Chapter 11

UART1 With HDLC and Modem Control Signals

11

11.1 Introduction

UART1 is the collection of a UART block along with a block to support a 9 pin modem interface and a block to support synchronous and asynchronous HDLC protocol support for full duplex transmit and receive. The following sections address each of these blocks.

11.2 UART Overview

Transmit and Receive data transfers through UART1 can either be managed by the DMA, interrupt driven, or CPU polled operations. A loopback control bit is available to enable system testing by routing the transmit data stream into the receiver.

The UART performs:

- Serial-to-parallel conversion on data received from a peripheral device.
- Parallel-to-serial conversion on data transmitted to the peripheral device.

The CPU reads and writes data and control/status information via the AMBA APB interface. The transmit and receive paths are buffered with internal FIFO memories allowing up to 16 bytes to be stored independently in both transmit and receive modes.

The UART:

- Includes a programmable baud rate generator which generates a common transmit and receive internal clock from the UART internal reference clock input, UARTCLK.
- Offers similar functionality to the industry-standard 16C550 UART device.
- Supports baud rates of up to 115.2 Kbits/s and beyond, subject to UARTCLK reference clock frequency.

The UART operation and baud rate values are controlled by the line control register (UART1LinCtrl).

The UART can generate:

- Four individually-maskable interrupts from the receive, transmit and modem status logic blocks.
- A single combined interrupt so that the output is asserted if any of the

individual interrupts are asserted and unmasked.

If a framing, parity or break error occurs during reception, the appropriate error bit is set, and is stored in the FIFO. If an overrun condition occurs, the overrun register bit is set immediately and FIFO data is prevented from being overwritten.

The FIFOs can be programmed to be 1 byte deep providing a conventional double-buffered UART interface.

The modem status input signals Clear To Send (**CTS**), Data Carrier Detect (**DCD**) and Data Set Ready (**DSR**) are supported. The additional modem status input Ring Indicator (**RI**) is not supported. Output modem control lines, such as Request To Send (**RTS**) and Data Terminal Ready (**DTR**), are not explicitly supported. Note that the separate modem block described later in this chapter does provide support for **RI**, **RTS**, and **DTR**.

11.2.1 UART Functional Description

A diagrammatic view of the UART is shown in Figure 11-1.

11.2.1.1 AMBA APB Interface

The AMBA APB interface generates read and write decodes for accesses to status and control registers and transmit and receive FIFO memories.

The AMBA APB is a local secondary bus which provides a low-power extension to the higher bandwidth Advanced High-performance Bus (AHB) within the AMBA system hierarchy. The AMBA APB groups narrow-bus peripherals to avoid loading the system bus and provides an interface using memory-mapped registers which are accessed under program control.

11.2.1.2 DMA Block

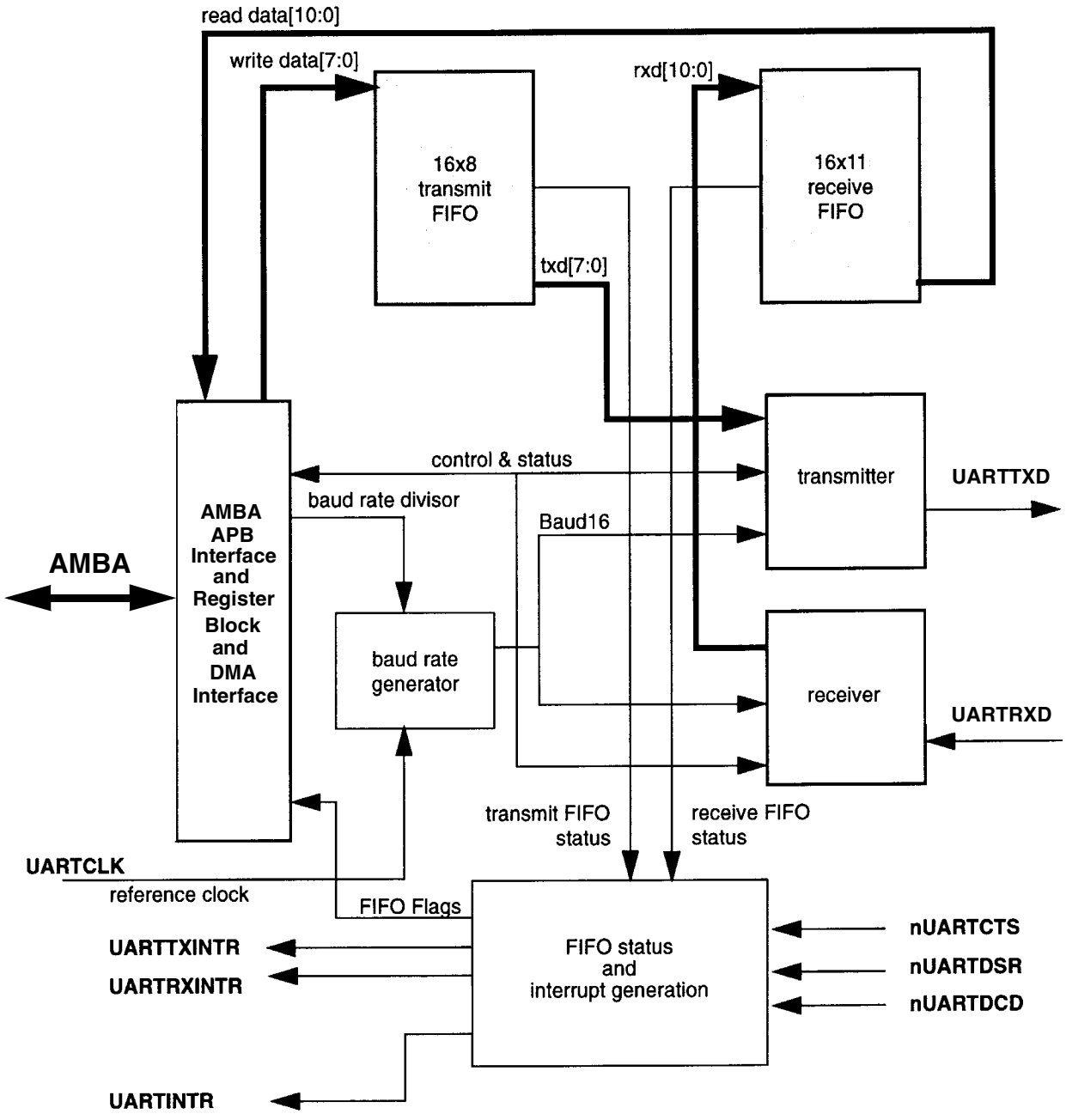
The DMA interface passes data between the UART FIFOs and an external DMA engine as an alternative to AMBA APB accesses. (See Chapter 7, “DMA Controller” on page 213 for additional details.) It may be configured to automatically move characters from the DMA engine to the transmit FIFO and from the receive FIFO to the DMA engine. The DMA engine may also indicate certain error conditions in the receive data to the DMA engine. Note that the DMA interface only supports 8-bit accesses to the FIFOs; status information in the receive FIFO is not passed to the DMA engine.

The UART1DMACtrl register controls the private interface between the DMA engine and the UART. Setting bit TXDMAE enables the transmit channel, while setting bit RXDMAE enables the receive channel. Setting bit DMAERR allows the UART to communicate certain error conditions to the DMA engine via RxEnd on the DMA channel. These conditions include receiving a break, a parity error, or a framing error. Note that configuration of the DMA channels in the DMA engine is also required for DMA operation with the UART.

11.2.1.3 Register Block

The register block stores data written or to be read across the AMBA APB interface.

Figure 11-1. UART Block Diagram



11.2.1.4 Baud Rate Generator

The baud rate generator contains free-running counters which generate the internal x16 clocks and the Baud16 signal. Baud16 provides timing information for UART transmit and receive control. Baud16 is a stream of pulses with a width of one UARTCLK clock period and a frequency of sixteen times the baud rate.

11.2.1.5 Transmit FIFO

The transmit FIFO is an 8-bit wide, 16-entry deep, first-in, first-out memory buffer. CPU data written across the APB interface and data written across the DMA interface is stored in the FIFO until read out by the transmit logic. The transmit FIFO can be disabled to act as a one-byte holding register.

11.2.1.6 Receive FIFO

The receive FIFO is an 11-bit wide, 16-entry deep, FIFO memory buffer. Received data, and corresponding error bits, are stored in the receive FIFO by the receive logic until read out by the CPU across the APB interface or across the DMA interface. The FIFO can be disabled to act as a one-byte holding register.

11.2.1.7 Transmit Logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. Control logic outputs the serial bit stream beginning with a start bit, data bits, least significant bit (LSB) first, followed by parity bit, and then stop bits according to the programmed configuration in control registers.

11.2.1.8 Receive Logic

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Parity, frame error checking and line break detection are also performed, and the data with associated parity, framing and break error bits is written to the receive FIFO.

11.2.1.9 Interrupt Generation Logic

Four individual maskable active HIGH interrupts are generated by the UART, and a combined interrupt output is also generated as an OR function of the individual interrupt requests.

The single combined UART interrupt (**UARTINTR**) is routed to the system interrupt controller. In addition, a separate receive FIFO interrupt **UARTRXINTR** and a transmit FIFO interrupt **UARTTXINTR** are routed to the system interrupt controller. (See Chapter 5, “Vectored Interrupt Controller” on page 99 for additional details.) Separate receive and transmit FIFO status signals indicate to the DMA interface when there is room in the transmit FIFO for more data and when there is data in the receive FIFO.

11.2.1.10 Synchronizing Registers and Logic

The UART supports both asynchronous and synchronous operation of the clocks, PCLK and UARTCLK. Synchronization registers and handshaking logic have been implemented, and are active at all times. This has a minimal impact on performance or area. Synchronization of control signals is performed on both directions of data flow, that is, from the PCLK to the UARTCLK domain and from the UARTCLK domain to the PCLK.

11.2.2 UART Operation

Control data is written to the UART line control register, UARTLCR. This register is 23 bits wide internally, but is externally accessed through the AMBA APB bus by three 8-bit wide register locations, UARTLCR_H, UARTLCR_M and UARTLCR.L.

UARTLCR defines the baud rate divisor and transmission parameters, word length, buffer mode, number of transmitted stop bits, parity mode and break generation.

The baud rate divisor is a 16-bit number used by the baud rate generator to determine the bit period. The baud rate generator contains a 16-bit down counter, clocked by the UART reference clock. When the value of the baud rate divisor has decremented to zero, the value of the baud rate divisor is reloaded into the down counter, and an internal clock enable signal, Baud16, is generated. This signal is then divided by 16 to give the transmit clock. A low number in the baud rate divisor gives a short bit period and vice versa.

Data received or transmitted is stored in two 16-byte FIFOs, though the receive FIFO has an extra three bits per character for status information.

For transmission, data is written into the transmit FIFO. This causes a data frame to start transmitting with the parameters indicated in UARTLCR. Data continues to be transmitted until there is no data left in the transmit FIFO. The **BUSY** signal goes HIGH as soon as data is written to the transmit FIFO (that is, the FIFO is non-empty) and remains asserted HIGH while data is being transmitted. **BUSY** is negated only when the transmit FIFO is empty, and the last character has been transmitted from the shift register, including the stop bits. **BUSY** can be asserted HIGH even though the UART may no longer be enabled.

When the receiver is idle (**UARTRXD** continuously 1, in the marking state) and a LOW is detected on the data input (a start bit has been received), the receive counter, with the clock enabled by Baud16, begins running and data is sampled on the eighth cycle of that counter (half way through a bit period).

The start bit is valid if **UARTRXD** is still LOW on the eighth cycle of Baud16, otherwise a false start bit is detected and it is ignored.

If the start bit was valid, successive data bits are sampled on every 16th cycle of Baud16 (that is, one bit period later) according to the programmed length of the data characters. The parity bit is then checked if parity mode was enabled.

Lastly, a valid stop bit is confirmed if **UARTRXD** is HIGH, otherwise a framing error has occurred. When a full word has been received, the data is stored in the receive FIFO, with any error bits associated with that word (see Table 2-1).

11.2.2.1 Error Bits

Three error bits are stored in bits [10:8] of the receive FIFO, and are associated with a particular character. There is an additional error which indicates an overrun error but it is not associated with a particular character in the receive FIFO. The overrun error is set when the FIFO is full and the next character has been completely received in the shift register. The data in the shift register is overwritten but it is not written into the FIFO.

Table 11-1: Receive FIFO Bit Functions

FIFO bit	Function
10	Break error
9	Parity error
8	Framing error
7:0	Received data

11.2.2.2 Disabling the FIFOs

Additionally, it is possible to disable the FIFOs. In this case, the transmit and receive sides of the UART have 1-byte holding registers (the bottom entry of the FIFOs). The overrun bit is set when a word has been received and the previous one was not yet read. In this implementation, the FIFOs are not physically disabled, but the flags are manipulated to give the illusion of a 1-byte register.

11.2.2.3 System/diagnostic Loopback Testing

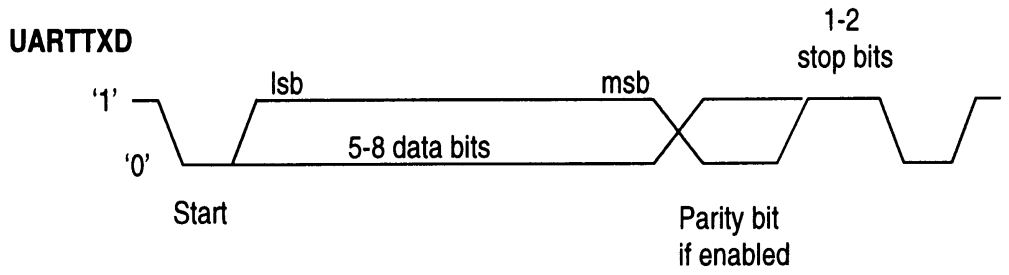
It is possible to perform loopback testing for UART data by setting the Loop Back Enable (LBE) bit to 1 in the control register UARTxCtrl (bit 7).

Data transmitted on **UARTTXD** output will be received on the **UARTRXD** input.

11.2.2.4 UART Character Frame

The UART character frame is shown in Figure 11-2:

Figure 11-2. UART Character Frame



11.2.3 Interrupts

There are five interrupts generated by the UART. Four of these are individual maskable active HIGH interrupts:

- **UARTMSINTR**
- **UARTRXINTR**
- **UARTRTINTR**
- **UARTTXINTR**

The interrupts are also output as a combined single interrupt **UARTINTR**.

Each of the four individual maskable interrupts is enabled or disabled by changing the mask bits in UARTCR. Setting the appropriate mask bit HIGH enables the interrupt.

The transmit and receive dataflow interrupts **UARTRXINTR** and **UARTTXINTR** have been separated from the status interrupts. This allows **UARTRXINTR** and **UARTTXINTR** to be used in a DMA controller, so that data can be read or written in response to just the FIFO trigger levels. The status of the individual interrupt sources can be read from UARTIIR.

11.2.3.1 UARTMSINTR

The modem status interrupt is asserted if any of the modem status lines (**nUARTCTS, nUARTDCD** and **nUARTDSR**) change. It is cleared by writing to the UART1IntIDIntClr register.

This interrupt is not independently connected to the system interrupt controller.

11.2.3.2 UARTRXINTR

The receive interrupt changes state when one of the following events occurs:

If the FIFOs are enabled and the receive FIFO is half or more full (it contains eight or more words), then the receive interrupt is asserted HIGH. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than half full.

If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the receive interrupt is asserted HIGH. The receive interrupt is cleared by performing a single read of the receive FIFO.

This interrupt is connected to the system interrupt controller.

11.2.3.3 UARTTXINTR

The transmit interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO is at least half empty (it has space for eight or more words), then the transmit interrupt is asserted HIGH. It is cleared by filling the transmit FIFO to more than half full.
- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the transmit FIFO is asserted HIGH. It is cleared by performing a single write to the transmitter FIFO.

The transmit interrupt **UARTTXINTR** is not qualified with the UART Enable signal, which allows operation in one of two ways. Data can be written to the transmit FIFO prior to enabling the UART and the interrupts. Alternatively, the UART and interrupts can be enabled so that data can be written to the transmit FIFO by an interrupt service routine.

This interrupt is connected to the system interrupt controller.

11.2.3.4 UARTRTINTR

The receive timeout interrupt is asserted when the receive FIFO is not empty and no further data is received over a 32-bit period. The receive timeout interrupt is cleared when the FIFO becomes empty through reading all the data (or by reading the holding register).

This interrupt is not independently connected to the system interrupt controller.

11.2.3.5 UARTINTR

The interrupts are also combined into a single output which is an OR function of the individual masked sources. This output is connected to the system interrupt controller to provide another level of masking on a individual peripheral basis. The combined UART interrupt is asserted if any of the four individual interrupts above are asserted and enabled.

11.3 Modem

The modem hardware adds modem control signals **RTSn**, **DTRn**, and **RI**. Two modem support registers provide a 16550 compatible modem control interface.

11.4 HDLC

The HDLC receiver handles framing, address matching, CRC checking, control-octet transparency or bit-stuffing, and optionally passes the CRC to the CPU at the end of the packet. The HDLC transmitter handles framing, CRC generation, and control-octet transparency or bit-stuffing. The CPU must assemble the frame in memory before transmission. The HDLC receiver and transmitter use the UART FIFOs to buffer the data streams.

When entering HDLC mode, always enable HDLC transmit and/or receive first by setting the TXE and/or RXE bit in the UART1HDLCCtrl, and then enable the UART. When leaving HDLC mode, disable the UART first, and then disable HDLC transmit and/or receive by clearing the TXE and/or RXE bit. This insures that no bytes are sent by the UART transmitter without proper HDLC framing, and that no bytes are received via the UART receiver without proper HDLC decoding. In HDLC mode, the UART should be configured to use 8-bit characters and no parity bit.

11.4.1 Overview of HDLC Modes

HDLC may operate in one of two basic modes, synchronous or asynchronous. Most configuration options affect both modes identically. Setting the UART1HDLCCtrl.SYNC bit selects synchronous mode and clearing it selects asynchronous mode. In asynchronous mode, each byte is transmitted using standard UART protocol framing (that is, start bit, data, parity, stop bit(s)). In synchronous mode, UART framing is bypassed.

The synchronous HDLC bit stream may be either a NRZ or Manchester encoded. In NRZ mode, both the transmitter and receiver may be synchronized to either an external or internal clock running at one cycle per bit period. The transmitter and receiver may operate independently in any of the four modes:

- Simple NRZ mode
- Manchester encoded
- NRZ mode with an internal clock
- NRZ mode with an external clock

In the first NRZ mode, the data stream does not contain an explicit or implicit clock, so synchronization between an HDLC transmitter and receiver cannot be guaranteed. A data bit value of “1” is encoded as a one in the bit stream, and a value of “0” as a zero.

The second mode, Manchester encoding, combines the HDLC data and clock into a single bit stream. In Manchester encoding, a transition always occurs in the middle of a transmitted bit and the value after this transition is the actual value of the bit. That is, a “0” bit is represented by a transition from high to low, and a “1” bit by a transition from low to high. Because a transition always

occurs in the middle of a bit, the receiver can always extract the proper data after a suitable period of synchronization, provided the signal quality is good.

The third and fourth modes utilize NRZ encoding of the data accompanied by a separate clock signal. The period of the clock signal is one bit period. When using an internal clock, the HDLC transmitter generates a clock such that the data is stable at the clock's rising edge. Hence, an external receiver may sample each data bit at the rising edge of the clock. The internal receiver will also use the same clock to sample input data if programmed to do so.

The internal transmitter and/or receiver may also synchronize to an external, rather than internal, clock. The internal receiver gets this clock along with the incoming HDLC data, allowing it to always sample bits at the right time. In addition, the internal transmitter will synchronize the data it transmits to this clock if programmed to do so. The transmitter will insure that its data is valid before the rising edge of the clock, and the receiver expects the same of the incoming bit stream.

11.4.2 Selecting HDLC Modes

By default, HDLC is NRZ-encoded. Set bit UART1HDLCCtrl.TXENC to force Manchester encoding in the transmitter, and set bit UART1HDLCCtrl.RXENC to make the receiver expect Manchester encoding.

The receiver utilizes a digital PLL to synchronize to the incoming encoded bit stream. The digital PLL should always successfully lock on to an incoming data stream within two bytes provided that the first two bits of the first byte are either "01" or "10". Hence, at a minimum, two bytes must precede the final opening flag to insure that the HDLC receiver sees the packet. To meet this requirement, the simplest approach is to insure that at least three opening flags are received if the packet is Manchester encoded. (Note that to meet this requirement when transmitting, field HDLC1Ctrl.FLAG should be set to 0010b.)

