |
| | | 15:8 | AMODE[1:0] | | MSSEN | | | | SSDE | |
| | | 23:16 | FIFOCLR[1:0] | | | | | | RXEN | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | BAUD | 7:0 | BAUD[7:0] | | | | | | | |
| 0x0D ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | ERROR | | | | SSL | RXC | TXC | DRE |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENSET | 7:0 | ERROR | | | | SSL | RXC | TXC | DRE |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTFLAG | 7:0 | ERROR | | | | SSL | RXC | TXC | DRE |
| 0x19 | Reserved | | | | | | | | | |
| 0x1A | STATUS | 7:0 | | | | | | BUFOVF | | |
| | | 15:8 | | | | | | | | |
| 0x1C | SYNCBUSY | 7:0 | | | | | | CTRLB | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | ADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | ADDRMASK[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | DATA[8:8] |
| 0x2A ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | DBGCTRL | 7:0 | | | | | | | | DBGSTOP |
| 0x31 ... 0x33 | Reserved | | | | | | | | | |
| 0x34 | FIFOSPACE | 7:0 | | | | TXSPACE[4:0] | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 15:8 | | | | RXSPACE[4:0] | | | | |
| 0x36 | FIFOPTR | 7:0 | | | | | CPUWRPTR[3:0] | | | |
| | | 15:8 | | | | | CPURDPTR[3:0] | | | |

## 27.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Refer to 27.6.6 Synchronization

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Refer to 27.5.8 Register Access Protection.

### 27.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | DORD | CPOL | CPHA | FORM[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | DIPO[1:0] | | | | DOPO[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | IBON |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – DORD** Data Order
This bit selects the data order when a character is shifted out from the shift register.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | MSB is transferred first. |
| 1 | LSB is transferred first. |

**Bit 29 – CPOL** Clock Polarity
In combination with the Clock Phase bit (CPHA), this bit determines the SPI transfer mode.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | SCK is low when idle. The leading edge of a clock cycle is a rising edge, while the trailing edge is a falling edge. |
| 1 | SCK is high when idle. The leading edge of a clock cycle is a falling edge, while the trailing edge is a rising edge. |

**Bit 28 – CPHA** Clock Phase
In combination with the Clock Polarity bit (CPOL), this bit determines the SPI transfer mode.

This bit is not synchronized.

| Mode | CPOL | CPHA | Leading Edge | Trailing Edge |
|---|---|---|---|---|
| 0x0 | 0 | 0 | Rising, sample | Falling, change |
| 0x1 | 0 | 1 | Rising, change | Falling, sample |
| 0x2 | 1 | 0 | Falling, sample | Rising, change |
| 0x3 | 1 | 1 | Falling, change | Rising, sample |

| Value | Description |
|---|---|
| 0 | The data is sampled on a leading SCK edge and changed on a trailing SCK edge. |
| 1 | The data is sampled on a trailing SCK edge and changed on a leading SCK edge. |

**Bits 27:24 – FORM[3:0]** Frame Format
This bit field selects the various frame formats supported by the SPI in slave mode. When the 'SPI frame with address' format is selected, the first byte received is checked against the ADDR register.

| FORM[3:0] | Name | Description |
|---|---|---|
| 0x0 | SPI | SPI frame |
| 0x1 | - | Reserved |
| 0x2 | SPI_ADDR | SPI frame with address |
| 0x3-0xF | - | Reserved |

**Bits 21:20 – DIPO[1:0]** Data In Pinout
These bits define the data in (DI) pad configurations.

In master operation, DI is MISO.

In slave operation, DI is MOSI.

These bits are not synchronized.

| DIPO[1:0] | Name | Description |
|---|---|---|
| 0x0 | PAD[0] | SERCOM PAD[0] is used as data input |
| 0x1 | PAD[1] | SERCOM PAD[1] is used as data input |
| 0x2 | PAD[2] | SERCOM PAD[2] is used as data input |
| 0x3 | PAD[3] | SERCOM PAD[3] is used as data input |

**Bits 17:16 – DOPO[1:0]** Data Out Pinout
This bit defines the available pad configurations for data out (DO) and the serial clock (SCK). In slave operation, the slave select line ($\overline{SS}$) is controlled by DOPO, while in master operation the $\overline{SS}$ line is controlled by the port configuration.

In master operation, DO is MOSI.

In slave operation, DO is MISO.

These bits are not synchronized.

| DOPO | DO | SCK | Slave $\overline{SS}$ | Master $\overline{SS}$ |
|---|---|---|---|---|
| 0x0 | PAD[0] | PAD[1] | PAD[2] | System configuration |
| 0x1 | PAD[2] | PAD[3] | PAD[1] | System configuration |
| 0x2 | PAD[3] | PAD[1] | PAD[2] | System configuration |
| 0x3 | PAD[0] | PAD[3] | PAD[1] | System configuration |

**Bit 8 – IBON** Immediate Buffer Overflow Notification
This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is set when a buffer overflow occurs.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | STATUS.BUFOVF is set when it occurs in the data stream. |
| 1 | STATUS.BUFOVF is set immediately upon buffer overflow. |

**Bit 7 – RUNSTDBY** Run In Standby
This bit defines the functionality in standby sleep mode.

These bits are not synchronized.

| RUNSTDBY | Slave | Master |
|---|---|---|
| 0x0 | Disabled. All reception is dropped, including the ongoing transaction. | Generic clock is disabled when ongoing transaction is finished. All interrupts can wake up the device. |
| 0x1 | Ongoing transaction continues, wake on Receive Complete interrupt. | Generic clock is enabled while in sleep modes. All interrupts can wake up the device. |

**Bits 4:2 – MODE[2:0]** Operating Mode
These bits must be written to 0x2 or 0x3 to select the SPI serial communication interface of the SERCOM.

0x2: SPI slave operation

0x3: SPI master operation

These bits are not synchronized.

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Enable Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE is cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing "1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY. SWRST will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 27.8.2 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIFOCLR[1:0] | | | | | | RXEN | |
| Access | R/W | R/W | | | | | R/W | |
| Reset | 0 | 0 | | | | | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AMODE[1:0] | | MSSEN | | | | SSDE | |
| Access | R/W | R/W | R/W | | | | R/W | |
| Reset | 0 | 0 | 0 | | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PLOADEN | | | | CHSIZE[2:0] | | |
| Access | | R/W | | | | R/W | R/W | R/W |
| Reset | | 0 | | | | 0 | 0 | 0 |

**Bits 23:22 – FIFOCLR[1:0]** FIFO Clear
When these bits are set, the corresponding FIFO will be cleared. The bits will automatically clear when SYNCBUSY.CTRLB = 0.

These bits are not enable-protected.

| FIFOCLR[1:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | TXFIFO | Clear TX FIFO |
| 0x2 | RXFIFO | Clear RX FIFO |
| 0x3 | BOTH | Clear both TX/RX FIFO |

**Bit 17 – RXEN** Receiver Enable
Writing '0' to this bit will disable the SPI receiver immediately. The receive buffer will be flushed, data from ongoing receptions will be lost and STATUS.BUFOVF will be cleared.

Writing '1' to CTRLB.RXEN when the SPI is disabled will set CTRLB.RXEN immediately. When the SPI is enabled, CTRLB.RXEN will be cleared, SYNCBUSY.CTRLB will be set and remain set until the receiver is enabled. When the receiver is enabled CTRLB.RXEN will read back as '1'.

Writing '1' to CTRLB.RXEN when the SPI is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as '1'.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The receiver is disabled or being enabled. |
| 1 | The receiver is enabled or it will be enabled when SPI is enabled. |

**Bits 15:14 – AMODE[1:0]** Address Mode
These bits set the slave addressing mode when the frame format (CTRLA.FORM) with address is used. They are unused in master mode.

| AMODE[1:0] | Name | Description |
|---|---|---|
| 0x0 | MASK | ADDRMASK is used as a mask to the ADDR register |
| 0x1 | 2_ADDRS | The slave responds to the two unique addresses in ADDR and ADDRMASK |
| 0x2 | RANGE | The slave responds to the range of addresses between and including ADDR and ADDRMASK. ADDR is the upper limit |
| 0x3 | - | Reserved |

**Bit 13 – MSSEN** Master Slave Select Enable
This bit enables hardware slave select ($\overline{SS}$) control.

| Value | Description |
|---|---|
| 0 | Hardware $\overline{SS}$ control is disabled. |
| 1 | Hardware $\overline{SS}$ control is enabled. |

**Bit 9 – SSDE** Slave Select Low Detect Enable
This bit enables wake up when the slave select ($\overline{SS}$) pin transitions from high to low.

| Value | Description |
|---|---|
| 0 | $\overline{SS}$ low detector is disabled. |
| 1 | $\overline{SS}$ low detector is enabled. |

**Bit 6 – PLOADEN** Slave Data Preload Enable
Setting this bit will enable preloading of the slave shift register when there is no transfer in progress. If the $\overline{SS}$ line is high when DATA is written, it will be transferred immediately to the shift register.

**Bits 2:0 – CHSIZE[2:0]** Character Size

| CHSIZE[2:0] | Name | Description |
|---|---|---|
| 0x0 | 8BIT | 8 bits |
| 0x1 | 9BIT | 9 bits |
| 0x2-0x7 | - | Reserved |

### 27.8.3 Baud Rate

**Name:** BAUD
**Offset:** 0x0C
**Reset:** 0x00
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – BAUD[7:0]** Baud Register
These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator*.

**Related Links**

25.6.2.3 Clock Generation – Baud-Rate Generator

25.6.2.3.1 Asynchronous Arithmetic Mode BAUD Value Selection

### 27.8.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | | SSL | RXC | TXC | DRE |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 3 – SSL** Slave Select Low Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Slave Select Low Interrupt Enable bit, which disables the Slave Select Low interrupt.

| Value | Description |
|---|---|
| 0 | Slave Select Low interrupt is disabled. |
| 1 | Slave Select Low interrupt is enabled. |

**Bit 2 – RXC** Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC** Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Transmit Complete Interrupt Enable bit, which disable the Transmit Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE** Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 27.8.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x16
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | | SSL | RXC | TXC | DRE |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 3 – SSL** Slave Select Low Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Slave Select Low Interrupt Enable bit, which enables the Slave Select Low interrupt.

| Value | Description |
|---|---|
| 0 | Slave Select Low interrupt is disabled. |
| 1 | Slave Select Low interrupt is enabled. |

**Bit 2 – RXC** Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC** Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE** Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 27.8.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | | SSL | RXC | TXC | DRE |
| Access | R/W | | | | R/W | R | R/W | R |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error
This flag is cleared by writing '1' to it.

This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. The BUFOVF error will set this interrupt flag.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 3 – SSL** Slave Select Low
This flag is cleared by writing '1' to it.

This bit is set when a high to low transition is detected on the _SS pin in slave mode and Slave Select Low Detect (CTRLB.SSDE) is enabled.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 2 – RXC** Receive Complete
This flag is cleared by reading the Data (DATA) register or by disabling the receiver.

This flag is set when there are unread data in the receive buffer. If address matching is enabled, the first data received in a transaction will be an address.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

**Bit 1 – TXC** Transmit Complete
This flag is cleared by writing '1' to it or by writing new data to DATA.

In master mode, this flag is set when the data have been shifted out and there are no new data in DATA.

In slave mode, this flag is set when the _SS pin is pulled high. If address matching is enabled, this flag is only set if the transaction was initiated with an address match.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 0 – DRE** Data Register Empty
This flag is cleared by writing new data to DATA.

This flag is set when DATA is empty and ready for new data to transmit.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

### 27.8.7 Status

**Name:** STATUS
**Offset:** 0x1A
**Reset:** 0x0000
**Property:** –

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BUFOVF | | |
| Access | | | | | | R/W | | |
| Reset | | | | | | 0 | | |

**Bit 2 – BUFOVF** Buffer Overflow
Reading this bit before reading DATA will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when a buffer overflow condition is detected. See also CTRLA.IBON for overflow handling.

When set, the corresponding RxDATA will be zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

| Value | Description |
|---|---|
| 0 | No Buffer Overflow has occurred. |
| 1 | A Buffer Overflow has occurred. |

### 27.8.8 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CTRLB | ENABLE | SWRST |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – CTRLB** CTRLB Synchronization Busy
Writing to the CTRLB when the SERCOM is enabled requires synchronization. Ongoing synchronization is indicated by SYNCBUSY.CTRLB=1 until synchronization is complete. If CTRLB is written while SYNCBUSY.CTRLB=1, an APB error will be generated.

| Value | Description |
|---|---|
| 0 | CTRLB synchronization is not busy. |
| 1 | CTRLB synchronization is busy. |

**Bit 1 – ENABLE** SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. Ongoing synchronization is indicated by SYNCBUSY.ENABLE=1 until synchronization is complete.

| Value | Description |
|---|---|
| 0 | Enable synchronization is not busy. |
| 1 | Enable synchronization is busy. |

**Bit 0 – SWRST** Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. Ongoing synchronization is indicated by SYNCBUSY.SWRST=1 until synchronization is complete.

| Value | Description |
|---|---|
| 0 | SWRST synchronization is not busy. |
| 1 | SWRST synchronization is busy. |

### 27.8.9 Address

**Name:** ADDR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDRMASK[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – ADDRMASK[7:0]** Address Mask
These bits hold the address mask when the transaction format with address is used (CTRLA.FORM, CTRLB.AMODE).

**Bits 7:0 – ADDR[7:0]** Address
These bits hold the address when the transaction format with address is used (CTRLA.FORM, CTRLB.AMODE).

### 27.8.10 Data

**Name:** DATA
**Offset:** 0x28
**Reset:** 0x0000
**Property:** –

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATA[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – DATA[8:0]** Data
Reading these bits will return the contents of the receive data buffer. The register should be read only when the Receive Complete Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set.

Writing these bits will write the transmit data buffer. This register should be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

### 27.8.11 Debug Control

**Name:** DBGCTRL
**Offset:** 0x30
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGSTOP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGSTOP** Debug Stop Mode
This bit controls the functionality when the CPU is halted by an external debugger.

| Value | Description |
|---|---|
| 0 | The baud-rate generator continues normal operation when the CPU is halted by an external debugger. |
| 1 | The baud-rate generator is halted when the CPU is halted by an external debugger. |

### 27.8.12 FIFO Space

**Name:** FIFOSPACE
**Offset:** 0x34
**Reset:** 0x0000

This register allows the user to identify the number of bytes present in each TX and RX FIFO.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | TXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 12:8 – RXSPACE[4:0]** RX FIFO Filled Space
These bits return the number filled locations in the RX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

**Bits 4:0 – TXSPACE[4:0]** TX FIFO Filled Space
These bits return the number of available locations in the TX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

### 27.8.13 FIFO CPU Pointers

**Name:** FIFOPTR
**Offset:** 0x36
**Reset:** 0x0000

This register provides a copy of internal CPU TX and RX FIFO pointers.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPURDPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPUWRPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – CPURDPTR[3:0]** RX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. Reading DATA register, will return RXFIFO[CPURDPTR] location value.

**Bits 3:0 – CPUWRPTR[3:0]** TX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. When writting to DATA register, the DATA will be written to TXFIFO[CPUWRPTR] location.

# 28. SERCOM I²C – Inter-Integrated Circuit

## 28.1 Overview

The inter-integrated circuit ( I²C) interface is one of the available modes in the serial communication interface (SERCOM).

The I²C interface uses the SERCOM transmitter and receiver configured as shown in Figure 28-1. Labels in capital letters are registers accessible by the CPU, while lowercase labels are internal to the SERCOM.

A SERCOM instance can be configured to be either an I²C master or an I²C slave. Both master and slave have an interface containing a shift register, a transmit buffer and a receive buffer. In addition, the I²C master uses the SERCOM baud-rate generator, while the I²C slave uses the SERCOM address match logic.

**Related Links**

25. SERCOM – Serial Communication Interface

## 28.2 Features

SERCOM I²C includes the following features:

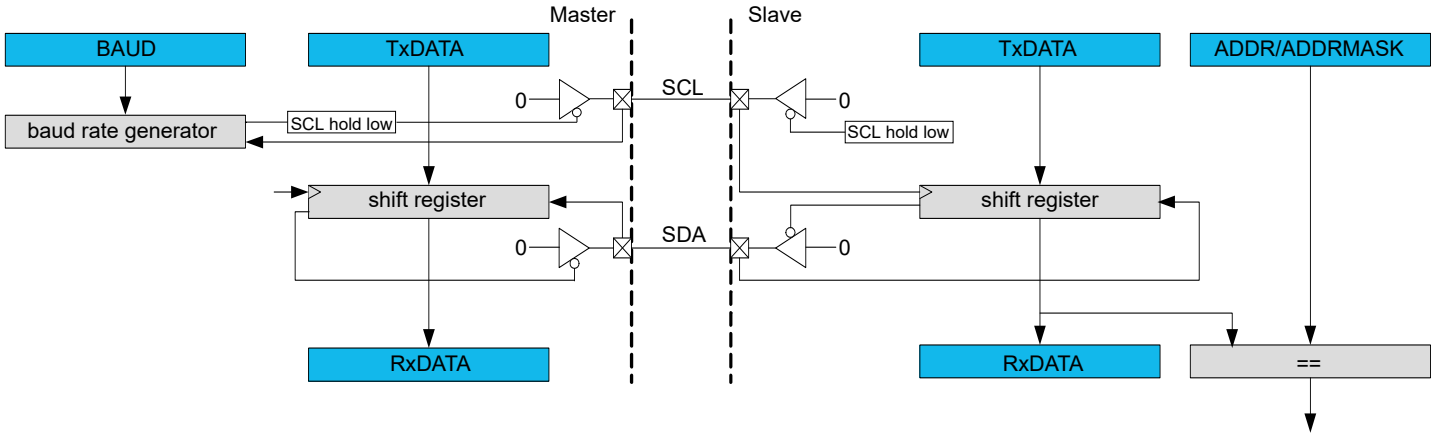
- Master or slave operation
- Can be used with DMA
- Philips I²C compatible
- SMBus™ compatible
- PMBus compatible
- Support of 100kHz and 400kHz, 1MHz and 3.4MHz I²C mode
- Up to 16-bytes internal FIFO
- 4-Wire operation supported
- Physical interface includes:
  - Slew-rate limited outputs
  - Filtered inputs
- Slave operation:
  - Operation in all sleep modes
  - Wake-up on address match
  - 7-bit and 10-bit Address match in hardware for:
  - 
    - Unique address and/or 7-bit general call address
    - Address range
    - Two unique addresses can be used with DMA

**Related Links**

25.2 Features

## 28.3 Block Diagram

**Figure 28-1. I²C Single-Master Single-Slave Interconnection**




## 28.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| PAD[0] | Digital I/O | SDA |
| PAD[1] | Digital I/O | SCL |
| PAD[2] | Digital I/O | SDA_OUT (4-wire operation) |
| PAD[3] | Digital I/O | SCL_OUT (4-wire operation) |

One signal can be mapped on several pins.

Not all the pins are I²C pins.

**Related Links**

7. I/O Multiplexing and Considerations

28.6.3.3 4-Wire Mode

## 28.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 28.5.1 I/O Lines

In order to use the I/O lines of this peripheral, the I/O pins must be configured using the I/O Pin Controller (PORT).

When the SERCOM is used in I²C mode, the SERCOM controls the direction and value of the I/O pins. If the receiver or transmitter is disabled, these pins can be used for other purposes.

**Related Links**

23. PORT - I/O Pin Controller

### 28.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes.

**Related Links**

16. PM – Power Manager

### 28.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the Power Manager. Refer to *Peripheral Clock Masking* for details and default status of this clock.

Two generic clocks are used by SERCOM, GCLK_SERCOMx_CORE and GCLK_SERCOM_SLOW. The core clock (GCLK_SERCOMx_CORE) can clock the $I^2C$ when working as a master. The slow clock (GCLK_SERCOM_SLOW) is required only for certain functions, e.g. SMBus timing. These two clocks must be configured and enabled in the Generic Clock Controller (GCLK) before using the $I^2C$.

These generic clocks are asynchronous to the bus clock (CLK_SERCOMx_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to 28.6.6 Synchronization for further details.

**Related Links**

15. GCLK - Generic Clock Controller
16. PM – Power Manager
16.6.2.6 Peripheral Clock Masking

### 28.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 28.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 28.5.6 Events

Not applicable.

### 28.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

### 28.5.8 Register Access Protection

Registers with write-access can be write-protected optionally by the peripheral access controller (PAC).

PAC Write-Protection is not available for the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)
- Address register (ADDR)

Optional PAC Write-Protection is denoted by the "PAC Write-Protection" property in each individual register description.

Write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 28.5.9 Analog Connections

Not applicable.

## 28.6 Functional Description

### 28.6.1 Principle of Operation

The $I^2C$ interface uses two physical lines for communication:

- Serial Data Line (SDA) for data transfer
- Serial Clock Line (SCL) for the bus clock

A transaction starts with the $I^2C$ master sending the start condition, followed by a 7-bit address and a direction bit (read or write to/from the slave).

The addressed $I^2C$ slave will then acknowledge (ACK) the address, and data packet transactions can begin. Every 9-bit data packet consists of 8 data bits followed by a one-bit reply indicating whether the data was acknowledged or not.

If a data packet is not acknowledged (NACK), whether by the $I^2C$ slave or master, the $I^2C$ master takes action by either terminating the transaction by sending the stop condition, or by sending a repeated start to transfer more data.

The figure below illustrates the possible transaction formats and Transaction Diagram Symbols explains the transaction symbols. These symbols will be used in the following descriptions.

**Figure 28-2. Transaction Diagram Symbols**

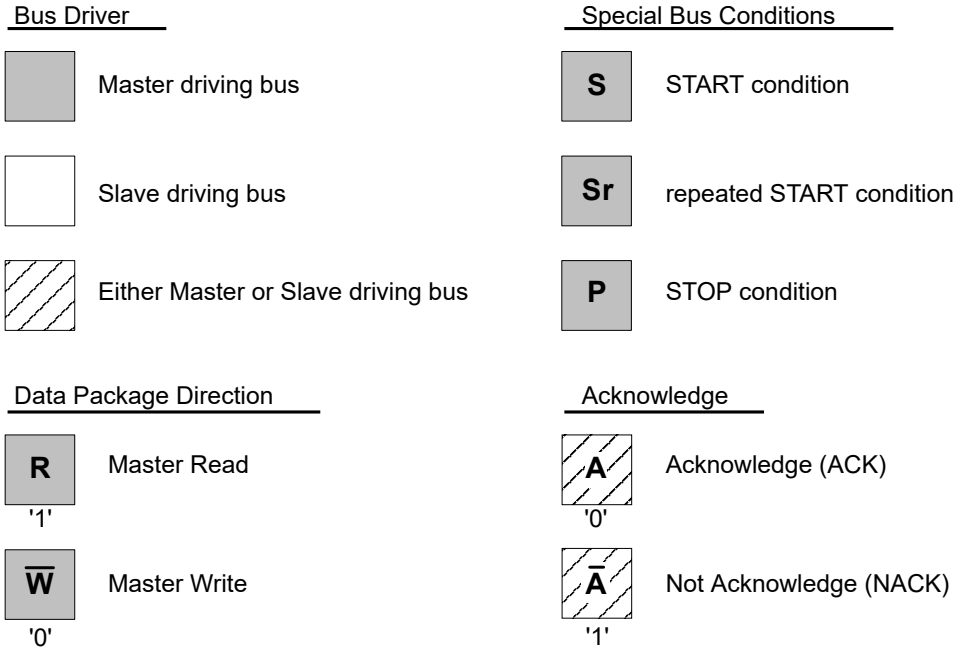
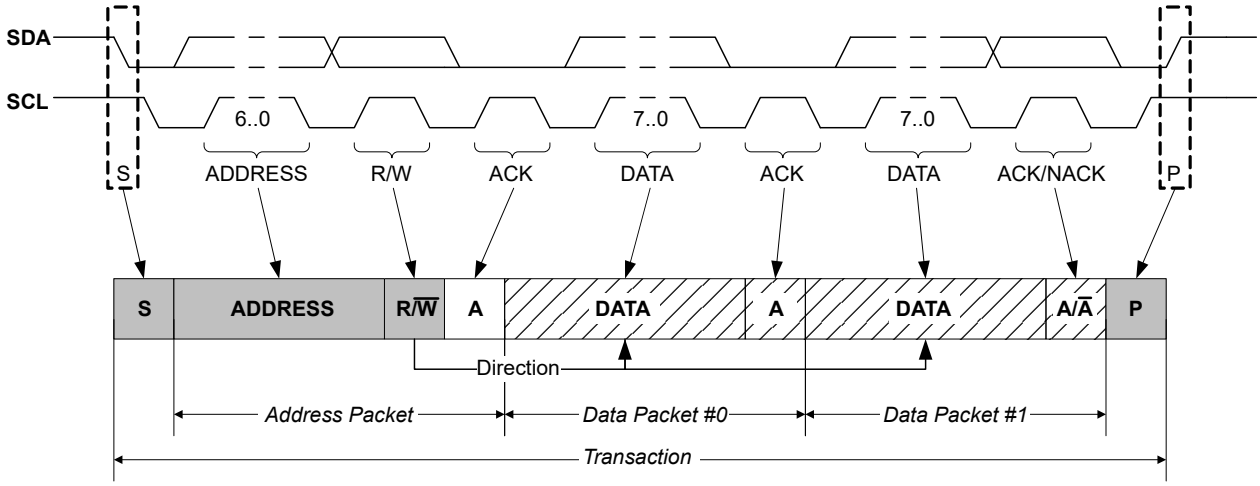



**Figure 28-3. Basic I²C Transaction Diagram**




### 28.6.2 Basic Operation

#### 28.6.2.1 Initialization

The following registers are enable-protected, meaning they can be written only when the I²C interface is disabled (CTRLA.ENABLE is '0'):

- Control A register (CTRLA), except Enable (CTRLA.ENABLE) and Software Reset (CTRLA.SWRST) bits
- Control B register (CTRLB), except Acknowledge Action (CTRLB.ACKACT) and Command (CTRLB.CMD) bits
- Baud register (BAUD)
- Address register (ADDR) in slave operation.

When the I²C is enabled or is being enabled (CTRLA.ENABLE=1), writing to these registers will be discarded. If the I²C is being disabled, writing to these registers will be completed after the disabling.

Enable-protection is denoted by the "Enable-Protection" property in the register description.

Before the I²C is enabled it must be configured as outlined by the following steps:

1. Select I²C Master or Slave mode by writing 0x4 (Slave mode) or 0x5 (Master mode) to the Operating Mode bits in the CTRLA register (CTRLA.MODE).
2. If desired, select the SDA Hold Time value in the CTRLA register (CTRLA.SDAHOLD).
3. If desired, enable smart operation by setting the Smart Mode Enable bit in the CTRLB register (CTRLB.SMEN).
4. If desired, enable SCL low time-out by setting the SCL Low Time-Out bit in the Control A register (CTRLA.LOWTOUT).
5. In Master mode:
   5.1. Select the inactive bus time-out in the Inactive Time-Out bit group in the CTRLA register (CTRLA.INACTOUT).
   5.2. Write the Baud Rate register (BAUD) to generate the desired baud rate.

   In Slave mode:
   5.1. Configure the address match configuration by writing the Address Mode value in the CTRLB register (CTRLB.AMODE).
   5.2. Set the Address and Address Mask value in the Address register (ADDR.ADDR and ADDR.ADDRMASK) according to the address configuration.

#### 28.6.2.2 Enabling, Disabling, and Resetting

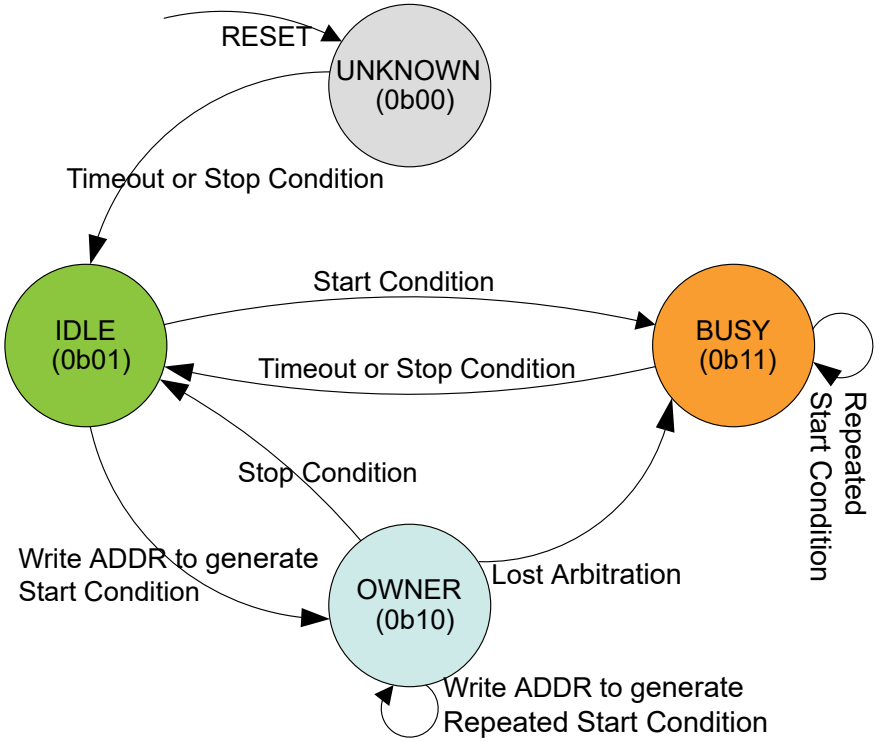
This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

#### 28.6.2.3 I²C Bus State Logic

The bus state logic includes several logic blocks that continuously monitor the activity on the I²C bus lines in all sleep modes with running GCLK_SERCOM_x clocks. The start and stop detectors and the bit counter are all essential in the process of determining the current bus state. The bus state is determined according to Bus State Diagram. Software can get the current bus state by reading the Master Bus State bits in the Status register (STATUS.BUSSTATE). The value of STATUS.BUSSTATE in the figure is shown in binary.

**Figure 28-4. Bus State Diagram**




The bus state machine is active when the I²C master is enabled.

After the I²C master has been enabled, the bus state is UNKNOWN (0b00). From the UNKNOWN state, the bus will transition to IDLE (0b01) by either:

- Forcing by writing 0b01 to STATUS.BUSSTATE
- A stop condition is detected on the bus
- If the inactive bus time-out is configured for SMBus compatibility (CTRLA.INACTOUT) and a time-out occurs.

**Note:** Once a known bus state is established, the bus state logic will not re-enter the UNKNOWN state.

When the bus is IDLE it is ready for a new transaction. If a start condition is issued on the bus by another I²C master in a multi-master setup, the bus becomes BUSY (0b11). The bus will re-enter IDLE either when a stop condition is detected, or when a time-out occurs (inactive bus time-out needs to be configured).

If a start condition is generated internally by writing the Address bit group in the Address register (ADDR.ADDR) while IDLE, the OWNER state (0b10) is entered. If the complete transaction was performed without interference, i.e., arbitration was not lost, the I²C master can issue a stop condition, which will change the bus state back to IDLE.

However, if a packet collision is detected while in OWNER state, the arbitration is assumed lost and the bus state becomes BUSY until a stop condition is detected. A repeated start condition will change the bus state only if arbitration is lost while issuing a repeated start.

**Note:** Violating the protocol may cause the I²C to hang. If this happens it is possible to recover from this state by a software reset (CTRLA.SWRST='1').

**Related Links**

28.10.1 CTRLA

### 28.6.2.4 I²C Master Operation

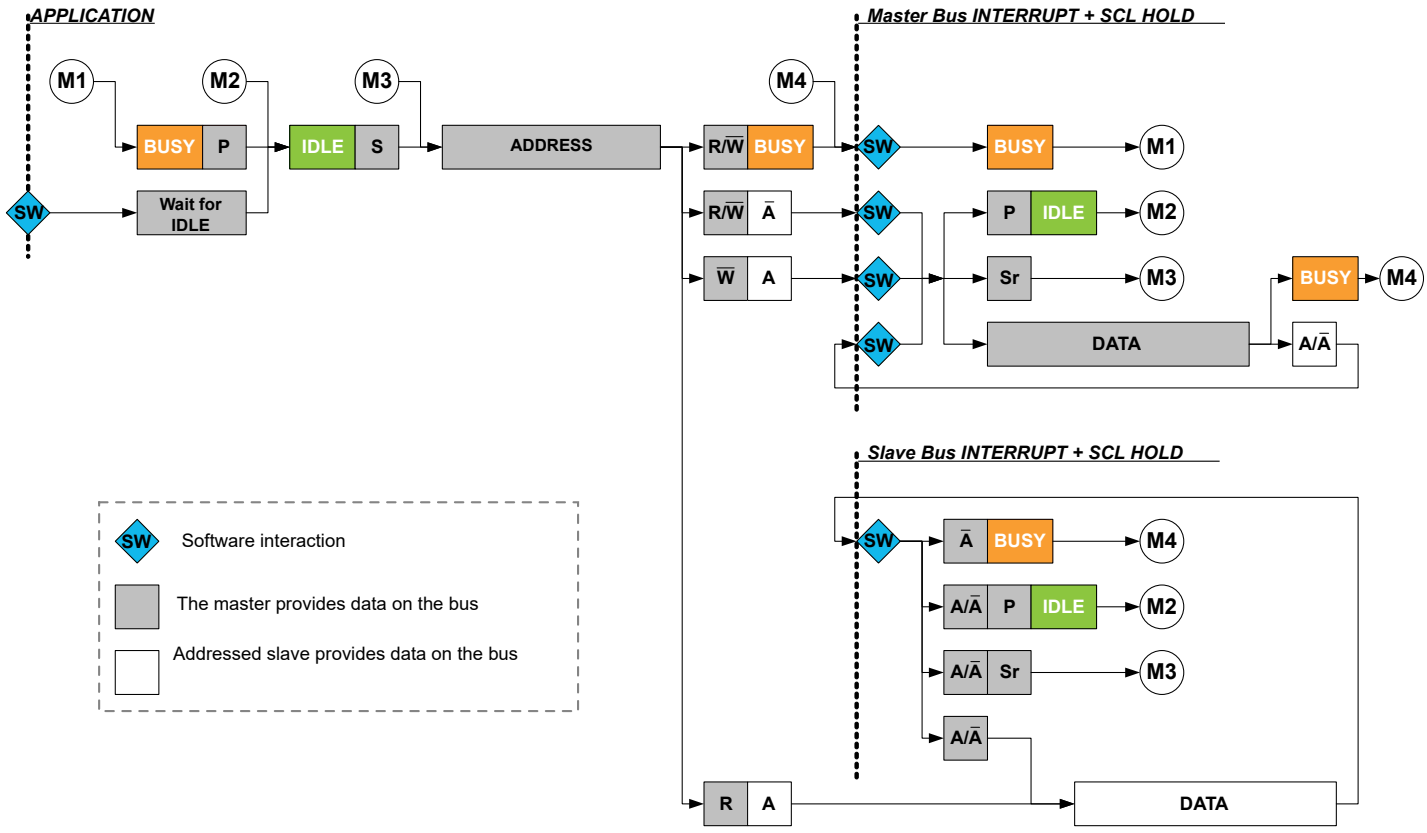
The I²C master is byte-oriented and interrupt based. The number of interrupts generated is kept at a minimum by automatic handling of most incidents. The software driver complexity and code size are reduced by auto-triggering of operations, and a special smart mode, which can be enabled by the Smart Mode Enable bit in the Control A register (CTRLA.SMEN).

The I²C master has two interrupt strategies.

When SCL Stretch Mode (CTRLA.SCLSM) is '0', SCL is stretched before or after the acknowledge bit . In this mode the I²C master operates according to Master Behavioral Diagram (SCLSM=0). The circles labelled "M*n*" (M1, M2..) indicate the nodes the bus logic can jump to, based on software or hardware interaction.

This diagram is used as reference for the description of the I²C master operation throughout the document.

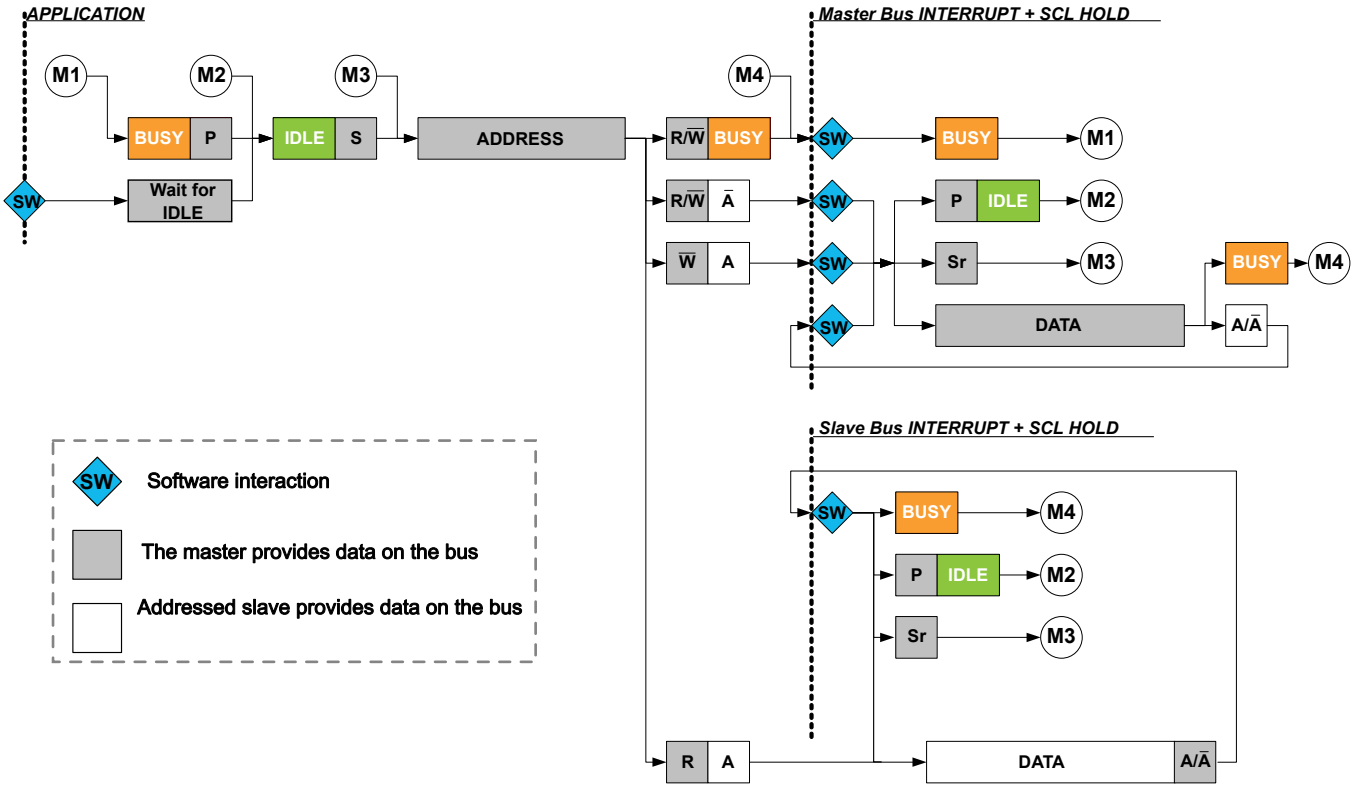
**Figure 28-5. I²C Master Behavioral Diagram (SCLSM=0)**




In the second strategy (CTRLA.SCLSM=1), interrupts only occur after the ACK bit, as in Master Behavioral Diagram (SCLSM=1). This strategy can be used when it is not necessary to check DATA before acknowledging.

**Note:** I²C High-speed (*Hs*) mode requires CTRLA.SCLSM=1.

**Figure 28-6. I²C Master Behavioral Diagram (SCLSM=1)**




#### 28.6.2.4.1 Master Clock Generation

The SERCOM peripheral supports several I²C bidirectional modes:

- Standard mode (*Sm*) up to 100kHz
- Fast mode (*Fm*) up to 400kHz
- Fast mode Plus (*Fm+*) up to 1MHz
- High-speed mode (*Hs*) up to 3.4MHz

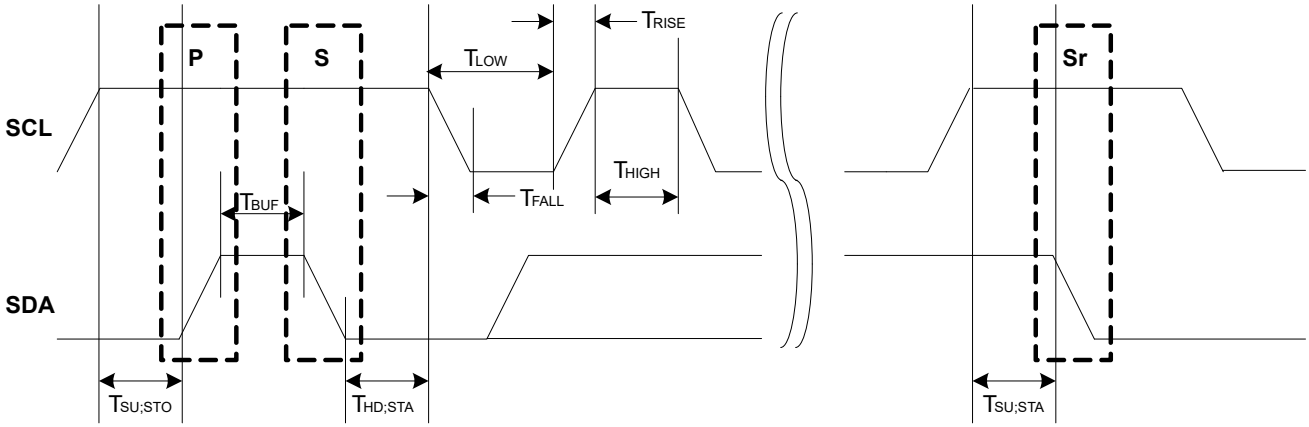
The Master clock configuration for *Sm, Fm,* and *Fm+* are described in Clock Generation (Standard-Mode, Fast-Mode, and Fast-Mode Plus). For *Hs*, refer to Master Clock Generation (High-Speed Mode).

***Clock Generation (Standard-Mode, Fast-Mode, and Fast-Mode Plus)***

In I²C *Sm, Fm*, and *Fm+* mode, the Master clock (SCL) frequency is determined as described in this section:

The low ($T_{LOW}$) and high ($T_{HIGH}$) times are determined by the Baud Rate register (BAUD), while the rise ($T_{RISE}$) and fall ($T_{FALL}$) times are determined by the bus topology. Because of the wired-AND logic of the bus, $T_{FALL}$ will be considered as part of $T_{LOW}$. Likewise, $T_{RISE}$ will be in a state between $T_{LOW}$ and $T_{HIGH}$ until a high state has been detected.

**Figure 28-7. SCL Timing**




The following parameters are timed using the SCL low time period $T_{LOW}$. This comes from the Master Baud Rate Low bit group in the Baud Rate register (BAUD.BAUDLOW). When BAUD.BAUDLOW=0, or the Master Baud Rate bit group in the Baud Rate register (BAUD.BAUD) determines it.

- $T_{LOW}$ – Low period of SCL clock
- $T_{SU;STO}$ – Set-up time for stop condition
- $T_{BUF}$ – Bus free time between stop and start conditions
- $T_{HD;STA}$ – Hold time (repeated) start condition
- $T_{SU;STA}$ – Set-up time for repeated start condition
- $T_{HIGH}$ is timed using the SCL high time count from BAUD.BAUD
- $T_{RISE}$ is determined by the bus impedance; for internal pull-ups. Refer to *Electrical Characteristics*.
- $T_{FALL}$ is determined by the open-drain current limit and bus impedance; can typically be regarded as zero. Refer to *Electrical Characteristics* for details.

The SCL frequency is given by:

$$f_{\mathrm{SCL}} = \frac{1}{T_{\mathrm{LOW}} + T_{\mathrm{HIGH}} + T_{\mathrm{RISE}}}$$

When BAUD.BAUDLOW is zero, the BAUD.BAUD value is used to time both SCL high and SCL low. In this case the following formula will give the SCL frequency:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + 2BAUD + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

When BAUD.BAUDLOW is non-zero, the following formula determines the SCL frequency:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + BAUD + BAUDLOW + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

The following formulas can determine the SCL $T_{LOW}$ and $T_{HIGH}$ times:

$$T_{\mathrm{LOW}} = \frac{BAUDLOW + 5}{f_{\mathrm{GCLK}}}$$

$$T_{\mathrm{HIGH}} = \frac{BAUD + 5}{f_{\mathrm{GCLK}}}$$

**Note:** The I²C standard *Fm+* (Fast-mode plus) requires a nominal high to low SCL ratio of 1:2, and BAUD should be set accordingly. At a minimum, BAUD.BAUD and/or BAUD.BAUDLOW must be non-zero.

**Startup Timing** The minimum time between SDA transition and SCL rising edge is 6 APB cycles when the DATA register is written in smart mode. If a greater startup time is required due to long rise times, the time between DATA write and IF clear must be controlled by software.
**Note:** When timing is controlled by user, the Smart Mode cannot be enabled.

**Related Links**

37. Electrical Characteristics

***Master Clock Generation (High-Speed Mode)***

For I²C *Hs* transfers, there is no SCL synchronization. Instead, the SCL frequency is determined by the GCLK_SERCOMx_CORE frequency ($f_{GCLK}$) and the High-Speed Baud setting in the Baud register (BAUD.HSBAUD). When BAUD.HSBAUDLOW=0, the HSBAUD value will determine both SCL high and SCL low. In this case the following formula determines the SCL frequency.

$$f_{SCL} = \frac{f_{GCLK}}{2 + 2 \cdot HS\ BAUD}$$

When HSBAUDLOW is non-zero, the following formula determines the SCL frequency.

$$f_{SCL} = \frac{f_{GCLK}}{2 + HS\ BAUD + HSBAUDLOW}$$

**Note:** The I²C standard *Hs* (High-speed) requires a nominal high to low SCL ratio of 1:2, and HSBAUD should be set accordingly. At a minimum, BAUD.HSBAUD and/or BAUD.HSBAUDLOW must be non-zero.

#### 28.6.2.4.2 Transmitting Address Packets

The I²C master starts a bus transaction by writing the I²C slave address to ADDR.ADDR and the direction bit, as described in 28.6.1 Principle of Operation. If the bus is busy, the I²C master will wait until the bus becomes idle before continuing the operation. When the bus is idle, the I²C master will issue a start condition on the bus. The I²C master will then transmit an address packet using the address written to ADDR.ADDR. After the address packet has been transmitted by the I²C master, one of four cases will arise according to arbitration and transfer direction.

**Case 1: Arbitration lost or bus error during address packet transmission**

If arbitration was lost during transmission of the address packet, the Master on Bus bit in the Interrupt Flag Status and Clear register (INTFLAG.MB) and the Arbitration Lost bit in the Status register (STATUS.ARBLOST) are both set. Serial data output to SDA is disabled, and the SCL is released, which disables clock stretching. In effect the I²C master is no longer allowed to execute any operation on the bus until the bus is idle again. A bus error will behave similarly to the arbitration lost condition. In this case, the MB interrupt flag and Master Bus Error bit in the Status register (STATUS.BUSERR) are both set in addition to STATUS.ARBLOST.

The Master Received Not Acknowledge bit in the Status register (STATUS.RXNACK) will always contain the last successfully received acknowledge or not acknowledge indication.

In this case, software will typically inform the application code of the condition and then clear the interrupt flag before exiting the interrupt routine. No other flags have to be cleared at this moment, because all flags will be cleared automatically the next time the ADDR.ADDR register is written.

**Case 2: Address packet transmit complete – No ACK received**

If there is no I²C slave device responding to the address packet, then the INTFLAG.MB interrupt flag and STATUS.RXNACK will be set. The clock hold is active at this point, preventing further activity on the bus.

The missing ACK response can indicate that the I²C slave is busy with other tasks or sleeping. Therefore, it is not able to respond. In this event, the next step can be either issuing a stop condition (recommended) or resending the address packet by a repeated start condition. When using SMBus logic, the slave must ACK the address. If there is no response, it means that the slave is not available on the bus.

**Case 3: Address packet transmit complete – Write packet, Master on Bus set**

If the I²C master receives an acknowledge response from the I²C slave, INTFLAG.MB will be set and STATUS.RXNACK will be cleared. The clock hold is active at this point, preventing further activity on the bus.

In this case, the software implementation becomes highly protocol dependent. Three possible actions can enable the I²C operation to continue:

- Initiate a data transmit operation by writing the data byte to be transmitted into DATA.DATA.
- Transmit a new address packet by writing ADDR.ADDR. A repeated start condition will automatically be inserted before the address packet.
- Issue a stop condition, consequently terminating the transaction.

**Case 4: Address packet transmit complete – Read packet, Slave on Bus set**

If the I²C master receives an ACK from the I²C slave, the I²C master proceeds to receive the next byte of data from the I²C slave. When the first data byte is received, the Slave on Bus bit in the Interrupt Flag register (INTFLAG.SB) will be set and STATUS.RXNACK will be cleared. The clock hold is active at this point, preventing further activity on the bus.

In this case, the software implementation becomes highly protocol dependent. Three possible actions can enable the I²C operation to continue:

- Let the I²C master continue to read data by acknowledging the data received. ACK can be sent by software, or automatically in smart mode.
- Transmit a new address packet.
- Terminate the transaction by issuing a stop condition.

**Note:** An ACK or NACK will be automatically transmitted if smart mode is enabled. The Acknowledge Action bit in the Control B register (CTRLB.ACKACT) determines whether ACK or NACK should be sent.

#### 28.6.2.4.3 Transmitting Data Packets

When an address packet with direction Master Write (see Figure 28-3) was transmitted successfully , INTFLAG.MB will be set. The I²C master will start transmitting data via the I²C bus by writing to DATA.DATA, and monitor continuously for packet collisions. I

If a collision is detected, the I²C master will lose arbitration and STATUS.ARBLOST will be set. If the transmit was successful, the I²C master will receive an ACK bit from the I²C slave, and STATUS.RXNACK will be cleared. INTFLAG.MB will be set in both cases, regardless of arbitration outcome.

It is recommended to read STATUS.ARBLOST and handle the arbitration lost condition in the beginning of the I²C Master on Bus interrupt. This can be done as there is no difference between handling address and data packet arbitration.

STATUS.RXNACK must be checked for each data packet transmitted before the next data packet transmission can commence. The I²C master is not allowed to continue transmitting data packets if a NACK is received from the I²C slave.

#### 28.6.2.4.4 Receiving Data Packets (SCLSM=0)

When INTFLAG.SB is set, the I²C master will already have received one data packet. The I²C master must respond by sending either an ACK or NACK. Sending a NACK may be unsuccessful when arbitration is lost during the transmission. In this case, a lost arbitration will prevent setting INTFLAG.SB. Instead, INTFLAG.MB will indicate a change in arbitration. Handling of lost arbitration is the same as for data bit transmission.

#### 28.6.2.4.5 Receiving Data Packets (SCLSM=1)

When INTFLAG.SB is set, the I²C master will already have received one data packet and transmitted an ACK or NACK, depending on CTRLB.ACKACT. At this point, CTRLB.ACKACT must be set to the correct value for the next ACK bit, and the transaction can continue by reading DATA and issuing a command if not in the smart mode.

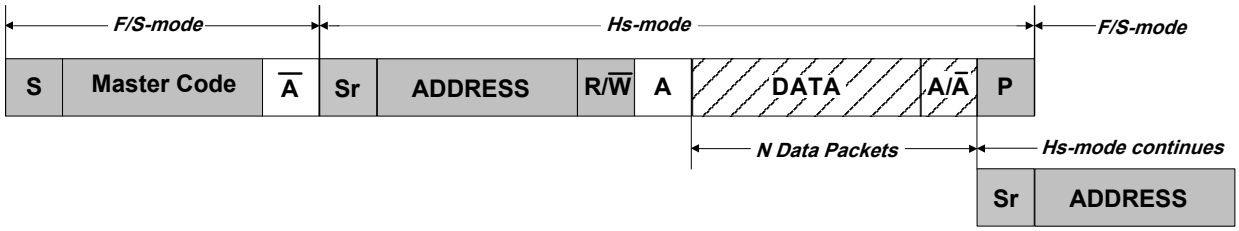
#### 28.6.2.4.6 High-Speed Mode

High-speed transfers are a multi-step process, see High Speed Transfer.

First, a master code (0b00001nnn, where 'nnn' is a unique master code) is transmitted in Full-speed mode, followed by a NACK since no slaveshould acknowledge. Arbitration is performed only during the Full-speed Master Code phase. The master code is transmitted by writing the master code to the address register (ADDR.ADDR) and writing the high-speed bit (ADDR.HS) to '0'.

After the master code and NACK have been transmitted, the master write interrupt will be asserted. In the meanwhile, the slave address can be written to the ADDR.ADDR register together with ADDR.HS=1. Now in High-speed mode, the master will generate a repeated start, followed by the slave address with RW-direction. The bus will remain in High-speed mode until a stop is generated. If a repeated start is desired, the ADDR.HS bit must again be written to '1', along with the new address ADDR.ADDR to be transmitted.

**Figure 28-8. High Speed Transfer**



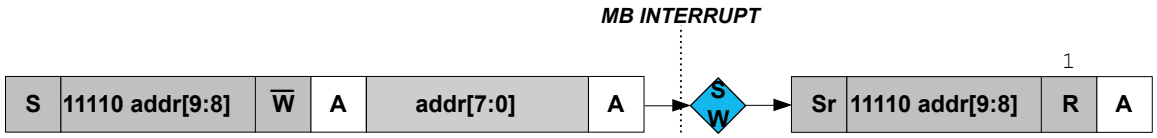

Transmitting in High-speed mode requires the I²C master to be configured in High-speed mode (CTRLA.SPEED=0x2) and the SCL clock stretch mode (CTRLA.SCLSM) bit set to '1'.

#### 28.6.2.4.7 10-Bit Addressing

When 10-bit addressing is enabled by the Ten Bit Addressing Enable bit in the Address register (ADDR.TENBITEN=1) and the Address bit field ADDR.ADDR is written, the two address bytes will be transmitted, see 10-bit Address Transmission for a Read Transaction. The addressed slave acknowledges the two address bytes, and the transaction continues. Regardless of whether the transaction is a read or write, the master must start by sending the 10-bit address with the direction bit (ADDR.ADDR[0]) being zero.

If the master receives a NACK after the first byte, the write interrupt flag will be raised and the STATUS.RXNACK bit will be set. If the first byte is acknowledged by one or more slaves, then the master will proceed to transmit the second address byte and the master will first see the write interrupt flag after the second byte is transmitted. If the transaction direction is read-from-slave, the 10-bit address transmission must be followed by a repeated start and the first 7 bits of the address with the read/write bit equal to '1'.

**Figure 28-9. 10-bit Address Transmission for a Read Transaction**




This implies the following procedure for a 10-bit read operation:

1. Write the 10-bit address to ADDR.ADDR[10:1]. ADDR.TENBITEN must be '1', the direction bit (ADDR.ADDR[0]) must be '0' (can be written simultaneously with ADDR).
2. Once the Master on Bus interrupt is asserted, Write ADDR[7:0] register to '11110 `address[9:8]` 1'. ADDR.TENBITEN must be cleared (can be written simultaneously with ADDR).
3. Proceed to transmit data.

### 28.6.2.5 I²C Slave Operation

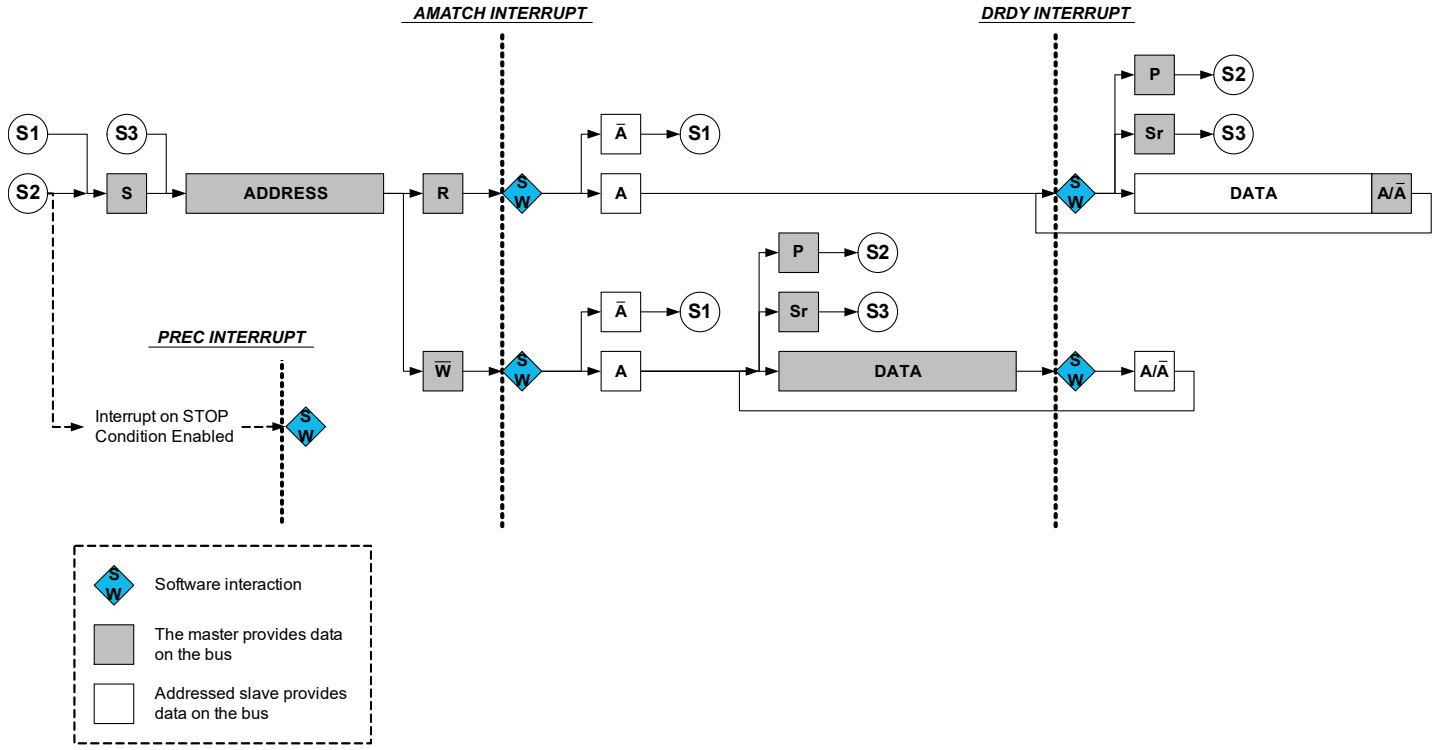
The I²C slave is byte-oriented and interrupt-based. The number of interrupts generated is kept at a minimum by automatic handling of most events. The software driver complexity and code size are reduced by auto-triggering of operations, and a special smart mode, which can be enabled by the Smart Mode Enable bit in the Control A register (CTRLA.SMEN).

The I²C slave has two interrupt strategies.

When SCL Stretch Mode bit (CTRLA.SCLSM) is '0', SCL is stretched before or after the acknowledge bit. In this mode, the I²C slave operates according to I²C Slave Behavioral Diagram (SCLSM=0). The circles labelled "S*n*" (S1, S2..) indicate the nodes the bus logic can jump to, based on software or hardware interaction.

This diagram is used as reference for the description of the I²C slave operation throughout the document.

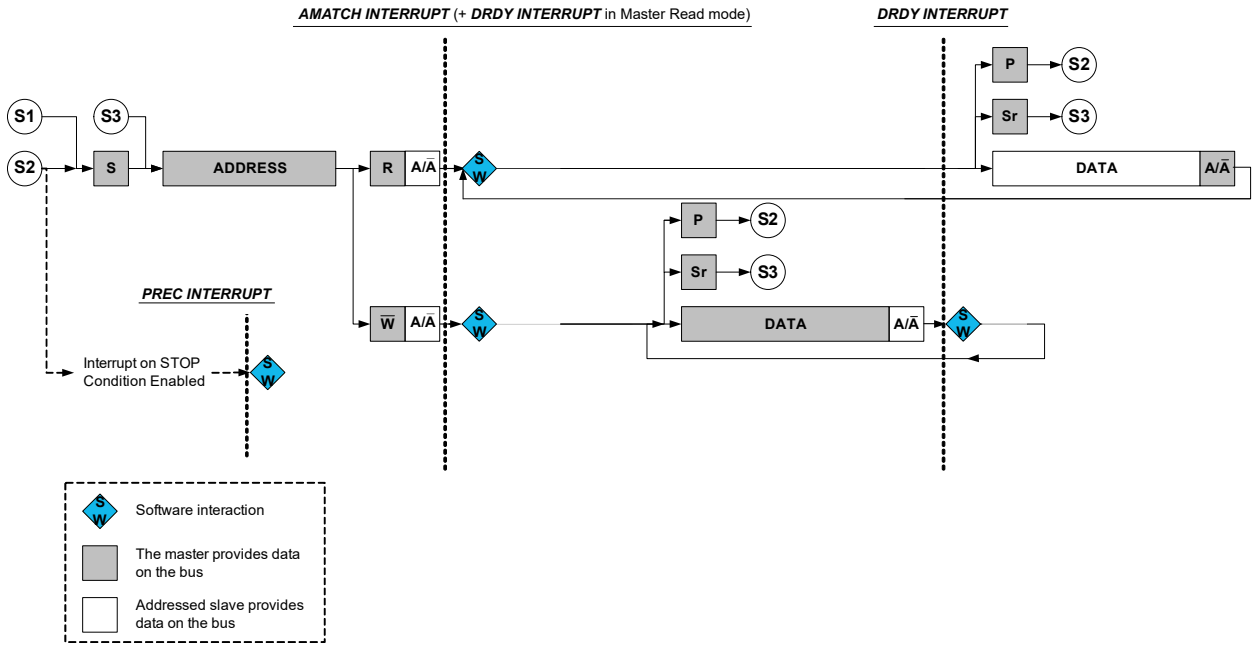
**Figure 28-10. I²C Slave Behavioral Diagram (SCLSM=0)**



In the second strategy (CTRLA.SCLSM=1), interrupts only occur after the ACK bit is sent as shown in Slave Behavioral Diagram (SCLSM=1). This strategy can be used when it is not necessary to check DATA before acknowledging. For master reads, an address and data interrupt will be issued simultaneously after the address acknowledge. However, for master writes, the first data interrupt will be seen after the first data byte has been received by the slave and the acknowledge bit has been sent to the master.

**Note:** For I²C High-speed mode (*Hs*), SCLSM=1 is required.

**Figure 28-11. I²C Slave Behavioral Diagram (SCLSM=1)**




#### 28.6.2.5.1 Receiving Address Packets (SCLSM=0)

When CTRLA.SCLSM=0, the I2C slave stretches the SCL line according to Figure 28-10. When the I²C slave is properly configured, it will wait for a start condition.

When a start condition is detected, the successive address packet will be received and checked by the address match logic. If the received address is not a match, the packet will be rejected, and the I²C slave will wait for a new start condition. If the received address is a match, the Address Match bit in the Interrupt Flag register (INTFLAG.AMATCH) will be set.

SCL will be stretched until the I²C slave clears INTFLAG.AMATCH. As the I²C slave holds the clock by forcing SCL low, the software has unlimited time to respond.

The direction of a transaction is determined by reading the Read / Write Direction bit in the Status register (STATUS.DIR). This bit will be updated only when a valid address packet is received.

If the Transmit Collision bit in the Status register (STATUS.COLL) is set, this indicates that the last packet addressed to the I²C slave had a packet collision. A collision causes the SDA and SCL lines to be released without any notification to software. Therefore, the next AMATCH interrupt is the first indication of the previous packet's collision. Collisions are intended to follow the SMBus Address Resolution Protocol (ARP).

After the address packet has been received from the I²C master, one of two cases will arise based on transfer direction.

**Case 1: Address packet accepted – Read flag set**

The STATUS.DIR bit is '1', indicating an I²C master read operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, I²C slave hardware will set the Data Ready bit in the Interrupt Flag register (INTFLAG.DRDY), indicating data are needed for transmit. If a NACK is sent, the I²C slave will wait for a new start condition and address match.

Typically, software will immediately acknowledge the address packet by sending an ACK/NACK bit. The I²C slave Command bit field in the Control B register (CTRLB.CMD) can be written to '0x3' for both read and write operations as the command execution is dependent on the STATUS.DIR bit. Writing '1' to INTFLAG.AMATCH will also cause an ACK/NACK to be sent corresponding to the CTRLB.ACKACT bit.

**Case 2: Address packet accepted – Write flag set**

The STATUS.DIR bit is cleared, indicating an I²C master write operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, the I²C slave will wait for data to be received. Data, repeated start or stop can be received.

If a NACK is sent, the I²C slave will wait for a new start condition and address match. Typically, software will immediately acknowledge the address packet by sending an ACK/NACK. The I²C slave command CTRLB.CMD = 3 can be used for both read and write operation as the command execution is dependent on STATUS.DIR.

Writing '1' to INTFLAG.AMATCH will also cause an ACK/NACK to be sent corresponding to the CTRLB.ACKACT bit.

#### 28.6.2.5.2 Receiving Address Packets (SCLSM=1)

When SCLSM=1, the I²C slave will stretch the SCL line only after an ACK, see Slave Behavioral Diagram (SCLSM=1). When the I²C slave is properly configured, it will wait for a start condition to be detected.

When a start condition is detected, the successive address packet will be received and checked by the address match logic.

If the received address is not a match, the packet will be rejected and the I²C slave will wait for a new start condition.

If the address matches, the acknowledge action as configured by the Acknowledge Action bit Control B register (CTRLB.ACKACT) will be sent and the Address Match bit in the Interrupt Flag register (INTFLAG.AMATCH) is set. SCL will be stretched until the I²C slave clears INTFLAG.AMATCH. As the I²C slave holds the clock by forcing SCL low, the software is given unlimited time to respond to the address.

The direction of a transaction is determined by reading the Read/Write Direction bit in the Status register (STATUS.DIR). This bit will be updated only when a valid address packet is received.

If the Transmit Collision bit in the Status register (STATUS.COLL) is set, the last packet addressed to the I²C slave had a packet collision. A collision causes the SDA and SCL lines to be released without any notification to software. The next AMATCH interrupt is, therefore, the first indication of the previous packet's collision. Collisions are intended to follow the SMBus Address Resolution Protocol (*ARP*).

After the address packet has been received from the I²C master, INTFLAG.AMATCH be set to '1' to clear it.

#### 28.6.2.5.3 Receiving and Transmitting Data Packets

After the I²C slave has received an address packet, it will respond according to the direction either by waiting for the data packet to be received or by starting to send a data packet by writing to DATA.DATA.

When a data packet is received or sent, INTFLAG.DRDY will be set. After receiving data, the I²C slave will send an acknowledge according to CTRLB.ACKACT.

**Case 1: Data received**

INTFLAG.DRDY is set, and SCL is held low, pending for SW interaction.

**Case 2: Data sent**

When a byte transmission is successfully completed, the INTFLAG.DRDY interrupt flag is set. If NACK is received, indicated by STATUS.RXNACK=1, the I²C slave must expect a stop or a repeated start to be received. The I²C slave must release the data line to allow the I²C master to generate a stop or repeated start. Upon detecting a stop condition, the Stop Received bit in the Interrupt Flag register (INTFLAG.PREC) will be set and the I²C slave will return to IDLE state.

#### 28.6.2.5.4 High-Speed Mode

When the I²C slave is configured in High-speed mode (*Hs*, CTRLA.SPEED=0x2) and CTRLA.SCLSM=1, switching between Full-speed and High-speed modes is automatic. When the slave recognizes a START followed by a master code transmission and a NACK, it automatically switches to High-speed mode and sets the High-speed status bit (STATUS.HS). The slave will then remain in High-speed mode until a STOP is received.

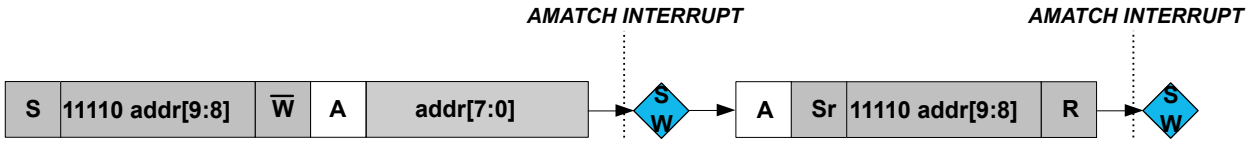
#### 28.6.2.5.5 10-Bit Addressing

When 10-bit addressing is enabled (ADDR.TENBITEN=1), the two address bytes following a START will be checked against the 10-bit slave address recognition. The first byte of the address will always be acknowledged, and the second byte will raise the address interrupt flag, see 10-bit Addressing.

If the transaction is a write, then the 10-bit address will be followed by *N* data bytes.

If the operation is a read, the 10-bit address will be followed by a repeated START and reception of '11110 `ADDR[9:8]` 1', and the second address interrupt will be received with the DIR bit set. The slave matches on the second address as it it was addressed by the previous 10-bit address.

**Figure 28-12. 10-bit Addressing**



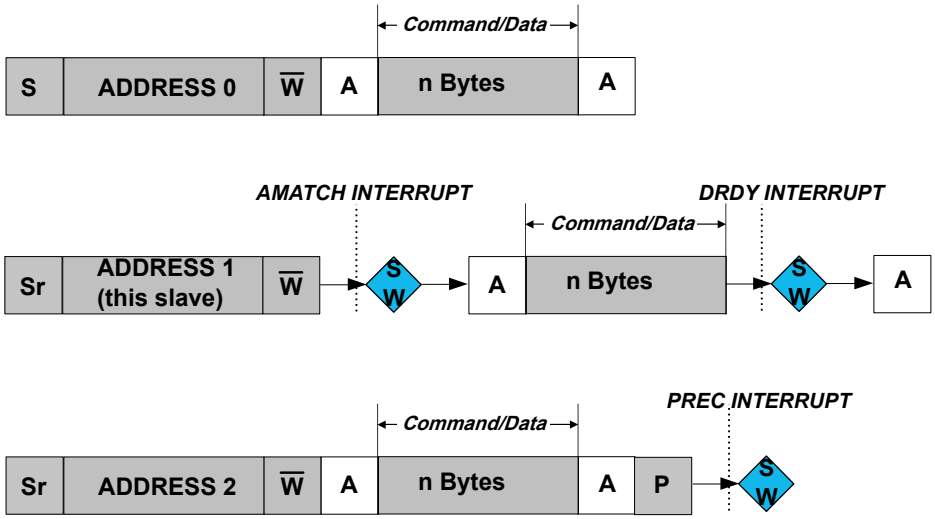

#### 28.6.2.5.6 PMBus Group Command

When the PMBus Group Command bit in the CTRLB register is set (CTRLB.GCMD=1) and 7-bit addressing is used, INTFLAG.PREC will be set if the slave has been addressed since the last STOP condition. When CTRLB.GCMD=0, a STOP condition without address match will not be set INTFLAG.PREC.

The group command protocol is used to send commands to more than one device. The commands are sent in one continuous transmission with a single STOP condition at the end. When the STOP condition is detected by the slaves addressed during the group command, they all begin executing the command they received.

PMBus Group Command Example shows an example where this slave, bearing ADDRESS 1, is addressed after a repeated START condition. There can be multiple slaves addressed before and after this slave. Eventually, at the end of the group command, a single STOP is generated by the master. At this point a STOP interrupt is asserted.

**Figure 28-13. PMBus Group Command Example**




### 28.6.3 Additional Features

#### 28.6.3.1 SMBus

The I$^2$C includes three hardware SCL low time-outs which allow a time-out to occur for SMBus SCL low time-out, master extend time-out, and slave extend time-out. This allows for SMBus functionality These time-outs are driven by the GCLK_SERCOM_SLOW clock. The GCLK_SERCOM_SLOW clock is used to accurately time the time-out and must be configured to use a 32KHz oscillator. The I$^2$C interface also allows for a SMBus compatible SDA hold time.

- $T_{TIMEOUT}$: SCL low time of 25..35ms – Measured for a single SCL low period. It is enabled by CTRLA.LOWTOUTEN.
- $T_{LOW:SEXT}$: Cumulative clock low extend time of 25 ms – Measured as the cumulative SCL low extend time by a slave device in a single message from the initial START to the STOP. It is enabled by CTRLA.SEXTTOEN.
- $T_{LOW:MEXT}$: Cumulative clock low extend time of 10 ms – Measured as the cumulative SCL low extend time by the master device within a single byte from START-to-ACK, ACK-to-ACK, or ACK-to-STOP. It is enabled by CTRLA.MEXTTOEN.

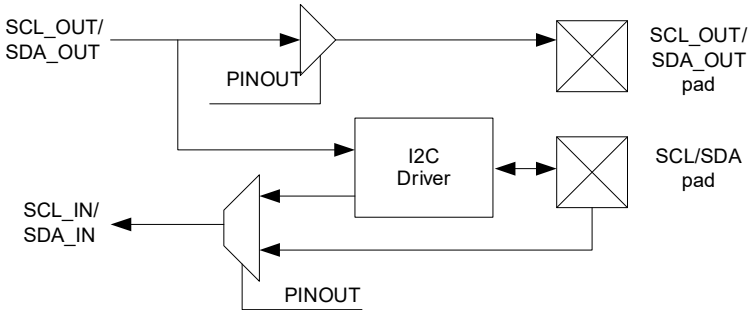
#### 28.6.3.2 Smart Mode

The I$^2$C interface has a smart mode that simplifies application code and minimizes the user interaction needed to adhere to the I$^2$C protocol. The smart mode accomplishes this by automatically issuing an ACK or NACK (based on the content of CTRLB.ACKACT) as soon as DATA.DATA is read.

#### 28.6.3.3 4-Wire Mode

Writing a '1' to the Pin Usage bit in the Control A register (CTRLA.PINOUT) will enable 4-wire mode operation. In this mode, the internal I$^2$C tri-state drivers are bypassed, and an external I$^2$C compliant tri-state driver is needed when connecting to an I$^2$C bus.

**Figure 28-14. I²C Pad Interface**




#### 28.6.3.4 Quick Command

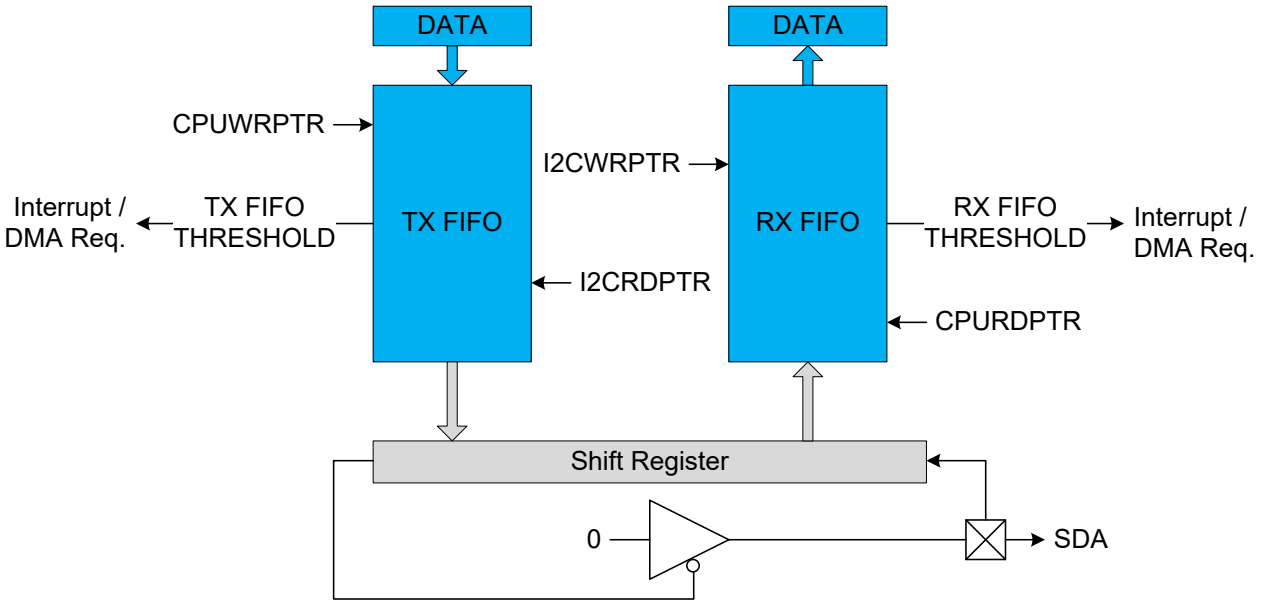
Setting the Quick Command Enable bit in the Control B register (CTRLB.QCEN) enables quick command. When quick command is enabled, the corresponding interrupt flag (INTFLAG.SB or INTFLAG.MB) is set immediately after the slave acknowledges the address. At this point, the software can either issue a stop command or a repeated start by writing CTRLB.CMD or ADDR.ADDR.

#### 28.6.3.5 FIFO Operation

For better system bus utilization, the I2C embeds up to 16-bytes FIFO capability. The receive / transmit buffer is considered to have the FIFO mode enabled when the FIFOEN bit in CTRLC register is set (CTRLC.FIFOEN = 1). By default, the FIFO can act as a 16-by-8-bit array, or as a 4-by-32-bit array, depending on the setting of the CTRLC.DATA32B bit.

The hardware around this array implements four pointers, called the CPU Write Pointer (CPUWRPTR), the CPU Read Pointer (CPURDPTR), the I2C Write pointer (I2CWRPTR) and the I2C Read pointer (I2CRDPTR). All of these pointers reset to '0'. The CPUWRPTR and CPURDPTR pointers are native to the CPU clock domain, while the I2CWRPTR and I2CRDPTR are native to the I2C domain. The location pointed to by the CPUWRPTR is the current TX FIFO. The location pointed to by the CPURDPTR becomes the current RX FIFO. Writes to DATA register by the CPU will point to TX FIFO. Reads to DATA register by the CPU will point to RX FIFO. The location pointed to by the I2CWRPTR / I2CRDPTR is logically the current shift register.

**Figure 28-15. FIFO Overview**



When using the I2C configured as Master, the Address register must be written with the desired address (ADDR.ADDR), and optionally, the transaction Length and transaction Length Enable bits (ADDR.LEN and ADDR.LENEN) can be written if the 32-bit extension is enabled (CTRLC.DATA32B).

In slave operation, the Address Match interrupt in the Interrupt Flag Status and Clear register (INTFLAG.AMATCH) is set after the address is received and and the SCL clock is stretched as long as the address is not read from the FIFO. When the address is read, the FIFO pointers are cleared in preparation for the incomming frame.

The FIFO threshold settings allow (CTRLC.TXTRHOLD, CTRLC.RXTRHOLD) allow flexible interrupt, DMA trigger and bus condition generations, as described bellow.

The FIFO is fully accessible if the SERCOM is halted, by writting the corresponding CPU FIFO pointer in the FIFOPTR register. The RX or TX FIFO can be individually cleared, by setting the respective FIFO Clear bit in the Control B register (CTRLB.FIFOCLR). The FIFO Clear must be written before data transfer begins. Writing the FIFO Clear bits while a frame is in progress will produce unpredicatable results.

#### 28.6.3.5.1 Hardware Actions in Master Mode

**Table 28-1. Interrupts Request Conditions for Valid SERCOM I²C Master Configurations**

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Action |
|---|---|---|---|---|
| Master Write | 0 | 0 | 0 | • INTFLAG.TXFE = 1 if TX FIFO is empty<br>• INTFLAG.TXFE = 1 if TX FIFO threshold reached<br>• INTFLAG.MB = 1 if TX FIFO is empty and SCL hold |
| | 0 | 1 | 0 | |
| | 0 | 1 | 1 | • INTFLAG.TXFE = 1 if TX FIFO is empty or length transaction is completed<br>• INTFLAG.TXFE = 1 if TX FIFO threshold is reached and length transaction is not completed<br>• INTFLAG.MB = 1 if TX FIFO is empty and SCL hold, or length transaction is completed |
| | 1 | 0 | 0 | • INTFLAG.TXFE = 1 if TX FIFO is empty<br>• INTFLAG.TXFE = 1 if TX FIFO threshold reached<br>• INTFLAG.MB = 1 if TX FIFO is empty and SCL hold |
| | 1 | 1 | 0 | |
| | 1 | 1 | 1 | |
| Master Read | 0 | 0 | 0 | • INTFLAG.SB = 1 if RX FIFO is full<br>• INTFLAG.RXFE = 1 if RX FIFO threshold reached |
| | 0 | 1 | 0 | |
| | 0 | 1 | 1 | |
| | 1 | 0 | 0 | |
| | 1 | 1 | 0 | |
| | 1 | 1 | 1 | |

**Table 28-2. Bus Actions for Valid SERCOM I²C Master Configurations**

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Actions |
|---|---|---|---|---|
| Master Write | 0 | 0 | 0 | • SCL hold if TX FIFO is empty |
| | 0 | 1 | 0 | |
| | 0 | 1 | 1 | • SCL hold if TX FIFO is empty and length transaction not completed<br>• Issue STOP when transaction is completed |
| | 1 | 0 | 0 | • SCL hold if TX FIFO is empty, when no automatic stop is sent<br>• STOP is sent on SW decision |
| | 1 | 1 | 0 | • SCL hold if TX FIFO is empty, when no automatic stop is sent<br>• STOP is sent on SW decision |
| | 1 | 1 | 1 | • SCL hold if TX FIFO is empty<br>• Issue STOP when transaction is completed |
| Master Read | 0 | 0 | 0 | • SCL hold if RX FIFO is full |
| | 0 | 1 | 0 | • SCL hold if RX FIFO is full |
| | 0 | 1 | 1 | • SCL stretched if RX FIFO is full<br>• ACK/NACK last frame byte, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |
| | 1 | 0 | 0 | • SCL stretched if data present in RX FIFO |
| | 1 | 1 | 0 | |
| | 1 | 1 | 1 | • SCL stretched if RX FIFO is full<br>• ACK/NACK last frame byte, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |

#### 28.6.3.5.2 Hardware Actions in Slave Mode

**Table 28-3. Interrupt Request Conditions for Valid SERCOM I²C Slave Configurations**

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Condition |
|---|---|---|---|---|
| Master Write | 0 | 0 | 0 | • INTFLAG.DRDY = 1 if RX FIFO is full<br>• INTFLAG.RXFF = 1 RX FIFO threshold is reached |
| | 0 | 1 | 0 | |
| | 0 | 1 | 1 | |

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Condition |
|---|---|---|---|---|
| | 1 | 0 | 0 | |
| | 1 | 1 | 0 | |
| | 1 | 1 | 1 | |
| Master Read | 0 | 0 | 0 | • INTFLAG.DRDY = 1 if TX FIFO is empty and SCL hold<br>• INTFLAG.TXFE = 1 if TX FIFO is empty or TX FIFO threshold is reached |
| | 0 | 1 | 0 | |
| | 0 | 1 | 1 | • INTFLAG.DRDY = 1 if TX FIFO is empty and SCL hold<br>• INTFLAG.TXFE = 1 if TX FIFO empty and lenght not completed<br>• INTFLAG.TXFE = 1 if TX FIFO threshold reached and lenght not completed |
| | 1 | 0 | 0 | • INTFLAG.DRDY = 1 if TX FIFO is empty and SCL hold<br>• INTFLAG.TXFE = 1 if TX FIFO is empty or TX FIFO threshold is reached |
| | 1 | 1 | 0 | |
| | 1 | 1 | 1 | • INTFLAG.DRDY = 1 if TX FIFO is empty and SCL hold<br>• INTFLAG.TXFE = 1 if TX FIFO empty and lenght not completed<br>• INTFLAG.TXFE = 1 if TX FIFO threshold reached and lenght not completed |

**Table 28-4. Bus Actions for Valid SERCOM $I^2C$ Slave Configurations**

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Actions |
|---|---|---|---|---|
| Master Write | 0 | 0 | 0 | • Byte mode operation<br>• SCL stretched if RX FIFO is full |
| | 0 | 1 | 0 | • 32-bit mode operation<br>• SCL stretched if RX FIFO is full<br>• ACK/NACK each 4th byte, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |
| | 0 | 1 | 1 | • 32-bit mode operation with length control<br>• SCL stretched if RX FIFO is full |

| Direction | CTRLB.SMEN | CTRLC.DATA32B | LENGTH.LENEN | Actions |
|---|---|---|---|---|
| | | | | • ACK/NACK last byte of the frame, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |
| | 1 | 0 | 0 | • SCL stretched if RX FIFO is full<br>• ACK all bytes received |
| | 1 | 1 | 0 | • 32-bit mode operation<br>• SCL stretched if RX FIFO is full<br>• ACK/NACK each 4th byte, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |
| | 1 | 1 | 1 | • 32-bit mode operation with length control<br>• SCL stretched if RX FIFO is full<br>• ACK/NACK last byte of the frame, depending on Acknowledge Action (CTRLB.ACKACT)<br>• ACK all other bytes |
| Master Read | 0 | 0 | 0 | • SCL stretched if TX FIFO is empty |
| | 0 | 1 | 0 | • SCL stretched if TX FIFO is empty |
| | 0 | 1 | 1 | • SCL stretched if TX FIFO is empty or length transfer completed |
| | 1 | 0 | 0 | • SCL stretched if TX FIFO is empty |
| | 1 | 1 | 0 | • SCL stretched if TX FIFO is empty |
| | 1 | 1 | 1 | • SCL stretched if TX FIFO is empty and length transfer is not completed |

### 28.6.4 DMA, Interrupts and Events

This chapter provides DMA and interrupt conditions when the optional FIFO is disabled. For details when the FIFO is enabled, refer to FIFO Support.

Each interrupt source has its own interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is meet. Each interrupt can be individually enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request is active until the interrupt flag is cleared, the interrupt is disabled or the $I^2C$ is reset. See the 28.8.5 INTFLAG (Slave) or 28.10.6 INTFLAG (Master) register for details on how to clear interrupt flags.

**Table 28-5. Module Request for SERCOM I²C Slave**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Data needed for transmit (TX) (Slave transmit mode) | Yes (request cleared when data is written) | | NA |
| Data received (RX) (Slave receive mode) | Yes (request cleared when data is read) | | |
| Data Ready (DRDY) | | Yes | |
| Address Match (AMATCH) | | Yes | |
| Stop received (PREC) | | Yes | |
| TX FIFO Empty (TXFE) | | Yes | |
| RX FIFO Full (RXFF) | | Yes | |
| Error (ERROR) | | Yes | |

**Table 28-6. Module Request for SERCOM I²C Master**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Data needed for transmit (TX) (Master transmit mode) | Yes (request cleared when data is written) | | NA |
| Data needed for transmit (RX) (Master transmit mode) | Yes (request cleared when data is read) | | |
| Master on Bus (MB) | | Yes | |
| Stop received (SB) | | Yes | |
| TX FIFO Empty (TXFE) | | Yes | |
| RX FIFO Full (RXFF) | | Yes | |
| Error (ERROR) | | Yes | |

#### 28.6.4.1 DMA Operation

Smart mode must be enabled for DMA operation in the Control B register by writing CTRLB.SMEN=1.

#### 28.6.4.1.1 Slave DMA

When using the I²C slave with DMA, an address match will cause the address interrupt flag (INTFLAG.ADDRMATCH) to be raised. After the interrupt has been serviced, data transfer will be performed through DMA.

The I²C slave generates the following requests:

- Write data received (RX): The request is set when master write data is received. The request is cleared when DATA is read.
- Read data needed for transmit (TX): The request is set when data is needed for a master read operation. The request is cleared when DATA is written.
- Write data received (RX): If the FIFO is disabled, the request is set when master write data is received. If the FIFO is enabled, the request is set when the RX FIFO threshold is reached (CTRLC.RXTRHOLD). The request is cleared when DATA is read.
- Read data needed for transmit (TX): If the FIFO is disabled, the request is set when data is needed for a master read operation. If the FIFO is enabled, the request is set when the TX FIFO threshold is reached (CTRLC.TXTRHOLD). The request is cleared when DATA is written.

#### 28.6.4.1.2 Master DMA

When using the I²C master with DMA, the ADDR register must be written with the desired address (ADDR.ADDR), transaction length (ADDR.LEN), and transaction length enable (ADDR.LENEN). When ADDR.LENEN is written to 1 along with ADDR.ADDR, ADDR.LEN determines the number of data bytes in the transaction from 0 to 255. DMA is then used to transfer ADDR.LEN bytes followed by an automatically generated NACK (for master reads) and a STOP.

If a NACK is received by the slave for a master write transaction before ADDR.LEN bytes, a STOP will be automatically generated and the length error (STATUS.LENERR) will be raised along with the INTFLAG.ERROR interrupt.

The I²C master generates the following requests:

- Read data received (RX): The request is set when master read data is received. The request is cleared when DATA is read.
- Write data needed for transmit (TX): The request is set when data is needed for a master write operation. The request is cleared when DATA is written.
- Read data received (RX): If the FIFO is disabled, the request is set when master read data is received. If the FIFO is enabled, the request is set when the RX FIFO threshold is reached. The request is cleared when DATA is read.
- Write data needed for transmit (TX): If the FIFO is disabled, the request is set when data is needed for a master write operation. If the FIFO is enabled, the request is set when the TX FIFO threshold is reached (CTRLC.TXTRHOLD). The request is cleared when DATA is written.

### 28.6.4.2 Interrupts

The I²C slave has the following interrupt sources. These are asynchronous interrupts. They can wake-up the device from any sleep mode:

- Error (ERROR)
- RX FIFO Full (RXFF)
- TX FIFO Empty (TXFE)
- Data Ready (DRDY)
- Address Match (AMATCH)

- Stop Received (PREC)

The I²C master has the following interrupt sources. These are asynchronous interrupts. They can wake-up the device from any sleep mode:

- Error (ERROR)
- RX FIFO Full (RXFF)
- TX FIFO Empty (TXFE)
- Slave on Bus (SB)
- Master on Bus (MB)

Each interrupt source has its own interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is meet. Each interrupt can be individually enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request active until the interrupt flag is cleared, the interrupt is disabled or the I²C is reset. See the INTFLAG register for details on how to clear interrupt flags.

The value of INTFLAG indicates which interrupt is executed. Note that interrupts must be globally enabled for interrupt requests. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 28.6.4.3 Events

Not applicable.

### 28.6.5 Sleep Mode Operation

**I²C Master Operation**

The generic clock (GCLK_SERCOMx_CORE) will continue to run in idle sleep mode. If the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) is '1', the GLK_SERCOMx_CORE will also run in standby sleep mode. Any interrupt can wake up the device.

If CTRLA.RUNSTDBY=0, the GLK_SERCOMx_CORE will be disabled after any ongoing transaction is finished. Any interrupt can wake up the device.

**I²C Slave Operation**

Writing CTRLA.RUNSTDBY=1 will allow the Address Match interrupt to wake up the device.

When CTRLA.RUNSTDBY=0, all receptions will be dropped.

### 28.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in the CTRLA register (CTRLA.SWRST)
- Enable bit in the CTRLA register (CTRLA.ENABLE)
- Command bits in CTRLB register (CTRLB.CMD)
- FIFO Clear bits in CTRLB register (CTRLB.FIFOCLR)
- Write to Bus State bits in the Status register (STATUS.BUSSTATE)

- Address bits in the Address register (ADDR.ADDR) when in master operation.

The following registers are synchronized when written:

- Data (DATA) when in master operation

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 28.7 Register Summary - I2C Slave

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | SEXTTOEN | | SDAHOLD[1:0] | | | | | PINOUT |
| | | 31:24 | | LOWTOUT | | | SCLSM | | SPEED[1:0] | |
| 0x04 | CTRLB | 7:0 | | | | | | | | |
| | | 15:8 | AMODE[1:0] | | | | | AACKEN | GCMD | SMEN |
| | | 23:16 | FIFOCLR[1:0] | | | | | ACKACT | CMD[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENSET | 7:0 | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTFLAG | 7:0 | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| 0x19 | Reserved | | | | | | | | | |
| 0x1A | STATUS | 7:0 | CLKHOLD | LOWTOUT | | SR | DIR | RXNACK | COLL | BUSERR |
| | | 15:8 | | | | | LENERR | HS | SEXTTOUT | |
| 0x1C | SYNCBUSY | 7:0 | | | | | | SYSOP | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | ADDR | 7:0 | ADDR[6:0] | | | | | | | GENCEN |
| | | 15:8 | TENBITEN | | | | | ADDR[9:7] | | |
| | | 23:16 | ADDRMASK[6:0] | | | | | | | |
| | | 31:24 | | | | | | ADDRMASK[9:7] | | |
| 0x28 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| 0x2A ... 0x33 | Reserved | | | | | | | | | |
| 0x34 | FIFOSPACE | 7:0 | | | | TXSPACE[4:0] | | | | |
| | | 15:8 | | | | RXSPACE[4:0] | | | | |
| 0x36 | FIFOPTR | 7:0 | | | | | CPUWRPTR[3:0] | | | |
| | | 15:8 | | | | | CPURDPTR[3:0] | | | |

## 28.8 Register Description - I²C Slave

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 28.5.8 Register Access Protection.

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to 28.6.6 Synchronization.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 28.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | LOWTOUT | | | SCLSM | | SPEED[1:0] | |
| Access | | R/W | | | R/W | | R/W | R/W |
| Reset | | 0 | | | 0 | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEXTTOEN | | SDAHOLD[1:0] | | | | | PINOUT |
| Access | R/W | | R/W | R/W | | | | R/W |
| Reset | 0 | | 0 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – LOWTOUT** SCL Low Time-Out
This bit enables the SCL low time-out. If SCL is held low for 25ms-35ms, the slave will release its clock hold, if enabled, and reset the internal state machine. Any interrupt flags set at the time of time-out will remain set.

| Value | Description |
|---|---|
| 0 | Time-out disabled. |
| 1 | Time-out enabled. |

**Bit 27 – SCLSM** SCL Clock Stretch Mode
This bit controls when SCL will be stretched for software interaction.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | SCL stretch according to Figure 28-10 |
| 1 | SCL stretch only after ACK bit according to Figure 28-11 |

**Bits 25:24 – SPEED[1:0]** Transfer Speed
These bits define bus speed.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0x0 | Standard-mode (Sm) up to 100 kHz and Fast-mode (Fm) up to 400 kHz |
| 0x1 | Fast-mode Plus (Fm+) up to 1 MHz |
| 0x2 | High-speed mode (Hs-mode) up to 3.4 MHz |
| 0x3 | Reserved |

**Bit 23 – SEXTTOEN** Slave SCL Low Extend Time-Out
This bit enables the slave SCL low extend time-out. If SCL is cumulatively held low for greater than 25ms from the initial START to a STOP, the slave will release its clock hold if enabled and reset the internal state machine. Any interrupt flags set at the time of time-out will remain set. If the address was recognized, PREC will be set when a STOP is received.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Time-out disabled |
| 1 | Time-out enabled |

**Bits 21:20 – SDAHOLD[1:0]** SDA Hold Time
These bits define the SDA hold time with respect to the negative edge of SCL.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | DIS | Disabled |
| 0x1 | 75 | 50-100ns hold time |
| 0x2 | 450 | 300-600ns hold time |
| 0x3 | 600 | 400-800ns hold time |

**Bit 16 – PINOUT** Pin Usage
This bit sets the pin usage to either two- or four-wire operation:

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | 4-wire operation disabled |
| 1 | 4-wire operation enabled |

**Bit 7 – RUNSTDBY** Run in Standby
This bit defines the functionality in standby sleep mode.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Disabled – All reception is dropped. |
| 1 | Wake on address match, if enabled. |

**Bits 4:2 – MODE[2:0]** Operating Mode
These bits must be written to 0x04 to select the $I^2C$ slave serial communication interface of the SERCOM.

These bits are not synchronized.

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 28.8.2 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIFOCLR[1:0] | | | | | ACKACT | CMD[1:0] | |
| Access | R/W | R/W | | | | R/W | W | W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AMODE[1:0] | | | | | AACKEN | GCMD | SMEN |
| Access | R/W | R/W | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 23:22 – FIFOCLR[1:0]** FIFO Clear
When these bits are set, the corresponding FIFO will be cleared. The bits will automatically clear when SYNCBUSY.SYSOP = 0.

These bits are not enable-protected.

| FIFOCLR[1:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | TXFIFO | Clear TX FIFO |
| 0x2 | RXFIFO | Clear RX FIFO |
| 0x3 | BOTH | Clear both TX/RX FIFO |

**Bit 18 – ACKACT** Acknowledge Action
This bit defines the slave's acknowledge behavior after an address or data byte is received from the master. The acknowledge action is executed when a command is written to the CMD bits. If smart mode is enabled (CTRLB.SMEN=1), the acknowledge action is performed when the DATA register is read.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | Send ACK |
| 1 | Send NACK |

**Bits 17:16 – CMD[1:0]** Command
This bit field triggers the slave operation as the below. The CMD bits are strobe bits, and always read as zero. The operation is dependent on the slave interrupt flags, INTFLAG.DRDY and INTFLAG.AMATCH, in addition to STATUS.DIR.

All interrupt flags (INTFLAG.DRDY, INTFLAG.AMATCH and INTFLAG.PREC) are automatically cleared when a command is given.

This bit is not enable-protected.

**Table 28-7. Command Description**

| CMD[1:0] | DIR | Action |
|---|---|---|
| 0x0 | X | (No action) |
| 0x1 | X | (Reserved) |
| 0x2 | Used to complete a transaction in response to a data interrupt (DRDY) | |
| | 0 (Master write) | Execute acknowledge action succeeded by waiting for any start (S/Sr) condition |
| | 1 (Master read) | Wait for any start (S/Sr) condition |
| 0x3 | Used in response to an address interrupt (AMATCH) | |
| | 0 (Master write) | Execute acknowledge action succeeded by reception of next byte |
| | 1 (Master read) | Execute acknowledge action succeeded by slave data interrupt |
| | Used in response to a data interrupt (DRDY) | |
| | 0 (Master write) | Execute acknowledge action succeeded by reception of next byte |
| | 1 (Master read) | Execute a byte read operation followed by ACK/NACK reception |

**Bits 15:14 – AMODE[1:0]** Address Mode
These bits set the addressing mode.

These bits are not write-synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | MASK | The slave responds to the address written in ADDR.ADDR masked by the value in ADDR.ADDRMASK.<br>See *SERCOM – Serial Communication Interface* for additional information. |
| 0x1 | 2_ADDRS | The slave responds to the two unique addresses in ADDR.ADDR and ADDR.ADDRMASK. |
| 0x2 | RANGE | The slave responds to the range of addresses between and including ADDR.ADDR and ADDR.ADDRMASK. ADDR.ADDR is the upper limit. |
| 0x3 | - | Reserved. |

**Bit 10 – AACKEN** Automatic Acknowledge Enable
This bit enables the address to be automatically acknowledged if there is an address match.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Automatic acknowledge is disabled. |
| 1 | Automatic acknowledge is enabled. |

**Bit 9 – GCMD** PMBus Group Command
This bit enables PMBus group command support. When enabled, the Stop Recived interrupt flag (INTFLAG.PREC) will be set when a STOP condition is detected if the slave has been addressed since the last STOP condition on the bus.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Group command is disabled. |
| 1 | Group command is enabled. |

**Bit 8 – SMEN** Smart Mode Enable
When smart mode is enabled, data is acknowledged automatically when DATA.DATA is read.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Smart mode is disabled. |
| 1 | Smart mode is enabled. |

**Related Links**
25. SERCOM – Serial Communication Interface

### 28.8.3 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 4 – RXFF** RX FIFO Full Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the RX FIFO Full bit, which disables the RX FIFO Full interrupt.

| Value | Description |
|---|---|
| 0 | The RX FIFO Full interrupt is disabled. |
| 1 | The RX FIFO Full interrupt is enabled. |

**Bit 3 – TXFE** TX FIFO Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the TX FIFO Empty bit, which disables the TX FIFO Empty interrupt.

| Value | Description |
|---|---|
| 0 | The TX FIFO Empty interrupt is disabled. |
| 1 | The TX FIFO Empty interrupt is enabled. |

**Bit 2 – DRDY** Data Ready Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Data Ready bit, which disables the Data Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Data Ready interrupt is disabled. |
| 1 | The Data Ready interrupt is enabled. |

**Bit 1 – AMATCH** Address Match Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Address Match Interrupt Enable bit, which disables the Address Match interrupt.

| Value | Description |
|---|---|
| 0 | The Address Match interrupt is disabled. |
| 1 | The Address Match interrupt is enabled. |

**Bit 0 – PREC** Stop Received Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Stop Received Interrupt Enable bit, which disables the Stop Received interrupt.

| Value | Description |
|---|---|
| 0 | The Stop Received interrupt is disabled. |
| 1 | The Stop Received interrupt is enabled. |

### 28.8.4 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x16
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 4 – RXFF** RX FIFO Full Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the RX FIFO Full bit, which enables the RX FIFO Full interrupt.

| Value | Description |
|---|---|
| 0 | The RX FIFO Full interrupt is disabled. |
| 1 | The RX FIFO Full interrupt is enabled. |

**Bit 3 – TXFE** TX FIFO Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the TX FIFO Empty bit, which enables the TX FIFO Empty interrupt.

| Value | Description |
|---|---|
| 0 | The TX FIFO Empty interrupt is disabled. |
| 1 | The TX FIFO Empty interrupt is enabled. |

**Bit 2 – DRDY** Data Ready Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Data Ready bit, which enables the Data Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Data Ready interrupt is disabled. |
| 1 | The Data Ready interrupt is enabled. |

**Bit 1 – AMATCH** Address Match Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Address Match Interrupt Enable bit, which enables the Address Match interrupt.

| Value | Description |
|---|---|
| 0 | The Address Match interrupt is disabled. |
| 1 | The Address Match interrupt is enabled. |

**Bit 0 – PREC** Stop Received Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Stop Received Interrupt Enable bit, which enables the Stop Received interrupt.

| Value | Description |
|---|---|
| 0 | The Stop Received interrupt is disabled. |
| 1 | The Stop Received interrupt is enabled. |

### 28.8.5 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | DRDY | AMATCH | PREC |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR** Error
This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. The corresponding bits in STATUS are SEXTTOUT, LOWTOUT, COLL, and BUSERR.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 4 – RXFF** RX FIFO Full
This flag is set when RX FIFO Threshold locations are fullfilled.

The flag is cleared when the RX FIFO is empty.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the RX FIFO Full interrupt flag.

**Bit 3 – TXFE** TX FIFO Empty
This flag is set when TX FIFO Threshold locations are available.

The flag is cleared when the TX FIFO is full.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the TX FIFO Empty interrupt flag.

**Bit 2 – DRDY** Data Ready
This flag is set when a $I^2C$ slave byte transmission is successfully completed.

The flag is cleared by hardware when either:

- Writing to the DATA register.
- Reading the DATA register with smart mode enabled.
- Writing a valid command to the CMD register.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Data Ready interrupt flag.

**Bit 1 – AMATCH** Address Match
This flag is set when the $I^2C$ slave address match logic detects that a valid address has been received.

The flag is cleared by hardware when CTRL.CMD is written.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Address Match interrupt flag. When cleared, an ACK/NACK will be sent according to CTRLB.ACKACT.

**Bit 0 – PREC** Stop Received
This flag is set when a stop condition is detected for a transaction being processed. A stop condition detected between a bus master and another slave will not set this flag, unless the PMBus Group Command is enabled in the Control B register (CTRLB.GCMD=1).

This flag is cleared by hardware after a command is issued on the next address match.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Stop Received interrupt flag.

### 28.8.6 Status

**Name:** STATUS
**Offset:** 0x1A
**Reset:** 0x0000

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LENERR | HS | SEXTTOUT | |
| Access | | | | | R/W | R/W | R/W | |
| Reset | | | | | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLKHOLD | LOWTOUT | | SR | DIR | RXNACK | COLL | BUSERR |
| Access | R | R/W | | R | R | R | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bit 11 – LENERR** Transaction Length Error
This bit is set when the length counter is enabled (LENGTH.LENEN) and a STOP or repeated START is received before or after the length in LENGTH.LEN is reached.
This bit is cleared automatically when responding to a new start condition with ACK or NACK (CTRLB.CMD=0x3) or when INTFLAG.AMATCH is cleared.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the status.

**Bit 10 – HS** High-speed
This bit is set if the slave detects a START followed by a Master Code transmission.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the status. However, this flag is automatically cleared when a STOP is received.

**Bit 9 – SEXTTOUT** Slave SCL Low Extend Time-Out
This bit is set if a slave SCL low extend time-out occurs.

This bit is cleared automatically if responding to a new start condition with ACK or NACK (write 3 to CTRLB.CMD) or when INTFLAG.AMATCH is cleared.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the status.

| Value | Description |
|---|---|
| 0 | No SCL low extend time-out has occurred. |
| 1 | SCL low extend time-out has occurred. |

**Bit 7 – CLKHOLD** Clock Hold
The slave Clock Hold bit (STATUS.CLKHOLD) is set when the slave is holding the SCL line low, stretching the I2C clock. Software should consider this bit a read-only status flag that is set when INTFLAG.DRDY or INTFLAG.AMATCH is set.

This bit is automatically cleared when the corresponding interrupt is also cleared.

**Bit 6 – LOWTOUT** SCL Low Time-out
This bit is set if an SCL low time-out occurs.

This bit is cleared automatically if responding to a new start condition with ACK or NACK (write 3 to CTRLB.CMD) or when INTFLAG.AMATCH is cleared.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the status.

| Value | Description |
|---|---|
| 0 | No SCL low time-out has occurred. |
| 1 | SCL low time-out has occurred. |

**Bit 4 – SR** Repeated Start
When INTFLAG.AMATCH is raised due to an address match, SR indicates a repeated start or start condition.

This flag is only valid while the INTFLAG.AMATCH flag is one.

| Value | Description |
|---|---|
| 0 | Start condition on last address match |
| 1 | Repeated start condition on last address match |

**Bit 3 – DIR** Read / Write Direction
The Read/Write Direction (STATUS.DIR) bit stores the direction of the last address packet received from a master.

| Value | Description |
|---|---|
| 0 | Master write operation is in progress. |
| 1 | Master read operation is in progress. |

**Bit 2 – RXNACK** Received Not Acknowledge
This bit indicates whether the last data packet sent was acknowledged or not.

| Value | Description |
|---|---|
| 0 | Master responded with ACK. |
| 1 | Master responded with NACK. |

**Bit 1 – COLL** Transmit Collision
If set, the I2C slave was not able to transmit a high data or NACK bit, the I2C slave will immediately release the SDA and SCL lines and wait for the next packet addressed to it.

This flag is intended for the SMBus address resolution protocol (ARP). A detected collision in non-ARP situations indicates that there has been a protocol violation, and should be treated as a bus error.

Note that this status will not trigger any interrupt, and should be checked by software to verify that the data were sent correctly. This bit is cleared automatically if responding to an address match with an ACK or a NACK (writing 0x3 to CTRLB.CMD), or INTFLAG.AMATCH is cleared.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the status.

| Value | Description |
|---|---|
| 0 | No collision detected on last data byte sent. |
| 1 | Collision detected on last data byte sent. |

**Bit 0 – BUSERR** Bus Error
The Bus Error bit (STATUS.BUSERR) indicates that an illegal bus condition has occurred on the bus, regardless of bus ownership. An illegal bus condition is detected if a protocol violating start, repeated start or stop is detected on the I2C bus lines. A start condition directly followed by a stop condition is one example of a protocol violation. If a time-out occurs during a frame, this is also considered a protocol violation, and will set STATUS.BUSERR.

This bit is cleared automatically if responding to an address match with an ACK or a NACK (writing 0x3 to CTRLB.CMD) or INTFLAG.AMATCH is cleared.

Writing a '1' to this bit will clear the status.

Writing a '0' to this bit has no effect.

| Value | Description |
|---|---|
| 0 | No bus error detected. |
| 1 | Bus error detected. |

### 28.8.7 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x1C
**Reset:** 0x00000000

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SYSOP | ENABLE | SWRST |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SYSOP** System Operation Synchronization Busy
Writing CTRLB.FIFOCLR when the SERCOM is enabled requires synchronization. When written, the SYNCBUSY.SYSOP bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | System operation synchronization is not busy. |
| 1 | System operation synchronization is busy. |

**Bit 1 – ENABLE** SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | Enable synchronization is not busy. |
| 1 | Enable synchronization is busy. |

**Bit 0 – SWRST** Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | SWRST synchronization is not busy. |
| 1 | SWRST synchronization is busy. |

### 28.8.8 Address

**Name:** ADDR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ADDRMASK[9:7] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDRMASK[6:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TENBITEN | | | | | ADDR[9:7] | | |
| Access | R/W | | | | | R/W | R/W | R/W |
| Reset | 0 | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[6:0] | | | | | | | GENCEN |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 26:17 – ADDRMASK[9:0]** Address Mask
These bits act as a second address match register, an address mask register or the lower limit of an address range, depending on the CTRLB.AMODE setting.

**Bit 15 – TENBITEN** Ten Bit Addressing Enable

| Value | Description |
|---|---|
| 0 | 10-bit address recognition disabled. |
| 1 | 10-bit address recognition enabled. |

**Bits 10:1 – ADDR[9:0]** Address
These bits contain the $I^2C$ slave address used by the slave address match logic to determine if a master has addressed the slave.

When using 7-bit addressing, the slave address is represented by ADDR[6:0].

When using 10-bit addressing (ADDR.TENBITEN=1), the slave address is represented by ADDR[9:0]

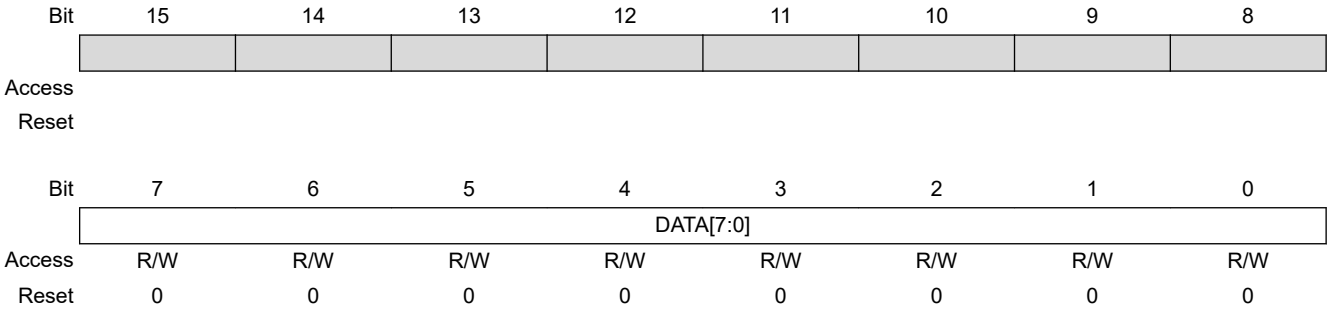
When the address match logic detects a match, INTFLAG.AMATCH is set and STATUS.DIR is updated to indicate whether it is a read or a write transaction.

**Bit 0 – GENCEN** General Call Address Enable
A general call address is an address consisting of all-zeroes, including the direction bit (master write).

| Value | Description |
|---|---|
| 0 | General call address recognition disabled. |
| 1 | General call address recognition enabled. |

### 28.8.9 Data

**Name:** DATA
**Offset:** 0x28
**Reset:** 0x0000
**Property:** Write-Synchronized, Read-Synchronized




**Bits 7:0 – DATA[7:0]** Data
The slave data register I/O location (DATA.DATA) provides access to the master transmit and receive data buffers. Reading valid data or writing data to be transmitted can be successfully done only when SCL is held low by the slave (STATUS.CLKHOLD is set). An exception occurs when reading the last data byte after the stop condition has been received.

Accessing DATA.DATA auto-triggers $I^2C$ bus operations. The operation performed depends on the state of CTRLB.ACKACT, CTRLB.SMEN and the type of access (read/write).

Writing or reading DATA.DATA when not in smart mode does not require synchronization.

### 28.8.10 FIFO Space

**Name:** FIFOSPACE
**Offset:** 0x34
**Reset:** 0x0000
**Property:** -

This register allows the user to identify the number of bytes present in each TX and RX FIFO.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | TXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 12:8 – RXSPACE[4:0]** RX FIFO Filled Space
These bits return the number filled locations in the RX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

**Bits 4:0 – TXSPACE[4:0]** TX FIFO Empty Space
These bits return the number of available locations in the TX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

### 28.8.11 FIFO CPU Pointers

**Name:** FIFOPTR
**Offset:** 0x36
**Reset:** 0x0000
**Property:** -

This register provides a copy of internal CPU TX and RX FIFO pointers.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPURDPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPUWRPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – CPURDPTR[3:0]** RX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. Reading DATA register, will return RXFIFO[CPURDPTR] location value.

**Bits 3:0 – CPUWRPTR[3:0]** TX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. When writting to DATA register, the DATA will be written to TXFIFO[CPUWRPTR] location.

## 28.9 Register Summary - I2C Master

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | SEXTTOEN | MEXTTOEN | SDAHOLD[1:0] | | | | | PINOUT |
| | | 31:24 | | LOWTOUT | INACTOUT[1:0] | | SCLSM | | SPEED[1:0] | |
| 0x04 | CTRLB | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | QCEN | SMEN |
| | | 23:16 | FIFOCLR[1:0] | | | | | ACKACT | CMD[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | BAUD | 7:0 | BAUD[7:0] | | | | | | | |
| | | 15:8 | BAUDLOW[7:0] | | | | | | | |
| | | 23:16 | HSBAUD[7:0] | | | | | | | |
| | | 31:24 | HSBAUDLOW[7:0] | | | | | | | |
| 0x10 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | ERROR | | | RXFF | TXFE | | SB | MB |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENSET | 7:0 | ERROR | | | RXFF | TXFE | | SB | MB |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTFLAG | 7:0 | ERROR | | | RXFF | TXFE | | SB | MB |
| 0x18 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| 0x1A | STATUS | 7:0 | CLKHOLD | LOWTOUT | BUSSTATE[1:0] | | | RXNACK | ARBLOST | BUSERR |
| | | 15:8 | | | | | | LENERR | SEXTTOUT | MEXTTOUT |
| 0x1C | SYNCBUSY | 7:0 | | | | | | SYSOP | ENABLE | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | ADDR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | TENBITEN | HS | LENEN | | | ADDR[10:8] | | |
| | | 23:16 | LEN[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | DBGCTRL | 7:0 | | | | | | | | DBGSTOP |
| 0x31 ... | Reserved | | | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x33 | | | | | | | | | | |
| 0x34 | FIFOSPACE | 7:0 | | | | TXSPACE[4:0] | | | | |
| | | 15:8 | | | | RXSPACE[4:0] | | | | |
| 0x36 | FIFOPTR | 7:0 | | | | | CPUWRPTR[3:0] | | | |
| | | 15:8 | | | | | CPURDPTR[3:0] | | | |

## 28.10 Register Description - I²C Master

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 28.5.8 Register Access Protection.

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to 28.6.6 Synchronization.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 28.10.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | LOWTOUT | INACTOUT[1:0] | | SCLSM | | SPEED[1:0] | |
| Access | | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEXTTOEN | MEXTTOEN | SDAHOLD[1:0] | | | | | PINOUT |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – LOWTOUT** SCL Low Time-Out
This bit enables the SCL low time-out. If SCL is held low for 25ms-35ms, the master will release its clock hold, if enabled, and complete the current transaction. A stop condition will automatically be transmitted.

INTFLAG.SB or INTFLAG.MB will be set as normal, but the clock hold will be released. The STATUS.LOWTOUT and STATUS.BUSERR status bits will be set.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Time-out disabled. |
| 1 | Time-out enabled. |

**Bits 29:28 – INACTOUT[1:0]** Inactive Time-Out
If the inactive bus time-out is enabled and the bus is inactive for longer than the time-out setting, the bus state logic will be set to idle. An inactive bus arise when either an $I^2C$ master or slave is holding the SCL low.

Enabling this option is necessary for SMBus compatibility, but can also be used in a non-SMBus set-up.

Calculated time-out periods are based on a 100kHz baud rate.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | DIS | Disabled |
| 0x1 | 55US | 5-6 SCL cycle time-out (50-60µs) |
| 0x2 | 105US | 10-11 SCL cycle time-out (100-110µs) |
| 0x3 | 205US | 20-21 SCL cycle time-out (200-210µs) |

**Bit 27 – SCLSM** SCL Clock Stretch Mode
This bit controls when SCL will be stretched for software interaction.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | SCL stretch according to Figure 28-5. |
| 1 | SCL stretch only after ACK bit, Figure 28-6. |

**Bits 25:24 – SPEED[1:0]** Transfer Speed
These bits define bus speed.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0x0 | Standard-mode (Sm) up to 100 kHz and Fast-mode (Fm) up to 400 kHz |
| 0x1 | Fast-mode Plus (Fm+) up to 1 MHz |
| 0x2 | High-speed mode (Hs-mode) up to 3.4 MHz |
| 0x3 | Reserved |

**Bit 23 – SEXTTOEN** Slave SCL Low Extend Time-Out
This bit enables the slave SCL low extend time-out. If SCL is cumulatively held low for greater than 25ms from the initial START to a STOP, the master will release its clock hold if enabled, and complete the current transaction. A STOP will automatically be transmitted.

SB or MB will be set as normal, but CLKHOLD will be release. The MEXTTOUT and BUSERR status bits will be set.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Time-out disabled |
| 1 | Time-out enabled |

**Bit 22 – MEXTTOEN** Master SCL Low Extend Time-Out
This bit enables the master SCL low extend time-out. If SCL is cumulatively held low for greater than 10ms from START-to-ACK, ACK-to-ACK, or ACK-to-STOP the master will release its clock hold if enabled, and complete the current transaction. A STOP will automatically be transmitted.

SB or MB will be set as normal, but CLKHOLD will be released. The MEXTTOUT and BUSERR status bits will be set.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Time-out disabled |
| 1 | Time-out enabled |

**Bits 21:20 – SDAHOLD[1:0]** SDA Hold Time
These bits define the SDA hold time with respect to the negative edge of SCL.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | DIS | Disabled |
| 0x1 | 75NS | 50-100ns hold time |
| 0x2 | 450NS | 300-600ns hold time |
| 0x3 | 600NS | 400-800ns hold time |

**Bit 16 – PINOUT** Pin Usage
This bit set the pin usage to either two- or four-wire operation:

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | 4-wire operation disabled. |
| 1 | 4-wire operation enabled. |

**Bit 7 – RUNSTDBY** Run in Standby
This bit defines the functionality in standby sleep mode.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | GCLK_SERCOMx_CORE is disabled and the $I^2C$ master will not operate in standby sleep mode. |
| 1 | GCLK_SERCOMx_CORE is enabled in all sleep modes. |

**Bits 4:2 – MODE[2:0]** Operating Mode
These bits must be written to 0x5 to select the $I^2C$ master serial communication interface of the SERCOM.

These bits are not synchronized.

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Enable Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 28.10.2 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FIFOCLR[1:0] | | | | | ACKACT | CMD[1:0] | |
| Access | R/W | R/W | | | | R/W | W | W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | QCEN | SMEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 23:22 – FIFOCLR[1:0]** FIFO Clear
When these bits are set, the corresponding FIFO will be cleared. The bits will automatically clear when SYNCBUSY.SYSOP = 0.

These bits are not enable-protected.

| FIFOCLR[1:0] | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | TXFIFO | Clear TX FIFO |
| 0x2 | RXFIFO | Clear RX FIFO |
| 0x3 | BOTH | Clear both TX/RX FIFO |

**Bit 18 – ACKACT** Acknowledge Action
This bit defines the $I^2C$ master's acknowledge behavior after a data byte is received from the $I^2C$ slave. The acknowledge action is executed when a command is written to CTRLB.CMD, or if smart mode is enabled (CTRLB.SMEN is written to one), when DATA.DATA is read.

This bit is not enable-protected.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Send ACK. |
| 1 | Send NACK. |

**Bits 17:16 – CMD[1:0]** Command
Writing these bits triggers a master operation as described below. The CMD bits are strobe bits, and always read as zero. The acknowledge action is only valid in master read mode. In master write mode, a command will only result in a repeated start or stop condition. The CTRLB.ACKACT bit and the CMD bits can be written at the same time, and then the acknowledge action will be updated before the command is triggered.

Commands can only be issued when either the Slave on Bus interrupt flag (INTFLAG.SB) or Master on Bus interrupt flag (INTFLAG.MB) is '1'.

If CMD 0x1 is issued, a repeated start will be issued followed by the transmission of the current address in ADDR.ADDR. If another address is desired, ADDR.ADDR must be written instead of the CMD bits. This will trigger a repeated start followed by transmission of the new address.

Issuing a command will set the System Operation bit in the Synchronization Busy register (SYNCBUSY.SYSOP).

**Table 28-8. Command Description**

| CMD[1:0] | Direction | Action |
|---|---|---|
| 0x0 | X | (No action) |
| 0x1 | X | Execute acknowledge action succeeded by repeated Start |
| 0x2 | 0 (Write) | No operation |
| | 1 (Read) | Execute acknowledge action succeeded by a byte read operation |
| 0x3 | X | Execute acknowledge action succeeded by issuing a stop condition |

These bits are not enable-protected.

**Bit 9 – QCEN** Quick Command Enable
This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Quick Command is disabled. |
| 1 | Quick Command is enabled. |

**Bit 8 – SMEN** Smart Mode Enable
When smart mode is enabled, acknowledge action is sent when DATA.DATA is read.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Smart mode is disabled. |
| 1 | Smart mode is enabled. |

### 28.10.3 Baud Rate

**Name:** BAUD
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HSBAUDLOW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HSBAUD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BAUDLOW[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – HSBAUDLOW[7:0]** High Speed Master Baud Rate Low
HSBAUDLOW non-zero: HSBAUDLOW indicates the SCL low time in High-speed mode according to

$$\mathrm{HSBAUDLOW} = f_{\mathrm{GCLK}} \cdot T_{\mathrm{LOW}} - 1$$

HSBAUDLOW equal to zero: The HSBAUD register is used to time $T_{LOW}$, $T_{HIGH}$, $T_{SU;STO}$, $T_{HD;STA}$ and $T_{SU;STA}$. $T_{BUF}$ is timed by the BAUD register.

**Bits 23:16 – HSBAUD[7:0]** High Speed Master Baud Rate
This bit field indicates the SCL high time in High-speed mode according to the following formula. When HSBAUDLOW is zero, $T_{LOW}$, $T_{HIGH}$, $T_{SU;STO}$, $T_{HD;STA}$ and $T_{SU;STA}$ are derived using this formula. $T_{BUF}$ is timed by the BAUD register.

$$\mathrm{HSBAUD} = f_{\mathrm{GCLK}} \cdot T_{\mathrm{HIGH}} - 1$$

**Bits 15:8 – BAUDLOW[7:0]** Master Baud Rate Low
If this bit field is non-zero, the SCL low time will be described by the value written.

For more information on how to calculate the frequency, see SERCOM 25.6.2.3 Clock Generation – Baud-Rate Generator.

**Bits 7:0 – BAUD[7:0]** Master Baud Rate
This bit field is used to derive the SCL high time if BAUD.BAUDLOW is non-zero. If BAUD.BAUDLOW is zero, BAUD will be used to generate both high and low periods of the SCL.

For more information on how to calculate the frequency, see SERCOM 25.6.2.3 Clock Generation – Baud-Rate Generator.

### 28.10.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | | SB | MB |
| Access | R/W | | | R/W | R/W | | R/W | R/W |
| Reset | 0 | | | 0 | 0 | | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 4 – RXFF** RX FIFO Full Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the RX FIFO Full bit, which disables the RX FIFO Full interrupt.

| Value | Description |
|---|---|
| 0 | The RX FIFO Full interrupt is disabled. |
| 1 | The RX FIFO Full interrupt is enabled. |

**Bit 3 – TXFE** TX FIFO Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the TX FIFO Empty bit, which disables the TX FIFO Empty interrupt.

| Value | Description |
|---|---|
| 0 | The TX FIFO Empty interrupt is disabled. |
| 1 | The TX FIFO Empty interrupt is enabled. |

**Bit 1 – SB** Slave on Bus Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Slave on Bus Interrupt Enable bit, which disables the Slave on Bus interrupt.

| Value | Description |
|---|---|
| 0 | The Slave on Bus interrupt is disabled. |
| 1 | The Slave on Bus interrupt is enabled. |

**Bit 0 – MB** Master on Bus Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Master on Bus Interrupt Enable bit, which disables the Master on Bus interrupt.

| Value | Description |
|---|---|
| 0 | The Master on Bus interrupt is disabled. |
| 1 | The Master on Bus interrupt is enabled. |

### 28.10.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x16
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | | SB | MB |
| Access | R/W | | | R/W | R/W | | R/W | R/W |
| Reset | 0 | | | 0 | 0 | | 0 | 0 |

**Bit 7 – ERROR** Error Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 4 – RXFF** RX FIFO Full Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the RX FIFO Full bit, which enables the RX FIFO Full interrupt.

| Value | Description |
|---|---|
| 0 | The RX FIFO Full interrupt is disabled. |
| 1 | The RX FIFO Full interrupt is enabled. |

**Bit 3 – TXFE** TX FIFO Empty Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the TX FIFO Empty bit, which enables the TX FIFO Empty interrupt.

| Value | Description |
|---|---|
| 0 | The TX FIFO Empty interrupt is disabled. |
| 1 | The TX FIFO Empty interrupt is enabled. |

**Bit 1 – SB** Slave on Bus Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Slave on Bus Interrupt Enable bit, which enables the Slave on Bus interrupt.

| Value | Description |
|---|---|
| 0 | The Slave on Bus interrupt is disabled. |
| 1 | The Slave on Bus interrupt is enabled. |

**Bit 0 – MB** Master on Bus Interrupt Enable
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Master on Bus Interrupt Enable bit, which enables the Master on Bus interrupt.

| Value | Description |
|---|---|
| 0 | The Master on Bus interrupt is disabled. |
| 1 | The Master on Bus interrupt is enabled. |

### 28.10.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | | RXFF | TXFE | | SB | MB |
| Access | R/W | | | R/W | R/W | | R/W | R/W |
| Reset | 0 | | | 0 | 0 | | 0 | 0 |

**Bit 7 – ERROR** Error
This flag is cleared by writing '1' to it.

This bit is set when any error is detected. Errors that will set this flag have corresponding status bits in the STATUS register. These status bits are LENERR, SEXTTOUT, MEXTTOUT, LOWTOUT, ARBLOST, and BUSERR.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 4 – RXFF** RX FIFO Full
This flag is set when RX FIFO Threshold locations are fullfilled.

The flag is cleared when the RX FIFO is empty.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the RX FIFO Full interrupt flag.

**Bit 3 – TXFE** TX FIFO Empty
This flag is set when TX FIFO Threshold locations are available.

The flag is cleared when the TX FIFO is full.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the TX FIFO Empty interrupt flag.

**Bit 1 – SB** Slave on Bus
The Slave on Bus flag (SB) is set when a byte is successfully received in master read mode, i.e., no arbitration lost or bus error occurred during the operation. When this flag is set, the master forces the SCL line low, stretching the $I^2C$ clock period. The SCL line will be released and SB will be cleared on one of the following actions:

- Writing to ADDR.ADDR
- Writing to DATA.DATA
- Reading DATA.DATA when smart mode is enabled (CTRLB.SMEN)
- Writing a valid command to CTRLB.CMD

Writing '1' to this bit location will clear the SB flag. The transaction will not continue or be terminated until one of the above actions is performed.

Writing '0' to this bit has no effect.

**Bit 0 – MB** Master on Bus

This flag is set when a byte is transmitted in master write mode. The flag is set regardless of the occurrence of a bus error or an arbitration lost condition. MB is also set when arbitration is lost during sending of NACK in master read mode, or when issuing a start condition if the bus state is unknown. When this flag is set and arbitration is not lost, the master forces the SCL line low, stretching the $I^2C$ clock period. The SCL line will be released and MB will be cleared on one of the following actions:

- Writing to ADDR.ADDR
- Writing to DATA.DATA
- Reading DATA.DATA when smart mode is enabled (CTRLB.SMEN)
- Writing a valid command to CTRLB.CMD

Writing '1' to this bit location will clear the MB flag. The transaction will not continue or be terminated until one of the above actions is performed.

Writing '0' to this bit has no effect.

### 28.10.7 Status

**Name:** STATUS
**Offset:** 0x1A
**Reset:** 0x0000
**Property:** Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | LENERR | SEXTTOUT | MEXTTOUT |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLKHOLD | LOWTOUT | BUSSTATE[1:0] | | | RXNACK | ARBLOST | BUSERR |
| Access | R | R/W | R/W | R/W | | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 10 – LENERR** Transaction Length Error
This bit is set when automatic length is used for a DMA transaction and the slave sends a NACK before ADDR.LEN bytes have been written by the master.

Writing '1' to this bit location will clear STATUS.LENERR. This flag is automatically cleared when writing to the ADDR register.

Writing '0' to this bit has no effect.

This bit is not write-synchronized.

**Bit 9 – SEXTTOUT** Slave SCL Low Extend Time-Out
This bit is set if a slave SCL low extend time-out occurs.

This bit is automatically cleared when writing to the ADDR register.

Writing '1' to this bit location will clear SEXTTOUT. Normal use of the $I^2C$ interface does not require the SEXTTOUT flag to be cleared by this method.

Writing '0' to this bit has no effect.

This bit is not write-synchronized.

**Bit 8 – MEXTTOUT** Master SCL Low Extend Time-Out
This bit is set if a master SCL low time-out occurs.

Writing '1' to this bit location will clear STATUS.MEXTTOUT. This flag is automatically cleared when writing to the ADDR register.

Writing '0' to this bit has no effect.

This bit is not write-synchronized.

**Bit 7 – CLKHOLD** Clock Hold
This bit is set when the master is holding the SCL line low, stretching the $I^2C$ clock. Software should consider this bit when INTFLAG.SB or INTFLAG.MB is set.

This bit is cleared when the corresponding interrupt flag is cleared and the next operation is given.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

This bit is not write-synchronized.

**Bit 6 – LOWTOUT** SCL Low Time-Out
This bit is set if an SCL low time-out occurs.

Writing '1' to this bit location will clear this bit. This flag is automatically cleared when writing to the ADDR register.

Writing '0' to this bit has no effect.

This bit is not write-synchronized.

**Bits 5:4 – BUSSTATE[1:0]** Bus State
These bits indicate the current $I^2C$ bus state.

When in UNKNOWN state, writing 0x1 to BUSSTATE forces the bus state into the IDLE state. The bus state cannot be forced into any other state.

Writing BUSSTATE to idle will set SYNCBUSY.SYSOP.

| Value | Name | Description |
|---|---|---|
| 0x0 | UNKNOWN | The bus state is unknown to the $I^2C$ master and will wait for a stop condition to be detected or wait to be forced into an idle state by software |
| 0x1 | IDLE | The bus state is waiting for a transaction to be initialized |
| 0x2 | OWNER | The $I^2C$ master is the current owner of the bus |
| 0x3 | BUSY | Some other $I^2C$ master owns the bus |

**Bit 2 – RXNACK** Received Not Acknowledge
This bit indicates whether the last address or data packet sent was acknowledged or not.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

This bit is not write-synchronized.

| Value | Description |
|---|---|
| 0 | Slave responded with ACK. |
| 1 | Slave responded with NACK. |

**Bit 1 – ARBLOST** Arbitration Lost
This bit is set if arbitration is lost while transmitting a high data bit or a NACK bit, or while issuing a start or repeated start condition on the bus. The Master on Bus interrupt flag (INTFLAG.MB) will be set when STATUS.ARBLOST is set.

Writing the ADDR.ADDR register will automatically clear STATUS.ARBLOST.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

This bit is not write-synchronized.

**Bit 0 – BUSERR** Bus Error

This bit indicates that an illegal bus condition has occurred on the bus, regardless of bus ownership. An illegal bus condition is detected if a protocol violating start, repeated start or stop is detected on the $I^2C$ bus lines. A start condition directly followed by a stop condition is one example of a protocol violation. If a time-out occurs during a frame, this is also considered a protocol violation, and will set BUSERR.

If the $I^2C$ master is the bus owner at the time a bus error occurs, STATUS.ARBLOST and INTFLAG.MB will be set in addition to BUSERR.

Writing the ADDR.ADDR register will automatically clear the BUSERR flag.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

This bit is not write-synchronized.

### 28.10.8 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x1C
**Reset:** 0x00000000

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SYSOP | ENABLE | SWRST |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SYSOP** System Operation Synchronization Busy
Writing CTRLB.CMD, STATUS.BUSSTATE, ADDR, or DATA when the SERCOM is enabled requires synchronization. When written, the SYNCBUSY.SYSOP bit will be set until synchronization is complete.

Writing CTRLB.CMD or CTRLB.FIFOCLR, STATUS.BUSSTATE, ADDR, or DATA when the SERCOM is enabled requires synchronization. When written, the SYNCBUSY.SYSOP bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | System operation synchronization is not busy. |
| 1 | System operation synchronization is busy. |

**Bit 1 – ENABLE** SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | Enable synchronization is not busy. |
| 1 | Enable synchronization is busy. |

**Bit 0 – SWRST** Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

| Value | Description |
|---|---|
| 0 | SWRST synchronization is not busy. |
| 1 | SWRST synchronization is busy. |

### 28.10.9 Address

**Name:** ADDR
**Offset:** 0x24
**Reset:** 0x0000
**Property:** Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TENBITEN | HS | LENEN | | | ADDR[10:8] | | |
| Access | R/W | R/W | R/W | | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – LEN[7:0]** Transaction Length
These bits define the transaction length of a DMA transaction from 0 to 255 bytes. The Transfer Length Enable (LENEN) bit must be written to '1' in order to use DMA.

**Bit 15 – TENBITEN** Ten Bit Addressing Enable
This bit enables 10-bit addressing. This bit can be written simultaneously with ADDR to indicate a 10-bit or 7-bit address transmission.

| Value | Description |
|---|---|
| 0 | 10-bit addressing disabled. |
| 1 | 10-bit addressing enabled. |

**Bit 14 – HS** High Speed
This bit enables High-speed mode for the current transfer from repeated START to STOP. This bit can be written simultaneously with ADDR for a high speed transfer.

| Value | Description |
|---|---|
| 0 | High-speed transfer disabled. |
| 1 | High-speed transfer enabled. |

**Bit 13 – LENEN** Transfer Length Enable

| Value | Description |
|---|---|
| 0 | Automatic transfer length disabled. |
| 1 | Automatic transfer length enabled. |

**Bits 10:0 – ADDR[10:0]** Address
When ADDR is written, the consecutive operation will depend on the bus state:

UNKNOWN: INTFLAG.MB and STATUS.BUSERR are set, and the operation is terminated.

BUSY: The $I^2C$ master will await further operation until the bus becomes IDLE.

IDLE: The $I^2C$ master will issue a start condition followed by the address written in ADDR. If the address is acknowledged, SCL is forced and held low, and STATUS.CLKHOLD and INTFLAG.MB are set.

OWNER: A repeated start sequence will be performed. If the previous transaction was a read, the acknowledge action is sent before the repeated start bus condition is issued on the bus. Writing ADDR to issue a repeated start is performed while INTFLAG.MB or INTFLAG.SB is set.

STATUS.BUSERR, STATUS.ARBLOST, INTFLAG.MB and INTFLAG.SB will be cleared when ADDR is written.

The ADDR register can be read at any time without interfering with ongoing bus activity, as a read access does not trigger the master logic to perform any bus protocol related operations.

The $I^2C$ master control logic uses bit 0 of ADDR as the bus protocol's read/write flag (R/W); 0 for write and 1 for read.

### 28.10.10 Data

**Name:** DATA
**Offset:** 0x18
**Reset:** 0x0000
**Property:** Write-Synchronized, Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – DATA[7:0]** Data
The master data register I/O location (DATA) provides access to the master transmit and receive data buffers. Reading valid data or writing data to be transmitted can be successfully done only when SCL is held low by the master (STATUS.CLKHOLD is set). An exception is reading the last data byte after the stop condition has been sent.

Accessing DATA.DATA auto-triggers $I^2C$ bus operations. The operation performed depends on the state of CTRLB.ACKACT, CTRLB.SMEN and the type of access (read/write).

Writing or reading DATA.DATA when not in smart mode does not require synchronization.

### 28.10.11 Debug Control

**Name:** DBGCTRL
**Offset:** 0x30
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGSTOP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGSTOP** Debug Stop Mode
This bit controls functionality when the CPU is halted by an external debugger.

| Value | Description |
|---|---|
| 0 | The baud-rate generator continues normal operation when the CPU is halted by an external debugger. |
| 1 | The baud-rate generator is halted when the CPU is halted by an external debugger. |

### 28.10.12 FIFO Space

**Name:** FIFOSPACE
**Offset:** 0x34
**Reset:** 0x0000
**Property:** -

This register allows the user to identify the number of bytes present in each TX and RX FIFO.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | TXSPACE[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 12:8 – RXSPACE[4:0]** RX FIFO Filled Space
These bits return the number filled locations in the RX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

**Bits 4:0 – TXSPACE[4:0]** TX FIFO Empty Space
These bits return the number of available locations in the TX FIFO (bytes or words, depending on CTRLC.DATA32B setting).

### 28.10.13 FIFO CPU Pointers

**Name:** FIFOPTR
**Offset:** 0x36
**Reset:** 0x0000
**Property:** -

This register provides a copy of internal CPU TX and RX FIFO pointers.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPURDPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPUWRPTR[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 11:8 – CPURDPTR[3:0]** RX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. Reading DATA register, will return RXFIFO[CPURDPTR] location value.

**Bits 3:0 – CPUWRPTR[3:0]** TX FIFO Filled Space
These bits return the CPURDPTR pointer value. These bits can be written only if the SERCOM is halted during debugging. When writting to DATA register, the DATA will be written to TXFIFO[CPUWRPTR] location.

# 29. I2S - Inter-IC Sound Controller

## 29.1 Overview

The Inter-IC Sound Controller ($I^2S$) provides bidirectional, synchronous and digital audio link with external audio devices.

This controller is compliant with the Inter-IC Sound ($I^2S$) bus specification. It supports TDM interface with external multi-slot audio codecs. It also supports PDM interface with external MEMS microphones.

The $I^2S$ consists of two Clock Units and two Serializers, that can be enabled separately, to provide Master, Slave, or controller modes, and operate as Receiver or Transmitter.

The pins associated with $I^2S$ peripheral are SDm, FSn, SCKn, and MCKn pins, where n=[0,1] denotes the of clock unit and m=[0,1] is the Serializers instance.

FSn is referred to as Word Select in standard $I^2S$ mode operation and as Frame Sync in TDM mode. Peripheral DMAC channels, separate for each Serializer, allow a continuous high bitrate data transfer without processor intervention to the following:

- Audio codecs in Master, Slave, or Controller mode
- Stereo DAC or ADC through dedicated $I^2S$ serial interface
- Multi-slot or multiple stereo DACs or ADCs, using the TDM format
- Mono or stereo MEMS microphones, using the PDM interface
- 1-channel burst transfer with non-periodic Frame Sync

Each Serializer supports using either a single DMAC channel for all data channels, or two separate DMAC channels for different data channels.

The $I^2S$ supports 8-bit and 16-bit compact stereo format. This helps in reducing the required DMA bandwidth by transferring the left and right samples within the same data word.

Usually, an external audio codec or digital signal processor (DSP) requires a clock which is a multiple of the sampling frequency *fs* (for example, 384×*fs*). The $I^2S$ peripheral in Master Mode and Controller mode is capable of outputting an output clock ranging from 16×*fs* to 1024×*fs* on the Master Clock pin (MCKn).

