

TUSB5152

**USB to 2-Serial + 1-Parallel Controller
With Configurable Optional Hub**

Data Manual

NOTE

Designing with this device may require extensive support. Before incorporating this device into a design, customers should contact TI or an Authorized TI Distributor.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

Contents

| <i>Section</i> | <i>Title</i> | <i>Page</i> |
|----------------|--|-------------|
| 1 | Controller Description | 1–1 |
| 2 | Main Features | 2–1 |
| 2.1 | General Features | 2–1 |
| 2.2 | Enhanced UART Features | 2–1 |
| 2.3 | IEEE-1284 Port | 2–2 |
| 3 | Device Parameter Information | 3–1 |
| 3.1 | Hub-Ports LED Status Definition | 3–4 |
| 3.2 | Connecting an External Microcontroller for Development | 3–5 |
| 3.3 | Setting the TRST and $\overline{\text{TEST}}[2:0]$ Pins for Various Applications | 3–6 |
| 3.4 | Pin State in Suspend Mode | 3–6 |
| 3.5 | Reset Timing | 3–7 |
| 4 | MCU Memory Map (Internal Operation) | 4–1 |
| 4.1 | Miscellaneous Registers | 4–2 |
| 4.1.1 | ROMS: ROM Shadow Configuration Register | 4–2 |
| 4.1.2 | Boot Operation (MCU Firmware Loading) | 4–2 |
| 4.1.3 | GLOBCTL: Global Control Register | 4–3 |
| 4.2 | Buffers + I/O RAM Map | 4–3 |
| 4.3 | Endpoint Descriptor Block (EDB-1 to EDB-7) | 4–6 |
| 4.3.1 | OEPCNF_n: Output Endpoint Configuration | 4–7 |
| 4.3.2 | OEPBBAX_n: Output Endpoint X-Buffer Base-Address | 4–7 |
| 4.3.3 | OEPBCTX_n: Output Endpoint X-Byte Count | 4–8 |
| 4.3.4 | OEPBBAY_n: Output Endpoint Y-Buffer Base-Address | 4–8 |
| 4.3.5 | OEPBCTY_n: Output Endpoint Y-Byte Count | 4–8 |
| 4.3.6 | OEPSIZXY_n: Output Endpoint X/Y-Buffer Size | 4–9 |
| 4.3.7 | IEPCNF_n: Input Endpoint Configuration | 4–9 |
| 4.3.8 | IEPBBAX_n: Input Endpoint X-Buffer Base-Address | 4–9 |
| 4.3.9 | IEPBCTX_n: Input Endpoint X-Byte Count | 4–10 |
| 4.3.10 | IEPBBAY_n: Input Endpoint Y-Buffer Base-Address | 4–10 |
| 4.3.11 | IEPBCTY_n: Input Endpoint Y-Byte Count | 4–10 |
| 4.3.12 | IEPSIZXY_n: Input Endpoint X/Y-Buffer Size | 4–11 |
| 4.4 | Endpoint-0 Descriptor Registers | 4–11 |
| 4.4.1 | IEPCNFG_0: Input Endpoint-0 Configuration Register | 4–11 |
| 4.4.2 | IEPBCNT_0: Input Endpoint-0 Byte Count Register | 4–12 |
| 4.4.3 | OEPCNFG_0: Output Endpoint-0 Configuration Register | 4–12 |
| 4.4.4 | OEPBCNT_0: Output Endpoint X-Byte Count Register | 4–13 |
| 5 | USB Registers | 5–1 |

| | | |
|----------|---|------------|
| 5.1 | FUNADR: Function Address Register | 5-1 |
| 5.2 | USBSTA: USB Status Register | 5-1 |
| 5.3 | OEPNCFG_0: Output Endpoint-0 Configuration Register | 5-2 |
| 5.4 | OEPBCNT_0: Output Endpoint-0 Byte Count Register | 5-2 |
| 5.5 | USBMSK: USB Interrupt Mask Register | 5-3 |
| 5.6 | USBCTL: USB Control Register | 5-4 |
| 5.7 | HUBCNF1: Hub-Configuration-1 Register | 5-5 |
| 5.8 | HUBCNF2: Hub-Configuration-2 Register | 5-6 |
| 5.9 | HUBPOTG: Hub Power-On to Power-Good Descriptor Register ... | 5-6 |
| 5.10 | HUBPIDL: Hub-PID Register (Low-Byte) | 5-6 |
| 5.11 | HUBPIDH: Hub-PID Register (High-Byte) | 5-6 |
| 5.12 | HUBVIDL: Hub-VID Register (Low-Byte) | 5-7 |
| 5.13 | HUBVIDH: Hub-VID Register (High-Byte) | 5-7 |
| 5.14 | Function Reset and Power-Up Reset Interconnect | 5-7 |
| 5.15 | Pullup Resistor Connect/Disconnect | 5-8 |
| 6 | DMA Controller | 6-1 |
| 6.1 | DMA Controller Registers | 6-1 |
| 6.1.1 | DMACDR[2:1]: DMA Channel Definition Register | 6-2 |
| 6.1.2 | DMACSR[2:1]: DMA Control and Status Register | 6-3 |
| 6.1.3 | DMACDR[4:3]: DMA Channel Definition Register | 6-4 |
| 6.1.4 | DMACSR[4:3]: DMA Control and Status Register | 6-5 |
| 6.1.5 | DMACDR[5]: DMA-5 Channel Definition Register | 6-6 |
| 6.1.6 | DMACSR[5]: DMA-5 Control and Status Register | 6-7 |
| 6.2 | Bulk Data I/O Using the EDB | 6-7 |
| 7 | UARTs | 7-1 |
| 7.1 | UART Registers | 7-1 |
| 7.1.1 | RDR[2:1]: Receiver Data Registers | 7-1 |
| 7.1.2 | TDR[2:1]: Transmitter Data Registers | 7-1 |
| 7.1.3 | LCR[2:1]: Line Control Registers | 7-2 |
| 7.1.4 | FCRL[2:1]: UART Flow Control Registers | 7-3 |
| 7.1.5 | MCR[2:1]: Modem-Control Registers | 7-5 |
| 7.1.6 | LSR[2:1]: Line-Status Registers | 7-6 |
| 7.1.7 | MSR[2:1]: Modem-Status Registers | 7-7 |
| 7.1.8 | DLL[2:1]: Divisor Low-Byte Registers | 7-7 |
| 7.1.9 | DLH[2:1]: Divisor High-Byte Registers | 7-8 |
| 7.1.10 | XON[2:1]: Xon Registers | 7-9 |
| 7.1.11 | XOFF[2:1]: Xoff Registers | 7-9 |
| 7.1.12 | MASK[2:1]: UART Interrupt-Mask Registers | 7-9 |
| 7.2 | UART Data Transfer | 7-9 |
| 7.2.1 | Receiver Data Flow | 7-10 |
| 7.2.2 | Hardware Flow Control | 7-10 |
| 7.2.3 | Auto-RTS (Receiver Control) | 7-10 |
| 7.2.4 | Auto-CTS (Transmitter Control) | 7-11 |
| 7.2.5 | Xon/Xoff Receiver Flow Control | 7-11 |

| | | |
|-----------|---|-------------|
| 7.2.6 | Xon/Xoff Transmitter Flow Control | 7–11 |
| 8 | IEEE 1284 Parallel Port | 8–1 |
| 8.1 | 1284 Registers | 8–1 |
| 8.1.1 | PPCNT: P-Port Counter Register | 8–2 |
| 8.1.2 | PPMCR: P-Port Mode Control Register | 8–3 |
| 8.1.3 | PPIMSK: P-Port Interrupt Mask Register | 8–4 |
| 8.1.4 | PPSTA: P-Port Status Register | 8–4 |
| 8.1.5 | PPCTL: P-Port Control Register | 8–5 |
| 8.1.6 | PPADR: EPP/ECP Address Register | 8–5 |
| 8.1.7 | PPDAT: P-Port Data Register | 8–6 |
| 8.2 | Mode-1 (SPP): Bidirectional Centronics Mode | 8–6 |
| 8.2.1 | MCU Output Sequence | 8–6 |
| 8.2.2 | MCU Input Sequence | 8–6 |
| 8.2.3 | DMA Input Sequence | 8–6 |
| 8.3 | Mode-2/3: EPP and ECP Modes | 8–7 |
| 8.3.1 | DMA Output Sequence | 8–7 |
| 8.3.2 | DMA Input Sequence | 8–7 |
| 8.4 | Host IN/OUT Transaction From/To P-Port | 8–8 |
| 8.4.1 | Host IN Transaction | 8–8 |
| 8.4.2 | Host OUT Transaction | 8–9 |
| 9 | Interrupts | 9–1 |
| 9.1 | 8052 Interrupt and Status Registers | 9–1 |
| 9.1.1 | 8052 Standard Interrupt Enable Register | 9–1 |
| 9.1.2 | Additional Interrupt Sources | 9–1 |
| 9.1.3 | IEPINT: Input Endpoint Interrupt Request Register | 9–2 |
| 9.1.4 | OEPINT: Output Endpoint Interrupt Request Register | 9–2 |
| 9.1.5 | VECINT: Vector Interrupt Register | 9–2 |
| 9.1.6 | Logical Interrupt Connection Diagram (Internal/External) . | 9–4 |
| 10 | I²C Port | 10–1 |
| 10.1 | I ² C Registers | 10–1 |
| 10.1.1 | I ² CSTA: I ² C Status and Control Register | 10–1 |
| 10.1.2 | I ² CADR: I ² C Address Register | 10–2 |
| 10.1.3 | I ² CDAI: I ² C Data-Input Register | 10–2 |
| 10.1.4 | I ² CDAO: I ² C Data-Output Register | 10–2 |
| 10.2 | Random-Read Operation | 10–2 |
| 10.2.1 | Device Address + EEPROM [High Byte] | 10–2 |
| 10.2.2 | EEPROM [Low Byte] | 10–3 |
| 10.3 | Current-Address Read Operation | 10–3 |
| 10.4 | Sequential Read Operation | 10–3 |
| 10.4.1 | Device Address | 10–3 |
| 10.4.2 | N-Byte Read (31 Bytes) | 10–4 |
| 10.4.3 | Last-Byte Read (Byte 32) | 10–4 |
| 10.5 | Byte-Write Operation | 10–4 |
| 10.5.1 | Device Address + EEPROM [High Byte] | 10–4 |

| | | |
|-----------|---|-------------|
| 10.5.2 | EEPROM [Low Byte] | 10-4 |
| 10.5.3 | EEPROM [DATA] | 10-4 |
| 10.6 | Page Write | 10-5 |
| 10.6.1 | Device Address + EEPROM [High Byte] | 10-5 |
| 10.6.2 | EEPROM [Low Byte] | 10-5 |
| 10.6.3 | EEPROM [DATA] - 31 Bytes | 10-5 |
| 10.6.4 | EEPROM [DATA] - Last Byte | 10-5 |
| 11 | Electrical Specifications | 11-1 |
| 11.1 | Absolute Maximum Ratings Over Operating Free-Air Temperature Range | 11-1 |
| 11.2 | Recommended Operating Conditions | 11-1 |
| 11.3 | Electrical Characteristics Over Recommended Ranges of Operating Free-Air Temperature and Supply Voltage | 11-2 |
| 11.4 | Differential Driver Switching Characteristics Over Recommended Ranges of Operating Free-Air Temperature and Supply Voltage, $C_L = 50 \text{ pF}$ | 11-2 |
| 11.4.1 | Full-Speed Mode | 11-2 |
| 11.4.2 | Low-Speed Mode | 11-2 |
| 11.5 | Current Consumption, $T_A = 25^\circ\text{C}$, $V_{CC} = V_{CCS} = 3.3 \text{ V} \pm 5\%$, $V_{SS} = 0 \text{ V}$ | 11-2 |
| 12 | Application | 12-1 |
| 13 | Boot Code | 13-1 |
| 14 | Mechanical Information | 14-1 |

List of Illustrations

| <i>Figure</i> | <i>Title</i> | <i>Page</i> |
|---------------|--|-------------|
| 2-1 | Controller Block Diagram | 2-2 |
| 3-1 | Controller 100-Pin TQFP Pin Diagram | 3-1 |
| 3-2 | Dual LED Connection to LED and SUSP Signals | 3-4 |
| 3-3 | Connecting to an External Microcontroller | 3-5 |
| 3-4 | Crystal Circuit | 3-6 |
| 3-5 | Relationship of Power, Reset, and Clock Signals | 3-7 |
| 4-1 | MCU Memory Map | 4-1 |
| 5-1 | Reset Diagram | 5-8 |
| 5-2 | Pullup Resistor Connect/Disconnect Circuit | 5-8 |
| 6-1 | Transaction Time-Out Diagram | 6-5 |
| 7-1 | MSR and MCR Registers in Loopback Mode | 7-6 |
| 7-2 | Receiver/Transmitter Data Flow | 7-10 |
| 7-3 | Auto Flow Control Interconnect | 7-10 |
| 8-1 | P-Port Data I/O Flow | 8-1 |
| 8-2 | USB IN Transaction From P-Port (DMA CNT Mode) | 8-8 |
| 8-3 | USB OUT Transaction to P-Port | 8-9 |
| 9-1 | Internal Vector Interrupt | 9-4 |
| 11-1 | Differential Driver Switching Load | 11-3 |
| 11-2 | Differential Driver Timing Waveforms | 11-3 |
| 11-3 | Differential Receiver Input Sensitivity vs Common Mode Input Range ... | 11-3 |
| 11-4 | Single-Ended Receiver Input Signal Parameter Definitions | 11-3 |
| 12-1 | 4-Port Hub, Two Serial- and One Parallel-Port Implementation | 12-1 |
| 12-2 | RS485 Bus Implementation | 12-2 |
| 12-3 | Quad UART Implementation | 12-2 |

List of Tables

| <i>Table</i> | <i>Title</i> | <i>Page</i> |
|--------------|--|-------------|
| 3-1 | Controller Pin Description (100-Pin TQFP) | 3-1 |
| 3-2 | Internal/External Microcontroller Signal Definitions | 3-4 |
| 3-3 | Relation Between LED Signals and Hub-Port States | 3-4 |
| 3-4 | Signals Used for External Microcontroller | 3-5 |
| 3-5 | SUSP Pin Output | 3-7 |
| 4-1 | ROM/RAM Size Definition Table | 4-2 |
| 4-2 | XDATA Space | 4-3 |
| 4-3 | Memory Mapped Registers Summary (XDATA Range = FF80-FFFF) | 4-4 |
| 4-4 | EDB Memory Locations | 4-6 |
| 4-5 | EDB Entries in RAM | 4-7 |
| 4-6 | Input/Output EDB-0 Registers | 4-11 |
| 6-1 | DMA Channel Allocation | 6-1 |
| 6-2 | DMA OUT-Termination Condition | 6-3 |
| 6-3 | DMA IN-Termination Condition | 6-5 |
| 7-1 | UART Register Summary | 7-1 |
| 7-2 | Transmitter Flow-Control Modes | 7-4 |
| 7-3 | Receiver Flow-Control Modes | 7-4 |
| 7-4 | DLL/DLH Values and Resulting Baud Rates | 7-8 |
| 8-1 | Parallel-Port Registers | 8-1 |
| 8-2 | Parallel-Port Mode Summary | 8-1 |
| 8-3 | Parallel-Port Signal Definition Summary | 8-2 |
| 8-4 | PP Output-Pin Activation Source | 8-5 |
| 9-1 | 8052 Interrupt Location Map | 9-1 |
| 9-2 | Vector Interrupt Values | 9-3 |

1 Controller Description

This controller provides bridging between the USB, the dual-enhanced UARTs, and the IEEE-1284 bidirectional parallel port. This device contains all the necessary logic to communicate with the host computer using the USB bus. In addition to the USB host interface, the device contains a 5-port hub that can be used for USB expansion. The internal 8052 microcontroller contains 16K RAM that can be loaded from the host. All the device functions, such as the USB command decoding, UARTs/1284-port setup, and error reporting are managed by the internal microcontroller unit (MCU) firmware.

2 Main Features

2.1 General Features

- Fully compliant with USB release 1.1 specifications
- Supports 12-Mbps USB data rate (full speed)
- Supports USB suspend and resume operations
- One upstream port and five downstream ports (full and low speed)
- Can support a total of 8 input endpoints and 8 output endpoints
- Dual-enhanced UARTs
- Integrated 8052 microcontroller with
 - 256 × 8 RAM for internal data
 - 6K × 8 ROM (with USB and I²C boot loader)
 - 16K × 8 RAM for code space. Totally loadable from host or I²C port.
 - 2K × 8 synchronous RAM used for data buffers and EDBs
 - Two 8052 GPIO ports, Port-1 and Port-3
 - Master I²C controller for external device accesses.
- Supports external microcontroller interface for development ease
- Built-in five-channel DMA controller for USB/UART/P-port bulk I/O
- Operates from a 6-MHz crystal. On-chip PLL generates 48/24 MHz and 7.384615 MHz for internal baud-rate generator
- Power-down mode
- Available in 100-pin TQFP
- 3.3/5-V operation

Figure 2–1 is a block diagram showing the major functional areas of the TUSB5152.

2.2 Enhanced UART Features

- Software/hardware flow control
 - Programmable Xon/Xoff characters
 - Programmable auto- $\overline{\text{RTS}}/\overline{\text{DTR}}$ and auto- $\overline{\text{CTS}}/\overline{\text{DSR}}$
- Automatic RS485-bus transceiver control, with and without echo
- Software selectable baud rate from 50 to 460,800 baud
- Programmable serial-interface characteristics
 - 5-, 6-, 7- or 8-bit characters
 - Even, odd, or no parity-bit generation and detection
- 1, 1.5 or 2 stop-bit generation
- Line break generation and detection
- Internal test and loop-back capabilities
- Modem-control functions ($\overline{\text{CTS}}$, $\overline{\text{RTS}}$, $\overline{\text{DSR}}$, $\overline{\text{DTR}}$, $\overline{\text{RI}}$, and $\overline{\text{DCD}}$)
- Internal diagnostics capability
 - Loopback control for communications link-fault isolation
 - Break, parity, overrun, framing-error simulations

3 Device Parameter Information

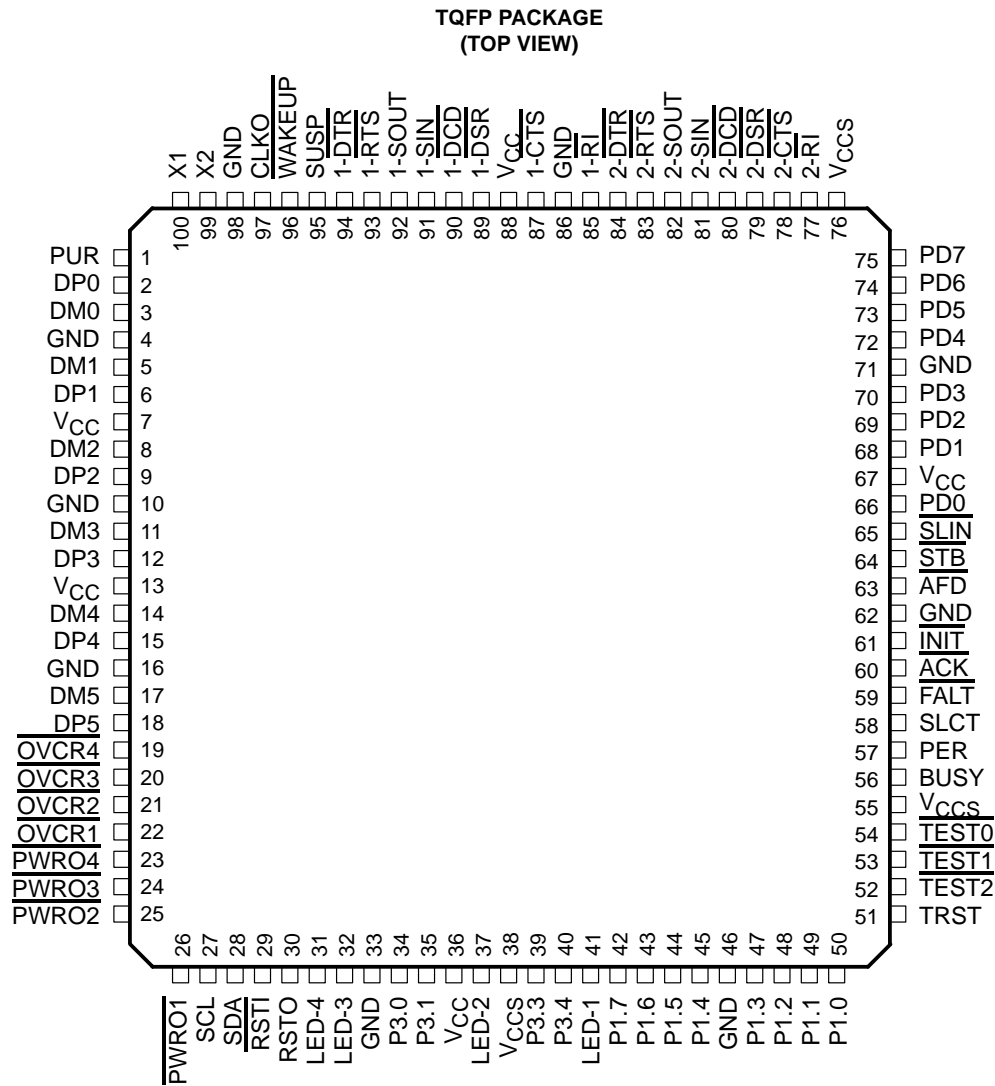


Figure 3–1. Controller 100-Pin TQFP Pin Diagram

Table 3–1. Controller Pin Description (100-Pin TQFP)

| NAME | PIN NO. | I/O | NOTES | 3/5V (see Note 9) | DESCRIPTION |
|-------|---------|-----|--------|----------------------|--|
| DP0 | 2 | I/O | (2) | T | Root USB port differential data plus |
| DM0 | 3 | I/O | (2) | | Root USB port differential data minus |
| PUR | 1 | O | (7) | | Pullup resistor connection |
| DP1 | 6 | I/O | (2)(4) | | Port-1: Downstream differential data plus |
| DM1 | 5 | I/O | (2)(4) | | Port-1: Downstream differential data minus |
| PWRO1 | 26 | O | (1) | | Port-1: Power on/off control signal |
| OVCR1 | 22 | I | (2) | | Port-1: Overcorrect indicator |
| DP2 | 9 | I/O | (2)(4) | | Port-2: Downstream differential data plus |

Table 3–1. Controller Pin Description (100-Pin TQFP) (Continued)

| NAME | PIN NO. | I/O | NOTES | 3/5V (see Note 9) | DESCRIPTION |
|--------|---------|-----|--------|----------------------|--|
| DM2 | 8 | I/O | (2)(4) | T | Port-2: Downstream differential data minus |
| PWRO2 | 25 | O | (1) | | Port-2: Power on/off control signal |
| OVCR2 | 21 | I | (2) | | Port-2: Overcorrect indicator |
| DP3 | 12 | I/O | (2)(4) | | Port-3: Downstream differential data plus |
| DM3 | 11 | I/O | (2)(4) | T | Port-3: Downstream differential data minus |
| PWRO3 | 24 | O | (1) | | Port-3: Power on/off control signal |
| OVCR3 | 20 | I | (2) | | Port-3: Overcorrect indicator |
| DP4 | 15 | I/O | (2)(4) | | Port-4: Downstream differential data plus |
| DM4 | 14 | I/O | (2)(4) | T | Port-4: Downstream differential data minus |
| PWRO4 | 23 | O | (1) | | Port-4: Power on/off control signal |
| OVCR4 | 19 | I | (2) | | Port-4: Overcorrect indicator |
| DP5 | 18 | I/O | (2)(4) | | Port-5: Downstream differential data plus |
| DM5 | 17 | I/O | (2)(4) | | Port-5: Downstream differential data minus |
| SUSP | 95 | O | (7) | | Suspend condition signal |
| | | | | | MCU Port-1 (8-bits) |
| P1.0 | 50 | I/O | (6) | T | Bit-0 GPIO |
| P1.1 | 49 | I/O | (6) | T | Bit-1 GPIO |
| P1.2 | 48 | I/O | (6) | T | Bit-2 GPIO |
| P1.3 | 47 | I/O | (6) | T | Bit-3 GPIO |
| P1.4 | 45 | I/O | (6) | T | Bit-4 GPIO |
| P1.5 | 44 | I/O | (6) | T | Bit-5 GPIO |
| P1.6 | 43 | I/O | (6) | T | Bit-6 GPIO |
| P1.7 | 42 | I/O | (6) | T | Bit-7 GPIO |
| P3.0 | 34 | I/O | (6) | T | Port-3.0 |
| P3.1 | 35 | I/O | (6) | T | Port-3.1 |
| P3.3 | 39 | I/O | (6) | T | Port-3.3 |
| P3.4 | 40 | I/O | (6) | T | Port-3.4 |
| LED-1 | 41 | I/O | (2) | | ALE or LED-1 output |
| LED-2 | 37 | O | (1) | | External interrupt output or LED-2 output |
| LED-3 | 32 | I/O | (2) | | Write-signal input or LED-3 output |
| LED-4 | 31 | I/O | (2) | | Read-signal input or LED-4 output |
| 1-DTR | 94 | O | (7) | | UART1: Data terminal ready |
| 1-RTS | 93 | O | (7) | | UART1: Request to send |
| 1-SOUT | 92 | O | (7) | | UART1: Serial output data |
| 1-SIN | 91 | I | (2)(3) | | UART1: Serial input data |
| 1-DCD | 90 | I | | | UART1: Data carrier detect |
| 1-DSR | 89 | I | | | UART1: Data set ready |
| 1-CTS | 87 | I | | | UART1: Clear to send |
| 1-RI | 85 | I | | | UART1: Ring indicator |
| 2-DTR | 84 | O | (7) | | UART2: Data terminal ready |
| 2-RTS | 83 | O | (7) | | UART2: Request to send |
| 2-SOUT | 82 | O | (7) | | UART2: Serial output data |

Table 3–1. Controller Pin Description (100-Pin TQFP) (Continued)

| NAME | PIN NO. | I/O | NOTES | 3/5V (see Note 9) | DESCRIPTION |
|------------------|--|-----|-----------|----------------------|--|
| 2-SIN | 81 | I | (2)(3) | | UART2: Serial input data |
| 2-DCD | 80 | I | | | UART2: Data carrier detect |
| 2-DSR | 79 | I | | | UART2: Data set ready |
| 2-CTS | 78 | I | | | UART2: Clear to send |
| 2-RI | 77 | I | | | UART2: Ring indicator |
| PD[7:0] | 75–72, 70–68, 66 | I/O | (2)(5) | Y | 1284-port: Data bus |
| SLIN | 65 | O | (5) | Y | 1284-port: Select input |
| STB | 64 | O | (5) | Y | 1284-port: Data strobe |
| AFD | 63 | O | (5) | Y | 1284-port: Auto feed |
| INIT | 61 | O | (5) | Y | 1284-port: Initialize |
| ACK | 60 | I | (2)(5) | Y | 1284-port: Acknowledge signal |
| FALT | 59 | I | (2)(5) | Y | 1284-port: Fault |
| SLCT | 58 | I | (2)(5) | Y | 1284-port: Select |
| PER | 57 | I | (2)(5) | Y | 1284-port: Paper end |
| BUSY | 56 | I | (2)(5) | Y | 1284-port: Busy indication |
| SDA | 28 | I/O | (1)(2)(3) | T | Master I ² C controller: data signal |
| SCL | 27 | O | (8) | T | Master I ² C controller: clock signal |
| CLKO | 97 | O | (1) | | Programmable clock output (see GLOBCTL: global control register) |
| WAKEUP | 96 | I | (2)(3) | T | Remote wake-up request bit. When low, wakes up system |
| X2 | 99 | O | | | 6-MHz crystal output |
| X1 | 100 | I | | | 6-MHz crystal input or clock input |
| TRST | 51 | I | (3) | | Test reset input (left open in normal operation) |
| RSTI | 29 | I | (2) | T | Controller master reset signal |
| RSTO | 30 | O | (7) | Y | Reset output (see GLOBCTL: global control register) |
| TEST0 | 54 | I | (3) | | Test inputs (tied to V _{CC} in normal operation) |
| TEST1 | 53 | I | (3) | | Test inputs (tied to V _{CC} in normal operation) |
| TEST2 | 52 | I | (3) | | Test inputs (tied to V _{CC} in normal operation) |
| GND | 4, 10, 16, 33, 46, 62, 71, 86, 98 | GND | | | Digital ground |
| V _{CC} | 7, 13, 36, 67, 88 | PWR | | | 3.3V |
| V _{CCS} | 38, 55, 76 | PWR | | | 5V |

- NOTES:
- 3-state CMOS output
 - Schmitt-trigger input
 - 100 μ A active pullup
 - Unused downstream ports can be left open or tied to ground.
 - IEEE 1284 compliant port without internal pullup
 - 3-state CMOS output configured to emulate an open drain (without internal pullup). The MCU drives the pin high for two clock cycles before entering the high-impedance state.
 - Push-pull output
 - Open drain
 - Y indicates 5-V operation. T indicates 5-V tolerance (output is 3.3 V, input can be 3.3 V or 5 V).