Three bits in various combination determine how an external or internal clock may be used along with NRZ data. The clock will have a period equal to the bit period of the data stream, and it is expected that the internal or external receiver will sample the bit at or near the rising edge of this clock.

To generate an internal clock suitable for sending along with the transmitted data, set UART1HDLCCtrl.TXCM and UART1HDLCCtrl.CMAS. To make the receiver use the same internal clock, set UART1HDLCCtrl.RXCM. To make the receiver use an externally generated clock, clear UART1HDLCCtrl.CMAS, but set UART1HDLCCtrl.RXCM.

To force the transmitter to use the same external clock, also set UART1HDLCCtrl.TXCM. The clock is either internal or external, that is, the receiver cannot use an external clock while the transmitter generates and sends an internal one. Refer to the documentation for the DeviceCfg register

in Syscon for the use and routing of HDLC clocks to or from external pins on the device.

The internal clock is generated by the transmitter only while it is sending data or flags; the clock is not generated while the transmitter is idle. For this reason, another transmitter which expects to use this clock to at any time send its own packets cannot reliably do so. To insure that a clock is continuously generated, the IDLE bit in the UART1HDLCCtrl register may be set, which causes this transmitter to continously send flags between packets instead of going idle.

Table 11-2 summarizes the legal HDLC mode configurations.

Table 11-2: Legal HDLC Mode Configurations

UART1HDLCCtrl Bits Set						Transmit Mode	Receive Mode
CMAS	TXCM	RXCM	TXENC	RXENC	SYNC		
-	-	-	-	-	-	Asynchronous NRZ	Asynchronous NRZ
-	-	-	-	-	1	Synchronous NRZ	Synchronous NRZ
-	-	-	-	1	1	Synchronous NRZ	Manchester
-	-	-	1	-	1	Manchester	Synchronous NRZ
-	-	-	1	1	1	Manchester	Manchester
-	-	1	-	-	1	Synchronous NRZ	External clock
-	-	1	1	-	1	Manchester	External clock
-	1	-	-	-	1	External clock	Synchronous NRZ
-	1	-	-	1	1	External clock	Manchester
1	1	-	-	-	1	Internal clock	Synchronous NRZ
1	1	-	-	1	1	Internal clock	Manchester
-	1	1	-	-	1	External clock	External clock
1	1	1	-	-	1	Internal clock	Internal clock

11.4.3 HDLC Transmit

In normal operation, the HDLC transmitter either continuously sends flags or holds the transmit pin in a marking state, depending on the setting of the UART1HDLCCtrl.IDLE bit. When data appears in the transmit FIFO, it begins sending a packet. If in the marking state, it sends from 1 to 16 opening flags, as specified by the UART1HDLCCtrl.FLAG field. If already sending flags, it ensures that at least the specified number have been sent. It then begins sending the bytes in the FIFO, inserting and modifying the data depending on the HDLC mode.

In asynchronous HDLC, the transmitter enforces control-octet transparency. Whenever a flag byte (01111110b) or an escape byte (01111101b) appears in the data, the transmitter inverts the fifth bit and precedes it with an escape byte.

In synchronous HDLC, the transmitter performs bit-stuffing (except for flags). Whenever five consecutive “1” bits appear in the transmitted bit stream, a “0” bit is inserted, preventing six ones from appearing consecutively.

When the transmit FIFO underruns, the HDLC transmitter does one of two things (depending on the setting of the UART1HDLCCtrl.TUS bit). If the TUS bit is zero, the transmitter first sends the CRC (if CRC is enabled) and then sends from 1 to 16 closing flags, as specified in the UART1HDLCCtrl.FLAG field, terminating the packet.

If TUS is one, the transmitter aborts the packet. In synchronous HDLC, it sends a byte of all ones (since seven consecutive ones signifies an abort), following by at least one closing flag. In asynchronous HDLC, it sends an escape and then at least one closing flag. The number of closing flags is from 1 to 16, as specified in the UART1HDLCCtrl.FLAG field.

When a packet ends, the UART1HDLCSts.TFC bit is set, and if UART1HDLCCtrl.TFCEN is set, an interrupt is generated. When a packet is aborted, the UART1HDLCCtrl.TAB bit is set, also generating an interrupt if UART1HDLCCtrl.TABEN is set.

11.4.4 HDLC Receive

The HDLC receiver continuously reads bytes from the UART receiver until it finds a flag followed by a byte other than a flag. Then, if in asynchronous mode, it processes the incoming bytes (including the first after the flag), reversing control-octet transparency, or, if in synchronous mode, it reverses bit-stuffing. Processed bytes are placed in the receive FIFO. When programmed to receive a Manchester encoded bit stream, UART1HDLCSts.PLLCS indicates whether the DPLL in the receiver has locked on to the carrier.

When the last byte of data for a packet is read from the receive FIFO, the HDLC logic sets a number of bits in the UART1HDLCSts depending on the state of the system and the way the packet was terminated. In all cases, the RFC bit and EOF bit are set. If the receive FIFO overflowed while the packet was being received, the ROR bit is also set. If CRC is enabled and the received CRC does not match the calculated one, the CRE bit is set. The RFC bit is set and, if UART1HDLCCtrl.RFCEN is set, an interrupt is generated. If the packet was aborted, the RAB bit is set, and an interrupt generated if the UART1HDLCCtrl.RABEN bit is set. If using Manchester encoding and the packet was aborted due to losing synchronization with the encoded clock, the UART1HDLCCtrl.PLLE bit is set.

Besides setting bits in the UART1HDLCSts and possibly causing interrupts, reading the last byte of a packet also loads the UART1HDLCRXInfoBuf register with data describing the packet. BRAB, BCRE, BROR, and BPLLE are copied from RAB, CRE, ROR, and PLLE in the UART1HDLCSts. BFRE is copied from the FE bit in the UART1RXSts. BC is set to the number of bytes in

the packet that were read from the FIFO. Whenever this register is written by the receiver and has not been read since previously it was previously written, the UART1HDLCSts.RIL bit is set, and, if UART1HDLCSts.RILEN is set, an interrupt is generated.

If a new packet is received and the first byte of that packet cannot be written into the receive FIFO because it has overflowed, the UART1HDLCSts.RFL bit is set and the packet is discarded. An interrupt is generated if the UART1HDLCCtrl.RFLEN bit is also set.

11.4.5 CRCs

Several bits in the UART1HDLCCtrl determine how CRCs are generated by the transmitter and processed by the receiver. By setting the CRCE bit, the HDLC transmitter will calculate and append a CRC to each packet. The CRC may be either 16-bit or 32-bit, depending on the CRCS bit. Furthermore, it will be inverted prior to transmission if the CRCN bit is set. If CRCs are enabled, the receiver will expect the same type of CRC that the transmitter sends. It will automatically calculate the CRC for the received packet in the fly, and if the calculated CRC does not match the received one, the UART1RXSts.CRE bit will be set when the last byte of the received packet is read from the UART1Data. The receiver does not pass the CRC to the CPU unless the CRCApd bit is set.

11.4.6 Address Matching

When address matching is enabled, the HDLC receiver will ignore any packet whose address does not match the programmed configuration. Address matching is enabled and address size specified by the UART1HDLCCtrl.AME bits. The UART1HDLCAddMtchVal specifies the addresses that are compared while the UART1HDLCAddMask controls which bits in each address are compared If one-byte addressing is used, each byte in UART1HDLCAddMtchVal specifies an address to match, while the corresponding byte in UART1HDLCAddMask specifies which bits of each address must match. If two-byte addressing is used, each halfword in UART1HDLCAddMtchVal specifies an address to match and the corresponding halfword in UART1HDLCAddMask specifies which bits of each address to match. Hence, up to four different one-byte addresses and two different two-byte addresses may be specified. An incoming address consisting entirely of “1”s, that is, 0xFF or 0xFFFF, will always match, as it is expected to be the broadcast address. For packets whose addresses do not match, the HDLC receiver will generate no interrupts, modify no status bits, and place no data in the receive FIFO.

Table 11-3: HDLC Receive Address Matching Modes

AME	Match Function	Address Match Test
00	No matching	
01	One byte address	***NOT***((AMV[31:24] ***XOR*** ADDR) ***AND*** AMSK[31:24]) ***OR*** ***NOT***((AMV[23:16] ***XOR*** ADDR) ***AND*** AMSK[23:16]) ***OR*** ***NOT***((AMV[15:8] ***XOR*** ADDR) ***AND*** AMSK[15:8]) ***OR*** ***NOT***((AMV[7:0] ***XOR*** ADDR) ***AND*** AMSK[7:0]) ***OR*** ADDR = 0xFF
10	Two byte address	***NOT***((AMV[31:16] ***XOR*** ADDR) ***AND*** AMSK[31:16]) ***OR*** ***NOT***((AMV[15:0] ***XOR*** ADDR) ***AND*** AMSK[15:0]) ***OR*** ADDR = 0xFFFF
11	Undefined	

11.4.7 Aborts

If a packet is aborted or is too short, or if using Manchester encoding and the receiver DPLL loses the carrier signal, the CPU will see at least some part of the packet in the receive FIFO. In all cases, reading the last byte of the packet from the receive FIFO will set the EOF and RAB bits in the UART1HDLCSts (and possibly generate an interrupt). In the case of an abort indicated by an HDLC transmitter, that is, an escape-closing flag sequence in asynchronous mode or an all “1”s byte in synchronous mode, all bytes received in the frame will appear in the receive FIFO.

In asynchronous mode, if the abort is caused by a framing error (a missing stop bit), all bytes up to and including the misframed byte will appear in the receive FIFO. Reading the last byte will also set the UART1HDLCSts.FRE bit.

In synchronous mode, if the abort is caused by a misaligned flag or a series of seven consecutive “1”s, all bytes except the one containing the bit after the sixth “1” will appear in the receive FIFO. If the abort is caused by the receiver DPLL losing synchronization with a Manchester encoded bit stream, the UART1HDLCSts.DPLLE bit is set.

Finally, if the packet is too short, that is, there are not enough received bytes to hold the specified number of address and CRC bytes, the entire packet will appear in the receive FIFO. In all cases, the packet is illegal and will be ignored by the CPU.

11.4.8 DMA

The DMA engine may be used with the UART when transmitting and receiving HDLC packets. The transmit and receive channels may operate completely independently.

When receiving data in HDLC mode, the DMA channel reads the packet data byte by byte from the RX FIFO. When it reads the final byte, the HDLC RFC interrupt will occur if enabled. However, the DMA channel, which buffers the data, may not write all of the data to memory. To insure that the DMA channel dumps the data, the interrupt handling routine must do the following:

1. Note the values in the MAXCNTx and REMAIN registers for the DMA channel. The difference is the number of bytes read from the UART, which is the size of the HDLC packet. Call this difference N. Note that the BC field of the UART1HDLCRXInfoBuf register should also be N.
2. Temporarily disable the UART DMA RX interface by clearing the RXDMAE bit in the UART1DMACtrl register.
3. Wait until the difference between the CURRENTx and BASEx registers in the DMA channel is equal to N + 1.

An extra byte will be read from the UART by the DMA channel. It should be ignored.

Note that if the DMAERR bit in the UART1DMACtrl register is set and the HDLC receiver is in asynchronous mode, if the receiver sees a break, parity, or framing error, it will indicate an error condtion via RxEnd on the DMA channel.

11.4.9 Writing Configuration Registers

It is assumed that various configuration registers for the UART/HDLC are not written more than once in quick succession, in order to insure proper synchronization of configuration information across the implementation. Such registers include UART1Ctrl and UART1LinCtrlHigh as well as UART1HDLCCtrl, UART1HDLCAddMtchVal, UART1HDLCAddMask. These registers should not change often in typical use.

The simplest way to fulfill this requirement with respect to writing the UART1Ctrl and UART1HDLCCtrl registers is to insure that the HDLC transmitter is enabled before the UART transmit logic. This will ensure that the UART does not transmit incorrect characters or unexpectedly transmit characters with UART framing,

First the UART1HDLCCtrl register should be written, setting the TXE bit. Then the UART1Ctrl register should be written, setting the UARTE bit. In between the two writes, at least two UARTCLK periods must occur. Under worst case conditions, at least 55 HCLK periods must separate the two writes. The simplest way to due this is separate the two writes by 55 NOPs.

11.5 UART1 Package Dependency

UART1 uses package pins **RXD0**, **TXD0**, **CTSn**, **DSRn**, **DTRn**, **RTSn**, **EGPIO[3]**, and **EGPIO[0]**, which are described in Table 11-4.

Table 11-4: UART1 Pin Functionality

PIN	Description
RXD0	UART1 input pin
TXD0	UART1 output pin
CTSn	Modem input: Clear To Send
DSRn	Modem input: Data Set Ready (also used for DCDn Data Carrier Detect)
EGPIO[0]	Modem input RIn: Ring Indicator if Syscon register DeviceCfg[25] MODonGPIO is set. Otherwise, RIn is driven low.
DTRn	Modem output Data Terminal Ready if Syscon register TESTCR[27] RTConGPIO is clear.
RTSn	Modem output: Ready To Send
EGPIO[3]	HDLC clock

The use of **EGPIO[3]** is determined by several bits in Syscon register DeviceCfg. See Table 11-5.

Table 11-5: DeviceCfg Register Bit Functions

bit 13 HC1IN	bit 12 HC1EN	Function
0	x	External HDLC clock input is driven low.
1	1	External HDLC clock input is driven by EGPIO[3].
0	1	Internal HDLC clock output drives EGPIO[3].

11.5.1 Clocking Requirements

There are two clocks, PCLK and UARTCLK.

UARTCLK frequency must accommodate the desired range of baud rates:

$$F_{UARTCLK_{MIN}} \geq 32 \times baudrate_{MAX}$$

$$F_{UARTCLK_{MAX}} \leq 32 \times 65536 \times baudrate_{MIN}$$

The frequency of UARTCLK must also be within the required error limits for all baud rates to be used.

To allow sufficient time to write the received data to the receive FIFO, UARTCLK must be less than or equal to four times the frequency of PCLK:

$$F_{UARTCLK} \leq 4 \times F_{PCLK}$$

11.5.2 Bus Bandwidth Requirements

There are two basic ways of moving data to and from the UART FIFOs:

- Direct DMA interface - This permits byte-wide access to the UART without using the APB. The DMA block will pack or unpack individual bytes so that it reads or writes full 32-bit words rather than individual bytes.

- Accessing the UART via the APB - This requires APB/AHB bus bandwidth. Then, both a read and write are required for each 8-bit data byte.

Bandwidth requirements also depend on the selected baud rate, character size, parity selection, number of stop bits, and spacing between characters (if receiving).

For example, assume transmission protocols of 115,200 baud, 8-bit characters, even parity, one stop bit, no space between characters. There are 11 bits per character, so 115,200 / 11 = 10,473 characters per second. If both transmitting and receiving, 20,945 characters per second pass through the UART. Accessing the UART through the DMA interface requires one access per 32-bits, implying only 20,945 / 4 = 5,236 AHB accesses per second. Accessing the UART through the APB requires two accesses per byte, implying 20,945 APB bus accesses.

As another example, assume 230,400 baud (the maximum with a UARTCLK equal to 7.3728 Mhz), 5-bit characters, no parity, one stop bit, and no space between characters. There are 7 bits per character, so 230,400 / 7 = 32,914 characters per second. Simultaneous transmitting and receiving implies 65,829 characters per second. Using the DMA interface would result in 16,457 AHB accesses per second, while using the APB to access the UART leads to 65,829 bus accesses per second.

11.6 Registers

11

UART Register Descriptions

UART1Data

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								DATA							

Address:

0x808C_0000 - Read/Write

Default:

0x0000_0000

Definition:

UART Data Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DATA: UART Data: read for receive data, write for transmit data
For words to be transmitted:
• if the FIFOs are enabled, data written to this location is pushed onto the transmit FIFO
• if the FIFOs are not enabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). The write operation initiates transmission from the UART. The data is prefixed with a start bit, appended with the appropriate parity bit (if parity is enabled), and a stop bit. The resultant word is then transmitted.

For received words:
• if the FIFOs are enabled, the data byte is extracted, and

a 3-bit status (break, frame and parity) is pushed onto the 11-bit wide receive FIFO
• if the FIFOs are not enabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO).

The received data byte is read by performing reads from the UART1Data register while the corresponding status information can be read by a successive read of the UART1RXSts register.

UART1RXSts

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												OE	BE	PE	FE

Address:

0x808C_0004 - Read/Write

Default:

0x0000_0000

Definition:

UART1 Receive Status Register/Error Clear Register. Provides receive status of the data value last read from the UART1Data. A write to this register clears the framing, parity, break and overrun errors. The data value is not important. Note that BE, PE and FE are not used for synchronous HDLC.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

OE: Overrun Error. This bit is set to “1” if data is received and the FIFO is already full. This bit is cleared to “0” by a write to UART1RXSts. The FIFO contents remain valid since no further data is written when the FIFO is full. Only the contents of the shift register are overwritten. The data must be read in order to empty the FIFO.

BE: Break Error. This bit is set to 1 if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). This bit is cleared to 0 after a write to UART1RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received.

PE: Parity Error. When this bit is set to 1, it indicates that the parity of the received data character does not match the parity selected in UART1LinCtrlHigh (bit 2). This bit is cleared to 0 by a write to UART1RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO.

FE: Framing Error. When this bit is set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is “1”). This bit is cleared to 0 by a write to UART1RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO.

UART1LinCtrlHigh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									WLEN		FEN	STP2	EPS	PEN	BRK

Address:

0x808C_0008 - Read/Write

Default:

0x0000_0000

Definition:

UART1 Line Control Register High. UART1LinCtrlHigh, UART1LinCtrlMid and UART1LinCtrlLow form a single 23-bit wide register (UART1LinCtrl) which is updated on a single write strobe generated by an UART1LinCtrlHigh write. In order to internally update the contents of UART1LinCtrlMid or UART1LinCtrlLow, a UART1LinCtrlHigh write must always be performed at the end.

To update the three registers there are two possible sequences:

- UART1LinCtrlLow write, UART1LinCtrlMid write and UART1LinCtrlHigh write
- UART1LinCtrlMid write, UART1LinCtrlLow write and UART1LinCtrlHigh write.

To update UART1LinCtrlLow or UART1LinCtrlMid only:

- UART1LinCtrlLow write (or UART1LinCtrlMid write) and UART1LinCtrlHigh write.

Bit Descriptions:

RSVD:	Reserved. Unknown During Read.
WLEN:	Number of bits per frame: 11 = 8 bits 10 = 7 bits 01 = 6 bits 00 = 5 bits
FEN:	FIFO Enable. 1 - Transmit and receive FIFO buffers are enabled (FIFO mode). 0 - The FIFOs are disabled (character mode) that is, the FIFOs become 1-byte-deep holding registers.
STP2:	Two Stop Bits Select. 1 - Two stop bits are transmitted at the end of the frame. 0 - One stop bit is transmitted at the end of the frame. The receive logic does not check for two stop bits being received.
EPS:	Even Parity Select. 1 - Even parity generation and checking is performed during transmission and reception, which checks for an even number of “1”s in data and parity bits. 0 - Odd parity checking is performed, which checks for an odd number of “1”s. This bit has no effect when parity is disabled by Parity Enable (bit 1) being cleared to 0.
PEN:	Parity Enable. 1 - Parity checking and generation is enabled, 0 - Parity checking and generation is disabled and no parity bit is added to the data frame.

BRK: Send Break.
1 - A low level is continually output on the **UARTTXD** output, after completing transmission of the current character. This bit must be asserted for at least one complete frame transmission time in order to generate a break condition. The transmit FIFO contents remain unaffected during a break condition.
0 - For normal use, this bit must be cleared.

UART1LinCtrlMid

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								BR							

Address:
0x808C_000C - Read/Write

Default:
0x0000_0000

Definition:
UART Line Control Register Middle.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BR: Baud Rate Divisor bits [15:8]. Most significant byte of baud rate divisor. These bits are cleared to 0 on reset.

UART1LinCtrlLow

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								BR							

Address:
0x808C_0010 - Read/Write

Default:
0x0000_0000

Definition:

UART Line Control Register Low.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BR: Baud Rate Divisor bits [7:0]. Least significant byte of baud rate divisor. These bits are cleared to 0 on reset. The baud rate divisor is calculated as follows:

Baud rate divisor
$BAUDDIV = (F_{UARTCLK} / 16 * \text{Baud rate})) - 1$

where $F_{UARTCLK}$ is the UART reference clock frequency. A baud rate divisor of zero is not allowed and will result in no data transfer.

