# MCS7840
# USB 2.0 to Quad Serial Controller

**Features**

- USB-2.0 Device Controller
- On-Chip USB-2.0 PHY
- On-Chip Voltage Regulators
- Four 16c450/16c550 compatible UARTs
- Supports SIR IrDA Mode on any/all ports
- Supports RS-232, RS-485 and RS-422 Serial Ports
- 5, 6, 7 & 8-bit Serial Data support
- Hardware and Software Flow Control
- Serial Port speeds from 50 bps to 6 Mbps
- Custom BAUD Rates supported through external clock and/or by programming the internal PLL
- On-Chip 512-Byte FIFOs for upstream and downstream data transfers for each Serial Port
- Supports Remote Wakeup and Power Management features
- Serial Port Transceiver Shut-Down support
- Two-Wire $I^2C$ Interface for EEPROM
- EEPROM read/write through USB
- iSerial feature support with EEPROM
- One Bi-directional multi-function GPIO
- On-Chip buffers for Serial Port signals to operate without external Transceivers over short cable lengths
- Bus-Powered Device

**Applications**

- Serial Attached Devices
- Modems, Serial Mouse, Generic Serial Devices
- Serial-Port Server
- Data Acquisition System
- POS Terminal & Industrial PC

**Application Note**

- AN-7840

**Evaluation Board**

- MCS7840-EVB

**Package**

- 64-pin LQFP Package

**General Description**

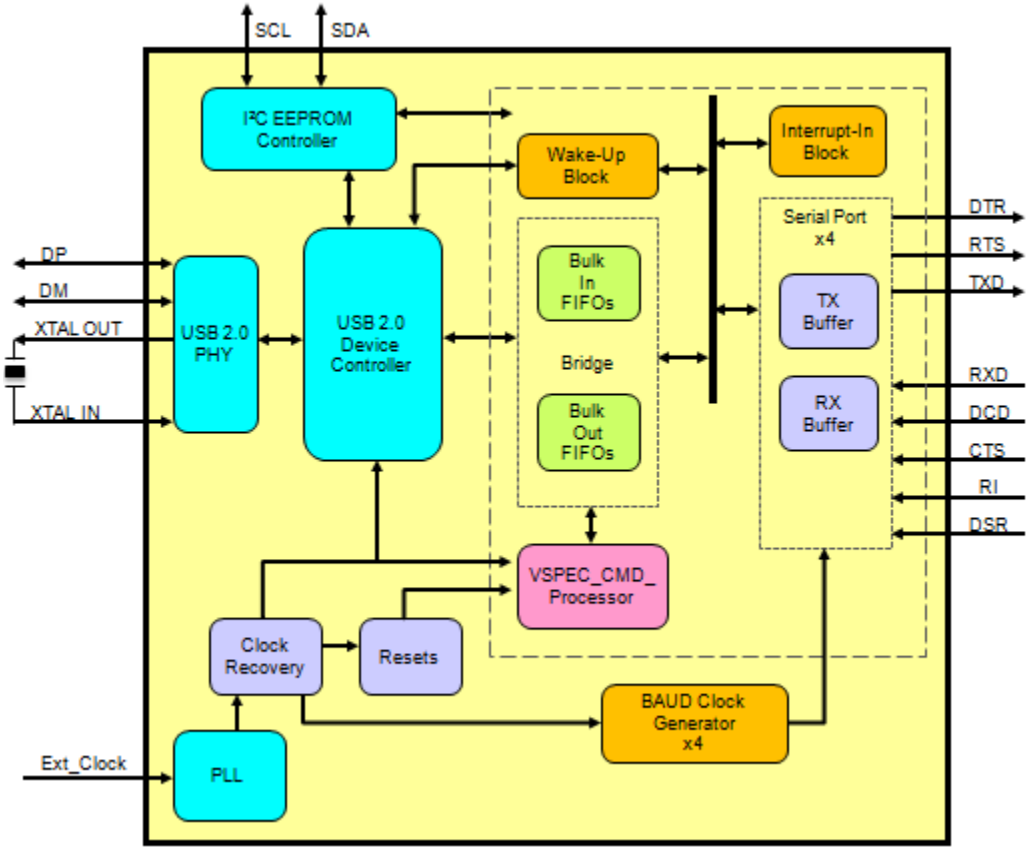
The MCS7840 is a USB-2.0 to Quad-Serial Port device. It has been developed to connect a wide range of standard serial devices to a USB host.

The MCS7840 has a USB Device Controller connected to four (4) individual UARTs.

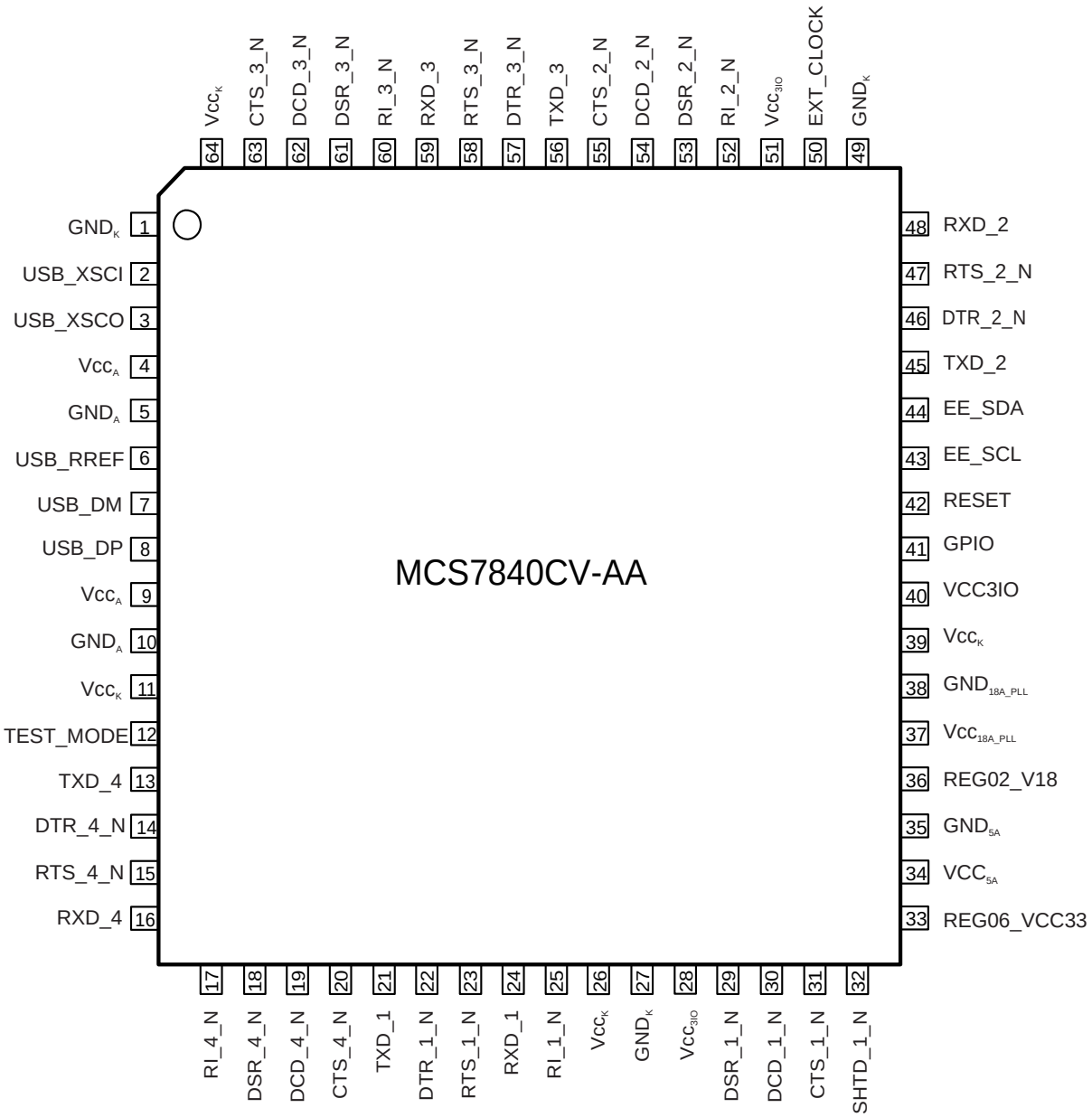
Support for the following serial communication programs is included:
HyperTerminal, PComm, Windows direct connection, Windows dial-up connection through modem, Networking over IrDA and Windows direct connection over IrDA, Minicom.

| Ordering Information | | |
|---|---|---|
| Commercial Grade (0 °C to +70 °C) | | |
| MCS7840CV-AA | 64-LQFP | RoHS |

## Block Diagram



## Pin-Out Diagram



**Pin Assignments**

| Pin | Name | Type | Functional Description |
|---|---|---|---|
| 1 | $GND_K$ | Power | Core Ground |
| 2 | USB_XSCI | Input | Crystal Oscillator Input |
| 3 | USB_XSCO | Output | Crystal Oscillator Output |
| 4 | $Vcc_A$ | Power | Power Pin (A3V3) |
| 5 | $GND_A$ | Power | Analog Ground |
| 6 | USB_RREF | Input | External Reference Resistor (12.1 KΩ, 1%)<br>Connect resistor to Analog GND. |
| 7 | USB_DM | I/O | USB D- Signal |
| 8 | USB_DP | I/O | USB D+ Signal |
| 9 | $Vcc_A$ | Power | Power Pin (A3V3) |
| 10 | $GND_A$ | Power | Analog Ground |
| 11 | $Vcc_K$ | Power | Power Pin (1.8V) |
| 12 | TEST_MODE | Input | Test Mode Pin, (active high). Default = Low (0)<br>When TEST_MODE = 1, PLL, Core, and SCAN/BIST/ Memory BIST testing can be performed.<br>Set TEST_MODE = 0 for normal operation. |
| 13 | TXD_4 | Output | Serial Port 4 Transmit Data out to transceiver or IrDA data out to IR LED |
| 14 | DTR_4_N | Output | Serial Port 4 Data Terminal Ready (in serial protocol), active low. |
| 15 | RTS_4_N | Output | Serial Port 4 Request To Send (in serial protocol), active low. |
| 16 | RXD_4 | Input | Serial Port 4 Serial Receive Data in from transceiver or IrDA data in from IrDA detector. |
| 17 | RI_4_N | Input | Serial Port 4 Ring Indicator, active low |
| 18 | DSR_4_N | Input | Serial Port 4 Data Set Ready (in serial protocol), active low |
| 19 | DCD_4_N | Input | Serial Port 4 Data Carrier Detect (in serial protocol), active low |
| 20 | CTS_4_N | Input | Serial Port 4 Clear To Send (in serial protocol), active low |
| 21 | TXD_1 | Output | Serial Port 1 Transmit Data out to transceiver, or IrDA data out to IR LED |
| 22 | DTR_1_N | Output | Serial Port 1 Data Terminal Ready (in serial protocol), active low. |

| Pin | Name | Type | Functional Description |
|---|---|---|---|
| 23 | RTS_1_N | Output | Serial Port 1 Request To Send (in serial protocol), active low. |
| 24 | RXD_1 | Input | Serial Port 1 Serial Receive Data in from transceiver, or IrDA data in from IrDA detector. |
| 25 | RI_1_N | Input | Serial Port 1 Ring Indicator, active low. |
| 26 | $Vcc_K$ | Power | Power Pin (1.8V) |
| 27 | $GND_K$ | Power | Core Ground |
| 28 | $Vcc_{3IO}$ | Power | Power Pin (D3V3) |
| 29 | DSR_1_N | Input | Serial Port 1 Data Set Ready (in serial protocol), active low |
| 30 | DCD_1_N | Input | Serial Port 1 Data Carrier Detect (in serial protocol), active low |
| 31 | CTS_1_N | Input | Serial Port 1 Clear To Send (in serial protocol), active low |
| 32 | SHTD_1_N | Output | Shut Down External Serial Transceiver during normal operation, active low by default, can be configured active high by using DCR setting. |
| 33 | REG06_VCC33 | Power | Power Pin (3.3V OUTPUT) |
| 34 | $Vcc_{5A}$ | Power | Power Pin (5V INPUT) |
| 35 | $GND_{5A}$ | Power | Ground Pin for 5V Input |
| 36 | REG02_V18 | Power | Power Pin (1.8V OUTPUT) |
| 37 | $Vcc_{18A\_PLL}$ | Power | PLL Power (1.8V) |
| 38 | $GND_{18A\_PLL}$ | Power | PLL Ground |
| 39 | $Vcc_K$ | Power | Power Pin (1.8V) |
| 40 | VCC3IO | Power | Power pin D3V3. |
| 41 | GPIO | I/O | GPIO_MODE - Bidirectional GPIO bit. The direction (Input or Output) is controlled by the DCR for Serial Port #1. |
| 42 | RESET | I | Power-On Reset signal (active high). |
| 43 | EE_SCL | I/O | 2-Wire EEPROM Clock. Default = High (1) |
| 44 | EE_SDA | I/O | 2-Wire EEPROM Data in/out. Default = High (1) |
| 45 | TXD_2 | Output | Serial Port 2 Transmit Data out to transceiver, or IrDA data out to IR LED |
| 46 | DTR_2_N | Output | Serial Port 2 Data Terminal Ready (in serial protocol), active low. |

