Section 34. UART (Part II)

HIGHLIGHTS

This section of the manual contains the following major topics:

34.1 Introduction

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the dsPIC30F Switch Mode Power Supply (SMPS) device family. The UART is a full-duplex, asynchronous communication channel that communicates with peripheral devices and personal computers, using protocols such as RS-232, RS-485, LIN 1.2 and IrDA®. The module also includes the IrDA encoder and decoder.

The primary features of the UART module are:

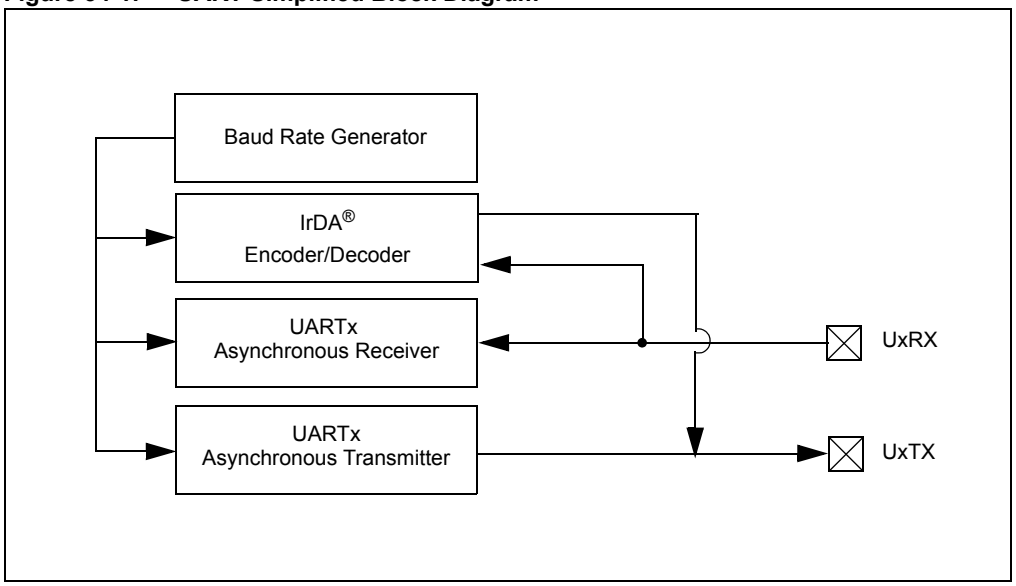
- Full-duplex, 8- or 9-bit data transmission through the UxTX and UxRX pins
- Even, odd or no parity options (for 8-bit data)
- One or two Stop bits
- Hardware Auto-Baud feature
- Fully Integrated Baud Rate Generator with 16-bit prescaler
- 4-deep First-In First-Out (FIFO) Transmit Data buffer
- 4-deep FIFO Receive Data buffer
- Parity, framing and buffer-overrun error detection
- Support for 9-bit mode with Address Detect (9th bit = 1)
- Transmit and Receive interrupts
- Loopback mode for diagnostic support
- IrDA encoder and decoder logic
- LIN 1.2 protocol support

Note: Each dsPIC30F SMPS device variant has one or more UART modules. An 'x' used in the names of pins, control/status bits and registers denotes the particular UART module. For more details, refer to the specific device data sheets.

Figure 34-1 displays a simplified block diagram of the UART. The UART module consists of the following key hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver
- IrDA Encoder and Decoder Logic

Figure 34-1: UART Simplified Block Diagram



34.2 Control Registers

Register 34-1: UxMODE: UARTx Mode Register

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-0	U-0	R/W-0	U-0	U-0
UARTEN	—	USIDL	IREN	—	ALTIO	—	—
bit 15							bit 8

Lower Byte:							
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

bit 15 **UARTEN:** UARTx Enable bit
1 = Enable UARTx; UARTx pins are controlled by UARTx module
0 = Disable UARTx; UARTx pins are controlled by I/O Port control registers

bit 14 **Unimplemented:** Read as '0'

bit 13 **USIDL:** Stop in Idle Mode bit
1 = Discontinue operation when device enters Idle mode
0 = Continue operation in Idle mode

bit 12 **IREN:** IrDA Encoder and Decoder Enable bit
1 = Enable IrDA encoder and decoder
0 = Disable IrDA encoder and decoder

Note: This feature is available for the 16x BRG mode (BRGH = 0).

bit 11 **Unimplemented:** Read as '0'

bit 10 **ALTIO:** UARTx Alternate I/O Selection bit
1 = Use UxATX and UxARX I/O pins
0 = Use UxTX and UxRX I/O pins

bit 9-8 **Unimplemented:** Read as '0'

bit 7 **WAKE:** Enable Wake-up on Start bit
1 = Enable Wake-up
0 = Disable Wake-up

bit 6 **LPBACK:** UARTx Loopback Mode Select bit
1 = Enable Loopback mode
0 = Disable Loopback mode

bit 5 **ABAUD:** Auto-Baud Enable bit
1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in 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 Select bit
1 = High speed
0 = Low speed

Register 34-1: UxMODE: UARTx Mode Register (Continued)

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

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Register 34-2: UxSTA: UARTx Status and Control Register

Upper Byte:							
R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R-0	R-1
UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

bit 15,13 **UTXISEL<1:0>:** Transmission Interrupt Mode Selection bits
11 = Reserved
10 = Interrupt generated when a character is transferred to the Transmit Shift register and the transmit buffer becomes empty
01 = Interrupt generated when the last transmission is over (last character shifted out of Transmit Shift register) and all the transmit operations are completed
00 = Interrupt generated when any character is transferred to the Transmit Shift Register (this implies at least one location is empty in the transmit buffer)

bit 14 **UTXINV:** IrDA Encoder Transmit Polarity Inversion bit
1 = IrDA encoded UxTX Idle state is '1'
0 = IrDA encoded UxTX Idle state is '0'

Note: Value of the bit affects the transmit properties of the module only when the IrDA encoder is enabled (IREN = 1).

bit 12 **Unimplemented:** Read as '0'

bit 11 **UTXBRK:** Transmit Break bit
1 = UxTX pin is driven low regardless of transmitter state (Sync Break transmission - Start bit followed by twelve '0's and followed by a Stop bit)
0 = Sync Break transmission is disabled or completed

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

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 data word 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 bits
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

Register 34-2: UxSTA: UARTx Status and Control Register (Continued)

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 (clear/read-only)
1 = Receive buffer has overflowed
0 = Receive buffer has not overflowed (clearing a previously set OERR bit resets the receiver buffer and RSR to empty state)

bit 0 **URXDA:** 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

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Register 34-3: UxRXREG: UARTx Receive Register

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0
—	—	—	—	—	—	—	URX8
bit 15							bit 8

Lower Byte:							
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
URX<7:0>							
bit 7							bit 0

bit 15-9 **Unimplemented:** Read as '0'

bit 8 **URX8:** Data bit 8 of the Received Character (in 9-bit mode)

bit 7-0 **URX<7:0>:** Data bits 7-0 of the Received Character

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Register 34-4: UxTXREG: UARTx Transmit Register (Write-Only)

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	W-x
—	—	—	—	—	—	—	UTX8
bit 15							bit 8

Lower Byte:							
W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
UTX<7:0>							
bit 7							bit 0

bit 15-9 **Unimplemented:** Read as '0'

bit 8 **UTX8:** Data bit 8 of the Transmitted Character (in 9-bit mode)

bit 7-0 **URX<7:0>:** Data bits 7-0 of the Transmitted Character

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Register 34-5: UxBRG: UARTx Baud Rate Register

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

Lower Byte:							
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

bit 15 **BRG<15:0>:** Baud Rate Divisor bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

34.3 UART Baud Rate Generator (BRG)

The UART has a full 16-bit baud rate generator to allow maximum flexibility in baud-rate generation. The BRG operates in either high-speed or low-speed modes, as selected by the High Baud Rate Select (BRGH) bit in the UART Mode (UxMODE<3>) register.

- High-speed mode (BRGH = 1) generates 4 clock pulses for every bit time
- Low-speed mode (BRGH = 0) generates 16 clock pulses for every bit time

Equation 34-1 shows the formula for computing the baud rate for low-speed mode (BRGH = 0).

Equation 34-1: UART Baud Rate with BRGH = 0

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

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

Note: FCY denotes the instruction cycle clock frequency (FOSC/2).

Example 34-1 shows how to calculate the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600
- BRGH = 0

Example 34-1: Baud Rate Error Calculation (BRGH = 0)

Desired Baud Rate = FCY/(16 (UxBRG + 1))

Solving for UxBRG value:

UxBRG = ((FCY/Desired Baud Rate)/16) - 1
UxBRG = ((4000000/9600)/16) - 1
UxBRG = 25

Calculated Baud Rate = 4000000/(16 (25 + 1))
= 9615

$$\text{Error} = \frac{\text{(Calculated Baud Rate - Desired Baud Rate)}}{\text{Desired Baud Rate}}$$

= (9615 - 9600)/9600
= 0.16%

The maximum baud rate (BRGH = 0) possible is FCY/16 (for UxBRG = 0), and the minimum baud rate possible is FCY/(16 • 65536).