UART1Ctrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								LBE	RTIE	TIE	RIE	MSIE	RSVD		UARTE

Address:

0x808C_0014 - Read/Write

Default:

0x0000_0000

Definition:

UART1 Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

LBE: Loopback Enable. If this bit is set to 1, data sent to **TXD** is received on **RXD**. This bit is cleared to 0 on reset, which disables the loopback mode.

RTIE: Receive Timeout Enable. If this bit is set to 1, the receive timeout interrupt is enabled.

TIE: Transmit Interrupt Enable. If this bit is set to 1, the transmit interrupt is enabled.

RIE: Receive Interrupt Enable. If this bit is set to 1, the receive interrupt is enabled.

MSIE: Modem Status Interrupt Enable. If this bit is set to 1, the modem status interrupt is enabled.

UARTE: UART Enable. If this bit is set to 1, the UART is enabled. Data transmission and reception occurs for UART signals.

UART1Flag

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS

Address:

0x808C_0018 - Read Only

Default:

0x0000_0000

Definition:

UART Flag Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

TXFE: Transmit FIFO Empty. The meaning of this bit depends on the state of the FEN bit in the UART1LinCtrlHigh register. If the FIFO is disabled, this bit is set when the transmit holding register is empty. If the FIFO is enabled, the TXFE bit is set when the transmit FIFO is empty.

RXFF: Receive FIFO Full. The meaning of this bit depends on the state of the FEN bit in the UART1LinCtrlHigh register. If the FIFO is disabled, this bit is set when the receive holding register is full. If the FIFO is enabled, the RXFF bit is set when the receive FIFO is full.

TXFF: Transmit FIFO Full. The meaning of this bit depends on the state of the FEN bit in the UART1LinCtrlHigh register. If the FIFO is disabled, this bit is set when the transmit holding register is full. If the FIFO is enabled, the TXFF bit is set when the transmit FIFO is full.

RXFE: Receive FIFO Empty. The meaning of this bit depends on the state of the FEN bit in the UART1LinCtrlHigh register. If the FIFO is disabled, this bit is set when the receive holding register is empty. If the FIFO is enabled, the RXFE bit is set when the receive FIFO is empty.

BUSY: UART Busy. If this bit is set to 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether the UART is enabled or not).

DCD: Data Carrier Detect status. This bit is the complement of the UART data carrier detect (**nUARTDCD**) modem status input. That is, the bit is 1 when the modem status input is 0.

DSR: Data Set Ready status. This bit is the complement of the UART data set ready (**nUARTDSR**) modem status input. That is, the bit is 1 when the modem status input is 0.

CTS: Clear To Send status. This bit is the complement of the UART clear to send (**nUARTCTS**) modem status input. That is, the bit is 1 when the modem status input is 0.

UART1IntIDIntClr

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												RTIS	TIS	RIS	MIS

Address:

0x808C_001C - Read/Write

Default:

0x0000_0000

Definition:

UART Interrupt Identification and Interrupt Clear Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RTIS: Receive Timeout Interrupt Status. This bit is set to 1 if the **UARTRTINTR** receive timeout interrupt is asserted. This bit is cleared when the receive FIFO is empty or the receive line goes active.

TIS: Transmit Interrupt Status.
1 - The **UARTTXINTR** transmit interrupt is asserted, which occurs when the transmit FIFO is not full.
0 - The transmit FIFO is full.

RIS: Receive Interrupt Status.
1 - The **UARTRXINTR** receive interrupt is asserted, which occurs when the receive FIFO is not empty.
0 - The receive FIFO is empty.

MIS: Modem Interrupt Status. This bit is set to 1 if the **UARTMSINTR** modem status interrupt is asserted. This bit is cleared by writing any value to this register.

UART1DMACtrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													DMAERR	TXDMAE	RXDMAE

Address:

0x808C_0028 - Read/Write

Default:

0x0000_0000

Definition:

UART DMA Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DMAERR: RX DMA error handing enable. If 0, the RX DMA interface ignores error conditions in the UART receive section. If 1, the DMA interface stops and notifies the DMA block when an error occurs. Errors include break errors, parity errors, and framing errors.

TXDMAE: TX DMA interface enable. Setting to 1 enables the private DMA interface to the transmit FIFO.

RXDMAE: RX DMA interface enable. Setting to 1 enables the private DMA interface to the receive FIFO.

Modem Register Descriptions

UART1ModemCtrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								0	0	0	LOOP	OUT2	OUT1	RTS	DTR

Address:

0x808C_0100 - Read/Write

Default:

0x0000_0000

Definition:

Modem Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

0: Must be written as “0”.

LOOP: Activate internal modem control loopback function. This internal loopback only affects the hardware handshake signals. Use the UART1Ctrl LBE bit to loopback the serial data.
When high, modem control outputs **RTSn** and **DTRn** are forced high (inactive), and modem control inputs are driven by outputs:
DSR = DTR
CTS = RTS
RI2 = OUT1
DCD = OUT2

OUT2: OUT2 function. Used for internal loopback.

OUT1: OUT1 function. Used for internal loopback.

RTS: RTS output signal:
1 - **RTSn** pin low
0 - **RTSn** pin high

DTR: DTR output signal:
1 - **DTRn** pin low
0 - **DTRn** pin high

UART1ModemSts

11

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS

Address:

0x808C_0104 - Read Only

Default:

0x0000_0000

Definition:

Modem Status Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DCD: Inverse of **DCDn** input pin. Note that this is identical to the **DSR** device pin.

RI: Inverse of **RI** input pin.

DSR: Inverse of the **DSRn** pin. Note that this is identical to the **DCD** device pin

CTS: Inverse **CTSn** input pin.

DDCD: Delta **DCD** - **DCDn** pin changed state since last read.

TERI: Trailing Edge Ring Indicator. **RI** input pin has changed from low to high.

DDSR: Delta **DSR** - **DSRn** pin has changed state since last read.

DCTS: Delta **CTS** - **CTSn** pin has changed state since last read.

HDLC Register Descriptions

UART1HDLCCtrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD				CMAS	TXCM	RXCM	TXENC	RXENC	SYNC	TFCEN	TABEN	RFCEN	RILEN	RFLEN	RTOEN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLAG				CRCN	CRCApd	IDLE	AME		IDLSpc	CRCZ	RXE	TXE	TUS	CRCE	CRCS

Address:

0x808C_020C - Read/Write

Default:

0x0000_0000

Definition:

HDLC Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

CMAS: Clock Master:
1 - Transmitter and/or receiver use 1x clock generated by the internal transmitter.
0 - Transmitter and/or receiver use 1x clock generated externally.

TXCM: Transmit Clock Mode.
1 - Generate 1x clock when in synchronous HDLC mode using NRZ encoding.
0 - Do not generate clock.
This bit has no effect unless TXENC is clear and synchronous HDLC is enabled.

RXCM: Receive Clock Mode.
1 - Use external 1x clock when in synchronous HDLC mode using NRZ encoding.
0 - Do not use external clock.
This bit has no effect unless RXENC is clear and synchronous HDLC is enabled.

TXENC: Transmit Encoding method.
1 - Use Manchester bit encoding.
0 - Use NRZ bit encoding.
This bit has no effect unless synchronous HDLC is enabled

RXENC: Receive Encoding method.
1 - Use Manchester bit encoding.
0 - Use NRZ bit encoding.
This bit has no effect unless synchronous HDLC is enabled.

SYNC: Synchronous / Asynchronous HDLC Enable.
0 - Select asynchronous HDLC for TX and RX.
1 - Select synchronous HDLC for TX and RX.

TFCEN: Transmit Frame Complete Interrupt Enable.
0 - TFC interrupt will not occur.
1 - TFC interrupt will occur whenever TFC bit is set.

TABEN: Transmit Frame Abort Interrupt Enable.
0 - TAB interrupt will not occur.
1 - TAB interrupt will occur whenever TAB bit is set.

RFCEN: Receive Frame Complete Interrupt Enable.
0 - RFC interrupt will not occur.
1 - RFC interrupt will occur whenever RAB bit or EOF bit is set.

RILEN: Receive Information Lost Interrupt Enable.
0 - RIL interrupt will not occur.
1 - RIL interrupt will occur whenever RIL bit is set.

RFLEN: Receive Frame Lost Interrupt Enable.
0 - RFL interrupt will not occur.
1 - RFL interrupt will occur whenever RFL bit is set.

RTOEN: Receiver Time Out Interrupt Enable.
0 - RTO interrupt will not occur.
1 - RTO interrupt will occur whenever RTO bit is set.

FLAG: Minimum number of opening and closing flags for HDLC TX. The minimum number of flags between packets is this 4-bit value plus one. Hence, 0000b forces at least one opening flag and one closing flag for each packet, and 1111b forces at least 16 opening and closing flags. The closing flags of one packet may also be the opening flags of the next one if the transmit line does not go idle in between. Note that HDLC RX does not count flags; only one is necessary (or three in Manchester mode).

CRCN: CRC polarity control.
0 - CRC transmitted not inverted.
1 - CRC transmitted inverted.

CRCApd: CRC pass through.
0 - Do not pass received CRC to CPU.
1 - Pass received CRC to CPU.

IDLE: Idle mode.
0 - Idle-in Mark mode - When HDLC is idle (not transmitting starting/stop flags or packets), hold the transmit data pin high.
1 - Idle-in Flag mode - When HDLC is idle, transmit continuous flags.

AME: Address Match Enable. Activates address matching on received frames.
00 - No address matching
01 - 4 x 1 byte matching
10 - 2 x 2 byte matching
11 - Undefined, no matching

IDLSpc: Idle in space
0 - TX idle in mark (normal)
1 - TX idle in space
RX will receive Manchester encoded data whether it idles in mark or space.

CRCZ: CRC zero seed
0 - Seed CRC calculations with all ones; that is, 0xFFFF for 16 bit words and 0xFFFF_FFFF for 32 bit words.
1 - Seed CRC calculations with all zeros.
Applies to both RX and TX.

RXE: HDLC Receive Enable.
0 - Disable HDLC RX. If UART is still enabled, UART may still receive normally.
1 - Enable HDLC RX.

TXE: HDLC Transmit Enable.
0 - Disable HDLC TX. If UART is still enabled, UART may still transmit normally.
1 - Enable HDLC TX.

TUS: Transmit FIFO Underrun Select
0 - TX FIFO underrun causes CRC (if enabled) and stop flag to be transmitted.
1 - TX FIFO underrun causes abort (escape-flag) to be transmitted.

CRCE: CRC enable.
0 - No CRC is generated by TX or expected by RX.
1 - HDLC TX automatically generates and sends a CRC at the end of a packet, and HDLC RX expects a CRC at the end of a packet.

CRCS: CRC size.
0 - CRC-CCITT (16 bits): $x^{16} + x^{12} + x^5 + 1$
1 - CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
If inverted (see CRCN bit) the CRC-16 check value is 0x1D0F and the CRC-32 check value is 0xC704_DD7B. Otherwise the check value is zero.

UART1HDLCAddMtchVal

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AMV															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMV															

Address:
0x808C_0210 - Read/Write

Default:
0x0000_0000

Definition:
HDLC Address Match Value

Bit Descriptions:

AMV: Address match value. Supports 8-bit and 16-bit address matching. If UART1HDLCCtrl.AME[1:0] is 00b or 11b, this register is not used.

UART1HDLCAddMask

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AMSK															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AMSK															

Address:

0x808C_0214 - Read/Write

Default:

0x0000_0000

Definition:

HDLC Address Mask

Bit Descriptions:

AMSK: Address mask value. Supports 8-bit and 16-bit address masking. If UART1HDLCCtrl.AME[1:0] is 00b or 11b, this register is not used.

UART1HDLCRXInfoBuf

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD														BPLLE	RSVD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	BC											BFRE	BROR	BCRE	BRAB

Address:

0x808C_0218 - Read Only

Default:

0x0000_0000

Definition:

HDLC Receive Information Buffer Register. This register is loaded when the last data byte in a received frame is read from the receive FIFO. The CPU has until the end of the next frame to read this register, or the RIL bit in the HDLC Status Register is set.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BPLLE: Buffered Digital PLL Error.
1 - Receiver aborted last frame because DPLL lost the carrier.
0 - Receiver did not abort because DPLL lost the carrier.
This bit is only valid when receiving Manchester-encoded synchronous HDLC.

BC: Received frame Byte Count.
The total number of valid bytes read from the RX FIFO during the last HDLC frame.

BFRE: Buffered Framing Error.
0 - No framing errors were encountered in the last frame.
1 - A framing error occurred during the last frame, causing the remainder of the frame to be discarded.

BROR: Buffered Receiver Over Run.
0 - The RX buffer did not overrun during the last frame.
1 - The receive FIFO did overrun during the last frame. The remainder of the frame was discarded.

BCRE: Buffered CRC Error.
0 - No CRC check errors occurred in the last frame.
1 - The CRC calculated on the incoming data did not match the CRC value contained in the last frame.

BRAB: Buffered Receiver Abort.
0 - No abort occurred in the last frame.
1 - The last frame was aborted.

UART1HDLCSts

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD														PLLE	PLLCC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LNKIDL	CRE	ROR	TBY	RIF	RSVD	RAB	RTO	EOF	RFL	RIL	RFC	RFS	TAB	TFC	TFS

Address:
0x808C_021C - Read/Write

Default:
0x0000_0000

Definition:
HDLC Status Register. The TFS and RFS bits in this register are replicas of bits in the UART status register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

PLLE: Digital PLL Error. (Read Only)
1 - A frame receive was aborted because the DPLL lost synchronization with the carrier.
0 - DPLL has not lost carrier during frame reception.
This bit is only valid when set up to receive Manchester-encoded synchronous HDLC.

Note: This bit reflects the status associated with the last character read from the RX FIFO. It changes with reads from the RX FIFO.

PLLCC: Digital PLL Carrier Sense. (Read Only)
1 - DPLL tacked onto a carrier.
0 - DPLL does not sense a carrier.

LNKIDL: Link Idle. (Read Only)
0 - RX data signal has changed within two bit periods
1 - RX data signal has not changed within two bit periods. This bit is only valid when set up to receive Manchester-encoded synchronous HDLC.

CRE: CRC Error. (Read Only)
0 - No CRC check errors encountered in incoming frame.
1 - CRC calculated on the incoming data does not match CRC value contained within the received frame. This bit is set with the last data in the incoming frame along with EOF.

Note: This bit reflects the status associated with the last character read from the RX FIFO. It changes with reads from the RX FIFO.

ROR: Receive FIFO Overrun. (Read Only)
0 - RX FIFO has not overrun.
1 - RX logic attempted to place data in the RX FIFO while it was full. The most recently read data is the last valid data before the overrun. The rest of the incoming frame is dropped. EOF is also set.

Note: This bit reflects the status associated with the last character read from the RX FIFO. It changes with reads from the RX FIFO.

TBY: Transmitter Busy. (Read Only)
0 - TX is idle, disabled, or transmitting an abort.
1 - TX is currently sending a frame (address, control, data, CRC or start/stop flag).

RIF: Receiver In Frame. (Read Only)
0 - RX is idle, disabled, or receiving start flags.
1 - RX is receiving a frame.

RAB: Receiver Abort. (Read Only)
0 - No abort has been detected for the incoming frame.
1 - Abort detected during receipt of incoming frame. The most recently read data is the last valid data before the abort. EOF is also set.

Note: This bit reflects the status associated with the last character read from the RX FIFO. It changes with reads from the RX FIFO.

RTO: Receiver Time Out.
Set to “1” whenever the HDLC RX has received four consecutive flags, or four character times of idle or space. Cleared by writing a “1” to this bit.

EOF: End of Frame (read only).
0 - Current frame has not been received completely.
1 - The data most recently read from the RX FIFO is the last byte of data within the frame.

Note: This bit reflects the status associated with the last character read from the RX FIFO. It changes with reads from the RX FIFO.

RFL: Receive Frame Lost. (Read/Write)
Set to “1” when an ROR occurred at the start of a new frame, before any data for the frame could be put into the RX FIFO. Cleared by writing a “1” to this bit.

RIL: Receive Information buffer Lost. (Read/Write)
Set to “1” when the last data for a frame is read from the RX FIFO and the UART1HDLCRXInfoBuf has not been read since the last data of the previous frame was read. That is, the information loaded into the UART1HDLCRXInfoBuf about the previous frame was never read and has been overwritten. Cleared by writing a “1” to this bit.

RFC: Received Frame Complete. (Read/Write)
Set to “1” when the last data byte for the frame is read from the RX FIFO (this also triggers an update of the UART1HDLCRXInfoBuf). Cleared by writing to a “1” to this bit.

RFS: Receive FIFO Service request. (Read Only)
This bit is a copy of the RIS bit in the UART interrupt identification register.
0 - RX FIFO is empty or RX is disabled.
1 - RX FIFO not empty and RX enabled.
May generate an interrupt and signal a DMA service request.

TAB: Transmitted Frame Aborted. (Read/Write)
Set “1” when a transmitted frame is terminated with an abort. Cleared by writing to a “1” to this bit.

TFC: Transmit Frame Complete. (Read/Write)
Set to “1” whenever a transmitted frame completes, whether terminated normally or aborted. Cleared by writing to a “1” to this bit.

TFS: Transmit FIFO Service request. (Read Only)
This bit is a copy of the TIS bit in the UART interrupt identification register.
0 - TX FIFO is full or TX disabled.
1 - TX FIFO not full and TX enabled. May generate an interrupt and signal a DMA service request.

11

This page intentionally blank.

Chapter 12

UART2

12

12.1 Introduction

UART2 implements a UART interface identical to that of UART1. UART2 does *not* implement a modem or HDLC interface. For additional details about UART1, refer to Chapter 11, “UART1 With HDLC and Modem Control Signals” on page 331.

UART2 and the IrDA blocks cooperatively implement a Slow Infrared (SIR) interface. The register interface for each block is separate. The UART2 control registers are at base address 0x808D_0000 and the IrDA controller registers are at base address 0x808B_0000. For additional details about IrDA, refer to Chapter 13, “IrDA” on page 387. The UART SIR interface is described below.

12.2 IrDA SIR Block

The IrDA SIR block contains an IrDA SIR protocol Encoder/decoder. The SIR protocol Encoder/decoder can be enabled for serial communication via signals **nSIROUT** and **SIRIN** to an infrared transducer instead of using the UART signals **UARTTXD** and **UARTRXD**.

If the SIR protocol Encoder/decoder is enabled, the **UARTTXD** line is held in the passive state (HIGH) and transitions of the modem status or the **UARTRXD** line will have no effect. The SIR protocol Encoder/decoder can both receive and transmit, but it is half-duplex only, so it cannot receive while transmitting, or vice versa.

The IrDA SIR physical layer specifies a minimum 10 ms delay between transmission and reception.

12.2.1 IrDA SIR Encoder/decoder Functional Description

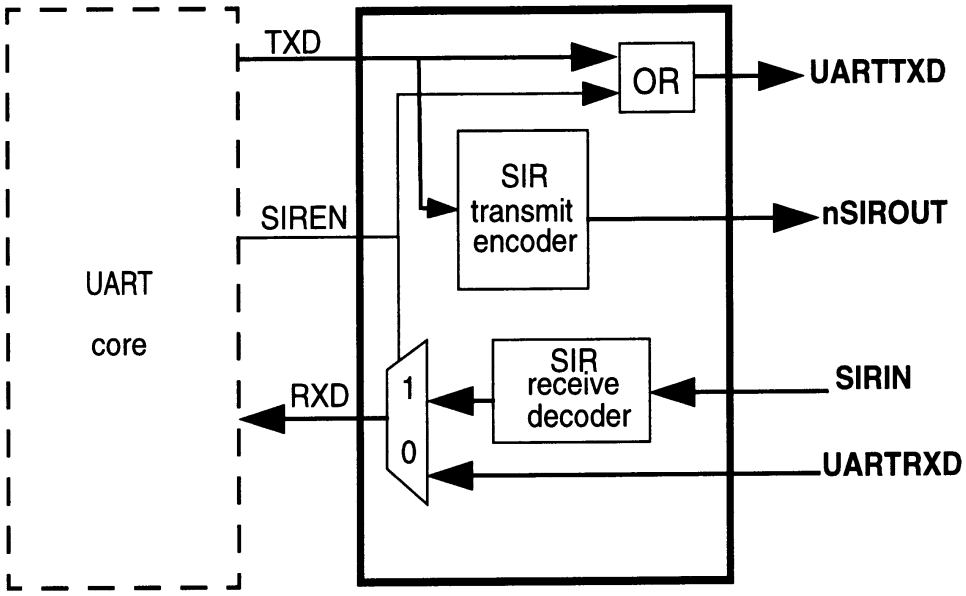
The IrDA SIR Encoder/decoder comprises:

- IrDA SIR transmit encoder
- IrDA SIR receive decoder

This is shown in Figure 12-1, below.

