Section 21. UART

HIGHLIGHTS

This section of the manual contains the following topics:

21.1 INTRODUCTION

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the PIC32MX family of devices. The UART is a full-duplex, asynchronous communication channel that communicates with peripheral devices and personal computers through protocols such as RS-232, RS-485, LIN 1.2 and IrDA®. The module also supports the hardware flow control option, with UxCTS and UxRTS pins, and also includes the IrDA encoder and decoder.

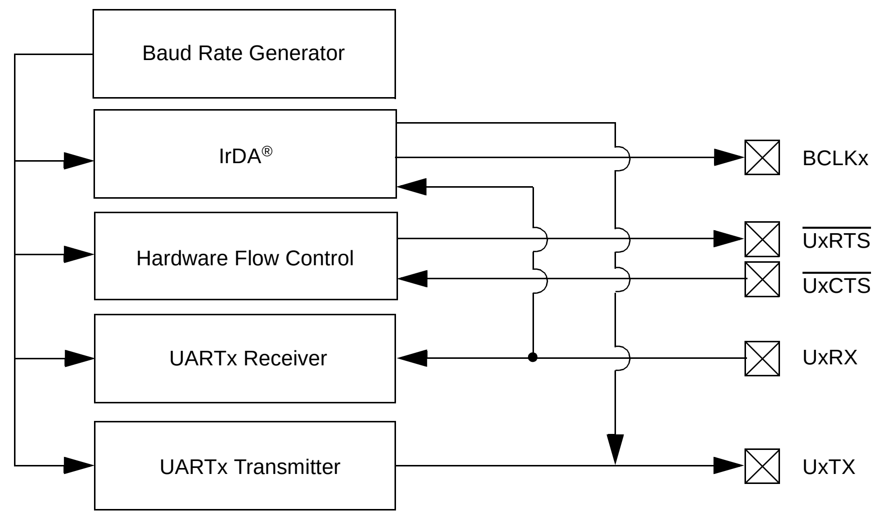
The primary features of the UART module are:

- Full-duplex, 8-bit or 9-bit data transmission
- Even, odd or no parity options (for 8-bit data)
- One or two Stop bits
- Hardware auto-baud feature
- Hardware flow control option
- Fully integrated Baud Rate Generator with 16-bit prescaler
- Baud rates ranging from 76 bps to 20 Mbps at 80 MHz
- 4-level-deep First-In First-Out (FIFO) transmit data buffer
- 4-level-deep FIFO receive data buffer
- Parity, framing and buffer overrun error detection
- Support for interrupt only on address detect (9th bit = 1)
- Separate transmit and receive interrupts
- Loopback mode for diagnostic support
- LIN 1.2 protocol support
- IrDA encoder and decoder with 16x baud clock output for external IrDA encoder/decoder support

A simplified block diagram of the UART is shown in Figure 21-1. The UART module consists of these important hardware elements:

- Baud Rate Generator
- Asynchronous transmitter
- Asynchronous receiver and IrDA encoder/decoder

Figure 21-1: UART Simplified Block Diagram



21.2 CONTROL REGISTERS

Note:	Each PIC32MX family device variant may have one or more UART modules. An ‘x’ used in the names of pins, control/Status bits, and registers denotes the particular module. Refer to the specific device data sheets for more details.

Each UART module consists of the following Special Function Registers (SFRs):

- UxMODE: Control Register for module ‘x’

  UxMODECLR, UxMODESET, UxMODEINV: Atomic Bit Manipulation Registers for UxMODE
- UxSTA: Status Register for module ‘x’

  UxSTACLR, UxSTASET, UxSTAINV: Atomic Bit Manipulation Registers for UxSTA
- UxTXREG: Transmit Buffer Register for module ‘x’
- UxRXREG: Receive Buffer Register for module ‘x’
- UxBRG: Baud Rate Generator Register for module ‘x’

  UxBRGCLR, UxBRGSET, UxBRGINV: Atomic Bit Manipulation Registers for UxBRG

Each UART module also has the associated bits for interrupt control:

- UxTXIE: Transmit Interrupt Enable Control Bit – in IEC0, IEC1 INT Registers
- UxTXIF: Transmit Interrupt Flag Status Bit – in IFC0, IFC1 INT Registers
- UxRXIE: Receive Interrupt Enable Control Bit – in IEC0, IEC1 INT Registers
- UxRXIF: Receive Interrupt Flag Status Bit – in IFC0, IFC1 INT Registers
- UxEIE: Error Interrupt Enable Control Bit – in IEC0, IEC1 INT Registers
- UxEIF: Error Interrupt Flag Status Bit – in IEC0, IEC1 INT Registers
- UxIP<2:0>Interrupt Priority Control Bits – in IPC6, IPC8 INT Registers
- UxIS<1:0>: Interrupt Subpriority Control Bits – in IPC6, IPC8 INT Registers

The following table summarizes all UART-related registers. Corresponding registers appear after the summary, followed by a detailed description of each register.

Table 21-1: UART SFRs Summary

Name		**Bit 31/23/15/7**	**Bit 30/22/14/6**	**Bit 29/21/13/5**	**Bit 28/20/12/4**	**Bit 27/19/11/3**	**Bit 26/18/10/2**	**Bit 25/17/9/1**	**Bit 24/16/8/0**
UxMODE	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	ON	FRZ	SIDL	IREN	RTSMD	—	UEN<1:0>	
	7:0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL<1:0>		STSEL
UxMODECLR	31:0	Write clears selected bits in UxMODE, read yields undefined value							
UxMODESET	31:0	Write sets selected bits in UxMODE, read yields undefined value							
UxMODEINV	31:0	Write inverts selected bits in UxMODE, read yields undefined value							
UxSTA	31:24	—	—	—	—	—	—	—	ADM_EN
	23:16	ADDR<7:0>							
	15:8	UTXISEL0<1:0>		UTXINV	URXEN	UTXBRK	UTXEN	UTXBF	TRMT
	7:0	URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	RXDA
UxSTACLR	31:0	Write clears selected bits in UxSTA, read yields undefined value							
UxSTASET	31:0	Write sets selected bits in UxSTA, read yields undefined value							
UxSTAINV	31:0	Write inverts selected bits in UxSTA, read yields undefined value							
UxTXREG	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	—	—	—	—	—	—	—	UTX8
	7:0	Transmit Register							

Table 21-1: UART SFRs Summary (Continued)

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
UxRXREG	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	—	—	—	—	—	—	—	RX8
	7:0	Receive Register							
UxBRG	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	BRG <15:8>							
	7:0	BRG<7:0>							
UxBRGCLR	31:0	Write clears selected bits in UxBRG, read yields undefined value							
UxBRGSET	31:0	Write sets selected bits in UxBRG, read yields undefined value							
UxBRGINV	31:0	Write inverts selected bits in UxBRG, read yields undefined value							
IFS0	31:24	I2C1MIF	I2C1SIF	I2C1BIF	U1TXIF	U1RXIF	U1EIF	SPI1RXIF	SPI1TXIF
	23:16	SPI1EIF	OC5IF	IC5IF	T5IF	INT4IF	OC4IF	IC4IF	T4IF
	15:8	INT3IF	OC3IF	IC3IF	T3IF	INT2IF	OC2IF	IC2IF	T2IF
	7:0	INT1IF	OC1IF	IC1IF	T1IF	INT0IF	CS1IF	CS0IF	CTIF
IFS0CLR	31:0	Write clears selected bits in IFSO, read yields undefined value							
IFS0SET	31:0	Write sets the selected bits in IFS0, read yields undefined value							
IFS0INV	31:0	Write inverts the selected bits in IFS, read yields undefined value							
IFS1	31:24	—	—	—	—	—	—	USBIF	FCEIF
	23:16	—	—	—	—	DMA3IF	DMA2IF	DMA1IF	DMA0IF
	15:8	RTCCIF	FSCMIF	I2C2MIF	I2C2SIF	I2C2BIF	U2TXIF	U2RXIF	U2EIF
	7:0	SPI2RXIF	SPI2TXIF	SPI2EIF	CMP2IF	CMP1IF	PMPIF	AD1IF	CNIF
IFS1CLR	31:0	Write clears the selected bits in IFS1, read yields undefined value							
IFS1SET	31:0	Write sets the selected bits in IFS1, read yields undefined value							
IFS1INV	31:0	Write inverts the selected bits in IFS1, read yields undefined value							
IEC0	31:24	I2C1MIE	I2C1SIE	I2C1BIE	U1TXIE	U1RXIE	U1EIE	SPI1RXIE	SPI1TXIE
	23:16	SPI1EIE	OC5IE	IC5IE	T5IE	INT4IE	OC4IE	IC4IE	T4IE
	15:8	INT3IE	OC3IE	IC3IE	T3IE	INT2IE	OC2IE	IC2IE	T2IE
	7:0	INT1IE	OC1IE	IC1IE	T1IE	INT0IE	CS1IE	CS0IE	CTIE
IEC0CLR	31:0	Write clears the selected bits in IEC0, read yields undefined value							
IEC0SET	31:0	Write sets the selected bits in IEC0, read yields undefined value							
IEC0INV	31:0	Write inverts the selected bits in IEC0, read yields undefined value							
IEC1	31:24	—	—	—	—	—	—	USBIE	FCEIE
	23:16	—	—	—	—	DMA3IE	DMA2IE	DMA1IE	DMA0IE
	15:8	RTCCIE	FSCMIE	I2C2MIE	I2C2SIE	I2C2BIE	U2TXIE	U2RXIE	U2EIE
	7:0	SPI2RXIE	SPI2TXIE	SPI2EIE	CMP2IE	CMP1IE	PMPIE	AD1IE	CNIE
IEC1CLR	31:0	Write clears the selected bits in IEC1, read yields undefined value							
IEC1SET	31:0	Write sets the selected bits in IEC1, read yields undefined value							
IEC1INV	31:0	Write inverts the selected bits in IEC1, read yields undefined value							
IPC6	31:24	—	—	—	AD1IP<2:0>			AD1IS<1:0>	
	23:16	—	—	—	CNIP<2:0>			CNIS<1:0>	
	15:8	—	—	—	I2C1IP<2:0>			I2C1IS<1:0>	
	7:0	—	—	—	U1IP<2:0>			U1IS<1:0>	

Table 21-1: UART SFRs Summary (Continued)

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
IPC6CLR	31:0	Write clears the selected bits in IPC6, read yields undefined value							
IPC6SET	31:0	Write sets the selected bits in IPC6, read yields undefined value							
IPC6INV	31:0	Write inverts the selected bits in IPC6, read yields undefined value							
IPC8	31:24	—	—	—	RTCCIP<2:0>			RTCCIS<1:0>	
	23:16	—	—	—	FSCMIP<2:0>			FSCMIS<1:0>	
	15:8	—	—	—	I2C2IP<2:0>			I2C2IS<1:0>	
	7:0	—	—	—	U2IP<2:0>			U2IS<1:0>	
IPC8CLR	31:0	Write clears the selected bits in IPC8, read yields undefined value							
IPC8SET	31:0	Write sets the selected bits in IPC8, read yields undefined value							
IPC8INV	31:0	Write inverts the selected bits in IPC8, read yields undefined value							

Register 21-1: UxMODE: UART ‘x’ Mode Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	r-x	R/W-0	R/W-0
ON	FRZ	SIDL	IREN	RTSMD	—	UEN<1:0>	
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL<1:0>		STSEL
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16 **Reserved:** Write ‘0’; ignore read

bit 15 **ON:** UARTx Enable bit

1 = UARTx is enabled; UARTx pins are controlled by UARTx as defined by UEN<1:0> and UTXEN control bits
0 = UARTx is disabled, all UARTx pins are controlled by corresponding PORT TRIS and LAT bits; UARTx power consumption is minimal

Note: When using 1:1 PBCLK divisor, the user’s software should not read/write the peripheral’s SFRs in the SYSCLK cycle immediately following the instruction that clears the module’s ON bit.

bit 14 **FRZ:** Freeze in Debug Exception Mode bit

1 = Freeze operation when CPU is in Debug Exception mode
0 = Continue operation when CPU is in Debug Exception mode
Note: FRZ is writable in Debug Exception mode only, it is forced to ‘0’ in Normal mode.

bit 13 **SIDL:** Stop in SLEEP Mode bit

1 = Discontinue operation when device enters in SLEEP mode
0 = Continue operation in SLEEP mode

bit 12 **IREN:** IrDA Encoder and Decoder Enable bit

1 = IrDA is enabled
0 = IrDA is disabled

bit 11 **RTSMD:** Mode Selection for UxRTS Pin bit

1 = UxRTS pin is in Simplex mode
0 = UxRTS pin is in Flow Control mode

bit 10 **Reserved:** Write ‘0’; ignore read

bit 9-8 **UEN<1:0>:** UARTx Enable bits

11 = UxTX, UxRX, and UxBCLK pins are enabled and used; CTS pin is controlled by PORT latches
10 = UxTX, UxRX, UxCTS, and UxRTS pins are enabled and used
01 = UxTX, UxRX and UxRTS pins are enabled and used; UxCTS pin is controlled by PORT latches
00 = UxTX and UxRX pins are enabled and used; UxCTS and UxRTS/UxBCLK pins are controlled by PORT latches

Register 21-1: UxMODE: UART 'x' Mode Register (Continued)

bit 7	**WAKE:** Enable Wake-up on Start bit Detect During SLEEP Mode bit `1` = Wake-up enabled `0` = Wake-up disabled
bit 6	**LPBACK:** UARTx Loopback Mode Select bit `1` = Enable Loopback mode `0` = Loopback mode is disabled
bit 5	**ABAUD:** Auto-Baud Enable bit `1` = Enable baud rate measurement on the next character – requires reception of Sync character (0x55); cleared by hardware upon completion `0` = Baud rate measurement disabled or completed
bit 4	**RXINV:** Receive Polarity Inversion bit `1` = UxRX IDLE state is '`0`' `0` = UxRX IDLE state is '`1`'
bit 3	**BRGH:** High Baud Rate Enable bit `1` = High-Speed mode – 4x baud clock enabled `0` = Standard Speed mode – 16x baud clock enabled
bit 2-1	**PDSEL<1:0>:** Parity and Data Selection bits `11` = 9-bit data, no parity `10` = 8-bit data, odd parity `01` = 8-bit data, even parity `00` = 8-bit data, no parity
bit 0	**STSEL:** Stop Selection bit `1` = 2 Stop bits `0` = 1 Stop bit

Register 21-2: UxMODECLR: UART 'x' Mode Clear Register

Write clears selected bits in UxMODE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in UxMODE**

A write of '`1`' in one or more bit positions clears the corresponding bit(s) in UxMODE register and does not affect unimplemented or read-only bits. A write of '`0`' will not affect the register.

Example: `UxMODECLR = 0x00008001` will clear bits 15 and 0 in UxMODE register.

Register 21-3: UxMODESET: UART 'x' Mode Set Register

Write sets selected bits in UxMODE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in UxMODE**

A write of '`1`' in one or more bit positions sets the corresponding bit(s) in UxMODE register and does not affect unimplemented or read-only bits. A write of '`0`' will not affect the register.

Example: `UxMODE = 0x00008001` will set bits 15 and 0 in UxMODE register.

Register 21-4: UxMODEINV: UART 'x' Mode Invert Register

Write inverts selected bits in UxMODE, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in UxMODE**

A write of '`1`' in one or more bit positions inverts the corresponding bit(s) in UxMODE register and does not affect unimplemented or read-only bits. A write of '`0`' will not affect the register.

Example: `UxMODEINV = 0x00008001` will invert bits 15 and 0 in UxMODE register.

