To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

User's Manual

8

7641 Group

User's Manual

RENESAS 8-BIT SINGLE-CHIP
MICROCOMPUTER
740 FAMILY / 7600 SERIES

Renesas Electronics
www.renesas.com

Rev.2.00 2006.08

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

BEFORE USING THIS MANUAL

This user's manual consists of the following three chapters. Refer to the chapter appropriate to your conditions, such as hardware design or software development. Chapter 3 also includes necessary information for systems development. You must refer to that chapter.

1. Organization

- **CHAPTER 1 HARDWARE**
  This chapter describes features of the microcomputer and operation of each peripheral function.

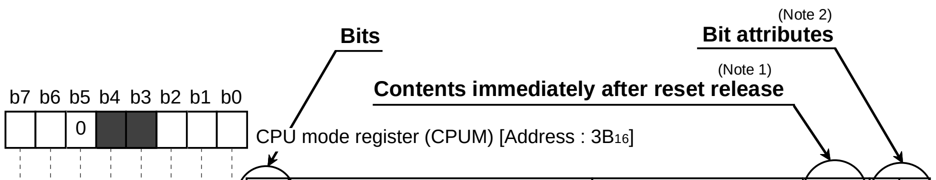
- **CHAPTER 2 APPLICATION**
  This chapter describes usage and application examples of peripheral functions, based mainly on setting examples of relevant registers.

- **CHAPTER 3 APPENDIX**
  This chapter includes necessary information for systems development using the microcomputer, such as the electrical characteristics, the notes, and the list of registers.

  ✻For the mask ROM confirmation form, the ROM programming confirmation form, and the mark specifications, refer to the "Renesas Technology" Homepage (http://www.renesas.com/en/rom).

2. Structure of register

The figure of each register structure describes its functions, contents at reset, and attributes as follows :



B	Name	Function	At reset	R	W
0	Processor mode bits	b1 b0 0 0 : Single-chip mode 0 1 : 1 0 : Not available 1 1 :	0	○	○
1			0	○	○
2	Stack page selection bit	0 : 0 page 1 : 1 page	0	○	○
3	Nothing arranged for these bits. These are write disabled bits. When these bits are read out, the contents are "0."		0	○	×
4			0	○	×
5	Fix this bit to "0."		1	○	○
6	Main clock (X_{IN}-X_{OUT}) stop bit	0 : Operating 1 : Stopped	✻	○	○
7	Internal system clock selection bit	0 : X_{IN}-X_{OUT} selected 1 : X_{CIN}-X_{COUT} selected	✻	○	○

■ : Bit in which nothing is arranged　　▒ : Bit that is not used for control of the corresponding function

Note 1:. Contents immediately after reset release
0....... "0" at reset release
1....... "1" at reset release
?....... Undefined at reset release
✻.......Contents determined by option at reset release

Note 2: Bit attributes.........The attributes of control register bits are classified into 3 bytes : read-only, write-only and read and write. In the figure, these attributes are represented as follows :

R.......Read　　W......Write
○...... Read enabled　　○ ..... Write enabled
×.......Read disabled　　×...... Write disabled

3. Supplementation

For details of development support tools, refer to the "Renesas Technology" Homepage (http://www.renesas.com).

Table of contents

CHAPTER 1 HARDWARE

CHAPTER 2 APPLICATION

CHAPTER 3 APPENDIX

List of figures

CHAPTER 1 HARDWARE

CHAPTER 2 APPLICATION

CHAPTER 3 APPENDIX

List of tables

CHAPTER 1 HARDWARE

CHAPTER 2 APPLICATION

CHAPTER 3 APPENDIX

CHAPTER 1

HARDWARE

DESCRIPTION

The 7641 group is the 8-bit microcomputer based on the 7600 series core (740 family core compatible) technology.

The 7641 group is designed for PC peripheral devices, including the USB, DMAC, Serial I/O, UART, Timer, Master CPU bus interface and so on.

FEATURES

<Microcomputer mode>

- Basic machine-language instructions 71
- Minimum instruction execution time .... 83 ns
  (at 24 MHz oscillation frequency)
- Memory size
  ROM .... 32 Kbytes
  RAM .... 1 Kbytes
- Programmable input/output ports 66
- Software pull-up resistors .... Built-in
- Interrupts .... 24 sources, 24 vectors
  (external 5 including Key input, internal 18, software 1)
- USB function control unit
  Transceiver .... Full-Speed USB2.0 specification
- Timers .... 16-bit ✕ 2 (Timers X, Y)
  8-bit ✕ 3 (Timers 1, 2, 3)
- Serial Interface
  Serial I/O .... 8-bit ✕ 1
  UART .... 8-bit ✕ 2
- DMAC .... 2 channels
- Master CPU bus interface .... 2 bytes
- Special count source generator .... 8-bit ✕ 1
- Clock generating circuit .... Built-in
  (connect to external ceramic resonator or quartz-crystal oscillator)
- Power source voltage
  At 24 MHz oscillation frequency, ϕ = 12 MHz .... 4.15 to 5.25 V
  At 24 MHz oscillation frequency, ϕ = 6 MHz .... 3.00 to 3.60 V
- Operating temperature range .... –20 to 70°C
- Packages
  FP .... PRQP0080GB-A (80-pin QFP)
  HP .... PLQP0080KB-A (80-pin LQFP)

<Flash memory mode>

- Power source voltage
  At 24 MHz oscillation frequency, ϕ = 12 MHz .... 4.15 to 5.25 V
  At 24 MHz oscillation frequency, ϕ = 6 MHz .... 3.00 to 3.60 V
- Program/Erase voltage
  .... V_{CC} = 4.50 V to 5.25 V, or 3.00 V to 3.60 V
  .... V_{PP} = 4.50 V to 5.25 V
  At 24 MHz oscillation frequency, ϕ = 6 MHz (See Table 10.)
- Memory size
  Flash ROM .... 32 Kbytes
  RAM .... 2.5 Kbytes
- Flash memory mode .... 3 modes
  Parallel I/O mode
  Standard serial I/O mode
  CPU rewrite mode
- Programming method .... Programming in unit of byte
- Erasing method
  Batch erasing
  Block erasing
- Program/Erase control by software command
- Command number .... 6 commands
- Number of times for programming/erasing 100
- ROM code protection
  Available in parallel I/O mode and standard serial I/O mode
- Operating temperature range (at programming/erasing) ....
  .... Normal temperature

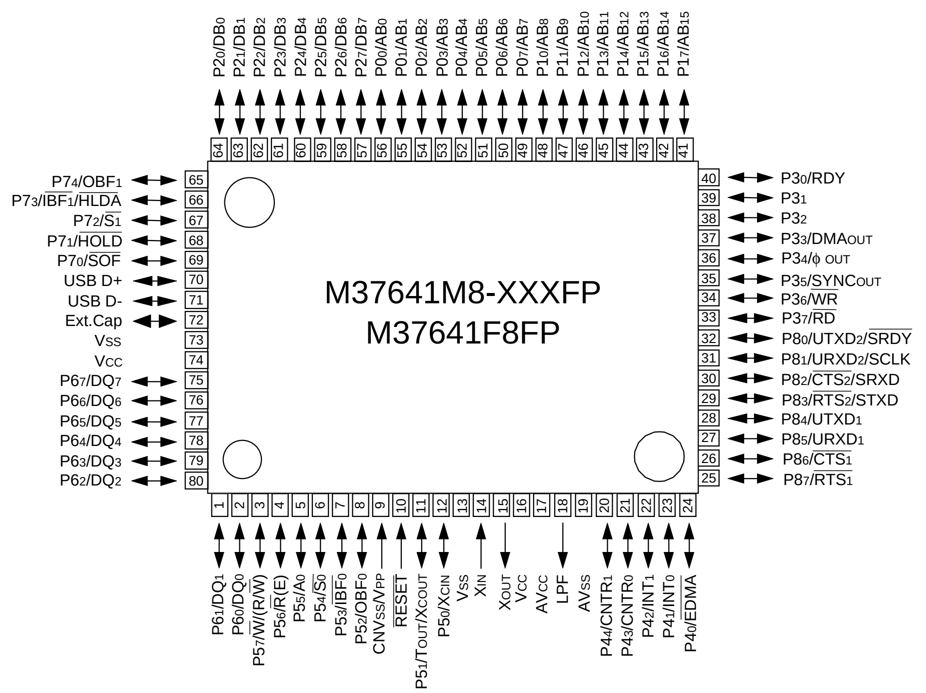
APPLICATION

Audio, musical instrument, printer, scanner, modem, other PC peripheral devices

■Notes

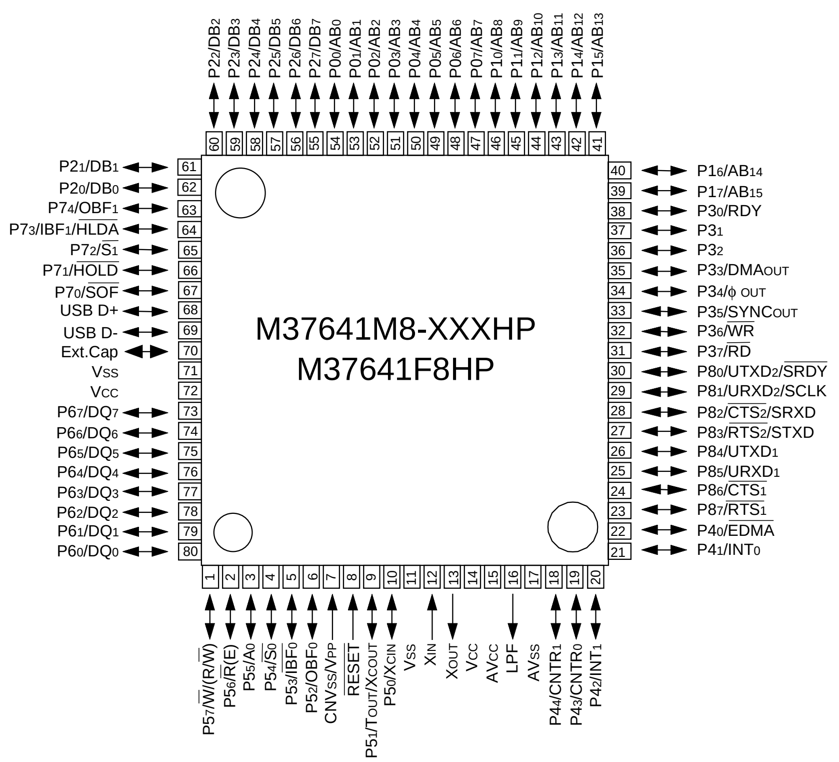
The flash memory version cannot be used for application embedded in the MCU card.

PIN CONFIGURATION (TOP VIEW)



Package type : PRQP0080GB-A (80P6N-A)

Fig. 1 M37641M8-XXXFP, M37641F8FP pin configuration



Package type : PLQP0080KB-A (80P6Q-A)

Fig. 2 M37641M8-XXXHP, M37641F8HP pin configuration

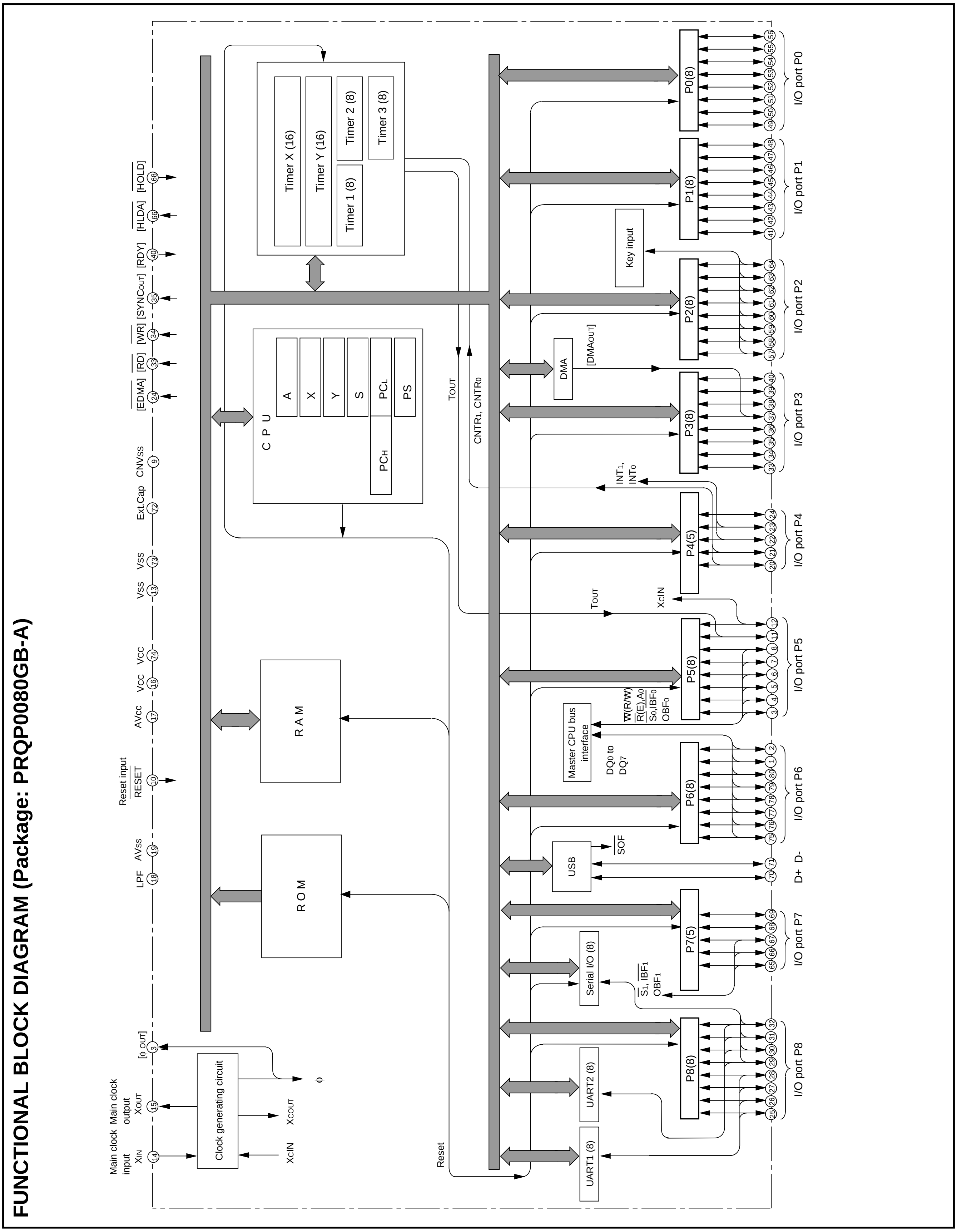


Fig. 3 Functional block diagram

PIN DESCRIPTION

Table 1 Pin description (1)

Pin	Name	Function	Function except a port function
V_{CC}, V_{SS}	Power source	• Apply 4.15 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the Vcc pin. Apply 0 V to the Vss pin.	
CNVss/V_{PP}	CNVss	• This controls the MCU operating mode. Connect this pin to Vss. If connecting this pin to Vcc, the internal ROM is inhibited. In the flash memory version this pin functions as a V_{PP} power supply input pin.	
AVss/AVcc	Analog power supply	• These pins are the power supply inputs for analog circuitry.	
$\overline{RESET}$	Reset input	• Reset input pin for active “L.”	
X_{IN}	Clock input	• Connect a ceramic resonator or a quartz-crystal oscillator between the X_{IN} and X_{OUT} pins to set the oscillation frequency. • If an external clock is used, connect the clock source to the X_{IN} pin and leave the X_{OUT} pin open.	
X_{OUT}	Clock output		
LPF	LPF	• Loop filter for the frequency synthesizer.	
Ext. Cap.	3.3 V line power supply	• It is a capacitor connection pin for built-in DC-DC converter. At Vcc=5 V, use built-in DC-DC converter by permitting a USB line driver and connect a capacitor. Refer to "Notes on use" for details. Built-in DC-DC converter cannot be used at Vcc = 3.3 V. Supply 3.3V power supply to this pin from the externals.	
USB D+	USB D+	• USB D+ voltage signal port. Connect a 27 to 33 Ω (recommended) resistor in series.	
USB D-	USB D-	• USB D- voltage signal port. Connect a 27 to 33 Ω (recommended) resistor in series.	
$P0_0/AB_0$–$P0_7/AB_7$		• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the address bus.	
$P1_0/AB_8$–$P1_7/AB_{15}$	I/O port P0	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the address bus.	
$P2_0/DB_0$–$P2_7/DB_7$	I/O port P1	• 8-bit I/O port. • CMOS compatible input level or VIHL input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the data bus.	• Key-on wake-up interrupt input pin
$P3_0$/RDY, $P3_1$, $P3_2$, $P3_3/DMA_{OUT}$, $P3_4/\phi_{OUT}$, $P3_5/SYNC_{OUT}$, $P3_6/\overline{WR}$, $P3_7/\overline{RD}$	I/O port P2 I/O port P3	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the control bus.	• External memory control pin
$P4_0/\overline{EDMA}$, $P4_1/INT_0$, $P4_2/INT_1$, $P4_3/CNTR_0$, $P4_4/CNTR_1$	(See Remarks.)	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the control bus.	• External memory control pin • External interrupt pin • Timer X, Timer Y pin
$P5_0/X_{CIN}$, $P5_1/T_{OUT}/X_{COUT}$, $P5_2/OBF_0$, $P5_3/IBF_0$, $P5_4/S_0$, $P5_5/A_0$, $P5_6/\overline{R}(E)$, $P5_7/\overline{W}(R/\overline{W})$	I/O port P4	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When enabling the Master CPU bus interface function, CMOS or TTL input level can be selected as an input.	• Sub-clock generating input pin • Timers 1, 2 pulse output pins • Sub-clock generating output pin • Master CPU bus interface pin

Table 2 Pin description (2)

Pin	Name	Function	Function except a port function
$P6_0/DQ_0$– $P6_7/DQ_7$	I/O port P5	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When enabling the bus interface function, CMOS or TTL input level can be selected as its input.	• Master CPU bus interface pin
$P7_0/\overline{SOF}$, $P7_1/\overline{HOLD}$, $P7_2/\overline{S_1}$, $P7_3/\overline{IBF_1}/\overline{HLDA}$, $P7_4/OBF_1$	I/O port P6	• 5-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output.	• USB function pin • Master CPU bus interface pin
$P8_0/UTXD_2/\overline{SRDY}$, $P8_1/URXD_2/SCLK$, $P8_2/\overline{CTS_2}/SRXD$, $P8_3/\overline{RTS_2}/STXD$,	I/O port P7 I/O port P8	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output.	• Serial I/O pin • UART2 pin
$P8_4/UTXD_1$, $P8_5/URXD_1$, $P8_6/\overline{CTS_1}$, $P8_7/\overline{RTS_1}$			• UART1 pin

Remarks

•DMA_{OUT} pin
If externally detecting the timing of DMA execution, use the signal from this pin. It is "H" level during DMA transferring. This signal is valid in the memory expansion and microprocessor modes.

•$SYNC_{OUT}$ pin
If externally detecting the timing of OP code fetch, use the signal from this pin. This signal is valid in the memory expansion and microprocessor modes.

PART NUMBERING

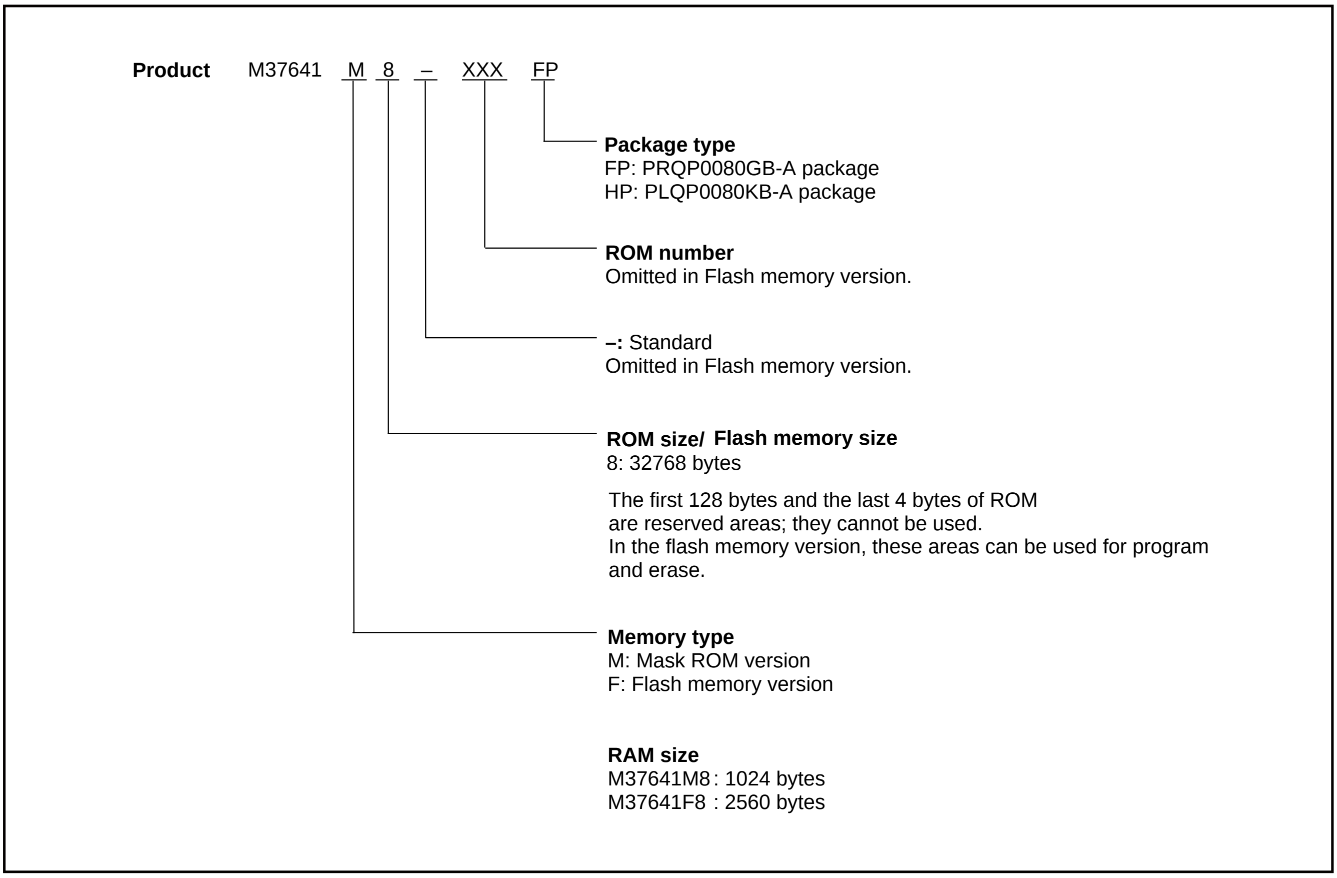


Fig. 4 Part numbering

GROUP EXPANSION

Mitsubishi plans to expand the 7641 group as follows.

Memory Type

Supports for mask ROM and flash memory versions.

Memory Size

ROM size .... 32 Kbytes
RAM size .... 1024 to 2560 bytes

Packages

PRQP0080GB-A .... 0.8 mm-pitch plastic molded QFP
PLQP0080KB-A .... 0.5 mm-pitch plastic molded LQFP

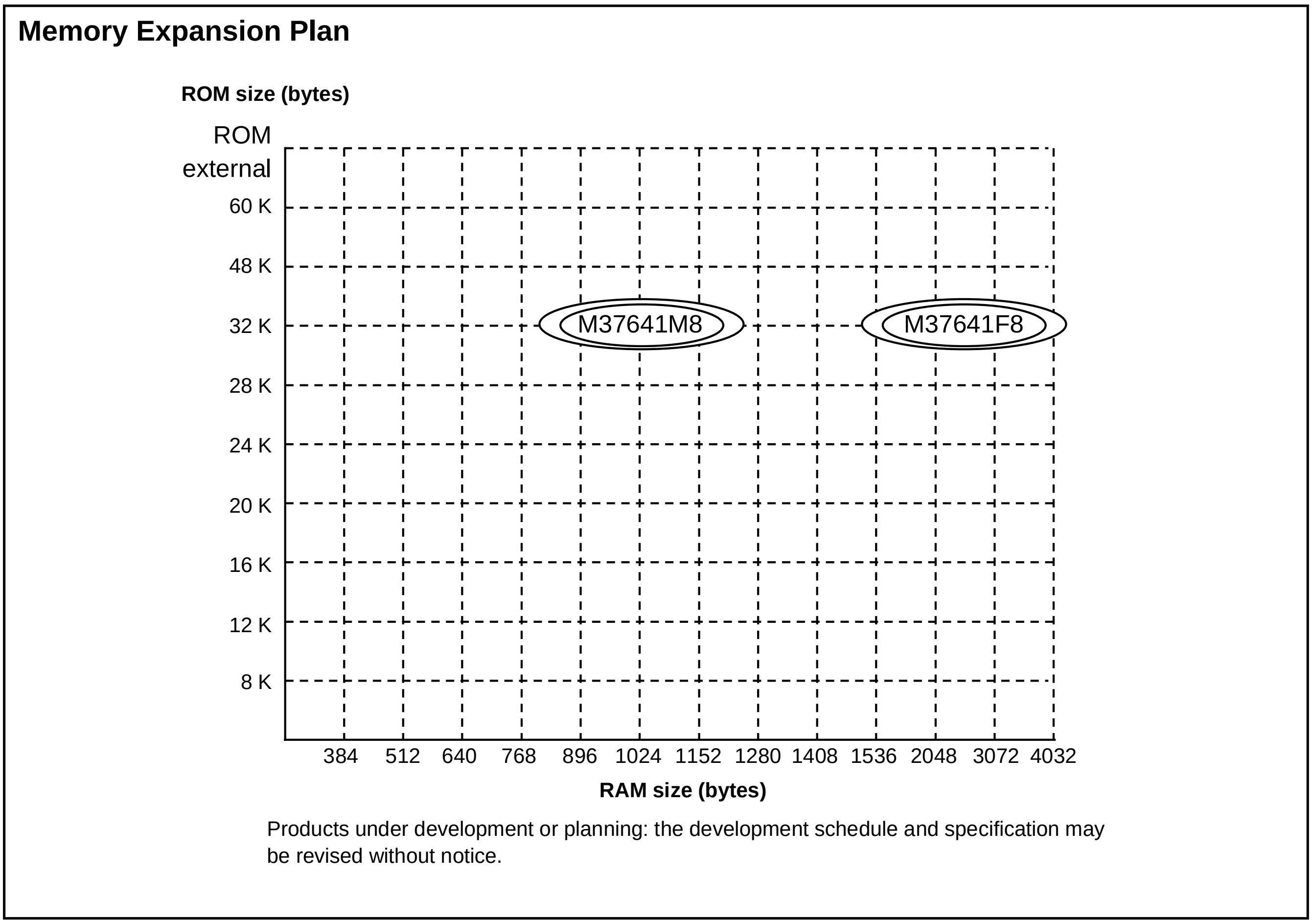


Fig. 5 Memory expansion plan

Currently planning products are listed below.

Table 3 Support products

As of Aug. 2006

Product name	ROM size (bytes) ROM size for User in ( )	RAM size (bytes)	Package	Remarks
M37641M8-XXXFP	32768 (32636)	1024	PRQP0080GB-A	Mask ROM version
M37641M8-XXXHP			PLQP0080KB-A	
M37641F8FP	32768	2560	PRQP0080GB-A	Flash memory version
M37641F8HP			PLQP0080KB-A	

FUNCTIONAL DESCRIPTION
CENTRAL PROCESSING UNIT (CPU)

The 7641 group uses the standard 7600 series instruction set. Refer to the 7600 Series Software Manual for details on the instruction set. The 7600 series has an upward compatible instruction set, of which instruction execution cycles are shortened, for 740 series.

[Accumulator (A)]

The accumulator is an 8-bit register. Data operations such as data transfer, etc., are executed mainly through the accumulator.

[Index Register X (X)]

The index register X is an 8-bit register. In the index addressing modes, the value of the OPERAND is added to the contents of register X and specifies the real address.

[Index Register Y (Y)]

The index register Y is an 8-bit register. In partial instruction, the value of the OPERAND is added to the contents of register Y and specifies the real address.

[Stack Pointer (S)]

The stack pointer is an 8-bit register used during subroutine calls and interrupts. This register indicates start address of stored area (stack) for storing registers during subroutine calls and interrupts.
The low-order 8 bits of the stack address are determined by the contents of the stack pointer. The high-order 8 bits of the stack address are determined by the stack page selection bit. If the stack page selection bit is “0” , the high-order 8 bits becomes “00_{16}”. If the stack page selection bit is “1”, the high-order 8 bits becomes “01_{16}”.
The operations of pushing register contents onto the stack and popping them from the stack are shown in Figure 7.
Store registers other than those described in Figure 7 with program when the user needs them during interrupts or subroutine calls.

[Program Counter (PC)]

The program counter is a 16-bit counter consisting of two 8-bit registers PC_H and PC_L. It is used to indicate the address of the next instruction to be executed.

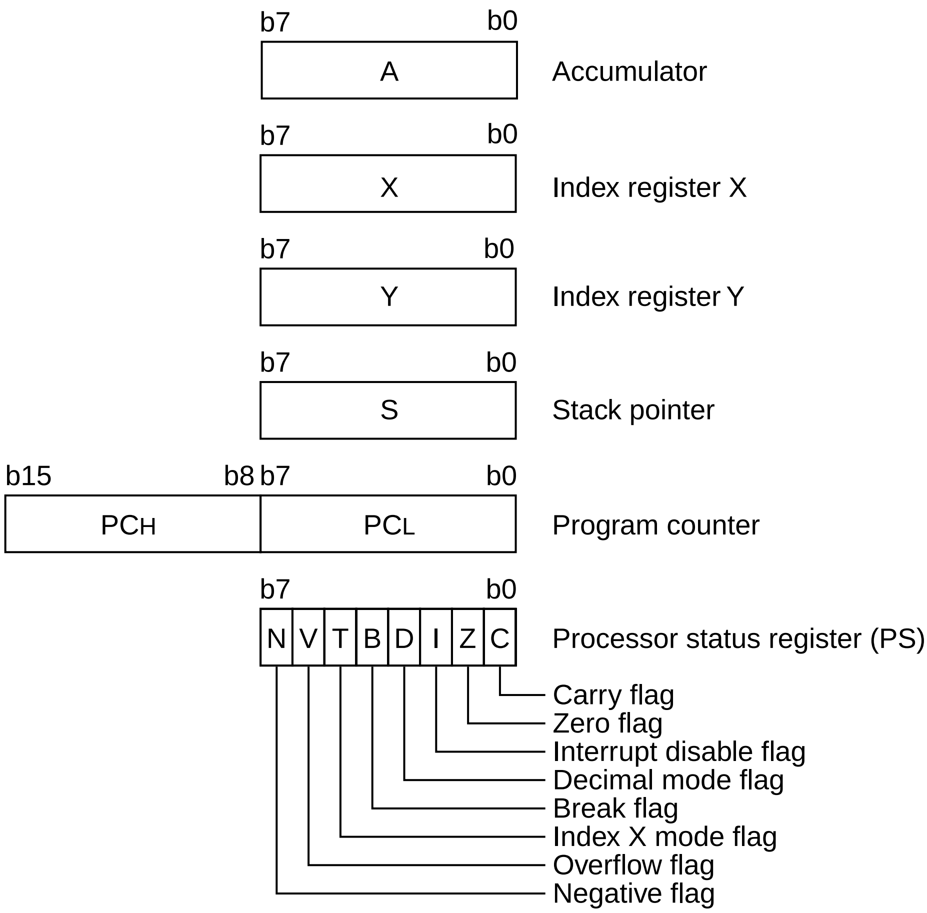


Fig. 6 7600 series CPU register structure

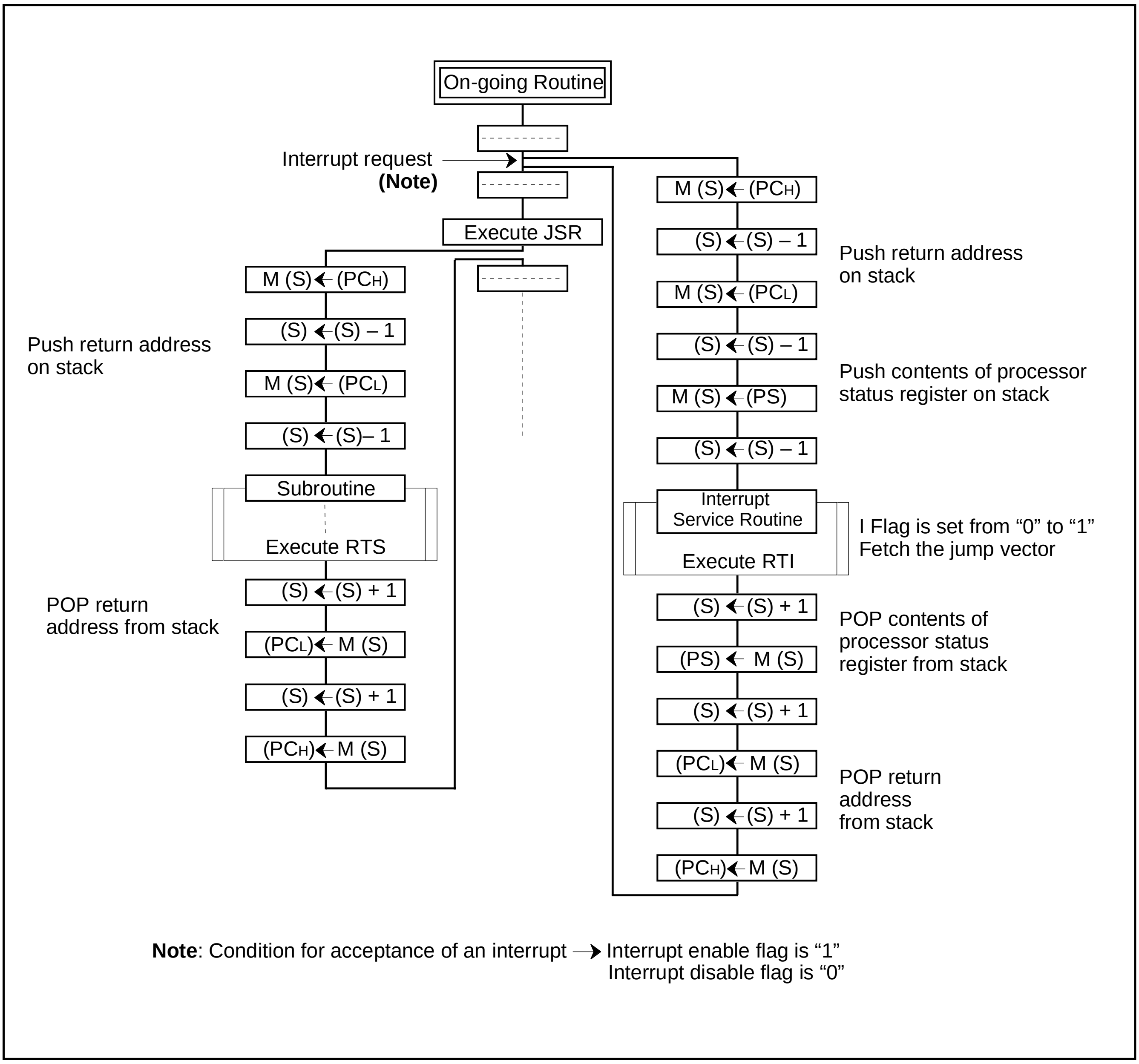


Fig. 7 Register push and pop at interrupt generation and subroutine call

Table 4 Push and pop instructions of accumulator or processor status register

	Push instruction to stack	Pop instruction from stack
Accumulator	PHA	PLA
Processor status register	PHP	PLP

[Processor status register (PS)]

The processor status register is an 8-bit register consisting of 5 flags which indicate the status of the processor after an arithmetic operation and 3 flags which decide MCU operation. Branch operations can be performed by testing the Carry (C) flag , Zero (Z) flag, Overflow (V) flag, or the Negative (N) flag. In decimal mode, the Z, V, N flags are not valid.

•Bit 0: Carry flag (C)
The C flag contains a carry or borrow generated by the arithmetic logic unit (ALU) immediately after an arithmetic operation. It can also be changed by a shift or rotate instruction.

•Bit 1: Zero flag (Z)
The Z flag is set if the result of an immediate arithmetic operation or a data transfer is “0”, and cleared if the result is anything other than “0”.

•Bit 2: Interrupt disable flag (I)
The I flag disables all interrupts except for the interrupt generated by the BRK instruction.
Interrupts are disabled when the I flag is “1”.

•Bit 3: Decimal mode flag (D)
The D flag determines whether additions and subtractions are executed in binary or decimal. Binary arithmetic is executed when this flag is “0”; decimal arithmetic is executed when it is “1”. Decimal correction is automatic in decimal mode. Only the ADC

•Bit 4: Break flag (B)
The B flag is used to indicate that the current interrupt was generated by the BRK instruction. The BRK flag in the processor status register is always “0”. When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set to “1”.

•Bit 5: Index X mode flag (T)
When the T flag is “0”, arithmetic operations are performed between accumulator and memory. When the T flag is “1”, direct arithmetic operations and direct data transfers are enabled between memory locations.

•Bit 6: Overflow flag (V)
The V flag is used during the addition or subtraction of one byte of signed data. It is set if the result exceeds +127 to -128. When the BIT instruction is executed, bit 6 of the memory location operated on by the BIT instruction is stored in the overflow flag.

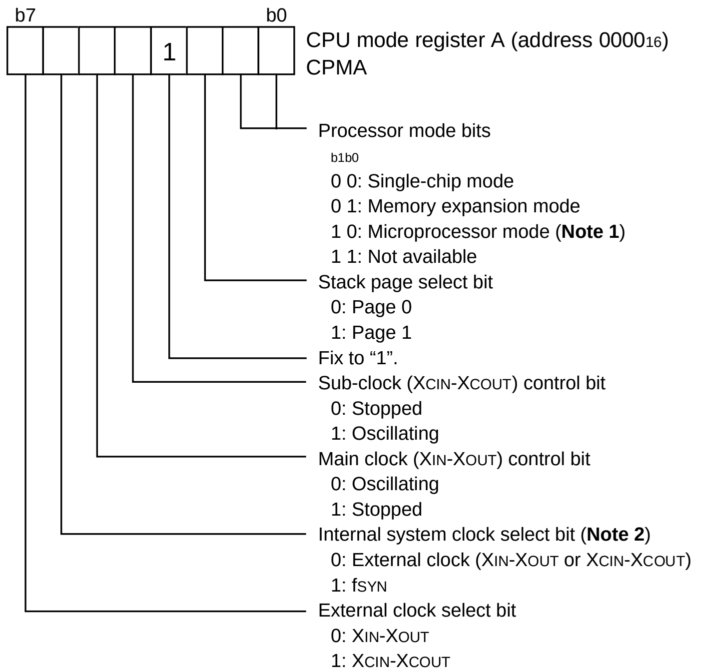
•Bit 7: Negative flag (N)
The N flag is set if the result of an arithmetic operation or data transfer is negative. When the BIT instruction is executed, bit 7 of the memory location operated on by the BIT instruction is stored in the negative flag.

Table 5 Set and clear instructions of each bit of processor status register

	C flag	Z flag	I flag	D flag	B flag	T flag	V flag	N flag
Set instruction	SEC	–	SEI	SED	–	SET	–	–
Clear instruction	CLC	–	CLI	CLD	–	CLT	CLV	–

[CPU Mode Registers A, B (CPUMA, CPUMB)] 0000_{16}, 0001_{16}
The CPU mode register contains the stack page select bit and the CPU operating mode select bit and so on.
The CPU mode registers are allocated at address 0000_{16}, 0001_{16}.

■ Notes
Do not use the microprocessor mode in the flash memory version.



Notes 1: This is not available in the flash memory version.
2: When (CPMA 6, 7) = (0, 0), the internal system clock can be selected between f(XIN) or f(XIN)/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

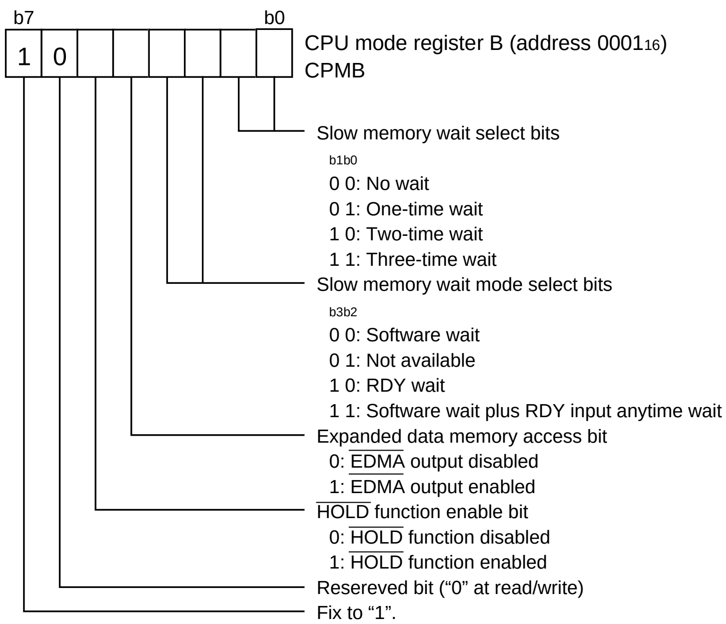


Fig. 8 Structure of CPU mode register

MEMORY

Special Function Register (SFR) Area

The Special Function Register area in the zero page contains control registers such as I/O ports and timers.

RAM

RAM is used for data storage and for stack area of subroutine calls and interrupts.

ROM

The first 128 bytes and the last 4 bytes of ROM are reserved for device testing and the rest is user area for storing programs. In the flash memory version, program and erase can be performed in the reserved area.

Interrupt Vector Area

The interrupt vector area contains reset and interrupt vectors.

Zero Page

Access to this area with only 2 bytes is possible in the zero page addressing mode.

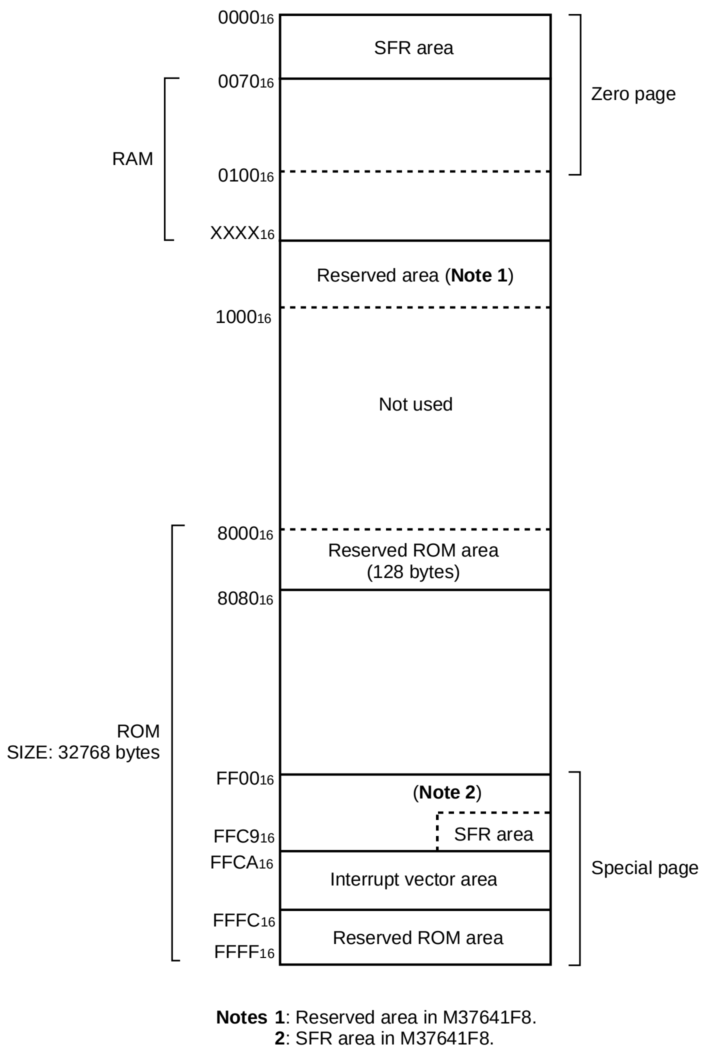
Special Page

Access to this area with only 2 bytes is possible in the special page addressing mode.

Refer to page 74 for the memory map of memory expansion and microprocessor modes.

RAM area

	RAM size (bytes)	Address $XXXX_{16}$
M37641M8	1024	$046F_{16}$
M37641F8	2560	$0A6F_{16}$



Notes 1: Reserved area in M37641F8.
2: SFR area in M37641F8.

Fig. 9 Memory map diagram

0000_{16}	CPU mode register A (CPUA)	0038_{16}	UART2 mode register (U2MOD)
0001_{16}	CPU mode register B (CPUB)	0039_{16}	UART2 baud rate generator (U2BRG)
0002_{16}	Interrupt request register A (IREQA)	$003A_{16}$	UART2 status register (U2STS)
0003_{16}	Interrupt request register B (IREQB)	$003B_{16}$	UART2 control register (U2CON)
0004_{16}	Interrupt request register C (IREQC)	$003C_{16}$	UART2 transmit/receive buffer register 1 (U2TRB1)
0005_{16}	Interrupt control register A (ICONA)	$003D_{16}$	UART2 transmit/receive buffer register 2 (U2TRB2)
0006_{16}	Interrupt control register B (ICONB)	$003E_{16}$	UART2 RTS control register (U2RTSC)
0007_{16}	Interrupt control register C (ICONC)	$003F_{16}$	DMAC index and status register (DMAIS)
0008_{16}	Port P0 (P0)	0040_{16}	DMAC channel x mode register 1 (DMAx1)
0009_{16}	Port P0 direction register (P0D)	0041_{16}	DMAC channel x mode register 2 (DMAx2)
$000A_{16}$	Port P1 (P1)	0042_{16}	DMAC channel x source register Low (DMAxSL)
$000B_{16}$	Port P1 direction register (P1D)	0043_{16}	DMAC channel x source register High (DMAxSH)
$000C_{16}$	Port P2 (P2)	0044_{16}	DMAC channel x destination register Low (DMAxDL)
$000D_{16}$	Port P2 direction register (P2D)	0045_{16}	DMAC channel x destination register High (DMAxDH)
$000E_{16}$	Port P3 (P3)	0046_{16}	DMAC channel x transfer count register Low (DMAxCL)
$000F_{16}$	Port P3 direction register (P3D)	0047_{16}	DMAC channel x transfer count register High (DMAxCH)
0010_{16}	Port control register (PTC)	0048_{16}	Data bus buffer register 0 (DBB0)
0011_{16}	Interrupt polarity select register (IPOL)	0049_{16}	Data bus buffer status register 0 (DBBS0)
0012_{16}	Port P2 pull-up control register (PUP2)	$004A_{16}$	Data bus buffer control register 0 (DBBC0)
0013_{16}	USB control register (USBC)	$004B_{16}$	Resereved (**Note 1**)
0014_{16}	Port P6 (P6)	$004C_{16}$	Data bus buffer register 1 (DBB1)
0015_{16}	Port P6 direction register (P6D)	$004D_{16}$	Data bus buffer status register 1 (DBBS1)
0016_{16}	Port P5 (P5)	$004E_{16}$	Data bus buffer control register 1 (DBBC1)
0017_{16}	Port P5 direction register (P5D)	$004F_{16}$	Resereved (**Note 1**)
0018_{16}	Port P4 (P4)	0050_{16}	USB address register (USBA)
0019_{16}	Port P4 direction register (P4D)	0051_{16}	USB power management register (USBPM)
$001A_{16}$	Port P7 (P7)	0052_{16}	USB interrupt status register 1 (USBIS1)
$001B_{16}$	Port P7 direction register (P7D)	0053_{16}	USB interrupt status register 2 (USBIS2)
$001C_{16}$	Port P8 (P8)	0054_{16}	USB interrupt enable register 1 (USBIE1)
$001D_{16}$	Port P8 direction register (P8D)	0055_{16}	USB interrupt enable register 2 (USBIE2)
$001E_{16}$	Resereved (**Note 1**)	0056_{16}	USB frame number register Low (USBSOFL)
$001F_{16}$	Clock control register (CCR)	0057_{16}	USB frame number register High (USBSOFH)
0020_{16}	Timer XL (TXL)	0058_{16}	USB endpoint index register (USBINDEX)
0021_{16}	Timer XH (TXH)	0059_{16}	USB endpoint x IN control register (IN_CSR)
0022_{16}	Timer YL (TYL)	$005A_{16}$	USB endpoint x OUT control register (OUT_CSR)
0023_{16}	Timer YH (TYH)	$005B_{16}$	USB endpoint x IN max. packet size register (IN_MAXP)
0024_{16}	Timer 1 (T1)	$005C_{16}$	USB endpoint x OUT max. packet size register (OUT_MAXP)
0025_{16}	Timer 2 (T2)	$005D_{16}$	USB endpoint x OUT write count register Low (WRT_CNTL)
0026_{16}	Timer 3 (T3)	$005E_{16}$	USB endpoint x OUT write count register High (WRT_CNTH)
0027_{16}	Timer X mode register (TXM)	$005F_{16}$	USB endpoint FIFO mode register (USBFIFOMR)
0028_{16}	Timer Y mode register (TYM)	0060_{16}	USB endpoint 0 FIFO (USBFIFO0)
0029_{16}	Timer 123 mode register (T123M)	0061_{16}	USB endpoint 1 FIFO (USBFIFO1)
$002A_{16}$	Serial I/O shift register (SIOSHT)	0062_{16}	USB endpoint 2 FIFO (USBFIFO2)
$002B_{16}$	Serial I/O control register 1 (SIOCON1)	0063_{16}	USB endpoint 3 FIFO (USBFIFO3)
$002C_{16}$	Serial I/O control register 2 (SIOCON2)	0064_{16}	USB endpoint 4 FIFO (USBFIFO4)
$002D_{16}$	Special count source generator 1 (SCSG1)	0065_{16}	Resereved (**Note 1**)
$002E_{16}$	Special count source generator 2 (SCSG2)	0066_{16}	Resereved (**Note 1**)
$002F_{16}$	Special count source mode register (SCSGM)	0067_{16}	Resereved (**Note 1**)
0030_{16}	UART1 mode register (U1MOD)	0068_{16}	Resereved (**Note 1**)
0031_{16}	UART1 baud rate generator (U1BRG)	0069_{16}	Resereved (**Note 1**)
0032_{16}	UART1 status register (U1STS)	$006A_{16}$	Flash memory control register (FMCR) (**Note 2**)
0033_{16}	UART1 control register (U1CON)	$006B_{16}$	Resereved (**Note 1**)
0034_{16}	UART1 transmit/receive buffer register 1 (U1TRB1)	$006C_{16}$	Frequency synthesizer control register (FSC)
0035_{16}	UART1 transmit/receive buffer register 2 (U1TRB2)	$006D_{16}$	Frequency synthesizer multiply register 1 (FSM1)
0036_{16}	UART1 RTS control register (U1RTSC)	$006E_{16}$	Frequency synthesizer multiply register 2 (FSM2)
0037_{16}	Resereved (**Note 1**)	$006F_{16}$	Frequency synthesizer divide register (FSD)
		$FFC9_{16}$	ROM code protect control register (ROMCP) (**Note 3**)

Notes 1: Do not write any data to this addresses, because these areas are reserved.
2: This area is reserved in the mask ROM version.
3: This area is on the ROM in the mask ROM version.

Fig. 10 Memory map of special function register (SFR)

I/O PORTS

Direction Registers

The I/O ports P0–P8 have direction registers which determine the input/output direction of each individual pin. Each bit in a direction register corresponds to one pin, each pin can be set to be input port or output port.

When “0” is written to the bit corresponding to a pin, that pin becomes an input pin. When “1” is written to that bit, that pin becomes an output pin.

If data is read from a pin set to output, the value of the port output latch is read, not the value of the pin itself. Pins set to input are floating. If a pin set to input is written to, only the port output latch is written to and the pin remains floating.

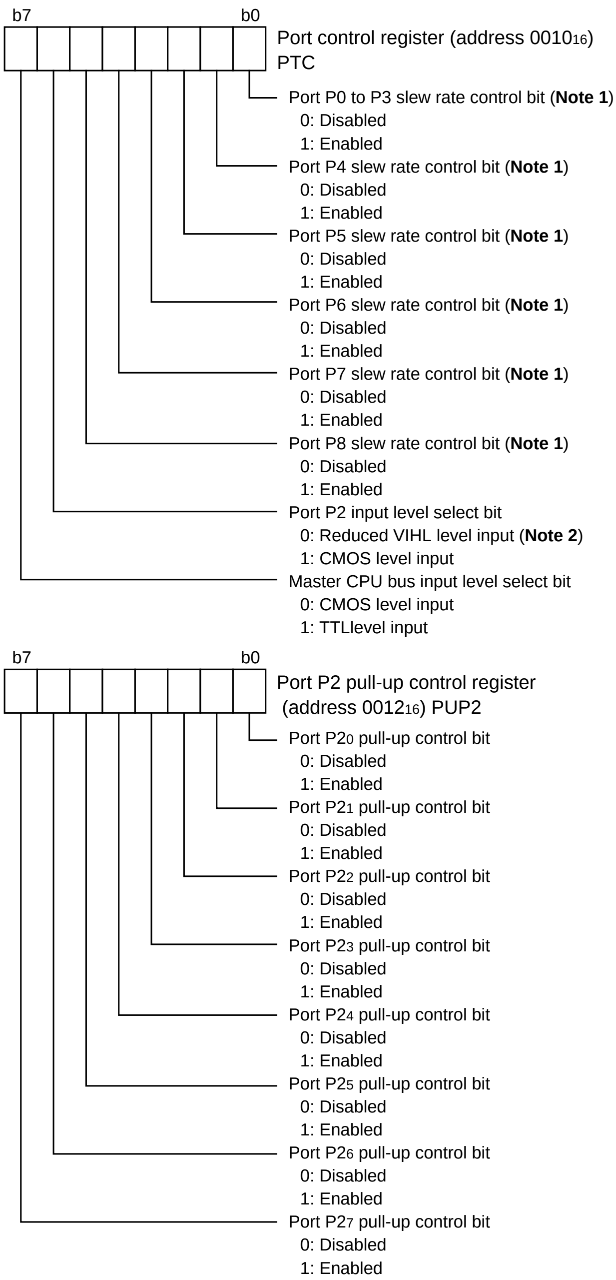
Slew Rate Control

By setting bits 0 to 5 of the port control register (address 0010_{16}) to “1”, slew rate control is enabled. VIHL or CMOS level can be used as a port P2 input level; CMOS or TTL level can be used as an input level of master CPU bus interface.

Pull-up Control

By setting the port P2 pull-up control register (address 0012_{16}), pull-up of each pin of port P2 can be controlled with a program.

However, the contents of port P2 pull-up control register do not affect ports programmed as the output ports but as the input ports.



Notes 1: The slew rate function can reduce di/dt by modifying an internal buffer structure.

2: The characteristics of VIHL level is basically the same as that of TTL level. But, its switching center point is a little higher than TTL’s. Refer to section “Recommended operating conditions”.

Fig. 11 Structure of port control and port P2 pull-up control registers

Table 6 List of I/O port function

Pin	Name	Input/Output	I/O format	Non-port function	Related SFRs	Ref. No.
P00/AB0–P07/AB7	Port P0	Input/Output, individual bits	CMOS input level CMOS 3-state output	Lower address output	CPU mode register A Port control register	(1)
P10/AB8–P17/AB15	Port P1			Higher address output		
P20/DB0–P27/DB7	Port P2		CMOS input level/VIHL input level CMOS 3-state output	Data bus I/O	CPU mode register A Port control register Port P2 pull-up control register	(2)
P30/RDY–P37/RD	Port P3		CMOS input level CMOS 3-state output	Control signal I/O	CPU mode register A CPU mode register B Port control register	(1) (3)
P40/EDMA,	Port P4					
P41/INT0, P42/INT1, P43/CNTR0, P44/CNTR1				Control signal I/O External interrupt	CPU mode register A CPU mode register B Port control register Timer X mode register Timer Y mode register Interrupt polarity select register	(4) (5)
P50/XCIN, P51/TOUT/XCOUT	Port P5		CMOS input level CMOS 3-state output	Timer 1, Timer 2 output pin Sub-clock generating input pin	CPU mode register A Port control register Clock control register Timer 123 mode register	(6) (7)
P52/OBF0, P53/IBF0, P54/S0, P55/A0, P56/R(E), P57/W(R/W)			CMOS input level CMOS 3-state output CMOS input level/TTL input level in Master CPU bus inferface function	Master CPU bus interface I/O pin	Data bus buffer control register 0 Port control register	(8) (9) (10)
P60/DQ0–P67/DQ7	Port P6		CMOS input level/TTL input level CMOS 3-state output	Master CPU bus interface I/O pin	Data bus buffer control register 0 Port control register	(11)
P70/SOF,	Port P7		CMOS input level CMOS 3-state output	USB function output pin	USB control register Port control register	(12)
P71/HOLD, P72/S1, P73/IBF1/HLDA, P74/OBF1			CMOS input level CMOS 3-state output CMOS input level/TTL input level in Master CPU bus inferface function	Control signal I/O Master CPU bus interface I/O pin	Data bus buffer control register 1 Port control register CPU mode register B	(13) (14) (15) (16)
P80/UTXD2/SRDY, P81/URXD2/SCLK, P82/CTS2/SRXD, P83/RTS2/STXD, P84/UTXD1, P85/URXD1, P86/CTS1, P87/RTS1	Port P8		CMOS input level CMOS 3-state output	Serial I/O I/O pin UART2 I/O pin UART1 I/O pin	UART1, 2 control registers Serial I/O control register 1 Serial I/O control register 2 Port control register	(17) (18) (19) (20) (21) (22) (23) (24)

Notes 1: For details of the ports functions in modes other than single-chip mode, and how to use double-function ports as function I/O ports, refer to the applicable sections.
2: Make sure that the input level at each pin is either 0 V or VCC during execution of the STP instruction.
When an input level is at an intermediate potential, a rush current will flow from VCC to VSS through the input-stage gate.

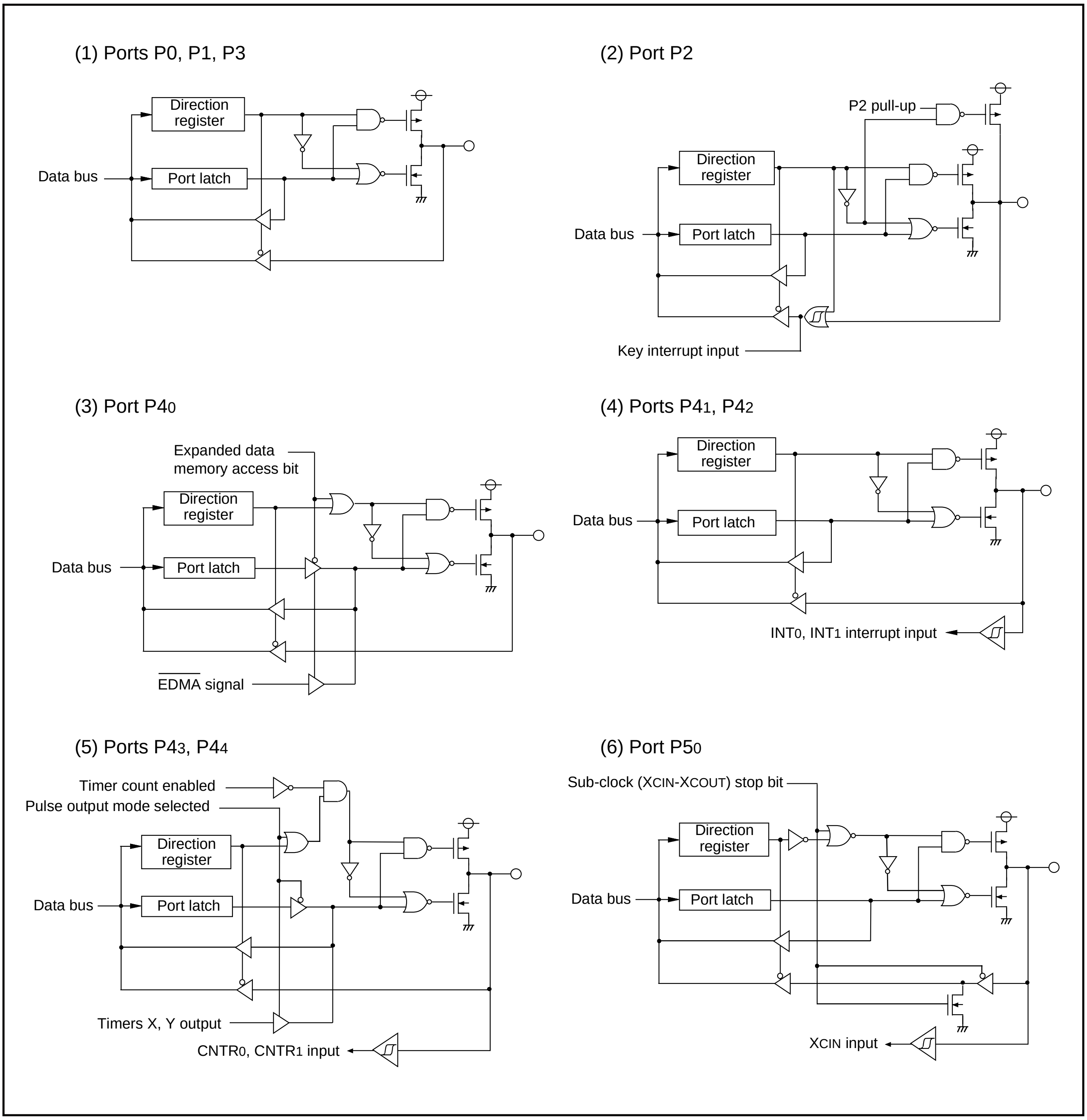
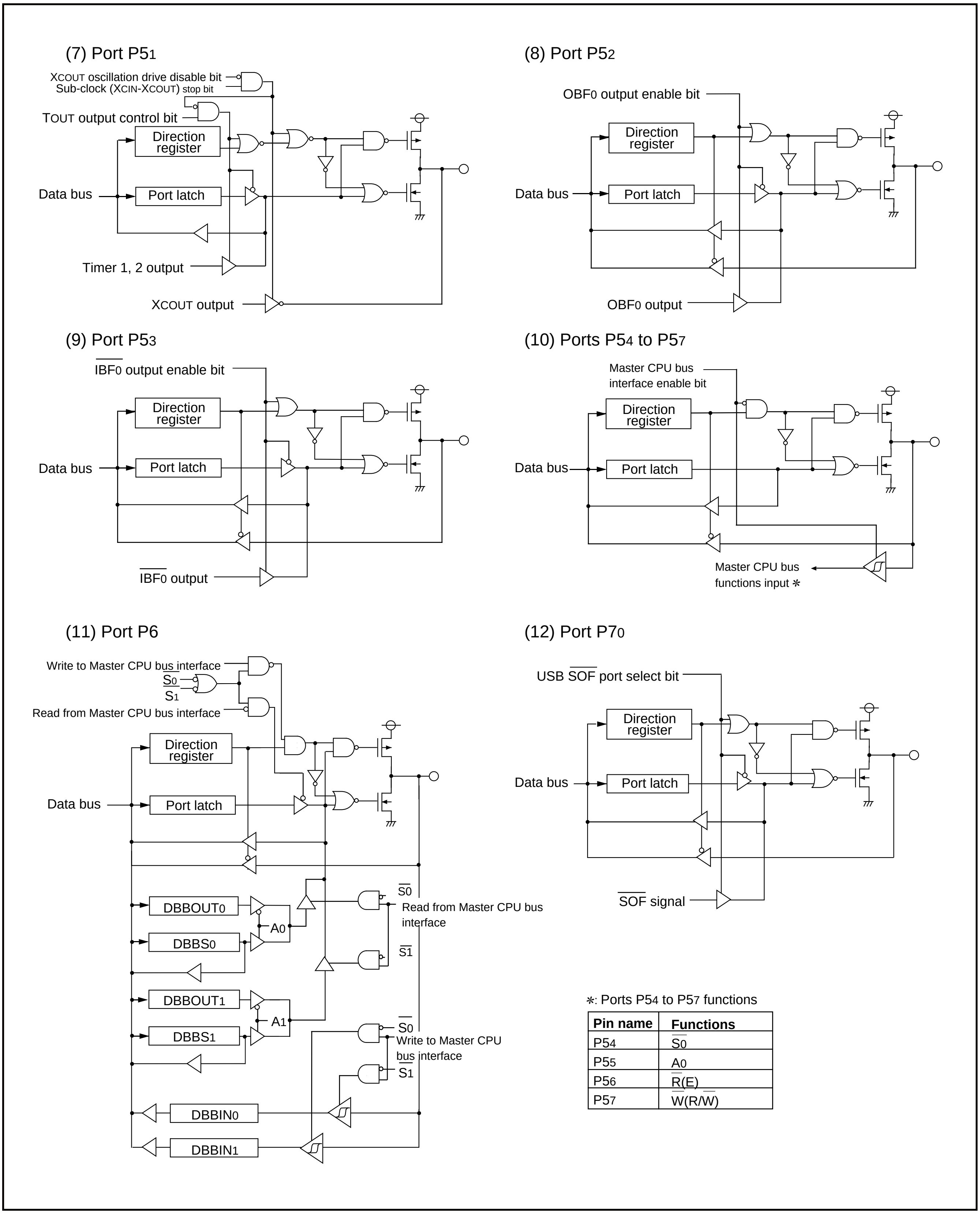


Fig. 12 Port block diagram (1)



Pin name	Functions
P54	S0
P55	A0
P56	R(E)
P57	W(R/W)

Fig. 13 Port block diagram (2)

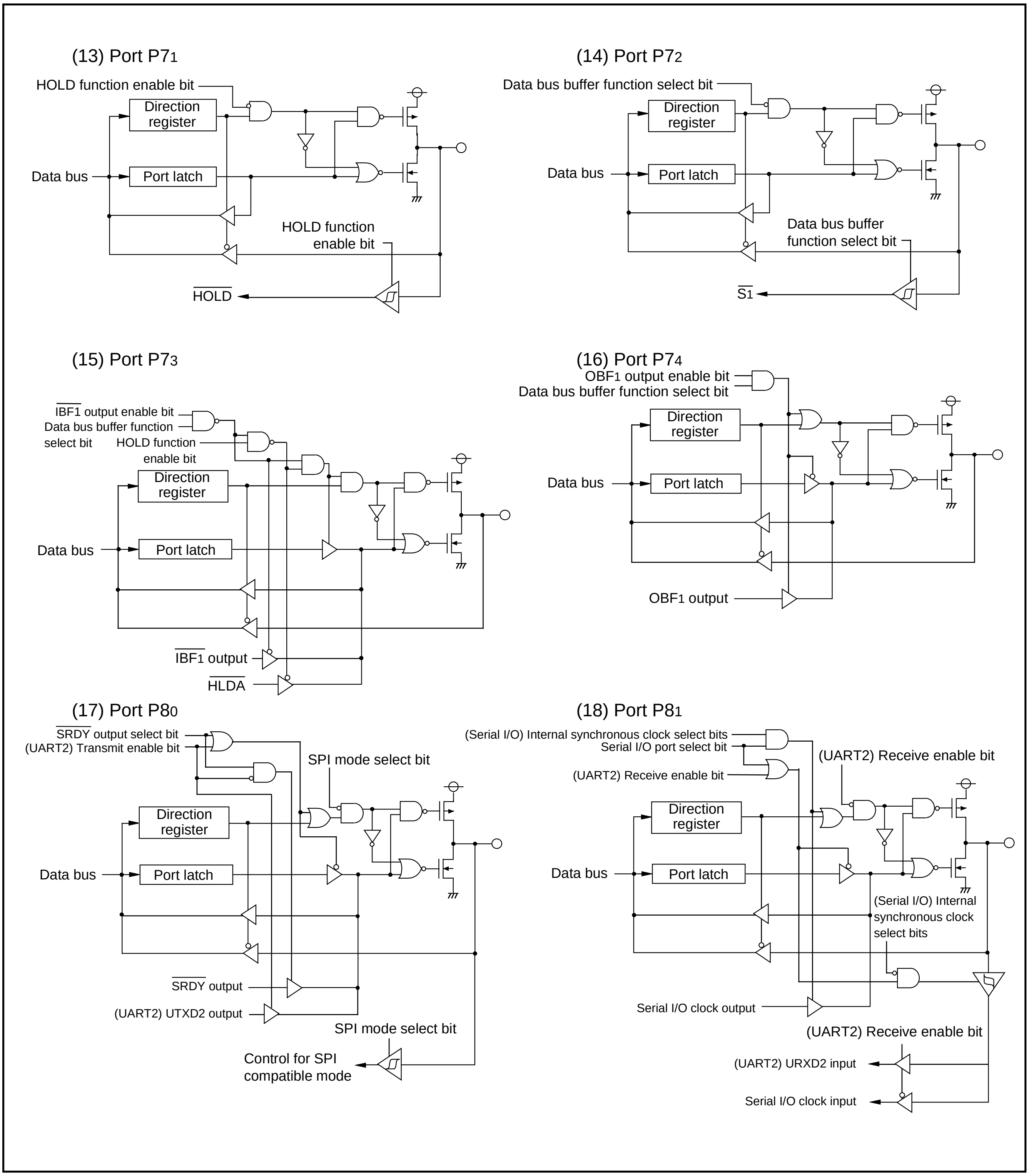


Fig. 14 Port block diagram (3)

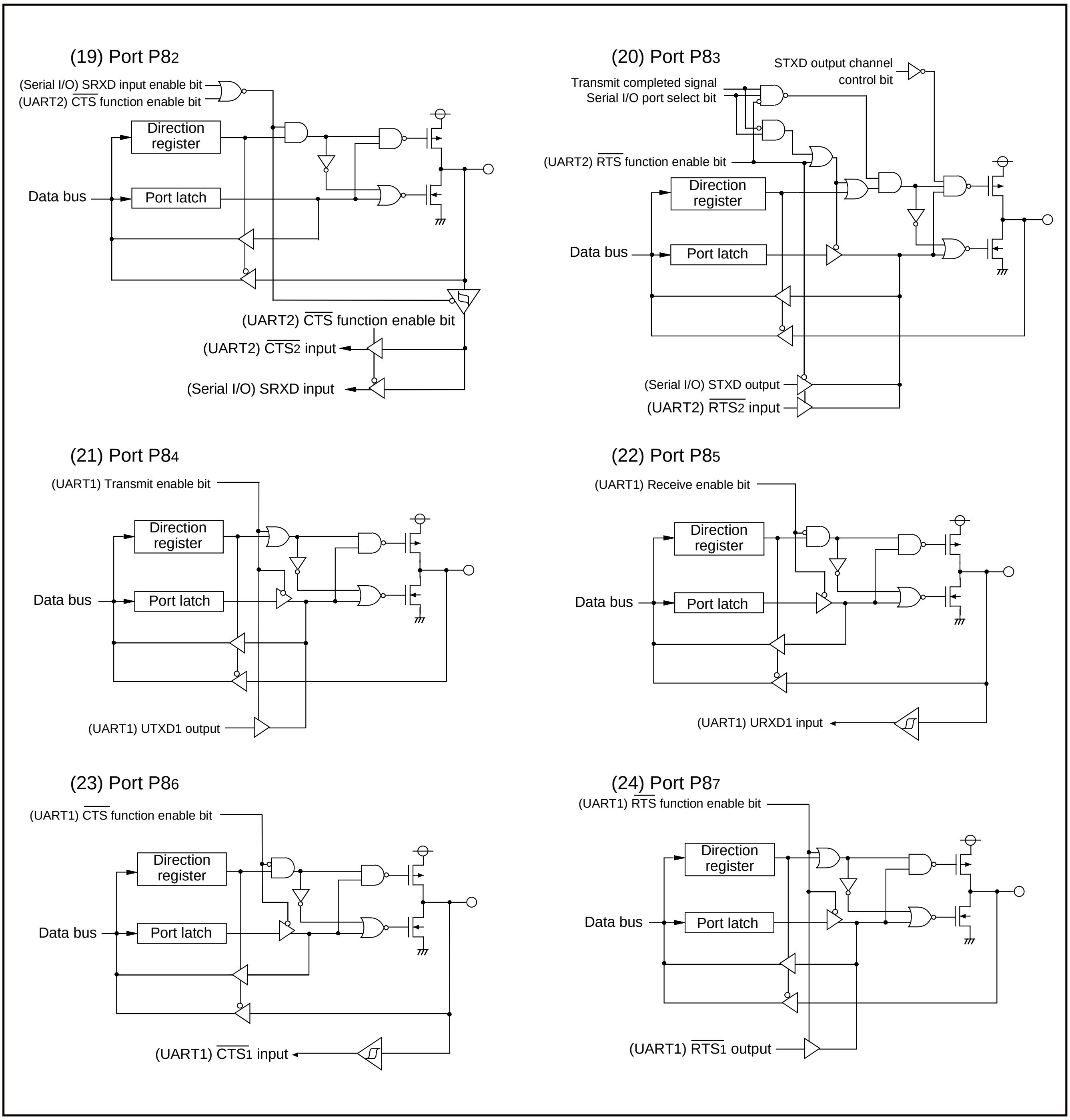


Fig. 15 Port block diagram (4)

INTERRUPTS

There are twenty-four interrupt sources: five externals, eighteen internals, and one software.

Interrupt Control

Each interrupt except the BRK instruction interrupt has both an Interrupt Request Bit and an Interrupt Enable Bit, and is controlled by the Interrupt Disable Flag (I). An interrupt occurs if the corresponding Interrupt Request and Enable Bits are “1” and the Interrupt Disable Flag is “0”.
Interrupt Enable Bits can be set or cleared by software. Interrupt Request Bits can be cleared by software, but cannot be set by software. Additionally, an active edge of INT_1 and INT_2 can be selected by using the interrupt edge select register (address 0011_{16}); an active edge of $CNTR_0$ can be done by using the timer X mode register (address 0027_{16}); an active edge of $CNTR_1$ can be done by using the timer Y mode register (address 0028_{16}).
The BRK instruction interrupt and reset cannot be disabled with any flag or bit. The I Flag disables all interrupts except the BRK instruction interrupt and reset. If several interrupts requests occur at the same time, the interrupt with the highest priority is accepted first.

Interrupt Operation

When an interrupt request occurs, the following operations are automatically performed:

1. The processing being executed is stopped.
2. The contents of the program counter and processor status register are automatically pushed onto the stack.
3. The Interrupt Disable Flag is set and the corresponding interrupt request bit is cleared.
4. The interrupt jump destination address is read from the vector table into the program counter.

■Notes

When setting the followings, the interrupt request bit may be set to “1”.
•When setting external interrupt active edge
Related register: Interrupt polarity select register (address 0011_{16})
Timer X mode register (address 0027_{16})
Timer Y mode register (address 0028_{16})

When not requiring for the interrupt occurrence synchronized with these setting, take the following sequence.
①Set the corresponding Interrupt Enable Bit to “0” (disabled).
②Set the Interrupt Edge Select Bit (Active Edge Switch Bit).
③Set the corresponding Interrupt Request Bit to “0” after 1 or more instructions have been executed.
④Set the corresponding Interrupt Enable Bit to “1” (enabled).

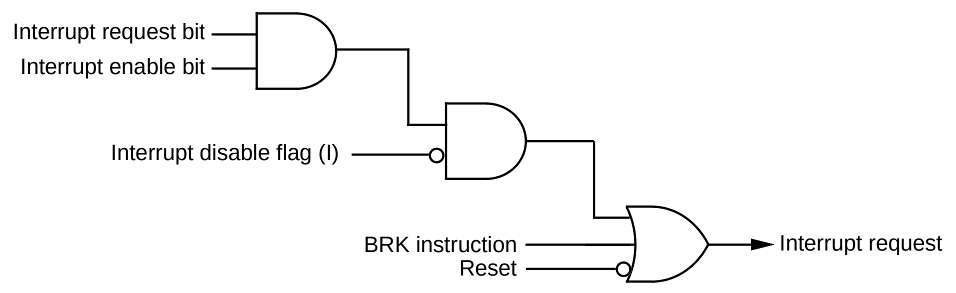


Fig. 16 Interrupt control

Table 7 Interrupt vector addresses and priority

Interrupt Source	Priority	Vector Addresses **(Note 1)**		Interrupt Request Generating Conditions	Remarks
		High	Low		
Reset **(Note 3)**	1	$FFFB_{16}$	$FFFA_{16}$	At reset	Non-maskable
USB function	2	$FFF9_{16}$	$FFF8_{16}$	(**Note 2**)	
USB SOF	3	$FFF7_{16}$	$FFF6_{16}$	At reception of SOF packet	
INT_0	4	$FFF5_{16}$	$FFF4_{16}$	At detection of either rising or falling edge of INT_0 intput	External interrupt (active edge selectable)
INT_1	5	$FFF3_{16}$	$FFF2_{16}$	At detection of either rising or falling edge of INT_1 input	External interrupt (active edge selectable)
DMAC0	6	$FFF1_{16}$	$FFF0_{16}$	At completion of DMAC0 transfer	
DMAC1	7	$FFEF_{16}$	$FFEE_{16}$	At completion of DMAC1 transfer	
UART1 receive buffer full	8	$FFED_{16}$	$FFEC_{16}$	At completion of UART1 reception	
UART1 transmit	9	$FFEB_{16}$	$FFEA_{16}$	At completion of UART1 transmission	
UART1 summing error	10	$FFE9_{16}$	$FFE8_{16}$	At detection of UART1 summing error	
UART2 receive buffer full	11	$FFE7_{16}$	$FFE6_{16}$	At completion of UART2 reception	
UART2 transmit	12	$FFE5_{16}$	$FFE4_{16}$	At completion of UART2 transmission	
UART2 summing error	13	$FFE3_{16}$	$FFE2_{16}$	At detection of UART2 summing error	
Timer X	14	$FFE1_{16}$	$FFE0_{16}$	At timer X underflow	
Timer Y	15	$FFDF_{16}$	$FFDE_{16}$	At timer Y underflow	
Timer 1	16	$FFDD_{16}$	$FFDC_{16}$	At timer 1 underflow	
Timer 2	17	$FFDB_{16}$	$FFDA_{16}$	At timer 2 underflow	
Timer 3	18	$FFD9_{16}$	$FFD8_{16}$	At timer 3 underflow	
$CNTR_0$	19	$FFD7_{16}$	$FFD6_{16}$	At detection of either rising or falling edge of $CNTR_0$ input	External interrupt (active edge selectable)
$CNTR_1$	20	$FFD5_{16}$	$FFD4_{16}$	At detection of either rising or falling edge of $CNTR_1$ input	External interrupt (active edge selectable)
Serial I/O	21	$FFD3_{16}$	$FFD2_{16}$	At completion of serial I/O transmission/re-ception	
Input buffer full	22	$FFD1_{16}$	$FFD0_{16}$	At writing to input data bus buffer	
Output buffer empty	23	$FFCF_{16}$	$FFCE_{16}$	At reading from output data bus buffer	
Key input (Key-on wake-up)	24	$FFCD_{16}$	$FFCC_{16}$	At falling of port P2 input logical level AND	External interrupt (falling valid)
BRK instruction	25	$FFCB_{16}$	$FFCA_{16}$	At BRK instruction execution	Non-maskable software interrupt

Notes 1: Vector addresses contain interrupt jump destination addresses.
2: USB function interrupt occurs owing to an interrupt request of the endpoint x (x = 0 to 4) IN, endpoint x OUT, overrun/underrun, USB reset or suspend/ resume.
3: Reset functions in the same way as an interrupt with the highest priority.

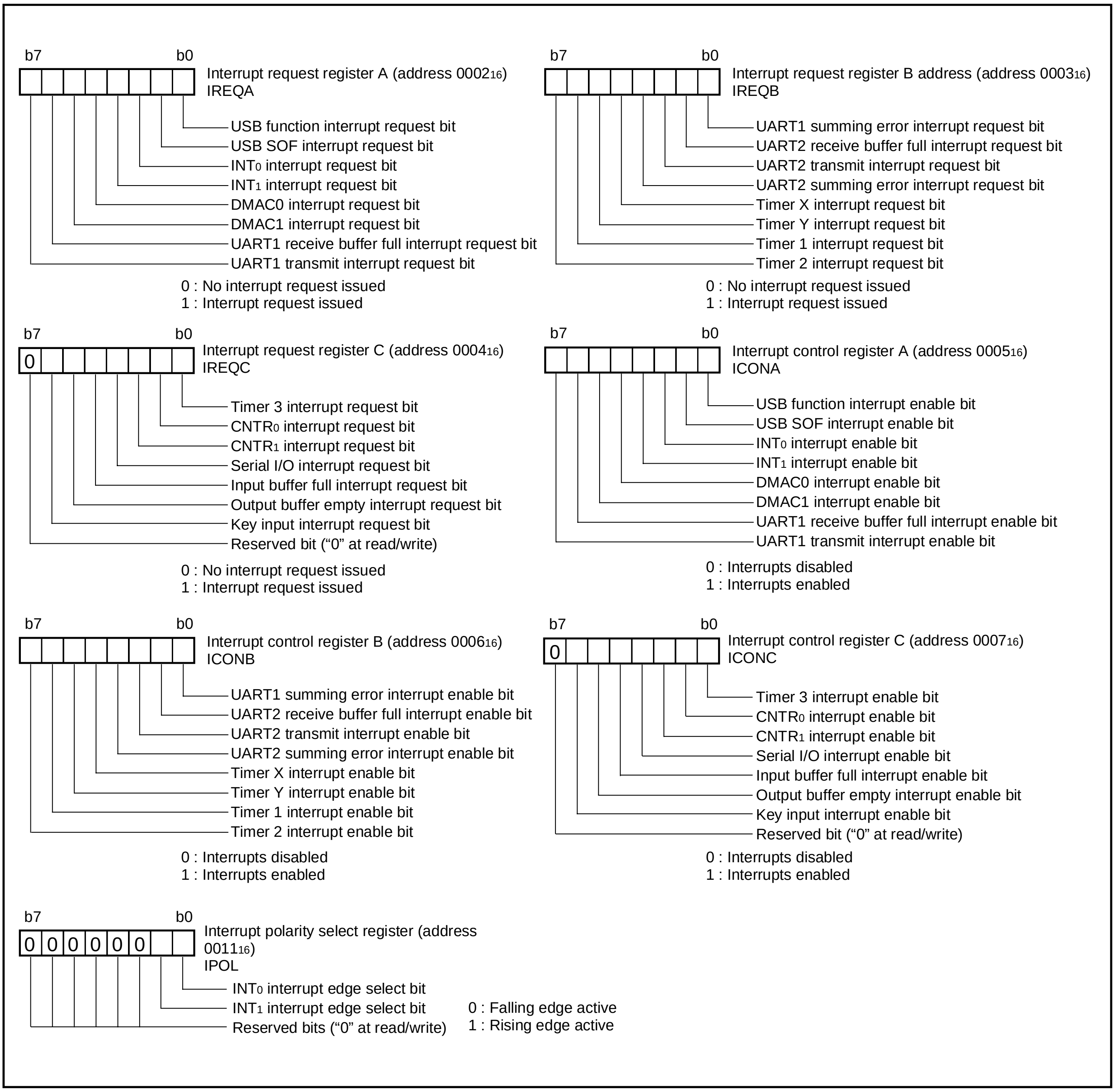


Fig. 17 Structure of interrupt-related registers

Key Input Interrupt (Key-on Wake-Up)

A key input interrupt request is generated by applying "L" level to any pin of port P2 that have been set to input mode. In other words, it is generated when AND of input level goes from "1" to "0". An example of using a key input interrupt is shown in Figure 18, where an interrupt request is generated by pressing one of the keys consisted as an active-low key matrix which inputs to ports $P2_0$–$P2_4$.

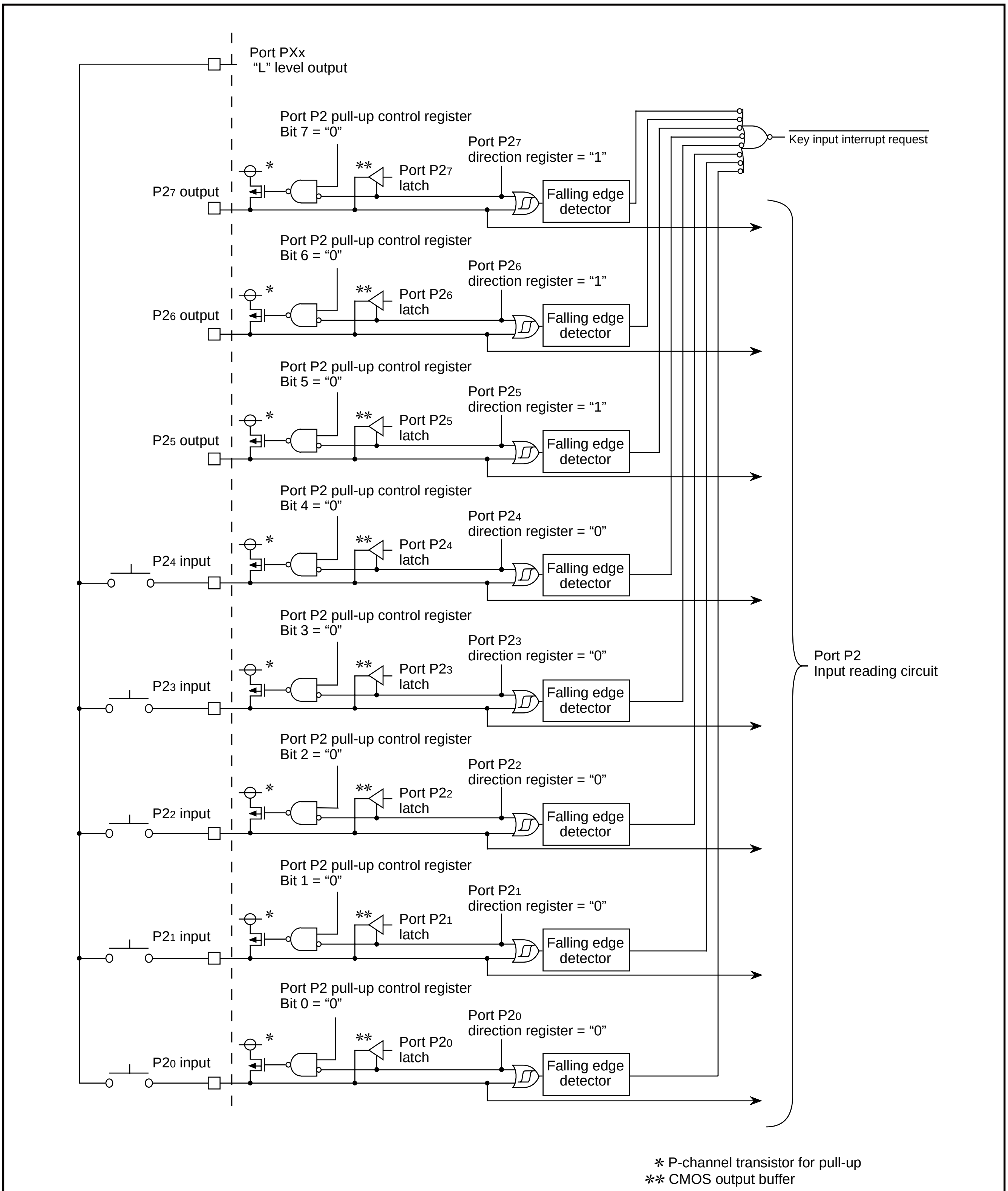


Fig. 18 Connection example when using key input interrupt and port P2 block diagram

TIMERS

The 7641 group has five timers: timer X, timer Y, timer 1, timer 2, and timer 3. Timer X and timer Y are 16-bit timers, and timer 1, timer 2, and timer 3 are 8-bit timers.
All timers are down count timers. When the timer reaches “00_{16}” or “0000_{16}”, an underflow occurs at the next count pulse and the corresponding timer latch is reloaded into the timer and the count is continued. When a timer underflows, the interrupt request bit corresponding to that timer is set to “1”.

Read and write operation on 16-bit timer must be performed for both high and low-order bytes. When reading a 16-bit timer, read the high-order byte first. When writing to a 16-bit timer, write the low-order byte first. The 16-bit timer cannot perform the correct operation when reading during the write operation, or when writing during the read operation.

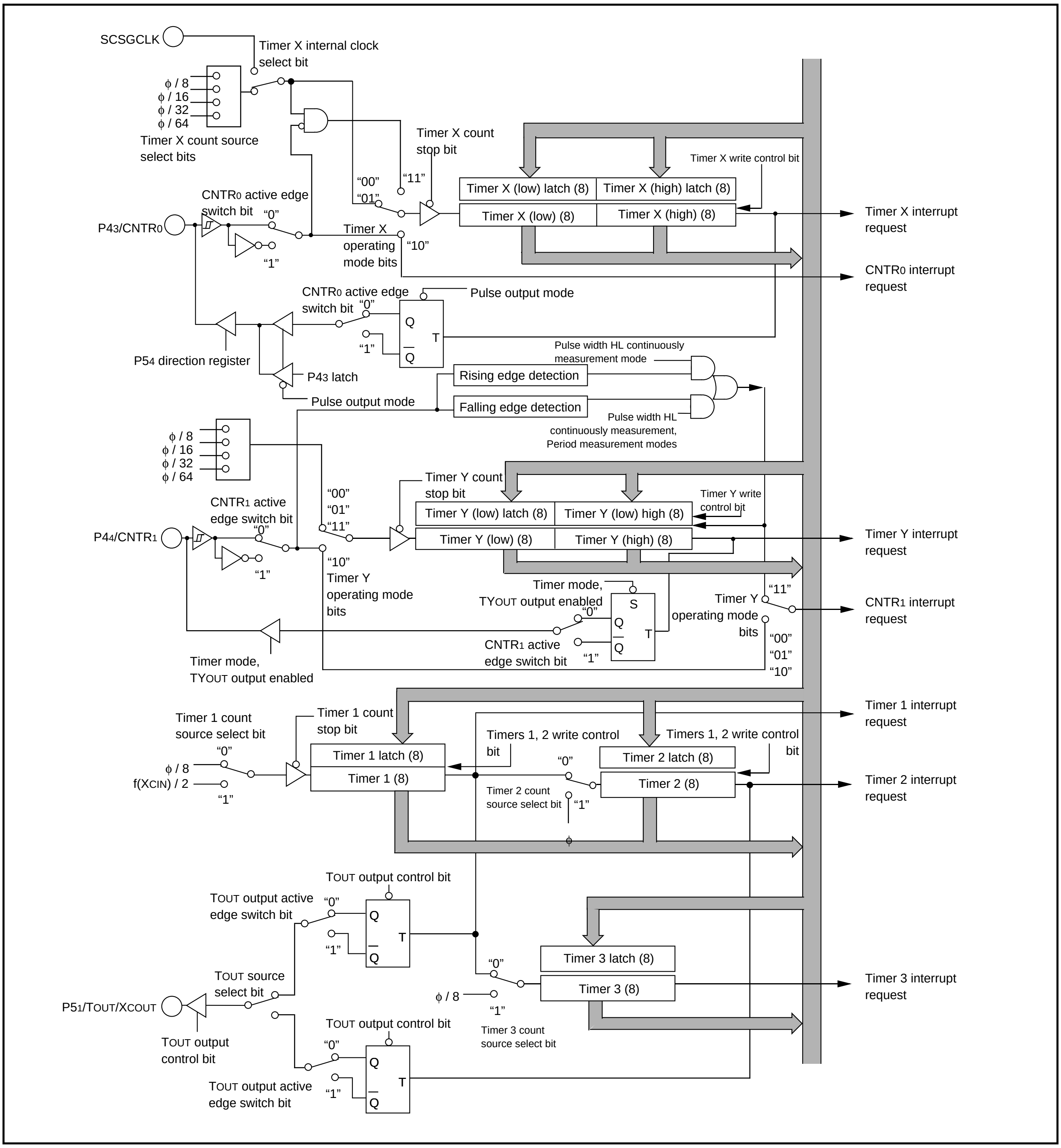


Fig. 19 Timer block diagram

Timer X

Timer X is a 16-bit timer that can be selected in one of four modes. The timer X's internal clock and count source can be selected and a write control is possible by using the timer X mode register.
In all modes the count operation can halt by setting the Timer X Count Stop Bit to "1". Additionally, each timer underflow sets the Interrupt Request Bit to "1".

(1) Timer Mode

The timer counts the SCSGCLK (Special Count Source Generator) or one of the internal clock ϕ divided by 8, 16, 32, 64.

(2) Pulse Output Mode

Each time the timer underflows, a signal output from the $CNTR_0$ pin is inverted. Except for this, the operation in pulse output mode is the same as in timer mode.
When the $CNTR_0$ Active Edge Switch Bit is "0", the $CNTR_0$ pin starts pulses output beginning at "H"; when this bit is "1", the $CNTR_0$ pin starts pulses output beginning at "L".
When using a timer in this mode, set the port $P4_3$ direction register to output mode.

(3) Event Counter Mode

The timer counts signals input through the $CNTR_0$ pin.
Except for this, the operation in event counter mode is the same as in timer mode.
When the $CNTR_0$ Active Edge Switch Bit is "0", the rising edge is counted; when this bit is "1", the falling edge is counted.
When using a timer in this mode, set the port $P4_3$ direction register to input mode.

(4) Pulse Width Measurement Mode

When the $CNTR_0$ Active Edge Switch Bit is "0", the timer counts while the input signal of $CNTR_0$ pin is at "H"; when it is "1", the timer counts while the input signal of $CNTR_0$ pin is at "L".
The timer counts the SCSGCLK or one of the internal clock ϕ divided by 8, 16, 32, 64 as its count source.
When using a timer in this mode, set the port $P4_3$ direction register to input mode.

■ Notes

● Timer X Write Control

If the Timer X Write Control Bit is "1", when the value is written in the address of timer X, the value is loaded only in the latch. The value in the latch is loaded in timer X after timer X underflows.
If the Timer X Write Control Bit is "0", when the value is written in the address of timer X, the value is loaded in the timer X and the latch at the same time.
When the value is to be written in latch only, unexpected value may be set in the high-order timer if the writing in high-order latch and the underflow of timer X are performed at the same timing.

● $CNTR_0$ Interrupt Active Edge Selection

The $CNTR_0$ interrupt active edge depends on the selection of $CNTR_0$ Active Edge Switch Bit.

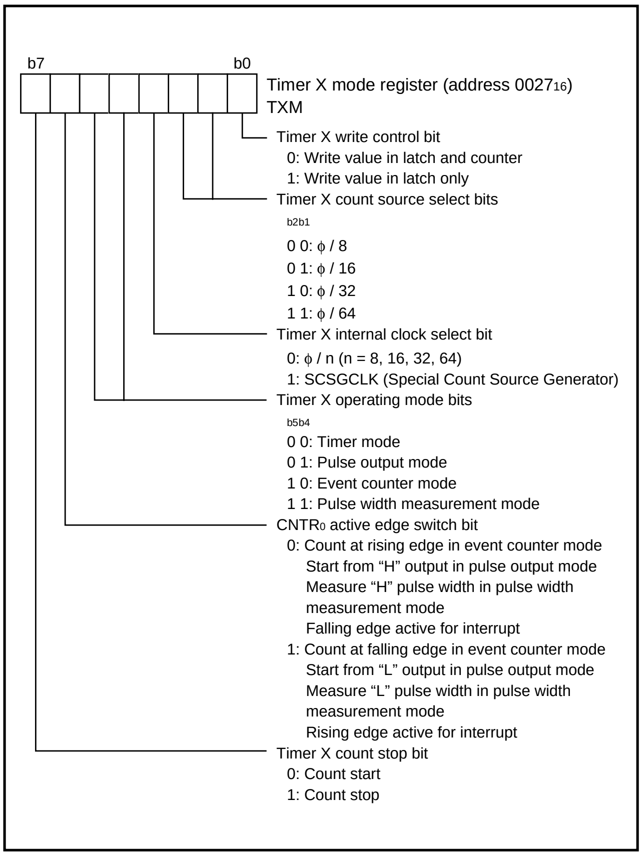


Fig. 20 Structure of timer X mode register

Timer Y

Timer Y is a 16-bit timer that can be selected in one of four modes.

(1) Timer Mode

The timer counts one of the internal clock ϕ divided by 8, 16, 32, 64.

● TY_{OUT} Output Function

In the timer mode, a signal of which polarity is inverted each time the timer underflows is output from the $CNTR_1$ pin. This is enabled by setting the Timer Y Output Control Bit to “1”.
When the $CNTR_1$ Active Edge Switch Bit is “0”, the $CNTR_1$ pin starts pulses output beginning at “H”; when this bit is “1”, the $CNTR_1$ pin starts pulses output beginning at “L”.
When using a timer in this mode, set the port $P4_4$ direction register to output mode.

(2) Period Measurement Mode

$CNTR_1$ interrupt request is generated at a rising/falling edge of $CNTR_1$ pin input signal. Simultaneously, the value in timer Y latch is reloaded in timer Y and timer Y continues counting down. Except for the aforementioned operation, the operation in period measurement mode is the same as in timer mode. (The TY_{OUT} output function is not usable.)
The timer value just before the reloading at rising/falling of $CNTR_1$ pin input signal is retained until the timer Y is read once after the reload.
The rising/falling timing of $CNTR_1$ pin input signal is found by $CNTR_1$ interrupt.
When the $CNTR_1$ Active Edge Switch Bit is “0”, the falling edge is detected; when this bit is “1”, the rising edge is detected.
When using a timer in this mode, set the port $P4_4$ direction register to input mode.

(3) Event Counter Mode

The timer counts signals input through the $CNTR_1$ pin.
Except for this, the operation in event counter mode is the same as in timer mode. (The TY_{OUT} output function is not usable.)
When the $CNTR_1$ Active Edge Switch Bit is “0”, the rising edge is counted; when this bit is “1”, the falling edge is counted.
When using a timer in this mode, set the port $P4_4$ direction register to input mode.

(4) Pulse Width HL Continuously Measurement Mode

$CNTR_1$ interrupt request is generated at both rising and falling edges of $CNTR_1$ pin input signal. Except for this, the operation in pulse width HL continuously measurement mode is the same as in period measurement mode.
When using a timer in this mode, set the port $P4_4$ direction register to input mode.

■ Notes

● Timer Y Write Control

If the Timer Y Write Control Bit is “1”, when the value is written in the address of timer Y, the value is loaded only in the latch. The value in the latch is loaded in timer Y after timer Y underflows.
If the Timer Y Write Control Bit is “0”, when the value is written in the address of timer Y, the value is loaded in the timer Y and the latch at the same time.
When the value is to be written in latch only, unexpected value may be set in the high-order timer if the writing in high-order latch and the underflow of timer Y are performed at the same timing.

● $CNTR_1$ Interrupt Active Edge Selection

The $CNTR_1$ interrupt active edge depends on the selection of $CNTR_1$ Active Edge Switch Bit.
However, in pulse width HL continuously measurement mode, $CNTR_1$ interrupt request is generated at both rising and falling edges of $CNTR_1$ pin input signal regardless of the setting of $CNTR_1$ Active Edge Switch Bit.

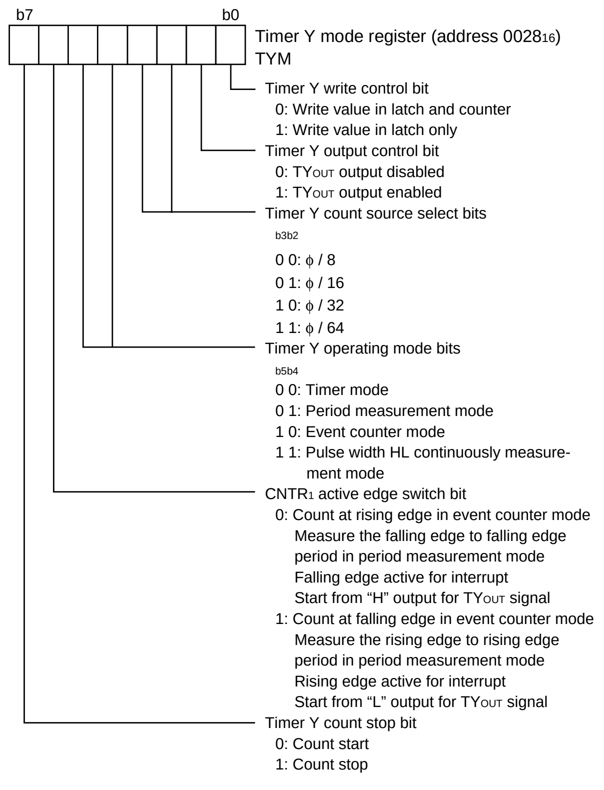


Fig. 21 Structure of timer Y mode register

Timer 1, Timer 2, Timer 3

Timer 1, timer 2, and timer 3 are 8-bit timers. The count source for each timer can be selected by timer 123 mode register.

● Timers 1, 2 Write Control

When the Timers 1, 2 Write Control Bit is “1” and the values are written in the address of timers 1 and 2, the values are loaded only in their latches. The values in the latches are loaded in timers 1 and 2 after timers 1 and 2 underflow.
When the Timers 1, 2 Write Control Bit is “0” and the values are written in the address of timers 1 and 2, the values are loaded in the timers 1 and 2 and their latches at the same time.

● Timers 1, 2 Output Control

A signal of which polarity is inverted each time the timer selected by the TOUT Factor Select Bit underflows is output from the TOUT pin. This is enabled by setting the TOUT Output Control Bit to “1”. When the TOUT Output Active Edge Switch Bit is “0”, the TOUT pin starts pulses output beginning at “H”; when this bit is “1”, the TOUT pin starts pulses output beginning at “L”.
When using a timer in this mode, set the port P51 direction register to output mode.

■ Notes

● Timer 1 to Timer 3

Switching of the count sources of timers 1 to 3 does not affect the values of reload latches. However, that may make count operation started. Therefore, write values again in the order of timers 1, 2 and then timer 3 after their count sources have been switched.

● Timers 1, 2 Write Control

When the value is to be written in latch only, unexpected value may be set in the timer if the writing in the latch and the timer underflow are performed at the same timing.

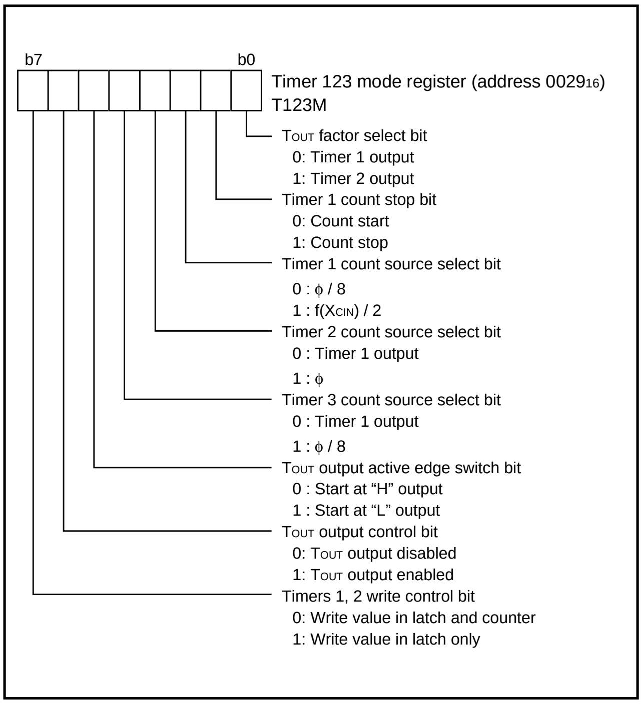


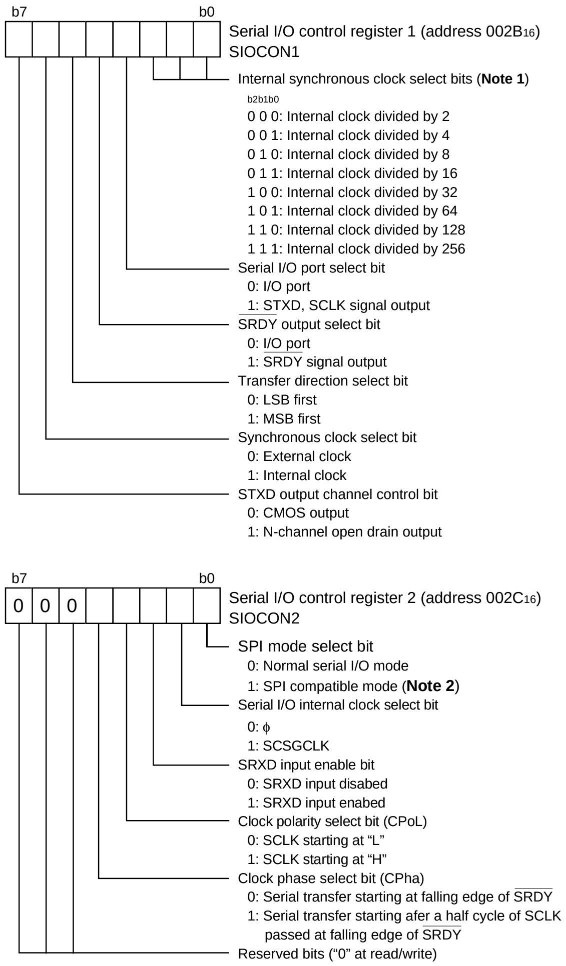
Fig. 22 Structure of timer 123 mode register

SERIAL INTERFACE

Serial I/O

The serial I/O can be used only for clock synchronous serial I/O. The transmitter and the receiver must use the same clock. If the internal clock is used, transfer is started by a write signal to the serial I/O shift register.

[Serial I/O Control Register 1 (SIOCON1)] $002B_{16}$
[Serial I/O Control Register 2 (SIOCON2)] $002C_{16}$
Each of the serial I/O control registers 1 and 2 contains eight bits which control various serial I/O functions.



Notes 1: The source of serial I/O internal synchronous clock can be selected by bit 1 of serial I/O control register 2.
2 : To set the slave mode, also set bit 4 of serial I/O control register 1 to "1".

Fig. 23 Structure of serial I/O control registers 1, 2

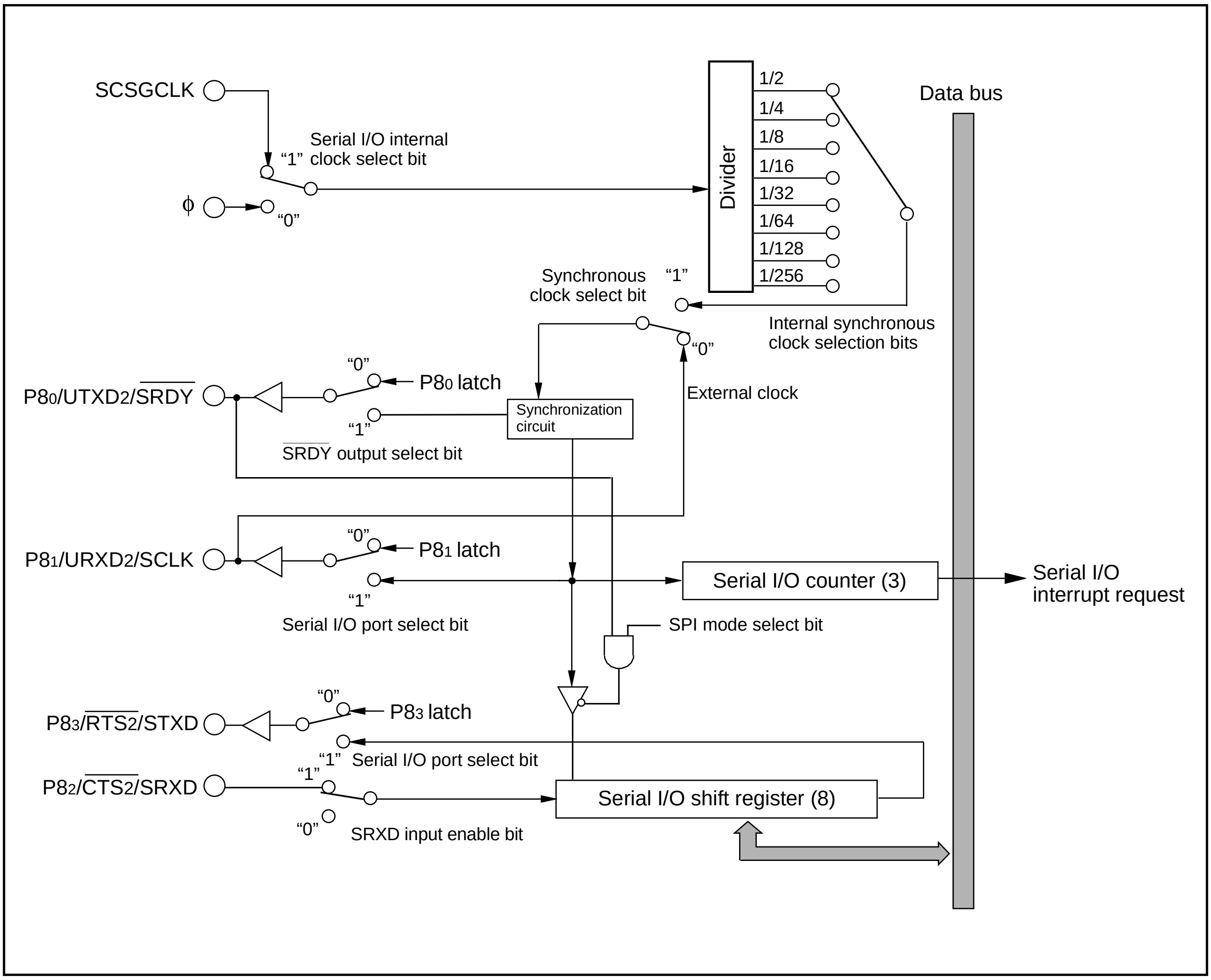


Fig. 24 Block diagram of serial I/O

● **Serial I/O Normal Operation**

The serial I/O counter is set to “7” by writing operation to the serial I/O shift register (address $002A_{16}$). When the SRDY Output Select bit is “1”, the SRDY pin goes “L” after that writing. On the negative edge of the transfer clock the SRDY pin returns “H” and the data of the first bit is transmitted from the STXD pin. The remaining data are done from the STXD pin bit by bit on each falling edge of the transfer clock.

Additionally, the data is latched from the SRXD pin on each rising edge of the transfer clock and then the contents of the serial I/O shift register are shifted by one bit.

When the internal system clock is selected as the transfer clock, the followings occur at counting eight transfer clocks:

•The serial I/O counter reaches “0”
•The transfer clock halts at “H”
•The serial I/O interrupt request bit is set to “1”
•The STXD pin goes a high-impedance state after an 8-bit transfer is completed.

When the external clock is selected as the transfer clock, the followings occur at counting eight transfer clocks:

•The serial I/O counter reaches “0”
•The serial I/O interrupt request bit is set to “1”

In this case, the transfer clock needs to be controlled by the external source because the transfer clock does not halt. Additionally, the STXD pin does not go a high-impedance state after an 8-bit transfer is completed.

Figure 25 shows serial I/O timing.

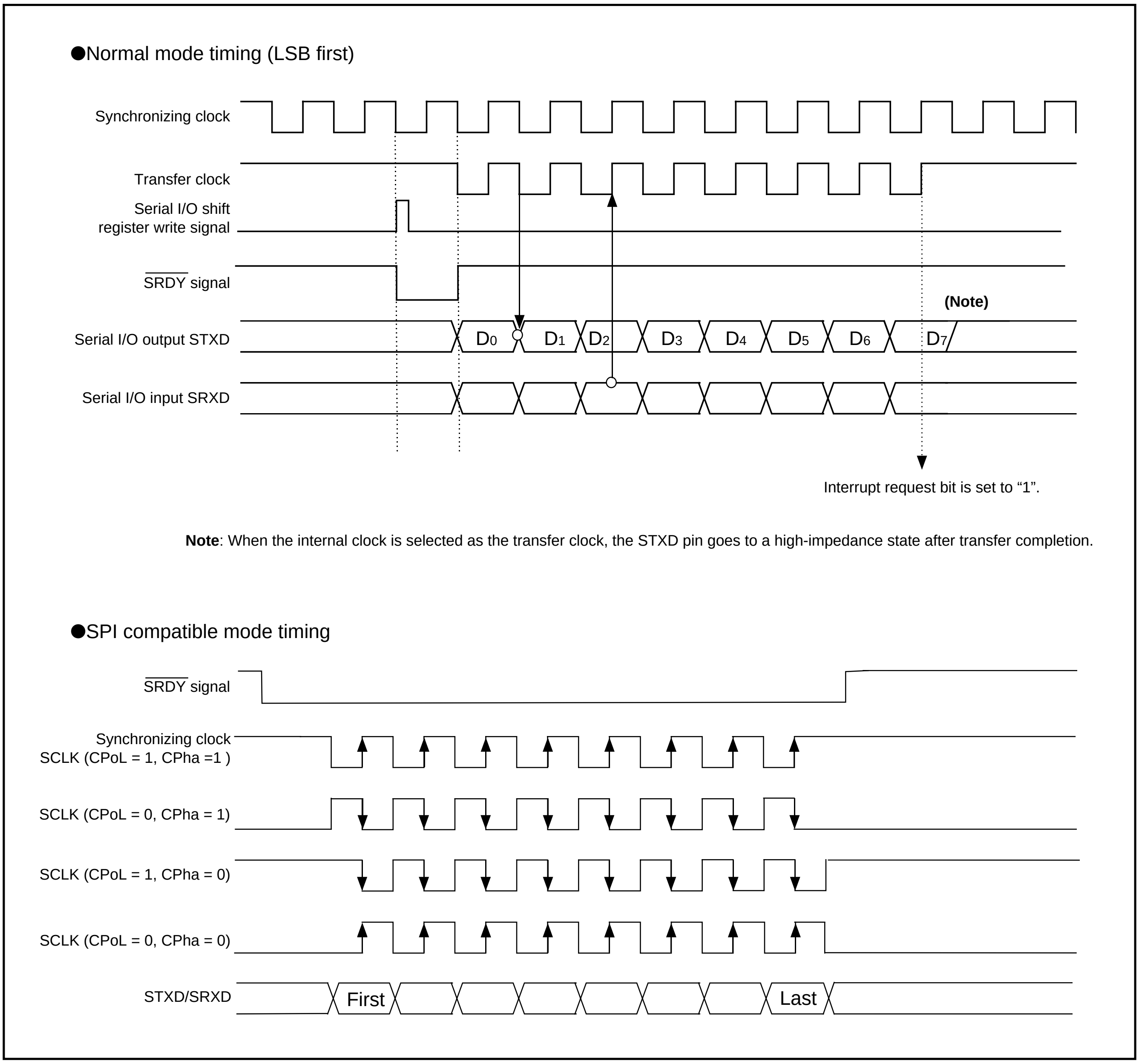


Fig. 25 Serial I/O timing

● SPI Compatible Mode Operation

Setting the SPI Mode Select Bit (bit 0 of SIOCON2) puts the serial I/O in SPI compatible mode. The Synchronous Clock Select Bit (bit 6 of SIOCON1) determines whether the serial I/O is an SPI master or slave. When the external clock is selected (“0”), the serial I/O is in slave mode; When the internal clock is selected (“1”), the serial I/O is in master mode.

In SPI compatible mode the SRXD pin functions as a MISO (Master In/Slave Out) pin and the STXD pin functions as a MOSI (Master Out/Slave In) pin.

In slave mode the transmit data is output from the MISO pin and the receive data is input from the MISO pin. The $\overline{\text{SRDY}}$ pin functions as the chip-select signal input pin from an external.

In master mode the transmit data is output from the MOSI pin and the receive data is input from the MISO pin. The $\overline{\text{SRDY}}$ pin functions as the chip-select signal output pin to an external.

• Slave Mode Operation

In slave mode of SPI compatible mode 4 types of clock polarity and clock phase can be usable by bits 3 and 4 of serial I/O control register 2.

If the $\overline{\text{SRDY}}$ pin is held “H”, the shift clock is inhibited, the serial I/O counter is set to “7”. If the $\overline{\text{SRDY}}$ pin is held “L”, then the shift clock will start.

Make sure during transfer to maintain the $\overline{\text{SRDY}}$ input at “L” and not to write data to the serial I/O counter.

Figure 25 shows the serial I/O timing.

UART1, UART2

The UART consists of two channels: UART1 and UART2. Each has a dedicated timer provided to generate transfer clocks and operates independently. Both UART1 and UART2 have the same functions.

Twelve serial data transfer formats can be selected, and the transfer formats used by a transmitter and receiver must be identical.

The transmit and receive shift registers each have a buffer, but the two buffers have the same address in a memory. Since the shift register cannot be written to or read from directly, transmit data is written to the transmit buffer register, and receive data is read from the receive buffer register.

The transmit buffer register can also hold the next data to be transmitted, and the receive buffer register can hold a character while the next character is being received.

The transfer speed (baud rate) is expression as follows:

Transfer speed (baud rate) = fi / {(n + 1) × 16 }

n: The contents of UARTx (x = 1, 2) baud rate generator

fi: Using UART clock prescaling select bits, select any one of ϕ, ϕ/8, ϕ/32, ϕ/256, SCSGCLK, SCSGCLK/8, SCSGCLK/32 and SCSGCLK/256

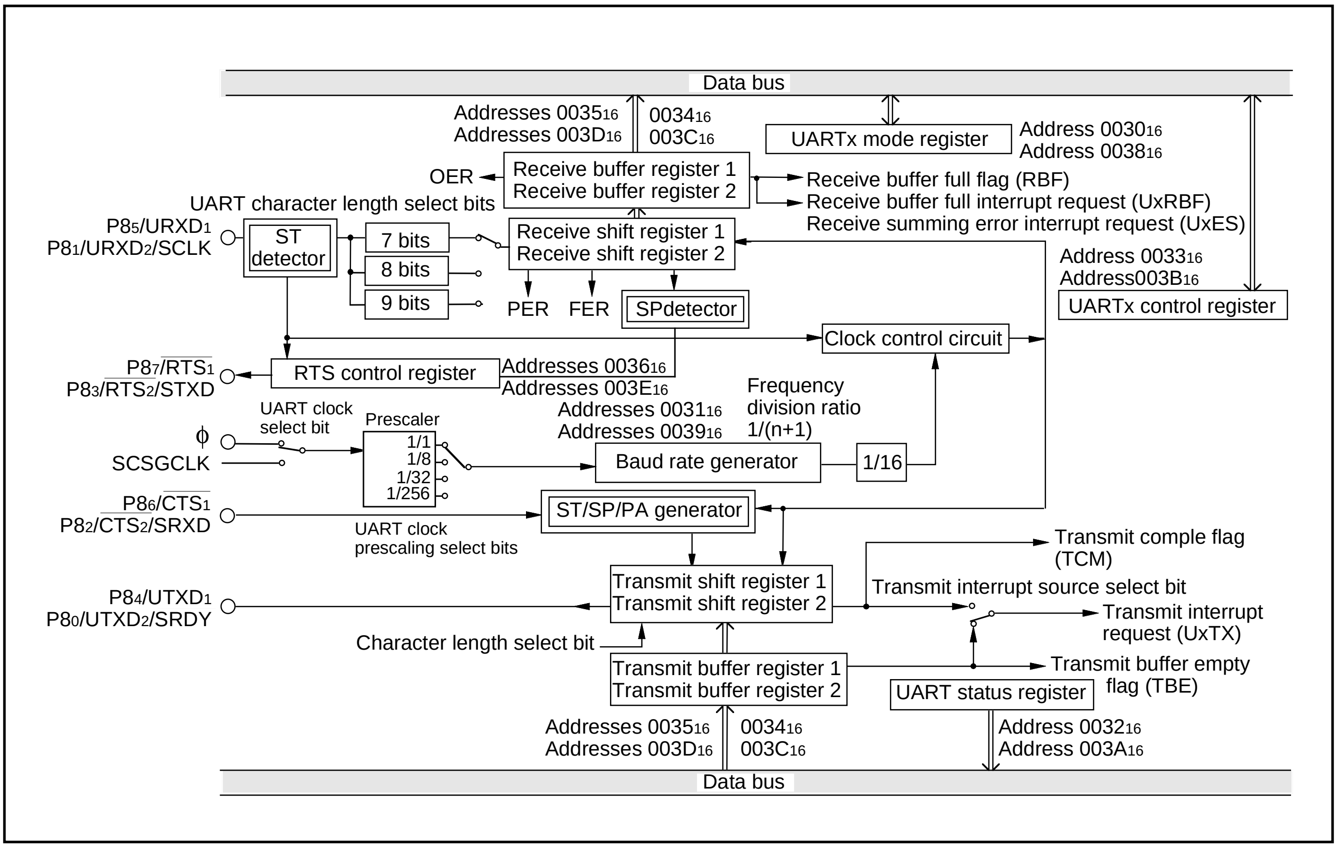


Fig. 26 UARTx (x = 1, 2) block diagram

● **UART Transmit Operation**

Transmission starts when the Transmit Enable Bit is “1” and the Transmit Buffer Empty Flag is “0”. Additionally, when CTS function enabled, the CTSx pin must be “L” to be started. The data in which Start Bit and Stop Bit or Parity Bit are also added is transmitted from the low-order byte sequentially. When using 9-bit character length, set the data into the UARTx transmit buffer register 2 (high-order byte) first before the UARTx transmit buffer register 1 (low-order byte).

Once the transmission starts, the Transmit Enable Bit, the Transmit Buffer Empty Flag and the CTSx pin state (when this is enabled) could not be checked until the transmission in progress has ended.

Transmission requires the following setup:

(1) Define a baud rate by setting a value n (n = 0 to 255) into UARTx baud rate generator (addresses 0031_{16}, 0039_{16}).

(2) Set the Transmit Initialization Bit (bit 2 of UxCON) to “1”. This will set the UARTx status register to “03_{16}”.

(3) Select the interrupt source with the Transmit Interrupt Source Select Bit (bit 4 of UxCON).

(4) Configure the data format and clock selection by setting the UARTx mode register.

(5) Set the CTS Function Enable Bit (bit 5 of UxCON) if CTS function will be used.

(6) Set the Transmit Enable Bit (bit 0 of UxCON) to “1”.

If updating a value of UARTx baud rate generator while the data is being transmitted, be sure to disable the transmission before updating. If the former data remains in the UARTx transmit buffer registers 1 and 2 at retransmission, an undefined data might be output.

● **UART Receive Operation**

Reception is enabled when the Receive Enable Bit is “1”. Detection of the start bit makes transfer clocks generated and the data reception starts in the LSB first.

When using 9-bit character length, read the received data from the UARTx receive buffer register 2 (high-order byte) first before the UARTx receive buffer register 1 (low-order byte).

Reception requires the following setup:

(1) Define a baud rate by setting a value n (n = 0 to 255) into UARTx baud rate generator (addresses 0031_{16}, 0039_{16}).

(2) Set the Receive Initialization Bit (bit 3 of UxCON) to “1”.

(3) Configure the data format and clock selection by setting the UARTx mode register.

(4) Set the RTS Function Enable Bit (bit 5 of UxCON) if RTS function will be used.

(5) Set the Receive Enable Bit (bit 1 of UxCON) to “1”.

● **CTS (Clear-to-Send) Function**

As a transmitter, the UART can be configured to recognize the Clear-to-Send (CTSx) input as a handshaking signal. This is enabled by setting the CTS Function Enable Bit (bit 5 of UxCON) to “1”. If CTS function is enabled, even when transmission is enabled and the UARTx transmit buffer register is filled with the data, the transmission never starts; but it will start when inputting “L” to the CTSx pin.

Figures 27 and 28 show the UARTx transmit timings.

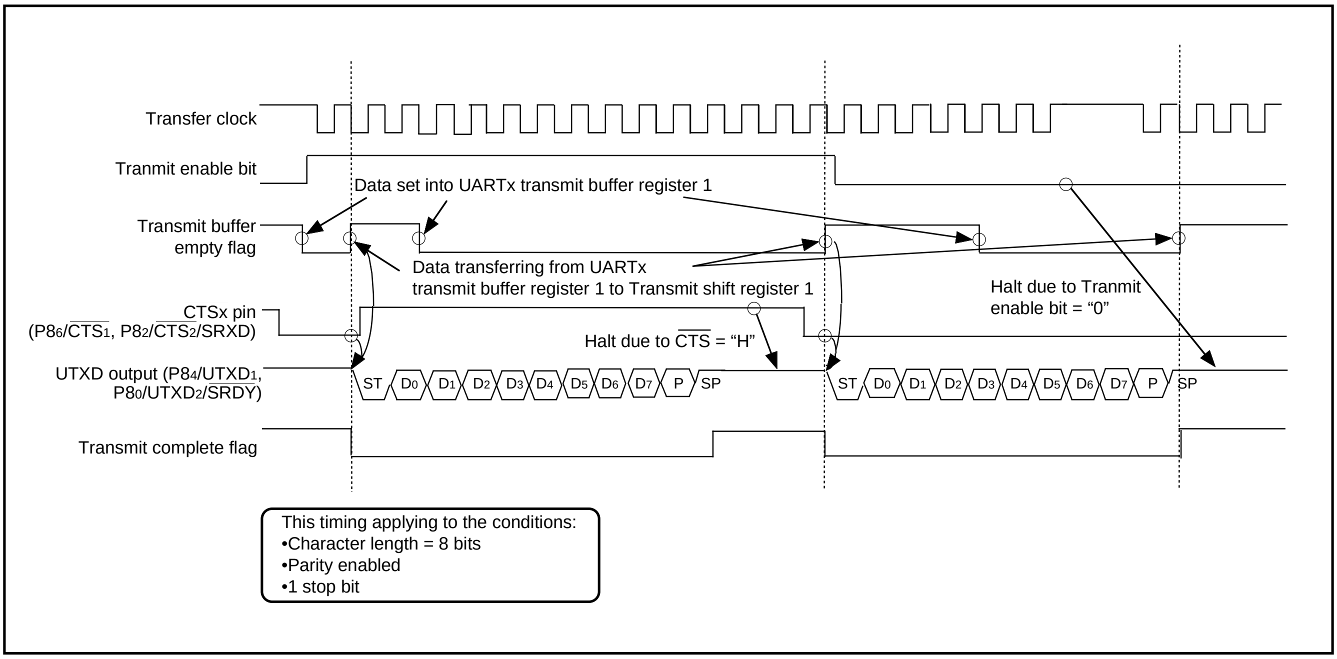


Fig. 27 UARTx transmit timing (CTS function enabled)

● **RTS (Request-to-Send) Function**

As a receiver, the UART can be configured to generate the Request-to-Send (RTSx) handshaking signal. This is enabled by setting the RTS Function Enable Bit (bit 6 of UxCON) to “1”.

When reception is enabled, that is the Receive Enable Bit is “1”, the RTSx pin goes “L” to inform a transmitter that reception is possible. The RTSx pin goes “H” at reception starting and does “L” at receiving of the last bit.

The delay time from the reception of the last stop bit to the assertion of RTSx is selectable using the RTS Assertion Delay Count Select Bits.

When the Receive Enable Bit is set to “0” or the Receive initialization bit is set to “1”, the RTSx pin goes “H”. Even when the Receive Enable Bit is set to “1”, the RTSx pin goes “H” if detecting an invalid start bit.

Figure 29 shows the UARTx receive timing.

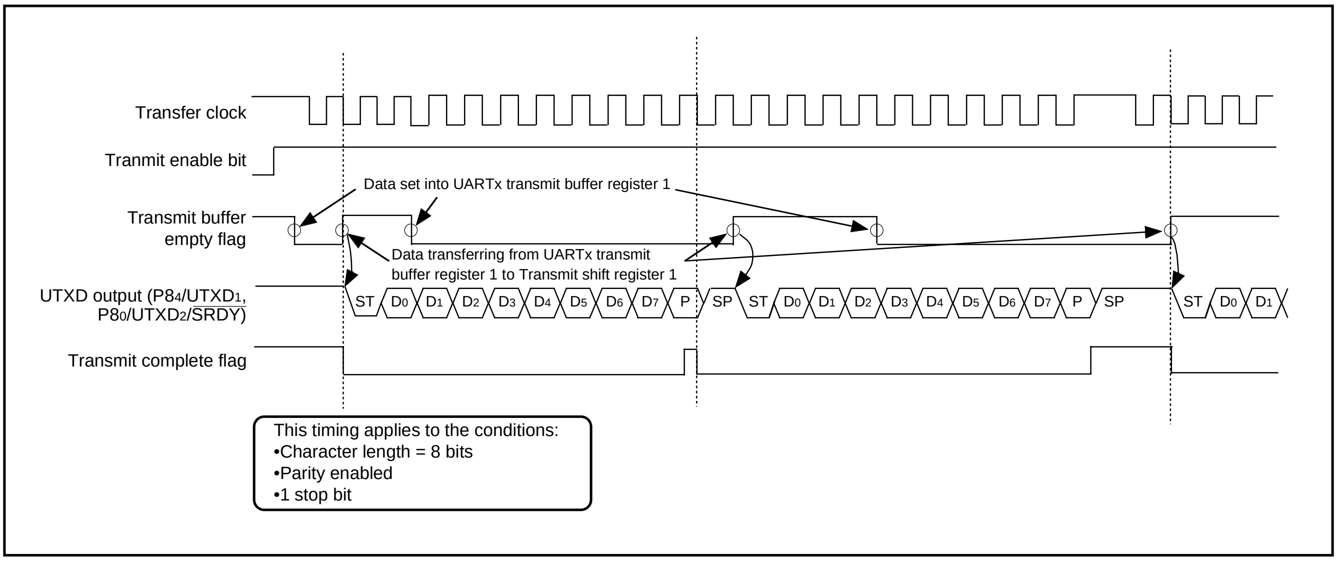


Fig. 28 UARTx transmit timing (CTS function disbled)

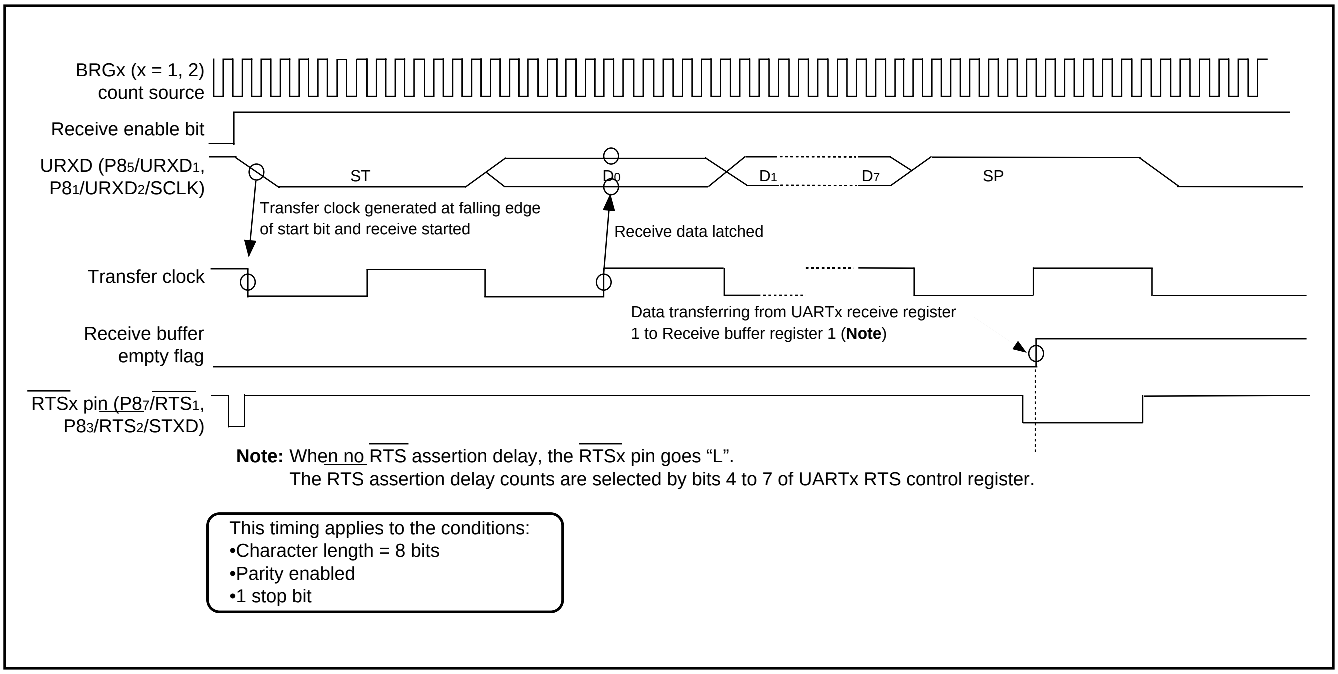


Fig. 29 UARTx transmit timing (RTS function enabled)

● **UART Address Mode**

The UART address mode is intended for use to communicate between the specified MCUs in a multi-MCU environment. The UART address mode can be used in either an 8-bit or 9-bit character length. An address is identified by the MSB of the incoming data being “1”. The bit is “0” for non-address data.

