TMS320DM646x DMSoC Universal Asynchronous Receiver/Transmitter (UART)

User's Guide

Literature Number: SPRUER6D
May 2009

Contents

List of Figures

List of Tables

Read This First

About This Manual

Describes the operation of the universal asynchronous receiver/transmitter (UART) in the TMS320DM646x Digital Media System-on-Chip (DMSoC).

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 TMS320DM646x Digital Media System-on-Chip (DMSoC). Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

The current documentation that describes the DM646x DMSoC, related peripherals, and other technical collateral, is available in the C6000 DSP product folder at: www.ti.com/c6000.

SPRUEP8 — ***TMS320DM646x DMSoC DSP Subsystem Reference Guide.*** Describes the digital signal processor (DSP) subsystem in the TMS320DM646x Digital Media System-on-Chip (DMSoC).

SPRUEP9 — ***TMS320DM646x DMSoC ARM Subsystem Reference Guide.*** Describes the ARM subsystem in the TMS320DM646x Digital Media System-on-Chip (DMSoC). The ARM subsystem is designed to give the ARM926EJ-S (ARM9) master control of the device. In general, the ARM is responsible for configuration and control of the device; including the DSP subsystem and a majority of the peripherals and external memories.

SPRUEQ0 — ***TMS320DM646x DMSoC Peripherals Overview Reference Guide.*** Provides an overview and briefly describes the peripherals available on the TMS320DM646x Digital Media System-on-Chip (DMSoC).

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 operation of the universal asynchronous receiver/transmitter (UART), infrared data association (IrDA), and consumer infrared (CIR) module in theTMS320DM646x Digital Media System-on-Chip (DMSoC). In this document, we refer to this module as the UART module with UART modem, IrDA, and CIR functionalities. There are three UART instances in DM646x: UART0, UART1, and UART2.

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 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.

This module is also capable of performing standard infrared communication in slow infrared mode (SIR) and medium infrared mode (MIR) defined by the Infrared Data Association (IrDA).

Moreover, this module also supports consumer infrared (CIR) communications. The CIR mode uses a variable pulse-width modulation technique to encompass the various formats of infrared encoding for remote control applications. The CIR logic is to transmit and receive data packets according to the user-definable frame structure and packet content.

1.2 Features

An overview of the features are:

UART consists of the following main features:

- Selectable UART/IrDA/CIR modes.
- Dual 64 entry FIFOs for received and transmitted data payload.
- Programmable and selectable transmit and receive FIFO trigger levels for DMA and interrupt generation.
- Frequency prescaler values from 0 to 16383 to generate the appropriate baud rates.
- Two DMA requests, 1 interrupt request to the system.

UART/Modem includes the following functions:

- Baud-rate up to 1.8432 Mbits/s.
- Software/Hardware flow control.
  - Programmable XON/XOFF characters.
  - Programmable Auto-$\overline{\text{RTS}}$ and Auto-$\overline{\text{CTS}}$.

- Programmable serial interface characteristics.
  - 5, 6, 7, or 8 bit characters
  - Even, odd, mark (always = 1), space (always = 0), or no parity (non parity bit frame) bit generation and detection.
  - 1, 1.5, or 2 stop bit generation.
- False start bit detection.
- Line break generation and detection.
- Fully prioritized interrupt system controls.
- Internal test and loopback capabilities.
- Modem control functions ($\overline{\text{CTS}}$, $\overline{\text{RTS}}$, $\overline{\text{DSR}}$, $\overline{\text{DTR}}$, $\overline{\text{RI}}$, and $\overline{\text{DCD}}$) ($\overline{\text{DCD}}$, $\overline{\text{DSR}}$, $\overline{\text{DTR}}$, and $\overline{\text{RI}}$ are only pinned out on UART0 as $\overline{\text{UDCD0}}$, $\overline{\text{UDSR0}}$, $\overline{\text{UDTR0}}$, and $\overline{\text{URIN0}}$).

IR-IrDA functions are listed below:

- Slow infrared (SIR, baud-rate up to 57.6 Kbits/s) and medium infrared (MIR, baud-rate up to 0.576Mbits/s) operations.
- Framing error, cyclic redundancy check (CRC) error, abort pattern (SIR, MIR) detection.
- 8-entry status FIFO (with selectable trigger levels) available to monitor frame length and frame errors.

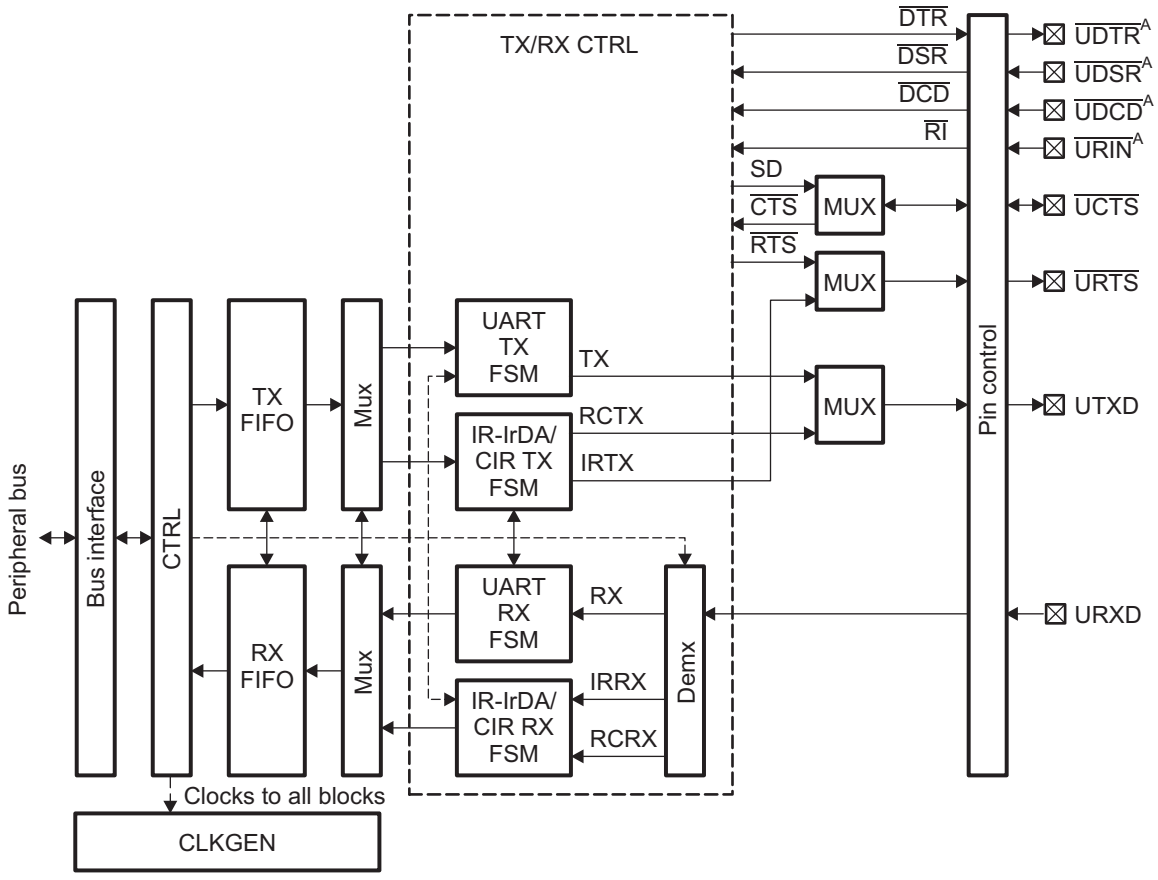
IR-CIR functions include:

- Consumer Infrared remote control mode with programmable data encoding.

1.3 Functional Block Diagram

A block diagram of the UART is shown in Figure 1.

Figure 1. UART Block Diagram



A Only supported on UART0.

1.4 Application Example

Several example applications of the UART peripheral in different operation modes are shown in Figure 2, Figure 3, and Figure 4.

Figure 2. UART Modem Application Example

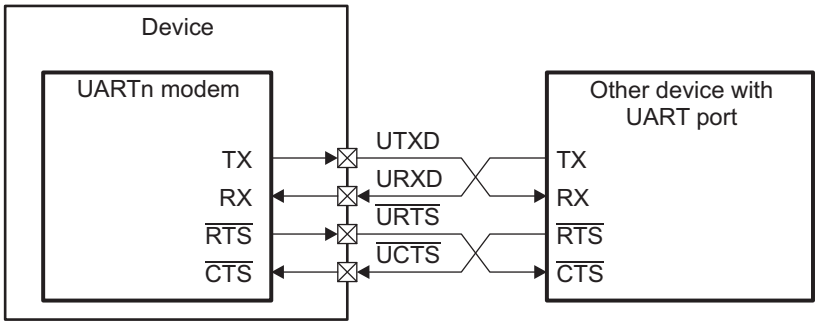


Figure 3. UART IrDA Application Example

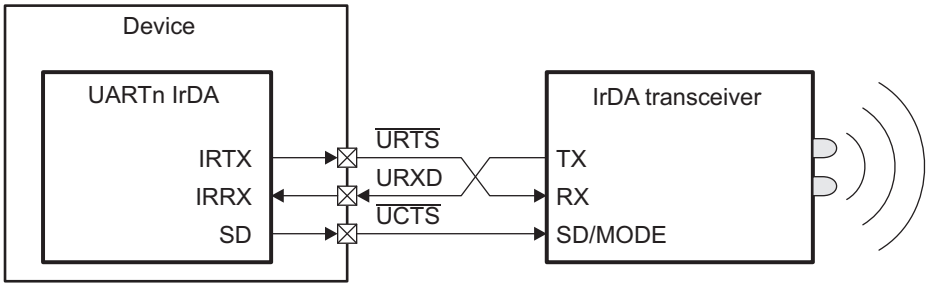
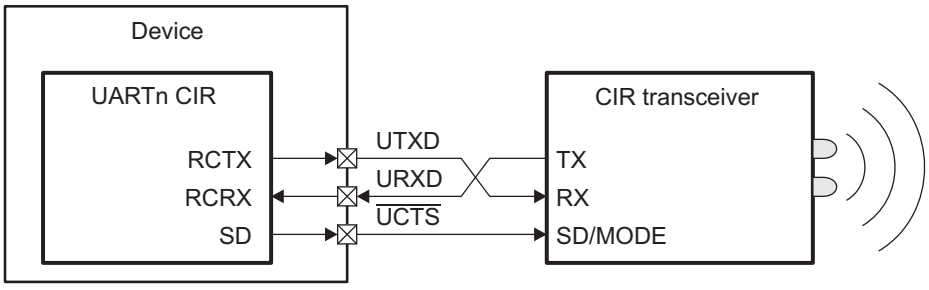


Figure 4. UART CIR Application Example



1.5 Industry Standard Compatibility Statement

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element. It also supports IrDA modes including SIR and MIR. Moreover, it supports the flexible CIR mode.

2 Architecture

This section describes the architecture of the UART.

2.1 Clock Control

The UART bit clock is derived from the 24-MHz clock, AUX_CLKIN (the AUX_CLKIN clock is provided through either an on-chip oscillator with external crystal or an external 1.8V LYCMOS-compatible clock input). Several functionalities (FIR and autobaud) require the default 48-MHz clock, and they are not supported. The UART internal system clock is derived from SYSCLK3, which is the PLL0 clock divided by 4. For detailed information on the PLLs and clock distribution on the processor, see the *TMS320DM646x DMSoC ARM Subsystem Reference Guide* (SPRUEP9).

Before using UART, X1AUX must be enabled by clearing the OSCDIS bit in the Clock and Oscillator Control register to 0 (CLKCTL[24] = 0). The CLKCTL is a System Module register at address 01C4 005Ch. For more information about CLKCTL, see the *TMS320DM646x DMSoC ARM Subsystem Reference Guide* (SPRUEP9).

2.2 Signal Descriptions

UART signals are described in Table 1.

Table 1. Signal Descriptions

Signal	Pin	Type	Description	Default
RX	URXD	I	Serial data input for all modes.	Unknown
TX	UTXD	O	Serial data output in UART modes. In other modes, this pin is set to the reset value (inactive state).	1
IRTX	$\overline{\text{URTS}}$	O	Serial data output in IrDA modes. In other modes, this pin is set to the reset value (inactive state). Pin multiplexed with $\overline{\text{RTS}}$ at the pin-out.	0
RCTX	UTXD	O	Serial data output in CIR mode. In other modes, this pin is set to the reset value (inactive state). Pin multiplexed with TX at the pin-out.	0
$\overline{\text{CTS}}$	$\overline{\text{UCTS}}$	I	Clear To Send.	Unknown
			Active low modem status signal. Reading MSR[4] checks the condition of $\overline{\text{CTS}}$. Reading MSR[0] checks a change of state of $\overline{\text{CTS}}$ since the last read of MSR. $\overline{\text{CTS}}$ is used in auto-$\overline{\text{CTS}}$ mode to control the transmitter.	
$\overline{\text{RTS}}$	$\overline{\text{URTS}}$	O	Request To Send.	1
			When active (low), the module is ready to receive data. Setting MCR[1] activates $\overline{\text{RTS}}$. It becomes inactive as a result of a module reset, loop back mode or by clearing MCR[1]. In auto-RTS mode, it becomes inactive as a result of the receiver threshold logic.	
SD	$\overline{\text{UCTS}}$	O	SD mode is used to configure the transceivers. The SD pin out is an inverted value of ACREG[6]. Pin multiplexed with $\overline{\text{CTS}}$ at the pin-out.	1
$\overline{\text{DSR}}$	$\overline{\text{UDSR0}}$ (1)	I	Data Set Ready.	Unknown
			Active-low modem status signal. Reading MSR[5] checks the condition of $\overline{\text{DSR}}$. Reading MSR[1] checks a change of state of $\overline{\text{DSR}}$ since the last read of MSR.	
$\overline{\text{DTR}}$	$\overline{\text{UDTR0}}$ (1)	O	Data Terminal Ready.	1
			When active (low), this signal informs the modem that the module is ready to communicate. It's activated by setting MCR[0].	
$\overline{\text{DCD}}$	$\overline{\text{UDCD0}}$ (1)	I	Data Carrier Detect.	Unknown
			Active-low modem status signal. The condition of $\overline{\text{DCD}}$ can be checked by reading MSR[7] and any change in its state can be detected by reading MSR[3].	
$\overline{\text{RI}}$	$\overline{\text{URIN0}}$ (1)	I	Ring Indicator.	Unknown
			Active-low modem status signal. The condition of $\overline{\text{RI}}$ can be checked by reading MSR[6] and any change in its state can be detected by reading MSR[2].	

(1) Only supported on UART0.

2.3 Pin Multiplexing

On the DM646x DMSoC, 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 Modes of Operation

The UART/IrDA module can operate in five different modes:

1. UART 16× mode (≤ 115.2 Kbits/s)
2. UART 13× mode (≥ 230.4 Kbits/s)
3. IrDA SIR mode (≤ 57.6 Kbits/s)
4. IrDA MIR mode (0.576 Mbits/s)
5. CIR mode (programmable modulation rates specific to remote control applications)

The module performs serial-to-parallel conversion on received data characters and parallel-to-serial conversion on transmitted data characters by the processor. The complete status of each channel of the module and each received character/frame can be read at any time during functional operation via the line status register (LSR).

The module can be placed in an alternate mode (FIFO mode) relieving the processor of excessive software overhead by buffering the received/transmitted characters. Both receiver and transmitter FIFOs can store up to 64 bytes of data (plus three additional bits of error status per byte for the receiver FIFO) and have selectable trigger levels.

Both interrupts and DMA are available to control the data-flow between the system (LH) and the module.

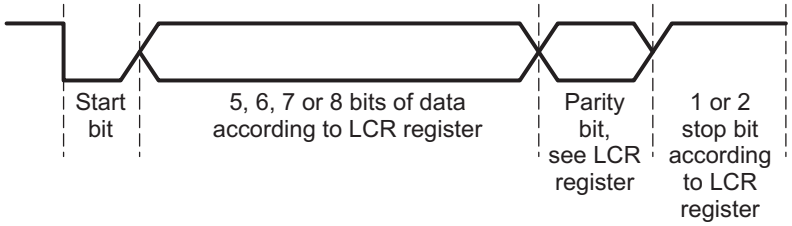
2.4.1 UART Modes

The UART uses a wired interface for serial communication with a remote device.

The UART module is functionally compatible with the TL16C750 UART and is also functionally compatible to earlier designs such as the TL16C550.

This module can use hardware or software flow control to manage transmission/reception. Hardware flow control significantly reduces software overhead and increases system efficiency by automatically controlling serial data flow using the $\overline{RTS}$ output and $\overline{CTS}$ input signals. Software flow control automatically controls data flow by using programmable XON/XOFF characters.

Figure 5. UART Data Format



2.4.2 SIR Mode

In slow infrared (SIR) mode, data transfer takes place between the LH and the peripheral devices at speeds of up to 57600 bauds. A SIR transmit frame starts with start flags (either a single C0h, multiple C0h, or a single C0h preceded by a number of FFh flags), followed by frame data, CRC-16 and ends with a stop flag (C1h). The bit format for a single word uses a single start bit, eight data bits and one stop bit and is unaffected by the use and settings of the LCR register.

When multiple start flags are required, XBOF_TYPE in the BLR register is used to select whether C0h or FFh start patterns are used.

The SIR transmit state machine attaches start flags, CRC-16, and stop flags. It checks the outgoing data to establish if data transparency is required.

SIR transparency is carried out if the outgoing data, between the start and stop flags, contains C0h, C1h or 7Dh. If one of these is about to be transmitted, then the SIR state machine sends an escape character (7Dh) first, then inverts the fifth bit of the real data to be sent, and sends this data immediately after the 7Dh character.

The SIR receive state machine recovers the receive clock, removes the start flags, removes any transparency from the incoming data, and determines frame boundary with reception of the stop flag. It also checks for errors such as: frame abort (7Dh character followed immediately by a C1h stop flag, without transparency), CRC error, and frame-length error. At the end of a frame reception, the LH reads the line status register (LSR) to find out possible errors of the received frame.

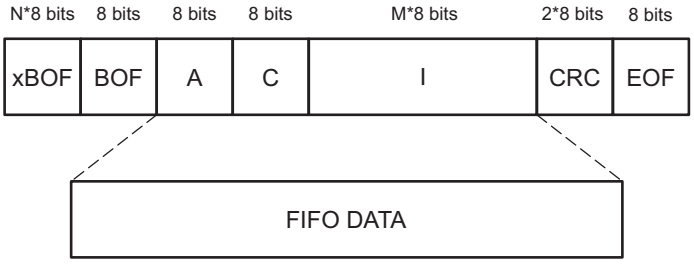
Data can be transferred both ways by the module but when the device is transmitting, the IRRX circuitry is automatically disabled by hardware. Refer to the DIS_IR_RX in the ACREG register for a description of the logical operation. This applies to both SIR and MIR.

The infrared output in SIR mode can either be 1.6µs or 3/16 encoding, selected by PULSE_TYPE in the ACREG register. In 1.6µs encoding, the infrared pulse width is 1.6µs and in 3/16 encoding the infrared pulse width is 3/16 of a bit duration (1/baud-rate).

The transmitting device must send at least two start flags at the start of each frame for back-to-back frames. Reception does support variable-length stop bits.

2.4.2.1 Frame Format

Figure 6. IrDA SIR Frame Format



The CRC is applied on the address (A), control (C) and information (I) bytes.

The two words of CRC are written in the FIFO in reception.

2.4.2.2 Asynchronous Transparency

Before transmitting a byte, the UART IrDA controller examines each byte of the payload and the CRC field (between BOF and EOF). The following occurs with each byte equal to C0h (BOF), C1h (EOF), or 7Dh (control escape).

In transmission:

1. Insert a control escape (CE) byte preceding the byte.
2. Complements bit 5 of the byte (that is, exclusive-OR's the byte with 20h).

The byte sent for the CRC computation is the initial byte written in the TX FIFO (before the XOR with 20h).

In reception, for the A, C, I, CRC field:

1. Compare the byte with CE byte, if not equal send it to the CRC detector and store it in the RX FIFO.
2. If equal to CE, discard the CE byte.
3. Complements the bit 5 of the byte following the CE.
4. Send the complemented byte to the CRC detector and store it in the RX FIFO.

2.4.2.3 Abort Sequence

The transmitter may decide to prematurely close a frame. The transmitter aborts by sending the following sequence: 7DC1h. The abort pattern closes the frame without a CRC field or an ending flag.

It is possible to abort a transmission frame by programming the ACREG[1].

When this bit is set to 1, 7Dh and C1h are transmitted and the frame is not terminated with CRC or stop flags.

The receiver treats a frame as an aborted frame when a 70Dh character followed immediately by a C1h character have been received without transparency.

2.4.2.4 Pulse Shaping

In SIR mode both the 3/16th and the 1.6µs pulse duration methods are supported. PULSE_TYPE in the ACREG register selects the pulse width method in transmit mode.

2.4.2.5 IR Address Checking

In all IR modes, if address checking has been enabled, only frames intended for the device are written to the RX FIFO. This is to avoid receiving frames not meant for this device in a multi-point infrared environment. It is possible to program two frame addresses that the UART IrDA receives with XON1/ADDR1 and XON2/ADDR2 registers.

Selecting address1 checking is done by setting EFR[0] to 1. And address2 checking is done by setting EFR[1] to 1. Setting EFR[1:0] to 0 disables all address checking operations. If both bits are set, then the incoming frame is checked for both private and public addresses.

If address checking is disabled, then all received frames are written into the reception FIFO.

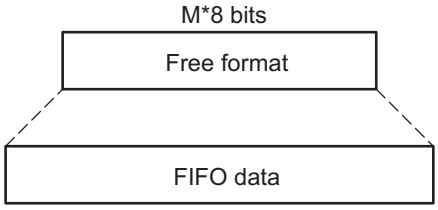
2.4.2.6 SIR Free Format Mode

To allow complete software flexibility in the transmission and reception of Infrared data packets, the SIR Free Format mode is a sub function of the existing SIR mode such that all frames going to and from the FIFO buffers are untouched with respect to appending and removing control characters and CRC values. In transmission phase, it uses the IRTX pin as in SIR mode.

This mode corresponds to a UART mode with a pulse modulation of 3/16 of baud-rate pulse width.

For example, a normal SIR packet has BOF control and CRC error checking data appended (transmitting) or removed (receiving) from the data going to and from the FIFOs. In SIR FF (Free Format) mode only the data termed the FIFO DATA area shown in Figure 7 (and shown in Figure 6) would be transmitted and received.

Figure 7. SIR Free Format Mode



In this mode, the entire FIFO data packet is constructed (encoded and decoded) by the LH software.

The SIR Free Format mode is selected by setting the module in UART mode (MODE_SELECT in the MDR1 register = 0) and UART_PULSE in the MDR2 register to 1 to allow the pulse shaping.