Table 3–2. Internal/External Microcontroller Signal Definitions

| EXTEN = 1; INTERNAL MICROCONTROLLER | | | EXTEN = 0; EXTERNAL MICROCONTROLLER | | |
|-------------------------------------|--------------------|-----|-------------------------------------|-----------------------------------|-----|
| P1.[7:0] | Port-1: 8-bit GPIO | I/O | P0.[7:0] | 8-Bit data/address bus | I/O |
| P3.0 | GPIO/RxD | I/O | AD8 | Address line A8 | I |
| P3.1 | GPIO/TxD | I/O | AD9 | Address line A9 | I |
| P3.3 | GPIO/INT1 | I/O | AD10 | Address line A10 | I |
| P3.4 | GPIO/T0 | I/O | P3.4 | Address line A15 | I |
| LED-1 | LED-1 output | O | ALE | Address latch enable | I |
| LED-2 | LED-2 output | O | XINT0 | Interrupt output | O |
| LED-3 | LED-3 output | O | WR | External data memory write strobe | I |
| LED-4 | LED-4 output | O | RD | External data memory read strobe | I |

3.1 Hub-Ports LED Status Definition

Four LED outputs are provided to indicate the corresponding USB port states (LED-1 corresponds to Port-1, etc.). All the LED signals are CMOS with ± 4 -mA sink/drive capability. Table 3–3 describes the LED signal level with its corresponding port states. The colors generated correspond to the circuit implementation shown in Figure 3–2.

Table 3–3. Relation Between LED Signals and Hub-Port States

| PORT-POWERED | ENABLED | DEVICE CONNECTED | SUSP-SIGNAL | LEDn OUTPUT | COLOR | PORT SUSPEND |
|--------------|---------|------------------|-------------|---------------------|---------------|--------------|
| No | X | X | 0 | Flash (see Note 10) | Red/Green | No |
| Yes | X | No | 0 | 1 | Red | No |
| Yes | Yes | Yes | 0 | 0 | Green | No |
| Yes | No | Yes | 0 | Flash (see Note 10) | Red/Green | No |
| X | X | X | 1 | No change | Blank | No |
| Yes | Yes | Yes | 1 | 3-State | Blank | Yes |
| Yes | Yes | Yes | 0 | 3-State | Red and green | Yes |

NOTE 10: LED output is 512 ms on and 512 ms off.

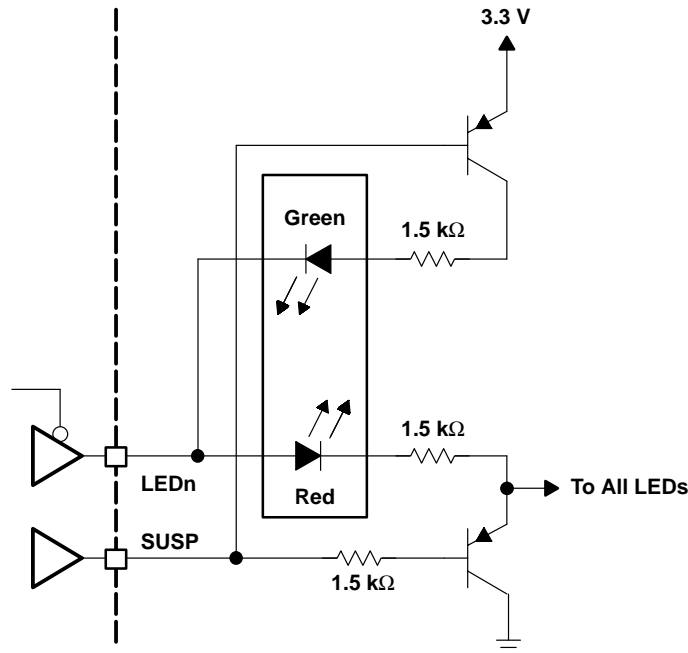


Figure 3–2. Dual LED Connection to LED and SUSP Signals

3.2 Connecting an External Microcontroller for Development

Figure 3–3 illustrates how to connect an external 8052 microcontroller to the TUSB5152. Pin $\overline{\text{TEST0}}$ must be connected to ground to enable the external mode ($\overline{\text{TEST}}[2:0] = 110\text{b}$). Table 3–4 outlines the signals used to connect the TUSB5152 to the external microcontroller.

Table 3–4. Signals Used for External Microcontroller

| TUSB5152 NAME | EXTERNAL 8052 MICROCONTROLLER | |
|--------------------------|-------------------------------|------------------------------------|
| | NAME | COMMENTS |
| P1.0 | P0.0 | Port-0: Bit-0 data bus |
| P1.1 | P0.1 | Port-0: Bit-1 data bus |
| P1.2 | P0.2 | Port-0: Bit-2 data bus |
| P1.3 | P0.3 | Port-0: Bit-3 data bus |
| P1.4 | P0.4 | Port-0: Bit-4 data bus |
| P1.5 | P0.5 | Port-0: Bit-5 data bus |
| P1.6 | P0.6 | Port-0: Bit-6 data bus |
| P1.7 | P0.7 | Port-0: Bit-7 data bus |
| P3.0 | P2.0 | Address line A8 |
| P3.1 | P2.1 | Address line A9 |
| P3.3 | P2.2 | Address line A10 |
| P3.4 | P2.7 | Address line A15 |
| LED-1 | ALE | Address latch enable |
| LED-2 | $\overline{\text{INT0}}$ | Interrupt input to microcontroller |
| LED-3 | $\overline{\text{WR}}$ | External data memory write strobe |
| LED-4 | $\overline{\text{RD}}$ | External data memory read strobe |
| $\overline{\text{RST0}}$ | RST | Master reset input |

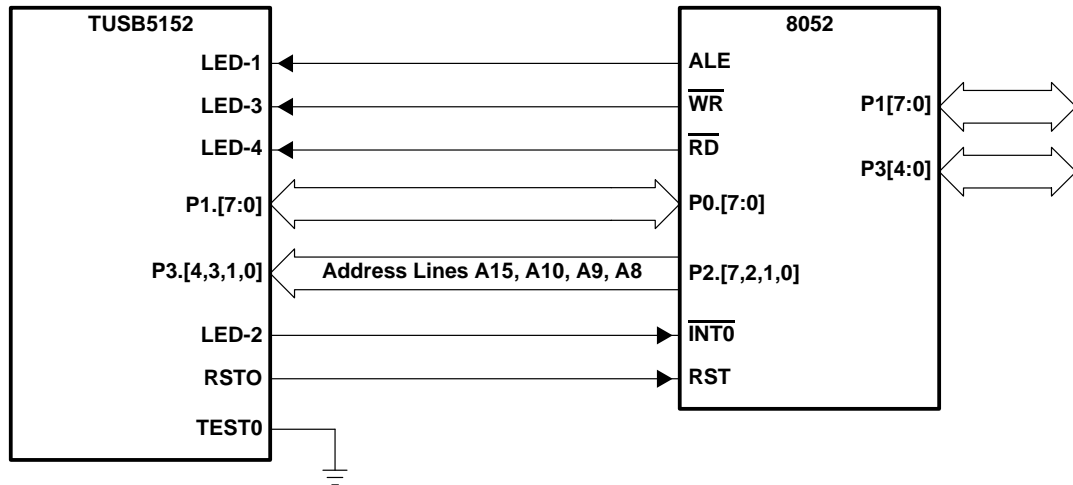


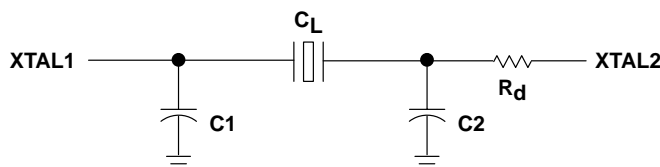
Figure 3–3. Connecting to an External Microcontroller

3.3 Setting the TRST and $\overline{\text{TEST}}[2:0]$ Pins for Various Applications

Various combinations of the TRST and $\overline{\text{TEST}}[2:0]$ pins must be grounded to enable the following operational modes.

- 6-MHz operation. This is the normal operational mode. A 6-MHz crystal is required. To enable this mode, TRST and $\overline{\text{TEST}}[2:0]$ are tied high.
- 48-MHz operation. This mode bypasses the internal APLL. A 48-MHz external oscillator or clock source is required. To enable this mode, $\overline{\text{TEST}}[2:0]$ must be grounded. TRST must be tied high.
- Bypassing the internal hub. In this mode, downstream port 1 to port 5 is disabled. Port 6 (function) is logically connected directly to the upstream port. To enable this mode, $\overline{\text{TEST}}1$ and $\overline{\text{TEST}}0$ must be grounded. The TRST and $\overline{\text{TEST}}2$ pins must be tied high. A 6-MHz crystal is required in this mode.

For 6-MHz operation, the TUSB5152 requires a 6-MHz crystal to be connected across the XTAL1 and XTAL2 pins. Internal APLL circuitry generates a 48-MHz internal clock to sample the data from the upstream port. For 48-MHz operation, the device accepts a 48-MHz clock on the XTAL1 input, and the XTAL2 input should be left unconnected. A crystal cannot be used in this case. If low-power suspend and resume are desired, a 6-MHz crystal/resonator must be used, unless there is a way to stop the 48-MHz clock source. If the 48-MHz clock is stopped in suspend mode, it has to be turned on in the event of a remote wake-up in order for the device to pass the resume signal upstream. If an oscillator is used by connecting its output to the XTAL1 pin and leaving the XTAL2 pin open, its TTL output level cannot exceed 3.6 V. Figure 3–4 shows a sample crystal circuit.



NOTE: Figure 3–4 assumes a 6-MHz fundamental crystal that is parallel-loaded. The component values of C1, C2 and R_d were determined using a crystal from Fox Electronics—part number HC49U–6.00MHz30\50\0±70\20—which means ± 30 ppm at 25°C and 50 ppm from 0°C to 70°C. The characteristics for the crystal are load capacitance (C_L) of 20 pF, maximum shunt capacitance (C_0) of 7 pF and a maximum ESR of 50 Ω . If the load capacitance is lower, then the maximum ESR can be higher. For example, if $C_L = 15$ pF and $C_0 = 7$ pF, then a maximum ESR of 100 Ω will still work. In order to ensure enough negative resistance, use $C1 = C2 = 2 \times C_L - 5$ pF stray capacitance. The resistor R_d is used to trim the gain, and $R_d = 2.2$ k Ω is recommended.

Figure 3–4. Crystal Circuit

3.4 Pin State in Suspend Mode

In suspend mode, the following pins become high-impedance:

DP/DM—USB data bus

When going into suspend mode, the following pins remain in their pre-suspended state:

P1—8051 GPIO

P3—8051 GPIO

LED-1 to LED-4—Hub downstream port status

PD[7:0]—IEEE 1284 data bus

SLIN, STB, AFD and INIT—IEEE 1284 control outputs

DTR, RTS, SOUT—UART outputs

SDA, SCL—I²C

CLKO—Clock output

RSTO—Reset output

PUR—Pullup resistor connection

PWRO—Power on/off control signal

All the input pins must be pulled to either high or low to prevent the bus line from oscillating.

The output of the SUSP pin relative to other conditions of the device is listed in Table 3–5.

Table 3–5. SUSP Pin Output

| HUB BYPASSED | HUB SUSPENDED | FUNCTION SUSPENDED | IDLE BIT | SUSP OUTPUT |
|--------------|---------------|--------------------|----------|-------------|
| No | Yes | X | 1 | 1 |
| No | Yes | X | 0 | 0 |
| No | No | Yes | 1 | Z |
| No | No | X | 0 | 0 |
| No | No | No | X | 0 |
| Yes | X | Yes | 1 | 1 |
| Yes | X | X | 0 | 0 |
| Yes | X | No | X | 0 |

3.5 Reset Timing

There are two requirements for the reset signal timing. First, the minimum reset-pulse duration is 200 μs . At power up, this time is measured from the time the power ramps up to 90% of the nominal V_{CC} until the reset signal is no longer active (reset is active as long as it is less than 1.2 V). The second requirement is that the clock must be valid during the last 180 μs of the reset window. The clock is valid when the oscillation on the XTAL2 pin exceeds 1.2 V p-p. Figure 3–5 illustrates the relationships among the power, reset, and clock signals. Note that when using a 6-MHz crystal and the on-chip oscillator, the clock signal may take several milliseconds following power up to ramp up and become valid. Therefore, the reset window may need to be extended up to 10 ms or more to ensure that there is a 180- μs overlap with a valid clock. Also, note that according to the USB specification, the chip has 100 ms to come out of reset and be ready to respond to the host. The reset signal is inactive when it goes above 1.8 V.

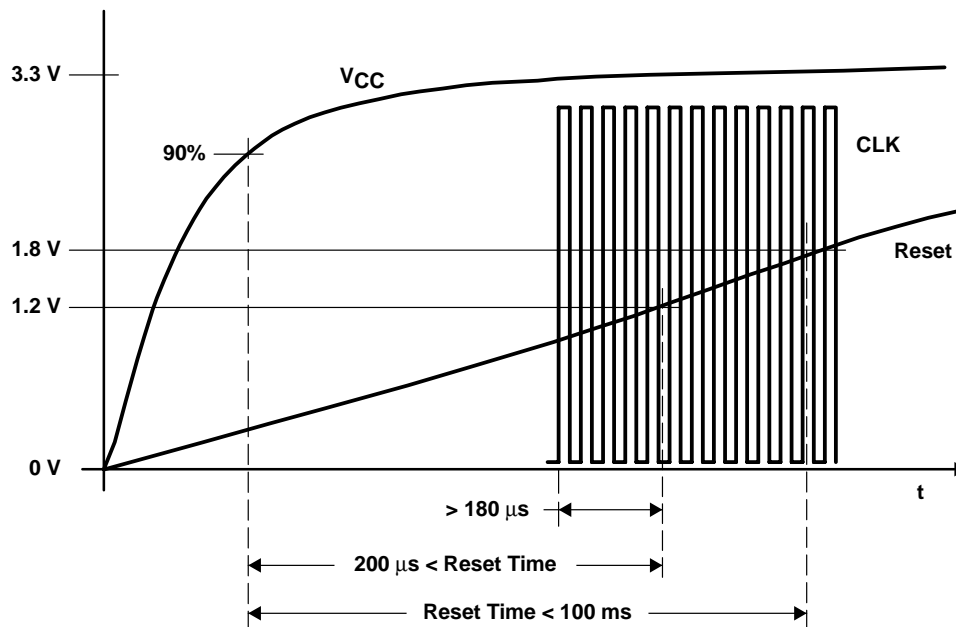


Figure 3–5. Relationship of Power, Reset, and Clock Signals

4 MCU Memory Map (Internal Operation)

Figure 4–1 illustrates the MCU memory map under boot and normal operation. Note that the internal 256 bytes of RAM are not shown since they are assumed to be in the standard 8052 location (0000 to 00FF). The shaded areas represent the internal ROM/RAM.

When SDW bit = 0 (boot mode): The 6K ROM is mapped to address (0x0000–0x17FF) and is duplicated in location (0x8000–0x97FF) in code space. The internal 16K RAM is mapped to address range (0x0000–0x3FFF) in data space. Buffers, memory-map registers (MMR) and I/O are mapped to address range (0xF800–0xFFFF) in data space.

When SDW bit = 1 (normal mode): The 6K ROM is mapped to (0x8000–0x97FF) in code space. The internal 16K RAM is mapped to address range (0x0000–0x3FFF) in code space. Buffers, MMR and I/O are mapped to address range (0xF800–0xFFFF) in data space.

| | | Boot Mode (SDW = 0) | | | Normal Mode (SDW = 1) | | |
|------|--|---------------------|--|---------------------|------------------------------|--|------------|
| | | CODE | | XDATA | CODE | | XDATA |
| 0000 | | 6K Boot ROM | | (16K) Read/Write | 16K Code RAM Read Only | | |
| 17FF | | | | | | | |
| | | | | | | | |
| 3FFF | | | | | | | |
| | | | | | | | |
| 8000 | | 6K Boot ROM | | | 6K Boot ROM | | |
| 97FF | | | | | | | |
| | | | | | | | |
| F800 | | | | 2K Data | | | 2K Data |
| FF7F | | | | | | | |
| FF80 | | | | MMR | | | MMR |
| FFFF | | | | | | | |

Figure 4–1. MCU Memory Map

4.1 Miscellaneous Registers

4.1.1 ROMS: ROM Shadow Configuration Register

This register is used by the MCU to switch from boot mode to normal operation mode (boot mode is set on power-on-reset only). In addition, this register provides the device revision number and the ROM/RAM configuration.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ROA | S1 | S0 | R3 | R2 | R1 | R0 | SDW |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-----------|--|
| 0 | SDW | 0 | <p>This bit enables/disables boot ROM. (Shadow the ROM)</p> <p>SDW = 0 When clear, the MCU executes from the 6K boot-ROM space. The boot ROM appears in two locations: 0000 and 8000h. The 16K RAM is mapped to XDATA space; therefore, read/write operation is possible. This bit is set by the MCU after the RAM load is completed. MCU cannot clear this bit; it is cleared on power-up-reset or function-reset.</p> <p>SDW = 1 When set by the MCU, the 6K boot-ROM maps to location 8000h, and the 16K RAM is mapped to code space, starting at location 0000h. At this point, the MCU executes from RAM, and the write operation is disabled (no write operation is possible in code space).</p> |
| 4–1 | R[3:0] | No effect | These bits reflect the device revision number. |
| 6–5 | S[1:0] | No effect | <p>Code space size. These bits define the ROM or RAM code-space size (ROA bit defines ROM or RAM). These bits are permanently set and are not affected by reset (see Table 4–1).</p> <p>00 = 4K bytes code space size 01 = 8K bytes code space size 10 = 16K bytes code space size 11 = 32K bytes code space size</p> |
| 7 | ROA | No effect | <p>ROM or RAM version. This bit indicates whether the code space is RAM or ROM based. This bit is permanently set and is not affected by reset (see Table 4–1).</p> <p>ROA = 0 Code space is ROM ROA = 1 Code space is RAM</p> |

Table 4–1. ROM/RAM Size Definition Table

| ROMS REGISTER | | | Boot-ROM | RAM CODE | ROM CODE |
|---------------|----|----|----------|----------------|----------------|
| ROA | S1 | S0 | | | |
| 0 | 0 | 0 | None | None | 4K |
| 0 | 0 | 1 | None | None | 8K |
| 0 | 1 | 0 | None | None | 16K (reserved) |
| 0 | 1 | 1 | None | None | 32K (reserved) |
| 1 | 0 | 0 | 6K | 4K | None |
| 1 | 0 | 1 | 6K | 8K | None |
| 1 | 1 | 0 | 6K | 16K | None |
| 1 | 1 | 1 | 6K | 32K (reserved) | None |

4.1.2 Boot Operation (MCU Firmware Loading)

Since the code space is in RAM (with the exception of the boot ROM), the TUSB5152 firmware must be loaded from an external source. Two sources are available for booting: one from an external serial EEPROM connected to the I²C bus, and the other from the host via the USB. On device reset, the SDW bit (in ROMS register) and CONT bit (in USBCTL: USB control register) are cleared. This configures the memory space to *boot mode* (see Memory-Map) and keeps the device *disconnected* from the host. The first instruction is fetched from location 0000h (which is in the 6K-ROM). The 16K-RAM is mapped to XDATA space (location 0000h). The MCU executes a read from an external EEPROM and tests whether it contains the code (by testing for boot signature). If it contains the code, the MCU reads from EEPROM and writes to the 16K-RAM in XDATA space. If it does not contain the code, the MCU proceeds to boot from the USB.

Once the code is loaded, the MCU sets SDW = 1. This switches the memory map to *normal-mode*, i.e. the 16K-RAM is mapped to code space, and the MCU starts executing from location 0000h. Once the switch is done, the MCU sets CONT = 1 (in the USBCTL register). This *connects* the device to the USB and results in normal USB device enumeration.

4.1.3 GLOBCTL: Global Control Register

This register is used to control the MCU clock rate, power-down mode, and the CLKO frequency.

| | | | | | | | |
|-------|-----|-----|-------|------|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 12/24 | C1 | C0 | CLKOE | RSTP | RSV | RSV | RSV |
| R/W | R/W | R/W | R/W | R/W | R/O | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 2–0 | RSV | 00b | Reserved = 0 |
| 3 | RSTP | 1 | Reset-output polarity RSTP = 0 RSTO output is active low RSTP = 1 RSTO output is active high |
| 4 | CLKOE | 0 | 0 = enable 1 = disable |
| 6–5 | C[1:0] | 00b | Output clock frequency selection 00 = Output clock = 48 MHz 01 = Output clock = 24 MHz 10 = Output clock = 12 MHz 11 = Output clock = 6 MHz |
| 7 | 12/24 | 0 | This bit selects a 12-MHz or 24-MHz clock for the MCU 12/24=0 12 MHz 12/24=1 24 MHz |

4.2 Buffers + I/O RAM Map

The address range from F800 to FFFF (2K bytes) is reserved for data buffers, setup packet, end-point descriptors block (EDB) and all I/O. There are 128 locations reserved for memory mapped registers (MMR). Table 4–2 represents the XDATA space allocation and access restriction for the DMA, UBM and MCU. Table 4–3 lists the memory-mapped registers in the TUSB5152.

