

# **TMS320C642x DSP Universal Asynchronous Receiver/Transmitter (UART)**

## **User's Guide**



Literature Number: SPRUEN6C  
December 2009



<b>Preface</b> .....	<b>6</b>
<b>1 Introduction</b> .....	<b>7</b>
1.1 Purpose of the Peripheral .....	7
1.2 Features .....	7
1.3 Functional Block Diagram .....	8
1.4 Industry Standard(s) Compliance Statement .....	8
<b>2 Peripheral Architecture</b> .....	<b>10</b>
2.1 Clock Generation and Control .....	10
2.2 Signal Descriptions .....	12
2.3 Pin Multiplexing .....	12
2.4 Protocol Description .....	12
2.5 Endianness Considerations .....	13
2.6 Operation .....	14
2.7 Reset Considerations .....	18
2.8 Initialization .....	18
2.9 Interrupt Support .....	18
2.10 DMA Event Support .....	20
2.11 Power Management .....	20
2.12 Emulation Considerations .....	20
2.13 Exception Processing .....	21
<b>3 Registers</b> .....	<b>21</b>
3.1 Receiver Buffer Register (RBR) .....	22
3.2 Transmitter Holding Register (THR) .....	23
3.3 Interrupt Enable Register (IER) .....	24
3.4 Interrupt Identification Register (IIR) .....	25
3.5 FIFO Control Register (FCR) .....	26
3.6 Line Control Register (LCR) .....	28
3.7 Modem Control Register (MCR) .....	30
3.8 Line Status Register (LSR) .....	31
3.9 Divisor Latches (DLL and DLH) .....	33
3.10 Peripheral Identification Registers (PID1 and PID2) .....	35
3.11 Power and Emulation Management Register (PWREMU_MGMT) .....	36
<b>Appendix A Revision History</b> .....	<b>37</b>

## List of Figures

1	UART Block Diagram.....	9
2	UART Clock Generation Diagram .....	10
3	Relationships Between Data Bit, BCLK, and UART Input Clock.....	11
4	UART Protocol Formats .....	13
5	UART Interface Using Autoflow Diagram .....	16
6	Autoflow Functional Timing Waveforms for RTS .....	17
7	Autoflow Functional Timing Waveforms for CTS .....	17
8	UART Interrupt Request Enable Paths .....	19
9	Receiver Buffer Register (RBR) .....	22
10	Transmitter Holding Register (THR) .....	23
11	Interrupt Enable Register (IER) .....	24
12	Interrupt Identification Register (IIR) .....	25
13	FIFO Control Register (FCR) .....	27
14	Line Control Register (LCR) .....	28
15	Modem Control Register (MCR) .....	30
16	Line Status Register (LSR).....	31
17	Divisor LSB Latch (DLL).....	34
18	Divisor MSB Latch (DLH).....	34
19	Peripheral Identification Register 1 (PID1) .....	35
20	Peripheral Identification Register 2 (PID2) .....	35
21	Power and Emulation Management Register (PWREMU_MGMT) .....	36

## List of Tables

1	UART Supported Features/Characteristics by Instance .....	8
2	Baud Rate Examples for 27 MHz UART Input Clock .....	11
3	UART Signal Descriptions .....	12
4	Address Offset Required for Each Data Size in Big-Endian Mode .....	14
5	Character Time for Word Lengths .....	15
6	UART Interrupt Requests Descriptions .....	19
7	UART Registers .....	21
8	Receiver Buffer Register (RBR) Field Descriptions .....	22
9	Transmitter Holding Register (THR) Field Descriptions .....	23
10	Interrupt Enable Register (IER) Field Descriptions .....	24
11	Interrupt Identification Register (IIR) Field Descriptions .....	25
12	Interrupt Identification and Interrupt Clearing Information .....	26
13	FIFO Control Register (FCR) Field Descriptions .....	27
14	Line Control Register (LCR) Field Descriptions .....	28
15	Relationship Between ST, EPS, and PEN Bits in LCR .....	29
16	Number of STOP Bits Generated .....	29
17	Modem Control Register (MCR) Field Descriptions .....	30
18	Line Status Register (LSR) Field Descriptions .....	31
19	Divisor LSB Latch (DLL) Field Descriptions .....	34
20	Divisor MSB Latch (DLH) Field Descriptions .....	34
21	Peripheral Identification Register 1 (PID1) Field Descriptions .....	35
22	Peripheral Identification Register 2 (PID2) Field Descriptions .....	35
23	Power and Emulation Management Register (PWREMU_MGMT) Field Descriptions .....	36
24	Document Revision History .....	37

---

---

## Read This First

---

---

### About This Manual

This document describes the universal asynchronous receiver/transmitter (UART) peripheral in the TMS320C642x Digital Signal Processor (DSP).

### Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

### Related Documentation From Texas Instruments

The following documents describe the TMS320C642x Digital Signal Processor (DSP). Copies of these documents are available on the Internet at [www.ti.com](http://www.ti.com). *Tip:* Enter the literature number in the search box provided at [www.ti.com](http://www.ti.com).

The current documentation that describes the C642x DSP, related peripherals, and other technical collateral, is available in the C6000 DSP product folder at: [www.ti.com/c6000](http://www.ti.com/c6000).

**[SPRUJEM3](#)** — *TMS320C642x DSP Peripherals Overview Reference Guide*. Provides an overview and briefly describes the peripherals available on the TMS320C642x Digital Signal Processor (DSP).

**[SPRAA84](#)** — *TMS320C64x to TMS320C64x+ CPU Migration Guide*. Describes migrating from the Texas Instruments TMS320C64x digital signal processor (DSP) to the TMS320C64x+ DSP. The objective of this document is to indicate differences between the two cores. Functionality in the devices that is identical is not included.

**[SPRU732](#)** — *TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide*. Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C64x and TMS320C64x+ digital signal processors (DSPs) of the TMS320C6000 DSP family. The C64x/C64x+ DSP generation comprises fixed-point devices in the C6000 DSP platform. The C64x+ DSP is an enhancement of the C64x DSP with added functionality and an expanded instruction set.

**[SPRU871](#)** — *TMS320C64x+ DSP Megamodule Reference Guide*. Describes the TMS320C64x+ digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

# **Universal Asynchronous Receiver/Transmitter (UART)**

---

---

---

## **1 Introduction**

This document describes the universal asynchronous receiver/transmitter (UART) peripheral in the TMS320C642x Digital Signal Processor (DSP).

### **1.1 Purpose of the Peripheral**

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element, which in turn is a functional upgrade of the TL16C450. Functionally similar to the TL16C450 on power up (single character or TL16C450 mode), the UART can be placed in an alternate FIFO (TL16C550) mode. This relieves the CPU of excessive software overhead by buffering received and transmitted characters. The receiver and transmitter FIFOs store up to 16 bytes including three additional bits of error status per byte for the receiver FIFO.

The UART performs serial-to-parallel conversions on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU. The CPU can read the UART status at any time. The UART includes control capability and a processor interrupt system that can be tailored to minimize software management of the communications link.

The UART includes a programmable baud generator capable of dividing the UART input clock by divisors from 1 to 65 535 and producing a 16 x reference clock for the internal transmitter and receiver logic. For detailed timing and electrical specifications for the UART, see the device specific data manual.

### **1.2 Features**

The UART peripheral has the following features:

- Programmable baud rates up to 128 kbps (frequency pre-scale values from 1 to 65535)
- Fully 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
- 16-byte depth transmitter and receiver FIFOs:
  - The UART can be operated with or without the FIFOs
  - 1, 4, 8, or 14 byte selectable receiver FIFO trigger level for autoflow control and DMA
- DMA signaling capability for both received and transmitted data
- CPU interrupt capability for both received and transmitted data
- Operates in either big-endian or little-endian mode
- False START bit detection
- Line break generation and detection
- Internal diagnostic capabilities:
  - Loopback controls for communications link fault isolation
  - Break, parity, overrun, and framing error simulation
- Programmable autoflow control using CTS and RTS signals (not supported on all UARTs. See the device-specific data manual for supported features.)
- Modem control functions using CTS and RTS signals (not supported on all UARTs. See the device-specific data manual for supported features.)

**Table 1** summarizes the capabilities supported on the UART. Note that the number of UARTs and their supported features vary on each device, see the device-specific data manual for more details.

**Table 1. UART Supported Features/Characteristics by Instance**

Feature	Support
5, 6, 7 or 8-bit characters	Supported
Even, odd, or no PARITY bit	Supported
1, 1.5, or 2 STOP bit generation	Supported
Line break generation and detection	Supported
Internal loop back	Supported
DMA sync events for both received and transmitted data	Supported
1, 4, 8, or 14 byte selectable receiver FIFO trigger level	Supported
Polling/Interrupt	Supported
Max speed 128 kbps	Supported
Modem control functions using CTS and RTS	Supported <sup>(1)</sup>
Autoflow control using CTS and RTS	Supported <sup>(1)</sup>
DTR and DSR	Not supported
Ring indication	Not supported
Carrier detection	Not supported
Single-character transfer mode (mode 0) in DMA mode	Not supported

<sup>(1)</sup> Not supported on all UARTs. See the device-specific data manual for supported features.

### 1.3 **Functional Block Diagram**

A functional block diagram of the UART is shown in [Figure 1](#).