| Pin | Name | Type | Functional Description |
|---|---|---|---|
| 47 | RTS_2_N | Output | Serial Port 2 Request To Send (in serial protocol), active low. |
| 48 | RXD_2 | Input | Serial Port 2 Serial Receive Data in from transceiver or IrDA data in from IrDA detector. |
| 49 | $GND_K$ | Power | Core Ground. |
| 50 | EXT_CLOCK | Input | Input Clock from external world. In normal operation mode, clock can be supplied to serial ports and used for custom BAUD Rate of user's choice. In test mode, clock will be the test clock input from external world. |
| 51 | $Vcc_{3IO}$ | Power | Power Pin (D3V3). |
| 52 | RI_2_N | Input | Serial Port 2 Ring Indicator, active low. |
| 53 | DSR_2_N | Input | Serial Port 2 Data Set Ready (in serial protocol), active low. |
| 54 | DCD_2_N | Input | Serial Port 2 Data Carrier Detect (in serial protocol), active low. |
| 55 | CTS_2_N | Input | Serial Port 2 Clear To Send (in serial protocol), active low. |
| 56 | TXD_3 | Output | Serial Port 3 Transmit Data out to transceiver, or IrDA data out to IR LED. |
| 57 | DTR_3_N | Output | Serial Port 3 Data Terminal Ready (in serial protocol), active low. |
| 58 | RTS_3_N | Output | Serial Port 3 Request To Send (in serial protocol), active low. |
| 59 | RXD_3 | Input | Serial Port 3 Serial Receive Data in from transceiver, or IrDA data in from IrDA detector. |
| 60 | RI_3_N | Input | Serial Port 3 Ring Indicator, active low. |
| 61 | DSR_3_N | Input | Serial Port 3 Data Set Ready (in serial protocol), active low. |
| 62 | DCD_3_N | Input | Serial Port 3 Data Carrier Detect (in serial protocol), active low. |
| 63 | CTS_3_N | Input | Serial Port 3 Clear To Send (in serial protocol), active low. |
| 64 | $Vcc_K$ | Power | Power Pin (1.8V) |

## Functional Block Descriptions

### Internal Regulators
An internal DC-DC Regulator is provided to convert 5V to 1.8V for Core Logic. An additional regulator is provided to convert the 5V input to 3.3V for I/O functions. These regulators eliminate the need for external voltage sources.

### *USB-2.0 PHY*
This is the physical layer of the USB interface. The USB-2.0 PHY communicates with the USB-2.0 Device Controller logic through a UTMI interface to send/receive data on the USB bus.

### *USB-2.0 Device Controller*
The USB-2.0 Device Controller interfaces to the internal bridge and communicates with the serial ports through the bridge logic. The device controller logic is connected to a physical layer USB-2.0 PHY which provides the USB bus interface for the chip. The device controller responds to standard as well as vendor specific requests from USB-2.0 and USB-1.1 Hosts.

### *Bridge*
The bridge logic controls traffic between the USB-2.0 Device Controller and the Serial Port Controllers. The bridge logic has synchronous RAM memories with ping-pong FIFO control logic to buffer data in either direction (Bulk-In and Bulk-Out) and send it to the other side without loss. Control logic prevents overflow or underflow conditions in the memory.

### *UART / Serial Port Controllers*
The Serial Port Controllers are linked to the bridge and send/receive data from the bridge interface. Each serial port controller has register logic controlling BAUD rates (50 bps – 6 Mbps), stop-bits, and parity bit settings. Each serial port has synchronous RAM memories acting as transmit and receive FIFOs to buffer outgoing and incoming data. This block has registers for interrupts, line status, and line control features which can be accessed by software. The Serial Port Controllers can interface to external RS-232 / RS-422 / RS-485 transceivers.

### *Vendor Specific Command Processor*
The bridge logic interfaces to a vendor specific command processor block containing commands/register settings (BAUD settings etc.) which are specific to this device.

### *Interrupt-In Block*
The Interrupt-In controller block gives the status of the serial port interrupt registers to the USB-2.0 Device Controller. The USB host controller periodically polls the interrupt endpoint and reads the status of the interrupts.

### *Wakeup Block*
The Wakeup block is used for remote wakeup control. The USB host can suspend operation of the device. The remote wakeup block checks for activity on the serial port pins, and if information is available, it issues a remote wakeup request to the USB-2.0 Device Controller. The Device Controller in turn requests a remote wakeup by the external host. The host issues the “Resume Signaling” command to the device, which then resumes normal operation.

### *$I^2C$ EEPROM Controller*
The $I^2C$ EEPROM Controller interfaces to an external EEPROM and retrieves information necessary for serial port settings, Product-IDs, Vendor-IDs and other control information. The EEPROM controller logic communicates with the USB-2.0 Device Controller block which uses the information from the external EEPROM.

### *Clock Generation and Resets*
The Clock Generation logic is used to generate the clocks for the various BAUD rates supported by the device. The Resets block has logic for synchronous de-assertion and asynchronous assertion of Resets in the respective clock domains to various blocks.

### *BAUD Clock Generators*
The BAUD Clock Generator block generates clocks for each of the Serial Port Controllers depending on the BAUD settings from the host. A source clock is generated from the Clock Recovery block which is further divided or used as is by the BAUD Clock Generator logic depending on the BAUD settings.

### *PLL Clock Generator*
The PLL generates a master clock which the other blocks use to generate the various BAUD rates. The PLL supports a wide range of clock inputs to support industrial standard serial port bit rates, as well as custom BAUD rates.

**UART Functional Description**

***Overview***

The UARTs are high performance serial ports that comply with the 16c550 specification. All UARTs are similar in operation and function, and are described in this section. The function of a single UART is described below.

***Operation Modes***

The UARTs are backward compatible with 16c450 and 16c550 devices. The operation of the port depends upon the mode settings, which are described throughout the rest of this section. The modes, conditions and corresponding FIFO depth are tabulated below.

| UART Mode | FIFO Size | FCR[0] |
|---|---|---|
| 450 | 1 | 0 |
| 550 | 16 | 1 |

***450 Mode***

After the hardware reset, bit-0 of the FIFO Control Register (FCR) is cleared, and the UART is compatible with the 16c450 mode of operation.

The transmitter and receiver FIFOs (referred to as the “Transmitter Holding Register” and “Receiver Holding Register” respectively) have a depth of one.

This mode of operation is known as “Byte Mode”.

***550 Mode***

After the hardware reset, writing a 1 to FCR[0] will increase the FIFO size to 16, providing compatibility with 16c550 devices.

In 16c550 mode, the device has the following features:

- RTS/CTS hardware flow control or DSR/DTR hardware flow control
- Infrared IrDA format transmit & receive mode
- Deeper (16-Byte) FIFOs

**UART Register-Set and Register Descriptions**

The UART has 10 registers, but only three address lines to access those registers. The mapping of the registers is dependent upon the Line Control Register (LCR).

LCR[7] enables the Divider Latch Registers (DLL & DLM).

The following table gives the various UART registers and their offsets.

| Register Name | Offset | R/W | Bit-7 | Bit-6 | Bit-5 | Bit-4 | Bit-3 | Bit-2 | Bit-1 | Bit-0 |
|---|---|---|---|---|---|---|---|---|---|---|
| THR | 0 | W | Data to be transmitted ( Transmitter Holding Register) | | | | | | | |
| RHR | 0 | R | Data to be received (Receiver Holding Register) | | | | | | | |
| IER | 1 | R/W | Reserved | | | Sleep Mode | Modem Int Mask | Rx Stat Int Mask | Tx Rdy Int Mask | Rx Rdy Int Mask |
| FCR | 2 | W | RHR Trigger Level | | Reserved | | Reserved | Flush THR | Flush RHR | FIFO Enable |
| ISR | 2 | R | FIFOs Enabled | | Reserved | | Interrupt Priority | | | Interrupt Pending |
| LCR | 3 | R/W | DLE | Tx Break | Force Parity | Odd/Even Parity | Parity Enable | Stop Bits | Data Length | |
| MCR | 4 | R/W | DTR – DSR/ DCD Flow Control | | RTS/CTS Flow Control | Loop | Unused | | RTS | DTR |
| LSR | 5 | R | Data Error | Tx Empty | THR Empty | Rx Break | Framing Error | Parity Error | Overrun Error | Rx Rdy |
| MSR | 6 | R | DCD | RI | DSR | CTS | ΔDCD | Teri | ΔDSR | ΔCTS |
| SPR | 7 | R/W | Scratch Pad Register | | | | | | | |

***Additional standard registers - these are accessed when LCR[7] = 1***

| | | | |
|---|---|---|---|
| DLL | 0 | R/W | Divisor Latch bits[7:0] |
| DLM | 1 | R/W | Divisor Latch bits[15:8] |

**Transmitter Holding Register & Receiver Holding Register (THR & RHR):**

Data is written into the bottom of the THR queue & read from the top of the RHR queue completely asynchronously to the operation of the transmitter & receiver. The size of the FIFOs is dependent upon the setting of the FCR register.

Data written to the THR when it is full, is lost. Data read from the RHR when it is empty, is invalid. The empty and full status of the FIFOs is indicated in the Line Status Register.

**Register:** THR
**Description:** Data to be transmitted
**Offset:** 0
**Permissions:** Write Only
**Access Condition:** LCR[7] = 0
**Default Value:** (unknown) – based on memory

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Data to be transmitted | | | | | | | |

**Register:** RHR
**Description:** Data to be received
**Offset:** 0
**Permissions:** Read Only
**Access Condition:** LCR[7] = 0
**Default Value:** (unknown) – based on memory

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Data to be received | | | | | | | |

**Interrupt Enable Register (IER):**

Serial channel interrupts are enabled using the Interrupt Enable Register (IER).

**Register:** IER
**Description:** Interrupt Enable Register
**Offset:** 1
**Permissions:** Read/Write
**Access Condition:** LCR[7] = 0
**Default Value:** 0x0C

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | | Sleep Mode | Modem Int Mask | Rx Stat Int Mask | Tx Rdy Int Mask | Rx Rdy Int Mask |

| Bit | Description | Operation |
|---|---|---|
| 0 | Rx Rdy Interrupt Mask | Logic 0: Disable the Receiver Ready Interrupt<br>Logic 1: Enable the Receiver Ready Interrupt |
| 1 | Tx Rdy Interrupt Mask | Logic 0: Disable the Transmitter Ready Interrupt<br>Logic 1: Enable the Transmitter Ready Interrupt |
| 2 | Rx Stat Interrupt Mask | Logic 0: Disable the Receiver Status Interrupt (Normal Mode)<br>Logic 1: Enable the Receiver Status Interrupt (Normal Mode) |
| 3 | Modem Interrupt Mask | Logic 0: Disable the Modem Status Interrupt<br>Logic 1: Enable the Modem Status Interrupt |
| 4 | Sleep Mode | Logic 0: Disable Sleep Mode<br>Logic 1: Enable Sleep Mode where by the internal clock of the channel is switched OFF |
| [7:5] | Reserved | Reserved |

**FIFO Control Register (FCR):**

The FCR controls the UART behavior in various modes.

**Register:** FCR
**Description:** FIFO Control Register
**Offset:** 2
**Permissions:** Write
**Access Condition:**
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| RHR Trigger Level | | Reserved | | Reserved | Flush THR | Flush RHR | Enable FIFOs |

| Bit | Description | Operation |
|---|---|---|
| 0 | Enable FIFO Mode | Logic 0: Byte Mode<br>Logic 1: FIFO Mode |
| 1 | Flush RHR | Logic 0: No change<br>Logic 1: Flushes the contents of RHR, This is operative only in FIFO mode. The RHR is automatically flushed whenever changing between Byte Mode and FIFO Mode. The bit will return to zero after clearing the FIFO. |
| 2 | Flush THR | Logic 0: No change<br>Logic 1: Flushes the content of the THR, in the same manner as FCR[1] does the RHR |
| 3 | Reserved | Reserved |
| [5:4] | Reserved | Reserved |
| [7:6] | RHR Trigger Level | See Table Below |

In 550 Mode, the receiver FIFO trigger levels are defined by FCR[7:6].