Equation 34-2 shows the formula for computing the baud rate with BRGH = 1.

Equation 34-2: UART Baud Rate with BRGH = 1

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

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

Note: FCY denotes the instruction cycle clock frequency.

The maximum baud rate (BRGH = 1) possible is FCY/4 (for UxBRG = 0), and the minimum baud rate possible is FCY/(4 • 65536).

Writing a new value to the UxBRG register causes the BRG timer to be reset (cleared). This ensures that the BRG does not wait for a timer overflow before generating the new baud rate.

34.4 UART Configuration

The UART module 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 can be configured 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 (POR) setting. The number of data bits and Stop bits and the parity, are specified in the PDSEL<1:0> bits (UxMODE<2:1>). The Stop bits are specified in the STSEL bits (UxMODE<0>). An on-chip, dedicated, 16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The UART transmits and receives the Least Significant bit (LSb) first. The UART module's transmitter and receiver are functionally independent, but use the same data format and baud rate.

34.4.1 Enabling the UART

To enable the UART module:

1. Set the UART Enable (UARTEN) bit (UxMODE<15>)
2. Then set the Transmit Enable (UTXEN) bit (UxSTA<10>)

When these configuration bits are 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 no transmission is taking place.

Note: The UTXEN bit should not be set until the UARTEN bit has been set; otherwise, UART transmissions are not enabled.

34.4.2 Disabling the UART

To disable the UART module, clear the UARTEN (UxMODE<15>) 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.

In the UARTx Status and Control (UxSTA) register, these bits are cleared:

- Receive Buffer Available (URXDA)
- Receiver Buffer Overrun Error Status (OERR)
- Framing Error Status (FERR)
- Parity Error Status (PERR)
- Transmit Enable (UTXEN)
- Transmit Break (UTXBRK)
- Transmit Buffer Full Status (UTXBF)

These bits are set:

- Receive Idle (RIDLE)
- Transmit Shift Register Empty (TRMT)

Other control bits, including the Address Character Detect (ADDEN), Receive Interrupt Mode Selection (URXISEL<1:0>) and Transmit Interrupt Mode Selection (UTXISEL<1:0>) bits, as well as the UxMODE and UxBRG registers, are not affected when the module is disabled.

Clearing the UARTEN bit while the UART is active aborts all pending transmission and reception and resets the module, as previously described. Re-enabling the UART restarts the UART in the same configuration.

34.4.3 Alternate UART I/O Pins

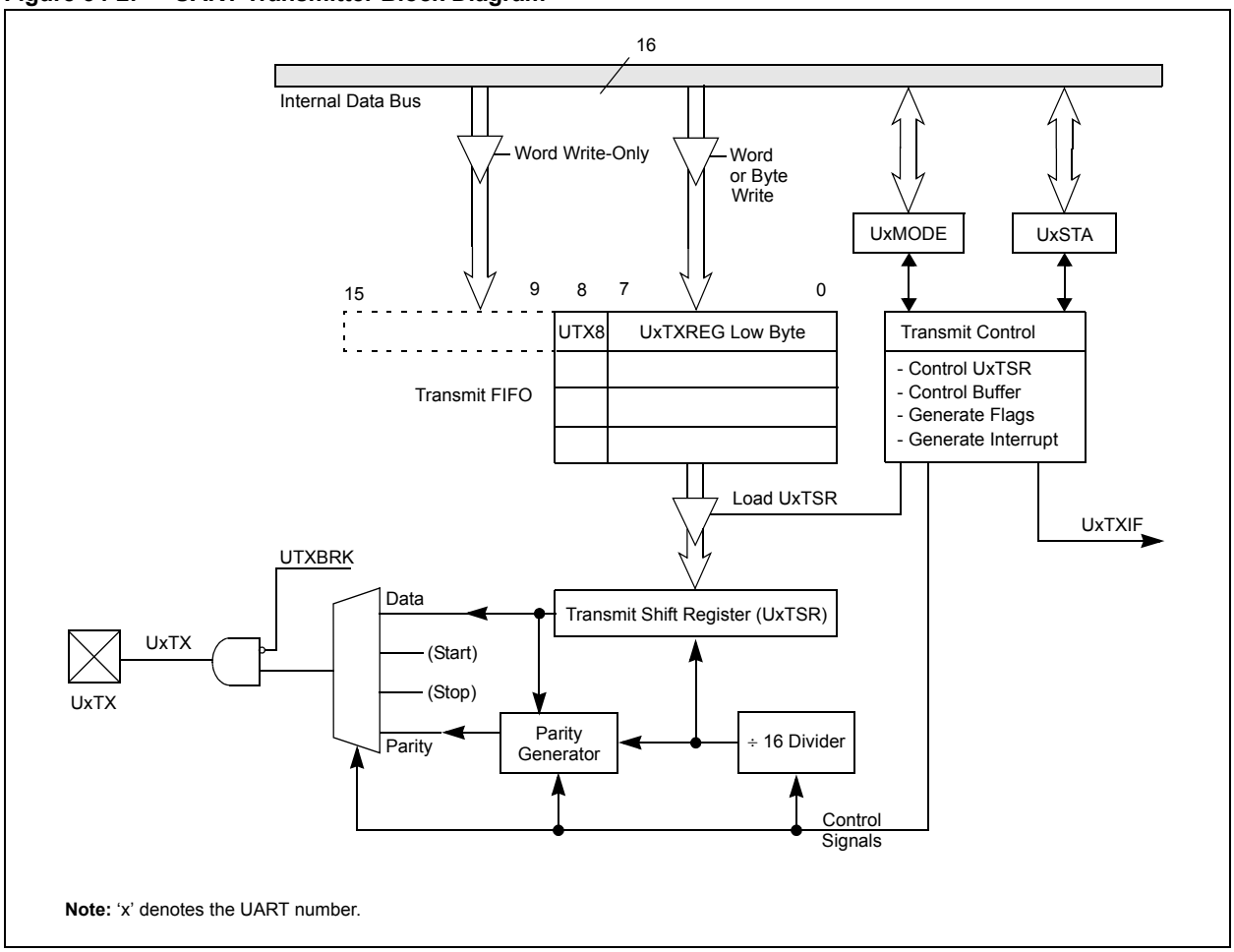
Some dsPIC30F SMPS devices have an alternate set of UART transmit and receive pins that can be used for communication. The alternate UART pins are useful when the primary UART pins are shared by other peripherals. The alternate I/O pins are enabled by setting the ALTIO bit (UxMODE<10>). If ALTIO = 1, the UART module uses the UxATX and UxARX pins (alternate transmit and alternate receive pins, respectively) instead of the UxTX and UxRX pins. If ALTIO = 0, the UART module uses the UxTX and UxRX pins.

34.5 UART Transmitter

Figure 34-2 shows the UART transmitter block diagram. The heart of the transmitter is the Transmit shift register (UxTSR). This shift register 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 has been 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 data memory, so it is not available to the user.

Figure 34-2: UART Transmitter Block Diagram



Note: 'x' denotes the UART number.

Enable transmission by setting the UTXEN enable bit (UxSTA<10>). The actual transmission does not occur until the UARTx Transmit (UxTXREG) register is loaded with data and the UARTx Baud Rate Generator (UxBRG) produces a shift clock (refer to Figure 34-2).

When transmission is first started, the UxTSR register is empty, so a transfer to the UxTXREG register results in an immediate transfer to UxTSR. Clearing the UTXEN bit during a transmission aborts the transmission and resets the transmitter. As a result, the UxTX pin reverts to a high-impedance state.

To select the 9-bit transmission, set the Parity and Data Selection (PDSEL<1:0>) bits (UxMODE<2:1>) to '11'. The ninth bit should be written to the UTX8 bit (UxTXREG<8>). A word write should be performed to 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.

There should be a delay between enabling the UARTx (UARTEN = 1) and initiating the first transmission. The delay is baud rate dependent and should be equal to, or longer than, the time it takes to transmit one data bit.

34.5.1 Transmit Buffer (UxTXREG)

The transmit buffer is 9 bits wide and 4 levels deep. When these are considered with the Transmit Shift registers (UxTSR), the user application effectively has a 5-level-deep buffer organized as FIFO. Once 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 when the buffer is full. If a user application attempts to write to a full buffer, the new data is not accepted into the FIFO buffer.

The FIFO buffer is reset during any device Reset, but it is not affected when the device enters or wakes up from a power-saving mode.