12

Figure 12-1. IrDA SIR Encoder/decoder Block Diagram



12.2.1.1 IrDA SIR Transmit Encoder

The SIR transmit encoder modulates the Non Return-to-Zero (NRZ) transmit bit stream output from the UART. The IrDA SIR physical layer specifies use of a Return To Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infrared light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared Light Emitting Diode (LED).

In normal mode, the transmitted pulse width is specified as three times the period of the internal x16 clock (Baud16), that is, 3/16 of a bit period.

In low-power mode, the transmit pulse width is specified as 3/16 of a 115.2 Kbits/s bit period. This is implemented as three times the period of a nominal 1.8432 MHz clock (IrLPBaud16) derived by dividing down the UARTCLK clock. The frequency of IrLPBaud16 is set up by writing the appropriate divisor value to UARTILPR. The active low encoder output is normally LOW for the marking state (no light pulse). The encoder outputs a high pulse to generate a infrared light pulse representing a logic “0” or spacing state.

12.2.1.2 IrDA SIR Receive Decoder

The SIR receive decoder demodulates the return-to-zero bit stream from the infrared detector and outputs the received NRZ serial bit stream to the UART received data input. The decoder input is normally HIGH (marking state) in the idle state (the transmit encoder output has the opposite polarity to the decoder input).

A start bit is detected when the decoder input is LOW.

Regardless of being in normal or low-power mode, a start bit is deemed valid if the decoder is still LOW, one period of IrLPBaud16 after the LOW was first detected. This allows a normal-mode UART to receive data from a low-power mode UART, which may transmit pulses as small as 1.41 µsec.

12.2.2 IrDA SIR Operation

The IrDA SIR Encoder/decoder provides functionality which converts between an asynchronous UART data stream and half-duplex serial SIR interface. No analog processing is performed on-chip. The role of the SIR encoder/decoder is only to provide a digital encoded output and decoded input to the UART. There are two modes of operation:

- In normal IrDA mode, a zero logic level is transmitted as high pulse of $3/16^{th}$ duration of the selected baud rate bit period on the **nSIROUT** signal, while logic one levels are transmitted as a static LOW signal. These levels control the driver of an infrared transmitter, sending a pulse of light for each zero. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling its output LOW. This then drives the **SIRIN** signal LOW.
- In low-power IrDA mode, the width of the transmitted infrared pulse is set to 3 times the period of the internally generated IrLPBaud16 signal (1.63 ns assuming a nominal 1.8432MHz frequency) by changing the appropriate bit in UARTCR.

In both normal and low-power IrDA modes, during transmission, the UART data bit is used as the base for encoding, while during reception the decoded bits are transferred to the UART receive logic.

The IrDA SIR physical layer specifies a half duplex communication link with a minimum 10ms delay between transmission and reception. This delay must be generated by software since it is not supported by the UART. The delay is required since the Infrared receiver electronics may become biased or even saturated from the optical power coupled from the adjacent transmitter LED. This delay is known as latency or receiver setup time. Shorter delays may be able to be used when the link first starts up.

The IrLPBaud16 signal is generated by dividing down the UARTCLK signal according to the low-power divisor value written to UARTILPR.

The low-power divisor value is calculated as:

$$\text{Low-power divisor} = (\text{FUARTCLK} / \text{FirLPBaud16}) - 1$$

where FirLPBaud16 is nominally 1.8432 MHz.

The divisor must be chosen so that 1.42 MHz < IrLPBaud16 < 2.12 MHz.

12

12.2.2.1 System/diagnostic Loopback Testing

It is possible to perform loopback testing for SIR data by setting the Loop Back Enable (LBE) bit to 1 in the control register UARTCR (bit 7), and setting the SIRTEST bit to 1 in the test register UARTTMR (bit 1).

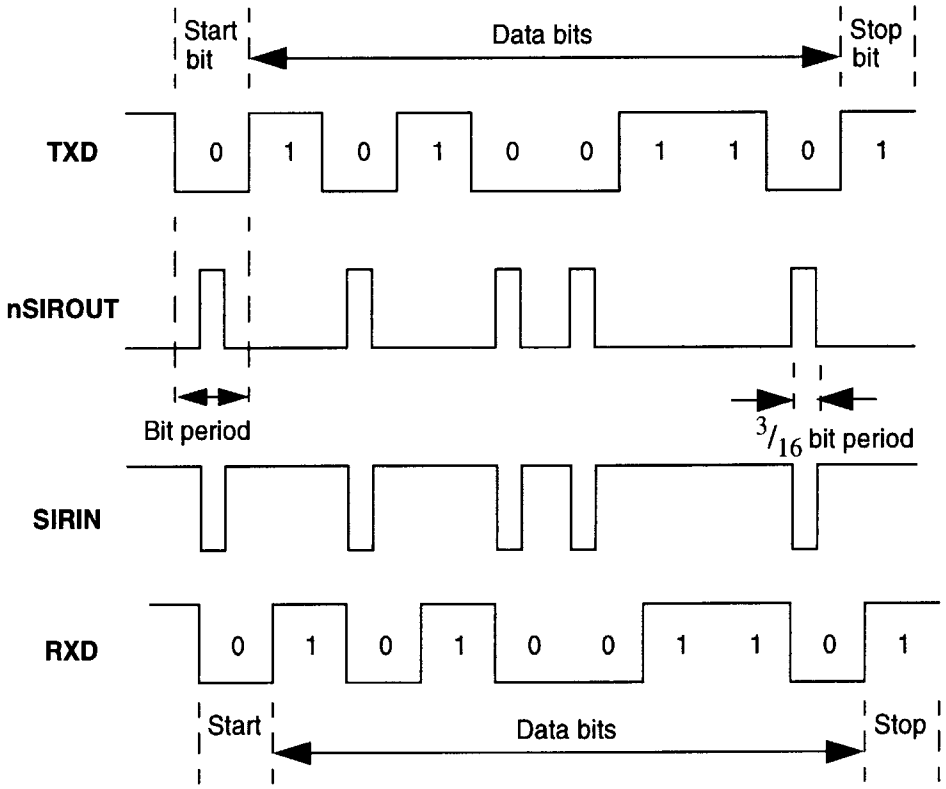
Data transmitted on nSIROUT will be received on the SIRIN input.

Note: UART2TMR is the only occasion that a test register needs to be accessed during normal operation.

12.2.3 IrDA Data Modulation

The effect of IrDA 3/16 data modulation can be seen in Figure 12-2, below.

Figure 12-2. IrDA Data Modulation (3/16)



12.2.4 Enabling Infrared (Ir) Modes

Table 12-1: UART2 / IrDA Modes

Mode	DeviceCfg Register		UART2Ctrl Register		IrEnable Register	
	U2EN	IonU2	SirEn	UARTE	EN[1]	EN[0]
Disabled	0	x	0	0	0	0
UART2	1	0	0	1	0	0
SIR	1	1	1	1	0	1
MIR	x	1	0	0	1	0
FIR	x	1	0	0	1	1

12.3 UART2 Package Dependency

UART2 uses package pins **RXD1** and **TXD1**. Pin **RXD1** drives both the UART2 UART input and the UART2 SIR input.

However, Syscon register DeviceCfg[28] (IonU2) controls what drives pin **TXD1**. See Table 12-2.

Table 12-2: IonU2 Pin Function

IonU2	Pin TXD1 Function
0	UART2 UART is the output signal
1	Logical OR of IrDA output signal and UART2 SIR output signal

Therefore, to use any IrDA mode, FIR, MIR or SIR, set IonU2. To use UART2 as a UART, clear IonU2.

12.3.1 Clocking Requirements

There are two clocks, PCLK and UARTCLK.

UARTCLK frequency must accommodate the desired range of baud rates:

Fuartclk(min) >= 32 x baud_rate(max)

Fuartclk(max) <= 32 x 65,536 x baud_rate(min)

The frequency of UARTCLK must also be within the required error limits for all baud rates to be used.

To allow sufficient time to write the received data to the receive FIFO, UARTCLK must be less than or equal to four times the frequency of PCLK:

Fuartclk <= 4 x Fpclk

If the IrDA SIR functionality is required, UARTCLK must have a frequency between 2.7 MHz and 542.7 MHz to ensure that the low-power mode transmit pulse duration complies with the IrDA SIR specification.

12.3.2 Bus Bandwidth Requirements

There are two basic ways of moving data to and from the UART FIFOs:

- Direct DMA interface - this permits byte-wide access to the UART without using the APB. The DMA block will pack/unpack individual bytes so that it reads or writes full 32-bit words rather than individual bytes.
- Accessing the UART via the APB - this requires APB/AHB bus bandwidth. Then, both a read and write are required for each 8-bit data byte.

Bandwidth requirements also depend on the selected baud rate, character size, parity selection, number of stop bits, and spacing between characters (if receiving).

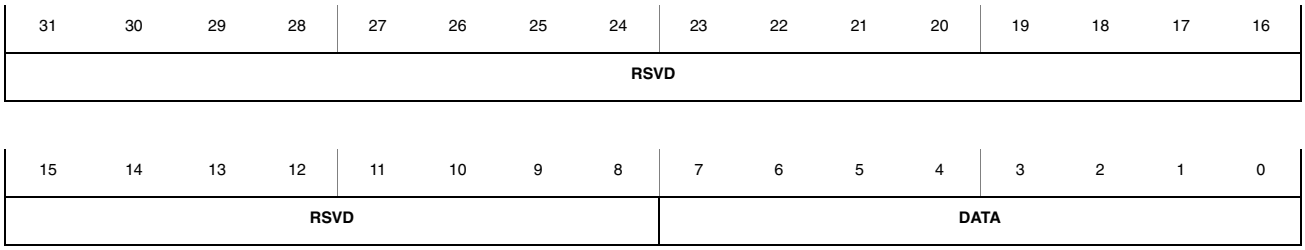
For example, assume 115,200 baud, 8-bit characters, even parity, one stop bit, no space between characters. There are 11 bits per character, so 115,200 / 11 = 10473 characters per second. If both transmitting and receiving, 20,945 characters per second pass through the UART. Accessing the UART through the DMA interface requires one access per 32 bits, implying only 20,945 / 4 = 5,236 AHB accesses per second. Accessing the UART through the APB requires two accesses per byte, implying 20,945 APB bus accesses.

As another example, assume 230,400 baud (the maximum with a UARTCLK equal to 7.3728 Mhz), 5-bit characters, no parity, one stop bit, and no space between characters. There are 7 bits per character, so 230400 / 7 = 32,914 characters per second. Simultaneous transmitting and receiving implies 65829 APB characters per second. Using the DMA interface would result in 16457 AHB accesses per second, while using the APB to access the UART leads to 65829 bus accesses per second.

12.4 Registers

Register Descriptions

UART2Data



Address:

0x808D_0000 - Read/Write

Default:

0x0000_0000

Definition:

UART Data Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DATA: UART Data, read for receive data, write for transmit data
For words to be transmitted:
• if the FIFOs are enabled, data written to this location is pushed onto the transmit FIFO
• if the FIFOs are not enabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). The write operation initiates transmission from the UART. The data is prefixed with a start bit, appended with the appropriate parity bit (if parity is enabled), and a stop bit. The resultant word is then transmitted.
For received words:
• if the FIFOs are enabled, the data byte is extracted, and a 3-bit status (break, frame and parity) is pushed onto the 11-bit wide receive FIFO
• if the FIFOs are not enabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO).

UART2RXSts

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												OE	BE	PE	FE

Address:

0x808D_0004 - Read/Write

Default:

0x0000_0000

Definition:

UART Receive Status Register and Error Clear Register. Provides receive status of the data value last read from the UART2Data. A write to this register clears the framing, parity, break and overrun errors. The data value is not important.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

OE: Overrun Error. This bit is set to “1” if data is received and the FIFO is already full. This bit is cleared to 0 by a write to UART2RXSts. The FIFO contents remain valid since no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The CPU must now read the data in order to empty the FIFO.

BE: Break Error. This bit is set to “1” if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). This bit is cleared to 0 after a write to UART2RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a “1” (marking state) and the next valid start bit is received.

PE: Parity Error. When this bit is set to “1”, it indicates that the parity of the received data character does not match the parity selected in UART2LinCtrlHigh (bit 2). This bit is cleared to 0 by a write to UART2RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO.

FE: Framing Error. When this bit is set to “1”, it indicates that the received character did not have a valid stop bit (a valid stop bit is “1”). This bit is cleared to 0 by a write to UART2RXSts. In FIFO mode, this error is associated with the character at the top of the FIFO.

UART2LinCtrlHigh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									WLEN		FEN	STP2	EPS	PEN	BRK

Address:

0x808D_0008 - Read/Write

Default:

0x0000_0000

Definition:

UART - High. UART2LinCtrlHigh, UART2LinCtrlMid and UART2LinCtrlLow form a single 23-bit wide register (UART2LinCtrl) which is updated on a single write strobe generated by an UART2LinCtrlHigh write. So, in order to internally update the contents of UART2LinCtrlMid or UART2LinCtrlLow, a UART2LinCtrlHigh write must always be performed at the end.

To update the three registers there are two possible sequences:
• UART2LinCtrlLow write, UART2LinCtrlMid write and UART2LinCtrlHigh write
• UART2LinCtrlMid write, UART2LinCtrlLow write and UART2LinCtrlHigh write.

To update UART2LinCtrlLow or UART2LinCtrlMid only:
• UART2LinCtrlLow write (or UART2LinCtrlMid write) and UART2LinCtrlHigh write.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

WLEN: Number of bits per frame:
11 = 8 bits
10 = 7 bits
01 = 6 bits
00 = 5 bits

12

FEN: FIFO Enable.
1 - Transmit and receive FIFO buffers are enabled (FIFO mode).
0 - The FIFOs are disabled (character mode). (That is, the FIFOs become 1-byte-deep holding registers.)

STP2: Two Stop Bits Select.
1 - Two stop bits are transmitted at the end of the frame.
0 - One stop bit is transmitted at the end of the frame.
The receive logic does not check for two stop bits being received.

EPS: Even Parity Select.
1 - Even parity generation and checking is performed during transmission and reception (this checks for an even number of “1”s in data and parity bits).
0 - Odd parity is performed (this checks for an odd number of “1”s).
This bit has no effect when parity is disabled by Parity Enable (bit 1) being cleared to 0.

PEN: Parity Enable.
1 - Parity checking and generation is enabled,
0 - Parity checking is disabled and no parity bit added to the data frame.

BRK: Send Break.
1 - A low level is continually output on the **UARTTXD** output, after completing transmission of the current character. This bit must be asserted for at least one complete frame transmission time in order to generate a break condition. The transmit FIFO contents remain unaffected during a break condition.
0 - For normal use, this bit must be cleared.

UART2LinCtrlMid

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								BR							

Address:
0x808D_000C - Read/Write

Default:
0x0000_0000

Definition:

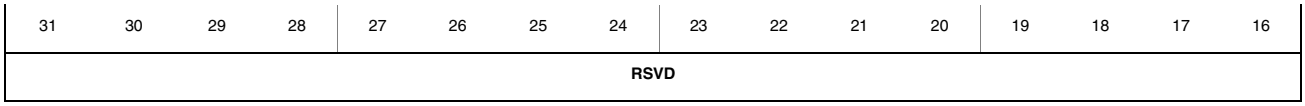
UART Line Control Register Middle.

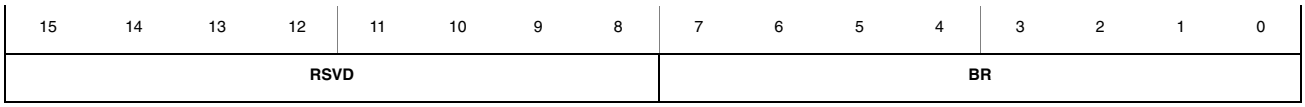
Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BR: Baud Rate Divisor bits [15:8]. Most significant byte of baud rate divisor. These bits are cleared to 0 on reset.

UART2LinCtrlLow





Address:

0x808D_0010 - Read/Write

Default:

0x0000_0000

Definition:

UART Line Control Register Low.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BR: Baud Rate Divisor bits [7:0]. Least significant byte of baud rate divisor. These bits are cleared to 0 on reset. The baud rate divisor is calculated as follows:

Baud rate divisor BAUDDIV = ($F_{UARTCLK}$ / (16 * Baud rate)) –1

where $F_{UARTCLK}$ is the UART reference clock frequency. A baud rate divisor of zero is not allowed and will result in no data transfer.

UART2Ctrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								LBE	RTIE	TIE	RIE	MSIE	SIRLP	SIREN	UARTE

Address:

0x808D_0014 - Read/Write

Default:

0x0000_0000

Definition:

UART Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

LBE: Loopback Enable, for SIR and UART only.
1 - If the SIR Enable bit is also set to “1”, and register UART2TMR bit 1 (SIRTEST) is set to “1”, the SIR output path is inverted and fed through to the SIR input path. The SIRTEST bit in the test register must be set to “1” to override the normal half-duplex SIR operation. This should be the requirement for accessing the test registers during normal operation, and SIRTEST must be cleared to “0” when loopback testing is finished. This feature reduces the amount of external coupling required during system test.
0 - This bit is cleared to “0” on reset, which disables the loopback mode.

RTIE: Receive Timeout Enable. If this bit is set to “1”, the receive timeout interrupt is enabled.

TIE: Transmit Interrupt Enable. If this bit is set to “1”, the transmit interrupt is enabled.

RIE: Receive Interrupt Enable. If this bit is set to “1”, the receive interrupt is enabled.

MSIE: Modem Status Interrupt Enable. If this bit is set to “1”, the modem status interrupt is enabled.

SIRLP: SIR Low Power Mode. This bit selects the IrDA encoding mode. If this bit is cleared to 0, low level bits are transmitted as an active high pulse with a width of 3/16th of the bit period. If this bit is set to “1”, low level bits are transmitted with a pulse width which is 3 times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. Setting this bit uses less power, but may reduce transmission distances.

SIREN: SIR Enable. If this bit is set to “1”, the IrDA SIR encoder/decoder is enabled. This bit has no effect if the UART is not enabled by bit 0 being set to “1”. When the IrDA SIR encoder/decoder is enabled, data is transmitted and received on **nSIROUT** and **SIRIN**. **UARTTXD** remains in the marking state (set to “1”). Signal transitions on **UARTRXD** or modem status inputs will have no effect. When the IrDA SIR encoder/decoder is disabled, **nSIROUT** remains cleared to 0 (no light pulse generated), and signal transitions on **SIRIN** will have no effect.

UARTE: UART Enable. If this bit is set to “1”, the UART is enabled. Data transmission and reception occurs for UART signals.

UART2Flag

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS

Address:

0x808D_0018 - Read/Write

Default:

0x0000_0000

Definition:

UART Flag Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

TXFE: Transmit FIFO Empty. The meaning of this bit depends on the state of the FEN bit in the UART2LinCtrlHigh register. If the FIFO is disabled, this bit is set when the transmit holding register is empty. If the FIFO is enabled, the TXFE bit is set when the transmit FIFO is empty.

12

RXFF: Receive FIFO Full. The meaning of this bit depends on the state of the FEN bit in the UART2LinCtrlHigh register. If the FIFO is disabled, this bit is set when the receive holding register is full. If the FIFO is enabled, the RXFF bit is set when the receive FIFO is full.

TXFF: Transmit FIFO Full. The meaning of this bit depends on the state of the FEN bit in the UART2LinCtrlHigh register. If the FIFO is disabled, this bit is set when the transmit holding register is full. If the FIFO is enabled, the TXFF bit is set when the transmit FIFO is full.

RXFE: Receive FIFO Empty. The meaning of this bit depends on the state of the FEN bit in the UART2LinCtrlHigh register. If the FIFO is disabled, this bit is set when the receive holding register is empty. If the FIFO is enabled, the RXFE bit is set when the receive FIFO is empty.

BUSY: UART Busy. If this bit is set to “1”, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether the UART is enabled or not).

DCD: Data Carrier Detect status. This bit is the complement of the UART data carrier detect (**nUARTDCD**) modem status input. That is, the bit is “1” when the modem status input is 0.

DSR: Data Set Ready status. This bit is the complement of the UART data set ready (**nUARTDSR**) modem status input. That is, the bit is “1” when the modem status input is 0.

CTS: Clear To Send status. This bit is the complement of the UART clear to send (**nUARTCTS**) modem status input. That is, the bit is “1” when the modem status input is 0.