When the MSB of the incoming data is “0” in the UART address mode, the Receive Buffer Full Flag is set to “1”, but the Receive Buffer Full Interrupt Request Bit is not set to “1”. When the MSB of the incoming data is “1”, normal receive operation is performed. In the UART address mode an overrun error is not detected for reception of the 2nd and onward bytes. An occurrence of framing error or parity error sets the Summing Error Interrupt Request Bit to “1” and the data is not received independent of its MSB contents.

Usage of UART address mode is explained as follows:

(1) Set the UART Address Mode Enable Bit to “1”.

(2) Sends the address data of a slave MCU first from a host MCU to all slave MCUs. The MSB of address data must be “1” and the remaining 7 bits specify the address.

(3) The all slave MCUs automatically check for the received data whether its stop bit is valid or not, and whether the parity error occurs or not (when the parity enabled). If these errors occur, the Framing Error Flag or Parity Error Flag and the Summing Error Flag are set to “1”. Then, the Summing Error Interrupt Request Bit is also set to “1”.

(4) When received data has no error, the all slave MCUs must judge whether the address of the received address data matches with their own addresses by a program. After the MSB being “1” is received, the UART Address Mode Enable Bit is automatically set to “0” (disabled).

(5) The UART Address Mode Enable Bit of the slave MCUs which have be judged that the address does not match with them must be set to “1” (enabled) again by a program to disable reception of the following data.

(6) Transmit the data of which MSB is “0” from the host MCU. The slave MCUs disabling the UART address mode receive the data, and their Receive Buffer Full Flags and the Receive Buffer Full Interrupt Request Bits are set to “1”. For the other slave MCUs enabling the UART address mode, their Receive Buffer Full Flag are set to “1”, but their Receive Buffer Full Interrupt Request Bits are not set to “1”.

(7) An overrun error cannot be detected after the first data has been received in UART Address Mode. Accordingly, even if the slave MCUs does not read the received data and the next data has been received, an overrun error does not occur.

Thus, a communication between a host MCU and the specified MCU can be realized.

[UARTx (x = 1, 2) Mode Register (UxMOD)] 0030_{16}, 0038_{16}

The UART x mode register consists of 8 bits which set a transfer data format and an used clock.

[UARTx (x = 1, 2) Baud Rate Generator (UxBRG)] 0031_{16}, 0039_{16}

The UARTx baud rate generator determines the baud rate for transfer.

The baud rate generator divides the frequency of the count source by 1/(n + 1), where n is the value written to the baud rate generator.

The reset cannot affect the contents of baud rate generator.

[UARTx (x = 1, 2) Status Register (UxSTS)] 0032_{16}, $003A_{16}$

The read-only UARTx status register consists of seven flags (bits 0 to 6) which indicate the UART operating status and various errors.

When the UART address mode is enabled , the setting and clearing conditions of each flag differ from the following explanations. These differences are explained in section “**UART Address Mode**”.

•Transmit complete flag (TCM)

In the case where no data is contained in the transmit buffer register, the Transmit Complete Flag (TCM) is set to “1” when the last bit in the transmit shift register is transmitted.

The TCM flag is also set to “1” at reset or initialization by setting the Transmit Initialization Bit (bit 2 of UxCON). It is set to “0” when transmission starts, and it is kept during the transmission.

•Transmit buffer empty flag (TBE)

The Transmit Buffer Empty Flag (TBE) is set to “1” when the contents of the transmit buffer register are loaded into the transmit shift register. The TBE flag is also set “1” at the hardware reset or initialization by setting the Transmit Initialization Bit. It is set to “0” when a write operation is performed to the low-order byte of the transmit buffer register.

•Receive buffer full flag (RBF)

The Receive Buffer Full Flag (RBF) is set to “1” when the last stop bit of the data is received. The RBF flag is set to “0” when the low-order byte of the receive buffer register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit.

●**Receive Errors**

If there is an error, it is detected at the same time that data is transferred from the receive shift register to the receive buffer register, and the Receive Buffer Full Flag is set to "1". The all error flags PER, FER, OER and SER are cleared to "0" when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit.

The Summing Error Flag (SER) is set to "1" when any one of the PER, FER and OER is set to "1".

The Parity Error Flag (PER) is set to "1" when the sum total of 1s of received data and the parity does not correspond with the selection with the Parity Select Bit (PMD). It is enabled only if the Parity Enable Bit (bit 5 of UxMOD) is set to "1".

The Framing Error Flag (FER) is set to "1" when the number of stop bit of the received data does not correspond with the selection with the Stop Bit Length Select Bit (STB).

The Overrun Flag Flag (OER) is set to "1" if the previous data in the low-order byte of the receive buffer register 1 (addresses 0034_{16}, $003C_{16}$) is not read before the current receive operation is completed. It is also set "1" if any one of error flags is "1" for the previous data and the current receive operation is completed. Be sure to read UARTx status register to clear the error flags before the next reception has been completed.

[UARTx (x = 1, 2) Control Register (UxCON)] 0033_{16}, $003B_{16}$

The UARTx control register consists of eight control bits for the UARTx function. This register can enable the $\overline{CTS}$, $\overline{RTS}$ and UART address mode.

If the Transmit Enable Bit (TEN) is set to "0" (disabled) while a data is being transmitted, the transmitting operation will stop after the data has been transmitted. If the Receive Enable Bit (REN) is set to "0" (diabled) while a data is being received, the receiving operation will stop after the data has been received.

When setting the Transmit Initialization Bit (TIN) to "1", the TEN bit is set to "0" and the UARTx status register will be set to "03_{16}" after the data has been transmitted. To retransmit, set the TEN to "1" and set a data to the transmit buffer register again. The TIN bit will be cleared to "0" one cycle later after the TIN bit has been set to "1".

Setting the Receive Initialization Bit (RIN) to "1" sets all of the REN, RBF and the receive error flags (PER, FER, OER, SER) to "0". The RIN bit will be cleared to "0" one cycle later after the RIN bit has been set to "1".

When $\overline{CTS}$ or $\overline{RTS}$ function is disabled, pins $\overline{CTS_1}$ and $\overline{CTS_2}$ or $\overline{RTS_1}$ and $\overline{RTS_2}$ can be used as ordinary I/O ports, correspondingly.

[UARTx Transmit/Receive Buffer Registers 1, 2 (UxTRB1/UxTRB2)] 0034_{16}, 0035_{16}, $003C_{16}$, $003D_{16}$

The transmit buffer register and the receive buffer register are located at the same address. The transmit buffer register is write-only and the receive buffer register is read-only. If a character bit length is 7 bits, the MSB of received data is invalid. If a character bit length is 7 or 8 bits, the received contents of UxTRB2 are also invalid. If a character bit length is 9 bits, the received high-order 7 bits of UxTRB2 are "0".

[UARTx (x = 1, 2) RTS Control Register (UxRTS)] 0036_{16}, $003E_{16}$

The delay time from the reception of the last stop bit to the assertion of $\overline{RTSx}$ is selectable using the RTS Assertion Delay Count Select Bits. If the stop bit is detected before $\overline{RTS}$ assertion delay time has expired, the $\overline{RTSx}$ pin is kept "H". The $\overline{RTS}$ assertion delay count starts after the last data reception is completed.

Setting the RIN bit to "1" resets the UxRTS. After setting the RIN bit to "1", set this UxRTS.

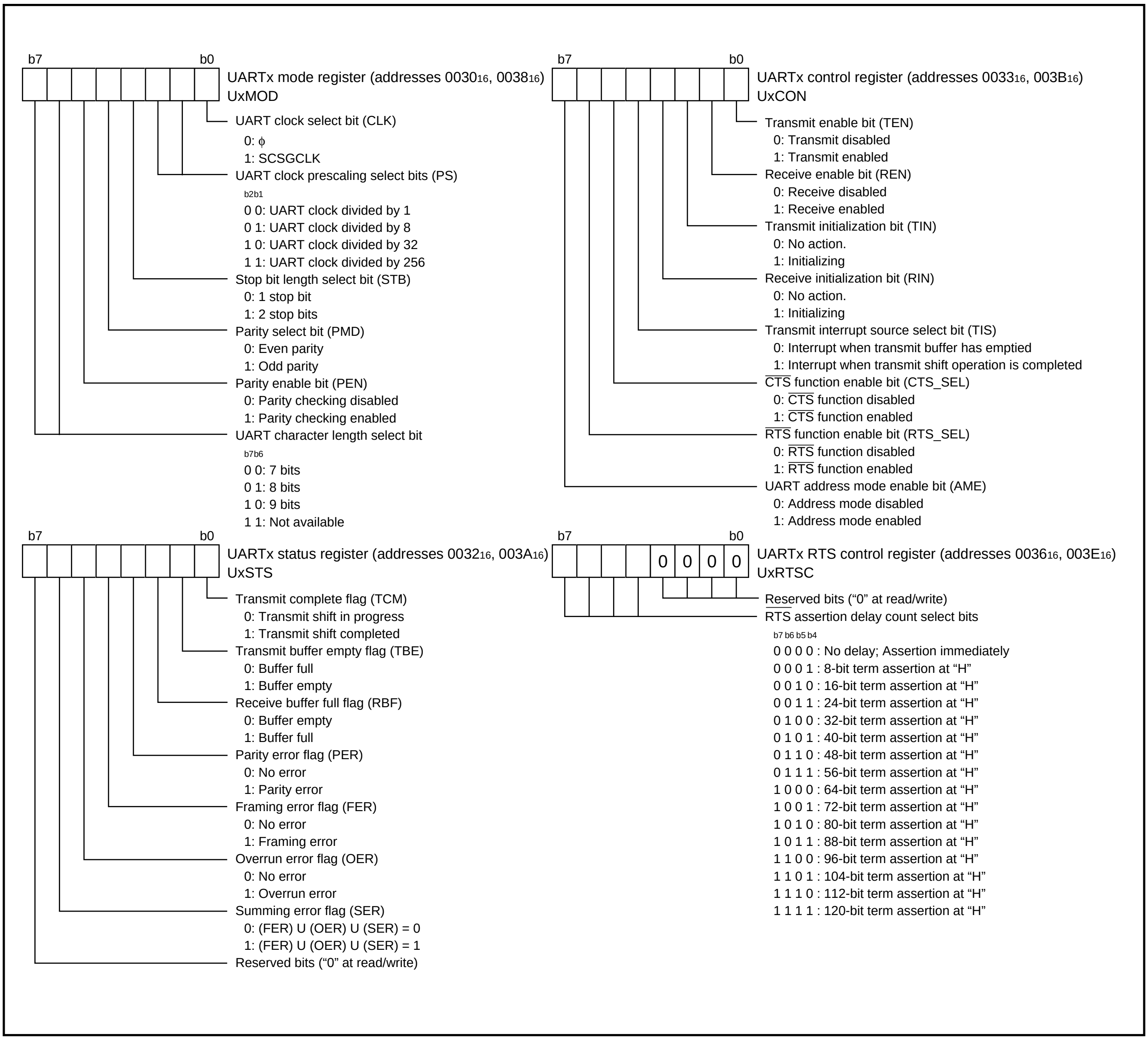


Fig. 30 Structure of UART related registers

DMAC

The 7641 group is equipped with 2 channels of DMAC (direct memory access controller) which enable high speed data transfer from a memory to a memory without use of the CPU.
The DMAC initiates the data transfer with an interrupt factor specified by the DMAC channel x (x = 0, 1) hardware transfer request source bit (DxHR), or with a software trigger.
The DxTMS [DMA Channel x (x = 0, 1) Transfer Mode Selection Bit] selects one of two transfer modes; cycle steal mode or burst transfer mode. In the cycle steal mode, the DMAC transfers one byte of data for each request. In the burst transfer mode, the DMAC transfers the number of bytes data specified by the transfer count register for each request. The count register is a 16-bit counter; the maximum number of data is 65,536 bytes per one request.
Figure 31 shows the DMA control block diagram and Figure 32 shows the structure of DMAC related registers.

[DMAC Index and Status Register] DMAIS

The DMAC Index and Status Register consists of various control bits for the DMAC and its status flags.
The DMA Channel Index Bit (DCI) selects which channel (0 or 1) will be accessed, since the mode registers, source registers, destination registers and transfer count register of both DMAC channels share the same SFR addresses, respectively.

[DMAC Channel x (x = 0, 1) Mode Registers 1, 2] DMAxM1, DMAxM2

The 16 bits of DMAC Channel x Mode Registers 1 and 2 control each operation of DMAC channels 0 and 1.
When the DMAC Channel x (x = 0, 1) Write Bit (DxDWC) is “0”, data is simultaneously written into each latch and register of the Source Registers, Destination Register, and Transfer Count Registers. When this bit is “1”, data is written only into their latches.
When data is read from each register, it must be read from the higher bytes first, then the lower bytes. When writing data, write to the lower bytes first, then the higher bytes.

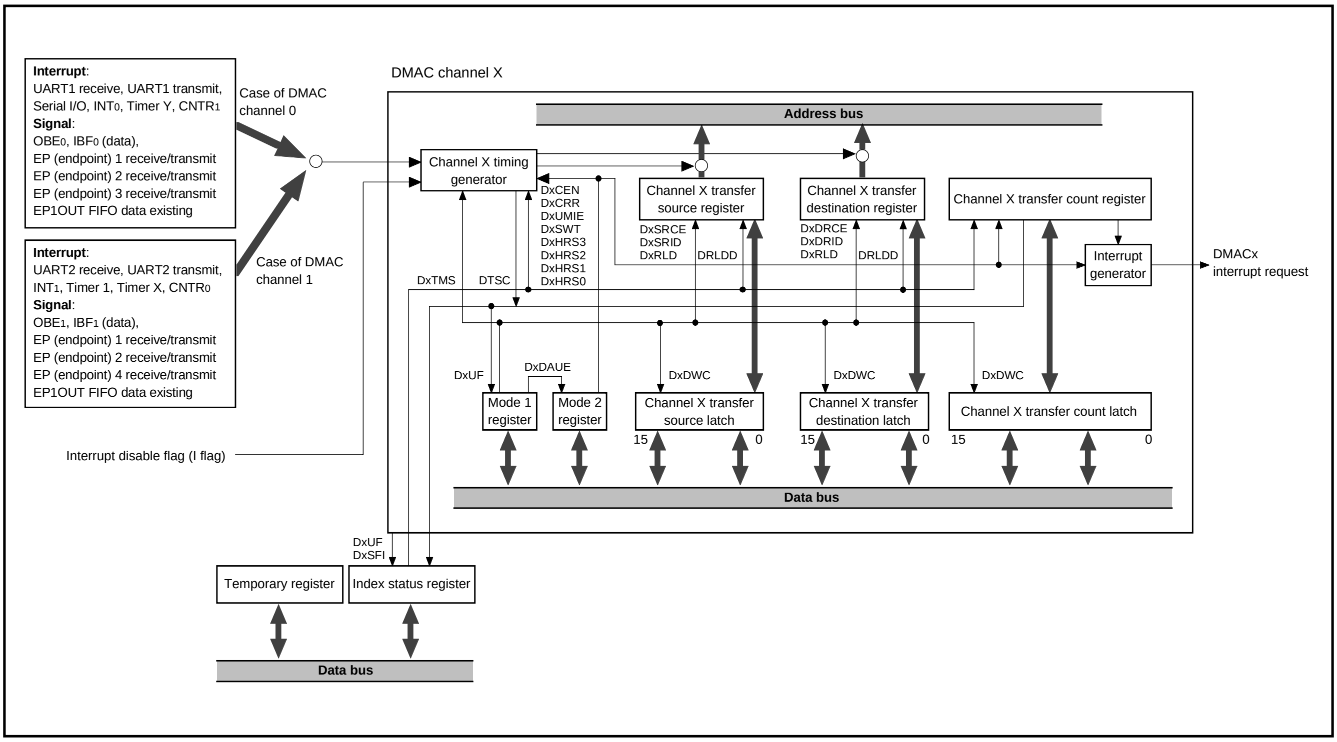


Fig. 31 DMACx (x = 0, 1) block diagram

b7 b0
DMAC index and status register (address $003F_{16}$)
DMAIS

- DMAC channel 0 count register underflow flag (D0UF)
  - 0: No underflow
  - 1: Underflow generated
- DMAC channel 0 suspend flag (D0SFI)
  - 0: Not suspended
  - 1: Suspended
- DMAC channel 1 count register underflow flag (D1UF)
  - 0: No underflow
  - 1: Underflow generated
- DMAC channel 1 suspend flag (D1SFI)
  - 0: Not suspended
  - 1: Suspended
- DMAC transfer suspend control bit (DTSC)
  - 0: Suspending only burst transfers during interrupt process
  - 1: Suspending both burst and cycle steal transfers during interrupt process
- DMAC register reload disable bit (DRLDD)
  - 0: Enabling reload of source and destination registers of both channels
  - 1: Disabling reload of source and destination registers of both channels
- Reserved bit (“0” at read/write)
- Channel index bit (DCI)
  - 0: Channel 0 accessible
  - 1: Channel 1 accessible

b7 b0
DMAC channel x mode register 1 (address 0040_{16})
DMAxM1

- DMAC channel x source register increment/decrement selection bit (DxSRID)
  - 0: Increment after transfer
  - 1: Decrement after transfer
- DMAC channel x source register increment/decrement enable bit (DxSRCE)
  - 0: Increment/Decrement disabled (No change after transfer)
  - 1: Increment/Decrement enabled
- DMAC channel x destination register increment/decrement selection bit (DxDRID)
  - 0: Increment after transfer
  - 1: Decrement after transfer
- DMAC channel x destination register increment/decrement enable bit (DxDRCE)
  - 0: Increment/Decrement disabled (No change after transfer)
  - 1: Increment/Decrement enabled
- DMAC channel x data write control bit (DxDWC)
  - 0: Writing data in reload latches and registers
  - 1: Writing data in reload latches only
- DMAC channel x disable after count register underflow enable bit (DxDAUE)
  - 0: Channel x enabled after count register underflow
  - 1: Channel x disabled after count register underflow
- DMAC channel x register reload bit (DxRLD)
  - 0: Not reloaded (Bit is always read as “0”)
  - 1: Source, destination, and transfer count registers contents of channel x to be reloaded
- DMAC channel x transfer mode selection bit (DxTMS)
  - 0: Cycle steal transfer mode
  - 1: Burst transfer mode

b7 b0
DMAC channel 0 mode register 2 (address 0041_{16})
DMA0M2

- DMAC channel 0 hardware transfer request source bits (D0HR)
  - b3b2b1b0
  - 0 0 0 0: Not used
  - 0 0 0 1: UART1 receive interrupt
  - 0 0 1 0: UART1 transmit interrupt
  - 0 0 1 1: Timer Y interrupt
  - 0 1 0 0: INT_0 interrupt
  - 0 1 0 1: USB endpoint 1 IN_PKT_RDY signal (falling edge active)
  - 0 1 1 0: USB endpoint 2 IN_PKT_RDY signal (falling edge active)
  - 0 1 1 1: USB endpoint 3 IN_PKT_RDY signal (falling edge active)
  - 1 0 0 0: USB endpoint 1 OUT_PKT_RDY signal (rising edge active)
  - 1 0 0 1: USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active)
  - 1 0 1 0: USB endpoint 2 OUT_PKT_RDY signal (rising edge active)
  - 1 0 1 1: USB endpoint 3 OUT_PKT_RDY signal (rising edge active)
  - 1 1 0 0: Master CPU bus interface OBE_0 signal (rising edge active)
  - 1 1 0 1: Master CPU bus interface IBF_0 signal, data (rising edge active)
  - 1 1 1 0: Serial I/O trasmit/receive interrupt
  - 1 1 1 1: $CNTR_1$ interrupt
- DMAC channel 0 software transfer trigger (D0SWT)
  - 0: No action (Bit is always read as “0”)
  - 1: Request of channel 0 transfer by writing “1” (**Note 1**)
- DMAC channel 0 USB and master CPU bus interface enable bit (D0UMIE)
  - 0: Disabled
  - 1: Enabled
- DMAC channel 0 transfer initiation source capture register reset bit (D0CRR)
  - 0: No action (Bit is always read as “0”)
  - 1: Reset of channel 0 capture register by writing “1” (**Note 1**)
- DMAC channel 0 enable bit (D0CEN)
  - 0: Channel 0 disabled
  - 1: Channel 0 enabled (**Note 2**)

b7 b0
DMAC channel 1 mode register 2 (address 0041_{16})
DMA1M2

- DMAC channel 1 hardware transfer request source bits (D1HR)
  - b3b2b1b0
  - 0 0 0 0: Not used
  - 0 0 0 1: UART2 receive interrupt
  - 0 0 1 0: UART2 transmit interrupt
  - 0 0 1 1: Timer X interrupt
  - 0 1 0 0: INT_1 interrupt
  - 0 1 0 1: USB endpoint 1 IN_PKT_RDY signal (falling edge active)
  - 0 1 1 0: USB endpoint 2 IN_PKT_RDY signal (falling edge active)
  - 0 1 1 1: USB endpoint 4 IN_PKT_RDY signal (falling edge active)
  - 1 0 0 0: USB endpoint 1 OUT_PKT_RDY signal (rising edge active)
  - 1 0 0 1: USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active)
  - 1 0 1 0: USB endpoint 2 OUT_PKT_RDY signal (rising edge active)
  - 1 0 1 1: USB endpoint 4 OUT_PKT_RDY signal (rising edge active)
  - 1 1 0 0: Master CPU bus interface OBE_1 signal (rising edge active)
  - 1 1 0 1: Master CPU bus interface IBF_1 signal, data (rising edge active)
  - 1 1 1 0: Timer 1 trasmit/receive interrupt
  - 1 1 1 1: $CNTR_0$ interrupt
- DMAC channel 1 software transfer trigger (D1SWT)
  - 0: No action (Bit is always read as “0”)
  - 1: Request of channel 0 transfer by writing “1” (**Note 1**)
- DMAC channel 1 USB and master CPU bus interface enable bit (D1UMIE)
  - 0: Disabled
  - 1: Enabled
- DMAC channel 1 transfer initiation source capture register reset bit (D1CRR)
  - 0: No action (Bit is always read as “0”)
  - 1: Reset of channel 1 capture register by writing “1” (**Note 1**)
- DMAC channel 1 enable bit (D1CEN)
  - 0: Channel 0 disabled
  - 1: Channel 0 enabled (**Note 2**)

Notes 1: This bit is automatically cleared to “0” after writing “1”.
2: When setting this bit to “1”, simultaneously set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of DMAxM2) to “1”.

Fig. 32 Structure of DMACx related register

(1) Cycle Steal Transfer Mode

When the DMAC Channel x (x = 0, 1) Transfer Mode Selection Bit (DxTMS) is set to “0”, the respective DMAC Channel x operates in the cycle steal transfer mode.

When a request of the specified transfer factor is generated, the selected channel transfers one byte of data from the address indicated by the Source Register into the address indicated by the Destination Register.

There are two kinds of DMA transfer triggers supported: hardware transfer factor and software trigger. Hardware transfer factors can be selected by the DMACx (x = 0, 1) Hardware Transfer Request Factor Bit (DxHR). To only use the Interrupt Request Bit, the interrupt can be disabled by setting its Interrupt Enable Bit of Interrupt Control Register to “0”.

The DMA transfer request as a software trigger can be generated by setting the DMA Channel x (x = 0, 1) Software Transfer Trigger Bit (DxSWT) to “1”.

The Source Registers and Transfer Destination Registers can be either decreased or increased by 1 after transfer completion by setting bits 0 to 3 in the DMAC Channel x (x = 0, 1) Mode Register. When the Transfer Count Register underflows, the Source Registers and Destination Registers are reloaded from their latches if the DMAC Register Reload Disable Bit (DRLDD) is “0”.

The Transfer Count Register value is reloaded after an underflow regardless of DRLDD setting. At the same time, the DMAC Interrupt Request Bit and the DMA Channel x (x = 0, 1) Count Register Underflow Flag are set to “1”.

The DMAC Channel x Disable After Count Register Underflow Enable Bit (DxDAUE) is “1”, the DMAC Channel x Enable Bit (DxCEN) goes to “0” at an under flows of Transfer Count Register.

By setting the DMAC Channel x (x = 0, 1) Register Reload Bit (DxRLD) to “1”, the Source Registers, Destination Registers, and Transfer Count Registers can be updated to the values in their respective latches.

When one signal among USB endpoint signals is selected as the hardware transfer request factor, and DMAC Channel x (x = 0, 1) USB and Master CPU Bus Interface Enable Bit (DxUMIE) is “1”; transfer between the USB FIFO and the master CPU bus interface input/output buffer can be performed effectively. This transfer function is only valid in the cycle steal mode. To validate this function, the DMAC Channel x (x = 0, 1) USB and the Master CPU Bus Interface Enable Bit (bit 5 of DxTR) must be set to “1”. The following shows an example of a transfer using this function.

Packet Transfer from USB FIFO to Master CPU Bus Interface Buffer

When the USB OUT_PKT_RDY is selected as the hardware transfer request factor; if the USB OUT_PKT_RDY is “1” and the master CPU bus interface output buffer is empty, the transfer request is generated and the transfer is initiated. The OUT_PKT_RDY retains “1” and a transfer request is generated each time the output buffer empties until all the data in the corresponding endpoint FIFO has been transferred.

The transfer ends when the last byte in the USB receive packet is transferred and the OUT_PKT_RDY flag goes to “0” (in the case of AUTO_CLR bit = “1”).

Byte Transfer from USB FIFO to Master CPU Bus Interface Buffer

When the USB Endpoint 1 OUT_FIFO_NOT_EMPTY is selected as a hardware transfer request factor, if there is data in the USB Endpoint 1 FIFO and the master CPU bus interface output buffer is empty; a transfer request is generated and the transfer is initiated. The transfer is performed by unit of one byte.

Transfer from Master CPU Bus Interface Buffer to USB FIFO

When the USB Endpoint X (X = 1 to 4) IN_PKT_RDY (IN_PKT_RDY = “0”) is selected as a hardware transfer request factor, if there is data in the master CPU bus interface output buffer and the data in the USB FIFO is within the specified packet size, a transfer request is generated.

The DMA transfer is terminated when a command (A0 = “1”) is input to the master CPU bus interface input buffer.

The timing chart for a cycle steal transfer caused by a hardware-related transfer request and a software trigger are shown in Figure 33 and 34, respectively.

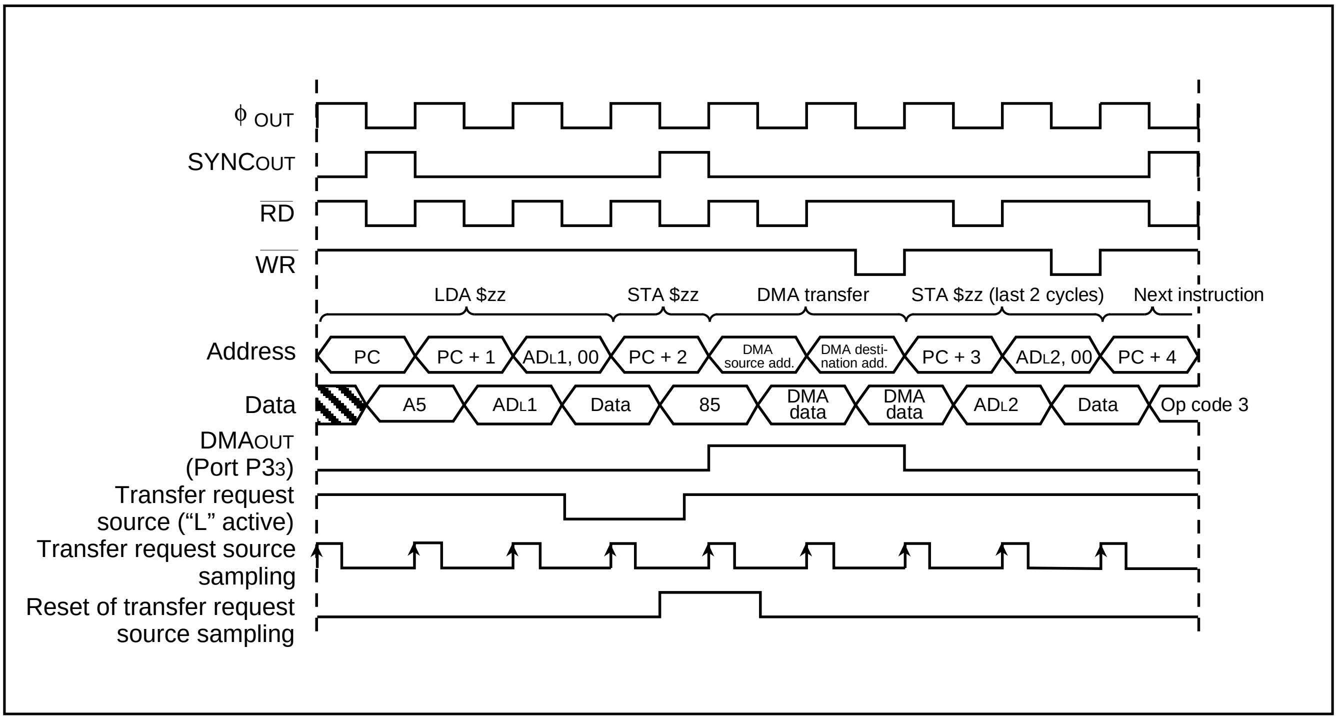


Fig. 33 Timing chart for cycle steal transfer caused by hardware-related transfer request

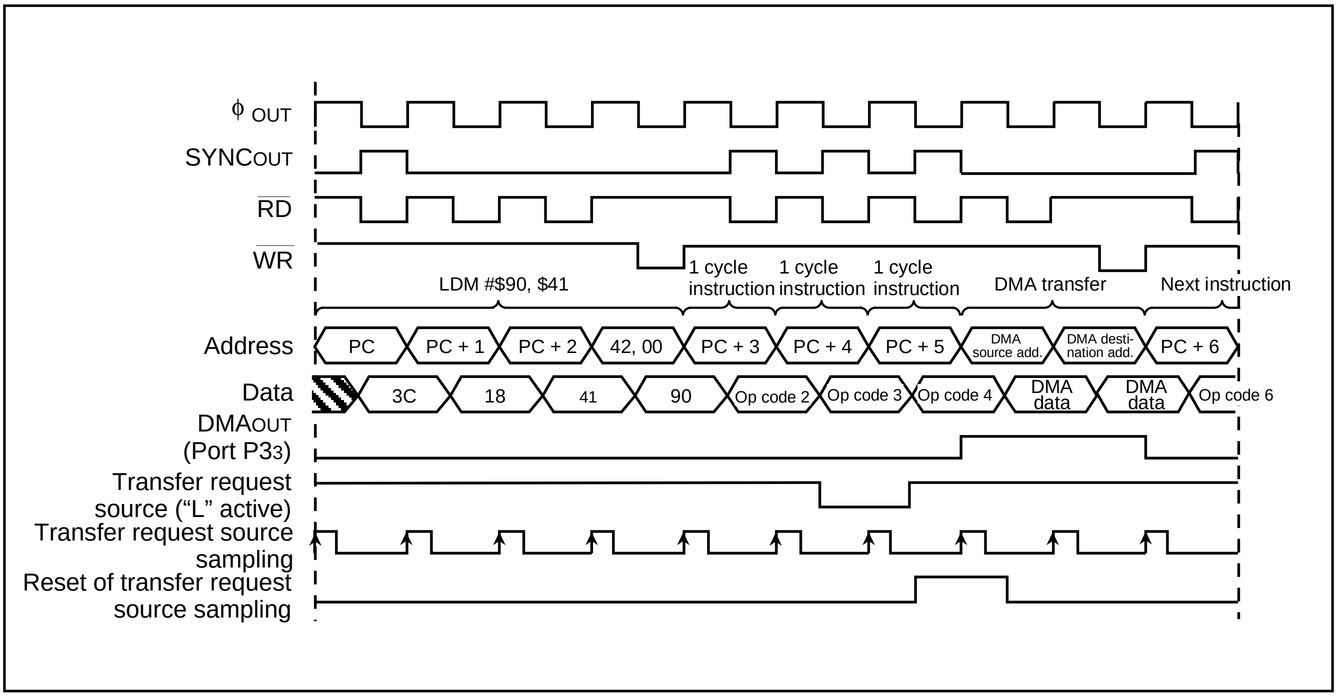


Fig. 34 Timing chart for cycle steal transfer caused by software trigger transfer request

(2) Burst Transfer Mode

When the DMAC Channel x Transfer Mode Selection Bit (DxTMS) is set to “1”, the respective DMAC channel operates in the burst transfer mode.
In the burst transfer mode, the DMAC continually transfers the number of bytes of data specified by the Transfer Count Register for one transfer request. Other than this, the burst transfer mode operation is the same as the cycle steal mode operation.

Priority

The DMAC places a higher priority on Channel-0 transfer requests than on Channel-1 transfer requests.
If a Channel-0 transfer request occurs during a Channel-1 burst transfer operation, the DMAC completes the next transfer source and destination read/write operation first, and then starts the Channel-0 transfer operation. As soon as the Channel-0 transfer is completed, the DMAC resumes the Channel-1 transfer operation.

When an interrupt request occurs during any DMA operation, the transfer operation is suspended and the interrupt process routine is initiated. During the interrupt operation, the DMAC automatically sets the corresponding DMAC Channel x (x = 0, 1) Suspend Flag (DxSFI) to “1”. As soon as the CPU completes the interrupt operation, the DMAC clears the flag to “0” and resumes the original operation from the point where it was suspended.
The suspended transfer due to the interrupt can also be resumed during its interrupt process routine by writing “1” to the DMAC Channel x (x = 0,1) Enable Bit (DxCEN).

The timing charts for a burst transfer caused by a hardware-related transfer request are shown in Figure 35.

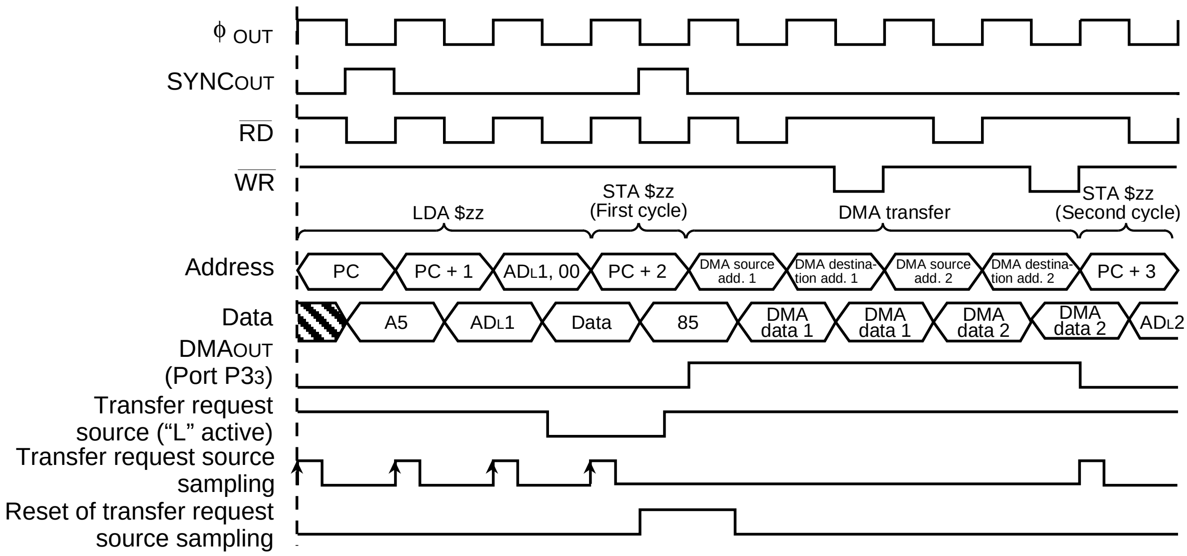


Fig. 35 Timing chart for burst transfer caused by hardware-related transfer request

USB FUNCTION

The 7641 Group MCU is equipped with a USB Function Control Unit (USB FCU). This USB FCU allows the MCU to communicate with a host PC using a minimum amount of the MCU power. This built-in USB FCU complies with Full-Speed USB2.0 specification that supports four transfer types: Control Transfer, Isochronous Transfer, Interrupt Transfer, and Bulk Transfer. This built-in USB FCU performs the data transfer error detection and transfer retry operation by hardware. The default transfer mode of the USB FCU is bulk transfer mode at reset. The user must set the USB FCU for the required transfer mode by software.

The USB FCU has five endpoints (Endpoint 0 to Endpoint 4). The EPINDEX bit selects one of these five endpoints for the USB FCU to use. Each endpoint has IN (transmit) FIFO and OUT (receive) FIFO. To use the USB FCU, the USB enable bit (USBC7) must be set to “1”. There are two USB related interrupts supported for this MCU: USB Function Interrupt and USB SOF Interrupt.

Figure 36 shows the USB FCU (USB Function Control Unit) block diagram. The USB FCU consists of the SIE (Serial Interface Engine) performing the USB data transfer, GFI (Generic Function Interface) performing USB protocol handing, SIU (Serial Engine Interface Unit) performing a received address and endpoint decoding, MCI (Microcontroller Interface) handling the MCU interface or performing address decoding and synchronization of control signals, and the USB transceiver.

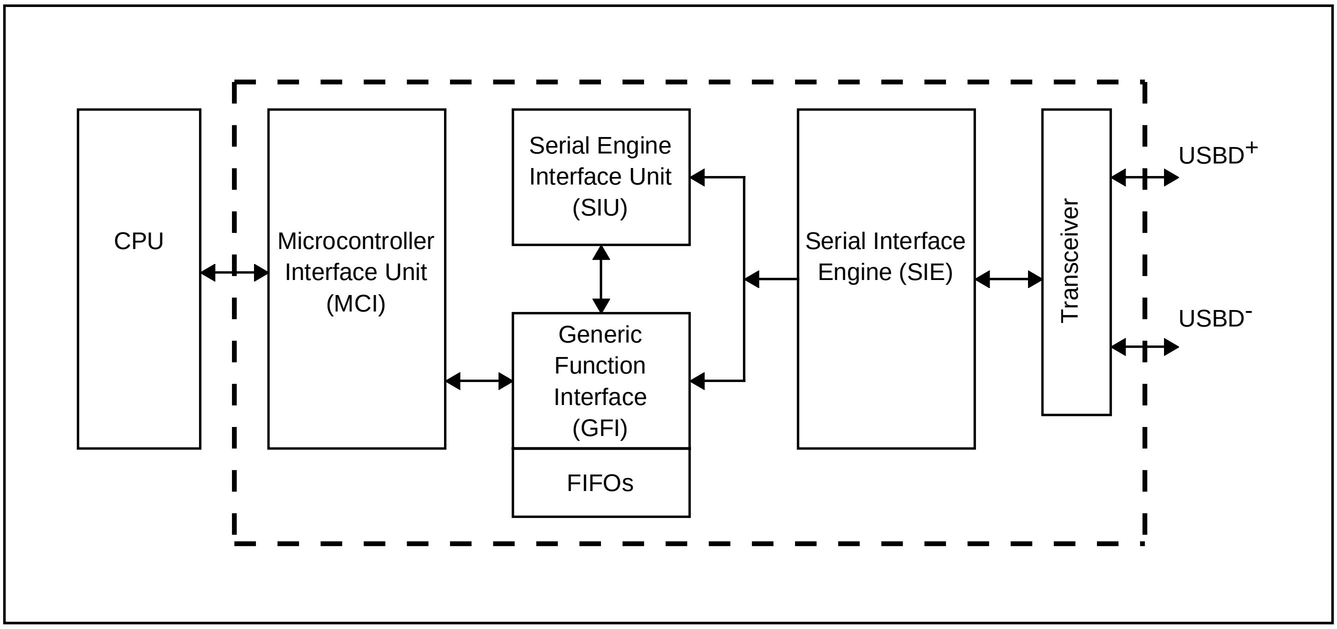


Fig. 36 USB FCU (USB Function Control Unit) block

USB Transmission

Endpoint 0 to Endpoint 4 have IN (transmit) FIFOs individually. Each endpoint's FIFO is configured in following way:

Endpoint 0: 16-byte
Endpoint 1: Mode 0: 512-byte
Mode 1: 1024-byte
Mode 2: 0-byte
Mode 3: 2048-byte
Mode 4: 768-byte
Mode 5: 880-byte
Endpoint 2: Mode 0: 32-byte
Mode 1: 128-byte
Endpoint 3: 16-byte
Endpoint 4: 16-byte

When Endpoint 1 or Endpoint 2 is used for data transmit, the IN FIFO size can be selected. Endpoint 1 and Endpoint 2 have programmable IN-FIFOs size; 6 modes for Endpoint 1, and 2 modes for Endpoint 2. Each mode can be selected by the USB endpoint FIFO mode selection register (address $005F_{16}$).

When writing data to the USB Endpoint-x FIFO (addresses 0060_{16} to 0064_{16}) in the SFR area, the internal write pointer for the IN FIFO is automatically increased by 1. When the AUTO_SET bit is "1" and if the stored data reaches to the max. packet value set in USB Endpoint x IN max. packet size register (address $005B_{16}$), the USB FCU sets the IN_PKT_RDY bit to "1". When the AUTO_SET bit is "0", the IN_PKT_RDY bit will not be automatically set to "1"; it must be set to "1" by software. (The AUTO_SET bit function is not applicable to Endpoint 0.)

The USB FCU transmits the data when it receives the next IN token. The IN_PKT_RDY bit automatically goes to "0" when the data transfer is complete.

●Isochronous transfer

Endpoints 1 to 4 can be used in isochronous transfer mode. When using isochronous transfer mode, the ISO/TOGGLE_INIT bit must be set to "1". When ISO_UPDATE = "1" and the corresponding endpoint's ISO/TOGGLE_INIT bit = "1", the USB FCU delays the rise of the IN_PKT_RDY bit until the next SOF signal transmission. In this way, the USB FCU can synchronize a transmit data to the SOF signal.

●Interrupt transfer mode

Endpoints 1 to 4 can be used in interrupt transfer mode. During a regular interrupt transfer, an interrupt transaction is similar to the bulk transfer. Therefore, there is no special setting required. When IN-endpoint is used for a rate feedback interrupt transfer, INTPT bit of the IN_CSR register must be set to "1". The following steps show how to configure the IN-endpoint for the rate feedback interrupt transfer.

1. Set a value which is larger than 1/2 of the USB Endpoint-x FIFO size to the USB Endpoint x IN max. package size register.
2. Set INTPT bit to "1".
3. Flush the old data in the FIFO.
4. Store transmission data to the IN FIFO and set the IN_PKT_RDY bit to "1".
5. Repeat steps 3 and 4.

In a real application, the function-side always has transfer data when the function sends an endpoint in a rate feedback interrupt. Accordingly, the USB FCU never returns a NAK against the host IN token for the rate feedback interrupt. The USB FCU always transmits data in the FIFO in response to an IN token, regardless of IN_PKT_RDY. However, this premises that there is always an ACK response from Host PC after the 7641 Group has transmitted data to IN token.

When MAXP size ≤ (a half of IN FIFO size), the IN FIFO can store two packets (called double buffer). At this time, the IN FIFO status can be checked by monitoring the IN_PKT_RDY bit and the TX_NOT_EPT flag. The TX_NOT_EPT flag is a read-only flag which shows the FIFO state. When IN_PKY_RDY = 0 and TX_NOT_EPT = 0, IN FIFO is empty. When IN_PKY_RDY = 0 and TX_NOT_EPT = 1, IN FIFO has one packet.

In double buffer mode, as long as the IN FIFO is not filled with double packets, IN_PKT_RDY will not be set to "1", even if it is set to "1" by software, but TX_NOT_EPT flag will be set to "1". In single buffer mode, if MAXP > (a half of IN FIFO), this condition never occurs.

When IN_PKT_RDY = "1" and TX_NOT_EPT = "1", IN FIFO holds two packets in double buffer mode and one packet in single packet mode. In single packet mode, when the IN_PKT_RDY bit is set to "1" by software, the TX_NOT_EPT flag is set to "1" as well. During double buffer mode, if you want to load two packets sequentially, you must set the IN_PKT_RDY bit to "1" each time a packet is loaded.

USB Reception

Endpoint 0 to Endpoint 4 have OUT (receive) FIFOs individually. Each endpoint's FIFO is configured in following way:

Endpoint 0: 16-byte
Endpoint 1: Mode 0: 800-byte
Mode 1: 1024-byte
Mode 2: 2048-byte
Mode 3: 0-byte
Mode 4: 1280-byte
Mode 5: 1168-byte
Endpoint 2: Mode 0: 32-byte
Mode 1: 128-byte
Endpoint 3: 16-byte
Endpoint 4: 16-byte

When Endpoint 1 or Endpoint 2 is used for data receive, the OUT FIFO size can be selected. Endpoint 1 and Endpoint 2 have programmable IN-FIFOs size; 6 modes for Endpoint 1, and 2 modes for Endpoint 2. Each mode can be selected by the USB endpoint FIFO mode selection register (address $005F_{16}$).

Data transmitted from the host-PC is stored in Endpoint x FIFO (0060_{16} to 0064_{16}). Every time the data is stored in the FIFO, the internal OUT FIFO write pointer is increased by 1. When one complete data packet is stored, the OUT_PKT_RDY flag is set to "1" and the number of received data packets is stored in USB Endpoint x OUT write count registers (Low and High). When the AUTO_CLR bit is "1" and the received data is read out from the OUT FIFO, the OUT_PKT_RDY flag is cleared to "0". When the AUTO_CLR bit is "1", the OUT_PKT_RDY flag will not be cleared automatically by the FIFO read; it must be cleared by software. (The AUTO-CLR bit function is not applicable in Endpoint 0.)

When MAXP size ≤ (a half of OUT FIFO size), the OUT_FIFO can receive 2 packets (double buffer). At this time, the OUT_ FIFO status can be checked by the OUT_PKT_RDY flag. When the FIFO holds two packets and one packet is read from the FIFO, the OUT_PKT_RDY flag is not cleared even if it is set to "0". (The flag returns from "0" to "1" in one ϕ cycle after the read-out). During double buffer mode, the USB Endpoint x OUT write count registers (Low and High) holds the number of previously received packets. This count register is updated after reading out one of packets in the OUT FIFO and clearing the OUT_PKT_RDY flag to "0".

TOGGLE Initialization

In order to initialize the data toggle sequence bit of the endpoint, in other words, resetting the next data packet to DATA0; set the ISO/TOGGLE_INT bit to "1" and then clear back to "0".

USB Interrupts

The USB FCU has two interrupts, USB Function Interrupt and USB SOF (Start Of Frame) Interrupt.

●USB Function Interrupt (USBF-INT)

The USBF-INT is usable for the USB data flow control and power management. The USBF-INT request occurs at data transmit/receive completion, overrun/underrun, reset, or receiving suspend/resume signal. To enable this interrupt, the USB function interrupt enable bit in the interrupt control register A (address 0005_{16}) and the respective bit in the USB interrupt enable registers 1 and 2 (addresses 0054_{16} and 0055_{16}) must be set to “1”. When setting bit 7 in USB interrupt enable register 2 to “1”, the suspend interrupt and the resume interrupt are enabled.

Endpoint x (x = 0 to 4) IN interrupt request occurs when the USB Endpoint x IN interrupt status flag (INTST 0, 2, 4, 6, 8) of USB interrupt status registers 1 and 2 (addresses 0052_{16} and 0053_{16}) is “1”. The USB Endpoint x IN interrupt status flag is set to “1” when the respective endpoint IN_PKT_RDY bit is “1”.
Endpoint x (x = 0 to 4) OUT interrupt request occurs when the USB endpoint x OUT interrupt status flag (INTST3, 5, 7, 9) in USB interrupt status registers 1 and 2 is set to “1”. The USB Endpoint x OUT interrupt status flag is set to “1” when the respective endpoint OUT_PKT_RDY flag is “1”.

The overrun/underrun interrupt request occurs when the USB overrun/underrun interrupt status flag (INTST12) in USB interrupt status register 2 is set to “1”. This flag is set to “1” when the FIFO data overruns or underruns in isochronous transfer mode.
The USB reset interrupt request occurs when the USB reset interrupt status flag (INTST13) in USB interrupt status register 2 is set to “1”. This flag is set when the SE0 is detected on the D+/D- line for at least 2.5 µs. When this situation happens, all USB internal registers (addresses 0050_{16} to $005F_{16}$), except this flag, are initialized to the default state at reset. The USB reset interrupt is always enabled.

The suspend/resume interrupt request occurs when either the USB resume signal interrupt status flag (INTST14) or the USB suspend signal interrupt status flag (INTST15) in USB interrupt status register 2 is set to “1”.
The bits in both interrupt status registers 1 and 2 can be cleared by writing “1” to each bit.

●USB SOF interrupt

The USB SOF interrupt is usable in isochronous transfers. This interrupt request occurs when an SOF packet is received. To enable a USB SOF interrupt, set the USB SOF interrupt enable bit of interrupt control register A to “1”.

Suspend/Resume Functions

If no bus activity is detected on the D+/D- line for at least 3 ms, the USB suspend signal detect flag (SUSPEND) of the USB power control register (address 0051_{16}) and the USB suspend signal interrupt status flag of USB interrupt status register 2 are set to “1” and the suspend interrupt request occurs. The following procedure must be executed after pushing the internal registers (A, X, Y) to memories during the suspend interrupt process routine.

(1) Clear all bits of USB interrupt status register 1 (address 0052_{16}) and USB interrupt status register 2 (address 0053_{16}) to “0”.
(2) Set the USB clock enable bit to “0”. (After disabling the USB clock, do not write to any of the USB internal registers (addresses 0050_{16} to 0064_{16}), except for the USB control register (address 0013_{16}), clock control register (address $001F_{16}$), and frequency synthesizer control register (address $006C_{16}$).
(3) Set the frequency synthesizer enable bit to “0”.
(4) Set the USB line driver current control bit to “1”. (Always keep the USB line driver current control bit set to “0” during USB function operations. When operating at Vcc = 3.3 V, this bit does not need to be set.)
(5) Keep total drive current at 500 µA or less.
(6) Disable the timer 1 interrupt.
(7) Disable the timer 2 interrupt. (Disable all the other external interrupts.)
(8) Set the timer 1 interrupt request bit to “0”.
(9) Set the timer 2 interrupt request bit to “0”.
(10) Set the interrupt disable flag (I) to “0”.
(11) Execute the STP instruction.

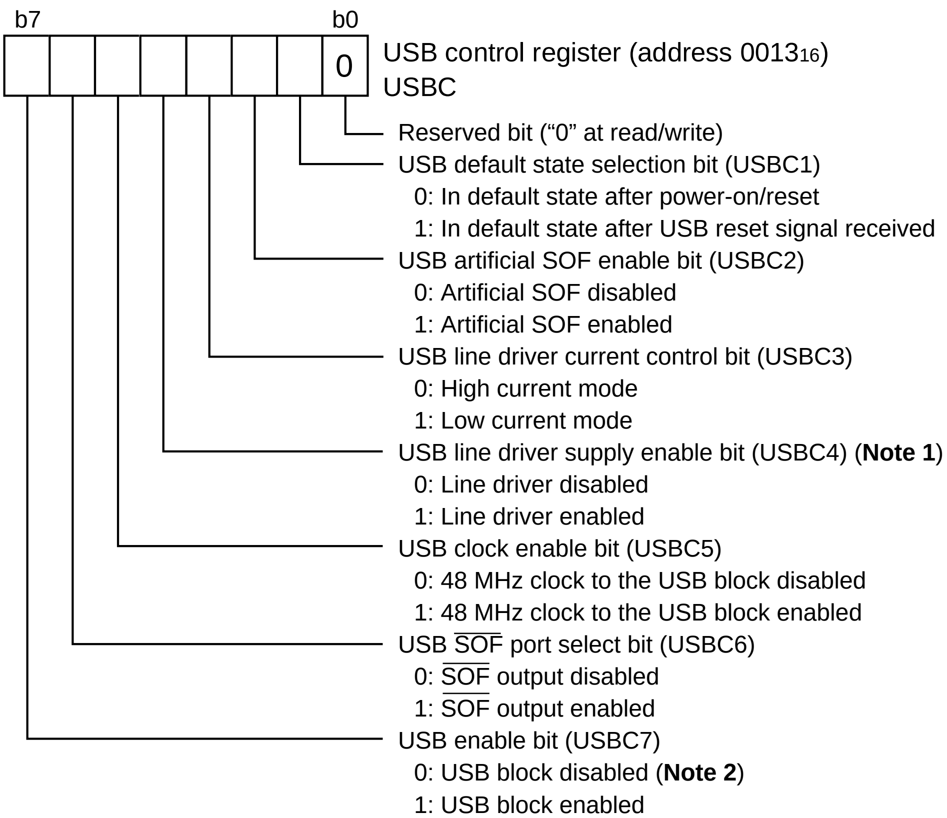
At this point, the MCU will be in stop mode (suspend mode). Before executing the STP instruction, make sure to set the USB function interrupt request bit (bit 0 at address 0002_{16}) to “0” and the USB function interrupt enable bit (bit 0 at address 0005_{16}) to “1”.

The USB suspend detect signal flag goes to “0” when the USB resume signal detect flag (RESUME) is set to “1”. During suspend mode, if the clock operation is started up with a process (remote wake-up) other than the resume interrupt process (for example; reset or timer), make sure to clear the USB suspend detect signal flag to “0” when you set the USB remote wake-up bit to “1”. When the USB FCU is in suspend mode and detects a non-idle signal on the D+/D- line, the USB resume detect flag and the USB resume signal interrupt status flag both go to “1” and a resume interrupt request occurs. At this point, pull the internal registers (A, X, Y) in this interrupt process routine. Take the following procedure in the USB resume interrupt process.

(1) Set the USB line driver current control bit to “0”. (When operating at Vcc = 3.3 V, this bit does not need to be set.)
(2) Set the frequency synthesizer enable bit to “1” and set a 2 ms to 5 ms wait.
(3) Check the frequency synthesizer lock status bit. If “0”, it must be checked again after a 0.1 ms wait.
(4) Enable the USB clock.

Set the USB resume signal interrupt status flag to “0” after the wake-up sequence process. The USB resume detect flag goes to “0” at the same time. When the clock operation is started up with a remote wake-up, set the USB remote wake-up bit to “1” after the wake-up sequence process. (keep it set to “1” for a minimum of 10 ms and maximum of 15 ms). By doing this, the MCU will send a resume signal to the host CPU and let it know that the suspend state has been released.
After that, set the USB remote wake-up bit and the USB suspend detection flag to “0”, because the USB suspend detection flag is not automatically cleared to “0” with a remote wake-up.

[USB Control Register] USBC
When using the USB function, the USB enable bit must be set to “1”. The USB line driver supply bit must be set to “0” (DC-DC converter is disabled) when operating at Vcc = 3.3V. In this condition, the setting of the USB line driver current control bit has no effect on USB operations.
When the USB artificial SOF enable bit is set to “1”, the MCU judges that a SOF packet is received within 250 ns from a frame starting if an SOF packet is destroyed owing to some cause.



Notes 1: When using the MCU in Vcc = 3.3 V, set this bit to “0” and disable the built-in DC-DC converter
2: Setting this bit to 0” causes the contents of all USB registers to have the values at reset.

Fig. 37 Structure of USB control register

[USB Address Register] USBA

The USB address register maintains the USB function control unit address assigned by the host computer. When receiving the SET_ADDRESS, keep it in this register. The values of this register are “0” when the device is not yet configured. The values of this register are also set to “0” when the USB block is disabled (bit 7 of USB control register is set to “0”). In addition, no matter what value is written to this register, it will have no effect on the set value.

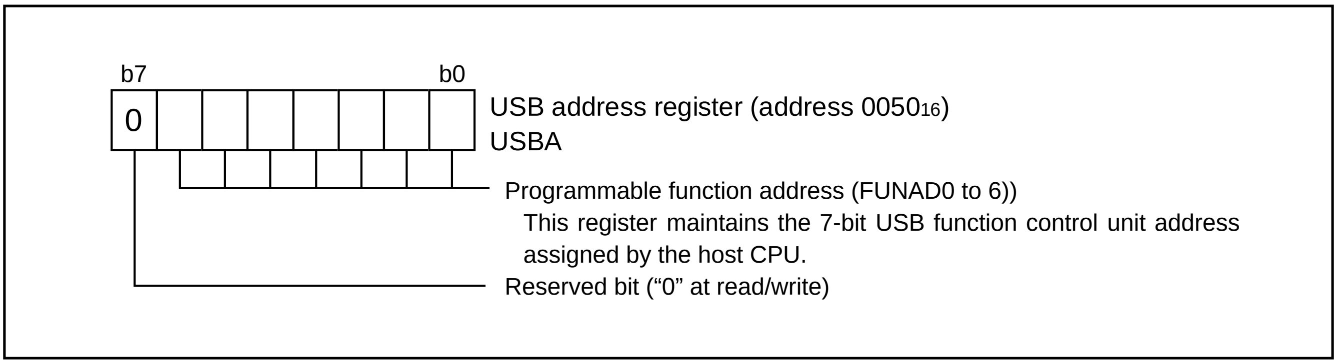


Fig. 38 Structure of USB address register

[USB Power Management Register] USBPM

The USB power management register is used for power management in the USB FCU. This register needs to be set only when using the remote wake-up to resume the MCU from suspend mode.

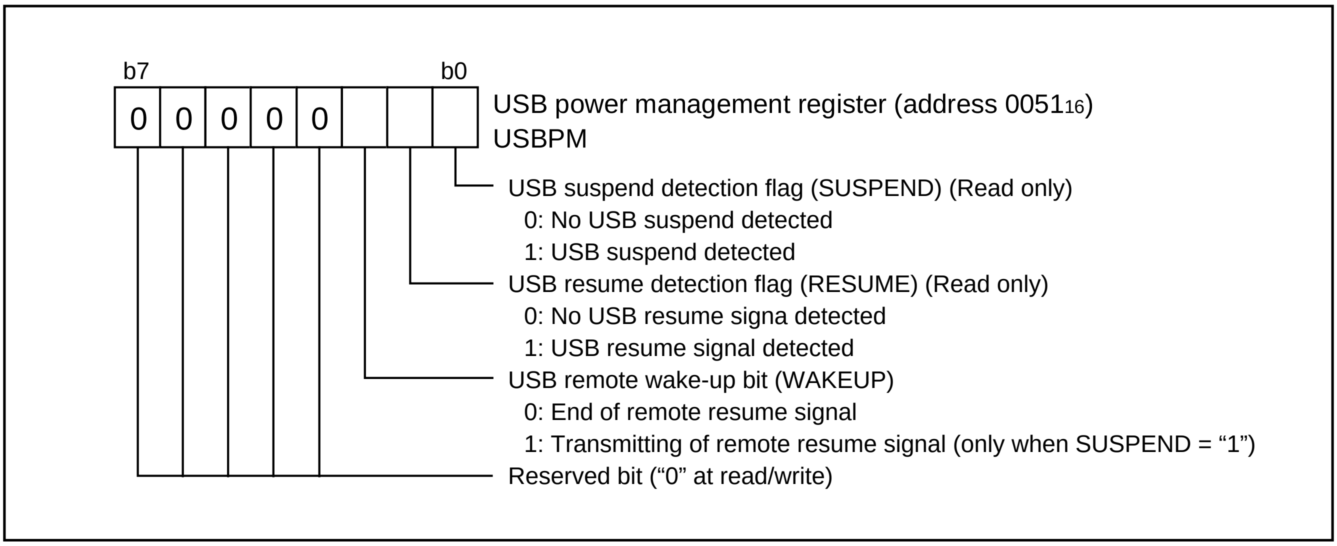


Fig. 39 Structure of USB power management register

[USB Interrupt Status Registers 1 and 2] USBIS1, USBIS2

The USB interrupt status registers are used to indicate the condition that caused a USB function interrupt to be generated. Each status flag and bit can be cleared to “0” by writing “1” to the corresponding bit. Make sure to write to/read from the USB interrupt status register 1 first and then USB interrupt status register 2.

When an IN token is received during an isochronous transfer, and the IN FIFO is empty, an underrun error occurs and INTST12 and IN_CSR2 are set to “1”. When an OUT token is received and the OUT FIFO is full, an overrun error occurs and INTST12 and OUT_CSR2 are set to “1”. Underruns and overruns are not detected by the CPU in bulk transfers and normal interrupt transfers, however in this case, the MCU will send a NAK signal to the host CPU.

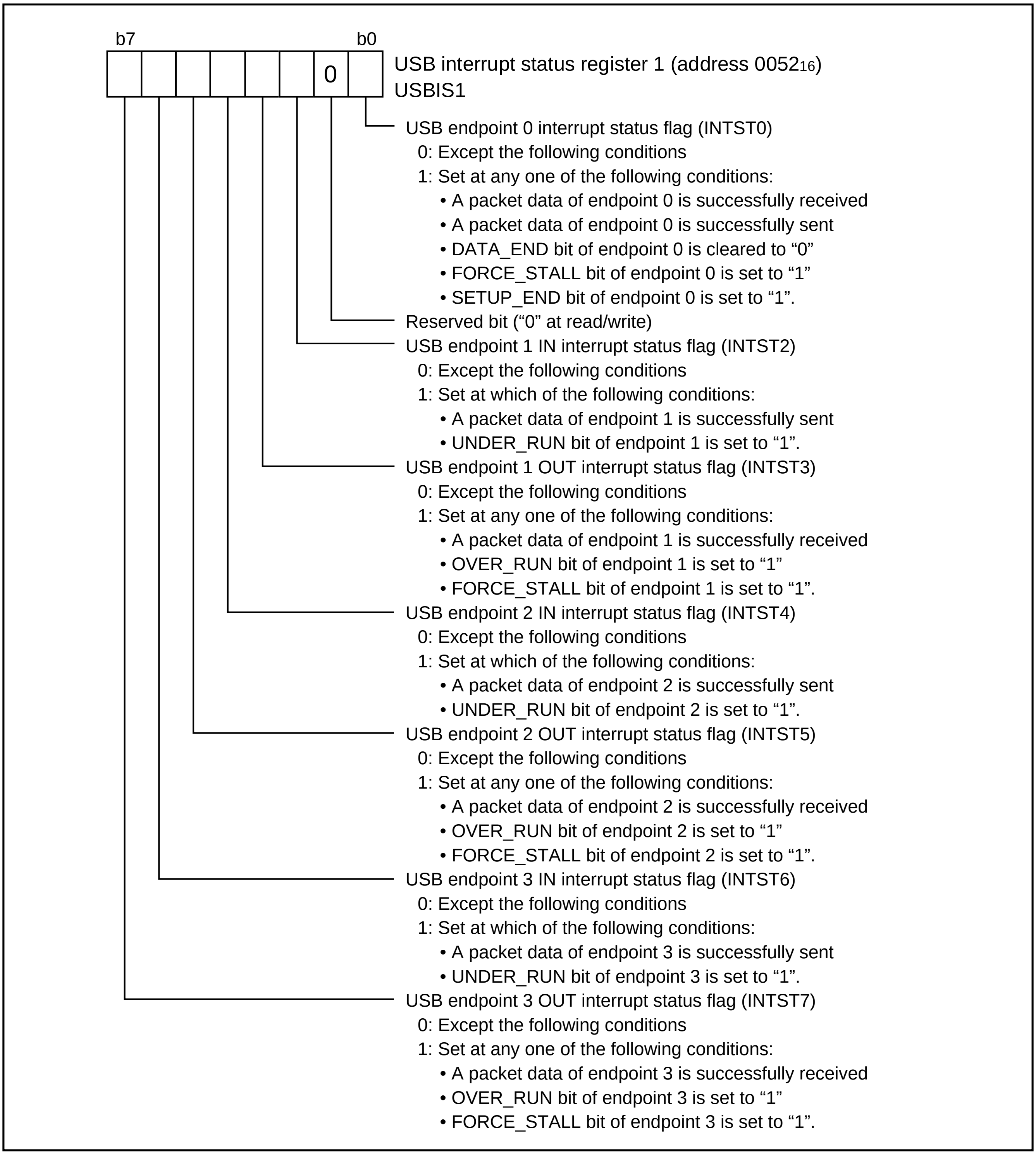


Fig. 40 Structure of USB interrupt status register 1

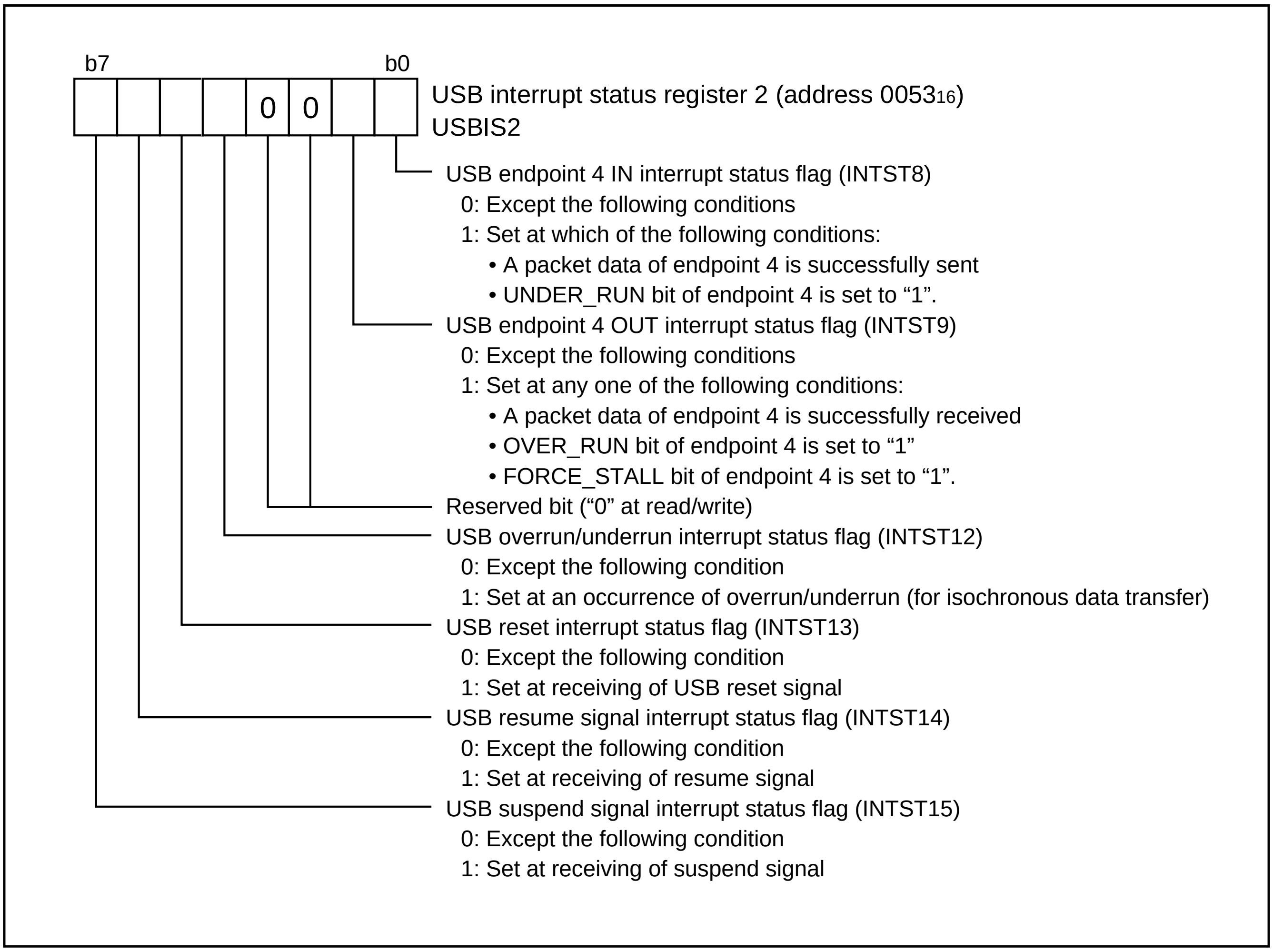


Fig. 41 Structure of USB interrupt status register 2

[USB Interrupt Enable Registers 1 and 2] USBIE1, USBIE2
The USB interrupt enable registers are used to enable the USB function interrupt. Upon reset, all USB interrupts except the USB suspend and USB resume interrupts are enabled.

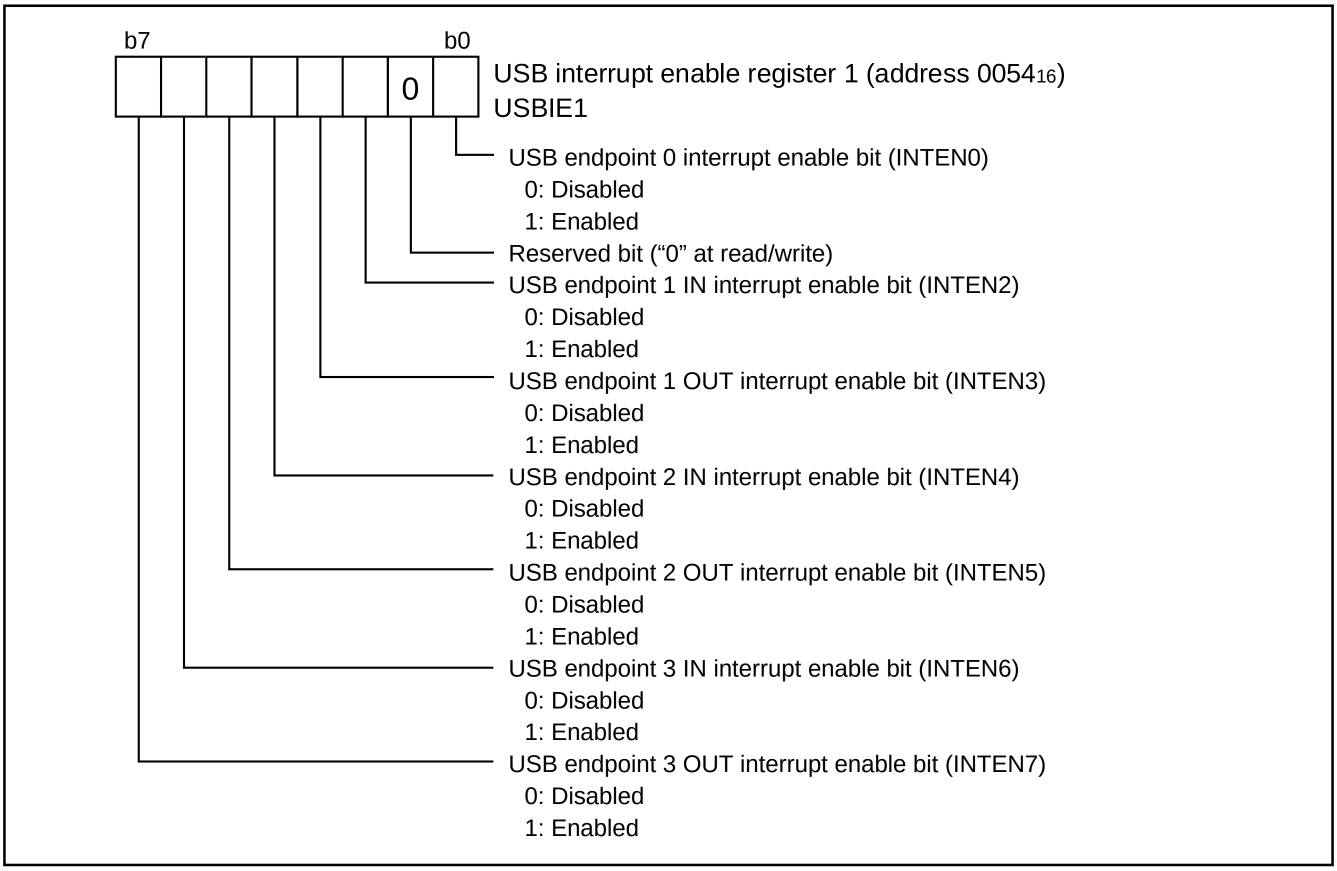


Fig. 42 Structure of USB interrupt enable register 1

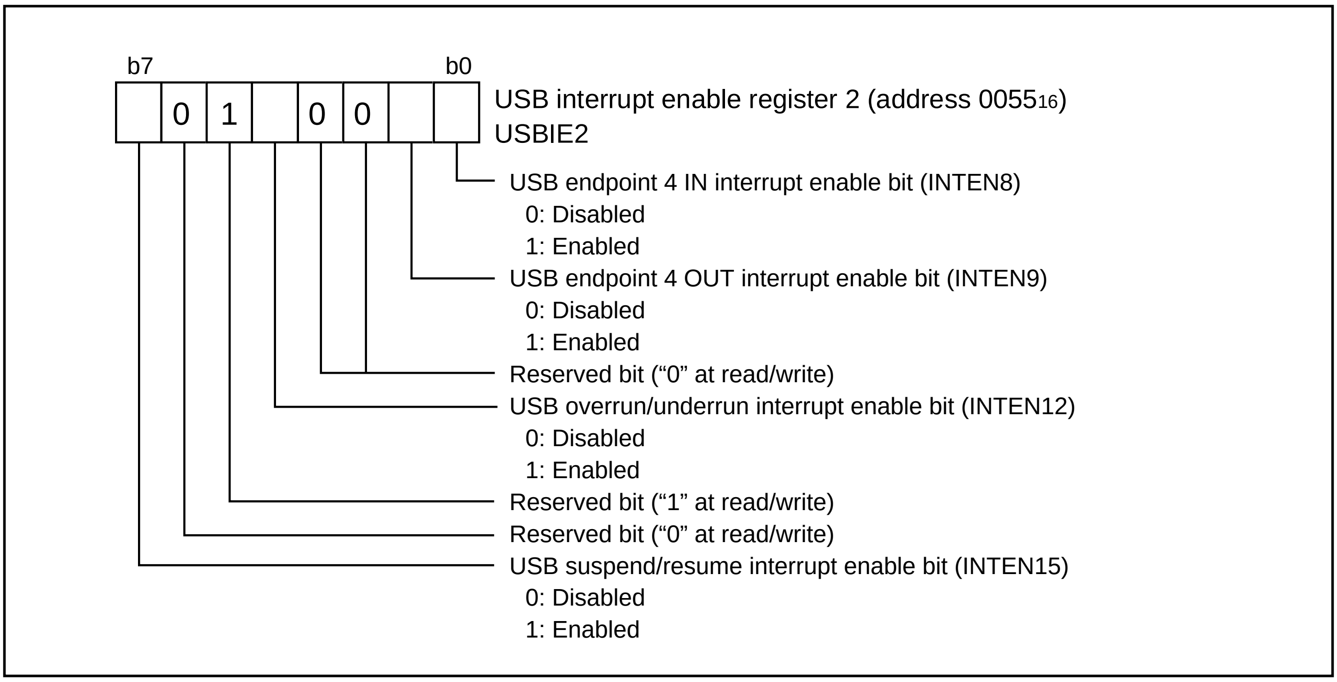


Fig. 43 Structure of USB interrupt enable register 2

[USB Frame Number Registers Low and High] USBSOFL, USBFOFH

These 11-bit registers contain the frame number of the SOF token received from the host computer. These are read-only registers.

[USB Endpoint Index Register] USBINDEX

This register specifies the accessible endpoint. It serves as an index to endpoint-specific USB Endpoint x IN Control Register, USB Endpoint x OUT Control Register, USB Endpoint x IN Max. Packet Size Register, USB Endpoint x OUT Max. Packet Size Register, USB Endpoint x OUT Write Count Register, and USB FIFO Mode Selection Register (x = 0 to 4).

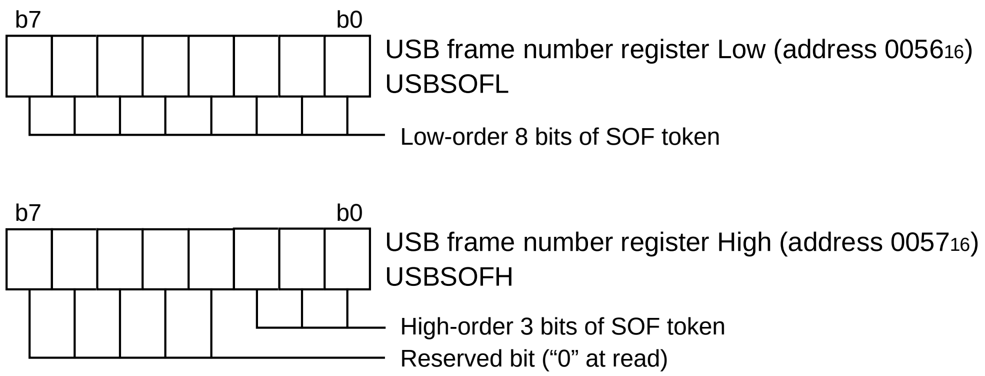


Fig. 44 Structure of USB frame number registers

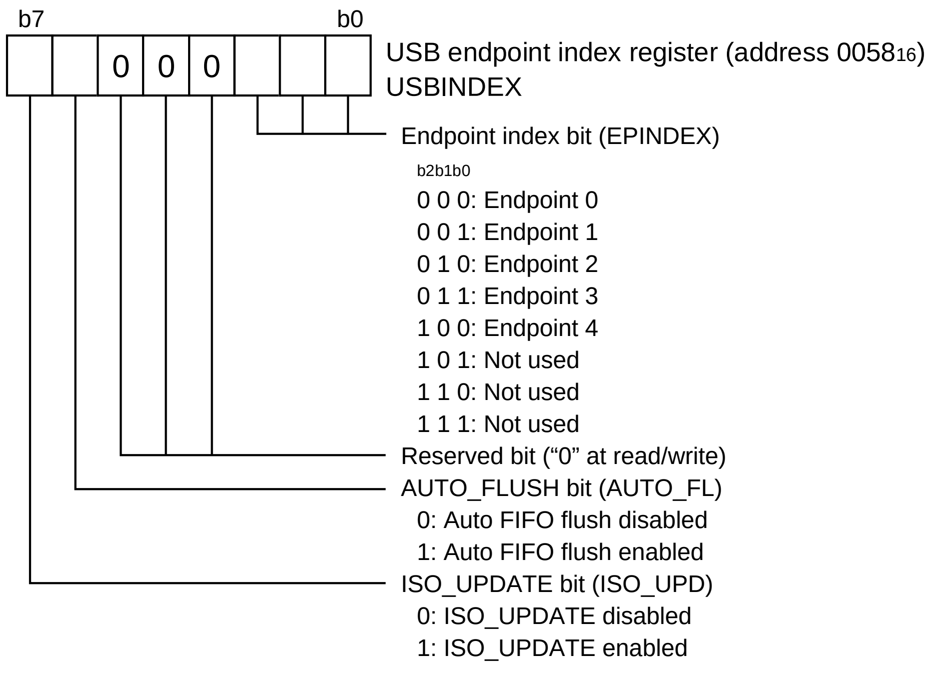


Fig. 45 Structure of USB frame number registers

[USB Endpoint 0 IN Control Register] IN_CSR

This register contains the control and status information of the endpoint 0. This USB FCU sets the OUT_PKT_RDY flag to “1” upon having received a data packet in the OUT FIFO. When reading its one data packet from the OUT FIFO, be sure to set this flag to “0”.

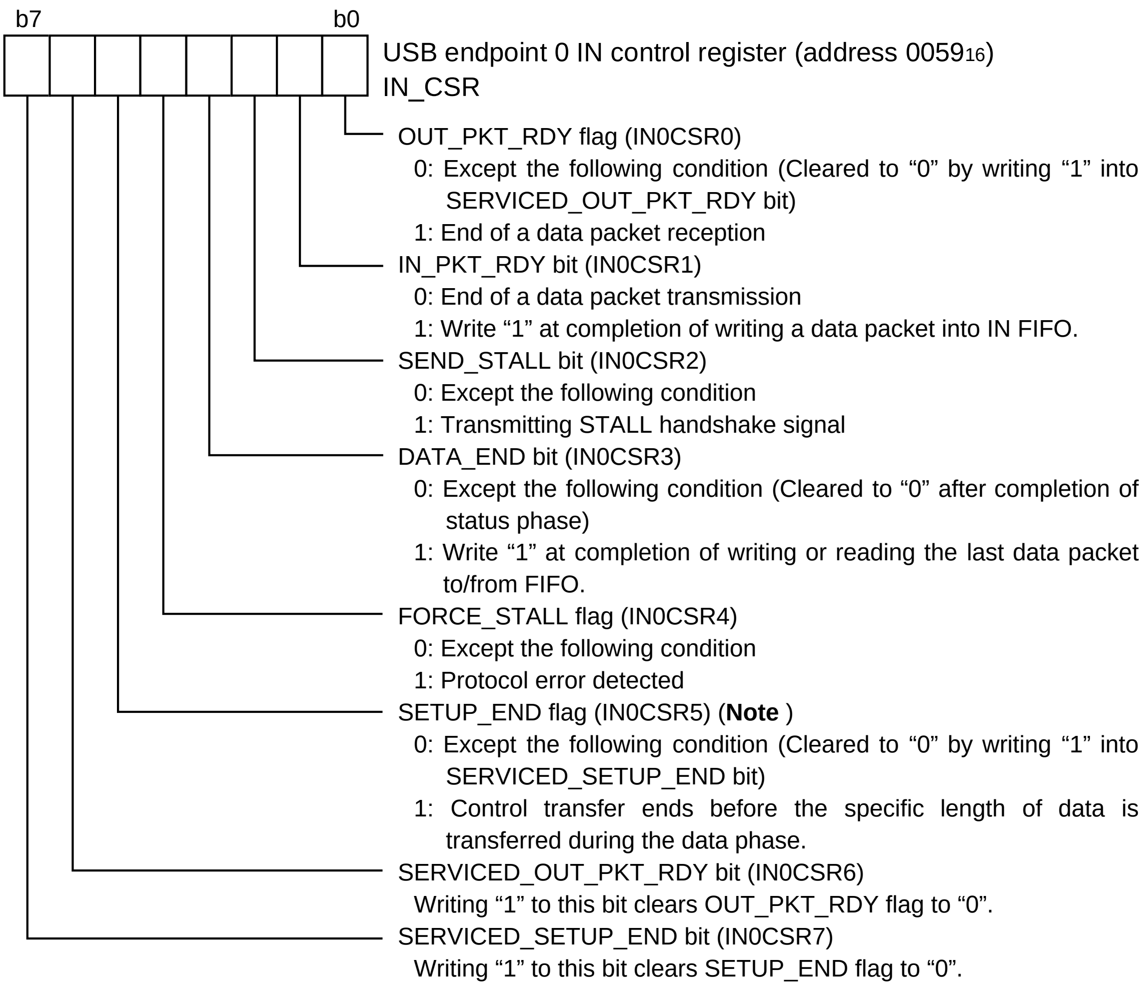
After a SETUP token is received, the MCU is in the “decode wait state” until the OUT_PKT_RDY flag is cleared. If the OUT_PKT_RDY flag is not cleared (indicating that the host request has not been successfully decoded), the USB FCU keep returning a NAK to the host for all IN/OUT tokens.

Set the IN_PKT_RDY bit to “1” after the data packet has been written to the IN FIFO. If this bit is set to “1” even though nothing has been written to the IN FIFO, a “0” length data (NULL packet) is sent to the host. The SEND_STALL bit is for sending a STALL to the host if an unsupported request is received by the USB FCU. This bit must be set to “1”. When the OUT_PKT_RDY flag is set to “0” for request reception, the USB FCU transmits a STALL signal to the Host CPU. Perform the following three processes simultaneously:

- Set SEND_STALL bit to “1”
- Set DATA_END bit to “1”
- Set OUT_PKT_RDY flag to “0” by setting SERVICED_OUT _PKT_RDY bit to “1”.

Note that if “0” is written to the SEND_STALL bit before the CLEAR_FEATURE (endpoint STALL) request has been received, the next STALL will not be generated.

The DATA_END bit informs the USB FCU of the completion of the process indicated in the SETUP packet. Set this bit to “1” when the process requested in the SETUP packet is completed. (Control Read Transfer: set this bit after writing all of the requested data to the FIFO; Control Write Transfer: set this bit to “1” after reading all of the requested data from the FIFO.) When this bit is “1”, the host request is ignored and a STALL is returned. After the status phase process is completed, the USB FCU automatically clears it to “0”.



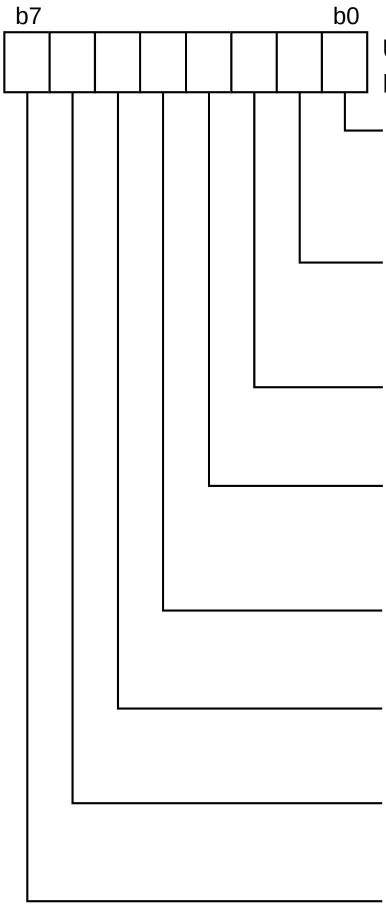
Note: If this bit is set to “0”, stop accessing the FIFO to serve the previous setup transaction.

Fig. 46 Structure of USB endpoint 0 IN control register

[USB Endpoint x (x = 1 to 4) IN Control Register] IN_CSR

This register contains the control and status information of the respective IN Endpoints 1 to 4.

Set the IN_PKT_RDY bit to "1" after the data packet has been written to the IN FIFO. This bit is cleared to "0" when the data transfer is completed. In a bulk IN transfer, this bit is cleared when an ACK signal is received from the host. If an ACK signal is not received, this bit (and the TX_NOT_EMPTY bit) remains as "1". This same data packet is sent after the next IN token is received. The FLUSH bit is for flushing the data in the IN FIFO.



USB endpoint x IN control register (address 0059_{16})
IN_CSR

INT_PKT_RDY bit (INXCSR0)
0: End of a data packet transmission (**Note 1**)
1: Write "1" at completion of writing a data packet into IN FIFO. (**Note 3**)

UNDER_RUN flag (INXCSR1) (In isochronous data transfer)
0: No FIFO underrun (**Note 2**)
1: FIFO underrun occurred (**Note 1**) (USB overrun/underrun interrupt status flag is set to "0".)

SEND_STALL bit (INXCSR2) (**Note 2**)
0: Except the following condition
1: Transmitting STALL handshake signal

ISO/TOGGLE_INIT bit (INXCSR3) (**Note 2**)
0: Except the following condition
1: Initializing to endpoint used for isochronous transfer; Initializing the data toggle sequence bit

INTPT bit (INXCSR4) (**Note 2**)
0: Except the following condition
1: Initializing to endpoint used for interrupt transfer, rate feedback

TX_NOT_EPT flag (INXCSR5) (**Note 1**)
0: Empty in IN FIFO
1: Full in IN FIFO

FLUSH bit (INXCSR6)
0: Except the following condition (**Note 1**)
1: Flush FIFO. (**Note 2**)

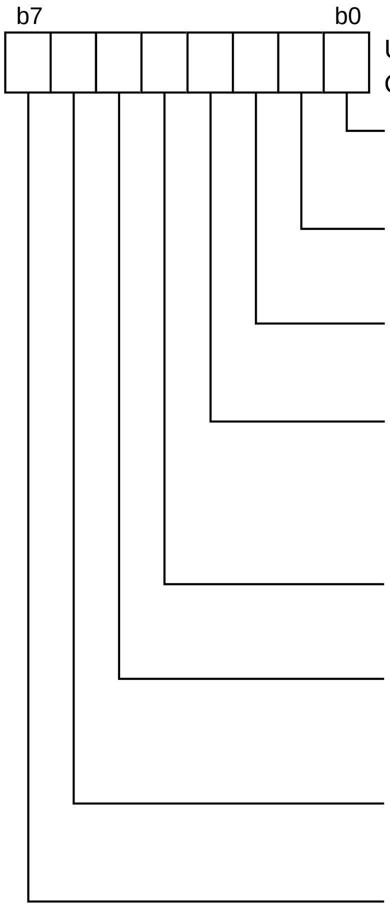
AUTO_SET bit (INXCSR7) (**Note 2**)
0: AUTO_SET disabled
1: AUTO_SET enabled (**Note 4**)

Notes 1: This bit is automatically set to "1" or cleared to "0".
2: The user must program to "1" or "0".
3: When AUTO_SET bit is "0", the user must set to "1". When AUTO_SET bit is "1", this bit is automatically set to "1".
4: To use the AUTO_SET function for an IN transfer when the AUTO_SET bit is set to "1", set the FIFO to single buffer mode.

Fig. 47 Structure of USB endpoint x (x = 1 to 4) IN control register

[USB Endpoint x (x = 1 to 4) OUT Control Register] OUT_CSR

This register contains the information and status of the respective OUT endpoints 1 to 4. In the endpoint 0, all bits are reserved and cannot be used (they will all be read out as “0”). The USB FCU sets the OUT_PKT_RDY flag to “1” after a data packet has been received into the OUT FIFO. After reading the data packet in the OUT FIFO, clear this flag to “0”. However, if there is still data in the OUT FIFO, the flag cannot be cleared even by writing “0” by software.



USB endpoint x OUT control register (address $005A_{16}$)
OUT_CSR

OUT_PKT_RDY flag (OUTXCSR0)
0: Except the following condition (**Note 3**)
1: End of a data packet reception (**Note 2**)

OVER RUN flag (OUTXCSR1) (In isochronous data transfer)
0: No FIFO overrun (**Note 2**)
1: FIFO overrun occurred (**Note 1**)

SEND_STALL bit (OUTXCSR2) (**Note 2**)
0: Except the following condition
1: Transmitting STALL handshake signal

ISO/TOGGLE_INIT bit (OUTXCSR3) (**Note 2**)
0: Except the following condition
1: Initializing to endpoint used for isochronous transfer; Enabling reception of DATA0 and DATA1 as PID (Initializing the toggle)

FORCE_STALL flag (OUTXCSR4)
0: Except the following condition (**Note 2**)
1: Protocol error detected (**Note 1**)

DATA_ERR flag (OUTXCSR5)
0: Except the following condition (**Note 2**)
1: CRC or bit stuffing error detected in transferring isochronous data (**Note 1**)

FLUSH bit (OUTXCSR6)
0: Except the following condition (**Note 1**)
1: Flush FIFO. (**Note 2**)

AUTO_CLR bit (OUTXCSR7) (**Note 2**)
0: AUTO_CLR disabled
1: AUTO_CLR enabled

Notes 1: This bit is automatically set to “1” or cleared to “0”.
2: The user must program to “1” or “0”.
3: When AUTO_CLR bit is “0”, the user must clear to “0”. When AUTO_CLR bit is “1”, this bit is automatically cleared to “0”.

Fig. 48 Structure of USB endpoint x (x = 1 to 4) OUT control register

[USB Endpoint x (x = 0 to 4) IN Max. Packet Size Register] IN_MAXP

This register specifies the maximum packet size (MAXP) of an endpoint x IN packet. The value set for endpoint 1 is the number of transmitted bytes divided by 8, and the value set for endpoints 0, 2, 3, and 4 is the actual number of transmitted bytes. The CPU can change these values using the SET_DESCRIPTOR command.

The initial value for endpoints 0, 2, 3 and 4 is 8, and the initial value for endpoint 1 is 1.

[USB Endpoint x (x = 0 to 4) OUT Max. Packet Size Register] OUT_MAXP

This register specifies the maximum packet size (MAXP) of an Endpoint x OUT packet. The value set for endpoint 1 is the number of received bytes divided by 8, and the value set for endpoints 0, 2, 3, and 4 is the actual number of received bytes. The CPU can change these values using the SET_DESCRIPTOR command.

The initial value for endpoints 0, 2, 3, and 4 is 8, and the initial value for endpoint 1 is 1. When using the endpoint 0, both USB endpoint x IN max. packet size register (IN _MAXP) and USB endpoint x OUT max. packet size register (OUT_MAXP) are set to the same value. Changing one register's value effectively changes the value of the other register as well.

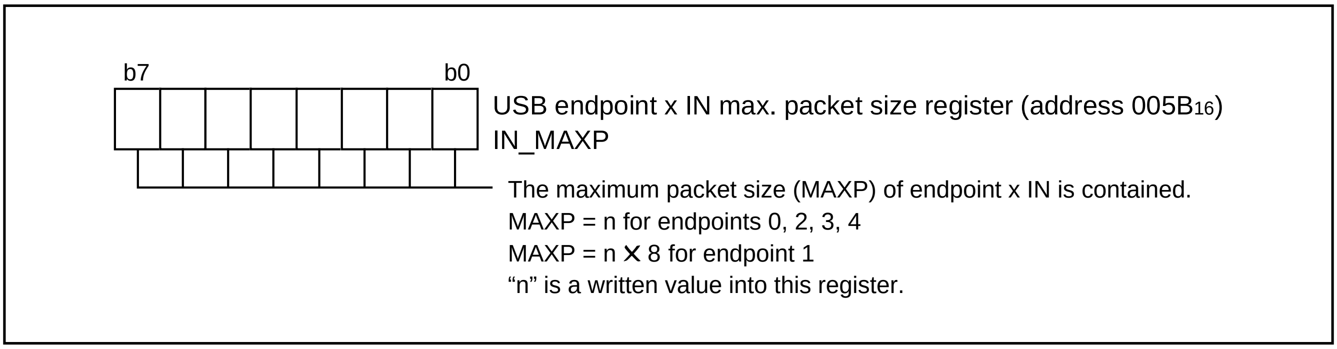


Fig. 49 Structure of USB endpoint x IN max. packet size register

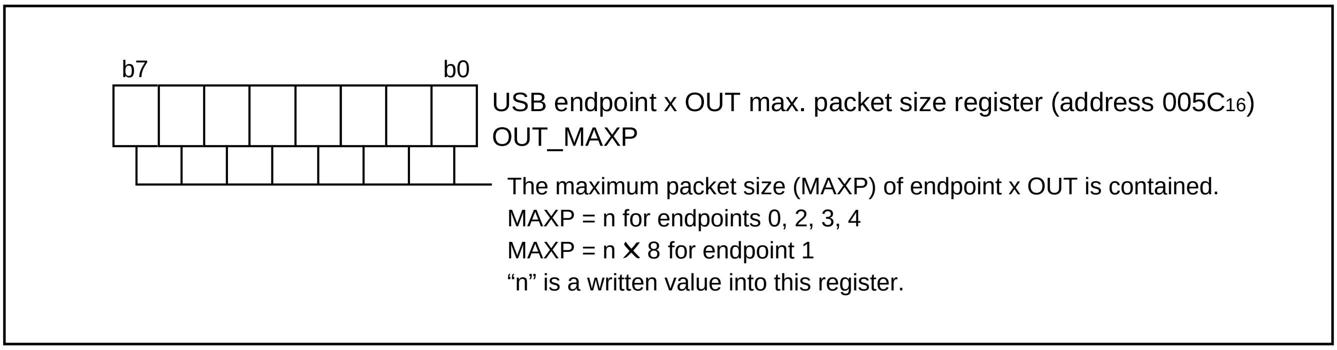


Fig. 50 Structure of USB endpoint x OUT max. packet size register

[USB endpoint x (x = 0 to 4) OUT Write Count Registers (Low and High)] WRT_CNTRL, WRT_CNTH
These registers contain the number of bytes in the endpoint x OUT FIFO. These are read-only registers. These two registers must be read after the USB FCU has received a packet of data from the host. When reading these registers, the lower byte must be read first, then the higher byte.
When the OUT FIF0 is in double buffer mode, the CPU first reads the received number of bytes of the former data packet. The next CPU read can obtain that of the new data packet.

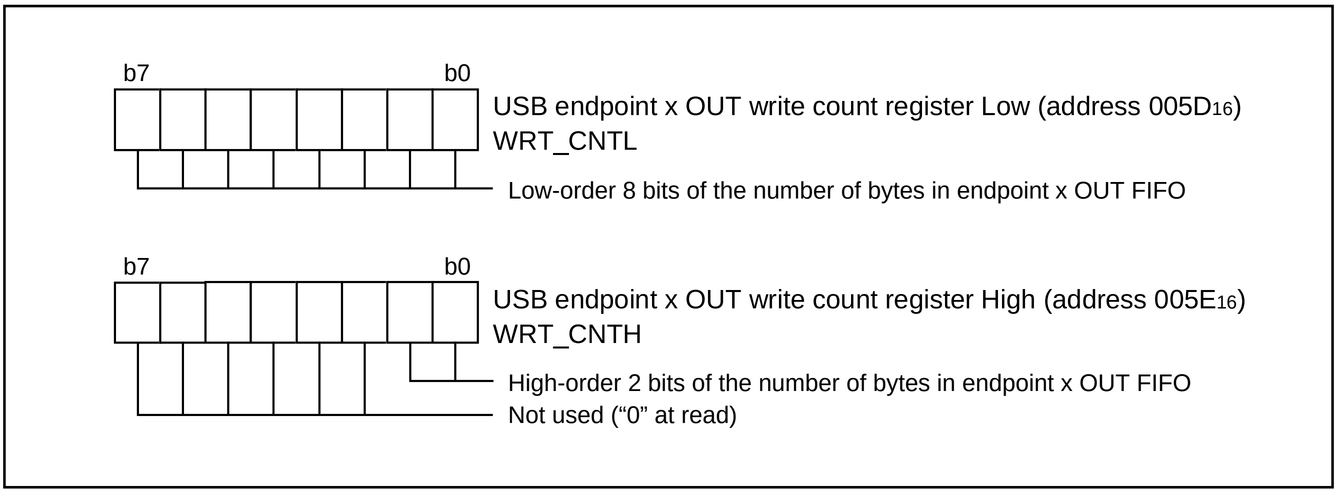


Fig. 51 Structure of USB endpoint x (x = 0 to 4) OUT write count registers

[USB Endpoint x (x = 0 to 4) FIFO Register] USBFIFOx
These registers are the USB IN (transmit) and OUT (receive) FIFO data registers. Write data to the corresponding register, and read data from the corresponding register.
When the maximum packet size is equal to or less than half the FIFO size, these registers function in double buffer mode and can hold two packets of data. When the IN_PKT_RDY bit is “0” and the TX_NOT_EMPTY bit is “1”, these bits indicate that one packet of data is stored in the IN FIFO. When the OUT FIFO is in double buffer mode, the OUT_PKT_RDY flag remains as “1” after the first packet of data is read out (it actually goes to “0” and returns to “1” after one ϕ cycle).

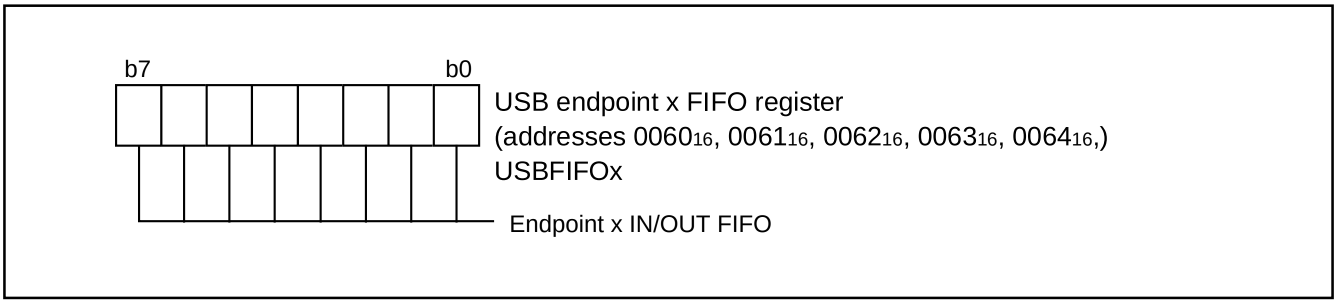


Fig. 52 Structure of USB endpoint x (x = 0 to 4) FIFO register

[USB Endpoint FIFO Mode Selection Register] USBFIFOMR

This register determines IN/OUT FIFO size mode for endpoint 1 or endpoint 2. This register is invalid when using endpoint 0, 3, or 4.

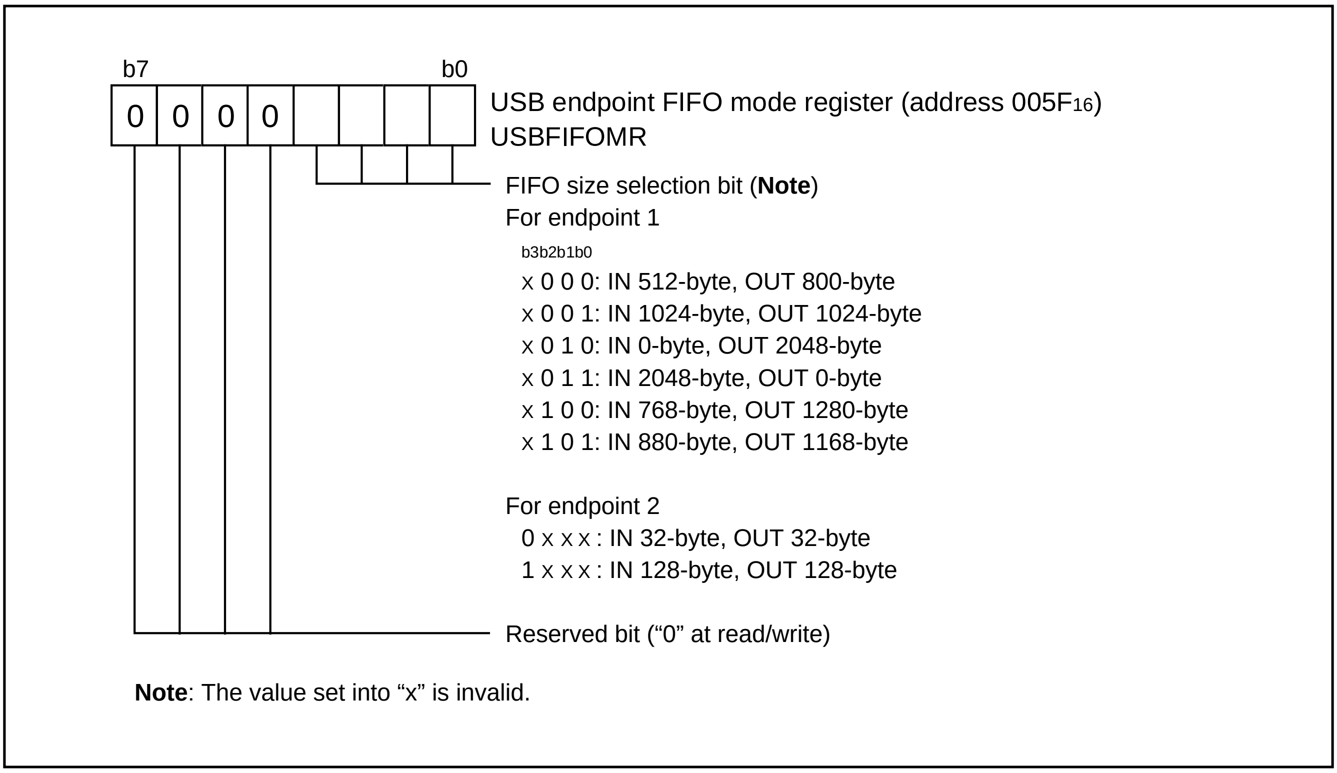


Fig. 53 Structure of USB endpoint FIFO mode register

MASTER CPU BUS INTERFACE

The 7641 group internally has a 2-byte bus interface which control signals from the host CPU side can operate (slave mode).
This bus interface allows the 7641 group to be directly connected with a R/W type of CPU bus or a $\overline{RD}$ and $\overline{WR}$ separated type of CPU bus. Figure 56 shows the block diagram of master CPU bus interface function.

The data bus buffer function I/O pins ($P5_2$ – $P5_7$, P6, $P7_2$–$P7_4$) also function as the normal I/O ports. When the Master CPU Bus Interface Enable bit of Data Bus Buffer Control Register (bit 6 of address $004A_{16}$) is "0", these pins become the normal I/O ports. When it is "1", these pins become the master CPU bus interface function pins.
Additionally, when using the master CPU bus interface function, set port P6 to input mode by setting "00_{16}" into its port direction register (address 0015_{16}).

The selection of either the single data bus buffer mode, which uses 1 byte: data bus buffer 0 only, or the double data bus buffer mode, which uses 2 bytes: data bus buffer 0 and data bus buffer 1, is performed by the Data Bus Buffer Function Select Bit of Data Bus Buffer Control Register 1 (bit 7 of address $004E_{16}$). Port $P7_2$ becomes $\overline{S_1}$ input pin in the double data bus buffer mode.
When data is written from the host CPU side, an input buffer full interrupt occurs. When data is read from the host CPU, an output buffer empty interrupt occurs. The 7641 group shares two input buffer full interrupt requests and two output buffer empty interrupt requests as shown in Figure 54, respectively.

The 7641 group can also operate the master CPU bus interface connecting with the Built-in DMAC. This could transfer a large amount of data fast.
An input signal level of data bus buffer function input pins can be selected between a CMOS level and a TTL level. Set it using the Master CPU Bus Input Level Select Bit of Port Control Register (address 0010_{16})
.

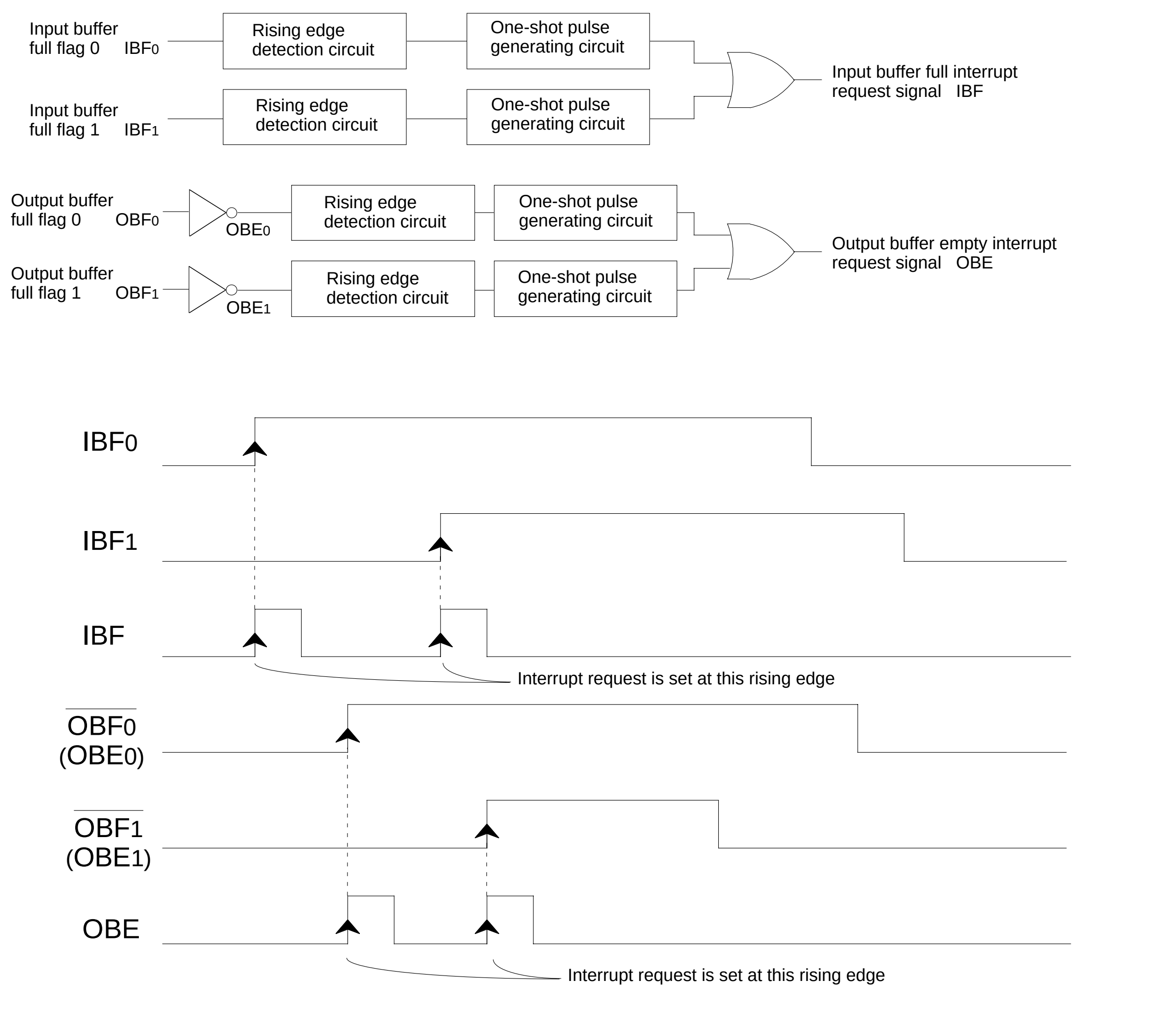


Fig. 54 Interrupt request circuit of data bus buffer

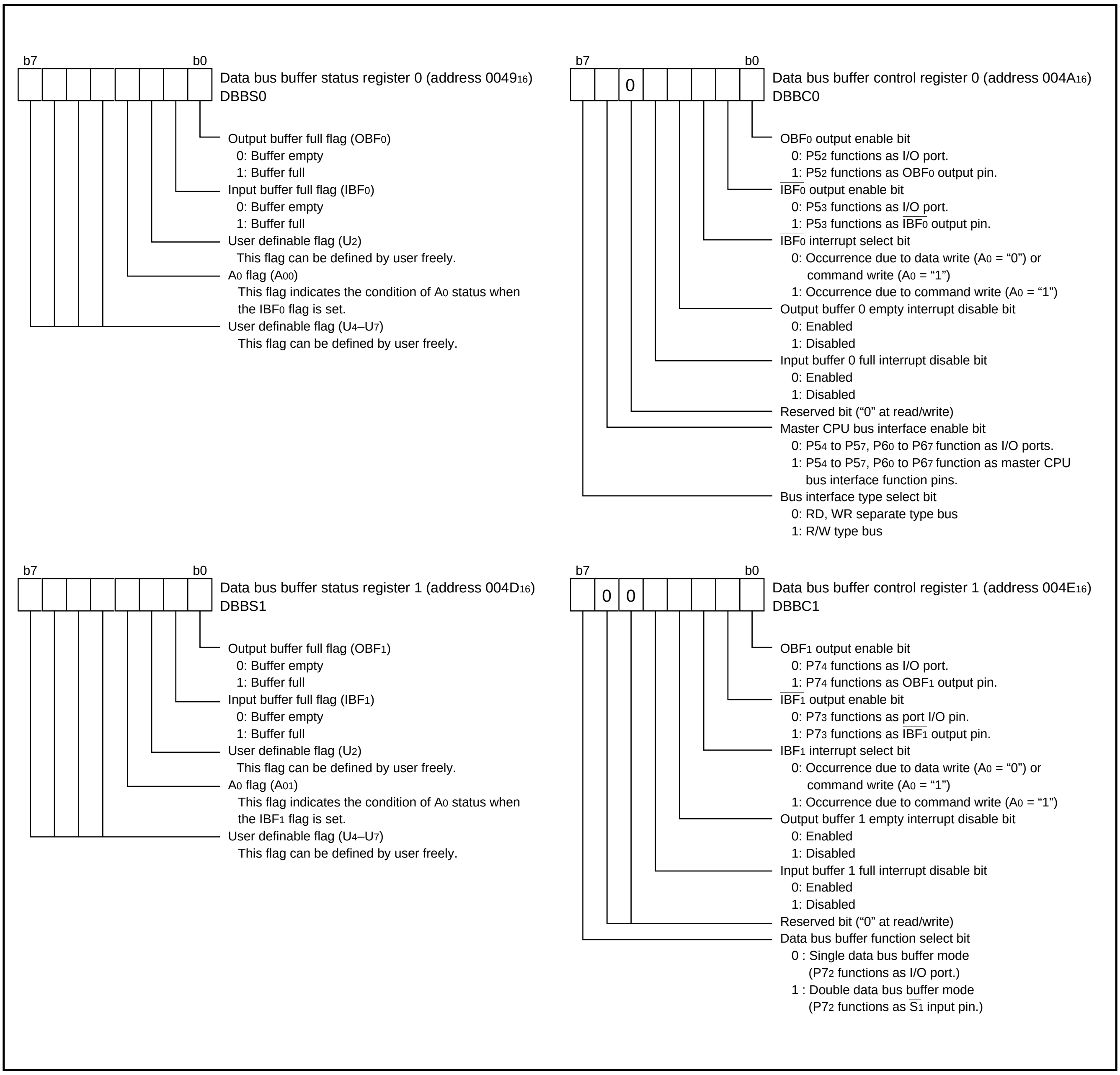


Fig. 55 Structure of master CPU bus interface related registers

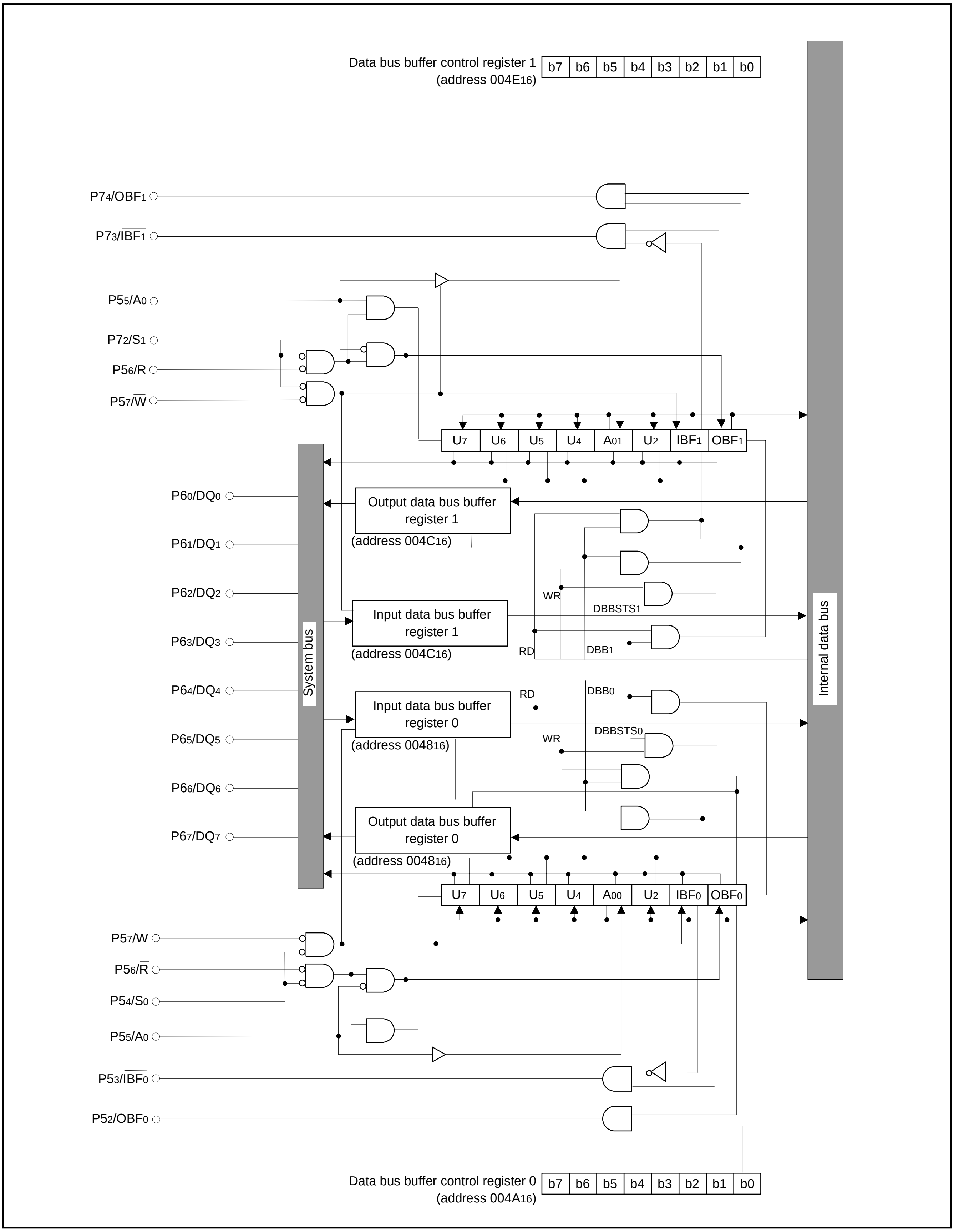


Fig. 56 Master CPU bus interface block diagram

[Data Bus Buffer Status Register 0, 1 (DBBS0, DBBS1)] 0049_{16}, $004D_{16}$

The data bus buffer status registers 0, 1 consist of eight bits each. Bits 0, 1, and 3 are read-only bits and indicate the status of the data bus buffer. Bits 2, 4, 5, 6, and 7 are user definable flags which can be programed, and can be read/written. The host CPU can only read this register when the A_0 pin is set to "H".

•Bit 0: Output buffer full flag OBF_0, OBF_1

When writing data to the output data bus buffer, this flag is set to "1". When reading the output data bus buffer from the host CPU, this flag is cleared to "0".

•Bit 1: Input buffer full flag IBF_0, IBF_1

When writing data from the host CPU to the input data bus buffer, this flag is set to "1". When reading the input data bus buffer from the slave CPU side, this flag is are cleared to "0".

•Bit 3: A0 flag $A0_0$, $A0_1$

When writing data from the host CPU to the input data bus buffer, the level of the A_0 pin is latched.

[Input Data Bus Buffer Registers 0, 1 ($DBBIN_0$, $DBBIN_1$)] 0048_{16}, $004C_{16}$

Data on the data bus is latched to $DBBIN_0$ or $DBBIN_1$ by writing request from the host CPU. Data of DBBINs can be read from the Data Bus Buffer Registers (address 0048_{16} or $004C_{16}$) on the SFR area.

[Output Data Bus Buffer Registers 0, 1 ($DBBOUT_0$, $DBBOUT_1$)] 0048_{16}, $004C_{16}$

When writing data to the Data Bus Buffer Registers (address 0048_{16} or $004C_{16}$) on the SFR area, data is set to $DBBOUT_0$ or $DBBOUT_1$. Data of DBBOUTs is output onto the data bus by performing the reading request from the host CPU when the A_0 pin is set to "L".

Table 8 Function description of control I/O pins of master CPU bus interface

Pin	Name	OBF_0 output enable bit	$\overline{IBF_0}$ output enable bit	OBF_1 output enable bit	$\overline{IBF_1}$ output enable bit	Input/ Output	Functions
$P5_2/OBF_0$	OBF_0	1	0	0	0	Output	Status output signal. OBF_0 signal is output.
$P5_3/\overline{IBF_0}$	$\overline{IBF_0}$	0	1	0	0	Output	Status output signal. $\overline{IBF_0}$ signal is output.
$P5_4/\overline{S_0}$	$\overline{S_0}$	—	—	—	—	Input	Chip select input. This is used for selecting the data bus buffer, which is selected at “L” level.
$P5_5/A_0$	A_0	—	—	—	—	Input	Address input. This is used for selecting DBBSTS and DBBOUT when the host CPU reads. This is used for distinguishing command from data when the host CPU writes.
$P5_6/\overline{R}$ (E)	$\overline{R}$ (E)	—	—	—	—	Input	This is a timing signal for reading data from the data bus buffer to the host CPU.
$P5_7/\overline{W}$ (R/$\overline{W}$)	$\overline{W}$ (R/$\overline{W}$)	—	—	—	—	Input	This is a timing signal for writing data to the data bus buffer by the host CPU.
$P7_2/\overline{S_1}$	$\overline{S_1}$	—	—	—	—	Input	Chip select input. This is used for selecting the data bus buffer, which is selected at “L” level.
$P7_3/\overline{IBF_1}/\overline{HLDA}$	$\overline{IBF_1}$	0	0	0	1	Output	Status output signal. $\overline{IBF_1}$ signal is output.
$P7_4/OBF_1$	OBF_1	0	0	1	0	Output	Status output signal. OBF_1 signal is output.

COUNT SOURCE GENERATOR

The 7641 Group has a built-in special count source generator, SCSG. This generator consists of two 8-bit timers: SCSG1 and SCSG2. The output of the special count source generator can be used as a clock source for the timer X, serial I/O and two UARTs.

SCSG Operation

Timers SCSG1 and SCSG2 are both down count timers. When the count reaches “0”, an underflow occurs at the next count source rising edge and the contents of the corresponding timer latch are loaded to the timer. The division ratio of each SCSG-x timer is given by 1 / (n+1), where “n” is the value set to the SCSG-x timer. The output of Timer SCSG1 is ANDed with the original clock (ϕ) to make a count source for Timer SCSG2.

The SCSG output is Clock SCSGCLK. The frequency is calculated as follows:

SCSGCLK = $\phi \times \{n1 / (n1+1)\} \times \{1 / (n2+1)\}$
n1: value set to SCSG1
n2: value set to SCSG2

If the SCSG1 Count Stop Bit (SCSGM1) is set to “1”, or Timer SCSG1 is set to “0”, the SCSG1 count stops. When this happens, the count source for Timer SCSG2 becomes ϕ.

Data Write Control

When the SCSG1 Data Write Control Bit or SCSG2 Data Write Control Bit is set to “0”, and data is written to the SCSG-x timer; the data is written to the corresponding latch and timer at the same time. When that bit is set to “1”, the data is only written to the latch.

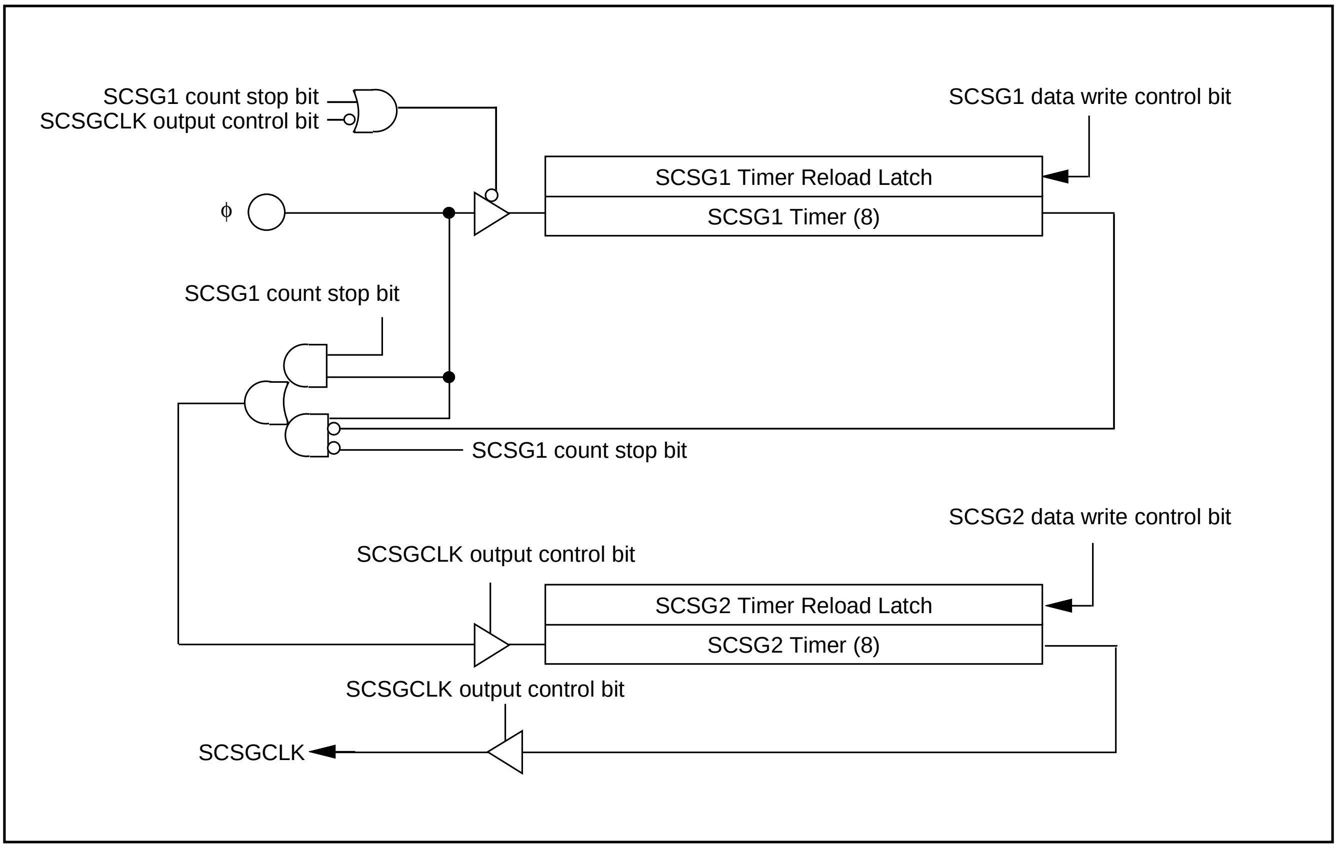


Fig. 57 Special count source generator block diagram

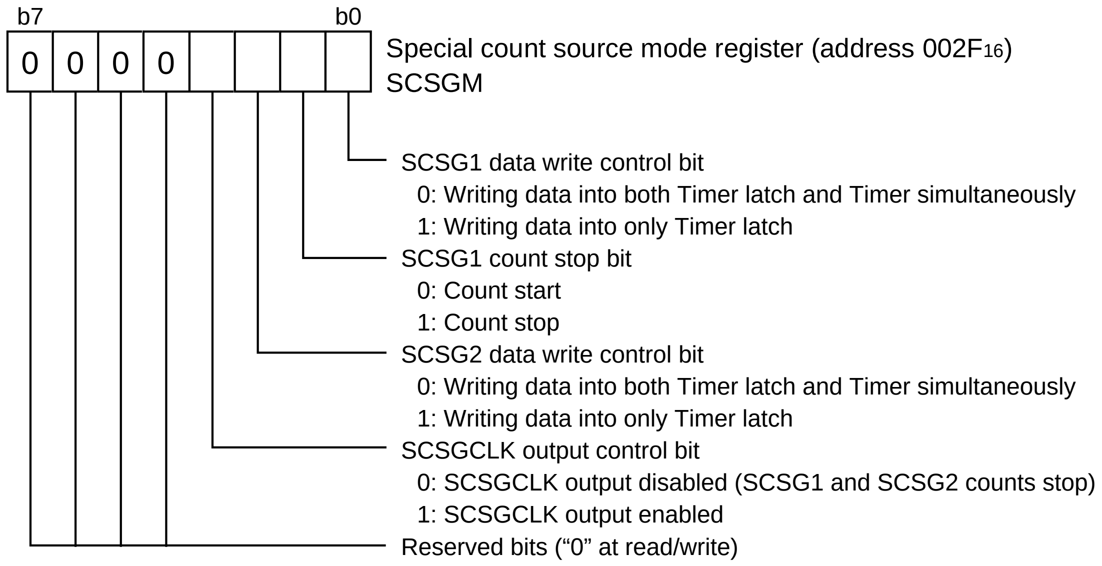


Fig. 58 Structure of special count source generator mode register

FREQUENCY SYNTHESIZER (PLL)

The frequency synthesizer generates the 48 MHz clock required by f_{USB} and f_{SYN}, which are multiples of the external input reference $f(X_{IN})$. Figure 59 shows the block diagram for the frequency synthesizer circuit.

The Frequency Synthesizer Input Bit selects either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer.

The Frequency Synthesizer Multiply Register 2 (FSM2: address $006E_{16}$) divides f_{IN} to generate f_{PIN}, where

$f_{PIN} = f_{IN} / 2(n + 1)$, n: value set to FSM2.

When the value of Frequency Synthesizer Multiply Register 2 is set to 255, the division is not performed and f_{PIN} will equal f_{IN}.

f_{VCO} is generated according to the contents of Frequency Synthesizer Multiply Register 1 (FSM1: address $006D_{16}$), where

$f_{VCO} = f_{PIN} \times \{2(n + 1)\}$, n: value set to FSM1.

Set the value of FSM1 so that the value of f_{VCO} is 48 MHz.

f_{SYN} is generated according to the contents of the Frequency Synthesizer Divide Register (FSD: address $006F_{16}$), where

$f_{SYN} = f_{VCO} / 2(m + 1)$, m: value set to FSD.

When the value of the Frequency Synthesizer Divide Register is set to 255, the division is not performed and f_{SYN} becomes invalid.

[Frequency Synthesizer Control Register] FSC

Setting the Frequency Synthesizer Enable Bit (FSE) to "1" enables the frequency synthesizer. When the Frequency Synthesizer Lock Status Bit (LS) is "1" in the frequency synthesizer enabled, this indicates that f_{SYN} and f_{VCO} have correct frequencies.

■Notes

Make sure to connect a low-pulse filter to the LPF pin when using the frequency synthesizer. In addition, please refer to "Programming Notes: Frequency Synthesizer" when recovering from a Hardware Reset.

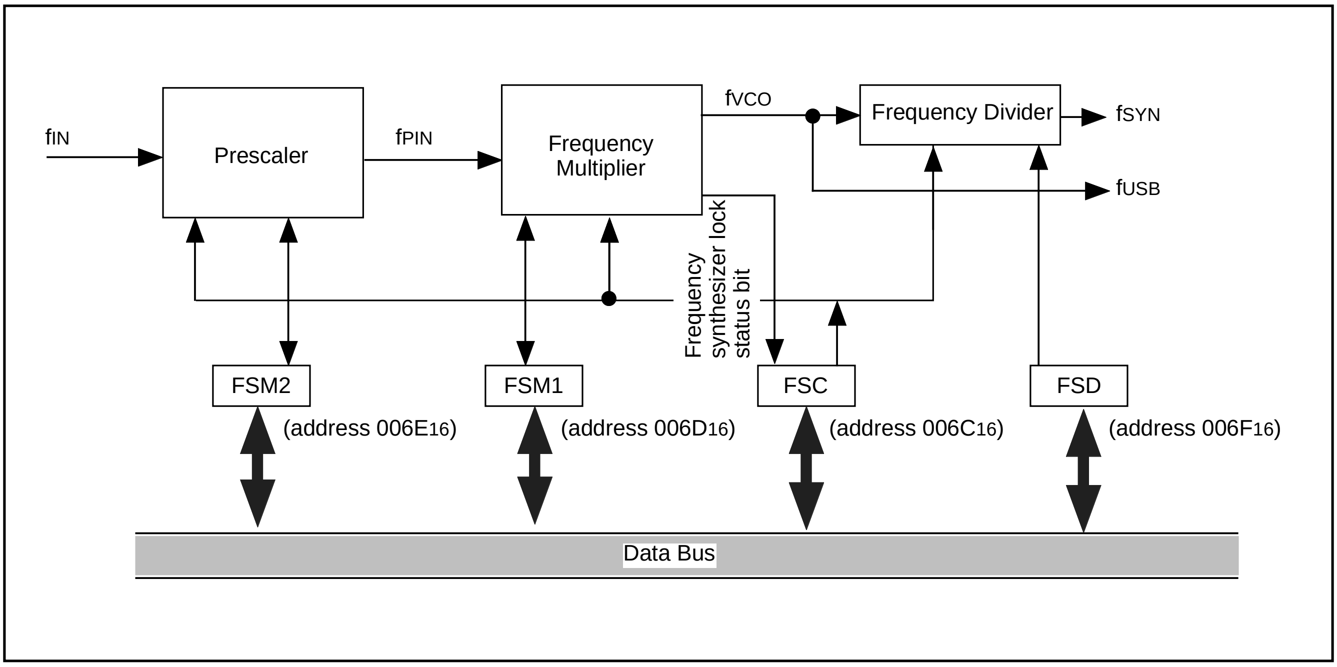
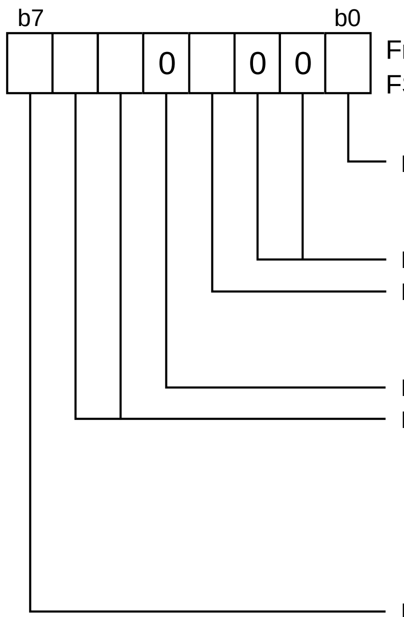


Fig. 59 Frequency synthesizer block diagram



Frequency synthesizer control register (address $006C_{16}$)
FSC

Frequency synthesizer enable bit (FSE)
0: Disabled
1: Enabled

Fix to "00".