34.5.2 Transmit Interrupt

The Transmit Interrupt Flag (UxTXIF) is located in the corresponding Interrupt Flag Status (IFS) register. The UTXISEL<1:0> control bits (UxSTA<15,13>) determine when the UART generates a transmit interrupt.

- If UTXISEL<1:0> = 00, the UART Transmit interrupt is generated when a byte is transferred from UART Transmit FIFO buffer to UART Transmit Shift register
- If UTXISEL<1:0> = 01, the UART Transmit interrupt is generated when the UART Transmit FIFO buffer becomes empty
- If UTXISEL<1:0> = 10, the UART Transmit interrupt is generated when the UART Transmit FIFO buffer is empty and the UART Transmit shift register becomes empty

Figure 34-3 illustrates the UART Transmit interrupt generation for different UTXISEL<1:0> bit settings.

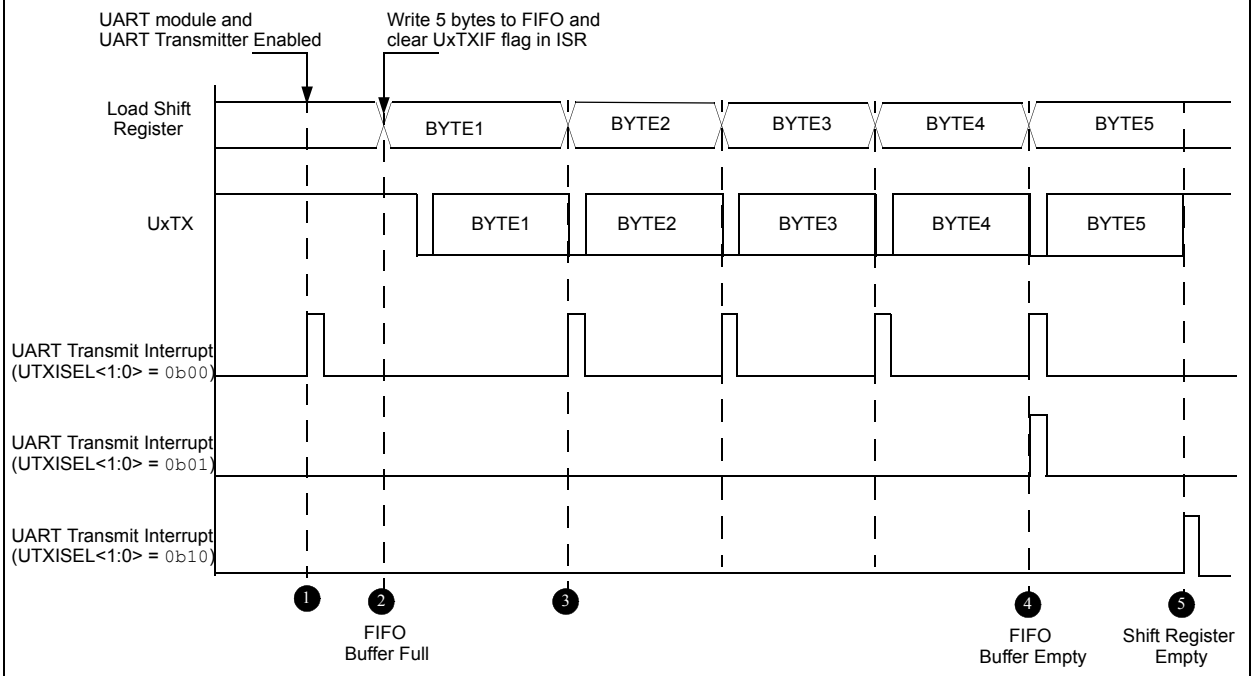
The UxTXIF bit is set when the module is first enabled. The user application should clear the UxTXIF bit in the ISR.

Switching between Interrupt modes during operation is possible.

Note:	When the UTXEN bit is set, the UxTXIF flag bit is also set, if UTXISEL<1:0> = 00, since the transmit buffer is not yet full (transmit data can be moved 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. The read-only TRMT status bit is set when the UxTSR is empty. No interrupt logic is tied to this bit, so the user application must poll this bit to determine if the UxTSR is empty.

Figure 34-3: Transmission (Back-to-Back)



1. Enable the UART module by setting the UARTEN (UxMODE<15>) bit. Enable UART transmission by setting the UTXEN (UxSTA<10>) bit. This generates a UART Transmit interrupt.
2. Write 5 bytes of data into the UART Transmit Register (UxTXREG), then clear the UART interrupt flag in the Interrupt Service Routine.
3. If UTXISEL<1:0> = 00, the UART Transmit interrupt is generated when a byte is transferred from UART Transmit FIFO buffer to UART Transmit Shift Register.
4. If UTXISEL<1:0> = 01, the UART Transmit interrupt is generated when the UART Transmit FIFO buffer becomes empty.
5. If UTXISEL<1:0> = 10, the UART Transmit interrupt is generated when the UART Transmit FIFO buffer is empty and the UART Transmit shift register becomes empty.

34.5.3 Setup for UART Transmit

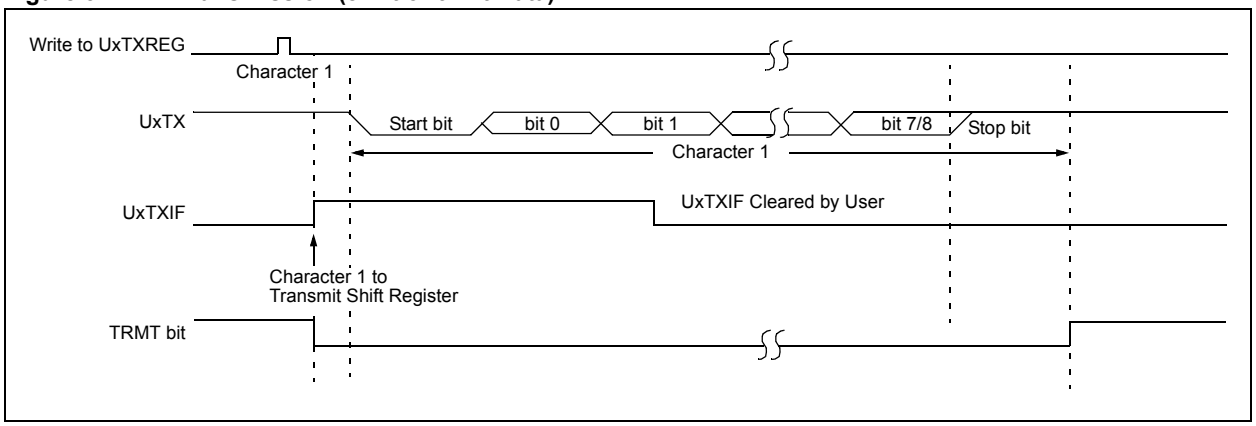
Follow these steps to set up UART transmission:

1. Initialize the UxBRG register for the appropriate baud rate (refer to **Section 34.3 "UART Baud Rate Generator (BRG)"**).
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 (IEC) register. Specify the interrupt priority for the transmit interrupt using the UxTXIP<2:0> control bits in the corresponding Interrupt Priority Control (IPC) register. Also, select the Transmit Interrupt mode by writing the UTXISEL<1:0> (UxSTA<15,13>) bits.
4. Enable the UART module by setting the UARTEN (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. Operation of the UxTXIF bit is controlled by the UTXISEL<1:0> control bits.
6. Load data to the UxTXREG register (which 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 UTXBF status bit (UxSTA<9>) is set.

Note:	The UTXEN bit should not be set until the UARTEN bit has been set; otherwise, UART transmissions are not enabled.

Example 34-2 provides sample code that sets up the UART for transmission. Figure 34-4 illustrates transmission after the UART is set up.