Table 4–2. XDATA Space

| DESCRIPTION | ADDRESS RANGE | UBM ACCESS | DMA ACCESS | MCU ACCESS |
|--|-------------------|------------------------|---|------------|
| Internal MMRs (memory mapped registers) | FFFF ↑ FF80 | No (Only EDB-0) | No (Only data register and EDB-0) | Yes |
| EDB (endpoint descriptors block) | FF7F ↑ FF08 | Only for EDB update | Only for EDB update | Yes |
| Setup packet | FF07 ↑ FF00 | Yes | No | Yes |
| Input endpoint_0 buffer | FEFF ↑ FEF8 | Yes | Yes | Yes |
| Output endpoint_0 buffer | FEF7 ↑ FEF0 | Yes | Yes | Yes |
| Data buffers | FEFF ↑ F800 | Yes | Yes | Yes |

Table 4–3. Memory Mapped Registers Summary (XDATA Range = FF80–FFFF)

| ADDRESS | REGISTER | DESCRIPTION |
|---------|----------------------|--|
| FFFF | FUNADR | Function address register |
| FFFE | USBSTA | USB status register |
| FFFD | USBMSK | USB interrupt mask register |
| FFFC | USBCTL | USB control register |
| FFFB | HUBVIDH | Hub VID high-byte register |
| FFFA | HUBVIDL | Hub VID low-byte register |
| FFF9 | HUBPIDH | Hub PID high-byte register |
| FFF8 | HUBPIDL | Hub PID low-byte register |
| FFF7 | HUBCNF1 | Hub-configuration-1 register |
| FFF6 | HUBCNF2 | Hub-configuration-2 register |
| FFF5 | HUBPOTG | Hub power-on to power-good descriptor register |
| ↑ | RESERVED | |
| FFF3 | I ² CADR | I ² C-port address register |
| FFF2 | I ² CDATI | I ² C-port data input register |
| FFF1 | I ² CDATO | I ² C-port data output register |
| FFF0 | I ² CSTA | I ² C-port status register |
| ↑ | RESERVED | |
| FFE9 | DMACSR5 | DMA-5: Control and status register |
| FFE8 | DMACDR5 | DMA-5: Channel definition register |
| FFE7 | DMACSR4 | DMA-4: Control and status register |
| FFE6 | DMACDR4 | DMA-4: Channel definition register |
| FFE5 | DMACSR3 | DMA-3: Control and status register |
| FFE4 | DMACDR3 | DMA-3: Channel definition register |
| FFE3 | DMACSR2 | DMA-2: Control and status register |
| FFE2 | DMACDR2 | DMA-2: Channel definition register |
| FFE1 | DMACSR1 | DMA-1: Control and status register |
| FFE0 | DMACDR1 | DMA-1: Channel definition register |
| ↑ | RESERVED | |
| FFBB | MASK2 | UART2: Interrupt mask register |
| FFBA | XOFF2 | UART2: Xoff register |
| FFB9 | XON2 | UART2: Xon register |
| FFB8 | DLH2 | UART2: Divisor high-byte register |
| FFB7 | DLL2 | UART2: Divisor low-byte register |
| FFB6 | MSR2 | UART2: Modem status register |
| FFB5 | LSR2 | UART2: Line status register |
| FFB4 | MCR2 | UART2: Modem control register |
| FFB3 | FCRL2 | UART2: Flow control register |
| FFB2 | LCR2 | UART2: Line control registers |
| FFB1 | TDR2 | UART2: Transmitter data registers |
| FFB0 | RDR2 | UART2: Receiver data registers |
| ↑ | RESERVED | |
| FFAB | MASK1 | UART1: Interrupt mask register |
| FFAA | XOFF1 | UART1: Xoff register |
| FFA9 | XON1 | UART1: Xon register |

Table 4–3. Memory Mapped Registers Summary (XDATA Range = FF80–FFFF) (Continued)

| ADDRESS | REGISTER | DESCRIPTION |
|---------|-----------|---|
| FFA8 | DLH1 | UART1: Divisor high-byte register |
| FFA7 | DLL1 | UART1: Divisor low-byte register |
| FFA6 | MSR1 | UART1: Modem status register |
| FFA5 | LSR1 | UART1: Line status register |
| FFA4 | MCR1 | UART1: Modem control register |
| FFA3 | FCRL1 | UART1: Flow control register |
| FFA2 | LCR1 | UART1: Line control registers |
| FFA1 | TDR1 | UART1: Transmitter data registers |
| FFA0 | RDR1 | UART1: Receiver data registers |
| FF9F | PPADR | P-Port address register |
| FF9E | PPDAT | P-Port data register |
| FF9D | PPCTL | P-Port control register |
| FF9C | PPSTA | P-Port status register |
| FF9B | PPIMSK | P-Port interrupt mask register |
| FF9A | PPMCR | P-Port mode control register |
| FF99 | PPCNT | P-Port counter register |
| ↑ | RESERVED | |
| FF94 | OEPINT | Output endpoint interrupt register |
| FF93 | IEPINT | Input endpoint interrupt register |
| FF92 | VECINT | Vector interrupt register |
| FF91 | GLOBCTL | Global control register |
| FF90 | ROMS | ROM shadow configuration register |
| ↑ | RESERVED | |
| FF83 | OEPBCNT_0 | Output endpoint_0: Byte count register |
| FF82 | OEPCNFG_0 | Output endpoint_0: Configuration register |
| FF81 | IEPBCNT_0 | Input endpoint_0: Byte count register |
| FF80 | IEPCNFG_0 | Input endpoint_0: Configuration register |

4.3 Endpoint Descriptor Block (EDB-1 to EDB-7)

Data transfers between the USB, the MCU and external devices are defined by an endpoint descriptor block (EDB). Eight input- and eight output-EDBs are provided. With the exception of EDB-0 (I/O endpoint_0), all EDBs are located in SRAM as per Table 4–4. Each EDB contains information describing the X and Y buffers. In addition, each EDB provides general status information.

Table 4–5 illustrates the EDB entries for EDB-1 to EDB-7. EDB-0 registers are described separately.

Table 4–4. EDB Memory Locations

| ADDRESS | REGISTER | DESCRIPTION |
|-------------------|------------------------|--|
| FF78 | IEPCNF_7 | Input endpoint_7: Configuration |
| FF70 | IEPCNF_6 | Input endpoint_6: Configuration |
| FF68 | IEPCNF_5 | Input endpoint_5: Configuration |
| FF60 | IEPCNF_4 | Input endpoint_4: Configuration |
| FF58 | IEPCNF_3 | Input endpoint_3: Configuration |
| FF50 | IEPCNF_2 | Input endpoint_2: Configuration |
| FF48 | IEPCNF_1 | Input endpoint_1: Configuration |
| FF47 ↑ FF40 | RESERVED | |
| FF38 | OEPCNF_7 | Output endpoint_7: Configuration |
| FF30 | OEPCNF_6 | Output endpoint_6: Configuration |
| FF28 | OEPCNF_5 | Output endpoint_5: Configuration |
| FF20 | OEPCNF_4 | Output endpoint_4: Configuration |
| FF18 | OEPCNF_3 | Output endpoint_3: Configuration |
| FF10 | OEPCNF_2 | Output endpoint_2: Configuration |
| FF08 | OEPCNF_1 | Output endpoint_1: Configuration |
| FF07 ↑ FF00 | (8-bytes) | Setup packet block |
| FEFF ↑ FEF8 | (8-bytes) | Input endpoint_0 buffer |
| FEF7 ↑ FEF0 | (8-bytes) | Output endpoint_0 buffer |
| FEEF ↑ F800 | TOPBUFF STABUFF | Top of buffer space Buffer space Start of buffer space |

Table 4–5. EDB Entries in RAM (n = 1 to 7)

| OFFSET | ENTRY NAME | DESCRIPTION |
|--------|------------|---------------------------------------|
| 07 | EPSIZXY_n | I/O endpoint_n: X/Y-buffer size |
| 06 | EPBCTY_n | I/O endpoint_n: Y-byte count |
| 05 | EPBBAY_n | I/O endpoint_n: Y-buffer base address |
| 04 | SPARE | Not used |
| 03 | SPARE | Not used |
| 02 | EPBCTX_n | I/O endpoint_n: X-byte count |
| 01 | EPBBAX_n | I/O endpoint_n: X-buffer base address |
| 00 | EPCNF_n | I/O endpoint_n: Configuration |

4.3.1 OEPCNF_n: Output Endpoint Configuration (n = 1 to 7)

| | | | | | | | |
|------|-----|--------|------|-------|-------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UBME | ISO | TOGGLE | DBUF | STALL | USBIE | RSV | RSV |
| R/W | R/W | R/W | R/W | R/W | R/W | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 1–0 | RSV | x | Reserved = 0 |
| 2 | USBIE | x | USB interrupt enable on transaction completion. Set/cleared by the MCU USBIE = 0 No interrupt USBIE = 1 Interrupt on transaction completion |
| 3 | STALL | 0 | USB stall condition indication. Set/cleared by the MCU STALL = 0 No stall STALL = 1 USB stall condition. If set by the MCU, a STALL handshake is initiated and the bit is cleared by the MCU. |
| 4 | DBUF | x | Double buffer enable. Set/cleared by the MCU. DBUF = 0 Primary buffer only (X-buffer only) DBUF = 1 Toggle bit selects buffer |
| 5 | TOGGLE | x | USB toggle bit. This bit reflects the toggle sequence bit of DATA0, DATA1 |
| 6 | ISO | x | ISO = 0 Non-isochronous transfer. This bit must be cleared by the MCU, because only non-isochronous transfer is supported. |
| 7 | UBME | x | UBM enable/disable bit. Set/cleared by the MCU UBME = 0 UBM cannot use this endpoint UBME = 1 UBM can use this endpoint |

4.3.2 OEPBBAX_n: Output Endpoint X-Buffer Base-Address (n = 1 to 7)

| | | | | | | | |
|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A ₁₀ | A ₉ | A ₈ | A ₇ | A ₆ | A ₅ | A ₄ | A ₃ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 7–0 | A[10:3] | x | A[10:3] of X-buffer base address (padded with 3-LSB of zeros for a total of 11-bits). This value is set by the MCU. The UBM or DMA uses this value as the start-address of a given transaction. Note that the UBM or DMA does <i>not</i> change this value at the end of a transaction. |

4.3.3 OEPBCTX_n: Output Endpoint X-Byte Count (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | C ₆ | C ₅ | C ₄ | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 6–0 | C[6:0] | x | X-Buffer Byte count: 000.0000b Count = 0 000.0001b Count = 1 byte : : 011.1111b Count = 63 bytes 100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | NAK | x | NAK = 0 No valid data in buffer. Ready for host-OUT request NAK = 1 Buffer contains a valid packet from host (gives NAK response to host-OUT request). |

4.3.4 OEPBBAY_n: Output Endpoint Y-Buffer Base-Address (n = 1 to 7)

| | | | | | | | |
|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A ₁₀ | A ₉ | A ₈ | A ₇ | A ₆ | A ₅ | A ₄ | A ₃ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|--|
| 7–0 | A[10:3] | x | A[10:3] of Y-buffer base address (padded with 3-LSB of zeros for a total of 11-bits). This value is set by the MCU. UBM or DMA uses this value as the start-address of a given transaction. Furthermore, UBM or DMA does <i>not</i> change this value at the end of a transaction. |

4.3.5 OEPBCTY_n: Output Endpoint Y-Byte Count (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | C ₆ | C ₅ | C ₄ | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 6–0 | C[6:0] | x | Y-Byte count: 000.0000b Count = 0 000.0001b Count = 1 byte : : 011.1111b Count = 63 bytes 100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | NAK | x | NAK = 0 No valid data in buffer. Ready for host-OUT request NAK = 1 Buffer contains a valid packet from host (gives NAK response to host-OUT request). |

4.3.6 OEPSIZXY_n: Output Endpoint X/Y-Buffer Size (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | S ₆ | S ₅ | S ₄ | S ₃ | S ₂ | S ₁ | S ₀ |
| R/O | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 6–0 | S[6:0] | x | X and Y-buffer size: 0000.0000b Count = 0 0000.0001b Count = 1 byte : : 0011.1111b Count = 63 bytes 0100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | RSV | x | Reserved = 0 |

4.3.7 IEPCNF_n: Input Endpoint Configuration (n = 1 to 7)

| | | | | | | | |
|------|-------|--------|------|-------|-------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UBME | ISO=0 | TOGGLE | DBUF | STALL | USBIE | RSV | RSV |
| R/W | R/W | R/W | R/W | R/W | R/W | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 1–0 | RSV | x | Reserved = 0 |
| 2 | USBIE | x | USB interrupt enable on transaction completion USBIE = 0 No interrupt USBIE = 1 Interrupt on transaction completion |
| 3 | STALL | 0 | USB stall condition indication. Set by the UBM but can be set/cleared by the MCU STALL = 0 No stall STALL = 1 USB stall condition. If set by the MCU, a STALL handshake is initiated and the bit is cleared by the MCU. |
| 4 | DBUF | x | Double buffer enable. Set/cleared by the MCU DBUF = 0 Primary buffer only (X-buffer only) DBUF = 1 Toggle bit selects buffer |
| 5 | TOGGLE | x | USB toggle bit. This bit reflects the toggle sequence bit of DATA0, DATA1. |
| 6 | ISO | x | ISO = 0 Non-isochronous transfer. This bit must be cleared by the MCU, because only non-isochronous transfer is supported. |
| 7 | UBME | x | UBM enable/disable bit. Set/cleared by the MCU UBME = 0 UBM cannot use this endpoint. UBME = 1 UBM can use this endpoint. |

4.3.8 IEPBBAX_n: Input Endpoint X-Buffer Base-Address (n = 1 to 7)

| | | | | | | | |
|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A ₁₀ | A ₉ | A ₈ | A ₇ | A ₆ | A ₅ | A ₄ | A ₃ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 7–0 | A[10:3] | x | A[10:3] of X-buffer base address (padded with 3-LSB of zeros for a total of 11-bits). This value is set by the MCU. The UBM or DMA uses this value as the start-address of a given transaction. Note that the UBM or DMA does <i>not</i> change this value at the end of a transaction. |

4.3.9 IEPBCTX_n: Input Endpoint X-Byte Count (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | C ₆ | C ₅ | C ₄ | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 6–0 | C[6:0] | x | X-Buffer Byte count: 000.0000b Count = 0 000.0001b Count = 1 byte : : 011.1111b Count = 63 bytes 100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | NAK | x | NAK = 0 Valid data in buffer. Ready for host-IN request NAK = 1 Buffer is empty (gives NAK response to host-IN request). |

4.3.10 IEPBBAY_n: Input Endpoint Y-Buffer Base-Address (n = 1 to 7)

| | | | | | | | |
|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A ₁₀ | A ₉ | A ₈ | A ₇ | A ₆ | A ₅ | A ₄ | A ₃ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 7–0 | A[10:3] | x | A[10:3] of Y-buffer base address (padded with 3-LSB of zeros for a total of 11 bits). This value is set by the MCU. UBM or DMA uses this value as the start-address of a given transaction. Note that the UBM or DMA does <i>not</i> change this value at the end of a transaction. |

4.3.11 IEPBCTY_n: Input Endpoint Y-Byte Count (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | C ₆ | C ₅ | C ₄ | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 6–0 | C[6:0] | x | Y-Byte count: 000.0000b Count = 0 000.0001b Count = 1 byte : : 011.1111b Count = 63 bytes 100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | NAK | x | NAK = 0 Valid data in buffer. Ready for host-IN request NAK = 1 Buffer is empty (gives NAK response to host-IN request). |

4.3.12 IEPSIZXY_n: Input Endpoint X/Y-Buffer Size (n = 1 to 7)

| | | | | | | | |
|-----|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | S ₆ | S ₅ | S ₄ | S ₃ | S ₂ | S ₁ | S ₀ |
| R/O | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 6–0 | S[6:0] | x | X and Y-buffer size: 0000.0000b Count = 0 0000.0001b Count = 1 byte : : 0011.1111b Count = 63 bytes 0100.0000b Count = 64 bytes Any value ≥ 100.0001b may result in unpredictable results. |
| 7 | RSV | x | Reserved = 0 |

4.4 Endpoint-0 Descriptor Registers

Unlike registers EDB-1 to EDB-7, which are defined as memory entries in SRAM, endpoint_0 is described by a set of 4 registers (two for output and two for input). The registers and their respective addresses, used for EDB-0 description, are defined in Table 4–6. EDB-0 has no *Base-Address-Register*, since these addresses are hardwired into FEF8 and FEF0. Note that the bit positions have been preserved to provide consistency with EDB-n (n = 1 to 7).

Table 4–6. Input/Output EDB-0 Registers

| ADDRESS | REGISTER NAME | DESCRIPTION | BASE ADDRESS |
|--------------|------------------------|---|--------------|
| FF83 FF82 | OEPBCNT_0 OEPCNFG_0 | Output endpoint_0: Byte count register Output endpoint_0: Configuration register | FEF0 |
| FF81 FF80 | IEPBCNT_0 IEPCNFG_0 | Input endpoint_0: Byte count register Input endpoint_0: Configuration register | FEF8 |

4.4.1 IEPCNFG_0: Input Endpoint-0 Configuration Register

| | | | | | | | |
|------|-----|--------|-----|-------|-------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UBME | RSV | TOGGLE | RSV | STALL | USBIE | RSV | RSV |
| R/W | R/O | R/O | R/O | R/W | R/W | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 1–0 | RSV | 0 | Reserved = 0 |
| 2 | USBIE | 0 | USB interrupt enable on transaction completion. Set/cleared by the MCU USBIE = 0 No interrupt USBIE = 1 Interrupt on transaction completion |
| 3 | STALL | 0 | USB stall condition indication. Set/cleared by the MCU STALL = 0 No stall STALL = 1 USB stall condition. If set by the MCU, a STALL handshake is initiated and the bit is cleared automatically by the next setup transaction. |
| 4 | RSV | 0 | Reserved = 0 |
| 5 | TOGGLE | 0 | USB toggle bit. This bit reflects the toggle sequence bit of DATA0, DATA1. |
| 6 | RSV | 0 | Reserved = 0 |
| 7 | UBME | 0 | UBM enable/disable bit. Set/cleared by the MCU UBME = 0 UBM cannot use this endpoint. UBME = 1 UBM can use this endpoint. |

4.4.2 IEPBCNT_0: Input Endpoint-0 Byte Count Register

| | | | | | | | |
|-----|-----|-----|-----|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | RSV | RSV | RSV | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/O | R/O | R/O | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 3–0 | C[3:0] | 0h | Byte Count 0000b Count = 0 : : 0111b Count = 7 1000b Count = 8 1001b to 1111b are reserved. (If used, they default to 8.) |
| 6–4 | RSV | 0 | Reserved = 0 |
| 7 | NAK | 1 | NAK = 0 Buffer contains a valid packet for host-IN transaction. NAK = 1 Buffer is empty (gives NAK response to host-IN request). |

4.4.3 OEPCNFG_0: Output Endpoint-0 Configuration Register

| | | | | | | | |
|------|-----|-------|-----|-------|-------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UBME | RSV | TOGLE | RSV | STALL | USBIE | RSV | RSV |
| R/W | R/O | R/O | R/O | R/W | R/W | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 1–0 | RSV | 0 | Reserved = 0 |
| 2 | USBIE | 0 | USB interrupt enable on transaction completion. Set/cleared by the MCU USBIE = 0 No interrupt USBIE = 1 Interrupt on transaction completion |
| 3 | STALL | 0 | USB stall condition indication. Set/cleared by the MCU STALL = 0 No stall STALL = 1 USB stall condition. If set by the MCU, a STALL handshake is initiated and the bit is cleared automatically. |
| 4 | RSV | 0 | Reserved = 0 |
| 5 | TOGGLE | 0 | USB toggle bit. This bit reflects the toggle sequence bit of DATA0, DATA1. |
| 6 | RSV | 0 | Reserved = 0 |
| 7 | UBME | 0 | UBM enable/disable bit. Set/cleared by the MCU UBME = 0 UBM cannot use this endpoint. UBME = 1 UBM can use this endpoint. |

4.4.4 OEPBCNT_0: Output Endpoint X-Byte Count Register

| | | | | | | | |
|-----|-----|-----|-----|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | RSV | RSV | RSV | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/O | R/O | R/O | R/O | R/O | R/O | R/W O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 3–0 | C[3:0] | 0h | Byte count: 0000b Count = 0 : : 1111b Count = 7 1000b Count = 8 1001b to 1111b are reserved |
| 6–4 | RSV | 0 | Reserved = 0 |
| 7 | NAK | x | NAK = 0 No valid data in buffer. Ready for host-OUT request NAK = 1 Buffer contains a valid packet from host (gives NAK response to host-OUT request). |

5 USB Registers

5.1 FUNADR: Function Address Register

This register contains the device function address.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | FA6 | FA5 | FA4 | FA3 | FA2 | FA1 | FA0 |
| R/O | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|--|
| 6-0 | FA[6:0] | 00 | These bits define the current device address assigned to the function. The MCU writes a value to this register as a result of the <i>SET-ADDRESS</i> host command. |
| 7 | RSV | 0 | Reserved = 0 |

5.2 USBSTA: USB Status Register

All bits in this register are set by the hardware and are cleared by the MCU when writing a 1 to the proper bit location (writing a 0 has no effect). In addition, each bit can generate an interrupt if its corresponding mask bit is set (R/C notation indicates read and clear only by the MCU).

| | | | | | | | |
|------|------|------|-------|-------|-------|--------|-------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSTR | SUSR | RESR | UR2RI | UR1RI | SETUP | WAKEUP | STPOW |
| R/C | R/C | R/C | R/C | R/C | R/C | R/C | R/C |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 0 | STPOW | 0 | SETUP overwrite bit. Set by hardware when setup packet is received while there is already a packet in the setup buffer STPOW = 0 MCU can clear this bit by writing a 1. Writing 0 has no effect. STPOW = 1 SETUP overwrite |
| 1 | WAKEUP | 0 | Remote wake-up bit WAKEUP = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. WAKEUP = 1 Remote wake-up request from WAKEUP pin |
| 2 | SETUP | 0 | SETUP transaction received bit. As long as SETUP is 1, IN and OUT on endpoint_0 will be NAKed, regardless of their real NAK bits value. SETUP = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. SETUP = 1 SETUP transaction received |
| 3 | UR1RI | 0 | UART1 RI status bit UR1RI = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. UR1RI = 1 Ring detected |
| 4 | UR2RI | 0 | UART2 RI status bit UR2RI = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. UR2RI = 1 Ring detected |
| 5 | RESR | 0 | Function resume request bit RESR = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. RESR = 1 Function resume is detected. |
| 6 | SUSR | 0 | Function suspended request bit. This bit is set in response to a global or selective suspend condition. FSUSP = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. FSUSP = 1 Function suspend is detected. |
| 7 | RSTR | 0 | Function reset request bit. This bit is set in response to host initiating a port reset. This bit will not be affected by USB function reset. FRST = 0 The MCU can clear this bit by writing a 1. Writing 0 has no effect. FRST = 1 Function reset is detected. |

5.3 OEPCNFG_0: Output Endpoint-0 Configuration Register

| | | | | | | | |
|------|-----|--------|-----|-------|-------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| UBME | RSV | TOGGLE | RSV | STALL | USBIE | RSV | RSV |
| R/W | R/O | R/O | R/O | R/W | R/W | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 1–0 | RSV | 0 | Reserved = 0 |
| 2 | USBIE | 0 | USB interrupt enable on transaction completion. Set/cleared by the MCU USBIE = 0 No interrupt USBIE = 1 Interrupt on transaction completion |
| 3 | STALL | 0 | USB stall condition indication. Set/cleared by the MCU STALL = 0 No stall STALL = 1 USB stall condition. If set by the MCU, a STALL handshake is initiated and the bit is cleared automatically. |
| 4 | RSV | 0 | Reserved = 0 |
| 5 | TOGGLE | 0 | USB toggle bit. This bit reflects the toggle sequence bit of DATA0, DATA1 |
| 6 | RSV | 0 | Reserved = 0 |
| 7 | UBME | 0 | UBM enable/disable bit. Set/cleared by the MCU UBME = 0 UBM cannot use this endpoint. UBME = 1 UBM can use this endpoint. |

5.4 OEPBCNT_0: Output Endpoint-0 Byte Count Register

| | | | | | | | |
|-----|-----|-----|-----|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NAK | RSV | RSV | RSV | C ₃ | C ₂ | C ₁ | C ₀ |
| R/W | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 3–0 | C[3:0] | 0h | Byte Count 0000b Count = 0 : : 0111b Count = 7 1000b Count = 8 1001b to 1111b are reserved. |
| 6–4 | RSV | 0 | Reserved = 0 |
| 7 | NAK | 1 | NAK = 0 No valid data in buffer. Ready for host-OUT request NAK = 1 Buffer contains a valid packet from host (gives NAK response to host-OUT request). |

5.5 USBMSK: USB Interrupt Mask Register

| | | | | | | | |
|------|------|------|-------|-------|-------|--------|-------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSTR | SUSR | RESR | UR2RI | UR1RI | SETUP | WAKEUP | STPOW |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 0 | STPOW | 0 | SETUP overwrite interrupt-enable bit STPOW = 0 STPOW interrupt disabled STPOW = 1 STPOW interrupt enabled |
| 1 | WAKEUP | 0 | Remote wake-up interrupt enable bit WAKEUP = 0 WAKEUP interrupt disable WAKEUP = 1 WAKEUP interrupt enable |
| 2 | SETUP | 0 | SETUP interrupt enable bit SETUP = 0 SETUP interrupt disabled SETUP = 1 SETUP interrupt enabled |
| 3 | UR1RI | 0 | UART 1 RI interrupt enable bit UR1RI = 0 UR1RI interrupt disabled UR1RI = 1 UR1RI interrupt enabled |
| 4 | UR2RI | 0 | UART 2 RI interrupt enable bit UR1RI = 0 UR1RI interrupt disabled UR1RI = 1 UR1RI interrupt enabled |
| 5 | RESR | 0 | Function resume interrupt enable bit RESR = 0 Function resume interrupt disabled RESR = 1 Function resume interrupt enabled |
| 6 | SUSR | 0 | Function suspend interrupt enable FSUSP = 0 Function suspend interrupt disabled FSUSP = 1 Function suspend interrupt enabled |
| 7 | RSTR | 0 | Function reset interrupt bit. This bit is not affected by USB function reset. FRST = 0 Function reset interrupt disabled FRST = 1 Function reset interrupt enabled |

5.6 USBCTL: USB Control Register

Unlike the rest of the registers, this register is cleared by the power-up-reset signal only. The USB reset cannot reset this register (see Figure 5–1).

| | | | | | | | |
|------|------|------|-------|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CONT | U1/2 | RWUP | FRSTE | RSV | B/S | SIR | DIR |
| R/W | R/O | R/W | R/W | R/O | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|-------|-------|---|
| 0 | DIR | 0 | As a response to a setup packet, the MCU decodes the request and sets/clears this bit to reflect the data transfer direction. DIR = 0 USB data-OUT transaction (from host to TUSB5152). DIR = 1 USB data-IN transaction (from TUSB5152 to host). |
| 1 | SIR | 0 | SETUP interrupt-status bit. This bit is controlled by the MCU to indicate to the hardware when the SETUP interrupt is being served. SIR = 0 SETUP interrupt is <i>not</i> served. The MCU clears this bit before exiting the SETUP interrupt routine. SIR = 1 SETUP interrupt is in progress. The MCU sets this bit when servicing the SETUP interrupt. |
| 2 | BS | 0 | BS = 0 Bus-powered hub BS = 1 Self-powered hub |
| 3 | RSV | 0 | Reserved |
| 4 | FRSTE | 1 | Function reset-connection bit. This bit connects/disconnects the USB function reset to/from the MCU reset. FRSTE = 0 Function reset is not connected to MCU reset. FRSTE = 1 Function reset is connected to MCU reset. |
| 5 | RWUP | 0 | Device remote wake-up request. This bit is set by the MCU and is cleared automatically. RWUP = 0 Writing a 0 to this bit has no effect. RWUP = 1 When MCU writes a 1, a remote wake-up pulse is generated. |
| 6 | U1/2 | 0 | USB hub version U1/2 = 0 This is a USB1.x hub. U1/2 = 1 This is a USB2.x hub. |
| 7 | CONT | 0 | Connect/disconnect bit CONT = 0 Upstream port is disconnected. Pullup disabled CONT = 1 Upstream port is connected. Pullup enabled |