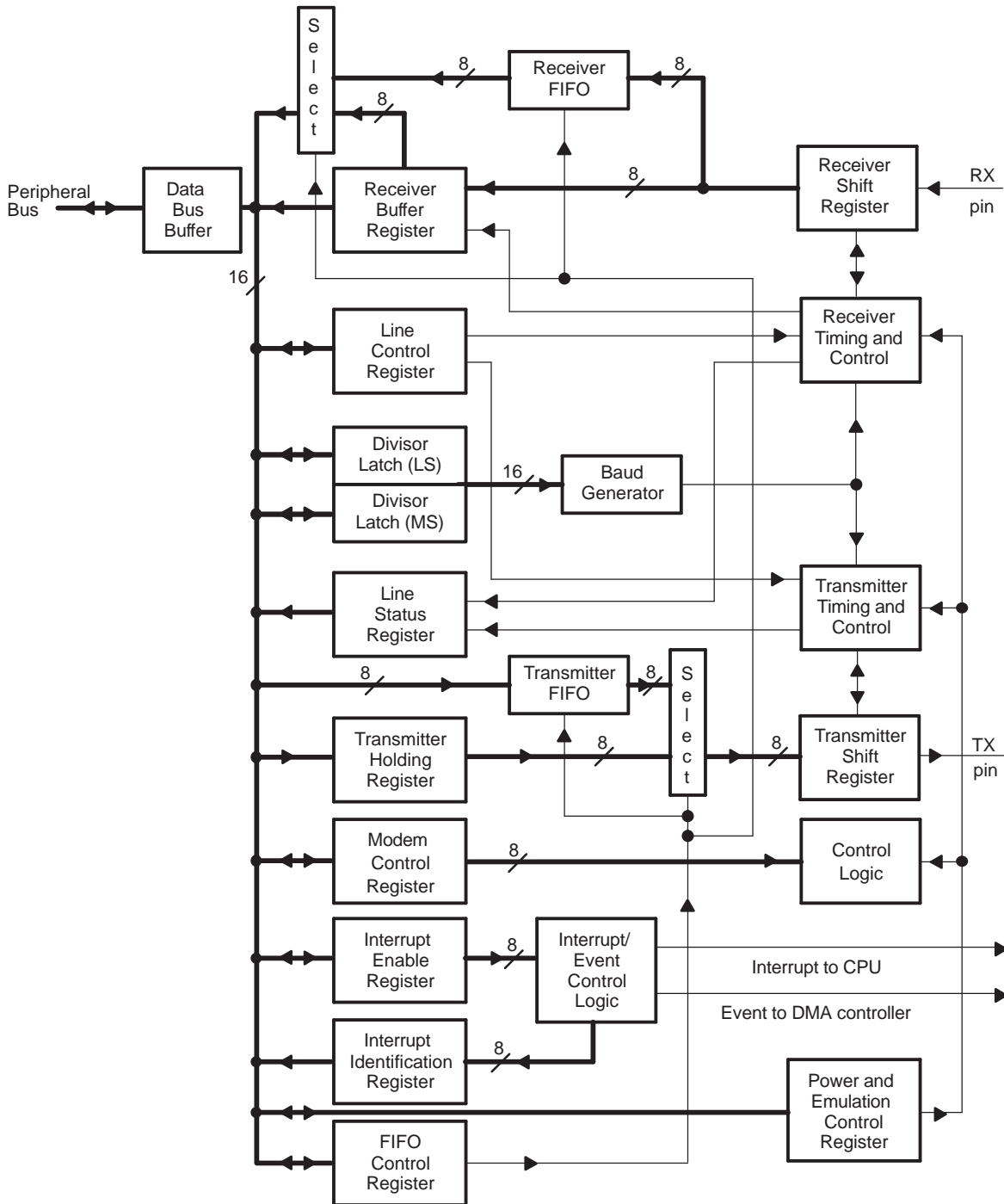
### 1.4 **Industry Standard(s) Compliance Statement**

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element, which is a functional upgrade of the TL16C450. Any deviations in supported functions are indicated in [Table 1](#).

The information in this document assumes the reader is familiar with these standards.



Figure 1. UART Block Diagram



## 2 Peripheral Architecture

### 2.1 Clock Generation and Control

The UART bit clock is sourced from the PLLC1 AUXCLK. It supports up to 128 kbps maximum data rate.

Figure 2 is a conceptual clock generation diagram for the UART. The processor clock generator receives a signal from an external clock source and produces a UART input clock with a programmed frequency. The UART contains a programmable baud generator that takes an input clock and divides it by a divisor in the range between 1 and  $(2^{16} - 1)$  to produce a baud clock (BCLK). The frequency of BCLK is sixteen times ( $16 \times$ ) the baud rate; each received or transmitted bit lasts 16 BCLK cycles. When the UART is receiving, the bit is sampled in the 8th BCLK cycle. The formula to calculate the divisor is:

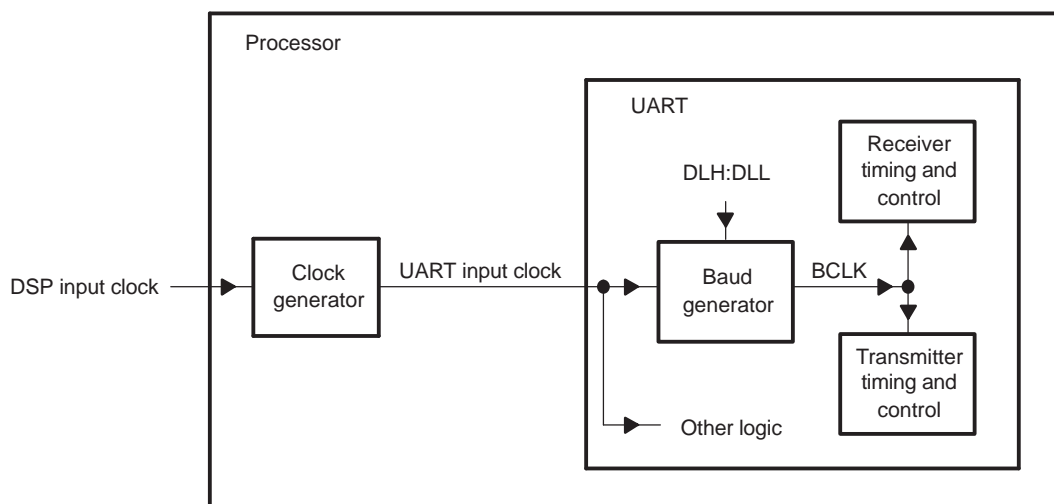
$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 16} \quad (1)$$

Two 8-bit register fields (DLH and DLL), called divisor latches, hold this 16-bit divisor. DLH holds the most significant bits of the divisor, and DLL holds the least significant bits of the divisor. For information about these register fields, see Section 3. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

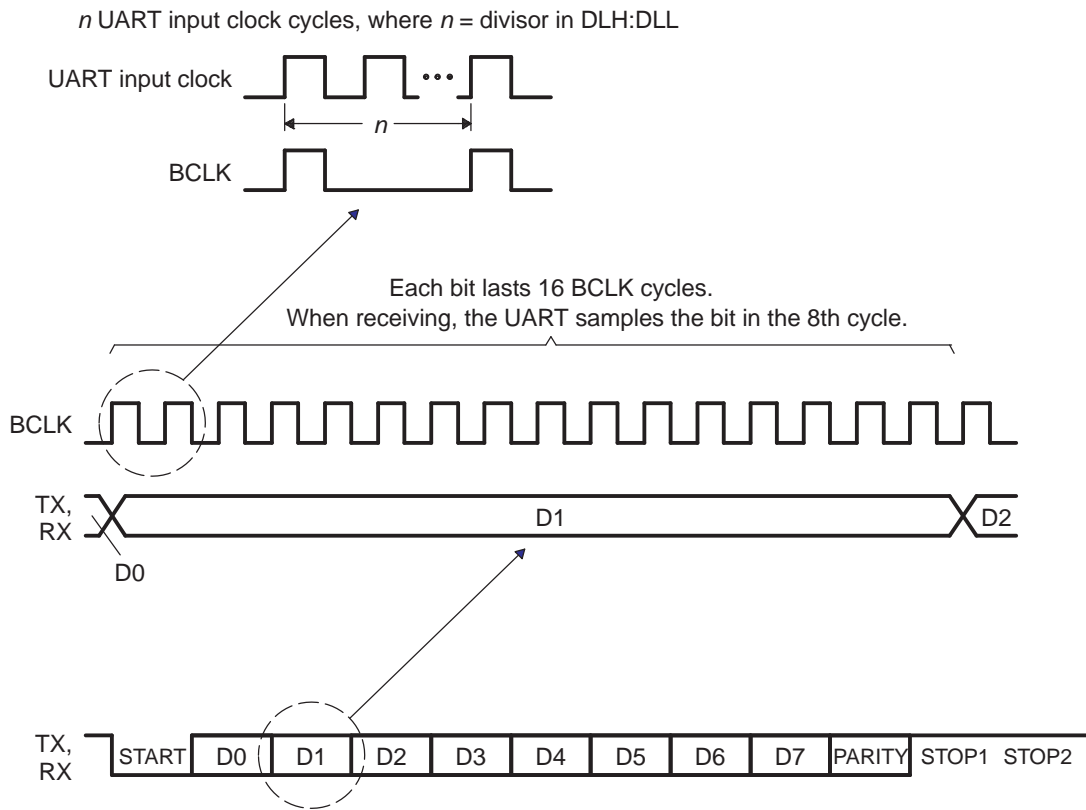
Figure 3 summarizes the relationship between the transferred data bit, BCLK, and the UART input clock.

Example baud rates and divisor values relative to a 27-MHz UART input clock are shown in Table 2.

**Figure 2. UART Clock Generation Diagram**



**Figure 3. Relationships Between Data Bit, BCLK, and UART Input Clock**



**Table 2. Baud Rate Examples for 27 MHz UART Input Clock**

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	703	2400.427	0.018
4800	352	4794.034	-0.124
9600	176	9588.068	-0.124
19200	88	19176.14	-0.124
38400	44	38352.27	-0.124
56000	30	56250	0.446
128000	13	129807.7	1.412

## 2.2 Signal Descriptions

The UARTs utilize a minimal number of signal connections to interface with external devices. The UART signal descriptions are included in [Table 3](#). Note that the number of UARTs and their supported features vary on each device, see the device-specific data manual for more details.

**Table 3. UART Signal Descriptions**

Signal Name <sup>(1)</sup>	Signal Type	Function
UTXD <sub>n</sub>	Output	Serial data transmit
URXD <sub>n</sub>	Input	Serial data receive
UCTS <sub>n</sub>	Input	Clear-to-Send handshaking signal
URTS <sub>n</sub>	Output	Request-to-Send handshaking signal

<sup>(1)</sup> The value n indicates the applicable UART; that is, UART0, UART1, etc.

## 2.3 Pin Multiplexing

On the C642x DSP extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. Refer to the device-specific data manual to determine how pin multiplexing affects the UART.