Because the bit format is to remain the same, some UART mode configuration register bits need to be set to a specific value:

- LCR[1:0] = 3h (8 data bits)

- LCR[2] = 0 (1 stop bit)
- LCR[3] = 0 (no parity)
- ACREG[7] = 0 (3/16 of baud-rate pulse width)

The features defined through IRRXINVERT in the MDR2 register and DIS_IR_RX in the ACREG register are also supported.

Note: All other configuration registers need to be at the reset value.

The UART mode interrupts are used for the SIR FF mode.

2.4.3 MIR Mode

In medium infrared (MIR) mode, data transfer takes place between LH and peripheral devices at 0.576 Mbits/s speed. A MIR transmit frame starts with start flags (at least two), followed by frame data, then CRC-16, and ends with a stop flag. See Figure 8.

Figure 8. MIR Transmit Frame Format

On transmit, the MIR state machine attaches start flags, CRC-16, and stop flags. It also looks for 5 consecutive 1s in the frame data and automatically inserts 0 after 5 consecutive 1s (this is called bit stuffing).

On receive, the MIR receive state machine recovers the receive clock, removes the start flags, de-stuffs the incoming data, and determines frame boundary with reception of the stop flag. It also checks for errors such as: frame abort, CRC error, or frame-length error. At the end of a frame reception, the LH reads the line status register (LSR) to find out possible errors of received frame.

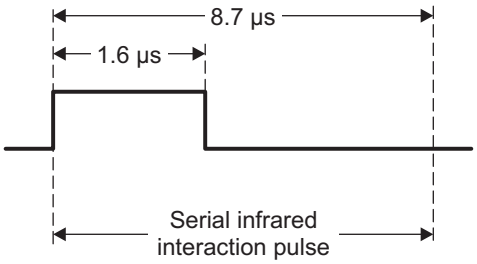
Data can be transferred both ways by the module but when the device is transmitting, the IR RX circuitry is automatically disabled by hardware. Refer to DIS_IR_RX in the ACREG register for a description of the logical operation. This applies to both SIR and MIR.

2.4.3.1 Serial Infrared Interaction Pulse Generation (SIP)

In MIR mode, the transmitter needs to send a serial infrared interaction pulse (SIP) at least once every 500 ms. The purpose of the SIP is to let slow devices (operating in SIR mode) know that the medium is currently occupied.

The SIP pulse is shown in Figure 9.

Figure 9. SIP Pulse



When MDR1[6] equals 1, the TX state machine will always send one SIP at the end of a transmission frame. But when MDR1[6] equals 0, the transmission of the SIP depends on ACREG[3]. The system (LH) can set ACREG[3] at least once every 500ms. The advantage of this approach over the default approach is that the TX state machine does not need to send the SIP at the end of each frame which may reduce the overhead required.

2.4.4 CIR Mode

In consumer infrared mode, the infrared operation is designed to function as a programmable (universal) remote control. By setting the MDR1 register, the UART can be set to CIR mode in the same way as the other IrDA modes are set using the MDR1 register.

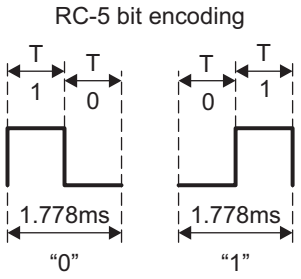
The CIR mode uses a variable pulse width modulation technique (based on multiples of a programmable T period) to encompass the various formats of infrared encoding for remote control applications. The CIR logic is designed to transmit and receive data packets according to the user definable frame structure and packet content.

2.4.4.1 Consumer IR Encoding

There are two distinct methods of encoding for remote control applications. The first uses time extended bit forms, that is, a variable pulse distance (or duration) whereby the difference between a logic 1 and logic 0 is the length of the pulse width; and the second is the use of a bi-phase where the encoding of the logic 0 and logic 1 is in the change of signal level from 1 ≥ 0 or 0 ≥ 1, respectively.

The CIR mode is designed to use a completely flexible free format encoding where a digit 1 from the TX/RX FIFO is transmitted/received as a modulated pulse with duration T. Equally, 0 is transmitted/received as a blank duration T. The protocol of the data is constructed and deciphered by the host CPU. For example, the RC-5 protocol using Manchester encoding can be emulated as using a 01 pair for 1 and a 10 pair for 0. RC-5 bit encoding is shown in Figure 10.

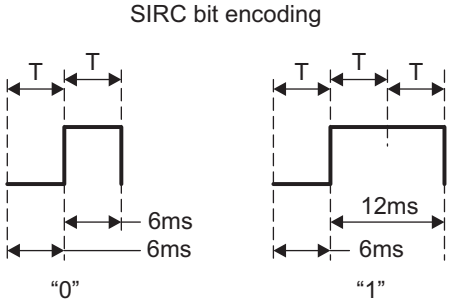
Figure 10. RC-5 Bit Encoding



Since the CIR mode logic does not impose a fixed format for infrared packets of data, the CPU software is at liberty to define the format through the use of simple data structures that will then be modulated into an industry standard such as RC-5, SIRC, and so on. To send a sequence of 0101 in RC-5, the host software must write an eight bit binary character of 1001 1001 to the data TX FIFO of the UART.

For SIRC, the modulation length (multiples of T) is the method used to distinguish between a 1 or a 0. The following SIRC digits show the difference in encoding between this and RC-5 (see Figure 11). The pulse width is extended for 1 digits.

Figure 11. SIRC Bit Encoding



To construct comprehensive packets that constitute remote control commands, the host software must combine a number of eight bit data characters in a sequence that follows one of the universally accepted formats. For illustrative purposes, a standard RC-5 frame is shown in Figure 12 (the SIRC format follows this). Each of the above fields in RC-5 can be considered as two T pulses (digital bits) from the TX and RX FIFOs.

The standard RC-5 format as seen by the UART_IrDA in CIR mode.

Figure 12. RC-5 Standard Packet Format

S1	S2	T	A4	A3	A2	A1	A0	C5	C4	C3	C2	C1	C0

Where
S1, S2: Start bits (always 1)
T: Toggle Bit
A4..A0: Address (or system) bits
C5..C0: Command bits

The toggle bit T changes each time a new command is transmitted to allow detection of pressing the same key twice (or to detect receiving the same data from the host consecutively). Since a code is being sent as long as the CPU transmits characters to the UART for transmission, a brief delay in the transmission of the same command would be detected by the use of the toggle bit. The address bits define the machine or device that the Infrared transmission is intended for and the command defines the operation.

To accommodate an extended RC-5 format, the S2 bit is replaced by a further command bit (C6) that allows the command range to increase to 7-bits. This format is known as the extended RC-5 format.

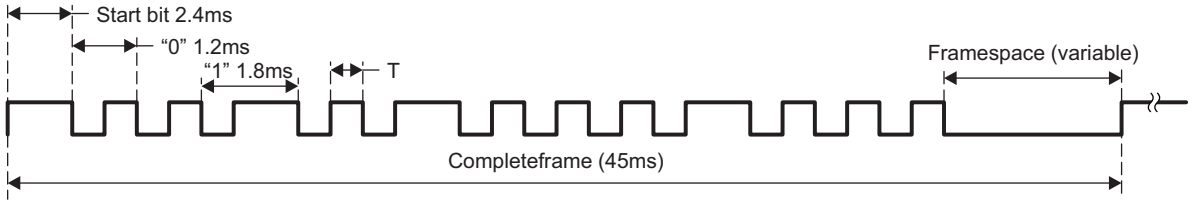
The SIRC encoding uses the duration of modulation for mark and space; hence the duration of data bits inside the standard frame length will vary depending upon the logic 1 content. The packet format is shown in Figure 13 and bit encoding is illustrated in Figure 14. There is one start bit of two milliseconds and control codes followed by data that constitute the whole frame.

Figure 13. SIRC Packet Format

S	C0	C1	C2	C3	C4	C5	C6	D0	D1	D2	D3	D4

It should be noted that the encoding must take a standard duration but the contents of the data may vary. This implies that the control software for emitting and receiving data packets must exercise a scheme of inter-packet delay, where the emission of successive packets can only be done after a real time delay has expired.

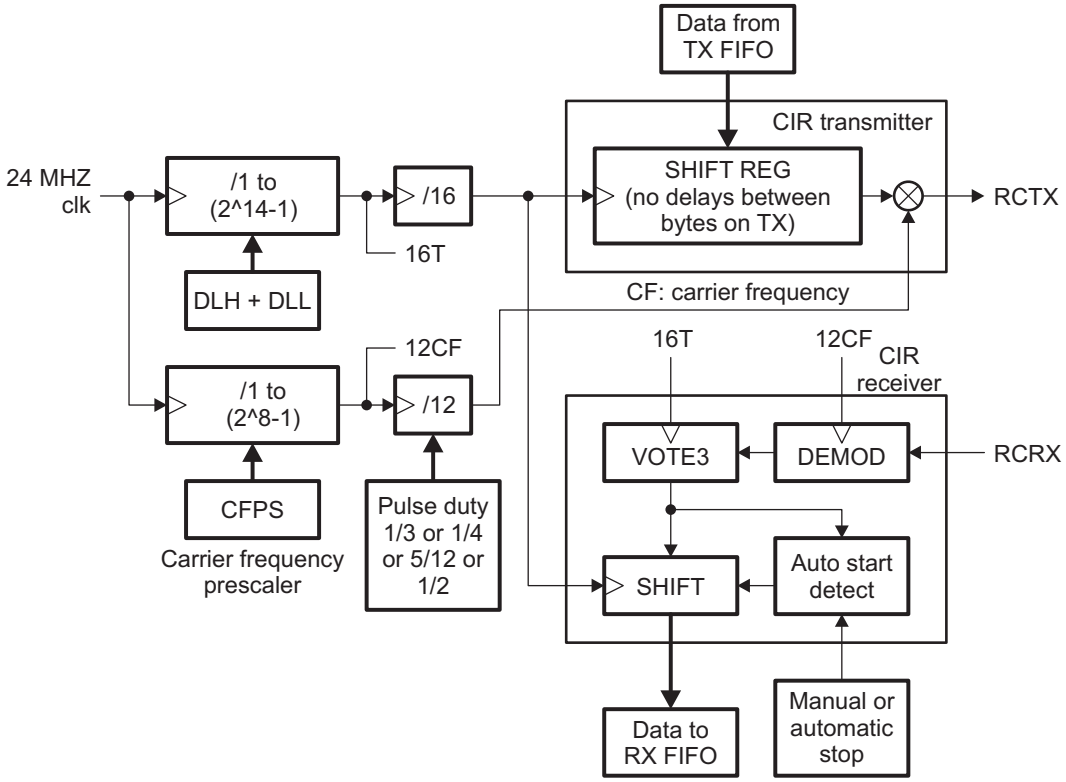
Figure 14. SIRC BIT Transmission Example



2.4.4.2 CIR Mode Operation

As explain below, depending on the encoding method (variable pulse distance/bi-phase), the LH should develop a data structure that combines the 1 and 0 with a T period in order to encode the complete frame to transmit. This can then be transmitted to the infrared output with a method of modulation shown in Figure 15.

Figure 15. CIR Mode Block Components



In transmission, the LH software must exercise an element of real time control for transmitting data packets; they must each be emitted at a constant delay from the start bits of each of the individual packets which means when sending a series of packets, the packet to packet delay must respect a specific delay. To control this delay, two methods can be used:

- By filling the TX FIFO with a number of zero bits that are transmitted with a T period,
- By using an external system timer that controls the delay either between each start of a frame or between the end of a frame and the start of the next one. This can be performed by the following methods:
  - controlling the start of the frame through the configuration register SCT bit in MDR1 and SCTX_EN bit in ACREG depending on the timer status.
  - using the TX_STATUS interrupt (TX_STATUS_IT bit in IIR register) to pre-load the next frame in the TX FIFO and to control the start of the timer (in case of control delay between the end of one frame and the start of the next frame).

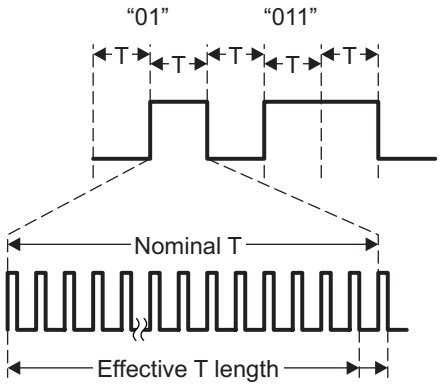
In reception, there are two ways to stop it :

- The LH can disable the reception by setting the DIS_IR_RX in the ACREG register to 1 when it considers that the reception is finished because a large number of 0 has been received. To receive a new frame, the DIS_IR_RX in the ACREG register must be cleared to 0.
- A specific mechanism, depending on the value set in the BOF length register (EBLR), allows stopping automatically the reception. If the value set in the EBLR register is different than 0, this feature is enabled and will count the number of bits received at 0. When the counter achieved the value defined in the EBLR register, the reception is automatically stopped and RX_STOP_IT (IIR[2]) is set. When a 1 is detected on the RCRX pin, the reception is automatically enabled.

2.4.4.3 Carrier Modulation

Looking more closely at the actual modulation pulses of the infrared data stream, it should be noted that each modulated pulse that constitutes a digit is a series of on/off pulses. CIR pulse modulation is shown in Figure 16.

Figure 16. CIR Pulse Modulation



A minimum of 4 modulation pulses per bit is required by the module.

Based on the requested modulation frequency, the CFPS register must be set with the correct dividing value to provide the more accurate pulse frequency:

Dividing value = (FCLK/12)/MODfreq.

Where FCLK = System bit clock frequency (24 MHz).
12 = real value of BAUD multiple.
MODfreq = Effective frequency of the modulation (MHz).

For a targeted modulation frequency of 18 KHz, the CFPS value must be set to 111 (decimal) that provides a modulation frequency of 18.02 KHz.

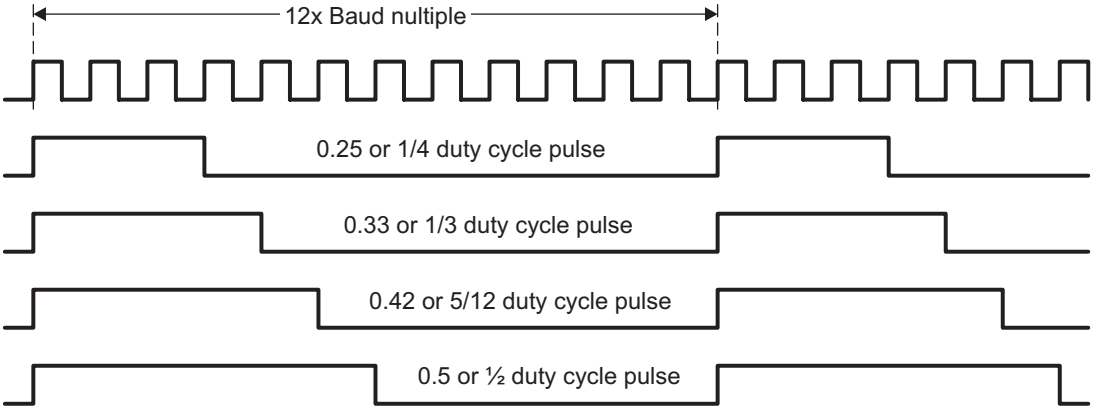
The duty cycle of these pulses is user-defined by the pulse duty register bits in the MDR2 configuration register, see Table 2 and Figure 17.

The transmission logic ensures that all pulses are transmitted completely. There is no cut off of any pulses during transmission. Furthermore, while transmitting continuous bytes back to back, no delay is inserted between 2 transmitted bytes.

Table 2. CIR Modulation Duty Cycle

MDR2[5:4]	Duty cycle (high level)
0	1/4
1	1/3
2	5/12
3	1/2

Figure 17. CIR Modulation Duty Cycle



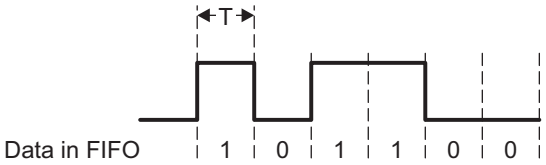
2.4.4.4 Frequency Divider Values

As previously explained, the data transferred is a succession of pulse with a T period.

Depending on the standards used, the T period is defined through the DLL and DLH registers that defined the value to divide the functional clock (24 MHz):

Dividing value = (FCLK/16)/Tfreq.
Where FCLK = System bit clock frequency (24 MHz).
16 = real value of BAUD multiple.
Tfreq = Effective frequency of the T pulse (MHz).

Figure 18. Variable Pulse Duration Definitions



For a logical 1, the pulse duration is equal to 2T and for a logical 0, it is equal to 4T. If T = 1.12 ms, the value coded into the DLH and DLL register must be 1680 in decimal.

2.4.4.5 CIR Demodulator Bypass Mode

Note: The demodulator bypass function is supported only on silicon revision 3.0 and later revisions.

In certain applications, the on-board CIR receivers have built-in demodulators. In this case, the demodulator in the CIR receiver can be optionally disabled. As a result, data coming from the RCRX pin directly enters the VOTE3 block. No other operation of the CIR module is effected when the it enters the demodulator bypass mode.

To enter the demodulator bypass mode, the following register setup is needed:

1. MDR1[2:0] = 6h
2. LCR[3] = 0h
3. LCR[4] = 1h

2.5 Functional Description

This subsection describes the functions of the peripheral.

2.5.1 Trigger Levels

The UART provides programmable trigger levels for both receiver and transmitter DMA and Interrupt generation. After reset, both transmitter and receiver FIFOs are disabled. In effect, the trigger level is the default value of one byte. The programmable trigger levels are an enhanced feature available via the trigger level register (TLR).

2.5.2 Interrupts

The UART IrDA CIR module generates interrupts. All interrupts can be enabled/disabled by writing to the appropriate bit in the interrupt enable register (IER). The interrupt status of the device can be checked at any time by reading the interrupt identification register (IIR).

The UART, IrDA, and CIR modes have different interrupts in the UART IrDA CIR module and therefore different IER and IIR mappings according to the selected mode.

2.5.2.1 UART Mode Interrupts

In UART modes, there are seven possible interrupts (Table 3). These interrupts are prioritized to six different levels.

When an interrupt is generated, the interrupt identification register (IIR) indicates that an interrupt is pending by bringing IIR[0] to 0 and provides the type of interrupt through IIR[5:1]. summarizes the interrupt control functions.

Table 3. UART Mode Interrupts

Priority Level	IIR[5-0]	Interrupt Type	Interrupt Source	Interrupt Reset Method
None	1h	None	None	None
1	6h	Receiver line status	OE, FE, PE, or BI errors occur in characters in the RX FIFO	FE, PE, BI: Read RHR. OE: Read LSR
2	Ch	RX Timeout	Stale data in RX FIFO	Read RHR
2	4h	RHR interrupt	DRDY (data ready) (FIFO disable) RX FIFO above trigger level (FIFO enable)	Read RHR until interrupt condition disappears.
3	2h	THR interrupt	TFE (THR empty) (FIFO disable) TX FIFO below trigger level (FIFO enable)	Write to THR until interrupt condition disappears.
4	0	Modem status	MSR[1:0] /= 0	Read MSR
5	10h	XOFF interrupt/special character interrupt	Receive XOFF characters(s)/special character	Receive XON character(s), if XOFF interrupt / Read of IIR, if special character interrupt
6	20h	$\overline{\text{CTS}}$, $\overline{\text{RTS}}$, $\overline{\text{DSR}}$	$\overline{\text{RTS}}$ pin, $\overline{\text{CTS}}$ pin or $\overline{\text{DSR}}$ pin change state from active (low) to inactive (high).	Read IIR

It is important to note that for the receiver line status interrupt, LSR[7] generates the interrupt.

For the XOFF interrupt, if a XOFF flow character detection caused the interrupt, the interrupt is cleared by a XON flow character detection. If special character detection caused the interrupt, the interrupt is cleared by a read of the IIR.

2.5.2.2 IrDA Mode Interrupts

In the IrDA modes there are eight possible interrupts (Table 4). The interrupt line is activated when any of the eight interrupts is generated (there is no priority). For TX_STATUS_IT bit in IIR register, interrupt source 1 is used with interrupt reset method 1 and interrupt source 2 is used with interrupt reset method 2.

Table 4. IrDA Mode Interrupts

IIR Bit	Interrupt Type	Interrupt Source	Interrupt Reset Method
0	RHR interrupt	DRDY (data ready) (FIFO disable RX FIFO above trigger level (FIFO enable).	Read RHR until interrupt condition disappears.
1	THR interrupt	TFE (THR empty) (FIFO disable) TX FIFO below trigger level (FIFO enable).	Write to THR until interrupt condition disappears.
2	Last byte in RX FIFO	Last byte of frame in RX FIFO is available to be read at the RHR port.	Read RHR.
3	RX overrun	Write to RHR when RX FIFO full.	Read RESUME register.
4	Status FIFO interrupt	Status FIFO triggers level reached.	Read STATUS1 FIFO.
5	TX Status	1.THR empty before EOF sent. Last bit of transmission of the IrDA frame has occurred but with an underrun error. OR	1.Read RESUME register. OR
		2.Transmission of the last bit of the IrDA frame is finished successfully.	2.Read IIR
6	Receiver line status interrupt	CRC, ABORT or Frame-length error is written into STATUS FIFO.	Read STATUS FIFO. [read until empty - max 8 reads required]
7	Received EOF	Received end-of -frame.	Read IIR.

Note: The meaning of TX Status is indicated by IRTX_UNDERRUN bit in MDR2.

2.5.2.3 CIR Mode interrupts

The CIR mode uses a subset of the existing IrDA mode interrupts. The following table illustrates the interrupt modes that are to be maintained. In CIR mode, TX_STATUS_IT bit in IIR-CIR mode register has a sole purpose of indicating that the last bit of infrared data has been passed to the IR TX pin.