Frequency synthesizer input bit (FIN)
0: $f(X_{IN})$
1: $f(X_{CIN})$

Reserved bit ("0" at read/write)

LPF current control (CHG1, CHG0) (**Note**)
b6b5
0 0: Not available
0 1: Low current
1 0: Intermediate current (recommended)
1 1: High current

Frequency synthesizer lock status bit
0: Unlocked
1: Locked

Note: Bits 6 and 5 are set to (bit 6, bit 5) = (1, 1) at reset.
When using the frequency synthesizer, we recommend to set to (bit 6, bit 5) = (1, 0) after locking the frequency synthesizer.

Fig. 60 Structure of frequency synthesizer control register

RESET CIRCUIT

To reset the microcomputer, $\overline{RESET}$ pin should be held at an "L" level for 20 cycles or more of ϕ. Then the $\overline{RESET}$ pin is returned to an "H" level, and reset is released. They must be performed when the power source voltages are between 3.00 V and 3.60 V or 4.15 V and 5.25 V.

After the reset is completed, the program starts from the address contained in address $FFFA_{16}$ (high-order byte) and address $FFFB_{16}$ (low-order byte).

After oscillation has restarted, the timers 1 and 2 secures waiting time for the internal clock ϕ oscillation stabilized automatically by setting the timer 1 to "FF_{16}" and timer 2 to "01_{16}". The internal clock ϕ retains "H" level until Timer 2's underflow and it cannot be supplied until the underflow.

The pins state during reset are follows:

•When CNVss = "H"

Ports P0, P1, $P3_3$ to $P3_7$	: Outputting
Pins other than above mentioned ports	: Inputting

•When CNVss = "L"

All pins	: Inputting.

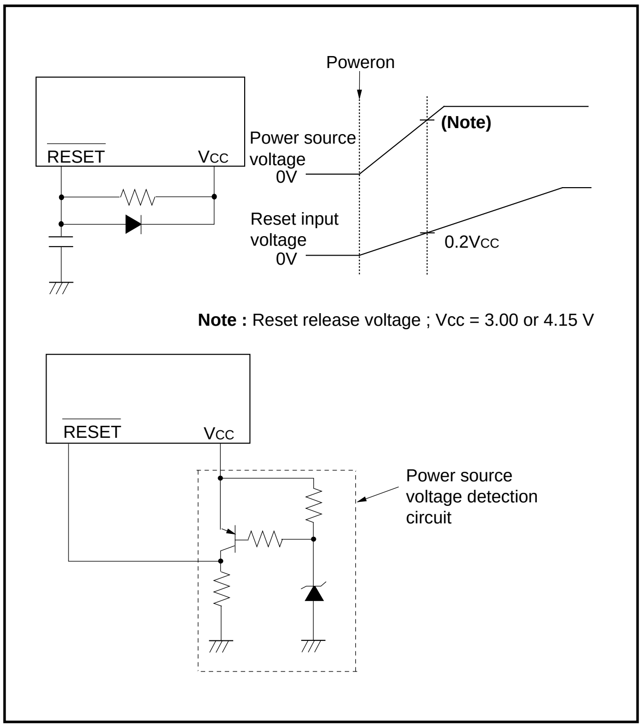


Fig. 61 Reset circuit example

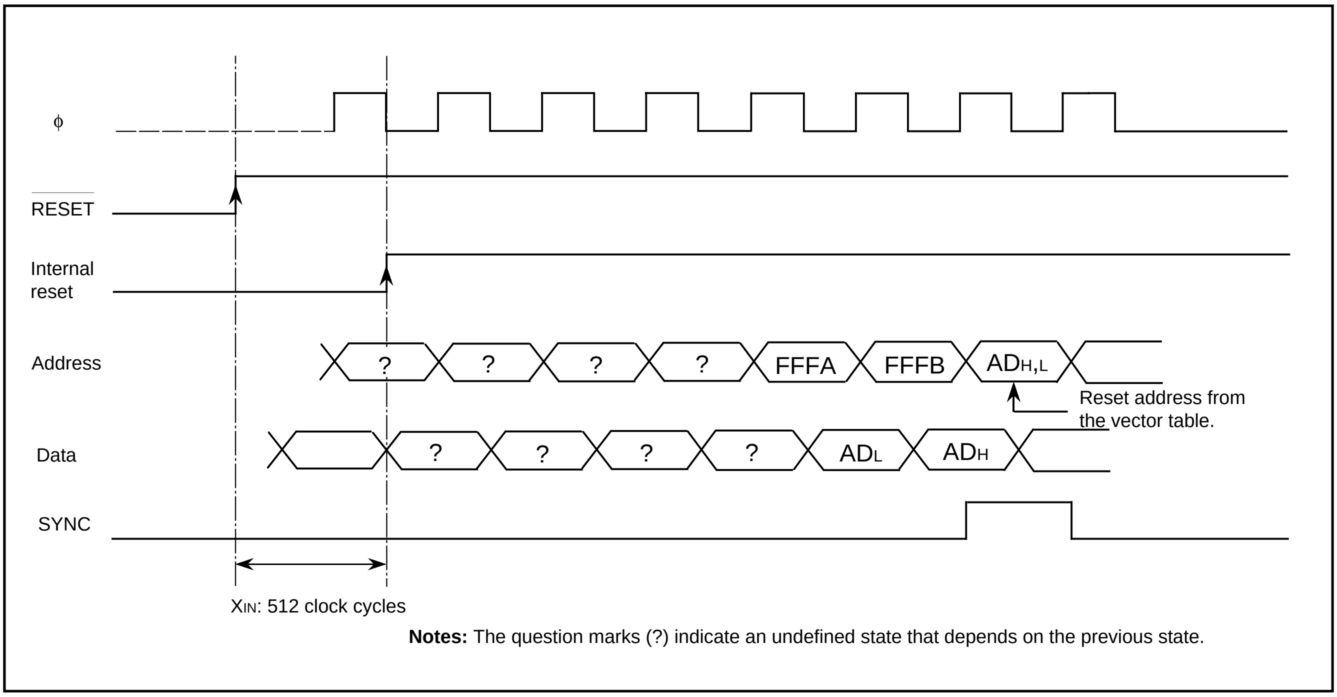


Fig. 62 Reset sequence

No.	Register	Address	Register contents
(1)	CPU mode register A (CPUA)	0000_{16}	00001100
(2)	CPU mode register B (CPUB)	0001_{16}	10000011
(3)	Interrupt request register A (IREQA)	0002_{16}	00_{16}
(4)	Interrupt request register B (IREQB)	0003_{16}	00_{16}
(5)	Interrupt request register C (IREQC)	0004_{16}	00_{16}
(6)	Interrupt control register A (ICONA)	0005_{16}	00_{16}
(7)	Interrupt control register B (ICONB)	0006_{16}	00_{16}
(8)	Interrupt control register C (ICONC)	0007_{16}	00_{16}
(9)	Port P0 (P0)	0008_{16}	00_{16}
(10)	Port P0 direction register (P0D)	0009_{16}	00_{16}
(11)	Port P1 (P1)	$000A_{16}$	00_{16}
(12)	Port P1 direction register (P1D)	$000B_{16}$	00_{16}
(13)	Port P2 (P2)	$000C_{16}$	00_{16}
(14)	Port P2 direction register (P2D)	$000D_{16}$	00_{16}
(15)	Port P3 (P3)	$000E_{16}$	00_{16}
(16)	Port P3 direction register (P3D)	$000F_{16}$	00_{16}
(17)	Port control register (PTC)	0010_{16}	00_{16}
(18)	Interrupt polarity select register (IPOL)	0011_{16}	00_{16}
(19)	Port P2 pull-up control register (PUP2)	0012_{16}	00_{16}
(20)	USB control register (USBC)	0013_{16}	00_{16}
(21)	Port P6 (P6)	0014_{16}	00_{16}
(22)	Port P6 direction register (P6D)	0015_{16}	00_{16}
(23)	Port P5 (P5)	0016_{16}	00_{16}
(24)	Port P5 direction register (P5D)	0017_{16}	00_{16}
(25)	Port P4 (P4)	0018_{16}	00_{16}
(26)	Port P4 direction register (P4D)	0019_{16}	00_{16}
(27)	Port P7 (P7)	$001A_{16}$	00_{16}
(28)	Port P7 direction register (P7D)	$001B_{16}$	00_{16}
(29)	Port P8 (P8)	$001C_{16}$	00_{16}
(30)	Port P8 direction register (P8D)	$001D_{16}$	00_{16}
(31)	Clock control register (CCR)	$001F_{16}$	00_{16}
(32)	Timer XL (TXL)	0020_{16}	FF_{16}
(33)	Timer XH (TXH)	0021_{16}	FF_{16}
(34)	Timer YL (TYL)	0022_{16}	FF_{16}
(35)	Timer YH (TYH)	0023_{16}	FF_{16}
(36)	Timer 1 (T1)	0024_{16}	FF_{16}
(37)	Timer 2 (T2)	0025_{16}	00000001
(38)	Timer 3 (T3)	0026_{16}	FF_{16}
(39)	Timer X mode register (TXM)	0027_{16}	00_{16}
(40)	Timer Y mode register (TYM)	0028_{16}	00_{16}
(41)	Timer 123 mode register (T123M)	0029_{16}	00_{16}
(42)	Serial I/O control register 1 (SIOCON1)	$002B_{16}$	01000000
(43)	Serial I/O control register 2 (SIOCON2)	$002C_{16}$	00011000
(44)	Special count source generator 1 (SCSG1)	$002D_{16}$	FF_{16}
(45)	Special count source generator 2 (SCSG2)	$002E_{16}$	FF_{16}
(46)	Special count source mode register (SCSGM)	$002F_{16}$	00_{16}
(47)	UART1 mode register (U1MOD)	0030_{16}	00_{16}

No.	Register	Address	Register contents
(48)	UART1 status register (U1STS)	0032_{16}	00000011
(49)	UART1 control register (U1CON)	0033_{16}	00_{16}
(50)	UART1 RTS control register (U1RTSC)	0036_{16}	10000000
(51)	UART2 mode register (U2MOD)	0038_{16}	00_{16}
(52)	UART2 status register (U2STS)	$003A_{16}$	00000011
(53)	UART2 control register (U2CON)	$003B_{16}$	00_{16}
(54)	UART2 RTS control register (U2RTSC)	$003E_{16}$	10000000
(55)	DMAC index and status register (DMAIS)	$003F_{16}$	00_{16}
(56)	DMAC channel x mode register 1 (DMAx1)	0040_{16}	00_{16}
(57)	DMAC channel x mode register 2 (DMAx2)	0041_{16}	00_{16}
(58)	DMAC channel x source register Low (DMAxSL)	0042_{16}	00_{16}
(59)	DMAC channel x source register High (DMAxSH)	0043_{16}	00_{16}
(60)	DMAC channel x destination register Low (DMAxDL)	0044_{16}	00_{16}
(61)	DMAC channel x destination register High (DMAxDH)	0045_{16}	00_{16}
(62)	DMAC channel x transfer count register Low (DMAxCL)	0046_{16}	00_{16}
(63)	DMAC channel x transfer count register High (DMAxCH)	0047_{16}	00_{16}
(64)	Data bus buffer register 0 (DBB0)	0048_{16}	00_{16}
(65)	Data bus buffer status register 0 (DBBS0)	0049_{16}	00_{16}
(66)	Data bus buffer control register 0 (DBBC0)	$004A_{16}$	00_{16}
(67)	Data bus buffer register 1 (DBB1)	$004C_{16}$	00_{16}
(68)	Data bus buffer status register 1 (DBBS1)	$004D_{16}$	00_{16}
(69)	Data bus buffer control register 1 (DBBC1)	$004E_{16}$	00_{16}
(70)	USB address register (USBA)	0050_{16}	00_{16}
(71)	USB power management register (USBPM)	0051_{16}	00_{16}
(72)	USB interrupt status register 1 (USBIS1)	0052_{16}	00_{16}
(73)	USB interrupt status register 2 (USBIS2)	0053_{16}	00_{16}
(74)	USB interrupt enable register 1 (USBIE1)	0054_{16}	FF_{16}
(75)	USB interrupt enable register 2 (USBIE2)	0055_{16}	00110011
(76)	USB frame number register Low (USBSOFL)	0056_{16}	00_{16}
(77)	USB frame number register High (USBSOFH)	0057_{16}	00_{16}
(78)	USB endpoint index register (USBINDEX)	0058_{16}	00_{16}
(79)	USB endpoint x IN control register (IN_CSR)	0059_{16}	00_{16}
(80)	USB endpoint x OUT control register (OUT_CSR)	$005A_{16}$	00_{16}
(81)	USB endpoint x IN max. packet size register (IN_MAXP) **(Note 1)**	$005B_{16}$	00001000
(82)	USB endpoint x OUT max. packet size register (OUT_MAXP) **(Note 1)**	$005C_{16}$	00001000
(83)	USB endpoint x OUT write count register Low (WRT_CNTL)	$005D_{16}$	00_{16}
(84)	USB endpoint x OUT write count register High (WRT_CNTH)	$005E_{16}$	00_{16}
(85)	USB endpoint FIFO mode register (USBFIFOMR)	$005F_{16}$	00_{16}
(86)	Flash memory control register (FMCR) **(Note 3)**	$006A_{16}$	00000001
(87)	Frequency synthesizer control register (FSC)	$006C_{16}$	01100000
(88)	Frequency synthesizer multiply register 1 (FSM1)	$006D_{16}$	FF_{16}
(89)	Frequency synthesizer multiply register 2 (FSM2)	$006E_{16}$	FF_{16}
(90)	Frequency synthesizer divide register (FSM2)	$006F_{16}$	FF_{16}
(91)	ROM code protect control register (ROMCP) **(Note 3)**	$FFC9_{16}$	FF_{16}
(92)	Processor status register	(PS)	XXXXX1XX
(93)	Program counter	(PC_H)	$FFFB_{16}$ contents
		(PC_L)	$FFFA_{16}$ contents

X : Not fixed

Notes 1: When using the endpoint 1, this contents are "01_{16}".

2: Since the initial values for other than above mentioned registers and RAM contents are indefinite at reset, they must be set.

3: The flash memory control register and the ROM code protect control register exists in the flash memory version only.

Fig. 63 Internal status at reset

CLOCK GENERATING CIRCUIT

The 7641 group has two built-in oscillation circuits. An oscillation circuit can be formed by connecting a resonator between X_{IN} and X_{OUT} (X_{CIN} and X_{COUT}). Use the circuit constants in accordance with the resonator manufacturer's recommended values. No external resistor is needed between X_{IN} and X_{OUT} since a feed-back resistor exists on-chip. (An external feed-back resistor may be needed depending on conditions.) However, an external feed-back resistor is needed between X_{CIN} and X_{COUT}.

When using an external clock, input the clocks to the X_{IN} or X_{CIN} pin and leave the X_{OUT} or X_{COUT} pin open.

Immediately after power on, only the X_{IN} oscillation circuit starts oscillating, and X_{CIN} and X_{COUT} pins function as I/O ports.

Frequency Control

The internal system clock can be selected among f_{SYN}, f(X_{IN}), f(X_{IN})/2, and f(X_{CIN}). The internal clock ϕ is half the frequency of internal system clock.

(1) f_{SYN} clock

This is made by the frequency synthesizer. f(X_{IN}) or f(X_{CIN}) can be selected as its input clock. See also section "FREQUENCY SYNTHESIZER".

(2) f(X_{IN}) clock

The frequency of internal system clock is the frequency of X_{IN} pin.

(3) f(X_{IN})/2 clock

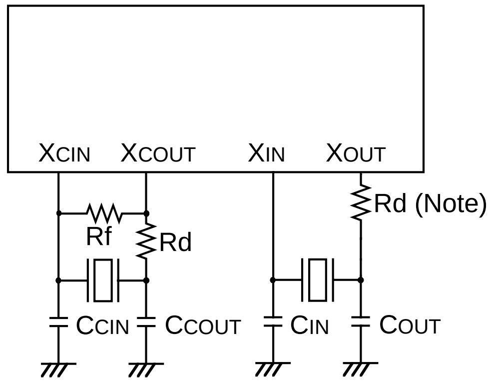
The frequency of internal system clock is half the frequency of X_{IN} pin.

(4) f(X_{CIN}) clock

The frequency of internal system clock is the frequency of X_{CIN} pin.

■Note

If you switch the oscillation between X_{IN} - X_{OUT} and X_{CIN} - X_{COUT}, stabilize both X_{IN} and X_{CIN} oscillations. The sufficient time is required for the X_{CIN} oscillation to stabilize, especially immediately after power on and at returning from the stop mode.



Note : Insert a damping resistor if required.
The resistance will vary depending on the oscillator and the oscillation drive capacity setting.
Use the value recommended by the maker of the oscillator.
Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip though a feedback resistor exists on-chip, insert a feedback resistor between X_{IN} and X_{OUT} following the instruction.

Fig. 64 Ceramic resonator or quartz-crystal oscillator external circuit

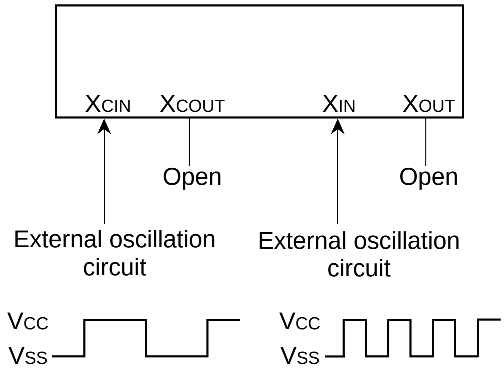


Fig. 65 External clock input circuit

(5) Low power dissipation mode

- The low power dissipation operation can be realized by stopping the main clock XIN when using f(XCIN) as the internal system clock. To stop the main clock, set the Main Clock (XIN-XOUT) Stop Bit of the CPU mode register A to "1".
- The low power dissipation operation can be realized by disabling the reversed amplifier when inputting external clocks to the XIN pin or XCIN pin. To disable the reversed amplifier, set the XCOUT Oscillation Drive Disable Bit (CCR5) or XOUT Oscillation Drive Disable Bit (CCR6) of the clock control register to "1".

Oscillation Control

(1) Stop mode

If the STP instruction is executed, the internal clock ϕ stops at "H" level, and XIN and XCIN oscillators stop. Then the timer 1 is set to "FF_{16}" and the internal clock ϕ divided by 8 is automatically selected as its count source. Additionally, the timer 2 is set to "01_{16}" and the timer 1's output is automatically selected as its count source.

Set the Timer 1 and Timer 2 Interrupt Enable Bits to disabled ("0") before executing the STP instruction. When using an external interrupt to release the stop mode, set the Interrupt Enable Bit to be used to enabled ("1") and the Interrupt Disable Flag (I) to "0".

Oscillator restarts at reset or when an external interrupt including USB resume interrupts is received, but the internal clock ϕ remains at "H" until the timer 2 underflows. The internal clock ϕ is supplied for the first time when the timer 2 underflows. Therefore make sure not to set the Timer 1 Interrupt Request Bit and Timer 2 Interrupt Request Bit to "1" before the STP instruction stops the oscillator.

(2) Wait mode

If the WIT instruction is executed, the internal clock ϕ stops at "H" level, but the oscillator does not stop. The internal clock ϕ restarts at reset or when an interrupt is received. Since the oscillator does not stop, normal operation can be started immediately after the internal clock ϕ is restarted.

Set the Interrupt Enable Bit to be used to release the wait mode to enabled ("1") and the Interrupt Disable Flag (I) to "0".

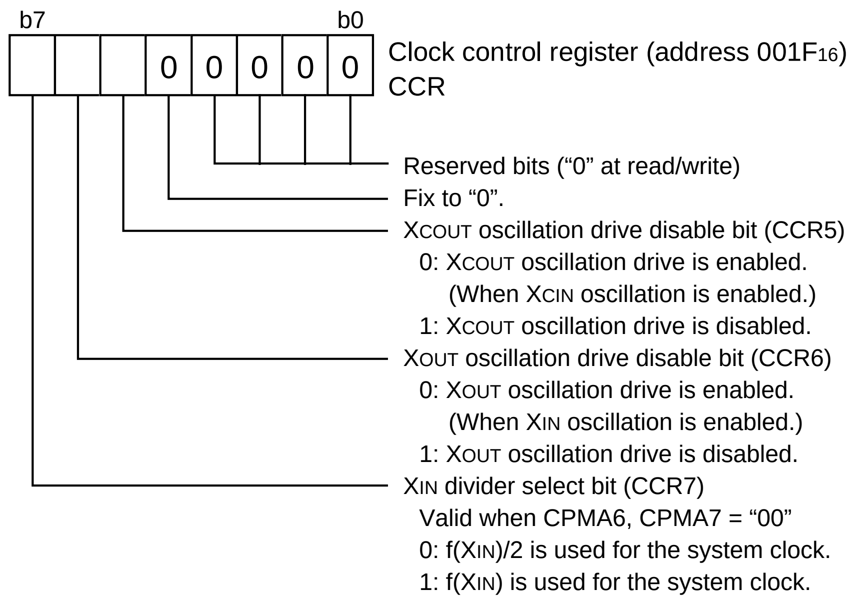


Fig. 66 Structure of clock control register

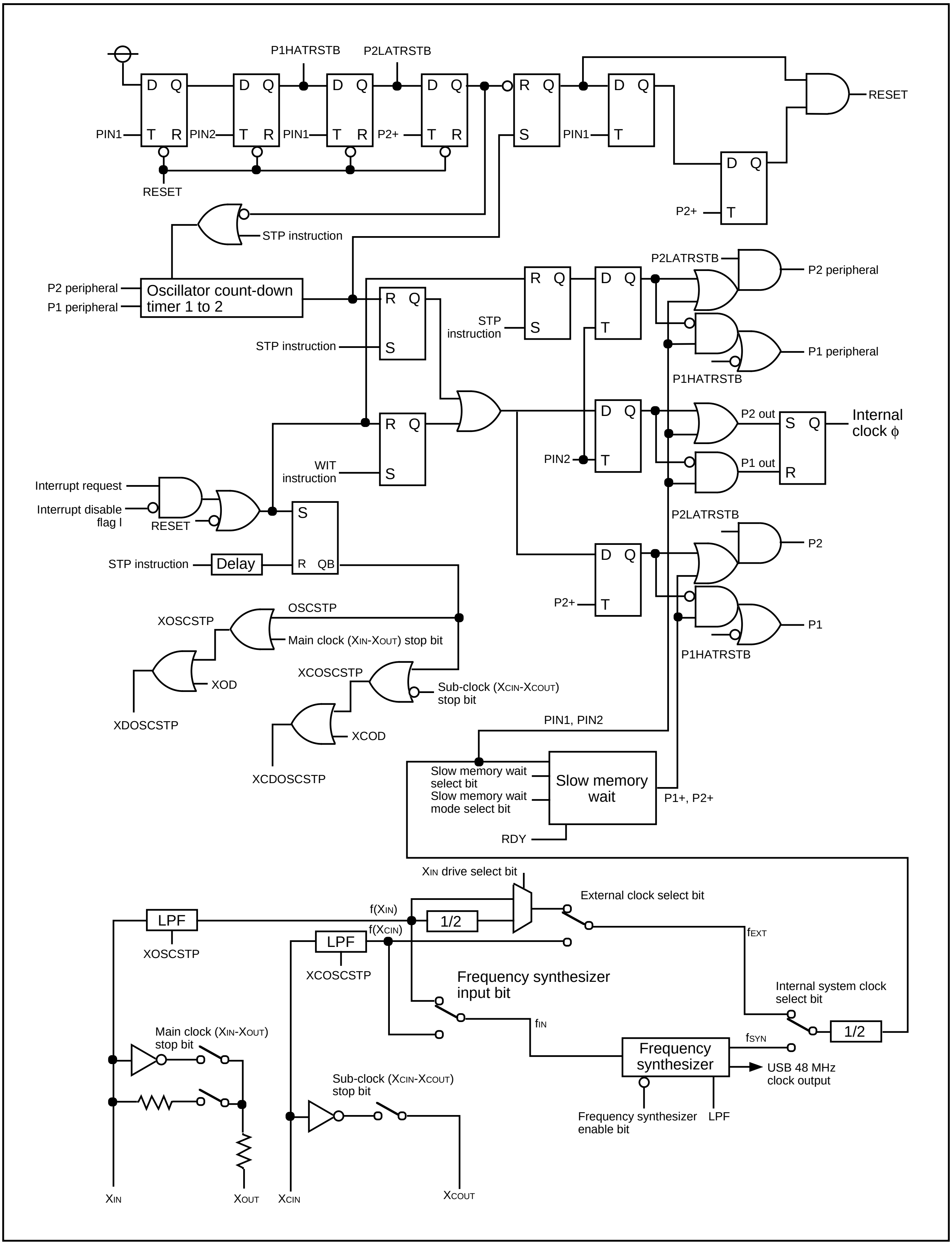
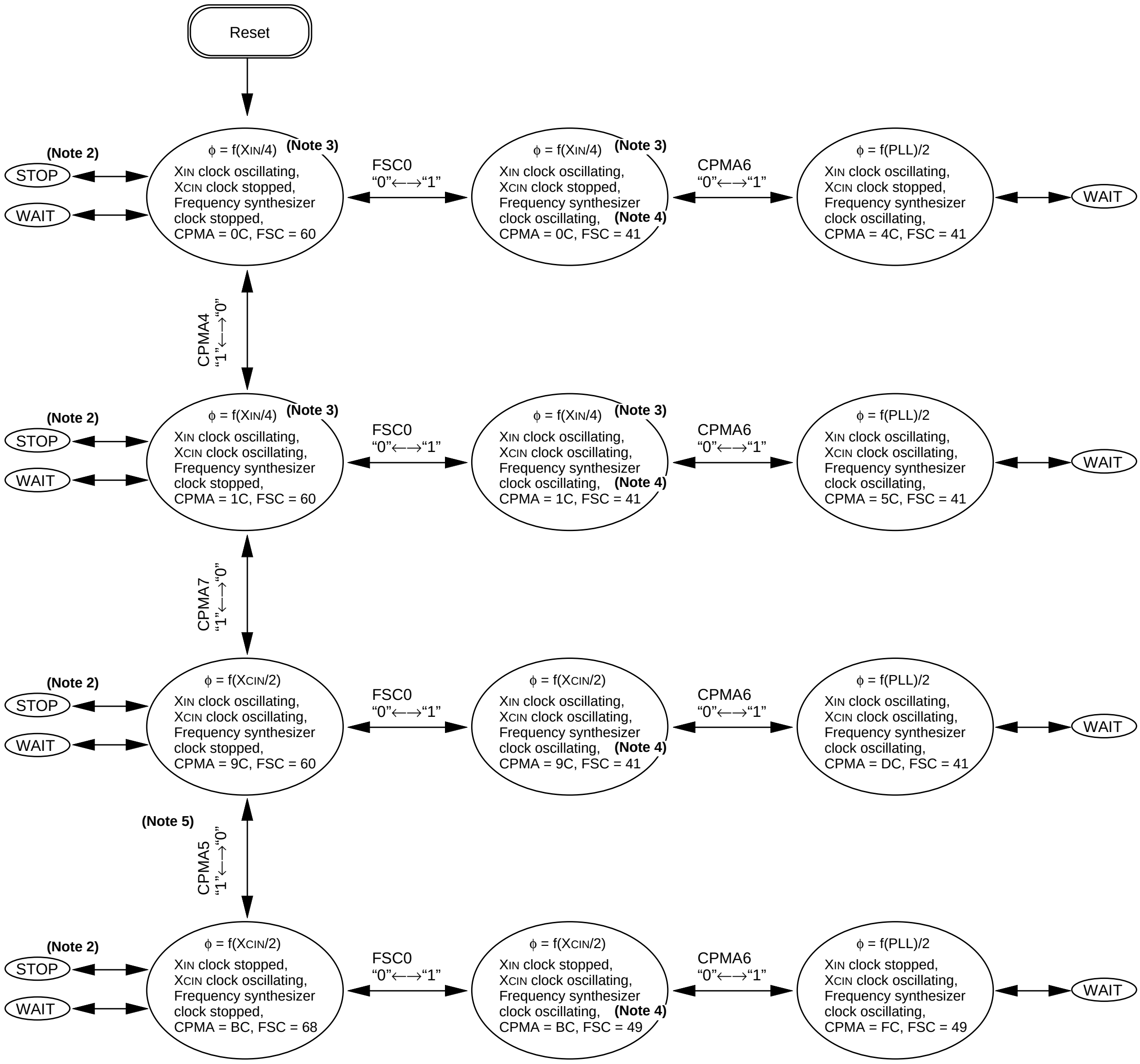


Fig. 67 Clock generating circuit block diagram



Notes 1 : Switch the mode by the allows shown between the mode blocks. (Do not switch between the modes directly without an allow.)

2 : In Stop mode, though the frequency synthesizer is not automatically disabled, the oscillator which sends clocks to the frequency synthesizer stops. Set the system clock and disable the frequency synthesizer before execution of the STP instruction.

3 : φ = f(XIN)/2 can be also used by setting the XIN divider select bit (CCR7) to "1". Then this diagram also applies to that case.

4 : The frequency synthesizer's input can be selected between XIN input and XCIN input regardless of the system clock. This diagram assumes the frequency synthesizer's input to be the system clock. Enable the oscillator to be used for the frequency synthesizer's input before enabling the frequency synthesizer.

5 : Select the XCIN input as the frequency synthesizer's input by setting the frequency synthesizer input bit (FSC3) to "1" before stopping XIN oscillation.

Remarks : This diagram assumes that:
- •Stack page is page 1
- •In single-chip mode
  (Depending on the CPU mode register A)
- •φ expresses the internal clock.

Fig. 68 State transitions of clock

PROCESSOR MODE

Single-chip mode, memory expansion mode, and microprocessor mode which is only in the mask ROM version can be selected by using the Processor Mode Bits of CPU mode register A (bits 0 and 1 of address 0000_{16}). In the memory expansion mode and microprocessor mode, a memory can be expanded externally via ports P0 to P3. In these modes, ports P0 to P3 lose their I/O port functions and become bus pins.
The port direction registers corresponding to those ports become external memory areas.

Table 9 Port functions in memory expansion mode and microprocessor mode

Port Name	Function
Port P0	Outputs low-order 8 bits of address.
Port P1	Outputs high-order 8 bits of address.
Port P2	Operates as I/O pins for data D_7 to D_0 (including instruction code).
Port P3	$P3_0$ is the RDY input pin. $P3_1$ and $P3_2$ function only as output pins $P3_3$ is the DMA_{OUT} output pin. $P3_4$ is the ϕ_{OUT} output pin. $P3_5$ is the $SYNC_{OUT}$ output pin. $P3_6$ is the $\overline{WR}$ output pin, and $P3_7$ is the $\overline{RD}$ output pin.
Port P4	$P4_0$ is the $\overline{EDMA}$ pin.

(1) Single-chip mode

Select this mode by resetting the MCU with CNV_{SS} connected to V_{SS}.

(2) Memory expansion mode

Select this mode by setting the Processor Mode Bits (b1, b0) to “01” in software with CNV_{SS} connected to V_{SS}. This mode enables external memory expansion while maintaining the validity of the internal ROM.

(3) Microprocessor mode

Select this mode by resetting the MCU with CNV_{SS} connected to V_{CC}, or by setting the Processor Mode Bits (b1, b0) to “10” in software with CNV_{SS} connected to V_{SS}. In the microprocessor mode, the internal ROM is no longer valid and an external memory must be used.
Do not set this mode in the flash memory version.

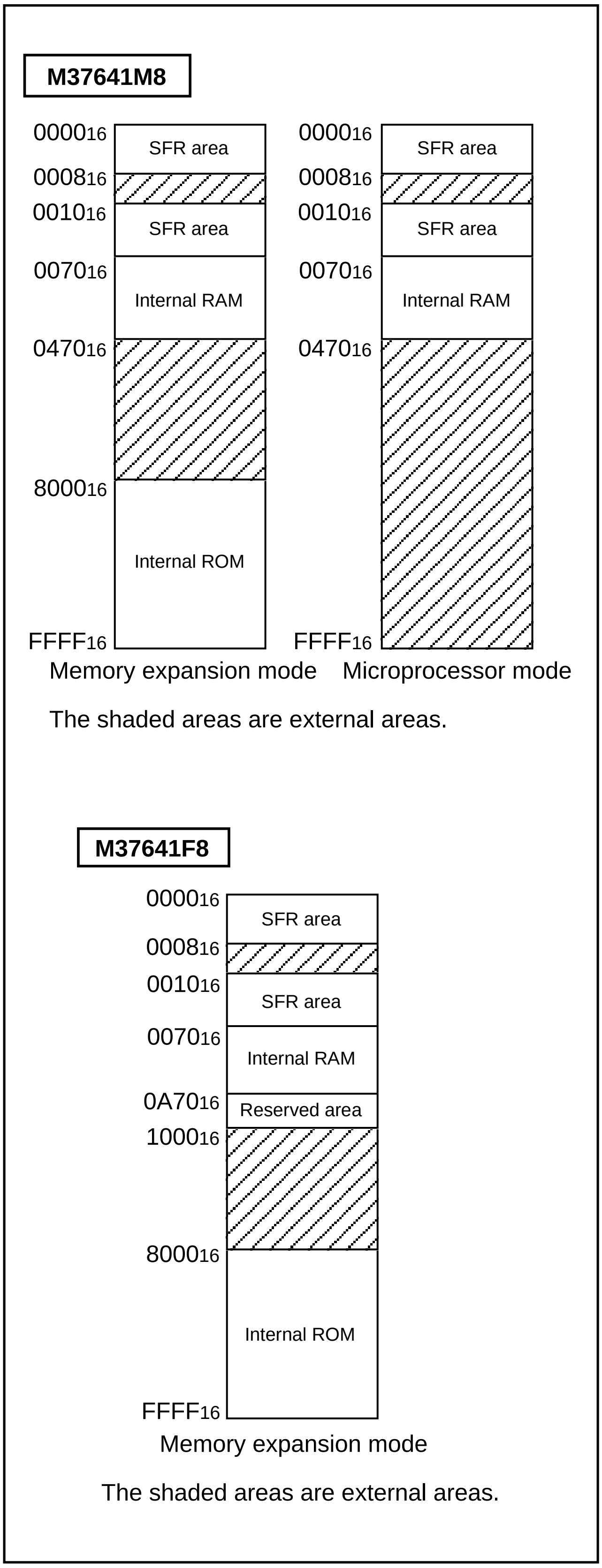
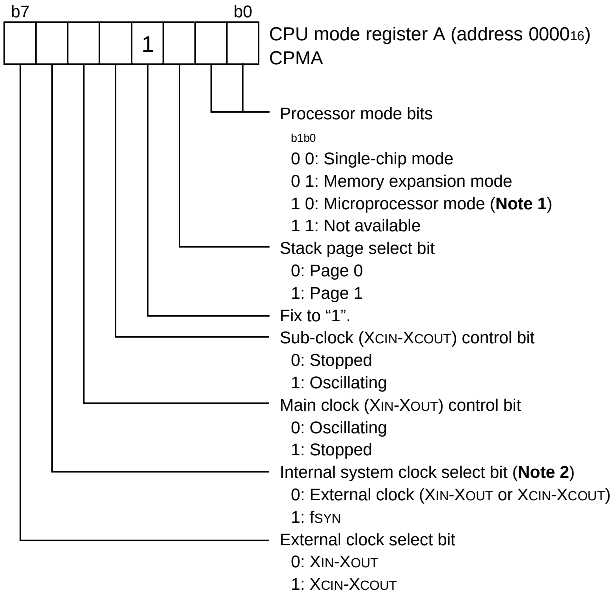


Fig. 69 Memory maps in processor modes other than single-chip mode



Notes 1: This is not available in the flash memory version.
2: When (CPMA 6, 7) = (0, 0), the internal system clock can be selected between $f(X_{IN})$ or $f(X_{IN})/2$ by CCR7.
The internal clock φ is the internal system clock divided by 2.

Fig. 70 Structure of CPU mode register A

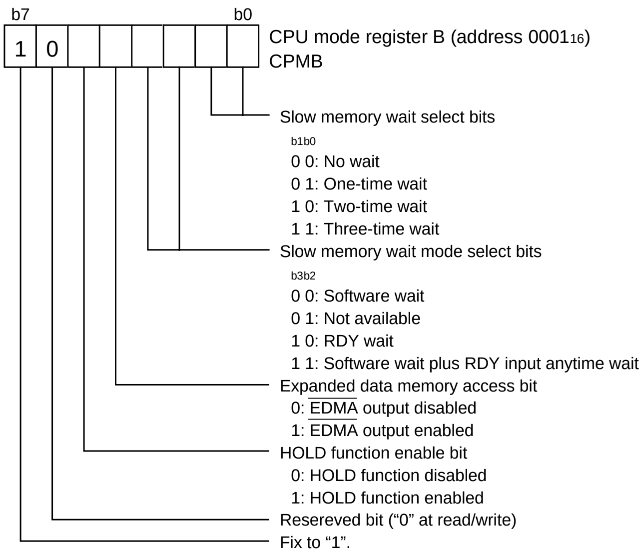


Fig. 71 Structure of CPU mode register B

Slow Memory Wait

The 7641 Group is equipped with the slow memory wait function (Software wait, RDY wait, and Extended RDY wait: software wait plus RDY input anytime wait) for easier interfacing with external devices that have long access times. The slow memory wait function can be enabled in the memory expansion mode and microprocessor mode. The appropriate wait mode is selected by setting bits 0 to 3 of CPU mode register B (address 0001_{16}). This function can extend the read cycle or write cycle only for access to an external memory. However, this wait function cannot be enabled for access to addresses 0008_{16} to $000F_{16}$.

(1) Software wait

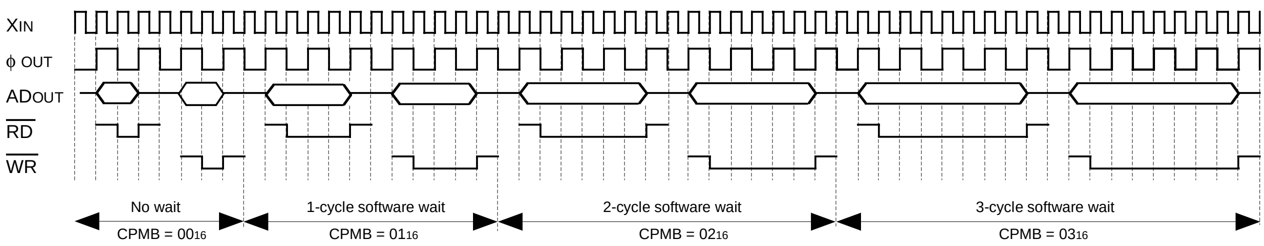
The software wait is selected by setting “00” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). Read/write cycles (“L” width of $\overline{RD}$ pin/$\overline{WR}$ pin) can be extended by one to three ϕ cycles. The number of cycles to be extended can be selected with the Slow Memory Wait Select Bits. When the software wait function is selected, the RDY pin status becomes invalid.

(2) RDY wait

RDY Wait is selected by setting “10” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). When a fixed time of “L” is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls), the MCU goes to the RDY state. The read/write cycle can then be extended by one to three ϕ cycles. The number of ϕ cycles to be added can be selected by the Slow Memory Wait Bits.

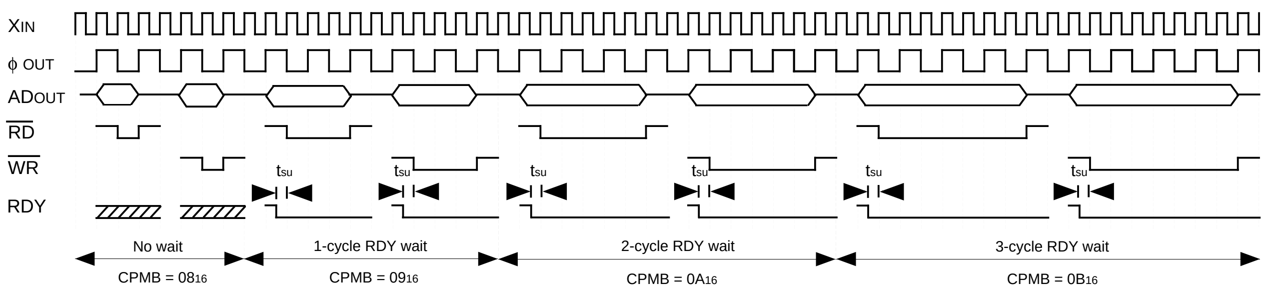
(3) Software wait + Extended RDY wait

Extended RDY Wait is selected by setting “11” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). The read/write cycle can be extended when a fixed time of “L” is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls). The RDY pin state is checked continually at each fall of ϕ cycle until the RDY pin goes to “H”. When “H” is input to the RDY pin, the wait is released within 1, 2, or 3 ϕ cycles (as selected with the Slow Memory Wait Bits).



Note: This diagram assumes ϕ = XIN/2.

Fig. 72 Software wait timing diagram



Note: This diagram assumes ϕ = XIN/2.

Fig. 73 RDY wait timing diagram

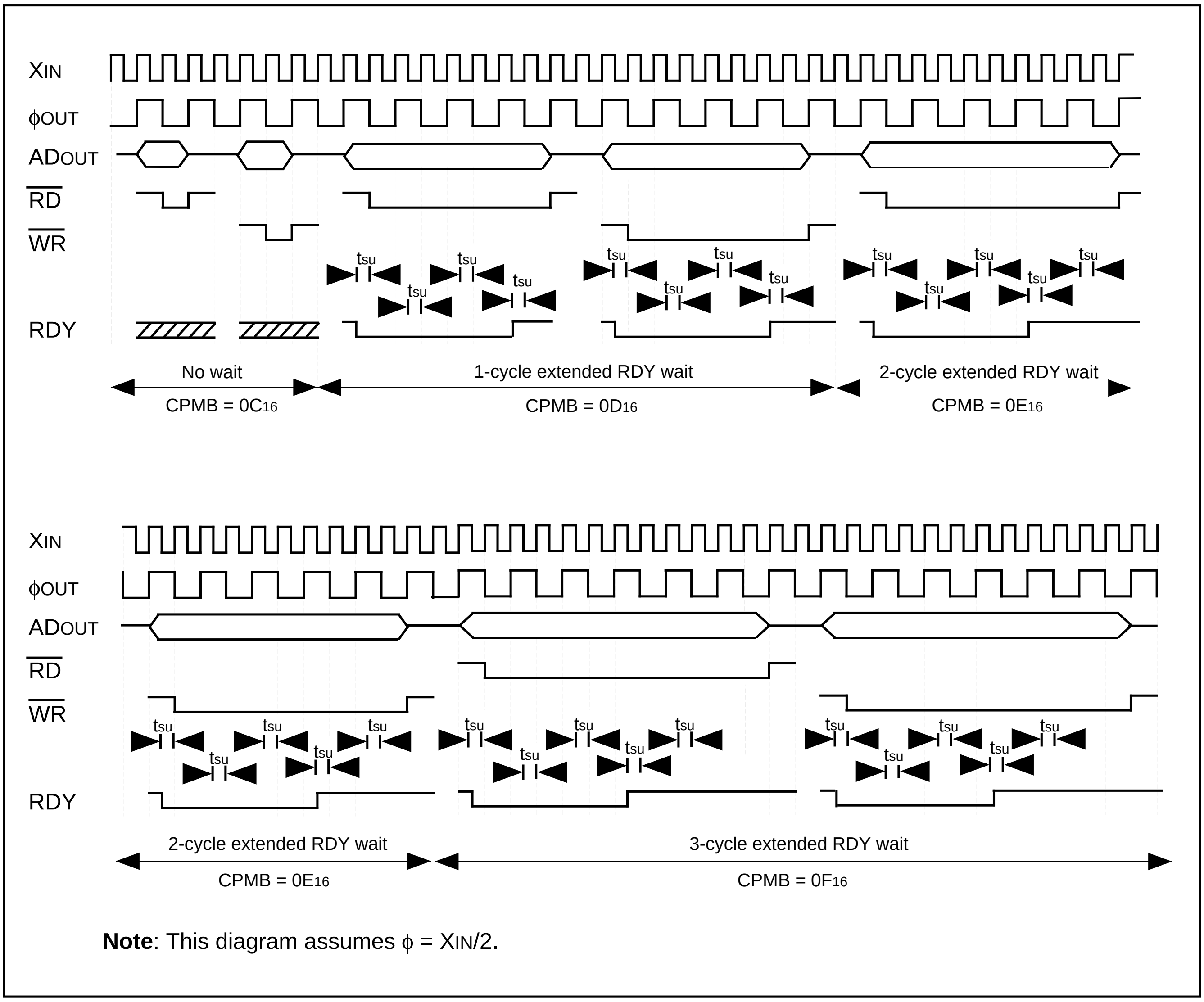


Fig. 74 Extended RDY wait (software wait plus RDY input anytime wait) timing diagram

HOLD Function

The HOLD function is used for systems that consist of external circuits that access MCU buses without use of the CPU (Central Processing Unit). The HOLD function is used to generate the timing in which the MCU will relinquish the bus from the CPU to the external circuits. To use the HOLD function, set the HOLD function Enable Bit of CPU mode register B (address 0001_{16}) to “1”. This function can be used with both the $\overline{HOLD}$ pin and the $\overline{HLDA}$ pin.

The $\overline{HOLD}$ signal is a signal from an external circuit requesting the MCU to relinquish use of the bus. When “L” level is input, the MCU goes to the HOLD state and remains so while the pin is at “L”. The oscillator does not stop oscillating during the HOLD state, therefore allowing the internal peripheral functions to operate during this time.

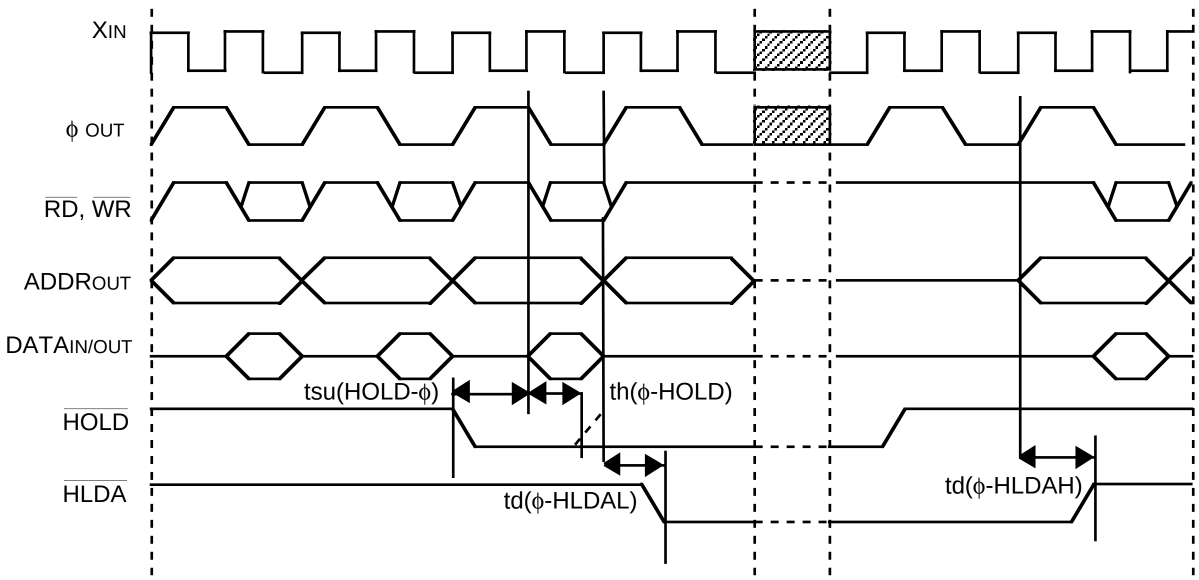
When the MCU relinquishes use of the bus, “L” level is output from the $\overline{HLDA}$ pin. The MCU makes ports P0 and P1 (address buses) and port P2 (data bus) tri-state outputs and holds port $P3_7$ ($\overline{RD}$ pin) and port $P3_6$ ($\overline{WR}$ pin) “H” level. Port $P3_4$ (ϕ_{OUT} pin) continues to oscillate. This function is not valid when the MCU is using the $\overline{IBF_1}$ function with the $\overline{HLDA}$ pin.

Expanded Data Memory Access

In Expanded Data Memory Access Mode, the MCU can access a data area larger than 64 Kbytes with the LDA ($zz), Y (indirect Y) instruction and the STA ($zz), Y (indirect Y) instruction.

To use this mode, set the Expanded Data Memory Access Bit of CPU mode register B (address 0001_{16}) to “1”. In this case, port $P4_0$ (EDMA pin) goes “L” level during the read/write cycle of the LDA or STA instruction.

The determination of which bank to access is done by using an I/O port to represent expanded addresses exceeding address bus AB_{15}. For example, when accessing 4 banks, use two I/O ports to represent address buses AB_{16} and AB_{17}.



Note: This diagram assumes $\phi = X_{IN}/2$.

Fig. 75 Hold function timing diagram

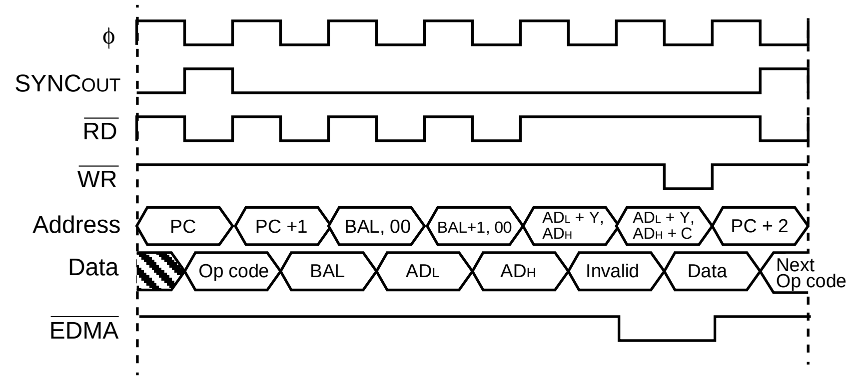


Fig. 76 STA ($ zz), Y instruction sequence when EDMA enabled

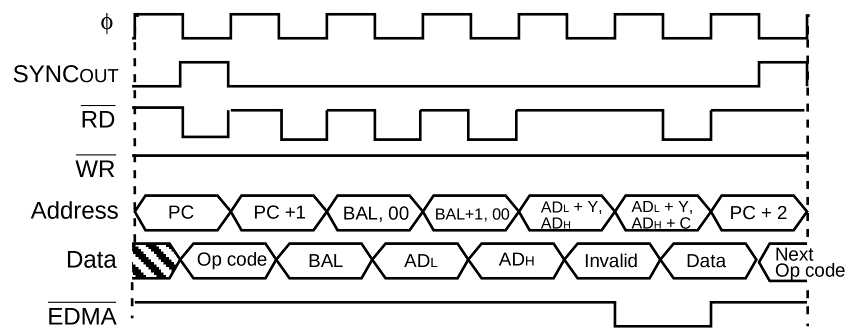


Fig. 77 LDA ($ zz), Y instruction sequence when EDMA enabled and T flag = “0”

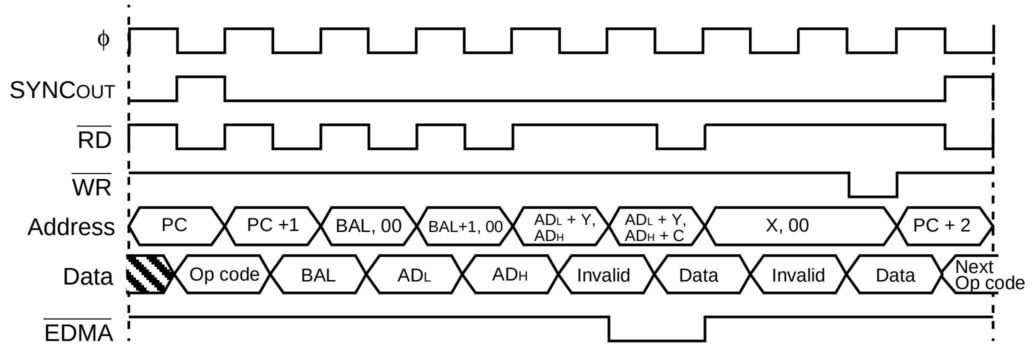


Fig. 78 LDA ($ zz), Y instruction sequence when EDMA enabled and T flag = “1”

FLASH MEMORY MODE

The M37641F8FP/HP (flash memory version) has an internal new DINOR (DIvided bit line NOR) flash memory that can be rewritten with a single power source when V_{CC} is 5 V, and 2 power sources when V_{PP} is 5 V and V_{CC} is 3.3 V in the CPU rewrite and standard serial I/O modes.

For this flash memory, three flash memory modes are available in which to read, program, and erase: the parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and the CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU).

Summary

Table 10 lists the summary of the M37641F8 (flash memory version).

This flash memory version has some blocks on the flash memory as shown in Figure 79 and each block can be erased. The flash memory is divided into User ROM area and Boot ROM area.

In addition to the ordinary User ROM area to store the MCU operation control program, the flash memory has a Boot ROM area that is used to store a program to control rewriting in CPU rewrite and standard serial I/O modes. This Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. However, the user can write a rewrite control program in this area that suits the user's application system. This Boot ROM area can be rewritten in only parallel I/O mode.

Table 10 Summary of M37641F8 (flash memory version)

Item		Specifications
Power source voltage (For Program/Erase)		Vcc = 3.00 – 3.60 V, 4.50 – 5.25 V ($f(X_{IN})$ = 24 MHz, ϕ = 6 MHz) **(Note 1)**
V_{PP} voltage (For Program/Erase)		V_{PP} = 4.50 – 5.25 V
Flash memory mode		3 modes; Flash memory can be manipulated as follows: (1) CPU rewrite mode: Manipulated by the Central Processing Unit (CPU) (2) Parallel I/O mode: Manipulated using an external programmer **(Note 2)** (3) Standard serial I/O mode: Manipulated using an external programmer **(Note 2)**.
Erase block division	User ROM area	See Figure 79.
	Boot ROM area	1 block (4 Kbytes) (**Note 3**)
Program method		Byte program
Erase method		Batch erasing/Block erasing
Program/Erase control method		Program/Erase control by software command
Number of commands		6 commands
Number of program/Erase times		100 times
ROM code protection		Available in parallel I/O mode and standard serial I/O mode

Notes 1: After programming/erasing at Vcc = 3.0 to 3.6 V, the MCU can operate only at Vcc = 3.0 to 3.6 V.
After programming/erasing at Vcc = 4.5 to 5.25 V or programming/erasing with the exclusive external equipment flash programmer, the MCU can operate at both Vcc = 3.0 to 3.6 V and 4.15 to 5.25 V.

2: In the parallel I/O mode or the standard serial I/O mode, use the exclusive external equipment flash programmer which supports the 7641 Group (flash memory version).

3: The Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. This Boot ROM area can be rewritten in only parallel I/O mode.

(1) CPU Rewrite Mode

In CPU rewrite mode, the internal flash memory can be operated on (read, program, or erase) under control of the Central Processing Unit (CPU).
In CPU rewrite mode, only the User ROM area shown in Figure 79 can be rewritten; the Boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the User ROM area and each block area.
The control program for CPU rewrite mode can be stored in either User ROM or Boot ROM area. In the CPU rewrite mode, because the flash memory cannot be read from the CPU, the rewrite control program must be transferred to internal RAM area to be executed before it can be executed.

Microcomputer Mode and Boot Mode

The control program for CPU rewrite mode must be written into the User ROM or Boot ROM area in parallel I/O mode beforehand. (If the control program is written into the Boot ROM area, the standard serial I/O mode becomes unusable.)
See Figure 79 for details about the Boot ROM area.
Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVss pin low. In this case, the CPU starts operating using the control program in the User ROM area.
When the microcomputer is reset by pulling the $P3_6$ ($\overline{CE}$) pin high, the $P8_1$ (SCLK) pin high, the CNVss pin high, the CPU starts operating using the control program in the Boot ROM area. This mode is called the "Boot" mode.

Block Address

Block addresses refer to the maximum address of each block. These addresses are used in the block erase command.

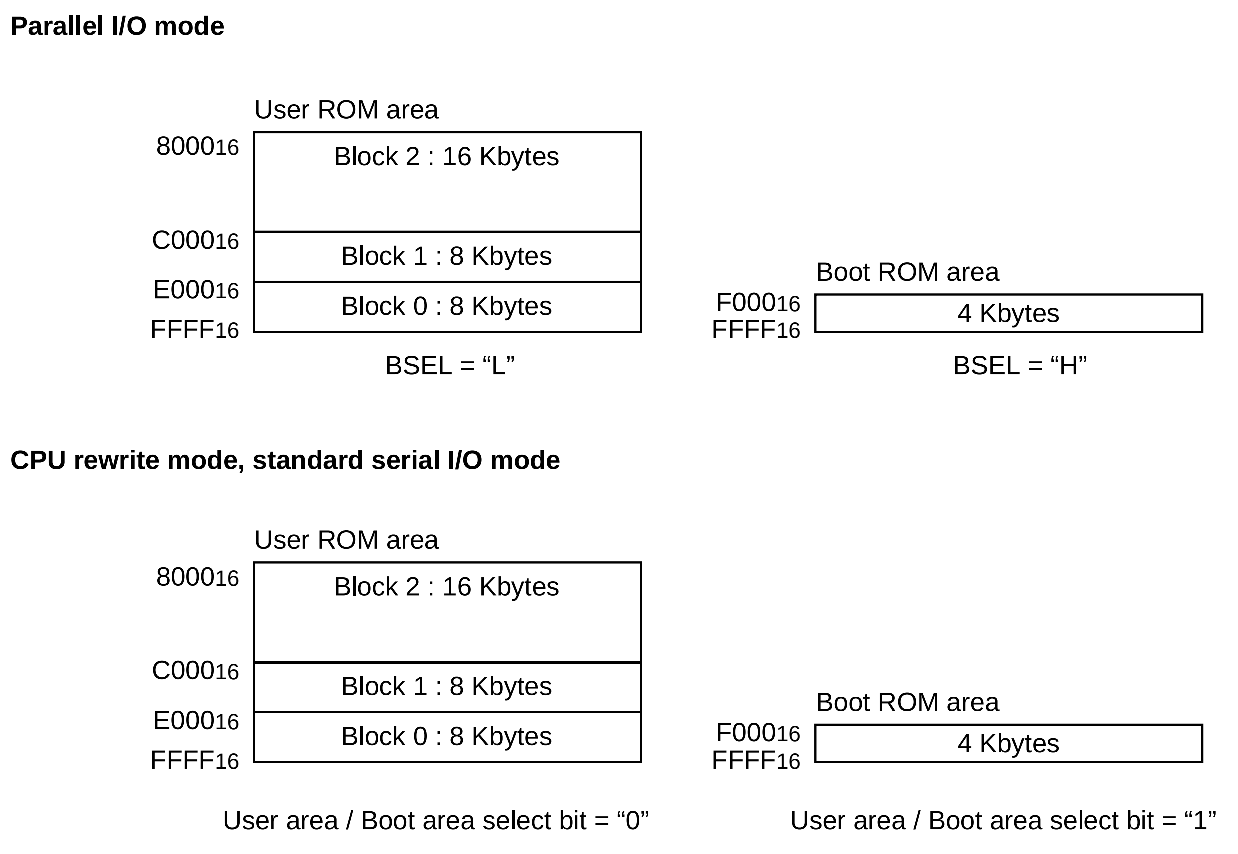


Fig. 79 Block diagram of built-in flash memory

Outline Performance (CPU Rewrite Mode)

CPU rewrite mode is usable in the single-chip, memory expansion or Boot mode. The only User ROM area can be rewritten in CPU rewrite mode.

In CPU rewrite mode, the CPU erases, programs and reads the internal flash memory by executing software commands. This rewrite control program must be transferred to a memory such as the internal RAM before it can be executed.

The MCU enters CPU rewrite mode by applying 4.50 V to 5.25 V to the CNV$_{SS}$ pin and setting "1" to the CPU Rewrite Mode Select Bit (bit 1 of address $006A_{16}$). Software commands are accepted once the mode is entered.

Use software commands to control program and erase operations. Whether a program or erase operation has terminated normally or in error can be verified by reading the status register.

Figure 80 shows the flash memory control register.

Bit 0 is the RY/$\overline{BY}$ status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is "0" (busy). Otherwise, it is "1" (ready).

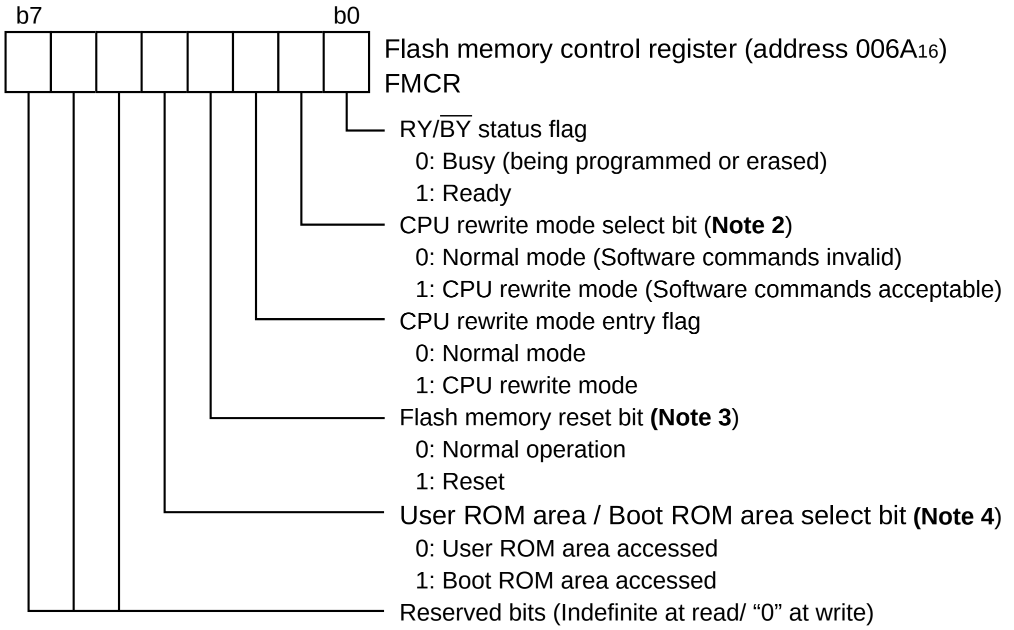
Bit 1 is the CPU Rewrite Mode Select Bit. When this bit is set to "1", the MCU enters CPU rewrite mode. Software commands are accepted once the mode is entered. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, use the control program in a memory other than internal flash memory for write to bit 1. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. The bit can be set to "0" by only writing "0".

Bit 2 is the CPU Rewrite Mode Entry Flag. This flag indicates "1" in CPU rewrite mode, so that reading this flag can check whether CPU rewrite mode has been entered or not.

Bit 3 is the flash memory reset bit used to reset the control circuit of internal flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. When the CPU Rewrite Mode Select Bit is "1", setting "1" for this bit resets the control circuit. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. To release the reset, it is necessary to set this bit to "0".

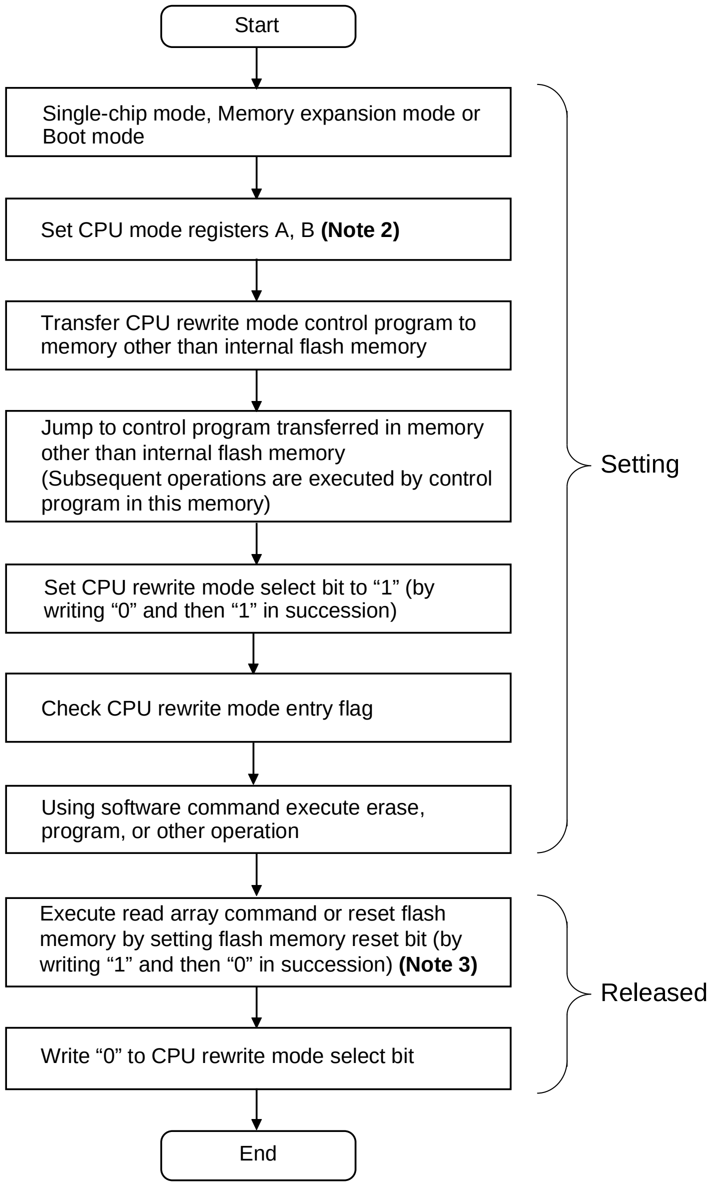
Bit 4 is the User Area/Boot Area Select Bit. When this bit is set to "1", Boot ROM area is accessed, and CPU rewrite mode in Boot ROM area is available. In Boot mode, this bit is set to "1" automatically. Reprogramming of this bit must be in a memory other than internal flash memory.

Figure 81 shows a flowchart for setting/releasing CPU rewrite mode.



Notes **1**: The contents of flash memory control register are "XXX00001" just after reset release.
2: For this bit to be set to "1", the user needs to write "0" and then "1" to it in succession. If it is not this procedure, this bit will not be set to "1". Additionally, it is required to ensure that no interrupt will be generated during that interval.
Use the control program in the area except the built-in flash memory for write to this bit.
3: This bit is valid when the CPU rewrite mode select bit is "1". Set this bit 3 to "0" subsequently after setting bit 3 to "1".
4: Use the control program in the area except the built-in flash memory for write to this bit.

Fig. 80 Structure of flash memory control register



Notes 1: When starting the MCU in the single-chip mode or memory expansion mode, supply 4.5 V to 5.25 V to the CNVss pin until checking the CPU rewrite mode entry flag.
2: Set the main clock as follows depending on the X_{IN} divider select bit of clock control register (bit 7 of address $001F_{16}$):
When X_{IN} divider select bit = “0” ($\phi = f(X_{IN})/4$), the main clock is 24 MHz or less
When X_{IN} divider select bit = “1” ($\phi = f(X_{IN})/2$), the main clock is 12 MHz or less.
3: Before exiting the CPU rewrite mode after completing erase or program operation, always be sure to execute the read array command or reset the flash memory.

Fig. 81 CPU rewrite mode set/release flowchart

Notes on CPU Rewrite Mode

The below notes applies when rewriting the flash memory in CPU rewrite mode.

●Operation speed

During CPU rewrite mode, set the internal clock ϕ to 6 MHz or less using the X_{IN} Divider Select Bit (bit 7 of address $001F_{16}$).

●Instructions inhibited against use

The instructions which refer to the internal data of the flash memory cannot be used during CPU rewrite mode .

●Interrupts inhibited against use

The interrupts cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory.

●Reset

Reset is always valid. When CNV_{SS} is “H” at reset release, the program starts from the address stored in addresses $FFFA_{16}$ and $FFFB_{16}$ of the boot ROM area in order that CPU may start in boot mode.

Software Commands (CPU Rewrite Mode)

Table 11 lists the software commands.
After setting the CPU Rewrite Mode Select Bit of the flash memory control register to “1”, execute a software command to specify an erase or program operation.
Each software command is explained below.

●Read Array Command (FF_{16})

The read array mode is entered by writing the command code “FF_{16}” in the first bus cycle. When an address to be read is input in one of the bus cycles that follow, the contents of the specified address are read out at the data bus (DB_0 to DB_7).
The read array mode is retained intact until another command is written.

●Read Status Register Command (70_{16})

The read status register mode is entered by writing the command code “70_{16}” in the first bus cycle. The contents of the status register are read out at the data bus (DB_0 to DB_7) by a read in the second bus cycle.
The status register is explained in the next section.

●Clear Status Register Command (50_{16})

This command is used to clear the bits SR4 and SR5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code “50_{16}” in the first bus cycle.

●Program Command (40_{16})

Program operation starts when the command code “40_{16}” is written in the first bus cycle. Then, if the address and data to program are written in the 2nd bus cycle, program operation (data programming and verification) will start.
Whether the write operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag of the flash memory control register. When the program starts, the read status register mode is entered automatically and the contents of the status register is read at the data bus (DB_0 to DB_7). The status register bit 7 (SR7) is set to “0” at the same time the write operation starts and is returned to “1” upon completion of the write operation. In this case, the read status register mode remains active until the next command is written.
The RY/$\overline{BY}$ Status Flag is “0” (busy) during write operation and “1” (ready) when the write operation is completed as is the status register bit 7.
At program end, program results can be checked by reading bit 4 (SR4) of the status register.

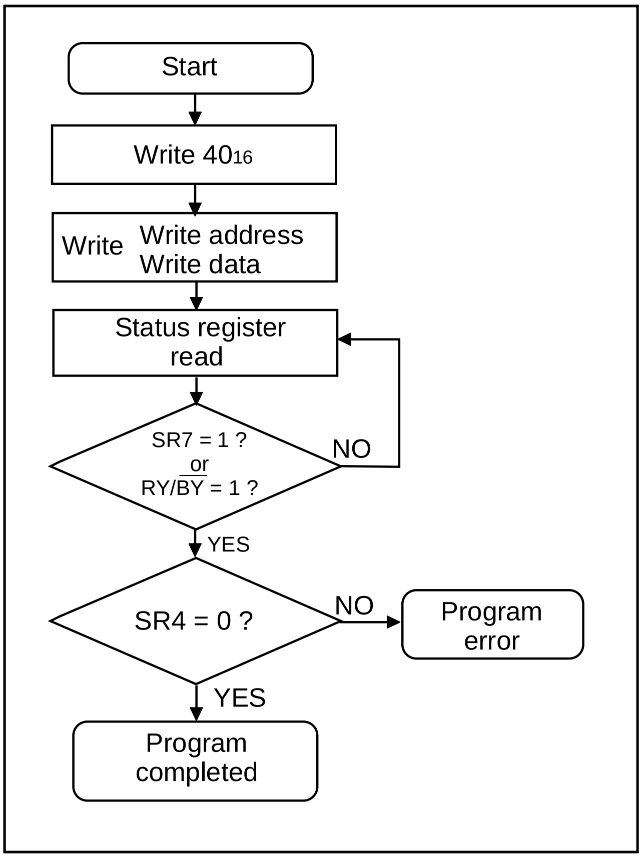


Fig. 82 Program flowchart

Table 11 List of software commands (CPU rewrite mode)

Command	Cycle number	First bus cycle			Second bus cycle		
		Mode	Address	Data (DB_0 to DB_7)	Mode	Address	Data (DB_0 to DB_7)
Read array	1	Write	X **(Note 4)**	FF_{16}			
Read status register	2	Write	X	70_{16}	Read	X	SRD **(Note 1)**
Clear status register	1	Write	X	50_{16}			
Program	2	Write	X	40_{16}	Write	WA **(Note 2)**	WD **(Note 2)**
Erase all blocks	2	Write	X	20_{16}	Write	X	20_{16}
Block erase	2	Write	X	20_{16}	Write	BA **(Note 3)**	$D0_{16}$

Notes 1: SRD = Status Register Data
2: WA = Write Address, WD = Write Data
3: BA = Block Address to be erased (Input the maximum address of each block.)
4: X denotes a given address in the User ROM area .

●Erase All Blocks Command (20_{16}/20_{16})

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "20_{16}" in the second bus cycle that follows, the operation of erase all blocks (erase and erase verify) starts.

Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag of flash memory control register. When the erase all blocks operation starts, the read status register mode is entered automatically and the contents of the status register can be read out at the data bus (DB_0 to DB_7). The status register bit 7 (SR7) is set to "0" at the same time the erase operation starts and is returned to "1" upon completion of the erase operation. In this case, the read status register mode remains active until another command is written.

The RY/$\overline{BY}$ Status Flag is "0" during erase operation and "1" when the erase operation is completed as is the status register bit 7 (SR7).

After the erase all blocks end, erase results can be checked by reading bit 5 (SRS) of the status register. For details, refer to the section where the status register is detailed.

●Block Erase Command (20_{16}/$D0_{16}$)

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "$D0_{16}$" and the blobk address in the second bus cycle that follows, the block erase (erase and erase verify) operation starts for the block address of the flash memory to be specified.

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag of flash memory control register. At the same time the block erase operation starts, the read status register mode is automatically entered, so that the contents of the status register can be read out. The status register bit 7 (SR7) is set to "0" at the same time the block erase operation starts and is returned to "1" upon completion of the block erase operation. In this case, the read status register mode remains active until the read array command (FF_{16}) is written.

The RY/$\overline{BY}$ Status Flag is "0" during block erase operation and "1" when the block erase operation is completed as is the status register bit 7.

After the block erase ends, erase results can be checked by reading bit 5 (SRS) of the status register. For details, refer to the section where the status register is detailed.

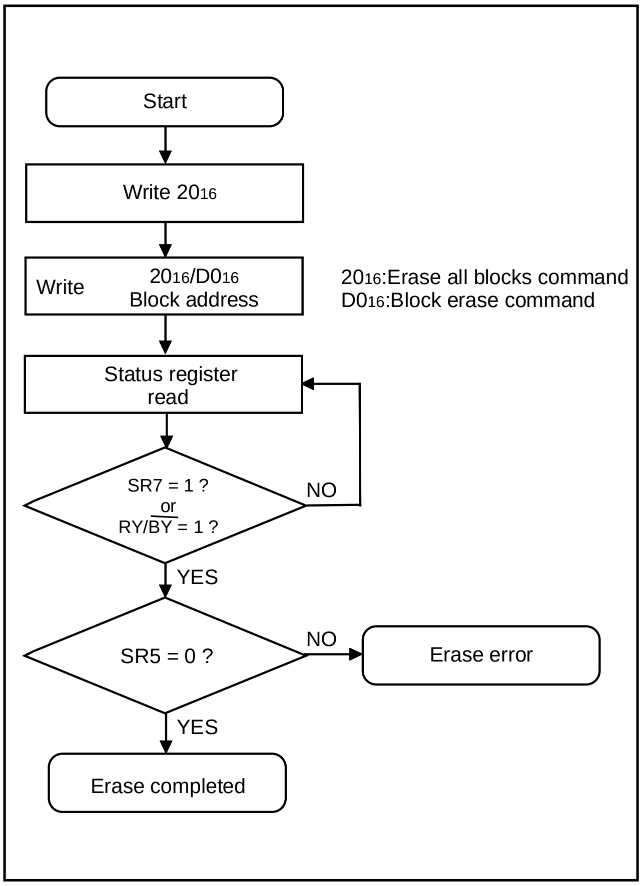


Fig. 83 Erase flowchart

Status Register (SRD)

The status register shows the operating status of the flash memory and whether erase operations and programs ended successfully or in error. It can be read in the following ways:

(1) By reading an arbitrary address from the User ROM area after writing the read status register command (70_{16})

(2) By reading an arbitrary address from the User ROM area in the period from when the program starts or erase operation starts to when the read array command (FF_{16}) is input.

Also, the status register can be cleared by writing the clear status register command (50_{16}).
After reset, the status register is set to "80_{16}".
Table 12 shows the status register. Each bit in this register is explained below.

•Sequencer status (SR7)
The sequencer status indicates the operating status of the flash memory. This bit is set to "0" (busy) during write or erase operation and is set to "1" when these operations ends.
After power-on, the sequencer status is set to "1" (ready).

•Erase status (SR5)
The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

•Program status (SR4)
The program status indicates the operating status of write operation. When a write error occurs, it is set to "1".
The program status is set to "0" when it is cleared.

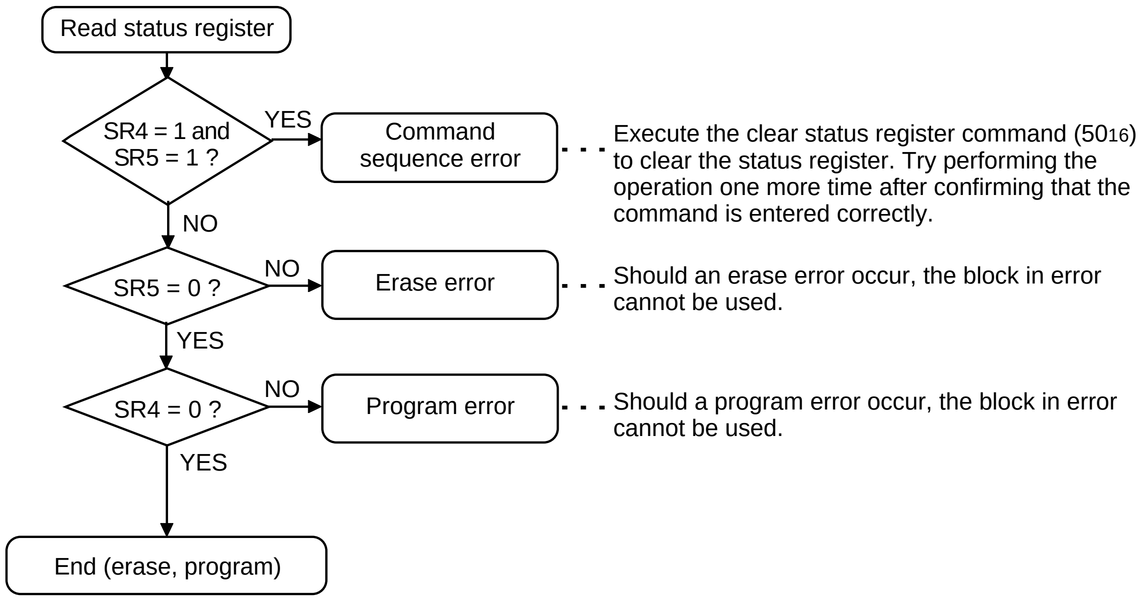
If "1" is written for any of the SR5 and SR4 bits, the program, erase all blocks, and block erase commands are not accepted. Before executing these commands, execute the clear status register command (50_{16}) and clear the status register.
Also, if any commands are not correct, both SR5 and SR4 are set to "1".

Table 12 Definition of each bit in status register (SRD)

Symbol	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 84 shows a full status check flowchart and the action to be taken when each error occurs.



Note: When one of SR5 and SR4 is set to "1", none of the read aray, the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

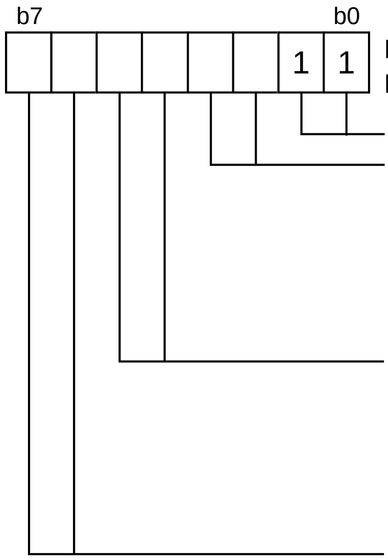
Fig. 84 Full status check flowchart and remedial procedure for errors

Functions To Inhibit Rewriting Flash Memory Version

To prevent the contents of internal flash memory from being read out or rewritten easily, this MCU incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

●ROM Code Protect Function (in Pararell I/O Mode)

The ROM code protect function is the function to inhibit reading out or modifying the contents of internal flash memory by using the ROM code protect control (address $FFC9_{16}$) in parallel I/O mode. Figure 85 shows the ROM code protect control (address $FFC9_{16}$). (This address exists in the User ROM area.)
If one or both of the pair of ROM Code Protect Bits is set to “0”, the ROM code protect is turned on, so that the contents of internal flash memory are protected against readout and modification. The ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.
If both of the two ROM Code Protect Reset Bits are set to “00”, the ROM code protect is turned off, so that the contents of internal flash memory can be read out or modified. Once the ROM code protect is turned on, the contents of the ROM Code Protect Reset Bits cannot be modified in parallel I/O mode. Use the serial I/O or CPU rewrite mode to rewrite the contents of the ROM Code Protect Reset Bits.



ROM code protect control (address $FFC9_{16}$) (**Note 1**)
ROMCP

Reserved bits (“1” at read/write)

ROM code protect level 2 set bits (ROMCP2) (**Notes 2, 3**)
b3b2
0 0: Protect enabled
0 1: Protect enabled
1 0: Protect enabled
1 1: Protect disabled

ROM code protect reset bits (**Note 4**)
b5b4
0 0: Protect removed
0 1: Protect set bits effective
1 0: Protect set bits effective
1 1: Protect set bits effective

ROM code protect level 1 set bits (ROMCP1) (**Note 2**)
b7b6
0 0: Protect enabled
0 1: Protect enabled
1 0: Protect enabled
1 1: Protect disabled

Notes 1: This area is on the ROM in the mask ROM version.
2: When ROM code protect is turned on, the internal flash memory is protected against readout or modification in parallel I/O mode.
3: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LSI tester, etc. also is inhibited.
4: The ROM code protect reset bits can be used to turn off ROM code protect level 1 and ROM code protect level 2. However, since these bits cannot be modified in parallel I/O mode, they need to be rewritten in standard serial I/O mode or CPU rewrite mode.

Fig. 85 Structure of ROM code protect control

ID Code Check Function (in Standard serial I/O mode)

Use this function in standard serial I/O mode. When the contents of the flash memory are not blank, the ID code sent from the programmer is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the programmer are not accepted. The ID code consists of 8-bit data, and its areas are $FFC2_{16}$ to $FFC8_{16}$. Write a program which has had the ID code preset at these addresses to the flash memory.

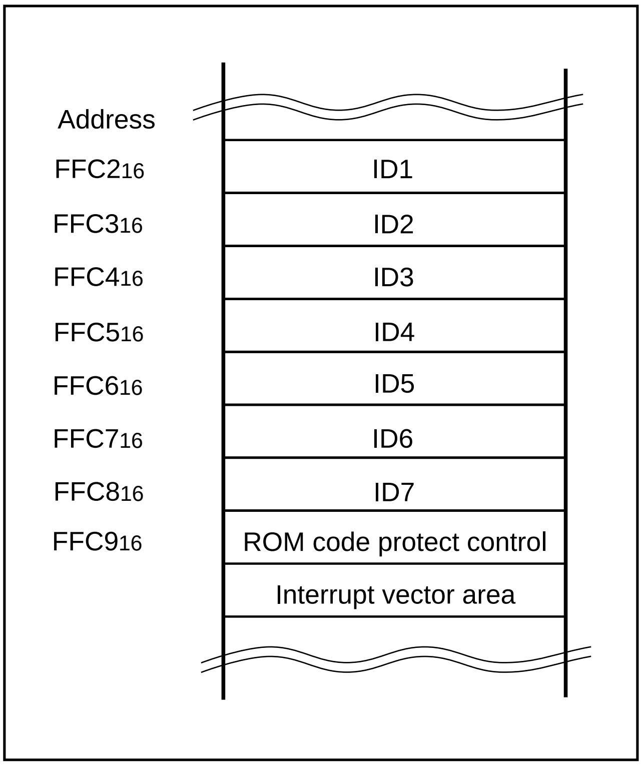


Fig. 86 ID code store addresses

(2) Parallel I/O Mode

Parallel I/O mode is the mode which parallel output and input software command, address, and data required for the operations (read, program, erase, etc.) to a built-in flash memory. Use the exclusive external equipment flash programmer which supports the 7641 Group (flash memory version). Refer to each programmer maker's handling manual for the details of the usage.

User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 79 can be rewritten. Both areas of flash memory can be operated on in the same way.
Program and block erase operations can be performed in the user ROM area. The user ROM area and its block is shown in Figure 79.
The boot ROM area is 4 Kbytes in size. It is located at addresses $F000_{16}$ through $FFFF_{16}$. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)
In the Boot ROM area, an erase block operation is applied to only one 4 Kbyte block. The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the device in standard serial I/O mode, you do not need to write to the boot ROM area.

(3) Standard serial I/O Mode

The standard serial I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is clock synchronized serial. This mode requires the exclusive external equipment (flash programmer).

The standard serial I/O mode is different from the parallel I/O mode in that the CPU controls flash memory rewrite (uses the CPU rewrite mode), rewrite data input and so forth. The standard serial I/O mode is started by connecting “H” to the $P3_6$ ($\overline{CE}$) pin and “H” to the $P8_1$ (SCLK) pin and “H” to the CNV_{SS} pin (apply 4.5 V to 5.25 V to Vpp from an external source), and releasing the reset operation. (In the ordinary microcomputer mode, set CNVss pin to “L” level.)

This control program is written in the Boot ROM area when the product is shipped from Mitsubishi. Accordingly, make note of the fact that the standard serial I/O mode cannot be used if the Boot ROM area is rewritten in parallel I/O mode. Figures 87 and 88 show the pin connections for the standard serial I/O mode.

In standard serial I/O mode, serial data I/O uses the four serial I/O pins SCLK, SRXD, STXD and $\overline{SRDY}$ (BUSY). The SCLK pin is the transfer clock input pin through which an external transfer clock is input. The STXD pin is for CMOS output. The $\overline{SRDY}$ (BUSY) pin outputs “L” level when ready for reception and “H” level when reception starts.

Serial data I/O is transferred serially in 8-bit units.

In standard serial I/O mode, only the User ROM area shown in Figure 79 can be rewritten. The Boot ROM area cannot.

In standard serial I/O mode, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit (programmer) are not accepted unless the ID code matches.

Outline Performance (Standard Serial I/O Mode)

In standard serial I/O mode, software commands, addresses and data are input and output between the MCU and peripheral units (flash programer, etc.) using 4-wire clock-synchronized serial I/O.

In reception, software commands, addresses and program data are synchronized with the rise of the transfer clock that is input to the SCLK pin, and are then input to the MCU via the SRXD pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the STXD pin.

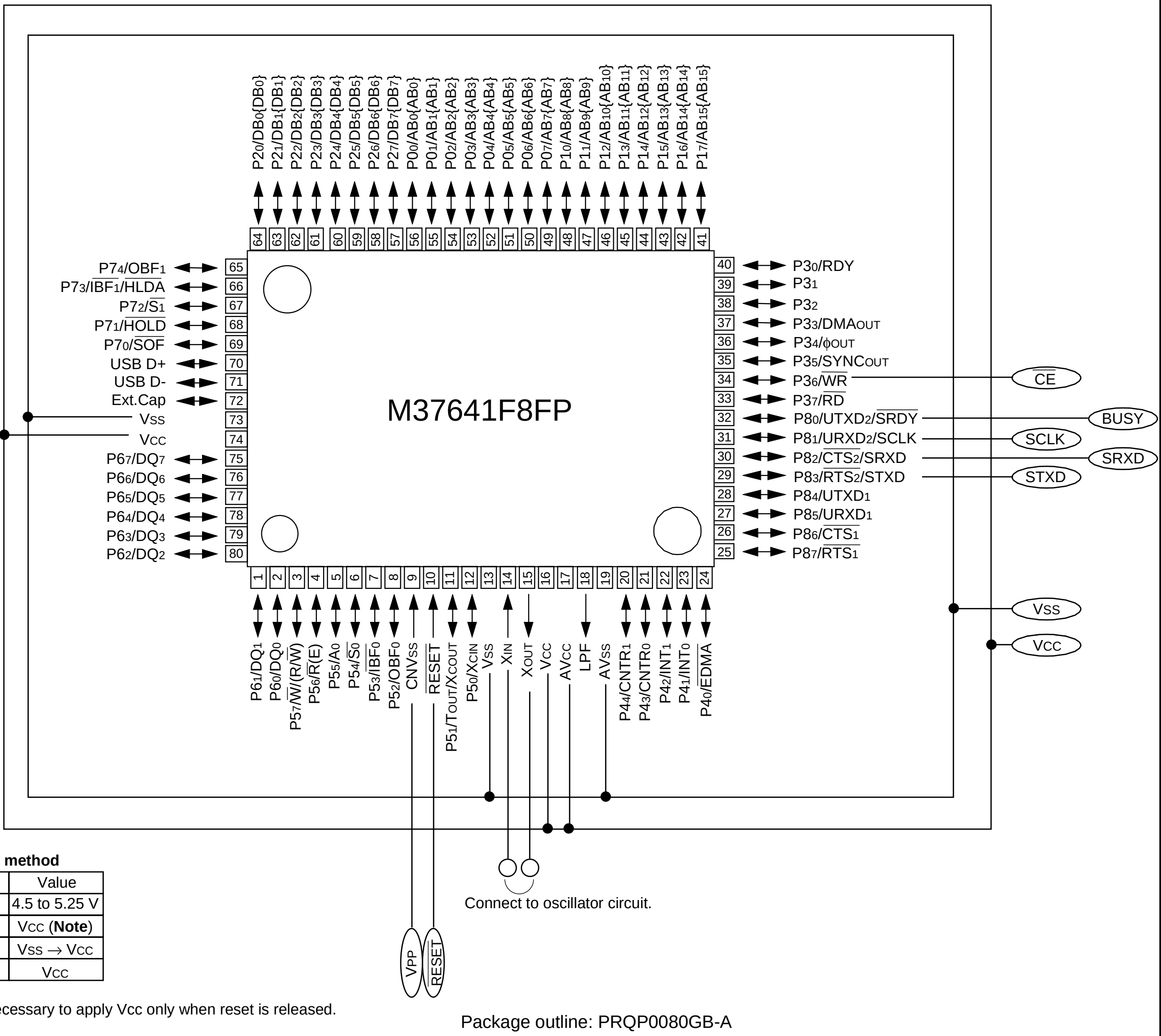
The STXD pin is for CMOS output. Transfer is in 8-bit units with LSB first.

When busy, such as during transmission, reception, erasing or program execution, the $\overline{SRDY}$ (BUSY) pin is “H” level. Accordingly, always start the next transfer after the $\overline{SRDY}$ (BUSY) pin is “L” level.

Also, data and status registers in a memory can be read after inputting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following explains software commands, status registers, etc.

Table 13 Description of pin function (Standard Serial I/O Mode)

Pin name	Signal name	I/O	Function
VCC,VSS	Power supply input		Apply 4.50 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the VCC pin. Apply 0 V to the Vss pin.
CNVSS	CNVSS	I	This controls the MCU operating mode. Connect this pin to VPP (= 4.50 V – 5.25 V
$\overline{\text{RESET}}$	Reset input	I	To reset, input “L” level for 20 cycles or longer clocks of ϕ.
XIN	Clock input		Connect a ceramic or crystal resonator between the XIN and XOUT pins. When inputting an externally derived clock, input it from XIN and leave XOUT open.
XOUT	Clock output		
AVCC, AVSS	Analog power supply input		Apply 4.50 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the AVCC pin. Apply 0 V to the AVss pin.
LPF	LPF	O	Loop filter for the frequency synthesizer. When this pin is not used, leave this open.
Ext.Cap	3.3 V line power supply input	I	Power supply input pin for 3.3 V USB line driver. When this pin is not used, input “H” level.
USB D+	USB D+	I/O	USB D+ signal port. When this pin is not used, input “H” level.
USB D-	USB D-	I/O	USB D- signal port. When this pin is not used, input “L” level.
P00 to P07	I/O port P0	I/O	When these ports are not used, input “L” or “H” level, or leave them open in output mode.
P10 to P17	I/O port P1	I/O	
P20 to P27	I/O port P2	I/O	
P30 to P35, P37	I/O port P3	I/O	
P36	$\overline{\text{CE}}$ input	I	Input “H” level.
P40 to P44	I/O port P4	I/O	When these ports are not used, input “L” or “H” level, or leave them open in output mode.
P50 to P57	I/O port P5	I/O	
P60 to P67	I/O port P6	I/O	
P70 to P74	I/O port P7	I/O	
P80	BUSY output	O	This is a BUSY output pin.
P81	SCLK input	I	This is a serial clock input pin.
P82	SRXD input	I	This is a serial data input pin.
P83	STXDoutput	O	This is a serial data output pin.
P84 to P87	I/O port P8	I/O	When these ports are not used, input “L” or “H” level, or leave them open in output mode.

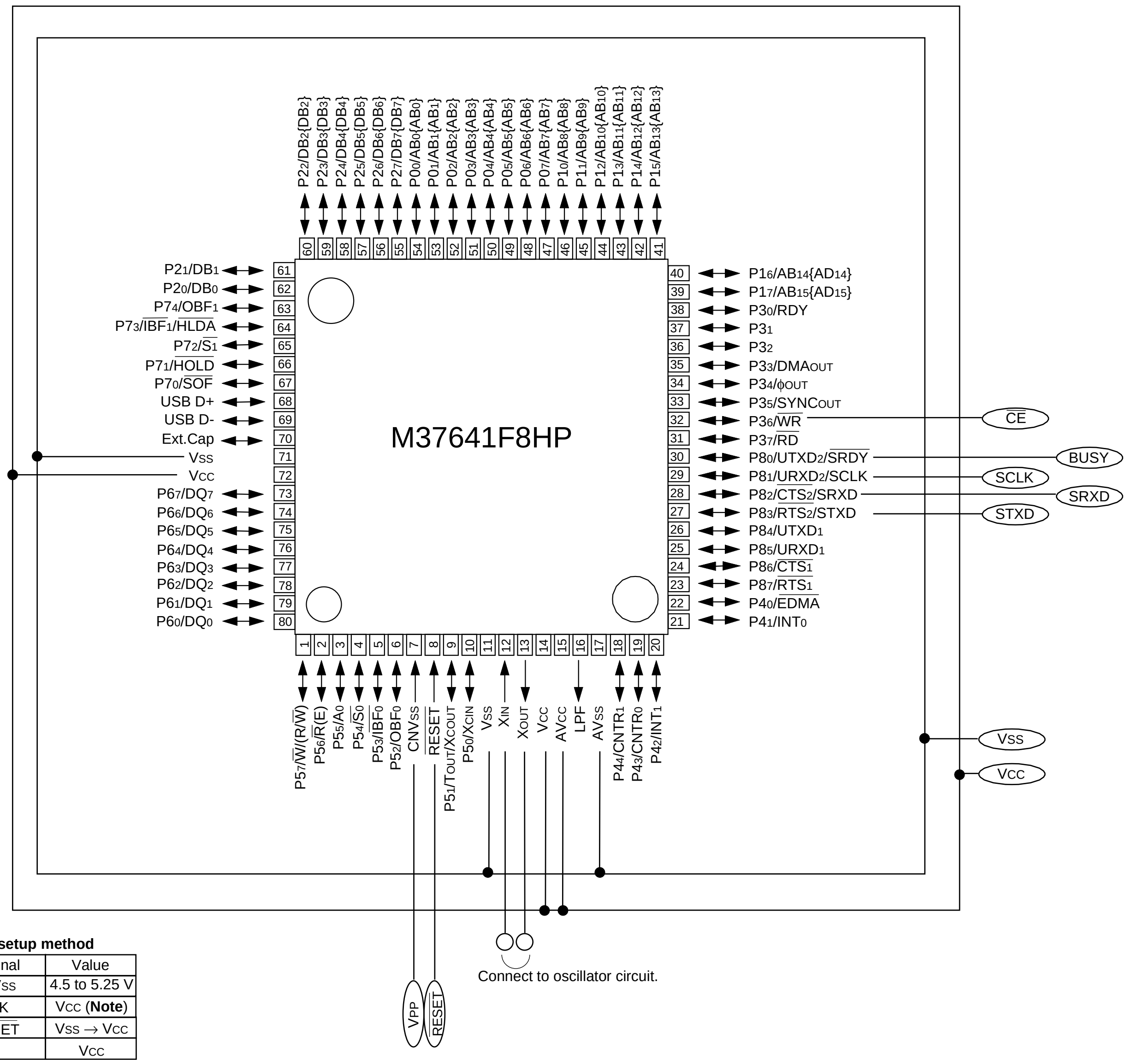


Mode setup method

Signal	Value
CNVss	4.5 to 5.25 V
SCLK	Vcc (**Note**)
RESET	Vss → Vcc
CE	Vcc

Note: It is necessary to apply Vcc only when reset is released.

Fig. 87 Pin connection diagram in standard serial I/O mode (1)



Mode setup method

Signal	Value
CNVss	4.5 to 5.25 V
SCLK	Vcc (**Note**)
RESET	Vss → Vcc
CE	Vcc

Note: It is necessary to apply Vcc only when reset is released.

Fig. 88 Pin connection diagram in standard serial I/O mode (2)

Software Commands (Standard Serial I/O Mode)

Table 14 lists software commands. In standard serial I/O mode, erase, program and read are controlled by transferring software commands via the SRXD pin. Software commands are explained here below.

Table 14 Software commands (Standard serial I/O mode)

	Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte	.....	When ID is not verified
1	Page read	FF_{16}	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41_{16}	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Block erase	20_{16}	Address (middle)	Address (high)	$D0_{16}$				Not acceptable
4	Erase all blocks	$A7_{16}$	$D0_{16}$						Not acceptable
5	Read status register	70_{16}	SRD output	SRD1 output					Acceptable
6	Clear status register	50_{16}							Not acceptable
7	ID code check	$F5_{16}$	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
8	Download function	FA_{16}	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
9	Version data output function	FB_{16}	Version data output	Version data output	Version data output	Version data output	Version data output	Version data output to 9th byte	Acceptable
10	Boot ROM area output function	FC_{16}	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable

Notes1: Shading indicates transfer from the internal flash memory microcomputer to a programmer. All other data is transferred from an external equipment (programmer) to the internal flash memory microcomputer.
2: SRD refers to status register data. SRD1 refers to status register 1 data.
3: All commands can be accepted for the products of which boot ROM area is totally blank.
4: Address low is AB_0 to AB_7; Address middle is AB_8 to AB_{15}; Address high is AB_{16} to AB_{23}.

●**Page Read Command**
This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

(1) Transfer the "FF_{16}" command code with the 1st byte.
(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.
(3) From the 4th byte onward, data (DB_0 to DB_7) for the page (256 bytes) specified with addresses AB_8 to AB_{23} will be output sequentially from the smallest address first synchronized with the fall of the clock.

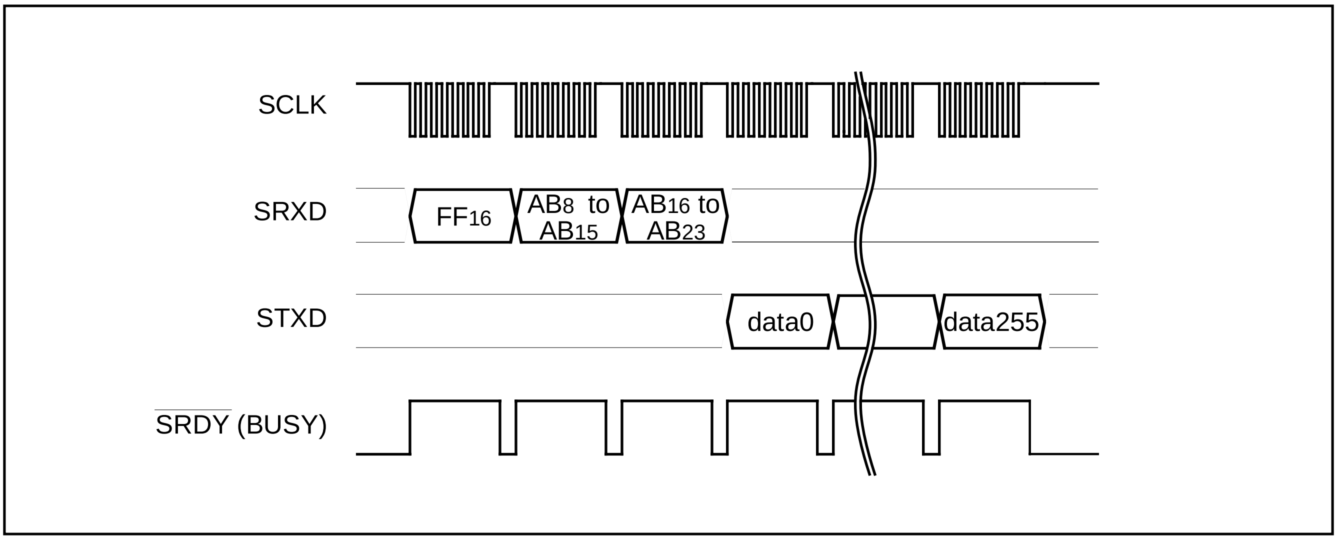


Fig. 89 Timing for page read

●**Read Status Register Command**
This command reads status information. When the "70_{16}" command code is transferred with the 1st byte, the contents of the status register (SRD) with the 2nd byte and the contents of status register 1 (SRD1) with the 3rd byte are read.

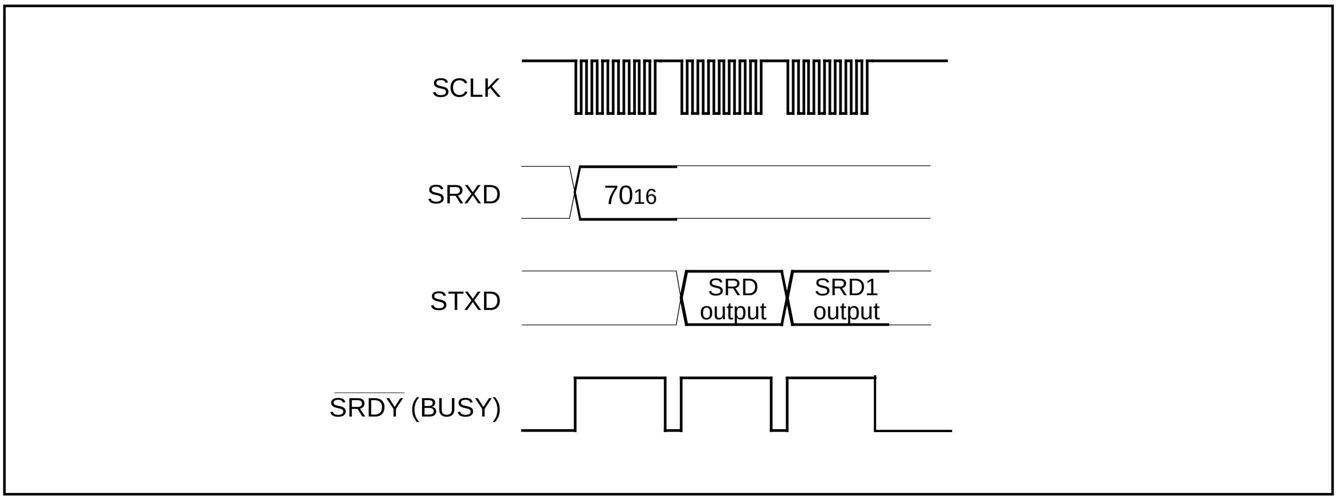


Fig. 90 Timing for reading status register

●Clear Status Register Command

This command clears the bits (SR3 to SR5) which are set when the status register operation ends in error. When the "50_{16}" command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the $\overline{\text{SRDY}}$ (BUSY) signal changes from "H" to "L" level.

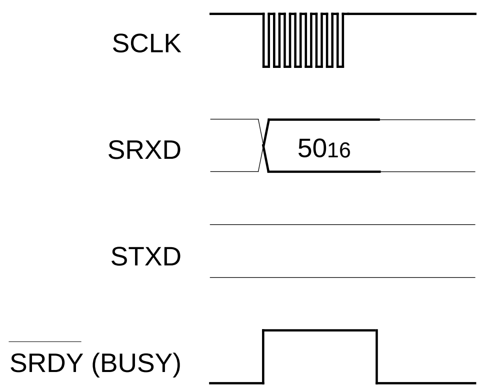


Fig. 91 Timing for clear status register

●Page Program Command

This command writes the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page program command as explained here following.

(1) Transfer the "41_{16}" command code with the 1st byte.

(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data (DB_0 to DB_7) for the page (256 bytes) specified with addresses A_8 to A_{23} is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the $\overline{\text{SRDY}}$ (BUSY) signal changes from "H" to "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

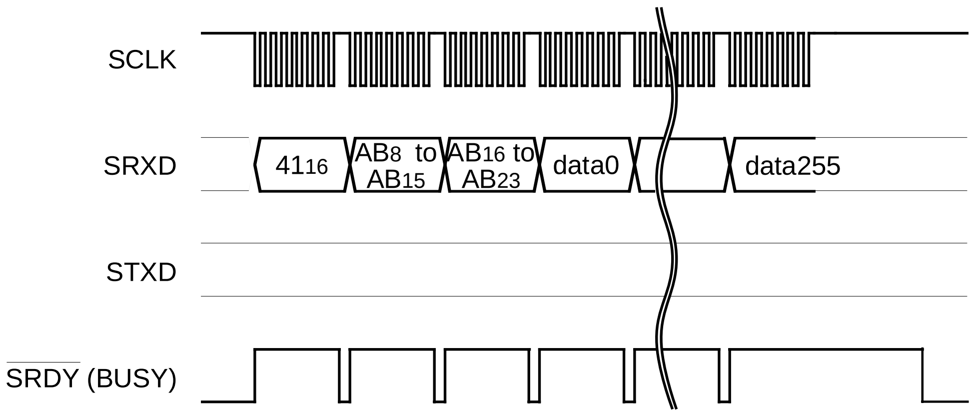


Fig. 92 Timing for page program

●**Block Erase Command**

This command erases the contents of the specified block. Execute the block erase command as explained here following.

(1) Transfer the "20_{16}" command code with the 1st byte.

(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.

(3) Transfer the verify command code "$D0_{16}$" with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Set the addresses AB_8 to AB_{23} to the maximum address of the specified block.

When block erasing ends, the $\overline{SRDY}$ (BUSY) signal changes from "H" to "L" level. The result of the erase operation can be known by reading the status register.

For more information, see the section on the status register.

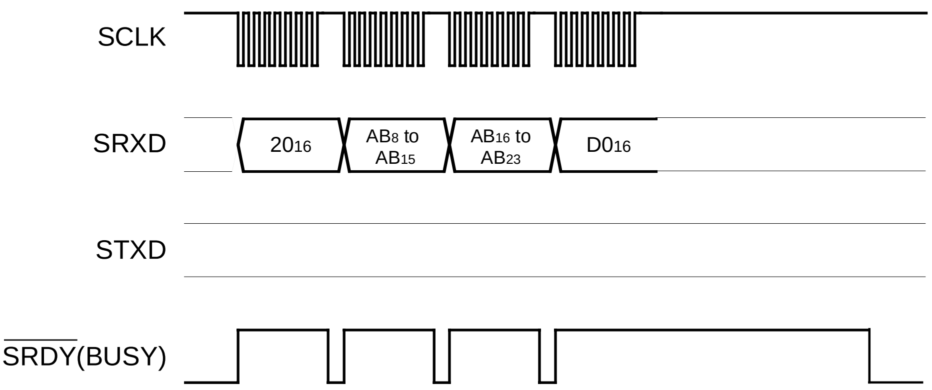


Fig. 93 Timing for block erasing

●**Erase All Blocks Command**

This command erases the contents of all blocks. Execute the erase all blocks command as explained here following.

(1) Transfer the "$A7_{16}$" command code with the 1st byte.

(2) Transfer the verify command code "$D0_{16}$" with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When erase all blocks end, the $\overline{SRDY}$ (BUSY) signal changes from "H" to "L" level. The result of the erase operation can be known by reading the status register.

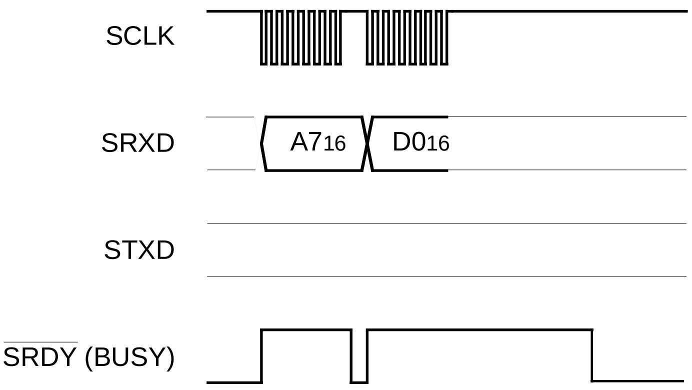


Fig. 94 Timing for erase all blocks

●**Download Command**

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

(1) Transfer the "FA16" command code with the 1st byte.

(2) Transfer the program size with the 2nd and 3rd bytes.

(3) Transfer the check sum with the 4th byte. The check sum is added to all data sent with the 5th byte onward.

(4) The program to execute is sent with the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.

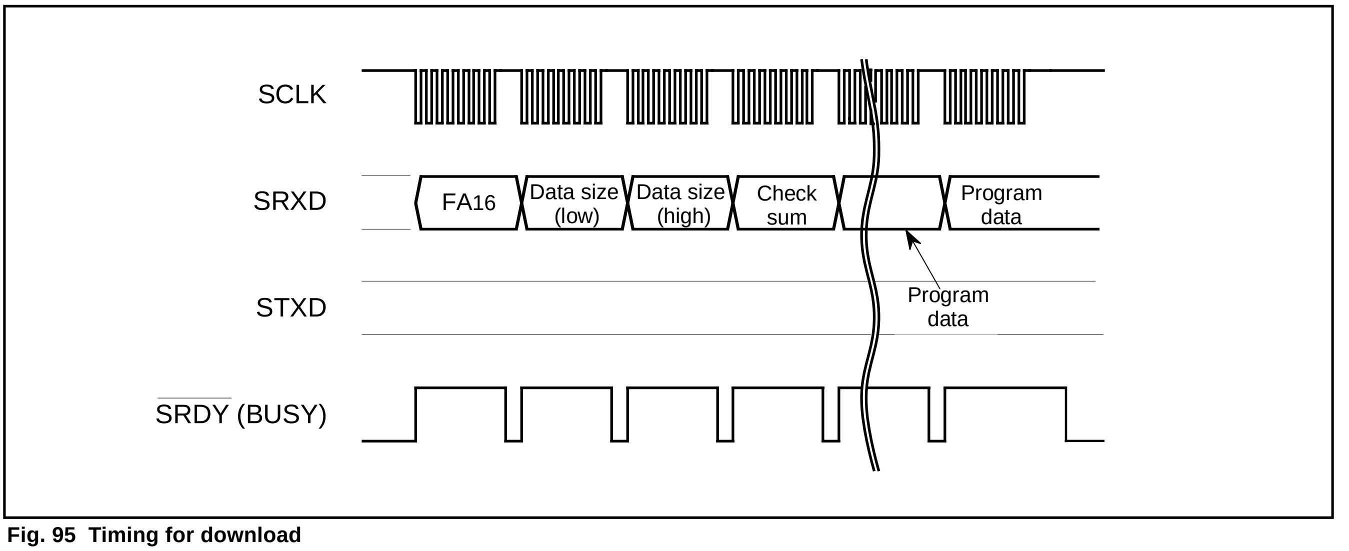


Fig. 95 Timing for download

●**Version Information Output Command**

This command outputs the version information of the control program stored in the Boot ROM area. Execute the version information output command as explained here following.

(1) Transfer the "FB_{16}" command code with the 1st byte.
(2) The version information will be output from the 2nd byte onward.

This data is composed of 8 ASCII code characters.

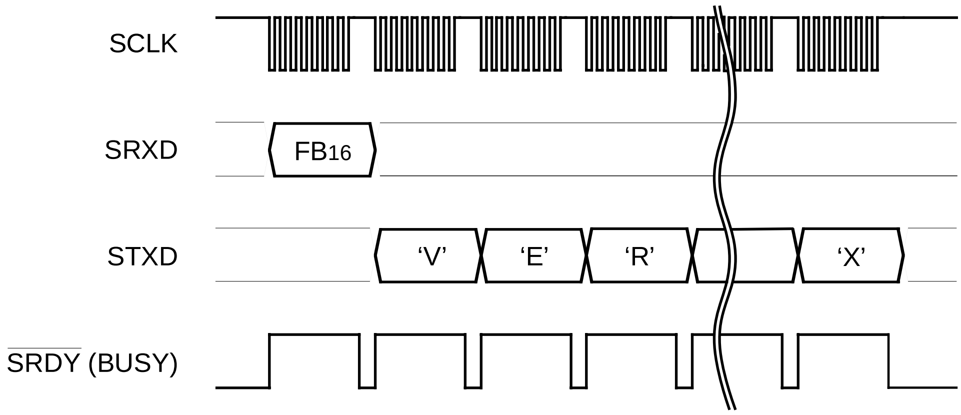


Fig. 96 Timing for version information output

●**Boot ROM Area Output Command**

This command reads the control program stored in the Boot ROM area in page (256 bytes) unit. Execute the Boot ROM area output command as explained here following.

(1) Transfer the "FC_{16}" command code with the 1st byte.
(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.
(3) From the 4th byte onward, data (DB_0 to DB_7) for the page (256 bytes) specified with addresses AB_8 to AB_{23} will be output sequentially from the smallest address first synchronized with the fall of the clock.

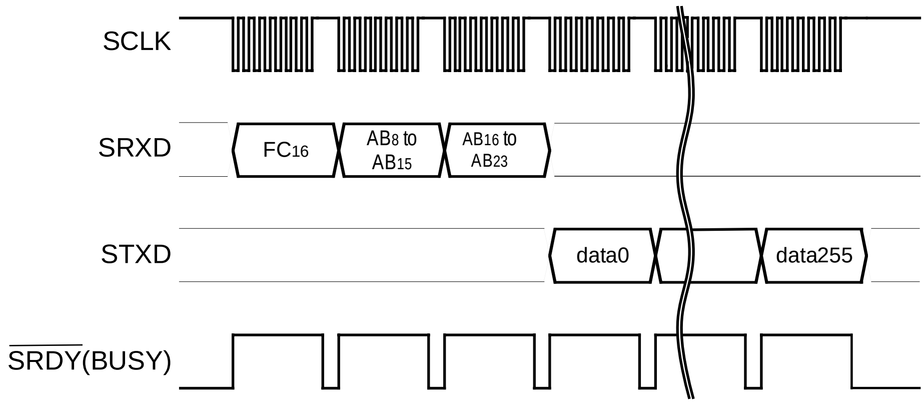


Fig. 97 Timing for Boot ROM area output

●ID Code Check

This command checks the ID code. Execute the boot ID check command as explained here following.

(1) Transfer the "F5$_{16}$" command code with the 1st byte.
(2) Transfer addresses AB_0 to AB_7, AB_8 to AB_{15} and AB_{16} to AB_{23} ("00$_{16}$") of the 1st byte of the ID code with the 2nd and 3rd respectively.
(3) Transfer the number of data sets of the ID code with the 5th byte.
(4) Transfer the ID code with the 6th byte onward, starting with the 1st byte of the code.

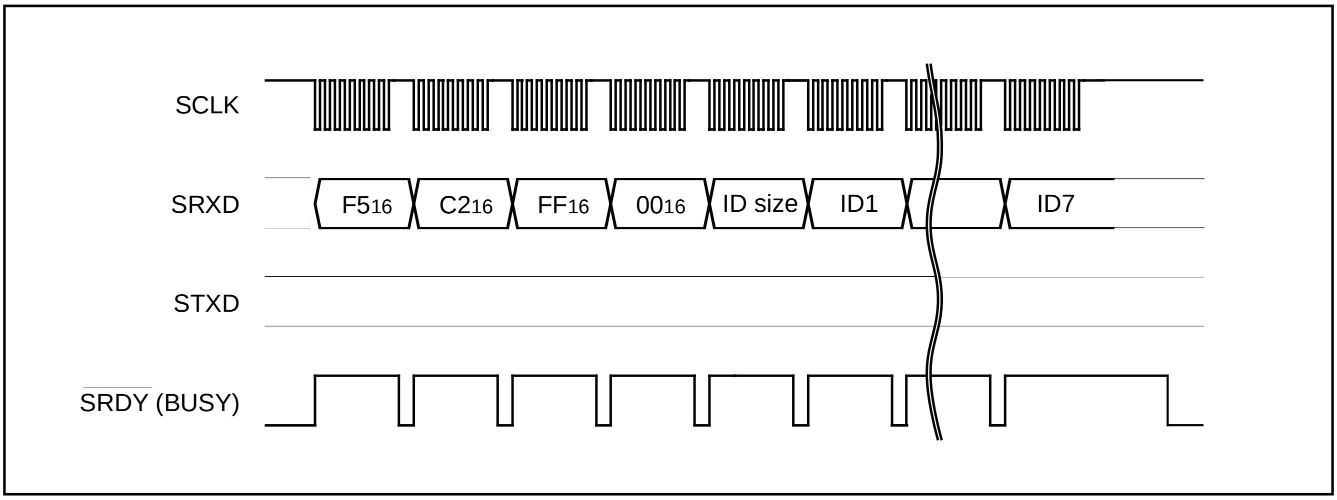


Fig. 98 Timing for ID check

●ID Code

When the flash memory is not blank, the ID code sent from the serial programmer and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the serial programmer is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses FFC2$_{16}$ to FFC8$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

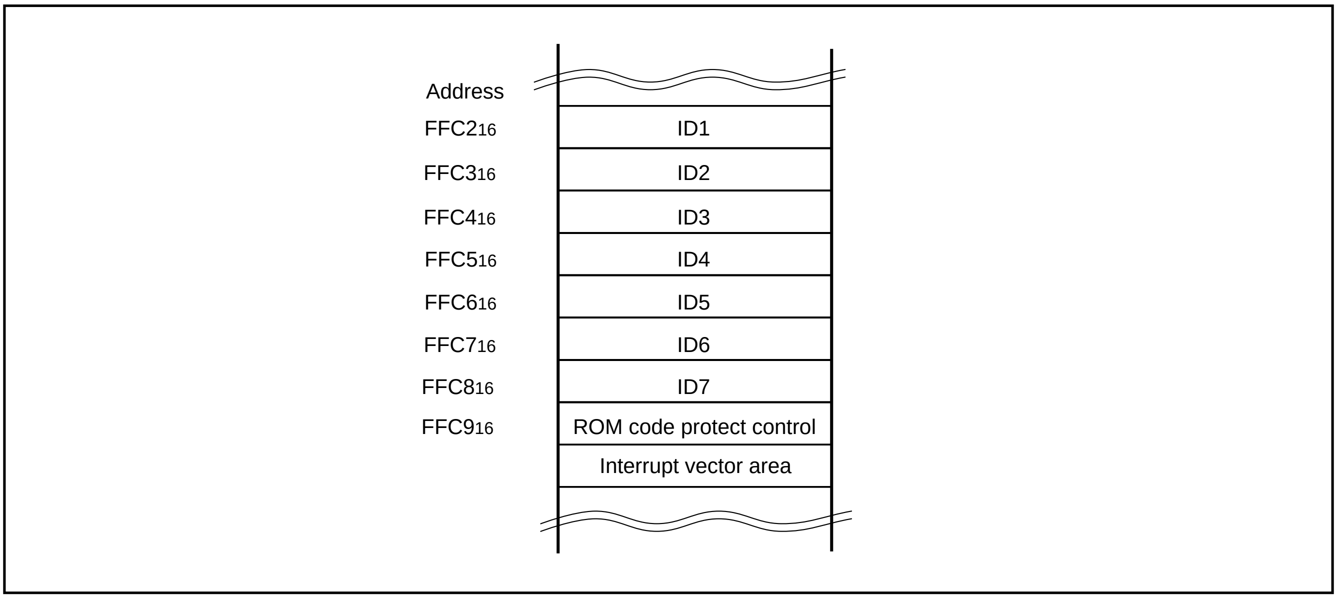


Fig. 99 ID code storage addresses

●Status Register (SRD)
The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (70_{16}). Also, the status register is cleared by writing the clear status register command (50_{16}).
Table 15 lists the definition of each status register bit. After releasing the reset, the status register becomes "80_{16}".

•Sequencer status (SR7)
The sequencer status indicates the operating status of the the flash memory.
After power-on and recover from deep power down mode, the sequencer status is set to "1" (ready).
This status bit is set to "0" (busy) during write or erase operation and is set to "1" upon completion of these operations.

•Erase status (SR5)
The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

•Program status (SR4)
The program status indicates the operating status of write operation. If a program error occurs, it is set to "1". When the program status is cleared, it is set to "0".

Table 15 Definition of each bit of status register (SRD)

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

●Status Register 1 (SRD1)
The status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the status register (SRD) by writing the read status register command (70_{16}). Also, status register 1 is cleared by writing the clear status register command (50_{16}).
Table 16 lists the definition of each status register 1 bit. This register becomes "00_{16}" when power is turned on and the flag status is maintained even after the reset.

•Boot update completed bit (SR15)
This flag indicates whether the control program was downloaded to the RAM or not, using the download function.

•Check sum consistency bit (SR12)
This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

•ID code check completed bits (SR11 and SR10)
These flags indicate the result of ID code checks. Some commands cannot be accepted without an ID code check.

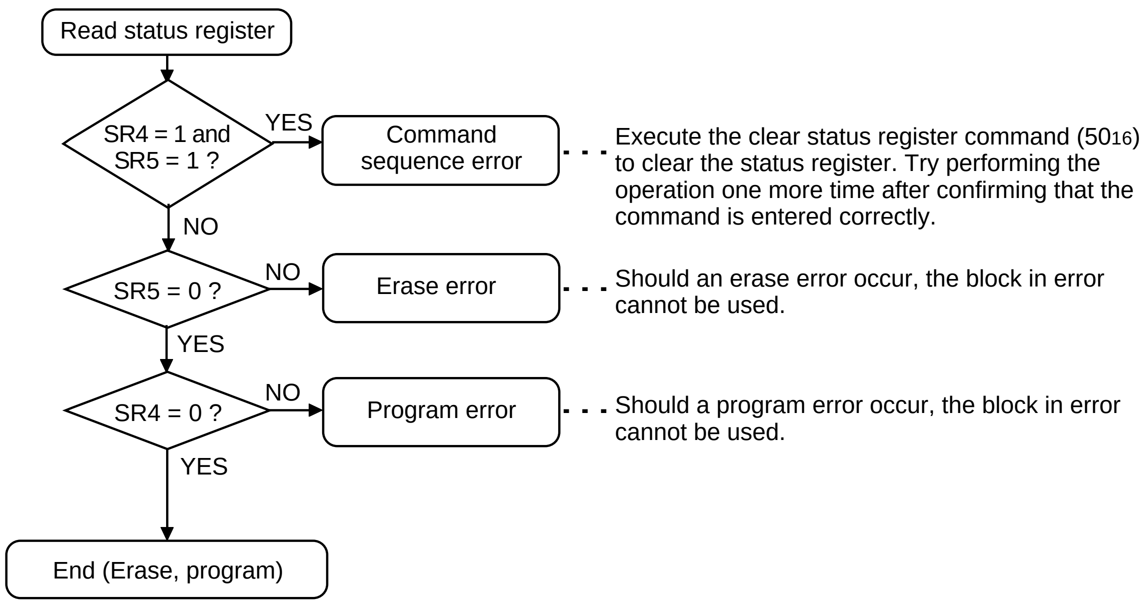
•Data reception time out (SR9)
This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the MCU returns to the command wait state.

Table 16 Definition of each bit of status register 1 (SRD1)

SRD1 bits	Status name	Definition	
		"1"	"0"
SR15 (bit7)	Boot update completed bit	Update completed	Not Update
SR14 (bit6)	Reserved	-	-
SR13 (bit5)	Reserved	-	-
SR12 (bit4)	Checksum match bit	Match	Mismatch
SR11 (bit3) SR10 (bit2)	ID code check completed bits	00 01 10 11	Not verified Verification mismatch Reserved Verified
SR9 (bit1)	Data reception time out	Time out	Normal operation
SR8 (bit0)	Reserved	-	-

Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 100 shows a flow-chart of the full status check and explains how to remedy errors which occur.

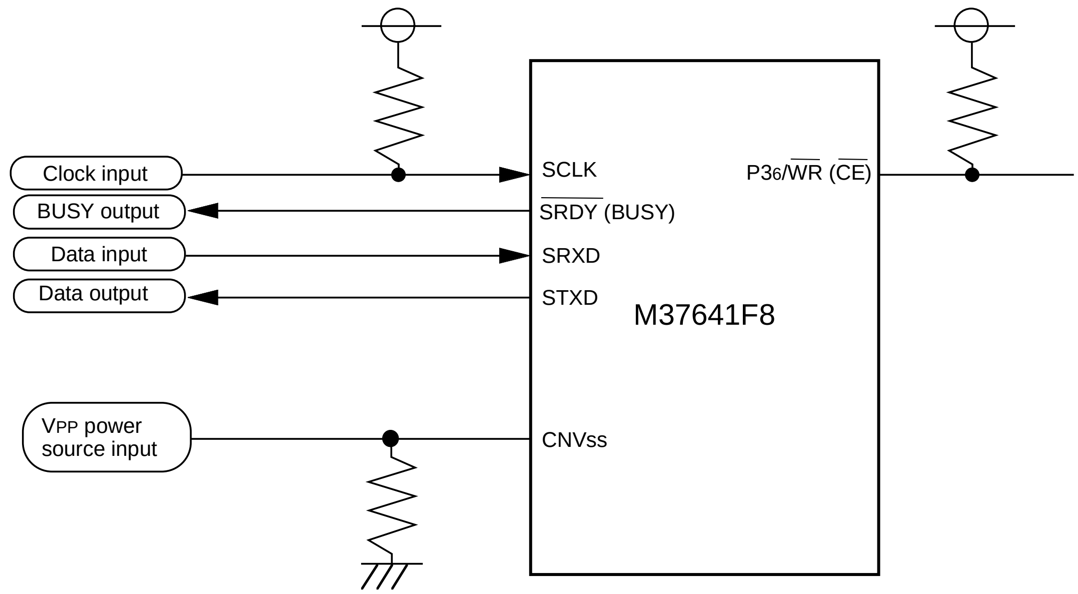


Note: When one of SR5 to SR4 is set to “1” , none of the page read, program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Fig. 100 Full status check flowchart and remedial procedure for errors

Example Circuit Application for Standard Serial I/O Mode

Figure 101 shows a circuit application for the standard serial I/O mode. Control pins will vary according to a programmer, therefore see a programmer manual for more information.



Notes 1: Control pins and external circuitry will vary according to a programmer. For more information, see the programmer manual.
2: In this example, the Vpp power supply is supplied from an external source (programmer). To use the user’s power source, connect to 4.5 V to 5.25 V.

Fig. 101 Example circuit application for standard serial I/O mode

NOTES ON PROGRAMMING
Processor Status Register

•The contents of the processor status register (PS) after a reset are undefined, except for the interrupt disable flag (I) which is “1”. After a reset, initialize flags which affect program execution. In particular, it is essential to initialize the index X mode (T) and the decimal mode (D) flags because of their effect on calculations.

•To reference the contents of the processor status register (PS), execute the **PHP** instruction once then read the contents of (S+1). If necessary, execute the **PLP** instruction to return the PS to its original status.
A **NOP** instruction must be executed after every **PLP** instruction.

•A **SEI** instruction must be executed before every **PLP** instruction.
A **NOP** instruction must be executed before every **CLI** instruction.

BRK Instruction

It can be detected that the **BRK** instruction interrupt event or the least priority interrupt event by referring the stored B flag state. Refer to the stored B flag state in the interrupt routine.

Decimal Calculations

When decimal mode is selected, the values of the V flags are invalid.
The carry flag (C) is set to “1” if a carry is generated as a result of the calculation, or is cleared to “0” if a borrow is generated. To determine whether a calculation has generated a carry, the C flag must be initialized to “0” before each calculation. To check for a borrow, the C flag must be initialized to “1” before each calculation.

Multiplication and Division Instructions

•The index X mode (T) and the decimal mode (D) flags do not affect the MUL and DIV instruction.

Instruction Execution Time

The instruction execution time is obtained by multiplying the frequency of the internal clock ϕ by the number of cycles needed to execute an instruction.
The number of cycles required to execute an instruction is shown in the list of machine instructions.

Timers

•If a value n (between 0 and 255) is written to a timer latch, the frequency division ratio is 1/(n+1).

•$P5_1$/X_{COUT}/T_{OUT} pin cannot function as an I/O port when X_{CIN} - X_{COUT} is oscillating. When X_{CIN} - X_{COUT} oscillation is not used or X_{COUT} oscillation drive is disabled, this pin can function as the T_{OUT} output pin of the timer 1 or 2.
When using the T_{OUT} output function and f(X_{CIN}) divided by 2 is used as the timer 1 count source (bit 2 of T123M = “1”), disable X_{COUT} oscillation drive (bit 5 of CCR = “1”).

Ports

•When the data register (port latch) of an I/O port is modified with the bit managing instruction (**SEB**, **CLB** instructions) the value of the unspecified bit may be changed.

•In standby state (the stop mode by executing the **STP** instruction, and the wait mode by executing the **WIT** instruction) for low-power dissipation, do not make input levels of an I/O port “undefined”, especially for I/O ports of the P-channel and the N-channel open-drain.
Pull-up (connect the port to Vcc) or pull-down (connect the port to Vss) these ports through a resistor.
When determining a resistance value, note the following points:
(1) External circuit
(2) Variation of output levels during the ordinary operation

When using built-in pull-up or pull-down resistor, note on varied current values.
(1) When setting as an input port : Fix its input level
(2) When setting as an output port : Prevent current from flowing out to external

Serial I/O

Do not write to the serial I/O shift register during a transfer when in SPI compatible mode.

UART

•The all error flags PER, FER, OER and SER are cleared to “0” when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit. These flags are also cleared to “0” by execution of bit test instructions such as **BBC** and **BCS**.

•The transmission interrupt request bit is set and the interrupt request is generated by setting the transmit enable bit to “1” even when selecting timing that either of the following flags is set to “1” as timing where the transmission interrupt is generated:
(1) Transmit buffer empty flag is set to “1”
(2) Transmit complete flag is set to “1”.

Therefore, when the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as the following sequence:
(1) Transmit enable bit is set to “1”
(2) Transmit interrupt request bit is set to “0”
(3) Transmit interrupt enable bit is set to “1”.

•Do not update a value of UARTx baud rate generator in the condition of transmission enabled or reception enabled. Disable transmission and reception before updating the value. If the former data remains in the UARTx transmit buffer registers 1 and 2 when transmission is enabled, an undefined data might be output.

•The receive buffer full interrupt request is not generated if receive errors are detected at receiving.

•If a character bit length is 7 bits, bit 7 of the UARTx transmit/receive buffer register 1 and bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
If a character bit length is 8 bits, bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
If a character bit length is 9 bits, bits 1 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are "0" at receiving.

USB

•When the USB Reset Interrupt Status Flag is kept at "1", all other flags in the USB internal registers (addresses 0050_{16} to $005F_{16}$) will return to their reset status. However, the following registers are not affected by the USB reset: USB control register (address 0013_{16}), Frequency synthesizer control register (address $006C_{16}$), Clock control register (address $001F_{16}$), and USB endpoint-x FIFO register (addresses 0060_{16} to 0064_{16}).

•When not using the USB function, set the USB Line Driver Supply Enable Bit of the USB control register (address 0013_{16}) to "1" for power supply to the internal circuits (at Vcc = 5V).

•When using an isochronous transfer, set the FLUSH Bit (bit 6 of address 0059_{16} and bit 6 of address $005A_{16}$) as follows:
IN FIFO: use AUTO_FLUSH Bit (bit 6 of address 0058_{16})
OUT FIFO: when OUT_PKT_RDY Bit is "1", set FLUSH Bit to "1"

•When the USB SOF Port Select Bit is "1", the reference pulse of 83.3 ns (ϕ = 12 MHz) is output from the $P7_0/\overline{SOF}$ pin and synchronized with the SOF packet.