The interrupt trigger level & flow control trigger level where appropriate are defined by L2 in the table.

L1 defines a lower flow control trigger level. The two trigger levels used together introduce a hysteresis element into the hardware RTS/CTS flow control.

In Byte Mode (450 Mode) trigger levels are all set to 1.

| FCR[7:6] | 550 Mode (FIFO = 16) | |
|---|---|---|
| | ***L1*** | ***L2*** |
| 2'b00 | 1 | 1 |
| 2'b01 | 1 | 4 |
| 2'b10 | 1 | 8 |
| 2'b11 | 1 | 14 |

**Interrupt Status Register (ISR):**

The source of the highest priority pending interrupt is indicated by the contents of the Interrupt Status Register. There are five sources of interrupts and four levels of priority (1 is the highest) as tabulated below:

**Register:** ISR
**Description:** Interrupt Status Register
**Offset:** 2
**Permissions:** Read
**Access Condition:**
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| FIFOs Enabled | | Interrupt Priority (Enhanced Mode) | | Interrupt Priority (All Modes) | | | Interrupt Pending |

***Interrupt Source and Priority Table***

| Priority Level | Interrupt Source | ISR[5:0] |
|---|---|---|
| - | No interrupt pending | 6'b000001 |
| 1 | Receiver Status Error or address bit detected in 9-bit mode | 6'b000110 |
| 2a | Receiver Data Available | 6'b000100 |
| 2b | Receiver Time-Out | 6'b001100 |
| 3 | Transmitter THR Empty | 6'b000010 |
| 4 | Modem Status Change | 6'b000000 |

**Note:** ISR[0] indicates whether any interrupt is pending

**Line Control Register (LCR):**

The LCR specifies the data format that is common to both transmitter and receiver.

| | |
|---|---|
| **Register:** | LCR |
| **Description:** | Line Control Register |
| **Offset:** | 3 |
| **Permissions:** | Read/Write |
| **Access Condition:** | |
| **Default Value:** | 0x00 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| DLE | TX Break | Force Parity | Odd/Even Parity | Parity Enable | Number of Stop-Bits | | Data Length |

LCR[1:0] Data Length of serial characters.

LCR[2] Number of Stop-Bits per serial character.

LCR[5:3] Parity Type
The selected parity type will be generated during transmission and checked by the receiver, which may produce a parity error as a result. In 9-bit mode parity is disabled and LCR[5:3] are ignored.

LCR[6] Transmission Break
Logic 0: Transmission Break Disabled.
Logic 1: Forces the transmitter data output SOUT low to alert the communications channel, or sends zeroes in IrDA mode.

LCR[7] Divisor Latch Enable
Logic 0: Accesses to DLL and DLM registers disabled.
Logic 1: Accesses to DLL and DLM registers enabled.

| LCR[1:0] | Data Length |
|---|---|
| 2'b00 | 5 bits |
| 2'b01 | 6 bits |
| 2'b10 | 7 bits |
| 2'b11 | 8 bits |

| LCR[2] | Data Length | Number of Stop-Bits |
|---|---|---|
| 0 | 5, 6, 7, 8 | 1 |
| 1 | 5 | 1.5 |
| 1 | 6, 7, 8 | 2 |

| LCR[5:3] | Parity Type |
|---|---|
| 3'bxx0 | No Parity |
| 3'b001 | Odd Parity |
| 3'b011 | Even Parity |
| 3'b101 | Parity bit forced to 1 |
| 3'b111 | Parity bit forced to 0 |

**Line Status Register (LSR):**

This register provides the status of the data transfer to CPU.

| | |
|---|---|
| **Register:** | LSR |
| **Description:** | Line Status Register |
| **Offset:** | 5 |
| **Permissions:** | Read |
| **Access Condition:** | |
| **Default Value:** | 0x00 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Data Error | Tx Empty | THR Empty | Rx Break | Framing Error | Parity Error | Overrun Error | Rx Rdy |

| Bit | Description | Operation |
|---|---|---|
| 0 | RHR Data Available | Logic 0: RHR is empty<br>Logic 1: RHR is not empty. Data is available to be read |
| 1 | RHR Overrun | Logic 0: No overrun error<br>Logic 1: Data was received when the RHR was full, An overrun has occurred. The error is flagged when the data would normally have been transferred to the RHR. |
| 2 | Received Data Parity Error | Logic 0: No parity error in normal mode or 9th bit received data is “0” in 9-bit mode.<br>Logic 1: Data has been received that did not have correct parity |
| 3 | Received Data Framing Error | Logic 0: No framing error<br>Logic 1: Data has been received with an invalid stop-bit. |
| 4 | Receiver Break Error | Logic 0: No receiver break error<br>Logic 1: The receiver received a break error |
| 5 | THR Empty | Logic 0: Transmitter FIFO is not empty<br>Logic 1: Transmitter FIFO is empty |
| 6 | Transmitter & THR Empty | Logic 0: The transmitter is not idle<br>Logic 1: THR is empty & the transmitter has completed the character in the shift register and is in the idle mode |
| 7 | Receiver Data Error | Logic 0: Either there is no receiver data error in the FIFO or it was cleared by an earlier read of LSR<br>Logic 1: At least one parity error, framing error or break indication is present in the FIFO. |

**Modem Control Register (MCR):**

This register controls the UART's flow control and self diagnostic features.

| | |
|---|---|
| **Register:** | MCR |
| **Description:** | Modem Control Register |
| **Offset:** | 4 |
| **Permissions:** | Read/Write |
| **Access Condition:** | |
| **Default Value:** | 0x00 |

| 550 Mode | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit[7]** | **Bit[6]** | **Bit[5]** | **Bit[4]** | **Bit[3]** | **Bit[2]** | **Bit[1]** | **Bit[0]** |
| DTR-DSR/DCD Flow Control | | CTS/RTS Flow Control | Internal Loop Back Enable | Reserved | Reserved | RTS | DTR |

| Bit | Description | Operation |
|---|---|---|
| 0 | DTR | Logic 0: Forces DTR# output to inactive (high)<br>Logic 1: Forces DTR# output to active (low) |
| 1 | RTS | Logic 0: Forces RTS# output to inactive (high)<br>Logic 1: Forces RTS# output to active (low) |
| 2 | Reserved | Reserved |
| 3 | Reserved | Reserved |
| 4 | Loop-Back Mode | Logic 0: Normal operating mode<br>Logic 1: Enable local Loop-Back Mode |
| 5 | CTS/RTS Flow Control | Logic 0: CTS/RTS flow control disabled in 550 mode<br>Logic 1: CTS/RTS flow control enabled in 550 mode |
| 6 | DTR/DSR Flow Control | Logic 0: DTR/DSR flow control disabled in 550 mode<br>Logic 1: DTR/DSR flow control enabled in 550 mode |
| 7 | DCD Flow Control | Logic 0: DCD flow control disabled in 550 mode<br>Logic 1: DCD flow control enabled in 550 mode |

**Modem Status Register (MSR):**

This register provides the status of the modem control lines to CPU.

| | |
|---|---|
| **Register:** | MSR |
| **Description:** | Modem Status Register |
| **Offset:** | 6 |
| **Permissions:** | Read |
| **Access Condition:** | |
| **Default Value:** | 0x00 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| DCD | RI | DSR | CTS | ΔDCD | Teri | ΔDSR | ΔCTS |

| Bit | Description | Operation |
|---|---|---|
| 0 | Delta CTS | Logic 0: No change in the CTS signal<br>Logic 1: Indicates that the CTS input has changed since the last time the MSR was read |
| 1 | Delta DSR | Logic 0: No change in the DSR signal<br>Logic 1: Indicates that the DSR input has changed since the last time the MSR was read |
| 2 | Trailing Edge of RI | Logic 0: No change in the RI signal<br>Logic 1: Indicates that the RI input has changed from low to high since the last time the MSR was read |
| 3 | Delta DCD | Logic 0: No change in the DCD signal<br>Logic 1: Indicates that the DCD input has changed since the last time the MSR was read |
| 4 | CTS | Logic 0: CTS# line is 1<br>Logic 1: CTS# line is 0 |
| 5 | DSR | Logic 0: DSR# line is 1<br>Logic 1: DSR# line is 0 |
| 6 | RI | Logic 0: RI# line is 1<br>Logic 1: RI# line is 0 |
| 7 | DCD | Logic 0: DCD# line is 1<br>Logic 1: DCD# line is 0 |

**Scratch Pad Register (SPR):**

The scratch pad register does not influence operation of the UART in RS-232 mode in any way, and is used for temporary data storage. When using RS-422/485 Mode, bit[6] and bit[7] of the Scratch Pad Register are used for mode setting and DTR active level settings.

| | |
|---|---|
| **Register:** | SPR |
| **Description:** | Scratch Pad Register |
| **Offset:** | 7 |
| **Permissions:** | Read/Write |
| **Access Condition:** | |
| **Default Value:** | 0x00 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Scratch Pad Register Data | | | | | | | |

**Divisor Latch Registers (DLL & DLM):**

The Divisor Latch Registers are used to program the BAUD Rate divisor.
This is a value between 1 and 65535 by which the input clock is divided in order to generate serial BAUD rates.

After the hardware reset, the BAUD Rate used by the transmitter & receiver is given by:

BAUD Rate = Input Clock / (16 * Divisor)

where divisor is given by (256 * DLM) + DLL.

More flexible BAUD rate generation options are also available.

| | |
|---|---|
| **Register:** | DLL |
| **Description:** | Divisor Latch (Least Significant Byte) |
| **Offset:** | 0 |
| **Permissions:** | Read/Write |
| **Access Condition:** | LCR[7] = 1 |
| **Default Value:** | 0x01 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Least Significant Byte of divisor latch | | | | | | | |

| | |
|---|---|
| **Register:** | DLM |
| **Description:** | Divisor Latch (Most Significant Byte) |
| **Offset:** | 1 |
| **Permissions:** | Read/Write |
| **Access Condition:** | LCR[7] = 1 |
| **Default Value:** | 0x00 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Most Significant Byte of divisor latch | | | | | | | |

**RS-422 / RS-485 Mode Support**

Two additional modes of serial port operation are supported, these are:

- RS-422 Mode – Full Duplex Serial Port for industrial applications
- RS-485 Mode – Half Duplex Serial Port for industrial applications

***RS-485***

The RS-485 mode can be set using the Scratch Pad Register bit[6] and bit[7] for each serial port.

This mode is a half duplex mode and the external transceiver is controlled for transmission or reception using the enable signal.

| Scratch Pad Bit[7] | Scratch Pad Bit[6] | Operation Summary |
|---|---|---|
| 0 | X | RS-485 Mode Disabled |
| 1 | 0 | RS-485 Mode Enabled, DTR High = Rx DTR Low = Tx |
| 1 | 1 | RS-485 Mode Enabled DTR Low = Rx DTR High = Tx<br><br>This is the default selection when RS485 mode is selected through driver property sheets. |

***RS-422***

This is the full duplex mode.

This mode will work without the use of the DTR signal for external transceiver control.

**Configuration Options**
Four serial ports can be configured for operation.
To program and access the serial ports via software, endpoint numbers have been assigned so that serial ports can be configured from the USB side.

| Endpoint | Type | Function | Size (Bytes) (USB-1.1 / USB-2.0) |
|---|---|---|---|
| 0 | Control Endpoint | Default Functionality | 8 / 64 |
| 1 | Bulk-In | Serial Port – 1 | 64 / 512 |
| 2 | Bulk-Out | Serial Port – 1 | 64 / 512 |
| 3 | Bulk-In | Serial Port – 2 | 64 / 512 |
| 4 | Bulk-Out | Serial Port – 2 | 64 / 512 |
| 5 | Bulk-In | Serial Port – 3 | 64 / 512 |
| 6 | Bulk-Out | Serial Port – 3 | 64 / 512 |
| 7 | Bulk-In | Serial Port – 4 | 64 / 512 |
| 8 | Bulk-Out | Serial Port – 4 | 64 / 512 |
| 9 | Interrupt | Status Endpoint | 5 or 13 * |

* Controlled by DCR1 bit-6

**Serial Port Set/Get Commands**
Vendor commands are the vendor specific USB setup commands. The purpose of the vendor commands is to set/get the contents of the application registers. The following table provides information on the various vendor specific commands.