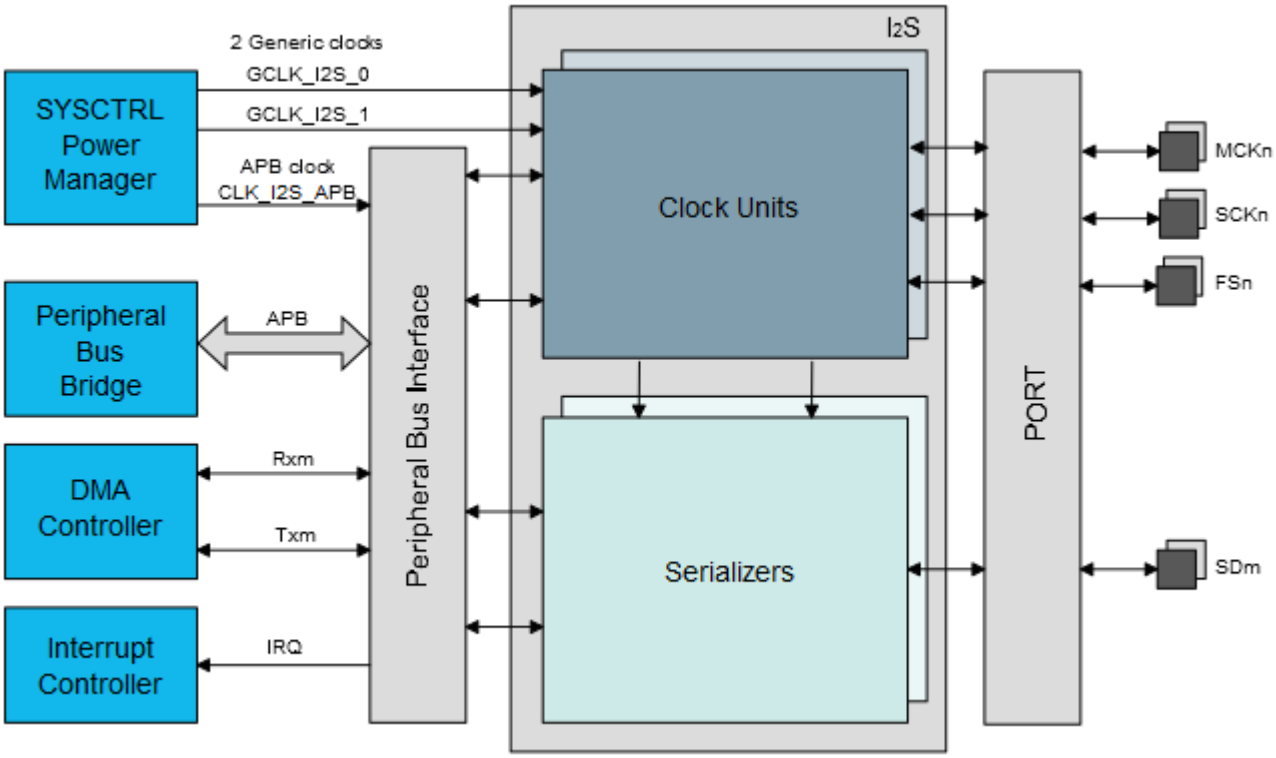
## 29.2 Features

- Compliant with Inter-IC Sound ($I^2S$) bus specification
- 2 independent Serializers configurable as receiver or as transmitter
- Supported data formats:
  - 32-, 24-, 20-, 18-, 16-, and 8-bit mono or stereo format
  - 16- and 8-bit compact stereo format, with left and right samples packed in the same word to reduce data transfers
- Supported data frame formats:
  - 2-channel $I^2S$ with Word Select
  - 1- to 8-slot Time Division Multiplexed (TDM) with Frame Sync and individually enabled slots
  - 1- or 2-channel Pulse Density Modulation (PDM) reception for MEMS microphones
  - 1-channel burst transfer with non-periodic Frame Sync
- 2 independent Clock Units handling either the same clock or separate clocks for the Serializers:

  - Suitable for a wide range of sample frequencies *fs*, including 32kHz, 44.1kHz, 48kHz, 88.2kHz, 96kHz, and 192kHz
  - 16×*fs* to 1024×*fs* Master Clock generated for external audio CODECs
- Master, slave, and controller modes:
  - Master: Data received/transmitted based on internally-generated clocks. Output Serial Clock on SCKn pin, Master Clock on MCKn pin, and Frame Sync Clock on FSn pin
  - Slave: Data received/transmitted based on external clocks on Serial Clock pin (SCKn) or Master Clock pin (MCKn)
  - Controller: Only output internally generated Master clock (MCKn), Serial Clock (SCKn), and Frame Sync Clock (FSn)
- Individual enabling and disabling of Clock Units and Serializers
- DMA interfaces for each Serializer receiver or transmitter to reduce processor overhead:
  - Either one DMA channel for all data slots or
  - One DMA channel per data channel in stereo
- Smart Data Holding register management to avoid data slots mix after overrun or underrun

## 29.3 Block Diagram

**Figure 29-1. I²S Block Diagram**



## 29.4 Signal Description

**Table 29-1. Master Mode**

| Pin Name | Pin Description | Type |
|---|---|---|
| MCKn | Master Clock for Clock Unit n | Input/Output |
| SCKn | Serial Clock for Clock Unit n | Input/Output |
| FSn | $I^2S$ Word Select or TDM Frame Sync for Clock Unit n | Input/Output |
| SDm | Serial Data Input or Output for Serializer m | Input/Output |

**Table 29-2. Slave Mode**

| Pin Name | Pin Description | Type |
|---|---|---|
| MCKn | Master Clock | Input |
| SCKn | Serial Clock for Clock Unit n | Input |
| FSn | $I^2S$ Word Select or TDM Frame Sync | Input |
| SDm | Serial Data Input or Output for Serializer m | Input/Output |

**Table 29-3. Controller Mode**

| Pin Name | Pin Description | Type |
|---|---|---|
| MCKn | Master Clock for Clock Unit n | Output |
| SCKn | Serial Clock for Clock Unit n | Output |
| FSn | $I^2S$ Word Select or TDM Frame Sync | Output |
| SDm | Not Applicable | Not Applicable |

**Note:** One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 29.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

### 29.5.1 I/O Lines

Using the $I^2S$ I/O lines requires the I/O pins to be configured.

The $I^2S$ pins may be multiplexed with I/O Controller lines. The user must first program the I/O Controller to assign the desired $I^2S$ pins to their peripheral function. If the $I^2S$ I/O lines are not used by the application, they can be used for other purposes by the I/O Controller. It is required to enable only the $I^2S$ inputs and outputs actually in use.

**Related Links**

23. PORT - I/O Pin Controller

### 29.5.2 Power Management

The $I^2S$ will continue to operate in any sleep mode where the selected source clocks are running.

### 29.5.3 Clocks

The clock for the $I^2S$ bus interface (CLK_I2S_APB) is generated by the Power Manager. This clock is disabled at reset, and can be enabled in the Power Manager. It is recommended to disable the $I^2S$ before disabling the clock, to avoid freezing the $I^2S$ in an undefined state.

There are two generic clocks, GCLK_I2S_0 and GCLK_I2S_1, connected to the $I^2S$ peripheral, one for each $I^2S$ clock unit. The generic clocks (GCLK_I2S_n, n=0..1) can be set to a wide range of frequencies and clock sources. The GCLK_I2S_n must be enabled and configured before use.

The GCLK_I2S_n clocks must be enabled and configured before triggering Software Reset, so that the logic in all clock domains can be reset.

The generic clocks are only used in Master mode and Controller mode. In Master mode, the clock from a single clock unit can be used for both Serializers to handle synchronous transfers, or a separate clock from different clock units can be used for each Serializer to handle transfers on non-related clocks.

**Related Links**

15. GCLK - Generic Clock Controller

### 29.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). Using the $I^2S$ DMA requests requires the DMA Controller to be configured first.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 29.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using $I^2S$ interrupts requires the interrupt controller to be configured first.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 29.5.6 Events

Not applicable.

### 29.5.7 Debug Operation

When the CPU is halted in Debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging.

### 29.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- DATAm
- INTFLAG
- SYNCBUSY

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 29.5.9 Analog Connections

Not applicable.

## 29.6 Functional Description

### 29.6.1 Principle of Operation

The I²S uses three or four communication lines for synchronous data transfer:

- SDm for receiving or transmitting in Serializer m (m=0..1)
- SCKn for the serial clock in Clock Unit n (n=0..1)
- FSn for the frame synchronization or I²S word select, identifying the beginning of each frame
- Optionally, MCKn to output an oversampling clock to an external codec

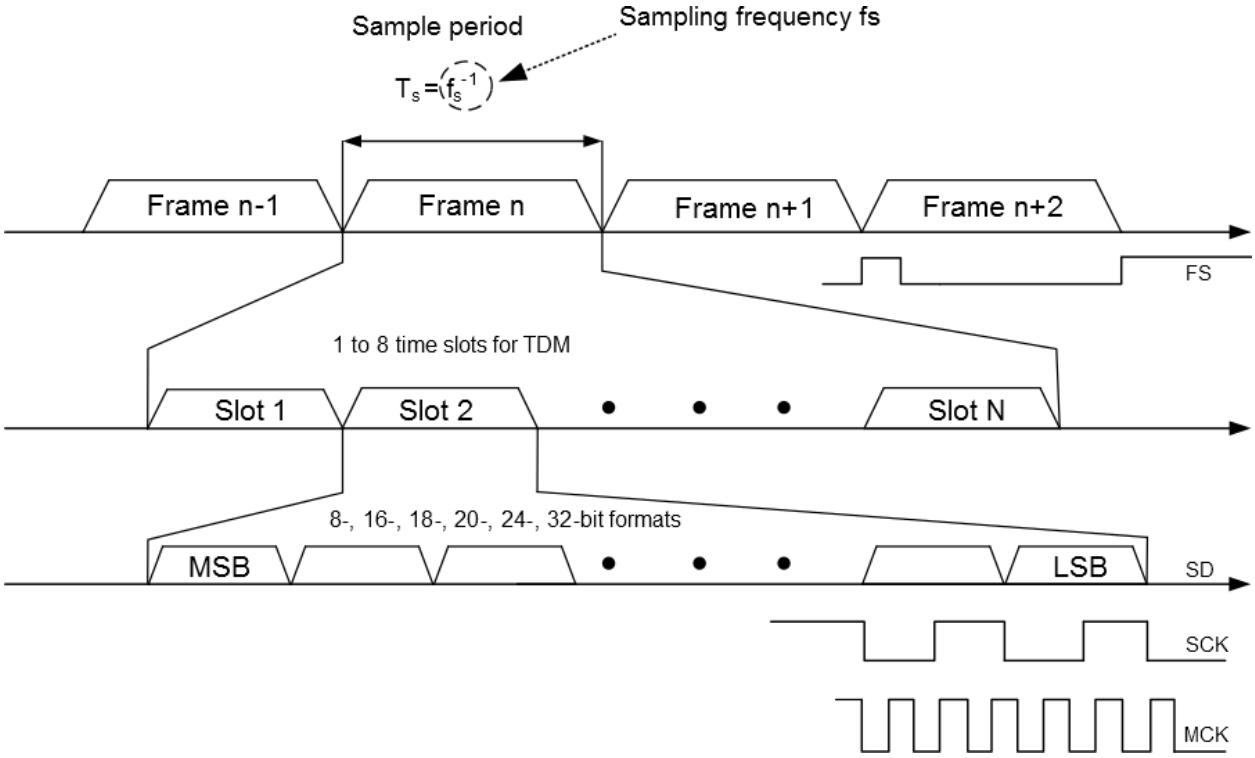
I²S data transfer is frame based, where a serial frame:

- Starts with the frame synchronization active edge, and
- Consists of 1 to 8 data slots, that are 8-, 16-, 24-, or 32-bit wide.

Each data slot is used to transfer one data sample of 8, 16, 18, 20, 24 or 32 bits.

Frame based data transfer is described in the following figure:

**Figure 29-2. Data Format: Frames, Slot, Bits and Clocks**




I²S supports multiple data formats such as:

- 32-, 24-, 20-, 18-, 16-, and 8-bit mono or stereo format
- 16- and 8-bit compact stereo format, with left and right samples packed in the same word to reduce data transfers

In mono format, Transmit mode, data written to the left channel is duplicated to the right output channel. In mono format, Receiver mode, data received from the right channel is ignored and data received from the left channel is duplicated in to the right channel.

In mono format, TDM Transmit mode with more than two slots, data written to the even-numbered slots is duplicated in to the following odd-numbered slot.

In mono format, TDM Receiver mode with more than two slots, data received from the even-numbered slots is duplicated in to the following odd-numbered slot.

Mono format can be enabled by writing a '1' to the MONO bit in the Serializer m Control register (SERCTRLm.MONO).

$I^2S$ support different data frame formats:

- 2-channel $I^2S$ with Word Select
- 1- to 8-slot Time Division Multiplexed (TDM) with Frame Sync and individually enabled slots
- 1- or 2-channel Pulse Density Modulation (PDM) reception for MEMS microphones
- 1-channel burst transfer with non-periodic Frame Sync

In 2 channel $I^2S$ mode, number of slots configured is one or two and successive data words corresponds to left and right channel. Left and right channel are identified by polarity of Word Select signal (FSn signal). Each frame consists of one or two data word(s). In the case of compact stereo format, the number of slots can be one. When 32-bit slot size is used, the number of slots can be two.

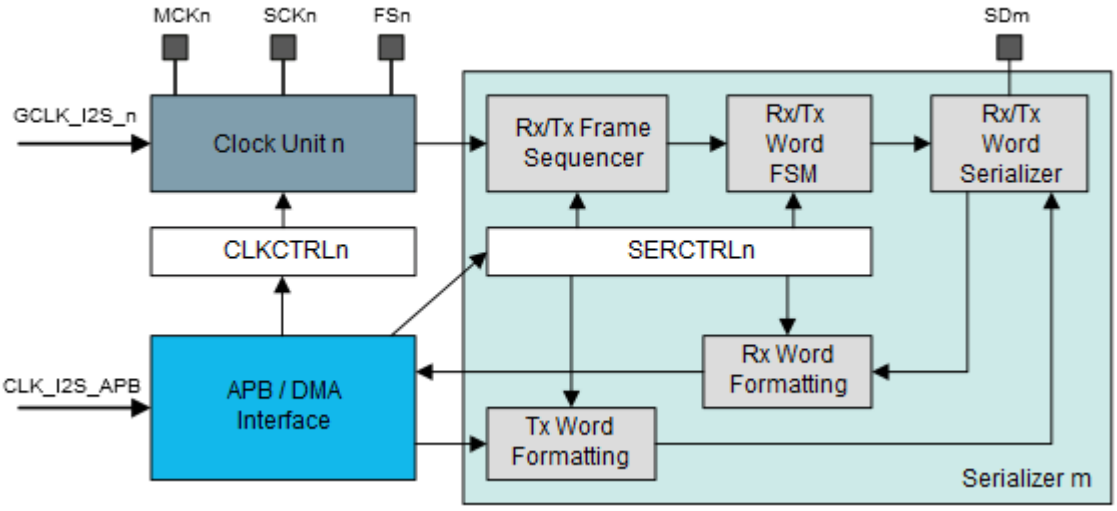
In TDM format, number slots can be configured up to 8 slots. If 4 slots are configured, each frame consists of 4 data words.

In PDM format, continuous 1-bit data samples are available on the SDI line for each SCKn rising and SCKn falling edge as in case of a MEMS microphone with PDM interface.

1-channel burst transfer with non-periodic Frame Sync mode is useful typically for passing control non-auto data as in case of DSP. In Burst mode, a single Data transfer starts at each Frame Sync pulse, and these pulses are 1-bit wide and occur only when a Data transfer is requested.

Sections 29.6.4 I2S Format - Reception and Transmission Sequence with Word Select, 29.6.5 TDM Format - Reception and Transmission Sequence and 29.7 I2S Application Examples describe more about frame/data formats and register settings required for different $I^2S$ applications.

**Figure 29-3. I²S Functional Block Diagram**




### 29.6.1.1 Initialization

The I²S features two Clock Units, and two Serializers configurable as Receiver or Transmitter. The two Serializers can either share the same Clock Unit or use separate Clock Units.

Before enabling the I²S, the following registers must be configured:

- Clock Control registers (CLKCTRLn)
- Serializer Control registers (SERCTRLm)

In Master mode, one of the generic clocks for the I²S must also be configured to operate at the required frequency, as described in 29.6.1 Principle of Operation.

- *fs* is the sampling frequency that defines the frame period
- CLKCTRLn.NBSLOTS defines the number of slots in each frame
- CLKCTRLn.SLOTSIZE defines the number of bits in each slot
- SCKn frequency must be $f_{SCKn}$ = *fs* × number_of_slots × number_of_bits_per_slot)

Once the configuration has been written, the I²S Clock Units and Serializers can be enabled by writing a '1' to the CKENn and SERENm bits and to the ENABLE bit in the Control register (CTRLA). The Clock Unit n can be enabled alone, in Controller Mode, to output clocks to the MCKn, SCKn, and FSn pins. The Clock Units must be enabled if Serializers are enabled.

The Clock Units and the Serializers can be disabled independently by writing a '0' to CTRLA.CKENn or CTRLA.SERENm, respectively. Once requested to stop, they will only stop when the pending transmit frames will be completed, if any. When requested to stop, the ongoing reception of the current slot will be completed and then the Serializer will be stopped.

**Example 29-1. *Example Requirements: fs*=48kHz, MCKn=384×*fs***

If a 384×*fs* MCKn Master Clock is required (i.e. 18.432MHz), the I²S generic clock could run at 18.432MHz with a Master Clock Output Division Factor of 1 (selected by writing CLKCTRLn.MCKOUTDIV=0x0) in order to obtain the desired MCKn frequency.

When using 6 slots per frame (CLKCTRLn.NBSLOTS=0x5) and 32-bit slots (CLKCTRLn.SLOTSIZE=0x3), the desired SCKn frequency is

$f_{SCKn}$ = 48kHz × 6 × 32 = 9.216MHz

This frequency can be achieved by dividing the I²S generic clock output of 18.432MHz by factor 2: Writing CLKCTRLn.MCKDIV=0x1 will select the correct division factor and output the desired SCKn frequency of 9.216MHz to the SCKn pin.

If MCKn is not required, the generic clock could be set to 9.216MHz and CLKCTRLn.MCKDIV=0x0.

### 29.6.2 Basic Operation

The Receiver can be operated by reading the Data Holding register (DATAm), whenever the Receive Ready m bit in the Interrupt Flag Status and Clear register (INTFLAG.RXRDYm) is set. Successive values read from DATAm register will correspond to the samples from the left and right audio channels. In TDM mode, the successive values read from DATAm register correspond to the first slot to the last slot. For instance, if I²S is configured in TDM mode with 4 slots in a frame, then successive values written to DATAm register correspond to first, second, third, and fourth slot. The number of slots in TDM is configured in CLKCTRLn.NBSLOTS.

The Transmitter can be operated by writing to the Data Holding register (DATAm), whenever the Transmit Ready m bit in the Interrupt Flag Status and Clear register (INTFLAG.TXRDYm) is set. Successive values written to DATAm register should correspond to the samples from the left and right audio channels. In TDM mode, the successive values written to DATAm register correspond to the first, second, third, slot to the last slot. The number of slots in TDM is configured in CLKCTRLn.NBSLOTS.

The Receive Ready and Transmit Ready bits can be polled by reading the INTFLAG register.

The processor load can be reduced by enabling interrupt-driven operation. The RXRDYm and/or TXRDYm interrupt requests can be enabled by writing a '1' to the corresponding bit in the Interrupt Enable register (INTENSET). The interrupt service routine associated to the I²S interrupt request will then be executed whenever Receive Ready or Transmit Ready status bits are set.

The processor load can be reduced further by enabling DMA-driven operation. Then, the DMA channels support up to four trigger sources from the I²S peripheral. These four trigger sources in DMAC channel are

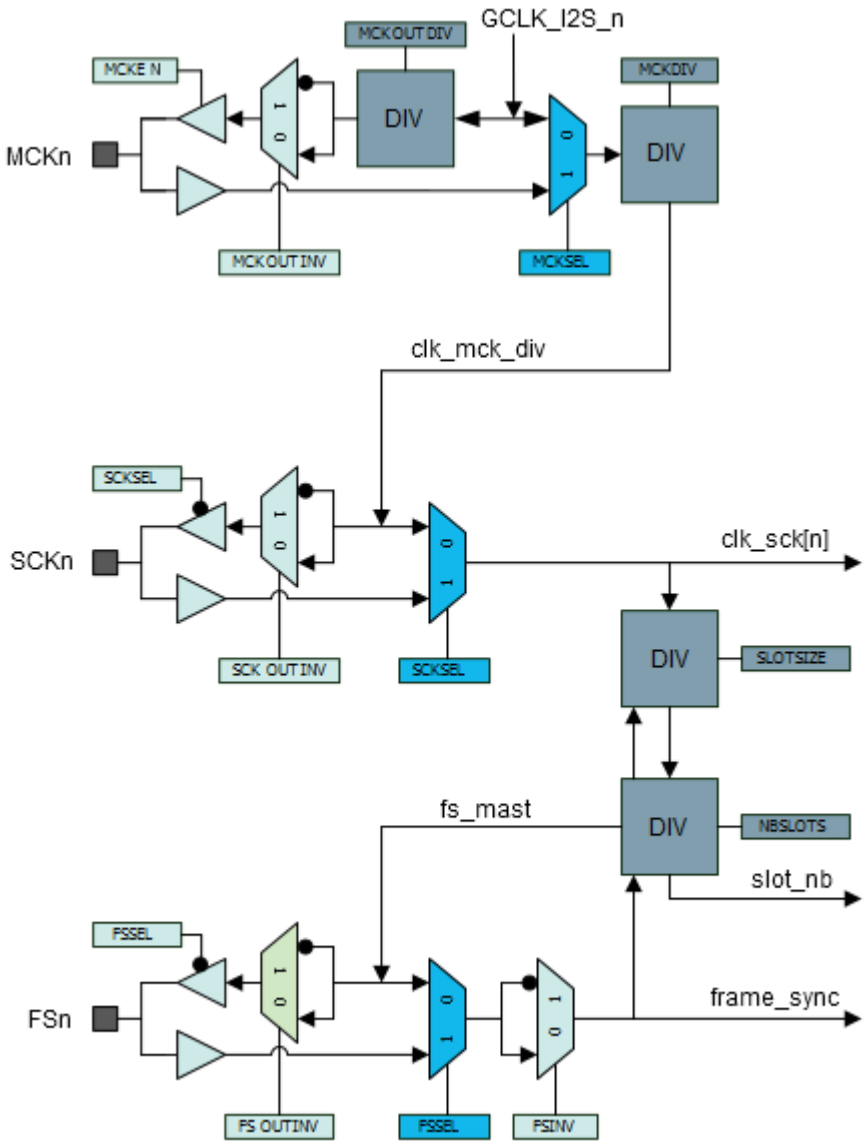
- I2S RX 0,
- I2S RX 1,
- I2S TX 0, and
- I2S TX 1.

For further reference, these are called I2S_DMAC_ID_RX_m and I2S_DMAC_ID_TX_m triggers (m=0..1). By using these trigger sources, one DMA data transfer will be executed whenever the Receive Ready or Transmit Ready status bits are set.

#### 29.6.2.1 Master Clock, Serial Clock, and Frame Sync Generation

The generation of clocks in the I²S is described in the next figure.

**Figure 29-4. I²S Clocks Generation**




#### 29.6.2.1.1 Slave Mode

In Slave mode, the Serial Clock and Frame Sync (Word Select in I²S mode and Frame Sync in TDM mode) are driven by an external master. SCKn and FSn pins are inputs and no generic clock is required by the I²S.

#### 29.6.2.1.2 Master Mode and Controller Mode

In Master Mode, the Master Clock (MCKn), the Serial Clock (SCKn), and the Frame Sync Clock (FSn) are generated by the I²S controller. The user can configure the Master Clock, Serial Clock, and Word Select Frame Sync signal (Word Select in I²S mode and Frame Sync in TDM mode) using the Clock Unit n Control register (CLKCTRLn). MCKn, SCKn, and FSn pins are outputs and a generic clock is used to derive the I²S clocks.

In some applications, audio CODECs connected to the I²S pins may require a Master Clock signal with a frequency multiple of the audio sample frequency *fs*, such as 256×*fs*.

In Controller mode, only the Clock generation unit needs to be configured by writing to the CTRLA and CLKCTRLn registers, where parameters such as clock division factors, Number of slots, Slot size, Frame Sync signal, clock enable are selected.

#### 29.6.2.1.3 MCKn Clock Frequency

When the $I^2S$ is in Master mode, writing a '1' to CLKCTRLn.MCKEN will output GCLK_I2S_n as Master Clock to the MCKn pin. The Master Clock to MCKn pin can be divided by writing to CLKCTRLn.MCKSEL and CLKCTRLn.MCKOUTDIV. The Master Clock (MCKn) frequency is GCLK_I2S_n frequency divided by (MCLKOUTDIV+1).

$$f(\mathrm{MCKn}) = \frac{f(\mathrm{GCLK_I2S_}n)}{(\mathrm{MCKOUTDIV}+1)}$$

#### 29.6.2.1.4 SCKn Clock Frequency

When the Serial Clock (SCKn) is generated from GCLK_I2S_n and both CLKCTRLn.MCKSEL and CLKCTRLn.SCKSEL are zero, the Serial Clock (SCKn) frequency is GCLK_I2S_n frequency divided by (MCKDIV+1).

i.e.

$$f(S\mathrm{CKn}) = \frac{f(\mathrm{GCLK_I2S_}n)}{(\mathrm{MCKDIV}+1)}$$

#### 29.6.2.1.5 Relation Between MCKn, SCKn, and Sampling Frequency *fs*

Based on sampling frequency ***fs***, the SCKn frequency requirement can be calculated:

- SCKn frequency: $f_{\mathrm{SCKn}} = fs \times \mathrm{total_number_of_bits_per_frame}$,
- Where $\mathrm{total_number_of_bits_per_frame} = \mathrm{number_of_slots} \times \mathrm{number_of_bits_per_slots}$.
- The number of slots is selected by writing to the Number of Slots in Frame bit field in the Clock Unit n Control (CLKCTRLn) register: $\mathrm{number_of_slots} = \mathrm{NBSLOTS} + 1$.
- The number of bits per slot (8, 16, 24, or 32 bit) is selected by writing to the Slot Size bit field in CLKCTRLn: .
- Consequently, $f_{\mathrm{SCKn}} = 8 \times fs \times (\mathrm{NBSLOTS} + 1) \times (\mathrm{SLOTSIZE} + 1)$.

The clock frequencies $f_{\mathrm{SCKn}}$ and $f_{\mathrm{MCKn}}$ are derived from the generic clock frequency $f_{\mathrm{GCLK_I2S_n}}$ :

- $f_{\mathrm{GCLK_I2S_n}} = f_{\mathrm{SCKn}} \times (\mathrm{CLKCTRLn.MCKDIV} + 1)$
  $= 8 \times fs \times (\mathrm{NBSLOTS} + 1) \times (\mathrm{SLOTSIZE} + 1) \times (\mathrm{MCKDIV} + 1)$

  , and
- $f_{\mathrm{GCLK_I2S_n}} = f_{\mathrm{MCKn}} \times (\mathrm{MCKOUTDIV} + 1)$.

Substituting the right hand sides of the two last equations yields:

$$f_{\mathrm{MCKn}} = \frac{f_{\mathrm{GCLK_I2S_n}}}{\mathrm{MCKOUTDIV}+1}$$

$$f_{\mathrm{MCKn}} = \frac{8 \cdot (\mathrm{SLOTSIZE}+1) \cdot (\mathrm{NBSLOTS}+1) \cdot (\mathrm{MCKDIV}+1)}{\mathrm{MCKOUTDIV}+1}$$

If a Master Clock output is not required, the GCLK_I2S generic clock can be configured as SCKn by writing a '0'to CLKCTRLn.MCKDIV. Alternatively, if the frequency of the generic clock is a multiple of the required SCKn frequency, the MCKn-to-SCKn divider can be used with the ratio defined by writing the CLKCTRLn.MCKDIV field.

The FSn pin is used as Word Select in I²S format and as Frame Synchronization in TDM format, as described in 29.6.4 I2S Format - Reception and Transmission Sequence with Word Select and 29.6.5 TDM Format - Reception and Transmission Sequence, respectively.

#### 29.6.2.2 Data Holding Registers

For each Serializer m, the I²S user interface includes a Data m register (DATAm). They are used to access data samples for all data slots.

##### 29.6.2.2.1 Data Reception Mode

In receiver mode, the DATAm registers store the received data.

When a new data word is available in the DATAm register, the Receive Ready bit (RXRDYm) in the Interrupt Flag Status and Clear register (INTFLAG) is set. Reading the DATAm register will clear this bit.

A receive overrun condition occurs if a new data word becomes available before the previous data word has been read from the DATAm register. Then, the Receive Overrun bit in INTFLAG will be set (INTFLAG.RXORm). This interrupt can be cleared by writing a '1' to it.

##### 29.6.2.2.2 Data Transmission Mode

In Transmitter mode, the DATAm registers contain the data to be transmitted.

when DATAm is empty, the Transmit Ready bit in the Interrupt Flag Status and Clear register is set (INTFLAG.TXRDYm). Writing to DATAm will clear this bit.

A transmit underrun condition occurs if a new data word needs to be transmitted before it has been written to DATAm. Then, the Transmit Underrun bit in INTFLAG will be set (INTFLAG.TXURm). This interrupt can be cleared by writing a '1' to it. The Transmit Data when Underrun bit in the Serializer n Control register (SERCTRLm.TXSAME) configures whether a zero data word is transmitted in case of underrun (SERCTRLm.TXSAME=0), or the previous data word for the current transmit slot number is transmitted again (SERCTRLm.TXSAME=1).

### 29.6.3 Master, Controller, and Slave Modes

In Master and Controller modes, the I²S provides the Serial Clock, a Word Select/Frame Sync signal and optionally a Master Clock.

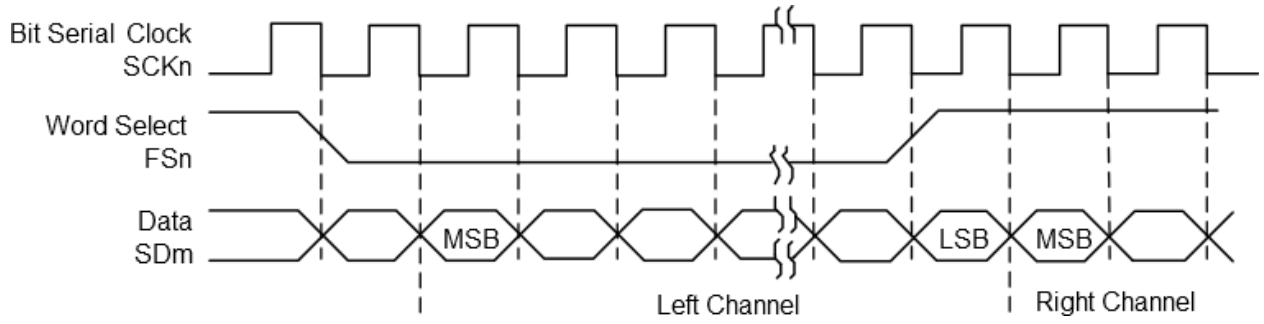
In Controller mode, the I²S Serializers are disabled. Only the clocks are enabled and output for external receivers and/or transmitters.

In Slave mode, the I²S receives the Serial Clock and the Word Select/Frame Sync Signal from an external master. SCKn and FSn pins are inputs.

### 29.6.4 I²S Format - Reception and Transmission Sequence with Word Select

As specified in the I²S protocol, data bits are left-adjusted in the Word Select slot, with the MSB transmitted first, starting one clock period after the transition on the Word Select line.

**Figure 29-5. I²S Reception and Transmission Sequence**



Data bits are sent on the falling edge of the Serial Clock and sampled on the rising edge of the Serial Clock. The Word Select line indicates the channel in transmission, a low level for the left channel and a high level for the right channel.

In $I^2S$ format, typical configurations are described below. These configurations do not list all necessary settings, but only basic ones. Other configuration settings are to be done as per requirement such as clock and DMA configurations.

**Case 1: $I^2S$ 16-bit compact stereo**

- Slot size configured as 16 bits (CLKCTRL0.SLOTSIZE = 0x1)
- Number of slots configured as 2 (CLKCTRL0.NBSLOTS = 0x1)
- Data size configured as 16-bit compact stereo (SERCTRL0.DATASIZE = 0x05)
- Data delay from Frame Sync configured as 1-bit delay (CLKCTRLn.BITDELAY = 0x01)
- Frame Sync Width configured as HALF frame (CLKCTRLn.FSWIDTH = 0x01)

**Case 2: $I^2S$ 24-bit stereo Transmitterwith 24-bit slot**

- Slot size configured as 24 bits (CLKCTRL0.SLOTSIZE = 0x2)
- Number of slots configured as 2 (CLKCTRL0.NBSLOTS = 0x1)
- Data size configured as 24 bits (SERCTRL0.DATASIZE = 0x01)
- Data delay from Frame Sync configured as 1-bit delay (CLKCTRLn.BITDELAY = 0x01)
- Frame Sync Width configured as HALF frame (CLKCTRLn.FSWIDTH = 0x01)

In both cases, it will ensure that Word select signal is 'low level' for the left channel and 'high level' for the right channel.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the Data Word Size bit group in the Serializer Control mregister (SERCTRLm.DATASIZE).

If the slot allows for more data bits than the number of bits specified in the respective DATASIZE field, additional bits are appended to the transmitted or received data word as specified in the SERCTRLm.EXTEND field. If the slot allows less data bits than programmed, the extra bits are not transmitted, or received data word is extended based on the EXTEND field value.

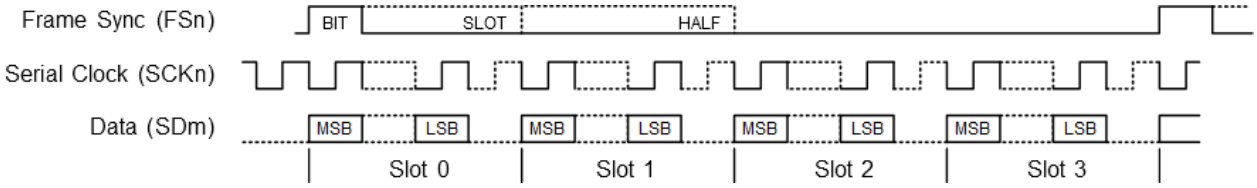
### 29.6.5 TDM Format - Reception and Transmission Sequence

In Time Division Multiplexed (TDM) format, the number of data slots sent or received within each frame will be (CLKCTRLn.NBSLOTS + 1).

By configuring the CLKCTRLn register (CLKCTRLn.FSWIDTH and CLKCTRLn.FSINV), the Frame Sync pulse width and polarity can be modified.

By configuring SERCTRLm, data bits can be left-adjusted or right-adjusted in the slot. It can also configure the data transmission/reception with either the MSB or the LSB transmitted/received first and starting the transmission/reception either at the transition of the FSn pin or one clock period after.

**Figure 29-6. TDM Format Reception and Transmission Sequence**



Data bits are sent on the falling edge of the Serial Clock and sampled on the rising edge of the Serial Clock. The FSn pin provides a frame synchronization signal, at the beginning of slot 0. The delay between the frame start and the first data bit is defined by writing the CLKCTRLn.BITDELAY field.

The Frame Sync pulse can be either one SCKn period (BIT), one slot (SLOT), or one half frame (HALF). This selection is done by writing the CLKCTRLn.FSWIDTH field.

The number of slots is selected by writing the CLKCTRLn.NBSLOTS field.

The number of bits in each slot is selected by writing the CLKCTRLn.SLOTSIZE field.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the DATASIZE field in the Serializer Control register (SERCTRLm).

If the slot allows more data bits than the number of bits specified in the SERCTRLmDATASIZE bit field, additional bits are appended to the transmitted or received data word as specified in the SERCTRLmEXTEND bit field. If the slot allows less data bits than programmed, the extra bits are not transmitted, or received data word is extended based on the EXTEND field value.

### 29.6.6 PDM Reception

In Pulse Density Modulation (PDM) reception mode, continuous 1-bit data samples are available on the SDI line on each SCKn rising edge, e.g. by a MEMS microphone with PDM interface. When using two channel PDM microphones, the second one (right channel) is configured to output data on each SCKn falling edge.

For one PDM microphone, the $I^2S$ controller should be configured in normal Receive mode with one slot and 16- or 32-bit data size, so that 16 or 32 samples of the microphone are stored into each data word.

For two PDM microphones, the $I^2S$ controller should be configured in PDM2 mode with one slot and 32-bit data size. The Serializer will store 16 samples of each microphone in one half of the data word, with left microphone bits in lower half and right microphone bits in upper half, like in compact stereo format.

Based on oversampling frequency requirement from PDM microphone, the SCKn frequency must be configured in the $I^2S$ controller.

> A microphone that requires a sampling frequency of *fs* = 48 kHz and an oversampling frequency of *fo*=64 × *fs* would require an SCKn frequency of 3.072 MHz.

After selecting a proper frequency for GCLK_I2S_n and according Master Clock Division Factor in the Clock Unit n Control register (CLKCTRLn.MCKDIV), SCKn must be selected as per required frequency.

In PDM mode, only the clock and data line (SCKn and SDIn) pins are used.

To configure PDM2 mode, set SLOTSIZE = 0x01 (16-bits), NBSLOTS = 0x00 (1 slots) and SERCTRL0.DATASIZE = 0x00 (32-bit).

### 29.6.7 Data Formatting Unit

To allow more flexibility, data words received by Serializer m will be formatted by the Receive Formatting Unit before being stored into the Data Holding register (RXDATA). The data words written into TXDATA register will be formatted by the Transmit Formatting Unit before transmission by Serializer m .

The formatting options are defined in SERCTRLm:

- SLOTADJ for left or right justification in the slot
- BITREV for bit reversal
- WORDADJ for left or right justification in the data word

- EXTEND for extension to the word size

### 29.6.8 DMA, Interrupts and Events

**Table 29-4. Module Request for I²S**

| Condition | DMA request | DMA request is cleared | Interrupt request | Event input/ output |
|---|---|---|---|---|
| Receive Ready | YES | When data is read | YES | |
| Transmit Ready (Buffer empty) | YES | When data is written | YES | |
| Receive Overrun | | | YES | |
| Transmit Underrun | | | YES | |

#### 29.6.8.1 DMA Operation

Each Serializer can be connected either to one single DMAC channel or to one DMAC channel per data slot in stereo mode. This is selected by writing the SERCTRLm.DMA bit:

**Table 29-5. I²C DMA Request Generation**

| SERCTRLm.DMA | Mode | Slot Parity | DMA Request Trigger |
|---|---|---|---|
| 0 | Receiver | all | I2S_DMAC_ID_RX_m |
| | Transmitter | all | I2S_DMAC_ID_TX_m |
| 1 | Receiver | even | I2S_DMAC_ID_RX_0 |
| | | odd | I2S_DMAC_ID_RX_1 |
| | Transmitter | even | I2S_DMAC_ID_TX_0 |
| | | odd | I2S_DMAC_ID_TX_1 |

The DMAC reads from the DATAm register and writes to the DATAm register for all data slots, successively.

The DMAC transfers may use 32-, 16- or or 8-bit transactions according to the value of the SERCTRLm.DATASIZE field. 8-bit compact stereo uses 16-bit and 16-bit compact stereo uses 32-bit transactions.

#### 29.6.8.2 Interrupts

The I²S has the following interrupt sources:

- Receive Ready (RXRDYm): this is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.
- Receive Overrun (RXORm): this is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.
- Transmit Ready (TXRDYm): this is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.
- Transmit Underrun (TXURm): this is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be

individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the $I^2S$ is reset. See INTFLAG register for details on how to clear interrupt flags. All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. Refer to *Nested Vector Interrupt Controller* for details. The user must read the INTFLAG register to determine which interrupt condition is present.

**Note:** Interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 29.6.8.3 Events

Not applicable.

### 29.6.9 Sleep Mode Operation

The $I^2S$ continues to operate in all sleep modes that still provide its clocks.

### 29.6.10 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the corresponding Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while the corresponding SYNCBUSY bit is '1', a peripheral bus error is generated.

The following bits are synchronized when written:

- Software Reset bit in the Control A register (CTRLA.SWRST). SYNCBUSY.SWRST is set to '1' while synchronization is in progress.
- Enable bit in the Control A register (CTRLA.ENABLE). SYNCBUSY.ENABLE is set to '1' while synchronization is in progress.
- Clock Unit x Enable bits in the Control A register (CTRLA.CKENx). SYNCBUSY.CKENx is set to '1' while synchronization is in progress.
- Serializer x Enable bits in the Control A register (CTRLA.SERENx). SYNCBUSY.SERENx is set to '1' while synchronization is in progress.

The following registers require synchronization when read or written:

- Data n registers (DATAn), Read-Synchronized when Serializer n is in Rx mode or Write-Synchronized when in Tx mode. SYNCBUSY.DATAn is set to '1' while synchronization is in progress.

Synchronization is denoted by the Read-Synchronized or Write-Synchronized property in the register description.

**Related Links**

14.3 Register Synchronization

### 29.6.11 Loop-Back Mode

For debugging purposes, the I²S can be configured to loop back the Transmitter to the Receiver. Writing a '1' to the Loop-Back Test Mode bit in the Serializer m Control register (SERCTRLm.RXLOOP) configures SDm as input and the remaining SD as output. Both SD will be connected internally, so the transmitted data is also received. For instance, writing SERCTRL0.RXLOOP=1 will connect SD1 output to SD0 input, or writing SERCTRL1.RXLOOP=1 will connect SD0 output to SD1 input.

RXLOOP=1 will connect the Transmitter output of the other Serializer to the Receiver input of the current Serializer. For the Loop-back Mode to work, the current Serializer must be configured as receiver and the other Serializer as transmitter.

Writing SERCTRLm.RXLOOP=0 will restore normal behavior and connection between Serializer m and SDm pin input.

As for other changes to the Serializer configuration, Serializer m must be disabled before writing the SERCTRLm register to update SERCTRLm.RXLOOP.

## 29.7 I²S Application Examples

The I²S can support several serial communication modes used in audio or high-speed serial links. Some standard applications are shown in the following figures.

**Note:** The following examples are not a complete list of serial link applications supported by the I²S.

**Figure 29-7. Audio Application Block Diagram**

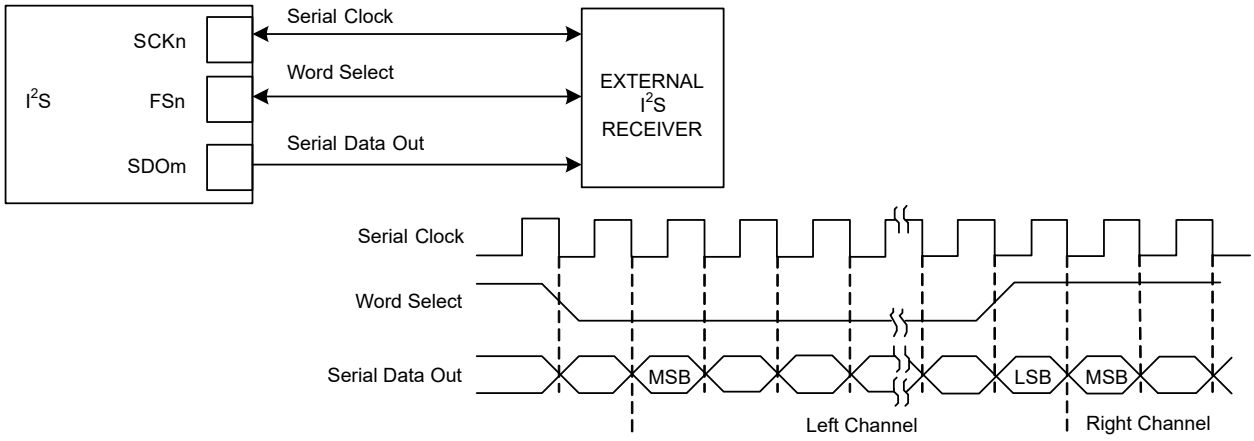


**Figure 29-8. Time Slot Application Block Diagram**

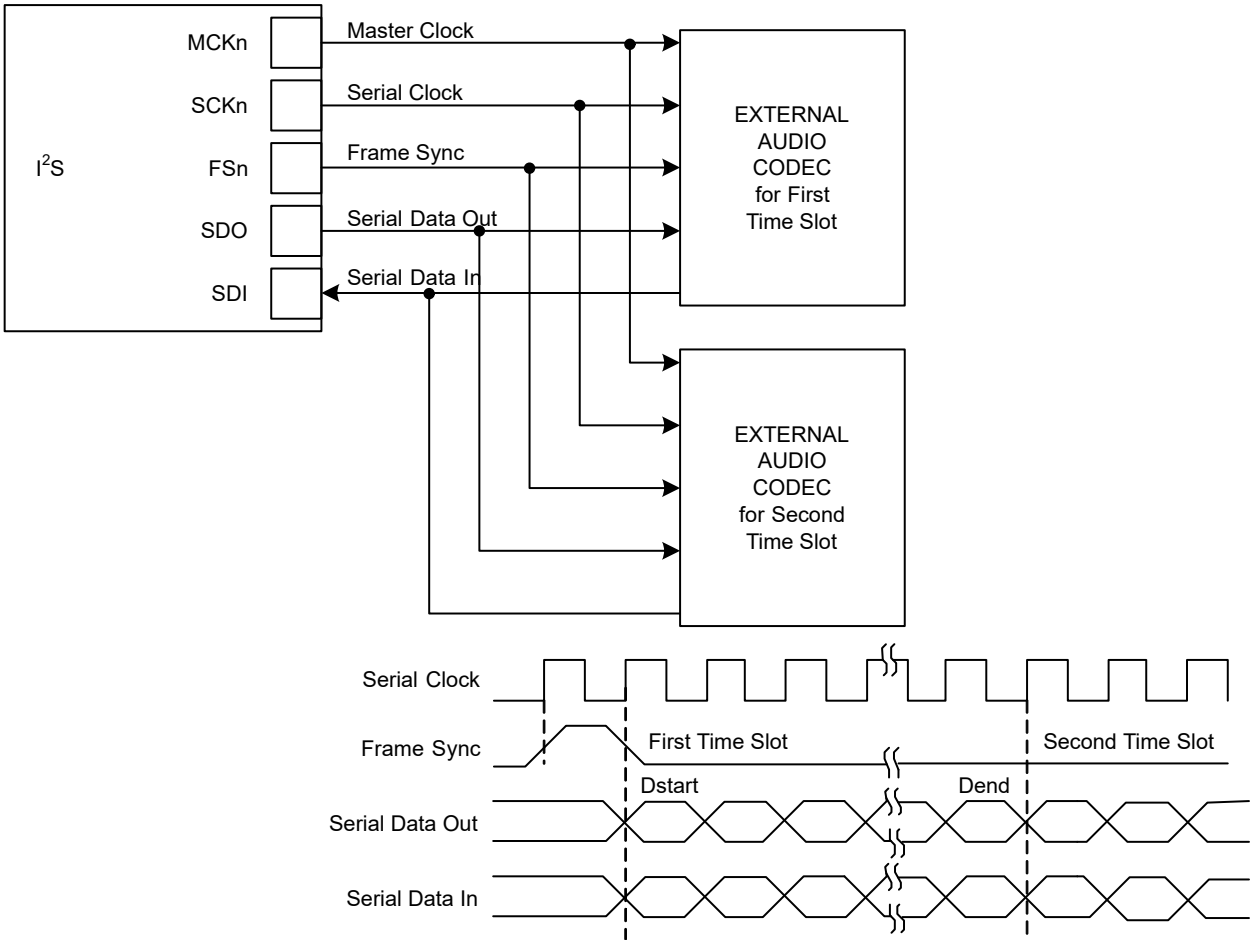



**Figure 29-9. Codec Application Block Diagram**

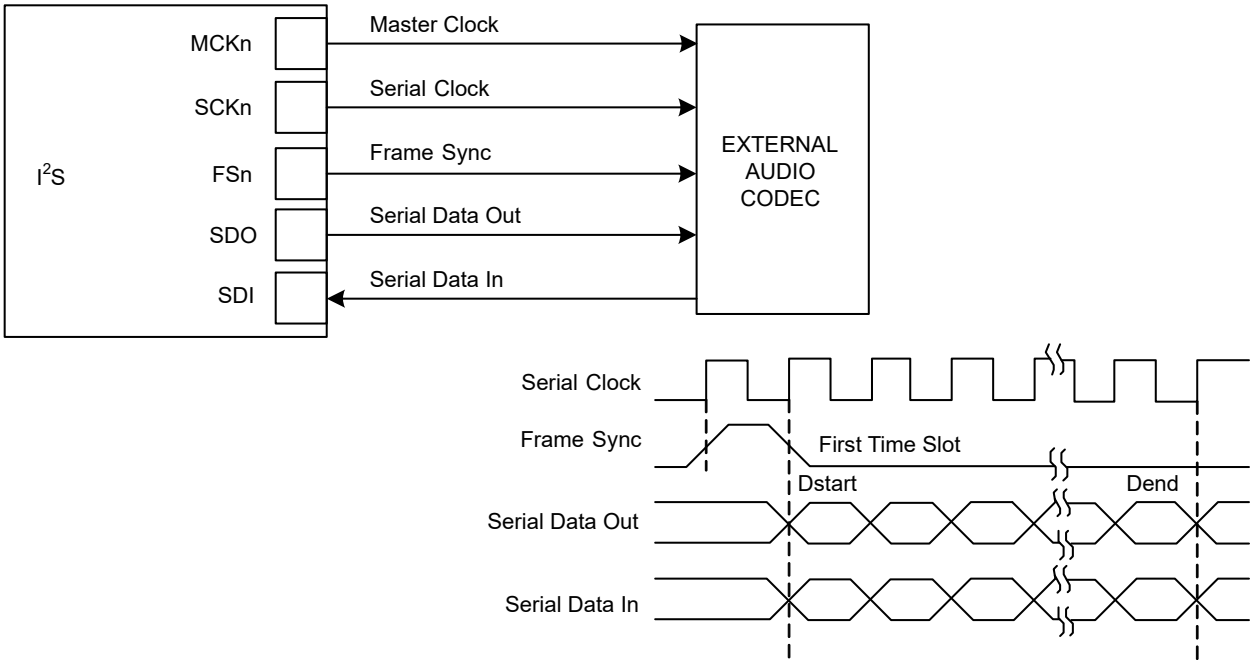
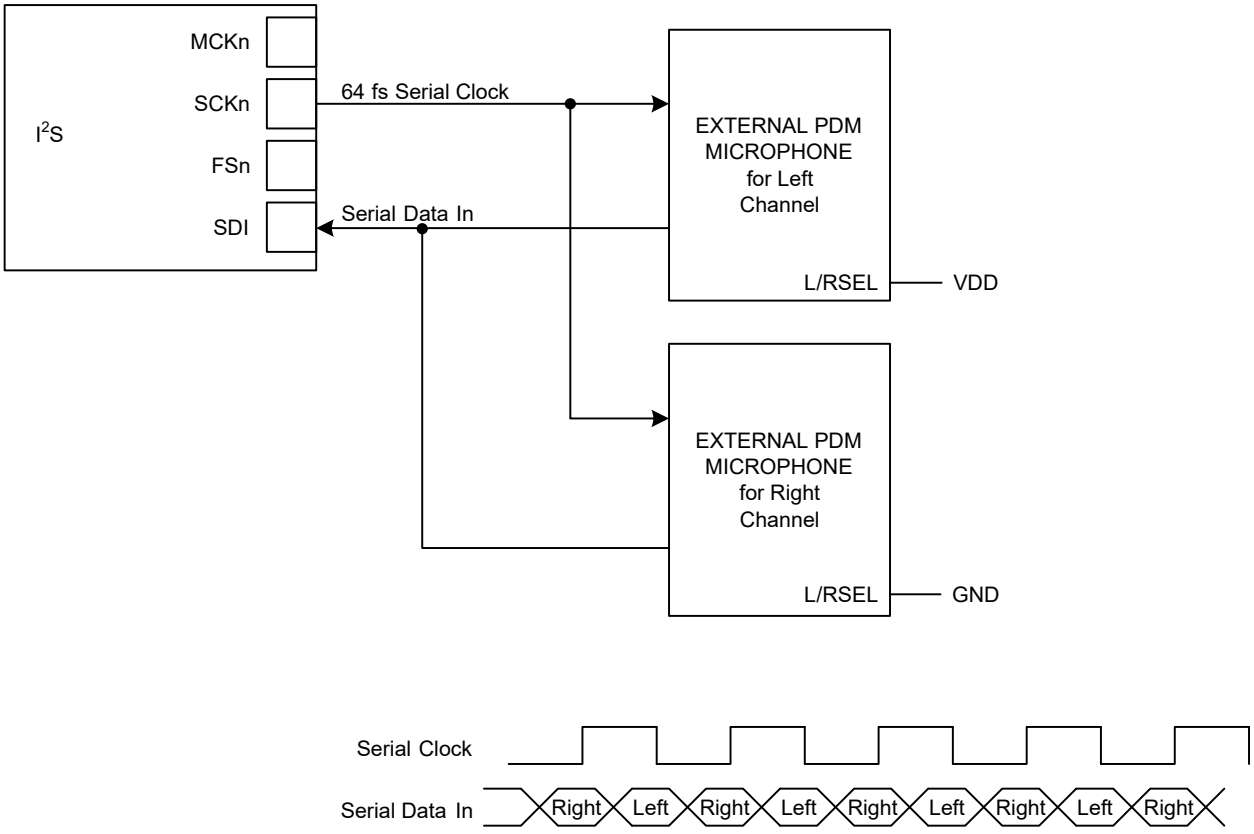


**Figure 29-10. PDM Microphones Application Block Diagram**



## 29.8 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | | SERENx | SERENx | CKENx | CKENx | ENABLE | SWRST |
| 0x01<br>...<br>0x03 | Reserved | | | | | | | | | |
| 0x04 | CLKCTRLn0 | 7:0 | BITDELAY | FSWIDTH[1:0] | | NBSLOTS[2:0] | | | SLOTSIZE[1:0] | |
| | | 15:8 | | | | SCKSEL | FSINV | | | FSSEL |
| | | 23:16 | MCKDIV[4:0] | | | | | MCKEN | | MCKSEL |
| | | 31:24 | MCKOUTINV | SCKOUTINV | FSOUTINV | MCKOUTDIV[4:0] | | | | |
| 0x08 | CLKCTRLn1 | 7:0 | BITDELAY | FSWIDTH[1:0] | | NBSLOTS[2:0] | | | SLOTSIZE[1:0] | |
| | | 15:8 | | | | SCKSEL | FSINV | | | FSSEL |
| | | 23:16 | MCKDIV[4:0] | | | | | MCKEN | | MCKSEL |
| | | 31:24 | MCKOUTINV | SCKOUTINV | FSOUTINV | MCKOUTDIV[4:0] | | | | |
| 0x0C | INTENCLR | 7:0 | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| | | 15:8 | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| 0x0E<br>...<br>0x0F | Reserved | | | | | | | | | |
| 0x10 | INTENSET | 7:0 | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| | | 15:8 | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| 0x12<br>...<br>0x13 | Reserved | | | | | | | | | |
| 0x14 | INTFLAG | 7:0 | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| | | 15:8 | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| 0x16<br>...<br>0x17 | Reserved | | | | | | | | | |
| 0x18 | SYNCBUSY | 7:0 | | | SERENx | SERENx | CKENx | CKENx | ENABLE | SWRST |
| | | 15:8 | | | | | | | DATAx | DATAx |
| 0x1A<br>...<br>0x1F | Reserved | | | | | | | | | |
| 0x20 | SERCTRLn0 | 7:0 | SLOTADJ | | CLKSEL | TXSAME | TXDEFAULT[1:0] | | SERMODE[1:0] | |
| | | 15:8 | BITREV | EXTEND[1:0] | | WORDADJ | | DATASIZE[2:0] | | |
| | | 23:16 | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx |
| | | 31:24 | | | | | | RXLOOP | DMA | MONO |
| 0x24 | SERCTRLn1 | 7:0 | SLOTADJ | | CLKSEL | TXSAME | TXDEFAULT[1:0] | | SERMODE[1:0] | |
| | | 15:8 | BITREV | EXTEND[1:0] | | WORDADJ | | DATASIZE[2:0] | | |
| | | 23:16 | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx |
| | | 31:24 | | | | | | RXLOOP | DMA | MONO |
| 0x28<br>...<br>0x2F | Reserved | | | | | | | | | |
| 0x30 | DATAm0 | 7:0 | DATA[7:0] | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x34 | DATAm1 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x34 | RXDATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |

## 29.9 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

### 29.9.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | SERENx | SERENx | CKENx | CKENx | ENABLE | SWRST |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5,4 – SERENx** Serializer x Enable [x=1..0]
Writing a '0' to this bit will disable the Serializer x.

Writing a '1' to this bit will enable the Serializer x.

| Value | Description |
|---|---|
| 0 | The Serializer x is disabled. |
| 1 | The Serializer x is enabled. |

**Bits 3,2 – CKENx** Clock Unit x Enable [x=1..0]
Writing a '0' to this bit will disable the Clock Unit x.

Writing a '1' to this bit will enable the Clock Unit x.

| Value | Description |
|---|---|
| 0 | The Clock Unit x is disabled. |
| 1 | The Clock Unit x is enabled. |

**Bit 1 – ENABLE** Enable
Writing a '0' to this bit will disable the module.

Writing a '1' to this bit will enable the module.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers to their initial state, and the peripheral will be disabled.

Writing a '1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

The I2S generic clocks must be enabled before triggering Software Reset, so that the logic in all clock domains can be reset.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 29.9.2 Clock Unit n Control

**Name:** CLKCTRLn
**Offset:** 0x04 + n*0x04 [n=0..1]
**Reset:** 0x00000000
**Property:** Enable-Protected, PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MCKOUTINV | SCKOUTINV | FSOUTINV | MCKOUTDIV[4:0] | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MCKDIV[4:0] | | | | | MCKEN | | MCKSEL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | SCKSEL | FSINV | | | FSSEL |
| Access | | | | R/W | R/W | | | R/W |
| Reset | | | | 0 | 0 | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BITDELAY | FSWIDTH[1:0] | | NBSLOTS[2:0] | | | SLOTSIZE[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – MCKOUTINV** Master Clock Output Invert

| Value | Description |
|---|---|
| 0 | The Master Clock n is output without inversion. |
| 1 | The Master Clock n is inverted before being output. |

**Bit 30 – SCKOUTINV** Serial Clock Output Invert

| Value | Description |
|---|---|
| 0 | The Serial Clock n is output without inversion. |
| 1 | The Serial Clock n is inverted before being output. |

**Bit 29 – FSOUTINV** Frame Sync Output Invert

| Value | Description |
|---|---|
| 0 | The Frame Sync n is output without inversion. |
| 1 | The Frame Sync n is inverted before being output. |

**Bits 28:24 – MCKOUTDIV[4:0]** Master Clock Output Division Factor
The generic clock selected by MCKSEL is divided by (MCKOUTDIV + 1) to obtain the Master Clock n output.

**Bits 23:19 – MCKDIV[4:0]** Master Clock Division Factor
The Master Clock n is divided by (MCKDIV + 1) to obtain the Serial Clock n.

**Bit 18 – MCKEN** Master Clock Enable

| Value | Description |
|---|---|
| 0 | The Master Clock n division and output is disabled. |
| 1 | The Master Clock n division and output is enabled. |

**Bit 16 – MCKSEL** Master Clock Select
This field selects the source of the Master Clock n.

| MCKSEL | Name | Description |
|---|---|---|
| 0x0 | GCLK | GCLK_I2S_n is used as Master Clock n source |
| 0x1 | MCKPIN | MCKn input pin is used as Master Clock n source |

**Bit 12 – SCKSEL** Serial Clock Select
This field selects the source of the Serial Clock n.

| SCKSEL | Name | Description |
|---|---|---|
| 0x0 | MCKDIV | Divided Master Clock n is used as Serial Clock n source |
| 0x1 | SCKPIN | SCKn input pin is used as Serial Clock n source |

**Bit 11 – FSINV** Frame Sync Invert

| Value | Description |
|---|---|
| 0 | The Frame Sync n is used without inversion. |
| 1 | The Frame Sync n is inverted before being used. |

**Bit 8 – FSSEL** Frame Sync Select
This field selects the source of the Frame Sync n.

| FSSEL | Name | Description |
|---|---|---|
| 0x0 | SCKDIV | Divided Serial Clock n is used as Frame Sync n source |
| 0x1 | FSPIN | FSn input pin is used as Frame Sync n source |

**Bit 7 – BITDELAY** Data Delay from Frame Sync

| BITDELAY | Name | Description |
|---|---|---|
| 0x0 | LJ | Left Justified (0 Bit Delay) |
| 0x1 | I2S | I2S (1 Bit Delay) |

**Bits 6:5 – FSWIDTH[1:0]** Frame Sync Width
This field selects the duration of the Frame Sync output pulses.

When not in Burst mode, the Clock unit n operates in continuous mode when enabled, with periodic Frame Sync pulses and Data samples.

In Burst mode, a single Data transfer starts at each Frame Sync pulse; these pulses are 1-bit wide and occur only when a Data transfer is requested. Note that the compact stereo modes (16C and 8C) are not supported in the Burst mode.

| FSWIDTH[1:0] | Name | Description |
|---|---|---|
| 0x0 | SLOT | Frame Sync Pulse is 1 Slot wide (default for I2S protocol) |
| 0x1 | HALF | Frame Sync Pulse is half a Frame wide |
| 0x2 | BIT | Frame Sync Pulse is 1 Bit wide |
| 0x3 | BURST | Clock Unit n operates in Burst mode, with a 1-bit wide Frame Sync pulse per Data sample, only when Data transfer is requested |

**Bits 4:2 – NBSLOTS[2:0]** Number of Slots in Frame
Each Frame for Clock Unit n is composed of (NBSLOTS + 1) Slots.

**Bits 1:0 – SLOTSIZE[1:0]** Slot Size
Each Slot for Clock Unit n is composed of a number of bits specified by SLOTSIZE.

| SLOTSIZE[1:0] | Name | Description |
|---|---|---|
| 0x0 | 8 | 8-bit Slot for Clock Unit n |
| 0x1 | 16 | 16-bit Slot for Clock Unit n |
| 0x2 | 24 | 24-bit Slot for Clock Unit n |
| 0x3 | 32 | 32-bit Slot for Clock Unit n |

### 29.9.3 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 13,12 – TXURx** Transmit Underrun x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transmit Underrun x Interrupt Enable bit, which disables the Transmit Underrun x interrupt.

| Value | Description |
|---|---|
| 0 | The Transmit Underrun x interrupt is disabled. |
| 1 | The Transmit Underrun x interrupt is enabled. |

**Bits 9,8 – TXRDYx** Transmit Ready x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transmit Ready x Interrupt Enable bit, which disables the Transmit Ready x interrupt.

| Value | Description |
|---|---|
| 0 | The Transmit Ready x interrupt is disabled. |
| 1 | The Transmit Ready x interrupt is enabled. |

**Bits 4,5 – RXORx** Receive Overrun x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Receive Overrun x Interrupt Enable bit, which disables the Receive Overrun x interrupt.

| Value | Description |
|---|---|
| 0 | The Receive Overrun x interrupt is disabled. |
| 1 | The Receive Overrun x interrupt is enabled. |

**Bits 1,0 – RXRDYx** Receive Ready x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Receive Ready x Interrupt Enable bit, which disables the Receive Ready x interrupt.

| Value | Description |
|---|---|
| 0 | The Receive Ready x interrupt is disabled. |
| 1 | The Receive Ready x interrupt is enabled. |

### 29.9.4 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x10
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 13,12 – TXURx** Transmit Underrun x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Transmit Underrun Interrupt Enable bit, which enables the Transmit Underrun interrupt.

| Value | Description |
|---|---|
| 0 | The Transmit Underrun interrupt is disabled. |
| 1 | The Transmit Underrun interrupt is enabled. |

**Bits 9,8 – TXRDYx** Transmit Ready x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Transmit Ready Interrupt Enable bit, which enables the Transmit Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Transmit Ready interrupt is disabled. |
| 1 | The Transmit Ready interrupt is enabled. |

**Bits 4,5 – RXORx** Receive Overrun x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Receive Overrun Interrupt Enable bit, which enables the Receive Overrun interrupt.

| Value | Description |
|---|---|
| 0 | The Receive Overrun interrupt is disabled. |
| 1 | The Receive Overrun interrupt is enabled. |

**Bits 1,0 – RXRDYx** Receive Ready x Interrupt Enable [x=1..0]
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Receive Ready Interrupt Enable bit, which enables the Receive Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Receive Ready interrupt is disabled. |
| 1 | The Receive Ready interrupt is enabled. |

### 29.9.5 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x14
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURx | TXURx | | | TXRDYx | TXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RXORx | RXORx | | | RXRDYx | RXRDYx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 13,12 – TXURx** Transmit Underrun x [x=1..0]
This flag is cleared by writing a '1' to it.

This flag is set when a Transmit Underrun condition occurs in Sequencer x, and will generate an interrupt request if INTENCLR/SET.TXURx is set to '1'.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transmit Underrun x interrupt flag.

**Bits 9,8 – TXRDYx** Transmit Ready x [x=1..0]
This flag is cleared by writing to DATAx register or writing a '1' to it.

This flag is set when Sequencer x is ready to accept a new data word to be transmitted, and will generate an interrupt request if INTENCLR/SET.TXRDYx is set to '1'.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Transmit Ready x interrupt flag.

**Bits 4,5 – RXORx** Receive Overrun x [x=1..0]
This flag is cleared by writing a '1' to it.

This flag is set when a Receive Overrun condition occurs in Sequencer x, and will generate an interrupt request if INTENCLR/SET.RXORx is set to '1'.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Receive Overrun x interrupt flag.

**Bits 1,0 – RXRDYx** Receive Ready x [x=1..0]
This flag is cleared by reading from DATAx register or writing a '1' to it.

This flag is set when a Sequencer x has received a new data word, and will generate an interrupt request if INTENCLR/SET.RXRDYx is set to '1'.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Receive Ready x interrupt flag.

### 29.9.6 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x18
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DATAx | DATAx |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | SERENx | SERENx | CKENx | CKENx | ENABLE | SWRST |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8,9 – DATAx** Data x Synchronization Status [x=1..0]
Bit DATAx is cleared when the synchronization of DATA Holding register (DATAx) between the clock domains is complete.

Bit DATAx is set when the synchronization of DATA Holding register (DATAx) between the clock domains is started.

**Bits 4,5 – SERENx** Serializer x Enable Synchronization Status [x=1..0]
Bit SERENx is cleared when the synchronization of CTRLA.SERENx bit between the clock domains is complete.

Bit SERENx is set when the synchronization of CTRLA.SERENx bit between the clock domains is started.

**Bits 2,3 – CKENx** Clock Unit x Enable Synchronization Status [x=1..0]
Bit CKENx is cleared when the synchronization of CTRLA.CKENx bit between the clock domains is complete.

Bit CKENx is set when the synchronization of CTRLA.CKENx bit between the clock domains is started.

**Bit 1 – ENABLE** Enable Synchronization Status
This bit is cleared when the synchronization of CTRLA.ENABLE bit between the clock domains is complete.

This bit is set when the synchronization of CTRLA.ENABLE bit between the clock domains is started.

**Bit 0 – SWRST** Software Reset Synchronization Status
This bit is cleared when the synchronization of CTRLA.SWRST bit between the clock domains is complete.

This bit is set when the synchronization of CTRLA.SWRST bit between the clock domains is started.

### 29.9.7 Serializer n Control

**Name:** SERCTRLn
**Offset:** 0x20 + n*0x04 [n=0..1]
**Reset:** 0x00000000
**Property:** Enable-Protected, PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RXLOOP | DMA | MONO |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx | SLOTDISx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BITREV | EXTEND[1:0] | | WORDADJ | | DATASIZE[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLOTADJ | | CLKSEL | TXSAME | TXDEFAULT[1:0] | | SERMODE[1:0] | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 26 – RXLOOP** Loop-back Test Mode
This bit enables a loop-back test mode:

| Value | Description |
|---|---|
| 0 | Each Receiver uses its SDn pin as input (default mode). |
| 1 | Receiver uses as input the transmitter output of the other Serializer in the pair: e.g. SD1 for SD0 or SD0 for SD1. |

**Bit 25 – DMA** Single or Multiple DMA Channels
This bit selects whether even- and odd-numbered slots use separate DMA channels or the same DMA channel.

| DMA | Name | Description |
|---|---|---|
| 0x0 | SINGLE | Single DMA channel |
| 0x1 | MULTIPLE | One DMA channel per data channel |

**Bit 24 – MONO** Mono Mode.

| MONO | Name | Description |
|---|---|---|
| 0x0 | STEREO | Normal mode |
| 0x1 | MONO | Left channel data is duplicated to right channel |

**Bits 23,22,21,20,18,19,18,17,16 – SLOTDISx** Slot x Disabled for this Serializer [x=7..0]
This field allows disabling some slots in each transmit frame:

| Value | Description |
|---|---|
| 0 | Slot x is used for data transfer. |
| 1 | Slot x is not used for data transfer and will be output as specified in the TXDEFAULT field. |

**Bit 15 – BITREV** Data Formatting Bit Reverse
This bit allows changing the order of data bits in the word in the Formatting Unit.

| BITREV | Name | Description |
|---|---|---|
| 0x0 | MSBIT | Transfer Data Most Significant Bit (MSB) first (default for I2S protocol) |
| 0x1 | LSBIT | Transfer Data Least Significant Bit (LSB) first |

**Bits 14:13 – EXTEND[1:0]** Data Formatting Bit Extension
This field defines the bit value used to extend data samples in the Formatting Unit.

| EXTEND[1:0] | Name | Description |
|---|---|---|
| 0x0 | ZERO | Extend with zeros |
| 0x1 | ONE | Extend with ones |
| 0x2 | MSBIT | Extend with Most Significant Bit |
| 0x3 | LSBIT | Extend with Least Significant Bit |

**Bit 12 – WORDADJ** Data Word Formatting Adjust
This field defines left or right adjustment of data samples in the word in the Formatting Unit. for details.

| WORDADJ | Name | Description |
|---|---|---|
| 0x0 | RIGHT | Data is right adjusted in word |
| 0x1 | LEFT | Data is left adjusted in word |

**Bits 10:8 – DATASIZE[2:0]** Data Word Size
This field defines the number of bits in each data sample. For 8-bit compact stereo, two 8-bit data samples are packed in bits 15 to 0 of the DATAm register. For 16-bit compact stereo, two 16-bit data samples are packed in bits 31 to 0 of the DATAm register.

| DATASIZE[2:0] | Name | Description |
|---|---|---|
| 0x0 | 32 | 32 bits |
| 0x1 | 24 | 24 bits |

| DATASIZE[2:0] | Name | Description |
|---|---|---|
| 0x2 | 20 | 20 bits |
| 0x3 | 18 | 18 bits |
| 0x4 | 16 | 16 bits |
| 0x5 | 16C | 16 bits compact stereo |
| 0x6 | 8 | 8 bits |
| 0x7 | 8C | 8 bits compact stereo |

**Bit 7 – SLOTADJ** Data Slot Formatting Adjust
This field defines left or right adjustment of data samples in the slot.

| SLOTADJ | Name | Description |
|---|---|---|
| 0x0 | RIGHT | Data is right adjusted in slot |
| 0x1 | LEFT | Data is left adjusted in slot |

**Bit 5 – CLKSEL** Clock Unit Selection.

| CLKSEL | Name | Description |
|---|---|---|
| 0x0 | CLK0 | Use Clock Unit 0 |
| 0x1 | CLK1 | Use Clock Unit 1 |

**Bit 4 – TXSAME** Transmit Data when Underrun.

| TXSAME | Name | Description |
|---|---|---|
| 0x0 | ZERO | Zero data transmitted in case of underrun |
| 0x1 | SAME | Last data transmitted in case of underrun |

**Bits 3:2 – TXDEFAULT[1:0]** Line Default Line when Slot Disabled
This field defines the default value driven on the SDn output pin during all disabled Slots.

| TXDEFAULT[1:0] | Name | Description |
|---|---|---|
| 0x0 | ZERO | Output Default Value is 0 |
| 0x1 | ONE | Output Default Value is 1 |
| 0x2 | | Reserved |
| 0x3 | HIZ | Output Default Value is high impedance |

**Bits 1:0 – SERMODE[1:0]** Serializer Mode.

| SERMODE[1:0] | Name | Description |
|---|---|---|
| 0x0 | RX | Receive |
| 0x1 | TX | Transmit |

| SERMODE[1:0] | Name | Description |
|---|---|---|
| 0x2 | PDM2 | Receive one PDM data on each serial clock edge |
| 0x3 | | Reserved |

### 29.9.8 Data Holding m

**Name:** DATAm
**Offset:** 0x30 + m*0x04 [m=0..1]
**Reset:** 0x00000000
**Property:** Read-Synchronized, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Sample Data
This register is used to transfer data to or from Serializer n.

Data samples written to DATAn register will be sent to Serializer n for transmission, through the Transmit Formatting Unit that will apply the formatting specified in the SERCTRLn register.

Data samples received by Serializer n will be available for reading from DATAn register, through the Receive Formatting Unit, according to formatting information for Serializer n in the SERCTRLn register.

### 29.9.9 Rx Data

**Name:** RXDATA
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Sample Data
This register is used to transfer data from the Rx Serializer.

Data samples received by Rx Serializer will be available for reading from RXDATA register, through the Receive Formatting Unit, according to formatting information for Rx Serializer in the RXCTRL register.

# 30. TC – Timer/Counter

## 30.1 Overview

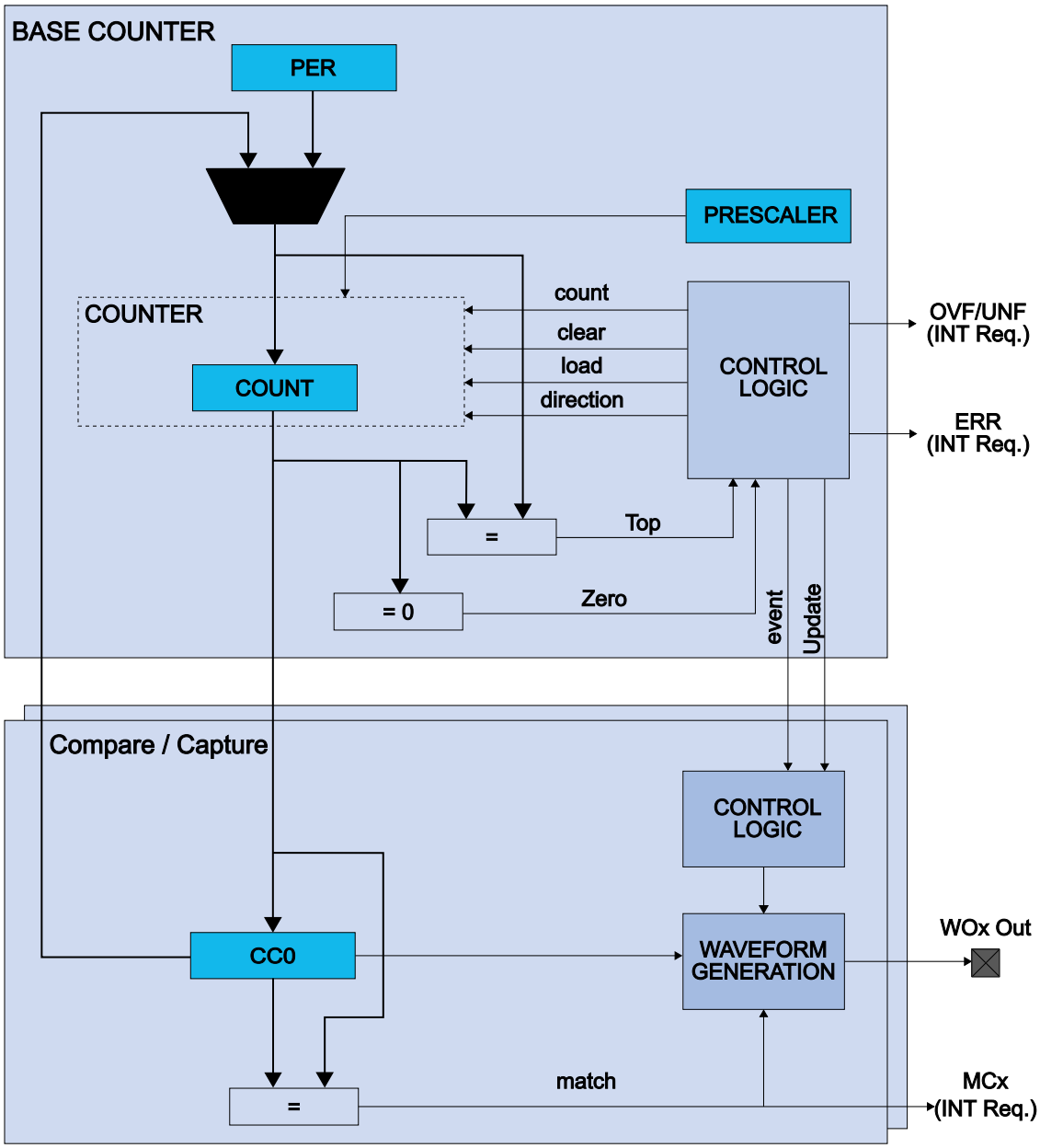
The TC consists of a counter, a prescaler, compare/capture channels and control logic. The counter can be set to count events, or it can be configured to count clock pulses. The counter, together with the compare/capture channels, can be configured to timestamp input events, allowing capture of frequency and pulse width. It can also perform waveform generation, such as frequency generation and pulse-width modulation (PWM).

## 30.2 Features

- Selectable configuration
  - Up to five 16-bit Timer/Counters (TC), each configurable as:
    - 8-bit TC with two compare/capture channels
    - 16-bit TC with two compare/capture channels
    - 32-bit TC with two compare/capture channels, by using two TCs
- Waveform generation
  - Frequency generation
  - Single-slope pulse-width modulation
- Input capture
  - Event capture
  - Frequency capture
  - Pulse-width capture
- One input event
- Interrupts/output events on:
  - Counter overflow/underflow
  - Compare match or capture
- Internal prescaler
- Can be used with DMA and to trigger DMA transactions

## 30.3 Block Diagram

**Figure 30-1. Timer/Counter Block Diagram**




## 30.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| WO[1:0] | Digital output | Waveform output |

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 30.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 30.5.1 I/O Lines

In order to use the I/O lines of this peripheral, the I/O pins must be configured using the I/O Pin Controller (PORT).

**Related Links**

23. PORT - I/O Pin Controller

### 30.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 30.5.3 Clocks

The TC bus clock (CLK_TCx_APB, where x represents the specific TC instance number) can be enabled and disabled in the Power Manager, and the default state of CLK_TCx_APB can be found in the *Peripheral Clock Masking* section in “PM – Power Manager”.

The different TC instances are paired, even and odd, starting from TC3, and use the same generic clock, GCLK_TCx. This means that the TC instances in a TC pair cannot be set up to use different GCLK_TCx clocks.

This generic clock is asynchronous to the user interface clock (CLK_TCx_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to 30.6.6 Synchronization for further details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

### 30.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 30.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 30.5.6 Events

The events of this peripheral are connected to the Event System.

**Related Links**

24. EVSYS – Event System

### 30.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will halt normal operation. This peripheral can be forced to continue operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

### 30.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag register (INTFLAG)
- Status register (STATUS)
- Read Request register (READREQ)
- Count register (COUNT)
- Period register (PER)
- Compare/Capture Value registers (CCx)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 30.5.9 Analog Connections

Not applicable.

## 30.6 Functional Description

### 30.6.1 Principle of Operation

The following definitions are used throughout the documentation:

**Table 30-1. Timer/Counter Definitions**

| Name | Description |
|---|---|
| TOP | The counter reaches TOP when it becomes equal to the highest value in the count sequence. The TOP value can be the same as Period (PER) or the Compare Channel 0 (CC0) register value depending on the waveform generator mode in Waveform Output Operations. |
| ZERO | The counter is ZERO when it contains all zeroes |
| MAX | The counter reaches MAX when it contains all ones |
| UPDATE | The timer/counter signals an update when it reaches ZERO or TOP, depending on the direction settings. |

| Name | Description |
|---|---|
| Timer | The timer/counter clock control is handled by an internal source |
| Counter | The clock control is handled externally (e.g. counting external events) |
| CC | For compare operations, the CC are referred to as “compare channels”<br>For capture operations, the CC are referred to as “capture channels.” |

The counter in the TC can either count events from the Event System, or clock ticks of the GCLK_TCx clock, which may be divided by the prescaler.

The counter value is passed to the CCx where it can be either compared to user-defined values or captured.

The compare and capture registers (CCx) and counter register (COUNT) can be configured as 8-, 16- or 32-bit registers, with according MAX values. Mode settings determine the maximum range of the counter.

In 8-bit mode, Period Value (PER) is also available. The counter range and the operating frequency determine the maximum time resolution achievable with the TC peripheral.

The TC can be set to count up or down. Under normal operation, the counter value is continuously compared to the TOP or ZERO value to determine whether the counter has reached that value. On a comparison match the TC can request DMA transactions, or generate interrupts or events for the Event System. On a comparison match the TC can request DMA transactions, or generate interrupts or events for the Event System.

In compare operation, the counter value is continuously compared to the values in the CCx registers. In case of a match the TC can request DMA transactions, or generate interrupts or events for the Event System. In waveform generator mode, these comparisons are used to set the waveform period or pulse width.

Capture operation can be enabled to perform input signal period and pulse width measurements, or to capture selectable edges from an internal event from Event System.

### 30.6.2 Basic Operation

#### 30.6.2.1 Initialization

The following registers are enable-protected, meaning that they can only be written when the TC is disabled (CTRLA.ENABLE =0):

- Control A register (CTRLA), except the Run Standby (RUNSTDBY), Enable (ENABLE) and Software Reset (SWRST) bits

Enable-protected bits in the CTRLA register can be written at the same time as CTRLA.ENABLE is written to '1', but not at the same time as CTRLA.ENABLE is written to '0'. Enable-protection is denoted by the "Enable-Protected" property in the register description. The following bits are enable-protected:

- Event Action bits in the Event Control register (EVCTRL.EVACT)

Before enabling the TC, the peripheral must be configured by the following steps:

1. Enable the TC bus clock (CLK_TCx_APB).
2. Select 8-, 16- or 32-bit counter mode via the TC Mode bit group in the Control A register (CTRLA.MODE). The default mode is 16-bit.
3. Select one wave generation operation in the Waveform Generation Operation bit group in the Control A register (CTRLA.WAVEGEN).

4. If desired, the GCLK_TCx clock can be prescaled via the Prescaler bit group in the Control A register (CTRLA.PRESCALER).
   – If the prescaler is used, select a prescaler synchronization operation via the Prescaler and Counter Synchronization bit group in the Control A register (CTRLA.PRESYNC).
5. Select one-shot operation by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT).
6. If desired, configure the counting direction 'down' (starting from the TOP value) by writing a '1' to the Counter Direction bit in the Control B register (CTRLBSET.DIR).
7. For capture operation, enable the individual channels to capture in the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTEN).
8. If desired, enable inversion of the waveform output or IO pin input signal for individual channels via the Waveform Output Invert Enable bit group in the Control C register (CTRLC.INVEN).

#### 30.6.2.2 Enabling, Disabling and Resetting

The TC is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The TC is disbled by writing a zero to CTRLA.ENABLE.

The TC is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the TC, except DBGCTRL, will be reset to their initial state, and the TC will be disabled. Refer to the CTRLA register for details.

The TC should be disabled before the TC is reset in order to avoid undefined behavior.

#### 30.6.2.3 Prescaler Selection

The GCLK_TCx is fed into the internal prescaler.

The prescaler consists of a counter that counts up to the selected prescaler value, whereupon the output of the prescaler toggles.

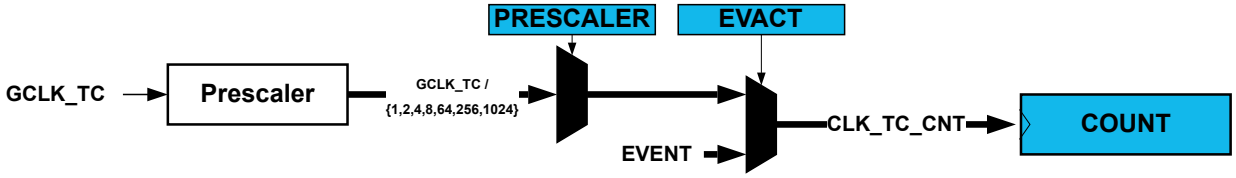
If the prescaler value is higher than one, the counter update condition can be optionally executed on the next GCLK_TCx clock pulse or the next prescaled clock pulse. For further details, refer to Prescaler (CTRLA.PRESCALER) and Counter Synchronization (CTRLA.PRESYNC) description.

Prescaler outputs from 1 to 1/1024 are available. For a complete list of available prescaler outputs, see the register description for the Prescaler bit group in the Control A register (CTRLA.PRESCALER).

**Note:** When counting events, the prescaler is bypassed.

The joint stream of prescaler ticks and event action ticks is called CLK_TC_CNT.

**Figure 30-2. Prescaler**




#### 30.6.2.4 Counter Mode

The counter mode is selected by the Mode bit group in the Control A register (CTRLA.MODE). By default, the counter is enabled in the 16-bit counter resolution. Three counter resolutions are available:

- COUNT8: The 8-bit TC has its own Period register (PER). This register is used to store the period value that can be used as the top value for waveform generation.
- COUNT16: 16-bit is the default counter mode. There is no dedicated period register in this mode.

- COUNT32: This mode is achieved by pairing two 16-bit TC peripherals. TC3 is paired with TC4, and TC5 is paired with TC6. TC7 does not support 32-bit resolution.

When paired, the TC peripherals are configured using the registers of the even-numbered TC (TC4 or TC6 respectively). The odd-numbered partner (TC3 or TC5 respectively) will act as slave, and the Slave bit in the Status register (STATUS.SLAVE) will be set. The register values of a slave will not reflect the registers of the 32-bit counter. Writing to any of the slave registers will not affect the 32-bit counter. Normal access to the slave COUNT and CCx registers is not allowed.

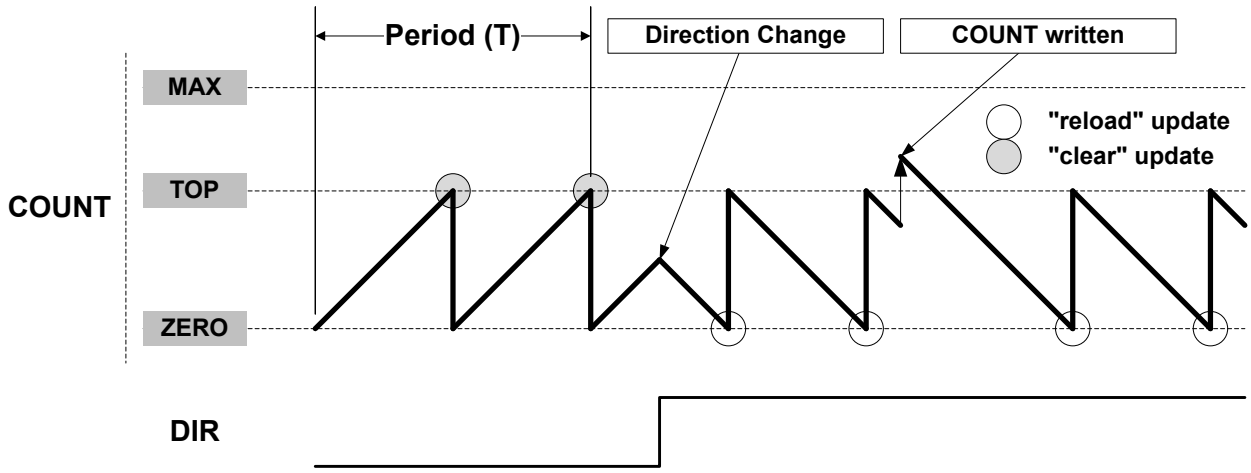
#### 30.6.2.5 Counter Operations

The counter can be set to count up or down. When the counter is counting up and the top value is reached, the counter will wrap around to zero on the next clock cycle. When counting down, the counter will wrap around to the top value when zero is reached. In one-shot mode, the counter will stop counting after a wraparound occurs.

The counting direction is set by the Direction bit in the Control B register (CTRLB.DIR). If this bit is zero the counter is counting up, and counting down if CTRLB.DIR=1. The counter will count up or down for each tick (clock or event) until it reaches TOP or ZERO. When it is counting up and TOP is reached, the counter will be set to zero at the next tick (overflow) and the Overflow Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF) will be set. It is also possible to generate an event on overflow or underflow when the Overflow/Underflow Event Output Enable bit in the Event Control register (EVCTRL.OVFEO) is one.

It is possible to change the counter value (by writing directly in the COUNT register) even when the counter is running. When starting the TC, the COUNT value will be either ZERO or TOP (depending on the counting direction set by CTRLBSET.DIR or CTRLBCLR.DIR), unless a different value has been written to it, or the TC has been stopped at a value other than ZERO. The write access has higher priority than count, clear, or reload. The direction of the counter can also be changed during normal operation. See also the figure below.

**Figure 30-3. Counter Operation**




##### 30.6.2.5.1 Stop Command and Event Action

A Stop command can be issued from software by using Command bits in the Control B Set register (CTRLBSET.CMD = 0x2, STOP). When a Stop is detected while the counter is running, the counter will retain its current value. All waveforms are cleared and the Stop bit in the Status register is set (STATUS.STOP).

#### 30.6.2.5.2 Re-Trigger Command and Event Action

A re-trigger command can be issued from software by writing the Command bits in the Control B Set register (CTRLBSET.CMD = 0x1, RETRIGGER), or from event when a re-trigger event action is configured in the Event Control register (EVCTRL.EVACT = 0x1, RETRIGGER).

When the command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction (CTRLBSET.DIR or CTRLBCLR.DIR). When the re-trigger command is detected while the counter is stopped, the counter will resume counting from the current value in the COUNT register.

**Note:** When a re-trigger event action is configured in the Event Action bits in the Event Control register (EVCTRL.EVACT=0x1, RETRIGGER), enabling the counter will not start the counter. The counter will start on the next incoming event and restart on corresponding following event.

#### 30.6.2.5.3 Count Event Action

The TC can count events. When an event is received, the counter increases or decreases the value, depending on direction settings (CTRLBSET.DIR or CTRLBCLR.DIR). The count event action can be selected by the Event Action bit group in the Event Control register (EVCTRL.EVACT=0x2, COUNT).

#### 30.6.2.5.4 Start Event Action

The TC can start counting operation on an event when previously stopped. In this configuration, the event has no effect if the counter is already counting. When the peripheral is enabled, the counter operation starts when the event is received or when a re-trigger software command is applied.

The Start TC on Event action can be selected by the Event Action bit group in the Event Control register (EVCTRL.EVACT=0x3, START).

### 30.6.2.6 Compare Operations

By default, the Compare/Capture channel is configured for compare operations.

When using the TC and the Compare/Capture Value registers (CCx) for compare operations, the counter value is continuously compared to the values in the CCx registers. This can be used for timer or for waveform operation.

#### 30.6.2.6.1 Waveform Output Operations

The compare channels can be used for waveform generation on output port pins. To make the waveform available on the connected pin, the following requirements must be fulfilled:

1. Choose a waveform generation mode in the Waveform Generation Operation bit in Waveform register (CTRLA.WAVEGEN).
2. Optionally invert the waveform output by writing the corresponding Waveform Output Invert Enable bit in the Control C register (CTRLC.INVx).
3. Configure the pins with the I/O Pin Controller. Refer to *PORT - I/O Pin Controller* for details.

The counter value is continuously compared with each CCx value. On a comparison match, the Match or Capture Channel x bit in the Interrupt Flag Status and Clear register (INTFLAG.MCx) will be set on the next zero-to-one transition of CLK_TC_CNT (see the next figure). An interrupt/and or event can be generated on comparison match when INTENSET.MCx=1 and/or EVCTRL.MCEOx=1.

There are four waveform configurations for the Waveform Generation Operation bit group in the Control A register (CTRLA.WAVEGEN) . This will influence how the waveform is generated and impose restrictions on the top value. The configurations are:

- Normal frequency (NFRQ)
- Match frequency (MFRQ)
- Normal pulse-width modulation (NPWM)

- Match pulse-width modulation (MPWM)

When using NPWM or NFRQ configuration, the TOP will be determined by the counter resolution. In 8-bit counter mode, the Period register (PER) is used as TOP, and the TOP can be changed by writing to the PER register. In 16- and 32-bit counter mode, TOP is fixed to the maximum (MAX) value of the counter.

**Related Links**

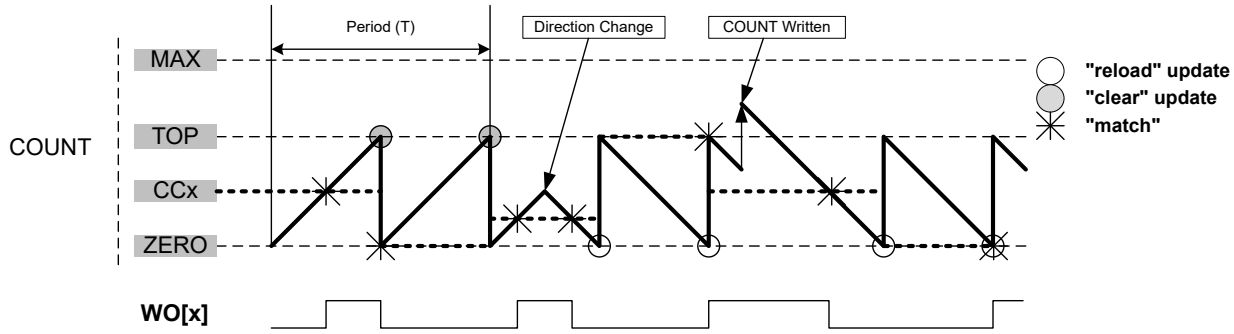
23. PORT - I/O Pin Controller

#### 30.6.2.6.2 Frequency Operation

**Normal Frequency Generation (NFRQ)**

For Normal Frequency Generation, the period time (T) is controlled by the period register (PER) for 8-bit counter mode and MAX for 16- and 32-bit mode. The waveform generation output (WO[x]) is toggled on each compare match between COUNT and CCx, and the corresponding Match or Capture Channel x Interrupt Flag (INTFLAG.MCx) will be set.

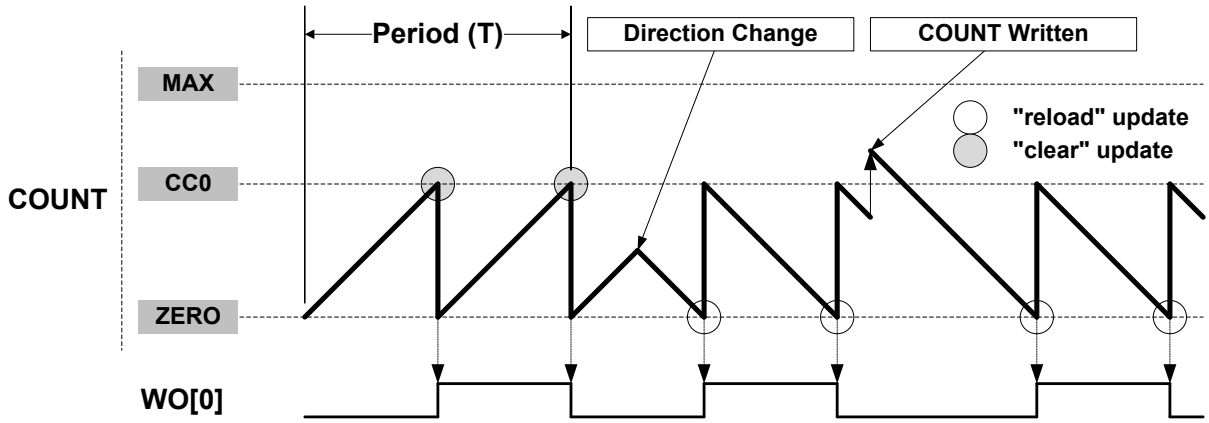
**Figure 30-4. Normal Frequency Operation**




**Match Frequency Generation (MFRQ)**

For Match Frequency Generation, the period time (T) is controlled by the CC0 register instead of PER or MAX. WO[0] toggles on each update condition.

**Figure 30-5. Match Frequency Operation**




#### 30.6.2.6.3 PWM Operation

**Normal Pulse-Width Modulation Operation (NPWM)**

NPWM uses single-slope PWM generation.

For single-slope PWM generation, the period time (T) is controlled by the TOP value, and CCx controls the duty cycle of the generated waveform output. When up-counting, the WO[x] is set at start or compare match between the COUNT and TOP values, and cleared on compare match between COUNT and CCx

register values. When down-counting, the WO[x] is cleared at start or compare match between the COUNT and ZERO values, and set on compare match between COUNT and CCx register values.

The following equation calculates the exact resolution for a single-slope PWM ($R_{PWM_SS}$) waveform:

$$R_{PWM_SS} = \frac{\log(TOP+1)}{\log(2)}$$

The PWM frequency ($f_{PWM_SS}$) depends on TOP value and the peripheral clock frequency ($f_{GCLK_TC}$), and can be calculated by the following equation:

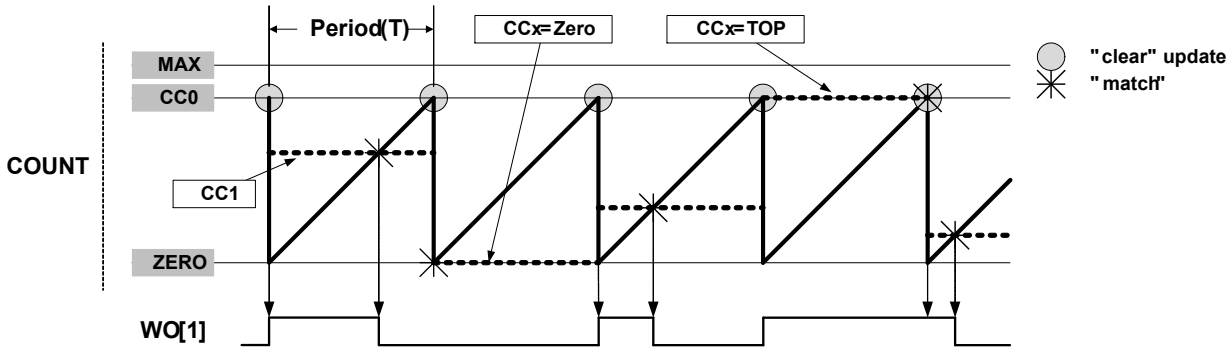
$$f_{PWM_SS} = \frac{f_{GCLK_TC}}{N(TOP+1)}$$

Where N represents the prescaler divider used (1, 2, 4, 8, 16, 64, 256, 1024).

**Match Pulse-Width Modulation Operation (MPWM)**

In MPWM, the output of WO[1] is depending on CC1 as shown in the figure below. On every overflow/ underflow, a one-TC-clock-cycle negative pulse is put out on WO[0] (not shown in the figure).

**Figure 30-6. Match PWM Operation**




The table below shows the update counter and overflow event/interrupt generation conditions in different operation modes.

**Table 30-2. Counter Update and Overflow Event/interrupt Conditions in TC**

| Name | Operation | TOP | Update | Output Waveform | | OVFIF/Event | |
|---|---|---|---|---|---|---|---|
| | | | | On Match | On Update | Up | Down |
| NFRQ | Normal Frequency | PER | TOP/ ZERO | Toggle | Stable | TOP | ZERO |
| MFRQ | Match Frequency | CC0 | TOP/ ZERO | Toggle | Stable | TOP | ZERO |
| NPWM | Single-slope PWM | PER | TOP/ ZERO | See description above. | | TOP | ZERO |
| MPWM | Single-slope PWM | CC0 | TOP/ ZERO | Toggle | Toggle | TOP | ZERO |

#### 30.6.2.6.4 Changing the Top Value

The counter period is changed by writing a new TOP value to the Period register (PER or CC0, depending on the waveform generation mode). If a new TOP value is written when the counter value is close to zero and counting down, the counter can be reloaded with the previous TOP value, due to synchronization delays. Then, the counter will count one extra cycle before the new TOP value is used.

COUNT and TOP are continuously compared, so when a new TOP value that is lower than current COUNT is written to TOP, COUNT will wrap before a compare match.

A counter wraparound can occur in any operation mode when up-counting without buffering, see the figure below.

**Figure 30-7. Changing the Top value with Up-Counting Operation**

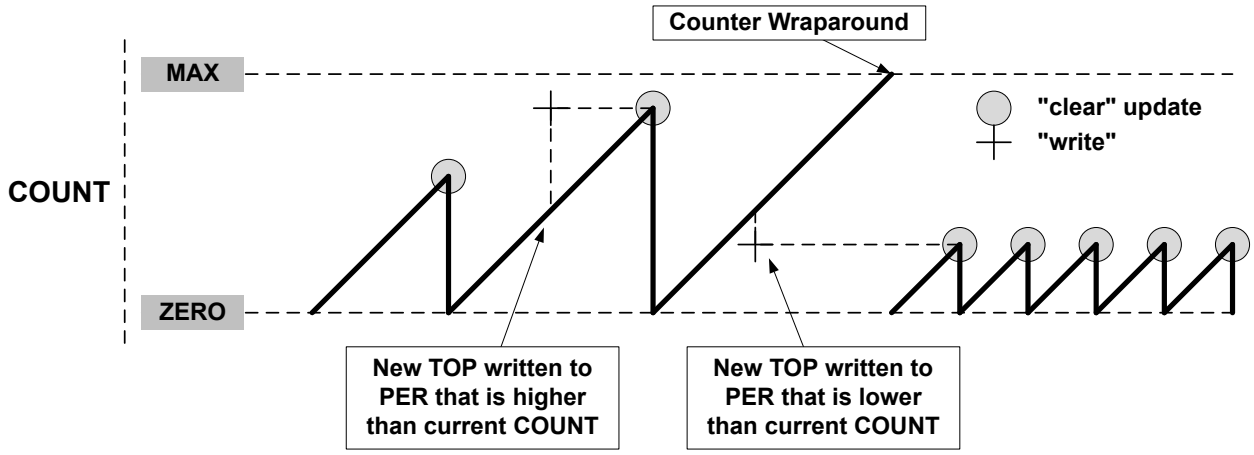
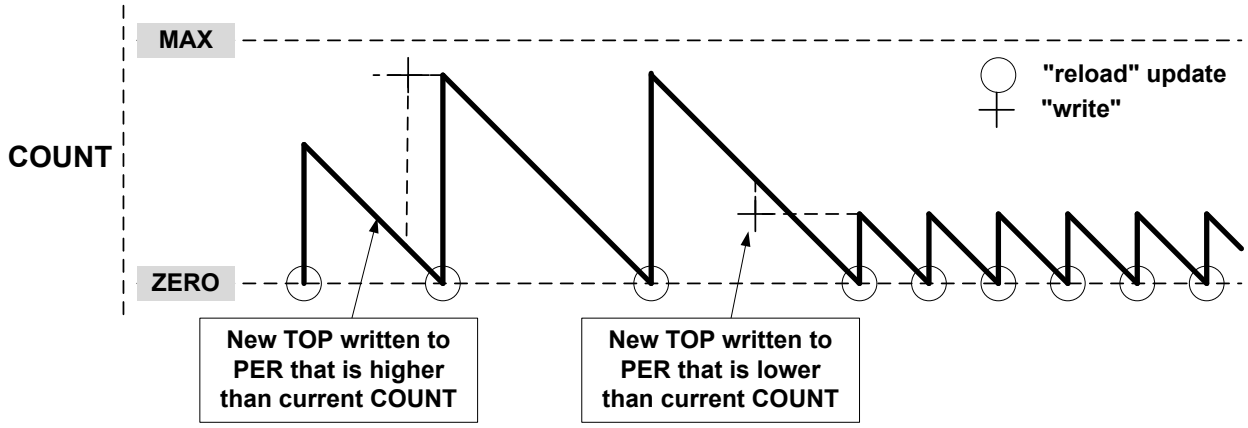



**Figure 30-8. Changing the Top Value with Down-Counting Operation**




#### 30.6.2.7 Capture Operations

To enable and use capture operations, the event line into the TC must be enabled using the TC Event Input bit in the Event Control register (EVCTRL.TCEI). The capture channels to be used must also be enabled in the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTENx) before capture can be performed.

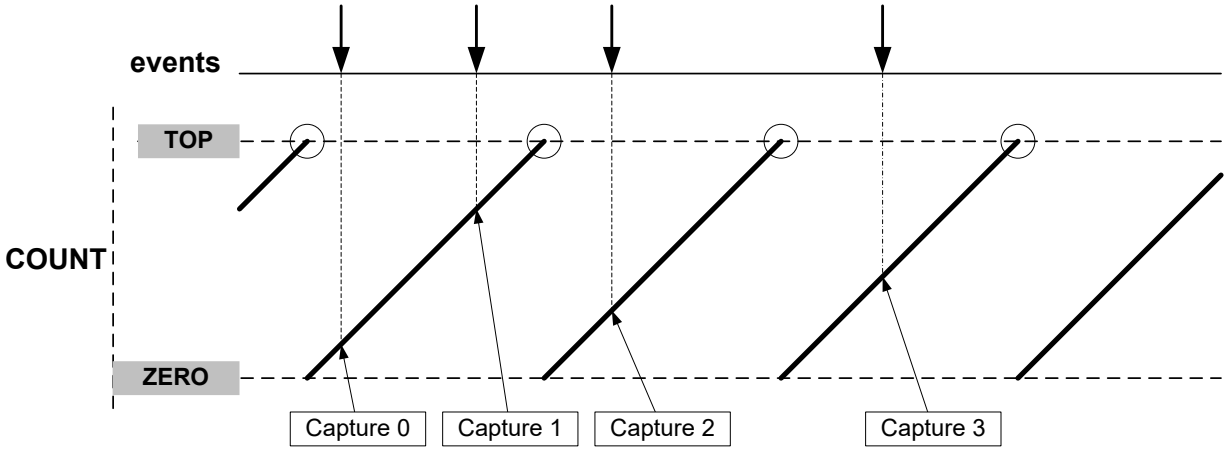
To enable and use capture operations, the corresponding Capture Channel x Enable bit in the Control C register (CTRLC.CAPTENx) must be written to '1'.

**Note:** The RETRIGGER, COUNT and START event actions are available only on an event from the Event System.

##### 30.6.2.7.1 Event Capture Action

The compare/capture channels can be used as input capture channels to capture events from the Event System and give them a timestamp. The following figure shows four capture events for one capture channel.

**Figure 30-9. Input Capture Timing**




The TC can detect capture overflow of the input capture channels: When a new capture event is detected while the Capture Interrupt flag (INTFLAG.MCx) is still set, the new timestamp will not be stored and INTFLAG.ERR will be set.

#### 30.6.2.7.2 Period and Pulse-Width (PPW) Capture Action

The TC can perform two input captures and restart the counter on one of the edges. This enables the TC to measure the pulse width and period and to characterize the frequency *f* and duty cycle of an input signal:

$$f = \frac{1}{T} \qquad \text{dutyCycle} = \frac{t_p}{T}$$

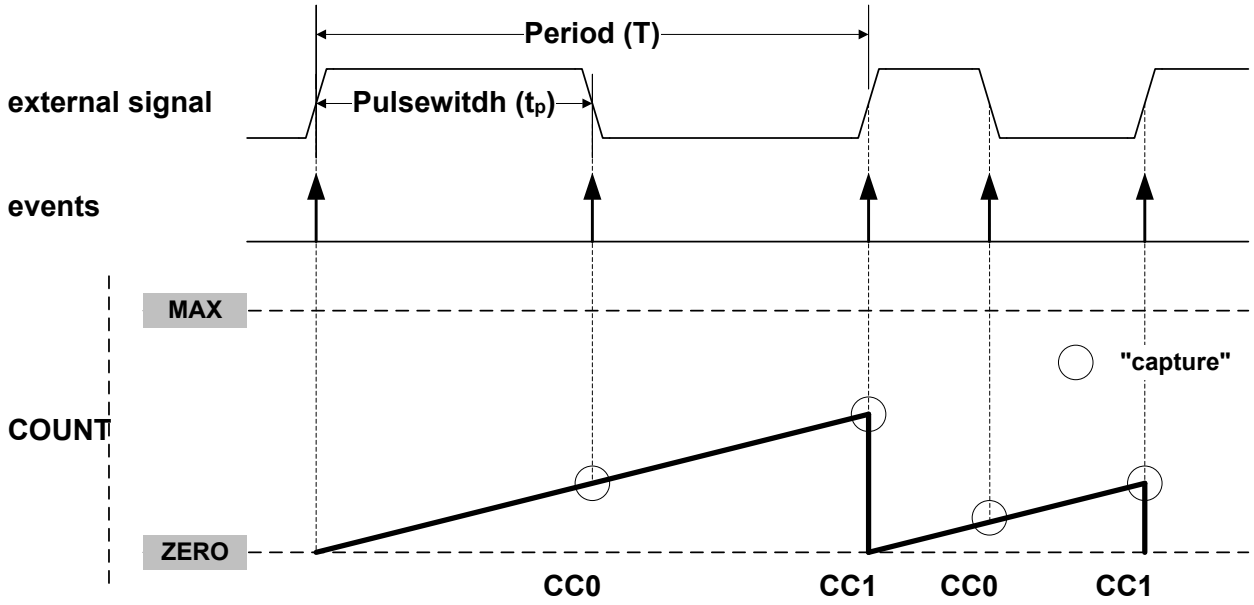
Selecting PWP (pulse-width, period) in the Event Action bit group in the Event Control register (EVCTRL.EVACT) enables the TC to perform one capture action on the rising edge and the other one on the falling edge. The period *T* will be captured into CC1 and the pulse width $t_p$ in CC0. EVCTRL.EVACT=PPW (period and pulse-width)offers identical functionality, but will capture *T* into CC0 and $t_p$ into CC1.

The TC Event Input Invert Enable bit in the Event Control register (EVCTRL.TCINV) is used to select whether the wraparound should occur on the rising edge or the falling edge. If EVCTRL.TCINV=1, the wraparound will happen on the falling edge.

To fully characterize the frequency and duty cycle of the input signal, activate capture on CC0 and CC1 by writing 0x3 to the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTEN). When only one of these measurements is required, the second channel can be used for other purposes.

The TC can detect capture overflow of the input capture channels: When a new capture event is detected while the Capture Interrupt flag (INTFLAG.MCx) is still set, the new timestamp will not be stored and INTFLAG.ERR will be set.

**Figure 30-10. PWP Capture**




### 30.6.3 Additional Features

#### 30.6.3.1 One-Shot Operation

When one-shot is enabled, the counter automatically stops on the next counter overflow or underflow condition. When the counter is stopped, the Stop bit in the Status register (STATUS.STOP) is automatically set and the waveform outputs are set to zero.

One-shot operation is enabled by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT), and disabled by writing a '1' to CTRLBCLR.ONESHOT. When enabled, the TC will count until an overflow or underflow occurs and stops counting operation. The one-shot operation can be restarted by a re-trigger software command, a re-trigger event, or a start event. When the counter restarts its operation, STATUS.STOP is automatically cleared.

### 30.6.4 DMA, Interrupts and Events

**Table 30-3. Module Request for TC**

| Condition | Interrupt request | Event output | Event input | DMA request | DMA request is cleared |
|---|---|---|---|---|---|
| Overflow / Underflow | YES | YES | | YES | Cleared on next clock cycle |
| Channel Compare Match or Capture | YES | YES | | YES[1] | For compare channel – Cleared on next clock cycle. For capture channel – cleared when CCx register is read |

| Condition | Interrupt request | Event output | Event input | DMA request | DMA request is cleared |
|---|---|---|---|---|---|
| Capture Overflow Error | YES | | | | |
| Synchronization Ready | YES | | | | |
| Start Counter | | | YES | | |
| Retrigger Counter | | | YES | | |
| Increment / Decrement counter | | | YES | | |
| Simple Capture | | | YES | | |
| Period Capture | | | YES | | |
| Pulse Width Capture | | | YES | | |

Note: 1. Two DMA requests lines are available, one for each compare/capture channel.

#### 30.6.4.1 DMA Operation

The TC can generate the following DMA requests:

- Overflow (OVF): the request is set when an update condition (overflow, underflow) is detected. The request is cleared on next clock cycle.
- Channel Match or Capture (MCx): for a compare channel, the request is set on each compare match detection and cleared on next clock cycle. For a capture channel, the request is set when valid data is present in CCx register, and cleared when CCx register is read.

When using the TC with the DMA OVF request, the new value will be transferred to the register after the update condition. This means that the value is updated after the DMA and synchronization delay, and if the COUNT value has reached the new value before PER or CCx is updated, a match will not happen.

When using the TC with the DMA MCx request and updating CCx with a value that is lower than the current COUNT when down-counting, or higher than the current COUNT when up-counting, this value could cause a new compare match before the counter overflows. This will trigger the next DMA transfer, update CCx again, and the previous value is disregarded from the output signal WO[x].

#### 30.6.4.2 Interrupts

The TC has the following interrupt sources:

- Overflow/Underflow (OVF)
- Match or Capture Channel x (MCx)
- Capture Overflow Error (ERR)
- Synchronization Ready (SYNCRDY)

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs.

Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled, or the TC is reset. on how to clear interrupt flags.

The TC has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

#### 30.6.4.3 Events

The TC can generate the following output events:

- Overflow/Underflow (OVF)
- Match or Capture (MC)

Writing a '1' to an Event Output bit in the Event Control register (EVCTRL.MCEOx) enables the corresponding output event. The output event is disabled by writing EVCTRL.MCEOx=0.

One of the following event actions can be selected by the Event Action bit group in the Event Control register (EVCTRL.EVACT):

- Start TC (START)
- Re-trigger TC (RETRIGGER)
- Increment or decrement counter (depends on counter direction)
- Count on event (COUNT)
- Capture Period (PPW and PWP)

Writing a '1' to the TC Event Input bit in the Event Control register (EVCTRL.TCEI) enables input events to the TC. Writing a '0' to this bit disables input events to the TC. The TC requires only asynchronous event inputs. For further details on how configuring the asynchronous events, refer to *EVSYS - Event System*.

**Related Links**

24. EVSYS – Event System

### 30.6.5 Sleep Mode Operation

The TC can be configured to operate in any sleep mode. To be able to run in standby, the RUNSTDBY bit in the Control A register (CTRLA.RUNSTDBY) must be written to one. The TC can wake up the device using interrupts from any sleep mode or perform actions through the Event System.

### 30.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)

- Enable bit in the Control A register (CTRLA.ENABLE)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

The following registers are synchronized when written:

- Control B Clear register (CTRLBCLR)
- Control B Set register (CTRLBSET)
- Control C register (CTRLC)
- Count Value register (COUNT)
- Period Value register (PER)
- Compare/Capture Value registers (CCx)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

The following registers are synchronized when read:

- Control B Clear register (CTRLBCLR)
- Control B Set register (CTRLBSET)
- Control C register (CTRLC)
- Count Value register (COUNT)
- Period Value register (PER)
- Compare/Capture Value registers (CCx)

Required read-synchronization is denoted by the "Read-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 30.7 Register Summary

**Table 30-4. Register Summary – 8-bit Mode**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | WAVEGEN[1:0] | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | PRESCSYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| 0x02 | READREQ | 7:0 | | | | ADDR[4:0] | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | CTRLBCLR | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x05 | CTRLBSET | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x06 | CTRLC | 7:0 | | | CPTEN1 | CPTEN0 | | | INVEN1 | INVEN0 |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | EVCTRL | 7:0 | | | TCEI | TCINV | | EVACT[2:0] | | |
| 0x0B | | 15:8 | | | MCEO1 | MCEO0 | | | | OVFEO |
| 0x0C | INTENCLR | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0D | INTENSET | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0E | INTFLAG | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0F | STATUS | 7:0 | SYNCBUSY | | | SLAVE | STOP | | | |
| 0x10 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| 0x11 | Reserved | | | | | | | | | |
| 0x12 | Reserved | | | | | | | | | |
| 0x13 | Reserved | | | | | | | | | |
| 0x14 | PER | 7:0 | PER[7:0] | | | | | | | |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | CC0 | 7:0 | CC[7:0] | | | | | | | |
| 0x19 | CC1 | 7:0 | CC[7:0] | | | | | | | |
| 0x1A | Reserved | | | | | | | | | |
| 0x1B | Reserved | | | | | | | | | |
| 0x1C | Reserved | | | | | | | | | |
| 0x1D | Reserved | | | | | | | | | |
| 0x1E | Reserved | | | | | | | | | |
| 0x1F | Reserved | | | | | | | | | |

**Table 30-5. Register Summary – 16-bit Mode**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | WAVEGEN[1:0] | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | PRESCSYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| 0x02 | READREQ | 7:0 | | | | ADDR[4:0] | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | CTRLBCLR | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x05 | CTRLBSET | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x06 | CTRLC | 7:0 | | | CPTEN1 | CPTEN0 | | | INVEN1 | INVEN0 |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | EVCTRL | 7:0 | | | TCEI | TCINV | | EVACT[2:0] | | |
| 0x0B | | 15:8 | | | MCEO1 | MCEO0 | | | | OVFEO |
| 0x0C | INTENCLR | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0D | INTENSET | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0E | INTFLAG | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0F | STATUS | 7:0 | SYNCBUSY | | | SLAVE | STOP | | | |
| 0x10 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| 0x11 | | 15:8 | COUNT[15:8] | | | | | | | |
| 0x12 | Reserved | | | | | | | | | |
| 0x13 | Reserved | | | | | | | | | |
| 0x14 | Reserved | | | | | | | | | |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | CC0 | 7:0 | CC[7:0] | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x19 | | 15:8 | CC[15:8] | | | | | | | |
| 0x1A | CC1 | 7:0 | CC[7:0] | | | | | | | |
| 0x1B | | 15:8 | CC[15:8] | | | | | | | |
| 0x1C | Reserved | | | | | | | | | |
| 0x1D | Reserved | | | | | | | | | |
| 0x1E | Reserved | | | | | | | | | |
| 0x1F | Reserved | | | | | | | | | |

**Table 30-6. Register Summary – 32-bit Mode**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | WAVEGEN[1:0] | | | MODE[1:0] | | ENABLE | SWRST |
| 0x01 | | 15:8 | | | PRESCSYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| 0x02 | READREQ | 7:0 | | | | ADDR[4:0] | | | | |
| 0x03 | | 15:8 | RREQ | RCONT | | | | | | |
| 0x04 | CTRLBCLR | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x05 | CTRLBSET | 7:0 | CMD[1:0] | | | | | ONESHOT | | DIR |
| 0x06 | CTRLC | 7:0 | | | CPTEN1 | CPTEN0 | | | INVEN1 | INVEN0 |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | DBGCTRL | 7:0 | | | | | | | | DBGRUN |
| 0x09 | Reserved | | | | | | | | | |
| 0x0A | EVCTRL | 7:0 | | | TCEI | TCINV | | EVACT[2:0] | | |
| 0x0B | | 15:8 | | | MCEO1 | MCEO0 | | | | OVFEO |
| 0x0C | INTENCLR | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0D | INTENSET | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0E | INTFLAG | 7:0 | | | MC1 | MC0 | SYNCRDY | | ERR | OVF |
| 0x0F | STATUS | 7:0 | SYNCBUSY | | | SLAVE | STOP | | | |
| 0x10 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| 0x11 | | 15:8 | COUNT[15:8] | | | | | | | |
| 0x12 | | 23:16 | COUNT[23:16] | | | | | | | |
| 0x13 | | 31:24 | COUNT[31:24] | | | | | | | |
| 0x14 | Reserved | | | | | | | | | |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | CC0 | 7:0 | CC[7:0] | | | | | | | |
| 0x19 | | 15:8 | CC[15:8] | | | | | | | |
| 0x1A | | 23:16 | CC[23:16] | | | | | | | |
| 0x1B | | 31:24 | CC[31:24] | | | | | | | |
| 0x1C | CC1 | 7:0 | CC[7:0] | | | | | | | |
| 0x1D | | 15:8 | CC[15:8] | | | | | | | |
| 0x1E | | 23:16 | CC[23:16] | | | | | | | |
| 0x1F | | 31:24 | CC[31:24] | | | | | | | |

## 30.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 30.5.8 Register Access Protection

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to 30.6.6 Synchronization.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 30.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Write-Synchronized, Enable-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PRESCSYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | WAVEGEN[1:0] | | | MODE[1:0] | | ENABLE | SWRST |
| Access | | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | | 0 | 0 | 0 | 0 |

**Bits 13:12 – PRESCSYNC[1:0]** Prescaler and Counter Synchronization
These bits select whether the counter should wrap around on the next GCLK_TCx clock or the next prescaled GCLK_TCx clock. It also makes it possible to reset the prescaler.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | GCLK | Reload or reset the counter on next generic clock |
| 0x1 | PRESC | Reload or reset the counter on next prescaler clock |
| 0x2 | RESYNC | Reload or reset the counter on next generic clock. Reset the prescaler counter |
| 0x3 | - | Reserved |

**Bit 11 – RUNSTDBY** Run in Standby
This bit is used to keep the TC running in standby mode.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The TC is halted in standby. |
| 1 | The TC continues to run in standby. |

**Bits 10:8 – PRESCALER[2:0]** Prescaler
These bits select the counter prescaler factor.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Prescaler: GCLK_TC |
| 0x1 | DIV2 | Prescaler: GCLK_TC/2 |
| 0x2 | DIV4 | Prescaler: GCLK_TC/4 |
| 0x3 | DIV8 | Prescaler: GCLK_TC/8 |
| 0x4 | DIV16 | Prescaler: GCLK_TC/16 |
| 0x5 | DIV64 | Prescaler: GCLK_TC/64 |

| Value | Name | Description |
|---|---|---|
| 0x6 | DIV256 | Prescaler: GCLK_TC/256 |
| 0x7 | DIV1024 | Prescaler: GCLK_TC/1024 |

**Bits 6:5 – WAVEGEN[1:0]** Waveform Generation Operation
These bits select the waveform generation operation. They affect the top value, as shown in “Waveform Output Operations”. It also controls whether frequency or PWM waveform generation should be used. How these modes differ can also be seen from “Waveform Output Operations”.

These bits are not synchronized.

**Table 30-7. Waveform Generation Operation**

| Value | Name | Operation | Top Value | Waveform Output on Match | Waveform Output on Wraparound |
|---|---|---|---|---|---|
| 0x0 | NFRQ | Normal frequency | PER(1)/Max | Toggle | No action |
| 0x1 | MFRQ | Match frequency | CC0 | Toggle | No action |
| 0x2 | NPWM | Normal PWM | PER(1)/Max | Clear when counting up Set when counting down | Set when counting up Clear when counting down |
| 0x3 | MPWM | Match PWM | CC0 | Clear when counting up Set when counting down | Set when counting up Clear when counting down |

**Note:**

1. This depends on the TC mode. In 8-bit mode, the top value is the Period Value register (PER). In 16- and 32-bit mode it is the maximum value.

**Bits 3:2 – MODE[1:0]** Timer Counter Mode
These bits select the counter mode.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | COUNT16 | Counter in 16-bit mode |
| 0x1 | COUNT8 | Counter in 8-bit mode |
| 0x2 | COUNT32 | Counter in 32-bit mode |
| 0x3 | - | Reserved |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately, and the ENABLE Synchronization Busy bit in the SYNCBUSY register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

This bit is not enable protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers in the TC, except DBGCTRL, to their initial state, and the TC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write-operation will be discarded.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is not enable protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 30.8.2 Read Request

**Name:** READREQ
**Offset:** 0x02
**Reset:** 0x0000

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RREQ | RCONT | | | | | | |
| Access | W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ADDR[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – RREQ** Read Request
Writing a zero to this bit has no effect.

This bit will always read as zero.

Writing a one to this bit requests synchronization of the register pointed to by the Address bit group (READREQ. ADDR) and sets the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY).

**Bit 14 – RCONT** Read Continuously
When continuous synchronization is enabled, the register pointed to by the Address bit group (READREQ.ADDR) will be synchronized automatically every time the register is updated.

| Value | Description |
|---|---|
| 0 | Continuous synchronization is disabled. |
| 1 | Continuous synchronization is enabled. |

**Bits 4:0 – ADDR[4:0]** Address
These bits select the offset of the register that needs read synchronization. In the TC, only COUNT and CCx are available for read synchronization.

### 30.8.3 Control B Clear

**Name:** CTRLBCLR
**Offset:** 0x04
**Reset:** 0x00
**Property:** PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to clear bits in the CTRLB register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set register (CTRLBSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD[1:0] | | | | | ONESHOT | | DIR |
| Access | R/W | R/W | | | | R/W | | R/W |
| Reset | 0 | 0 | | | | 0 | | 0 |

**Bits 7:6 – CMD[1:0]** Command
These bits are used for software control of the TC. The commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit group will be read back as zero.

Writing 0x0 to these bits has no effect.

Writing a '1' to any of these bits will clear the pending command.

**Table 30-8. Command**

| Value | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | RETRIGGER | Force a start, restart or retrigger |
| 0x2 | STOP | Force a stop |
| 0x3 | - | Reserved |

**Bit 2 – ONESHOT** One-Shot on Counter
This bit controls one-shot operation of the TC.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will disable one-shot operation.

| Value | Description |
|---|---|
| 0 | The TC will wrap around and continue counting on an overflow/underflow condition. |
| 1 | The TC will wrap around and stop on the next underflow/overflow condition. |

**Bit 0 – DIR** Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the bit and make the counter count up.

| Value | Description |
|---|---|
| 0 | The timer/counter is counting up (incrementing). |
| 1 | The timer/counter is counting down (decrementing). |

### 30.8.4 Control B Set

**Name:** CTRLBSET
**Offset:** 0x05
**Reset:** 0x00
**Property:** PAC Write-Protection, Read-synchronized, Write-Synchronized

This register allows the user to set bits in the CTRLB register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Clear register (CTRLBCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD[1:0] | | | | | ONESHOT | | DIR |
| Access | R/W | R/W | | | | R/W | | R/W |
| Reset | 0 | 0 | | | | 0 | | 0 |

**Bits 7:6 – CMD[1:0]** Command
These bits are used for software control of the TC. The commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit group will be read back as zero.

Writing 0x0 to these bits has no effect.

Writing a '1' to any of these bits will clear the pending command.

**Table 30-9. Command**

| Value | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | RETRIGGER | Force a start, restart or retrigger |
| 0x2 | STOP | Force a stop |
| 0x3 | - | Reserved |

**Bit 2 – ONESHOT** One-Shot on Counter
This bit controls one-shot operation of the TC.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will disable one-shot operation.

| Value | Description |
|---|---|
| 0 | The TC will wrap around and continue counting on an overflow/underflow condition. |
| 1 | The TC will wrap around and stop on the next underflow/overflow condition. |

**Bit 0 – DIR** Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the bit and make the counter count up.

| Value | Description |
|---|---|
| 0 | The timer/counter is counting up (incrementing). |
| 1 | The timer/counter is counting down (decrementing). |

### 30.8.5 Control C

**Name:** CTRLC
**Offset:** 0x06
**Reset:** 0x00
**Property:** PAC Write-Protection, Read-synchronized, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CPTENx | CPTENx | | | INVENx | INVENx |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 5,4 – CPTENx** Capture Channel x Enable
These bits are used to select the capture or compare operation on channel x.

Writing a '1' to CPTENx enables capture on channel x.

Writing a '0' to CPTENx disables capture on channel x.

**Bits 1,0 – INVENx** Waveform Output x Inversion Enable
These bits are used to select inversion on the output of channel x.

Writing a '1' to INVENx inverts output from WO[x].

Writing a '0' to INVENx disables inversion of output from WO[x].

### 30.8.6 Debug Control

**Name:** DBGCTRL
**Offset:** 0x08
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGRUN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGRUN** Debug Run Mode
This bit is not affected by a software reset, and should not be changed by software while the TC is enabled.

| Value | Description |
|---|---|
| 0 | The TC is halted when the device is halted in debug mode. |
| 1 | The TC continues normal operation when the device is halted in debug mode. |

### 30.8.7 Event Control

**Name:** EVCTRL
**Offset:** 0x0A
**Reset:** 0x0000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | MCEOx | MCEOx | | | | OVFEO |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TCEI | TCINV | | EVACT[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bits 13,12 – MCEOx** Match or Capture Channel x Event Output Enable [x = 1..0]
These bits enable the generation of an event for every match or capture on channel x.

| Value | Description |
|---|---|
| 0 | Match/Capture event on channel x is disabled and will not be generated. |
| 1 | Match/Capture event on channel x is enabled and will be generated for every compare/capture. |

**Bit 8 – OVFEO** Overflow/Underflow Event Output Enable
This bit enables the Overflow/Underflow event. When enabled, an event will be generated when the counter overflows/underflows.

| Value | Description |
|---|---|
| 0 | Overflow/Underflow event is disabled and will not be generated. |
| 1 | Overflow/Underflow event is enabled and will be generated for every counter overflow/underflow. |

**Bit 5 – TCEI** TC Event Enable
This bit is used to enable asynchronous input events to the TC.

| Value | Description |
|---|---|
| 0 | Incoming events are disabled. |
| 1 | Incoming events are enabled. |

**Bit 4 – TCINV** TC Inverted Event Input Polarity
This bit inverts the asynchronous input event source.

| Value | Description |
|---|---|
| 0 | Input event source is not inverted. |
| 1 | Input event source is inverted. |

**Bits 2:0 – EVACT[2:0]** Event Action
These bits define the event action the TC will perform on an event.

| Value | Name | Description |
|---|---|---|
| 0x0 | OFF | Event action disabled |
| 0x1 | RETRIGGER | Start, restart or retrigger TC on event |
| 0x2 | COUNT | Count on event |
| 0x3 | START | Start TC on event |
| 0x4 | - | Reserved |
| 0x5 | PPW | Period captured in CC0, pulse width in CC1 |
| 0x6 | PWP | Period captured in CC1, pulse width in CC0 |
| 0x7 | - | Reserved |

### 30.8.8 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x0C
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | MCx | MCx | SYNCRDY | | ERR | OVF |
| Access | | | R/W | R/W | R/W | | R/W | R/W |
| Reset | | | 0 | 0 | 0 | | 0 | 0 |

**Bits 5,4 – MCx** Match or Capture Channel x Interrupt Enable [x = 1..0]
Writing a '0' to these bits has no effect.

Writing a '1' to MCx will clear the corresponding Match or Capture Channel x Interrupt Enable bit, which disables the Match or Capture Channel x interrupt.

| Value | Description |
|---|---|
| 0 | The Match or Capture Channel x interrupt is disabled. |
| 1 | The Match or Capture Channel x interrupt is enabled. |

**Bit 3 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Disable/Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – ERR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | The Error interrupt is disabled. |
| 1 | The Error interrupt is enabled. |

**Bit 0 – OVF** Overflow Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, which disables the Overflow interrupt request.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled. |

### 30.8.9 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x0D
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | MCx | MCx | SYNCRDY | | ERR | OVF |
| Access | | | R/W | R/W | R/W | | R/W | R/W |
| Reset | | | 0 | 0 | 0 | | 0 | 0 |

**Bits 5,4 – MCx** Match or Capture Channel x Interrupt Enable [x = 1..0]
Writing a '0' to these bits has no effect.

Writing a '1' to MCx will set the corresponding Match or Capture Channel x Interrupt Enable bit, which enables the Match or Capture Channel x interrupt.

| Value | Description |
|---|---|
| 0 | The Match or Capture Channel x interrupt is disabled. |
| 1 | The Match or Capture Channel x interrupt is enabled. |

**Bit 3 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Disable/Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – ERR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | The Error interrupt is disabled. |
| 1 | The Error interrupt is enabled. |

**Bit 0 – OVF** Overflow Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Overflow Interrupt Enable bit, which enables the Overflow interrupt request.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled. |

### 30.8.10 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x0E
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | MCx | MCx | SYNCRDY | | ERR | OVF |
| Access | | | R/W | R/W | R/W | | R/W | R/W |
| Reset | | | 0 | 0 | 0 | | 0 | 0 |

**Bits 5,4 – MCx** Match or Capture Channel x [x = 1..0]
This flag is set on a comparison match, or when the corresponding CCx register contains a valid capture value. This flag is set on the next CLK_TC_CNT cycle, and will generate an interrupt request if the corresponding Match or Capture Channel x Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCx) is '1'.

Writing a '0' to one of these bits has no effect.

Writing a '1' to one of these bits will clear the corresponding Match or Capture Channel x interrupt flag

In capture operation, this flag is automatically cleared when CCx register is read.

**Bit 3 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Disable/Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – ERR** Error Interrupt Flag
This flag is set when a new capture occurs on a channel while the corresponding Match or Capture Channel x interrupt flag is set, in which case there is nowhere to store the new capture.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Error interrupt flag.

**Bit 0 – OVF** Overflow Interrupt Flag
This flag is set on the next CLK_TC_CNT cycle after an overflow condition occurs, and will generate an interrupt request if INTENCLR.OVF or INTENSET.OVF is '1'.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Overflow interrupt flag.

### 30.8.11 Status

**Name:** STATUS
**Offset:** 0x0F
**Reset:** 0x08
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | SLAVE | STOP | | | |
| Access | R | | | R | R | | | |
| Reset | 0 | | | 0 | 1 | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

**Bit 4 – SLAVE** Slave Status Flag
This bit is only available in 32-bit mode on the slave TC (i.e., TC5 and/or TC7). The bit is set when the associated master TC (TC4 and TC6, respectively) is set to run in 32-bit mode.

**Bit 3 – STOP** Stop Status Flag
This bit is set when the TC is disabled, on a Stop command, or on an overflow/underflow condition when the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) is '1'.

| Value | Description |
|---|---|
| 0 | Counter is running. |
| 1 | Counter is stopped. |

### 30.8.12 Counter Value

#### 30.8.12.1 Counter Value, 8-bit Mode

**Name:** COUNT
**Offset:** 0x10
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – COUNT[7:0]** Counter Value
These bits contain the current counter value.

#### 30.8.12.2 Counter Value, 16-bit Mode

**Name:** COUNT
**Offset:** 0x10
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COUNT[15:0]** Counter Value
These bits contain the current counter value.

#### 30.8.12.3 Counter Value, 32-bit Mode

**Name:** COUNT
**Offset:** 0x10
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – COUNT[31:0]** Counter Value
These bits contain the current counter value.

### 30.8.13 Period Value

#### 30.8.13.1 Period Value, 8-bit Mode

**Name:** PER
**Offset:** 0x14
**Reset:** 0xFF
**Property:** Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PER[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 7:0 – PER[7:0]** Period Value
These bits hold the value of the Period Buffer register PERBUF. The value is copied to PER register on UPDATE condition.

### 30.8.14 Compare/Capture

#### 30.8.14.1 Channel x Compare/Capture Value, 8-bit Mode

**Name:** CCx
**Offset:** 0x18+i*0x1 [i=0..1]
**Reset:** 0x00
**Property:** Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – CC[7:0]** Channel x Compare/Capture Value
These bits contain the compare/capture value in 8-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

#### 30.8.14.2 Channel x Compare/Capture Value, 16-bit Mode

**Name:** CCx
**Offset:** 0x18+i*0x2 [i=0..1]
**Reset:** 0x0000
**Property:** Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CC[15:0]** Channel x Compare/Capture Value
These bits contain the compare/capture value in 16-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

### 30.8.14.3 Channel x Compare/Capture Value, 32-bit Mode

**Name:** CCx
**Offset:** 0x18+i*0x4 [i=0..1]
**Reset:** 0x00000000
**Property:** Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CC[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CC[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CC[31:0]** Channel x Compare/Capture Value
These bits contain the compare/capture value in 32-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

# 31. TCC – Timer/Counter for Control Applications

## 31.1 Overview

The device provides three instances of the Timer/Counter for Control applications (TCC) peripheral, TCC[2:0].

Each TCC instance consists of a counter, a prescaler, compare/capture channels and control logic. The counter can be set to count events or clock pulses. The counter together with the compare/capture channels can be configured to time stamp input events, allowing capture of frequency and pulse-width. It can also perform waveform generation such as frequency generation and pulse-width modulation.

Waveform extensions are featured for motor control, ballast, LED, H-bridge, power converters, and other types of power control applications. They allow for low- and high-side output with optional dead-time insertion. Waveform extensions can also generate a synchronized bit pattern across the waveform output pins. The fault options enable fault protection for safe and deterministic handling, disabling and/or shut down of external drivers.

Figure 31-1 shows all features in TCC.

**Note:** The TCC configurations, such as channel numbers and features, may be reduced for some of the TCC instances.

**Related Links**

7.2.5 TCC Configurations

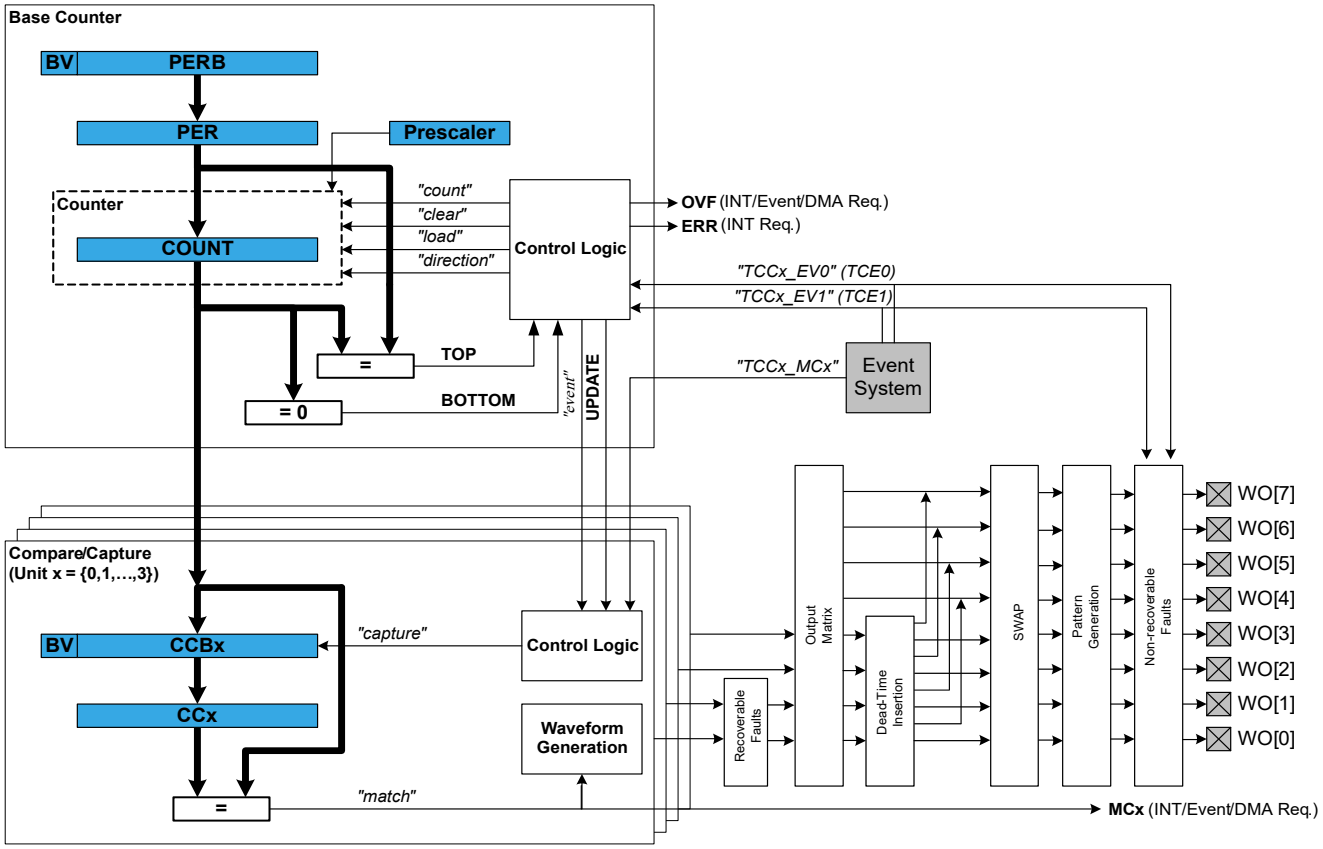
## 31.2 Features

- Up to four compare/capture channels (CC) with:
  - Double buffered period setting
  - Double buffered compare or capture channel
  - Circular buffer on period and compare channel registers
- Waveform generation:
  - Frequency generation
  - Single-slope pulse-width modulation (PWM)
  - Dual-slope pulse-width modulation with half-cycle reload capability
- Input capture:
  - Event capture
  - Frequency capture
  - Pulse-width capture
- Waveform extensions:
  - Configurable distribution of compare channels outputs across port pins
  - Low- and high-side output with programmable dead-time insertion
  - Waveform swap option with double buffer support
  - Pattern generation with double buffer support
  - Dithering support
- Fault protection for safe disabling of drivers:

  - Two recoverable fault sources
  - Two non-recoverable fault sources
  - Debugger can be source of non-recoverable fault
- Input events:
  - Two input events (EVx) for counter
  - One input event (MCx) for each channel
- Output events:
  - Three output events (Count, Re-Trigger and Overflow) available for counter
  - One Compare Match/Input Capture event output for each channel
- Interrupts:
  - Overflow and Re-Trigger interrupt
  - Compare Match/Input Capture interrupt
  - Interrupt on fault detection
- Can be used with DMA and can trigger DMA transactions

## 31.3 Block Diagram

**Figure 31-1. Timer/Counter for Control Applications - Block Diagram**



## 31.4 Signal Description

| Pin Name | Type | Description |
|---|---|---|
| TCCx/WO[0] | Digital output | Compare channel 0 waveform output |
| TCCx/WO[1] | Digital output | Compare channel 1 waveform output |
| … | ... | ... |
| TCCx/WO[WO_NUM-1] | Digital output | Compare channel n waveform output |

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 31.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 31.5.1 I/O Lines

In order to use the I/O lines of this peripheral, the I/O pins must be configured using the I/O Pin Controller (PORT).

**Related Links**

23. PORT - I/O Pin Controller

### 31.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes.

### 31.5.3 Clocks

The TCC bus clocks (CLK_TCCx_APB) can be enabled and disabled in the Power Manager module. The default state of CLK_TCCx_APB can be found in the Peripheral Clock Masking section (see the Related Links below).

A generic clock (GCLK_TCCx) is required to clock the TCC. This clock must be configured and enabled in the generic clock controller before using the TCC. Note that TCC0 and TCC1 share a peripheral clock generator.

The generic clocks (GCLK_TCCx) are asynchronous to the bus clock (CLK_TCCx_APB). Due to this asynchronicity, writing certain registers will require synchronization between the clock domains. Refer to 31.6.6 Synchronization for further details.

**Related Links**

15. GCLK - Generic Clock Controller

16.6.2.6 Peripheral Clock Masking

### 31.5.4 DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 31.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 31.5.6 Events

The events of this peripheral are connected to the Event System.

**Related Links**

24. EVSYS – Event System

### 31.5.7 Debug Operation

When the CPU is halted in debug mode, this peripheral will halt normal operation. This peripheral can be forced to continue operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

Refer to 31.8.8 DBGCTRL register for details.

### 31.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Interrupt Flag register (INTFLAG)
- Status register (STATUS)
- Period and Period Buffer registers (PER, PERB)
- Compare/Capture and Compare/Capture Buffer registers (CCx, CCBx)
- Control Waveform register (WAVE)
- Control Waveform Buffer register (WAVEB)
- Pattern Generation Value and Pattern Generation Value Buffer registers (PATT, PATTB)

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 31.5.9 Analog Connections

Not applicable.

## 31.6 Functional Description

### 31.6.1 Principle of Operation

The following definitions are used throughout the documentation:

**Table 31-1. Timer/Counter for Control Applications - Definitions**

| Name | Description |
|---|---|
| TOP | The counter reaches TOP when it becomes equal to the highest value in the count sequence. The TOP value can be the same as Period (PER) or the Compare Channel 0 (CC0) register value depending on the waveform generator mode in 31.6.2.5.1 Waveform Output Generation Operations. |
| ZERO | The counter reaches ZERO when it contains all zeroes. |
| MAX | The counter reaches maximum when it contains all ones. |
| UPDATE | The timer/counter signals an update when it reaches ZERO or TOP, depending on the direction settings. |
| Timer | The timer/counter clock control is handled by an internal source. |
| Counter | The clock control is handled externally (e.g., counting external events). |
| CC | For compare operations, the CC are referred to as "compare channels."<br>For capture operations, the CC are referred to as "capture channels." |

Each TCC instance has up to four compare/capture channels (CCx).