•The IN_PKT_RDY Bit can be set by software even when using the AUTO_SET function.

•When writing to USB-related registers, set the USB Clock Enable Bit to "1", then perform the write after four ϕ cycle waits.

•When using the MCU at Vcc = 3.3V, set the USB Line Driver Supply Enable Bit to "0" (line driver disable). Note that setting the USB Line Driver Current Control Bit (USBC3) doesn't affect the USB operation.

•Read one packet data from the OUT FIFO before clearing the OUT_PKT_RDY Flag. If the OUT_PKT_RDY Flag is cleared while one packet data is being read, the internal read pointer cannot operate normally.

•Use the AUTO_FLUSH Bit (bit 6 of address 0058_{16}) in double buffer mode.

•Use the transfer instructions such as **LDA** and **STA** to set the registers: USB interrupt status registers 1, 2 (addresses 0052_{16}, 0053_{16}); USB endpoint 0 IN control register (address 0059_{16}); USB endpoint x IN control register (address 0059_{16}); USB endpoint x OUT control register (address $005A_{16}$). Do not use the read-modify-write instructions such as the **SEB** or the **CLB** instruction.
When writing to bits shown by Table 32 using the transfer instruction such as **LDA** or **STA**, a value which never affect its bit state is required. Take the following sequence to change these bits contents:
(1) Store the register contents onto a variable or a data register.
(2) Change the target bit on the variable or the data register. Simultaneously mask the bit so that its bit state cannot be changed. (See to Table 39.)
(3) Write the value from the variable or the data register to the register using the transfer instruction such as **LDA** or **STA.**

•To use the AUTO_SET function for an IN transfer when the AUTO_SET bit is set to 1, set the FIFO to single buffer mode.

Table 17 Bits of which state might be changed owing to software write

Register name	Bit name	Value not affecting state (**Note**)
USB endpoint 0 IN control register	IN_PKT_RDY (b1)	"0"
	DATA_END (b3)	"0"
	FORCE_STALL (b4)	"1"
USB endpoint x (x = 1 to 4) IN control register	IN_PKT_RDY (b0)	"0"
	UNDER_RUN (b1)	"1"
USB endpoint x (x = 1 to 4) OUT control register	OUT_PKT_RDY (b0)	"1"
	OVER_RUN (b1)	"1"
	FORCE_STALL (b4)	"1"
	DATA_ERR (b5)	"1"

Note: Writing this value will not change the bit state, because this value cannot be written to the bit by software.

Frequency Synthesizer

•The frequency synthesizer and DC-DC converter must be set up as follows when recovering from a Hardware Reset:

(1) Enable the frequency synthesizer after setting the frequency synthesizer related registers (addresses $006C_{16}$ to $006F_{16}$). Then wait for 2 ms.

(2) Check the Frequency Synthesizer Lock Status Bit. If “0”, wait for 0.1 ms and then recheck.

(3) When using the USB built-in DC-DC converter, set the USB Line Driver Supply Enable Bit of the USB control register to “1”. This setting must be done 2 ms or more after the setup described in step (1). The USB Line Driver Current Control Bit must be set to “0” at this time. (When Vcc = 3.3V, the setting explained in this step is not necessary.)

(4) After waiting for (C + 1) ms so that the external capacitance pin (Ext. Cap. pin) can reach approximately 3.3 V, set the USB Clock Enable Bit to “1”. At this time, “C” equals the capacitance (µ F) of the capacitor connected to the Ext. Cap. pin. For example, if 2.2 µF and 0.1 µF capacitors are connected to the Ext. Cap. in parallel, the required wait will be (2.3 + 1) ms.

(5) After enabling the USB clock, wait for 4 or more φ cycles, and then set the USB Enable Bit to “1”.

•Bits 6 and 5 of the frequency synthesizer control register (address $006C_{16}$) are initialized to “11” after reset release. Make sure to set bits 6 and 5 to “10” after the Frequency Synthesizer Lock Status Bit goes to “1”.

•When using the frequency synthesized clock function, we recommend using the fastest frequency possible of $f(X_{IN})$ or $f(X_{CIN})$ as an input clock for the PLL. Owing to the PLL mechanism, the PLL controls the speed of multiplied clocks from the source clock. As a result, when the source clock input is lower, the generated clock becomes less stable. This is because more multipliers are needed and the speed control is very rough. Higher source clock input generates a stabler clock, as less multipliers are needed and the speed control is more accurate. However, if the input clock frequency is relatively high, the PLL clock generator can quickly lock-up the output clock to the source and make the output clock very stable.

•Set the value of frequency synthesizer multiply register 2 (FSM2) so that the fPIN is 1 MHZ or higher.

DMA

•In the memory expansion mode and microprocessor mode, the DMA_{OUT} pin outputs “H” during a DMA transfer.

•Do not access the DMAC-related registers by using a DMAC transfer. The destination address data and the source address data will collide in the DMAC internal bus.

•When using the USB FIFO as the DMA transfer source, make sure that, if you use the AUTO_SET function, short packet data does not get mixed in with the transfer data.

•When setting the DMAC channel x enable bit (bit 7 of address 0041_{16}) to “1”, be sure simultaneously to set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of address 0041_{16}) to “1”. If this is not performed, an incorrect data will be transferred at the same time when the DMAC is enabled.

Memory Expansion Mode & Microprocessor Mode

•In both memory expansion mode and microprocessor mode, use the LDM instruction or STA instruction to write to port P3 (address $000E_{16}$). When using the Read-Modify-Write instruction (**SEB** instruction, **CLB** instruction) you will need to map a memory that the CPU can read from and write to.

•In the memory expansion mode, if the internal and external memory areas overlap, the internal memory becomes the valid memory for the overlapping area. When the CPU performs a read or a write operation on this overlapped area, the following things happen:

(1) Read

The CPU reads out the data in the internal memory instead of in the external memory. Note that, since the CPU will output a proper read signal, address signal, etc., the memory data at the respective address will appear on the external data bus.

(2) Write

The CPU writes data to both the internal and external memories.

•The wait function is serviceable at accessing an external memory.

Stop Mode

•When the STP instruction is executed, bit 7 of the clock control register (address $001F_{16}$) goes to “0”. To return from stop mode, reset CCR7 to “1”.

•When using f_{SYN} (set Internal System Clock Select Bit (CPMA6) to “1”) as the internal system clock, switch CPMA6 to “0” before executing the **STP** instruction. Reset CPMA6 after the system returns from Stop Mode and the frequency synthesizer has stabilized.

CPMA6 does not need to be switched to “0” when using the **WIT** instruction.

•When the **STP** instruction is being executed, all bits except bit 4 of the timer 123 mode register (address 0029_{16}) are initialized to “0”. It is not necessary to set T123M1 (Timer 1 Count Stop Bit) to “0” before executing the **STP** instruction. After returning from Stop Mode, reset the timer 1 (address 0024_{16}), timer 2 (address 0025_{16}), and the timer 123 mode register (address 0029_{16}).

USAGE NOTES

Oscillator Connection Notice

The built-in feedback register (1 MΩ) and the dumping resistor (400 Ω) is internally connected between pins XIN and XOUT.

Power Source Voltage

When the power source voltage value of a microcomputer is less than the value which is indicated as the recommended operating conditions, the microcomputer does not operate normally and may perform unstable operation.

In a system where the power source voltage drops slowly when the power source voltage drops or the power supply is turned off, reset a microcomputer when the power source voltage is less than the recommended operating conditions and design a system not to cause errors to the system by this unstable operation.

Power Supply Pins Treatment Notice

Please connect 0.1 µF and 4.7 µF capacitors in parallel between pins Vcc and Vss, and pins AVss and AVcc.

These capacitors must be connected as close as possible between the DC supply and GND pins, and also the analog supply pin and corresponding GND pin.

Wiring patterns for these supply and GND pins must be wider than other signal patterns.

These filter capacitors should not be placed near the LPF pins as they will cause noise problems

Reset Pin Treatment Notice (Noise Elimination)

If the reset input signal rises very slowly, we recommend attaching a capacitor, such as a 1000 pF ceramic capacitor with excellent high frequency characteristics, between the RESET pin and the Vss pin.

Please note the following two issues for this capacitor connection.

(1) Capacitor wiring pattern must be as short as possible (within 20 mm).

(2) The user must perform an application level operation test.

LPF Pin Treatment Notice

All passive components must be located as close as possible to the LPF pin.

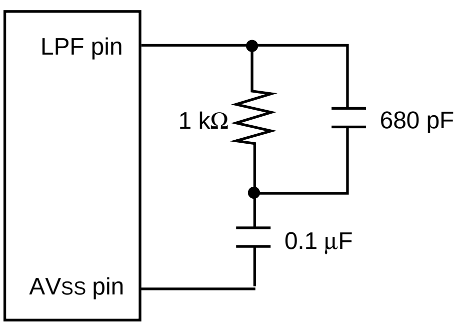


Fig. 102 Passive components near LPF pin

AVss and AVcc Pin Treatment Notice (Noise Elimination)

An insulation connector (Ferrite Beads) must be connected between AVss and Vss pins and between AVcc and Vcc pins.

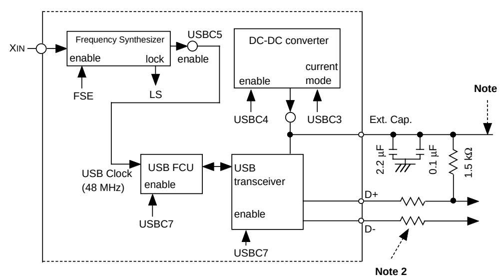
USB Transceiver Treatment (Noise Elimination)

•The Full-Speed USB2.0 specification requires a driver -impedance 28 to 44 Ω. (Refer to Clause 7.1.1.1 Full-speed (12 Mb/s) Driver Characteristics in the USB specification.) In order to meet the USB specification impedance requirements, connect a resistor (27 Ω to 33 Ω recommended) in series to the USB D+ pin and the USB D- pin.

In addition, in order to reduce the ringing and control the falling/rising timing of USB D+/D- and a crossover point, connect a capacitor between the USB D+/D- pins and the Vss pin if necessary.

The values and structure of those peripheral elements depend on the impedance characteristics and the layout of the printed circuit board. Accordingly, evaluate your system and observe waveforms before actual use and decide use of elements and the values of resistors and capacitors.

•Connect a capacitor between the Ext. Cap. pin and the Vss pin. The capacitor should have a 2.2 µF capacitor (Tantalum capacitor) and a 0.1 µF capacitor (ceramic capacitor) connected in parallel. Figure 103 for the proper positions of the peripheral components.



Notes 1: In Vcc = 3.3 V, connect to Vcc. In Vcc =5 V, do not connect the external DC-DC converter to the Ext. Cap pin.
2: The resistors values depend on the layout of the printed circuit board.

Fig.103 Peripheral circuit

•In Vcc = 3.3 V operation, connect the Ext. Cap. pin directly to the Vcc pin in order to supply power to the USB transceiver. In addition, you will need to disable the DC-DC converter in this operation (set bit 4 of the USB control register to “0”.) If you are using the bus powered supply in Vcc = 3.3 V operation, the DC-DC converter must be placed outside the MCU.

•In Vcc = 5 V operation, do not connect the external DC-DC converter to the Ext. Cap. pin. Use the built-in DC-DC converter by enabling the USB line driver.

•Make sure the USB D+/D- lines do not cross any other wires. Keep a large GND area to protect the USB lines. Also, make sure you use a USB specification compliant connecter for the connection.

USB Communication

In applications requiring high-reliability, we recommend providing the system with protective measures such as USB function initialization by software or USB reset by the host to prevent USB communication from being terminated unexpectedly, for example due to external causes such as noise.

Clock Input/Output Pin Wiring (Noise Elimination)

(1) Make the wiring for the input/output pins as short as possible.
(2) Make the wiring across the grounding lead of the capacitor which is connected to an oscillator and the Vss pin of the MCU as short as possible (within 20 mm)
(3) Make sure to isolate the oscillation Vss pattern from other patterns for oscillation circuit-use only.

Oscillator Wiring (Noise Elimination)

(1) Keeping oscillator away from large current signal lines

Install a microcomputer (and especially an oscillator) as far as possible from signal lines, including USB signal lines, where a current larger than the tolerance of current value flows. When a large current flows through those signal lines, strong noise occurs because of mutual inductance.

(2) Installing oscillator away from signal lines where potential levels change frequently

Install an oscillator and a connecting pattern of an oscillator away from signal lines where potential levels change frequently. Also, do not cross such signal lines over the clock lines or the signal lines which are sensitive to noise.

Terminate Unused Pins

(1) Output ports : Open

(2) Input ports :

Connect each pin to Vcc or Vss through each resistor of 1 kΩ to 10 kΩ.
Ports that permit the selecting of a built-in pull-up or pull-down resistor can also use this resistor. As for pins whose potential affects to operation modes such as pins CNVss, INT or others, select the Vcc pin or the Vss pin according to their operation mode.

(3) I/O ports :

• Set the I/O ports for the input mode and connect them to Vcc or Vss through each resistor of 1 kΩ to 10 kΩ.
Ports that permit the selecting of a built-in pull-up or pull-down resistor can also use this resistor. Set the I/O ports for the output mode and open them at “L” or “H”.

• When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.

• Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

• At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

Electric Characteristic Differences Between Mask ROM and Flash Memory Version MCUs

There are differences in electric characteristics, operation margin, noise immunity, and noise radiation between Mask ROM and Flash Memory version MCUs due to the difference in the manufacturing processes.
When manufacturing an application system with the Flash Memory version and then switching to use of the Mask ROM version, please perform sufficient evaluations for the commercial samples of the Mask ROM version.

DATA REQUIRED FOR MASK ORDERS

The following are necessary when ordering a mask ROM production:
1. Mask ROM Order Confirmation Form
2. Mark Specification Form
3. Data to be written to ROM, in EPROM form (three identical copies) or one floppy disk.

For the mask ROM confirmation and the mark specifications, refer to the “Renesas Technology Corp.” Homepage (http://www.renesas.com).

FUNCTIONAL DESCRIPTION SUPPLEMENT
Timing After Interrupt

The interrupt processing routine begins with the machine cycle following the completion of the instruction that is currently in execution. Figure 104 shows a timing chart after an interrupt occurs, and Figure 105 shows the time up to execution of the interrupt processing routine.

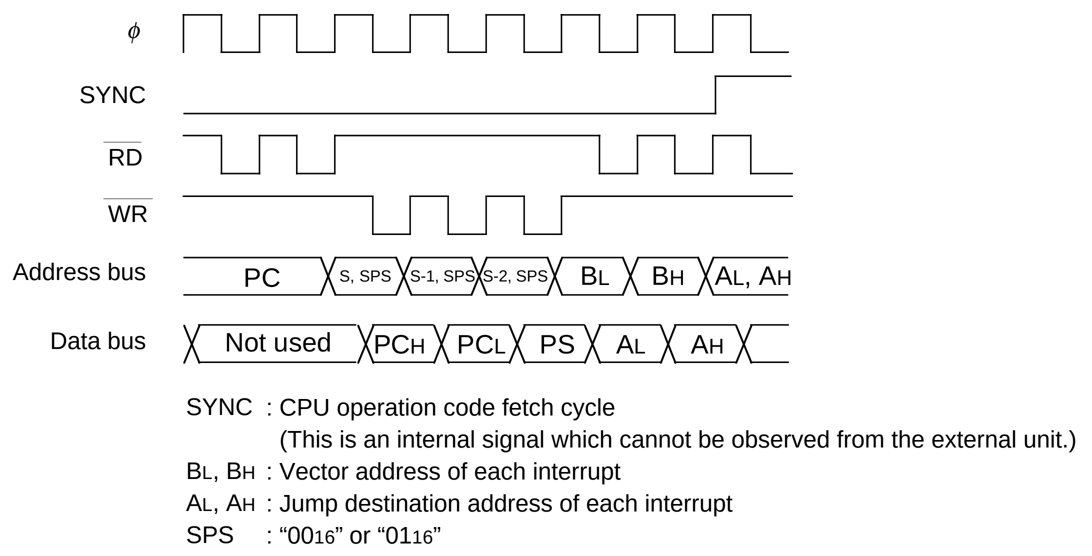


Fig. 104 Timing chart after interrupt occurs

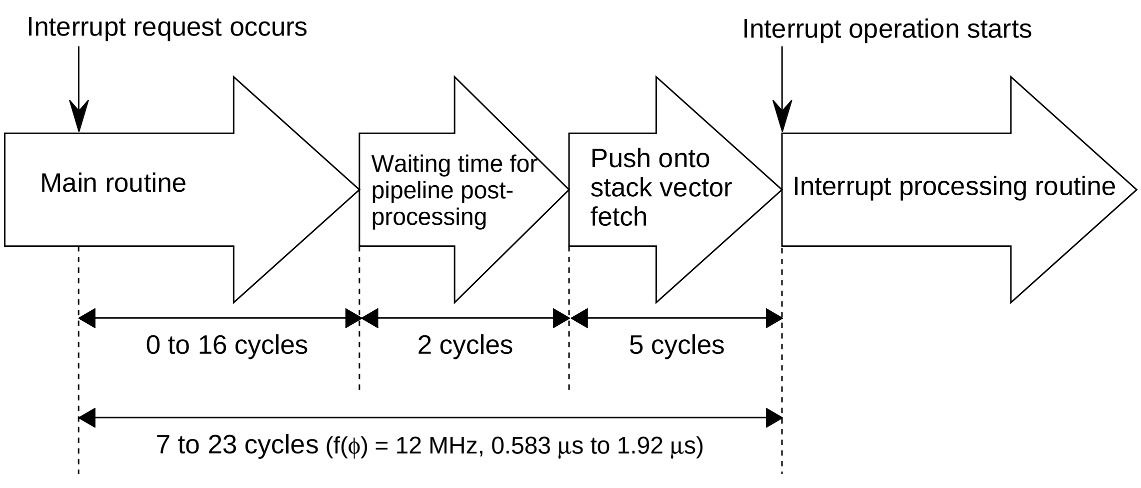


Fig. 105 Time up to execution of interrupt processing routine

THIS PAGE IS BLANK FOR REASONS OF LAYOUT.

CHAPTER 2

APPLICATION

2.1 I/O port

This paragraph explains the registers setting method and the notes related to the I/O port.

2.1.1 Memory map

Address	
0004_{16}	Interrupt request register C (IREQC)
0007_{16}	Interrupt control register C (ICONC)
0008_{16}	Port P0 (P0)
0009_{16}	Port P0 direction register (P0D)
$000A_{16}$	Port P1 (P1)
$000B_{16}$	Port P1 direction register (P1D)
$000C_{16}$	Port P2 (P2)
$000D_{16}$	Port P2 direction register (P2D)
$000E_{16}$	Port P3 (P3)
$000F_{16}$	Port P3 direction register (P3D)
0010_{16}	Port control register (PTC)
0012_{16}	Port P2 pull-up control register (PUP2)
0014_{16}	Port P6 (P6)
0015_{16}	Port P6 direction register (P6D)
0016_{16}	Port P5 (P5)
0017_{16}	Port P5 direction register (P5D)
0018_{16}	Port P4 (P4)
0019_{16}	Port P4 direction register (P4D)
$001A_{16}$	Port P7 (P7)
$001B_{16}$	Port P7 direction register (P7D)
$001C_{16}$	Port P8 (P8)
$001D_{16}$	Port P8 direction register (P8D)

Fig. 2.1.1 Memory map of registers related to I/O port

2.1.2 Related registers

Port Pi

b7 b6 b5 b4 b3 b2 b1 b0

Port Pi (i = 0, 1, 2, 3, 5, 6, 8)
(Pi : addresses 08_{16}, $0A_{16}$, $0C_{16}$, $0E_{16}$, 16_{16}, 14_{16}, $1C_{16}$)

b	Name	Functions	At reset	R	W
0	Port Pi_0	●In output mode Write •••••••• Port latch Read •••••••• Port latch ●In input mode Write •••••••• Port latch Read •••••••• Value of pin	0	○	○
1	Port Pi_1		0	○	○
2	Port Pi_2		0	○	○
3	Port Pi_3		0	○	○
4	Port Pi_4		0	○	○
5	Port Pi_5		0	○	○
6	Port Pi_6		0	○	○
7	Port Pi_7		0	○	○

Fig. 2.1.2 Structure of Port Pi register

Port P4, Port P7

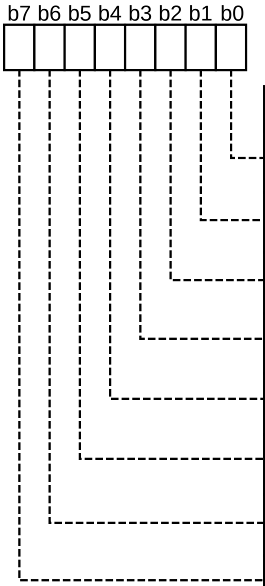
b7 b6 b5 b4 b3 b2 b1 b0

Port P4, Port P7
(P4, P7 : addresses 18_{16}, $1A_{16}$)

b	Name	Functions	At reset	R	W
0	Port $P4_0$ or Port $P7_0$	●In output mode Write •••••••• Port latch Read •••••••• Port latch ●In input mode Write •••••••• Port latch Read •••••••• Value of pin	0	○	○
1	Port $P4_1$ or Port $P7_1$		0	○	○
2	Port $P4_2$ or Port $P7_2$		0	○	○
3	Port $P4_3$ or Port $P7_3$		0	○	○
4	Port $P4_4$ or Port $P7_4$		0	○	○
5	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are undefined.		Undefined	×	×
6			Undefined	×	×
7			Undefined	×	×

Fig. 2.1.3 Structure of Port P4, Port P7 registers

Port Pi direction register

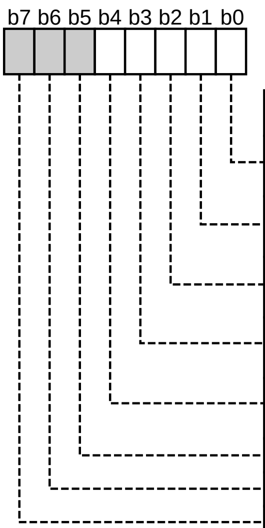


Port Pi direction register (i = 0, 1, 2, 3, 5, 6, 8)
(PiD : addresses 09_{16}, $0B_{16}$, $0D_{16}$, $0F_{16}$, 17_{16}, 15_{16}, $1D_{16}$)

b	Name	Functions	At reset	R	W
0	Port Pi direction register	0 : Port Pi_0 input mode 1 : Port Pi_0 output mode	0	×	○
1		0 : Port Pi_1 input mode 1 : Port Pi_1 output mode	0	×	○
2		0 : Port Pi_2 input mode 1 : Port Pi_2 output mode	0	×	○
3		0 : Port Pi_3 input mode 1 : Port Pi_3 output mode	0	×	○
4		0 : Port Pi_4 input mode 1 : Port Pi_4 output mode	0	×	○
5		0 : Port Pi_5 input mode 1 : Port Pi_5 output mode	0	×	○
6		0 : Port Pi_6 input mode 1 : Port Pi_6 output mode	0	×	○
7		0 : Port Pi_7 input mode 1 : Port Pi_7 output mode	0	×	○

Fig. 2.1.4 Structure of Port Pi direction register (i = 0, 1, 2, 3, 5, 6, 8)

Port P4, P7 direction registers



Port P4 direction register, Port P7 direction register
(P4D, P7D : addresses 19_{16}, $1B_{16}$)

b	Name	Functions	At reset	R	W
0	Port P4 direction register Port P7 direction register	0 : Port $P4_0$ or $P7_0$ input mode 1 : Port $P4_0$ or $P7_0$ output mode	0	×	○
1		0 : Port $P4_1$ or $P7_1$ input mode 1 : Port $P4_1$ or $P7_1$ output mode	0	×	○
2		0 : Port $P4_2$ or $P7_2$ input mode 1 : Port $P4_2$ or $P7_2$ output mode	0	×	○
3		0 : Port $P4_3$ or $P7_3$ input mode 1 : Port $P4_3$ or $P7_3$ output mode	0	×	○
4		0 : Port $P4_4$ or $P7_4$ input mode 1 : Port $P4_4$ or $P7_4$ output mode	0	×	○
5	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are undefined.		Undefined	×	×
6			Undefined	×	×
7			Undefined	×	×

Fig. 2.1.5 Structure of Port P4 direction, Port P7 direction registers

Port control register

b7 b6 b5 b4 b3 b2 b1 b0

Port control register
(PTC : address 10_{16})

b	Name	Functions	At reset	R	W
0	Port P0 to P3 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
1	Port P4 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
2	Port P5 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
3	Port P6 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
4	Port P7 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
5	Port P8 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
6	Port P2 input level select bit	0 : Reduced VIHL level input (**Note 2**) 1 : CMOS level input	0	○	○
7	Master CPU bus input level select bit	0 : CMOS level input 1 : TTL level input	0	○	○

Notes 1: The slew rate function can reduce di/dt by modifying an internal buffer structure.
2: The characteristics of VIHL level is basically the same as that of TTL level. But, its switching center point is a little higher than TTL's.

Fig. 2.1.6 Structure of Port control register

Port P2 pull-up control register

b7 b6 b5 b4 b3 b2 b1 b0

Port P2 pull-upt control register
(PUP2 : address 12_{16})

b	Name	Functions	At reset	R	W
0	Port $P2_0$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
1	Port $P2_1$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
2	Port $P2_2$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
3	Port $P2_3$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
4	Port $P2_4$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
5	Port $P2_5$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
6	Port $P2_6$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○
7	Port $P2_7$ pull-up control bit	0 : Disabled 1 : Enabled	0	○	○

Fig. 2.1.7 Structure of Port P2 pull-up control register

Interrupt request register C

b7 b6 b5 b4 b3 b2 b1 b0

0

Interrupt request register C
(IREQC : address 04_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
1	$CNTR_0$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
2	$CNTR_1$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
3	Serial I/O interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
4	Input buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
5	Output buffer empty interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
6	Key input interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
7	Fix this bit to "0".		0	○	○

✻: "0" can be set by software, but "1" cannot be set.

Fig. 2.1.8 Structure of Interrupt request register C

Interrupt control register C

b7 b6 b5 b4 b3 b2 b1 b0

0

Interrupt control register C
(ICONC : address 07_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
1	$CNTR_0$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
2	$CNTR_1$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
3	Serial I/O interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
4	Input buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
5	Output buffer empty interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
6	Key input interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
7	Fix this bit to "0".		0	○	○

Fig. 2.1.9 Structure of Interrupt control register C

2.1.3 Key-on wake-up interrupt application example

Outline : Key-on wake-up is realized, using internal pull-up resistors.

Figure 2.1.10 shows the registers setting; Figure 2.1.11 shows a connection diagram; Figure 2.1.12 shows the control procedure.

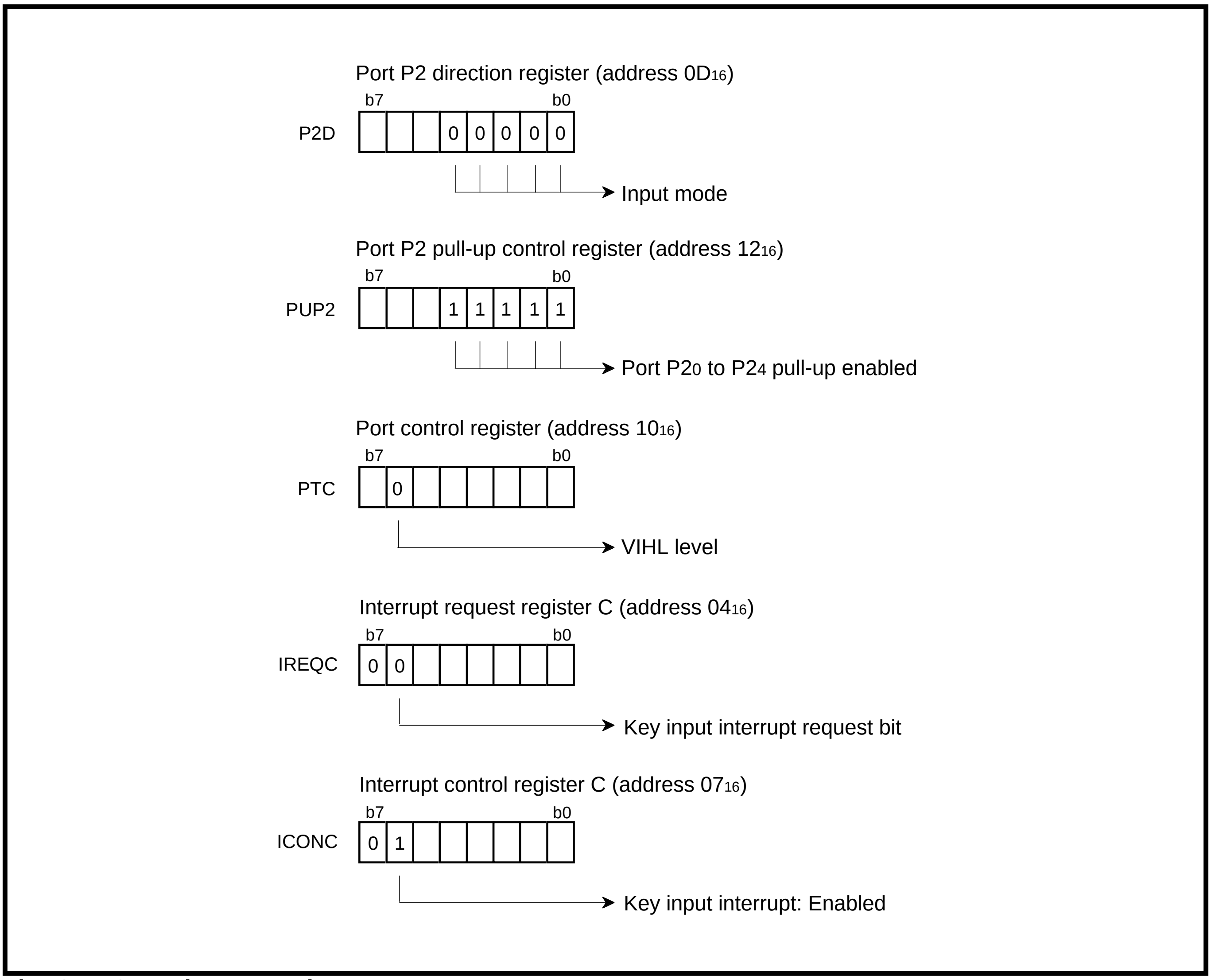


Fig. 2.1.10 Registers setting

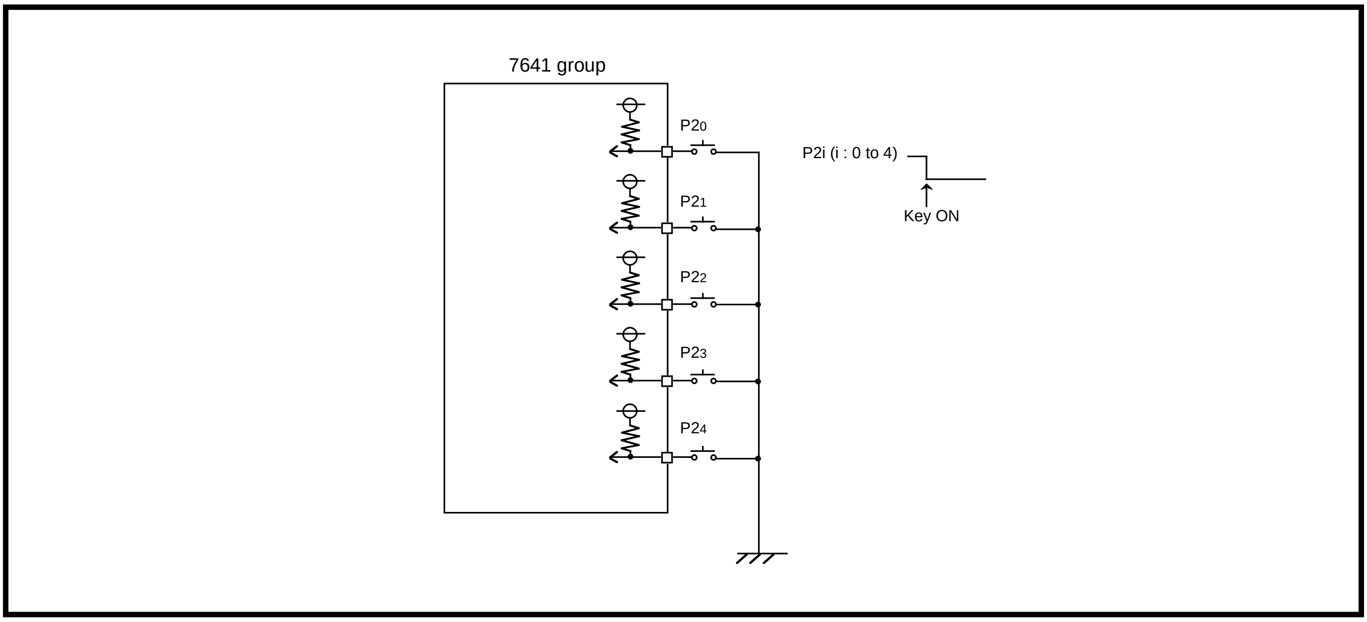


Fig. 2.1.11 Connection diagram

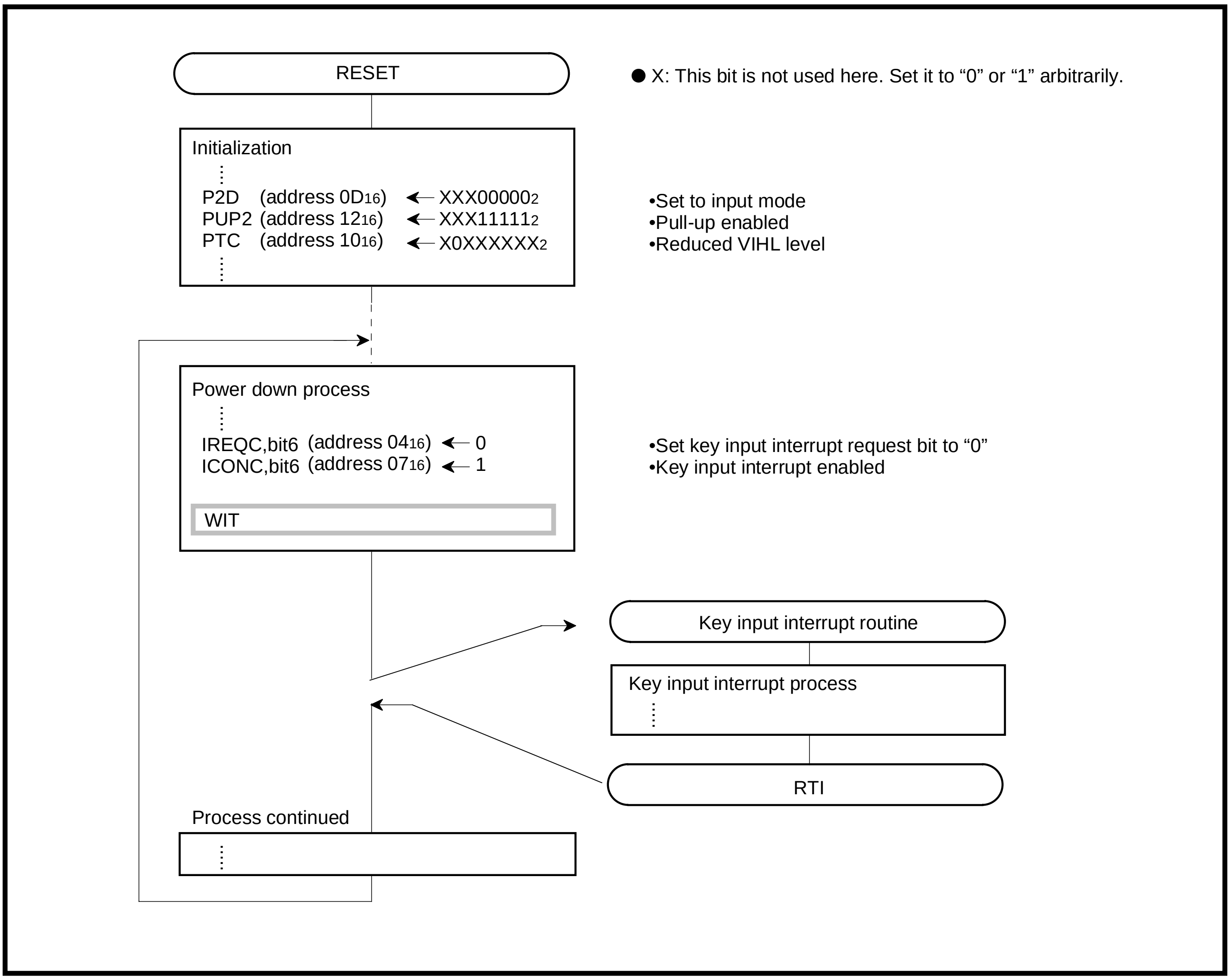


Fig. 2.1.12 Control procedure

2.1.4 Terminate unused pins

Table 2.1.1 Termination of unused pins

Pins/Ports name	Termination
P0, P1, P2, P3, P4, P5, P6, P7, P8	• Set to the input mode and connect each to V_{CC} or V_{SS} through a resistor of 1 kΩ to 10 kΩ. • Set to the output mode and open at “L” or “H” output state.
$\overline{\text{HOLD}}$, $\overline{\text{RDY}}$	Connect to Vcc through a resistor (pull-up).
CNV_{SS}	Connect to Vcc or Vss.
AV_{SS}	Connect to Vss (GND).
AV_{CC}	Connect to Vcc.
X_{OUT}	Open (only when using external clock)
USB D+ USB D-	Open
Ext. Cap.	Connect to Vcc (DC-DC converter disabled) when the USB function is not used.
$\overline{\text{SOF}}$	Open

2.1.5 Notes on I/O port

(1) Notes in standby state

In standby state*1 for low-power dissipation, do not make input levels of an I/O port "undefined". Pull-up (connect the port to V_{CC}) or pull-down (connect the port to V_{SS}) these ports through a resistor.

When determining a resistance value, note the following points:

- External circuit
- Variation of output levels during the ordinary operation

When using built-in pull-up resistor, note on varied current values:

- When setting as an input port : Fix its input level
- When setting as an output port : Prevent current from flowing out to external

● **Reason**

The potential which is input to the input buffer in a microcomputer is unstable in the state that input levels of an I/O port are "undefined". This may cause power source current.

✻1 standby state: stop mode by executing **STP** instruction
wait mode by executing **WIT** instruction

(2) Modifying output data with bit managing instruction

When the port latch of an I/O port is modified with the bit managing instruction*2, the value of the unspecified bit may be changed.

● **Reason**

The bit managing instructions are read-modify-write form instructions for reading and writing data by a byte unit. Accordingly, when these instructions are executed on a bit of the port latch of an I/O port, the following is executed to all bits of the port latch.

- As for bit which is set for input port:
  The pin state is read in the CPU, and is written to this bit after bit managing.
- As for bit which is set for output port:
  The bit value is read in the CPU, and is written to this bit after bit managing.

Note the following:

- Even when a port which is set as an output port is changed for an input port, its port latch holds the output data.
- As for a bit of which is set for an input port, its value may be changed even when not specified with a bit managing instruction in case where the pin state differs from its port latch contents.

✻2 Bit managing instructions: **SEB** and **CLB** instructions

(3) Pull-up control

When using port P2, which includes a pull-up resistor, as an output port, its port pull-up control is invalidated, that is, pull-up cannot be enabled.

● **Reason**

Pull-up/pull-down control is valid only when each direction register is set to the input mode.

2.1.6 Termination of unused pins

(1) Terminate unused pins

① I/O ports :

- Set the I/O ports for the input mode and connect them to V_{CC} or V_{SS} through each resistor of 1 kΩ to 10 kΩ.
  Ports that permit the selecting of a built-in pull-up resistor can also use this resistor. Set the I/O ports for the output mode and open them at "L" or "H".
- When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.
- Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

② The AVss pin when not using the A/D converter :

- When not using the A/D converter, handle a power source pin for the A/D converter, AVss pin as follows:
  AVss: Connect to the Vss pin.

(2) Termination remarks

① I/O ports :
Do not open in the input mode.

● **Reason**

- The power source current may increase depending on the first-stage circuit.
- An effect due to noise may be easily produced as compared with proper termination ② and shown on the above.

② I/O ports :
When setting for the input mode, do not connect to V_{CC} or V_{SS} directly.

● **Reason**

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between a port and V_{CC} (or V_{SS}).

③ I/O ports :
When setting for the input mode, do not connect multiple ports in a lump to V_{CC} or V_{SS} through a resistor.

● **Reason**

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between ports.

- At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

2.2 Timer

This paragraph explains the registers setting method and the notes related to the timers.

2.2.1 Memory map

Address	
0003_{16}	Interrupt request register B (IREQB)
0004_{16}	Interrupt request register C (IREQC)
0006_{16}	Interrupt control register B (ICONB)
0007_{16}	Interrupt control register C (ICONB)
0020_{16}	Timer XL (TXL)
0021_{16}	Timer XH (TXH)
0022_{16}	Timer YL (TYL)
0023_{16}	Timer YH (TYH)
0024_{16}	Timer 1 (T1)
0025_{16}	Timer 2 (T2)
0026_{16}	Timer 3 (T3)
0027_{16}	Timer X mode register (TXM)
0028_{16}	Timer Y mode register (TYM)
0029_{16}	Timer 123 mode register (T123M)

Fig. 2.2.1 Memory map of registers relevant to timers

2.2.2 Related registers

(1) 8-bit timer

Timer i (i = 1 to 3)

b7 b6 b5 b4 b3 b2 b1 b0

Timer 1, Timer 2, Timer 3
(T1, T2, T3: addresses 24_{16}, 25_{16}, 26_{16})

b	Functions	At reset	R	W
0	●Timer i's count value is set through this register. ●Timer 1 and Timer 2 Writing operation depends on the timers 1, 2 write control bit. When it is "0", the values are simultaneously written into their latches and counters. When it is "1", the values are written into only their latches. ●Timer 3 The values are simultaneously written into their latches and counters. ●When reading this register's address, its timer's count values are read out. ●The timer causes an underflow at the count pulse following the count where the timer contents reaches "00_{16}". Then The contents of latches are automatically reloaded into the timer.	(Note)	○	○
1		(Note)	○	○
2		(Note)	○	○
3		(Note)	○	○
4		(Note)	○	○
5		(Note)	○	○
6		(Note)	○	○
7		(Note)	○	○

Note: Timer 1 and Timer 3's values are "FF_{16}". Timer 2 's value are "01_{16}".

Fig. 2.2.2 Structure of Timer i (i=1, 2, 3)

Timer 123 mode register

b7 b6 b5 b4 b3 b2 b1 b0

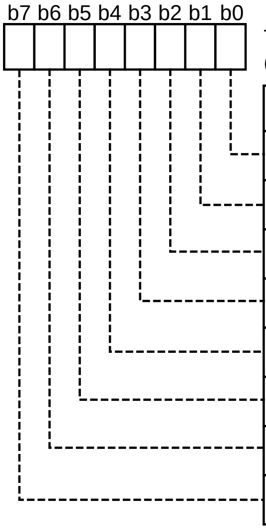
Timer 123 mode register
(T123M : address 29_{16})

b	Name	Functions	At reset	R	W
0	T_{OUT} factor select bit	0 : Timer 1 output 1 : Timer 2 output	0	○	○
1	Timer 1 count stop bit	0 : Count start 1 : Count stop	0	○	○
2	Timer 1 count source select bit	0 : ϕ / 8 1 : f(X_{CIN}) / 2	0	○	○
3	Timer 2 count source select bit	0 : Timer 1 output 1 : ϕ	0	○	○
4	Timer 3 count source select bit	0 : Timer 1 output 1 : ϕ / 8	0	○	○
5	T_{OUT} output active edge switch bit	0 : Start at "H" output 1 : Start at "L" output	0	○	○
6	T_{OUT} output control bit	0 : T_{OUT} output disabled 1 : T_{OUT} output enabled	0	○	○
7	Timers 1, 2 write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	○	○

Fig. 2.2.3 Structure of Timer 123 mode register

(2) 16-bit timer

Timer X (low, high)



Timer XL, Timer XH
(TXL, TXH: addresses 20_{16}, 21_{16})

b	Functions	At reset	R	W
0	●Timer X's count value is set through this register. ●Writing operation depends on the timer X write control bit. When it is "0", the values are simultaneously written into timer X latch and counter. When it is "1", the values are written into only timer X latch. ●Timer X is a down-count timer. ●When reading this register's address, the timer X's count value is read out.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Notes 1: Read and write operation to timer X must be performed for both high and low-order bytes.
2: When reading timer X, read the high-order byte first and then the low-order byte.
3: When writing to timer X, write the low-order byte first and then the high-order byte.
4: Do not read this register during the write operation, or do not write during the read operation.

Fig. 2.2.4 Structure of Timer X (low-order, high-order)

Timer X mode register

b7 b6 b5 b4 b3 b2 b1 b0

Timer X mode register
(TXM : address 27_{16})

b	Name	Functions	At reset	R	W
0	Timer X write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	O	O
1	Timer X count source select bits	b2b1 0 0 : φ / 8 0 1 : φ / 16 1 0 : φ / 32 1 1 : φ / 64	0	O	O
2			0	O	O
3	Timer X internal clock select bit	0 : φ / n (n = 8, 16, 32, 64) 1 : SCSGCLK (Special Count Source Generator)	0	O	O
4	Timer X operating mode bits	b5b4 0 0 : Timer mode 0 1 : Pulse output mode 1 0 : Event counter mode 1 1 : Pulse width measurement mode	0	O	O
5			0	O	O
6	$CNTR_0$ active edge switch bit	This function depends on the timer X operating mode. (See below.)	0	O	O
7	Timer X count stop bit	0 : Count start 1 : Count stop	0	O	O

Function of $CNTR_0$ active edge switch bit

Timer X operating mode	$CNTR_0$ active edge switch bit
Pulse output mode	0 : Starts at “H” output 1 : Starts at “L” output
Event counter mode	0 : Counts at rising edge 1 : Counts at falling edge
Pulse width measurement mode	0 : Measures “H” pulse width 1 : Measures “L” pulse width
Interrupt	0 : Falling edge active 1 : Rising edge active

Fig. 2.2.5 Structure of Timer X mode register

Timer Y (low, high)

b7 b6 b5 b4 b3 b2 b1 b0

Timer YL, Timer YH
(TYL, TYH: addresses 22_{16}, 23_{16})

b	Functions	At reset	R	W
0	●Timer Y’s count value is set through this register. ●Timer Y is a down-count timer. ●When reading this register’s address, the timer Y’s count value is read out.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Notes 1: Read and write operation to timer X must be performed for both high and low-order bytes.
2: When reading timer X, read the high-order byte first and then the low-order byte.
3: When writing to timer X, write the low-order byte first and then the high-order byte.
4: Do not read this register during the write operation, or do not write during the read operation.

Fig. 2.2.6 Structure of Timer Y (low-order, high-order)

Timer Y mode register

b7 b6 b5 b4 b3 b2 b1 b0

Timer Y mode register
(TYM : address 28_{16})

b	Name	Functions	At reset	R	W
0	Timer Y write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	O	O
1	Timer Y output control bit	0 : TYOUT output disabled 1 : TYOUT output enabled	0	O	O
2	Timer Y count source select bits	b3b2 0 0 : φ / 8 0 1 : φ / 16 1 0 : φ / 32 1 1 : φ / 64	0	O	O
3			0	O	O
4	Timer Y operating mode bits	b5b4 0 0 : Timer mode 0 1 : Period measurement mode 1 0 : Event counter mode 1 1 : Pulse width HL continuously measurement mode	0	O	O
5			0	O	O
6	CNTR1 active edge switch bit	This function depends on the timer Y operating mode. (See below.)	0	O	O
7	Timer Y count stop bit	0 : Count start 1 : Count stop	0	O	O

Function of CNTR1 active edge switch bit

Timer Y operating mode	CNTR1 active edge switch bit
Period measurement mode	0 : Measures between falling edges 1 : Measures between rising edges
Event counter mode	0 : Counts at rising edge 1 : Counts at falling edge
TYOUT output	0 : Starts at "H" output 1 : Starts at "L" output
Interrupt	0 : Falling edge active 1 : Rising edge active

Fig. 2.2.7 Structure of Timer Y mode register

(3) 8-bit timer, 16-bit timer

Interrupt request register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register B
(IREQB : address 03_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	UART2 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	UART2 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	UART2 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	Timer X interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	Timer Y interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	Timer 1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	Timer 2 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻

✻: "0" can be set by software, but "1" cannot be set.

Fig. 2.2.8 Structure of Interrupt request register B

Interrupt request register C

b7 b6 b5 b4 b3 b2 b1 b0

0

Interrupt request register C
(IREQC : address 04_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	$CNTR_0$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	$CNTR_1$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	Serial I/O interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	Input buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	Output buffer empty interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	Key input interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	Fix this bit to "0".		0	O	O

✻: "0" can be set by software, but "1" cannot be set.

Fig. 2.2.9 Structure of Interrupt request register C

Interrupt control register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register B
(ICONB : address 06_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	UART2 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	UART2 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	UART2 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	Timer X interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	Timer Y interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	Timer 1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	Timer 2 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O

Fig. 2.2.10 Structure of Interrupt control register B

Interrupt control register C

b7 b6 b5 b4 b3 b2 b1 b0
0

Interrupt control register C
(ICONC : address 07_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	$CNTR_0$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	$CNTR_1$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	Serial I/O interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	Input buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	Output buffer empty interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	Key input interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	Fix this bit to “0”.		0	O	O

Fig. 2.2.11 Structure of Interrupt control register C

2.2.3 Timer application examples

(1) Basic functions and uses

[Function 1] Control of event interval (Timer 1 to Timer 3, Timer X and Y: Timer mode)
When a certain time, by setting a count value to each timer, has passed, the timer interrupt request occurs.
<Use>
•Generating of an output signal timing
•Generating of a wait time

[Function 2] Control of cyclic operation (Timer 1 to Timer 3, Timer X and Y: Timer mode)
The value of the timer latch is automatically written to the corresponding timer each time the timer underflows, and each timer interrupt request occurs in cycles.
<Use>
•Generating of cyclic interrupts
•Clock function (measurement of 1 s); see "(2) Timer application example 1"
•Control of a main routine cycle

[Function 3] Output of rectangular waveform
(Timer 1, Timer 2, Timer X: Pulse output mode; Timer Y: TY_{OUT} output)
The output levels of the T_{OUT}, $CNTR_0$ and $CNTR_1$ pins are inverted each time the timer underflows.
<Use>
•Piezoelectric buzzer output; see "(3) Timer application example 2"
•Generating of the remote control carrier waveforms

[Function 4] Count of external pulses (Timer X, Timer Y: Event counter mode)
External pulses input to the $CNTR_0$ pin and $CNTR_1$ pin are respectively counted as the timer count source (in the event counter mode).
<Use>
•Frequency measurement; see "(4) Timer application example 3"
•Division of external pulses
•Generating of interrupts due to a cycle using external pulses as the count source; count of a reel pulse

[Function 5] Measurement of external pulse width 1 (Timer X: Pulse width measurement mode)
The "H" or "L" level width of external pulses input to $CNTR_0$ pin is measured.
<Use>
•Measurement of external pulse frequency (measurement of pulse width of FG pulse* for a motor); see "(5) Timer application example 4"
•Measurement of external pulse duty (when the frequency is fixed)
FG pulse*: Pulse used for detecting the motor speed to control the motor speed.

[Function 6] Measurement of external pulse width 2 (Timer Y: Period measurement mode)
The external pulse width input to $CNTR_1$ pin is measured.
<Use>
•Measurement of phase control signal; see "(6) Timer application example 5"

(2) Timer application example 1: Clock function (measurement of 1 s)

Outline: The input clock is divided by the timer so that the clock can count up at 1 s intervals.

Specifications:
- The clock $f(X_{CIN})$ = 32 kHz is divided by the timer.
- The timer 2 interrupt request bit is checked in main routine, and if the interrupt request is issued, the clock is counted up.
- The timer 1 interrupt occurs every 10 ms to execute processing of other interrupts.

Figure 2.2.12 shows the timers connection and setting of division ratios; Figure 2.2.13 shows the related registers setting; Figure 2.2.14 shows the control procedure.

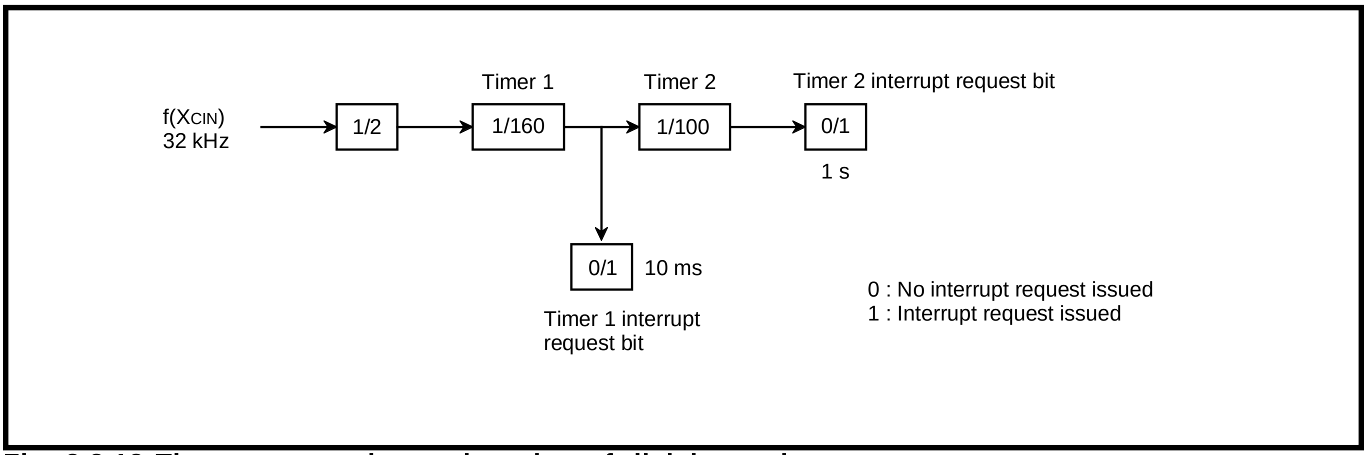


Fig. 2.2.12 Timers connection and setting of division ratios

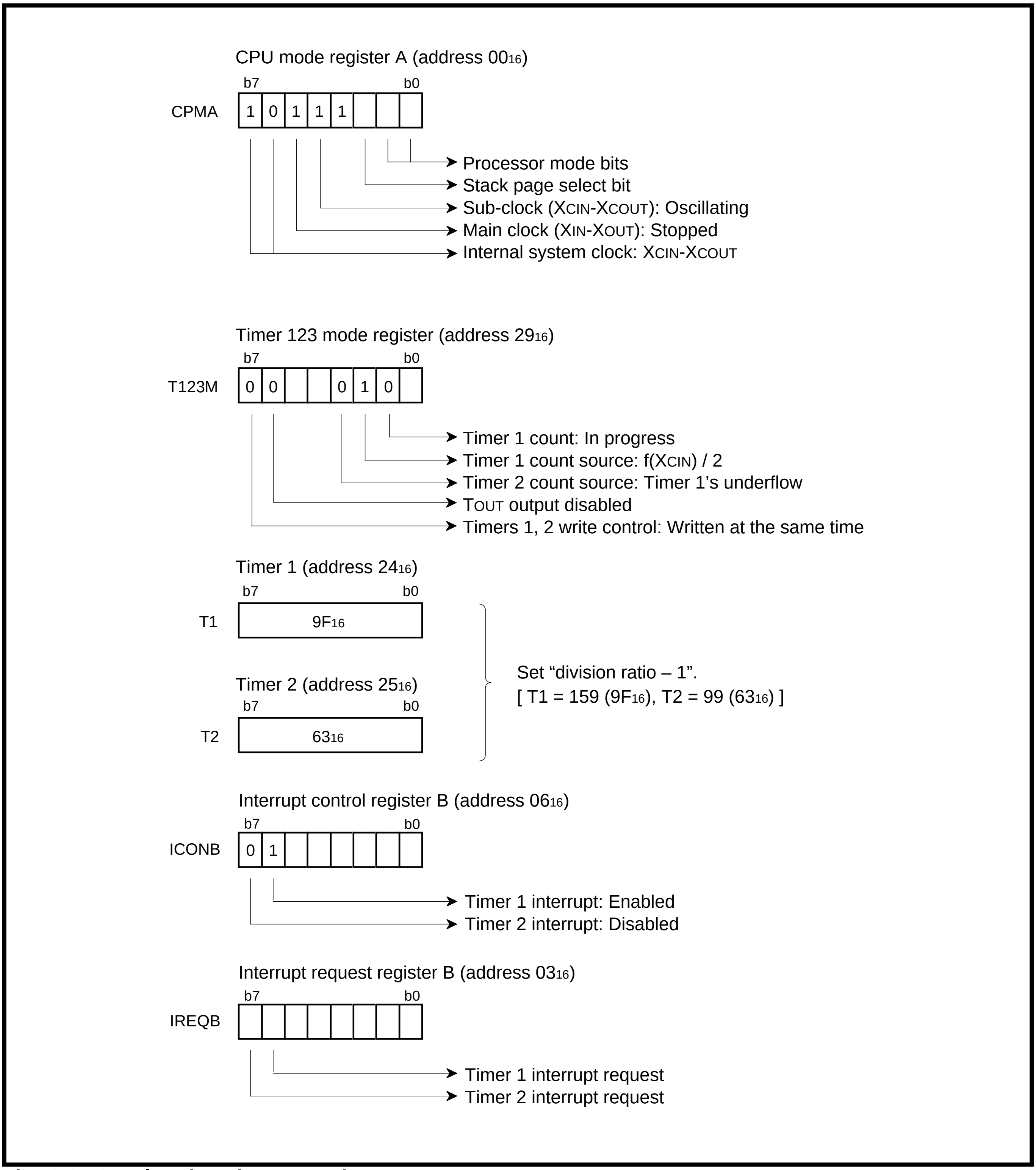


Fig. 2.2.13 Related registers setting

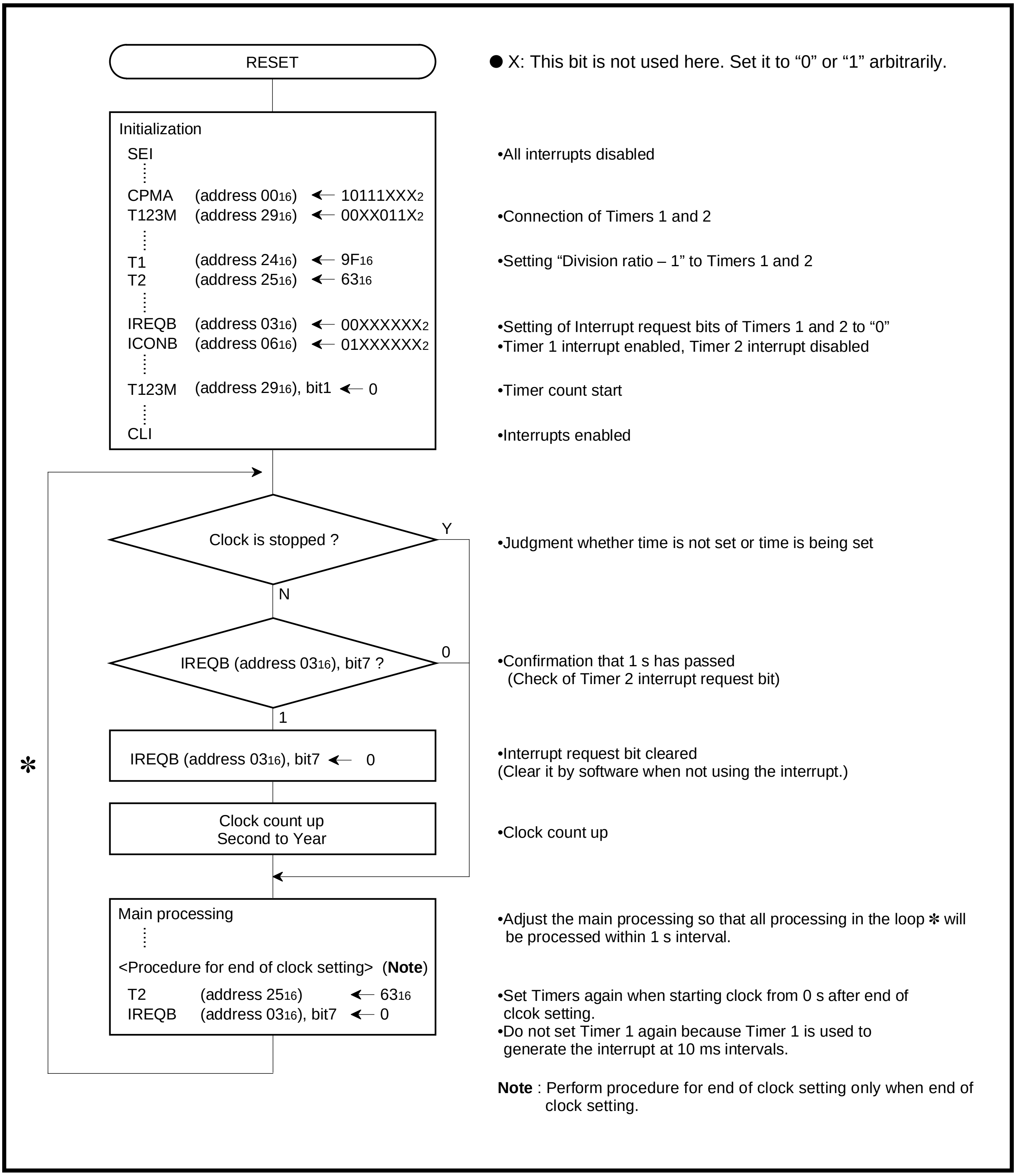


Fig. 2.2.14 Control procedure

(3) Timer application example 2: Piezoelectric buzzer output

Outline: The rectangular waveform output function of the timer is applied for a piezoelectric buzzer output.

Specifications: •The rectangular waveform, dividing the clock f(X_{IN}) = 4.19 MHz (2^{22} Hz) into about 2 kHz (2048 Hz), is output from the $P4_3/CNTR_0$ pin.

•The level of the $P4_3/CNTR_0$ pin is fixed to “H” while a piezoelectric buzzer output stops.

Figure 2.2.15 shows a peripheral circuit example, and Figure 2.2.16 shows the timers connection and setting of division ratios. Figure 2.2.17 shows the related registers setting, and Figure 2.2.18 shows the control procedure.

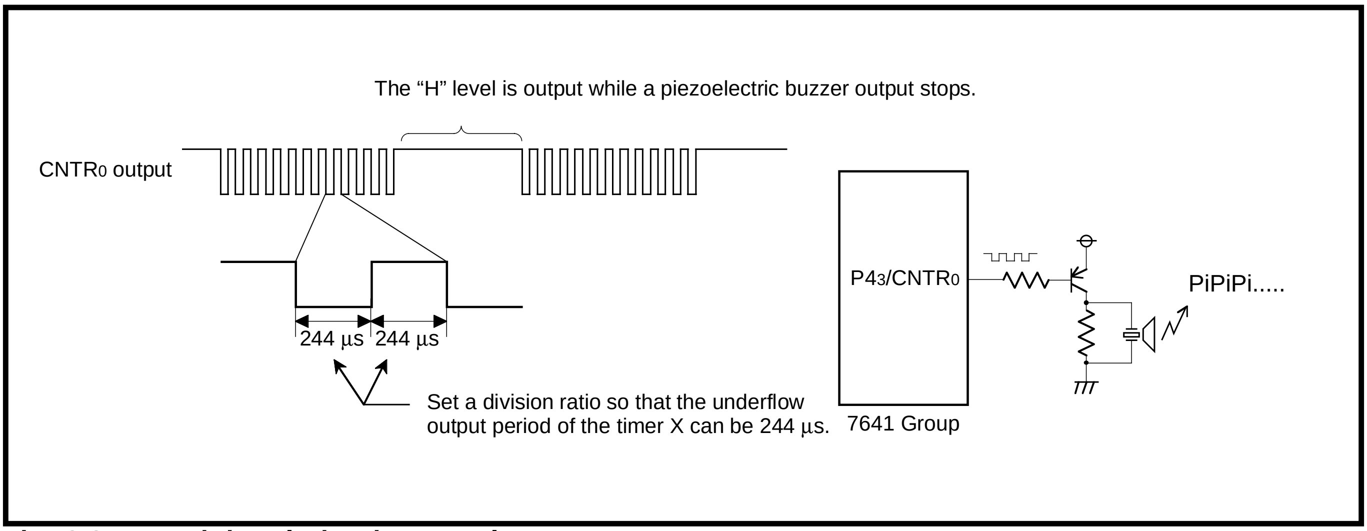


Fig. 2.2.15 Peripheral circuit example

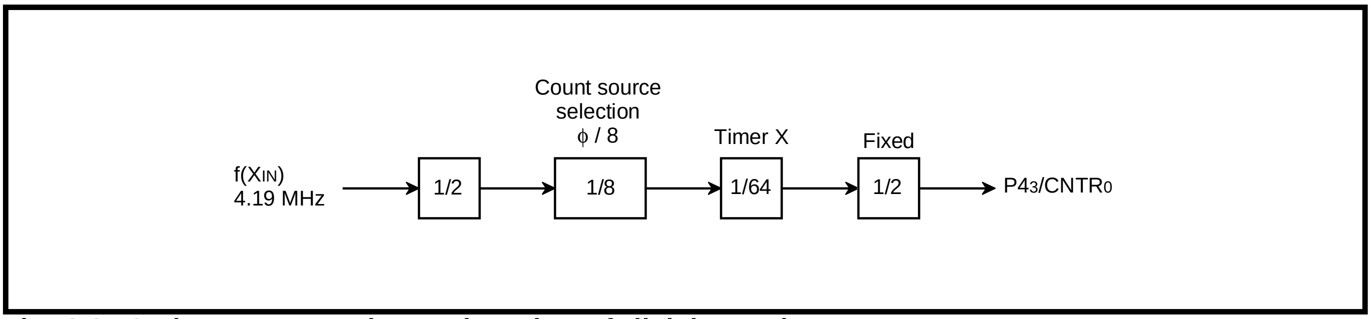


Fig. 2.2.16 Timers connection and setting of division ratios

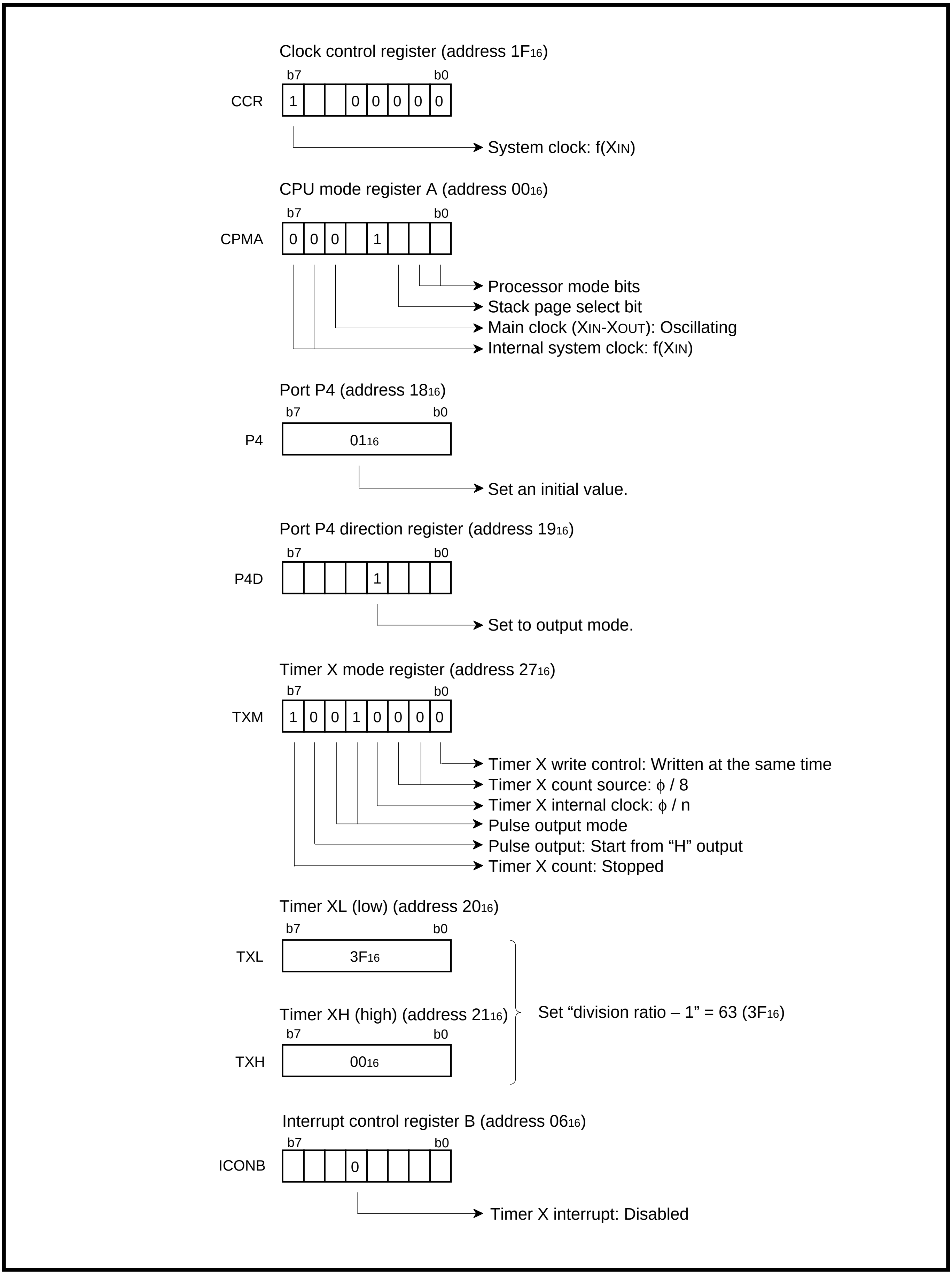


Fig. 2.2.17 Relevant registers setting

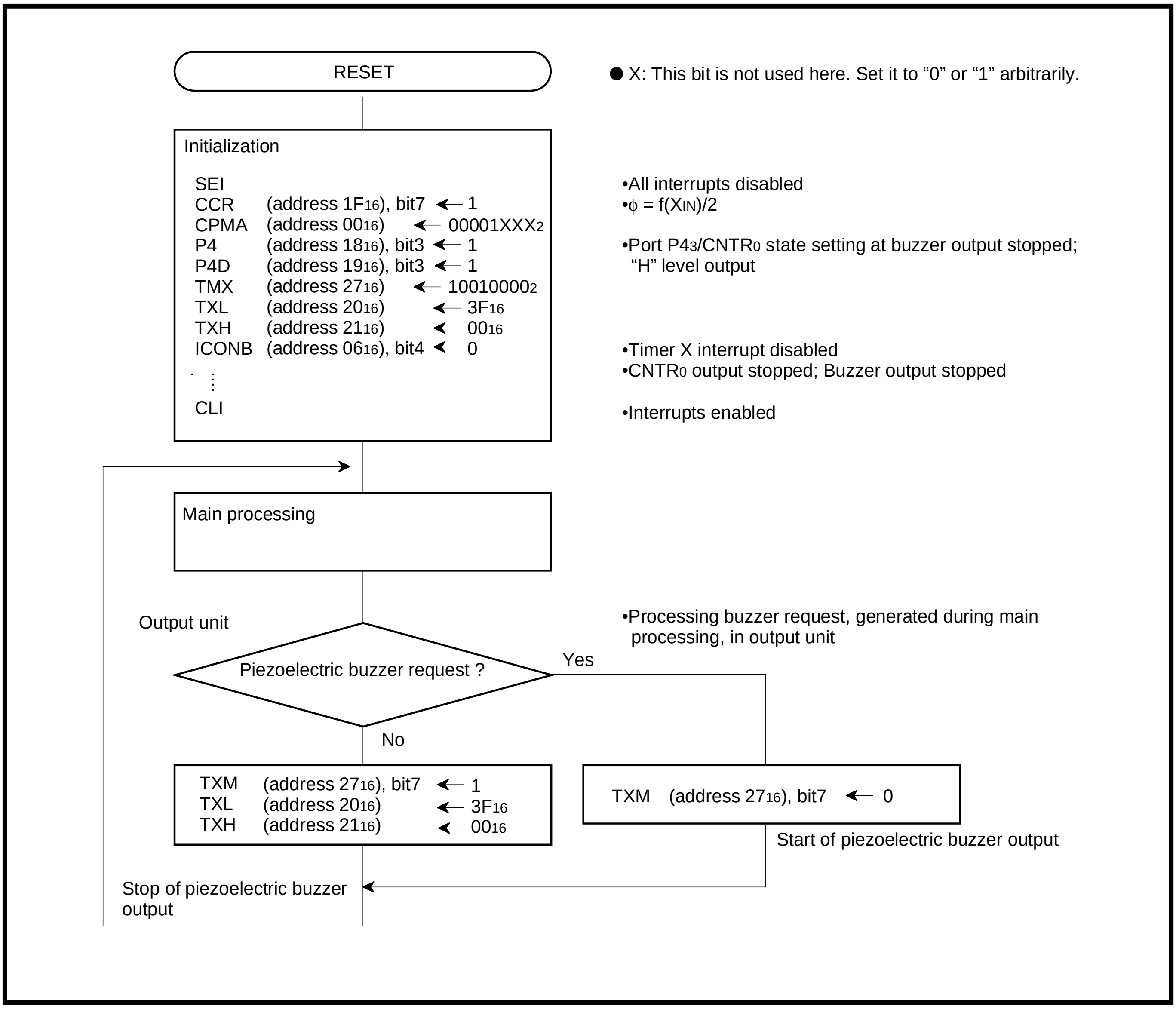


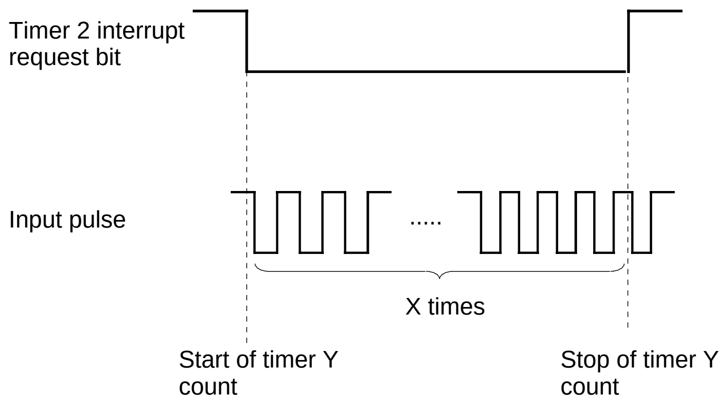
Fig. 2.2.18 Control procedure

(4) Timer application example 3: Frequency measurement

Outline: The pulse frequency input to $P4_4/CNTR_1$ pin is measured by measuring the event number within a fixed term.

Specifications:
•The pulse is input to the $P4_4/CNTR_1$ pin and counted by the timer Y.
•A count value of timer Y is read out at 1 ms intervals, which is the timer 2 interrupt interval. As the result, the frequency can be calculated.
(This example is in $f(X_{IN})$ = 24 MHz and $\phi = f(X_{IN})/4$.)
•The input event number must be "$FFFF_{16}$" within 1 ms.

Figure 2.2.19 shows how to measure the frequency; Figure 2.2.20 shows the related registers setting; Figure 2.2.21 shows the control procedure.



Note: Frequency = $\frac{\text{X times}}{\text{1 ms}}$ kHz

Fig. 2.2.19 How to measure frequency

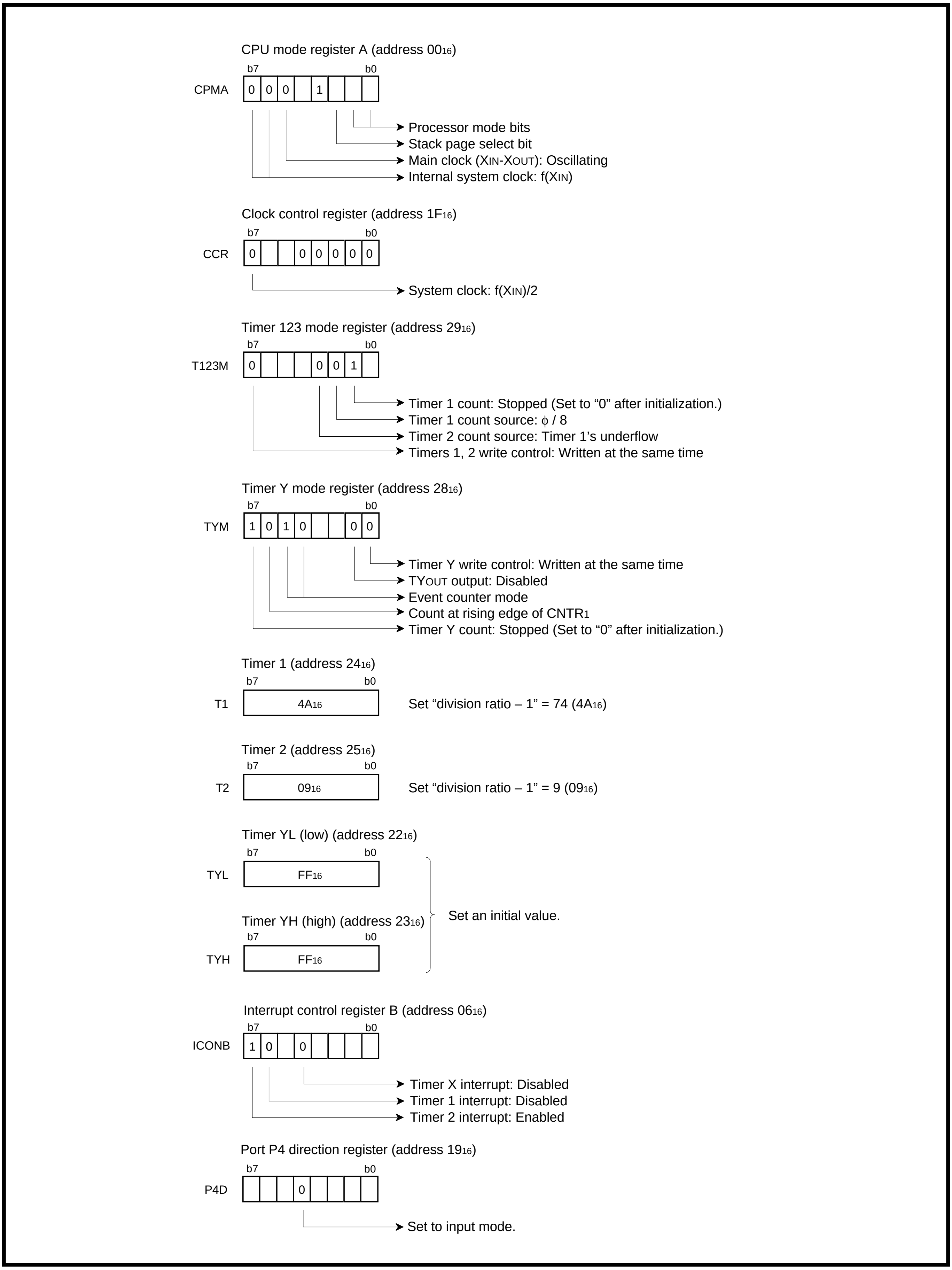


Fig. 2.2.20 Related registers setting

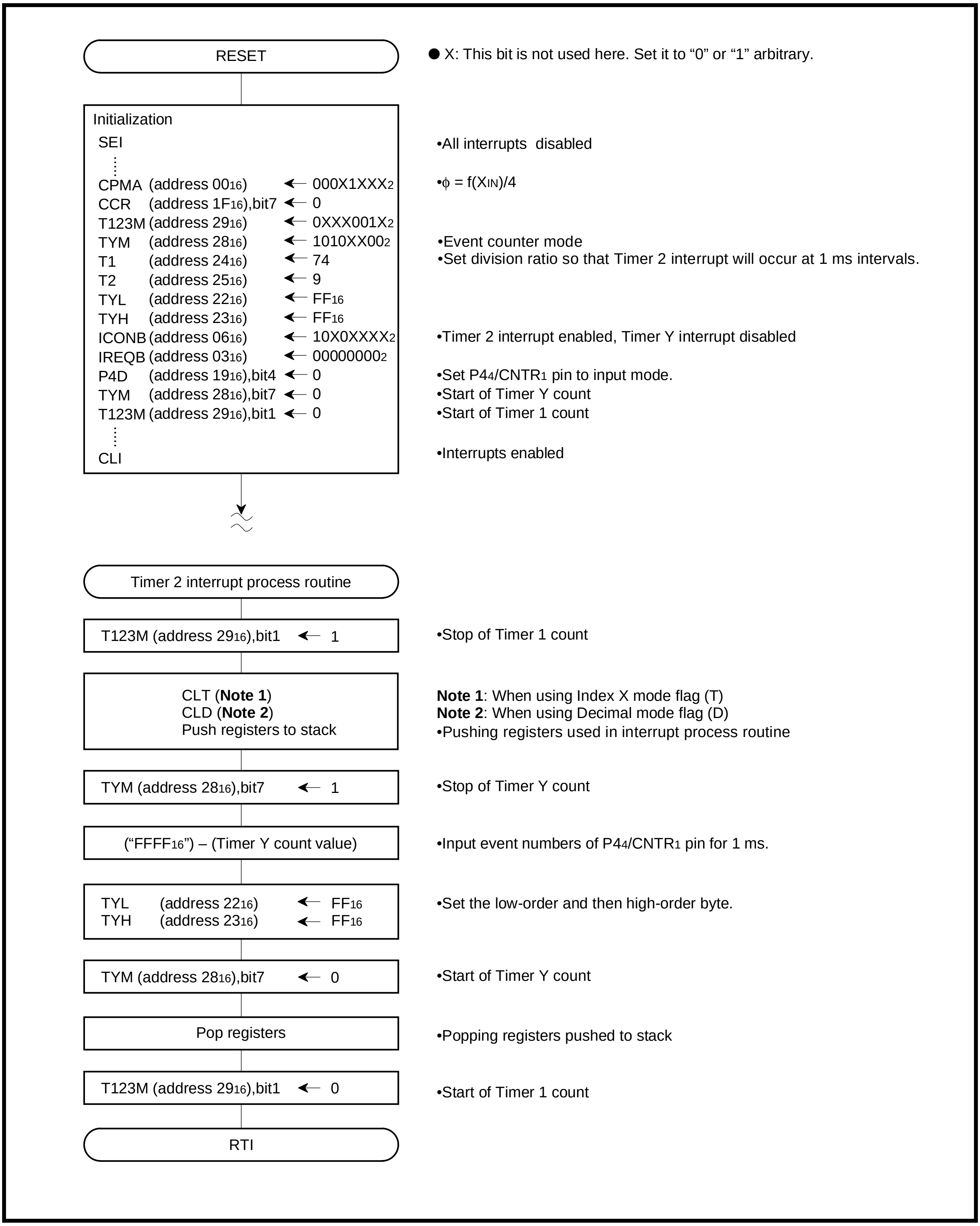


Fig. 2.2.21 Control procedure

(5) Timer application example 4: Measurement of FG pulse width for motor

Outline: The timer X counts the “H” level width of the pulses input to the $P4_3/CNTR_0$ pin.

Specifications: •The timer X counts the “H” level width of the FG pulse input to the $P4_3/CNTR_0$.

<Example>

When $f(X_{IN})$ = 24 MHz and ϕ = 6 MHz, the count source is 10.6 µs, which is obtained by dividing the ϕ by 64. Measurement can be made up to 1 s in the range of $FFFF_{16}$ to 0000_{16}.

Figure 2.2.22 shows the timers connection and setting of division ratio; Figure 2.2.23 shows the related registers setting; Figures 2.2.24 and 2.2.25 show the control procedure.

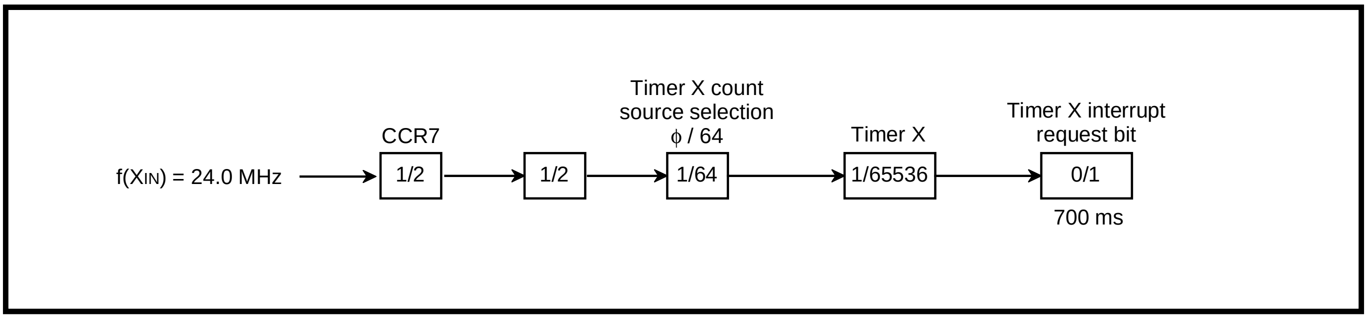


Fig. 2.2.22 Timers connection and setting of division ratios

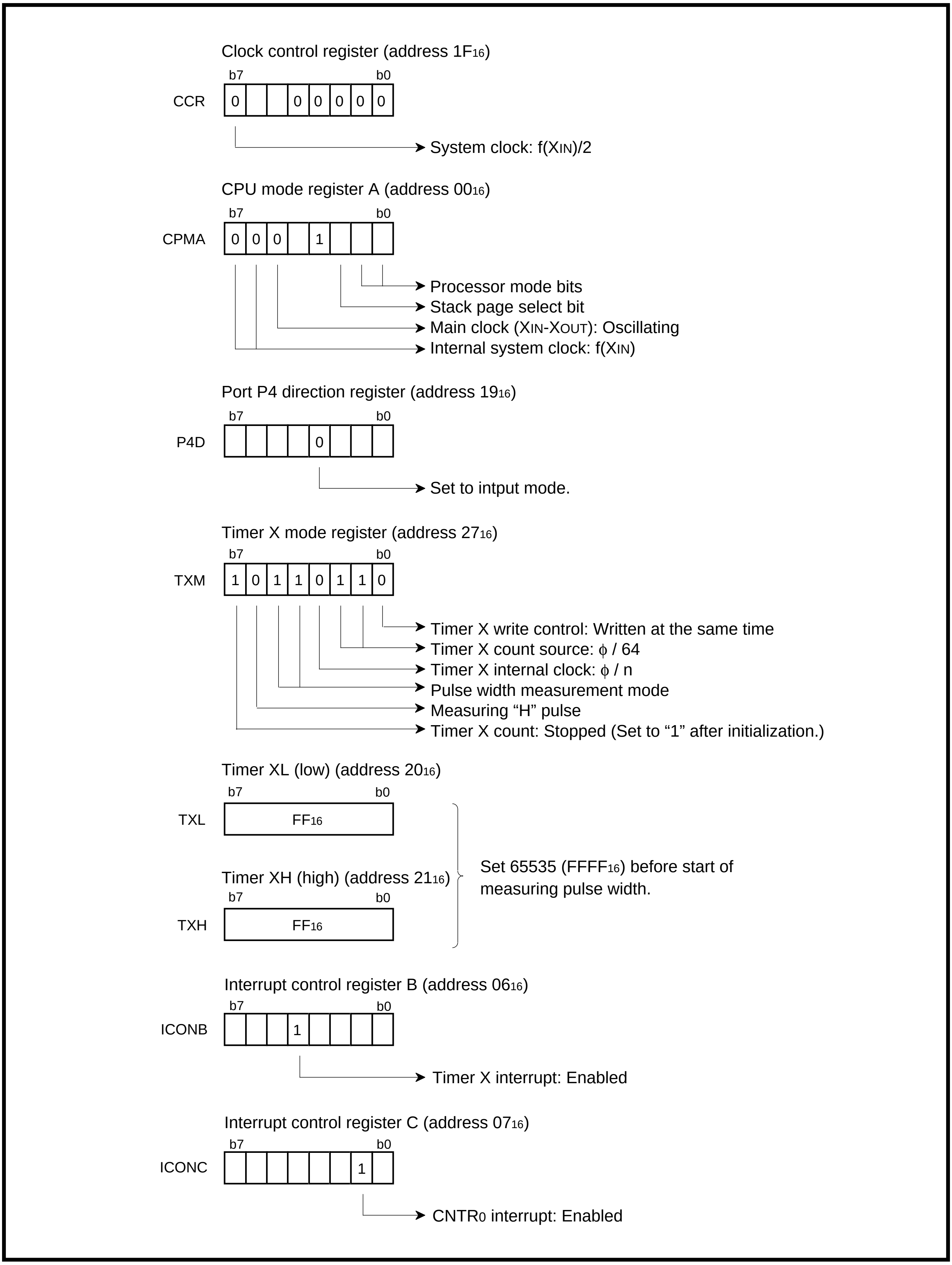


Fig. 2.2.23 Relevant registers setting

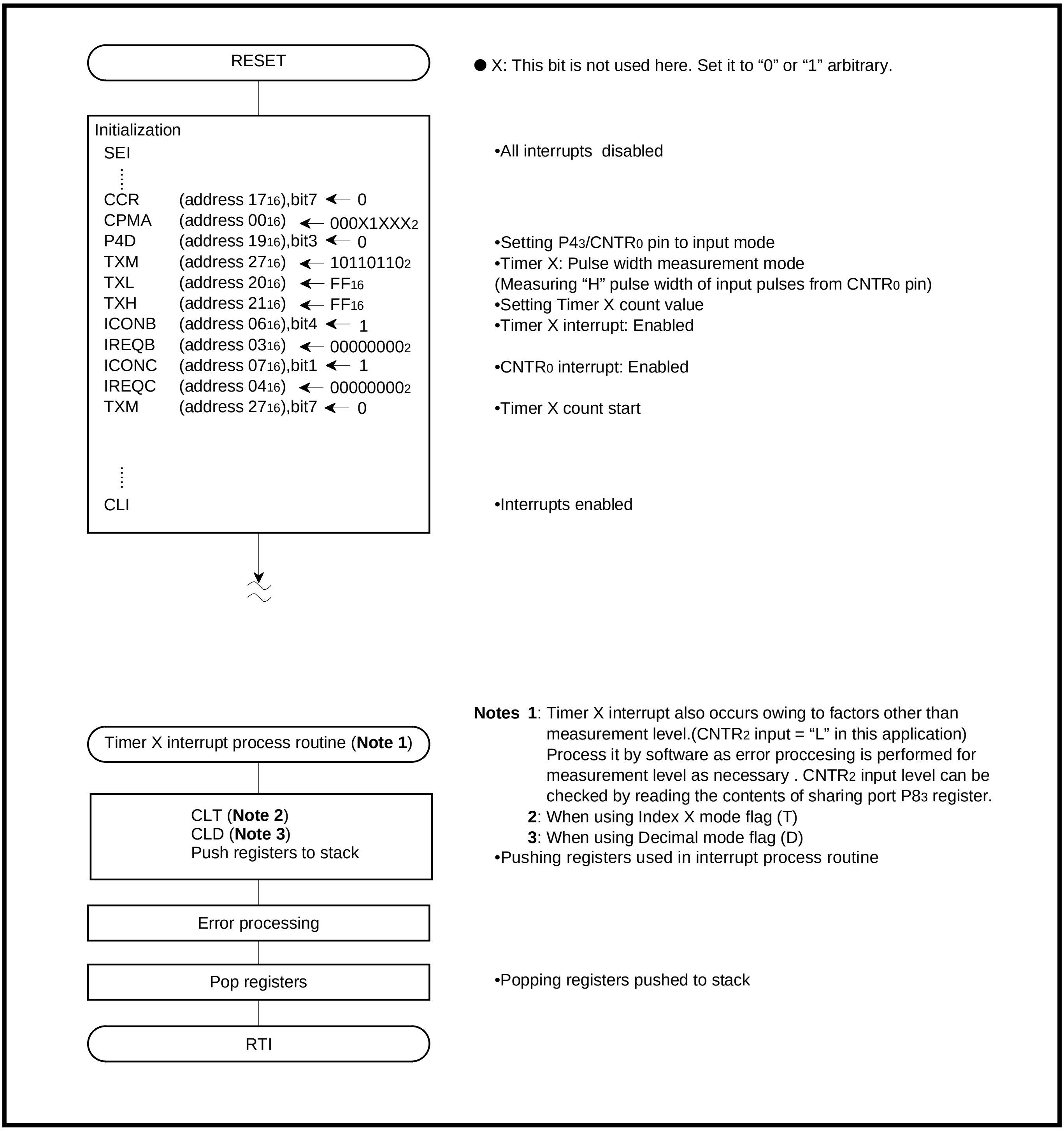


Fig. 2.2.24 Control procedure (1)

CNTR$_0$ interrupt process routine

↓

CLT (**Note 1**)
CLD (**Note 2**)
Push registers to stack

— **Notes 1**: When using Index X mode flag (T)
2: When using Decimal mode flag (D)
•Pushing registers used in interrupt process routine

↓

(A) ← TXH
Measurement result (high-order 8 bits) ← (A)
(A) ← TXL
Measurement result (low-order 8 bits) ← (A)
TXL (address 20_{16}) ← FF_{16}
TXH (address 21_{16}) ← FF_{16}

— •Count value read and storing it to RAM
Read the high-order and then low-order byte.
Write to the low-order and then high-order byte.

↓

Pop registers

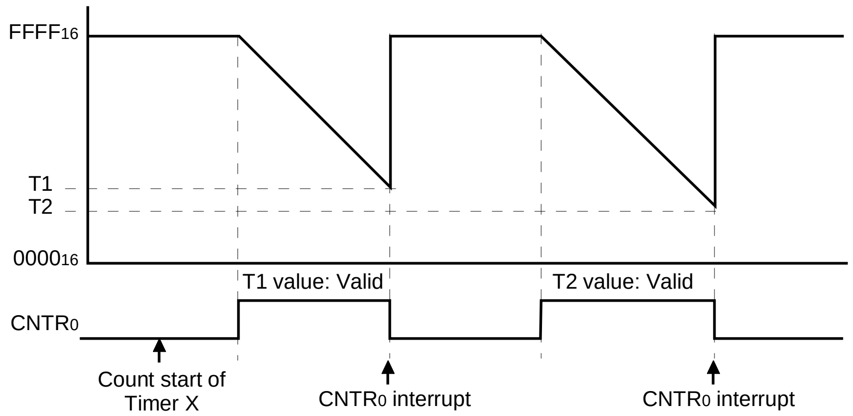
— •Popping registers pushed to stack

↓

RTI

Note : The first value becomes invalid depending on start timing of Time X count shown by the following figure.
Process it by software as necessary.

[Example 1] • Start Timer X count when CNTR$_0$ input level is "L".
(CNTR$_0$ input level can be checked by reading the contents of sharing port P4$_3$ register.



[Example 2] • Start Timer X count when CNTR$_0$ input level is "H".
Invalidate the first CNTR$_0$ interrupt after start of Timer X count.

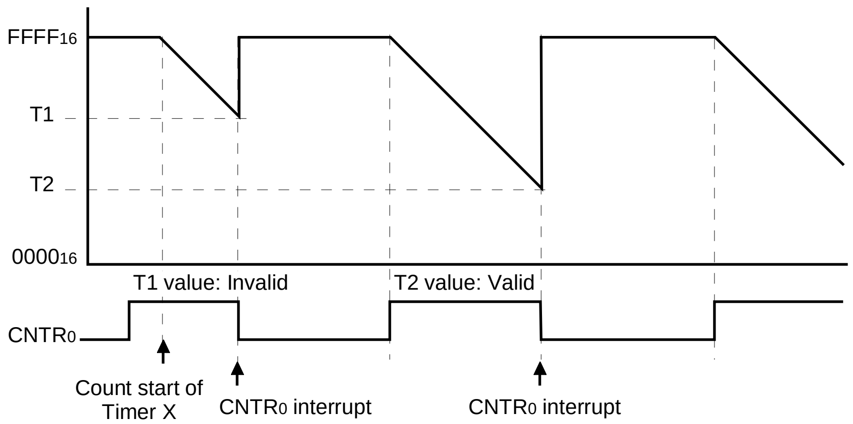


Fig. 2.2.25 Control procedure (2)

(6) Timer application example 5: Adjustment of phase control signal

Outline: A phase control signal is adjusted in the period measurement mode.

Specifications: •To control the phase of the load, a phase control signal is output to the load.
•The pulse width of feedback signal input from the load is measured and a phase control signal for the load is adjusted.

Figure 2.2.26 shows the circuit example; Figure 2.2.27 shows the related registers setting; Figure 2.2.28 shows the control procedure.

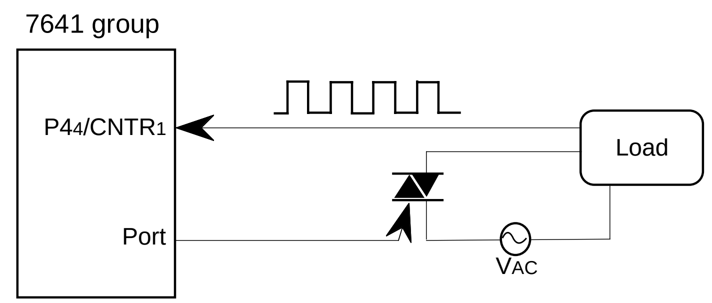


Fig. 2.2.26 Circuit example

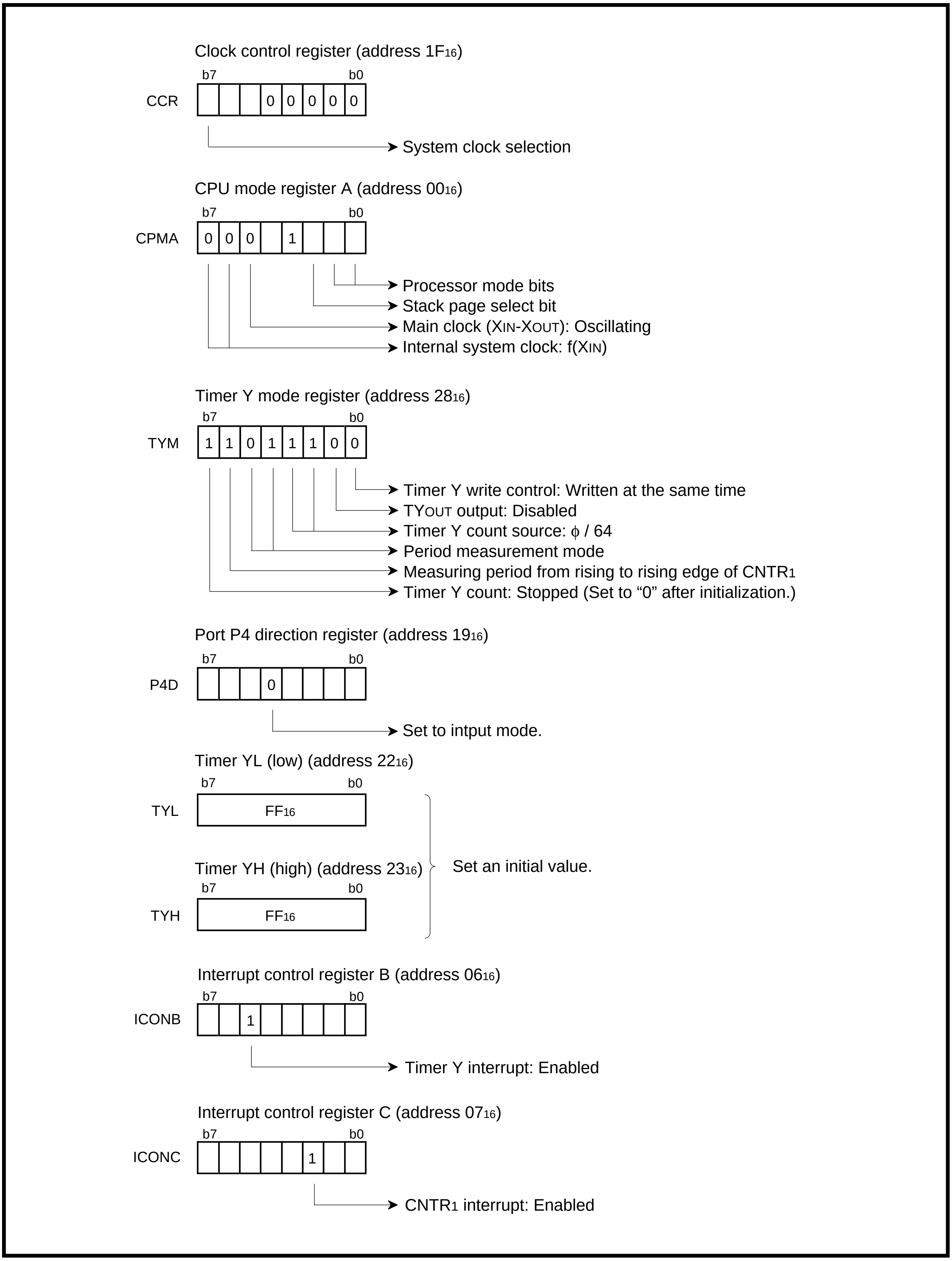


Fig. 2.2.27 Related registers setting

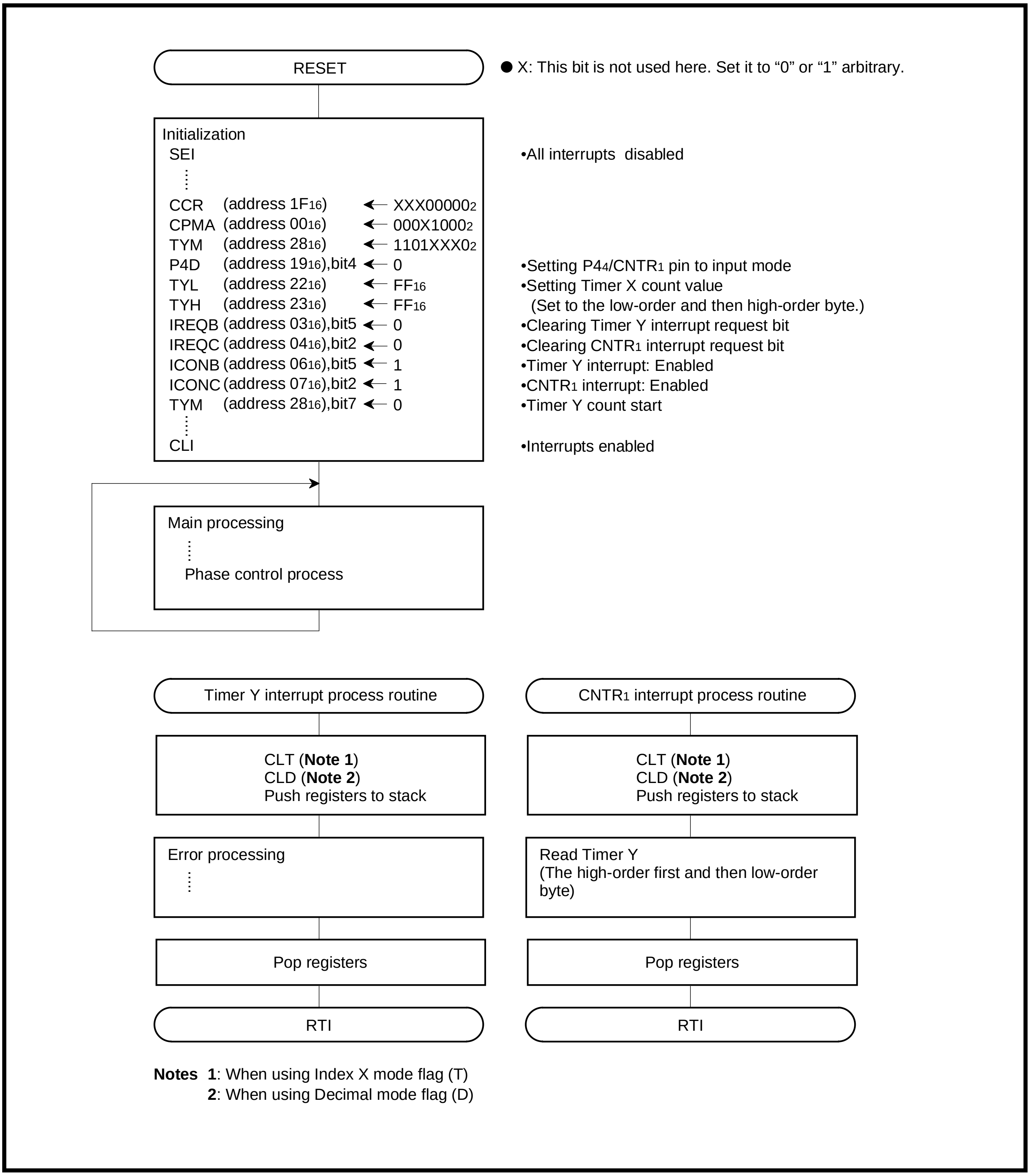


Fig. 2.2.28 Control procedure

2.2.4 Notes on timer

(1) Read/Write for timer

•The timer division ratio is : 1 / (n + 1)
(n = “0” to “255” written into the timer)

•Read and write operation on 16-bit timer (Timers X and Y) must be performed for both high and low-order bytes.

•When reading the 16-bit timer (Timers X and Y), read the high-order byte first and then the low-order byte. When writing to the 16-bit timer, write the low-order byte first and then the high-order byte.

•Do not read the 16-bit timer during the write operation, or do not write to it during the read operation.

•When the value is loaded only in the latch, the value is loaded in the timer at the count pulse following the count where the timer reaches “00_{16}”.

•In the timers 1 to 3, switching of the count sources of timers 1 to 3 does not affect the values of reload latches. However, that may make count operation started. Therefore, write values again in the order of timers 1, 2 and then timer 3 after their count sources have been switched.

•In the timer mode (for timers X, Y, 1 to 3), event counter mode (for timers X, Y), pulse output mode (for timers X, Y, 1, 2), the timer current count value can be read out by reading the timer.

•In the pulse width measurement mode (for timer X), period measurement mode (for timer Y), pulse width HL continuously measurement mode (for timer Y), the measured timer value is stored into the internal temporary register. When reading the timer, the value of internal temporary register is read out. The contents of internal temporary register is updated after the next measurement.

(2) Pulse output

•When using the pulse output mode of timer X, set bit 3 of port P4 direction register to “1” (output mode).

•When using the TY_{OUT} output of timer Y, set bit 4 of port P4 direction register to “1” (output mode).

•When using the T_{OUT} output of timer 1 or timer 2, set bit 1 of port P5 direction register to “1” (output mode).

•The T_{OUT} output pin is shared with the X_{COUT} pin. Accordingly, when using $f(X_{CIN})/2$ as the timer 1 count source (bit 2 of timer 123 mode register = “0”), X_{COUT} oscillation drive must be disabled (bit 5 of clock control register = “1”) to input clocks from the X_{CIN} pin.

•The $P5_1/X_{COUT}/T_{OUT}$ pin cannot function as an ordinary I/O port while X_{CIN}-X_{COUT} is oscillating. When X_{CIN}-X_{COUT} oscillation is stopped or X_{COUT} oscillation drive is disabled, this can be used as the T_{OUT} output pin of timer 1 or 2.

(3) Pulse input

•When using the timer X in the event counter or pulse width measurement mode, set bit 3 of port P4 direction register to “0” (input mode).

•When using the timer Y in the period measurement, event counter or pulse width HL continuously measurement mode, set bit 4 of port P4 direction register to “0” (input mode).

(4) Interrupt

In the timer Y’s pulse width HL continuously measurement mode, $CNTR_1$ interrupt request is generated at both rising and falling edges of $CNTR_1$ pin input signal regardless of the setting of $CNTR_1$ active edge switch bit of timer Y mode register.

2.3 Serial I/O

This paragraph explains the registers setting method and the notes related to the serial I/O.

2.3.1 Memory map

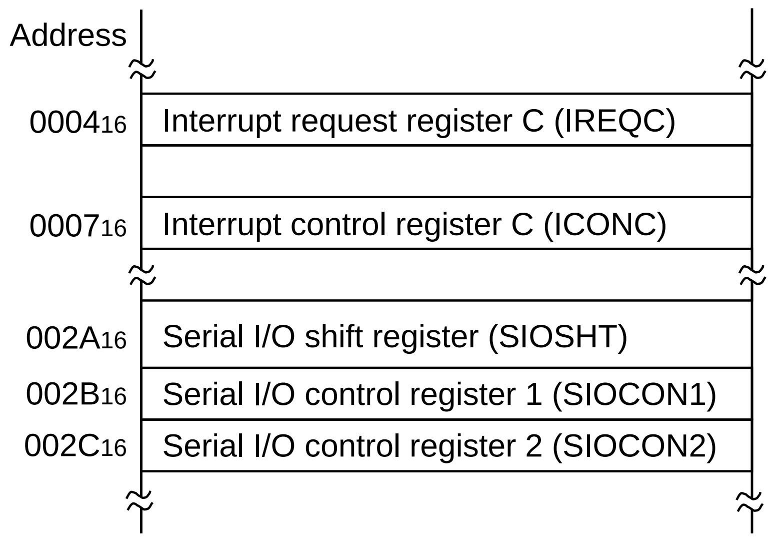


Fig. 2.3.1 Memory map of registers related to serial I/O

2.3.2 Related registers

Serial I/O shift register

b7 b6 b5 b4 b3 b2 b1 b0

Serial I/O shift register
(SIOSHT: address $2A_{16}$)

b	Functions	At reset	R	W
0	●At transmitting	Undefined	O	O
1	Writing transmitted data to this register starts transmitting operation.	Undefined	O	O
2	●At receiving	Undefined	O	O
3	Read received data through this register.	Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7		Undefined	O	O

Fig. 2.3.2 Structure of Serial I/O shift register

Serial I/O control register 1

b7 b6 b5 b4 b3 b2 b1 b0

Serial I/O control register 1
(SIOCON1 : address $2B_{16}$)

b	Name	Functions	At reset	R	W
0	Internal synchronous clock select bits (**Note**)	b2b1b0 0 0 0 : Internal clock divided by 2 0 0 1 : Internal clock divided by 4 0 1 0 : Internal clock divided by 8 0 1 1 : Internal clock divided by 16 1 0 0 : Internal clock divided by 32 1 0 1 : Internal clock divided by 64 1 1 0 : Internal clock divided by 128 1 1 1 : Internal clock divided by 256	0	O	O
1			0	O	O
2			0	O	O
3	Serial I/O port select bit	0 : I/O port 1 : STXD, SCLK signal output	0	O	O
4	$\overline{SRDY}$ output select bit	0 : I/O port 1 : $\overline{SRDY}$ signal output	0	O	O
5	Transfer direction select bit	0 : LSB first 1 : MSB first	0	O	O
6	Synchronous clock select bit	0 : External clock 1 : Internal clock	1	O	O
7	STXD output channel control bit	0 : CMOS output 1 : N-channel open drain output	0	O	O

Note: The source of serial I/O internal synchronous clock can be selected by bit 1 of serial I/O control register 2

Fig. 2.3.3 Structure of Serial I/O control register 1

Serial I/O control register 2

b7 b6 b5 b4 b3 b2 b1 b0: 0 0 0 _ _ _ _ _

Serial I/O control register 2
(SIOCON2 : address $2C_{16}$)

b	Name	Functions	At reset	R	W
0	SPI mode select bit	0 : Normal serial I/O mode 1 : SPI compatible mode (**Note**)	0	O	O
1	Serial I/O internal clock select bit	0 : φ 1 : SCSGCLK	0	O	O
2	SRXD input enable bit	0 : SRXD input disabed 1 : SRXD input enabed	0	O	O
3	Clock polarity select bit (CPoL)	0 : SCLK starting at “L” 1 : SCLK starting at “H”	1	O	O
4	Clock phase select bit (CPha)	0 : Serial transfer starting at falling edge of SRDY 1 : Serial transfer starting after a half cycle of SCLK passed at falling edge of SRDY	1	O	O
5	Fix these bits to “0”.		0	O	O
6			0	O	O
7			0	O	O

Note: To set the slave mode, also set bit 4 of serial I/O control register 1 to “1”.

Fig. 2.3.4 Structure of Serial I/O control register 2

Interrupt request register C

b7 b6 b5 b4 b3 b2 b1 b0: 0 (b7)

Interrupt request register C
(IREQC : address 04_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	CNTR0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	CNTR1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	Serial I/O interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	Input buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	Output buffer empty interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	Key input interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	Fix this bit to “0”.		0	O	O

✻: “0” can be set by software, but “1” cannot be set.

Fig. 2.3.5 Structure of Interrupt request register C

Interrupt control register C

b7 b6 b5 b4 b3 b2 b1 b0: 0 (b7)

Interrupt control register C
(ICONC : address 07_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	CNTR0 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	CNTR1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	Serial I/O interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	Input buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	Output buffer empty interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	Key input interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	Fix this bit to “0”.		0	O	O

Fig. 2.3.6 Structure of Interrupt control register C

2.3.3 Serial I/O connection examples

(1) Control of peripheral IC equipped with CS pin

Figure 2.3.7 shows connection examples of a peripheral IC equipped with the CS pin.

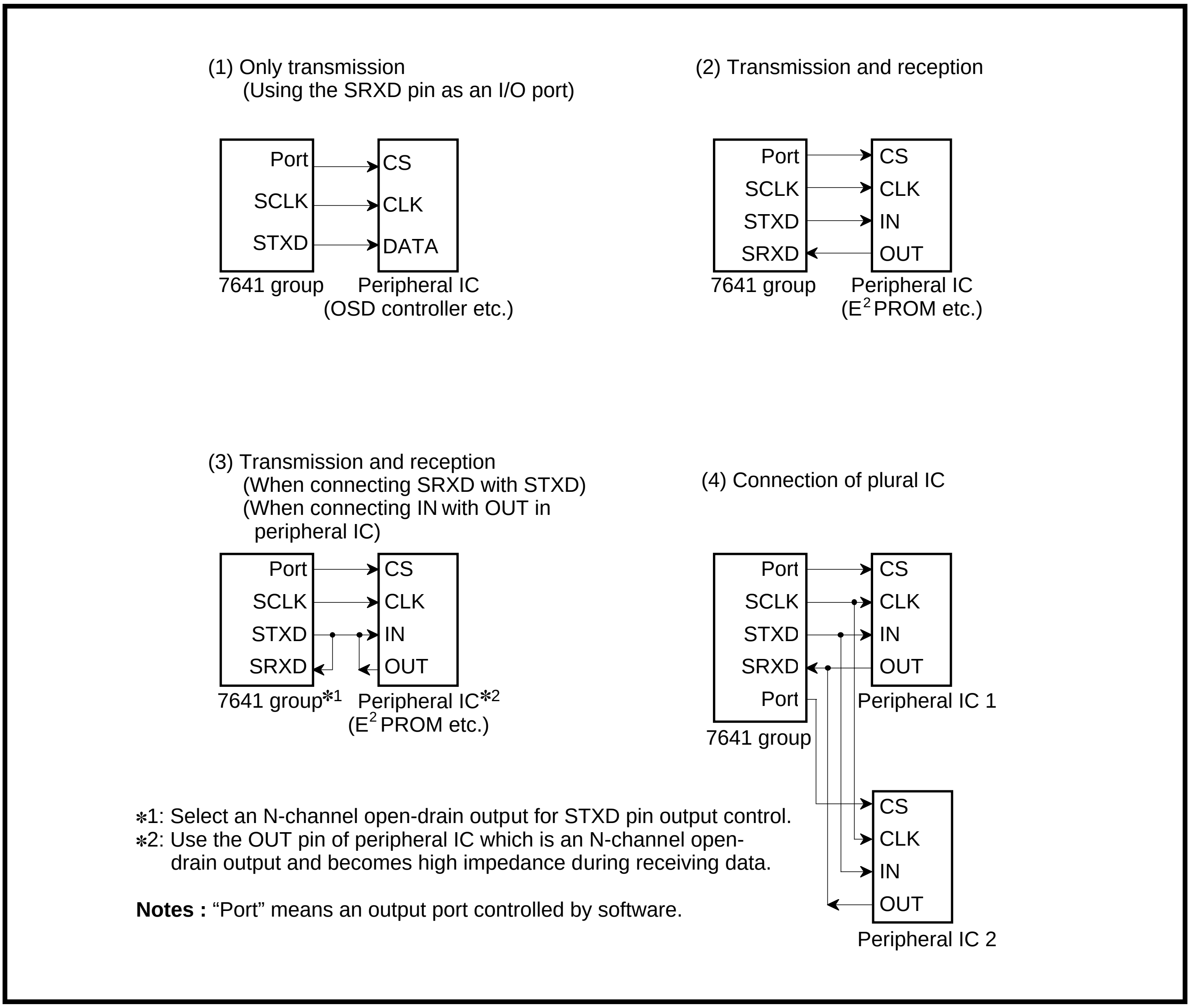


Fig. 2.3.7 Serial I/O connection examples (1)

(2) Connection with microcomputer

Figure 2.3.8 shows connection examples with another microcomputer.

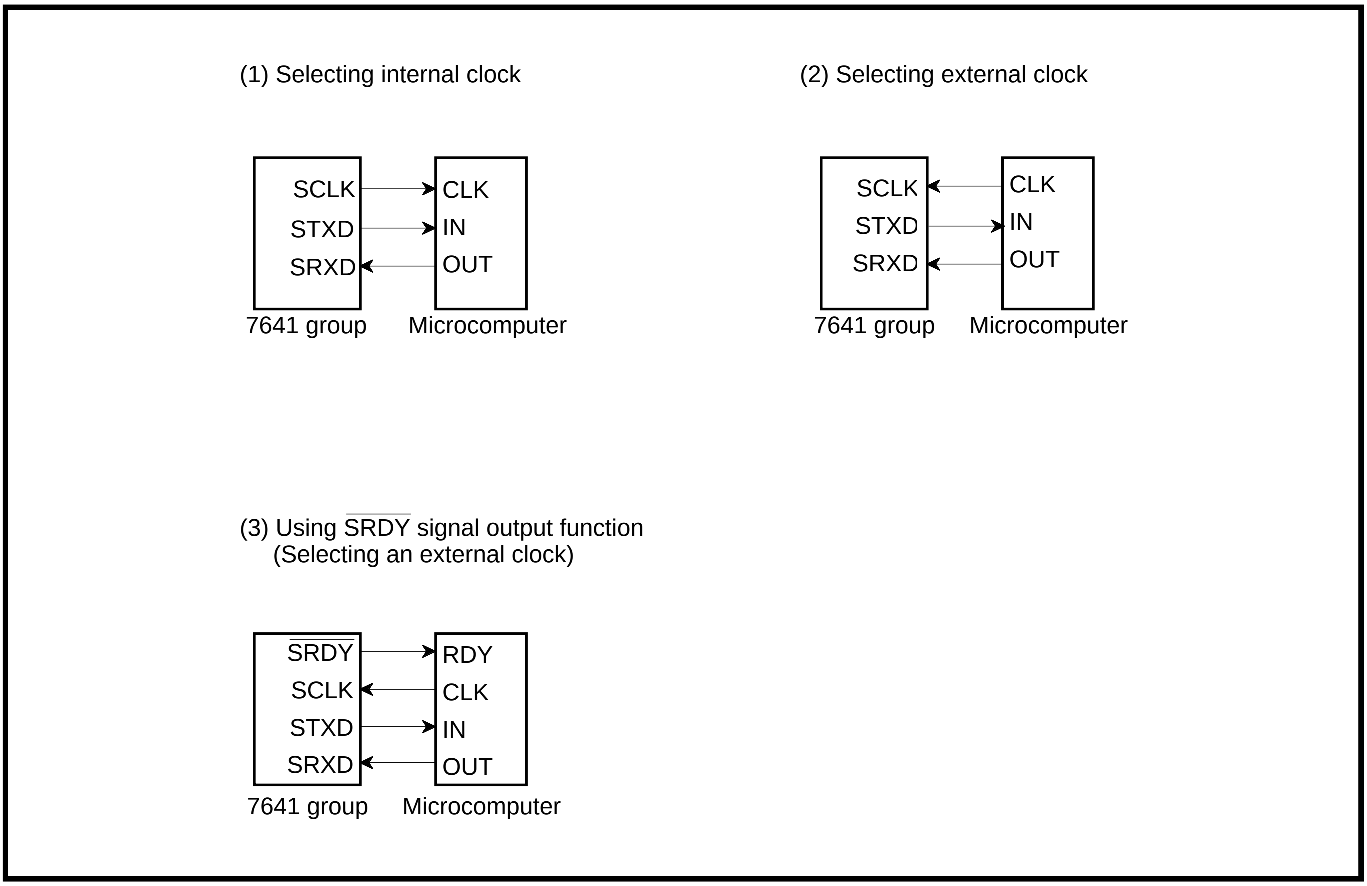


Fig. 2.3.8 Serial I/O connection examples (2)

2.3.4 Serial I/O application example

(1) Output of serial data (control of peripheral IC)

Outline : Serial communication is performed, connecting port to $\overline{CS}$ pin of peripheral IC. To perform reception, it needs to write dummy data into serial I/O shift register.

Figure 2.3.9 shows a connection diagram, and Figure 2.3.10 shows a timing chart.

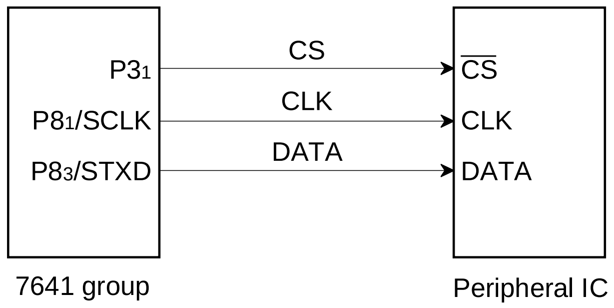


Fig. 2.3.9 Connection diagram

Specifications :
- Synchronous clock frequency : 187.5 kHz ($f(X_{IN})$ = 24 MHz)
- Transfer direction : LSB first
- Serial I/O interrupt is not used.
- Port $P3_1$ is connected to the $\overline{CS}$ pin ("L" active) of the peripheral IC for transmission control; the output level of port $P3_1$ is controlled by software.

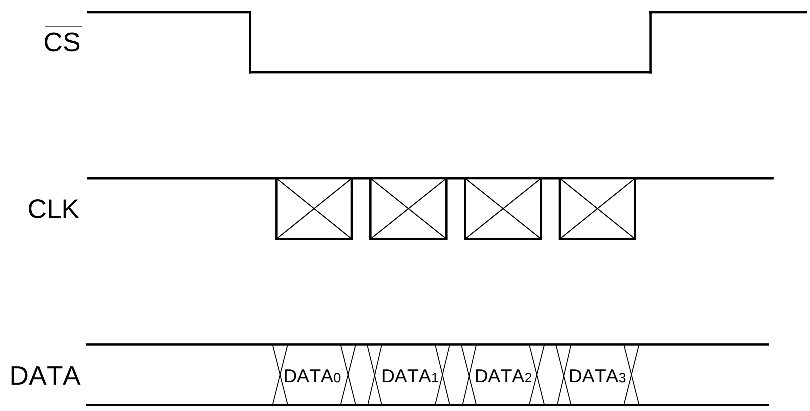


Fig. 2.3.10 Timing chart

Figure 2.3.11 shows the registers setting for the transmitter, and Figure 2.3.12 shows a setting of serial I/O transmission data.

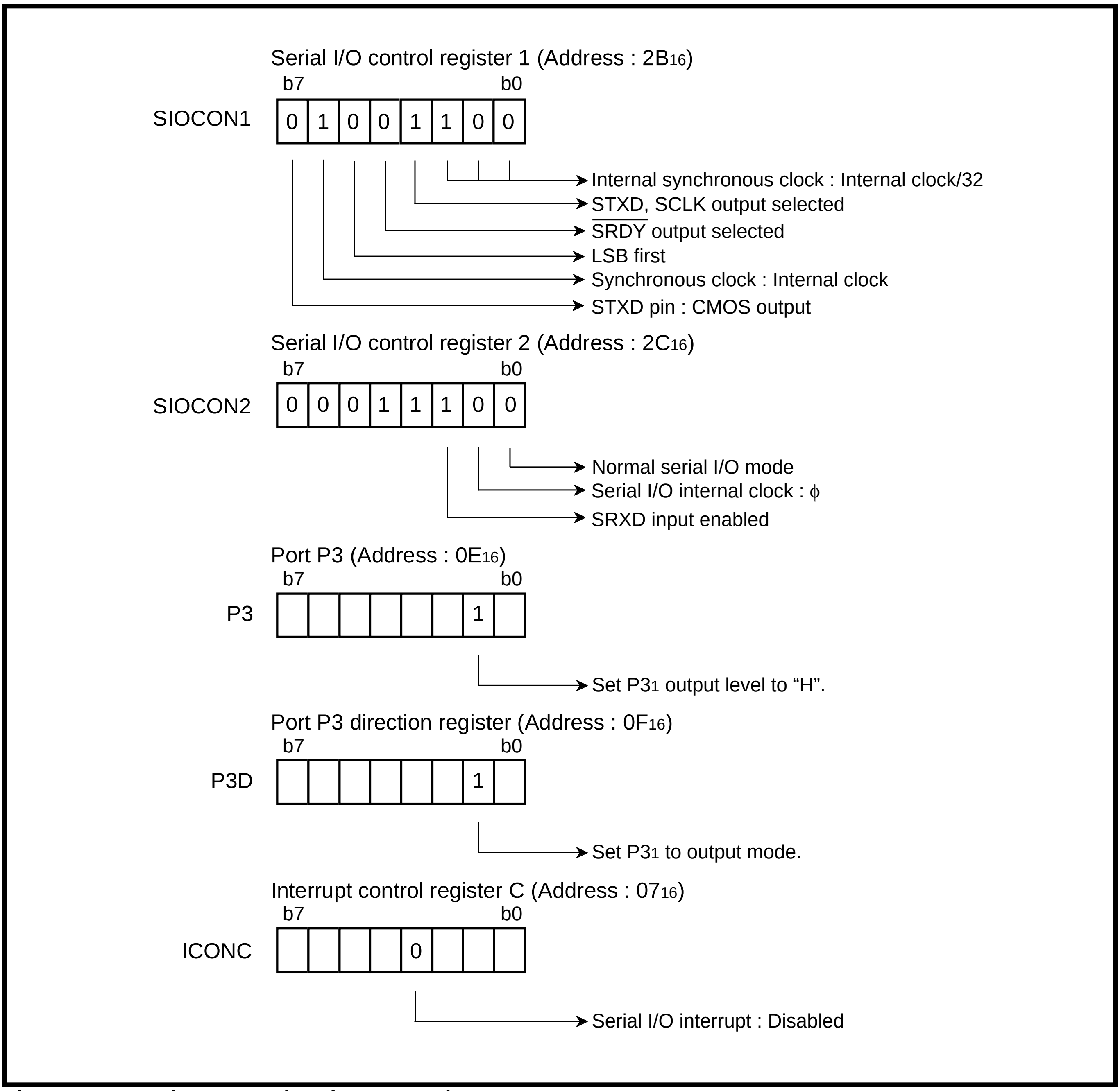


Fig. 2.3.11 Registers setting for transmitter

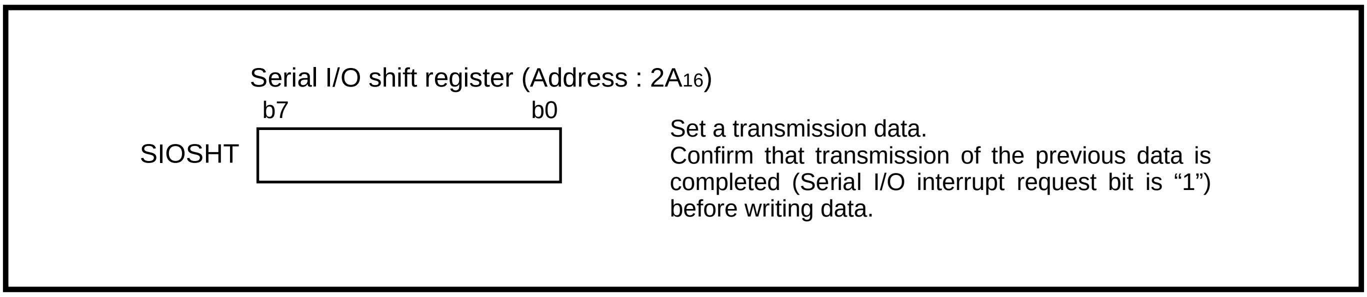


Fig. 2.3.12 Setting of serial I/O transmission data

When the registers are set as shown in Figure 2.3.13, the serial I/O can transmit 1-byte data by writing data into the serial I/O shift register.
Thus, after setting the $\overline{CS}$ signal to “L”, write the transmission data to the serial I/O shift register by each 1 byte, and return the $\overline{CS}$ signal to “H” when all required data have been transmitted.
Figure 2.3.13 shows a control procedure of transmitter.

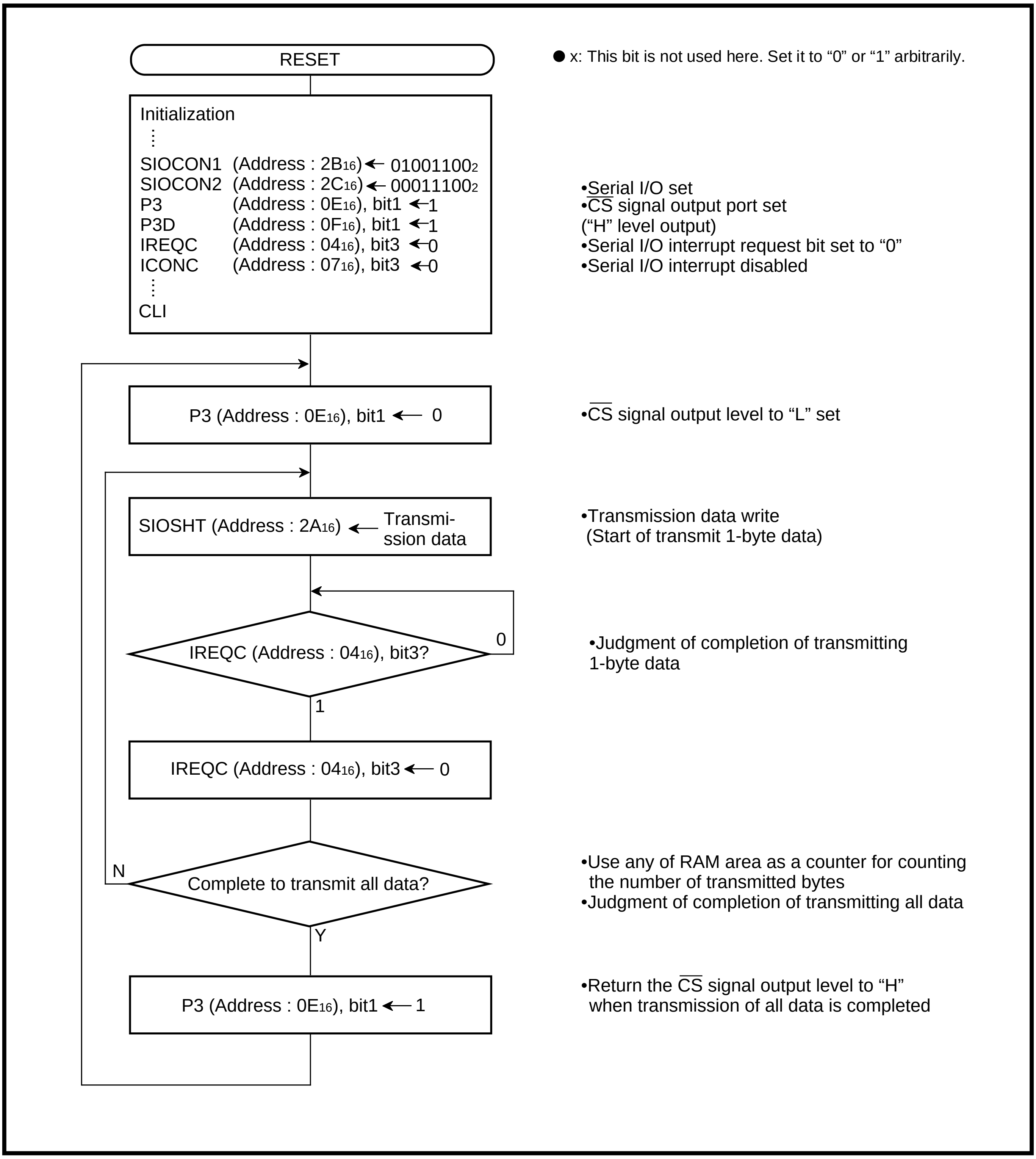


Fig. 2.3.13 Control procedure of transmitter

(2) Serial communication using SPI compatible mode

●Explanation of SPI compatible mode

Setting the SPI mode select bit (bit 0 of SIOCON2) to “1” puts the serial I/O in SPI compatible mode. The synchronous clock select bit (bit 6 of SIOCON1) determines whether the serial I/O is an SPI master or slave. When the external clock is selected (“0”), the serial I/O is in slave mode; when the internal clock is selected (“1”), the serial I/O is in master mode.