UART2IntIDIntClr

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												RTIS	TIS	RIS	MIS

Address:

0x808D_001C

Default:

0x0000_0000

Definition:

UART Interrupt Identification and Interrupt Clear Register. Interrupt status is read from UART2IntIDIntClr. A write to UART2IntIDIntClr clears the modem status interrupt. All the bits are cleared to 0 when reset.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

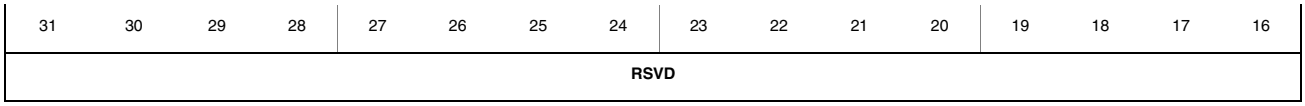
RTIS: Receive Timeout Interrupt Status. This bit is set to “1” if the receive timeout interrupt is asserted.

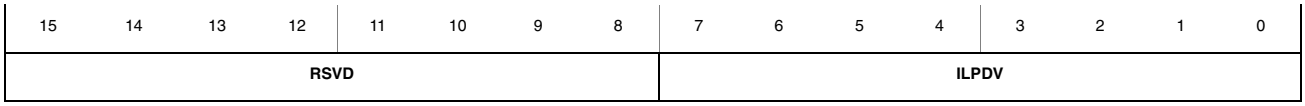
TIS: Transmit Interrupt Status. This bit is set to “1” if the transmit interrupt is asserted.

RIS: Receive Interrupt Status. This bit is set to “1” if the receive interrupt is asserted.

MIS: Modem Interrupt Status. This bit is set to “1” if the modem status interrupt is asserted.

UART2IrLowPwrCntr





Address:

0x808D_0020 - Read/Write

Default:

0x0000_0000

Definition:

UART IrDA Low Power Divisor Register. This is an 8-bit read/write register that stores the low-power counter divisor value used to generate the **IrLPBaud16** signal by dividing down of UARTCLK. All the bits are cleared to 0 when reset.

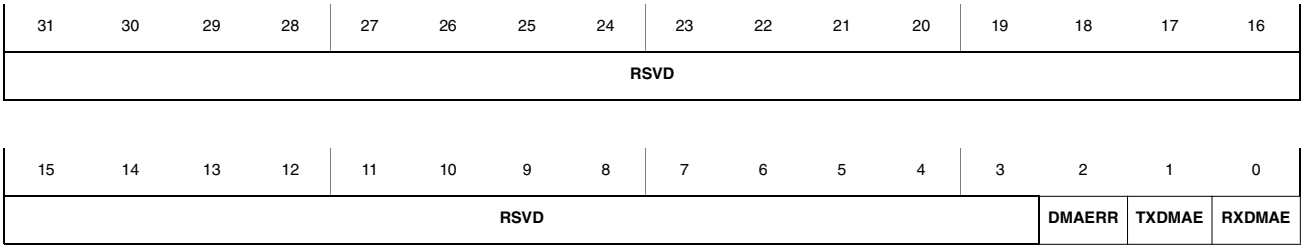
Bit Descriptions:

RSVD: Reserved. Unknown During Read.

12

ILPDV: IrDA Low Power Divisor bits [7:0]. 8-bit low-power divisor value. These bits are cleared to 0 at reset. The divisor must be chosen so that the relationship 1.42 MHz < **IrLPBaud16** < 2.12 MHz is maintained, which results in a low power pulse duration of 1.41–2.11 µs (three times the period of **IrLPBaud16**). The minimum frequency of **IrLPBaud16** ensures that pulses less than one period of **IrLPBaud16** are rejected, but that pulses greater than 1.4 µs are accepted as valid pulses. **Zero is an illegal value. Programming a zero value will result in no IrLPBaud16 pulses being generated.**

UART2DMACtrl



Address:

0x808D_0028 - Read/Write

Default:

0x0000_0000

Definition:

UART DMA Control Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DMAERR: RX DMA error handing enable. If 0, the RX DMA interface ignores error conditions in the UART receive section. If “1”, the DMA interface stops and notifies the DMA block when an error occurs. Errors include break errors, parity errors, and framing errors.

TXDMAE: TX DMA interface enable. Setting to “1” enables the private DMA interface to the transmit FIFO.

RXDMAE: RX DMA interface enable. Setting to “1” enables the private DMA interface to the receive FIFO.

UART2TMR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								0						SIRTEST	0

Address:

0x808D_0084 - Read/Write

Default:

0x0000_0000

Definition:

UART SIR Loopback Register

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

0: Must be written as “0”. Unknown During Read.

SIRTEST: SIR test enable. Setting this bit to “1” enables the receive data path during IrDA transmission (testing requires SIR to be configured in full-duplex mode). This bit must be set to “1” to enable SIR system loopback testing, when the normal mode control register UART2Ctrl bit 7, Loop Back Enable (LBE), has been set to “1”. Clearing this bit to 0 disabled the receive logic when the SIR is transmitting (normal operation). This bit defaults to 0 for normal (half-duplex) operation.

12

This page intentionally blank.

Chapter 13

IrDA

13.1 Introduction

This module implements the physical layer of an infrared serial port that is compliant with Version 1.1 of the Infrared Data Association (IrDA) standard. It supports communication speeds of up to 4 MBit/s. When combined with analog transducer components, it provides a complete interface between infrared media and an AMBA compliant peripheral bus (APB).

Three different encoder/decoder units implement the supported modulation schemes and data encoding systems defined by the IrDA standard:

- Slow Infrared (SIR) - This interface attaches to the output of UART2. The UART2 registers handle the data and control for this interface, though the IrDA interface enable register selects the SIR function.
- Medium Infrared (MIR) - Transmission/reception rates can be 0.576 or 1.152 Mb/s.
- Fast Infrared (FIR) - Transmission/reception rate is 4 Mb/s.

13.2 IrDA Interfaces

The Infrared Interface Module implements in hardware the physical layer of an infrared serial port, compliant with version 1.1 of the IrDA standard. Communication speeds of up to 4 Mbit/sec are supported. When combined with analog transducer components, it provides a complete interface between infrared media and an AMBA compliant peripheral bus (APB).

The Module comprises three separate encoder/decoder units for implementing three different combinations of modulation scheme and data encoding system defined by the IrDA standard. These are:

- Slow Infrared - SIR - This interface attaches to the output of a UART. All data and control for this interface is done through the UART registers. The SIR encoder function is selected using the IrDA interface enable register.
- Medium Infrared - MIR - This interface is independent of a UART. Transmission/reception rates can be 0.576 or 1.152 Mbit/sec.
- Fast Infrared - FIR - This interface is independent of a UART. Transmission/reception rates can be 4 Mbit/sec.

13.3 Shared IrDA Interface Feature

This section describes features common to the MIR and FIR interfaces (the SIR interface has been designed to share the enable register and device pins but is otherwise a separate interface assumed to be controlled by UART2).

13.3.1 Overview

The Slow Infrared (SIR) Encoder/Decoder is used to modulate and demodulate serial data using the Hewlett-Packard Serial Infrared standard (HP-SIR) for bit encoding. Serial transmit data from UART2 is modulated using return-to-zero (RTZ) encoding to produce an output to drive the Ir transmitter LED, while data received from the Ir detector is converted into a serial bit stream to drive a UART's serial input. The SIR supports data rates up to 115.2 kbit/s.

The Medium Speed Infrared (MIR) Encoder/Decoder encodes/decodes peripheral bus data according to a modified HDLC standard, using flag characters, bit stuffing and a 16 bit CRC checker. MIR uses the same RTZ modulation and demodulation scheme used by the SIR. Two signal bit rates are supported: 0.576 Mbit/s and 1.152 Mbit/s.

The Fast Infrared Encoder/Decoder (FIR) operates at a fixed bit rate of 4 Mbit/s. Modulation/demodulation is by a phase shift key scheme called pulse position modulation (4 PPM). One of four signalling symbols represent each possible pair of data bits. Data encoding uses a packet format that prefixes bit and symbol synchronization flags to data and appends a 32-bit CRC and stop flag to the end of each packet. The start and stop flags use signalling symbols that are not used to encode data, hence bit stuffing of data is not required in this mode.

Only one of the Encoder/Decoder modules can be enabled to transmit and receive data from the IrDA transducers at one time Selection of an Ir sub-module is by means of the IrEnable register. The MIR and FIR sub-modules can be regarded by programmers as independent entities which are operated using common control and data registers, but which report status data via separate read registers.

Detailed descriptions of the MIR and FIR are given in the following sections. The SIR, however, has no data or control registers. It interfaces directly to a UART's serial stream. With the exception of the IrEnable register, it has no presence on the memory map and has no interface to the APB via the Infrared interface.

13.3.2 Functional Description

This section gives a programmer's guide to operating the IrDA interface. It includes detail on the general configuration and the transmit and receive processes.

13.3.2.1 General Configuration

13.3.2.1.1 Select Ir Mode

The IrEnable register selects which of the three Ir sub-modules is used to operate the IrDA interface. Only one of the three may be active at any one time. The reset value for this register is zero, which disables all three encoder/decoder modules. The bottom two bits of this register select the encoder/decoder module according to the tabulated values:

Table 13-1: Bit Values to Select Ir Module

IrEnable EN1	IrEnable EN0	Encoder Selected
0	0	None
0	1	SIR
1	0	MIR
1	1	FIR

SIR does not use the data transfer mechanism described in this section. After selecting SIR mode, all data transfer operations are made through a UART, as if connection is through a serial cable without handshake lines. The features described below are implemented for the MIR and FIR modes.

13.3.2.1.2 Select Data Rate

The data rates for MIR and FIR are as follows:

- MIR - Clear BRD bit in IrControl (IrCon) for 0.576 Mbit/sec, Set BRD bit in IrCon for 1.152 Mbit/sec.
- FIR - Fixed at 4 Mbit/sec.

13.3.2.2 Transmitting Data

13.3.2.2.1 Initialization

The principal method of data transfer from memory to the active IrDA encoder (MIR or FIR) is by DMA. Typically DMA can be used to transfer data of any length into the transmit FIFO when requested by the infrared peripheral. When polling or interrupts are used to perform the data transfer, a mechanism exists for transmitting data packets that are not a multiple of 4 bytes in length. This uses a register called IrDataTail and its use is described in the next section.

The DMA route is usually provided to overcome any large interrupt response times that may exist in the SoC where the Infrared module is going to be used. These large interrupt response times can make programmed I/O an impractical method for transferring large Ir data packets.

13.3.2.2.2 The Transmit Process

This section describes the transmission process in detail.

Is last transmission complete? - Ensure that the Infrared peripheral is not currently receiving or transmitting data by reading the RSY (for half-duplex communications) and TBY bits in the IrFlag register. If either is set, postpone the start of transmission.

Disable IrDA - If you are changing Ir mode, first disable Ir. To disable IrDA, first clear IrCtrl.RXE and IrCtrl.TXE. Secondly, clear the IrEnable.EN field to be “00”.

Disabling UART2 for MIR and FIR - For MIR and FIR, disable UART2 by writing “0” to UART2Ctrl and 0 to IrCtrl.

Set up the DMA Engine - If DMA is being used, set up the DMA engine by setting up the registers of the DMA block.

Enabling Clocks - For MIR, set up the MIR clock in MIRClkDiv. Select 0.576 or 1.152 Mbps mode by clearing or setting IrCtrl.BRD. For FIR, enable the FIR clock by setting PwrCnt.FIR_EN.

Select Ir Mode - Select SIR, MIR, or FIR mode by writing the IrEnable.EN bit field to be “01”, “10”, or “11”.

Clear Interrupt Sticky Bits - For MIR, write the MISR register, setting the TFC, TAB, RFL, and RIL bits to clear them. Then read the IrRIB register to clear the RFC bit. For FIR, write the FISR register, setting the TFC, TAB, RFL, and RIL bits to clear them. Then read the IrRIB register to clear the RFC bit.

Select Transmit Underrun Action - When DMA is used, the TUS bit should be cleared.

Enable Transmit - Set the IrCtrl.TXE Transmit Enable bit. Also set IrCtrl.RXE if receive is to be enabled. If DMA is used, also set IrDMACR.TXDMAE (and IrDMACR.RXDMAE if receive is to be enabled).

Preloading the Transmit FIFO - Copy the first two full words of data into the transmit FIFO by writing them into the IrData register. The Ir encode block can hold up to 11 bytes of data (two words in the FIFO plus up to three bytes in the IrDataTail register). If this is sufficient to hold the complete transmission data packet, DMA will not be needed. The IrCon.TUS bit should be cleared. This will cause the Ir encoder to correctly send the CRC and end of frame flag. **Note:** Prefilling the FIFO must happen immediately after enabling MIR or FIR. Preloading the FIFO is unnecessary for SIR. Also note that preloading the FIFO is unnecessary for MIR and FIR if DMA is used.

Loading the IrDataTail Register - In the PIO and IRQ case, once the FIFO has been preloaded, the IrDataTail register can be loaded. The IrDataTail register contains the last bytes in the frame (1, 2 or 3 bytes left over from the last whole word provided by PIO or IRQ). **Note:** If DMA is used, loading the IrDataTail register is unnecessary, as the IrDataTail register is disabled in that case.

Send out the data - If DMA is being used, everything is now enabled for the transmission process to begin. If PIO or IRQ is being used, data should be written to the IrData register.

13.3.2.2.3 Sending Packets Which are Not a Multiple of 4 Bytes In Length

The transmit FIFO is 32 bits wide. When using polling or interrupts to effect the transfer, loading the FIFO with less than 32 bits would cause extraneous zero bits to be transmitted. This is taken care of automatically by DMA and needs no special action in that case. However in the case of polling or interrupt-driven transfers, the IrDataTail register is the mechanism used to preload the last 1, 2 or 3 bytes of a frame. When the transfer is complete and the FIFO is empty, any bytes stored in the IrDataTail register are transmitted before the Ir encoder sends the CRC and end-of-frame flags. There are three distinct addresses to write the end of frame data to. This allows a single word write to specify the data to be transmitted and the number of trailing bytes to send.

Table 13-2: Address Offsets for End-of-frame Data

Bytes to transmit	Address offset to use
1	0x014
2	0x018
3	0x01C

If there is a single trailing byte to transmit, write to address offset 0x014, for two bytes write to 0x018 and if there are three trailing bytes write to 0x01 C.

13.3.2.2.4 End of Frame Interrupt

Once all the data sent to the FIFO has been taken by the Ir interface, the FIFO will underrun. When this occurs any data that has been preloaded into the IrDataTail register will be used and the Transmitted Frame Complete (TFC) interrupt will be generated.

13.3.2.2.5 Disable Transmit Circuitry

To save power, the Transmit Enable (TXE) bit can be cleared in the IrEnable register if there are no frames that need to be sent.

13.3.2.2.6 Error conditions

Transmitted frame abort is only signalled if IrCon register bit TUS is set to 1.

13.3.2.3 Receiving Data

The end of a reception frame will cause an interrupt, which may be masked using the mask register (MIMR/FIMR). The end of frame interrupt occurs after the last data value has been transferred, including any odd bytes in the frame tail.

13.3.2.3.1 Initialization

Address Matching	To use Address Match filtering, set the local 8 bit address in the Address Match Value Register and set the Address Match Enable bit in the IrCon register.
Set up DMA	Set up a DMA buffer (the buffer should be greater than twice the maximum possible size of received frames). Enable DMA. Alternatively, two buffers maybe used which are each the maximum possible frame size long. The DMA would then be programmed to switch between the two buffers.
Enable Ir Receive	Set the Receive Enable bit (RXE) in IrEnable.

13.3.2.3.2 End of Frame Interrupt

The Receive Frame Complete (RFC) interrupt is generated when the last data in a frame is read from the receive FIFO. To check whether the frame was received correctly (no errors) and for information on frame size the Receive Information Buffer register (IrRIB) must be read by the interrupt service routine (this also clears the RFC interrupt condition).

Note: By the time the processor responds to this interrupt the interface may have already started reception of a new frame.

13.3.2.3.3 End of Frame: Using Programmed I/O

If interrupt driven programmed I/O is used instead of DMA, every time the Receive Buffer Service (RFS) interrupt is serviced the IrFlag register must be read before the IrData register, if the IrFlag values are needed. Their Flag register gives information about error conditions that correspond to the data value at the head of the receive FIFO.

Note: The IrRIB registers stores status flags for a complete frame.

13.3.2.3.4 Error Conditions

Receive error conditions do not generate interrupts. Reading the IrData word clears the IrFlag register bits listed below.

Receiver Abort Detected When set, this indicates that the transmitter sent an abort signal during frame transmission.

Receiver Overrun This indicates that data has not been read for the IrData register in time and has resulted in data loss from the frame. When this occurs the interface automatically discards the remainder of the incoming frame.

CRC Error If the CRC for the received data does not match the CRC value contained in the incoming data stream this condition will occur.

Frame Error (FIR only) This indicates that a framing error has been detected.

The data word and flags are held in the 39-bit wide receiver FIFO. Reading an IrData word removes both the data and its associated flag bits from the FIFO causing the next word in the FIFO (if present) to be transferred into the IrFlag and data registers. However, all error conditions encountered during a frame are remembered. At the end of frame they can be read form the IrRIB register.

When a receive overrun (ROR) or FIR framing error (FRE) is detected the remainder of the frame will be discarded by the receive logic (not put into the receive FIFO). In the case of receive overruns, if the end of frame (EOF) bit in the last entry in the FIFO is clear then the Receive Buffer Overrun (ROR) and EOF bits will be set. If an overrun occurs and the last entry in the FIFO already has the EOF bit set then the RFL interrupt will be triggered. In the case of a framing error an extra entry will be put into the FIFO with FIR Framing Error (FRE) and EOF set, this entry will not contain any valid data.

If programmed IO is used to service the IrDA interface instead of DMA a similar process occurs. Interrupt requests to service the receive FIFO will not occur until the rest of the frame has been discarded.

At the end of a frame, a valid end of frame (EOF) or an abort (RAB), a DMA request corresponding to the last word (which may hold 1, 2, 3, or 4 bytes of valid data) of the received frame will be raised. DMA will take the word. At that point the receive FIFO should be empty and the DMA request may be deasserted. The DMA request will be reasserted when data for a following frame is loaded into the receive FIFO.

The above behavior means there is no need for processor intervention to service the IrDA interface between successive receive frames.

13.3.2.4 Special Conditions

13.3.2.4.1 Early Termination of Transmission

Clearing IrCon.TXE (transmit enable bit) stops transmission immediately. All data within the FIFO, transmit buffer and serial output shifter is cleared.

13.3.2.4.2 Early Termination of Reception

Clearing IrCon.RXE receive enable bit stops reception immediately. All data within the receive buffer, serial input shifter and FIFO is cleared.

13.3.2.4.3 Changing IrDA Mode

Poll the Transmitter Disabled bits – FD or MD bits – in IrEnable register until end of transmission is indicated. The new mode can then be set as described in *4.2.1General Configuration*.

13.3.2.4.4 Loopback Mode

For test purposes, data will be looped back – internally – from the output of the transmit serial shifter into the input of the receive serial shifter when IrEnable.LBM is set.

13.3.3 Control Information Buffering

The processor needs several items of information about a received frame that are not held in data DMAed from the receive FIFO, or stored in the DMA controller itself (because the DMA unit may be receiving the next frame by the time the processor starts to work on the frame just completed). The additional information is as follows:

- A receive overrun or framing error occurred during frame reception.
- The frame failed the CRC check at the end of reception.
- Transmission of the frame was aborted.
- The number of bytes of valid data received in the frame (i.e. up to the end of frame or the overrun/framing error condition).

A control information buffer register is loaded whenever an end of received frame condition occurs. This event also generates an interrupt, which must be serviced before the end of the next received frame (at which point the buffered control information would be overwritten). The interrupt may be cleared by reading from the control information buffer register or by writing a ‘1’ to its status bit position.

13.4 Medium IrDA Specific Features

The MIR comprises a dedicated serial port and RZI modulator/demodulator supporting the Infrared Data Association (IrDA) standard for transmission/reception at 0.576 and 1.152 Mb/s.