## 2.4 Protocol Description

### 2.4.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

### 2.4.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

### 2.4.3 Data Format

The UART transmits in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1, 1.5, 2)

It transmits 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1, 1.5, or 2 STOP bits, depending on the STOP bit selection.

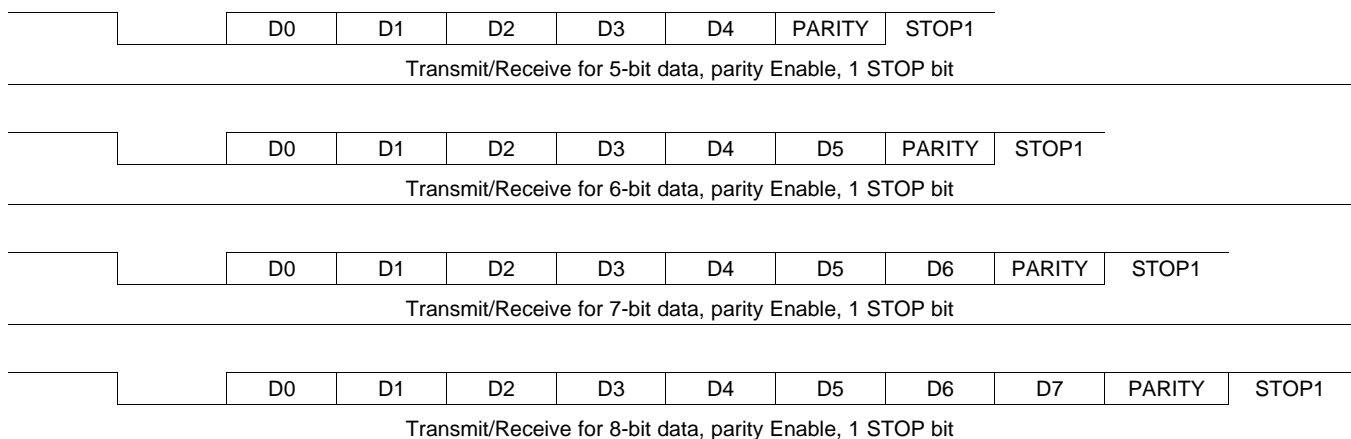
The UART receives in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1)

It receives 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1 STOP bit.

The protocol formats are shown in [Figure 4](#)

**Figure 4. UART Protocol Formats**



## 2.5 Endianness Considerations

Since the UART transfers 8-bit data externally, the endianness operating mode selected within the C642x DSP is not a concern to the external devices connected to the C642x UART interface.

Internal to the C642x DSP, however, the selected endianness operating mode can affect the way in which data is transferred to and from the UART.

Since the C642x DSP utilizes 32-bit data words, endianness is only a consideration when an attempt is made to access data of less than 4 bytes at a time. For example, when performing 32-bit accesses to the UART memory-mapped registers with the CPU, endianness is not a consideration in these cases. However, when performing 8-, 16-, 24-bit accesses, such as with the EDMA, endianness must be considered for these accesses to occur correctly.

Since less than 4-byte accesses within a 32-bit word are least-significant-byte justified when using little-endian operating mode, data is accessed with the proper alignment when data is addressed normally. However, accordingly, when performing less than 32-bit transfers in big-endian operating mode, addresses for these accesses must have an offset added to them in order to access data with the proper alignment.

Specifically, when accessing a single-byte location in big-endian mode, an address offset of 3 must be added to the normal memory-mapped register address. When accessing a 16-bit value, an address offset of 2 must be added, and so on. [Table 4](#) shows the address offset required for each data size when accessing the UART in big-endian mode.

**Table 4. Address Offset Required for Each Data Size in Big-Endian Mode**

Access Bit Width	Address Offset
8 bits	3
16 bits	2
24 bits	1
32 bits	0

Within the UART peripheral, although the memory-mapped registers occupy a full 32-bit data word, all but two of the registers contain only eight allocated bits and the rest of the bits are reserved. The peripheral identification register 1 (PID1) and the power and emulation management register (PWREMU\_MGMT) contain 16 allocated bits with the remaining 16 bits being reserved. Therefore, if any of these registers are accessed in less than 32-bit data widths in big-endian mode, the correct addressing offset, described in [Table 4](#), must be used for these accesses.

## 2.6 Operation

### 2.6.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

THR receives data from the internal data bus, and when TSR is ready, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the TX pin. In the non-FIFO mode, if THR is empty and the THR empty interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO.

### 2.6.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16x receiver clock. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

RSR receives the data bits from the RX pin. Then RSR concatenates the data bits and moves the resulting value into RBR (or the receiver FIFO). The UART also stores three bits of error status information next to each received character, to record a parity error, framing error, or break.

In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

### 2.6.3 FIFO Modes

The following two modes can be used for servicing the receiver and transmitter FIFOs:

- FIFO interrupt mode. The FIFO is enabled and the associated interrupts are enabled. Interrupts are sent to the CPU to indicate when specific events occur.
- FIFO poll mode. The FIFO is enabled but the associated interrupts are disabled. The CPU polls status bits to detect specific events.

Because the receiver FIFO and the transmitter FIFO are controlled separately, either one or both can be placed into the interrupt mode or the poll mode.

#### 2.6.3.1 FIFO Interrupt Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are enabled in the interrupt enable register (IER), the interrupt mode is selected for the receiver FIFO. The following are important points about the receiver interrupts:

- The receiver data-ready interrupt is issued to the CPU when the FIFO has reached the trigger level that is programmed in FCR. It is cleared when the CPU or the DMA controller reads enough characters from the FIFO such that the FIFO drops below its programmed trigger level.
- The receiver line status interrupt is generated in response to an overrun error, a parity error, a framing error, or a break. This interrupt has higher priority than the receiver data-ready interrupt. For details, see [Section 2.9](#).
- The data-ready (DR) bit in the line status register (LSR) indicates the presence or absence of characters in the receiver FIFO. The DR bit is set when a character is transferred from the receiver shift register (RSR) to the empty receiver FIFO. The DR bit remains set until the FIFO is empty again.
- A receiver time-out interrupt occurs if all of the following conditions exist:
  - At least one character is in the FIFO,
  - The most recent character was received more than four continuous character times ago. A character time is the time allotted for 1 START bit,  $n$  data bits, 1 PARITY bit, and 1 STOP bit, where  $n$  depends on the word length selected with the WLS bits in the line control register (LCR). See [Table 5](#).
  - The most recent read of the FIFO has occurred more than four continuous character times before.
- Character times are calculated by using the baud rate.
- When a receiver time-out interrupt has occurred, it is cleared and the time-out timer is cleared when the CPU or the EDMA controller reads one character from the receiver FIFO. The interrupt is also cleared if a new character is received in the FIFO or if the URRST bit is cleared in the power and emulation management register (PWREMU\_MGMT).
- If a receiver time-out interrupt has not occurred, the time-out timer is cleared after a new character is received or after the CPU or EDMA reads the receiver FIFO.

When the transmitter FIFO is enabled in FCR and the transmitter holding register empty interrupt is enabled in IER, the interrupt mode is selected for the transmitter FIFO. The transmitter holding register empty interrupt occurs when the transmitter FIFO is empty. It is cleared when the transmitter hold register (THR) is loaded (1 to 16 characters may be written to the transmitter FIFO while servicing this interrupt).

**Table 5. Character Time for Word Lengths**

Word Length ( $n$ )	Character Time	Four Character Times
5	Time for 8 bits	Time for 32 bits
6	Time for 9 bits	Time for 36 bits
7	Time for 10 bits	Time for 40 bits
8	Time for 11 bits	Time for 44 bits

### 2.6.3.2 FIFO Poll Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are disabled in the interrupt enable register (IER), the poll mode is selected for the receiver FIFO. Similarly, when the transmitter FIFO is enabled and the transmitter interrupts are disabled, the transmitted FIFO is in the poll mode. In the poll mode, the CPU detects events by checking bits in the line status register (LSR):

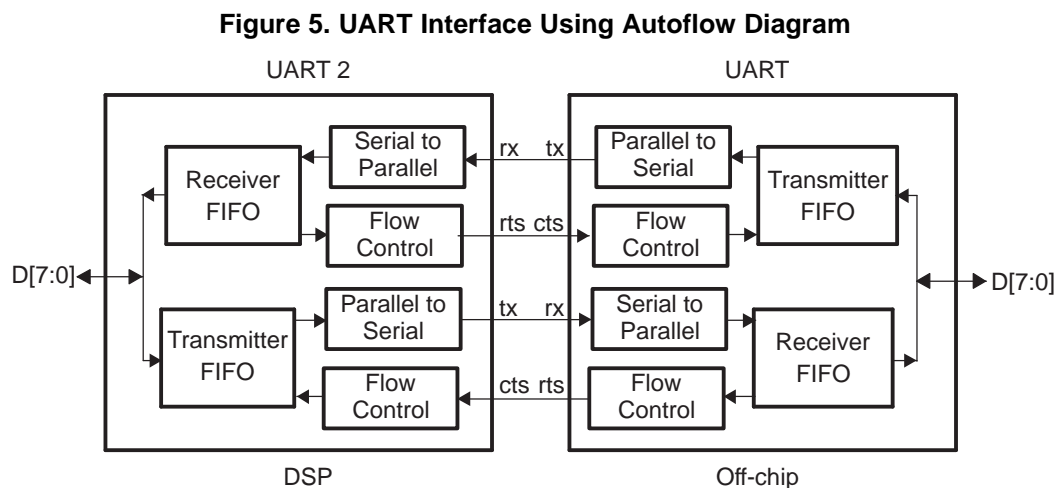
- The RXFIFOE bit indicates whether there are any errors in the receiver FIFO.
- The TEMT bit indicates that both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
- The THRE bit indicates when THR is empty.
- The BI (break), FE (framing error), PE (parity error), and OE (overrun error) bits specify which error or errors have occurred.
- The DR (data-ready) bit is set as long as there is at least one byte in the receiver FIFO.

Also, in the FIFO poll mode:

- The interrupt identification register (IIR) is not affected by any events because the interrupts are disabled.
- The UART does not indicate when the receiver FIFO trigger level is reached or when a receiver time-out occurs.