Figure 34-4: Transmission (8-Bit or 9-Bit Data)



Example 34-2: UART Transmission with Interrupts

```
#define FCY 29100000
#define BAUDRATE 9600
#define BRGVAL ((FCY/BAUDRATE)/16)-1

int main(void)
{
  U1MODEbits.STSEL = 0;      // 1-stop bit
  U1MODEbits.PDSEL = 0;      // No Parity, 8-data bits
  U1MODEbits.ABAUD = 0;      // Autobaud Disabled
  U1MODEbits.BRGH = 0;       // Low Speed mode

  U1BRG = BRGVAL;            // BAUD Rate Setting for 9600

  U1STAbits.UTXISEL = 0;     // Interrupt for every data transfer
  IEC0bits.U1TXIE = 1;       // Enable UART Transmit interrupt
  U1MODEbits.UARTEN = 1;     // Enable UART
  U1STAbits.UTXEN = 1;       // Enable UART Tx

  /* wait at least 104 usec (1/9600) before sending first char */
  for(i = 0; i < 4160; i++){
        Nop();
  }

  U1TXREG = 'a';             // Transmit one character

  while(1) {}
}

//UART Transmit ISR
void __attribute__((interrupt, no_auto_psv)) _U1TXInterrupt(void)
{
  IFS0bits.U1TXIF = 0;       // clear TX interrupt flag
  U1TXREG = 'a';             // Transmit one character
}
```

34.5.4 Transmission of Break Characters

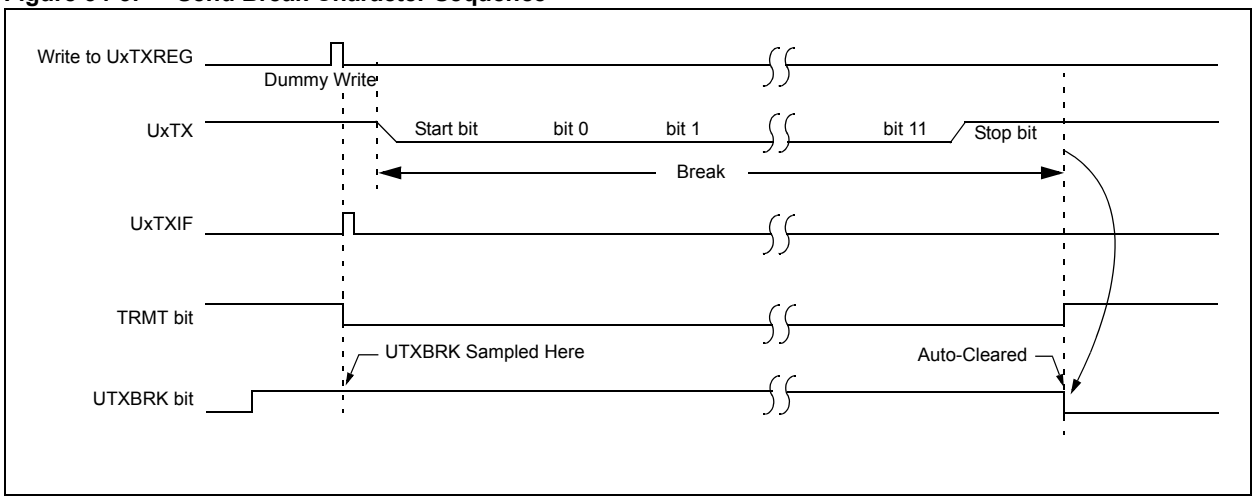
A Break character transmission consists of a Start bit, followed by 12 zeros and a Stop bit. A Frame Break character is sent whenever the UTXBRK and UTXEN bits are set while the Transmit Shift register is loaded with data. A dummy write to the UxTXREG register is necessary to initiate the Break character transmission. The data value written to the UxTXREG for the Break character is ignored. The write operation initiates the proper sequence by transmitting all zeros.

The UTXBRK bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user application 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 application should wait for the transmitter to be Idle (TRMT = 1) before setting the UTXBRK bit. The UTXBRK bit overrides any other transmitter activity. If the user application clears the TXBRK bit prior to sequence completion, unexpected module behavior can result. Sending a Break character does not generate a transmit interrupt.

The TRMT bit indicates when the Transmit Shift register is empty or full, just as it does during normal transmission. Figure 34-5 shows the timing of the Break character sequence.

Figure 34-5: Send Break Character Sequence



34.5.4.1 Break and Sync Transmit Sequence

The following sequence sends a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the UART for the desired mode.
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 '55h' to UxTXREG to load a Sync character into the transmit FIFO buffer.

After the Break is sent, the UTXBRK bit is reset by hardware. The Sync character is then transmitted.

34.6 UART Receiver

Figure 34-8 shows the receiver block diagram. 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 the UxRX pin is sampled for the Stop bit, the received data in UxRSR is transferred to the receive FIFO buffer (if it is empty).

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

The data on the UxRX pin is sampled multiple times by a majority detect circuit to determine if a high or a low signal is present at the UxRX pin.

34.6.1 Receive Buffer (UxRXREG)

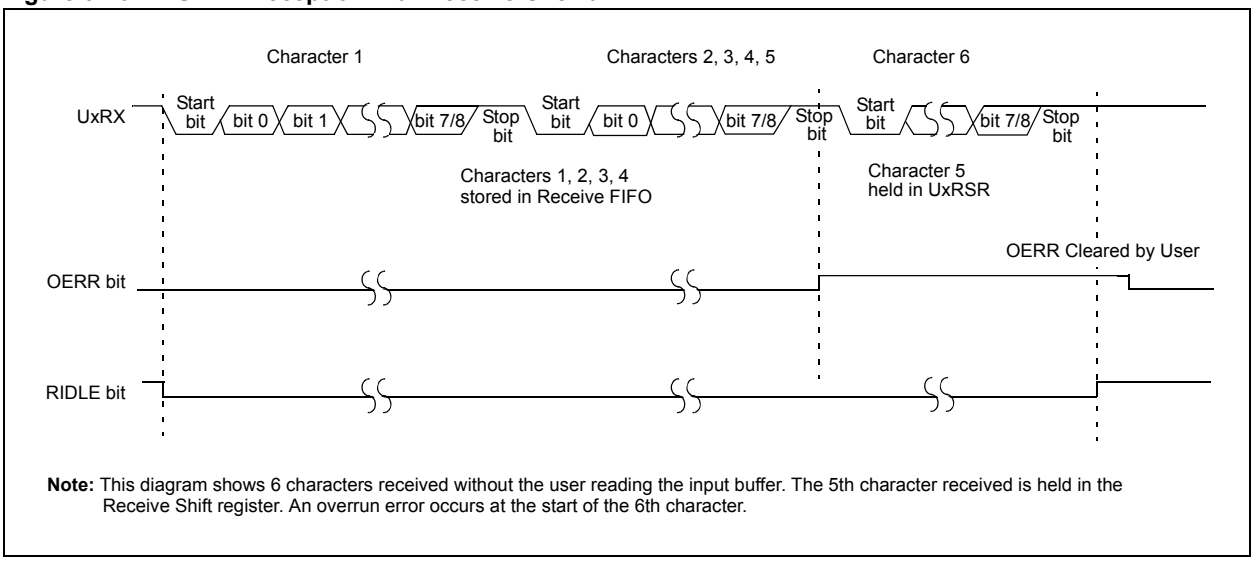
The UART receiver has a 4-deep, 9-bit wide FIFO receive data buffer. UxRXREG is a memory mapped register that provides access to the output of the FIFO. It is possible for four words of data to be received and transferred to the FIFO buffer and a fifth word to begin shifting to the UxRSR register before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

34.6.2 Receiver Error Handling

If the FIFO buffer is full (four characters) and a fifth character is fully received into the UxRSR register, the Overrun Error (OERR) bit (UxSTA<1>) is set. The word in UxRSR is retained, but further transfers to the receive FIFO buffer are inhibited as long as the OERR bit is set. The user application 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 application should first read all five characters, then clear the OERR bit. If the five characters can be discarded, the user application can simply clear the OERR bit. This effectively resets the receive FIFO and all prior received data is lost. Figure 34-6 illustrates an overrun condition.

Figure 34-6: 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.

The Framing Error (FERR) bit (UxSTA<2>) is set if a Stop bit is detected at a logic-low level.

The Parity Error (PERR) bit (UxSTA<3>) is set if a parity error has been detected in the data word at the top of the buffer (the current word). For example, a parity error occurs if the parity is set to be even, but the total number of ones in the data are detected to be 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 the data word is read.

34.6.3 Receive Interrupt

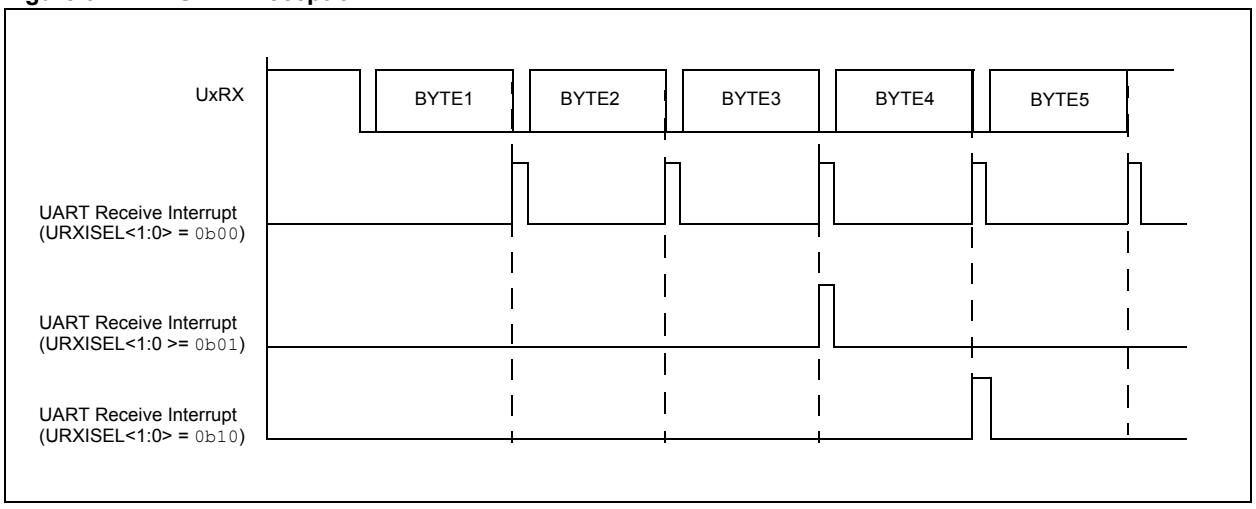
The UART Receive Interrupt Flag (UxRXIF) is located in the corresponding Interrupt Flag Status (IFS) register. The URXISEL<1:0> (UxSTA<7:6>) control bits determine when the UART receiver generates an interrupt.