Register 21-5: UxSTA: UART ‘x’ Status and Control Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	R/W-0
—	—	—	—	—	—	—	ADM_EN
bit 31							bit 24

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADDR<7:0>							
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-1
UTXISEL0<1:0>		UTXINV	URXEN	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/W-0	R-0
URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	RXDA
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-25 **Reserved:** Write ‘0’; ignore read

bit 24 **ADM_EN:** Automatic Address Detect Mode Enable bit
1 = Automatic Address Detect mode is enabled
0 = Automatic Address Detect mode is disabled

bit 23-16 **ADDR<7:0>:** Automatic Address Mask bits
When ADM_EN bit is ‘1’, this value defines the address character to use for automatic address detection.

bit 15-14 **UTXISEL0<1:0>:** Tx Interrupt Mode Selection bits
11 = Reserved, do not use
10 = Interrupt is generated when the transmit buffer becomes empty
01 = Interrupt is generated when all characters are transmitted
00 = Interrupt is generated when the transmit buffer contains at least one empty space

bit 13 **UTXINV:** Transmit Polarity Inversion bit
If IrDA mode is disabled (i.e., IREN (UxMODE<12>) is ‘0’)
1 = UxTX IDLE state is ‘0’
0 = UxTX IDLE state is ‘1’
If IrDA mode is enabled (i.e., IREN (UxMODE<12>) is ‘1’)
1 = IrDA encoded UxTX IDLE state is ‘1’
0 = IrDA encoded UxTX IDLE state is ‘0’

bit 12 **URXEN:** Receiver Enable bit
1 = UARTx receiver is enabled, UxRX pin controlled by UARTx (if ON = 1)
0 = UARTx receiver is disabled, the UxRX pin is ignored by the UARTx module. UxRX pin controlled by port.

bit 11 **UTXBRK:** Transmit Break bit
1 = Send Break on next transmission – Start bit followed by twelve ‘0’ bits, followed by Stop bit; cleared by hardware upon completion
0 = Break transmission is disabled or completed

bit 10 **UTXEN:** Transmit Enable bit
1 = UARTx transmitter enabled, UxTX pin controlled by UARTx (if ON = 1)
0 = UARTx transmitter disabled, any pending transmission is aborted and buffer is reset. UxTX pin controlled by port.

Register 21-5: UxSTA: UART 'x' Status and Control Register (Continued)

bit 9	**UTXBF:** Transmit Buffer Full Status bit (read-only) `1` = Transmit buffer is full `0` = Transmit buffer is not full, at least one more character can be written
bit 8	**TRMT:** Transmit Shift Register is Empty bit (read-only) `1` = Transmit shift register is empty and transmit buffer is empty (the last transmission has completed) `0` = Transmit shift register is not empty, a transmission is in progress or queued in the transmit buffer
bit 7-6	**URXISEL<1:0>:** Receive Interrupt Mode Selection bit `11` = Interrupt flag bit is set when receive buffer is full (i.e., has 4 data characters) `10` = Interrupt flag bit is set when receive buffer is 3/4 full (i.e., has 3 data characters) `0x` = Interrupt flag bit is set when a character is received
bit 5	**ADDEN:** Address Character Detect bit (bit 8 of received data = `1`) `1` = Address Detect mode enabled. If 9-bit mode is not selected, this control bit has no effect. `0` = Address Detect mode disabled
bit 4	**RIDLE:** Receiver IDLE bit (read-only) `1` = Receiver is IDLE `0` = Data is being received
bit 3	**PERR:** Parity Error Status bit (read-only) `1` = Parity error has been detected for the current character `0` = Parity error has not been detected
bit 2	**FERR:** Framing Error Status bit (read-only) `1` = Framing error has been detected for the current character `0` = Framing error has not been detected
bit 1	**OERR:** Receive Buffer Overrun Error Status bit. This bit is set in hardware, can only be cleared (= `0`) in software. `1` = Receive buffer has overflowed `0` = Receive buffer has not overflowed **Note:** Clearing a previously set OERR bit resets the receiver buffer and RSR to empty state.
bit 0	**RXDA:** Receive Buffer Data Available bit (read-only) `1` = Receive buffer has data, at least one more character can be read `0` = Receive buffer is empty

Register 21-6: UxSTACLR: UART ‘x’ Status Clear Register

Write clears selected bits in UxSTA, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in UxSTA**

A write of ‘`1`’ in one or more bit positions clears the corresponding bit(s) in UxSTA register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxSTACLR = 0x00008001` will clear bits 15 and 0 in UxSTA register.

Register 21-7: UxSTASET: UART ‘x’ Status Set Register

Write sets selected bits in UxSTA, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in UxSTA**

A write of ‘`1`’ in one or more bit positions sets the corresponding bit(s) in UxSTA register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxSTA = 0x00008001` will set bits 15 and 0 in UxSTA register.

Register 21-8: UxSTAINV: UART ‘x’ Status Invert Register

Write inverts selected bits in UxSTA, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in UxSTA**

A write of ‘`1`’ in one or more bit positions inverts the corresponding bit(s) in UxSTA register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxSTAINV = 0x00008001` will invert bits 15 and 0 in UxSTA register.

Register 21-9: UxTXREG: UART 'x' Transmit Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

r-x	r-x	r-x	r-x	r-x	r-x	r-x	R/W-0
—	—	—	—	—	—	—	TX8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: ('0', '1', x = Unknown)		

bit 31-9	**Reserved:** Write '0'; ignore read
bit 8	**TX8:** Data bit 8 of the character to be transmitted (in 9-bit mode)
bit 7-0	**TX<7:0>:** Data bits 7-0 of the character to be transmitted

Register 21-10: UxRXREG: UART 'x' Receive Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

r-x	r-x	r-x	r-x	r-x	r-x	r-x	R-0
—	—	—	—	—	—	—	RX8
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RX<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: ('0', '1', x = Unknown)		

bit 31-9	**Reserved:** Write '0'; ignore read
bit 8	**RX8:** Data bit 8 of the received character (in 9-bit mode)
bit 7-0	**RX<7:0>:** Data bits 7-0 of the received character

Register 21-11: UxBRG: UART ‘x’ Baud Rate Register

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31							bit 24

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BRG<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BRG<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 31-16	**Reserved:** Write ‘0’; ignore read
bit 15-0	**BRG<15:0>:** Baud Rate Divisor bits

Register 21-12: UxBRGCLR: UART ‘x’ BRG Clear Register

Write clears selected bits in UxBRG, read yields undefined value	
bit 31	bit 0

bit 31-0 **Clears selected bits in UxBRG**

A write of ‘`1`’ in one or more bit positions clears the corresponding bit(s) in UxBRG register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxBRGCLR = 0x00008001` will clear bits 15 and 0 in UxBRG register.

Register 21-13: UxBRGSET: UART ‘x’ BRG Set Register

Write sets selected bits in UxBRG, read yields undefined value	
bit 31	bit 0

bit 31-0 **Sets selected bits in UxBRG**

A write of ‘`1`’ in one or more bit positions sets the corresponding bit(s) in UxBRG register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxBRG = 0x00008001` will set bits 15 and 0 in UxBRG register.

Register 21-14: UxBRGINV: UART ‘x’ BRG Invert Register

Write inverts selected bits in UxBRG, read yields undefined value	
bit 31	bit 0

bit 31-0 **Inverts selected bits in UxBRG**

A write of ‘`1`’ in one or more bit positions inverts the corresponding bit(s) in UxBRG register and does not affect unimplemented or read-only bits. A write of ‘`0`’ will not affect the register.

Example: `UxBRGINV = 0x00008001` will invert bits 15 and 0 in UxBRG register.

Register 21-15: IFS0: Interrupt Flag Status Register 0[1]

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
I2C1MIF	I2C1SIF	I2C1BIF	U1TXIF	U1RXIF	U1EIF	SPI1RXIF	SPI1TXIF
bit 31							bit 24

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPI1EIF	OC5IF	IC5IF	T5IF	INT4IF	OC4IF	IC4IF	T4IF
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT3IF	OC3IF	IC3IF	T3IF	INT2IF	OC2IF	IC2IF	T2IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT1IF	OC1IF	IC1IF	T1IF	INT0IF	CS1IF	CS0IF	CTIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 28 **U1TXIF:** UART 1 Transmitter Interrupt Request Flag bit

`1` = Interrupt request has occurred
`0` = No interrupt request has occurred

bit 27 **U1RXIF:** UART 1 Receiver Interrupt Request Flag bit

`1` = Interrupt request has occurred
`0` = No interrupt request has occurred

bit 26 **U1EIF:** UART 1 Error Interrupt Request Flag bit

`1` = Interrupt request has occurred
`0` = No interrupt request has occurred

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

Register 21-16: IFS1: Interrupt Flag Status Register 1(1)

r-x	r-x	r-x	r-x	r-x	r-x	R/W-0	R/W-0
—	—	—	—	—	—	USBIF	FCEIF
bit 31							bit 24

r-0	r-0	r-0	r-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	DMA3IF	DMA2IF	DMA1IF	DMA0IF
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RTCCIF	FSCMIF	I2C2MIF	I2C2SIF	I2C2BIF	U2TXIF	U2RXIF	U2EIF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPI2RXIF	SPI2TXIF	SPI2EIF	CMP2IF	CMP1IF	PMPIF	AD1IF	CNIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 10 **U2TXIF:** UART 2 Transmitter Interrupt Request bit

1 = Interrupt request has occurred
0 = No interrupt request has occurred

bit 9 **U2RXIF:** UART 2 Receiver Interrupt Request Flag bit

1 = Interrupt request has occurred
0 = No interrupt request has occurred

bit 8 **U2EIF:** UART 2 Error Interrupt Request bit

1 = Interrupt request has occurred
0 = No interrupt request has occurred

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

Register 21-17: IEC0: Interrupt Enable Control Register 0(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
I2C1MIE	I2C1SIE	I2C1BIE	U1TXIE	U1RXIE	U1EIE	SPI1RXIE	SPI1TXIE
bit 31							bit 24

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPI1EIE	OC5IE	IC5IE	T5IE	INT4IE	OC4IE	IC4IE	T4IE
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT3IE	OC3IE	IC3IE	T3IE	INT2IE	OC2IE	IC2IE	T2IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT1IE	OC1IE	IC1IE	T1IE	INT0IE	CS1IE	CS0IE	CTIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 28 **U1TXIE:** UART 1 Transmitter Interrupt Enable bit

`1` = Interrupt is enabled
`0` = Interrupt is disabled

bit 27 **U1RXIE:** UART 1 Receiver Interrupt Enable bit

`1` = Interrupt is enabled
`0` = Interrupt is disabled

bit 26 **U1EIE:** UART 1 Error Interrupt Enable bit

`1` = Interrupt is enabled
`0` = Interrupt is disabled

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

Register 21-18: IEC1: Interrupt Enable Control Register 1[(1)]

r-x	r-x	r-x	r-x	r-x	r-x	R/W-0	R/W-0
—	—	—	—	—	—	USBIE	FCEIE
bit 31							bit 24

r-0	r-0	r-0	r-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	DMA3IE	DMA2IE	DMA1IE	DMA0IE
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RTCCIE	FSCMIE	I2C2MIE	I2C2SIE	I2C2BIE	U2TXIE	U2RXIE	U2EIE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPI2RXIE	SPI2TXIE	SPI2EIE	CMP2IE	CMP1IE	PMPIE	AD1IE	CNIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 10	**U2TXIE:** UART 2 Transmitter Interrupt Request bit `1` = Interrupt is enabled `0` = Interrupt is disabled
bit 9	**U2RXIE:** UART 2 Receiver Interrupt Enable bit `1` = Interrupt is enabled `0` = Interrupt is disabled
bit 8	**U2EIE:** UART 2 Error Interrupt Request bit `1` = Interrupt is enabled `0` = Interrupt is disabled

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

Register 21-19: IPC6; Interrupt Priority Control Register 6[(1)]

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	AD1IP<2:0>			AD1IS<1:0>	
bit 31							bit 24

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	CNIP<2:0>			CNIS<1:0>	
bit 23							bit 16

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	I2C1IP<2:0>			I2C1IS<1:0>	
bit 15							bit 8

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	U1IP<2:0>			U1IS<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 4-2	**U1IP<2:0>:** UART 1 Interrupt Priority bits `111` = Interrupt Priority is 7 `110` = Interrupt Priority is 6 `101` = Interrupt Priority is 5 `100` = Interrupt Priority is 4 `011` = Interrupt Priority is 3 `010` = Interrupt Priority is 2 `001` = Interrupt Priority is 1 `000` = Interrupt is disabled
bit 1-0	**U1IS<1:0>:** UART 1 Interrupt Subpriority bits `11` = Interrupt Subpriority is 3 `10` = Interrupt Subpriority is 2 `01` = Interrupt Subpriority is 1 `00` = Interrupt Subpriority is 0

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

Register 21-20: IPC8: Interrupt Priority Control Register[(1)]

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RTCCIP<2:0>			RTCCIS<1:0>	
bit 31							bit 24

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	FSCMIP<2:0>			FSCMIS<1:0>	
bit 23							bit 16

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	I2C2IP<2:0>			I2C2IS<1:0>	
bit 15							bit 8

r-x	r-x	r-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	U2IP<2:0>			U2IS<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: (‘0’, ‘1’, x = Unknown)		

bit 4-2 **U2IP<2:0>:** UART 2 Interrupt Priority bits

`111` = Interrupt Priority is 7
`110` = Interrupt Priority is 6
`101` = Interrupt Priority is 5
`100` = Interrupt Priority is 4
`011` = Interrupt Priority is 3
`010` = Interrupt Priority is 2
`001` = Interrupt Priority is 1
`000` = Interrupt is disabled

bit 1-0 **U2IS<1:0>:** UART 2 Subpriority bits

`11` = Interrupt Subpriority is 3
`10` = Interrupt Subpriority is 2
`01` = Interrupt Subpriority is 1
`00` = Interrupt Subpriority is 0

Note 1: Shaded bit names in this Interrupt register control other PIC32MX peripherals and are not related to the UART.

21.3 UART BAUD RATE GENERATOR

The UART module includes a dedicated 16-bit Baud Rate Generator. The UxBRG register controls the period of a free-running 16-bit timer. Equation 21-1 shows the formula for computation of the baud rate with BRGH = `0`.

Equation 21-1: UART Baud Rate with BRGH = 0

$$\text{Baud Rate} = \frac{F_{PB}}{16 \bullet (\text{UxBRG} + 1)}$$

$$\text{UxBRG} = \frac{F_{PB}}{16 \bullet \text{Baud Rate}} - 1$$

Note: F_{PB} denotes the PBCLK frequency.

Example 21-1 shows the calculation of the baud rate error for the following conditions:

- F_{PB} = 4 MHz
- Desired Baud Rate = 9600

Example 21-1: Baud Rate Error Calculation (BRGH = 0)

```
Desired Baud Rate     =   FPB/(16 (UxBRG + 1))
Solving for UxBRG value:
    UxBRG             =   ( (FPB/Desired Baud Rate)/16) – 1
    UxBRG             =   ((4000000/9600)/16) – 1
    UxBRG             =   [25.042] = 25
Calculated Baud Rate  =   4000000/(16 (25 + 1))
                      =   9615
Error                 =   (Calculated Baud Rate – Desired Baud Rate)
            Desired Baud Rate
                      =   (9615 – 9600)/9600
                      =   0.16%
```

The maximum possible baud rate (BRGH = `0`) is F_{PB}/16 (for UxBRG = `0`), and the minimum possible baud rate is F_{PB} /16 * 65536).

Equation 21-2 shows the formula for computation of the baud rate with BRGH = `1`.

Equation 21-2: UART Baud Rate with BRGH = 1

$$\text{Baud Rate} = \frac{F_{PB}}{4 \bullet (\text{UxBRG} + 1)}$$

$$\text{UxBRG} = \frac{F_{PB}}{4 \bullet \text{Baud Rate}} - 1$$

Note: F_{PB} denotes the PBCLK frequency.