In SPI compatible mode the SRXD pin functions as a MISO (Master In/Slave Out) pin and the STXD pin functions as a MOSI (Master Out/Slave In) pin.

In slave mode the transmit data is output from the MISO pin and the receive data is input from the MISO pin. The $\overline{SRDY}$ pin functions as the chip-select signal input pin from an external.

In master mode the transmit data is output from the MOSI pin and the receive data is input from the MISO pin. The $\overline{SRDY}$ pin functions as the chip-select signal output pin to an external.

●Slave mode operation

In slave mode of SPI compatible mode 4 types of clock polarity and clock phase can be usable by bits 3 and 4 of serial I/O control register 2.

If the $\overline{SRDY}$ pin is held “H”, the shift clock is inhibited, the serial I/O counter is set to “7”. If the $\overline{SRDY}$ pin is held “L”, then the shift clock will start.

Make sure during transfer to maintain the $\overline{SRDY}$ input at “L” and not to write data to the serial I/O counter.

Outline : Serial communication is performed between 7641 group MCUs, using SPI compatible mode.

Specifications : • Synchronous clock frequency : 187.5 kHz ($f(X_{IN})$ = 24 MHz)
• Transfer direction : LSB first

Figure 2.3.14 shows a connection diagram; Figure 2.3.15 shows the registers setting for SPI compatible mode; Figures 2.3.16 and 2.3.17 show a control procedure of SPI compatible mode.

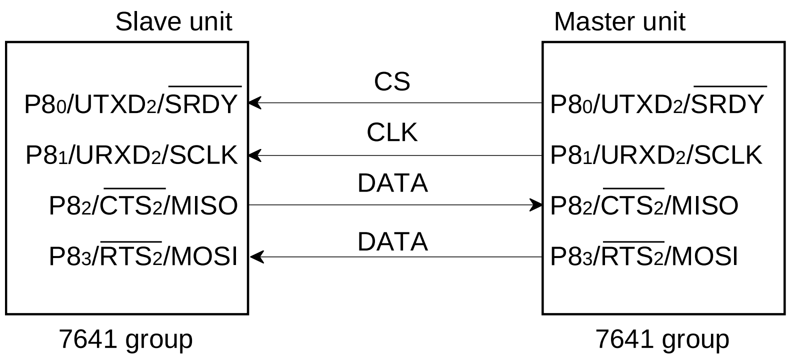


Fig. 2.3.14 Connection diagram

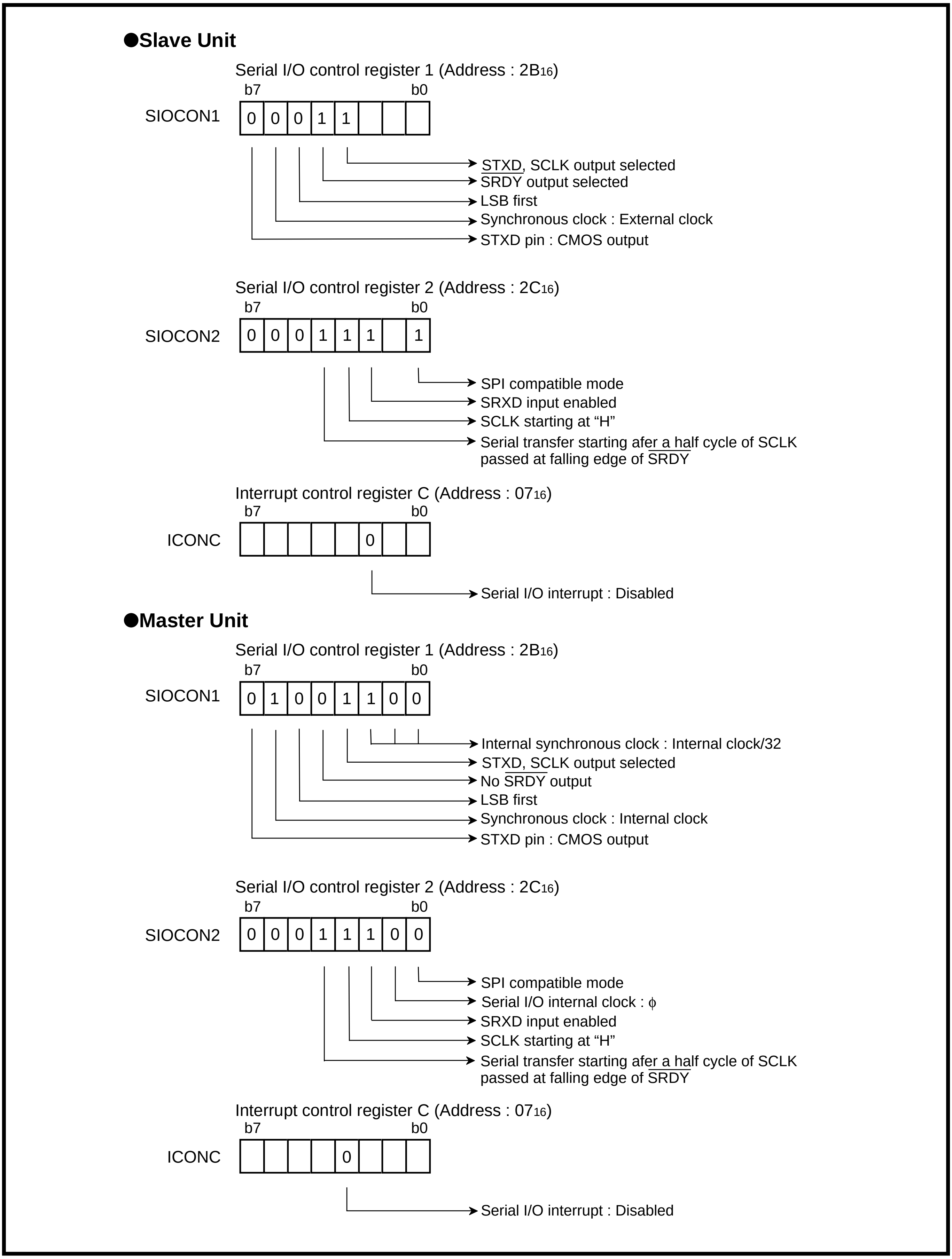


Fig. 2.3.15 Registers setting for SPI compatible mode

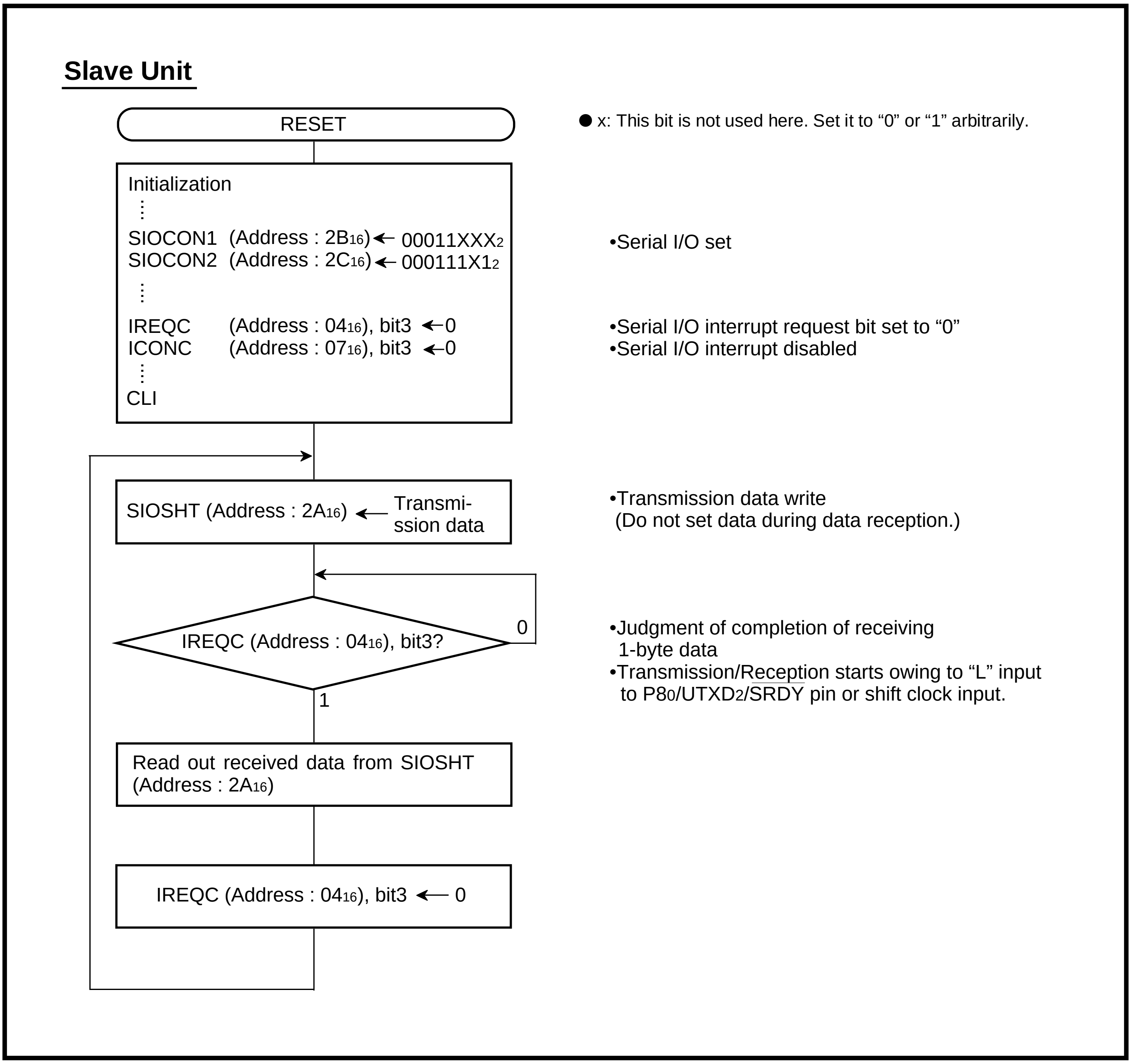


Fig. 2.3.16 Control procedure of SPI compatible mode in slave

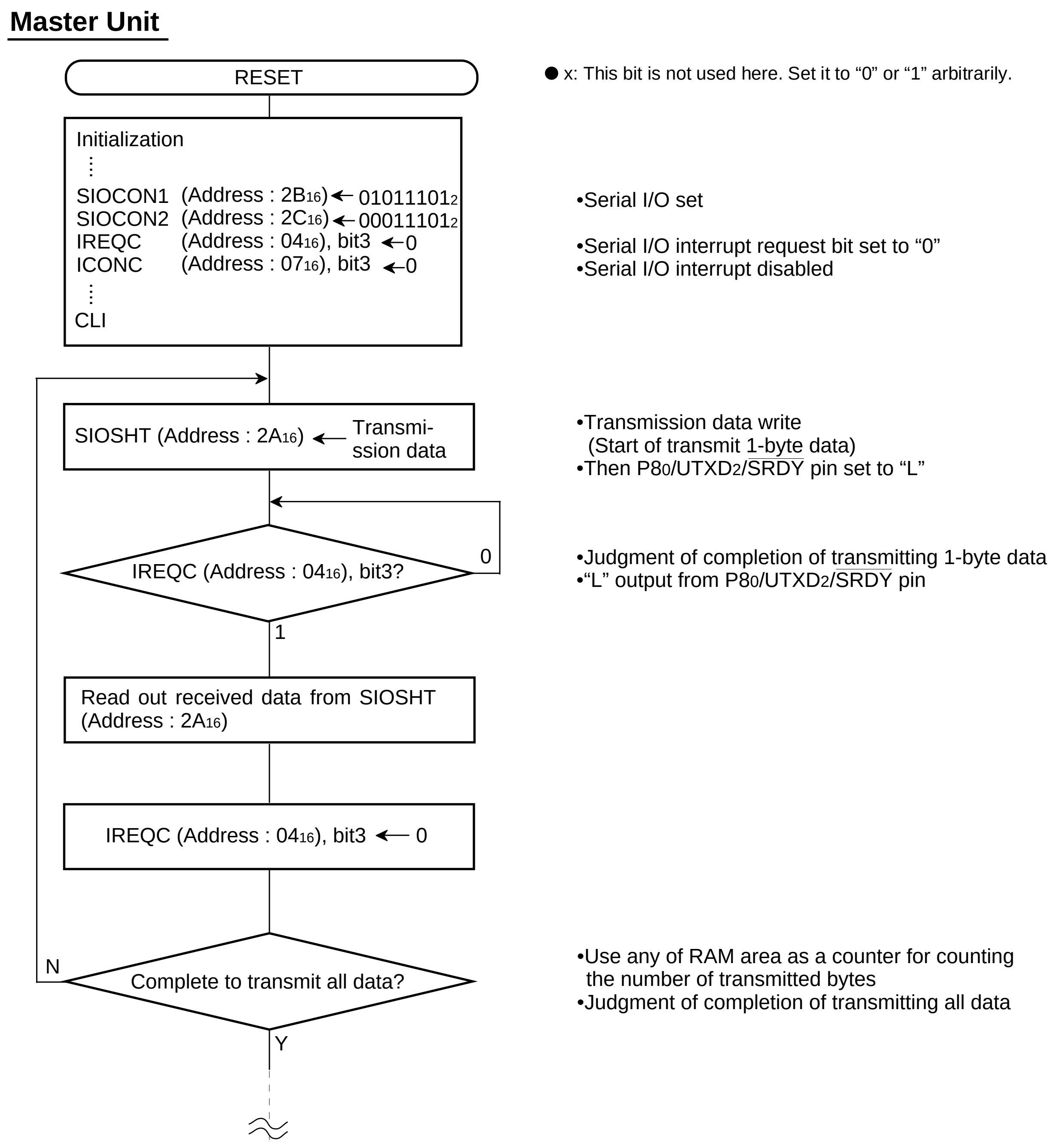


Fig. 2.3.17 Control procedure of SPI compatible mode in master

2.3.5 Notes on serial I/O

(1) Clock

When the external clock is selected as the transfer clock, its transfer clock needs to be controlled by the external source because the serial I/O shift register will keep being shifted while transfer clock is input even after transfer completion.

(2) Reception

When the external clock is selected as the transfer clock for reception, the receiving operation will start owing to the shift clock input even if write operation to the serial I/O shift register (SIOSHT) is not performed. The serial I/O interrupt request also occurs at completion of receiving. However, we recommend to write dummy data in the serial I/O shift register. Because this will cause followings and improve transfer reliability.

•Write to SIOSHT puts the $\overline{SRDY}$ pin to “L”. This enables shift clock output of an external device.
•Write to SIOSHT clears the internal serial I/O counter.

Note: Do not read the serial I/O shift register which is shifting. Because this will cause incorrect-data read.

(3) STXD output

•When the internal clock is selected as the transfer clock, the STXD pin goes a high-impedance state after transfer completion.
•When the external clock is selected as the transfer clock, the STXD pin does not go a high-impedance state after transfer completion.

(4) SPI compatible mode

•When using the SPI compatible mode, set the $\overline{SRDY}$ select bit to “1” ($\overline{SRDY}$ signal output).
•When the external clock is selected in SPI compatible mode, the SRXD pin functions as a data output pin and the STXD pin functions as a data input pin.
•Do not write to the serial I/O shift register (SIOSHT) during a transfer as slave when in SPI compatible mode.
•Master operation of SPI compatible mode requires the timings:
-From write operation to the SIOSHT to $\overline{SRDY}$ pin put to “L”
Requires 2 cycles of internal clock ϕ + 2 cycles of serial I/O synchronous clock + 35 ns
-From $\overline{SRDY}$ pin put to “L” to SCLK switch
Requires 35 ns
-From the last pulse of SCLK to $\overline{SRDY}$ pin put to “H”
Requires 35 ns.

2.4 UART

This paragraph explains the registers setting method and the notes related to the UART.

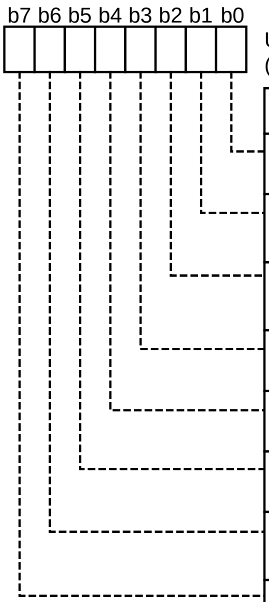
2.4.1 Memory map

Address	
0002_{16}	Interrupt request register A (IREQA)
0003_{16}	Interrupt request register B (IREQB)
0005_{16}	Interrupt control register A (ICONA)
0006_{16}	Interrupt control register B (ICONB)
0030_{16}	UART1 mode register (U1MOD)
0031_{16}	UART1 baud rate generator (U1BRG)
0032_{16}	UART1 status register (U1STS)
0033_{16}	UART1 control register (U1CON)
0034_{16}	UART1 transmit/receive buffer register 1 (U1TRB1)
0035_{16}	UART1 transmit/receive buffer register 2 (U1TRB2)
0036_{16}	UART1 RTS control register (U1RTSC)
0038_{16}	UART2 mode register (U2MOD)
0039_{16}	UART2 baud rate generator (U2BRG)
$003A_{16}$	UART2 status register (U2STS)
$003B_{16}$	UART2 control register (U2CON)
$003C_{16}$	UART2 transmit/receive buffer register 1 (U2TRB1)
$003D_{16}$	UART2 transmit/receive buffer register 2 (U2TRB2)
$003E_{16}$	UART2 RTS control register (U2RTSC)

Fig. 2.4.1 Memory map of registers related to UART

2.4.2 Related registers

UARTx mode register

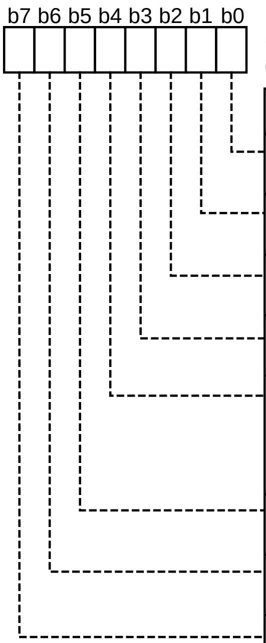


UARTx (x = 1, 2) mode register
(UxMOD : addresses 30_{16}, 38_{16})

b	Name	Functions	At reset	R	W
0	UART clock select bit (CLK)	0 : φ 1 : SCSGCLK output	0	O	O
1	UART clock prescaling select bits (PS)	b2b1 0 0 : UART clock divided by 1 0 1 : UART clock divided by 8 1 0 : UART clock divided by 32 1 1 : UART clock divided by 256	0	O	O
2			0	O	O
3	Stop bit length select bit (STB)	0 : 1 stop bit 1 : 2 stop bits	0	O	O
4	Parity select bit (PMD)	0 : Even parity 1 : Odd parity	0	O	O
5	Parity enable bit (PEN)	0 : Parity checking disabled 1 : Parity checking enabled	0	O	O
6	UART character length select bit (LE1, 0)	b7b6 0 0 : 7 bits 0 1 : 8 bits 1 0 : 9 bits 1 1 : Not available	0	O	O
7			0	O	O

Fig. 2.4.2 Structure of UARTx (x = 1, 2) mode register

UARTx control register



UARTx (x = 1, 2) control register
(UxCON : addresses 33_{16}, $3B_{16}$)

b	Name	Functions	At reset	R	W
0	Transmit enable bit (TEN)	0 : Transmit disabled 1 : Transmit enabled	0	O	O
1	Receive enable bit (REN)	0 : Receive disabled 1 : Receive enabled	0	O	O
2	Transmit initialization bit (TIN)	0 : No action 1 : Initializing (**Note 1**)	0	O	O
3	Receive initialization bit (RIN)	0 : No action 1 : Initializing (**Note 2**)	0	O	O
4	Transmit interrupt source select bit (TIS)	0 : Interrupt when transmit buffer has emptied 1 : Interrupt when transmit shift operation is completed	0	O	O
5	$\overline{CTS}$ function enable bit (CTS_SEL)	0 : $\overline{CTS}$ function disabled (**Note 3**) 1 : CTS function enabled	0	O	O
6	$\overline{RTS}$ function enable bit (RTS_SEL)	0 : $\overline{RTS}$ function disabled (**Note 4**) 1 : $\overline{RTS}$ function enabled	0	O	O
7	UART address mode enable bit (AME)	0 : Address mode disabled 1 : Address mode enabled	0	O	O

Notes 1: When setting the TIN bit to “1”, the TEN bit is set to “0” and the UARTx status register will be set to “03_{16}” after the data has been transmitted. To retransmit, set the TEN bit to “1” and set a data to the transmit buffer register again. The TIN bit will be cleared to “0” one cycle later after the TIN bit has been set to “1”.

2: Setting the RIN bit to “1” suspends the receiving operation and will set all of the REN, RBF and the receive error flags (PER, FER, OER, SER) to “0”. The RIN bit will be cleared to “0” one cycle later after the RIN bit has been set to “1”.

3: When $\overline{CTS}$ function is disabled (CTS_SEl = “0”), pins $P8_2$ and $P8_6$ can be used as ordinary I/O ports.

4: When $\overline{RTS}$ function is disabled (RTS_SEl = “0”), pins $P8_3$ and $P8_7$ can be used as ordinary I/O ports.

Fig. 2.4.3 Structure of UARTx (x = 1, 2) control register

UARTx status register

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) status register
(UxSTS : addresses 32_{16}, $3A_{16}$)

b	Name	Functions	At reset	R	W
0	Transmit complete flag (TCM)	0 : Transmit shift in progress 1 : Transmit shift completed	1	O	×
1	Transmit buffer empty flag (TBE)	0 : Buffer full 1 : Buffer empty	1	O	×
2	Receive buffer full flag (RBF)	0 : Buffer empty 1 : Buffer full	0	O	×
3	Parity error flag (PER)	0 : No error 1 : Parity error	0	O	×
4	Framing error flag (FER)	0 : No error 1 : Framing error	0	O	×
5	Overrun error flag (OER)	0 : No error 1 : Overrun error	0	O	×
6	Summing error flag (SER)	0 : (PER) U (FER) U (OER) = 0 1 : (PER) U (FER) U (OER) = 1	0	O	×
7	Nothing is arranged for this bit. This is a write disable bit. When this bit is read out, the contents are “0”.		0	O	×

Fig. 2.4.4 Structure of UARTx (x = 1, 2) status register

UARTx RTS control register

b7 b6 b5 b4 b3 b2 b1 b0

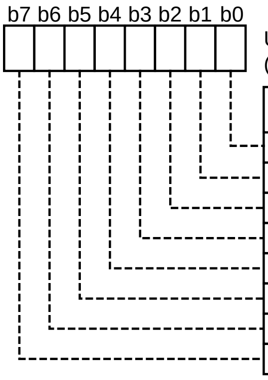
0 0 0 0 (b3 to b0)

UARTx (x = 1, 2) RTS control register
(UxRTSC : addresses 36_{16}, $3E_{16}$)

b	Name	Functions	At reset	R	W
0	Fix these bits to “0”.		0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4	$\overline{RTS}$ assertion delay count select bits (RTS)	b7b6b5b4 0 0 0 0 : No delay; Assertion immediately 0 0 0 1 : 8-bit term assertion at “H” 0 0 1 0 : 16-bit term assertion at “H” 0 0 1 1 : 24-bit term assertion at “H” 0 1 0 0 : 32-bit term assertion at “H” 0 1 0 1 : 40-bit term assertion at “H” 0 1 1 0 : 48-bit term assertion at “H” 0 1 1 1 : 56-bit term assertion at “H” 1 0 0 0 : 64-bit term assertion at “H” 1 0 0 1 : 72-bit term assertion at “H” 1 0 1 0 : 80-bit term assertion at “H” 1 0 1 1 : 88-bit term assertion at “H” 1 1 0 0 : 96-bit term assertion at “H” 1 1 0 1 : 104-bit term assertion at “H” 1 1 1 0 : 112-bit term assertion at “H” 1 1 1 1 : 120-bit term assertion at “H”	0	O	O
5			0	O	O
6			0	O	O
7			1	O	O

Fig. 2.4.5 Structure of UARTx (x = 1, 2) RTS control register

UARTx baud rate generator



UARTx (x = 1, 2) baud rate generator
(UxBRG: addresses 31_{16}, 39_{16})

b	Functions	At reset	R	W
0	●The UxBRG determines the baud rate for transfer. ●This is a 8-bit counter with its reload register. This generator divides the frequency of the count source by 1/(n + 1), where "n" is the value written to the UxBRG.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7		Undefined	O	O

Fig. 2.4.6 Structure of UARTx (x = 1, 2) baud rate generator

UARTx transmit/receive buffer registers 1, 2

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) transmit/receive buffer register 1
(UxTRB1: addresses 34_{16}, $3C_{16}$)

b	Functions	At reset	R	W
0	The transmit buffer register and the receive buffer register are located at the same address. Writing a transmitting data and reading a received data are performed through the UxTRB. This is its low-order byte. •At write The data is written into the transmit buffer register. It is not done into the receive buffer register. •At read The contents of receive buffer register is read. If a character bit length is 7 bits, the MSB of received data is invalid.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7	Note that the contents of transmit buffer register cannot be read.	Undefined	O	O

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) transmit/receive buffer register 2
(UxTRB2: addresses 35_{16}, $3D_{16}$)

b	Functions	At reset	R	W
0	The transmit buffer register and the receive buffer register are located at the same address. Writing a transmitting data and reading a received data are performed through the UxTRB. This is its high-order byte. •At write The data is written into the transmit buffer register. It is not done into the receive buffer register. •At read The contents of receive buffer register is read. If a character bit length is 9 bits, the received high-order 7 bits of UxTRB2 are "0" Note that the contents of transmit buffer register cannot be read. If a character bit length is 7 or 8 bits, the received contents of UxTRB2 are invalid.	Undefined	O	O
1	Nothing is arranged for this bit. This is a write disable bit. When this bit is read out, the contents are "0".	Undefined	O	X
2		Undefined	O	X
3		Undefined	O	X
4		Undefined	O	X
5		Undefined	O	X
6		Undefined	O	X
7		Undefined	O	X

Fig. 2.4.7 Structure of UARTx (x = 1, 2) transmit/receive buffer registers 1, 2

Interrupt request register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register A
(IREQA : address 02_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	USB SOF interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	INT$_0$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	INT$_1$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	DMAC0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	DMAC1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	UART1 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	UART1 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻

✻: "0" can be set by software, but "1" cannot be set.

Fig. 2.4.8 Structure of Interrupt request register A

Interrupt request register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register B
(IREQB : address 03_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	UART2 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	UART2 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	UART2 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	Timer X interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	Timer Y interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	Timer 1 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	Timer 2 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻

✻: "0" can be set by software, but "1" cannot be set.

Fig. 2.4.9 Structure of Interrupt request register B

Interrupt control register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register A
(ICONA : address 05_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	USB SOF interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	INT$_0$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	INT$_1$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	DMAC0 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	DMAC1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	UART1 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	UART1 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O

Fig. 2.4.10 Structure of Interrupt control register A

Interrupt control register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register B
(ICONB : address 06_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	UART2 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	UART2 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	UART2 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	Timer X interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	Timer Y interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	Timer 1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	Timer 2 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O

Fig. 2.4.11 Structure of Interrupt control register B

2.4.3 UART transfer data format

Figure 2.4.12 shows the UART transfer data format.

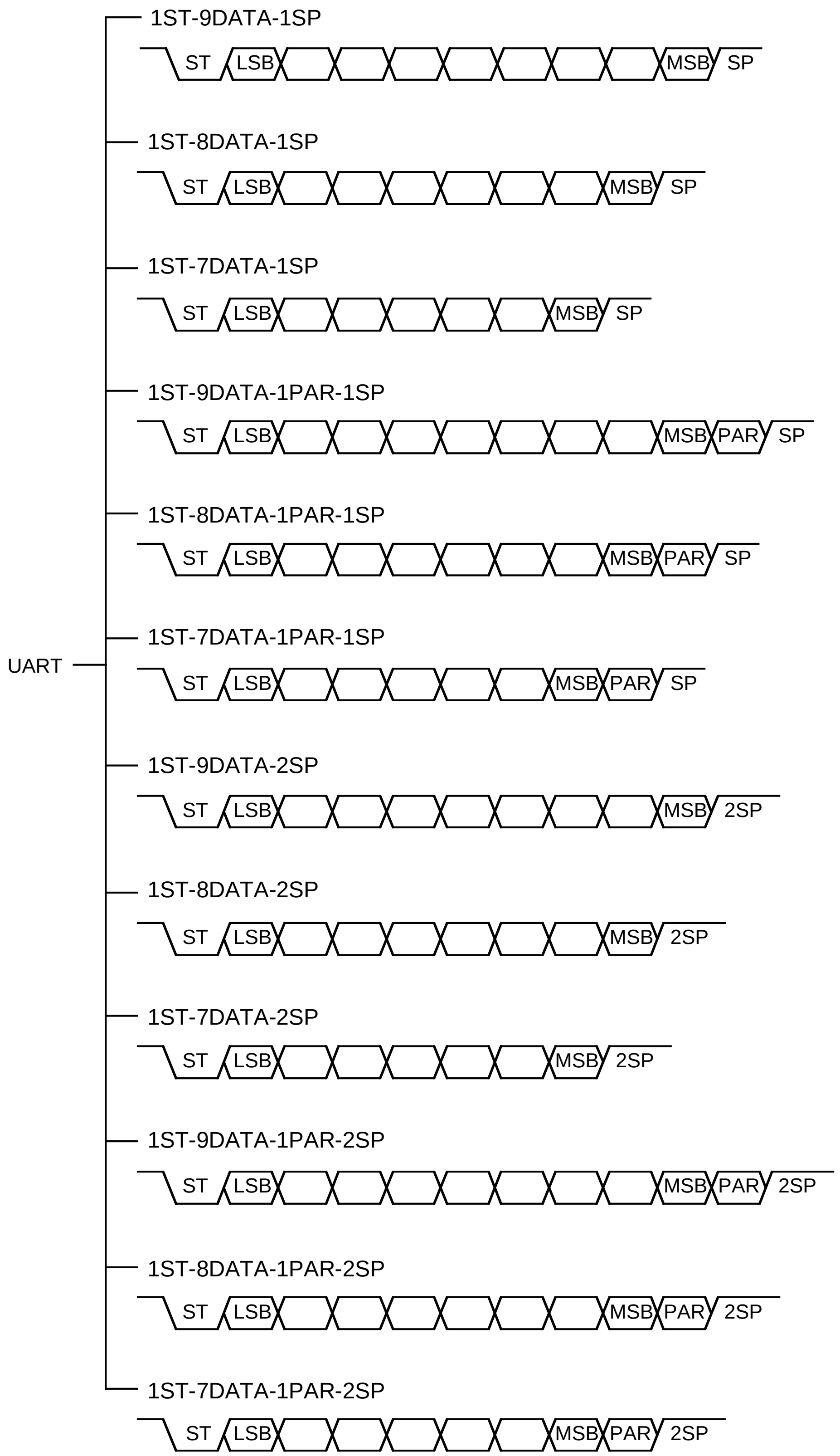


Fig. 2.4.12 UART transfer data format

2.4.4 Transfer bit rate

UART1 and UART2 can use either internal clock φ or SCSGCLK (Special Count Source Generator output) as its clock.

(1) Setting examples using internal clock φ

Table 2.4.1 shows setting examples of the baud rate generator (BRG) values and transfer bit rate values.

Table 2.4.1 Setting examples of baud rate generator values and transfer bit rate values (φ = 12 MHz))

φ/1 (**Note 1**)		φ/8 (**Note 1**)		φ/32 (**Note 1**)		φ/32 (**Note 1**)	
BRG setting value	Transfer bit rate (bps)	BRG setting value	Transfer bit rate (bps)	BRG setting value	Transfer bit rate (bps)	BRG setting value	Transfer bit rate (bps)
00 (00$_{16}$)	750,000.0						
01 (01$_{16}$)	375,000.0						
02 (02$_{16}$)	250,000.0						
03 (03$_{16}$)	187,500.0						
04 (04$_{16}$)	150,000.0						
05 (05$_{16}$)	125,000.0						
06 (06$_{16}$)	107,142.9						
07 (07$_{16}$)	93,750.0	00 (00$_{16}$)	93,750.0				
08 (08$_{16}$)	83,333.3	01 (01$_{16}$)	46,875.0				
09 (09$_{16}$)	75,000.0	02 (02$_{16}$)	31,250.0				
10 (0A$_{16}$)	68,181.8	03 (03$_{16}$)	23,437.5	00 (00$_{16}$)	23,437.5		
11 (0B$_{16}$)	65,250.0	04 (04$_{16}$)	18,750.0	01 (01$_{16}$)	11,718.7		
12 (0C$_{16}$)	57,692.3	05 (05$_{16}$)	15,625.0	02 (02$_{16}$)	7,812.5		
13 (0D$_{16}$)	53,571.4	06 (06$_{16}$)	13,392.8	03 (03$_{16}$)	5,859.4		
14 (0E$_{16}$)	50,000.0	07 (07$_{16}$)	11,718.7	04 (04$_{16}$)	4,687.5		
15 (0F$_{16}$)	46,875.0	08 (08$_{16}$)	10,416.6	05 (05$_{16}$)	3,906.3		
⋮	⋮	09 (09$_{16}$)	9,375.0	06 (06$_{16}$)	3,348.2		
⋮	⋮	10 (0A$_{16}$)	8,522.7	07 (07$_{16}$)	2,929.7	00 (00$_{16}$)	2,929.7
⋮	⋮	11 (0B$_{16}$)	7,812.5	08 (08$_{16}$)	2,604.2	01 (01$_{16}$)	1,464.8
⋮	⋮	12 (0C$_{16}$)	7,211.5	09 (09$_{16}$)	2,343.7	02 (02$_{16}$)	976.6
⋮	⋮	13 (0D$_{16}$)	6,696.4	10 (0A$_{16}$)	2,130.6	03 (03$_{16}$)	732.4
⋮	⋮	14 (0E$_{16}$)	6,250.0	11 (0B$_{16}$)	1,953.1	04 (04$_{16}$)	585.9
⋮	⋮	15 (0F$_{16}$)	5,859.3	12 (0C$_{16}$)	1,802.8	05 (05$_{16}$)	488.3
⋮	⋮	⋮	⋮	13 (0D$_{16}$)	1,674.1	06 (06$_{16}$)	418.5
⋮	⋮	⋮	⋮	14 (0E$_{16}$)	1,562.5	07 (07$_{16}$)	366.2
⋮	⋮	⋮	⋮	15 (0F$_{16}$)	1,464.8	08 (08$_{16}$)	325.5
⋮	⋮	⋮	⋮	⋮	⋮	09 (09$_{16}$)	292.9
⋮	⋮	⋮	⋮	⋮	⋮	10 (0A$_{16}$)	266.3
⋮	⋮	⋮	⋮	⋮	⋮	11 (0B$_{16}$)	244.1
⋮	⋮	⋮	⋮	⋮	⋮	12 (0C$_{16}$)	225.3
⋮	⋮	⋮	⋮	⋮	⋮	13 (0D$_{16}$)	209.2
⋮	⋮	⋮	⋮	⋮	⋮	14 (0E$_{16}$)	195.3
⋮	⋮	⋮	⋮	⋮	⋮	15 (0F$_{16}$)	183.1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
253 (FD$_{16}$)	2,952.7	253 (FD$_{16}$)	369.0	253 (FD$_{16}$)	92.2	253 (FD$_{16}$)	11.5
254 (FE$_{16}$)	2,941.1	254 (FE$_{16}$)	367.6	254 (FE$_{16}$)	91.9	254 (FE$_{16}$)	11.4
255 (FF$_{16}$)	2,929.7	255 (FF$_{16}$)	366.2	255 (FF$_{16}$)	91.6	255 (FF$_{16}$)	11.4

Notes 1: Select the UART clock prescaling with bits 1 and 2 of UARTx mode register.
2: Equation of transfer bit rate:

$$\text{Transfer bit rate (bps)} = \frac{fi^{*}}{(\text{BRG setting value} + 1) \times 16}$$

✻ : fi is selectable among ϕ/1, ϕ/8, ϕ/32, and ϕ/256 with bits 1 and 2 of UARTx mode register.

(2) Setting examples using SCSGCLK output

Table 2.4.2 shows setting examples of the SCSG1, SCSG2 and the baud rate generator (BRG) values and transfer bit rate values.

Table 2.4.2 Setting examples of SCSG1, SCSG2 and baud rate generator values and transfer bit rate values (ϕ = 12 MHz))

Transfer bit rate (bps) (**Note 1**)	Special Count Source Generator			UART clock prescaling (**Note 2**)	BRG setting value	Real rate (bps)
	SCSG1 setting value (**Note 3**)	SCSG2 setting value	SCSGCLK (Hz) (**Note 4**)			
50	bypassed	149 (95_{16})	80000.00	1/1	99 (63_{16})	50.00
75	bypassed	99 (63_{16})	120000.00	1/1	99 (63_{16})	75.00
110	78 ($4E_{16}$)	67 (43_{16})	174236.78	1/1	98 (62_{16})	110.00
134.5	172 (AC_{16})	55 (37_{16})	213047.07	1/1	98 (62_{16})	134.50
150	bypassed	49 (31_{16})	240000.00	1/1	99 (63_{16})	150.00
300	bypassed	24 (18_{16})	480000.00	1/1	99 (63_{16})	300.00
600	24 (18_{16})	11 ($0B_{16}$)	960000.00	1/1	99 (63_{16})	600.00
1200	bypassed	24 (18_{16})	480000.00	1/1	24 (18_{16})	1200.00
1800	24 (18_{16})	19 (13_{16})	576000.00	1/1	19 (13_{16})	1800.00
2000	bypassed	24 (18_{16})	480000.00	1/1	14 ($0E_{6}$)	2000.00
2400	24 (18_{16})	19 (13_{16})	576000.00	1/1	14 ($0E_{6}$)	2400.00
3600	24 (18_{16})	19 (13_{16})	576000.00	1/1	9 (09_{16})	3600.00
4800	24 (18_{16})	14 ($0E_{6}$)	768000.00	1/1	9 (09_{16})	4800.00
7200	24 (18_{16})	19 (13_{16})	576000.00	1/1	4 (04_{16})	7200.00
9600	24 (18_{16})	14 ($0E_{6}$)	768000.00	1/1	4 (04_{16})	9600.00
14400	24 (18_{16})	09 (09_{16})	1152000.00	1/1	4 (04_{16})	14400.00
19200	35 (23_{16})	00 (00_{16})	11666666.66	1/1	37 (25_{16})	19188.60
28800	24 (18_{16})	00 (00_{16})	11520000.00	1/1	24 (18_{16})	28800.00
31250	bypassed	bypassed	12000000.00	1/1	23 (17_{16})	31250.00
38400	35 (23_{16})	00 (00_{16})	11666666.66	1/1	18 (12_{16})	38377.19
57600	bypassed	bypassed	12000000.00	1/1	12 ($0C_{16}$)	57692.31
115200	11 ($0B_{16}$)	00 (00_{16})	11000000.00	1/1	5 (05_{16})	114583.33

Notes 1: Equation of transfer bit rate:

$$\text{Transfer bit rate (bps)} = \frac{fi^{*}}{(\text{BRG setting value} + 1) \times 16}$$

* : fi is selectable among SCSGCLK/1, SCSGCLK/8, SCSGCLK/32, and SCSGCLK/256 with bits 1 and 2 of UARTx mode register.

2: Select the UART clock prescaling with bits 1 and 2 of UARTx mode register.

3: The internal clock ϕ is used as the SCSG2 count source when the SCSG1 count stop bit is “1” or the SCSG1 timer set value is “00_{16}”.

4: Equation of SCSGCLK frequency:

$$\text{SCSGCLK (MHz)} = \phi \times \frac{\text{SCSG1 setting value}}{(\text{SCSG1 setting value} + 1)} \times \frac{1}{(\text{SCSG2 setting value} + 1)}$$

2.4.5 Operation of transmitting and receiving

(1) Transmit operation

- The transmit buffer empty flag (TBE) is set to "0" when the low-order byte of transmitted data is written into the UARTx (x = 1, 2) transmit buffer register 1 in the condition of transmission enabled. When using 9-bit character length, set the data into the UARTx transmit buffer register 2 (high-order byte) first before the UARTx transmit buffer register 1 (low-order byte).
- If the transmit shift register is empty in the condition of $\overline{\mathrm{CTS}}$ function disabled, the transmitted data which is written into the UARTx (x = 1, 2) transmit buffer register 1 will be transferred to the transmit shift register at the same time. When the TBE flag becomes "1", the following data can be set to the UARTx (x = 1, 2) transmit buffer. At this point, the UART transmit interrupt request occurs when the transmit interrupt source select bit (TIS) is "0".
- When the $\overline{\mathrm{CTS}}$ function is enabled, the transmitted data is not transferred to the transmit shift register until "L" is input to the $\overline{\mathrm{CTSx}}$ pin ($P8_6$/$\overline{\mathrm{CTS}_1}$, $P8_2$/$\overline{\mathrm{CTS}_2}$/SRXD).
- The data is transmitted with the LSB first format. Once the transmission starts, it continues until the last bit has been transmitted even though clearing the transmit enable bit (TEN) to "0" (disabled) or inputting "H" to the $\overline{\mathrm{CTSx}}$ pin.
- After completion of the last bit transmitting, if the TBE flag is "1", or the TEN bit is "0" (disabled) or "H" is input to the $\overline{\mathrm{CTSx}}$ pin, the transmit complete flag (TCM) goes to "1". At this point, the UARTx transmit interrupt request occurs when the TIS bit is "1".

(2) Receive operation

- The data is received with the LSB first format in the condition of reception enabled.
- When the stop bit is detected, the received data is transferred from the receive shift register to the UARTx (x = 1, 2) receive buffer register. At the same time, if there is no error, the receive buffer full flag (RBF) is set to "1" and the UARTx receive buffer full interrupt request occurs.
- If receive errors occur, the corresponding error flags of UARTx status register are set to "1" and the UARTx summing error interrupt request occurs.
- The receive buffer full flag (RBF) is set to "0" when the contents of UARTx receive buffer register 1 is read out. Then when the $\overline{\mathrm{RTS}}$ function is disabled, the following data can be received.
  When using 9-bit character length, read the data from the UARTx receive buffer register 2 (high-order byte) first before the UARTx receive buffer register 1 (low-order byte).
- When the $\overline{\mathrm{RTS}}$ function is enabled, the $\overline{\mathrm{RTS}}$ assertion delay count is specified by the UARTx RTS control register. The delay time from the reception of the last stop bit to the start bit is selectable. The $\overline{\mathrm{RTSx}}$ pin ($P8_7$/$\overline{\mathrm{RTS}_1}$, $P8_3$/$\overline{\mathrm{RTS}_2}$/STXD) outputs "H" during the delayed time. After that, the $\overline{\mathrm{RTSx}}$ pin outputs "L" and a reception is enabled.
- If the start bit is detected in the term of "H" assertion of $\overline{\mathrm{RTS}}$, its assertion count is suspended and the $\overline{\mathrm{RTSx}}$ pin remains "H" output. After receiving the last stop bit, the count is resumed.

(3) Countermeasure for errors

Three errors can be detected at reception. Each error is detected simultaneously when the data is transferred from the receive shift register to the receive buffer register. If receive errors occur, the corresponding error flags of UARTx status register are set to "1". When any one of errors occurs, the summing error flag is set to "1" and the UARTx summing error interrupt request bit is also set to "1". If a receive error occurs, the reception does not set the UARTx receive buffer full interrupt request bit to "1".

If receive errors occur, initialize the error flags and the UARTx receive buffer register and then retransmit the data.

Table 2.4.3 shows the error flags set condition and how to clear error flags.

Table 2.4.3 Error flags set condition and how to clear error flags

Error flag	Error flag set condition	How to clear error flag
Overrun flag (OER)	•If the previous data in the receive buffer register is not read before the current receive operation is completed. •If any one of error flags is "1" for the previous data and the current receive operation is completed.	•Reading UARTx status register •Hardware reset •Setting the receive initialization bit (RIN) to "1"
Framing error flag (FER)	•When the number of stop bit of the received data does not correspond with the selection with the stop bit length select bit (STB).	
Parity error flag (PER)	•When the sum total of 1s of received data and the parity does not correspond with the selection with the parity select bit (PMD).	
Summing error flag (SER)	•When any one of the PER, FER and OER is set to "1".	

2.4.6 UART application example

(1) Data output (control of peripheral IC)

Outline : Data is transmitted and received, using the UART.

Figure 2.4.13 shows a connection diagram, and Figure 2.4.14 shows a timing chart.

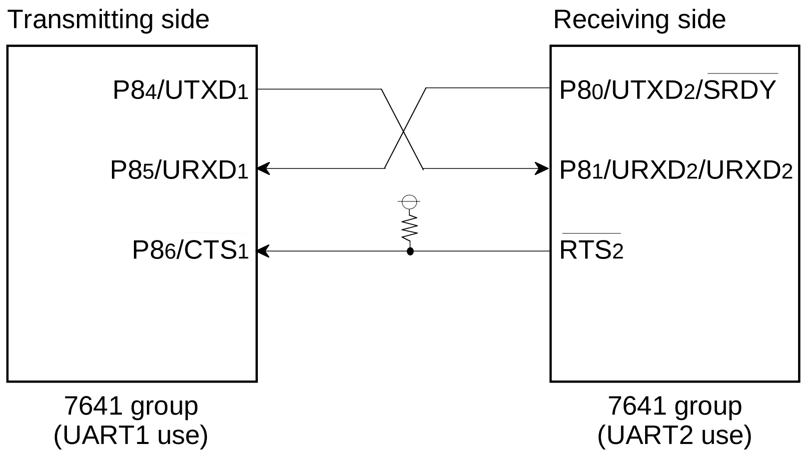


Fig. 2.4.13 Connection diagram

Specifications :
- Transmitter: UART1 is used.
- Receiver: UART2 is used.
- Transfer bit rate : 9600 bps (ϕ = f(X_{IN})/4 = 4 MHz divided by 416)
- Data format: 1ST-9DATA-2ST
- Use of $\overline{CTS}$ and $\overline{RTS}$ functions
- 2-byte data is transferred from the transmitting side to the receiving side at intervals of 10 ms generated by the timer.

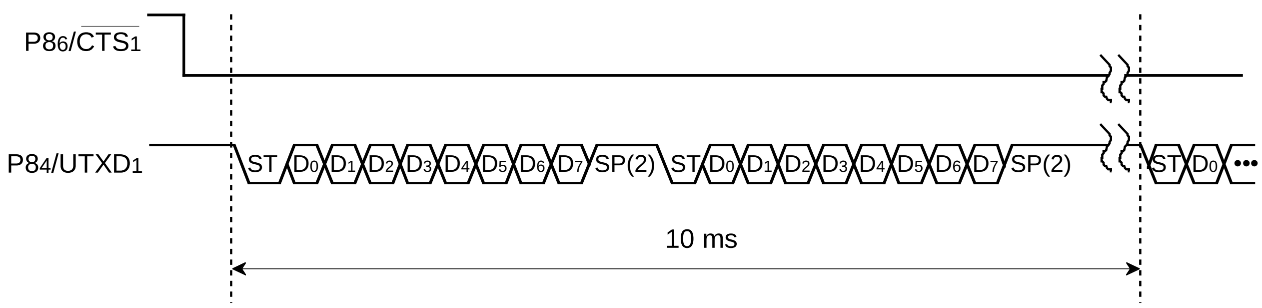


Fig. 2.4.14 Timing chart

Figure 2.4.15 shows the registers setting for the transmitter, and Figures 2.4.16 and 2.4.17 show the registers setting for the receiver.

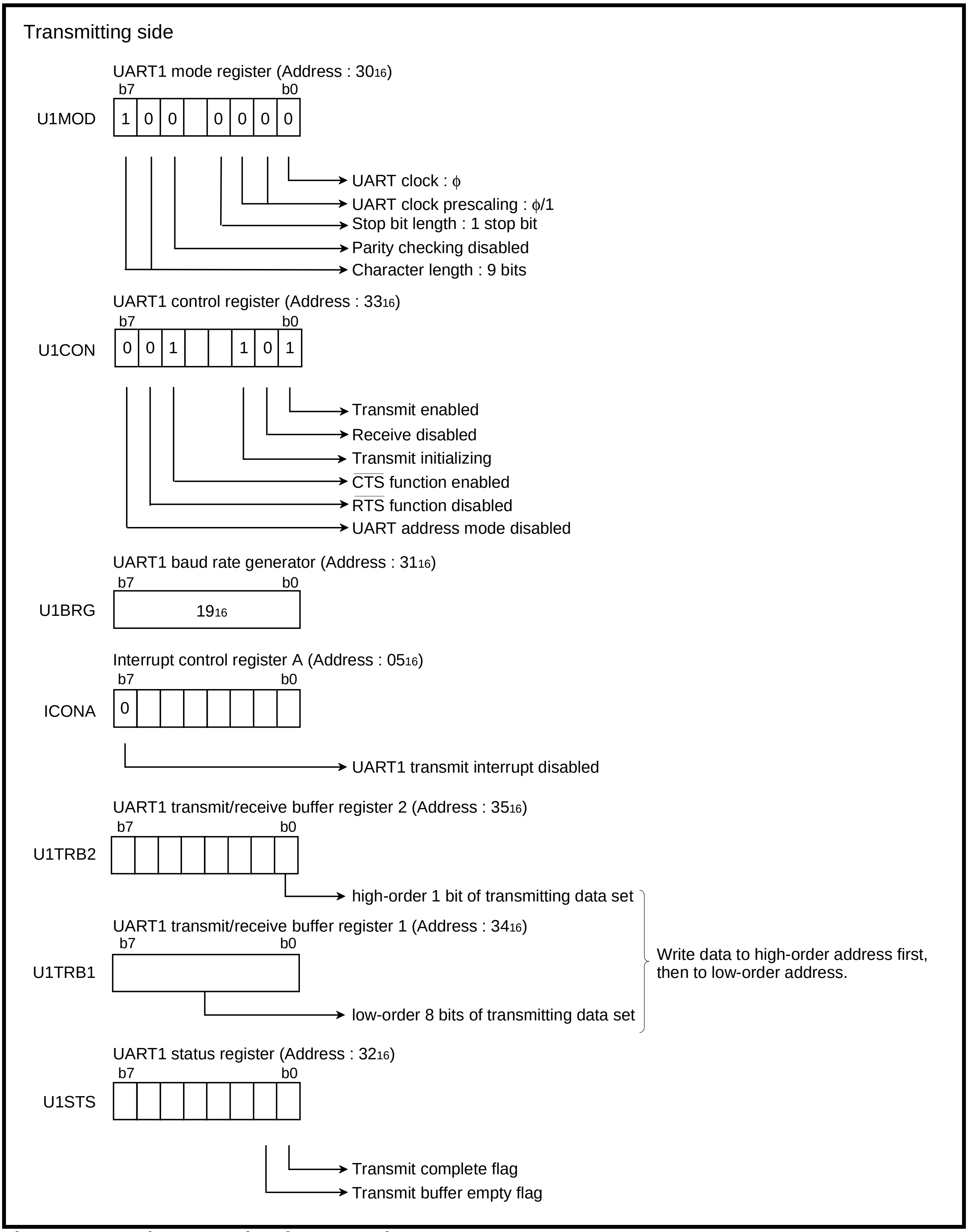


Fig. 2.4.15 Registers setting for transmitter

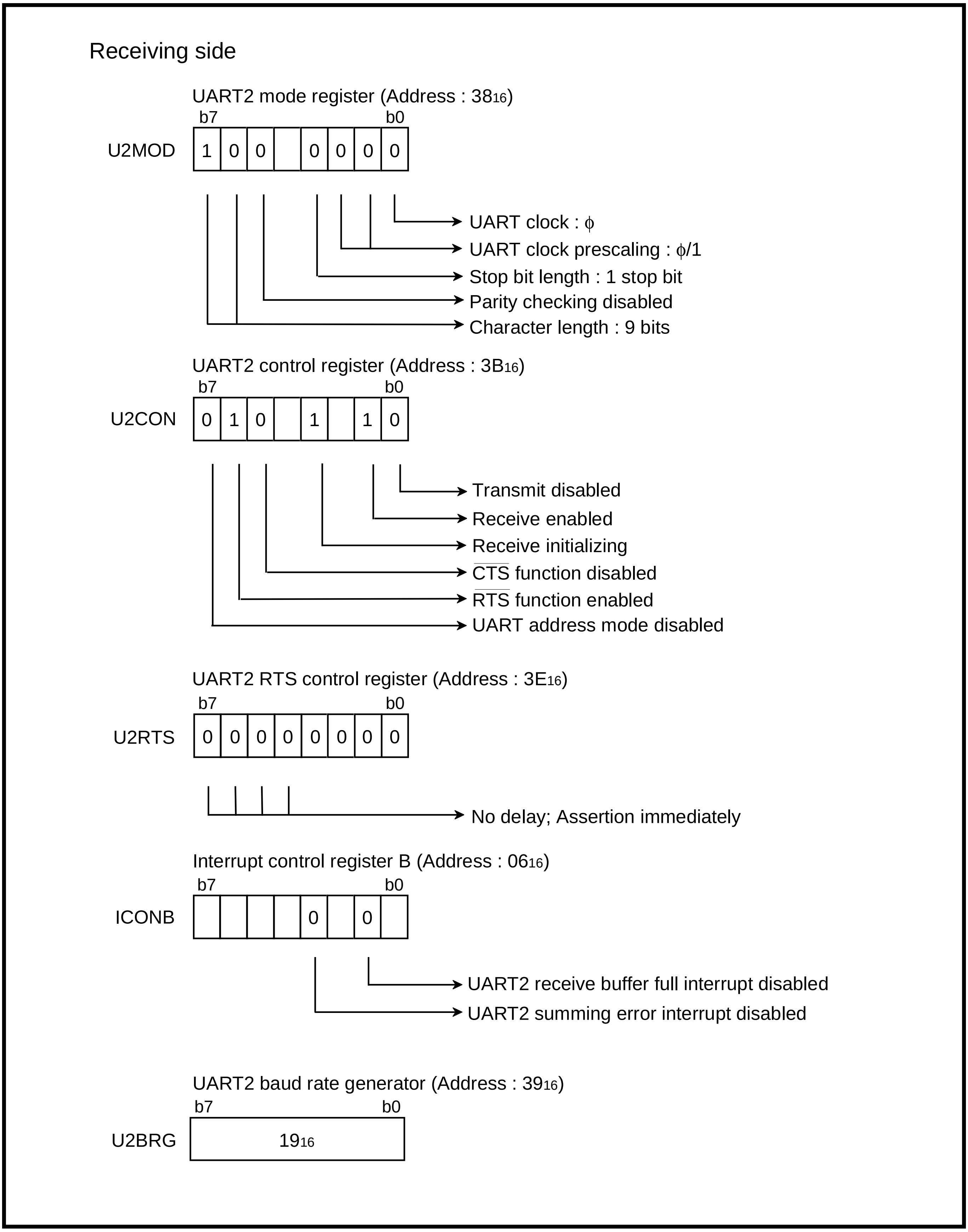


Fig. 2.4.16 Registers setting for receiver (1)

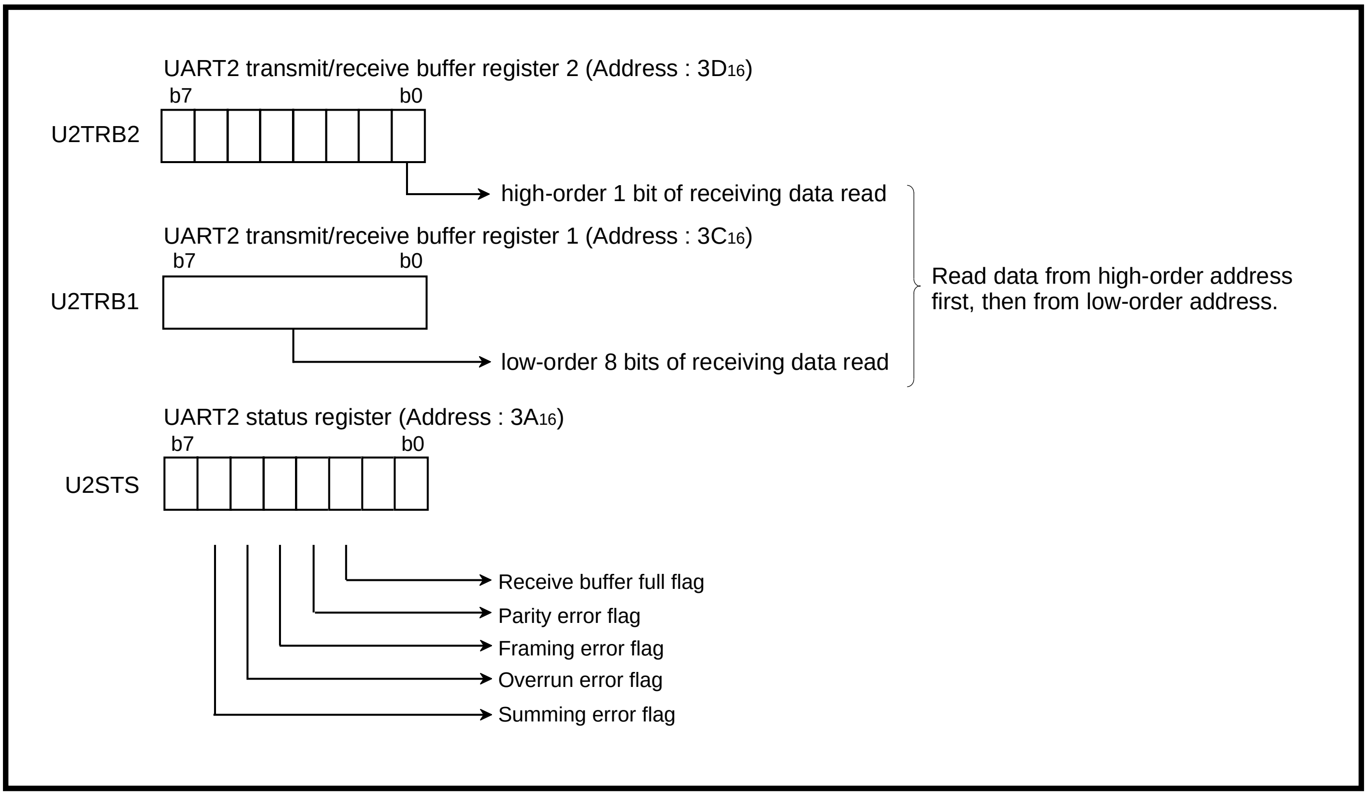


Fig. 2.4.17 Registers setting for receiver (2)

Figure 2.4.18 shows a control procedure of transmitter, and Figure 2.4.19 shows a control procedure of receiver.

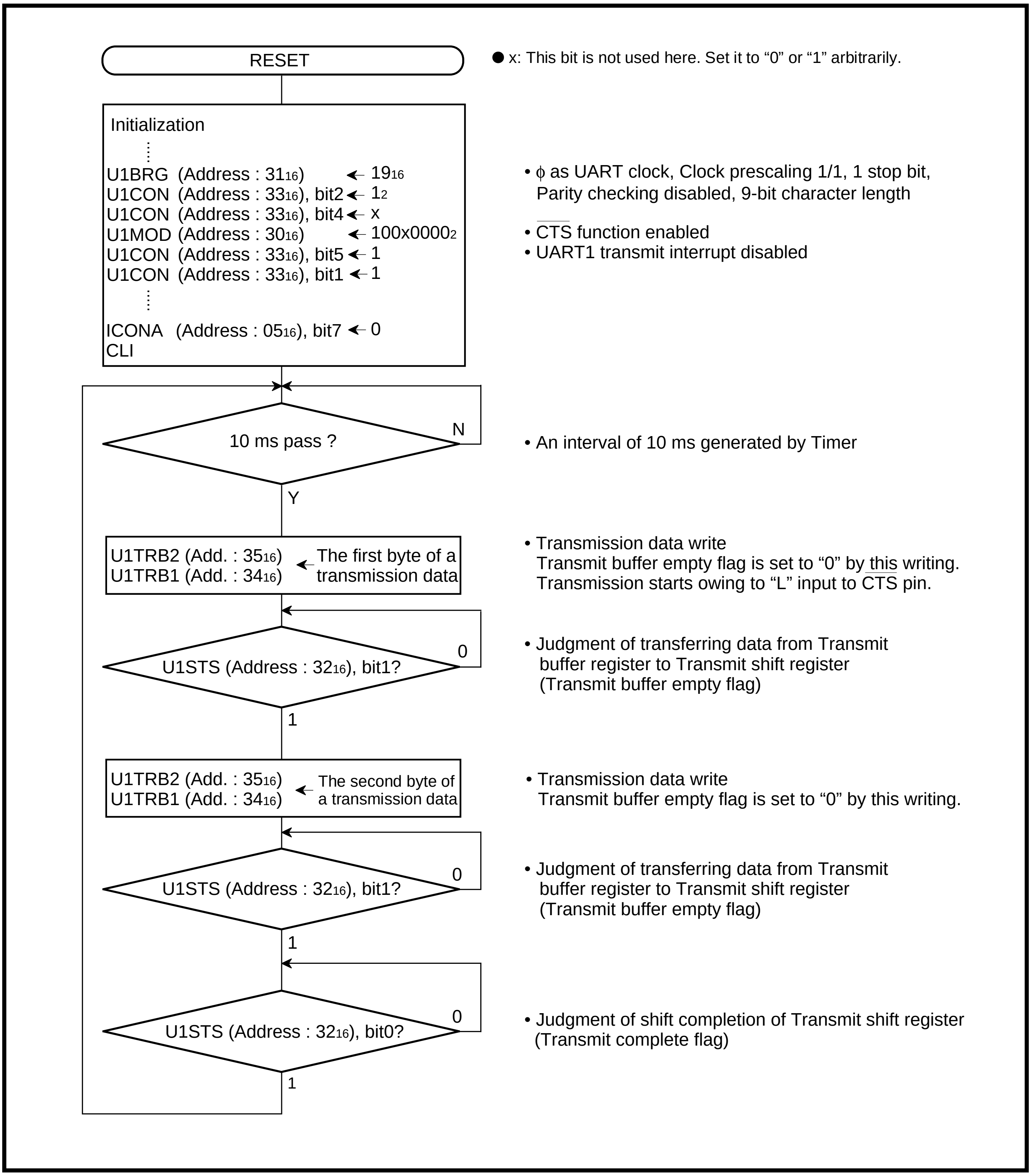


Fig. 2.4.18 Control procedure of transmitter

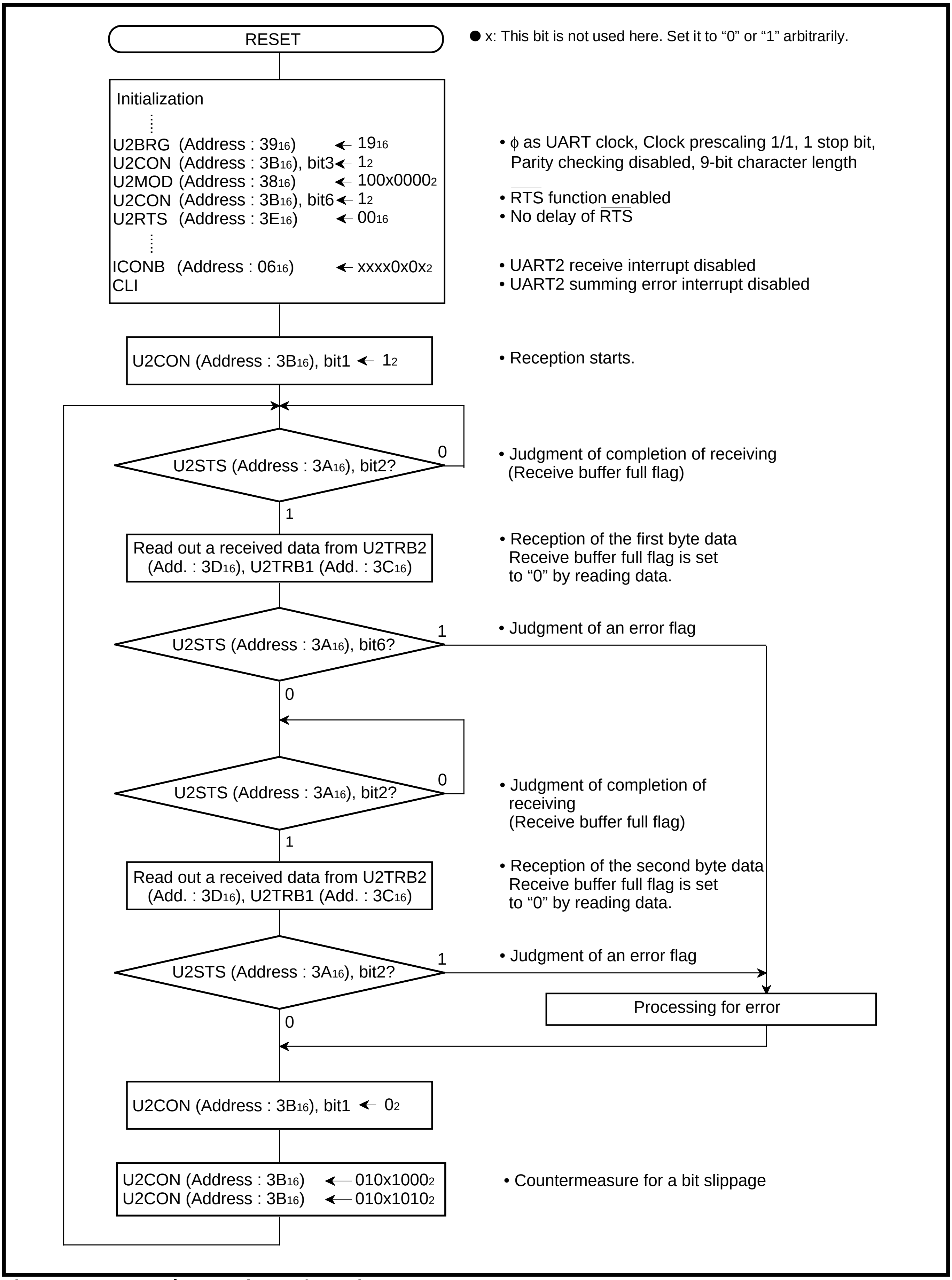


Fig. 2.4.19 Control procedure of receiver

(2) UART address mode

●**Operation explanation**

The UART address mode is intended for use to communicate between the specified MCUs in a multi-MCU environment. The UART address mode can be used in either an 8-bit or 9-bit character length. An address is identified by the MSB of the incoming data being “1”. The bit is “0” for non-address data.

When the MSB of the incoming data is “0” in the UART address mode, the Receive Buffer Full Flag is set to “1”, but the Receive Buffer Full Interrupt Request Bit is not set to “1”. When the MSB of the incoming data is “1”, normal receive operation is performed. In the UART address mode an overrun error is not detected for reception of the 2nd and onward bytes. An occurrence of framing error or parity error sets the Summing Error Interrupt Request Bit to “1” and the data is not received independent of its MSB contents.

Usage of UART address mode is explained as follows:

(1) Set the UART Address Mode Enable Bit to “1”.

(2) Sends the address data of a slave MCU first from a host MCU to all slave MCUs. The MSB of address data must be “1” and the remaining 7 bits specify the address.

(3) The all slave MCUs automatically check for the received data whether its stop bit is valid or not, and whether the parity error occurs or not (when the parity enabled). If these errors occur, the Framing Error Flag or Parity Error Flag and the Summing Error Flag are set to “1”. Then, the Summing Error Interrupt Request Bit is also set to “1”.

(4) When received data has no error, the all slave MCUs must judge whether the address of the received address data matches with their own addresses by a program. After the MSB being “1” is received, the UART Address Mode Enable Bit is automatically set to “0” (disabled).

(5) The UART Address Mode Enable Bit of the slave MCUs which have be judged that the address does not match with them must be set to “1” (enabled) again by a program to disable reception of the following data.

(6) Transmit the data of which MSB is “0” from the host MCU. The slave MCUs disabling the UART address mode receive the data, and their Receive Buffer Full Flags and the Receive Buffer Full Interrupt Request Bits are set to “1”. For the other slave MCUs enabling the UART address mode, their Receive Buffer Full Flag are set to “1”, but their Receive Buffer Full Interrupt Request Bits are not set to “1”.

(7) An overrun error cannot be detected after the first data has been received in UART Address Mode. Accordingly, even if the slave MCUs does not read the received data and the next data has been received, an overrun error does not occur.

Thus, a communication between a host MCU and the specified MCU can be realized.

●**UART address mode application example**

Outline : The slave CPU (B) receives the data from the host CPU, using the UART address mode.

Specifications : •UART1 is used.
•Transfer bit rate : 9600 bps
•Data format: 1ST-8DATA-2ST
•Use of port $P3_1$ for communication control

Figure 2.4.20 shows a connection diagram; Figure 2.4.21 shows the registers setting related to UART address mode; Figures 2.4.22 and 2.4.23 show the control procedures.

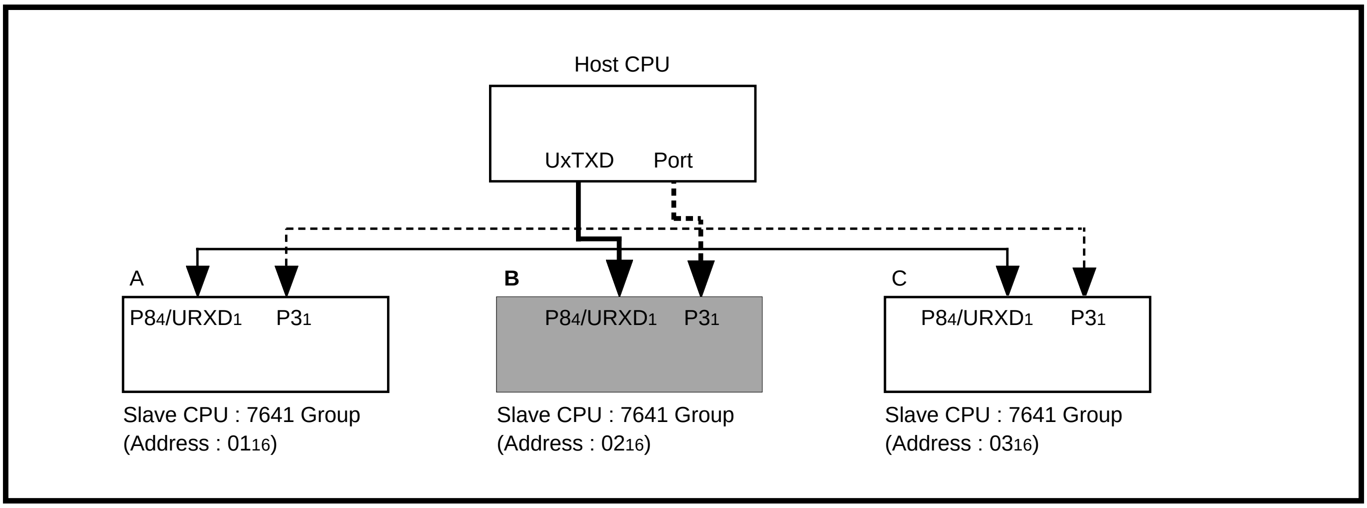


Fig. 2.4.20 Connection diagram

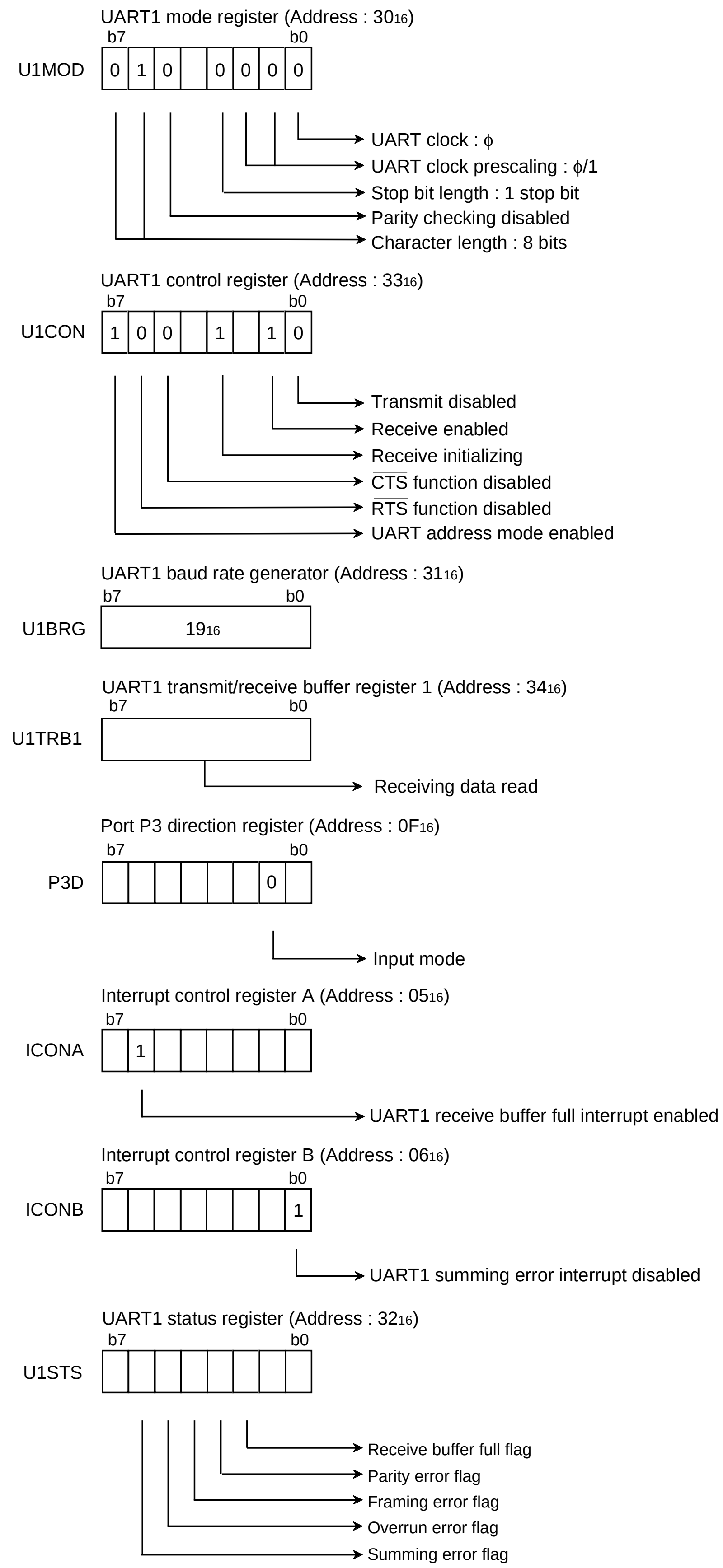


Fig. 2.4.21 Registers setting related to UART address mode

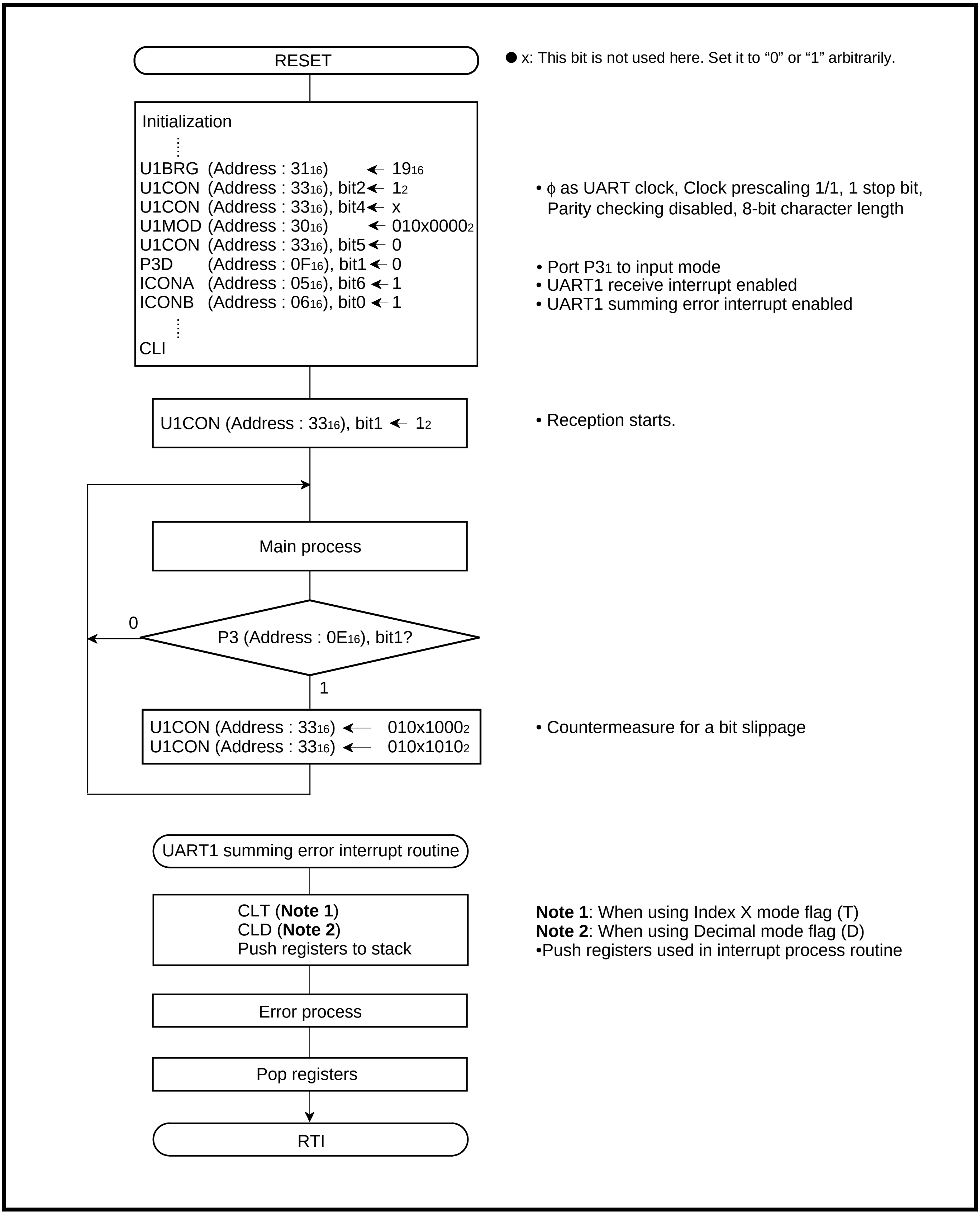


Fig. 2.4.22 Control procedure (1)

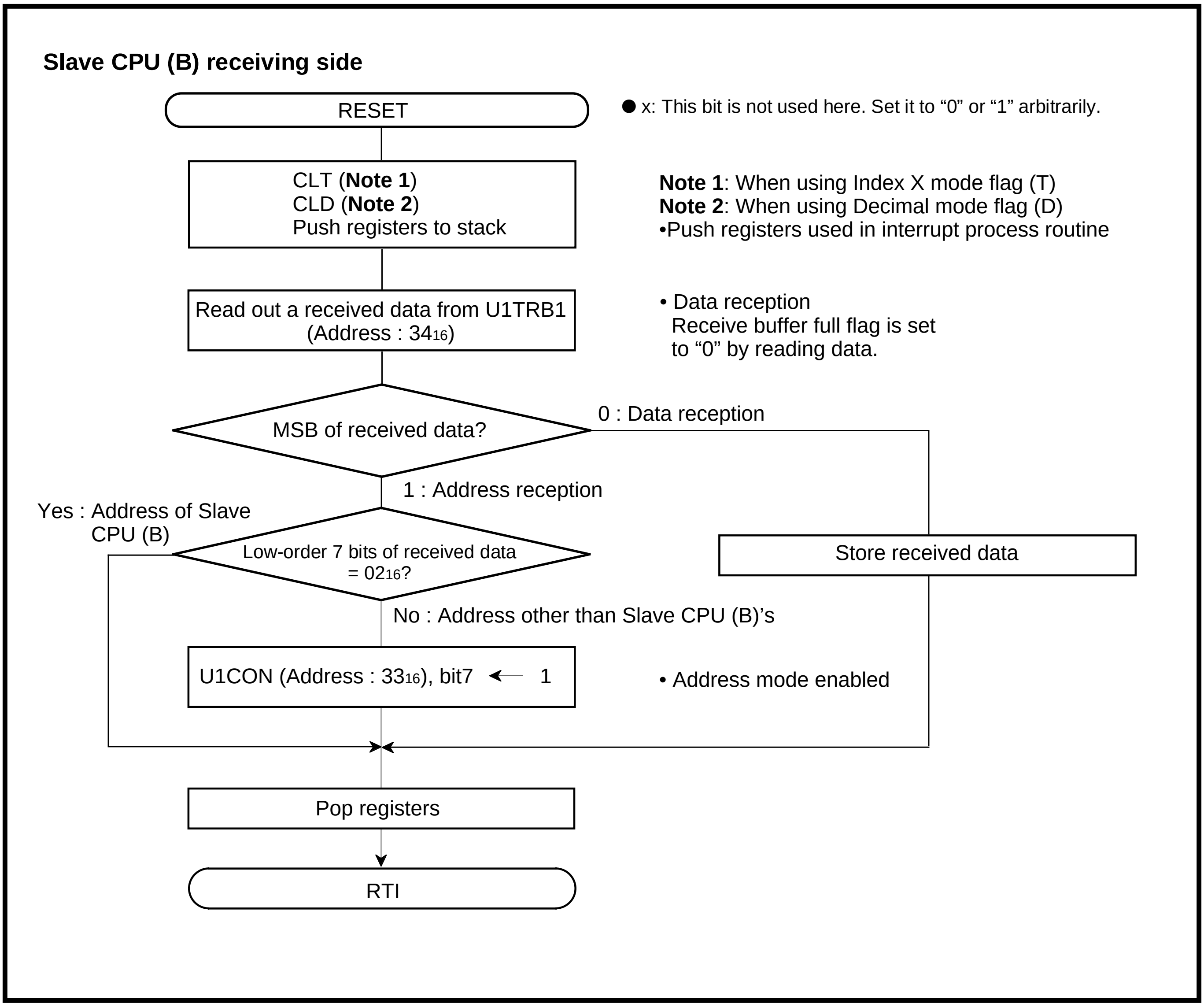


Fig. 2.4.22 Control procedure (2)

2.4.7 Notes on UART

(1) Receive

- When any one of errors occurs, the summing error flag is set to “1” and the UARTx summing error interrupt request bit is also set to “1”. If a receive error occurs, the reception does not set the UARTx receive buffer full interrupt request bit to “1”.
- If the receive enable bit (REN) is set to “0” (disabled) while a data is being received, the receiving operation will stop after the data has been received.
- Setting the receive initialization bit (RIN) to “1” resets the UARTx RTS control register (UxRTS) to “80_{16}”. After setting the RIN bit to “1”, set this UxRTS.

(2) Transmit

- Once the transmission starts, it continues until the last bit has been transmitted even though clearing the transmit enable bit (TEN) to “0” (disabled) or inputting “H” to the $\overline{\text{CTSx}}$ pin. After completion of the current transmission, the transmission is disabled.
- The transmit complete flag (TCM) is changed from “1” to “0” later than 0.5 to 1.5 clocks of the shift clock. Accordingly, take it in consideration to transmit data confirming the TCM flag after the data is written into the transmit buffer register.

(3) Register settings

- If updating a value of UARTx baud rate generator while the data is being transmitted or received, be sure to disable the transmission and reception before updating. If the former data remains in the UARTx transmit buffer registers 1 and 2 at retransmission, an undefined data might be output.
- The all error flags PER, FER, OER and SER are cleared to “0” when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit. These flags are also cleared to “0” by execution of bit test instructions such as **BBC** and **BCS**.
- The transmit buffer empty flag (TBE) is set to “0” when the low-order byte of transmitted data is written into the UARTx (x = 1, 2) transmit buffer register 1. When using 9-bit character length, set the data into the UARTx transmit buffer register 2 (high-order byte) first before the UARTx transmit buffer register 1 (low-order byte).
- The receive buffer full flag (RBF) is set to “0” when the contents of UARTx receive buffer register 1 is read out. When using 9-bit character length, read the data from the UARTx receive buffer register 2 (high-order byte) first before the UARTx receive buffer register 1 (low-order byte).
- If a character bit length is 7 bits, bit 7 of the UARTx transmit/receive buffer register 1 and bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
  If a character bit length is 8 bits, bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
  If a character bit length is 9 bits, bits 1 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are “0” at receiving.
- The reset cannot affect the contents of baud rate generator.

(4) UART address mode

- When the MSB of the incoming data is “0” in the UART address mode, the receive buffer full flag (RBF) is set to “1”, but the receive buffer full interrupt request bit is not set to “1”.
- An overrun error cannot be detected after the first data has been received in UART address mode.
- The UART address mode can be used in either an 8-bit or 9-bit character length. 7-bit character length cannot be used.

(5) CTS function

When the CTS function is enabled, the transmitted data is not transferred to the transmit shift register until "L" is input to the CTSx pin ($P8_6$/$\overline{CTS_1}$, $P8_2$/$\overline{CTS_2}$/SRXD). As the result, do not set the following data to the transmit buffer register.

(6) RTS function

- If the start bit is detected in the term of "H" assertion of RTS, its assertion count is suspended and the RTSx pin remains "H" output. After receiving the last stop bit, the count is resumed.
- Setting the receive initialization bit (RIN) to "1" resets the UARTx RTS control register (UxRTS) to "80_{16}". After setting the RIN bit to "1", set this UxRTS.

(7) Interrupt

- When setting the transmit initialization bit (TIN) to "1", both the transmit buffer empty flag (TBE) and the transmit complete flag (TCM) are set to "1", so that the transmit interrupt request occurs independent of its interrupt source. After setting the transmit initialization bit (TIN) to "1", clear the transmit interrupt request bit to "0" before setting the transmit enable bit (TEN) to "1".
- The transmit interrupt request bit is set and the interrupt request is generated by setting the transmit enable bit (TIN) to "1" even when selecting timing that either of the following flags is set to "1" as timing where the transmission interrupt is generated:

  (1) Transmit buffer empty flag is set to "1"

  (2) Transmit complete flag is set to "1".

  Therefore, when the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as the following sequence:

  (1) Transmit enable bit is set to "1"

  (2) Transmit interrupt request bit is set to "0"

  (3) Transmit interrupt enable bit is set to "1".

2.5 DMAC

This paragraph explains the registers setting method and the notes related to the DMAC.

2.5.1 Memory map

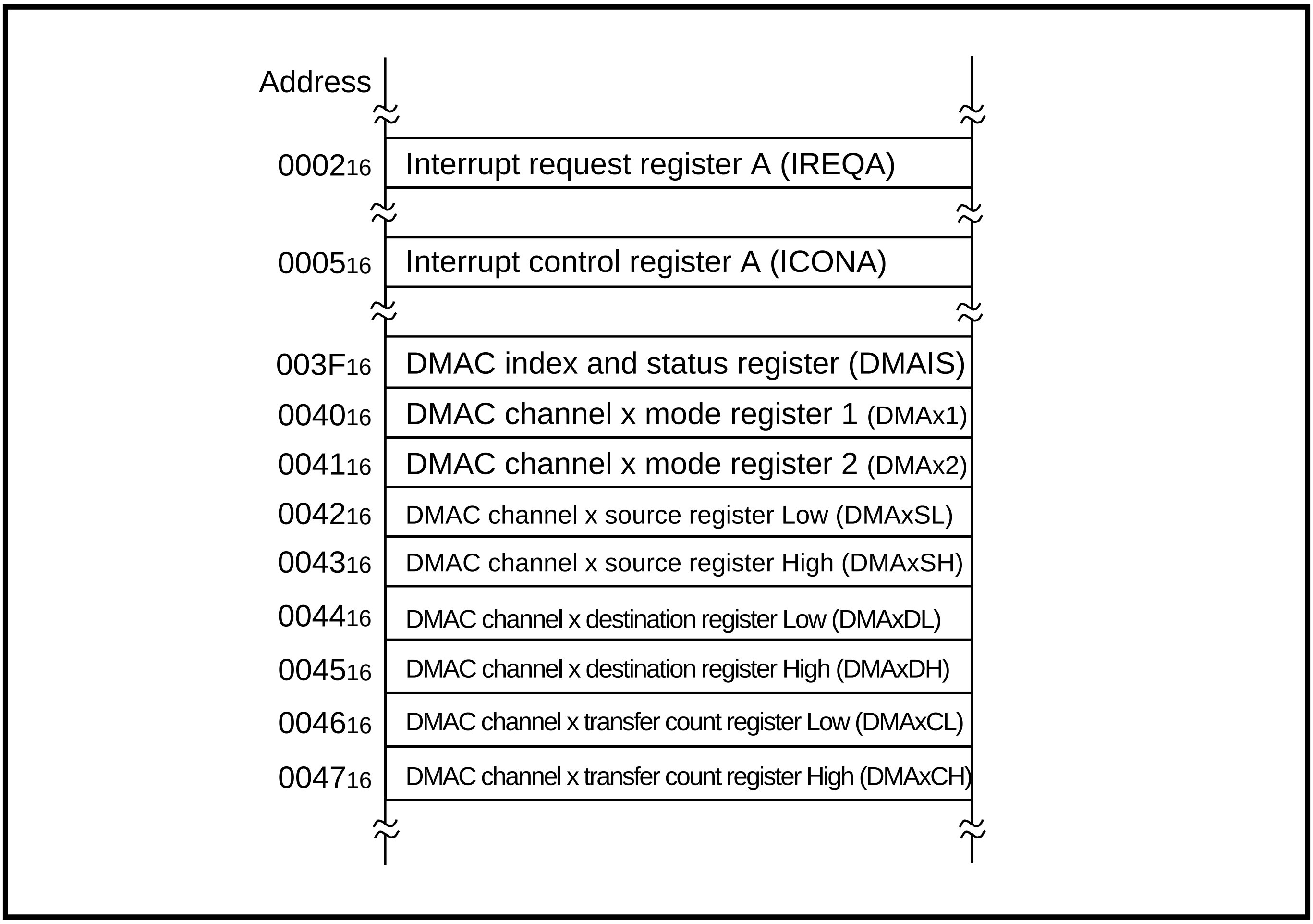


Fig. 2.5.1 Memory map of registers related to DMAC

2.5.2 Related registers

(1) DMAC index and status register

•DMAC channel x (x = 0, 1) count register underflow flag (DxUF)

When the corresponding transfer count register Low (address 46_{16}) underflows, this DxUF flag is set to “1”. Writing “0” into this flag clears it.

•DMAC channel x (x = 0, 1) suspend flag (DxSFI)

When an interrupt routine is processed during any DMA operation, the transfer operation is suspended and the DMAC automatically sets the corresponding DxSFI flag to “1”. As soon as the CPU completes the interrupt operation, the DMAC clears the DxSFI flag to “0” and resumes the original operation from the point where it was suspended.

•DMAC transfer suspend control bit (DTSC)

This bit specifies the transfer mode which can be suspended by an interrupt process.

•DMAC register reload disable bit (DRLDD)

If the DRLDD bit is “1”, when the DMAC channel x transfer count register underflows, the DMAC channel x source registers and destination registers are disabled from being reloaded from their latches.

•Channel index bit (DCI)

The related registers of channels 1 and 2 are assigned on the same SFR addresses. This DCI bit specifies the accessible channel.

DMAC index and status register

b7 b6 b5 b4 b3 b2 b1 b0
[ ][0][ ][ ][ ][ ][ ][ ]

DMAC index and status register
(DMAIS : address $3F_{16}$)

b	Name	Functions	At reset	R	W
0	DMAC channel 0 count register underflow flag (D0UF)	0 : No underflow 1 : Underflow generated	0	○	✻
1	DMAC channel 0 suspend flag (D0SFI) (**Note 1**)	0 : Not suspended 1 : Suspended	0	○	×
2	DMAC channel 1 count register underflow flag (D1UF)	0 : No underflow 1 : Underflow generated	0	○	✻
3	DMAC channel 1 suspend flag (D1SFI) (**Note 1**)	0 : Not suspended 1 : Suspended	0	○	×
4	DMAC transfer suspend control bit (DTSC) (**Note 2**)	0 :Suspending only burst transfers during interrupt process 1 : Suspending both burst and cycle steal transfers during interrupt process	0	○	○
5	DMAC register reload disable bit (DRLDD) (**Note 3**)	0 :Enabling reload of source and destination registers of both channels 1 : Disabling reload of source and destination registers of both channels	0	○	○
6	Fix this bit to “0”.		0	○	○
7	Channel index bit (DCI)	0 : Channel 0 accessible 1 : Channel 1 accessible Accessed registers: Mode register, source register, destination register, transfer count register.	0	○	○

✻: “0” can be set by software, but “1” cannot be set.

Notes 1: Suspended by an interrupt.
2: Transfer suspended during interrupt process
3: This settings affect the source and destination registers of both channels.

Fig. 2.5.2 Structure of DMAC index and status register

(2) DMAC channel x (x = 0, 1) mode register 1

•DMAC channel x source register increment/decrement selection bit (DxSRID)
•DMAC channel x source register increment/decrement enable bit (DxSRCE)
•DMAC channel x destination register increment/decrement selection bit (DxDRID)
•DMAC channel x destination register increment/decrement enable bit (DxDRCE)

These bits select that the DMAC channel X source registers and destination registers are either decreased or increased by 1 after transfer completion.

•DMAC channel x data write control bit (DxDWC)

The DxDWC bit controls write operation to the following registers and their latches: Low and High bytes of DMAC channel x source registers, destination registers and transfer count registers. When the DxDWC bit is “0”, data is simultaneously written into each latch and register. When this bit is “1”, data is written only into their latches.

•DMAC channel x disable after count register underflow enable bit (DxDAUE)

When the DxDAUE bit is “1”, after the DMAC channel x transfer count register Low underflows the corresponding channel x is disabled. The DMAC channel x enable bit (DxCEN, bit 7 of DMAxM2) goes to “0” at the same time.

•DMAC channel x register reload bit (DxRLD)

Writing “1” to the DxRLD bit can update the DMAC channel x source registers, destination registers and transfer count registers with the values in their respective latches. It can be performed at anytime. This bit is fixed to “0” at read.

•DMAC channel x transfer mode selection bit (DxTMS)

The DxTMS bit selects the transfer mode.

DMAC channel x mode register 1 (x = 0, 1)

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x mode register 1
(DMAxM1 : address 40_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel x source register increment/decrement selection bit (DxSRID)	0 : Increment after transfer 1 : Decrement after transfer	0	O	O
1	DMAC channel x source register increment/decrement enable bit (DxSRCE)	0 : Disabled (No change after transfer) 1 : Enabled	0	O	O
2	DMAC channel x destination register increment/decrement selection bit (DxDRID)	0 : Increment after transfer 1 : Decrement after transfer	0	O	O
3	DMAC channel x destination register increment/decrement enable bit (DxDRCE)	0 : Disabled (No change after transfer) 1 : Enabled	0	O	O
4	DMAC channel x data write control bit (DxDWC)	0 : Writing data in reload latches and registers 1 : Writing data in reload latches only	0	O	O
5	DMAC channel x disable after count register underflow enable bit (DxDAUE)	0 : Channel x enabled after count register underflow 1 : Channel x disabled after count register underflow	0	O	O
6	DMAC channel x register reload bit (DxRLD)	0 : Not reloaded 1 : Source, destination, and transfer count registers contents of channel x to be reloaded	0	O (**Note 2**)	O
7	DMAC channel x transfer mode selection bit (DxTMS)	0 : Cycle steal transfer mode 1 : Burst transfer mode	0	O	O

Notes 1: Channels 1 and 2 share this register. The channel selection which can use this register depends on channel index bit, bit 7 of DMAC index and status register.
2: These bits’ contents are “0” at read.

Fig. 2.5.3 Structure of DMAC channel x (x = 0, 1) mode register 1

(3) DMAC channel x (x = 0, 1) mode register 2

•DMAC channel x software transfer trigger bit (DxSWT)

Writing “1” to the DxSWT bit can generate a transfer request as a software trigger. If all of DMACx hardware transfer request source bits (DxHR) are “0”, the software trigger is only transfer request factor. This bit is fixed to “0” at read.

•DMAC channel x USB and master CPU bus interface enable bit (DxUMIE)

When both USB and master CPU bus interface is used as a hardware transfer request source, set the DxUMIE bit to “1”. When not that the master CPU bus interface is used, but that the USB is only used, set the DxUMIE bit to “0” (disabled).

•DMAC channel x transfer initiation source capture register reset bit (DxCRR)

Writing “1” to the DxCRR bit can reset the transfer initiation source capture register. The request of the transfer initiation source is latched asynchronously and it is sampled into the transfer initiation source capture register at a rising edge of ϕ. This bit is fixed to “0” at read.

•DMAC channel x enable bit (DxCEN)

The DMAC channel x is enabled by setting this bit to “1”. When clearing this to “0”, the DMA transfer is finished.

DMAC channel 0 mode register 2

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel 0 mode register 2
(DMA0M2 : address 41_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel 0 hardware transfer request source bits (D0HR)	b3b2b1b0 0 0 0 0 : Not used 0 0 0 1 : UART1 receive interrupt 0 0 1 0 : UART1 transmit interrupt 0 0 1 1 : Timer Y interrupt 0 1 0 0 : INT_0 interrupt 0 1 0 1 : USB endpoint 1 IN_PKT_RDY signal (falling edge active) 0 1 1 0 : USB endpoint 2 IN_PKT_RDY signal (falling edge active) 0 1 1 1 : USB endpoint 3 IN_PKT_RDY signal (falling edge active) 1 0 0 0 : USB endpoint 1 OUT_PKT_RDY signal (rising edge active) 1 0 0 1 : USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active) 1 0 1 0 : USB endpoint 2 OUT_PKT_RDY signal (rising edge active) 1 0 1 1 : USB endpoint 3 OUT_PKT_RDY signal (rising edge active) 1 1 0 0 : Master CPU bus interface OBE_0 signal (rising edge active) 1 1 0 1 : Master CPU bus interface IBF_0 signal, data (rising edge active) 1 1 1 0 : Serial I/O transmit/receive interrupt 1 1 1 1 : $CNTR_1$ interrupt	0	○	○
1			0	○	○
2			0	○	○
3			0	○	○
4	DMAC channel 0 software transfer trigger (D0SWT)	0 : No action 1 : Request of channel 0 transfer by writing "1"	0	○ (**Note 2**)	○
5	DMAC channel 0 USB and master CPU bus interface enable bit (D0UMIE)	0 : Disabled 1 : Enabled	0	○	○
6	DMAC channel 0 transfer initiation source capture register reset bit (D0CRR)	0 : No action 1 : Reset of channel 0 capture register by writing "1"	0	○ (**Note 2**)	○
7	DMAC channel 0 enable bit (D0CEN)	0 : Channel 0 disabled 1 : Channel 0 enabled (**Note 3**)	0	○	○

Notes 1: DMAC channel 0 mode register 2 and DMAC channel 1 mode register 2 are assigned at the same address 41_{16}. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.

2: These bits' contents are "0" at read. This bit is automatically cleared to "0" after writing "1".

3: When setting this bit to "1", simultaneously set the DMAC channel 0 transfer initiation source capture register reset bit (bit 6 of DMA0M2) to "1".

Fig. 2.5.4 Structure of DMAC channel 0 mode register 2

DMAC channel 1 mode register 2

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel 1 mode register 2
(DMA1M2 : address 41_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel 1 hardware transfer request source bits (D1HR)	b3b2b1b0 0 0 0 0 : Not used 0 0 0 1 : UART2 receive interrupt 0 0 1 0 : UART2 transmit interrupt 0 0 1 1 : Timer X interrupt 0 1 0 0 : INT_1 interrupt 0 1 0 1 : USB endpoint 1 IN_PKT_RDY signal (falling edge active) 0 1 1 0 : USB endpoint 2 IN_PKT_RDY signal (falling edge active) 0 1 1 1 : USB endpoint 4 IN_PKT_RDY signal (falling edge active) 1 0 0 0 : USB endpoint 1 OUT_PKT_RDY signal (rising edge active) 1 0 0 1 : USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active) 1 0 1 0 : USB endpoint 2 OUT_PKT_RDY signal (rising edge active) 1 0 1 1 : USB endpoint 4 OUT_PKT_RDY signal (rising edge active) 1 1 0 0 : Master CPU bus interface OBE_1 signal (rising edge active) 1 1 0 1 : Master CPU bus interface IBF_1 signal, data (rising edge active) 1 1 1 0 : Timer 1 interrupt 1 1 1 1 : $CNTR_0$ interrupt	0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4	DMAC channel 1 software transfer trigger (D1SWT)	0 : No action 1 : Request of channel 1 transfer by writing "1"	0	O (**Note 2**)	O
5	DMAC channel 1 USB and master CPU bus interface enable bit (D1UMIE)	0 : Disabled 1 : Enabled	0	O	O
6	DMAC channel 1 transfer initiation source capture register reset bit (D1CRR)	0 : No action 1 : Reset of channel 1 capture register by writing "1"	0	O (**Note 2**)	O
7	DMAC channel 1 enable bit (D1CEN)	0 : Channel 1 disabled 1 : Channel 1 enabled (**Note 3**)	0	O	O

Notes 1: DMAC channel 0 mode register 2 and DMAC channel 1 mode register 2 are assigned at the same address 41_{16}. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: These bits' contents are "0" at read. This bit is automatically cleared to "0" after writing "1".
3: When setting this bit to "1", simultaneously set the DMAC channel 1 transfer initiation source capture register reset bit (bit 6 of DMA1M2) to "1".

Fig. 2.5.5 Structure of DMAC channel 1 mode register 2

DMAC channel x source registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x source registers Low, High
(DMAxSL,DMAxSH : addresses 42_{16}, 43_{16})

b	Functions	At reset	R	W
0	●This is a 16-bit register with a latch.	0	○	○
1		0	○	○
2	●This register indicates the source address for data transfer.	0	○	○
3		0	○	○
4		0	○	○
5		0	○	○
6		0	○	○
7		0	○	○

Notes 1: DMAC channel x source registers low, high of channels 0 and 1 are assigned at the same addresses. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: Write data into the lower bytes first, and then the higher bytes.
3: Read the contents from the higher bytes first, and then the lower bytes.

Fig. 2.5.6 Structure of DMAC channel x source registers Low, High

DMAC channel x destination registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x destination registers Low, High
(DMAxDL,DMAxDH : addresses 44_{16}, 45_{16})

b	Functions	At reset	R	W
0	●This is a 16-bit register with a latch.	0	○	○
1		0	○	○
2	●This register indicates the destination address for data transfer.	0	○	○
3		0	○	○
4		0	○	○
5		0	○	○
6		0	○	○
7		0	○	○

Notes 1: DMAC channel x destination registers low, high of channels 0 and 1 are assigned at the same addresses. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: Write data into the lower bytes first, and then the higher bytes.
3: Read the contents from the higher bytes first, and then the lower bytes.

Fig. 2.5.7 Structure of DMAC channel x destination registers Low, High

DMAC channel x transfer count registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x transfer count registers Low
(DMAxCL : address 46_{16})

b	Functions	At reset	R	W
0	●This is the lower 8-bit register with a latch. Set the lower 8 bits of transfer numbers. ●This register indicates the remaining transfer numbers while transfer is continuing. ●This contents are decreased by 1 at every transfer operation. ●When this register underflows, the DMAC interrupt request bit and the count register underflow flag (**Note 2**) are set to “1”.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x transfer count registers High
(DMAxCH : address 47_{16})

b	Functions	At reset	R	W
0	●This is the higher 8-bit register with a latch. Set the higher 8 bits of transfer numbers. ●This register indicates the remaining transfer numbers while transfer is continuing. ●This contents are decreased by 1 at every transfer operation.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

Notes 1: DMAC channel x transfer count registers low, high of channels 0 and 1 are assigned at the same addresses 46_{16} and 47_{16}. The accessible channel depends on channel index bit, bit 7 of DMAC index and status register.
2: Channel 0 used: Bit 0 of DMAC index and status register
Channel 1 used: Bit 2 of DMAC index and status register
3: Write data into the lower byte first, and then the higher byte.
4: Read the contents from the higher byte first, and then the lower byte.

Fig. 2.5.8 Structure of DMAC channel x transfer count registers Low, High

Interrupt request register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register A
(IREQA : address 02_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	USB SOF interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	$\mathrm{INT_0}$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	$\mathrm{INT_1}$ interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	DMAC0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	DMAC1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	UART1 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	UART1 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻

✻: “0” can be set by software, but “1” cannot be set.

Fig. 2.5.9 Structure of Interrupt request register A

Interrupt control register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register A
(ICONA : address 05_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	USB SOF interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	$\mathrm{INT_0}$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	$\mathrm{INT_1}$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	DMAC0 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	DMAC1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	UART1 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	UART1 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O

Fig. 2.5.10 Structure of Interrupt control register A

2.5.3 DMAC operation description

The DMAC transfers data using the bus without use of the CPU. The DMAC consists of DMAC0 and DMAC1, which have the same function each.
There are two transfer modes: Burst transfer mode or Cycle steal transfer mode.

•Burst transfer mode

Once a DMA transfer request is accepted, an entire batch of data is transferred. The right to use bus is not returned to the CPU until the transfer of all data has been completed.
The DMAC transfers the number of bytes data specified by the transfer count register for each request. The count register is a 16-bit counter; the maximum number of data is 65,536 bytes per one request.

•Cycle steal mode

The DMAC transfers one byte of data for each request. If one byte transfer has been completed and then a DMA transfer request is not generated, the right to use bus is returned to the CPU.

Figure 2.5.11 shows the transfer mode overview.

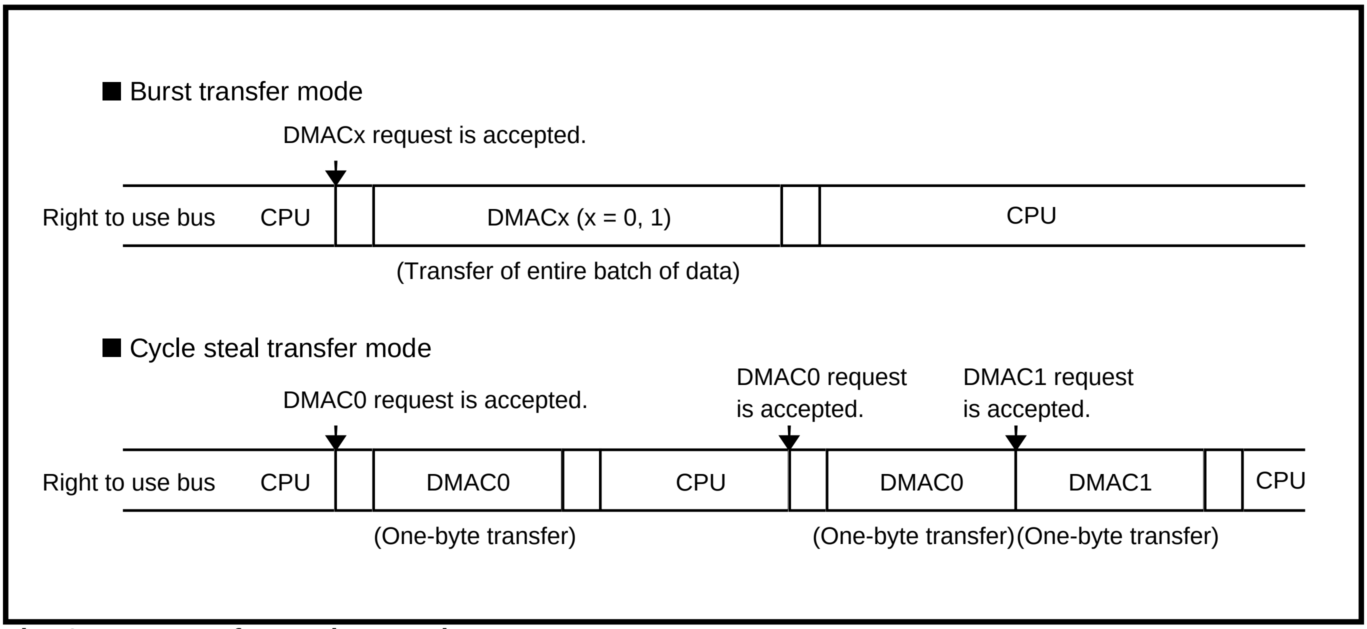


Fig. 2.5.11 Transfer mode overview

(1) Priority

The DMAC places a higher priority on Channel-0 transfer requests than on Channel-1 transfer requests.
If a channel-0 transfer request occurs during a channel-1 burst transfer operation, the DMAC completes the next transfer source and destination read/write operation first, and then starts the channel-0 transfer operation. As soon as the channel-0 transfer is completed, the DMAC resumes the channel-1 transfer operation.

(2) Transfer request acceptance

A transfer request is confirmed at every rising of ϕ. After that a channel priority and a right to use the bus is judged.
A software trigger and/or a hardware factor can be selected as a transfer request source. The DMAC channel x hardware transfer request source bits (DxHR) selects a hardware factor.
Writing “1” to the DMAC channel x software transfer trigger bit (DxSWT) can generate a transfer request as a software trigger.

(3) DMA execution

The selected channel transfers one byte of data from the address indicated by its source register (address 42_{16} or 43_{16}) into the address indicated by its destination register (address 44_{16} or 45_{16}) with at 2 cycles of ϕ.

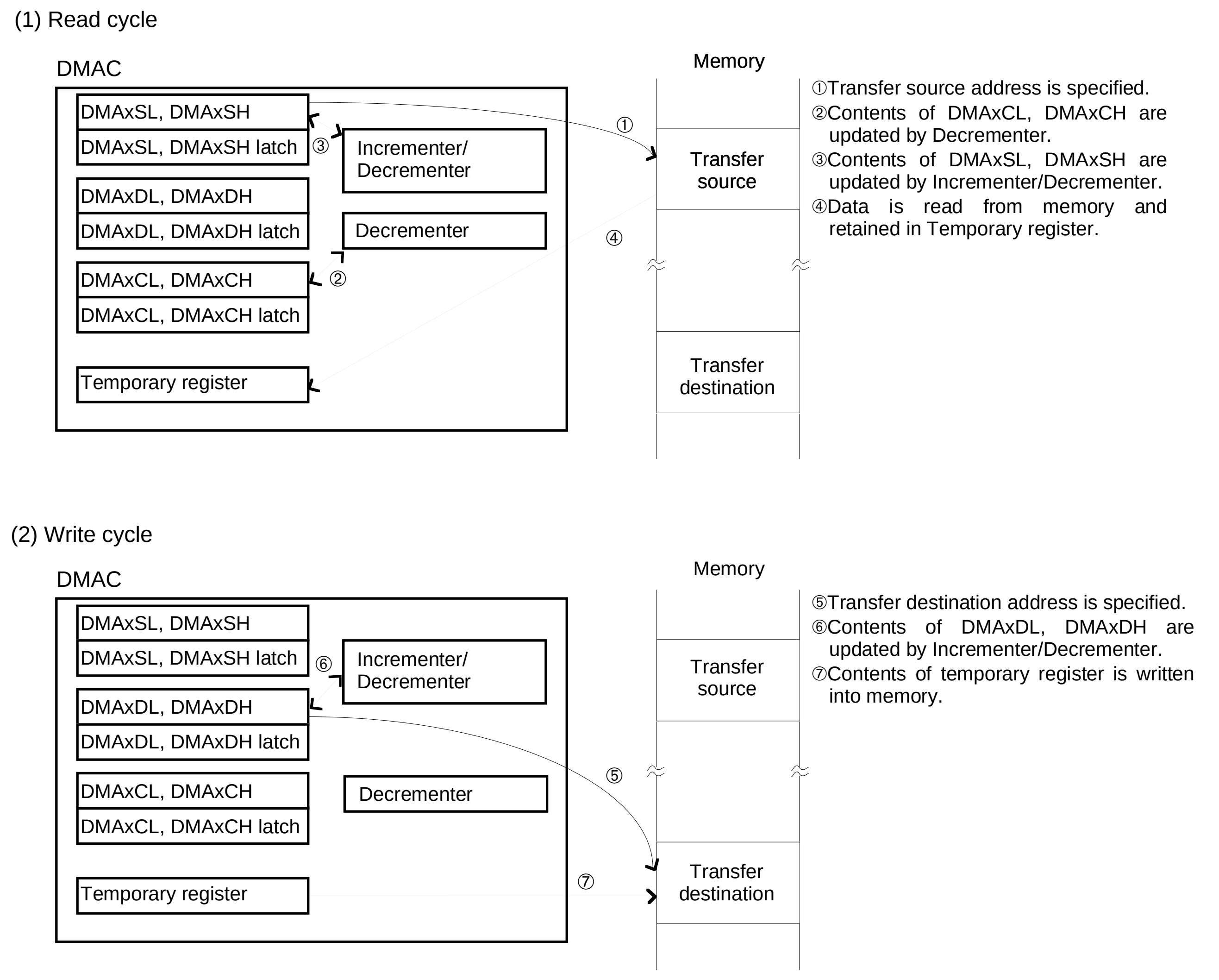
The operataion of the source registers and destination registers after transfer completion can be selected between decreased/increased by 1 and no change with bits 0 to 3 in the DMAC channel x mode register 1.

When the transfer count register underflows, the source registers and destination registers are reloaded from their latches when the DMAC register reload disable bit (DRLDD) is “0”. If the DRLDD bit is set to “1”, a reload is disabled.

A read/write must be performed to the source registers, destination registers and transfer count registers as follows:

Read from each higher byte first, then the lower byte

Write to each lower byte first, then the higher byte.



Note: The temporary register is an internal use register, so that it cannot be read out or written into by program.

Fig. 2.5.12 Basic operation of registers transferring

Figure 2.5.12 shows the basic operation of registers transferring.
Tables 2.5.1 and 2.5.2 shows the address directions and examples of transfer result.

Table 2.5.1 Address directions and examples of transfer result (1)

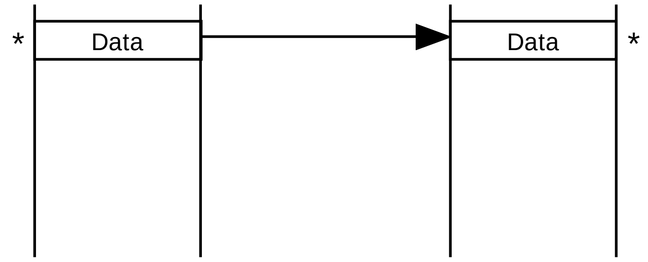
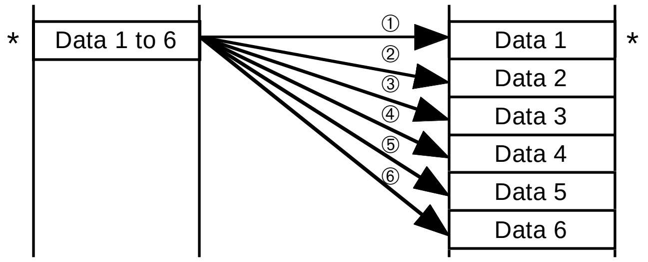
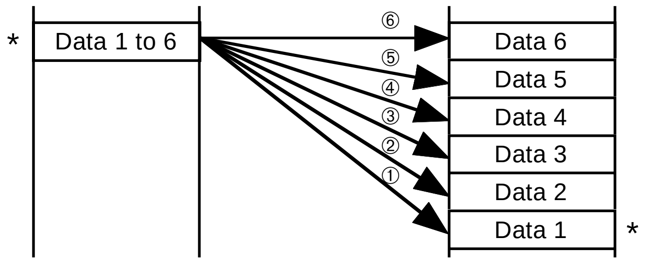
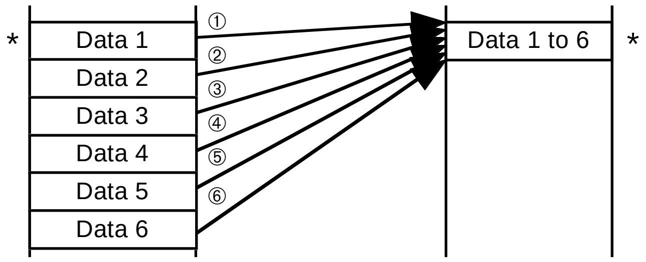
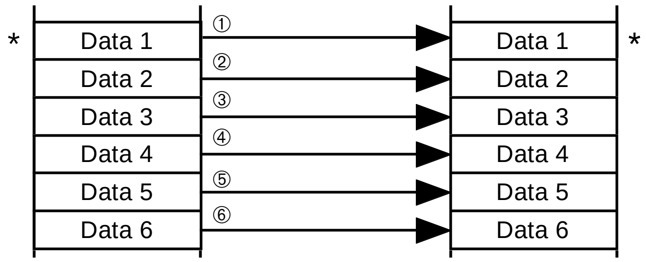
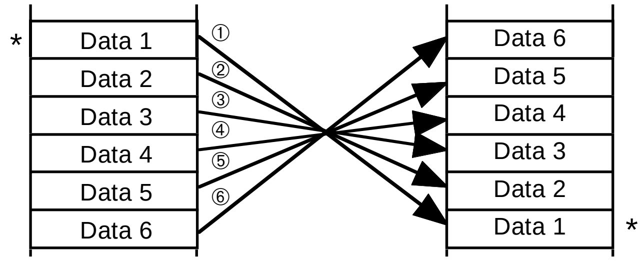
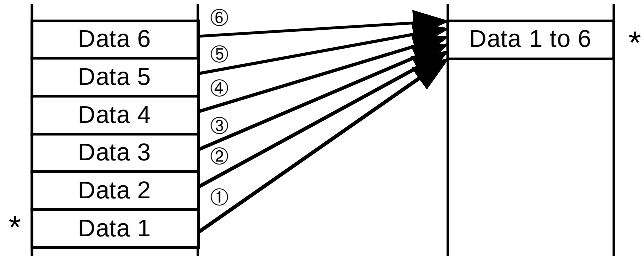
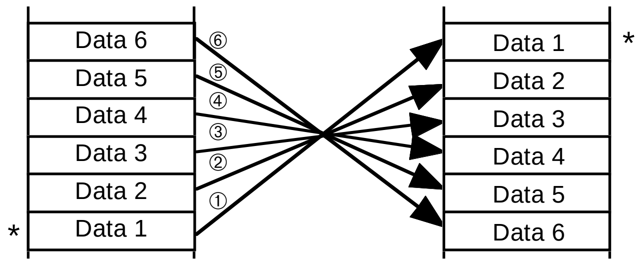
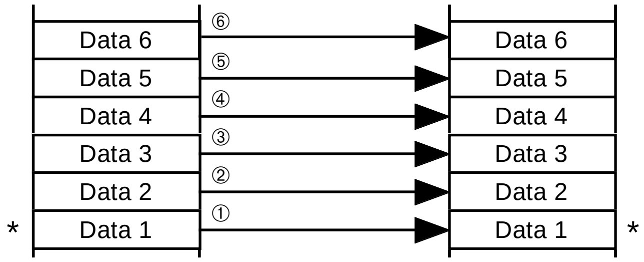
Address direction		Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (Ttransfer result)
Source	Destination			
Fixed	Fixed			
Fixed	Forward			
Fixed	Backward			
Forward	Fixed			
Forward	Forward			

Table 2.5.2 Address directions and examples of transfer result (2)

Address direction		Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (Ttransfer result)
Source	Destination			
Forward	Backward			
Backward	Fixed			
Backward	Forward			
Backward	Backward			

(4) Transfer suspension

Writing “0” to the DMAC channel x enable bit (DxCEN) can compulsorily suspend the transfer being executed. The suspended transfer can be resumed by writing “1” to the DxCEN bit.

When an interrupt request, which is enabled, occurs during any DMA operation, the transfer operation is suspended and the interrupt process routine is initiated. During the interrupt operation, the DMAC automatically sets the corresponding DMAC channel x suspend flag to “1”. When the DMAC transfer suspend control bit (DTSC) is “1”, the transfer is suspended in both burst transfer and cycle steal transfer modes during an interrupt process; when the DTSC bit is “0”, it is suspended in only burst transfer mode.

The suspended transfer due to the interrupt can also be resumed during its interrupt process routine by writing “1” to the DxCEN bit.

2.5.4 DMAC arbitration

The DMA transfer request is accepted at a rising of ϕ. If the bus is not released 1 cycle of ϕ later than the transfer request acceptance, the DMAC will wait for the bus released. When the bus is released, the DMAC has a right to use the bus and starts the DMA transfer unless a request of the right to use the bus having priority over the DMAC occurs.
Table 2.5.3 shows the priority to use the bus.

Table 2.5.3 Priority to use bus

Priority	Factor requesting right to use bus
1 (Higher)	Hold request via $\overline{\text{HOLD}}$ pin
2	DMAC
3	CPU data access
4 (Lower)	CPU instruction access

2.5.5 Transfer time

One-byte transfer of the cycle steal transfer mode requires the time calculated by the following equation:

Time (T) = A + B + C

A: This means the time from the occurrence of DMA transfer source request to sampling it. It needs 1 cycle of ϕ at the maximum. The sampling is asynchronously performed.
B: This means the delay time to sample the DMA transfer source request. It needs 1 cycle of ϕ at the maximum.
C: This means the time to transfer data. It needs 2 cycles of ϕ.

Figures 2.5.13 to 2.5.15 show the timing chart for DMA transfer.

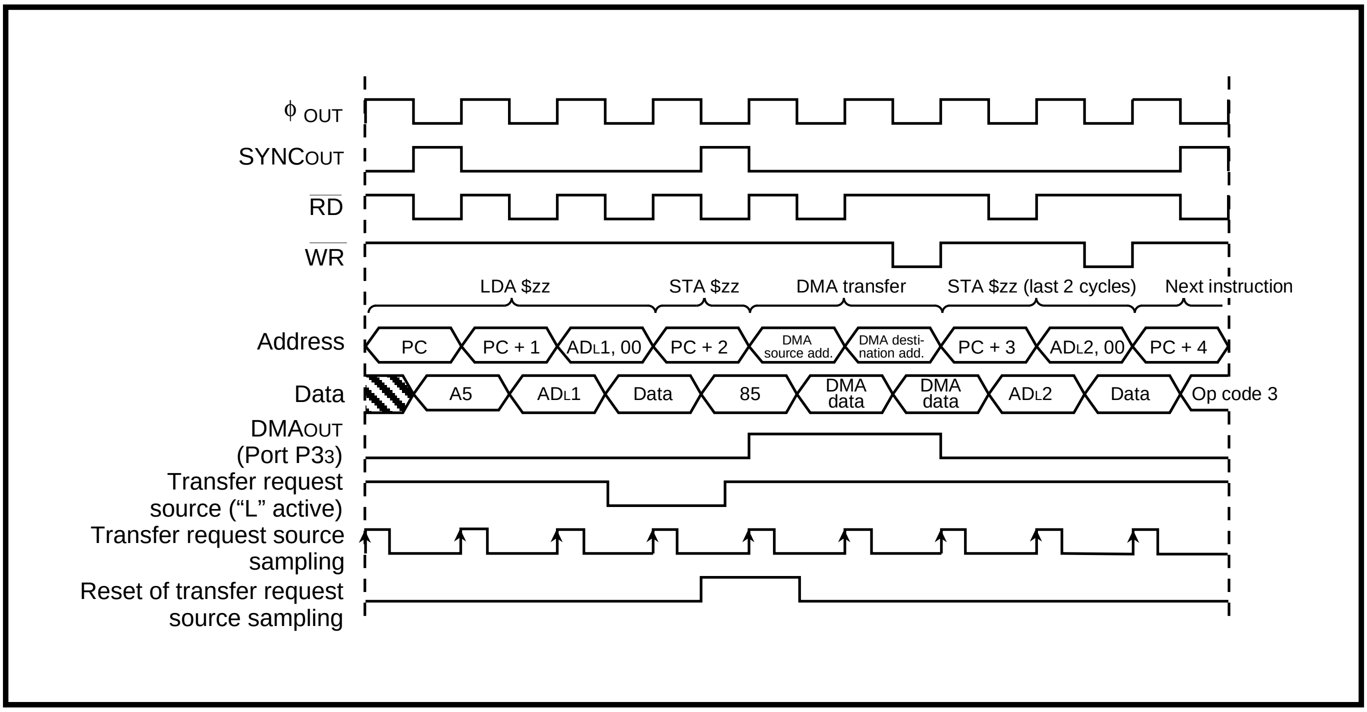


Fig. 2.5.13 Timing chart for cycle steal transfer caused by hardware-related transfer request

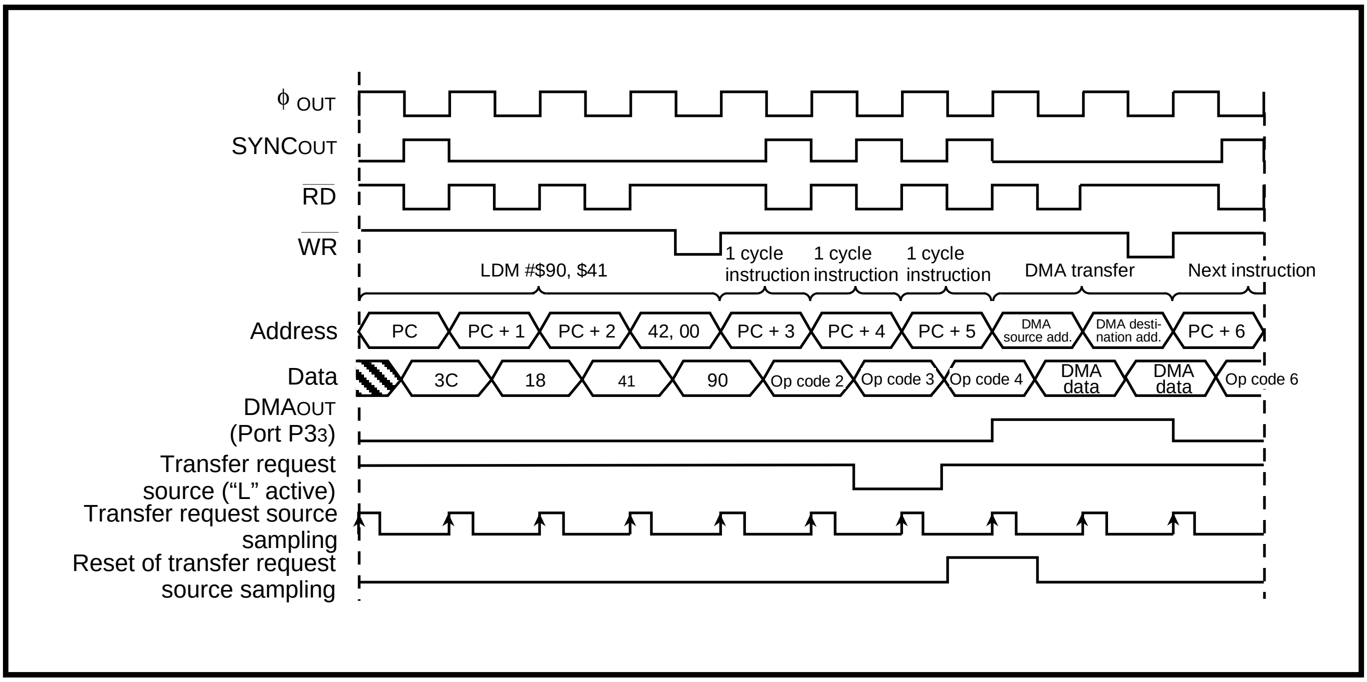


Fig. 2.5.14 Timing chart for cycle steal transfer caused by software trigger transfer request

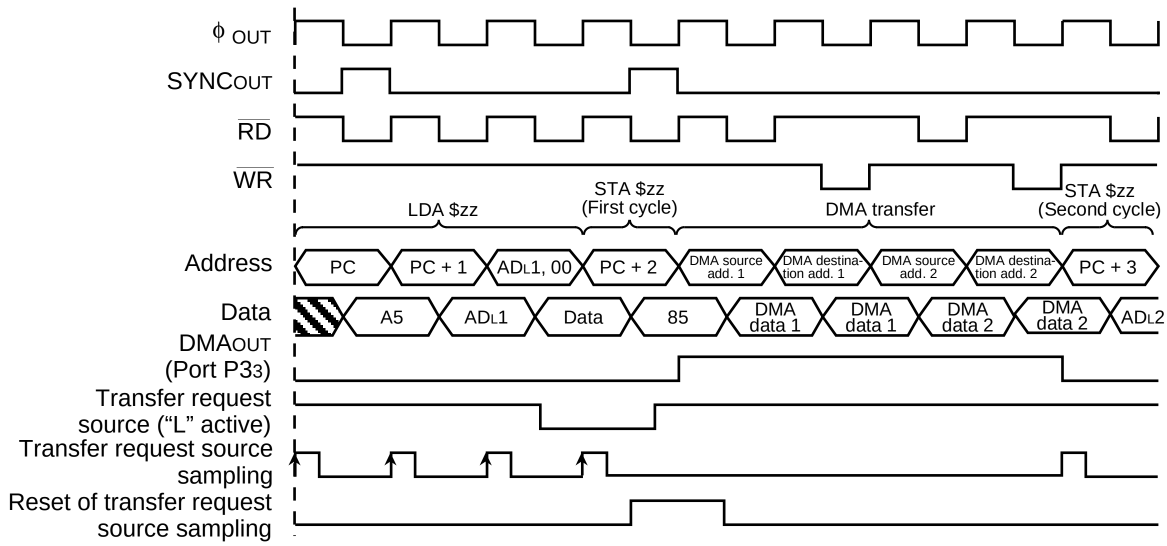


Fig. 2.5.15 Timing chart for burst transfer caused by hardware-related transfer request

2.5.6 DMAC application example

(1) Transfer from external FIFO to USB FIFO

Outline: Data are transferred from external FIFOs to USB FIFOs.

Specifications:
- •A burst transfer is used and 128-byte (128 bytes/packet) continuous transfer is performed. The external FIFO size must be set as 128 bytes of an integral multiple.
- •If the data is deficient in the external FIFO, the DMAC channel 0 transferring is stopped in the DMAC channel 0 interrupt routine process. The DMA transfer is resumed in the main routine process when the data is sufficient.
- •To confirm the external FIFO state after completion of 128 bytes transferring, set the DMAC channel x disable after count register underflow enable bit (DxDAUE) to "1".
- •USB endpoint 2 IN_PKT_RDY (falling edge) selected as hardware transfer request source.
- •Master CPU bus interface unused.

Figures 2.5.16 and 2.5.17 show a setting of the related registers and Figure 2.5.18 shows a control procedure.

If data are transferred from USB FIFOs to external FIFOs as well as this, use USB endpoint 2 OUT_PKT_RDY (rising edge) selected as hardware transfer request source.