The counter register (COUNT), period registers with buffer (PER and PERB), and compare and capture registers with buffers (CCx and CCBx) are 16- or 24-bit registers, depending on each TCC instance. Each buffer register has a buffer valid (BUFV) flag that indicates when the buffer contains a new value.

Under normal operation, the counter value is continuously compared to the TOP or ZERO value to determine whether the counter has reached TOP or ZERO. In either case, the TCC can generate interrupt requests, request DMA transactions, or generate events for the Event System. In waveform generator mode, these comparisons are used to set the waveform period or pulse width.

A prescaled generic clock (GCLK_TCCx) and events from the event system can be used to control the counter. The event system is also used as a source to the input capture.

The Recoverable Fault Unit enables event controlled waveforms by acting directly on the generated waveforms of the TCC compare channels output. These events can restart, halt the timer/counter period, shorten the output pulse active time, or disable waveform output as long as the fault condition is present. This can typically be used for current sensing regulation, and zero-crossing and demagnetization re-triggering.

The MCE0 and MCE1 asynchronous event sources are shared with the Recoverable Fault Unit. Only asynchronous events are used internally when fault unit extension is enabled. For further details on how to configure asynchronous events routing, refer to *EVSYS – Event System*.

Recoverable fault sources can be filtered and/or windowed to avoid false triggering, for example from I/O pin glitches, by using digital filtering, input blanking, and qualification options. See also 31.6.3.5 Recoverable Faults.

In order to support applications with different types of motor control, ballast, LED, H-bridge, power converter, and other types of power switching applications, the following independent units are implemented in some of the TCC instances as optional and successive units:

- Recoverable faults and non-recoverable faults
- Output matrix
- Dead-time insertion
- Swap
- Pattern generation

See also Figure 31-1.

The output matrix (OTMX) can distribute and route out the TCC waveform outputs across the port pins in different configurations, each optimized for different application types. The Dead-Time Insertion (DTI) unit splits the four lower OTMX outputs into two non-overlapping signals: the non-inverted low side (LS) and inverted high side (HS) of the waveform output with optional dead-time insertion between LS and HS switching. The SWAP unit can swap the LS and HS pin outputs, and can be used for fast decay motor control.

The pattern generation unit can be used to generate synchronized waveforms with constant logic level on TCC UPDATE conditions. This is useful for easy stepper motor and full bridge control.

The non-recoverable fault module enables event controlled fault protection by acting directly on the generated waveforms of the timer/counter compare channel outputs. When a non-recoverable fault condition is detected, the output waveforms are forced to a preconfigured value that is safe for the application. This is typically used for instant and predictable shut down and disabling high current or voltage drives.

The count event sources (TCE0 and TCE1) are shared with the non-recoverable fault extension. The events can be optionally filtered. If the filter options are not used, the non-recoverable faults provide an immediate asynchronous action on waveform output, even for cases where the clock is not present. For further details on how to configure asynchronous events routing, refer to section *EVSYS – Event System*.

**Related Links**

24. EVSYS – Event System

### 31.6.2 Basic Operation

#### 31.6.2.1 Initialization

The following registers are enable-protected, meaning that they can only be written when the TCC is disabled(CTRLA.ENABLE=0):

- Control A (CTRLA) register, except Run Standby (RUNSTDBY), Enable (ENABLE) and Software Reset (SWRST) bits
- Recoverable Fault n Control registers (FCTRLA and FCTRLB)
- Waveform Extension Control register (WEXCTRL)
- Drive Control register (DRVCTRL)
- Event Control register (EVCTRL)

Enable-protected bits in the CTRLA register can be written at the same time as CTRLA.ENABLE is written to '1', but not at the same time as CTRLA.ENABLE is written to '0'. Enable-protection is denoted by the “Enable-Protected” property in the register description.

Before the TCC is enabled, it must be configured as outlined by the following steps:

1. Enable the TCC bus clock (CLK_TCCx_APB).
2. If Capture mode is required, enable the channel in capture mode by writing a '1' to the Capture Enable bit in the Control A register (CTRLA.CPTEN).

Optionally, the following configurations can be set before enabling TCC:

1. Select PRESCALER setting in the Control A register (CTRLA.PRESCALER).
2. Select Prescaler Synchronization setting in Control A register (CTRLA.PRESCSYNC).
3. If down-counting operation is desired, write the Counter Direction bit in the Control B Set register (CTRLBSET.DIR) to '1'.
4. Select the Waveform Generation operation in the WAVE register (WAVE.WAVEGEN).
5. Select the Waveform Output Polarity in the WAVE register (WAVE.POL).
6. The waveform output can be inverted for the individual channels using the Waveform Output Invert Enable bit group in the Driver register (DRVCTRL.INVEN).

#### 31.6.2.2 Enabling, Disabling, and Resetting

The TCC is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The TCC is disabled by writing a zero to CTRLA.ENABLE.

The TCC is reset by writing '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the TCC, except DBGCTRL, will be reset to their initial state, and the TCC will be disabled. Refer to Control A (31.8.1 CTRLA) register for details.

The TCC should be disabled before the TCC is reset to avoid undefined behavior.

#### 31.6.2.3 Prescaler Selection

The GCLK_TCCx clock is fed into the internal prescaler.

The prescaler consists of a counter that counts up to the selected prescaler value, whereupon the output of the prescaler toggles.

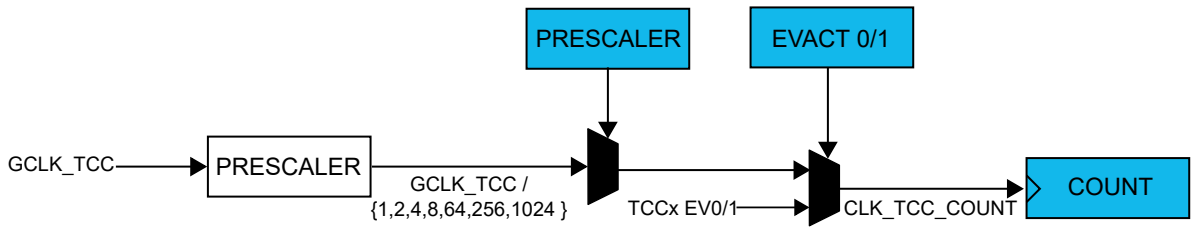
If the prescaler value is higher than one, the counter update condition can be optionally executed on the next GCLK_TCC clock pulse or the next prescaled clock pulse. For further details, refer to the Prescaler (CTRLA.PRESCALER) and Counter Synchronization (CTRLA.PRESYNC) descriptions.

Prescaler outputs from 1 to 1/1024 are available. For a complete list of available prescaler outputs, see the register description for the Prescaler bit group in the Control A register (CTRLA.PRESCALER).

**Note:** When counting events, the prescaler is bypassed.

The joint stream of prescaler ticks and event action ticks is called CLK_TCC_COUNT.

**Figure 31-2. Prescaler**



#### 31.6.2.4 Counter Operation

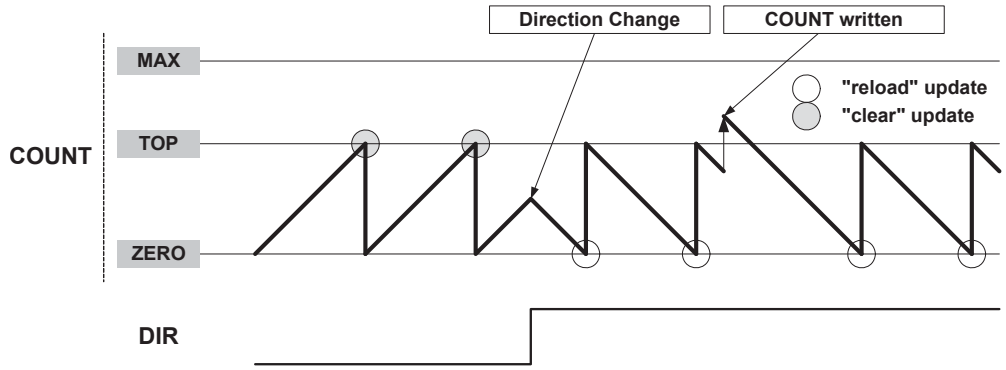
Depending on the mode of operation, the counter is cleared, reloaded, incremented, or decremented at each TCC clock input (CLK_TCC_COUNT). A counter clear or reload mark the end of current counter cycle and the start of a new one.

The counting direction is set by the Direction bit in the Control B register (CTRLB.DIR). If the bit is zero, it's counting up and one if counting down.

The counter will count up or down for each tick (clock or event) until it reaches TOP or ZERO. When it's counting up and TOP is reached, the counter will be set to zero at the next tick (overflow) and the Overflow Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF) will be set. When down-counting, the counter is reloaded with the TOP value when ZERO is reached (underflow), and INTFLAG.OVF is set.

INTFLAG.OVF can be used to trigger an interrupt, a DMA request, or an event. An overflow/underflow occurrence (i.e. a compare match with TOP/ZERO) will stop counting if the One-Shot bit in the Control B register is set (CTRLBSET.ONESHOT). The One-Shot feature is explained in the Additional Features section.

**Figure 31-3. Counter Operation**




It is possible to change the counter value (by writing directly in the COUNT register) even when the counter is running. The COUNT value will always be ZERO or TOP, depending on direction set by CTRLBSET.DIR or CTRLBCLR.DIR, when starting the TCC, unless a different value has been written to it, or the TCC has been stopped at a value other than ZERO. The write access has higher priority than count, clear, or reload. The direction of the counter can also be changed during normal operation. See also Figure 31-3.

**Stop Command**

A stop command can be issued from software by using TCC Command bits in Control B Set register (CTRLBSET.CMD=0x2, STOP).

When a stop is detected while the counter is running, the counter will maintain its current value. If the waveform generation (WG) is used, all waveforms are set to a state defined in Non-Recoverable State x Output Enable bit and Non- Recoverable State x Output Value bit in the Driver Control register (DRVCTRL.NREx and DRVCTRL.NRVx), and the Stop bit in the Status register is set (STATUS.STOP).

**Pause Event Action**

A pause command can be issued when the stop event action is configured in the Input Event Action 1 bits in Event Control register (EVCTRL.EVACT1=0x3, STOP).

When a pause is detected, the counter can stop immediatly maintaining its current value and all waveforms keep their current state, as long as a start event action is detected: Input Event Action 0 bits in Event Control register (EVCTRL.EVACT0=0x3, START).

**Re-Trigger Command and Event Action**

A re-trigger command can be issued from software by using TCC Command bits in Control B Set register (CTRLBSET.CMD=0x1, RETRIGGER), or from event when the re-trigger event action is configured in the Input Event 0/1 Action bits in Event Control register (EVCTRL.EVACTn=0x1, RETRIGGER).

When the command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction (CTRLBSET.DIR or CTRLBCLR.DIR). The Re-Trigger bit in the Interrupt Flag Status and Clear register will be set (INTFLAG.TRG). It is also possible to generate an event by writing a '1' to the Re-Trigger Event Output Enable bit in the Event Control register (EVCTRL.TRGEO). If the re-trigger command is detected when the counter is stopped, the counter will resume counting operation from the value in COUNT.

**Note:**
When a re-trigger event action is configured in the Event Action bits in the Event Control register (EVCTRL.EVACTn=0x1, RETRIGGER), enabling the counter will not start the counter. The counter will start on the next incoming event and restart on corresponding following event.

**Start Event Action**

The start action can be selected in the Event Control register (EVCTRL.EVACT0=0x3, START) and can start the counting operation when previously stopped. The event has no effect if the counter is already counting. When the module is enabled, the counter operation starts when the event is received or when a re-trigger software command is applied.

**Note:**
When a start event action is configured in the Event Action bits in the Event Control register (EVCTRL.EVACT0=0x3, START), enabling the counter will not start the counter. The counter will start on the next incoming event, but it will not restart on subsequent events.

**Count Event Action**

The TCC can count events. When an event is received, the counter increases or decreases the value, depending on direction settings (CTRLBSET.DIR or CTRLBCLR.DIR).

The count event action is selected by the Event Action 0 bit group in the Event Control register (EVCTRL.EVACT0=0x5, COUNT).

**Direction Event Action**

The direction event action can be selected in the Event Control register (EVCTRL.EVACT1=0x2, DIR). When this event is used, the asynchronous event path specified in the event system must be configured or selected. The direction event action can be used to control the direction of the counter operation, depending on external events level. When received, the event level overrides the Direction settings (CTRLBSET.DIR or CTRLBCLR.DIR) and the direction bit value is updated accordingly.

**Increment Event Action**

The increment event action can be selected in the Event Control register (EVCTRL.EVACT0=0x4, INC) and can change the counter state when an event is received. When the TCE0 event (TCCx_EV0) is received, the counter increments, whatever the direction setting (CTRLBSET.DIR or CTRLBCLR.DIR) is.

**Decrement Event Action**

The decrement event action can be selected in the Event Control register (EVCTRL.EVACT1=0x4, DEC) and can change the counter state when an event is received. When the TCE1 (TCCx_EV1) event is received, the counter decrements, whatever the direction setting (CTRLBSET.DIR or CTRLBCLR.DIR) is.

**Non-recoverable Fault Event Action**

Non-recoverable fault actions can be selected in the Event Control register (EVCTRL.EVACTn=0x7, FAULT). When received, the counter will be stopped and the output of the compare channels is overridden according to the Driver Control register settings (DRVCTRL.NREx and DRVCTRL.NRVx). TCE0 and TCE1 must be configured as asynchronous events.

**Event Action Off**

If the event action is disabled (EVCTRL.EVACTn=0x0, OFF), enabling the counter will also start the counter.

**Related Links**

31.6.3.1 One-Shot Operation

#### 31.6.2.5 Compare Operations

By default, the Compare/Capture channel is configured for compare operations. To perform capture operations, it must be re-configured.

When using the TCC with the Compare/Capture Value registers (CCx) for compare operations, the counter value is continuously compared to the values in the CCx registers. This can be used for timer or for waveform operation.

The Channel x Compare/Capture Buffer Value (CCBx) registers provide double buffer capability. The double buffering synchronizes the update of the CCx register with the buffer value at the UPDATE condition or a force update command (CTRLBSET.CMD=0x3, UPDATE). For further details, refer to 31.6.2.6 Double Buffering. The synchronization prevents the occurrence of odd-length, non-symmetrical pulses and ensures glitch-free output.

##### 31.6.2.5.1 Waveform Output Generation Operations

The compare channels can be used for waveform generation on output port pins. To make the waveform available on the connected pin, the following requirements must be fulfilled:

1. Choose a waveform generation mode in the Waveform Generation Operation bit in Waveform register (WAVE.WAVEGEN).
2. Optionally invert the waveform output WO[x] by writing the corresponding Waveform Output x Inversion bit in the Driver Control register (DRVCTRL.INVENx).
3. Configure the pins with the I/O Pin Controller. Refer to *PORT - I/O Pin Controller* for details.

The counter value is continuously compared with each CCx value. On a comparison match, the Match or Capture Channel x bit in the Interrupt Flag Status and Clear register (INTFLAG.MCx) will be set on the next zero-to-one transition of CLK_TCC_COUNT (see Normal Frequency Operation). An interrupt and/or event can be generated on the same condition if Match/Capture occurs, i.e. INTENSET.MCx and/or EVCTRL.MCEOx is '1'. Both interrupt and event can be generated simultaneously. The same condition generates a DMA request.

There are seven waveform configurations for the Waveform Generation Operation bit group in the Waveform register (WAVE.WAVEGEN). This will influence how the waveform is generated and impose restrictions on the top value. The configurations are:

- Normal Frequency (NFRQ)
- Match Frequency (MFRQ)

- Normal Pulse-Width Modulation (NPWM)
- Dual-slope, interrupt/event at TOP (DSTOP)
- Dual-slope, interrupt/event at ZERO (DSBOTTOM)
- Dual-slope, interrupt/event at Top and ZERO (DSBOTH)
- Dual-slope, critical interrupt/event at ZERO (DSCRITICAL)

When using MFRQ configuration, the TOP value is defined by the CC0 register value. For the other waveform operations, the TOP value is defined by the Period (PER) register value.

For dual-slope waveform operations, the update time occurs when the counter reaches ZERO. For the other waveforms generation modes, the update time occurs on counter wraparound, on overflow, underflow, or re-trigger.

The table below shows the update counter and overflow event/interrupt generation conditions in different operation modes.

**Table 31-2. Counter Update and Overflow Event/interrupt Conditions**

| Name | Operation | TOP | Update | Output Waveform | | OVFIF/Event | |
|---|---|---|---|---|---|---|---|
| | | | | On Match | On Update | Up | Down |
| NFRQ | Normal Frequency | PER | TOP/ ZERO | Toggle | Stable | TOP | ZERO |
| MFRQ | Match Frequency | CC0 | TOP/ ZERO | Toggle | Stable | TOP | ZERO |
| NPWM | Single-slope PWM | PER | TOP/ ZERO | See section 'Output Polarity' below | | TOP | ZERO |
| DSCRITICAL | Dual-slope PWM | PER | ZERO | | | - | ZERO |
| DSBOTTOM | Dual-slope PWM | PER | ZERO | | | - | ZERO |
| DSBOTH | Dual-slope PWM | PER | TOP(1) & ZERO | | | TOP | ZERO |
| DSTOP | Dual-slope PWM | PER | ZERO | | | TOP | – |

1. The UPDATE condition on TOP only will occur when circular buffer is enabled for the channel.

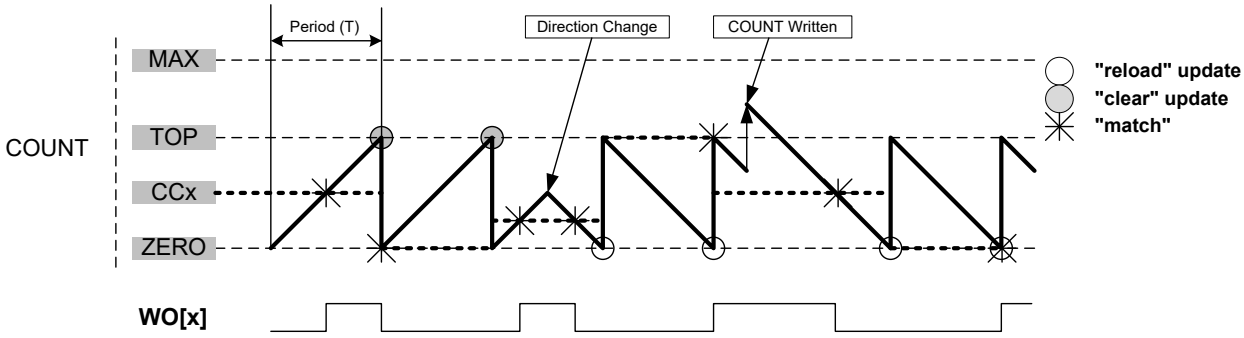
**Related Links**

31.6.3.2 Circular Buffer
23. PORT - I/O Pin Controller

#### 31.6.2.5.2 Normal Frequency (NFRQ)

For Normal Frequency generation, the period time (T) is controlled by the period register (PER). The waveform generation output (WO[x]) is toggled on each compare match between COUNT and CCx, and the corresponding Match or Capture Channel x Interrupt Flag (EVCTRL.MCEOx) will be set.

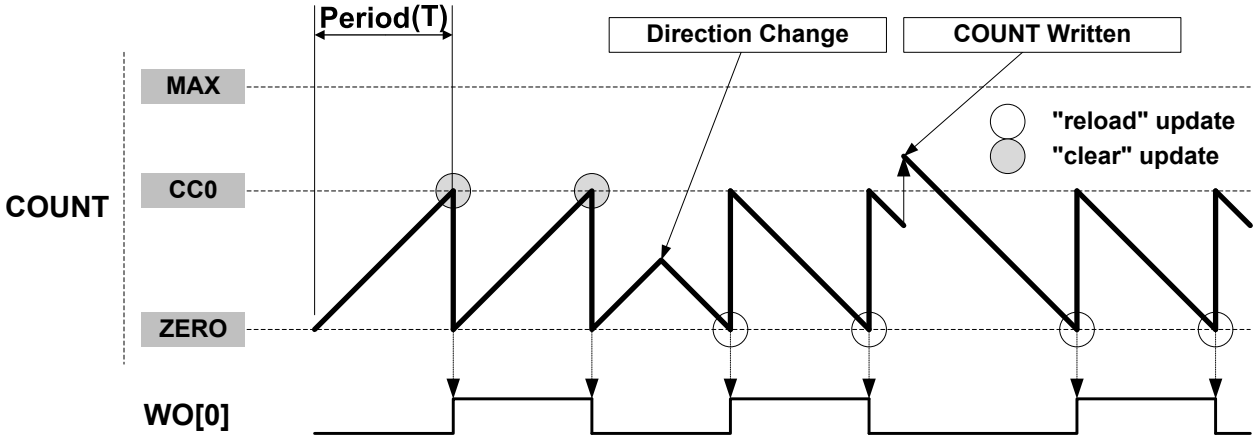
**Figure 31-4. Normal Frequency Operation**




#### 31.6.2.5.3 Match Frequency (MFRQ)

For Match Frequency generation, the period time (T) is controlled by CC0 register instead of PER. WO[0] toggles on each update condition.

**Figure 31-5. Match Frequency Operation**



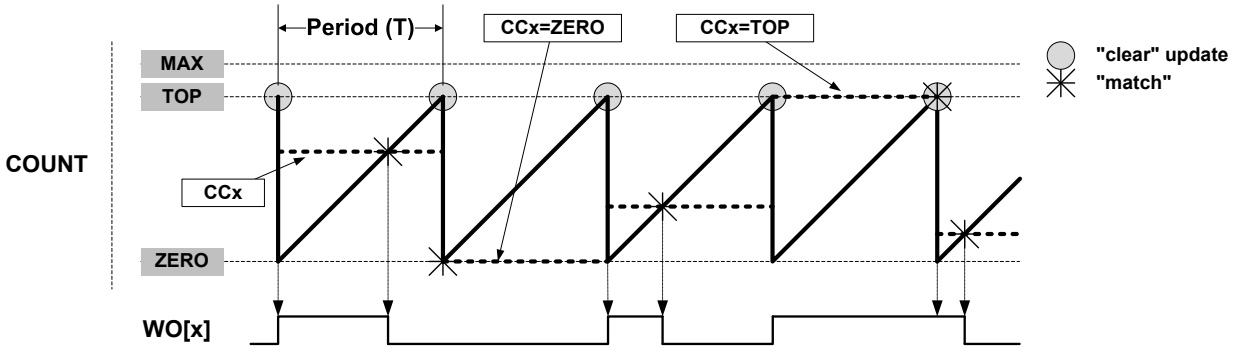

#### 31.6.2.5.4 Normal Pulse-Width Modulation (NPWM)

NPWM uses single-slope PWM generation.

#### 31.6.2.5.5 Single-Slope PWM Operation

For single-slope PWM generation, the period time (T) is controlled by Top value, and CCx controls the duty cycle of the generated waveform output. When up-counting, the WO[x] is set at start or compare match between the COUNT and TOP values, and cleared on compare match between COUNT and CCx register values. When down-counting, the WO[x] is cleared at start or compare match between the COUNT and ZERO values, and set on compare match between COUNT and CCx register values.

**Figure 31-6. Single-Slope PWM Operation**



The following equation calculates the exact resolution for a single-slope PWM ($R_{PWM_SS}$) waveform:

$$R_{PWM_SS} = \frac{\log(TOP+1)}{\log(2)}$$

The PWM frequency depends on the Period register value (PER) and the peripheral clock frequency ($f_{GCLK_TCC}$), and can be calculated by the following equation:

$$f_{PWM_SS} = \frac{f_{GCLK_TCC}}{N(TOP+1)}$$

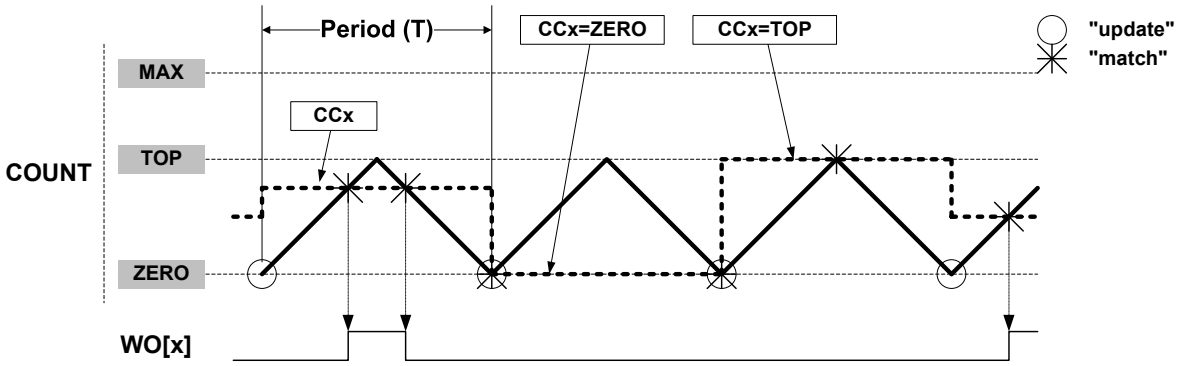
Where N represents the prescaler divider used (1, 2, 4, 8, 16, 64, 256, 1024).

#### 31.6.2.5.6 Dual-Slope PWM Generation

For dual-slope PWM generation, the period setting (TOP) is controlled by PER, while CCx control the duty cycle of the generated waveform output. The figure below shows how the counter repeatedly counts from ZERO to PER and then from PER to ZERO. The waveform generator output is set on compare match when up-counting, and cleared on compare match when down-counting. An interrupt and/or event is generated on TOP (when counting upwards) and/or ZERO (when counting up or down).

In DSBOTH operation, the circular buffer must be enabled to enable the update condition on TOP.

**Figure 31-7. Dual-Slope Pulse Width Modulation**




Using dual-slope PWM results in a lower maximum operation frequency compared to single-slope PWM generation. The period (TOP) defines the PWM resolution. The minimum resolution is 1 bit (TOP=0x00000001).

The following equation calculates the exact resolution for dual-slope PWM ($R_{PWM_DS}$):

$$R_{PWM_DS} = \frac{\log(PER+1)}{\log(2)}.$$

The PWM frequency $f_{PWM_DS}$ depends on the period setting (TOP) and the peripheral clock frequency $f_{GCLK_TCC}$, and can be calculated by the following equation:

$$f_{PWM_DS} = \frac{f_{GCLK_TCC}}{2N \cdot PER}$$

*N* represents the prescaler divider used. The waveform generated will have a maximum frequency of half of the TCC clock frequency ($f_{GCLK_TCC}$) when TOP is set to 0x00000001 and no prescaling is used.

The pulse width ($P_{PWM_DS}$) depends on the compare channel (CCx) register value and the peripheral clock frequency ($f_{GCLK_TCC}$), and can be calculated by the following equation:

$$P_{PWM_DS} = \frac{2N \cdot (TOP - CCx)}{f_{GCLK_TCC}}$$

*N* represents the prescaler divider used.

**Note:** In DSTOP, DSBOTTOM and DSBOTH operation, when TOP is lower than MAX/2, the CCx MSB bit defines the ramp on which the CCx Match interrupt or event is generated. (Rising if CCx[MSB] = 0, falling if CCx[MSB] = 1.)

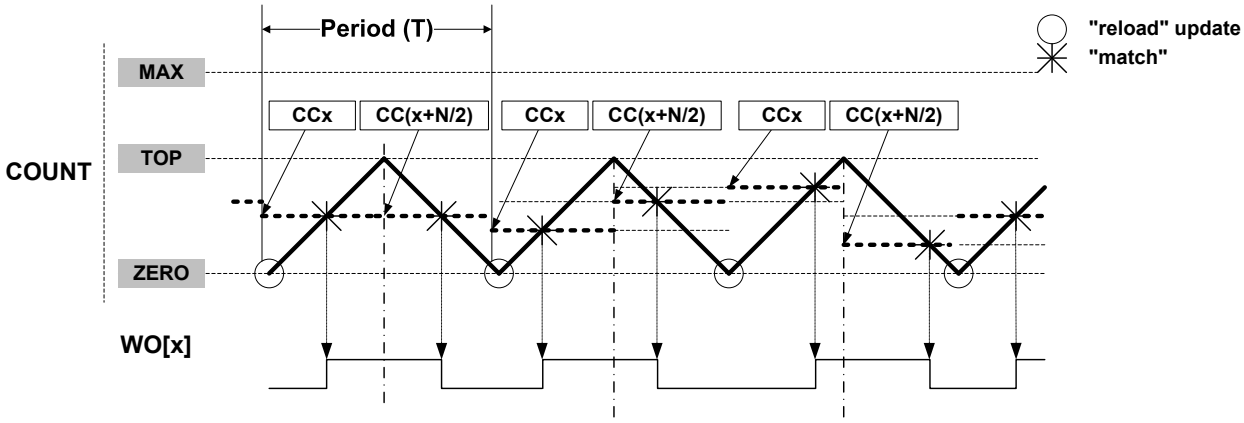
**Related Links**

31.6.3.2 Circular Buffer

#### 31.6.2.5.7 Dual-Slope Critical PWM Generation

Critical mode generation allows generation of non-aligned centered pulses. In this mode, the period time is controlled by PER while CCx control the generated waveform output edge during up-counting and CC(x+CC_NUM/2) control the generated waveform output edge during down-counting.

**Figure 31-8. Dual-Slope Critical Pulse Width Modulation (N=CC_NUM)**




#### 31.6.2.5.8 Output Polarity

The polarity (WAVE.POLx) is available in all waveform output generation. In single-slope and dual-slope PWM operation, it is possible to invert the pulse edge alignment individually on start or end of a PWM cycle for each compare channels. The table below shows the waveform output set/clear conditions, depending on the settings of timer/counter, direction, and polarity.

**Table 31-3. Waveform Generation Set/Clear Conditions**

| Waveform Generation operation | DIR | POLx | Waveform Generation Output Update | |
|---|---|---|---|---|
| | | | Set | Clear |
| Single-Slope PWM | 0 | 0 | Timer/counter matches TOP | Timer/counter matches CCx |
| | | 1 | Timer/counter matches CC | Timer/counter matches TOP |
| | 1 | 0 | Timer/counter matches CC | Timer/counter matches ZERO |
| | | 1 | Timer/counter matches ZERO | Timer/counter matches CC |
| Dual-Slope PWM | x | 0 | Timer/counter matches CC when counting up | Timer/counter matches CC when counting down |
| | | 1 | Timer/counter matches CC when counting down | Timer/counter matches CC when counting up |

In Normal and Match Frequency, the WAVE.POLx value represents the initial state of the waveform output.

#### 31.6.2.6 Double Buffering

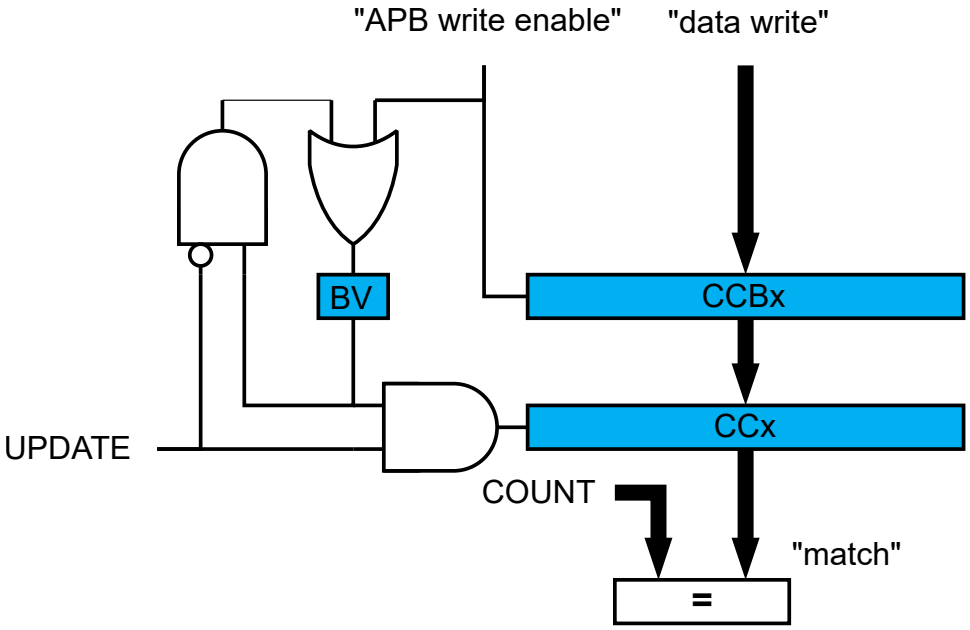
The Pattern (PATT), Waveform (WAVE), Period (PER) and Compare Channels (CCx) registers are all double buffered. Each buffer register has a buffer valid (PATTBV, WAVEBV, PERBV and CCBVx) bit in the STATUS register, which indicates that the buffer register contains a valid value that can be copied into the corresponding register. .

When the buffer valid flag bit in the STATUS register is '1' and the Lock Update bit in the CTRLB register is set to '0', (writing CTRLBCLR.LUPD to '1'), double buffering is enabled: the data from buffer registers will be copied into the corresponding register under hardware UPDATE conditions, then the buffer valid flags bit in the STATUS register are automatically cleared by hardware.

**Note:** Software update command (CTRLBSET.CMD=0x3) act independently of LUPD value.

A compare register is double buffered as in the following figure.

**Figure 31-9. Compare Channel Double Buffering**




Both the registers (PATT/WAVE/PER/CCx) and corresponding buffer registers (PATTB/WAVEBV/PERB/CCBx) are available in the I/O register map, and the double buffering feature is not mandatory. The double buffering is disabled by writing a '1' to CTRLSET.LUPD.

**Note:** In NFRQ, MFRQ or PWM down-counting counter mode (CTRLBSET.DIR=1), when double buffering is enabled (CTRLBCLR.LUPD=1), PERB register is continuously copied into the PER independently of update conditions.

**Changing the Period**

The counter period can be changed by writing a new Top value to the Period register (PER or CC0, depending on the waveform generation mode), any period update on registers (PER or CCx) is effective after the synchronization delay, whatever double buffering enabling is.

**Figure 31-10. Unbuffered Single-Slope Up-Counting Operation**

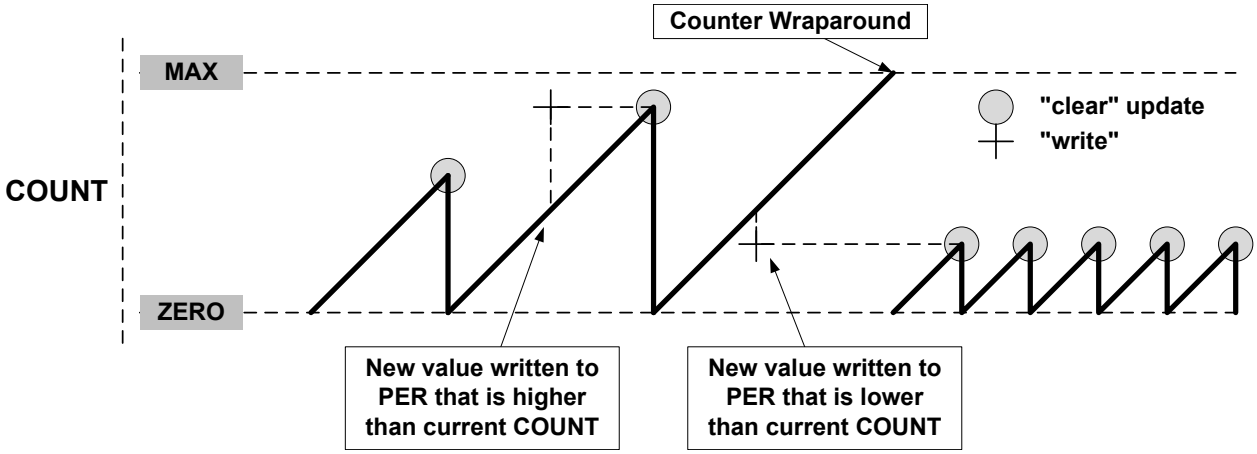
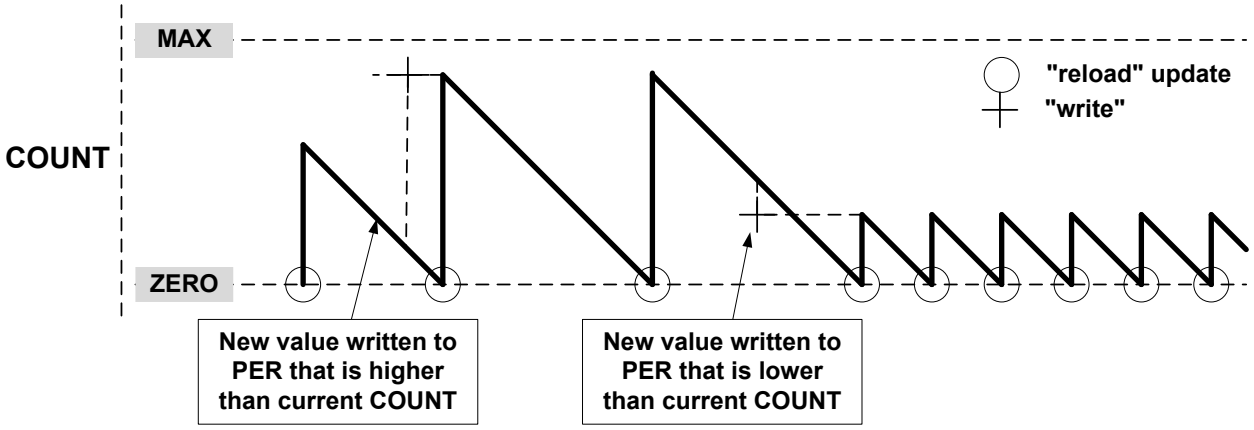
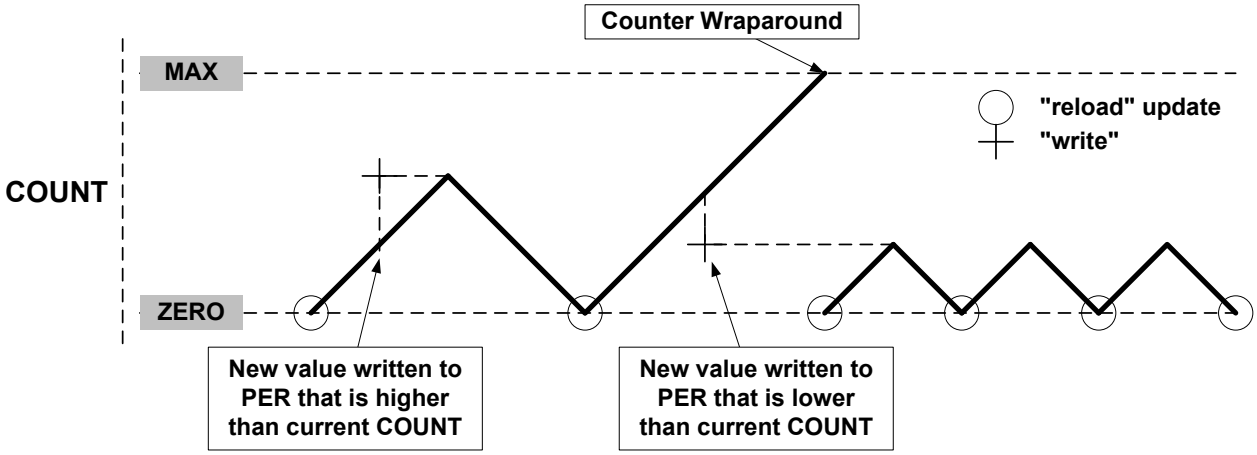



**Figure 31-11. Unbuffered Single-Slope Down-Counting Operation**



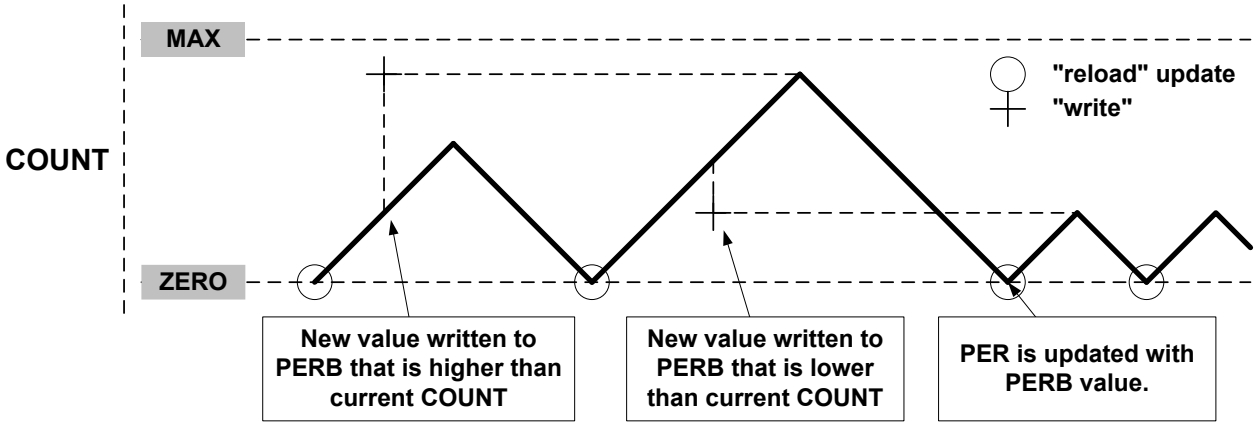

A counter wraparound can occur in any operation mode when up-counting without buffering, see Figure 31-10. COUNT and TOP are continuously compared, so when a new value that is lower than the current COUNT is written to TOP, COUNT will wrap before a compare match.

**Figure 31-12. Unbuffered Dual-Slope Operation**




When double buffering is used, the buffer can be written at any time and the counter will still maintain correct operation. The period register is always updated on the update condition, as shown in Figure 31-13. This prevents wraparound and the generation of odd waveforms.

**Figure 31-13. Changing the Period Using Buffering**



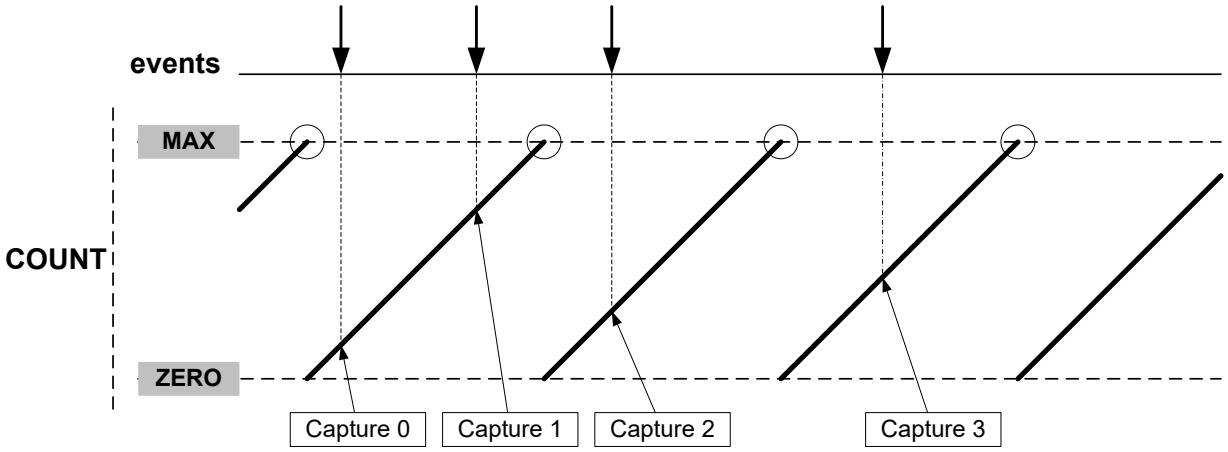

### 31.6.2.7 Capture Operations

To enable and use capture operations, the Match or Capture Channel x Event Input Enable bit in the Event Control register (EVCTRL.MCEIx) must be written to '1'. The capture channels to be used must also be enabled in the Capture Channel x Enable bit in the Control A register (CTRLA.CPTENx) before capturing can be performed.

**Event Capture Action**

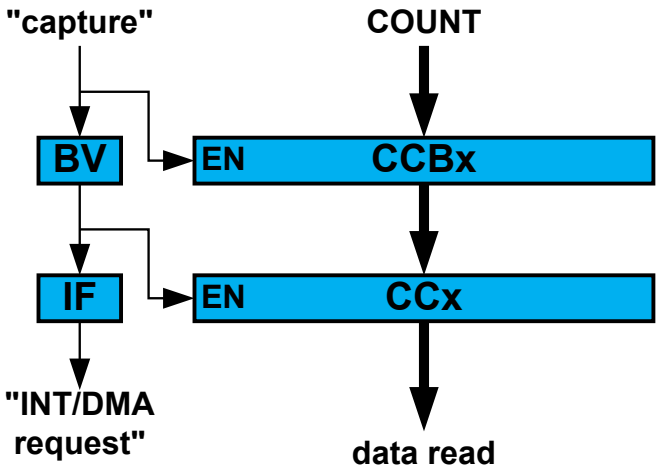
The compare/capture channels can be used as input capture channels to capture events from the Event System, and give them a timestamp. The following figure shows four capture events for one capture channel.

**Figure 31-14. Input Capture Timing**




For input capture, the buffer register and the corresponding CCx act like a FIFO. When CCx is empty or read, any content in CCBx is transferred to CCx. The buffer valid flag is passed to set the CCx interrupt flag (IF) and generate the optional interrupt, event or DMA request. CCBx register value can't be read, all captured data must be read from CCx register.

**Figure 31-15. Capture Double Buffering**



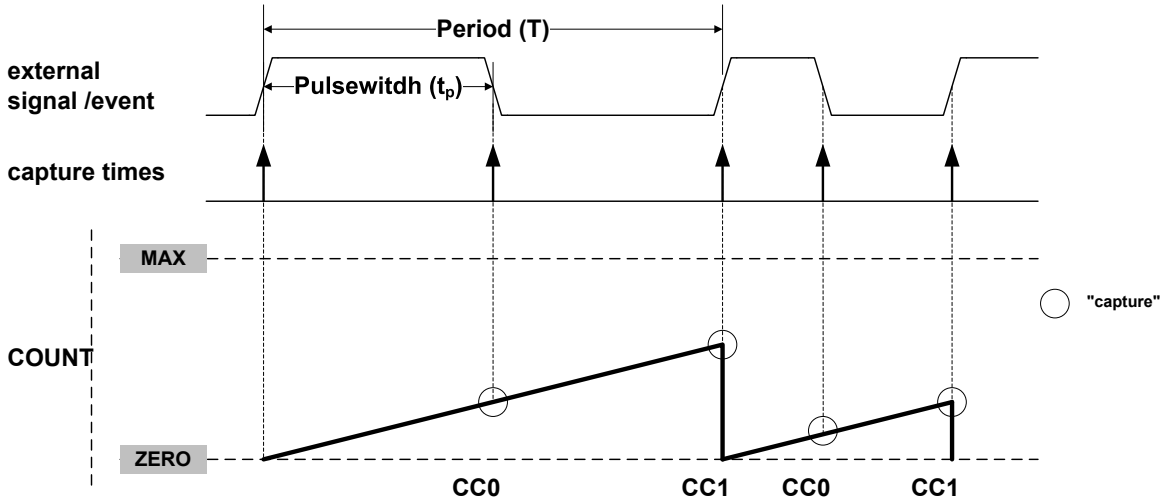

The TCC can detect capture overflow of the input capture channels: When a new capture event is detected while the Capture Buffer Valid flag (STATUS.CCBV) is still set, the new timestamp will not be stored and INTFLAG.ERR will be set.

**Period and Pulse-Width (PPW) Capture Action**

The TCC can perform two input captures and restart the counter on one of the edges. This enables the TCC to measure the pulse-width and period and to characterize the frequency *f* and *dutyCycle* of an input signal:

$$f = \frac{1}{T} \quad , \quad dutyCycle = \frac{t_p}{T}$$

**Figure 31-16. PWP Capture**




Selecting PWP or PPW in the Timer/Counter Event Input 1 Action bit group in the Event Control register (EVCTRL.EVACT1) enables the TCC to perform one capture action on the rising edge and the other one on the falling edge. When using PPW (period and pulse-width) event action, period *T* will be captured into CC0 and the pulse-width $t_p$ into CC1. The PWP (Pulse-width and Period) event action offers the same functionality, but *T* will be captured into CC1 and $t_p$ into CC0.

The Timer/Counter Event x Invert Enable bit in Event Control register (EVCTRL.TCEINVx) is used for event source x to select whether the wraparound should occur on the rising edge or the falling edge. If EVCTRL.TCEINVx=1, the wraparound will happen on the falling edge.

The corresponding capture is done only if the channel is enabled in capture mode (CTRLA.CPTENx=1). If not, the capture action will be ignored and the channel will be enabled in compare mode of operation. When only one of these channel is required, the other channel can be used for other purposes.

The TCC can detect capture overflow of the input capture channels: When a new capture event is detected while the INTFLAG.MCx is still set, the new timestamp will not be stored and INTFLAG.ERR will be set.

**Note:** When up-counting (CTRLBSET.DIR=0), counter values lower than 1 cannot be captured in Capture Minimum mode (FCTRLn.CAPTURE=CAPTMIN). To capture the full range including value 0, the TCC must be configured in down-counting mode (CTRLBSET.DIR=0).

**Note:** In dual-slope PWM operation, and when TOP is lower than MAX/2, the CCx MSB captures the CTRLB.DIR state to identify the ramp on which the capture has been done. For rising ramps CCx[MSB] is zero, for falling ramps CCx[MSB]=1.

### 31.6.3 Additional Features

#### 31.6.3.1 One-Shot Operation

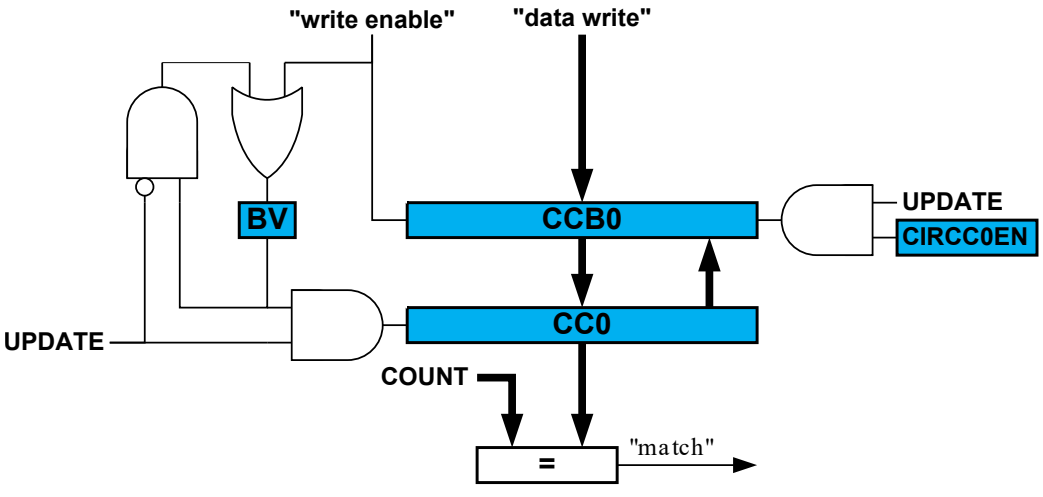
When one-shot is enabled, the counter automatically stops on the next counter overflow or underflow condition. When the counter is stopped, the Stop bit in the Status register (STATUS.STOP) is set and the waveform outputs are set to the value defined by DRVCTRL.NREx and DRVCTRL.NRVx.

One-shot operation can be enabled by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) and disabled by writing a '1' to CTRLBCLR.ONESHOT. When enabled, the TCC will count until an overflow or underflow occurs and stop counting. The one-shot operation can be restarted by a re-trigger software command, a re-trigger event or a start event. When the counter restarts its operation, STATUS.STOP is automatically cleared.

#### 31.6.3.2 Circular Buffer

The Period register (PER) and the compare channels register (CC0 to CC3) support circular buffer operation. When circular buffer operation is enabled, the PER or CCx values are copied into the corresponding buffer registers at each update condition. Circular buffering is dedicated to RAMP2, RAMP2A, and DSBOTH operations.

**Figure 31-17. Circular Buffer on Channel 0**




#### 31.6.3.3 Dithering Operation

The TCC supports dithering on Pulse-width or Period on a 16, 32 or 64 PWM cycles frame.

Dithering consists in adding some extra clocks cycles in a frame of several PWM cycles, and can improve the accuracy of the *average* output pulse width and period. The extra clock cycles are added on some of the compare match signals, one at a time, through a "blue noise" process that minimizes the flickering on the resulting dither patterns.

Dithering is enabled by writing the corresponding configuration in the Enhanced Resolution bits in CTRLA register (CTRLA.RESOLUTION):

- DITH4 enable dithering every 16 PWM frames
- DITH5 enable dithering every 32 PWM frames
- DITH6 enable dithering every 64 PWM frames

The DITHERCY bits of COUNT, PER and CCx define the number of extra cycles to add into the frame (DITHERCY bits from the respective COUNT, PER or CCx registers). The remaining bits of COUNT, PER, CCx define the compare value itself.

The pseudo code, giving the extra cycles insertion regarding the cycle is:

```
int extra_cycle(resolution, dithercy, cycle){
  int MASK;
  int value
  switch (resolution){
    DITH4: MASK = 0x0f;
    DITH5: MASK = 0x1f;
    DITH6: MASK = 0x3f;
  }
  value = cycle * dithercy;
  if (((MASK & value) + dithercy) > MASK)
    return 1;
 return 0;
}
```

**Dithering on Period**

Writing DITHERCY in PER will lead to an average PWM period configured by the following formulas.

DITH4 mode:

$$PwmPeriod = \left(\frac{\mathrm{DITHERCY}}{16} + \mathrm{PER}\right)\left(\frac{1}{f_{\mathrm{GCLK_TCC}}}\right)$$

**Note:** If DITH4 mode is enabled, the last 4 significant bits from PER/CCx or COUNT register correspond to the DITHERCY value, rest of the bits corresponds to PER/CCx or COUNT value.

DITH5 mode:

$$PwmPeriod = \left(\frac{\mathrm{DITHERCY}}{32} + \mathrm{PER}\right)\left(\frac{1}{f_{\mathrm{GCLK_TCC}}}\right)$$

DITH6 mode:

$$PwmPeriod = \left(\frac{\mathrm{DITHERCY}}{64} + \mathrm{PER}\right)\left(\frac{1}{f_{\mathrm{GCLK_TCC}}}\right)$$

**Dithering on Pulse Width**

Writing DITHERCY in CCx will lead to an average PWM pulse width configured by the following formula.

DITH4 mode:

$$PwmPulseWidth = \left(\frac{\mathrm{DITHERCY}}{16} + \mathrm{CCx}\right)\left(\frac{1}{f_{\mathrm{GCLK_TCC}}}\right)$$

DITH5 mode:

$$PwmPulseWidth = \left(\frac{\text{DITHERCY}}{32} + \text{CCx}\right)\left(\frac{1}{f_{\text{GCLK_TCC}}}\right)$$

DITH6 mode:

$$PwmPulseWidth = \left(\frac{\text{DITHERCY}}{64} + \text{CCx}\right)\left(\frac{1}{f_{\text{GCLK_TCC}}}\right)$$

**Note:** The PWM period will remain static in this case.

#### 31.6.3.4 Ramp Operations

Three ramp operation modes are supported. All of them require the timer/counter running in single-slope PWM generation. The ramp mode is selected by writing to the Ramp Mode bits in the Waveform Control register (WAVE.RAMP).

**RAMP1 Operation**

This is the default PWM operation, described in Single-Slope PWM Generation.

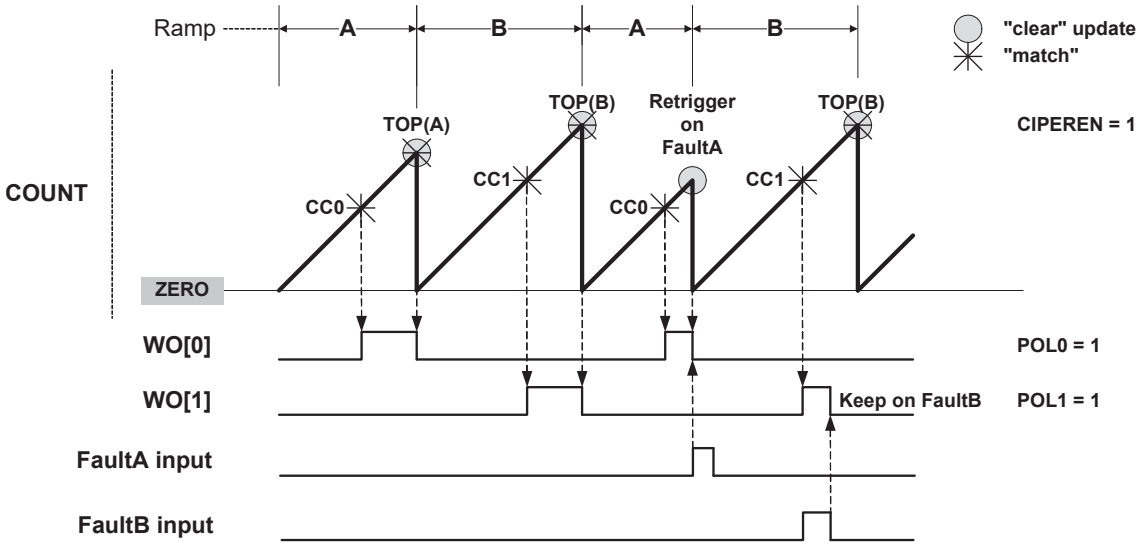
**RAMP2 Operation**

These operation modes are dedicated for power factor correction (PFC), Half-Bridge and Push-Pull SMPS topologies, where two consecutive timer/counter cycles are interleaved, see Figure 31-18. In cycle A, odd channel output is disabled, and in cycle B, even channel output is disabled. The ramp index changes after each update, but can be software modified using the Ramp index command bits in Control B Set register (CTRLBSET.IDXCMD).

**Standard RAMP2 (RAMP2) Operation**

Ramp A and B periods are controlled by the PER register value. The PER value can be different on each ramp by the Circular Period buffer option in the Wave register (WAVE.CIPEREN=1). This mode uses a two-channel TCC to generate two output signals, or one output signal with another CC channel enabled in capture mode.

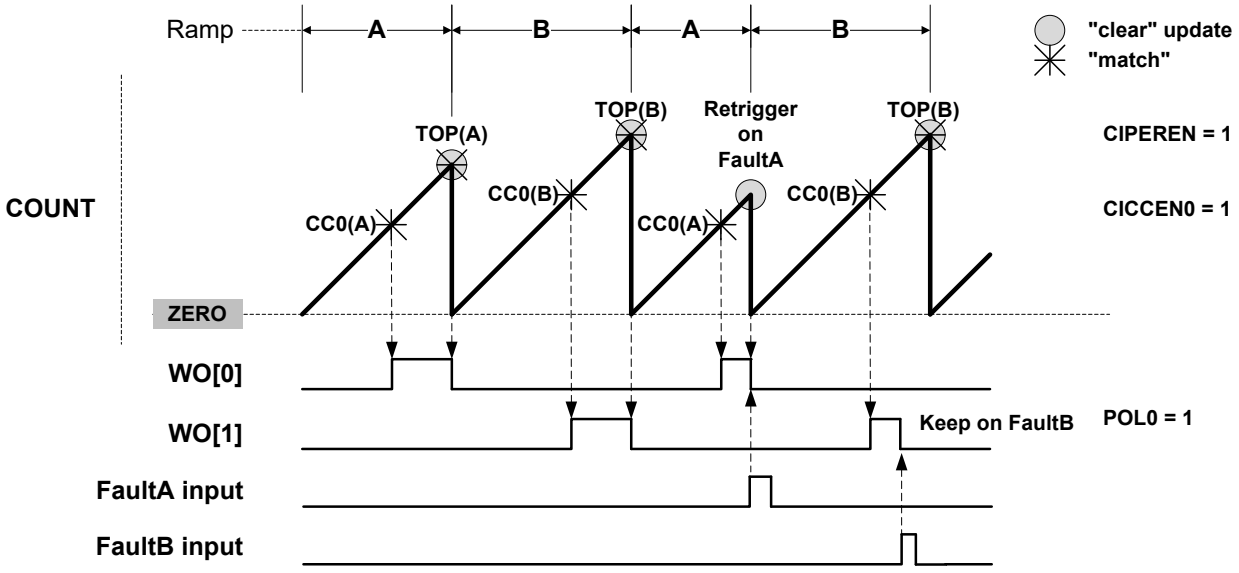
**Figure 31-18. RAMP2 Standard Operation**



### Alternate RAMP2 (RAMP2A) Operation

Alternate RAMP2 operation is similar to RAMP2, but CC0 controls both WO[0] and WO[1] waveforms when the corresponding circular buffer option is enabled (CIPEREN=1). The waveform polarity is the same on both outputs. Channel 1 can be used in capture mode.

**Figure 31-19. RAMP2 Alternate Operation**




### Critical RAMP2 (RAMP2C) Operation

Critical RAMP2 operation provides a way to cover RAMP2 operation requirements without the update constraint associated with the use of circular buffers. In this mode, CC0 is controlling the period of ramp A and PER is controlling the period of ramp B. When using more than two channels, WO[0] output is controlled by CC2 (HIGH) and CC0 (LOW). On TCC with 2 channels, a pulse on WO[0] will last the entire period of ramp A, if WAVE.POL0=0.

**Figure 31-20. RAMP2 Critical Operation With More Than 2 Channels**

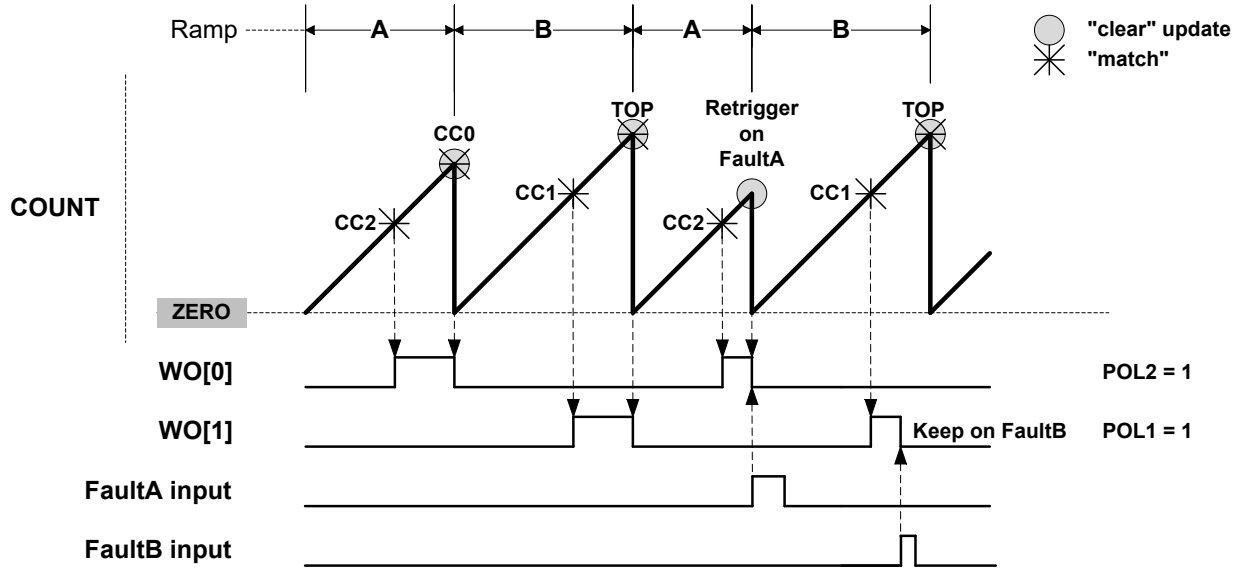
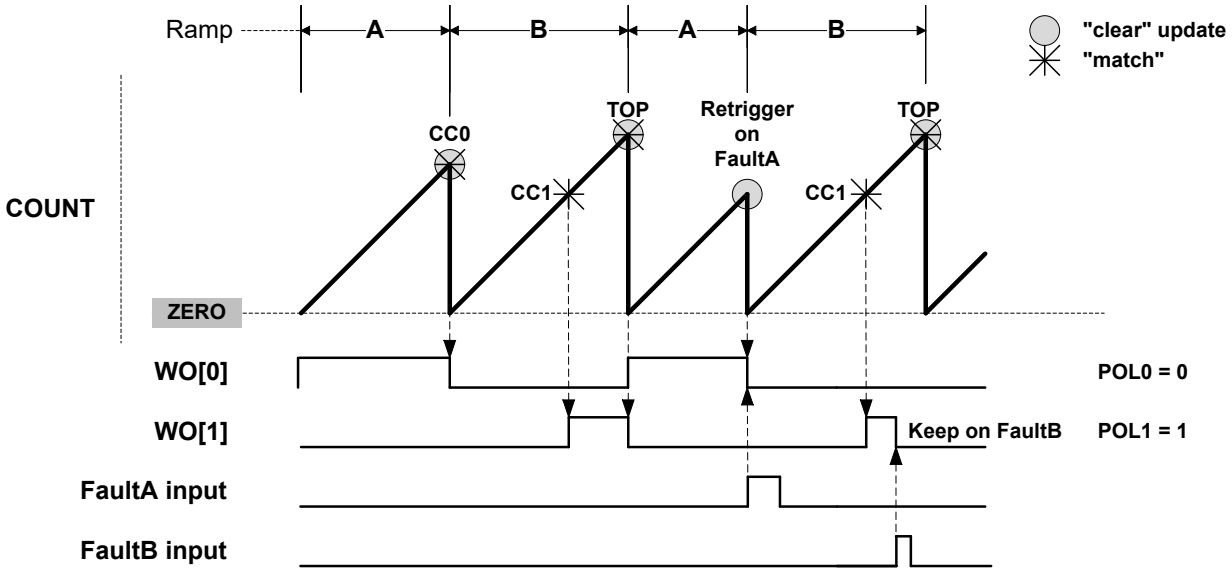


**Figure 31-21. RAMP2 Critical Operation With 2 Channels**




### 31.6.3.5 Recoverable Faults

Recoverable faults can restart or halt the timer/counter. Two faults, called Fault A and Fault B, can trigger recoverable fault actions on the compare channels CC0 and CC1 of the TCC. The compare channels' outputs can be clamped to inactive state either as long as the fault condition is present, or from the first valid fault condition detection on until the end of the timer/counter cycle.

**Fault Inputs**

The first two channel input events (TCCxMC0 and TCCxMC1) can be used as Fault A and Fault B inputs, respectively. Event system channels connected to these fault inputs must be configured as asynchronous. The TCC must work in a PWM mode.

**Fault Filtering**

There are three filters available for each input Fault A and Fault B. They are configured by the corresponding Recoverable Fault n Configuration registers (FCTRLA and FCTRLB). The three filters can either be used independently or in any combination.

**Input Filtering** By default, the event detection is asynchronous. When the event occurs, the fault system will immediately and asynchronously perform the selected fault action on the compare channel output, also in device power modes where the clock is not available. To avoid false fault detection on external events (e.g. due to a glitch on an I/O port) a digital filter can be enabled and configured by the Fault B Filter Value bits in the Fault n Configuration registers (FCTRLn.FILTERVAL). If the event width is less than FILTERVAL (in clock cycles), the event will be discarded. A valid event will be delayed by FILTERVAL clock cycles.

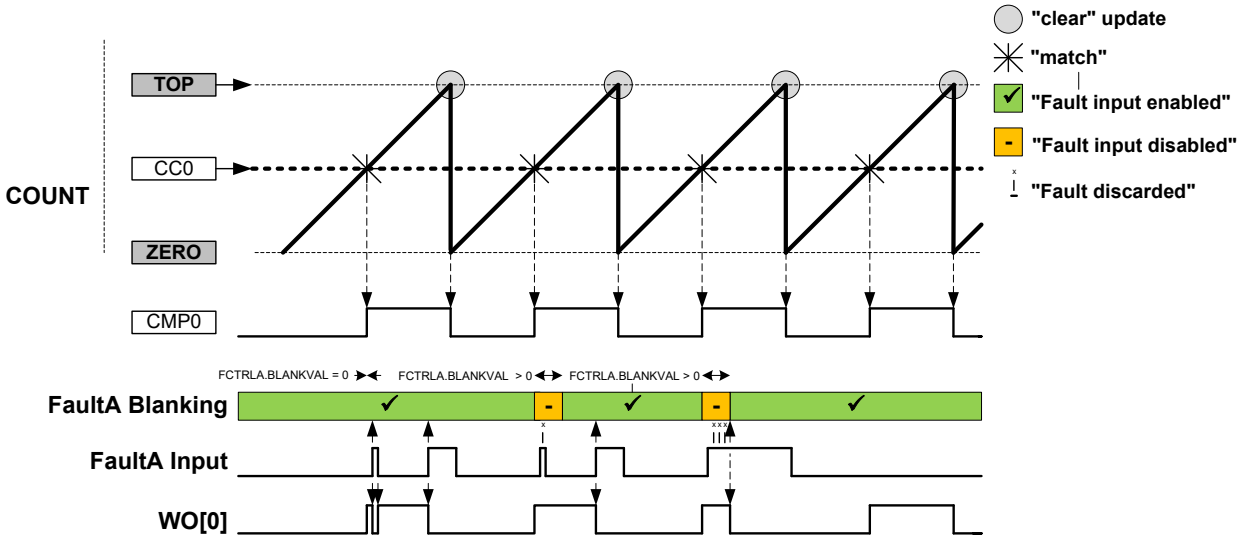
**Fault Blanking** This ignores any fault input for a certain time just after a selected waveform output edge. This can be used to prevent false fault triggering due to signal bouncing, as shown in the figure below. Blanking can be enabled by writing an edge triggering configuration to the Fault n Blanking Mode bits in the Recoverable Fault n Configuration register (FCTRLn.BLANK). The desired duration of the blanking must be written to the Fault n Blanking Time bits (FCTRLn.BLANKVAL).
The blanking time $t_b$is calculated by

$$t_b = \frac{1 + \text{BLANKVAL}}{f_{\text{GCLK_TCCx_PRESC}}}$$

Here, $f_{\text{GCLK_TCCx_PRESC}}$ is the frequency of the prescaled peripheral clock frequency $f_{\text{GCLK_TCCx}}$.

The maximum blanking time (FCTRLn.BLANKVAL= 255) at $f_{\text{GCLK_TCCx}}$=96MHz is 2.67µs (no prescaler) or 170µs (prescaling). For $f_{\text{GCLK_TCCx}}$=1MHz, the maximum blanking time is either 170µs (no prescaling) or 10.9ms (prescaling enabled).

**Figure 31-22. Fault Blanking in RAMP1 Operation with Inverted Polarity**




**Fault Qualification**

This is enabled by writing a '1' to the Fault n Qualification bit in the Recoverable Fault n Configuration register (FCTRLn.QUAL). When the recoverable fault qualification is enabled (FCTRLn.QUAL=1), the fault input is disabled all the time the corresponding channel output has an inactive level, as shown in the figures below.

**Figure 31-23. Fault Qualification in RAMP1 Operation**

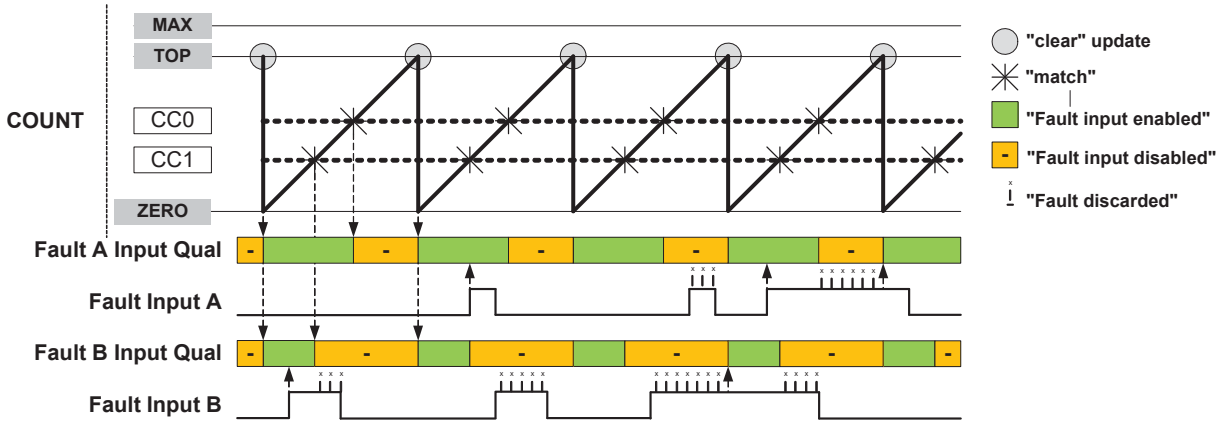
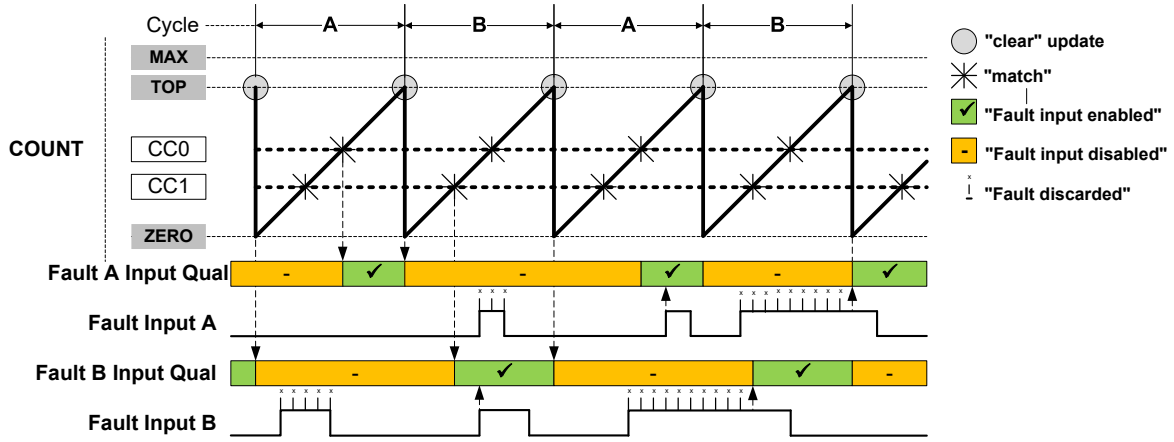


**Figure 31-24. Fault Qualification in RAMP2 Operation with Inverted Polarity**

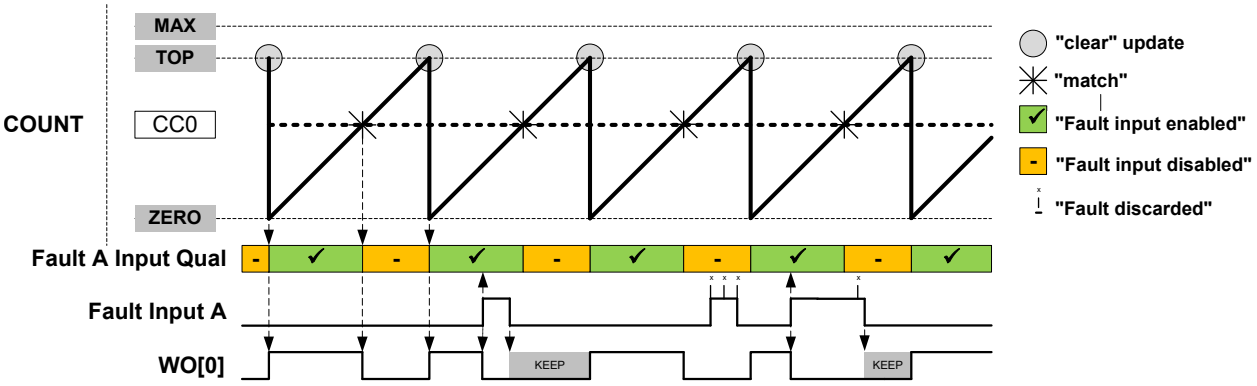



## Fault Actions

Different fault actions can be configured individually for Fault A and Fault B. Most fault actions are not mutually exclusive; hence two or more actions can be enabled at the same time to achieve a result that is a combination of fault actions.

**Keep Action** This is enabled by writing the Fault n Keeper bit in the Recoverable Fault n Configuration register (FCTRLn.KEEP) to '1'. When enabled, the corresponding channel output will be clamped to zero as long as the fault condition is present. The clamp will be released on the start of the first cycle after the fault condition is no longer present, see next Figure.

**Figure 31-25. Waveform Generation with Fault Qualification and Keep Action**




**Restart Action** This is enabled by writing the Fault n Restart bit in Recoverable Fault n Configuration register (FCTRLn.RESTART) to '1'. When enabled, the timer/counter will be restarted as soon as the corresponding fault condition is present. The ongoing cycle is stopped and the timer/counter starts a new cycle, see Figure 31-26. In Ramp 1 mode, when the new cycle starts, the compare outputs will be clamped to inactive level as long as the fault condition is present.

**Note:** For RAMP2 operation, when a new timer/counter cycle starts the cycle index will change automatically, see Figure 31-27. Fault A and Fault B are qualified only during the cycle A and cycle B respectively: Fault A is disabled during cycle B, and Fault B is disabled during cycle A.

**Figure 31-26. Waveform Generation in RAMP1 mode with Restart Action**

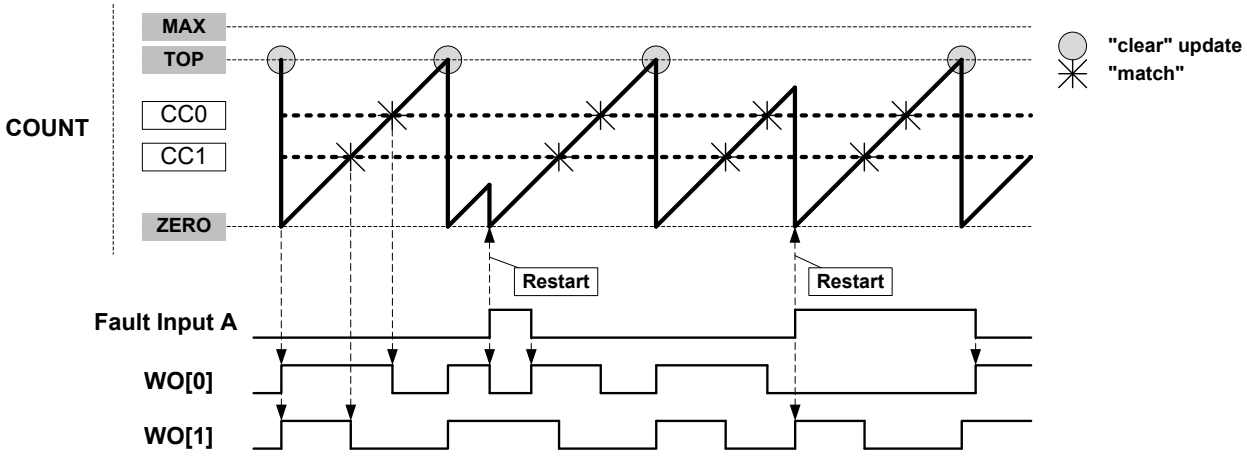
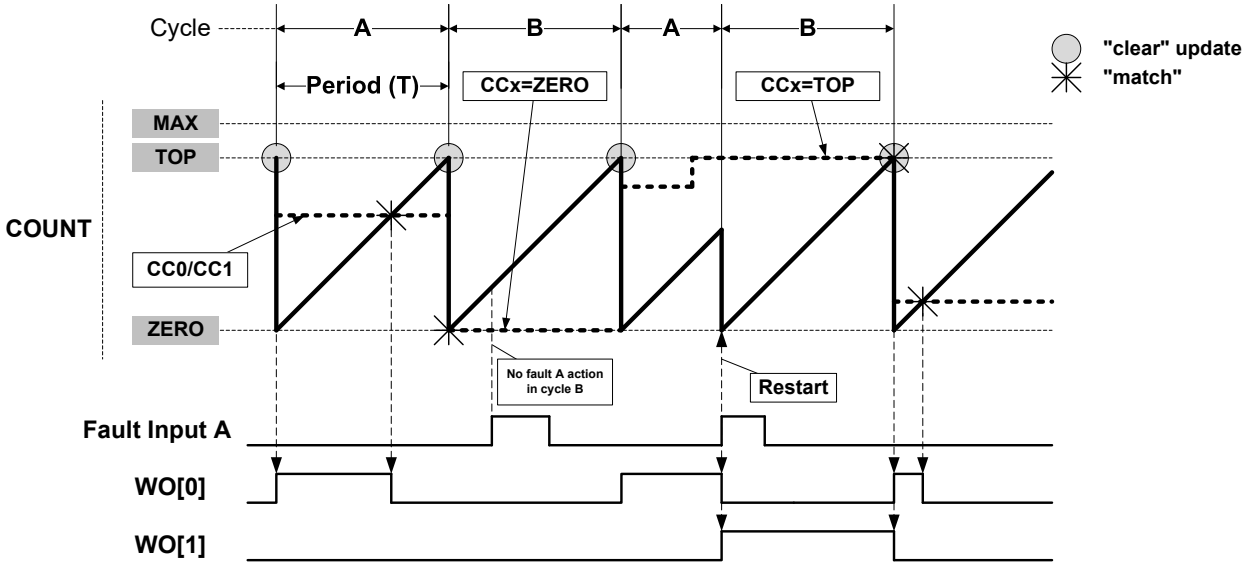



**Figure 31-27. Waveform Generation in RAMP2 mode with Restart Action**




**Capture Action**

Several capture actions can be selected by writing the Fault n Capture Action bits in the Fault n Control register (FCTRLn.CAPTURE). When one of the capture operations is selected, the counter value is captured when the fault occurs. These capture operations are available:

- CAPT - the equivalent to a standard capture operation, for further details refer to 31.6.2.7 Capture Operations
- CAPTMIN - gets the minimum time stamped value: on each new local minimum captured value, an event or interrupt is issued.
- CAPTMAX - gets the maximum time stamped value: on each new local maximum captured value, an event or interrupt (IT) is issued, see Figure 31-28.
- LOCMIN - notifies by event or interrupt when a local minimum captured value is detected.
- LOCMAX - notifies by event or interrupt when a local maximum captured value is detected.
- DERIV0 - notifies by event or interrupt when a local extreme captured value is detected, see Figure 31-29.

*CCx Content:*

In CAPTMIN and CAPTMAX operations, CCx keeps the respective extremum captured values, see Figure 31-28. In LOCMIN, LOCMAX or DERIV0 operation, CCx follows the counter value at fault time, see Figure 31-29.

Before enabling CAPTMIN or CAPTMAX mode of capture, the user must initialize the corresponding CCx register value to a value different from zero (for CAPTMIN) top (for CAPTMAX). If the CCx register initial value is zero (for CAPTMIN) top (for CAPTMAX), no captures will be performed using the corresponding channel.

*MCx Behaviour:*

In LOCMIN and LOCMAX operation, capture is performed on each capture event. The MCx interrupt flag is set only when the captured value is above or equal (for LOCMIN) or below or equal (for LOCMAX) to the previous captured value. So interrupt flag is set when a new relative local Minimum (for CAPTMIN) or Maximum (for CAPTMAX) value has been detected. DERIV0 is equivalent to an OR function of (LOCMIN, LOCMAX).

In CAPT operation, capture is performed on each capture event. The MCx interrupt flag is set on each new capture.

In CAPTMIN and CAPTMAX operation, capture is performed only when on capture event time, the counter value is lower (for CAPTMIN) or higher (for CAPMAX) than the last captured value. The MCx interrupt flag is set only when on capture event time, the counter value is higher or equal (for CAPTMIN) or lower or equal (for CAPTMAX) to the value captured on the previous event. So interrupt flag is set when a new absolute local Minimum (for CAPTMIN) or Maximum (for CAPTMAX) value has been detected.

*Interrupt Generation*

In CAPT mode, an interrupt is generated on each filtered Fault n and each dedicated CCx channel capture counter value. In other modes, an interrupt is only generated on an extreme captured value.

**Figure 31-28. Capture Action “CAPTMAX”**

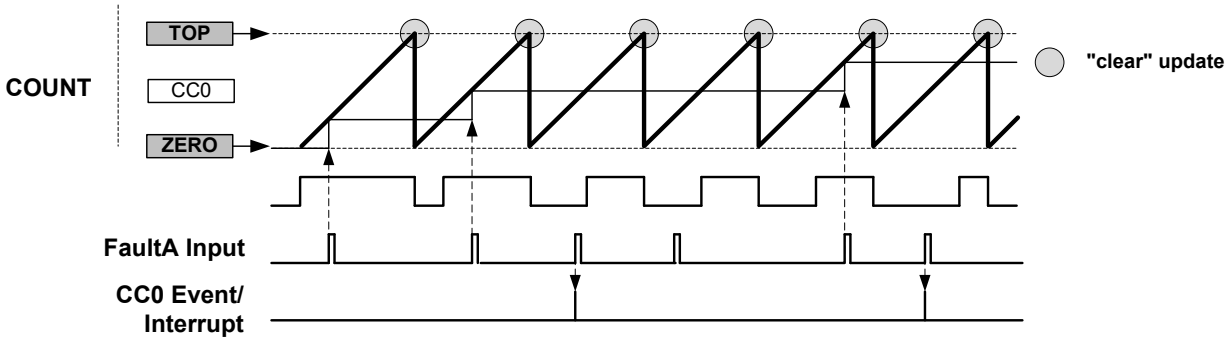
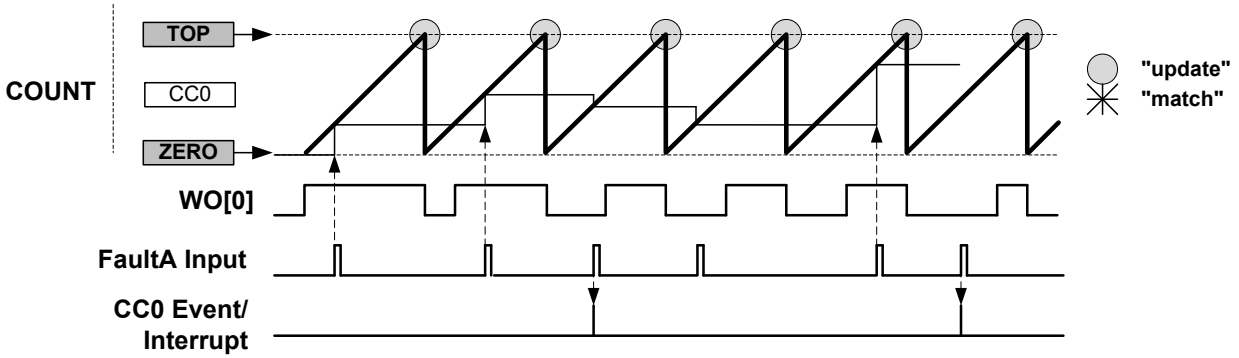


**Figure 31-29. Capture Action “DERIV0”**




**Hardware Halt Action**

This is configured by writing 0x1 to the Fault n Halt mode bits in the Recoverable Fault n Configuration register (FCTRLn.HALT). When enabled, the timer/counter is halted and the cycle is extended as long as the corresponding fault is present.

The next figure ('Waveform Generation with Halt and Restart Actions') shows an example where both restart action and hardware halt action are enabled for Fault A. The compare channel 0 output is clamped to inactive level as long as the timer/counter is halted. The timer/counter resumes the counting operation as soon as the fault condition is no longer present. As the restart action is enabled in this example, the timer/counter is restarted after the fault condition is no longer present.

The figure after that ('Waveform Generation with Fault Qualification, Halt, and Restart Actions') shows a similar example, but with additionally enabled fault qualification. Here, counting is resumed after the fault condition is no longer present.

Note that in RAMP2 and RAMP2A operations, when a new timer/counter cycle starts, the cycle index will automatically change.

**Figure 31-30. Waveform Generation with Halt and Restart Actions**

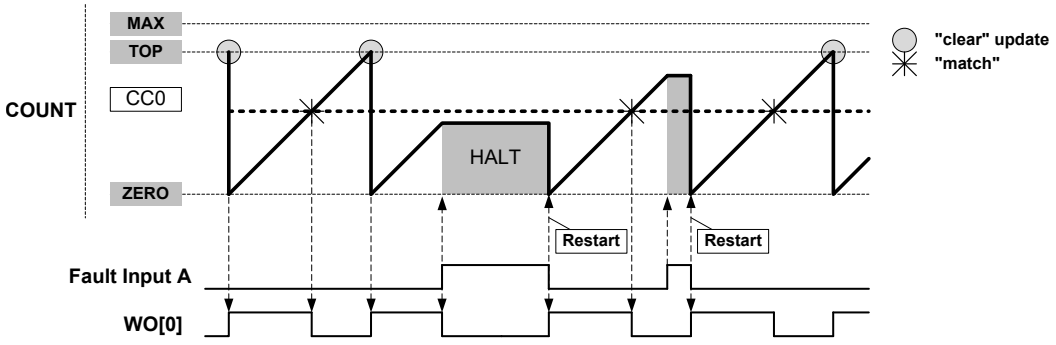
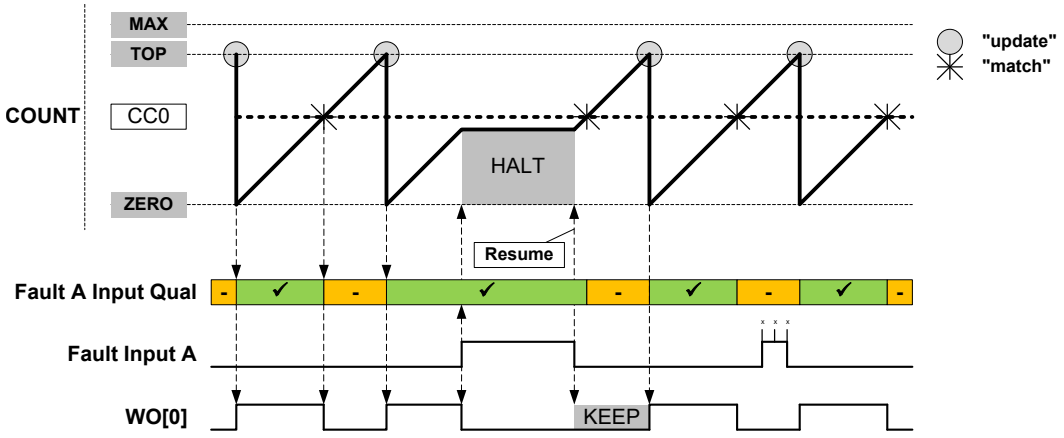


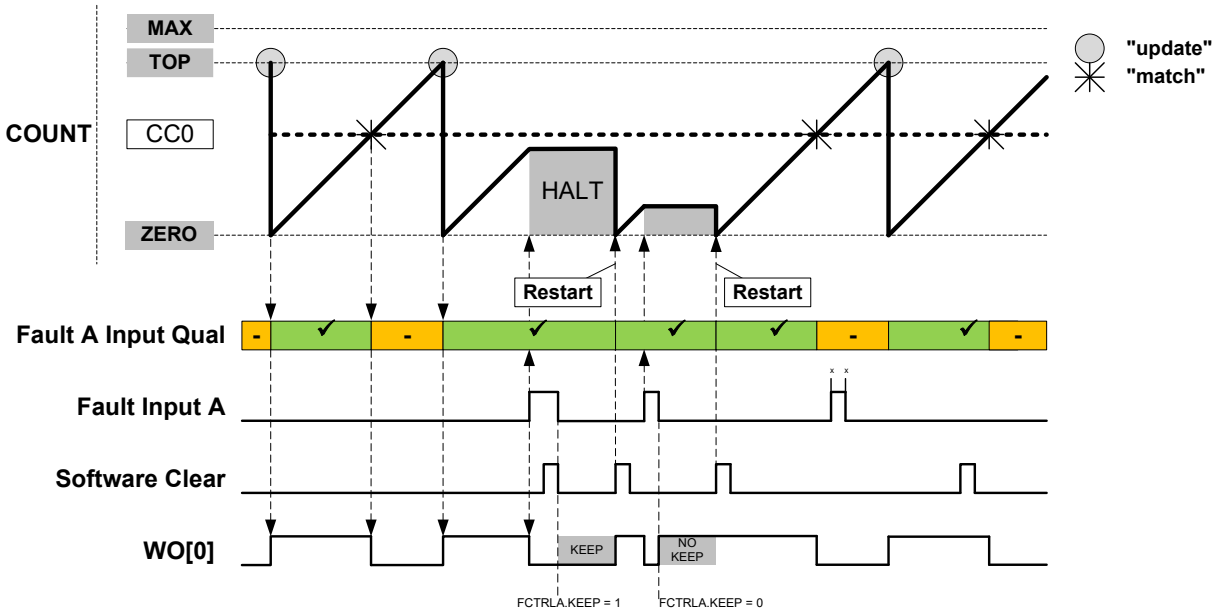
**Figure 31-31. Waveform Generation with Fault Qualification, Halt, and Restart Actions**




**Software Halt Action**

This is configured by writing 0x2 to the Fault n Halt mode bits in the Recoverable Fault n configuration register (FCTRLn.HALT). Software halt action is similar to hardware halt action, but in order to restart the timer/counter, the corresponding fault condition must not be present anymore, and the corresponding FAULT n bit in the STATUS register must be cleared by software.

**Figure 31-32. Waveform Generation with Software Halt, Fault Qualification, Keep and Restart Actions**




### 31.6.3.6 Non-Recoverable Faults

The non-recoverable fault action will force all the compare outputs to a pre-defined level programmed into the Driver Control register (DRVCTRL.NRE and DRVCTRL.NRV). The non-recoverable fault input (EV0 and EV1) actions are enabled in Event Control register (EVCTRL.EVACT0 and EVCTRL.EVACT1).

To avoid false fault detection on external events (e.g. a glitch on an I/O port) a digital filter can be enabled using Non-Recoverable Fault Input x Filter Value bits in the Driver Control register

(DRVCTRL.FILTERVALn). Therefore, the event detection is synchronous, and event action is delayed by the selected digital filter value clock cycles.

When the Fault Detection on Debug Break Detection bit in Debug Control register (DGBCTRL.FDDBD) is written to '1', a non-recoverable Debug Faults State and an interrupt (DFS) is generated when the system goes in debug operation.

In RAMP2, RAMP2A, or DSBOTH operation, when the Lock Update bit in the Control B register is set by writing CTRLBSET.LUPD=1 and the ramp index or counter direction changes, a non-recoverable Update Fault State and the respective interrupt (UFS) are generated.

#### 31.6.3.7 Waveform Extension

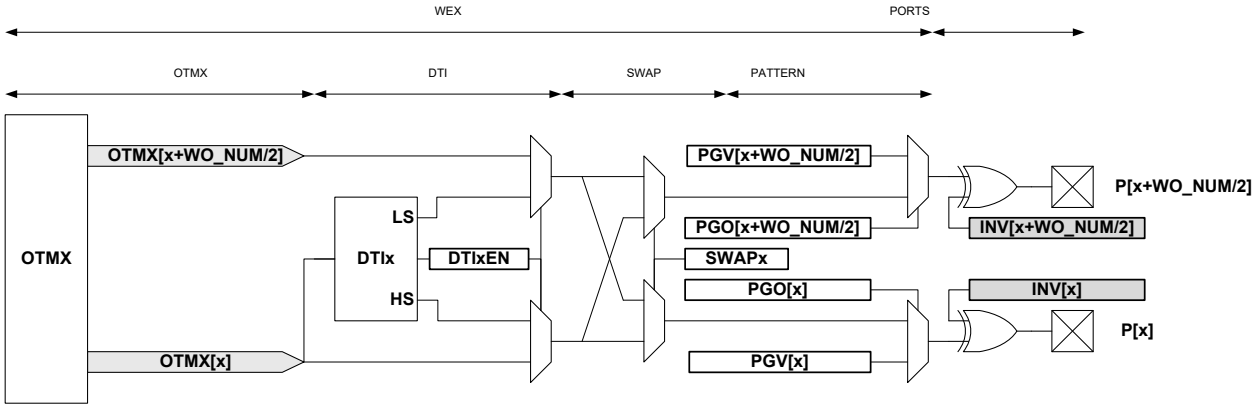
Figure 31-33 shows a schematic diagram of actions of the four optional units that follow the recoverable fault stage on a port pin pair: Output Matrix (OTMX), Dead-Time Insertion (DTI), SWAP and Pattern Generation. The DTI and SWAP units can be seen as a four port pair slices:

- Slice 0 DTI0 / SWAP0 acting on port pins (WO[0], WO[WO_NUM/2 +0])
- Slice 1 DTI1 / SWAP1 acting on port pins (WO[1], WO[WO_NUM/2 +1])

And more generally:

- Slice n DTIx / SWAPx acting on port pins (WO[x], WO[WO_NUM/2 +x])

**Figure 31-33. Waveform Extension Stage Details**




**The output matrix (OTMX)** unit distributes compare channels, according to the selectable configurations in Table 31-4.

**Table 31-4. Output Matrix Channel Pin Routing Configuration**

| Value | OTMX[x] | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0x0 | CC3 | CC2 | CC1 | CC0 | CC3 | CC2 | CC1 | CC0 |
| 0x1 | CC1 | CC0 | CC1 | CC0 | CC1 | CC0 | CC1 | CC0 |
| 0x2 | CC0 | CC0 | CC0 | CC0 | CC0 | CC0 | CC0 | CC0 |
| 0x3 | CC1 | CC1 | CC1 | CC1 | CC1 | CC1 | CC1 | CC0 |

Notes on Table 31-4:

- Configuration 0x0 is the default configuration. The channel location is the default one, and channels are distributed on outputs modulo the number of channels. Channel 0 is routed to the Output matrix output OTMX[0], and Channel 1 to OTMX[1]. If there are more outputs than channels, then channel

0 is duplicated to the Output matrix output OTMX[CC_NUM], channel 1 to OTMX[CC_NUM+1] and so on.

- Configuration 0x1 distributes the channels on output modulo half the number of channels. This assigns twice the number of output locations to the lower channels than the default configuration. This can be used, for example, to control the four transistors of a full bridge using only two compare channels.
  Using pattern generation, some of these four outputs can be overwritten by a constant level, enabling flexible drive of a full bridge in all quadrant configurations.
- Configuration 0x2 distributes compare channel 0 (CC0) to all port pins. With pattern generation, this configuration can control a stepper motor.
- Configuration 0x3 distributes the compare channel CC0 to the first output, and the channel CC1 to all other outputs. Together with pattern generation and the fault extension, this configuration can control up to seven LED strings, with a boost stage.

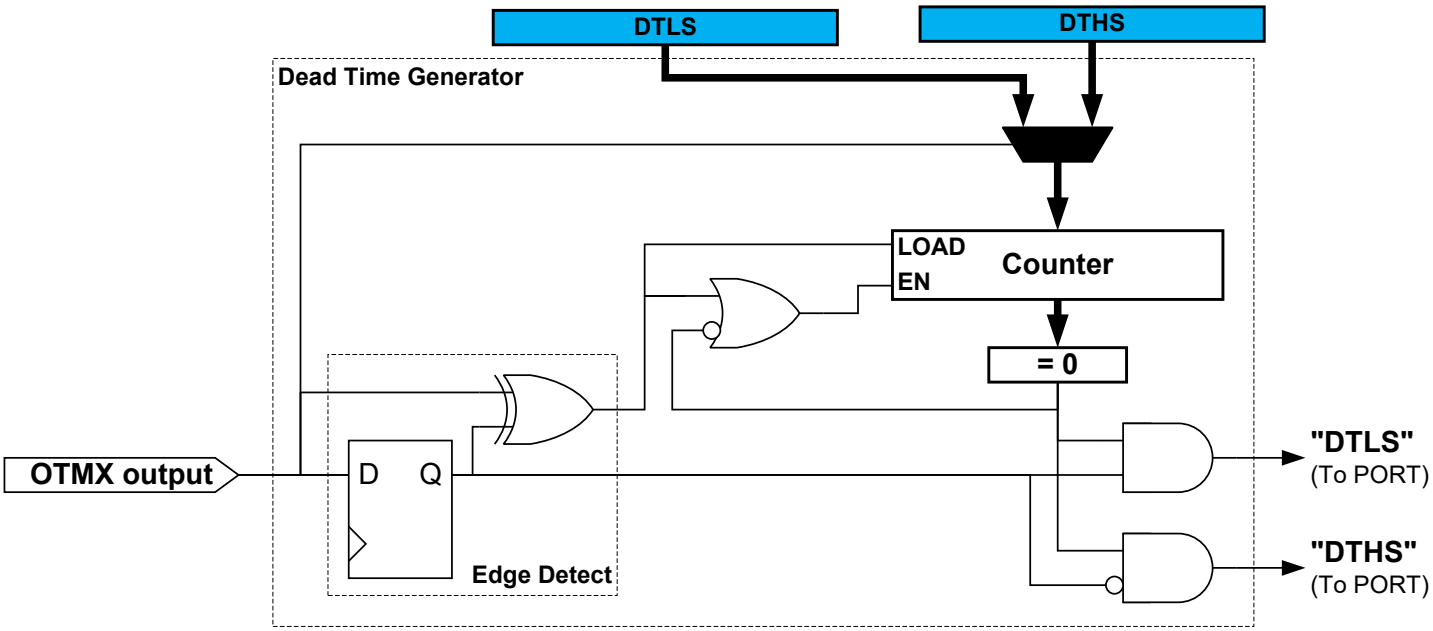
**Table 31-5. Example: four compare channels on four outputs**

| Value | OTMX[3] | OTMX[2] | OTMX[1] | OTMX[0] |
|---|---|---|---|---|
| 0x0 | CC3 | CC2 | CC1 | CC0 |
| 0x1 | CC1 | CC0 | CC1 | CC0 |
| 0x2 | CC0 | CC0 | CC0 | CC0 |
| 0x3 | CC1 | CC1 | CC1 | CC0 |

**The dead-time insertion (DTI)** unit generates OFF time with the non-inverted low side (LS) and inverted high side (HS) of the wave generator output forced at low level. This OFF time is called dead time. Dead-time insertion ensures that the LS and HS will never switch simultaneously.

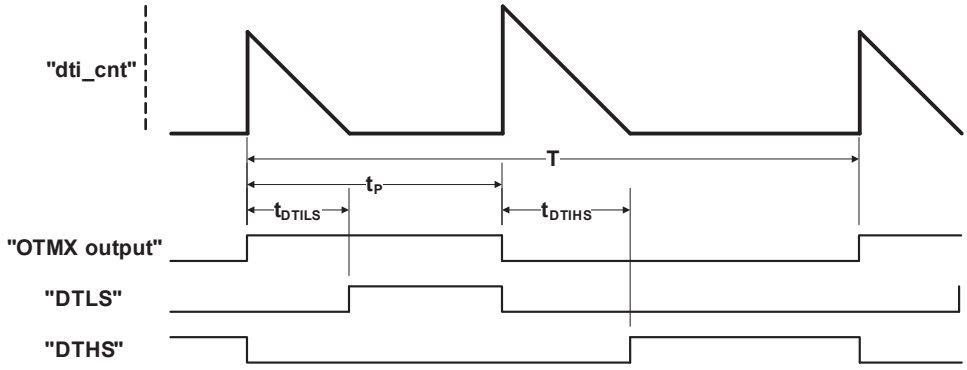
The DTI stage consists of four equal dead-time insertion generators; one for each of the first four compare channels. Figure 31-34 shows the block diagram of one DTI generator. The four channels have a common register which controls the dead time, which is independent of high side and low side setting.

**Figure 31-34. Dead-Time Generator Block Diagram**



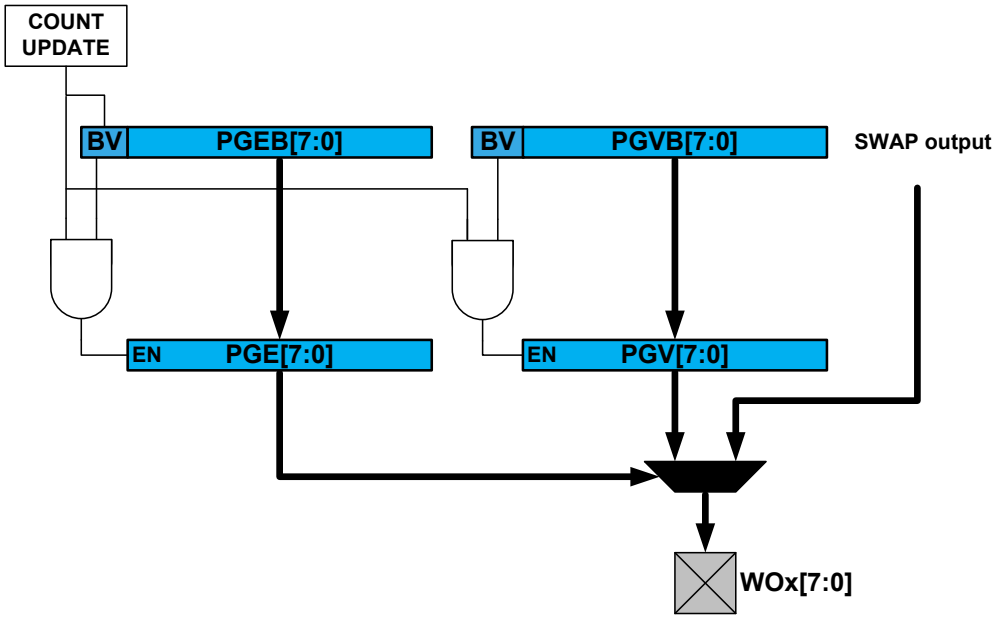
As shown in Figure 31-35, the 8-bit dead-time counter is decremented by one for each peripheral clock cycle until it reaches zero. A non-zero counter value will force both the low side and high side outputs into their OFF state. When the output matrix (OTMX) output changes, the dead-time counter is reloaded according to the edge of the input. When the output changes from low to high (positive edge) it initiates a counter reload of the DTLS register. When the output changes from high to low (negative edge) it reloads the DTHS register.

**Figure 31-35. Dead-Time Generator Timing Diagram**




**The pattern generator unit** produces a synchronized bit pattern across the port pins it is connected to. The pattern generation features are primarily intended for handling the commutation sequence in brushless DC motors (BLDC), stepper motors, and full bridge control. See also Figure 31-36.

**Figure 31-36. Pattern Generator Block Diagram**




As with other double-buffered timer/counter registers, the register update is synchronized to the UPDATE condition set by the timer/counter waveform generation operation. If synchronization is not required by the application, the software can simply access directly the PATT.PGE, PATT.PGV bits registers.

### 31.6.4 DMA, Interrupts, and Events

**Table 31-6. Module Requests for TCC**

| Condition | Interrupt request | Event output | Event input | DMA request | DMA request is cleared |
|---|---|---|---|---|---|
| Overflow / Underflow | Yes | Yes | | Yes(1) | On DMA acknowledge |
| Channel Compare Match or Capture | Yes | Yes | Yes(2) | Yes(3) | For circular buffering: on DMA acknowledge For capture channel: when CCx register is read |
| Retrigger | Yes | Yes | | | |
| Count | Yes | Yes | | | |
| Capture Overflow Error | Yes | | | | |
| Debug Fault State | Yes | | | | |
| Recoverable Faults | Yes | | | | |
| Non-Recoverable Faults | Yes | | | | |
| TCCx Event 0 input | | | Yes(4) | | |
| TCCx Event 1 input | | | Yes(5) | | |

Notes:

1. DMA request set on overflow, underflow or re-trigger conditions.
2. Can perform capture or generate recoverable fault on an event input.
3. In capture or circular modes.
4. On event input, either action can be executed:
   - re-trigger counter
   - control counter direction
   - stop the counter
   - decrement the counter
   - perform period and pulse width capture
   - generate non-recoverable fault
5. On event input, either action can be executed:
   - re-trigger counter
   - increment or decrement counter depending on direction
   - start the counter
   - increment or decrement counter based on direction
   - increment counter regardless of direction
   - generate non-recoverable fault

#### 31.6.4.1 DMA Operation

The TCC can generate the following DMA requests:

| | |
|---|---|
| **Counter overflow (OVF)** | If the Ones-shot Trigger mode in the control A register (CTRLA.DMAOS) is written to '0', the TCC generates a DMA request on each cycle when an update condition (overflow, underflow or re-trigger) is detected.<br>When an update condition (overflow, underflow or re-trigger) is detected while CTRLA.DMAOS=1, the TCC generates a DMA trigger on the cycle following the DMA One-Shot Command written to the Control B register (CTRLBSET.CMD=DMAOS).<br><br>In both cases, the request is cleared by hardware on DMA acknowledge. |
| **Channel Match (MCx)** | A DMA request is set only on a compare match if CTRLA.DMAOS=0. The request is cleared by hardware on DMA acknowledge.<br>When CTRLA.DMAOS=1, the DMA requests are not generated. |
| **Channel Capture (MCx)** | For a capture channel, the request is set when valid data is present in the CCx register, and cleared once the CCx register is read.<br>In this operation mode, the CTRLA.DMAOS bit value is ignored. |

**DMA Operation with Circular Buffer**

When circular buffer operation is enabled, the buffer registers must be written in a correct order and synchronized to the update times of the timer. The DMA triggers of the TCC provide a way to ensure a safe and correct update of circular buffers.

**Note:** Circular buffer are intended to be used with RAMP2, RAMP2A and DSBOTH operation only.

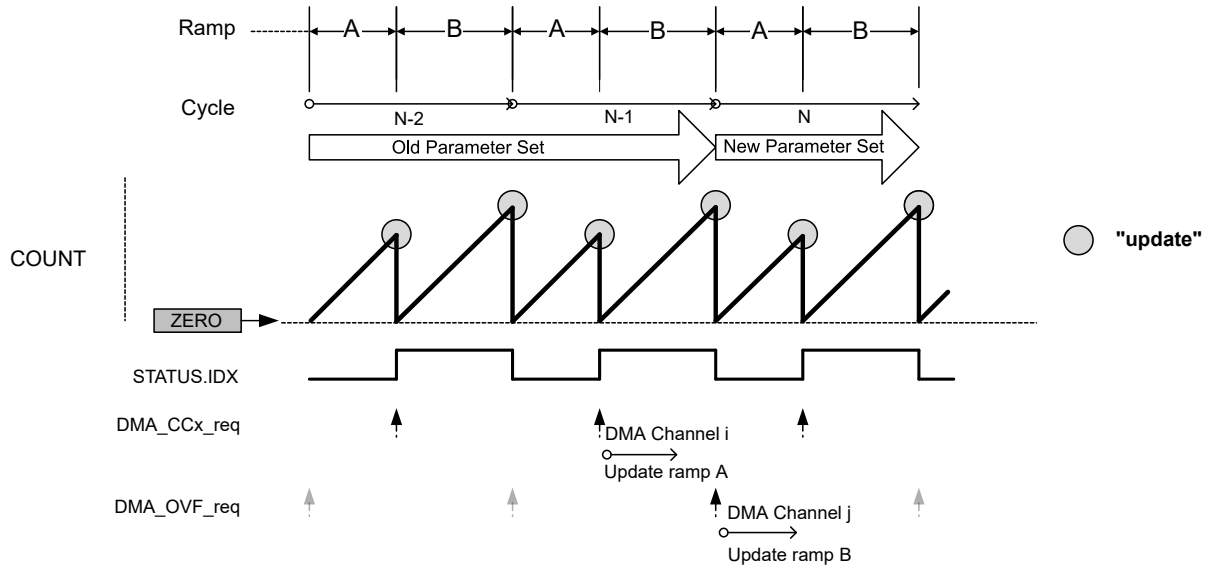
*DMA Operation with Circular Buffer in RAMP2 and RAMP2A Mode*

When a CCx channel is selected as a circular buffer, the related DMA request is not set on a compare match detection, but on start of ramp B.

If at least one circular buffer is enabled, the DMA overflow request is conditioned to the start of ramp A with an effective DMA transfer on previous ramp B (DMA acknowledge).

The update of all circular buffer values for ramp A can be done through a DMA channel triggered on a MC trigger. The update of all circular buffer values for ramp B, can be done through a second DMA channel triggered by the overflow DMA request.

**Figure 31-37. DMA Triggers in RAMP and RAMP2 Operation Mode and Circular Buffer Enabled**

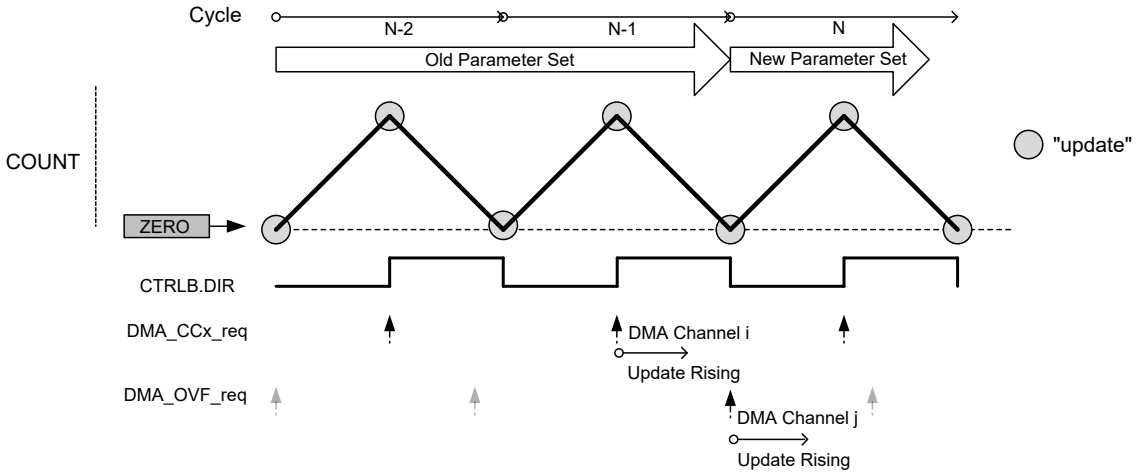



DMA Operation with Circular Buffer in DSBOTH Mode
When a CC channel is selected as a circular buffer, the related DMA request is not set on a compare match detection, but on start of down-counting phase.

If at least one circular buffer is enabled, the DMA overflow request is conditioned to the start of up-counting phase with an effective DMA transfer on previous down-counting phase (DMA acknowledge).

When up-counting, all circular buffer values can be updated through a DMA channel triggered by MC trigger. When down-counting, all circular buffer values can be updated through a second DMA channel, triggered by the OVF DMA request.

**Figure 31-38. DMA Triggers in DSBOTH Operation Mode and Circular Buffer Enabled**




### 31.6.4.2 Interrupts

The TCC has the following interrupt sources:

- Overflow/Underflow (OVF)
- Retrigger (TRG)

- Count (CNT) - refer also to description of EVCTRL.CNTSEL.
- Capture Overflow Error (ERR)
- Non-Recoverable Update Fault (UFS)
- Debug Fault State (DFS)
- Recoverable Faults (FAULTn)
- Non-recoverable Faults (FAULTx)
- Compare Match or Capture Channels (MCx)

These interrupts are asynchronous wake-up sources. See Sleep Mode Entry and Exit Table in PM/Sleep Mode Controller section for details.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the TCC is reset. See 31.8.12 INTFLAG for details on how to clear interrupt flags. The TCC has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

Note: Interrupts must be globally enabled for interrupt requests to be generated. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

16.6.2.8 Sleep Mode Controller

16.6.2.8.1 IDLE Mode

16.6.2.8.2 STANDBY Mode

#### 31.6.4.3 Events

The TCC can generate the following output events:

- Overflow/Underflow (OVF)
- Trigger (TRG)
- Counter (CNT) For further details, refer to EVCTRL.CNTSEL description.
- Compare Match or Capture on compare/capture channels: MCx

Writing a '1' ('0') to an Event Output bit in the Event Control Register (EVCTRL.xxEO) enables (disables) the corresponding output event. Refer also to *EVSYS – Event System*.

The TCC can take the following actions on a channel input event (MCx):

- Capture event
- Generate a recoverable or non-recoverable fault

The TCC can take the following actions on counter Event 1 (TCCx EV1):

- Counter re-trigger
- Counter direction control
- Stop the counter
- Decrement the counter on event
- Period and pulse width capture

- Non-recoverable fault

The TCC can take the following actions on counter Event 0 (TCCx EV0):

- Counter re-trigger
- Count on event (increment or decrement, depending on counter direction)
- Counter start - start counting on the event rising edge. Further events will not restart the counter; the counter will keep on counting using prescaled GCLK_TCCx, until it reaches TOP or ZERO, depending on the direction.
- Counter increment on event. This will increment the counter, irrespective of the counter direction.
- Count during active state of an asynchronous event (increment or decrement, depending on counter direction). In this case, the counter will be incremented or decremented on each cycle of the prescaled clock, as long as the event is active.
- Non-recoverable fault

The counter Event Actions are available in the Event Control registers (EVCTRL.EVACT0 and EVCTRL.EVACT1). For further details, refer to EVCTRL.

Writing a '1' ('0') to an Event Input bit in the Event Control register (EVCTRL.MCEIx or EVCTRL.TCEIx) enables (disables) the corresponding action on input event.

**Note:** When several events are connected to the TCC, the enabled action will apply for each of the incoming events. Refer to *EVSYS – Event System* for details on how to configure the event system.

**Related Links**

24. EVSYS – Event System

### 31.6.5 Sleep Mode Operation

The TCC can be configured to operate in any sleep mode. To be able to run in standby the RUNSTDBY bit in the Control A register (CTRLA.RUNSTDBY) must be '1'. The MODULE can in any sleep mode wake up the device using interrupts or perform actions through the Event System.

### 31.6.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset and Enable bits in Control A register (CTRLA.SWRST and CTRLA.ENABLE)

The following registers are synchronized when written:

- Control B Clear and Control B Set registers (CTRLBCLR and CTRLBSET)
- Status register (STATUS)
- Pattern and Pattern Buffer registers (PATT and PATTB)
- Waveform register (WAVE)
- Count Value register (COUNT)
- Period Value and Period Buffer Value registers (PER and PERB)
- Compare/Capture Channel x and Channel x Compare/Capture Buffer Value registers (CCx and CCBx)

The following registers are synchronized when read:

- Control B Clear and Control B Set registers (CTRLBCLR and CTRLBSET)

- Count Value register (COUNT): synchronization is done on demand through READSYNC command (CTRLBSET.CMD)
- Pattern and Pattern Buffer registers (PATT and PATTB)
- Waveform register (WAVE)
- Period Value and Period Buffer Value registers (PER and PERB)
- Compare/Capture Channel x and Channel x Compare/Capture Buffer Value registers (CCx and CCBx)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

Required read-synchronization is denoted by the "Read-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 31.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | RESOLUTION[1:0] | | | | | ENABLE | SWRST |
| | | 15:8 | | ALOCK | PRESCYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | CPTEN3 | CPTEN2 | CPTEN1 | CPTEN0 |
| 0x04 | CTRLBCLR | 7:0 | CMD[2:0] | | | IDXCMD[1:0] | | ONESHOT | LUPD | DIR |
| 0x05 | CTRLBSET | 7:0 | CMD[2:0] | | | IDXCMD[1:0] | | ONESHOT | LUPD | DIR |
| 0x06 ... 0x07 | Reserved | | | | | | | | | |
| 0x08 | SYNCBUSY | 7:0 | PER | WAVE | PATT | COUNT | STATUS | CTRLB | ENABLE | SWRST |
| | | 15:8 | | | | | CC3 | CC2 | CC1 | CC0 |
| | | 23:16 | | CCB3 | CCB2 | CCB1 | CCB0 | PERB | WAVEB | PATTB |
| | | 31:24 | | | | | | | | |
| 0x0C | FCTRLA | 7:0 | RESTART | BLANK[1:0] | | QUAL | KEEP | | SRC[1:0] | |
| | | 15:8 | | CAPTURE[2:0] | | | CHSEL[1:0] | | HALT[1:0] | |
| | | 23:16 | BLANKVAL[7:0] | | | | | | | |
| | | 31:24 | | | | | FILTERVAL[3:0] | | | |
| 0x10 | FCTRLBA | 7:0 | RESTART | BLANK[1:0] | | QUAL | KEEP | | SRC[1:0] | |
| | | 15:8 | | CAPTURE[2:0] | | | CHSEL[1:0] | | HALT[1:0] | |
| | | 23:16 | BLANKVAL[7:0] | | | | | | | |
| | | 31:24 | | | | | FILTERVAL[3:0] | | | |
| 0x14 | WEXCTRL | 7:0 | | | | | | | OTMX[1:0] | |
| | | 15:8 | | | | | DTIENx | DTIENx | DTIENx | DTIENx |
| | | 23:16 | DTLS[7:0] | | | | | | | |
| | | 31:24 | DTHS[7:0] | | | | | | | |
| 0x18 | DRVCTRL | 7:0 | NREx | NREx | NREx | NREx | NREx | NREx | NREx | NREx |
| | | 15:8 | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx |
| | | 23:16 | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx |
| | | 31:24 | FILTERVAL1[3:0] | | | | FILTERVAL0[3:0] | | | |
| 0x1C ... 0x1D | Reserved | | | | | | | | | |
| 0x1E | DBGCTRL | 7:0 | | | | | | FDDBD | | DBGRUN |
| 0x1F | Reserved | | | | | | | | | |
| 0x20 | EVCTRL | 7:0 | CNTSEL[1:0] | | EVACT1[2:0] | | | EVACT0[2:0] | | |
| | | 15:8 | TCEIx | TCEIx | TCINVx | TCINVx | | CNTEO | TRGEO | OVFEO |
| | | 23:16 | | | | | MCEIx | MCEIx | MCEIx | MCEIx |
| | | 31:24 | | | | | MCEOx | MCEOx | MCEOx | MCEOx |
| 0x24 | INTENCLR | 7:0 | | | | | ERR | CNT | TRG | OVF |
| | | 15:8 | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| | | 23:16 | | | | | MCx | MCx | MCx | MCx |
| | | 31:24 | | | | | | | | |
| 0x28 | INTENSET | 7:0 | | | | | ERR | CNT | TRG | OVF |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 15:8 | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| | | 23:16 | | | | | MCx | MCx | MCx | MCx |
| | | 31:24 | | | | | | | | |
| 0x2C | INTFLAG | 7:0 | | | | | ERR | CNT | TRG | OVF |
| | | 15:8 | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| | | 23:16 | | | | | MCx | MCx | MCx | MCx |
| | | 31:24 | | | | | | | | |
| 0x30 | STATUS | 7:0 | PERBV | WAVEBV | PATTBV | | DFS | UFS | IDX | STOP |
| | | 15:8 | FAULTx | FAULTx | FAULTB | FAULTA | FAULT1IN | FAULT0IN | FAULTBIN | FAULTAIN |
| | | 23:16 | | | | | CCBVx | CCBVx | CCBVx | CCBVx |
| | | 31:24 | | | | | CMPx | CMPx | CMPx | CMPx |
| 0x34 | COUNT | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | COUNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | PATT | 7:0 | PGE0[7:0] | | | | | | | |
| | | 15:8 | PGV0[7:0] | | | | | | | |
| 0x3A ... 0x3B | Reserved | | | | | | | | | |
| 0x3C | WAVE | 7:0 | CIPEREN | | RAMP[1:0] | | | WAVEGEN[2:0] | | |
| | | 15:8 | | | | | CICCEN3 | CICCEN2 | CICCEN1 | CICCEN0 |
| | | 23:16 | | | | | POL3 | POL2 | POL1 | POL0 |
| | | 31:24 | | | | | SWAP3 | SWAP2 | SWAP1 | SWAP0 |
| 0x40 | PER | 7:0 | PER[1:0] | | DITHER[5:0] | | | | | |
| | | 15:8 | PER[9:2] | | | | | | | |
| | | 23:16 | PER[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | CC0 | 7:0 | CC[1:0] | | DITHER[5:0] | | | | | |
| | | 15:8 | CC[9:2] | | | | | | | |
| | | 23:16 | CC[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | CC1 | 7:0 | CC[1:0] | | DITHER[5:0] | | | | | |
| | | 15:8 | CC[9:2] | | | | | | | |
| | | 23:16 | CC[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | CC2 | 7:0 | CC[1:0] | | DITHER[5:0] | | | | | |
| | | 15:8 | CC[9:2] | | | | | | | |
| | | 23:16 | CC[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 | CC3 | 7:0 | CC[1:0] | | DITHER[5:0] | | | | | |
| | | 15:8 | CC[9:2] | | | | | | | |
| | | 23:16 | CC[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 ... | Reserved | | | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x63 | | | | | | | | | | |
| 0x64 | PATTB | 7:0 | PGEB0[7:0] | | | | | | | |
| | | 15:8 | PGVB0[7:0] | | | | | | | |
| 0x66<br>...<br>0x67 | Reserved | | | | | | | | | |
| 0x68 | WAVEB | 7:0 | CIPERENB | | RAMPB[1:0] | | | WAVEGENB[2:0] | | |
| | | 15:8 | | | | | CICCENB3 | CICCENB2 | CICCENB1 | CICCENB0 |
| | | 23:16 | | | | | POLB3 | POLB2 | POLB1 | POLB0 |
| | | 31:24 | | | | | SWAPB 3 | SWAPB 2 | SWAPB 1 | SWAPB 0 |
| 0x6C | PERB | 7:0 | PERB[1:0] | | DITHERB[5:0] | | | | | |
| | | 15:8 | PERB[9:2] | | | | | | | |
| | | 23:16 | PERB[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | CCB0 | 7:0 | CCB[1:0] | | DITHERB[5:0] | | | | | |
| | | 15:8 | CCB[9:2] | | | | | | | |
| | | 23:16 | CCB[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | CCB1 | 7:0 | CCB[1:0] | | DITHERB[5:0] | | | | | |
| | | 15:8 | CCB[9:2] | | | | | | | |
| | | 23:16 | CCB[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | CCB2 | 7:0 | CCB[1:0] | | DITHERB[5:0] | | | | | |
| | | 15:8 | CCB[9:2] | | | | | | | |
| | | 23:16 | CCB[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | CCB3 | 7:0 | CCB[1:0] | | DITHERB[5:0] | | | | | |
| | | 15:8 | CCB[9:2] | | | | | | | |
| | | 23:16 | CCB[17:10] | | | | | | | |
| | | 31:24 | | | | | | | | |

## 31.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 31.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized (ENABLE, SWRST)

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPTEN3 | CPTEN2 | CPTEN1 | CPTEN0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ALOCK | PRESCYNC[1:0] | | RUNSTDBY | PRESCALER[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | RESOLUTION[1:0] | | | | | ENABLE | SWRST |
| Access | | R/W | R/W | | | | R/W | R/W |
| Reset | | 0 | 0 | | | | 0 | 0 |

**Bits 24, 25, 26, 27 – CPTEN** Capture Channel x Enable
These bits are used to select the capture or compare operation on channel x.

Writing a '1' to CPTENx enables capture on channel x.

Writing a '0' to CPTENx disables capture on channel x.

**Bit 14 – ALOCK** Auto Lock
This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The Lock Update bit in the Control B register (CTRLB.LUPD) is not affected by overflow/underflow, and re-trigger events |
| 1 | CTRLB.LUPD is set to '1' on each overflow/underflow or re-trigger event. |

**Bits 13:12 – PRESCYNC[1:0]** Prescaler and Counter Synchronization
These bits select if on re-trigger event, the Counter is cleared or reloaded on either the next GCLK_TCCx clock, or on the next prescaled GCLK_TCCx clock. It is also possible to reset the prescaler on re-trigger event.

These bits are not synchronized.

| Value | Name | Description | |
|---|---|---|---|
| | | Counter Reloaded | Prescaler |
| 0x0 | GCLK | Reload or reset Counter on next GCLK | - |
| 0x1 | PRESC | Reload or reset Counter on next prescaler clock | - |
| 0x2 | RESYNC | Reload or reset Counter on next GCLK | Reset prescaler counter |
| 0x3 | Reserved | | |

**Bit 11 – RUNSTDBY** Run in Standby
This bit is used to keep the TCC running in standby mode.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The TCC is halted in standby. |
| 1 | The TCC continues to run in standby. |

**Bits 10:8 – PRESCALER[2:0]** Prescaler
These bits select the Counter prescaler factor.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | DIV1 | Prescaler: GCLK_TCC |
| 0x1 | DIV2 | Prescaler: GCLK_TCC/2 |
| 0x2 | DIV4 | Prescaler: GCLK_TCC/4 |
| 0x3 | DIV8 | Prescaler: GCLK_TCC/8 |
| 0x4 | DIV16 | Prescaler: GCLK_TCC/16 |
| 0x5 | DIV64 | Prescaler: GCLK_TCC/64 |
| 0x6 | DIV256 | Prescaler: GCLK_TCC/256 |
| 0x7 | DIV1024 | Prescaler: GCLK_TCC/1024 |

**Bits 6:5 – RESOLUTION[1:0]** Dithering Resolution
These bits increase the TCC resolution by enabling the dithering options.

These bits are not synchronized.

**Table 31-7. Dithering**

| Value | Name | Description |
|---|---|---|
| 0x0 | NONE | The dithering is disabled. |
| 0x1 | DITH4 | Dithering is done every 16 PWM frames.<br>PER[3:0] and CCx[3:0] contain dithering pattern selection. |
| 0x2 | DITH5 | Dithering is done every 32 PWM frames. |

| Value | Name | Description |
|---|---|---|
| | | PER[4:0] and CCx[4:0] contain dithering pattern selection. |
| 0x3 | DITH6 | Dithering is done every 64 PWM frames.<br>PER[5:0] and CCx[5:0] contain dithering pattern selection. |

**Bit 1 – ENABLE** Enable
Due to synchronization there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the ENABLE bit in the SYNCBUSY register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled. |
| 1 | The peripheral is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers in the TCC (except DBGCTRL) to their initial state, and the TCC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write-operation will be discarded.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 31.8.2 Control B Clear

**Name:** CTRLBCLR
**Offset:** 0x04
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

This register allows the user to change this register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set (CTRLBSET) register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD[2:0] | | | IDXCMD[1:0] | | ONESHOT | LUPD | DIR |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:5 – CMD[2:0]** TCC Command
These bits can be used for software control of re-triggering and stop commands of the TCC. When a command has been executed, the CMD bit field will read back zero. The commands are executed on the next prescaled GCLK_TCC clock cycle.

Writing zero to this bit group has no effect.

Writing a '1' to any of these bits will clear the pending command.

| Value | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | RETRIGGER | Clear start, restart or retrigger |
| 0x2 | STOP | Force stop |
| 0x3 | UPDATE | Force update of double buffered registers |
| 0x4 | READSYNC | Force COUNT read synchronization |
| 0x5 | DMAOS | One-shot DMA trigger |

**Bits 4:3 – IDXCMD[1:0]** Ramp Index Command
These bits can be used to force cycle A and cycle B changes in RAMP2 and RAMP2A operation. On timer/counter update condition, the command is executed, the IDX flag in STATUS register is updated and the IDXCMD command is cleared.

Writing zero to these bits has no effect.

Writing a '1' to any of these bits will clear the pending command.

| Value | Name | Description |
|---|---|---|
| 0x0 | DISABLE | DISABLE Command disabled: IDX toggles between cycles A and B |
| 0x1 | SET | Set IDX: cycle B will be forced in the next cycle |
| 0x2 | CLEAR | Clear IDX: cycle A will be forced in next cycle |
| 0x3 | HOLD | Hold IDX: the next cycle will be the same as the current cycle. |

**Bit 2 – ONESHOT** One-Shot
This bit controls one-shot operation of the TCC. When one-shot operation is enabled, the TCC will stop counting on the next overflow/underflow condition or on a stop command.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will disable the one-shot operation.

| Value | Description |
|---|---|
| 0 | The TCC will update the counter value on overflow/underflow condition and continue operation. |
| 1 | The TCC will stop counting on the next underflow/overflow condition. |

**Bit 1 – LUPD** Lock Update
This bit controls the update operation of the TCC buffered registers.

When CTRLB.LUPD is cleared, the hardware UPDATE registers with value from their buffered registers is enabled.

This bit has no effect when input capture operation is enabled.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will enable the registers updates on hardware UPDATE condition.

| Value | Description |
|---|---|
| 0 | The CCBx, PERB, PGVB, PGOB, and SWAPBx buffer registers values *are* copied into the corresponding CCx, PER, PGV, PGO and SWAPx registers on hardware update condition. |
| 1 | The CCBx, PERB, PGVB, PGOB, and SWAPBx buffer registers values are *not* copied into the corresponding CCx, PER, PGV, PGO and SWAPx registers on hardware update condition. |

**Bit 0 – DIR** Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will clear the bit and make the counter count up.

| Value | Description |
|---|---|
| 0 | The timer/counter is counting up (incrementing). |
| 1 | The timer/counter is counting down (decrementing). |

### 31.8.3 Control B Set

**Name:** CTRLBSET
**Offset:** 0x05
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

This register allows the user to change this register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set (CTRLBCLR) register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMD[2:0] | | | IDXCMD[1:0] | | ONESHOT | LUPD | DIR |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:5 – CMD[2:0]** TCC Command
These bits can be used for software control of re-triggering and stop commands of the TCC. When a command has been executed, the CMD bit field will be read back as zero. The commands are executed on the next prescaled GCLK_TCC clock cycle.

Writing zero to this bit group has no effect

Writing a valid value to this bit group will set the associated command.

| Value | Name | Description |
|---|---|---|
| 0x0 | NONE | No action |
| 0x1 | RETRIGGER | Force start, restart or retrigger |
| 0x2 | STOP | Force stop |
| 0x3 | UPDATE | Force update of double buffered registers |
| 0x4 | READSYNC | Force a read synchronization of COUNT |
| 0x5 | DMAOS | One-shot DMA trigger |

**Bits 4:3 – IDXCMD[1:0]** Ramp Index Command
These bits can be used to force cycle A and cycle B changes in RAMP2 and RAMP2A operation. On timer/counter update condition, the command is executed, the IDX flag in STATUS register is updated and the IDXCMD command is cleared.

Writing a zero to these bits has no effect.

Writing a valid value to these bits will set a command.

| Value | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Command disabled: IDX toggles between cycles A and B |
| 0x1 | SET | Set IDX: cycle B will be forced in the next cycle |
| 0x2 | CLEAR | Clear IDX: cycle A will be forced in next cycle |
| 0x3 | HOLD | Hold IDX: the next cycle will be the same as the current cycle. |

**Bit 2 – ONESHOT** One-Shot
This bit controls one-shot operation of the TCC. When in one-shot operation, the TCC will stop counting on the next overflow/underflow condition or a stop command.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will enable the one-shot operation.

| Value | Description |
|---|---|
| 0 | The TCC will count continuously. |
| 1 | The TCC will stop counting on the next underflow/overflow condition. |

**Bit 1 – LUPD** Lock Update
This bit controls the update operation of the TCC buffered registers.

When CTRLB.LUPD is set, the hardware UPDATE registers with value from their buffered registers is disabled. Disabling the update ensures that all buffer registers are valid before an hardware update is performed. After all the buffer registers are loaded correctly, the buffered registers can be unlocked.

This bit has no effect when input capture operation is enabled.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will disable the registers updates on hardware UPDATE condition.

| Value | Description |
|---|---|
| 0 | The CCBx, PERB, PGVB, PGOB, and SWAPBx buffer registers values *are* copied into the corresponding CCx, PER, PGV, PGO and SWAPx registers on hardware update condition. |
| 1 | The CCBx, PERB, PGVB, PGOB, and SWAPBx buffer registers values are *not* copied into CCx, PER, PGV, PGO and SWAPx registers on hardware update condition. |

**Bit 0 – DIR** Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect

Writing a '1' to this bit will clear the bit and make the counter count up.

| Value | Description |
|---|---|
| 0 | The timer/counter is counting up (incrementing). |
| 1 | The timer/counter is counting down (decrementing). |

### 31.8.4 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | CCB3 | CCB2 | CCB1 | CCB0 | PERB | WAVEB | PATTB |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CC3 | CC2 | CC1 | CC0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PER | WAVE | PATT | COUNT | STATUS | CTRLB | ENABLE | SWRST |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19, 20, 21, 22 – CCB** Compare/Capture Buffer Channel x Synchronization Busy
This bit is cleared when the synchronization of Compare/Capture Buffer Channel x register between the clock domains is complete.

This bit is set when the synchronization of Compare/Capture Buffer Channel x register between clock domains is started.

CCBx bit is available only for existing Compare/Capture Channels. For details on CC channels number, refer to each TCC feature list.

**Bit 18 – PERB** PER Buffer Synchronization Busy
This bit is cleared when the synchronization of PERB register between the clock domains is complete.

This bit is set when the synchronization of PERB register between clock domains is started.

**Bit 17 – WAVEB** WAVE Buffer Synchronization Busy
This bit is cleared when the synchronization of WAVEB register between the clock domains is complete.

This bit is set when the synchronization of WAVEB register between clock domains is started.

**Bit 16 – PATTB** PATT Buffer Synchronization Busy
This bit is cleared when the synchronization of PATTB register between the clock domains is complete.

This bit is set when the synchronization of PATTB register between clock domains is started.

**Bits 8, 9, 10, 11 – CC** Compare/Capture Channel x Synchronization Busy
This bit is cleared when the synchronization of Compare/Capture Channel x register between the clock domains is complete.

This bit is set when the synchronization of Compare/Capture Channel x register between clock domains is started.

CCx bit is available only for existing Compare/Capture Channels. For details on CC channels number, refer to each TCC feature list.

This bit is set when the synchronization of CCx register between clock domains is started.

**Bit 7 – PER** PER Synchronization Busy
This bit is cleared when the synchronization of PER register between the clock domains is complete.

This bit is set when the synchronization of PER register between clock domains is started.

**Bit 6 – WAVE** WAVE Synchronization Busy
This bit is cleared when the synchronization of WAVE register between the clock domains is complete.

This bit is set when the synchronization of WAVE register between clock domains is started.

**Bit 5 – PATT** PATT Synchronization Busy
This bit is cleared when the synchronization of PATTERN register between the clock domains is complete.

This bit is set when the synchronization of PATTERN register between clock domains is started.

**Bit 4 – COUNT** COUNT Synchronization Busy
This bit is cleared when the synchronization of COUNT register between the clock domains is complete.

This bit is set when the synchronization of COUNT register between clock domains is started.

**Bit 3 – STATUS** STATUS Synchronization Busy
This bit is cleared when the synchronization of STATUS register between the clock domains is complete.

This bit is set when the synchronization of STATUS register between clock domains is started.

**Bit 2 – CTRLB** CTRLB Synchronization Busy
This bit is cleared when the synchronization of CTRLB register between the clock domains is complete.

This bit is set when the synchronization of CTRLB register between clock domains is started.

**Bit 1 – ENABLE** ENABLE Synchronization Busy
This bit is cleared when the synchronization of ENABLE bit between the clock domains is complete.

This bit is set when the synchronization of ENABLE bit between clock domains is started.

**Bit 0 – SWRST** SWRST Synchronization Busy
This bit is cleared when the synchronization of SWRST bit between the clock domains is complete.

This bit is set when the synchronization of SWRST bit between clock domains is started.

### 31.8.5 Fault Control A and B

**Name:** FCTRLA, FCTRLB
**Offset:** 0x0C + n*0x04 [n=0..1]
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FILTERVAL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BLANKVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CAPTURE[2:0] | | | CHSEL[1:0] | | HALT[1:0] | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RESTART | BLANK[1:0] | | QUAL | KEEP | | SRC[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bits 27:24 – FILTERVAL[3:0]** Recoverable Fault n Filter Value
These bits define the filter value applied on MCEx (x=0,1) event input line. The value must be set to zero when MCEx event is used as synchronous event.

**Bits 23:16 – BLANKVAL[7:0]** Recoverable Fault n Blanking Value
These bits determine the duration of the blanking of the fault input source. Activation and edge selection of the blank filtering are done by the BLANK bits (FCTRLn.BLANK).

When enabled, the fault input source is internally disabled for BLANKVAL* prescaled GCLK_TCC periods after the detection of the waveform edge.

**Bits 14:12 – CAPTURE[2:0]** Recoverable Fault n Capture Action
These bits select the capture and Fault n interrupt/event conditions.

**Table 31-8. Fault n Capture Action**

| Value | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Capture on valid recoverable Fault n is disabled |
| 0x1 | CAPT | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0].<br><br>INTFLAG.FAULTn flag rises on each new captured value. |

| Value | Name | Description |
|---|---|---|
| 0x2 | CAPTMIN | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0], if COUNT value is lower than the last stored capture value (CC).<br>INTFLAG.FAULTn flag rises on each local minimum detection. |
| 0x3 | CAPTMAX | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0], if COUNT value is higher than the last stored capture value (CC).<br>INTFLAG.FAULTn flag rises on each local maximun detection. |
| 0x4 | LOCMIN | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0].<br>INTFLAG.FAULTn flag rises on each local minimum value detection. |
| 0x5 | LOCMAX | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0].<br>INTFLAG.FAULTn flag rises on each local maximun detection. |
| 0x6 | DERIV0 | On rising edge of a valid recoverable Fault n, capture counter value on channel selected by CHSEL[1:0].<br>INTFLAG.FAULTn flag rises on each local maximun or minimum detection. |

**Bits 11:10 – CHSEL[1:0]** Recoverable Fault n Capture Channel
These bits select the channel for capture operation triggered by recoverable Fault n.

| Value | Name | Description |
|---|---|---|
| 0x0 | CC0 | Capture value stored into CC0 |
| 0x1 | CC1 | Capture value stored into CC1 |
| 0x2 | CC2 | Capture value stored into CC2 |
| 0x3 | CC3 | Capture value stored into CC3 |

**Bits 9:8 – HALT[1:0]** Recoverable Fault n Halt Operation
These bits select the halt action for recoverable Fault n.

| Value | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Halt action disabled |
| 0x1 | HW | Hardware halt action |
| 0x2 | SW | Software halt action |
| 0x3 | NR | Non-recoverable fault |

**Bit 7 – RESTART** Recoverable Fault n Restart
Setting this bit enables restart action for Fault n.

| Value | Description |
|---|---|
| 0 | Fault n restart action is disabled. |
| 1 | Fault n restart action is enabled. |

**Bits 6:5 – BLANK[1:0]** Recoverable Fault n Blanking Operation
These bits, select the blanking start point for recoverable Fault n.

| Value | Name | Description |
|---|---|---|
| 0x0 | START | Blanking applied from start of the Ramp period |
| 0x1 | RISE | Blanking applied from rising edge of the waveform output |
| 0x2 | FALL | Blanking applied from falling edge of the waveform output |
| 0x3 | BOTH | Blanking applied from each toggle of the waveform output |

**Bit 4 – QUAL** Recoverable Fault n Qualification
Setting this bit enables the recoverable Fault n input qualification.

| Value | Description |
|---|---|
| 0 | The recoverable Fault n input is not disabled on CMPx value condition. |
| 1 | The recoverable Fault n input is disabled when output signal is at inactive level (CMPx == 0). |

**Bit 3 – KEEP** Recoverable Fault n Keep
Setting this bit enables the Fault n keep action.

| Value | Description |
|---|---|
| 0 | The Fault n state is released as soon as the recoverable Fault n is released. |
| 1 | The Fault n state is released at the end of TCC cycle. |

**Bits 1:0 – SRC[1:0]** Recoverable Fault n Source
These bits select the TCC event input for recoverable Fault n.

Event system channel connected to MCEx event input, must be configured to route the event asynchronously, when used as a recoverable Fault n input.

| Value | Name | Description |
|---|---|---|
| 0x0 | DISABLE | Fault input disabled |
| 0x1 | ENABLE | MCEx (x=0,1) event input |
| 0x2 | INVERT | Inverted MCEx (x=0,1) event input |
| 0x3 | ALTFAULT | Alternate fault (A or B) state at the end of the previous period. |

### 31.8.6 Waveform Extension Control

**Name:** WEXCTRL
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DTHS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DTLS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DTIENx | DTIENx | DTIENx | DTIENx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OTMX[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 31:24 – DTHS[7:0]** Dead-Time High Side Outputs Value
This register holds the number of GCLK_TCC clock cycles for the dead-time high side.