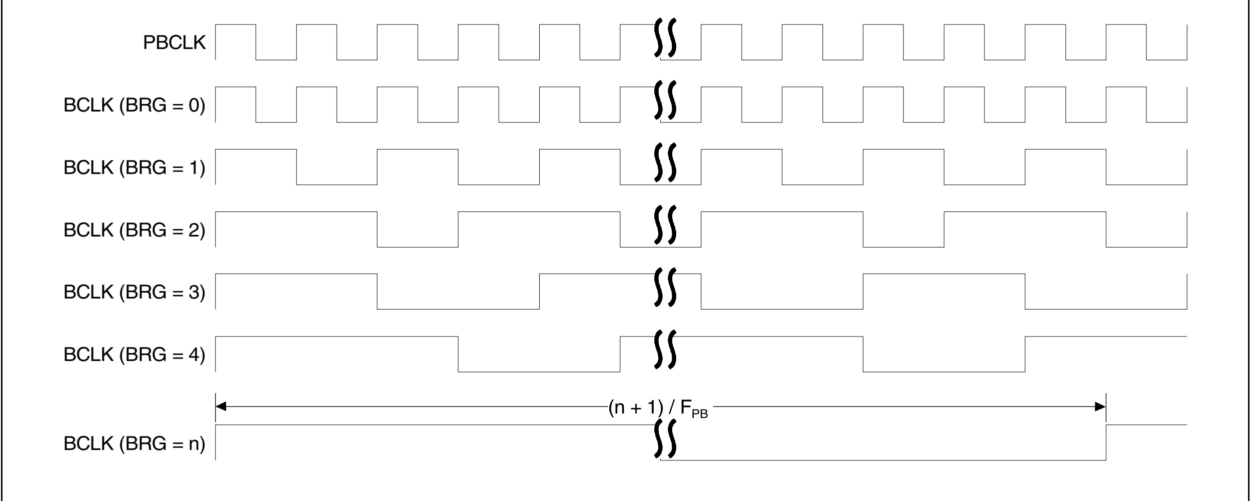
The maximum possible baud rate (BRGH = `1`) is F_{PB} /4 (for UxBRG = `0`), and the minimum possible baud rate is F_{PB}/(4 * 65536).

Writing a new value to the UxBRG register causes the baud rate counter to reset (clear). This ensures that the BRG does not wait for a timer overflow before it generates the new baud rate.

21.3.1 BCLKx Output

The BCLKx pin outputs the 16x baud clock if the UART and BCLKx output are enabled (UxMODE.UEN<1:0> = 11). This feature is used for external IrDA encoder/decoder support (refer to Figure 21-2). BCLKx output stays low during SLEEP mode. BCLKx is forced as an output as long as UART is kept in this mode (UxMODE.UEN<1:0> = 11), irrespective of PORTx and TRISx latch bits.

Figure 21-2: BCLKx Output vs. UxBRG Programming



21.3.2 Baud Rate Tables

UART baud rates are provided in Table 21-2 for common peripheral bus frequencies (F_{PB}). The minimum and maximum baud rates for each frequency are also provided.

Table 21-2: UART Baud Rates (UxMODE.BRGH = ‘0’, no PLL)

Target Baud Rate	Peripheral Bus Clock: 40 MHz			Peripheral Bus Clock: 33 MHz			Peripheral Bus Clock: 30 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	0.00%	22726.0	110.0	0.0%	18749.0	110.0	0.0%	17044.0
300	300.0	0.00%	8332.0	300.0	0.0%	6874.0	300.0	0.0%	6249.0
1200	1200.2	0.02%	2082.0	1199.8	0.0%	1718.0	1199.6	0.0%	1562.0
2400	2399.2	-0.03%	1041.0	2401.0	0.0%	858.0	2400.8	0.0%	780.0
9600	9615.4	0.16%	259.0	9593.0	-0.1%	214.0	9615.4	0.2%	194.0
19.2 K	19230.8	0.16%	129.0	19275.7	0.4%	106.0	19132.7	-0.4%	97.0
38.4 K	38461.5	0.16%	64.0	38194.4	-0.5%	53.0	38265.3	-0.4%	48.0
56 K	55555.6	-0.79%	44.0	55743.2	-0.5%	36.0	56818.2	1.5%	32.0
115 K	113636.4	-1.19%	21.0	114583.3	-0.4%	17.0	117187.5	1.9%	15.0
250 K	250000.0	0.00%	9.0	257812.5	3.1%	7.0			
300 K				294642.9	-1.8%	6.0			
500 K	500000.0	0.00%	4.0	515625.0	3.1%	3.0			
Min. Rate	38.1	0.0%	65535	31.5	0.0%	65535	28.6	0.0%	65535
Max. Rate	2500000	0.0%	0	2062500	0.0%	0	1875000	0.0%	0

Target Baud Rate	Peripheral Bus Clock: 25 MHz			Peripheral Bus Clock: 20 MHz			Peripheral Bus Clock: 18.432 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	0.00%	14204.0	110.0	0.0%	11363.0	110.0	0.0%	10472.0
300	300.0	0.01%	5207.0	300.0	0.0%	4166.0	300.0	0.0%	3839.0
1200	1200.1	0.01%	1301.0	1199.6	0.0%	1041.0	1200.0	0.0%	959.0
2400	2400.2	0.01%	650.0	2399.2	0.0%	520.0	2400.0	0.0%	479.0
9600	9585.9	-0.15%	162.0	9615.4	0.2%	129.0	9600.0	0.0%	119.0
19.2 K	19290.1	0.47%	80.0	19230.8	0.2%	64.0	19200.0	0.0%	59.0
38.4 K	38109.8	-0.76%	40.0	37878.8	-1.4%	32.0	38400.0	0.0%	29.0
56 K	55803.6	-0.35%	27.0	56818.2	1.5%	21.0	54857.1	-2.0%	20.0
115 K	111607.1	-2.95%	13.0	113636.4	-1.2%	10.0	115200.0	0.2%	9.0
250 K				250000.0	0.0%	4.0			
300 K									
500 K									
Min. Rate	23.8	0.0%	65535	19	0.0%	65535	18	0.0%	65535
Max. Rate	1562500	0.0%	0	1250000	0.0%	0	1152000	0.0%	0

Target Baud Rate	Peripheral Bus Clock: 16 MHz			Peripheral Bus Clock: 12 MHz			Peripheral Bus Clock: 10 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	0.00%	9090.0	110.0	0.0%	6817.0	110.0	0.0%	5681.0
300	300.0	0.01%	3332.0	300.0	0.0%	2499.0	300.0	0.0%	2082.0
1200	1200.5	0.04%	832.0	1200.0	0.0%	624.0	1199.6	0.0%	520.0
2400	2398.1	-0.08%	416.0	2396.2	-0.2%	312.0	2403.8	0.2%	259.0
9600	9615.4	0.16%	103.0	9615.4	0.2%	77.0	9615.4	0.2%	64.0
19.2 K	19230.8	0.16%	51.0	19230.8	0.2%	38.0	18939.4	-1.4%	32.0
38.4 K	38461.5	0.16%	25.0	37500.0	-2.3%	19.0	39062.5	1.7%	15.0
56 K	55555.6	-0.79%	17.0	57692.3	3.0%	12.0	56818.2	1.5%	10.0
115 K	111111.1	-3.38%	8.0			6.0			
250 K	250000.0	0.00%	3.0	250000.0	0.0%	2.0			
300 K									
500 K	500000.0	0.00%	1.0						
Min. Rate	15	0.0%	65535	11	0.0%	65535	10	0.0%	65535
Max. Rate	1000000	0.0%	0	750000	0.0%	0	625000	0.0%	0

Table 21-2: UART Baud Rates (UxMODE.BRGH = '0', no PLL) (Continued)

Target Baud Rate	Peripheral Bus Clock: 8 MHz			Peripheral Bus Clock: 5 MHz			Peripheral Bus Clock: 4 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	0.01%	4544.0	110.0	0.0%	2840.0	110.0	0.0%	2272.0
300	299.9	-0.02%	1666.0	299.9	0.0%	1041.0	300.1	0.0%	832.0
1200	1199.0	-0.08%	416.0	1201.9	0.2%	259.0	1201.9	0.2%	207.0
2400	2403.8	0.16%	207.0	2403.8	0.2%	129.0	2403.8	0.2%	103.0
9600	9615.4	0.16%	51.0	9469.7	-1.4%	32.0	9615.4	0.2%	25.0
19.2 K	19230.8	0.16%	25.0	19531.3	1.7%	15.0	19230.8	0.2%	12.0
38.4 K	38461.5	0.16%	12.0	39062.5	1.7%	7.0			
56 K	55555.6	-0.79%	8.0						
115 K									
250 K	250000.0	0.00%	1.0						
300 K									
500 K	500000.0	0.00%	0.0						
Min. Rate	8	0.0%	65535	5	0.0%	65535	4	0.0%	65535
Max. Rate	500000	0.0%	0	312500	0.0%	0	250000	0.0%	0

Target Baud Rate	Peripheral Bus Clock: 7.68 MHz			Peripheral Bus Clock: 7.15909 MHz			Peripheral Bus Clock: 5.0688 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	-0.01%	4363.0	110.0	0.0%	4067.0	110.0	0.0%	2879.0
300	300.0	0.00%	1599.0	300.1	0.0%	1490.0	300.0	0.0%	1055.0
1200	1200.0	0.00%	399.0	1199.6	0.0%	372.0	1200.0	0.0%	263.0
2400	2400.0	0.00%	199.0	2405.6	0.2%	185.0	2400.0	0.0%	131.0
9600	9600.0	0.00%	49.0	9520.1	-0.8%	46.0	9600.0	0.0%	32.0
19.2 K	19200.0	0.00%	24.0	19454.0	1.3%	22.0	18635.3	-2.9%	16.0
38.4 K	36923.1	-3.85%	12.0	37286.9	-2.9%	11.0	39600.0	3.1%	7.0
56 K	53333.3	-4.76%	8.0	55930.4	-0.1%	7.0			
115 K	120000.0	4.35%	3.0	111860.8	-2.7%	3.0			
250 K	240000.0	-4.00%	1.0						
300 K									
500 K									
Min. Rate	7	0.0%	65535	7	0.0%	65535	5	0.0%	65535
Max. Rate	480000	0.0%	0	447443	0.0%	0	316800	0.0%	0

Target Baud Rate	Peripheral Bus Clock: 3.579545 MHz			Peripheral Bus Clock: 3.072 MHz			Peripheral Bus Clock: 1.8432 MHz		
	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)	Actual Baud Rate	% Error	BRG Value (decimal)
110	110.0	-0.01%	2033.0	110.0	0.0%	1744.0	110.0	0.0%	1046.0
300	299.9	-0.04%	745.0	300.0	0.0%	639.0	300.0	0.0%	383.0
1200	1202.8	0.23%	185.0	1200.0	0.0%	159.0	1200.0	0.0%	95.0
2400	2405.6	0.23%	92.0	2400.0	0.0%	79.0	2400.0	0.0%	47.0
9600	9727.0	1.32%	22.0	9600.0	0.0%	19.0	9600.0	0.0%	11.0
19.2 K	18643.5	-2.90%	11.0	19200.0	0.0%	9.0	19200.0	0.0%	5.0
38.4 K	37286.9	-2.90%	5.0	38400.0	0.0%	4.0	38400.0	0.0%	2.0
56 K	55930.4	-0.12%	3.0						
115 K	111860.8	-2.73%	1.0						
250 K									
300 K									
500 K									
Min. Rate	3	0.0%	65535	3	0.0%	65535	2	0.0%	65535
Max. Rate	223722	0.0%	0	192000	0.0%	0	115200	0.0%	0

21.4 UART CONFIGURATION

The UART uses standard non-return-to-zero (NRZ) format (one Start bit, eight or nine data bits, and one or two Stop bits). Parity is supported by the hardware, and may be configured by the user as even, odd or no parity. The most common data format is 8 bits, no parity and one Stop bit (denoted as 8, N, 1), which is the default Power-on Reset (POR) setting. The number of data bits and Stop bits, and the parity, are specified in the PDSEL<1:0> (UxMODE<2:1>) and STSEL (UxMODE<0>) bits. An on-chip dedicated 16-bit Baud Rate Generator is used to derive standard baud rate frequencies from the oscillator. The UART transmits and receives the Least Significant bit (LSb) first. The UART's transmitter and receiver are functionally independent, but use the same data format and baud rate.

21.4.1 Enabling the UART

The UART module is enabled by setting the bits ON (UxMODE<15>), URXEN (UxSTA<12>), and UTXEN (UxSTA<10>). Once enabled, the UxTX and UxRX pins are configured as an output and an input, respectively, overriding the TRIS and PORT register bit settings for the corresponding I/O port pins. The UxTX pin is at logic '1' when a transmission is not taking place.

21.4.2 Disabling the UART

The UART module is disabled by clearing the ON bit. This is the default state after any Reset. If the UART is disabled, all UART pins operate as port pins under the control of their corresponding PORT and TRIS bits.

Disabling the UART module resets the buffers to empty states. Any data characters in the buffers are lost, and the baud rate counter is reset.

All error and status flags associated with the UART module are reset when the module is disabled. The RXDA, OERR, FERR, PERR, UTXEN, URXEN, UTXBRK and UTXBF bits in the UxSTA register are cleared, whereas the RIDLE and TRMT bits are set. Other control bits (including ADDEN, RXISEL<1:0> and UTXISEL0), as well as UxMODE and UxBRG registers, are not affected.

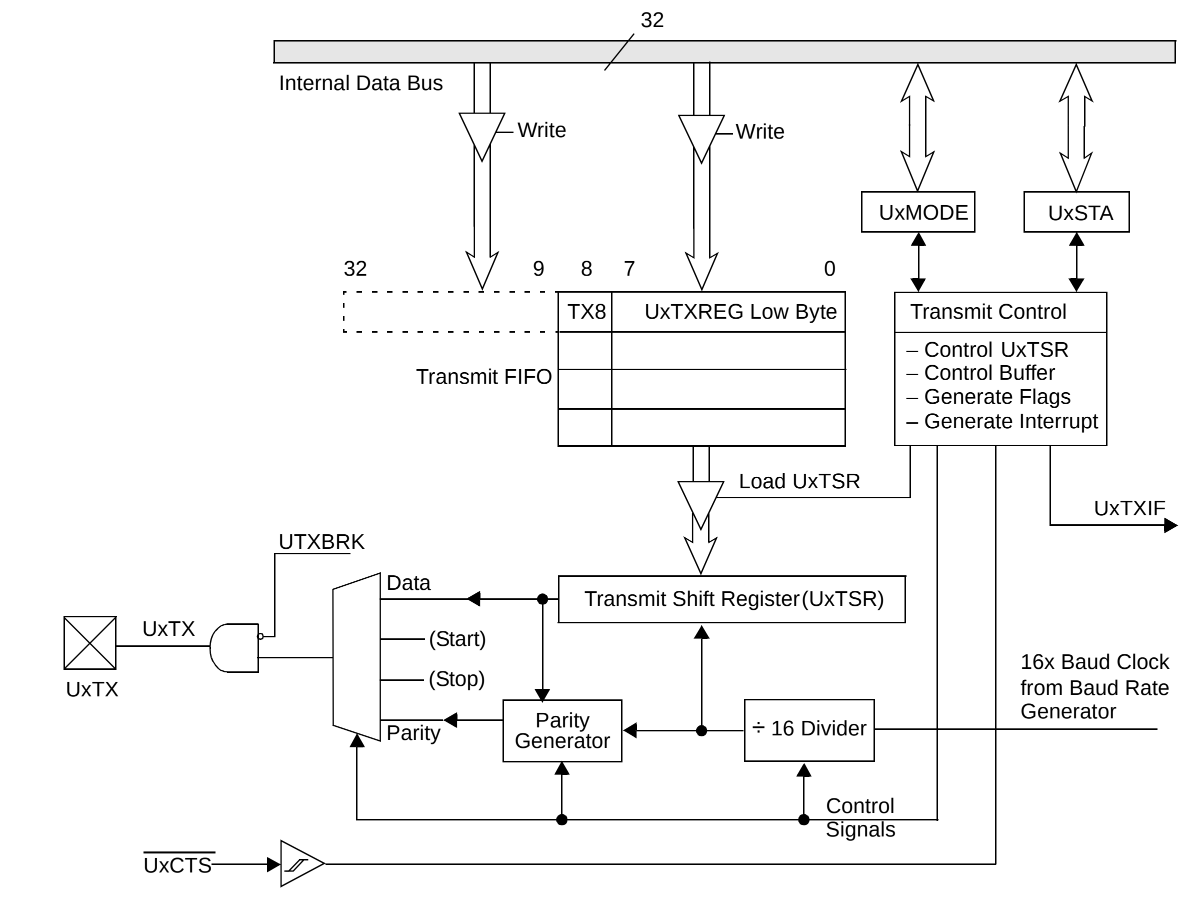
Clearing the ON bit while the UART is active aborts all pending transmissions and receptions and resets the module as defined above. Re-enabling the UART restarts the UART in the same configuration.