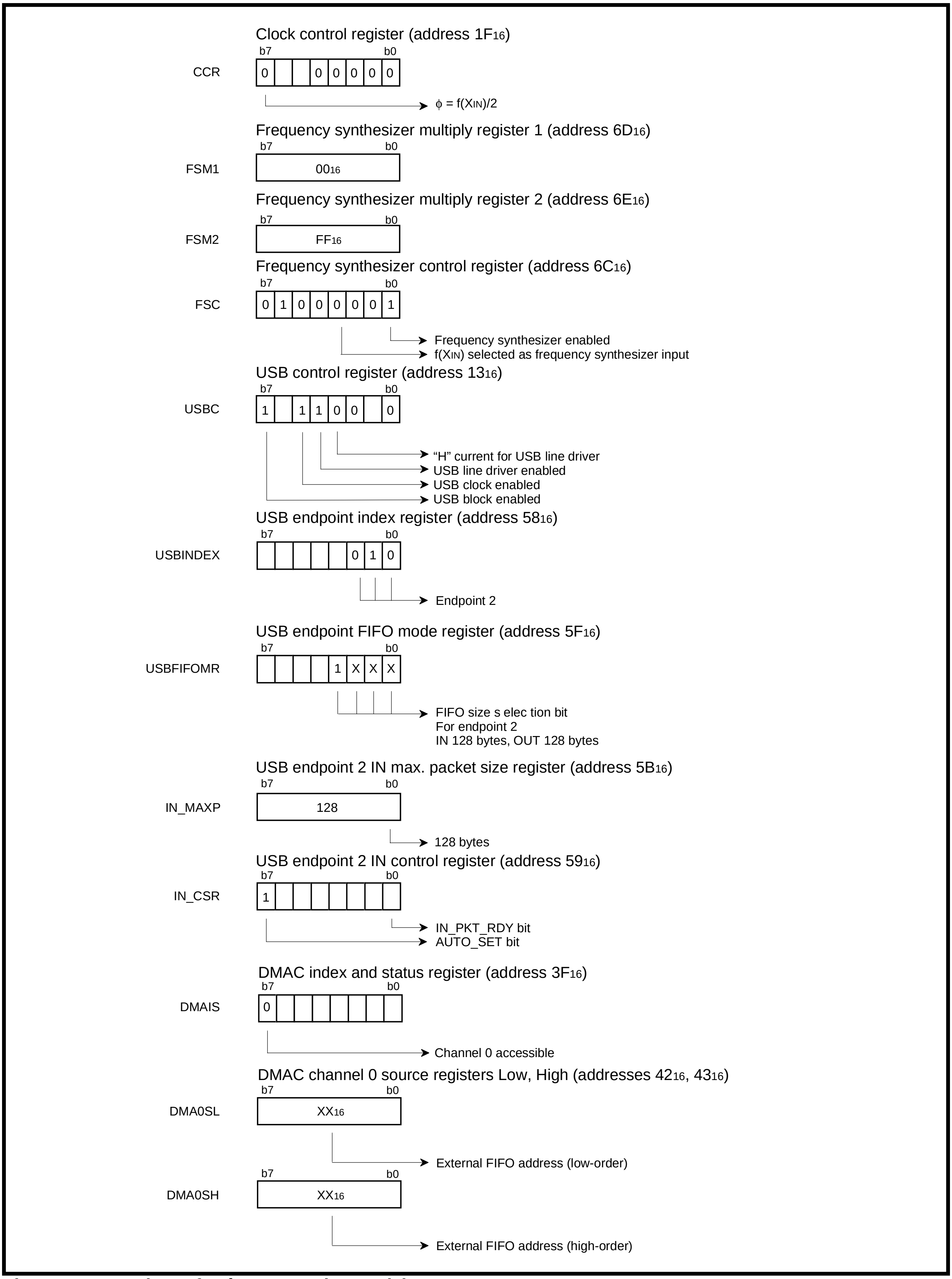


Fig. 2.5.16 Setting of relevant registers (1)

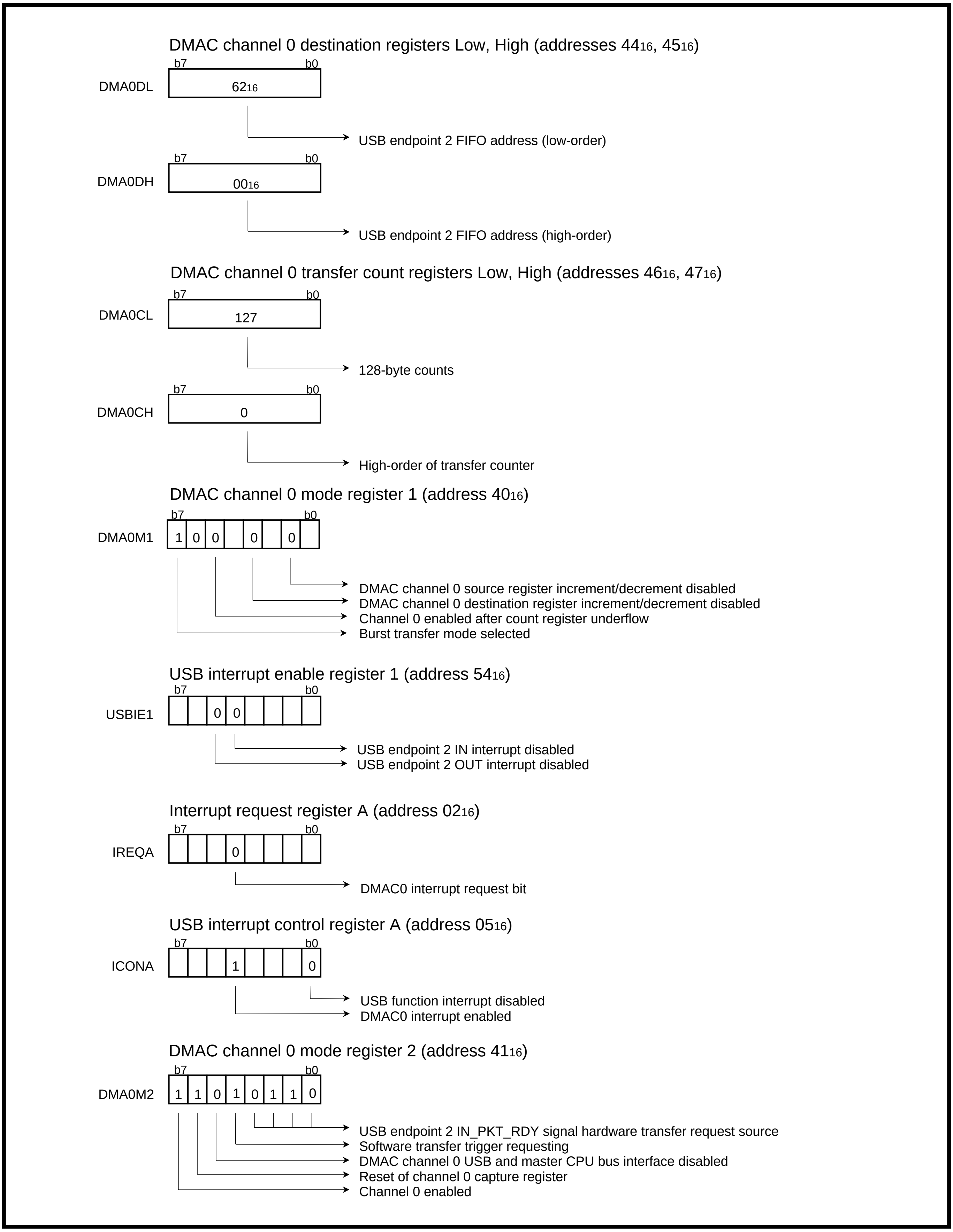


Fig. 2.5.17 Setting of relevant registers (2)

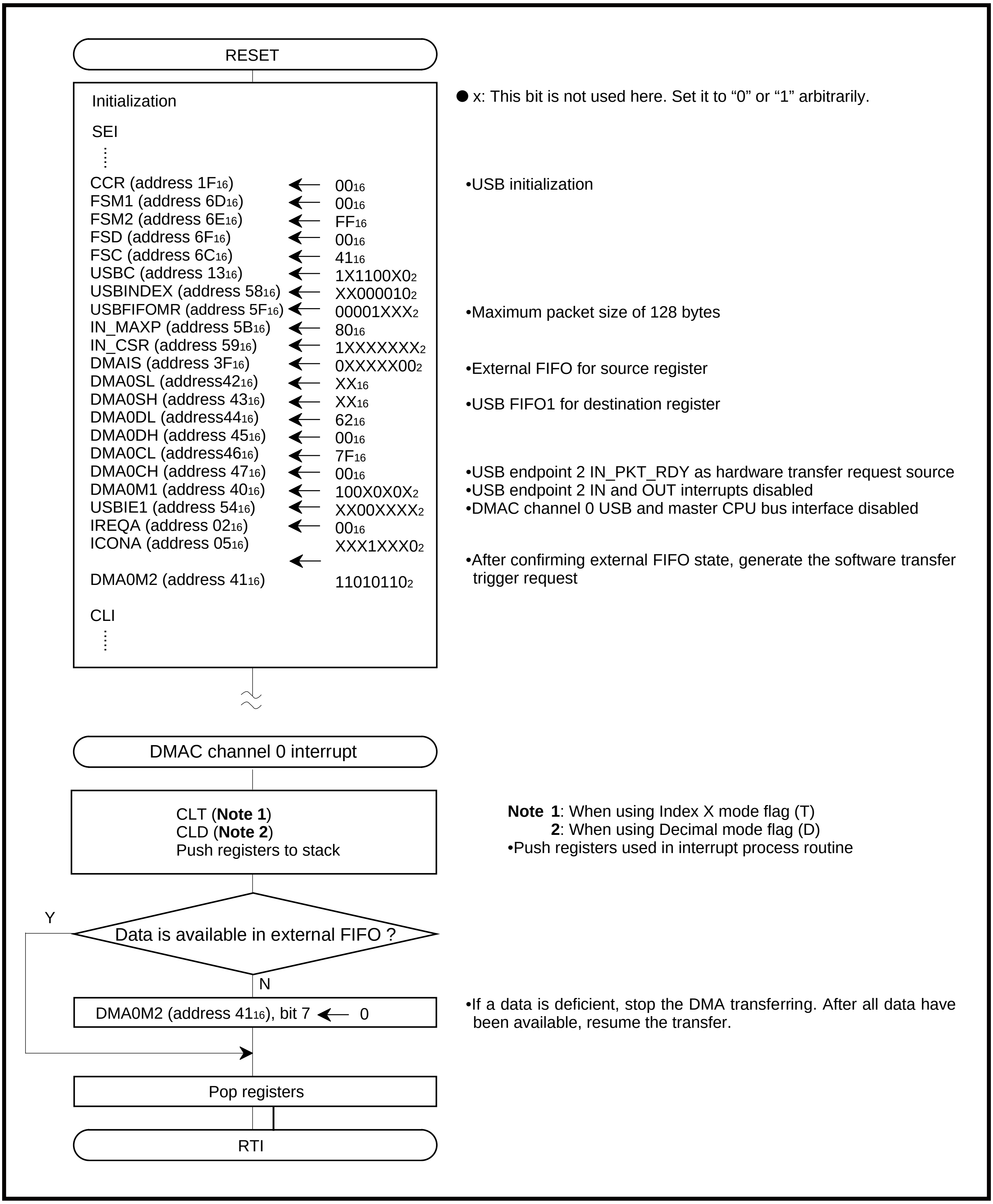


Fig. 2.5.18 Control procedure

2.5.7 Notes on DMAC

(1) Transfer time

- One-byte data transfer requires 2 cycles of ϕ (read and write cycles).
- To perform DMAC transfer due to the different transfer requests on the same DMAC channel or DMAC transfer between both DMAC channels, 1 cycle of ϕ or more is needed before transfer is started.

(2) Priority

- The DMAC places a higher priority on channel-0 transfer requests than on channel-1 transfer requests.
  If a channel-0 transfer request occurs during a channel-1 burst transfer operation, the DMAC completes the next transfer source and destination read/write operation first, and then stops the channel-1 transfer operation.
  The channel-1 transfer operation which has been suspended is automatically resumed from the point where it was suspended so that channel-1 transfer can complete its one-burst transfer unit. This will be performed even if another channel-0 transfer request occurs.
- The suspended transfer due to the interrupt can also be resumed during its interrupt process routine by writing “1” to the DMAC channel x enable bit (DxCEN).

(3) Related registers

- A read/write must be performed to the source registers, transfer destination registers and transfer count registers as follows:
  Read from each higher byte first, then the lower byte
  Write to each lower byte first, then the higher byte.
  Note that if the lower byte is read out first, the values are the higher byte’s.
- Do not access the DMAC-related registers by using a DMAC transfer. The destination address data and the source address data will collide in the DMAC internal bus.
- When setting the DMAC channel x enable bit (bit 7 of address 41_{16}) to “1”, be sure simultaneously to set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of address 41_{16}) to “1”. If this is not performed, an incorrect data will be transferred at the same time when the DMAC is enabled.

(4) USB transfer

One signal among USB endpoint signals 1 to 4 can be selected as the hardware transfer request source. This can realize that transfer between the USB FIFO and the master CPU bus interface input/output buffer is performed effectively. This transfer function is only valid in the cycle steal transfer mode.

(5) DMA_{OUT} pin

In the memory expansion mode and microprocessor mode, the DMA_{OUT} pin ($P3_3/DMA_{OUT}$) outputs “H” during a DMA transfer.

2.6 USB

Some application notes are available on the Web site:
http://www.renesas.com

Please refer to them for explanation and application of USB function.

2.7 Frequency synthesizer

This paragraph explains the registers setting method and the notes related to the frequency synthesizer.

2.7.1 Memory map

Address	
0000_{16}	CPU mode register A (CPMA)
≈	
$006C_{16}$	Frequency synthesizer control register (FSC)
$006D_{16}$	Frequency synthesizer multiply register 1 (FSM1)
$006E_{16}$	Frequency synthesizer multiply register 2 (FSM2)
$006F_{16}$	Frequency synthesizer divide register (FSD)

Fig. 2.7.1 Memory map of registers related to frequency synthesizer

2.7.2 Related registers

CPU mode register A

b7 b6 b5 b4 b3 b2 b1 b0: [ ] [ ] [ ] [ ] [1] [ ] [ ] [ ]

CPU mode register A
(CPMA : address 00_{16})

b	Name	Functions	At reset	R	W
0	Processor mode bits	b1b0 0 0 : Single-chip mode 0 1 : Memory expansion mode 1 0 : Microprocessor mode (**Note 1**) 1 1 : Not available	0	O	O
1			0	O	O
2	Stack page select bit	0 : Page 0 1 : Page 1	1	O	O
3	Fix this bit to “1”.		1	O	O
4	Sub-clock (XCIN-XCOUT) stop bit	0 : Stopped 1 : Oscillating	0	O	O
5	Main clock (XIN-XOUT) stop bit	0 : Oscillating 1 : Stopped	0	O	O
6	Internal system clock select bit (**Note 2**)	0 : External clock (XIN-XOUT or XCIN-XCOUT) 1 : fsyn	0	O	O
7	External clock select bit	0 : XIN-XOUT 1 : XCIN-XCOUT	0	O	O

Notes 1: This is not available in the flash memory version.
2: When (CPMA 7, 6) = (0, 0), the internal system clock can be selected between f(XIN) and f(XIN)/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

Fig. 2.7.2 Structure of CPU mode register A

Frequency synthesizer control register

b7 b6 b5 b4 b3 b2 b1 b0: [ ] [ ] [ ] [0] [ ] [0] [0] [ ]

Frequency synthesizer control register
(FSC : address $6C_{16}$)

b	Name	Functions	At reset	R	W
0	Frequency synthesizer enable bit (FSE)	0 : Disabled 1 : Enabled	0	O	O
1	Fix these bits to “0”.		0	O	O
2			0	O	O
3	Frequency synthesizer input bit (FIN)	0 : f(XIN) 1 : f(XCIN)	0	O	O
4	Fix this bit to “0”.		0	O	O
5	LPF current control bit (CHG1, CHG0) (**Note**)	b1b0 0 0 : Not available 0 1 : Low current 1 0 : Intermediate current (recommended) 1 1 : High current	1	O	O
6			1	O	O
7	Frequency synthesizer lock status bit	0 : Unlocked 1 : Locked	0	O	O

Note: Bits 6 and 5 are set to (bit 6, bit 5) = (1, 1) at reset.
When using the frequency synthesizer, we recommend to set to (bit 6, bit 5) = (1, 0) after locking the frequency synthesizer.

Fig. 2.7.3 Structure of Frequency synthesizer control register

Frequency synthesizer multiply register 1

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer multiply register 1
(FSM1: address $6D_{16}$)

b	Functions	At reset	R	W
0	●f_{VCO} clock is generated by multiplying f_{PIN} clock, which is generated by FSM2, by the contents of this register: $f_{VCO} = f_{PIN} \bullet \{2(n+1)\}$, n: value set to FSM1.	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 2.7.4 Structure of Frequency synthesizer multiply register 1

Frequency synthesizer multiply register 2

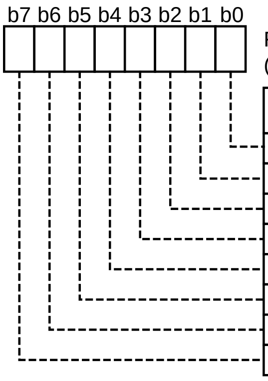
b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer multiply register 2
(FSM2: address $6E_{16}$)

b	Functions	At reset	R	W
0	●f_{PIN} clock is generated by dividing f_{IN} clock by the contents of this register. Either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer is selectable. $f_{PIN} = f_{IN} / \{2(n+1)\}$, n: value set to FSM2	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 2.7.5 Structure of Frequency synthesizer multiply register 2

Frequency synthesizer divide register



Frequency synthesizer divide register
(FSD: address $6F_{16}$)

b	Functions	At reset	R	W
0	●f_{SYN} clock is generated by dividing f_{VCO} clock by the contents of this register: $f_{SYN} = f_{VCO} / \{2(m+1)\}$, m: value set to FSD	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 2.7.6 Structure of Frequency synthesizer divide register

2.7.3 Functional description

The frequency synthesizer generates the 48 MHz clock required by f_{USB} and f_{SYN}, which are multiples of the external input reference $f(X_{IN})$ or $f(X_{CIN})$. To use the frequency synthesizer, set the frequency synthesizer enable bit of frequency synthesizer control register (address $6C_{16}$) to "1".
The frequency synthesizer input bit selects either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer.
Figure 2.7.7 shows the block diagram for the frequency synthesizer circuit.

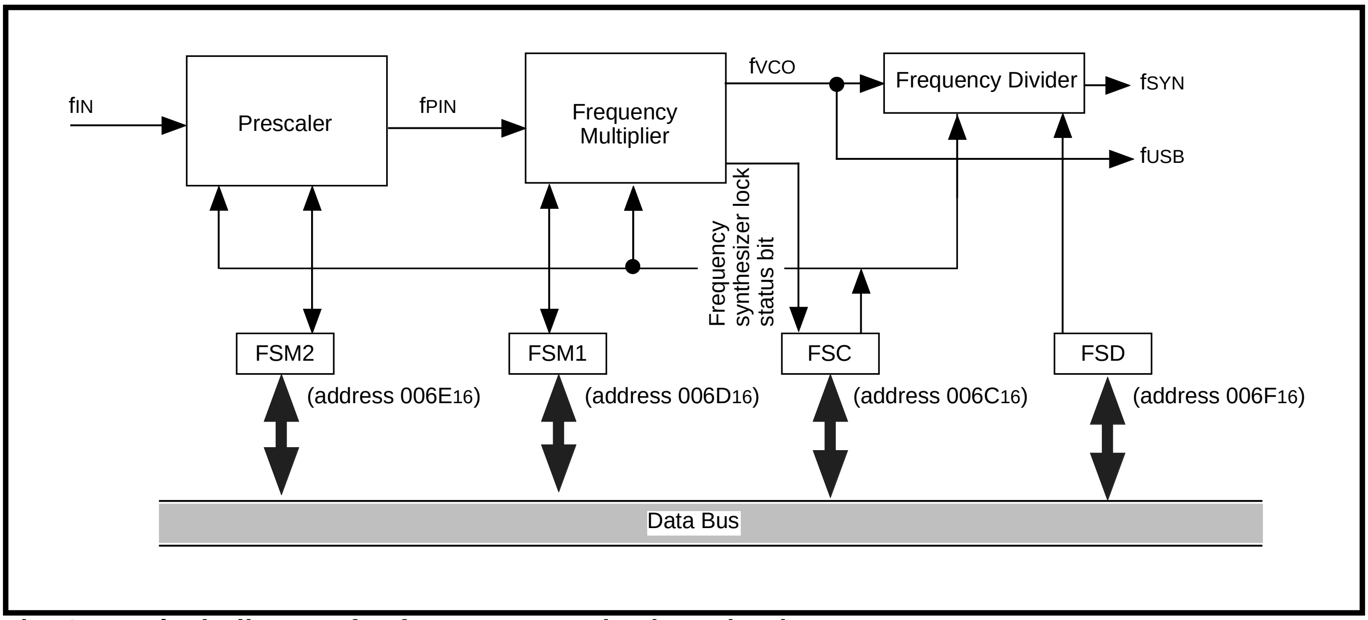


Fig. 2.7.7 Block diagram for frequency synthesizer circuit

(1) f_{PIN}

f_{IN} is divided by the contents of frequency synthesizer multiply register 2 (FSM2: address $6E_{16}$) to generate f_{PIN}, where
$f_{PIN} = f_{IN} / 2(n + 1)$, n: value set to FSM2.
When the value of FSM2 is set to 255, the division is not performed and f_{PIN} will equal f_{IN}.
Figure 2.7.8 shows the frequency synthesizer multiply register 2 setting example.
Note: Be sure to set f_{PIN} to 1 MHz or more.

f_{PIN}	FSM2 register set value		f_{IN}
	Decimal	Hexadecimal	
24 MHz	255	FF_{16}	24.00 MHz
1 MHz	11	$0B_{16}$	24.00 MHz
2 MHz	5	05_{16}	24.00 MHz
3 MHz	3	03_{16}	24.00 MHz
6 MHz	1	01_{16}	24.00 MHz
12 MHz	0	00_{16}	24.00 MHz

Fig. 2.7.8 Frequency synthesizer multiply register 2 setting example

(2) f_{VCO}

f_{VCO} is generated by multiplying f_{PIN} by the contents of frequency synthesizer multiply register 1 (FSM1: address $6D_{16}$), where
$f_{VCO} = f_{PIN} \times \{2(n + 1)\}$, n: value set to FSM1.
Set the value of FSM1 so that f_{VCO} will be 48 MHz.
f_{VCO} is optimized in the frequency synthesizer to be used as f_{USB} and it will be sent into the USB function control unit.
While the frequency synthesizer enable bit is "0" (disabled), f_{VCO} retains "H" or "L" level.
Figure 2.7.9 shows the frequency synthesizer multiply register 1 setting example.

fPIN	FSM1 register set value		fVCO
	Decimal	Hexadecimal	
320 kHz	74	$4A_{16}$	48.00 MHz
2 MHz	11	$0B_{16}$	48.00 MHz
4 MHz	5	05_{16}	48.00 MHz
6 MHz	3	03_{16}	48.00 MHz
12 MHz	1	01_{16}	48.00 MHz
24 MHz	0	00_{16}	48.00 MHz

Fig. 2.7.9 Frequency synthesizer multiply register 1 setting example

(3) f_{SYN}

f_{VCO} is divided by the contents of frequency synthesizer divide register (FSD: address $6F_{16}$) to generate f_{SYN}, where
$f_{SYN} = f_{VCO} / 2(m + 1)$, m: value set to FSD.
When the value of FSD is set to 255, the division is not performed and f_{SYN} becomes invalid.
f_{SYN} can be used as the internal system clock by setting the internal system clock select bit of CPU mode register A.
Figure 2.7.10 shows the frequency synthesizer divide register setting example.
When the frequency synthesizer lock status bit is "1" in the frequency synthesizer enabled, this indicates that f_{SYN} and f_{VCO} have correct frequencies.

fVCO	FSD register set value		fSYN
	Decimal	Hexadecimal	
48.00 MHz	00	00_{16}	24.00 MHz
48.00 MHz	127	$7F_{16}$	187.5 kHz

Fig. 2.7.10 Frequency synthesizer divide register setting example

(4) Recovering from hardware reset

The frequency synthesizer and DC-DC converter must be set up as follows when recovering from a Hardware Reset:

① Enable the frequency synthesizer after setting the frequency synthesizer related registers (addresses $6C_{16}$ to $6F_{16}$). Then wait for 2 ms.

② Check the frequency synthesizer lock status bit. If “0”, wait for 0.1 ms and then recheck.

③ To use the intermediate current, set the LPF current control bits of frequency synthesizer control register (address $6C_{16}$) to (b6, b5) = “10”.

④ When using the USB built-in DC-DC converter, set the USB line driver supply enable bit of the USB control register (address 13_{16}) to “1”. This setting must be done 2 ms or more later than the setup described in step ①. The USB line driver current control bit must be set to “0” at this time. (When Vcc = 3.3V, the setting explained in this step is not necessary.)

⑤ After waiting for (C + 1) ms so that the external capacitance pin (Ext. Cap. pin) can reach approximately 3.3 V, set the USB clock enable bit to “1”. At this time, “C” equals the capacitance (μF) of the capacitor connected to the Ext. Cap. pin. For example, if 2.2 μF and 0.1 μF capacitors are connected to the Ext. Cap. in parallel, the required wait will be (2.3 + 1) ms.

⑥ After enabling the USB clock, wait for 4 or more ϕ cycles, and then set the USB enable bit to “1”.

Do not write to any of the USB internal registers (addresses 50_{16} to 64_{16}) until the USB clock enabled, except for the USB control register (address 13_{16}), clock control register (address $1F_{16}$), and frequency synthesizer control register (address $6C_{16}$).

2.7.4 Notes on frequency synthesizer

- Bits 6 and 5 of the frequency synthesizer control register (address $6C_{16}$) are initialized to (b6, b5) = “11” after reset release. Make sure to set bits 6 and 5 to “10” after the frequency synthesizer lock status bit goes to “1”.
- Use the frequency synthesizer output clocks 2 ms to 5 ms later than setting the frequency synthesizer enable bit to “1” (enabled). After that do not change any register values because it might cause output clocks unstabilized temporarily.
- Make sure to connect a low-pulse filter to the LPF pin when using the frequency synthesizer.
- The frequency synthesizer divide register set value never affects f_{USB} frequency.
- When using the f_{SYN} as an internal system clock, set the frequency synthesizer divide register so that f_{SYN} could be 24 MHz or less.
- When using the frequency synthesized clock function, we recommend using the fastest frequency possible of $f(X_{IN})$ or $f(X_{CIN})$ as an input clock for the PLL.
- Set the value of frequency synthesizer multiply register 2 (FSM2) so that the f_{PIN} is 1 MHZ or higher.

2.8 Master CPU bus interface

This paragraph explains the registers setting method and the notes related to the master CPU bus interface.

2.8.1 Memory map

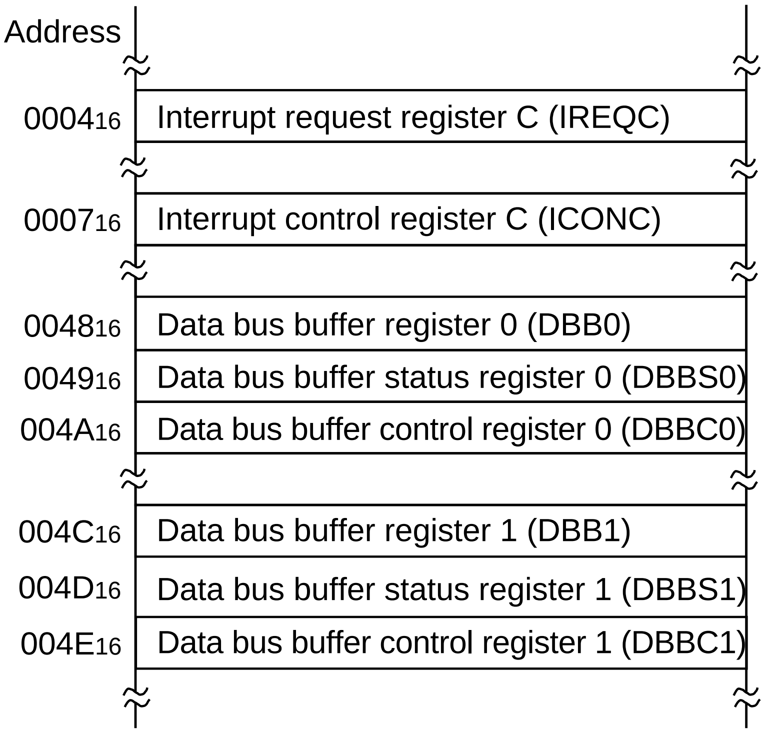


Fig. 2.8.1 Memory map of registers related to master CPU bus interface

2.8.2 Related registers

Data bus buffer register x

b7 b6 b5 b4 b3 b2 b1 b0

Data bus buffer register x (x = 0, 1)
(DBBx: addresses 48_{16}, $4C_{16}$)

b	Functions	At reset	R	W
0	●A data is latched into this register by a write request from the host CPU. ●Write an output data into this register. The data is output on to the data bus by a read request from the host CPU.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

Fig. 2.8.2 Structure of Data bus buffer register x (x = 0, 1)

Data bus buffer status register x

b7 b6 b5 b4 b3 b2 b1 b0

Data bus buffer status register x (x = 0, 1)
(DBBSx: addresses 49_{16}, $4D_{16}$)

b	Name	Functions	At reset	R	W
0	Output buffer full flag (OBFx)	0 : Buffer empty 1 : Buffer full	0	O	X
1	Input buffer full flag (IBFx)	0 : Buffer empty 1 : Buffer full	0	O	X
2	User definable flag (U_2)	This flag can be defined by user freely.	0	O	O
3	A_0 flag (A_{0x})	This flag indicates the condition of A_0 status when the IBFx flag is set.	0	O	X
4	User definable flag (U_4 to U_7)	This flag can be defined by user freely.	0	O	O
5				O	O
6				O	O
7				O	O

Fig. 2.8.3 Structure of Data bus buffer status register x (x = 0, 1)

Data bus buffer control register 0

b7 b6 b5 b4 b3 b2 b1 b0 (b5 = 0)

Data bus buffer control register 0
(DBBC0 : address $4A_{16}$)

b	Name	Functions	At reset	R	W
0	OBF_0 output enable bit	0 : $P5_2$ functions as I/O port. 1 : $P5_2$ functions as OBF_0 output pin.	0	O	O
1	$\overline{IBF_0}$ output enable bit	0 : $P5_3$ functions as I/O port. 1 : $P5_3$ functions as $\overline{IBF_0}$	0	O	O
2	$\overline{IBF_0}$ interrupt select bit	0 : Occurrence due to data write (A_0 = “0”) or command write (A_0 = “1”) 1 : Occurrence due to command write (A_0 = “1”)	0	O	O
3	Output buffer 0 empty interrupt disable bit	0 : Enabled 1 : Disabled	0	O	O
4	Input buffer 0 full interrupt disable bit	0 : Enabled 1 : Disabled	0	O	O
5	Nothing arranged for this bit. Fix this bit to “0”.		0	O	O
6	Master CPU bus interface enable bit	Function of $P5_4$ to $P5_7$, $P6_0$ to $P6_7$; 0 : As I/O ports. 1 : As master CPU bus interface function pins.	0	O	O
7	Bus interface type select bit	0 : RD,WR separate type bus 1 : R/W type bus	0	O	O

Fig. 2.8.4 Structure of Data bus buffer control register 0

Data bus buffer control register 1

b7 b6 b5 b4 b3 b2 b1 b0 (b6 = 0, b5 = 0)

Data bus buffer control register 1
(DBBC0 : address $4E_{16}$)

b	Name	Functions	At reset	R	W
0	OBF_1 output enable bit	0 : $P7_4$ functions as I/O port. 1 : $P7_4$ functions as OBF_1 output pin. (**Note**)	0	O	O
1	$\overline{IBF_1}$ output enable bit	0 : $P7_3$ functions as I/O port. 1 : $P7_3$ functions as $\overline{IBF_1}$ output pin. (**Note**)	0	O	O
2	$\overline{IBF_1}$ interrupt select bit	0 : Occurrence due to data write (A_0 = “0”) or command write (A_0 = “1”) 1 : Occurrence due to command write (A_0 = “1”)	0	O	O
3	Output buffer 1 empty interrupt disable bit	0 : Enabled 1 : Disabled	0	O	O
4	Input buffer 1 full interrupt disable bit	0 : Enabled 1 : Disabled	0	O	O
5	Nothing arranged for these bits.		0	O	O
6	Fix these bits to “0”.		0	O	O
7	Data bus buffer function select bit	0 : Single data bus buffer mode($P7_2$ functions as I/O port.) 1 : Double data bus buffer mode($P7_2$ functions as $\overline{S_1}$ input pin.)	0	O	O

Note: This can be selected when the data bus buffer function select bit is “1”.

Fig. 2.8.5 Structure of Data bus buffer control register 1

2.8.3 Functional description

The internal 2-byte bus interface can be controlled by the signals from the host CPU side; it is called the slave mode. This bus interface allows the 7641 group to be directly connected with a R/W type of CPU bus or a RD and WR separate type of CPU bus.

Ports $P5_2$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_4$ become the master CPU bus interface function pins when the master CPU bus interface is enabled.

$P6_0$ to $P6_7$: Function as data I/O pins (DQ_0 to DQ_7). (Be sure to set them to input mode by setting the port P6 direction register to “0”.)

$P5_2$, $P7_4$: Function as the status signal OBFx (x = 0, 1). The state of the output buffer DBBx (x = 0, 1) is output.

$P5_3$, $P7_3$: Function as the status signal $\overline{IBFx}$ (x = 0, 1). The state of the input buffer DBBx (x = 0, 1) is output.

$P5_4$, $P7_2$: Function as the status signal $\overline{Sx}$ (x = 0, 1). The chip select signals from the host CPU are input.

$P5_5$: Functions as the status signal A_0. When A_0 is “H”, the host CPU can read the contents of the data bus buffer status register x (x = 0, 1) (addresses 49_{16}, $4D_{16}$). When A_0 is “L”, the contents of data bus buffer register x (x = 0, 1) can be output owing to a read request from the host CPU.
When a data is written into the data bus buffer register x (x = 0, 1), if A_0 is “L”, this indicates that its contents are the data; if “H”, this indicates that they are the command.

$P7_2$ to $P7_4$ become the master CPU bus interface function pins only when the double data bus buffer mode is used.

Tables 2.8.1 and 2.8.2 shows the bus control signal and data bus state; Figure 2.8.6 shows a connection example.

Table 2.8.1 Bus control signal and data bus state-RD/WR separate type

$\overline{Sx}$ signal	$\overline{RD}$ signal	$\overline{WR}$ signal	A_0 signal	Data bus state	Data on data bus
L	L	H	L	Read	Output data
L	L	H	H	Read	DBBSx contents
L	H	L	L	Write	Input data (data)
L	H	L	H	Write	Input data (command)
H	–	–	–	High-impedance	–

Table 2.8.2 Bus control signal and data bus state-R/W type

$\overline{Sx}$ signal	$R/\overline{W}$ signal	E signal	A_0 signal	Data bus state	Data on data bus
L	H	H	L	Read	Output data
L	H	H	H	Read	DBBSx contents
L	L	H	L	Write	Input data (data)
L	L	H	H	Write	Input data (command)
H	–	–	–	High-impedance	–

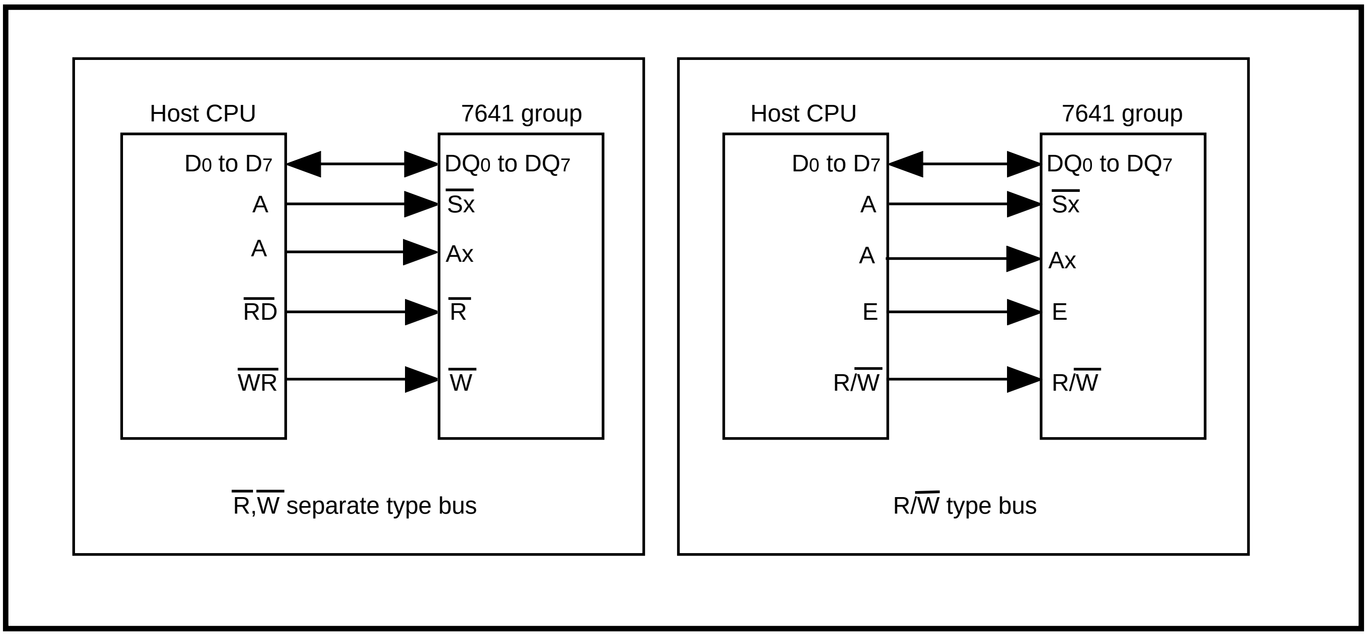


Fig. 2.8.6 Connection example

(1) Data bus buffer mode

The selection of either the single data bus buffer mode or the double data bus buffer mode can be performed by the Data Bus Buffer Function Select Bit.

•Single data bus buffer mode

The data bus buffer 0 of 1 byte only is used.

•Double data bus buffer mode

The data bus buffer 0 and data bus buffer 1, totalling 2 bytes, are used.

(2) Interrupt

•Input buffer full interrupt

When the data is input to the data bus buffer register x, the Input Buffer Full Flag (IBFx ; x = 0, 1) is set to “1” and the interrupt request occurs. When the data is read out from the data bus buffer register x, the IBFx flag is cleared to “0”.

•Output buffer empty interrupt

When the data is written to the data bus buffer register x, the Output Buffer Full Flag (OBFx ; x = 0, 1) is set to “1”. When the host CPU reads out from the data bus buffer register x, the OBFx flag is cleared to “0” and the output buffer empty interrupt request occurs.

2.8.4 Operation description

(1) Input operation

The bus interface input operation is explained as the following:

① When the logical OR of $\overline{Sx}$ (x = 0, 1) and $\overline{W}$ is "0", the data bus state is latched into the input data bus buffer register x (x = 0, 1) at the rising of $\overline{W}$ input signal.

② When the data is latched into the input data bus buffer register x, the Input Buffer Full Flag (IBFx ; x = 0, 1) of the data bus buffer status register x (x = 0, 1) is simultaneously set to "1".

③ When the IBFx flag is set to "1", the input buffer full interrupt request occurs.

④ At the timing ③, the A_0 level is stored into the A_0 flag, bit 3 of the data bus buffer status register x. Refer to the A_0 flag to judge whether the read contents of the input data bus buffer register x are data or a command.

(2) Output operation

The bus interface output operation is explained as the following:

① Writing data to the output data bus buffer register x (x = 0, 1) sets the Output Buffer Full Flag (OBFx ; x = 0, 1) of the data bus buffer status register x to "1".

② When the logical OR of $\overline{Sx}$, $\overline{R}$ and A_0 is "0", the contents of the output data bus buffer register x are output on the data bus and the OBFx flag is simultaneously cleared to "0".

③ At the rising of the $\overline{R}$ input signal, the output buffer empty interrupt request occurs.

2.8.5 Master CPU bus interface application example

(1) Data communication with host CPU

Outline : The MCU communicates with the host CPU by using the single data bus buffer mode.

Figure 2.8.7 shows a setting of the related registers and Figure 2.8.8 shows a control procedure.

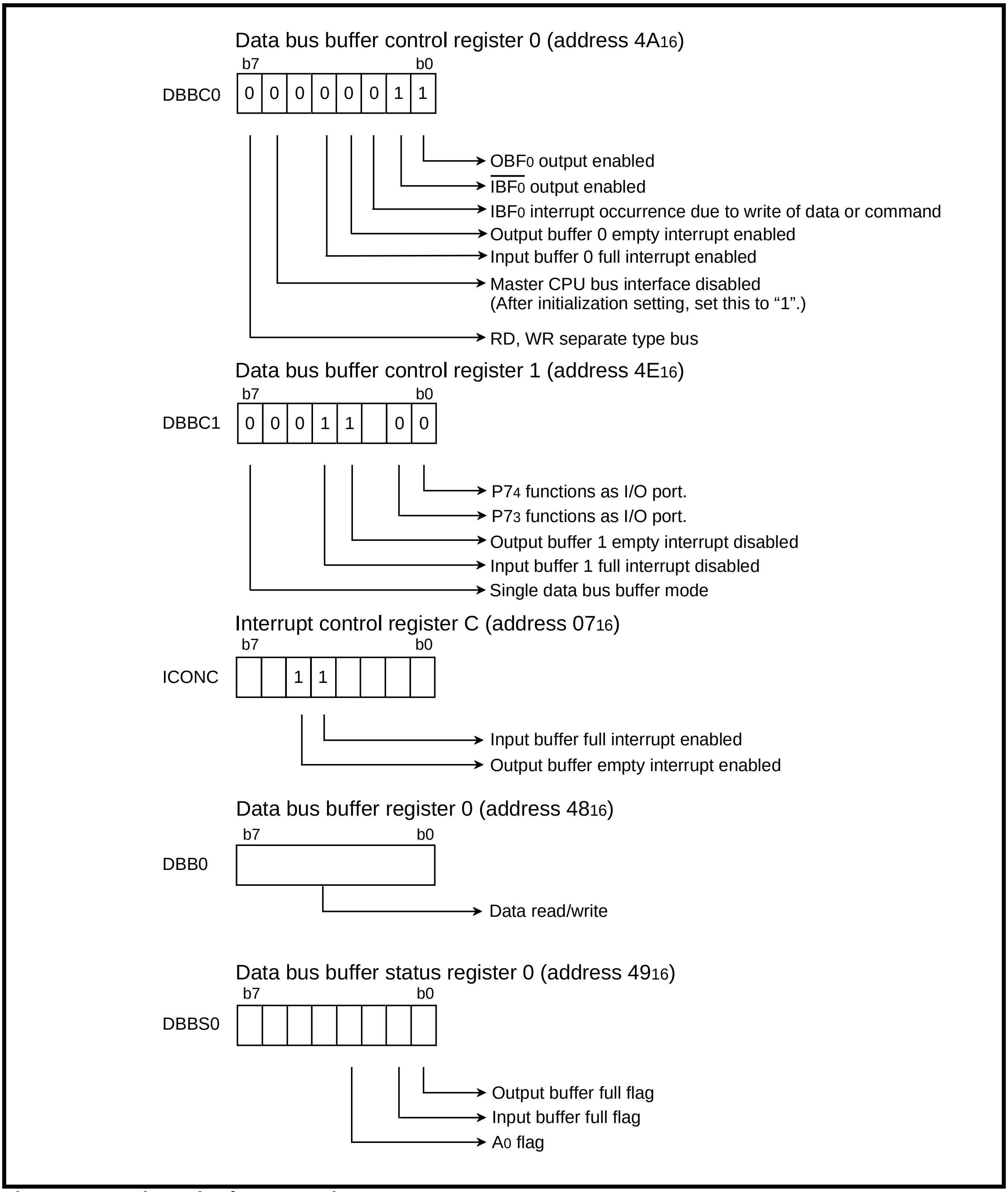


Fig. 2.8.7 Setting of relevant registers

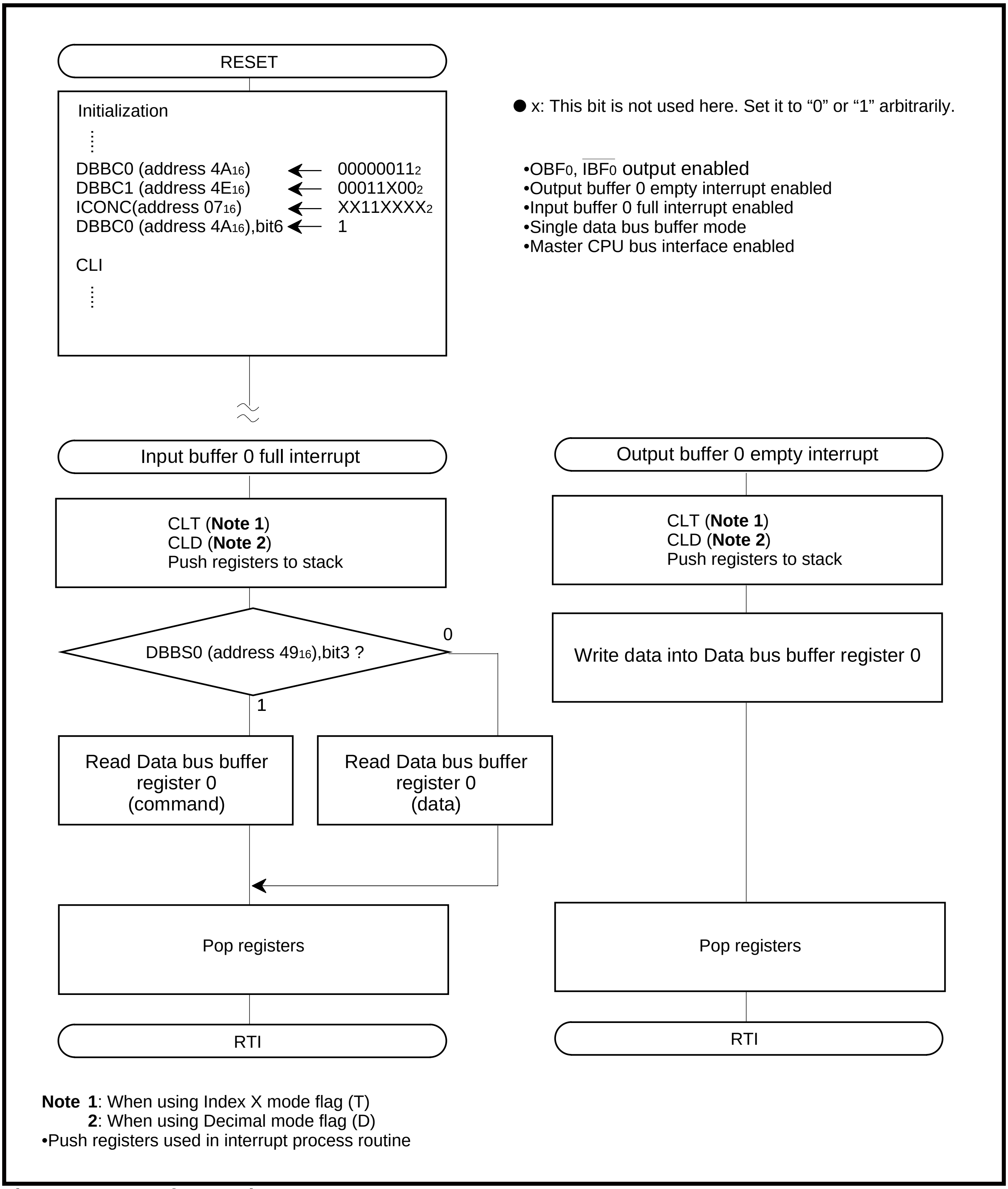


Fig. 2.8.8 Control procedure

2.8.6 Notes on master CPU bus interface

Be sure to set port P6 to input mode by setting the port P6 direction register to "0" when the master CPU bus interface is enabled.

2.9 Special count source generator (SCSG)

This paragraph explains the registers setting method and the notes related to the special count source generator (SCSG).

2.9.1 Memory map

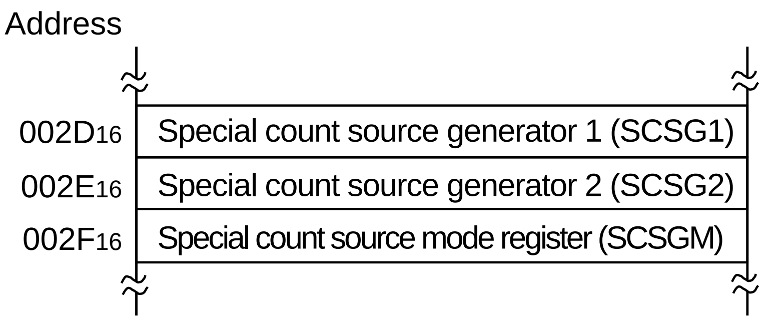


Fig. 2.9.1 Memory map of registers related to special count source generator

2.9.2 Related registers

Special count source generator 1

b7 b6 b5 b4 b3 b2 b1 b0

Special count source generator 1
(SCSG1: address $2D_{16}$)

b	Functions	At reset	R	W
0	●This is a 8-bit down-count timer. The division ratio : 1 / (n1 +1), n1: value set to SCSG1.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Fig. 2.9.2 Structure of Special count source generator 1

Special count source generator 2

b7 b6 b5 b4 b3 b2 b1 b0

Special count source generator 2
(SCSG2: address $2E_{16}$)

b	Functions	At reset	R	W
0	●This is a 8-bit down-count timer. An output from SCSG1 is divided by the contents of SCSG2 to generate SCSGCLK. ●The count source, an output from SCSG1, is $\phi \bullet$ n1 / (n1 +1). ●SCSGCLK frequency: $\phi \bullet$ n1 / (n1 +1) • 1 / (n2 +1) n1: value set to SCSG1 n2: value set to SCSG2	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Fig. 2.9.3 Structure of Special count source generator 2

Special count source mode register

b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0				

Special count source mode register
(SCSGM : address $2F_{16}$)

b	Name	Functions	At reset	R	W
0	SCSG1 data write control bit	0 : Writing data into both timer latch and timer simultaneously 1 : Writing data into timer latch	0	○	○
1	SCSG1 count stop bit	0 : Count start 1 : Count stop	0	○	○
2	SCSG2 data write control bit	0 : Writing data into both timer latch and timer simultaneously 1 : Writing data into timer latch	0	○	○
3	SCSGCLK output control bit	0 : SCSGCLK output disabled (SCSG1 and SCSG2 counts stop) 1 : SCSGCLK output enabled	0	○	○
4	Fix these bits to “0”.		0	○	○
5			0	○	○
6			0	○	○
7			0	○	○

Fig. 2.9.4 Structure of Special count source mode register

2.9.3 Functional description

The special count source generator, SCSG, consists of two 8-bit timers: SCSG1 and SCSG2. The output of the special count source generator can be used as a clock source for the timer X, serial I/O and two UARTs (UART1 and UART2).

(1) SCSG1

The SCSG1 is a 8-bit down-count timer. An output from SCSG1 is used as an count source of SCSG2. The division ratio of SCSG1 is given by 1 / (n1 + 1) and the count source of SCSG2 becomes $\phi \times \{n1 / (n1 + 1)\}$.
n1: value set to the SCSG1
Writing data to the SCSG1 can be performed anytime. When the SCSG1 data write control bit is set to “0” and data is written to the SCSG1, the data is written to the latch and timer at the same time. When that bit is set to “1”, the data is only written to the latch. When the count reaches “0”, an underflow occurs at the next count source rising edge and the contents of the timer latch are loaded to the timer.
If the SCSG1 count stop bit is set to “1” or the SCSG1 set value is “0”, this count source ϕ is not divided, so that the count source for SCSG2 becomes ϕ.

(2) SCSG2

The SCSG2 is a 8-bit down-count timer. This uses an output from SCSG1 as an count source.
The SCSG2 output is Clock SCSGCLK. The frequency is calculated as follows:

$$SCSGCLK = \phi \times \{n1 / (n1+1)\} \times \{1 / (n2+1)\}$$

n1: value set to SCSG1
n2: value set to SCSG2

Writing data to the SCSG2 can be performed anytime. When the SCSG2 data write control bit is set to “0” and data is written to the SCSG2, the data is written to the latch and timer at the same time. When that bit is set to “1”, the data is only written to the latch. When the count reaches “0”, an underflow occurs at the next count source rising edge and the contents of the timer latch are loaded to the timer.

If the SCSGCLK output control bit is set to “0”, the SCSG1 and SCSG2 stop their counts and SCSGCLK output is disabled. To use the SCSGCLK, set this bit to “1”.

2.10 External devices connection

This paragraph explains the registers setting method and the notes related to the external devices connection.

2.10.1 Memory map

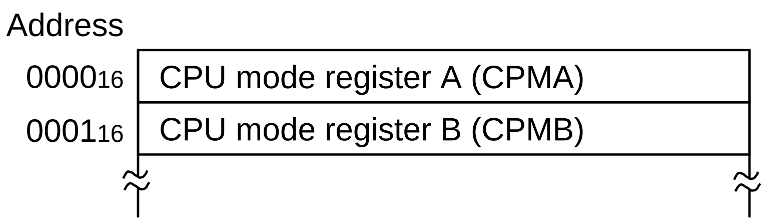


Fig. 2.10.1 Memory map of registers related to external devices connection

2.10.2 Related registers

CPU mode register A

b7 b6 b5 b4 b3 b2 b1 b0: [ ][ ][ ][ ][1][ ][ ][ ]

CPU mode register A
(CPMA : address 00_{16})

b	Name	Functions	At reset	R	W
0	Processor mode bits	b1b0 0 0 : Single-chip mode 0 1 : Memory expansion mode 1 0 : Microprocessor mode (**Note 1**) 1 1 : Not available	0	O	O
1			0	O	O
2	Stack page select bit	0 : Page 0 1 : Page 1	1	O	O
3	Fix this bit to “1”.		1	O	O
4	Sub-clock (X_{CIN}-X_{COUT}) stop bit	0 : Stopped 1 : Oscillating	0	O	O
5	Main clock (X_{IN}-X_{OUT}) stop bit	0 : Oscillating 1 : Stopped	0	O	O
6	Internal system clock select bit (**Note 2**)	0 : External clock (X_{IN}-X_{OUT} or X_{CIN}-X_{COUT}) 1 : f_{syn}	0	O	O
7	External clock select bit	0 : X_{IN}-X_{OUT} 1 : X_{CIN}-X_{COUT}	0	O	O

Notes 1: This is not available in the flash memory version.
2: When (CPMA 7, 6) = (0, 0), the internal system clock can be selected between f(X_{IN}) and f(X_{IN})/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

Fig. 2.10.2 Structure of CPU mode register A

CPU mode register B

b7 b6 b5 b4 b3 b2 b1 b0: [1][0][ ][ ][ ][ ][ ][ ]

CPU mode register B
(CPMB : address 01_{16})

b	Name	Functions	At reset	R	W
0	Slow memory wait select bits	b1b0 0 0 : No wait 0 1 : One-time wait 1 0 : Two-time wait 1 1 : Three-time wait	1	O	O
1			1	O	O
2	Slow memory wait mode select bits	b3b2 0 0 : Software wait 0 1 : Not available 1 0 : RDY wait 1 1 : Software wait plus RDY input anytime wait anytime wait	0	O	O
3			0	O	O
4	Expanded data memory access bit	0 : $\overline{\text{EDMA}}$ output disabled 1 : EDMA output enabled	0	O	O
5	HOLD function enable bit	0 : $\overline{\text{HOLD}}$ function disabled 1 : $\overline{\text{HOLD}}$ function enabled	0	O	O
6	Fix this bit to “0”.		0	O	O
7	Fix this bit to “1”.		1	O	O

Fig. 2.10.3 Structure of CPU mode register B

2.10.3 Functional description

This MCU starts its operation in the single-chip mode just after reset.

(1) Memory expansion mode

This mode is selected by setting “01” to the processor mode bits (b1, b0 of CPMA) in software with CNV_{SS} connected to V_{SS}.
In this mode, the ports function as follows:
Ports P0 and P1 as address buses (AB_0 to AB_{15})
Port P2 as data buses (DB_0 to DB_7)
Ports $P3_3$ to $P3_7$ as DMA_{OUT}, ϕ_{OUT}, $SYNC_{OUT}$, $\overline{WR}$, $\overline{RD}$ pins respectively.

This mode enables external memory expansion while maintaining the validity of the internal ROM.

(2) Microprocessor mode

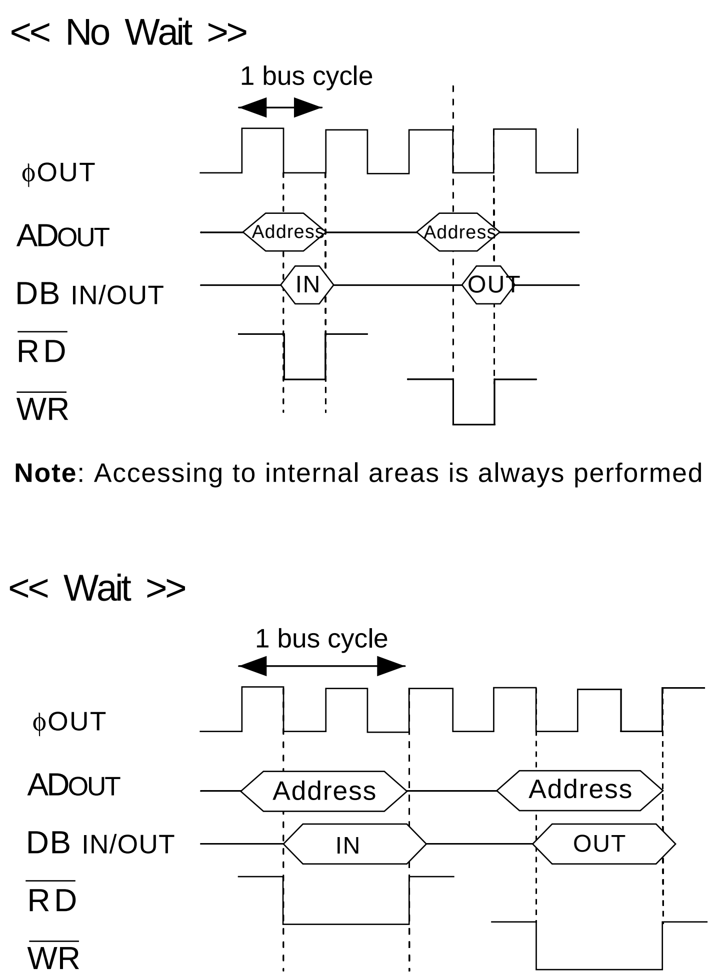
This mode is selected by resetting the MCU with CNV_{SS} connected to V_{CC}, or by setting “10” to the processor mode bits (b1, b0 of CPMA) in software with CNV_{SS} connected to V_{SS}. The function is the same as that of memory expansion mode.
In the microprocessor mode, the internal ROM is no longer valid and an external memory must be used.
Do not set this mode in the flash memory version.

2.10.4 Slow memory wait

The slow memory wait function is for easier interfacing with external devices that have long access times. This can be enabled in the memory expansion mode and microprocessor mode.
The wait is effective only to external areas. Access to internal area is always performed without wait.

(1) Software wait

The software wait is selected by setting “00” to the slow memory wait mode select bits (b3, b2 of CPMB).
Figure 2.10.4 shows the software wait timing example.



Note: Accessing to internal areas is always performed on this waveform.

Note: This example is the 1-cycle software wait.
Refer to Chapter 1 “Slow Memory Wait in PROCESSOR MODE” for 2- and 3-cycle software wait timings.

Fig. 2.10.4 Software wait timing example

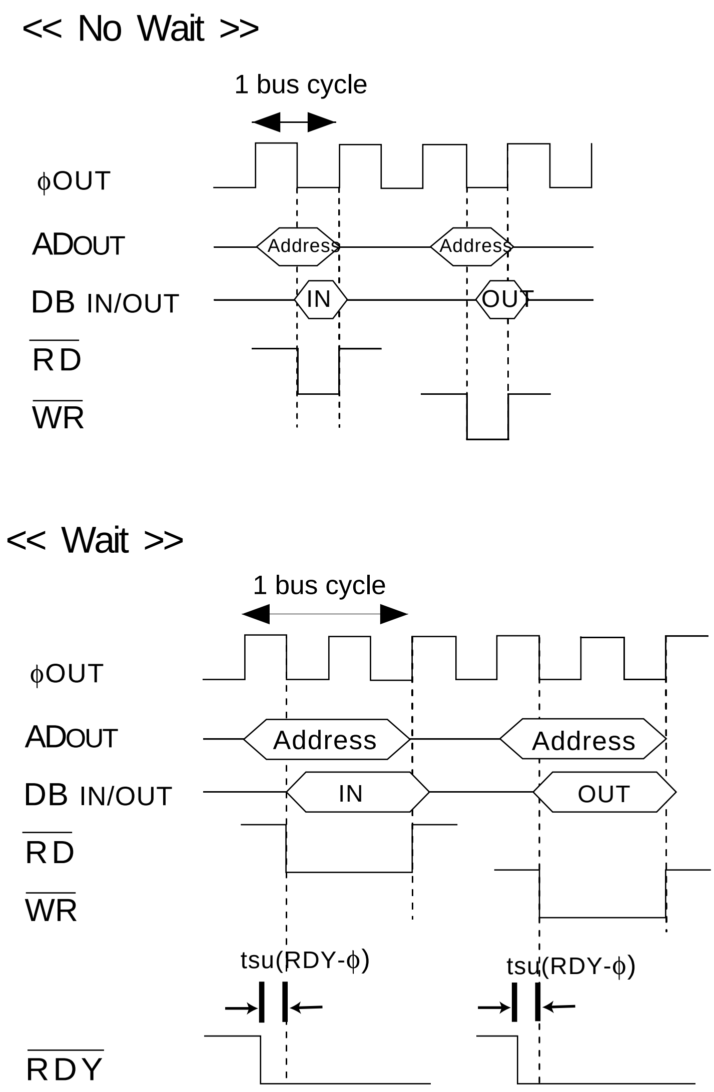
(2) RDY wait

RDY wait is selected by setting “10” to the slow memory wait mode select bits (b3, b2 of CPMB). When a fixed time of “L” (tsu) is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls), the MCU goes to the RDY state. Then the read/write cycle can be extended by one to three ϕ cycles. The number of ϕ cycles to be extended can be selected by the slow memory wait select bits (b1, b0 of CPMB).

Even if “L” is input to the RDY pin at the end of waited read/write cycle, the cycle is not extended.

When a fixed time of “H” (tsu) is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls), the MCU is released from the RDY state.

Figure 2.10.5 shows the RDY wait timing example.



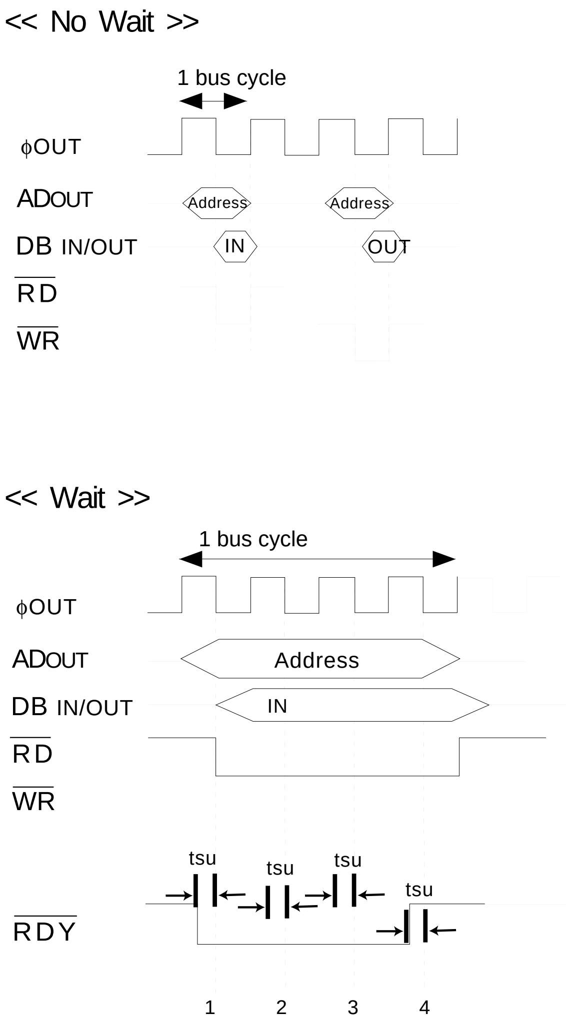
Note: This example is the 1-cycle RDY wait .
Refer to Chapter 1 “Slow Memory Wait in PROCESSOR MODE” for 2- and 3-cycle RDY wait timings.

Fig. 2.10.5 RDY wait timing example

(3) Software wait + Extended RDY wait

Extended RDY wait (software wait plus RDY input anytime wait) is selected by setting “11” to the slow memory wait mode select bits (b3, b2 of CPMB). The read/write cycle can be extended when a fixed time (tsu) of “L” is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls). The RDY pin state is checked continually at each fall of ϕ cycle until the RDY pin goes to “H” and the cycle keeps being extended. When a fixed time of “H” (tsu) is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls), the MCU is released from the wait within 1, 2, or 3 ϕ cycles as selected with the slow memory wait bits (b1, b0 of CPMB).

Figure 2.10.6 shows the extended RDY wait (software wait plus RDY input anytime wait) timing example



1. When a fixed time (tsu) of “L” is input to the RDY pin before ϕ falls, the MCU goes to RDY state.
2. The RDY pin state is checked continually at each fall of ϕ cycle.
3. The RDY pin is “L” at the ϕ fall of the end of the current wait time, so that the read/write cycle is extended for one wait once again.
4. When a fixed time (tsu) of “H” is input to the RDY pin before ϕ falls, the RDY state is released after the current wait time.

Note: This example is the 1-cycle extended RDY wait (RD).
The 1-cycle extended RDY wait (WR) is the same timing as this.
Refer to Chapter 1 “Slow Memory Wait in PROCESSOR MODE” for 2- and 3-cycle extended RDY wait timings.

Fig. 2.10.6 Extended RDY wait (software wait plus RDY input anytime wait) timing example

2.10.5 HOLD function

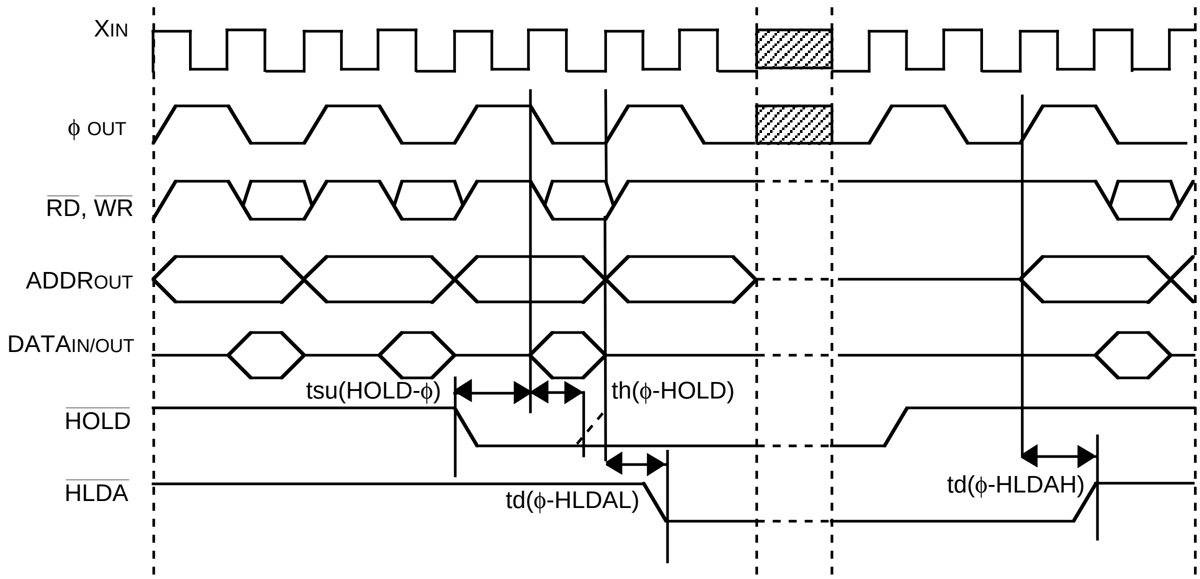
The HOLD function is used for systems that consist of external circuits that access MCU buses without use of the CPU (Central Processing Unit). The HOLD function is used to generate the timing in which the MCU will relinquish the bus from the CPU to the external circuits. To use the HOLD function, set the HOLD function enable bit (b5 of CPMB) to “1”. This can be enabled in the memory expansion mode and microprocessor mode.

When “L” level is input to the $\overline{HOLD}$ pin, the MCU goes to the HOLD state and remains so while the pin is at “L”. When the MCU relinquishes use of the bus, “L” level is output from the $\overline{HLDA}$ pin. The MCU puts ports P0 and P1 (address buses) and port P2 (data bus) to tri-state outputs and holds the $\overline{RD}$ pin ($P3_7$) and $\overline{WR}$ pin ($P3_6$) “H” level. This will prevent incorrect operations of external devices.

Though the clock supply to the CPU halts in HOLD state, the internal peripheral clocks and ϕ_{OUT} ($P3_4$) continues to be supplied.

When “H” level is input to the $\overline{HOLD}$ pin,“H” level is output from the $\overline{HLDA}$ pin and the MCU can use address buses, data bus, signals $\overline{RD}$ and $\overline{WR}$ to access to external devices.

Figure 2.10.7 shows the Hold function timing diagram.



Note: This diagram assumes $\phi = X_{IN}/2$.

Fig. 2.10.7 Hold function timing diagram

2.10.6 Expanded data memory access

In Expanded Data Memory Access Mode (EDMA mode), the MCU can access a data area larger than 64 Kbytes with the LDA ($zz), Y (indirect Y) instruction and the STA ($zz), Y (indirect Y) instruction. It is only able to store and read data for the expanded data memory area. The access can be performed with T flag = "0" or "1". (T flag is Index X mode flag.)

To use this mode, set the expanded data memory access bit (b4 of CPMB) to "1". In this case, $\overline{EDMA}$ pin (port $P4_0$) goes "L" level during the read/write cycle of the LDA or STA instruction. The determination of which bank to access is done by using an I/O port to represent expanded addresses exceeding address bus (AB_{15})16. This signal and the port output signal are put together, and it becomes the chip select (selecting a bank, expanded memory).

In EDMA mode, the area from addresses 0000_{16} to $FFFF_{16}$ can be accessed. When accessing a bank, follow this procedure (four banks are assumed):

- Bank specification (Data output to the port for a bank setup 16 (AB_{16}), and 16 (AB_{17})). The user must setup 16 (AB_{16}), and 16 (AB_{17}).
- Enabling $\overline{EDMA}$ output. (Set b4 of CPMB to "1".)
- Executing LDA ($zz), Y (indirect Y) (In the case of read data of expanded data memory)

The data of the address (bank 0) + Y stored in the internal RAM address ($zz) are loaded to the accumulator.

Figure 2.10.8 shows a connection example of memory access up to 256 Kbytes.

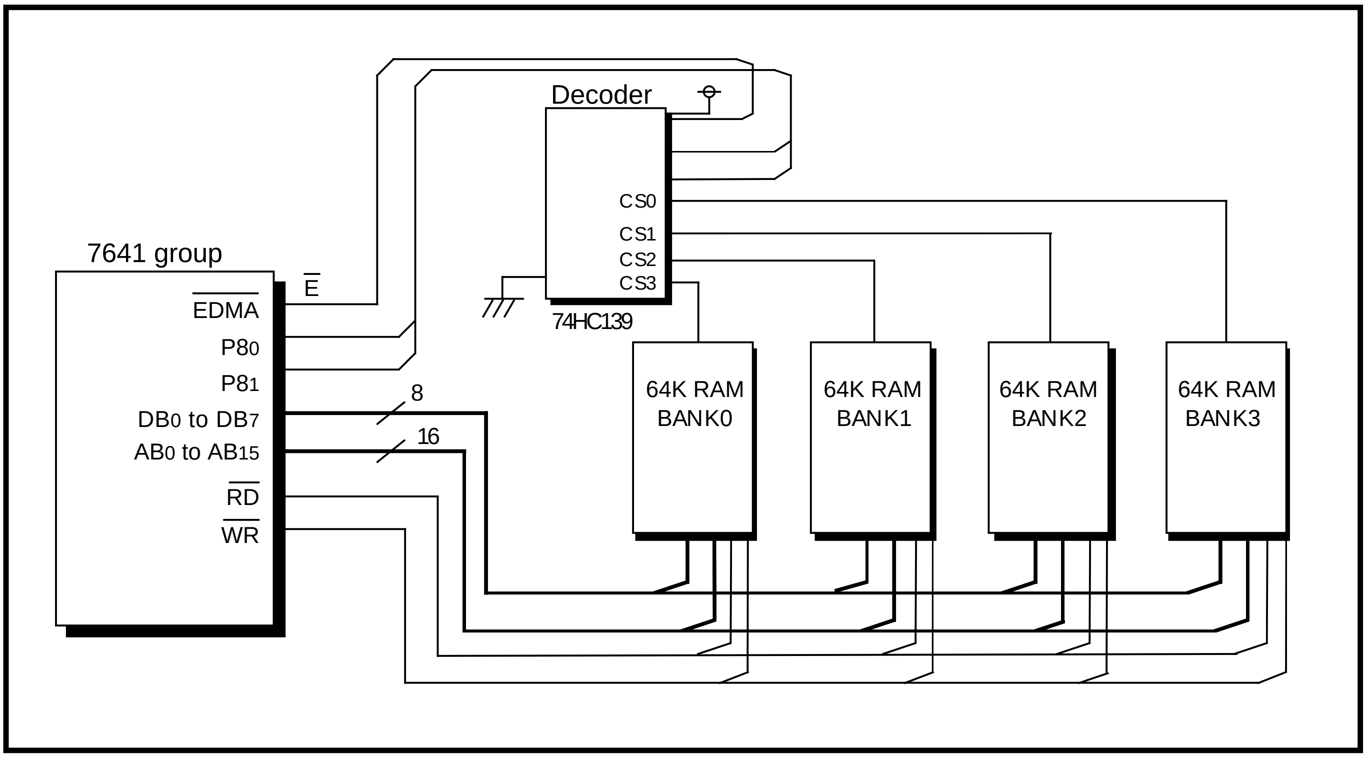


Fig. 2.10.8 Connection example of memory access up to 256 Kbytes

2.10.7 External devices connection example

Connection example for controlling external memory is shown bellow.

(1) External memory connection example : No Wait function

Outline: In microprocessor mode the external memory is accessed.

Figure 2.10.9 shows the external ROM and RAM example.

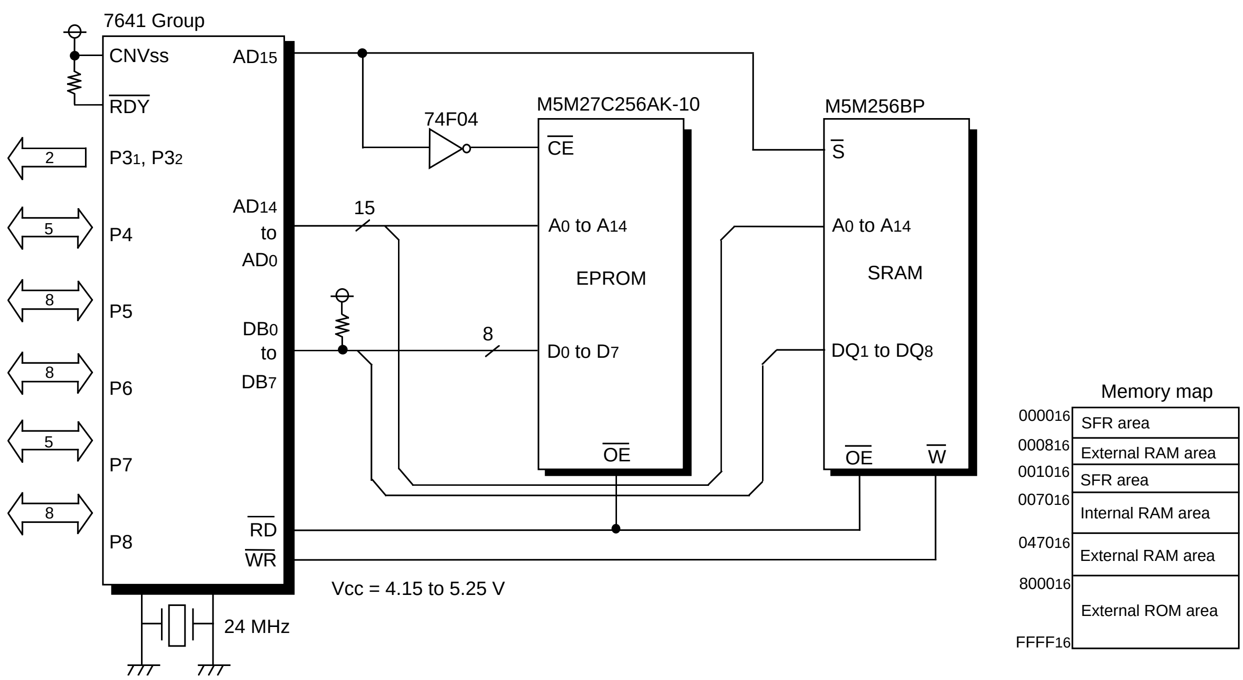


Fig. 2.10.9 External ROM and RAM example

(2) External memory connection example : RDY function (one wait) in use

Outline: RDY function is used because the external memory has a slow-memory-access speed.

Specifications:
- In read/write status of the CPU, input "L" to the RDY($P3_0$) pin will cause extension of 1 to 3 ϕ cycles on its read/write cycle.
- When the slow memory wait select bit (b1, b0) is = "01", it is extended by one cycle.
- RD/WR signal holds "L" level during the extended time.
- Usable RAM and ROM at $f(X_{IN})$ = 24 MHz, Vcc = 5 V are given bellow:
  - OE access time : ta(OE) ≤ 107 ns
  - Data setup time at write : tsu(D) ≤ 100 ns
  - Examples satisfying these specifications: M5M27C256AK-10 (EPROM), M5M5256BP-10 (SRAM)

Figure 2.10.10 shows the RDY function use example, and Figures 2.10.11 to 2.10.13 show the read and write cycles.

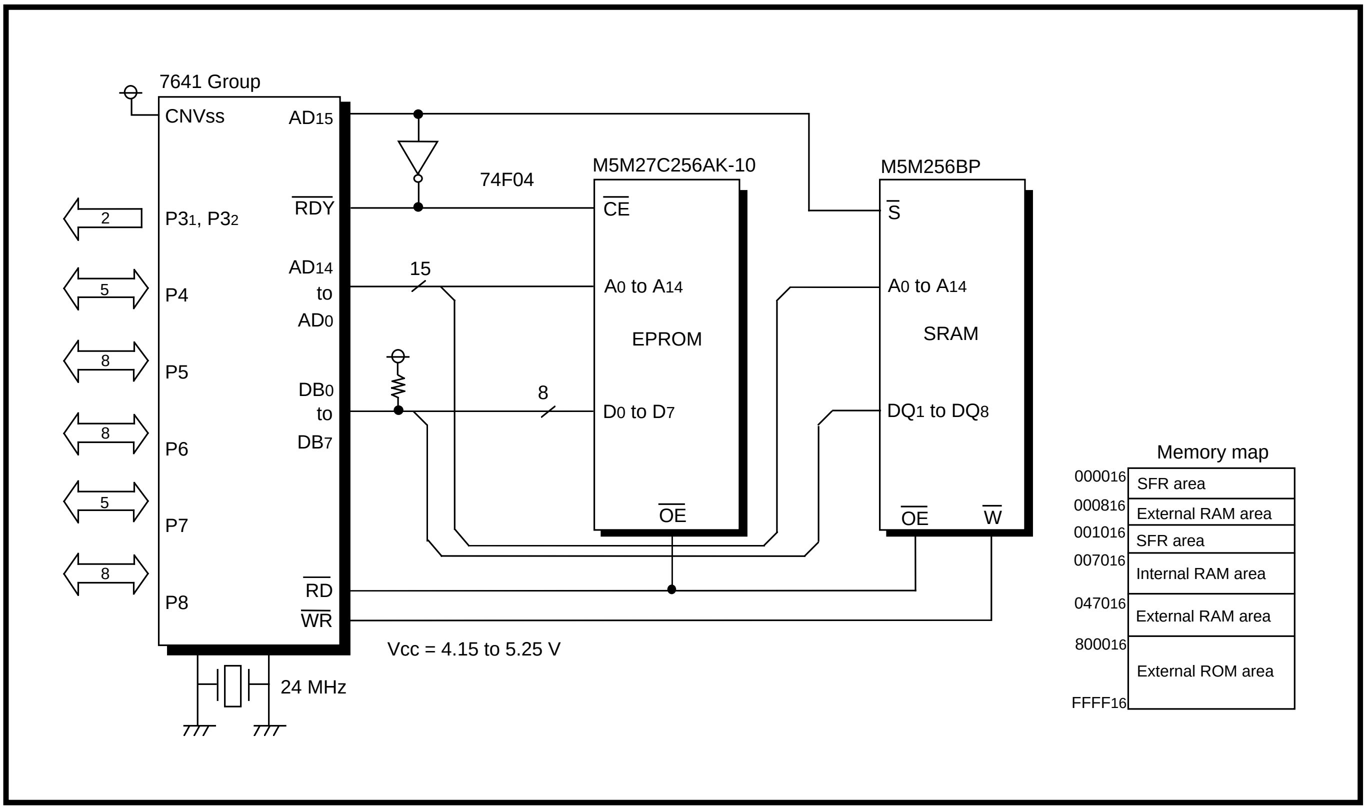


Fig. 2.10.10 RDY function use example

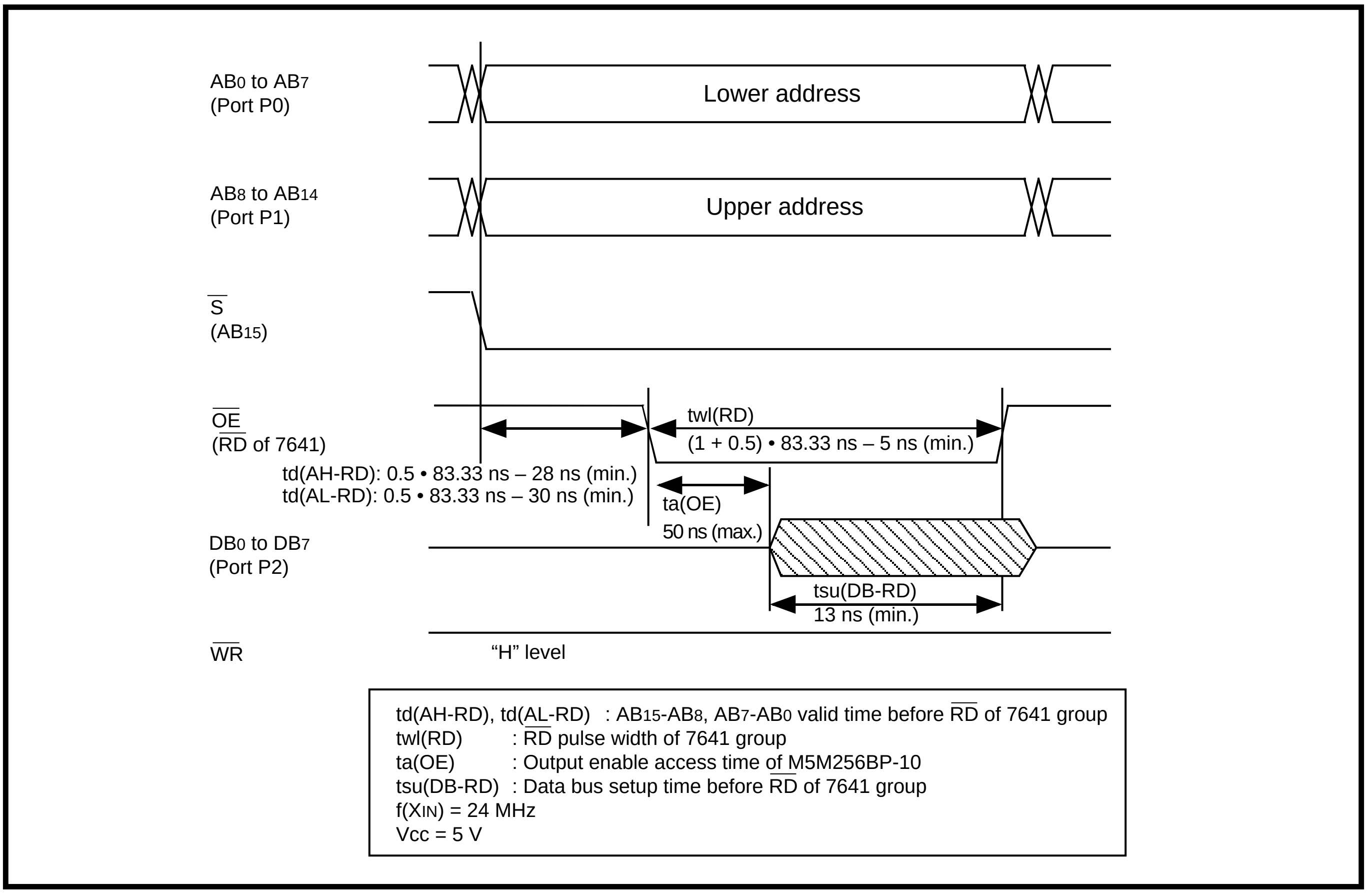


Fig. 2.10.11 Read cycle (OE access, SRAM)

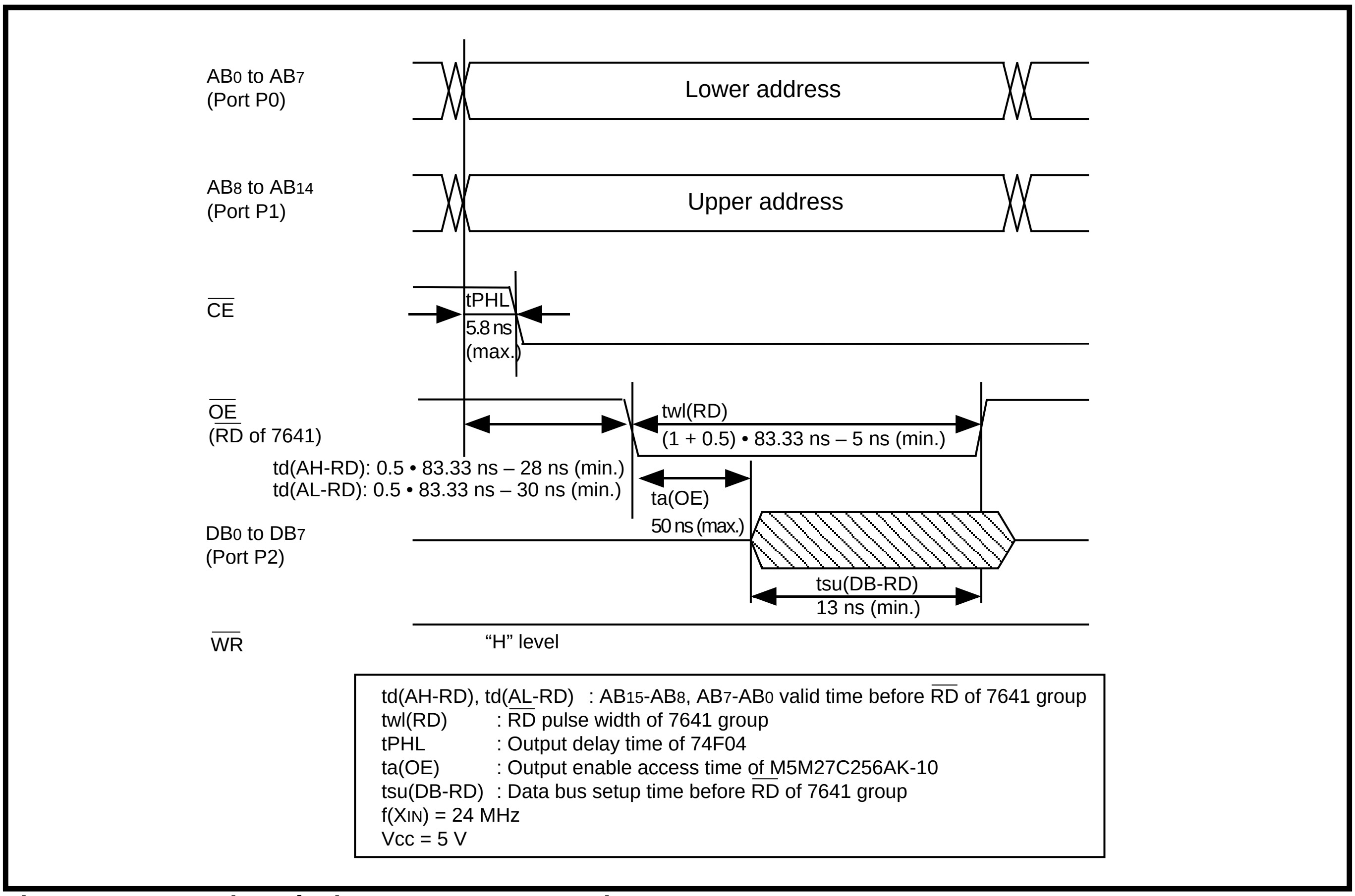


Fig. 2.10.12 Read cycle (OE access, EPROM)

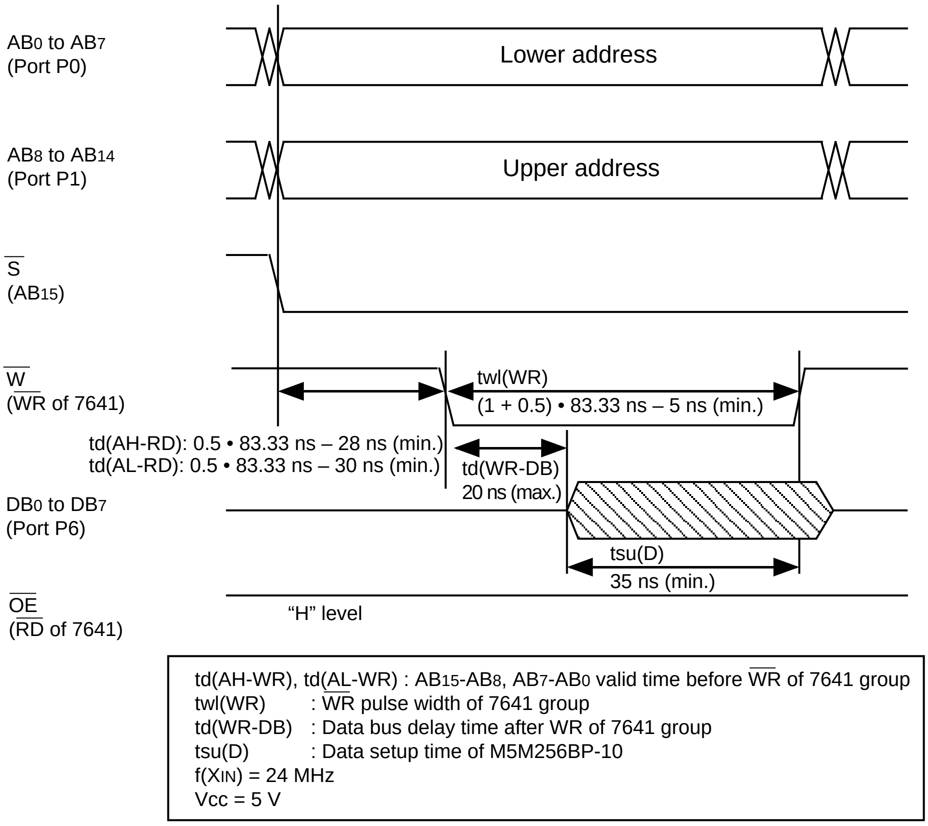


Fig. 2.10.13 Write cycle (W control, SRAM)

2.10.8 Notes on external devices connection

(1) Rewrite port P3 latch

In both memory expansion mode and microprocessor mode, ports $P3_1$ and $P3_2$ can be used as output ports. We recommend to use the **LDM** instruction or **STA** instruction to write to port P3 register (address $000E_{16}$). If using the Read-Modify-Write instruction (**SEB** instruction, **CLB** instruction) you will need to map a memory that the CPU can read from and write to.

[Reason]

The access to address $000E_{16}$ is performed:
•Read from external memory
•Write to both port P3 latch and external memory.
It is because address $000E_{16}$ is assigned on an external area In the memory expansion mode and microprocessor mode.
Accordingly, if a Read-Modify-Write instruction is executed to address $000E_{16}$, the external memory contents is read out and after its modification it will be written into both port P3 latch and an external memory. As a result, if an external memory is not allocated in address $000E_{16}$ then, the MCU will read an undefined value and write its modified value into the port P3 latch. Therefore port P3 latch value will become undefined.

(2) overlap of internal and external memories

In the memory expansion mode, if the internal and external memory areas overlap, the internal memory becomes the valid memory for the overlapping area. When the CPU performs a read or a write operation on this overlapped area, the following things happen:
•Read
The CPU reads out the data in the internal memory instead of in the external memory. Note that, since the CPU will output a proper read signal, address signal, etc., the memory data at the respective address will appear on the external data bus.
•Write
The CPU writes data to both the internal and external memories.

(3) $\overline{RD}$, $\overline{WR}$ pins

In the memory expansion mode or microprocessor mode, a read-out control signal is output from the $\overline{RD}$ pin ($P3_6$), and a write-in control signal is output from the $\overline{WR}$ pin ($P3_7$). “L” level is output from the $\overline{RD}$ pin at CPU read-out and from the $\overline{WR}$ pin at CPU write-in. These signals function for internal access and external access.

(4) $\overline{HLDA}$ pin

In spite of enabling the Hold function, the $\overline{HLDA}$ pin does not function when IBF_1 output is enabled in the master CPU bus interface.

(5) RDY function

When using RDY function in usual connection, it does not operate at 12 MHz of ϕ or faster.

[Reason]

td(ϕ-AH) + tsu(RDY-ϕ) = 31 ns (max.) + 21 ns (min.) = 52 ns.

twh (ϕ), twl (ϕ) = 0.5 × 83.33 – 5 = 36.665 ns

Therefore, it becomes 52 ns > 36.665 ns, so that the timing to enter RDY wait does not match.

However, if the timings can match owing to $\overline{\text{RDY}}$ pin by "L" fixation and others, the RDY function can be used even at ϕ = 12 MHz. In this situation the slow memory wait always functions.

(6) Wait function

The Wait function is serviceable at accessing an external memory in the memory expansion mode and microprocessor mode. However, in these modes even if an external memory is assigned to addresses 0008_{16} to $000F_{16}$, the Wait function cannot function to these areas.

(7) Processor mode switch

Note when the processor mode is switched by setting of the processor mode bits (b1, b0 of CPMA), that will immediately switch the accessible memory from external to internal or from internal to external. If this is done, the first cycle of the next instruction will be operated from the accidental memory.

To prevent this problem, follow the procedure below:

(a) Duplicate the next instruction at the same address both in internal and external memories.

(b) Switch from single-chip mode to memory expansion mode, jump to external memory, and then switch from memory expansion mode to microprocessor mode. (Because in general, the problem will not occur when switching the modes as long as the same memory is accessed after the switch.

(c) Load a simple program in RAM that switches the modes, jump to RAM and execute the program, then jump to the location of the code to run after the processor mode has switched.

2.11 Reset

2.11.1 Connection example of reset IC

Figure 2.11.1 shows the system example which switches to the RAM backup mode by detecting a drop of the system power source voltage with the INT interrupt.

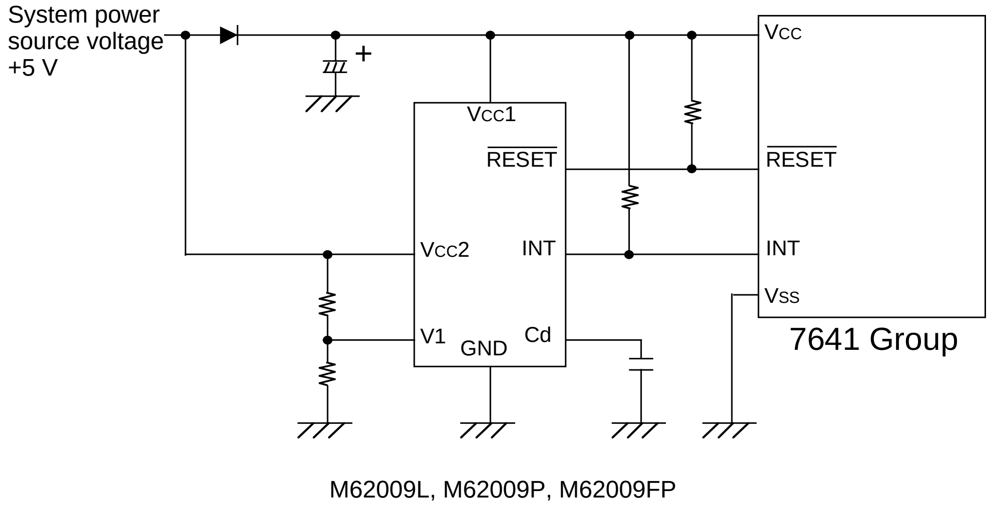


Fig. 2.11.1 RAM backup system

2.11.2 Notes on reset

(1) Connecting capacitor

In case where the RESET signal rise time is long, connect a ceramic capacitor or others across the RESET pin and the Vss pin. Use a 1000 pF or more capacitor for high frequency use. When connecting the capacitor, note the following :

- Make the length of the wiring which is connected to a capacitor as short as possible.
- Be sure to verify the operation of application products on the user side.

● Reason

If the several nanosecond or several ten nanosecond impulse noise enters the RESET pin, it may cause a microcomputer failure.

2.12 Clock generating circuit

This paragraph explains the registers setting method and the notes related to the clock generating circuit. Besides, two modes to realize less power dissipation due to the CPU and some peripherals halted are explained: Stop mode due to STP instruction, Wait mode due to WIT instruction.

2.12.1 Memory map

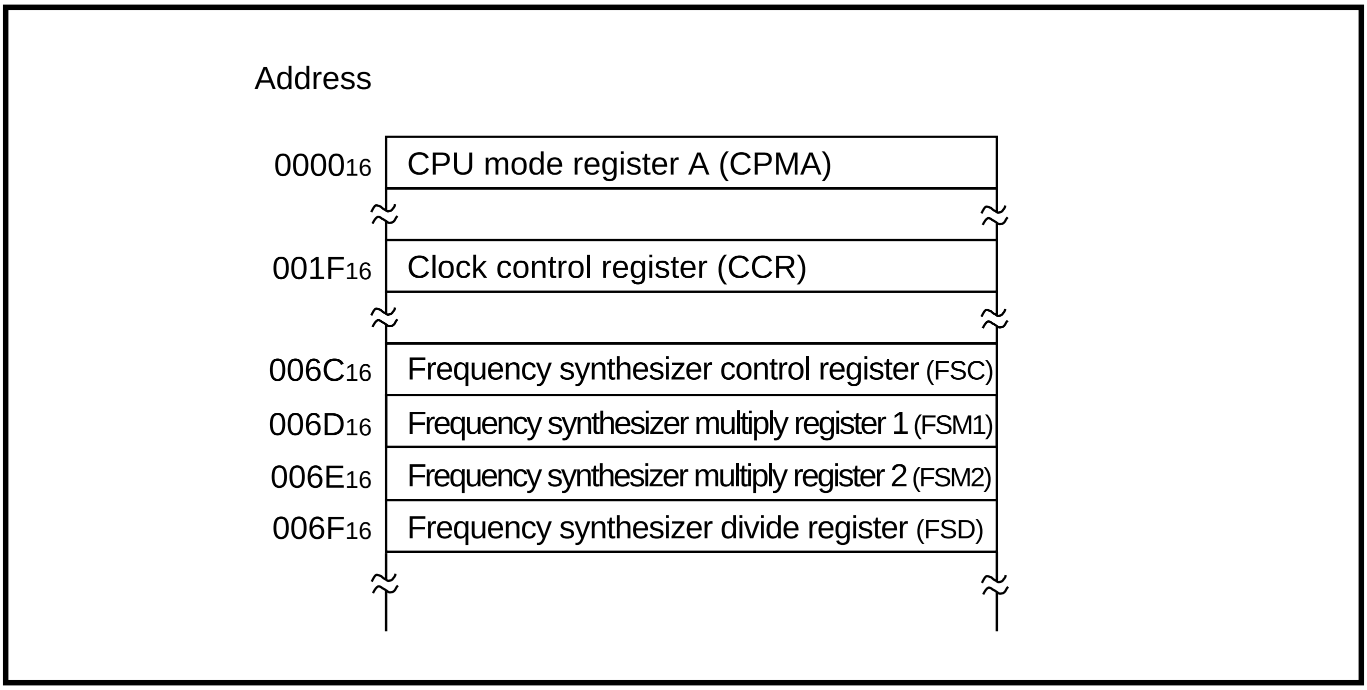


Fig. 2.12.1 Memory map of registers related to clock generating circuit

2.12.2 Related registers

CPU mode register A

b7 b6 b5 b4 b3 b2 b1 b0
[ ][ ][ ][ ][1][ ][ ][ ]

CPU mode register A
(CPMA : address 00_{16})

b	Name	Functions	At reset	R	W
0	Processor mode bits	b1b0 0 0 : Single-chip mode 0 1 : Memory expansion mode 1 0 : Microprocessor mode (**Note 1**) 1 1 : Not available	0	O	O
1			0	O	O
2	Stack page select bit	0 : Page 0 1 : Page 1	1	O	O
3	Fix this bit to “1”.		1	O	O
4	Sub-clock (XCIN-XCOUT) stop bit	0 : Stopped 1 : Oscillating	0	O	O
5	Main clock (XIN-XOUT) stop bit	0 : Oscillating 1 : Stopped	0	O	O
6	Internal system clock select bit (**Note 2**)	0 : External clock (XIN-XOUT or XCIN-XCOUT) 1 : fsyn	0	O	O
7	External clock select bit	0 : XIN-XOUT 1 : XCIN-XCOUT	0	O	O

Notes 1: This is not available in the flash memory version.
2: When (CPMA 7, 6) = (0, 0), the internal system clock can be selected between f(XIN) and f(XIN)/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

Fig. 2.12.2 Structure of CPU mode register A

Clock control register

b7 b6 b5 b4 b3 b2 b1 b0
[ ][ ][ ][0][0][0][0][0]

Clock control register
(CCR : address $1F_{16}$)

b	Name	Functions	At reset	R	W
0	Fix these bits to “0”.		0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4			0	O	O
5	XCOUT oscillation drive disable bit (CCR5)	0 : XCOUT oscillation drive is enabled. (When XCIN oscillation is enabled.) 1 : XCOUT oscillation drive is disabled.	0	O	O
6	XOUT oscillation drive disable bit (CCR6)	0 : XOUT oscillation drive is enabled. (When XIN oscillation is enabled.) 1 : XOUT oscillation drive is disabled.	0	O	O
7	XIN divider select bit (CCR7) (**Note**)	0 : f(XIN)/2 is used for the system clock. 1 : f(XIN) is used for the system clock.	0	O	O

Note: This bit is valid when (b7, b6 of CPMA) = “00”.

Fig. 2.12.3 Structure of Clock control register

Frequency synthesizer control register

b7 b6 b5 b4 b3 b2 b1 b0: [ ][ ][ ][0][ ][0][0][ ]

Frequency synthesizer control register
(FSC : address $6C_{16}$)

b	Name	Functions	At reset	R	W
0	Frequency synthesizer enable bit (FSE)	0 : Disabled 1 : Enabled	0	O	O
1	Fix these bits to “0”.		0	O	O
2			0	O	O
3	Frequency synthesizer input bit (FIN)	0 : $f(X_{IN})$ 1 : $f(X_{CIN})$	0	O	O
4	Fix this bit to “0”.		0	O	O
5	LPF current control bit (CHG1, CHG0) (**Note**)	b1b0 0 0 : Not available 0 1 : Low current 1 0 : Intermediate current (recommended) 1 1 : High current	1	O	O
6			1	O	O
7	Frequency synthesizer lock status bit	0 : Unlocked 1 : Locked	0	O	O

Note: Bits 6 and 5 are set to (bit 6, bit 5) = (1, 1) at reset.
When using the frequency synthesizer, we recommend to set to (bit 6, bit 5) = (1, 0) after locking the frequency synthesizer.

Fig. 2.12.4 Structure of Frequency synthesizer control register

Frequency synthesizer multiply register 1

b7 b6 b5 b4 b3 b2 b1 b0: [ ][ ][ ][ ][ ][ ][ ][ ]

Frequency synthesizer multiply register 1
(FSM1: address $6D_{16}$)

b	Functions	At reset	R	W
0	●f_{VCO} clock is generated by multiplying f_{PIN} clock, which is generated by FSM2, by the contents of this register: $f_{VCO} = f_{PIN} \cdot \{2(n+1)\}$, n: value set to FSM1.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Fig. 2.12.5 Structure of Frequency synthesizer multiply register 1

Frequency synthesizer multiply register 2

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer multiply register 2
(FSM2: address $6E_{16}$)

b	Functions	At reset	R	W
0	●f_{PIN} clock is generated by dividing f_{IN} clock by the contents of this register. Either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer is selectable. $f_{PIN} = f_{IN} / \{2(n+1)\}$, n: value set to FSM2	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 2.12.6 Structure of Frequency synthesizer multiply register 2

Frequency synthesizer divide register

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer divide register
(FSD: address $6F_{16}$)

b	Functions	At reset	R	W
0	●f_{SYN} clock is generated by dividing f_{VCO} clock by the contents of this register: $f_{SYN} = f_{VCO} / \{2(m+1)\}$, m: value set to FSD	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 2.12.7 Structure of Frequency synthesizer divide register

2.12.3 Stop mode

The Stop mode is set by executing the STP instruction. In Stop mode, the oscillation of both clocks (X_{IN}–X_{OUT}, X_{CIN}–X_{COUT}) stop and the internal clock ϕ stops at the "H" level. The CPU stops and peripheral units stop operating. As a result, power dissipation is reduced.

(1) State in Stop mode

Table 2.12.1 shows the state in Stop mode.

Table 2.12.1 State in Stop mode

Item	State in Stop mode
Oscillation	Stopped.
CPU	Stopped.
Internal clock ϕ	Stopped at "H" level.
I/O ports	Retains the state at the STP instruction execution.
Timer	When using internal count source: Stopped.
	When using external count source: Operating.
UART	Stopped.
DMAC	Stopped.
Serial I/O	When using internal syncronous clock: Stopped.
	When using external syncronous clock: Operating.
USB	Stopped.
RAM	Retained.
SFR	Retained (except for Timer 1, Timer 2).
CPU registers	Retained: Accumulator, Index register X, Index register Y, Stack pointer, Program counter, Processor status register.

(2) Release of Stop mode

The Stop mode is released by a reset input or by the occurrence of an interrupt request.
These interrupt sources can be used for restoration:
•INT_0, INT_1
•$CNTR_0$, $CNTR_1$
•Timers X, Y using an external count source
•Serial I/Os using an external clock
•Key-on wake-up
•USB function resume
However, when using any of these interrupt requests for restoration from Stop mode, <u>in order to enable the selected interrupt, set the following conditions before execution of STP instruction.</u>
[Necessary register setting]
① Timer 1 interrupt enable bit (b6 of ICONB) = "0" (interrupt disabled)
② Timer 2 interrupt enable bit (b7 of ICONB) = "0" (interrupt disabled)
③ Timer 1 interrupt request bit (b6 of IREQB) = "0" (no interrupt request issued)
④ Timer 2 interrupt request bit (b7 of IREQB) = "0" (no interrupt request issued)
⑤ Interrupt request bit of interrupt source to be used for restoration = "0" (no interrupt request issued)
⑥ Interrupt enable bit of interrupt source to be used for restoration = "1" (interrupts enabled)
⑦ Interrupt disable flag I = "0" (interrupt enabled)

(3) Notes on STP instruction

•Execution of STP instruction clears the timer 123 mode register (address 29_{16}) except bit 4 to "0".
•When using f_{SYN} as the internal system clock, switch to f(X_{IN}) or f(X_{CIN}) before execution of STP instruction.
•Execution of STP instruction clears bit 7 of clock control register to "0" (f(X_{IN})/2).

2.12.4 Wait mode

The Wait mode is set by execution of the WIT instruction. In Wait mode, oscillation continues, but the internal clock ϕ stops at the "H" level.
The CPU stops, but most of the peripheral units continue operating.

(1) State in Wait mode

Table 2.12.2 shows the state in Wait mode.

Table 2.12.2 State in Wait mode

Item	State in Wait mode
Oscillation	Oparating.
CPU	Stopped.
Internal clock ϕ	Stopped at "H" level.
I/O ports	Retains the state at the WIT instruction execution.
Timer	Operating.
UART	Operating.
DMAC	Stopped.
Serial I/O	Operating.
USB	Operating.
RAM	Retained.
SFR	Retained.
CPU registers	Retained: Accumulator, Index register X, Index register Y, Stack pointer, Program counter, Processor status register.

(2) Release of wait mode

The Wait mode is released by reset input or by the occurrence of an interrupt request.
In Wait mode oscillation is continued, so that an instruction can be executed immediately after the Wait mode is released.
These interrupt sources can be used for restoration:
•INT_0, INT_1
•$CNTR_0$, $CNTR_1$
•Timers
•Serial I/Os
•UART
•DMAC
•Key-on wake-up
•Master CPU bus interface
•USB function
•USB SOF

However, when using any of these interrupt requests for restoration from Stop mode, in order to enable the selected interrupt, set the following conditions before execution of WIT instruction.

[Necessary register setting]
① Interrupt request bit of interrupt source to be used for restoration = "0" (no interrupt request issued)
② Interrupt enable bit of interrupt source to be used for restoration = "1" (interrupts enabled)
③ Interrupt disable flag I = "0" (interrupt enabled)

2.12.5 Clock generating circuit application examples

(1) Status transition during power failure

Outline: The clock is counted up every one second by using the timer interrupt during a power failure.

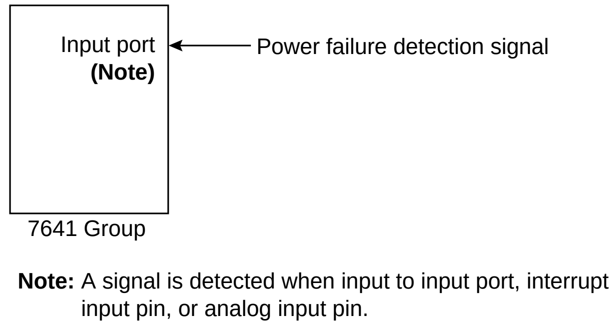


Fig. 2.12.8 Connection diagram

Specifications: •Reducing power dissipation as low as possible while maintaining clock function

•Clock: $f(X_{IN})$ = 4.19 MHz, $f(X_{CIN})$ = 32.768 kHz

•Port processing

Input port: Fixed to "H" or "L" level on the external

Output port: Fixed to output level that does not cause current flow to the external

(Example) When a circuit turns on LED at "L" output level, fix the output level to "H".

I/O port: Input port → Fixed to "H" or "L" level on the external

Output port → Output of data that does not consume current

Figure 2.12.9 shows the status transition diagram during power failure and Figure 2.12.10 shows the setting of relevant registers.

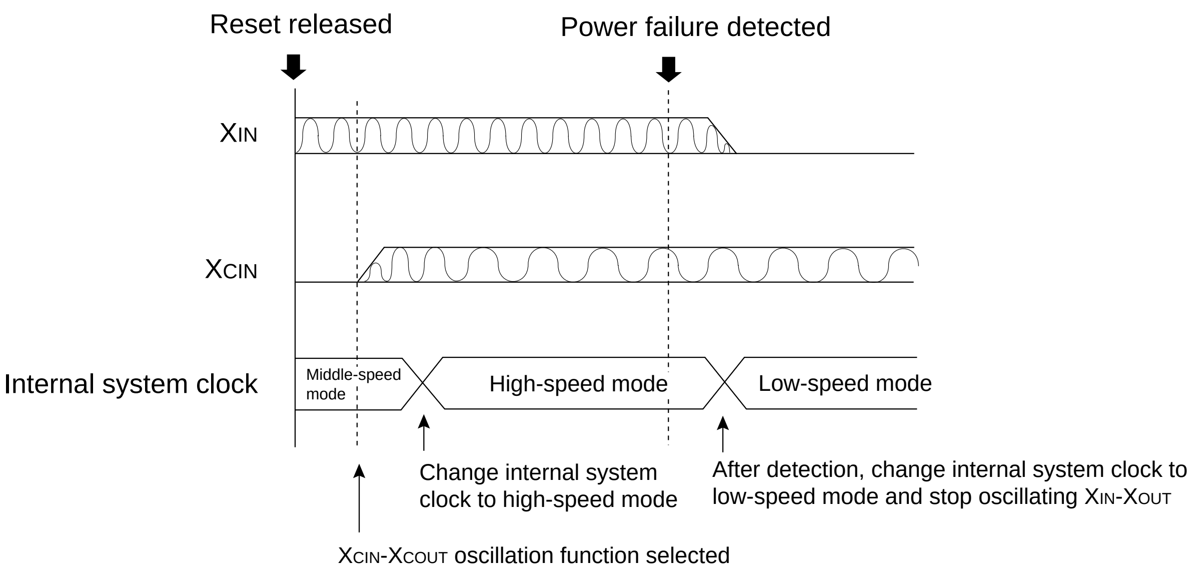


Fig. 2.12.9 Status transition diagram during power failure

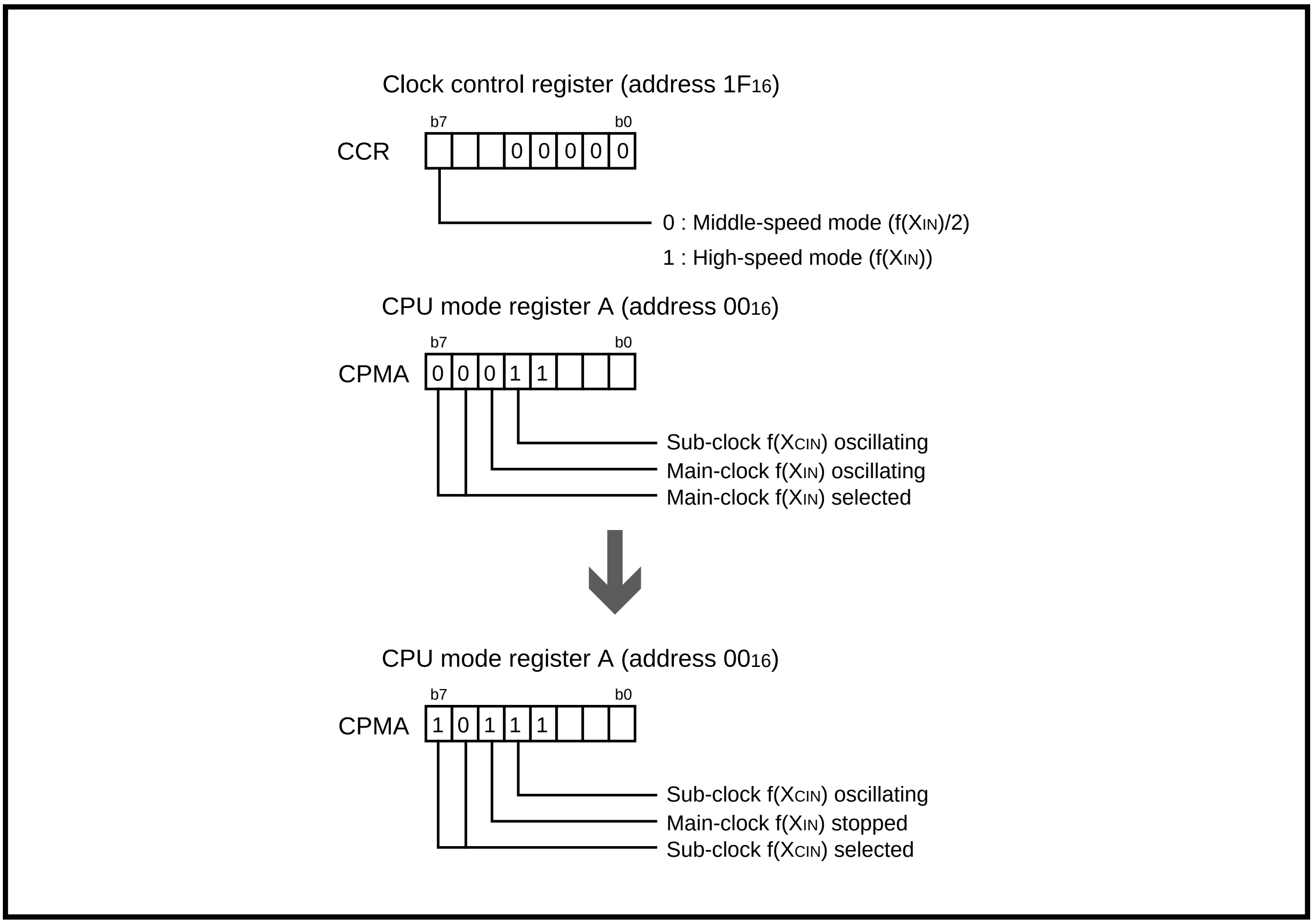


Fig. 2.12.10 Setting of relevant registers

Control procedure: Set the relevant registers in the order shown below to prepare for a power failure.

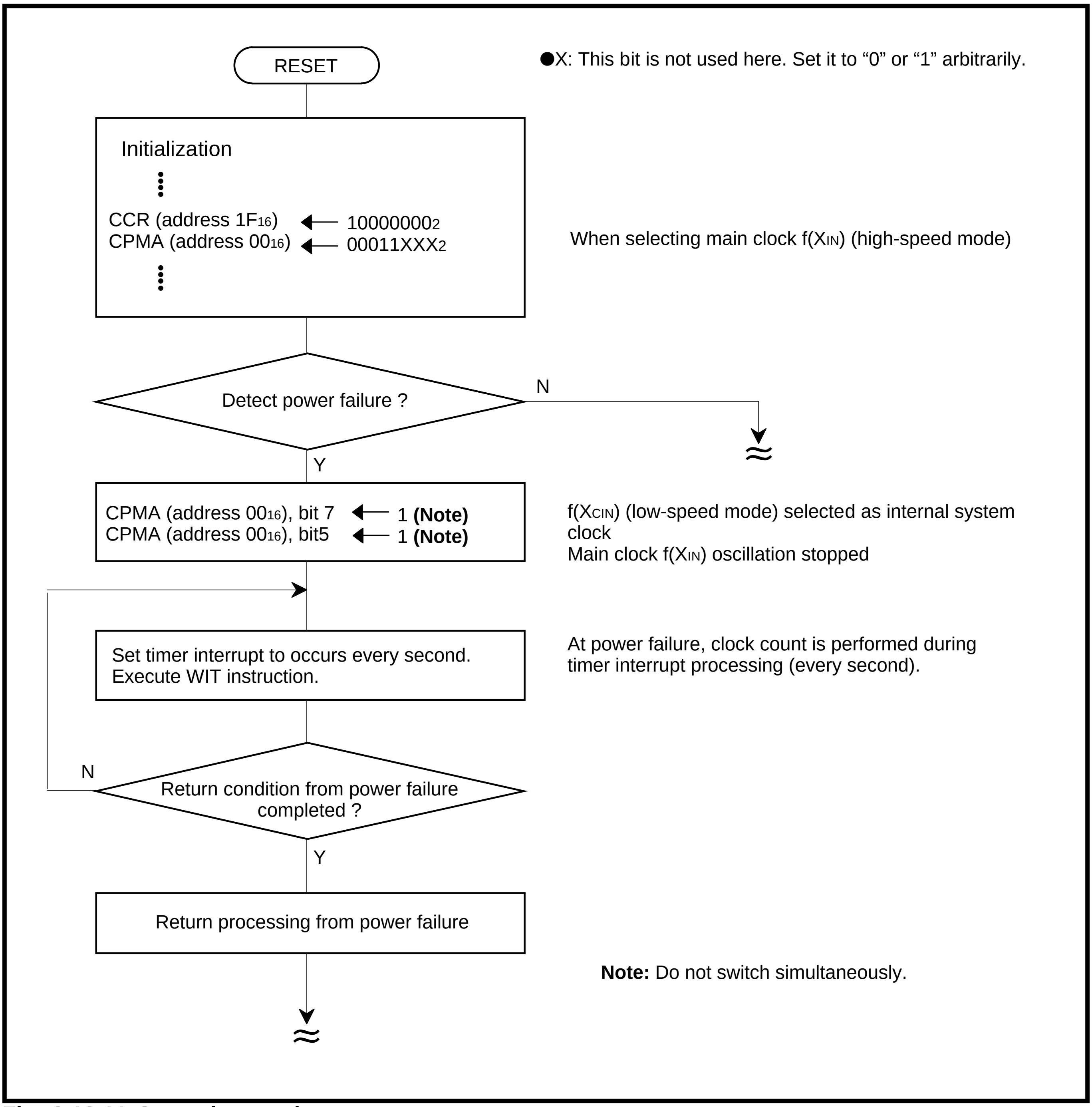


Fig. 2.12.11 Control procedure

(2) Counting without clock error during power failure

Outline: It keeps counting without clock error during a power failure.

Specifications:
- •Reducing power consumption as low as possible while maintaining clock function
- •Clock: $f(X_{IN})$ = 24 MHz
- •Sub clock: $f(X_{CIN})$ = 32.768 kHz
- •Use of Timer 2 interrupt

For the peripheral circuit and the status transition during a power failure, refer to Figures 2.12.8 and 2.12.9.

Figure 2.12.12 shows the structure of clock counter, Figures 2.12.13 and 2.12.14 show the setting of relevant registers.

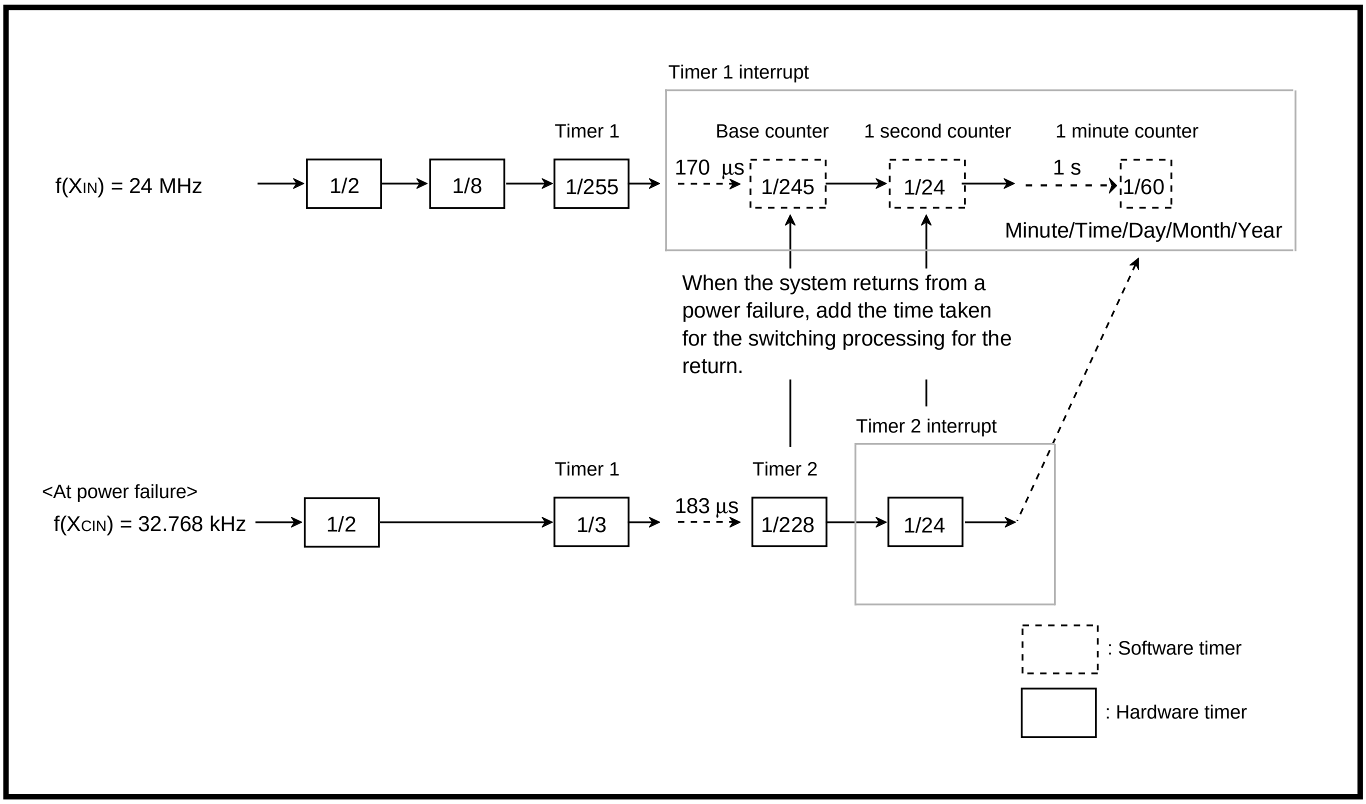


Fig. 2.12.12 Structure of clock counter

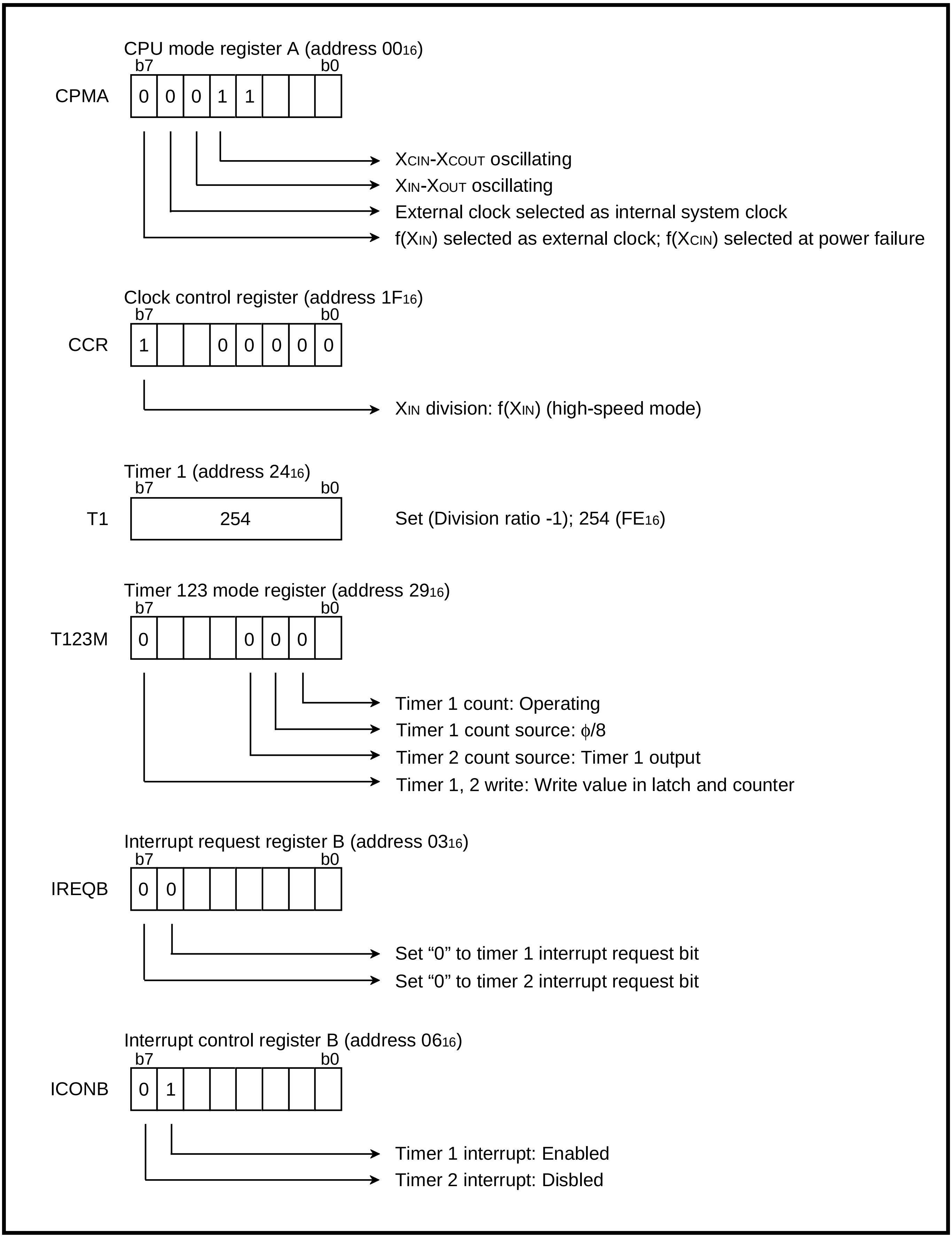


Fig. 2.12.13 Initial setting of relevant registers

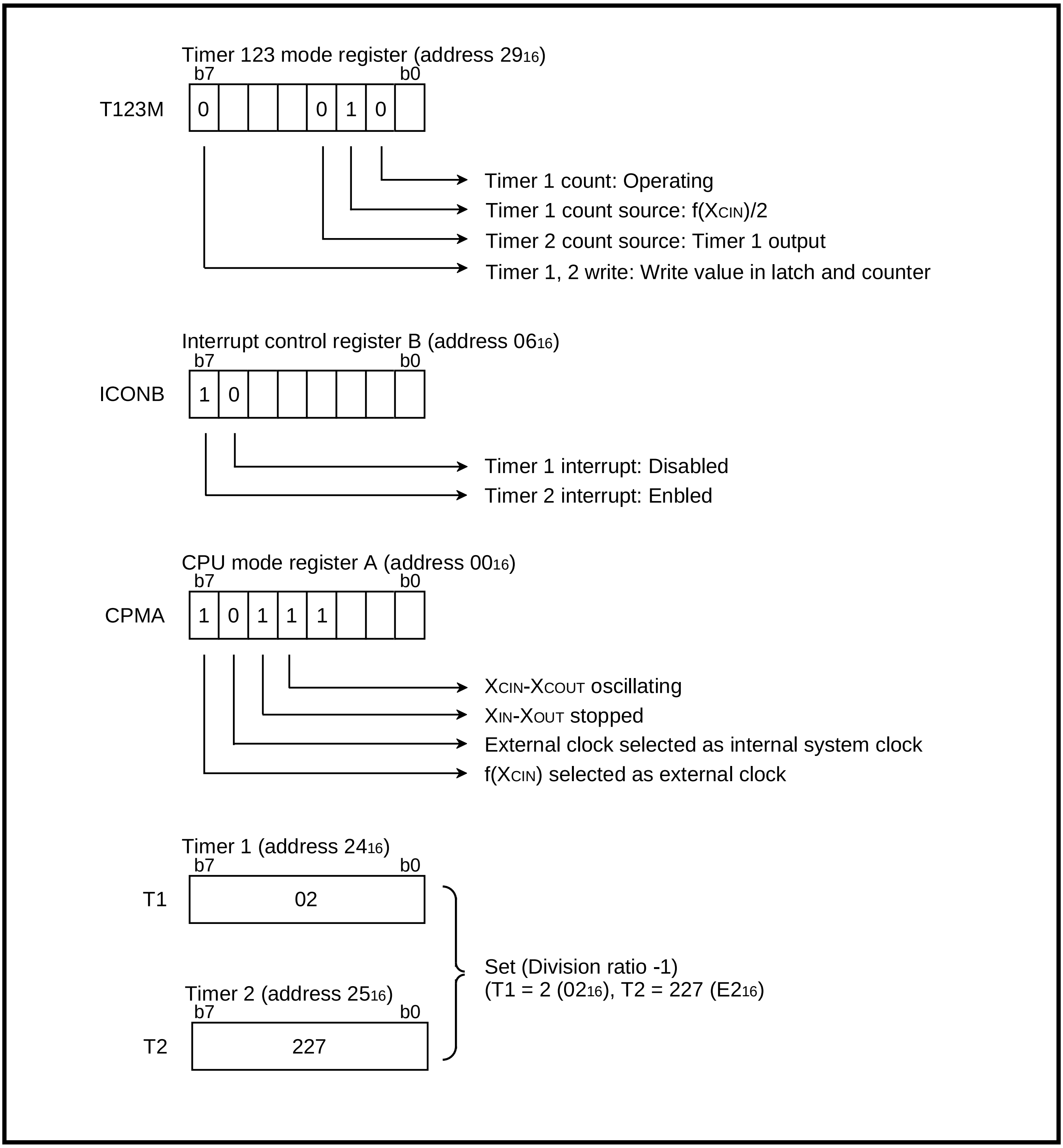


Fig. 2.12.14 Setting of relevant registers after detecting power failure

Control procedure: Set the relevant registers in the order shown below to prepare for a power failure.