5.7 HUBCNF1: Hub-Configuration-1 Register

This register is cleared by the power-up-reset signal only. The USB reset cannot reset this register.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| P4A | P4E | P3A | P3E | P2A | P2E | P1A | P1E |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | P1E | 0 | Hub port-1. Enable/disable control bit P1E = 0 Port-1 is disabled. P1E = 1 Port-1 is enabled. |
| 1 | P1A | 0 | Hub port-1. Permanent-attachment control bit P1A = 0 Port-1 is connected to a removable function. P1A = 1 Port-1 is connected to a permanent-attachment function. |
| 2 | P2E | 0 | Hub port-2. Enable/disable control bit P2E = 0 Port-2 is disabled. P2E = 1 Port-2 is enabled. |
| 3 | P2A | 0 | Hub port-2. Permanent-attachment control bit P2A = 0 Port-2 is connected to a removable function. P2A = 1 Port-2 is connected to a permanent-attachment function. |
| 4 | P3E | 0 | Hub port-3. Enable/disable control bit P3E = 0 Port-3 is disabled. P3E = 1 Port-3 is enabled. |
| 5 | P3A | 0 | Hub port-3. Permanent-attachment control bit P3A = 0 Port-3 is connected to a removable function. P3A = 1 Port-3 is connected to a permanent-attachment function. |
| 6 | P4E | 0 | Hub port-4. Enable/disable control bit P4E = 0 Port-4 is disabled. P4E = 1 Port-4 is enabled. |
| 7 | P4A | 0 | Hub port-4. Permanent-attachment control bit P4A = 0 Port-4 is connected to a removable function. P4A = 1 Port-4 is connected to a permanent-attachment function. |

5.8 HUBCNF2: Hub-Configuration-2 Register

This register is cleared by the power-up-reset signal only. The USB-reset cannot reset this register.

| | | | | | | | |
|-------|-----|------|------|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWRSW | RSV | P6.1 | P6.0 | RSV | RSV | P5A | P5E |
| R/W | R/W | R/W | R/W | R/O | R/O | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 0 | P5E | 0 | Hub port-5. Enable/disable control bit P5E = 0 Port-5 is disabled. P5E = 1 Port-5 is enabled. |
| 1 | P5A | 0 | Hub port-5. Permanent-attachment control bit P5A = 0 > Port-5 is connected to a removable function. P5A = 1 Port-5 is connected to a permanent-attachment function. |
| 3–2 | RSV | 0 | Reserved = 0 |
| 5–4 | P6[1:0] | 00b | Hub port-6. Embedded-function control field 00b = Port-6 is disabled (does not exist). 01b = Port-6 is permanently attached. 10b = Port-6 is connected to a removable function, but is <i>not</i> attached. 11b = Port-6 is connected to a removable function, and is attached. |
| 6 | RSV | 0 | Reserved = 0 |
| 7 | PWRSW | 0 | Power switch control bit PWRSW = 0 Not available PWRSW = 1 Available |

5.9 HUBPOTG: Hub Power-On to Power-Good Descriptor Register

This register is cleared by the power-up-reset signal only. The USB reset cannot reset this register.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7 | D[7:0] | 0 | Offset-5 in hub descriptor table Time (in 2 ms intervals) from the time the power-on sequence begins on a port until power is good on that port (11.15.2.1 Hub Descriptor in the USB 1.1 specification) |

5.10 HUBPIDL: Hub-PID Register (Low-Byte)

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|------------------------|
| 7 | D[7:0] | 0 | Hub PID low-byte value |

5.11 HUBPIDH: Hub-PID Register (High-Byte)

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|-------------------------|
| 7 | D[7:0] | 0 | Hub PID high-byte value |

5.12 HUBVIDL: Hub-VID Register (Low-Byte)

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|------------------------|
| 7 | D[7:0] | 0 | Hub VID low-byte value |

5.13 HUBVIDH: Hub-VID Register (High-Byte)

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|-------------------------|
| 7 | D[7:0] | 0 | Hub VID high-byte value |

5.14 Function Reset and Power-Up Reset Interconnect

Figure 5–1 represents the logical connection of the USB-function-reset ($\overline{\text{USB}\overline{\text{R}}}$) and power-up-reset ($\overline{\text{RSTI}}$) pins. The internal $\overline{\text{RESET}}$ signal is generated from the $\overline{\text{RSTI}}$ pin ($\overline{\text{PURS}}$ signal) or from the USB-reset ($\overline{\text{USB}\overline{\text{R}}}$ signal). The $\overline{\text{USB}\overline{\text{R}}}$ can be enabled or disabled by the FRSTE bit in the USBCTL register (on power-up, FRSTE = 0). The internal $\overline{\text{RESET}}$ is used to reset all registers and logic with the exception of the USBCTL and GLOBCTL registers, which are cleared by the $\overline{\text{PURS}}$ signal only. As shown, the $\overline{\text{RESET}}$ /RESET is connected to the MUX, which controls the RSTO output polarity (the MUX is controlled by the GLOBCTL register). In the development mode, the RSTO signal is used to reset the external microcontroller or other devices.

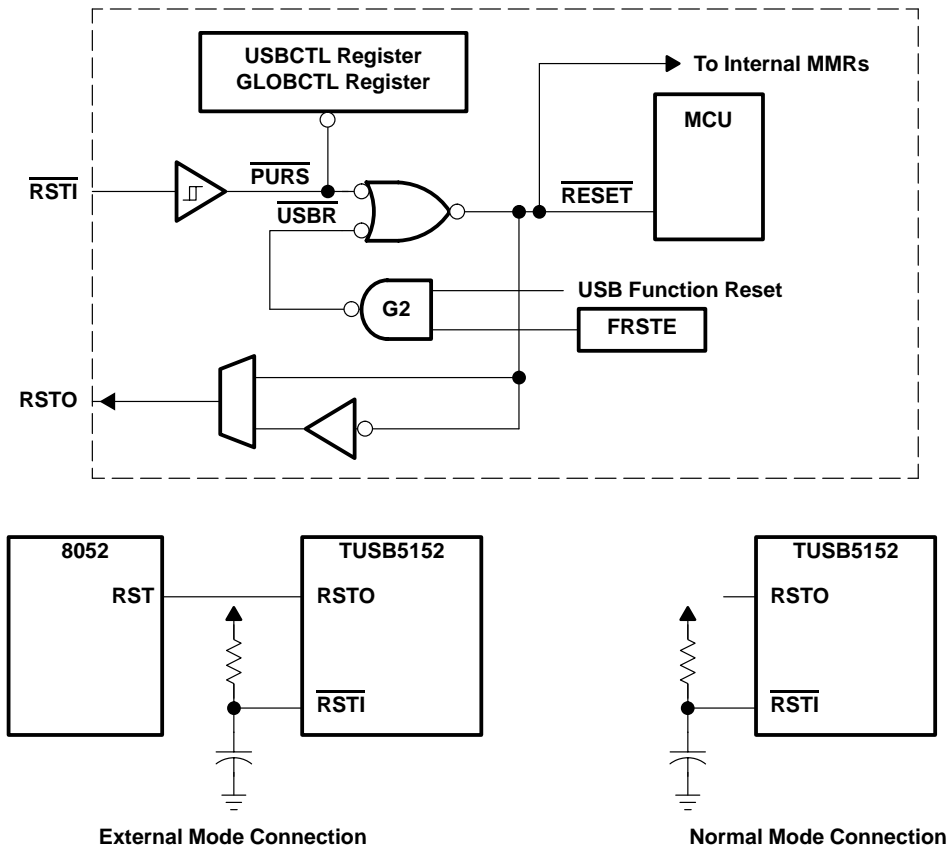


Figure 5–1. Reset Diagram

5.15 Pullup Resistor Connect/Disconnect

TUSB5152 numeration can be activated by the MCU (there is no need to disconnect the cable physically). Figure 5–2 represents the implementation of the TUSB5152 *Connect* and *Disconnect* from a USB upstream port. When $CONT = 1$ in the USBCTL register, the CMOS driver sources V_{DD} to the pullup resistor (PUR pin) presenting a normal connect condition to the USB hub (high speed). When $CONT = 0$, the PUR pin is driven low. In this state, the 1.5-k Ω resistor is connected to GND, resulting in the device *disconnection* state. The PUR driver is a CMOS driver that can provide $(V_{DD}-0.1)V$ minimum at 8-mA source current.

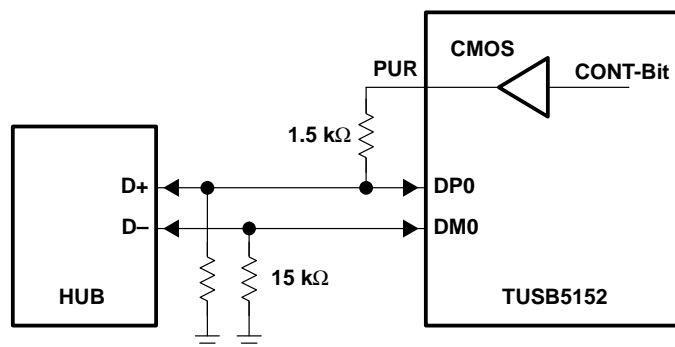


Figure 5–2. Pullup Resistor Connect/Disconnect Circuit

6 DMA Controller

Table 6–1 outlines the five DMA channels and their associated transfer directions. Four channels are provided for data transfer between the host and the UARTs. DMA-5 is provided for half-duplex parallel-port transfer (the transfer direction is defined by the DIR bit in the PPMCR register).

Table 6–1. DMA Channel Allocation

| DMA CHANNEL | TRANSFER DIRECTION | COMMENTS |
|-------------|---------------------|---|
| DMA-1 | Host to UART-1 | DMA writes to UART-1 TDR register |
| DMA-2 | Host to UART-2 | DMA writes to UART-2 TDR register |
| DMA-3 | UART-1 to host | DMA reads from UART-1 RDR register |
| DMA-4 | UART-2 to host | DMA reads from UART-2 RDR register |
| DMA-5 | Host to/from P-port | DMA reads/writes from/to PPDAT register. Direction is defined by DIR-bit in PPMCR register. |

6.1 DMA Controller Registers

Each DMA channel can point to one of seven EDBs (EDB[7:1]) and transfer data to/from a UART channel or the parallel port. The DMA can move data from a given output-endpoint buffer (defined by EDB) to the destination port. Similarly, the DMA can move data from a port to a given input-endpoint buffer. Two modes of DMA transfers are supported: burst and continuous.

In burst (CNT = 0) mode the DMA stops at the end of a block-data transfer (or if an error condition occurred) and interrupts the MCU. It is the responsibility of the MCU to update the X/Y bit and the NAK bit in the EDB.

In continuous (CNT = 1) mode, at the end of a block transfer the DMA updates the byte-count and NAK-bit in the EDB when receiving. In addition, it uses the X/Y-bit to switch automatically, without interrupting the MCU (the X/Y bit toggle is performed by the UBM). The DMA stops only when a time-out or error condition occurs. When the DMA is transmitting (from the X/Y buffer) it continues alternating between X/Y buffers until it detects a byte-count smaller than the buffer size (buffer size is typically 64 bytes). At that point it completes the transfer, then stops.

6.1.1 DMACDR[2:1]: DMA Channel Definition Register (2 UART Transmit Channels)

These registers are used to define the EDB number that the DMA uses for data transfer to the UARTs. In addition, these registers define the data transfer direction and select X or Y as the transaction buffer.

| | | | | | | | |
|-----|-----|-----|-----|------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EN | INE | CNT | XY | XY_WR_PROT | E ₂ | E ₁ | E ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------------|-------|---|
| 2–0 | E[2:0] | 0 | Endpoint descriptor pointer. This field points to a set of EDB registers that is to be used for a given transfer. |
| 3 | XY_WR_PROT | 1 | XY protection bit. This bit always returns 1 when read. If this bit is 1 during a write, XY (bit 4) is not affected. If this bit is 0 during a write, the value of XY is loaded from MCU bit 4. |
| 4 | XY | 0 | X/Y buffer select bit. Valid only when CNT = 0 XY = 0 Next buffer to transmit/receive is the X buffer XY = 1 Next buffer to transmit/receive is the Y buffer |
| 5 | CNT | 0 | DMA continuous transfer control bit. This bit defines the mode of the DMA transfer. CNT = 0 Burst Mode: DMA stops the transfer when the byte count is zero or when a partial packet has been received (byte count < 64). At the end of transfer, the high-to-low transition of EN interrupts the MCU (if enabled). In this mode, the X/Y bit is set by the MCU to define the current buffer (X or Y). Single buffering is required in this mode. CNT = 1 Continuous Mode: In this mode, the DMA and UBM alternate between the X and Y buffers. The DMA sets the X/Y bit and the UBM uses it for the transfer. The DMA alternates between the X/Y buffers and continues transmitting (from X/Y buffer) without MCU intervention. The DMA terminates, and interrupts the MCU, under the following conditions: 1. When the UBM byte count < buffer size (in EDB), the DMA transfers the partial packet and interrupts the MCU on completion. 2. Transaction timer expires. The DMA interrupts the MCU. |
| 6 | INE | 0 | DMA interrupt enable/disable bit. This bit is used to enable/disable the interrupt on transfer completion. INE = 0 Interrupt is disabled. In addition, PPKT and TXFT do not clear the EN bit and the DMAC is <i>not</i> disabled. INE = 1 Enables the EN interrupt. When this bit is set, the DMA interrupts the MCU on a 1-to-0 transition of the EN bit. (When transfer is completed, EN = 0) |
| 7 | EN | 0 | DMA channel enable bit. The MCU sets this bit to start the DMA transfer. When the transfer completes, or when it is terminated due to error, this bit is cleared. The 1 to 0 transition of this bit generates an interrupt (if interrupt is enabled). EN = 0 DMA is halted. The DMA is halted when the byte count reaches zero or when transaction time-out occurs. When halted, the DMA updates the byte count, sets NAK = 0 in OEDB, and interrupts the MCU (if INE = 1). EN = 1 Setting this bit starts the DMA transfer. |

6.1.2 DMACSR[2:1]: DMA Control and Status Register (2 UART Transmit Channels)

This register is used to define the transaction time-out value. In addition, it contains a completion code that reports any errors or a time-out condition. Note that the XY bit sets XYO/XYI in the DMACDR5 register.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | RSV | RSV | XY | RSV | RSV | RSV | PPKT |
| R/O | R/O | R/O | R/W | R/O | R/O | R/O | R/C |

| BITS | NAME | RESET | FUNCTION |
|------|------|-------|---|
| 0 | PPKT | 0 | Partial packet condition bit. This bit is set by the DMA and cleared by the MCU (see Table 6–2). PPKT = 0 No partial-packet condition PPKT = 1 Partial-packet condition detected. When IEN = 0, this bit does not clear the EN bit in DMACDR; therefore, the DMAC stays enabled, ready for the next transaction. Clears when MCU writes a 1. Writing a 0 has no effect. |
| 3–1 | RSV | 0 | Reserved = 0 |
| 4 | XY | 0 | DMACSR[1] = X/Y bit DMA IN for IEEE 1284 DMACSR[2] = X/Y bit DMA OUT for IEEE 1284 |
| 7–5 | RSV | 0 | Reserved = 0 |

Table 6–2. DMA OUT-Termination Condition

| OUT-TERMINATION | TXFT | PPKT | COMMENTS |
|---------------------|------|------|---|
| UART partial packet | 0 | 1 | This condition occurs when the host sends a partial packet. |

6.1.3 DMACDR[4:3]: DMA Channel Definition Register (2 UART Receive Channels)

These registers are used to define the EDB number that the DMA uses for data transfer to the UARTs. In addition, these registers define the data transfer direction and select X or Y as the transaction buffer.

| | | | | | | | |
|-----|-----|-----|-----|------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EN | INE | CNT | XY | XY_WR_PROT | E ₂ | E ₁ | E ₀ |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------------|-------|--|
| 2-0 | E[2:0] | 0 | Endpoint descriptor pointer. This field points to a set of EDB registers that is to be used for a given transfer. |
| 3 | XY_WR_PROT | 1 | XY protection bit. This bit always returns 1 when read. If this bit is 1 during a write, XY (bit 4) is not affected. If this bit is 0 during a write, the value of XY is loaded from MCU bit 4. |
| 4 | XY | 0 | XY buffer select bit. Valid only when CNT = 0 XY = 0 Next buffer to transmit/receive is X XY = 1 Next buffer to transmit/receive is Y |
| 5 | CNT | 0 | DMA continuous transfer control bit. This bit defines the mode of the DMA transfer. CNT = 0 Burst Mode: DMA stops the transfer when the byte count = 0 or when a receiver error occurs. At the end of transfer, the high-to-low transition of EN interrupts the MCU (if enabled). In this mode, the XY bit is set by the MCU to define the current buffer (X or Y). Single buffering is required in this mode. CNT = 1 Continuous Mode: In this mode, the DMA and UBM alternate between the X and Y buffers. The UBM sets the XY bit and the DMA uses it for the transfer. The DMA alternates between the X/Y-buffers and continues transmitting (from X/Y buffer) without MCU intervention. The DMA terminates, and interrupts the MCU, under the following conditions: 1. Transaction time-out expired: DMA updates EDB and interrupts the MCU. UBM transfers the partial packet to host. 2. UART receiver error condition: DMA updates EDB and does <i>not</i> interrupt the MCU. UBM transfers the partial packet to host. |
| 6 | INE | 0 | DMA interrupt enable/disable bit. This bit is used to enable/disable the interrupt on transfer completion. INE = 0 Interrupt is disabled. In addition, OVRUN and TXFT do not clear the EN bit and the DMAC is <i>not</i> disabled. INE = 1 Enables the EN interrupt. When this bit is set, the DMA interrupts the MCU on a 1-to-0 transition of the EN bit. (When transfer is completed, EN = 0) |
| 7 | EN | 0 | DMA channel enable bit. The MCU sets this bit to start the DMA transfer. When the transfer completes, or when it is terminated due to error, this bit is cleared. The 1-to-0 transition of this bit generates an interrupt (if interrupt is enabled). EN = 0 DMA is halted. The DMA is halted when transaction time-out occurs, or under a UART receiver-error condition. When halted, the DMA updates the byte-count and sets NAK = 0 in IEDB. If the termination is due to transaction time-out, the DMA generates an interrupt. However, if the termination is due to a UART error condition, the DMA does not generate an interrupt. (The UART generates the interrupt.) EN = 1 Setting this bit starts the DMA transfer. |

6.1.4 DMACSR[4:3]: DMA Control and Status Register (2 UART Receive Channels)

This register is used to define the transaction time-out value. In addition, this register contains a completion code that reports any errors or a time-out condition.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|------|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TEN | C4 | C3 | C2 | C1 | C0 | TXFT | RSV |
| R/W | R/W | R/W | R/W | R/W | R/W | R/C | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 0 | RSV | 0 | Reserved = 0 |
| 1 | TXFT | 0 | Transfer time-out condition bit (see Table 6–3) TXFT = 0 DMA stopped transfer without time-out TXFT = 1 DMA stopped due to transaction time-out. When IEN = 0, this bit does not clear the EN bit in DMACDR; therefore, the DMAC stays enabled, ready for the next transaction. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 6-2 | C[4:0] | 0 | This field is used to define the transaction time-out value in 1 ms increments. This value is loaded to a down counter every time a byte transfer occurs. The down counter is decremented every SOF pulse (1 ms). If the counter decrements to zero, it sets TXFT = 1 (in DMACSR register) and halts the DMA transfer. The counter starts counting only when TEN = 1 and EN = 1 (in DMACDR) and the first byte has been received (see Figure 6–1). 00000 = Do not use if TEN is 1 00001 = 1-ms time-out : : 11111 = 31-ms time-out |
| 7 | TEN | 0 | Transaction time-out counter enable/disable bit TEN = 0 Counter is disabled (does <i>not</i> time out). TEN = 1 Counter is enabled. |

Table 6–3. DMA IN-Termination Condition

| IN-TERMINATION | TXFT | COMMENTS |
|----------------|------|---|
| UART error | 0 | UART error condition detected |
| UART time-out | 1 | This condition occurs when UART receiver has no more data for host (data starvation). |

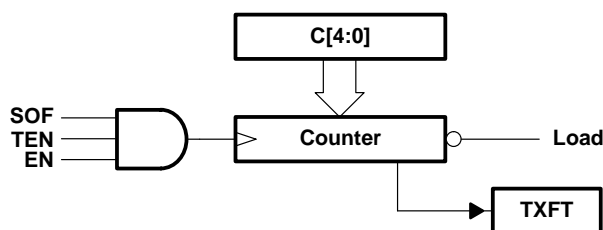


Figure 6–1. Transaction Time-Out Diagram

6.1.5 DMACDR[5]: DMA-5 Channel Definition Register (P-Port Channel)

This register is used to define the EDB number that the DMA used for data transfer to/from the P-port. In addition, this register defines the data transfer mode and selects X or Y as the transaction buffer. Note: the transfer direction is defined by the DIR bit in the PPMCR register.

| | | | | | | | |
|-----|-----|-----|-----|-----|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EN | INE | CNT | XYI | XYO | E ₂ | E ₁ | E ₀ |
| R/W | R/W | R/W | R/O | R/O | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 2–0 | E[2:0] | 0 | Endpoint descriptor pointer. This field points to a set of EDB registers that is to be used for a given transfer. |
| 3 | XYO | 0 | X/Y-output buffer select bit. This bit is set/cleared by writing to the DMACSR register, bit 4. XYO = 0 Next output buffer to transmit/receive is X. XYO = 1 Next output buffer to transmit/receive is Y. |
| 4 | XYI | 0 | X/Y-input buffer select bit. This bit is set/cleared by writing to the DMACSR register, bit 4. XYI = 0 Next input buffer to transmit/receive is X. XYI = 1 Next input buffer to transmit/receive is Y. |
| 5 | CNT_EN | 0 | This bit must be set to 1 if DMA is used. |
| 6 | INE | 0 | DMA interrupt enable/disable bit. This bit is used to enable/disable the interrupt on transfer completion. INE = 0 Interrupt is disabled. INE = 1 Enables the EN interrupt. When this bit is set, the DMA interrupts the MCU on a 1-to-0 transition of the EN bit. (When transfer is completed, EN = 0.) |
| 7 | EN | 0 | DMA channel enable. The MCU sets this bit to start the DMA transfer. When the transfer completes, or when it is terminated due to error, this bit is cleared. The 1-to-0 transition of this bit generates an interrupt (if interrupt is enabled). EN = 0 DMA is halted. 1 to 0 transition interrupts the MCU (if INE = 1). EN = 1 Setting this bit starts the DMA transfer. |

6.1.6 DMACSR[5]: DMA-5 Control and Status Register (P-Port Channel)

This register is used to define the transaction time-out value. In addition, it contains a completion code that reports any errors or a time-out condition.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|------|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TEN | C4 | C3 | C2 | C1 | C0 | TXFT | PPKT |
| R/W | R/W | R/W | R/W | R/W | R/W | R/C | R/C |

| BITS | NAME | RESET | FUNCTION |
|------|--------|-------|--|
| 0 | RSV | 0 | Reserved = 0 |
| 1 | TXFT | 0 | Transfer time-out condition bit (see Table 6–2 and Table 6–3) TXFT = 0 DMA stopped transfer without time-out TXFT = 1 DMA stopped due to transaction time-out. Clears when MCU writes a 1. Writing a 0 has no effect. |
| 6-2 | C[4:0] | 0000b | This field is used to define the transaction time-out value in 1-ms increments. This value is loaded to a down counter every time a byte transfer occurs. The down counter is decremented every SOF pulse (1 ms). If the counter decrements to zero it sets TXFT = 1 (in DMACSR register) and halts the DMA transfer. <ul style="list-style-type: none"> • For OUT transaction: Time-out is ignored. • For IN transaction: The counter starts counting only when TEN = 1 and EN = 1 (in DMACDR) and the first byte has been received by DMA (see Figure 6–1). 00000 = Do not use if TEN is 1 00001 = 1-ms time-out : : 11111 = 31-ms time-out |
| 7 | TEN | 0 | Transaction time-out counter enable/disable bit TEN = 0 Counter is disabled (does <i>not</i> time out). TEN = 1 Counter is enabled. |

6.2 Bulk Data I/O Using the EDB

The UBM (USB buffer manager) and the DMAC (DMA controller) access the EDB to fetch buffer parameters for IN and OUT transactions (IN and OUT are with respect to host). In this discussion, it is assumed that (a) the MCU initialized the EDBs, (b) DMA-continuous mode is being used, (c) double buffering is being used, and (d) the X/Y toggle is controlled by the UBM.

NOTE: The IN and OUT transfers apply to UART and P-port transactions (SPP, EPP and ECP modes).

IN Transaction (TUSB5152 to Host):

- MCU initializes the IEDB (64-byte packet, and double-buffering is used) and the following DMA registers:
 - DMACSR: Defines the transaction time-out value.
 - DMACDR: Defines the IEDB being used and the DMA mode of operation (continuous-mode). Once this register is set with EN = 1, the transfer starts.
- DMA transfers data from a device (UART or P-port) to the X-buffer. When a block of 64-bytes is transferred, the DMA updates the byte-count and sets NAK = 0 in IEDB (indicating to the UBM that the X-buffer is ready to be transferred to the host). The UBM starts X-buffer transfer to host using the byte-count value in IEDB and toggles the X/Y-bit. The DMA continues transferring data from a device to Y-buffer. At the end of the block transfer, the DMA updates the byte-count and sets NAK = 0 in IEDB (indicating to the UBM that the Y-buffer is ready to be transferred to host). The DMA continues the transfer from the device to the host, alternating between X and Y buffers without MCU intervention.
- Transfer termination: As mentioned, the DMA/UBM continues the data transfer, alternating between the X and Y buffers. Termination of the transfer can happen under the following conditions:
 - **Stop Transfer:** The host notifies the MCU (via control-end-point) to stop the transfer. Under this condition, the MCU sets EN = 0 in the DMACDR register.