Windex [7:0] is the register index from where data is to be read.

Brequest specifies whether to read or write.
- 0x0E = write to the application register
- 0x0D = read from the application register

Wvalue specifies the application number and data to be written (ww = data).
- 0x01ww is the application number for Serial Port-1
- 0x02ww is the application number for Serial Port-2
- 0x03ww is the application number for Serial Port-3
- 0x04ww is the application number for Serial Port-4
- 0x09ww is the application number for EEPROM Write/Read
- 0x00ww is the application number provided for accessing the Control Registers which control the UARTs. It is possible to enable higher BAUD rates, and features like auto hardware flow control using the Control Registers

**Note:** "N" in Wvalue and Register Name columns indicate the corresponding serial port number.

Windex is the offset of the register to read/write.

Wlength is the length of the data to read/write.

**Get Application Vendor Specific Command (Serial Port -N)**

| bmrequestType | Brequest | Wvalue | Windex | Wlength | Register Name |
|---|---|---|---|---|---|
| 0xC0 | 0x0D | 0x0N00 | 0x0000 | 0x0001 | SPN_RHR |
| 0xC0 | 0x0D | 0x0N00 | 0x0001 | 0x0001 | SPN_IER |
| 0xC0 | 0x0D | 0x0N00 | 0x0002 | 0x0001 | SPN_IIR |
| 0xC0 | 0x0D | 0x0N00 | 0x0003 | 0x0001 | SPN_LCR |
| 0xC0 | 0x0D | 0x0N00 | 0x0004 | 0x0001 | SPN_MCR |
| 0xC0 | 0x0D | 0x0N00 | 0x0005 | 0x0001 | SPN_LSR |
| 0xC0 | 0x0D | 0x0N00 | 0x0006 | 0x0001 | SPN_MSR |
| 0xC0 | 0x0D | 0x0N00 | 0x0007 | 0x0001 | SPN_SPR |
| 0xC0 | 0x0D | 0x0N00 | 0x0000 | 0x0001 | SPN_DLL |
| 0xC0 | 0x0D | 0x0N00 | 0x0001 | 0x0001 | SPN_DLM |

**Set Application Vendor Specific Command (Serial Port -N)**

| bmrequestType | Brequest | Wvalue | Windex | Wlength | Register Name |
|---|---|---|---|---|---|
| 0x40 | 0x0E | 0x0Nww | 0x0000 | 0x0001 | SPN_THR |
| 0x40 | 0x0E | 0x0Nww | 0x0001 | 0x0001 | SPN_IER |
| 0x40 | 0x0E | 0x0Nww | 0x0002 | 0x0001 | SPN_FCR |
| 0x40 | 0x0E | 0x0Nww | 0x0003 | 0x0001 | SPN_LCR |
| 0x40 | 0x0E | 0x0Nww | 0x0004 | 0x0001 | SPN_MCR |
| 0x40 | 0x0E | 0x0Nww | 0x0005 | 0x0001 | SPN_LSR |
| 0x40 | 0x0E | 0x0Nww | 0x0006 | 0x0001 | SPN_MSR |
| 0x40 | 0x0E | 0x0Nww | 0x0007 | 0x0001 | SPN_SPR |
| 0x40 | 0x0E | 0x0Nww | 0x0000 | 0x0001 | SPN_DLL |
| 0x40 | 0x0E | 0x0Nww | 0x0001 | 0x0001 | SPN_DLM |

**USB Device Descriptors**

| Device Descriptor | Location | Data |
|---|---|---|
| BLength | 0 | 8'h12 |
| BDescriptorType | 1 | 8'h01 |
| BcdUSB | 2 | 8'h00 |
| BcdUSB | 3 | 8'h02 |
| BDeviceClass | 4 | 8'hFF |
| BDeviceSubClass | 5 | 8'h00 |
| BDeviceProtocol | 6 | 8'hFF |
| bMaxPacketSize0 | 7 | 8'h40 |
| IdVendor | 8 | 8'h10 |
| IdVendor | 9 | 8'h97 |
| IdProduct | 10 | 8'h40 |
| IdProduct | 11 | 8'h78 |
| BcdDevice | 12 | 8'h01 |
| BcdDevice | 13 | 8'h00 |
| iManufacturer | 14 | 8'h00 / 02 * |
| iProduct | 15 | 8'h00 / 03 * |
| iSerialNumber | 16 | 8'h00 / 01 * |
| BNumConfigurations | 17 | 8'h01 |

* Values returned Without / With the Serial EEPROM present.

**USB Configuration Descriptors**

| Configuration Descriptor | Index | Data |
|---|---|---|
| BLength | 0 | 8'h09 |
| BDescriptorType | 1 | 8'h02 |
| WtotalLength(L) | 2 | 8'h51 |
| WtotalLength(M) | 3 | 8'h00 |
| BNumInterfaces | 4 | 8'h01 |
| BConfigurationValue | 5 | 8'h01 |
| IConfiguration | 6 | 8'h00 |
| BmAttributes | 7 | 8'hA0 |
| BMaxPower | 8 | 8'h32 (100 mA) |

**USB Interface Descriptors**

| Configuration Descriptor | Index | Data |
|---|---|---|
| BLength | 0 | 8'h09 |
| BDescriptorType | 1 | 8'h04 |
| BInterfaceNumber | 2 | 8'h00 |
| BAlternateSetting | 3 | 8'h00 |
| BNumEndpoints | 4 | 8'h09 |
| BInterfaceClass | 5 | 8'hFF |
| BInterfaceSubClass | 6 | 8'h00 |
| BInterfaceProtocol | 7 | 8'hFF |
| IInterface | 8 | 8'h00 |

**Endpoint-1 Serial Port 1 Bulk-In**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h81 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-2 Serial Port 1 Bulk-Out**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h02 |
| bmAttributes | 3 | 8'h02 |
| WmaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| WmaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-3 Serial Port 2 Bulk-In**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h83 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-4 Serial Port 2 Bulk-Out**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h04 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

* Values for Full Speed & High Speed USB

**Endpoint-5 Serial Port 3 Bulk-In**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h85 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-6 Serial Port 3 Bulk-Out**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h06 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-7 Serial Port 4 Bulk-In**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h87 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

**Endpoint-8 Serial Port 4 Bulk-Out**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h08 |
| bmAttributes | 3 | 8'h02 |
| wMaxPacketSize(L) | 4 | 8'h40/8'h00 * |
| wMaxPacketSize(M) | 5 | 8'h00/8'h02 * |
| bInterval | 6 | 8'hFF |

* Values for Full Speed & High Speed USB

**Endpoint-9**
**Interrupt Endpoint**

| Configuration Descriptor | Index | Data |
|---|---|---|
| bLength | 0 | 8'h07 |
| bDescriptorType | 1 | 8'h05 |
| bEndpointAddress | 2 | 8'h89 |
| bmAttributes | 3 | 8'h03 |
| wMaxPacketSize(L) | 4 | 8'h0A |
| wMaxPacketSize(M) | 5 | 8'h00 |
| bInterval | 6 | * 8'h01 / 8'h05 (default FS/HS) |

* programmable using intr_pg_fs , intr_pg_hs

**Vendor Specific Command Registers**

There are a total of 35 registers, out of which 10 Vendor Specific Registers can be used to tune the behavior and performance of each serial port. The remaining registers are EEPROM registers. These registers are listed below.

***VSPEC Control Registers Description Table***

The vspec control registers are accessed through Vendor Specific Commands, with Application_ Number = 0.

| Register | Offset | App # |
|---|---|---|
| sp1_reg | 0 | 0 |
| control_reg1 | 1 | 0 |
| ping_pong_high | 2 | 0 |
| ping_pong-low | 3 | 0 |
| eeprom_reg | 4, 5, 6 | 0 |
| eeprom_reg | 22-30 | 0 |
| sp2_reg | 8 | 0 |
| control_reg2 | 9 | 0 |
| sp3_reg | 10 | 0 |
| control_reg3 | 11 | 0 |
| sp4_reg | 12 | 0 |
| control_reg4 | 13 | 0 |
| pll_m_reg | 14 | 0 |
| pll_n_reg | 16 | 0 |
| clk_mux_reg | 18 | 0 |
| clk_select_reg1 | 19 | 0 |
| clk_select_reg2 | 20 | 0 |
| mode_reg* | 43 | 0 |
| sp1_ICG_reg | 44 | 0 |
| sp2_ICG_reg | 45 | 0 |
| sp3_ICG_reg | 46 | 0 |
| sp4_ICG_reg | 47 | 0 |
| RX_sampling_reg1 | 48 | 0 |
| RX_sampling_reg2 | 49 | 0 |
| bi_fifo_stat1* | 50 | 0 |
| bo_fifo_stat1* | 51 | 0 |
| bi_fifo_stat2* | 52 | 0 |
| bo_fifo_stat2* | 53 | 0 |
| bi_fifo_stat3* | 54 | 0 |
| bo_fifo_stat3* | 55 | 0 |
| bi_fifo_stat4* | 56 | 0 |
| bo_fifo_stat4* | 57 | 0 |
| zero_len_reg1 | 58 | 0 |
| zero_len_reg2 | 59 | 0 |
| zero_len_reg3 | 60 | 0 |
| zero_len_reg4 | 61 | 0 |
| zero_len_flag_en | 62 | 0 |
| thr_value1_sp1 | 63 | 0 |
| thr_value2_sp1 | 64 | 0 |
| thr_value1_sp2 | 65 | 0 |
| thr_value2_sp2 | 66 | 0 |
| thr_value1_sp3 | 67 | 0 |
| thr_value2_sp3 | 68 | 0 |
| thr_value1_sp4 | 69 | 0 |
| thr_value2_sp4 | 70 | 0 |

* indicates a read-only register

**sp1 (2, 3, 4)_reg**

This register is used to configure Endpoint-1 (3, 5, 7). This enables the designer to pin point the problem in the design. There is a register bit which resets the UART. There are register bits which control the input clock fed to the UART, thereby providing options for higher BAUD rates.

**Register:** SP1 (2, 3, 4)_REG
**Description:** Serial Port 1 (2, 3, 4) Register
**Register Index:** 0 (8, 10, 12)
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| UART_ Reset | clk_UART_select | | | sp_bi_clear | sp_bo_clear | ser_line_err_ctl_en | udc_loop |

| Bit | Description |
|---|---|
| udc_loop | When enabled, loops the data from Bulk-Out FIFO to the Bulk-In FIFO. |
| ser_line_err_ctl_en | When enabled, will not allow data from the UART to be written into the Bulk-In FIFO if there are any errors in the received data. |
| sp_bo_clear | Reset the Bulk-Out FIFO |
| sp_bi_clear | Reset the Bulk-In FIFO |
| clk_UART_sel | Changes the clock fed to the UART as shown in table below |
| UART1_reset | Resets the UART |

**clk_UART_sel**

These bits are used by the BAUD clock generators when generating clocks for higher BAUD rates. The clock frequency and maximum BAUD rates achieved are shown in the table.

| Option | Input Clock Frequency | Max BAUD Rate |
|---|---|---|
| 3'b000 | 1.8432 MHz | 115200 bps |
| 3'b001 | 1.8432 x 2 = 3.6864 MHz | 230400 bps |
| 3'b010 | 1.8432 x 3.5 = 6.4512 MHz | 403200 bps |
| 3'b011 | 1.8432 x 4 = 7.3728 MHz | 460800 bps |
| 3'b100 | 1.8432 x 7 = 12.9024 MHz | 806400 bps |
| 3'b101 | 1.8432 x 8 = 14.7456 MHz | 921600 bps |
| 3'b110 | 24 MHz | 1.5 Mbps |
| 3'b111 | 48 MHz | 3 Mbps |

**control_reg1 (2, 3, 4)**

The Control register is used for controlling the flow control, driver done bit setting after setting all serial port controls and IrDA related register bits of Serial Port 1.