21.5 UART TRANSMITTER

Figure 21-3 shows the UART transmitter block diagram. The heart of the transmitter is the Transmit Shift register (UxTSR). UxTSR obtains its data from the transmit FIFO buffer, UxTXREG. The UxTXREG register is loaded with data in software. The UxTSR register is not loaded until the Stop bit is transmitted from the previous load. As soon as the Stop bit is transmitted, the UxTSR is loaded with new data from the UxTXREG register (if available).

Note: The UxTSR register is not mapped in memory, so it is not available to the user.

Figure 21-3: UART Transmitter Block Diagram



Note: 'x' denotes the UART number.

Transmission is enabled by setting the UTXEN enable bit (UxSTA<10>). The actual transmission will not occur until the UxTXREG register is loaded with data and the Baud Rate Generator UxBRG has produced a shift clock (see Figure 21-3). The transmission can also be started by first loading the UxTXREG register and then setting the UTXEN enable bit. Normally, when transmission is initially started, the UxTSR register is empty, so a transfer to the UxTXREG register results in an immediate transfer to the UxTSR. Clearing the UTXEN bit during a transmission causes the transmission to be aborted and resets the transmitter. As a result, the UxTX pin reverts to a high-impedance state.

To select 9-bit transmission, the PDSEL<1:0> bits, in the UxMODE<2:1>, should be set to '11' and the ninth bit should be written to the UTX8 bit (UxTXREG<8>). A word write should be performed to the UxTXREG so that all nine bits are written at the same time.

Note: There is no parity in the case of 9-bit data transmission.

21.5.1 Transmit Buffer (UxTXREG)

The transmit buffer is 9 bits wide and 4 levels deep. Together with the Transmit Shift registers (UxTSR), the user effectively has a 5-level-deep buffer. It is organized as FIFO. When the UxTXREG contents are transferred to the UxTSR register, the current buffer location becomes available for new data to be written, and the next buffer location is sourced to the UxTSR register. The UTXBF (UxSTA<9>) Status bit is set whenever the buffer is full. If a user attempts to write to a full buffer, the new data will not be accepted into the FIFO.

The FIFO is reset during any device Reset, but is not affected when the device enters a Power-Saving mode or wakes up from a Power-Saving mode.

21.5.2 Transmit Interrupt

The transmit interrupt flag (UxTXIF) is located in the corresponding interrupt flag status (IFS) register. The UTXISEL0 control bit (UxSTA<15:14>) determines when the UART will generate a transmit interrupt.

1. UTXISEL0<1:0> = `00`, the UxTXIF is set when a character is transferred from the transmit buffer to the Transmit Shift register (UxTSR). This implies at least one location is empty in the transmit buffer.
2. UTXISEL0<1:0> = `01`, the UxTXIF is set when the last character is shifted out of the Transmit Shift register (UxTSR). This implies that all the transmit operations are completed.
3. UTXISEL0<1:0> = `10`, the UxTXIF is set when the character is transferred to the Transmit Shift register (UxTSR) and the transmit buffer is empty.

The UxTXIF bit is set when the module is first enabled. The user should clear the UxTXIF bit in the ISR.

Switching between the two Interrupt modes during operation is possible.

Note:	When the UTXEN bit is set, the UxTXIF flag bit is also set if UTXISEL0<1:0> = `00` (since the transmit buffer is not yet full, i.e.,transmit data can move to the UxTXREG register).

While the UxTXIF flag bit indicates the status of the UxTXREG register, the TRMT bit (UxSTA<8>) shows the status of the UxTSR register. The TRMT Status bit is a read-only bit, which is set when the UxTSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit to determine if the UxTSR register is empty.

21.5.3 Setup for UART Transmit

Use the following steps to set up a UART transmission:

1. Initialize the UxBRG register for the appropriate baud rate (refer to **Section 21.3 “UART Baud Rate Generator”**).
2. Set the number of data bits, number of Stop bits, and parity selection by writing to the PDSEL<1:0> (UxMODE<2:1>) and STSEL (UxMODE<0>) bits.
3. If transmit interrupts are desired, set the UxTXIE control bit in the corresponding Interrupt Enable Control register (IEC). Specify the interrupt priority and subpriority for the transmit interrupt using the UxIP<2:0> and UxIS<1:0> control bits in the corresponding Interrupt Priority Control register (IPC). Also, select the Transmit Interrupt mode by writing the UTXISEL0 (UxSTA<15:14>) bits.
4. Enable the UART module by setting the ON (UxMODE<15>) bit.
5. Enable the transmission by setting the UTXEN (UxSTA<10>) bit, which also sets the UxTXIF bit. The UxTXIF bit should be cleared in the software routine that services the UART transmit interrupt. The operation of the UxTXIF bit is controlled by the UTXISEL0 control bits.
6. Load data to the UxTXREG register (starts transmission). If 9-bit transmission is selected, load a word. If 8-bit transmission is used, load a byte. Data can be loaded into the buffer until the TXBF Status bit (UxSTA<9>) is set.

Note:	The UTXEN bit should not be set until the ON bit has been set. Otherwise, UART transmissions will not be enabled.

Figure 21-4: Transmission (8-Bit or 9-Bit Data)

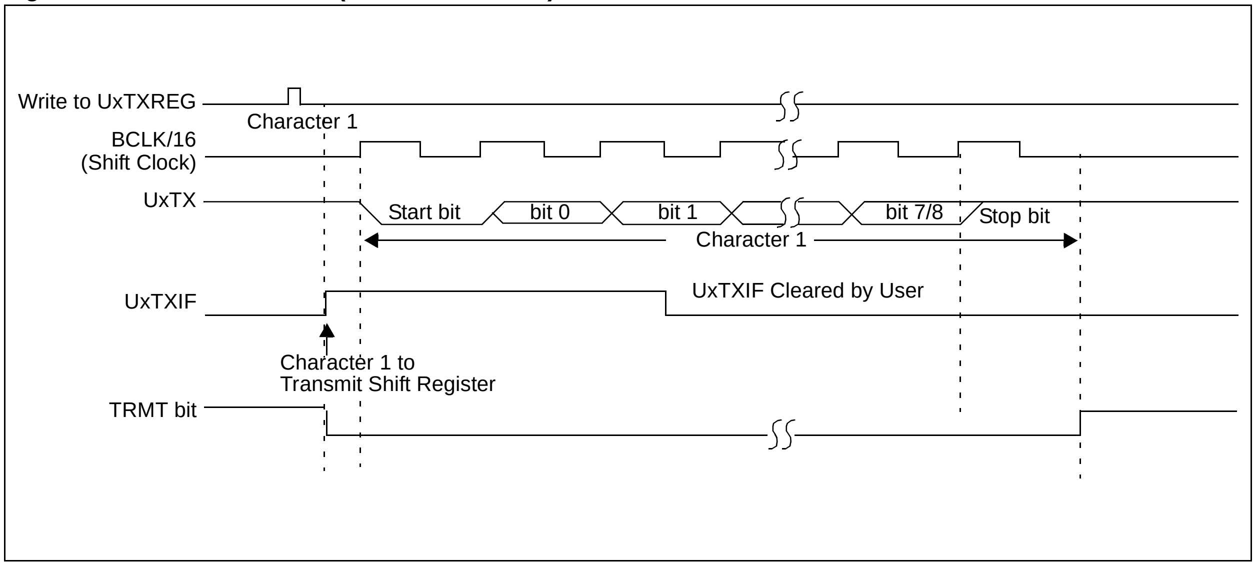
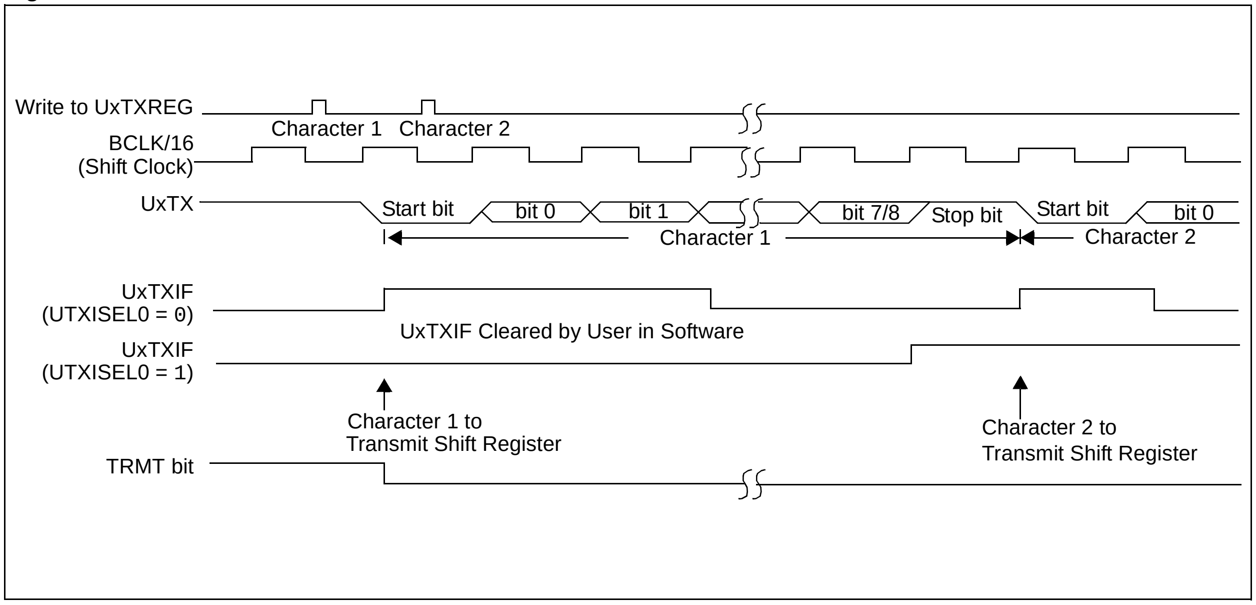


Figure 21-5: Two Consecutive Transmissions



21.5.4 Transmission of Break Characters

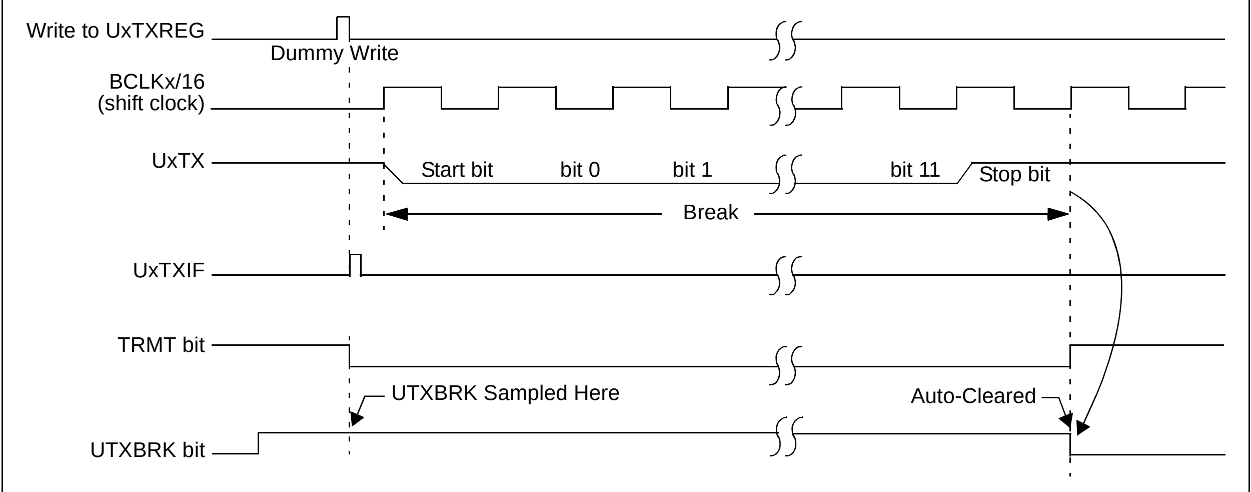
A Break character transmit consists of a Start bit, followed by twelve bits of '`0`' and a Stop bit. A Frame Break character is sent whenever the UTXBRK and UTXEN bits are set while the Transmit Shift register is loaded with data. A dummy write to the UxTXREG register is necessary to initiate the Break character transmission. Note that the data value written to the UxTXREG for the Break character is ignored. The write merely initiates the proper sequence, so that all zeroes are transmitted.

The UTXBRK bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

Note: The user should wait for the transmitter to be IDLE (TRMT = `1`) before setting the UTXBRK. The UTXBRK overrides any other transmitter activity. If the user clears the UTXBRK bit prior to sequence completion, unexpected module behavior can result. Sending a Break character does not generate a transmit interrupt.

The TRMT bit indicates whether the Transmit Shift register is empty or full, just as it does during normal transmission. See Figure 21-6 for the timing of the Break character sequence.

Figure 21-6: Send Break Character Sequence



21.5.5 Break and Sync Transmit Sequence

The following sequence is performed to send a message frame header that is composed of a Break character, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the UART for the desired mode, refer to **Section 21.5.3 "Setup for UART Transmit"**for set up information.
2. Set UTXEN and UTXBRK to set up the Break character.
3. Load the UxTXREG with a dummy character to initiate transmission (value is ignored).
4. Write '`0x55`' to UxTXREG to load the Sync character into the transmit FIFO.

After the Break is sent, the UTXBRK bit is reset by hardware. The Sync character now transmits.

21.6 UART RECEIVER

The heart of the receiver is the Receive (Serial) Shift register (UxRSR). The data is received on the UxRX pin and is sent to the data recovery block. The data recovery block operates at 16 times the baud rate, whereas the main receive serial shifter operates at the baud rate. After sampling the UxRX pin for the Stop bit, the received data in UxRSR is transferred to the receive FIFO, if it is empty. See Figure 21-7 for a UART receiver block diagram.

Note: The UxRSR register is not mapped in memory, so it is not available to the user.

Reception is enabled by setting the URXEN bit (UxSTA<12>). The data on the UxRX pin is sampled three times by a majority detect circuit to determine whether a high or a low level is present at the UxRX pin.