Table 5. CIR Mode Interrupts

IIR Bit	Interrupt Type	Interrupt Source	Interrupt Reset Method
0	RHR interrupt	DRDY (data ready) (FIFO disable) RX FIFO above trigger level (FIFO enable)	Read RHR until interrupt condition disappears.
1	THR interrupt	TFE (THR empty) (FIFO disable) TX FIFO below trigger level (FIFO enable)	Write to THR until interrupt condition disappears.
2	RX_STOP_IT	Receive stop interrupt (depending on value set in the BOF Length register (EBLR))	Read IIR
3	RX overrun	Write to RHR when RX FIFO full.	Read RESUME register.
4	N/A for CIR	N/A for CIR	N/A for CIR
5	TX Status	Transmission of the last bit of the frame is finished successfully.	Read IIR
6	N/A for CIR	N/A for CIR	N/A for CIR
7	N/A for CIR	N/A for CIR	N/A for CIR

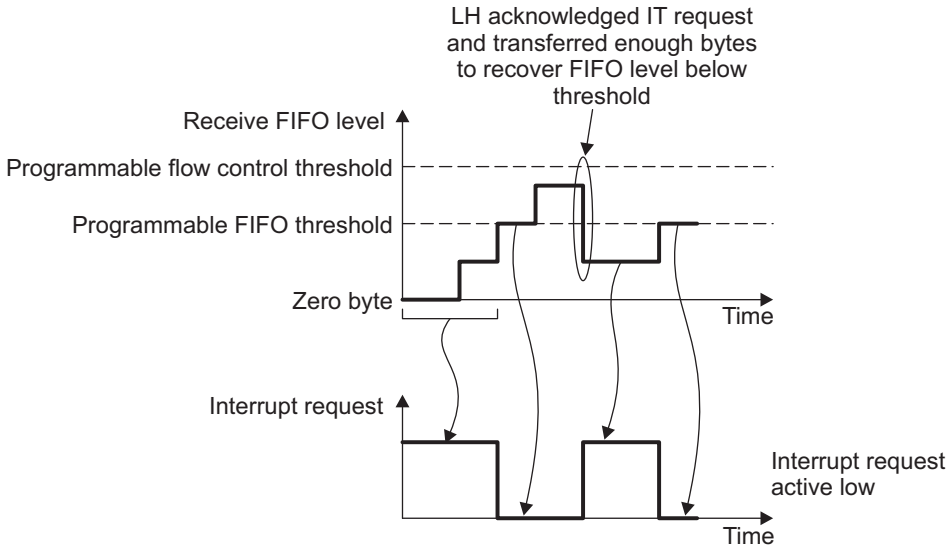
2.5.3 FIFO Interrupt Mode Operation

In FIFO interrupt mode (FIFO_EN bit in FCR = 1, relevant interrupts are enabled by IER), the processor is informed of the status of the receiver and transmitter by an interrupt signal. These interrupts are raised when receive/transmit FIFO threshold (respectively TLR[7:4] and TLR[3:0] or RX_FIFO_TRIG bit in FCR and TX_FIFO_TRIG bit in FCR) are reached; the interrupt signals instruct the Local Host to transfer data to the destination (from the UART module in receive mode and/or from any source to the UART FIFO in transmit mode).

In the case of the UART flow control being enabled along with the interrupt capabilities, the user must ensure that the UART flow control FIFO threshold (RX_FIFO_TRIG_HALT bit in TCR) is greater than or equal to the receive FIFO threshold.

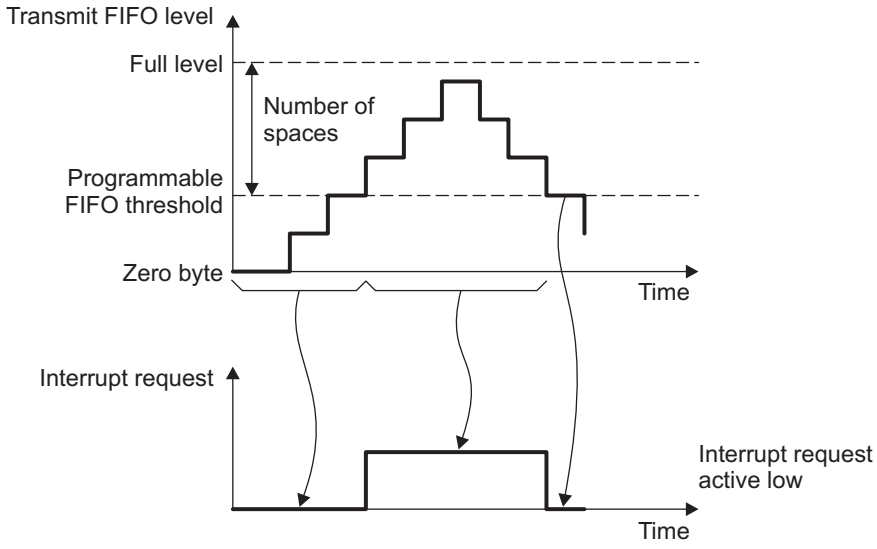
In receive, no interrupt is generated until receive FIFO reaches its threshold. Once low the interrupt can only be de-asserted when the Local Host has handled enough bytes to make the FIFO level below threshold. The flow control threshold is set at a higher value than FIFO threshold. Receive FIFO IT request generation is shown in Figure 19.

Figure 19. Receive FIFO IT Request Generation



In transmit mode, an interrupt request is automatically asserted when TX FIFO is empty. This request is de-asserted when the TX FIFO crossed the threshold level. The interrupt line is de-asserted until a sufficient number of elements have been transmitted to go below TX FIFO threshold. Transmit FIFO IT request generation is shown in Figure 20.

Figure 20. Transmit FIFO IT Request Generation



2.5.4 FIFO Polled Mode Operation

In FIFO polled mode (FIFO_EN bit in FCR= 0 , relevant interrupts disabled via interrupt enable register (IER)) the status of the receiver and transmitter can then be checked by polling the line status register (LSR). This mode is an alternative to the FIFO interrupt mode of operation where the status of the receiver and transmitter is automatically known by means of interrupts sent to the LH.

2.5.5 FIFO DMA Mode Operation

When the EDMA is used to service data to the UART, the following guidelines must be followed:

- EDMA transfer size per event should be the same as UART receive FIFO trigger level (EDMA empties the UART receive FIFO for each event it receives).
- EDMA transfer size per event should be the same as UART transmit FIFO trigger level (EDMA fills the UART transmit FIFO for each event it receives).

For example, the receive FIFO trigger level is chosen to be 20 bytes (when there are 20 bytes in the RX FIFO, an RX EDMA event is generated) and the transmit FIFO trigger level is chosen to be 52 bytes (when there are 52 byte-spaces left, a TX EDMA event is generated). When servicing the receive FIFO, the following EDMA setup is needed: Acnt = 1, Bcnt = 20, Ccnt = number of 20-byte transfers. When servicing the transmit FIFO, the following setup is needed: Acnt = 1, Bcnt = 52, Ccnt = number of 52-byte transfers. For details on the setup of the EDMA, see the *TMS320DM646x DMSoC Enhanced Direct Memory Access (EDMA3) Controller User's Guide* (SPRUEQ5).

2.5.5.1 DMA Signaling

There are four modes of DMA operation, DMA mode 0/1/2/3. They can be selected as follows.

When SCR[0] = 0:	Setting DMA_MODE to 0 enables DMA mode 0.
	Setting DMA_MODE to 1 enables DMA mode 1.
When SCR[0] = 1:	SCR[2:1] determines DMA mode 0 to 3 according to supplementary control register (SCR) description.

For instance:

- if no DMA operation is desired: set SCR[0] to 1 and SCR[2:1] to 00 (DMA_MODE bit in FCR is discarded);
- if DMA mode 1 is desired: either set SCR[0] to 0 and DMA_MODE bit in FCR to 1 or set SCR[0] to 1 SCR[2:1] to 01 (DMA_MODE bit in FCR is discarded).

If the FIFOs are disabled (FIFO_EN bit in FCR = 0), DMA occurs in single character transfers.

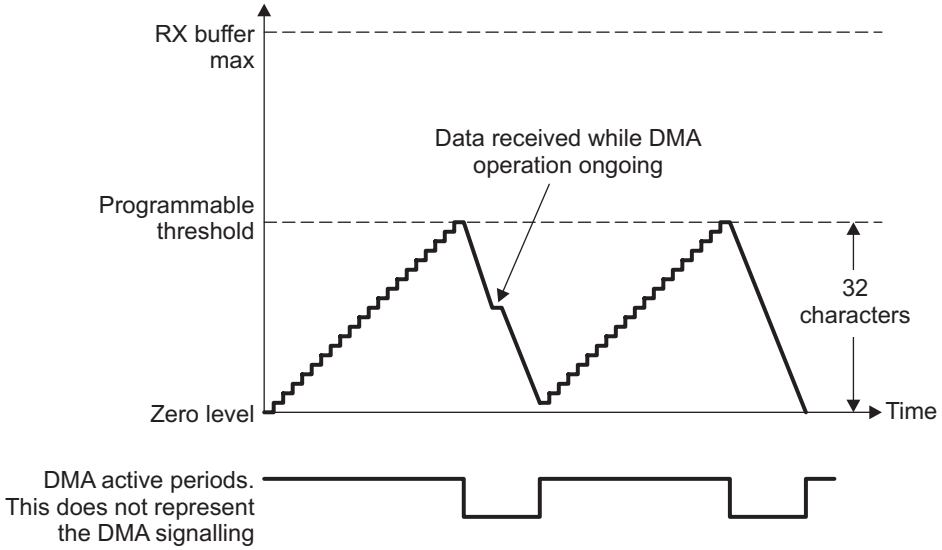
When DMA Mode 0 has been programmed, the signals associated with DMA operation are not active.

2.5.5.2 DMA Transfers (DMA Mode 1, 2, or 3)

Figure 21, Figure 22, Figure 23, and Figure 24 show the supported DMA operations.

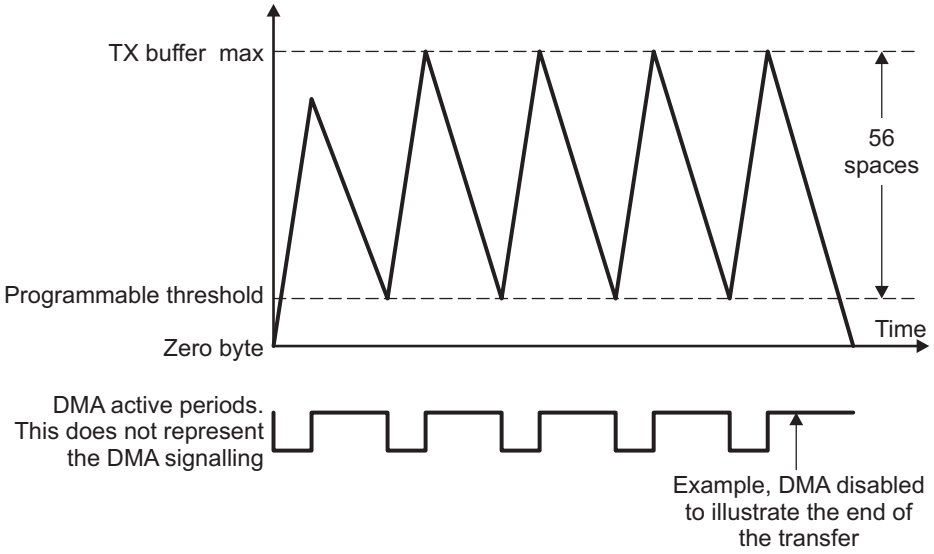
In receive mode (Figure 21), a DMA request is generated as soon as the receive FIFO reaches its threshold level defined in the Trigger Level Register (refer to Table 43). This request is de-asserted when the number of bytes defined by the threshold level has been read by the system DMA.

Figure 21. Receive FIFO DMA Request Generation (32 chars)



In transmit mode (Figure 22), a DMA request is automatically asserted when the transmit FIFO is empty. This request is de-asserted when the number of bytes defined by the number of spaces in the Trigger Level Register (TLR) has been written by the system DMA. If an insufficient number of characters are written, then the DMA request will remain active.

Figure 22. Transmit FIFO DMA Request Generation (56 spaces)

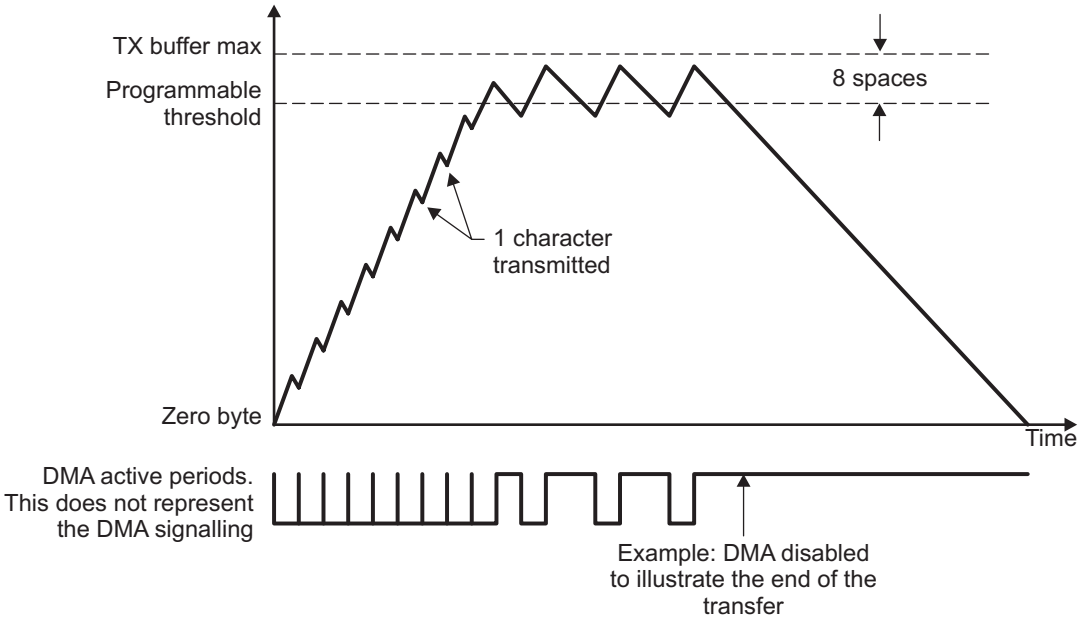


The DMA request is again asserted if the FIFO is able to receive the number of bytes defined by the TLR register (refer to Table 42).

There are a number of ways the threshold can be programmed. Figure 22 shows a DMA transfer that operates with a space setting of 56 which could arise from the use of the auto settings in TX_FIFO_TRIG bit in FCR or the use of the TLR[3:0] concatenated with the TX_FIFO_TRIG bit in FCR. The setting of 56 spaces in the UART IrDA CIR module should correlate with settings of the system DMA so that the buffer does not overflow (program the DMA request size of the Local Host controller to be equal to the number of spaces value in the UART IrDA CIR module).

Figure 23 shows another example with 8 spaces to illustrate the buffer level crossing the space threshold. Again, the Local Host DMA controller settings should correspond with that of the UART IrDA CIR module.

Figure 23. Transmit FIFO DMA Request Generation (8 spaces)

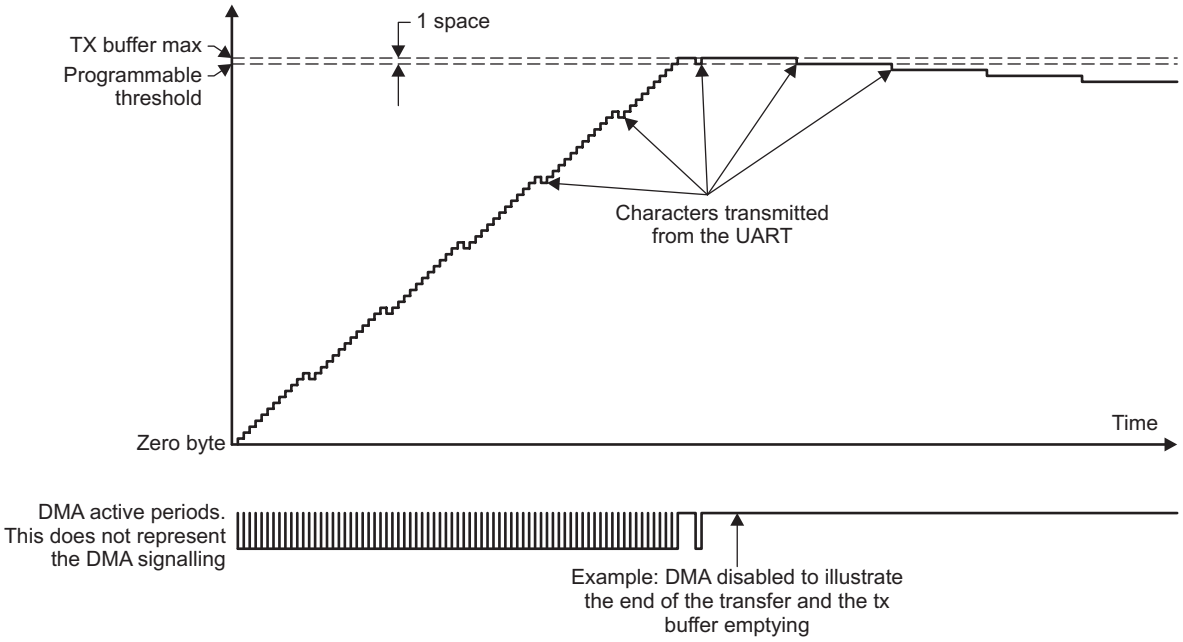


The final example, Figure 24, illustrates the setting of 1 space that uses the DMA for each transfer of one character to the transmit buffer. The buffer is filled at a faster rate than the BAUD rate transmits data to the TX pin. Eventually, the buffer is completely full and the DMA operations stop transferring data to the transmit buffer.

There are two occasions where the buffer holds the maximum amount of data words, shortly after this, the DMA is disabled to illustrate the slower transmission of the data words to the TX pin. Eventually, the buffer will be emptied at the rate specified by the BAUD Rate Settings of the DLL and DLH registers.

Again, the DMA settings should correspond to the system's Local Host DMA controller settings in order to ensure the correct operation of this logic.

Figure 24. Transmit FIFO DMA Request Generation (1 space)



2.5.6 Sleep Mode

2.5.6.1 UART Modes

In UART modes, sleep mode is enabled by writing a 1 to SLEEP_MODE (when ENHANCED_EN bit in EFR = 1).

Sleep mode is entered when:

- The serial data input line, RX is idle.
- The TX FIFO and TX shift register are empty.
- The RX FIFO is empty.
- There are no interrupts pending except THR interrupts.

Sleep mode is a good way to lower power consumption of UART but this state can be achieved only when the UART is set in modem mode. Therefore, even if the UART has no functional key role, it must be initialized in a functional mode to take advantage of sleep mode.

In sleep mode the module clock and baud rate clock are stopped internally. Because most registers are clocked using these clocks the power consumption is greatly reduced. The module wakes up when any change is detected on the RX line, if data is written to the TX FIFO, when there is any change in the state of the modem input pins. An interrupt can be generated on a wake up event by setting RX_CTS_DSR_WAKE_UP_ENABLE in the SCR register to 1. See Section 2.5.2 to know how to manage it.

Note: Writing to the divisor latch registers (DLL and DLH) to set the baud clock (BCLK) must not be done during sleep mode; it is advisable to disable sleep mode using IER[4] before writing to DLL or DLH.

2.5.6.2 IR-IrDA and IR-CIR Modes

In IrDA/CIR modes, sleep mode is enabled by writing a 1 to MDR1[3].

Sleep mode is entered when:

- The serial data input line, RX is idle.
- The TX FIFO and TX shift register are empty.
- The RX FIFO is empty.
- There are no interrupts pending except THR interrupts.

The module wakes up when any change is detected on the RX line or if data is written to the TX FIFO.

2.5.7 Idle Modes

Sleep and AutoIdle modes are embedded power saving features. At the system level, power reduction techniques can be applied by shutting down certain internal clock and power domains of the device.

- The UART supports an idle req - idle ack handshaking protocol. This protocol is used at system level to shut down clocks of the UART in a clean and controlled manner and to switch the UART from the interrupt generation mode to a wakeup generation mode for unmasked events (Refer to SYSC[2] and WER).
- In the wakeup generation mode, interrupt request generation and DMA request generation are disabled.

2.5.8 Break and Time-Out Conditions

2.5.8.1 Time-Out Counter

An RX idle condition is detected when the receiver line, RX, has been high for a time equivalent to 4X programmed word length + 12 bits. The receiver line is sampled midway through each bit.

For sleep mode the counter is reset when there is activity on the RX line.

For the timeout interrupt, the counter only counts when there is data in the RX FIFO and the count is reset when there is activity on the RX line or when the RHR is read.

2.5.8.2 Break Condition

When a break condition occurs the TX line is pulled low. A break condition is activated by setting BREAK_EN in the LCR register. Be aware that the break condition is not aligned on word stream; that is, a break condition can occur in the middle of a character. The only way to send a break condition on a full character, is:

- Reset transmit FIFO (if enabled).
- Wait for transmit shift register becomes empty (LSR[6] = 1).
- Take a guard time according to stop bit definition.
- Set BREAK_EN in the LCR register to 1.

Break condition is asserted as long as BREAK_EN in the LCR register is set to 1.

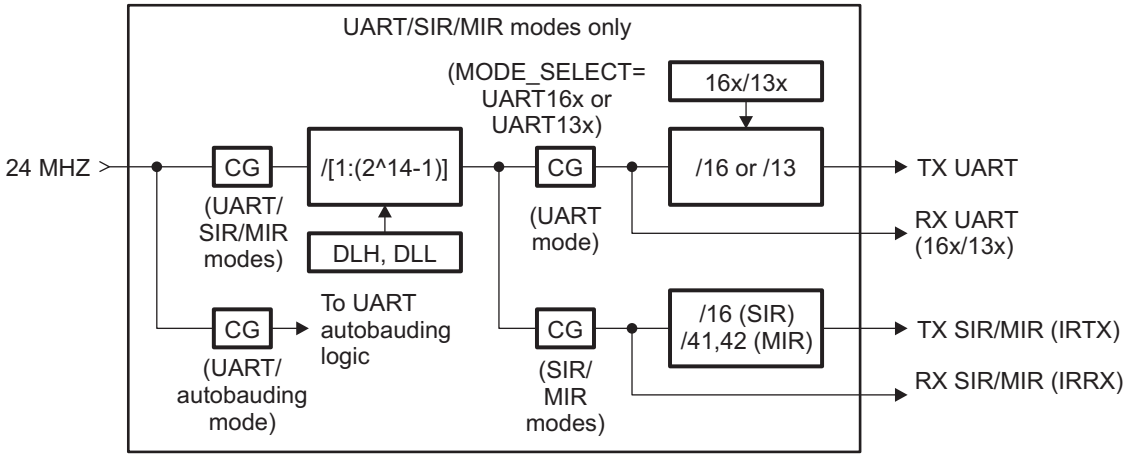
The above functionality (timeout counter and break condition) will only apply to the UART Modem operation and does not extend to the IrDA/CIR modes of operation.

2.5.9 Programmable Baud Rate Generator

The UART IrDA CIR module contains a programmable baud generator and a set of fixed divider that takes the 48 MHz clock input and divides it down to the expected baud-rate.

The baud rate generator and associated controls is depicted in Figure 25. For CIR mode information, see Figure 15.

Figure 25. BAUD Rate Generator



CAUTION

It is recommended to clear the set MODE_SELECT bit in the MDR1 register to 0 before attempting to initialize or modify the clock parameter controls (DLH, DLL). Nonobservance to this rule may result in an unpredictable behavior of the module.