- **Partial-Packet:** Device-receiver has no data to be transferred to host. Under this condition, the byte-count value is less than 64 when the transaction-timer time-out occurs. When the DMA detects this condition, it sets TXFT = 1 and OVRUN = 0, updates the byte count and NAK bit (partial-packet) in the IEDB, and interrupts the MCU. UBM transfers the partial packet to host.
- **Buffer-Overrun:** The host is busy, X and Y-buffers are full (X-NAK = 0 and Y-NAK = 0) and the DMA cannot write to these buffers. The transaction time-out stops the DMA transfer, the DMA sets TXFT = 1 and OVRUN = 1, and interrupts the MCU.
- **UART Error Condition:** When receiving from a UART, a receiver-error condition stops the DMA and sets TXFT = 1 and OVRUN = 0, but the EN-bit remains set at 1. Therefore, the DMA does not interrupt the MCU. However, the UART generates a status interrupt, notifying the MCU that an error condition has occurred.

OUT Transaction (Host to TUSB5152):

1. The MCU initializes the OEDB (64-byte packet, and double-buffering is used) and the following DMA registers:
 - **DMACSR:** Defines the transaction time-out value.
 - **DMAHDR:** Defines the OEDB being used, and the DMA mode of operation (continuous-mode). Once the EN bit is set to 1 in this register, the transfer starts.
2. The UBM transfers data from host to X-buffer. When a block of 64-bytes is transferred, the UBM updates the byte-count and sets NAK = 1 in OEDB (indicating to DMA that the X-buffer is ready to be transferred to UART/PPT). The DMA starts X-buffer transfer using the byte-count value in OEDB. The UBM continues transferring data from host to Y-buffer. At the end of the block transfer, the UBM updates the byte-count and sets NAK = 1 in OEDB (indicating to DMA that the Y-buffer is ready to be transferred to device). The DMA continues the transfer from the X/Y-buffers to the device, alternating between X and Y buffers without MCU intervention.
3. Transfer termination: As mentioned, the DMA/UBM continues the data transfer alternating between X and Y buffers. The termination of the transfer can happen under the following conditions:
 - **Stop Transfer:** The host notifies the MCU (via control-end-point) to stop the transfer. Under this condition, the MCU sets EN = 0 in the DMAHDR register.
 - **Partial-Packet:** UBM receives a partial packet from host. Under this condition, the byte-count value is less than 64 and the transaction-timer does *not* time out. When the DMA detects this condition, it transfers the partial packet to the device, sets TXFT = 0 and PPKT = 1, updates NAK = 0 in OEDB, and interrupts the MCU.
 - **Timeout:** The device is busy, X and Y-buffers are full (X-NAK = 1 and Y-NAK = 1) and the UBM cannot write to these buffers. Under these conditions the transaction-timer time-out stops the DMA transfer, sets TXFT = 1 and OVRUN = 0, and interrupts the MCU.

7 UARTs

7.1 UART Registers

Table 7–1 summarizes the UART registers. These registers are used for data I/O, control, and status information. UART setup is done by the MCU. Data transfer is typically performed by the DMAC. However, the MCU can perform data transfer without the DMA; this is useful when debugging the firmware.

Table 7–1. UART Register Summary

| REGISTER NAME | ACCESS | FUNCTION | COMMENTS |
|---------------|--------|-----------------------------------|-------------------------------------|
| RDR | R/O | UART Receiver data register | Can be accessed by MCU or DMA |
| TDR | W/O | UART Transmitter data register | Can be accessed by MCU or DMA |
| LCR | R/W | UART Line control register | |
| FCRL | R/W | UART Flow control register | |
| MCR | R/W | UART Modem control register | |
| LSR | R/O | UART Line status register | Can generate an interrupt |
| MSR | R/O | UART Modem status register | Can generate an interrupt |
| DLL | R/W | UART Divisor register (low-byte) | |
| DLH | R/W | UART Divisor register (high-byte) | |
| XON | R/W | UART Xon register | |
| XOFF | R/W | UART Xoff register | |
| MASK | R/W | UART Interrupt mask register | Can control three interrupt sources |

7.1.1 RDR[2:1]: Receiver Data Registers (2 Registers)

Each of the two receiver data registers consists of a 32-byte FIFO. Data received from the SIN pin are converted from serial to parallel format and stored in this FIFO. Data transfer from these registers to the RAM buffer is the responsibility of the DMA controller.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---------------|
| 7–0 | D[7:0] | 0 | Receiver byte |

7.1.2 TDR[2:1]: Transmitter Data Registers (2 Registers)

Each of the two transmitter data registers is double buffered. Data written to these registers are loaded into the shift-register, and shifted out on SOUT. Data transfer from the RAM buffer to these registers is the responsibility of the DMA controller.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| W/O | W/O | W/O | W/O | W/O | W/O | W/O | W/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|------------------|
| 7–0 | D[7:0] | 0 | Transmitter byte |

7.1.3 LCR[2:1]: Line Control Registers (2 Registers)

These registers control the data communication format. The word length, number of stop bits, and parity type are selected by writing the appropriate bits to the LCR.

| | | | | | | | |
|-----|-----|------|-------|------|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FEN | BRK | FPTY | EPRTY | PRTY | STP | WL1 | WL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 1–0 | WL[1:0] | 0 | Specifies the word length for transmit and receive. 00b = 5 bits 01b = 6 bits 10b = 7 bits 11b = 8 bits |
| 2 | STP | 0 | Specifies the number of stop bits for transmit and receive. STP = 0 1 stop bit (word length = 5, 6, 7, 8) STP = 1 1.5 stop bits (word length = 5) STP = 1 2 stop bits (word length = 6, 7, 8) |
| 3 | PRTY | 0 | Specifies whether parity is used. PRTY = 0 No parity PRTY = 1 Parity is generated. |
| 4 | EPRTY | 0 | Specifies whether even or odd parity is generated. EPRTY = 0 Odd parity is generated (if PRTY = 1). EPRTY = 1 Even parity is generated (if PRTY = 1). |
| 5 | FPTY | 0 | Selects the forced parity bit FPTY = 0 Parity is not forced. FPTY = 1 Parity bit is forced. If EPRTY = 0, the parity bit is forced to 1. |
| 6 | BRK | 0 | This bit is the break-control bit. BRK = 0 Normal operation BRK = 1 Forces SOUT into break condition (logic 0) |
| 7 | FEN | 0 | FIFO enable. This bit is used to disable/enable the FIFO. To reset the FIFO, the MCU clears and then sets this bit. FEN = 0 The FIFO is cleared and disabled. When disabled the selected receiver flow control is activated. FEN = 1 The FIFO is enabled and it can receive data. |

7.1.4 FCRL[2:1]: UART Flow Control Registers (2 Registers)

These registers provide the flow-control modes of operation (see Table 7–2 and Table 7–3).

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|------|-----|-----|------|------|
| 485E | DTR | RTS | RXOF | DSR | CTS | TXOA | TXOF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | TXOF | 0 | This bit controls the transmitter Xon/Xoff flow control. TXOF = 0 Disables transmitter Xon/Xoff flow control TXOF = 1 Enables transmitter Xon/Xoff flow control |
| 1 | TXOA | 0 | This bit controls the transmitter Xon-on-any/Xoff flow control. TXOA = 0 Disables the transmitter Xon-on-any/Xoff flow control TXOA = 1 Enables the transmitter Xon-on-any/Xoff flow control |
| 2 | CTS | 0 | Transmitter $\overline{\text{CTS}}$ flow-control enable bit. CTS = 0 Disables transmitter $\overline{\text{CTS}}$ flow control CTS = 1 $\overline{\text{CTS}}$ flow control is enabled, i.e., when $\overline{\text{CTS}}$ input pin is high, transmission is halted; when $\overline{\text{CTS}}$ pin is low, transmission resumes. |
| 3 | DSR | 0 | Transmitter $\overline{\text{DSR}}$ flow control enable bit. DSR = 0 Disables transmitter $\overline{\text{DSR}}$ flow control DSR = 1 $\overline{\text{DSR}}$ flow control is enabled, i.e., when $\overline{\text{DSR}}$ input pin is high, transmission is halted; when $\overline{\text{DSR}}$ pin is low, transmission resumes. |
| 4 | RXOF | 0 | This bit controls the receiver Xon/Xoff flow control. RXOF = 0 Receiver does not attempt to match Xon/Xoff characters. RXOF = 1 Receiver searches for Xon/Xoff characters. |
| 5 | RTS | 0 | Receiver $\overline{\text{RTS}}$ flow control enable bit RTS = 0 Disables receiver $\overline{\text{RTS}}$ flow control RTS = 1 Receiver $\overline{\text{RTS}}$ flow control is enabled. $\overline{\text{RTS}}$ output pin goes high when the receiver FIFO HALT trigger level is reached; it goes low, when the receiver FIFO RESTORE receiving trigger level is reached. |
| 6 | DTR | 0 | Receiver $\overline{\text{DTR}}$ flow control enable bit DTR = 0 Disables receiver $\overline{\text{DTR}}$ flow control DTR = 1 Receiver $\overline{\text{DTR}}$ flow control is enabled. $\overline{\text{DTR}}$ output pin goes high when the receiver FIFO HALT trigger level is reached; it goes low when the receiver FIFO RESTORE receiving trigger level is reached. |
| 7 | 485E | 0 | RS485 enable bit. This bit is used to configure the UARTs to control external RS485 transceivers. When configured in half-duplex mode, $\overline{\text{RTS}}$ and $\overline{\text{DTR}}$ are used to enable the RS485 driver and receiver (see Figure 12–2). 485E = 0 UART is in normal operation mode (full duplex). 485E = 1 The UART is in RS485 mode. In this mode $\overline{\text{RTS}}$ and $\overline{\text{DTR}}$ are active with opposite polarity (when $\overline{\text{RTS}} = 0$, $\overline{\text{DTR}} = 1$). When the DMA is ready to transmit, it drives $\overline{\text{RTS}} = 1$ (and $\overline{\text{DTR}} = 0$) 2-bit-time before transmission starts. When DMA terminates the transmission, it drives $\overline{\text{RTS}} = 0$ (and $\overline{\text{DTR}} = 1$) as soon as possible after transmission stops. When 485E is set to 1, the DTR and RTS bits in the MCR register have no effect. |

Table 7–2. Transmitter Flow-Control Modes

| MODE | | 3 | 2 | 1 | 0 |
|------|---------------------------------------|-----|-----|------|------|
| | | DSR | CTS | TXOA | TXOF |
| 0 | All flow control is disabled. | 0 | 0 | 0 | 0 |
| 1 | Xon/Xoff flow control is enabled. | 0 | 0 | 0 | 1 |
| 2 | Xon on any/ Xoff flow control | 0 | 0 | 1 | 0 |
| 3 | Not permissible (see Note 1) | X | X | 1 | 1 |
| 4 | $\overline{\text{CTS}}$ flow control | 0 | 1 | 0 | 0 |
| 5 | Combination flow control (see Note 2) | 0 | 1 | 0 | 1 |
| 6 | Combination flow control | 0 | 1 | 1 | 0 |
| 8 | $\overline{\text{DSR}}$ flow control | 1 | 0 | 0 | 0 |
| 9-E | Combination flow control | | | | |

NOTES: 1. This is an impermissible combination. If used, TXOA and TXOF are cleared.
2. Combination example: Transmitter stops when either CTS or Xoff is detected. Transmitter resumes when both CTS is negated and Xon is detected.

Table 7–3. Receiver Flow-Control Modes

| MODE | | 6 | 5 | 4 |
|------|---------------------------------------|-----|------|------|
| | | CTS | TXOA | TXOF |
| 0 | All flow control is disabled. | 0 | 0 | 0 |
| 1 | Xon/Xoff flow control is enabled. | 0 | 0 | 1 |
| 2 | $\overline{\text{RTS}}$ flow control | 0 | 1 | 0 |
| 3 | Combination flow control (see Note 3) | 0 | 1 | 1 |
| 4 | $\overline{\text{DTR}}$ flow control | 1 | 0 | 0 |
| 5 | Combination flow control | 1 | 0 | 1 |
| 6 | Combination flow control (see Note 4) | 1 | 1 | 0 |
| 7 | Combination flow control | 1 | 1 | 1 |

NOTES: 3. Combination example: TRS is asserted and Xoff is transmitted when FIFO is full. TRS is deasserted and Xon is transmitted when FIFO is empty.
4. Combination example: Both DTR and RTS are asserted when FIFO is full. Both DTR and RTS are deasserted when FIFO is empty.

7.1.4.1 Transmitter Flow Control

On reset (power-up, USB, or soft-reset) the transmitter defaults to the Xon state and the flow control is set to mode 0 (flow control is disabled).

7.1.5 MCR[2:1]: Modem-Control Registers (2 Registers)

These registers provide control for modem interface I/O, and definition of the flow control mode.

| | | | | | | | |
|-----|-----|-----|-----|-----|------|------|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LCD | LRI | RTS | RTR | SEN | LOOP | RCVE | URST |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | URST | 0 | <p>UART soft reset. This bit can be used by the MCU to reset the UART.</p> <p>URST = 0 Normal operation. Writing a 0 by MCU has no effect.</p> <p>URST = 1 When the MCU writes a 1 to this bit, a UART reset is generated (ORed with hard reset). When the UART exits the reset state, URST is cleared. The MCU can monitor this bit to determine if the UART completed the reset cycle.</p> |
| 1 | RCVE | 0 | <p>Receiver enable bit. This bit is valid only when 485E in FCRL is 1 (RS485 mode). When 485E = 0, this bit has no effect on the receiver.</p> <p>RCVE = 0 When 485E = 1, the UART receiver is disabled if $\overline{\text{RTS}}$ and $\overline{\text{DTR}}$ are active. (When data are transmitted the UART-receiver is disabled.)</p> <p>RCVE = 1 When 485 = 1, the UART receiver is enabled regardless of the $\overline{\text{RTS}}/\overline{\text{DTR}}$ state (UART receiver is on all the time).</p> |
| 2 | LOOP | 0 | <p>This bit controls the normal/loop-back mode of operation (see Figure 7–1).</p> <p>LOOP = 0 Normal operation</p> <p>LOOP = 1 Enable loop-back mode of operation. In this mode the following occurs:</p> <ul style="list-style-type: none"> • SOUT is set high. • SIN is disconnected. • The transmitter is looped back into the receiver. • The four modem-control inputs, $\overline{\text{CTS}}$, $\overline{\text{DSR}}$, $\overline{\text{DCD}}$, and $\overline{\text{RI}}$ are disconnected. • $\overline{\text{DTR}}$, $\overline{\text{RTS}}$, LRI and LCD are internally connected to the four modem-control inputs, and read in the MSR register as follows: <ul style="list-style-type: none"> • $\overline{\text{DTR}}$ is reflected in MSR[4] bit. • $\overline{\text{RTS}}$ is reflected in MSR[5] bit. • LRI is reflected in MSR[6] bit. • LCD is reflected in MSR[7] bit. |
| 3 | RSV | 0 | Reserved = 0 |
| 4 | DTR | 0 | <p>This bit controls the state of the $\overline{\text{DTR}}$ output pin (see Figure 7–1). This bit has no effect when auto-flow control is used.</p> <p>DTR = 0 Forces the $\overline{\text{DTR}}$ output pin to inactive (high)</p> <p>DTR = 1 Forces the $\overline{\text{DTR}}$ output pin to active (low)</p> |
| 5 | RTS | 0 | <p>This bit controls the state of the $\overline{\text{RTS}}$ output pin (see Figure 7–1). This bit has no effect when auto-flow control is used.</p> <p>RTS = 0 Forces the $\overline{\text{RTS}}$ output pin to inactive (high)</p> <p>RTS = 1 Forces the $\overline{\text{RTS}}$ output pin to active (low)</p> |
| 6 | LRI | 0 | <p>This bit is used for loopback mode only. When in loopback mode, this bit is reflected in MSR[6]-bit (see Figure 7–1).</p> <p>LRI = 0 Clears MSR[6] = 0</p> <p>LRI = 1 Sets MSR[6] = 1</p> |
| 7 | LCD | 0 | <p>This bit is used for loopback mode only. When in loopback mode, this bit is reflected in MSR[7]-bit (see Figure 7–1).</p> <p>LCD = 0 Clears MSR[7] = 0</p> <p>LCD = 1 Sets MSR[7] = 1</p> |

7.1.6 LSR[2:1]: Line-Status Registers (2 Registers)

These registers provide the status of the data transfer. DMA transfer is halted when any of OVR, PTE, FRE, BRK or EXIT is 1.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | RSV | TxE | RxF | BRK | FRE | PTE | OVR |
| R/O | R/O | R/O | R/O | R/C | R/C | R/C | R/C |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|--|
| 0 | OVR | 0 | This bit indicates the overrun condition of the receiver. If set, it halts the DMA transfer and generates a status interrupt (if enabled). OVR = 0 No overrun error OVR = 1 Overrun error has occurred. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 1 | PTE | 0 | This bit indicates the parity condition of the received byte. If set, it halts the DMA transfer and generates a status interrupt (if enabled). PTE = 0 No parity error in data received PTE = 1 Parity error in data received. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 2 | FRE | 0 | This bit indicates the framing condition of the received byte. If set, it halts the DMA transfer and generates a status interrupt (if enabled). FRE = 0 No framing error in data received FRE = 1 Framing error in data received. Clears when MCU writes a 1. Writing a 0 has no effect. |
| 3 | BRK | 0 | This bit indicates the break condition of the received byte. If set, it halts the DMA transfer and generates a status interrupt (if enabled). BRK = 0 No break condition BRK = 1 A break condition in data received was detected. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 4 | RxF | 0 | This bit indicates the condition of the receiver data register. Typically, the MCU does not monitor this bit since data transfer is done by the DMA controller. RxF = 0 No data in the RDR RxF = 1 RDR contains data. Generates Rx interrupt (if enabled). |
| 5 | TxE | 1 | This bit indicates the condition of the transmitter data register. Typically, the MCU does not monitor this bit since data transfer is done by the DMA controller. TxE = 0 TDR is <i>not</i> empty TxE = 1 TDR is empty. Generates Tx interrupt (if enabled). |
| 6–7 | RSV | 0 | Reserved = 0 |

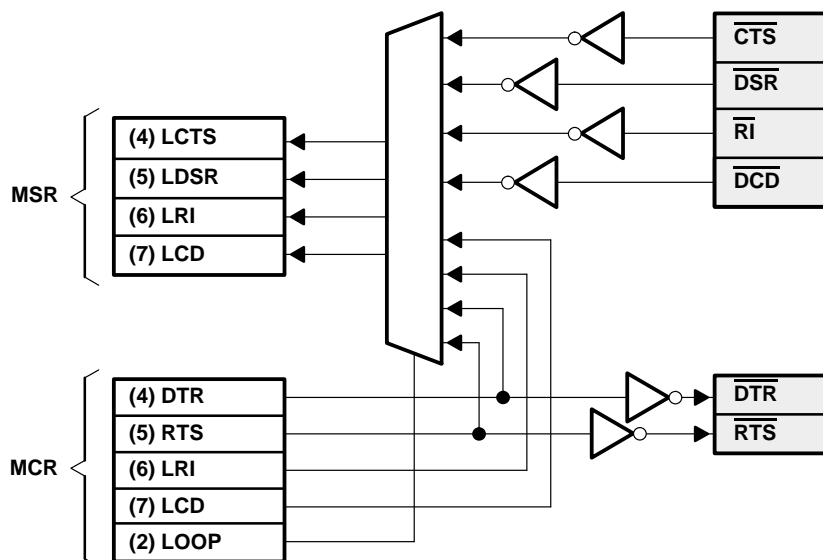


Figure 7–1. MSR and MCR Registers in Loopback Mode

7.1.7 MSR[2:1]: Modem-Status Registers (2 Registers)

These registers provide information about the current state of the control lines from the modem.

| | | | | | | | |
|-----|-----|------|------|-------------|-----|--------------|--------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LCD | LRI | LDSR | LCTS | Δ CD | TRI | Δ DSR | Δ CTS |
| R/O | R/O | R/O | R/O | R/C | R/C | R/C | R/C |

| BIT | NAME | RESET | FUNCTION |
|-----|--------------|-------|--|
| 0 | Δ CTS | 0 | This bit indicates that $\overline{\text{CTS}}$ input has changed state. Δ CTS = 0 Indicates no change in $\overline{\text{CTS}}$ input. Δ CTS = 1 Indicates that $\overline{\text{CTS}}$ input has changed state since the last time it was read. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 1 | Δ DSR | 0 | This bit indicates that $\overline{\text{DSR}}$ input has changed state. Δ DSR = 0 Indicates no change in $\overline{\text{DSR}}$ input. Δ DSR = 1 Indicates that $\overline{\text{DSR}}$ input has changed state since the last time it was read. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 2 | TRI | 0 | Trailing edge of the ring-indicator. This bit indicates that $\overline{\text{RI}}$ input has changed from low to high level. RI = 0 Indicates that $\overline{\text{RI}}$ input is high. RI = 1 Indicates that a transition from low to high level has occurred on $\overline{\text{RI}}$ input. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 3 | Δ CD | 0 | This bit indicates that $\overline{\text{CD}}$ input has changed state. Δ DC = 0 Indicates no change in $\overline{\text{CD}}$ input. Δ DC = 1 Indicates that $\overline{\text{CD}}$ input has changed state since the last time it was read. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 4 | LCTS | 0 | During loopback, this bit reflects the status of MCR[5] (see Figure 7–1). LCTS = 0 $\overline{\text{CTS}}$ input is high LCTS = 1 $\overline{\text{CTS}}$ input is low |
| 5 | LDSR | 0 | During loopback, this bit reflects the status of MCR[4] (see Figure 7–1). LDSR = 0 $\overline{\text{DSR}}$ input is high LDSR = 1 $\overline{\text{DSR}}$ input is low |
| 6 | LRI | 0 | During loopback, this bit reflects the status of MCR[6] (see Figure 7–1). LRI = 0 $\overline{\text{RI}}$ input is high LRI = 1 $\overline{\text{RI}}$ input is low |
| 7 | LCD | 0 | During loopback, this bit reflects the status of MCR[7] (see Figure 7–1). LCD = 0 $\overline{\text{CD}}$ input is high LCD = 1 $\overline{\text{CD}}$ input is low |

7.1.8 DLL[2:1]: Divisor Low-Byte Registers (2 Registers)

These registers contain the low-byte of the baud-rate divisor. The 1.8462-MHz clock is derived from the 48-MHz clock (dividing by 26). This clock is used for the baud-rate calculation (see Baud-Rate table)

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7–0 | D[7:0] | 08h | Low-byte value of the 16-bit divisor for generation of the baud clock in the baud-rate generator. |

7.1.9 DLH[2:1]: Divisor High-Byte Registers (2 Registers)

These registers contain the high-byte of the baud-rate divisor.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|---------|-------|---|
| 7–0 | D[15:8] | 00h | High-byte value of the 16-bit divisor for generation of the baud clocks in the baud rate generator. |

7.1.9.1 Baud Rate Calculation

The following formulas are used to calculate the baud-rate clock and the divisors. The baud-rate clock is derived from the 48-MHz master-clock (dividing by 6.5). Table 7–4 presents the divisors used to achieve the desired baud rates, together with the associated rounding errors.

$$\text{Baud CLK} = \frac{48 \text{ MHz}}{6.5} = 7.384615 \text{ MHz}$$

$$\text{Divisor} = \frac{7.384615 \times 10^6}{\text{Baud Rate} \times 16}$$

Table 7–4. DLL/DLH Values and Resulting Baud Rates

| DESIRED BAUD | DLL/DLH VALUE | | ACTUAL BAUD | ERROR % |
|-----------------|---------------|------|----------------|---------|
| | DEC. | HEX. | | |
| 50 | 9 231 | 240F | 50.00 | 0.002 |
| 75 | 6 154 | 180A | 75.00 | 0.002 |
| 110 | 4 196 | 1064 | 109.99 | 0.005 |
| 135 | 3 432 | 0D68 | 134.48 | 0.014 |
| 150 | 3 077 | 0C05 | 150.00 | 0.002 |
| 300 | 1 538 | 0602 | 300.09 | 0.030 |
| 600 | 769 | 0301 | 600.18 | 0.030 |
| 1 200 | 385 | 0181 | 1 198.80 | 0.100 |
| 1 800 | 256 | 0100 | 1 802.88 | 0.160 |
| 2 000 | 231 | 00E7 | 1 998.00 | 0.100 |
| 2 400 | 192 | 00C0 | 2 403.85 | 0.160 |
| 3 600 | 128 | 0080 | 3 605.77 | 0.160 |
| 4 800 | 96 | 0060 | 4 807.69 | 0.160 |
| 7 200 | 64 | 0040 | 7 211.54 | 0.160 |
| 9 600 | 48 | 0030 | 9 615.38 | 0.160 |
| 19 200 | 24 | 0018 | 19 230.77 | 0.160 |
| 38 400 | 12 | 000C | 38 461.54 | 0.160 |
| 57 600 | 8 | 0008 | 57 692.31 | 0.160 |
| 115 200 | 4 | 0004 | 115 384.62 | 0.160 |
| 230 400 | 2 | 0002 | 230 769.23 | 0.160 |
| 460 800 | 1 | 0001 | 461 538.46 | 0.160 |

7.1.10 XON[2:1]: Xon Registers (2 Registers)

These registers contain a value that is compared to the received data stream. Detection of a match interrupts the MCU (only if the interrupt enable bit is set). This value is also used for Xon transmission.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7–0 | D[7:0] | 0x00 | Xon value to be compared to the incoming data stream. |

7.1.11 XOFF[2:1]: Xoff Registers (2 Registers)

These registers contain a value that is compared to the received data stream. Detection of a match halts the DMA transfer, and interrupts the MCU (only if the interrupt enable bit is set). This value is also used for Xoff transmission.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7–0 | D[7:0] | 0x00 | Xon value to be compared to the incoming data stream. |

7.1.12 MASK[2:1]: UART Interrupt-Mask Registers (2 Registers)

These registers control the UART interrupt sources. The corresponding DMA must be disabled before changing register values to ensure data integrity.

| | | | | | | | |
|-----|-----|-----|-----|-----|------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | RSV | RSV | RSV | RSV | TRIE | SIE | MIE |
| R/O | R/O | R/O | R/O | R/O | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | MIE | 0 | This bit controls the UART-modem interrupt. MIE = 0 Modem interrupt is disabled. MIE = 1 Modem interrupt is enabled. |
| 1 | SIE | 0 | This bit controls the UART-status interrupt. SIE = 0 Status interrupt is disabled. SIE = 1 Status interrupt is enabled. |
| 2 | TRIE | 0 | This bit controls the UART-TxE/RxF interrupts. TRIE = 0 TxE/RxF interrupts are disabled. TRIE = 1 TxE/RxF interrupts are enabled. |
| 7–3 | RSV | 0 | Reserved = 0 |

7.2 UART Data Transfer

Figure 7–2 illustrates the data transfer between the UARTs and the host using the DMA controller and the USB buffer manager (UBM). A buffer of 512 bytes is reserved for buffering all UART channels (2 transmit and 2 receive buffers). Each UART channel has 64-bytes of double-buffer space (X- and Y-buffer). When the DMA writes to the X-buffer, the UBM reads from the Y-buffer. Similarly, when the DMA reads from the X-buffer, the UBM writes to the Y-buffer. The DMA channel is configured to operate in the continuous mode (by setting DMACDR[CNT] = 1). Once the MCU enables the DMA, data transfer toggles between the UMB and the DMA without MCU intervention. See section 6.2 for DMA transfer-termination conditions.