21.6.0.1 Receive Buffer (UxRXREG)

The UART receiver has a 4-level deep, 9-bit wide FIFO receive data buffer. UxRXREG is a memory mapped register that provides access to the output of the FIFO. It is possible for four words of data to be received and transferred to the FIFO and a fifth word to begin shifting to the UxRSR register before a buffer overrun occurs.

21.6.0.2 Receiver Error Handling

If the FIFO is full (contains four characters) and a fifth character is fully received into the UxRSR register, the overrun error bit OERR (UxSTA<1>) is set. The word in UxRSR will be kept, but further transfers to the receive FIFO are inhibited as long as the OERR bit is set. The user must clear the OERR bit in software to allow further data to be received.

To keep the data received prior to the overrun, the user should first read all five characters, then clear the OERR bit. If the five characters can be discarded, the user can simply clear the OERR bit. This effectively resets the receive FIFO, and all prior received data is lost.

Note: The data in the receive FIFO should be read prior to clearing the OERR bit. The FIFO is reset when OERR is cleared, which causes all data in the buffer to be lost.

The framing error bit FERR (UxSTA<2>) is set when a Stop bit is detected as a logic low level.

The parity error bit PERR (UxSTA<3>) is set if a parity error has been detected in the data word at the top of the buffer (i.e., the current word). For example, a parity error occurs if the parity is set as even, but the total number of ones in the data has been detected as odd. The PERR bit is irrelevant in the 9-bit mode. The FERR and PERR bits are buffered along with the corresponding word and should be read before reading the data word.

21.6.0.3 Receive Interrupt

The UART receive interrupt flag (UxRXIF) is located in the corresponding Interrupt Flag Status (IFSx) register. The RXISEL<1:0> (UxSTA<7:6>) control bits determine when the UART receiver generates an interrupt.

1. If RXISEL<1:0> = `00` or `01`, an interrupt is generated each time a data word is transferred from the Receive Shift register (UxRSR) to the receive buffer. There may be one or more characters in the receive buffer.
2. If RXISEL<1:0> = `10`, an interrupt is generated when a word is transferred from the Receive Shift register (UxRSR) to the receive buffer and as a result, the receive buffer contains three or four characters.
3. If RXISEL<1:0> = `11`, an interrupt is generated when a word is transferred from the Receive Shift register (UxRSR) to the receive buffer and as a result, the receive buffer contains four characters, i.e., becomes full.

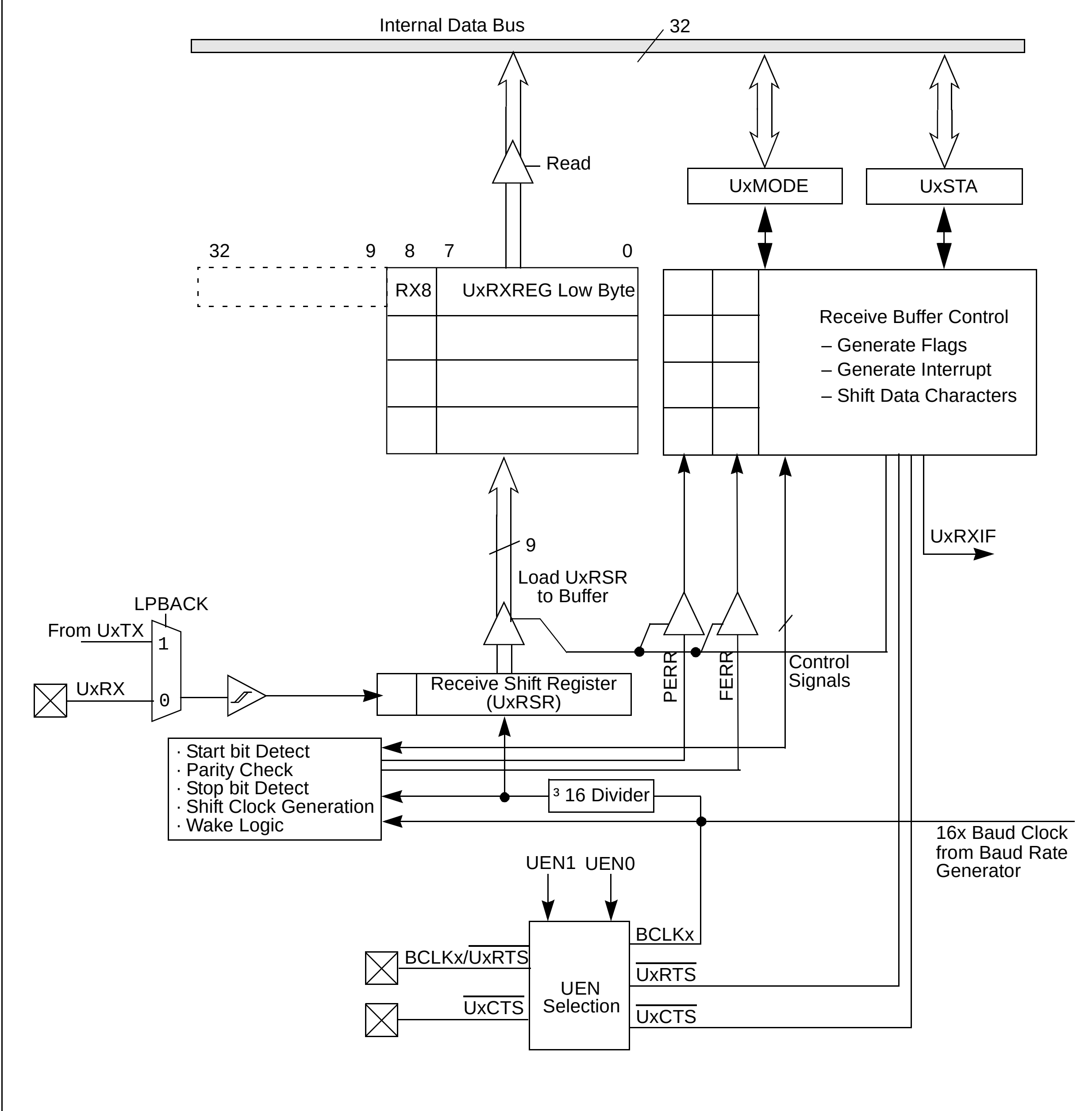
Switching between the three Interrupt modes during operation is possible.

While the RXDA and UxRXIF flag bits indicate the status of the UxRXREG register, the RIDLE bit (UxSTA<4>) shows the status of the UxRSR register. The RIDLE Status bit is a read-only bit that is set when the receiver is IDLE, i.e., the UxRSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit to determine whether the UxRSR is IDLE.

The RXDA bit (UxSTA<0>) indicates whether the receive buffer has data or is empty. This bit is set as long as there is at least one character to be read from the receive buffer. RXDA is a read-only bit.

A block diagram of the UART receiver is shown in Figure 21-7.

Figure 21-7: UART Receiver Block Diagram



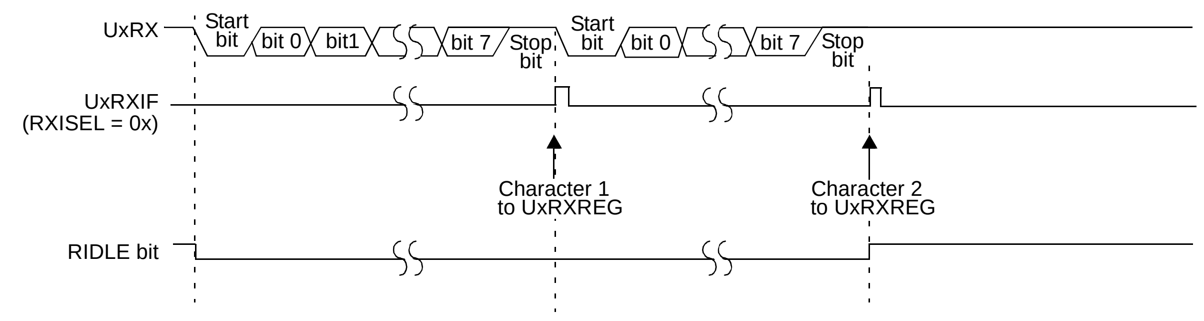
Note: 'x' denotes the UART number.

21.6.0.4 Setup for UART Reception

The following steps are performed to set up a UART reception:

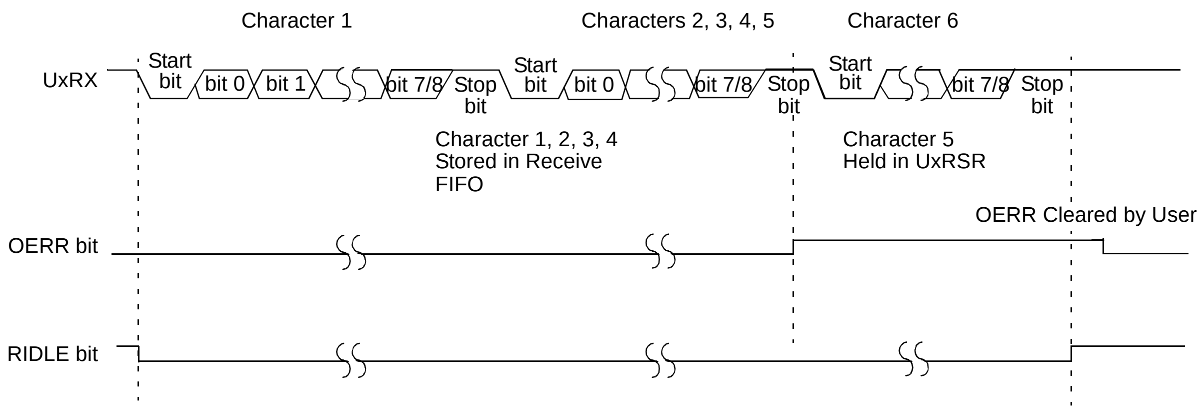
1. Initialize the UxBRG register for the appropriate baud rate (see **Section 21.3 “UART Baud Rate Generator”**).
2. Set the number of data bits, number of Stop bits and parity selection by writing to the PDSEL<1:0> (UxMODE<2:1>) and STSEL (UxMODE<0>) bits.
3. If interrupts are desired, set the UxRXIE bit in the corresponding Interrupt Enable Control (IEC) register. Specify the interrupt priority and subpriority for the interrupt using the UxIP<2:0> and UxIS<1:0> control bits in the corresponding Interrupt Priority Control (IPC) register. Also, select the Receive Interrupt mode by writing to the RXISEL<1:0> (UxSTA<7:6>) bits.
4. Enable the UART receiver by setting the URXEN (UxSTA<12>) bit.
5. Enable the UART module by setting the ON (UxMODE<15>) bit.
6. Receive interrupts are dependent on the RXISEL<1:0> control bit settings. If receive interrupts are not enabled, the user can poll the RXDA bit. The UxRXIF bit should be cleared in the software routine that services the UART receive interrupt.
7. Read data from the receive buffer. If 9-bit transmission has been selected, read a word; otherwise, read a byte. The RXDA Status bit (UxSTA<0>) is set whenever data is available in the buffer.

Figure 21-8: UART Reception



Note: This timing diagram shows 2 characters received on the UxRX input.

Figure 21-9: UART Reception with Receive Overrun



Note: This diagram shows 6 characters received without the user reading the input buffer. The 5th character received is held in the Receive Shift register. An overrun error occurs at the start of the 6th character.

21.7 USING THE UART FOR 9-BIT COMMUNICATION

The UART receiver in 9-bit Data mode is used for communication in a multiprocessor environment. With the ADDEN bit set in 9-bit Data mode, a receiver can ignore the data when the 9th bit of the data is '0'.

21.7.1 Multiprocessor Communications

A typical multi-processor communication protocol differentiates between data bytes and address/control bytes. A common scheme is to use a 9th data bit to identify whether a data byte is address or data information. If the 9th bit is set, the data is processed as address or control information. If the 9th bit is cleared, the received data word is processed as data associated with the previous address/control byte.

The protocol operates in the following sequence:

- The master device transmits a data word with the 9th bit set. The data word contains the address of a slave device and is considered the address word.
- All slave devices in the communication chain receive the address word and check the slave address value.
- The slave device that is specified by the address word receives and processes subsequent data bytes sent by the master device. All other slave devices discard subsequent data bytes until a new address word is received.

21.7.1.1 ADDEN Control Bit

The UART receiver has an Address Detect mode which allows it to ignore data words with the 9th bit cleared. This reduces the interrupt overhead, since data words with the 9th bit cleared are not buffered. This feature is enabled by setting the ADDEN bit (UxSTA<5>).

The UART must be configured for 9-bit data to use the Address Detect mode. The ADDEN bit has no effect when the receiver is configured in 8-bit Data mode.

21.7.1.2 Setup for 9-Bit Transmit Mode

The setup procedure for 9-bit transmission is identical to the 8-bit Transmit modes, except that PDSEL<1:0> (UxMODE<2:1) should be set to '11.' Word writes should be performed to the UxTXREG register (starts transmission). Refer to **Section 21.5.3 "Setup for UART Transmit"** for more information on setting up for transmission.

21.7.1.3 Setup for 9-Bit Reception Using Address Detect Mode

The setup procedure for 9-bit reception is similar to the 8-bit Receive modes, except that PDSEL<1:0> (UxMODE<2:1) should be set to '11' (refer to **Section 21.6.0.4 "Setup for UART Reception"** for more information about setting up for UART reception).

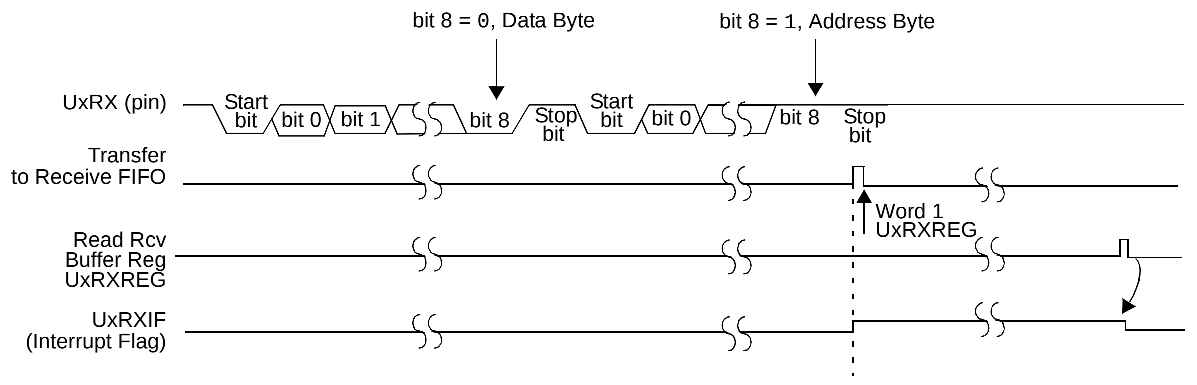
Receive Interrupt mode should be configured by writing to the RXISEL<1:0> (UxSTA<7:6>) bits.

Note: An interrupt is generated when an Address character is detected and the Address Detect mode is enabled (ADDEN = 1), regardless of how the RXISEL<1:0> control bits are set.

Perform the following steps to use the Address Detect mode:

1. Set PDSEL<1:0> (UxMODE<2:1) to '11' to choose 9-bit mode.
2. Set the ADDEN (UxSTA<5>) bit to enable address detect.
3. Set ADDR (UxSTA<23:16>) to the desired device address character.
4. Set the ADM_EN (UxSTA<24>) bit to enable Address Detect mode.
5. If this device has been addressed, the UxRXREG is discarded, all subsequent characters received that have UxRXREG<8> = 0 are transferred to the UART receive buffer, and interrupts are generated according to RXISEL<1:0>.

Figure 21-10: Reception with Address Detect (ADDEN = 1)



Note: This timing diagram shows a data byte followed by an address byte. The data byte is not read into the UxRXREG (receive buffer) because ADDEN = 1 and bit 8 = 0.