Choosing the appropriate divisor value:

- UART 16× mode: Divisor value = Operating Frequency / (16× baud rate).
- UART 13× mode: Divisor value = Operating Frequency / (13× baud rate).
- SIR mode: Divisor value = Operating Frequency / (16× baud-rate).
- MIR mode: Divisor value = Operating Frequency / (41×/42× baud-rate).

2.5.9.1 UART Baud Rates (24 MHz clock):

Table 6. UART BAUD Rate Settings (24 MHz clock)

Baud Rate	Baud Multiple	DLH,DLL (Decimal)	DLH,DLL (Hex)	Actual Baud Rate	Error (%)[1]
0.3 Kb/s	16x	5000	13h, 88h	0.3 Kb/s	0
1.2 Kb/s	16x	1250	4h, E2h	1.2 Kb/s	0
2.4 Kb/s	16x	625	2h, 71h	2.4 Kb/s	0
14.4 Kb/s	16x	104	0, 68h	14.423 Kb/s	+0.16
28.8 Kb/s	16x	52	0, 34h	28.846Kb/s	+0.16
57.6 Kb/s	16x	26	0, 1Ah	57.692 Kb/s	+0.16
115.2 Kb/s	16x	13	0, Dh	115.38 Kb/s	+0.16
230.4 Kb/s	13x	8	0, 8h	230.77 Kb/s	+0.16
460.8 Kb/s	13x	4	0, 4h	461.54 Kb/s	+0.16
921.6 Mb/s	13x	2	0, 2h	923.08 Mb/s	+0.16
1.8432 Mb/s	13x	1	0, 1h	1.8462 Mb/s	+0.16

(1) Baud rate error and source jitter table values do not include 24MHz reference clock error and jitter.

2.5.9.2 IR Baud Rates (24 MHz clock):

Table 7. IrDA BAUD Rate Settings (24 MHz clock)

Baud Rate	IR Mode	Baud Multiple	Encoding	DLH,DLL	Actual Baud Rate (*= Avg)	Error (%)[1]
2.4 Kb/s	SIR	16x	3/16	625	2.4 Kb/s	0
9.6 Kb/s	SIR	16x	3/16	156	9.6153 Kb/s	+0.16
19.2 Kb/s	SIR	16x	3/16	78	19.231 Kb/s	+0.16
57.6 Kb/s	SIR	16x	3/16	26	57.692 Kb/s	+0.16
0.576 Mb/s	MIR	41x/ 42x	1/4	1	0.5756 Mb/s	0

(1) Baud rate error and source jitter table values do not include 24MHz reference clock error and jitter.

2.5.10 Hardware Flow Control

Hardware flow control is composed of auto-$\overline{\text{CTS}}$ and auto-$\overline{\text{RTS}}$. Auto-$\overline{\text{CTS}}$ and auto-$\overline{\text{RTS}}$ can be enabled/disabled independently by programming EFR[7:6].

With auto-$\overline{\text{CTS}}$, $\overline{\text{CTS}}$ must be active before the module can transmit data.

Auto-$\overline{\text{RTS}}$ only activates the $\overline{\text{RTS}}$ output when there is enough room in the RX FIFO to receive data and de-activates the $\overline{\text{RTS}}$ output when the RX FIFO is sufficiently full. The HALT and RESTORE trigger levels in the TCR determine the levels at which $\overline{\text{RTS}}$ is activated/de-activated.

If both auto-$\overline{\text{CTS}}$ and auto-$\overline{\text{RTS}}$ are enabled, data transmission does not occur unless the receiver FIFO has empty space. Thus, overrun errors are eliminated during hardware flow control. If not enabled, overrun errors occur if the transmit data rate exceeds the receive FIFO latency.

2.5.10.1 Auto-$\overline{RTS}$

Auto-$\overline{RTS}$ data flow control originates in the receiver block (see functional block diagram). The receiver FIFO trigger levels used in auto-$\overline{RTS}$ are stored in the TCR. $\overline{RTS}$ is active if the RX FIFO level is below the HALT trigger level in RX_FIFO_TRIG_HALT bit in TCR. When the receiver FIFO HALT trigger level is reached, $\overline{RTS}$ is de-asserted. The sending device (for example, another UART) may send an additional byte after the trigger level is reached because it may not recognize the de-assertion of $\overline{RTS}$ until it has begun sending the additional byte. $\overline{RTS}$ is automatically reasserted once the receiver FIFO reaches the RESUME trigger level programmed via RX_FIFO_TRIG_START bit in TCR. This reassertion requests the sending device to resume transmission.

2.5.10.2 Auto-$\overline{CTS}$

The transmitter circuitry checks $\overline{CTS}$ before sending the next data byte. When $\overline{CTS}$ is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, $\overline{CTS}$ must be de-asserted before the middle of the last stop bit that is currently being sent. The auto-$\overline{CTS}$ function reduces interrupts to the host system. When auto-$\overline{CTS}$ flow control is enabled, the $\overline{CTS}$ state changes need not trigger host interrupts because the device automatically controls its own transmitter. Without auto-$\overline{CTS}$, the transmitter sends any data present in the transmit FIFO and a receiver overrun error can result.

2.5.11 Software Flow Control

Software flow control is enabled through the enhanced feature register (EFR) and the modem control register (MCR). Different combinations of software flow control can be enabled by setting different combinations of EFR[3:0].

There are two other enhanced features relating to S/W flow control:

- XON any function (XON_EN bit in MCR): operation will resume after receiving any character after recognizing the XOFF character.

  Note: The XON-any character is written into the RX FIFO even if it is a software flow character.
- Special character (ERF[5]): Incoming data is compared to XOFF2. Detection of the special character sets the XOFF interrupt (IIR[4]) but does not halt transmission. The XOFF interrupt is cleared by a read of the IIR. The special character is transferred to the RX FIFO.

2.5.11.1 Receive (RX)

When software flow control operation is enabled, the UART compares incoming data with XOFF1/2 programmed characters (in certain cases XOFF1 and XOFF2 must be received sequentially). When the correct XOFF characters are received, transmission is halted after completing transmission of the current character. XOFF detection also sets IIR[4] (if enabled via IER[5]) and causes the interrupt line to go low.

To resume transmission a XON1/2 character must be received (in certain cases XON1 and XON2 must be received sequentially). When the correct XON characters are received IIR[4] is cleared and the XOFF interrupt disappears.

If a parity, framing, or break error occurs while receiving a software flow control character, this character will be treated as normal data and will be written to the RX FIFO.

When XON-Any and special character detect are disabled and software flow control is enabled no valid XON or XOFF characters are written to the RX FIFO. For example, EFR[1:0] = 2h, if XON1 and XOFF1 characters are received they do not get written to the RX FIFO.

In the case where pairs of software flow characters are programmed to be received sequentially (EFR[1:0] = 3h) the software flow characters are not written to the RX FIFO if they are received sequentially. However, received XON1/XOFF1 characters must be written to the RX FIFO if the subsequent character is not XON2/XOFF2.

2.5.11.2 Transmit (TX)

XOFF1: Two characters are transmitted when the RX FIFO has passed the programmed trigger level RX_FIFO_TRIG_HALT bit in TCR.

XON1: Two characters are transmitted when the RX FIFO reaches the trigger level programmed via RX_FIFO_TRIG_START bit in TCR.

If after an XOFF character has been sent, software flow control is disabled, the module transmits XON characters automatically to enable normal transmission to proceed.

The transmission of XOFF/XON (s) follows the exact same protocol as transmission of an ordinary byte from the TX FIFO. This means that even if the word length is set to be 5, 6 or 7 characters then the 5, 6 or 7 least significant bits of XOFF1,2/XON1,2 are transmitted. Note that the transmission of 5, 6, or 7 bits of a character is seldom done but this functionality is included to maintain compatibility with earlier designs.

It is assumed that software flow control and hardware flow control will never be enabled simultaneously.

2.5.12 Frame Closing

There are two ways by which a transmission-frame can be properly terminated.

- Frame-length method: Frame-length method is selected when FRAME_END_MODE bit in MDR1= 0. The LH writes the frame-length value to TXFLH and TXFLL registers. The device automatically attaches end flags to the frame once the number of bytes transmitted becomes equal to the frame-length value.
- Set-EOT bit Method: Set-EOT bit method is selected when FRAME_END_MODE bit in MDR1 = 1. The LH writes 1 to EOT_EN bit in ACREG (EOT bit) just before it writes the last byte to the TX FIFO. When the LH writes the last byte to the TX FIFO, the device internally sets the tag bit for that particular character in the TX FIFO. As the TX state machine reads data from the TX FIFO it uses this tag-bit information to attach end flags and properly terminate the frame.

2.5.13 Store and Controlled Transmission (SCT)

In SCT the LH first starts writing data into the TX FIFO. Then, after it writes a part of a frame (for a bigger frame) or a whole frame (a small frame, that is, supervisory frame), it writes a 1 to SCTX_EN bit in ACREG (deferred TX start) to start transmission. SCT is enabled when SCT bit in MDR1 = 1. This method of transmission is different from the normal mode, where transmission of data starts immediately after data is written to the TX FIFO. SCT is useful to send short frames without TX underrun.

2.5.14 Underrun During Transmission

Underrun in transmission occurs when the TX FIFO becomes empty before the end of the frame is transmitted. When underrun occurs, the device closes the frame with end-flags but attaches an incorrect CRC value. The receiving device detects a CRC error and discards the frame; it can then ask for a re-transmission. Underrun also causes an internal flag to be set which disables further transmission. Before the next frame can be transmitted the system (LH) must:

- Reset the TX FIFO
- Read the RESUME register. This clears the internal flag.

This functionality can be disabled with DIS_TX_UNDERRUN bit in ACREG, compensated by the extension of the stop bit in transmission in case the TX FIFO is empty.

2.5.15 Overrun During Receive

Overrun occurs during receive if the RX state machine tries to write data into the RX FIFO when it is already full. When overrun occurs, the device interrupts the LH with IIR[3] and discards the remaining portion of the frame. Overrun also causes an internal flag to be set, which disables further reception. Before the next frame can be received the system (LH) must:

- Reset the RX FIFO
- Read the RESUME register. This clears the internal flag

2.5.16 Status FIFO

In IrDA modes, a status FIFO is used to record the received frame status. When a complete frame is received, the length of the frame and the error bits associated with the frame are written into the status FIFO.

The frame length and error status can be read by reading SFREGL/H and SFLSR. Reading the SFLSR causes the read pointer to be incremented. The status FIFO is eight entries deep and therefore can hold the status of eight frames.

The LH uses the frame-length information to locate the frame-boundary in the received frame data. The LH can screen bad frames using the error-status information and later request the sender to resend only the bad frames.

This status FIFO can be used very effectively in DMA as the LH need not be interrupted every time a frame is received but only whenever the programmed status FIFO trigger level is reached.

2.6 Interrupt and EDMA Event Assignment

2.6.1 Interrupts

The UART generates interrupts to the CPU as shown in Table 8.

Table 8. UART Interrupts

ARM Event	Acronym	Source
40	UARTINT0	UART0
41	UARTINT1	UART1
42	UARTINT2	UART2

2.6.2 EDMA Events

The UART generates events to the EDMA controller as shown in Table 9.

Table 9. UART EDMA Events

Event	Acronym	Source
18	URXEVT0	UART0 Receive Event
19	UTXEVT0	UART0 Transmit Event
20	URXEVT1	UART1 Receive Event
21	UTXEVT1	UART1 Transmit Event
22	URXEVT2	UART2 Receive Event
23	UTXEVT2	UART2 Transmit Event

2.7 UART Bootloader

UART0 can be used to bootload the device. For detailed information on bootloading, see the *TMS320DM646x DMSoC ARM Subsystem Reference Guide* (SPRUEP9).

2.8 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 *TMS320DM646x DMSoC ARM Subsystem Reference Guide* (SPRUEP9).

2.9 Programming Examples

This section outlines the programming stages to operate one UART in Modem and IrDA mode. The first stage covers software reset of the module (interrupts, status, and controls), and it is identical for all different modes. The second stage deals with FIFO configuration and enable, and it is identical for all three different modes. The last stage deals with the baud rate data and stop configuration, and it is different for different modes.

2.9.1 UART Reset

Table 10. UART Reset

Register	Description
LCR = BFh	This enables access to EFR.
EFR[4] = 1	This is the first requirement to access MCR and IER.
LCR[7] = 0	This is the second requirement to access MCR and IER.
IER = 0	Disable interrupt.
MCR = 0	Force control signals inactive.
LCR[6] = 0	UART breaks are removed.
MDR1 = 7h	UART is in reset or disabled.

Alternatively, the SYSC[1] can be set to one to instigate a hardware reset from the generic synchronous reset module. The reset progress can be monitored via the SYSS[0]. Once complete, the above sequence must ensure the UART is in the equivalent disabled mode with reference to MODE_SELECT in the MDR1 register.

2.9.2 UART FIFO Configuration

To write into both the TLR and TCR registers, ENHANCED_EN bit in EFR must be set to 1 and TCR_TLR bit in MCR to 1. To write into FCR, ENHANCED_EN bit in EFR must be set to 1. Note that ENHANCED_EN bit in EFR = 1 has already been done in the previous section; hence, a simple write to TCR_TLR bit in MCR is necessary.

Table 11. UART FIFO Configuration

Register	Description
MCR[6] = 1	This enables access to TLR and TCR.
TCR, TLR, FCR	Set to desired value.
LCR = BFh	Provides access to EFR and disable access to TCR, TLR, and FCR.
EFR[4] = 0	Clear EFR[4].
LCR[7] = 0	This enables access to MCR.
MCR[6] = 0	Clear MCR[6].

2.9.3 UART Baud Rate Configuration

2.9.3.1 Modem Mode

Table 12. UART Baud Rate Configuration - Modem Mode

Register	Description
LCR	Set to desired value.
LCR[7] = 1	Gives access to DLH and DLL registers.
DLH, DLL	Set to desired value.
LCR[7] = 0	Removes access to DLH and DLL registers.
IER	Set to desired value.
MDR1[2:0] = 0	Enables UART without autobaud.

2.9.3.2 SIR Mode

Table 13. UART Baud Rate Configuration - SIR Mode

Register	Description
LCR	Set to desired value.
LCR[7] = 1	Gives access to DLH and DLL registers.
DLH, DLL	Set to desired value.
LCR[7] = 0	Removes access to DLH and DLL registers.
IER	Set to desired value.
EBLR, TXFLL, RXFLL	Set to desired value.
MDR1[2:0] = 1	Enables UART SIR mode.

2.9.3.3 CIR Mode

Table 14. UART Baud Rate Configuration - CIR Mode

Register	Description
LCR	Set to desired value.
LCR[7] = 1	Gives access to DLH and DLL registers.
DLH, DLL	Set to desired value.
LCR[7] = 0	Removes access to DLH and DLL registers.
IER	Set to desired value.
EBLR, TXFLL, RXFLL	Set to desired value.
MDR1[2:0] = 1	Enables UART SIR mode.

2.10 Reset Considerations

The UART peripheral has two reset sources: software reset and hardware reset.

2.10.1 Software Reset Considerations

The UART module can be disabled at any time by setting MODE_SELECT in the MDR1 register to 7h. The entire UART module may also be reset through the Power and Sleep Controller (PSC). Note that from the UART perspective, this reset appears as a hardware reset to the entire module. For detailed information on the PSC, see the *TMS320DM646x DMSoC ARM Subsystem Reference Guide* (SPRUEP9).

2.10.2 Hardware Reset Considerations

When the UART is reset due to device reset, the entire module is disabled.

2.11 Emulation Considerations

The UART does not support emulation suspend.

3 Registers

Table 15 lists the memory-mapped registers for the UART. See the device-specific data manual for the memory address of these registers.

The following registers are accessible by the local host (LH) at address = module base address + address offset. The module base address is the module start address. The module base address is 01C2 0000h for UART0, 01C2 0400h for UART1, and 01C2 0800h for UART2.

S = 1 for 8-bit aligned addresses
2 for 16-bit aligned addresses
4 for 32-bit aligned addresses

Note: Register address offsets depend upon the module address alignment at the system top level.

Table 15. Universal Asynchronous Receiver/Transmitter (UART) Registers

Offset	Acronym	Register Description	Section
0 × S	RHR	Receive Holding Register	Section 3.1
0 × S	THR	Transmit Holding Register	Section 3.2
0 × S	DLL	Divisor Latch Low Register	Section 3.3
1 × S	IER-UART Mode	Interrupt Enable Register UART Mode	Section 3.4.1
1 × S	IER-IrDA Mode	Interrupt Enable Register IrDA Mode	Section 3.4.2
1 × S	IER-CIR Mode	Interrupt Enable Register CIR Mode	Section 3.4.3
1 × S	DLH	Divisor Latch High Register	Section 3.5
2 × S	IIR-UART Mode	Interrupt Identification Register UART Mode	Section 3.6.1
2 × S	IIR-IrDA Mode	Interrupt Identification Register IrDA Mode	Section 3.6.2
2 × S	IIR-CIR Mode	Interrupt Identification Register CIR Mode	Section 3.6.3
2 × S	FCR	FIFO Control Register	Section 3.7
2 × S	EFR	Enhanced Feature Register	Section 3.8
3 × S	LCR	Line Control Register	Section 3.9

Table 15. Universal Asynchronous Receiver/Transmitter (UART) Registers (continued)

Offset	Acronym	Register Description	Section
4 × S	MCR	Modem Control Register	Section 3.10
4 × S	XON1	XON1 Character Register	Section 3.11
5 × S	LSR-UART Mode	Line Status Register UART Mode	Section 3.12.1
5 × S	LSR-IrDA Mode	Line Status Register IrDA Mode	Section 3.12.2
5 × S	LSR-CIR Mode	Line Status Register CIR Mode	Section 3.12.3
5 × S	XON2	XON2 Character Register	Section 3.13
6 × S	MSR	Modem Status Register	Section 3.14
6 × S	TCR	Transmission Control Register	Section 3.15
6 × S	XOFF1	XOFF1 Character Register	Section 3.16
7 × S	SPR	Scratchpad Register	Section 3.17
7 × S	TLR	Trigger Level Register	Section 3.18
7 × S	XOFF2	XOFF2 Character Register	Section 3.19
8 × S	MDR1	Mode Definition Register 1	Section 3.20
9 × S	MDR2	Mode Definition Register 2	Section 3.21
Ah × S	SFLSR	Status FIFO Line Status Register	Section 3.22
Ah × S	TXFLL	Transmit Frame Length Low Register	Section 3.23
Bh × S	RESUME	Resume Register	Section 3.24
Bh × S	TXFLH	Transmit Frame Length High Register	Section 3.25
Ch × S	SFREGL	Status FIFO Low Register	Section 3.26
Ch × S	RXFLL	Receive Frame Length Low Register	Section 3.27
Dh × S	SFREGH	Status FIFO High Register	Section 3.28
Dh × S	RXFLH	Receive Frame Length High Register	Section 3.29
Eh × S	BLR	Begin of Frame Control Register	Section 3.30
Eh × S	UASR	Autobauding Status Register	Section 3.31
Fh × S	ACREG	Auxiliary Control Register	Section 3.32
10h × S	SCR	Supplementary Control Register	Section 3.33
11h × S	SSR	Supplementary Status Register	Section 3.34
12h × S	EBLR	Begin of Frame Length Register	Section 3.35
14h × S	MVR	Module Version Register	Section 3.36
15h × S	SYSC	System Configuration Register	Section 3.37
16h × S	SYSS	System Status Register	Section 3.38
17h × S	WER	Wake-Up Enable Register	Section 3.39
18h × S	CFPS	Carrier Frequency Prescaler Register	Section 3.40

Table 16. UART/IrDA/CIR Register Program

	Register					
	LCR[7] = 0		LCR[7] = 1 & LCR[7:0] ≠ BFh		LCR[7:0] = BFh	
Offset	Read	Write	Read	Write	Read	Write
0 × S	RHR	THR	DLL	DLL	DLL	DLL
1 × S	IER(1)	IER(1)	DLH	DLH	DLH	DLH
2 × S	IIR	FCR(2)	IIR	FCR(2)	EFR	EFR
3 × S	LCR	LCR	LCR	LCR	LCR	LCR
4 × S	MCR(2)	MCR(2)	MCR(2)	MCR(2)	XON1/ADDR1	XON1/ADDR1
5 × S	LSR	-	LSR	-	XON2/ADDR2	XON2/ADDR2
6 × S	MSR/TCR(3)	TCR(3)	MSR/TCR(3)	TCR(3)	XOFF1/TCR(3)	XOFF1/TCR(3)
7 × S	SPR/TLR(3)	SPR/TLR(3)	SPR/TLR(3)	SPR/TLR(3)	XOFF2/TLR(3)	XOFF2/TLR(3)
8 × S	MDR1	MDR1	MDR1	MDR1	MDR1	MDR1
9 × S	MDR2	MDR2	MDR2	MDR2	MDR2	MDR2
Ah × S	SFLSR	TXFLL	SFLSR	TXFLL	SFLSR	TXFLL
Bh × S	RESUME	TXFLH	RESUME	TXFLH	RESUME	TXFLH
Ch × S	SFREGL	RXFLL	SFREGL	RXFLL	SFREGL	RXFLL
Dh × S	SFREGH	RXFLH	SFREGH	RXFLH	SFREGH	RXFLH
Eh × S	BLR	BLR	UASR	-	UASR	-
Fh × S	ACREG	ACREG	-	-	-	-
10h × S	SCR	SCR	SCR	SCR	SCR	SCR
11h × S	SSR	-	SSR	-	SSR	-
12h × S	EBLR	EBLR	-	-	-	-
14h × S	MVR	-	MVR	-	MVR	-
15h × S	SYSC	SYSC	SYSC	SYSC	SYSC	SYSC
16h × S	SYSS	-	SYSS	-	SYSS	-
17h × S	WER	WER	WER	WER	WER	WER
18h × S	CFPS	CFPS	CFPS	CFPS	CFPS	CFPS

(1) In UART modes, CTS_IT, RTS_IT, XOFF_IT, and SLEEP_MODE bits in IER can only be written when ENHANCED_EN in EFR = 1. In IrDA/CIR modes, ENHANCED_EN in EFR has no impact on these bits.
(2) TCR_TLR and XON_EN bits in MCR and TX_FIFO_TRIG bit in FCR can only be written when ENHANCED_EN in EFR = 1.
(3) Transmission control register (TCR) and trigger level register (TLR) are accessible only when ENHANCED_EN in EFR = 1 and TCR_TLR in MCR = 1.