7.2.1 Receiver Data Flow

Every UART receiver has a 32-byte FIFO. The receiver FIFO has two trigger levels. One is the high-level mark (HALT), which is set to 12 bytes, and the other is the low-level mark (RESTORE), which is set to 4-bytes. When the HALT mark is reached, either the $\overline{\text{RTS}}$ pin goes high or Xoff is transmitted (depending on the auto setting). When the FIFO reaches the RESTORE mark, then either the $\overline{\text{RTS}}$ pin goes low or Xon is transmitted.

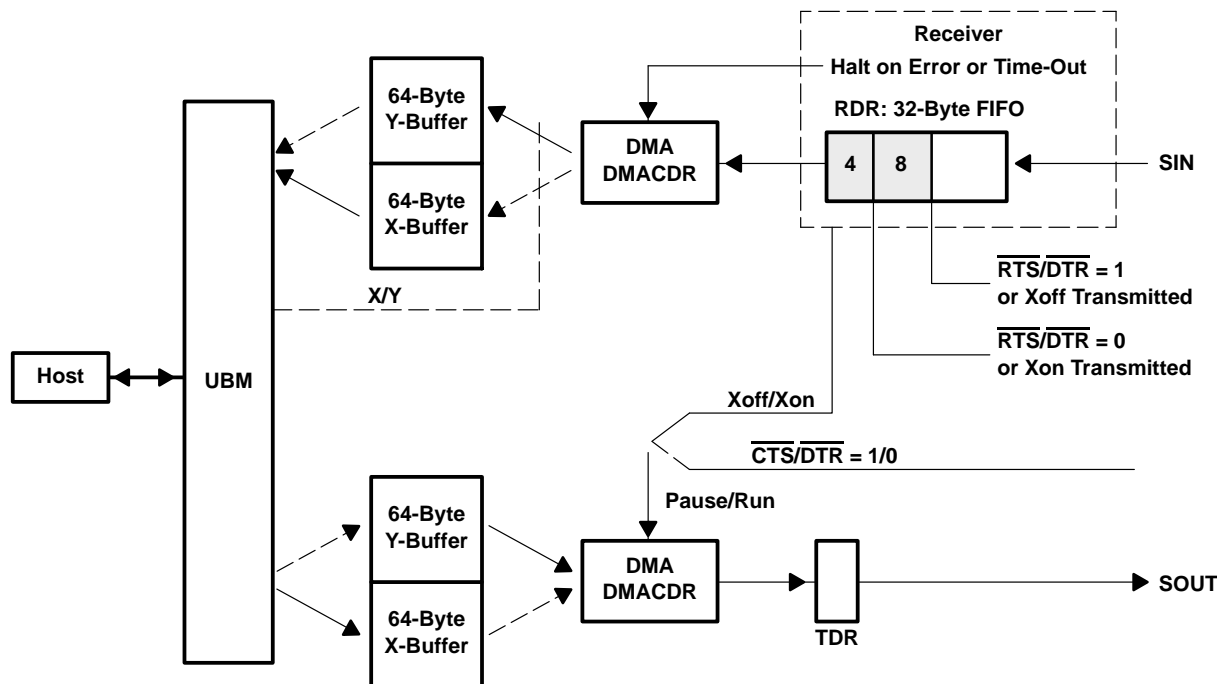


Figure 7-2. Receiver/Transmitter Data Flow

7.2.2 Hardware Flow Control

Figure 7-3 illustrates the connection necessary to achieve hardware flow control. $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ signals are provided for this purpose. Auto-CTS and auto-RTS (and Xon/Xoff) can be enabled/disabled independently by programming the FCRL register.

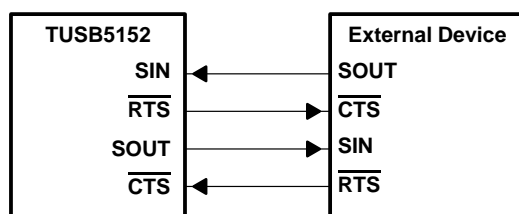


Figure 7-3. Auto Flow Control Interconnect

7.2.3 Auto-RTS (Receiver Control)

In this mode, the $\overline{\text{RTS}}$ output pin signals the receiver FIFO status to an external device. The $\overline{\text{RTS}}$ output signal is controlled by the high-level and low-level marks of the FIFO. When the high-level mark is reached, $\overline{\text{RTS}}$ goes high, signaling to an external sending device to halt its transfer. Conversely, when the low-level mark is reached, $\overline{\text{RTS}}$ goes low, signaling to an external sending device to resume its transfer. Data transfer from the FIFO to the X/Y-buffer is performed by the DMA controller. See Section 6.2 for DMA transfer termination conditions.

7.2.4 Auto-CTS (Transmitter Control)

In this mode the $\overline{\text{CTS}}$ input pin controls the transfer from internal buffer (X or Y) to the TDR. When the DMA controller transfers data from the Y-buffer to the TDR and the $\overline{\text{CTS}}$ input pin goes high, the DMA controller is suspended until

$\overline{\text{CTS}}$ goes low. Meanwhile, the UBM is transferring data from the host to the X-buffer. When $\overline{\text{CTS}}$ goes low, the DMA resumes the transfer. Data transfer continues alternating between the X and Y buffers, without MCU intervention. See Section 6.2 for DMA transfer termination conditions.

7.2.5 Xon/Xoff Receiver Flow Control

To enable Xon/Xoff flow control, certain MCR bits must be set as follows: $\text{MCR}[5] = 1$ and $\text{MCR}[7:6] = 0$. In this mode, the Xon/Xoff bytes are transmitted to an external sending device to control the device transmission. When the high-level mark (of the FIFO) is reached, the Xoff byte is transmitted, signaling to an external sending device to halt its transfer. Conversely, when the low-level mark is reached, the Xon byte is transmitted, signaling to an external sending device to resume its transfer. Data transfer from the FIFO to X/Y-buffer is performed by the DMA controller.

7.2.6 Xon/Xoff Transmitter Flow Control

To enable Xon/Xoff flow control, certain MCR bits must be set as follows: $\text{MCR}[5] = 1$ and $\text{MCR}[7:6] = 0$. In this mode, the incoming data are compared to the XON and XOFF registers. If a match to XOFF is detected, the DMA is paused. If a match to XON is detected, the DMA resumes. Meanwhile, the UBM is transferring data from the host to the X-buffer. The MCU does not switch the buffers unless the Y-buffer is empty and X-buffer is full. When Xon is detected, the DMA resumes the transfer.

8 IEEE 1284 Parallel Port

8.1 1284 Registers

Table 8-2 is a summary of the parallel-port modes of operations, and Figure 8–1 illustrates the data flow for output and input transfers. A single PPDAT register is shared for output and input transfers; the RxF/TxE signal is used for flow control. Table 8–1 summarizes the registers provided for configuration, status monitoring, and data I/O. Table 8–3 summarizes the port signal definition.

Table 8–1. Parallel-Port Registers

| REGISTER NAME | ACCESS | MODE | FUNCTION |
|---------------|--------|------|--------------------------|
| PPMCR | R/W | All | Mode control register |
| PPIMSK | R/W | All | Interrupt mask register |
| PPSTA | R/W | All | Status register |
| PPCTL | R/W | All | Control register |
| PPDAT | R/W | All | Data register |
| PPADR | R/W | 2,3 | EPP/ECP address register |

Table 8–2. Parallel-Port Mode Summary

| MODE | DESCRIPTION | OUTPUT |
|------|--|------------|
| 00 | Centronics mode: In this mode, the port direction is output only, and the DIR bit has no effect. The MCU writes data to PPDAT and the control signals are generated by the MCU. | See Note 2 |
| 01 | SPP mode (bidirectional auto Centronics): This is the bidirectional Centronics mode. In this mode, the port direction can be changed by the DIR bit. The MCU or DMA writes data to PPDAT and the control signals are generated automatically. This mode can support nibble operation. | See Note 2 |
| 10 | ECP mode: In the output direction (DIR = 0) data written to PPADR and to PPDAT are transmitted automatically using the ECP protocol. In the input direction (DIR = 1), data bytes are transferred from the peripheral and placed in the PPDAT register. CPU or DMA transfer is supported. | See Note 1 |
| 11 | EPP mode: In the output direction (DIR = 0) data written to PPADR and to PPDAT are transmitted automatically using the EPP protocol. In the input direction (DIR = 1), data bytes are transferred from the peripheral and placed in the PPDAT register. CPU or DMA transfer is supported. | See Note 1 |

NOTES: 1. 3-state CMOS output ± 14 -mA drive/sink (control signals only)
2. Open-drain output -14 -mA sink (control signals only)

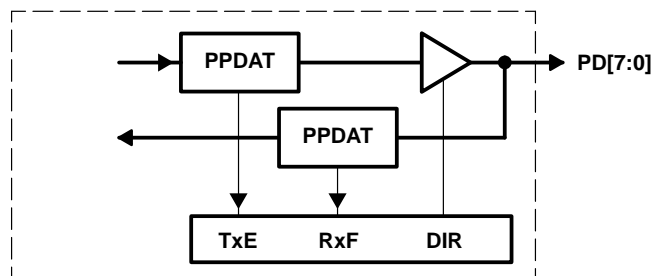


Figure 8–1. P-Port Data I/O Flow

Table 8–3. Parallel-Port Signal Definition Summary

| DRIVEN BY | SPP | EPP | ECP |
|------------|--------------------------|---------------------------|----------------------------|
| I/O | PD[7:0] | PD[7:0] | PD[7:0] |
| TUSB5152 | $\overline{\text{SLIN}}$ | $\overline{\text{ASTRB}}$ | $\overline{\text{SLIN}}$ |
| TUSB5152 | $\overline{\text{STB}}$ | $\overline{\text{WRITE}}$ | HOSTCLK |
| TUSB5152 | $\overline{\text{AFD}}$ | $\overline{\text{DSTRB}}$ | HOSTACK |
| TUSB5152 | $\overline{\text{INIT}}$ | $\overline{\text{INIT}}$ | $\overline{\text{REVREQ}}$ |
| Peripheral | $\overline{\text{ACK}}$ | INTR | PERIPCLK |
| Peripheral | $\overline{\text{FALT}}$ | User defined | PERIPREQ |
| Peripheral | SLCT | User defined | User defined |
| Peripheral | PER | User defined | MPRIPREQ |
| Peripheral | BUSY | $\overline{\text{WAIT}}$ | PERIPHACK |

8.1.1 PPCNT: P-Port Counter Register

As a delay counter, this register records the time delay in 40-ns increments between assertion of the BUSY signal and assertion of the STB signal. The same counter is also used for the STB pulse duration.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7–0 | D[7:0] | 12h | Delay and pulse counter for STB signal in 40-ns increments. |

8.1.2 PPMCR: P-Port Mode Control Register

This register is used to configure the port mode of operation. In addition, it provides bits that control the transfer direction.

| | | | | | | | |
|-----|-----|-------|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PEN | RSV | GPIEN | RSV | DIR | NBL | M1 | M0 |
| R/W | R/O | R/W | R/O | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 1–0 | M[1:0] | 00b | Port mode definition. This field defines the P-port mode of operation (see Table 8–2 for more details). 00 = Centronics mode. DIR bit has no effect. All control signals are generated by the MCU. 01 = SPP mode: Bidirectional auto Centronics mode 10 = EPP mode 11 = ECP mode |
| 2 | NBL | 0 | Nibble mode control bit. This bit is valid only in SPP mode. NBL = 0 SPP is in byte mode. NBL = 1 SPP is in nibble mode. |
| 3 | DIR | 0 | Transfer direction bit. Valid for modes 1 to 3. DIR = 0 The PPDAT drives PD[7:0] data bus (transmitter). DIR = 1 PD[7:0] data bus is input to PPDAT register (receiver). |
| 4 | RSV | 0 | Reserved = 0 |
| 5 | GPIEN | 1 | GPI-mode enable bit GPIEN = 0 Normal IEEE 1284 operation GPIEN = 1 The PPDAT register continuously latches PD[7:0]. The DIR and M[1:0] bits are ignored and PD[7:0] goes into the input mode. |
| 6 | RSV | 0 | Reserved = 0 |
| 7 | PEN | 0 | PP enable bit. This bit is used by the MCU to enable/disable the P-port data transfer in all modes. PEN = 0 The PP is disabled. Data transfer (via DMA or MCU) cannot take place. To terminate PP transfers, the MCU must clear this bit. PEN = 1 The PP is enabled and data transfer (via DMA or MCU) can take place. |

8.1.3 PPIMSK: P-Port Interrupt Mask Register

This register is used to mask the parallel-port interrupt sources.

| | | | | | | | |
|-----|------|------|-----|-------|-----|------|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | ACKE | PERE | RSV | FLINE | RSV | ETxE | ERxF |
| R/O | R/W | R/W | R/O | R/W | R/O | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|-------|-------|---|
| 0 | ERxF | 0 | This bit disables/enables the RxF interrupt. ERxF = 0 RxF does not generate an interrupt. ERxF = 1 When RxF is set to 1, an interrupt is generated. |
| 1 | ETxE | 0 | This bit is used to enable/disable the TxE interrupt. ETxE = 0 TxE does not generate an interrupt. ETxE = 1 When TxE is set to 1, an interrupt is generated. |
| 2 | RSV | 0 | Reserved =0 |
| 3 | FLINE | 0 | This bit is used to enable/disable the FALT interrupt. FLINE = 0 FALT does not generate an interrupt. FLINE = 1 When FALT is set to 1, an interrupt is generated. |
| 4 | RSV | 0 | Reserved =0 |
| 5 | PERE | 0 | PERE = 0 PER does not generate an interrupt. PERE = 1 When PER is set to 1, an interrupt is generated. |
| 6 | ACKE | 0 | ACKE = 0 ACK does not generate an interrupt. ACKE = 1 When ACK is set to 1, an interrupt is generated. |
| 7 | RSV | 0 | Reserved =0 |

8.1.4 PPSTA: P-Port Status Register

This register is used to monitor status conditions for all modes.

| | | | | | | | |
|------|-----|-----|------|------|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BUSY | ACK | PER | SLCT | FALT | RSV | TxE | RxF |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | RxF | 0 | Receiver full indication. This bit is used only when the MCU is performing the data transfer and is valid only when DIR = 1. This bit indicates the condition of the PPDAT register in the input direction. RxF =0 PPDAT has no data RxF =1 PPDAT contains a byte. This bit is cleared when the MCU reads the byte from the PPDAT register. |
| 1 | TxE | 1 | Transmitter empty indication. This bit is used only when the MCU is performing the data transfer and is valid only when DIR = 0. This bit indicates the condition of the PPDAT register in the output direction. TxE =0 PPDAT contain a byte and the MCU cannot write to it. This bit is cleared when the byte is accepted by the peripheral. TxE =1 PPDAT is empty and the MCU can write data to it. |
| 2 | RSV | 0 | Reserved = 0 |
| 3 | FALT | X | Corresponds to inverse of $\overline{\text{FALT}}$ input signal |
| 4 | SLCT | X | Corresponds to SLCT input signal |
| 5 | PER | X | Corresponds to PER input signal |
| 6 | ACK | X | Corresponds to the inverse of $\overline{\text{ACK}}$ input signal |
| 7 | BUSY | X | Corresponds to BUSY input signal |

8.1.5 PPCTL: P-Port Control Register

This register is used to control the P-port output signals.

| | | | | | | | |
|-----|-----|-----|-----|------|------|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | RSV | RSV | RSV | SLIN | INIT | AFD | STB |
| R/O | R/O | R/O | R/O | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|--|
| 0 | STB | 0 | This bit affects the inverse of $\overline{\text{STB}}$ output pin. STB = 0 $\overline{\text{STB}}$ output is set to 1 (inactive). STB = 1 $\overline{\text{STB}}$ output is set to 0 (active). |
| 1 | AFD | 0 | This bit affects the inverse of $\overline{\text{AFD}}$ output pin. AFD = 0 $\overline{\text{AFD}}$ output is set to 1 (inactive). AFD = 1 $\overline{\text{AFD}}$ output is set to 0 (active). |
| 2 | INIT | 0 | This bit affects the inverse of $\overline{\text{INIT}}$ output pin. INIT = 0 $\overline{\text{INIT}}$ output is set to 1 (inactive). INIT = 1 $\overline{\text{INIT}}$ output is set to 0 (active). |
| 3 | SLIN | 0 | This bit affects the inverse of $\overline{\text{SLIN}}$ output pin. SLIN = 0 $\overline{\text{SLIN}}$ output is set to 1 (inactive). SLIN = 1 $\overline{\text{SLIN}}$ output is set to 0 (active). |
| 7–4 | RSV | 0 | Reserved = 0 |

Table 8–4. PP Output-Pin Activation Source

| PPCTL REGISTER | SPP | | EPP | | ECP | | OUTPUT PIN |
|----------------|-----|-----|-----|-----|-----|-----|--------------------------|
| | IN | OUT | IN | OUT | IN | OUT | |
| STB | H | H | H | H | H | H | $\overline{\text{STB}}$ |
| AFD | H | H | H | H | H | H | $\overline{\text{AFD}}$ |
| INIT | F | F | F | F | HF† | HF† | $\overline{\text{INIT}}$ |
| SLIN | F | F | F | H | F | F | $\overline{\text{SLIN}}$ |

† INIT is controlled by the DIR bit in the PPMCR register.

H = Output pin is driven by hardware.

F = Output pin is driven by firmware (the MCU can toggle this bit).

8.1.6 PPADR: EPP/ECP Address Register

This register is used in the EPP/ECP modes of operation only. In these modes, it is used as the command/address register. Writing to this register generates the EPP/ECP address-protocol signals automatically.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RSV | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| R/O | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 6–0 | A[6:0] | 00h | Channel address or run-length count value (0–127) |
| 7 | RSV | 0 | Reserved = 0 |

8.1.7 PPDAT: P-Port Data Register

When GPIEN is 0, this register is used for data I/O for all modes. When in the output direction (DIR = 0), writing to this register places the data into PD[7:0]. When in the input direction (DIR = 1), this register contains the byte entered by the peripheral (see Figure 8–1). When GPIEN is 1, this register contains the real-time value of the PD[7:0] bus.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|------------------------------|
| 7–0 | D[7:0] | 00h | P-Port data output and input |

8.2 Mode-1 (SPP): Bidirectional Centronics Mode

This mode supports data transfer in the output and input directions. The PPMCR[DIR] bit controls the transceiver direction (when DIR = 0, the port is an output; when DIR = 1, the port is an input). In this mode, all handshake signals are generated automatically without the need for the MCU to toggle bits. The following describes the output and input sequences:

8.2.1 MCU Output Sequence

1. MCU writes M[2:0] = 001b to PPMCR to set the port to mode-1, and sets PPMCR[DIR] = 0.
2. MCU sets PPIMSK[ETxE] = 1. This enables the TxE interrupt.
3. MCU reads a byte from buffer (in SRAM).
4. MCU writes the byte to PPDAT register.
5. Writing to PPDAT starts the transfer.
6. Transferring the byte interrupts the MCU, indicating that the port is ready for the next byte.
7. Steps 3 to 6 are repeated until the SRAM buffer is empty.
8. MCU sets PPIMSK[ETxE] = 0. This disables the TxE interrupt.

8.2.2 MCU Input Sequence

1. MCU writes M[2:0] = 001b to PPMCR to set the port to mode-1, and sets PPMCR[DIR] = 1.
2. MCU sets PPIMSK[ERxF] = 1. This enables the RxF interrupt.
3. When a byte is received, PPMCR[RxF] is set to 1. This interrupts the MCU, indicating that the port contains a byte.
4. MCU reads a byte from PPDAT register.
5. MCU writes the byte to buffer (in SRAM).
6. Steps 3 to 6 are repeated until the buffer (in SRAM) is full.
7. MCU sets PPIMSK[ERxF] = 0. This disables the RxF interrupt.

8.2.3 DMA Input Sequence

1. MCU writes M[2:0] = 001b to PPMCR to set the port to mode-1.
2. MCU sets PPIMSK[ERxF] = 0 (no RxF interrupt).
3. MCU sets the proper DMA for input data transfer.
4. DMA reads a byte from the PPDAT register (RxF is used for DMA handshake).
5. DMA writes the byte to buffer (in SRAM).
6. Steps 4 and 5 are repeated without MCU involvement until the full buffer is received or until an error condition is detected.
7. When the full buffer is received, the DMA switches to X/Y-buffer.
8. Steps 4 to 7 continue until terminated by time-out or by the MCU.
9. When terminated, the MCU sets the DMASTA register.

8.3 Mode-2/3: EPP and ECP Modes

These modes of operation support bidirectional data transfer per ECP and EPP protocols. In the output direction, data/command written to PPDAT/PPADR are placed in the PPDAT register and are transmitted automatically using ECP/EPP signaling. Typically, the MCU writes to PPADR to set the device address, and data transfer is done by the DMA to PPDAT register. In the input direction, reverse-channel addressing is not supported. Only data input is supported. Data input can be accomplished by either MCU or DMA transfer. The following describes the output and input sequence:

8.3.1 DMA Output Sequence

1. MCU writes M[2:0] = 010b to PPMCR to set the port to mode-2.
2. MCU sets PPIMSK[ETxE] = 0 (no TxE interrupt).
3. MCU writes an address value to the PPADR register. This byte is placed into PD[7:0] and tagged as a command byte.
4. MCU sets the proper DMA for output data transfer.
5. DMA reads a byte from buffer (in SRAM).
6. DMA writes the byte to the PPDAT register (TxE is used for DMA handshake).
7. Steps 5 and 6 are repeated without MCU involvement until either the full buffer has been transmitted, or an error condition has been detected. (A buffer size of up to 64 bytes can be transferred.)
8. When the full buffer has been transmitted, the DMA interrupts the MCU.
9. MCU checks for any error condition.

8.3.2 DMA Input Sequence

1. MCU writes M[2:0] = 010b to PPMCR to set the port to mode-2.
2. MCU sets PPIMSK[ERxF] = 0 (no RxF interrupt).
3. MCU sets the proper DMA for input data transfer.
4. DMA reads a byte from the PPDAT register (RxF is used for DMA handshake).
5. DMA writes the byte to buffer (in SRAM).
6. Steps 4 and 5 are repeated without MCU involvement until either the full buffer has been transmitted, or an error condition has been detected. (A buffer size of up to 64 bytes can be transferred.)
7. When the full buffer has been received, the DMA interrupts the MCU.
8. MCU checks for any error condition.

8.4 Host IN/OUT Transaction From/To P-Port

8.4.1 Host IN Transaction

Figure 8–2 illustrates the data flow for a USB IN transaction (from P-port to host) using the DMA transfer. The MCU sets the IEDB (input-EDB) with the proper parameters (start-address, byte-count, etc.). In addition, the MCU sets the DMA and P-port transfer direction. Once initialization has completed, MCU starts the DMA transfer by setting EN = 1 (DMA is in continuous mode). Initially, the UBM NAKs the host because the buffer is empty (NAK = 1). Once the DMA completes the 64-byte transfer to the X-Buffer, it updates the byte-count (and sets NAK = 0) in IEDB and switches to the Y-buffer (DMA controls the X/Y-bit). The UBM starts transmitting the X-Buffer to host and the DMA transfers data from the parallel port to the Y-Buffer. When the UBM completes the transfer, it sets NAK = 1 indicating to the DMA that the buffer is ready. This ping-pong transfer continues until the MCU halts the transfer (or times out). If for any reason the DMA does not transfer a full 64-bytes, it updates the (partial) byte-count value (and sets NAK = 0) in IEDB, and interrupts the MCU. The UBM uses this count to transfer a partial packet to host. Note: the DMA increments the byte-count and save it in IEDB at the end of the transfer. On the other hand, the UBM uses the byte-count from IEDB as the transfer count but does not update the IEDB at the end of the transfer, i.e., the DMA both increments and updates; the UBM only decrements.