**Register:** control_reg1 (2, 3, 4)
**Description:** Control Register1 (2, 3, 4)
**Register Index:** 1 (9, 11, 13)
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | fsm_ control | rx_ disable | rx_ negate | drv_ done | sp_ bit | Reserved | sp_ autoflow_ pwrdwn_en |

| Bit | Description |
|---|---|
| sp_ autoflow_ pwrdwn_en | This bit is used for enabling the hardware flow control. |
| Reserved | Reserved (Unused) |
| sp_bit | when set, CTS change is reflected in Modem Status Register, else the delta CTS is not set. |
| drv_done (used only in control_ reg1) | 0: UART RAMS are used for string descriptor.<br>1: UART RAMS are used for sending/receiving the data. This bit is to be set after USB enumeration, and before setting all serial port controls for transmission/reception. |
| rx_negate | The input from the transmitter is inverted when this bit is enabled. This bit is valid only in the IrDA mode. |
| rx_disable | When this bit is enabled, and IrDA bit is not set, the Serial Port will not receive any data. |
| fsm_control | fsm_control doesn't receive data while the UART is transmitting, this bit is valid only in IrDA mode |
| Reserved | Reserved (Unused) |

Bit 3 is used only in control_reg1 as drv_done. For the rest of the control_regs (2-4) bit-3 is reserved.

**ping_pong_high & ping_pong_low**

Each serial port has a 512-Byte Bulk-In FIFO. The FIFO actually has two 512-Byte banks. The two banks use a ping-pong mechanism, so that only one is active at a time. In some situations, these registers can increase receiver performance by enabling a timeout mechanism in the serial ports to control when the FIFO switches banks. The current software Driver does not enable this feature.

Each UART transfers the data it receives to its Bulk-In FIFO. When the host controller initiates a Bulk-In transfer to that endpoint and reads the data, the other bank of the FIFO is activated. This allows the UART to continue receiving data asynchronously, while the host reads the data received prior to the Bulk-In request. If the host controller tries to read the FIFO when it is empty, it gets a NAK response.

These registers can reduce the number of NAKs by controlling the time at which the FIFO switches banks. They are combined into a 15-bit value using this formula:

(128 * ping_pong_high) + ping_pong_low

The "correct" settings for these registers are very application and data rate specific. Improper settings can actually reduce performance. In most cases, it will not be necessary to change these values. This feature affects all serial ports equally; it is not possible to set different values for each port.

**Register:** ping_pong_high
**Description:** MSB of 15-bit Memory Switching Threshold Value
**Register Index:** 2
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| ping_pong_high | | | | | | | |

**Register:** ping_pong_low
**Description:** LSB of 15-bit Memory Switching Threshold Value
**Register Index:** 3
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| 0 | ping_pong_low | | | | | | |

**PLL Dividers**
The PLL has two programmable dividers. The "Pre-Divider" (M) divides the Input clock before it goes to the phase comparator. The "Loop-Divider" (N) divides the Output clock signal before it goes to the phase comparator. Together, these two registers are used to set the ratio of the Output clock to Input clock frequencies.

Because they are used as divisors, neither M nor N should be set to zero.

The formula used to set the Output frequency is:

$F_{Out} = (N/M) * F_{In}$

The default Input clock is 12 MHz.
To obtain a 30 MHz Output from the PLL, the values M=2 and N=5 could be used ( (5/2) * 12 = 30 ).

In order to maintain a stable Output clock frequency, this important relationship must be maintained:

$5\ MHz \leq (F_{In} / M) \leq 100\ MHz$

**pll_m_reg**
Register writes/reads are possible for lower 6 bits of the register.

| | |
|---|---|
| **Register:** | pll_m_reg |
| **Description:** | PLL Divider Register – "Pre" Divisor |
| **Register Index:** | 14 |
| **Permissions:** | Read/Write |
| **Application Number:** | 0 |
| **Default Value:** | 0x01; (Reset value = 0x03) |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | M | | | | | |

**pll_n_reg**
Register writes/reads are possible for lower 6 bits of the register.

| | |
|---|---|
| **Register:** | pll_n_reg |
| **Description:** | PLL Divider Register – "Loop" Divisor |
| **Register Index:** | 16 |
| **Permissions:** | Read/Write |
| **Application Number:** | 0 |
| **Default Value:** | 0x08 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | N | | | | | |

**clk_mux_reg**

Register writes are possible only for bit[3:0] of the register. Internally the register is configured as an 8-bit register and clk_mux_reg[7:4] are reserved for future use.

**Register:** clk_mux_reg
**Description:** clk_mux_reg
**Register Index:** 18
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | | | FIFO Status Enable | FRANGE Setting | Clock Input to PLL | |

| Bit | Description |
|---|---|
| [1:0] | ***Selects the Input Clock for the PLL***<br><br>0: 12 MHz<br>1: External Clock Input (EXT_CLK)<br>2: Reserved<br>3: Reserved |
| 2 | ***PLL Output Frequency Range***<br><br>0: 20 MHz – 100 MHz<br>1: 100 MHz – 300 MHz |
| 3 | ***Enable additional Status Information***<br><br>0: Endpoint-9 returns 5 Bytes of data.<br>1: Endpoint-9 returns 5 Bytes of data, plus 8 Bytes of FIFO Status. |
| [7:4] | Reserved |

**clk_select_reg1**

Input Clock Selector for Serial Ports 1 & 2.
Register writes are possible for bit[5:0] of the register.

**Register:** clk_select_reg1
**Description:** clk_select_reg1
**Register Index:** 19
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | Serial Port 2 Clock Input | | | Serial Port 1 Clock Input | | |

**clk_select_reg2**

Input Clock Selector for Serial Ports 3 & 4
Register writes are possible for bit[5:0] of the register.

**Register:** clk_select_reg2
**Description:** clk_select_reg2
**Register Index:** 20
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | Serial Port 4 Clock Input | | | Serial Port 3 Clock Input | | |

Each Serial Port can be configured independently.

The input clock for each Port can be selected from one of the following:

| Selector Value | Description |
|---|---|
| 0 | Standard BAUD Rates (Derived from 96 MHz) |
| 1 | 30 MHz |
| 2 | 96 MHz |
| 3 | 120 MHz |
| 4 | PLL Output |
| 5 | External Clock Input |
| 6 | Reserved |
| 7 | Reserved |

**Mode Register**

A separate 8-bit Mode Register is defined to indicate the mode of operation. The contents of this Mode Register are tabulated below. Bits in the Mode Register are set by bonding options available on the die and can only be read by software during normal operation.

**Register:** Mode Register
**Description:** Mode Register
**Register Index:** 43
**Permissions:** Read Only
**Application Number:** 0
**Default Value:** 0xC4

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| irda_en_i | ee_wr_en_i | Reserved | bypass_ por | Pll_ bypass | ser_ prsnt | reset_ sel | Reserved |

| Bit | Name | Definition | Default Value |
|---|---|---|---|
| 0 | Reserved | Reserved | 0 |
| 1 | reset_sel | 0: RESET = Active High<br>1: Reserved | 0 |
| 2 | ser_prsnt | 0: Reserved<br>1: Do not use hard coded values | 1 |
| 3 | pll_bypass | 1: PLL clock output is bypassed | 0 |
| 4 | bypass_por | 1: Internal Power-On Reset is bypassed | 0 |
| 5 | Reserved | Reserved | 0 |
| 6 | ee_wr_en_i | 1: EEPROM write access is enabled | 1 |
| 7 | irda_en_i | 1: IrDA mode is activated | 1 |

**SP1 (2, 3, 4)_ICG_Register (Inter Character Gap Register)**

The Inter Character Gap Register controls the amount of time the serial port transmitter will wait before transmitting the next character.

Each serial port has an 8-bit ICG_reg which can be programmed by software. The decimal value of the register times the BAUD clock period for that serial port gives the amount of time that the transmitter will wait between successive character transmissions.

| | |
|---|---|
| **Register:** | Sp1 (2, 3, 4)_ICG_reg |
| **Description:** | Inter Character Gap Register |
| **Register Index:** | 44 (45, 46, 47) |
| **Permissions:** | Read/Write |
| **Application Number:** | 0 |
| **Default Value:** | 0x24 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Inter Character Gap value (to be multiplied by BAUD clock period) | | | | | | | |

**RX Sampling Register**

All UARTs have an internal clock signal that runs sixteen times as fast as the currently selected BAUD Clock.

The RX Sampling Controller register is used to select the sampling time for the UART receiver logic. The default value (7) samples the data at the middle of the bit time. This register allows sampling the data more towards the beginning or end of the bit time if desired.

There is a 4-bit register value for each serial port.
Two 8-bit registers (Rx_sampling_reg1 and Rx_sampling_reg2) are used to control the sampling duration for the UART receivers.

| | |
|---|---|
| **Register:** | Rx_sampling_reg (1, 2) |
| **Description:** | RX Sampling Register |
| **Register Index:** | 48, 49 |
| **Permissions:** | Read/Write |
| **Application Number:** | 0 |
| **Default Value:** | 0x77 |

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| See Table | | | | See Table | | | |

The table shows the configuration for the Rx_sampling_reg1 and Rx_sampling_reg2 registers.

| Register | Bits | Serial Port |
|---|---|---|
| Rx_sampling_reg1 | [3:0] | 1 |
| Rx_sampling_reg1 | [7:4] | 3 |
| Rx_sampling_reg2 | [3:0] | 2 |
| Rx_sampling_reg2 | [7:4] | 4 |

**zero_len_reg1 (2, 3, 4)**

Zero-Length packets are generated if there is no data to send. Zero-Length packet generation only occurs when the Bulk-In FIFO is empty after completion of all the requests coming from the host. This value indicates the number of Bulk-In requests coming from the host, before a Zero-Length packet is generated.

**Register:** zero_len_reg1 (2, 3, 4)
**Description:** This value indicates the number of Bulk-In requests from the host.
**Register Index:** 58 (59, 60, 61)
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x14

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Number of Bulk-Ins sent from the host controller before Zero-Length Packet is Generated | | | | | | | |

**zero_len_flag_en**

Enable/Disable the generation of Zero-Length packets by programming this register for all serial ports.

**Register:** zero_len_flag_en
**Description:** Enable/Disable the generation of Zero-Length packets.
**Register Index:** 62
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x0F

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | | | | Enable Zero Length Packet for SP4 | Enable Zero Length Packet for SP3 | Enable Zero Length Packet for SP2 | Enable Zero Length Packet for SP1 |

**thr_value1_sp1 (2, 3, 4) Register**

**Register:** thr_value 1_sp1 (2, 3, 4)
**Description:** Host can program the size of the Bulk-In packet for the Serial Port.
**Register Index:** 63 (65, 67, 69)
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x0F

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Least Significant bits of Threshold Value | | | | | | | |

**thr_value2_sp1 (2, 3, 4) Register**

**Register:** thr_value 2_sp1 (2, 3, 4)
**Description:** Host can program the size of the Bulk-In packet . The value can be from 1 Byte to 512 Bytes.
**Register Index:** 64 (66, 68, 70)
**Permissions:** Read/Write
**Application Number:** 0
**Default Value:** 0x80

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Threshold Value Enable Bit | Reserved | | | | | | Most Significant bit of Threshold Value |

| Bits | Description |
|---|---|
| 0 | Most Significant bit of Threshold Value |
| [6:1] | Reserved |
| 7 | Threshold Value Enable Bit<br>Programmed values of thr_value1_sp1, thr_value2_sp1, thr_value1_sp2, thr_value2_sp2, thr_value1_sp3, thr_value2_sp3, thr_value1_sp4, thr_value2_sp4 registers are valid only when this bit is enabled. |

**EEPROM Write Commands**

| bmrequestType | Brequest | Wvalue | Windex | Wlength | Register Name |
|---|---|---|---|---|---|
| 0x40 | 0x0E | 0x09ww | 0x00rr | 0x0000 | EEPROM Register |

Wvalue [15:8] is the Application Number. (0x09 is for EEPROM access)
Wvalue [7:0] is the data to be written.
Windex [7:0] is the Register Index where the data is to be written.

**EEPROM Read Commands**

| bmrequestType | Brequest | Wvalue | Windex | Wlength | Register name |
|---|---|---|---|---|---|
| 0xC0 | 0x0D | 0x0900 | 0x00rr | 0x0001 | EEPROM Register |

**Status Endpoint**

The Status Endpoint returns the interrupt status each time it is polled by the host.. These Bytes are the status information of all the Serial Ports.

| Byte | Function | Port |
|---|---|---|
| 1 | Interrupt Identification Register | 1 |
| 2 | Interrupt Identification Register | 2 |
| 3 | Interrupt Identification Register | 3 |
| 4 | Interrupt Identification Register | 4 |
| 5 | FIFO Status | All |

***FIFO Status – Additional Bytes***

If the DCR1[6] bit is set, the Interrupt Endpoint will return eight additional Bytes after the five Bytes described earlier. The additional eight Bytes provide information about the number of Bytes currently present in the Bulk-In and Bulk-Out FIFOs for each endpoint.