3.1 Receive Holding Register (RHR)

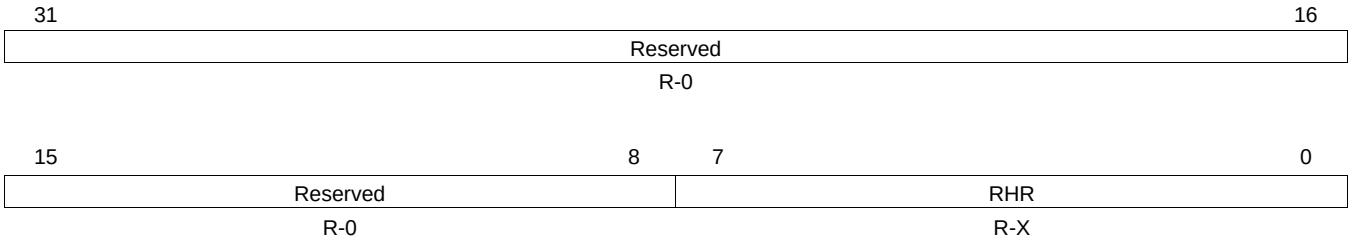
The receive holding register (RHR) is shown in Figure 26 and described in Table 17.

Offset Address (hex): 0 × S and DIV_EN in LCR = 0 and read to access this register.

The receiver section consists of the receiver holding register (RHR) and the receiver shift register. The RHR is actually a 64-byte FIFO. The receiver shift register receives serial data from RX input. The data is converted to parallel data and moved to the RHR. If the FIFO is disabled location zero of the FIFO is used to store the single data character.

If an overflow occurs the data in the RHR is not overwritten.

Figure 26. Receive Holding Register (RHR)



LEGEND: R = Read only; X = Unknown; -*n* = value after reset.

Table 17. Receive Holding Register (RHR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	RHR	0-FFh	Receive Holding Register.

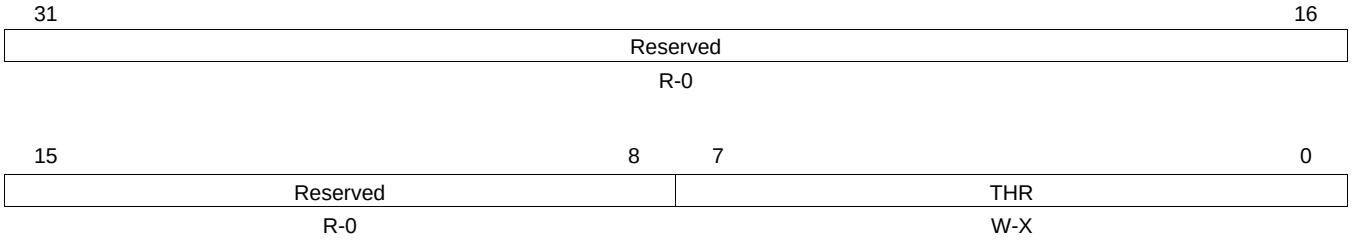
3.2 Transmit Holding Register (THR)

The transmit holding register (THR) is shown in Figure 27 and described in Table 18.

Offset Address (hex): 0 × S and DIV_EN in LCR = 0 and write.

The transmitter section consists of the transmit holding register (THR) and the transmit shift register. The transmit holding register is actually a 64-byte FIFO. The LH writes data to the THR. The data is placed into the transmit shift register where it is shifted out serially on the TX output. If the FIFO is disabled location zero of the FIFO is used to store the data.

Figure 27. Transmit Holding Register (THR)



LEGEND: R = Read only; W = Write only; X = Unknown; -*n* = value after reset.

Table 18. Transmit Holding Register (THR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	THR	0-FFh	Transmit Holding Register.

3.3 Divisor Latch Low Register (DLL)

Offset Address (hex): 0 × S and DIV_EN in LCR = 1.

The divisor latch low register (DLL) is shown in Figure 28 and described in Table 19.

Both DLL and Section 3.5 store the 14-bit divisor for generation of the baud clock in the baud rate generator. Section 3.5 stores the most significant part of the divisor. DLL stores the least significant part of the divisor.

Note: DLL and Section 3.5 can only be written to before SLEEP_MODE in IER-UART Mode is enabled (i.e. before SLEEP_MODE in IER-UART Mode is set).

Figure 28. Divisor Latch Low Register (DLL)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		CLOCK_LSB	
R-0		R/W-0	

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

Table 19. Divisor Latch Low Register (DLL)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	CLOCK_LSB	0-FFh	Used to store the 8-bit LSB divisor value.

3.4 Interrupt Enable Register (IER)

Offset Address (hex): 1h × S and DIV_EN in LCR = 0 (and ENHANCED_EN in EFR = 1 for IER[7:4] - UART modes only).

3.4.1 Interrupt Enable Register UART Mode (IER-UART Mode)

The interrupt enable register UART mode (IER-UART Mode) is shown in Figure 29 and described in Table 20.

The interrupt enable register (IER) can be programmed to enable/disable any interrupt. There are seven types of interrupt in this mode: receiver error, RHR interrupt, THR interrupt, XOFF received and $\overline{CTS}$/$\overline{RTS}$ change of state from low to high. Each interrupt can be enabled/disabled individually. There is also a sleep mode enable bit.

Note: SLEEP_MODE, XOFF_IT, RTS_IT, and CTS_IT can only be written when ENHANCED_EN in EFR = 1.

Figure 29. Interrupt Enable Register UART Mode (IER-UART Mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTS_IT	RTS_IT	XOFF_IT	SLEEP_MODE	MODEM_STS_IT	LINE_STS_IT	THR_IT	RHR_IT
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

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

Table 20. Interrupt Enable Register UART Mode (IER-UART Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	CTS_IT		UART mode only.
		0	Disables the $\overline{CTS}$ interrupt.
		1	Enables the $\overline{CTS}$ interrupt.
			Note: Can only be written when ENHANCED_EN in EFR = 1.
6	RTS_IT		UART mode only.
		0	Disables the $\overline{RTS}$ interrupt.
		1	Enables the $\overline{RTS}$ interrupt.
			Note: Can only be written when ENHANCED_EN in EFR = 1.
5	XOFF_IT		UART mode only.
		0	Disables the XOFF interrupt.
		1	Enables the XOFF interrupt.
			Note: Can only be written when ENHANCED_EN in EFR = 1.
4	SLEEP_MODE		UART mode only.
		0	Disables sleep mode.
		1	Enables sleep mode (stop baud rate clock when the module is inactive).
			Note: Can only be written when ENHANCED_EN in EFR = 1.

Table 20. Interrupt Enable Register UART Mode (IER-UART Mode) (continued)

Bit	Field	Value	Description
3	MODEM_STS_IT		UART mode only.
		0	Disables the modem status register interrupt.
		1	Enables the modem status register interrupt.
2	LINE_STS_IT		UART mode only.
		0	Disables the receiver line status interrupt.
		1	Enables the receiver line status interrupt.
1	THR_IT		THR interrupt.
		0	Disables the THR interrupt.
		1	Enables the THR interrupt.
0	RHR_IT		RHR interrupt.
		0	Disables the RHR interrupt and time out interrupt.
		1	Enables the RHR interrupt and time out interrupt.

3.4.2 Interrupt Enable Register IrDA Mode (IER-IrDA Mode)

The interrupt enable register IrDA mode (IER-IrDA Mode) is shown in Figure 30 and described in Table 21.

There are 8 types of interrupt in these modes, received EOF, LSR interrupt, TX status, status FIFO interrupt, RX overrun, last byte in RX FIFO, THR interrupt and RHR interrupt and they can be enabled/disabled individually.

Note: The TX_STATUS_IT interrupt reflects two possible conditions. The IRTX_UNDERRUN bit in MDR2 should be read to determine the status in the event of this interrupt.

Figure 30. Interrupt Enable Register IrDA Mode (IER-IrDA Mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
EOF_IT	LINE_STS_IT	TX_STATUS_IT	STS_FIFO_TRIG_IT	RX_OVERRUN_IT	LAST_RX_BYTE_IT	THR_IT	RHR_IT
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

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

Table 21. Interrupt Enable Register IrDA Mode (IER-IrDA Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	EOF_IT		IrDA mode only.
		0	Disables the received EOF interrupt.
		1	Enables the received EOF interrupt.
6	LINE_STS_IT		IrDA mode only.
		0	Disables the receiver line status interrupt.
		1	Enables the receiver line status interrupt.
5	TX_STATUS_IT		IrDA and CIR mode only.
		0	Disables the TX status interrupt.
		1	Enables the TX status interrupt.
4	STS_FIFO_TRIG_IT		IrDA mode only.
		0	Disables status FIFO trigger level interrupt.
		1	Enables status FIFO trigger level interrupt.
3	RX_OVERRUN_IT		IrDA and CIR mode only.
		0	Disables the RX overrun interrupt.
		1	Enables the RX overrun interrupt.
2	LAST_RX_BYTE_IT		IrDA mode only.
		0	Disables the last byte of frame in RX FIFO interrupt.
		1	Enables the last byte of frame in RX FIFO interrupt.
1	THR_IT		THR interrupt.
		0	Disables the THR interrupt.
		1	Enables the THR interrupt.

Table 21. Interrupt Enable Register IrDA Mode (IER-IrDA Mode) (continued)

Bit	Field	Value	Description
0	RHR_IT		RHR interrupt.
		0	Disables the RHR interrupt.
		1	Enables the RHR interrupt.

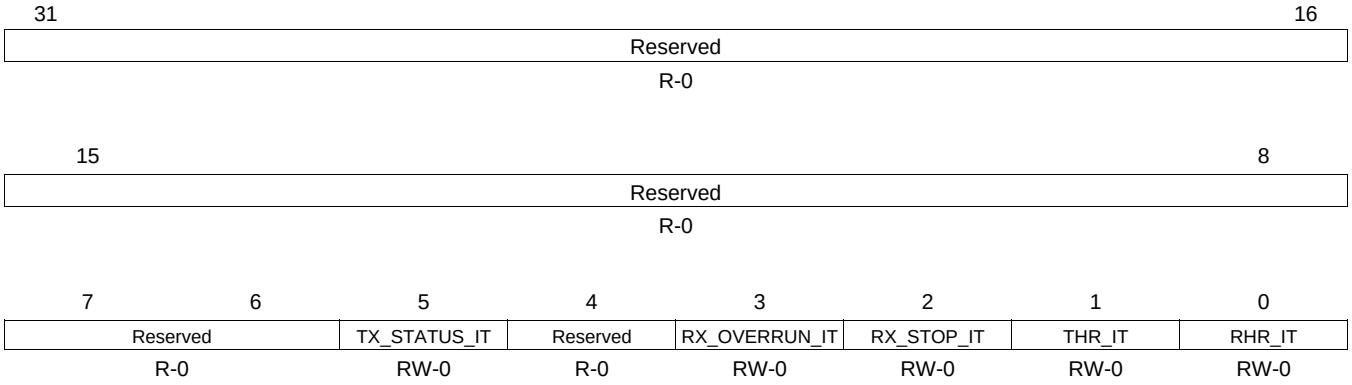
3.4.3 Interrupt Enable Register CIR Mode (IER-CIR Mode)

The interrupt enable register CIR mode (IER-CIR Mode) is shown in Figure 31 and described in Table 22.

There are 6 types of interrupt in these modes, TX status, status FIFO interrupt, RX overrun, last byte in RX FIFO, THR interrupt and RHR interrupt and they can be enabled/disabled individually.

Note: The RX_STOP_IT interrupt is generated based on the value set in the BOF Length register (EBLR). In IR-CIR mode, contrary to the IR-IrDA mode, the TX_STATUS_IT has only one meaning corresponding to the case IRTX_UNDERRUN bit in MDR2 = 0.

Figure 31. Interrupt Enable Register CIR Mode (IER-CIR Mode)



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

Table 22. Interrupt Enable Register CIR Mode (IER-CIR Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-6	Reserved	0	Not used in CIR Mode
5	TX_STATUS_IT		IrDA and CIR mode only.
		0	Disables the TX status interrupt.
		1	Enables the TX status interrupt.
4	Reserved	0	Not used in CIR Mode
3	RX_OVERRUN_IT		IrDA and CIR mode only.
		0	Disables the RX overrun interrupt.
		1	Enables the RX overrun interrupt.
2	RX_STOP_IT		CIR mode only.
		0	Disables the receive stop interrupt.
		1	Disables the receive stop interrupt.
1	THR_IT		THR interrupt.
		0	Disables the THR interrupt.
		1	Enables the THR interrupt.
0	RHR_IT		RHR interrupt.
		0	Disables the RHR interrupt.
		1	Enables the RHR interrupt.

3.5 Divisor Latch High Register (DLH)

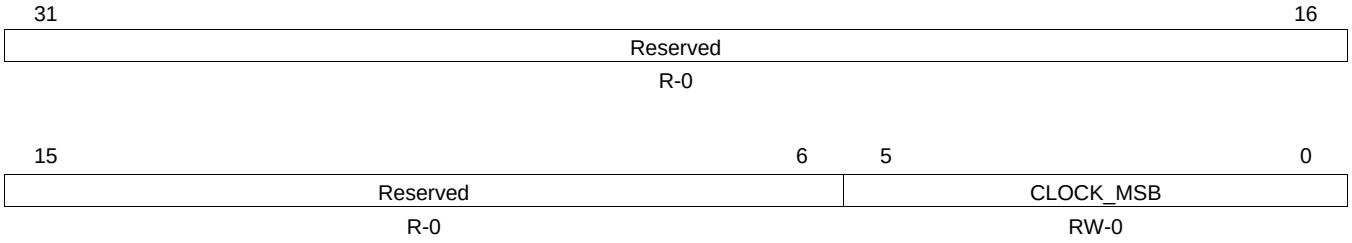
Offset Address (hex): 1h × S and DIV_EN in LCR = 1.

The divisor latch high register (DLH) is shown in Figure 32 and described in Table 23.

Both Section 3.3 and DLH store the 14-bit divisor for generation of the baud clock in the baud rate generator. DLH stores the most significant part of the divisor. Section 3.3 stores the least significant part of the divisor.

Note: Section 3.3 and DLH can only be written to before SLEEP_MODE in IER-UART Mode is enabled (that is, before the SLEEP_MODE bit in IER-UART Mode is set).

Figure 32. Divisor Latch High Register (DLH)



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

Table 23. Divisor Latch High Register (DLH)

Bit	Field	Value	Description
31-6	Reserved	0	Reserved.
5-0	CLOCK_MSB	0-3Fh	Used to store the 6-bit MSB divisor value.

3.6 Interrupt Identification Register (IIR)

Offset Address (hex): 2h × S and LCR ≠ BFh and read.

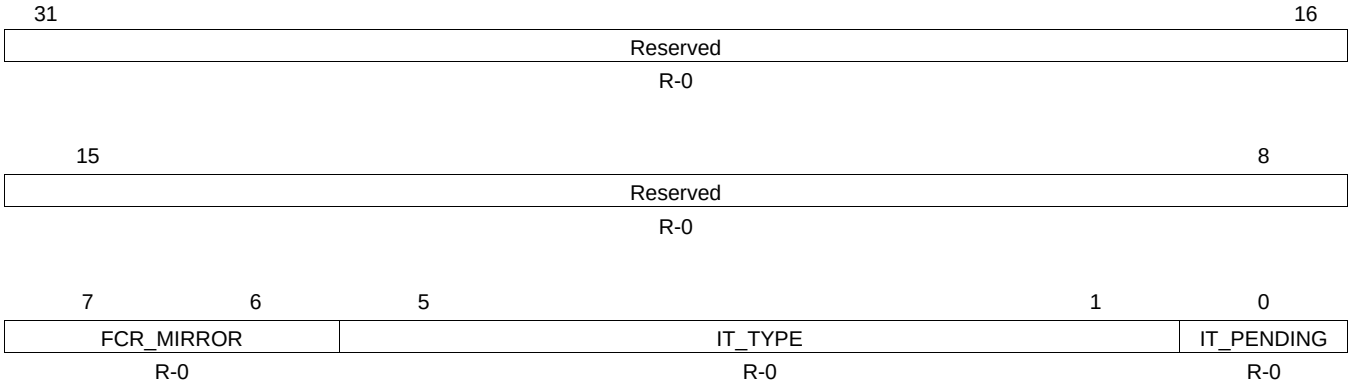
3.6.1 Interrupt Identification Register UART Mode (IIR-UART Mode)

The interrupt identification register UART mode (IIR-UART Mode) is shown in Figure 33 and described in Table 24.

The IIR is a read-only register, which provides the source of the interrupt in a prioritized manner.

Note: An interrupt source can be flagged only if enabled in the IER register. Clearing of interrupts is described in Table 3.

Figure 33. Interrupt Identification Register UART Mode (IIR-UART Mode)



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

Table 24. Interrupt Identification Register UART Mode (IIR-UART Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-6	FCR_MIRROR	0-3h	Mirror the contents of the FIFO_EN bit in FCR on both bits.
5-1	IT_TYPE	0-1Fh	Interrupt type.
		0	Modem Interrupt. (Priority 4)
		1h	THR interrupt. (Priority 3)
		2h	RHR interrupt. (Priority 2)
		3h	Receiver line status error. (Priority 1)
		4h-5h	Reserved
		6h	Rx timeout. (Priority 2)
		7h	Reserved
		8h	XOFF/Special character. (Priority 5)
		9h-Fh	Reserved
		10h	$\overline{CTS}$, $\overline{RTS}$, $\overline{DSR}$ change stat from active (low) to inactive (high). (Priority 6)
		11h-1Fh	Reserved
0	IT_PENDING		Interrupt pending. UART mode only.
		0	An interrupt is pending.
		1	No interrupt is pending.

3.6.2 Interrupt Identification Register IrDA Mode (IIR-IrDA mode)

The interrupt identification register IrDA mode (IIR-IrDA mode) is shown in Figure 34 and described in Table 25.

The interrupt line is activated whenever one of the eight interrupts is active.

An interrupt source can be flagged only if enabled in the IER register. Clearing of interrupts is described in Table 4.

Figure 34. Interrupt Identification Register IrDA Mode (IIR-IrDA mode)

31							16
Reserved							
R-0							

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
EOF_IT	LINE_STS_IT	TX_STATUS_IT	STS_FIFO_IT	RX_OE_IT	RX_FIFO_LAST_BYTE_IT	THR_IT	RHR_IT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

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

Table 25. Interrupt Identification Register IrDA Mode (IIR-IrDA mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	EOF_IT		IrDA mode only.
		0	Received EOF interrupt inactive.
		1	Received EOF interrupt active.
6	LINE_STS_IT		IrDA mode only.
		0	Receiver line status interrupt inactive.
		1	Receiver line status interrupt active.
5	TX_STATUS_IT		IrDA and CIR mode only.
		0	TX status interrupt inactive.
		1	TX status interrupt active.
4	STS_FIFO_IT		IrDA mode only.
		0	Status FIFO trigger level interrupt inactive.
		1	Status FIFO trigger level interrupt active.
3	RX_OE_IT		IrDA and CIR mode only.
		0	RX overrun interrupt inactive.
		1	RX overrun interrupt active.
2	RX_FIFO_LAST_BYTE_IT		IrDA mode only.
		0	Last byte of frame in RX FIFO interrupt inactive.
		1	Last byte of frame in RX FIFO interrupt active.
1	THR_IT		IrDA and CIR mode only.
		0	THR interrupt inactive.
		1	THR interrupt active.

Table 25. Interrupt Identification Register IrDA Mode (IIR-IrDA mode) (continued)

Bit	Field	Value	Description
0	RHR_IT		IrDA and CIR mode only.
		0	RHR interrupt inactive.
		1	RHR interrupt active.

3.6.3 Interrupt Identification Register CIR Mode (IIR-CIR mode)

The Interrupt Identification Register CIR Mode (IIR-CIR mode) is shown in Figure 35 and described in Table 26.

The interrupt line is activated whenever one of the six interrupts is active.

An interrupt source can be flagged only if enabled in the IER register. Clearing of interrupts is described in Table 5.

Figure 35. Interrupt Identification Register CIR Mode (IIR-CIR mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
Reserved		TX_STATUS_IT	Reserved	RX_OE_IT	RX_STOP_IT	THR_IT	RHR_IT
R-0		R-0	R-0	R-0	R-0	R-0	R-0

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

Table 26. Interrupt Identification Register CIR Mode (IIR-CIR mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-6	Reserved	0	Not used in CIR Mode
5	TX_STATUS_IT		IrDA and CIR mode only.
		0	TX status interrupt inactive.
		1	TX status interrupt active.
4	Reserved	0	Not used in CIR Mode
3	RX_OE_IT		IrDA and CIR mode only.
		0	RX overrun interrupt inactive.
		1	RX overrun interrupt active.
2	RX_STOP_IT		CIR mode only.
		0	Receive stop interrupt is inactive.
		1	Receive stop interrupt is active.
1	THR_IT		IrDA and CIR mode only.
		0	THR interrupt inactive.
		1	THR interrupt active.
0	RHR_IT		IrDA and CIR mode only.
		0	RHR interrupt inactive.
		1	RHR interrupt active.

3.7 FIFO Control Register (FCR)

The FIFO control register (FCR) is shown in Figure 36 and described in Table 27.

Offset Address (hex): 2h × S and LCR ≠ BFh (and ENHANCED_EN = 1 for TX_FIFO_TRIG) and write.

Below are FCR restrictions:

- Bits 4 and 5 can only be written to when ENHANCED_EN = 1.
- Bits 0 to 3 can be changed only when the baud clock is not running (DLL and DLH cleared to 0).
- See Table 42 for TX_FIFO_TRIG setting restriction when TX_TRIG_GRANU1 = 1.
- See Table 43 for RX_FIFO_TRIG setting restriction when RX_TRIG_GRANU1 = 1.

Figure 36. FIFO Control Register (FCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
RX_FIFO_TRIG		TX_FIFO_TRIG		DMA_MODE	TX_FIFO_CLEAR	RX_FIFO_CLEAR	FIFO_EN
W-0		W-0		W-0	W-0	W-0	W-0

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

Table 27. FIFO Control Register (FCR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-6	RX_FIFO_TRIG	0-3h	Set trigger level for RX FIFO.
			If RX_TRIG_GRANU1 = 0 and RX_FIFO_TRIG_DMA ≠ 0000, RX_FIFO_TRIG is not considered. If RX_TRIG_GRANU1 = 1, RX_FIFO_TRIG is 2 LSB of the trigger level (1-63 on 6 bits) with the granularity 1.
			If RX_TRIG_GRANU1 = 0 and RX_FIFO_TRIG_DMA = 0000:
		0	8 characters.
		1h	16 characters.
		2h	56 characters.
		3h	60 characters.
5-4	TX_FIFO_TRIG	0-3h	Set trigger level for TX FIFO.
			If TX_TRIG_GRANU1 = 0 and TX_FIFO_TRIG_DMA ≠ 0000, TX_FIFO_TRIG is not considered. If TX_TRIG_GRANU1 = 1, TX_FIFO_TRIG is 2 LSB of the trigger level (1-63 on 6 bits) with the granularity 1.
			If RX_TRIG_GRANU1 = 0 and RX_FIFO_TRIG_DMA = 0000:
		0	8 spaces.
		1h	16 spaces.
		2h	32 spaces.
		3h	56 spaces.
3	DMA_MODE		This register is considered if DMA_MODE_CTL = 0.
		0	DMA_MODE 0 (No DMA).
		1	DMA_MODE 1 (UART_nDMA_REQ[0] in TX, UART_nDMA_REQ[1] in RX).