### 2.6.4 Autoflow Control (UART2 only)

The UART can employ autoflow control by connecting the CTS and RTS signals. Note that all UARTs do not support autoflow control, see the device-specific data manual for supported features. The CTS input must be active before the transmitter FIFO can transmit data. The RTS becomes active when the receiver needs more data and notifies the sending device. When RTS is connected to CTS, data transmission does not occur unless the receiver FIFO has space for the data. Therefore, when two UARTs are connected as shown in [Figure 5](#) with autoflow enabled, overrun errors are eliminated.

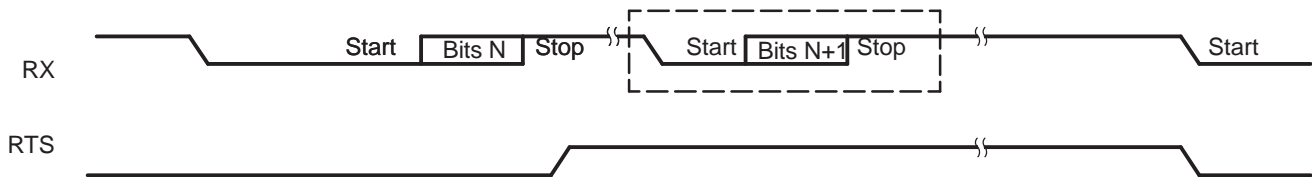




### 2.6.4.1 RTS Behavior

RTS data flow control originates in the receiver block (see Figure 1). When the receiver FIFO level reaches a trigger level of 1, 4, 8, or 14 (see Figure 6), RTS is deasserted. The sending UART may send an additional byte after the trigger level is reached (assuming the sending UART has another byte to send), because it may not recognize the deassertion of RTS until after it has begun sending the additional byte. For trigger level 1, 4, and 8, RTS is automatically reasserted once the receiver FIFO is emptied. For trigger level 14, RTS is automatically reasserted once the receiver FIFO drops below the trigger level.

**Figure 6. Autoflow Functional Timing Waveforms for RTS**

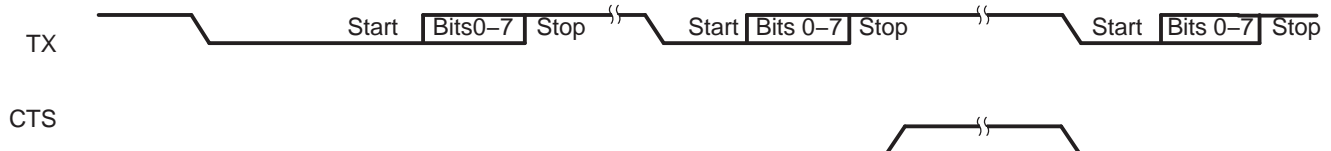


- (1) N = Receiver FIFO trigger level.
- (2) The two blocks in dashed lines cover the case where an additional byte is sent.

### 2.6.4.2 CTS Behavior

The transmitter checks CTS before sending the next data byte. If CTS is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, CTS must be released before the middle of the last STOP bit that is currently being sent (see Figure 7). When flow control is enabled, CTS level changes do not trigger interrupts because the device automatically controls its own transmitter. Without autoflow control, the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.

**Figure 7. Autoflow Functional Timing Waveforms for CTS**



- (1) When CTS is active (low), the transmitter keeps sending serial data out.
- (2) When CTS goes high before the middle of the last STOP bit of the current byte, the transmitter finishes sending the current byte but it does not send the next byte.
- (3) When CTS goes from high to low, the transmitter begins sending data again.

### 2.6.5 Loopback Control

The UART can be placed in the diagnostic mode using the LOOP bit in the modem control register (MCR), which internally connects the UART output back to the UART input. In this mode, the transmit and receive data paths, the transmitter and receiver interrupts, and the modem control interrupts can be verified without connecting to another UART.

## 2.7 Reset Considerations

### 2.7.1 Software Reset Considerations

Two bits in the power and emulation management register (PWREMU\_MGMT) control resetting the parts of the UART:

- The UTRST bit controls resetting the transmitter only. If UTRST = 1, the transmitter is active; if UTRST = 0, the transmitter is in reset.
- The URRST bit controls resetting the receiver only. If URRST = 1, the receiver is active; if URRST = 0, the receiver is in reset.

In each case, putting the receiver and/or transmitter in reset will reset the state machine of the affected portion but does not affect the UART registers.

### 2.7.2 Hardware Reset Considerations

When the processor RESET pin is asserted, the entire processor is reset and is held in the reset state until the RESET pin is released. As part of a device reset, the UART state machine is reset and the UART registers are forced to their default states. The default states of the registers are shown in [Section 3](#).

## 2.8 Initialization

The following steps are required to initialize the UART:

1. Perform the necessary device pin multiplexing setup (see the device-specific data manual).
2. Program the VDD3P3V\_PWDN register to power up the IO pins for the UART (see the device-specific data manual).
3. Set the desired baud rate by writing the appropriate clock divisor values to the divisor latch registers (DLL and DLH).
4. If the FIFOs will be used, select the desired trigger level and enable the FIFOs by writing the appropriate values to the FIFO control register (FCR). The FIFOEN bit in FCR must be set first, before the other bits in FCR are configured.
5. Choose the desired protocol settings by writing the appropriate values to the line control register (LCR).
6. If autoflow control is desired, write appropriate values to the modem control register (MCR). Note that all UARTs do not support autoflow control, see the device-specific data manual for supported features.
7. Choose the desired response to emulation suspend events by configuring the FREE bit and enable the UART by setting the UTRST and URRST bits in the power and emulation management register (PWREMU\_MGMT).

## 2.9 Interrupt Support

### 2.9.1 Interrupt Events and Requests

The UART generates the interrupt requests described in [Table 6](#). All requests are multiplexed through an arbiter to a single UART interrupt request to the CPU, as shown in [Figure 8](#). Each of the interrupt requests has an enable bit in the interrupt enable register (IER) and is recorded in the interrupt identification register (IIR).

If an interrupt occurs and the corresponding enable bit is set to 1, the interrupt request is recorded in IIR and is forwarded to the CPU. If an interrupt occurs and the corresponding enable bit is cleared to 0, the interrupt request is blocked. The interrupt request is neither recorded in IIR nor forwarded to the CPU.

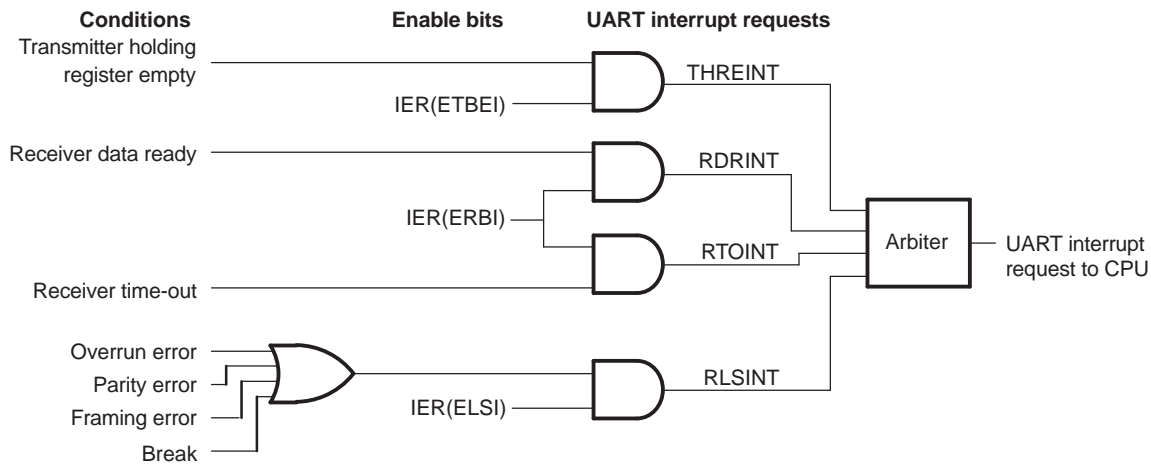
### 2.9.2 Interrupt Multiplexing

The UARTs have dedicated interrupt signals to the DSP CPU and the interrupts are not multiplexed with any other interrupt source.

**Table 6. UART Interrupt Requests Descriptions**

UART Interrupt Request	Interrupt Source	Comment
THREINT	THR-empty condition: The transmitter holding register (THR) or the transmitter FIFO is empty. All of the data has been copied from THR to the transmitter shift register (TSR).	If THREINT is enabled in IER, by setting the ETBEI bit, it is recorded in IIR. As an alternative to using THREINT, the CPU can poll the THRE bit in the line status register (LSR).
RDAINT	Receive data available in non-FIFO mode or trigger level reached in the FIFO mode.	If RDAINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. As an alternative to using RDAINT, the CPU can poll the DR bit in the line status register (LSR). In the FIFO mode, this is not a functionally equivalent alternative because the DR bit does not respond to the FIFO trigger level. The DR bit only indicates the presence or absence of unread characters.
RTOINT	Receiver time-out condition (in the FIFO mode only): No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 5), and there is at least one character in the receiver FIFO during this time.	The receiver time-out interrupt prevents the UART from waiting indefinitely, in the case when the receiver FIFO level is below the trigger level and thus does not generate a receiver data-ready interrupt. If RTOINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. There is no status bit to reflect the occurrence of a time-out condition.
RLSINT	Receiver line status condition: An overrun error, parity error, framing error, or break has occurred.	If RLSINT is enabled in IER, by setting the ELSI bit, it is recorded in IIR. As an alternative to using RLSINT, the CPU can poll the following bits in the line status register (LSR): overrun error indicator (OE), parity error indicator (PE), framing error indicator (FE), and break indicator (BI).