These Bytes are:

| Byte | Bytes in FIFO | Endpoint |
|---|---|---|
| 6 | Bulk-In for SP1 | 1 |
| 7 | Bulk-Out for SP1 | 2 |
| 8 | Bulk-In for SP2 | 3 |
| 9 | Bulk-Out for SP2 | 4 |
| 10 | Bulk-In for SP3 | 5 |
| 11 | Bulk-Out for SP3 | 6 |
| 12 | Bulk-In for SP4 | 7 |
| 13 | Bulk-Out for SP4 | 8 |

**Status Endpoint Byte-5**

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| InFIFO Status_ Sp4 | OutFIFO Status_ Sp4 | InFIFO Status_ Sp3 | OutFIFO Status_ Sp3 | InFIFO Status_ Sp2 | OutFIFO Status_ Sp2 | InFIFO Status_ Sp1 | OutFIFO Status_ Sp1 |

| Bit | Description |
|---|---|
| OutFIFOStatus_Sp1 | 0: Indicates the Bulk-Out FIFO is filled with Tx data. (Serial Port-1)<br>1: Indicates the Bulk-Out FIFO is empty. |
| InFIFOStatus_Sp1 | 0: Indicates the Bulk-In FIFO is empty. (Serial Port-1)<br>1: Indicates the Bulk-In FIFO is filled with Rx data. |
| OutFIFOStatus_Sp2 | 0: Indicates the Bulk-Out FIFO is filled with Tx data. (Serial Port-2)<br>1: Indicates the Bulk-Out FIFO is empty. |
| InFIFOStatus_Sp2 | 0: Indicates the Bulk-In FIFO is empty. (Serial Port-2)<br>1: Indicates the Bulk-In FIFO is filled with Rx data. |
| OutFIFOStatus_Sp3 | 0: Indicates the Bulk-Out FIFO is filled with Tx data. (Serial Port-3)<br>1: Indicates the Bulk-Out FIFO is empty. |
| InFIFOStatus_Sp3 | 0: Indicates the Bulk-In FIFO is empty. (Serial Port-3)<br>1: Indicates the Bulk-In FIFO is filled with Rx data. |
| OutFIFOStatus_Sp4 | 0: Indicates the Bulk-Out FIFO is filled with Tx data. (Serial Port-4)<br>1: Indicates the Bulk-Out FIFO is empty. |
| InFIFOStatus_Sp4 | 0: Indicates the Bulk-In FIFO is empty. (Serial Port-4)<br>1: Indicates the Bulk-In FIFO is filled with Rx data. |

Alternatively, the software has the provision of disabling the DCR1[6] bit and receiving only the first 5 Bytes from the Status Endpoint. In this case, the software can still read the status of the Bulk-In and Bulk-Out FIFOs for each endpoint through a Vendor Specific read to the internal registers described below.

**bi_fifo_stat1 (2, 3, 4) Register**

**Register:** bi_fifo_stat1 (2, 3, 4)
**Description:** Bulk-In FIFO Status Register for SP1 (2, 3, 4)
**Register Index:** 50 (52, 54, 56)
**Permissions:** Read
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Number of Bytes available in Bulk-In FIFO for Endpoint 1 (3, 5, 7) | | | | | | | |

**bo_fifo_stat1 (2, 3, 4) Register**

**Register:** bo_fifo_stat1 (2, 3, 4)
**Description:** Bulk-Out FIFO Status Register for SP1 (2, 3, 4)
**Register Index:** 51 (53, 55, 57)
**Permissions:** Read
**Application Number:** 0
**Default Value:** 0x00

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Number of Bytes available in Bulk-Out FIFO for Endpoint 2 (4, 6, 8) | | | | | | | |

**Note:** These registers are provided for diagnostic purposes only.

Each of these registers are 8-bits wide and can support up to 256 bytes.

| Register Name | Index | # of Bytes in |
|---|---|---|
| bi_fifo_stat1 | 50 | Bulk-In FIFO 1 |
| bo_fifo_stat1 | 51 | Bulk-Out FIFO 1 |
| bi_fifo_stat2 | 52 | Bulk-In FIFO 2 |
| bo_fifo_stat2 | 53 | Bulk-Out FIFO 2 |
| bi_fifo_stat3 | 54 | Bulk-In FIFO 3 |
| bo_fifo_stat3 | 55 | Bulk-Out FIFO 3 |
| bi_fifo_stat4 | 56 | Bulk-In FIFO 4 |
| bo_fifo_stat4 | 57 | Bulk-Out FIFO 4 |

**EEPROM Content Layout**

| Bytes | # of Bytes | Name | Description |
|---|---|---|---|
| [1:0] | 2 | EE Check | EEPROM Present Check value = 0x9710 |
| [3:2] | 2 | VID | Vendor ID = 0x9710 |
| [5:4] | 2 | PID | Product ID = 0x7840 |
| [7:6] | 2 | RN | Release Number in BCD format = 0x0001 |
| 8 | 1 | SER1_DCR0 | Device Configuration Registers (SER1_DCR0) |
| 9 | 1 | SER1_DCR1 | Device Configuration Registers (SER1_DCR1) |
| 10 | 1 | SER1_DCR2 | Device Configuration Registers (SER1_DCR2) |
| 11 | 1 | SER2_DCR0 | Device Configuration Registers (SER2_DCR0) |
| 12 | 1 | SER2_DCR1 | Device Configuration Registers (SER2_DCR1) |
| 13 | 1 | SER2_DCR2 | Device Configuration Registers (SER2_DCR2) |
| 14 | 1 | SER3_DCR0 | Device Configuration Registers (SER3_DCR0) |
| 15 | 1 | SER3_DCR1 | Device Configuration Registers (SER3_DCR1) |
| 16 | 1 | SER3_DCR2 | Device Configuration Registers (SER3_DCR2) |
| 17 | 1 | SER4_DCR0 | Device Configuration Registers (SER4_DCR0) |
| 18 | 1 | SER4_DCR1 | Device Configuration Registers (SER4_DCR1) |
| 19 | 1 | SER4_DCR2 | Device Configuration Registers (SER4_DCR2) |
| 20 | 1 | intr_pg_fs | Binterval value for Full Speed |
| 21 | 1 | intr_pg_hs | Binterval value for High Speed |
| [23:22] | 2 | Language ID | Language ID in HEX Format (0x0409 default) |
| [71:24] | 48 | Manufacture ID | “MosChip Semiconductor” in UNICODE |
| [113:72] | 42 | Product Name | “USB-Serial Controller” in UNICODE |
| [129:114] | 16 | Serial Number | “X7X6X5X4X3X2X1X0” in UNICODE |

### EEPROM Contents for MCS7840 (Example Contents)

EE_Check,
VID,
PID,
RN,
SER1_DRC0, SER1_DRC1, SER1_DRC2,
SER2_DRC0, SER2_DRC1, SER2_DRC2,
SER3_DRC0, SER3_DRC1, SER3_DRC2,
SER4_DRC0, SER4_DRC1, SER4_DRC2,
INTR_PG_FS,
INTR_PG_HS,
Language ID,
Manufacture ID,

| M | o | s | C | h | i | p |
|---|---|---|---|---|---|---|
| 4D | 6F | 73 | 43 | 68 | 69 | 70 |

| | S | e | m | i | c | o | n | d | u | c | t | o | r |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 20 | 53 | 65 | 6D | 69 | 63 | 6F | 6E | 64 | 75 | 63 | 74 | 6F | 72 |

Product Name,

| U | S | B | - | S | e | r | i | a | l |
|---|---|---|---|---|---|---|---|---|---|
| 55 | 53 | 42 | 2D | 53 | 65 | 72 | 69 | 61 | 6C |

| | C | o | n | t | r | o | l | l | e | r |
|---|---|---|---|---|---|---|---|---|---|---|
| 20 | 43 | 6F | 6E | 74 | 72 | 6F | 6C | 6C | 65 | 72 |

Serial Number

| Location | HEX | ASCII |
|---|---|---|
| 0 | 10 | |
| 1 | 97 | |
| 2 | 10 | |
| 3 | 97 | |
| 4 | 40 | |
| 5 | 78 | |
| 6 | 01 | |
| 7 | 00 | |
| 8 | 01 | |
| 9 | 85 | |
| 10 | 24 | |
| 11 | 01 | |
| 12 | 80 | |
| 13 | 24 | |
| 14 | 01 | |
| 15 | 80 | |
| 16 | 24 | |
| 17 | 01 | |
| 18 | 80 | |
| 19 | 24 | |
| 20 | 01 | |
| 21 | 05 | |
| 22 | 09 | |
| 23 | 04 | |
| 24 | 4D | M |
| 25 | 00 | |
| 26 | 6F | o |
| 27 | 00 | |
| 28 | 73 | s |
| 29 | 00 | |
| 30 | 43 | C |
| 31 | 00 | |
| 32 | 68 | h |
| 33 | 00 | |
| 34 | 69 | i |
| 35 | 00 | |
| 36 | 70 | p |
| 37 | 00 | |
| 38 | 20 | Space |
| 39 | 00 | |
| 40 | 53 | S |
| 41 | 00 | |
| 42 | 65 | e |
| 43 | 00 | |

| Location | HEX | ASCII |
|---|---|---|
| 44 | 6D | m |
| 45 | 00 | |
| 46 | 69 | i |
| 47 | 00 | |
| 48 | 63 | c |
| 49 | 00 | |
| 50 | 6F | o |
| 51 | 00 | |
| 52 | 6E | n |
| 53 | 00 | |
| 54 | 64 | d |
| 55 | 00 | |
| 56 | 75 | u |
| 57 | 00 | |
| 58 | 63 | c |
| 59 | 00 | |
| 60 | 74 | t |
| 61 | 00 | |
| 62 | 6F | o |
| 63 | 00 | |
| 64 | 72 | r |
| 65 | 00 | |
| 66 | 20 | Space |
| 67 | 00 | |
| 68 | 20 | Space |
| 69 | 00 | |
| 70 | 20 | Space |
| 71 | 00 | |
| 72 | 55 | U |
| 73 | 00 | |
| 74 | 53 | S |
| 75 | 00 | |
| 76 | 42 | B |
| 77 | 00 | |
| 78 | 2D | - |
| 79 | 00 | |
| 80 | 53 | S |
| 81 | 00 | |
| 82 | 65 | e |
| 83 | 00 | |
| 84 | 72 | r |
| 85 | 00 | |
| 86 | 69 | i |
| 87 | 00 | |

| Location | HEX | ASCII |
|---|---|---|
| 88 | 61 | a |
| 89 | 00 | |
| 90 | 6C | l |
| 91 | 00 | |
| 92 | 20 | Space |
| 93 | 00 | |
| 94 | 43 | C |
| 95 | 00 | |
| 96 | 6F | o |
| 97 | 00 | |
| 98 | 6E | n |
| 99 | 00 | |
| 100 | 74 | t |
| 101 | 00 | |
| 102 | 72 | r |
| 103 | 00 | |
| 104 | 6F | o |
| 105 | 00 | |
| 106 | 6C | l |
| 107 | 00 | |
| 108 | 6C | l |
| 109 | 00 | |
| 110 | 65 | e |
| 111 | 00 | |
| 112 | 72 | r |
| 113 | 00 | |
| 114 | 4D | M |
| 115 | 00 | |
| 116 | 6F | o |
| 117 | 00 | |
| 118 | 73 | s |
| 119 | 00 | |
| 120 | 43 | C |
| 121 | 00 | |
| 122 | 68 | h |
| 123 | 00 | |
| 124 | 69 | i |
| 125 | 00 | |
| 126 | 70 | p |
| 127 | 00 | |
| 128 | 20 | Space |
| 129 | 00 | |
| | | |
| | | |

**Device Configuration Bit Fields and Descriptions**

Bytes 4, 5, 6 and 22-30 form twenty-four 8-bit DCR Registers. These Bytes are read from the EEPROM, and loaded into the Global Device Configuration Registers after Power-On Reset. They can be programmed by software using the following application number and register indexes as shown in the table.

| EEPROM Location | DCR Bit | DCR Name | Application Number | Register Index | Default Value |
|---|---|---|---|---|---|
| 8 | SER1_DCR[7:0] | SER1_DCR0 | 0 | 4 | 0x01 |
| 9 | SER1_DCR[15:8] | SER1_DCR1 | 0 | 5 | 0x85 |
| 10 | SER1_DCR[23:16] | SER1_DCR2 | 0 | 6 | 0x24 |
| 11 | SER2_DCR[7:0] | SER2_DCR0 | 0 | 22 | 0x01 |
| 12 | SER2_DCR[15:8] | SER2_DCR1 | 0 | 23 | 0x84 |
| 13 | SER2_DCR[23:16] | SER2_DCR2 | 0 | 24 | 0x24 |
| 14 | SER3_DCR[7:0] | SER3_DCR0 | 0 | 25 | 0x01 |
| 15 | SER3_DCR[15:8] | SER3_DCR1 | 0 | 26 | 0x84 |
| 16 | SER3_DCR[23:16] | SER3_DCR2 | 0 | 27 | 0x24 |
| 17 | SER4_DCR[7:0] | SER4_DCR0 | 0 | 28 | 0x01 |
| 18 | SER4_DCR[15:8] | SER4_DCR1 | 0 | 29 | 0x84 |
| 19 | SER4_DCR[23:16] | SER4_DCR2 | 0 | 30 | 0x24 |

The following tables describe the function of each bit in the DCR registers. There are three DCR registers for each Serial Port (IrDA). In the absence of an EEPROM, the default values are taken from the Device Configuration Registers.