21.8 RECEIVING BREAK CHARACTERS

The wake-up feature is enabled by setting the WAKE bit (UxMODE <7>) = `1`. In this mode, the module receives the Start bit, data, and the invalid Stop bit (which sets FERR); but, the receiver waits for a valid Stop bit before looking for the next Start bit. It will not assume that the Break condition on the line is the next Start bit. Break is regarded as a character containing all zeros with the FERR bit set. The Break character is loaded into the buffer. No further reception can occur until a Stop bit is received. The WAKE bit is cleared automatically when the Stop bit is received after the 13-bit Break character. Note that RIDLE goes high when the Stop bit is received.

The receiver counts and expects a certain number of bit times based on the values programmed in the PDSEL<1:0> (UxMODE<2:1>) and STSEL (UxMODE<0>) bits.

If the Break is longer than 13 bit times, the reception is considered complete after the number of bit times specified by the PDSEL and STSEL bits elapses. The RXDA bit is set, FERR is set, zeros are loaded into the receive FIFO and interrupts are generated.

If the wake-up feature is not set, WAKE (UxMODE <7>) = `0`, Break reception is not special. The Break is counted as one character loaded into the buffer (all '`0`' bits) with FERR set.

21.9 INITIALIZATION

An initialization routine for the Transmitter/Receiver in 8-bit mode is shown in Example 21-2. An initialization of the Addressable UART in 9-bit Address Detect mode is shown in Example 21-3. In both examples, the value to load into the UxBRG register is dependent on the desired baud rate and the device frequency.

Note:	UTXEN bit should not be set until the ON bit has been set. Otherwise, UART transmissions is not enabled.

Example 21-2: 8-bit Transmit/Receive (UART1)

```
U1BRG   =   BaudRate;          //Set Baud rate

U1STA   =   0;
U1MODE  =   0x8000;            //Enable Uart for 8-bit data
                               //no parity, 1 STOP bit
U1STASET =  0x1400;            //Enable Transmit and Receive
```

Example 21-3: 8-bit Transmit/Receive (UART1), Address Detect Enabled

```
U1BRG   =   BaudRate;          //Set Baud rate

U1MODE  =   0x8006;            //Enable Uart for 9-bit data
                               //no parity,1 STOP bit
U1STA   =   0x1211420;         //Address detect enabled
                               //Device Address=0x21
                               //Enable Automatic Address Detect Mode
                               //Enable Transmit and Receive
```

21.10 OTHER FEATURES OF THE UART

21.10.1 UART in Loopback Mode

Setting the LPBACK bit enables this special mode in which the UxTX output is internally connected to the UxRX input. When configured for the Loopback mode, the UxRX pin is disconnected from the internal UART receive logic; however, the UxTX pin still functions normally.

Use the following steps to select Loopback mode:

1. Configure UART for the desired mode of operation (see Section 21.5.3).
2. Enable transmission as defined in **Section 21.5 “UART Transmitter”** in this document.
3. Set LPBACK = `1` (UxMODE<6>) to enable Loopback mode.

Table 21-3 shows how the Loopback mode is dependent on the UEN<1:0> bits.

Table 21-3: Loopback Mode Pin Function

UEN<1:0>	Pin Function, LPBACK = 1[1]
`00`	UxRX input connected to UxTX UxTX pin functions UxRX pin ignored $\overline{\text{UxCTS}}$/$\overline{\text{UxRTS}}$ unused
`01`	UxRX input connected to UxTX UxTX pin functions UxRX pin ignored $\overline{\text{UxRTS}}$ pin functions $\overline{\text{UxCTS}}$ unused
`10`	UxRX input connected to UxTX UxTX pin functions UxRX pin ignored $\overline{\text{UxRTS}}$ pin functions $\overline{\text{UxCTS}}$ input connected to $\overline{\text{UxRTS}}$ $\overline{\text{UxCTS}}$ pin ignored
`11`	UxRX input connected to UxTX UxTX pin functions UxRX pin ignored BCLKx pin functions $\overline{\text{UxCTS}}$/$\overline{\text{UxRTS}}$ unused

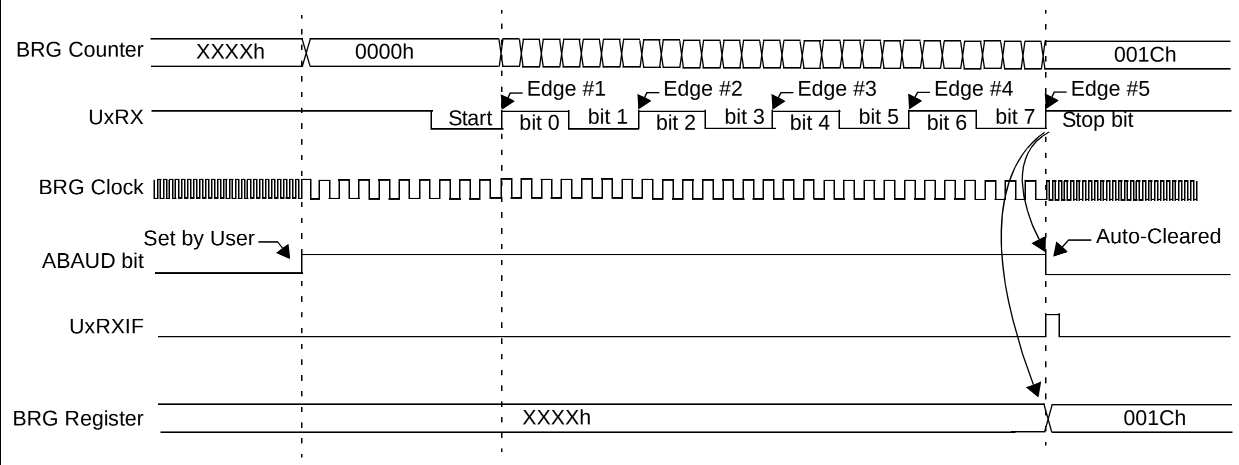
Note 1: LPBACK = `1` should be set only after enabling the other bits associated with the UART module.

21.10.2 Auto-Baud Support

To allow the system to determine the baud rates of the received characters, the ABAUD bit is enabled. The UART begins an automatic baud rate measurement sequence whenever a Start bit is received, and when the Auto-Baud Rate Detect is enabled (ABAUD = `1`). The calculation is self-averaging. This feature is active only while the auto-wake-up is disabled (WAKE = `0`). In addition, LPBACK must equal '`0`' for the auto-baud operation. When the ABAUD bit is set, the BRG counter value clears and looks for a Start bit – which, in this case, is defined as a high-to-low transition, followed by a low-to-high transition.

Following the Start bit, the auto-baud expects to receive an ASCII 'U' (55h) to calculate the proper bit rate. The measurement is taken over both the low and the high bit time to minimize any effects caused by asymmetry of the incoming signal. At the end of the Start bit (rising edge), the BRG counter begins counting up using a $F_{PB}/8$ clock. On the 5th UxRX pin rising edge, an accumulated BRG counter value totaling the proper BRG period is transferred to the UxBRG register. The ABAUD bit automatically clears. If the user clears the ABAUD bit prior to sequence completion, unexpected module behavior can result. Refer to Figure 21-1 for the ABD sequence.

Figure 21-11: Automatic Baud Rate Calculation



While the auto-baud sequence is in progress, the UART state machine is held in IDLE. The UxRXIF interrupt is set on the 5th UxRX rising edge, independent of the RXISEL<1:0> settings. The receiver FIFO is not updated.

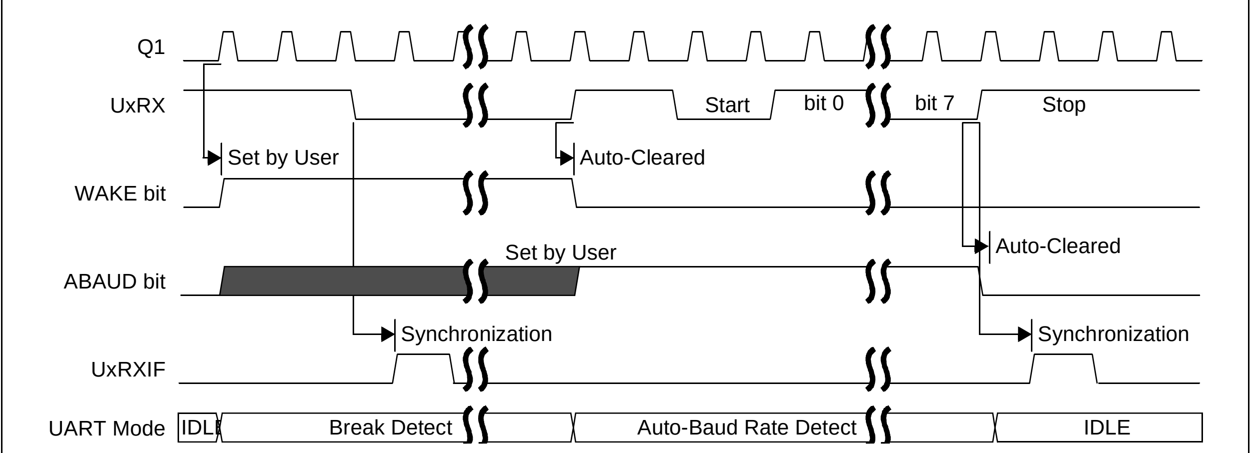
21.10.3 Break Detect Sequence

The user can configure the auto-baud to occur immediately following the Break detect. This is done by setting the ABAUD bit with the WAKE bit set. Figure 21-12 shows a Break detect followed by an auto-baud sequence. The WAKE bit takes priority over the ABAUD bit setting.

Note: If the WAKE bit is set with the ABAUD bit, auto-baud rate detection occurs on the byte following the Break character. The user must ensure that the incoming character baud rate is within the range of the selected UxBRG clock source, considering the baud rate possible with the given clock.

The UART transmitter cannot be used during an auto-baud sequence. Furthermore, the user should ensures that the ABAUD bit is not set while a transmit sequence is already in progress. Otherwise, the UART may exhibit unpredictable behavior.

Figure 21-12: Break Detect Followed by Auto-Baud Sequence



21.11 OPERATION OF $\overline{\text{UxCTS}}$ AND $\overline{\text{UxRTS}}$ CONTROL PINS

$\overline{\text{UxCTS}}$ (Clear to Send) and $\overline{\text{UxRTS}}$ (Request to Send) are the two hardware controlled pins associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control modes, which are explained in detail in **Section 21.11.2** and **Section 21.11.3**, respectively. They are implemented to control the transmission and reception among the Data Terminal Equipment (DTE).

21.11.1 $\overline{\text{UxCTS}}$ Function

In the UART operation, the $\overline{\text{UxCTS}}$ acts as an input pin that can control the transmission. This pin is controlled by another device (typically a PC). The $\overline{\text{UxCTS}}$ pin is configured using UEN<1:0>. When UEN<1:0> = `10`, $\overline{\text{UxCTS}}$ is configured as an input. If $\overline{\text{UxCTS}}$ = `1`, then the transmitter will go as far as loading the data in the Transmit Shift register, but will not initiate a transmission. This allows the DTE to control and receive the data accordingly from the controller, per its requirement.

The $\overline{\text{UxCTS}}$ pin is sampled simultaneously with a transmit data change (i.e., at the beginning of the 16 baud clocks). Transmission begins only when the $\overline{\text{UxCTS}}$ is sampled low. The $\overline{\text{UxCTS}}$ is sampled internally with a Q clock, which means that there is a minimum pulse width on $\overline{\text{CTS}}$ of one peripheral clock. However, this cannot be a specification, as the F_{PB} can vary, depending on the clock used.

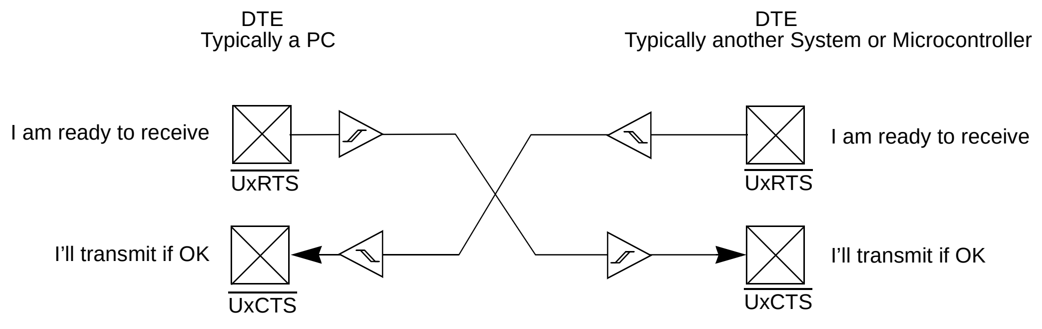
The user can also read the status of the $\overline{\text{UxCTS}}$ by reading the associated port pin.

21.11.2 $\overline{\text{UxRTS}}$ Function in Flow Control Mode

In the Flow Control mode, the $\overline{\text{UxRTS}}$ of one DTE is connected to the $\overline{\text{UxCTS}}$ of the PIC32MX and the $\overline{\text{UxCTS}}$ of the DTE is connected to the $\overline{\text{UxRTS}}$ of the PIC32MX, as shown in Figure 21-13. The $\overline{\text{UxRTS}}$ signal indicates that the device is ready to receive the data. The $\overline{\text{UxRTS}}$ pin is driven as an output whenever UEN<1:0> = `01` or `10`. The $\overline{\text{UxRTS}}$ pin is asserted (driven low) whenever the receiver is ready to receive data. When the RTSMD bit = `0` (when the device is in Flow Control mode), the $\overline{\text{UxRTS}}$ pin is driven low whenever the receive buffer is not full or the OERR bit is not set. When the RTSMD bit = `0`, the $\overline{\text{UxRTS}}$ pin is driven high whenever the device is not ready to receive (i.e., when the receiver buffer is either full or in the process of shifting).

Since the $\overline{\text{UxRTS}}$ of the DTE is connected to the $\overline{\text{UxCTS}}$ of the PIC32MX, the $\overline{\text{UxRTS}}$ drives the $\overline{\text{UxCTS}}$ low whenever it is ready to receive the data. Transmission of the data begins when the $\overline{\text{UxCTS}}$ goes low, as explained in **Section 21.11.1**.

Figure 21-13: $\overline{\text{UxRTS}}$/$\overline{\text{UxCTS}}$ Flow Control for DTE-DTE (RTSMD = `0`, Flow Control Mode)



21.11.3 $\overline{\text{UxRTS}}$ Function in Simplex Mode

In the Simplex mode, the $\overline{\text{UxRTS}}$ of the DCE is connected to the $\overline{\text{UxRTS}}$ of the PIC32MX and the $\overline{\text{UxCTS}}$ of the DCE is connected to the $\overline{\text{UxCTS}}$ of the PIC32MX, respectively, as shown in Figure 21-14. In the Simplex mode, the $\overline{\text{UxRTS}}$ signal indicates that the DTE is ready to transmit. The DCE replies to the $\overline{\text{UxRTS}}$ signal with the valid $\overline{\text{UxCTS}}$ when the DCE is ready to receive the transmission. When the DTE receives a valid $\overline{\text{UxCTS}}$, it begins transmission.

Figure 21-15 shows the Simplex mode is also used in IEEE-485 systems to enable transmitters. When $\overline{\text{UxRTS}}$ indicates that the DTE is ready to transmit, the $\overline{\text{UxRTS}}$ signal enables the driver.