**Figure 8. UART Interrupt Request Enable Paths**



## 2.10 DMA Event Support

In the FIFO mode, the UART generates the following two DMA events:

- **Receive event (URXEVT):** The trigger level for the receiver FIFO (1, 4, 8, or 14 characters) is set with the RXFIFTL bit in the FIFO control register (FCR). Every time the trigger level is reached or a receiver time-out occurs, the UART sends a receive event to the EDMA controller. In response, the EDMA controller reads the data from the receiver FIFO by way of the receiver buffer register (RBR). Note that the receive event is not asserted if the data at the top of the receiver FIFO is erroneous even if the trigger level has been reached.
- **Transmit event (UTXEVT):** When the transmitter FIFO is empty (when the last byte in the transmitter FIFO has been copied to the transmitter shift register), the UART sends an UTXEVT signal to the EDMA controller. In response, the EDMA controller refills the transmitter FIFO by way of the transmitter holding register (THR). The UTXEVT signal is also sent to the DMA controller when the UART is taken out of reset using the UTRST bit in the power and emulation management register (PWREMU\_MGMT).

Activity in DMA channels can be synchronized to these events. In the non-FIFO mode, the UART generates no DMA events. Any DMA channel synchronized to either of these events must be enabled at the time the UART event is generated. Otherwise, the DMA channel will miss the event and, unless the UART generates a new event, no data transfer will occur.

## 2.11 Power Management

The UART peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the UART peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the *TMS320C642x DSP Power and Sleep Controller (PSC) User's Guide* ([SPRUEN8](#)).

## 2.12 Emulation Considerations

The FREE bit in the power and emulation management register (PWREMU\_MGMT) determines how the UART responds to an emulation suspend event such as an emulator halt or breakpoint. If FREE = 0 and a transmission is in progress, the UART halts after completing the one-word transmission; if FREE = 0 and a transmission is not in progress, the UART halts immediately. If FREE = 1, the UART does not halt and continues operating normally.

Note also that emulator accesses are essentially transparent to UART operation. Emulator read operations do not affect any register contents, status bits, or operating states. Emulator writes, however, may affect register contents and may affect UART operation, depending on what register is accessed and what value is written.

The UART registers can be read from or written to during emulation suspend events, even if the UART activity has stopped.

## 2.13 Exception Processing

### 2.13.1 Divisor Latch Not Programmed

Since the processor reset signal has no effect on the divisor latch, the divisor latch will have an unknown value after power up. If the divisor latch is not programmed after power up, the baud clock (BCLK) will not operate and will instead be set to a constant logic 1 state.

The divisor latch values should always be reinitialized following a processor reset.

### 2.13.2 Changing Operating Mode During Busy Serial Communication

Since the serial link characteristics are based on how the control registers are programmed, the UART will expect the control registers to be static while it is busy engaging in a serial communication. Therefore, changing the control registers while the module is still busy communicating with another serial device will most likely cause an error condition and should be avoided.

## 3 Registers

The system programmer has access to and control over any of the UART registers that are listed in [Table 7](#). These registers, which control UART operations, receive data, and transmit data, are available at 32-bit addresses in the device memory map. See the device-specific data manual for the memory address of these registers.

- RBR, THR, and DLL share one address. When the DLAB bit in LCR is 0, reading from the address gives the content of RBR, and writing to the address modifies THR. When DLAB = 1, all accesses at the address read or modify DLL. DLL can also be accessed with address offset 20h.
- IER and DLH share one address. When DLAB = 0, all accesses read or modify IER. When DLAB = 1, all accesses read or modify DLH. DLH can also be accessed with address offset 24h.
- IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing modifies FCR.

**Table 7. UART Registers**

Offset	Acronym	Register Description	Section
0h	RBR	Receiver Buffer Register (read only)	<a href="#">Section 3.1</a>
0h	THR	Transmitter Holding Register (write only)	<a href="#">Section 3.2</a>
4h	IER	Interrupt Enable Register	<a href="#">Section 3.3</a>
8h	IIR	Interrupt Identification Register (read only)	<a href="#">Section 3.4</a>
8h	FCR	FIFO Control Register (write only)	<a href="#">Section 3.5</a>
Ch	LCR	Line Control Register	<a href="#">Section 3.6</a>
10h	MCR	Modem Control Register	<a href="#">Section 3.7</a>
14h	LSR	Line Status Register	<a href="#">Section 3.8</a>
20h	DLL	Divisor LSB Latch	<a href="#">Section 3.9</a>
24h	DLH	Divisor MSB Latch	<a href="#">Section 3.9</a>
28h	PID1	Peripheral Identification Register 1	<a href="#">Section 3.10</a>
2Ch	PID2	Peripheral Identification Register 2	<a href="#">Section 3.10</a>
30h	PWREMU_MGMT	Power and Emulation Management Register	<a href="#">Section 3.11</a>

### 3.1 Receiver Buffer Register (RBR)

The receiver buffer register (RBR) is shown in [Figure 9](#) and described in [Table 8](#).

The UART receiver section consists of a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16x receiver clock. Receiver section control is a function of the line control register (LCR).

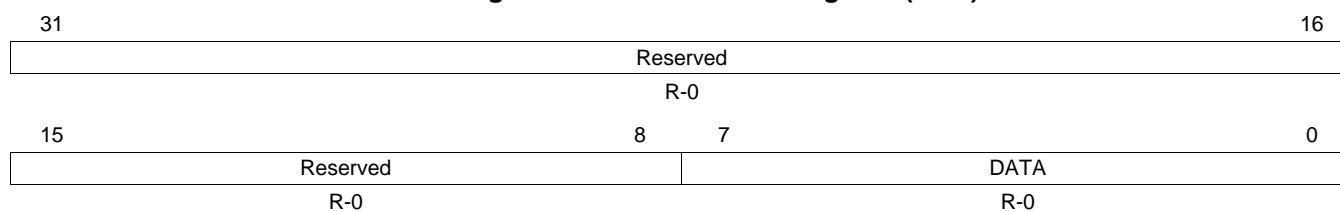
RSR receives serial data from the RX pin. Then RSR concatenates the data and moves it into RBR (or the receiver FIFO). In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled ( $DR = 1$  in IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

#### Access considerations:

RBR, THR, and DLL share one address. To read RBR, write 0 to the DLAB bit in LCR, and read from the shared address. When  $DLAB = 0$ , writing to the shared address modifies THR. When  $DLAB = 1$ , all accesses at the shared address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep  $DLAB = 0$ , so that RBR and THR are always selected at the shared address.

**Figure 9. Receiver Buffer Register (RBR)**



LEGEND: R = Read only; -n = value after reset

**Table 8. Receiver Buffer Register (RBR) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Received data

### 3.2 Transmitter Holding Register (THR)

The transmitter holding register (THR) is shown in [Figure 10](#) and described in [Table 9](#).

The UART transmitter section consists of a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the line control register (LCR).

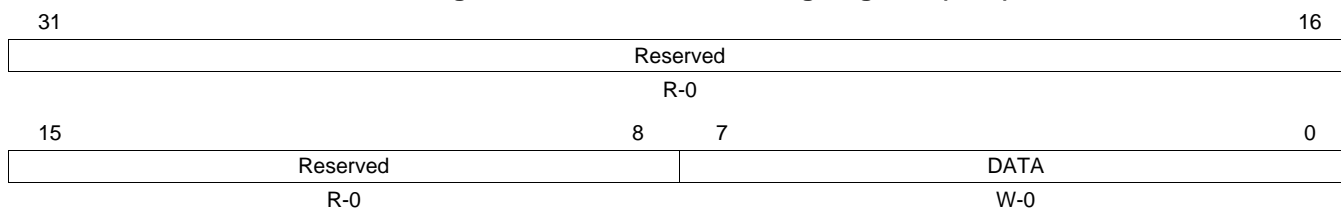
THR receives data from the internal data bus and when TSR is idle, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the TX pin. In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled (ETBEI = 1 in IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO.

#### Access considerations:

RBR, THR, and DLL share one address. To load THR, write 0 to the DLAB bit of LCR, and write to the shared address. When DLAB = 0, reading from the shared address gives the content of RBR. When DLAB = 1, all accesses at the address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that RBR and THR are always selected at the shared address.

**Figure 10. Transmitter Holding Register (THR)**



LEGEND: R = Read only; W = Write only; -n = value after reset

**Table 9. Transmitter Holding Register (THR) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Data to transmit

### 3.3 Interrupt Enable Register (IER)

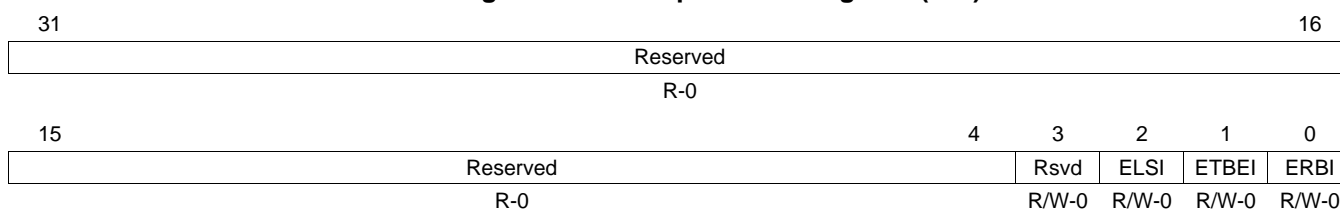
The interrupt enable register (IER) is used to individually enable or disable each type of interrupt request that can be generated by the UART. Each interrupt request that is enabled in IER is forwarded to the CPU. IER is shown in Figure 11 and described in Table 10.

#### Access considerations:

IER and DLH share one address. To read or modify IER, write 0 to the DLAB bit in LCR. When DLAB = 1, all accesses at the shared address read or modify DLH.

DLH also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that IER is always selected at the shared address.

**Figure 11. Interrupt Enable Register (IER)**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 10. Interrupt Enable Register (IER) Field Descriptions**

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	Reserved	0	Reserved. This bit must always be written with a 0.
2	ELSI	0	Receiver line status interrupt enable. Receiver line status interrupt is disabled.
		1	Receiver line status interrupt is enabled.
1	ETBEI	0	Transmitter holding register empty interrupt enable. Transmitter holding register empty interrupt is disabled.
		1	Transmitter holding register empty interrupt is enabled.
0	ERBI	0	Receiver data available interrupt and character timeout indication interrupt enable. Receiver data available interrupt and character timeout indication interrupt is disabled.
		1	Receiver data available interrupt and character timeout indication interrupt is enabled.