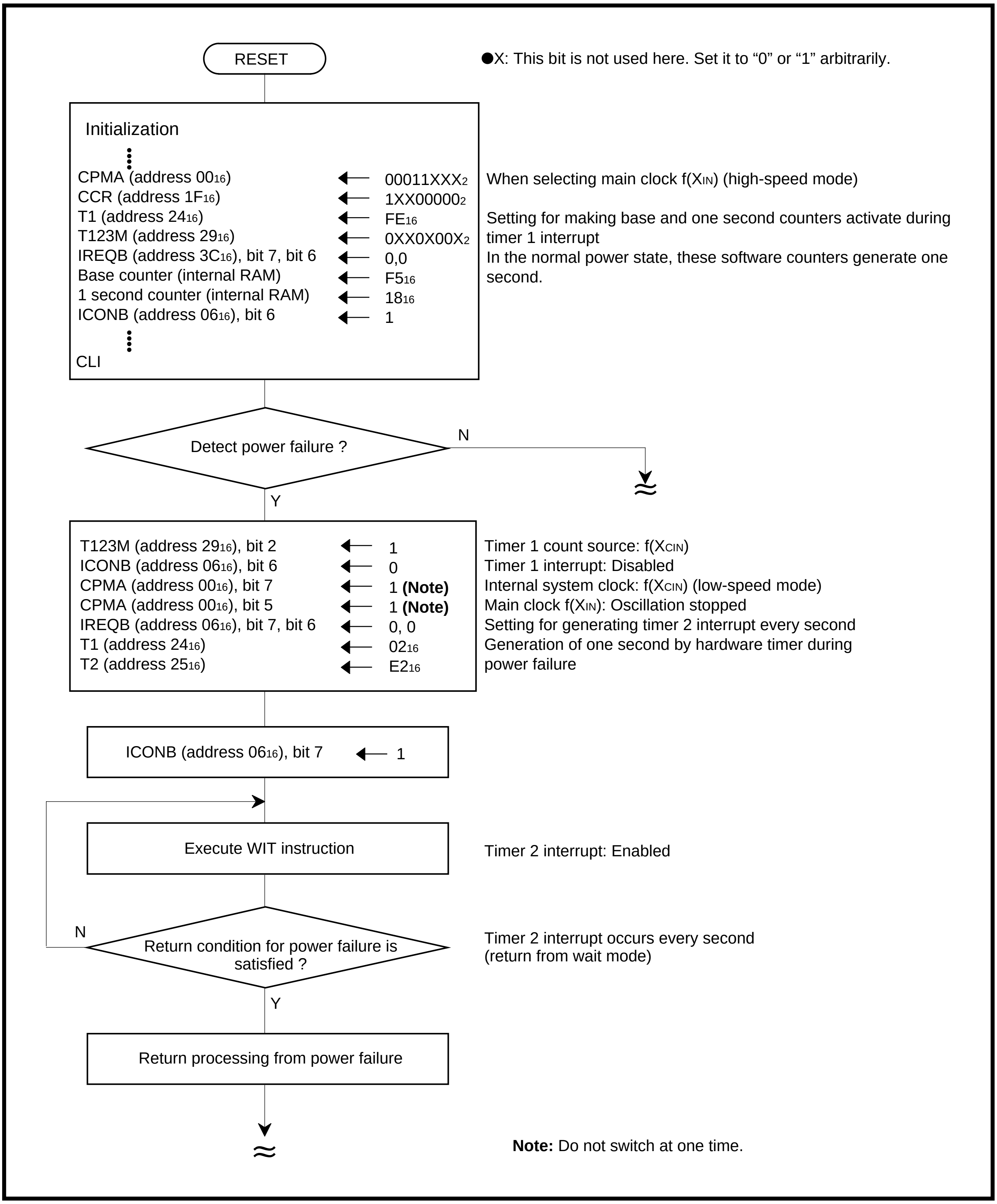


Fig. 2.12.15 Control procedure (1)

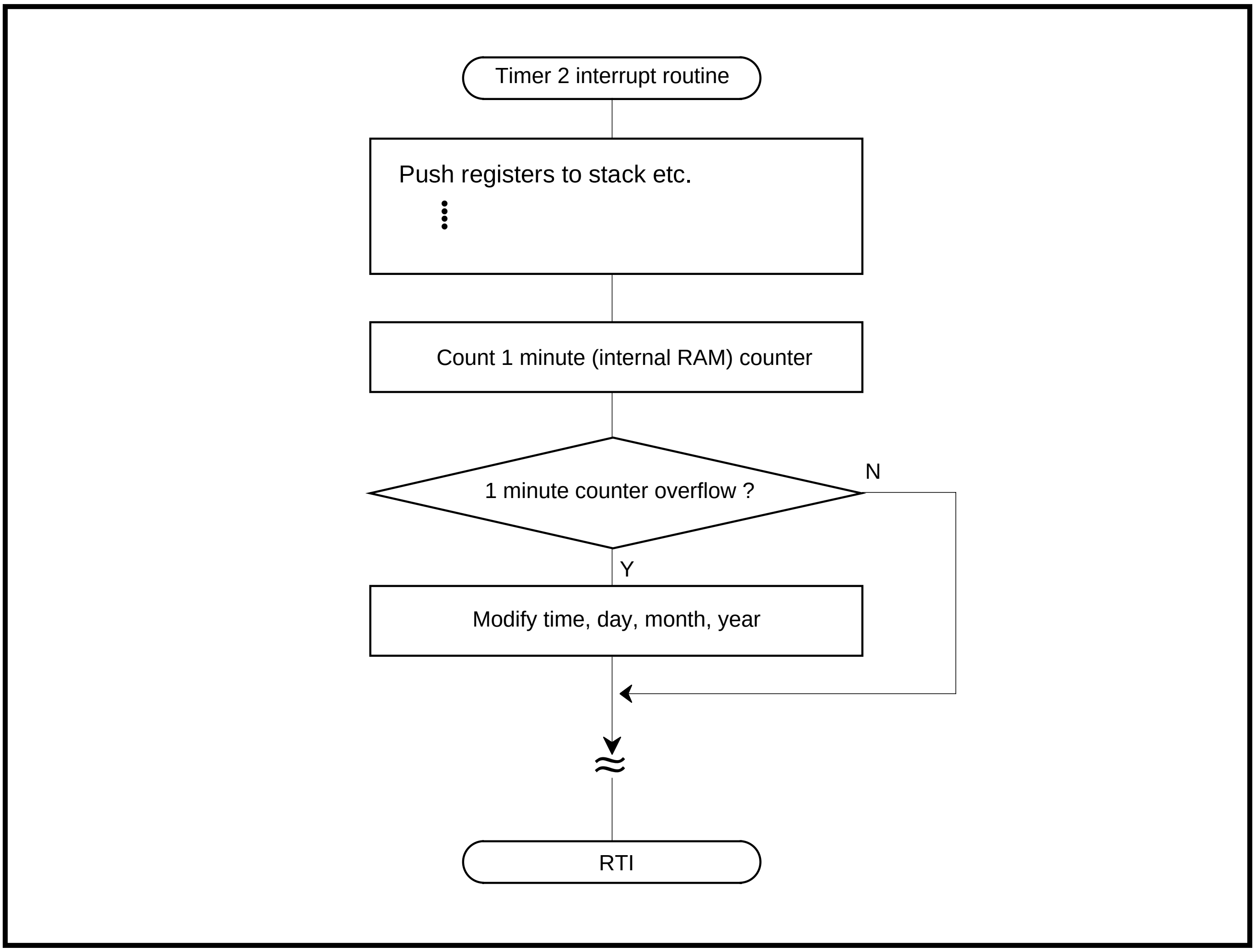


Fig. 2.12.16 Control procedure (2)

CHAPTER 3

APPENDIX

3.1 Electrical characteristics

3.1.1 Absolute maximum ratings

Table 3.1.1 Absolute maximum ratings

Symbol	Parameter			Conditions	Ratings	Unit
V_{CC}	Power source voltage			All voltages are based on Vss. Output transistors are cut off.	–0.3 to 6.5	V
AV_{CC}	Analog power source voltage AVcc, Ext.Cap				–0.3 to V_{CC}+0.3	V
V_I	Input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–0.3 to V_{CC}+0.3	V
V_I	Input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}			–0.3 to V_{CC}+0.3	V
V_I	Input voltage	CNV_{SS}	Mask ROM version		–0.3 to Vcc + 0.3	V
			Flash memory version		–0.3 to 6.5	V
V_I	Input voltage	USB D+, USB D–			–0.5 to 3.8	V
V_O	Output voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$, X_{OUT}, X_{COUT}, LPF			–0.3 to V_{CC}+0.3	V
V_O	Output voltage	USB D+, USB D–, Ext. Cap			–0.5 to 3.8	V
P_d	Power dissipation (**Note**)			Ta = 25°C	750	mW
T_{opr}	Operating temperature				–20 to 70	°C
T_{stg}	Storage temperature				–40 to 125	°C

Note: The maximum power dissipation depends on the MCU's power dissipation and the specific heat consumption of the package.

3.1.2 Recommended operating conditions (In Vcc = 5 V)

Table 3.1.2 Recommended operating conditions (V_{CC} = 4.15 to 5.25 V, V_{SS} = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
V_{CC}	Power source voltage		4.15	5.0	5.25	V
AV_{CC}	Analog reference voltage		4.15	5.0	V_{CC}	V
V_{SS}	Power source voltage			0		V
AV_{SS}	Analog reference voltage			0		V
V_{IH}	"H" input voltage	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting VIHL level input)	P20–P27	0.5V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting TTL level input for MBI input)	P54–P57, P60–P67, P72	2.0		V_{CC}	V
V_{IH}	"H" input voltage	RESET, X_{IN}, X_{CIN}, CNVss	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	USB D+, USB D–	2.0		3.8	V
V_{IL}	"L" input voltage	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	0		0.2V_{CC}	V
V_{IL}	"L" input voltage (Selecting VIHL level input)	P20–P27	0		0.16V_{CC}	V
V_{IL}	"L" input voltage (Selecting TTL level input for MBI input)	P54–P57, P60–P67, P72	0		0.8	V
V_{IL}	"L" input voltage	RESET, X_{IN}, X_{CIN}, CNVss	0		0.2V_{CC}	V
V_{IL}	"L" input voltage	USB D+, USB D–			0.8	V
$\Sigma I_{OH(peak)}$	"H" total peak output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			80	mA
$\Sigma I_{OH(avg)}$	"H" total average output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			40	mA
$I_{OH(peak)}$	"H" peak output current (**Note 2**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–10	mA
$I_{OL(peak)}$	"L" peak output current (**Note 2**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			10	mA
$I_{OH(avg)}$	"H" average output current (**Note 3**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–5.0	mA
$I_{OL(avg)}$	"L" average output current (**Note 3**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			5.0	mA
f(CNTR0)	Timer X input frequency (**Note 4**)				5.0	MHz
f(CNTR1)	Timer Y input frequency (**Note 4**)				5.0	MHz
f(X_{IN})	Main clock input frequency (**Notes 4, 5**)		1		24	MHz
f(X_{CIN})	Sub-clock input frequency (**Notes 4, 6**)			32.768	50/5.0	kHz/MHz

Notes 1: The total peak output current is the peak value of the peak currents flowing through all the applicable ports. The total average output current is the average value measured over 100 ms flowing through all the applicable ports.
2: The peak output current is the peak current flowing in each port.
3: The average output current is an average value measured over 100 ms.
4: The duty of oscillation frequency is 50 %.
5: Connect a ceramic resonator or a quartz-crystal oscillator between the X_{IN} and X_{OUT} pins. Its maximum oscillation frequency must be 24 MHz. However, make sure to set ϕ to 12 MHz or slower. More faster clocks are required as the f(X_{IN}) when using the frequency synthesizer as possible.
6: Connect a ceramic resonator or a quartz-crystal oscillator between the X_{CIN} and X_{COUT} pins. Its maximum oscillation frequency must be 50 kHz. Input an external clock having 5 MHz frequency (max.) from the X_{CIN} pin.

3.1.3 Electrical characteristics (In Vcc = 5 V)

Table 3.1.3 Electrical characteristics (1) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V_{OH}	"H" output voltage P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	I_{OH} = –10 mA	V_{CC}–2.0			V
V_{OH}	"H" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	2.8		3.6	V
V_{OL}	"L" output voltage P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	I_{OL} = 10 mA			2.0	V
V_{OL}	"L" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %			0.3	V
V_{T+}–V_{T-}	Hysteresis CNTR0, CNTR1, INT0, INT1, RDY, $\overline{\text{HOLD}}$, P20–P27			0.5		V
V_{T+}–V_{T-}	Hysteresis URXD1, URXD2 (SCLK), $\overline{\text{CTS2}}$ (SRXD), SRDY, $\overline{\text{CTS1}}$			0.5		V
V_{T+}–V_{T-}	Hysteresis $\overline{\text{RESET}}$			0.5		V
I_{IH}	"H" input current P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	V_I = V_{CC}			5.0	µA
I_{IH}	"H" input current $\overline{\text{RESET}}$, CNV_{SS}				5.0	µA
I_{IH}	"H" input current X_{IN}			9.0	20	µA
I_{IH}	"H" input current X_{CIN}				5.0	µA
I_{IL}	"L" input current P00–P07, P10–P17, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	V_I = V_{SS}			–5.0	µA
I_{IL}	"L" input current $\overline{\text{RESET}}$				–5.0	µA
I_{IL}	"L" input current CNV_{SS}				–20	µA
I_{IL}	"L" input current X_{IN}			–9.0	–20	µA
I_{IL}	"L" input current X_{CIN}				–5.0	µA
I_{IL}	"L" input current P20–P27	V_I = V_{SS} Pull-ups "off"			–5.0	µA
		V_{CC} = 5.0 V, V_I = V_{SS} Pull-ups "on"	–30	–65	–140	µA
V_{RAM}	RAM hold voltage	When clock is stopped	2.0		5.25	V

In Vcc = 5 V

Table 3.1.4 Electrical characteristics (2) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
I_{CC}	Power source current (Output transistor is isolated.)	Normal mode (**Note 1**) f(X_{IN}) = 24 MHz, ϕ = 12 MHz USB operating Frequency synthesizer ON		40	90	mA
		Wait mode (**Note 2**) f(X_{IN}) = 24 MHz, ϕ = 12 MHz USB block enabled, USB clock stopped, Frequency synthesizer ON		5.0	11	mA
		Wait mode (**Note 3**) f(X_{CIN}) = 32 kHz, ϕ = 16 kHz USB block disabled Frequency synthesizer OFF USB transceiver DC-DC converter OFF			10	µA
		Stop mode USB transceiver DC-DC converter ON Low current mode (USBC3 = “1”)		100	250	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 25 °C			1.0	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 70 °C			10	µA

<Test conditions>

Notes 1: Operating in single-chip mode
Clock input from X_{IN} pin (X_{OUT} oscillator stopped)
USB operating with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, CPU, two UARTs, DMAC, Timers and Count source generator
Disabled functions: Master CPU bus interface and Serial I/O

2: Operating in single-chip mode with Wait mode
Clock input from X_{IN} pin (X_{OUT} oscillator stopped)
USB suspended due to USB clock stopped with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, Timers and Count source generator
Disabled functions: CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

3: Operating in single-chip mode with Wait mode
X_{IN} - X_{OUT} oscillator stopped
Clock input from X_{CIN} pin (X_{COUT} oscillator stopped)
USB stopped, USB clock stopped and USB transceiver DC-DC converter disabled
Operating functions: Timers and Count source generator
Disabled functions: Frequency synthesizer, CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

Timing Requirements

In Vcc = 5 V

Table 3.1.5 Timing requirements (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_W(\overline{RESET})$	Reset input “L” pulse width	2			µs
$t_C(X_{IN})$	Main clock input cycle time (**Note**)	41.66			ns
$t_{WH}(X_{IN})$	Main clock input “H” pulse width	0.4•tc(XIN)			ns
$t_{WL}(X_{IN})$	Main clock input “L” pulse width	0.4•tc(XIN)			ns
$t_C(X_{CIN})$	Sub-clock input cycle time	200			ns
$t_{WH}(X_{CIN})$	Sub-clock input “H” pulse width	0.4•tc(XCIN)			ns
$t_{WL}(X_{CIN})$	Sub-clock input “L” pulse width	0.4•tc(XCIN)			ns
$t_C(INT)$	INT0, INT1 input cycle time	200			ns
$t_{WH}(INT)$	INT0, INT1 input “H” pulse width	90			ns
$t_{WL}(INT)$	INT0, INT1 input “L” pulse width	90			ns
$t_C(CNTRI)$	CNTR0, CNTR1 input cycle time	200			ns
$t_{WH}(CNTRI)$	CNTR0, CNTR1 input “H” pulse width	80			ns
$t_{WL}(CNTRI)$	CNTR0, CNTR1 input “L” pulse width	80			ns
$t_d(\phi\text{-}T_{OUT})$	Timer TOUT delay time			15	ns
$t_d(\phi\text{-}CNTR_0)$	Timer CNTR0 delay time (Pulse output mode)			15	ns
$t_C(CNTR_{E0})$	Timer CNTR0 input cycle time (Event counter mode)	200			ns
$t_{WH}(CNTR_{E0})$	Timer CNTR0 input “H” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{WL}(CNTR_{E0})$	Timer CNTR0 input “L” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_d(\phi\text{-}CNTR_1)$	Timer CNTR1 delay time (Pulse output mode)			15	ns
$t_C(CNTR_{E1})$	Timer CNTR1 input cycle time (Event counter mode)	200			ns
$t_{WH}(CNTR_{E1})$	Timer CNTR1 input “H” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{WL}(CNTR_{E1})$	Timer CNTR1 input “L” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_C(SCLK_E)$	Serial I/O external clock input cycle time	400			ns
$t_{WH}(SCLK_E)$	Serial I/O external clock input “H” pulse width	190			ns
$t_{WL}(SCLK_E)$	Serial I/O external clock input “L” pulse width	180			ns
$t_{su}(SRXD\text{-}SCLK_E)$	Serial I/O input setup time (external clock)	15			ns
$t_h(SCLK_E\text{-}SRXD)$	Serial I/O input hold time (external clock)	10			ns
$t_d(SCLK_E\text{-}STXD)$	Serial I/O output delay time (external clock)			25	ns
$t_v(SCLK_E\text{-}SRDY)$	Serial I/O $\overline{SRDY}$ valid time (external clock)			26	ns
$t_c(SCLK_I)$	Serial I/O internal clock output cycle time	166.66			ns
$t_{WH}(SCLK_I)$	Serial I/O internal clock output “H” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{WL}(SCLK_I)$	Serial I/O internal clock output “L” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{su}(SRXD\text{-}SCLK_I)$	Serial I/O input setup time (internal clock)	20			ns
$t_h(SCLK_I\text{-}SRXD)$	Serial I/O input hold time (internal clock)	5			ns
$t_d(SCLK_I\text{-}STXD)$	Serial I/O output delay time (internal clock)			5	ns

Note: Make sure not to exceed 12 MHz of φ, in other words, tc(φ) ≥ 83.33 ns). For example, set bit 7 of the clock control register (CCR) to “0” in the case of $t_c(X_{IN})$ < 41.66 ns.

In Vcc = 5 V

Table 3.1.6 Master CPU bus interface (MBI; RD, WR separate type) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-R)	$\overline{S_0}$, $\overline{S_1}$ setup time for read	0			ns
t_{su}(S-W)	$\overline{S_0}$, $\overline{S_1}$ setup time for write	0			ns
t_h(R-S)	$\overline{S_0}$, $\overline{S_1}$ hold time for read	0			ns
t_h(W-S)	$\overline{S_0}$, $\overline{S_1}$ hold time for write	0			ns
t_{su}(A-R)	A_0 setup time for read	10			ns
t_{su}(A-W)	A_0 setup time for write	10			ns
t_h(R-A)	A_0 hold time for read	0			ns
t_h(W-A)	A_0 hold time for write	0			ns
t_w(R)	Read pulse width	50			ns
t_w(W)	Write pulse width	50			ns
t_{su}(D-W)	Data input setup time before write	25			ns
t_h(W-D)	Data input hold time after write	0			ns
t_a(R-D)	Data output enable time after read			40	ns
t_v(R-D)	Data output disable time after read	10			ns
t_v(R-OBF)	OBF output transmission time after read			40	ns
t_d(W-IBF)	$\overline{IBF}$ output transmission time after write			40	ns

In Vcc = 5 V

Table 3.1.7 Master CPU bus interface (MBI; R/$\overline{W}$ type) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-E)	$\overline{S_0}$, $\overline{S_1}$ setup time	0			ns
t_h(E-S)	$\overline{S_0}$, $\overline{S_1}$ hold time	0			ns
t_{su}(A-E)	A_0 setup time	10			ns
t_h(E-A)	A_0 hold time	0			ns
t_{su}(RW-E)	R/$\overline{W}$ setup time	10			ns
t_h(E-RW)	R/$\overline{W}$ hold time	10			ns
t_w(E)	Enable pulse width	50			ns
t_w(E-E)	Enable pulse interval	50			ns
t_{su}(D-E)	Data input setup time before write	25			ns
t_h(E-D)	Data input hold time after write	0			ns
t_a(E-D)	Data output enable time after read			40	ns
t_v(E-D)	Data output disable time after read	10			ns
t_v(E-OBF)	OBF output transmission time after E inactive			40	ns
t_d(E-IBF)	$\overline{IBF}$ output transmission time after E inactive			40	ns

In Vcc = 5 V

Table 3.1.8 Timing requirements and switching characteristics in memory expansion and microprocessor modes

(Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{C}(\phi)$	ϕ clock cycle time	83.33			ns
$t_{WH}(\phi)$	ϕ clock “H” pulse width	0.5•tc(ϕ) – 5			ns
$t_{WL}(\phi)$	ϕ clock “L” pulse width	0.5•tc(ϕ) – 5			ns
$t_{d}(\phi$ -AH)	AB$_{15}$–AB$_{8}$ delay time			31	ns
$t_{v}(\phi$ -AH)	AB$_{15}$–AB$_{8}$ valid time	5			ns
$t_{d}(\phi$ -AL)	AB$_{7}$–AB$_{0}$ delay time			33	ns
$t_{v}(\phi$ -AL)	AB$_{7}$–AB$_{0}$ valid time	5			ns
$t_{d}(\phi$ -WR)	$\overline{\mathrm{WR}}$ delay time			6	ns
$t_{v}(\phi$ -WR)	$\overline{\mathrm{WR}}$ valid time	3			ns
$t_{d}(\phi$ -RD)	$\overline{\mathrm{RD}}$ delay time			6	ns
$t_{v}(\phi$ -RD)	$\overline{\mathrm{RD}}$ valid time	3			ns
$t_{d}(\phi$ -SYNC)	SYNCOUT delay time			6	ns
$t_{v}(\phi$ -SYNC)	SYNCOUT valid time	4			ns
$t_{d}(\phi$ -DMA)	DMAOUT delay time			25	ns
$t_{v}(\phi$ -DMA)	DMAOUT valid time	5			ns
t_{su}(RDY- ϕ)	RDY setup time	21			ns
$t_{h}(\phi$ -RDY)	RDY hold time	0			ns
t_{su}(HOLD- ϕ)	$\overline{\mathrm{HOLD}}$ setup time	21			ns
$t_{h}(\phi$ -HOLD)	$\overline{\mathrm{HOLD}}$ hold time	0			ns
$t_{d}(\phi$ -HLDAL)	$\overline{\mathrm{HOLD}}$ “L” delay time			25	ns
$t_{d}(\phi$ -HLDAH)	$\overline{\mathrm{HOLD}}$ “H” delay time			25	ns
t_{su}(DB- ϕ)	Data bus setup time	7			ns
$t_{h}(\phi$ -DB)	Data bus hold time	0			ns
$t_{d}(\phi$ -DB)	Data bus delay time			22	ns
tV(ϕ -DB)	Data bus valid time (**Note 1**)	13			ns
$t_{d}(\phi$ -EDMA)	$\overline{\mathrm{EDMA}}$ delay time			9	ns
$t_{v}(\phi$ -EDMA)	$\overline{\mathrm{EDMA}}$ valid time	4			ns
t_{WL}(WR) (**Note 2**)	$\overline{\mathrm{WR}}$ pulse width	0.5•tc(ϕ) – 5			ns
t_{WL}(RD) (**Note 2**)	$\overline{\mathrm{RD}}$ pulse width	0.5•tc(ϕ) – 5			ns
t_{d}(AH-WR)	AB$_{15}$–AB$_{8}$ valid time before $\overline{\mathrm{WR}}$	0.5•tc(ϕ) – 28			ns
t_{d}(AL-WR)	AB$_{7}$–AB$_{0}$ valid time before $\overline{\mathrm{WR}}$	0.5•tc(ϕ) – 30			ns
t_{v}(WR-AH)	AB$_{15}$–AB$_{8}$ valid time after $\overline{\mathrm{WR}}$	0			ns
t_{v}(WR-AL)	AB$_{7}$–AB$_{0}$ valid time after $\overline{\mathrm{WR}}$	0			ns
t_{d}(AH-RD)	AB$_{15}$–AB$_{8}$ valid time before $\overline{\mathrm{RD}}$	0.5•tc(ϕ) – 28			ns
t_{d}(AL-RD)	AB$_{7}$–AB$_{0}$ valid time before $\overline{\mathrm{RD}}$	0.5•tc(ϕ) – 30			ns
t_{v}(RD-AH)	AB$_{15}$–AB$_{8}$ valid time after $\overline{\mathrm{RD}}$	0			ns
t_{v}(RD-AL)	AB$_{7}$–AB$_{0}$ valid time after $\overline{\mathrm{RD}}$	0			ns
t_{su}(RDY-WR)	RDY setup time before $\overline{\mathrm{WR}}$	27			ns
t_{h}(WR-RDY)	RDY hold time after $\overline{\mathrm{WR}}$	0			ns
t_{su}(RDY-RD)	RDY setup time before $\overline{\mathrm{RD}}$	27			ns
t_{h}(RD-RDY)	RDY hold time after $\overline{\mathrm{RD}}$	0			ns
t_{su}(DB-RD)	Data bus setup time before $\overline{\mathrm{RD}}$	13			ns
t_{h}(RD-DB)	Data bus hold time after $\overline{\mathrm{RD}}$	0			ns
t_{d}(WR-DB)	Data bus delay time before $\overline{\mathrm{WR}}$			20	ns
t_{v}(WR-DB)	Data bus valid time after $\overline{\mathrm{WR}}$ (**Note 1**)	10			ns
t_{v}(WR-EDMA)	$\overline{\mathrm{EDMA}}$ delay time after $\overline{\mathrm{WR}}$	2			ns
t_{v}(RD-EDMA)	$\overline{\mathrm{EDMA}}$ valid time after $\overline{\mathrm{RD}}$	2			ns
t_{r}(D+), t_{r}(D-)	USB output rise time, C_{L} = 50 pF	4		20	ns
t_{f}(D+), t_{f}(D-)	USB output fall time, C_{L} = 50 pF	4		20	ns

Notes 1: Test conditions: I_{OHL} = ± 5mA, C_L = 50 pF
2: t_{WL}(RD) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
t_{WL}(WR) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
For example, two software waits, PHI = 12 MHz operating
t_{WL}(RD) = 2.5 • tc(PHI) – 5 ns = 203.33 ns

3.1.4 Recommended Operating Conditions (In Vcc = 3 V)

Table 3.1.9 Recommended operating conditions (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter		Limits Min.	Limits Typ.	Limits Max.	Unit
V_{CC}	Power source voltage		3.0	3.3	3.6	V
AV_{CC}	Analog reference voltage		3.0	3.3	V_{CC}	V
V_{SS}	Power source voltage			0		V
AV_{SS}	Analog reference voltage			0		V
Ext. Cap.	DC-DC converter voltage		3.0	3.3	3.6	V
V_{IH}	"H" input voltage	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting VIHL level input)	P20–P27	0.5V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	RESET, XIN, XCIN, CNVss	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	USB D+, USB D–	2.0			V
V_{IL}	"L" input voltage	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	0		0.2V_{CC}	V V
V_{IL}	"L" input voltage (Selecting VIHL level input)	P20–P27	0		0.16V_{CC}	V
V_{IL}	"L" input voltage	RESET, XIN, XCIN, CNVss	0		0.2V_{CC}	V
V_{IL}	"L" input voltage	USB D+, USB D–			0.8	mA
$\Sigma I_{OH(peak)}$	"H" total peak output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			80	mA
$\Sigma I_{OH(avg)}$	"H" total average output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current (**Note 1**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			40	mA
$I_{OH(peak)}$	"H" peak output current (**Note 2**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–10	mA
$I_{OL(peak)}$	"L" peak output current (**Note 2**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			10	mA
$I_{OH(avg)}$	"H" average output current (**Note 3**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			–5.0	mA
$I_{OL(avg)}$	"L" average output current (**Note 3**)	P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87			5.0	mA
f(CNTR0)	Timer X input frequency (**Note 4**)				5.0	MHz
f(CNTR1)	Timer Y input frequency (**Note 4**)				5.0	MHz
f(XIN)	Main clock input frequency (**Notes 4, 5**)		1		24	MHz
f(XCIN)	Sub-clock input frequency (**Notes 4, 6**)			32.768	50/5.0	kHz/MHz

Notes 1: The total peak output current is the peak value of the peak currents flowing through all the applicable ports. The total average output current is the average value measured over 100 ms flowing through all the applicable ports.
2: The peak output current is the peak current flowing in each port.
3: The average output current is an average value measured over 100 ms.
4: The duty of oscillation frequency is 50 %.
5: Connect a ceramic resonator or a quartz-crystal oscillator between the XIN and XOUT pins. Its maximum oscillation frequency must be 24 MHz. However, make sure to set ϕ to 6 MHz or slower. More faster clocks are required as the f(XIN) when using the frequency synthesizer as possible.
6: Connect a ceramic resonator or a quartz-crystal oscillator between the XCIN and XCOUT pins. Its maximum oscillation frequency must be 50 kHz. Input an external clock having 5 MHz (max.) frequency from the XCIN pin.

3.1.5 Electrical Characteristics (In Vcc = 3 V)

Table 3.1.10 Electrical characteristics (1) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V_{OH}	"H" output voltage P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	I_{OH} = –1 mA	V_{CC}–1.0			V
V_{OH}	"H" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	2.8		3.6	V
V_{OL}	"L" output voltage P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	I_{OL} = 1 mA			1.0	V
V_{OL}	"L" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	0		0.3	V
V_{T+}–V_{T-}	Hysteresis CNTR0, CNTR1, INT0, INT1, RDY, $\overline{HOLD}$, P20–P27			0.3		V
V_{T+}–V_{T-}	Hysteresis URXD1, URXD2 (SCLK), $\overline{CTS2}$ (SRXD), SRDY, $\overline{CTS1}$			0.3		V
V_{T+}–V_{T-}	Hysteresis $\overline{RESET}$			0.3		V
I_{IH}	"H" input current P00–P07, P10–P17, P20–P27, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	V_I = V_{CC}			5.0	µA
I_{IH}	"H" input current $\overline{RESET}$, CNV_{SS}				5.0	µA
I_{IH}	"H" input current X_{IN}			9.0	20	µA
I_{IH}	"H" input current X_{CIN}				5.0	µA
I_{IL}	"L" input current P00–P07, P10–P17, P30–P37, P40–P44, P50–P57, P60–P67, P70–P74, P80–P87	V_I = V_{SS}			–5.0	µA
I_{IL}	"L" input current $\overline{RESET}$				–5.0	µA
I_{IL}	"L" input current CNV_{SS}				–20	µA
I_{IL}	"L" input current X_{IN}			–9.0	–20	µA
I_{IL}	"L" input current X_{CIN}				–5.0	µA
I_{IL}	"L" input current P20–P27	V_I = V_{SS} Pull-ups "off"			–5.0	µA
		V_{CC} = 3.0 V, V_I = V_{SS} Pull-ups "on"	–10	–20	–50	µA
V_{RAM}	RAM hold voltage	When clock is stopped	2.0			V

In Vcc = 3 V

Table 3.1.11 Electrical characteristics (2) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
I_{CC}	Power source current (Output transistor is isolated.)	Normal mode (**Note 1**) f(X_{IN}) = 24 MHz, ϕ = 6 MHz USB operating Frequency synthesizer ON		25	45	mA
		Wait mode (**Note 2**) f(X_{IN}) = 24 MHz, ϕ = 6 MHz USB block enabled, USB clock stopped, Frequency synthesizer ON		2.5	6	mA
		Wait mode (**Note 3**) f(X_{CIN}) = 32 kHz, ϕ = 16 kHz USB block disabled Frequency synthesizer OFF USB transceiver DC-DC converter OFF			6	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 25 °C			1.0	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 70 °C			10	µA

<Test conditions>

Notes 1: Operating in single-chip mode
Clock input from X_{IN} pin (X_{OUT} oscillator stopped)
USB operating with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, CPU, two UARTs, DMAC, Timers and Count source generator
Disabled functions: Master CPU bus interface and Serial I/O

2: Operating in single-chip mode with Wait mode
Clock input from X_{IN} pin (X_{OUT} oscillator stopped)
USB suspended due to USB clock stopped with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, Timers and Count source generator
Disabled functions: CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

3: Operating in single-chip mode with Wait mode
X_{IN} - X_{OUT} oscillator stopped
Clock input from X_{CIN} pin (X_{COUT} oscillator stopped)
USB stopped, USB clock stopped and USB transceiver DC-DC converter disabled
Operating functions: Timers and Count source generator
Disabled functions: Frequency synthesizer, CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

Timing Requirements
In Vcc = 3 V

Table 3.1.12 Timing requirements (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{W}(\overline{RESET})$	Reset input "L" pulse width	2			µs
$t_{C}(X_{IN})$	Main clock input cycle time (**Note**)	41.66			ns
$t_{WH}(X_{IN})$	Main clock input "H" pulse width	0.4•tc(XIN)			ns
$t_{WL}(X_{IN})$	Main clock input "L" pulse width	0.4•tc(XIN)			ns
$t_{C}(X_{CIN})$	Sub-clock input cycle time	200			ns
$t_{WH}(X_{CIN})$	Sub-clock input "H" pulse width	0.4•tc(XCIN)			ns
$t_{WL}(X_{CIN})$	Sub-clock input "L" pulse width	0.4•tc(XCIN)			ns
$t_{C}(INT)$	INT_0, INT_1 input cycle time	250			ns
$t_{WH}(INT)$	INT_0, INT_1 input "H" pulse width	110			ns
$t_{WL}(INT)$	INT_0, INT_1 input "L" pulse width	110			ns
$t_{C}(CNTRI)$	$CNTR_0$, $CNTR_1$ input cycle time	250			ns
$t_{WH}(CNTRI)$	$CNTR_0$, $CNTR_1$ input "H" pulse width	110			ns
$t_{WL}(CNTRI)$	$CNTR_0$, $CNTR_1$ input "L" pulse width	110			ns
$t_{d}(\phi\text{-TOUT})$	Timer T_{OUT} delay time			17	ns
$t_{d}(\phi\text{-}CNTR_0)$	Timer $CNTR_0$ delay time (Pulse output mode)			16	ns
$t_{C}(CNTR_{E0})$	Timer $CNTR_0$ input cycle time (Event counter mode)	250			ns
$t_{WH}(CNTR_{E0})$	Timer $CNTR_0$ input "H" pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{WL}(CNTR_{E0})$	Timer $CNTR_0$ input "L" pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{d}(\phi\text{-}CNTR_1)$	Timer $CNTR_1$ delay time (Pulse output mode)			15	ns
$t_{C}(CNTR_{E1})$	Timer $CNTR_1$ input cycle time (Event counter mode)	250			ns
$t_{WH}(CNTR_{E1})$	Timer $CNTR_1$ input "H" pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{WL}(CNTR_{E1})$	Timer $CNTR_1$ input "L" pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{C}(SCLK_{E})$	Serial I/O external clock input cycle time	450			ns
$t_{WH}(SCLK_{E})$	Serial I/O external clock input "H" pulse width	220			ns
$t_{WL}(SCLK_{E})$	Serial I/O external clock input "L" pulse width	190			ns
$t_{su}(SRXD\text{-}SCLK_{E})$	Serial I/O input setup time (external clock)	20			ns
$t_{h}(SCLK_{E}\text{-}SRXD)$	Serial I/O input hold time (external clock)	15			ns
$t_{d}(SCLK_{E}\text{-}STXD)$	Serial I/O output delay time (external clock)			34	ns
$t_{v}(SCLK_{E}\text{-}SRDY)$	Serial I/O $\overline{SRDY}$ valid time (external clock)			35	ns
$t_{c}(SCLK_{I})$	Serial I/O internal clock output cycle time	300			ns
$t_{WH}(SCLK_{I})$	Serial I/O internal clock output "H" pulse width	0.5•tc(SCLKI) – 5			ns
$t_{WL}(SCLK_{I})$	Serial I/O internal clock output "L" pulse width	0.5•tc(SCLKI) – 5			ns
$t_{su}(SRXD\text{-}SCLK_{I})$	Serial I/O input setup time (internal clock)	20			ns
$t_{h}(SCLK_{I}\text{-}SRXD)$	Serial I/O input hold time (internal clock)	5			ns
$t_{d}(SCLK_{I}\text{-}STXD)$	Serial I/O output delay time (internal clock)			5	ns

Note: Make sure not to exceed 6 MHz of ϕ, in other words, $tc(\phi) \geq 166.66$ ns).

In Vcc = 3 V

Table 3.1.13 Master CPU bus interface (MBI; RD, WR separate type) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-R)	$\overline{S_0}$, $\overline{S_1}$ setup time for read	0			ns
t_{su}(S-W)	$\overline{S_0}$, $\overline{S_1}$ setup time for write	0			ns
t_h(R-S)	$\overline{S_0}$, $\overline{S_1}$ hold time for read	0			ns
t_h(W-S)	$\overline{S_0}$, $\overline{S_1}$ hold time for write	0			ns
t_{su}(A-R)	A_0 setup time for read	10			ns
t_{su}(A-W)	A_0 setup time for write	10			ns
t_h(R-A)	A_0 hold time for read	0			ns
t_h(W-A)	A_0 hold time for write	0			ns
t_w(R)	Read pulse width	80			ns
t_w(W)	Write pulse width	80			ns
t_{su}(D-W)	Data input setup time before write	35			ns
t_h(W-D)	Data input hold time after write	0			ns
t_a(R-D)	Data output enable time after read			65	ns
t_v(R-D)	Data output disable time after read	10			ns
t_v(R-OBF)	OBF output transmission time after read			50	ns
t_d(W-IBF)	$\overline{IBF}$ output transmission time after write			50	ns

In Vcc = 3 V

Table 3.1.14 Master CPU bus interface (MBI; R/$\overline{W}$ type) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-E)	$\overline{S_0}$, $\overline{S_1}$ setup time	0			ns
t_h(E-S)	$\overline{S_0}$, $\overline{S_1}$ hold time	0			ns
t_{su}(A-E)	A_0 setup time	10			ns
t_h(E-A)	A_0 hold time	0			ns
t_{su}(RW-E)	R/$\overline{W}$ setup time	10			ns
t_h(E-RW)	R/$\overline{W}$ hold time	10			ns
t_w(E)	Enable pulse width	80			ns
t_w(E-E)	Enable pulse interval	80			ns
t_{su}(D-E)	Data input setup time before write	35			ns
t_h(E-D)	Data input hold time after write	0			ns
t_a(E-D)	Data output enable time after read			65	ns
t_v(E-D)	Data output disable time after read	10			ns
t_v(E-OBF)	OBF output transmission time after E inactive			50	ns
t_d(E-IBF)	$\overline{IBF}$ output transmission time after E inactive			50	ns

In Vcc = 3 V

Table 3.1.15 Timing requirements and switching characteristics in memory expansion and microprocessor modes

(Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
tC(ϕ)	ϕ clock cycle time	166.66			µs
tWH(ϕ)	ϕ clock “H” pulse width	0.5•tc(ϕ) – 5			ns
tWL(ϕ)	ϕ clock “L” pulse width	0.5•tc(ϕ) – 5			ns
td(ϕ -AH)	AB15–AB8 delay time			45	ns
tv(ϕ -AH)	AB15–AB8 valid time	7			ns
td(ϕ -AL)	AB7–AB0 delay time			47	ns
tv(ϕ -AL)	AB7–AB0 valid time	7			ns
td(ϕ -WR)	WR delay time			8	ns
tv(ϕ -WR)	WR valid time	4			ns
td(ϕ -RD)	RD delay time			8	ns
tv(ϕ -RD)	RD valid time	3			ns
td(ϕ -SYNC)	SYNCOUT delay time			11	ns
tv(ϕ -SYNC)	SYNCOUT valid time	4			ns
td(ϕ -DMA)	DMAOUT delay time			26	ns
tv(ϕ -DMA)	DMAOUT valid time	9			ns
tsu(RDY- ϕ)	RDY setup time	35			ns
th(ϕ -RDY)	RDY hold time	0			ns
tsu(HOLD- ϕ)	HOLD setup time	21			ns
th(ϕ -HOLD)	HOLD hold time	0			ns
td(ϕ -HLDAL)	HOLD “L” delay time			30	ns
td(ϕ -HLDAH)	HOLD “H” delay time			30	ns
tsu(DB- ϕ)	Data bus setup time	9			ns
th(ϕ -DB)	Data bus hold time	0			ns
td(ϕ -DB)	Data bus delay time			30	ns
tv(ϕ -DB)	Data bus valid time (**Note 1**)	15			ns
td(ϕ -EDMA)	EDMA delay time			12	ns
tv(ϕ -EDMA)	EDMA valid time	8			ns
tWL(WR) (**Note 2**)	WR pulse width	0.5•tc(ϕ) – 6			ns
tWL(RD) (**Note 2**)	RD pulse width	0.5•tc(ϕ) – 6			ns
td(AH-WR)	AB15–AB8 valid time before WR	0.5•tc(ϕ) – 33			ns
td(AL-WR)	AB7–AB0 valid time before WR	0.5•tc(ϕ) – 35			ns
tv(WR-AH)	AB15–AB8 valid time after WR	0			ns
tv(WR-AL)	AB7–AB0 valid time after WR	0			ns
td(AH-RD)	AB15–AB8 valid time before RD	0.5•tc(ϕ) – 33			ns
td(AL-RD)	AB7–AB0 valid time before RD	0.5•tc(ϕ) – 35			ns
tv(RD-AH)	AB15–AB8 valid time after RD	0			ns
tv(RD-AL)	AB7–AB0 valid time after RD	0			ns
tsu(RDY-WR)	RDY setup time before WR	45			ns
th(WR-RDY)	RDY hold time after WR	0			ns
tsu(RDY-RD)	RDY setup time before RD	45			ns
th(RD-RDY)	RDY hold time after RD	0			ns
tsu(DB-RD)	Data bus setup time before RD	18			ns
th(RD-DB)	Data bus hold time after RD	0			ns
td(WR-DB)	Data bus delay time after WR			28	ns
tv(WR-DB)	Data bus valid time after WR (**Note 1**)	12			ns
tv(WR-EDMA)	EDMA delay time after WR	3			ns
tv(RD-EDMA)	EDMA valid time after RD	3			ns
tr(D+), tr(D-)	USB output rise time, CL = 50 pF	4		20	ns
tf(D+), tf(D-)	USB output fall time, CL = 50 pF	4		20	ns

Notes 1: Test conditions: I_{OHL} = ± 5mA, C_L = 50 pF
2: t_{WL}(RD) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
t_{WL}(WR) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
For example, two software waits, PHI = 12 MHz operating
t_{WL}(RD) = 2.5 • tc(PHI) – 5 ns = 203.33 ns

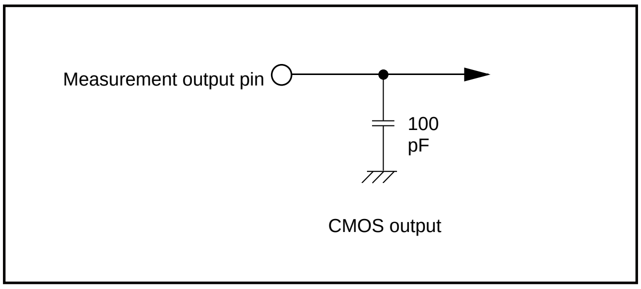


Fig. 3.1.1 Circuit for measuring output switching characteristics (1)

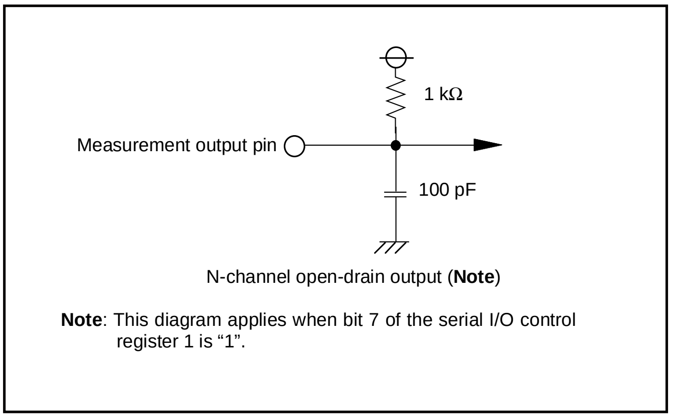


Fig. 3.1.2 Circuit for measuring output switching characteristics (2)

● Timing diagram

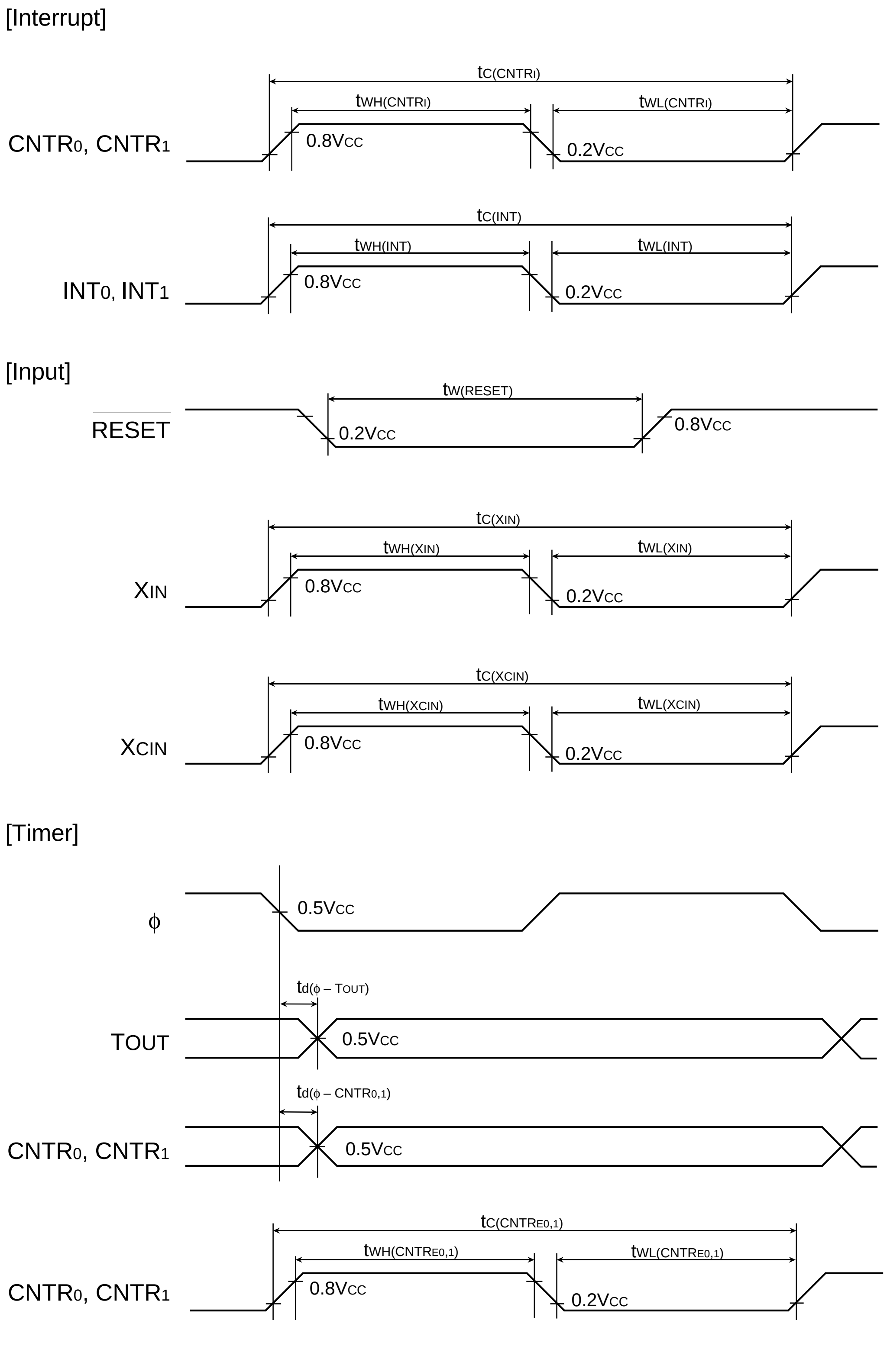


Fig. 3.1.3 Timing diagram (1)

● Timing diagram

[Serial I/O]

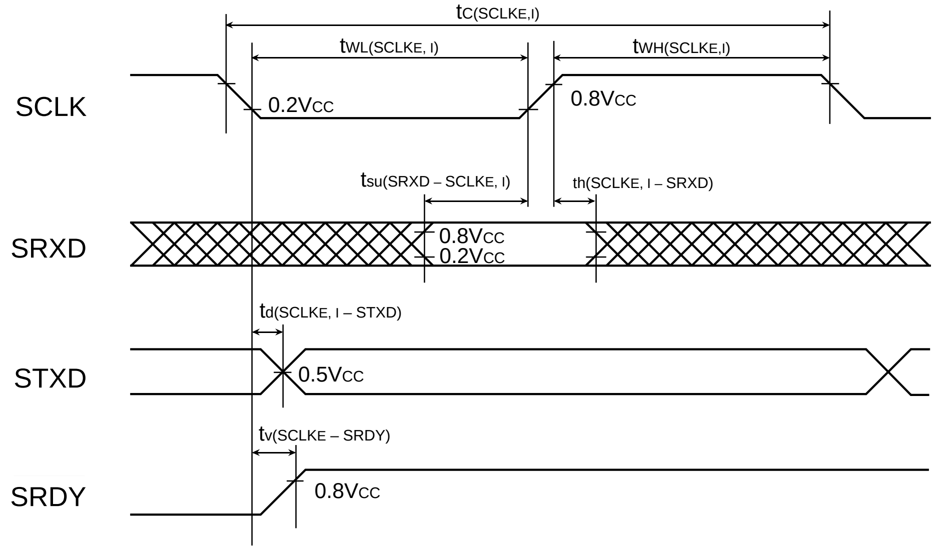
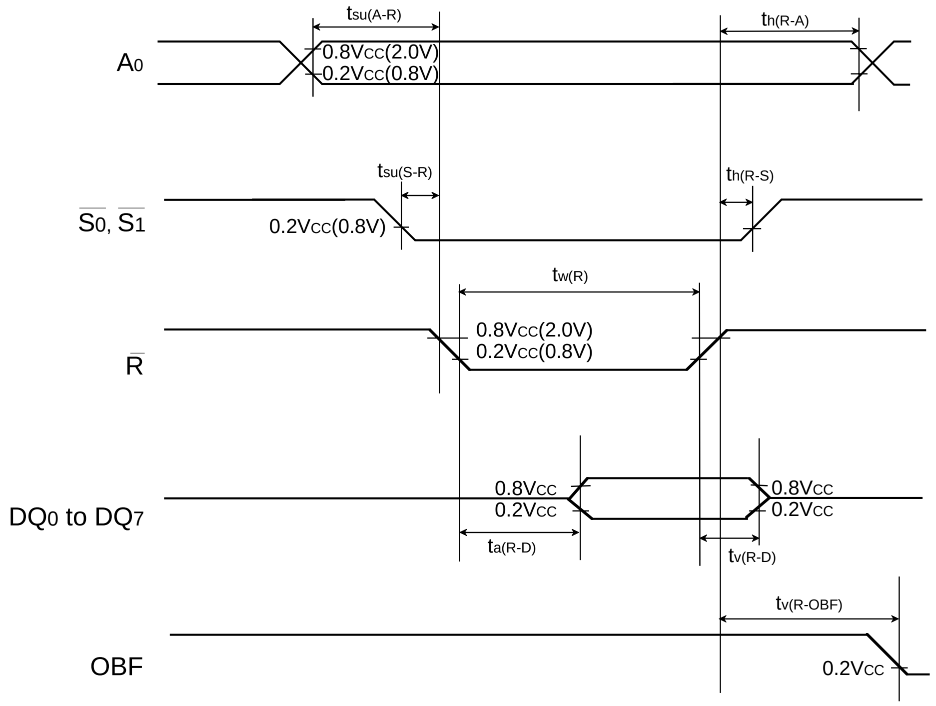


Fig. 3.1.4 Timing diagram (2)

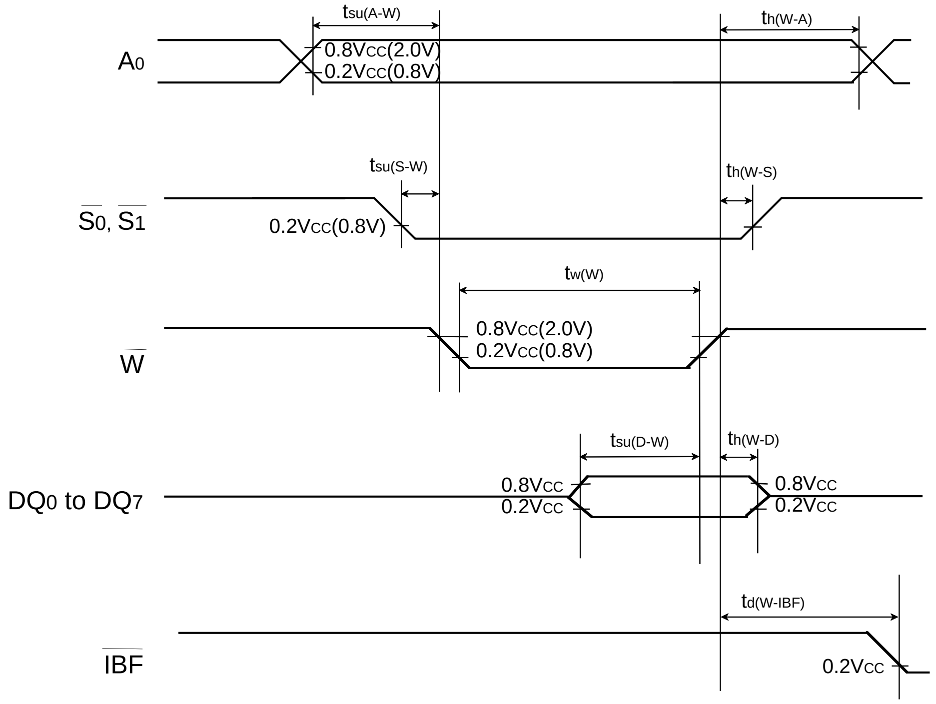
Fig. 3.1.5 Timing diagram (3)

● Timing diagram

[Master CPU bus interface: R/$\overline{W}$ separate mode]
<Read>



<Write>

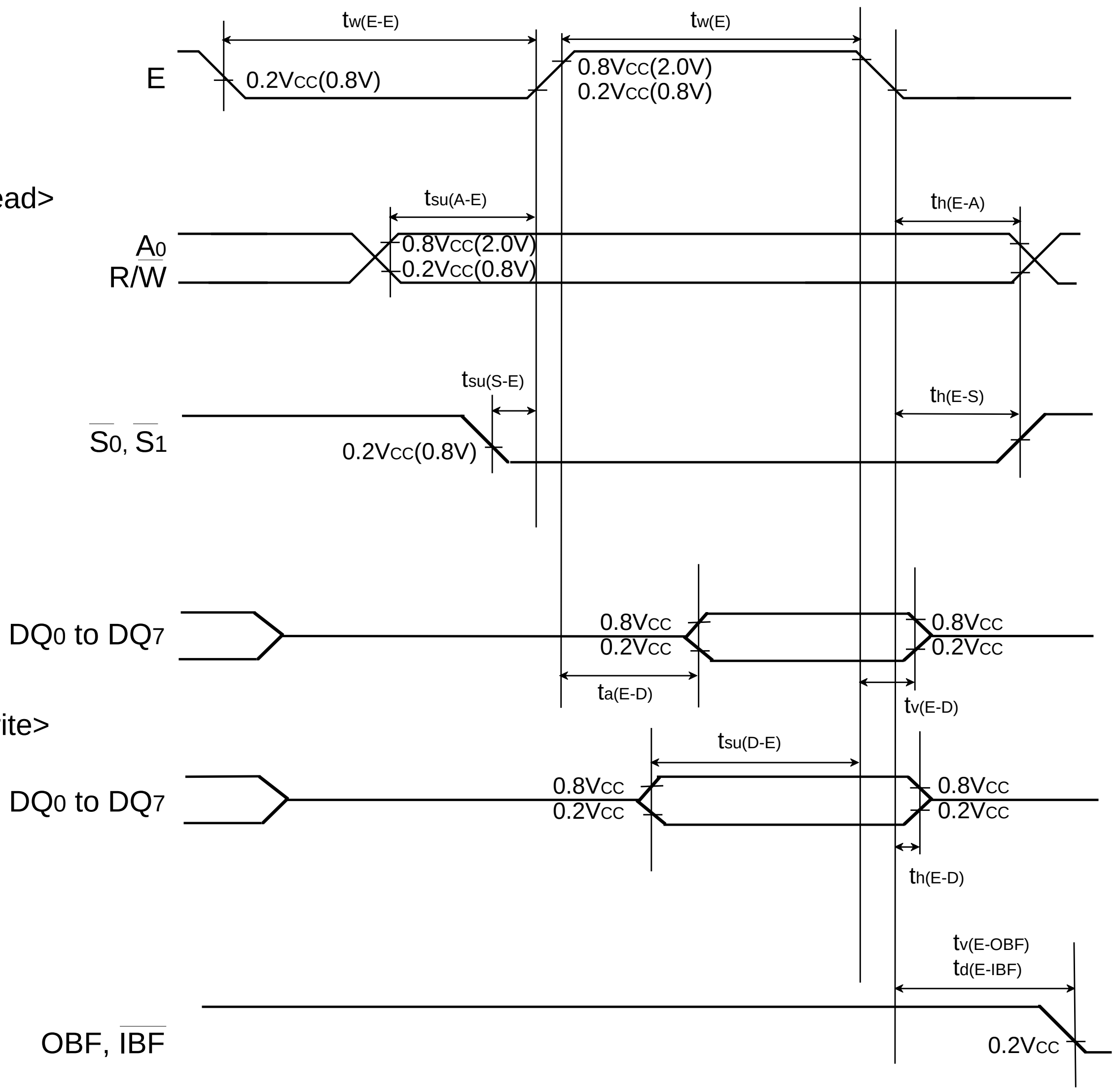


Note: This timing applies in the case of the master bus input level select bit (PTC7) = “1” (TTL level input)

Fig. 3.1.6 Timing diagram (4)

● Timing diagram

[Master CPU bus interface: R/$\overline{W}$ mode]



Note: This timing applies in the case of the master bus input level select bit (PTC7) = “1” (TTL level input)

Fig. 3.1.7 Timing diagram (5)

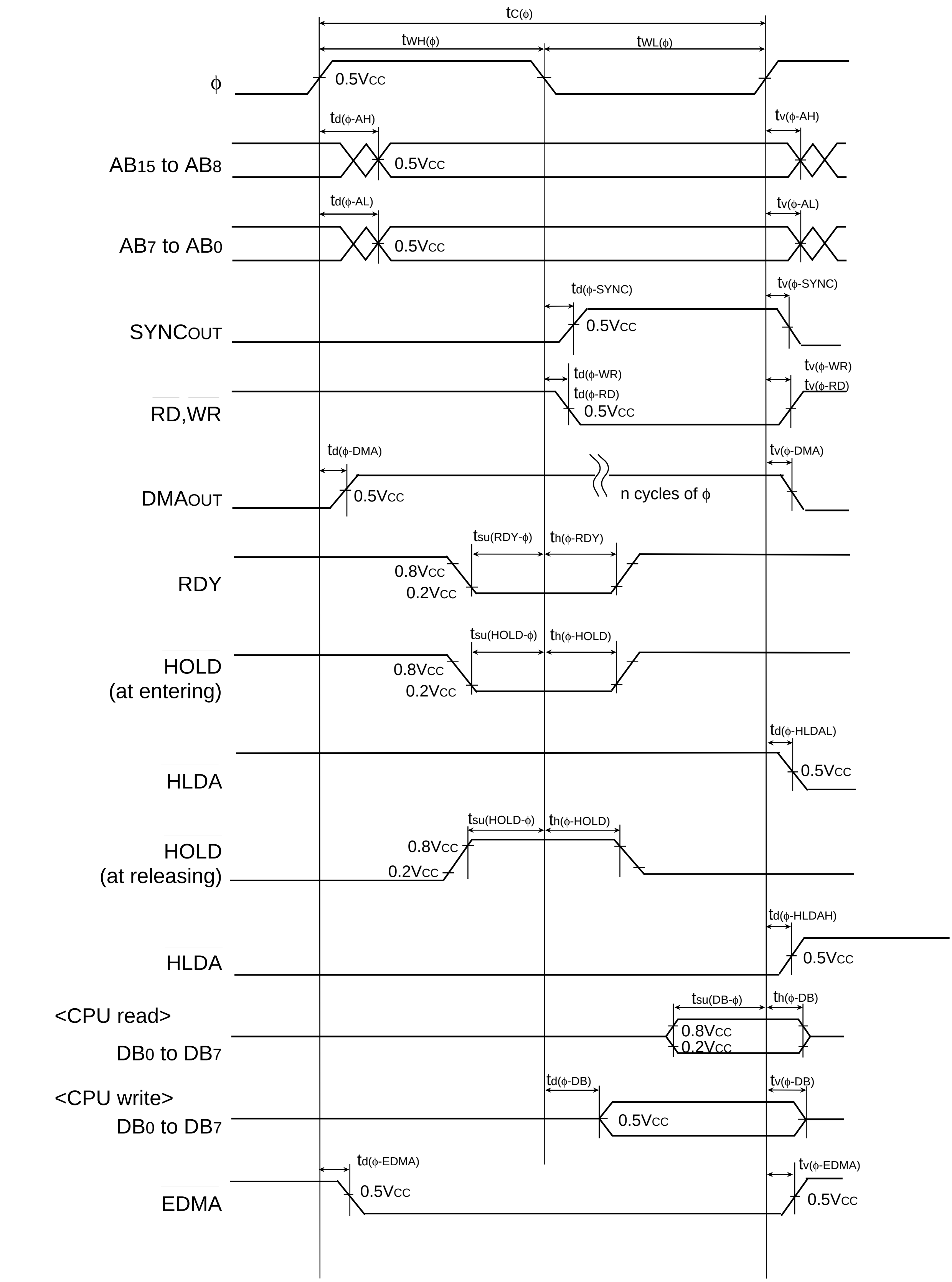


Fig. 3.1.8 Timing diagram (6); Memory expansion and microprocessor modes

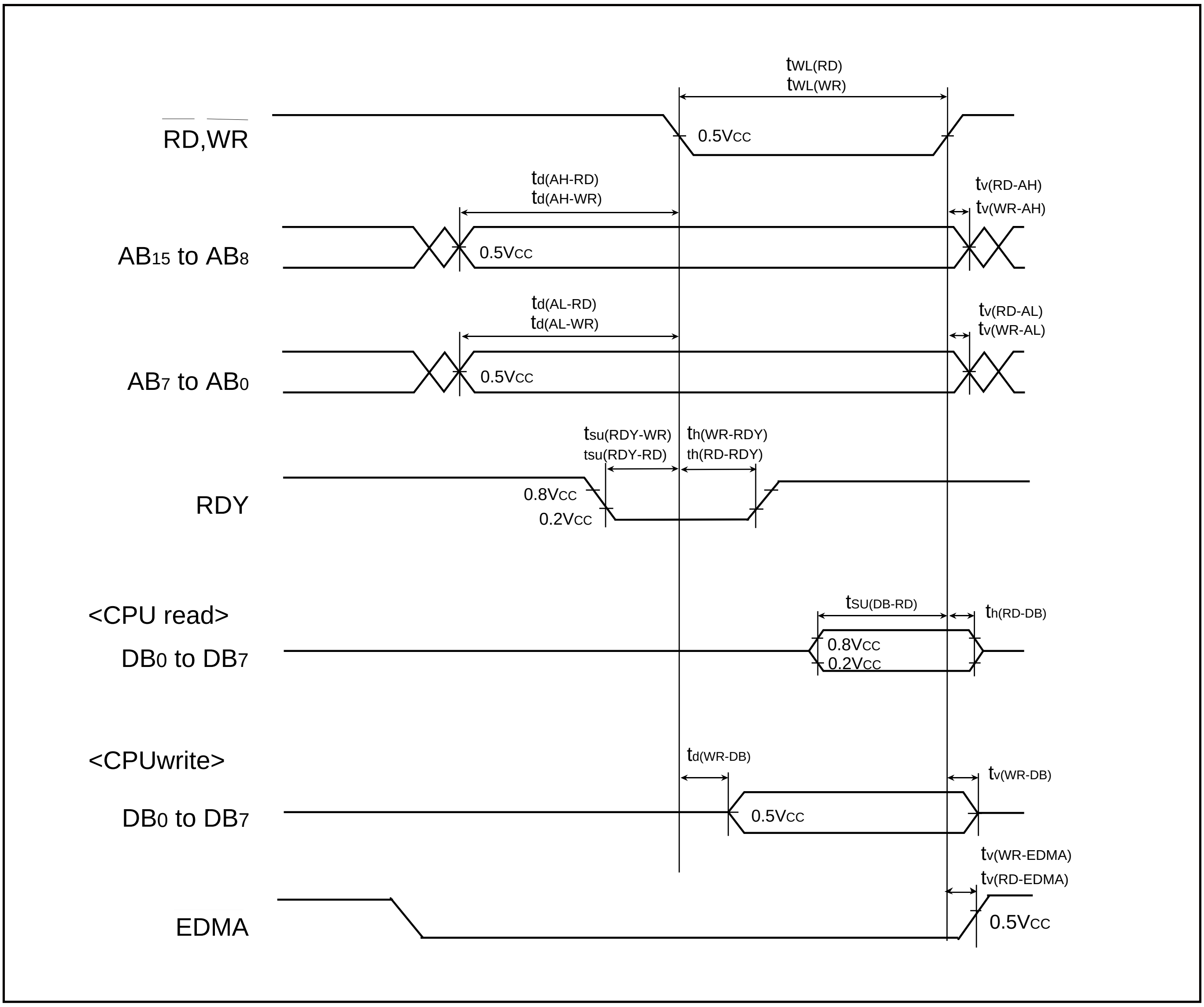


Fig. 3.1.9 Timing diagram (7); Memory expansion and microprocessor modes

3.2 Standard characteristics

Standard characteristics described below are just examples of the 7641 Group's characteristics and are not guaranteed. For rated values, refer to **"3.1 Electrical characteristics"**.

3.2.1 Power source current standard characteristics

Figure 3.2.1 shows power source current standard characteristics.

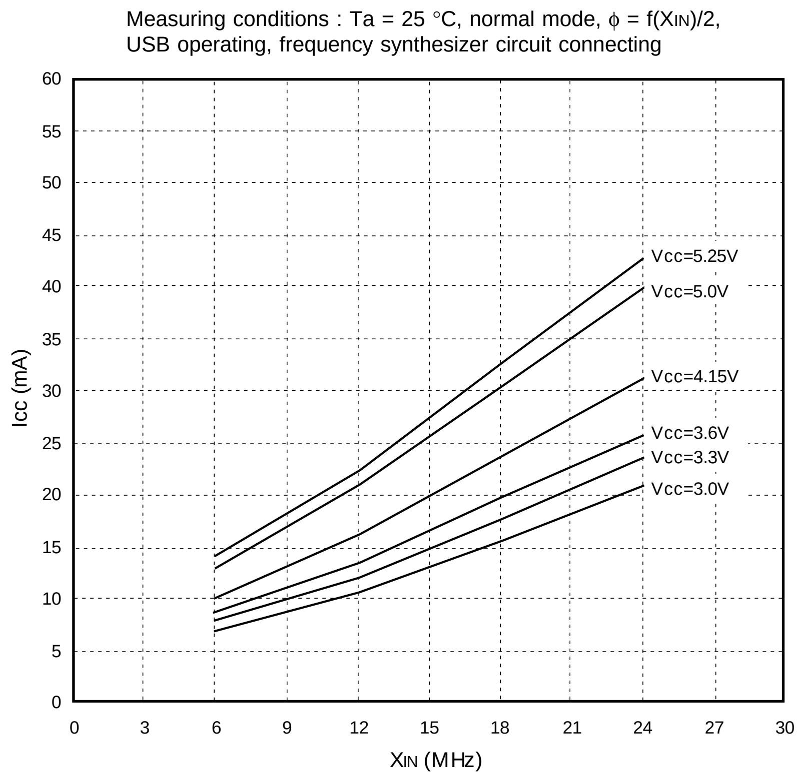


Fig. 3.2.1 Power source current standard characteristics (Ta = 25 °C)

3.2.2 Port standard characteristics

Figure 3.2.2 to Figure 3.2.7 show port standard characteristics.

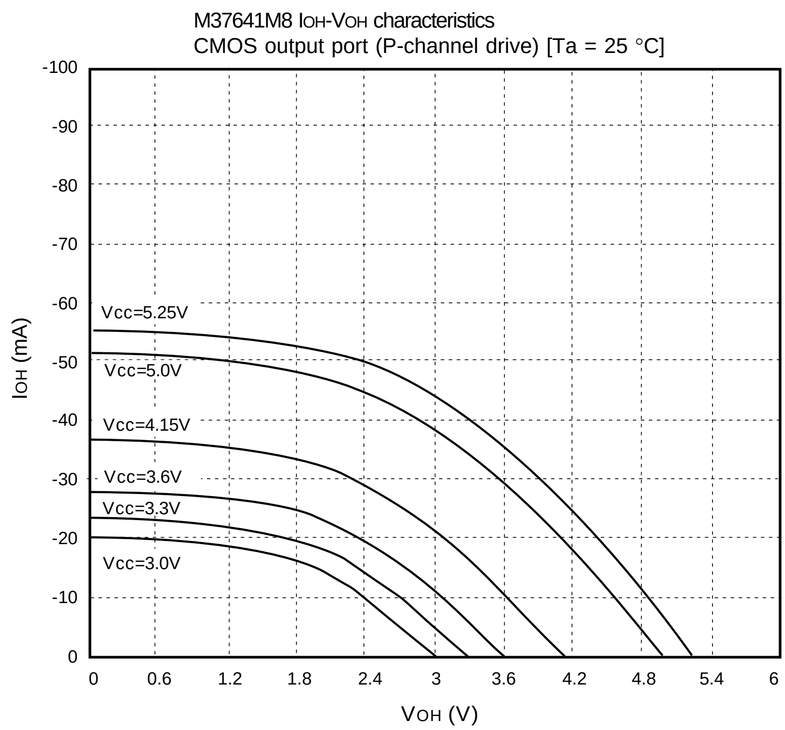


Fig. 3.2.2 CMOS output port P-channel side characteristics (Ta = 25 °C)

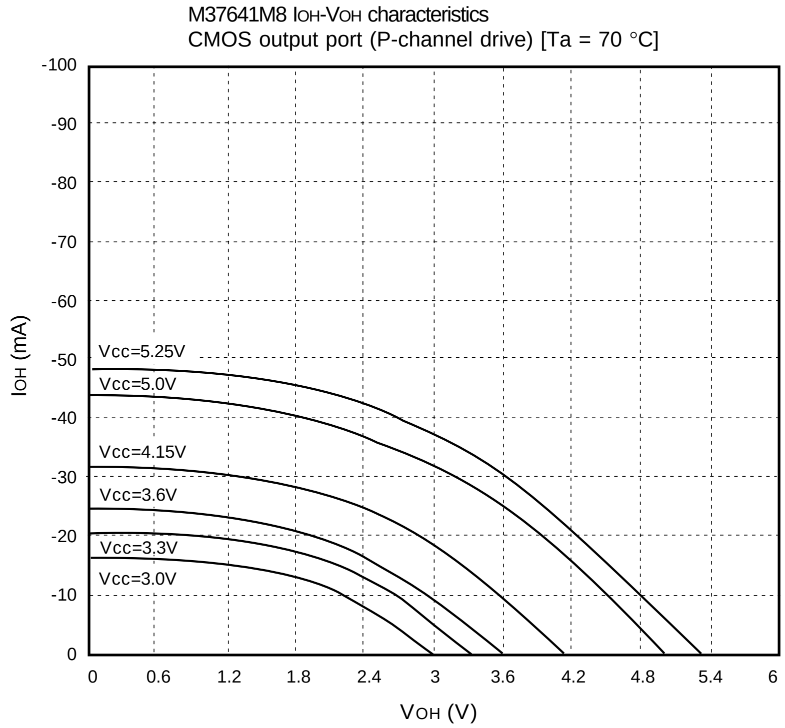


Fig. 3.2.3 CMOS output port P-channel side characteristics (Ta = 70 °C)

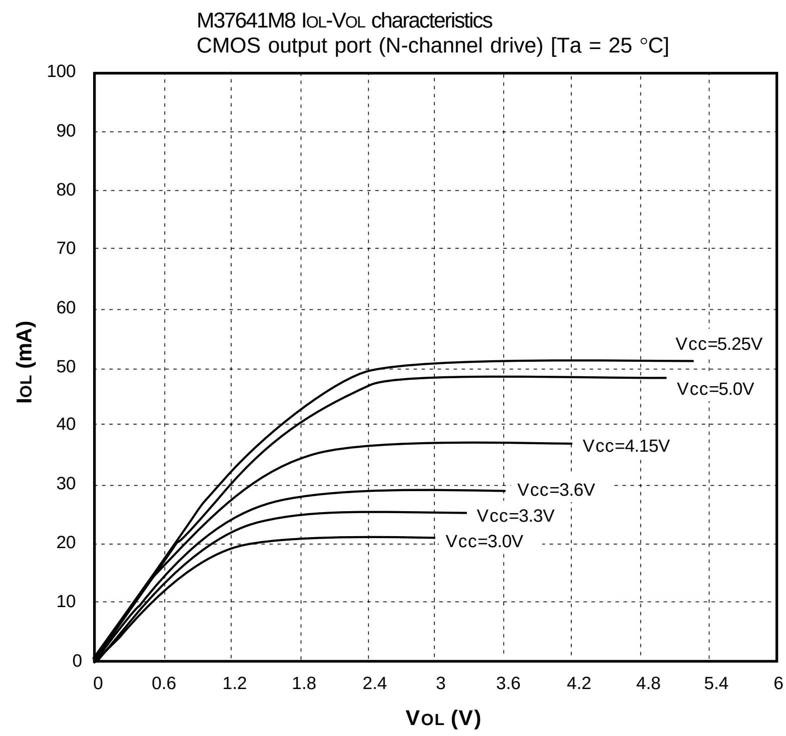


Fig. 3.2.4 CMOS output port N-channel side characteristics (Ta = 25 °C)

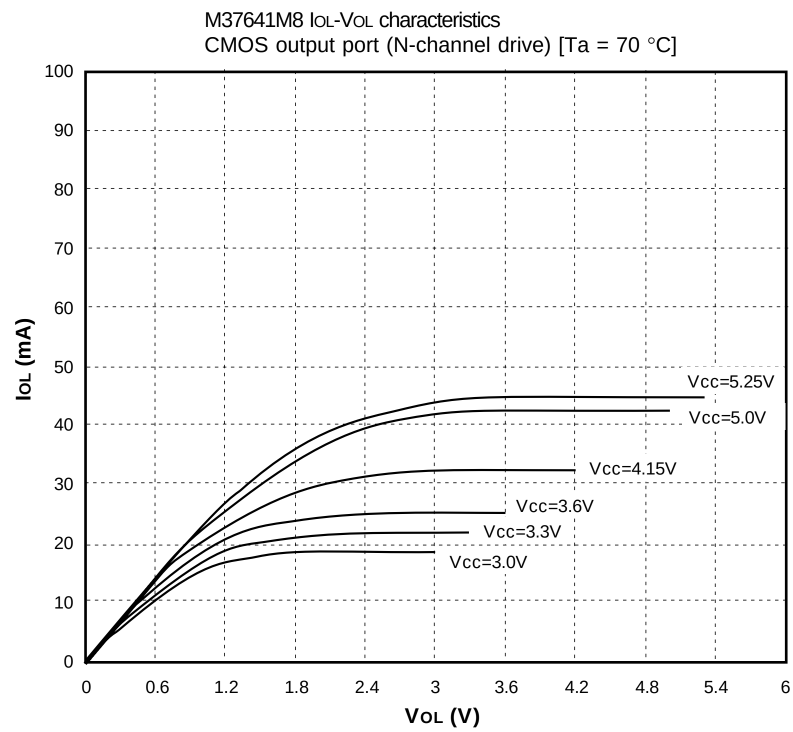


Fig. 3.2.5 CMOS output port N-channel side characteristics (Ta = 70 °C)

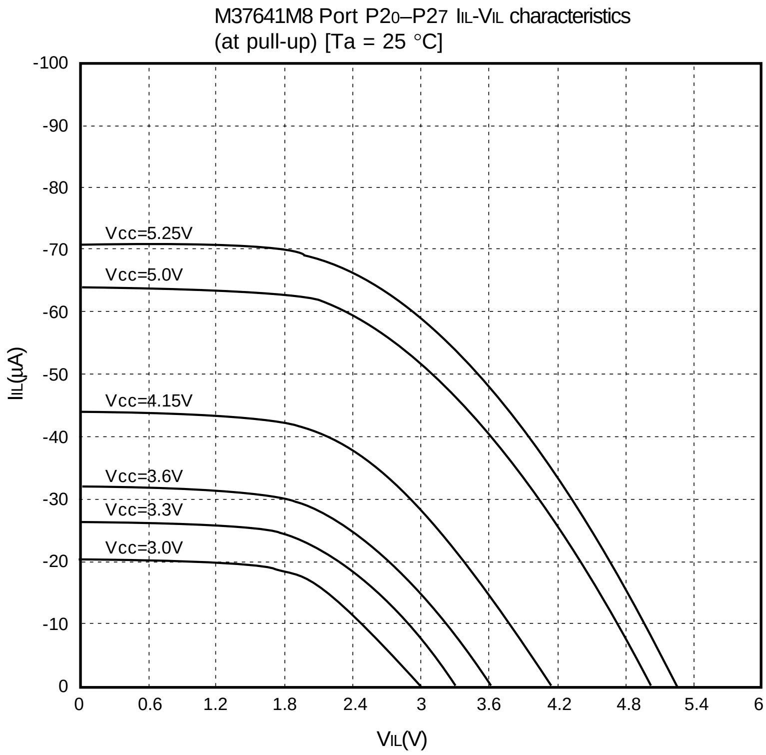


Fig. 3.2.6 Port $P2_0$–$P2_7$ at pull-up characteristics (Ta = 25 °C)

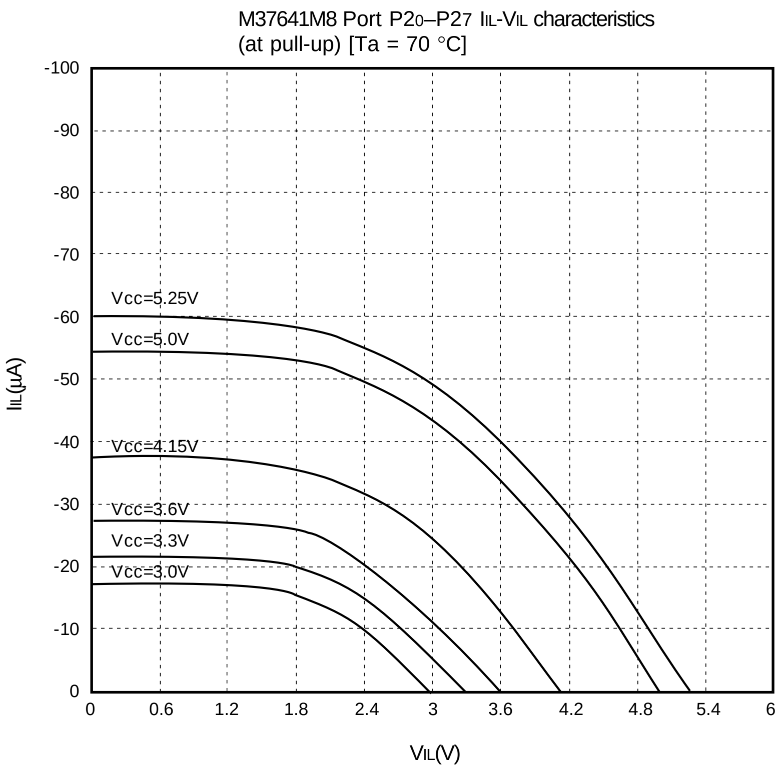


Fig. 3.2.7 Port $P2_0$–$P2_7$ at pull-up characteristics (Ta = 70 °C)

3.3 Notes on use

3.3.1 Notes on interrupts

(1) When setting external interrupt active edge

When setting the external interrupt active edge (INT_0, INT_1, $CNTR_0$, $CNTR_1$), the interrupt request bit may be set to “1”. When not requiring the interrupt occurrence synchronized with these setting, take the following sequence.

•Interrupt polarity select register (address 0011_{16})
•Timer X mode register (address 0027_{16})
•Timer Y mode register (address 0028_{16})

Set the above listed registers or bits as the following sequence.

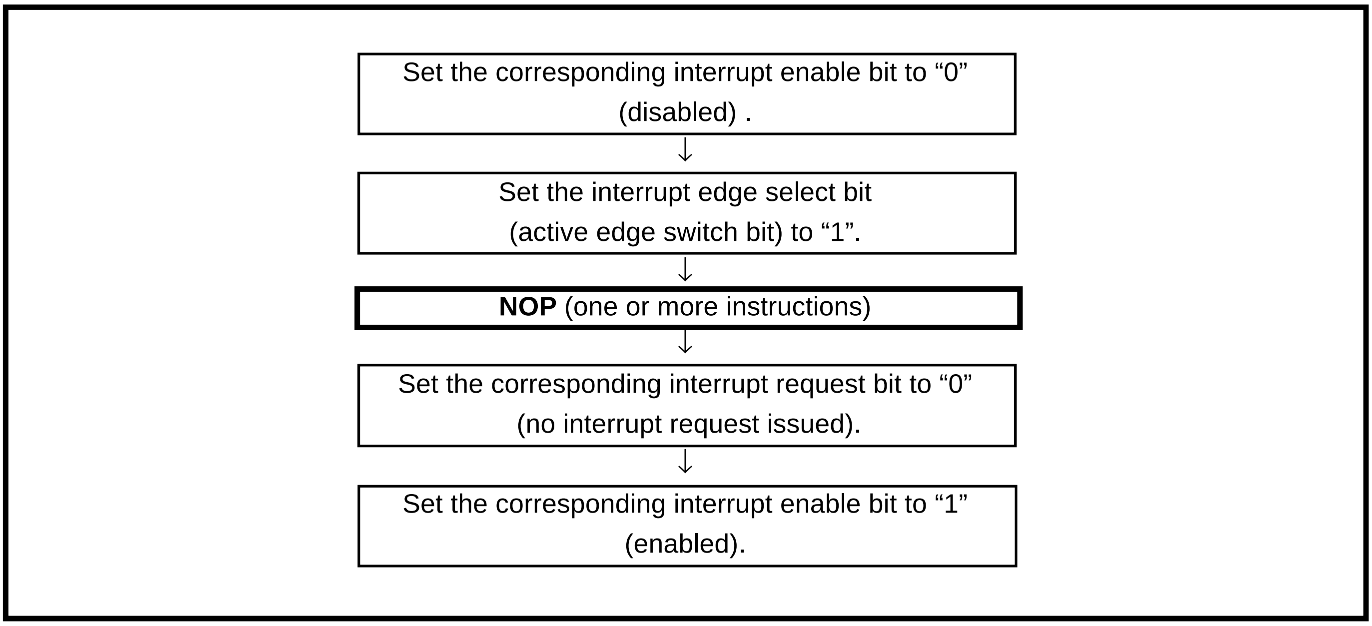


Fig. 3.3.1 Sequence of setting external interrupt active edge

3.3.2 Notes on serial I/O

(1) Clock

When the external clock is selected as the transfer clock, its transfer clock needs to be controlled by the external source because the serial I/O shift register will keep being shifted while transfer clock is input even after transfer completion.

(2) Reception

When the external clock is selected as the transfer clock for reception, the receiving operation will start owing to the shift clock input even if write operation to the serial I/O shift register (SIOSHT) is not performed. The serial I/O interrupt request also occurs at completion of receiving. However, we recommend to write dummy data in the serial I/O shift register. Because this will cause followings and improve transfer reliability.

•Write to SIOSHT puts the $\overline{\text{SRDY}}$ pin to “L”. This enables shift clock output of an external device.
•Write to SIOSHT clears the internal serial I/O counter.

Note: Do not read the serial I/O shift register which is shifting. Because this will cause incorrect-data read.

(3) STXD output

•When the internal clock is selected as the transfer clock, the STXD pin goes a high-impedance state after transfer completion.
•When the external clock is selected as the transfer clock, the STXD pin does not go a high-impedance state after transfer completion.

(4) SPI compatible mode

•When using the SPI compatible mode, set the $\overline{\text{SRDY}}$ select bit to “1” ($\overline{\text{SRDY}}$ signal output).
•When the external clock is selected in SPI compatible mode, the SRXD pin functions as a data output pin and the STXD pin functions as a data input pin.
•Do not write to the serial I/O shift register (SIOSHT) during a transfer as slave when in SPI compatible mode.
•Master operation of SPI compatible mode requires the timings:
-From write operation to the SIOSHT to $\overline{\text{SRDY}}$ pin put to “L”
Requires 2 cycles of internal clock ϕ + 2 cycles of serial I/O synchronous clock + 35 ns
-From $\overline{\text{SRDY}}$ pin put to “L” to SCLK switch
Requires 35 ns
-From the last pulse of SCLK to $\overline{\text{SRDY}}$ pin put to “H”
Requires 35 ns.

3.3.3 Notes on UART

(1) Receive

- When any one of errors occurs, the summing error flag is set to “1” and the UARTx summing error interrupt request bit is also set to “1”. If a receive error occurs, the reception does not set the UARTx receive buffer full interrupt request bit to “1”.
- If the receive enable bit (REN) is set to “0” (disabled) while a data is being received, the receiving operation will stop after the data has been received.
- Setting the receive initialization bit (RIN) to “1” resets the UARTx RTS control register (UxRTS) to “80_{16}”. After setting the RIN bit to “1”, set this UxRTS.

(2) Transmit

- Once the transmission starts, it continues until the last bit has been transmitted even though clearing the transmit enable bit (TEN) to “0” (disabled) or inputting “H” to the $\overline{CTSx}$ pin. After completion of the current transmission, the transmission is disabled.
- The transmit complete flag (TCM) is changed from “1” to “0” later than 0.5 to 1.5 clocks of the shift clock. Accordingly, take it in consideration to transmit data confirming the TCM flag after the data is written into the transmit buffer register.

(3) Register settings

- If updating a value of UARTx baud rate generator while the data is being transmitted or received, be sure to disable the transmission and reception before updating. If the former data remains in the UARTx transmit buffer registers 1 and 2 at retransmission, an undefined data might be output.
- The all error flags PER, FER, OER and SER are cleared to “0” when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit. These flags are also cleared to “0” by execution of bit test instructions such as **BBC** and **BCS**.
- The transmit buffer empty flag (TBE) is set to “0” when the low-order byte of transmitted data is written into the UARTx (x = 1, 2) transmit buffer register 1. When using 9-bit character length, set the data into the UARTx transmit buffer register 2 (high-order byte) first before the UARTx transmit buffer register 1 (low-order byte).
- The receive buffer full flag (RBF) is set to “0” when the contents of UARTx receive buffer register 1 is read out. When using 9-bit character length, read the data from the UARTx receive buffer register 2 (high-order byte) first before the UARTx receive buffer register 1 (low-order byte).
- If a character bit length is 7 bits, bit 7 of the UARTx transmit/receive buffer register 1 and bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
  If a character bit length is 8 bits, bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
  If a character bit length is 9 bits, bits 1 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are “0” at receiving.
- The reset cannot affect the contents of baud rate generator.

(4) UART address mode

- When the MSB of the incoming data is “0” in the UART address mode, the receive buffer full flag (RBF) is set to “1”, but the receive buffer full interrupt request bit is not set to “1”.
- An overrun error cannot be detected after the first data has been received in UART address mode.
- The UART address mode can be used in either an 8-bit or 9-bit character length. 7-bit character length cannot be used.

(5) Receive error flag

•The all error flags PER, FER, OER and SER are cleared to “0” when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit. Accordingly, note that these flags are also cleared to “0” by execution of bit test instructions such as BBC and BBS, not only LDA.

(6) $\overline{\text{CTS}}$ function

•When the $\overline{\text{CTS}}$ function is enabled, the transmitted data is not transferred to the transmit shift register until “L” is input to the $\overline{\text{CTSx}}$ pin ($P8_6/\overline{CTS_1}$, $P8_2/\overline{CTS_2}$/SRXD). As the result, do not set the following data to the transmit buffer register.

(7) $\overline{\text{RTS}}$ function

•If the start bit is detected in the term of “H” assertion of $\overline{\text{RTS}}$, its assertion count is suspended and the $\overline{\text{RTSx}}$ pin remains “H” output. After receiving the last stop bit, the count is resumed.
•Setting the receive initialization bit (RIN) to “1” resets the UARTx RTS control register (UxRTS) to “80_{16}”. After setting the RIN bit to “1”, set this UxRTS.

(8) Interrupt

•When setting the transmit initialization bit (TIN) to “1”, both the transmit buffer empty flag (TBE) and the transmit complete flag (TCM) are set to “1”, so that the transmit interrupt request occurs independent of its interrupt source. After setting the transmit initialization bit (TIN) to “1”, clear the transmit interrupt request bit to “0” before setting the transmit enable bit (TEN) to “1”.
•The transmit interrupt request bit is set and the interrupt request is generated by setting the transmit enable bit (TIN) to “1” even when selecting timing that either of the following flags is set to “1” as timing where the transmission interrupt is generated:
(1) Transmit buffer empty flag is set to “1”
(2) Transmit complete flag is set to “1”.

Therefore, when the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as the following sequence:
(1) Transmit enable bit is set to “1”
(2) Transmit interrupt request bit is set to “0”
(3) Transmit interrupt enable bit is set to “1”.

3.3.4 Notes on DMAC

(1) Transfer time

•One-byte data transfer requires 2 cycles of ϕ (read and write cycles).

•To perform DMAC transfer due to the different transfer requests on the same DMAC channel or DMAC transfer between both DMAC channels, 1 cycle of ϕ or more is needed before transfer is started.

(2) Priority

•The DMAC places a higher priority on channel-0 transfer requests than on channel-1 transfer requests.

If a channel-0 transfer request occurs during a channel-1 burst transfer operation, the DMAC completes the next transfer source and destination read/write operation first, and then stops the channel-1 transfer operation.

The channel-1 transfer operation which has been suspended is automatically resumed from the point where it was suspended so that channel-1 transfer can complete its one-burst transfer unit. This will be performed even if another channel-0 transfer request occurs.

•The suspended transfer due to the interrupt can also be resumed during its interrupt process routine by writing “1” to the DMAC channel x enable bit (DxCEN).

(3) Related registers

•A read/write must be performed to the source registers, transfer destination registers and transfer count registers as follows:

Read from each higher byte first, then the lower byte

Write to each lower byte first, then the higher byte.

Note that if the lower byte is read out first, the values are the higher byte’s.

•Do not access the DMAC-related registers by using a DMAC transfer. The destination address data and the source address data will collide in the DMAC internal bus.

•When setting the DMAC channel x enable bit (bit 7 of address 41_{16}) to “1”, be sure simultaneously to set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of address 41_{16}) to “1”. If this is not performed, an incorrect data will be transferred at the same time when the DMAC is enabled.

(4) USB transfer

One signal among USB endpoint signals 1 to 4 can be selected as the hardware transfer request source. This can realize that transfer between the USB FIFO and the master CPU bus interface input/output buffer is performed effectively. This transfer function is only valid in the cycle steal transfer mode.

(5) DMA_{OUT} pin

In the memory expansion mode and microprocessor mode, the DMA_{OUT} pin ($P3_3/DMA_{OUT}$) outputs “H” during a DMA transfer.

3.3.5 Notes on USB

(1) USB reception

- When reading the USB endpoint x (x = 0 to 4) OUT write count registers, the lower byte must be read first, and then the higher byte.
- When the OUT FIFO contains 2-data packets in the endpoints 1 to 4 used, one-data packet will still remain in the OUT FIFO even after the data of the OUT max. packet size has been read. In this case the OUT_PKT_RDY flag is not cleared even if it is set to “0”. (The flag returns from “0” to “1” 83 ns later (Vcc = 5 V, $f(X_{IN})$ = 24 MHz) than the clearing.)
- Read one packet data from the OUT FIFO before clearing the OUT_PKT_RDY flag. If the OUT_PKT_RDY flag is cleared while one-data packet is being read, the internal read pointer cannot operate normally.

(2) USB transmission

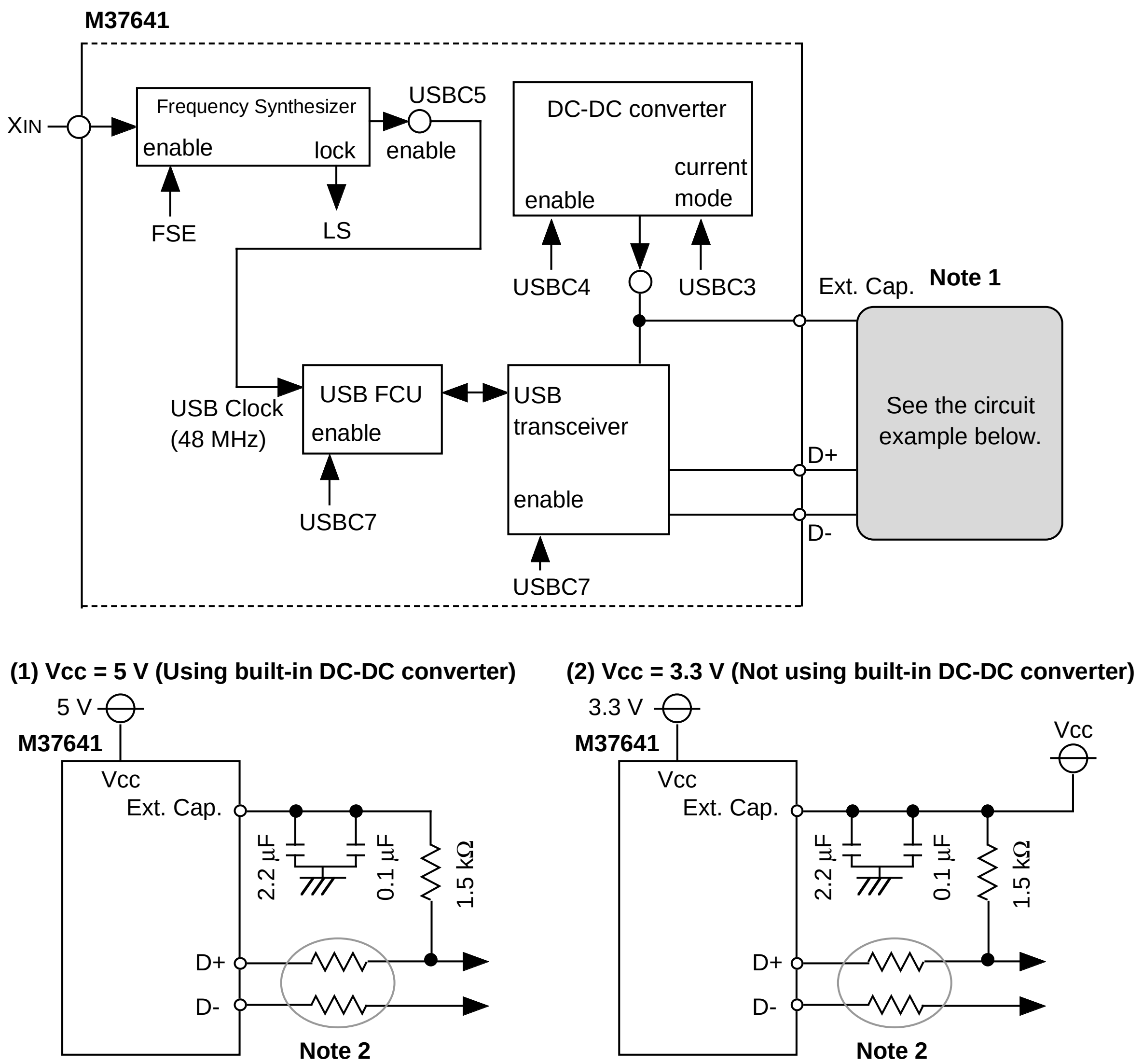
- The IN FIFO status can be checked by monitoring the IN_PKT_RDY bit and the TX_NOT_EPT flag.
- When NULL packet transmission is required in the endpoints 1 to 4 used, perform it under the conditions of the IN_PKT_RDY bit set to “1” and no data in FIFO.

(3) External circuit

- Connect a capacitor between the Ext. Cap. pin and the Vss pin. The capacitor should have a 2.2 µF capacitor (Tantalum capacitor) and a 0.1 µF capacitor (ceramic capacitor) connected in parallel. Additionally, connect a 1.5 kΩ (± 5 %) resistor between the Ext. Cap. pin and the D+ pin.
- The Full-Speed USB2.0 specification requires a driver -impedance 28 to 44 Ω. (Refer to Clause 7.1.1.1 Full-speed (12 Mb/s) Driver Characteristics in the USB specification.) In order to meet the USB specification impedance requirements, connect a resistor (27 to 33 Ω recommended) in series to the USB D+ pin and the USB D- pin.
  In addition, in order to reduce the ringing and control the falling/rising timing of USB D+/D- and a crossover point, connect a capacitor between the USB D+/D- pins and the Vss pin if necessary.
  The values and structure of those peripheral elements depend on the impedance characteristics and the layout of the printed circuit board. Accordingly, evaluate your system and observe waveforms before actual use and decide use of elements and the values of resistors and capacitors.
  Figure 3.3.2 shows the circuit example for the proper positions of the peripheral components.
- In Vcc = 3.3 V operation, connect the Ext. Cap. pin directly to the Vcc pin in order to supply power to the USB transceiver. In addition, you will need to disable the DC-DC converter in this operation (set bit 4 of the USB control register to “0”.) If you are using the bus powered supply in Vcc = 3.3 V operation, the DC-DC converter must be placed outside the MCU.
- In Vcc = 5 V operation, do not connect the external DC-DC converter to the Ext. Cap. pin. Use the built-in DC-DC converter by enabling the USB line driver.
- Make sure the USB D+/D- lines do not cross any other wires. Keep a large GND area to protect the USB lines. Also, make sure you use a USB specification compliant connecter for the connection.
- All passive components must be located as close as possible to the LPF pin. Figure. 3.3.3 shows the passive components near LPF pin
- An insulation connector (Ferrite Beads) must be connected between AVss and Vss pins and between AVcc and Vcc pins. (See Figure 3.3.4.)

(4) USB Communication

- In applications requiring high-reliability, we recommend providing the system with protective measures such as USB function initialization by software or USB reset by the host to prevent USB communication from being terminated unexpectedly, for example due to external causes such as noise.



Notes 1: In Vcc = 3.3 V, connect to Vcc. In Vcc =5 V, do not connect the external DC-DC converter to the Ext. Cap pin.

2: The resistors values depend on the layout of the printed circuit board.

Fig. 3.3.2 Circuit example for the proper positions of the peripheral components

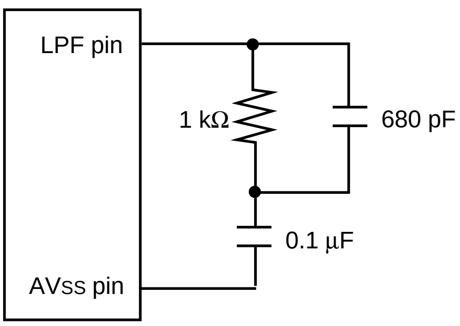


Fig. 3.3.3 Passive components near LPF pin

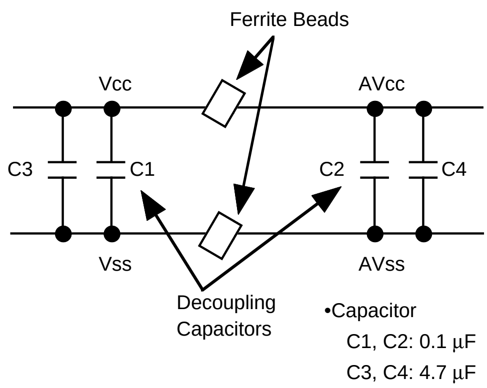


Fig. 3.3.4 Insulation connector connection

(5) Registers and bits

- •When using the endpoint 0, use the USB endpoint 0 IN max. packet size register for transmission and reception (IN packet size and OUT packet size).
- •When not using the USB endpoint x (x = 0 to 4) IN max. packet size register and USB endpoint x OUT max. packet size register, set them to “0”.
- •To write to/read from the USB interrupt status registers 1 and 2, perform it for the USB interrupt status register 1 first and then the register 2.
- •To read from the USB endpoint x (x = 0 to 4) OUT write count registers Low and High, the lower byte must be read first, then the higher byte.
- •Make sure the index indicated by the USB endpoint index register is correct when accessing the registers: USB endpoint x (x = 0 to 4) IN control register, USB endpoint x OUT control register, USB endpoint x IN max. packet size register, USB endpoint x OUT max. packet size register, USB endpoint x (x = 0 to 4) OUT write count registers Low and High, USB endpoint FIFO mode register.
- •When the USB reset interrupt status flag is kept at “1”, all other flags in the USB internal registers (addresses 0050_{16} to $005F_{16}$) will return to their reset status. However, the following registers are not affected by the USB reset: USB control register (address 0013_{16}), Frequency synthesizer control register (address $006C_{16}$), Clock control register (address $001F_{16}$), and USB endpoint x FIFO register (addresses 0060_{16} to 0064_{16}).
- •When not using the USB function, set the USB line driver supply enable bit of the USB control register (address 0013_{16}) to “1” for power supply to the internal circuits (at Vcc = 5 V).
- •The IN_PKT_RDY Bit can be set by software even when using the AUTO_SET function.
- •Do not write to USB-related registers (addresses 0050_{16} to 0064_{16}) except the USBC, CCR and FSC until the USB clock is enabled.
- •When the MCU is in the USB-suspend state, the USB enable bit is kept “1”; the USB block is enabled. To write to USB-related registers (addresses 0050_{16} to 0064_{16}) except the USBC, CCR and FSC after returning from the USB-suspend state; after enabling the USB clock, wait for 4 or more ϕ cycles and then set those registers.
- •When using the MCU at Vcc = 3.3V, set the USB line driver supply enable bit to “0” (line driver disable). Note that setting the USB line driver current control bit (USBC3) doesn’t affect the USB operation.
- •Setting the FLUSH bit to “1” eliminates the data in IN FIFO and OUT FIFO. If there are 2 or more-data packets in them, the oldest data is eliminated. The FLUSH bit setting also affects the IN_PKT_RDY bit or the OUT_PKT_RDY flag.
- •If the FLUSH bit is set to “1” while transmission/reception is being performed, the data might be corrupted. When receiving, setting the FLUSH bit must be done while the OUT_PKT_RDY flag is “1”. When transmitting in the isochronous transfer mode, use the AUTO_FLUSH function.
- •Use the AUTO_FLUSH bit (bit 6 of address 58_{16}) in the double buffer mode.

•Use the transfer instructions such as **LDA** and **STA** to set the registers: USB interrupt status registers 1, 2 (addresses 0052_{16}, 0053_{16}); USB endpoint 0 IN control register (address 0059_{16}); USB endpoint x IN control register (address 0059_{16}); USB endpoint x OUT control register (address $005A_{16}$). Do not use the read-modify-write instructions such as the **SEB** or the **CLB** instruction.
When writing to bits shown by Table 32 using the transfer instruction such as **LDA** or **STA**, a value which never affect its bit state is required. Take the following sequence to change these bits contents:
(1) Store the register contents onto a variable or a data register.
(2) Change the target bit on the variable or the data register. Simultaneously mask the bit so that its bit state cannot be changed. (See to Table 3.3.1.)
(3) Write the value from the variable or the data register to the register using the transfer instruction such as **LDA** or **STA.**

•To use the AUTO_SET function for an IN transfer when the AUTO_SET bit is set to 1, set the FIFO to single buffer mode.

Table 3.3.1 Bits of which state might be changed owing to software write

Register name	Bit name	Value not affecting state (**Note**)
USB endpoint 0 IN control register	IN_PKT_RDY (b1)	“0”
	DATA_END (b3)	“0”
	FORCE_STALL (b4)	“1”
USB endpoint x (x = 1 to 4) IN control register	IN_PKT_RDY (b0)	“0”
	UNDER_RUN (b1)	“1”
USB endpoint x (x = 1 to 4) OUT control register	OUT_PKT_RDY (b0)	“1”
	OVER_RUN (b1)	“1”
	FORCE_STALL (b4)	“1”
	DATA_ERR (b5)	“1”

Note: Writing this value will not change the bit state, because this value cannot be written to the bit by software.

(6) Others

•When the USB SOF Port Select Bit is “1”, the reference pulse of 83.3 ns (ϕ = 12 MHz) is output from the $P7_0$/SOF pin and synchronized with the SOF packet.

3.3.6 Notes on frequency synthesizer

•Bits 6 and 5 of the frequency synthesizer control register (address $006C_{16}$) are initialized to (b6, b5) = “11” after reset release. Make sure to set bits 6 and 5 to “10” after the frequency synthesizer lock status bit goes to “1”.
•Use the frequency synthesizer output clocks 2 ms to 5 ms later than setting the frequency synthesizer enable bit to “1” (enabled). After that do not change any register values because it might cause output clocks unstabilized temporarily.
•Make sure to connect a low-pulse filter to the LPF pin when using the frequency synthesizer.
•The frequency synthesizer divide register set value never affects f_{USB} frequency.
•When using the f_{SYN} as an internal system clock, set the frequency synthesizer divide register so that f_{SYN} could be 24 MHz or less.
•When using the frequency synthesized clock function, we recommend using the fastest frequency possible of $f(X_{IN})$ or $f(X_{CIN})$ as an input clock for the PLL.
•Set the value of frequency synthesizer multiply register 2 (FSM2) so that the f_{PIN} is 1 MHZ or higher.

3.3.7 Notes on master CPU bus interface

Be sure to set port P6 to input mode by setting the port P6 direction register to “0” when the master CPU bus interface is enabled.

3.3.8 Notes on external devices connection

(1) Rewrite port P3 latch

In both memory expansion mode and microprocessor mode, ports $P3_1$ and $P3_2$ can be used as output ports. We recommend to use the **LDM** instruction or **STA** instruction to write to port P3 register (address $000E_{16}$). If using the Read-Modify-Write instruction (**SEB** instruction, **CLB** instruction) you will need to map a memory that the CPU can read from and write to.

[Reason]

The access to address $000E_{16}$ is performed:
•Read from external memory
•Write to both port P3 latch and external memory.
It is because address $000E_{16}$ is assigned on an external area In the memory expansion mode and microprocessor mode.
Accordingly, if a Read-Modify-Write instruction is executed to address $000E_{16}$, the external memory contents is read out and after its modification it will be written into both port P3 latch and an external memory. As a result, if an external memory is not allocated in address $000E_{16}$ then, the MCU will read an undefined value and write its modified value into the port P3 latch. Therefore port P3 latch value will become undefined.

(2) overlap of internal and external memories

In the memory expansion mode, if the internal and external memory areas overlap, the internal memory becomes the valid memory for the overlapping area. When the CPU performs a read or a write operation on this overlapped area, the following things happen:
•Read
The CPU reads out the data in the internal memory instead of in the external memory. Note that, since the CPU will output a proper read signal, address signal, etc., the memory data at the respective address will appear on the external data bus.
•Write
The CPU writes data to both the internal and external memories.

(3) $\overline{RD}$, $\overline{WR}$ pins

In the memory expansion mode or microprocessor mode, a read-out control signal is output from the $\overline{RD}$ pin ($P3_6$), and a write-in control signal is output from the $\overline{WR}$ pin ($P3_7$). "L" level is output from the $\overline{RD}$ pin at CPU read-out and from the $\overline{WR}$ pin at CPU write-in. These signals function for internal access and external access.

(4) $\overline{HLDA}$ pin

In spite of enabling the Hold function, the $\overline{HLDA}$ pin does not function when IBF_1 output is enabled in the master CPU bus interface.

(5) RDY function

When using RDY function in usual connection, it does not operate at 12 MHz of ϕ or faster.

[Reason]

$td(\phi\text{-}AH) + tsu(RDY\text{-}\phi)$ = 31 ns (max.) + 21 ns (min.) = 52 ns.

$twh(\phi)$, $twl(\phi)$ = 0.5 × 83.33 – 5 = 36.665 ns

Therefore, it becomes 52 ns > 36.665 ns, so that the timing to enter RDY wait does not match.

However, if the timings can match owing to $\overline{RDY}$ pin by "L" fixation and others, the RDY function can be used even at ϕ = 12 MHz. In this situation the slow memory wait always functions.

(6) Wait function

The Wait function is serviceable at accessing an external memory in the memory expansion mode and microprocessor mode. However, in these modes even if an external memory is assigned to addresses 0008_{16} to $000F_{16}$, the Wait function cannot function to these areas.

(7) Processor mode switch

Note when the processor mode is switched by setting of the processor mode bits (b1, b0 of CPMA), that will immediately switch the accessible memory from external to internal or from internal to external. If this is done, the first cycle of the next instruction will be operated from the accidental memory.

To prevent this problem, follow the procedure below:

(a) Duplicate the next instruction at the same address both in internal and external memories.

(b) Switch from single-chip mode to memory expansion mode, jump to external memory, and then switch from memory expansion mode to microprocessor mode. (Because in general, the problem will not occur when switching the modes as long as the same memory is accessed after the switch.

(c) Load a simple program in RAM that switches the modes, jump to RAM and execute the program, then jump to the location of the code to run after the processor mode has switched.

3.3.9 Notes on timer

(1) Read/Write for timer

•The timer division ratio is : 1 / (n + 1)
(n = "0" to "255" written into the timer)

•Read and write operation on 16-bit timer (Timers X and Y) must be performed for both high and low-order bytes.

•When reading the 16-bit timer (Timers X and Y), read the high-order byte first and then the low-order byte. When writing to the 16-bit timer, write the low-order byte first and then the high-order byte.

•Do not read the 16-bit timer during the write operation, or do not write to it during the read operation.

•When the value is loaded only in the latch, the value is loaded in the timer at the count pulse following the count where the timer reaches "00_{16}".

•In the timers 1 to 3, switching of the count sources of timers 1 to 3 does not affect the values of reload latches. However, that may make count operation started. Therefore, write values again in the order of timers 1, 2 and then timer 3 after their count sources have been switched.

•In the timer mode (for timers X, Y, 1 to 3), event counter mode (for timers X, Y), pulse output mode (for timers X, Y, 1, 2), the timer current count value can be read out by reading the timer.

•In the pulse width measurement mode (for timer X), period measurement mode (for timer Y), pulse width HL continuously measurement mode (for timer Y), the measured timer value is stored into the internal temporary register. When reading the timer, the value of internal temporary register is read out. The contents of internal temporary register is updated after the next measurement.

(2) Pulse output

•When using the pulse output mode of timer X, set bit 3 of port P4 direction register to "1" (output mode).

•When using the TY_{OUT} output of timer Y, set bit 4 of port P4 direction register to "1" (output mode).

•When using the T_{OUT} output of timer 1 or timer 2, set bit 1 of port P5 direction register to "1" (output mode).

•The T_{OUT} output pin is shared with the X_{COUT} pin. Accordingly, when using $f(X_{CIN})/2$ as the timer 1 count source (bit 2 of timer 123 mode register = "0"), X_{COUT} oscillation drive must be disabled (bit 5 of clock control register = "1") to input clocks from the X_{CIN} pin.

•The $P5_1/X_{COUT}/T_{OUT}$ pin cannot function as an ordinary I/O port while X_{CIN}-X_{COUT} is oscillating. When X_{CIN}-X_{COUT} oscillation is stopped or X_{COUT} oscillation drive is disabled, this can be used as the T_{OUT} output pin of timer 1 or 2.

(3) Pulse input

•When using the timer X in the event counter or pulse width measurement mode, set bit 3 of port P4 direction register to "0" (input mode).

•When using the timer Y in the period measurement, event counter or pulse width HL continuously measurement mode, set bit 4 of port P4 direction register to "0" (input mode).

(4) Interrupt

In the timer Y's pulse width HL continuously measurement mode, $CNTR_1$ interrupt request is generated at both rising and falling edges of $CNTR_1$ pin input signal regardless of the setting of $CNTR_1$ active edge switch bit of timer Y mode register.

(5) At STP instruction executed

When the STP instruction is executed or Reset occurs, the timer 1 is set to "FF_{16}" and the internal clock ϕ divided by 8 is automatically selected as its count source. Additionally, the timer 2 is set to "01_{16}" and the timer 1's output is automatically selected as its count source. When the **STP** instruction is being executed, all bits except bit 4 of the timer 123 mode register (address 0029_{16}) are initialized to "0". It is not necessary to set T123M1 (timer 1 count stop bit) to "0" before executing the **STP** instruction. After returning from Stop mode, reset the timer 1 (address 0024_{16}), timer 2 (address 0025_{16}), and the timer 123 mode register (address 0029_{16}).

3.3.10 Notes on Stop mode

•When the STP instruction is executed, bit 7 of the clock control register (address $001F_{16}$) goes to "0". To return from stop mode, reset CCR7 to "1".

•When using f_{SYN} (set internal system clock select bit (CPMA6) to "1") as the internal system clock, switch CPMA6 to "0" before executing the **STP** instruction. Reset CPMA6 after the system returns from Stop Mode and the frequency synthesizer has stabilized.
CPMA6 does not need to be switched to "0" when using the **WIT** instruction.

•When the STP instruction is executed or Reset occurs, the timer 1 is set to "FF_{16}" and the internal clock ϕ divided by 8 is automatically selected as its count source. Additionally, the timer 2 is set to "01_{16}" and the timer 1's output is automatically selected as its count source. When the **STP** instruction is being executed, all bits except bit 4 of the timer 123 mode register (address 0029_{16}) are initialized to "0". It is not necessary to set T123M1 (timer 1 count stop bit) to "0" before executing the **STP** instruction. After returning from Stop mode, reset the timer 1 (address 0024_{16}), timer 2 (address 0025_{16}), and the timer 123 mode register (address 0029_{16}).

3.3.11 Notes on reset

(1) Connecting capacitor

In case where the $\overline{\text{RESET}}$ signal rise time is long, connect a ceramic capacitor or others across the $\overline{\text{RESET}}$ pin and the Vss pin. Use a 1000 pF or more capacitor for high frequency use. When connecting the capacitor, note the following :

• Make the length of the wiring which is connected to a capacitor as short as possible.
• Be sure to verify the operation of application products on the user side.

● **Reason**

If the several nanosecond or several ten nanosecond impulse noise enters the $\overline{\text{RESET}}$ pin, it may cause a microcomputer failure.

3.3.12 Notes on I/O port

(1) Notes in standby state

In standby state*1 for low-power dissipation, do not make input levels of an I/O port "undefined".
Pull-up (connect the port to Vcc) or pull-down (connect the port to Vss) these ports through a resistor.
When determining a resistance value, note the following points:

• External circuit
• Variation of output levels during the ordinary operation

When using built-in pull-up resistor, note on varied current values:

• When setting as an input port : Fix its input level
• When setting as an output port : Prevent current from flowing out to external

● **Reason**

The potential which is input to the input buffer in a microcomputer is unstable in the state that input levels of an I/O port are "undefined". This may cause power source current.

✻1 standby state: stop mode by executing **STP** instruction
wait mode by executing **WIT** instruction

(2) Modifying output data with bit managing instruction

When the port latch of an I/O port is modified with the bit managing instruction*2, the value of the unspecified bit may be changed.

● **Reason**

The bit managing instructions are read-modify-write form instructions for reading and writing data by a byte unit. Accordingly, when these instructions are executed on a bit of the port latch of an I/O port, the following is executed to all bits of the port latch.

•As for bit which is set for input port:
The pin state is read in the CPU, and is written to this bit after bit managing.

•As for bit which is set for output port:
The bit value is read in the CPU, and is written to this bit after bit managing.

Note the following:

•Even when a port which is set as an output port is changed for an input port, its port latch holds the output data.

•As for a bit of which is set for an input port, its value may be changed even when not specified with a bit managing instruction in case where the pin state differs from its port latch contents.

✻2 Bit managing instructions: **SEB** and **CLB** instructions

(3) Pull-up control

When using port P2, which includes a pull-up resistor, as an output port, its port pull-up control is invalidated, that is, pull-up cannot be enabled.

● **Reason**

Pull-up/pull-down control is valid only when each direction register is set to the input mode.

3.3.13 Notes on programming

(1) Processor status register

① **Initializing of processor status register**

Flags which affect program execution must be initialized after a reset.
In particular, it is essential to initialize the T and D flags because they have an important effect on calculations.

● **Reason**

After a reset, the contents of the processor status register (PS) are undefined except for the I flag which is “1”.

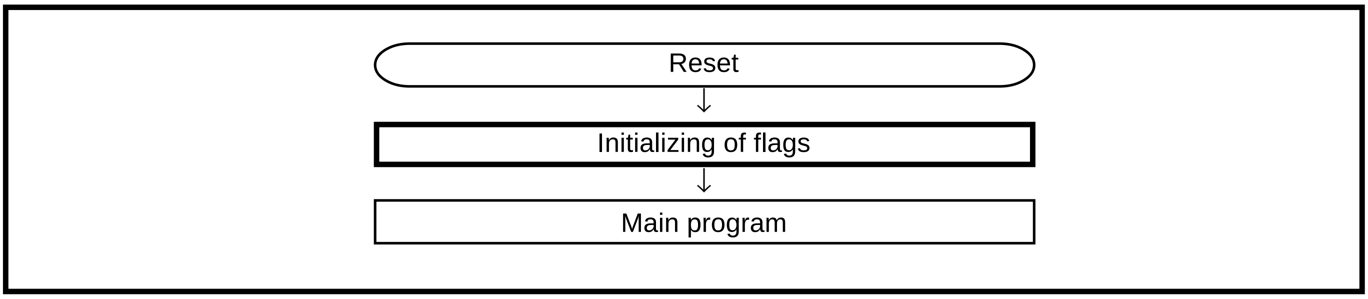


Fig. 3.3.5 Initialization of processor status register

② **How to reference the processor status register**

To reference the contents of the processor status register (PS), execute the **PHP** instruction once then read the contents of (S+1). If necessary, execute the **PLP** instruction to return the PS to its original status. A **NOP** instruction should be executed after every **PLP** instruction.

Be sure to execute the **SEI** instruction before the **PLP** instruction. If executing the **CLI** instruction, do it after the **NOP** instruction

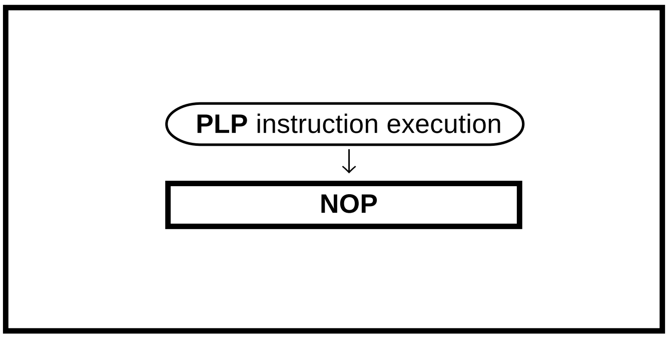


Fig. 3.3.6 Sequence of PLP instruction execution

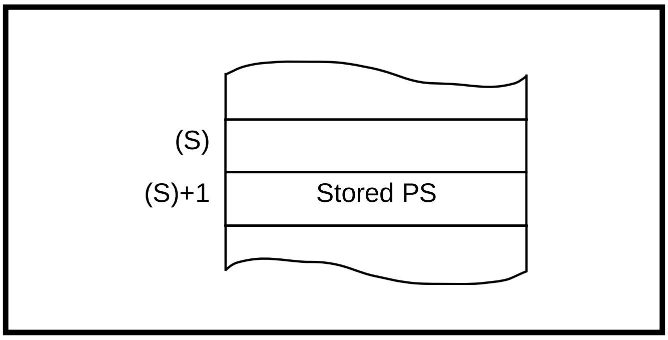


Fig. 3.3.7 Stack memory contents after PHP instruction execution

(2) BRK Instruction

It can be detected that the **BRK** instruction interrupt event or the least priority interrupt event by referring the stored B flag state. Refer to the stored B flag state in the interrupt routine.

(3) Decimal Calculations

When decimal mode is selected, the values of the V flags are invalid.

The carry flag (C) is set to “1” if a carry is generated as a result of the calculation, or is cleared to “0” if a borrow is generated. To determine whether a calculation has generated a carry, the C flag must be initialized to “0” before each calculation. To check for a borrow, the C flag must be initialized to “1” before each calculation.

(4) Multiplication and Division Instructions

The index X mode (T) and the decimal mode (D) flags do not affect the MUL and DIV instruction.

(5) Instruction Execution Time

The instruction execution time is obtained by multiplying the frequency of the internal clock ϕ by the number of cycles needed to execute an instruction.

The number of cycles required to execute an instruction is shown in the list of machine instructions.

3.3.14 Termination of unused pins

(1) Terminate unused pins

① I/O ports :

- Set the I/O ports for the input mode and connect them to Vcc or Vss through each resistor of 1 kΩ to 10 kΩ.
  Ports that permit the selecting of a built-in pull-up resistor can also use this resistor. Set the I/O ports for the output mode and open them at “L” or “H”.
- When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.
- Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

(2) Termination remarks

① I/O ports :

Do not open in the input mode.

● **Reason**

- The power source current may increase depending on the first-stage circuit.
- An effect due to noise may be easily produced as compared with proper termination ② and shown on the above.

② I/O ports :

When setting for the input mode, do not connect to Vcc or Vss directly.

● **Reason**

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between a port and Vcc (or Vss).

③ I/O ports :

When setting for the input mode, do not connect multiple ports in a lump to Vcc or Vss through a resistor.

● **Reason**

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between ports.

- At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

3.3.15 Notes on CPU rewrite mode for flash memory version

The below notes applies when rewriting the flash memory in CPU rewrite mode.

(1) Operation speed

During CPU rewrite mode, set the internal clock φ to 6 MHz or less using the X_{IN} divider select bit (bit 7 of address $001F_{16}$).

(2) Instructions inhibited against use

The instructions which refer to the internal data of the flash memory cannot be used during CPU rewrite mode .

(3) Interrupts inhibited against use

The interrupts cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory.

(4) Reset

Reset is always valid. When CNV_{SS} is “H” at reset release, the program starts from the address stored in addresses $FFFA_{16}$ and $FFFB_{16}$ of the boot ROM area in order that CPU may start in boot mode.

3.4 Countermeasures against noise

Countermeasures against noise are described below. The following countermeasures are effective against noise in theory, however, it is necessary not only to take measures as follows but to evaluate before actual use.

3.4.1 Shortest wiring length

The wiring on a printed circuit board can function as an antenna which feeds noise into the microcomputer. The shorter the total wiring length (by mm unit), the less the possibility of noise insertion into a microcomputer.

(1) Wiring for $\overline{\mathrm{RESET}}$ pin

Make the length of wiring which is connected to the $\overline{\mathrm{RESET}}$ pin as short as possible. Especially, connect a capacitor across the RESET pin and the Vss pin with the shortest possible wiring (within 20mm).

● **Reason**

The width of a pulse input into the $\overline{\mathrm{RESET}}$ pin is determined by the timing necessary conditions. If noise having a shorter pulse width than the standard is input to the $\overline{\mathrm{RESET}}$ pin, the reset is released before the internal state of the microcomputer is completely initialized. This may cause a program runaway.

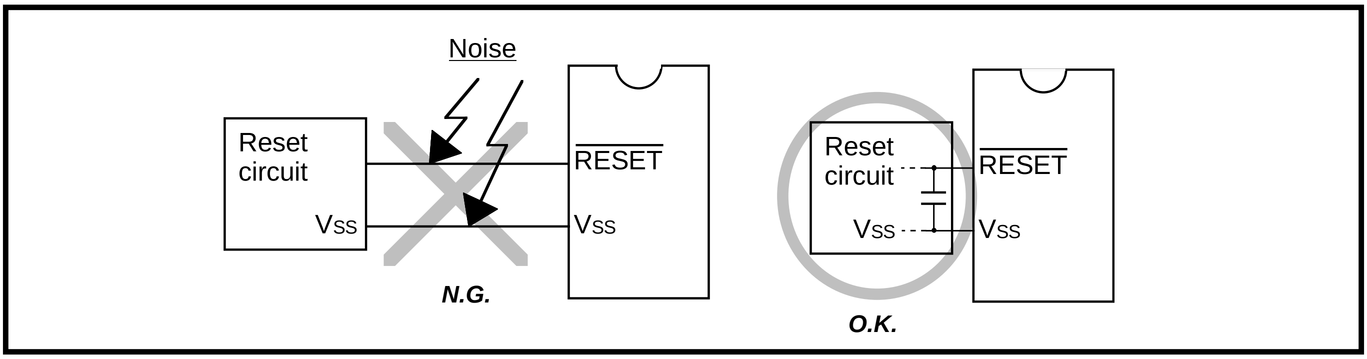


Fig. 3.4.1 Wiring for the $\overline{\mathrm{RESET}}$ pin

(2) Wiring for clock input/output pins

- Make the length of wiring which is connected to clock I/O pins as short as possible.
- Make the length of wiring (within 20 mm) across the grounding lead of a capacitor which is connected to an oscillator and the Vss pin of a microcomputer as short as possible.
- Separate the Vss pattern only for oscillation from other Vss patterns.

● **Reason**

If noise enters clock I/O pins, clock waveforms may be deformed. This may cause a program failure or program runaway. Also, if a potential difference is caused by the noise between the Vss level of a microcomputer and the Vss level of an oscillator, the correct clock will not be input in the microcomputer.

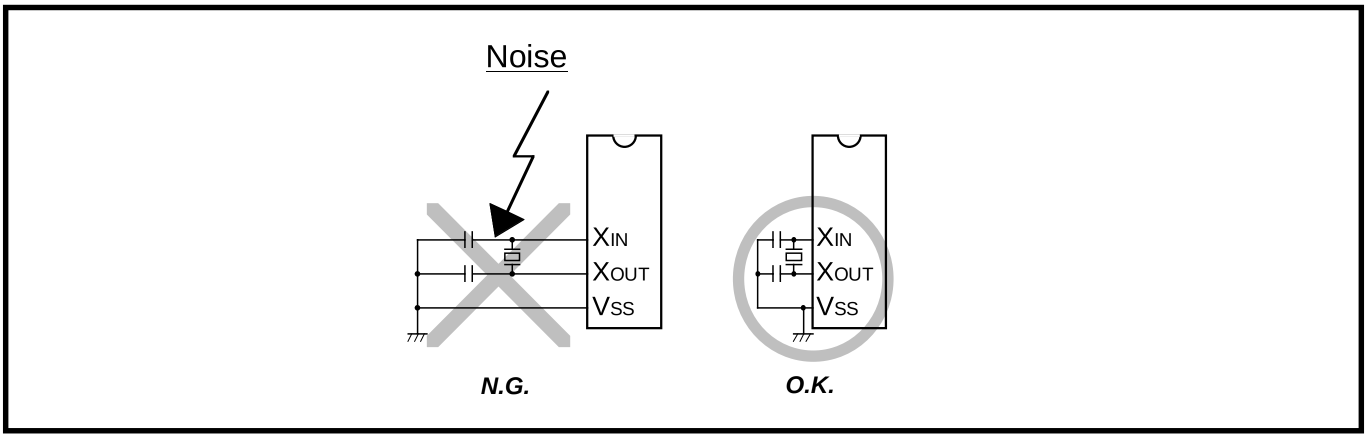


Fig. 3.4.2 Wiring for clock I/O pins

3.4.2 Connection of bypass capacitor across VSS line and VCC line

Connect an approximately 0.1 µF bypass capacitor across the VSS line and the VCC line as follows:

- Connect a bypass capacitor across the VSS pin and the VCC pin at equal length.
- Connect a bypass capacitor across the VSS pin and the VCC pin with the shortest possible wiring.
- Use lines with a larger diameter than other signal lines for VSS line and VCC line.
- Connect the power source wiring via a bypass capacitor to the VSS pin and the VCC pin.

In use of the 7641 group it is recommended to connect 0.1 µF and 4.7 µF capacitors in parallel between pins Vcc and Vss, and pins AVss and AVcc. However, their capacitors must not be allocated near the LPF pin.

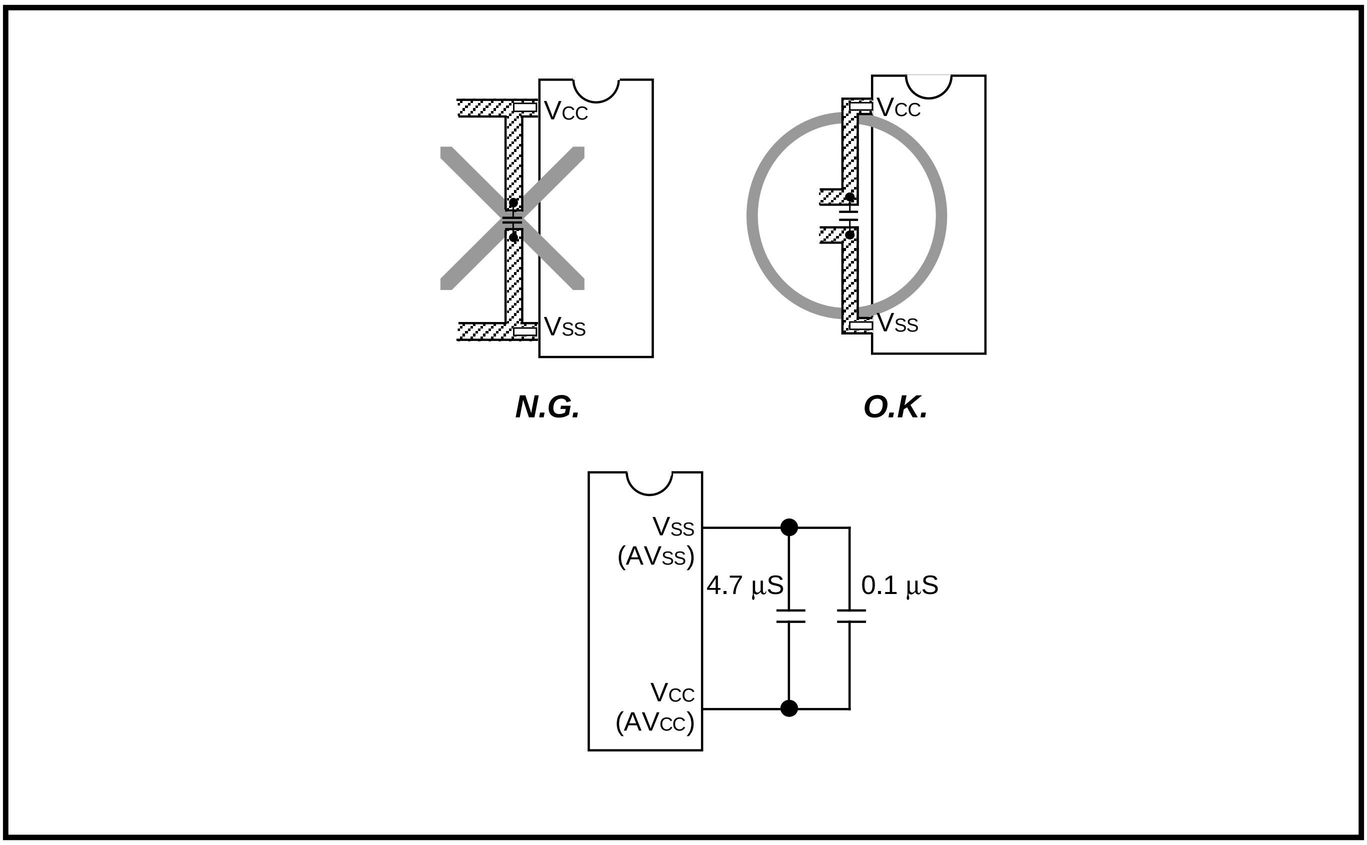


Fig. 3.4.3 Bypass capacitor across the VSS line and the VCC line

3.4.3 Oscillator concerns

Take care to prevent an oscillator that generates clocks for a microcomputer operation from being affected by other signals.