The $\overline{\text{UxRTS}}$ pin is configured as an output and is driven whenever UEN<1:0> = `01` or `10`. When RTSMD = `1`, the $\overline{\text{UxRTS}}$ is asserted (driven low) whenever the data is available to transmit (TRMT = `0`). When RTSMD = `1`, $\overline{\text{UxRTS}}$ is deasserted (driven high) when the transmitter is empty (TRMT = `1`).

Figure 21-14: $\overline{\text{UxRTS}}/\overline{\text{UxCTS}}$ Handshake for DTE-DCE (RTSMD = `1`**, Simplex Mode)**

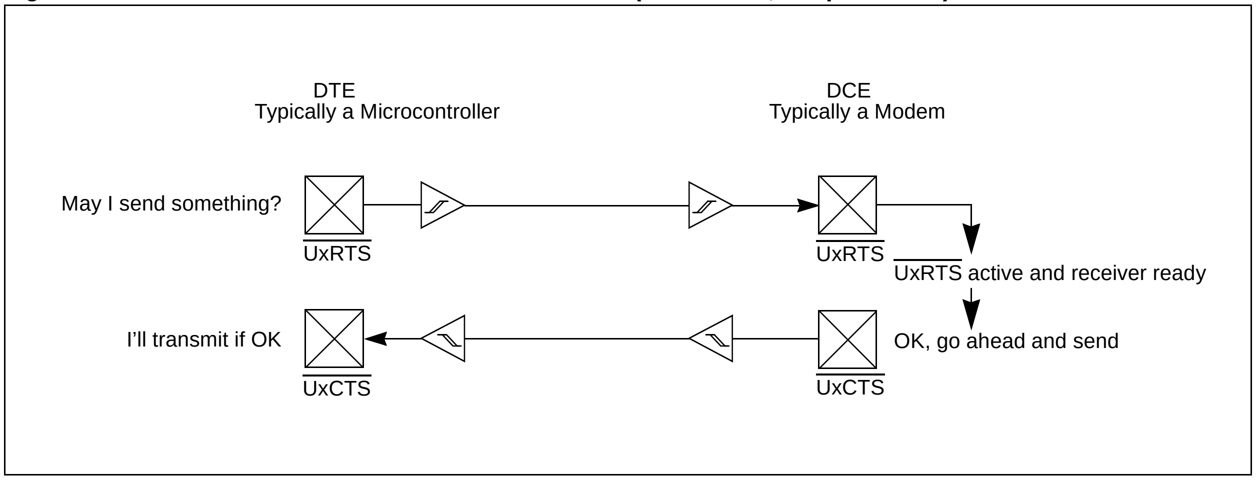
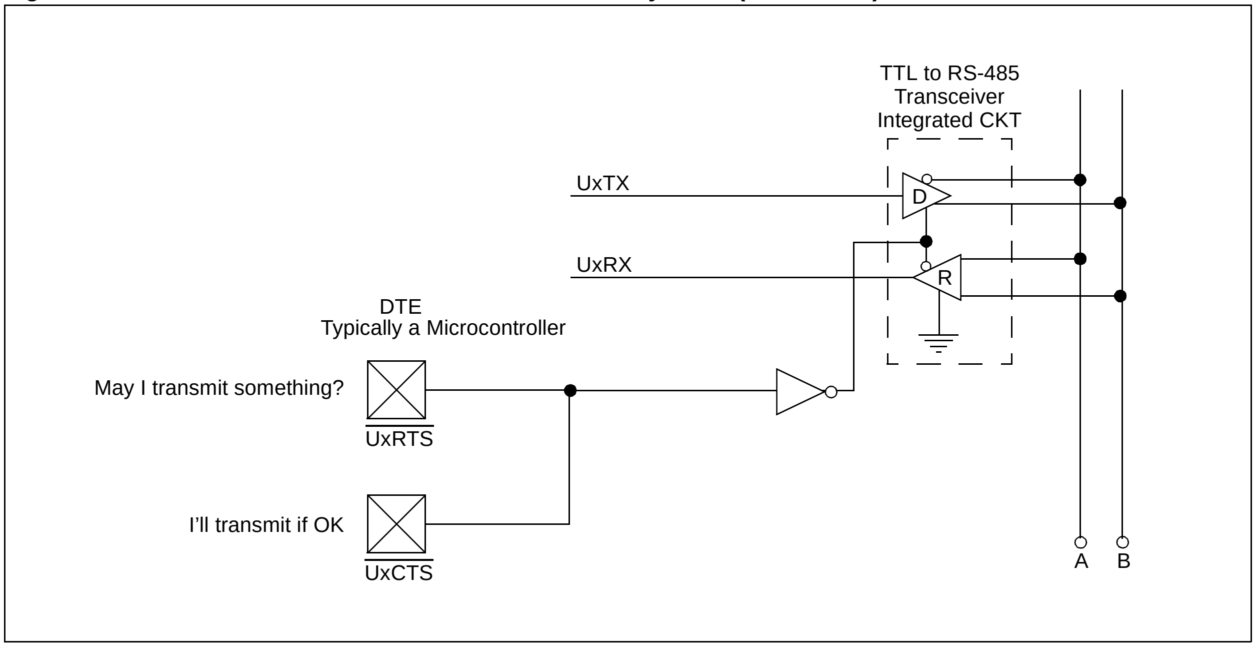


Figure 21-15: $\overline{\text{UxRTS}}/\overline{\text{UxCTS}}$ Bus Enable for IEEE-485 Systems (RTSMD = `1`**)**



21.12 INFRARED SUPPORT

The UART module provides the following two types of infrared UART support:

- IrDA clock output to support external IrDA encoder and decoder devices (legacy module support)
- Full implementation of the IrDA encoder and decoder

21.12.1 External IrDA Support – IrDA Clock Output

To support external IrDA encoder and decoder devices, the BCLKx pin can be configured to generate the 16x baud clock. When UEN<1:0> = `11`, the BCLKx pin will output the 16x baud clock if the UART module is enabled; it can be used to support the IrDA codec chip.

21.12.2 Built-In IrDA Encoder and Decoder

The UART has full implementation of the IrDA encoder and decoder as part of the UART module. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit (UxMODE<12>). When enabled (IREN = `1`), the receive pin UxRX acts as the input from the infrared receiver. The transmit pin UxTX acts as the output to the infrared transmitter.

21.12.2.1 IrDA Encoder Function

The encoder works by taking the serial data from the UART and replacing it in the following manner:

- Transmit bit data of ‘`1`’ gets encoded as ‘`0`’ for the entire 16 periods of the 16x baud clock.
- Transmit bit data of ‘`0`’ gets encoded as ‘`0`’ for the first 7 periods of the 16x baud clock, as ‘`1`’ for the next 3 periods and as ‘`0`’ for the remaining 6 periods.

See Figure 21-16 and Figure 21-18 for details.

21.12.2.2 IrDA Transmit Polarity

The IrDA transmit polarity is selected using the UTXINV bit (UxSTA<13>). This bit only affects the module when the IrDA encoder and decoder are enabled (IREN = `1`). The UTXINV bit does not affect the receiver or the module operation for normal transmission and reception. When UTXINV = `0`, the IDLE state of the UxTX line is ‘`0`’ (see Figure 21-16). When UTXINV = `1`, the IDLE state of the UxTX line is ‘`1`’ (see Figure 21-17).

Figure 21-16: IrDA® Encode Scheme

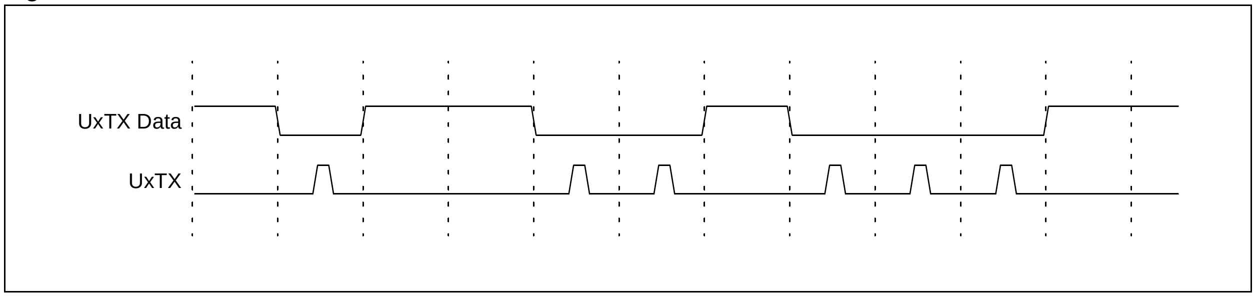


Figure 21-17: IrDA® Encode Scheme for ‘`0`’ Bit Data

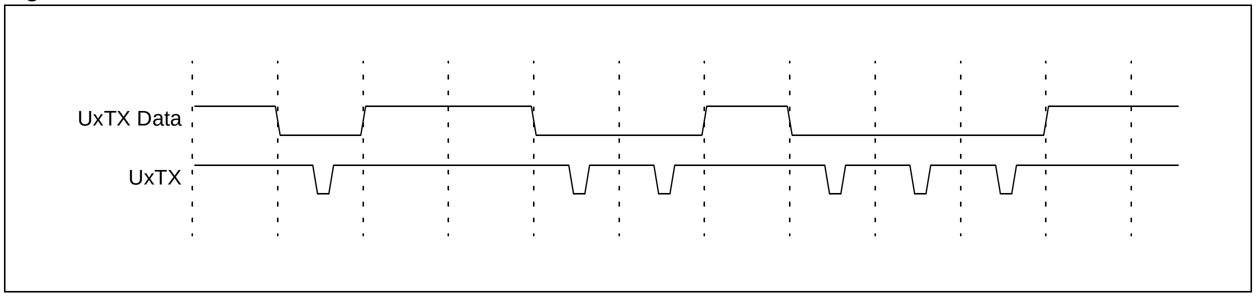
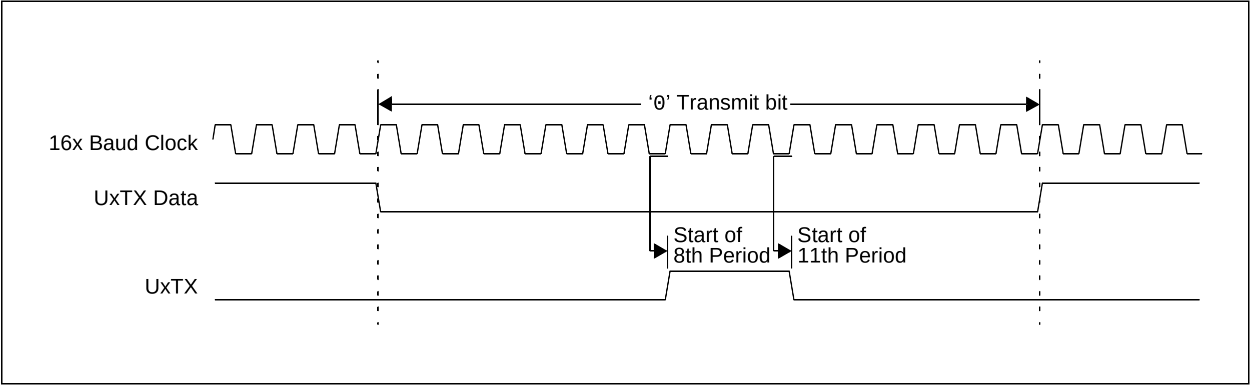


Figure 21-18: IrDA® Encode Scheme for ‘0’ Bit Data with Respect to 16x Baud Clock



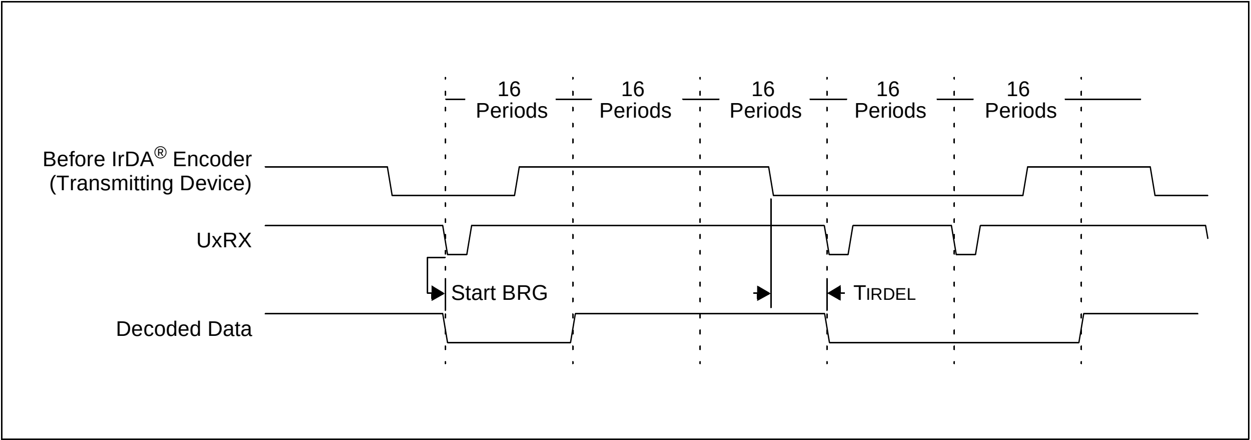
21.12.2.3 IrDA Decoder Function

The decoder works by taking the serial data from the UxRX pin and replacing it with the decoded data stream. The stream is decoded based on falling edge detection of the UxRX input.

Each falling edge of UxRX causes the decoded data to be driven low for 16 periods of the 16x baud clock. If, by the time the 16 periods expire, another falling edge is detected, the decoded data remains low for another 16 periods. If no falling edge is detected, the decoded data is driven high.

Note that the data stream into the device is shifted anywhere from 7 to 8 periods of the 16x baud clock from the actual message source. The one clock uncertainty is due to the clock edge resolution (see Figure 21-19 for details).

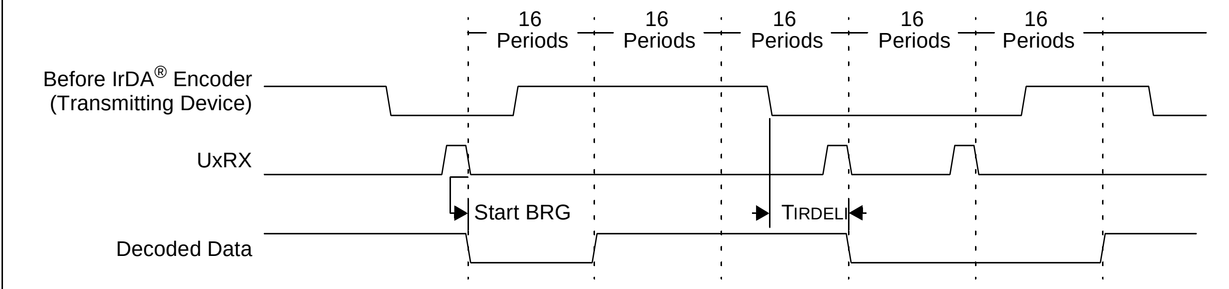
Figure 21-19: Macro View of IrDA® Decoding Scheme



21.12.2.4 IrDA Receive Polarity

The input of the IrDA signal can have an inverted polarity. The same logic is able to decode the signal train, but in this case, the decoded data stream is shifted from 10 to 11 periods of the 16x baud clock from the original message source. Again, the one clock uncertainty is due to the clock edge resolution (see Figure 21-20 for details).