### 3.4 Interrupt Identification Register (IIR)

The interrupt identification register (IIR) is a read-only register at the same address as the FIFO control register (FCR), which is a write-only register. When an interrupt is generated and enabled in the interrupt enable register (IER), IIR indicates that an interrupt is pending in the IPEND bit and encodes the type of interrupt in the INTID bits. IIR is shown in Figure 12 and described in Figure 12.

The UART has an on-chip interrupt generation and prioritization capability that permits flexible communication with the CPU. The UART provides three priority levels of interrupts:

- Priority 1 - Receiver line status (highest priority)
- Priority 2 - Receiver data ready or receiver timeout
- Priority 3 - Transmitter holding register empty

The FIFOEN bit in IIR can be checked to determine whether the UART is in the FIFO mode or the non-FIFO mode.

#### Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit in LCR, reading from the address gives the content of IIR, and writing to the address modifies FCR.

**Figure 12. Interrupt Identification Register (IIR)**

31	Reserved										16
	R-0										
15	Reserved				FIFOEN	Reserved		INTID		IPEND	0
	R-0				R-0	R-0		R-0		R-1	

LEGEND: R = Read only; -n = value after reset

**Table 11. Interrupt Identification Register (IIR) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	FIFOEN	0-3h 0 1h-2h 3h	FIFOs enabled. Non-FIFO mode Reserved FIFOs are enabled. FIFOEN bit in the FIFO control register (FCR) is set to 1.
5-4	Reserved	0	Reserved
3-1	INTID	0-7h 0 1h 2h 3h 4h-5h 6h 7h	Interrupt type. See Table 12. Reserved Transmitter holding register empty (priority 3) Receiver data available (priority 2) Receiver line status (priority 1, highest) Reserved Character timeout indication (priority 2) Reserved
0	IPEND	0 1	Interrupt pending. When any UART interrupt is generated and is enabled in IER, IPEND is forced to 0. IPEND remains 0 until all pending interrupts are cleared or until a hardware reset occurs. If no interrupts are enabled, IPEND is never forced to 0. Interrupts pending. No interrupts pending.

**Table 12. Interrupt Identification and Interrupt Clearing Information**

Priority Level	IIR Bits				Interrupt Type	Interrupt Source	Event That Clears Interrupt
	3	2	1	0			
None	0	0	0	1	None	None	None
1	0	1	1	0	Receiver line status	Overrun error, parity error, framing error, or break is detected.	For an overrun error, reading the line status register (LSR) clears the interrupt. For a parity error, framing error, or break, the interrupt is cleared only after all the erroneous data have been read.
2	0	1	0	0	Receiver data-ready	Non-FIFO mode: Receiver data is ready. FIFO mode: Trigger level reached. If four character times (see <a href="#">Table 5</a> ) pass with no access of the FIFO, the interrupt is asserted again.	Non-FIFO mode: The receiver buffer register (RBR) is read. FIFO mode: The FIFO drops below the trigger level. <sup>(1)</sup>
2	1	1	0	0	Receiver time-out	FIFO mode only: No characters have been removed from or input to the receiver FIFO during the last four character times (see <a href="#">Table 5</a> ), and there is at least one character in the receiver FIFO during this time.	One of the following events: <ul style="list-style-type: none"> <li>• A character is read from the receiver FIFO.<sup>(1)</sup></li> <li>• A new character arrives in the receiver FIFO.</li> <li>• The URRST bit in the power and emulation management register (PWREMU_MGMT) is loaded with 0.</li> </ul>
3	0	0	1	0	Transmitter holding register empty	Non-FIFO mode: Transmitter holding register (THR) is empty. FIFO mode: Transmitter FIFO is empty.	A character is written to the transmitter holding register (THR).

<sup>(1)</sup> In the FIFO mode, the receiver data-ready interrupt or receiver time-out interrupt is cleared by the CPU or by the DMA controller, whichever reads from the receiver FIFO first.

### 3.5 FIFO Control Register (FCR)

The FIFO control register (FCR) is a write-only register at the same address as the interrupt identification register (IIR), which is a read-only register. Use FCR to enable and clear the FIFOs and to select the receiver FIFO trigger level FCR is shown in [Figure 13](#) and described in [Table 13](#). The FIFOEN bit must be set to 1 before other FCR bits are written to or the FCR bits are not programmed.

#### Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing to the address modifies FCR.

#### CAUTION

For proper communication between the UART and the EDMA controller, the DMAMODE1 bit must be set to 1. Always write a 1 to the DMAMODE1 bit, and after a hardware reset, change the DMAMODE1 bit from 0 to 1.

**Figure 13. FIFO Control Register (FCR)**

31	Reserved	16					
R-0							
15	Reserved	8					
R-0							
7	6	5	4	3	2	1	0
RXFIFTL	Reserved	DMAMODE1 <sup>(1)</sup>	TXCLR	RXCLR	FIFOEN		
W-0	R-0	W-0	W1C-0	W1C-0	W-0		

LEGEND: R = Read only; W = Write only; W1C = Write 1 to clear (writing 0 has no effect); -n = value after reset

<sup>(1)</sup> Always write 1 to the DMAMODE1 bit. After a hardware reset, change the DMAMODE1 bit from 0 to 1. DMAMODE1 = 1 is required for proper communication between the UART and the DMA controller.

**Table 13. FIFO Control Register (FCR) Field Descriptions**

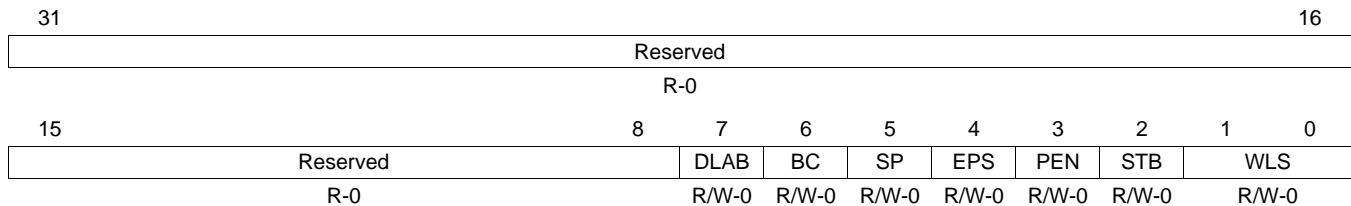
Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	RXFIFTL	0-3h	Receiver FIFO trigger level. RXFIFTL sets the trigger level for the receiver FIFO. When the trigger level is reached, a receiver data-ready interrupt is generated (if the interrupt request is enabled). Once the FIFO drops below the trigger level, the interrupt is cleared.
		0	1 byte
		1h	4 bytes
		2h	8 bytes
		3h	14 bytes
5-4	Reserved	0	Reserved
3	DMAMODE1	0	DMA MODE1 enable if FIFOs are enabled. Always write 1 to DMAMODE1. After a hardware reset, change DMAMODE1 from 0 to 1. DMAMODE1 = 1 is a requirement for proper communication between the UART and the EDMA controller.
		0	DMA MODE1 is disabled.
		1	DMA MODE1 is enabled.
2	TXCLR	0	Transmitter FIFO clear. Write a 1 to TXCLR to clear the bit.
		0	No effect.
		1	Clears transmitter FIFO and resets the transmitter FIFO counter. The shift register is not cleared.
1	RXCLR	0	Receiver FIFO clear. Write a 1 to RXCLR to clear the bit.
		0	No effect.
		1	Clears receiver FIFO and resets the receiver FIFO counter. The shift register is not cleared.
0	FIFOEN	0	Transmitter and receiver FIFOs mode enable. FIFOEN must be set before other FCR bits are written to or the FCR bits are not programmed. Clearing this bit clears the FIFO counters.
		0	Non-FIFO mode. The transmitter and receiver FIFOs are disabled, and the FIFO pointers are cleared.
		1	FIFO mode. The transmitter and receiver FIFOs are enabled.

### 3.6 Line Control Register (LCR)

The line control register (LCR) is shown in [Figure 14](#) and described in [Table 14](#).

The system programmer controls the format of the asynchronous data communication exchange by using LCR. In addition, the programmer can retrieve, inspect, and modify the content of LCR; this eliminates the need for separate storage of the line characteristics in system memory.

**Figure 14. Line Control Register (LCR)**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 14. Line Control Register (LCR) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	DLAB	0	Divisor latch access bit. The divisor latch registers (DLL and DLH) can be accessed at dedicated addresses or at addresses shared by RBR, THR, and IER. Using the shared addresses requires toggling DLAB to change which registers are selected. If you use the dedicated addresses, you can keep DLAB = 0.
		1	Allows access to the receiver buffer register (RBR), the transmitter holding register (THR), and the interrupt enable register (IER) selected. At the address shared by RBR, THR, and DLL, the CPU can read from RBR and write to THR. At the address shared by IER and DLH, the CPU can read from and write to IER.
		1	Allows access to the divisor latches of the baud generator during a read or write operation (DLL and DLH). At the address shared by RBR, THR, and DLL, the CPU can read from and write to DLL. At the address shared by IER and DLH, the CPU can read from and write to DLH.
6	BC	0	Break control.
		1	Break condition is transmitted to the receiving UART. A break condition is a condition where the UART_TX signal is forced to the spacing (cleared) state.
5	SP	0	Stick parity. The SP bit works in conjunction with the EPS and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in <a href="#">Table 15</a> .
		1	Stick parity is enabled. <ul style="list-style-type: none"> <li>• When odd parity is selected (EPS = 0), the PARITY bit is transmitted and checked as set.</li> <li>• When even parity is selected (EPS = 1), the PARITY bit is transmitted and checked as cleared.</li> </ul>
4	EPS	0	Even parity select. Selects the parity when parity is enabled (PEN = 1). The EPS bit works in conjunction with the SP and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in <a href="#">Table 15</a> .
		1	Odd parity is selected (an odd number of logic 1s is transmitted or checked in the data and PARITY bits).
		1	Even parity is selected (an even number of logic 1s is transmitted or checked in the data and PARITY bits).
3	PEN	0	Parity enable. The PEN bit works in conjunction with the SP and EPS bits. The relationship between the SP, EPS, and PEN bits is summarized in <a href="#">Table 15</a> .
		1	No PARITY bit is transmitted or checked.
		1	Parity bit is generated in transmitted data and is checked in received data between the last data word bit and the first STOP bit.