Table 27. FIFO Control Register (FCR) (continued)

Bit	Field	Value	Description
2	TX_FIFO_CLEAR		Clears transmit FIFO.
		0	No change.
		1	Clears the transmit FIFO and resets its counter logic to zero. Returns to zero after clearing FIFO.
1	RX_FIFO_CLEAR		Clears receive FIFO.
		0	No change.
		1	Clears the receive FIFO and resets its counter logic to zero. Returns to zero after clearing FIFO.
0	FIFO_EN		Enables FIFO.
		0	Disables the transmit and receive FIFOs. The transmit and receive holding registers are one byte FIFOs.
		1	Enables the transmit and receive FIFOs. The transmit and receive holding registers are 64-bytes FIFOs.

3.8 Enhanced Feature Register (EFR)

Offset Address (hex): 2h × S and LCR = BFh.

The enhanced feature register (EFR) is shown in Figure 37 and described in Table 28.

This register enables or disables enhanced features. Most of the enhanced functions only apply to UART modes, but ENHANCED_EN enables write accesses to TX_FIFO_TRIG, the TX trigger level, which is also used in IrDA modes.

In all IR modes EFR[1:0] controls address checking, as described on Section 2.4.2.5.

Figure 37. Enhanced Feature Register (EFR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
AUTO_CTS_EN	AUTO_RTS_EN	SPECIAL_CHAR_DETECT	ENHANCED_EN	SW_FLOW_CONTROL			
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0			

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

Table 28. Enhanced Feature Register (EFR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	AUTO_CTS_EN		Auto-$\overline{CTS}$ enable bit.
		0	Normal operation.
		1	Auto-$\overline{CTS}$ flow control is enabled. Transmission is halted when the $\overline{CTS}$ pin is high (inactive).
6	AUTO_RTS_EN		Auto-$\overline{RTS}$ enable bit.
		0	Normal operation.
		1	Auto- $\overline{RTS}$ flow control is enabled. $\overline{RTS}$ pin goes high (inactive) when the receiver FIFO HALT trigger level, RX_FIFO_TRIG_HALT, is reached, and goes low (active) when the receiver FIFO RESTORE transmission trigger level is reached.
5	SPECIAL_CHAR_DETECT		Special character detect enable bit.
		0	Normal operation.
		1	Special character detect enable. Received data is compared with XOFF2 data. If a match occurs the received data is transferred to RX FIFO and IIR bit 4 is set to 1 to indicate a special character has been detected.
4	ENHANCED_EN		Enhanced functions write enable bit.
		0	Disables writing to IER bits 4-7, TX_FIFO_TRIG bit in FCR, and MCR bits 5-7.
		1	Enables writing to IER bits 4-7, TX_FIFO_TRIG bit in FCR, and MCR bits 5-7.
3-0	SW_FLOW_CONTROL	0-Fh	Defines software flow control. Combinations of Software flow control can be selected by programming bits 3-0. See Table 29.

Table 29. Software Flow Control Options

Bit 3	Bit 2	Bit 1	Bit 0	Tx, Rx software flow controls
0	0	X	X	No transmit flow control.
1	0	X	X	Transmit XON1, XOFF1.
0	1	X	X	Transmit XON2, XOFF2.
1	1	X	X	Transmit XON1, XON2: XOFF1, XOFF2(1).
X	X	0	0	No receive flow control.
X	X	1	0	Receiver compares XON1, XOFF1.
X	X	0	1	Receiver compares XON2, XOFF2.
X	X	1	1	Receiver compares XON1, XON2: XOFF1, XOFF2(1).

(1) XON1 and XON2 should be set to different values if the software flow control is enabled.

3.9 Line Control Register (LCR)

The line control register (LCR) is shown in Figure 38 and described in Table 30.

Offset Address (hex): 3h × S.

LCR[6:0] define parameters of the transmission and reception. As soon as BREAK_EN is set to 1, the TX line is forced to 0 and remains in this state as long as BREAK_EN = 1.

Figure 38. Line Control Register (LCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
DIV_EN	BREAK_EN	PARITY_TYPE2	PARITY_TYPE1	PARITY_EN	NB_STOP	CHAR_LENGTH	
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	

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

Table 30. Line Control Register (LCR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	DIV_EN		Allows to access to DLL, DLH and other registers (refer to the registers mapping).
		0	Normal operating condition.
		1	Enables divisor latch.
6	BREAK_EN		Break control bit.
		0	Normal operating condition.
		1	Forces the transmitter output to go low to alert the communication terminal.
5	PARITY_TYPE2	0-1	Selects the forced parity format (if PARITY_EN = 1). If PARITY_TYPE2 = 1 and PARITY_TYPE1 = 0, the parity bit is forced to 1 in the transmitted and received data. If PARITY_TYPE2 = 1 and PARITY_TYPE1 = 1, the parity bit is forced to 0 in the transmitted and received data.
4	PARITY_TYPE1		Selects odd/even parity format.
		0	Odd parity is generated (if PARITY_EN = 1).
		1	Even parity is generated (if PARITY_EN = 1). When PARITY_EN = 0 and MDR1.MODE_SELECT = 6h, UART enters CIR demodulator bypass mode.
3	PARITY_EN		Enable parity bit generation.
		0	No parity. When PARITY_TYPE1 = 1 and MDR1.MODE_SELECT = 6h, UART enters CIR demodulator bypass mode.
		1	A parity bit is generated during transmission and the receiver checks for received parity.
2	NB_STOP		Specifies number of stop bits.
		0	1 stop bits (word length = 5, 6, 7, 8)
		1	1.5 stop bits (word length = 5). 2 stop bits (word length = 6, 7, 8).
1-0	CHAR_LENGTH	0-3h	Specifies the word length to be transmitted or received.
		0	5 bits
		1h	6 bits
		2h	7 bits
		3h	8 bits

3.10 Modem Control Register (MCR)

The modem control register (MCR) is shown in Figure 39 and described in Table 31.

Offset Address (hex): 4h × S and LCR ≠ BFh (and ENHANCED_EN = 1 for MCR[7:5]).

MCR[3:0] controls the interface with the modem, data set or peripheral device that is emulating the modem.

Figure 39. Modem Control Register (MCR)

31	30	29	28	27	26	25	16
Reserved							
R-0							

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
Reserved	TCR_TLR	XON_EN	LOOPBACK_EN	CD_STS_CH	RI_STS_CH	RTS	DTR
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

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

Table 31. Modem Control Register (MCR)

Bit	Field	Value	Description
31-7	Reserved	0	Reserved.
6	TCR_TLR		Enables access to TCR and TLR. Can be written only when ENHANCED_EN = 1.
		0	No action.
		1	Enables access to the TCR and TLR registers.
5	XON_EN		Enables XON any function. Can be written only when ENHANCED_EN = 1.
		0	Disable XON any function.
		1	Enable XON any function.
4	LOOPBACK_EN		Enables internal local loopback mode. In this mode the MCR[3:0] signals are looped back into MSR[7:4]. The transmit output is looped back to the receive input internally.
		0	Normal operating mode.
		1	Enable local loopback mode (internal).
3	CD_STS_CH		Loopback forces $\overline{DCD}$ input.
		0	In loopback forces $\overline{DCD}$ input high and IRQ outputs to inactive state.
		1	In loopback forces $\overline{DCD}$ input low and IRQ outputs to inactive state.
2	RI_STS_CH		Loopback forces $\overline{RI}$ input high/ low.
		0	In loopback forces $\overline{RI}$ input high.
		1	In loopback forces $\overline{RI}$ input low.
1	RTS		In loop back controls MSR[4]. If auto-RTS is enabled the $\overline{RTS}$ output is controlled by hardware flow control.
		0	Force $\overline{RTS}$ output to inactive (high).
		1	Force $\overline{RTS}$ output to active (low).
0	DTR		Forces $\overline{DTR}$ output.
		0	Force $\overline{DTR}$ output to inactive (high).
		1	Force $\overline{DTR}$ output to active (low).

3.11 XON1 Character Register (XON1)

Offset Address (hex): 04h × S and LCR=BFh.

The XON1 character register (XON1) is shown in Figure 40 and described in Table 32.

In these cases the XON1 and XON2 characters or the XOFF1 and XOFF2 characters must be transmitted/received sequentially with XON1/XOFF1 followed by XON2/XOFF2.

Figure 40. XON1 Character Register (XON1)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		XON_WORD1	
R-0		R/W-0	

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

Table 32. XON1 Character Register (XON1)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	XON_WORD1	0-FFh	Used to store the 8-bit XON1 character in UART modes and ADDR1 address 1 for IrDA modes.

3.12 Line Status Register (LSR)

Offset Address (hex): 5h × S and LCR ≠ BFh and read.

3.12.1 Line Status Register UART Mode (LSR-UART Mode)

The line status register UART mode (LSR-UART mode) is shown in Figure 41 and described in Table 33.

When the LSR is read, LSR[4:2] reflect the error bits [BI, FE, PE] of the character at the top of the RX FIFO (next character to be read). Therefore, reading the LSR and then reading the RHR identifies errors in a character.

Reading RHR updates [BI, FE, PE] (see Table 3 UART Mode Interrupts).

RX_FIFO_STS is set when there is an error anywhere in the RX FIFO and is cleared only when there are no more errors remaining in the RX FIFO.

Reading the LSR does not cause an increment of the RX FIFO read pointer. The RX FIFO read pointer is incremented by reading the RHR. Reading LSR clears [OE] if set (see table for UART Mode Interrupts).

Figure 41. Line Status Register UART Mode (LSR-UART Mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
RX_FIFO_STS	TX_SR_E	TX_FIFO_E	RX_BI	RX_FE	RX_PE	RX_OE	RX_FIFO_E
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

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

Table 33. Line Status Register UART Mode (LSR-UART Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	RX_FIFO_STS		UART mode only.
		0	Normal operation.
		1	At least one parity error, framing error or break indication in the RX FIFO. RX_FIFO_STS is cleared when no more errors are present in the RX FIFO.
6	TX_SR_E		UART mode only.
		0	Transmitter hold (TX FIFO) and shift registers are not empty.
		1	Transmitter hold (TX FIFO) and shift registers are empty.
5	TX_FIFO_E		UART mode only.
		0	Transmit hold register (TX FIFO) is not empty.
		1	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily completed.
4	RX_BI		UART mode only.
		0	No break condition.
		1	A break was detected while the data being read from the RX FIFO was being received. (that is, RX input was low for one character + 1 bit time frame).

Table 33. Line Status Register UART Mode (LSR-UART Mode) (continued)

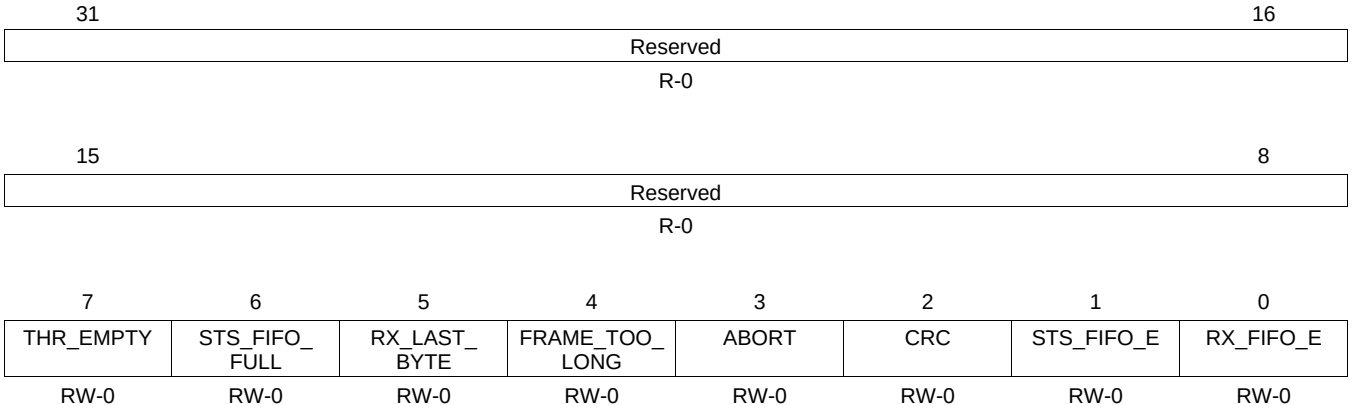
Bit	Field	Value	Description
3	RX_FE		UART mode only.
		0	No framing error in data being read from RX FIFO.
		1	Framing error occurred in data being read from RX FIFO.(received data did not have a valid stop bit).
2	RX_PE		UART mode only.
		0	No parity error in data being read from RX FIFO.
		1	Parity error in data being read from RX FIFO.
1	RX_OE		UART mode only.
		0	No overrun error.
		1	Overrun error has occurred. Set when the character held in the receive shift register is not transferred to the RX FIFO. This case can occurs only when receive FIFO is full.
0	RX_FIFO_E		UART mode only.
		0	No data in the receive FIFO.
		1	At least one data character in the RX FIFO.

3.12.2 Line Status Register IrDA Mode (LSR-IrDA Mode)

The line status register IrDA mode (LSR-IrDA Mode) is shown in Figure 42 and described in Table 34.

When the LSR is read, LSR[4:2] reflect the error bits [FL, CRC, ABORT] of the frame at the top of the STATUS FIFO (next frame status to be read). See Table 4 IrDA Mode Interrupts.

Figure 42. Line Status Register IrDA Mode (LSR-IrDA Mode)



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

Table 34. Line Status Register IrDA Mode (LSR-IrDA Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	THR_EMPTY		IrDA mode only.
		0	Transmit holding register (TX FIFO) is not empty.
		1	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily completed.
6	STS_FIFO_FULL		IrDA mode only.
		0	Status FIFO not full.
		1	Status FIFO full.
5	RX_LAST_BYTE		IrDA mode only.
		0	The RX FIFO (RHR) does not contain the last byte of the frame to be read.
		1	The RX FIFO (RHR) contains the last byte of the frame to be read. This bit is only set when the last byte of a frame is available to be read. It is used to determine the frame boundary. It is cleared on a single read of the LSR register. (1)
4	FRAME_TOO_LONG		IrDA mode only.
		0	No frame-too-long error in frame.
		1	Frame-too-long error in the frame at the top of the STATUS FIFO, [next character to be read]. This bit is set to 1 when a frame exceeding the maximum length (set by RXFLH and RXFLL registers) has been received. When this error is detected, current frame reception is terminated. Reception is stopped until the next START flag is detected.
3	ABORT		IrDA mode only.
		0	No abort pattern error in frame.
		1	Abort pattern is received. • SIR & MIR: Abort pattern. • FIR: Illegal symbol.
2	CRC		IrDA mode only.
		0	No CRC error in frame.
		1	CRC error in the frame at the top of the STATUS FIFO (next character to be read).

(1) Legacy SW design (1510 and equivalent silicon) must take into account the above changes. The 1510 documentation will contain erratum that describes the flow of operations to rectify the 1510 and equivalent silicon as it exists for those versions of silicon.

Table 34. Line Status Register IrDA Mode (LSR-IrDA Mode) (continued)

Bit	Field	Value	Description
1	STS_FIFO_E		IrDA mode only.
		0	Status FIFO not empty.
		1	Status FIFO empty.
0	RX_FIFO_E		IrDA mode only.
		0	Transmit holding register (TX FIFO) is not empty.
		1	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily completed.

3.12.3 Line Status Register CIR Mode (LSR-CIR Mode)

The line status register CIR mode (LSR-CIR Mode) is shown in Figure 43 and described in Table 35.

Figure 43. Line Status Register CIR Mode (LSR-CIR Mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
THR_EMPTY	Reserved	RX_STOP	Reserved				RX_FIFO_E
RW-0	R-0	RW-0	R-0				RW-0

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

Table 35. Line Status Register CIR Mode (LSR-CIR Mode)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	THR_EMPTY		CIR mode only.
		0	Transmit holding register (TX FIFO) is not empty.
		1	Transmit hold register (TX FIFO) is empty. The transmission is not necessarily completed.
6	Reserved	0	Reserved. Not used in CIR Mode
5	RX_STOP		CIR mode only.
			The RX_STOP is generated based on the value set in the BOF Length register (EBLR). It is cleared on a single read of the LSR register.
		0	Reception is on going or waiting for a new frame.
		1	Reception is completed.
4-1	Reserved	0	Reserved. Not used in CIR Mode
0	RX_FIFO_E		CIR mode only.
		0	At least one data character in the RX FIFO.
		1	No data in the receive FIFO.

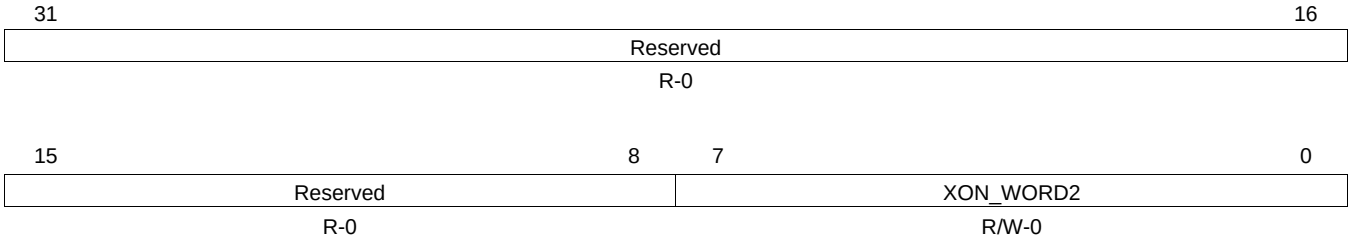
3.13 XON2 Character Register (XON2)

Offset Address (hex): 5h × S and LCR=BFh.

The XON2 character register (XON2) is shown in Figure 44 and described in Table 36.

In these cases the XON1 and XON2 characters or the XOFF1 and XOFF2 characters must be transmitted/received sequentially with XON1/XOFF1 followed by XON2/XOFF2.

Figure 44. XON2 Character Register (XON2)



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

Table 36. XON2 Character Register (XON2)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	XON_WORD2	0-FFh	Used to store the 8-bit XON2 character in UART modes and ADDR2 address 2 for IrDA modes.

3.14 Modem Status Register (MSR)

The modem status register (MSR) is shown in Figure 45 and described in Table 37.

Offset Address (hex): 6h × S and LCR ≠ BFh and (ENHANCED_EN = 0 or TCR_TLR = 0) and read.

This register provides information about the current state of the control lines from the modem, data set or peripheral device to the LH. It also indicates when a control input from the modem changes state.

Figure 45. Modem Status Register (MSR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
NCD_STS	NRI_STS	NDSR_STS	NCTS_STS	DCD_STS	RI_STS	DSR_STS	CTS_STS
R-X	R-X	R-X	R-X	R-0	R-0	R-0	R-0

LEGEND: R = Read only; X = Unknown; -*n* = value after reset.

Table 37. Modem Status Register (MSR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	NCD_STS	0-1	This bit is the complement of the $\overline{DCD}$ input. In loop-back mode it is equivalent to CD_STS_CH bit in MCR.
6	NRI_STS	0-1	This bit is the complement of the $\overline{RI}$ input. In loop-back mode it is equivalent to RI_STS_CH bit in MCR.
5	NDSR_STS	0-1	This bit is the complement of the $\overline{DSR}$ input. In loop-back mode, it is equivalent to $\overline{DTR}$.
4	NCTS_STS	0-1	This bit is the complement of the $\overline{CTS}$ input. In loop-back mode it is equivalent to $\overline{RTS}$.
3	DCD_STS	0-1	Indicates that $\overline{DCD}$ input (or CD_STS_CH bit in MCR in loop back) has changed. Cleared on a read.
2	RI_STS	0-1	Indicates that $\overline{RI}$ input (or RI_STS_CH bit in MCR in loop back) has changed state from low to high. Cleared on a read.
1	DSR_STS	0-1	Indicates that $\overline{DSR}$ input (or $\overline{DTR}$ in loop back) has changed state. Cleared on a read.
0	CTS_STS	0-1	Indicates that $\overline{CTS}$ input (or $\overline{RTS}$ in loop back) has changed state. Cleared on a read.

3.15 Transmission Control Register (TCR)

Offset Address (hex): 6h × S and ENHANCED_EN = 1 and TCR_TLR = 1.

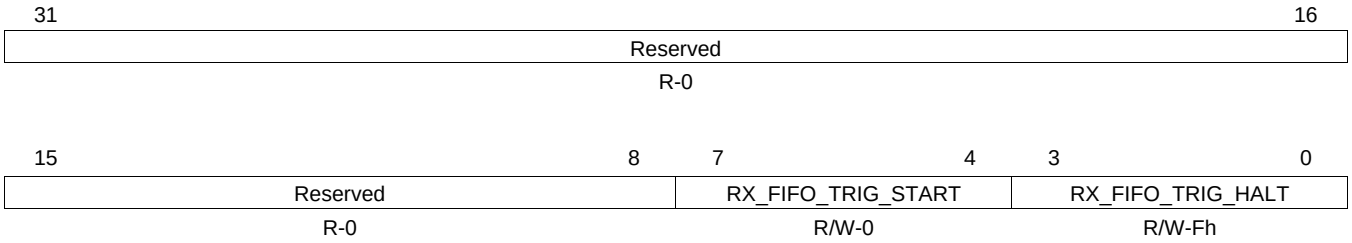
The transmission control register (TCR) is shown in Figure 46 and described in Table 38.

This register is used to store the receive FIFO threshold levels to start/stop transmission during hardware/software flow control.

Trigger levels from 0-60 bytes are available with a granularity of four. (Trigger level = 4 × [4-bit register value]). The programmer must ensure that RX_FIFO_TRIG_HALT > RX_FIFO_TRIG_START whenever auto-RTS or software flow control is enabled to avoid a mis-operation of the device.

In FIFO interrupt mode with flow control, programmer has also to ensure that trigger level to HALT transmission is greater or equal to Receive FIFO trigger level (either RX_FIFO_TRIG_DMA or RX_FIFO_TRIG), otherwise FIFO operation stalls. In FIFO DMA mode with flow control, this concept does not exist because DMA request is sent each time a byte is received.