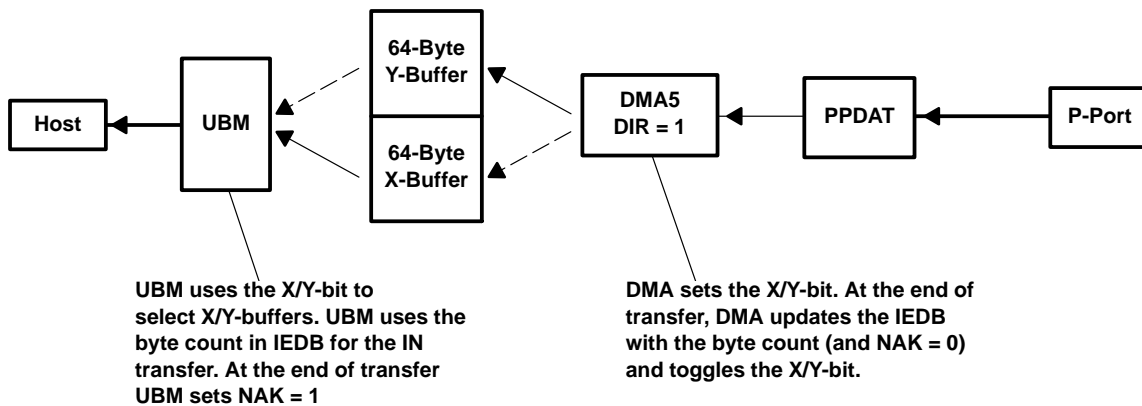


Figure 8–2. USB IN Transaction From P-Port (DMA CNT Mode)

8.4.2 Host OUT Transaction

Figure 8–3 illustrates the data flow for a USB OUT transaction (from host to P-port), using the DMA transfer (DMA in continuous mode). The MCU sets the OEDB (Output-EDB) with the proper parameters (start-address, byte-count, etc.). In addition, the MCU sets the DMA and P-port transfer direction. Once initialization completes, the MCU clears the NAK bit to 0 (in OEDB-X) which starts the UBM data transfer from host to X-buffer. Once the X-buffer is full, the UBM updates OEDB-X, sets NAK = 1, and notifies the DMA. The DMA toggles the X/Y-bit. UBM starts the transfer from the host to the Y-buffer (X/Y-bit points to Y) and the DMA transfers from the X-buffer to P-port. The ping-pong transfer continues until the MCU (or an error) halts the transfer. If the UBM does not transfer a full 64-bytes, it updates the byte-count value (and sets NAK = 1) in OEDB to reflect the partial buffer size. The DMA uses this count to transfer a partial packet to P-port and interrupt the MCU. Note that the UBM increments the byte-count and saves it in OEDB at the end of the transfer. On the other hand, the DMA uses the byte-count from OEDB as the transfer count but does not update the OEDB at the end of transfer, i.e., the UBM both increments and updates; whereas, the DMA only decrements.

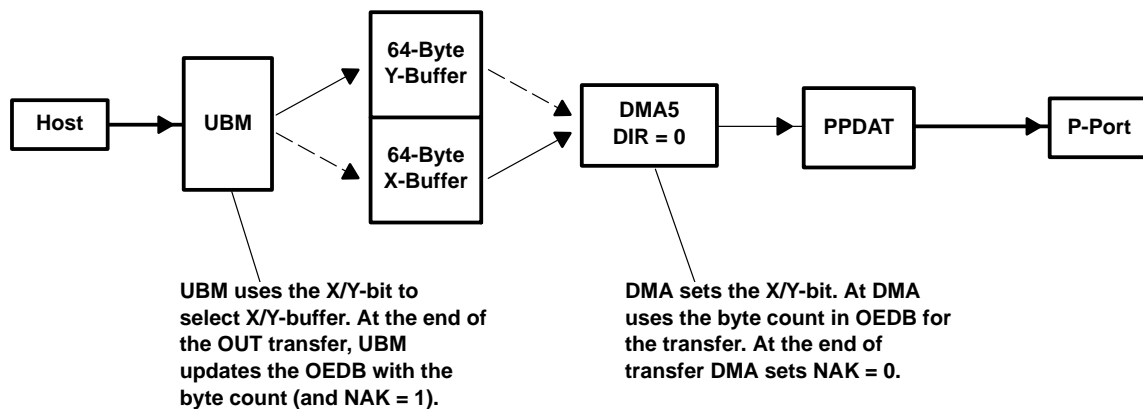


Figure 8–3. USB OUT Transaction to P-Port

9 Interrupts

9.1 8052 Interrupt and Status Registers

All 8052-standard, 5-interrupt sources are preserved. SIE is the standard interrupt-enable register that controls the five interrupt sources. All the additional interrupt sources are ORed together to generate EX0. The $\overline{\text{XINT0}}$ signal is provided to interrupt an external MCU (see interrupt connection diagram, Figure 9–1). All the interrupts are listed in Table 9–1.

Table 9–1. 8052 Interrupt Location Map

| INTERRUPT SOURCE | DESCRIPTION | START ADDRESS | COMMENTS |
|------------------|----------------------|---------------|-----------------------------------|
| ES | UART interrupt | 0023h | |
| ET1 | Timer-1 interrupt | 001Bh | |
| EX1 | External interrupt-1 | 0013h | |
| ET0 | Timer-0 interrupt | 000Bh | |
| EX0 | External interrupt-0 | 0003h | Used for all internal peripherals |
| Reset | | 0000h | |

9.1.1 8052 Standard Interrupt Enable Register

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EA | RSV | RSV | ES | ET1 | EX1 | ET0 | EX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|------|------|-------|---|
| 0 | EX0 | 0 | Enable or disable external interrupt-0 EX0 = 0 External interrupt-0 is disabled. EX0 = 1 External interrupt-0 is enabled. |
| 1 | ET0 | 0 | Enable or disable timer-0 interrupt ET0 = 0 Timer-0 interrupt is disabled. ET0 = 1 Timer-0 interrupt is enabled. |
| 2 | EX1 | 0 | Enable or disable external interrupt-1 EX1 = 0 External interrupt-1 is disabled. EX1 = 1 External interrupt-1 is enabled. |
| 3 | ET1 | 0 | Enable or disable timer-1 interrupt ET1 = 0 Timer-1 interrupt is disabled. ET1 = 1 Timer-1 interrupt is enabled. |
| 4 | ES | 0 | Enable or disable serial port interrupts ES = 0 Serial-port interrupt is disabled. ES = 1 Serial-port interrupt is enabled. |
| 5, 6 | RSV | 0 | Reserved |
| 7 | EA | 0 | Enable or disable all interrupts (global disable) EA = 0 Disable all interrupts. EA = 1 Each interrupt source is individually controlled. |

9.1.2 Additional Interrupt Sources

All non-standard 8052 interrupts (DMA, I²C, etc.) are ORed to generate an internal INT0. Note: the *external* INT0 is *not* used. Furthermore, the INT0 must be programmed as an active, low-level interrupt (not edge-triggered). A vector interrupt register is provided to identify all interrupt sources (see vector-interrupt register definition). Up to 64 interrupt vectors are provided. It is the responsibility of the MCU to read the vector and dispatch to the proper interrupt routine.

9.1.3 IEPINT: Input Endpoint Interrupt Request Register

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BITS | NAME | RESET | FUNCTION |
|------|--------|-------|---|
| 7–0 | E[7:0] | 0 | These bits indicate which input endpoint interrupt is pending (IEP 7–0). E[n] = 0 No interrupt pending (n = 7:0). E[n] = 1 Indicates that the corresponding endpoint generated an interrupt. This is set by the hardware and is cleared when the MCU writes to the VECINT register. |

9.1.4 OEPINT: Output Endpoint Interrupt Request Register

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BITS | NAME | RESET | FUNCTION |
|------|--------|-------|---|
| 7–0 | E[7:0] | 0 | These bits indicate which output endpoint interrupt is pending (OEP 7–0). E[n] = 0 No interrupt pending E[n] = 1 Indicates that the corresponding endpoint generated an interrupt. This is set by the hardware and is cleared when the MCU writes to the VECINT register. |

9.1.5 VECINT: Vector Interrupt Register

This register contains a vector value, which identifies the internal interrupt source that trapped to location 0003H. Writing (any value) to this register removes the vector and updates the next vector value (if another interrupt is pending). Note that the vector value is offset; therefore, its value is in increments of two (bit-0 is set to 0). When no interrupt is pending, the vector is set to 00h (see Table 9–2). As shown, the interrupt vector is divided to two fields: I[2:0] and G[3:0]. The I-field defines the interrupt source within a group (on a first-come-first-served basis). In the G-field, which defines the group number, group G0 is the lowest, and G15 is the highest, priority.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| G3 | G2 | G1 | G0 | I2 | I1 | I0 | 0 |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BITS | NAME | RESET | FUNCTION |
|------|--------|-------|---|
| 0 | RSV | 0 | Reserved |
| 3–1 | I[2:0] | 000b | This field defines the interrupt source in a given group (see Table 9–2). Bit-0 = 0 always; therefore, vector values are offset by two. |
| 7–4 | G[3:0] | 0x0 | This field defines the interrupt group. I[2:0] and G[3:0] combine to produce the actual interrupt vector. |

Table 9–2. Vector Interrupt Values

| G[3:0] (Hex) | I[2:0] (Hex) | VECTOR (Hex) | INTERRUPT SOURCE |
|-----------------|-----------------|-----------------|--------------------------------|
| 0 | 0 | 00 | <i>No interrupt</i> |
| 1 | 0 | 10 | <i>Not used</i> |
| 1 | 1 | 12 | Output endpoint_1 |
| 1 | 2 | 14 | Output endpoint_2 |
| 1 | 3 | 16 | Output endpoint_3 |
| 1 | 4 | 18 | Output endpoint_4 |
| 1 | 5 | 1A | Output endpoint_5 |
| 1 | 6 | 1C | Output endpoint_6 |
| 1 | 7 | 1E | Output endpoint_7 |
| 2 | 0 | 20 | <i>Not used</i> |
| 2 | 1 | 22 | Input endpoint_1 |
| 2 | 2 | 24 | Input endpoint_2 |
| 2 | 3 | 26 | Input endpoint_3 |
| 2 | 4 | 28 | Input endpoint_4 |
| 2 | 5 | 2A | Input endpoint_5 |
| 2 | 6 | 2C | Input endpoint_6 |
| 2 | 7 | 2E | Input endpoint_7 |
| 3 | 0 | 30 | STPOW packet received |
| 3 | 1 | 32 | SETUP packet received |
| 3 | 2 | 34 | RESERVED |
| 3 | 3 | 36 | RESERVED |
| 3 | 4 | 38 | RESR interrupt |
| 3 | 5 | 3A | SUSR interrupt |
| 3 | 6 | 3C | RSTR interrupt |
| 3 | 7 | 3E | Wake-up interrupt |
| 4 | 0 | 40 | I ² C TxE interrupt |
| 4 | 1 | 42 | I ² C RxF interrupt |
| 4 | 2 | 44 | Input endpoint_0 |
| 4 | 3 | 46 | Output endpoint_0 |
| 4 | 4–7 | 48–4E | <i>Not used</i> |
| 5 | 0 | 50 | UART1-status interrupt |
| 5 | 1 | 52 | UART1-modem interrupt |
| 5 | 2 | 54 | UART2-status interrupt |
| 5 | 3 | 56 | UART2-modem interrupt |
| 5 | 4–7 | 58–5E | <i>Not used</i> |
| 6 | 0 | 60 | UART1-RxF interrupt |
| 6 | 1 | 62 | UART1-TxE interrupt |
| 6 | 2 | 64 | UART2-RxF interrupt |
| 6 | 3 | 66 | UART2-TxE interrupt |
| 6 | 4–7 | 68–6E | <i>Not used</i> |

Table 9–2. Vector Interrupt Values (Continued)

| G[3:0] (Hex) | I[2:0] (Hex) | VECTOR (Hex) | INTERRUPT SOURCE |
|-----------------|-----------------|-----------------|--------------------|
| 7 | 0 | 70 | PP: RxF interrupt |
| 7 | 1 | 72 | PP: TxE interrupt |
| 7 | 2 | 74 | PP: FALT interrupt |
| 7 | 3 | 76 | PP: ACK interrupt |
| 7 | 4 | 78 | PP: PER interrupt |
| 7 | 5–7 | 7A–7E | <i>Not used</i> |
| 8 | 0 | 80 | DMA1 interrupt |
| 8 | 1 | 82 | DMA2 interrupt |
| 8 | 2 | 84 | DMA3 interrupt |
| 8 | 3 | 86 | DMA4 interrupt |
| 8 | 4 | 88 | DMA5 interrupt |
| 8 | 5–7 | 8A–8E | <i>Not used</i> |
| 9–15 | X | 90–FE | <i>Not used</i> |

9.1.6 Logical Interrupt Connection Diagram (Internal/External)

Figure 9–1 shows the logical connection of the interrupt sources and its relation with $\overline{\text{XINT0}}$. The priority encoder generates an 8-bit vector, corresponding to 64 interrupt sources (not all are used). The interrupt priorities are hard wired. Vector 0x88 is the highest and 0x12 is the lowest.

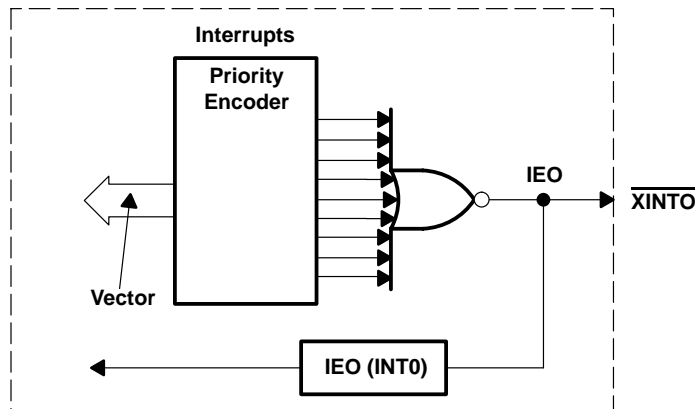


Figure 9–1. Internal Vector Interrupt

10 I²C Port

10.1 I²C Registers

10.1.1 I²CSTA: I²C Status and Control Register

This register is used to control the stop condition for read and write operations. In addition, it provides transmitter and receiver handshake signals with their respective interrupt-enable bits.

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RxF | RIE | ERR | 1/4 | TxE | TIE | SRD | SWR |
| R/W | R/W | R/C | R/W | R/O | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|------|-------|---|
| 0 | SWR | 0 | Stop write condition. This bit determines if the I ² C controller generates a stop condition when data from the I ² CDAO register are transmitted to an external device. SWR = 0 Stop condition is <i>not</i> generated when data from the I ² CDAO register are shifted out to an external device. SWR = 1 Stop condition is generated when data from the I ² CDAO register are shifted out to an external device. |
| 1 | SRD | 0 | Stop read condition. This bit determines if the I ² C controller generates a stop condition when data are received and loaded into the I ² CDAI register. SRD = 0 Stop condition is <i>not</i> generated when data from SDA line is shifted into I ² CDAI register. SRD = 1 Stop condition is generated when data from SDA line are shifted into I ² CDAI register. |
| 2 | TIE | 0 | I ² C transmitter empty interrupt enable. TIE = 0 Interrupt disable TIE = 1 Interrupt enable |
| 3 | TxE | 1 | I ² C transmitter empty. This bit indicates that data can be written to the transmitter. It can be used for polling or it can generate an interrupt. TxE = 0 Transmitter is full. This bit is cleared when the MCU writes a byte to the I ² CDAO register. TxE = 1 Transmitter is empty. The I ² C controller sets this bit when the contents of the I ² CDAO register are copied to the SDA shift register. |
| 4 | 1/4 | 0 | Bus speed selection. 1/4 = 0 100-kHz bus speed 1/4 = 1 400-kHz bus speed |
| 5 | ERR | 0 | Bus error condition. This bit is set by the hardware when the device does not respond. It is cleared by the MCU. ERR = 0 No bus error ERR = 1 Bus error condition has been detected. Clears when the MCU writes a 1. Writing a 0 has no effect. |
| 6 | RIE | 0 | I ² C receiver ready interrupt enable. RIE = 0 Interrupt disable RIE = 1 Interrupt enable |
| 7 | RxF | 0 | I ² C receiver full. This bit indicates that the receiver contains new data. It can be used for polling or it can generate an interrupt. RxF = 0 Receiver is empty. This bit is cleared when the MCU reads the I ² CDAI register. RxF = 1 Receiver contains new data. This bit is set by the I ² C controller when the received serial data has been loaded into the I ² CDAI register. |

10.1.2 I²CADR: I²C Address Register

This register holds the device address and the read/write command bit.

| | | | | | | | |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| A ₆ | A ₅ | A ₄ | A ₃ | A ₂ | A ₁ | A ₀ | R/W |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 0 | R/W | 0 | Read/write command bit R/W = 0 Write operation R/W = 1 Read operation |
| 7–1 | A[6:0] | 0h | Seven address bits for device addressing |

10.1.3 I²CDAI: I²C Data-Input Register

This register holds the received data from an external device.

| | | | | | | | |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D ₇ | D ₆ | D ₅ | D ₄ | D ₃ | D ₂ | D ₁ | D ₀ |
| R/O | R/O | R/O | R/O | R/O | R/O | R/O | R/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|--|
| 7–0 | D[7:0] | 0 | 8-bit input data from an I ² C device |

10.1.4 I²CDAO: I²C Data-Output Register

This register holds the data to be transmitted to an external device. Writing to this register starts the transfer on the SDA line.

| | | | | | | | |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D ₇ | D ₆ | D ₅ | D ₄ | D ₃ | D ₂ | D ₁ | D ₀ |
| W/O | W/O | W/O | W/O | W/O | W/O | W/O | W/O |

| BIT | NAME | RESET | FUNCTION |
|-----|--------|-------|---|
| 7–0 | D[7:0] | 0 | 8-bit output data to an I ² C device |

10.2 Random-Read Operation

A random read requires a dummy byte-write sequence to load in the data word address. Once the device-address word and the data-word address are clocked out and acknowledged by the device, the MCU starts a current-address sequence. The following describes the sequence of events to accomplish this transaction:

10.2.1 Device Address + EEPROM [High-Byte]

1. MCU sets I²CSTA[SRD] = 0. This forces the I²C controller *not* to generate a stop condition after the contents of the I²CDAI register are received.
2. MCU sets I²CSTA[SWR] = 0. This forces the I²C controller *not* to generate a stop condition after the contents of the I²CDAO register are transmitted.
3. MCU writes the device address (R/W bit = 0) to the I²CADR register (write operation).
4. MCU writes the high byte of the EEPROM address into the I²CDAO register (this starts the transfer on the SDA line).
5. TxE bit in the I²CSTA register is cleared (indicates busy).

6. The content of the I²CADR register is transmitted to EEPROM (preceded by start condition on SDA).
7. The contents of the I²CDAO register are transmitted to EEPROM (EEPROM address).
8. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register has been transmitted.
9. Stop condition is *not* generated.

10.2.2 EEPROM [Low Byte]

1. MCU writes the low-byte of the EEPROM address into the I²CDAO register.
2. TxE bit in the I²CSTA register is cleared (indicates busy).
3. The contents of the I²CDAO register are transmitted to the device (EEPROM address).
4. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register has been transmitted.
5. This completes the dummy write operation. At this point, the E2ROM address is set and the MCU can do either a single- or a sequential-read operation.

10.3 Current-Address Read Operation

Once the EEPROM address is set, the MCU can read a single byte by executing the following steps:

1. MCU sets I²CSTA[SRD] = 1. This forces the I²C controller to generate a stop condition after the I²CDAI-register contents are received.
2. MCU writes the device address (R/W bit = 1) to the I²CADR register (read operation).
3. MCU writes a dummy byte to the I²CDAO register (this starts the transfer on the SDA line).
4. RxF bit in the I²CSTA register is cleared.
5. Contents of the I²CADR register are transmitted to the device (preceded by start condition on SDA).
6. Data from EEPROM are latched into the I²CDAI register (stop condition is transmitted).
7. RxF bit in the I²CSTA register is set and interrupts the MCU, indicating that the data are available.
8. MCU reads the I²CDAI register. This clears the RxF bit (I²CSTA[RxF] = 0).
9. End

10.4 Sequential Read Operation

Once the EEPROM address is set, the MCU can execute a sequential read operation by executing the following steps (this example illustrates a 32-byte sequential read):

10.4.1 Device Address

1. MCU sets I²CSTA[SRD] = 0. This forces the I²C controller *not* to generate a stop condition after the I²CDAI register contents are received.
2. MCU writes the device address (R/W bit = 1) to the I²CADR register (read operation).
3. MCU writes a dummy byte to the I²CDAO register (this starts the transfer on the SDA line).
4. RxF bit in the I²CSTA register is cleared.
5. The contents of the I²CADR register are transmitted to the device (preceded by start condition on SDA).

10.4.2 N-Byte Read (31 Bytes)

1. Data from the device are latched into the I²CDAI register (stop condition is *not* transmitted).
2. RxF bit in the I²CSTA register is set and interrupts the MCU, indicating that data are available.
3. MCU reads the I²CDAI register. This clears the RxF bit (I²CSTA[RxF] = 0).
4. This operation repeats 31 times.

10.4.3 Last-Byte Read (Byte 32)

1. MCU sets I²CSTA[SRD] = 1. This forces the I²C controller to generate a stop condition after the I²CDAI register contents are received.
2. Data from the device are latched into the I²CDAI register (stop condition is transmitted).
3. RxF bit in the I²CSTA register is set and interrupts the MCU, indicating that data are available.
4. MCU reads the I²CDAI register. This clears the RxF bit (I²CSTA[RxF] = 0).
5. End

10.5 Byte-Write Operation

10.5.1 Device Address + EEPROM [High Byte]

1. MCU sets I²CSTA[SWR] = 0. This forces the I²C controller *not* to generate a stop condition after the contents of the I²CDAO register are transmitted.
2. MCU writes the device address (R/W bit = 0) to the I²CADR register (write operation).
3. MCU writes the high byte of the EEPROM address into the I²CDAO register (this starts the transfer on the SDA line).
4. TxE bit in the I²CSTA register is cleared (indicates busy).
5. The contents of the I²CADR register are transmitted to the device (preceded by start condition on SDA).
6. The contents of the I²CDAO register are transmitted to the device (EEPROM high-address).
7. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.

10.5.2 EEPROM [Low Byte]

1. MCU writes the low byte of the EEPROM address into the I²CDAO register.
2. TxE bit in the I²CSTA register is cleared (indicating busy).
3. The contents of the I²CDAO register are transmitted to the device (EEPROM address).
4. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.

10.5.3 EEPROM [DATA]

1. MCU sets I²CSTA[SWR] = 1. This forces the I²C controller to generate a stop condition after the contents of I²CDAO register are transmitted.
2. The data to be written to EEPROM are written by the MCU into the I²CDAO register.
3. TxE bit in the I²CSTA register is cleared (indicates busy).

4. The contents of the I²CDAO register are transmitted to the device (EEPROM data).
5. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.
6. I²C controller generates a stop condition after the contents of the I²CDAO register are transmitted.
7. End

10.6 Page Write

The page-write operation is initiated in the same way as byte-write, with the exception that a stop condition is not generated after the first EEPROM [DATA] is transmitted. The following describes the sequence of writing 32-bytes in page mode:

10.6.1 Device Address + EEPROM [High Byte]

1. MCU sets I²CSTA[SWR] = 0. This forces the I²C controller *not* to generate a stop condition after the contents of the I²CDAO register are transmitted.
2. MCU writes the device address (R/W bit = 0) to the I²CADR register (write operation).
3. MCU writes the high byte of the EEPROM address into the I²CDAO register.
4. TxE bit in the I²CSTA register is cleared (indicating busy).
5. The contents of the I²CADR register are transmitted to the device (preceded by start condition on SDA).
6. The contents of the I²CDAO register are transmitted to the device (EEPROM address).
7. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.

10.6.2 EEPROM [Low Byte]

1. MCU writes the low byte of the EEPROM address into the I²CDAO register.
2. TxE bit in the I²CSTA register is cleared (indicates busy).
3. The contents of the I²CDAO register are transmitted to the device (EEPROM address).
4. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.

10.6.3 EEPROM [DATA] - 31 Bytes

1. The data to be written to the EEPROM are written by the MCU into the I²CDAO register.
2. TxE bit in the I²CSTA register is cleared (indicates busy).
3. The contents of the I²CDAO register are transmitted to the device (EEPROM data).
4. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.
5. This operation repeats 31 times.

10.6.4 EEPROM [DATA] - Last Byte

1. MCU sets I²CSTA[SWR] = 1. This forces the I²C controller to generate a stop condition after the contents of the I²CDAO register are transmitted.
2. MCU writes the last data byte to be written to the EEPROM, into I²CDAO register.

3. TxE bit in the I²CSTA register is cleared (indicates busy).
4. The contents of the I²CDAO register are transmitted to EEPROM (EEPROM data).
5. TxE bit in the I²CSTA register is set and interrupts the MCU, indicating that the I²CDAO register contents have been transmitted.
6. I²C controller generates a stop condition after the contents of I²CDAO register are transmitted.
7. End of 32-byte page-write operation

11 Electrical Specifications

11.1 Absolute Maximum Ratings Over Operating Free-Air Temperature Range (unless otherwise noted)[†]

| | |
|--|----------------------------|
| Supply voltage range, V_{CC} (see Note 1) | −0.5 V to 3.6 V |
| Input voltage range, V_I | −0.5 V to $V_{CC} + 0.5$ V |
| Output voltage range, V_O | −0.5 V to $V_{CC} + 0.5$ V |
| Input clamp current, I_{IK} , ($V_I < 0$ V or $V_I > V_{CC}$) | ±20 mA |
| Output clamp current, I_{OK} , ($V_O < 0$ V or $V_O > V_{CC}$) | ±20 mA |
| Storage temperature range, T_{stg} | −65°C to 150°C |
| Operating free-air temperature range | 0°C to 70°C |

[†] Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltage levels are with respect to GND.