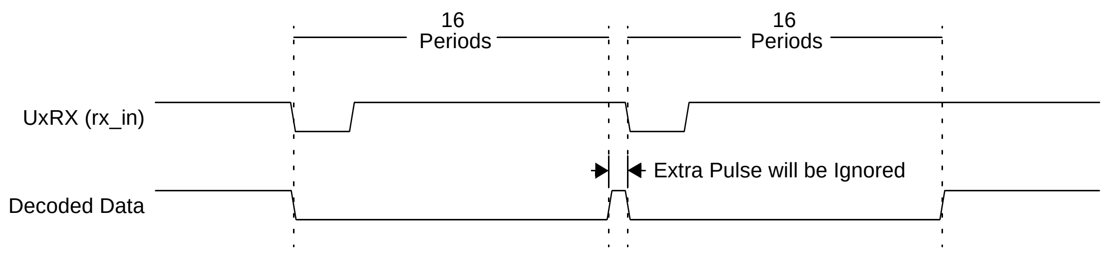
Figure 21-20: Inverted Polarity Decoding Results



21.12.2.5 Clock Jitter

Due to jitter, or slight frequency differences between devices, it is possible for the next falling bit edge to be missed for one of the 16x periods. In that case, a one clock-wide-pulse appears on the decoded data stream. Since the UART performs a majority detect around the bit center, this does not cause erroneous data (see Figure 21-21 for details).

Figure 21-21: Clock Jitter Causing a Pulse Between Consecutive Zeros



21.13 INTERRUPTS

The UART has the ability to generate interrupts reflecting the events that occur during the data communication. The following types of interrupts can be generated:

- Receiver-data-available interrupt, signalled by bits U1RXIF (IFS0<27>) and U2RXIF (IFS1<9>). This event occurs based on the value of RXISEL<1:0> (UxSTA<7:6>) control bits. Refer to **Section 21.6.0.3** for details.
- Transmitter buffer-empty interrupt, signalled by bits U1TXIF (IFS0<28>) and U2TXIF (IFS1<10>). This event occurs based on the value of control bits UTXISEL0<1:0> (UxSTA<15:14>). Refer to **Section 21.5.2** for details.
- UART-error interrupt, signalled by bits U1EIF (IFS0<26>) and U2EIF (IFS1<8>).
  - This event occurs when any of the following error conditions occur:
  - Parity error PERR (UxSTA<3>) is detected
  - Framing Error FERR (UxSTA<2>) is detected
  - Overflow condition for the receive buffer OERR (UxSTA<1>) occurs

All these interrupt flags must be cleared in software.

A UART device is enabled as a source of interrupts via the following respective UART interrupt enable bits:

- U1RXIE (IEC0<27>) and U2RXIE (IEC1<9>)
- U1TXIE (IEC0<28>) and U2TXIE (IEC1<10>)
- U1EIE (IEC0<26>) and U2EIE (IEC1<8>)

The interrupt priority-level bits and interrupt subpriority-level bits must be also be configured:

- U1IP (IPC6<4:2>) and U1IS (IPC6<1:0>)
- U2IP (IPC8<4:2>) and U2IS (IPC8<1:0>)

Refer to **Section 2. “Interrupts”** in this manual for details about priority and subpriority bits.

21.13.1 Interrupt Configuration

Each UART module has the following dedicated interrupt flag bits:

- UxEIF
- UxRXIF
- UxTXIF

Each UART module also has the following corresponding interrupt enable/mask bits:

- UxEIE
- UxRXIE
- UxTXIE

These bits determine the source of an interrupt and enable or disable an individual interrupt source. Note that all the interrupt sources for a specific UART module share just one interrupt vector. Each UART module can have its own priority level, independent of other UART modules.

Note that the UxTXIF, UxRXIF and UxEIF bits will be set without regard to the state of the corresponding enable bits. The IF bits can be polled by software if desired.

The UxEIE, UxTXIE, UxRXIE bits define the behavior of the Vector Interrupt Controller (VIC) when a corresponding UxEIF, UxTXIF or UxRXIF bit is set. When the corresponding IE bit is clear the VIC module does not generate a CPU interrupt for the event. If the IE bit is set, the VIC module will generate an interrupt to the CPU when the corresponding IF bit is set (subject to the priority and subpriority as outlined in the following paragraphs).

It is the responsibility of the user’s software routine that services a particular interrupt to clear the appropriate Interrupt Flag bit before the service routine is complete.

The priority of each UART module can be set independently with the UxIP<2:0> bits. This priority defines the priority group to which the interrupt source is assigned. The priority groups range from a value of 7 (the highest priority), to a value of 0, which does not generate an interrupt. An interrupt being serviced is preempted by an interrupt in a higher priority group.

The subpriority bits allow setting the priority of a interrupt source within a priority group. The values of the subpriority, UxIS<1:0>, range from 3 (the highest priority), to 0 the lowest priority. An interrupt with the same priority group but having a higher subpriority value, does not preempt a lower subpriority interrupt that is in progress.

The priority group and subpriority bits allow more than one interrupt source to share the same priority and subpriority. If simultaneous interrupts occur in this configuration the natural order of the interrupt sources within a priority/subpriority-group pair determine the interrupt generated. The natural priority is based on the vector numbers of the interrupt sources. The lower the vector number, the higher the natural priority of the interrupt. Any interrupts that are overridden by natural order generate their respective interrupts based on priority, subpriority, and natural order, after the interrupt flag for the current interrupt is cleared.

After an enabled interrupt is generated, the CPU jumps to the vector assigned to that interrupt. The vector number for the interrupt is the same as the natural order number. Then the CPU begins executing code at the vector address. The user's code at this vector address should perform any application specific operations, clear the UxEIF, UxTXIF or UxRXIF interrupt flag, and then exit. Refer to **Section 2. "Interrupts"** in this manual for the vector address table details and more information on interrupts.

Table 21-4: UART Interrupt Vectors for Various Offsets with EBASE = 0x8000:0000

Interrupt	Vector/ Natural Order	IRQ Number	Vector Address IntCtl.VS = 0x01	Vector Address IntCtl.VS = 0x02	Vector Address IntCtl.VS = 0x04	Vector Address IntCtl.VS = 0x08	Vector Address IntCtl.VS = 0x10
U1E	24	26	8000 0500	8000 0800	8000 0E00	8000 1A00	8000 3200
U1TX	24	28	8000 0500	8000 0800	8000 0E00	8000 1A00	8000 3200
U1RX	24	27	8000 0500	8000 0800	8000 0E00	8000 1A00	8000 3200
U2E	32	40	8000 0600	8000 0A00	8000 1200	8000 2200	8000 4200
U2TX	32	42	8000 0600	8000 0A00	8000 1200	8000 2200	8000 4200
U2RX	32	41	8000 0600	8000 0A00	8000 1200	8000 2200	8000 4200

21.14 I/O PIN CONTROL

When enabling the UART module ON (UxMODE<15>), the UART module will control the I/O pins as defined by the UEN<1:0> (UxMODE<9:8>) bits, overriding the port TRIS and LATCH register bit settings.

UxTX is forced as an output and UxRX as an input. Additionally, if $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ are enabled, the $\overline{\text{UxCTS}}$ is forced as an input and the $\overline{\text{UxRTS}}$/BLCK pin functions as $\overline{\text{UxRTS}}$ output. If BLCK is enabled, then the $\overline{\text{UxRTS}}$/BLCK output drives the 16x baud clock output.

Table 21-5 provides a summary of UART modes and the specific I/O pins required for each mode.

Table 21-5: Required I/O Pin Resources

UxMODE<9:8> Setting		**Device Pins**			
UEN<1>	UEN<0>	UxTX	UxRX	$\overline{\text{UxCTS}}$	$\overline{\text{UxRTS}}$/BCLK
0	0	Yes	Yes	No	No
0	1	Yes	Yes	No	Yes
1	0	Yes	Yes	Yes	Yes
1	1	Yes	Yes	No	Yes

Note: "No" indicates that the pin is not required and can be used as a general purpose I/O pin.

21.15 UART OPERATION IN POWER-SAVING AND DEBUG MODES

Note:	In this manual, a distinction is made between a power mode as it is used in a specific module, and a power mode as it is used by the device, e.g., Sleep mode of the Comparator and SLEEP mode of the CPU. To indicate which type of power mode is intended, uppercase and lowercase letters (Sleep, Idle, Debug) signify a module power mode, and all uppercase letters (SLEEP, IDLE, DEBUG) signify a device power mode.

21.15.1 Operation in SLEEP Mode

When the device enters SLEEP mode, the system clock is disabled. The UART does not function in SLEEP mode. If entry into SLEEP mode occurs while a transmission is in progress, then the transmission is aborted and the UxTX pin is driven to logic '1'. Similarly, if entry into SLEEP mode occurs while a reception is in progress, then the reception is aborted. RTS and BCLK pins are driven to '0'.

The UART can be used optionally to wake the PIC32MX device from SLEEP mode on the detection of a Start bit. If the WAKE bit UxMODE<7> is set before device enters SLEEP mode and the UART receive interrupt is enabled (UxRXIE = 1), then a falling edge on the UxRX pin generates a receive interrupt and device wakes up. The Receive Interrupt Select mode bit (RXISEL) has no effect on this function. The ON bit must be set to generate a wake-up interrupt.

21.15.2 Operation in SLEEP Mode

When the device enters SLEEP mode, the system clock sources remain functional and the CPU stops executing code. The SIDL bit (UxMODE<13>) selects whether the UART module stops operation or continues normal operation when the device enters SLEEP mode.

- If SIDL = 1, the module stops operation in SLEEP mode. The module performs the same procedures when stopped in SLEEP mode (SIDL = 1) as it does for SLEEP mode.
- If SIDL = 0, the module continues operation in SLEEP mode.

21.15.3 Operation in DEBUG Mode

The FRZ bit (UxMODE<14>) determines whether the UART module runs or stops while the CPU is executing DEBUG Exception code (i.e., the application is halted) in DEBUG mode.

Specifically, The FRZ bit affects operation in the following manner:

- If FRZ = 1, the module freezes its operations and make no changes to the state of the UART module when the application is halted in DEBUG mode. The module resumes its operation after the application resumes execution.
- If FRZ = 0, the module continues to run even when application is halted in DEBUG mode.

Note:	The FRZ bit is readable and writable only when the CPU is executing in Debug Exception mode. In all other modes, the FRZ bit reads as '0'. If FRZ bit is changed during DEBUG mode, the new value does not take effect until the current Debug Exception mode is exited and re-entered. During the Debug Exception mode, the FRZ bit reads the state of the peripheral when entering DEBUG mode.

21.15.4 Auto-Wake-up on Sync Break Character

The auto-wake-up feature is enabled using the WAKE bit (UxMODE<7>). When WAKE is active, the typical receive sequence on UxRX is disabled. Following the wake-up event, the module generates the UxRXIF interrupt.

Note that the LPBACK bit (UxMODE<6>) must equal '0' for wake-up to operate.

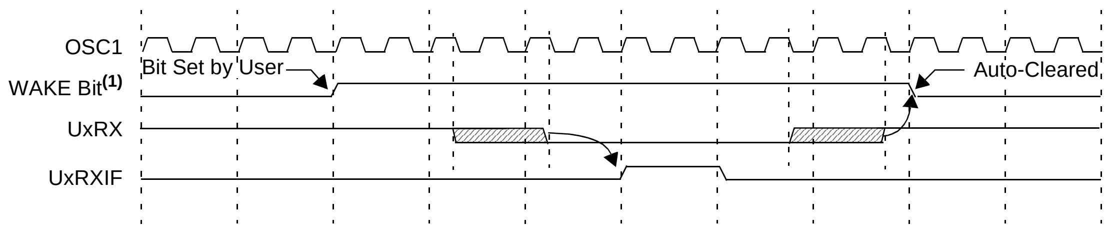
A wake-up event consists of a high-to-low transition on the UxRX line. This coincides with the start of a Sync Break or a Wake-up Signal character for the LIN protocol. When WAKE is active, the UxRX line is monitored independently from the CPU mode. The UxRXIF interrupt is generated synchronously to the Q clocks in Normal User mode; and asynchronously, if the module is disabled due to SLEEP or SLEEP mode. To ensure that no actual data is lost, the WAKE bit should be set just prior to entering the SLEEP mode and while the UART module is in IDLE mode.

The WAKE bit is automatically cleared once a low-to-high transition is observed on the UxRX line following the wake-up event. At this point, the UART module is in IDLE mode and is returned to normal operation. This signals to the user that the Sync Break event is over. If the user clears the WAKE bit prior to sequence completion, unexpected module behavior may result.

The wake-up event causes a receive interrupt by setting the UxRXIF bit. The Receive Interrupt Select mode bits RXISEL<1:0> (UxSTA<7:6>) are ignored for this function. If the UxRXIF interrupt is enabled, it wakes up the device.

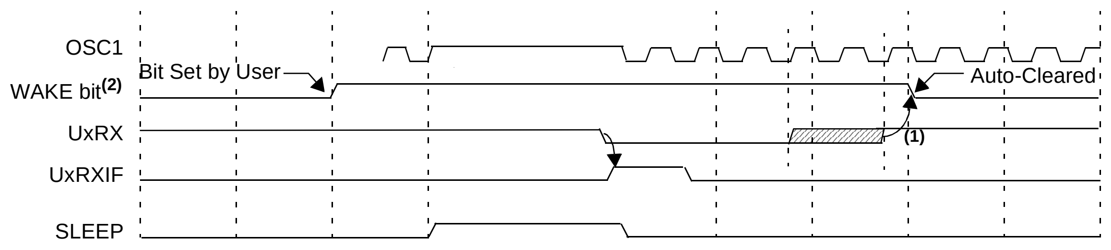
Note: The Sync Break (or Wake-up Signal) character must be of sufficient length to allow time for the selected oscillator to start and provide proper initialization of the UART. To ensure that the part woke up in time, the user should read the value of the WAKE bit. If it is clear, it is possible that the UART was not ready in time to receive the next character and the module might need to be resynchronized to the bus.

Figure 21-22: Auto-Wake-up Bit (WAKE) Timings During Normal Operation



Note 1: UART state machine is held in IDLE while WAKE bit is active.

Figure 21-23: Auto-Wake-up Bit (WAKE) Timings During SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the auto-clear of the WAKE bit can occur while the system clocks are still active. This sequence should not depend on the presence of Q clocks.

2: UART state machine is held in IDLE while WAKE bit is active.

21.16 EFFECTS OF VARIOUS RESETS

21.16.1 Device Reset

All UART registers are forced to their Reset states upon a device Reset.

21.16.2 Power-on Reset

All UART registers are forced to their Reset states upon a Power-on Reset.

21.16.3 Watchdog Reset

All UART registers are unchanged upon a Watchdog Reset.

21.17 DESIGN TIPS

Question 1: ***The data I transmit with the UART is not received correctly. What could cause this?***

Answer: The most common reason for reception errors is that an incorrect value has been calculated for the UART Baud Rate Generator. Ensure the value written to the UxBRG register is correct.

Question 2: ***I am getting framing errors even though the signal on the UART receive pin looks correct. What are the possible causes?***

Answer: Ensure the following control bits have been set up correctly:

- BRGH (UxBRG<15:0) Baud Rate Divisor bits
- PDSEL (UxMODE<1:0>) Parity and Data Selection bits
- STSEL (UxMODE<0>) Stop Selection bit

21.18 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the PIC32MX device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the UART module are:

Title	**Application Note #**
No related application notes at this time.	N/A

Note: Please visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the PIC32MX device family.

21.19 REVISION HISTORY

Revision A (August 2007)

This is the initial released version of this document.

Revision B (October 2007)

Updated document to remove Confidential status.

Revision C (April 2008)

Revised status to Preliminary; Revised U-0 to r-x; Revised Register 21-1 bit 10; Revised Table 21-1, IEC1; Revised Register 21-16, bit 25; Revised Register 21-18, bit 25; Revised bit names.

Revision D (June 2008)

Revised Section 21.1; Added Footnote number to Registers 21-15-21-20; Change Reserved bits from “Maintain as” to “Write”; Added Note to ON bit (UxMODE Register).