**Bits 23:16 – DTLS[7:0]** Dead-time Low Side Outputs Value
This register holds the number of GCLK_TCC clock cycles for the dead-time low side.

**Bits 11,10,9,8 – DTIENx** Dead-time Insertion Generator x Enable
Setting any of these bits enables the dead-time insertion generator for the corresponding output matrix. This will override the output matrix [x] and [x+WO_NUM/2], with the low side and high side waveform respectively.

| Value | Description |
|---|---|
| 0 | No dead-time insertion override. |
| 1 | Dead time insertion override on signal outputs[x] and [x+WO_NUM/2], from matrix outputs[x] signal. |

**Bits 1:0 – OTMX[1:0]** Output Matrix
These bits define the matrix routing of the TCC waveform generation outputs to the port pins, according to Table 31-4.

### 31.8.7 Driver Control

**Name:** DRVCTRL
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FILTERVAL1[3:0] | | | | FILTERVAL0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx | INVENx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx | NRVx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NREx | NREx | NREx | NREx | NREx | NREx | NREx | NREx |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – FILTERVAL1[3:0]** Non-Recoverable Fault Input 1 Filter Value
These bits define the filter value applied on TCE1 event input line. When the TCE1 event input line is configured as a synchronous event, this value must be 0x0.

**Bits 27:24 – FILTERVAL0[3:0]** Non-Recoverable Fault Input 0 Filter Value
These bits define the filter value applied on TCE0 event input line. When the TCE0 event input line is configured as a synchronous event, this value must be 0x0.

**Bits 23,22,21,20,19,18,17,16 – INVENx** Waveform Output x Inversion
These bits are used to select inversion on the output of channel x.

Writing a '1' to INVENx inverts output from WO[x].

Writing a '0' to INVENx disables inversion of output from WO[x].

**Bits 15,14,13,12,11,10,9,8 – NRVx** NRVx Non-Recoverable State x Output Value
These bits define the value of the enabled override outputs, under non-recoverable fault condition.

**Bits 7,6,5,4,3,2,1,0 – NREx** Non-Recoverable State x Output Enable
These bits enable the override of individual outputs by NRVx value, under non-recoverable fault condition.

| Value | Description |
|---|---|
| 0 | Non-recoverable fault tri-state the output. |
| 1 | Non-recoverable faults set the output to NRVx level. |

### 31.8.8 Debug control

**Name:** DBGCTRL
**Offset:** 0x1E
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FDDBD | | DBGRUN |
| Access | | | | | | R/W | | R/W |
| Reset | | | | | | 0 | | 0 |

**Bit 2 – FDDBD** Fault Detection on Debug Break Detection
This bit is not affected by software reset and should not be changed by software while the TCC is enabled.

By default this bit is zero, and the on-chip debug (OCD) fault protection is disabled. When this bit is written to '1', OCD break request from the OCD system will trigger non-recoverable fault. When this bit is set, OCD fault protection is enabled and OCD break request from the OCD system will trigger a non-recoverable fault.

| Value | Description |
|---|---|
| 0 | No faults are generated when TCC is halted in debug mode. |
| 1 | A non recoverable fault is generated and FAULTD flag is set when TCC is halted in debug mode. |

**Bit 0 – DBGRUN** Debug Running State
This bit is not affected by software reset and should not be changed by software while the TCC is enabled.

| Value | Description |
|---|---|
| 0 | The TCC is halted when the device is halted in debug mode. |
| 1 | The TCC continues normal operation when the device is halted in debug mode. |

### 31.8.9 Event Control

**Name:** EVCTRL
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCEOx | MCEOx | MCEOx | MCEOx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCEIx | MCEIx | MCEIx | MCEIx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TCEIx | TCEIx | TCINVx | TCINVx | | CNTEO | TRGEO | OVFEO |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNTSEL[1:0] | | EVACT1[2:0] | | | EVACT0[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27,26,25,24 – MCEOx** Match or Capture Channel x Event Output Enable
These bits control if the Match/capture event on channel x is enabled and will be generated for every match or capture.

| Value | Description |
|---|---|
| 0 | Match/capture x event is disabled and will not be generated. |
| 1 | Match/capture x event is enabled and will be generated for every compare/capture on channel x. |

**Bits 19,18,17,16 – MCEIx** Match or Capture Channel x Event Input Enable
These bits indicate if the Match/capture x incoming event is enabled

These bits are used to enable match or capture input events to the CCx channel of TCC.

| Value | Description |
|---|---|
| 0 | Incoming events are disabled. |
| 1 | Incoming events are enabled. |

**Bits 15,14 – TCEIx** Timer/Counter Event Input x Enable
This bit is used to enable input event x to the TCC.

| Value | Description |
|---|---|
| 0 | Incoming event x is disabled. |
| 1 | Incoming event x is enabled. |

**Bits 13,12 – TCINVx** Timer/Counter Event x Invert Enable
This bit inverts the event x input.

| Value | Description |
|---|---|
| 0 | Input event source x is not inverted. |
| 1 | Input event source x is inverted. |

**Bit 10 – CNTEO** Timer/Counter Event Output Enable
This bit is used to enable the counter cycle event. When enabled, an event will be generated on begin or end of counter cycle depending of CNTSEL[1:0] settings.

| Value | Description |
|---|---|
| 0 | Counter cycle output event is disabled and will not be generated. |
| 1 | Counter cycle output event is enabled and will be generated depend of CNTSEL[1:0] value. |

**Bit 9 – TRGEO** Retrigger Event Output Enable
This bit is used to enable the counter retrigger event. When enabled, an event will be generated when the counter retriggers operation.

| Value | Description |
|---|---|
| 0 | Counter retrigger event is disabled and will not be generated. |
| 1 | Counter retrigger event is enabled and will be generated for every counter retrigger. |

**Bit 8 – OVFEO** Overflow/Underflow Event Output Enable
This bit is used to enable the overflow/underflow event. When enabled an event will be generated when the counter reaches the TOP or the ZERO value.

| Value | Description |
|---|---|
| 0 | Overflow/underflow counter event is disabled and will not be generated. |
| 1 | Overflow/underflow counter event is enabled and will be generated for every counter overflow/underflow. |

**Bits 7:6 – CNTSEL[1:0]** Timer/Counter Interrupt and Event Output Selection
These bits define on which part of the counter cycle the counter event output is generated.

| Value | Name | Description |
|---|---|---|
| 0x0 | BEGIN | An interrupt/event is generated at begin of each counter cycle |
| 0x1 | END | An interrupt/event is generated at end of each counter cycle |
| 0x2 | BETWEEN | An interrupt/event is generated between each counter cycle. |
| 0x3 | BOUNDARY | An interrupt/event is generated at begin of first counter cycle, and end of last counter cycle. |

**Bits 5:3 – EVACT1[2:0]** Timer/Counter Event Input 1 Action
These bits define the action the TCC will perform on TCE1 event input.

| Value | Name | Description |
|---|---|---|
| 0x0 | OFF | Event action disabled. |
| 0x1 | RETRIGGER | Start restart or re-trigger TC on event |

| Value | Name | Description |
|---|---|---|
| 0x2 | DIR (asynch) | Direction control |
| 0x3 | STOP | Stop TC on event |
| 0x4 | DEC | Decrement TC on event |
| 0x5 | PPW | Period captured into CC0 Pulse Width on CC1 |
| 0x6 | PWP | Period captured into CC1 Pulse Width on CC0 |
| 0x7 | FAULT | Non-recoverable Fault |

**Bits 2:0 – EVACT0[2:0]** Timer/Counter Event Input 0 Action
These bits define the action the TCC will perform on TCE0 event input 0.

| Value | Name | Description |
|---|---|---|
| 0x0 | OFF | Event action disabled. |
| 0x1 | RETRIGGER | Start restart or re-trigger TC on event |
| 0x2 | COUNTEV | Count on event. |
| 0x3 | START | Start TC on event |
| 0x4 | INC | Increment TC on EVENT |
| 0x5 | COUNT (async) | Count on active state of asynchronous event |
| 0x6 | - | Reserved |
| 0x7 | FAULT | Non-recoverable Fault |

### 31.8.10 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set (INTENSET) register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCx | MCx | MCx | MCx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ERR | CNT | TRG | OVF |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 19,18,17,16 – MCx** Match or Capture Channel x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the corresponding Match or Capture Channel x Interrupt Disable/Enable bit, which disables the Match or Capture Channel x interrupt.

| Value | Description |
|---|---|
| 0 | The Match or Capture Channel x interrupt is disabled. |
| 1 | The Match or Capture Channel x interrupt is enabled. |

**Bit 15 – FAULT1** Non-Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Non-Recoverable Fault x Interrupt Disable/Enable bit, which disables the Non-Recoverable Fault x interrupt.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Fault x interrupt is disabled. |
| 1 | The Non-Recoverable Fault x interrupt is enabled. |

**Bit 14 – FAULT0** Non-Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Non-Recoverable Fault x Interrupt Disable/Enable bit, which disables the Non-Recoverable Fault x interrupt.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Fault x interrupt is disabled. |
| 1 | The Non-Recoverable Fault x interrupt is enabled. |

**Bit 13 – FAULTB** Recoverable Fault B Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Recoverable Fault B Interrupt Disable/Enable bit, which disables the Recoverable Fault B interrupt.

| Value | Description |
|---|---|
| 0 | The Recoverable Fault B interrupt is disabled. |
| 1 | The Recoverable Fault B interrupt is enabled. |

**Bit 12 – FAULTA** Recoverable Fault A Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Recoverable Fault A Interrupt Disable/Enable bit, which disables the Recoverable Fault A interrupt.

| Value | Description |
|---|---|
| 0 | The Recoverable Fault A interrupt is disabled. |
| 1 | The Recoverable Fault A interrupt is enabled. |

**Bit 11 – DFS** Non-Recoverable Debug Fault Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Debug Fault State Interrupt Disable/Enable bit, which disables the Debug Fault State interrupt.

| Value | Description |
|---|---|
| 0 | The Debug Fault State interrupt is disabled. |
| 1 | The Debug Fault State interrupt is enabled. |

**Bit 10 – UFS** Non-Recoverable Update Fault Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Non-Recoverable Update Fault Interrupt Disable/Enable bit, which disables the Non-Recoverable Update Fault interrupt.
**Note:** This bit is only available on variant L devices. Refer to the *Configuration Summary* for more information.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Update Fault interrupt is disabled. |
| 1 | The Non-Recoverable Update Fault interrupt is enabled. |

**Bit 3 – ERR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Error Interrupt Disable/Enable bit, which disables the Compare interrupt.

| Value | Description |
|---|---|
| 0 | The Error interrupt is disabled. |
| 1 | The Error interrupt is enabled. |

**Bit 2 – CNT** Counter Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Counter Interrupt Disable/Enable bit, which disables the Counter interrupt.

| Value | Description |
|---|---|
| 0 | The Counter interrupt is disabled. |
| 1 | The Counter interrupt is enabled. |

**Bit 1 – TRG** Retrigger Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Retrigger Interrupt Disable/Enable bit, which disables the Retrigger interrupt.

| Value | Description |
|---|---|
| 0 | The Retrigger interrupt is disabled. |
| 1 | The Retrigger interrupt is enabled. |

**Bit 0 – OVF** Overflow Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Overflow Interrupt Disable/Enable bit, which disables the Overflow interrupt request.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled. |

### 31.8.11 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCx | MCx | MCx | MCx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ERR | CNT | TRG | OVF |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 19,18,17,16 – MCx** Match or Capture Channel x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the corresponding Match or Capture Channel x Interrupt Disable/Enable bit, which enables the Match or Capture Channel x interrupt.

| Value | Description |
|---|---|
| 0 | The Match or Capture Channel x interrupt is disabled. |
| 1 | The Match or Capture Channel x interrupt is enabled. |

**Bit 15 – FAULT1** Non-Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Non-Recoverable Fault x Interrupt Disable/Enable bit, which enables the Non-Recoverable Fault x interrupt.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Fault x interrupt is disabled. |
| 1 | The Non-Recoverable Fault x interrupt is enabled. |

**Bit 14 – FAULT0** Non-Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Non-Recoverable Fault x Interrupt Disable/Enable bit, which disables the Non-Recoverable Fault x interrupt.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Fault x interrupt is disabled. |
| 1 | The Non-Recoverable Fault x interrupt is enabled. |

**Bit 13 – FAULTB** Recoverable Fault B Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Recoverable Fault B Interrupt Disable/Enable bit, which enables the Recoverable Fault B interrupt.

| Value | Description |
|---|---|
| 0 | The Recoverable Fault B interrupt is disabled. |
| 1 | The Recoverable Fault B interrupt is enabled. |

**Bit 12 – FAULTA** Recoverable Fault A Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Recoverable Fault A Interrupt Disable/Enable bit, which enables the Recoverable Fault A interrupt.

| Value | Description |
|---|---|
| 0 | The Recoverable Fault A interrupt is disabled. |
| 1 | The Recoverable Fault A interrupt is enabled. |

**Bit 11 – DFS** Non-Recoverable Debug Fault Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Debug Fault State Interrupt Disable/Enable bit, which enables the Debug Fault State interrupt.

| Value | Description |
|---|---|
| 0 | The Debug Fault State interrupt is disabled. |
| 1 | The Debug Fault State interrupt is enabled. |

**Bit 10 – UFS** Non-Recoverable Update Fault Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Non-Recoverable Update Fault Interrupt Disable/Enable bit, which enables the Non-Recoverable Update Fault interrupt.
**Note:** This bit is only available on variant L devices. Refer to the *Configuration Summary* for more information.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Update Fault interrupt is disabled. |
| 1 | The Non-Recoverable Update Fault interrupt is enabled. |

**Bit 3 – ERR** Error Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Error Interrupt Disable/Enable bit, which enables the Compare interrupt.

| Value | Description |
|---|---|
| 0 | The Error interrupt is disabled. |
| 1 | The Error interrupt is enabled. |

**Bit 2 – CNT** Counter Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Retrigger Interrupt Disable/Enable bit, which enables the Counter interrupt.

| Value | Description |
|---|---|
| 0 | The Counter interrupt is disabled. |
| 1 | The Counter interrupt is enabled. |

**Bit 1 – TRG** Retrigger Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Retrigger Interrupt Disable/Enable bit, which enables the Retrigger interrupt.

| Value | Description |
|---|---|
| 0 | The Retrigger interrupt is disabled. |
| 1 | The Retrigger interrupt is enabled. |

**Bit 0 – OVF** Overflow Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Overflow Interrupt Disable/Enable bit, which enables the Overflow interrupt request.

| Value | Description |
|---|---|
| 0 | The Overflow interrupt is disabled. |
| 1 | The Overflow interrupt is enabled. |

### 31.8.12 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MCx | MCx | MCx | MCx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FAULT1 | FAULT0 | FAULTB | FAULTA | DFS | UFS | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ERR | CNT | TRG | OVF |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 19,18,17,16 – MCx** Match or Capture Channel x Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a match with the compare condition or once CCx register contain a valid capture value.

Writing a '0' to one of these bits has no effect.

Writing a '1' to one of these bits will clear the corresponding Match or Capture Channel x interrupt flag

In Capture operation, this flag is automatically cleared when CCx register is read.

**Bit 15 – FAULT1** Non-Recoverable Fault x Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a Non-Recoverable Fault x occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Non-Recoverable Fault x interrupt flag.

**Bit 14 – FAULT0** Non-Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Non-Recoverable Fault x Interrupt Disable/Enable bit, which disables the Non-Recoverable Fault x interrupt.

| Value | Description |
|---|---|
| 0 | The Non-Recoverable Fault x interrupt is disabled. |
| 1 | The Non-Recoverable Fault x interrupt is enabled. |

**Bit 13 – FAULTB** Recoverable Fault B Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a Recoverable Fault B occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Recoverable Fault B interrupt flag.

**Bit 12 – FAULTA** Recoverable Fault A Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a Recoverable Fault B occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Recoverable Fault B interrupt flag.

**Bit 11 – DFS** Non-Recoverable Debug Fault State Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after an Debug Fault State occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Debug Fault State interrupt flag.

**Bit 10 – UFS** Non-Recoverable Update Fault
This flag is set when the RAMP index changes and the Lock Update bit is set (CTRLBSET.LUPD).

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Non-Recoverable Update Fault interrupt flag.

**Note:** This bit is only available on variant L devices. Refer to the *Configuration Summary* for more information.

**Bit 3 – ERR** Error Interrupt Flag
This flag is set if a new capture occurs on a channel when the corresponding Match or Capture Channel x interrupt flag is one. In which case there is nowhere to store the new capture.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the error interrupt flag.

**Bit 2 – CNT** Counter Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a counter event occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the CNT interrupt flag.

**Bit 1 – TRG** Retrigger Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after a counter retrigger occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the re-trigger interrupt flag.

**Bit 0 – OVF** Overflow Interrupt Flag
This flag is set on the next CLK_TCC_COUNT cycle after an overflow condition occurs.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Overflow interrupt flag.

### 31.8.13 Status

**Name:** STATUS
**Offset:** 0x30
**Reset:** 0x00000001
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CMPx | CMPx | CMPx | CMPx |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CCBVx | CCBVx | CCBVx | CCBVx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FAULTx | FAULTx | FAULTB | FAULTA | FAULT1IN | FAULT0IN | FAULTBIN | FAULTAIN |
| Access | R/W | R/W | R/W | R/W | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PERBV | WAVEBV | PATTBV | | DFS | UFS | IDX | STOP |
| Access | R/W | R/W | R/W | | R/W | R/W | R | R |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | 1 |

**Bits 27,26,25,24 – CMPx** Channel x Compare Value
This bit reflects the channel x output compare value.

| Value | Description |
|---|---|
| 0 | Channel compare output value is 0. |
| 1 | Channel compare output value is 1. |

**Bits 19,18,17,16 – CCBVx** Channel x Compare or Capture Buffer Valid
For a compare channel, this bit is set when a new value is written to the corresponding CCBx register. The bit is cleared either by writing a '1' to the corresponding location when CTRLB.LUPD is set, or automatically on an UPDATE condition.

For a capture channel, the bit is set when a valid capture value is stored in the CCBx register. The bit is automatically cleared when the CCx register is read.

**Bits 15,14 – FAULTx** Non-recoverable Fault x State
This bit is set by hardware as soon as non-recoverable Fault x condition occurs.

This bit is cleared by writing a one to this bit and when the corresponding FAULTxIN status bit is low.

Once this bit is clear, the timer/counter will restart from the last COUNT value. To restart the timer/counter from BOTTOM, the timer/counter restart command must be executed before clearing the corresponding

STATEx bit. For further details on timer/counter commands, refer to available commands description (31.8.3 CTRLBSET.CMD).

**Bit 13 – FAULTB** Recoverable Fault B State
This bit is set by hardware as soon as recoverable Fault B condition occurs.

This bit can be clear by hardware when Fault B action is resumed, or by writing a '1' to this bit when the corresponding FAULTBIN bit is low. If software halt command is enabled (FAULTB.HALT=SW), clearing this bit will release the timer/counter.

**Bit 12 – FAULTA** Recoverable Fault A State
This bit is set by hardware as soon as recoverable Fault A condition occurs.

This bit can be clear by hardware when Fault A action is resumed, or by writing a '1' to this bit when the corresponding FAULTAIN bit is low. If software halt command is enabled (FAULTA.HALT=SW), clearing this bit will release the timer/counter.

**Bit 11 – FAULT1IN** Non-Recoverable Fault 1 Input
This bit is set while an active Non-Recoverable Fault 1 input is present.

**Bit 10 – FAULT0IN** Non-Recoverable Fault 0 Input
This bit is set while an active Non-Recoverable Fault 0 input is present.

**Bit 9 – FAULTBIN** Recoverable Fault B Input
This bit is set while an active Recoverable Fault B input is present.

**Bit 8 – FAULTAIN** Recoverable Fault A Input
This bit is set while an active Recoverable Fault A input is present.

**Bit 7 – PERBV** Period Buffer Valid
This bit is set when a new value is written to the PERB register. This bit is automatically cleared by hardware on UPDATE condition when CTRLB.LUPD is set, or by writing a '1' to this bit.

**Bit 6 – WAVEBV** Waveform Control Buffer Valid
This bit is set when a new value is written to the WAVEB register. This bit is automatically cleared by hardware on UPDATE condition when CTRLB.LUPD is set, or by writing a '1' to this bit.

**Bit 5 – PATTBV** Pattern Generator Value Buffer Valid
This bit is set when a new value is written to the PATTB register. This bit is automatically cleared by hardware on UPDATE condition when CTRLB.LUPD is set, or by writing a '1' to this bit.

**Bit 3 – DFS** Debug Fault State
This bit is set by hardware in debug mode when DDBGCTRL.FDDBD bit is set. The bit is cleared by writing a '1' to this bit and when the TCC is not in debug mode.

When the bit is set, the counter is halted and the waveforms state depend on DRVCTRL.NRE and DRVCTRL.NRV registers.

**Bit 2 – UFS** Non-recoverable Update Fault State
This bit is set by hardware when the RAMP index changes and the Lock Update bit is set (CTRLBSET.LUPD). The bit is cleared by writing a one to this bit.

When the bit is set, the waveforms state depend on DRVCTRL.NRE and DRVCTRL.NRV registers.

**Bit 1 – IDX** Ramp Index
In RAMP2 and RAMP2A operation, the bit is cleared during the cycle A and set during the cycle B. In RAMP1 operation, the bit always reads zero. For details on ramp operations, refer to 31.6.3.4 Ramp Operations.

**Bit 0 – STOP** Stop
This bit is set when the TCC is disabled either on a STOP command or on an UPDATE condition when One-Shot operation mode is enabled (CTRLBSET.ONESHOT=1).

This bit is clear on the next incoming counter increment or decrement.

| Value | Description |
|---|---|
| 0 | Counter is running. |
| 1 | Counter is stopped. |

### 31.8.14 Counter Value

**Name:** COUNT
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

**Note:** Prior to any read access, this register must be synchronized by user by writing the according TCC Command value to the Control B Set register (CTRLBSET.CMD=READSYNC).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – COUNT[23:0]** Counter Value
These bits hold the value of the counter register.

**Note:** When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

**Note:** This bit field occupies the MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [23:m] |
|---|---|
| 0x0 - NONE | 23:0 (depicted) |
| 0x1 - DITH4 | 23:4 |
| 0x2 - DITH5 | 23:5 |
| 0x3 - DITH6 | 23:6 |

### 31.8.15 Pattern

**Name:** PATT
**Offset:** 0x38
**Reset:** 0x0000
**Property:** Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PGV0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PGE0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:15, 16:23, 24:31, 32:39, 40:47, 48:55, 56:63, 64:71 – PGV** Pattern Generation Output Value
This register holds the values of pattern for each waveform output.

**Bits 0:7, 8:15, 16:23, 24:31, 32:39, 40:47, 48:55, 56:63 – PGE** Pattern Generation Output Enable
This register holds the enable status of pattern generation for each waveform output. A bit written to '1' will override the corresponding SWAP output with the corresponding PGVn value.

### 31.8.16 Waveform

**Name:** WAVE
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SWAP3 | SWAP2 | SWAP1 | SWAP0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | POL3 | POL2 | POL1 | POL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CICCEN3 | CICCEN2 | CICCEN1 | CICCEN0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CIPEREN | | RAMP[1:0] | | | WAVEGEN[2:0] | | |
| Access | R/W | | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | | 0 | 0 | 0 |

**Bits 24, 25, 26, 27 – SWAP** Swap DTI Output Pair x
Setting these bits enables output swap of DTI outputs [x] and [x+WO_NUM/2]. Note the DTIxEN settings will not affect the swap operation.

**Bits 16, 17, 18, 19 – POL** Channel Polarity x
Setting these bits enables the output polarity in single-slope and dual-slope PWM operations.

| Value | Name | Description |
|---|---|---|
| 0 | (single-slope PWM waveform generation) | Compare output is initialized to ~DIR and set to DIR when TCC counter matches CCx value |
| 1 | (single-slope PWM waveform generation) | Compare output is initialized to DIR and set to ~DIR when TCC counter matches CCx value. |
| 0 | (dual-slope PWM waveform generation) | Compare output is set to ~DIR when TCC counter matches CCx value |
| 1 | (dual-slope PWM waveform generation) | Compare output is set to DIR when TCC counter matches CCx value. |

**Bits 8, 9, 10, 11 – CICCEN** Circular CC Enable x
Setting this bits enables the compare circular buffer option on channel. When the bit is set, CCx register value is copied-back into the CCx register on UPDATE condition.

**Bit 7 – CIPEREN** Circular Period Enable
Setting this bits enable the period circular buffer option. When the bit is set, the PER register value is copied-back into the PERB register on UPDATE condition.

**Bits 5:4 – RAMP[1:0]** Ramp Operation
These bits select Ramp operation (RAMP). These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | RAMP1 | RAMP1 operation |
| 0x1 | RAMP2A | Alternative RAMP2 operation |
| 0x2 | RAMP2 | RAMP2 operation |
| 0x3 | RAMP2C. This bit is only available in variant L devices. Refer to *Configuration Summary* for more information. | Critical RAMP2 operation |
| 0x4 | - | Reserved |

**Bits 2:0 – WAVEGEN[2:0]** Waveform Generation Operation
These bits select the waveform generation operation. The settings impact the top value and control if frequency or PWM waveform generation should be used. These bits are not synchronized.

| Value | Name | Description | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | Operation | Top | Update | Waveform Output On Match | Waveform Output On Update | OVFIF/Event Up Down | |
| 0x0 | NFRQ | Normal Frequency | PER | TOP/Zero | Toggle | Stable | TOP | Zero |
| 0x1 | MFRQ | Match Frequency | CC0 | TOP/Zero | Toggle | Stable | TOP | Zero |
| 0x2 | NPWM | Normal PWM | PER | TOP/Zero | Set | Clear | TOP | Zero |
| 0x3 | Reserved | - | - | - | - | - | - | - |
| 0x4 | DSCRITICAL | Dual-slope PWM | PER | Zero | ~DIR | Stable | – | Zero |
| 0x5 | DSBOTTOM | Dual-slope PWM | PER | Zero | ~DIR | Stable | – | Zero |
| 0x6 | DSBOTH | Dual-slope PWM | PER | TOP & Zero | ~DIR | Stable | TOP | Zero |
| 0x7 | DSTOP | Dual-slope PWM | PER | Zero | ~DIR | Stable | TOP | – |

### 31.8.17 Period Value

**Name:** PER
**Offset:** 0x40
**Reset:** 0xFFFFFFFF
**Property:** Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PER[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PER[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PER[1:0] | | DITHER[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 23:6 – PER[17:0]** Period Value
These bits hold the value of the period buffer register.

**Note:** When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

**Note:** This bit field occupies the MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [23:m] |
|---|---|
| 0x0 - NONE | 23:0 |
| 0x1 - DITH4 | 23:4 |
| 0x2 - DITH5 | 23:5 |
| 0x3 - DITH6 | 23:6 (depicted) |

**Bits 5:0 – DITHER[5:0]** Dithering Cycle Number
These bits hold the number of extra cycles that are added on the PWM pulse period every 64 PWM frames.
**Note:** This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [n:0] |
|---|---|
| 0x0 - NONE | - |
| 0x1 - DITH4 | 3:0 |
| 0x2 - DITH5 | 4:0 |
| 0x3 - DITH6 | 5:0 (depicted) |

### 31.8.18 Compare/Capture Channel x

**Name:** CC
**Offset:** 0x44 + n*0x04 [n=0..3]
**Reset:** 0x00000000
**Property:** Write-Synchronized, Read-Synchronized

The CCx register represents the 16-, 24- bit value, CCx. The register has two functions, depending of the mode of operation.

For capture operation, this register represents the second buffer level and access point for the CPU and DMA.

For compare operation, this register is continuously compared to the counter value. Normally, the output form the comparator is then used for generating waveforms.

CCx register is updated with the buffer value from their corresponding CCBx register when an UPDATE condition occurs.

In addition, in match frequency operation, the CC0 register controls the counter period.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CC[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CC[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CC[1:0] | | DITHER[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:6 – CC[17:0]** Channel x Compare/Capture Value
These bits hold the value of the Channel x compare/capture register.

**Note:** When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

**Note:** This bit field occupies the m MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [23:m] |
|---|---|
| 0x0 - NONE | 23:0 |
| 0x1 - DITH4 | 23:4 |
| 0x2 - DITH5 | 23:5 |
| 0x3 - DITH6 | 23:6 (depicted) |

**Bits 5:0 – DITHER[5:0]** Dithering Cycle Number
These bits hold the number of extra cycles that are added on the PWM pulse width every 64 PWM frames.
**Note:** This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [n:0] |
|---|---|
| 0x0 - NONE | - |
| 0x1 - DITH4 | 3:0 |
| 0x2 - DITH5 | 4:0 |
| 0x3 - DITH6 | 5:0 (depicted) |

### 31.8.19 Pattern Buffer

**Name:** PATTB
**Offset:** 0x64
**Reset:** 0x0000
**Property:** Write-Synchronized, Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PGVB0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PGEB0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:15, 16:23, 24:31, 32:39, 40:47, 48:55, 56:63, 64:71 – PGVB** Pattern Generation Output Value Buffer
This register is the buffer for the PGV register. If double buffering is used, valid content in this register is copied to the PGV register on an UPDATE condition.

**Bits 0:7, 8:15, 16:23, 24:31, 32:39, 40:47, 48:55, 56:63 – PGEB** Pattern Generation Output Enable Buffer
This register is the buffer of the PGE register. If double buffering is used, valid content in this register is copied into the PGE register at an UPDATE condition.

### 31.8.20 Waveform Buffer

**Name:** WAVEB
**Offset:** 0x68
**Reset:** 0x00000000
**Property:** Write-Synchronized, Read-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SWAPB 3 | SWAPB 2 | SWAPB 1 | SWAPB 0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | POLB3 | POLB2 | POLB1 | POLB0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CICCENB3 | CICCENB2 | CICCENB1 | CICCENB0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CIPERENB | | RAMPB[1:0] | | | WAVEGENB[2:0] | | |
| Access | R/W | | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | | 0 | 0 | 0 |

**Bits 24, 25, 26, 27 – SWAPB** Swap DTI output pair x Buffer
These register bits are the buffer bits for the SWAP register bits. If double buffering is used, valid content in these bits is copied to the corresponding SWAPx bits on an UPDATE condition.

**Bits 16, 17, 18, 19 – POLB** Channel Polarity x Buffer
These register bits are the buffer bits for POLx register bits. If double buffering is used, valid content in these bits is copied to the corresponding POBx bits on an UPDATE condition.

**Bits 8, 9, 10, 11 – CICCENB** Circular CCx Buffer Enable
These register bits are the buffer bits for CICCENx register bits. If double buffering is used, valid content in these bits is copied to the corresponding CICCENx bits on a UPDATE condition.

**Bit 7 – CIPERENB** Circular Period Enable Buffer
This register bit is the buffer bit for CIPEREN register bit. If double buffering is used, valid content in this bit is copied to the corresponding CIPEREN bit on a UPDATE condition.

**Bits 5:4 – RAMPB[1:0]** Ramp Operation Buffer
These register bits are the buffer bits for RAMP register bits. If double buffering is used, valid content in these bits is copied to the corresponding RAMP bits on a UPDATE condition.

| Value | Name | Description |
|---|---|---|
| 0x0 | RAMP1 | RAMP1 operation |
| 0x1 | RAMP2A | Alternative RAMP2 operation |
| 0x2 | RAMP2 | RAMP2 operation |
| 0x3 | RAMP2C. This bit is only available in variant L devices. Refer to *Configuration Summary* for more information. | Critical RAMP2 operation |
| 0x4 | - | Reserved |

**Bits 2:0 – WAVEGENB[2:0]** Waveform Generation Operation Buffer
These register bits are the buffer bits for WAVEGEN register bits. If double buffering is used, valid content in these bits is copied to the corresponding WAVEGEN bits on a UPDATE condition.

| Value | Name | Description | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | Operation | Top | Update | Waveform Output On Match | Waveform Output On Update | OVFIF/Event Up | Down |
| 0x0 | NFRQ | Normal Frequency | PER | TOP/Zero | Toggle | Stable | TOP | Zero |
| 0x1 | MFRQ | Match Frequency | CC0 | TOP/Zero | Toggle | Stable | TOP | Zero |
| 0x2 | NPWM | Normal PWM | PER | TOP/Zero | Set | Clear | TOP | Zero |
| 0x3 | Reserved | - | - | - | - | - | - | - |
| 0x4 | DSCRITICAL | Dual-slope PWM | PER | Zero | ~DIR | Stable | – | Zero |
| 0x5 | DSBOTTOM | Dual-slope PWM | PER | Zero | ~DIR | Stable | – | Zero |
| 0x6 | DSBOTH | Dual-slope PWM | PER | TOP & Zero | ~DIR | Stable | TOP | Zero |
| 0x7 | DSTOP | Dual-slope PWM | PER | Zero | ~DIR | Stable | TOP | – |

### 31.8.21 Period Buffer Value

**Name:** PERB
**Offset:** 0x6C
**Reset:** 0xFFFFFFFF
**Property:** Write-Synchronized, Read-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PERB[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PERB[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PERB[1:0] | | DITHERB[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 23:6 – PERB[17:0]** Period Buffer Value
These bits hold the value of the period buffer register. The value is copied to PER register on UPDATE condition.

**Note:** When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

**Note:** This bit field occupies the MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [23:m] |
|---|---|
| 0x0 - NONE | 23:0 |
| 0x1 - DITH4 | 23:4 |
| 0x2 - DITH5 | 23:5 |
| 0x3 - DITH6 | 23:6 (depicted) |

**Bits 5:0 – DITHERB[5:0]** Dithering Buffer Cycle Number
These bits represent the PER.DITHER bits buffer. When the double buffering is enabled, the value of this bit field is copied to the PER.DITHER bits on an UPDATE condition.
**Note:** This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [n:0] |
|---|---|
| 0x0 - NONE | - |
| 0x1 - DITH4 | 3:0 |
| 0x2 - DITH5 | 4:0 |
| 0x3 - DITH6 | 5:0 (depicted) |

### 31.8.22 Channel x Compare/Capture Buffer Value

**Name:** CCB
**Offset:** 0x70 + n*0x04 [n=0..3]
**Reset:** 0x00000000
**Property:** Write-Synchronized, Read-Synchronized

CCBx is copied into CCx at TCC update time

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CCB[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CCB[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CCB[1:0] | | DITHERB[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:6 – CCB[17:0]** Channel x Compare/Capture Buffer Value
These bits hold the value of the Channel x Compare/Capture Buffer Value register. The register serves as the buffer for the associated compare or capture registers (CCx). Accessing this register using the CPU or DMA will affect the corresponding CCBVx status bit.

**Note:** When the TCC is configured as 16-bit timer/counter, the excess bits are read zero.

**Note:** This bit field occupies the MSB of the register, [23:m]. m is dependent on the Resolution bit in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [23:m] |
|---|---|
| 0x0 - NONE | 23:0 |
| 0x1 - DITH4 | 23:4 |
| 0x2 - DITH5 | 23:5 |
| 0x3 - DITH6 | 23:6 (depicted) |

**Bits 5:0 – DITHERB[5:0]** Dithering Buffer Cycle Number
These bits represent the CCx.DITHER bits buffer. When the double buffering is enable, DITHERBUF bits value is copied to the CCx.DITHER bits on an UPDATE condition.

**Note:** This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

| CTRLA.RESOLUTION | Bits [n:0] |
|---|---|
| 0x0 - NONE | - |
| 0x1 - DITH4 | 3:0 |
| 0x2 - DITH5 | 4:0 |
| 0x3 - DITH6 | 5:0 (depicted) |

# 32. USB – Universal Serial Bus

## 32.1 Overview

The Universal Serial Bus interface (USB) module complies with the Universal Serial Bus (USB) 2.1 specification supporting both device and embedded host modes.

The USB device mode supports 8 endpoint addresses. All endpoint addresses have one input and one output endpoint, for a total of 16 endpoints. Each endpoint is fully configurable in any of the four transfer types: control, interrupt, bulk or isochronous. The USB host mode supports up to 8 pipes. The maximum data payload size is selectable up to 1023 bytes.

Internal SRAM is used to keep the configuration and data buffer for each endpoint. The memory locations used for the endpoint configurations and data buffers is fully configurable. The amount of memory allocated is dynamic according to the number of endpoints in use, and the configuration of these. The USB module has a built-in Direct Memory Access (DMA) and will read/write data from/to the system RAM when a USB transaction takes place. No CPU or DMA Controller resources are required.

To maximize throughput, an endpoint can be configured for ping-pong operation. When this is done the input and output endpoint with the same address are used in the same direction. The CPU or DMA Controller can then read/write one data buffer while the USB module writes/reads from the other buffer. This gives double buffered communication.

Multi-packet transfer enables a data payload exceeding the maximum packet size of an endpoint to be transferred as multiple packets without any software intervention. This reduces the number of interrupts and software intervention needed for USB transfers.

For low power operation the USB module can put the microcontroller in any sleep mode when the USB bus is idle and a suspend condition is given. Upon bus resume, the USB module can wake the microcontroller from any sleep mode.

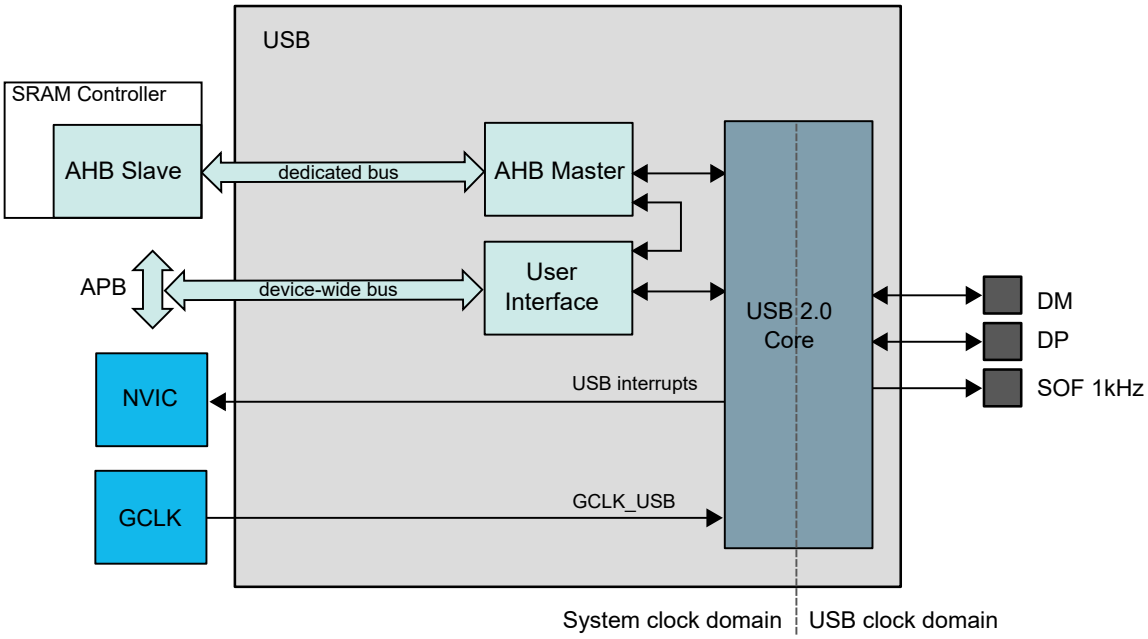
## 32.2 Features

- Compatible with the USB 2.1 specification
- USB Embedded Host and Device mode
- Supports full (12Mbit/s) and low (1.5Mbit/s) speed communication
- Supports Link Power Management (LPM-L1) protocol
- On-chip transceivers with built-in pull-ups and pull-downs
- On-Chip USB serial resistors
- 1kHz SOF clock available on external pin
- Device mode
  - Supports 8 IN endpoints and 8 OUT endpoints
  - No endpoint size limitations
  - Built-in DMA with multi-packet and dual bank for all endpoints
  - Supports feedback endpoint
  - Supports crystal less clock
- Host mode
  - Supports 8 physical pipes

- No pipe size limitations
- Supports multiplexed virtual pipe on one physical pipe to allow an unlimited USB tree
- Built-in DMA with multi-packet support and dual bank for all pipes
- Supports feedback endpoint
- Supports the USB 2.0 Phase-locked SOFs feature

## 32.3 USB Block Diagram

**Figure 32-1. LS/FS Implementation: USB Block Diagram**




## 32.4 Signal Description

| Pin Name | Pin Description | Type |
|---|---|---|
| DM | Data -: Differential Data Line - Port | Input/Output |
| DP | Data +: Differential Data Line + Port | Input/Output |
| SOF 1kHZ | SOF Output | Output |

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 32.5 Product Dependencies

In order to use this peripheral module, other parts of the system must be configured correctly, as described below.

### 32.5.1 I/O Lines

The USB pins may be multiplexed with the I/O lines Controller. The user must first configure the I/O Controller to assign the USB pins to their peripheral functions.

A 1kHz SOF clock is available on an external pin. The user must first configure the I/O Controller to assign the 1kHz SOF clock to the peripheral function. The SOF clock is available for device and host mode.

### 32.5.2 Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 32.5.3 Clocks

The USB bus clock (CLK_USB_AHB) can be enabled and disabled in the Main Clock module, MCLK, and the default state of CLK_USB_AHB can be found in the *Peripheral Clock Masking*.

A generic clock (GCLK_USB) is required to clock the USB. This clock must be configured and enabled in the Generic Clock Controller before using the USB.

This generic clock is asynchronous to the bus clock (CLK_USB_AHB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains.

The USB module requires a GCLK_USB of 48 MHz ± 0.25% clock for low speed and full speed operation. To follow the USB data rate at 12 Mbit/s in full-speed mode, the CLK_USB_AHB clock should be at minimum 8 MHz.

Clock recovery is achieved by a digital phase-locked loop in the USB module, which complies with the USB jitter specifications. If crystal-less operation is used in USB device mode, refer to *USB Clock Recovery Module*.

**Related Links**

15. GCLK - Generic Clock Controller

16.6.2.6 Peripheral Clock Masking

15.6.6 Synchronization

### 32.5.4 DMA

The USB has a built-in Direct Memory Access (DMA) and will read/write data to/from the system RAM when a USB transaction takes place. No CPU or DMA Controller resources are required.

### 32.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 32.5.6 Events

Not applicable.

### 32.5.7 Debug Operation

When the CPU is halted in debug mode the USB peripheral continues normal operation. If the USB peripheral is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

### 32.5.8 Register Access Protection

Registers with write-access can be optionally write-protected by the Peripheral Access Controller (PAC), except for the following:

- Device Interrupt Flag (INTFLAG) register
- Endpoint Interrupt Flag (EPINTFLAG) register
- Host Interrupt Flag (INTFLAG) register
- Pipe Interrupt Flag (PINTFLAG) register

**Note:** Optional write-protection is indicated by the "PAC Write-Protection" property in the register description.

Write-protection does not apply for accesses through an external debugger.

### 32.5.9 Analog Connections

Not applicable.

### 32.5.10 Calibration

The output drivers for the DP/DM USB line interface can be fine tuned with calibration values from production tests. The calibration values must be loaded from the NVM Software Calibration Area into the USB Pad Calibration register (PADCAL) by software, before enabling the USB, to achieve the specified accuracy. Refer to *NVM Software Calibration Area Mapping* for further details.

For details on Pad Calibration, refer to Pad Calibration (32.8.1.6 PADCAL) register.

**Related Links**

10.3.2 NVM Software Calibration Area Mapping

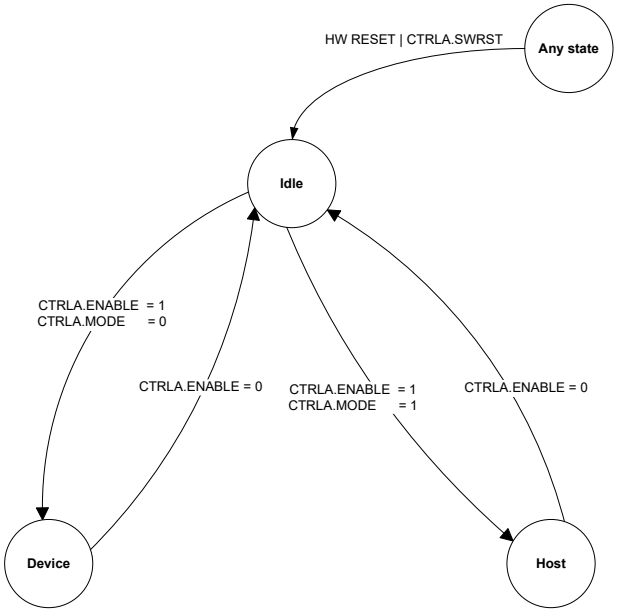
## 32.6 Functional Description

### 32.6.1 USB General Operation

#### 32.6.1.1 Initialization

After a hardware reset, the USB is disabled. The user should first enable the USB (CTRLA.ENABLE) in either device mode or host mode (CTRLA.MODE).

**Figure 32-2. General States**




After a hardware reset, the USB is in the idle state. In this state:

- The module is disabled. The USB Enable bit in the Control A register (CTRLA.ENABLE) is reset.
- The module clock is stopped in order to minimize power consumption.
- The USB pad is in suspend mode.
- The internal states and registers of the device and host are reset.

Before using the USB, the Pad Calibration register (PADCAL) must be loaded with production calibration values from the NVM Software Calibration Area.

The USB is enabled by writing a '1' to CTRLA.ENABLE. The USB is disabled by writing a '0' to CTRLA.ENABLE.

The USB is reset by writing a '1' to the Software Reset bit in CTRLA (CTRLA.SWRST). All registers in the USB will be reset to their initial state, and the USB will be disabled. Refer to the CTRLA register for details.

The user can configure pads and speed before enabling the USB by writing to the Operating Mode bit in the Control A register (CTRLA.MODE) and the Speed Configuration field in the Control B register (CTRLB.SPDCONF). These values are taken into account once the USB has been enabled by writing a '1' to CTRLA.ENABLE.

After writing a '1' to CTRLA.ENABLE, the USB enters device mode or host mode (according to CTRLA.MODE).

The USB can be disabled at any time by writing a '0' to CTRLA.ENABLE.

Refer to 32.6.2 USB Device Operations for the basic operation of the device mode.

Refer to 32.6.3 Host Operations for the basic operation of the host mode.

**Related Links**

10.3.2 NVM Software Calibration Area Mapping

### 32.6.2 USB Device Operations

This section gives an overview of the USB module device operation during normal transactions. For more details on general USB and USB protocol, refer to the Universal Serial Bus specification revision 2.1.

#### 32.6.2.1 Initialization

To attach the USB device to start the USB communications from the USB host, a zero should be written to the Detach bit in the Device Control B register (CTRLB.DETACH). To detach the device from the USB host, a one must be written to the CTRLB.DETACH.

After the device is attached, the host will request the USB device descriptor using the default device address zero. On successful transmission, it will send a USB reset. After that, it sends an address to be configured for the device. All further transactions will be directed to this device address. This address should be configured in the Device Address field in the Device Address register (DADD.DADD) and the Address Enable bit in DADD (DADD.ADDEN) should be written to one to accept communications directed to this address. DADD.ADDEN is automatically cleared on receiving a USB reset.

#### 32.6.2.2 Endpoint Configuration

Endpoint data can be placed anywhere in the device RAM. The USB controller accesses these endpoints directly through the AHB master (built-in DMA) with the help of the endpoint descriptors. The base address of the endpoint descriptors needs to be written in the Descriptor Address register (DESCADD) by the user. Refer also to the Endpoint Descriptor structure in 32.8.4.1 Endpoint Descriptor Structure.

Before using an endpoint, the user should configure the direction and type of the endpoint in Type of Endpoint field in the Device Endpoint Configuration register (EPCFG.EPTYPE0/1). The endpoint descriptor registers should be initialized to known values before using the endpoint, so that the USB controller does not read random values from the RAM.

The Endpoint Size field in the Packet Size register (PCKSIZE.SIZE) should be configured as per the size reported to the host for that endpoint. The Address of Data Buffer register (ADDR) should be set to the data buffer used for endpoint transfers.

The RAM Access Interrupt bit in Device Interrupt Flag register (INTFLAG.RAMACER) is set when a RAM access underflow error occurs during IN data stage.

When an endpoint is disabled, the following registers are cleared for that endpoint:

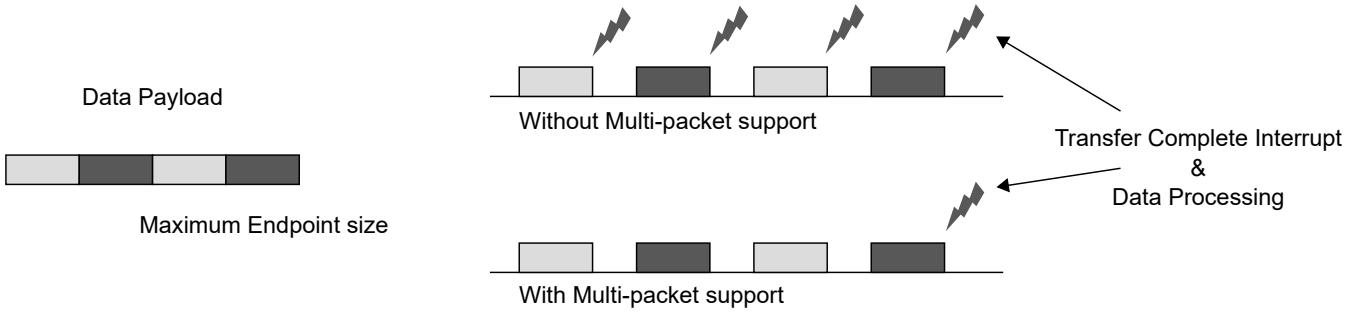
- Device Endpoint Interrupt Enable Clear/Set (EPINTENCLR/SET) register
- Device Endpoint Interrupt Flag (EPINTFLAG) register
- Transmit Stall 0 bit in the Endpoint Status register (EPSTATUS.STALLRQ0)
- Transmit Stall 1 bit in the Endpoint Status register (EPSTATUS.STALLRQ1)

#### 32.6.2.3 Multi-Packet Transfers

Multi-packet transfer enables a data payload exceeding the endpoint maximum transfer size to be transferred as multiple packets without software intervention. This reduces the number of interrupts and software intervention required to manage higher level USB transfers. Multi-packet transfer is identical to the IN and OUT transactions described below unless otherwise noted in this section.

The application software provides the size and address of the RAM buffer to be proceeded by the USB module for a specific endpoint, and the USB module will split the buffer in the required USB data transfers without any software intervention.

**Figure 32-3. Multi-Packet Feature - Reduction of CPU Overhead**




#### 32.6.2.4 USB Reset

The USB bus reset is initiated by a connected host and managed by hardware.

During USB reset the following registers are cleared:

- Device Endpoint Configuration (EPCFG) register - except for Endpoint 0
- Device Frame Number (FNUM) register
- Device Address (DADD) register
- Device Endpoint Interrupt Enable Clear/Set (EPINTENCLR/SET) register
- Device Endpoint Interrupt Flag (EPINTFLAG) register
- Transmit Stall 0 bit in the Endpoint Status register (EPSTATUS.STALLRQ0)
- Transmit Stall 1 bit in the Endpoint Status register (EPSTATUS.STALLRQ1)
- Endpoint Interrupt Summary (EPINTSMRY) register
- Upstream resume bit in the Control B register (CTRLB.UPRSM)

At the end of the reset process, the End of Reset bit is set in the Interrupt Flag register (INTFLAG.EORST).

#### 32.6.2.5 Start-of-Frame

When a Start-of-Frame (SOF) token is detected, the frame number from the token is stored in the Frame Number field in the Device Frame Number register (FNUM.FNUM), and the Start-of-Frame interrupt bit in the Device Interrupt Flag register (INTFLAG.SOF) is set. If there is a CRC or bit-stuff error, the Frame Number Error status flag (FNUM.FNCERR) in the FNUM register is set.

#### 32.6.2.6 Management of SETUP Transactions

When a SETUP token is detected and the device address of the token packet does not match DADD.DADD, the packet is discarded and the USB module returns to idle and waits for the next token packet.

When the address matches, the USB module checks if the endpoint is enabled in EPCFG. If the addressed endpoint is disabled, the packet is discarded and the USB module returns to idle and waits for the next token packet.

When the endpoint is enabled, the USB module then checks on the EPCFG of the addressed endpoint. If the EPCFG.EPTYPE0 is not set to control, the USB module returns to idle and waits for the next token packet.

When the EPCFG.EPTYPE0 matches, the USB module then fetches the Data Buffer Address (ADDR) from the addressed endpoint's descriptor and waits for a DATA0 packet. If a PID error or any other PID than DATA0 is detected, the USB module returns to idle and waits for the next token packet.

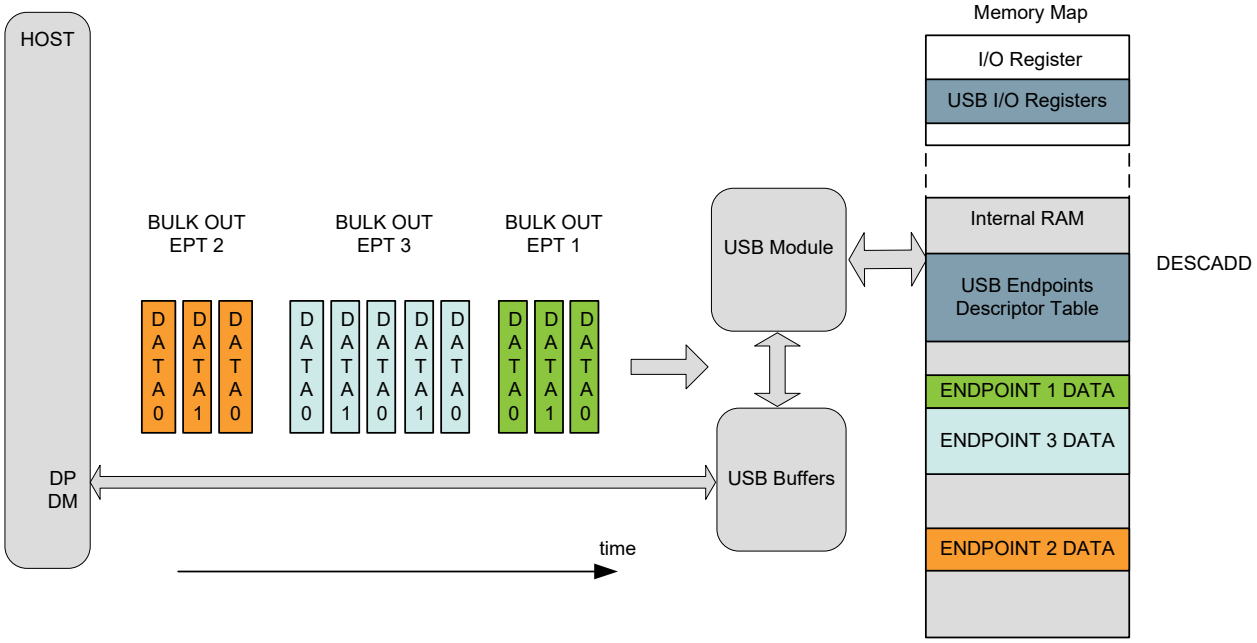
When the data PID matches and if the Received Setup Complete interrupt bit in the Device Endpoint Interrupt Flag register (EPINTFLAG.RXSTP) is equal to zero, ignoring the Bank 0 Ready bit in the Device Endpoint Status register (EPSTATUS.BK0RDY), the incoming data is written to the data buffer pointed to by the Data Buffer Address (ADDR). If the number of received data bytes exceeds the endpoint's maximum data payload size as specified by the PCKSIZE.SIZE, the remainders of the received data bytes are discarded. The packet will still be checked for bit-stuff and CRC errors. Software must never report a endpoint size to the host that is greater than the value configured in PCKSIZE.SIZE. If a bit-stuff or CRC error is detected in the packet, the USB module returns to idle and waits for the next token packet.

If data is successfully received, an ACK handshake is returned to the host, and the number of received data bytes, excluding the CRC, is written to the Byte Count (PCKSIZE.BYTE_COUNT). If the number of received data bytes is the maximum data payload specified by PCKSIZE.SIZE, no CRC data is written to the data buffer. If the number of received data bytes is the maximum data payload specified by PCKSIZE.SIZE minus one, only the first CRC data is written to the data buffer. If the number of received data is equal or less than the data payload specified by PCKSIZE.SIZE minus two, both CRC data bytes are written to the data buffer.

Finally the EPSTATUS is updated. Data Toggle OUT bit (EPSTATUS.DTGLOUT), the Data Toggle IN bit (EPSTATUS.DTGLIN), the current bank bit (EPSTATUS.CURRBK) and the Bank Ready 0 bit (EPSTATUS.BK0RDY) are set. Bank Ready 1 bit (EPSTATUS.BK1RDY) and the Stall Bank 0/1 bit (EPSTATUS.STALLQR0/1) are cleared on receiving the SETUP request. The RXSTP bit is set and triggers an interrupt if the Received Setup Interrupt Enable bit is set in Endpoint Interrupt Enable Set/Clear register (EPINTENSET/CLR.RXSTP).

### 32.6.2.7 Management of OUT Transactions

**Figure 32-4. OUT Transfer: Data Packet Host to USB Device**



When an OUT token is detected, and the device address of the token packet does not match DADD.DADD, the packet is discarded and the USB module returns to idle and waits for the next token packet.

If the address matches, the USB module checks if the endpoint number received is enabled in the EPCFG of the addressed endpoint. If the addressed endpoint is disabled, the packet is discarded and the USB module returns to idle and waits for the next token packet.

When the endpoint is enabled, the USB module then checks the Endpoint Configuration register (EPCFG) of the addressed output endpoint. If the type of the endpoint (EPCFG.EPTYPE0) is not set to OUT, the USB module returns to idle and waits for the next token packet.

The USB module then fetches the Data Buffer Address (ADDR) from the addressed endpoint's descriptor, and waits for a DATA0 or DATA1 packet. If a PID error or any other PID than DATA0 or DATA1 is detected, the USB module returns to idle and waits for the next token packet.

If EPSTATUS.STALLRQ0 in EPSTATUS is set, the incoming data is discarded. If the endpoint is not isochronous, a STALL handshake is returned to the host and the Transmit Stall Bank 0 interrupt bit in EPINTFLAG (EPINTFLAG.STALL0) is set.

For isochronous endpoints, data from both a DATA0 and DATA1 packet will be accepted. For other endpoint types the PID is checked against EPSTATUS.DTGLOUT. If a PID mismatch occurs, the incoming data is discarded, and an ACK handshake is returned to the host.

If EPSTATUS.BK0RDY is set, the incoming data is discarded, the bit Transmit Fail 0 interrupt bit in EPINTFLAG (EPINTFLAG.TRFAIL0) and the status bit STATUS_BK.ERRORFLOW are set. If the endpoint is not isochronous, a NAK handshake is returned to the host.

The incoming data is written to the data buffer pointed to by the Data Buffer Address (ADDR). If the number of received data bytes exceeds the maximum data payload specified as PCKSIZE.SIZE, the remainders of the received data bytes are discarded. The packet will still be checked for bit-stuff and CRC errors. If a bit-stuff or CRC error is detected in the packet, the USB module returns to idle and waits for the next token packet.

If the endpoint is isochronous and a bit-stuff or CRC error in the incoming data, the number of received data bytes, excluding CRC, is written to PCKSIZE.BYTE_COUNT. Finally the EPINTFLAG.TRFAIL0 and CRC Error bit in the Device Bank Status register (STATUS_BK.CRCERR) is set for the addressed endpoint.

If data was successfully received, an ACK handshake is returned to the host if the endpoint is not isochronous, and the number of received data bytes, excluding CRC, is written to PCKSIZE.BYTE_COUNT. If the number of received data bytes is the maximum data payload specified by PCKSIZE.SIZE no CRC data bytes are written to the data buffer. If the number of received data bytes is the maximum data payload specified by PCKSIZE.SIZE minus one, only the first CRC data byte is written to the data buffer If the number of received data is equal or less than the data payload specified by PCKSIZE.SIZE minus two, both CRC data bytes are written to the data buffer.

Finally in EPSTATUS for the addressed output endpoint, EPSTATUS.BK0RDY is set and EPSTATUS.DTGLOUT is toggled if the endpoint is not isochronous. The flag Transmit Complete 0 interrupt bit in EPINTFLAG (EPINTFLAG.TRCPT0) is set for the addressed endpoint.

#### 32.6.2.8 Multi-Packet Transfers for OUT Endpoint

The number of data bytes received is stored in endpoint PCKSIZE.BYTE_COUNT as for normal operation. Since PCKSIZE.BYTE_COUNT is updated after each transaction, it must be set to zero when setting up a new transfer. The total number of bytes to be received must be written to

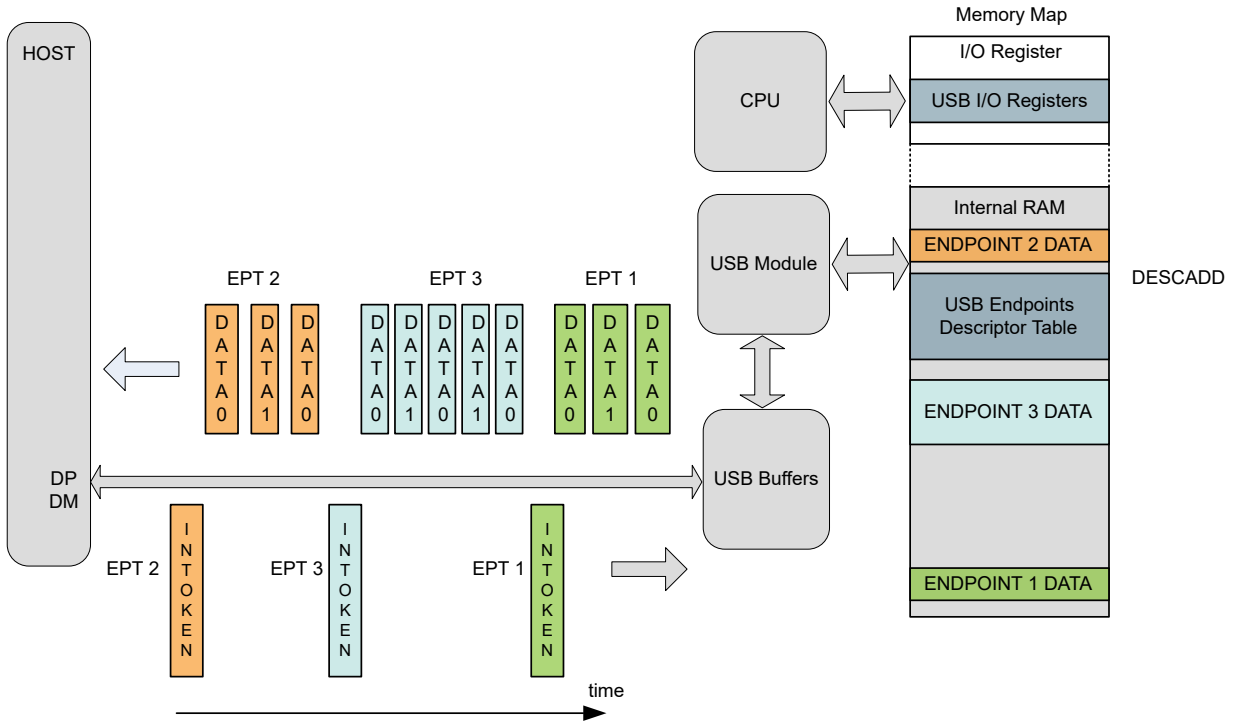
PCKSIZE.MULTI_PACKET_SIZE. This value must be a multiple of PCKSIZE.SIZE, otherwise excess data may be written to SRAM locations used by other parts of the application.

EPSTATUS.DTGLOUT management for non-isochronous packets and EPINTFLAG.BK1RDY/BK0RDY management are as for normal operation.

If a maximum payload size packet is received, PCKSIZE.BYTE_COUNT will be incremented by PCKSIZE.SIZE after the transaction has completed, and EPSTATUS.DTGLOUT will be toggled if the endpoint is not isochronous. If the updated PCKSIZE.BYTE_COUNT is equal to PCKSIZE.MULTI_PACKET_SIZE (i.e. the last transaction), EPSTATUS.BK1RDY/BK0RDY, and EPINTFLAG.TRCPT0/TRCPT1 will be set.

#### 32.6.2.9 Management of IN Transactions

**Figure 32-5. IN Transfer: Data Packet USB Device to Host After Request from Host**




When an IN token is detected, and if the device address of the token packet does not match DADD.DADD, the packet is discarded and the USB module returns to idle and waits for the next token packet.

When the address matches, the USB module checks if the endpoint received is enabled in the EPCFG of the addressed endpoint and if not, the packet is discarded and the USB module returns to idle and waits for the next token packet.

When the endpoint is enabled, the USB module then checks on the EPCFG of the addressed input endpoint. If the EPCFG.EPTYPE1 is not set to IN, the USB module returns to idle and waits for the next token packet.

If EPSTATUS.STALLRQ1 in EPSTATUS is set, and the endpoint is not isochronous, a STALL handshake is returned to the host and EPINTFLAG.STALL1 is set.

If EPSTATUS.BK1RDY is cleared, the flag EPINTFLAG.TRFAIL1 is set. If the endpoint is not isochronous, a NAK handshake is returned to the host.

The USB module then fetches the Data Buffer Address (ADDR) from the addressed endpoint's descriptor. The data pointed to by the Data Buffer Address (ADDR) is sent to the host in a DATA0 packet if the endpoint is isochronous. For non-isochronous endpoints a DATA0 or DATA1 packet is sent depending on the state of EPSTATUS.DTGLIN. When the number of data bytes specified in endpoint PCKSIZE.BYTE_COUNT is sent, the CRC is appended and sent to the host.

For isochronous endpoints, EPSTATUS.BK1RDY is cleared and EPINTFLAG.TRCPT1 is set.

For all non-isochronous endpoints the USB module waits for an ACK handshake from the host. If an ACK handshake is not received within 16 bit times, the USB module returns to idle and waits for the next token packet. If an ACK handshake is successfully received EPSTATUS.BK1RDY is cleared, EPINTFLAG.TRCPT1 is set and EPSTATUS.DTGLIN is toggled.

#### 32.6.2.10 Multi-Packet Transfers for IN Endpoint

The total number of data bytes to be sent is written to PCKSIZE.BYTE_COUNT as for normal operation. The Multi-packet size register (PCKSIZE.MULTI_PACKET_SIZE) is used to store the number of bytes that are sent, and must be written to zero when setting up a new transfer.

When an IN token is received, PCKSIZE.BYTE_COUNT and PCKSIZE.MULTI_PACKET_SIZE are fetched. If PCKSIZE.BYTE_COUNT minus PCKSIZE.MULTI_PACKET_SIZE is less than the endpoint PCKSIZE.SIZE, endpoint BYTE_COUNT minus endpoint PCKSIZE.MULTI_PACKET_SIZE bytes are transmitted, otherwise PCKSIZE.SIZE number of bytes are transmitted. If endpoint PCKSIZE.BYTE_COUNT is a multiple of PCKSIZE.SIZE, the last packet sent will be zero-length if the AUTOZLP bit is set.

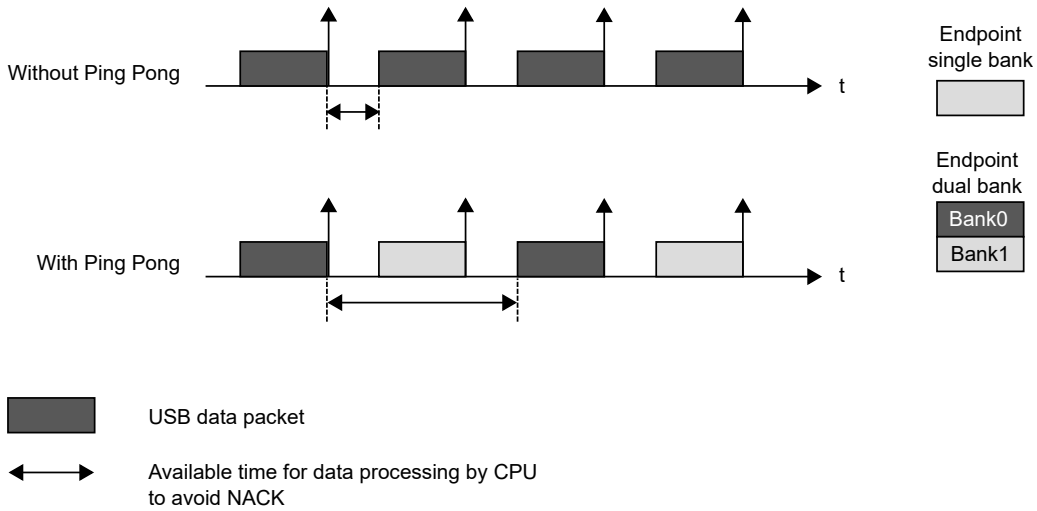
If a maximum payload size packet was sent (i.e. not the last transaction), MULTI_PACKET_SIZE will be incremented by the PCKSIZE.SIZE. If the endpoint is not isochronous the EPSTATUS.DTLGIN bit will be toggled when the transaction has completed. If a short packet was sent (i.e. the last transaction), MULTI_PACKET_SIZE is incremented by the data payload. EPSTATUS.BK0/1RDY will be cleared and EPINTFLAG.TRCPT0/1 will be set.

#### 32.6.2.11 Ping-Pong Operation

When an endpoint is configured for ping-pong operation, it uses both the input and output data buffers (banks) for a given endpoint in a single direction. The direction is selected by enabling one of the IN or OUT direction in EPCFG.EPTYPE0/1 and configuring the opposite direction in EPCFG.EPTYPE1/0 as Dual Bank.

When ping-pong operation is enabled for an endpoint, the endpoint in the opposite direction must be configured as dual bank. The data buffer, data address pointer and byte counter from the enabled endpoint are used as Bank 0, while the matching registers from the disabled endpoint are used as Bank 1.

**Figure 32-6. Ping-Pong Overview**




The Bank Select flag in EPSTATUS.CURBK indicates which bank data will be used in the next transaction, and is updated after each transaction. According to EPSTATUS.CURBK, EPINTFLAG.TRCPT0 or EPINTFLAG.TRFAIL0 or EPINTFLAG.TRCPT1 or EPINTFLAG.TRFAIL1 in EPINTFLAG and Data Buffer 0/1 ready (EPSTATUS.BK0RDY and EPSTATUS.BK1RDY) are set. The EPSTATUS.DTGLOUT and EPSTATUS.DTGLIN are updated for the enabled endpoint direction only.

#### 32.6.2.12 Feedback Operation

Feedback endpoints are endpoints with same the address but in different directions. This is usually used in explicit feedback mechanism in USB Audio, where a feedback endpoint is associated to one or more isochronous data endpoints to which it provides feedback service. The feedback endpoint always has the opposite direction from the data endpoint.

The feedback endpoint always has the opposite direction from the data endpoint(s). The feedback endpoint has the same endpoint number as the first (lower) data endpoint. A feedback endpoint can be created by configuring an endpoint with different endpoint size (PCKSIZE.SIZE) and different endpoint type (EPCFG.EPTYPE0/1) for the IN and OUT direction.

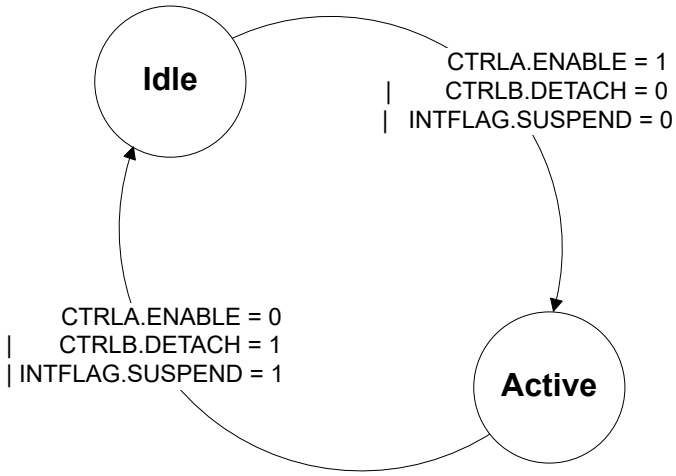
Example Configuration for Feedback Operation:

- Endpoint n / IN: EPCFG.EPTYPE1 = Interrupt IN, PCKSIZE.SIZE = 64.
- Endpoint n / OUT: EPCFG.EPTYPE0= Isochronous OUT, PCKSIZE.SIZE = 512.

#### 32.6.2.13 Suspend State and Pad Behavior

The following figure, Pad Behavior, illustrates the behavior of the USB pad in Device mode.

**Figure 32-7. Pad Behavior**

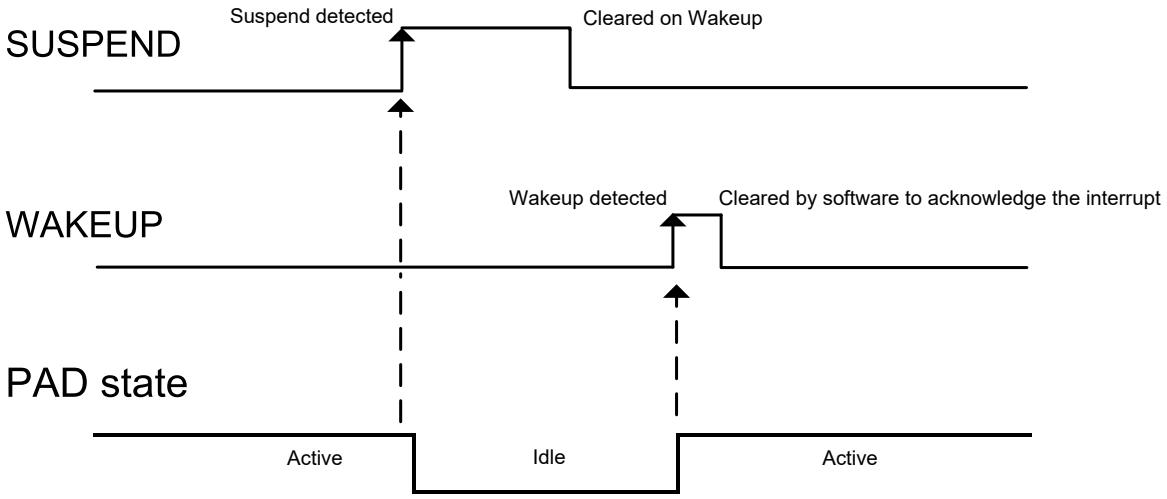



In Idle state, the pad is in Low Power Consumption mode.

In Active state, the pad is active.

The following figure, Pad Events, illustrates the pad events leading to a PAD state change.

**Figure 32-8. Pad Events**




The Suspend Interrupt bit in the Device Interrupt Flag register (INTFLAG.SUSPEND) is set when a USB Suspend state has been detected on the USB bus. The USB pad is then automatically put in the Idle state. The detection of a non-idle state sets the Wake Up Interrupt bit (INTFLAG.WAKEUP) and wakes the USB pad.

The pad goes to the Idle state if the USB module is disabled or if CTRLB.DETACH is written to one. It returns to the Active state when CTRLA.ENABLE is written to one and CTRLB.DETACH is written to zero.

#### 32.6.2.14 Remote Wakeup

The remote wakeup request (also known as upstream resume) is the only request the device may send on its own initiative. This should be preceded by a DEVICE_REMOTE_WAKEUP request from the host.

First, the USB must have detected a “Suspend” state on the bus, i.e. the remote wakeup request can only be sent after INTFLAG.SUSPEND has been set.

The user may then write a one to the Remote Wakeup bit (CTRLB.UPRSM) to send an Upstream Resume to the host initiating the wakeup. This will automatically be done by the controller after 5 ms of inactivity on the USB bus.

When the controller sends the Upstream Resume INTFLAG.WAKEUP is set and INTFLAG.SUSPEND is cleared.

The CTRLB.UPRSM is cleared at the end of the transmitting Upstream Resume.

In case of a rebroadcast resume initiated by the host, the End of Resume bit (INTFLAG.EORSM) flag is set when the rebroadcast resume is completed.

In the case where the CTRLB.UPRSM bit is set while a host initiated downstream resume is already started, the CTRLB.UPRSM is cleared and the upstream resume request is ignored.

#### 32.6.2.15 Link Power Management L1 (LPM-L1) Suspend State Entry and Exit as Device

The LPM Handshake bit in CTRLB.LPMHDSK should be configured to accept the LPM transaction.

When a LPM transaction is received on any enabled endpoint n and a handshake has been sent in response by the controller according to CTRLB.LPMHDSK, the Device Link Power Manager (EXTREG) register is updated in the bank 0 of the addressed endpoint's descriptor. It contains information such as the Best Effort Service Latency (BESL), the Remote Wake bit (bRemoteWake), and the Link State parameter (bLinkState). Usually, the LPM transaction uses only the endpoint number 0.

If the LPM transaction was positively acknowledged (ACK handshake), USB sets the Link Power Management Interrupt bit (INTFLAG.LPMSUSP) bit which indicates that the USB transceiver is suspended, reducing power consumption. This suspend occurs 9 microseconds after the LPM transaction according to the specification.

To further reduce consumption, it is recommended to stop the USB clock while the device is suspended.

The MCU can also enter in one of the available sleep modes if the wakeup time latency of the selected sleep mode complies with the host latency constraint (see the BESL parameter in 32.8.4.4 EXTREG register).

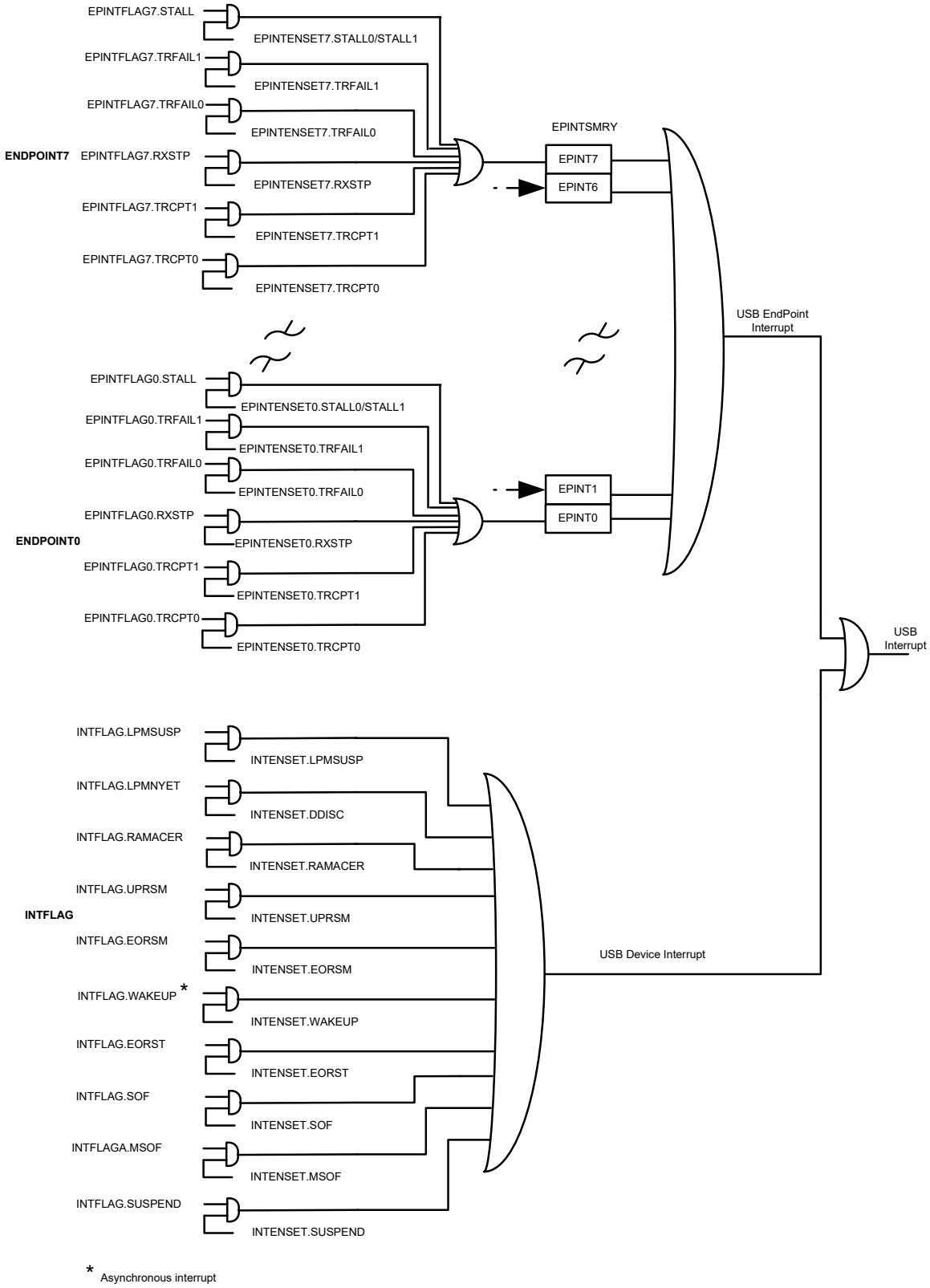
Recovering from this LPM-L1 suspend state is exactly the same as the Suspend state (see Section 32.6.2.13 Suspend State and Pad Behavior) except that the remote wakeup duration initiated by USB is shorter to comply with the Link Power Management specification.

If the LPM transaction is responded with a NYET, the Link Power Management Not Yet Interrupt Flag (INTFLAG.LPMNYET) is set. This generates an interrupt if the Link Power Management Not Yet Interrupt Enable bit (INTENCLR/SET.LPMNYET) is set.

If the LPM transaction is responded with a STALL or no handshake, no flag is set, and the transaction is ignored.

### 32.6.2.16 USB Device Interrupt

**Figure 32-9. Device Interrupt**




* Asynchronous interrupt

The WAKEUP is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.

### 32.6.3 Host Operations

This section gives an overview of the USB module Host operation during normal transactions. For more details on general USB and USB protocol, refer to Universal Serial Bus Specification revision 2.1.

#### 32.6.3.1 Device Detection and Disconnection

Prior to device detection the software must set the VBUS is OK bit (CTRLB.VBUSOK) register when the VBUS is available. This notifies the USB host that USB operations can be started. When the bit CTRLB.VBUSOK is zero and even if the USB HOST is configured and enabled, host operation is halted. Setting the bit CTRLB.VBUSOK will allow host operation when the USB is configured.

The Device detection is managed by the software using the Line State field in the Host Status (STATUS.LINESTATE) register. The device connection is detected by the host controller when DP or DM is pulled high, depending of the speed of the device.

The device disconnection is detected by the host controller when both DP and DM are pulled down using the STATUS.LINESTATE registers.

The Device Connection Interrupt bit (INTFLAG.DCONN) is set if a device connection is detected.

The Device Disconnection Interrupt bit (INTFLAG.DDISC) is set if a device disconnection is detected.

#### 32.6.3.2 Host Terminology

In host mode, the term pipe is used instead of endpoint. A host pipe corresponds to a device endpoint, refer to "Universal Serial Bus Specification revision 2.1." for more information.

#### 32.6.3.3 USB Reset

The USB sends a USB reset signal when the user writes a one to the USB Reset bit (CTRLB.BUSRESET). When the USB reset has been sent, the USB Reset Sent Interrupt bit in the INTFLAG (INTFLAG.RST) is set and all pipes will be disabled.

If the bus was previously in a suspended state (i.e., the Start of Frame Generation Enable bit (CTRLB.SOFE) is zero), the USB will switch it to the Resume state, causing the bus to asynchronously set the Host Wakeup Interrupt flag (INTFLAG.WAKEUP). The CTRLB.SOFE bit will be set in order to generate SOFs immediately after the USB reset.

During USB reset the following registers are cleared:

- All Host Pipe Configuration register (PCFG)
- Host Frame Number register (FNUM)
- Interval for the Bulk-Out/Ping transaction register (BINTERVAL)
- Host Start-of-Frame Control register (HSOFC)
- Pipe Interrupt Enable Clear/Set register (PINTENCLR/SET)
- Pipe Interrupt Flag register (PINTFLAG)
- Pipe Freeze bit in Pipe Status register (PSTATUS.FREEZE)

After the reset the user should check the Speed Status field in the Status register (STATUS.SPEED) to find out the current speed according to the capability of the peripheral.

#### 32.6.3.4 Pipe Configuration

Pipe data can be placed anywhere in the RAM. The USB controller accesses these pipes directly through the AHB master (built-in DMA) with the help of the pipe descriptors. The base address of the pipe descriptors needs to be written in the Descriptor Address register (DESCADD) by the user. Refer also to 32.8.7.1 Pipe Descriptor Structure.

Before using a pipe, the user should configure the direction and type of the pipe in Type of Pipe field in the Host Pipe Configuration register (PCFG.PTYPE). The pipe descriptor registers should be initialized to

known values before using the pipe, so that the USB controller does not read the random values from the RAM.

The Pipe Size field in the Packet Size register (PCKSIZE.SIZE) should be configured as per the size reported by the device for the endpoint associated with this pipe. The Address of Data Buffer register (ADDR) should be set to the data buffer used for pipe transfers.

The Pipe Bank bit (PCFG.BK) should be set to one if dual banking is desired. Dual bank is not supported for Control pipes.

The Ram Access Interrupt bit in Host Interrupt Flag register (INTFLAG.RAMACER) is set when a RAM access underflow error occurs during an OUT stage.

When a pipe is disabled, the following registers are cleared for that pipe:

- Interval for the Bulk-Out/Ping transaction register (BINTERVAL)
- Pipe Interrupt Enable Clear/Set register (PINTENCLR/SET)
- Pipe Interrupt Flag register (PINTFLAG)
- Pipe Freeze bit in Pipe Status register (PSTATUS.FREEZE)

#### 32.6.3.5 Pipe Activation

A disabled pipe is inactive, and will be reset along with its context registers (pipe registers for the pipe n). Pipes are enabled by writing the Type of the Pipe bit (PCFG.PTYPE) to a value different than 0x0 (disabled).

When a pipe is enabled, the Pipe Freeze bit in the Pipe Status register (PSTATUS.FREEZE) is set. This allows the user to complete the configuration of the pipe, without starting a USB transfer.

When starting an enumeration, the user retrieves the device descriptor by sending a GET_DESCRIPTOR USB request. This descriptor contains the maximal packet size of the device default control endpoint (bMaxPacketSize0), which the user should use to reconfigure the size of the default control pipe.

#### 32.6.3.6 Pipe Address Setup

Once the device has answered the first host requests with the default device address 0, the host assigns a new address to the device. The host controller has to send a USB reset to the device and a SET_ADDRESS(addr) SETUP request with the new address to be used by the device. Once this SETUP transaction is complete, the user writes the new address to the Pipe Device Address field in the Host Control Pipe register (CTRL_PIPE.PDADDR) in Pipe descriptor. All following requests by this pipe will be performed using this new address.

#### 32.6.3.7 Suspend and Wakeup

Setting CTRLB.SOFE to zero when in host mode will cause the USB to cease sending Start-of-Frames on the USB bus and enter the Suspend state. The USB device will enter the Suspend state 3ms later.

Before entering suspend by writing CTRLB.SOFE to zero, the user must freeze the active pipes by setting their PSTATUS.FREEZE bit. Any current on-going pipe will complete its transaction, and then all pipes will be inactive. The user should wait at least 1 complete frame before entering the suspend mode to avoid any data loss.

The device can awaken the host by sending an Upstream Resume (Remote Wakeup feature). When the host detects a non-idle state on the USB bus, it sets the INTFLAG.WAKEUP. If the non-idle bus state corresponds to an Upstream Resume (K state), the Upstream Resume Received Interrupt bit in INTFLAG (INTFLAG.UPRSM) is set and the user must generate a Downstream Resume within 1 ms and for at least 20 ms. It is required to first write a one to the Send USB Resume bit in CTRLB (CTRLB.RESUME) to respond to the upstream resume with a downstream resume. Alternatively, the host can resume from a

suspend state by sending a Downstream Resume on the USB bus (CTRLB.RESUME set to 1). In both cases, when the downstream resume is completed, the CTRLB.SOFE bit is automatically set and the host enters again the active state.

#### 32.6.3.8 Phase-locked SOFs

To support the Synchronous Endpoints capability, the period of the emitted Start-of-Frame is maintained while the USB connection is not in the active state. This does not apply for the disconnected/connected/reset states. It applies for active/idle/suspend/resume states. The period of Start-of-Frame will be 1ms when the USB connection is in active state and an integer number of milli-seconds across idle/suspend/resume states.

To ensure the Synchronous Endpoints capability, the GCLK_USB clock must be kept running. If the GCLK_USB is interrupted, the period of the emitted Start-of-Frame will be erratic.

#### 32.6.3.9 Management of Control Pipes

A control transaction is composed of three stages:

- SETUP
- Data (IN or OUT)
- Status (IN or OUT)

The user has to change the pipe token according to each stage using the Pipe Token field in PCFG (PCFG.PTOKEN).

For control pipes only, the token is assigned a specific initial data toggle sequence:

- SETUP: Data0
- IN: Data1
- OUT: Data1

#### 32.6.3.10 Management of IN Pipes

IN packets are sent by the USB device controller upon IN request reception from the host. All the received data from the device to the host will be stored in the bank provided the bank is empty. The pipe and its descriptor in RAM must be configured.

The host indicates it is able to receive data from the device by clearing the Bank 0/1 Ready bit in PSTATUS (PSTATUS.BK0/1RDY), which means that the memory for the bank is available for new USB transfer.

The USB will perform IN requests as long as the pipe is not frozen by the user.

The generation of IN requests starts when the pipe is unfrozen (PSTATUS.PFREEZE is set to zero).

When the current bank is full, the Transmit Complete 0/1 bit in PINTFLAG (PINTFLAG.TRCPT0/1) will be set and trigger an interrupt if enabled and the PSTATUS.BK0/1RDY bit will be set.

PINTFLAG.TRCPT0/1 must be cleared by software to acknowledge the interrupt. This is done by writing a one to the PINTFLAG.TRCPT0/1 of the addressed pipe.

The user reads the PCKSIZE.BYTE_COUNT to know how many bytes should be read.

To free the bank the user must read the IN data from the address ADDR in the pipe descriptor and clear the PKSTATUS.BK0/1RDY bit. When the IN pipe is composed of multiple banks, a successful IN transaction will switch to the next bank. Another IN request will be performed by the host as long as the PSTATUS.BK0/1RDY bit for that bank is set. The PINTFLAG.TRCPT0/1 and PSTATUS.BK0/1RDY will be updated accordingly.

© 2018 Microchip Technology Inc.

The user can follow the current bank looking at Current Bank bit in PSTATUS (PSTATUS.CURBK) and by looking at Data Toggle for IN pipe bit in PSTATUS (PSTATUS.DTGLIN).

When the pipe is configured as single bank (Pipe Bank bit in PCFG (PCFG.BK) is 0), only PINTFLAG.TRCPT0 and PSTATUS.BK0 are used. When the pipe is configured as dual bank (PCFG.BK is 1), both PINTFLAG.TRCPT0/1 and PSTATUS.BK0/1 are used.

#### 32.6.3.11 Management of OUT Pipes

OUT packets are sent by the host. All the data stored in the bank will be sent to the device provided the bank is filled. The pipe and its descriptor in RAM must be configured.

The host can send data to the device by writing to the data bank 0 in single bank or the data bank 0/1 in dual bank.

The generation of OUT packet starts when the pipe is unfrozen (PSTATUS.PFREEZE is zero).

The user writes the OUT data to the data buffer pointer by ADDR in the pipe descriptor and allows the USB to send the data by writing a one to the PSTATUS.BK0/1RDY. This will also cause a switch to the next bank if the OUT pipe is part of a dual bank configuration.

PINTFLAGn.TRCPT0/1 must be cleared before setting PSTATUS.BK0/1RDY to avoid missing an PINTFLAGn.TRCPT0/1 event.

#### 32.6.3.12 Alternate Pipe

The user has the possibility to run sequentially several logical pipes on the same physical pipe. It allows addressing of any device endpoint of any attached device on the bus.

Before switching pipe, the user should save the pipe context (Pipe registers and descriptor for pipe n).

After switching pipe, the user should restore the pipe context (Pipe registers and descriptor for pipe n) and in particular PCFG, and PSTATUS.

#### 32.6.3.13 Data Flow Error

This error exists only for isochronous and interrupt pipes for both IN and OUT directions. It sets the Transmit Fail bit in PINTFLAG (PINTFLAG.TRFAIL), which triggers an interrupt if the Transmit Fail bit in PINTENCLR/SET(PINTENCLR/SET.TRFAIL) is set. The user must check the Pipe Interrupt Summary register (PINTSMRY) to find out the pipe which triggered the interrupt. Then the user must check the origin of the interrupt's bank by looking at the Pipe Bank Status register (STATUS_BK) for each bank. If the Error Flow bit in the STATUS_BK (STATUS_BK.ERRORFLOW) is set then the user is able to determine the origin of the data flow error. As the user knows that the endpoint is an IN or OUT the error flow can be deduced as OUT underflow or as an IN overflow.

An underflow can occur during an OUT stage if the host attempts to send data from an empty bank. If a new transaction is successful, the relevant bank descriptor STATUS_BK.ERRORFLOW will be cleared.

An overflow can occur during an IN stage if the device tries to send a packet while the bank is full. Typically this occurs when a CPU is not fast enough. The packet data is not written to the bank and is lost. If a new transaction is successful, the relevant bank descriptor STATUS_BK.ERRORFLOW will be cleared.

#### 32.6.3.14 CRC Error

This error exists only for isochronous IN pipes. It sets the PINTFLAG.TRFAIL, which triggers an interrupt if PINTENCLR/SET.TRFAIL is set. The user must check the PINTSMRY to find out the pipe which triggered the interrupt. Then the user must check the origin of the interrupt's bank by looking at the bank descriptor STATUS_BK for each bank and if the CRC Error bit in STATUS_BK (STATUS_BK.CRCERR) is set then the user is able to determine the origin of the CRC error. A CRC error can occur during the IN

stage if the USB detects a corrupted packet. The IN packet will remain stored in the bank and PINTFLAG.TRCPT0/1 will be set.

#### 32.6.3.15 PERR Error

This error exists for all pipes. It sets the PINTFLAG.PERR Interrupt, which triggers an interrupt if PINTFLAG.PERR is set. The user must check the PINTSMRY register to find out the pipe which can cause an interrupt.

A PERR error occurs if one of the error field in the STATUS_PIPE register in the Host pipe descriptor is set and the Error Count field in STATUS_PIPE (STATUS_PIPE.ERCNT) exceeds the maximum allowed number of Pipe error(s) as defined in Pipe Error Max Number field in CTRL_PIPE (CTRL_PIPE.PERMAX). Refer to section 32.8.7.7 STATUS_PIPE register.

If one of the error field in the STATUS_PIPE register from the Host Pipe Descriptor is set and the STATUS_PIPE.ERCNT is less than the CTRL_PIPE.PERMAX, the STATUS_PIPE.ERCNT is incremented.

#### 32.6.3.16 Link Power Management L1 (LPM-L1) Suspend State Entry and Exit as Host.

An EXTENDED LPM transaction can be transmitted by any enabled pipe. The PCFGn.PTYPE should be set to EXTENDED. Other fields as PCFG.PTOKEN, PCFG.BK and PCKSIZE.SIZE are irrelevant in this configuration. The user should also set the EXTREG.VARIABLE in the descriptor as described in 32.8.7.4 EXTREG register.

When the pipe is configured and enabled, an EXTENDED TOKEN followed by a LPM TOKEN are transmitted. The device responds with a valid HANDSHAKE, corrupted HANDSHAKE or no HANDSHAKE (TIME-OUT).

If the valid HANDSHAKE is an ACK, the host will immediately proceed to L1 SLEEP and the PINTFLAG.TRCT0 is set. The minimum duration of the L1 SLEEP state will be the TL1RetryAndResidency as defined in the reference document "ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum". When entering the L1 SLEEP state, the CTRLB.SOFE is cleared, avoiding Start-of-Frame generation.

If the valid HANDSHAKE is a NYET PINTFLAG.TRFAIL is set.

If the valid HANDSHAKE is a STALL the PINTFLAG.STALL is set.

If there is no HANDSHAKE or corrupted HANDSHAKE, the EXTENDED/LPM pair of TOKENS will be transmitted again until reaching the maximum number of retries as defined by the CTRL_PIPE.PERMAX in the pipe descriptor.

If the last retry returns no valid HANDSHAKE, the PINTFLAGn.PERR is set, and the STATUS_BK is updated in the pipe descriptor.

All LPM transactions, should they end up with a ACK, a NYET, a STALL or a PERR, will set the PSTATUS.PFREEZE bit, freezing the pipe before a succeeding operation. The user should unfreeze the pipe to start a new LPM transaction.

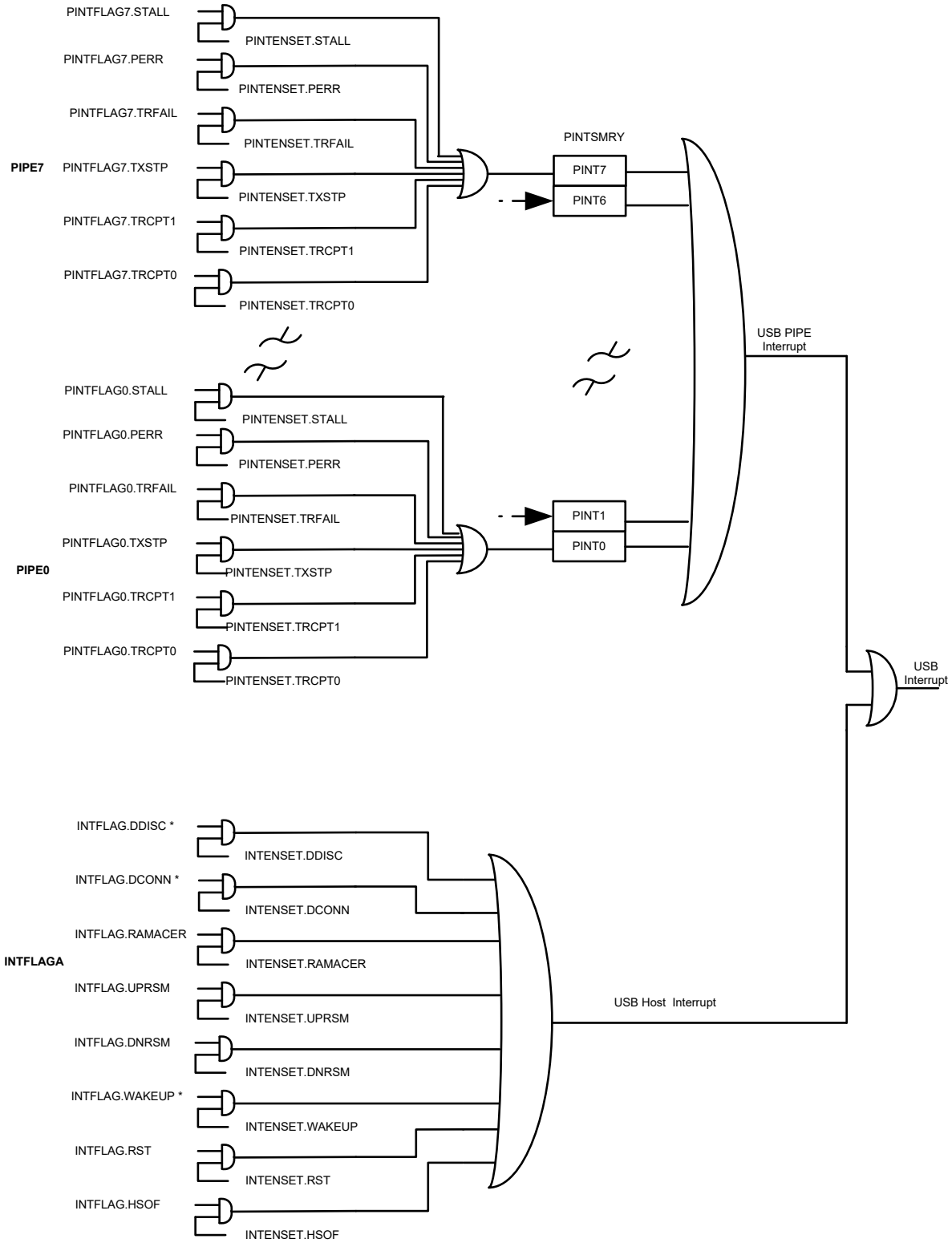
To exit the L1 STATE, the user initiate a DOWNSTREAM RESUME by setting the bit CTRLB.RESUME or a L1 RESUME by setting the Send L1 Resume bit in CTRLB (CTRLB.L1RESUME). In the case of a L1 RESUME, the K STATE duration is given by the BESL bit field in the EXTREG.VARIABLE field. See 32.8.7.4 EXTREG.

When the host is in the L1 SLEEP state after a successful LPM transmitted, the device can initiate an UPSTREAM RESUME. This will set the Upstream Resume Interrupt bit in INTFLAG (INTFLAG.UPRSM). The host should proceed then to a L1 RESUME as described above.

After resuming from the L1 SLEEP state, the bit CTRLB.SOFE is set, allowing Start-of-Frame generation.

### 32.6.3.17 Host Interrupt

**Figure 32-10. Host Interrupt**




* Asynchronous interrupt

The WAKEUP is an asynchronous interrupt and can be used to wake-up the device from any sleep mode.

## 32.7 Register Summary

The register mapping depends on the Operating Mode field in the Control A register (CTRLA.MODE). The register summary is detailed below.

### 32.7.1 Common Device Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | MODE | | | | | RUNSTBY | ENABLE | SWRST |
| 0x01 | Reserved | | | | | | | | | |
| 0x02 | SYNCBUSY | 7:0 | | | | | | | ENABLE | SWRST |
| 0x03 | QOSCTRL | 7:0 | | | | | DQOS[1:0] | | CQOS[1:0] | |
| 0x0D | FSMSTATUS | 7:0 | | FSMSTATE[6:0] | | | | | | |
| 0x24 | DESCADD | 7:0 | DESCADD[7:0] | | | | | | | |
| 0x25 | | 15:8 | DESCADD[15:8] | | | | | | | |
| 0x26 | | 23:16 | DESCADD[23:16] | | | | | | | |
| 0x27 | | 31:24 | DESCADD[31:24] | | | | | | | |
| 0x28 | PADCAL | 7:0 | TRANSN[1:0] | | | TRANSP[4:0] | | | | |
| 0x29 | | 15:8 | | TRIM[2:0] | | | | TRANSN[4:2] | | |

### 32.7.2 Device Summary

**Table 32-1. General Device Registers**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04 | Reserved | | | | | | | | | |
| 0x05 | Reserved | | | | | | | | | |
| 0x06 | Reserved | | | | | | | | | |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | CTRLB | 7:0 | | | | NREPLY | SPDCONF[1:0] | | UPRSM | DETACH |
| 0x09 | | 15:8 | | | | | LPMHDSK[1:0] | | GNAK | |
| 0x0A | DADD | | ADDEN | DADD[6:0] | | | | | | |
| 0x0B | Reserved | | | | | | | | | |
| 0x0C | STATUS | 7:0 | LINESTATE[1:0] | | | | SPEED[1:0] | | | |
| 0x0E | Reserved | | | | | | | | | |
| 0x0F | Reserved | | | | | | | | | |
| 0x10 | FNUM | 7:0 | FNUM[4:0] | | | | | | | |
| 0x11 | | 15:8 | FNCERR | | FNUM[10:5] | | | | | |
| 0x12 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |
| 0x15 | | 15:8 | | | | | | | LPMSUSP | LPMNYET |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTENSET | 7:0 | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |
| 0x19 | | 15:8 | | | | | | | LPMSUSP | LPMNYET |
| 0x1A | Reserved | | | | | | | | | |
| 0x1B | Reserved | | | | | | | | | |
| 0x1C | INTFLAG | 7:0 | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1D | | 15:8 | | | | | | | LPMSUSP | LPMNYET |
| 0x1E | Reserved | | | | | | | | | |
| 0x1F | Reserved | | | | | | | | | |
| 0x20 | EPINTSMRY | 7:0 | EPINT[7:0] | | | | | | | |
| 0x21 | | 15:8 | EPINT[15:8] | | | | | | | |
| 0x22 | Reserved | | | | | | | | | |
| 0x23 | Reserved | | | | | | | | | |

**Table 32-2. Device Endpoint Register n**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1m0 | EPCFGn | 7:0 | | EPTYPE1[1:0] | | | | EPTYPE0[1:0] | | |
| 0x1m1 | Reserved | | | | | | | | | |
| 0x1m2 | Reserved | | | | | | | | | |
| 0x1m3 | Reserved | | | | | | | | | |
| 0x1m4 | EPSTATUSCLRn | 7:0 | BK1RDY | BK0RDY | STALLRQ1 | STALLRQ0 | | CURBK | DTGLIN | DTGLOUT |
| 0x1m5 | EPSTATUSSETn | 7:0 | BK1RDY | BK0RDY | STALLRQ1 | STALLRQ0 | | CURBK | DTGLIN | DTGLOUT |
| 0x1m6 | EPSTATUSn | 7:0 | BK1RDY | BK0RDY | STALLRQ1 | STALLRQ0 | | CURBK | DTGLIN | DTGLOUT |
| 0x1m7 | EPINTFLAGn | 7:0 | | STALL1 | STALL0 | RXSTP | TRFAIL1 | TRFAIL0 | TRCPT1 | TRCPT0 |
| 0x1m8 | EPINTENCLRn | 7:0 | | STALL1 | STALL0 | RXSTP | TRFAIL1 | TRFAIL0 | TRCPT1 | TRCPT0 |
| 0x1m9 | EPINTENSETn | 7:0 | | STALL1 | STALL0 | RXSTP | TRFAIL1 | TRFAIL0 | TRCPT1 | TRCPT0 |
| 0x1mA | Reserved | | | | | | | | | |
| 0x1mB | Reserved | | | | | | | | | |

**Table 32-3. Device Endpoint n Descriptor Bank 0**

| Offset 0x n0 + index | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ADDR | 7:0 | ADD[7:0] | | | | | | | |
| 0x01 | | 15:8 | ADD[15:8] | | | | | | | |
| 0x02 | | 23:16 | ADD[23:16] | | | | | | | |
| 0x03 | | 31:24 | ADD[31:24] | | | | | | | |
| 0x04 | PCKSIZE | 7:0 | BYTE_COUNT[7:0] | | | | | | | |
| 0x05 | | 15:8 | MULTI_PACKET_SIZE[1:0] | | BYTE_COUNT[13:8] | | | | | |
| 0x06 | | 23:16 | MULTI_PACKET_SIZE[9:2] | | | | | | | |
| 0x07 | | 31:24 | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | |
| 0x08 | EXTREG | 7:0 | VARIABLE[3:0] | | | | SUBPID[3:0] | | | |
| 0x09 | | 15:8 | | VARIABLE[10:4] | | | | | | |
| 0x0A | STATUS_BK | 7:0 | | | | | | | ERRORFLOW | CRCERR |
| 0x0B | Reserved | 7:0 | | | | | | | | |
| 0x0C | Reserved | 7:0 | | | | | | | | |
| 0x0D | Reserved | 7:0 | | | | | | | | |
| 0x0E | Reserved | 7:0 | | | | | | | | |
| 0x0F | Reserved | 7:0 | | | | | | | | |

**Table 32-4. Device Endpoint n Descriptor Bank 1**

| Offset 0x n0 + 0x10 + index | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ADDR | 7:0 | ADD[7:0] | | | | | | | |
| 0x01 | | 15:8 | ADD[15:8] | | | | | | | |
| 0x02 | | 23:16 | ADD[23:16] | | | | | | | |
| 0x03 | | 31:24 | ADD[31:24] | | | | | | | |
| 0x04 | PCKSIZE | 7:0 | BYTE_COUNT[7:0] | | | | | | | |
| 0x05 | | 15:8 | MULTI_PACKET_SIZE[1:0] | | BYTE_COUNT[13:8] | | | | | |
| 0x06 | | 23:16 | MULTI_PACKET_SIZE[9:2] | | | | | | | |
| 0x07 | | 31:24 | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | |
| 0x08 | Reserved | 7:0 | | | | | | | | |
| 0x09 | Reserved | 15:8 | | | | | | | | |
| 0x0A | STATUS_BK | 7:0 | | | | | | | ERRORFLOW | CRCERR |
| 0x0B | Reserved | 7:0 | | | | | | | | |
| 0x0C | Reserved | 7:0 | | | | | | | | |
| 0x0D | Reserved | 7:0 | | | | | | | | |
| 0x0E | Reserved | 7:0 | | | | | | | | |
| 0x0F | Reserved | 7:0 | | | | | | | | |

### 32.7.3 Host Summary

**Table 32-5. General Host Registers**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04 | Reserved | | | | | | | | | |
| 0x05 | Reserved | | | | | | | | | |
| 0x06 | Reserved | | | | | | | | | |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | CTRLB | 7:0 | | TSTK | TSTJ | | SPDCONF[1:0] | | RESUME | |
| 0x09 | | 15:8 | | | | | L1RESUME | VBUSOK | BUSRESET | SOFE |
| 0x0A | HSOFC | 7:0 | FLENCE | | | | FLENC[3:0] | | | |
| 0x0B | Reserved | | | | | | | | | |
| 0x0C | STATUS | 7:0 | LINESTATE[1:0] | | | | SPEED[1:0] | | | |
| 0x0E | Reserved | | | | | | | | | |
| 0x0F | Reserved | | | | | | | | | |
| 0x10 | FNUM | 7:0 | FNUM[4:0] | | | | | | | |
| 0x11 | | 15:8 | | | FNUM[10:5] | | | | | |
| 0x12 | FLENHIGH | 7:0 | FLENHIGH[7:0] | | | | | | | |
| 0x14 | INTENCLR | 7:0 | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |
| 0x15 | | 15:8 | | | | | | | DDISC | DCONN |
| 0x16 | Reserved | | | | | | | | | |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTENSET | 7:0 | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x19 | | 15:8 | | | | | | | DDISC | DCONN |
| 0x1A | Reserved | | | | | | | | | |
| 0x1B | Reserved | | | | | | | | | |
| 0x1C | INTFLAG | 7:0 | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |
| 0x1D | | 15:8 | | | | | | | DDISC | DCONN |
| 0x1E | Reserved | | | | | | | | | |
| 0x1F | Reserved | | | | | | | | | |
| 0x20 | PINTSMRY | 7:0 | PINT[7:0] | | | | | | | |
| 0x21 | | 15:8 | PINT[15:8] | | | | | | | |
| 0x22 | Reserved | | | | | | | | | |
| 0x23 | | | | | | | | | | |

**Table 32-6. Host Pipe Register n**

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1m0 | PCFGn | 7:0 | | | PTYPE[2:0] | | | BK | PTOKEN[1:0] | |
| 0x1m1 | Reserved | | | | | | | | | |
| 0x1m2 | Reserved | | | | | | | | | |
| 0x1m3 | BINTERVAL | 7:0 | BINTERVAL[7:0] | | | | | | | |
| 0x1m4 | PSTATUSCLRn | 7:0 | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| 0x1m5 | PSTATUSETn | 7:0 | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| 0x1m6 | PSTATUSn | 7:0 | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| 0x1m7 | PINTFLAGn | 7:0 | | | STALL | TXSTP | PERR | TRFAIL | TRCPT1 | TRCPT0 |
| 0x1m8 | PINTENCLRn | 7:0 | | | STALL | TXSTP | PERR | TRFAIL | TRCPT1 | TRCPT0 |
| 0x1m9 | PINTENSETn | 7:0 | | | STALL | TXSTP | PERR | TRFAIL | TRCPT1 | TRCPT0 |
| 0x1mA | Reserved | | | | | | | | | |
| 0x1mB | Reserved | | | | | | | | | |

**Table 32-7. Host Pipe n Descriptor Bank 0**

| Offset 0x n0 + index | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ADDR | 7:0 | ADD[7:0] | | | | | | | |
| 0x01 | | 15:8 | ADD[15:8] | | | | | | | |
| 0x02 | | 23:16 | ADD[23:16] | | | | | | | |
| 0x03 | | 31:24 | ADD[31:24] | | | | | | | |
| 0x04 | PCKSIZE | 7:0 | BYTE_COUNT[7:0] | | | | | | | |
| 0x05 | | 15:8 | MULTI_PACKET_SIZE[1:0] | | BYTE_COUNT[13:8] | | | | | |
| 0x06 | | 23:16 | MULTI_PACKET_SIZE[9:2] | | | | | | | |
| 0x07 | | 31:24 | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | |
| 0x08 | EXTREG | 7:0 | VARIABLE[3:0] | | | | SUBPID[3:0] | | | |
| 0x09 | | 15:8 | | VARIABLE[10:4] | | | | | | |
| 0x0A | STATUS_BK | 7:0 | | | | | | | ERRORFLOW | CRCERR |
| 0x0B | | 15:8 | | | | | | | | |
| 0x0C | CTRL_PIPE | 7:0 | | PDADDR[6:0] | | | | | | |
| 0x0D | | 15:8 | PEPMAX[3:0] | | | | PEPNUM[3:0] | | | |

| Offset 0x n0 + index | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x0E | STATUS_PIPE | 7:0 | | | ERCNT[2:0] | | | CRC16ER | TOUTER | PIDER | DAPIDER | DTGLER |
| 0x0F | | 15:8 | | | | | | | | | | |

**Table 32-8. Host Pipe n Descriptor Bank 1**

| Offset 0x n0 +0x10 +index | Name | Bit Pos. | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ADDR | 7:0 | ADD[7:0] | | | | | | | | |
| 0x01 | | 15:8 | ADD[15:8] | | | | | | | | |
| 0x02 | | 23:16 | ADD[23:16] | | | | | | | | |
| 0x03 | | 31:24 | ADD[31:24] | | | | | | | | |
| 0x04 | PCKSIZE | 7:0 | BYTE_COUNT[7:0] | | | | | | | | |
| 0x05 | | 15:8 | MULTI_PACKET_SIZE[1:0 | | BYTE_COUNT[13:8] | | | | | | |
| 0x06 | | 23:16 | MULTI_PACKET_SIZE[9:2] | | | | | | | | |
| 0x07 | | 31:24 | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | | |
| 0x08 | | 7:0 | | | | | | | | | |
| 0x09 | | 15:8 | | | | | | | | | |
| 0x0A | STATUS_BK | 7:0 | | | | | | | | ERRORFLOW | CRCERR |
| 0x0B | | 15:8 | | | | | | | | | |
| 0x0C | | 7:0 | | | | | | | | | |
| 0x0D | | 15:8 | | | | | | | | | |
| 0x0E | STATUS_PIPE | 7:0 | | | ERCNT[2:0] | | | CRC16ER | TOUTER | PIDER | DAPIDER | DTGLER |
| 0x0F | | 15:8 | | | | | | | | | | |

## 32.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Refer to the 32.5.8 Register Access Protection, *PAC - Peripheral Access Controller* and *GCLK Synchronization* for details.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 32.8.1 Communication Device Host Registers

#### 32.8.1.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronised

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MODE | | | | | RUNSTDBY | ENABLE | SWRST |
| Access | R/W | | | | | R/W | R/W | R/W |
| Reset | 0 | | | | | 0 | 0 | 0 |

**Bit 7 – MODE** Operating Mode
This bit defines the operating mode of the USB.

| Value | Description |
|---|---|
| 0 | USB Device mode |
| 1 | USB Host mode |

**Bit 2 – RUNSTDBY** Run in Standby Mode
This bit is Enable-Protected.

| Value | Description |
|---|---|
| 0 | USB clock is stopped in standby mode. |
| 1 | USB clock is running in standby mode |

**Bit 1 – ENABLE** Enable
Due to synchronization there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the Synchronization status enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

This bit is Write-Synchronized.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a '1' to this bit resets all registers in the USB, to their initial state, and the USB will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is Write-Synchronized.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

#### 32.8.1.2 Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x02
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ENABLE | SWRST |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ENABLE** Synchronization Enable status bit
This bit is cleared when the synchronization of ENABLE register between the clock domains is complete.

This bit is set when the synchronization of ENABLE register between clock domains is started.

**Bit 0 – SWRST** Synchronization Software Reset status bit
This bit is cleared when the synchronization of SWRST register between the clock domains is complete.

This bit is set when the synchronization of SWRST register between clock domains is started.

#### 32.8.1.3 QOS Control

**Name:** QOSCTRL
**Offset:** 0x03
**Reset:** 0x05
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DQOS[1:0] | | CQOS[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 1 | 0 | 1 |

**Bits 3:2 – DQOS[1:0]** Data Quality of Service
These bits define the memory priority access during the endpoint or pipe read/write data operation. Refer to *SRAM Quality of Service*.

**Bits 1:0 – CQOS[1:0]** Configuration Quality of Service
These bits define the memory priority access during the endpoint or pipe read/write configuration operation. Refer to *SRAM Quality of Service*.

**Related Links**
11.4.3 SRAM Quality of Service

#### 32.8.1.4 Finite State Machine Status

**Name:** FSMSTATUS
**Offset:** 0x0D
**Reset:** 0xXXXX
**Property:** Read only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | FSMSTATE[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 6:0 – FSMSTATE[6:0]** Fine State Machine Status
These bits indicate the state of the finite state machine of the USB controller.

| Value | Name | Description |
|---|---|---|
| 0x01 | OFF (L3) | Corresponds to the powered-off, disconnected, and disabled state. |
| 0x02 | ON (L0) | Corresponds to the Idle and Active states. |
| 0x04 | SUSPEND (L2) | |
| 0x08 | SLEEP (L1) | |
| 0x10 | DNRESUME | Down Stream Resume. |
| 0x20 | UPRESUME | Up Stream Resume. |
| 0x40 | RESET | USB lines Reset. |
| Others | | Reserved |

#### 32.8.1.5 Descriptor Address

**Name:** DESCADD
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** PAC Write-Protection

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DESCADD[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DESCADD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DESCADD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DESCADD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DESCADD[31:0]** Descriptor Address Value
These bits define the base address of the main USB descriptor in RAM. The two least significant bits must be written to zero.

#### 32.8.1.6 Pad Calibration

**Name:** PADCAL
**Offset:** 0x28
**Reset:** 0x0000
**Property:** PAC Write-Protection

The Pad Calibration values must be loaded from the NVM Software Calibration Area into the USB Pad Calibration register by software, before enabling the USB, to achieve the specified accuracy.
Refer to *NVM Software Calibration Area Mapping* for further details.

Refer to for further details.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TRIM[2:0] | | | | TRANSN[4:2] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRANSN[1:0] | | | TRANSP[4:0] | | | | |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bits 14:12 – TRIM[2:0]** Trim bits for DP/DM
These bits calibrate the matching of rise/fall of DP/DM.

**Bits 10:6 – TRANSN[4:0]** Trimmable Output Driver Impedance N
These bits calibrate the NMOS output impedance of DP/DM drivers.

**Bits 4:0 – TRANSP[4:0]** Trimmable Output Driver Impedance P
These bits calibrate the PMOS output impedance of DP/DM drivers.

**Related Links**
10.3.2 NVM Software Calibration Area Mapping

### 32.8.2 Device Registers - Common

#### 32.8.2.1 Control B

**Name:** CTRLB
**Offset:** 0x08
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | LPMHDSK[1:0] | | GNAK | |
| Access | | | | | R/W | R/W | R/W | |
| Reset | | | | | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NREPLY | SPDCONF[1:0] | | UPRSM | DETACH |
| Access | | | | R | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 11:10 – LPMHDSK[1:0]** Link Power Management Handshake
These bits select the Link Power Management Handshake configuration.

| Value | Description |
|---|---|
| 0x0 | No handshake. LPM is not supported. |
| 0x1 | ACK |
| 0x2 | NYET |
| 0x3 | Reserved |

**Bit 9 – GNAK** Global NAK
This bit configures the operating mode of the NAK.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | The handshake packet reports the status of the USB transaction |
| 1 | A NAK handshake is answered for each USB transaction regardless of the current endpoint memory bank status |

**Bit 4 – NREPLY** No reply excepted SETUP Token
This bit is cleared by hardware when receiving a SETUP packet.

This bit has no effect for any other endpoint but endpoint 0.

| Value | Description |
|---|---|
| 0 | Disable the “NO_REPLY” feature: Any transaction to endpoint 0 will be handled according to the USB2.0 standard. |
| 1 | Enable the “NO_REPLY” feature: Any transaction to endpoint 0 will be ignored except SETUP. |

**Bits 3:2 – SPDCONF[1:0]** Speed Configuration
These bits select the speed configuration.

| Value | Description |
|---|---|
| 0x0 | FS: Full-speed |
| 0x1 | LS: Low-speed |
| 0x2 | Reserved |
| 0x3 | Reserved |

**Bit 1 – UPRSM** Upstream Resume
This bit is cleared when the USB receives a USB reset or once the upstream resume has been sent.

| Value | Description |
|---|---|
| 0 | Writing a zero to this bit has no effect. |
| 1 | Writing a one to this bit will generate an upstream resume to the host for a remote wakeup. |

**Bit 0 – DETACH** Detach

| Value | Description |
|---|---|
| 0 | The device is attached to the USB bus so that communications may occur. |
| 1 | It is the default value at reset. The internal device pull-ups are disabled, removing the device from the USB bus. |

#### 32.8.2.2 Device Address

**Name:** DADD
**Offset:** 0x0A
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDEN | DADD[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ADDEN** Device Address Enable
This bit is cleared when a USB reset is received.

| Value | Description |
|---|---|
| 0 | Writing a zero will deactivate the DADD field (USB device address) and return the device to default address 0. |
| 1 | Writing a one will activate the DADD field (USB device address). |

**Bits 6:0 – DADD[6:0]** Device Address
These bits define the device address. The DADD register is reset when a USB reset is received.

#### 32.8.2.3 Status

**Name:** STATUS
**Offset:** 0x0C
**Reset:** 0x40
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LINESTATE[1:0] | | | | SPEED[1:0] | | | |
| Access | R | R | | | R/W | R/W | | |
| Reset | 0 | 1 | | | 0 | 1 | | |

**Bits 7:6 – LINESTATE[1:0]** USB Line State Status
These bits define the current line state DP/DM.

| LINESTATE[1:0] | USB Line Status |
|---|---|
| 0x0 | SE0/RESET |
| 0x1 | FS-J or LS-K State |
| 0x2 | FS-K or LS-J State |

**Bits 3:2 – SPEED[1:0]** Speed Status
These bits define the current speed used of the device

.

| SPEED[1:0] | SPEED STATUS |
|---|---|
| 0x0 | Low-speed mode |
| 0x1 | Full-speed mode |
| 0x2 | Reserved |
| 0x3 | Reserved |

#### 32.8.2.4 Device Frame Number

**Name:** FNUM
**Offset:** 0x10
**Reset:** 0x0000
**Property:** Read only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FNCERR | | FNUM[10:5] | | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FNUM[4:0] | | | | | MFNUM[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – FNCERR** Frame Number CRC Error
This bit is cleared upon receiving a USB reset.

This bit is set when a corrupted frame number (or micro-frame number) is received.

This bit and the SOF (or MSOF) interrupt bit are updated at the same time.

**Bits 13:3 – FNUM[10:0]** Frame Number
These bits are cleared upon receiving a USB reset.

These bits are updated with the frame number information as provided from the last SOF packet even if a corrupted SOF is received.

**Bits 2:0 – MFNUM[2:0]** Micro Frame Number
These bits are cleared upon receiving a USB reset or at the beginning of each Start-of-Frame (SOF interrupt).

These bits are updated with the micro-frame number information as provided from the last MSOF packet even if a corrupted MSOF is received.

#### 32.8.2.5 Device Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x0000
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set (INTENSET) register.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LPMSUSP | LPMNYET |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bit 9 – LPMSUSP** Link Power Management Suspend Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Link Power Management Suspend Interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Link Power Management Suspend interrupt is disabled. |
| 1 | The Link Power Management Suspend interrupt is enabled and an interrupt request will be generated when the Link Power Management Suspend interrupt Flag is set. |

**Bit 8 – LPMNYET** Link Power Management Not Yet Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Link Power Management Not Yet interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Link Power Management Not Yet interrupt is disabled. |
| 1 | The Link Power Management Not Yet interrupt is enabled and an interrupt request will be generated when the Link Power Management Not Yet interrupt Flag is set. |

**Bit 7 – RAMACER** RAM Access Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the RAM Access interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The RAM Access interrupt is disabled. |
| 1 | The RAM Access interrupt is enabled and an interrupt request will be generated when the RAM Access interrupt Flag is set. |

**Bit 6 – UPRSM** Upstream Resume Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Upstream Resume interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Upstream Resume interrupt is disabled. |
| 1 | The Upstream Resume interrupt is enabled and an interrupt request will be generated when the Upstream Resume interrupt Flag is set. |

**Bit 5 – EORSM** End Of Resume Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the End Of Resume interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The End Of Resume interrupt is disabled. |
| 1 | The End Of Resume interrupt is enabled and an interrupt request will be generated when the End Of Resume interrupt Flag is set. |

**Bit 4 – WAKEUP** Wake-Up Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Wake Up interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Wake Up interrupt is disabled. |
| 1 | The Wake Up interrupt is enabled and an interrupt request will be generated when the Wake Up interrupt Flag is set. |

**Bit 3 – EORST** End of Reset Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the End of Reset interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The End of Reset interrupt is disabled. |
| 1 | The End of Reset interrupt is enabled and an interrupt request will be generated when the End of Reset interrupt Flag is set. |

**Bit 2 – SOF** Start-of-Frame Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Start-of-Frame interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Start-of-Frame interrupt is disabled. |
| 1 | The Start-of-Frame interrupt is enabled and an interrupt request will be generated when the Start-of-Frame interrupt Flag is set. |

**Bit 0 – SUSPEND** Suspend Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Suspend Interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Suspend interrupt is disabled. |
| 1 | The Suspend interrupt is enabled and an interrupt request will be generated when the Suspend interrupt Flag is set. |

#### 32.8.2.6 Device Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x18
**Reset:** 0x0000
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LPMSUSP | LPMNYET |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bit 9 – LPMSUSP** Link Power Management Suspend Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Link Power Management Suspend Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Link Power Management Suspend interrupt is disabled. |
| 1 | The Link Power Management Suspend interrupt is enabled. |

**Bit 8 – LPMNYET** Link Power Management Not Yet Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Link Power Management Not Yet interrupt bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Link Power Management Not Yet interrupt is disabled. |
| 1 | The Link Power Management Not Yet interrupt is enabled. |

**Bit 7 – RAMACER** RAM Access Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the RAM Access Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The RAM Access interrupt is disabled. |
| 1 | The RAM Access interrupt is enabled. |

**Bit 6 – UPRSM** Upstream Resume Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Upstream Resume Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Upstream Resume interrupt is disabled. |
| 1 | The Upstream Resume interrupt is enabled. |

**Bit 5 – EORSM** End Of Resume Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the End Of Resume interrupt Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The End Of Resume interrupt is disabled. |
| 1 | The End Of Resume interrupt is enabled. |

**Bit 4 – WAKEUP** Wake-Up Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Wake Up interrupt Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Wake Up interrupt is disabled. |
| 1 | The Wake Up interrupt is enabled. |

**Bit 3 – EORST** End of Reset Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the End of Reset interrupt Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The End of Reset interrupt is disabled. |
| 1 | The End of Reset interrupt is enabled. |

**Bit 2 – SOF** Start-of-Frame Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Start-of-Frame interrupt Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Start-of-Frame interrupt is disabled. |
| 1 | The Start-of-Frame interrupt is enabled. |

**Bit 0 – SUSPEND** Suspend Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Suspend interrupt Enable bit and enable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Suspend interrupt is disabled. |
| 1 | The Suspend interrupt is enabled. |

#### 32.8.2.7 Device Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x01C
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LPMSUSP | LPMNYET |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | EORSM | WAKEUP | EORST | SOF | | SUSPEND |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bit 9 – LPMSUSP** Link Power Management Suspend Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB module acknowledge a Link Power Management Transaction (ACK handshake) and has entered the Suspended state and will generate an interrupt if INTENCLR/SET.LPMSUSP is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the LPMSUSP Interrupt Flag.

**Bit 8 – LPMNYET** Link Power Management Not Yet Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB module acknowledges a Link Power Management Transaction (handshake is NYET) and will generate an interrupt if INTENCLR/SET.LPMNYET is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the LPMNYET Interrupt Flag.

**Bit 7 – RAMACER** RAM Access Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a RAM access underflow error occurs during IN data stage. This bit will generate an interrupt if INTENCLR/SET.RAMACER is one.

Writing a zero to this bit has no effect.

**Bit 6 – UPRSM** Upstream Resume Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB sends a resume signal called “Upstream Resume” and will generate an interrupt if INTENCLR/SET.UPRSM is one.

Writing a zero to this bit has no effect.

**Bit 5 – EORSM** End Of Resume Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB detects a valid “End of Resume” signal initiated by the host and will generate an interrupt if INTENCLR/SET.EORSM is one.

Writing a zero to this bit has no effect.

**Bit 4 – WAKEUP** Wake Up Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB is reactivated by a filtered non-idle signal from the lines and will generate an interrupt if INTENCLR/SET.WAKEUP is one.

Writing a zero to this bit has no effect.

**Bit 3 – EORST** End of Reset Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “End of Reset” has been detected and will generate an interrupt if INTENCLR/SET.EORST is one.

Writing a zero to this bit has no effect.

**Bit 2 – SOF** Start-of-Frame Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “Start-of-Frame” has been detected (every 1 ms) and will generate an interrupt if INTENCLR/SET.SOF is one.

The FNUM is updated. In High Speed mode, the MFNUM register is cleared.

Writing a zero to this bit has no effect.

**Bit 0 – SUSPEND** Suspend Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “Suspend” idle state has been detected for 3 frame periods (J state for 3 ms) and will generate an interrupt if INTENCLR/SET.SUSPEND is one.

Writing a zero to this bit has no effect.

#### 32.8.2.8 Endpoint Interrupt Summary

**Name:** EPINTSMRY
**Offset:** 0x20
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EPINT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EPINT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – EPINT[15:0]** EndPoint Interrupt
The flag EPINT[n] is set when an interrupt is triggered by the EndPoint n. See 32.8.3.5 EPINTFLAGn register in the device EndPoint section.

This bit will be cleared when no interrupts are pending for EndPoint n.

### 32.8.3 Device Registers - Endpoint

#### 32.8.3.1 Device Endpoint Configuration register n

**Name:** EPCFGn
**Offset:** 0x100 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | EPTYPE1[2:0] | | | | EPTYPE0[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 6:4 – EPTYPE1[2:0]** Endpoint Type for IN direction
These bits contains the endpoint type for IN direction.

Upon receiving a USB reset EPCFGn.EPTYPE1 is cleared except for endpoint 0 which is unchanged.

| Value | Description |
|---|---|
| 0x0 | Bank1 is disabled. |
| 0x1 | Bank1 is enabled and configured as Control IN. |
| 0x2 | Bank1 is enabled and configured as Isochronous IN. |
| 0x3 | Bank1 is enabled and configured as Bulk IN. |
| 0x4 | Bank1 is enabled and configured as Interrupt IN. |
| 0x5 | Bank1 is enabled and configured as Dual-Bank OUT<br>(Endpoint type is the same as the one defined in EPTYPE0) |
| 0x6-0x7 | Reserved |

**Bits 2:0 – EPTYPE0[2:0]** Endpoint Type for OUT direction
These bits contains the endpoint type for OUT direction.

Upon receiving a USB reset EPCFGn.EPTYPE0 is cleared except for endpoint 0 which is unchanged.

| Value | Description |
|---|---|
| 0x0 | Bank0 is disabled. |
| 0x1 | Bank0 is enabled and configured as Control SETUP / Control OUT. |
| 0x2 | Bank0 is enabled and configured as Isochronous OUT. |
| 0x3 | Bank0 is enabled and configured as Bulk OUT. |
| 0x4 | Bank0 is enabled and configured as Interrupt OUT. |
| 0x5 | Bank0 is enabled and configured as Dual Bank IN<br>(Endpoint type is the same as the one defined in EPTYPE1) |
| 0x6-0x7 | Reserved |

#### 32.8.3.2 EndPoint Status Clear n

**Name:** EPSTATUSCLRn
**Offset:** 0x104 + (n * 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | STALLRQ1 | STALLRQ0 | | CURBK | DTGLIN | DTGLOUT |
| Access | W | W | W | W | | W | W | W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – BK1RDY** Bank 1 Ready Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.BK1RDY bit.

**Bit 6 – BK0RDY** Bank 0 Ready Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.BK0RDY bit.

**Bit 5 – STALLRQ1** STALL bank 1 Request Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.STALLRQ1 bit.

**Bit 4 – STALLRQ0** STALL bank 0 Request Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.STALLRQ0 bit.

**Bit 2 – CURBK** Current Bank Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.CURBK bit.

**Bit 1 – DTGLIN** Data Toggle IN Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear EPSTATUS.DTGLIN bit.

**Bit 0 – DTGLOUT** Data Toggle OUT Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the EPSTATUS.DTGLOUT bit.

#### 32.8.3.3 EndPoint Status Set n

**Name:** EPSTATUSSETn
**Offset:** 0x105 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | STALLRQ1 | STALLRQ0 | | CURBK | DTGLIN | DTGLOUT |
| Access | W | W | W | W | | W | W | W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – BK1RDY** Bank 1 Ready Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.BK1RDY bit.

**Bit 6 – BK0RDY** Bank 0 Ready Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.BK0RDY bit.

**Bit 5 – STALLRQ1** STALL Request bank 1 Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.STALLRQ1 bit.

**Bit 4 – STALLRQ0** STALL Request bank 0 Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.STALLRQ0 bit.

**Bit 2 – CURBK** Current Bank Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.CURBK bit.

**Bit 1 – DTGLIN** Data Toggle IN Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set EPSTATUS.DTGLIN bit.

**Bit 0 – DTGLOUT** Data Toggle OUT Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set the EPSTATUS.DTGLOUT bit.

#### 32.8.3.4 EndPoint Status n

**Name:** EPSTATUSn
**Offset:** 0x106 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | | STALLRQ | | CURBK | DTGLIN | DTGLOUT |
| Access | R | R | | R | | R | R | R |
| Reset | 0 | 0 | | 2 | | 0 | 0 | 0 |

**Bit 7 – BK1RDY** Bank 1 is ready
For Control/OUT direction Endpoints, the bank is empty.

Writing a one to the bit EPSTATUSCLR.BK1RDY will clear this bit.

Writing a one to the bit EPSTATUSSET.BK1RDY will set this bit.

| Value | Description |
|---|---|
| 0 | The bank number 1 is not ready : For IN direction Endpoints, the bank is not yet filled in. |
| 1 | The bank number 1 is ready: For IN direction Endpoints, the bank is filled in. For Control/OUT direction Endpoints, the bank is full. |

**Bit 6 – BK0RDY** Bank 0 is ready
Writing a one to the bit EPSTATUSCLR.BK0RDY will clear this bit.

Writing a one to the bit EPSTATUSSET.BK0RDY will set this bit.

| Value | Description |
|---|---|
| 0 | The bank number 0 is not ready : For IN direction Endpoints, the bank is not yet filled in. For Control/OUT direction Endpoints, the bank is empty. |
| 1 | The bank number 0 is ready: For IN direction Endpoints, the bank is filled in. For Control/OUT direction Endpoints, the bank is full. |

**Bit 4 – STALLRQ** STALL bank x request
Writing a zero to the bit EPSTATUSCLR.STALLRQ will clear this bit.

Writing a one to the bit EPSTATUSSET.STALLRQ will set this bit.

This bit is cleared by hardware when receiving a SETUP packet.

| Value | Description |
|---|---|
| 0 | Disable STALLRQx feature. |
| 1 | Enable STALLRQx feature: a STALL handshake will be sent to the host in regards to bank x. |

**Bit 2 – CURBK** Current Bank
Writing a zero to the bit EPSTATUSCLR.CURBK will clear this bit.

Writing a one to the bit EPSTATUSSET.CURBK will set this bit.

| Value | Description |
|---|---|
| 0 | The bank0 is the bank that will be used in the next single/multi USB packet. |
| 1 | The bank1 is the bank that will be used in the next single/multi USB packet. |

**Bit 1 – DTGLIN** Data Toggle IN Sequence
Writing a zero to the bit EPSTATUSCLR.DTGLINCLR will clear this bit.

Writing a one to the bit EPSTATUSSET.DTGLINSET will set this bit.

| Value | Description |
|---|---|
| 0 | The PID of the next expected IN transaction will be zero: data 0. |
| 1 | The PID of the next expected IN transaction will be one: data 1. |

**Bit 0 – DTGLOUT** Data Toggle OUT Sequence
Writing a zero to the bit EPSTATUSCLR.DTGLOUTCLR will clear this bit.

Writing a one to the bit EPSTATUSSET.DTGLOUTSET will set this bit.

| Value | Description |
|---|---|
| 0 | The PID of the next expected OUT transaction will be zero: data 0. |
| 1 | The PID of the next expected OUR transaction will be one: data 1. |

#### 32.8.3.5 Device EndPoint Interrupt Flag n

**Name:** EPINTFLAGn
**Offset:** 0x107 + (n x 0x20)
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | RXSTP | | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | | R/W | | R/W |
| Reset | | | 0 | 0 | | 0 | | 0 |

**Bit 5 – STALL** Transmit Stall x Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Transmit Stall occurs and will generate an interrupt if EPINTENCLR/SET.STALL is one.

EPINTFLAG.STALL is set for a single bank OUT endpoint or double bank IN/OUT endpoint when current bank is "0".

Writing a zero to this bit has no effect.

Writing a one to this bit clears the STALL Interrupt Flag.

**Bit 4 – RXSTP** Received Setup Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Received Setup occurs and will generate an interrupt if EPINTENCLR/SET.RXSTP is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the RXSTP Interrupt Flag.

**Bit 2 – TRFAIL** Transfer Fail x Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a transfer fail occurs and will generate an interrupt if EPINTENCLR/SET.TRFAIL is one.

EPINTFLAG.TRFAIL is set for a single bank OUT endpoint or double bank IN/OUT endpoint when current bank is "0".

Writing a zero to this bit has no effect.

Writing a one to this bit clears the TRFAIL Interrupt Flag.

**Bit 0 – TRCPT** Transfer Complete x interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Transfer complete occurs and will generate an interrupt if EPINTENCLR/SET.TRCPT is one. EPINTFLAG.TRCPT is set for a single bank OUT endpoint or double bank IN/OUT endpoint when current bank is "0".

Writing a zero to this bit has no effect.

Writing a one to this bit clears the TRCPT0 Interrupt Flag.

#### 32.8.3.6 Device EndPoint Interrupt Enable n

**Name:** EPINTENCLRn
**Offset:** 0x108 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Endpoint Interrupt Enable Set (EPINTENSET) register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | RXSTP | | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | | R/W | | R/W |
| Reset | | | 0 | 0 | | 0 | | 0 |

**Bit 5 – STALL** Transmit STALL x Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transmit Stall x Interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transmit Stall x interrupt is disabled. |
| 1 | The Transmit Stall x interrupt is enabled and an interrupt request will be generated when the Transmit Stall x Interrupt Flag is set. |

**Bit 4 – RXSTP** Received Setup Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Received Setup Interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Received Setup interrupt is disabled. |
| 1 | The Received Setup interrupt is enabled and an interrupt request will be generated when the Received Setup Interrupt Flag is set. |

**Bit 2 – TRFAIL** Transfer Fail x Interrupt Enable
The user should look into the descriptor table status located in ram to be informed about the error condition : ERRORFLOW, CRC.

Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transfer Fail x Interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transfer Fail bank x interrupt is disabled. |
| 1 | The Transfer Fail bank x interrupt is enabled and an interrupt request will be generated when the Transfer Fail x Interrupt Flag is set. |

**Bit 0 – TRCPT** Transfer Complete x interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transfer Complete x interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transfer Complete bank x interrupt is disabled. |
| 1 | The Transfer Complete bank x interrupt is enabled and an interrupt request will be generated when the Transfer Complete x Interrupt Flag is set. |

#### 32.8.3.7 Device Interrupt EndPoint Set n

**Name:** EPINTENSETn
**Offset:** 0x109 + (n x 0x20)
**Reset:** 0x0000
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Endpoint Interrupt Enable Set (EPINTENCLR) register. This register is cleared by USB reset or when EPEN[n] is zero.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | RXSTP | | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | | R/W | | R/W |
| Reset | | | 0 | 0 | | 0 | | 0 |

**Bit 5 – STALL** Transmit Stall x Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Transmit bank x Stall interrupt.

| Value | Description |
|---|---|
| 0 | The Transmit Stall x interrupt is disabled. |
| 1 | The Transmit Stall x interrupt is enabled. |

**Bit 4 – RXSTP** Received Setup Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Received Setup interrupt.

| Value | Description |
|---|---|
| 0 | The Received Setup interrupt is disabled. |
| 1 | The Received Setup interrupt is enabled. |

**Bit 2 – TRFAIL** Transfer Fail bank x Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Transfer Fail interrupt.

| Value | Description |
|---|---|
| 0 | The Transfer Fail interrupt is disabled. |
| 1 | The Transfer Fail interrupt is enabled. |

**Bit 0 – TRCPT** Transfer Complete bank x interrupt Enable
Writing a zero to this bit has no effect.

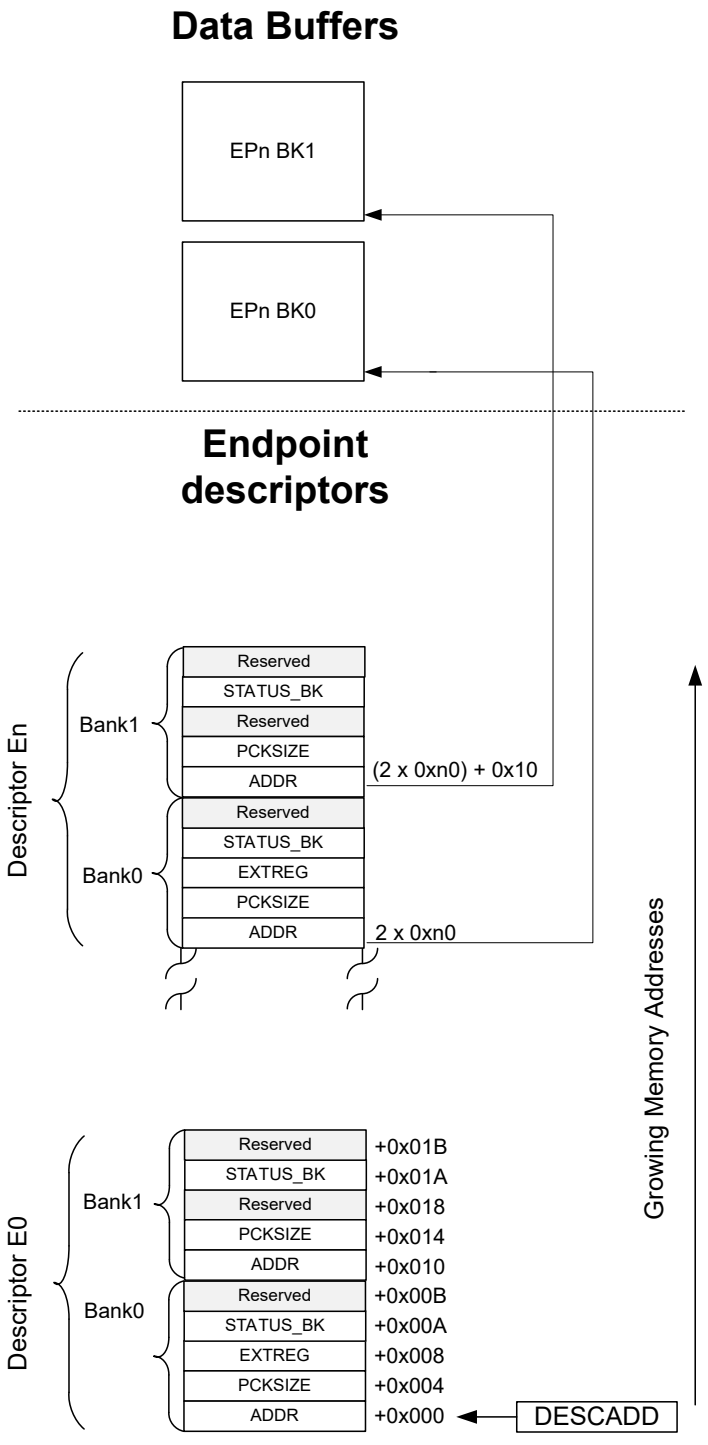
Writing a one to this bit will enable the Transfer Complete x interrupt.

0.2.4 Device Registers - Endpoint RAM

| Value | Description |
|---|---|
| 0 | The Transfer Complete bank x interrupt is disabled. |
| 1 | The Transfer Complete bank x interrupt is enabled. |

### 32.8.4 Device Registers - Endpoint RAM

#### 32.8.4.1 Endpoint Descriptor Structure



#### 32.8.4.2 Address of Data Buffer

**Name:** ADDR
**Offset:** 0x00 & 0x10
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 31:0 – ADDR[31:0]** Data Pointer Address Value
These bits define the data pointer address as an absolute word address in RAM. The two least significant bits must be zero to ensure the start address is 32-bit aligned.

### 32.8.4.3 Packet Size

**Name:** PCKSIZE
**Offset:** 0x04 & 0x14
**Reset:** 0xxxxxxxxx
**Property:** NA

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | 0 | 0 | x | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MULTI_PACKET_SIZE[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MULTI_PACKET_SIZE[1:0] | | BYTE_COUNT[13:8] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | x | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BYTE_COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

**Bit 31 – AUTO_ZLP** Automatic Zero Length Packet
This bit defines the automatic Zero Length Packet mode of the endpoint.

When enabled, the USB module will manage the ZLP handshake by hardware. This bit is for IN endpoints only. When disabled the handshake should be managed by firmware.

| Value | Description |
|---|---|
| 0 | Automatic Zero Length Packet is disabled. |
| 1 | Automatic Zero Length Packet is enabled. |

**Bits 30:28 – SIZE[2:0]** Endpoint size
These bits contains the maximum packet size of the endpoint.

| Value | Description |
|---|---|
| 0x0 | 8 Byte |
| 0x1 | 16 Byte |
| 0x2 | 32 Byte |
| 0x3 | 64 Byte |
| 0x4 | 128 Byte(1) |

| Value | Description |
|---|---|
| 0x5 | 256 Byte(1) |
| 0x6 | 512 Byte(1) |
| 0x7 | 1023 Byte(1) |

(1) for Isochronous endpoints only.

**Bits 27:14 – MULTI_PACKET_SIZE[13:0]** Multiple Packet Size
These bits define the 14-bit value that is used for multi-packet transfers.

For IN endpoints, MULTI_PACKET_SIZE holds the total number of bytes sent. MULTI_PACKET_SIZE should be written to zero when setting up a new transfer.

For OUT endpoints, MULTI_PACKET_SIZE holds the total data size for the complete transfer. This value must be a multiple of the maximum packet size.

**Bits 13:0 – BYTE_COUNT[13:0]** Byte Count
These bits define the 14-bit value that is used for the byte count.

For IN endpoints, BYTE_COUNT holds the number of bytes to be sent in the next IN transaction.

For OUT endpoint or SETUP endpoints, BYTE_COUNT holds the number of bytes received upon the last OUT or SETUP transaction.

#### 32.8.4.4 Extended Register

**Name:** EXTREG
**Offset:** 0x08
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | VARIABLE[10:4] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VARIABLE[3:0] | | | | SUBPID[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | x | 0 | 0 | 0 | x |

**Bits 14:4 – VARIABLE[10:0]** Variable field send with extended token
These bits define the VARIABLE field of a received extended token. These bits are updated when the USB has answered by an handshake token ACK to a LPM transaction. See Section 2.1.1 Protocol Extension Token in the reference document “ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum”.

To support the USB2.0 Link Power Management addition the VARIABLE field should be read as described below.

| VARIABLES | Description |
|---|---|
| VARIABLE[3:0] | bLinkState (1) |
| VARIABLE[7:4] | BESL (2) |
| VARIABLE[8] | bRemoteWake (1) |
| VARIABLE[10:9] | Reserved |

1. For a definition of LPM Token bRemoteWake and bLinkState fields, refer to "Table 2-3 in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum".
2. For a definition of LPM Token BESL field, refer to "Table 2-3 in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum" and "Table X-X1 in Errata for ECN USB 2.0 Link Power Management.

**Bits 3:0 – SUBPID[3:0]** SUBPID field send with extended token
These bits define the SUBPID field of a received extended token. These bits are updated when the USB has answered by an handshake token ACK to a LPM transaction. See Section 2.1.1 Protocol Extension Token in the reference document “ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum”.

#### 32.8.4.5 Device Status Bank

**Name:** STATUS_BK
**Offset:** 0x0A & 0x1A
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ERRORFLOW | CRCERR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | x | x |

**Bit 1 – ERRORFLOW** Error Flow Status
This bit defines the Error Flow Status.

This bit is set when a Error Flow has been detected during transfer from/towards this bank.

For OUT transfer, a NAK handshake has been sent.

For Isochronous OUT transfer, an overrun condition has occurred.

For IN transfer, this bit is not valid. EPSTATUS.TRFAIL0 and EPSTATUS.TRFAIL1 should reflect the flow errors.

| Value | Description |
|---|---|
| 0 | No Error Flow detected. |
| 1 | A Error Flow has been detected. |

**Bit 0 – CRCERR** CRC Error
This bit defines the CRC Error Status.

This bit is set when a CRC error has been detected in an isochronous OUT endpoint bank.

0.2.5 Host Registers - Common

| Value | Description |
|---|---|
| 0 | No CRC Error. |
| 1 | CRC Error detected. |

### 32.8.5 Host Registers - Common

### 32.8.5.1 Control B

**Name:** CTRLB
**Offset:** 0x08
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | L1RESUME | VBUSOK | BUSRESET | SOFE |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SPDCONF[1:0] | | RESUME | |
| Access | | | | | R/W | R/W | R/W | |
| Reset | | | | | 0 | 0 | 0 | |

**Bit 11 – L1RESUME** Send USB L1 Resume
Writing 0 to this bit has no effect.
1: Generates a USB L1 Resume on the USB bus. This bit should only be set when the Start-of-Frame generation is enabled (SOFE bit set). The duration of the USB L1 Resume is defined by the EXTREG.VARIABLE[7:4] bits field also known as BESL (See LPM ECN).See also 32.8.7.4 EXTREG Register.

This bit is cleared when the USB L1 Resume has been sent or when a USB reset is requested.

**Bit 10 – VBUSOK** VBUS is OK
This notifies the USB HOST that USB operations can be started. When this bit is zero and even if the USB HOST is configured and enabled, HOST operation is halted. Setting this bit will allow HOST operation when the USB is configured and enabled.

| Value | Description |
|---|---|
| 0 | The USB module is notified that the VBUS on the USB line is not powered. |
| 1 | The USB module is notified that the VBUS on the USB line is powered. |

**Bit 9 – BUSRESET** Send USB Reset

| Value | Description |
|---|---|
| 0 | Reset generation is disabled. It is written to zero when the USB reset is completed or when a device disconnection is detected. Writing zero has no effect. |
| 1 | Generates a USB Reset on the USB bus. |

**Bit 8 – SOFE** Start-of-Frame Generation Enable

| Value | Description |
|---|---|
| 0 | The SOF generation is disabled and the USB bus is in suspend state. |
| 1 | Generates SOF on the USB bus in full speed and keep it alive in low speed mode. This bit is automatically set at the end of a USB reset (INTFLAG.RST) or at the end of a downstream resume (INTFLAG.DNRSM) or at the end of L1 resume. |

**Bits 3:2 – SPDCONF[1:0]** Speed Configuration for Host
These bits select the host speed configuration as shown below

| Value | Description |
|---|---|
| 0x0 | Low and Full Speed capable |
| 0x1 | Reserved |
| 0x2 | Reserved |
| 0x3 | Reserved |

**Bit 1 – RESUME** Send USB Resume
Writing 0 to this bit has no effect.
1: Generates a USB Resume on the USB bus.

This bit is cleared when the USB Resume has been sent or when a USB reset is requested.

#### 32.8.5.2 Host Start-of-Frame Control

**Name:** HSOFC
**Offset:** 0x0A
**Reset:** 0x00
**Property:** PAC Write-Protection

During a very short period just before transmitting a Start-of-Frame, this register is locked. Thus, after writing, it is recommended to check the register value, and write this register again if necessary. This register is cleared upon a USB reset.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLENCE | | | | FLENC[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – FLENCE** Frame Length Control Enable
When this bit is '1', the time between Start-of-Frames can be tuned by up to +/-0.06% using FLENC[3:0].
**Note:** In Low Speed mode, FLENCE must be '0'.

| Value | Description |
|---|---|
| 0 | Start-of-Frame is generated every 1ms. |
| 1 | Start-of-Frame generation depends on the signed value of FLENC[3:0].<br>USB Start-of-Frame period equals 1ms + (FLENC[3:0]/12000)ms |

**Bits 3:0 – FLENC[3:0]** Frame Length Control
These bits define the signed value of the 4-bit FLENC that is added to the Internal Frame Length when FLENCE is '1'. The internal Frame length is the top value of the frame counter when FLENCE is zero.

### 32.8.5.3 Status

**Name:** STATUS
**Offset:** 0x0C
**Reset:** 0x00
**Property:** Read only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LINESTATE[1:0] | | | | SPEED[1:0] | | | |
| Access | R | R | | | R/W | R/W | | |
| Reset | 0 | 0 | | | 0 | 0 | | |

**Bits 7:6 – LINESTATE[1:0]** USB Line State Status
These bits define the current line state DP/DM.

| LINESTATE[1:0] | USB Line Status |
|---|---|
| 0x0 | SE0/RESET |
| 0x1 | FS-J or LS-K State |
| 0x2 | FS-K or LS-J State |

**Bits 3:2 – SPEED[1:0]** Speed Status
These bits define the current speed used by the host.

| SPEED[1:0] | Speed Status |
|---|---|
| 0x0 | Full-speed mode |
| 0x1 | Low-speed mode |
| 0x2 | Reserved |
| 0x3 | Reserved |

#### 32.8.5.4 Host Frame Number

**Name:** FNUM
**Offset:** 0x10
**Reset:** 0x0000
**Property:** PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FNUM[10:5] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FNUM[4:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | | | |

**Bits 13:3 – FNUM[10:0]** Frame Number
These bits contains the current SOF number.

These bits can be written by software to initialize a new frame number value. In this case, at the next SOF, the FNUM field takes its new value.

As the FNUM register lies across two consecutive byte addresses, writing byte-wise (8-bits) to the FNUM register may produce incorrect frame number generation. It is recommended to write FNUM register word-wise (32-bits) or half-word-wise (16-bits).

#### 32.8.5.5 Host Frame Length

**Name:** FLENHIGH
**Offset:** 0x12
**Reset:** 0x00
**Property:** Read-Only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLENHIGH[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – FLENHIGH[7:0]** Frame Length
These bits contains the 8 high-order bits of the internal frame counter.

**Table 32-9. Counter Description vs. Speed**

| Host Register STATUS.SPEED | Description |
|---|---|
| Full Speed | With a USB clock running at 12MHz, counter length is 12000 to ensure a SOF generation every 1 ms. |

#### 32.8.5.6 Host Interrupt Enable Register Clear

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x0000
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set (INTENSET) register.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DDISC | DCONN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 9 – DDISC** Device Disconnection Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Device Disconnection interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Device Disconnection interrupt is disabled. |
| 1 | The Device Disconnection interrupt is enabled and an interrupt request will be generated when the Device Disconnection interrupt Flag is set. |

**Bit 8 – DCONN** Device Connection Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Device Connection interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Device Connection interrupt is disabled. |
| 1 | The Device Connection interrupt is enabled and an interrupt request will be generated when the Device Connection interrupt Flag is set. |

**Bit 7 – RAMACER** RAM Access Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the RAM Access interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The RAM Access interrupt is disabled. |
| 1 | The RAM Access interrupt is enabled and an interrupt request will be generated when the RAM Access interrupt Flag is set. |

**Bit 6 – UPRSM** Upstream Resume from Device Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Upstream Resume interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Upstream Resume interrupt is disabled. |
| 1 | The Upstream Resume interrupt is enabled and an interrupt request will be generated when the Upstream Resume interrupt Flag is set. |

**Bit 5 – DNRSM** Down Resume Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Down Resume interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Down Resume interrupt is disabled. |
| 1 | The Down Resume interrupt is enabled and an interrupt request will be generated when the Down Resume interrupt Flag is set. |

**Bit 4 – WAKEUP** Wake Up Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Wake Up interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Wake Up interrupt is disabled. |
| 1 | The Wake Up interrupt is enabled and an interrupt request will be generated when the Wake Up interrupt Flag is set. |

**Bit 3 – RST** BUS Reset Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Bus Reset interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Bus Reset interrupt is disabled. |
| 1 | The Bus Reset interrupt is enabled and an interrupt request will be generated when the Bus Reset interrupt Flag is set. |

**Bit 2 – HSOF** Host Start-of-Frame Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Host Start-of-Frame interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Host Start-of-Frame interrupt is disabled. |
| 1 | The Host Start-of-Frame interrupt is enabled and an interrupt request will be generated when the Host Start-of-Frame interrupt Flag is set. |

#### 32.8.5.7 Host Interrupt Enable Register Set

**Name:** INTENSET
**Offset:** 0x18
**Reset:** 0x0000
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DDISC | DCONN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 9 – DDISC** Device Disconnection Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Device Disconnection interrupt bit and enable the DDSIC interrupt.

| Value | Description |
|---|---|
| 0 | The Device Disconnection interrupt is disabled. |
| 1 | The Device Disconnection interrupt is enabled. |

**Bit 8 – DCONN** Device Connection Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Device Connection interrupt bit and enable the DCONN interrupt.

| Value | Description |
|---|---|
| 0 | The Device Connection interrupt is disabled. |
| 1 | The Device Connection interrupt is enabled. |

**Bit 7 – RAMACER** RAM Access Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the RAM Access interrupt bit and enable the RAMACER interrupt.

| Value | Description |
|---|---|
| 0 | The RAM Access interrupt is disabled. |
| 1 | The RAM Access interrupt is enabled. |

**Bit 6 – UPRSM** Upstream Resume from the device Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Upstream Resume interrupt bit and enable the UPRSM interrupt.

| Value | Description |
|---|---|
| 0 | The Upstream Resume interrupt is disabled. |
| 1 | The Upstream Resume interrupt is enabled. |

**Bit 5 – DNRSM** Down Resume Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Down Resume interrupt Enable bit and enable the DNRSM interrupt.

| Value | Description |
|---|---|
| 0 | The Down Resume interrupt is disabled. |
| 1 | The Down Resume interrupt is enabled. |

**Bit 4 – WAKEUP** Wake Up Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Wake Up interrupt Enable bit and enable the WAKEUP interrupt request.

| Value | Description |
|---|---|
| 0 | The WakeUp interrupt is disabled. |
| 1 | The WakeUp interrupt is enabled. |

**Bit 3 – RST** Bus Reset Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Bus Reset interrupt Enable bit and enable the Bus RST interrupt.

| Value | Description |
|---|---|
| 0 | The Bus Reset interrupt is disabled. |
| 1 | The Bus Reset interrupt is enabled. |

**Bit 2 – HSOF** Host Start-of-Frame Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Host Start-of-Frame interrupt Enable bit and enable the HSOF interrupt.

| Value | Description |
|---|---|
| 0 | The Host Start-of-Frame interrupt is disabled. |
| 1 | The Host Start-of-Frame interrupt is enabled. |

#### 32.8.5.8 Host Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x1C
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DDISC | DCONN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAMACER | UPRSM | DNRSM | WAKEUP | RST | HSOF | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 9 – DDISC** Device Disconnection Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the device has been removed from the USB Bus and will generate an interrupt if INTENCLR/SET.DDISC is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DDISC Interrupt Flag.

**Bit 8 – DCONN** Device Connection Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a new device has been connected to the USB BUS and will generate an interrupt if INTENCLR/SET.DCONN is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the DCONN Interrupt Flag.

**Bit 7 – RAMACER** RAM Access Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a RAM access error occurs during an OUT stage and will generate an interrupt if INTENCLR/SET.RAMACER is one.

Writing a zero to this bit has no effect.

**Bit 6 – UPRSM** Upstream Resume from the Device Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB has received an Upstream Resume signal from the Device and will generate an interrupt if INTENCLR/SET.UPRSM is one.

Writing a zero to this bit has no effect.

**Bit 5 – DNRSM** Down Resume Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when the USB has sent a Down Resume and will generate an interrupt if INTENCLR/SET.DRSM is one.

Writing a zero to this bit has no effect.

**Bit 4 – WAKEUP** Wake Up Interrupt Flag
This flag is cleared by writing a one.

This flag is set when:

l The host controller is in suspend mode (SOFE is zero) and an upstream resume from the device is detected.

l The host controller is in suspend mode (SOFE is zero) and an device disconnection is detected.

l The host controller is in operational state (VBUSOK is one) and an device connection is detected.

In all cases it will generate an interrupt if INTENCLR/SET.WAKEUP is one.

Writing a zero to this bit has no effect.

**Bit 3 – RST** Bus Reset Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Bus “Reset” has been sent to the Device and will generate an interrupt if INTENCLR/SET.RST is one.

Writing a zero to this bit has no effect.

**Bit 2 – HSOF** Host Start-of-Frame Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a USB “Host Start-of-Frame” in Full Speed/High Speed or a keep-alive in Low Speed has been sent (every 1 ms) and will generate an interrupt if INTENCLR/SET.HSOF is one.

The value of the FNUM register is updated.

Writing a zero to this bit has no effect.

### 32.8.5.9 Pipe Interrupt Summary

**Name:** PINTSMRY
**Offset:** 0x20
**Reset:** 0x0000
**Property:** Read-only

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PINT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PINT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PINT[15:0]**
The flag PINT[n] is set when an interrupt is triggered by the pipe n. See 32.8.6.6 PINTFLAG register in the Host Pipe Register section.

This bit will be cleared when there are no interrupts pending for Pipe n.

Writing to this bit has no effect.

## 32.8.6 Host Registers - Pipe

#### 32.8.6.1 Host Pipe n Configuration

**Name:** PCFGn
**Offset:** 0x100 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PTYPE[2:0] | | | BK | PTOKEN[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:3 – PTYPE[2:0]** Type of the Pipe
These bits contains the pipe type.

| PTYPE[2:0] | Description |
|---|---|
| 0x0 | Pipe is disabled |
| 0x1 | Pipe is enabled and configured as CONTROL |
| 0x2 | Pipe is enabled and configured as ISO |
| 0x3 | Pipe is enabled and configured as BULK |
| 0x4 | Pipe is enabled and configured as INTERRUPT |
| 0x5 | Pipe is enabled and configured as EXTENDED |
| 0x06-0x7 | Reserved |

These bits are cleared upon sending a USB reset.

**Bit 2 – BK** Pipe Bank
This bit selects the number of banks for the pipe.

For control endpoints writing a zero to this bit is required as only Bank0 is used for Setup/In/Out transactions.

This bit is cleared when a USB reset is sent.

| BK(1) | Description |
|---|---|
| 0x0 | Single-bank endpoint |
| 0x1 | Dual-bank endpoint |

1. Bank field is ignored when PTYPE is configured as EXTENDED.

| Value | Description |
|---|---|
| 0 | A single bank is used for the pipe. |
| 1 | A dual bank is used for the pipe. |

**Bits 1:0 – PTOKEN[1:0]** Pipe Token
These bits contains the pipe token.

| PTOKEN[1:0](1) | Description |
|---|---|
| 0x0 | SETUP(2) |
| 0x1 | IN |
| 0x2 | OUT |
| 0x3 | Reserved |

1. PTOKEN field is ignored when PTYPE is configured as EXTENDED.
2. Available only when PTYPE is configured as CONTROL

Theses bits are cleared upon sending a USB reset.

#### 32.8.6.2 Interval for the Bulk-Out/Ping Transaction

**Name:** BINTERVAL
**Offset:** 0x103 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BINTERVAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – BINTERVAL[7:0]** BINTERVAL
These bits contains the Ping/Bulk-out period.

These bits are cleared when a USB reset is sent or when PEN[n] is zero.

| BINTERVAL | Description |
|---|---|
| =0 | Multiple consecutive OUT token is sent in the same frame until it is acked by the peripheral |
| >0 | One OUT token is sent every BINTERVAL frame until it is acked by the peripheral |

Depending from the type of pipe the desired period is defined as:

| PTYPE | Description |
|---|---|
| Interrupt | 1 ms to 255 ms |
| Isochronous | 2^(Binterval) * 1 ms |
| Bulk or control | 1 ms to 255 ms |
| EXT LPM | bInterval ignored. Always 1 ms when a NYET is received. |

### 32.8.6.3 Pipe Status Clear n

**Name:** PSTATUSCLR
**Offset:** 0x104 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| Access | W | W | | W | | W | | W |
| Reset | 0 | 0 | | 0 | | 0 | | 0 |

**Bit 7 – BK1RDY** Bank 1 Ready Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear PSTATUS.BK1RDY bit.

**Bit 6 – BK0RDY** Bank 0 Ready Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear PSTATUS.BK0RDY bit.

**Bit 4 – PFREEZE** Pipe Freeze Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear PSTATUS.PFREEZE bit.

**Bit 2 – CURBK** Current Bank Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear PSTATUS.CURBK bit.

**Bit 0 – DTGL** Data Toggle Clear
Writing a zero to this bit has no effect.

Writing a one to this bit will clear PSTATUS.DTGL bit.

#### 32.8.6.4 Pipe Status Set Register n

**Name:** PSTATUSSET
**Offset:** 0x105 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| Access | W | W | | W | | W | | W |
| Reset | 0 | 0 | | 0 | | 0 | | 0 |

**Bit 7 – BK1RDY** Bank 1 Ready Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set the bit PSTATUS.BK1RDY.

**Bit 6 – BK0RDY** Bank 0 Ready Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set the bit PSTATUS.BK0RDY.

**Bit 4 – PFREEZE** Pipe Freeze Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set PSTATUS.PFREEZE bit.

**Bit 2 – CURBK** Current Bank Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set PSTATUS.CURBK bit.

**Bit 0 – DTGL** Data Toggle Set
Writing a zero to this bit has no effect.

Writing a one to this bit will set PSTATUS.DTGL bit.

© 2018 Microchip Technology Inc.

### 32.8.6.5 Pipe Status Register n

**Name:** PSTATUS
**Offset:** 0x106 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BK1RDY | BK0RDY | | PFREEZE | | CURBK | | DTGL |
| Access | R | R | | R | | R | | R |
| Reset | 0 | 0 | | 0 | | 0 | | 0 |

**Bit 7 – BK1RDY** Bank 1 is ready
Writing a one to the bit EPSTATUSCLR.BK1RDY will clear this bit.

Writing a one to the bit EPSTATUSSET.BK1RDY will set this bit.

This bank is not used for Control pipe.

| Value | Description |
|---|---|
| 0 | The bank number 1 is not ready: For IN the bank is empty. For Control/OUT the bank is not yet fill in. |
| 1 | The bank number 1 is ready: For IN the bank is filled full. For Control/OUT the bank is filled in. |

**Bit 6 – BK0RDY** Bank 0 is ready
Writing a one to the bit EPSTATUSCLR.BK0RDY will clear this bit.

Writing a one to the bit EPSTATUSSET.BK0RDY will set this bit.

This bank is the only one used for Control pipe.

| Value | Description |
|---|---|
| 0 | The bank number 0 is not ready: For IN the bank is not empty. For Control/OUT the bank is not yet fill in. |
| 1 | The bank number 0 is ready: For IN the bank is filled full. For Control/OUT the bank is filled in. |

**Bit 4 – PFREEZE** Pipe Freeze
Writing a one to the bit EPSTATUSCLR.PFREEZE will clear this bit.

Writing a one to the bit EPSTATUSSET.PFREEZE will set this bit.

This bit is also set by the hardware:

- When a STALL handshake has been received.
- After a PIPE has been enabled (rising of bit PEN.N).
- When an LPM transaction has completed whatever handshake is returned or the transaction was timed-out.
- When a pipe transfer was completed with a pipe error. See 32.8.6.6 PINTFLAG register.

When PFREEZE bit is set while a transaction is in progress on the USB bus, this transaction will be properly completed. PFREEZE bit will be read as “1” only when the ongoing transaction will have been completed.

| Value | Description |
|---|---|
| 0 | The Pipe operates in normal operation. |
| 1 | The Pipe is frozen and no additional requests will be sent to the device on this pipe address. |

**Bit 2 – CURBK** Current Bank

| Value | Description |
|---|---|
| 0 | The bank0 is the bank that will be used in the next single/multi USB packet. |
| 1 | The bank1 is the bank that will be used in the next single/multi USB packet. |

**Bit 0 – DTGL** Data Toggle Sequence
Writing a one to the bit EPSTATUSCLR.DTGL will clear this bit.

Writing a one to the bit EPSTATUSSET.DTGL will set this bit.

This bit is toggled automatically by hardware after a data transaction.

This bit will reflect the data toggle in regards of the token type (IN/OUT/SETUP).

| Value | Description |
|---|---|
| 0 | The PID of the next expected transaction will be zero: data 0. |
| 1 | The PID of the next expected transaction will be one: data 1. |

#### 32.8.6.6 Host Pipe Interrupt Flag Register

**Name:** PINTFLAG
**Offset:** 0x107 + (n x 0x20)
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | TXSTP | PERR | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 2 |

**Bit 5 – STALL** STALL Received Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a stall occurs and will generate an interrupt if PINTENCLR/SET.STALL is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the STALL Interrupt Flag.

**Bit 4 – TXSTP** Transmitted Setup Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Transfer Complete occurs and will generate an interrupt if PINTENCLR/SET.TXSTP is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the TXSTP Interrupt Flag.

**Bit 3 – PERR** Pipe Error Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a pipe error occurs and will generate an interrupt if PINTENCLR/SET.PERR is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the PERR Interrupt Flag.

**Bit 2 – TRFAIL** Transfer Fail Interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Transfer Fail occurs and will generate an interrupt if PINTENCLR/SET.TRFAIL is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the TRFAIL Interrupt Flag.

**Bit 0 – TRCPT** Transfer Complete x interrupt Flag
This flag is cleared by writing a one to the flag.

This flag is set when a Transfer complete occurs and will generate an interrupt if PINTENCLR/SET.TRCPT is one. PINTFLAG.TRCPT is set for a single bank IN/OUT pipe or a double bank IN/OUT pipe when current bank is 0.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the TRCPT Interrupt Flag.

### 32.8.6.7 Host Pipe Interrupt Clear Register

**Name:** PINTENCLR
**Offset:** 0x108 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Pipe Interrupt Enable Set (PINTENSET) register.

This register is cleared by USB reset or when PEN[n] is zero.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | TXSTP | PERR | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 2 |

**Bit 5 – STALL** Received Stall Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Received Stall interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The received Stall interrupt is disabled. |
| 1 | The received Stall interrupt is enabled and an interrupt request will be generated when the received Stall interrupt Flag is set. |

**Bit 4 – TXSTP** Transmitted Setup Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transmitted Setup interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transmitted Setup interrupt is disabled. |
| 1 | The Transmitted Setup interrupt is enabled and an interrupt request will be generated when the Transmitted Setup interrupt Flag is set. |

**Bit 3 – PERR** Pipe Error Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Pipe Error interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Pipe Error interrupt is disabled. |
| 1 | The Pipe Error interrupt is enabled and an interrupt request will be generated when the Pipe Error interrupt Flag is set. |

**Bit 2 – TRFAIL** Transfer Fail Interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transfer Fail interrupt Enable bit and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transfer Fail interrupt is disabled. |
| 1 | The Transfer Fail interrupt is enabled and an interrupt request will be generated when the Transfer Fail interrupt Flag is set. |

**Bit 0 – TRCPT** Transfer Complete Bank x interrupt Disable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Transfer Complete interrupt Enable bit x and disable the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Transfer Complete Bank x interrupt is disabled. |
| 1 | The Transfer Complete Bank x interrupt is enabled and an interrupt request will be generated when the Transfer Complete interrupt x Flag is set. |

### 32.8.6.8 Host Interrupt Pipe Set Register

**Name:** PINTENSET
**Offset:** 0x109 + (n x 0x20)
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Pipe Interrupt Enable Set (PINTENCLR) register.

This register is cleared by USB reset or when PEN[n] is zero.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | STALL | TXSTP | PERR | TRFAIL | | TRCPT |
| Access | | | R/W | R/W | R/W | R/W | | R/W |
| Reset | | | 0 | 0 | 0 | 0 | | 2 |

**Bit 5 – STALL** Stall Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Stall interrupt.

| Value | Description |
|---|---|
| 0 | The Stall interrupt is disabled. |
| 1 | The Stall interrupt is enabled. |

**Bit 4 – TXSTP** Transmitted Setup Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Transmitted Setup interrupt.

| Value | Description |
|---|---|
| 0 | The Transmitted Setup interrupt is disabled. |
| 1 | The Transmitted Setup interrupt is enabled. |

**Bit 3 – PERR** Pipe Error Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Pipe Error interrupt.

| Value | Description |
|---|---|
| 0 | The Pipe Error interrupt is disabled. |
| 1 | The Pipe Error interrupt is enabled. |

**Bit 2 – TRFAIL** Transfer Fail Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will enable the Transfer Fail interrupt.

| Value | Description |
|---|---|
| 0 | The Transfer Fail interrupt is disabled. |
| 1 | The Transfer Fail interrupt is enabled. |

**Bit 0 – TRCPT** Transfer Complete x interrupt Enable
Writing a zero to this bit has no effect.

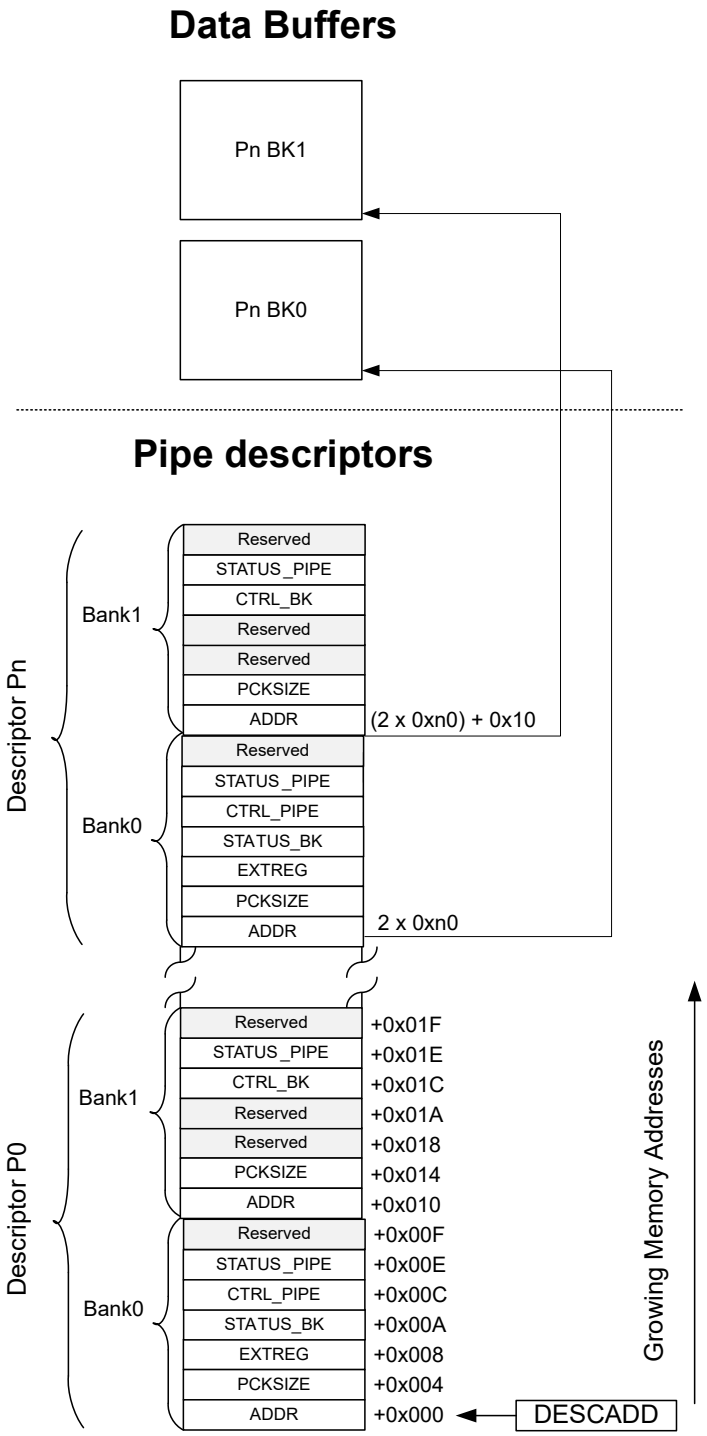
Writing a one to this bit will enable the Transfer Complete interrupt Enable bit x.

0.2.7 Host Registers - Pipe RAM

| Value | Description |
|---|---|
| 0 | The Transfer Complete x interrupt is disabled. |
| 1 | The Transfer Complete x interrupt is enabled. |

### 32.8.7 Host Registers - Pipe RAM

#### 32.8.7.1 Pipe Descriptor Structure



#### 32.8.7.2 Address of the Data Buffer

**Name:** ADDR
**Offset:** 0x00 & 0x10
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x |

**Bits 31:0 – ADDR[31:0]** Data Pointer Address Value
These bits define the data pointer address as an absolute double word address in RAM. The two least significant bits must be zero to ensure the descriptor is 32-bit aligned.

#### 32.8.7.3 Packet Size

**Name:** PCKSIZE
**Offset:** 0x04 & 0x14
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AUTO_ZLP | SIZE[2:0] | | | MULTI_PACKET_SIZE[13:10] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | 0 | 0 | x | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MULTI_PACKET_SIZE[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MULTI_PACKET_SIZE[1:0] | | BYTE_COUNT[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | x | 0 | 0 | 0 | 0 | 0 | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – AUTO_ZLP** Automatic Zero Length Packet
This bit defines the automatic Zero Length Packet mode of the pipe.

When enabled, the USB module will manage the ZLP handshake by hardware. This bit is for OUT pipes only. When disabled the handshake should be managed by firmware.

| Value | Description |
|---|---|
| 0 | Automatic Zero Length Packet is disabled. |
| 1 | Automatic Zero Length Packet is enabled. |

**Bits 30:28 – SIZE[2:0]** Pipe size
These bits contains the size of the pipe.

Theses bits are cleared upon sending a USB reset.

| SIZE[2:0] | Description |
|---|---|
| 0x0 | 8 Byte |
| 0x1 | 16 Byte |
| 0x2 | 32 Byte |
| 0x3 | 64 Byte |

| SIZE[2:0] | Description |
|---|---|
| 0x4 | 128 Byte(1) |
| 0x5 | 256 Byte(1) |
| 0x6 | 512 Byte(1) |
| 0x7 | 1024 Byte in HS mode(1)<br>1023 Byte in FS mode(1) |

1. For Isochronous pipe only.

**Bits 27:14 – MULTI_PACKET_SIZE[13:0]** Multi Packet IN or OUT size
These bits define the 14-bit value that is used for multi-packet transfers.

For IN pipes, MULTI_PACKET_SIZE holds the total number of bytes sent. MULTI_PACKET_SIZE should be written to zero when setting up a new transfer.

For OUT pipes, MULTI_PACKET_SIZE holds the total data size for the complete transfer. This value must be a multiple of the maximum packet size.

**Bits 13:8 – BYTE_COUNT[5:0]** Byte Count
These bits define the 14-bit value that contains number of bytes sent in the last OUT or SETUP transaction for an OUT pipe, or of the number of bytes to be received in the next IN transaction for an input pipe.

### 32.8.7.4 Extended Register

**Name:** EXTREG
**Offset:** 0x08
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | VARIABLE[10:4] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VARIABLE[3:0] | | | | SUBPID[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | x | 0 | 0 | 0 | x |

**Bits 14:4 – VARIABLE[10:0]** Variable field send with extended token
These bits define the VARIABLE field sent with extended token. See “Section 2.1.1 Protocol Extension Token in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum.”

To support the USB2.0 Link Power Management addition the VARIABLE field should be set as described below.

| VARIABLE | Description |
|---|---|
| VARIABLE[3:0] | bLinkState(1) |
| VARIABLE[7:4] | BESL (See LPM ECN)(2) |
| VARIABLE[8] | bRemoteWake(1) |
| VARIABLE[10:9] | Reserved |

(1) for a definition of LPM Token bRemoteWake and bLinkState fields, refer to "Table 2-3 in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum"
(2) for a definition of LPM Token BESL field, refer to "Table 2-3 in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum" and "Table X-X1 in Errata for ECN USB 2.0 Link Power Management.

**Bits 3:0 – SUBPID[3:0]** SUBPID field send with extended token
These bits define the SUBPID field sent with extended token. See “Section 2.1.1 Protocol Extension Token in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum”.

To support the USB2.0 Link Power Management addition the SUBPID field should be set as described in “Table 2.2 SubPID Types in the reference document ENGINEERING CHANGE NOTICE, USB 2.0 Link Power Management Addendum”.

#### 32.8.7.5 Host Status Bank

**Name:** STATUS_BK
**Offset:** 0x0A & 0x1A
**Reset:** 0xxxxxxxx
**Property:** NA

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ERRORFLOW | CRCERR |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | x | x |

**Bit 1 – ERRORFLOW** Error Flow Status
This bit defines the Error Flow Status.

This bit is set when a Error Flow has been detected during transfer from/towards this bank.

For IN transfer, a NAK handshake has been received. For OUT transfer, a NAK handshake has been received. For Isochronous IN transfer, an overrun condition has occurred. For Isochronous OUT transfer, an underflow condition has occurred.

| Value | Description |
|---|---|
| 0 | No Error Flow detected. |
| 1 | A Error Flow has been detected. |

**Bit 0 – CRCERR** CRC Error
This bit defines the CRC Error Status.

This bit is set when a CRC error has been detected in an isochronous IN endpoint bank.

| Value | Description |
|---|---|
| 0 | No CRC Error. |
| 1 | CRC Error detected. |

#### 32.8.7.6 Host Control Pipe

**Name:** CTRL_PIPE
**Offset:** 0x0C
**Reset:** 0xXXXX
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PERMAX[3:0] | | | | PEPNUM[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | x | 0 | 0 | 0 | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PDADDR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | x |

**Bits 15:12 – PERMAX[3:0]** Pipe Error Max Number
These bits define the maximum number of error for this Pipe before freezing the pipe automatically.

**Bits 11:8 – PEPNUM[3:0]** Pipe EndPoint Number
These bits define the number of endpoint for this Pipe.

**Bits 6:0 – PDADDR[6:0]** Pipe Device Address
These bits define the Device Address for this pipe.

#### 32.8.7.7 Host Status Pipe

**Name:** STATUS_PIPE
**Offset:** 0x0E & 0x1E
**Reset:** 0xxxxxxxx
**Property:** PAC Write-Protection, Write-Synchronized, Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERCNT[2:0] | | | CRC16ER | TOUTER | PIDER | DAPIDER | DTGLER |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | x | x | x | x | x | x |

**Bits 7:5 – ERCNT[2:0]** Pipe Error Counter
These bits define the number of errors detected on the pipe.

**Bit 4 – CRC16ER** CRC16 ERROR
This bit defines the CRC16 Error Status.

This bit is set when a CRC 16 error has been detected during a IN transactions.

| Value | Description |
|---|---|
| 0 | No CRC 16 Error detected. |
| 1 | A CRC 16 error has been detected. |

**Bit 3 – TOUTER** TIME OUT ERROR
This bit defines the Time Out Error Status.

This bit is set when a Time Out error has been detected during a USB transaction.

| Value | Description |
|---|---|
| 0 | No Time Out Error detected. |
| 1 | A Time Out error has been detected. |

**Bit 2 – PIDER** PID ERROR
This bit defines the PID Error Status.

This bit is set when a PID error has been detected during a USB transaction.

| Value | Description |
|---|---|
| 0 | No PID Error detected. |
| 1 | A PID error has been detected. |

**Bit 1 – DAPIDER** Data PID ERROR
This bit defines the PID Error Status.

This bit is set when a Data PID error has been detected during a USB transaction.

| Value | Description |
|---|---|
| 0 | No Data PID Error detected. |
| 1 | A Data PID error has been detected. |

**Bit 0 – DTGLER** Data Toggle Error
This bit defines the Data Toggle Error Status.

This bit is set when a Data Toggle Error has been detected.

| Value | Description |
|---|---|
| 0 | No Data Toggle Error. |
| 1 | Data Toggle Error detected. |

# 33. ADC – Analog-to-Digital Converter

## 33.1 Overview

The Analog-to-Digital Converter (ADC) converts analog signals to digital values. The ADC has 12-bit resolution, and is capable of converting up to 350ksps. The input selection is flexible, and both differential and single-ended measurements can be performed. An optional gain stage is available to increase the dynamic range. In addition, several internal signal inputs are available. The ADC can provide both signed and unsigned results.

ADC measurements can be started by either application software or an incoming event from another peripheral in the device. ADC measurements can be started with predictable timing, and without software intervention.

Both internal and external reference voltages can be used.

An integrated temperature sensor is available for use with the ADC. The bandgap voltage as well as the scaled I/O and core voltages can also be measured by the ADC.

The ADC has a compare function for accurate monitoring of user-defined thresholds, with minimum software intervention required.

The ADC may be configured for 8-, 10- or 12-bit results, reducing the conversion time. ADC conversion results are provided left- or right-adjusted, which eases calculation when the result is represented as a signed value. It is possible to use DMA to move ADC results directly to memory or peripherals when conversions are done.

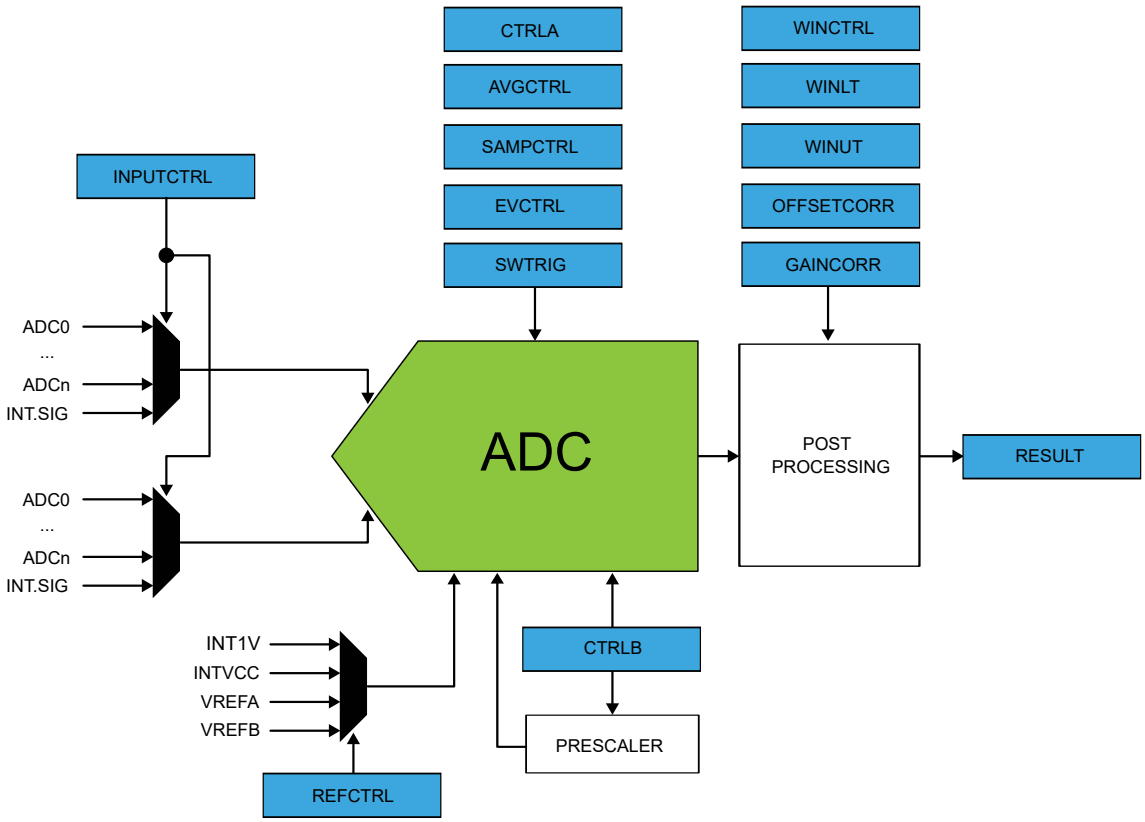
## 33.2 Features

- 8-, 10- or 12-bit resolution
- Up to 350,000 samples per second (350ksps)
- Differential and single-ended inputs
  - Up to 32 analog input
  - 25 positive and 10 negative, including internal and external
- Five internal inputs
  - Bandgap
  - Temperature sensor
  - DAC
  - Scaled core supply
  - Scaled I/O supply
- 1/2x to 16x gain
- Single, continuous and pin-scan conversion options
- Windowing monitor with selectable channel
- Conversion range:
  - $V_{ref}$ [1v to $V_{DDANA}$ - 0.6V]
  - ADCx * GAIN [0V to -$V_{ref}$]
- Built-in internal reference and external reference options

  - Four bits for reference selection
- Event-triggered conversion for accurate timing (one event input)
- Optional DMA transfer of conversion result
- Hardware gain and offset compensation
- Averaging and oversampling with decimation to support, up to 16-bit result
- Selectable sampling time

## 33.3 Block Diagram

**Figure 33-1. ADC Block Diagram**




## 33.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| VREFA | Analog input | External reference voltage A |
| VREFB | Analog input | External reference voltage B |
| ADC[19..0](1) | Analog input | Analog input channels |

**Note:** Refer to *Configuration Summary* for details on exact number of analog input channels.

**Note:** Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations
2. Configuration Summary

## 33.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 33.5.1 I/O Lines

Using the ADC's I/O lines requires the I/O pins to be configured using the port configuration (PORT).

**Related Links**

23. PORT - I/O Pin Controller

### 33.5.2 Power Management

The ADC will continue to operate in any Sleep mode where the selected source clock is running. The ADC's interrupts, except the OVERRUN interrupt, can be used to wake up the device from sleep modes, except the OVERRUN interrupt. Events connected to the event system can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 33.5.3 Clocks

The ADC bus clock (CLK_APB_ADCx) can be enabled in the Main Clock, which also defines the default state.

The ADC requires a generic clock (GCLK_ADC). This clock must be configured and enabled in the Generic Clock Controller (GCLK) before using the ADC.

A generic clock is asynchronous to the bus clock. Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to *Synchronization* for further details.

**Related Links**

16.6.2.6 Peripheral Clock Masking
15. GCLK - Generic Clock Controller

### 33.5.4 DMA

The DMA request line is connected to the DMA Controller (DMAC). Using the ADC DMA requests requires the DMA Controller to be configured first.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 33.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the ADC interrupt requires the interrupt controller to be configured first.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 33.5.6 Events

The events are connected to the Event System.

**Related Links**

24. EVSYS – Event System

### 33.5.7 Debug Operation

When the CPU is halted in debug mode the ADC will halt normal operation. The ADC can be forced to continue operation during debugging.

### 33.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the peripheral access controller (PAC), except the following register:

- Interrupt Flag Status and Clear (INTFLAG) register

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 33.5.9 Analog Connections

I/O-pins AIN0 to AIN19 as well as the VREFA/VREFB reference voltage pin are analog inputs to the ADC.

### 33.5.10 Calibration

The BIAS and LINEARITY calibration values from the production test must be loaded from the NVM Software Calibration Area into the ADC Calibration register (CALIB) by software to achieve specified accuracy.

**Related Links**

10.3.2 NVM Software Calibration Area Mapping

## 33.6 Functional Description

### 33.6.1 Principle of Operation

By default, the ADC provides results with 12-bit resolution. 8-bit or 10-bit results can be selected in order to reduce the conversion time.

The ADC has an oversampling with decimation option that can extend the resolution to 16 bits. The input values can be either internal (e.g., internal temperature sensor) or external (connected I/O pins). The user can also configure whether the conversion should be single-ended or differential.

### 33.6.2 Basic Operation

#### 33.6.2.1 Initialization

Before enabling the ADC, the asynchronous clock source must be selected and enabled, and the ADC reference must be configured. The first conversion after the reference is changed must not be used. All

other configuration registers must be stable during the conversion. The source for GCLK_ADC is selected and enabled in the System Controller (SYSCTRL). Refer to *SYSCTRL – System Controller* for more details.

When GCLK_ADC is enabled, the ADC can be enabled by writing a one to the Enable bit in the Control Register A (CTRLA.ENABLE).

**Related Links**

17. SYSCTRL – System Controller

#### 33.6.2.2 Enabling, Disabling and Reset

The ADC is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The ADC is disabled by writing CTRLA.ENABLE=0. The ADC is reset by writing a '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the ADC, except DBGCTRL, will be reset to their initial state, and the ADC will be disabled.

The ADC must be disabled before it is reset.

#### 33.6.2.3 Operation

In the most basic configuration, the ADC samples values from the configured internal or external sources (INPUTCTRL register). The rate of the conversion depends on the combination of the GCLK_ADCx frequency and the clock prescaler.

To convert analog values to digital values, the ADC needs to be initialized first, as described in 33.6.2.1 Initialization. Data conversion can be started either manually by setting the Start bit in the Software Trigger register (SWTRIG.START=1), or automatically by configuring an automatic trigger to initiate the conversions. A free-running mode can be used to continuously convert an input channel. When using free-running mode the first conversion must be started, while subsequent conversions will start automatically at the end of previous conversions.

The automatic trigger can be configured to trigger on many different conditions.

The result of the conversion is stored in the Result register (RESULT) overwriting the result from the previous conversion.

To avoid data loss if more than one channel is enabled, the conversion result must be read as soon as it is available (INTFLAG.RESRDY). Failing to do so will result in an overrun error condition, indicated by the OVERRUN bit in the Interrupt Flag Status and Clear register (INTFLAG.OVERRUN). When the RESRDY interrupt flag is set, the new result has been synchronized to the RESULT register.

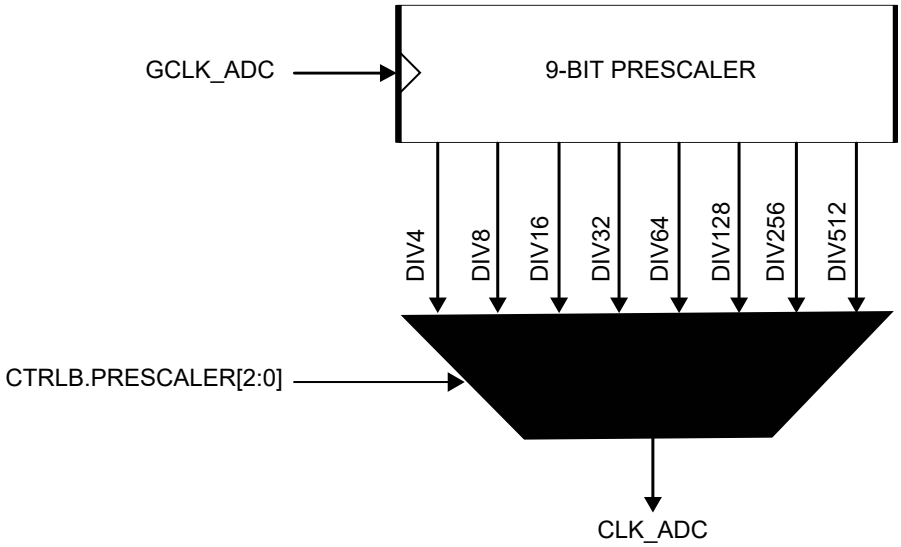
To enable one of the available interrupts sources, the corresponding bit in the Interrupt Enable Set register (INTENSET) must be written to '1'.

### 33.6.3 Prescaler

The ADC is clocked by GCLK_ADC. There is also a prescaler in the ADC to enable conversion at lower clock rates.

Refer to CTRLB for details on prescaler settings.

**Figure 33-2. ADC Prescaler**




The propagation delay of an ADC measurement depends on the selected mode and is given by:

- Single-shot mode:

$$\text{PropagationDelay} = \frac{1 + \frac{\text{Resolution}}{2} + \text{DelayGain}}{f_{\text{CLK+ - ADC}}}$$

- Free-running mode:

$$\text{PropagationDelay} = \frac{\frac{\text{Resolution}}{2} + \text{DelayGain}}{f_{\text{CLK+ - ADC}}}$$

**Table 33-1. Delay Gain**

| | | Delay Gain (in CLK_ADC Period) | | | |
|---|---|---|---|---|---|
| | INTPUTCTRL.GAIN[3:0] | Free-running mode | | Single shot mode | |
| Name | | Differential Mode | Single-Ended Mode | Differential mode | Single-Ended mode |
| 1X | 0x0 | 0 | 0 | 0 | 1 |
| 2X | 0x1 | 0 | 1 | 0.5 | 1.5 |
| 4X | 0x2 | 1 | 1 | 1 | 2 |
| 8X | 0x3 | 1 | 2 | 1.5 | 2.5 |
| 16X | 0x4 | 2 | 2 | 2 | 3 |
| Reserved | 0x5 ... 0xE | Reserved | Reserved | Reserved | Reserved |
| DIV2 | 0xF | 0 | 1 | 0.5 | 1.5 |

### 33.6.4 ADC Resolution

The ADC supports 8-bit, 10-bit or 12-bit resolution. Resolution can be changed by writing the Resolution bit group in the Control B register (CTRLB.RESSEL). By default, the ADC resolution is set to 12 bits.

### 33.6.5 Differential and Single-Ended Conversions

The ADC has two conversion options: differential and single-ended:

- If the positive input may go below the negative input, the **differential** mode should be used in order to get correct results.
- If the positive input is always positive, the **single-ended** conversion should be used in order to have full 12-bit resolution in the conversion.

The negative input must be connected to ground. This ground could be the internal GND, IOGND or an external ground connected to a pin. Refer to the Control B (CTRLB) register for selection details.

If the positive input may go below the negative input, creating some negative results, the differential mode should be used in order to get correct results. The differential mode is enabled by setting DIFFMODE bit in the Control B register (CTRLB.DIFFMODE). Both conversion types could be run in single mode or in free-running mode. When the free-running mode is selected, an ADC input will continuously sample the input and performs a new conversion. The INTFLAG.RESRDY bit will be set at the end of each conversion.

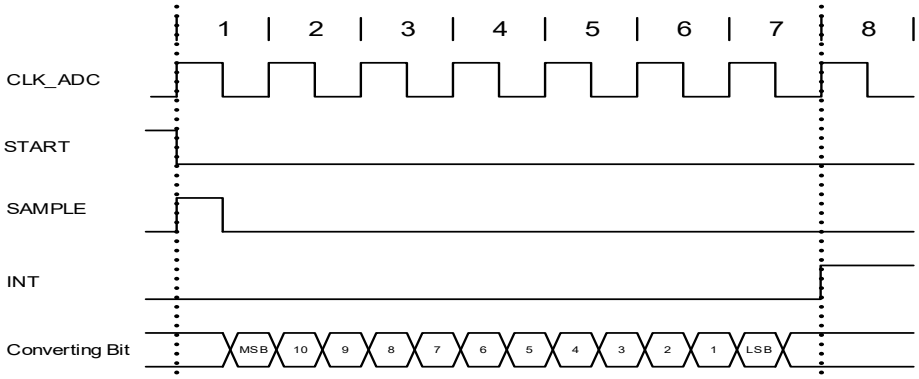
**Related Links**

33.8.5 CTRLB

#### 33.6.5.1 Conversion Timing

The following figure shows the ADC timing for one single conversion. A conversion starts after the software or event start are synchronized with the GCLK_ADC clock. The input channel is sampled in the first half CLK_ADC period.

**Figure 33-3. ADC Timing for One Conversion in Differential Mode without Gain**




The sampling time can be increased by using the Sampling Time Length bit group in the Sampling Time Control register (SAMPCTRL.SAMPLEN). As example, the next figure is showing the timing conversion.

**Figure 33-4. ADC Timing for One Conversion in Differential Mode without Gain, but with Increased Sampling Time**

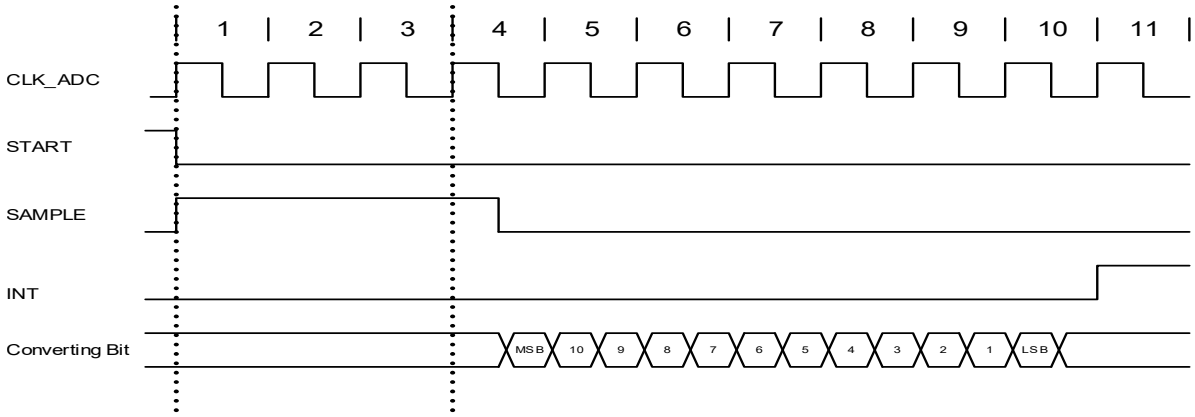



**Figure 33-5. ADC Timing for Free Running in Differential Mode without Gain**

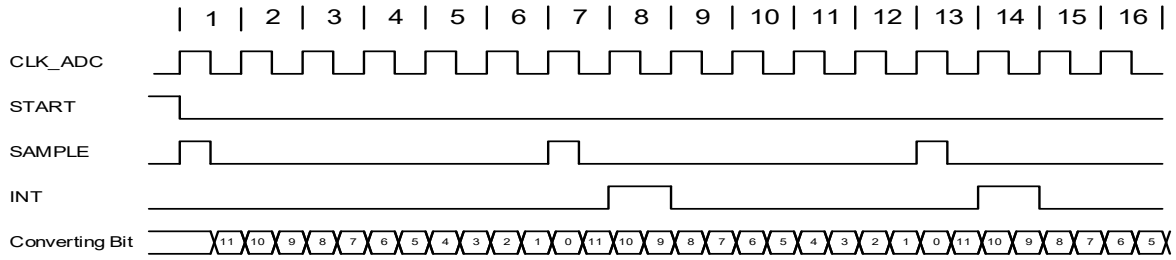



**Figure 33-6. ADC Timing for One Conversion in Single-Ended Mode without Gain**

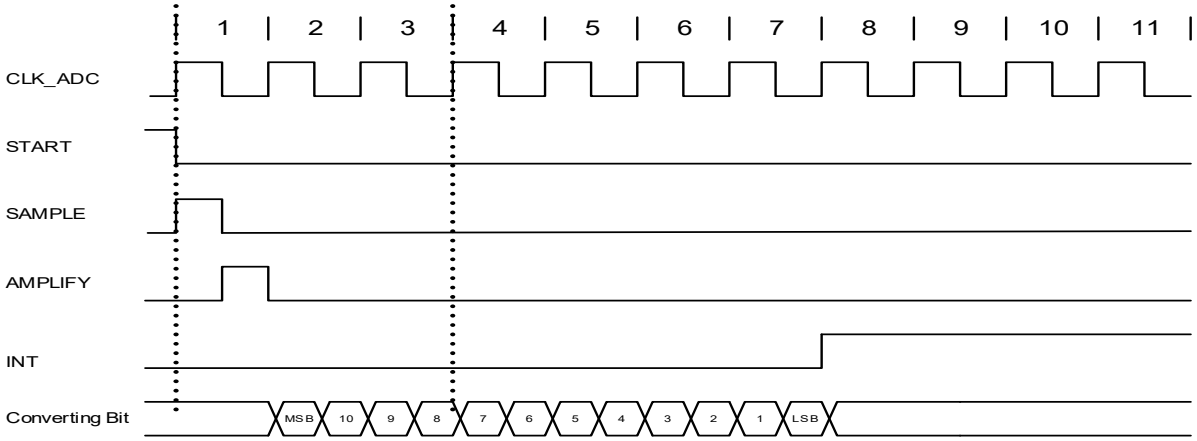
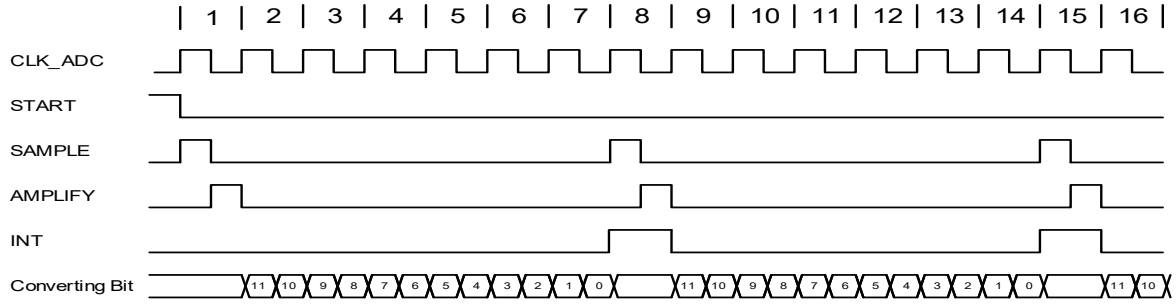



**Figure 33-7. ADC Timing for Free Running in Single-Ended Mode without Gain**



### 33.6.6 Accumulation

The result from multiple consecutive conversions can be accumulated. The number of samples to be accumulated is specified by the Number of Samples to be Collected field in the Average Control register (AVGCTRL.SAMPLENUM). When accumulating more than 16 samples, the result will be too large to match the 16-bit RESULT register size. To avoid overflow, the result is right shifted automatically to fit within the available register size. The number of automatic right shifts is specified in the table below.

**Note:** To perform the accumulation of two or more samples, the Conversion Result Resolution field in the Control B register (CTRLB.RESSEL) must be set.

**Table 33-2. Accumulation**

| Number of Accumulated Samples | AVGCTRL. SAMPLENUM | Intermediate Result Precision | Number of Automatic Right Shifts | Final Result Precision | Automatic Division Factor |
|---|---|---|---|---|---|
| 1 | 0x0 | 12 bits | 0 | 12 bits | 0 |
| 2 | 0x1 | 13 bits | 0 | 13 bits | 0 |
| 4 | 0x2 | 14 bits | 0 | 14 bits | 0 |
| 8 | 0x3 | 15 bits | 0 | 15 bits | 0 |
| 16 | 0x4 | 16 bits | 0 | 16 bits | 0 |
| 32 | 0x5 | 17 bits | 1 | 16 bits | 2 |
| 64 | 0x6 | 18 bits | 2 | 16 bits | 4 |
| 128 | 0x7 | 19 bits | 3 | 16 bits | 8 |
| 256 | 0x8 | 20 bits | 4 | 16 bits | 16 |
| 512 | 0x9 | 21 bits | 5 | 16 bits | 32 |
| 1024 | 0xA | 22 bits | 6 | 16 bits | 64 |
| Reserved | 0xB - 0xF | 12 bits | | 12 bits | 0 |

### 33.6.7 Averaging

Averaging is a feature that increases the sample accuracy, at the cost of a reduced sampling rate. This feature is suitable when operating in noisy conditions.

Averaging is done by accumulating m samples, as described in 33.6.6 Accumulation, and dividing the result by m. The averaged result is available in the RESULT register. The number of samples to be accumulated is specified by writing to AVGCTRL.SAMPLENUM.

The division is obtained by a combination of the automatic right shift described above, and an additional right shift that must be specified by writing to the Adjusting Result/Division Coefficient field in AVGCTRL (AVGCTRL.ADJRES).

**Note:** To perform the averaging of two or more samples, the Conversion Result Resolution field in the Control B register (CTRLB.RESSEL) must be set to '1'.

Averaging AVGCTRL.SAMPLENUM samples will reduce the un-averaged sampling rate by a factor $\frac{1}{\text{AVGCTRL.SAMPLENUM}}$.

When the averaged result is available, the INTFLAG.RESRDY bit will be set.

**Table 33-3. Averaging**

| Number of Accumulated Samples | AVGCTRL. SAMPLENUM | Intermediate Result Precision | Number of Automatic Right Shifts | Division Factor | AVGCTRL.ADJRES | Total Number of Right Shifts | Final Result Precision | Automatic Division Factor |
|---|---|---|---|---|---|---|---|---|
| 1 | 0x0 | 12 bits | 0 | 1 | 0x0 | | 12 bits | 0 |
| 2 | 0x1 | 13 | 0 | 2 | 0x1 | 1 | 12 bits | 0 |
| 4 | 0x2 | 14 | 0 | 4 | 0x2 | 2 | 12 bits | 0 |
| 8 | 0x3 | 15 | 0 | 8 | 0x3 | 3 | 12 bits | 0 |
| 16 | 0x4 | 16 | 0 | 16 | 0x4 | 4 | 12 bits | 0 |
| 32 | 0x5 | 17 | 1 | 16 | 0x4 | 5 | 12 bits | 2 |
| 64 | 0x6 | 18 | 2 | 16 | 0x4 | 6 | 12 bits | 4 |
| 128 | 0x7 | 19 | 3 | 16 | 0x4 | 7 | 12 bits | 8 |
| 256 | 0x8 | 20 | 4 | 16 | 0x4 | 8 | 12 bits | 16 |
| 512 | 0x9 | 21 | 5 | 16 | 0x4 | 9 | 12 bits | 32 |
| 1024 | 0xA | 22 | 6 | 16 | 0x4 | 10 | 12 bits | 64 |
| Reserved | 0xB-0xF | | | | 0x0 | | 12 bits | 0 |

### 33.6.8 Oversampling and Decimation

By using oversampling and decimation, the ADC resolution can be increased from 12 bits up to 16 bits, for the cost of reduced effective sampling rate.

To increase the resolution by n bits, $4^n$ samples must be accumulated. The result must then be right-shifted by n bits. This right-shift is a combination of the automatic right-shift and the value written to AVGCTRL.ADJRES. To obtain the correct resolution, the ADJRES must be configured as described in the table below. This method will result in n bit extra LSB resolution.

**Table 33-4. Configuration Required for Oversampling and Decimation**

| Result Resolution | Number of Samples to Average | AVGCTRL.SAMPLENUM[3:0] | Number of Automatic Right Shifts | AVGCTRL.ADJRES[2:0] |
|---|---|---|---|---|
| 13 bits | $4^1 = 4$ | 0x2 | 0 | 0x1 |
| 14 bits | $4^2 = 16$ | 0x4 | 0 | 0x2 |
| 15 bits | $4^3 = 64$ | 0x6 | 2 | 0x1 |
| 16 bits | $4^4 = 256$ | 0x8 | 4 | 0x0 |

### 33.6.9 Window Monitor

The window monitor feature allows the conversion result in the RESULT register to be compared to predefined threshold values. The window mode is selected by setting the Window Monitor Mode bits in the Window Monitor Control register (WINCTRL.WINMODE[2:0]). Threshold values must be written in the Window Monitor Lower Threshold register (WINLT) and Window Monitor Upper Threshold register (WINUT).

If differential input is selected, the WINLT and WINUT are evaluated as signed values. Otherwise they are evaluated as unsigned values. The significant WINLT and WINUT bits are given by the precision selected in the Conversion Result Resolution bit group in the Control B register (CTRLB.RESSEL). This means

that e.g. in 8-bit mode, only the eight lower bits will be considered. In addition, in differential mode, the eighth bit will be considered as the sign bit, even if the ninth bit is zero.

The INTFLAG.WINMON interrupt flag will be set if the conversion result matches the window monitor condition.

### 33.6.10 Offset and Gain Correction

Inherent gain and offset errors affect the absolute accuracy of the ADC.

The offset error is defined as the deviation of the actual ADC transfer function from an ideal straight line at zero input voltage. The offset error cancellation is handled by the Offset Correction register (OFFSETCORR). The offset correction value is subtracted from the converted data before writing the Result register (RESULT).

The gain error is defined as the deviation of the last output step's midpoint from the ideal straight line, after compensating for offset error. The gain error cancellation is handled by the Gain Correction register (GAINCORR).

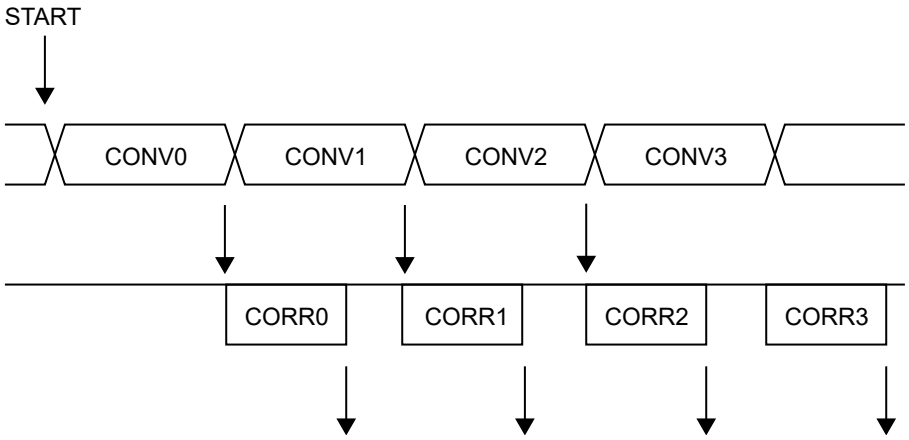
To correct these two errors, the Digital Correction Logic Enabled bit in the Control B register (CTRLB.CORREN) must be set to ".

Offset and gain error compensation results are both calculated according to:

$$\text{Result} = (\text{Conversion value} + - \text{OFFSETCORR}) \cdot \text{GAINCORR}$$

The correction will introduce a latency of 13 CLK_ADC clock cycles. In free running mode this latency is introduced on the first conversion only, since its duration is always less than the propagation delay. In single conversion mode this latency is introduced for each conversion.

**Figure 33-8. ADC Timing Correction Enabled**




### 33.6.11 DMA Operation

The ADC generates the following DMA request:

- Result Conversion Ready (RESRDY): the request is set when a conversion result is available and cleared when the RESULT register is read. When the averaging operation is enabled, the DMA request is set when the averaging is completed and result is available.

### 33.6.12 Interrupts

The ADC has the following interrupt sources:

- Result Conversion Ready: RESRDY

- Window Monitor: WINMON
- Overrun: OVERRUN

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the ADC is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register. Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. This is device dependent.

Refer to *Nested Vector Interrupt Controller* for details. The user must read the INTFLAG register to determine which interrupt condition is present.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 33.6.13 Events

The ADC can generate the following output events:

- Result Ready (RESRDY): Generated when the conversion is complete and the result is available.
- Window Monitor (WINMON): Generated when the window monitor condition match.

Setting an Event Output bit in the Event Control Register (EVCTRL.xxEO=1) enables the corresponding output event. Clearing this bit disables the corresponding output event. Refer to the *Event System* chapter for details on configuring the event system.

The peripheral can take the following actions on an input event:

- Start conversion (START): Start a conversion.
- Conversion flush (FLUSH): Flush the conversion.

Setting an Event Input bit in the Event Control register (EVCTRL.xxEI=1) enables the corresponding action on input event. Clearing this bit disables the corresponding action on input event.

**Note:** If several events are connected to the ADC, the enabled action will be taken on any of the incoming events. The events must be correctly routed in the Event System.

**Related Links**

24. EVSYS – Event System

### 33.6.14 Sleep Mode Operation

The Run in Standby bit in the Control A register (CTRLA.RUNSTDBY) controls the behavior of the ADC during standby sleep mode. When CTRLA.RUNSTDBY=0, the ADC is disabled during sleep, but maintains its current configuration. When CTRLA.RUNSTDBY=1, the ADC continues to operate during sleep. Note that when CTRLA.RUNSTDBY=0, the analog blocks are powered off for the lowest power consumption. This necessitates a start-up time delay when the system returns from sleep.

When CTRLA.RUNSTDBY=1, any enabled ADC interrupt source can wake up the CPU, except the OVERRUN interrupt.. While the CPU is sleeping, ADC conversion can only be triggered by events.

### 33.6.15 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The Synchronization Ready interrupt can be used to signal when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY=1, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits are synchronized when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)

The following registers are synchronized when written:

- Control B (CTRLB)
- Software Trigger (SWTRIG)
- Window Monitor Control (WINCTRL)
- Input Control (INPUTCTRL)
- Window Upper/Lower Threshold (WINUT/WINLT)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

The following registers are synchronized when read:

- Software Trigger (SWTRIG)
- Input Control (INPUTCTRL)

Required read-synchronization is denoted by the "Read-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 33.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | | | | | RUNSTDBY | ENABLE | SWRST |
| 0x01 | REFCTRL | 7:0 | REFCOMP | | | | REFSEL[3:0] | | | |
| 0x02 | AVGCTRL | 7:0 | | ADJRES[2:0] | | | SAMPLENUM[3:0] | | | |
| 0x03 | SAMPCTRL | 7:0 | | | SAMPLEN[5:0] | | | | | |
| 0x04 | CTRLB | 7:0 | | | RESSEL[1:0] | | CORREN | FREERUN | LEFTADJ | DIFFMODE |
| | | 15:8 | | | | | | PRESCALER[2:0] | | |
| 0x06 ... 0x07 | Reserved | | | | | | | | | |
| 0x08 | WINCTRL | 7:0 | | | | | | WINMODE[2:0] | | |
| 0x09 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | SWTRIG | 7:0 | | | | | | | START | FLUSH |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | INPUTCTRL | 7:0 | | | | MUXPOS[4:0] | | | | |
| | | 15:8 | | | | MUXNEG[4:0] | | | | |
| | | 23:16 | INPUTOFFSET[3:0] | | | | INPUTSCAN[3:0] | | | |
| | | 31:24 | | | | | GAIN[3:0] | | | |
| 0x14 | EVCTRL | 7:0 | | | WINMONEO | RESRDYEO | | | SYNCEI | STARTEI |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENCLR | 7:0 | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| 0x17 | INTENSET | 7:0 | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| 0x18 | INTFLAG | 7:0 | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| 0x19 | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x1A | RESULT | 7:0 | RESULT[7:0] | | | | | | | |
| | | 15:8 | RESULT[15:8] | | | | | | | |
| 0x1C | WINLT | 7:0 | WINLT[7:0] | | | | | | | |
| | | 15:8 | WINLT[15:8] | | | | | | | |
| 0x1E ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | WINUT | 7:0 | WINUT[7:0] | | | | | | | |
| | | 15:8 | WINUT[15:8] | | | | | | | |
| 0x22 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | GAINCORR | 7:0 | GAINCORR[7:0] | | | | | | | |
| | | 15:8 | | | | | GAINCORR[11:8] | | | |
| 0x26 | OFFSETCORR | 7:0 | OFFSETCORR[7:0] | | | | | | | |
| | | 15:8 | | | | | OFFSETCORR[11:8] | | | |
| 0x28 | CALIB | 7:0 | LINEARITY_CAL[7:0] | | | | | | | |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 15:8 | | | | | | BIAS_CAL[2:0] | | |
| 0x2A | DBGCTRL | 7:0 | | | | | | | | DBGRUN |

## 33.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description.

Some registers are enable-protected, meaning they can be written only when the ADC is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

### 33.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RUNSTDBY | ENABLE | SWRST |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – RUNSTDBY** Run in Standby
This bit indicates whether the ADC will continue running in standby sleep mode or not:

| Value | Description |
|---|---|
| 0 | The ADC is halted during standby sleep mode. |
| 1 | The ADC continues normal operation during standby sleep mode. |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is a delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

| Value | Description |
|---|---|
| 0 | The ADC is disabled. |
| 1 | The ADC is enabled. |

**Bit 0 – SWRST** Software Reset
Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the ADC, except DBGCTRL, to their initial state, and the ADC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 33.8.2 Reference Control

**Name:** REFCTRL
**Offset:** 0x01
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REFCOMP | | | | REFSEL[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – REFCOMP** Reference Buffer Offset Compensation Enable
The accuracy of the gain stage can be increased by enabling the reference buffer offset compensation. This will decrease the input impedance and thus increase the start-up time of the reference.

| Value | Description |
|---|---|
| 0 | Reference buffer offset compensation is disabled. |
| 1 | Reference buffer offset compensation is enabled. |

**Bits 3:0 – REFSEL[3:0]** Reference Selection
These bits select the reference for the ADC.

**Table 33-5. Reference Selection**

| REFSEL[3:0] | Name | Description |
|---|---|---|
| 0x0 | INT1V | 1.0V voltage reference |
| 0x1 | INTVCC0 | 1/1.48 VDDANA |
| 0x2 | INTVCC1 | 1/2 VDDANA (only for VDDANA > 2.0V) |
| 0x3 | VREFA | External reference |
| 0x4 | VREFB | External reference |
| 0x5-0xF | | Reserved |

### 33.8.3 Average Control

**Name:** AVGCTRL
**Offset:** 0x02
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ADJRES[2:0] | | | SAMPLENUM[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:4 – ADJRES[2:0]** Adjusting Result / Division Coefficient
These bits define the division coefficient in 2n steps.

**Bits 3:0 – SAMPLENUM[3:0]** Number of Samples to be Collected
These bits define how many samples should be added together.The result will be available in the Result register (RESULT). Note: if the result width increases, CTRLB.RESSEL must be changed.

| SAMPLENUM[3:0] | Name | Description |
|---|---|---|
| 0x0 | 1 | 1 sample |
| 0x1 | 2 | 2 samples |
| 0x2 | 4 | 4 samples |
| 0x3 | 8 | 8 samples |
| 0x4 | 16 | 16 samples |
| 0x5 | 32 | 32 samples |
| 0x6 | 64 | 64 samples |
| 0x7 | 128 | 128 samples |
| 0x8 | 256 | 256 samples |
| 0x9 | 512 | 512 samples |
| 0xA | 1024 | 1024 samples |
| 0xB-0xF | | Reserved |

### 33.8.4 Sampling Time Control

**Name:** SAMPCTRL
**Offset:** 0x03
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | SAMPLEN[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – SAMPLEN[5:0]** Sampling Time Length
These bits control the ADC sampling time in number of half CLK_ADC cycles, depending of the prescaler value, thus controlling the ADC input impedance. Sampling time is set according to the equation:

$$\text{Sampling time} = (\text{SAMPLEN}+1) \cdot \left(\frac{\text{CLK}_{\text{ADC}}}{2}\right)$$

### 33.8.5 Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x0000
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PRESCALER[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RESSEL[1:0] | | CORREN | FREERUN | LEFTADJ | DIFFMODE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 10:8 – PRESCALER[2:0]** Prescaler Configuration
These bits define the ADC clock relative to the peripheral clock.

| PRESCALER[2:0] | Name | Description |
|---|---|---|
| 0x0 | DIV4 | Peripheral clock divided by 4 |
| 0x1 | DIV8 | Peripheral clock divided by 8 |
| 0x2 | DIV16 | Peripheral clock divided by 16 |
| 0x3 | DIV32 | Peripheral clock divided by 32 |
| 0x4 | DIV64 | Peripheral clock divided by 64 |
| 0x5 | DIV128 | Peripheral clock divided by 128 |
| 0x6 | DIV256 | Peripheral clock divided by 256 |
| 0x7 | DIV512 | Peripheral clock divided by 512 |

**Bits 5:4 – RESSEL[1:0]** Conversion Result Resolution
These bits define whether the ADC completes the conversion at 12-, 10- or 8-bit result resolution.

| RESSEL[1:0] | Name | Description |
|---|---|---|
| 0x0 | 12BIT | 12-bit result |
| 0x1 | 16BIT | For averaging mode output |
| 0x2 | 10BIT | 10-bit result |
| 0x3 | 8BIT | 8-bit result |

**Bit 3 – CORREN** Digital Correction Logic Enabled

| Value | Description |
|---|---|
| 0 | Disable the digital result correction. |
| 1 | Enable the digital result correction. The ADC conversion result in the RESULT register is then corrected for gain and offset based on the values in the GAINCAL and OFFSETCAL registers. Conversion time will be increased by X cycles according to the value in the Offset Correction Value bit group in the Offset Correction register. |

**Bit 2 – FREERUN** Free Running Mode

| Value | Description |
|---|---|
| 0 | The ADC run is single conversion mode. |
| 1 | The ADC is in free running mode and a new conversion will be initiated when a previous conversion completes. |

**Bit 1 – LEFTADJ** Left-Adjusted Result

| Value | Description |
|---|---|
| 0 | The ADC conversion result is right-adjusted in the RESULT register. |
| 1 | The ADC conversion result is left-adjusted in the RESULT register. The high byte of the 12-bit result will be present in the upper part of the result register. Writing this bit to zero (default) will right-adjust the value in the RESULT register. |

**Bit 0 – DIFFMODE** Differential Mode

| Value | Description |
|---|---|
| 0 | The ADC is running in singled-ended mode. |
| 1 | The ADC is running in differential mode. In this mode, the voltage difference between the MUXPOS and MUXNEG inputs will be converted by the ADC. |

### 33.8.6 Window Monitor Control

**Name:** WINCTRL
**Offset:** 0x08
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WINMODE[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – WINMODE[2:0]** Window Monitor Mode
These bits enable and define the window monitor mode.

| WINMODE[2:0] | Name | Description |
|---|---|---|
| 0x0 | DISABLE | No window mode (default) |
| 0x1 | MODE1 | Mode 1: RESULT > WINLT |
| 0x2 | MODE2 | Mode 2: RESULT < WINUT |
| 0x3 | MODE3 | Mode 3: WINLT < RESULT < WINUT |
| 0x4 | MODE4 | Mode 4: !(WINLT < RESULT < WINUT) |
| 0x5-0x7 | | Reserved |

### 33.8.7 Software Trigger

**Name:** SWTRIG
**Offset:** 0x0C
**Reset:** 0x00
**Property:** Write-Protected, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | START | FLUSH |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – START** ADC Start Conversion
Writing this bit to zero will have no effect.

| Value | Description |
|---|---|
| 0 | The ADC will not start a conversion. |
| 1 | The ADC will start a conversion. The bit is cleared by hardware when the conversion has started. Setting this bit when it is already set has no effect. |

**Bit 0 – FLUSH** ADC Conversion Flush
After the flush, the ADC will resume where it left off; i.e., if a conversion was pending, the ADC will start a new conversion.

Writing this bit to zero will have no effect.

| Value | Description |
|---|---|
| 0 | No flush action. |
| 1 | "Writing a '1' to this bit will flush the ADC pipeline. A flush will restart the ADC clock on the next peripheral clock edge, and all conversions in progress will be aborted and lost. This bit will be cleared after the ADC has been flushed.<br>After the flush, the ADC will resume where it left off; i.e., if a conversion was pending, the ADC will start a new conversion. |

### 33.8.8 Input Control

**Name:** INPUTCTRL
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Write-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | GAIN[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | INPUTOFFSET[3:0] | | | | INPUTSCAN[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | MUXNEG[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | MUXPOS[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 27:24 – GAIN[3:0]** Gain Factor Selection
These bits set the gain factor of the ADC gain stage.

| GAIN[3:0] | Name | Description |
|---|---|---|
| 0x0 | 1X | 1x |
| 0x1 | 2X | 2x |
| 0x2 | 4X | 4x |
| 0x3 | 8X | 8x |
| 0x4 | 16X | 16x |
| 0x5-0xE | | Reserved |
| 0xF | DIV2 | 1/2x |

**Bits 23:20 – INPUTOFFSET[3:0]** Positive Mux Setting Offset
The pin scan is enabled when INPUTSCAN != 0. Writing these bits to a value other than zero causes the first conversion triggered to be converted using a positive input equal to MUXPOS + INPUTOFFSET. Setting this register to zero causes the first conversion to use a positive input equal to MUXPOS.

After a conversion, the INPUTOFFSET register will be incremented by one, causing the next conversion to be done with the positive input equal to MUXPOS + INPUTOFFSET. The sum of MUXPOS and INPUTOFFSET gives the input that is actually converted.

**Bits 19:16 – INPUTSCAN[3:0]** Number of Input Channels Included in Scan
This register gives the number of input sources included in the pin scan. The number of input sources included is INPUTSCAN + 1. The input channels included are in the range from MUXPOS + INPUTOFFSET to MUXPOS + INPUTOFFSET + INPUTSCAN.

The range of the scan mode must not exceed the number of input channels available on the device.

**Bits 12:8 – MUXNEG[4:0]** Negative Mux Input Selection
These bits define the Mux selection for the negative ADC input. selections.

| Value | Name | Description |
|---|---|---|
| 0x00 | PIN0 | ADC AIN0 pin |
| 0x01 | PIN1 | ADC AIN1 pin |
| 0x02 | PIN2 | ADC AIN2 pin |
| 0x03 | PIN3 | ADC AIN3 pin |
| 0x04 | PIN4 | ADC AIN4 pin |
| 0x05 | PIN5 | ADC AIN5 pin |
| 0x06 | PIN6 | ADC AIN6 pin |
| 0x07 | PIN7 | ADC AIN7 pin |
| 0x08-0x17 | Reserved | |
| 0x18 | GND | Internal ground |
| 0x19 | IOGND | I/O ground |
| 0x1A-0x1F | Reserved<br>Note: 1. Only available in SAM R21G. | |

**Bits 4:0 – MUXPOS[4:0]** Positive Mux Input Selection
These bits define the Mux selection for the positive ADC input. The following table shows the possible input selections. If the internal bandgap voltage or temperature sensor input channel is selected, then the Sampling Time Length bit group in the SamplingControl register must be written.

| MUXPOS[4:0] | Group configuration | Description |
|---|---|---|
| 0x00 | PIN0 | ADC AIN0 pin |
| 0x01 | PIN1 | ADC AIN1 pin |
| 0x02 | PIN2 | ADC AIN2 pin |
| 0x03 | PIN3 | ADC AIN3 pin |
| 0x04 | PIN4 | ADC AIN4 pin |
| 0x05 | PIN5 | ADC AIN5 pin |
| 0x06 | PIN6 | ADC AIN6 pin |
| 0x07 | PIN7 | ADC AIN7 pin |

| MUXPOS[4:0] | Group configuration | Description |
|---|---|---|
| 0x08 | PIN8 | ADC AIN8 pin |
| 0x09 | PIN9 | ADC AIN9 pin |
| 0x0A | PIN10 | ADC AIN10 pin |
| 0x0B | PIN11 | ADC AIN11 pin |
| 0x0C | PIN12 | ADC AIN12 pin |
| 0x0D | PIN13 | ADC AIN13 pin |
| 0x0E | PIN14 | ADC AIN14 pin |
| 0x0F | PIN15 | ADC AIN15 pin |
| 0x10 | PIN16 | ADC AIN16 pin |
| 0x11 | PIN17 | ADC AIN17 pin |
| 0x12 | PIN18 | ADC AIN18 pin |
| 0x13 | PIN19 | ADC AIN19 pin |
| 0x14-0x17 | | Reserved |
| 0x19 | BANDGAP | Bandgap voltage |
| 0x1A | SCALEDCOREVCC | 1/4 scaled core supply |
| 0x1B | SCALEDIOVCC | 1/4 scaled I/O supply |
| 0x1C | DAC | DAC output |
| 0x1D-0x1F | | Reserved |

### 33.8.9 Event Control

**Name:** EVCTRL
**Offset:** 0x14
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | WINMONEO | RESRDYEO | | | SYNCEI | STARTEI |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 5 – WINMONEO** Window Monitor Event Out
This bit indicates whether the Window Monitor event output is enabled or not and an output event will be generated when the window monitor detects something.

| Value | Description |
|---|---|
| 0 | Window Monitor event output is disabled and an event will not be generated. |
| 1 | Window Monitor event output is enabled and an event will be generated. |

**Bit 4 – RESRDYEO** Result Ready Event Out
This bit indicates whether the Result Ready event output is enabled or not and an output event will be generated when the conversion result is available.

| Value | Description |
|---|---|
| 0 | Result Ready event output is disabled and an event will not be generated. |
| 1 | Result Ready event output is enabled and an event will be generated. |

**Bit 1 – SYNCEI** Synchronization Event In

| Value | Description |
|---|---|
| 0 | A flush and new conversion will not be triggered on any incoming event. |
| 1 | A flush and new conversion will be triggered on any incoming event. |

**Bit 0 – STARTEI** Start Conversion Event In

| Value | Description |
|---|---|
| 0 | A new conversion will not be triggered on any incoming event. |
| 1 | A new conversion will be triggered on any incoming event. |

### 33.8.10 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x16
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set. |

**Bit 2 – WINMON** Window Monitor Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Window Monitor Interrupt Enable bit and the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The window monitor interrupt is disabled. |
| 1 | The window monitor interrupt is enabled, and an interrupt request will be generated when the Window Monitor interrupt flag is set. |

**Bit 1 – OVERRUN** Overrun Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Overrun Interrupt Enable bit and the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Overrun interrupt is disabled. |
| 1 | The Overrun interrupt is enabled, and an interrupt request will be generated when the Overrun interrupt flag is set. |

**Bit 0 – RESRDY** Result Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will clear the Result Ready Interrupt Enable bit and the corresponding interrupt request.

| Value | Description |
|---|---|
| 0 | The Result Ready interrupt is disabled. |
| 1 | The Result Ready interrupt is enabled, and an interrupt request will be generated when the Result Ready interrupt flag is set. |

### 33.8.11 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x17
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit, which enables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 2 – WINMON** Window Monitor Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Window Monitor Interrupt bit and enable the Window Monitor interrupt.

| Value | Description |
|---|---|
| 0 | The Window Monitor interrupt is disabled. |
| 1 | The Window Monitor interrupt is enabled. |

**Bit 1 – OVERRUN** Overrun Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Overrun Interrupt bit and enable the Overrun interrupt.

| Value | Description |
|---|---|
| 0 | The Overrun interrupt is disabled. |
| 1 | The Overrun interrupt is enabled. |

**Bit 0 – RESRDY** Result Ready Interrupt Enable
Writing a zero to this bit has no effect.

Writing a one to this bit will set the Result Ready Interrupt bit and enable the Result Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Result Ready interrupt is disabled. |
| 1 | The Result Ready interrupt is enabled. |

### 33.8.12 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SYNCRDY | WINMON | OVERRUN | RESRDY |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – SYNCRDY** Synchronization Ready
This flag is cleared by writing a one to the flag.

This flag is set on a one-to-zero transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by an enable or software reset, and will generate an interrupt request if INTENCLR/SET.SYNCRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Synchronization Ready interrupt flag.

**Bit 2 – WINMON** Window Monitor
This flag is cleared by writing a one to the flag or by reading the RESULT register.

This flag is set on the next GCLK_ADC cycle after a match with the window monitor condition, and an interrupt request will be generated if INTENCLR/SET.WINMON is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Window Monitor interrupt flag.

**Bit 1 – OVERRUN** Overrun
This flag is cleared by writing a one to the flag.

This flag is set if RESULT is written before the previous value has been read by CPU, and an interrupt request will be generated if INTENCLR/SET.OVERRUN is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Overrun interrupt flag.

**Bit 0 – RESRDY** Result Ready
This flag is cleared by writing a one to the flag or by reading the RESULT register.

This flag is set when the conversion result is available, and an interrupt will be generated if INTENCLR/SET.RESRDY is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the Result Ready interrupt flag.

### 33.8.13 Status

**Name:** STATUS
**Offset:** 0x19
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 33.8.14 Result

**Name:** RESULT
**Offset:** 0x1A
**Reset:** 0x0000
**Property:** Read-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RESULT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RESULT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RESULT[15:0]** Result Conversion Value
These bits will hold up to a 16-bit ADC result, depending on the configuration.

In single conversion mode without averaging, the ADC conversion will produce a 12-bit result, which can be left- or right-shifted, depending on the setting of CTRLB.LEFTADJ.

If the result is left-adjusted (CTRLB.LEFTADJ), the high byte of the result will be in bit position [15:8], while the remaining 4 bits of the result will be placed in bit locations [7:4]. This can be used only if an 8-bit result is required; i.e., one can read only the high byte of the entire 16-bit register.

If the result is not left-adjusted (CTRLB.LEFTADJ) and no oversampling is used, the result will be available in bit locations [11:0], and the result is then 12 bits long.

If oversampling is used, the result will be located in bit locations [15:0], depending on the settings of the Average Control register (AVGCTRL).

### 33.8.15 Window Monitor Lower Threshold

**Name:** WINLT
**Offset:** 0x1C
**Reset:** 0x0000
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WINLT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WINLT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – WINLT[15:0]** Window Lower Threshold
If the window monitor is enabled, these bits define the lower threshold value.

### 33.8.16 Window Monitor Upper Threshold

**Name:** WINUT
**Offset:** 0x20
**Reset:** 0x0000
**Property:** Write-Protected, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WINUT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WINUT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – WINUT[15:0]** Window Upper Threshold
If the window monitor is enabled, these bits define the upper threshold value.

### 33.8.17 Gain Correction

**Name:** GAINCORR
**Offset:** 0x24
**Reset:** 0x0000
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | GAINCORR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GAINCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – GAINCORR[11:0]** Gain Correction Value
If the CTRLB.CORREN bit is one, these bits define how the ADC conversion result is compensated for gain error before being written to the result register. The gain-correction is a fractional value, a 1-bit integer plusan 11-bit fraction, and therefore 1/2 <= GAINCORR < 2. GAINCORR values range from 0.10000000000 to 1.11111111111.

### 33.8.18 Offset Correction

**Name:** OFFSETCORR
**Offset:** 0x26
**Reset:** 0x0000
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OFFSETCORR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFFSETCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – OFFSETCORR[11:0]** Offset Correction Value
If the CTRLB.CORREN bit is one, these bits define how the ADC conversion result is compensated for offset error before being written to the Result register. This OFFSETCORR value is in two's complement format.

### 33.8.19 Calibration

**Name:** CALIB
**Offset:** 0x28
**Reset:** 0x0000
**Property:** Write-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | BIAS_CAL[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LINEARITY_CAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 10:8 – BIAS_CAL[2:0]** Bias Calibration Value
This value from production test must be loaded from the NVM software calibration area into the CALIB register by software to achieve the specified accuracy.

The value must be copied only, and must not be changed.

**Bits 7:0 – LINEARITY_CAL[7:0]** Linearity Calibration Value
This value from production test must be loaded from the NVM software calibration area into the CALIB register by software to achieve the specified accuracy.

The value must be copied only, and must not be changed.

### 33.8.20 Debug Control

**Name:** DBGCTRL
**Offset:** 0x2A
**Reset:** 0x00
**Property:** Write-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGRUN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGRUN** Debug Run
This bit can be changed only while the ADC is disabled.

This bit should be written only while a conversion is not ongoing.

| Value | Description |
|---|---|
| 0 | The ADC is halted during debug mode. |
| 1 | The ADC continues normal operation during debug mode. |

# 34. AC – Analog Comparators

## 34.1 Overview

The Analog Comparator (AC) supports multiple individual comparators. Each comparator (COMP) compares the voltage levels on two inputs, and provides a digital output based on this comparison. Each comparator may be configured to generate interrupt requests and/or peripheral events upon several different combinations of input change.

Hysteresis can be adjusted to achieve the optimal operation for each application.

The input selection includes four shared analog port pins and several internal signals. Each comparator output state can also be output on a pin for use by external devices.

The comparators are grouped in pairs on each port. The AC peripheral implements one or two pairs of comparators . These are called Comparator 0 (COMP0) and Comparator 1 (COMP1) for the first pair and Comparator 2 (COMP2) and Comparator 3 (COMP3) for the second pair. They have identical behaviors, but separate control registers. Each pair can be set in window mode to compare a signal to a voltage range instead of a single voltage level.

## 34.2 Features

- Four individual comparators
- Analog comparator outputs available on pins
  - Asynchronous or synchronous
- Flexible input selection:
  - Four pins selectable for positive or negative inputs
  - Ground (for zero crossing)
  - Bandgap reference voltage
  - 64-level programmable VDD scaler per comparator
  - DAC (if available)
- Interrupt generation on:
  - Rising or falling edge
  - Toggle
  - End of comparison
- Window function interrupt generation on:
  - Signal above window
  - Signal inside window
  - Signal below window
  - Signal outside window
- Event generation on:
  - Comparator output
  - Window function inside/outside window
- Optional digital filter on comparator output

© 2018 Microchip Technology Inc.

## 34.3 Block Diagram

**Figure 34-1. Analog Comparator Block Diagram (First Pair)**

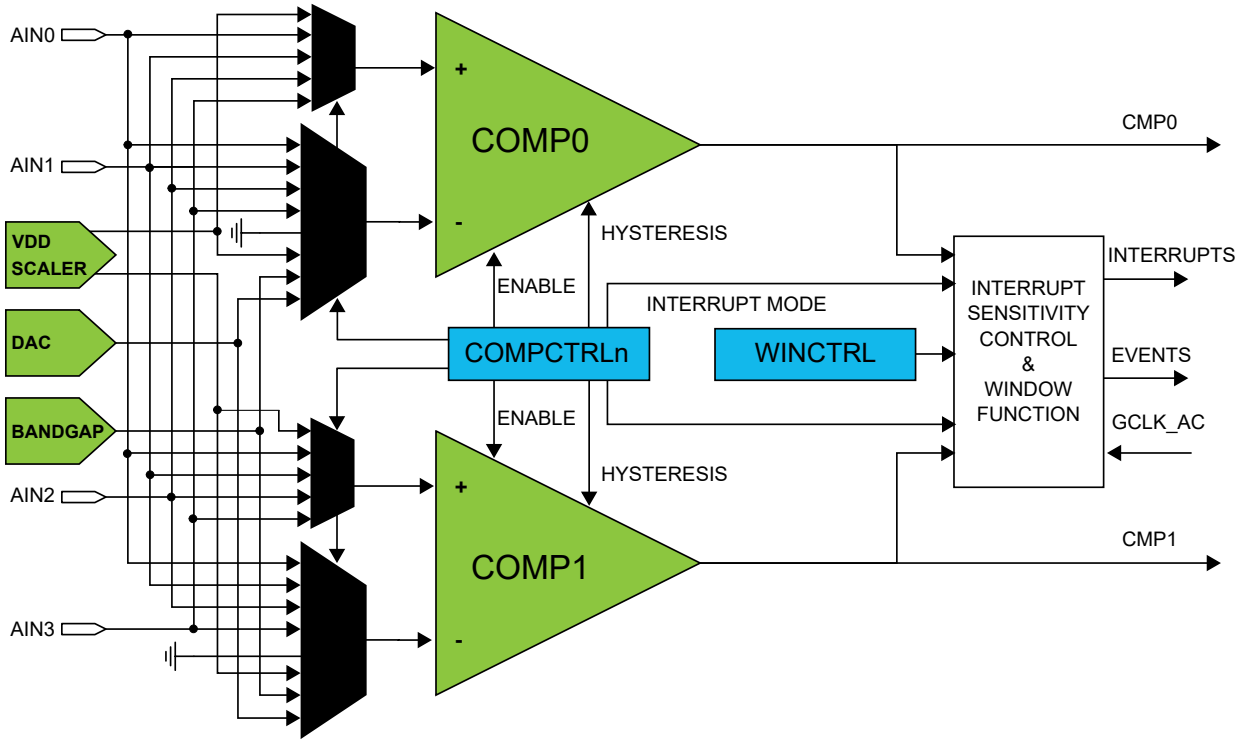
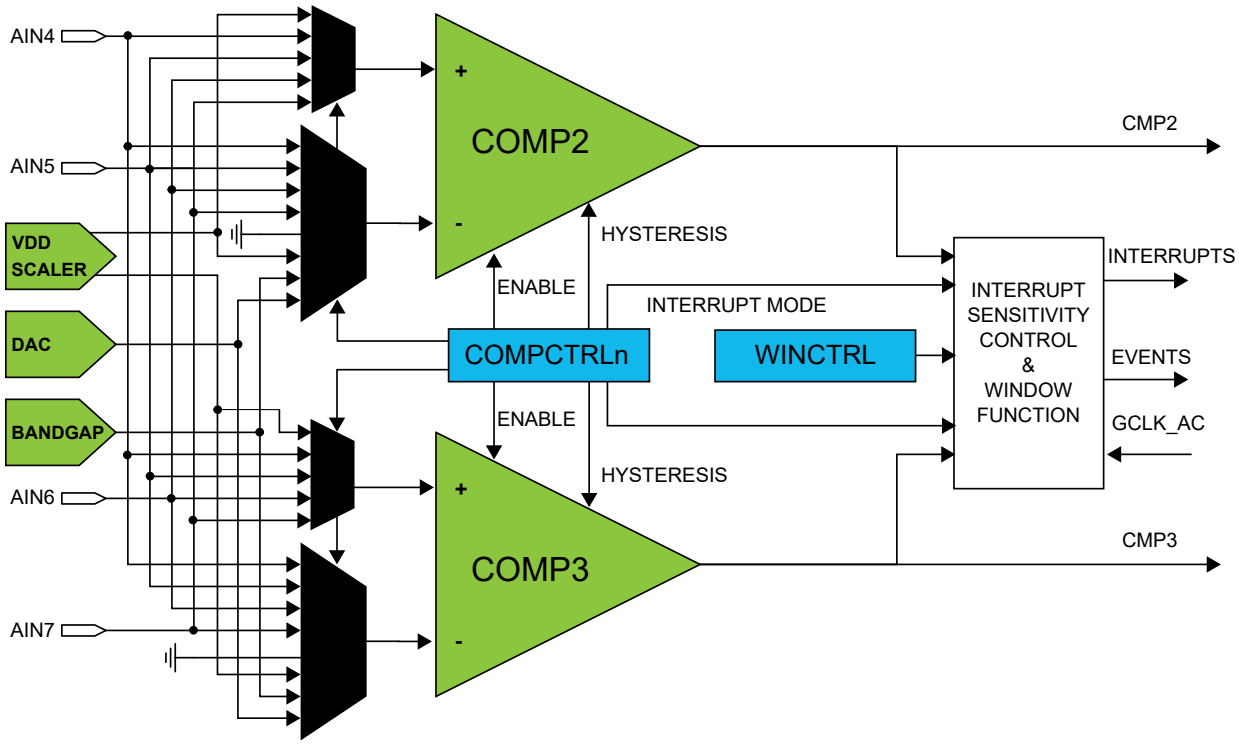



**Figure 34-2. Analog Comparator Block Diagram (Second Pair)**



## 34.4 Signal Description

| Signal | Description | Type |
|---|---|---|
| AIN[7..0] | Analog input | Comparator inputs |
| CMP[2..0] | Digital output | Comparator outputs |

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations

## 34.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 34.5.1 I/O Lines

Using the AC's I/O lines requires the I/O pins to be configured. Refer to *PORT - I/O Pin Controller* for details.

**Related Links**

23. PORT - I/O Pin Controller

### 34.5.2 Power Management

The AC will continue to operate in any sleep mode where the selected source clock is running. The AC's interrupts can be used to wake up the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 34.5.3 Clocks

The AC bus clock (CLK_AC_APB) can be enabled and disabled in the Main Clock module, MCLK (see *MCLK - Main Clock*, and the default state of CLK_AC_APB can be found in *Peripheral Clock Masking*.

Two generic clocks (GCLK_AC_DIG and GCLK_AC_ANA) are used by the AC. The digital clock (GCLK_AC_DIG) is required to provide the sampling rate for the comparators, while the analog clock (GCLK_AC_ANA) is required for low voltage operation (VDDANA < 2.5V) to ensure that the resistance of the analog input multiplexors remains low. These clocks must be configured and enabled in the Generic Clock Controller before using the peripheral.

This generic clock is asynchronous to the bus clock (CLK_AC_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to 34.6.15 Synchronization for further details.

**Related Links**

16. PM – Power Manager

### 34.5.4 DMA

Not applicable.

### 34.5.5 Interrupts

The interrupt request lines are connected to the interrupt controller. Using the AC interrupts requires the interrupt controller to be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 34.5.6 Events

The events are connected to the Event System. Refer to *EVSYS – Event System* for details on how to configure the Event System.

**Related Links**

24. EVSYS – Event System

### 34.5.7 Debug Operation

When the CPU is halted in Debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging.

### 34.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except for the following registers:

- Control B register (CTRLB)
- Interrupt Flag register (INTFLAG)

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger.

**Related Links**

11.6 PAC - Peripheral Access Controller

### 34.5.9 Analog Connections

Each comparator has up to four I/O pins that can be used as analog inputs. Each pair of comparators shares the same four pins. These pins must be configured for analog operation before using them as comparator inputs.

Any internal reference source, such as a bandgap voltage reference, or DAC must be configured and enabled prior to its use as a comparator input.

## 34.6 Functional Description

### 34.6.1 Principle of Operation

Each comparator has one positive input and one negative input. Each positive input may be chosen from a selection of analog input pins. Each negative input may be chosen from a selection of both analog input pins and internal inputs, such as a bandgap voltage reference.

The digital output from the comparator is '1' when the difference between the positive and the negative input voltage is positive, and '0' otherwise.

The individual comparators can be used independently (normal mode) or paired to form a window comparison (window mode).

### 34.6.2 Basic Operation

#### 34.6.2.1 Initialization

Before enabling the AC, the input and output events must be configured in the Event Control register (EVCTRL). These settings cannot be changed while the AC is enabled.

#### 34.6.2.2 Enabling, Disabling and Resetting

The AC is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The AC is disabled writing a '0' to CTRLA.ENABLE.

The AC is reset by writing a '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the AC will be reset to their initial state, and the AC will be disabled. Refer to *CTRLA* for details.

The individual comparators must be also enabled by writing a '1' to the Enable bit in the Comparator x Control registers (COMPCTRLx.ENABLE). However, when the AC is disabled, this will also disable the individual comparators, but will not clear their COMPCTRLx.ENABLE bits.

**Related Links**

34.8.1 CTRLA

#### 34.6.2.3 Comparator Configuration

Each individual comparator must be configured by its respective Comparator Control register (COMPCTRLx) before that comparator is enabled. These settings cannot be changed while the comparator is enabled.

- Select the desired measurement mode with COMPCTRLx.SINGLE. See Starting a Comparison for more details.
- Select the hysteresis with the COMPCTRLx.HYST bit. See Input Hysteresis for more details.
- Select the comparator speed versus power with COMPCTRLx.SPEED. See Propagation Delay vs. Power Consumption for more details.
- Select the interrupt source with COMPCTRLx.INTSEL.
- Select the positive and negative input sources with the COMPCTRLx.MUXPOS and COMPCTRLx.MUXNEG bits. See Selecting Comparator Inputs for more details.
- Select the filtering option with COMPCTRLx.FLEN.
- Select standby operation with Run in Standby bit (COMPCTRLx.RUNSTDBY).

The individual comparators are enabled by writing a '1' to the Enable bit in the Comparator x Control registers (COMPCTRLx.ENABLE). The individual comparators are disabled by writing a '0' to COMPCTRLx.ENABLE. Writing a '0' to CTRLA.ENABLE will also disable all the comparators, but will not clear their COMPCTRLx.ENABLE bits.

#### 34.6.2.4 Starting a Comparison

Each comparator channel can be in one of two different measurement modes, determined by the Single bit in the Comparator x Control register (COMPCTRLx.SINGLE):

- Continuous measurement
- Single-shot

After being enabled, a start-up delay is required before the result of the comparison is ready. This start-up time is measured automatically to account for environmental changes, such as temperature or voltage

supply level, and is specified in *Electrical Characteristics*. During the start-up time, the COMP output is not available.

The comparator can be configured to generate interrupts when the output toggles, when the output changes from '0' to '1' (rising edge), when the output changes from '1' to '0' (falling edge) or at the end of the comparison. An end-of-comparison interrupt can be used with the single-shot mode to chain further events in the system, regardless of the state of the comparator outputs. The interrupt mode is set by the Interrupt Selection bit group in the Comparator Control register (COMPCTRLx.INTSEL). Events are generated using the comparator output state, regardless of whether the interrupt is enabled or not.

**Related Links**

37. Electrical Characteristics

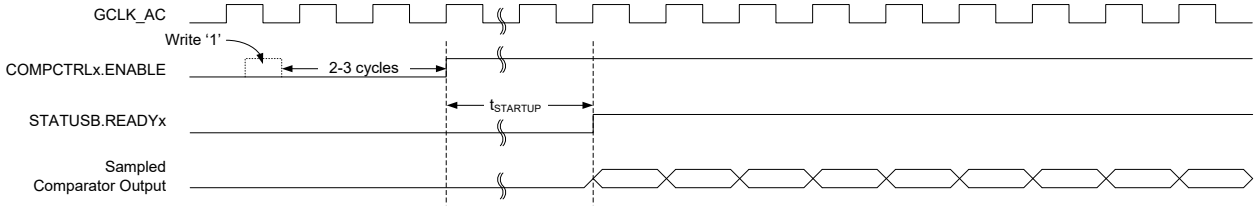
#### 34.6.2.4.1 Continuous Measurement

Continuous measurement is selected by writing COMPCTRLx.SINGLE to zero. In continuous mode, the comparator is continuously enabled and performing comparisons. This ensures that the result of the latest comparison is always available in the Current State bit in the Status A register (STATUSA.STATEx).

After the start-up time has passed, a comparison is done and STATUSA is updated. The Comparator x Ready bit in the Status B register (STATUSB.READYx) is set, and the appropriate peripheral events and interrupts are also generated. New comparisons are performed continuously until the COMPCTRLx.ENABLE bit is written to zero. The start-up time applies only to the first comparison.

In continuous operation, edge detection of the comparator output for interrupts is done by comparing the current and previous sample. The sampling rate is the CLK_AC_DIG frequency. An example of continuous measurement is shown in the next figure.

**Figure 34-3. Continuous Measurement Example**




For low-power operation, comparisons can be performed during sleep modes without a clock. The comparator is enabled continuously, and changes of the comparator state are detected asynchronously. When a toggle occurs, the Power Manager will start CLK_AC_DIG to register the appropriate peripheral events and interrupts. The CLK_AC_DIG clock is then disabled again automatically, unless configured to wake up the system from sleep.

**Related Links**

37. Electrical Characteristics

#### 34.6.2.4.2 Single-Shot

Single-shot operation is selected by writing COMPCTRLx.SINGLE to '1'. During single-shot operation, the comparator is normally idle. The user starts a single comparison by writing '1' to the respective Start Comparison bit in the write-only Control B register (CTRLB.STARTx). The comparator is enabled, and after the start-up time has passed, a single comparison is done and STATUSA is updated. Appropriate peripheral events and interrupts are also generated. No new comparisons will be performed.

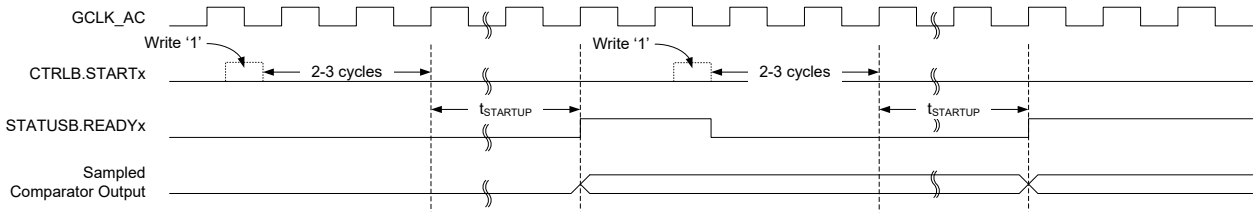
Writing '1' to CTRLB.STARTx also clears the Comparator x Ready bit in the Status B register (STATUSB.READYx). STATUSB.READYx is set automatically by hardware when the single comparison has completed.

To remove the need for polling, an additional means of starting the comparison is also available. A read of the Status C register (STATUSC) will start a comparison on all comparators currently configured for single-shot operation. The read will stall the bus until all enabled comparators are ready. If a comparator is already busy with a comparison, the read will stall until the current comparison is compete, and a new comparison will not be started.

A single-shot measurement can also be triggered by the Event System. Setting the Comparator x Event Input bit in the Event Control Register (EVCTRL.COMPEIx) enables triggering on incoming peripheral events. Each comparator can be triggered independently by separate events. Event-triggered operation is similar to user-triggered operation; the difference is that a peripheral event from another hardware module causes the hardware to automatically start the comparison and clear STATUSB.READYx.

To detect an edge of the comparator output in single-shot operation for the purpose of interrupts, the result of the current measurement is compared with the result of the previous measurement (one sampling period earlier). An example of single-shot operation is shown in the figure below.

**Figure 34-4. Single-Shot Example**




For low-power operation, event-triggered measurements can be performed during sleep modes. When the event occurs, the Power Manager will start CLK_AC_DIG. The comparator is enabled, and after the startup time has passed, a comparison is done and appropriate peripheral events and interrupts are also generated. The comparator and CLK_AC_DIG are then disabled again automatically, unless configured to wake up the system from sleep.

**Related Links**

37. Electrical Characteristics

### 34.6.3 Selecting Comparator Inputs

Each comparator has one positive and one negative input. The positive input is one of the external input pins (AINx). The negative input can be fed either from an external input pin (AINx) or from one of the several internal reference voltage sources common to all comparators. The user selects the input source as follows:

- The positive input is selected by the Positive Input MUX Select bit group in the Comparator Control register (COMPCTRLx.MUXPOS)
- The negative input is selected by the Negative Input MUX Select bit group in the Comparator Control register (COMPCTRLx.MUXNEG)

In the case of using an external I/O pin, the selected pin must be configured for analog use in the PORT Controller by disabling the digital input and output. The switching of the analog input multiplexers is controlled to minimize crosstalk between the channels. The input selection must be changed only while the individual comparator is disabled.

**Note:** For internal use of the comparison results by the CCL, this bit must be 0x1 or 0x2.

### 34.6.4 Window Operation

Each comparator pair can be configured to work together in window mode. In this mode, a voltage range is defined, and the comparators give information about whether an input signal is within this range or not.

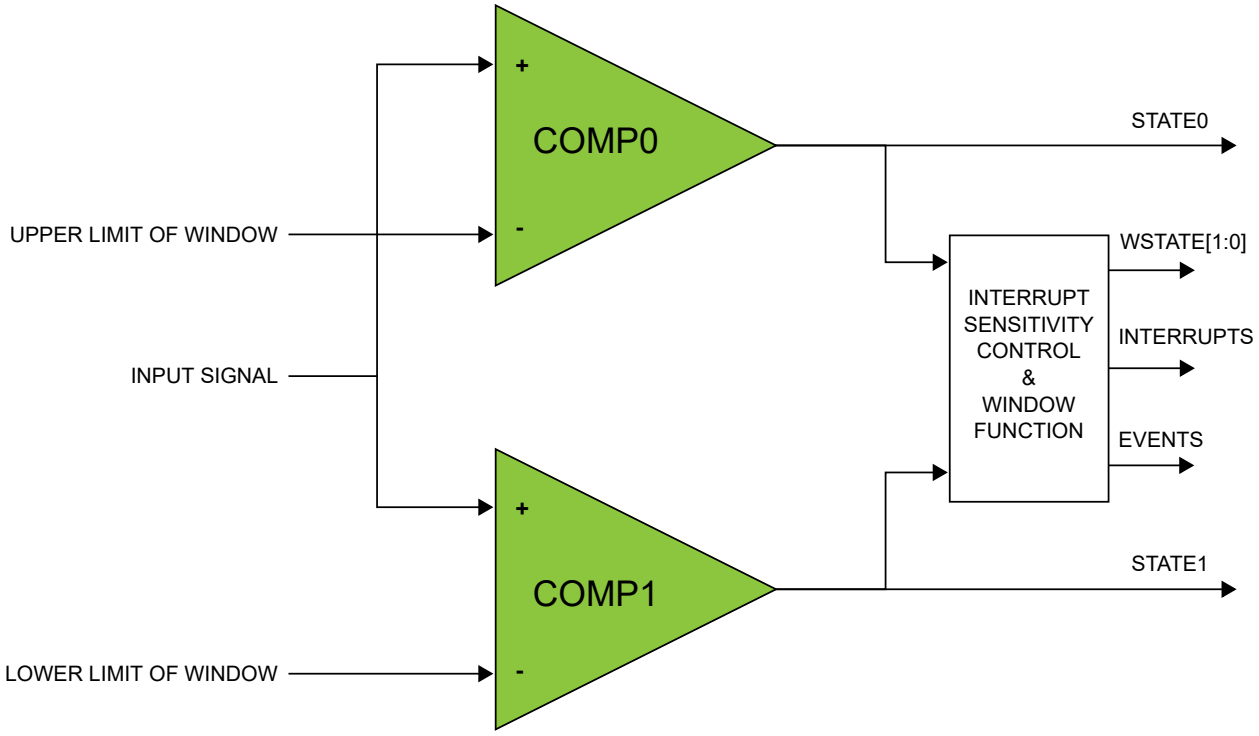
Window mode is enabled by the Window Enable x bit in the Window Control register (WINCTRL.WENx). Both comparators in a pair must have the same measurement mode setting in their respective Comparator Control Registers (COMPCTRLx.SINGLE).

To physically configure the pair of comparators for window mode, the same I/O pin must be chosen as positive input for each comparator, providing a shared input signal. The negative inputs define the range for the window. In Figure 34-5, COMP0 defines the upper limit and COMP1 defines the lower limit of the window, as shown but the window will also work in the opposite configuration with COMP0 lower and COMP1 higher. The current state of the window function is available in the Window x State bit group of the Status register (STATUS.WSTATEx).

Window mode can be configured to generate interrupts when the input voltage changes to below the window, when the input voltage changes to above the window, when the input voltage changes into the window or when the input voltage changes outside the window. The interrupt selections are set by the Window Interrupt Selection bit field in the Window Control register (WINCTRL.WINTSEL). Events are generated using the inside/outside state of the window, regardless of whether the interrupt is enabled or not. Note that the individual comparator outputs, interrupts and events continue to function normally during window mode.

When the comparators are configured for window mode and single-shot mode, measurements are performed simultaneously on both comparators. Writing '1' to either Start Comparison bit in the Control B register (CTRLB.STARTx) will start a measurement. Likewise either peripheral event can start a measurement.

**Figure 34-5. Comparators in Window Mode**




### 34.6.5 Voltage Doubler

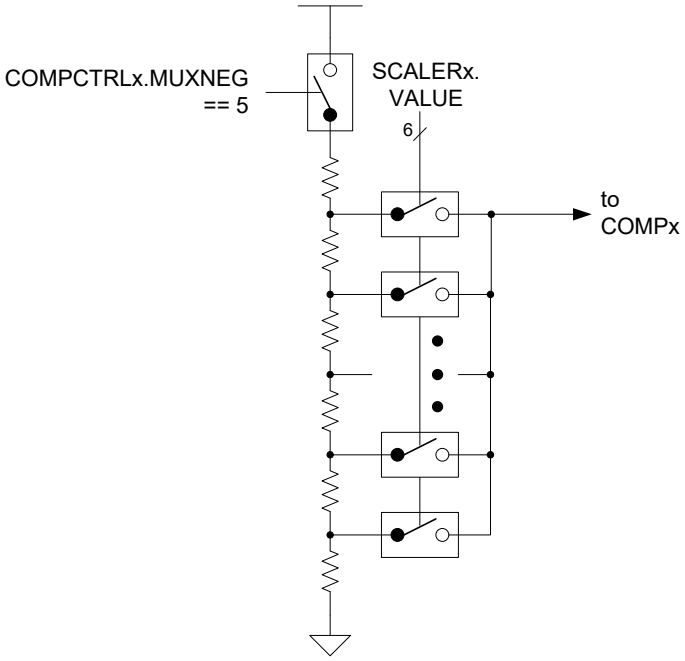
The AC contains a voltage doubler that can reduce the resistance of the analog multiplexors when the supply voltage is below 2.5V. The voltage doubler is normally switched on/off automatically based on the supply level. When enabling the comparators, additional start-up time is required for the voltage doubler

to settle. If the supply voltage is guaranteed to be above 2.5V, the voltage doubler can be disabled by writing the Low-Power Mux bit in the Control A register (CTRLA.LPMUX) to one. Disabling the voltage doubler saves power and reduces the start-up time.

### 34.6.6 $V_{DDANA}$ Scaler

The $V_{DDANA}$ scaler generates a reference voltage that is a fraction of the device's supply voltage, with 64 levels. One independent voltage channel is dedicated for each comparator. The scaler of a comparator is enabled when the Negative Input Mux bit field in the respective Comparator Control register (COMPCTRLx.MUXNEG) is set to 0x5 and the comparator is enabled. The voltage of each channel is selected by the Value bit field in the Scaler x registers (SCALERx.VALUE).

**Figure 34-6. $V_{DDANA}$ Scaler**




### 34.6.7 Input Hysteresis

Application software can selectively enable/disable hysteresis for the comparison. Applying hysteresis will help prevent constant toggling of the output, which can be caused by noise when the input signals are close to each other.

Hysteresis is enabled for each comparator individually by the Hysteresis Enable bit in the Comparator x Control register (COMPCTRLx.HYSTEN). Hysteresis is available only in continuous mode (COMPCTRLx.SINGLE=0).

### 34.6.8 Propagation Delay vs. Power Consumption

It is possible to trade off comparison speed for power efficiency to get the shortest possible propagation delay or the lowest power consumption. The speed setting is configured for each comparator individually by the Speed bit group in the Comparator x Control register (COMPCTRLx.SPEED). The Speed bits select the amount of bias current provided to the comparator, and as such will also affect the start-up time.

### 34.6.9 Filtering

The output of the comparators can be filtered digitally to reduce noise. The filtering is determined by the Filter Length bits in the Comparator Control x register (COMPCTRLx.FLEN), and is independent for each

comparator. Filtering is selectable from none, 3-bit majority (N=3) or 5-bit majority (N=5) functions. Any change in the comparator output is considered valid only if N/2+1 out of the last N samples agree. The filter sampling rate is the GCLK_AC frequency.

Note that filtering creates an additional delay of N-1 sampling cycles from when a comparison is started until the comparator output is validated. For continuous mode, the first valid output will occur when the required number of filter samples is taken. Subsequent outputs will be generated every cycle based on the current sample plus the previous N-1 samples, as shown in Figure 34-7. For single-shot mode, the comparison completes after the Nth filter sample, as shown in Figure 34-8.

**Figure 34-7. Continuous Mode Filtering**

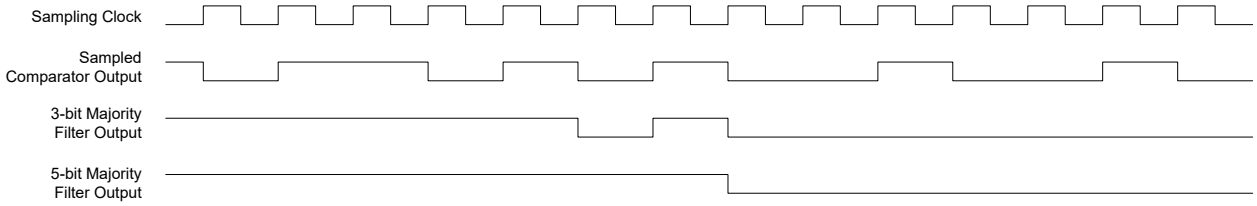
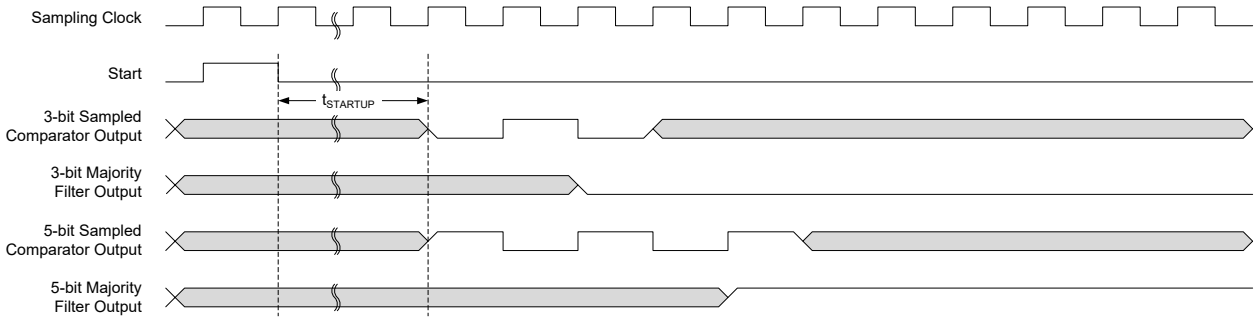



**Figure 34-8. Single-Shot Filtering**




During sleep modes, filtering is supported only for single-shot measurements. Filtering must be disabled if continuous measurements will be done during sleep modes, or the resulting interrupt/event may be generated incorrectly.

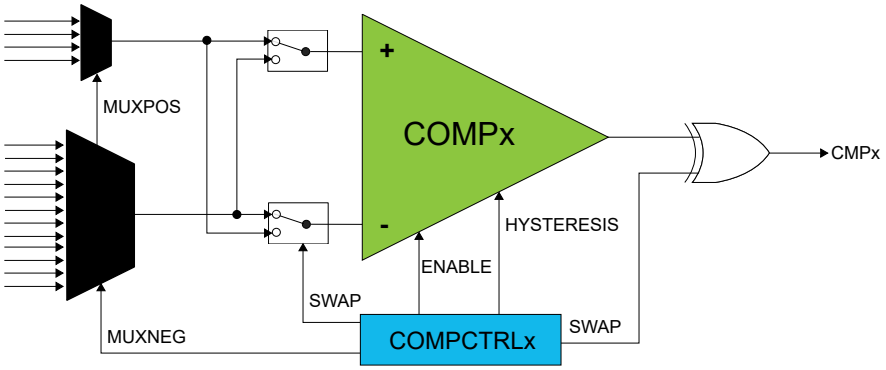
### 34.6.10 Comparator Output

The output of each comparator can be routed to an I/O pin by setting the Output bit group in the Comparator Control x register (COMPCTRLx.OUT). This allows the comparator to be used by external circuitry. Either the raw, non-synchronized output of the comparator or the CLK_AC-synchronized version, including filtering, can be used as the I/O signal source. The output appears on the corresponding CMP[x] pin.

### 34.6.11 Offset Compensation

The Swap bit in the Comparator Control registers (COMPCTRLx.SWAP) controls switching of the input signals to a comparator's positive and negative terminals. When the comparator terminals are swapped, the output signal from the comparator is also inverted, as shown in Figure 34-9. This allows the user to measure or compensate for the comparator input offset voltage. As part of the input selection, COMPCTRLx.SWAP can be changed only while the comparator is disabled.

**Figure 34-9. Input Swapping for Offset Compensation**




### 34.6.12 Interrupts

The AC has the following interrupt sources:

- Comparator (COMP0, COMP1, COMP2, COMP3): Indicates a change in comparator status.
- Window (WIN0, WIN1): Indicates a change in the window status.

Comparator interrupts are generated based on the conditions selected by the Interrupt Selection bit group in the Comparator Control registers (COMPCTRLx.INTSEL). Window interrupts are generated based on the conditions selected by the Window Interrupt Selection bit group in the Window Control register (WINCTRL.WINTSELx[1:0]).

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the AC is reset. See INFLAG register for details on how to clear interrupt flags. All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 34.6.13 Events

The AC can generate the following output events:

- Comparator (COMP0, COMP1, COMP2, COMP3): Generated as a copy of the comparator status
- Window (WIN0, WIN1): Generated as a copy of the window inside/outside status

Output events must be enabled to be generated. Writing a one to an Event Output bit in the Event Control register (EVCTRL.COMPEOx) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. The events must be correctly routed in the Event System.

The AC can take the following action on an input event:

- Single-shot measurement
- Single-shot measurement in window mode

Writing a one to an Event Input bit into the Event Control register (EVCTRL.COMPEIx) enables the corresponding action on input event. Writing a zero to this bit disables the corresponding action on input event. Note that if several events are connected to the AC, the enabled action will be taken on any of the incoming events. Refer to the Event System chapter for details on configuring the event system.

When EVCTRL.COMPEIx is one, the event will start a comparison on COMPx after the start-up time delay. In normal mode, each comparator responds to its corresponding input event independently. For a pair of comparators in window mode, either comparator event will trigger a comparison on both comparators simultaneously.

### 34.6.14 Sleep Mode Operation

The Run in Standby bits in the Comparator x Control registers (COMPCTRLx.RUNSTDBY) control the behavior of the AC during standby sleep mode. Each RUNSTDBY bit controls one comparator. When the bit is zero, the comparator is disabled during sleep, but maintains its current configuration. When the bit is one, the comparator continues to operate during sleep. Note that when RUNSTDBY is zero, the analog blocks are powered off for the lowest power consumption. This necessitates a start-up time delay when the system returns from sleep.

When RUNSTDBY is one, any enabled AC interrupt source can wake up the CPU. While the CPU is sleeping, single-shot comparisons are only triggerable by events. The AC can also be used during sleep modes where the clock used by the AC is disabled, provided that the AC is still powered (not in shutdown). In this case, the behavior is slightly different and depends on the measurement mode.

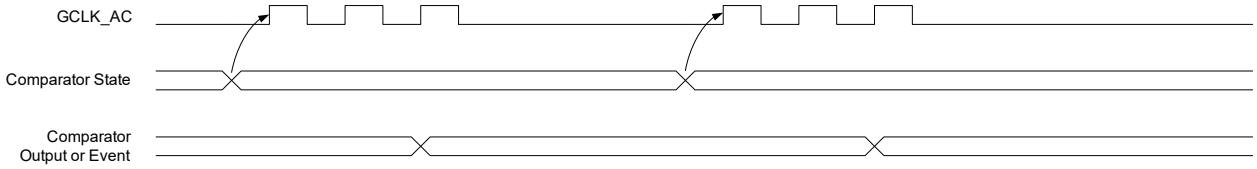
**Table 34-1. Sleep Mode Operation**

| COMPCTRLx.MODE | RUNSTDBY=0 | RUNSTDBY=1 |
|---|---|---|
| 0 (Continuous) | COMPx disabled | GCLK_AC_DIG stopped, COMPx enabled |
| 1 (Single-shot) | COMPx disabled | GCLK_AC_DIG stopped, COMPx enabled only when triggered by an input event |

#### 34.6.14.1 Continuous Measurement during Sleep

When a comparator is enabled in Continuous Measurement mode and GCLK_AC_DIG is disabled during sleep, the comparator will remain continuously enabled and will function asynchronously. The current state of the comparator is asynchronously monitored for changes. If an edge matching the interrupt condition is found, GCLK_AC_DIG is started to register the interrupt condition and generate events. If the interrupt is enabled in the Interrupt Enable registers (INTENCLR/SET), the AC can wake up the device; otherwise GCLK_AC_DIG is disabled until the next edge detection. Filtering is not possible with this configuration.

**Figure 34-10. Continuous Mode SleepWalking**

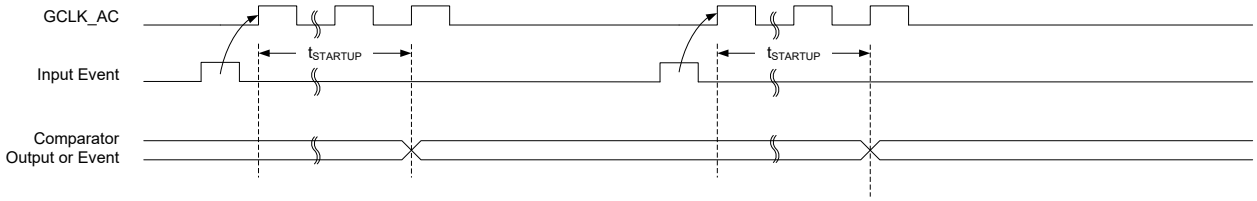



#### 34.6.14.2 Single-Shot Measurement during Sleep

For low-power operation, event-triggered measurements can be performed during sleep modes. When the event occurs, the Power Manager will start GCLK_AC_DIG. The comparator is enabled, and after the start-up time has passed, a comparison is done, with filtering if desired, and the appropriate peripheral events and interrupts are also generated, as the figure below. The comparator and GCLK_AC_DIG are

then disabled again automatically, unless configured to wake the system from sleep. Filtering is allowed with this configuration.

**Figure 34-11. Single-Shot SleepWalking**




### 34.6.15 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in control register (CTRLA.SWRST)
- Enable bit in control register (CTRLA.ENABLE)
- Enable bit in Comparator Control register (COMPCTRLn.ENABLE)

The following registers are synchronized when written:

- Window Control register (WINCTRL)

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

14.3 Register Synchronization

## 34.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | LPMUX | | | | | RUNSTDBY | ENABLE | SWRST |
| 0x01 | CTRLB | 7:0 | | | | | STARTx | STARTx | STARTx | STARTx |
| 0x02 | EVCTRL | 7:0 | | | WINEOx | WINEOx | COMPEOx | COMPEOx | COMPEOx | COMPEOx |
| | | 15:8 | | | | | COMPEIx | COMPEIx | COMPEIx | COMPEIx |
| 0x04 | INTENCLR | 7:0 | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| 0x05 | INTENSET | 7:0 | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| 0x06 | INTFLAG | 7:0 | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| 0x07 | Reserved | | | | | | | | | |
| 0x08 | STATUSA | 7:0 | WSTATE1[1:0] | | WSTATE0[1:0] | | STATEx | STATEx | STATEx | STATEx |
| 0x09 | STATUSB | 7:0 | SYNCBUSY | | | | READYx | READYx | READYx | READYx |
| 0x0A | STATUSC | 7:0 | WSTATE1[1:0] | | WSTATE0[1:0] | | STATEx | STATEx | STATEx | STATEx |
| 0x0B | Reserved | | | | | | | | | |
| 0x0C | WINCTRL | 7:0 | | WINTSEL1[1:0] | | WEN1 | | WINTSEL0[1:0] | | WEN0 |
| 0x0D ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | COMPCTRL0 | 7:0 | | INTSEL[1:0] | | | SPEED[1:0] | | SINGLE | ENABLE |
| | | 15:8 | SWAP | | MUXPOS[1:0] | | | MUXNEG[2:0] | | |
| | | 23:16 | | | | | HYST | | OUT[1:0] | |
| | | 31:24 | | | | | | FLEN[2:0] | | |
| 0x14 | COMPCTRL1 | 7:0 | | INTSEL[1:0] | | | SPEED[1:0] | | SINGLE | ENABLE |
| | | 15:8 | SWAP | | MUXPOS[1:0] | | | MUXNEG[2:0] | | |
| | | 23:16 | | | | | HYST | | OUT[1:0] | |
| | | 31:24 | | | | | | FLEN[2:0] | | |
| 0x18 ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | SCALER0 | 7:0 | | | VALUE[5:0] | | | | | |
| 0x21 | SCALER1 | 7:0 | | | VALUE[5:0] | | | | | |

## 34.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to *Register Access Protection*.

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to *Synchronization*.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 34.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPMUX | | | | | RUNSTDBY | ENABLE | SWRST |
| Access | R/W | | | | | R/W | R/W | R/W |
| Reset | 0 | | | | | 0 | 0 | 0 |

**Bit 7 – LPMUX** Low-Power Mux
This bit is not synchronized

| Value | Description |
|---|---|
| 0 | The analog input muxes have low resistance, but consume more power at lower voltages (e.g., are driven by the voltage doubler). |
| 1 | The analog input muxes have high resistance, but consume less power at lower voltages (e.g., the voltage doubler is disabled). |

**Bit 2 – RUNSTDBY** Run in Standby
This bit controls the behavior of the comparators during standby sleep mode.

This bit is not synchronized

| Value | Description |
|---|---|
| 0 | The comparator pair is disabled during sleep. |
| 1 | The comparator pair continues to operate during sleep. |

**Bit 1 – ENABLE** Enable
Due to synchronization, there is a delay from the time when the register is updated until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately after being written. STATUS.SYNCBUSY is set. STATUS.SYNCBUSY is cleared when the peripheral is enabled/disabled

| Value | Description |
|---|---|
| 0 | The AC is disabled. |
| 1 | The AC is enabled. Each comparator must also be enabled individually by the Enable bit in the Comparator Control register (COMPCTRLn.ENABLE). |

**Bit 0 – SWRST** Software Reset
Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers in the AC to their initial state, and the AC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 34.8.2 Control B

**Name:** CTRLB
**Offset:** 0x01
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | STARTx | STARTx | STARTx | STARTx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3,2,1,0 – STARTx** Comparator x Start Comparison
Writing a '0' to this field has no effect.

Writing a '1' to STARTx starts a single-shot comparison on COMPx if both the Single-Shot and Enable bits in the Comparator x Control Register are '1' (COMPCTRLx.SINGLE and COMPCTRLx.ENABLE). If comparator x is not implemented, or if it is not enabled in single-shot mode, Writing a '1' has no effect.

This bit always reads as zero.

### 34.8.3 Event Control

**Name:** EVCTRL
**Offset:** 0x02
**Reset:** 0x0000
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | COMPEIx | COMPEIx | COMPEIx | COMPEIx |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | WINEOx | WINEOx | COMPEOx | COMPEOx | COMPEOx | COMPEOx |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11,10,9,8 – COMPEIx** Comparator x Event Input
Note that several actions can be enabled for incoming events. If several events are connected to the peripheral, the enabled action will be taken for any of the incoming events. There is no way to tell which of the incoming events caused the action.

These bits indicate whether a comparison will start or not on any incoming event.

| Value | Description |
|---|---|
| 0 | Comparison will not start on any incoming event. |
| 1 | Comparison will start on any incoming event. |

**Bits 5,4 – WINEOx** Window x Event Output Enable
These bits indicate whether the window x function can generate a peripheral event or not.

| Value | Description |
|---|---|
| 0 | Window x Event is disabled. |
| 1 | Window x Event is enabled. |

**Bits 3,2,1,0 – COMPEOx** Comparator x Event Output Enable
These bits indicate whether the comparator x output can generate a peripheral event or not.

| Value | Description |
|---|---|
| 0 | COMPx event generation is disabled. |
| 1 | COMPx event generation is enabled. |

### 34.8.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x04
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5,4 – WINx** Window x Interrupt Enable
Reading this bit returns the state of the Window x interrupt enable.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit disables the Window x interrupt.

| Value | Description |
|---|---|
| 0 | The Window x interrupt is disabled. |
| 1 | The Window x interrupt is enabled. |

**Bits 3,2,1,0 – COMPx** Comparator x Interrupt Enable
Reading this bit returns the state of the Comparator x interrupt enable.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit disables the Comparator x interrupt.

| Value | Description |
|---|---|
| 0 | The Comparator x interrupt is disabled. |
| 1 | The Comparator x interrupt is enabled. |

### 34.8.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x05
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5,4 – WINx** Window x Interrupt Enable
Reading this bit returns the state of the Window x interrupt enable.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit enables the Window x interrupt.

| Value | Description |
|---|---|
| 0 | The Window x interrupt is disabled. |
| 1 | The Window x interrupt is enabled. |

**Bits 3,2,1,0 – COMPx** Comparator x Interrupt Enable
Reading this bit returns the state of the Comparator x interrupt enable.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Ready interrupt bit and enable the Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Comparator x interrupt is disabled. |
| 1 | The Comparator x interrupt is enabled. |

### 34.8.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x06
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | WINx | WINx | COMPx | COMPx | COMPx | COMPx |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5,4 – WINx** Window x
This flag is set according to the Window x Interrupt Selection bit group in the WINCTRL register (WINCTRL.WINTSELx) and will generate an interrupt if INTENCLR/SET.WINx is also one.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Window x interrupt flag.