### Serial Port 1 – Device Configuration Register 0

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | IrDA_ Mode | RTS_ CM | | GPIO_ Mode | | Reserved | RS_ SDM |

| DCR0 Bit | Name | Definition | Default Value |
|---|---|---|---|
| 0 | RS_ SDM | ***RS-232 / RS-422 / RS-485 Transceiver Shut-Down Mode:***<br>0: Do not shut down the transceiver Even when USB SUSPEND is engaged<br>1: Shut down the transceiver when USB SUSPEND is engaged | 1 |
| 1 | Reserved | Reserved | 0 |
| [3:2] | GPIO_ Mode | 00: GPIO = Input<br>10: GPIO = Output | 00 |
| [5:4] | RTS_ CM | ***RTSM RTS Control Method:***<br>00: RTS is controlled by Control Bit Map. Signal is active low;<br>01: RTS is controlled by Control Bit Map. Signal is active high;<br>10: Drive RTS active when Downstream Data Buffer is NOT EMPTY; Otherwise Drive RTS inactive.<br>11: Drive RTS inactive when Downstream Data Buffer is NOT EMPTY; Otherwise Drive RTS active. | 00 |
| 6 | IrDA_ Mode | 0: RS-232 / RS-422 / RS-485 Serial Port Mode.<br>1: IrDA Mode. | 0 |
| 7 | Reserved | Reserved | 0 |

### Serial Port 1 – Device Configuration Register 1

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | Interrupt IN Endpoint Status | PLL_ Power-Down Bypass Control | RW_ INHB | Tx_I_ PMG | | GPIO_I_ PMG | |

| DCR1 Bit | Name | Definition | Default Value |
|---|---|---|---|
| [1:0] | GPIO_I_ PMG | ***These two bits set the output current of the GPIO lines:***<br>00: 6 mA<br>01: 8 mA (Default)<br>10: 10 mA<br>11: 12 mA | 01 |
| [3:2] | Tx_I_ PMG | ***These two bits set the output current of Serial output signals TxD, DTR_n and RTS_n:***<br>00: 6 mA<br>01: 8 mA (Default)<br>10: 10 mA<br>11: 12 mA | 01 |
| 4 | RW_ INHB | ***RW_INH Remote Wake Inhibit:***<br>0: Enable the USB Remote Wakeup function<br>1: Inhibit the USB Remote Wakeup function | 0 |
| 5 | PLL_ Power-Down Bypass Control | 0: Enables PLL Power-Down<br>1: Disables PLL Power-Down | 0 |
| 6 | Interrupt IN Endpoint Status | 0: Interrupt Endpoint returns 5 Bytes of data.<br>1: Interrupt Endpoint returns 5 Bytes + 8 Bytes of the Bulk-In/Out memory controller status | 0 |
| 7 | Reserved | Reserved | 1 |

### Serial Port 1 – Device Configuration Register 2

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| SHDN_ POL | Reserved | RWU_ Mode | EWU_ Rx | EWU_ DSR | EWU_ RI | EWU_ DCD | EWU_ CTS |

| DCR2 Bit | Name | Definition | Default Value |
|---|---|---|---|
| 0 | EWU_ CTS | ***Enable Wake Up Trigger on CTS:***<br>0: Disabled<br>1: Enable Wake Up Trigger on CTS State Changes. | 0 |
| 1 | EWU_ DCD | ***Enable Wake Up Trigger on DCD:***<br>0: Disabled<br>1: Enable Wake Up Trigger on DCD State Changes. | 0 |
| 2 | EWU_ RI | ***Enable Wake Up Trigger on RI:***<br>0: Disabled<br>1: Enable Wake Up Trigger on RI State Changes. | 1 |
| 3 | EWU_ DSR | ***Enable Wake Up Trigger on DSR:***<br>0: Disabled<br>1: Enable Wake Up Trigger on DSR State Changes. | 0 |
| 4 | EWU_ Rx | ***Enable Wake Up Trigger on RXD:***<br>0: Disabled<br>1: Enable Wake Up Trigger on RXD State Changes. | 0 |
| 5 | RWU_ Mode | ***Remote Wakeup Mode:***<br>0: Engages Remote Wakeup, The device issues Disconnect Signal.<br>1: Engages Remote Wakeup, The device issues Resume Signal. | 1 |
| 6 | Reserved | Reserved | 0 |
| 7 | SHDN_ POL | ***SHDN Polarity:***<br>0: Pin 12 Active Low Shut-Down Signal.<br>1: Pin 12 Active High Shut-Down Signal. | 0 |

Note: Wake up defined above can work only when DCR0[6] = 0 and DCR1[4] = 0.

**Serial Port (2, 3, & 4) – Device Configuration Register 0**

The Configuration Registers for these three Serial Ports are all identical.
They are very similar to Serial Port 1, but have a few less configuration options.

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | IrDA_ Mode | RTS_ CM | | Reserved | | Reserved | RS_ SDM |

| DCR0 Bit | Name | Definition | Default Value |
|---|---|---|---|
| 0 | RS_ SDM | ***RS-232 / RS-422 / RS-485 Transceiver Shut-Down Mode:***<br>0: Do not shut down the transceiver Even when USB SUSPEND is engaged<br>1: Shut down the transceiver when USB SUSPEND is engaged | 1 |
| 1 | Reserved | Reserved | 0 |
| [3:2] | Reserved | Reserved | 00 |
| [5:4] | RTS_ CM | ***RTSM RTS Control Method:***<br>00: RTS is controlled by Control Bit Map. Signal is active low;<br>01: RTS is controlled by Control Bit Map. Signal is active high;<br>10: Drive RTS active when Downstream Data Buffer is NOT EMPTY; Otherwise Drive RTS inactive.<br>11: Drive RTS inactive when Downstream Data Buffer is NOT EMPTY; Otherwise Drive RTS active. | 00 |
| 6 | IrDA_ Mode | 0: RS-232 / RS-422 / RS-485 Serial Port Mode.<br>1: IrDA Mode. | 0 |
| 7 | Reserved | Reserved | 0 |

**Serial Port (2, 3, & 4) – Device Configuration Register 1**

The Configuration Registers for these three Serial Ports are all identical.
They are very similar to Serial Port 1, but have a few less configuration options.

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | Reserved | Reserved | RW_ INHB | Tx_I_ PMG | | Reserved | |

| DCR1 Bit | Name | Definition | Default Value |
|---|---|---|---|
| [1:0] | Reserved | Reserved | 00 |
| [3:2] | Tx_I_ PMG | ***These two bits set the output current of Serial output signals TxD, DTR_n and RTS_n:***<br>00: 6 mA<br>01: 8 mA (Default)<br>10: 10 mA<br>11: 12 mA | 01 |
| 4 | RW_ INHB | ***RW_INH Remote Wake Inhibit:***<br>0: Enable the USB Remote Wakeup function<br>1: Inhibit the USB Remote Wakeup function | 0 |
| 5 | Reserved | Reserved | 0 |
| 6 | Reserved | Reserved | 0 |
| 7 | Reserved | Reserved | 1 |

**Serial Port (2, 3, & 4) – Device Configuration Register 2**

The Configuration Registers for these three Serial Ports are all identical.
They are very similar to Serial Port 1, but have a few less configuration options.

| Bit[7] | Bit[6] | Bit[5] | Bit[4] | Bit[3] | Bit[2] | Bit[1] | Bit[0] |
|---|---|---|---|---|---|---|---|
| Reserved | Reserved | RWU_ Mode | EWU_ Rx | EWU_ DSR | EWU_ RI | EWU_ DCD | EWU_ CTS |

| DCR2 Bit | Name | Definition | Default Value |
|---|---|---|---|
| 0 | EWU_ CTS | ***Enable Wake Up Trigger on CTS:***<br>0: Disabled<br>1: Enable Wake Up Trigger on CTS State Changes. | 0 |
| 1 | EWU_ DCD | ***Enable Wake Up Trigger on DCD:***<br>0: Disabled<br>1: Enable Wake Up Trigger on DCD State Changes. | 0 |
| 2 | EWU_ RI | ***Enable Wake Up Trigger on RI:***<br>0: Disabled<br>1: Enable Wake Up Trigger on RI State Changes. | 1 |
| 3 | EWU_ DSR | ***Enable Wake Up Trigger on DSR:***<br>0: Disabled<br>1: Enable Wake Up Trigger on DSR State Changes. | 0 |
| 4 | EWU_ Rx | ***Enable Wake Up Trigger on RXD:***<br>0: Disabled<br>1: Enable Wake Up Trigger on RXD State Changes. | 0 |
| 5 | RWU_ Mode | ***Remote Wakeup Mode:***<br>0: Engages Remote Wakeup,<br>The device issues Disconnect Signal.<br>1: engages remote wakeup,<br>the Device issues resume signal. | 1 |
| 6 | Reserved | Reserved | 0 |
| 7 | Reserved | Reserved | 0 |

Note: Wake up defined above can work only when DCR0[6] = 0 and DCR1[4] = 0.

**USB Software Access For Control Endpoint Registers**

***Register Write Access***

| Register Name | BmRequestType | brequest | wValue | wIndex | wLength |
|---|---|---|---|---|---|
| sp1_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0000 | 16'h0000 |
| sp2_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0008 | 16'h0000 |
| sp3_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h000A | 16'h0000 |
| sp4_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h000C | 16'h0000 |
| control_reg1 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0001 | 16'h0000 |
| control_reg2 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0009 | 16'h0000 |
| control_reg3 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h000B | 16'h0000 |
| control_reg4 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h000D | 16'h0000 |
| ping_pong_high | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0002 | 16'h0000 |
| ping_pong_low* | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0003 | 16'h0000 |
| pll_m_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h000E | 16'h0000 |
| pll_n_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0010 | 16'h0000 |
| clk_mux_reg* | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0012 | 16'h0000 |
| clk_select_reg1* | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0013 | 16'h0000 |
| clk_select_reg2* | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0014 | 16'h0000 |
| sp1_ICG_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h002C | 16'h0000 |
| sp2_ICG_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h002D | 16'h0000 |
| sp3_ICG_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h002E | 16'h0000 |
| sp4_ICG_reg | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h002F | 16'h0000 |
| Rx_sampling_reg1 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0030 | 16'h0000 |
| Rx_sampling_reg2 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h0031 | 16'h0000 |
| zero_len_reg1 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h003A | 16'h0000 |
| zero_len_reg2 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h003B | 16'h0000 |
| zero_len_reg3 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h003C | 16'h0000 |
| zero_len_reg4 | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h003D | 16'h0000 |
| zero_len_flag_en | 8'h40 | 8'h0E | 16'h0000_8'hxx | 16'h003E | 16'h0000 |

**Note: ***

- pll_m_reg, pll_n_reg: Only bits [5:0] can be written.
- clk_mux_reg: Only bits [3:0] can be written.
- clk_select_reg1, clk_select_reg2: Only bits [5:0] can be written.
- ping_pong_low: Only bits [6:0] can be written.