(1) Keeping oscillator away from large current signal lines

Install a microcomputer (and especially an oscillator) as far as possible from signal lines where a current larger than the tolerance of current value flows.

● **Reason**

In the system using a microcomputer, there are signal lines for controlling motors, LEDs, and thermal heads or others. When a large current flows through those signal lines, strong noise occurs because of mutual inductance.

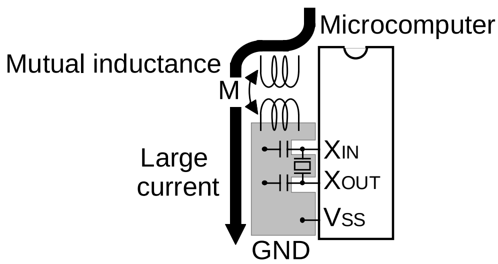


Fig. 3.4.4 Wiring for a large current signal line

(2) Installing oscillator away from signal lines where potential levels change frequently

Install an oscillator and a connecting pattern of an oscillator away from signal lines where potential levels change frequently. Also, do not cross such signal lines over the clock lines or the signal lines which are sensitive to noise.

● **Reason**

Signal lines where potential levels change frequently (such as the CNTR pin signal line) may affect other lines at signal rising edge or falling edge. If such lines cross over a clock line, clock waveforms may be deformed, which causes a microcomputer failure or a program runaway.

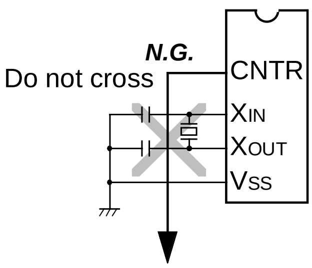


Fig. 3.4.5 Wiring for signal lines where potential levels change frequently

(3) Oscillator protection using Vss pattern

As for a two-sided printed circuit board, print a Vss pattern on the underside (soldering side) of the position (on the component side) where an oscillator is mounted.
Connect the Vss pattern to the microcomputer Vss pin with the shortest possible wiring. Besides, separate this Vss pattern from other Vss patterns.

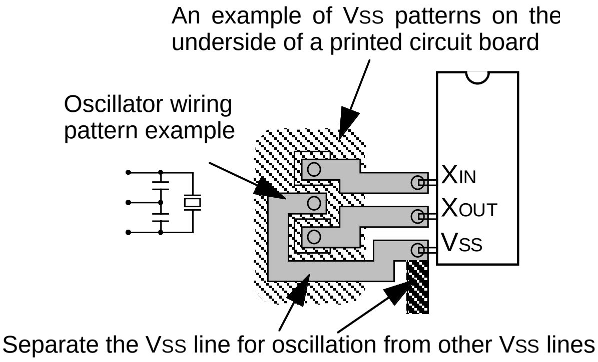


Fig. 3.4.6 Vss pattern on the underside of an oscillator

3.4.4 Setup for I/O ports

Setup I/O ports using hardware and software as follows:

<Hardware>

- Connect a resistor of 100 Ω or more to an I/O port in series.

<Software>

- As for an input port, read data several times by a program for checking whether input levels are equal or not.
- As for an output port, since the output data may reverse because of noise, rewrite data to its port latch at fixed periods.
- Rewrite data to direction registers at fixed periods.

Note: When a direction register is set for input port again at fixed periods, a several-nanosecond short pulse may be output from this port. If this is undesirable, connect a capacitor to this port to remove the noise pulse.

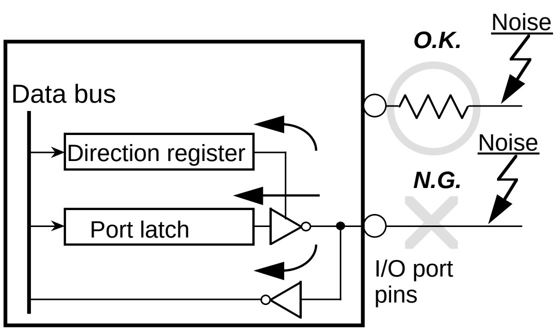


Fig. 3.4.7 Setup for I/O ports

3.4.5 Providing of watchdog timer function by software

If a microcomputer runs away because of noise or others, it can be detected by a software watchdog timer and the microcomputer can be reset to normal operation. This is equal to or more effective than program runaway detection by a hardware watchdog timer. The following shows an example of a watchdog timer provided by software.

In the following example, to reset a microcomputer to normal operation, the main routine detects errors of the interrupt processing routine and the interrupt processing routine detects errors of the main routine. This example assumes that interrupt processing is repeated multiple times in a single main routine processing.

<The main routine>

- Assigns a single byte of RAM to a software watchdog timer (SWDT) and writes the initial value N in the SWDT once at each execution of the main routine. The initial value N should satisfy the following condition:
  N+1 ≥ (Counts of interrupt processing executed in each main routine)
  As the main routine execution cycle may change because of an interrupt processing or others, the initial value N should have a margin.
- Watches the operation of the interrupt processing routine by comparing the SWDT contents with counts of interrupt processing after the initial value N has been set.
- Detects that the interrupt processing routine has failed and determines to branch to the program initialization routine for recovery processing in the following case:
  If the SWDT contents do not change after interrupt processing.

<The interrupt processing routine>

- Decrements the SWDT contents by 1 at each interrupt processing.
- Determines that the main routine operates normally when the SWDT contents are reset to the initial value N at almost fixed cycles (at the fixed interrupt processing count).
- Detects that the main routine has failed and determines to branch to the program initialization routine for recovery processing in the following case:
  If the SWDT contents are not initialized to the initial value N but continued to decrement and if they reach 0 or less.

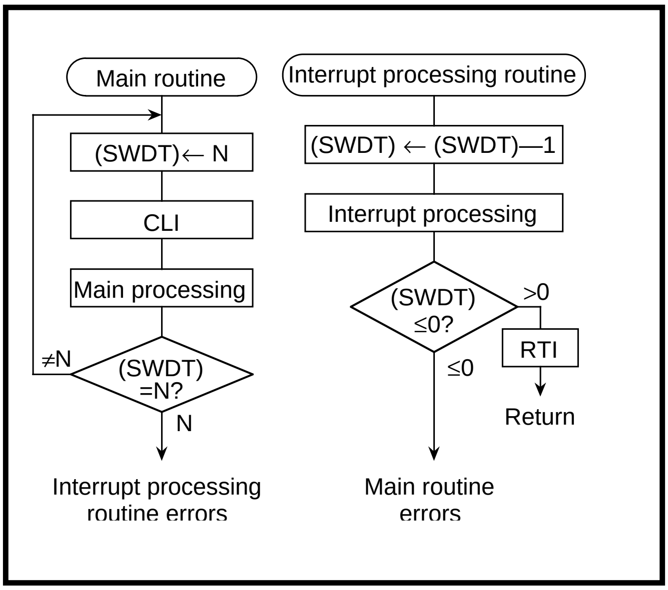


Fig. 3.4.8 Watchdog timer by software

3.5 Control registers

CPU mode register A

b7 b6 b5 b4 b3 b2 b1 b0: [ ][ ][ ][ ][1][ ][ ][ ]

CPU mode register A
(CPMA : address 00_{16})

b	Name	Functions	At reset	R	W
0	Processor mode bits	b1b0 0 0 : Single-chip mode 0 1 : Memory expansion mode 1 0 : Microprocessor mode (**Note 1**) 1 1 : Not available	0	O	O
1			0	O	O
2	Stack page select bit	0 : Page 0 1 : Page 1	1	O	O
3	Fix this bit to “1”.		1	O	O
4	Sub-clock (XCIN-XCOUT) stop bit	0 : Stopped 1 : Oscillating	0	O	O
5	Main clock (XIN-XOUT) stop bit	0 : Oscillating 1 : Stopped	0	O	O
6	Internal system clock select bit (**Note 2**)	0 : External clock (XIN-XOUT or XCIN-XCOUT) 1 : fsyn	0	O	O
7	External clock select bit	0 : XIN-XOUT 1 : XCIN-XCOUT	0	O	O

Notes 1: This is not available in the flash memory version.
2: When (CPMA 7, 6) = (0, 0), the internal system clock can be selected between f(XIN) and f(XIN)/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

Fig. 3.5.1 Structure of CPU mode register A

CPU mode register B

b7 b6 b5 b4 b3 b2 b1 b0: [1][0][ ][ ][ ][ ][ ][ ]

CPU mode register B
(CPMB : address 01_{16})

b	Name	Functions	At reset	R	W
0	Slow memory wait select bits	b1b0 0 0 : No wait 0 1 : One-time wait 1 0 : Two-time wait 1 1 : Three-time wait	1	O	O
1			1	O	O
2	Slow memory wait mode select bits	b3b2 0 0 : Software wait 0 1 : Not available 1 0 : RDY wait 1 1 : Software wait plus RDY input anytime wait anytime wait	0	O	O
3			0	O	O
4	Expanded data memory access bit	0 : EDMA output disabled 1 : EDMA output enabled	0	O	O
5	HOLD function enable bit	0 : HOLD function disabled 1 : HOLD function enabled	0	O	O
6	Fix this bit to “0”.		0	O	O
7	Fix this bit to “1”.		1	O	O

Fig. 3.5.2 Structure of CPU mode register B

Interrupt request register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register A
(IREQA : address 02_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
1	USB SOF interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
2	INT_0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
3	INT_1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
4	DMAC0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
5	DMAC1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
6	UART1 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
7	UART1 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻

✻: "0" can be set by software, but "1" cannot be set.

Fig. 3.5.3 Structure of Interrupt request register A

Interrupt request register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt request register B
(IREQB : address 03_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
1	UART2 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
2	UART2 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
3	UART2 summing error interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
4	Timer X interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
5	Timer Y interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
6	Timer 1 receive buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻
7	Timer 2 transmit interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	○	✻

✻: "0" can be set by software, but "1" cannot be set.

Fig. 3.5.4 Structure of Interrupt request register B

Interrupt request register C

b7 b6 b5 b4 b3 b2 b1 b0: 0 (b7 fixed)

Interrupt request register C
(IREQC : address 04_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
1	CNTR0 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
2	CNTR1 interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
3	Serial I/O interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
4	Input buffer full interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
5	Output buffer empty interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
6	Key input interrupt request bit	0 : No interrupt request issued 1 : Interrupt request issued	0	O	✻
7	Fix this bit to "0".		0	O	O

✻: "0" can be set by software, but "1" cannot be set.

Fig. 3.5.5 Structure of Interrupt request register C

Interrupt control register A

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register A
(ICONA : address 05_{16})

b	Name	Functions	At reset	R	W
0	USB function interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
1	USB SOF interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
2	INT0 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
3	INT1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
4	DMAC0 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	DMAC1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
6	UART1 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
7	UART1 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O

Fig. 3.5.6 Structure of Interrupt control register A

Interrupt control register B

b7 b6 b5 b4 b3 b2 b1 b0

Interrupt control register B
(ICONB : address 06_{16})

b	Name	Functions	At reset	R	W
0	UART1 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
1	UART2 receive buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
2	UART2 transmit interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
3	UART2 summing error interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
4	Timer X interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
5	Timer Y interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
6	Timer 1 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
7	Timer 2 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○

Fig. 3.5.7 Structure of Interrupt control register B

Interrupt control register C

b7 b6 b5 b4 b3 b2 b1 b0
0

Interrupt control register C
(ICONC : address 07_{16})

b	Name	Functions	At reset	R	W
0	Timer 3 interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
1	CNTR$_0$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
2	CNTR$_1$ interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
3	Serial I/O interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
4	Input buffer full interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
5	Output buffer empty interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
6	Key input interrupt enable bit	0 : Interrupt disabled 1 : Interrupt enabled	0	○	○
7	Fix this bit to “0”.		0	○	○

Fig. 3.5.8 Structure of Interrupt control register C

Port Pi

b7 b6 b5 b4 b3 b2 b1 b0

Port Pi (i = 0, 1, 2, 3, 5, 6, 8)
(Pi : addresses 08_{16}, $0A_{16}$, $0C_{16}$, $0E_{16}$, 16_{16}, 14_{16}, $1C_{16}$)

b	Name	Functions	At reset	R	W
0	Port Pi_0	●In output mode Write •••••••• Port latch Read •••••••• Port latch ●In input mode Write •••••••• Port latch Read •••••••• Value of pin	0	○	○
1	Port Pi_1		0	○	○
2	Port Pi_2		0	○	○
3	Port Pi_3		0	○	○
4	Port Pi_4		0	○	○
5	Port Pi_5		0	○	○
6	Port Pi_6		0	○	○
7	Port Pi_7		0	○	○

Fig. 3.5.9 Structure of Port Pi

Port P4, Port P7

b7 b6 b5 b4 b3 b2 b1 b0

Port P4, Port P7
(P4, P7 : addresses 18_{16}, $1A_{16}$)

b	Name	Functions	At reset	R	W
0	Port $P4_0$ or Port $P7_0$	●In output mode Write •••••••• Port latch Read •••••••• Port latch ●In input mode Write •••••••• Port latch Read •••••••• Value of pin	0	○	○
1	Port $P4_1$ or Port $P7_1$		0	○	○
2	Port $P4_2$ or Port $P7_2$		0	○	○
3	Port $P4_3$ or Port $P7_3$		0	○	○
4	Port $P4_4$ or Port $P7_4$		0	○	○
5	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are undefined.		Undefined	×	×
6			Undefined	×	×
7			Undefined	×	×

Fig. 3.5.10 Structure of Port P4, Port P7

Port Pi direction register

b7 b6 b5 b4 b3 b2 b1 b0

Port Pi direction register (i = 0, 1, 2, 3, 5, 6, 8)
(PiD : addresses 09_{16}, $0B_{16}$, $0D_{16}$, $0F_{16}$, 17_{16}, 15_{16}, $1D_{16}$)

b	Name	Functions	At reset	R	W
0	Port Pi direction register	0 : Port Pi_0 input mode 1 : Port Pi_0 output mode	0	×	O
1		0 : Port Pi_1 input mode 1 : Port Pi_1 output mode	0	×	O
2		0 : Port Pi_2 input mode 1 : Port Pi_2 output mode	0	×	O
3		0 : Port Pi_3 input mode 1 : Port Pi_3 output mode	0	×	O
4		0 : Port Pi_4 input mode 1 : Port Pi_4 output mode	0	×	O
5		0 : Port Pi_5 input mode 1 : Port Pi_5 output mode	0	×	O
6		0 : Port Pi_6 input mode 1 : Port Pi_6 output mode	0	×	O
7		0 : Port Pi_7 input mode 1 : Port Pi_7 output mode	0	×	O

Fig. 3.5.11 Structure of Port Pi direction register

Port P4, P7 direction registers

b7 b6 b5 b4 b3 b2 b1 b0

Port P4 direction register, Port P7 direction register
(P4D, P7D : addresses 19_{16}, $1B_{16}$)

b	Name	Functions	At reset	R	W
0	Port P4 direction register Port P7 direction register	0 : Port $P4_0$ or $P7_0$ input mode 1 : Port $P4_0$ or $P7_0$ output mode	0	×	O
1		0 : Port $P4_1$ or $P7_1$ input mode 1 : Port $P4_1$ or $P7_1$ output mode	0	×	O
2		0 : Port $P4_2$ or $P7_2$ input mode 1 : Port $P4_2$ or $P7_2$ output mode	0	×	O
3		0 : Port $P4_3$ or $P7_3$ input mode 1 : Port $P4_3$ or $P7_3$ output mode	0	×	O
4		0 : Port $P4_4$ or $P7_4$ input mode 1 : Port $P4_4$ or $P7_4$ output mode	0	×	O
5	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are undefined.		Undefined	×	×
6			Undefined	×	×
7			Undefined	×	×

Fig. 3.5.12 Structure of Port P4, Port P7 direction registers

Port control register

b7 b6 b5 b4 b3 b2 b1 b0

Port control register
(PTC : address 10_{16})

b	Name	Functions	At reset	R	W
0	Port P0 to P3 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
1	Port P4 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
2	Port P5 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
3	Port P6 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
4	Port P7 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
5	Port P8 slew rate control bit (**Note 1**)	0 : Disabled 1 : Enabled	0	○	○
6	Port P2 input level select bit	0 : Reduced VIHL level input (**Note 2**) 1 : CMOS level input	0	○	○
7	Master CPU bus input level select bit	0 : CMOS level input 1 : TTL level input	0	○	○

Notes 1: The slew rate function can reduce di/dt by modifying an internal buffer structure.
2: The characteristics of VIHL level is basically the same as that of TTL level. But, its switching center point is a little higher than TTL's.

Fig. 3.5.13 Structure of Port control register

Interrupt polarity select register

b7 b6 b5 b4 b3 b2 b1 b0
0 0 0 0 0 0

Interrupt polarity select register
(IPOL : address 11_{16})

b	Name	Functions	At reset	R	W
0	INT_0 interrupt edge select bit	0 : Falling edge active 1 : Rising edge active	0	○	○
1	INT_1 interrupt edge select bit	0 : Falling edge active 1 : Rising edge active	0	○	○
2	Fix these bits to "0".		0	○	○
3			0	○	○
4			0	○	○
5			0	○	○
6			0	○	○
7			0	○	○

Fig. 3.5.14 Structure of Interrupt polarity select register

Port P2 pull-up control register

b7 b6 b5 b4 b3 b2 b1 b0

Port P2 pull-upt control register
(PUP2 : address 12_{16})

b	Name	Functions	At reset	R	W
0	Port $P2_0$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
1	Port $P2_1$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
2	Port $P2_2$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
3	Port $P2_3$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
4	Port $P2_4$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
5	Port $P2_5$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
6	Port $P2_6$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O
7	Port $P2_7$ pull-up control bit	0 : Disabled 1 : Enabled	0	O	O

Fig. 3.5.15 Structure of Port P2 pull-up control register

USB control register

b7 b6 b5 b4 b3 b2 b1 b0
0 (b0)

USB control register
(USBC : address 13_{16})

b	Name	Functions	At reset	R	W
0	Fix this bit to “0”.		0	O	O
1	USB default state selection bit (USBC1)	0 : In default state after power-on/reset 1 : In default state after USB reset signal received	0	O	O
2	USB artificial SOF enable bit (USBC2)	0 : Artificial SOF disabled 1 : Artificial SOF enabled	0	O	O
3	USB line driver current control bit (USBC3)	0 : High current mode 1 : Low current mode	0	O	O
4	USB line driver supply enable bit (USBC4) (**Note 1**)	0 : Line driver disabled 1 : Line driver enabled	0	O	O
5	USB clock enable bit (USBC5)	0 : 48 MHz clock to the USB block disabled 1 : 48 MHz clock to the USB block enabled	0	O	O
6	USB $\overline{\text{SOF}}$ port select bit (USBC6)	0 : $\overline{\text{SOF}}$ output disabled 1 : $\overline{\text{SOF}}$ output enabled	0	O	O
7	USB enable bit (USBC7)	0 : USB block disabled (**Note 2**) 1 : USB block enabled	0	O	O

Notes 1: When using the MCU in Vcc = 3.3 V, set this bit to “0” and disable the built-in DC-DC converter.
2: Setting this bit to 0” causes the contents of all USB registers to have the values at reset.

Fig. 3.5.16 Structure of USB control register

Clock control register

b7 b6 b5 b4 b3 b2 b1 b0: – – – 0 0 0 0 0

Clock control register
(CCR : address $1F_{16}$)

b	Name	Functions	At reset	R	W
0	Fix these bits to "0".		0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4			0	O	O
5	XCOUT oscillation drive disable bit (CCR5)	0 : XCOUT oscillation drive is enabled. (When XCIN oscillation is enabled.) 1 : XCOUT oscillation drive is disabled.	0	O	O
6	XOUT oscillation drive disable bit (CCR6)	0 : XOUT oscillation drive is enabled. (When XIN oscillation is enabled.) 1 : XOUT oscillation drive is disabled.	0	O	O
7	XIN divider select bit (CCR7) (**Note**)	0 : f(XIN)/2 is used for the system clock. 1 : f(XIN) is used for the system clock.	0	O	O

Note: This bit is valid when (b7, b6 of CPMA) = "00".

Fig. 3.5.17 Structure of Clock control register

Timer X (low, high)

b7 b6 b5 b4 b3 b2 b1 b0

Timer XL, Timer XH
(TXL, TXH: addresses 20_{16}, 21_{16})

b	Functions	At reset	R	W
0	●Timer X's count value is set through this register. ●Writing operation depends on the timer X write control bit. When it is "0", the values are simultaneously written into timer X latch and counter. When it is "1", the values are written into only timer X latch. ●Timer X is a down-count timer. ●When reading this register's address, the timer X's count value is read out.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Notes 1: Read and write operation to timer X must be performed for both high and low-order bytes.
2: When reading timer X, read the high-order byte first and then the low-order byte.
3: When writing to timer X, write the low-order byte first and then the high-order byte.
4: Do not read this register during the write operation, or do not write during the read operation.

Fig. 3.5.18 Structure of Timer X

Timer Y (low, high)

b7 b6 b5 b4 b3 b2 b1 b0

Timer YL, Timer YH
(TYL, TYH: addresses 22_{16}, 23_{16})

b	Functions	At reset	R	W
0	●Timer Y's count value is set through this register. ●Timer Y is a down-count timer. ●When reading this register's address, the timer Y's count value is read out.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Notes 1: Read and write operation to timer X must be performed for both high and low-order bytes.
2: When reading timer X, read the high-order byte first and then the low-order byte.
3: When writing to timer X, write the low-order byte first and then the high-order byte.
4: Do not read this register during the write operation, or do not write during the read operation.

Fig. 3.5.19 Structure of Timer Y

Timer i (i = 1 to 3)

b7 b6 b5 b4 b3 b2 b1 b0

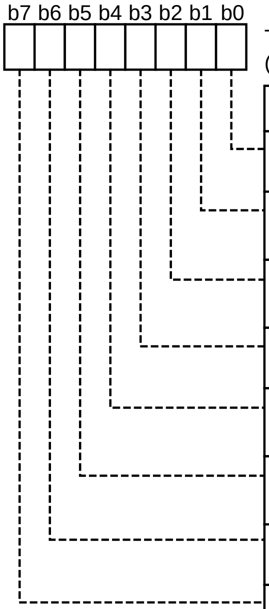
Timer 1, Timer 2, Timer 3
(T1, T2, T3: addresses 24_{16}, 25_{16}, 26_{16})

b	Functions	At reset	R	W
0	●Timer i's count value is set through this register. ●Timer 1 and Timer 2 Writing operation depends on the timers 1, 2 write control bit. When it is "0", the values are simultaneously written into their latches and counters. When it is "1", the values are written into only their latches. ●Timer 3 The values are simultaneously written into their latches and counters. ●When reading this register's address, its timer's count values are read out. ●The timer causes an underflow at the count pulse following the count where the timer contents reaches "00_{16}". Then The contents of latches are automatically reloaded into the timer.	(**Note**)	O	O
1		(**Note**)	O	O
2		(**Note**)	O	O
3		(**Note**)	O	O
4		(**Note**)	O	O
5		(**Note**)	O	O
6		(**Note**)	O	O
7		(**Note**)	O	O

Note: Timer 1 and Timer 3's values are "FF_{16}". Timer 2 's value are "01_{16}".

Fig. 3.5.20 Structure of Timer i

Timer X mode register



Timer X mode register
(TXM : address 27_{16})

b	Name	Functions	At reset	R	W
0	Timer X write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	O	O
1	Timer X count source select bits	b2b1 0 0 : ϕ / 8 0 1 : ϕ / 16 1 0 : ϕ / 32 1 1 : ϕ / 64	0	O	O
2			0	O	O
3	Timer X internal clock select bit	0 : ϕ / n (n = 8, 16, 32, 64) 1 : SCSGCLK (Special Count Source Generator)	0	O	O
4	Timer X operating mode bits	b5b4 0 0 : Timer mode 0 1 : Pulse output mode 1 0 : Event counter mode 1 1 : Pulse width measurement mode	0	O	O
5			0	O	O
6	$CNTR_0$ active edge switch bit	This function depends on the timer X operating mode. (See below.)	0	O	O
7	Timer X count stop bit	0 : Count start 1 : Count stop	0	O	O

Function of $CNTR_0$ active edge switch bit

Timer X operating mode	$CNTR_0$ active edge switch bit
Pulse output mode	0 : Starts at "H" output 1 : Starts at "L" output
Event counter mode	0 : Counts at rising edge 1 : Counts at falling edge
Pulse width measurement mode	0 : Measures "H" pulse width 1 : Measures "L" pulse width
Interrupt	0 : Falling edge active 1 : Rising edge active

Fig. 3.5.21 Structure of Timer X mode register

Timer Y mode register

b7 b6 b5 b4 b3 b2 b1 b0

Timer Y mode register
(TYM : address 28_{16})

b	Name	Functions	At reset	R	W
0	Timer Y write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	○	○
1	Timer Y output control bit	0 : TYOUT output disabled 1 : TYOUT output enabled	0	○	○
2	Timer Y count source select bits	b3b2 0 0 : φ / 8 0 1 : φ / 16 1 0 : φ / 32 1 1 : φ / 64	0	○	○
3			0	○	○
4	Timer Y operating mode bits	b5b4 0 0 : Timer mode 0 1 : Period measurement mode 1 0 : Event counter mode 1 1 : Pulse width HL continuously measurement mode	0	○	○
5			0	○	○
6	CNTR1 active edge switch bit	This function depends on the timer Y operating mode. (See below.)	0	○	○
7	Timer Y count stop bit	0 : Count start 1 : Count stop	0	○	○

Function of CNTR1 active edge switch bit

Timer Y operating mode	CNTR1 active edge switch bit
Period measurement mode	0 : Measures between falling edges 1 : Measures between rising edges
Event counter mode	0 : Counts at rising edge 1 : Counts at falling edge
TYOUT output	0 : Starts at "H" output 1 : Starts at "L" output
Interrupt	0 : Falling edge active 1 : Rising edge active

Fig. 3.5.22 Structure of Timer Y mode register

Timer 123 mode register

b7 b6 b5 b4 b3 b2 b1 b0

Timer 123 mode register
(T123M : address 29_{16})

b	Name	Functions	At reset	R	W
0	TOUT factor select bit	0 : Timer 1 output 1 : Timer 2 output	0	O	O
1	Timer 1 count stop bit	0 : Count start 1 : Count stop	0	O	O
2	Timer 1 count source select bit	0 : φ / 8 1 : f(XCIN) / 2	0	O	O
3	Timer 2 count source select bit	0 : Timer 1 output 1 : φ	0	O	O
4	Timer 3 count source select bit	0 : Timer 1 output 1 : φ / 8	0	O	O
5	TOUT output active edge switch bit	0 : Start at “H” output 1 : Start at “L” output	0	O	O
6	TOUT output control bit	0 : TOUT output disabled 1 : TOUT output enabled	0	O	O
7	Timers 1, 2 write control bit	0 : Write value in latch and counter 1 : Write value in latch only	0	O	O

Fig. 3.5.23 Structure of Timer 123 mode register

Serial I/O shift register

b7 b6 b5 b4 b3 b2 b1 b0

Serial I/O shift register
(SIOSHT: address $2A_{16}$)

b	Functions	At reset	R	W
0	●At transmitting Writing transmitted data to this register starts transmitting operation. ●At receiving Read received data through this register.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7		Undefined	O	O

Fig. 3.5.24 Structure of Serial I/O shift register

Serial I/O control register 1

b7 b6 b5 b4 b3 b2 b1 b0

Serial I/O control register 1
(SIOCON1 : address $2B_{16}$)

b	Name	Functions	At reset	R	W
0	Internal synchronous clock select bits (**Note**)	b2b1b0 0 0 0 : Internal clock divided by 2 0 0 1 : Internal clock divided by 4 0 1 0 : Internal clock divided by 8 0 1 1 : Internal clock divided by 16 1 0 0 : Internal clock divided by 32 1 0 1 : Internal clock divided by 64 1 1 0 : Internal clock divided by 128 1 1 1 : Internal clock divided by 256	0	O	O
1			0	O	O
2			0	O	O
3	Serial I/O port select bit	0 : I/O port 1 : STXD, SCLK signal output	0	O	O
4	$\overline{SRDY}$ output select bit	0 : I/O port 1 : $\overline{SRDY}$ signal output	0	O	O
5	Transfer direction select bit	0 : LSB first 1 : MSB first	0	O	O
6	Synchronous clock select bit	0 : External clock 1 : Internal clock	1	O	O
7	STXD output channel control bit	0 : CMOS output 1 : N-channel open drain output	0	O	O

Note: The source of serial I/O internal synchronous clock can be selected by bit 1 of serial I/O control register 2

Fig. 3.5.25 Structure of Serial I/O control register 1

Serial I/O control register 2

b7 b6 b5 b4 b3 b2 b1 b0
0 0 0

Serial I/O control register 2
(SIOCON2 : address $2C_{16}$)

b	Name	Functions	At reset	R	W
0	SPI mode select bit	0 : Normal serial I/O mode 1 : SPI compatible mode (**Note**)	0	O	O
1	Serial I/O internal clock select bit	0 : ϕ 1 : SCSGCLK	0	O	O
2	SRXD input enable bit	0 : SRXD input disabed 1 : SRXD input enabed	0	O	O
3	Clock polarity select bit (CPoL)	0 : SCLK starting at “L” 1 : SCLK starting at “H”	1	O	O
4	Clock phase select bit (CPha)	0 : Serial transfer starting at falling edge of $\overline{SRDY}$ 1 : Serial transfer starting after a half cycle of SCLK passed at falling edge of $\overline{SRDY}$	1	O	O
5	Fix these bits to “0”.		0	O	O
6			0	O	O
7			0	O	O

Note: To set the slave mode, also set bit 4 of serial I/O control register 1 to “1”.

Fig. 3.5.26 Structure of Serial I/O control register 2

Special count source generator 1

b7 b6 b5 b4 b3 b2 b1 b0

Special count source generator 1
(SCSG1: address $2D_{16}$)

b	Functions	At reset	R	W
0	●This is a 8-bit down-count timer. The division ratio : 1 / (n1 +1), n1: value set to SCSG1.	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 3.5.27 Structure of Special count source generator 1

Special count source generator 2

b7 b6 b5 b4 b3 b2 b1 b0

Special count source generator 2
(SCSG2: address $2E_{16}$)

b	Functions	At reset	R	W
0	●This is a 8-bit down-count timer. An output from SCSG1 is divided by the contents of SCSG2 to generate SCSGCLK. ●The count source, an output from SCSG1, is $\phi \cdot n1 / (n1 +1)$. ●SCSGCLK frequency: $\phi \cdot n1 / (n1 +1) \cdot 1 / (n2 +1)$ n1: value set to SCSG1 n2: value set to SCSG2	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 3.5.28 Structure of Special count source generator 2

Special count source mode register

b7 b6 b5 b4 b3 b2 b1 b0
0 0 0 0

Special count source mode register
(SCSGM : address $2F_{16}$)

b	Name	Functions	At reset	R	W
0	SCSG1 data write control bit	0 : Writing data into both timer latch and timer simultaneously 1 : Writing data into timer latch	0	○	○
1	SCSG1 count stop bit	0 : Count start 1 : Count stop	0	○	○
2	SCSG2 data write control bit	0 : Writing data into both timer latch and timer simultaneously 1 : Writing data into timer latch	0	○	○
3	SCSGCLK output control bit	0 : SCSGCLK output disabled (SCSG1 and SCSG2 counts stop) 1 : SCSGCLK output enabled	0	○	○
4	Fix these bits to “0”.		0	○	○
5			0	○	○
6			0	○	○
7			0	○	○

Fig. 3.5.29 Structure of Special count source mode register

UARTx mode register

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) mode register
(UxMOD : addresses 30_{16}, 38_{16})

b	Name	Functions	At reset	R	W
0	UART clock select bit (CLK)	0 : ϕ 1 : SCSGCLK output	0	○	○
1	UART clock prescaling select bits (PS)	b2b1 0 0 : UART clock divided by 1 0 1 : UART clock divided by 8 1 0 : UART clock divided by 32 1 1 : UART clock divided by 256	0	○	○
2			0	○	○
3	Stop bit length select bit (STB)	0 : 1 stop bit 1 : 2 stop bits	0	○	○
4	Parity select bit (PMD)	0 : Even parity 1 : Odd parity	0	○	○
5	Parity enable bit (PEN)	0 : Parity checking disabled 1 : Parity checking enabled	0	○	○
6	UART character length select bit (LE1, 0)	b7b6 0 0 : 7 bits 0 1 : 8 bits 1 0 : 9 bits 1 1 : Not available	0	○	○
7			0	○	○

Fig. 3.5.30 Structure of UARTx (x = 1, 2) mode register

UARTx baud rate generator

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) baud rate generator
(UxBRG: addresses 31_{16}, 39_{16})

b	Functions	At reset	R	W
0	●The UxBRG determines the baud rate for transfer. ●This is a 8-bit counter with its reload register. This generator divides the frequency of the count source by 1/(n + 1), where “n” is the value written to the UxBRG.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7		Undefined	O	O

Fig. 3.5.31 Structure of UARTx (x = 1, 2) baud rate generator

UARTx status register

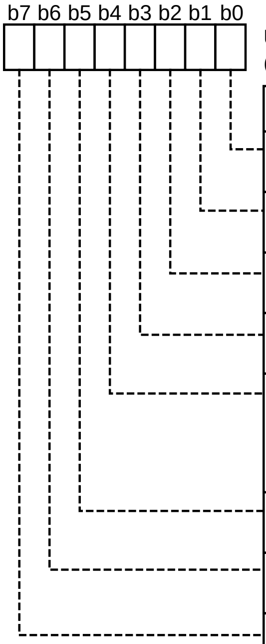
b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) status register
(UxSTS : addresses 32_{16}, $3A_{16}$)

b	Name	Functions	At reset	R	W
0	Transmit complete flag (TCM)	0 : Transmit shift in progress 1 : Transmit shift completed	1	O	×
1	Transmit buffer empty flag (TBE)	0 : Buffer full 1 : Buffer empty	1	O	×
2	Receive buffer full flag (RBF)	0 : Buffer empty 1 : Buffer full	0	O	×
3	Parity error flag (PER)	0 : No error 1 : Parity error	0	O	×
4	Framing error flag (FER)	0 : No error 1 : Framing error	0	O	×
5	Overrun error flag (OER)	0 : No error 1 : Overrun error	0	O	×
6	Summing error flag (SER)	0 : (PER) U (FER) U (OER) = 0 1 : (PER) U (FER) U (OER) = 1	0	O	×
7	Nothing is arranged for this bit. This is a write disable bit. When this bit is read out, the contents are “0”.		0	O	×

Fig. 3.5.32 Structure of UARTx (x = 1, 2) status register

UARTx control register



UARTx (x = 1, 2) control register
(UxCON : addresses 33_{16}, $3B_{16}$)

b	Name	Functions	At reset	R	W
0	Transmit enable bit (TEN)	0 : Transmit disabled 1 : Transmit enabled	0	O	O
1	Receive enable bit (REN)	0 : Receive disabled 1 : Receive enabled	0	O	O
2	Transmit initialization bit (TIN)	0 : No action 1 : Initializing (**Note 1**)	0	O	O
3	Receive initialization bit (RIN)	0 : No action 1 : Initializing (**Note 2**)	0	O	O
4	Transmit interrupt source select bit (TIS)	0 : Interrupt when transmit buffer has emptied 1 : Interrupt when transmit shift operation is completed	0	O	O
5	$\overline{CTS}$ function enable bit (CTS_SEL)	0 : $\overline{CTS}$ function disabled (**Note 3**) 1 : $\overline{CTS}$ function enabled	0	O	O
6	$\overline{RTS}$ function enable bit (RTS_SEL)	0 : $\overline{RTS}$ function disabled (**Note 4**) 1 : $\overline{RTS}$ function enabled	0	O	O
7	UART address mode enable bit (AME)	0 : Address mode disabled 1 : Address mode enabled	0	O	O

Notes 1: When setting the TIN bit to “1”, the TEN bit is set to “0” and the UARTx status register will be set to “03_{16}” after the data has been transmitted. To retransmit, set the TEN bit to “1” and set a data to the transmit buffer register again. The TIN bit will be cleared to “0” one cycle later after the TIN bit has been set to “1”.

2: Setting the RIN bit to “1” suspends the receiving operation and will set all of the REN, RBF and the receive error flags (PER, FER, OER, SER) to “0”. The RIN bit will be cleared to “0” one cycle later after the RIN bit has been set to “1”.

3: When $\overline{CTS}$ function is disabled (CTS_SEl = “0”), pins $P8_2$ and $P8_6$ can be used as ordinary I/O ports.

4: When $\overline{RTS}$ function is disabled (RTS_SEl = “0”), pins $P8_3$ and $P8_7$ can be used as ordinary I/O ports.

Fig. 3.5.33 Structure of UARTx (x = 1, 2) control register

UARTx transmit/receive buffer registers 1, 2

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) transmit/receive buffer register 1
(UxTRB1: addresses 34_{16}, $3C_{16}$)

b	Functions	At reset	R	W
0	The transmit buffer register and the receive buffer register are located at the same address. Writing a transmitting data and reading a received data are performed through the UxTRB. This is its low-order byte. •At write The data is written into the transmit buffer register. It is not done into the receive buffer register. •At read The contents of receive buffer register is read. If a character bit length is 7 bits, the MSB of received data is invalid.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7	Note that the contents of transmit buffer register cannot be read.	Undefined	O	O

b7 b6 b5 b4 b3 b2 b1 b0

UARTx (x = 1, 2) transmit/receive buffer register 2
(UxTRB2: addresses 35_{16}, $3D_{16}$)

b	Functions	At reset	R	W
0	The transmit buffer register and the receive buffer register are located at the same address. Writing a transmitting data and reading a received data are performed through the UxTRB. This is its high-order byte. •At write The data is written into the transmit buffer register. It is not done into the receive buffer register. •At read The contents of receive buffer register is read. If a character bit length is 9 bits, the received high-order 7 bits of UxTRB2 are “0” Note that the contents of transmit buffer register cannot be read. If a character bit length is 7 or 8 bits, the received contents of UxTRB2 are invalid.	Undefined	O	O
1	Nothing is arranged for this bit. This is a write disable bit. When this bit is read out, the contents are “0”.	Undefined	O	×
2		Undefined	O	×
3		Undefined	O	×
4		Undefined	O	×
5		Undefined	O	×
6		Undefined	O	×
7		Undefined	O	×

Fig. 3.5.34 Structure of UARTx (x = 1, 2) transmit/receive buffer registers 1, 2

UARTx RTS control register

b7 b6 b5 b4 b3 b2 b1 b0: _ _ _ _ 0 0 0 0

UARTx (x = 1, 2) RTS control register
(UxRTSC : addresses 36_{16}, $3E_{16}$)

b	Name	Functions	At reset	R	W
0	Fix these bits to “0”.		0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4	$\overline{RTS}$ assertion delay count select bits (RTS)	b7b6b5b4 0 0 0 0 : No delay; Assertion immediately 0 0 0 1 : 8-bit term assertion at “H” 0 0 1 0 : 16-bit term assertion at “H” 0 0 1 1 : 24-bit term assertion at “H” 0 1 0 0 : 32-bit term assertion at “H” 0 1 0 1 : 40-bit term assertion at “H” 0 1 1 0 : 48-bit term assertion at “H” 0 1 1 1 : 56-bit term assertion at “H” 1 0 0 0 : 64-bit term assertion at “H” 1 0 0 1 : 72-bit term assertion at “H” 1 0 1 0 : 80-bit term assertion at “H” 1 0 1 1 : 88-bit term assertion at “H” 1 1 0 0 : 96-bit term assertion at “H” 1 1 0 1 : 104-bit term assertion at “H” 1 1 1 0 : 112-bit term assertion at “H” 1 1 1 1 : 120-bit term assertion at “H”	0	O	O
5			0	O	O
6			0	O	O
7			1	O	O

Fig. 3.5.35 Structure of UARTx (x = 1, 2) RTS control register

DMAC index and status register

b7 b6 b5 b4 b3 b2 b1 b0
(b6 = 0)

DMAC index and status register
(DMAIS : address $3F_{16}$)

b	Name	Functions	At reset	R	W
0	DMAC channel 0 count register underflow flag (D0UF)	0 : No underflow 1 : Underflow generated	0	○	✻
1	DMAC channel 0 suspend flag (D0SFI) (**Note 1**)	0 : Not suspended 1 : Suspended	0	○	×
2	DMAC channel 1 count register underflow flag (D1UF)	0 : No underflow 1 : Underflow generated	0	○	✻
3	DMAC channel 1 suspend flag (D1SFI) (**Note 1**)	0 : Not suspended 1 : Suspended	0	○	×
4	DMAC transfer suspend control bit (DTSC) (**Note 2**)	0 :Suspending only burst transfers during interrupt process 1 : Suspending both burst and cycle steal transfers during interrupt process	0	○	○
5	DMAC register reload disable bit (DRLDD) (**Note 3**)	0 :Enabling reload of source and destination registers of both channels 1 : Disabling reload of source and destination registers of both channels	0	○	○
6	Fix this bit to “0”.		0	○	○
7	Channel index bit (DCI)	0 : Channel 0 accessible 1 : Channel 1 accessible Accessed registers: Mode register, source register, destination register, transfer count register.	0	○	○

✻: “0” can be set by software, but “1” cannot be set.

Notes 1: Suspended by an interrupt.
2: Transfer suspended during interrupt process
3: This settings affect the source and destination registers of both channels.

Fig. 3.5.36 Structure of DMAC index and status register

DMAC channel x mode register 1 (x = 0, 1)

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x mode register 1
(DMAxM1 : address 40_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel x source register increment/decrement selection bit (DxSRID)	0 : Increment after transfer 1 : Decrement after transfer	0	O	O
1	DMAC channel x source register increment/decrement enable bit (DxSRCE)	0 : Disabled (No change after transfer) 1 : Enabled	0	O	O
2	DMAC channel x destination register increment/decrement selection bit (DxDRID)	0 : Increment after transfer 1 : Decrement after transfer	0	O	O
3	DMAC channel x destination register increment/decrement enable bit (DxDRCE)	0 : Disabled (No change after transfer) 1 : Enabled	0	O	O
4	DMAC channel x data write control bit (DxDWC)	0 : Writing data in reload latches and registers 1 : Writing data in reload latches only	0	O	O
5	DMAC channel x disable after count register underflow enable bit (DxDAUE)	0 : Channel x enabled after count register underflow 1 : Channel x disabled after count register underflow	0	O	O
6	DMAC channel x register reload bit (DxRLD)	0 : Not reloaded 1 : Source, destination, and transfer count registers contents of channel x to be reloaded	0	O (**Note 2**)	O
7	DMAC channel x transfer mode selection bit (DxTMS)	0 : Cycle steal transfer mode 1 : Burst transfer mode	0	O	O

Notes 1: Channels 1 and 2 share this register. The channel selection which can use this register depends on channel index bit, bit 7 of DMAC index and status register.
2: These bits' contents are "0" at read.

Fig. 3.5.37 Structure of DMAC channel x (x = 0, 1) mode register 1

DMAC channel 0 mode register 2

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel 0 mode register 2
(DMA0M2 : address 41_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel 0 hardware transfer request source bits (D0HR)	b3b2b1b0 0 0 0 0 : Not used 0 0 0 1 : UART1 receive interrupt 0 0 1 0 : UART1 transmit interrupt 0 0 1 1 : Timer Y interrupt 0 1 0 0 : INT_0 interrupt 0 1 0 1 : USB endpoint 1 IN_PKT_RDY signal (falling edge active) 0 1 1 0 : USB endpoint 2 IN_PKT_RDY signal (falling edge active) 0 1 1 1 : USB endpoint 3 IN_PKT_RDY signal (falling edge active) 1 0 0 0 : USB endpoint 1 OUT_PKT_RDY signal (rising edge active) 1 0 0 1 : USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active) 1 0 1 0 : USB endpoint 2 OUT_PKT_RDY signal (rising edge active) 1 0 1 1 : USB endpoint 3 OUT_PKT_RDY signal (rising edge active) 1 1 0 0 : Master CPU bus interface OBE_0 signal (rising edge active) 1 1 0 1 : Master CPU bus interface IBF_0 signal, data (rising edge active) 1 1 1 0 : Serial I/O transmit/receive interrupt 1 1 1 1 : $CNTR_1$ interrupt	0	○	○
1			0	○	○
2			0	○	○
3			0	○	○
4	DMAC channel 0 software transfer trigger (D0SWT)	0 : No action 1 : Request of channel 0 transfer by writing “1”	0	○ (**Note 2**)	○
5	DMAC channel 0 USB and master CPU bus interface enable bit (D0UMIE)	0 : Disabled 1 : Enabled	0	○	○
6	DMAC channel 0 transfer initiation source capture register reset bit (D0CRR)	0 : No action 1 : Reset of channel 0 capture register by writing “1”	0	○ (**Note 2**)	○
7	DMAC channel 0 enable bit (D0CEN)	0 : Channel 0 disabled 1 : Channel 0 enabled (**Note 3**)	0	○	○

Notes 1: DMAC channel 0 mode register 2 and DMAC channel 1 mode register 2 are assigned at the same address 41_{16}. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: These bits’ contents are “0” at read. This bit is automatically cleared to “0” after writing “1”.
3: When setting this bit to “1”, simultaneously set the DMAC channel 0 transfer initiation source capture register reset bit (bit 6 of DMA0M2) to “1”.

Fig. 3.5.38 Structure of DMAC channel 0 mode register 2

DMAC channel 1 mode register 2

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel 1 mode register 2
(DMA1M2 : address 41_{16}) (**Note 1**)

b	Name	Functions	At reset	R	W
0	DMAC channel 1 hardware transfer request source bits (D1HR)	b3b2b1b0 0 0 0 0 : Not used 0 0 0 1 : UART2 receive interrupt 0 0 1 0 : UART2 transmit interrupt 0 0 1 1 : Timer X interrupt 0 1 0 0 : INT_1 interrupt 0 1 0 1 : USB endpoint 1 IN_PKT_RDY signal (falling edge active) 0 1 1 0 : USB endpoint 2 IN_PKT_RDY signal (falling edge active) 0 1 1 1 : USB endpoint 4 IN_PKT_RDY signal (falling edge active) 1 0 0 0 : USB endpoint 1 OUT_PKT_RDY signal (rising edge active) 1 0 0 1 : USB endpoint 1 OUT_FIFO_NOT_EMPTY signal (rising edge active) 1 0 1 0 : USB endpoint 2 OUT_PKT_RDY signal (rising edge active) 1 0 1 1 : USB endpoint 4 OUT_PKT_RDY signal (rising edge active) 1 1 0 0 : Master CPU bus interface OBE_1 signal (rising edge active) 1 1 0 1 : Master CPU bus interface IBF_1 signal, data (rising edge active) 1 1 1 0 : Timer 1 interrupt 1 1 1 1 : $CNTR_0$ interrupt	0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4	DMAC channel 1 software transfer trigger (D1SWT)	0 : No action 1 : Request of channel 1 transfer by writing “1”	0	O (**Note 2**)	O
5	DMAC channel 1 USB and master CPU bus interface enable bit (D1UMIE)	0 : Disabled 1 : Enabled	0	O	O
6	DMAC channel 1 transfer initiation source capture register reset bit (D1CRR)	0 : No action 1 : Reset of channel 1 capture register by writing “1”	0	O (**Note 2**)	O
7	DMAC channel 1 enable bit (D1CEN)	0 : Channel 1 disabled 1 : Channel 1 enabled (**Note 3**)	0	O	O

Notes 1: DMAC channel 0 mode register 2 and DMAC channel 1 mode register 2 are assigned at the same address 41_{16}. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: These bits' contents are “0” at read. This bit is automatically cleared to “0” after writing “1”.
3: When setting this bit to “1”, simultaneously set the DMAC channel 1 transfer initiation source capture register reset bit (bit 6 of DMA1M2) to “1”.

Fig. 3.5.39 Structure of DMAC channel 1 mode register 2

DMAC channel x source registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x source registers Low, High
(DMAxSL,DMAxSH : addresses 42_{16}, 43_{16})

b	Functions	At reset	R	W
0	●This is a 16-bit register with a latch. ●This register indicates the source address for data transfer.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

Notes 1: DMAC channel x source registers low, high of channels 0 and 1 are assigned at the same addresses. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: Write data into the lower bytes first, and then the higher bytes.
3: Read the contents from the higher bytes first, and then the lower bytes.

Fig. 3.5.40 Structure of DMAC channel x (x = 0, 1) source registers Low, High

DMAC channel x destination registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x destination registers Low, High
(DMAxDL,DMAxDH : addresses 44_{16}, 45_{16})

b	Functions	At reset	R	W
0	●This is a 16-bit register with a latch. ●This register indicates the destination address for data transfer.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

Notes 1: DMAC channel x destination registers low, high of channels 0 and 1 are assigned at the same addresses. The accessible register depends on channel index bit, bit 7 of DMAC index and status register.
2: Write data into the lower bytes first, and then the higher bytes.
3: Read the contents from the higher bytes first, and then the lower bytes.

Fig. 3.5.41 Structure of DMAC channel x (x = 0, 1) destination registers Low, High

DMAC channel x transfer count registers Low, High

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x transfer count registers Low
(DMAxCL : address 46_{16})

b	Functions	At reset	R	W
0	●This is the lower 8-bit register with a latch. Set the lower 8 bits of transfer numbers. ●This register indicates the remaining transfer numbers while transfer is continuing. ●This contents are decreased by 1 at every transfer operation. ●When this register underflows, the DMAC interrupt request bit and the count register underflow flag (**Note 2**) are set to “1”.	0	○	○
1		0	○	○
2		0	○	○
3		0	○	○
4		0	○	○
5		0	○	○
6		0	○	○
7		0	○	○

b7 b6 b5 b4 b3 b2 b1 b0

DMAC channel x transfer count registers High
(DMAxCH : address 47_{16})

b	Functions	At reset	R	W
0	●This is the higher 8-bit register with a latch. Set the higher 8 bits of transfer numbers. ●This register indicates the remaining transfer numbers while transfer is continuing. ●This contents are decreased by 1 at every transfer operation.	0	○	○
1		0	○	○
2		0	○	○
3		0	○	○
4		0	○	○
5		0	○	○
6		0	○	○
7		0	○	○

Notes 1: DMAC channel x transfer count registers low, high of channels 0 and 1 are assigned at the same addresses 46_{16} and 47_{16}. The accessible channel depends on channel index bit, bit 7 of DMAC index and status register.

2: Channel 0 used: Bit 0 of DMAC index and status register
Channel 1 used: Bit 2 of DMAC index and status register

3: Write data into the lower byte first, and then the higher byte.

4: Read the contents from the higher byte first, and then the lower byte.

Fig. 3.5.42 Structure of DMAC channel x (x = 0, 1) transfer count registers Low, High

Data bus buffer register x

b7 b6 b5 b4 b3 b2 b1 b0

Data bus buffer register x (x = 0, 1)
(DBBx: addresses 48_{16}, $4C_{16}$)

b	Functions	At reset	R	W
0	●A data is latched into this register by a write request from the host CPU. ●Write an output data into this register. The data is output on to the data bus by a read request from the host CPU.	0	O	O
1		0	O	O
2		0	O	O
3		0	O	O
4		0	O	O
5		0	O	O
6		0	O	O
7		0	O	O

Fig. 3.5.43 Structure of Data bus buffer register x (x = 0, 1)

Data bus buffer status register x

b7 b6 b5 b4 b3 b2 b1 b0

Data bus buffer status register x (x = 0, 1)
(DBBSx: addresses 49_{16}, $4D_{16}$)

b	Name	Functions	At reset	R	W
0	Output buffer full flag (OBFx)	0 : Buffer empty 1 : Buffer full	0	O	×
1	Input buffer full flag (IBFx)	0 : Buffer empty 1 : Buffer full	0	O	×
2	User definable flag (U_2)	This flag can be defined by user freely.	0	O	O
3	A_0 flag (A_{0x})	This flag indicates the condition of A_0 status when the IBFx flag is set.	0	O	×
4	User definable flag (U_4 to U_7)	This flag can be defined by user freely.	0	O	O
5				O	O
6				O	O
7				O	O

Fig. 3.5.44 Structure of Data bus buffer status register x (x = 0, 1)

Data bus buffer control register 0

b7 b6 b5 b4 b3 b2 b1 b0 — b5 fixed to 0

Data bus buffer control register 0
(DBBC0 : address $4A_{16}$)

b	Name	Functions	At reset	R	W
0	OBF_0 output enable bit	0 : $P5_2$ functions as I/O port. 1 : $P5_2$ functions as OBF_0 output pin.	0	○	○
1	$\overline{IBF_0}$ output enable bit	0 : $P5_3$ functions as I/O port. 1 : $P5_3$ functions as $\overline{IBF_0}$	0	○	○
2	$\overline{IBF_0}$ interrupt select bit	0 : Occurrence due to data write (A_0 = “0”) or command write (A_0 = “1”) 1 : Occurrence due to command write (A_0 = “1”)	0	○	○
3	Output buffer 0 empty interrupt disable bit	0 : Enabled 1 : Disabled	0	○	○
4	Input buffer 0 full interrupt disable bit	0 : Enabled 1 : Disabled	0	○	○
5	Nothing arranged for this bit. Fix this bit to “0”.		0	○	○
6	Master CPU bus interface enable bit	Function of $P5_4$ to $P5_7$, $P6_0$ to $P6_7$; 0 : As I/O ports. 1 : As master CPU bus interface function pins.	0	○	○
7	Bus interface type select bit	0 : RD,WR separate type bus 1 : R/W type bus	0	○	○

Fig. 3.5.45 Structure of Data bus buffer control register 0

Data bus buffer control register 1

b7 b6 b5 b4 b3 b2 b1 b0 — b6, b5 fixed to 0

Data bus buffer control register 1
(DBBC0 : address $4E_{16}$)

b	Name	Functions	At reset	R	W
0	OBF_1 output enable bit	0 : $P7_4$ functions as I/O port. 1 : $P7_4$ functions as OBF_1 output pin. (**Note**)	0	○	○
1	$\overline{IBF_1}$ output enable bit	0 : $P7_3$ functions as I/O port. 1 : $P7_3$ functions as $\overline{IBF_1}$ output pin. (**Note**)	0	○	○
2	$\overline{IBF_1}$ interrupt select bit	0 : Occurrence due to data write (A_0 = “0”) or command write (A_0 = “1”) 1 : Occurrence due to command write (A_0 = “1”)	0	○	○
3	Output buffer 1 empty interrupt disable bit	0 : Enabled 1 : Disabled	0	○	○
4	Input buffer 1 full interrupt disable bit	0 : Enabled 1 : Disabled	0	○	○
5	Nothing arranged for these bits.		0	○	○
6	Fix these bits to “0”.		0	○	○
7	Data bus buffer function select bit	0 : Single data bus buffer mode($P7_2$ functions as I/O port.) 1 : Double data bus buffer mode($P7_2$ functions as $\overline{S_1}$ input pin.)	0	○	○

Note: This can be selected when the data bus buffer function select bit is “1”.

Fig. 3.5.46 Structure of Data bus buffer control register 1

USB address register

b7 b6 b5 b4 b3 b2 b1 b0
0 | | | | | | |

USB address register
(USBA : address 50_{16})

b	Name	Functions	At reset	R	W
0	This register maintains the 7-bit USB function control unit address assigned by the host CPU.		0	○	○
1			0	○	○
2			0	○	○
3			0	○	○
4			0	○	○
5			0	○	○
6			0	○	○
7	Nothing arranged for this bit. Fix this bit to “0”.		0	○	○

Fig. 3.5.47 Structure of USB address register

USB power management register

b7 b6 b5 b4 b3 b2 b1 b0
0 | 0 | 0 | 0 | 0 | | |

USB power management register
(USBPM : address 51_{16})

b	Name	Functions	At reset	R	W
0	USB suspend detection flag (SUSPEND)	0 : No USB suspend detected 1 : USB suspend detected (**Note 1**)	0	○	×
1	USB resume detection flag (RESUME)	0 : No USB resume signal detected 1 : USB resume signal detected	0	○	×
2	USB remote wake-up bit (WAKEUP)	0 : End of remote resume signal 1 : Transmitting of remote resume signal (only when SUSPEND = “1”) (**Note 2**)	0	○	○
3	Nothing arranged for this bit. Fix these bits to “0”.		0	○	○
4			0	○	○
5			0	○	○
6			0	○	○
7			0	○	○

Notes 1: This bit is cleared when the WAKEUP bit is “1”.
2: When the SUSPEND bit is “1”, set this bit to “1” and keep “1” for 10 ms to 15 ms.

Fig. 3.5.48 Structure of USB power management register

USB interrupt status register 1

b7 b6 b5 b4 b3 b2 b1 b0 (b1 = 0)

USB interrupt status register 1
(USBIS1 : address 52_{16})

b	Name	Functions	At reset	R	W
0	USB endpoint 0 interrupt status flag (INTST0)	0 : Except the following conditions 1 : Set at any one of the following conditions: • A packet data of endpoint 0 is successfully received • A packet data of endpoint 0 is successfully sent • DATA_END bit of endpoint 0 is cleared to “0” • FORCE_STALL bit of endpoint 0 is set to “1” • SETUP_END bit of endpoint 0 is set to “1”.	0	O	✻
1	Nothing arranged for this bit. Fix this bit to “0”.		0	O	O
2	USB endpoint 1 IN interrupt status flag (INTST2)	0 : Except the following conditions 1 : Set at which of the following conditions: • A packet data of endpoint 1 is successfully sent • UNDER_RUN bit of endpoint 1 is set to “1”.	0	O	✻
3	USB endpoint 1 OUT interrupt status flag (INTST3)	0 : Except the following conditions 1 : Set at any one of the following conditions: • A packet data of endpoint 1 is successfully received • OVER_RUN bit of endpoint 1 is set to “1” • FORCE_STALL bit of endpoint 1 is set to “1”.	0	O	✻
4	USB endpoint 2 IN interrupt status flag (INTST4)	0 : Except the following conditions 1 : Set at which of the following conditions: • A packet data of endpoint 2 is successfully sent • UNDER_RUN bit of endpoint 2 is set to “1”.	0	O	✻
5	USB endpoint 2 OUT interrupt status flag (INTST5)	0 : Except the following conditions 1 : Set at any one of the following conditions: • A packet data of endpoint 2 is successfully received • OVER_RUN bit of endpoint 2 is set to “1” • FORCE_STALL bit of endpoint 2 is set to “1”.	0	O	✻
6	USB endpoint 3 IN interrupt status flag (INTST6)	0 : Except the following conditions 1 : Set at which of the following conditions: • A packet data of endpoint 3 is successfully sent • UNDER_RUN bit of endpoint 3 is set to “1”.	0	O	✻
7	USB endpoint 3 OUT interrupt status flag (INTST7)	0 : Except the following conditions 1 : Set at any one of the following conditions: • A packet data of endpoint 3 is successfully received • OVER_RUN bit of endpoint 3 is set to “1” • FORCE_STALL bit of endpoint 3 is set to “1”.	0	O	✻

✻: “0” can be set by software, but “1” cannot be set.
To clear the bit set to “1”, write “1” to the bit.

Fig. 3.5.49 Structure of USB interrupt status register 1

USB interrupt status register 2

b7	b6	b5	b4	b3	b2	b1	b0
				0	0		

USB interrupt status register 2
(USBIS2 : address 53_{16})

b	Name	Functions	At reset	R	W
0	USB endpoint 4 IN interrupt status flag (INTST8)	0 : Except the following conditions 1 : Set at which of the following conditions: • A packet data of endpoint 4 is successfully sent • UNDER_RUN bit of endpoint 4 is set to “1”.	0	O	✻
1	USB endpoint 4 OUT interrupt status flag (INTST9)	0 : Except the following conditions 1 : Set at any one of the following conditions: • A packet data of endpoint 4 is successfully received • OVER_RUN bit of endpoint 4 is set to “1” • FORCE_STALL bit of endpoint 4 is set to “1”.	0	O	✻
2	Nothing arranged for these bits. Fix these bist to “0”.		0	O	O
3			0	O	O
4	USB overrun/underrun interrupt status flag (INTST12)	0 : Except the following conditions 1 : Set at an occurrence of overrun / underrun (for isochronous data transfer)	0	O	✻
5	USB reset interrupt status flag (INTST13)	0 : Except the following conditions 1 : Set at receiving of USB reset signal	0	O	✻
6	USB resume signal interrupt status flag (INTST14)	0 : Except the following conditions 1 : Set at receiving of resume signal	0	O	✻
7	USB suspend signal interrupt status flag (INTST15)	0 : Except the following conditions 1 : Set at receiving of suspend signal	0	O	✻

✻: “0” can be set by software, but “1” cannot be set.
To clear the bit set to “1”, write “1” to the bit.

Fig. 3.5.50 Structure of USB interrupt status register 2

RENESAS

USB interrupt enable register 1

b7	b6	b5	b4	b3	b2	b1	b0
						0	

USB interrupt enable register 1
(USBIE1 : address 54_{16})

b	Name	Functions	At reset	R	W
0	USB endpoint 0 interrupt enable bit (INTEN0)	0 : Disabled 1 : Enabled	1	O	O
1	Nothing arranged for this bit. Fix this bit to “0”.		1	O	O
2	USB endpoint 1 IN interrupt enable bit (INTEN2)	0 : Disabled 1 : Enabled	1	O	O
3	USB endpoint 1 OUT interrupt enable bit (INTEN3)	0 : Disabled 1 : Enabled	1	O	O
4	USB endpoint 2 IN interrupt enable bit (INTEN4)	0 : Disabled 1 : Enabled	1	O	O
5	USB endpoint 2 OUT interrupt enable bit (INTEN5)	0 : Disabled 1 : Enabled	1	O	O
6	USB endpoint 3 IN interrupt enable bit (INTEN6)	0 : Disabled 1 : Enabled	1	O	O
7	USB endpoint 3 OUT interrupt enable bit (INTEN7)	0 : Disabled 1 : Enabled	1	O	O

Fig. 3.5.51 Structure of USB interrupt enable register 1

USB interrupt enable register 2

b7	b6	b5	b4	b3	b2	b1	b0
	0	1		0	0		

USB interrupt enable register 2
(USBIE2 : address 55_{16})

b	Name	Functions	At reset	R	W
0	USB endpoint 4 IN interrupt enable bit (INTEN8)	0 : Disabled 1 : Enabled	1	O	O
1	USB endpoint 4 OUT interrupt enable bit (INTEN9)	0 : Disabled 1 : Enabled	1	O	O
2	Nothing arranged for these bits. Fix these bits to “0”.		0	O	O
3			0	O	O
4	USB overrun/underrun interrupt enable bit (INTEN12)	0 : Disabled 1 : Enabled	1	O	O
5	Nothing arranged for this bit. Fix this bit to “1”.		1	O	O
6	Nothing arranged for this bit. Fix this bit to “0”.		0	O	O
7	USB suspend/resume interrupt enable bit (INTEN15)	0 : Disabled 1 : Enabled	0	O	O

Fig. 3.5.52 Structure of USB interrupt enable register 2

USB frame number register Low

b7 b6 b5 b4 b3 b2 b1 b0

USB frame number register Low
(USBSOFL : address 56_{16})

b	Name	Functions	At reset	R	W
0	FN0	Contains the low-order 8 bits of SOF token frame number.	0	○	×
1	FN1		0	○	×
2	FN2		0	○	×
3	FN3		0	○	×
4	FN4		0	○	×
5	FN5		0	○	×
6	FN6		0	○	×
7	FN7		0	○	×

USB frame number register High

b7 b6 b5 b4 b3 b2 b1 b0

USB frame number register High
(USBSOFH : address 57_{16})

b	Name	Functions	At reset	R	W
0	FN8	Contains the high-order 3 bits of SOF token frame number.	0	○	×
1	FN9		0	○	×
2	FN10		0	○	×
3	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are “0”.		0	○	×
4			0	○	×
5			0	○	×
6			0	○	×
7			0	○	×

Fig. 3.5.53 Structure of USB frame nmber registers Low, High

USB endpoint index register

b7 b6 b5 b4 b3 b2 b1 b0
| | | 0 | 0 | 0 | | | |

USB endpoint index register
(USBINDEX : address 58_{16})

b	Name	Functions	At reset	R	W
0	Endpoint index bit (EPINDEX)	b2b1b0 0 0 0 : Endpoint 0 0 0 1 : Endpoint 1 0 1 0 : Endpoint 2 0 1 1 : Endpoint 3 1 0 0 : Endpoint 4 1 0 1 : Not used 1 1 0 : Not used 1 1 1 : Not used	0	○	○
1			0	○	○
2			0	○	○
3	Nothing arranged for these bits. Fix these bist to		0	○	○
4			0	○	○
5			0	○	○
6	AUTO_FLUSH bit (AUTO_FL) (**Note**)	0 : Auto FIFO flush disabled 1 : Auto FIFO flush enabled	0	○	○
7	ISO_UPDATE bit (ISO_UPD) (**Note**)	0 : ISO_UPDATE disabled 1 : ISO_UPDATE enabled	0	○	○

Note: This bit is valid for an isochronous transfer.

Fig. 3.5.54 Structure of USB endpoint index register

USB endpoint 0 IN control register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint 0 IN control register
(IN_CSR : address 59_{16})

b	Name	Functions	At reset	R	W
0	OUT_PKT_RDY flag (IN0CSR0) ✻2	0 : Except the following condition (Cleared to “0” by writing “1” into SERVICED_OUT_PKT_RDY bit) 1 : End of a data packet reception	0	O	O
1	IN_PKT_RDY bit (IN0CSR1) ✻1	0 : End of a data packet transmission 1 : Write “1” at completion of writing a data packet into IN FIFO.	0	O	O
2	SEND_STALL bit (IN0CSR2) ✻3, ✻4	0 : Except the following condition 1 : Transmitting STALL handshake signal	0	O	O
3	DATA_END bit (IN0CSR3) ✻1	0 : Except the following condition (Cleared to “0” after completion of status phase) 1 : Write “1” at completion of writing or reading the last data packet to/from FIFO.	0	O	O
4	FORCE_STALL flag (IN0CSR4) ✻2, ✻3	0 : Except the following condition 1 : Protocol error detected	0	O	O
5	SETUP_END flag (IN0CSR5) (**Note 1**) ✻2	0 : Except the following condition (Cleared to “0” by writing “1” into SERVICED_SETUP_END bit) 1 : Control transfer ends before the specific length of data is transferred during the data phase.	0	O	O
6	SERVICED_OUT_PKT_RDY bit (IN0CSR6)	0 : Except the following condition 1 : Writing “1” to this bit clears OUT_PKT_RDY flag to “0”.	0	O	O
7	SERVICED_SETUP_END bit (IN0CSR7)	0 : Except the following condition 1 : Writing “1” to this bit clears SETUP_END flag to “0”.	0	O	O

USB endpoint 1, 2, 3, 4 IN control register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint 1, 2, 3, 4 IN control register
(IN_CSR : address 59_{16})

b	Name	Functions	At reset	R	W
0	INT_PKT_RDY bit (INXCSR0) ✻1	0 : End of a data packet transmission 1 : Write “1” at completion of writing a data packet into IN FIFO. (**Note 2**)	0	O	O
1	UNDER_RUN flag (INXCSR1) (In isochronous data transfer) ✻2, ✻3	0 : No FIFO underrun 1 : FIFO underrun occurred (USB overrun/underrun interrupt status flag is set to “1”.)	0	O	O
2	SEND_STALL bit (INXCSR2) ✻3, ✻4	0 : Except the following condition 1 : Transmitting STALL handshake signal	0	O	O
3	ISO/TOGGLE_INIT bit (INXCSR3) ✻3, ✻4	0 : Except the following condition 1 : Initializing to endpoint used for isochronous transfer; Initializing the data toggle sequence bit	0	O	O
4	INTPT bit (INXCSR4)	0 : Except the following condition 1 : Initializing to endpoint used for interrupt transfer, rate feedback	0	O	O
5	TX_NOT_EPT flag (INXCSR5) ✻1, ✻2	0 : Empty in IN FIFO 1 : Full in IN FIFO	0	O	×
6	FLUSH bit (INXCSR6) ✻1, ✻4	0 : Except the following condition 1 : Flush FIFO	0	O	O
7	AUTO_SET bit (INXCSR7) ✻3, ✻4	0 : AUTO_SET disabled 1 : AUTO_SET enabled (**Note 3**)	0	O	O

✻ **1:** This bit is automatically cleared to “0”.
✻ **2:** This bit is automatically set to “1”.
✻ **3:** The user must program to “0”.
✻ **4:** The user must program to “1”.

Notes 1: If this bit is set to “1”, stop accessing the FIFO to serve the previous setup transaction.
2: When AUTO_SET bit is “0”, the user must set to “1”. When AUTO_SET bit is “1”, this bit is automatically set to “1”. Additionally, when writing to other bits of this register, write “0” to this bit.
3: To use the AUTO_SET function for an IN transfer when the AUTO_SET bit is set to “1”, set the FIFO to single buffer mode.

Fig. 3.5.55 Structure of USB endpoint x (x = 0 to 4) IN control register

USB endpoint x OUT control register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x OUT control register
(OUT_CSR : address $5A_{16}$)

b	Name	Functions	At reset	R	W
0	OUT_PKT_RDY flag (OUTXCSR0) ✻4	0 : Except the following condition (**Note**) 1 : End of a data packet reception	0	O	O
1	OVER RUN flag (OUTXCSR1) (In isochronous data transfer) ✻2, ✻3	0 : No FIFO overrun 1 : FIFO overrun occurred	0	O	O
2	SEND_STALL bit (OUTXCSR2) ✻3, ✻4	0 : Except the following condition 1 : Transmitting STALL handshake signal	0	O	O
3	ISO/TOGGLE_INIT bit (OUTXCSR3) ✻3, ✻4	0 : Except the following condition 1 : Initializing to endpoint used for isochronous transfer; Enabling reception of DATA0 and DATA1 as PID (Initializing the toggle)	0	O	O
4	FORCE_STALL flag (OUTXCSR4) ✻2, ✻3	0 : Except the following condition 1 : Protocol error detected	0	O	O
5	DATA_ERR flag (OUTXCSR5) ✻2, ✻3	0 : Except the following condition 1 : CRC or bit stuffing error detected in transferring isochronous data	0	O	O
6	FLUSH bit (OUTXCSR6) ✻1, ✻4	0 : Except the following condition 1 : Flush FIFO	0	O	O
7	AUTO_SET bit (OUTXCSR7) ✻3, ✻4	0 : AUTO_SET disabled 1 : AUTO_SET enabled	0	O	O

✻ **1:** This bit is automatically cleared to "0".
✻ **2:** This bit is automatically set to "1".
✻ **3:** The user must program to "0".
✻ **4:** The user must program to "1".
Note : When AUTO_CLR bit is "0", the user must set to "0". When AUTO_CLR bit is "1", this bit is automatically set to "0".

Fig. 3.5.56 Structure of USB endpoint x (x = 1 to 4) OUT control register

USB endpoint x IN max. packet size register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x IN max. packet size register
(IN_MAXP: address $5B_{16}$)

b	Functions	At reset	R	W
0	The maximum packet size (MAXP) of endpoint x IN is contained.	(**Note**)	O	O
1	●MAXP = n for endpoints 0, 2, 3, 4	(**Note**)	O	O
2	●MAXP = n ✕ 8 for endpoint 1	(**Note**)	O	O
3	"n" is a written value into this register.	(**Note**)	O	O
4		(**Note**)	O	O
5		(**Note**)	O	O
6		(**Note**)	O	O
7		(**Note**)	O	O

Note: The value is "08_{16}" in the endpoint 1 used.
The value is "01_{16}" in the endpoint 0, 2, 3 or 4 used.

Fig. 3.5.57 Structure of USB endpoint x (x = 0 to 4) IN max. packet size register

USB endpoint x OUT max. packet size register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x OUT max. packet size register
(OUT_MAXP: address $5C_{16}$)

b	Functions	At reset	R	W
0	The maximum packet size (MAXP) of endpoint x OUT is contained. ●MAXP = n for endpoints 0, 2, 3, 4 ●MAXP = n ✕ 8 for endpoint 1 "n" is a written value into this register.	(Note)	○	○
1		(Note)	○	○
2		(Note)	○	○
3		(Note)	○	○
4		(Note)	○	○
5		(Note)	○	○
6		(Note)	○	○
7		(Note)	○	○

Note: The value is "08_{16}" in the endpoint 0, 2, 3 or 4 used.
The value is "01_{16}" in the endpoint 1 used.

Fig. 3.5.58 Structure of USB endpoint x (x = 0 to 4) OUT max. packet size register

USB endpoint x OUT write count register Low

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x OUT write count register Low
(WRT_CNTL : address $5D_{16}$)

b	Name	Functions	At reset	R	W
0	Contains the low-order 8 bits of the number of bytes in endpoint x OUT FIFO.		0	O	×
1			0	O	×
2			0	O	×
3			0	O	×
4			0	O	×
5			0	O	×
6			0	O	×
7			0	O	×

USB endpoint x OUT write count register High

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x OUT write count register High
(WRT_CNTH : address $5E_{16}$)

b	Name	Functions	At reset	R	W
0	Contains the high-order 2 bits of the number of bytes in endpoint x OUT FIFO.		0	O	×
1			0	O	×
2	Nothing is arranged for these bits. These are write disable bits. When these bits are read out, the contents are “0”.		0	O	×
3			0	O	×
4			0	O	×
5			0	O	×
6			0	O	×
7			0	O	×

Fig. 3.5.59 Structure of USB endpoint x (x = 0 to 4) OUT write count registers Low, High

USB endpoint FIFO mode register

b7 b6 b5 b4 b3 b2 b1 b0
0 0 0 0

USB endpoint FIFO mode register
(USBFIFOMR : address $5F_{16}$)

b	Name	Functions	At reset	R	W
0	FIFO size selection bit (**Note**)	For endpoint 1 b4b2b1b0 ✕ 0 0 0 : IN 512-byte, OUT 800-byte ✕ 0 0 1 : IN 1024-byte, OUT 1024-byte ✕ 0 1 0 : IN 0-byte, OUT 2048-byte ✕ 0 1 1 : IN 2048-byte, OUT 0-byte ✕ 1 0 0 : IN 768-byte, OUT 1280-byte ✕ 1 0 1 : IN 880-byte, OUT 1168-byte For endpoint 2 b4b2b1b0 0 ✕✕✕ : IN 32-byte, OUT 32-byte 1 ✕✕✕ : IN 128-byte, OUT 128-byte	0	O	O
1			0	O	O
2			0	O	O
3			0	O	O
4	Nothing arranged for these bits. Fix these bits to “0”.		0	O	O
5			0	O	O
6			0	O	O
7			0	O	O

Note: The value set into “x” is invalid.

Fig. 3.5.60 Structure of USB endpoint FIFO mode register

USB endpoint x FIFO register

b7 b6 b5 b4 b3 b2 b1 b0

USB endpoint x FIFO register
(USBFIFOx: addresses 60_{16}, 61_{16}, 62_{16}, 63_{16}, 64_{16})

b	Functions	At reset	R	W
0	●This is Endpoint x IN/OUT FIFO. ●Write a data to be transmitted into this IN FIFO. ●Read a received data from this OUT FIFO.	Undefined	O	O
1		Undefined	O	O
2		Undefined	O	O
3		Undefined	O	O
4		Undefined	O	O
5		Undefined	O	O
6		Undefined	O	O
7		Undefined	O	O

Fig. 3.5.61 Structure of USB endpoint x (x = 0 to 4) FIFO register

Flash memory control register

b7 b6 b5 b4 b3 b2 b1 b0

Flash memory control register
(FMCR : address $6A_{16}$)

b	Name	Functions	At reset	R	W
0	RY/$\overline{BY}$ status flag (FLCA0)	0 : Busy (being programmed or erased) 1 : Ready	1	O	O
1	CPU rewrite mode select bit (FLCA1) (**Note 2**)	0 : Normal mode (Software commands invalid) 1 : CPU rewrite mode (Software commands acceptable)	0	O	O
2	CPU rewrite mode entry flag (FLCA2)	0 : Normal mode 1 : CPU rewrite mode	0	O	×
3	Flash memory reset bit (FLCA3) (**Note 3**)	0 : Normal operation 1 : Reset	0	O	O
4	User ROM area / Boot ROM area select bit (FLCA4) (**Note 4**)	0 : Interrupt disabled 1 : Interrupt enabled	0	O	O
5	Indefinite at read. Write "0" at write.		0	O	O
6			0	O	O
7			0	O	O

Notes 1: The contents of flash memory control register are "XXX00001" just after reset release.
In the mask ROM version this area is reserved, so that do not write any data to this address.
2: For this bit to be set to "1", the user needs to write "0" and then "1" to it in succession. If it is not this procedure, this bit will not be set to "1". Additionally, it is required to ensure that no interrupt will be generated during that interval.
Use the control program in the area except the built-in flash memory for write to this bit.
3: This bit is valid when the CPU rewrite mode select bit is "1". Set this bit 3 to "0" subsequently after setting bit 3 to "1".
4: Use the control program in the area except the built-in flash memory for write to this bit.

Fig. 3.5.62 Structure of Flash memory control register

Frequency synthesizer control register

b7 b6 b5 b4 b3 b2 b1 b0

b7	b6	b5	b4	b3	b2	b1	b0
			0		0	0	

Frequency synthesizer control register
(FSC : address $6C_{16}$)

b	Name	Functions	At reset	R	W
0	Frequency synthesizer enable bit (FSE)	0 : Disabled 1 : Enabled	0	O	O
1	Fix these bits to “0”.		0	O	O
2			0	O	O
3	Frequency synthesizer input bit (FIN)	0 : f(XIN) 1 : f(XCIN)	0	O	O
4	Fix this bit to “0”.		0	O	O
5	LPF current control bit (CHG1, CHG0) (**Note**)	b1b0 0 0 : Not available 0 1 : Low current 1 0 : Intermediate current (recommended) 1 1 : High current	1	O	O
6			1	O	O
7	Frequency synthesizer lock status bit	0 : Unlocked 1 : Locked	0	O	O

Note: Bits 6 and 5 are set to (bit 6, bit 5) = (1, 1) at reset.
When using the frequency synthesizer, we recommend to set to (bit 6, bit 5) = (1, 0) after locking the frequency synthesizer.

Fig. 3.5.63 Structure of Frequency synthesizer control register

Frequency synthesizer multiply register 1

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer multiply register 1
(FSM1: address $6D_{16}$)

b	Functions	At reset	R	W
0	●fVCO clock is generated by multiplying fPIN clock, which is generated by FSM2, by the contents of this register: fVCO = fPIN • {2(n +1)}, n: value set to FSM1.	1	O	O
1		1	O	O
2		1	O	O
3		1	O	O
4		1	O	O
5		1	O	O
6		1	O	O
7		1	O	O

Fig. 3.5.64 Structure of Frequency synthesizer multiply register 1

Frequency synthesizer multiply register 2

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer multiply register 2
(FSM2: address $6E_{16}$)

b	Functions	At reset	R	W
0	●f_{PIN} clock is generated by dividing f_{IN} clock by the contents of this register. Either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer is selectable. $f_{PIN} = f_{IN} / \{2(n+1)\}$, n: value set to FSM2	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 3.5.65 Structure of Frequency synthesizer multiply register 2

Frequency synthesizer divide register

b7 b6 b5 b4 b3 b2 b1 b0

Frequency synthesizer divide register
(FSD: address $6F_{16}$)

b	Functions	At reset	R	W
0	●f_{SYN} clock is generated by dividing f_{VCO} clock by the contents of this register: $f_{SYN} = f_{VCO} / \{2(m+1)\}$, m: value set to FSD	1	○	○
1		1	○	○
2		1	○	○
3		1	○	○
4		1	○	○
5		1	○	○
6		1	○	○
7		1	○	○

Fig. 3.5.66 Structure of Frequency synthesizer divide register

ROM code protect control register

b7	b6	b5	b4	b3	b2	b1	b0
						1	1

ROM code protect control register
(ROMCP : address FFC9$_{16}$) (**Note 1**)

b	Name	Functions	At reset	R	W
0	Fix these bits to “1”.		1	O	O
1			1	O	O
2	ROM code protect level 2 set bits (ROMCP2) (**Notes 2, 3**)	b3b2 0 0 : Protect enabled 0 1 : Protect enabled 1 0 : Protect enabled 1 1 : Protect disabled	1	O	O
3			1	O	O
4	ROM code protect reset bits (ROMCR) (**Note 4**)	b5b4 0 0 : Protect removed 0 1 : Protect set bits effective 1 0 : Protect set bits effective 1 1 : Protect set bits effective	1	O	O
5			1	O	O
6	ROM code protect level 1 set bits (ROMCP1) (**Note 2**)	b7b6 0 0 : Protect enabled 0 1 : Protect enabled 1 0 : Protect enabled 1 1 : Protect disabled	1	O	O
7			1	O	O

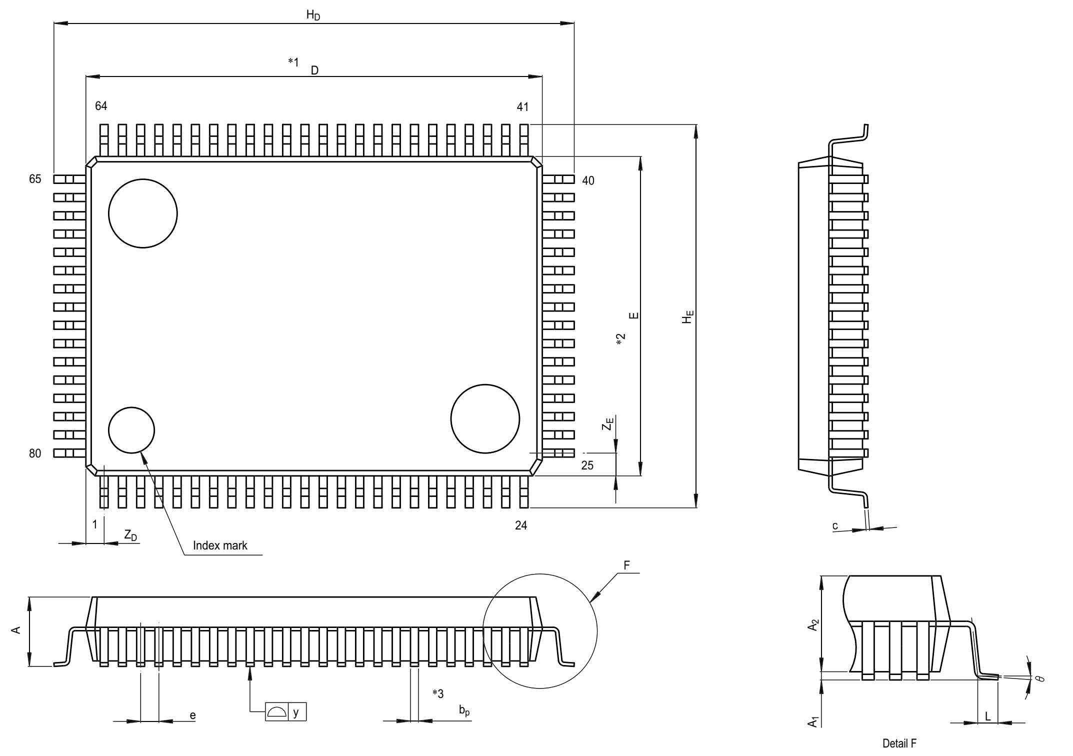
Notes 1: This area is on the ROM in the mask ROM version.
2: When ROM code protect is turned on, the internal flash memory is protected against readout or modification in parallel I/O mode.
3: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LSI tester, etc. also is inhibited.
4: The ROM code protect reset bits can be used to turn off ROM code protect level 1 and ROM code protect level 2. However, since these bits cannot be modified in parallel I/O mode, they need to be rewritten in standard serial I/O mode or CPU rewrite mode.

Fig. 3.5.67 Structure of ROM code protect control register

3.6 Package outline

PRQP0080GB-A

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-QFP80-14x20-0.80	PRQP0080GB-A	80P6N-A	1.6g

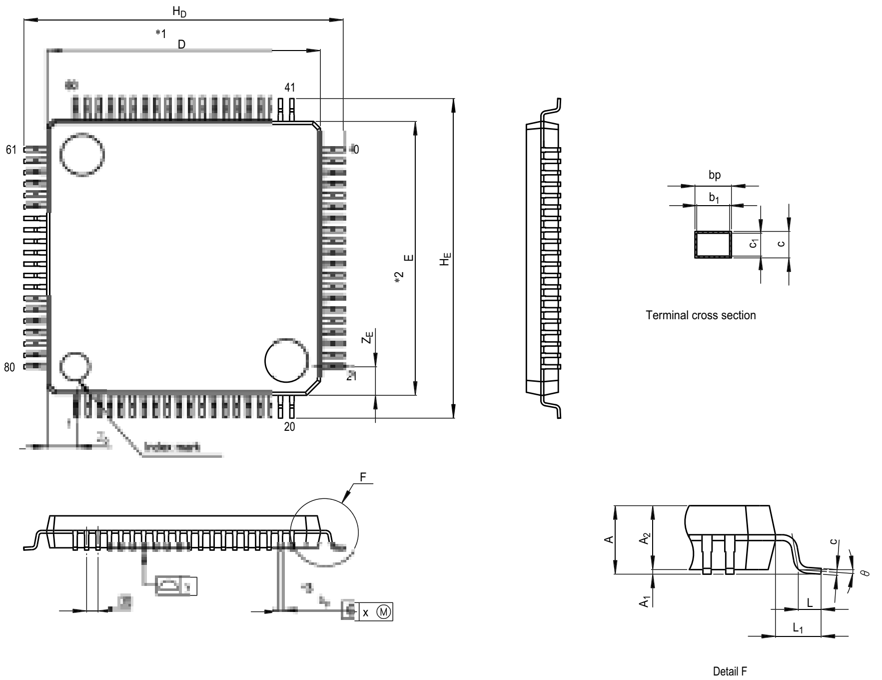


NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	19.8	20.0	20.2
E	13.8	14.0	14.2
A_2	—	2.8	—
H_D	22.5	22.8	23.1
H_E	16.5	16.8	17.1
A	—	—	3.05
A_1	0	0.1	0.2
b_p	0.3	0.35	0.45
c	0.13	0.15	0.2
θ	0°	—	10°
e	0.65	0.8	0.95
y	—	—	0.10
Z_D	—	0.8	—
Z_E	—	1.0	—
L	0.4	0.6	0.8

PLQP0080KB-A

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LQFP80-12x12-0.50	PLQP0080KB-A	80P6Q-A	0.5g



NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	11.9	12.0	12.1
E	11.9	12.0	12.1
A_2	—	1.4	—
H_D	13.8	14.0	14.2
H_E	13.8	14.0	14.2
A	—	—	1.7
A_1	0	0.1	0.2
b_p	0.15	0.20	0.25
b_1	—	0.18	—
c	0.09	0.145	0.20
c_1	—	0.125	—
θ	0°	—	10°
e	—	0.5	—
x	—	—	0.08
y	—	—	0.08
Z_D	—	1.25	—
Z_E	—	1.25	—
L	0.3	0.5	0.7
L_1	—	1.0	—

3.7 Machine instructions

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT, A, R			ZP			BIT, ZP, R		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
ADC (Note 1) (Note 7)	When T = 0 A ← A + M + C When T = 1 M(X) ← M(X) + M + C	When T = 0, this instruction adds the contents M, C, and A; and stores the results in A and C. When T = 1, this instruction adds the contents of M(X), M and C; and stores the results in M(X) and C. When T=1, the contents of A remain unchanged, but the contents of status flags are changed. M(X) represents the contents of memory where is indicated by X.				69	2	2							65	3	2			
AND (Note 1)	When T = 0 A ← A ∧ M When T = 1 M(X) ← M(X) ∧ M	When T = 0, this instruction transfers the contents of A and M to the ALU which performs a bit-wise AND operation and stores the result back in A. When T = 1, this instruction transfers the contents M(X) and M to the ALU which performs a bit-wise AND operation and stores the results back in M(X). When T = 1, the contents of A remain unchanged, but status flags are changed. M(X) represents the contents of memory where is indicated by X.				29	2	2							25	3	2			
ASL	7 0 C ← [ ] ← 0	This instruction shifts the content of A or M by one bit to the left, with bit 0 always being set to 0 and bit 7 of A or M always being contained in C.							0A	1	1				06	5	2			
BBC	Ai or Mi = 0?	This instruction tests the designated bit i of M or A and takes a branch if the bit is 0. The branch address is specified by a relative address. If the bit is 1, next instruction is executed.										13 + 20i (Note 4)	4	2				17 + 20i (Note 6)	5	3
BBS	Ai or Mi = 1?	This instruction tests the designated bit i of the M or A and takes a branch if the bit is 1. The branch address is specified by a relative address. If the bit is 0, next instruction is executed.										03 + 20i (Note 4)	4	2				07 + 20i (Note 6)	5	3
BCC (Note 5) (Note 9)	C = 0?	This instruction takes a branch to the appointed address if C is 0. The branch address is specified by a relative address. If C is 1, the next instruction is executed.																		
BCS (Note 5) (Note 9)	C = 1?	This instruction takes a branch to the appointed address if C is 1. The branch address is specified by a relative address. If C is 0, the next instruction is executed.																		
BEQ (Note 5) (Note 8)	Z = 1?	This instruction takes a branch to the appointed address when Z is 1. The branch address is specified by a relative address. If Z is 0, the next instruction is executed.																		
BIT	A ∧ M	This instruction takes a bit-wise logical AND of A and M contents; however, the contents of A and M are not modified. The contents of N, V, Z are changed, but the contents of A, M remain unchanged.													24	3	2			
BMI (Note 5) (Note 8)	N = 1?	This instruction takes a branch to the appointed address when N is 1. The branch address is specified by a relative address. If N is 0, the next instruction is executed.																		
BNE (Note 5) (Note 8)	Z = 0?	This instruction takes a branch to the appointed address if Z is 0. The branch address is specified by a relative address. If Z is 1, the next instruction is executed.																		

Symbol	Addressing mode																																	Processor status register							
	ZP, X			ZP, Y			ABS			ABS, X			ABS, Y			IND			ZP, IND			IND, X			IND, Y			REL			SP			7	6	5	4	3	2	1	0
	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
ADC	75	4	2				6D	4	3	7D	5	3	79	5	3							61	6	2	71	6	2							N	V	•	•	•	•	Z	C
AND	35	4	2				2D	4	3	3D	5	3	39	5	3							21	6	2	31	6	2							N	•	•	•	•	•	Z	•
ASL	16	6	2				0E	6	3	1E	7	3																						N	•	•	•	•	•	Z	C
BBC																																		•	•	•	•	•	•	•	•
BBS																																		•	•	•	•	•	•	•	•
BCC																												90	2	2				•	•	•	•	•	•	•	•
BCS																												B0	2	2				•	•	•	•	•	•	•	•
BEQ																												F0	2	2				•	•	•	•	•	•	•	•
BIT							2C	4	3																									M_7	M_6	•	•	•	•	Z	•
BMI																												30	2	2				•	•	•	•	•	•	•	•
BNE																												D0	2	2				•	•	•	•	•	•	•	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT, A			ZP			BIT, ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
BPL (Note 5) (Note 8)	N = 0?	This instruction takes a branch to the appointed address if N is 0. The branch address is specified by a relative address. If N is 1, the next instruction is executed.																		
BRA (Note 6)	PC ← PC ± offset	This instruction branches to the appointed address. The branch address is specified by a relative address.																		
BRK	B ← 1 (PC) ← (PC) + 2 M(S) ← PC_H S ← S – 1 M(S) ← PC_L S ← S – 1 M(S) ← PS S ← S – 1 I ← 1 PC_L ← AD_L PC_H ← AD_H	When the BRK instruction is executed, the CPU pushes the current PC contents onto the stack. The BADRS designated in the interrupt vector table is stored into the PC.	00	7	1															
BVC (Note 5)	V = 0?	This instruction takes a branch to the appointed address if V is 0. The branch address is specified by a relative address. If V is 1, the next instruction is executed.																		
BVS (Note 5)	V = 1?	This instruction takes a branch to the appointed address when V is 1. The branch address is specified by a relative address. When V is 0, the next instruction is executed.																		
CLB	Ai or Mi ← 0	This instruction clears the designated bit i of A or M.										1B + 20i	1	1				1F + 20i	5	2
CLC	C ← 0	This instruction clears C.	18	1	1															
CLD	D ← 0	This instruction clears D.	D8	1	1															
CLI	I ← 0	This instruction clears I.	58	2	1															
CLT	T ← 0	This instruction clears T.	12	1	1															
CLV	V ← 0	This instruction clears V.	B8	1	1															
CMP (Note 3)	When T = 0 A – M When T = 1 M(X) – M	When T = 0, this instruction subtracts the contents of M from the contents of A. The result is not stored and the contents of A or M are not modified. When T = 1, the CMP subtracts the contents of M from the contents of M(X). The result is not stored and the contents of X, M, and A are not modified. M(X) represents the contents of memory where is indicated by X.				C9	2	2							C5	3	2			
COM	M ← $\overline{M}$	This instruction takes the one's complement of the contents of M and stores the result in M.													44	5	2			
CPX	X – M	This instruction subtracts the contents of M from the contents of X. The result is not stored and the contents of X and M are not modified.				E0	2	2							E4	3	2			
CPY	Y – M	This instruction subtracts the contents of M from the contents of Y. The result is not stored and the contents of Y and M are not modified.				C0	2	2							C4	3	2			
DEC	A ← A – 1 or M ← M – 1	This instruction subtracts 1 from the contents of A or M.							1A	1	1				C6	5	2			

Symbol	Addressing mode																																Processor status register							
	ZP, X			ZP, Y			ABS			ABS, X			ABS, Y			IND			ZP, IND			IND, X			IND, Y			REL			SP			7	6	5	4	3	2	1	0
	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
BPL																												10	2	2				•	•	•	•	•	•	•	•
BRA																												80	3	2				•	•	•	•	•	•	•	•
BRK																																		•	•	•	1	•	1	•	•
BVC																												50	2	2				•	•	•	•	•	•	•	•
BVS																												70	2	2				•	•	•	•	•	•	•	•
CLB																																		•	•	•	•	•	•	•	•
CLC																																		•	•	•	•	•	•	•	0
CLD																																		•	•	•	•	0	•	•	•
CLI																																		•	•	•	•	•	0	•	•
CLT																																		•	•	0	•	•	•	•	•
CLV																																		•	0	•	•	•	•	•	•
CMP	D5	4	2				CD	4	3	DD	5	3	D9	5	3							C1	6	2	D1	6	2							N	•	•	•	•	•	Z	C
COM																																		N	•	•	•	•	•	Z	•
CPX							EC	4	3																									N	•	•	•	•	•	Z	C
CPY							CC	4	3																									N	•	•	•	•	•	Z	C
DEC	D6	6	2				CE	6	3	DE	7	3																						N	•	•	•	•	•	Z	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT, A			ZP			BIT, ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
DEX	X ← X – 1	This instruction subtracts one from the current contents of X.	CA	1	1															
DEY	Y ← Y – 1	This instruction subtracts one from the current contents of Y.	88	1	1															
DIV	A ← (M(zz + X + 1), M(zz + X)) / A M(S) ← one's complement of Remainder S ← S – 1	This instruction divides the 16-bit data in M(zz+(X)) (low-order byte) and M(zz+(X)+1) (high-order byte) by the contents of A. The quotient is stored in A and the one's complement of the remainder is pushed onto the stack.																		
EOR (Note 1)	When T = 0 A ← A $\forall$ M When T = 1 M(X) ← M(X) $\forall$ M	When T = 0, this instruction transfers the contents of the M and A to the ALU which performs a bit-wise Exclusive OR, and stores the result in A. When T = 1, the contents of M(X) and M are transferred to the ALU, which performs a bit-wise Exclusive OR and stores the results in M(X). The contents of A remain unchanged, but status flags are changed. M(X) represents the contents of memory where is indicated by X.				49	2	2							45	3	2			
INC	A ← A + 1 or M ← M + 1	This instruction adds one to the contents of A or M.							3A	1	1				E6	5	2			
INX	X ← X + 1	This instruction adds one to the contents of X.	E8	1	1															
INY	Y ← Y + 1	This instruction adds one to the contents of Y.	C8	1	1															
JMP	If addressing mode is ABS PC_L ← AD_L PC_H ← AD_H If addressing mode is IND PC_L ← M (AD_H, AD_L) PC_H ← M (AD_H, AD_L + 1) If addressing mode is ZP, IND PC_L ← M(00, AD_L) PC_H ← M(00, AD_L + 1)	This instruction jumps to the address designated by the following three addressing modes: Absolute Indirect Absolute Zero Page Indirect Absolute																		
JSR	M(S) ← PC_H S ← S – 1 M(S) ← PC_L S ← S – 1 After executing the above, if addressing mode is ABS, PC_L ← AD_L PC_H ← AD_H if addressing mode is SP, PC_L ← AD_L PC_H ← FF If addressing mode is ZP, IND, PC_L ← M(00, AD_L) PC_H ← M(00, AD_L + 1)	This instruction stores the contents of the PC in the stack, then jumps to the address designated by the following addressing modes: Absolute Special Page Zero Page Indirect Absolute																		
LDA (Note 2)	When T = 0 A ← M When T = 1 M(X) ← M	When T = 0, this instruction transfers the contents of M to A. When T = 1, this instruction transfers the contents of M to (M(X)). The contents of A remain unchanged, but status flags are changed. M(X) represents the contents of memory where is indicated by X.				A9	2	2							A5	3	2			
LDM	M ← nn	This instruction loads the immediate value in M.													3C	4	3			
LDX	X ← M	This instruction loads the contents of M in X.				A2	2	2							A6	3	2			
LDY	Y ← M	This instruction loads the contents of M in Y.				A0	2	2							A4	3	2			

Symbol	Addressing mode																																	Processor status register							
	ZP, X			ZP, Y			ABS			ABS, X			ABS, Y			IND			ZP, IND			IND, X			IND, Y			REL			SP			7	6	5	4	3	2	1	0
	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
DEX																																		N	•	•	•	•	•	Z	•
DEY																																		N	•	•	•	•	•	Z	•
DIV	E2	16	2																															N	V	•	•	•	•	Z	C
EOR	55	4	2				4D	4	3	5D	5	3	59	5	3							41	6	2	51	6	2							N	•	•	•	•	•	Z	•
INC	F6	6	2				EE	6	3	FE	7	3																						N	•	•	•	•	•	Z	•
INX																																		N	•	•	•	•	•	Z	•
INY																																		N	•	•	•	•	•	Z	•
JMP							4C	3	3							6C	5	3	B2	4	2													•	•	•	•	•	•	•	•
JSR							20	6	3										02	7	2										22	5	2	•	•	•	•	•	•	•	•
LDA	B5	4	2				AD	4	3	BD	5	3	B9	5	3							A1	6	2	B1	6	2							N	•	•	•	•	•	Z	•
LDM																																		•	•	•	•	•	•	•	•
LDX				B6	4	2	AE	4	3				BE	5	3																			N	•	•	•	•	•	Z	•
LDY	B4	4	2				AC	4	3	BC	5	3																						N	•	•	•	•	•	Z	•

Symbol	Function	Details	IMP OP	IMP n	IMP #	IMM OP	IMM n	IMM #	A OP	A n	A #	BIT, A OP	BIT, A n	BIT, A #	ZP OP	ZP n	ZP #	BIT, ZP OP	BIT, ZP n	BIT, ZP #
LSR	7 0 0→[ ]→C	This instruction shifts either A or M one bit to the right such that bit 7 of the result always is set to 0, and the bit 0 is stored in C.							4A	1	1				46	5	2			
MUL	M(S) • A ← A ✻ M(zz + X) S ← S – 1	This instruction multiply Accumulator with the memory specified by the Zero Page X address mode and stores the high-order byte of the result on the Stack and the low-order byte in A.																		
NOP	PC ← PC + 1	This instruction adds one to the PC but does no other operation.	EA	1	1															
ORA (Note 1)	When T = 0 A ← A V M When T = 1 M(X) ← M(X) V M	When T = 0, this instruction transfers the contents of A and M to the ALU which performs a bit-wise "OR", and stores the result in A. When T = 1, this instruction transfers the contents of M(X) and the M to the ALU which performs a bit-wise OR, and stores the result in M(X). The contents of A remain unchanged, but status flags are changed. M(X) represents the contents of memory where is indicated by X.				09	2	2							05	3	2			
PHA	M(S) ← A S ← S – 1	This instruction pushes the contents of A to the memory location designated by S, and decrements the contents of S by one.	48	3	1															
PHP	M(S) ← PS S ← S – 1	This instruction pushes the contents of PS to the memory location designated by S and decrements the contents of S by one.	08	3	1															
PLA	S ← S + 1 A ← M(S)	This instruction increments S by one and stores the contents of the memory designated by S in A.	68	4	1															
PLP	S ← S + 1 PS ← M(S)	This instruction increments S by one and stores the contents of the memory location designated by S in PS.	28	4	1															
ROL	7 0 ←[ ]←C←	This instruction shifts either A or M one bit left through C. C is stored in bit 0 and bit 7 is stored in C.							2A	1	1				26	5	2			
ROR	7 0 C→[ ]→	This instruction shifts either A or M one bit right through C. C is stored in bit 7 and bit 0 is stored in C.							6A	1	1				66	5	2			
RRF	7 0 →[ ]→	This instruction rotates 4 bits of the M content to the right.													82	8	2			
RTI	S ← S + 1 PS ← M(S) S ← S + 1 PC_L ← M(S) S ← S + 1 PC_H ← M(S)	This instruction increments S by one, and stores the contents of the memory location designated by S in PS. S is again incremented by one and stores the contents of the memory location designated by S in PC_L. S is again incremented by one and stores the contents of memory location designated by S in PC_H.	40	6	1															
RTS	S ← S + 1 PC_L ← M(S) S ← S + 1 PC_H ← M(S) (PC) ← (PC) + 1	This instruction increments S by one and stores the contents of the memory location designated by S in PC_L. S is again incremented by one and the contents of the memory location is stored in PC_H. PC is incremented by 1.	60	6	1															

Addressing mode / Processor status register

Symbol	ZP, X OP	ZP, X n	ZP, X #	ZP, Y OP	ZP, Y n	ZP, Y #	ABS OP	ABS n	ABS #	ABS, X OP	ABS, X n	ABS, X #	ABS, Y OP	ABS, Y n	ABS, Y #	IND OP	IND n	IND #	ZP, IND OP	ZP, IND n	ZP, IND #	IND, X OP	IND, X n	IND, X #	IND, Y OP	IND, Y n	IND, Y #	REL OP	REL n	REL #	SP OP	SP n	SP #	7 N	6 V	5 T	4 B	3 D	2 I	1 Z	0 C
LSR	56	6	2				4E	6	3	5E	7	3																						0	•	•	•	•	•	Z	C
MUL	62	14	2																															N	•	•	•	•	•	Z	•
NOP																																		•	•	•	•	•	•	•	•
ORA	15	4	2				0D	4	3	1D	5	3	19	5	3							01	6	2	11	6	2							N	•	•	•	•	•	Z	•
PHA																																		•	•	•	•	•	•	•	•
PHP																																		•	•	•	•	•	•	•	•
PLA																																		N	•	•	•	•	•	Z	•
PLP																																		(Value saved in stack)							
ROL	36	6	2				2E	6	3	3E	7	3																						N	•	•	•	•	•	Z	C
ROR	76	6	2				6E	6	3	7E	7	3																						N	•	•	•	•	•	Z	C
RRF																																		•	•	•	•	•	•	•	•
RTI																																		(Value saved in stack)							
RTS																																		•	•	•	•	•	•	•	•

Symbol	Function	Details	Addressing mode																	
			IMP			IMM			A			BIT, A			ZP			BIT, ZP		
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
SBC (Note 1) (Note 7)	When T = 0 A ← A – M – C When T = 1 M(X) ← M(X) – M – C	When T = 0, this instruction subtracts the value of M and the complement of C from A, and stores the results in A and C. When T = 1, the instruction subtracts the contents of M and the complement of C from the contents of M(X), and stores the results in M(X) and C. A remain unchanged, but status flag are changed. M(X) represents the contents of memory where is indicated by X.				E9	2	2							E5	3	2			
SEB	Ai or Mi ← 1	This instruction sets the designated bit i of A or M.										0B + 20i	1	1				0F + 20i	5	2
SEC	C ← 1	This instruction sets C.	38	1	1															
SED	D ← 1	This instruction set D.	F8	1	1															
SEI	I ← 1	This instruction set I.	78	2	1															
SET	T ← 1	This instruction set T.	32	1	1															
STA	M ← A	This instruction stores the contents of A in M. The contents of A does not change.													85	3	2			
STP		This instruction resets the oscillation control F/F and the oscillation stops. Reset or interrupt input is needed to wake up from this mode.	42	2	1															
STX	M ← X	This instruction stores the contents of X in M. The contents of X does not change.													86	3	2			
STY	M ← Y	This instruction stores the contents of Y in M. The contents of Y does not change.													84	3	2			
TAX	X ← A	This instruction stores the contents of A in X. The contents of A does not change.	AA	1	1															
TAY	Y ← A	This instruction stores the contents of A in Y. The contents of A does not change.	A8	1	1															
TST	M = 0?	This instruction tests whether the contents of M are “0” or not and modifies the N and Z.													64	3	2			
TSX	X ← S	This instruction transfers the contents of S in X.	BA	1	1															
TXA	A ← X	This instruction stores the contents of X in A.	8A	1	1															
TXS	S ← X	This instruction stores the contents of X in S.	9A	1	1															
TYA	A ← Y	This instruction stores the contents of Y in A.	98	1	1															
WIT		The WIT instruction stops the internal clock but not the oscillation of the oscillation circuit is not stopped. CPU starts its function after the Timer X over flows (comes to the terminal count). All registers or internal memory contents except Timer X will not change during this mode. (Of course needs VDD).	C2	2	1															

Addressing mode																																	Processor status register							
ZP, X			ZP, Y			ABS			ABS, X			ABS, Y			IND			ZP, IND			IND, X			IND, Y			REL			SP			7	6	5	4	3	2	1	0
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
F5	4	2				ED	4	3	FD	5	3	F9	5	3							E1	6	2	F1	6	2							N	V	•	•	•	•	Z	C
																																	•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	1
																																	•	•	•	•	1	•	•	•
																																	•	•	•	•	•	1	•	•
																																	•	•	1	•	•	•	•	•
95	4	2				8D	4	3	9D	5	3	99	5	3							81	6	2	91	6	2							•	•	•	•	•	•	•	•
																																	•	•	•	•	•	•	•	•
			96	4	2	8E	4	3																									•	•	•	•	•	•	•	•
94	4	2				8C	4	3																									•	•	•	•	•	•	•	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•
																																	N	•	•	•	•	•	Z	•
																																	•	•	•	•	•	•	•	•

Notes 1 : The number of cycles “n” is increased by 3 when T is 1.
2 : The number of cycles “n” is increased by 2 when T is 1.
3 : The number of cycles “n” is increased by 1 when T is 1.
4 : The number of cycles “n” is increased by 2 when branching has occurred.
5 : The number of cycles “n” is increased by 1 when branching to the same page has occurred. The number of cycles “n” is increased by 2 when branching to the other page has occurred.
6 : The number of cycles “n” is increased by 1 when branching to the other page has occurred.
7 : V flag is invalid in decimal operation mode.
8 : When this instruction is executed immediately after executing DEX, DEY, INX, INY, TAX, TSX, TXA, TYA, DEC, INC, ASL, LSR, ROL, or ROR instructions, the number of cycles “n” becomes “3”. Furthermore, the number of cycles “n” is increased by 1 (number of cycles “n” is “4”) when branching to the same page has occurred. The number of cycles “n” is increased by 2 (number of cycles “n” is “5”) when branching to the other page has occurred.
9 : When this instruction is executed immediately after executing ASL, LSR, ROL, or ROR instructions, the number of cycles “n” becomes “3”. Furthermore, the number of cycles “n” is increased by 1 (number of cycles “n” is “4”) when branching to the same page has occurred. The number of cycles “n” is increased by 2 (number of cycles “n” is “5”) when branching to the other page has occurred.

Symbol	Contents	Symbol	Contents
IMP	Implied addressing mode	+	Addition
IMM	Immediate addressing mode	–	Subtraction
A	Accumulator or Accumulator addressing mode	✻	Multiplication
BIT, A	Accumulator bit addressing mode	/	Division
BIT, A, R	Accumulator bit relative addressing mode	∧	Logical OR
ZP	Zero page addressing mode	V	Logical AND
BIT, ZP	Zero page bit addressing mode	∀	Logical exclusive OR
BIT, ZP, R	Zero page bit relative addressing mode	—	Negation
ZP, X	Zero page X addressing mode	←	Shows direction of data flow
ZP, Y	Zero page Y addressing mode	X	Index register X
ABS	Absolute addressing mode	Y	Index register Y
ABS, X	Absolute X addressing mode	S	Stack pointer
ABS, Y	Absolute Y addressing mode	PC	Program counter
IND	Indirect absolute addressing mode	PS	Processor status register
		PC_H	8 high-order bits of program counter
ZP, IND	Zero page indirect absolute addressing mode	PC_L	8 low-order bits of program counter
		AD_H	8 high-order bits of address
IND, X	Indirect X addressing mode	AD_L	8 low-order bits of address
IND, Y	Indirect Y addressing mode	FF	FF in Hexadecimal notation
REL	Relative addressing mode	nn	Immediate value
SP	Special page addressing mode	zz	Zero page address
C	Carry flag	M	Memory specified by address designation of any addressing mode
Z	Zero flag		
I	Interrupt disable flag	M(X)	Memory of address indicated by contents of index register X
D	Decimal mode flag		
B	Break flag	M(S)	Memory of address indicated by contents of stack pointer
T	X-modified arithmetic mode flag		
V	Overflow flag	M(AD_H, AD_L)	Contents of memory at address indicated by AD_H and AD_L, in AD_H is 8 high-order bits and AD_L is 8 low-order bits.
N	Negative flag		
		M(00, AD_L)	Contents of address indicated by zero page AD_L
		Ai	Bit i (i = 0 to 7) of accumulator
		Mi	Bit i (i = 0 to 7) of memory
		OP	Opcode
		n	Number of cycles
		#	Number of bytes

3.8 List of instruction code

D_3 – D_0		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
D_7 – D_4	Hexadecimal notation	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	BRK	ORA IND, X	JSR ZP, IND	BBS 0, A	—	ORA ZP	ASL ZP	BBS 0, ZP	PHP	ORA IMM	ASL A	SEB 0, A	—	ORA ABS	ASL ABS	SEB 0, ZP
0001	1	BPL	ORA IND, Y	CLT	BBC 0, A	—	ORA ZP, X	ASL ZP, X	BBC 0, ZP	CLC	ORA ABS, Y	DEC A	CLB 0, A	—	ORA ABS, X	ASL ABS, X	CLB 0, ZP
0010	2	JSR ABS	AND IND, X	JSR SP	BBS 1, A	BIT ZP	AND ZP	ROL ZP	BBS 1, ZP	PLP	AND IMM	ROL A	SEB 1, A	BIT ABS	AND ABS	ROL ABS	SEB 1, ZP
0011	3	BMI	AND IND, Y	SET	BBC 1, A	—	AND ZP, X	ROL ZP, X	BBC 1, ZP	SEC	AND ABS, Y	INC A	CLB 1, A	LDM ZP	AND ABS, X	ROL ABS, X	CLB 1, ZP
0100	4	RTI	EOR IND, X	STP	BBS 2, A	COM ZP	EOR ZP	LSR ZP	BBS 2, ZP	PHA	EOR IMM	LSR A	SEB 2, A	JMP ABS	EOR ABS	LSR ABS	SEB 2, ZP
0101	5	BVC	EOR IND, Y	—	BBC 2, A	—	EOR ZP, X	LSR ZP, X	BBC 2, ZP	CLI	EOR ABS, Y	—	CLB 2, A	—	EOR ABS, X	LSR ABS, X	CLB 2, ZP
0110	6	RTS	ADC IND, X	MUL ZP, X	BBS 3, A	TST ZP	ADC ZP	ROR ZP	BBS 3, ZP	PLA	ADC IMM	ROR A	SEB 3, A	JMP IND	ADC ABS	ROR ABS	SEB 3, ZP
0111	7	BVS	ADC IND, Y	—	BBC 3, A	—	ADC ZP, X	ROR ZP, X	BBC 3, ZP	SEI	ADC ABS, Y	—	CLB 3, A	—	ADC ABS, X	ROR ABS, X	CLB 3, ZP
1000	8	BRA	STA IND, X	RRF ZP	BBS 4, A	STY ZP	STA ZP	STX ZP	BBS 4, ZP	DEY	—	TXA	SEB 4, A	STY ABS	STA ABS	STX ABS	SEB 4, ZP
1001	9	BCC	STA IND, Y	—	BBC 4, A	STY ZP, X	STA ZP, X	STX ZP, Y	BBC 4, ZP	TYA	STA ABS, Y	TXS	CLB 4, A	—	STA ABS, X	—	CLB 4, ZP
1010	A	LDY IMM	LDA IND, X	LDX IMM	BBS 5, A	LDY ZP	LDA ZP	LDX ZP	BBS 5, ZP	TAY	LDA IMM	TAX	SEB 5, A	LDY ABS	LDA ABS	LDX ABS	SEB 5, ZP
1011	B	BCS	LDA IND, Y	JMP ZP, IND	BBC 5, A	LDY ZP, X	LDA ZP, X	LDX ZP, Y	BBC 5, ZP	CLV	LDA ABS, Y	TSX	CLB 5, A	LDY ABS, X	LDA ABS, X	LDX ABS, Y	CLB 5, ZP
1100	C	CPY IMM	CMP IND, X	WIT	BBS 6, A	CPY ZP	CMP ZP	DEC ZP	BBS 6, ZP	INY	CMP IMM	DEX	SEB 6, A	CPY ABS	CMP ABS	DEC ABS	SEB 6, ZP
1101	D	BNE	CMP IND, Y	—	BBC 6, A	—	CMP ZP, X	DEC ZP, X	BBC 6, ZP	CLD	CMP ABS, Y	—	CLB 6, A	—	CMP ABS, X	DEC ABS, X	CLB 6, ZP
1110	E	CPX IMM	SBC IND, X	DIV ZP, X	BBS 7, A	CPX ZP	SBC ZP	INC ZP	BBS 7, ZP	INX	SBC IMM	NOP	SEB 7, A	CPX ABS	SBC ABS	INC ABS	SEB 7, ZP
1111	F	BEQ	SBC IND, Y	—	BBC 7, A	—	SBC ZP, X	INC ZP, X	BBC 7, ZP	SED	SBC ABS, Y	—	CLB 7, A	—	SBC ABS, X	INC ABS, X	CLB 7, ZP

: 3-byte instruction

: 2-byte instruction

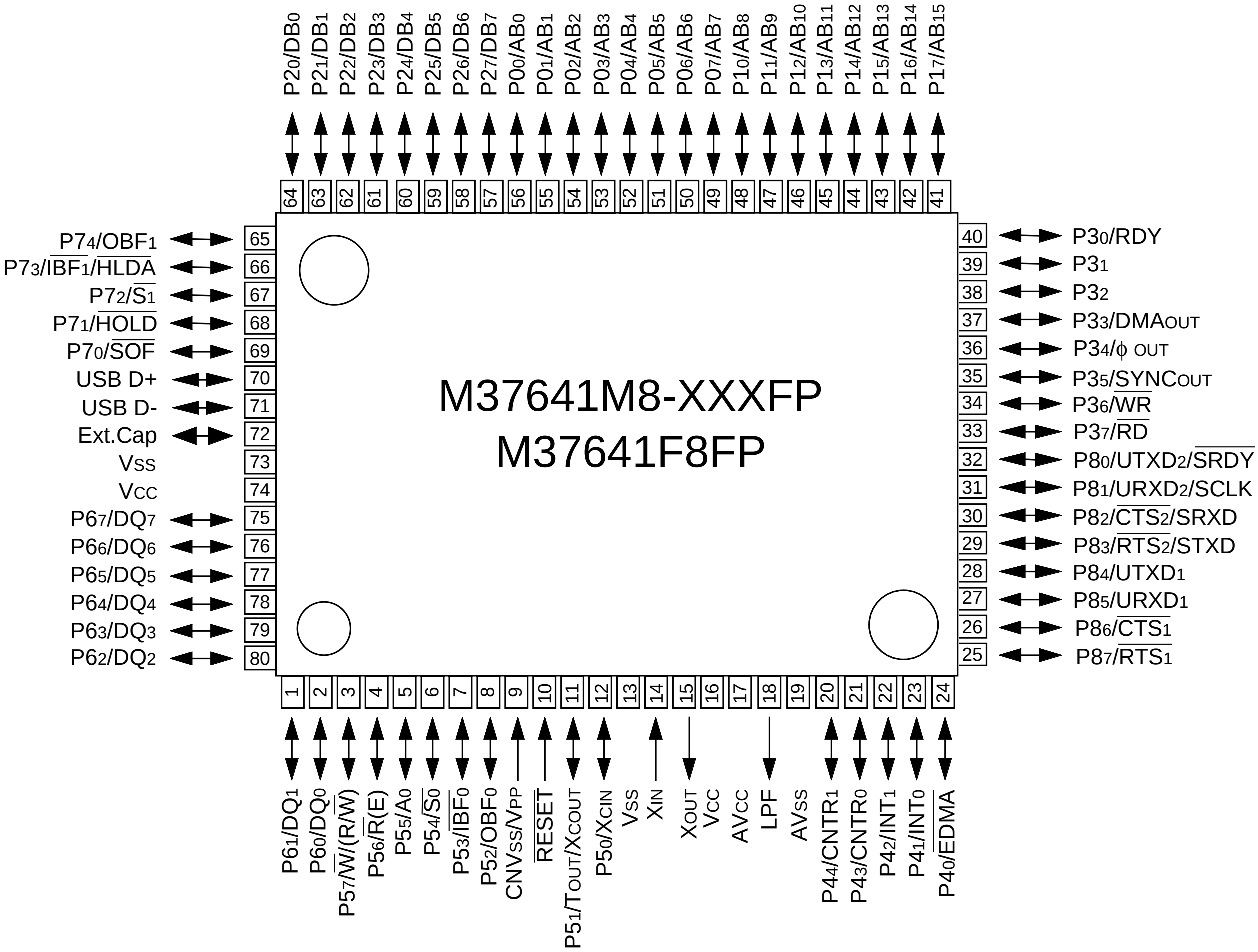
: 1-byte instruction

3.9 SFR memory map

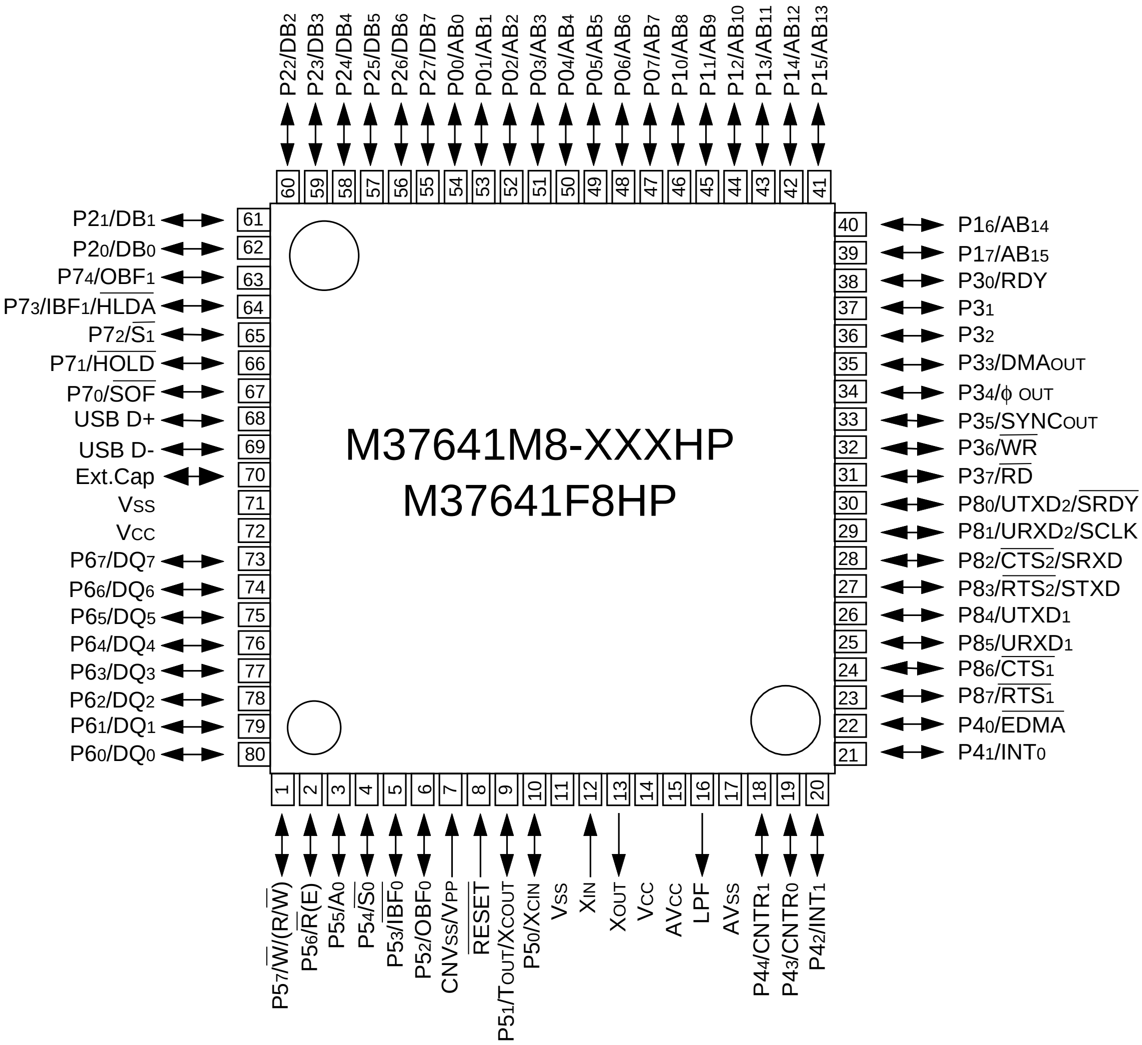
Address	Register	Address	Register
0000$_{16}$	CPU mode register A (CPUA)	0038$_{16}$	UART2 mode register (U2MOD)
0001$_{16}$	CPU mode register B (CPUB)	0039$_{16}$	UART2 baud rate generator (U2BRG)
0002$_{16}$	Interrupt request register A (IREQA)	003A$_{16}$	UART2 status register (U2STS)
0003$_{16}$	Interrupt request register B (IREQB)	003B$_{16}$	UART2 control register (U2CON)
0004$_{16}$	Interrupt request register C (IREQC)	003C$_{16}$	UART2 transmit/receive buffer register 1 (U2TRB1)
0005$_{16}$	Interrupt control register A (ICONA)	003D$_{16}$	UART2 transmit/receive buffer register 2 (U2TRB2)
0006$_{16}$	Interrupt control register B (ICONB)	003E$_{16}$	UART2 RTS control register (U2RTSC)
0007$_{16}$	Interrupt control register C (ICONC)	003F$_{16}$	DMAC index and status register (DMAIS)
0008$_{16}$	Port P0 (P0)	0040$_{16}$	DMAC channel x mode register 1 (DMAx1)
0009$_{16}$	Port P0 direction register (P0D)	0041$_{16}$	DMAC channel x mode register 2 (DMAx2)
000A$_{16}$	Port P1 (P1)	0042$_{16}$	DMAC channel x source register Low (DMAxSL)
000B$_{16}$	Port P1 direction register (P1D)	0043$_{16}$	DMAC channel x source register High (DMAxSH)
000C$_{16}$	Port P2 (P2)	0044$_{16}$	DMAC channel x destination register Low (DMAxDL)
000D$_{16}$	Port P2 direction register (P2D)	0045$_{16}$	DMAC channel x destination register High (DMAxDH)
000E$_{16}$	Port P3 (P3)	0046$_{16}$	DMAC channel x transfer count register Low (DMAxCL)
000F$_{16}$	Port P3 direction register (P3D)	0047$_{16}$	DMAC channel x transfer count register High (DMAxCH)
0010$_{16}$	Port control register (PTC)	0048$_{16}$	Data bus buffer register 0 (DBB0)
0011$_{16}$	Interrupt polarity select register (IPOL)	0049$_{16}$	Data bus buffer status register 0 (DBBS0)
0012$_{16}$	Port P2 pull-up control register (PUP2)	004A$_{16}$	Data bus buffer control register 0 (DBBC0)
0013$_{16}$	USB control register (USBC)	004B$_{16}$	Resereved (**Note 1**)
0014$_{16}$	Port P6 (P6)	004C$_{16}$	Data bus buffer register 1 (DBB1)
0015$_{16}$	Port P6 direction register (P6D)	004D$_{16}$	Data bus buffer status register 1 (DBBS1)
0016$_{16}$	Port P5 (P5)	004E$_{16}$	Data bus buffer control register 1 (DBBC1)
0017$_{16}$	Port P5 direction register (P5D)	004F$_{16}$	Resereved (**Note 1**)
0018$_{16}$	Port P4 (P4)	0050$_{16}$	USB address register (USBA)
0019$_{16}$	Port P4 direction register (P4D)	0051$_{16}$	USB power management register (USBPM)
001A$_{16}$	Port P7 (P7)	0052$_{16}$	USB interrupt status register 1 (USBIS1)
001B$_{16}$	Port P7 direction register (P7D)	0053$_{16}$	USB interrupt status register 2 (USBIS2)
001C$_{16}$	Port P8 (P8)	0054$_{16}$	USB interrupt enable register 1 (USBIE1)
001D$_{16}$	Port P8 direction register (P8D)	0055$_{16}$	USB interrupt enable register 2 (USBIE2)
001E$_{16}$	Resereved (**Note 1**)	0056$_{16}$	USB frame number register Low (USBSOFL)
001F$_{16}$	Clock control register (CCR)	0057$_{16}$	USB frame number register High (USBSOFH)
0020$_{16}$	Timer XL (TXL)	0058$_{16}$	USB endpoint index register (USBINDEX)
0021$_{16}$	Timer XH (TXH)	0059$_{16}$	USB endpoint x IN control register (IN_CSR)
0022$_{16}$	Timer YL (TYL)	005A$_{16}$	USB endpoint x OUT control register (OUT_CSR)
0023$_{16}$	Timer YH (TYH)	005B$_{16}$	USB endpoint x IN max. packet size register (IN_MAXP)
0024$_{16}$	Timer 1 (T1)	005C$_{16}$	USB endpoint x OUT max. packet size register (OUT_MAXP)
0025$_{16}$	Timer 2 (T2)	005D$_{16}$	USB endpoint x OUT write count register Low (WRT_CNTL)
0026$_{16}$	Timer 3 (T3)	005E$_{16}$	USB endpoint x OUT write count register High (WRT_CNTH)
0027$_{16}$	Timer X mode register (TXM)	005F$_{16}$	USB endpoint FIFO mode register (USBFIFOMR)
0028$_{16}$	Timer Y mode register (TYM)	0060$_{16}$	USB endpoint 0 FIFO (USBFIFO0)
0029$_{16}$	Timer 123 mode register (T123M)	0061$_{16}$	USB endpoint 1 FIFO (USBFIFO1)
002A$_{16}$	Serial I/O shift register (SIOSHT)	0062$_{16}$	USB endpoint 2 FIFO (USBFIFO2)
002B$_{16}$	Serial I/O control register 1 (SIOCON1)	0063$_{16}$	USB endpoint 3 FIFO (USBFIFO3)
002C$_{16}$	Serial I/O control register 2 (SIOCON2)	0064$_{16}$	USB endpoint 4 FIFO (USBFIFO4)
002D$_{16}$	Special count source generator 1 (SCSG1)	0065$_{16}$	Resereved (**Note 1**)
002E$_{16}$	Special count source generator 2 (SCSG2)	0066$_{16}$	Resereved (**Note 1**)
002F$_{16}$	Special count source mode register (SCSGM)	0067$_{16}$	Resereved (**Note 1**)
0030$_{16}$	UART1 mode register (U1MOD)	0068$_{16}$	Resereved (**Note 1**)
0031$_{16}$	UART1 baud rate generator (U1BRG)	0069$_{16}$	Resereved (**Note 1**)
0032$_{16}$	UART1 status register (U1STS)	006A$_{16}$	Flash memory control register (FMCR) (**Note 2**)
0033$_{16}$	UART1 control register (U1CON)	006B$_{16}$	Resereved (**Note 1**)
0034$_{16}$	UART1 transmit/receive buffer register 1 (U1TRB1)	006C$_{16}$	Frequency synthesizer control register (FSC)
0035$_{16}$	UART1 transmit/receive buffer register 2 (U1TRB2)	006D$_{16}$	Frequency synthesizer multiply register 1 (FSM1)
0036$_{16}$	UART1 RTS control register (U1RTSC)	006E$_{16}$	Frequency synthesizer multiply register 2 (FSM2)
0037$_{16}$	Resereved (**Note 1**)	006F$_{16}$	Frequency synthesizer divide register (FSD)
		FFC9$_{16}$	ROM code protect control register (ROMCP) (**Note 3**)

Notes 1: Do not write any data to this addresses, because these areas are reserved.
2: This area is reserved in the mask ROM version.
3: This area is on the ROM in the mask ROM version.

3.10 Pin configuration



Package type: PRQP0080GB-A (Top view)



Package type: PLQP0080KB-A (Top view)

REVISION HISTORY	7641 GROUP USER'S MANUAL

Rev.	Date	Description	
		Page	Summary
1.0	02/26/2002		First edition
2.0	08/28/2006	All pages	Package names "80P6N-A" → "PRQP0080GB-A" revised Package names "80P6Q-A" → "PLQP0080KB-A" revised
		All pages	"USB std. spec. ver.1.1" → "Full-Speed USB2.0 specification"
		Chapter 1	
		39	DMAC; "(DxCEN)" → "(DxHR)"
		55	Fig. 47 "4: To use the AUTO_SET function .... to single buffer mode." added
		71	CLOCK GENERATING CIRCUIT; "No external resistor is needed .... resistor exists on-chip." → "No external resistor is needed .... depending on conditions.) Fig. 64; Pulled up added, NOTE added
		108	UART; "•Do not update .... data might be output." added
		109	USB; "•Use the AUTO_FLUSH Bit .... buffer mode.", "•To use the AUTO_SET function .... to single buffer mode." added
		111	Oscillator Connection Notice; "The built-in feedback register (400) .... pins X_{IN} and X_{OUT}." → "The built-in feedback register (1 M) .... pins X_{IN} and X_{OUT}." Power Source Voltage added
		112	USB Communication added "For the mask ROM confirmation .... http://www.infomicom.maec.co.jp/indexe.htm" → "For the mask ROM confirmation .... (http://www.renesas.com)."
		Chapter 2	
		95	2.5.6 "64-byte", "64 bytes" → "128-byte", "128 bytes" "USB endpoint 1" → "USB endpoint 2"
		96-98	Fig. 2.5.16, Fig. 2.5.17, Fig. 2.5.18 revised
		Chapter 3	
		30	(5) Receive error flag added
		32	3.3.5 (4) USB Communication added
		35	(5) Registers and bits; "•To use the AUTO_SET function .... to single buffer mode." added
		83	Fig. 3.5.55 "3: To use the AUTO_SET function .... to single buffer mode." added
		92	3.6 Package outline revised

RENESAS 8-BIT SINGLE-CHIP MICROCOMPUTER
USER'S MANUAL
7641 Group

Publication Data : Rev.2.00 Aug 28, 2006

Published by : Sales Strategic Planning Div.
Renesas Technology Corp.

7641 Group
User's Manual

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ09B0336-0200