- If URXISEL<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
- If URXISEL<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 3 or 4 characters
- If URXISEL<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 4 characters (i.e., becomes full)

Figure 34-7 illustrates the UART Receive interrupt generation for different URXISEL<1:0> bit settings.

It is possible to switch between Interrupt modes during operation.

Figure 34-7: UART Reception

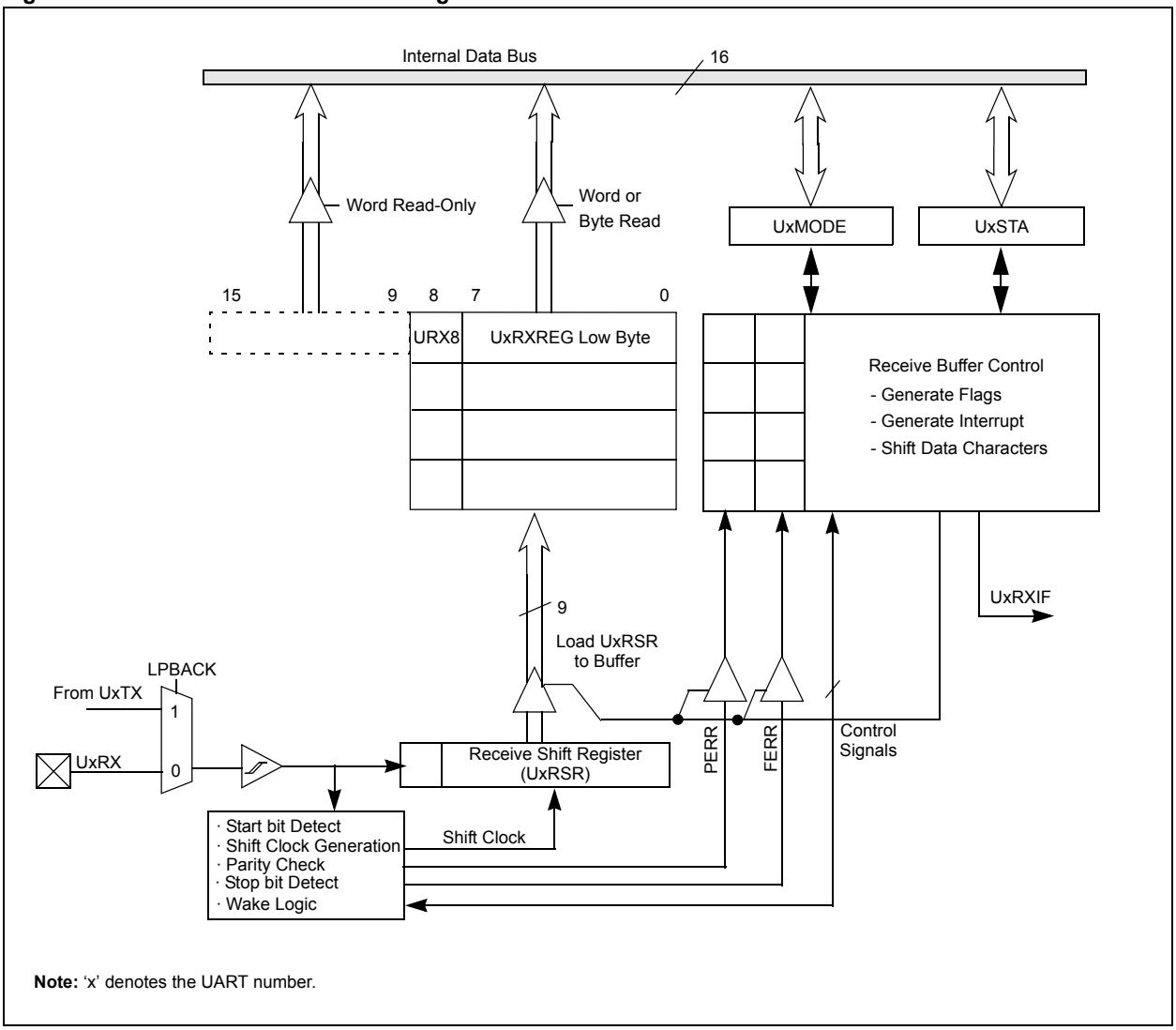


The URXDA and UxRXIF flag bits indicate the status of the UxRXREG register. The RIDLE bit (UxSTA<4>) shows the status of the UxRSR register. The read-only RIDLE status bit is set when the receiver is Idle (i.e., the UxRSR register is empty). No interrupt logic is tied to this bit, so the user application must poll this bit to determine if the UxRSR is Idle.

The URXDA 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. URXDA is a read-only bit.

Figure 34-8 shows a block diagram of the UART receiver.

Figure 34-8: UART Receiver Block Diagram



Note: 'x' denotes the UART number.

34.6.4 Setup for UART Reception

Use these steps to set up UART reception:

1. Initialize the UxBRG register for the appropriate baud rate (refer to **Section 34.3 "UART Baud Rate Generator (BRG)"**).
2. Set the number of data bits, number of Stop bits and parity selection by writing appropriate values 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 for the interrupt using the UxRXIP<2:0> control bits in the corresponding Interrupt Priority Control register (IPC). Also, select the Receive Interrupt mode by writing to the URXISEL<1:0> (UxSTA<7:6>) bits.
4. Enable the UART module by setting the UARTEN (UxMODE<15>) bit.

Receive interrupts depend on the URXISEL<1:0> control bit settings. If receive interrupts are not enabled, the user application can poll the URXDA bit. The UxRXIF bit should be cleared in the software routine that services the UART receive interrupt.

Read data from the receive buffer. If 9-bit transmission is selected, read a word; otherwise, read a byte. The URXDA status bit (UxSTA<0>) is set whenever data is available in the buffer.

Example 34-3 provides sample code that sets up the UART for reception.

Example 34-3: UART Receiving with Interrupt

```
    #define FCY 29100000
    #define BAUDRATE 9600
    #define BRGVAL ((FCY/BAUDRATE)/16)-1

    int main(void)
    {

        U1MODEbits.STSEL = 0;      // 1-stop bit
        U1MODEbits.PDSEL = 0;      // No Parity, 8-data bits
        U1MODEbits.ABAUD = 0;      // Autobaud Disabled
        U1MODEbits.BRGH = 0;       // Low Speed mode

        U1BRG = BRGVAL;            // BAUD Rate Setting for 9600
        U1STAbits.URXISEL = 0;     // Interrupt after a character is received
        IEC0bits.U1RXIE = 1;       // Enable UART Receive interrupt
        U1MODbits.UARTEN = 1;      // Enable UART

        while(1)
        {
        }
    }
    void __attribute__((interrupt, no_auto_psv)) _UxTXInterrupt(void)
    {
        if(U1STAbits.OERR == 1)  {
         U1STAbits.OERR = 0;       // Clear Overrun Error to receive data
        } elseif ((U1STAbits.FERR ==0) && (U1STAbits.PERR ==0) )  {
         ReceivedChar = U1RXREG;  // Read Data if there is no parity or
framing
            // error
        }

        IFS0bits.U1RXIF = 0;       // clear TX interrupt flag
}
```

34.7 Using the UART for 9-Bit Communication

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

34.7.1 Multiprocessor Communications

A typical multiprocessor communication protocol differentiates between data bytes and address/control bytes. A common scheme uses a 9th data bit to identify whether a data byte contains 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 as follows:

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

34.7.2 ADDEN Control Bit

The UART receiver has an Address Detect mode that 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. Enable this feature by setting the ADDEN bit (UxSTA<5>).