**Table 14. Line Control Register (LCR) Field Descriptions (continued)**

Bit	Field	Value	Description
2	STB	0 1	<p>Number of STOP bits generated. STB specifies 1, 1.5, or 2 STOP bits in each transmitted character. When STB = 1, the WLS bit determines the number of STOP bits. The receiver clocks only the first STOP bit, regardless of the number of STOP bits selected. The number of STOP bits generated is summarized in <a href="#">Table 16</a>.</p> <p>1 STOP bit is generated.</p> <p>WLS bit determines the number of STOP bits:</p> <ul style="list-style-type: none"> <li>When WLS = 0, 1.5 STOP bits are generated.</li> <li>When WLS = 1h, 2h, or 3h, 2 STOP bits are generated.</li> </ul>
1-0	WLS	0-3h 0 1h 2h 3h	<p>Word length select. Number of bits in each transmitted or received serial character. When STB = 1, the WLS bit determines the number of STOP bits.</p> <p>5 bits</p> <p>6 bits</p> <p>7 bits</p> <p>8 bits</p>

**Table 15. Relationship Between ST, EPS, and PEN Bits in LCR**

ST Bit	EPS Bit	PEN Bit	Parity Option
x	x	0	Parity disabled: No PARITY bit is transmitted or checked
0	0	1	Odd parity selected: Odd number of logic 1s
0	1	1	Even parity selected: Even number of logic 1s
1	0	1	Stick parity selected with PARITY bit transmitted and checked as set
1	1	1	Stick parity selected with PARITY bit transmitted and checked as cleared

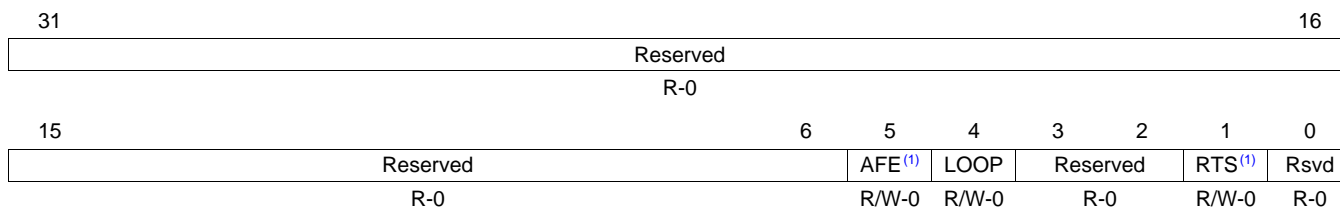
**Table 16. Number of STOP Bits Generated**

STB Bit	WLS Bits	Word Length Selected with WLS Bits	Number of STOP Bits Generated	Baud Clock (BCLK) Cycles
0	x	Any word length	1	16
1	0h	5 bits	1.5	24
1	1h	6 bits	2	32
1	2h	7 bits	2	32
1	3h	8 bits	2	32

### 3.7 Modem Control Register (MCR)

The modem control register (MCR) is shown in Figure 15 and described in Table 17. The modem control register provides the ability to enable/disable the autoflow functions, and enable/disable the loopback function for diagnostic purposes.

**Figure 15. Modem Control Register (MCR)**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

<sup>(1)</sup> All UARTs do not support this feature, see the device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.

**Table 17. Modem Control Register (MCR) Field Descriptions**

Bit	Field	Value	Description
31-6	Reserved	0	Reserved
5	AFE	0 1	Autoflow control enable. Autoflow control allows the RTS and CTS signals to provide handshaking between UARTs during data transfer. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see the device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.  0 Autoflow control is disabled. 1 Autoflow control is enabled: <ul style="list-style-type: none"> <li>• When RTS = 0, CTS is only enabled.</li> <li>• When RTS = 1, RTS and CTS are enabled.</li> </ul>
4	LOOP	0 1	Loop back mode enable. LOOP is used for the diagnostic testing using the loop back feature.  0 Loop back mode is disabled. 1 Loop back mode is enabled. When LOOP is set, the following occur: <ul style="list-style-type: none"> <li>• The UART_TX signal is set high.</li> <li>• The UART_RX pin is disconnected</li> <li>• The output of the transmitter shift register (TSR) is lopped back in to the receiver shift register (RSR) input.</li> </ul>
3-2	Reserved	0	Reserved
1	RTS	0 1	RTS control. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see the device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.  0 RTS is disabled, CTS is only enabled. 1 RTS and CTS are enabled.
0	Reserved	0	Reserved

### 3.8 Line Status Register (LSR)

The line status register (LSR) is shown in Figure 16 and described in Table 18. LSR provides information to the CPU concerning the status of data transfers. LSR is intended for read operations only; do not write to this register. Bits 1 through 4 record the error conditions that produce a receiver line status interrupt.

**Figure 16. Line Status Register (LSR)**

31	Reserved										16
R-0											
15	8	7	6	5	4	3	2	1	0		
Reserved		RXFIFOE	TEMT	THRE	BI	FE	PE	OE	DR		
R-0		R-0	R-1	R-1	R-0	R-0	R-0	R-0	R-0	R-0	

LEGEND: R = Read only; -n = value after reset

**Table 18. Line Status Register (LSR) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXFIFOE		Receiver FIFO error.
		0	<b>In non-FIFO mode:</b> There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	There is a parity error, framing error, or break indicator in the receiver buffer register (RBR).
			<b>In FIFO mode:</b>
	0	There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver FIFO and there are no more errors in the receiver FIFO.	
	1	At least one parity error, framing error, or break indicator in the receiver FIFO.	
6	TEMT		Transmitter empty (TEMT) indicator.
		0	<b>In non-FIFO mode:</b> Either the transmitter holding register (THR) or the transmitter shift register (TSR) contains a data character.
		1	Both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
			<b>In FIFO mode:</b>
	0	Either the transmitter FIFO or the transmitter shift register (TSR) contains a data character.	
	1	Both the transmitter FIFO and the transmitter shift register (TSR) are empty.	
5	THRE		Transmitter holding register empty (THRE) indicator. If the THRE bit is set and the corresponding interrupt enable bit is set (ETBEI = 1 in IER), an interrupt request is generated.
		0	<b>In non-FIFO mode:</b> Transmitter holding register (THR) is not empty. THR has been loaded by the CPU.
		1	Transmitter holding register (THR) is empty (ready to accept a new character). The content of THR has been transferred to the transmitter shift register (TSR).
			<b>In FIFO mode:</b>
	0	Transmitter FIFO is not empty. At least one character has been written to the transmitter FIFO. You can write to the transmitter FIFO if it is not full.	
	1	Transmitter FIFO is empty. The last character in the FIFO has been transferred to the transmitter shift register (TSR).	

**Table 18. Line Status Register (LSR) Field Descriptions (continued)**

Bit	Field	Value	Description
4	BI		Break indicator. The BI bit is set whenever the receive data input (RX) was held low for longer than a full-word transmission time. A full-word transmission time is defined as the total time to transmit the START, data, PARITY, and STOP bits. If the BI bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
		0	<b>In non-FIFO mode:</b> No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	A break has been detected with the character in the receiver buffer register (RBR).
		0	<b>In FIFO mode:</b> No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver FIFO and the next character to be read from the FIFO has no break indicator.
		1	A break has been detected with the character at the top of the receiver FIFO.
3	FE		Framing error (FE) indicator. A framing error occurs when the received character does not have a valid STOP bit. In response to a framing error, the UART sets the FE bit and waits until the signal on the RX pin goes high. Once the RX signal goes high, the receiver is ready to detect a new START bit and receive new data. If the FE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
		0	<b>In non-FIFO mode:</b> No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A framing error has been detected with the character in the receiver buffer register (RBR).
		0	<b>In FIFO mode:</b> No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no framing error.
		1	A framing error has been detected with the character at the top of the receiver FIFO.
2	PE		Parity error (PE) indicator. A parity error occurs when the parity of the received character does not match the parity selected with the EPS bit in the line control register (LCR). If the PE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
		0	<b>In non-FIFO mode:</b> No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A parity error has been detected with the character in the receiver buffer register (RBR).
		0	<b>In FIFO mode:</b> No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no parity error.
		1	A parity error has been detected with the character at the top of the receiver FIFO.
1	OE		Overrun error (OE) indicator. An overrun error in the non-FIFO mode is different from an overrun error in the FIFO mode. If the OE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
		0	<b>In non-FIFO mode:</b> No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. Before the character in the receiver buffer register (RBR) could be read, it was overwritten by the next character arriving in RBR.
		0	<b>In FIFO mode:</b> No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. If data continues to fill the FIFO beyond the trigger level, an overrun error occurs only after the FIFO is full and the next character has been completely received in the shift register. An overrun error is indicated to the CPU as soon as it happens. The new character overwrites the character in the shift register, but it is not transferred to the FIFO.



**Table 18. Line Status Register (LSR) Field Descriptions (continued)**

Bit	Field	Value	Description
0	DR		Data-ready (DR) indicator for the receiver. If the DR bit is set and the corresponding interrupt enable bit is set (ERBI = 1 in IER), an interrupt request is generated.
			<b>In non-FIFO mode:</b>
		0	Data is not ready, or the DR bit was cleared because the character was read from the receiver buffer register (RBR).
		1	Data is ready. A complete incoming character has been received and transferred into the receiver buffer register (RBR).
			<b>In FIFO mode:</b>
		0	Data is not ready, or the DR bit was cleared because all of the characters in the receiver FIFO have been read.
		1	Data is ready. There is at least one unread character in the receiver FIFO. If the FIFO is empty, the DR bit is set as soon as a complete incoming character has been received and transferred into the FIFO. The DR bit remains set until the FIFO is empty again.