Figure 46. Transmission Control Register (TCR)



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

Table 38. Transmission Control Register (TCR)

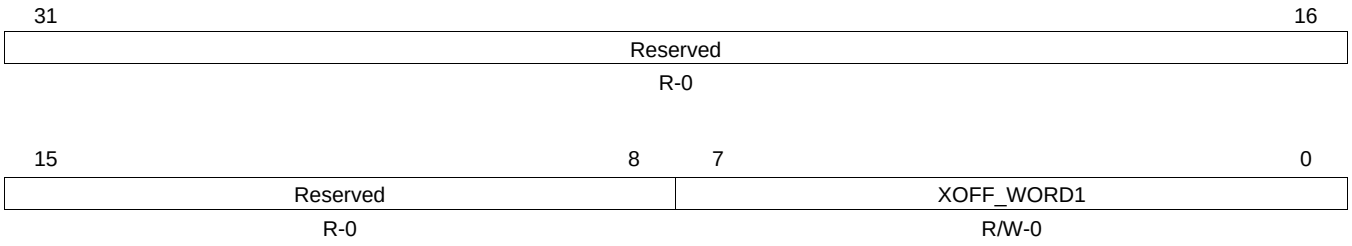
Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-4	RX_FIFO_TRIG_START	0-Fh	RX FIFO trigger level to RESTORE transmission (0-60).
3-0	RX_FIFO_TRIG_HALT	0-Fh	RX FIFO trigger level to HALT transmission (0-60).

3.16 XOFF1 Character Register (XOFF1)

Offset Address (hex): 6h × S and LCR=BFh and (ENHANCED_EN = 0 or TCR_TLR = 0).

The XOFF1 character register (XOFF1) is shown in Figure 47 and described in Table 39.

Figure 47. XOFF1 Character Register (XOFF1)



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

Table 39. XOFF1 Character Register (XOFF1)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	XOFF_WORD1	0-FFh	Used to store the 8-bit XOFF1 character used in UART modes.

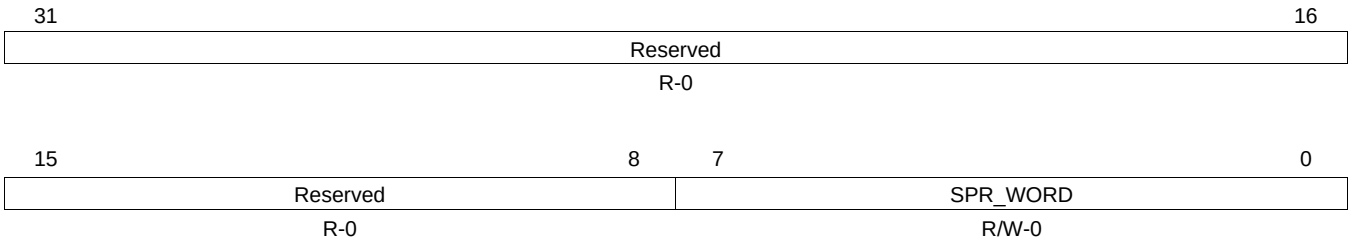
3.17 Scratchpad Register (SPR)

Offset Address (hex): 7h × S and LCR ≠ BFh and (ENHANCED_EN = 0 or TCR_TLR = 0)

The scratchpad register (SPR) is shown in Figure 48 and described in Table 40.

This read/write register does not control the module in anyway. It is intended as a scratchpad register to be used by the programmer to hold temporary data.

Figure 48. Scratchpad Register (SPR)



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

Table 40. Scratchpad Register (SPR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	SPR_WORD	0-FFh	Scratchpad register.

3.18 Trigger Level Register (TLR)

Offset Address (hex): 7H × S and ENHANCED_EN = 1 and TCR_TLR = 1.

The trigger level register (TLR) is shown in Figure 49 and described in Table 41.

This register is used to store the programmable transmit and receive FIFO trigger levels used for DMA and IRQ generation.

Figure 49. Trigger Level Register (TLR)

31	16
Reserved	
R-0	

15	8	7	4	3	0
Reserved		RX_FIFO_TRIG_DMA		TX_FIFO_TRIG_DMA	
R-0		R/W-0		R/W-Fh	

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

Table 41. Trigger Level Register (TLR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-4	RX_FIFO_TRIG_DMA	0-Fh	Receive FIFO trigger level.
3-0	TX_FIFO_TRIG_DMA	0-Fh	Transmit FIFO trigger level.

Table 42 and Table 43 summarize the different ways that can be used to set the trigger levels for respectively the transmit FIFO and the receive FIFO.

Table 42. TX FIFO Trigger Level Setting Summary

TX_TRIG_GRANU1	TX_FIFO_TRIG_DMA	TX FIFO Trigger Level
0	= 0000	Defined by TX_FIFO_TRIG (either 8, 16, 32, 56 spaces).
0	≠ 0000	Defined by TX_FIFO_TRIG_DMA (from 4 to 60 spaces with a granularity of 4 spaces).
1	Value	Defined by the concatenated value of TX_FIFO_TRIG_DMA and TX_FIFO_TRIG bit in FCR (from 1 to 63 spaces with a granularity of 1 space). The combination of TX_FIFO_TRIG_DMA = 0000 and TX_FIFO_TRIG bit in FCR = 00 (all zeros) is not supported (minimum 1 space required). All zeros will result in unpredictable behavior.

Table 43. RX FIFO Trigger Level Setting Summary

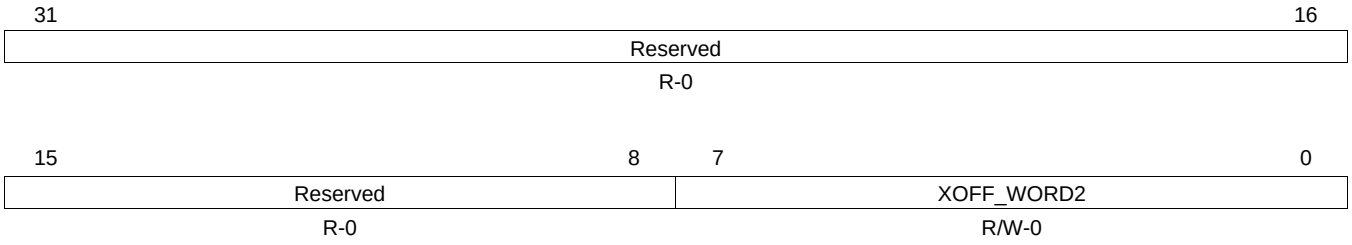
RX_TRIG_GRANU1	RX_FIFO_TRIG_DMA	RX FIFO Trigger Level
0	= 0000	Defined by RX_FIFO_TRIG (either 8, 16, 56, 60 characters).
0	≠ 0000	Defined by RX_FIFO_TRIG_DMA (from 4 to 60 characters with a granularity of 4 characters).
1	Value	Defined by the concatenated value of RX_FIFO_TRIG_DMA and RX_FIFO_TRIG bit in FCR (from 1 to 63 characters with a granularity of 1 character). The combination of RX_FIFO_TRIG_DMA = 0000 and RX_FIFO_TRIG bit in FCR = 00 (all zeros) is not supported (minimum 1 character required). All zeros will result in unpredictable behavior.

3.19 XOFF2 Character Register (XOFF2)

Offset Address (hex): 7h × S and LCR=BFh and (ENHANCED_EN = 0 or TCR_TLR = 0).

The XOFF2 character register (XOFF2) is shown in Figure 50 and described in Table 44.

Figure 50. XOFF2 Character Register (XOFF2)



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

Table 44. XOFF2 Character Register (XOFF2)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	XOFF_WORD2	0-FFh	Used to store the 8-bit XOFF2 character used in UART modes.

3.20 Mode Definition Register 1 (MDR1)

Offset Address (hex): 8h × S.

The mode definition register 1 (MDR1) is shown in Figure 51 and described in Table 45.

The mode of operation can be programmed by writing to MODE_SELECT and therefore the MDR1 must be programmed on start-up after configuration of the configuration registers (DLL, DLH, LCR). The value of MODE_SELECT must not be changed again during normal operation.

If the module is disabled by setting the MODE_SELECT field to 7h interrupt requests can still be generated unless disabled through the Interrupt Enable Register (IER). In this case UART mode interrupts are visible. Reading the Interrupt Identification Register (IIR) will show UART mode interrupt flags.

Figure 51. Mode Definition Register 1 (MDR1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
FRAME_END_MODE	SIP_MODE	SCT	SET_TXIR	IR_SLEEP	MODE_SELECT		
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-7h		

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

Table 45. Mode Definition Register 1 (MDR1)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	FRAME_END_MODE		IrDA mode only.
		0	Frame-length method.
		1	Set EOT bit method.
6	SIP_MODE		MIR/FIR modes only.
		0	Manual SIP mode: SIP is generated with the control of SEND_SIP.
		1	Automatic SIP mode: SIP is generated after each transmission.
5	SCT		Store and control the transmission. Before starting any transmission, there must be no reception on going.
		0	Starts the Infrared transmission as soon as a value is written to THR.
		1	Starts the Infrared transmission with the control of SCTX_EN.
4	SET_TXIR		Used to configure the Infrared transceiver.
		0	No action.
		1	IRTX pin output is forced high.
3	IR_SLEEP		IrDA/CIR sleep mode.
		0	IrDA/CIR sleep mode disabled.
		1	IrDA/CIR sleep mode enabled.

Table 45. Mode Definition Register 1 (MDR1) (continued)

Bit	Field	Value	Description
2-0	MODE_SELECT	0-7h	Mode select.
		0	UART 16× mode.
		1h	SIR mode.
		2h	UART 16× auto-baud.
		3h	UART 13× mode.
		4h	MIR mode.
		5h	FIR mode.
		6h	CIR mode.
		7h	Disable (default state).

3.21 Mode Definition Register 2 (MDR2)

Offset Address (hex): 9h × S.

The mode definition register 2 (MDR2) is shown in Figure 52 and described in Table 46.

IR-IrDA and IR-CIR modes only.

IRTX_UNDERRUN describes the status of the interrupt in TX_STATUS_IT bit in IIR register. The IRTX_UNDERRUN bit should be read after an TX_STATUS_IT interrupt has occurred. The bits [2:1] of this register sets the trigger level for the frame status FIFO (8 entries) and must be programmed before the mode is programmed in MODE_SELECT.

The IRRXINVERT gives the flexibility to invert the RX pin inside the UART module to ensure that the protocol at the input of the transceiver module has the same polarity at module level. By default, the RX pin is inverted because most of transceiver invert the IR receive pin.

Figure 52. Mode Definition Register 2 (MDR2)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
Reserved	IRRXINVERT	CIR_PULSE_MODE		UART_PULSE	STS_FIFO_TRIG		IRTX_UNDERRUN
R-0	R/W-0	R/W-0		R/W-0	R/W-0		R-0

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

Table 46. Mode Definition Register 2 (MDR2)

Bit	Field	Value	Description
31-7	Reserved	0	Reserved.
6	IRRXINVERT		IrDA and CIR MODE only. Invert RX pin inside the module before the voting or sampling system logic of the infra red block. This will not affect the RX path in UART Modem modes.
		0	Inversion is performed.
		1	No inversion is performed.
5-4	CIR_PULSE_MODE	0-3h	CIR Pulse modulation definition. It defines high level of the pulse width associated with a digit:
		0	Pulse width of 3 from 12 cycles.
		1h	Pulse width of 4 from 12 cycles.
		2h	Pulse width of 5 from 12 cycles.
		3h	Pulse width of 6 from 12 cycles.
3	UART_PULSE		UART mode only. Used to allow pulse shaping in UART mode.
		0	Normal UART mode.
		1	UART mode with a pulse shaping.
2-1	STS_FIFO_TRIG	0-3h	Only for IR-IrDA mode. Frame Status FIFO Threshold select:
		0	1 entry.
		1h	4 entries.
		2h	7 entries.
		3h	8 entries.

Table 46. Mode Definition Register 2 (MDR2) (continued)

Bit	Field	Value	Description
0	IRTX_UNDERRUN		IrDA Transmission status interrupt. When the TX_STATUS_IT bit in IIR register interrupt occurs, the meaning of the interrupt is:
		0	The last bit of the frame has been transmitted successfully without error.
		1	An underrun has occurred. The last bit of the frame has been transmitted but with an underrun error present. The bit is reset to 0 when the RESUME register is read.

3.22 Status FIFO Line Status Register (SFLSR)

Offset Address (hex): 0Ah × S and read.

The status FIFO line status register (SFLSR) is shown in Figure 53 and described in Table 47.

Reading this register effectively reads frame status information from the status FIFO (this register doesn't physically exist). Reading this register will increment the status FIFO read pointer (SFREGL and SFREGH must be read first).

Figure 53. Status FIFO Line Status Register (SFLSR)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	5	4	3	2	1	0
Reserved		OE_ERROR	FRAME_TOO_LONG_ERROR	ABORT_DETECT	CRC_ERROR	Reserved
R-0		R-X	R-X	R-X	R-X	R-0

LEGEND: R = Read only; X-Unknown; -*n* = value after reset.

Table 47. Status FIFO Line Status Register (SFLSR)

Bit	Field	Value	Description
31-5	Reserved	0	Reserved.
4	OE_ERROR		IrDA modes only.
		0	No error.
		1	Overrun error in RX FIFO when frame at top of RX FIFO was received.
3	FRAME_TOO_LONG_ERROR		IrDA modes only.
		0	No error.
		1	Frame-length too long error in frame at top of RX FIFO.
2	ABORT_DETECT		IrDA modes only.
		0	No error.
		1	Abort pattern detected in frame at top of RX FIFO.
1	CRC_ERROR		IrDA modes only.
		0	No error.
		1	CRC error in frame at top of RX FIFO.(1)
0	Reserved	0	Reserved.

(1) Top of RX FIFO' = Next frame to be read from RX FIFO.

3.23 Transmit Frame Length Low Register (TXFLL)

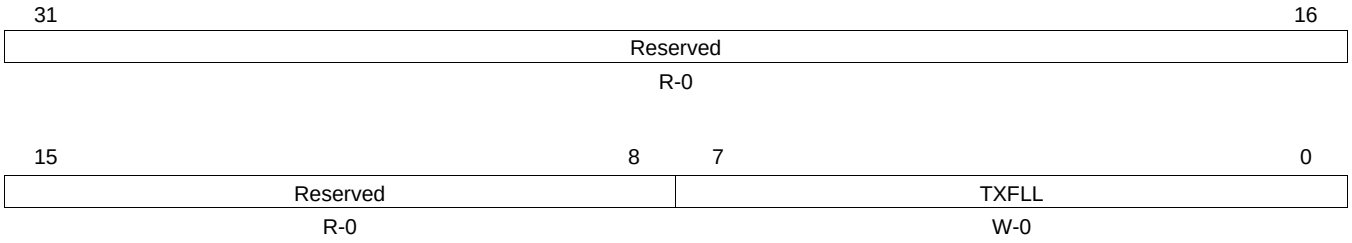
Offset Address (hex): 0Ah × S and write.

The transmit frame length low register (TXFLL) is shown in Figure 54 and described in Table 48.

IrDA modes only.

The registers TXFLL and TXFLH hold the 13-bit transmit frame length (expressed in bytes). TXFLL holds the least-significant bits and TXFLH holds the most-significant bits. The frame length value is used if the frame length method of frame closing is used.

Figure 54. Transmit Frame Length Low Register (TXFLL)



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

Table 48. Transmit Frame Length Low Register (TXFLL)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	TXFLL	0-FFh	LSB register used to specify the frame length.

3.24 RESUME Register (RESUME)

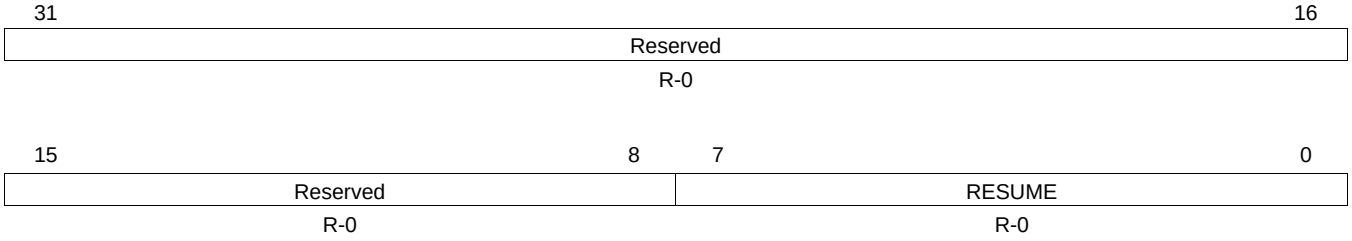
Offset Address (hex): 0Bh × S and read.

The resume register (RESUME) is shown in Figure 55 and described in Table 49.

IR-IrDA and IR-CIR modes only.

This register is used to clear internal flags, which halt transmission/reception when an underrun/overrun error occurs. Reading this register resumes the halted operation. This register does not physically exist and reads always as 0.

Figure 55. RESUME Register (RESUME)



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

Table 49. RESUME Register (RESUME)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	RESUME	0-FFh	Dummy read to restart TX or RX.

3.25 Transmit Frame Length High Register (TXFLH)

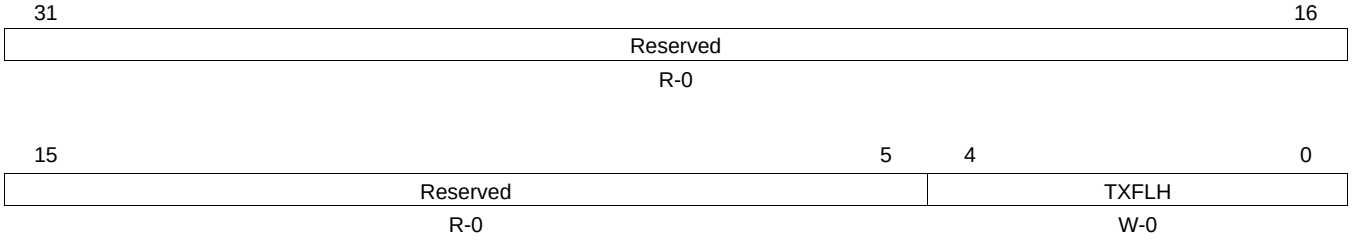
Offset Address (hex): 0Bh × S and write.

The transmit frame length high register (TXFLH) is shown in Figure 56 and described in Table 50.

IrDA modes only.

The TXFLL and TXFLH hold the 13-bit transmit frame length (expressed in bytes). TXFLL holds the least-significant bits and TXFLH holds the most-significant bits. The frame length value is used if the frame length method of frame closing is used.

Figure 56. Transmit Frame Length High Register (TXFLH)



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

Table 50. Transmit Frame Length High Register (TXFLH)

Bit	Field	Value	Description
31-5	Reserved	0	Reserved.
4-0	TXFLH	0-1Fh	MSB register used to specify the frame length.

3.26 Status FIFO Low Register (SFREGL)

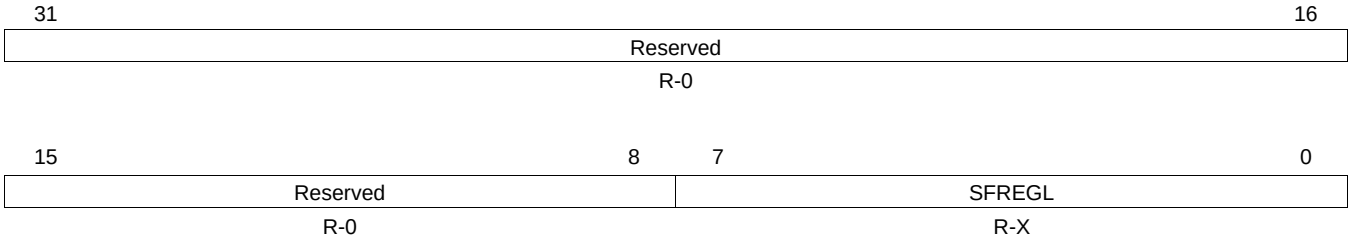
Offset Address (hex): 0Ch × S and read.

The status FIFO low register (SFREGL) is shown in Figure 57 and described inTable 51.

IrDA modes only.

The frame lengths of received frames are written into the status FIFO. This information can be read by reading the SFREGL and SFREGH registers (that is, these registers do not physically exist). The least-significant bits are read from SFREGL and the most-significant bits are read from SFREGH. Reading these registers does not alter the status FIFO read pointer. These registers should be read before the pointer is incremented by reading the SFLSR.

Figure 57. Status FIFO Low Register (SFREGL)



LEGEND: R = Read only; X = Unknown; -*n* = value after reset.

Table 51. Status FIFO Low Register (SFREGL)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	SFREGL	0-FFh	LSB part of the frame length.

3.27 Receive Frame Length Low Register (RXFLL)

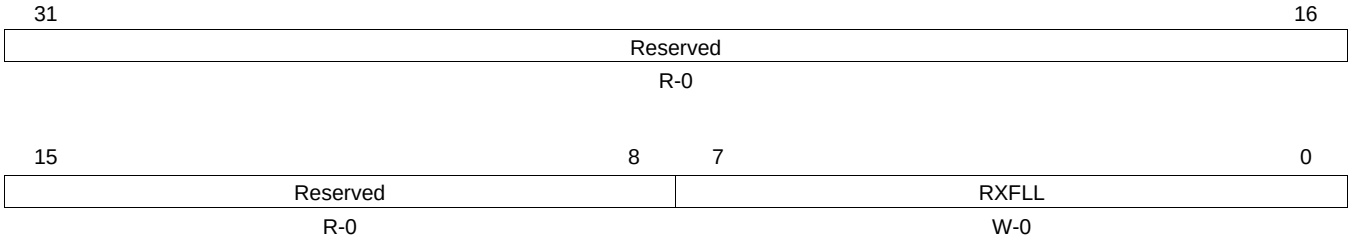
Offset Address (hex): 0Ch × S and write.

The receive frame length low register (RXFLL) is shown in Figure 58 and described in Table 52.

IrDA modes only.

The registers RXFLL and RXFLH hold the 12-bit receive maximum frame length. RXFLL holds the least-significant bits and RXFLH holds the most-significant bits. If the intended maximum receive frame length is n bytes, then program RXFLL and RXFLH to be n + 3 in SIR or MIR modes and n + 6 in FIR mode (the + 3 and + 6 are due to frame format with CRC and stop flag; there are two bytes associated with the FIR stop flag).

Figure 58. Receive Frame Length Low Register (RXFLL)



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

Table 52. Receive Frame Length Low Register (RXFLL)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	RXFLL	0-FFh	LSB register used to specify the frame length in reception.

3.28 Status FIFO High Register (SFREGH)

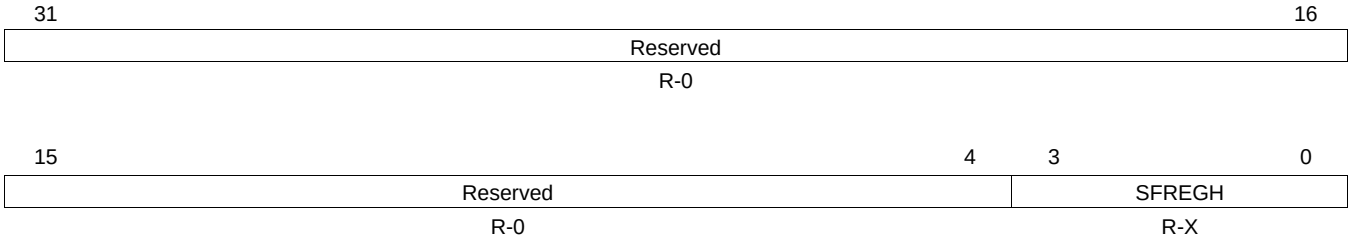
Offset Address (hex): 0Dh × S and read.