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

34.7.3 Setup for 9-Bit Transmission

The setup procedure for 9-bit transmission is identical to that for 8-bit Transmit modes, except that PDSEL<1:0> bits (UxMODE<2:1>) should be set to '11' (refer to **Section 34.5.3 "Setup for UART Transmit"**).

To start transmissions, words should be written to the UxTXREG register.

34.7.4 Setup for 9-Bit Reception Using Address Detect Mode

The setup process for 9-bit reception is similar to that for 8-bit Receive modes, except the PDSEL<1:0> bits (UxMODE<2:1>) should be set to '11' (refer to **Section 34.6.4 "Setup for UART Reception"**).

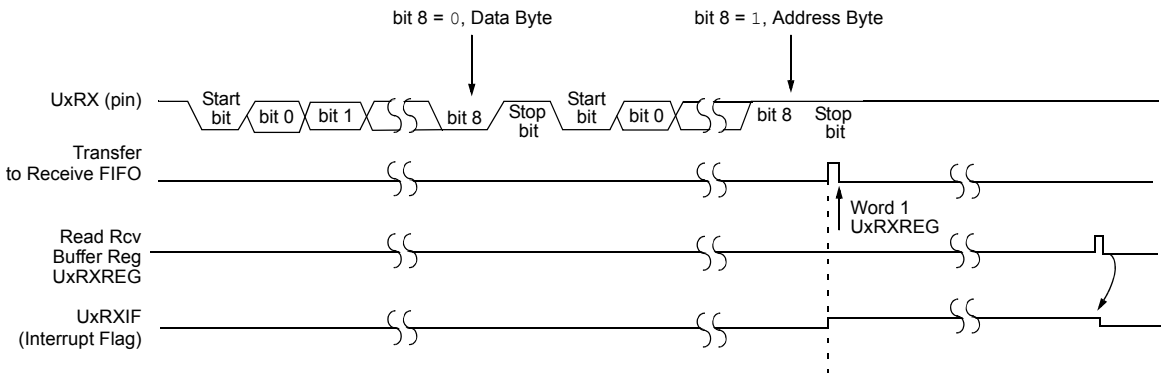
The Receive Interrupt mode should be configured by writing to the URXISEL<1:0> (UxSTA<7:6>) bits.

Note:	If the Address Detect mode is enabled (ADDEN = 1), the URXISEL<1:0> control bits should be configured so that an interrupt is generated after every received word. Each received data word must be checked in software for an address match immediately after reception.

The process for using the Address Detect mode is as follows:

1. Set the ADDEN (UxSTA<5>) bit to enable address detect. Ensure that the URXISEL control bits are configured to generate an interrupt after each received word.
2. Check each 8-bit address by reading the UxRXREG register to determine if the device is being addressed.
   - If this device has not been addressed, discard the received word.
   - If this device has been addressed, clear the ADDEN bit to allow subsequent data bytes to be read into the receive buffer and interrupt the CPU. If a long data packet is expected, the Receive Interrupt mode could be changed to buffer more than one data byte between interrupts.
3. When the last data byte has been received, set the ADDEN bit so that only address bytes are received. Also, ensure that the URXISEL control bits are configured to generate an interrupt after each received word.

Figure 34-9: 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.

34.8 Receiving Break Characters

The receiver counts and expects a certain number of bit times based on the values programmed in the PDSEL (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 PDSEL and STSEL. The URXDA bit is set, FERR is set, zeros are loaded into the receive FIFO, interrupts are generated, if appropriate, and the RIDLE bit is set.

When the module receives a break signal, and the receiver has detected the Start bit, the data bits and the invalid Stop bit (which sets the FERR), the receiver must wait for a valid Stop bit before looking for the next Start bit. It cannot assume that the break condition on the line is the next Start bit. Break is regarded as a character containing all '0's with the FERR bit set. The break character is loaded into the buffer. No further reception occurs until a Stop bit is received. RIDLE goes high when the Stop bit is received.

34.9 Other Features of the UART Module

34.9.1 UART in Loopback Mode

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

To select Loopback mode, do the following:

1. Configure UART for the desired mode of operation.
2. Enable transmission as defined in **Section 34.5 "UART Transmitter"**.
3. Set LPBACK = 1 (UxMODE<6>) to enable Loopback mode.

Table 34-1 shows how the Loopback mode is dependent on the UEN<1:0> bits.

Table 34-1: Loopback Mode Pin Function

UEN<1:0>	Pin Function, LPBACK = 1(1)
00	UxRX input connected to UxTX; UxTX pin functions; UxRX pin ignored; UxCTS/UxRTS unused
01	UxRX input connected to UxTX; UxTX pin functions; UxRX pin ignored; UxRTS pin functions, UxCTS unused
10	UxRX input connected to UxTX; UxTX pin functions; UxRX pin ignored; UxRTS pin functions, UxCTS input connected to UxRTS; UxCTS pin ignored
11	UxRX input connected to UxTX; UxTX pin functions; UxRX pin ignored; BCLKx pin functions; UxCTS/UxRTS unused

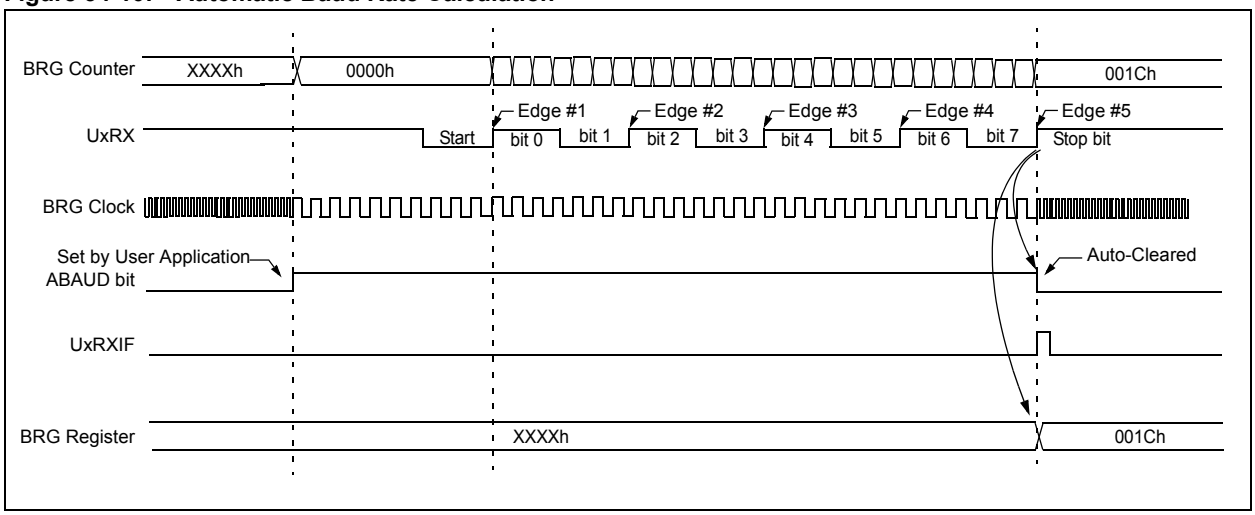
Note 1: LPBACK should be set to '1' only after enabling the other bits associated with the UART module.

34.9.2 Auto-Baud Support

The Auto Baud Enable (ABAUD) bit (UxMODE<5>), when set, allows the system to determine baud rates of the received characters. The UART begins an automatic baud rate measurement sequence whenever a Start bit is received while ABAUD = 1. The calculation is self-averaging. This feature is active only when WAKE = 0 and LPBACK = 0. Once the ABAUD bit is set, the BRG counter value is cleared and looks for a Start bit, which, for auto-baud support, is defined as a high-to-low transition followed by a low-to-high transition.

Following the Start bit, the auto-baud feature expects to receive an ASCII 'U' ('55h') in order to calculate the proper bit rate. The measurement is taken over both the low and 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 TCY/8 clock. On the 5th rising edge of the UxRX pin, an accumulated BRG counter value totaling the proper BRG period is transferred to the UxBRG register. The ABAUD bit automatically clears. If the user application clears the ABAUD bit before the sequence completes, unexpected module behavior can result. For the Auto-Baud Rate Detection (ABD) sequence, refer to Figure 34-10.