***Register Read Access***

| Register Name | BmRequestType | brequest | wValue | wIndex | wLength |
|---|---|---|---|---|---|
| sp1_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0000 | 16'h0001 |
| sp2_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0008 | 16'h0001 |
| sp3_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h000A | 16'h0001 |
| sp4_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h000C | 16'h0001 |
| control_reg1 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0001 | 16'h0001 |
| control_reg2 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0009 | 16'h0001 |
| control_reg3 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h000B | 16'h0001 |
| control_reg4 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h000D | 16'h0001 |
| ping_pong_high | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0002 | 16'h0001 |
| ping_pong_low | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0003 | 16'h0001 |
| pll_m_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h000E | 16'h0001 |
| pll_n_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0010 | 16'h0001 |
| clk_mux_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0012 | 16'h0001 |
| clk_select_reg1 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0013 | 16'h0001 |
| clk_select_reg2 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0014 | 16'h0001 |
| sp1_ICG_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h002C | 16'h0001 |
| sp2_ICG_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h002D | 16'h0001 |
| sp3_ICG_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h002E | 16'h0001 |
| sp4_ICG_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h002F | 16'h0001 |
| mode_reg* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h002B | 16'h0001 |
| Rx_sampling_reg1 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0030 | 16'h0001 |
| Rx_sampling_reg2 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0031 | 16'h0001 |
| bi_fifo_stat1* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0032 | 16'h0001 |
| bo_fifo_stat1* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0033 | 16'h0001 |
| bi_fifo_stat2* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0034 | 16'h0001 |
| bo_fifo_stat2* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0035 | 16'h0001 |
| bi_fifo_stat3* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0036 | 16'h0001 |
| bo_fifo_stat3* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0037 | 16'h0001 |
| bi_fifo_stat4* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0038 | 16'h0001 |
| bo_fifo_stat4* | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h0039 | 16'h0001 |
| zero_len_reg1 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h003A | 16'h0001 |
| zero_len_reg2 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h003B | 16'h0001 |
| zero_len_reg3 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h003C | 16'h0001 |
| zero_len_reg4 | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h003D | 16'h0001 |
| zero_len_flag_en_reg | 8'hC0 | 8'h0D | 16'h0000_8'h00 | 16'h003E | 16'h0001 |

**Note: ***

- mode_reg : Read-Only Register
- bi_fifo_stat1, bi_fifo_stat2, bi_fifo_stat3, bi_fifo_stat4: Read-Only Register
- bo_fifo_stat1,bo_fifo_stat2, bo_fifo_stat3, bo_fifo_stat4: Read-Only Register

## Electrical Specifications

### Absolute Maximum Ratings:

| | |
|---|---|
| Core Power Supply ($Vcc_K$) | -0.3 to 2.16 V |
| Power Supply of 3.3V I/O ($Vcc_{3IO}$) | -0.3 to 4.0 V |
| Input Voltage of 3.3V I/O ($Vin_3$) | -0.3 to 4.0 V |
| Input Voltage of 5V Tolerant I/O ($Vin_5$) | -0.3 to 5.8 V |
| Operating Temperature | 0 to +70 °C |
| Storage Temperature | -40 to +150 °C |
| ESD HBM (MIL-STD 883E Method 3015-7 Class 2) | 2000 V |
| ESD MM (JEDEC EIA/JESD22 A115-A) | 200 V |
| CDM (JEDEC/JESD22 C101-A) | 500 V |
| Latch-up (JESD No. 78, March 1997) | 200 mA, 1.5 x VCC |
| Junction Temperature (Tj) | 115 °C |
| Thermal Resistance of Junction to Ambient (Still Air) | 65 °C/W |

### Operating Conditions:

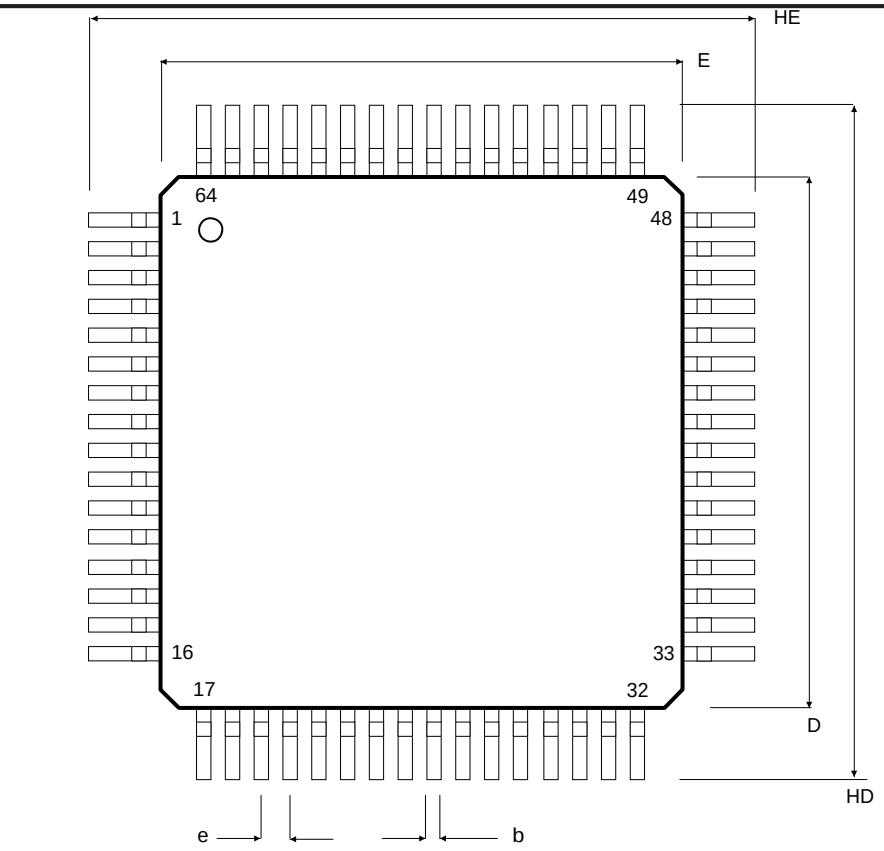
| Symbol | Parameter | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| $Vcc_{5A}$ | 5V Power Supply Input | 4.5 | 5.0 | 5.5 | V |
| $Vcc_K$ | Core Power Supply | 1.62 | 1.8 | 1.98 | V |
| $Vcc_{3IO}$ | Power Supply of 3.3V I/O | 2.97 | 3.3 | 3.63 | V |
| REG02_V18 | 1.8V Regulator Output | 1.71 | 1.8 | 1.89 | V |
| $I_{REG02\_V18}$ | 1.8V Regulator Current | | | 70 | mA |
| REG06_VCC33 | 3.3V Regulator Output | 3.14 | 3.3 | 3.46 | V |
| $I_{REG06\_VCC33}$ | 3.3V Regulator Current | | | 250 | mA |
| $I_{5V}$ | Operating current of 5V when 3.3V and 1.8V internal regulators are used. No serial load. | | 70 | | mA |
| $I_{3.3V}$ | Operating current of 3.3V. No serial load. | | 45 | | mA |
| $I_{1.8V}$ | Operating current of 1.8V. No serial load. | | 25 | | mA |

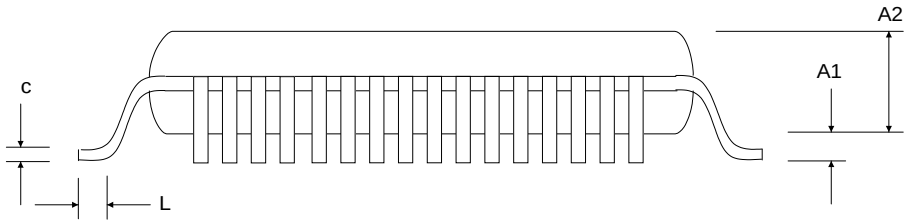
**DC Characteristics of 3.3V I/O Cells**

| Symbol | Parameter | Condition | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| $Vcc_K$ | Core Power Supply | Core Area | 1.62 | 1.8 | 1.98 | V |
| $Vcc_{3IO}$ | Power Supply | 3.3V I/O | 2.97 | 3.3 | 3.63 | V |
| $Vi_L$ | Input Low Voltage | LVTTL | | | 0.8 | V |
| $Vi_H$ | Input High Voltage | LVTTL | 2.0 | | | V |
| Vt | Switching Threshold | LVTTL | | 1.5 | | V |
| Vt-<br>Vt+ | Schmitt Trigger Threshold Voltage | LVTTL | 0.8 | 1.1<br>1.6 | 2.0 | V |
| $Vo_L$ | Output Low Voltage | $Io_L$= 2 to 24mA | | | 0.4 | V |
| $Vo_H$ | Output High Voltage | $Io_H$= -2 to -24mA | 2.4 | | | V |

**DC Characteristics of 5V Tolerant I/O Cells**

| Symbol | Parameter | Condition | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| $Vcc_{5A}$ | 5V Power Supply | 5V I/O | 4.5 | 5.0 | 5.5 | V |
| $Vi_L$ | Input Low Voltage | LVTTL | | | 0.8 | V |
| $Vi_H$ | Input High Voltage | LVTTL | 2.0 | | | V |
| Vt | Switching Threshold | LVTTL | | 1.5 | | V |
| Vt-<br>Vt+ | Schmitt Trigger Threshold Voltage | LVTTL | 0.8 | 1.1<br>1.6 | 2.0 | V |
| $Vo_H$ | Output Low Voltage | $Io_L$= 2 to 24 mA | | | 0.4 | V |
| $Vo_H$ | Output High Voltage | $Io_H$= -2 to -24 mA | 2.4 | | | V |




**64-Pin "CV" LQFP**
**Package Dimensions**

| Symbol | Millimeters | | |
|---|---|---|---|
| | MIN | TYP | MAX |
| A1 | 0.05 | | 0.15 |
| A2 | 1.35 | | 1.45 |
| b | 0.17 | | 0.27 |
| c | 0.09 | | 0.20 |
| e | | 0.50 | |
| L | 0.45 | | 0.75 |
| HD | 11.75 | | 12.25 |
| D | 9.90 | | 10.10 |
| HE | 11.75 | | 12.25 |
| E | 9.90 | | 10.10 |

**IMPORTANT NOTICE**



**DISCLAIMER**

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose, without the express written permission of ASIX. ASIX may make changes to the product specifications and descriptions in this document at any time, without notice.

ASIX provides this document "as is" without warranty of any kind, either expressed or implied, including without limitation warranties of merchantability, fitness for a particular purpose, and non-infringement.

Designers must not rely on the absence or characteristics of any features or registers marked "reserved", "undefined" or "NC". ASIX reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Always contact ASIX to get the latest document before starting a design of ASIX products.

**TRADEMARKS**

ASIX, the ASIX logo are registered trademarks of ASIX Electronics Corporation. All other trademarks are the property of their respective owners.

# Revision History

| Revision | Date | Comment |
|---|---|---|
| 0.9 | 30-May-2006 | Preliminary Release |
| 0.91 | 01-Jun-2006 | Corrected MaxPacketSize values (FS/HS) |
| 0.92 | 05-Jun-2006 | Corrected Wlength field in “Set Application Vendor Specific Command” |
| 1.0 | 28-Aug-2006 | 1. Removed Preliminary Notice.<br>2. Made change to reflect one GPIO port instead of two.<br>3. Added Driver Support entries on page 1. Made bits 2 and 1 of the MCR register reserved. Made bit 5 of the Mode register reserved.<br>4. Replaced Raid_reg1 with Rx_sampling_reg1 throughout document.<br>5. Modified product ID value in EEPROM Content Layout table.<br>6. Made bit 1 of Device Configuration register 0 reserved and added note.<br>7. Modified description of bit 1 of Device Configuration register 0. Made bit 6 of Device Configuration register 0 reserved and added note. |
| 1.1 | 16-Sept-2006 | 1. Clarified Linux Kernel support in Features<br>2. Deleted Windows CE5.0 and Vista release dates |
| 1.2 | 6-August-2007 | 1. Updated Absolute Maximum Rating table Deleted Leakage Current table Updated Operating Conditions table<br>2. Updated 3.3V DC Characteristics table<br>3. Updated 5V DC Characteristics table<br>4. Removed dimensions in Inches from Package Dimensions table<br>5. Removed ‘Confidential’ notice from all pages |
| 2.00 | 2011/08/05 | 1. Changed to ASIX Electronics Corp. logo, strings and contact information.<br>2. Added ASIX copyright legal header information.<br>3. Modified the Revision History table format.<br>4. Updated the block diagram.<br>5. Modified some descriptions in the Feature page. |
| 2.01 | 2011/11/01 | 1. Updated the ordering information. |

ASIX Electronics Corporation.

**4F, No. 8, Hsin Ann Rd., HsinChu Science Park,**
**HsinChu, Taiwan, R.O.C.**

**TEL: 886-3-5799500**
**FAX: 886-3-5799558**

**Sales Email: sales@asix.com.tw**
**Support Email: support@asix.com.tw**
**Web: http://www.asix.com.tw**