11.2 Recommended Operating Conditions

| | MIN | NOM | MAX | UNIT |
|--|----------|-----|----------|------|
| Supply voltage, V_{CC} | 3 | 3.3 | 3.6 | V |
| Input voltage, TTL/LVCMOS, V_I | 0 | | V_{CC} | V |
| Output voltage, TTL/LVCMOS, V_O | 0 | | V_{CC} | V |
| High-level input voltage, signal-ended receiver, $V_{IH(REC)}$ | 2 | | V_{CC} | V |
| Low-level input voltage, signal-ended receiver, $V_{IL(REC)}$ | | | 0.8 | V |
| High-level input voltage, TTL/LVCMOS, $V_{IH(TTL)}$ | 2 | | V_{CC} | V |
| Low-level input voltage, TTL/LVCMOS, $V_{IL(TTL)}$ | 0 | | 0.8 | V |
| Operating free-air temperature, T_A | 0 | | 70 | °C |
| External series, differential driver resistor, $R_{(DRV)}$ | 22 (−5%) | | 22 (5%) | Ω |
| Operating (dc differential driver) high speed mode, $f_{(OPRH)}$ | | | 12 | Mb/s |
| Operating (dc differential driver) low speed mode, $f_{(OPRL)}$ | | | 1.5 | Mb/s |
| Common mode, input range, differential receiver, $V_{(ICR)}$ | 0.8 | | 2.5 | V |
| Input transition times, t_t , TTL/LVCMOS | 0 | | 25 | ns |
| Junction temperature range, T_J | 0 | | 115 | °C |

11.3 Electrical Characteristics Over Recommended Ranges of Operating Free-Air Temperature and Supply Voltage (unless otherwise noted)

| PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|----------------------|--|-----------------|---|-----------------------|------|
| V _{OH} | High-level output voltage | TTL/LVCMOS | I _{OH} = −4 mA | V _{CC} − 0.5 | V |
| | | USB data lines | R(DRV) = 15 kΩ to GND | 2.8 | |
| | | | I _{OH} = −12 mA (without R(DRV)) | V _{CC} − 0.5 | |
| V _{OL} | Low-level output voltage | TTL/LVCMOS | I _{OL} = 4 mA | 0.5 | V |
| | | USB data lines | R(DRV) = 1.5 kΩ to 3.6 V | 0.3 | |
| | | | I _{OL} = 12 mA (without R(DRV)) | 0.5 | |
| V _{IT+} | Positive input threshold voltage | TTL/LVCMOS | | 1.8 | V |
| | | Single-ended | 0.8 V ≤ V _{ICR} ≤ 2.5 V | 1.8 | V |
| V _{IT−} | Negative-input threshold voltage | TTL/LVCMOS | | 0.8 | V |
| | | Single-ended | 0.8 V ≤ V _{ICR} ≤ 2.5 V | 1 | V |
| V _{hys} | Input hysteresis† (V _{T+} − V _{T−}) | TTL/LVCMOS | | 0.3 | 0.7 |
| | | Single-ended | 0.8 V ≤ V _{ICR} ≤ 2.5 V | 300 | 500 |
| I _{OZ} | High-impedance output current | TTL/LVCMOS | V = V _{CC} or GND‡ | ±10 | μA |
| | | USB data lines | 0 V ≤ V _O ≤ V _{CC} | ±10 | μA |
| I _{IL} | Low-level input current | TTL/LVCMOS | V _I = GND | −1 | μA |
| I _{IH} | High-level input current | TTL/LVCMOS | V _I = V _{CC} | 1 | μA |
| z _O (DRV) | Driver output impedance | USB data lines | Static V _{OH} or V _{OL} | 7.1 | 19.9 |
| V _{ID} | Differential input voltage | USB data lines | 0.8 V ≤ V _{ICR} ≤ 2.5 V | 0.2 | V |

† Applies for input buffers with hysteresis

‡ Applies for open drain buffers

11.4 Differential Driver Switching Characteristics Over Recommended Ranges of Operating Free-Air Temperature and Supply Voltage, C_L = 50 pF (unless otherwise noted)

11.4.1 Full-Speed Mode

| PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|----------------------|-----------------------------------|---|-----|------|------|
| t _r | Transition rise time for DP or DM | See Figure 11–1 and Figure 11–2 | 4 | 20 | ns |
| t _f | Transition fall time for DP or DM | See Figure 11–1 and Figure 11–2 | 4 | 20 | ns |
| t(RFM) | Rise/fall time matching§ | (t _r /t _f) × 100 | 90% | 110% | |
| V _O (CRS) | Signal crossover output voltage§ | | 1.3 | 2.0 | V |

§ Characterized only. Limits are approved by design and are not production tested.

11.4.2 Low-Speed Mode

| PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|----------------------|------------------------------------|--|-----|------|------|
| t _r | Transition rise time for DP or DM§ | C _L = 200 pF to 600 pF, see Figure 11–1 and Figure 11–2 | 75 | 300 | ns |
| t _f | Transition fall time for DP or DM§ | C _L = 200 pF to 600 pF, see Figure 11–1 and Figure 11–2 | 75 | 300 | ns |
| t(RFM) | Rise/fall time matching§ | (t _r /t _f) × 100 | 80% | 120% | |
| V _O (CRS) | Signal crossover output voltage§ | C _L = 200 pF to 600 pF | 1.3 | 2.0 | V |

§ Characterized only. Limits are approved by design and are not production tested.

11.5 Current Consumption, T_A = 25°C, V_{CC} = V_{CCS} = 3.3 V ±5%, V_{SS} = 0

| OPERATION | | MIN | TYP | MAX | UNIT |
|------------------|--|-----|-----|-----|------|
| Normal operation | | | 45 | | mA |
| Suspend mode | | | 50 | | μA |

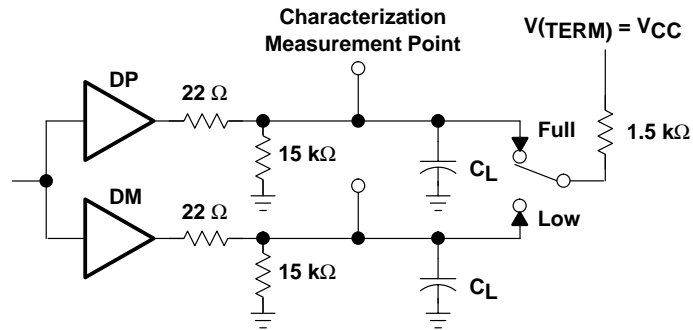
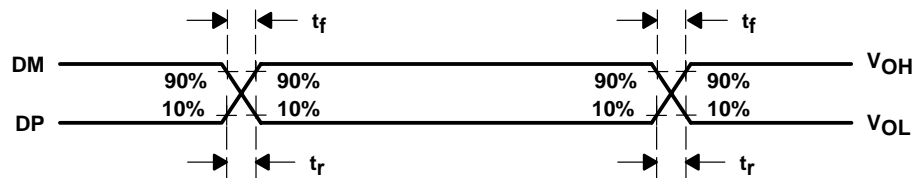


Figure 11–1. Differential Driver Switching Load



NOTE: The t_r/t_f ratio is measured as $t_r(\text{DP})/t_f(\text{DM})$ and $t_r(\text{DM})/t_f(\text{DP})$ at each crossover point.

Figure 11–2. Differential Driver Timing Waveforms

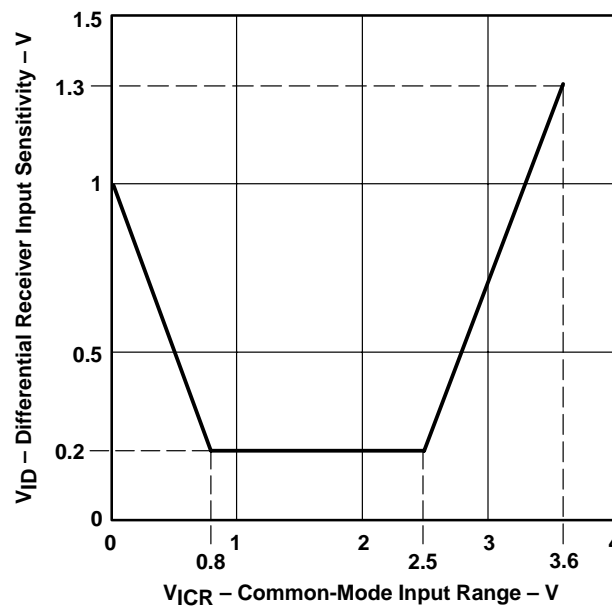


Figure 11–3. Differential Receiver Input Sensitivity vs Common-Mode Input Range

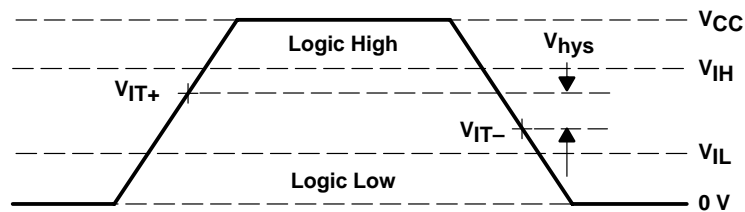


Figure 11–4. Single-Ended Receiver Input Signal Parameter Definitions

12 Application

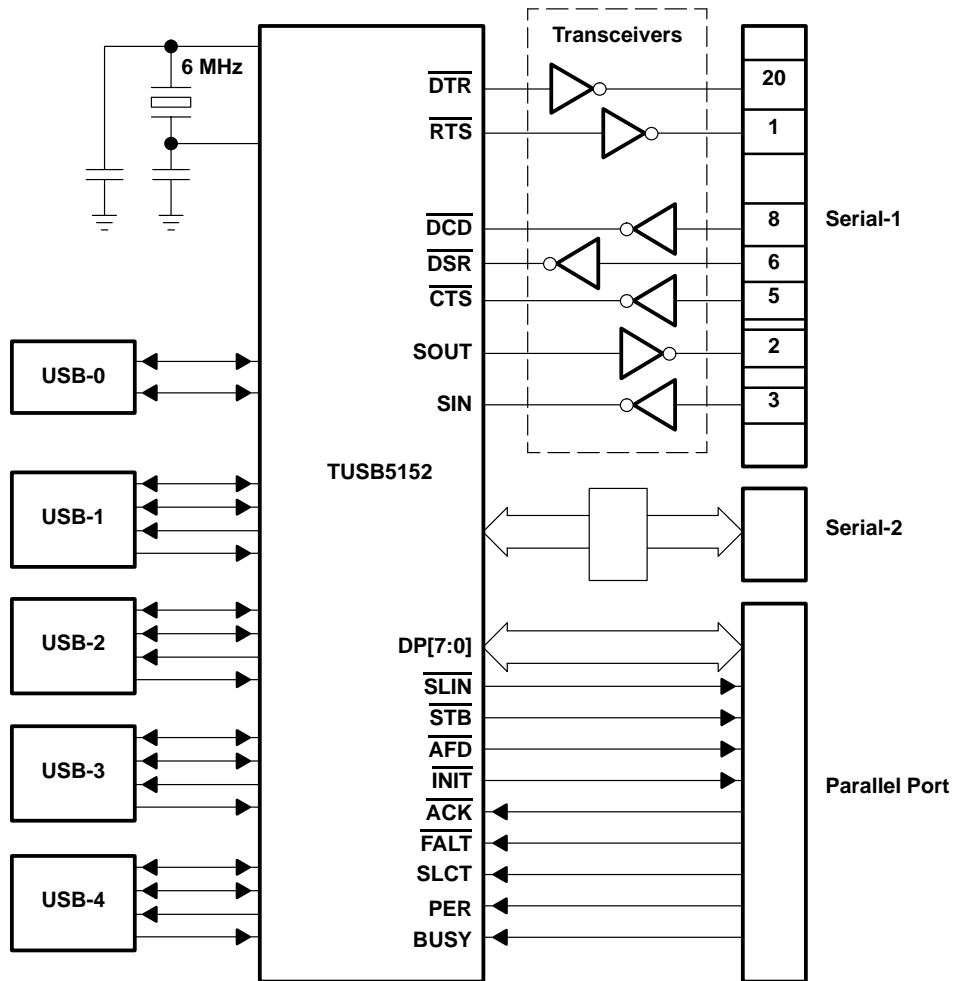


Figure 12–1. 4-Port Hub, Two Serial- and One Parallel-Port Implementation

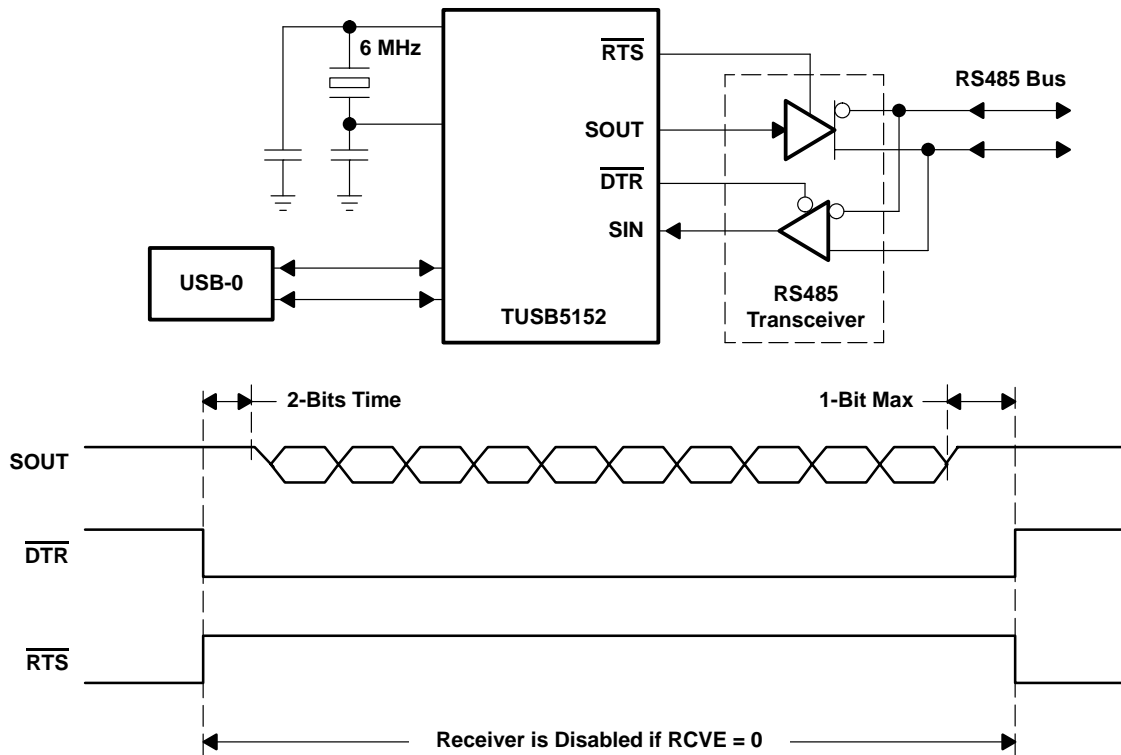


Figure 12-2. RS485 Bus Implementation

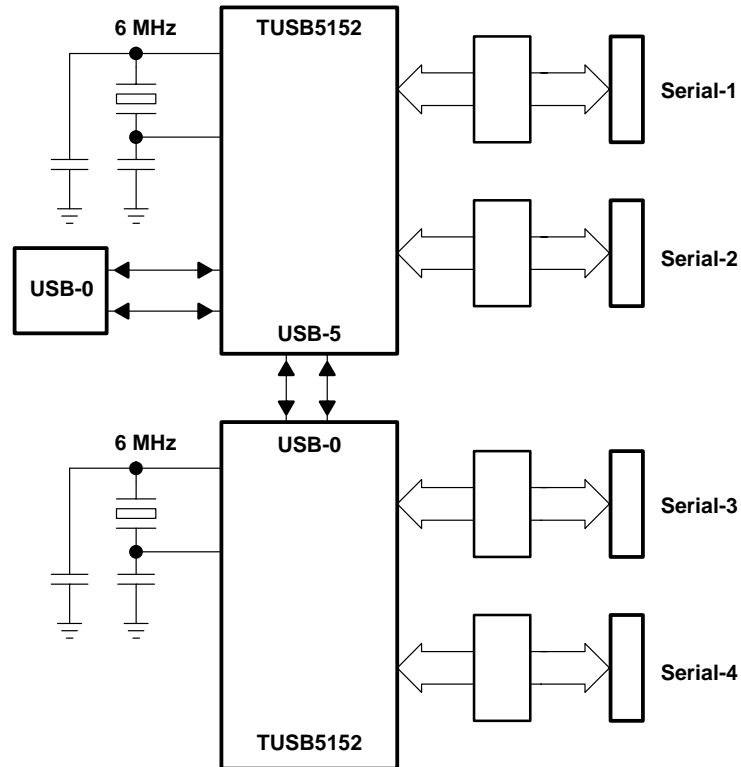


Figure 12-3. Quad UART Implementation

13 Boot Code

Boot code copies predefined USB descriptors to shared RAM. It then checks if EEPROM is present on the I²C port. If a valid signature is found, boot code reads in the data-type field to determine if the data section is application code or USB device information. If it is application code, boot code downloads the code to external data space. Once code is loaded and the checksum is correct, boot code then releases control to the application code. If the data contain USB device information, boot code reads in the data and, if the checksum is correct, copies it to hub registers and the embedded function device descriptor. Otherwise, it restores predefined settings to hub registers and the device descriptor.

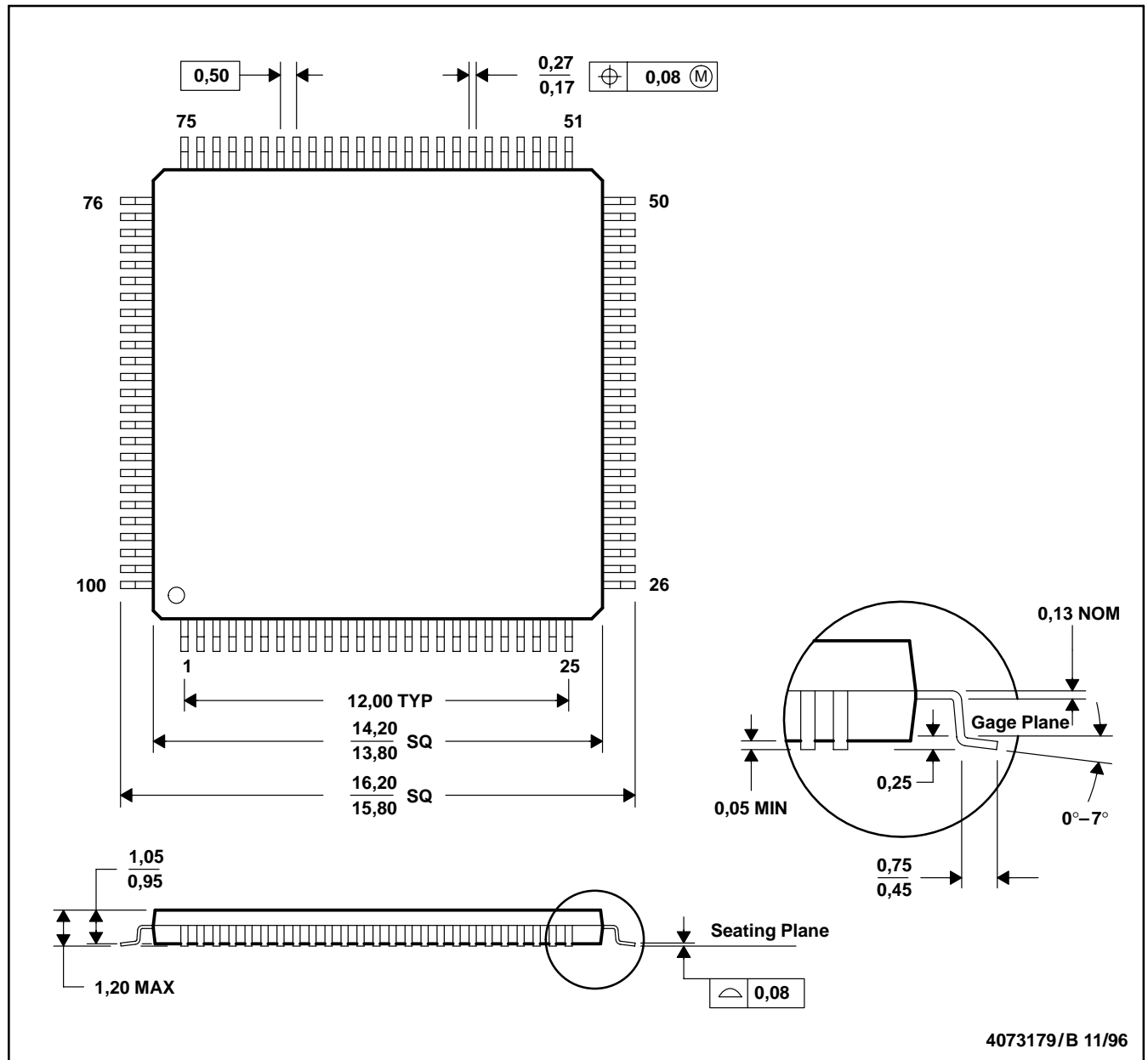
Boot code waits for the firmware to be downloaded from the host. Once the firmware is loaded, boot code disconnects from the USB and releases control to the firmware. For more information, see application notes for this product.

14 Mechanical Information

The TUSB5152 is packaged in a 100-pin PZT plastic quad flatpack. The following shows the mechanical dimensions for the PZT package.

PZT (S-PQFP-G100)

PLASTIC QUAD FLATPACK



- NOTES:
- A. All linear dimensions are in millimeters.
 - B. This drawing is subject to change without notice.
 - C. Falls within JEDEC MS-026



PRODUCT FOLDER | PRODUCT INFO: [FEATURES](#) | [DESCRIPTION](#) | [DATASHEETS](#) | [PRICING/AVAILABILITY/PKG](#) | [USER MANUALS](#) | [RELATED DOCUMENTS](#)

PRODUCT SUPPORT: [APPLICATIONS](#)

TUSB5152, 5-Port Hub with Integrated Bridge to IEEE 1284 Port and Two Serial Ports

DEVICE STATUS: **ACTIVE**

| PARAMETER NAME | TUSB5152 |
|-----------------------|----------|
| Supply Voltage(s) (V) | 3.3 |
| Number of USB Ports | 5 |
| I2C | Yes |

FEATURES

[▲Back to Top](#)

- Fully compliant with USB release 1.1 specifications
- Supports 12-Mbps USB data rate (full speed)
- Supports USB suspend and resume operations
- One upstream port and five downstream ports (full and low speed)
- Can support a total of 8 input endpoints and 8 output endpoints
- Dual-enhanced UARTs
- Integrated 8052 microcontroller with
 - 256 × 8 RAM for internal data
 - 6K × 8 ROM (with USB and I²C boot loader)
 - 16K × 8 RAM for code space. Totally loadable from host or I²C port.
 - 2K × 8 synchronous RAM used for data buffers and EDBs
 - Two 8052 GPIO ports, Port-1 and Port-3
 - Master I²C controller for external device accesses.
- Supports external microcontroller interface for development ease
- Built-in five-channel DMA controller for USB/UART/P-port bulk I/O
- Operates from a 6-MHz crystal. On-chip PLL generates 48/24 MHz and 7.384615 MHz for internal baud-rate generator
- Power-down mode
- Available in 100-pin TQFP
- 3.3/5-V operation
- Enhanced UART Features
 - Software/hardware flow control
 - Programmable Xon/Xoff characters
 - Programmable auto-RTS\DTR\ and auto-CTS\DSR\
 - Automatic RS485-bus transceiver control, with and without echo
 - Software selectable baud rate from 50 to 460,800 baud
 - Programmable serial-interface characteristics
 - 5-, 6-, 7- or 8-bit characters
 - Even, odd, or no parity-bit generation and detection
 - 1, 1.5 or 2 stop-bit generation
 - Line break generation and detection
 - Internal test and loop-back capabilities
 - Modem-control functions (CTS\, RTS\, DSR\, DTR\, RI\, and DCD\)
 - Internal diagnostics capability
 - Loopback control for communications link-fault isolation
 - Break, parity, overrun, framing-error simulations

DESCRIPTION

[▲Back to Top](#)

This controller provides bridging between the USB, the dual-enhanced UARTs, and the IEEE-1284 bidirectional parallel port. This device contains all the necessary logic to communicate with the host computer using the USB bus. In addition to the USB host interface, the device contains a 5-port hub that can be used for USB expansion. The internal 8052 microcontroller contains 16K RAM that can be loaded from the host. All the device functions, such as the USB command decoding, UARTs/1284-port setup, and error reporting are managed by the internal microcontroller unit (MCU) firmware.

TECHNICAL DOCUMENTS

[▲Back to Top](#)

To view the following documents, [Acrobat Reader 4.0](#) is required.
To download a document to your hard drive, right-click on the link and choose 'Save'.

DATASHEET

[▲Back to Top](#)

Full datasheet in Acrobat PDF: [tusb5152.pdf](#) (325 KB,Rev.B) (Updated: 11/13/2001)

RELATED DOCUMENTS

[▲Back to Top](#)

- [Enhanced Plastic Portfolio Brochure](#) (SGZB004, 385 KB - Updated: 08/19/2002)
- [Military Analog Selection Guide](#) (SGLB002, 318 KB - Updated: 11/09/2000)
- [Military Semiconductors Selection Guide 2002 \(Rev. B\)](#) (SGYC003B, 1648 KB - Updated: 04/22/2002)
- [TUSB2136/TUSB3210/TUSB5052/TUSB5152 USB Firmware Programming Flow 8052 Embedded](#) (SLLU020, 155 KB - Updated: 10/04/2000)
- [USB Solutions Guide 1Q 2002](#) (SLYB094, 748 KB - Updated: 12/05/2001)

USER MANUALS

[▲Back to Top](#)

- [TUSB5152 Bootcode Document for USB to 2-Serial + 1-Parallel Controller](#) (SLLU021 - Updated: 11/09/2001)

PRICING/AVAILABILITY/PKG

[▲Back to Top](#)

| DEVICE INFORMATION | | | | | | | TI INVENTORY STATUS AS OF 4:00 PM GMT, 26 Sep 2002 | | | REPORTED DISTRIBUTOR INVENTORY AS OF 4:00 PM GMT, 26 Sep 2002 | | |
|---------------------|--------|-------------------------------------|-----------|-------------------------------|------------------------------------|--------------------|---|-------------------------|---------------|--|----------|----------|
| ORDERABLE DEVICE | STATUS | PACKAGE TYPE PINS | TEMP (°C) | PRODUCT CONTENT | BUDGETARY PRICING QTY \$US | STD PACK QTY | IN STOCK | IN PROGRESS QTY DATE | LEAD TIME | DISTRIBUTOR COMPANY REGION | IN STOCK | PURCHASE |
| TUSB5152PZ | ACTIVE | LQFP (PZ) 100 | 0 TO 70 | View Contents | 1KU 10.62 | 90 | N/A* | | Not Available | | | |

Table Data Updated on: 9/26/2002