Figure 34-10: Automatic Baud Rate Calculation



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

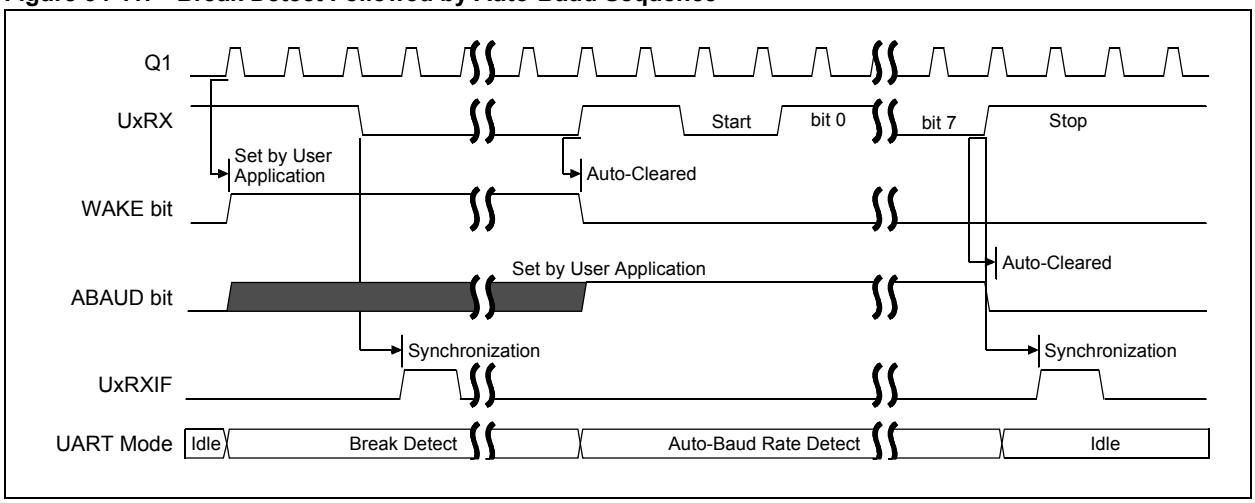
34.9.2.1 Break Detect Sequence

The user application can configure the auto-baud sequence to occur immediately following the Break detect by setting the ABAUD bit with the WAKE bit set. Figure 34-11 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 application 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. The user application should ensure that the ABAUD bit is not set while a transmit sequence is already in progress. Otherwise, the UART can exhibit unpredictable behavior.

Figure 34-11: Break Detect Followed by Auto-Baud Sequence



34.10 Infrared Support

34.10.1 Built-In IrDA Encoder and Decoder

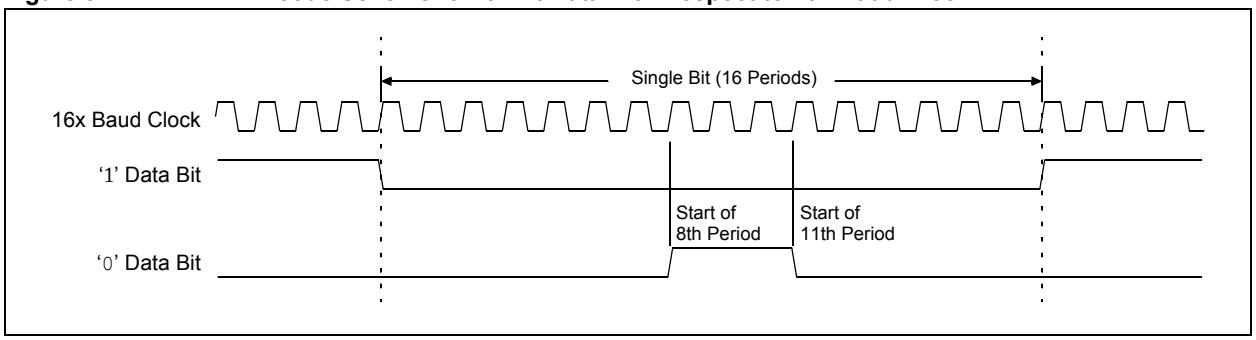
The UART module includes full implementation of the IrDA encoder and decoder. The built-in IrDA encoder and decoder functionality is enabled using the IrDA Encoder and Decoder Enable (IREN) bit in the UARTx Mode (UxMODE<12>) register. When IREN = 1, the receive pin (UxRX) acts as the input from the infrared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

34.10.1.1 IrDA Encoder Function

Figure 34-12 shows how a data bit from the IrDA encoder coincides with 16 periods of the 16x baud clock.

- A data bit with a value of '1' is encoded by the IrDA as '0' for the entire 16-period clock cycle
- A data bit with a value of '0' is encoded by the IrDA 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

Figure 34-12: IrDA® Encode Scheme for '0' Bit Data with Respect to 16x Baud Clock



34.10.1.2 IrDA Transmit Polarity

The IrDA encoder can be configured to control the transmit polarity on the UxTX pin. The transmit polarity is selected by the IrDA Encoder Transmit Polarity Inversion (UTXINV) bit (UxSTA<14>). This bit affects the module only when the IrDA encoder and decoder are enabled (IREN = 1).

- When UTXINV = 0, the Idle state of the UxTX line is '0' (refer to Figure 34-13)
- When UTXINV = 1, the Idle state of the UxTX line is '1' (refer to Figure 34-14)

Figure 34-13: IrDA® Encode Scheme (UTxINV = 0)

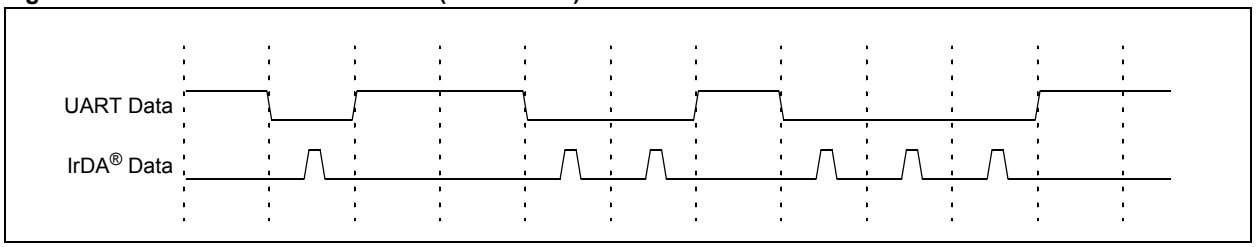
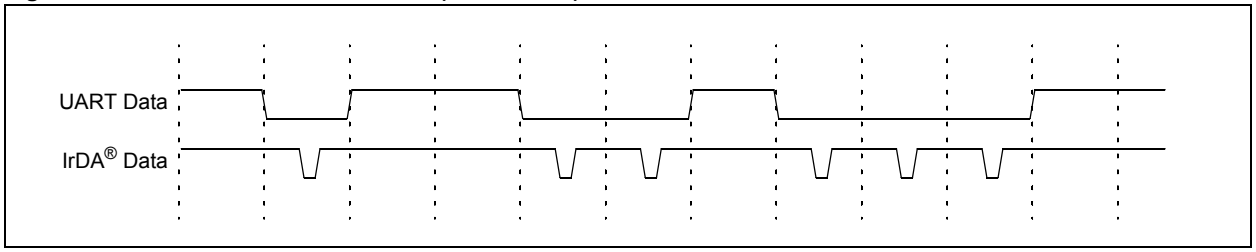


Figure 34-14: IrDA® Encode Scheme (UTxINV = 1)

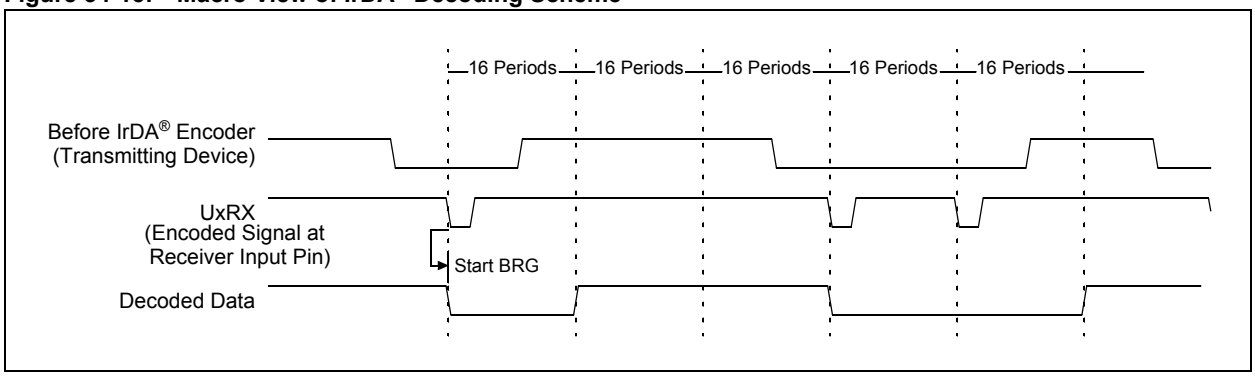


34.10.1.3 IrDA Decoder Function

The IrDA decoder receives serial data from the UxRX pin and replaces it with the decoded data stream. Figure 34-15 shows 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.