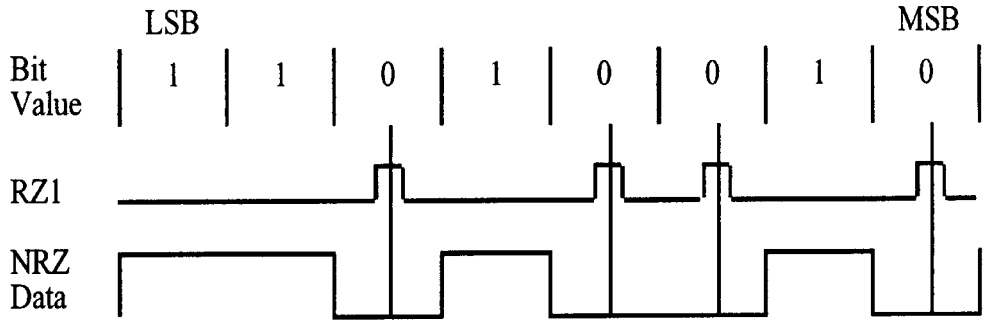
Frames contain an 8 bit address, an optional control field, a data field of any size that is a multiple of 8 bits and a 16-bit CRC-CCITT. The start/stop flag and CRC generation/checking is performed in the hardware. Data can be selectively saved in the receive buffer by programming an address with which to compare against all incoming frames. Interrupts are signalled when CRC checks performed on received data indicate an error, when a receiver abort occurs, when the transmit buffer underruns during an active frame and is aborted, when the receive buffer overruns and data is lost.

13.4.1 Introduction

13.4.1.1 Bit Encoding

The MIR bit encoding uses an RZI modulation scheme where a “0” is represented by a light pulse. For both 0.576 and 1.152 Mbit/sec data rates, the optical pulse duration is normally 1/4 of a bit duration. For example, if the data frame (in the order of transmission) is 11010010b, then Figure 13-1 represents the signal that is actually transmitted.

Figure 13-1. RZ1/NRZ Bit Encoding Example



13.4.1.2 Frame Format

MIR uses a flag (reserved bit pattern) to denote the beginning and end of a frame of information and to synchronize frame transmission. A double flag is used to indicate the start of a frame and a single flag the end. The flag contains eight bits, which start and end with a zero and contain six sequential ones in the middle (01111110b). This sequence of six ones is unique because all data between the start and stop flag is prohibited from having more than five consecutive ones. Data that violates this rule is altered before transmission by automatically inserting a zero after five consecutive ones are detected in the transmitted bit stream. This technique is commonly referred to

as “bit stuffing” and is transparent to the user. The information field within a MIR frame is placed between the start and stop flags, consisting of an 8 bit address, an optional 8 bit control field, a data field containing any multiple of 8 bits and a 16 bit cyclic redundancy check (CRC-CCITT). Note that each byte within the address, control and data fields is transmitted and received LSB first, ending with the byte’s MSB. However, the CRC is transmitted and received MSB first. The MIR frame format is outlined below in Table 13-3.

Table 13-3: MIR Frame Format

8 Bits	8 Bits	8 Bits	8 Bits (optional)	Any multiple of 8 Bits	16 Bits	8 Bits
Start Flag 0111 1110	Start Flag 0111 1110	Address	Control	Data	CRC-CCITT	Stop Flag 0111 1110

13.4.1.2.1 Address Field

The 8 bit address field is used by a transmitter to target a select group of receivers when multiple stations are connected using the infrared link. The address allows up to 255 stations to be uniquely addressed (00000000b to 11111110b). The global address (11111111b) is used to broadcast messages to all stations. The serial port contains an 8 bit register that is used to program a unique address for broadcast recognition as well as a control bit to enable/disable the address match function. Note that the address of received frames is stored in the receive buffer along with normal data and that it is transmitted and received starting with its LSB and ending with its MSB.

13.4.1.2.2 Control Field

The MIR control field is typically 8 bits, but can be any length. The serial port does not provide any hardware decode support for the control byte, but instead treats all bytes between the address and the CRC as data. Thus any control bits appear as data to the programmer. Note that the control field is transmitted and received starting with its LSB and ending with its MSB.

13.4.1.2.3 Data Field

The data field can be any length that is a multiple of 8 bits, including zero. The user determines the data field length according to the application requirements and transmission characteristics of the target system. Usually a length is selected which maximizes the amount of data that can be transmitted per frame, while allowing the CRC checker to be able to consistently detect all errors during transmission. All data fields must be a multiple of 8 bits. If a data field that is not a multiple of 8 bits is received, an abort is signalled and the end of frame tag is set within the receive buffer. Also note that each byte within the data field is transmitted and received starting with its LSB and ending with its MSB.

13.4.1.2.4 CRC Field

MIR uses the established CCITT cyclical redundancy check (CRC) to detect bit errors that occur during transmission. A 16 bit CRC-CCITT is computed using the address, control and data fields and is included in each frame. A separate CRC generator is implemented in both the transmit and receive logic. The transmitter calculates a CRC while data is actively transmitted and places the 16 bit value at the end of each frame before the stop flag is transmitted. The receiver calculates a CRC for each received data frame and compares the calculated CRC to the expected CRC value contained within the end of each received frame. If the calculated value does not match the expected value, an interrupt is signalled. The CRC computation logic is preset to all ones before reception/transmission of each frame. Note that the CRC is transmitted and received starting with its MSB and ending with its LSB. The CRC uses the four term polynomial:

$$CRC(x) = (x^{16} + x^{12} + x^5 + 1)$$

13.4.2 Functional Description

Following reset, the MIR is disabled. Reset also causes the transmit and receive buffers and tail register to be flushed (buffers marked as empty). To transmit data in MIR mode, use the following procedure:

1. Set the EN bits in the IrEnable register to 10b for MIR mode. Do not begin data transmission.
2. Before enabling the MIR, the user must first clear any writable or “sticky” status bits that are set by writing a one to each bit. (A sticky bit is a readable status bit that may be cleared by writing a one to its location.) Set the TAB and TFC bits in the MISR register, then read the MISR register to clear all interrupts.
3. Next, the desired mode of operation is programmed in the control register. Set the TXE and RXE bits in the IrCtrl register.
4. Write 1 to 3 bytes to the appropriate IrDataTail register.
5. Once the MIR is enabled, transmission/reception of data can begin on the transmit and receive pins.

13.4.2.1 Baud Rate Generation

The baud or bit rate is derived by dividing down an 18.423MHz clock. The clock is divided down by either 1 (BRD=1) or 2 (BRD=0) and then by a fixed value of four, generating the transmit clock for 1.152Mb/s and 0.576Mb/s data rates, respectively. The receive clock is generated by the receiver Digital Phase Locked Loop (DPLL). The DPLL uses a sample clock that is undivided. A sample rate counter (incremented at the sample clock rate) is used to generate a receive clock at the nominal data rate (sample clock divided by 41

and two-thirds). The sample rate counter is reset on the detection of each positive-going data transition (indicating the RZI encoding of a “0”) to ensure that synchronization with the incoming data stream is maintained.

13.4.2.2 Receive Operation

Once the MIR receiver is enabled it enters hunt mode, searching the incoming data stream for the flag (01111110b). The flag serves to achieve bit synchronization, denotes the beginning of a frame and delineates the boundaries of individual bytes of data. The end of the second flag denotes the beginning of the address byte. Once the flag is found, the receiver is synchronized to incoming data and hunt mode is exited.

After each bit is decoded, a serial shifter is used to receive the incoming data a byte at a time. Once the flag is recognized, each subsequent byte of data is decoded and placed within a two byte temporary buffer. A temporary buffer is used to prevent the CRC from being placed within the receive buffer. When the temporary buffer is filled, data values are pushed out one by one to the receive buffer. The first byte of a frame is the address. If receiver address matching is enabled, the received address is compared to the address programmed in the address match value field in a control register. If the two values are equal or if the incoming address contains all ones, all subsequent data bytes including the address byte are stored in the receive buffer. If the values do not match, the receive logic does not store any data in the receive buffer, ignores the remainder of the frame and begins to search for the stop flag. The second byte of the frame can contain an optional control field that must be decoded in software (There is no hardware support within the MIR). Use of a control byte is determined by the user.

When the receive buffer contains a word of data, an interrupt or DMA request is signalled. If the data is not removed soon enough and the buffer is completely filled, an overrun error is generated when the receive logic attempts to place additional data into the full buffer. If this occurs all subsequent data in the frame is discarded by the interface and the last valid entry in the buffer is marked with the ROR and EOF bits. The interface will stall in this state until the receive buffer is emptied.

Frames can contain any amount of data in multiples of 8 bits. Although the MIR protocol does not limit frame size, in practice they tend to be implemented in numbers ranging from hundreds to a couple of thousand bytes. In general this interface expects received frame size to be limited to 2047 bytes. However, the interface can continue to operate past this limit provided that software drivers are written that carefully check the indicated frame length with the amount of data transferred (in the DMA case this is a little more difficult).

The receive logic continuously searches for the stop flag at the end of the frame. Once it is recognized, the last byte that was placed within the receive buffer is flagged as the last byte of the frame and the two bytes remaining within the temporary buffer are removed and used as the 16 bit CRC value for

the frame. Instead of placing this in the receive buffer, the receive logic compares it to the CRC-CCITT value which is continuously calculated using the incoming data stream. If they do not match, the last byte that was placed within the receive buffer is also flagged with a CRC error. The CRC value is not placed in the receive buffer.

The MIR protocol permits back to back frames to be received. When this occurs, three flags separate back to back frames.

Most commercial IrDA transceivers can generate an abort (7 to 13 ones) when their transmit buffer underruns. The receive logic contains a counter that increments each time a one is decoded before entering the serial shifter and is reset any time a zero is decoded. When seven or more ones are detected, a receiver abort occurs. Note that data is moved from the serial shifter to the temporary buffer a byte at a time and seven consecutive ones may bridge two bytes. For this reason, after an abort is detected, the remaining data in the serial shifter is discarded along with the most recent byte of data placed in the temporary buffer. After this data is discarded, the oldest byte of data in the temporary buffer is placed in the receive buffer, the EOF tag is set within the top entry of the buffer (next to the byte transferred from the temporary buffer), the receiver abort interrupt is signalled and the receiver logic enters hunt mode until it recognizes the next flag.

This interface also generates an abort condition when a stop flag is received that is not byte aligned with the rest of the data in the frame. In this case the over flow data bits past the last byte boundary are discarded. It is not possible for the programmer to distinguish this condition for an normal abort condition.

If the user disables the receiver during operation, reception of the current data byte is stopped immediately, the serial shifter and receive buffer are cleared and all clocks used by the receive logic are automatically shut off to conserve power.

13.4.2.3 Transmit Operation

Immediately after enabling the MIR for transmission, the user may either “prime” the transmit buffer by filling it with data (see section “Functional Description” on page 397 for details) or allow service requests to cause the CPU or DMA to fill the buffer once the MIR is enabled. Once enabled, the transmit logic issues a service request if its buffer is empty. A Serial Infrared Interaction Pulse (SIP) is transmitted in order to guarantee non-disruptive co-existence with slower (up to 115.2 kbit/sec) systems, for example another device attempting to use its SIR. This is followed by continuous transmission of flags until valid data resides within the buffer. Once a byte of data resides at the bottom of the transmit buffer, it is transferred to the serial shifter, is encoded and shifted out onto the transmit pin clocked by the programmed baud rate clock. Note that the flags and CRC value are automatically transmitted and need not be placed in the transmit buffer.

When the transmit buffer has space for another word, an interrupt and/or DMA service request is signalled. If new data is not supplied soon enough, the buffer is completely emptied and the transmit logic attempts to take additional data from the empty buffer, one of two actions can be taken as programmed by the user. An underrun can either signal the normal completion of a frame or an unexpected termination of a frame in progress.

When normal frame completion is selected and an underrun occurs, the transmit logic transmits the 16 bit CRC value calculated during the transmission of all data within the frame (including the address and control bytes), followed by a flag to denote the end of the frame. The transmitter then transmits an SIP, followed by a continuous transmission of flags until data is once again available within the buffer. Once data is available, the transmitter begins transmission of the next frame.

When unexpected frame termination is selected and an underrun occurs, the transmit logic outputs an abort and interrupts the CPU. An abort continues to be transmitted until data is once again available in the transmit buffer. The MIR then transmits an SIP, followed by a double flag and starts the new frame. The off-chip receiver may choose to ignore the abort and continue to receive data, or to signal the serial port to retry transmission of the aborted frame. If the user disables the transmitter during operation, transmission of the current data byte is stopped immediately, the serial shifter and transmit buffer are cleared and all clocks used by the transmit logic are automatically disabled to conserve power.

13.5 Fast IrDA Specific Features

The Fast Infrared port (FIR) operates at half-duplex and provides direct connection to commercially available Infrared Data Association (IrDA) compliant LED transceivers. The FIR supports the 4.0 Mbps IrDA standard, using four pulse position modulation (4 PPM) and a specialized serial packet protocol developed expressly for IrDA transmission.

13.5.1 Introduction

13.5.1.1 4PPM Modulation

Four position pulse modulation (4PPM) is used for the high-speed transmission rate of 4.0 Mbps. Payload data is divided into data bit pairs (DBPs) for encoding with LSBs transmitted first. Each DBP is represented by one of four symbols (DDs) comprising a single 125 ms pulse within a 500 ms symbol period. The 125 ms quarters of a symbol are known as “chips”. The resulting signal waveform for the four data DDs is shown in Figure 13-2 and Figure 13-3 and shows modulation of the byte, 10110001b which is constructed using four DBPs.

Note: 1. Bits within each DBP are not reordered, but the least significant DBP is

transmitted first.

Note: 2. A "chip" in the context of the FIR is one time slice in the Position Modulation (PPM) symbol.

Figure 13-2. 4PPM Modulation Encoding

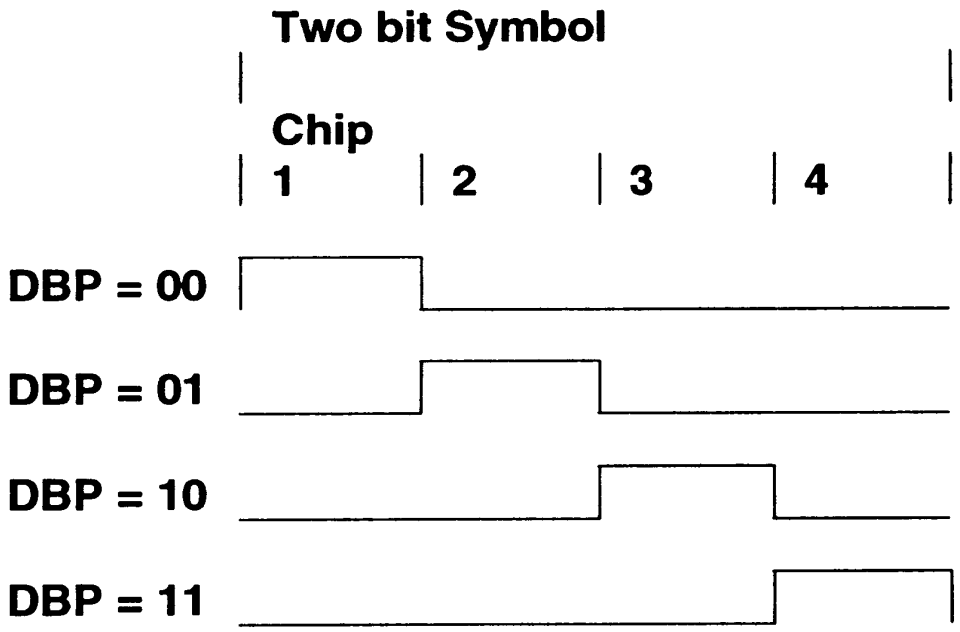
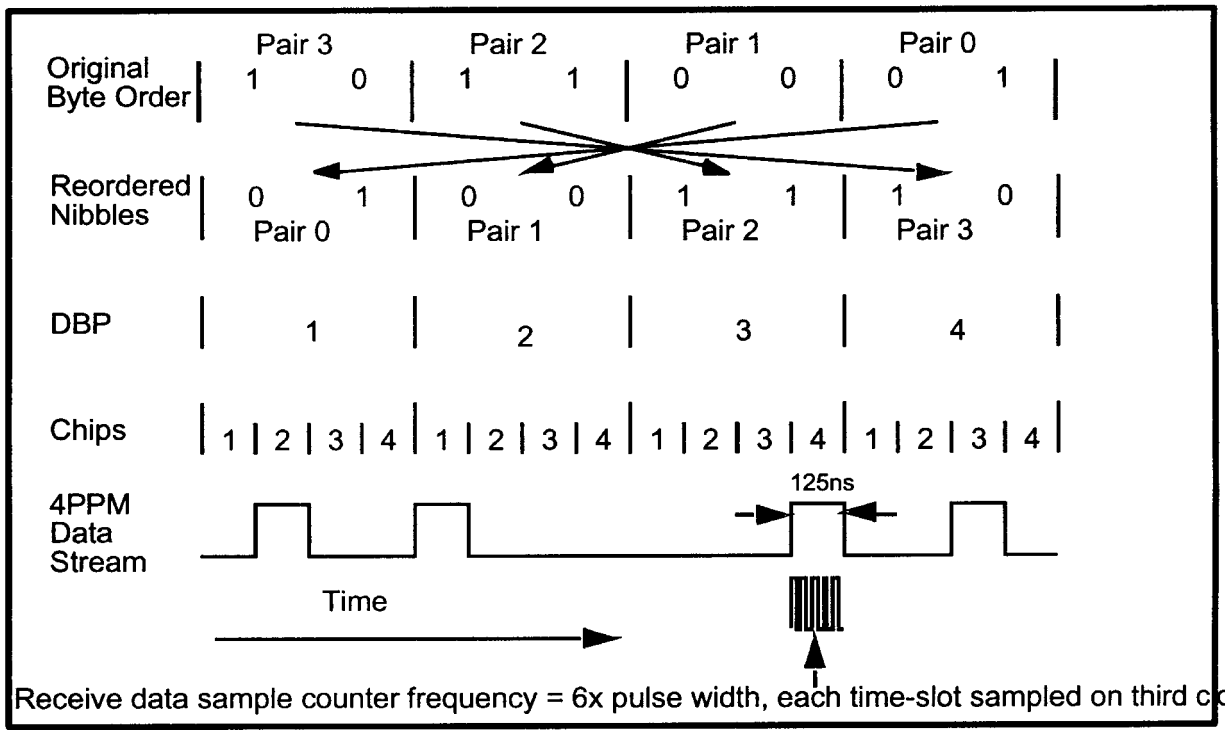


Figure 13-3. 4PPM Modulation Example



13.5.1.2 4.0 Mbps FIR Frame Format

When the 4.0 Mbps transmission rate is used, the high-speed serial/parallel (FIR) interface within the FIR is used along with the 4PPM bit encoding. The high-speed frame format shown in Figure 13-4 shown below, is similar to the SDLC format with several minor modifications: the start/stop flags and CRC are twice as long and instead of one start flag, a preamble and start flag of differing length are used.

Figure 13-4. IrDA (4.0 Mbps) Transmission Format

64 symbols	8 symbols	4 DDs (8 bits)	4 DDs (8 bits)	8180 DDs max (2045 bytes)	16 DDs (32 bits)	8 symbols
Preamble	**Start Flag**	**Address**	**Control (optional)**	**Data**	**CRC-32**	**Stop Flag**

Start Flag	|0000|1100|0000|1100|0110|0000|0110|0000|	
	|0000|1100|0000|1100|0000|0110|0000|0110|	**Stop Flag**
Preamble	|1000|0000|1010|1000|... repeated 16 times	

The preamble, start and stop flags are a mixture of symbols which contain either 0, 1, or 2 pulses within the four time slots. Symbols with 0 and 2 pulses are used to construct flags since they represent invalid data bit pairings (one pulse required per symbol to represent one of four bit pairs). The preamble contains sixteen repeated transmissions of the four symbols: **1000 0000 1010 1000**, the start flag contains one transmission of eight symbols: **0000 1100 0000 1100 0110 0000 0110 0000** and the stop flag contains one transmission of eight symbols: **0000 1100 0000 1100 0000 0110 0000 0110**. The address, control, data and CRC-32 all use the standard 4PPM DDs described above.

13.5.1.2.1 Address Field

The 8 bit address field is used by a transmitter to target a select group of receivers when multiple stations are connected to the same set of serial lines. The address allows up to 255 stations to be uniquely addressed (00000000b to 11111110b). The global address (11111111b) is use to broadcast messages to all stations. Serial port 1 contains an 8 bit register which is used to program a unique address for broadcast recognition as well as a control bit to enable/disable the address match function. Note that the address of received frames is stored in the receive buffer along with normal data and that it is transmitted and received starting with its LSB and ending with its MSB.

13.5.1.2.2 Control Field

The IPC control field is 8 bits and is optional (as defined by the user). The FIR does not provide any hardware decode support for the control byte, but instead treats all bytes between the address and the CRC as data. Note that

the control field is transmitted and received starting with its LSB and ending with its MSB.

13.5.1.2.3 Data Field

The data field can be any length which is a multiple of 8 bits, from 0 to 2045 bytes. The user determines the data field length according to the application requirements and transmission characteristics of the target system. Usually a length is selected which maximizes the amount of data which can be transmitted per frame, while allowing the CRC checker to be able to consistently detect all errors during transmission. Note that the serial port does not contain any hardware which restricts the maximum amount of data transmitted or received. It is up to the user to maintain these limits. If a data field which is not a multiple of 8 bits is received an abort is signalled. Also note that each byte within the data field is transmitted and received starting with its LSB and ending with its MSB.