The status FIFO high register (SFREGH) is shown in Figure 59 and described in Table 53.

IrDA modes only.

The frame lengths of received frames are written into the status FIFO. This information can be read by reading the SFREGL and SFREGH registers (that is, these registers do not physically exist). The least-significant bits are read from SFREGL and the most-significant bits are read from SFREGH. Reading these registers does not alter the status FIFO read pointer. These registers should be read before the pointer is incremented by reading the SFLSR.

Figure 59. Status FIFO High Register (SFREGH)



LEGEND: R = Read only; X = Unknown; -*n* = value after reset.

Table 53. Status FIFO High Register (SFREGH)

Bit	Field	Value	Description
31-4	Reserved	0	Reserved.
3-0	SFREGH	0-Fh	MSB part of the frame length.

3.29 Receive Frame Length High Register (RXFLH)

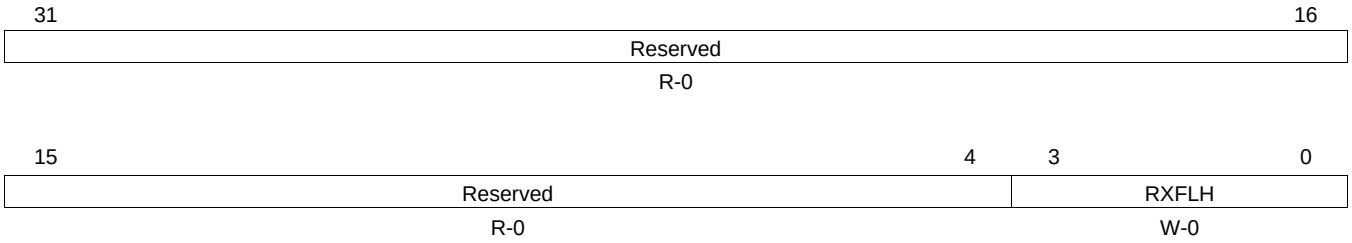
Offset Address (hex): 0Dh × S and write.

The receive frame length high register (RXFLH) is shown in Figure 60 and described in Table 54.

IrDA modes only.

The RXFLL and RXFLH hold the 12-bit receive maximum frame length. RXFLL holds the least-significant bits and RXFLH holds the most-significant bits. If the intended maximum receive frame length is n bytes, then program RXFLL and RXFLH to be n + 3 in SIR or MIR modes and n + 6 in FIR mode (the + 3 and + 6 are due to frame format with CRC and stop flag; there are two bytes associated with the FIR stop flag).

Figure 60. Receive Frame Length High Register (RXFLH)



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

Table 54. Receive Frame Length High Register (RXFLH)

Bit	Field	Value	Description
31-4	Reserved	0	Reserved.
3-0	RXFLH	0-Fh	MSB register used to specify the frame length in reception.

3.30 Begin Of Frame Control Register (BLR)

Offset Address (hex): 0Eh × S and DIV_EN = 0.

The begin of frame control register (BLR) is shown in Figure 61 and described in Table 55.

IrDA modes only.

XBOF_TYPE is used to select whether C0h or FFh start patterns are to be used, when multiple start flags are required in SIR Mode. If only one start flag is required, this will always be C0h. If *n* start flags are required, then either (*n*-1) C0h or (–1) FFh flags are sent, followed by a single C0h flag (immediately preceding the first data byte).

Figure 61. Begin Of Frame Control Register (BLR)

31	16
Reserved	
R-0	

15	8	7	6	5	0
Reserved		STS_FIFO_RESET	XBOF_TYPE	Reserved	
R-0		R/W-0	R/W-1	R-0	

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

Table 55. Begin Of Frame Control Register (BLR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	STS_FIFO_RESET	0-1	Status FIFO reset. This bit is self-clearing.
6	XBOF_TYPE		SIR XBOF select.
		0	FFh
		1	C0h
5-0	Reserved	0	Reserved.

3.31 UART Autobauding Status Register (UASR)

Offset Address (hex): 0Eh × S and DIV_EN = 1 and read.

The UART autobauding status register (UASR) is shown in Figure 62 and described in Table 56.

UART autobauding mode only.

This status register returns the speed, the number of bits by characters, the type of the parity in UART autobauding mode.

In autobauding mode the input frequency of the UART modem must be fixed to 48 MHz. Any other module clock frequency will result in incorrect baud rate recognition.

This register is used to set up transmission according to characteristics of previous reception instead of LCR, DLL and DLH registers when UART is in autobauding mode.

To reset the autobauding hardware (to start a new AT detection) or to set the UART in standard mode (no autobaud), the MODE_SELECT bit in the mode definition register 1 (MDR1) must be set to the reset value of 7h, then to the UART in autobaud mode value of 2h or the UART in standard mode value of 0.

Usage limitation:

- Only 7 and 8 bits character (5 and 6 bits not supported).
- 7 bits character with space parity not supported.
- Baud rate between 1200 and 115200 bp/s (10 possibilities).

Figure 62. UART Autobauding Status Register (UASR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	0
Reserved		PARITY_TYPE		BIT_BY_CHAR	SPEED	
R-0		R-0		R-0	R-0	

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

Table 56. UART Autobauding Status Register (UASR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-6	PARITY_TYPE	0-3h	Used to report parity type identified.
		0	No parity identified.
		1h	Parity space.
		2h	Even parity.
		3h	Odd parity.
5	BIT_BY_CHAR		Used to report 7-bit character or 8-bit character identified.
		0	7 bits character identified.
		1	8 bits character identified.

Table 56. UART Autobauding Status Register (UASR) (continued)

Bit	Field	Value	Description
4-0	SPEED	0-1Fh	Used to report identified speed.
		0	No speed identified.
		1h	115 200 bauds.
		2h	57 600 bauds.
		3h	38 400 bauds.
		4h	28 800 bauds.
		5h	19 200 bauds.
		6h	14 400 bauds.
		7h	9 600 bauds.
		8h	4 800 bauds.
		9h	2 400 bauds.
		Ah	1 200 bauds.
		Bh-1Fh	Reserved

3.32 Auxiliary Control Register (ACREG)

Offset Address (hex): 0Fh × S and DIV_EN = 0. IR-IrDA and IR-CIR modes only.

The auxiliary control register (ACREG) is shown in Figure 63 and described in Table 57.

Figure 63. Auxiliary Control Register (ACREG)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
PULSE_TYPE	SD_MOD	DIS_IR_RX	DIS_TX_UNDERRUN	SEND_SIP	SCTX_EN	ABORT_EN	EOT_EN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

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

Table 57. Auxiliary Control Register (ACREG)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	PULSE_TYPE		SIR pulse width select.
		0	3/16 of baud-rate pulse width.
		1	1.6us
6	SD_MOD		Primary output used to configure transceivers. Connected to the SD/MODE input pin of IrDA transceivers.
		0	SD pin is set to high.
		1	SD pin is set to low.
5	DIS_IR_RX		Disables RX input permanently.
		0	Normal operation (RX input automatically disabled during transmit but enabled outside of transmit operation).
		1	Disables RX input (permanent state - independent of transmit).
4	DIS_TX_UNDERRUN		Transmit long stop bits.
		0	Long stop bits cannot be transmitted, TX underrun is enabled.
		1	Long stop bits can be transmitted, TX underrun is disabled.
3	SEND_SIP		MIR/FIR Modes only. Send Serial Infrared Interaction Pulse (SIP). If this bit is set during a MIR/FIR transmission, the SIP will be send at the end of it.
			This bit automatically gets cleared at the end of the SIP transmission.
		0	No action.
		1	Send SIP pulse.
2	SCTX_EN	0-1	Store and controlled TX start. When SCT = 1 and the LH writes 1 to this bit, the TX state machine starts frame transmission. This bit is self-clearing.
1	ABORT_EN	0-1	Frame Abort. The LH can intentionally abort transmission of a frame by writing 1 to this bit. Neither the end flag nor the CRC bits are appended to the frame.
0	EOT_EN	0-1	EOT (end of transmission) bit. The LH writes 1 to this bit just before it writes the last byte to the TX FIFO in set-EOT bit frame closing method. This bit automatically gets cleared when the LH writes to the THR (TX FIFO).

3.33 Supplementary Control Register (SCR)

The supplementary control register (SCR) is shown in Figure 64 and described in Table 58.

Offset Address (hex): 11h × S.

RX_CTS_DSR_WAKE_UP_ENABLE enables the wake-up interrupt, but this interrupt is not mapped into the IIR register. Therefore, when an interrupt occurs and there is no interrupt pending in the IIR register, the RX_CTS_DSR_WAKE_UP_STS bit must be checked. To clear the wake-up interrupt, RX_CTS_DSR_WAKE_UP_ENABLE must be reset to 0.

Figure 64. Supplementary Control Register (SCR)

31	16
Reserved	
R-0	

15	8
Reserved	
R-0	

7	6	5	4	3	2	1	0
RX_TRIG_GRANU1	TX_TRIG_GRANU1	DSR_IT	RX_CTS_DSR_WAKE_UP_ENABLE	TX_EMPTY_CTL_IT	DMA_MODE_2		DMA_MODE_CTL
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0		RW-0

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

Table 58. Supplementary Control Register (SCR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7	RX_TRIG_GRANU1		Enables granularity of trigger RX level.
		0	Disables the granularity of 1 for Trigger RX level.
		1	Enables the granularity of 1 for Trigger RX level.
6	TX_TRIG_GRANU1		Enables the granularity of trigger TX level.
		0	Disables the granularity of 1 for Trigger TX level.
		1	Enables the granularity of 1 for trigger TX level.
5	DSR_IT		$\overline{\text{DSR}}$ interrupt.
		0	Disables $\overline{\text{DSR}}$ interrupt.
		1	Enables $\overline{\text{DSR}}$ interrupt.
4	RX_CTS_DSR_WAKE_UP_ENABLE		Generate interrupt.
		0	Disables the wake up interrupt and Clears RX_CTS_DSR_WAKE_UP_STS.
		1	Waits for a falling edge of pins RX, $\overline{\text{CTS}}$ or $\overline{\text{DSR}}$ to generate an interrupt.
3	TX_EMPTY_CTL_IT		THR interrupt setup.
		0	Normal mode for THR interrupt (See UART mode interrupts table).
		1	The THR interrupt is generated when Tx fifo and tx shift register are empty.
2-1	DMA_MODE_2	0-3h	Specify DMA mode.
		0	DMA mode 0 (no DMA).
		1h	DMA mode 1 (UART_nDMA_REQ[0] in TX, UART_nDMA_REQ[1] in RX).
		2h	DMA mode 2 (UART_nDMA_REQ[0] in RX).
		3h	DMA mode 3 (UART_nDMA_REQ[0] in TX).

Table 58. Supplementary Control Register (SCR) (continued)

Bit	Field	Value	Description
0	DMA_MODE_CTL		Set DMA mode.
		0	The DMA_MODE is set with DMA_MODE.
		1	The DMA_MODE is set with DMA_MODE_2.

3.34 Supplementary Status Register (SSR)

The supplementary status register (SSR) is shown in Figure 65 and described in Table 59.

Offset Address (hex): 12h × S and read.

Figure 65. Supplementary Status Register (SSR)

31	16
Reserved	
R-0	

15	2	1	0
Reserved		RX_CTS_DSR_WAKE_UP_STS	TX_FIFO_FULL
R-0		R-0	R-0

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

Table 59. Supplementary Status Register (SSR)

Bit	Field	Value	Description
31-2	Reserved	0	Reserved.
1	RX_CTS_DSR_WAKE_UP_STS		Reset only when RX_CTS_DSR_WAKE_UP_ENABLE is reset to 0.
		0	No falling edge event on RX, $\overline{CTS}$ and $\overline{DSR}$.
		1	A falling edge occurred on RX, $\overline{CTS}$ or $\overline{DSR}$.
0	TX_FIFO_FULL		Set FIFO to full.
		0	TX FIFO is not full.
		1	TX FIFO is full.

3.35 Begin Of Frame Length Register (EBLR)

Offset Address (hex): 13h × S and DIV_EN = 0.

The begin of frame length register (EBLR) is shown in Figure 66 and described in Table 60.

IR-IrDA and IR-CIR modes only.

In IR-IrDA SIR operation, this register specifies the number of BOF + xBOFs to transmit. Value set into this register must take into account the BOF character, therefore to only sent one BOF with no XBOF this register must be set to 1. To send one BOF with N XBOF this register must be set to N+1. Furthermore, the value 0 will send 1 BOF plus 255 XBOF.

In IR-IrDA MIR mode, this register specifies the number of additional start flags (MIR protocol mandates a minimum of 2 start flags).

In IR-CIR mode, this register specifies the number of consecutive zeros to be received before generating the RX_STOP interrupt (IIR[2]). All the received zeros are stored in the RX FIFO. When the register is cleared to 0, this feature is de-activated and always in reception state which can be disabled by setting the DIS_IR_RX to 1.

If the RX_STOP interrupt occurs before a byte boundary, the remaining bits of the last byte are filled with zeros and then passed into the RX FIFO.

Figure 66. Begin Of Frame Length Register (EBLR)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		EBLR	
R-0		R/W-0	

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

Table 60. Begin Of Frame Length Register (EBLR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	EBLR		IR-IrDA mode: This register allows to define up to 176 × BOFs, the maximum required by IrDA specification. IR-CIR mode: This register specifies the number of consecutive zeros to be received before generating the RX_STOP interrupt (IIR[2]).
		0-FFh	Generate RX_STOP interrupt after receiving 255 zero bits.
		0	Feature disabled.
		1	Generate RX_STOP interrupt after receiving one zero bit.

3.36 Module Version Register (MVR)

Offset Address (hex): 14h × S and read.

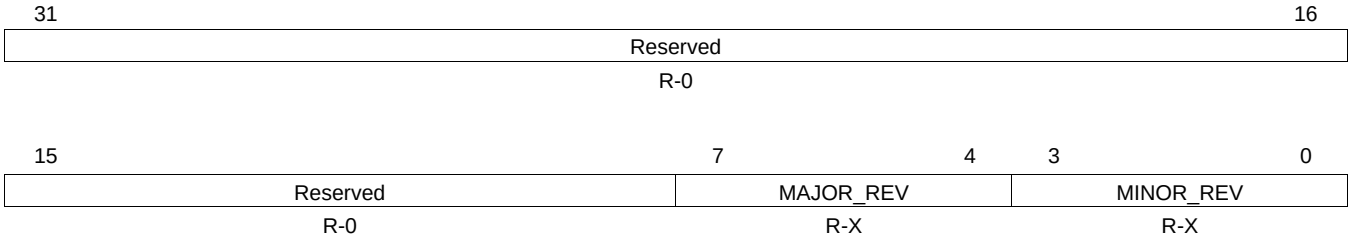
The module version register (MVR) is shown in Figure 67 and described in Table 61.

The reset value is fixed by hardware and corresponds to the RTL revision of this module. A reset has no effect on the value returned.

- UART / IrDA SIR only module is revision 1.x (WMU_012_1 specification).
- UART / IrDA with SIR, MIR and FIR support is revision 2.x (WMU_012_2 specification).
- UART / IrDA with SIR, MIR and FIR / CIR support is revision 3.x (this specification).

For example: MVR = 30h => Version 3.0 MVR = 38h => Version 3.8.

Figure 67. Module Version Register (MVR)



LEGEND: R = Read only; X = Unknown; -*n* = value after reset.

Table 61. Module Version Register (MVR)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-4	MAJOR_REV	0-Fh	Major revision number of the module.
3-0	MINOR_REV	0-Fh	Minor revision number of the module.

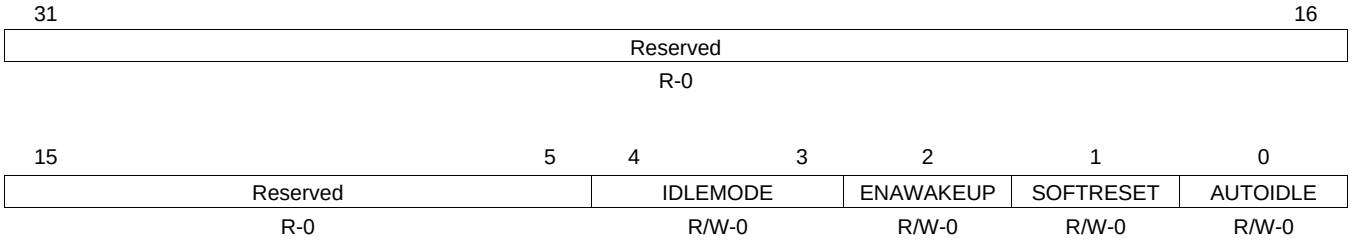
3.37 System Configuration Register (SYSC)

Offset Address (hex): 15h × S.

The system configuration register (SYSC) is shown in Figure 68 and described in Table 62.

The auto idle bit controls a power saving technique to reduce the logic power consumption of the OCP interface. That is to say when the feature is enabled, the clock will be gated off until an OCP command for this device has been detected. When the software reset bit is set high it causes a full device reset.

Figure 68. System Configuration Register (SYSC)



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

Table 62. System Configuration Register (SYSC)

Bit	Field	Value	Description
31-5	Reserved	0	Reserved.
4-3	IDLEMODE	0-3h	Power management REQ/ACK control. REF: OCP DESIGN GUIDELINES VERSION 1.1
		0	Force idle. An idle request is acknowledged unconditionally.
		1h	No-idle. An idle request is never acknowledged.
		2h	Smart idle. Acknowledgement to an idle request is given based in the internal activity of the module.
		3h	Reserved.
2	ENAWAKEUP		Wake up feature control.
		0	Wake up is disabled.
		1	Wake up capability is enabled.
1	SOFTRESET		Software reset. Set this bit to 1 to trigger a module reset. This bit is automatically reset by the hardware. During reads it always returns a 0.
		0	Normal mode.
		1	The module is reset.
0	AUTOIDLE		Internal OCP clock gating strategy.
		0	Clock is running.
		1	Automatic OCP clock gating strategy is applied, based on the OCP interface activity.

3.38 System Status Register (SYSS)

Offset Address (hex): 16h × S.

The system status register (SYSS) is shown in Figure 69 and described in Table 63.

Figure 69. System Status Register (SYSS)

31	16
Reserved	
R-0	

15	1	0
Reserved		RESETDONE
R-0		R-0

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

Table 63. System Status Register (SYSS)

Bit	Field	Value	Description
31-1	Reserved	0	Reserved.
0	RESETDONE		Internal reset monitoring.
		0	Internal Module Reset is ongoing.
		1	Reset completed.

3.39 Wake-Up Enable Register (WER)

Offset Address (hex): 17h × S.

The wake-up enable register (WER) is shown in Figure 70 and described in Table 64.

The UART wakeup enable register is used to mask and unmask a UART event that would subsequently notify the system. The events are any activity in the logic that could cause an interrupt and/ or an activity that would require the system to wakeup. It should be noted that even if the wakeup is disabled for certain events, if these events are also an interrupt to the UART, then the UART will still register the interrupt as such.

Figure 70. Wake-Up Enable Register (WER)

31								16
Reserved								
R-0								

15	7	6	5	4	3	2	1	0
Reserved		RLS	RHR	RX	DCD	RI	DSR	CTS
R-0		R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1

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

Table 64. Wake-Up Enable Register (WER)

Bit	Field	Value	Description
31-7	Reserved	0	Reserved.
6	RLS		Receiver line status interrupt can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
5	RHR		RHR interrupt can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
4	RX		RX event can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
3	DCD		$\overline{\text{DCD}}$ event can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
2	RI		$\overline{\text{RI}}$ event can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
1	DSR		$\overline{\text{DSR}}$ event can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.
0	CTS		$\overline{\text{CTS}}$ event can wake up the system.
		0	Event is not allowed to wake up the system.
		1	Event can wake up the system.

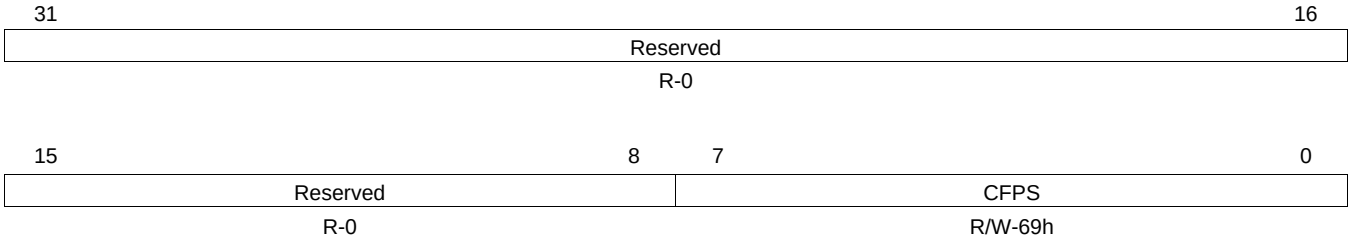
3.40 Carrier Frequency Prescaler Register (CFPS)

Offset Address (hex): 18h × S.

The carrier frequency prescaler register (CFPS) is shown in Figure 71 and described in Table 65.

Since the Consumer IR works at modulation rates of 30–56.8 KHz, the 48 MHz clock must be prescaled before the clock can drive the IR logic. This register sets the divisor rate to give a range to accommodate the remote control requirements in BAUD multiples of 12×. The value of the CFPS at reset is 105 (69h) that equates to a 38.1 KHz output from starting conditions. The 48 MHz carrier is prescaled by the CFPS that is then divided by the 12× BAUD multiple.

Figure 71. Carrier Frequency Prescaler Register (CFPS)



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

Table 65. Carrier Frequency Prescaler Register (CFPS)

Bit	Field	Value	Description
31-8	Reserved	0	Reserved.
7-0	CFPS	0-FFh	System clock frequency prescaler at (12x multiple). See Table 66 for CFPS values.

Table 66. (CFPS) Values

Target Freq (KHz)	CFPS (decimal)	Actual Freq (KHz)
30	133	30.08
32.75	122	32.79
36	111	36.04
36.7	109	36.69
38(1)	105	38.1
40	100	40
56.8	70	57.14

(1) Configured at reset to this value.

Appendix A Revision History

Table A-1 lists the changes made since the previous version of this document.

Table A-1. Document Revision History

Reference	Additions/Modifications/Deletions
Section 2.4.4.5	Added Note.

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		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265