**Bits 3,2,1,0 – COMPx** Comparator x
Reading this bit returns the status of the Comparator x interrupt flag. If comparator x is not implemented, COMPx always reads as zero.

This flag is set according to the Interrupt Selection bit group in the Comparator x Control register (COMPCTRLx.INTSEL) and will generate an interrupt if INTENCLR/SET.COMPx is also one.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Comparator x interrupt flag.

© 2018 Microchip Technology Inc.

### 34.8.7 Status A

**Name:** STATUSA
**Offset:** 0x08
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WSTATE1[1:0] | | WSTATE0[1:0] | | STATEx | STATEx | STATEx | STATEx |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:6 – WSTATE1[1:0]** Window 1 Current State
These bits show the current state of the signal if the window 1 mode is enabled.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Signal is above window |
| 0x1 | INSIDE | Signal is inside window |
| 0x2 | BELOW | Signal is below window |
| 0x3 | | Reserved |

**Bits 5:4 – WSTATE0[1:0]** Window 0 Current State
These bits show the current state of the signal if the window 0 mode is enabled.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Signal is above window |
| 0x1 | INSIDE | Signal is inside window |
| 0x2 | BELOW | Signal is below window |
| 0x3 | | Reserved |

**Bits 3,2,1,0 – STATEx** Comparator x Current State
This bit shows the current state of the output signal from COMPx. STATEx is valid only when STATUSB.READYx is one.

### 34.8.8 Status B

**Name:** STATUSB
**Offset:** 0x09
**Reset:** 0x00
**Property:** –

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | READYx | READYx | READYx | READYx |
| Access | R | | | | R | R | R | R |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

**Bit 7 – SYNCBUSY** Synchronization Busy
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

**Bits 3,2,1,0 – READYx** Comparator x Ready
This bit is cleared when the comparator x output is not ready.
This bit is set when the comparator x output is ready.

If comparator x is not implemented, READYx always reads as zero.

### 34.8.9 Status C

**Name:** STATUSC
**Offset:** 0x0A
**Reset:** 0x00
**Property:** –

STATUSC is a copy of STATUSA (see STATUSA register), with the additional feature of automatically starting single-shot comparisons. A read of STATUSC will start a comparison on all comparators currently configured for single-shot operation. The read will stall the bus until all enabled comparators are ready. If a comparator is already busy with a comparison, the read will stall until the current comparison is compete, and a new comparison will not be started.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WSTATE1[1:0] | | WSTATE0[1:0] | | STATEx | STATEx | STATEx | STATEx |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:6 – WSTATE1[1:0]** Window 1 Current State
These bits show the current state of the signal if the window 1 mode is enabled.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Signal is above window |
| 0x1 | INSIDE | Signal is inside window |
| 0x2 | BELOW | Signal is below window |
| 0x3 | | Reserved |

**Bits 5:4 – WSTATE0[1:0]** Window 0 Current State
These bits show the current state of the signal if the window 0 mode is enabled.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Signal is above window |
| 0x1 | INSIDE | Signal is inside window |
| 0x2 | BELOW | Signal is below window |
| 0x3 | | Reserved |

**Bits 3,2,1,0 – STATEx** Comparator x Current State
This bit shows the current state of the output signal from COMPx. STATEx is valid only when STATUSB.READYx is one.

### 34.8.10 Window Control

**Name:** WINCTRL
**Offset:** 0x0C
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | WINTSEL1[1:0] | | WEN1 | | WINTSEL0[1:0] | | WEN0 |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 6:5 – WINTSEL1[1:0]** Window 1 Interrupt Selection
These bits configure the interrupt mode for the comparator window 1 mode.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Interrupt on signal above window |
| 0x1 | INSIDE | Interrupt on signal inside window |
| 0x2 | BELOW | Interrupt on signal below window |
| 0x3 | OUTSIDE | Interrupt on signal outside window |

**Bit 4 – WEN1** Window 1 Mode Enable

| Value | Description |
|---|---|
| 0 | Window mode is disabled for comparators 2 and 3. |
| 1 | Window mode is enabled for comparators 2 and 3. |

**Bits 2:1 – WINTSEL0[1:0]** Window 0 Interrupt Selection
These bits configure the interrupt mode for the comparator window 0 mode.

| Value | Name | Description |
|---|---|---|
| 0x0 | ABOVE | Interrupt on signal above window |
| 0x1 | INSIDE | Interrupt on signal inside window |
| 0x2 | BELOW | Interrupt on signal below window |
| 0x3 | OUTSIDE | Interrupt on signal outside window |

**Bit 0 – WEN0** Window 0 Mode Enable

| Value | Description |
|---|---|
| 0 | Window mode is disabled for comparators 0 and 1. |
| 1 | Window mode is enabled for comparators 0 and 1. |

© 2018 Microchip Technology Inc.

### 34.8.11 Comparator Control n

**Name:** COMPCTRL
**Offset:** 0x10 + n*0x04 [n=0..1]
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FLEN[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HYST | | OUT[1:0] | |
| Access | | | | | R/W | | R/W | R/W |
| Reset | | | | | 0 | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWAP | | MUXPOS[1:0] | | | MUXNEG[2:0] | | |
| Access | R/W | | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | INTSEL[1:0] | | | SPEED[1:0] | | SINGLE | ENABLE |
| Access | | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | | 0 | 0 | 0 | 0 |

**Bits 26:24 – FLEN[2:0]** Filter Length
These bits configure the filtering for comparator n. COMPCTRLn.FLEN can only be written while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | OFF | No filtering |
| 0x1 | MAJ3 | 3-bit majority function (2 of 3) |
| 0x2 | MAJ5 | 5-bit majority function (3 of 5) |
| 0x3-0x7 | N/A | Reserved |

**Bit 19 – HYST** Hysteresis Enable
This bit indicates the hysteresis mode of comparator n. Hysteresis is available only for continuous mode (COMPCTRLn. SINGLE=0). COMPCTRLn.HYST can be written only while COMPCTRLn.ENABLE is zero.

This bit is not synchronized.

These bits are not synchronized.

| Value | Name |
|---|---|
| 0 | Hysteresis is disabled. |
| 1 | Hysteresis is enabled. |

**Bits 17:16 – OUT[1:0]** Output
These bits configure the output selection for comparator n. COMPCTRLn.OUT can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | OFF | The output of COMPn is not routed to the COMPn I/O port |
| 0x1 | ASYNC | The asynchronous output of COMPn is routed to the COMPn I/O port |
| 0x2 | SYNC | The synchronous output (including filtering) of COMPn is routed to the COMPn I/O port |
| 0x3 | N/A | Reserved |

**Bit 15 – SWAP** Swap Inputs and Invert
This bit swaps the positive and negative inputs to COMPn and inverts the output. This function can be used for offset cancellation. COMPCTRLn.SWAP can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0 | The output of MUXPOS connects to the positive input, and the output of MUXNEG connects to the negative input. |
| 1 | The output of MUXNEG connects to the positive input, and the output of MUXPOS connects to the negative input. |

**Bits 13:12 – MUXPOS[1:0]** Positive Input Mux Selection
These bits select which input will be connected to the positive input of comparator n. COMPCTRLn.MUXPOS can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | PIN0 | I/O pin 0 |
| 0x1 | PIN1 | I/O pin 1 |
| 0x2 | PIN2 | I/O pin 2 |
| 0x3 | PIN3 | I/O pin 3 |

**Bits 10:8 – MUXNEG[2:0]** Negative Input Mux Selection
These bits select which input will be connected to the negative input of comparator n. COMPCTRLn.MUXNEG can only be written while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | PIN0 | I/O pin 0 |
| 0x1 | PIN1 | I/O pin 1 |
| 0x2 | PIN2 | I/O pin 2 |
| 0x3 | PIN3 | I/O pin 3 |

© 2018 Microchip Technology Inc.

| Value | Name | Description |
|---|---|---|
| 0x4 | GND | Ground |
| 0x5 | VSCALE | VDD scaler |
| 0x6 | BANDGAP | Internal bandgap voltage |
| 0x7 | DAC | DAC output |

**Bits 6:5 – INTSEL[1:0]** Interrupt Selection
These bits select the condition for comparator n to generate an interrupt or event. COMPCTRLn.INTSEL can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | TOGGLE | Interrupt on comparator output toggle |
| 0x1 | RISING | Interrupt on comparator output rising |
| 0x2 | FALLING | Interrupt on comparator output falling |
| 0x3 | EOC | Interrupt on end of comparison (single-shot mode only) |

**Bits 3:2 – SPEED[1:0]** Speed Selection
This bit indicates the speed/propagation delay mode of comparator n. COMPCTRLn.SPEED can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Name | Description |
|---|---|---|
| 0x0 | LOW | Low speed |
| 0x1 | HIGH | High speed |
| 0x2-0x3 | N/A | Reserved |

**Bit 1 – SINGLE** Single-Shot Mode
This bit determines the operation of comparator n. COMPCTRLn.SINGLE can be written only while COMPCTRLn.ENABLE is zero.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0 | Comparator n operates in continuous measurement mode. |
| 1 | Comparator n operates in single-shot mode. |

**Bit 0 – ENABLE** Enable
Writing a zero to this bit disables comparator n.
Writing a one to this bit enables comparator n. After writing to this bit, the value read back will not change until the action initiated by the writing is complete.

Due to synchronization, there is a latency of at least two GCLK_AC_DIG clock cycles from updating the register until the comparator is enabled/disabled. The bit will continue to read the previous state while the change is in progress. Writing a one to COMPCTRLn.ENABLE will prevent further changes to the other bits in COMPCTRLn. These bits remain protected until COMPCTRLn.ENABLE is written to zero and the write is synchronized.

© 2018 Microchip Technology Inc.

### 34.8.12 Scaler n

**Name:** SCALER
**Offset:** 0x20 + n*0x01 [n=0..1]
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | VALUE[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – VALUE[5:0]** Scaler Value
These bits define the scaling factor for channel n of the $V_{DD}$ voltage scaler. The output voltage, $V_{SCALE}$, is:

$$V_{\mathrm{SCALE}} = \frac{V_{\mathrm{DD}} \cdot (\mathrm{VALUE}+1)}{64}$$

# 35. DAC – Digital-to-Analog Converter

## 35.1 Overview

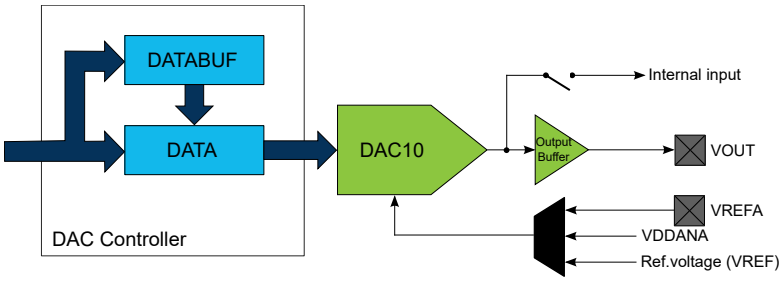
The Digital-to-Analog Converter (DAC) converts a digital value to a voltage. The DAC has one channel with 10-bit resolution, and it is capable of converting up to 350,000 samples per second (350ksps).

## 35.2 Features

- DAC with 10-bit resolution
- Up to 350ksps conversion rate
- Multiple trigger sources
- High-drive capabilities
- Output can be used as input to the Analog Comparator (AC)
- DMA support

## 35.3 Block Diagram

**Figure 35-1. DAC Block Diagram**




## 35.4 Signal Description

| Signal Name | Type | Description |
|---|---|---|
| VOUT | Analog output | DAC output |
| VREFA | Analog input | External reference |

**Related Links**

7. I/O Multiplexing and Considerations

## 35.5 Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 35.5.1 I/O Lines

Using the DAC Controller's I/O lines requires the I/O pins to be configured using the port configuration (PORT).

**Related Links**

23. PORT - I/O Pin Controller

### 35.5.2 Power Management

The DAC will continue to operate in any Sleep mode where the selected source clock is running.

The DAC interrupts can be used to wake up the device from sleep modes.

Events connected to the event system can trigger other operations in the system without exiting sleep modes.

**Related Links**

16. PM – Power Manager

### 35.5.3 Clocks

The DAC bus clock (CLK_DAC_APB) can be enabled and disabled by the Power Manager, and the default state of CLK_DAC_APB can be found in the *Peripheral Clock Masking* section.

A generic clock (GCLK_DAC) is required to clock the DAC Controller. This clock must be configured and enabled in the Generic Clock Controller before using the DAC Controller. Refer to *GCLK – Generic Clock Controller* for details.

This generic clock is asynchronous to the bus clock (CLK_DAC_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to 35.6.7 Synchronization for further details.

**Related Links**

16.6.2.6 Peripheral Clock Masking

15. GCLK - Generic Clock Controller

### 35.5.4 DMA

The DMA request line is connected to the DMA Controller (DMAC). Using the DAC Controller DMA requests requires to configure the DMAC first.

**Related Links**

20. DMAC – Direct Memory Access Controller

### 35.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the DAC Controller interrupt(s) requires the interrupt controller to be configured first.

**Related Links**

11.2 Nested Vector Interrupt Controller

### 35.5.6 Events

The events are connected to the Event System.

**Related Links**

24. EVSYS – Event System

### 35.5.7 Debug Operation

When the CPU is halted in debug mode the DAC will halt normal operation. Any on-going conversions will be completed. The DAC can be forced to continue normal operation during debugging. If the DAC is

configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

### 35.5.8 Register Access Protection

All registers with write-access can be write-protected optionally by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear (INTFLAG) register
- Data Buffer (DATABUF) register

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

PAC write-protection does not apply to accesses through an external debugger

**Related Links**

11.6 PAC - Peripheral Access Controller

### 35.5.9 Analog Connections

The DAC has one output pin (VOUT) and one analog input pin (VREFA) that must be configured first.

When internal input is used, it must be enabled before DAC Controller is enabled.

## 35.6 Functional Description

### 35.6.1 Principle of Operation

The DAC converts the digital value located in the Data register (DATA) into an analog voltage on the DAC output (VOUT).

A conversion is started when new data is written to the Data register. The resulting voltage is available on the DAC output after the conversion time. A conversion can also be started by input events from the Event System.

### 35.6.2 Basic Operation

#### 35.6.2.1 Initialization

The following registers are enable-protected, meaning they can only be written when the DAC is disabled (CTRLA.ENABLE is zero):

- Control B register (CTRLB)
- Event Control register (EVCTRL)

Enable-protection is denoted by the Enable-Protected property in the register description.

Before enabling the DAC, it must be configured by selecting the voltage reference using the Reference Selection bits in the Control B register (CTRLB.REFSEL).

#### 35.6.2.2 Enabling, Disabling and Resetting

The DAC Controller is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The DAC Controller is disabled by writing a '0' to CTRLA.ENABLE.

The DAC Controller is reset by writing a '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the DAC will be reset to their initial state, and the DAC Controller will be disabled. Refer to the CTRLA register for details.

#### 35.6.2.3 Enabling the Output Buffer

To enable the DAC output on the $V_{OUT}$ pin, the output driver must be enabled by writing a one to the External Output Enable bit in the Control B register (CTRLB.EOEN).

The DAC output buffer provides a high-drive-strength output, and is capable of driving both resistive and capacitive loads. To minimize power consumption, the output buffer should be enabled only when external output is needed.

#### 35.6.2.4 Digital to Analog Conversion

The DAC converts a digital value (stored in the DATA register) into an analog voltage. The conversion range is between GND and the selected DAC voltage reference. The default voltage reference is the internal reference voltage. Other voltage reference options are the analog supply voltage (VDDANA) and the external voltage reference (VREFA). The voltage reference is selected by writing to the Reference Selection bits in the Control B register (CTRLB.REFSEL).

The output voltage from the DAC can be calculated using the following formula:

$$V_{\mathrm{OUT}} = \frac{\mathrm{DATA}}{0x3\mathrm{FF}} \cdot \mathrm{VREF}$$

A new conversion starts as soon as a new value is loaded into DATA. DATA can either be loaded via the APB bus during a CPU write operation, using DMA, or from the DATABUF register when a START event occurs. Refer to 35.6.5 Events for details. As there is no automatic indication that a conversion is done, the sampling period must be greater than or equal to the specified conversion time.

### 35.6.3 DMA Operation

The DAC generates the following DMA request:

- Data Buffer Empty (EMPTY): The request is set when data is transferred from DATABUF to the internal data buffer of DAC. The request is cleared when DATABUF register is written, or by writing a one to the EMPTY bit in the Interrupt Flag register (INTFLAG.EMPTY).

For each Start Conversion event, DATABUF is transferred into DATA and the conversion starts. When DATABUF is empty, the DAC generates the DMA request for new data. As DATABUF is initially empty, a DMA request is generated whenever the DAC is enabled.

If the CPU accesses the registers that are the source of a DMA request set/clear condition, the DMA request can be lost or the DMA transfer can be corrupted, if enabled.

When DAC registers are write-protected by Peripheral Access Controller, DATABUF cannot be written. To bypass DATABUF write protection, Bypass DATABUF Write Protection bit (CTRLB.BDWP) must be written to '1'

### 35.6.4 Interrupts

The DAC Controller has the following interrupt sources:

- Data Buffer Empty (EMPTY): Indicates that the internal data buffer of the DAC is empty.
- Underrun (UNDERRUN): Indicates that the internal data buffer of the DAC is empty and a DAC start of conversion event occurred. Refer to 35.6.5 Events for details.
- Synchronization Ready (SYNCRDY): this asynchronous interrupt can be used to wake-up the device from any sleep mode.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register

(INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the DAC is reset. See INTFLAG register for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate one combined interrupt request to the NVIC. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated..

**Related Links**

11.2 Nested Vector Interrupt Controller

### 35.6.5 Events

The DAC Controller can generate the following output events:

- Data Buffer Empty (EMPTY): Generated when the internal data buffer of the DAC is empty. Refer to DMA Operation for details.

Writing a '1' to an Event Output bit in the Event Control register (EVCTRL.EMPTYEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The DAC can take the following action on an input event:

- Start Conversion (START): DATABUF value is transferred into DATA as soon as the DAC is ready for the next conversion, and then conversion is started. START is considered as asynchronous to GCLK_DAC thus it is resynchronized in DAC Controller. Refer to 35.6.2.4 Digital to Analog Conversion for details.

Writing a '1' to an Event Input bit in the Event Control register (EVCTRL.STARTEI) enables the corresponding action on an input event. Writing a '0' to this bit disables the corresponding action on input event.

**Note:** When several events are connected to the DAC Controller, the enabled action will be taken on any of the incoming events.

By default, DAC Controller detects rising edge events. Falling edge detection can be enabled by writing a '1' to EVCTRL.INVEIx.

**Related Links**

24. EVSYS – Event System

### 35.6.6 Sleep Mode Operation

The generic clock for the DAC is running in idle sleep mode. If the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) is one, the DAC output buffer will keep its value in standby sleep mode. If CTRLA.RUNSTDBY is zero, the DAC output buffer will be disabled in standby sleep mode.

### 35.6.7 Synchronization

Due to the asynchronicity between main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read. A register can require:

- Synchronization when written
- Synchronization when read

- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while its busy bit is one, the operation is discarded and an error is generated.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)
- All bits in the Data register (DATA)
- All bits in the Data Buffer register (DATABUF)

Write-synchronization is denoted by the Write-Synchronized property in the register description.

The following bits need synchronization when read:

- All bits in the Data register (DATA)

### 35.6.8 Additional Features

#### 35.6.8.1 DAC as an Internal Reference

The DAC output can be internally enabled as input to the analog comparator. This is enabled by writing a one to the Internal Output Enable bit in the Control B register (CTRLB.IOEN). It is possible to have the internal and external output enabled simultaneously.

The DAC output can also be enabled as input to the Analog-to-Digital Converter. In this case, the output buffer must be enabled.

#### 35.6.8.2 Data Buffer

The Data Buffer register (DATABUF) and the Data register (DATA) are linked together to form a two-stage FIFO. The DAC uses the Start Conversion event to load data from DATABUF into DATA and start a new conversion. The Start Conversion event is enabled by writing a one to the Start Event Input bit in the Event Control register (EVCTRL.STARTEI). If a Start Conversion event occurs when DATABUF is empty, an Underrun interrupt request is generated if the Underrun interrupt is enabled.

The DAC can generate a Data Buffer Empty event when DATABUF becomes empty and new data can be loaded to the buffer. The Data Buffer Empty event is enabled by writing a one to the Empty Event Output bit in the Event Control register (EVCTRL.EMPTYEO). A Data Buffer Empty interrupt request is generated if the Data Buffer Empty interrupt is enabled.

#### 35.6.8.3 Voltage Pump

When the DAC is used at operating voltages lower than 2.5V, the voltage pump must be enabled. This enabling is done automatically, depending on operating voltage.

The voltage pump can be disabled by writing a one to the Voltage Pump Disable bit in the Control B register (CTRLB.VPD). This can be used to reduce power consumption when the operating voltage is above 2.5V.

The voltage pump uses the asynchronous GCLK_DAC clock, and requires that the clock frequency be at least four times higher than the sampling period.

## 35.7 Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | | | | | | RUNSTDBY | ENABLE | SWRST |
| 0x01 | CTRLB | 7:0 | REFSEL[1:0] | | | BDWP | VPD | LEFTADJ | IOEN | EOEN |
| 0x02 | EVCTRL | 7:0 | | | | | | | EMPTYEO | STARTEI |
| 0x03 | Reserved | | | | | | | | | |
| 0x04 | INTENCLR | 7:0 | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| 0x05 | INTENSET | 7:0 | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| 0x06 | INTFLAG | 7:0 | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| 0x07 | STATUS | 7:0 | SYNCBUSY | | | | | | | |
| 0x08 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| 0x0A ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | DATABUF | 7:0 | DATABUF[7:0] | | | | | | | |
| | | 15:8 | DATABUF[15:8] | | | | | | | |

## 35.8 Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Optional PAC write-protection is denoted by the "PAC Write-Protection" property in each individual register description. For details, refer to 35.5.8 Register Access Protection.

Some registers are synchronized when read and/or written. Synchronization is denoted by the "Write-Synchronized" or the "Read-Synchronized" property in each individual register description. For details, refer to 35.6.7 Synchronization.

Some registers are enable-protected, meaning they can only be written when the peripheral is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 35.8.1 Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00
**Property:** PAC Write-Protection, Write-Synchronized

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RUNSTDBY | ENABLE | SWRST |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – RUNSTDBY** Run in Standby
This bit is not synchronized

| Value | Description |
|---|---|
| 0 | The DAC output buffer is disabled in standby sleep mode. |
| 1 | The DAC output buffer can be enabled in standby sleep mode. |

**Bit 1 – ENABLE** Enable DAC Controller
Due to synchronization there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the corresponding bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST** Software Reset
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the DAC to their initial state, and the DAC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 35.8.2 Control B

**Name:** CTRLB
**Offset:** 0x01
**Reset:** 0x00
**Property:** PAC Write-Protection, Enable-Protected

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REFSEL[1:0] | | | BDWP | VPD | LEFTADJ | IOEN | EOEN |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |

**Bits 7:6 – REFSEL[1:0]** Reference Selection
This bit field selects the Reference Voltage for the DAC.

| Value | Name | Description |
|---|---|---|
| 0x0 | INTREF | Internal voltage reference |
| 0x1 | VDDANA | Analog voltage supply |
| 0x2 | VREFA | External reference |
| 0x3 | | Reserved |

**Bit 4 – BDWP** Bypass DATABUF Write Protection
This bit can bypass DATABUF write protection.

| Value | Description |
|---|---|
| 0 | DATABUF register is write-protected by Peripheral Access Controller. |
| 1 | DATABUF register is not write-protected. |

**Bit 3 – VPD** Voltage Pump Disabled
This bit controls the behavior of the voltage pump.

| Value | Description |
|---|---|
| 0 | Voltage pump is turned on/off automatically |
| 1 | Voltage pump is disabled. |

**Bit 2 – LEFTADJ** Left-Adjusted Data
This bit controls how the 10-bit conversion data is adjusted in the Data and Data Buffer registers.

| Value | Description |
|---|---|
| 0 | DATA and DATABUF registers are right-adjusted. |
| 1 | DATA and DATABUF registers are left-adjusted. |

**Bit 1 – IOEN** Internal Output Enable

| Value | Description |
|---|---|
| 0 | Internal DAC output not enabled. |
| 1 | Internal DAC output enabled to be used by the AC. |

**Bit 0 – EOEN** External Output Enable

| Value | Description |
|---|---|
| 0 | The DAC output is turned off. |
| 1 | The high-drive output buffer drives the DAC output to the $V_{OUT}$ pin. |

### 35.8.3 Event Control

**Name:** EVCTRL
**Offset:** 0x02
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EMPTYEO | STARTEI |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – EMPTYEO** Data Buffer Empty Event Output
This bit indicates whether or not the Data Buffer Empty event is enabled and will be generated when the Data Buffer register is empty.

| Value | Description |
|---|---|
| 0 | Data Buffer Empty event is disabled and will not be generated. |
| 1 | Data Buffer Empty event is enabled and will be generated. |

**Bit 0 – STARTEI** Start Conversion Event Input
This bit indicates whether or not the Start Conversion event is enabled and data are loaded from the Data Buffer register to the Data register upon event reception.

| Value | Description |
|---|---|
| 0 | A new conversion will not be triggered on any incoming event. |
| 1 | A new conversion will be triggered on any incoming event. |

### 35.8.4 Interrupt Enable Clear

**Name:** INTENCLR
**Offset:** 0x04
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Synchronization Ready Interrupt Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – EMPTY** Data Buffer Empty Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer Empty Interrupt Enable bit, which disables the Data Buffer Empty interrupt.

| Value | Description |
|---|---|
| 0 | The Data Buffer Empty interrupt is disabled. |
| 1 | The Data Buffer Empty interrupt is enabled. |

**Bit 0 – UNDERRUN** Underrun Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer Underrun Interrupt Enable bit, which disables the Data Buffer Underrun interrupt.

| Value | Description |
|---|---|
| 0 | The Data Buffer Underrun interrupt is disabled. |
| 1 | The Data Buffer Underrun interrupt is enabled. |

### 35.8.5 Interrupt Enable Set

**Name:** INTENSET
**Offset:** 0x05
**Reset:** 0x00
**Property:** PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Synchronization Ready Interrupt Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – EMPTY** Data Buffer Empty Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Data Buffer Empty Interrupt Enable bit, which enables the Data Buffer Empty interrupt.

| Value | Description |
|---|---|
| 0 | The Data Buffer Empty interrupt is disabled. |
| 1 | The Data Buffer Empty interrupt is enabled. |

**Bit 0 – UNDERRUN** Underrun Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will set the Data Buffer Underrun Interrupt Enable bit, which enables the Data Buffer Underrun interrupt.

| Value | Description |
|---|---|
| 0 | The Data Buffer Underrun interrupt is disabled. |
| 1 | The Data Buffer Underrun interrupt is enabled. |

### 35.8.6 Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x06
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SYNCRDY | EMPTY | UNDERRUN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – SYNCRDY** Synchronization Ready Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Synchronization Ready Interrupt Enable bit, which disables the Synchronization Ready interrupt.

| Value | Description |
|---|---|
| 0 | The Synchronization Ready interrupt is disabled. |
| 1 | The Synchronization Ready interrupt is enabled. |

**Bit 1 – EMPTY** Data Buffer Empty
This flag is cleared by writing a '1' to it or by writing new data to DATABUF.

This flag is set when data is transferred from DATABUF to DATA, and the DAC is ready to receive new data in DATABUF, and will generate an interrupt request if INTENCLR/SET.EMPTY is one.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Data Buffer Empty interrupt flag.

**Bit 0 – UNDERRUN** Underrun
This flag is cleared by writing a '1' to it.

This flag is set when a start conversion event occurs when DATABUF is empty, and will generate an interrupt request if INTENCLR/SET.UNDERRUN is one.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Underrun interrupt flag.

### 35.8.7 Status

**Name:** STATUS
**Offset:** 0x07
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYNCBUSY | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bit 7 – SYNCBUSY** Synchronization Busy Status
This bit is cleared when the synchronization of registers between the clock domains is complete.

This bit is set when the synchronization of registers between clock domains is started.

### 35.8.8 Data DAC

**Name:** DATA
**Offset:** 0x08
**Reset:** 0x0000
**Property:** PAC Write-Protection, Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – DATA[15:0]** Data value to be converted
DATA register contains the 10-bit value that is converted to a voltage by the DAC. The adjustment of these 10 bits within the 16-bit register is controlled by CTRLB.LEFTADJ.

**Table 35-1. Valid Data Bits**

| CTRLB.LEFTADJ | DATA | Description |
|---|---|---|
| 0 | DATA[9:0] | Right adjusted, 10-bits |
| 1 | DATA[15:6] | Left adjusted, 10-bits |

### 35.8.9 Data Buffer

**Name:** DATABUF
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** Write-Synchronized

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATABUF[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATABUF[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – DATABUF[15:0]** Data Buffer
DATABUF contains the value to be transferred into DATA register.

# 36. Peripheral Touch Controller (PTC)

## 36.1 Overview

The Peripheral Touch Controller (PTC) acquires signals in order to detect a touch on the capacitive sensors. The external capacitive touch sensor is typically formed on a PCB, and the sensor electrodes are connected to the analog front end of the PTC through the I/O pins in the device. The PTC supports both self- and mutual-capacitance sensors.

In the mutual-capacitance mode, sensing is done using capacitive touch matrices in various X-Y configurations, including indium tin oxide (ITO) sensor grids. The PTC requires one pin per X-line and one pin per Y-line.

In the self-capacitance mode, the PTC requires only one pin (Y-line) for each touch sensor.

The number of available pins and the assignment of X- and Y-lines is depending on both package type and device configuration. Refer to the Configuration Summary and I/O Multiplexing table for details.

**Related Links**

7. I/O Multiplexing and Considerations
2. Configuration Summary

## 36.2 Features

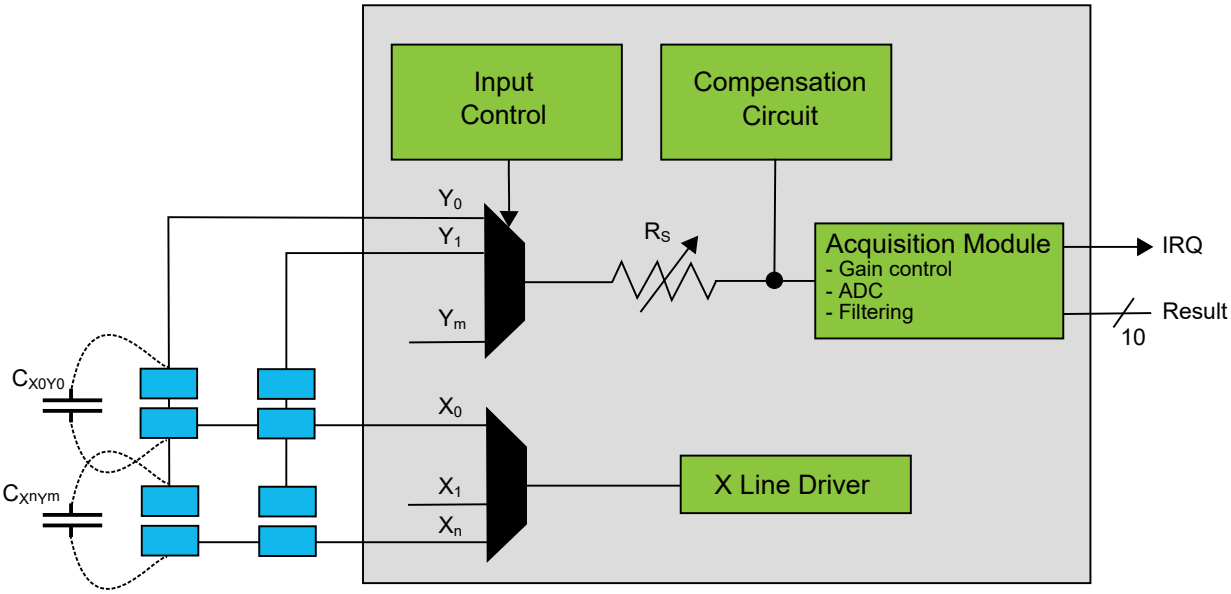
- Low-power, high-sensitivity, environmentally robust capacitive touch buttons, sliders, and wheels
- Supports wake-up on touch from Standby Sleep mode
- Supports mutual capacitance and self-capacitance sensing
  - 6/10/16 buttons in self-capacitance mode, for 32-/48-/64- pins respectively
  - 60/120/256 buttons in mutual-capacitance mode, for 32-/48-/64- pins respectively
  - Mix-and-match mutual-and self-capacitance sensors
- One pin per electrode – no external components
- Load compensating charge sensing
  - Parasitic capacitance compensation and adjustable gain for superior sensitivity
- Zero drift over the temperature and $V_{DD}$ range
  - Auto calibration and recalibration of sensors
- Single-shot charge measurement
- Hardware noise filtering and noise signal desynchronization for high conducted immunity
- Selectable channel change delay allows choosing the settling time on a new channel, as required
- Acquisition-start triggered by command or through auto-triggering feature
- Low CPU utilization through interrupt on acquisition-complete

**Related Links**

7. I/O Multiplexing and Considerations
2. Configuration Summary

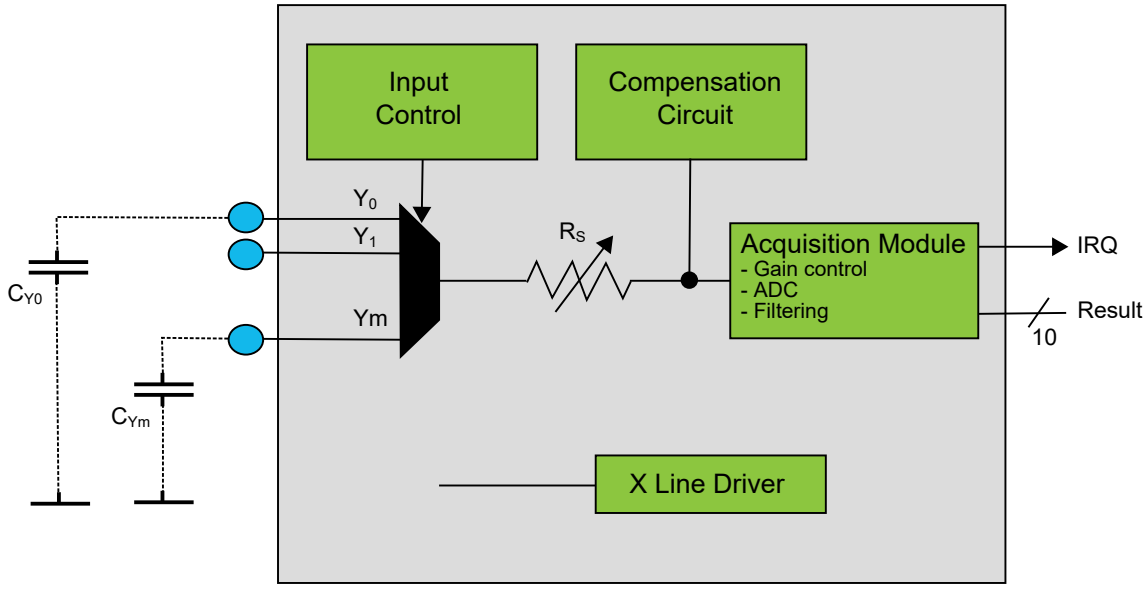
## 36.3 Block Diagram

**Figure 36-1. PTC Block Diagram Mutual-Capacitance**




**Note:** For SAM D21 the $R_S$ = 0, 20, 50, 100 KΩ.

**Figure 36-2. PTC Block Diagram Self-Capacitance**




**Note:** For SAM D21 the RS = 0, 20, 50, 100 KΩ.

## 36.4 Signal Description

**Table 36-1. Signal Description for PTC**

| Name | Type | Description |
|---|---|---|
| Y[m:0] | Analog | Y-line (Input/Output) |
| X[n:0] | Digital | X-line (Output) |

**Note:** The number of X- and Y-lines are device dependent. Refer to *Configuration Summary* for details.

Refer to *I/O Multiplexing and Considerations* for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

**Related Links**

7. I/O Multiplexing and Considerations
2. Configuration Summary

## 36.5 System Dependencies

In order to use this Peripheral, configure the other components of the system as described in the following sections.

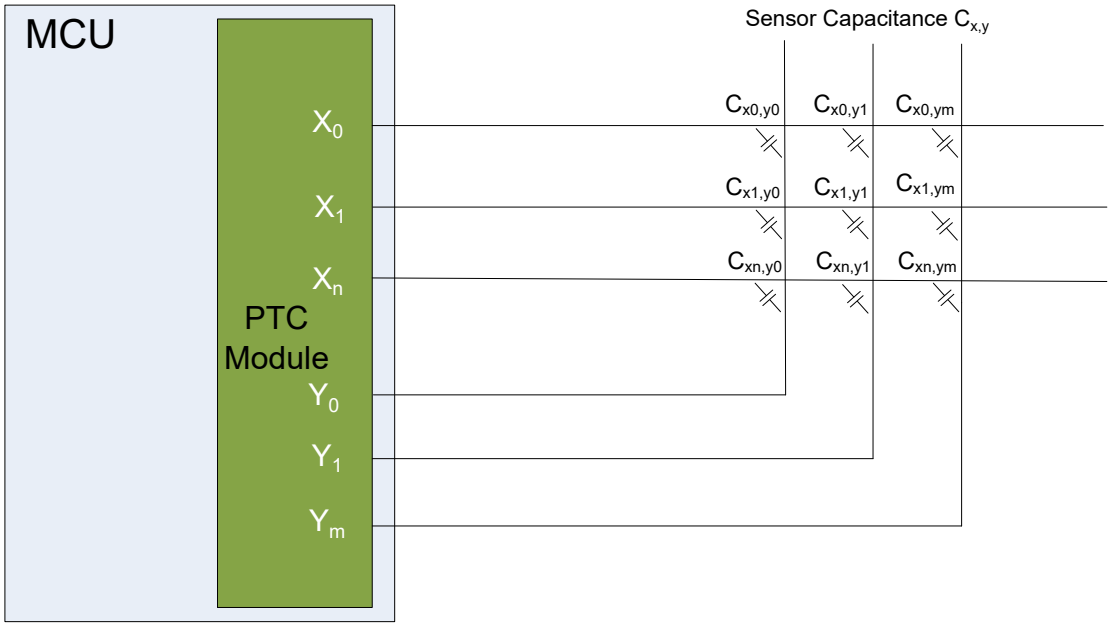
### 36.5.1 I/O Lines

The I/O lines used for analog X-lines and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not required for normal operation. However, to improve the EMC performance, a series resistor of 1 kΩ or more can be used on X-lines and Y-lines.

#### 36.5.1.1 Mutual-Capacitance Sensor Arrangement

A mutual-capacitance sensor is formed between two I/O lines - an X electrode for transmitting and Y electrode for sensing. The mutual capacitance between the X and Y electrode is measured by the Peripheral Touch Controller.

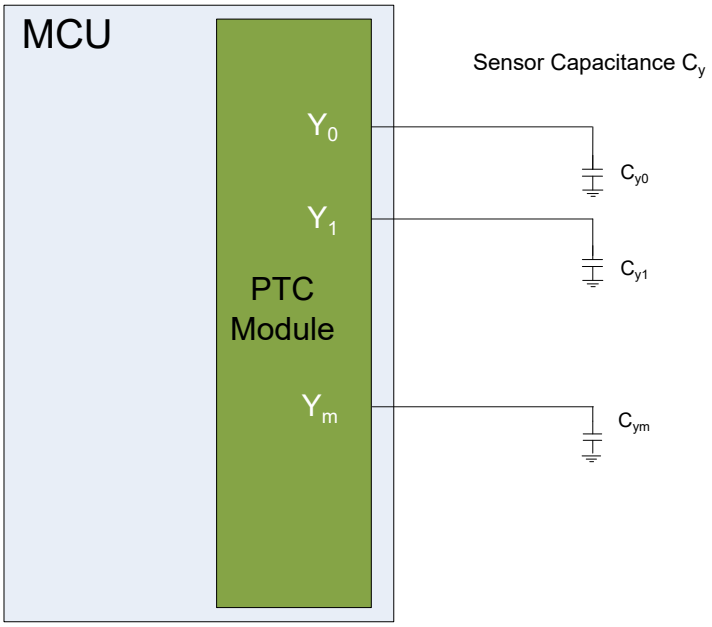
**Figure 36-3. Mutual Capacitance Sensor Arrangement**




### 36.5.1.2 Self-Capacitance Sensor Arrangement

A self-capacitance sensor is connected to a single pin on the Peripheral Touch Controller through the Y electrode for sensing the signal. The sense electrode capacitance is measured by the Peripheral Touch Controller.

**Figure 36-4. Self-Capacitance Sensor Arrangement**




For more information about designing the touch sensor, refer to Buttons, Sliders and Wheels Touch Sensor Design Guide.

### 36.5.2 Clocks

The PTC is clocked by the GCLK_PTC clock. The PTC operates from an asynchronous clock source and the operation is independent of the main system clock and its derivative clocks, such as the peripheral

bus clock (CLK_APB). A number of clock sources can be selected as the source for the asynchronous GCLK_PTC. The clock source is selected by configuring the Generic Clock Selection ID in the Generic Clock Control register. For more information about selecting the clock sources, refer to *GCLK - Generic Clock Controller*.

The selected clock must be enabled in the Power Manager, before it can be used by the PTC. By default these clocks are disabled. The frequency range of GCLK_PTC is 400kHz to 4MHz.

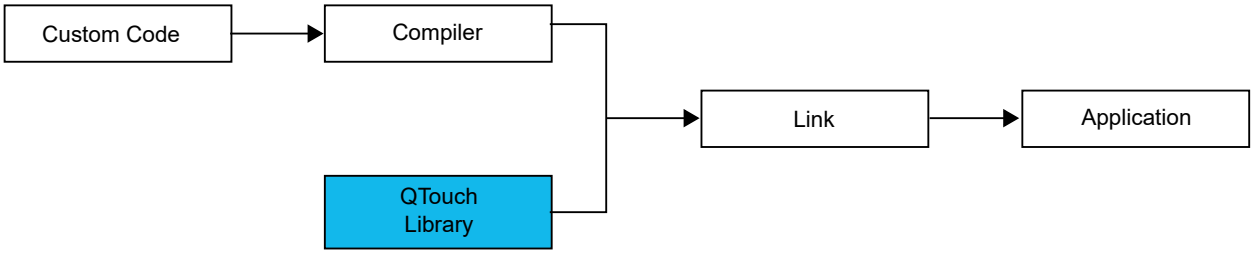
**Related Links**

15. GCLK - Generic Clock Controller
16. PM – Power Manager

## 36.6 Functional Description

In order to access the PTC, the user must use the Atmel Start QTouch® Configurator to configure and link the QTouch Library firmware with the application software. QTouch Library can be used to implement buttons, sliders, and wheels in a variety of combinations on a single interface.

**Figure 36-5. QTouch Library Usage**




For more information about QTouch Library, refer to the QTouch Library Peripheral Touch Controller User Guide.

# 37. Electrical Characteristics

## 37.1 Disclaimer

All typical values are measured at T = 25°C unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

## 37.2 Absolute Maximum Ratings

Stresses beyond those listed in this section may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**Table 37-1. Absolute Maximum Ratings**

| Symbol | Description | Min. | Max. | Units |
|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | 0 | 3.8 | V |
| $I_{VDD}$ | Current into a $V_{DD}$ pin | - | 92(1) | mA |
| $I_{GND}$ | Current out of a GND pin | - | 130(1) | mA |
| $V_{PIN}$ | Pin voltage with respect to GND and $V_{DD}$ | GND-0.6V | $V_{DD}$+0.6V | V |
| $T_{storage}$ | Storage temperature | -60 | 150 | °C |

1. Maximum source current is 46mA and maximum sink current is 65mA per cluster. A cluster is a group of GPIOs as shown in the table below. Also note that each VDD/GND pair is connected to two clusters so current consumption through the pair will be a sum of the clusters source/sink currents.

**⚠ CAUTION** This device is sensitive to electrostatic discharges (ESD). Improper handling may lead to permanent performance degradation or malfunctioning.
Handle the device following best practice ESD protection rules: Be aware that the human body can accumulate charges large enough to impair functionality or destroy the device.

**⚠ CAUTION** In debugger cold-plugging mode, NVM erase operations are not protected by the BOD33 and BOD12. NVM erase operation at supply voltages below specified minimum can cause corruption of NVM areas that are mandatory for correct device behavior.

**Related Links**

7.2.4 GPIO Clusters

## 37.3 General Operating Ratings

The device must operate within the ratings in order for all other electrical characteristics and typical characteristics of the device to be valid.

**Table 37-2. General Operating Conditions**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $V_{DDANA}$ | Analog supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $T_A$ | Temperature range | | -40 | 25 | 85 | °C |
| $T_J$ | Junction temperature | | - | - | 100 | °C |

1. With BOD33 disabled. If the BOD33 is enabled, check Table 37-20.

## 37.4 Supply Characteristics

The following characteristics are applicable to the operating temperature range: $T_A$ = -40°C to 85°C, unless otherwise specified and are valid for a junction temperature up to $T_J$ = 100°C. Refer to *Power Supply and Start-Up Considerations*.

**Table 37-3. Supply Characteristics**

| Symbol | Conditions | Voltage | | |
|---|---|---|---|---|
| | | Min. | Max. | Units |
| $V_{DDIO}$<br>$V_{DDIN}$<br>$V_{DDANA}$ | Full Voltage Range | 1.62 | 3.63 | V |

**Table 37-4. Supply Rise Rates**

| Symbol | Parameter | Rise Rate | Units |
|---|---|---|---|
| | | Max. | |
| $V_{DDIO}$<br>$V_{DDIN}$<br>$V_{DDANA}$ | DC supply peripheral I/Os, internal regulator and analog supply voltage | 0.1 | V/µs |

**Related Links**

8. Power Supply and Start-Up Considerations

## 37.5 Maximum Clock Frequencies

**Table 37-5. Maximum GCLK Generator Output Frequencies**

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN0}$ / $f_{GCLK_MAIN}$ | GCLK Generator Output Frequency | Undivided | 96 | MHz |
| $f_{GCLKGEN1}$<br>$f_{GCLKGEN2}$<br>$f_{GCLKGEN3}$ | | Divided | 48 | MHz |

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN4}$ | | | | |
| $f_{GCLKGEN5}$ | | | | |
| $f_{GCLKGEN6}$ | | | | |
| $f_{GCLKGEN7}$ | | | | |
| $f_{GCLKGEN8}$ | | | | |

**Table 37-6. Maximum Peripheral Clock Frequencies**

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{CPU}$ | CPU clock frequency | 48 | MHz |
| $f_{AHB}$ | AHB clock frequency | 48 | MHz |
| $f_{APBA}$ | APBA clock frequency | 48 | MHz |
| $f_{APBB}$ | APBB clock frequency | 48 | MHz |
| $f_{APBC}$ | APBC clock frequency | 48 | MHz |
| $f_{GCLK_DFLL48M_REF}$ | DFLL48M Reference clock frequency | 33 | KHz |
| $f_{GCLK_DPLL}$ | FDPLL96M Reference clock frequency | 2 | MHz |
| $f_{GCLK_DPLL_32K}$ | FDPLL96M 32k Reference clock frequency | 32 | KHz |
| $f_{GCLK_WDT}$ | WDT input clock frequency | 48 | MHz |
| $f_{GCLK_RTC}$ | RTC input clock frequency | 48 | MHz |
| $f_{GCLK_EIC}$ | EIC input clock frequency | 48 | MHz |
| $f_{GCLK_USB}$ | USB input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_0}$ | EVSYS channel 0 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_1}$ | EVSYS channel 1 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_2}$ | EVSYS channel 2 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_3}$ | EVSYS channel 3 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_4}$ | EVSYS channel 4 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_5}$ | EVSYS channel 5 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_6}$ | EVSYS channel 6 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_7}$ | EVSYS channel 7 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_8}$ | EVSYS channel 8 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_9}$ | EVSYS channel 9 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_10}$ | EVSYS channel 10 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_11}$ | EVSYS channel 11 input clock frequency | 48 | MHz |

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{GCLK_SERCOMx_SLOW}$ | Common SERCOM slow input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM0_CORE}$ | SERCOM0 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM1_CORE}$ | SERCOM1 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM2_CORE}$ | SERCOM2 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM3_CORE}$ | SERCOM3 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM4_CORE}$ | SERCOM4 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM5_CORE}$ | SERCOM5 input clock frequency | 48 | MHz |
| $f_{GCLK_TCC0,\ GCLK_TCC1}$ | TCC0,TCC1 input clock frequency | 96 | MHz |
| $f_{GCLK_TCC2,\ GCLK_TC3}$ | TCC2,TC3 input clock frequency | 96 | MHz |
| $f_{GCLK_TC4,\ GCLK_TC5}$ | TC4,TC5 input clock frequency | 48 | MHz |
| $f_{GCLK_TC6,\ GCLK_TC7}$ | TC6,TC7 input clock frequency | 48 | MHz |
| $f_{GCLK_ADC}$ | ADC input clock frequency | 48 | MHz |
| $f_{GCLK_AC_DIG}$ | AC digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC_ANA}$ | AC analog input clock frequency | 64 | KHz |
| $f_{GCLK_AC1_DIG}$ | AC1 digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC1_ANA}$ | AC1 analog input clock frequency | 64 | KHz |
| $f_{GCLK_DAC}$ | DAC input clock frequency | 350 | KHz |
| $f_{GCLK_PTC}$ | PTC input clock frequency | 48 | MHz |
| $f_{GCLK_I2S_0}$ | I2S serializer 0 input clock frequency | 13 | MHz |
| $f_{GCLK_I2S_1}$ | I2S serializer 1 input clock frequency | 13 | MHz |

## 37.6 Power Consumption

The values in this section are measured values of power consumption under the following conditions, except where noted:

- Operating conditions
  - $V_{VDDIN}$ = 3.3 V
  - VDDIN = 1.8V, CPU is running on Flash with three wait states
- Wake up time from sleep mode is measured from the edge of the wakeup signal to the execution of the first instruction fetched in flash.
- Oscillators
  - XOSC (crystal oscillator) stopped
  - XOSC32K (32 kHz crystal oscillator) running with external 32kHz crystal
  - DFLL48M using XOSC32K as reference and running at 48 MHz
- Clocks
  - DFLL48M used as main clock source, except otherwise specified

  - CPU, AHB clocks undivided
  - APBA clock divided by 4
  - APBB and APBC bridges off
- The following AHB module clocks are running: NVMCTRL, APBA bridge
  - All other AHB clocks stopped
- The following peripheral clocks running: PM, SYSCTRL, RTC
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait states
- Cache enabled
- BOD33 disabled

**Table 37-7. Current Consumption (Device Variant A)**

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ACTIVE | CPU running a While(1) algorithm | 25°C | 3.11 | 3.37 | 3.64 | mA |
| | | 85°C | 3.24 | 3.48 | 3.76 | |
| | CPU running a While(1) algorithm $V_{DDIN}$=1.8V, CPU is running on Flash with 3 wait states | 25°C | 3.10 | 3.36 | 3.64 | |
| | | 85°C | 3.24 | 3.48 | 3.75 | |
| | CPU running a While(1) algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | 25°C | 60*freq + 74 | 60*freq + 136 | 62*freq + 196 | µA (with freq in MHz) |
| | | 85°C | 62*freq + 154 | 62*freq + 228 | 62*freq + 302 | |
| | CPU running a Fibonacci algorithm | 25°C | 4.12 | 4.53 | 4.92 | mA |
| | | 85°C | 4.27 | 4.63 | 4.98 | |
| | CPU running a Fibonacci algorithm $V_{DDIN}$=1.8V, CPU is running on flash with 3 wait states | 25°C | 4.12 | 4.53 | 4.92 | |
| | | 85°C | 4.27 | 4.63 | 4.98 | |
| | CPU running a Fibonacci algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | 25°C | 86*freq + 76 | 88*freq + 136 | 88*freq + 196 | µA (with freq in MHz) |
| | | 85°C | 88*freq + 156 | 88*freq + 230 | 88*freq + 302 | |
| | CPU running a CoreMark algorithm | 25°C | 5.78 | 6.32 | 6.80 | mA |
| | | 85°C | 5.93 | 6.47 | 7.00 | |
| | CPU running a CoreMark algorithm $V_{DDIN}$=1.8V, CPU is running on | 25°C | 5.17 | 5.60 | 5.96 | |
| | | 85°C | 5.35 | 5.73 | 6.10 | |

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | flash with 3 wait states | | | | | |
| | CPU running a CoreMark algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | 25°C | 106*freq + 78 | 106*freq + 136 | 108*freq + 196 | µA (with freq in MHz) |
| | | 85°C | 106*freq + 154 | 108*freq + 232 | 108*freq + 310 | |
| IDLE0 | Default operating conditions | 25°C | 1.89 | 2.04 | 2.20 | mA |
| | | 85°C | 1.98 | 2.14 | 2.33 | |
| IDLE1 | Default operating conditions | 25°C | 1.34 | 1.46 | 1.58 | |
| | | 85°C | 1.41 | 1.55 | 1.71 | |
| IDLE2 | Default operating conditions | 25°C | 1.07 | 1.17 | 1.28 | |
| | | 85°C | 1.13 | 1.27 | 1.40 | |
| STANDBY | XOSC32K running RTC running at 1kHz | 25°C | - | 4.06 | 12.8 | µA |
| | | 85°C | - | 55.2 | 100 | |
| | XOSC32K and RTC stopped | 25°C | - | 2.70 | 12.2 | |
| | | 85°C | - | 53.3 | 100 | |

**Table 37-8. Current Consumption (Device Variant B and L With Silicon Revision E)**

| Mode | Conditions | $T_A$ | $V_{CC}$ | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ACTIVE | CPU running a While 1 algorithm | 25°C | 3.3V | 3.7 | 3.9 | mA |
| | | 85°C | 3.3V | 3.8 | 4 | |
| | CPU running a While 1 algorithm | 25°C | 1.8V | 3.7 | 3.9 | |
| | | 85°C | 1,8V | 3.8 | 4 | |
| | CPU running a While 1 algorithm, with GCLKIN as reference | 25°C | 3.3V | 72*Freq+107 | 76*Freq+111 | µA (with freq in MHz) |
| | | 85°C | 3.3V | 72*Freq+198 | 76*Freq+210 | |
| | CPU running a Fibonacci algorithm | 25°C | 3.3V | 4.2 | 4.6 | mA |
| | | 85°C | 3.3V | 4.3 | 4.7 | |
| | CPU running a Fibonacci algorithm | 25°C | 1.8V | 4.2 | 4.6 | |
| | | 85°C | 1.8V | 4.3 | 4.7 | |
| | CPU running a Fibonacci algorithm, with GCLKIN as reference | 25°C | 3.3V | 83*Freq+107 | 87*Freq+111 | µA(with freq in MHz) |
| | | 85°C | 3.3V | 84*Freq+199 | 87*Freq+216 | |
| | CPU running a CoreMark algorithm | 25°C | 3.3V | 5.2 | 5.7 | |

| Mode | Conditions | $T_A$ | $V_{CC}$ | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | 85°C | 3.3V | 5.3 | 5.8 | |
| | CPU running a CoreMark algorithm | 25°C | 1.8V | 4.8 | 5.1 | |
| | | 85°C | 1.8V | 4.9 | 5.2 | |
| | CPU running a CoreMark algorithm, with GCLKIN as reference | 25°C | 3.3V | 104*Freq+109 | 108*Freq+113 | µA (with freq in MHz) |
| | | 85°C | 3.3V | 104*Freq+200 | 109*Freq+212 | |
| IDLE0 | Default operating conditions | 25°C | 3.3V | 2.4 | 2.5 | mA |
| | | 85°C | 3.3V | 2.5 | 2.6 | |
| IDLE1 | Default operating conditions | 25°C | 3.3V | 1.8 | 1.9 | |
| | | 85°C | 3.3V | 1.9 | 2.0 | |
| IDLE2 | Default operating conditions | 25°C | 3.3V | 1.3 | 1.4 | |
| | | 85°C | 3.3V | 1.4 | 1.5 | |
| STANDBY | XOSC32K running, RTC running at 1kHz | 25°C | 3.3V | 4 | 6.2 | µA |
| | | 85°C | 3.3V | 54 | 100 | |
| | XOSC32K and RTC stopped | 25°C | 3.3V | 2.8 | 5.0 | |
| | | 85°C | 3.3V | 52 | 98.8 | |

**Table 37-9. Current Consumption (Device Variant B, C, and L With Silicon Revision F)**

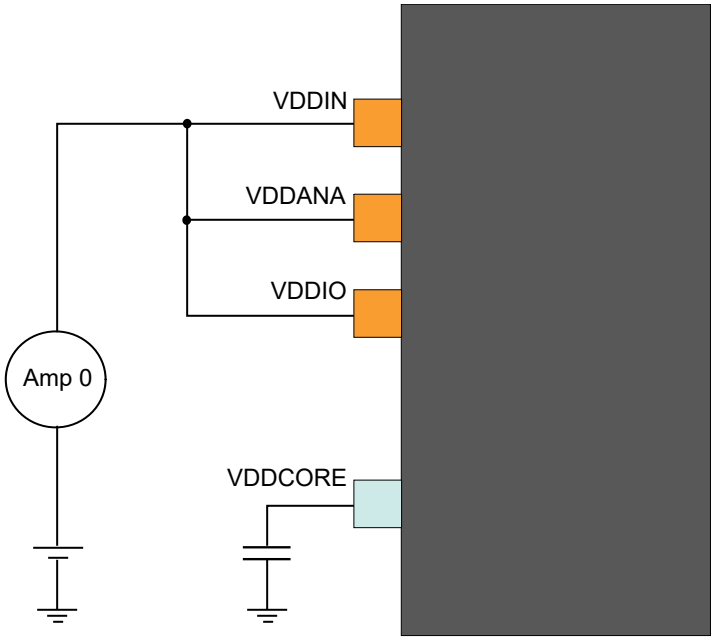
| Mode | Conditions | $T_A$ | $V_{CC}$ | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ACTIVE | CPU running a While 1 algorithm | 25°C | 3.3V | 3.7 | 3.9 | mA |
| | | 85°C | 3.3V | 3.8 | 4 | |
| | CPU running a While 1 algorithm | 25°C | 1.8V | 3.7 | 3.9 | |
| | | 85°C | 1,8V | 3.8 | 4 | |
| | CPU running a While 1 algorithm, with GCLKIN as reference | 25°C | 3.3V | 72*Freq+107 | 76*Freq+111 | µA (with freq in MHz) |
| | | 85°C | 3.3V | 72*Freq+198 | 76*Freq+210 | |
| | CPU running a Fibonacci algorithm | 25°C | 3.3V | 4.2 | 4.6 | mA |
| | | 85°C | 3.3V | 4.3 | 4.7 | |
| | CPU running a Fibonacci algorithm | 25°C | 1.8V | 4.2 | 4.6 | |
| | | 85°C | 1.8V | 4.3 | 4.7 | |
| | CPU running a Fibonacci algorithm, with GCLKIN as reference | 25°C | 3.3V | 83*Freq+107 | 87*Freq+111 | µA(with freq in MHz) |
| | | 85°C | 3.3V | 84*Freq+199 | 87*Freq+216 | |
| | CPU running a CoreMark algorithm | 25°C | 3.3V | 5.2 | 5.7 | |

| Mode | Conditions | $T_A$ | $V_{CC}$ | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | 85°C | 3.3V | 5.3 | 5.8 | |
| | CPU running a CoreMark algorithm | 25°C | 1.8V | 4.8 | 5.1 | |
| | | 85°C | 1.8V | 4.9 | 5.2 | |
| | CPU running a CoreMark algorithm, with GCLKIN as reference | 25°C | 3.3V | 104*Freq+109 | 108*Freq+113 | µA (with freq in MHz) |
| | | 85°C | 3.3V | 104*Freq+200 | 109*Freq+212 | |
| IDLE0 | Default operating conditions | 25°C | 3.3V | 2.4 | 2.5 | mA |
| | | 85°C | 3.3V | 2.5 | 2.6 | |
| IDLE1 | Default operating conditions | 25°C | 3.3V | 1.8 | 1.9 | |
| | | 85°C | 3.3V | 1.9 | 2 | |
| IDLE2 | Default operating conditions | 25°C | 3.3V | 1.3 | 1.4 | |
| | | 85°C | 3.3V | 1.4 | 1.5 | |
| STANDBY | XOSC32K running, RTC running at 1kHz | 25°C | 3.3V | 4.6 | 15.0 | µA |
| | | 85°C | 3.3V | 43 | 102.0 | |
| | XOSC32K and RTC stopped | 25°C | 3.3V | 3.4 | 14.0 | |
| | | 85°C | 3.3V | 42 | 100.0 | |

**Table 37-10. Wake-up Time**

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| IDLE0 | OSC8M used as main clock source, Cache disabled | 25°C | - | 4.0 | - | µs |
| | | 85°C | - | 4.0 | - | |
| IDLE1 | IOSC8M used as main clock source, Cache disabled | 25°C | - | 12.1 | - | |
| | | 85°C | - | 13.6 | - | |
| IDLE2 | IOSC8M used as main clock source, Cache disabled | 25°C | - | 13.0 | - | |
| | | 85°C | - | 14.5 | - | |
| STANDBY | IOSC8M used as main clock source, Cache disabled | 25°C | - | 19.6 | - | |
| | | 85°C | - | 19.7 | - | |

**Figure 37-1. Measurement Schematic**




## 37.7 Peripheral Power Consumption

Since USB peripheral complies with the Universal Serial Bus (USB) v2.0 standard, USB peripheral power consumption is described a specific section

### 37.7.1 All peripheral except USB

Default conditions, except where noted:

- Operating conditions
  - $V_{VDDIN}$ = 3.3 V
- Oscillators
  - XOSC (crystal oscillator) stopped
  - XOSC32K (32 kHz crystal oscillator) running with external 32kHz crystal
  - OSC8M at 8MHz
- Clocks
  - OSC8M used as main clock source
  - CPU, AHB and APBn clocks undivided
- The following AHB module clocks are running: NVMCTRL, HPB2 bridge, HPB1 bridge, HPB0 bridge
  - All other AHB clocks stopped
- The following peripheral clocks running: PM, SYSCTRL
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU in IDLE0 mode
- Cache enabled

- BOD33 disabled

In this default conditions, the power consumption $I_{default}$ is measured.

Operating mode for each peripheral in turn:

- Configure and enable the peripheral GCLK (When relevant, see conditions)
- Unmask the peripheral clock
- Enable the peripheral (when relevant)
- Set CPU in IDLE0 mode
- Measurement $I_{periph}$
- Wake-up CPU via EIC (async: level detection, filtering disabled)
- Disable the peripheral (when relevant)
- Mask the peripheral clock
- Disable the peripheral GCLK (when relevant, see conditions)

Each peripheral power consumption provided in table x.y is the value ($I_{periph}$ - $I_{default}$), using the same measurement method as for global power consumption measurement

**Table 37-11. Typical Peripheral Current Consumption**

| Peripheral | Conditions | Typ. | Units |
|---|---|---|---|
| RTC | $f_{GCLK_RTC}$ = 32kHz, 32bit counter mode | 7.4 | µA |
| WDT | $f_{GCLK_WDT}$=32kHz, normal mode with EW | 5.5 | µA |
| ACx | Both $f_{GCLK}$=8MHz, Enable both COMP | 31.3 | µA |
| TCx(1) | $f_{GCLK}$=8MHz, Enable + COUNTER in 8bit mode | 50 | µA |
| TCC2 | $f_{GCLK}$=8MHz, Enable + COUNTER | 95.5 | µA |
| TCC1 | $f_{GCLK}$=8MHz, Enable + COUNTER | 167.5 | µA |
| TCC0 | $f_{GCLK}$=8MHz, Enable + COUNTER | 180.3 | µA |
| SERCOMx.I2CM(2) | $f_{GCLK}$=8MHz, Enable | 69.7 | µA |
| SERCOMx.I2CS | $f_{GCLK}$=8MHz, Enable | 29.2 | µA |
| SERCOMx.SPI | $f_{GCLK}$=8MHz, Enable | 64.6 | µA |
| SERCOMx.USART | $f_{GCLK}$=8MHz, Enable | 65.5 | µA |
| I2S(3) | $f_{GCLK_I2S_0}$=12.288MHz with source FDPLL with $f_{FDPLL}$=49,152MHz | 26.4 | µA |
| DMAC(4) | RAM to RAM transfer | 399.5 | µA |

**Note:**

1. All TCs from 4 to 7 share the same power consumption values.
2. All SERCOMs from 0 to 5 share the same power consumption values.
3. The value includes the power consumption of the FDPLL.
4. The value includes the power consumption of the R/W access to the RAM.

### 37.7.2 USB Peripheral Power Consumption

Default conditions, except where noted:

- Operating conditions
  - $V_{VDDIN}$ = 3.3 V
- Oscillators
  - XOSC32K (32 kHz crystal oscillator) running with external 32kHz crystal in USB Host mode
- Clocks
  - USB Device mode: DFLL48M in USB recovery mode (Crystal less)
  - USB Host mode: DFLL48M in closed loop with XOSC32K (32 kHz crystal oscillator) running with external 32kHz crystal
  - CPU, AHB and APBn clocks undivided
- The following AHB module clocks are running: NVMCTRL, HPB2 bridge, HPB1 bridge, HPB0 bridge
  - All other AHB clocks stopped
- I/Os are inactive with internal pull-up
- CPU in IDLE0 mode
- Cache enabled
- BOD33 disabled

In this default conditions, the power consumption $I_{default}$ is measured.

Measurements do not include consumption of clock source (ex: DFLL48M or FDPLL96M) and CPU. However no CPU activity is required during all states (Suspend, IDLE, Data transfer).

Measurements have been done with an USB cable of 1.5m.

For USB Device mode, measurements include the maximum consumption (200µA) through pull-up resistor on the D+ line for USB attach. This value depends on USB Host characteristic.

Operating modes:

- Run the USB Device/Host states in regards of the Universal Serial Bus (USB) v2.0 standard.

USB power consumption is provided in the following tables.

**Table 37-12. Typical USB Device Full Speed mode Current Consumption**

| USB Device state | Conditions | Typ. | Units |
|---|---|---|---|
| Suspend | GCLK_USB is off, using USB wakeup asynchronous interrupt.<br>USB bus in suspend mode. | 201 | µA |
| Suspend | GCLK_USB is on.<br>USB bus in suspend mode. | 0.83 | mA |
| IDLE | Start Of Frame is running.<br>No packet transferred. | 1.17 | mA |
| Active OUT | Start Of Frame is running.<br>Bulk OUT on 100% bandwidth. | 2.17 | mA |
| Active IN | Start Of Frame is running.<br>Bulk IN on 100% bandwidth. | 10.3 | mA |

**Table 37-13. Typical USB Host Full Speed mode Current Consumption**

| USB Device state | Conditions | Typ. | Units |
|---|---|---|---|
| Wait connection | GCLK_USB is off, using USB wakeup asynchronous interrupt.<br>USB bus not connected. | 0.10 | µA |
| Wait connection | GCLK_USB is on.<br>USB bus not connected. | 0.19 | mA |
| Suspend | GCLK_USB is off, using USB wakeup asynchronous interrupt.<br>USB bus in suspend mode. | 201 | µA |
| Suspend | GCLK_USB is on.<br>USB bus in suspend mode. | 0.83 | mA |
| IDLE | Start Of Frame is running.<br>No packet transferred. | 1.17 | mA |
| Active OUT | Start Of Frame is running.<br>Bulk OUT on 100% bandwidth. | 2.17 | mA |
| Active IN | Start Of Frame is running.<br>Bulk IN on 100% bandwidth. | 10.3 | mA |

## 37.8 I/O Pin Characteristics

### 37.8.1 Normal I/O Pins

**Table 37-14. Normal I/O Pins Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $R_{PULL}$ | Pull-up - Pull-down resistance | All pins except for PA24 and PA25 | 20 | 40 | 60 | kΩ |
| $V_{IL}$ | Input low-level voltage | $V_{DD}$=1.62V-2.7V | - | - | 0.25*$V_{DD}$ | V |
| | | $V_{DD}$=2.7V-3.63V | - | - | 0.3*$V_{DD}$ | |
| $V_{IH}$ | Input high-level voltage | $V_{DD}$=1.62V-2.7V | 0.7*$V_{DD}$ | - | - | |
| | | $V_{DD}$=2.7V-3.63V | 0.55*$V_{DD}$ | - | - | |
| $V_{OL}$ | Output low-level voltage | $V_{DD}$>1.6V, $I_{OL}$ maxI | - | 0.1*$V_{DD}$ | 0.2*$V_{DD}$ | |
| $V_{OH}$ | Output high-level voltage | $V_{DD}$>1.6V, $I_{OH}$ maxII | 0.8*$V_{DD}$ | 0.9*$V_{DD}$ | - | |
| $I_{OL}$ | Output low-level current | $V_{DD}$=1.62V-3V, PORT.PINCFG.DRVSTR=0 | - | - | 1 | mA |
| | | $V_{DD}$=3V-3.63V, PORT.PINCFG.DRVSTR=0 | - | - | 2.5 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $V_{DD}$=1.62V-3V, PORT.PINCFG.DRVSTR=1 | - | - | 3 | |
| | | $V_{DD}$=3V-3.63V, PORT.PINCFG.DRVSTR=1 | - | - | 10 | |
| $I_{OH}$ | Output high-level current | $V_{DD}$=1.62V-3V, PORT.PINCFG.DRVSTR=0 | - | - | 0.70 | |
| | | $V_{DD}$=3V-3.63V, PORT.PINCFG.DRVSTR=0 | - | - | 2 | |
| | | $V_{DD}$=1.62V-3V, PORT.PINCFG.DRVSTR=1 | - | - | 2 | |
| | | $V_{DD}$=3V-3.63V, PORT.PINCFG.DRVSTR=1 | - | - | 7 | |
| $t_{RISE}$ | Rise time(1) | PORT.PINCFG.DRVSTR=0load = 5pF, $V_{DD}$ = 3.3V | - | - | 15 | ns |
| | | PORT.PINCFG.DRVSTR=1load = 20pF, $V_{DD}$ = 3.3V | - | - | 15 | |
| $t_{FALL}$ | Fall time(1) | PORT.PINCFG.DRVSTR=0load = 5pF, $V_{DD}$ = 3.3V | - | - | 15 | ns |
| | | PORT.PINCFG.DRVSTR=1load = 20pF, $V_{DD}$ = 3.3V | - | - | 15 | |
| $I_{LEAK}$ | Input leakage current | Pull-up resistors disabled | -1 | +/-0.015 | 1 | µA |

**Note:** 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

### 37.8.2 I2C Pins

Refer to *I/O Multiplexing and Considerations* to get the list of $I^2C$ pins.

**Table 37-15. $I^2C$ Pins Characteristics in $I^2C$ configuration**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $R_{PULL}$ | Pull-up - Pull-down resistance | I | 20 | 40 | 60 | kΩ |
| $V_{IL}$ | Input low-level voltage | $V_{DD}$=1.62V-2.7VI | - | - | 0.25*$V_{DD}$ | V |
| | | $V_{DD}$=2.7V-3.63V | - | - | 0.3*$V_{DD}$ | |
| $V_{IH}$ | Input high-level voltage | $V_{DD}$=1.62V-2.7V | 0.7*$V_{DD}$ | - | - | |
| | | $V_{DD}$=2.7V-3.63VI | 0.55*$V_{DD}$ | - | - | |
| $V_{HYS}$ | Hysteresis of Schmitt trigger inputs | | 0.08*$V_{DD}$ | - | - | |
| $V_{OL}$ | Output low-level voltage | $V_{DD}$> 2.0V, | - | - | 0.4 | |

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $I_{OL}$=3mA | | | | |
| | | $V_{DD}$≤2.0V , $I_{OL}$=2mA | - | - | 0.2*$V_{DD}$ | |
| $I_{OL}$ | Output low-level current | $V_{OL}$ =0.4V Standard, Fast and HS Modes | 3 | | | mA |
| | | $V_{OL}$ =0.4V Fast Mode + | 20 | - | - | |
| | | $V_{OL}$ =0.6V | 6 | - | - | |
| $f_{SCL}$ | SCL clock frequency | I | - | - | 3.4 | MHz |

$I^2C$ pins timing characteristics can be found in 37.15.3 SERCOM in I2C Mode Timing.

### 37.8.3 XOSC Pin

XOSC pins behave as normal pins when used as normal I/Os. Refer to table 37.8.1 Normal I/O Pins.

### 37.8.4 XOSC32 Pin

XOSC32 pins behave as normal pins when used as normal I/Os. Refer to table 37.8.1 Normal I/O Pins.

### 37.8.5 External Reset Pin

Reset pin has the same electrical characteristics as normal I/O pins. Refer to table 37.8.1 Normal I/O Pins.

## 37.9 Injection Current

Stresses beyond those listed in the table below may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**Table 37-16. Injection Current(1)**

| Symbol | Description | min | max | Unit |
|---|---|---|---|---|
| $I_{inj1}$ (2) | IO pin injection current | -1 | +1 | mA |
| $I_{inj2}$ (3) | IO pin injection current | -15 | +15 | mA |
| $I_{injtotal}$ | Sum of IO pins injection current | -45 | +45 | mA |

1. Injecting current may have an effect on the accuracy of Analog blocks
2. Conditions for $V_{pin}$: $V_{pin}$ < GND-0.6V or 3.6V<$V_{pin}$≤4.2V.

   Conditions for $V_{DD}$: 3V<$V_{DD}$≤3.6V.

   If $V_{pin}$ is lower than GND-0.6V, a current limiting resistor is required. The negative DC injection current limiting resistor *R* is calculated as $R = |(GND-0.6V - V_{pin})/I_{inj1}|$.

If $V_{pin}$ is greater than $V_{DD}$+0.6V, a current limiting resistor is required. The positive DC injection current limiting resistor *R* is calculated as $R = (V_{pin}-(V_{DD}+0.6))/I_{inj1}$.

3. Conditions for $V_{pin}$: $V_{pin}$ < GND-0.6V or $V_{pin}$≤3.6V.

   Conditions for $V_{DD}$: $V_{DD}$≤3V.

   If $V_{pin}$ is lower than GND-0.6V, a current limiting resistor is required. The negative DC injection current limiting resistor *R* is calculated as $R = |(GND-0.6V - V_{pin})/I_{inj2}|$.

   If $V_{pin}$ is greater than $V_{DD}$+0.6V, a current limiting resistor is required. The positive DC injection current limiting resistor *R* is calculated as $R = (V_{pin}-(V_{DD}+0.6))/I_{inj2}$.

## 37.10 Analog Characteristics

### 37.10.1 Voltage Regulator Characteristics

**Table 37-17. Voltage Regulator Electrical Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DDCORE}$ | DC calibrated output voltage | Voltage regulator normal mode | 1.1 | 1.23 | 1.30 | V |

**Note:** Supplying any external components using the $V_{DDCORE}$ pin is not allowed to assure the integrity of the core supply voltage.

**Table 37-18. Decoupling requirements**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $C_{IN}$(1) | Input regulator capacitor, between $V_{DDIN}$ and GND | | - | 10 | - | µF |
| $C_{OUT}$(2) | Output regulator capacitor, between $V_{DDCORE}$ and GND | | 0.8 | 1 | - | µF |

**Note:** 1. It is recommended to use ceramic X7R capacitor with low-series resistance. Refer to Figure 41-1 for a typical circuit connections of the decoupling capacitor.

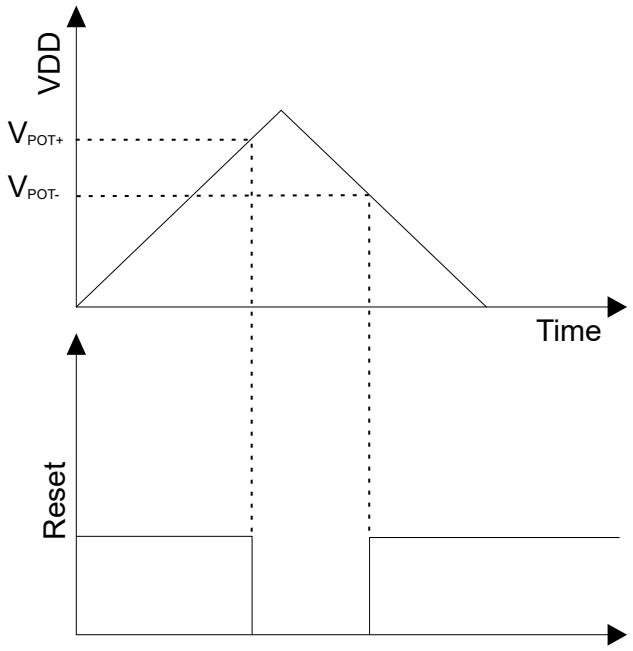
**Note:** 2. It is recommended to use ceramic or solid tantalum capacitor with low ESR <= 1 ohms.

### 37.10.2 Power-On Reset (POR) Characteristics

**Table 37-19. POR Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{POT+}$ | Voltage threshold on $V_{DD}$ rising | $V_{DD}$ falls at 1V/ms or slower | 1.27 | 1.45 | 1.58 | V |
| $V_{POT-}$ | Voltage threshold on $V_{DD}$ falling | | 0.72 | 0.99 | 1.32 | V |

**Figure 37-2. POR Operating Principle**




### 37.10.3 Brown-Out Detectors Characteristics

#### 37.10.3.1 BOD33

**Figure 37-3. BOD33 Hysteresis OFF**

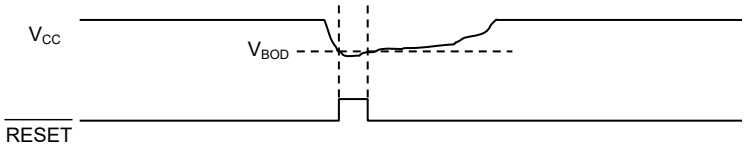



**Figure 37-4. BOD33 Hysteresis ON**

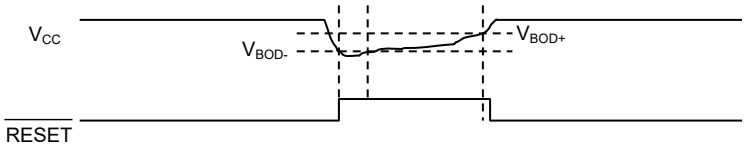



**Table 37-20. BOD33 LEVEL Value**

| Symbol | BOD33.LEVEL | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{BOD+}$ | 6 | Hysteresis ON | - | 1.715 | 1.745 | V |
| | 7 | | - | 1.750 | 1.779 | |
| | 39 | | - | 2.84 | 2.92 | |
| | 48 | | - | 3.2 | 3.3 | |
| $V_{BOD-}$ | 6 | Hysteresis ON | 1.62 | 1.64 | 1.67 | |

| Symbol | BOD33.LEVEL | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| or | 7 | or | 1.64 | 1.675 | 1.71 | |
| $V_{BOD}$ | 39 | Hysteresis OFF | 2.72 | 2.77 | 2.81 | |
| | 48 | | 3.0 | 3.07 | 3.2 | |

**Note:** See chapter Memories table *NVM User Row Mapping* for the BOD33 default value settings.

**Table 37-21. BOD33 Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Step size, between adjacent values in BOD33.LEVEL | - | - | 34 | - | mV |
| $V_{HYST}$ | $V_{BOD+}$ - $V_{BOD-}$ | Hysteresis ON | 35 | - | 170 | mV |
| $t_{DET}$ | Detection time | Time with $V_{DDANA} < V_{TH}$ necessary to generate a reset signal | - | 0.9(1) | - | µs |
| $I_{BOD33}$ | Current consumption | Idle2,Continuous mode, VDD=3V | - | 25 | 50 | µA |
| | | Idle2, Sampling mode, VDD=1.8V | - | 0.034 | 1.62 | |
| | | STANDBY,Sample Mode, VDD=3.3V | - | 0.132 | 1 | |
| $t_{STARTUP}$ | Startup time | - | - | 2.2(1) | - | µs |

**Note:** 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

**Related Links**

10.3.1 NVM User Row Mapping

### 37.10.4 Analog-to-Digital (ADC) characteristics

**Table 37-22. Operating Conditions**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| RES | Resolution | | 8 | - | 12 | bits |
| $f_{CLK_ADC}$ | ADC Clock frequency | | 30 | - | 2100 | kHz |
| | Sample rate(1) | Single shot | 5 | - | 300 | ksps |
| | | Free running | 5 | - | 350 | ksps |
| | Sampling time(1) | | 0.5 | - | - | cycles |
| | Conversion time(1) | 1x Gain | 6 | - | - | cycles |
| $V_{REF}$ | Voltage reference range | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| $V_{REFINT1V}$ | Internal 1V reference (2) | | - | 1.0 | - | V |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{REFINTVCC0}$ | Internal ratiometric reference 0(2) | | - | $V_{DDANA}$/ 1.48 | - | V |
| $V_{REFINTVCC0}$ Voltage Error | Internal ratiometric reference 0(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| $V_{REFINTVCC1}$ | Internal ratiometric reference 1(2) | $V_{DDANA}$>2.0V | - | $V_{DDANA}$/2 | - | V |
| $V_{REFINTVCC1}$ Voltage Error | Internal ratiometric reference 1(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| | Conversion range(1) | Differential mode | -$V_{REF}$/ GAIN | - | +$V_{REF}$/GAIN | V |
| | | Single-ended mode | 0.0 | - | +$V_{REF}$/GAIN | V |
| $C_{SAMPLE}$ | Sampling capacitance(2) | | - | 3.5 | - | pF |
| $R_{SAMPLE}$ | Input channel source resistance(2) | | - | - | 3.5 | kΩ |
| $I_{DD}$ | DC supply current(1) | $f_{CLK_ADC}$ = 2.1MHz(3) | - | 1.25 | 1.79 | mA |

**Note:**

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.
3. In this condition and for a sample rate of 350ksps, 1 Conversion at gain 1x takes 6 clock cycles of the ADC clock.

**Table 37-23. Differential Mode (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number Of Bits | With gain compensation | - | 10.5 | 11.1 | bits |
| TUE | Total Unadjusted Error | 1x gain | 1.5 | 4.3 | 15.0 | LSB |
| INLI | Integral Non Linearity | 1x gain | 1.0 | 1.3 | 4.5 | LSB |
| DNL | Differential Non Linearity | 1x gain | +/-0.3 | +/-0.5 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref 1x | -10.0 | 2.5 | +10.0 | mV |
| | | $V_{REF}$=$V_{DDANA}$/1.48 | -15.0 | -1.5 | +10.0 | mV |
| | | Bandgap | -20.0 | -5.0 | +20.0 | mV |
| | Gain Accuracy(5) | Ext. Ref. 0.5x | +/-0.1 | +/-0.2 | +/-0.45 | % |
| | | Ext. Ref. 2x to 16x | +/-0.05 | +/-0.1 | +/-0.11 | % |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Offset Error | Ext. Ref. 1x | -5.0 | -1.5 | +5.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -5.0 | 0.5 | +5.0 | mV |
| | | Bandgap | -5.0 | 3.0 | +5.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 62.7 | 70.0 | 75.0 | dB |
| SINAD | Signal-to-Noise and Distortion | | 54.1 | 65.0 | 68.5 | dB |
| SNR | Signal-to-Noise Ratio | | 54.5 | 65.5 | 68.6 | dB |
| THD | Total Harmonic Distortion | | -77.0 | -64.0 | -63.0 | dB |
| | Noise RMS | T=25°C | 0.6 | 1.0 | 1.6 | mV |

**Table 37-24. Differential Mode (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number Of Bits | With gain compensation | - | 10.5 | 10.8 | bits |
| TUE | Total Unadjusted Error | 1x gain | 1.5 | 2.7 | 10.0 | LSB |
| INLI | Integral Non Linearity | 1x gain | 0.9 | 1.3 | 4.0 | LSB |
| DNL | Differential Non Linearity | 1x gain | +/-0.3 | +/-0.5 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref 1x | -10.0 | 1.3 | +10.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -10.0 | -1.0 | +10.0 | mV |
| | | Bandgap | -20.0 | -2.0 | +20.0 | mV |
| | Gain Accuracy(5) | Ext. Ref. 0.5x | +/-0.02 | +/-0.05 | +/-0.1 | % |
| | | Ext. Ref. 2x to 16x | +/-0.01 | +/-0.03 | +/-0.5 | % |
| | Offset Error | Ext. Ref. 1x | -5.0 | -1.0 | +5.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -5.0 | 0.6 | +5.0 | mV |
| | | Bandgap | -5.0 | -1.0 | +5.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 65 | 71.3 | 77.0 | dB |
| SINAD | Signal-to-Noise and Distortion | | 58 | 65 | 67 | dB |
| SNR | Signal-to-Noise Ratio | | 60 | 66 | 68.6 | dB |
| THD | Total Harmonic Distortion | | -75.0 | -71.0 | -67.0 | dB |
| | Noise RMS | T=25°C | 0.6 | 1.0 | 1.6 | mV |

**Note:**

1. Maximum numbers are based on characterization and not tested in production, and valid for 5% to 95% of the input voltage range.
2. Dynamic parameter numbers are based on characterization and not tested in production.
3. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage):

– 1.1. If |VIN| > VREF/4
  – VCM_IN < 0.95*VDDANA + VREF/4 – 0.75V
  – VCM_IN > VREF/4 -0.05*VDDANA -0.1V
– 1.2. If |VIN| < VREF/4
  – VCM_IN < 1.2*VDDANA - 0.75V
  – VCM_IN > 0.2*VDDANA - 0.1V

4. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the VDDIO power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
5. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / (2*Vref/GAIN)

**Table 37-25. Single-Ended Mode (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number of Bits | With gain compensation | - | 9.5 | 9.8 | Bits |
| TUE | Total Unadjusted Error | 1x gain | - | 10.5 | 14.0 | LSB |
| INL | Integral Non-Linearity | 1x gain | 1.0 | 1.6 | 3.5 | LSB |
| DNL | Differential Non-Linearity | 1x gain | +/-0.5 | +/-0.6 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref. 1x | -5.0 | 0.7 | +5.0 | mV |
| | Gain Accuracy(4) | Ext. Ref. 0.5x | +/-0.2 | +/-0.34 | +/-0.4 | % |
| | | Ext. Ref. 2x to 16X | +/-0.01 | +/-0.1 | +/-0.2 | % |
| | Offset Error | Ext. Ref. 1x | -5.0 | 1.5 | +5.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 63.1 | 65.0 | 67.0 | dB |
| SINAD | Signal-to-Noise and Distortion | | 47.5 | 59.5 | 61.0 | dB |
| SNR | Signal-to-Noise Ratio | | 48.0 | 60.0 | 64.0 | dB |
| THD | Total Harmonic Distortion | | -65.4 | -63.0 | -62.1 | dB |
| | Noise RMS | T = 25°C | - | 1.0 | - | mV |

**Table 37-26. Single-Ended Mode (Device Variant B,C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number of Bits | With gain compensation | - | 9.7 | 10.1 | Bits |
| TUE | Total Unadjusted Error | 1x gain | - | 7.9 | 15.0 | LSB |
| INL | Integral Non-Linearity | 1x gain | 1.4 | 2.6 | 4.5 | LSB |
| DNL | Differential Non-Linearity | 1x gain | +/-0.6 | +/-0.7 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref. 1x | -5.0 | 0.6 | +5.0 | mV |
| | Gain Accuracy(4) | Ext. Ref. 0.5x | +/-0.2 | +/-0.37 | +/-0.55 | % |
| | | Ext. Ref. 2x to 16X | +/-0.01 | +/-0.1 | +/-0.2 | % |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Offset Error | Ext. Ref. 1x | -5.0 | 0.6 | +5.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 63.0 | 68.0 | 71.0 | dB |
| SINAD | Signal-to-Noise and Distortion | | 55.0 | 60.1 | 63.0 | dB |
| SNR | Signal-to-Noise Ratio | | 54.0 | 61.0 | 64.0 | dB |
| THD | Total Harmonic Distortion | | -70.8 | -68.0 | -65.0 | dB |
| | Noise RMS | T = 25°C | - | 1.0 | - | mV |

**Note:**

1. Maximum numbers are based on characterization and not tested in production, and for 5% to 95% of the input voltage range.
2. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage) for all VIN:
   - VCM_IN < 0.7*VDDANA + VREF/4 – 0.75V
   - VCM_IN > VREF/4 – 0.3*VDDANA - 0.1V
3. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the VDDIO power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
4. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / (Vref/GAIN)

#### 37.10.4.1 Performance with the Averaging Digital Feature

Averaging is a feature which increases the sample accuracy. ADC automatically computes an average value of multiple consecutive conversions. The numbers of samples to be averaged is specified by the Number-of-Samples-to-be-collected bit group in the Average Control register (AVGCTRL.SAMPLENUM[3:0]) and the averaged output is available in the Result register (RESULT).

**Table 37-27. Averaging Feature (Device Variant A)**

| Average Number | Conditions | SNR (dB) | SINAD (dB) | SFDR (dB) | ENOB (bits) |
|---|---|---|---|---|---|
| 1 | In differential mode, 1x gain, $V_{DDANA}$=3.0V, $V_{REF}$=1.0V, 350kSps at 25°C | 66.0 | 65.0 | 72.8 | 9.75 |
| 8 | | 67.6 | 65.8 | 75.1 | 10.62 |
| 32 | | 69.7 | 67.1 | 75.3 | 10.85 |
| 128 | | 70.4 | 67.5 | 75.5 | 10.91 |

**Table 37-28. Averaging Feature (Device Variant B,C and L)**

| Average Number | Conditions | SNR (dB) | SINAD (dB) | SFDR (dB) | ENOB (bits) |
|---|---|---|---|---|---|
| 1 | In differential mode, 1x gain, $V_{DDANA}$=3.0V, $V_{REF}$=1.0V, 350kSps at 25°C | 66.0 | 65.0 | 72.8 | 10.5 |
| 8 | | 67.6 | 65.8 | 75.1 | 10.62 |

| Average Number | Conditions | SNR (dB) | SINAD (dB) | SFDR (dB) | ENOB (bits) |
|---|---|---|---|---|---|
| 32 | | 69.7 | 67.1 | 75.3 | 10.85 |
| 128 | | 70.4 | 67.5 | 75.5 | 10.91 |

#### 37.10.4.2 Performance with the hardware offset and gain correction

Inherent gain and offset errors affect the absolute accuracy of the ADC. The offset error cancellation is handled by the Offset Correction register (OFFSETCORR) and the gain error cancellation, by the Gain Correction register (GAINCORR). The offset and gain correction value is subtracted from the converted data before writing the Result register (RESULT).

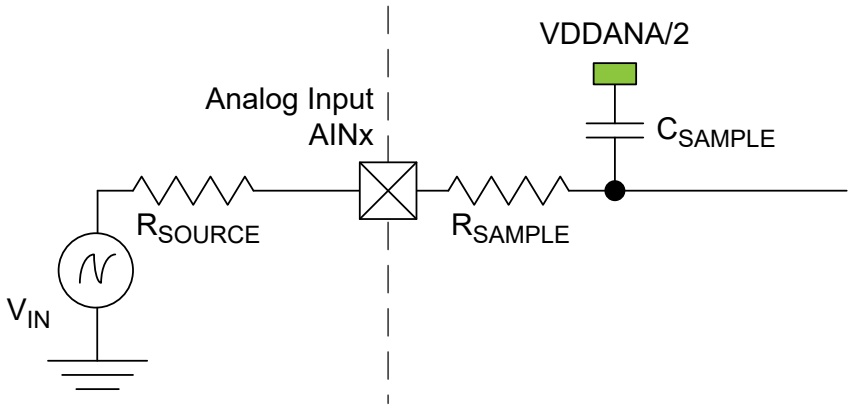
**Table 37-29. Offset and Gain correction feature**

| Gain Factor | Conditions | Offset Error (mV) | Gain Error (mV) | Total Unadjusted Error (LSB) |
|---|---|---|---|---|
| 0.5x | In differential mode, 1x gain, $V_{DDANA}$=3.0V, $V_{REF}$=1.0V, 350kSps at 25°C | 0.25 | 1.0 | 2.4 |
| 1x | | 0.20 | 0.10 | 1.5 |
| 2x | | 0.15 | -0.15 | 2.7 |
| 8x | | -0.05 | 0.05 | 3.2 |
| 16x | | 0.10 | -0.05 | 6.1 |

#### 37.10.4.3 Inputs and Sample and Hold Acquisition Times

The analog voltage source must be able to charge the sample and hold (S/H) capacitor in the ADC in order to achieve maximum accuracy. Seen externally, the ADC input consists of a resistor ($R_{\mathrm{SAMPLE}}$) and a capacitor ($C_{\mathrm{SAMPLE}}$). In addition, the source resistance ($R_{\mathrm{SOURCE}}$) must be taken into account when calculating the required sample and hold time. The next figure shows the ADC input channel equivalent circuit.

**Figure 37-5. ADC Input**




To achieve n bits of accuracy, the $C_{\mathrm{SAMPLE}}$ capacitor must be charged at least to a voltage of

$$V_{\mathrm{CSAMPLE}} \geq V_{\mathrm{IN}} \times \left(1 + -2^{-(n+1)}\right)$$

The minimum sampling time $t_{\mathrm{SAMPLEHOLD}}$ for a given $R_{\mathrm{SOURCE}}$can be found using this formula:

$t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times (n+1) \times \ln(2)$

for a 12 bits accuracy: $t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times 9.02$

where

$t_{SAMPLEHOLD} = \frac{1}{2 \times f_{ADC}}$

### 37.10.5 Digital to Analog Converter (DAC) Characteristics

**Table 37-30. Operating Conditions(1)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DDANA}$ | Analog supply voltage | | 1.62 | - | 3.63 | V |
| $AV_{REF}$ | External reference voltage | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| | Internal reference voltage 1 | | - | 1 | - | V |
| | Internal reference voltage 2 | | - | $V_{DDANA}$ | - | V |
| | Linear output voltage range | | 0.05 | - | $V_{DDANA}$-0.05 | V |
| | Minimum resistive load | | 5 | - | - | kΩ |
| | Maximum capacitance load | | - | - | 100 | pF |
| $I_{DD}$ | DC supply current(2) | Voltage pump disabled | - | 160 | 230 | µA |

1. These values are based on specifications otherwise noted.

2. These values are based on characterization. These values are not covered by test limits in production.

**Table 37-31. Clock and Timing(1)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | Conversion rate | $C_{load}$=100pF $R_{load}$ > 5kΩ | Normal mode | - | - | 350 | ksps |
| | | | For $\Delta_{DATA}$=+/-1 | - | - | 1000 | |
| | Startup time | $V_{DDNA}$ > 2.6V | | - | - | 2.85 | µs |
| | | $V_{DDNA}$ < 2.6V | | - | - | 10 | µs |

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

**Table 37-32. Accuracy Characteristics(1) (Device Variant A)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| RES | Input resolution | | | - | - | 10 | Bits |
| INL | Integral non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | 0.75 | 1.1 | 2.5 | LSB |
| | | | $V_{DD}$ = 3.6V | 0.6 | 1.2 | 1.5 | |
| | | $V_{REF}$ = $V_{DDANA}$ | $V_{DD}$ = 1.6V | 1.4 | 2.2 | 2.5 | |
| | | | $V_{DD}$ = 3.6V | 0.9 | 1.4 | 1.5 | |

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | 0.75 | 1.3 | 1.5 | |
| | | | $V_{DD}$ = 3.6V | 0.8 | 1.2 | 1.5 | |
| DNL | Differential non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | +/-0.9 | +/-1.2 | +/-1.5 | LSB |
| | | | $V_{DD}$ = 3.6V | +/-0.9 | +/-1.1 | +/-1.2 | |
| | | $V_{REF}$= $V_{DDANA}$ | $V_{DD}$ = 1.6V | +/-1.1 | +/-1.5 | +/-1.7 | |
| | | | $V_{DD}$ = 3.6V | +/-1.0 | +/-1.1 | +/-1.2 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | +/-1.1 | +/-1.4 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-1.0 | +/-1.5 | +/-1.6 | |
| | Gain error | Ext. $V_{REF}$ | | +/-1.5 | +/-5 | +/-10 | mV |
| | Offset error | Ext. $V_{REF}$ | | +/-2 | +/-3 | +/-6 | mV |

**Table 37-33. Accuracy Characteristics(1) (Device Variant B,C and L)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| RES | Input resolution | | | - | - | 10 | Bits |
| INL | Integral non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | 0.7 | 0.75 | 2 | LSB |
| | | | $V_{DD}$ = 3.6V | 0.6 | 0.65 | 1.5 | |
| | | $V_{REF}$ = $V_{DDANA}$ | $V_{DD}$ = 1.6V | 0.6 | 0.85 | 2 | |
| | | | $V_{DD}$ = 3.6V | 0.5 | 0.8 | 1.5 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | 0.5 | 0.75 | 1.5 | |
| | | | $V_{DD}$ = 3.6V | 0.7 | 0.8 | 1.5 | |
| DNL | Differential non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | +/-0.3 | +/-0.4 | +/-1.0 | LSB |
| | | | $V_{DD}$ = 3.6V | +/-0.25 | +/-0.4 | +/-0.75 | |
| | | $V_{REF}$= $V_{DDANA}$ | $V_{DD}$ = 1.6V | +/-0.4 | +/-0.55 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.2 | +/-0.3 | +/-0.75 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | +/-0.5 | +/-0.7 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.4 | +/-0.7 | +/-1.5 | |
| | Gain error | Ext. $V_{REF}$ | | +/-0.5 | +/-5 | +/-10 | mV |
| | Offset error | Ext. $V_{REF}$ | | +/-2 | +/-1.5 | +/-8 | mV |

Note: 1. All values measured using a conversion rate of 350ksps.

### 37.10.6 Analog Comparator Characteristics

**Table 37-34. Electrical and Timing (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Positive input voltage range | | 0 | - | $V_{DDANA}$ | V |
| | Negative input voltage range | | 0 | - | $V_{DDANA}$ | |
| | Offset | Hysteresis = 0, Fast mode | -15 | 0.0 | +15 | mV |
| | | Hysteresis = 0, Low power mode | -25 | 0.0 | +25 | mV |
| | Hysteresis | Hysteresis = 1, Fast mode | 20 | 50 | 80 | mV |
| | | Hysteresis = 1, Low power mode | 15 | 40 | 75 | mV |
| | Propagation delay | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Fast mode | - | 60 | 116 | ns |
| | | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Low power mode | - | 225 | 370 | ns |
| $t_{STARTUP}$ | Startup time | Enable to ready delay Fast mode | - | 1 | 2 | µs |
| | | Enable to ready delay Low power mode | - | 12 | 19 | µs |
| $V_{SCALE}$ | INL(3) | | -1.4 | 0.75 | +1.4 | LSB |
| | DNL(3) | | -0.9 | 0.25 | +0.9 | LSB |
| | Offset Error (1)(2) | | -0.200 | 0.260 | +0.920 | LSB |
| | Gain Error (1)(2) | | -0.89 | 0.215 | 0.89 | LSB |

**Table 37-35. Electrical and Timing (Device Variant B,C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Positive input voltage range | | 0 | - | $V_{DDANA}$ | V |
| | Negative input voltage range | | 0 | - | $V_{DDANA}$ | |
| | Offset | Hysteresis = 0, Fast mode | -15 | 0.0 | +15 | mV |
| | | Hysteresis = 0, Low power mode | -25 | 0.0 | +25 | mV |
| | Hysteresis | Hysteresis = 1, Fast mode | 20 | 50 | 85 | mV |
| | | Hysteresis = 1, Low power mode | 15 | 40 | 75 | mV |
| | Propagation delay | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Fast mode | - | 90 | 180 | ns |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | Changes for $V_{ACM}$=$V_{DDANA}$/2 100mV overdrive, Low power mode | - | 282 | 520 | ns |
| $t_{STARTUP}$ | Startup time | Enable to ready delay Fast mode | - | 1 | 2.6 | µs |
| | | Enable to ready delay Low power mode | - | 14 | 22 | µs |
| $V_{SCALE}$ | INL(3) | | -1.4 | 0.75 | +1.4 | LSB |
| | DNL(3) | | -0.9 | 0.25 | +0.9 | LSB |
| | Offset Error (1)(2) | | -0.200 | 0.260 | +0.920 | LSB |
| | Gain Error (1)(2) | | -0.89 | 0.215 | 0.89 | LSB |

**Note:**

1. According to the standard equation V(X)=$V_{LSB}$*(X+1); $V_{LSB}$=$V_{DDANA}$/64
2. Data computed with the Best Fit method
3. Data computed using histogram

### 37.10.7 Internal 1.1V Bandgap Reference Characteristics

**Table 37-36. Bandgap and Internal 1.1V reference characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| INT1V | Internal 1.1V Bandgap reference | After calibration at T= 25°C, over [-40, +85]C | 1.08 | 1.1 | 1.12 | V |
| | | Over voltage at 25°C | 1.09 | 1.1 | 1.11 | V |

### 37.10.8 Temperature Sensor Characteristics

#### 37.10.8.1 Temperature Sensor Characteristics

**Table 37-37. Temperature Sensor Characteristics(1) (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Temperature sensor output voltage | T= 25°C, $V_{DDANA}$ = 3.3V | - | 0.667 | - | V |
| | Temperature sensor slope | | 2.3 | 2.4 | 2.5 | mV/°C |
| | Variation over $V_{DDANA}$ voltage | $V_{DDANA}$=1.62V to 3.6V | -1.7 | 1 | 3.7 | mV/V |
| | Temperature Sensor accuracy | Using the method described in the 37.10.8.2 Software-based Refinement of the Actual Temperature | -10 | - | 10 | °C |

**Table 37-38. Temperature Sensor Characteristics(1) (Device Variant B,C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Temperature sensor output voltage | T= 25°C, $V_{DDANA}$ = 3.3V | - | 0.688 | - | V |
| | Temperature sensor slope | | 2.06 | 2.16 | 2.26 | mV/°C |
| | Variation over $V_{DDANA}$ voltage | $V_{DDANA}$=1.62V to 3.6V | -0.4 | 1.4 | 3 | mV/V |
| | Temperature Sensor accuracy | Using the method described in the 37.10.8.2 Software-based Refinement of the Actual Temperature | -10 | - | 10 | °C |

**Note:** 1. These values are based on characterization. These values are not covered by test limits in production.

### 37.10.8.2 Software-based Refinement of the Actual Temperature

The temperature sensor behavior is linear but depends on several parameters such as the internal voltage reference, which itself depends on the temperature. To take this into account, each device contains a Temperature Log row with data measured and written during the production tests. These calibration values should be read by software to infer the most accurate temperature readings possible.

This Software Temperature Log row can be read at address 0x00806030

This section specifies the Temperature Log row content and explains how to refine the temperature sensor output using the values in the Temperature Log row.

#### 37.10.8.2.1 Temperature Log Row

All values in this row were measured in the following conditions:

- $V_{DDIN}$ = $V_{DDIO}$ = $V_{DDANA}$ = 3.3V
- ADC Clock speed = 1MHz
- ADC mode: Free running mode, ADC averaging mode with 4 averaged samples
- ADC voltage reference = 1.0V internal reference (INT1V)
- ADC input = Temperature sensor

**Table 37-39. Temperature Log Row Content**

| Bit position | Name | Description |
|---|---|---|
| 7:0 | ROOM_TEMP_VAL_INT | Integer part of room temperature in °C |
| 11:8 | ROOM_TEMP_VAL_DEC | Decimal part of room temperature |
| 19:12 | HOT_TEMP_VAL_INT | Integer part of hot temperature in °C |
| 23:20 | HOT_TEMP_VAL_DEC | Decimal part of hot temperature |
| 31:24 | ROOM_INT1V_VAL | 2's complement of the internal 1V reference drift at room temperature (versus a 1.0 centered value) |
| 39:32 | HOT_INT1V_VAL | 2's complement of the internal 1V reference drift at hot temperature (versus a 1.0 centered value) |

| Bit position | Name | Description |
|---|---|---|
| 51:40 | ROOM_ADC_VAL | 12-bit ADC conversion at room temperature |
| 63:52 | HOT_ADC_VAL | 12-bit ADC conversion at hot temperature |

The temperature sensor values are logged during test production flow for Room and Hot insertions:

- ROOM_TEMP_VAL_INT and ROOM_TEMP_VAL_DEC contains the measured temperature at room insertion (e.g. for ROOM_TEMP_VAL_INT=25 and ROOM_TEMP_VAL_DEC=2, the measured temperature at room insertion is 25.2°C).
- HOT_TEMP_VAL_INT and HOT_TEMP_VAL_DEC contains the measured temperature at hot insertion (e.g. for HOT_TEMP_VAL_INT=83 and HOT_TEMP_VAL_DEC=3, the measured temperature at room insertion is 83.3°C).

The temperature log row also contains the corresponding 12-bit ADC conversions of both Room and Hot temperatures:

- ROOM_ADC_VAL contains the 12-bit ADC value corresponding to (ROOM_TEMP_VAL_INT, ROOM_TEMP_VAL_DEC)
- HOT_ADC_VAL contains the 12-bit ADC value corresponding to (HOT_TEMP_VAL_INT, HOT_TEMP_VAL_DEC)

The temperature log row also contains the corresponding 1V internal reference of both Room and Hot temperatures:

- ROOM_INT1V_VAL is the 2's complement of the internal 1V reference value corresponding to (ROOM_TEMP_VAL_INT, ROOM_TEMP_VAL_DEC)
- HOT_INT1V_VAL is the 2's complement of the internal 1V reference value corresponding to (HOT_TEMP_VAL_INT, HOT_TEMP_VAL_DEC)
- ROOM_INT1V_VAL and HOT_INT1V_VAL values are centered around 1V with a 0.001V step. In other words, the range of values [0,127] corresponds to [1V, 0.873V] and the range of values [-1, -127] corresponds to [1.001V, 1.127V]. INT1V == 1 - (VAL/1000) is valid for both ranges.

#### 37.10.8.2.2 Using Linear Interpolation

For concise equations, we will use the following notations:

- (ROOM_TEMP_VAL_INT, ROOM_TEMP_VAL_DEC) is denoted $temp_R$
- (HOT_TEMP_VAL_INT, HOT_TEMP_VAL_DEC) is denoted $temp_H$
- ROOM_ADC_VAL is denoted $ADC_R$, its conversion to Volt is denoted $V_{ADCR}$
- HOT_ADC_VAL is denoted $ADC_H$, its conversion to Volt is denoted $V_{ADCH}$
- ROOM_INT1V_VAL is denoted $INT1V_R$
- HOT_INT1V_VAL is denoted $INT1V_H$

Using the ($temp_R$, $ADC_R$) and ($temp_H$, $ADC_H$) points, using a linear interpolation we have the following equation:

$$\left(\frac{V_{\mathrm{ADC}} + \ - V_{\mathrm{ADCR}}}{\mathrm{temp} + \ - \mathrm{temp}_R}\right) = \left(\frac{V_{\mathrm{ADCH}} + \ - V_{\mathrm{ADCR}}}{\mathrm{temp}_H + \ - \mathrm{temp}_R}\right)$$

Given a temperature sensor ADC conversion value $ADC_m$, we can infer a coarse value of the temperature $temp_C$ as:

$$\mathrm{temp}_C = \mathrm{temp}_R + \left[\frac{\left\{\left(\mathrm{ADC}_m \cdot \frac{1}{\left(2^{12}+-1\right)}\right)+-\left(\mathrm{ADC}_R \cdot \frac{\mathrm{INT1V}_R}{\left(2^{12}+-1\right)}\right)\right\}\cdot\left(\mathrm{temp}_H+-\mathrm{temp}_R\right)}{\left\{\left(\mathrm{ADC}_H \cdot \frac{\mathrm{INT1V}_H}{\left(2^{12}+-1\right)}\right)+-\left(\mathrm{ADC}_R \cdot \frac{\mathrm{INT1V}_R}{\left(2^{12}+-1\right)}\right)\right\}}\right]$$

[Equation 1]

**Note:**

1. In the previous expression, we have added the conversion of the ADC register value to be expressed in V.
2. This is a coarse value because we assume INT1V=1V for this ADC conversion.

Using the ($temp_R$, $INT1V_R$) and ($temp_H$, $INT1V_H$) points, using a linear interpolation we have the following equation:

$$\left(\frac{\mathrm{INT1V}+-\mathrm{INT1V}_R}{\mathrm{temp}+-\mathrm{temp}_R}\right)=\left(\frac{\mathrm{INT1V}_H+-\mathrm{INT1V}_R}{\mathrm{temp}_H+-\mathrm{temp}_R}\right)$$

Then using the coarse temperature value, we can infer a closer to reality INT1V value during the ADC conversion as:

$$\mathrm{INT1V}_m = \mathrm{INT1V}_R + \left(\frac{\left(\mathrm{INT1V}_H+-\mathrm{INT1V}_R\right)\cdot\left(\mathrm{temp}_C+-\mathrm{temp}_R\right)}{\left(\mathrm{temp}_H+-\mathrm{temp}_R\right)}\right)$$

Back to [Equation 1], if we replace INT1V=1V by INT1V = $INT1V_m$, we can deduce a finer temperature value as:

$$\mathrm{temp}_f = \mathrm{temp}_R + \left[\frac{\left\{\left(\mathrm{ADC}_m \cdot \frac{\mathrm{INT1V}_m}{\left(2^{12}+-1\right)}\right)+-\left(\mathrm{ADC}_R \cdot \frac{\mathrm{INT1V}_R}{\left(2^{12}+-1\right)}\right)\right\}\cdot\left(\mathrm{temp}_H\cdot\mathrm{temp}_R\right)}{\left\{\left(\mathrm{ADC}_H \cdot \frac{\mathrm{INT1V}_H}{\left(2^{12}+-1\right)}\right)+-\left(\mathrm{ADC}_R \cdot \frac{\mathrm{INT1V}_R}{\left(2^{12}+-1\right)}\right)\right\}}\right]$$

[Equation 1bis]

## 37.11 NVM Characteristics

**Table 37-40. Maximum Operating Frequency**

| $V_{DD}$ range | NVM Wait States | Maximum Operating Frequency | Units |
|---|---|---|---|
| 1.62V to 2.7V | 0 | 14 | MHz |
| | 1 | 28 | |
| | 2 | 42 | |
| | 3 | 48 | |
| 2.7V to 3.63V | 0 | 24 | |
| | 1 | 48 | |

Note that on this flash technology, a max number of 8 consecutive write is allowed per row. Once this number is reached, a row erase is mandatory.

**Table 37-41. Flash Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{NVM25k}$ | Retention after up to 25k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{NVM2.5k}$ | Retention after up to 2.5k | Average ambient 55°C | 20 | 100 | - | Years |
| $Ret_{NVM100}$ | Retention after up to 100 | Average ambient 55°C | 25 | >100 | - | Years |
| $Cyc_{NVM}$ | Cycling Endurance(1) | -40°C < Ta < 125°C | 25k | 150k | - | Cycles |

1. An endurance cycle is a write and an erase operation.

**Table 37-42. EEPROM Emulation(1) Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{EEPROM100k}$ | Retention after up to 100k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{EEPROM10k}$ | Retention after up to 10k | Average ambient 55°C | 20 | 100 | - | Years |
| $Cyc_{EEPROM}$ | Cycling Endurance(2) | -40°C < Ta < 125°C | 100k | 600k | - | Cycles |

1. The EEPROM emulation is a software emulation described in the App note AT03265.

2. An endurance cycle is a write and an erase operation.

**Table 37-43. NVM Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{FPP}$ | Page programming time | - | - | - | 2.5 | ms |
| $t_{FRE}$ | Row erase time | - | - | - | 6 | ms |
| $t_{FCE}$ | DSU chip erase time (CHIP_ERASE) | - | - | - | 240 | ms |

**Table 37-44. NVM Characteristics (Device Variant B,C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{FPP}$ | Page programming time | - | - | - | 2.5 | ms |
| $t_{FRE}$ | Row erase time | - | - | 1.2 | 6 | ms |
| $t_{FCE}$ | DSU chip erase time (CHIP_ERASE) | - | - | - | 240 | ms |

## 37.12 Oscillators Characteristics

### 37.12.1 Crystal Oscillator (XOSC) Characteristics

#### 37.12.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

**Table 37-45. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN}$ | XIN clock frequency | | - | - | 32 | MHz |

#### 37.12.1.2 Crystal Oscillator Characteristics

The following table describes the characteristics for the oscillator when a crystal is connected between XIN and XOUT . The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal datasheet. The capacitance of the external capacitors ($C_{LEXT}$) can then be computed as follows:

$$C_{LEXT} = 2(C_L + - C_{STRAY} - C_{SHUNT})$$

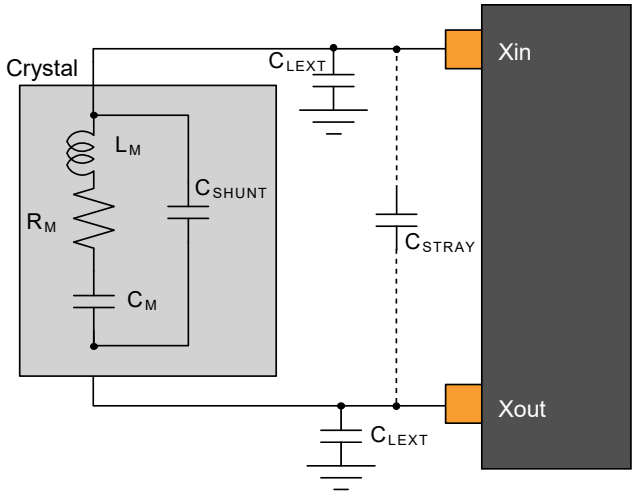
where $C_{STRAY}$ is the capacitance of the pins and PCB, $C_{SHUNT}$ is the shunt capacitance of the crystal.

**Table 37-46. Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | 0.4 | - | 32 | MHz |
| ESR | Crystal Equivalent Series Resistance<br>Safety Factor = 3<br>The AGC doesn't have any noticeable impact on these measurements. | f = 0.455 MHz, $C_L$ = 100pF<br>XOSC.GAIN = 0 | - | - | 5.6K | Ω |
| | | f = 2MHz, $C_L$ = 20pF<br>XOSC.GAIN = 0 | - | - | 416 | |
| | | f = 4MHz, $C_L$ = 20pF<br>XOSC.GAIN = 1 | - | - | 243 | |
| | | f = 8 MHz, $C_L$ = 20pF<br>XOSC.GAIN = 2 | - | - | 138 | |
| | | f = 16 MHz, $C_L$ = 20pF<br>XOSC.GAIN = 3 | - | - | 66 | |
| | | f = 32MHz, $C_L$ = 18pF<br>XOSC.GAIN = 4 | - | - | 56 | |
| $C_{XIN}$ | Parasitic capacitor load | | - | 5.9 | - | pF |
| $C_{XOUT}$ | Parasitic capacitor load | | - | 3.2 | - | pF |
| | Current Consumption | f = 2MHz, $C_L$ = 20pF, AGC off | 27 | 65 | 85 | µA |
| | | f = 2MHz, $C_L$ = 20pF, AGC on | 14 | 52 | 73 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC off | 61 | 117 | 150 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC on | 23 | 74 | 100 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC off | 131 | 226 | 296 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC on | 56 | 128 | 172 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC off | 305 | 502 | 687 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC on | 116 | 307 | 552 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC off | 1031 | 1622 | 2200 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC on | 278 | 615 | 1200 | |
| $t_{STARTUP}$ | Start-up time | f = 2MHz, $C_L$ = 20pF,<br>XOSC.GAIN = 0, ESR = 600Ω | - | 14K | 48K | cycles |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | f = 4MHz, $C_L$ = 20pF, XOSC.GAIN = 1, ESR = 100Ω | - | 6.8K | 19.5K | |
| | | f = 8 MHz, $C_L$ = 20pF, XOSC.GAIN = 2, ESR = 35Ω | - | 5.55K | 13K | |
| | | f = 16 MHz, $C_L$ = 20pF, XOSC.GAIN = 3, ESR = 25Ω | - | 6.75K | 14.5K | |
| | | f = 32MHz, $C_L$ = 18pF, XOSC.GAIN = 4, ESR = 40Ω | - | 5.3K | 9.6K | |

**Figure 37-6. Oscillator Connection**




### 37.12.2 External 32 kHz Crystal Oscillator (XOSC32K) Characteristics

#### 37.12.2.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN32 pin.

**Table 37-47. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN32}$ | XIN32 clock frequency | | - | 32.768 | - | kHz |
| | XIN32 clock duty cycle | | - | 50 | - | % |

##### 37.12.2.1.1 Crystal Oscillator Characteristics

Figure 37-6 and the equation in also applies to the 32 kHz oscillator connection. The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal datasheet.

**Table 37-48. 32kHz Crystal Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | - | 32768 | - | Hz |
| $t_{STARTUP}$ | Startup time | $ESR_{XTAL}$ = 39.9 kΩ, $C_L$ = 12.5 pF | - | 28K | 30K | cycles |
| $C_L$ | Crystal load capacitance | | - | - | 12.5 | pF |
| $C_{SHUNT}$ | Crystal shunt capacitance | | - | 0.1 | - | |
| $C_{XIN32}$ | Parasitic capacitor load | | - | 3.1 | - | |
| $C_{XOUT32}$ | Parasitic capacitor load | | - | 3.3 | - | |
| $I_{XOSC32K}$ | Current consumption | | - | 1.22 | 2.19 | µA |
| ESR | Crystal equivalent series resistance f=32.768kHz , Safety Factor = 3 | $C_L$=12.5pF | - | - | 141 | kΩ |

**Table 37-49. 32kHz Crystal Oscillator Characteristics (Device Variant B and C)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | - | 32768 | - | Hz |
| $t_{STARTUP}$ | Startup time | $ESR_{XTAL}$ = 39.9 kΩ, $C_L$ = 12.5 pF | - | 28K | 30K | cycles |
| $C_L$ | Crystal load capacitance | | - | - | 12.5 | pF |
| $C_{SHUNT}$ | Crystal shunt capacitance | | - | 0.1 | - | |
| $C_{XIN32}$ | Parasitic capacitor load | | - | 3.2 | - | |
| $C_{XOUT32}$ | Parasitic capacitor load | | - | 3.7 | - | |
| $I_{XOSC32K}$ | Current consumption | | - | 1.22 | 2.19 | µA |
| ESR | Crystal equivalent series resistance f=32.768kHz , Safety Factor = 3 | $C_L$=12.5pF | - | - | 100 | kΩ |

### 37.12.3 Digital Frequency Locked Loop (DFLL48M) Characteristics

**Table 37-50. DFLL48M Characteristics - Open Loop Mode(1)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | 47 | 48 | 49 | MHz |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | IDFLLVAL.COARSE = DFLL48M COARSE CAL | - | 403 | 453 | µA |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | DFLLVAL.FINE = 512 | | | | |
| $t_{STARTUP}$ | Start-up time | DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>$f_{OUT}$ within 90 % of final value | - | 8 | 9 | µs |

Note: 1. DFLL48M in Open loop after calibration at room temperature.

**Table 37-51. DFLL48M Characteristics - Closed Loop Mode(1) (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Average Output frequency | $f_{REF}$ = XTAL, 32 .768kHz, 100ppm<br>DFLLMUL = 1464 | 47.963 | 47.972 | 47.981 | MHz |
| $f_{REF}$ | Reference frequency | | 0.732 | 32.768 | 33 | kHz |
| Jitter | Cycle to Cycle jitter | $f_{REF}$ = XTAL, 32 .768kHz, 100ppm<br>DFLLMUL = 1464 | - | - | 0.42 | ns |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | $f_{REF}$ = XTAL, 32 .768kHz, 100ppm<br>DFLLMUL = 1464 | - | 425 | 482 | µA |
| $t_{LOCK}$ | Lock time | $f_{REF}$ = XTAL, 32 .768kHz, 100ppm<br>DFLLMUL = 1464<br>DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>DFLLCTRL.BPLCKC = 1<br>DFLLCTRL.QLDIS = 0<br>DFLLCTRL.CCDIS = 1<br>DFLLMUL.FSTEP = 10 | 100 | 200 | 500 | µs |

**Table 37-52. DFLL48M Characteristics - Closed Loop Mode(1) (Device Variant B and C)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Average Output frequency | $f_{REF}$ = 32 .768kHz | 47.963 | 47.972 | 47.981 | MHz |
| $f_{REF}$ | Reference frequency | | 0.732 | 32.768 | 33 | kHz |
| Jitter | Cycle to Cycle jitter | $f_{REF}$ = 32 .768kHz | - | - | 0.42 | ns |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | $f_{REF}$ =32 .768kHz | - | 403 | 453 | µA |
| $t_{LOCK}$ | Lock time | $f_{REF}$ = 32 .768kHz<br>DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>DFLLCTRL.BPLCKC = 1<br>DFLLCTRL.QLDIS = 0<br>DFLLCTRL.CCDIS = 1<br>DFLLMUL.FSTEP = 10 | - | 200 | 500 | µs |

1. To insure that the device stays within the maximum allowed clock frequency, any reference clock for DFLL in close loop must be within a 2% error accuracy.

### 37.12.4 32.768kHz Internal oscillator (OSC32K) Characteristics

**Table 37-53. 32kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 28.508 | 32.768 | 34.734 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 32.276 | 32.768 | 33.260 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.457 | 32.768 | 34.079 | |
| $I_{OSC32K}$ | Current consumption | | - | 0.67 | 1.31 | µA |
| $t_{STARTUP}$ | Start-up time | | - | 1 | 2 | cycle |
| Duty | Duty Cycle | | - | 50 | - | % |

### 37.12.5 Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics

**Table 37-54. Ultra Low Power Internal 32kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 25.559 | 32.768 | 38.011 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 31.293 | 32.768 | 34.570 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.293 | 32.768 | 34.570 | |
| $i_{OSCULP32K}$(1)(2) | | | - | - | 125 | nA |
| $t_{STARTUP}$ | Start-up time | | - | 10 | - | cycles |
| Duty | Duty Cycle | | - | 50 | - | % |

Notes: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

2. This oscillator is always on.

### 37.12.6 8MHz RC Oscillator (OSC8M) Characteristics

**Table 37-55. Internal 8MHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 8MHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 7.8 | 8 | 8.16 | MHz |
| | | Calibrated against a 8MHz reference at 25°C, at $V_{DD}$=3.3V | 7.94 | 8 | 8.06 | |
| | | Calibrated against a 8MHz reference at 25°C, over [1.62, 3.63]V | 7.92 | 8 | 8.08 | |
| TempCo | Frequency vs. Temperature drift | | -1.2 | | 1 | % |
| SupplyCo | Frequency vs. Supply drift | | -2 | | 2 | % |
| $I_{OSC8M}$ | Current consumption | IDLE2 on OSC32K versus IDLE2 on calibrated OSC8M enabled at 8MHz (FRANGE=1, PRESC=0) | | 64 | | µA |
| $t_{STARTUP}$ | Startup time | | - | 2.4 | 3.3 | µs |
| Duty | Duty cycle | | - | 50 | - | % |

### 37.12.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics

**Table 37-56. FDPLL96M Characteristics(1) (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | 700 | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | 1200 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 1.5 | 2.0 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 3.0 | 10.0 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 1.3 | 2.0 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 3.0 | 7.0 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.3 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Table 37-57. FDPLL96M Characteristics(1) (Device Variant B and L with Silicon Revision E)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | 700 | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | 1200 | |
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 1.5 | 2.1 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 4.0 | 10.0 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 1.6 | 2.2 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 4.6 | 10.2 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.2 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Table 37-58. FDPLL96M Characteristics(1) (Device Variant B, C and L with Silicon Revision F)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | - | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | - | |
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 2.2 | 3.0 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 3.7 | 9.0 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 2.2 | 3.0 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 4.4 | 9.7 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.0 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 22 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Note:** All values have been characterized with FILTSEL[1/0] as the default value.

## 37.13 PTC Typical Characteristics

### 37.13.1 Device Variant A

**Figure 37-7. Power Consumption [µA]**
**1 sensor, noise countermeasures disabled, f=48MHz, Vcc=3.3V**

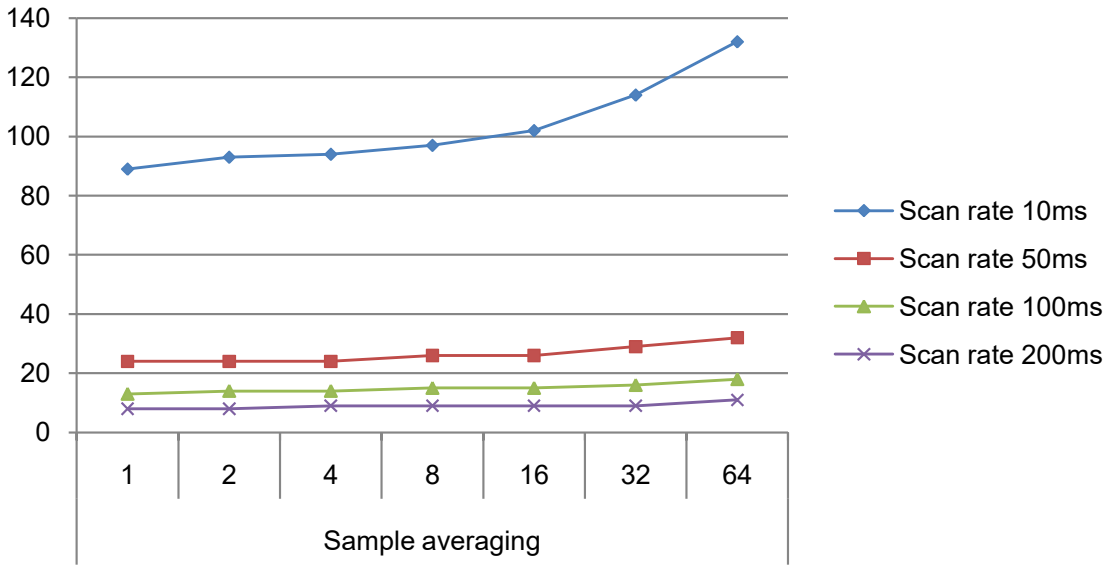


**Figure 37-8. Power Consumption [µA]**
**1 sensor, noise countermeasures Enabled, f=48MHz, Vcc=3.3V**

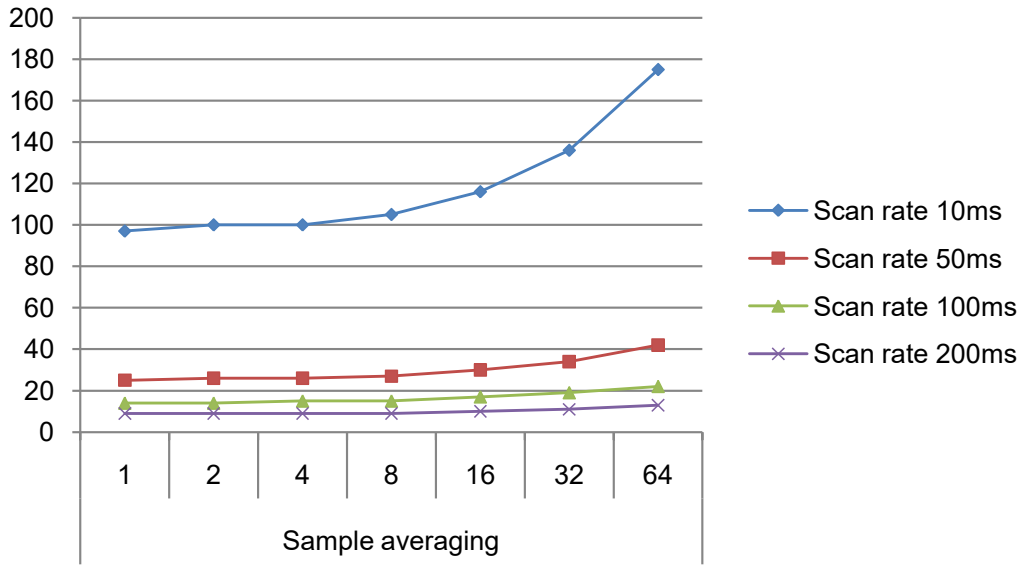



**Figure 37-9. Power Consumption [µA]**
**10 sensors, noise countermeasures disabled, f=48MHz, Vcc=3.3V**

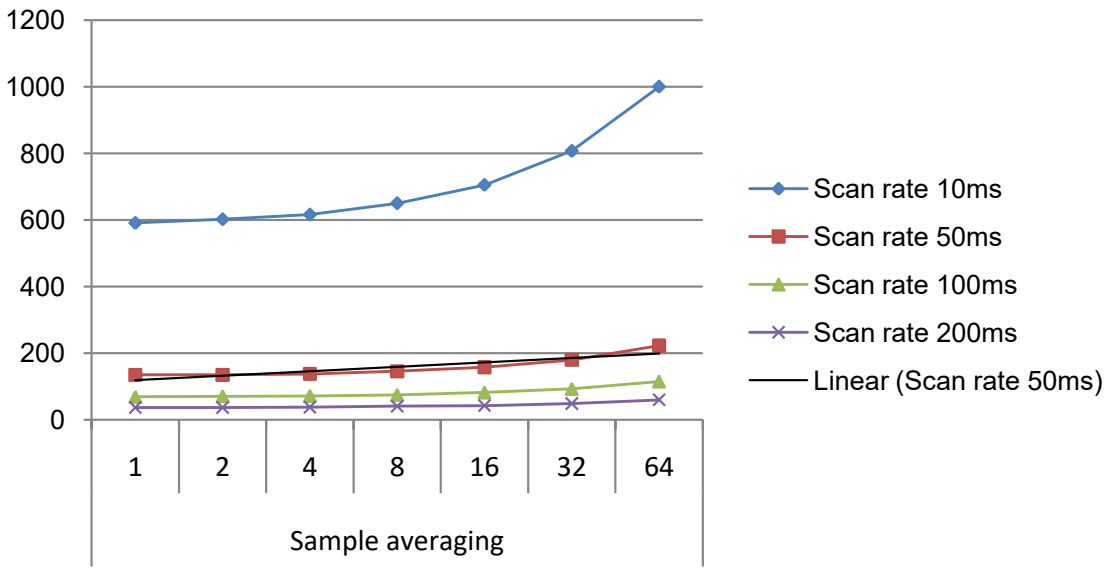


**Figure 37-10. Power Consumption [µA]**
**10 sensors, noise countermeasures Enabled, f=48MHz, Vcc=3.3V**

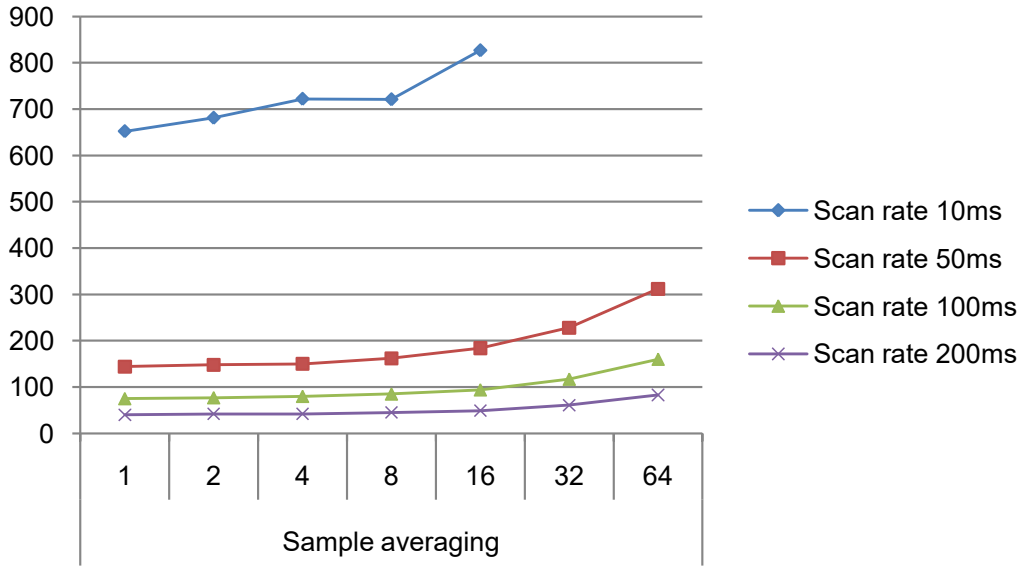



**Figure 37-11. Power Consumption [µA]**
**100 sensors, noise countermeasures disabled, f=48MHz, Vcc=3.3V**

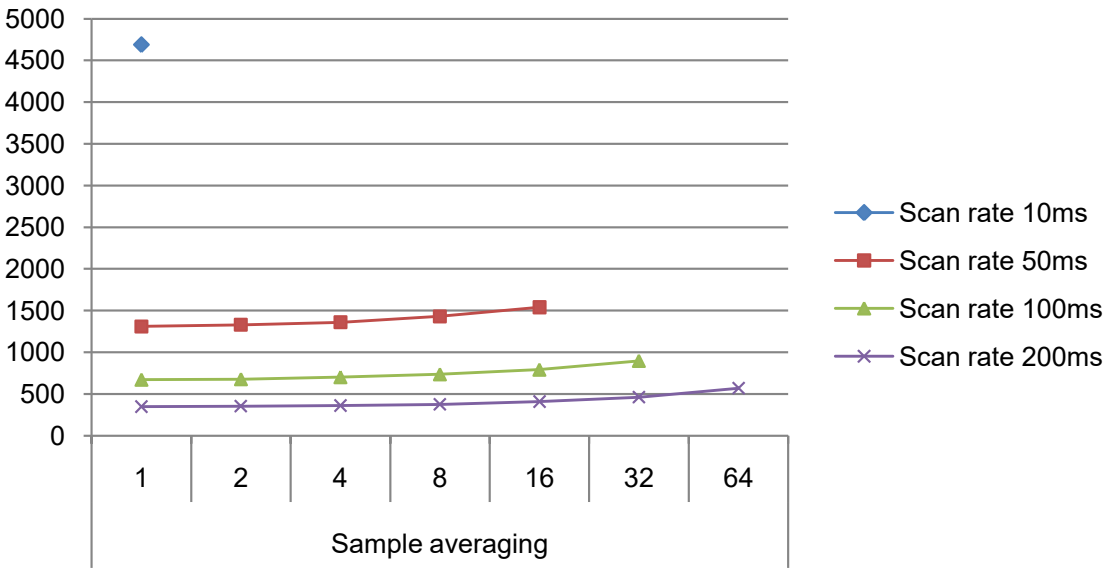


**Figure 37-12. Power Consumption [µA]**
**100 sensors, noise countermeasures Enabled, f=48MHz, Vcc=3.3V**

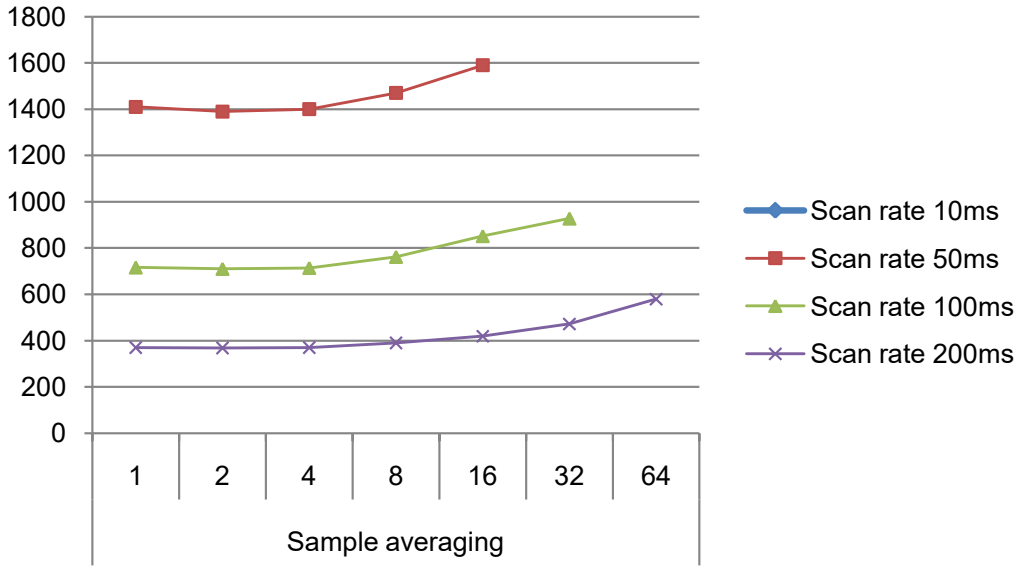
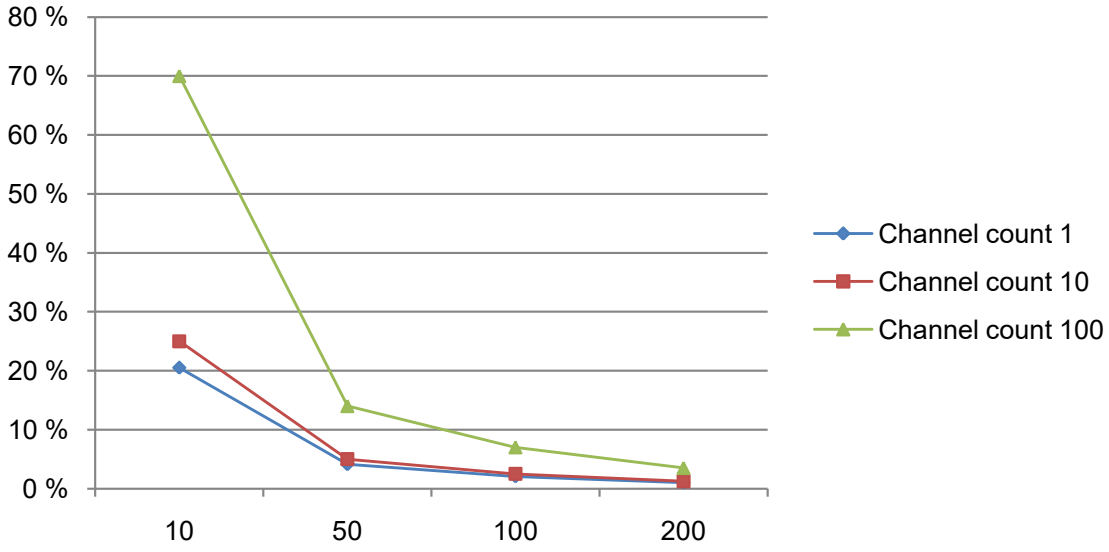



**Figure 37-13. CPU Utilization**




### 37.13.2 Device Variant B and C

$V_{CC}$ = 3.3C and $f_{CPU}$ = 48MHz for the following PTC measurements.

**Figure 37-14. 1 Sensor / PTC_GCLK = 4MHz / FREQ_MODE_NONE**

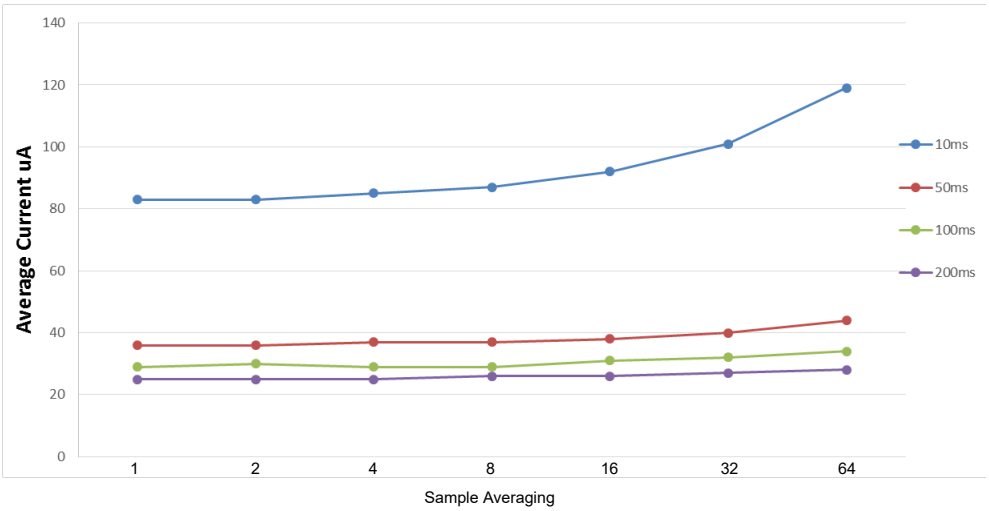



**Figure 37-15. 1 Sensor / PTC_GCLK = 2MHz / FREQ_MODE_HOP**

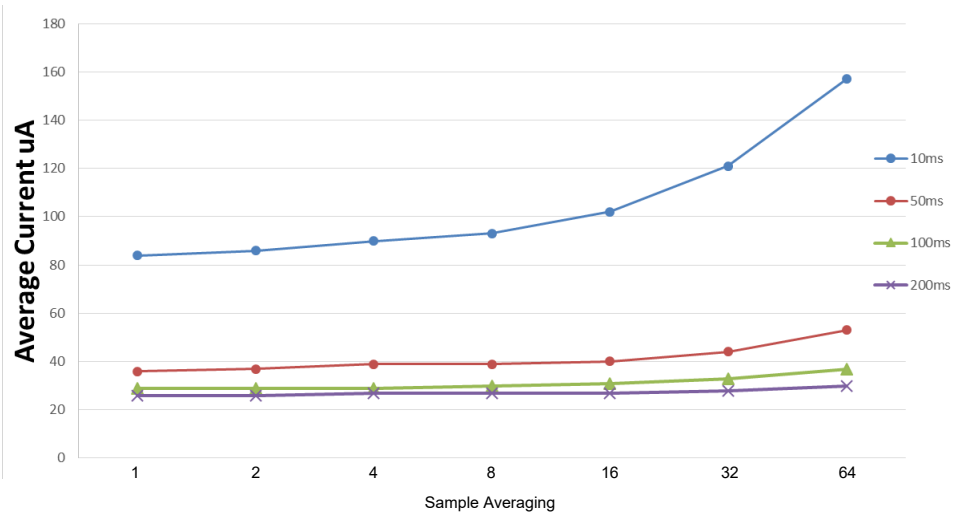


**Figure 37-16. 10 Sensor / PTC_GCLK = 4MHz / FREQ_MODE_NONE**

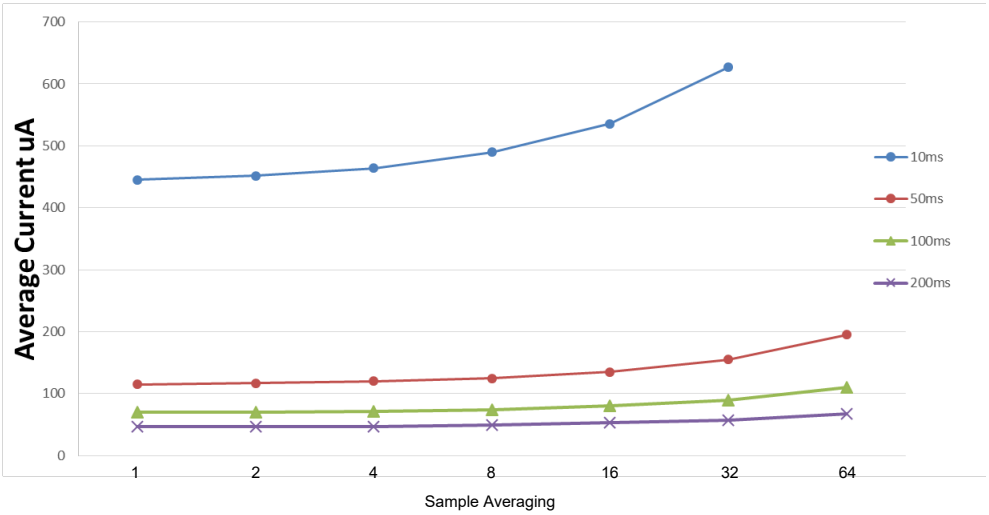



**Figure 37-17. 10 Sensor / PTC_GCLK = 2MHz / FREQ_MODE_HOP**

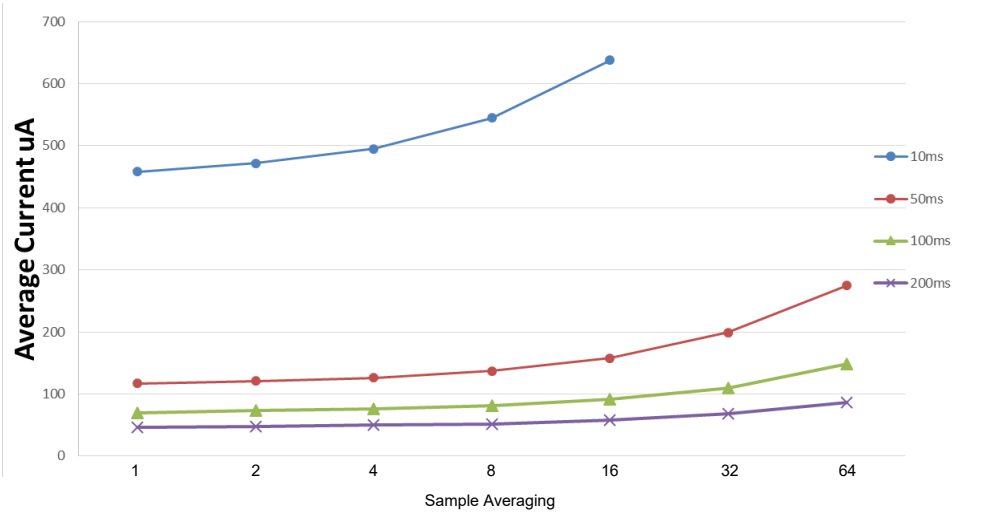


**Figure 37-18. 100 Sensor / PTC_GCLK = 4MHz / FREQ_MODE_NONE**

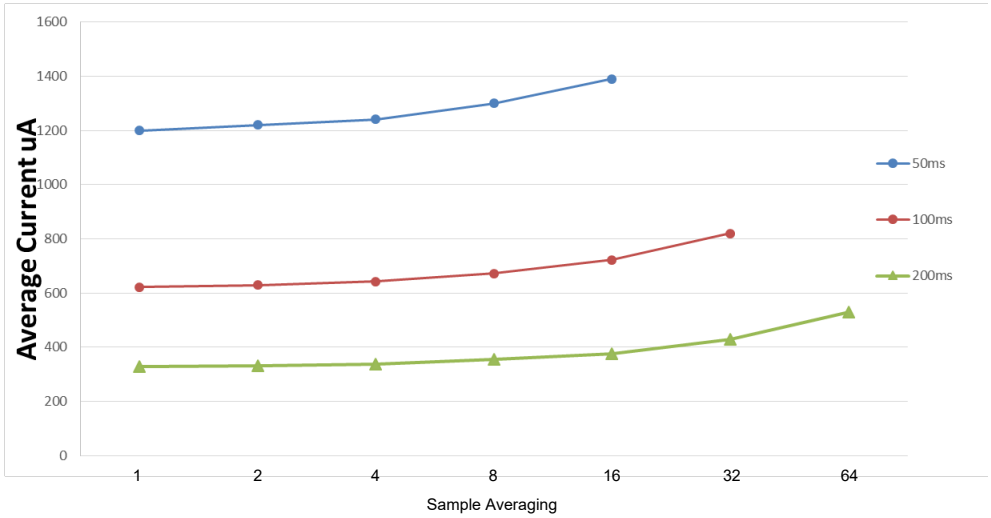
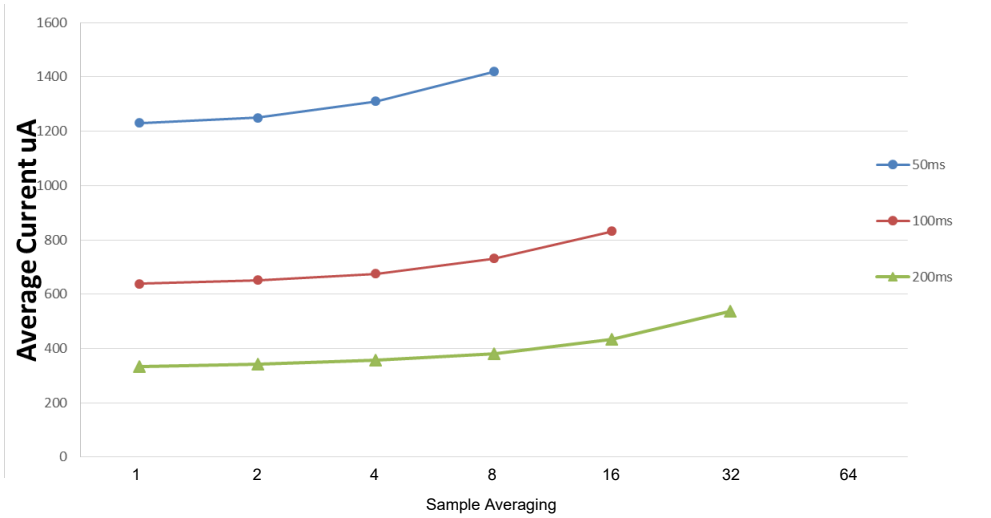



**Figure 37-19. 100 Sensor / PTC_GCLK = 2MHz / FREQ_MODE_HOP**




## 37.14 USB Characteristics

The USB on-chip buffers comply with the Universal Serial Bus (USB) v2.0 standard. All AC parameters related to these buffers can be found within the USB 2.0 electrical specifications.

The USB interface is USB-IF certified:

- TID 40001583 - Peripheral Silicon > Low/Full Speed > Silicon Building Blocks

- TID 120000272 - Embedded Hosts > Full Speed

Electrical configuration required to be USB compliance:

- The CPU frequency must be higher 8MHz when USB is active (No constraint for USB suspend mode)

- The operating voltages must be 3.3V (Min. 3.0V, Max. 3.6V).

- The GCLK_USB frequency accuracy source must be less than:

- In USB device mode, 48MHz +/-0.25%

- In USB host mode, 48MHz +/-0.05%

**Table 37-59. GCLK_USB Clock Setup Recommendations**

| Clock setup | | USB Device | USB Host |
|---|---|---|---|
| DFLL48M | Open loop | No | No |
| | Closed loop, any internal OSC source | No | No |
| | Closed loop, any external XOSC source | Yes | No |
| | Closed loop, USB SOF source (USB recovery mode)(1) | Yes(2) | N/A |
| FDPLL96M | Any internal OSC source (32K, 8M, ... ) | No | No |
| | Any external XOSC source (< 1MHz) | Yes | No |
| | Any external XOSC source (> 1MHz) | Yes(3) | Yes |

Notes: 1. When using DFLL48M in USB recovery mode, the Fine Step value must be Ah to guarantee a USB clock at +/-0.25% before 11ms after a resume.

2. Very high signal quality and crystal less. It is the best setup for USB Device mode.

3. FDPLL lock time is short when the clock frequency source is high (> 1MHz). Thus, FDPLL and external OSC can be stopped during USB suspend mode to reduce consumption and guarantee a USB wake-up time (See TDRSMDN in USB specification).

## 37.15 Timing Characteristics

### 37.15.1 External Reset

**Table 37-60. External Reset Characteristics**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{EXT}$ | Minimum reset pulse width | | 10 | - | - | ns |

### 37.15.2 SERCOM in SPI Mode Timing

**Figure 37-20. SPI Timing Requirements in Master Mode**

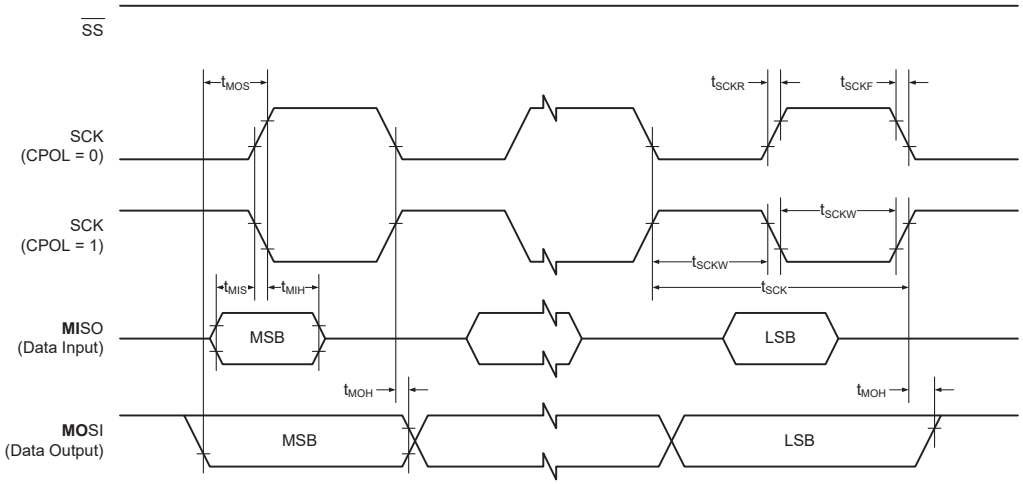



**Figure 37-21. SPI Timing Requirements in Slave Mode**

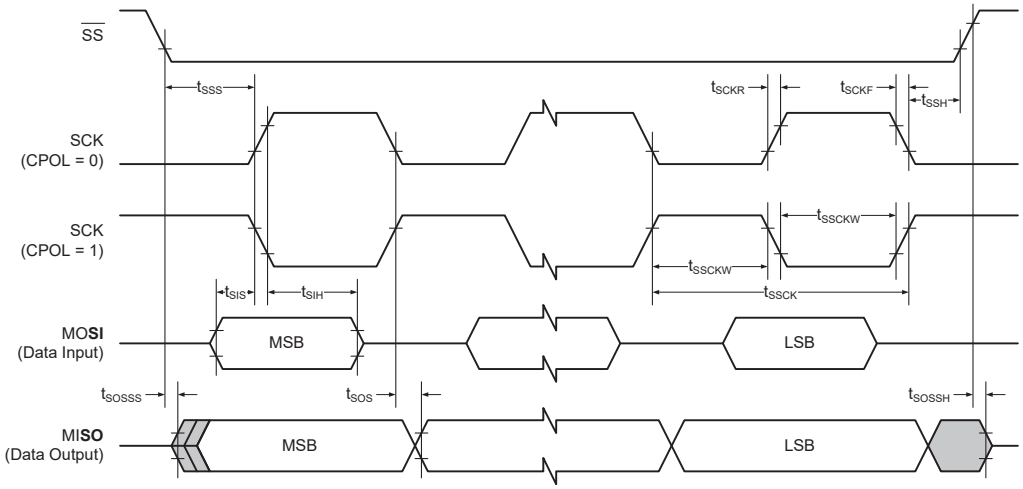



**Table 37-61. SPI Timing Characteristics and Requirements(1)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{SCK}$ | SCK period | Master | | 84 | | ns |
| $t_{SCKW}$ | SCK high/low width | Master | - | 0.5*$t_{SCK}$ | - | |
| $t_{SCKR}$ | SCK rise time(2) | Master | - | - | - | |
| $t_{SCKF}$ | SCK fall time(2) | Master | - | - | - | |
| $t_{MIS}$ | MISO setup to SCK | Master | - | 21 | - | |
| $t_{MIH}$ | MISO hold after SCK | Master | - | 13 | - | |
| $t_{MOS}$ | MOSI setup SCK | Master | - | $t_{SCK}$/2 - 3 | - | |
| $t_{MOH}$ | MOSI hold after SCK | Master | - | 3 | - | |
| $t_{SSCK}$ | Slave SCK Period | Slave | 1*$t_{CLK_APB}$ | - | - | |

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| $t_{SSCKW}$ | SCK high/low width | Slave | | 0.5*$t_{SSCK}$ | - | - | |
| $t_{SSCKR}$ | SCK rise time(2) | Slave | | - | - | - | |
| $t_{SSCKF}$ | SCK fall time(2) | Slave | | - | - | - | |
| $t_{SIS}$ | MOSI setup to SCK | Slave | | $t_{SSCK}$/2 - 9 | - | - | |
| $t_{SIH}$ | MOSI hold after SCK | Slave | | $t_{SSCK}$/2 - 3 | - | - | |
| $t_{SSS}$ | SS setup to SCK | Slave | PRELOADEN =1 | 2*$t_{CLK_APB}$ + $t_{SOS}$ | - | - | |
| | | | PRELOADEN =0 | $t_{SOS}$+7 | - | - | |
| $t_{SSH}$ | SS hold after SCK | Slave | | $t_{SIH}$ - 4 | - | - | |
| $t_{SOS}$ | MISO setup SCK | Slave | | - | $t_{SSCK}$/2 - 18 | - | |
| $t_{SOH}$ | MISO hold after SCK | Slave | | - | 18 | - | |
| $t_{SOSS}$ | MISO setup after SS low | Slave | | - | 18 | - | |
| $t_{SOSH}$ | MISO hold after SS high | Slave | | - | 10 | - | |

Notes: 1. These values are based on simulation. These values are not covered by test limits in production.

2. See 37.8 I/O Pin Characteristics

### 37.15.3 SERCOM in I²C Mode Timing

This section describes the requirements for devices connected to the I²C Interface Bus.

**Figure 37-22. I²C Interface Bus Timing**

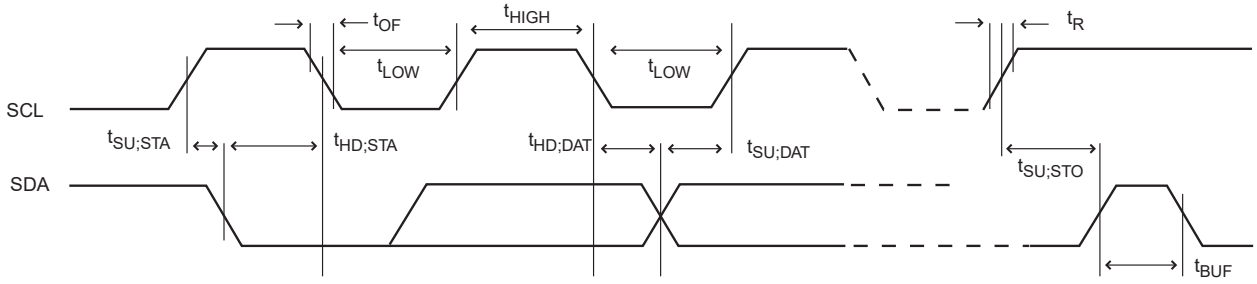



**Table 37-62. I²C Interface Timing (Device Variant A)**

| Symbol | Parameter | | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| $t_R$ | Rise time for both SDA and SCL | Standard / Fast Mode | $IC_b$(2) = 400pF | - | 215 | 300 | ns |
| | | Fast | $IC_b$(2) = 550pF | | 60 | 100 | |

| Symbol | Parameter | | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | | Mode + | | | | | |
| | | High Speed Mode | $IC_b$[2] = 100pF | | 20 | 40 | |
| $t_{OF}$ | Output fall time from $V_{IHmin}$ to $V_{ILmax}$ | Standard / Fast Mode | 10pF < $C_b$[2] < 400pF | | 20.0 | 50.0 | |
| | | Fast Mode + | 10pF < $C_b$[2] < 550pF | | 15.0 | 50.0 | |
| | | High Speed Mode | 10pF < $C_b$[2] < 100pF | | 10.0 | 40.0 | |
| $t_{HD;STA}$ | Hold time (repeated) START condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$-9 | - | - | |
| $t_{LOW}$ | Low period of SCL Clock | | $f_{SCL}$ > 100kHz | 113 | - | - | |
| $t_{BUF}$ | Bus free time between a STOP and a START condition | | $f_{SCL}$ > 100kHz | $t_{LOW}$ | - | - | |
| $t_{SU;STA}$ | Setup time for a repeated START condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$+7 | - | - | |
| $t_{HD;DAT}$ | Data hold time | | $f_{SCL}$ > 100kHz, Master | 9 | - | 12 | |
| $t_{SU;DAT}$ | Data setup time | | $f_{SCL}$ > 100kHz, Master | 104 | - | - | |
| $t_{SU;STO}$ | Setup time for STOP condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$+9 | - | - | |
| $t_{SU;DAT;rx}$ | Data setup time (receive mode) | | $f_{SCL}$ > 100kHz, Slave | 51 | - | 56 | |
| $t_{HD;DAT;tx}$ | Data hold time (send mode) | | $f_{SCL}$ > 100kHz, Slave | 71 | 90 | 138 | |

**Table 37-63. $I^2C$ Interface Timing (Device Variant B and C)**

| Symbol | Parameter | | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| $t_R$ | Rise time for both SDA and SCL | Standard / Fast Mode | $C_b$[2] = 400pF | - | 230 | 350 | ns |
| | | Fast Mode + | $C_b$[2] = 550pF | | 60 | 100 | |
| | | High Speed Mode | $C_b$[2] = 100pF | | 30 | 60 | |
| $t_{OF}$ | Output fall time from $V_{IHmin}$ to $V_{ILmax}$ | Standard / Fast Mode | 10pF < $C_b$[2] < 400pF | | 25 | 50 | |
| | | Fast | 10pF < $C_b$[2] < 550pF | | 20 | 30 | |

| Symbol | Parameter | | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | | Mode + | | | | | |
| | | High Speed Mode | 10pF < $C_b$[(2)] < 100pF | | 10 | 20 | |
| $t_{HD;STA}$ | Hold time (repeated) START condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$-9 | - | - | |
| $t_{LOW}$ | Low period of SCL Clock | | $f_{SCL}$ > 100kHz | 113 | - | - | |
| $t_{BUF}$ | Bus free time between a STOP and a START condition | | $f_{SCL}$ > 100kHz | $t_{LOW}$ | - | - | |
| $t_{SU;STA}$ | Setup time for a repeated START condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$+7 | - | - | |
| $t_{HD;DAT}$ | Data hold time | | $f_{SCL}$ > 100kHz, Master | 9 | - | 12 | |
| $t_{SU;DAT}$ | Data setup time | | $f_{SCL}$ > 100kHz, Master | 104 | - | - | |
| $t_{SU;STO}$ | Setup time for STOP condition | | $f_{SCL}$ > 100kHz, Master | $t_{LOW}$+9 | - | - | |
| $t_{SU;DAT;rx}$ | Data setup time (receive mode) | | $f_{SCL}$ > 100kHz, Slave | 51 | - | 56 | |
| $t_{HD;DAT;tx}$ | Data hold time (send mode) | | $f_{SCL}$ > 100kHz, Slave | 71 | 90 | 138 | |

1. These values are based on simulation. These values are not covered by test limits in production.
2. $C_b$ = Capacitive load on each bus line. Otherwise noted, value of $C_b$ set to 20pF.

### 37.15.4 SWD Timing

**Figure 37-23. SWD Interface Signals**

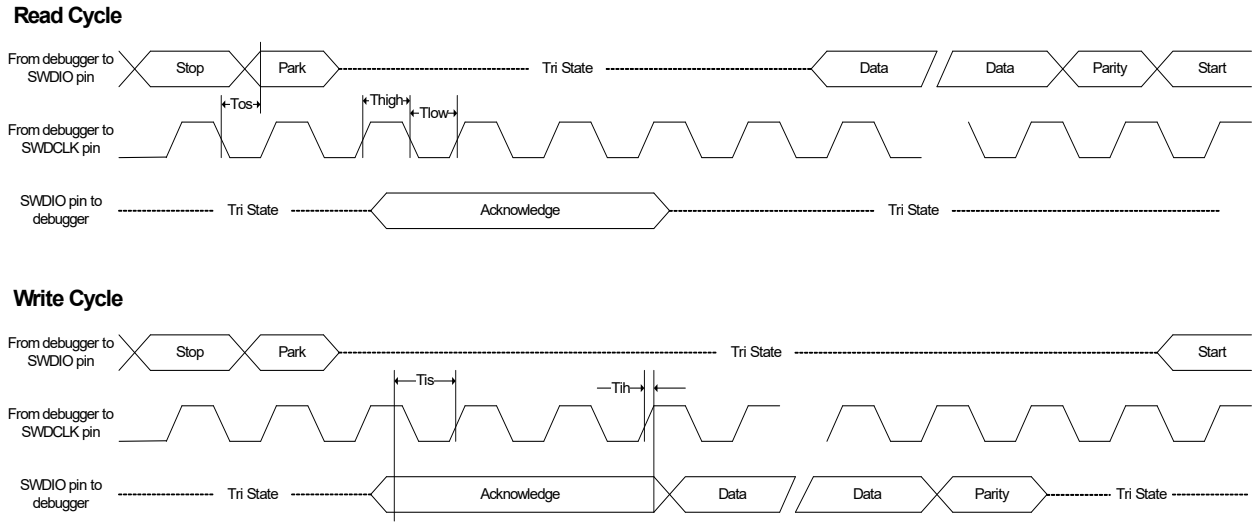


**Table 37-64. SWD Timings**[1]

| Symbol | Parameter | Conditions | Min. | Max. | Units |
|---|---|---|---|---|---|
| $T_{high}$ | SWDCLK High period | $V_{VDDIO}$ from 3.0 V to 3.6 V, maximum external capacitor = 40 pF | 10 | 500000 | ns |
| $T_{low}$ | SWDCLK Low period | | 10 | 500000 | |
| $T_{os}$ | SWDIO output skew to falling edge SWDCLK | | -5 | 5 | |
| $T_{is}$ | Input Setup time required between SWDIO | | 4 | - | |
| $T_{ih}$ | Input Hold time required between SWDIO and rising edge SWDCLK | | 1 | - | |

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

### 37.15.5 I2S Timing

**Figure 37-24. I2S Timing Master Mode**

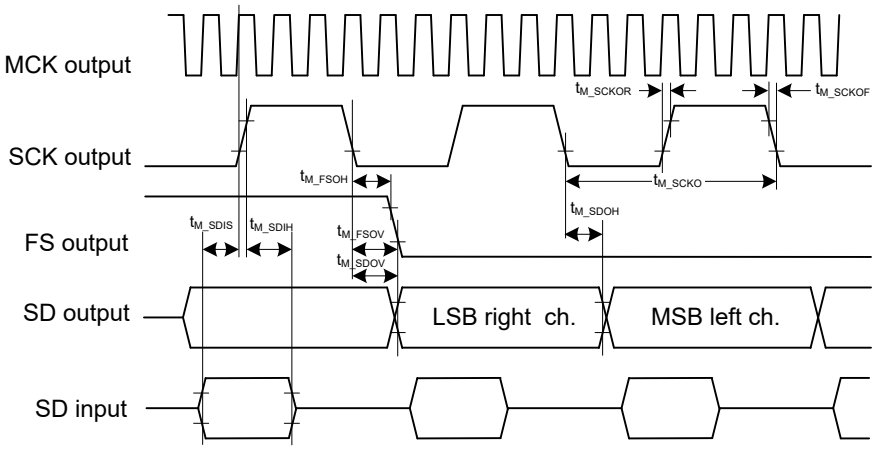



**Figure 37-25. I2S Timing Slave Mode**

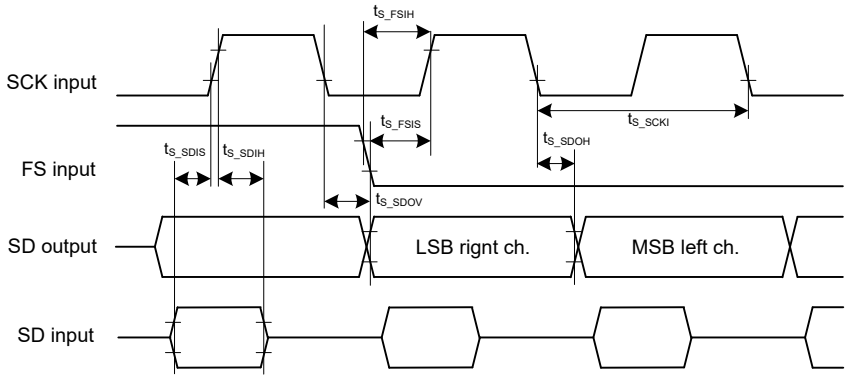


**Figure 37-26. I2S Timing PDM2 Mode**

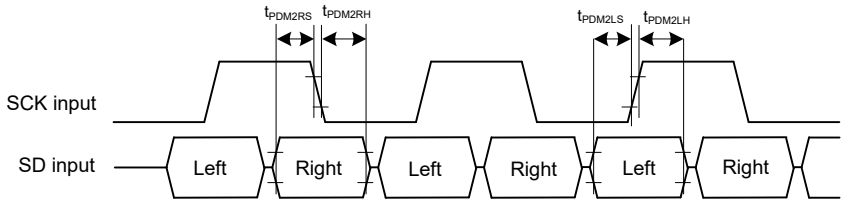



**Table 37-65. I2S Timing Characteristics and Requirements (Device Variant A)**

| Name | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max | Min. | Typ. | Max. | |
| $t_{M_MCKOR}$ | I2S MCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.2 | | | 4.7 | ns |
| $t_{M_MCKOF}$ | I2S MCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 11.5 | | | 5.3 | ns |
| $d_{M_MCKO}$ | I2S MCK duty cycle | Master mode | 45.4 | | 50 | 45.4 | | 50 | % |
| $d_{M_MCKI}$ | I2S MCK duty cycle | Master mode, pin is input (1b) | | 50 | | | 50 | | % |
| $t_{M_SCKOR}$ | I2S SCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9 | | | 4.6 | ns |
| $t_{M_SCKOF}$ | I2S SCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.7 | | | 4.5 | ns |
| $d_{M_SCKO}$ | I2S SCK duty cycle | Master mode | 45.6 | | 50 | 45.6 | | 50 | % |
| $f_{M_SCKO}$,1/ $t_{M_SCKO}$ | I2S SCK frequency | Master mode,Supposing external device response delay is 30ns | | | 8 | | | 9.5 | MHz |
| $f_{S_SCKI}$,1/ $t_{S_SCKI}$ | I2S SCK frequency | Slave mode,Supposing external device response delay is 30ns | | | 14.4 | | | 14.8 | MHz |
| $d_{S_SCKO}$ | I2S SCK duty cycle | Slave mode | | 50 | | | 50 | | % |
| $t_{M_FSOV}$ | FS valid time | Master mode | | | 4.1 | | | 4 | ns |
| $t_{M_FSOH}$ | FS hold time | Master mode | -0.9 | | | -0.9 | | | ns |
| $t_{S_FSIS}$ | FS setup time | Slave mode | 2.3 | | | 1.5 | | | ns |
| $t_{S_FSIH}$ | FS hold time | Slave mode | 0 | | | 0 | | | ns |
| $t_{M_SDIS}$ | Data input setup time | Master mode | 34.7 | | | 24.5 | | | ns |

| Name | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max | Min. | Typ. | Max. | |
| $t_{M_SDIH}$ | Data input hold time | Master mode | -8.2 | | | -8.2 | | | ns |
| $t_{S_SDIS}$ | Data input setup time | Slave mode | 4.6 | | | 3.9 | | | ns |
| $t_{S_SDIH}$ | Data input hold time | Slave mode | 1.2 | | | 1.2 | | | ns |
| $t_{M_SDOV}$ | Data output valid time | Master transmitter | | | 5.6 | | | 4.8 | ns |
| $t_{M_SDOH}$ | Data output hold time | Master transmitter | -0.5 | | | -0.5 | | | ns |
| $t_{S_SDOV}$ | Data output valid time | Slave transmitter | | | 36.2 | | | 25.9 | ns |
| $t_{S_SDOH}$ | Data output hold time | Slave transmitter | 36 | | | 25.7 | | | ns |
| $t_{PDM2LS}$ | Data input setup time | Master mode PDM2 Left | 34.7 | | | 24.5 | | | ns |
| $t_{PDM2LH}$ | Data input hold time | Master mode PDM2 Left | -8.2 | | | -8.2 | | | ns |
| $t_{PDM2RS}$ | Data input setup time | Master mode PDM2 Right | 30.5 | | | 20.9 | | | ns |
| $t_{PDM2RH}$ | Data input hold time | Master mode PDM2 Right | -6.7 | | | -6.7 | | | ns |

**Table 37-66. I2S Timing Characteristics and Requirements (Device Variant B and C)**

| Name | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | Min. | Typ. | Max. | |
| $t_{M_MCKOR}$ | I2S MCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.2 | | | 4.7 | ns |
| $t_{M_MCKOF}$ | I2S MCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 11.6 | | | 5.4 | ns |
| $d_{M_MCKO}$ | I2S MCK duty cycle | Master mode | 47.1 | | 50 | 47.3 | | 50 | % |
| $d_{M_MCKI}$ | I2S MCK duty cycle | Master mode, pin is input (1b) | | 50 | | | 50 | | % |
| $t_{M_SCKOR}$ | I2S SCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9 | | | 4.6 | ns |

| Name | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | Min. | Typ. | Max. | |
| $t_{M_SCKOF}$ | I2S SCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.7 | | | 4.6 | ns |
| $d_{M_SCKO}$ | I2S SCK duty cycle | Master mode | 47 | | 50 | 47.2 | | 50 | % |
| $f_{M_SCKO}$, 1/ $t_{M_SCKO}$ | I2S SCK frequency | Master mode, Supposing external device response delay is 30ns | | | 7.8 | | | 9.2 | MHz |
| $f_{S_SCKI}$, 1/ $t_{S_SCKI}$ | I2S SCK frequency | Slave mode, Supposing external device response delay is 30ns | | | 12.8 | | | 13 | MHz |
| $d_{S_SCKO}$ | I2S SCK duty cycle | Slave mode | | 50 | | | 50 | | % |
| $t_{M_FSOV}$ | FS valid time | Master mode | | | 2.4 | | | 1.9 | ns |
| $t_{M_FSOH}$ | FS hold time | Master mode | -0.1 | | | -0.1 | | | ns |
| $t_{S_FSIS}$ | FS setup time | Slave mode | 6 | | | 5.3 | | | ns |
| $t_{S_FSIH}$ | FS hold time | Slave mode | 0 | | | 0 | | | ns |
| $t_{M_SDIS}$ | Data input setup time | Master mode | 36 | | | 25.9 | | | ns |
| $t_{M_SDIH}$ | Data input hold time | Master mode | -8.2 | | | -8.2 | | | ns |
| $t_{S_SDIS}$ | Data input setup time | Slave mode | 9.1 | | | 8.3 | | | ns |
| $t_{S_SDIH}$ | Data input hold time | Slave mode | 3.8 | | | 3.7 | | | ns |
| $t_{M_SDOV}$ | Data output valid time | Master transmitter | | | 2.5 | | | 1.9 | ns |
| $t_{M_SDOH}$ | Data output hold time | Master transmitter | -0.1 | | | -0.1 | | | ns |
| $t_{S_SDOV}$ | Data output valid time | Slave transmitter | | | 29.8 | | | 19.7 | ns |
| $t_{S_SDOH}$ | Data output hold time | Slave transmitter | 29.1 | | | 18.9 | | | ns |
| $t_{PDM2LS}$ | Data input setup time | Master mode PDM2 Left | 35.5 | | | 25.3 | | | ns |
| $t_{PDM2LH}$ | Data input hold time | Master mode PDM2 Left | -8.2 | | | -8.2 | | | ns |
| $t_{PDM2RS}$ | Data input setup time | Master mode PDM2 Right | 30.6 | | | 21.1 | | | ns |
| $t_{PDM2RH}$ | Data input hold time | Master mode PDM2 Right | -7 | | | -7 | | | ns |

1. All timing characteristics given for 15pF capacitive load.
2. These values are based on simulations and not covered by test limits in production.

3. See 37.8 I/O Pin Characteristics

# 38. Electrical Characteristics at 105°C

## 38.1 Disclaimer

All typical values are measured at T = 25°C unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

## 38.2 Absolute Maximum Ratings

Stresses beyond those listed in table below may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**Table 38-1. Absolute Maximum Ratings**

| Symbol | Parameter | Condition | Min. | Max. | Units |
|---|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | | 0 | 3.8 | V |
| $I_{DD}$ | Current into a $V_{DD}$ pin | | - | 92(1) | mA |
| $I_{GND}$ | Current out of a GND pin | | - | 130(1) | mA |
| $V_{PIN}$ | Pin voltage with respect to GND and $V_{DD}$ | | GND-0.3V | $V_{DD}$+0.3V | V |
| $T_{storgae}$ | Storage temperature | | -60 | 150 | °C |

1. Maximum source current is 46mA and maximum sink current is 65mA per cluster. A cluster is a group of GPIOs. Also note that each VDD/GND pair is connected to two clusters so current consumption through the pair will be a sum of the clusters source/sink currents.

**Related Links**

7.2.4 GPIO Clusters

## 38.3 General Operating Ratings

The device must operate within the ratings listed in the table below in order for all other electrical characteristics and typical characteristics of the device to be valid.

**Table 38-2. General Operating Conditions**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $V_{DDANA}$ | Analog supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $T_A$ | Temperature range | | -40 | 25 | 105 | °C |
| $T_J$ | Junction temperature | | - | - | 125 | °C |

1. With BOD33 disabled. If the BOD33 is enabled, refer to the BOD33 characteristics.

**Related Links**

38.6.2.1 BOD33 Characteristics

## 38.4 Maximum Clock Frequencies

**Table 38-3. Maximum GCLK Generator Output Frequencies**

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN0}$ / $f_{GCLK_MAIN}$ | GCLK Generator Output Frequency | Undivided | 96 | MHz |
| $f_{GCLKGEN1}$ | | | | |
| $f_{GCLKGEN2}$ | | | | |
| $f_{GCLKGEN3}$ | | Divided | 48 | MHz |
| $f_{GCLKGEN4}$ | | | | |
| $f_{GCLKGEN5}$ | | | | |

**Table 38-4. Maximum Peripheral Clock Frequencies**

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{CPU}$ | CPU clock frequency | 48 | MHz |
| $f_{AHB}$ | AHB clock frequency | 48 | MHz |
| $f_{APBA}$ | APBA clock frequency | 48 | MHz |
| $f_{APBB}$ | APBB clock frequency | 48 | MHz |
| $f_{APBC}$ | APBC clock frequency | 48 | MHz |
| $f_{GCLK_DFLL48M_REF}$ | DFLL48M Reference clock frequency | 33 | KHz |
| $f_{GCLK_DPLL}$ | FDPLL96M Reference clock frequency | 2 | MHz |
| $f_{GCLK_DPLL_32K}$ | FDPLL96M 32k Reference clock frequency | 32 | KHz |
| $f_{GCLK_WDT}$ | WDT input clock frequency | 48 | MHz |
| $f_{GCLK_RTC}$ | RTC input clock frequency | 48 | MHz |
| $f_{GCLK_EIC}$ | EIC input clock frequency | 48 | MHz |
| $f_{GCLK_USB}$ | USB input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_0}$ | EVSYS channel 0 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_1}$ | EVSYS channel 1 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_2}$ | EVSYS channel 2 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_3}$ | EVSYS channel 3 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_4}$ | EVSYS channel 4 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_5}$ | EVSYS channel 5 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_6}$ | EVSYS channel 6 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_7}$ | EVSYS channel 7 input clock frequency | 48 | MHz |

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{GCLK_EVSYS_CHANNEL_8}$ | EVSYS channel 8 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_9}$ | EVSYS channel 9 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_10}$ | EVSYS channel 10 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_11}$ | EVSYS channel 11 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOMx_SLOW}$ | Common SERCOM slow input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM0_CORE}$ | SERCOM0 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM1_CORE}$ | SERCOM1 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM2_CORE}$ | SERCOM2 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM3_CORE}$ | SERCOM3 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM4_CORE}$ | SERCOM4 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM5_CORE}$ | SERCOM5 input clock frequency | 48 | MHz |
| $f_{GCLK_TCC0,\ GCLK_TCC1}$ | TCC0, TCC1 input clock frequency | 96 | MHz |
| $f_{GCLK_TCC2,\ GCLK_TC3}$ | TCC2,TC3 input clock frequency | 48 | MHz |
| $f_{GCLK_TC4,\ GCLK_TC5}$ | TC4, TC5 input clock frequency | 96 | MHz |
| $f_{GCLK_TC6,\ GCLK_TC7}$ | TC6,TC7 input clock frequency | 48 | MHz |
| $f_{GCLK_ADC}$ | ADC input clock frequency | 48 | MHz |
| $f_{GCLK_AC_DIG}$ | AC digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC_ANA}$ | AC analog input clock frequency | 64 | KHz |
| $f_{GCLK_AC1_DIG}$ | AC1 digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC1_ANA}$ | AC1 analog input clock frequency | 64 | KHz |
| $f_{GCLK_DAC}$ | DAC input clock frequency | 350 | KHz |
| $f_{GCLK_PTC}$ | PTC input clock frequency | 48 | MHz |
| $f_{GCLK_I2S_0}$ | I2S serial 0 input clock frequency | 13 | MHz |
| $f_{GCLK_I2S_1}$ | I2S serial 1 input clock frequency | 13 | MHz |

## 38.5 Power Consumption

The values in the below table are measured values of power consumption under the following conditions, except where noted:

- Operating conditions
  - $V_{VDDIN}$ = 3.3 V
- Wake up time from sleep mode is measured from the edge of the wakeup signal to the execution of the first instruction fetched in flash.
- Oscillators
  - XOSC32K (32 kHz crystal oscillator) stopped

  - XOSC (crystal oscillator) running with external 32MHz clock on XIN
  - DFLL48M stopped
- Clocks
  - XOSC used as main clock source, except otherwise specified
  - CPU, AHB clocks undivided
  - APBA clock divided by 4
  - APBB and APBC bridges off
- The following AHB module clocks are running: NVMCTRL, APBA bridge
  - All other AHB clocks stopped
- The following peripheral clocks running: PM, SYSCTRL, RTC
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait states
- Cache enabled
- BOD33 disabled

**Table 38-5. Current Consumption**

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ACTIVE | CPU running a While(1) algorithm | 105°C | - | 2.55 | 2.75 | mA |
| | CPU running a While(1) algorithm $V_{DDIN}$=1.8V, CPU is running on Flash with 3 wait states | | - | 2.56 | 2.82 | |
| | CPU running a While(1) algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | | - | 42*freq +318 | 42*freq +432 | µA (with freq in MHz) |
| | CPU running a Fibonacci algorithm | | - | 4.21 | 4.59 | mA |
| | CPU running a Fibonacci algorithm $V_{DDIN}$=1.8V, CPU is running on flash with 3 wait states | | - | 4.23 | 4.57 | |
| | CPU running a Fibonacci algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | | - | 80*freq +320 | 82*freq +432 | µA (with freq in MHz) |
| | CPU running a CoreMark algorithm | | - | 6.02 | 6.54 | mA |
| | CPU running a CoreMark algorithm $V_{DDIN}$=1.8V, CPU is running on flash with 3 wait states | | - | 5.21 | 5.57 | |

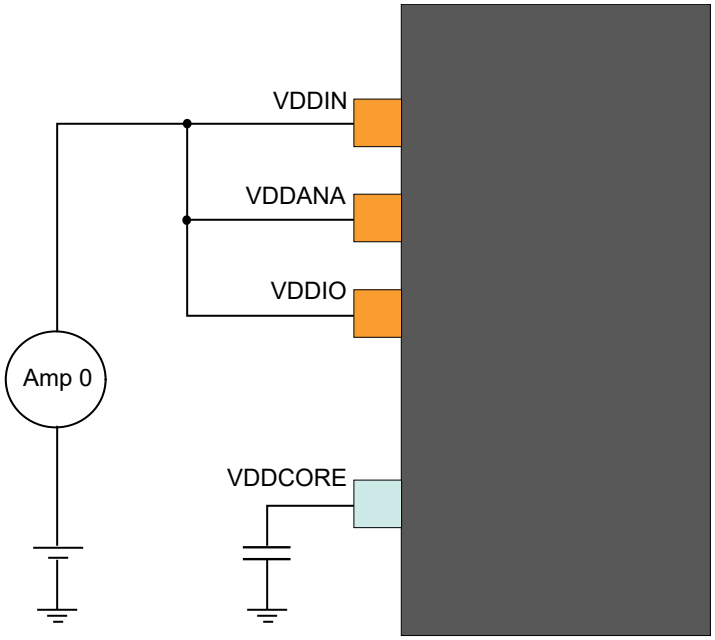
| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | CPU running a CoreMark algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | | - | 96*freq +322 | 98*freq +432 | µA (with freq in MHz) |
| IDLE0 | | | - | 1.55 | 1.62 | mA |
| IDLE1 | | | - | 1.13 | 1.18 | |
| IDLE2 | | | - | 0.96 | 1.01 | |
| STANDBY (rev. E silicon) | XOSC32K running / RTC running at 1kHz(1) | 105°C | - | 214 | 627 | µA |
| | XOSC32K and RTC stopped (1) | 105°C | - | 212 | 624 | |
| STANDBY (rev. F silicon) | XOSC32K running / RTC running at 1kHz (1) | 105°C | - | 175 | 452 | µA |
| | XOSC32K and RTC stopped (1) | 105°C | - | 173 | 450 | |

1. Measurements were done with SYSCTRL->VREG.bit.RUNSTDBY = 1

**Table 38-6. Wake-up Time**

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| IDLE0 | OSC8M used as main clock source, Cache disabled | 105°C | 3.8 | 4 | 4.1 | µs |
| IDLE1 | OSC8M used as main clock source, Cache disabled | | 12.8 | 14.3 | 15.7 | |
| IDLE2 | OSC8M used as main clock source, Cache disabled | | 13.7 | 15.2 | 16.6 | |
| STANDBY | OSC8M used as main clock source, Cache disabled | | 18.7 | 20.1 | 21.6 | |

**Figure 38-1. Measurement Schematic**



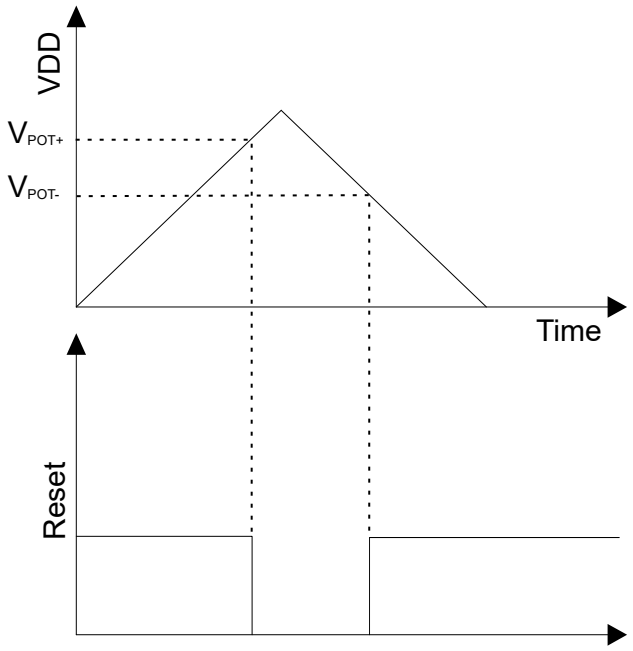

## 38.6 Analog Characteristics

### 38.6.1 Power-On Reset (POR) Characteristics

**Table 38-7. POR Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{POT+}$ | Voltage threshold on $V_{DD}$ rising | $IV_{DD}$ falls at 1V/ms or slower | 1.27 | 1.45 | 1.58 | V |
| $V_{POT-}$ | Voltage threshold on $V_{DD}$ falling | | 0.72 | 0.99 | 1.32 | V |

**Figure 38-2. POR Operating Principle**




### 38.6.2 Brown-Out Detectors Characteristics

#### 38.6.2.1 BOD33 Characteristics

**Table 38-8. BOD33 Characteristics**

| Symbol | Parameter | Conditions | Temp. | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| I | Step size, between adjacent values in BOD33.LEVEL | | | - | 34 | - | mV |
| $V_{HYST}$ | VBOD+ - VBOD- | Hysteresis ON | | 35 | - | 170 | mV |
| $t_{DET}$ | Detection time | Time with $V_{DDANA} < V_{TH}$ necessary to generate a reset signal | | - | 0.9(1) | - | µs |
| $I_{IdleBOD33}$ | Current consumption om Active/Idle mode | Continuous mode | 25°C | - | 25 | 48 | µA |
| | | | -40- to 105°C | - | - | 51 | |
| | | Sampling mode | 25°C | - | 0.034 | 0.21 | |
| | | | -40- to 105°C | - | - | 2.44 | |
| $I_{SbyBOD33}$ | Current consumption in Standby mode | Sampling mode | 25°C | | 0.132 | 0.38 | µA |

| Symbol | Parameter | Conditions | Temp. | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | | | -40- to 105°C | | | 1.5 | |
| $t_{STARTUP}$ | Startup time | | -40- to 105°C | - | 2.2(1) | - | µs |

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

### 38.6.3 Analog-to-Digital Converter (ADC) Characteristics

**Table 38-9. Operating Conditions**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| VDDANA | Power supply voltage | T > 85°C | 2.7 | | 3.6 | V |
| RES | Resolution | | 8 | - | 12 | bits |
| $f_{CLK_ADC}$ | ADC Clock frequency | | 30 | - | 2100 | kHz |
| | Sample rate(1) | Single shot | 5 | - | 300 | ksps |
| | | Free running | 5 | - | 350 | ksps |
| | Sampling time(1) | | 0.5 | - | - | cycles |
| | Conversion time(1) | 1x Gain | 6 | - | - | cycles |
| $V_{REF}$ | Voltage reference range | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| $V_{REFINT1V}$ | Internal 1V reference (2) | | - | 1.0 | - | V |
| $V_{REFINTVCC0}$ | Internal ratiometric reference 0(2) | | - | $V_{DDANA}$/ 1.48 | - | V |
| $V_{REFINTVCC0}$ Voltage Error | Internal ratiometric reference 0(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| $V_{REFINTVCC1}$ | Internal ratiometric reference 1(2) | $V_{DDANA}$>2.0V | - | $V_{DDANA}$/2 | - | V |
| $V_{REFINTVCC1}$ Voltage Error | Internal ratiometric reference 1(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| | Conversion range(1) | Differential mode | -$V_{REF}$/ GAIN | - | +$V_{REF}$/GAIN | V |
| | | Single-ended mode | 0.0 | - | +$V_{REF}$/GAIN | V |
| $C_{SAMPLE}$ | Sampling capacitance(2) | | - | 3.5 | - | pF |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $R_{SAMPLE}$ | Input channel source resistance(2) | | - | - | 3.5 | kΩ |
| $I_{DD}$ | DC supply current(1) | $f_{CLK_ADC}$ = 2.1MHz(3) | - | 1.25 | 1.83 | mA |

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.
3. In this condition and for a sample rate of 350ksps, a conversion takes 6 clock cycles of the ADC clock (conditions: 1X gain, 12-bit resolution, differential mode, free-running).

**Table 38-10. Differential Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number Of Bits | With gain compensation | - | 10.5 | 10.8 | bits |
| TUE | Total Unadjusted Error | 1x Gainn | 1.5 | 2.7 | 13 | LSB |
| INLI | Integral Non Linearity | 1x Gainn | 0.9 | 1.3 | 4 | LSB |
| DNL | Differential Non Linearity | 1x Gainn | +/-0.3 | +/-0.5 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref 1x | -10.0 | -1.3 | +10.0 | mV |
| | | $V_{REF}$=$V_{DDANA}$/1.48 | -20.0 | -10.0 | +10.0 | mV |
| | | Bandgap | -20.0 | +2 | +20.0 | mV |
| | Gain Accuracy(5) | Ext. Ref. 0.5x | +/-0.02 | +/-0.05 | +/-0.2 | % |
| | | Ext. Ref. 2x to 16x | +/-0.01 | +/-0.03 | +/-0.5 | % |
| | Offset Error | Ext. Ref. 1x | -8.0 | -1.0 | +8.0 | mV |
| | | $V_{REF}$=$V_{DDANA}$/1.48 | -8.0 | 0.6 | +8.0 | mV |
| | | Bandgap | -6.0 | -1.0 | +6.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 65.0 | 71.5 | 76.0 | dB |
| SINAD | Signal-to-Noise and Distortion | | 58.0 | 65.0 | 67.0 | dB |
| SNR | Signal-to-Noise Ratio | | 60.0 | 66.0 | 68.6 | dB |
| THD | Total Harmonic Distortion | | -75.0 | -71.0 | -67.0 | dB |
| | Noise RMS | T=25°C | 0.6 | 1.0 | 1.6 | mV |

1. Maximum numbers are based on characterization and not tested in production, and valid for 5% to 95% of the input voltage range.
2. Dynamic parameter numbers are based on characterization and not tested in production.
3. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage):
   3.1. If |VIN| > VREF/4

- VCM_IN < 0.95*VDDANA + VREF/4 – 0.75V
- VCM_IN > VREF/4 -0.05*VDDANA -0.1V

3.2. If |VIN| < VREF/4

- VCM_IN < 1.2*VDDANA - 0.75V
- VCM_IN > 0.2*VDDANA - 0.1V

4. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the VDDIO power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
5. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / (2*Vref/GAIN)

**Table 38-11. Single-Ended Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number of Bits | With gain compensation | - | 9.7 | 10.1 | Bits |
| TUE | Total Unadjusted Error | 1x gain | - | 7.9 | 30.0 | LSB |
| INL | Integral Non-Linearity | 1x gain | 1.4 | 2.6 | 5.0 | LSB |
| DNL | Differential Non-Linearity | 1x gain | +/-0.6 | +/-0.7 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref. 1x | -5.0 | 0.6 | 5.0 | mV |
| | Gain Accuracy(4) | Ext. Ref. 0.5x | +/-0.1 | +/-0.37 | +/-0.5 | % |
| | | Ext. Ref. 2x to 16X | +/-0.01 | +/-0.1 | +/-0.2 | % |
| | Offset Error | Ext. Ref. 1x | -5.0 | 0.6 | 10.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 63.0 | 68.0 | 70.1 | dB |
| SINAD | Signal-to-Noise and Distortion | | 55.0 | 60.1 | 62.5 | dB |
| SNR | Signal-to-Noise Ratio | | 54.0 | 61.0 | 64.0 | dB |
| THD | Total Harmonic Distortion | | -70.0 | -68.0 | -65.0 | dB |
| | Noise RMS | T = 25°C | - | 1.0 | - | mV |

1. Maximum numbers are based on characterization and not tested in production, and for 5% to 95% of the input voltage range.
2. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage) for all VIN:
   – VCM_IN < 0.7*VDDANA + VREF/4 – 0.75V
   – VCM_IN > VREF/4 – 0.3*VDDANA - 0.1V
3. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the VDDIO power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
4. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / (Vref/GAIN)

#### 38.6.3.1 Performance with the Averaging Digital Feature

Averaging is a feature which increases the sample accuracy. ADC automatically computes an average value of multiple consecutive conversions. The numbers of samples to be averaged is specified by the

Number-of-Samples-to-be-collected bit group in the Average Control register (AVGCTRL.SAMPLENUM[3:0]) and the averaged output is available in the Result register (RESULT).

**Table 38-12. Averaging Feature**

| Average Number | Condition | SNR(dB) | SINAD(dB) | SFDR(dB) | ENOB(bits) |
|---|---|---|---|---|---|
| 1 | In differential mode, 1x gain, VDDANA=3.0V, VREF=1.0V, 350kSps at 25°C | 66 | 65 | 72.8 | 10.5 |
| 8 | | 67.6 | 65.8 | 75.1 | 10.62 |
| 32 | | 69.7 | 67.1 | 75.3 | 10.85 |
| 128 | | 70.4 | 67.5 | 75.5 | 10.91 |

#### 38.6.3.2 Performance with the hardware offset and gain correction

Inherent gain and offset errors affect the absolute accuracy of the ADC. The offset error cancellation is handled by the Offset Correction register (OFFSETCORR) and the gain error cancellation, by the Gain Correction register (GAINCORR). The offset and gain correction value is subtracted from the converted data before writing the Result register (RESULT).

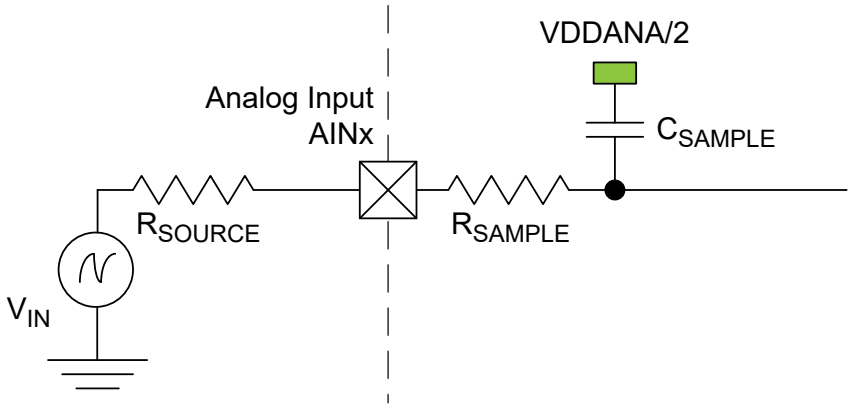
**Table 38-13. Offset and Gain Correction Feature**

| Gain Factor | Conditions | Offset Error (mV) | Gain Error (mV) | Total Unadjusted Error (LSB) |
|---|---|---|---|---|
| 0.5x | In differential mode, 1x gain, VDDANA=3.0V, VREF=1.0V, 350kSps at 25°C | 0.25 | 1 | 2.4 |
| 1x | | 0.2 | 0.1 | 1.5 |
| 2x | | 0.15 | -0.15 | 2.7 |
| 8x | | -0.05 | 0.05 | 3.2 |
| 16x | | 0.1 | -0.05 | 6.1 |

#### 38.6.3.3 Inputs and Sample and Hold Acquisition Times

The analog voltage source must be able to charge the sample and hold (S/H) capacitor in the ADC in order to achieve maximum accuracy. Seen externally the ADC input consists of a resistor ($R_{\mathrm{SAMPLE}}$) and a capacitor ($C_{\mathrm{SAMPLE}}$). In addition, the source resistance ($R_{\mathrm{SOURCE}}$) must be taken into account when calculating the required sample and hold time. The figure below shows the ADC input channel equivalent circuit.

**Figure 38-3. ADC Input**




To achieve n bits of accuracy, the $C_{SAMPLE}$ capacitor must be charged at least to a voltage of

$$V_{CSAMPLE} \geq V_{IN} \times \left(1 - 2^{-(n+1)}\right)$$

The minimum sampling time $t_{SAMPLEHOLD}$ for a given $R_{SOURCE}$can be found using this formula:

$$t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times (n+1) \times \ln(2)$$

for a 12 bits accuracy: $t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times 9.02$

where

$$t_{SAMPLEHOLD} = \frac{1}{2 \times f_{ADC}}$$

### 38.6.4 Digital to Analog Converter (DAC) Characteristics

**Table 38-14. Operating Conditions**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DDANA}$ | Analog supply voltage | | 1.62 | - | 3.63 | V |
| $AV_{REF}$ | External reference voltage | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| | Internal reference voltage 1 | | - | 1 | - | V |
| | Internal reference voltage 2 | | - | $V_{DDANA}$ | - | V |
| I | Linear output voltage range | | 0.05 | - | $V_{DDANA}$-0.05 | V |
| I | Minimum resistive load | | 5 | - | - | kΩ |
| I | Maximum capacitance load | | - | - | 100 | pF |
| $I_{DD}$ | DC supply current(2) | Voltage pump disabled | - | 160 | 245 | µA |

1. These values are based on specifications otherwise noted.
2. These values are based on characterization. These values are not covered by test limits in production.

**Table 38-15. Clock and Timing**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | Conversion rate | $C_{load}$=100pF $R_{load}$ > 5kΩ | Normal mode | - | - | 350 | ksps |
| | | | For $\Delta_{DATA}$=+/-1 | - | - | 1000 | |
| I | Startup time | $V_{DDNA}$ > 2.6V | | - | - | 2.85 | µs |
| | | $V_{DDNA}$ < 2.6V | | - | - | 10 | µs |

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

**Table 38-16. Accuracy Characteristics**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| RES | Input resolution | | | - | - | 10 | Bits |
| INL | Integral non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | 0.7 | 0.75 | 2.0 | LSB |
| | | | $V_{DD}$ = 3.6V | 0.6 | 0.65 | 1.5 | |
| | | $V_{REF}$ = $V_{DDANA}$ | $V_{DD}$ = 1.6V | 0.6 | 0.85 | 2.0 | |
| | | | $V_{DD}$ = 3.6V | 0.5 | 0.8 | 1.5 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | 0.5 | 0.75 | 1.5 | |
| | | | $V_{DD}$ = 3.6V | 0.7 | 0.8 | 1.5 | |
| DNL | Differential non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | +/-0.3 | +/-0.4 | +/-1.0 | LSB |
| | | | $V_{DD}$ = 3.6V | +/-0.25 | +/-0.4 | +/-0.75 | |
| | | $V_{REF}$= $V_{DDANA}$ | $V_{DD}$ = 1.6V | +/-0.4 | +/-0.55 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.2 | +/-0.3 | +/-0.75 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | +/-0.5 | +/-0.7 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.4 | +/-0.7 | +/-1.5 | |
| I | Gain error | Ext. $V_{REF}$ | | +/-0.5 | +/-5 | +/-12 | mV |
| I | Offset error | Ext. $V_{REF}$ | | +/-2 | +/-1.5 | +/-8 | mV |

1. All values measured using a conversion rate of 350ksps.

### 38.6.5 Analog Comparator Characteristics

**Table 38-17. Electrical and Timing Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Positive input voltage range | | 0 | - | $V_{DDANA}$ | V |
| | Negative input voltage range | | 0 | - | $V_{DDANA}$ | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Offset | Hysteresis = 0, Fast mode | -15 | 0.0 | +15 | mV |
| | | Hysteresis = 0, Low power mode | -25 | 0.0 | +25 | mV |
| | Hysteresis | Hysteresis = 1, Fast mode | 20 | 50 | 85 | mV |
| | | Hysteresis = 1, Low power mode | 15 | 40 | 75 | mV |
| | Propagation delay | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Fast mode | - | 90 | 180 | ns |
| | | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Low power mode | - | 282 | 520 | ns |
| $t_{STARTUP}$ | Startup time | Enable to ready delay Fast mode | - | 1 | 2.6 | µs |
| | | Enable to ready delay Low power mode | - | 14 | 22 | µs |
| $V_{SCALE}$ | INL(3) | | -1.4 | 0.75 | 1.4 | LSB |
| | DNL(3) | | -0.9 | 0.25 | 0.9 | LSB |
| | Offset Error (1)(2) | | -0.20 | 0.26 | +0.92 | LSB |
| | Gain Error (1)(2) | | -0.89 | 0.215 | 0.89 | LSB |

1. According to the standard equation $V(X)=V_{LSB}*(X+1)$; $V_{LSB}=V_{DDANA}/64$
2. Data computed with the Best Fit method
3. Data computed using histogram

## 38.7 NVM Characteristics

Note that on this flash technology, a max number of four consecutive write is allowed per row. Once this number is reached, a row erase is mandatory.

**Table 38-18. Flash Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{NVM25k}$ | Retention after up to 25k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{NVM2.5k}$ | Retention after up to 2.5k | Average ambient 55°C | 20 | 100 | - | Years |
| $Ret_{NVM100}$ | Retention after up to 100 | Average ambient 55°C | 25 | >100 | - | Years |
| $Cyc_{NVM}$ | Cycling Endurance(1) | -40°C < Ta < 105°C | 25k | 150k | - | Cycles |

1. An endurance cycle is a write and an erase operation.

**Table 38-19. EEPROM Emulation(1), Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{EEPROM100k}$ | Retention after up to 100k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{EEPROM10k}$ | Retention after up to 10k | Average ambient 55°C | 20 | 100 | - | Years |
| $Cyc_{EEPROM}$ | Cycling Endurance(2) | -40°C < Ta < 105°C | 100k | 600k | - | Cycles |

1. The EEPROM emulation is a software emulation described in the App note AT03265.
2. An endurance cycle is a write and an erase operation.

**Table 38-20. NVM Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{FPP}$ | Page programming time | - | - | - | 2.5 | ms |
| $t_{FRE}$ | Row erase time | - | - | - | 6 | ms |
| $t_{FCE}$ | DSU chip erase time (CHIP_ERASE) | - | - | - | 240 | ms |

## 38.8 Oscillators Characteristics

### 38.8.1 Crystal Oscillator (XOSC) Characteristics

#### 38.8.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

**Table 38-21. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN}$ | XIN clock frequency | | - | - | 32 | MHz |

#### 38.8.1.2 Crystal Oscillator Characteristics

The following table describes the characteristics for the oscillator when a crystal is connected between XIN and XOUT as shown in Figure 38-4. The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal datasheet. The capacitance of the external capacitors ($C_{LEXT}$) can then be computed as follows:

$$C_{LEXT} = 2(C_L - C_{STRAY} - C_{SHUNT})$$

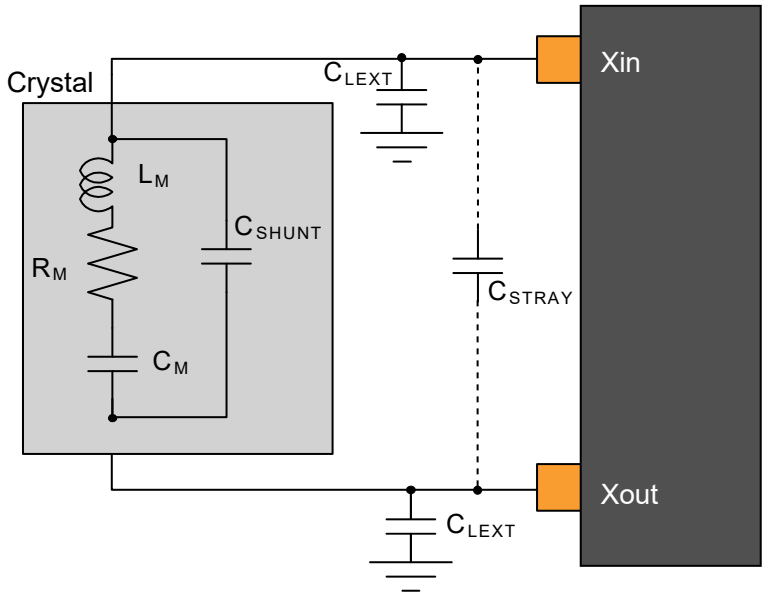
where $C_{STRAY}$ is the capacitance of the pins and PCB, $C_{SHUNT}$ is the shunt capacitance of the crystal.

**Table 38-22. Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | I | 0.4 | - | 32 | MHz |
| ESR | Crystal Equivalent Series Resistance<br>Safety Factor = 3 | f = 0.455 MHz, $C_L$ = 100pF<br>XOSC.GAIN = 0 | - | - | 5.6K | Ω |
| | | f = 2MHz, $C_L$ = 20pF<br>XOSC.GAIN = 0 | - | - | 416 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | The AGC doesn't have any noticeable impact on these measurements. | f = 4MHz, $C_L$ = 20pF XOSC.GAIN = 1 | - | - | 243 | |
| | | f = 8 MHz, $C_L$ = 20pF XOSC.GAIN = 2 | - | - | 138 | |
| | | f = 16 MHz, $C_L$ = 20pF XOSC.GAIN = 3 | - | - | 66 | |
| | | f = 32MHz, $C_L$ = 18pF XOSC.GAIN = 4 | - | - | 56 | |
| $C_{XIN}$ | Parasitic capacitor load | I | - | 5.9 | - | pF |
| $C_{XOUT}$ | Parasitic capacitor load | | - | 3.2 | - | pF |
| $I_{XOSC}$ | Current Consumption | f = 2MHz, $C_L$ = 20pF, AGC off | | 65 | 87 | µA |
| | | f = 2MHz, $C_L$ = 20pF, AGC on | | 52 | 76 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC off | | 117 | 155 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC on | | 74 | 104 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC off | | 226 | 308 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC on | | 128 | 180 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC off | | 502 | 714 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC on | | 307 | 590 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC off | | 1622 | 2257 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC on | | 615 | 1280 | |
| $t_{STARTUP}$ | Startup time | f = 2MHz, $C_L$ = 20pF, XOSC.GAIN = 0, ESR = 600Ω | - | 14K | 48K | cycles |
| | | f = 4MHz, $C_L$ = 20pF, XOSC.GAIN = 1, ESR = 100Ω | - | 6800 | 19.5K | |
| | | f = 8 MHz, $C_L$ = 20pF, XOSC.GAIN = 2, ESR = 35Ω | - | 5550 | 13K | |
| | | f = 16 MHz, $C_L$ = 20pF, XOSC.GAIN = 3, ESR = 25Ω | - | 6750 | 14.5K | |
| | | f = 32MHz, $C_L$ = 18pF, XOSC.GAIN = 4, ESR = 40Ω | - | 5.3K | 9.6K | |

**Figure 38-4. Crystal Oscillator Connection**




### 38.8.2 External 32 kHz Crystal Oscillator (XOSC32K) Characteristics

#### 38.8.2.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN32 pin.

**Table 38-23. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN32}$ | XIN32 clock frequency | | - | 32.768 | - | kHz |
| | XIN32 clock duty cycle | | - | 50 | - | % |

#### 38.8.2.2 Crystal Oscillator Characteristics

Figure 38-4 and the equation in 38.8.1.2 Crystal Oscillator Characteristics also applies to the 32 kHz oscillator connection. The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal datasheet.

**Table 38-24. 32kHz Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | I | - | 32768 | - | Hz |
| $t_{STARTUP}$ | Startup time | $ESR_{XTAL}$ = 39.9 kΩ, $C_L$ = 12.5 pF | - | 28K | 30K | cycles |
| $C_L$ | Crystal load capacitance | I | - | - | 12.5 | pF |
| $C_{SHUNT}$ | Crystal shunt capacitance | I | - | 0.1 | - | |
| $C_{XIN32}$ | Parasitic capacitor load | | - | 3.2 | - | |
| $C_{XOUT32}$ | Parasitic capacitor load | | - | 3.7 | - | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $I_{XOSC32K}$ | Current consumption | | - | 1.22 | 2.2 | µA |
| ESR | Crystal equivalent series resistance<br>f=32.768kHz<br>Safety Factor = 3 | $C_L$=12.5pF | - | - | 100 | kΩ |

### 38.8.3 Digital Frequency Locked Loop (DFLL48M) Characteristics

**Table 38-25. DFLL48M Characteristics - Open Loop Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | IDFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | 47 | 48 | 49 | MHz |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | IDFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | - | 403 | 453 | µA |
| $t_{STARTUP}$ | Startup time | DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>$f_{OUT}$ within 90 % of final value | | 8 | 9 | µs |

Note: 1. DFLL48M in Open loop after calibration at room temperature.

**Table 38-26. DFLL48M Characteristics - Closed Loop Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Average Output frequency | fREF = 32.768kHz | 47.76 | 48 | 48.24 | MHz |
| $f_{REF}$ | Reference frequency | I | 0.732 | 32.768 | 33 | kHz |
| Jitter | Period Jitter | fREF = 32.768kHz | - | - | 0.42 | ns |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | fREF = 32.768kHz | - | 403 | 453 | µA |
| $t_{LOCK}$ | Lock time | fREF = 32.768kHz<br>DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>DFLLCTRL.BPLCKC = 1<br>DFLLCTRL.QLDIS = 0<br>DFLLCTRL.CCDIS = 1<br>DFLLMUL.FSTEP = 10 | | 200 | 500 | µs |

### 38.8.4 32.768kHz Internal oscillator (OSC32K) Characteristics

**Table 38-27. 32kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +125]C, over [1.62, 3.63]V | 28.508 | 32.768 | 35.132 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 32.276 | 32.768 | 33.260 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.457 | 32.768 | 34.079 | |
| $I_{OSC32K}$ | Current consumption | I | - | 0.67 | 1.9 | µA |
| $t_{STARTUP}$ | Startup time | I | - | 1 | 2 | cycle |
| Duty | Duty Cycle | I | - | 50 | - | % |

### 38.8.5 Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics

**Table 38-28. Ultra Low Power Internal 32kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +125]C, over [1.62, 3.63]V | 25.559 | 32.768 | 40.305 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 31.293 | 32.768 | 34.570 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.293 | 32.768 | 34.570 | |
| Duty | Duty cycle | | - | 50 | - | % |

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. This oscillator is always on.

### 38.8.6 8MHz RC Oscillator (OSC8M) Characteristics

**Table 38-29. Internal 8MHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | ICalibrated against a 8MHz reference at 25°C, over [-40, +125]C, over [1.62, 3.63]V | 7.65 | 8 | 8.17 | MHz |
| | | Calibrated against a 8MHz reference at 25°C, at $V_{DD}$=3.3V | 7.94 | 8 | 8.06 | |
| | | Calibrated against a 8MHz reference at 25°C, over [1.62, 3.63]V | 7.92 | 8 | 8.08 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| TempCo | Frequency vs. temperature drift | | -1.2 | | 1 | |
| SupplyCo | Frequency vs. supply drift | | -2 | | 2 | |
| $I_{OSC8M}$ | Current consumption | IIDLEIDLE2 on OSC32K versus IDLE2 on calibrated OSC8M enabled at 8MHz (FRANGE=1, PRESC=0) | - | 64 | 96 | µA |
| $t_{STARTUP}$ | Startup time | I | - | 2.4 | 3.3 | µs |
| Duty | Duty cycle | I | - | 50 | - | % |

### 38.8.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics

**Table 38-30. FDPLL96M Characteristics (Variant B and L With Silicon Revision E)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | | 500 | 733 | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | | 900 | 1235 | |
| Jp | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 2.1 | 3.2 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | | 4.0 | 6.9 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | | 2.2 | 3.6 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | | 4.7 | 8.2 | |
| $t_{LOCK}$ | Lock Time | After startup, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | | 1.2 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Note:** All values have been characterized with FILTSEL[1/0] as default value.

**Table 38-31. FDPLL96M Characteristics(1) (Device Variant B, C and L With Silicon Revision F)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | - | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | - | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | - | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | - | |
| Jp | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 2.1 | 3.0 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 3.8 | 9.2 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 2.2 | 3.2 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 4.4 | 10.0 | |
| $t_{LOCK}$ | Lock Time | After startup, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.2 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | - | 40 | 50 | 60 | % |

**Note:** 1. All values have been characterized with FILTSEL[1/0] as default value.

## 38.9 USB Characteristics

The USB shares the same characteristics as in the -40°C to 85°C.

## 38.10 Timing Characteristics

### 38.10.1 SERCOM in SPI Mode Timing

Data are not available in this current datasheet revision

**Related Links**

37.15.2 SERCOM in SPI Mode Timing

### 38.10.2 SERCOM in $I^2C$ Mode Timing

Data are not available in this current datasheet revision

**Related Links**

37.15.3 SERCOM in I2C Mode Timing

# 39. Electrical Characteristics at 125°C

## 39.1 Disclaimer

All typical values are measured at T = 25°C unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

## 39.2 Absolute Maximum Ratings

Stresses beyond those listed in the table below may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**Table 39-1. Absolute Maximum Ratings**

| Symbol | Parameter | Condition | Min. | Max. | Units |
|---|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | | 0 | 3.8 | V |
| $I_{VDD}$ | Current into a $V_{DD}$ pin | | - | 28(1) | mA |
| $I_{GND}$ | Current out of a GND pin | | - | 39(1) | mA |
| $V_{PIN}$ | Pin voltage with respect to GND and $V_{DD}$ | | GND-0.6V | $V_{DD}$+0.6V | V |
| $T_{storage}$ | Storage temperature | | -60 | 150 | °C |

**Note:**

1. Maximum source current is 14mA and maximum sink current is 19.5mA per cluster. A cluster is a group of GPIOs, see related links. Also note that each VDD/GND pair is connected to 2 clusters so current consumption through the pair will be a sum of the clusters source/sink currents.

**Related Links**

7.2.4 GPIO Clusters

## 39.3 General Operating Ratings

The device must operate within the ratings in order for all other electrical characteristics and typical characteristics of the device to be valid.

**Table 39-2. General Operating Conditions**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DD}$ | Power supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $V_{DDANA}$ | Analog supply voltage | | 1.62(1) | 3.3 | 3.63 | V |
| $T_A$ | Temperature range | | -40 | 25 | 125 | °C |
| $T_J$ | Junction temperature | | - | - | 145 | °C |

**Note:**

Notes: 1. With BOD33 disabled. If the BOD33 is enabled, check Table 37-20.

## 39.4 Maximum Clock Frequencies

**Table 39-3. Maximum GCLK Generator Output Frequencies (Device Variant A)**

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN0}$ / $f_{GCLK_MAIN}$ | GCLK Generator Output Frequency | Undivided | 96 | MHz |
| $f_{GCLKGEN1}$<br>$f_{GCLKGEN2}$<br>$f_{GCLKGEN3}$<br>$f_{GCLKGEN4}$<br>$f_{GCLKGEN5}$<br>$f_{GCLKGEN6}$<br>$f_{GCLKGEN7}$<br>$f_{GCLKGEN8}$ | | Divided | 32 | MHz |

**Table 39-4. Maximum Peripheral Clock Frequencies (Device Variant A)**

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{CPU}$ | CPU clock frequency | 32 | MHz |
| $f_{AHB}$ | AHB clock frequency | 32 | MHz |
| $f_{APBA}$ | APBA clock frequency | 32 | MHz |
| $f_{APBB}$ | APBB clock frequency | 32 | MHz |
| $f_{APBC}$ | APBC clock frequency | 32 | MHz |
| $f_{GCLK_DFLL48M_REF}$ | DFLL48M Reference clock frequency | 33 | KHz |
| $f_{GCLK_DPLL}$ | FDPLL96M Reference clock frequency | 2 | MHz |
| $f_{GCLK_DPLL_32K}$ | FDPLL96M 32k Reference clock frequency | 32 | KHz |
| $f_{GCLK_WDT}$ | WDT input clock frequency | 48 | MHz |
| $f_{GCLK_RTC}$ | RTC input clock frequency | 48 | MHz |
| $f_{GCLK_EIC}$ | EIC input clock frequency | 48 | MHz |
| $f_{GCLK_USB}$ | USB input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_0}$ | EVSYS channel 0 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_1}$ | EVSYS channel 1 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_2}$ | EVSYS channel 2 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_3}$ | EVSYS channel 3 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_4}$ | EVSYS channel 4 input clock frequency | 48 | MHz |

© 2018 Microchip Technology Inc.

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{GCLK_EVSYS_CHANNEL_5}$ | EVSYS channel 5 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_6}$ | EVSYS channel 6 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_7}$ | EVSYS channel 7 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_8}$ | EVSYS channel 8 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_9}$ | EVSYS channel 9 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_10}$ | EVSYS channel 10 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_11}$ | EVSYS channel 11 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOMx_SLOW}$ | Common SERCOM slow input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM0_CORE}$ | SERCOM0 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM1_CORE}$ | SERCOM1 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM2_CORE}$ | SERCOM2 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM3_CORE}$ | SERCOM3 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM4_CORE}$ | SERCOM4 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM5_CORE}$ | SERCOM5 input clock frequency | 48 | MHz |
| $f_{GCLK_TCC0,\ GCLK_TCC1}$ | TCC0, TCC1 input clock frequency | 80 | MHz |
| $f_{GCLK_TCC2,\ GCLK_TC3}$ | TCC2, TC3 input clock frequency | 80 | MHz |
| $f_{GCLK_TC4,\ GCLK_TC5}$ | TC4, TC5 input clock frequency | 48 | MHz |
| $f_{GCLK_TC6,\ GCLK_TC7}$ | TC6, TC7 input clock frequency | 48 | MHz |
| $f_{GCLK_ADC}$ | ADC input clock frequency | 48 | MHz |
| $f_{GCLK_AC_DIG}$ | AC digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC_ANA}$ | AC analog input clock frequency | 64 | KHz |
| $f_{GCLK_DAC}$ | DAC input clock frequency | 350 | KHz |
| $f_{GCLK_PTC}$ | PTC input clock frequency | 48 | MHz |
| $f_{GCLK_I2S_0}$ | I2S serializer 0 input clock frequency | 13 | MHz |
| $f_{GCLK_I2S_1}$ | I2S serializer 1 input clock frequency | 13 | MHz |

**Table 39-5. Maximum GCLK Generator Output Frequencies (Device Variant B)**

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN0}$ / $f_{GCLK_MAIN}$ | GCLK Generator Output Frequency | Undivided | 96 | MHz |
| $f_{GCLKGEN1}$<br>$f_{GCLKGEN2}$<br>$f_{GCLKGEN3}$<br>$f_{GCLKGEN4}$ | | Divided | 48 | MHz |

| Symbol | Description | Conditions | Max. | Units |
|---|---|---|---|---|
| $f_{GCLKGEN5}$ | | | | |
| $f_{GCLKGEN6}$ | | | | |
| $f_{GCLKGEN7}$ | | | | |
| $f_{GCLKGEN8}$ | | | | |

**Table 39-6. Maximum Peripheral Clock Frequencies (Device Variant B)**

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{CPU}$ | CPU clock frequency | 48 | MHz |
| $f_{AHB}$ | AHB clock frequency | 48 | MHz |
| $f_{APBA}$ | APBA clock frequency | 48 | MHz |
| $f_{APBB}$ | APBB clock frequency | 48 | MHz |
| $f_{APBC}$ | APBC clock frequency | 48 | MHz |
| $f_{GCLK_DFLL48M_REF}$ | DFLL48M Reference clock frequency | 33 | KHz |
| $f_{GCLK_DPLL}$ | FDPLL96M Reference clock frequency | 2 | MHz |
| $f_{GCLK_DPLL_32K}$ | FDPLL96M 32k Reference clock frequency | 32 | KHz |
| $f_{GCLK_WDT}$ | WDT input clock frequency | 48 | MHz |
| $f_{GCLK_RTC}$ | RTC input clock frequency | 48 | MHz |
| $f_{GCLK_EIC}$ | EIC input clock frequency | 48 | MHz |
| $f_{GCLK_USB}$ | USB input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_0}$ | EVSYS channel 0 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_1}$ | EVSYS channel 1 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_2}$ | EVSYS channel 2 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_3}$ | EVSYS channel 3 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_4}$ | EVSYS channel 4 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_5}$ | EVSYS channel 5 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_6}$ | EVSYS channel 6 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_7}$ | EVSYS channel 7 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_8}$ | EVSYS channel 8 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_9}$ | EVSYS channel 9 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_10}$ | EVSYS channel 10 input clock frequency | 48 | MHz |
| $f_{GCLK_EVSYS_CHANNEL_11}$ | EVSYS channel 11 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOMx_SLOW}$ | Common SERCOM slow input clock frequency | 48 | MHz |

| Symbol | Description | Max. | Units |
|---|---|---|---|
| $f_{GCLK_SERCOM0_CORE}$ | SERCOM0 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM1_CORE}$ | SERCOM1 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM2_CORE}$ | SERCOM2 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM3_CORE}$ | SERCOM3 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM4_CORE}$ | SERCOM4 input clock frequency | 48 | MHz |
| $f_{GCLK_SERCOM5_CORE}$ | SERCOM5 input clock frequency | 48 | MHz |
| $f_{GCLK_TCC0,\ GCLK_TCC1}$ | TCC0, TCC1 input clock frequency | 96 | MHz |
| $f_{GCLK_TCC2,\ GCLK_TC3}$ | TCC2, TC3 input clock frequency | 48 | MHz |
| $f_{GCLK_TC4,\ GCLK_TC5}$ | TC4, TC5 input clock frequency | 96 | MHz |
| $f_{GCLK_TC6,\ GCLK_TC7}$ | TC6, TC7 input clock frequency | 48 | MHz |
| $f_{GCLK_ADC}$ | ADC input clock frequency | 48 | MHz |
| $f_{GCLK_AC_DIG}$ | AC digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC_ANA}$ | AC analog input clock frequency | 64 | KHz |
| $f_{GCLK_AC1_DIG}$ | AC1 digital input clock frequency | 48 | MHz |
| $f_{GCLK_AC1_ANA}$ | AC1 analog input clock frequency | 64 | KHz |
| $f_{GCLK_DAC}$ | DAC input clock frequency | 350 | KHz |
| $f_{GCLK_PTC}$ | PTC input clock frequency | 48 | MHz |
| $f_{GCLK_I2S_0}$ | I2S serializer 0 input clock frequency | 13 | MHz |
| $f_{GCLK_I2S_1}$ | I2S serializer 1 input clock frequency | 13 | MHz |

## 39.5 Power Consumption

The values in this section are measured values of power consumption under the following conditions, except where noted:

- Operating conditions
  - $V_{VDDIN}$ = 3.3 V
- Wake up time from sleep mode is measured from the edge of the wakeup signal to the execution of the first instruction fetched in flash.
- Oscillators
  - XOSC (crystal oscillator) stopped
  - XOSC32K (32 kHz crystal oscillator) running with external 32kHz crystal
  - DFLL48M using XOSC32K as reference and running at 48 MHz
- Clocks
  - DFLL48M used as main clock source, except otherwise specified
  - CPU, AHB clocks undivided
  - APBA clock divided by 4

  - APBB and APBC bridges off
- The following AHB module clocks are running: NVMCTRL, APBA bridge
  - All other AHB clocks stopped
- The following peripheral clocks running: PM, SYSCTRL, RTC
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait states
- Cache enabled
- BOD33 disabled

**Table 39-7. Current Consumption**

| Mode | Conditions | $T_A$ | Typ. | Max. | Units |
|---|---|---|---|---|---|
| ACTIVE | CPU running a While(1) algorithm | 125°C | 3.75 | 4.12 | mA |
| | CPU running a While(1) algorithm $V_{DDIN}$=1.8V, CPU is running on Flash with 3 wait states | 125°C | 3.77 | 4.13 | |
| | CPU running a While(1) algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | 125°C | 62*freq + 228 | 62*freq + 302 | µA (with freq in MHz) |
| | CPU running a Fibonacci algorithm | 125°C | 4.85 | 5.29 | mA |
| | CPU running a Fibonacci algorithm $V_{DDIN}$=1.8V, CPU is running on flash with 3 wait states | 125°C | 4.87 | 5.29 | |
| | CPU running a Fibonacci algorithm, CPU is running on Flash with 3 wait states with GCLKIN as reference | 125°C | 88*freq + 424 | 88*freq + 486 | µA (with freq in MHz) |
| | CPU running a CoreMark algorithm | 125°C | 6.70 | 7.30 | mA |
| | CPU running a CoreMark algorithm $V_{DDIN}$=1.8V, CPU is running on flash with 3 wait states | 125°C | 5.98 | 6.41 | |
| | CPU running a CoreMark algorithm, CPU is running on Flash with 3 wait states | 125°C | 108*freq + 426 | 108*freq + 492 | µA (with freq in MHz) |

| Mode | Conditions | $T_A$ | Typ. | Max. | Units |
|---|---|---|---|---|---|
| | with<br>GCLKIN as reference | | | | |
| IDLE0 | Default operating conditions | 125°C | 2.40 | 2.69 | mA |
| IDLE1 | Default operating conditions | 125°C | 1.79 | 2.05 | |
| IDLE2 | Default operating conditions | 125°C | 1.50 | 1.76 | |
| STANDBY<br>(Device Variant B, Die Revision E) | XOSC32K running<br>, RTC running at 1kHz(1) | 125°C | 348.0 | 850.0 | µA |
| | XOSC32K and RTC stopped(1) | 125°C | 346.0 | 848.0 | |
| STANDBY<br>(Device Variant B and C, Die Revision F) | XOSC32K running<br>, RTC running at 1kHz(1) | 125°C | 294.0 | 782.0 | µA |
| | XOSC32K and RTC stopped(1) | 125°C | 292.0 | 780.0 | |

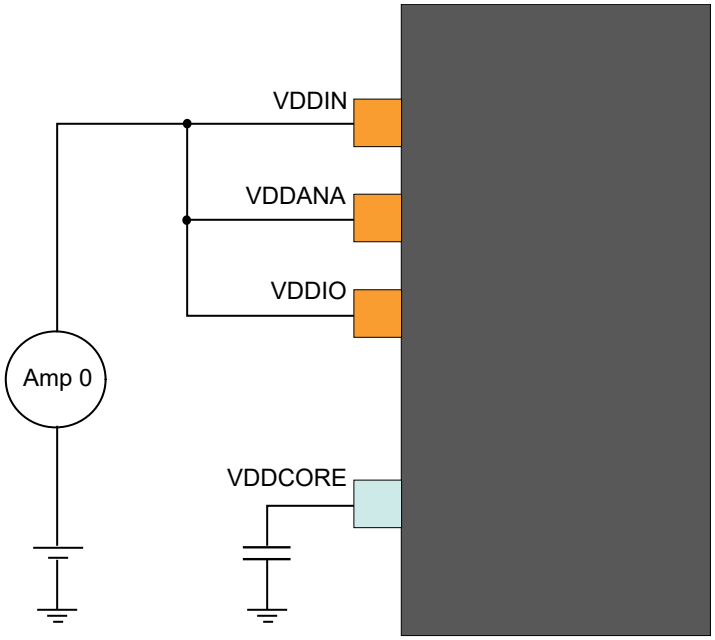
**Note:**

1. Measurements were done with SYSCTRL->VREG.bit.RUNSTDBY = 1

**Table 39-8. Wake-up Time (SAMD21)**

| Mode | Conditions | $T_A$ | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| IDLE0 | OSC8M used as main clock source, Cache disabled | 125°C | 3.9 | 4.0 | 4.1 | µs |
| IDLE1 | OSC8M used as main clock source, Cache disabled | 125°C | 13.5 | 14.9 | 16.4 | |
| IDLE2 | OSC8M used as main clock source, Cache disabled | 125°C | 14.4 | 15.8 | 17.2 | |
| STANDBY | OSC8M used as main clock source, Cache disabled | 125°C | 19.2 | 20.6 | 22.1 | |

**Figure 39-1. Measurement Schematic**



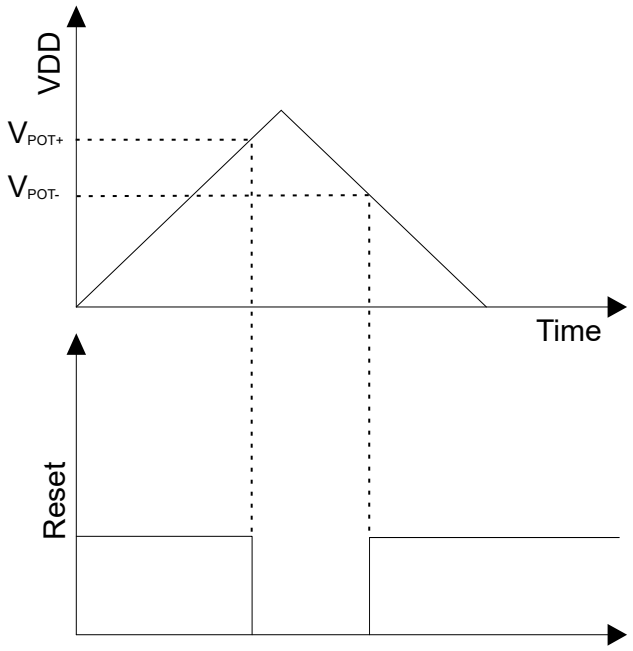

## 39.6 Analog Characteristics

### 39.6.1 Power-On Reset (POR) Characteristics

**Table 39-9. POR Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{POT+}$ | Voltage threshold on $V_{DD}$ rising | $V_{DD}$ falls at 1V/ms or slower | 1.27 | 1.45 | 1.58 | V |
| $V_{POT-}$ | Voltage threshold on $V_{DD}$ falling | | 0.72 | 0.99 | 1.32 | V |

**Figure 39-2. POR Operating Principle**




### 39.6.2 Brown-Out Detectors Characteristics

#### 39.6.2.1 BOD33

**Table 39-10. BOD33 Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Temp. | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | Step size, between adjacent values in BOD33.LEVEL | | | - | 34 | - | mV |
| $V_{HYST}$ | $V_{BOD+}$ - $V_{BOD-}$ | Hysteresis ON | | 35 | - | 170 | mV |
| $t_{DET}$ | Detection time | Time with $V_{DDANA}$ < $V_{TH}$ necessary to generate a reset signal | | - | 0.9(1) | - | µs |
| $I_{BOD33}$ | Current consumption | Continuous mode | 25°C | - | 25 | 48 | µA |
| | | | -40 to 125°C | - | - | 50 | |
| | | Sampling mode | 25°C | - | 0.034 | 0.21 | |
| | | | -40 to 125°C | -- | - | 2.29 | |
| $I_{SbyBOD33}$ | Current consumption in Standby mode | Sampling mode | 25°C | - | 0.132 | 0.38 | µA |

| Symbol | Parameter | Conditions | Temp. | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | | | -40 to 125°C | - | - | 1.62 | |
| $t_{STARTUP}$ | Start-up time | | | - | 1.2(1) | - | µs |

**Note:** 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

**Table 39-11. BOD33 Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Temp. | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | Step size, between adjacent values in BOD33.LEVEL | | | - | 34 | - | mV |
| $V_{HYST}$ | $V_{BOD+}$ - $V_{BOD-}$ | Hysteresis ON | | 35 | - | 170 | mV |
| $t_{DET}$ | Detection time | Time with $V_{DDANA}$ < $V_{TH}$ necessary to generate a reset signal | | - | 0.9(1) | - | µs |
| $I_{BOD33}$ | Current consumption | Continuous mode | 25°C | - | 25 | 48 | µA |
| | | | -40 to 125°C | - | - | 52 | |
| | | Sampling mode | 25°C | - | 0.03 | 0.21 | |
| | | | -40 to 125°C | -- | - | 2.91 | |
| $I_{SbyBOD33}$ | Current consumption in Standby mode | Sampling mode | 25°C | - | 0.13 | 0.38 | µA |
| | | | -40 to 125°C | - | - | 1.7 | |
| $t_{STARTUP}$ | Start-up time | | | - | 2.2(1) | - | µs |

### 39.6.3 Analog-to-Digital (ADC) characteristics

**Table 39-12. Operating Conditions (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| RES | Resolution | | 8 | - | 12 | bits |
| $f_{CLK_ADC}$ | ADC Clock frequency | | 30 | - | 2100 | kHz |
| | Sample rate(1) | Single shot (with VDDANA > 3.0V) (4) | 5 | - | 300 | ksps |
| | | Free running | 5 | - | 350 | ksps |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Sampling time(1) | | 0.5 | - | - | cycles |
| | Conversion time(1) | 1x Gain | - | 6 | - | cycles |
| $V_{REF}$ | Voltage reference range | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| $V_{REFINT1V}$ | Internal 1V reference (2) | | - | 1.0 | - | V |
| $V_{REFINTVCC0}$ | Internal ratiometric reference 0(2) | | - | $V_{DDANA}$/ 1.48 | - | V |
| $V_{REFINTVCC0}$ Voltage Error | Internal ratiometric reference 0(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| $V_{REFINTVCC1}$ | Internal ratiometric reference 1(2) | $V_{DDANA}$>2.0V | - | $V_{DDANA}$/2 | - | V |
| $V_{REFINTVCC1}$ Voltage Error | Internal ratiometric reference 1(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| | Conversion range(1) | Differential mode | -$V_{REF}$/ GAIN | - | +$V_{REF}$/GAIN | V |
| | | Single-ended mode | 0.0 | - | +$V_{REF}$/GAIN | V |
| $C_{SAMPLE}$ | Sampling capacitance(2) | | - | 3.5 | - | pF |
| $R_{SAMPLE}$ | Input channel source resistance(2) | | - | - | 3.5 | kΩ |
| $I_{DD}$ | DC supply current(1) | $f_{CLK_ADC}$ = 2.1MHz(3) | - | 1.25 | 1.85 | mA |

**Note:**

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.
3. In this condition and for a sample rate of 350ksps, 1 Conversion at gain 1x takes 6 clock cycles of the ADC clock (conditions: 1X gain, 12-bit resolution, differential mode, free-running).
4. All single-shot measurements are performed with $V_{DDANA}$ > 3.0V.

**Table 39-13. Operating Conditions (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| RES | Resolution | | 8 | - | 12 | bits |
| $f_{CLK_ADC}$ | ADC Clock frequency | | 30 | - | 2100 | kHz |
| | Conversion speed | | 10 | | 1000 | ksps |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Sample rate(1) | Single shot | 5 | - | 300 | ksps |
| | | Free running | 5 | - | 350 | ksps |
| | Sampling time(1) | | 0.5 | - | - | cycles |
| | Conversion time(1) | 1x Gain | 6 | - | - | cycles |
| $V_{REF}$ | Voltage reference range | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| $V_{REFINT1V}$ | Internal 1V reference (2) | | - | 1.0 | - | V |
| $V_{REFINTVCC0}$ | Internal ratiometric reference 0(2) | | - | $V_{DDANA}$/ 1.48 | - | V |
| $V_{REFINTVCC0}$ Voltage Error | Internal ratiometric reference 0(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| $V_{REFINTVCC1}$ | Internal ratiometric reference 1(2) | $V_{DDANA}$>2.0V | - | $V_{DDANA}$/2 | - | V |
| $V_{REFINTVCC1}$ Voltage Error | Internal ratiometric reference 1(2) error | 2.0V < $V_{DDANA}$<3.63V | -1.0 | - | +1.0 | % |
| | Conversion range(1) | Differential mode | -$V_{REF}$/ GAIN | - | +$V_{REF}$/GAIN | V |
| | | Single-ended mode | 0.0 | - | +$V_{REF}$/GAIN | V |
| $C_{SAMPLE}$ | Sampling capacitance(2) | | - | 3.5 | - | pF |
| $R_{SAMPLE}$ | Input channel source resistance(2) | | - | - | 3.5 | kΩ |
| $I_{DD}$ | DC supply current(1) | $f_{CLK_ADC}$ = 2.1MHz(3) | - | 1.25 | 1.85 | mA |

**Note:**

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.
3. In this condition and for a sample rate of 350ksps, 1 Conversion at gain 1x takes 6 clock cycles of the ADC clock (conditions: 1X gain, 12-bit resolution, differential mode, free-running).

**Table 39-14. Differential Mode (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number Of Bits | With gain compensation | - | 10.5 | 10.9 | bits |
| TUE | Total Unadjusted Error | 1x Gainn | 1.5 | 4.3 | 17.0 | LSB |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| INLI | Integral Non Linearity | 1x Gainn | 1.0 | 1.3 | 6.5 | LSB |
| DNL | Differential Non Linearity | 1x Gainn | +/-0.3 | +/-0.5 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref 1x | -15.0 | 2.5 | +20.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -20.0 | -1.5 | +20.0 | mV |
| | | Bandgap | -15.0 | -5.0 | +15.0 | mV |
| | Gain Accuracy(5) | Ext. Ref. 0.5x | +/-0.1 | +/-0.2 | +/-0.45 | % |
| | | Ext. Ref. 2x to 16x | +/-0.1 | +/-0.2 | +/-2.0 | % |
| | Offset Error | Ext. Ref. 1x | -10.0 | -1.5 | +10.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -10.0 | 0.5 | +15.0 | mV |
| | | Bandgap | -10.0 | 3.0 | +15.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 64.2 | 70.0 | 78.9 | dB |
| SINAD | Signal-to-Noise and Distortion | | 64.1 | 65.0 | 66 | dB |
| SNR | Signal-to-Noise Ratio | | 64.3 | 65.5 | 66.0 | dB |
| THD | Total Harmonic Distortion | | -74.8 | -64.0 | -65.0 | dB |
| | Noise RMS | T=25°C | 0.6 | 1.0 | 1.6 | mV |

**Table 39-15. Differential Mode (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number Of Bits | With gain compensation | - | 10.5 | 10.8 | bits |
| TUE | Total Unadjusted Error | 1x Gainn | 1.5 | 2.7 | 14 | LSB |
| INLI | Integral Non Linearity | 1x Gainn | 0.9 | 1.3 | 4 | LSB |
| DNL | Differential Non Linearity | 1x Gainn | +/-0.3 | +/-0.5 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref 1x | -10.0 | -1.3 | +10 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -25.0 | -10.1 | +10.0 | mV |
| | | Bandgap | -25.0 | +2 | +10.0 | mV |
| | Gain Accuracy(5) | Ext. Ref. 0.5x | +/-0.005 | +/-0.05 | +/-0.15 | % |
| | | Ext. Ref. 2x to 16x | +/-0.1 | +/-0.03 | +/-0.5 | % |
| | Offset Error | Ext. Ref. 1x | -8.0 | -1.0 | +8.0 | mV |
| | | $V_{REF}=V_{DDANA}/1.48$ | -8.0 | 0.6 | +8.0 | mV |
| | | Bandgap | -6.0 | -1.0 | +8.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz | 65.0 | 71.5 | 73.5 | dB |
| SINAD | Signal-to-Noise and Distortion | | 58.0 | 65.0 | 67.0 | dB |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| SNR | Signal-to-Noise Ratio | $A_{IN}$ = 95%FSR | 60.0 | 66 | 68.6 | dB |
| THD | Total Harmonic Distortion | | -73.0 | -71.0 | -67.0 | dB |
| | Noise RMS | T=25°C | 0.6 | 1.0 | 1.6 | mV |

**Note:**

1. Maximum numbers are based on characterization and not tested in production, and valid for 5% to 95% of the input voltage range.
2. Dynamic parameter numbers are based on characterization and not tested in production.
3. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage):
   - 1.1. If |VIN| > VREF/4
     - VCM_IN < 0.95*VDDANA + VREF/4 – 0.75V
     - VCM_IN > VREF/4 -0.05*VDDANA -0.1V
   - 1.2. If |VIN| < VREF/4
     - VCM_IN < 1.2*VDDANA - 0.75V
     - VCM_IN > 0.2*VDDANA - 0.1V
4. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the $V_{DDIO}$ power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
5. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / ($2*V_{ref}$/GAIN)

**Table 39-16. Single-Ended Mode (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number of Bits | With gain compensation | - | 9.5 | 9.8 | Bits |
| TUE | Total Unadjusted Error | 1x gain | - | 10.5 | 14.0 | LSB |
| INL | Integral Non-Linearity | 1x gain | 1.0 | 1.6 | 7.5 | LSB |
| DNL | Differential Non-Linearity | 1x gain | +/-0.5 | +/-0.6 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref. 1x | -10.0 | 0.7 | +10.0 | mV |
| | Gain Accuracy(4) | Ext. Ref. 0.5x | +/-0.1 | +/-0.34 | +/-0.4 | % |
| | | Ext. Ref. 2x to 16X | +/-0.01 | +/-0.1 | +/-0.15 | % |
| | Offset Error | Ext. Ref. 1x | -5.0 | 1.5 | +10.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 63.1 | 65.0 | 66.5 | dB |
| SINAD | Signal-to-Noise and Distortion | | 50.7 | 59.5 | 61.0 | dB |
| SNR | Signal-to-Noise Ratio | | 49.9 | 60.0 | 64.0 | dB |
| THD | Total Harmonic Distortion | | -65.4 | -63.0 | -62.1 | dB |
| | Noise RMS | T = 25°C | - | 1.0 | - | mV |

**Table 39-17. Single-Ended Mode (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| ENOB | Effective Number of Bits | With gain compensation | - | 9.7 | 10.1 | Bits |
| TUE | Total Unadjusted Error | 1x gain | - | 7.9 | 40.0 | LSB |
| INL | Integral Non-Linearity | 1x gain | 1.4 | 2.6 | 6.0 | LSB |
| DNL | Differential Non-Linearity | 1x gain | +/-0.6 | +/-0.7 | +/-0.95 | LSB |
| | Gain Error | Ext. Ref. 1x | -5.0 | 0.6 | +5.0 | mV |
| | Gain Accuracy(4) | Ext. Ref. 0.5x | +/-0.1 | +/-0.37 | +/-0.55 | % |
| | | Ext. Ref. 2x to 16X | +/-0.01 | +/-0.1 | +/-0.2 | % |
| | Offset Error | Ext. Ref. 1x | -5.0 | 0.6 | +10.0 | mV |
| SFDR | Spurious Free Dynamic Range | 1x Gain<br>$F_{CLK_ADC}$ = 2.1MHz<br>$F_{IN}$ = 40kHz<br>$A_{IN}$ = 95%FSR | 63.0 | 68.0 | 68.7 | dB |
| SINAD | Signal-to-Noise and Distortion | | 55.0 | 60.1 | 62.5 | dB |
| SNR | Signal-to-Noise Ratio | | 54.0 | 61.0 | 64.0 | dB |
| THD | Total Harmonic Distortion | | -69.0 | -68.0 | -65.0 | dB |
| | Noise RMS | T = 25°C | - | 1.0 | - | mV |

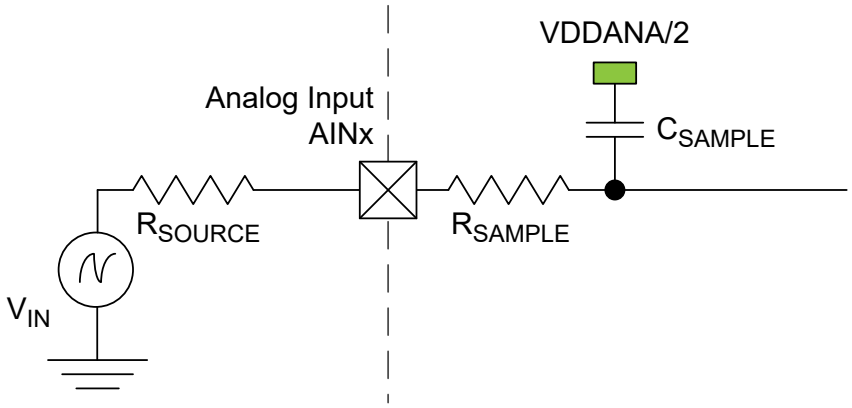
**Note:**

1. Maximum numbers are based on characterization and not tested in production, and for 5% to 95% of the input voltage range.
2. Respect the input common mode voltage through the following equations (where VCM_IN is the Input channel common mode voltage) for all VIN:
   - VCM_IN < 0.7*VDDANA + VREF/4 – 0.75V
   - VCM_IN > VREF/4 – 0.3*VDDANA - 0.1V
3. The ADC channels on pins PA08, PA09, PA10, PA11 are powered from the $V_{DDIO}$ power supply. The ADC performance of these pins will not be the same as all the other ADC channels on pins powered from the VDDANA power supply.
4. The gain accuracy represents the gain error expressed in percent. Gain accuracy (%) = (Gain Error in V x 100) / ($V_{ref}$/GAIN)

### 39.6.4 Inputs and Sample and Hold Acquisition Times

The analog voltage source must be able to charge the sample and hold (S/H) capacitor in the ADC in order to achieve maximum accuracy. Seen externally the ADC input consists of a resistor ($R_{\mathrm{SAMPLE}}$) and a capacitor ($C_{\mathrm{SAMPLE}}$). In addition, the source resistance ($R_{\mathrm{SOURCE}}$) must be taken into account when calculating the required sample and hold time. The next figure shows the ADC input channel equivalent circuit.

**Figure 39-3. ADC Input**




To achieve n bits of accuracy, the $C_{SAMPLE}$ capacitor must be charged at least to a voltage of

$V_{CSAMPLE} \geq V_{IN} \times \left(1 + -2^{-(n+1)}\right)$

The minimum sampling time $t_{SAMPLEHOLD}$ for a given $R_{SOURCE}$can be found using this formula:

$t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times (n+1) \times \ln(2)$

for a 12 bits accuracy: $t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times 9.02$

where

$t_{SAMPLEHOLD} = \frac{1}{2 \times f_{ADC}}$

### 39.6.5 Digital to Analog Converter (DAC) Characteristics

**Table 39-18. Operating Conditions(1)(Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DDANA}$ | Analog supply voltage | | 1.62 | - | 3.63 | V |
| $AV_{REF}$ | External reference voltage | | 1.0 | - | $V_{DDANA}$-0.6 | V |
| | Internal reference voltage 1 | | - | 1 | - | V |
| | Internal reference voltage 2 | | - | $V_{DDANA}$ | - | V |
| | Linear output voltage range | | 0.05 | - | $V_{DDANA}$-0.05 | V |
| | Minimum resistive load | | 5 | - | - | kΩ |
| | Maximum capacitance load | | - | - | 100 | pF |
| $I_{DD}$ | DC supply current(2) | Voltage pump disabled | - | 160 | 242 | µA |

**Table 39-19. Operating Conditions(1)(Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{DDANA}$ | Analog supply voltage | | 1.62 | - | 3.63 | V |
| $AV_{REF}$ | External reference voltage | | 1.0 | - | $V_{DDANA}$-0.6 | V |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Internal reference voltage 1 | | - | 1 | - | V |
| | Internal reference voltage 2 | | - | $V_{DDANA}$ | - | V |
| | Linear output voltage range | | 0.05 | - | $V_{DDANA}$-0.05 | V |
| | Minimum resistive load | | 5 | - | - | kΩ |
| | Maximum capacitance load | | - | - | 100 | pF |
| $I_{DD}$ | DC supply current(2) | Voltage pump disabled | - | 160 | 283 | µA |

1. These values are based on specifications otherwise noted.

2. These values are based on characterization. These values are not covered by test limits in production.

**Table 39-20. Clock and Timing(1)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| | Conversion rate | $C_{load}$=100pF $R_{load}$ > 5kΩ | Normal mode | - | - | 350 | ksps |
| | | | For $\Delta_{DATA}$=+/-1 | - | - | 1000 | |
| | Startup time | $V_{DDNA}$ > 2.6V | | - | - | 2.85 | µs |
| | | $V_{DDNA}$ < 2.6V | | - | - | 10 | µs |

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

**Table 39-21. Accuracy Characteristics(1)(Device Variant A)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| RES | Input resolution | | | - | - | 10 | Bits |
| INL | Integral non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | 0.75 | 1.1 | 2.0 | LSB |
| | | | $V_{DD}$ = 3.6V | 0.6 | 1.2 | 2.5 | |
| | | $V_{REF}$ = $V_{DDANA}$ | $V_{DD}$ = 1.6V | 1.4 | 2.2 | 3.5 | |
| | | | $V_{DD}$ = 3.6V | 0.9 | 1.4 | 1.5 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | 0.75 | 1.3 | 2.5 | |
| | | | $V_{DD}$ = 3.6V | 0.8 | 1.2 | 1.5 | |
| DNL | Differential non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | +/-0.9 | +/-1.2 | +/-2.0 | LSB |
| | | | $V_{DD}$ = 3.6V | +/-0.9 | +/-1.1 | +/-1.5 | |
| | | $V_{REF}$= $V_{DDANA}$ | $V_{DD}$ = 1.6V | +/-1.1 | +/-1.7 | +/-3.0 | |
| | | | $V_{DD}$ = 3.6V | +/-1.0 | +/-1.1 | +/-1.6 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | +/-1.1 | +/-1.4 | +/-2.5 | |
| | | | $V_{DD}$ = 3.6V | +/-1.0 | +/-1.5 | +/-1.8 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Gain error | Ext. $V_{REF}$ | +/-1.0 | +/-5 | +/-10 | mV |
| | Offset error | Ext. $V_{REF}$ | +/-2 | +/-3 | +/-8 | mV |

**Table 39-22. Accuracy Characteristics[1](Device Variant B, C and L)**

| Symbol | Parameter | Conditions | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| RES | Input resolution | | | - | - | 10 | Bits |
| INL | Integral non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | 0.7 | 0.75 | 2.0 | LSB |
| | | | $V_{DD}$ = 3.6V | 0.6 | 0.65 | 1.5 | |
| | | $V_{REF}$ = $V_{DDANA}$ | $V_{DD}$ = 1.6V | 0.6 | 0.85 | 2.0 | |
| | | | $V_{DD}$ = 3.6V | 0.5 | 0.8 | 1.5 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | 0.5 | 0.75 | 1.5 | |
| | | | $V_{DD}$ = 3.6V | 0.7 | 0.8 | 1.5 | |
| DNL | Differential non-linearity | $V_{REF}$= Ext 1.0V | $V_{DD}$ = 1.6V | +/-0.3 | +/-0.4 | +/-1.0 | LSB |
| | | | $V_{DD}$ = 3.6V | +/-0.25 | +/-0.4 | +/-0.75 | |
| | | $V_{REF}$= $V_{DDANA}$ | $V_{DD}$ = 1.6V | +/-0.4 | +/-0.55 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.2 | +/-0.3 | +/-0.75 | |
| | | $V_{REF}$= INT1V | $V_{DD}$ = 1.6V | +/-0.5 | +/-0.7 | +/-1.5 | |
| | | | $V_{DD}$ = 3.6V | +/-0.4 | +/-0.7 | +/-1.5 | |
| | Gain error | Ext. $V_{REF}$ | | +/-0.5 | +/-5 | +/-12 | mV |
| | Offset error | Ext. $V_{REF}$ | | +/-2 | +/-1.5 | +/-8 | mV |

1. All values measured using a conversion rate of 350ksps.

### 39.6.6 Analog Comparator Characteristics

**Table 39-23. Electrical and Timing (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Positive input voltage range | | 0 | - | $V_{DDANA}$ | V |
| | Negative input voltage range | | 0 | - | $V_{DDANA}$ | |
| | Offset | Hysteresis = 0, Fast mode | -15 | 0.0 | +15 | mV |
| | | Hysteresis = 0, Low power mode | -25 | 0.0 | +25 | mV |
| | Hysteresis | Hysteresis = 1, Fast mode | 20 | 50 | 83 | mV |
| | | Hysteresis = 1, Low power mode | 15 | 40 | 75 | mV |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Propagation delay | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Fast mode | - | 60 | 116 | ns |
| | | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Low power mode | - | 225 | 370 | ns |
| $t_{STARTUP}$ | Startup time | Enable to ready delay Fast mode | - | 1 | 2 | µs |
| | | Enable to ready delay Low power mode | - | 12 | 19 | µs |
| $V_{SCALE}$ | INL(3) | | - | 0.75 | +1.58 | LSB |
| | DNL(3) | | - | 0.25 | +0.95 | LSB |
| | Offset Error (1)(2) | | -0.200 | 0.260 | +1.035 | LSB |
| | Gain Error (1)(2) | | 0.55 | 1.2 | 2.0 | LSB |

**Table 39-24. Electrical and Timing (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Positive input voltage range | | 0 | - | $V_{DDANA}$ | V |
| | Negative input voltage range | | 0 | - | $V_{DDANA}$ | |
| | Offset | Hysteresis = 0, Fast mode | -15 | 0.0 | +15 | mV |
| | | Hysteresis = 0, Low power mode | -25 | 0.0 | +25 | mV |
| | Hysteresis | Hysteresis = 1, Fast mode | 20 | 50 | 90 | mV |
| | | Hysteresis = 1, Low power mode | 15 | 40 | 75 | mV |
| | Propagation delay | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Fast mode | - | 90 | 180 | ns |
| | | Changes for $V_{ACM}=V_{DDANA}/2$ 100mV overdrive, Low power mode | - | 282 | 520 | ns |
| $t_{STARTUP}$ | Startup time | Enable to ready delay Fast mode | - | 1 | 2.6 | µs |
| | | Enable to ready delay Low power mode | - | 14 | 22 | µs |
| $V_{SCALE}$ | INL(3) | | - | 0.75 | +1.58 | LSB |
| | DNL(3) | | - | 0.25 | +0.95 | LSB |
| | Offset Error (1)(2) | | -0.200 | 0.260 | +1.035 | LSB |
| | Gain Error (1)(2) | | 0.55 | 1.2 | 2.0 | LSB |

**Note:**

1. According to the standard equation $V(X)=V_{LSB}*(X+1)$; $V_{LSB}=V_{DDANA}/64$
2. Data computed with the Best Fit method
3. Data computed using histogram

### 39.6.7 Temperature Sensor Characteristics

**Table 39-25. Temperature Sensor Characteristics(1)(Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Temperature sensor output voltage | T= 25°C, $V_{DDANA}$ = 3.3V | - | 0.667 | - | V |
| | Temperature sensor slope | | 2.2 | 2.4 | 2.7 | mV/°C |
| | Variation over $V_{DDANA}$ voltage | $V_{DDANA}$=1.62V to 3.6V | -9 | 1 | 14 | mV/V |
| | Temperature Sensor accuracy | Using the method described in the 37.10.8.2 Software-based Refinement of the Actual Temperature | -13.0 | - | 13.0 | °C |

**Table 39-26. Temperature Sensor Characteristics(1)(Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | Temperature sensor output voltage | T= 25°C, $V_{DDANA}$ = 3.3V | - | 0.688 | - | V |
| | Temperature sensor slope | | 2.06 | 2.16 | 2.26 | mV/°C |
| | Variation over $V_{DDANA}$ voltage | $V_{DDANA}$=1.62V to 3.6V | -0.4 | 1.4 | 3 | mV/V |
| | Temperature Sensor accuracy | Using the method described in the 37.10.8.2 Software-based Refinement of the Actual Temperature | -13.0 | - | 13.0 | °C |

**Note:** 1. These values are based on characterization. These values are not covered by test limits in production.

## 39.7 NVM Characteristics

**Table 39-27. Maximum Operating Frequency**

| $V_{DD}$ range | NVM Wait States | Maximum Operating Frequency | Units |
|---|---|---|---|
| 1.62V to 2.7V | 0 | 14 | MHz |
| | 1 | 28 | |
| | 2 | 40 | |

| $V_{DD}$ range | NVM Wait States | Maximum Operating Frequency | Units |
|---|---|---|---|
| 2.7V to 3.63V | 0 | 24 | |
| | 1 | 40 | |

Note that on this flash technology, a max number of 8 consecutive write is allowed per row. Once this number is reached, a row erase is mandatory.

**Table 39-28. Flash Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{NVM25k}$ | Retention after up to 25k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{NVM2.5k}$ | Retention after up to 2.5k | Average ambient 55°C | 20 | 100 | - | Years |
| $Ret_{NVM100}$ | Retention after up to 100 | Average ambient 55°C | 25 | >100 | - | Years |
| $Cyc_{NVM}$ | Cycling Endurance(1) | -40°C < Ta < 125°C | 25k | 150k | - | Cycles |

Note: 1. An endurance cycle is a write and an erase operation.

**Table 39-29. EEPROM Emulation(1) Endurance and Data Retention**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $Ret_{EEPROM100k}$ | Retention after up to 100k | Average ambient 55°C | 10 | 50 | - | Years |
| $Ret_{EEPROM10k}$ | Retention after up to 10k | Average ambient 55°C | 20 | 100 | - | Years |
| $Cyc_{EEPROM}$ | Cycling Endurance(2) | -40°C < Ta < 125°C | 100k | 600k | - | Cycles |

Notes: 1. The EEPROM emulation is a software emulation described in the App note AT03265.

2. An endurance cycle is a write and an erase operation.

**Table 39-30. NVM Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{FPP}$ | Page programming time | - | - | - | 2.5 | ms |
| $t_{FRE}$ | Row erase time | - | - | - | 6 | ms |
| $t_{FCE}$ | DSU chip erase time (CHIP_ERASE) | - | - | - | 240 | ms |

## 39.8 Oscillators Characteristics

### 39.8.1 Crystal Oscillator (XOSC) Characteristics

#### 39.8.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

**Table 39-31. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN}$ | XIN clock frequency | | - | - | 32 | MHz |

#### 39.8.1.2 Crystal Oscillator Characteristics

The following table describes the characteristics for the oscillator when a crystal is connected between XIN and XOUT . The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal datasheet. The capacitance of the external capacitors ($C_{LEXT}$) can then be computed as follows:

$$C_{LEXT} = 2(C_L + \ - C_{STRAY} - C_{SHUNT})$$

where $C_{STRAY}$ is the capacitance of the pins and PCB, $C_{SHUNT}$ is the shunt capacitance of the crystal.

**Table 39-32. Crystal Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | 0.4 | - | 32 | MHz |
| ESR | Crystal Equivalent Series Resistance<br>Safety Factor = 3<br>The AGC doesn't have any noticeable impact on these measurements. | f = 0.455 MHz, $C_L$ = 100pF<br>XOSC.GAIN = 0 | - | - | 5.6K | Ω |
| | | f = 2MHz, $C_L$ = 20pF<br>XOSC.GAIN = 0 | - | - | 416 | |
| | | f = 4MHz, $C_L$ = 20pF<br>XOSC.GAIN = 1 | - | - | 243 | |
| | | f = 8 MHz, $C_L$ = 20pF<br>XOSC.GAIN = 2 | - | - | 138 | |
| | | f = 16 MHz, $C_L$ = 20pF<br>XOSC.GAIN = 3 | - | - | 66 | |
| | | f = 32MHz, $C_L$ = 18pF<br>XOSC.GAIN = 4 | - | - | 56 | |
| $C_{XIN}$ | Parasitic capacitor load | | - | 5.9 | - | pF |
| $C_{XOUT}$ | Parasitic capacitor load | | - | 3.2 | - | pF |
| | Current Consumption | f = 2MHz, $C_L$ = 20pF, AGC off | 27 | 65 | 90 | µA |
| | | f = 2MHz, $C_L$ = 20pF, AGC on | 14 | 52 | 79 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC off | 61 | 117 | 161 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC on | 23 | 74 | 110 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC off | 131 | 226 | 319 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC on | 56 | 128 | 193 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC off | 305 | 502 | 742 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC on | 116 | 307 | 627 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC off | 1031 | 1622 | 2344 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC on | 278 | 615 | 1422 | |
| $t_{STARTUP}$ | Start-up time | f = 2MHz, $C_L$ = 20pF,<br>XOSC.GAIN = 0, ESR = 600Ω | - | 14K | 48K | cycles |

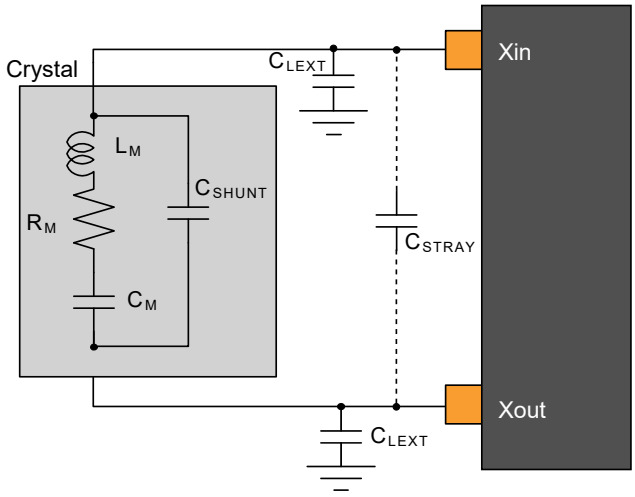
| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | f = 4MHz, $C_L$ = 20pF, XOSC.GAIN = 1, ESR = 100Ω | - | 6800 | 19.5K | |
| | | f = 8 MHz, $C_L$ = 20pF, XOSC.GAIN = 2, ESR = 35Ω | - | 5550 | 13K | |
| | | f = 16 MHz, $C_L$ = 20pF, XOSC.GAIN = 3, ESR = 25Ω | - | 6750 | 14.5K | |
| | | f = 32MHz, $C_L$ = 18pF, XOSC.GAIN = 4, ESR = 40Ω | - | 5.3K | 9.6K | |

**Table 39-33. Crystal Oscillator Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | 0.4 | - | 32 | MHz |
| ESR | Crystal Equivalent Series Resistance<br>Safety Factor = 3<br>The AGC doesn't have any noticeable impact on these measurements. | f = 0.455 MHz, $C_L$ = 100pF XOSC.GAIN = 0 | - | - | 5.6K | Ω |
| | | f = 2MHz, $C_L$ = 20pF XOSC.GAIN = 0 | - | - | 416 | |
| | | f = 4MHz, $C_L$ = 20pF XOSC.GAIN = 1 | - | - | 243 | |
| | | f = 8 MHz, $C_L$ = 20pF XOSC.GAIN = 2 | - | - | 138 | |
| | | f = 16 MHz, $C_L$ = 20pF XOSC.GAIN = 3 | - | - | 66 | |
| | | f = 32MHz, $C_L$ = 18pF XOSC.GAIN = 4 | - | - | 56 | |
| $C_{XIN}$ | Parasitic capacitor load | | - | 5.9 | - | pF |
| $C_{XOUT}$ | Parasitic capacitor load | | - | 3.2 | - | pF |
| | Current Consumption | f = 2MHz, $C_L$ = 20pF, AGC off | 27 | 65 | 90 | µA |
| | | f = 2MHz, $C_L$ = 20pF, AGC on | 14 | 52 | 79 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC off | 61 | 117 | 160 | |
| | | f = 4MHz, $C_L$ = 20pF, AGC on | 23 | 74 | 110 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC off | 131 | 226 | 319 | |
| | | f = 8MHz, $C_L$ = 20pF, AGC on | 56 | 128 | 193 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC off | 305 | 502 | 741 | |
| | | f = 16MHz, $C_L$ = 20pF, AGC on | 116 | 307 | 626 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC off | 1031 | 1622 | 2344 | |
| | | f = 32MHz, $C_L$ = 18pF, AGC on | 278 | 615 | 1400 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{STARTUP}$ | Start-up time | f = 2MHz, $C_L$ = 20pF, XOSC.GAIN = 0, ESR = 600Ω | - | 14K | 48K | cycles |
| | | f = 4MHz, $C_L$ = 20pF, XOSC.GAIN = 1, ESR = 100Ω | - | 6800 | 19.5K | |
| | | f = 8 MHz, $C_L$ = 20pF, XOSC.GAIN = 2, ESR = 35Ω | - | 5550 | 13K | |
| | | f = 16 MHz, $C_L$ = 20pF, XOSC.GAIN = 3, ESR = 25Ω | - | 6750 | 14.5K | |
| | | f = 32MHz, $C_L$ = 18pF, XOSC.GAIN = 4, ESR = 40Ω | - | 5.3K | 9.6K | |

**Figure 39-4. Oscillator Connection**




### 39.8.2 External 32 kHz Crystal Oscillator (XOSC32K) Characteristics

#### 39.8.2.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN32 pin.

**Table 39-34. Digital Clock Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{CPXIN32}$ | XIN32 clock frequency | | - | 32.768 | - | kHz |
| | XIN32 clock duty cycle | | - | 50 | - | % |

#### 39.8.2.2 Crystal Oscillator Characteristics

Figure 37-6 and the equation in also applies to the 32 kHz oscillator connection. The user must choose a crystal oscillator where the crystal load capacitance $C_L$ is within the range given in the table. The exact value of $C_L$ can be found in the crystal data sheet.

**Table 39-35. 32kHz Crystal Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | - | 32768 | - | Hz |
| $t_{STARTUP}$ | Startup time | $ESR_{XTAL}$ = 39.9 kΩ, $C_L$ = 12.5 pF | - | 28K | 31K | cycles |
| $C_L$ | Crystal load capacitance | | - | - | 12.5 | pF |
| $C_{SHUNT}$ | Crystal shunt capacitance | | - | 0.1 | - | |
| $C_{XIN32}$ | Parasitic capacitor load | TQFP64/48/32 packages | - | 3.1 | - | |
| $C_{XOUT32}$ | Parasitic capacitor load | | - | 3.3 | - | |
| $I_{XOSC32K}$ | Current consumption | | - | 1.22 | 2.44 | µA |
| ESR | Crystal equivalent series resistance f=32.768kHz , Safety Factor = 3 | $C_L$=12.5pF | - | - | 141 | kΩ |

**Table 39-36. 32kHz Crystal Oscillator Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Crystal oscillator frequency | | - | 32768 | - | Hz |
| $t_{STARTUP}$ | Startup time | $ESR_{XTAL}$ = 39.9 kΩ, $C_L$ = 12.5 pF | - | 28K | 30K | cycles |
| $C_L$ | Crystal load capacitance | | - | - | 12.5 | pF |
| $C_{SHUNT}$ | Crystal shunt capacitance | | - | 0.1 | - | |
| $C_{XIN32}$ | Parasitic capacitor load | TQFP64/48/32 packages | - | 3.2 | - | |
| $C_{XOUT32}$ | Parasitic capacitor load | | - | 3.7 | - | |
| $I_{XOSC32K}$ | Current consumption | | - | 1.22 | 2.2 | µA |
| ESR | Crystal equivalent series resistance f=32.768kHz , Safety Factor = 3 | $C_L$=12.5pF | - | - | 100 | kΩ |

### 39.8.3 Digital Frequency Locked Loop (DFLL48M) Characteristics

**Table 39-37. DFLL48M Characteristics - Open Loop Mode(1)(Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | IDFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | 47 | 48 | 49 | MHz |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | IDFLLVAL.COARSE = DFLL48M COARSE CAL | - | 403 | 457 | µA |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | DFLLVAL.FINE = 512 | | | | |
| $t_{STARTUP}$ | Start-up time | DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>$f_{OUT}$ within 90 % of final value | 7 | 8 | 9 | µs |

**Table 39-38. DFLL48M Characteristics - Open Loop Mode(1)(Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | IDFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | 47 | 48 | 49 | MHz |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | IDFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512 | - | 403 | 453 | µA |
| $t_{STARTUP}$ | Start-up time | DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>$f_{OUT}$ within 90 % of final value | - | 8 | 9 | µs |

Note: 1. DFLL48M in Open loop after calibration at room temperature.

**Table 39-39. DFLL48M Characteristics - Closed Loop Mode(1)(Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Average Output frequency | $f_{REF}$ = 32 .768kHz | 47.76 | 48 | 48.24 | MHz |
| $f_{REF}$ | Reference frequency | | 0.732 | 32.768 | 33 | kHz |
| Jitter | Cycle to Cycle jitter | $f_{REF}$ = 32 .768kHz | - | - | 1.04 | ns |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | $f_{REF}$ = 32 .768kHz | - | 425 | 482 | µA |
| $t_{LOCK}$ | Lock time | $f_{REF}$ = 32 .768kHz<br>DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>DFLLCTRL.BPLCKC = 1<br>DFLLCTRL.QLDIS = 0<br>DFLLCTRL.CCDIS = 1<br>DFLLMUL.FSTEP = 10 | 100 | 200 | 500 | µs |

**Table 39-40. DFLL48M Characteristics - Closed Loop Mode(1)(Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Average Output frequency | $f_{REF}$ = 32 .768kHz | 47.76 | 48 | 48.24 | MHz |
| $f_{REF}$ | Reference frequency | | 0.732 | 32.768 | 33 | kHz |
| Jitter | Cycle to Cycle jitter | $f_{REF}$ = 32 .768kHz | - | - | 0.42 | ns |
| $I_{DFLL}$ | Power consumption on $V_{DDIN}$ | $f_{REF}$ = 32 .768kHz | - | 403 | 453 | µA |
| $t_{LOCK}$ | Lock time | $f_{REF}$ = 32 .768kHz<br>DFLLVAL.COARSE = DFLL48M COARSE CAL<br>DFLLVAL.FINE = 512<br>DFLLCTRL.BPLCKC = 1<br>DFLLCTRL.QLDIS = 0<br>DFLLCTRL.CCDIS = 1<br>DFLLMUL.FSTEP = 10 | - | 200 | 500 | µs |

### 39.8.4 32.768kHz Internal oscillator (OSC32K) Characteristics

**Table 39-41. 32kHz RC Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 28.508 | 32.768 | 35.389 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 32.276 | 32.768 | 33.260 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.457 | 32.768 | 34.079 | |
| $I_{OSC32K}$ | Current consumption | | - | 0.79 | 1.80 | µA |
| $t_{STARTUP}$ | Start-up time | | - | 1 | 2 | cycle |
| Duty | Duty Cycle | | - | 50 | - | % |

**Table 39-42. 32kHz RC Oscillator Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 28.508 | 32.768 | 35.389 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 32.276 | 32.768 | 33.260 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.457 | 32.768 | 34.079 | |
| $I_{OSC32K}$ | Current consumption | | - | 0.67 | 2.80 | µA |
| $t_{STARTUP}$ | Start-up time | | - | 1 | 2 | cycle |
| Duty | Duty Cycle | | - | 50 | - | % |

### 39.8.5 Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics

**Table 39-43. Ultra Low Power Internal 32kHz RC Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 25.559 | 32.768 | 40.305 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 31.293 | 32.768 | 34.570 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.293 | 32.768 | 34.570 | |
| $i_{OSCULP32K}$(1)(2) | | | - | - | 180 | nA |
| $t_{STARTUP}$ | Start-up time | | - | 10 | - | cycles |
| Duty | Duty Cycle | | - | 50 | - | % |

**Table 39-44. Ultra Low Power Internal 32kHz RC Oscillator Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V | 25.559 | 32.768 | 40.305 | kHz |
| | | Calibrated against a 32.768kHz reference at 25°C, at $V_{DD}$=3.3V | 31.293 | 32.768 | 34.570 | |
| | | Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V | 31.293 | 32.768 | 34.570 | |
| Duty | Duty Cycle | | - | 50 | - | % |

**Note:**

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. This oscillator is always on.

### 39.8.6 8MHz RC Oscillator (OSC8M) Characteristics

**Table 39-45. Internal 8MHz RC Oscillator Characteristics (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 8MHz reference at 25°C, over [-40, +85]°C, over [1.62, 3.63]V | 7.54 | 8 | 8.19 | MHz |
| | | Calibrated against a 8MHz reference at 25°C, at $V_{DD}$=3.3V | 7.94 | 8 | 8.06 | |
| | | Calibrated against a 8MHz reference at 25°C, over [1.62, 3.63]V | 7.92 | 8 | 8.08 | |
| $I_{OSC8M}$ | Current consumption | IIDLEIDLE2 on OSC32K versus IDLE2 on calibrated OSC8M enabled at 8MHz (FRANGE=1, PRESC=0) | | 64 | 100 | µA |
| $t_{STARTUP}$ | Startup time | | - | 2.1 | 3 | µs |
| Duty | Duty cycle | | - | 50 | - | % |

**Table 39-46. Internal 8MHz RC Oscillator Characteristics (Device Variant B, C and L)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{OUT}$ | Output frequency | Calibrated against a 8MHz reference at 25°C, over [-40, +85]°C, over [1.62, 3.63]V | 7.54 | 8 | 8.19 | MHz |
| | | Calibrated against a 8MHz reference at 25°C, at $V_{DD}$=3.3V | 7.94 | 8 | 8.06 | |
| | | Calibrated against a 8MHz reference at 25°C, over [1.62, 3.63]V | 7.92 | 8 | 8.06 | |
| $I_{OSC8M}$ | Current consumption | IIDLEIDLE2 on OSC32K versus IDLE2 on calibrated OSC8M enabled at 8MHz (FRANGE=1, PRESC=0) | | 64 | 96 | µA |
| $t_{STARTUP}$ | Startup time | | - | 2.4 | 3.3 | µs |
| Duty | Duty cycle | | - | 50 | - | % |

### 39.8.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics

**Table 39-47. FDPLL96M Characteristics(1) (Device Variant A)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | 700 | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | 1200 | |
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 1.5 | 2.0 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 3.0 | 10.0 | |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 1.3 | 2.0 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 3.0 | 7.0 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.3 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Table 39-48. FDPLL96M Characteristics(1) (Device Variant B and L with Silicon Revision E)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | 740 | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | 1262 | |
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 1.5 | 2.5 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 4.0 | 10.5 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 1.6 | 2.5 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 4.6 | 11.0 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.2 | 2 | ms |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Table 39-49. FDPLL96M Characteristics(1) (Device Variant B, C and L with Silicon Revision F)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input frequency | | 32 | - | 2000 | KHz |
| $f_{OUT}$ | Output frequency | | 48 | - | 96 | MHz |
| $I_{FDPLL96M}$ | Current consumption | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 500 | - | µA |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 900 | - | |
| $J_p$ | Period jitter | $f_{IN}$= 32 kHz, $f_{OUT}$= 48 MHz | - | 2.1 | 3.2 | % |
| | | $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 3.8 | 9.2 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 48 MHz | - | 2.2 | 3.4 | |
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 5.0 | 10.5 | |
| $t_{LOCK}$ | Lock Time | After start-up, time to get lock signal. $f_{IN}$= 32 kHz, $f_{OUT}$= 96 MHz | - | 1.2 | 2 | ms |

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | | $f_{IN}$= 2 MHz, $f_{OUT}$= 96 MHz | - | 25 | 50 | µs |
| Duty | Duty cycle | | 40 | 50 | 60 | % |

**Note:**

1. All values have been characterized with FILTSEL[1/0] as default value.

### 39.8.8 USB Characteristics

The USB shares the same characteristics as in the -40°C to 85°C.

## 39.9 Timing Characteristics

### 39.9.1 SERCOM in SPI Mode Timing

Data are not available in this current datasheet revision

### 39.9.2 SERCOM in I²C Mode Timing

Data are not available in this current datasheet revision

### 39.9.3 I2S Timing

**Figure 39-5. I2S Timing Master Mode**

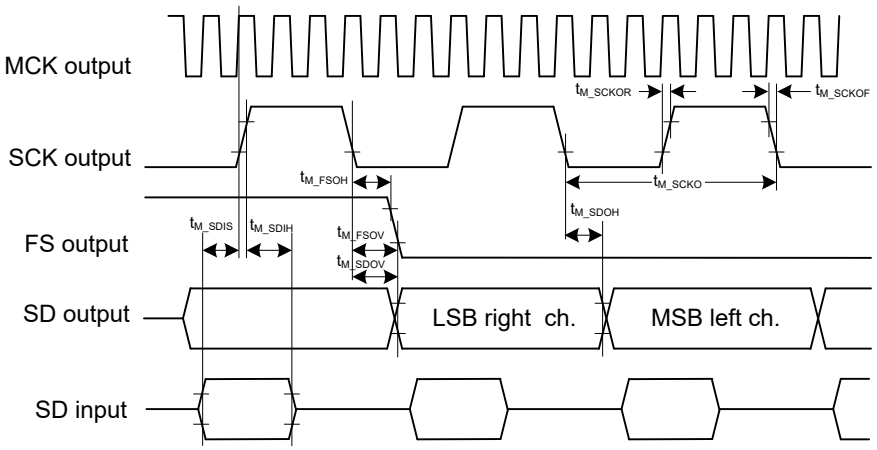



**Figure 39-6. I2S Timing Slave Mode**

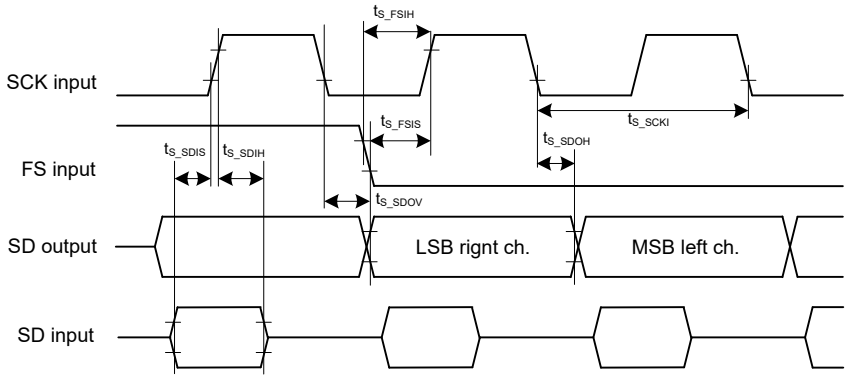


**Figure 39-7. I2S Timing PDM2 Mode**

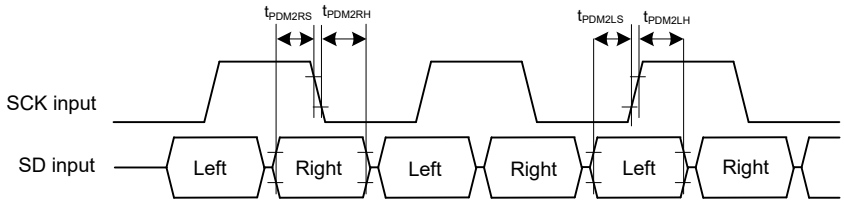



**Table 39-50. I2S Timing Characteristics and Requirements (Device Variant A)**

| Symbol | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | Min. | Typ. | Max | |
| $t_{M_MCKOR}$ | I2S MCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 10 | | | 4.7 | ns |
| $t_{M_MCKOF}$ | I2S MCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 12.1 | | | 5.3 | ns |
| $d_{M_MCKO}$ | I2S MCK duty cycle | Master mode | 45.4 | | 50 | 45.4 | | 50 | % |
| $d_{M_MCKI}$ | I2S MCK duty cycle | Master mode, pin is input (1b) | | 50 | | | 50 | | % |
| $t_{M_SCKOR}$ | I2S SCK rise time(2) | Master mode / Capacitive load CL = 15 pF | | | 9.7 | | | 4.6 | ns |
| $t_{M_SCKOF}$ | I2S SCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 10.2 | | | 4.5 | ns |
| $d_{M_SCKO}$ | I2S SCK duty cycle | Master mode | 45.6 | | 50 | 45.6 | | 50 | % |
| $f_{M_SCKO}$, 1/ $t_{M_SCKO}$ | I2S SCK frequency | Master mode, Supposing external device response delay is 30ns | | | 7.8 | | | 9.5 | MHz |
| $f_{S_SCKI}$, 1/ $t_{S_SCKI}$ | I2S SCK frequency | Slave mode, Supposing external device response delay is 30ns | | | 14.4 | | | 14.8 | MHz |
| $d_{S_SCKO}$ | I2S SCK duty cycle | Slave mode | | 50 | | | 50 | | % |
| $t_{M_FSOV}$ | FS valid time | Master mode | | | 4.1 | | | 4 | ns |
| $t_{M_FSOH}$ | FS hold time | Master mode | -0.9 | | | -0.9 | | | ns |
| $t_{S_FSIS}$ | FS setup time | Slave mode | 2.3 | | | 1.5 | | | ns |
| $t_{S_FSIH}$ | FS hold time | Slave mode | 0 | | | 0 | | | ns |

| Symbol | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | Min. | Typ. | Max | |
| $t_{M_SDIS}$ | Data input setup time | Master mode | 36.2 | | | 24.5 | | | ns |
| $t_{M_SDIH}$ | Data input hold time | Master mode | -8.2 | | | -8.2 | | | ns |
| $t_{S_SDIS}$ | Data input setup time | Slave mode | 4.8 | | | 3.9 | | | ns |
| $t_{S_SDIH}$ | Data input hold time | Slave mode | 1.2 | | | 1.2 | | | ns |
| $t_{M_SDOV}$ | Data output valid time | Master transmitter | | | 5.9 | | | 4.8 | ns |
| $t_{M_SDOH}$ | Data output hold time | Master transmitter | -0.5 | | | -0.5 | | | ns |
| $t_{S_SDOV}$ | Data output valid time | Slave transmitter | | | 37.8 | | | 25.9 | ns |
| $t_{S_SDOH}$ | Data output hold time | Slave transmitter | 37.6 | | | 25.7 | | | ns |
| $t_{PDM2LS}$ | Data input setup time | Master mode PDM2 Left | 36.2 | | | 24.5 | | | ns |
| $t_{PDM2LH}$ | Data input hold time | Master mode PDM2 Left | -8.2 | | | -8.2 | | | ns |
| $t_{PDM2RS}$ | Data input setup time | Master mode PDM2 Right | 31.9 | | | 20.9 | | | ns |
| $t_{PDM2RH}$ | Data input hold time | Master mode PDM2 Right | -6.7 | | | -6.7 | | | ns |

1. All timing characteristics given for 15pF capacitive load.
2. These values are based on simulations and not covered by test limits in production.
3. See 37.8 I/O Pin Characteristics.