13.5.1.2.4 CRC Field

The FIR uses the established 32 bit cyclical redundancy check (CRC-32) to detect bit errors which occur during transmission. A 32 bit CRC is computed using the address, control and data fields and is included in each frame. A separate CRC generator is implemented in both the transmit and receive logic. The transmitter calculates a CRC while data is actively transmitted byte shifting each byte transmitted through its serial shifter LSB first, then places the inverse of the resultant 32 bit value at the end of each frame before the flag is transmitted. In a similar manner, the receiver also calculates a CRC for each received data frame and compares the calculated CRC to the expected CRC value contained within the end of each received frame. If the calculated value does not match the expected value, an interrupt is signalled. The CRC computation logic is preset to all ones before reception/transmission of each frame and the result is inverted before it used for comparison or transmission. Note that unlike the address, control and data fields, the 32 bit inverted CRC value is transmitted and received from least significant byte to most significant and within each byte the least significant nibble is encoded/decoded first. The cyclical redundancy checker uses the 32 term polynomial:

$$CRC(x) = (x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1)$$

13.5.2 Functional Description

Following reset, the FIR is disabled. Reset also causes the transmit and receive buffers and tail register to be flushed (buffers marked as empty). To transmit data in FIR mode, use the following procedure:

1. Set the EN bits in the IrEnable register to 11b for FIR mode. Do not begin data transmission.

2. Before enabling the FIR, the user must first clear any writable or “sticky” status bits that are set by writing a one to each bit. (A sticky bit is a readable status bit that may be cleared by writing a one to its location.) Set the TAB and TFC bits in the FISR register, then read the FISR register to clear all interrupts.
3. Next, the desired mode of operation is programmed in the control register. Set the TXE and RXE bits in the IrCtrl register.
4. Write 1 to 3 bytes to the appropriate IrDataTail register.
5. Once the FIR is enabled, transmission/reception of data can begin on the transmit and receive pins.

13.5.2.1 Baud Rate Generation

The baud rate is derived by dividing down a fixed 48 MHz clock. The 8 MHz baud (time-slot) clock for the receiver is synchronized with the 4 PPM data stream each time a transition is detected on the receive data line using a digital PLL. To encode a 4.0 Mbps data stream, the required “symbol” frequency is 2.0 MHz, with four chips per symbol at a frequency of 8.0 MHz. Receive data is sampled half way through each time-slot period by counting three out of the six 48 MHz clock periods which make up each chip. Refer to Figure 13-3 on page 401. The symbols are synchronized during preamble reception. Recall that the preamble consists of four symbols repeated sixteen times. This repeating pattern is used to identify the first time-slot or beginning of a symbol and resets the two bit chip counter logic, such that the 4 PPM data is properly decoded.

13.5.2.2 Receive Operation

The IrDA standard specifies that all transmission occurs at half-duplex. This restriction forces the user to enable one direction at a given time; either the transmit or receive logic, but not both. However, the FIR’s hardware does not impose such a restriction. The user may enable both the transmitter and receiver at the same time. Although forbidden by the IrDA standard, this feature is particularly useful when using the FIR’s loop back mode, which internally connects the output of the transmit serial shifter to the input of the receive serial shifter.

After the FIR is enabled for 4.0 Mbps transmission, the receiver logic begins by selecting an arbitrary symbol boundary, receives four incoming 4 PPM symbols from the input pin using a serial shifter and latches and decodes the symbols one at a time. If the symbols do not decode to the correct preamble, the chip counter’s clock is forced to skip one 8MHz period, effectively delaying the chip count by one. This process is repeated until the preamble is recognized, signifying that the chip counter is synchronized. The preamble may be repeated as few as sixteen times, or may be continuously repeated to indicate an idle receive line.

At any time after the transmission of sixteen preambles, the start flag may be received. The start flag is eight symbols long. If any portion of the start flag does not match the standard encoding, the receiver signals a framing error and the receive logic once again begins to look for the frame preamble.

Once the correct start flag is recognized, each subsequent grouping of four DDs is decoded into a data byte, placed within a five byte temporary buffer which is used to prevent the CRC from being placed within the receive buffer. When the temporary buffer is filled, data values are pushed out one by one to the receive buffer. The first data byte of a frame is the address. If receiver address matching is enabled, the received address is compared to the address programmed in the address match value field in one of the control registers. If the two values are equal or if the incoming address contains all ones, all subsequent data bytes including the address byte are stored in the receive buffer. If the values do not match, the receiver logic does not store any data in the receive buffer, ignores the remainder of the frame and begins to search for the next preamble. The second data byte of the frame can contain an optional control field as defined by the user and must be decoded in software (there is no hardware support within the FIR).

Frames can contain any amount of data in multiples of 8 bits up to a maximum of 2047 bytes (including the address and control byte). In general this interface expects received frame size to be limited to 2047 bytes. However, the interface can continue to operate past this limit, thus it is the responsibility of the user to check that the size of each incoming frame does not exceed the IrDA protocol's maximum allowed frame size. The BC field in the IrRIB register can not be used for this since it will over flow (and wrap), the true frame length can be deduced from the DMA buffer position in combination with the BC field.

When the receive buffer contains a word of data, an interrupt or DMA request is signalled. If the data is not removed soon enough and the buffer is completely filled, an overrun error is generated when the receive logic attempts to place additional data into the full buffer. If this occurs all subsequent data in the frame is discarded by the interface and the last valid entry in the buffer is marked with the ROR and EOF bits. The interface will stall in this state until the receiver buffer is emptied.

When a framing error is detected all subsequent data in the frame is discarded by the interface and an entry is put into the buffer with the FRE and EOF bits set The data in this buffer entry is invalid.

If any two sequential symbols within the data field do not contain pulses (are 0000b), the frame is aborted. The oldest byte in the temporary buffer is moved to the receive buffer (the remaining four buffer entries are discarded). The end of frame (EOF) tag is set within the same buffer entry where the last "good" byte of data resides and the receiver logic begins to search for the preamble. An abort occurs if any data symbol contains 0011b, 1010b, 0101b, or 1001b (invalid symbols which do not occur in the stop flag).

The receiver continuously searches for the 8 symbol stop flag. Once it is recognized, the last byte placed within the receive buffer is flagged as the last byte of the frame and the data in the temporary buffer is removed and used as the 32 bit CRC value for the frame. Instead of placing this in the receive buffer, the receiver compares it to the CRC-32 value which is continuously calculated using the incoming data stream. If they do not match, the last byte which was placed in the receiver buffer is also tagged with a CRC error. The CRC value is not placed in the receive buffer.

If the user disables the FIR's receiver during operation, reception of the current data byte is stopped immediately, the serial shifter and receive buffer are cleared and all clocks used by the receive logic are automatically shut off to conserve power.

13.5.2.3 Transmit Operation

Immediately after enabling the FIR for transmission, the user may either "prime" the transmit buffer by filling it with data (see section "Functional Description" on page 403 for details) or allow service requests to cause the CPU or DMA to fill the buffer once the FIR is enabled. Once enabled, the transmit logic issues a service request if its buffer is empty. For each frame output, a minimum of sixteen preambles are transmitted. If data is not available after the sixteenth preamble, additional preambles are output until a byte of valid data resides within the bottom of the transmit buffer. The preambles are then followed by the start flag and then the data from the transmit buffer. Four symbols (8 bits) are encoded at a time and then loaded into a serial shift register. The contents are shifted out onto the transmit pin clocked by the 8 MHz baud clock. Note that the preamble, start and stop flags and CRC value is automatically transmitted and need not be placed in the transmit buffer.

When the transmit buffer is emptied, an interrupt and/or DMA service request is signalled. If new data is not supplied quickly enough and the transmit logic attempts to take additional data from the empty buffer, one of two actions can be taken as programmed by the user. An underrun can either signal the normal completion of a frame or an unexpected termination of a frame in progress.

When normal frame completion is selected and an underrun occurs, the transmit logic transmits the 32 bit CRC value calculated during the transmission of all data within the frame (including the address and control bytes), followed by the stop flag to denote the end of the frame. The transmitter then continuously transmits preambles until data is once again available within the buffer. Once data is available, the transmitter begins transmission of the next frame.

When unexpected frame termination is selected and an underrun occurs, the transmit logic outputs an abort and interrupts the CPU. An abort continues to be transmitted until data is once again available in the transmit buffer. The FIR

then transmits 16 preambles, a start flag and starts the new frame. The remote receiver may choose to ignore the abort and continue to receive data, or to signal the FIR to retry transmission of the aborted frame.

At the end of each frame transmitted, the FIR outputs a pulse called the serial infrared interaction pulse (SIP). A SIP is required at least every 500 ms to keep slower speed devices (115.2 kbps and slower) from colliding with the higher speed transmission. The SIP simulates a start bit which causes all low speed devices to stay off the bus for at least another 500 ms. Transmission of the SIP pulse causes the transmit pin to be forced high for a duration of 1.625 µs and low for 7.375 µs (total SIP period = 9.0 µs). After the 9.0 µs elapses, the preamble is then transmitted continuously to indicate to the remote receiver that the FIR's transmitter is in the idle state. The preamble continues to be transmitted until new data is available within the transmit buffer, or the FIR's transmitter is disabled. Note that it is the responsibility of the user to ensure that a frame completes once every 500 ms such that a SIP pulse is produced keeping all low speed devices from interrupting transmission. Because most IrDA compatible devices produce a SIP after each frame transmitted, the user may only need to ensure that a frame is either transmitted or received by the FIR every 500 ms.

Note that frame length does not represent a significant portion of the 500 ms time frame in which a SIP must be produced. At 4.0 Mbps, the longest frame allowed is 16,568 bits, which takes just over 4 ms to transmit. Also note that the FIR issues a SIP when the transmitter is first enabled, to ensure all low speed devices are silenced before transmitting it's first frame.

If the user disables the FIR's transmitter during operation, transmission of the current data byte is stopped immediately, the serial shifter and transmit buffer are cleared. All clocks used by the transmit logic are automatically shut off to conserve power.

13.5.3 IrDA Connectivity

The IrDA controller uses package pins **RXD1** and **TXD1**. The IrDA input signal is always **RXD1**. Syscon register DeviceCfg.IonU2 controls what drives bit **TXD1**. See Figure 13-4 on page 407.

Table 13-4: DeviceCfg.IonU2 Pin Function

DeviceCfg.IonU2	Pin TXD1 Function
0	UART2 is the output signal
1	Logical OR of IrDA output signal and UART2 SIR output signal

Therefore, to use any IrDA mode, FIR, MIR or SIR, set IonU2. To use UART2 as a UART, clear IonU2.

13.5.4 IrDA Integration Information

13.5.4.1 Enabling Infrared Modes

Table 13-5: UART2 / IrDA Modes

Mode	DeviceCfg Register		UART2Ctrl Register		IrEnable Register	
	U2EN	IonU2	SIREn	UARTE	EN[1]	EN[0]
Disabled	0	x	0	0	0	0
UART2	1	0	0	1	0	0
SIR	1	1	1	1	0	1
MIR	x	1	0	0	1	0
FIR	x	1	0	0	1	1

13.5.4.2 Clocking Requirements

There are four clocks, PCLK, MIRCLK, FIRCLK, and UARTCLK.

Version 1.1 of the Infrared Data Association standard indicates the following:

- FIRCLK must by 48.0 MHz with a tolerance of 0.01%.
- MIRCLK must be 18.432 MHz with a tolerance of 0.1%.

The worst case ratio that can be supported for PCLK:FIRCLK is a ratio of 1:5. The maximum that PCLK can be is 66 MHz, therefore:

$$\frac{1}{5}F_{FIRCLK} < F_{PCLK} < 66.0MHz$$

Any frequencies outside the above range are not supported and will result in incorrect behavior of the FIR mode of the infrared peripheral.

Since MIRCLK is 18.432 MHz, PCLK can be as low as 3.68 MHz and as high as 66 MHz. Any PCLK frequency in this range is allowable. Any PCLK frequencies outside the range are not supported and will result in incorrect behavior of the MIR mode of the infrared peripheral, therefore:

$$3.68MHz \leq F_{PCLK} \leq 66.0MHz$$

The tolerance of UARTCLK is defined by the UART to which it is connected.

UARTCLK frequency must accommodate the desired range of baud rates:

$$F_{UARTCLK_{MIN}} \geq 32 \times baudrate_{MAX}$$

$$F_{UARTCLK_{MAX}} \leq 32 \times 65536 \times baudrate_{MIN}$$

The frequency of UARTCLK must also be within the required error limits for all baud rates to be used.

To allow sufficient time to write the received data to the receive FIFO, UARTCLK must be less than or equal to four times the frequency of PCLK:

$$F_{UARTCLK} \leq 4 \times F_{PCLK}$$

If the IrDA SIR functionality is required, UARTCLK must have a frequency between 2.7 MHz and 542.7 MHz to ensure that the low-power mode transmit pulse duration complies with the IrDA SIR specification.

13.5.4.3 Bus Bandwidth Requirements

There are four different IrDA modes with different bandwidth requirements. Furthermore, there are two basic ways of moving data to or from the IrDA FIFOs:

- Direct DMA interface - this permits byte-wide access to the IrDA without using the APB. The DMA block will pack/unpack individual bytes so that it reads or writes full 32-bit words rather than individual bytes.
- Accessing the IrDA via the APB - this requires APB/AHB bus bandwidth. Then, both a read and write are required for each 32-bit data word.

Assuming most bytes in a packet are moved either via the DMA interface or via 32-bit word accesses to the IrDA controller on the APB, the following table indicates the maximum average number of memory accesses per second to service IrDA TX or RX:

Table 13-6: IrDA Service Memory Accesses / Second

Infrared Mode	Bit Rate (bits / second)	Bus accesses / second	
		DMA	APB
SIR	115,200	3,600	7,200
Slow MIR	576,000	18,000	36,000
Fast MIR	1,152,000	36,000	72,000
FIR	4,000,000	125,000	250,000

Note that the SIR mode bit rate is a worst case value.

13.6 Registers

Register Descriptions

IrEnable

13

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											FD	MD	LBM	EN	

Address:

0x808B_0000 - Read/Write

Default:

0x0000_0018

Definition:

IrDA Enable Register. This register selects which Infrared interface module is active. The Medium and Fast modules share common control, flag, and data interfaces while maintaining separate status registers.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

FD: Fast done status. Read-only bit indicating that the FIR transmit module has completed transition of the current frame and that it is safe to disable the module using the EN control bits.

MD: Medium done status. Read-only bit indicating that the MIR transmit module has completed transmission of the current frame and that it is safe to disable the module using the EN control bits.

LBM: Loopback Mode, for MIR and FIR operation.
0 - Normal operation.
1 - Loopback active, the transmit serial shifter is directly connected to the receive serial shifter.

EN: Enable value:
00 - No encoder selected
01 - SIR, 0 to 0.1152Mbit/s data rate, using the UART2 interface
10 - MIR, 0.576 or 1.152Mbit/s data rate, using IrDA interface
11 - FIR, 4.0Mbit/s data rate, using IrDA interface.

Note: While the FIR transmit section is enabled, the FD bit is low, and while the MIR transmit section is enabled, the MD bit is low. In FIR mode, the FD bit does not go high until the TXE bit in the IrCtrl register is cleared, and in MIR mode, the same bit must be cleared for MD to go high. Monitor the TBY bit in the IrFlag register to discover whether a packet is fully transmitted before clearing TXE.

IrCtrl

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								AME	RXP	TXP	RXE	TXE	TUS	BRD	0

Address:
0x808B_0004

Default:
0x0000_0000

Definition:
IrDA Control Register. This register selects various operating parameters. Note that the RXE and TXE bit must be cleared before selecting a different interface with the IrEnable register EN bits. The other bits in this register may be changed while the interface is active.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

AME: Address Match Enable.
0 - Disable receiver address match function, store data from all incoming frames in the receive buffer.
1 - Enable receiver address match function, do not buffer data unless address is recognized or incoming address contains all ones.

RXP: Receive Polarity Control.
0 - Data input is not inverted before decoding.
1 - Data input is inverted before decoding.

13

TXP: Transmit Polarity Control.
0 - Encoded data is not inverted before being passed to the pins.
1 - Encoded data is inverted before being passed to the pins.

RXE: Receive Enable.
0 - Ir receive logic is disabled and clocks are stopped.
1 - Ir receive logic is enabled.

TXE: Transmit Enable.
0 - Transmit logic is disabled and clocks are stopped.
1 - Transmit logic is enabled.

TUS: Transmit buffer Underrun Select.
0 - Transmit buffer underrun causes CRC, stop flag, and SIP to be transmitted.
1 - Transmit buffer underrun causes an abort to be transmitted.

BRD: MIR Bit rate select.
0 - MIR data rate is 0.576 Mbit/s.
1 - MIR data rate is 1.152 Mbit/s.

0: Must be written to “0”.

IrAdrMatchVal

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								AMV							

Address:

0x808B_0008 - Read/Write

Default:

0x0000_0000

Definition:

IrDA Address Match Value Register contains the 8 bit address match value field which is used by the receiver to selectively store only the data within the receive frames which have the same address. For incoming frames which have the same address value as the AMV field, the frame’s address, control and data is stored in the receive buffer. For those that do not match, the remainder of the frame is ignored and the receive logic searches for the beginning of the next frame. This register is used for both MIR and FIR. The AME bit in IrCtrl must be set to enable this function. Frames containing an

address of all ones are broadcast frames, and are always matched regardless of the value in the AMV. The AMV may be written at any time, allowing the address match value to be changed during active receive operation.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

AMV: Address Match Value.

IrFlag

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						TBY	RIF	RSY	EOF	WST	WST	FRE	ROR	CRE	RAB

Address:

0x808B_008B - Read Only

Default:

0x0000_0000

Definition:

IrDA Flag Register. Contains the nine read only flags which indicate the current state of the IrDA Interface.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

TBY: Transmitter Busy Flag.
0 - Transmitter is idle, or disabled, or an abort is being transmitted.
1 - Transmit logic is currently transmitting a frame.

RIF: Receiver In Frame.
0 - Receiver is in preamble/start flag or is in hunt mode.
1 - Receiver is in a frame.

RSY: Receiver Synchronized Flag.
0 - Receiver is in hunt mode.
1 - Receiver logic is synchronized within the incoming data.

EOF: End of Frame.
0 - Current frame is not completed.
1 - The word in the receive buffer contains the last byte of data within the frame. When the last word in the current frame is read this bit is cleared.

WST: Width Status.
00 - All four bytes in receive buffer are valid.
01 - Least significant byte is valid only.
10 - Least significant two bytes are valid only.
11 - Least significant three bytes are valid only.

FRE: FIR Framing Error.
0 - No framing errors encountered in the receipt of FIR data.
1 - Framing error occurred, FIR preamble followed by something other than another preamble or FIR start flag. The data in the buffer is invalid.

ROR: Receive buffer Overrun.
0 - Receive buffer has not experienced an overrun.
1 - Receive logic attempted to place data into receive buffer while it was full. The next data value in the buffer is the last piece of “good” data before the buffer was overrun.

CRE: CRC Error.
0 - No CRC check errors encountered in the data.
1 - CRC calculated on the incoming data does not match CRC value contained within the received frame.

RAB: Receiver Abort.
0 - No abort has been detected for the incoming frame.
1 - Abort detected during receipt of the incoming frame, EOF bit set in receive buffer next to the last piece of “good” data received before abort.

IrData

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															

Address:

0x808B_0010 - Read/Write

Default:

0x0000_0000

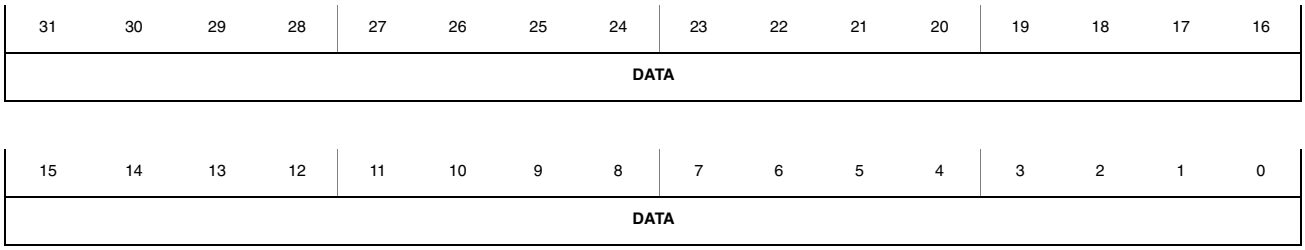
Definition:

IrDA Data Register. Provides access to the transmit and receive buffers used by the MIR and FIR interfaces.