### 3.9 Divisor Latches (DLL and DLH)

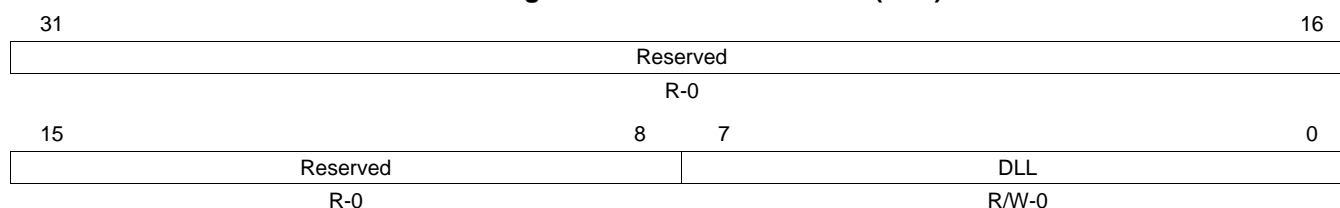
Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. The latches are in DLH and DLL. DLH holds the most-significant bits of the divisor, and DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

#### Access considerations:

- RBR, THR, and DLL share one address. When DLAB = 1 in LCR, all accesses at the shared address are accesses to DLL. When DLAB = 0, reading from the shared address gives the content of RBR, and writing to the shared address modifies THR.
- IER and DLH share one address. When DLAB = 1 in LCR, accesses to the shared address read or modify to DLH. When DLAB = 0, all accesses at the shared address read or modify IER.

DLL and DLH also have dedicated addresses. If you use the dedicated addresses, you can keep the DLAB bit cleared, so that RBR, THR, and IER are always selected at the shared addresses.

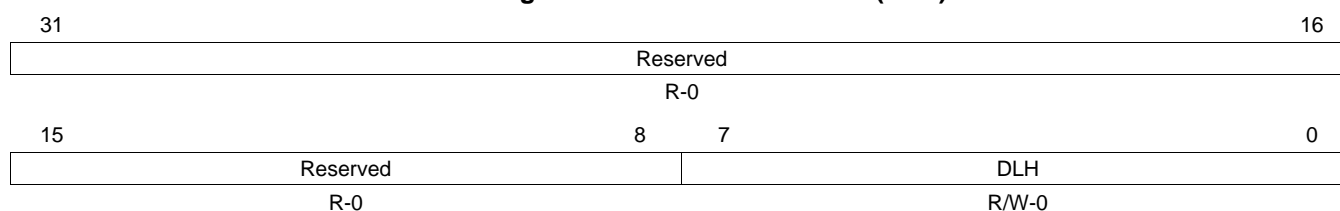
The divisor LSB latch (DLL) is shown in [Figure 17](#) and described in [Table 19](#). The divisor MSB latch (DLH) is shown in [Figure 18](#) and described in [Table 20](#).

**Figure 17. Divisor LSB Latch (DLL)**


LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 19. Divisor LSB Latch (DLL) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLL	0-Fh	The 8 least-significant bits (LSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator. Maximum baud rate is 128 kbps.

**Figure 18. Divisor MSB Latch (DLH)**


LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

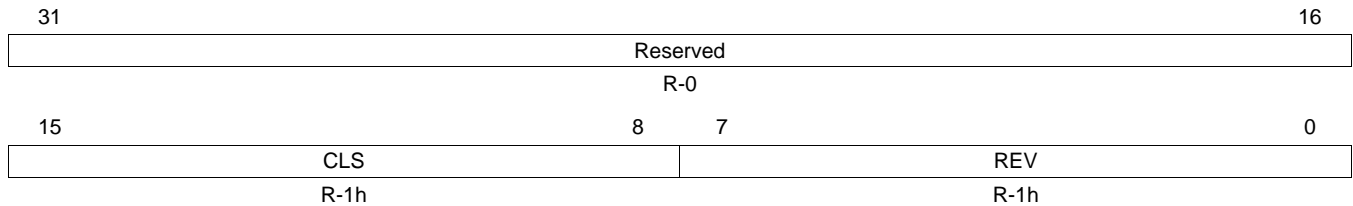
**Table 20. Divisor MSB Latch (DLH) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLH	0-Fh	The 8 most-significant bits (MSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator. Maximum baud rate is 128 kbps.

### 3.10 Peripheral Identification Registers (PID1 and PID2)

The peripheral identification registers (PID) contain identification data (class, revision, and type) for the peripheral. PID1 is shown in [Figure 19](#) and described in [Table 21](#). PID2 is shown in [Figure 20](#) and described in [Table 22](#).

**Figure 19. Peripheral Identification Register 1 (PID1)**

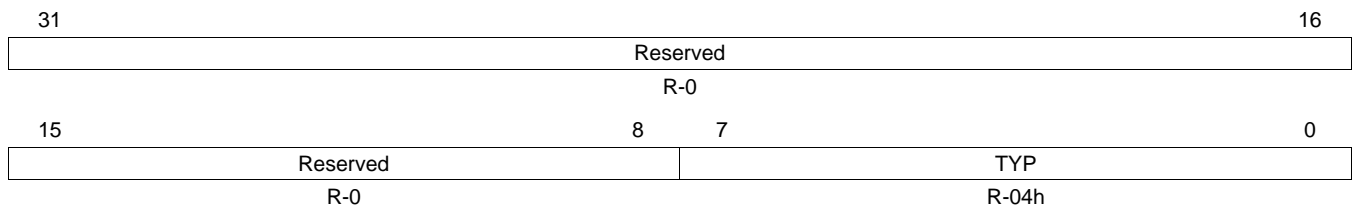


LEGEND: R = Read only; -n = value after reset

**Table 21. Peripheral Identification Register 1 (PID1) Field Descriptions**

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15-8	CLS	1	Identifies class of peripheral. Serial port
7-0	REV	1	Identifies revision of peripheral. Current revision of peripheral.

**Figure 20. Peripheral Identification Register 2 (PID2)**



LEGEND: R = Read only; -n = value after reset

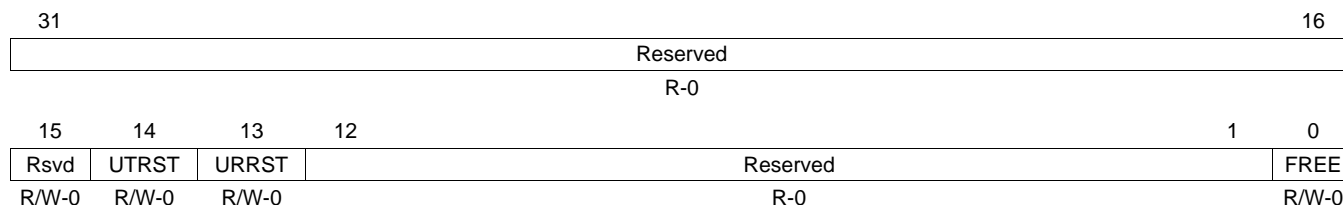
**Table 22. Peripheral Identification Register 2 (PID2) Field Descriptions**

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	TYP	4h	Identifies type of peripheral. UART

### 3.11 Power and Emulation Management Register (PWREMU\_MGMT)

The power and emulation management register (PWREMU\_MGMT) is shown in [Figure 21](#) and described in [Table 23](#).

**Figure 21. Power and Emulation Management Register (PWREMU\_MGMT)**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 23. Power and Emulation Management Register (PWREMU\_MGMT) Field Descriptions**

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	Reserved	0	Reserved. This bit must always be written with a 0.
14	UTRST	0	UART transmitter reset. Resets and enables the transmitter. Transmitter is disabled and in reset state.
		1	Transmitter is enabled.
13	URRST	0	UART receiver reset. Resets and enables the receiver. Receiver is disabled and in reset state.
		1	Receiver is enabled.
12-1	Reserved	1	Reserved
0	FREE	0	Free-running enable mode bit. This bit determines the emulation mode functionality of the UART. When halted, the UART can handle register read/write requests, but does not generate any transmission/reception, interrupts or events. If a transmission is not in progress, the UART halts immediately. If a transmission is in progress, the UART halts after completion of the one-word transmission.
		1	Free-running mode is enabled; UART continues to run normally.

## Appendix A Revision History

[Table 24](#) lists the changes made since the previous version of this document.

**Table 24. Document Revision History**

<b>Reference</b>	<b>Additions/Modifications/Deletions</b>
<a href="#">Section 2.1</a>	Changed first paragraph.

## 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 TI 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. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

### Products

Amplifiers	<a href="http://amplifier.ti.com">amplifier.ti.com</a>
Data Converters	<a href="http://dataconverter.ti.com">dataconverter.ti.com</a>
DLP® Products	<a href="http://www.dlp.com">www.dlp.com</a>
DSP	<a href="http://dsp.ti.com">dsp.ti.com</a>
Clocks and Timers	<a href="http://www.ti.com/clocks">www.ti.com/clocks</a>
Interface	<a href="http://interface.ti.com">interface.ti.com</a>
Logic	<a href="http://logic.ti.com">logic.ti.com</a>
Power Mgmt	<a href="http://power.ti.com">power.ti.com</a>
Microcontrollers	<a href="http://microcontroller.ti.com">microcontroller.ti.com</a>
RFID	<a href="http://www.ti-rfid.com">www.ti-rfid.com</a>
RF/IF and ZigBee® Solutions	<a href="http://www.ti.com/lprf">www.ti.com/lprf</a>

### Applications

Audio	<a href="http://www.ti.com/audio">www.ti.com/audio</a>
Automotive	<a href="http://www.ti.com/automotive">www.ti.com/automotive</a>
Broadband	<a href="http://www.ti.com/broadband">www.ti.com/broadband</a>
Digital Control	<a href="http://www.ti.com/digitalcontrol">www.ti.com/digitalcontrol</a>
Medical	<a href="http://www.ti.com/medical">www.ti.com/medical</a>
Military	<a href="http://www.ti.com/military">www.ti.com/military</a>
Optical Networking	<a href="http://www.ti.com/opticalnetwork">www.ti.com/opticalnetwork</a>
Security	<a href="http://www.ti.com/security">www.ti.com/security</a>
Telephony	<a href="http://www.ti.com/telephony">www.ti.com/telephony</a>
Video & Imaging	<a href="http://www.ti.com/video">www.ti.com/video</a>
Wireless	<a href="http://www.ti.com/wireless">www.ti.com/wireless</a>

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2009, Texas Instruments Incorporated