**Table 39-51. I2S Timing Characteristics and Requirements (Device Variant B, C and L)**

| Name | Description | Mode | VDD=1.8V | | | VDD=3.3V | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | Min. | Typ. | Max. | |
| $t_{M_MCKOR}$ | I2S MCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.9 | | | 4.7 | ns |
| $t_{M_MCKOF}$ | I2S MCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 12.3 | | | 5.4 | ns |
| $d_{M_MCKO}$ | I2S MCK duty cycle | Master mode | 46.9 | | 50 | 47.3 | | 50 | % |
| $d_{M_MCKI}$ | I2S MCK duty cycle | Master mode, pin is input (1b) | | 50 | | | 50 | | % |
| $t_{M_SCKOR}$ | I2S SCK rise time(3) | Master mode / Capacitive load CL = 15 pF | | | 9.7 | | | 4.6 | ns |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_{M_SCKOF}$ | I2S SCK fall time(3) | Master mode / Capacitive load CL = 15 pF | | | 10.3 | | | 4.6 | ns |
| $d_{M_SCKO}$ | I2S SCK duty cycle | Master mode | 46.9 | | 50 | 47.2 | | 50 | % |
| $f_{M_SCKO}$, 1/ $t_{M_SCKO}$ | I2S SCK frequency | Master mode, Supposing external device response delay is 30ns | | | 7.7 | | | 9.2 | MHz |
| $f_{S_SCKI}$, 1/ $t_{S_SCKI}$ | I2S SCK frequency | Slave mode, Supposing external device response delay is 30ns | | | 12.7 | | | 13 | MHz |
| $d_{S_SCKO}$ | I2S SCK duty cycle | Slave mode | | 50 | | | 50 | | % |
| $t_{M_FSOV}$ | FS valid time | Master mode | | | 2.6 | | | 1.9 | ns |
| $t_{M_FSOH}$ | FS hold time | Master mode | -0.1 | | | -0.1 | | | ns |
| $t_{S_FSIS}$ | FS setup time | Slave mode | 6.2 | | | 5.3 | | | ns |
| $t_{S_FSIH}$ | FS hold time | Slave mode | 0 | | | 0 | | | ns |
| $t_{M_SDIS}$ | Data input setup time | Master mode | 37.4 | | | 25.9 | | | ns |
| $t_{M_SDIH}$ | Data input hold time | Master mode | -8.2 | | | -8.2 | | | ns |
| $t_{S_SDIS}$ | Data input setup time | Slave mode | 9.2 | | | 8.3 | | | ns |
| $t_{S_SDIH}$ | Data input hold time | Slave mode | 3.8 | | | 3.7 | | | ns |
| $t_{M_SDOV}$ | Data output valid time | Master transmitter | | | 2.7 | | | 1.9 | ns |
| $t_{M_SDOH}$ | Data output hold time | Master transmitter | -0.1 | | | -0.1 | | | ns |
| $t_{S_SDOV}$ | Data output valid time | Slave transmitter | | | 31.2 | | | 19.7 | ns |
| $t_{S_SDOH}$ | Data output hold time | Slave transmitter | 30.6 | | | 18.9 | | | ns |
| $t_{PDM2LS}$ | Data input setup time | Master mode PDM2 Left | 36.9 | | | 25.3 | | | ns |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_{PDM2LH}$ | Data input hold time | Master mode PDM2 Left | -8.2 | | | -8.2 | | | ns |
| $t_{PDM2RS}$ | Data input setup time | Master mode PDM2 Right | 31.9 | | | 21.1 | | | ns |
| $t_{PDM2RH}$ | Data input hold time | Master mode PDM2 Right | -7 | | | -7 | | | ns |

1. All timing characteristics given for 15pF capacitive load.
2. These values are based on simulations and not covered by test limits in production.
3. See 37.8 I/O Pin Characteristics

# 40. Packaging Information

## 40.1 Thermal Considerations

**Related Links**

40.1.2 Junction Temperature

### 40.1.1 Thermal Resistance Data

The following Table summarizes the thermal resistance data depending on the package.

**Table 40-1. Thermal Resistance Data**

| Package Type | $\theta_{JA}$ | $\theta_{JC}$ |
|---|---|---|
| 32-pin TQFP | 64.7°C/W | 23.1°C/W |
| 48-pin TQFP | 63.6°C/W | 12.2°C/W |
| 64-pin TQFP | 60.9°C/W | 12.2°C/W |
| 32-pin QFN | 40.9°C/W | 15.2°C/W |
| 48-pin QFN | 32.0°C/W | 10.9°C/W |
| 64-pin QFN | 32.5°C/W | 10.7°C/W |
| 35-ball WLCSP | 41.8°C/W | 2.26°C/W |
| 45-ball WLCSP | 43.97°C/W | 2.91°C/W |
| 64-pin UFBGA | 53°C/W | 4.74°C/W |

### 40.1.2 Junction Temperature

The average chip-junction temperature, $T_J$, in °C can be obtained from the following:

1. $T_J = T_A + (P_D \times \theta_{JA})$
2. $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

where:

- $\theta_{JA}$ = Package thermal resistance, Junction-to-ambient (°C/W), see Thermal Resistance Data
- $\theta_{JC}$ = Package thermal resistance, Junction-to-case thermal resistance (°C/W), see Thermal Resistance Data
- $\theta_{HEATSINK}$ = Thermal resistance (°C/W) specification of the external cooling device
- $P_D$ = Device power consumption (W)
- $T_A$ = Ambient temperature (°C)

From the first equation, the user can derive the estimated lifetime of the chip and decide if a cooling device is necessary or not. If a cooling device has to be fitted on the chip, the second equation should be used to compute the resulting average chip-junction temperature $T_J$ in °C.

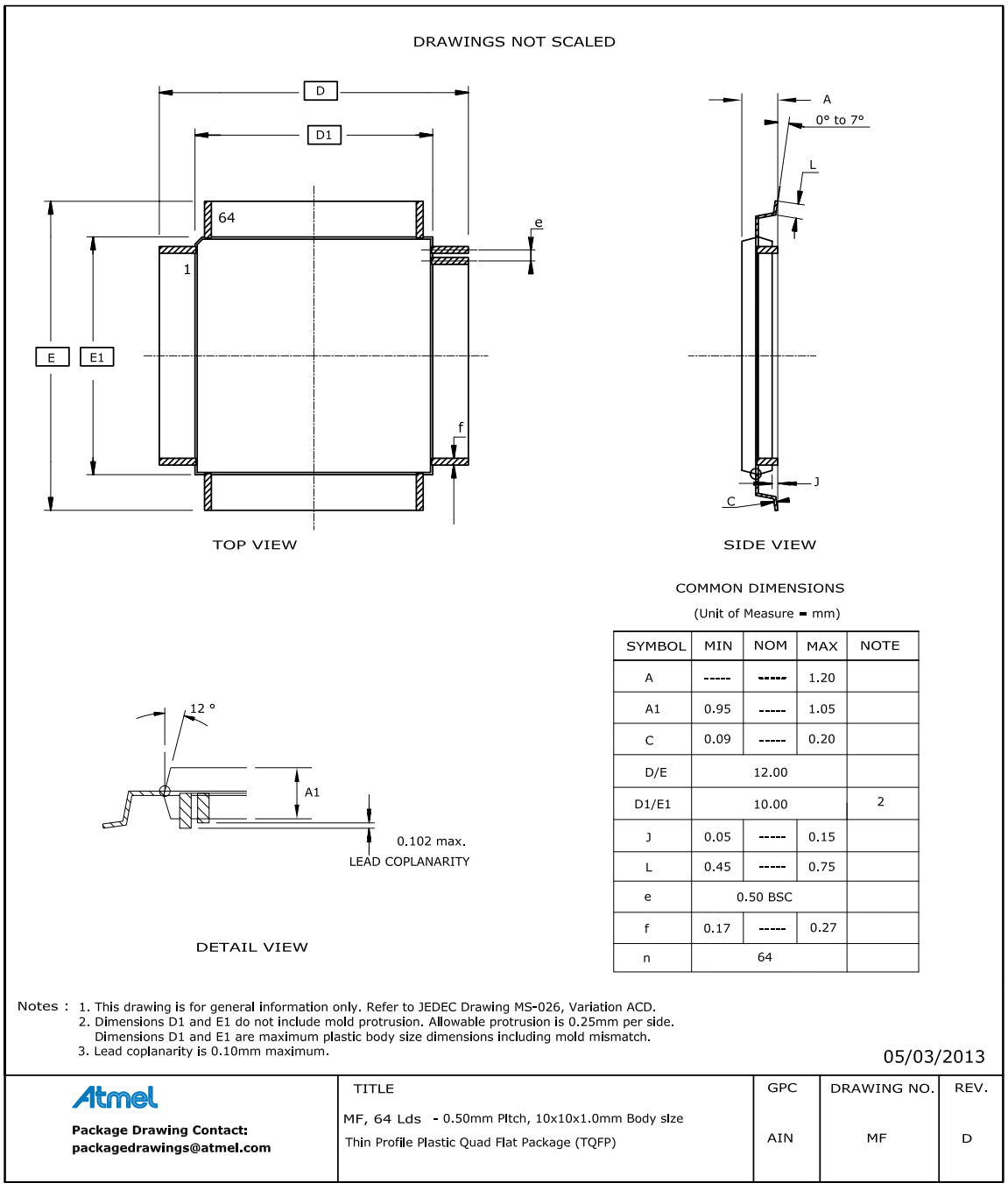
**Related Links**

40.1 Thermal Considerations

## 40.2 Package Drawings

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging.

### 40.2.1 64 pin TQFP




COMMON DIMENSIONS

(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | ----- | ----- | 1.20 | |
| A1 | 0.95 | ----- | 1.05 | |
| C | 0.09 | ----- | 0.20 | |
| D/E | | 12.00 | | |
| D1/E1 | | 10.00 | | 2 |
| J | 0.05 | ----- | 0.15 | |
| L | 0.45 | ----- | 0.75 | |
| e | | 0.50 BSC | | |
| f | 0.17 | ----- | 0.27 | |
| n | | 64 | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ACD.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

05/03/2013

| Atmel<br>Package Drawing Contact:<br>packagedrawings@atmel.com | TITLE<br>MF, 64 Lds - 0.50mm Pitch, 10x10x1.0mm Body size<br>Thin Profile Plastic Quad Flat Package (TQFP) | GPC<br>AIN | DRAWING NO.<br>MF | REV.<br>D |
|---|---|---|---|---|

**Table 40-2. Device and Package Maximum Weight**

| | |
|---|---|
| 300 | mg |

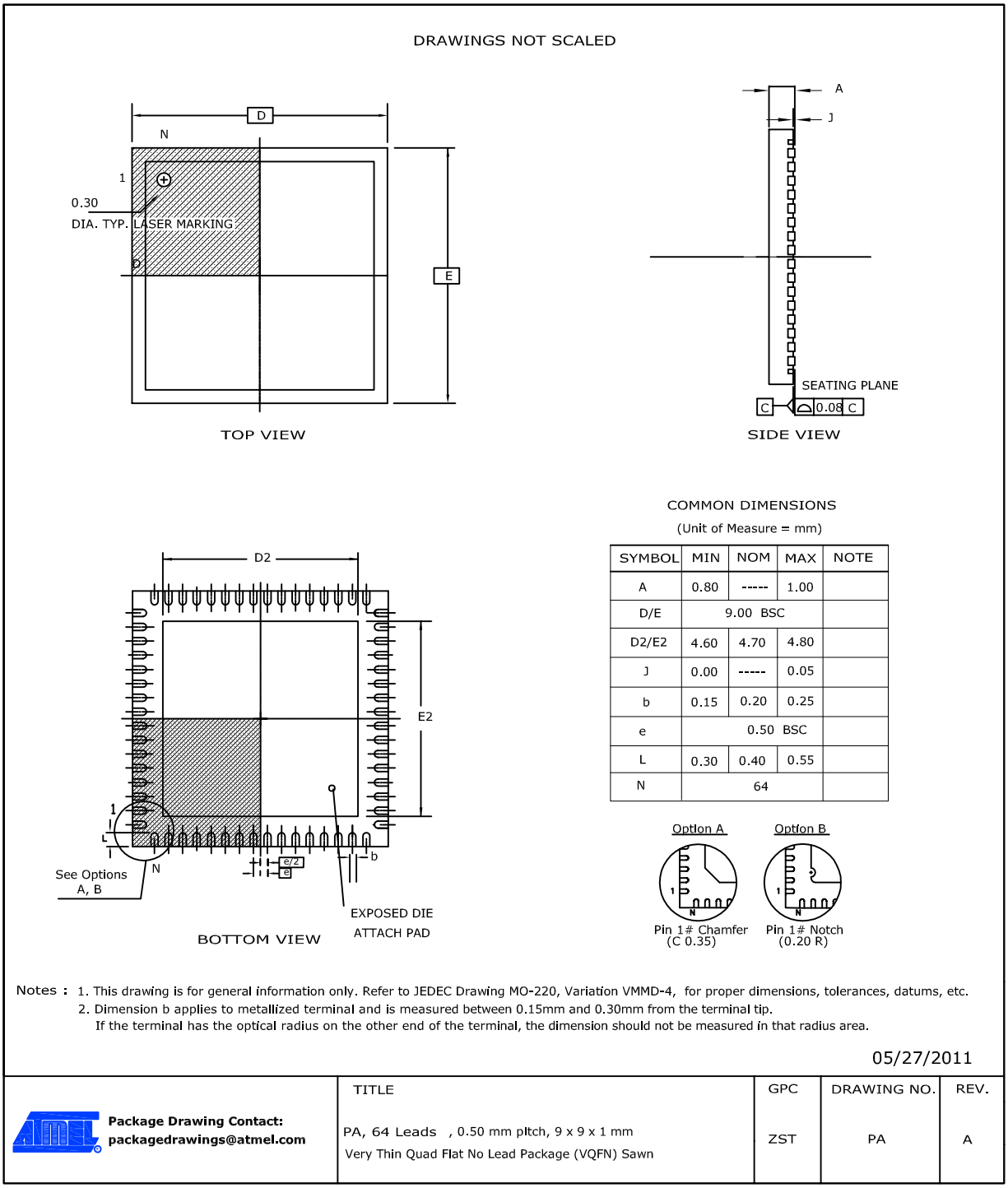
**Table 40-3. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

**Table 40-4. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MS-026 |
| JESD97 Classification | E3 |

### 40.2.2 64 pin QFN




COMMON DIMENSIONS

(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | 0.80 | ----- | 1.00 | |
| D/E | | 9.00 BSC | | |
| D2/E2 | 4.60 | 4.70 | 4.80 | |
| J | 0.00 | ----- | 0.05 | |
| b | 0.15 | 0.20 | 0.25 | |
| e | | 0.50 BSC | | |
| L | 0.30 | 0.40 | 0.55 | |
| N | | 64 | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-220, Variation VMMD-4, for proper dimensions, tolerances, datums, etc.
2. Dimension b applies to metallized terminal and is measured between 0.15mm and 0.30mm from the terminal tip.
If the terminal has the optical radius on the other end of the terminal, the dimension should not be measured in that radius area.

05/27/2011

| | TITLE | GPC | DRAWING NO. | REV. |
|---|---|---|---|---|
| Package Drawing Contact: packagedrawings@atmel.com | PA, 64 Leads , 0.50 mm pitch, 9 x 9 x 1 mm<br>Very Thin Quad Flat No Lead Package (VQFN) Sawn | ZST | PA | A |

**Note:** The exposed die attach pad is not connected electrically inside the device.

**Table 40-5. Device and Package Maximum Weight**

| | |
|---|---|
| 200 | mg |

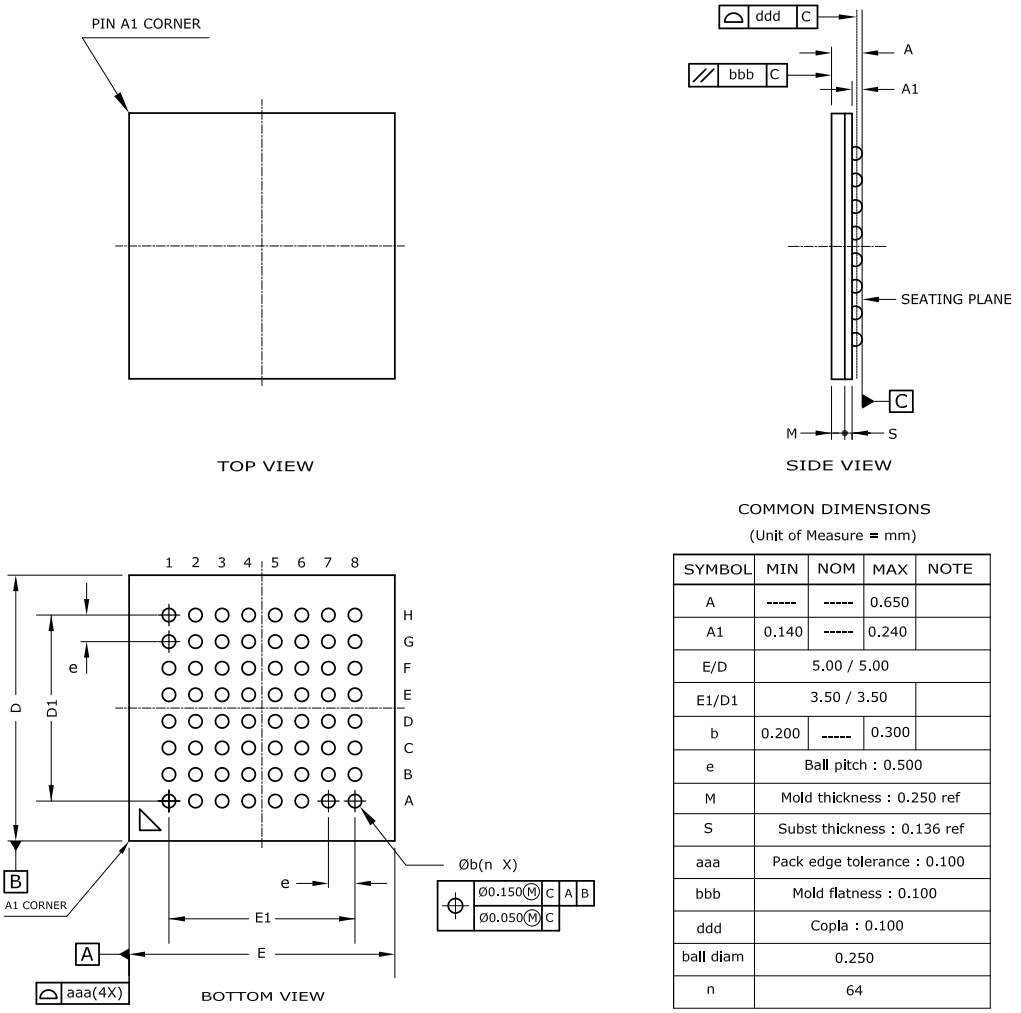
**Table 40-6. Package Charateristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

**Table 40-7. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E3 |

### 40.2.3 64-ball UFBGA




COMMON DIMENSIONS
(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | ----- | ----- | 0.650 | |
| A1 | 0.140 | ----- | 0.240 | |
| E/D | 5.00 / 5.00 | | | |
| E1/D1 | 3.50 / 3.50 | | | |
| b | 0.200 | ----- | 0.300 | |
| e | Ball pitch : 0.500 | | | |
| M | Mold thickness : 0.250 ref | | | |
| S | Subst thickness : 0.136 ref | | | |
| aaa | Pack edge tolerance : 0.100 | | | |
| bbb | Mold flatness : 0.100 | | | |
| ddd | Copla : 0.100 | | | |
| ball diam | 0.250 | | | |
| n | 64 | | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-280, Variation UCCBB for proper dimensions, tolerances, datums, etc.
2. Array as seen from the bottom of the package.
3. Dimension A includes stand-off height A1, package body thickness, and lid height, but does not include attached features.
4. Dimension b is measured at the maximum ball diameter, parallel to primary datum C.

**Table 40-8. Device and Package Maximum Weight**

| | |
|---|---|
| 27.4 | mg |

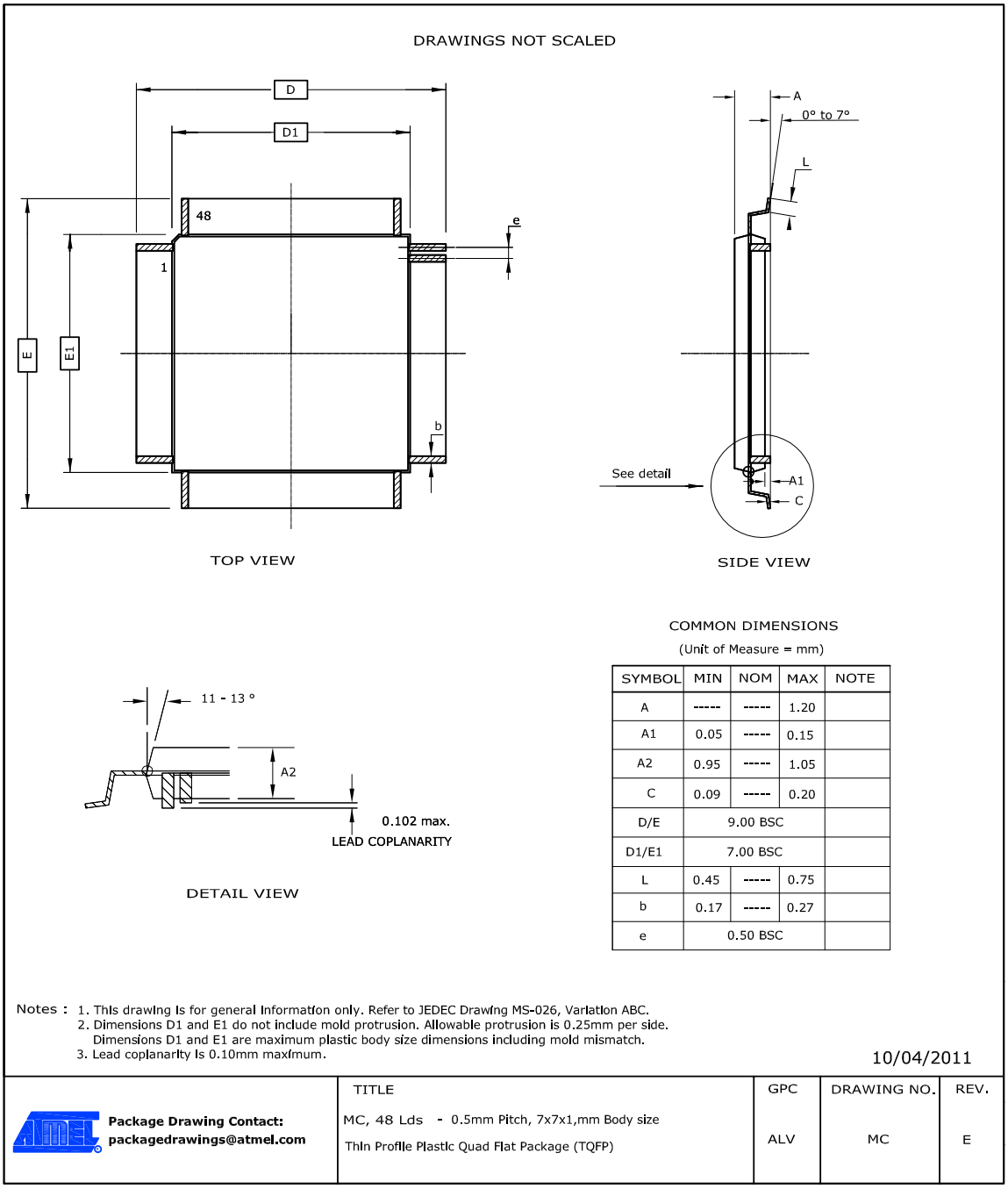
**Table 40-9. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

**Table 40-10. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E8 |

### 40.2.4 48 pin TQFP




COMMON DIMENSIONS
(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | ----- | ----- | 1.20 | |
| A1 | 0.05 | ----- | 0.15 | |
| A2 | 0.95 | ----- | 1.05 | |
| C | 0.09 | ----- | 0.20 | |
| D/E | | 9.00 BSC | | |
| D1/E1 | | 7.00 BSC | | |
| L | 0.45 | ----- | 0.75 | |
| b | 0.17 | ----- | 0.27 | |
| e | | 0.50 BSC | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ABC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

10/04/2011

| | TITLE | GPC | DRAWING NO. | REV. |
|---|---|---|---|---|
| Package Drawing Contact: packagedrawings@atmel.com | MC, 48 Lds - 0.5mm Pitch, 7x7x1,mm Body size<br>Thin Profile Plastic Quad Flat Package (TQFP) | ALV | MC | E |

**Table 40-11. Device and Package Maximum Weight**

| | |
|---|---|
| 140 | mg |

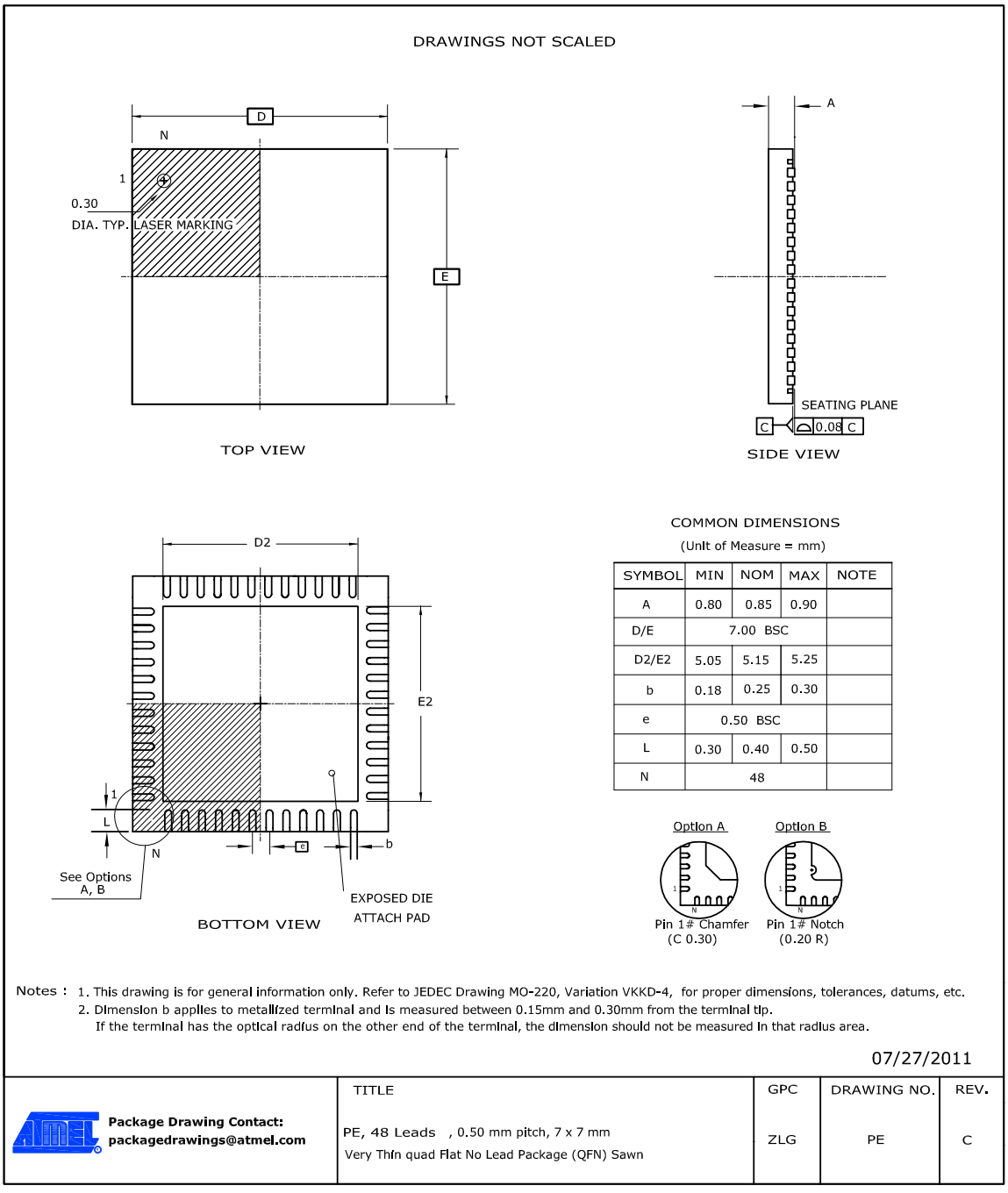
**Table 40-12. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

**Table 40-13. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MS-026 |
| JESD97 Classification | E3 |

### 40.2.5 48 pin QFN



**Note:** The exposed die attach pad is not connected electrically inside the device.

**Table 40-14. Device and Package Maximum Weight**

| | |
|---|---|
| 140 | mg |

**Table 40-15. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

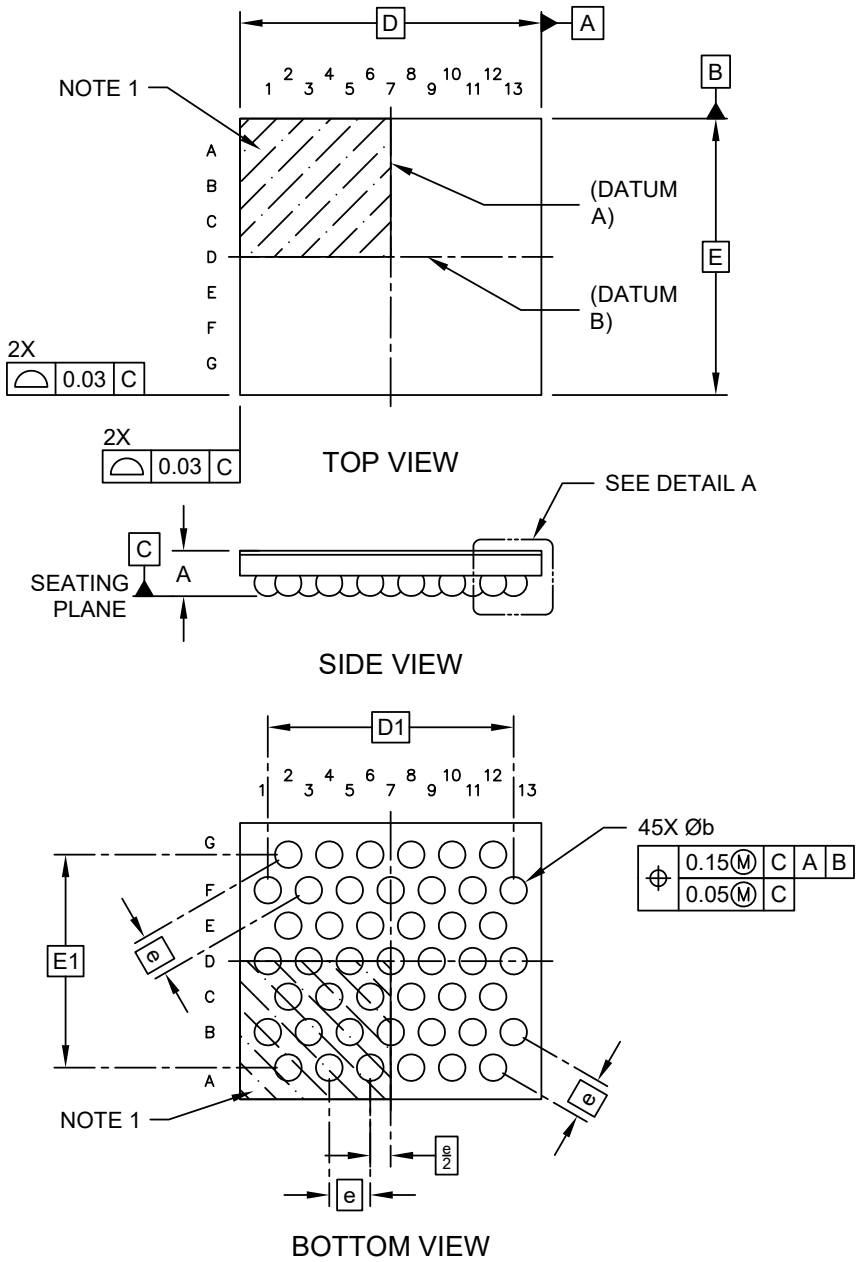
**Table 40-16. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E3 |

### 40.2.6 45-ball WLCSP

**45-Ball Wafer Level Chip Scale Package (FSB) - 2.944x2.699 mm Body [WLCSP]**

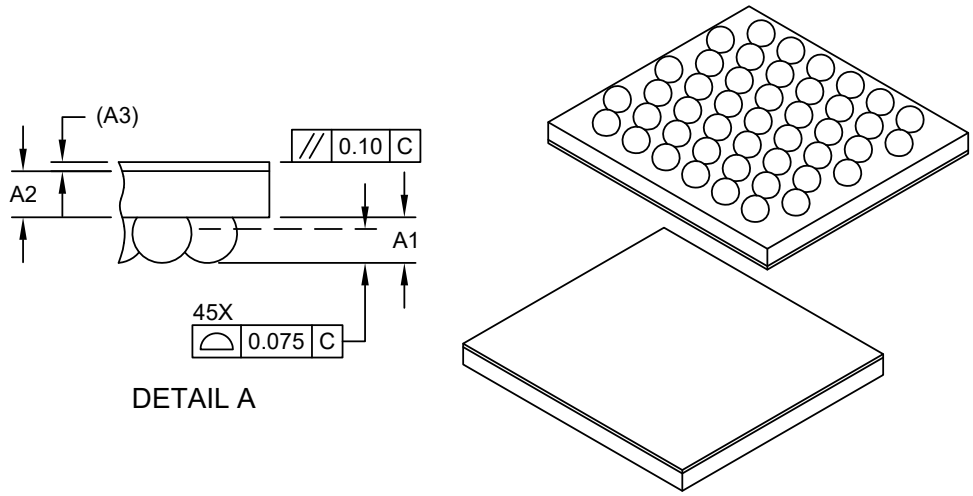
| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|




Microchip Technology Drawing C04-21247 Rev A Sheet 1 of 2

### 45-Ball Wafer Level Chip Scale Package (FSB) - 2.944x2.699 mm Body [WLCSP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 45 | |
| Pitch | e | | 0.40 BSC | |
| Overall Height | A | - | - | 0.483 |
| Bump Height | A1 | 0.17 | 0.20 | 0.23 |
| Die Thickness | A2 | 0.178 | 0.203 | 0.228 |
| Backside Coating | A3 | | 0.04 REF | |
| Overall Length | D | | 2.944 BSC | |
| Overall Bump Pitch | D1 | | 2.40 | |
| Overall Width | E | | 2.699 BSC | |
| Overall Bump Pitch | E1 | | 2.079 BSC | |
| Terminal Width | b | 0.23 | 0.26 | 0.29 |

Notes:

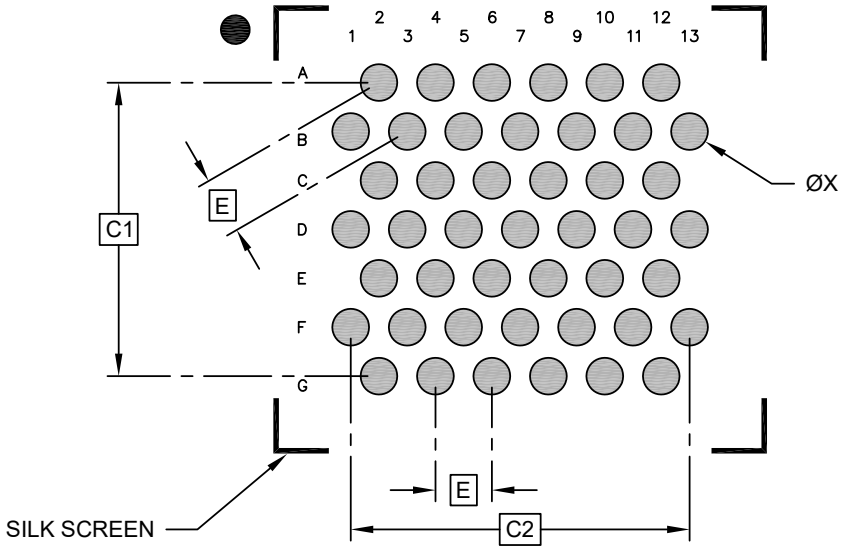
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21247 Rev A Sheet 2 of 2

### 45-Ball Wafer Level Chip Scale Package (FSB) - 2.944x2.699 mm Body [WLCSP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.40 BSC | |
| Contact Pad Spacing | C1 | | 2.079 BSC | |
| Contact Pad Spacing | C2 | | 2.40 BSC | |
| Contact Pad Diameter (X45) | X | | 0.26 | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-23247 Rev A

**Table 40-17. Device and Package Maximum Weight**

| | |
|---|---|
| 7.3 | mg |

**Table 40-18. Package Characteristics**

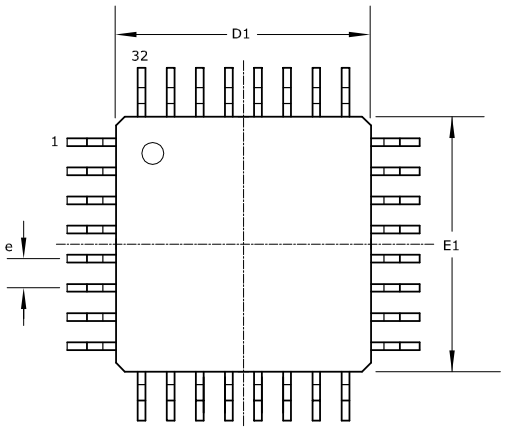
| | |
|---|---|
| Moisture Sensitivity Level | MSL1 |

**Table 40-19. Package Reference**

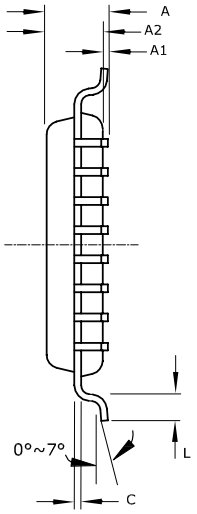
| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E1 |

### 40.2.7 32 pin TQFP

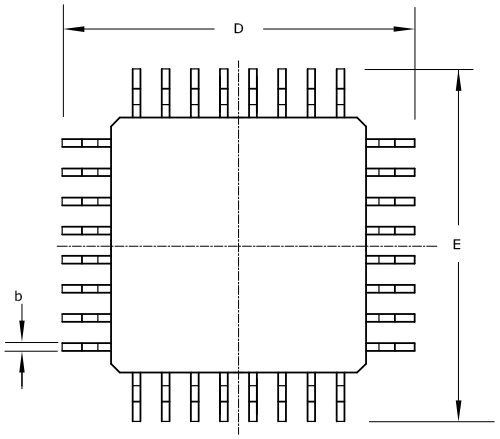
DRAWINGS NOT SCALED




TOP VIEW




SIDE VIEW




BOTTOM VIEW

COMMON DIMENSIONS

(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | ----- | ----- | 1.20 | |
| A1 | 0.05 | ----- | 0.15 | |
| A2 | 0.95 | 1.00 | 1.05 | |
| D/E | 8.75 | 9.00 | 9.25 | |
| D1/E1 | 6.90 | 7.00 | 7.10 | 2 |
| C | 0.09 | ----- | 0.20 | |
| L | 0.45 | ----- | 0.75 | |
| b | 0.30 | ----- | 0.45 | |
| e | 0.80 TYP | | | |
| n | 32 | | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ABA.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

02/29/2012

| | TITLE | GPC | DRAWING NO. | REV. |
|---|---|---|---|---|
| Package Drawing Contact: packagedrawings@atmel.com | MA, 32 Lds - 0.80mm Pitch, 7x7x1.00mm Body size Thin Profile Plastic Quad Flat Package (TQFP) | AUT | MA | C |

**Table 40-20. Device and Package Maximum Weight**

| | |
|---|---|
| 100 | mg |

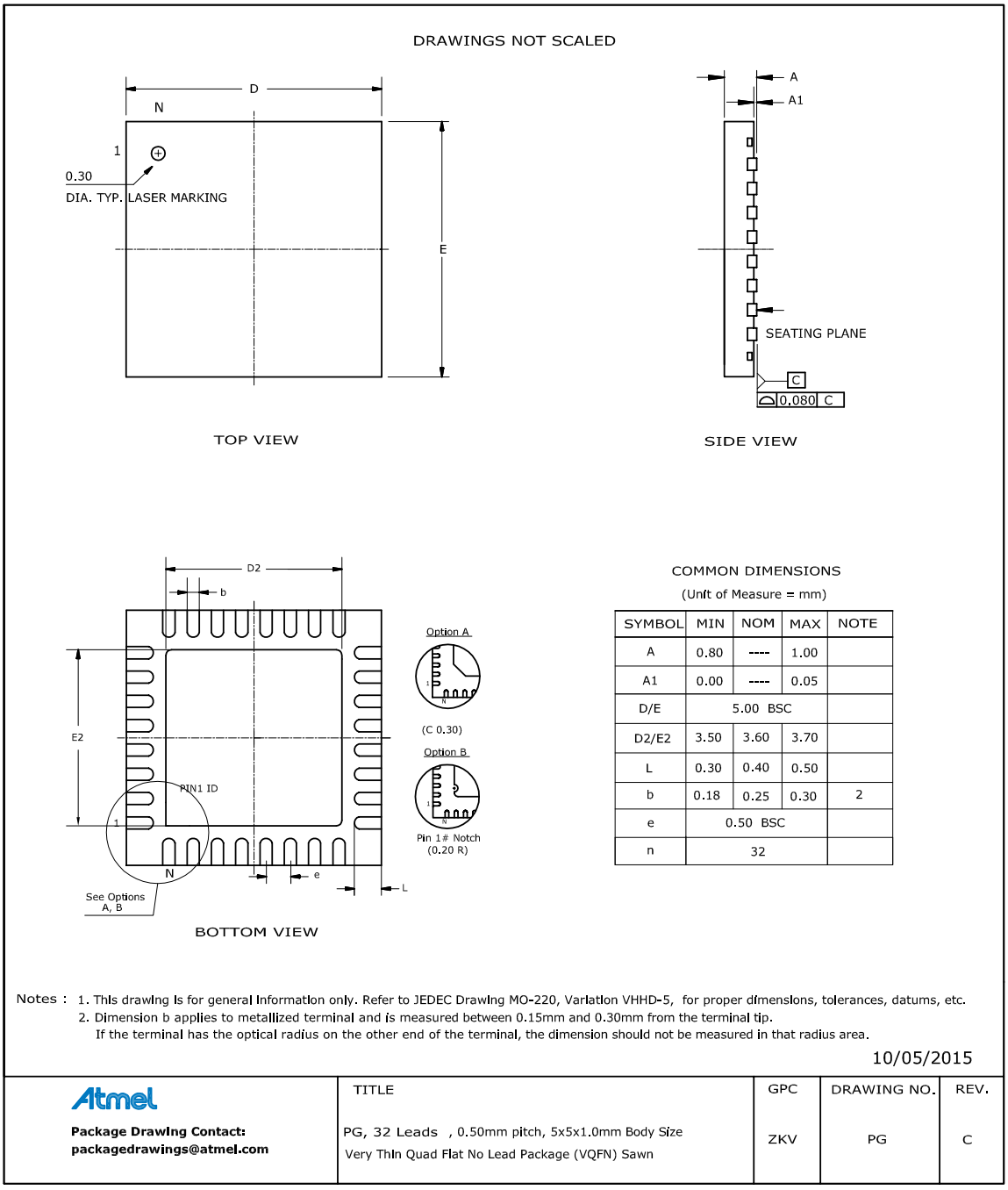
**Table 40-21. Package Charateristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

**Table 40-22. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MS-026 |
| JESD97 Classification | E3 |

### 40.2.8 32 pin QFN




COMMON DIMENSIONS

(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | 0.80 | ---- | 1.00 | |
| A1 | 0.00 | ---- | 0.05 | |
| D/E | | 5.00 BSC | | |
| D2/E2 | 3.50 | 3.60 | 3.70 | |
| L | 0.30 | 0.40 | 0.50 | |
| b | 0.18 | 0.25 | 0.30 | 2 |
| e | | 0.50 BSC | | |
| n | | 32 | | |

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-220, Variation VHHD-5, for proper dimensions, tolerances, datums, etc.
2. Dimension b applies to metallized terminal and is measured between 0.15mm and 0.30mm from the terminal tip.
If the terminal has the optical radius on the other end of the terminal, the dimension should not be measured in that radius area.

10/05/2015

| Atmel | TITLE | GPC | DRAWING NO. | REV. |
|---|---|---|---|---|
| **Package Drawing Contact:** **packagedrawings@atmel.com** | PG, 32 Leads , 0.50mm pitch, 5x5x1.0mm Body Size Very Thin Quad Flat No Lead Package (VQFN) Sawn | ZKV | PG | C |

**Note:** The exposed die attach pad is connected inside the device to GND and GNDANA.

**Table 40-23. Device and Package Maximum Weight**

| | |
|---|---|
| 90 | mg |

**Table 40-24. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL3 |

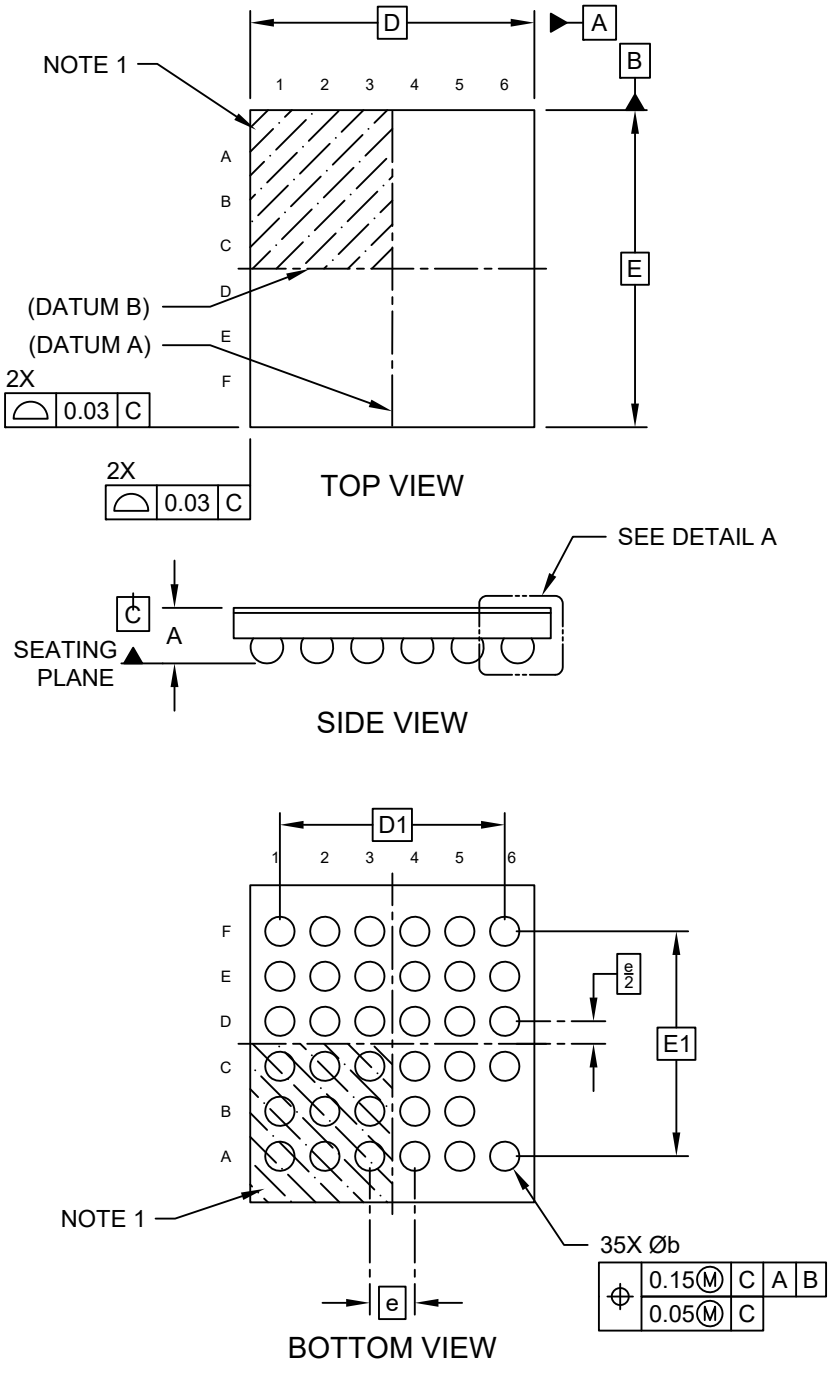
**Table 40-25. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E3 |

### 40.2.9 35 ball WLCSP (Device Variant B)

**35-Ball Wafer Level Chip Scale Package (FQB) - 2.821x2.529 mm Body [WLCSP]**

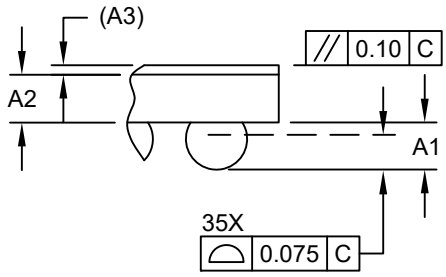
**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



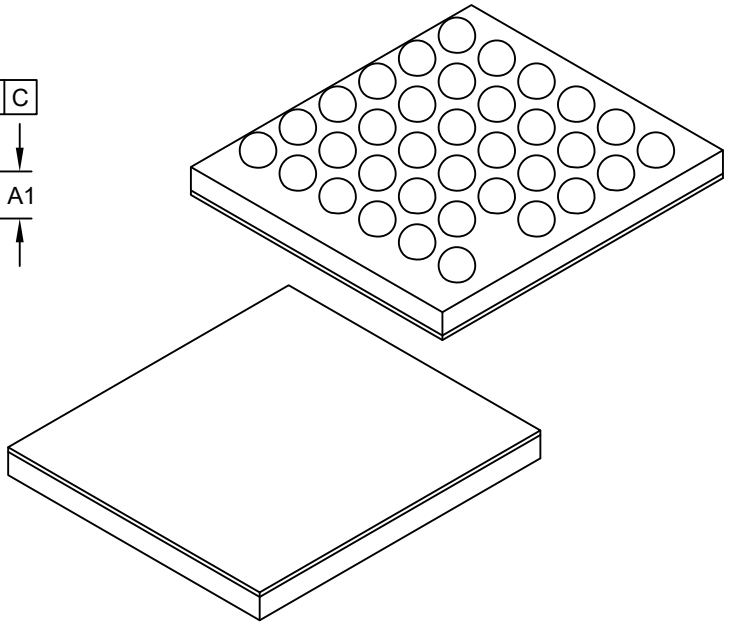

Microchip Technology Drawing C04-21245 Rev A Sheet 1 of 2

## 35-Ball Wafer Level Chip Scale Package (FQB) - 2.821x2.529 mm Body [WLCSP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




DETAIL A



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 35 | |
| Pitch | e | | 0.40 BSC | |
| Overall Height | A | - | - | 0.483 |
| Bump Height | A1 | 0.17 | 0.20 | 0.23 |
| Die Thickness | A2 | 0.178 | 0.203 | 0.228 |
| Backside Coating | A3 | | 0.04 REF | |
| Overall Length | D | | 2.529 BSC | |
| Overall Bump Pitch | D1 | | 2.00 BSC | |
| Overall Width | E | | 2.821 BSC | |
| Overall Bump Pitch | E1 | | 2.00 BSC | |
| Terminal Width | b | 0.23 | 0.26 | 0.29 |

Notes:

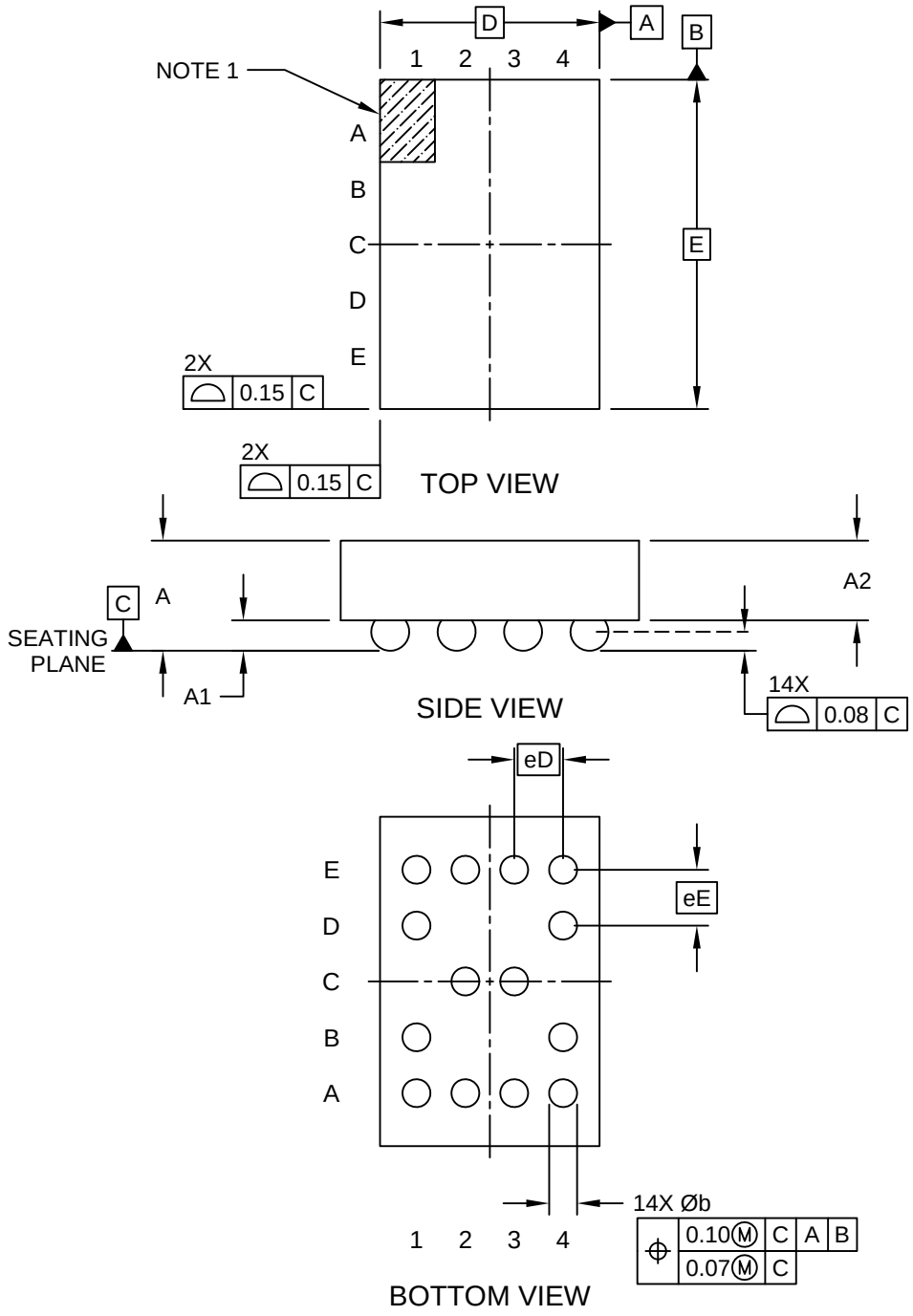
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21245 Rev A Sheet 2 of 2

### 14-Ball Wafer Level Chipscale Package (CS) - 1.57X2.36 Body [WLCSP] - PIC16LF1823

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




Microchip Technology Drawing C04-6009C Sheet 1 of 2

**Table 40-26. Device and Package Maximum Weight**

| | |
|---|---|
| 6.2 | mg |

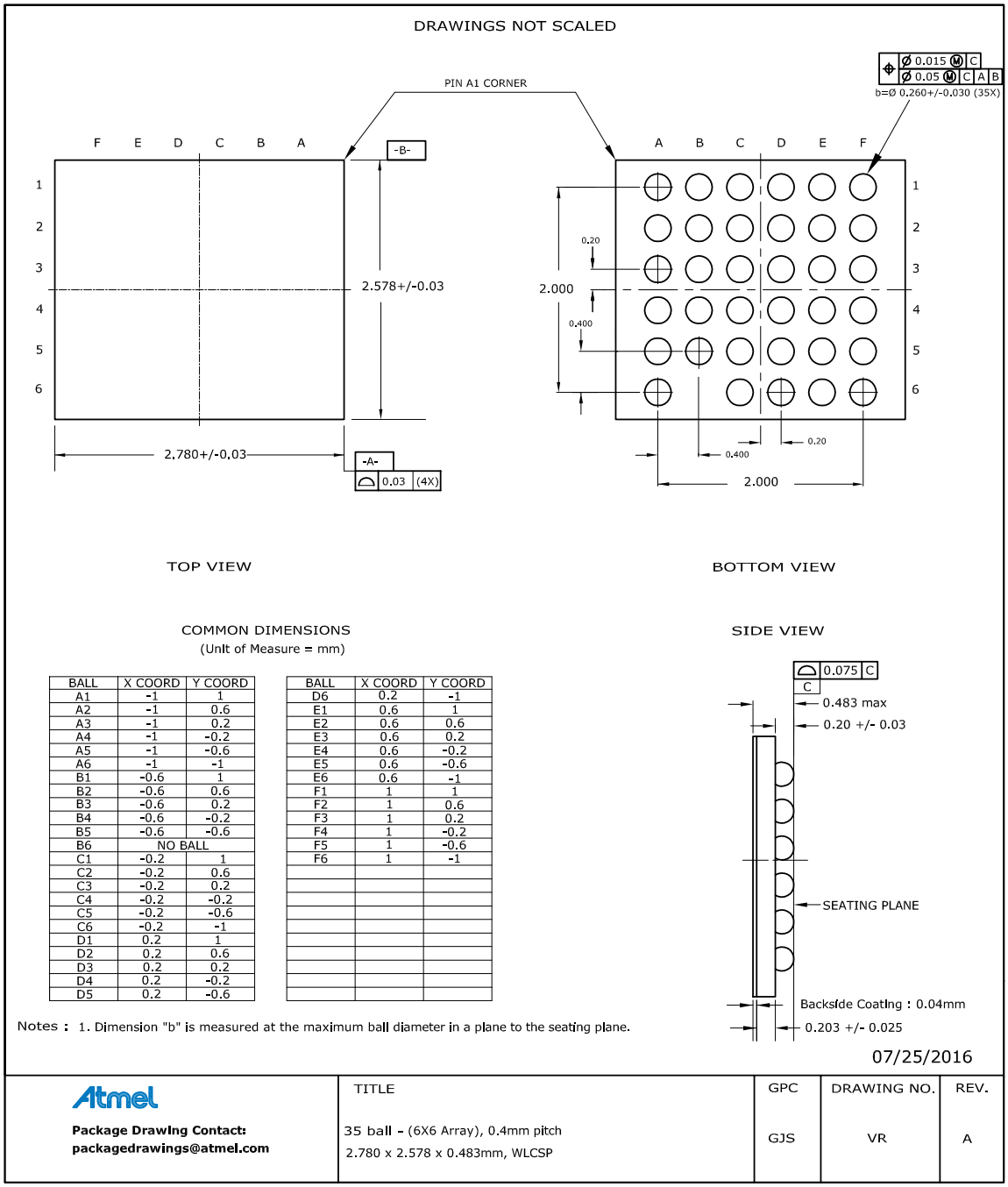
**Table 40-27. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL1 |

**Table 40-28. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | MO-220 |
| JESD97 Classification | E1 |

### 40.2.10 35 ball WLCSP (Device Variant C)




| BALL | X COORD | Y COORD |
|---|---|---|
| A1 | -1 | 1 |
| A2 | -1 | 0.6 |
| A3 | -1 | 0.2 |
| A4 | -1 | -0.2 |
| A5 | -1 | -0.6 |
| A6 | -1 | -1 |
| B1 | -0.6 | 1 |
| B2 | -0.6 | 0.6 |
| B3 | -0.6 | 0.2 |
| B4 | -0.6 | -0.2 |
| B5 | -0.6 | -0.6 |
| B6 | NO BALL | |
| C1 | -0.2 | 1 |
| C2 | -0.2 | 0.6 |
| C3 | -0.2 | 0.2 |
| C4 | -0.2 | -0.2 |
| C5 | -0.2 | -0.6 |
| C6 | -0.2 | -1 |
| D1 | 0.2 | 1 |
| D2 | 0.2 | 0.6 |
| D3 | 0.2 | 0.2 |
| D4 | 0.2 | -0.2 |
| D5 | 0.2 | -0.6 |

| BALL | X COORD | Y COORD |
|---|---|---|
| D6 | 0.2 | -1 |
| E1 | 0.6 | 1 |
| E2 | 0.6 | 0.6 |
| E3 | 0.6 | 0.2 |
| E4 | 0.6 | -0.2 |
| E5 | 0.6 | -0.6 |
| E6 | 0.6 | -1 |
| F1 | 1 | 1 |
| F2 | 1 | 0.6 |
| F3 | 1 | 0.2 |
| F4 | 1 | -0.2 |
| F5 | 1 | -0.6 |
| F6 | 1 | -1 |

Notes : 1. Dimension "b" is measured at the maximum ball diameter in a plane to the seating plane.

**Table 40-29. Device and Package Maximum Weight**

| | |
|---|---|
| 6.22 | mg |

**Table 40-30. Package Characteristics**

| | |
|---|---|
| Moisture Sensitivity Level | MSL1 |

**Table 40-31. Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | N/A |
| JESD97 Classification | e1 |

## 40.3 Soldering Profile

The following table gives the recommended soldering profile from J-STD-20.

**Table 40-32. Recommended Soldering Profile**

| Profile Feature | Green Package |
|---|---|
| Average Ramp-up Rate (217°C to peak) | 3°C/s max. |
| Preheat Temperature 175°C ±25°C | 150-200°C |
| Time Maintained Above 217°C | 60-150s |
| Time within 5°C of Actual Peak Temperature | 30s |
| Peak Temperature Range | 260°C |
| Ramp-down Rate | 6°C/s max. |
| Time 25°C to Peak Temperature | 8 minutes max. |

A maximum of three reflow passes is allowed per component.

# 41. Schematic Checklist

## 41.1 Introduction

This chapter describes a common checklist which should be used when starting and reviewing the schematics for a SAM D21 design. This chapter illustrates a recommended power supply connection, how to connect external analog references, programmer, debugger, oscillator and crystal.

### 41.1.1 Operation in Noisy Environment

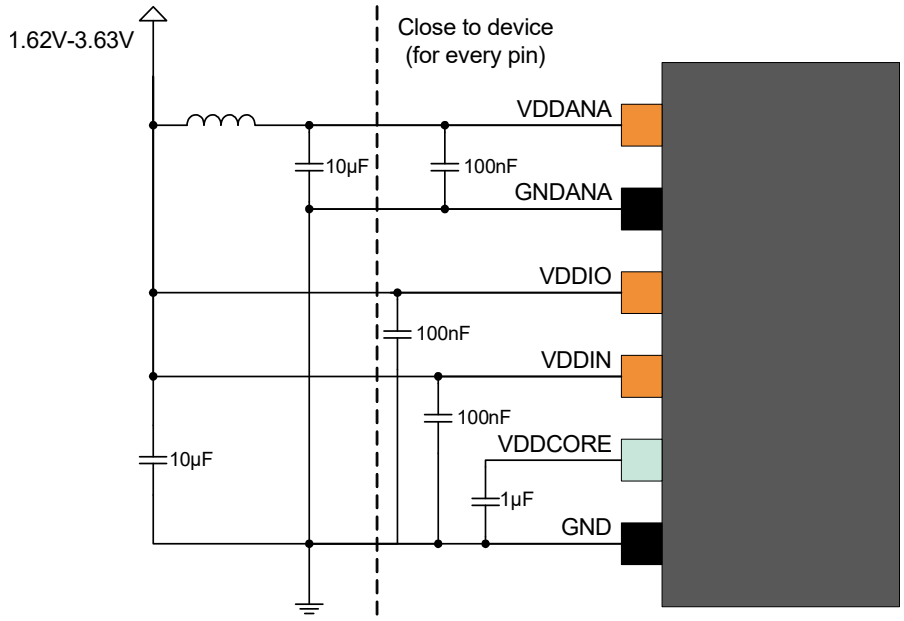
If the device is operating in an environment with much electromagnetic noise, it must be protected from this noise to ensure reliable operation. In addition to following best practice EMC design guidelines, the recommendations listed in the schematic checklist sections must be followed. In particular, placing decoupling capacitors very close to the power pins, an RC-filter on the $\overline{\text{RESET}}$ pin, and a pull-up resistor on the SWCLK pin is critical for reliable operations. It is also relevant to eliminate or attenuate noise in order to avoid that it reaches supply pins, I/O pins and crystals.

## 41.2 Power Supply

The SAM D21 supports a single power supply from 1.62V - 3.63V.

### 41.2.1 Power Supply Connections

**Figure 41-1. Power Supply Schematic(1)**




**Note:** 1. It is recommended to use a ceramic or solid tantalum capacitor with low ESR. Refer to table 37-18 in 37.10.1 Voltage Regulator Characteristics for additional details on ESR.

**Table 41-1. Power Supply Connections, $V_{DDCORE}$ From Internal Regulator**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| $V_{DDIO}$ | 1.62V - 3.63V<br>Decoupling/filtering capacitors 100nF(1)(2) and 10µF(1) | Digital supply voltage |

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| | Decoupling/filtering inductor 10µH(1)(3) | |
| $V_{DDANA}$ | 1.62V - 3.63V<br>Decoupling/filtering capacitors 100nF(1)(2) and 10µF(1)<br>Ferrite bead(4) prevents the $V_{DD}$ noise interfering the $V_{DDANA}$ | Analog supply voltage |
| $V_{DDCORE}$ | 1.6V to 1.8V<br>Decoupling/filtering capacitor 1µF(1)(2) | Core supply voltage / external decoupling pin |
| GND | | Ground |
| $GND_{ANA}$ | | Ground for the analog power domain |

**Note:**

1. These values are only given as typical examples.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group, low ESR caps should be used for better decoupling.
3. An inductor should be added between the external power and the $V_{DD}$ for power filtering.
4. Ferrite bead has better filtering performance than the common inductor at high frequencies. It can be added between $V_{DD}$ and $V_{DDANA}$ for preventing digital noise from entering the analog power domain. The bead should provide enough impedance (e.g. 50Ω at 20MHz and 220Ω at 100MHz) for separating the digital power from the analog power domain. Make sure to select a ferrite bead designed for filtering applications with a low DC resistance to avoid a large voltage drop across the ferrite bead.

## 41.3 External Analog Reference Connections

The following schematic checklist is only necessary if the application is using one or more of the external analog references. If the internal references are used instead, the following circuits are not necessary.

**Figure 41-2. External Analog Reference Schematic With Two References**

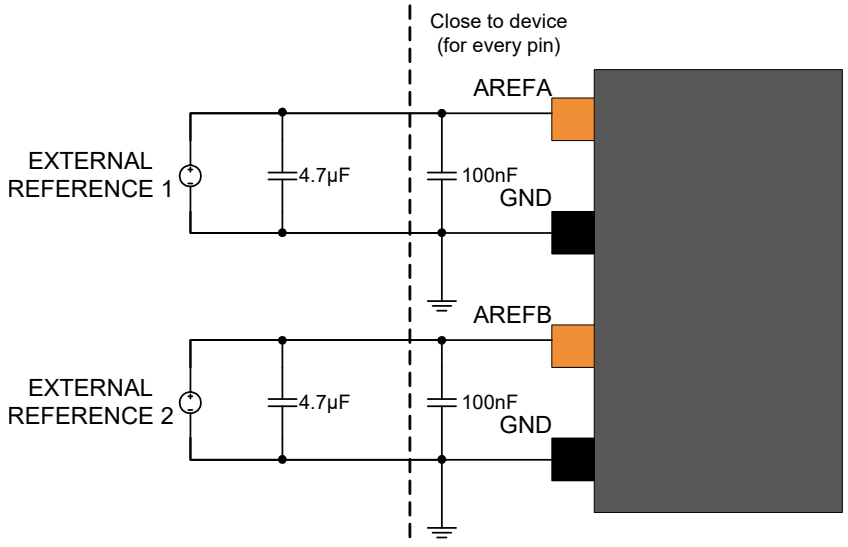


**Figure 41-3. External Analog Reference Schematic With One Reference**

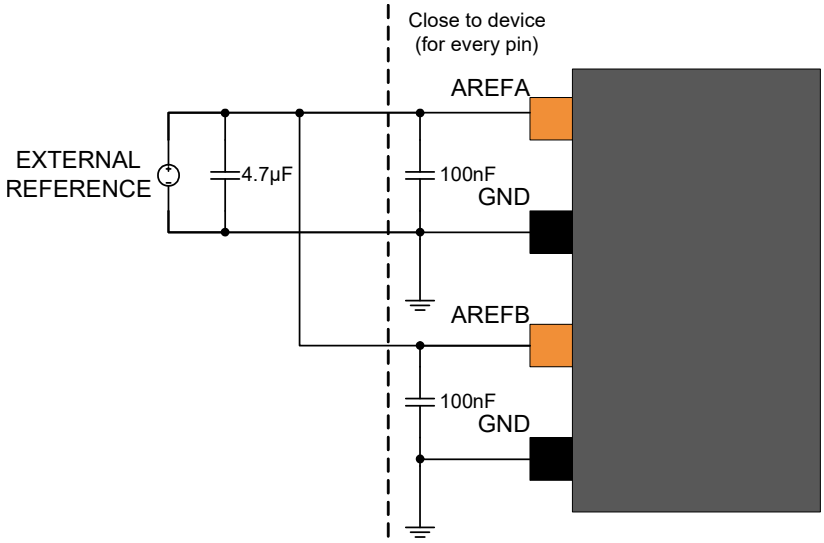



**Table 41-2. External Analog Reference Connections**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| AREFx | 1.0V to $V_{DDANA}$ - 0.6V for ADC<br>1.0V to $V_{DDANA}$- 0.6V for DAC<br>Decoupling/filtering capacitors 100nF(1)(2) and 4.7µF(1) | External reference from AREFx pin on the analog port |
| GND | | Ground |

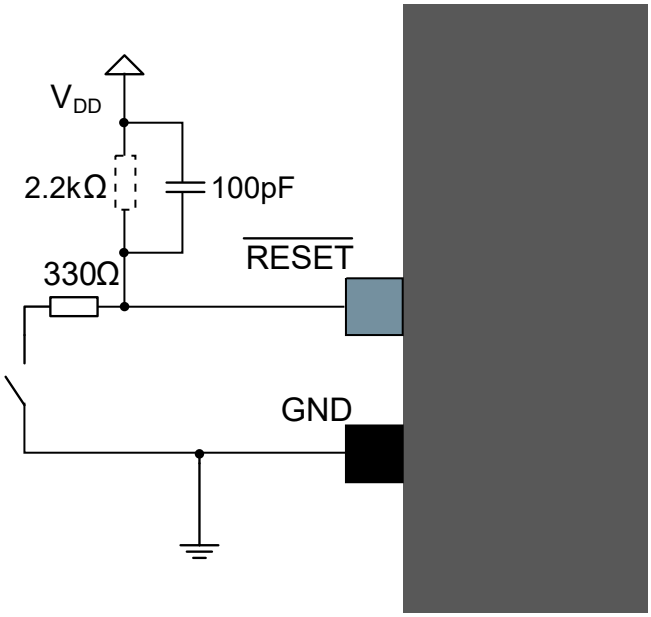
1. These values are given as a typical example.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

## 41.4 External Reset Circuit

The external Reset circuit is connected to the $\overline{\text{RESET}}$ pin when the external Reset function is used. The circuit is not necessary when the $\overline{\text{RESET}}$ pin is not driven LOW externally by the application circuitry.

The reset switch can also be removed, if a manual reset is not desired. The $\overline{\text{RESET}}$ pin itself has an internal pull-up resistor, hence it is optional to add any external pull-up resistor.

**Figure 41-4. External Reset Circuit Schematic**




A pull-up resistor makes sure that the reset does not go low and unintentionally cause a device reset. An additional resistor has been added in series with the switch to safely discharge the filtering capacitor, i.e., preventing a current surge when shorting the filtering capacitor which again can cause a noise spike that can have a negative effect on the system.

**Table 41-3. Reset Circuit Connections**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| $\overline{\text{RESET}}$ | Reset low level threshold voltage<br>$V_{DDIO}$ = 1.6V - 2.0V: Below 0.33 * $V_{DDIO}$ $V_{DDIO}$ = 2.7V - 3.6V: Below 0.36 * $V_{DDIO}$ Decoupling/filter capacitor 100 pF(1) Pull-up resistor 2.2 kΩ(1) (2) Resistor in series with the switch 330Ω(1) | Reset pin |

1. These values are given as a typical example.
2. The SAM D21 features an internal pull-up resistor on the $\overline{\text{RESET}}$ pin, hence an external pull-up is optional.

## 41.5 Clocks and Crystal Oscillators

The SAM D21 can be run from internal or external clock sources, or a mix of internal and external sources. An example of usage will be to use the internal 8MHz oscillator as source for the system clock, and an external 32.768kHz watch crystal as clock source for the Real-Time counter (RTC).

### 41.5.1 External Clock Source

**Figure 41-5. External Clock Source Example Schematic**

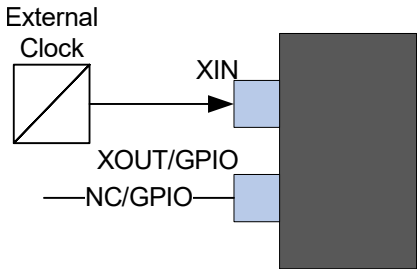
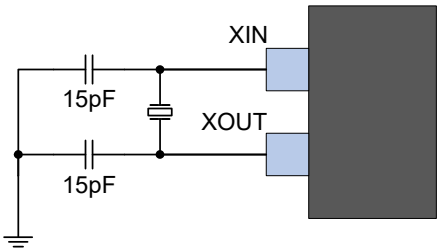



**Table 41-4. External Clock Source Connections**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| XIN | XIN is used as input for an external clock signal | Input for inverting oscillator pin |
| XOUT/GPIO | Can be left unconnected or used as normal GPIO | |

### 41.5.2 Crystal Oscillator

**Figure 41-6. Crystal Oscillator Example Schematic**




The crystal should be located as close to the device as possible. Long signal lines may cause a load too high to operate the crystal, and cause crosstalk to other parts of the system.

**Table 41-5. Crystal Oscillator Checklist**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| XIN | Load capacitor 15pF(1)(2) | External crystal between 0.4 to 30MHz |
| XOUT | Load capacitor 15pF(1)(2) | |

1. These values are given only as typical examples.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

### 41.5.3 External Real Time Oscillator

The low frequency crystal oscillator is optimized for use with a 32.768kHz watch crystal. When selecting crystals, load capacitance and crystal's Equivalent Series Resistance (ESR) must be taken into consideration. Both values are specified by the crystal vendor.

The SAM D21 oscillator is optimized for very low power consumption, so pay close attention when selecting crystals. See the table below for maximum ESR recommendations on 9pF and 12.5pF crystals.

The low-frequency crystal oscillator provides an internal load capacitance of typical values available in Table , *32kHz Crystal Oscillator Characteristics*. This internal load capacitance and PCB capacitance can

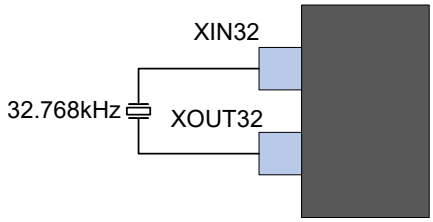
use a crystal inferior to 12.5pF load capacitance without external capacitors as shown in the following figure.

**Table 41-6. Maximum ESR Recommendation for 32.768kHz Crystal**

| Crystal $C_L$ (pF) | Max ESR [kΩ] |
|---|---|
| 12.5 | 313 |

Note: Maximum ESR is typical value based on characterization. These values are not covered by test limits in production.

**Figure 41-7. External Real Time Oscillator without Load Capacitor**




However, to improve crystal accuracy and safety factor, the data sheet recommends adding external capacitors as shown in the next figure.

To find suitable load capacitance for a 32.768kHz crystal, consult the crystal datasheet.

**Figure 41-8. External Real Time Oscillator with Load Capacitor**

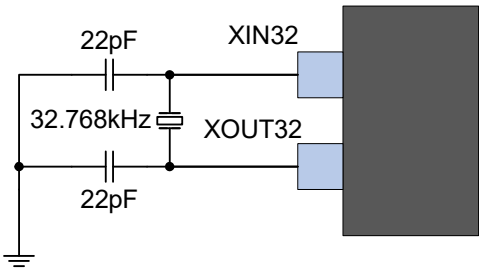



**Table 41-7. External Real Time Oscillator Checklist**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| XIN32 | Load capacitor 22pF(1)(2) | Timer oscillator input |
| XOUT32 | Load capacitor 22pF(1)(2) | Timer oscillator output |

1. These values are given only as typical examples.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

**Note:** In order to minimize the cycle-to-cycle jitter of the external oscillator, keep the neighboring pins as steady as possible. For neighboring pin details, refer to the Oscillator Pinout section.

**Related Links**

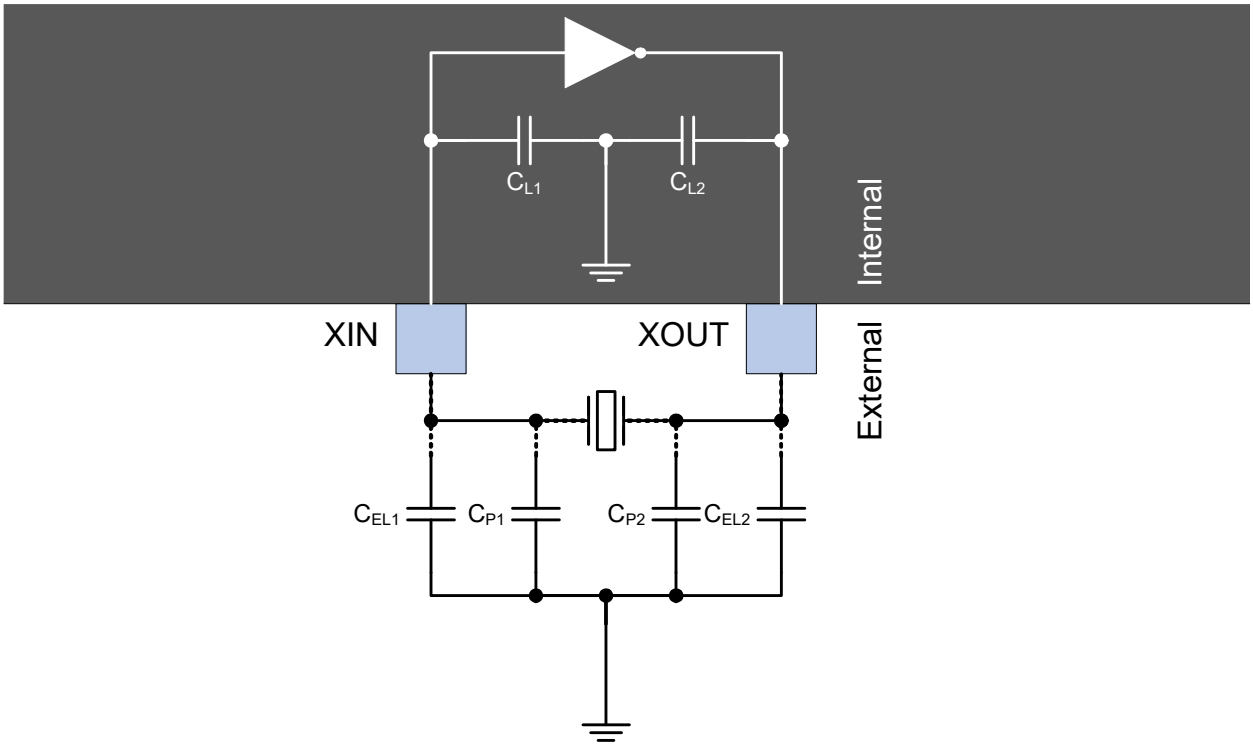
7.2.1 Oscillator Pinout

37.12 Oscillators Characteristics

### 41.5.4 Calculating the Correct Crystal Decoupling Capacitor

In order to calculate correct load capacitor for a given crystal, use the model shown in the next figure, which includes internal capacitors $C_{Ln}$, external parasitic capacitance $C_{ELn}$ and external load capacitance $C_{Pn}$.

**Figure 41-9. Crystal Circuit With Internal, External and Parasitic Capacitance**




Using this model the total capacitive load for the crystal can be calculated as shown in the equation below:

$$\sum C_{\text{tot}} = \frac{(C_{L1} + C_{P1} + C_{\text{EL1}})(C_{L2} + C_{P2} + C_{\text{EL2}})}{C_{L1} + C_{P1} + C_{\text{EL1}} + C_{L2} + C_{P2} + C_{\text{EL2}}}$$

where $C_{tot}$ is the total load capacitance seen by the crystal, this value should be equal to the load capacitance value found in the crystal manufacturer datasheet.

The parasitic capacitance $C_{ELn}$ can be disregarded in most applications as these are usually very small. If accounted for, the value is dependent on the PCB material and PCB layout.

For some crystals, the internal capacitive load provided by the device itself can be enough to calculate the total load capacitance in this case. $C_{ELn}$ and $C_{Pn}$ are both zero, $C_{L1} = C_{L2} = C_L$, and the equation reduces to the following:

$$\sum C_{\text{tot}} = \frac{C_L}{2}$$

The next table shows the device equivalent internal pin capacitance.

**Table 41-8. Equivalent Internal Pin Capacitance**

| Symbol | Value | Description |
|---|---|---|
| $C_{XIN32}$ | 3.05pF | Equivalent internal pin capacitance |
| $C_{XOUT32}$ | 3.29pF | Equivalent internal pin capacitance |

## 41.6 Unused or Unconnected Pins

For unused pins, the default state of the pins for the will provide the lowest current leakage. There is no need to do any configuration of the unused pins in order to lower the power consumption.

## 41.7 Programming and Debug Ports

For programming and/or debugging the SAM D21 the device should be connected using the Serial Wire Debug, SWD, interface. Currently the SWD interface is supported by several Microchip and third party programmers and debuggers, like the JTAGICE3, SAM-ICE, ATMEL_ICE or SAM D21 Xplained Pro ( SAM D21 evaluation kit) Embedded Debugger.

Refer to the JTAGICE3, SAM-ICE, ATMEL_ICE or SAM D21 Xplained Pro user guides for details on debugging and programming connections and options. For connecting to any other programming or debugging tool, refer to that specific programmer or debugger's user guide.

The SAM D21 Xplained Pro evaluation board for the SAM D21 supports programming and debugging through the onboard embedded debugger so no external programmer or debugger is needed.

Note that a pull-up resistor on the SWCLK pin is critical for reliable operations. Refer to related link for more information.

**Figure 41-10. SWCLK Circuit Connections**

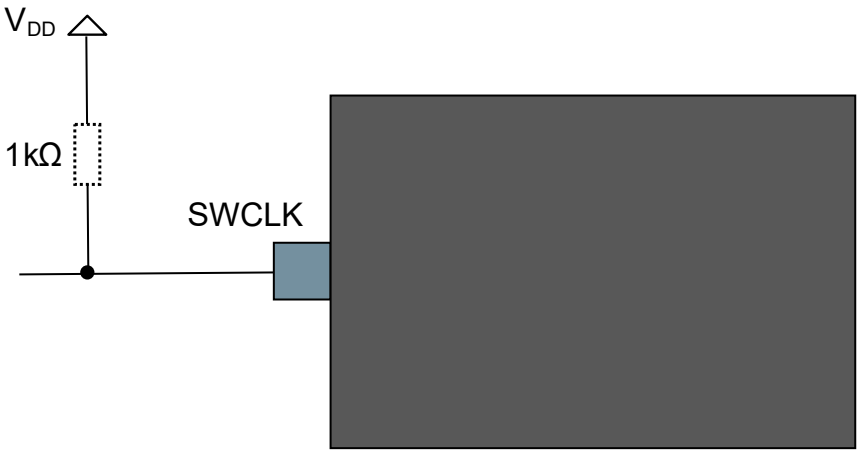



**Table 41-9. SWCLK Circuit Connections**

| Pin Name | Description | Recommended Pin Connection |
|---|---|---|
| SWCLK | Serial wire clock pin | Pull-up resistor 1kΩ |

**Related Links**

41.1.1 Operation in Noisy Environment

### 41.7.1 Cortex Debug Connector (10-pin)

For debuggers and/or programmers that support the Cortex Debug Connector (10-pin) interface the signals should be connected as shown in the figure below with details described in the next table.

**Figure 41-11. Cortex Debug Connector (10-pin)**

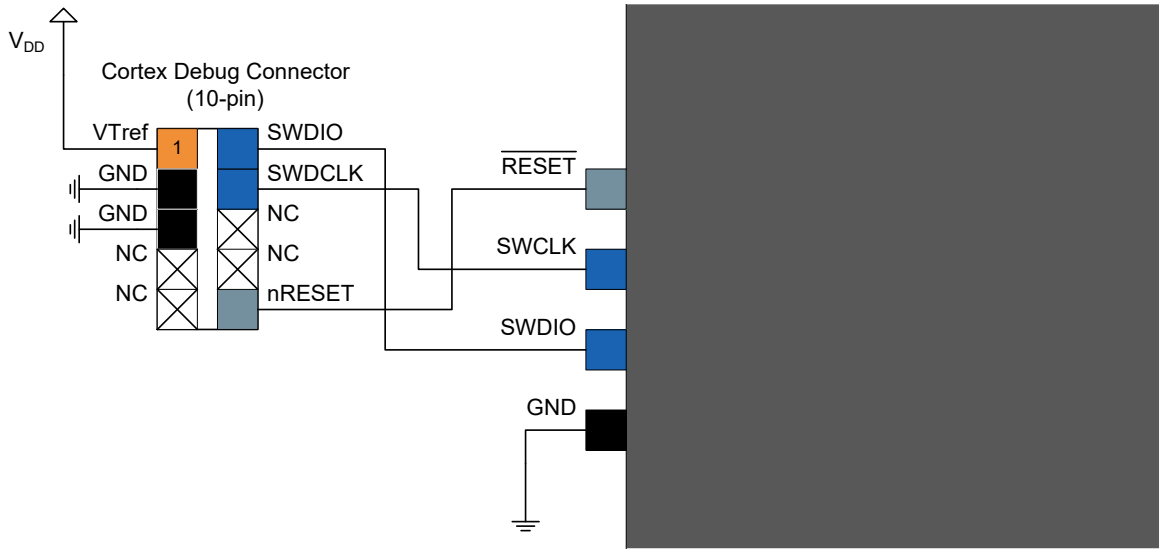



**Table 41-10. Cortex Debug Connector (10-pin)**

| Header Signal Name | Description | Recommended Pin Connection |
|---|---|---|
| SWDCLK | Serial wire clock pin | Pull-up resistor 1kΩ |
| SWDIO | Serial wire bidirectional data pin | |
| $\overline{\text{RESET}}$ | Target device reset pin, active low<br>Refer to 41.4 External Reset Circuit. | |
| VTref | Target voltage sense, should be connected to the device $V_{DD}$ | |
| GND | Ground | |

### 41.7.2 10-pin JTAGICE3 Compatible Serial Wire Debug Interface

The JTAGICE3 debugger and programmer does not support the Cortex Debug Connector (10-pin) directly, hence a special pinout is needed to directly connect the SAM D21 to the JTAGICE3, alternatively one can use the JTAGICE3 squid cable and manually match the signals between the JTAGICE3 and SAM D21. The following figure describes how to connect a 10-pin header that support connecting the JTAGICE3 directly to the SAM D21 without the need for a squid cable.

To connect the JTAGICE3 programmer and debugger to the SAM D21, one can either use the JTAGICE3 squid cable, or use a 10-pin connector as shown in the figure below with details given in the next table to connect to the target using the JTAGICE3 50 mil cable directly.

**Figure 41-12. 10-pin JTAGICE3 Compatible Serial Wire Debug Interface**

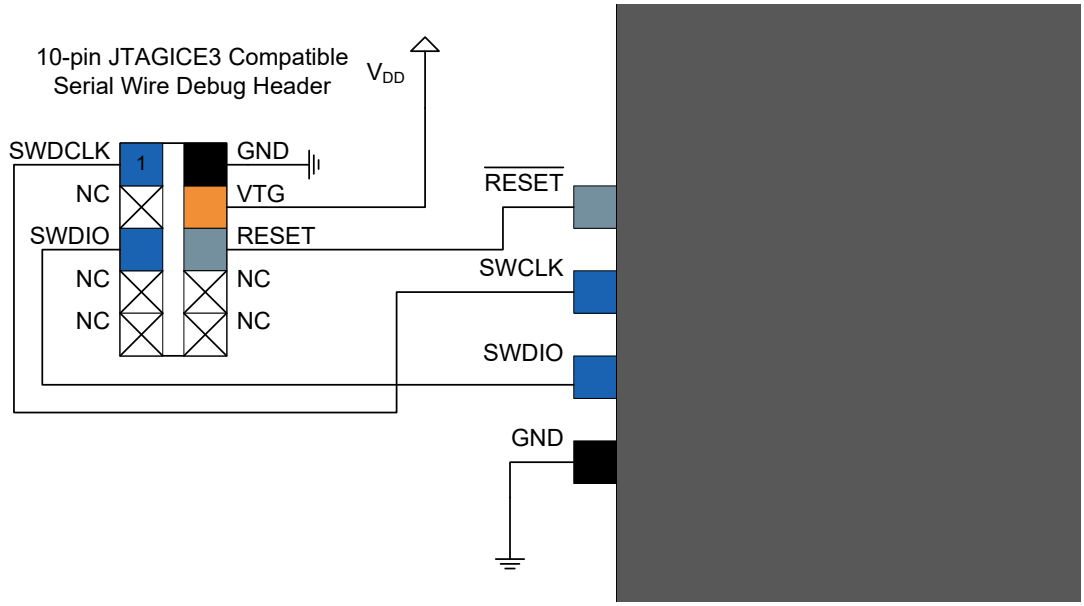



**Table 41-11. 10-pin JTAGICE3 Compatible Serial Wire Debug Interface**

| Header Signal Name | Description |
|---|---|
| SWDCLK | Serial wire clock pin |
| SWDIO | Serial wire bidirectional data pin |
| $\overline{\text{RESET}}$ | Target device reset pin, active low |
| VTG | Target voltage sense, should be connected to the device $V_{DD}$ |
| GND | Ground |

### 41.7.3 20-pin IDC JTAG Connector

For debuggers and/or programmers that support the 20-pin IDC JTAG Connector, e.g. the SAM-ICE, the signals should be connected as shown in the next figure with details described in the table.

**Figure 41-13. 20-pin IDC JTAG Connector**

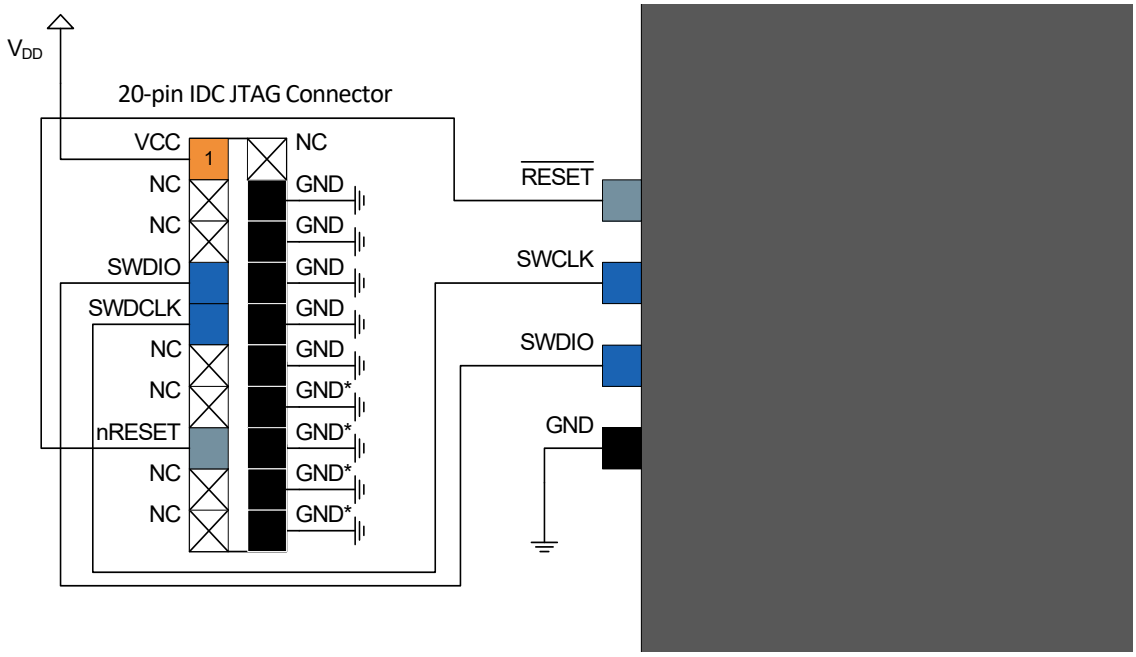



**Table 41-12. 20-pin IDC JTAG Connector**

| Header Signal Name | Description |
|---|---|
| SWDCLK | Serial wire clock pin |
| SWDIO | Serial wire bidirectional data pin |
| $\overline{\text{RESET}}$ | Target device reset pin, active low |
| VCC | Target voltage sense, should be connected to the device $V_{DD}$ |
| GND | Ground |
| GND* | These pins are reserved for firmware extension purposes. They can be left open or connected to GND in normal debug environment. They are not essential for SWD in general. |

## 41.8 USB Interface

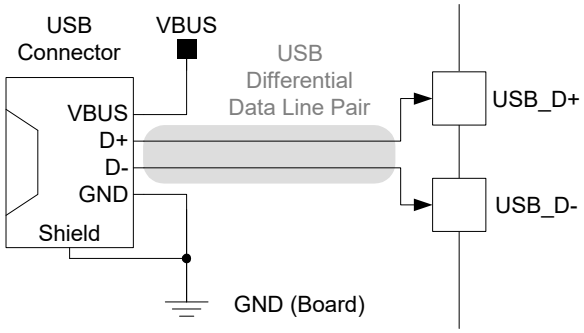
The USB interface consists of a differential data pair (D+/D-) and a power supply (VBUS, GND). Refer to the Electrical Characteristics for operating voltages which will allow USB operation.

**Table 41-13. USB Interface Checklist**

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| D+ | • The impedance of the pair should be matched on the PCB to minimize reflections.<br>• USB differential tracks should be routed with the same characteristics (length, width, number of vias, etc.) | USB full speed / low speed positive data upstream pin |

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| D- | • Signals should be routed as parallel as possible, with a minimum number of angles and vias | USB full speed / low speed negative data upstream pin |

**Figure 41-14. Low Cost USB Interface Example Schematic**

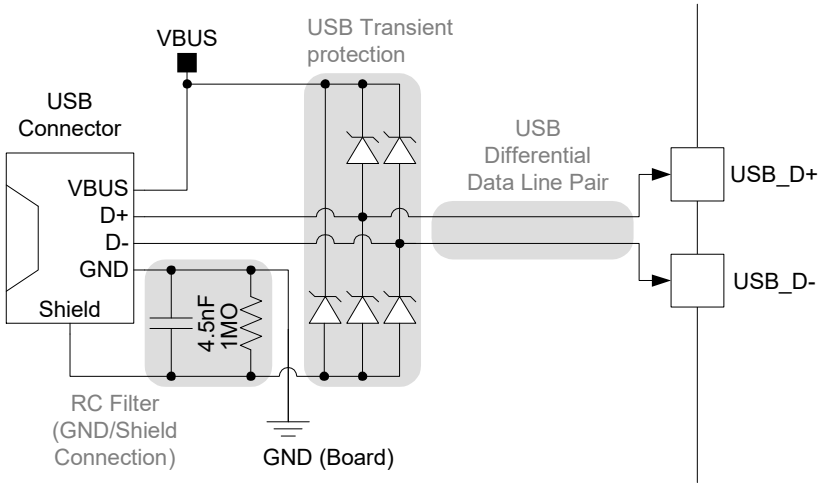



It is recommended to increase ESD protection on the USB D+, D-, and VBUS lines using dedicated transient suppressors. These protections should be located as close as possible to the USB connector to reduce the potential discharge path and reduce discharge propagation within the entire system.

The USB FS cable includes a dedicated shield wire that should be connected to the board with caution. Special attention should be paid to the connection between the board ground plane and the shield from the USB connector and the cable.

Tying the shield directly to ground would create a direct path from the ground plane to the shield, turning the USB cable into an antenna. To limit the USB cable antenna effect, it is recommended to connect the shield and ground through an RC filter.

**Figure 41-15. Protected USB Interface Example Schematic**



# 42. Conventions

## 42.1 Numerical Notation

**Table 42-1. Numerical Notation**

| Symbol | Description |
|---|---|
| 165 | Decimal number |
| 0b0101 | Binary number (example 0b0101 = 5 decimal) |
| '0101' | Binary numbers are given without prefix if unambiguous |
| 0x3B24 | Hexadecimal number |
| X | Represents an unknown or don't care value |
| Z | Represents a high-impedance (floating) state for either a signal or a bus |

## 42.2 Memory Size and Type

**Table 42-2. Memory Size and Bit Rate**

| Symbol | Description |
|---|---|
| KB (kbyte) | kilobyte ($2^{10}$ = 1024) |
| MB (Mbyte) | megabyte ($2^{20}$ = 1024*1024) |
| GB (Gbyte) | gigabyte ($2^{30}$ = 1024*1024*1024) |
| b | bit (binary '0' or '1') |
| B | byte (8 bits) |
| 1kbit/s | 1,000 bit/s rate (not 1,024 bit/s) |
| 1Mbit/s | 1,000,000 bit/s rate |
| 1Gbit/s | 1,000,000,000 bit/s rate |
| word | 32 bit |
| half-word | 16 bit |

## 42.3 Frequency and Time

**Table 42-3. Frequency and Time**

| Symbol | Description |
|---|---|
| kHz | 1 kHz = $10^3$ Hz = 1,000 Hz |
| KHz | 1 KHz = 1,024 Hz, 32 KHz = 32,768 Hz |

| Symbol | Description |
|---|---|
| MHz | 1 MHz = $10^6$ Hz = 1,000,000 Hz |
| GHz | 1 GHz = $10^9$ Hz = 1,000,000,000 Hz |
| s | second |
| ms | millisecond |
| µs | microsecond |
| ns | nanosecond |

## 42.4 Registers and Bits

**Table 42-4. Register and Bit Mnemonics**

| Symbol | Description |
|---|---|
| R/W | Read/Write accessible register bit. The user can read from and write to this bit. |
| R | Read-only accessible register bit. The user can only read this bit. Writes will be ignored. |
| W | Write-only accessible register bit. The user can only write this bit. Reading this bit will return an undefined value. |
| BIT | Bit names are shown in uppercase. (Example ENABLE) |
| FIELD[n:m] | A set of bits from bit n down to m. (Example: PINA[3:0] = {PINA3, PINA2, PINA1, PINA0} |
| Reserved | Reserved bits are unused and reserved for future use. For compatibility with future devices, always write reserved bits to zero when the register is written. Reserved bits will always return zero when read.<br>Reserved bit field values must not be written to a bit field. A reserved value will not be read from a read-only bit field. |
| PERIPHERAL*i* | If several instances of a peripheral exist, the peripheral name is followed by a number to indicate the number of the instance in the range 0-n. PERIPHERAL0 denotes one specific instance. |
| Reset | Value of a register after a power Reset. This is also the value of registers in a peripheral after performing a software Reset of the peripheral, except for the Debug Control registers. |
| SET/CLR | Registers with SET/CLR suffix allows the user to clear and set bits in a register without doing a read-modify-write operation. These registers always come in pairs. Writing a ‘1’ to a bit in the CLR register will clear the corresponding bit in both registers, while writing a ‘1’ to a bit in the SET register will set the corresponding bit in both registers. Both registers will return the same value when read. If both registers are written simultaneously, the write to the CLR register will take precedence. |

# 43. Acronyms and Abbreviations

The below table contains acronyms and abbreviations used in this document.

**Table 43-1. Acronyms and Abbreviations**

| Abbreviation | Description |
|---|---|
| AC | Analog Comparator |
| ADC | Analog-to-Digital Converter |
| ADDR | Address |
| AES | Advanced Encryption Standard |
| AHB | Advanced High-performance Bus |
| AMBA® | Advanced Microcontroller Bus Architecture |
| APB | AMBA Advanced Peripheral Bus |
| AREF | Analog reference voltage |
| BLB | Boot Lock Bit |
| BOD | Brown-out Detector |
| CAL | Calibration |
| CC | Compare/Capture |
| CCL | Configurable Custom Logic |
| CLK | Clock |
| CRC | Cyclic Redundancy Check |
| CTRL | Control |
| DAC | Digital-to-Analog Converter |
| DAP | Debug Access Port |
| DFLL | Digital Frequency Locked Loop |
| DPLL | Digital Phase Locked Loop |
| DMAC | DMA (Direct Memory Access) Controller |
| DSU | Device Service Unit |
| EEPROM | Electrically Erasable Programmable Read-Only Memory |
| EIC | External Interrupt Controller |
| EVSYS | Event System |
| FDPLL | Fractional Digital Phase Locked Loop, also DPLL |
| GCLK | Generic Clock Controller |
| GND | Ground |

| Abbreviation | Description |
|---|---|
| GPIO | General Purpose Input/Output |
| $I^2C$ | Inter-Integrated Circuit |
| IF | Interrupt flag |
| INT | Interrupt |
| MBIST | Memory built-in self-test |
| MEM-AP | Memory Access Port |
| MTB | Micro Trace Buffer |
| NMI | Non-maskable interrupt |
| NVIC | Nested Vector Interrupt Controller |
| NVM | Non-Volatile Memory |
| NVMCTRL | Non-Volatile Memory Controller |
| OSC | Oscillator |
| PAC | Peripheral Access Controller |
| PC | Program Counter |
| PER | Period |
| PM | Power Manager |
| POR | Power-on reset |
| PORT | I/O Pin Controller |
| PTC | Peripheral Touch Controller |
| PWM | Pulse Width Modulation |
| RAM | Random-Access Memory |
| REF | Reference |
| RTC | Real-Time Counter |
| RX | Receiver/Receive |
| SEEP | SmartEEPROM Page |
| SERCOM | Serial Communication Interface |
| SMBus™ | System Management Bus |
| SP | Stack Pointer |
| SPI | Serial Peripheral Interface |
| SRAM | Static Random-Access Memory |
| SUPC | Supply Controller |
| SWD | Serial Wire Debug |

| Abbreviation | Description |
|---|---|
| TC | Timer/Counter |
| TCC | Timer/Counter for Control Applications |
| TRNG | True Random Number Generator |
| TX | Transmitter/Transmit |
| ULP | Ultra low-power |
| USART | Universal Synchronous and Asynchronous Serial Receiver and Transmitter |
| USB | Universal Serial Bus |
| $V_{DD}$ | Common voltage to be applied to VDDIO, VDDIN and VDDANA |
| $V_{DDIN}$ | Digital supply voltage |
| $V_{DDIO}$ | Digital supply voltage |
| $V_{DDANA}$ | Analog supply voltage |
| VREF | Voltage reference |
| WDT | Watchdog Timer |
| XOSC | Crystal Oscillator |

# 44. Data Sheet Revision History

Page numbers listed in this section refer to this document. The revision listed in this section refers to the document revision.

## 44.1 Rev. B – 04/2018

| | |
|---|---|
| General | • This revision was updated to include the SAM D21EL and SAM D21GL Variant information, which was released separately in DS40001883A. The SAM D21EL/SAM D21GL Data Sheet (DS40001883A) is superseded by this revision (DS40001882B).<br>• IOBUS start addressed is added which was missing in previous revision (DS40001882A). |
| Electrical Characteristics | • Clarified ESR information for VDDCORE capacitor<br>• Typo addressed for VDDIN capacitor value |
| DMA | • RUNSTDBY not supported, typo addressed |
| AC | • Continuous Mode SleepWalking figure is updated |
| ADC | • Bandgap reference as input was omitted in previous version of the data sheet. It is added in this version. |
| Packaging Information | • WLCSP package drawings updated<br>• Thermal characteristic for 45-ball WLCSP & 64-pin UFBGA is added |

## 44.2 Rev. A – 01/2017

| | |
|---|---|
| General | • Template: Updated from Atmel to Microchip template.<br>• Document number: Changed from the Atmel 42181 to Microchip xxxxx.<br>• Document revision letter reset to A.<br>• ISBN number added. |
| Electrical Characteristics | • Die Revision F final characterization added.<br>• 37.6 Power Consumption: Added Standby typical numbers for Device Variant C / Die Revision F.<br>• 37.12.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C / Die Revision F. |
| Errata | • New errata added:<br>– B<br>– Device Variant A: Errata reverence 15625, 15683, 15753 added.<br>– Device Variant B: Errata reverence 15625, 15683, 15753 added. |

| | |
|---|---|
| | – Device Variant C: Errata reverence 15625, 15683, 15753 added. |
| 39. Electrical Characteristics at 125°C | • Die Revision F final characterization is preliminary.<br>• 39.5 Power Consumption: Added Standby typical numbers for Device Variant C / Die Revision F.<br>• 39.8.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C / Die Revision F. |

## 44.3 Rev. O – 12/2016

| | |
|---|---|
| General | • Introduced Device Variant C. |
| 37. Electrical Characteristics | • Die Revision F characterization is preliminary.<br>• 37.6 Power Consumption: Added Standby typical numbers for Device Variant C / Die Revision F.<br>• 37.12.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C / Die Revision F. |
| 39. Electrical Characteristics at 125°C | • Die Revision F characterization is preliminary.<br>• 39.5 Power Consumption: Added Standby typical numbers for Device Variant C / Die Revision F.<br>• 39.8.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C / Die Revision F. |

## 44.4 Rev. N – 10/2016

| | |
|---|---|
| 7. I/O Multiplexing and Considerations | • 7.1 Multiplexed Signals: Updated table note 6 with information on PA24 and PA25. |
| 10. Memories | • 10.3.1 NVM User Row Mapping: Added BOOTPROT default value for WLCSP. |
| 30. TC – Timer/Counter | • 30.5.3 Clocks: Corrected TC instance numbers.<br>• 30.6.2.4 Counter Mode: Corrected TC instance numbers. |
| 37. Electrical Characteristics | • 37.8.1 Normal I/O Pins: Added condition to Pull-up - Pull-down resistance. |

## 44.5 Rev. M – 09/2016

| | |
|---|---|
| 2. Configuration Summary | • Added information on number of pins for the SAM D21G WLCSP pakcage option. SAM D21G is offered in 48 pin packages, while the WLCSP has 45 pins. |
| 3. Ordering Information(1) | • Added information to the pin count explanation. For the The G letter indicates 48 pin packages, while the WLCSP option is 45 pins.<br>• ATSAMD21E18A-MFUT corrected to ATSAMD21E18A-MFT.<br>• Device Identification:<br>– Removed C variants.<br>– Added device identification values for the devices in WLCSP packages. These have separate device id's compared to the other package options. |
| 18. WDT – Watchdog Timer | • 18.5.7 Debug Operation: Removed the sentence "This peripheral can be forced to continue operation during debugging." The WDT can not be forced to continue operation in debug mode. |

## 44.6 Rev. L – 09/2016

| | |
|---|---|
| 2. Configuration Summary | • Added information on number of pins for the WLCSP pakcage. SAM D21E is offered in 32 pin packages, while the WLCSP has 35 pins. |
| 13. DSU - Device Service Unit | • 13.11.5 Testing of On-Board Memories MBIST: Updated description. |
| 14. Clock System | • 14.5 Disabling a Peripheral: New section added. |
| 17. SYSCTRL – System Controller | • 17.8.5 XOSC.AMPGC bit description updated. |
| 21. EIC – External Interrupt Controller | • 21.6.6 Interrupts: Added note explaining how it works when the same external interrupt (EXTINT) is common on sevral pins. |
| 24. EVSYS – Event System | • 24.8.1 CTRL.SWRST: Added recommendation when doing a software reset. |
| 28. SERCOM I2C – Inter-Integrated Circuit | • Corrected cross references in the Master 28.10.1 CTRLA.SCLSM and Slave 28.8.1 CTRLASCLSM bits. |
| 31. TCC – Timer/Counter for Control Applications | • Value 0 in CAPTMIN mode is captured only in down-counting mode. |
| 33. ADC – Analog-to-Digital Converter | • 33.6.5 Differential and Single-Ended Conversions: Corrected register reference from INPUTCTRL.DIFFMODE to CTRLB.DIFFMODE.<br>• 33.8.14 RESULT: Corrected description. Reference to "single-ended mode" corrected to "single conversion mode". |

| | |
|---|---|
| 37. Electrical Characteristics | • 37.2 Absolute Maximum Ratings: Add ESD warnings.<br>• 37.15.5 I2S Timing: $f_{M_SCKO}$ and $f_{M_SCKI}$ values for VDD=1.8V moved from the minimum to maximum column.<br>• XOSC32K 37.12.2.1.1 Crystal Oscillator Characteristics: Removed conditions from the parasitic capacitor loads $C_{XIN32}$ and $C_{XOUT32}$. The difference between package types is so small that it can be ingored. |
| 41. Schematic Checklist | • 41.5.3 External Real Time Oscillator: Added note on how to minimize jitter. |
| 39. Electrical Characteristics at 125°C | • 39.4 Maximum Clock Frequencies: Corrected heading of Table 39-6 say "Device Variant B". |

## 44.7 Rev. K – 09/2016

| | |
|---|---|
| 3. Ordering Information(1) | • SAM D21E: Added Device Variant C ordering codes.<br>• Device Identification: Added Device Variant C. |
| 7. I/O Multiplexing and Considerations | • The section is reorgnaized:<br>– 7.2.3 SERCOM I2C Pins: Replaces the "Type" column in 7.1 Multiplexed Signals.<br>– 7.2.4 GPIO Clusters: Moved from 37.2 Absolute Maximum Ratings.<br>– 7.2.5 TCC Configurations: Moved from the TCC 31.1 Overview. |
| 16. PM – Power Manager | • 16.8.10 APBCMASK updated. |
| 13. DSU - Device Service Unit | • 13.11.6 System Services Availability when Accessed Externally and Device is Protected: MBIST not available when device is operated from external address range and device is protected. |
| 19. RTC – Real-Time Counter | • 19.6.3.3 Clock/Calendar (Mode 2): Example added on how the clock counter works in calendar mode. |
| 30. TC – Timer/Counter | • 30.8.1 CTRLA.WAVEGEN[1:0]: Name column updated. |
| 31. TCC – Timer/ Counter for Control Applications | • 31.6.3.6 Non-Recoverable Faults: Removed references to Update Fault State (UFS).<br>• Removed the UFS bit from the INTENCLR, INTENSET, INTFLAG and STATUS registers.<br>• Removed RAMP2C from the 31.8.16 WAVE.WAVE[2:0]=0x3 |
| 37. Electrical Characteristics | • 37.2 Absolute Maximum Ratings: Updated $V_{PIN}$ minimum and maximum values. (Related to the new Injection Current definition section) |

| | |
|---|---|
| | • 37.4 Supply Characteristics: Corrected supply rise rates units from V/s to V/µs.<br>• 37.6 Power Consumption: Added power consumption numbers for Device Variant C.<br>• 37.12.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C.<br>• Added 37.9 Injection Current section. |
| 40. Packaging Information | • Added 40.2.10 35 ball WLCSP (Device Variant C) package outline drawing. |
| Errata | • Added errata for Device Variant C. |
| 39. Electrical Characteristics at 125°C | • 39.8.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added characterization data for Device Variant C.<br>• 39.2 Absolute Maximum Ratings: Updated $V_{PIN}$ minimum and maximum values. (Related to the new Injection Current definition section) |

## 44.8 Rev. J – 07/2016

| | |
|---|---|
| 3. Ordering Information(1) | • SAM D21E: Added ATSAMD21E15B-UUT. |
| 30. TC – Timer/Counter | • 30.8.7 EVCTRL:EVACT[2:0] bit description updated: Time stamp capture and pulse width capture removed<br>. |
| 31. TCC – Timer/Counter for Control Applications | • 31.6.3 Additional Features: Removed "Time-Stamp Capture" section.<br>• 31.8.9 EVCTRL:EVACT0[2:0] bit description updated: "Capture Overflow times (Max value)" option removed (Related to Time-Stamp Capture). |
| Errata | • Cleaned up errata section: Split between device variant A and B. |

## 44.9 Rev. I – 03/2016

| | |
|---|---|
| 2. Configuration Summary | |
| | Updated value for Waveform output channels per TCC to 8/4/2. |
| 7. I/O Multiplexing and Considerations | |
| | Added Note.6 for Table 7-1 |
| 10. Memories | |
| | Table 10-1: Updated *start address* in *Internal RWW section* to from 0x00010000 to 0x00400000. |

| | |
|---|---|
| 22. NVMCTRL – Nonvolatile Memory Controller | |
| | Updated value from "NVM Base Address + 0x00010000" to "NVM Base Address + 0x00400000"in Figure 22-3 |
| 26. SERCOM USART | |
| | Updated equation and added error calculation explained w/example in section 26.6.2.6.4 Asynchronous Operational Range |
| 31. TCC – Timer/Counter for Control Applications: | |
| | Register Summary: Remove INTENCLR.SYNCRDY. Add MC0 (located in bit 0) for INTENSET and INTFLAG, and left shift MC1, MC2 and MC3 for one bit. Therefore, MC0/1/2/3 are located in bit 0/1/2/3. |
| 37. Electrical Characteristics | |
| | Updated unit from 's' to 'us' in the following tables:<br>• Table 39-10<br>• Table 39-11<br>• Table 39-23<br>• Table 39-24<br>• Table 39-37<br>• Table 39-38<br>• Table 39-39<br>• Table 39-40<br>• Table 39-45<br>• Table 39-46<br>• Table 39-47<br>• Table 39-48<br>Update value and condition for Table 39-39 and Table 39-40 |
| 40. Packaging Information | |
| | Updated section 40.2.8 32 pin QFN. |

## 44.10 Rev. H – 01/2016

| | |
|---|---|
| 20. DMAC – Direct Memory Access Controller | |
| | Updated bit description of the PRICTRL0.LVLPRIn [n=3..0]. |
| 22. NVMCTRL – Nonvolatile Memory Controller | |
| | Updated description in 22.6.4.3 NVM Write: Removed reference to default NWM CTRLB.MANW default value.<br>Updated reset value for CTRLB.MANW from 0 to 1. Note that this change is only applicable for Device Variant B. Device Variant A will continue to have MANW bit reset value 0. |

| | |
|---|---|
| | Updated reset value of the CTRLB register from 0x00000000 to 0x00000080.<br>Note that this change is change is only applicable for Device Variant B. Device Variant A will continue to have CTRLB register reset value 0x00000000. |
| 13. DSU - Device Service Unit | |
| | Bit CTRL.CRC is write-only. |
| 32. USB – Universal Serial Bus: | |
| | Register HSOFC.FLENCE description updated.<br>USB Device Registers - Common: Bit description of CTRLB.SPDCONF[1:0] updated. |
| 40. Packaging Information | |
| | Updated values in 40.1.1 Thermal Resistance Data.<br>Corrected junction temperature equations: $T_C$ updated to $T_J$.<br>Updated package drawing for 40.2.9 35 ball WLCSP (Device Variant B): GPC corrected from GJP to GJR. No other changes. |
| 41. Schematic Checklist | |
| | Added 41.1.1 Operation in Noisy Environment.<br>Updated section 41.7 Programming and Debug Ports.<br>Updated recommended pin connection in 41.7.2 10-pin JTAGICE3 Compatible Serial Wire Debug Interface: Pull-up resistor value on SWCLK pin changed from 10kΩ to 1kΩ.<br>VDDCORE decoupling capacitor value updated from 100nF to 1µF in 41.2.1 Power Supply Connections |

## 44.11 Rev. G – 09/2015

| | |
|---|---|
| 17. SYSCTRL – System Controller: | |
| | Updated description in 17.6.7.1.5 Drift Compensation. |
| 37. Electrical Characteristics: | |
| | Removed note from Table 37-51 and Table 37-52. |
| 40. Packaging Information: | |
| | Package drawing updated 40.2.6 45-ball WLCSP. |
| 39. Electrical Characteristics at 125°C: | |
| | Removed note from:Table 39-39 and Table 39-40.<br>Updated power consumption units in Table 39-7. |

## 44.12 Rev. F – 07/2015

| | |
|---|---|
| 3. Ordering Information(1) | |
| | Added ATSAMD21E15B-UUT and ATSAMD21E16B-UUT ordering codes (WLCSP35 package option). |
| 4. Block Diagram | |
| | Updated system block diagram. |
| 5. Pinout | |
| | Added pinout figure for 5.4.2 WLCSP35. |
| 9. Product Mapping | |
| | Updated Internal RWW section to start address from 0x00010000 to 0x00400000. |
| 33. ADC – Analog-to-Digital Converter | |
| | References to AREFA and AREFB replaced with VREFA and VREFB respectively. |
| 37. Electrical Characteristics | |
| | Added GPIO cluster note to '37.2 Absolute Maximum Ratings.<br>Added 37.15.5 I2S Timing.<br>Updated BOD33 characteristics.<br>Added characterization data for Device Variant B. |
| 40. Packaging Information | |
| | Updated $\Theta_{JC}$ value from 3.1 to 15.0 °C/W for 32-pin QFN package in 40.1.1 Thermal Resistance Data.<br>Added package drawing for 40.2.9 35 ball WLCSP (Device Variant B). |
| 41. Schematic Checklist | |
| | 41.2.1 Power Supply Connections:<br>VDDCORE decoupling capacitor value updated from 100nF to 1µF.<br>References to AREFA and AREFB replaced with VREFA and VREFB respectively. |
| 39. Electrical Characteristics at 125°C | |
| | Added 39.9.3 I2S Timing.<br>Updated BOD33 characteristics.<br>Added characterization data for Device Variant B. |

## 44.13 Rev. E – 02/2015

| | |
|---|---|
| 1. Description: | |
| | CoreMark score updated from 2.14 to 2.46 CoreMark/MHz. |
| 3. Ordering Information(1): | |
| | Added Ordering codes for Device Variant B.<br>Added 125°C ordering codes for QFN and TQFP package options: SAMD21E, SAM D21E, SAM D21G, and SAM D21J, and .<br>Added WLCSP package option for SAM D21G .<br>Added UFBGA package option for SAM D21J . |
| 5. Pinout: | |
| | Added pinout figures for 5.1.2 UFBGA64 and 5.2.2 WLCSP45. |
| 9. Product Mapping: | |
| | Updated Product Mapping figure with Internal RWW section block for Device Variant B. |
| 10. Memories: | |
| | 10.2 Physical Memory Map: Added start address for Internal Read While Write (RWW) section for Device Variant B. |
| 11. Processor And Architecture: | |
| | 11.1.1 Cortex M0+ Configuration: Removed green connection dots between DMAC Data and AHB-APB Bridge A and Bridge B. |
| 22. NVMCTRL – Nonvolatile Memory Controller: | |
| | Introducing Read While Write (RWW) feature for Device Variant B.<br>Updated and New sections:<br>22.1 Overview<br>22.2 Features<br>22.3 Block Diagram<br>22.6.4.1 NVM Read<br>22.6.4.2 RWWEE Read<br>22.6.4.3 NVM Write<br>22.6.4.5 Erase Row<br>22.6.2 Memory Organization: Figure 22-2 updated.<br>Register Summary and 22.8 Register Description: 22.8.3 PARAM: Added RWWEEP[12:0] bits for Device Variant B. |
| 23. PORT - I/O Pin Controller: | |
| | 23.6.3 I/O Pin Configuration: Removed reference to “open-drain”. |

| | |
|---|---|
| | Access for DRVSTR bit in Pin Configuration n register (PINCFGn.DRVCTR) updated from W to R/W. |
| 17. SYSCTRL – System Controller: | |
| | Removed references to XOSC32K and OSC32 1kHz clock output option:<br>- XOSC32K: 17.6.3 32kHz External Crystal Oscillator (XOSC32K) Operation<br>- OSC32K: 17.6.4 32kHz Internal Oscillator (OSC32K) Operation<br>1kHz Output Enable (EN1K) bit set as reserved bit:<br>- Bit 4 in 17.8.6 XOSC32K<br>- Bit 3 in 17.8.7 OSC32K |
| 37. Electrical Characteristics: | |
| | 37.10.3 Brown-Out Detectors Characteristics: Added Figure 37-3 and Figure 37-4 and updated conditions in Table 37-20 and Table 37-21. |
| 40. Packaging Information: | |
| | Added 40.2.3 64-ball UFBGA and 40.2.6 45-ball WLCSP package drawings. |
| 41. Schematic Checklist | |
| | Updated description in 41.6 Unused or Unconnected Pins. |
| Errata: | |
| | Device Variant A:<br>- Updated errata for revision A: Added Errata Reference 12291, 13507, 13574.<br>- Updated errata for revision B: Added Errata Reference 12291, 13507, 13574.<br>- Updated errata for revision C: Added Errata Reference 12291, 13507, 13574, 13951.<br>- Added errata for revision D.<br>Device Variant B:<br>- Added errata for revision E (Only available for SAMD21x15/16). |
| 39. Electrical Characteristics at 125°C: | |
| | Electrical characteristics for 125°C added. |

## 44.14 Rev. D – 09/2014

| | |
|---|---|
| 4. Block Diagram | |
| | NVM Controller bus connection changed from Master to Slave. |
| 14. Clock System | |
| | 14.3 Register Synchronization updated by splitting the section into 14.3.1 Common Synchronizer Register Synchronization and 14.3.2 Distributed Synchronizer Register Synchronization. |

| | |
|---|---|
| 37. Electrical Characteristics | |
| | 37.10.4 Analog-to-Digital (ADC) characteristics: Added note defining gain accuracy parameter in:<br>- ADC Differential Mode, Table 37-23<br>- ADC Single-Ended Mode, Table 37-25 |
| Errata | |
| | Updated errata for revision A, B and C: Added Errata Reference 13140, 12860. |

## 44.15 Rev. C – 07/2014

| | |
|---|---|
| 37. Electrical Characteristics | |
| | Updated condition for Rise time for both SDA and SCL ($t_R$) in High Speed Mode: $C_b$ changed from 1000pF to 100pF in Table 37-15. |
| Errata | |
| | Errata for revision C and E added. |

## 44.16 Rev. B – 07/2014

| |
|---|
| General: |
| Introduced the new product family name: Atmel \| SMART<br>Removed references to Clock Failure Detection.<br>Sub sections within chapters might been moved to other location within the chapter.<br>Typo corrections. |
| 2. Configuration Summary |
| Added 32KB Flash and 4KB SRAM options to SAM D21J and SAM D21G. |
| 3. Ordering Information(1) |
| Added Tray to Carrier Type option for SAM D21E, SAMD 21G and SAMD21J ordering codes. |
| 7. I/O Multiplexing and Considerations: |
| Updated REF function on PA03 and PA04 in Table 7-1:<br>PA03: DAC/VREFP changed to DAC/VREFA.<br>PA04: ADC VREFP changed to ADC/VREFB.<br>Updated COM function on PA30 and PA31:<br>PA30: CORTEX_M0P/SWCLK changed to SWCKL.<br>PA31: Added SWDIO. |
| 10. Memories |
| Added a second note to Table 10-2. |

Added Figure 10-1 Calibration and Auxiliary Space.

Added default values for fuses in Table 10-4 NVM User Row Mapping.

11. Processor And Architecture

MTB renamed from “Memory Trace Buffer” to “Micro Trace Buffer”.

13. DSU - Device Service Unit

Updated description of 13.11.3.1 Starting CRC32 Calculation.
Updated title of Table 13-6.

Added Device Selection table to Device Selection bit description the Device Identification register (DID.DEVSEL).

15. GCLK - Generic Clock Controller

Signal names updated in Device Clocking Diagram, 15.3 Block Diagram.

16. PM – Power Manager

Added figure Figure 16-2.
Register Summary:

Removed CFD bit from INTENCLR, INTENSET and INTFLAG.
Added PTC bit to APBCMASK register.

Register Description:

AHB Mask register (AHBMASK): Full bit names updated.
APBC Mask register (APBCMASK.PTC): Added PTC to bit 19.
CFD bit removed from INTENCLR, INTENSET and INTFLAG.

17. SYSCTRL – System Controller

Updated description of 17.6.6 8MHz Internal Oscillator (OSC8M) Operation.
FDPLL96M section reorganized and more integrated in the SYSCTRL chapter: Features, Signal Description and Product Dependencies sub sections removed and integrated with the corresponding sections in SYSCTRL.

Register Summary: Added VREG register on address 0x3C - 0x3D.

Register Description:

Updated reset values in OSC8M.

Updated CALIB[11:0] bit description in OSC8M.

Updated LBYPASS bit description in DPLLCTRLB.

18. WDT – Watchdog Timer

Updated description in 18.6.1 Principle of Operation: Introducing the bits used in Table 18-1.
Updated description in 18.6.2.1 Initialization.

Updated description in 18.6.2.4 Normal Mode.

Updated description in 18.6.2.5 Window Mode.

Updated description in 18.6.4 Interrupts.

| |
|---|
| WEN bit description in the Control register (CTRL.WEN) updated with information on enable-protection. |
| 19. RTC – Real-Time Counter |
| 19.6.9.1 Periodic Events: Bit names updated fro, PERx to PEREOx in example, Figure 19-4.<br>CLOCK.HOUR[4:0]: Updated Table 19-4<br>Mode 0 and Mode 2: CMPx bit renamed to CMP0 since only one CMP0 is available.<br>Bit description of CLOCK.HOUR[4:0]: Updated Table 19-4<br>ALARMn register renamed to ALARM0. |
| 20. DMAC – Direct Memory Access Controller |
| Updated block diagram, 20.3 Block Diagram.<br>General updated description. |
| 21. EIC – External Interrupt Controller |
| Register Summary and Register Description:<br>EVCTRL register: Added bits EXTINTO17 and EXTINTO16 in bit position 17 and 16 respectively.<br>INTENCLR, INTENSET, INTENFLAG registers: Added bits EXTINT17 and EXTINT16 in bit position 17 and 16 respectively.<br>WAKEUP register: Added bits WAKEUPEN17 and WAKEUPEN16 in bit position 17 and 16 respectively.<br>CONFIG2 register added, CONFIG0 and CONFIG1 registers updated: Added bits FILTEN0...31 and SENSE0...31. |
| 22. NVMCTRL – Nonvolatile Memory Controller |
| CTRLB register: Removed table from NVM Read Wait States description (RWS[3:0]) |
| 23. PORT - I/O Pin Controller |
| Instances of the term “pad” replaced with “pin”.<br>Instances of the term “bundle” replaced with “group” and “interface”.<br>23.6.2 Basic Operation description updated.<br>Peripheral Multiplexing n (PMUX0) register: Offset formula updated. |
| 24. EVSYS – Event System |
| Updated information in 24.2 Features.<br>24.5.2 Power Management updated: Description of on how event generators can generate an event when the system clock is stopped moved to 24.6.4 Sleep Mode Operation.<br>24.5.3 Clocks updated: Renamed EVSYS channel dedicated generic clock from GCLK_EVSYS_x to GCLK_EVSYS_CHANNELx.<br>Updated description in 24.6.1 Principle of Operation.<br>Updated description in sub sections of 24.6.2 Basic Operation.<br>Updated description in 24.6.3.1 The Overrun Channel n Interrupt.<br>Channel x Overrun bit description in 24.8.7 INTFLAG updated. |

| |
|---|
| 26. SERCOM USART |
| Updated description in 26.6.3.4 Break Character Detection and Auto-Baud.<br>Updated description in 26.6.3.7 Start-of-Frame Detection. |
| 29. I2S - Inter-IC Sound Controller |
| Introducing Frame Synch Clock.<br>29.4 Signal Description: Added separate tables for Master-, Slave- and Controller mode.<br>Updated description in 29.5.7 Debug Operation and 29.5.8 Register Access Protection.<br>Updated description in 29.6.1 Principle of Operation.<br>Updated description in sub sections of 29.6.2 Basic Operation.<br>Updated formula in 29.6.2.1.3 MCKn Clock Frequency.<br>Updated formulas in 29.6.2.1.5 Relation Between MCKn, SCKn, and Sampling Frequency fs.<br>Updated description in 29.6.6 PDM Reception.<br>Section on MONO removed and information included in 29.6.2 Basic Operation.<br>Updated property of Control A (CTRLA) register: Added Write-Synchronized |
| 31. TCC – Timer/Counter for Control Applications |
| Updated description in 31.6.1 Principle of Operation.<br>Updated description in sub sections of 31.6.2 Basic Operation.<br>Updated description in sub sections of 31.6.3 Additional Features.<br>Updated description in 31.6.6 Synchronization.<br>Lock Update (LUPD) bit description updated in Control B Clear (CTRLBCLR) register.<br>Compare Channel Buffer x Busy (CCBx) bit description updated in Synchronization Busy (SYNCBUSY) register.<br>Event Control (EVCTRL) register property updated: Removed Enable-Protected.<br>Interrupt Enable Clear (INTENCLR), Interrupt Enable Set (INTENSET) and Interrupt Flag Status and Clear (INTFLAG) registers: Updated bit description of FAULT0, FAULT1, FAULTA and FAULTB.<br>STATUS register bit descriptions updated.<br>Wave Control (WAVE) register property updated: Removed Read-Synchronized.<br>Pattern Buffer (PATTB) register: Updated property and bit description.<br>Waveform Control Buffer (WAVEB) register: Updated property and bit descriptions. |
| 32. USB – Universal Serial Bus |
| Removed figures: Setup Transaction Overview, OUT Single Bank Transaction Overview, IN Single Bank Transaction Overview and USB Host Communication Flow.<br>Updated description and graphics in sub sections of 32.6.2 USB Device Operations.<br>Updated description in sub sections of 32.6.3 Host Operations.<br>Pad Calibration (PADCAL) register: Access updated. |

Upgraded bit descriptions.

32.8.7.1 Pipe Descriptor Structure: Updated register reset values.

33. ADC – Analog-to-Digital Converter

Register Description:
REFCTRL bit selection names updated from AREFA / AREFB to VREFA / VREFB in Table 33-5

35. DAC – Digital-to-Analog Converter

Updated block diagram and signal description: VREFP replaced with VREFB.

37. Electrical Characteristics

Updated $V_{DD}$ max from 3.63V to 3.63V in 37.2 Absolute Maximum Ratings.
Updated VDDIN pin from 57 to 56 in 7.2.4 GPIO Clusters.

37.6 Power Consumption: Updated Max values for STANDBY from 190.6µA and 197.3µA to 100µA in Table 37-7.

Added 37.7 Peripheral Power Consumption.

37.8 I/O Pin Characteristics: tRISE and tFALL updated with different load conditions depending on the DVRSTR value in .

37.8 I/O Pin Characteristics: Correct typo IOL and IOH Max values inverted between PORT.PINCFG.DRVSTR=0 and 1, tRISE and tFALL updated with different load conditions depending on the DVRSTR value in Table 37-14.

37.10 Analog Characteristics: Removed note from Table 37-18.

37.10.4 Analog-to-Digital (ADC) characteristics: Added Max DC supply current ($I_{DD}$), $R_{SAMPLE}$ maximum value changed from 2.8kW to 3.5kW, Conversion time Typ value change to Min Value in Table 37-22.

37.10.5 Digital to Analog Converter (DAC) Characteristics: Added Max DC supply current ($I_{DD}$) in Table 37-30.

37.10.6 Analog Comparator Characteristics: Added Min and Max values for VSCALE INL, DNL, Offset Error and Gain Error in Table 37-34.

37.10.7 Internal 1.1V Bandgap Reference Characteristics: Added Min and Max values, removed accuracy row in Table 37-36.

37.15.3 SERCOM in I2C Mode Timing: Add Typical values for $t_R$ in Table 37-62.

Removed Asynchronous Watchdog Clock Characterization.

37.12.4 32.768kHz Internal oscillator (OSC32K) Characteristics: Added Max current consumption ($I_{OSC32K}$) in Table 37-53.

Updated Crystal Oscillator Characteristics (XOSC32K) ESR maximum values, 37.12.2.1.1 Crystal Oscillator Characteristics.

Updated Crystal Oscillator Characteristics (XOSC) ESR maximum value, 37.12.1.2 Crystal Oscillator Characteristics from 348kΩ to 141kΩ.

37.12.3 Digital Frequency Locked Loop (DFLL48M) Characteristics: Updated presentation, now separating between Open- and Closed Loop Modes. Added $f_{REF}$ Min and Max values to Table 37-50.

| |
|---|
| Updated typical Startup time ( $t_{STARTUP}$) from 6.1µs to 8µs in Table 37-51.<br>Updated typical Fine lock time ($t_{LFINE}$) from 700µs to 600µs in Table 37-51.<br>37.12.7 Fractional Digital Phase Locked Loop (FDPLL96M) Characteristics: Added Current consumption ($I_{FDPLL96M}$), Period Jitter (Jp), Lock time ($t_{LOCK}$), Duty cycles parameters in Table 37-56.<br>Added 37.14 USB Characteristics.<br>37.15 Timing Characteristics: Added SCK period ($t_{SCK}$) Typ value in Table 37-60. |
| Errata |
| Errata for revision B added. |

## 44.17 Rev. A - 02/2014

| |
|---|
| Initial released version of this data sheet. |

## The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## Customer Support

Users of Microchip products can receive assistance through several channels:

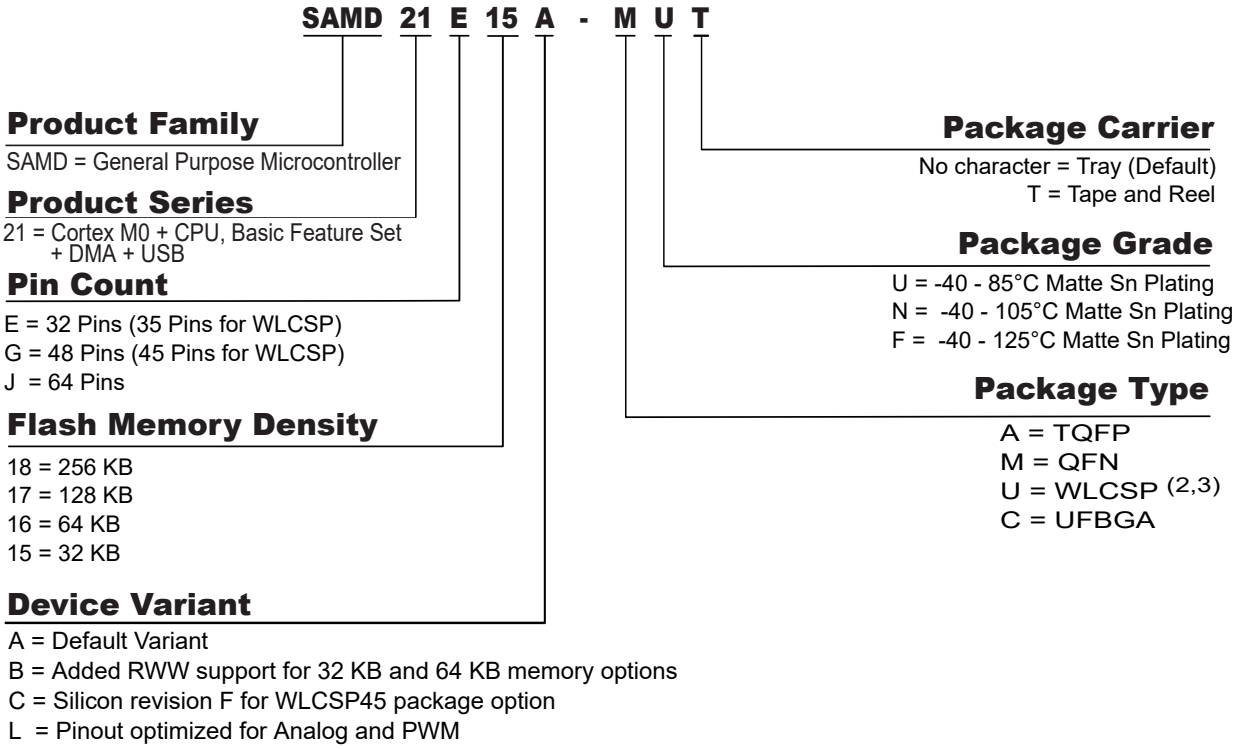
- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

## Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.




## Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

© 2018 Microchip Technology Inc.

## Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2018, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-2944-9

## Quality Management System Certified by DNV

**ISO/TS 16949**

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

# Worldwide Sales and Service

| AMERICAS | ASIA/PACIFIC | ASIA/PACIFIC | EUROPE |
|---|---|---|---|
| **Corporate Office**<br>2355 West Chandler Blvd.<br>Chandler, AZ 85224-6199<br>Tel: 480-792-7200<br>Fax: 480-792-7277<br>Technical Support:<br>http://www.microchip.com/support<br>Web Address:<br>www.microchip.com<br>**Atlanta**<br>Duluth, GA<br>Tel: 678-957-9614<br>Fax: 678-957-1455<br>**Austin, TX**<br>Tel: 512-257-3370<br>**Boston**<br>Westborough, MA<br>Tel: 774-760-0087<br>Fax: 774-760-0088<br>**Chicago**<br>Itasca, IL<br>Tel: 630-285-0071<br>Fax: 630-285-0075<br>**Dallas**<br>Addison, TX<br>Tel: 972-818-7423<br>Fax: 972-818-2924<br>**Detroit**<br>Novi, MI<br>Tel: 248-848-4000<br>**Houston, TX**<br>Tel: 281-894-5983<br>**Indianapolis**<br>Noblesville, IN<br>Tel: 317-773-8323<br>Fax: 317-773-5453<br>Tel: 317-536-2380<br>**Los Angeles**<br>Mission Viejo, CA<br>Tel: 949-462-9523<br>Fax: 949-462-9608<br>Tel: 951-273-7800<br>**Raleigh, NC**<br>Tel: 919-844-7510<br>**New York, NY**<br>Tel: 631-435-6000<br>**San Jose, CA**<br>Tel: 408-735-9110<br>Tel: 408-436-4270<br>**Canada - Toronto**<br>Tel: 905-695-1980<br>Fax: 905-695-2078 | **Australia - Sydney**<br>Tel: 61-2-9868-6733<br>**China - Beijing**<br>Tel: 86-10-8569-7000<br>**China - Chengdu**<br>Tel: 86-28-8665-5511<br>**China - Chongqing**<br>Tel: 86-23-8980-9588<br>**China - Dongguan**<br>Tel: 86-769-8702-9880<br>**China - Guangzhou**<br>Tel: 86-20-8755-8029<br>**China - Hangzhou**<br>Tel: 86-571-8792-8115<br>**China - Hong Kong SAR**<br>Tel: 852-2943-5100<br>**China - Nanjing**<br>Tel: 86-25-8473-2460<br>**China - Qingdao**<br>Tel: 86-532-8502-7355<br>**China - Shanghai**<br>Tel: 86-21-3326-8000<br>**China - Shenyang**<br>Tel: 86-24-2334-2829<br>**China - Shenzhen**<br>Tel: 86-755-8864-2200<br>**China - Suzhou**<br>Tel: 86-186-6233-1526<br>**China - Wuhan**<br>Tel: 86-27-5980-5300<br>**China - Xian**<br>Tel: 86-29-8833-7252<br>**China - Xiamen**<br>Tel: 86-592-2388138<br>**China - Zhuhai**<br>Tel: 86-756-3210040 | **India - Bangalore**<br>Tel: 91-80-3090-4444<br>**India - New Delhi**<br>Tel: 91-11-4160-8631<br>**India - Pune**<br>Tel: 91-20-4121-0141<br>**Japan - Osaka**<br>Tel: 81-6-6152-7160<br>**Japan - Tokyo**<br>Tel: 81-3-6880- 3770<br>**Korea - Daegu**<br>Tel: 82-53-744-4301<br>**Korea - Seoul**<br>Tel: 82-2-554-7200<br>**Malaysia - Kuala Lumpur**<br>Tel: 60-3-7651-7906<br>**Malaysia - Penang**<br>Tel: 60-4-227-8870<br>**Philippines - Manila**<br>Tel: 63-2-634-9065<br>**Singapore**<br>Tel: 65-6334-8870<br>**Taiwan - Hsin Chu**<br>Tel: 886-3-577-8366<br>**Taiwan - Kaohsiung**<br>Tel: 886-7-213-7830<br>**Taiwan - Taipei**<br>Tel: 886-2-2508-8600<br>**Thailand - Bangkok**<br>Tel: 66-2-694-1351<br>**Vietnam - Ho Chi Minh**<br>Tel: 84-28-5448-2100 | **Austria - Wels**<br>Tel: 43-7242-2244-39<br>Fax: 43-7242-2244-393<br>**Denmark - Copenhagen**<br>Tel: 45-4450-2828<br>Fax: 45-4485-2829<br>**Finland - Espoo**<br>Tel: 358-9-4520-820<br>**France - Paris**<br>Tel: 33-1-69-53-63-20<br>Fax: 33-1-69-30-90-79<br>**Germany - Garching**<br>Tel: 49-8931-9700<br>**Germany - Haan**<br>Tel: 49-2129-3766400<br>**Germany - Heilbronn**<br>Tel: 49-7131-67-3636<br>**Germany - Karlsruhe**<br>Tel: 49-721-625370<br>**Germany - Munich**<br>Tel: 49-89-627-144-0<br>Fax: 49-89-627-144-44<br>**Germany - Rosenheim**<br>Tel: 49-8031-354-560<br>**Israel - Ra'anana**<br>Tel: 972-9-744-7705<br>**Italy - Milan**<br>Tel: 39-0331-742611<br>Fax: 39-0331-466781<br>**Italy - Padova**<br>Tel: 39-049-7625286<br>**Netherlands - Drunen**<br>Tel: 31-416-690399<br>Fax: 31-416-690340<br>**Norway - Trondheim**<br>Tel: 47-7289-7561<br>**Poland - Warsaw**<br>Tel: 48-22-3325737<br>**Romania - Bucharest**<br>Tel: 40-21-407-87-50<br>**Spain - Madrid**<br>Tel: 34-91-708-08-90<br>Fax: 34-91-708-08-91<br>**Sweden - Gothenberg**<br>Tel: 46-31-704-60-40<br>**Sweden - Stockholm**<br>Tel: 46-8-5090-4654<br>**UK - Wokingham**<br>Tel: 44-118-921-5800<br>Fax: 44-118-921-5820 |