Bit Descriptions:

DATA: IrDA data word. Values written and sent to the transmit FIFO. Values read are from the receiver FIFO.

IrDataTail



Address:

0x808B_0014, 0x808B_0018, 0x808B_001C - Write Only

Default:

0x0000_0000

Definition:

IrDA Data Tail Register. This is a 24-bit write only register used for transmitting frames whose payload data is not an integer multiple of 4 bytes long. The bit locations are cleared when read by the transmit logic or when the TXE control bit is clear. The IrData Tail register may be written using one of three addresses. Bits two and three of the address determine how many bytes within the word are significant, that is, are intended for transmission. If none of the address is written, the register remains marked as empty and payload data will be read by the transmit logic from the 32-bit FIFO only. The status of this register does not affect the TFS flag, nor does it cause interrupts or DMA requests to be generated.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DATA: IrDA transmit payload data. Write to address 0x014, least significant byte is transmitted. Write to address 0x018, least significant two bytes are transmitted. Write to address 0x01C, least significant three bytes are transmitted.

IrRIB

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	BC											BFRE	BROR	BCRE	BRAB

Address:

0x808B_0020 - Read Only

Default:

0x0000_0000

Definition:

IrDA Receive Information Register. This register contains 15 read only bits that identify flag and byte count values from the last received frame. The bits are copied from the flag register when the last data in a frame is read from the receive FIFO.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BC: Byte Count. The total number of valid bytes read from the interface during the last frame. If the total number of bytes is greater than 2047, only the lower eleven bits are presented.

BFRE: Buffered Framing Error.
0 - No framing errors were encountered during the last frame.
1 - A framing error occurred during the last frame causing the remainder of the frame to be discarded.

BROR: Buffered Receive buffer Overrun.
0 - The receive buffer did not overrun during the last frame.
1 - Receive logic attempted to place data into receive buffer while it was full during the last frame causing the remainder of the frame to be discarded.

BCRE: Buffered CRC Error.
0 - No CRC check errors encountered in the last frame.
1 - CRC calculated on the incoming data did not match CRC value contained within the received frame for the last frame.

BRAB: Buffered Receiver Abort.
0 - No abort was detected in the last frame.
1 - The last frame was terminated with an abort condition.

IrTR0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								BC							

Address:
0x808B_0024 - Read Only

Default:
0x0000_0000

Definition:
IrDA Test Register 0. This register indicates the received byte count.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

BC: Byte Count. The total number of valid bytes read by the receiver.

IrDMACR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													DMAERR	TXDMAE	RXDMAE

Address:
0x808B_0028 - Read/Write

Default:
0x0000_0000

Definition:
IrDA DMA Control Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

DMAERR: RX DMA error handing enable. If 0, the RX DMA interface ignores error conditions in the IrDA receive section. If “1”, the DMA interface stops and notifies the DMA block when an error occurs. Errors include framing errors, receive abort, and CRC mismatch.

TXDMAE: TX DMA interface enable. Setting to “1” enables the private DMA interface to the transmit FIFO.

RXDMAE: RX DMA interface enable. Setting to “1” enables the private DMA interface to the receive FIFO.

SIRTR0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								SIREN	SIROUT	TXD	RXD	SIRT	SIRIN	S16CLK	TSIRC

Address:

0x808B_0030 - Read/Write

Default:

0x0000_0000, except that bit 4 is unknown at reset

Definition:

IrDA Slow InfraRed Test Register 0.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

SIREN: The state of the **SIREN** after synchronization. Read only.

SIROUT: The state of **SIROUT** output from the InfraRed block. Read only.

TXD: The state of the **TXD** input to the InfraRed block from UART2. Read only.

RXD: The state of the **RXD** output from the InfraRed block to UART2. Read only.

MISR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									RFL	RIL	RFC	RFS	TAB	TFC	TFS

Address:

0x808B_0080 - Read/Write

Default:

0x0000_0000

Definition:

MIR Status Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: Receive Frame Lost. Set to a “1” when a ROR occurred at the start of a new frame, before any data for the frame could be put into the receive FIFO. This bit is cleared by writing a “1” to this bit. This occurs if the last entry in the FIFO already contains a valid EOF bit from a previous frame when a FIFO overrun occurs. The ROR bit cannot be placed into the FIFO and all data associated with the frame is lost.

RIL: Receive Information Buffer Lost. Set to a “1” when the last data for a frame is read from the receive FIFO and the RFC bit is still set from a previous end of frame. This bit is cleared by writing a “1” to this bit. This is triggered if the RFC bit is already set before the last data from a frame is read from the IrData register. It indicates that the data from the IrRIB register was lost. This can occur if the CPU does not respond to the RFC interrupt before another (short) frame completes and is read from the IrData register by the DMA controller.

RFC: Received Frame Complete. Set to “1” when the last data for a frame is read from the receive FIFO (via the IrData register). This event also triggers the IrRIB to load the IrFlag and byte count. This bit is cleared when the IrRIB register is read.

13

RFS: Receive buffer Service Request (read only).
0 - Receive buffer is empty or the receiver is discarding data or the receiver is disabled.
1 - Receive buffer is not empty and the receiver is enabled, DMA service request signaled.

TAB: Transmit Frame Aborted. Set to “1” when a transmitted frame is terminated with an abort. This will only occur if the TUS bit is set in the IrCtrl register. Writing a “1” to this bit clears it.

TFC: Transmitted Frame Complete. Set to “1” whenever a transmitted frame completes, whether it is terminated with a CRC followed by a stop flag or terminated with an abort. Writing a “1” to this bit clears it.

TFS: Transmit buffer Service Request (read only).
0 - Transmit buffer is full or transmitter disabled.
1 - Transmit buffer is not full and the transmitter is enabled, DMA service is signaled.
The bit is automatically cleared after the buffer is filled.

MIMR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									RFL	RIL	RFC	RFS	TAB	TFC	TFS

Address:
0x808B_0084 - Read/Write

Default:
0x0000_0000

Definition:
MIR Interrupt Mask Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: RFL mask bit. When high, the MIR RFL status can generate an interrupt.

RIL: RIL mask bit. When high, the MIR RIL status can generate an interrupt.

RFC: RFC mask bit. When high, the MIR RFC status can generate an interrupt.

RFS: RFS mask bit. When high, the MIR RFS status can generate an interrupt.

TAB: TAB mask bit. When high, the MIR TAB status can generate an interrupt.

TFC: TFC mask bit. When high, the MIR TFC status can generate an interrupt.

TFS: TFS mask bit. When high, the MIR TFS status can generate an interrupt.

MIIR

Address:

0x808B_0088 - Read Only

Default:

0x0000_0000

Definition:

MIR Interrupt Register. The IrDA interrupt is asserted if any bit in the MIIR is high.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: Logical AND of MIR RFL status bit and RFL mask bit.

RIL: Logical AND of MIR RIL status bit and RIL mask bit.

RFC: Logical AND of MIR RFC status bit and RFC mask bit.

RFS: Logical AND of MIR RFS status bit and RFS mask bit.

TAB: Logical AND of MIR TAB status bit and TAB mask bit.

TFC: Logical AND of MIR TFC status bit and TFC mask bit.

TFS: Logical AND of MIR TFS status bit and TFS mask bit.

FISR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									RFL	RIL	RFC	RFS	TAB	TFC	TFS

Address:

0x808B_0180 - Read/Write

Default:

0x0000_0000

Definition:

FIR Status Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: Receive Frame Lost. Set to a “1” when a ROR occurred at the start of a new frame, before any data for the frame could be put into the receive FIFO. This bit is cleared by writing a “1” to this bit. This occurs if the last entry in the FIFO already contains a valid EOF bit from a previous frame when a FIFO overrun occurs. The ROR bit cannot be placed into the FIFO and all data associated with the frame is lost.

RIL: Receive Information Buffer Lost. Set to a “1” when the last data for a frame is read from the receive FIFO (via the IrData register) and the RFC bit is still set from a previous end of frame. It indicates that data in the IrRIB register for the previous frame was lost. This can occur if the CPU does not respond to the RFC interrupt before another frame completes and is read from the IrData register by the DMA controller. This bit is cleared by writing a “1” to this bit.

RFC: Received Frame Complete. Set to “1” when the last data for a frame is read from the receive FIFO (via the IrData register). This event also triggers the IrRIB to load the IrFlag and byte count. This bit is cleared when the IrRIB register is read.

RFS: Receive buffer Service Request (read only).
0 - Receive buffer is empty or the receiver is discarding data or the receiver is disabled.
1 - Receive buffer is not empty and the receiver is enabled, DMA service request signaled.
The bit is automatically cleared when the receive buffer is emptied.

TAB: Transmit Frame Aborted. Set to “1” when a transmitted frame is terminated with an abort. This will only occur if the TUS bit is set in the IrCtrl register. The bit is cleared by writing a “1” to this bit.

TFC: Transmitted Frame Complete. Set to “1” whenever a transmitted frame completes (whether it is terminated with a CRC followed by a stop flag or terminated with an abort). This bit is cleared by writing a “1” to this bit.

TFS: Transmit buffer Service Request (read only).
0 - Transmit buffer is full or transmitter disabled.
1 - Transmit buffer is not full and the transmitter is enabled, DMA service is signaled.
The bit is automatically cleared after the buffer is filled.

FIMR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									RFL	RIL	RFC	RFS	TAB	TFC	TFS

Address:
0x808B_0184 - Read/Write

Default:
0x0000_0000

Definition:
FIR Interrupt Mask Register.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: RFL mask bit. When high, the FIR RFL status can generate an interrupt.

RIL: RIL mask bit. When high, the FIR RIL status can generate an interrupt.

RFC: RFC mask bit. When high, the FIR RFC status can generate an interrupt.

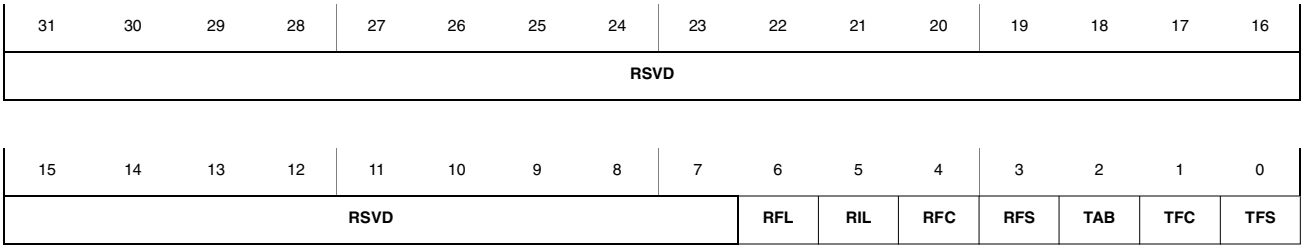
RFS: RFS mask bit. When high, the FIR RFS status can generate an interrupt.

TAB: TAB mask bit. When high, the FIR TAB status can generate an interrupt.

TFC: TFC mask bit. When high, the FIR TFC status can generate an interrupt.

TFS: TFS mask bit. When high, the FIR TFS status can generate an interrupt.

FIIR



Address:

0x808B_0188 - Read Only

Default:

0x0000_0000

Definition:

FIR Interrupt Register. An interrupt is signalled from this block if any bit is high in the FIIR.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

RFL: Logical AND of FIR RFL status bit and RFL mask bit.

RIL: Logical AND of FIR RIL status bit and RIL mask bit.

RFC: Logical AND of FIR RFC status bit and RFC mask bit.

RFS: Logical AND of FIR RFS status bit and RFS mask bit.

TAB: Logical AND of FIR TAB status bit and TAB mask bit.

TFC: Logical AND of FIR TFC status bit and TFC mask bit.

TFS: Logical AND of FIR TFS status bit and TFS mask bit.

Chapter 14

Timers

14.1 Introduction

The timers are used to control timed events in the system. For example, a wait can be inserted by setting the timer value to an appropriate value and waiting for the timer interrupt.

The Timers block contains two 16-bit timers, one 32-bit timer and one 40-bit time stamp debug timer.

14.1.1 Features

The EP9301 has the following timer features:

- Two 16-bit timers
  - Free running
  - Load based
- One 32-bit timer
  - Free running
  - Load based
- One 40-bit timer
  - Free running

14.1.2 16 and 32-bit Timer Operation

The two 16-bit timers are referred to as TC1 and TC2. Each of these timers has an associated 16-bit read/write data register and a control register. Each counter is loaded with the value written to the data register immediately. This value will then be decremented on the next active clock edge to arrive after the write. When the timer counter decrements to “0”, it will assert the appropriate interrupt. The timer counters can be read at any time. The clock source and mode is selectable by writing to various bits in the system control register. Clock sources are 508 kHz and 2 kHz. Both of these clock sources are synchronized to the main system AHB bus clock (HCLK).

Timer 3 (TC3) has the exact same operation as TC1 and TC2, but it is a 32-bit counter. It has the same register arrangement as TC1 and TC2, providing a

load, value, control and clear register. The 16 and 32-bit timer counters can operate in two modes, free running mode or pre-load mode.

14.1.2.1 Free Running Mode

In free running mode, counters TC1 and TC2 will wrap to 0xFFFF when they reach zero (underflow), and continue counting down. Counter TC3 will wrap to 0xFFFFFFFF when it underflows, and continues counting down.

14.1.2.2 Pre-load Mode

In pre-load (periodic) mode, the value written to the TC1, TC2 or TC3 Load registers is automatically re-loaded when the counter underflows. This mode can be used to generate a programmable periodic interrupt.

14.1.3 40-bit Timer Operation

The time stamp debug timer is a 40-bit up-counter used only for long term debugging (TC4). Its clock source is the 14.7456 MHz clock, divided by 15 to give a 983.04 kHz reference. The timer value may be read at any time by reading the lower 32-bit word first and then the high byte. Dividing the result by 983 yields a timestamp in milliseconds. The debug timer does not cause an interrupt. The timer is controlled by a single enable bit. When the timer is enabled, it begins counting from zero and when it is disabled, it is cleared back to zero. When it reaches its maximum value (0xFF_FFFF_FFFF) it wraps around to zero and continues counting upwards.

14.2 Registers

Table 14-1: Timers Register Map

Address	Read Location	Write Location	Size	Reset Value
0x8081_0000	Timer1Load	Timer1Load	16 bits	0
0x8081_0004	Timer1Value	-	16 bits	0
0x8081_0008	Timer1Control	Timer1Control	8 bits	0
0x8081_000C	Reserved	Timer1Clear	1 bit	-
0x8081_0020	Timer2Load	Timer2Load	16 bits	0
0x8081_0024	Timer2Value	-	16 bits	0
0x8081_0028	Timer2Control	Timer2Control	8 bits	0
0x8081_002C	Reserved	Timer2Clear	1 bit	-
0x8081_0060	Timer4ValueLow	-	32	0
0x8081_0064	Timer4Enable / Timer4ValueHigh	Timer4Enable	9	0
0x8081_0080	Timer3Load	Timer3Load	32 bits	0
0x8081_0084	Timer3Value	-	32 bits	0
0x8081_0088	Timer3Control	Timer3Control	32 bits	0
0x8081_008C	Reserved	Timer3Clear	1 bit	-
0x8081_0010	Reserved	Reserved	-	-
0x8081_0030	Reserved	Reserved	-	-
0x8081_0040	Reserved	Reserved	-	-
0x8081_0090	Reserved	Reserved	-	-

Register Descriptions

Timer1Load, Timer2Load

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Load															

Address:

Timer1 - 0x8081_0000 - Read/Write
Timer2 - 0x8081_0020 - Read/Write

Reset Value:

0x0000_0000

Definition:

The Load register contains the initial value of the timer and is also used as the reload value in periodic timer mode. The timer is loaded by writing to the Load register when the timer is disabled. The Timer Value register is updated with the Timer Load value as soon as the Timer Load register is written. The Load

register should not be written after the Timer is enabled because this causes the Timer Value register to be updated with an undetermined value.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

Load: Initial load value of the timer.

Timer3Load

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Load															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Load															

Address:

Timer3 - 0x8081_0080 - Read/Write

Reset Value:

0x0000_0000

Definition:

The Load register contains the initial value of the timer and is also used as the reload value in periodic timer mode. The timer is loaded by writing to the Load register when the timer is disabled. The Timer Value register is updated with the Timer Load value as soon as the Timer Load register is written to. The Load register should not be written to after the Timer is enabled as this causes the Timer Value register to be updated with an undetermined value.

Bit Descriptions:

Load: Initial load value of the timer.

Timer1Value, Timer2Value

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value															

Address:

Timer1 - 0x8081_0004 - Read Only
Timer2 - 0x8081_0024 - Read Only

Reset Value:

0x0000_0000

Definition:

The Value location gives the current value of the timer. When the Timer Load register is written to, the Value register is also updated with this Load value.

Bit Descriptions:

RSVD: Reserved. Unknown During Read.

Value: Current value of the timer.

Timer3Value

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Value															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value															

Address:

Timer3 - 0x8081_0084 - Read Only

Reset Value:

0x0000_0000

Definition:

The Value location gives the current value of the timer. When the Timer Load register is written to, the Value register is also updated with this Load value.

Bit Descriptions:

Value: Current value of the timer.

Timer1Clear, Timer2Clear, Timer3Clear

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															

Address:

Timer1 - 0x8081_000C - Write Only
Timer2 - 0x8081_002C - Write Only
Timer3 - 0x8081_008C - Write Only

Reset Value:

Not defined.

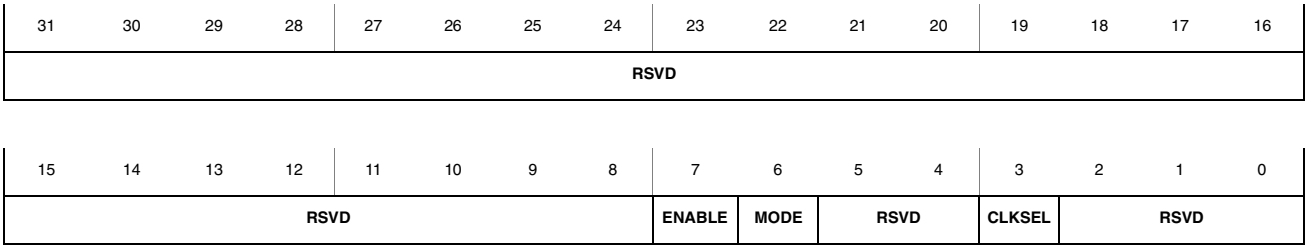
Definition:

Writing any value to the Clear location clears an interrupt generated by the timer.

Bit Descriptions:

RSVD: This register has no readable bits. It is just a write trigger.

Timer1Control, Timer2Control, Timer3Control



Address:

Timer1 - 0x8081_0008 - Read/Write
Timer2 - 0x8081_0028 - Read/Write
Timer3 - 0x8081_0088 - Read/Write

Reset Value:

0x0000_0000

Definition:

The Control register provides enable/disable and mode configurations for the timer.

Bit Descriptions:

RSVD: Reserved. Unknown during a Read operation.

ENABLE: Timer enable bit. This bit must be set to “1” to enable the timer. When the timer is disabled, its clock sources are turned off. Before re-enabling the timer, its Load register must be written to again.

MODE: This bit sets the mode of operation of the timer. When set to 1, the timer is in periodic timer mode and when set to “0”, the timer is in free running mode.

CLKSEL: When set to “1”, the 508 kHz clock is selected and when set to “0”, the 2 kHz clock is selected.

Timer4ValueLow

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Value															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value															

Address:

Timer4 - 0x8081_0060 - Read Only

Reset Value:

0x0000_0000

Definition:

This read-only register contains the low word of the time stamp debug timer (Timer4). When this register is read, the high byte of the Timer4 counter is saved in the Timer4ValueHigh register.

Bit Descriptions:

Value: Read Only Low Word of the Timer4 counter.

Timer4ValueHigh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSVD															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							Enable	Value							

Address:

Timer4 - 0x8081_0064 - Read/Write

Reset Value:

0x0000_0000

Definition:

This is a 9-bit read/write register.

Enable is the only bit that matters during a register write. When set to “1”, the timer is enabled and begins to count upwards.

Timer4ValueHigh is a read-only value and contains the high byte of the Timer4 counter. Note that the Timer4ValueLow register must first be read to store the high byte of the TC4 in Timer4ValueHigh register.

Bit Descriptions:

RSVD:	Reserved. Unknown during a Read operation.
Enable:	Read/Write. Enable for Timer4.
Value:	Read only. High Byte of the Timer4 counter.