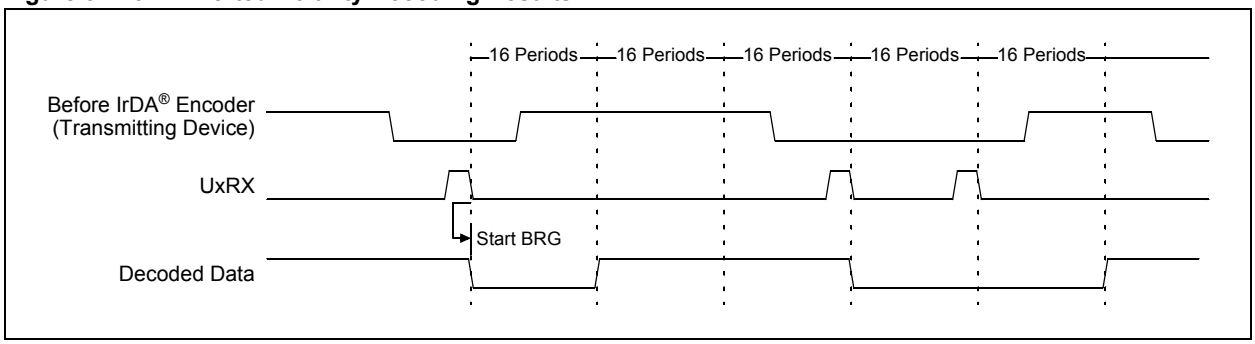
Figure 34-15: Macro View of IrDA® Decoding Scheme



34.10.1.4 IrDA Receive Polarity

The input of the IrDA signal can have an inverted polarity. Figure 34-16 shows the same logic is able to decode the signal train.

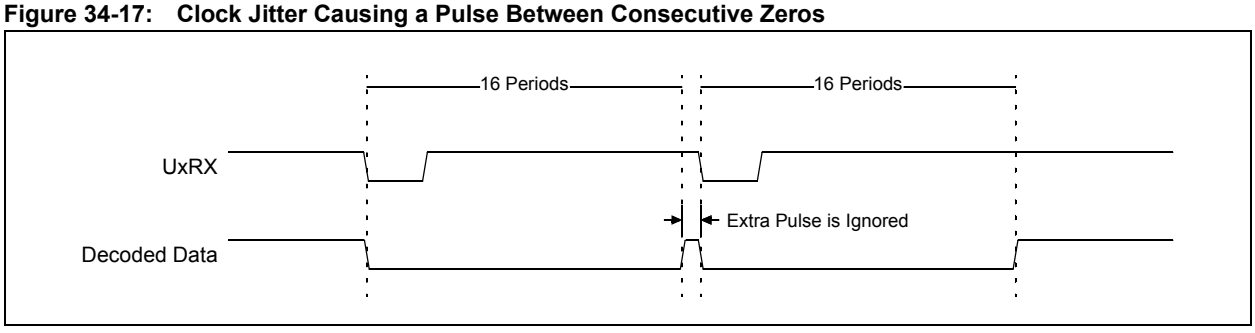
Figure 34-16: Inverted Polarity Decoding Results



34.10.1.5 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. Figure 34-17 shows that, 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.

Figure 34-17: Clock Jitter Causing a Pulse Between Consecutive Zeros



34.11 UART Operation During CPU Sleep and Idle Modes

The UART does not function in Sleep mode. If Sleep mode is entered while a transmission is in progress, the transmission is aborted and the UxTX pin is driven to logic '1'. Similarly, if Sleep mode is entered while a reception is in progress, the reception is aborted.

Note: The WAKE bit (UxMODE<7>) should be set just before entering Sleep mode to ensure that wake-up occurs only when a Start bit is detected while in sleep.

The UART resets itself during Sleep mode.

The following registers are not affected by entering or exiting Sleep mode:

- UxMODE and UxSTA registers
- Transmit and Receive registers and buffers
- UxBRG register

There is no automatic way to prevent entry into Sleep mode if a transmission or reception is pending. The user application can check the RIDLE bit before going to Sleep to avoid aborting reception. The user application is in control of the transmitter, so it must synchronize Sleep entry with UART operation to ensure that transmission is not aborted.

The Stop in Idle Mode (USIDL) bit configures the module to stop or continue on Idle.

- If USIDL = 0, the module continues operation in Idle mode
- If USIDL = 1, the module stops on Idle mode

The UART module performs the same procedures when stopped in Idle mode (USIDL = 1) as for Sleep mode.

34.11.1 Auto-Wake-up on Sync Break Character

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

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

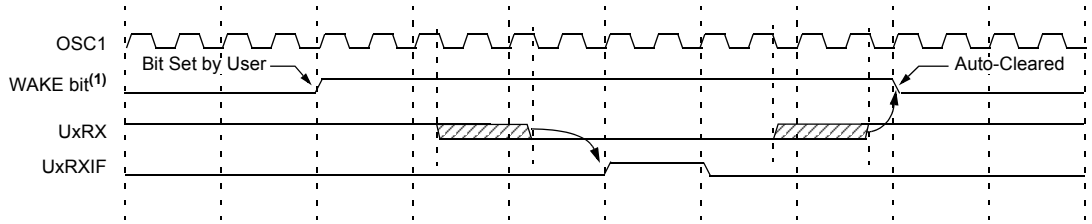
A wake-up event is a high-to-low transition on the UxRX line that coincides with the start of a Sync Break or a Wake-up Signal character for the LIN protocol. When WAKE is active, the UxRX line is monitored independently from the CPU mode. The UxRXIF interrupt is generated synchronously to the system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

The WAKE bit is automatically cleared after 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 returns to normal operation. This signals the user application that the Sync Break event is over. If the user application clears the WAKE bit prior to sequence completion, unexpected module behavior can result.

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

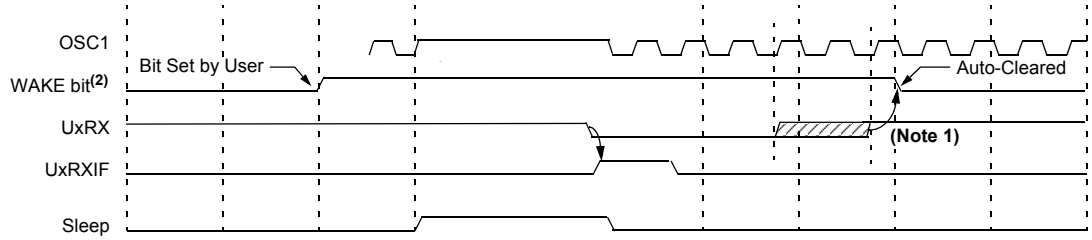
Note: The Sync Break (or Wake-up Signal) character must be of sufficient length to allow enough time for the selected oscillator to start and provide proper initialization of the UART module. To ensure that the part woke up in time, the user application should read the value of the WAKE bit. If it is 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 34-18: Auto-Wake-up Bit (WAKE) Timings During Normal Operation



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

Figure 34-19: 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 mode while the WAKE bit is active.

34.12 Registers Associated with UART Module

Table 34-2 summarizes the registers associated with the dsPIC30F SMPS UART1.

Table 34-2: Registers Associated with UART1

SFR Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
UxMODE	UARTEN	—	USIDL	IREN	—	ALTIO	—	—	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
UxSTA	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
UxTXREG	—	—	—	—	—	—	—	UTX8	Transmit Register								xxxx
UxRXREG	—	—	—	—	—	—	—	URX8	Receive Register								0000
UxBRG	Baud Rate Generator Prescaler																0000
IFS0	—	MI26IF	S12CIF	NVMIF	ADI F	U1TXIF	U1RXIF	SPI1IF	T3IF	T2IF	OC2IF	—	T1IF	OC1IF	IC1IF	IN0IF	0000
IEC0	—	MI26IE	S12CIE	NVMIE	ADI E	U1TXIE	U1RXIE	SPI1IE	T3IE	T2IE	OC2IE	—	T1IE	OC1IE	IC1IE	IN0IE	0000
IPC2	—	APID2	APID1	APID0	—	U1TXIP2	U1TXIP1	U1TXIP0	—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0	4444

Note: The registers associated with UART1 are shown for reference. The shaded bits are not used for UART operation. Refer to the device data sheet for the registers associated with other UART modules.

34.13 Design Tips

Question 1: ***The data I transmit with the UART does not get 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. Verify that 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:

- UxBRG: UART Baud Rate register
- PDSEL<1:0>: Parity and Data Size Selection bits
- STSEL: Stop bit Selection

34.14 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 dsPIC30F SMPS 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.

Note: For additional application notes and code examples for the dsPIC30F SMPS device family, visit the Microchip web site (www.microchip.com).

34.15 Revision History

Revision A (March 2007)

This is the initial released revision of this document.

Revision B (May 2007)

Minor updates were made to this document.

Revision C (February 2008)

- Removed first sentence in the third paragraph of **34.5 "UART Transmitter"**
- Added a note after the first paragraph in **34.11 "UART Operation During CPU Sleep and Idle Modes"**

NOTES: