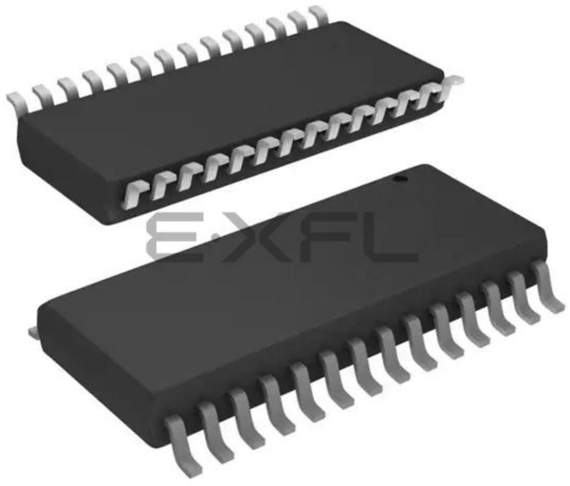


What is "Embedded - Microcontrollers"?

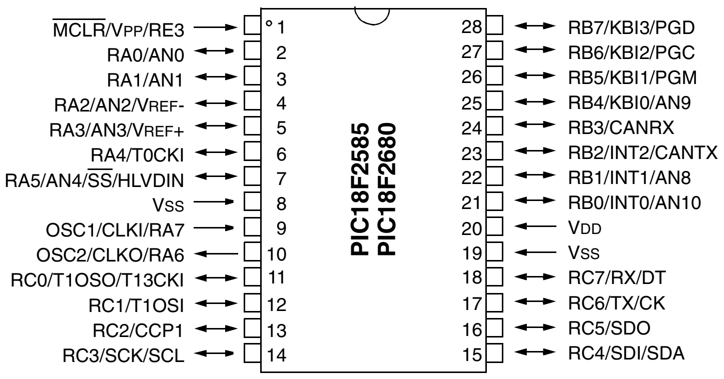
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2680-i-so

Pin Diagrams

28-Pin PDIP, SOIC



40-Pin PDIP

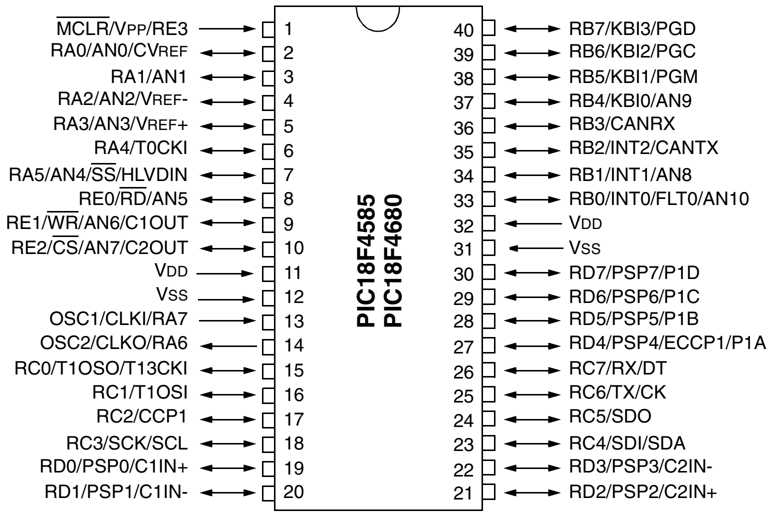
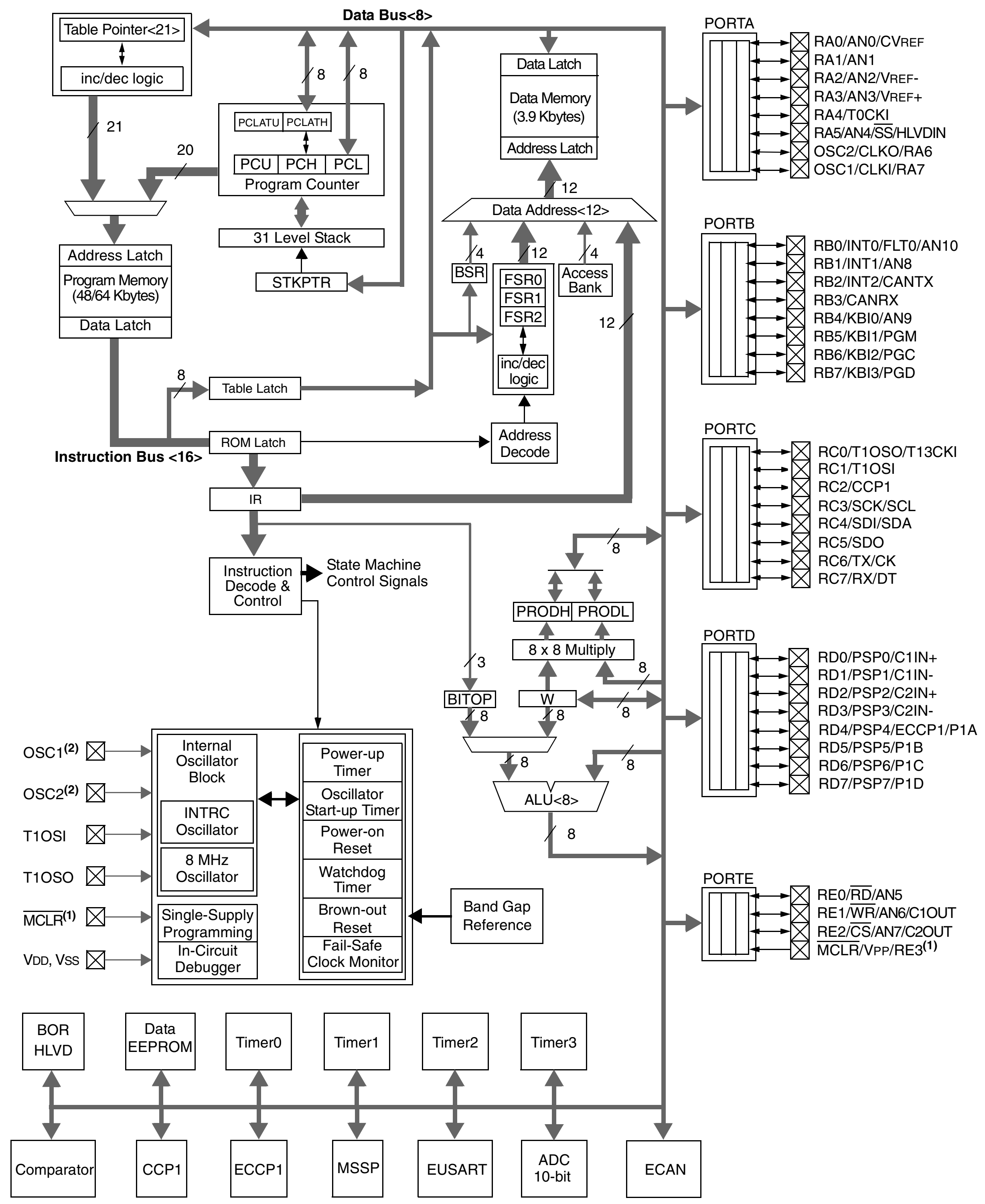


FIGURE 1-2: PIC18F4585/4680 (40/44-PIN) BLOCK DIAGRAM



Note 1: RE3 is multiplexed with MCLR and is only available when the MCLR Resets are disabled.

2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

TABLE 1-3: PIC18F4585/4680 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTD is a bidirectional I/O port or a Parallel Slave Port (PSP) for interfacing to a microprocessor port. These pins have TTL input buffers when PSP module is enabled.
RD0/PSP0/C1IN+	19	38	38			
RD0				I/O	ST	Digital I/O.
PSP0				I/O	TTL	Parallel Slave Port data.
C1IN+				I	Analog	Comparator 1 input (+).
RD1/PSP1/C1IN-	20	39	39			
RD1				I/O	ST	Digital I/O.
PSP1				I/O	TTL	Parallel Slave Port data.
C1IN-				I	Analog	Comparator 1 input (-)
RD2/PSP2/C2IN+	21	40	40			
RD2				I/O	ST	Digital I/O.
PSP2				I/O	TTL	Parallel Slave Port data.
C2IN+				I	Analog	Comparator 2 input (+).
RD3/PSP3/C2IN-	22	41	41			
RD3				I/O	ST	Digital I/O.
PSP3				I/O	TTL	Parallel Slave Port data.
C2IN-				I	Analog	Comparator 2 input (-).
RD4/PSP4/ECCP1/P1A	27	2	2			
RD4				I/O	ST	Digital I/O.
PSP4				I/O	TTL	Parallel Slave Port data.
ECCP1				I/O	ST	Capture2 input/Compare2 output/PWM2 output.
P1A				O	TTL	ECCP1 PWM output A.
RD5/PSP5/P1B	28	3	3			
RD5				I/O	ST	Digital I/O.
PSP5				I/O	TTL	Parallel Slave Port data.
P1B				O	TTL	ECCP1 PWM output B.
RD6/PSP6/P1C	29	4	4			
RD6				I/O	ST	Digital I/O.
PSP6				I/O	TTL	Parallel Slave Port data.
P1C				O	TTL	ECCP1 PWM output C.
RD7/PSP7/P1D	30	5	5			
RD7				I/O	ST	Digital I/O.
PSP7				I/O	TTL	Parallel Slave Port data.
P1D				O	TTL	ECCP1 PWM output D.

Legend:

TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	I	= Input
O	= Output	P	= Power

TABLE 15-3: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE, TIMER1 AND TIMER3

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
RCON	IPEN	SBOREN(3)	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	50
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	52
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	52
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	52
IPR2	OSCFIP	CMIP(2)	—	EEIP	BCLIP	HLVDIP	TMR3IP	ECCP1IP(2)	51
PIR2	OSCFIF	CMIF(2)	—	EEIF	BCLIF	HLVDIF	TMR3IF	ECCP1IF(2)	52
PIE2	OSCFIE	CMIE(2)	—	EEIE	BCLIE	HLVDIE	TMR3IE	ECCP1IE(2)	51
TRISB	PORTB Data Direction Register								52
TRISC	PORTC Data Direction Register								52
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								50
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								50
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	50
TMR3H	Timer3 Register High Byte								51
TMR3L	Timer3 Register Low Byte								51
T3CON	RD16	T3ECCP1(1)	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	51
CCPR1L	Capture/Compare/PWM Register 1 (LSB)								51
CCPR1H	Capture/Compare/PWM Register 1 (MSB)								51
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	51
ECCPR1L(1)	Enhanced Capture/Compare/PWM Register 1 (LSB)								51
ECCPR1H(1)	Enhanced Capture/Compare/PWM Register 1 (MSB)								51
ECCP1CON(1)	EPWM1M1	EPWM1M0	EDC1B1	EDC1B0	ECCP1M3	ECCP1M2	ECCP1M1	ECCP1M0	51

Legend: — = unimplemented, read as '0'. Shaded cells are not used by Capture and Compare, Timer1 or Timer3.

Note 1: These bits or registers are available on PIC18F4X8X devices only.

2: These bits are available on PIC18F4X8X devices and reserved on PIC18F2X8X devices.

3: The SBOREN bit is only available when CONFIG2L<1:0> = `01`; otherwise, it is disabled and reads as '0'.

17.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I^{2}C bus may be taken when the P bit is set or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I^{2}C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

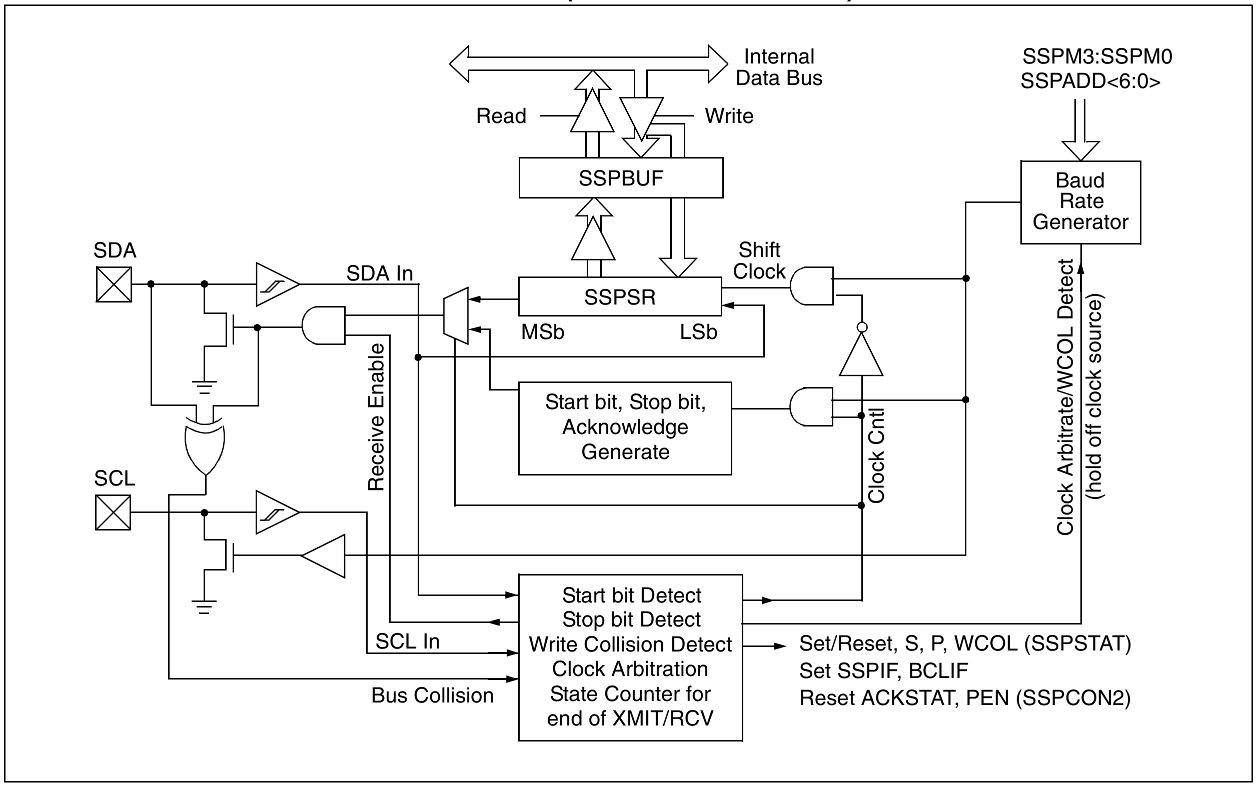
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I^{2}C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

Note:	The MSSP module, when configured in I^{2}C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP interrupt, if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmit
- Repeated Start

FIGURE 17-16: MSSP BLOCK DIAGRAM (I^{2}C™ MASTER MODE)



18.1 Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit or 16-bit generator that supports both the Asynchronous and Synchronous modes of the EUSART. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCON<3>) selects 16-bit mode.

The SPBRGH:SPBRG register pair controls the period of a free running timer. In Asynchronous mode, bits BRGH (TXSTA<2>) and BRG16 (BAUDCON<3>) also control the baud rate. In Synchronous mode, BRGH is ignored. Table 18-1 shows the formula for computation of the baud rate for different EUSART modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGH:SPBRG registers can be calculated using the formulas in Table 18-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 18-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 18-2. It may be advantageous to use the high baud rate (BRGH = `1`) or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGH:SPBRG registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

18.1.1 OPERATION IN POWER MANAGED MODES

The device clock is used to generate the desired baud rate. When one of the power managed modes is entered, the new clock source may be operating at a different frequency. This may require an adjustment to the value in the SPBRG register pair.

18.1.2 SAMPLING

The data on the RX pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 18-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
`0`	`0`	`0`	8-bit/Asynchronous	FOSC/[64 (n + 1)]
`0`	`0`	`1`	8-bit/Asynchronous	FOSC/[16 (n + 1)]
`0`	`1`	`0`	16-bit/Asynchronous	
`0`	`1`	`1`	16-bit/Asynchronous	FOSC/[4 (n + 1)]
`1`	`0`	`x`	8-bit/Synchronous	
`1`	`1`	`x`	16-bit/Synchronous	

Legend: `x` = Don't care, n = value of SPBRGH:SPBRG register pair

EXAMPLE 18-1: CALCULATING BAUD RATE ERROR

```
For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:
Desired Baud Rate      =   FOSC/(64 ([SPBRGH:SPBRG] + 1)
Solving for SPBRGH:SPBRG:
                  X    =   ((FOSC/Desired Baud Rate)/64) – 1
                       =   ((16000000/9600)/64) – 1
                       =   [25.042] = 25
Calculated Baud Rate   =   16000000/(64 (25 + 1))
                       =   9615
Error                  =   (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
                       =   (9615 – 9600)/9600 = 0.16%
```

TABLE 18-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	51
SPBRGH	EUSART Baud Rate Generator Register High Byte								51
SPBRG	EUSART Baud Rate Generator Register Low Byte								51

Legend: — = unimplemented, read as '`0`'. Shaded cells are not used by the BRG.

18.3 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the TX and RX pins to CK (clock) and DT (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK line. Clock polarity is selected with the SCKP bit (BAUDCON<4>); setting SCKP sets the Idle state on CK as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

18.3.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 18-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available).

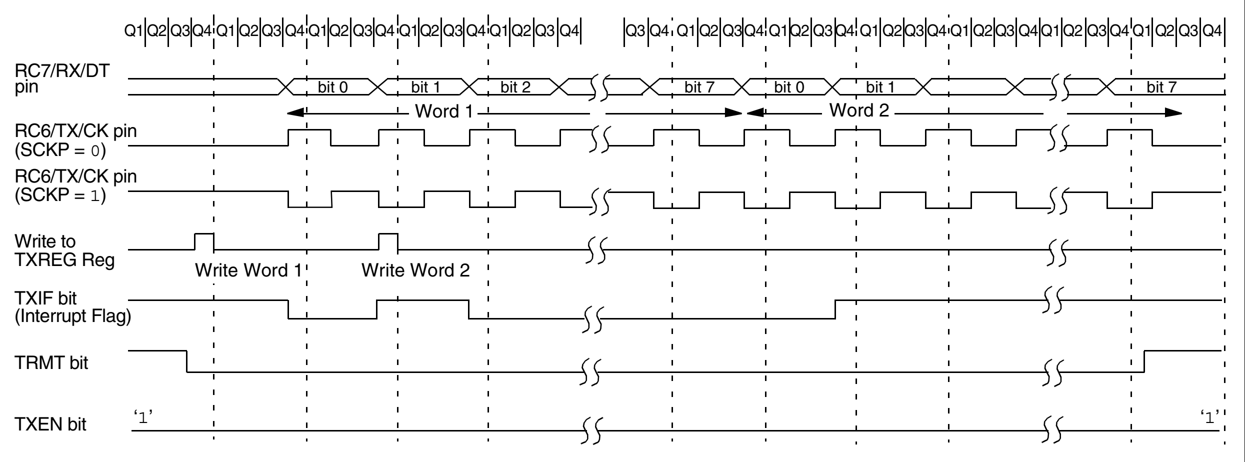
Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and the TXIF flag bit (PIR1<4>) is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXIE (PIE1<4>). TXIF is set regardless of the state of enable bit TXIE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREG register.

While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits SYNC SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 18-11: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG = 0, continuous transmission of two 8-bit words.

18.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

18.4.1 EUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the `SLEEP` instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREG register.
c) Flag bit TXIF will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will now be set.
e) If enable bit TXIE is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 18-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	52
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	52
IPR1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	52
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
TXREG	EUSART Transmit Register								51
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	51
SPBRGH	EUSART Baud Rate Generator Register High Byte								51
SPBRG	EUSART Baud Rate Generator Register Low Byte								51

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.
Note 1: Reserved in PIC18F2X8X devices; always maintain these bits clear.

NOTES:

REGISTER 23-14: RXB1CON: RECEIVE BUFFER 1 CONTROL REGISTER

Mode 0

R/C-0	R/W-0	R/W-0	U-0	R-0	R/W-0	R-0	R-0
RXFUL[(1)]	RXM1	RXM0	—	RXRTRRO	FILHIT2	FILHIT1	FILHIT0

Mode 1, 2

R/C-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
RXFUL[(1)]	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
bit 7							bit 0

bit 7 **RXFUL:** Receive Full Status bit[(1)]

`1` = Receive buffer contains a received message
`0` = Receive buffer is open to receive a new message

Note 1: This bit is set by the CAN module upon receiving a message and must be cleared by software after the buffer is read. As long as RXFUL is set, no new message will be loaded and buffer will be considered full.

bit 6 Mode 0:
RXM1: Receive Buffer Mode bit 1
Combines with RXM0 to form RXM<1:0> bits (see bit 5).

`11` = Receive all messages (including those with errors); filter criteria is ignored
`10` = Receive only valid messages with extended identifier; EXIDEN in RXFnSIDL must be ‘`1`’
`01` = Receive only valid messages with standard identifier, EXIDEN in RXFnSIDL must be ‘`0`’
`00` = Receive all valid messages as per EXIDEN bit in RXFnSIDL register

Mode 1, 2:
RXM1: Receive Buffer Mode bit

`1` = Receive all messages (including those with errors); acceptance filters are ignored
`0` = Receive all valid messages as per acceptance filters

bit 5 Mode 0:
RXM0: Receive Buffer Mode bit 0
Combines with RXM1 to form RXM<1:0> bits (see bit 6).

Mode 1, 2:
RTRRO: Remote Transmission Request bit for Received Message (read-only)

`1` = A remote transmission request is received
`0` = A remote transmission request is not received

bit 4 Mode 0:
Unimplemented: Read as ‘`0`’

Mode 1, 2:
FILHIT4: Filter Hit bit 4
This bit combines with other bits to form filter acceptance bits <4:0>.

bit 3 Mode 0:
RXRTRRO: Remote Transmission Request bit for Received Message (read-only)

`1` = A remote transmission request is received
`0` = A remote transmission request is not received

Mode 1, 2:
FILHIT3: Filter Hit bit 3
This bit combines with other bits to form filter acceptance bits <4:0>.

REGISTER 23-42: RXMnSIDL: RECEIVE ACCEPTANCE MASK n STANDARD IDENTIFIER MASK REGISTERS, LOW BYTE [$0 \leq n \leq 1$]

R/W-x	R/W-x	R/W-x	U-0	R/W-0	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDEN(1)	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0:** Standard Identifier Mask bits or Extended Identifier Mask bits EID20:EID18

bit 4 **Unimplemented:** Read as ‘0’

bit 3 Mode 0:
Unimplemented: Read as ‘0’
Mode 1, 2:
EXIDEN: Extended Identifier Filter Enable Mask bit(1)
1 = Messages selected by the EXIDEN bit in RXFnSIDL will be accepted
0 = Both standard and extended identifier messages will be accepted

Note 1: This bit is available in Mode 1 and 2 only.

bit 2 **Unimplemented:** Read as ‘0’

bit 1-0 **EID17:EID16:** Extended Identifier Mask 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

REGISTER 23-43: RXMnEIDH: RECEIVE ACCEPTANCE MASK n EXTENDED IDENTIFIER MASK REGISTERS, HIGH BYTE [$0 \leq n \leq 1$]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

bit 7-0 **EID15:EID8:** Extended Identifier Mask 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

REGISTER 23-44: RXMnEIDL: RECEIVE ACCEPTANCE MASK n EXTENDED IDENTIFIER MASK REGISTERS, LOW BYTE [$0 \leq n \leq 1$]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0:** Extended Identifier Mask 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

REGISTER 23-47: RXFBCONn: RECEIVE FILTER BUFFER CONTROL REGISTER n(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
RXFBCON0	F1BP_3	F1BP_2	F1BP_1	F1BP_0	F0BP_3	F0BP_2	F0BP_1	F0BP_0
	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
RXFBCON1	F3BP_3	F3BP_2	F3BP_1	F3BP_0	F2BP_3	F2BP_2	F2BP_1	F2BP_0
	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
RXFBCON2	F5BP_3	F5BP_2	F5BP_1	F5BP_0	F4BP_3	F4BP_2	F4BP_1	F4BP_0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RXFBCON3	F7BP_3	F7BP_2	F7BP_1	F7BP_0	F6BP_3	F6BP_2	F6BP_1	F6BP_0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RXFBCON4	F9BP_3	F9BP_2	F9BP_1	F9BP_0	F8BP_3	F8BP_2	F8BP_1	F8BP_0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RXFBCON5	F11BP_3	F11BP_2	F11BP_1	F11BP_0	F10BP_3	F10BP_2	F10BP_1	F10BP_0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RXFBCON6	F13BP_3	F13BP_2	F13BP_1	F13BP_0	F12BP_3	F12BP_2	F12BP_1	F12BP_0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RXFBCON7	F15BP_3	F15BP_2	F15BP_1	F15BP_0	F14BP_3	F14BP_2	F14BP_1	F14BP_0
	bit 7							bit 0

bit 7-0 **FnBP_3:FnBP_0:** Filter n Buffer Pointer Nibble bits
`0000` = Filter n is associated with RXB0
`0001` = Filter n is associated with RXB1
`0010` = Filter n is associated with B0
`0011` = Filter n is associated with B1
...
`0111` = Filter n is associated with B5
`1111-1000` = Reserved

Note 1: This register is available in Mode 1 and 2 only.

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

23.2.6 CAN INTERRUPT REGISTERS

The registers in this section are the same as described in **Section 9.0 "Interrupts"**. They are duplicated here for convenience.

REGISTER 23-56: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

	bit 7							bit 0
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
Mode 0	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF[(1)]	TXB0IF[(1)]	RXB1IF	RXB0IF
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
Mode 1, 2	IRXIF	WAKIF	ERRIF	TXBnIF	TXB1IF[(1)]	TXB0IF[(1)]	RXBnIF	FIFOWMIF

bit 7 **IRXIF:** CAN Invalid Received Message Interrupt Flag bit
1 = An invalid message has occurred on the CAN bus
0 = No invalid message on CAN bus

bit 6 **WAKIF:** CAN bus Activity Wake-up Interrupt Flag bit
1 = Activity on CAN bus has occurred
0 = No activity on CAN bus

bit 5 **ERRIF:** CAN bus Error Interrupt Flag bit
1 = An error has occurred in the CAN module (multiple sources)
0 = No CAN module errors

bit 4 When CAN is in Mode 0:
TXB2IF: CAN Transmit Buffer 2 Interrupt Flag bit
1 = Transmit Buffer 2 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 2 has not completed transmission of a message
When CAN is in Mode 1 or 2:
TXBnIF: Any Transmit Buffer Interrupt Flag bit
1 = One or more transmit buffers have completed transmission of a message and may be reloaded
0 = No transmit buffer is ready for reload

bit 3 **TXB1IF:** CAN Transmit Buffer 1 Interrupt Flag bit[(1)]
1 = Transmit Buffer 1 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 1 has not completed transmission of a message

bit 2 **TXB0IF:** CAN Transmit Buffer 0 Interrupt Flag bit[(1)]
1 = Transmit Buffer 0 has completed transmission of a message and may be reloaded
0 = Transmit Buffer 0 has not completed transmission of a message

bit 1 When CAN is in Mode 0:
RXB1IF: CAN Receive Buffer 1 Interrupt Flag bit
1 = Receive Buffer 1 has received a new message
0 = Receive Buffer 1 has not received a new message
When CAN is in Mode 1 or 2:
RXBnIF: Any Receive Buffer Interrupt Flag bit
1 = One or more receive buffers has received a new message
0 = No receive buffer has received a new message

bit 0 When CAN is in Mode 0:
RXB0IF: CAN Receive Buffer 0 Interrupt Flag bit
1 = Receive Buffer 0 has received a new message
0 = Receive Buffer 0 has not received a new message
When CAN is in Mode 1:
Unimplemented: Read as '0'
When CAN is in Mode 2:
FIFOWMIF: FIFO Watermark Interrupt Flag bit
1 = FIFO high watermark is reached
0 = FIFO high watermark is not reached

Note 1: In CAN Mode 1 and 2, this bit is forced to '0'.

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

TABLE 23-1: CAN CONTROLLER REGISTER MAP (CONTINUED)

Address[(1)]	Name	Address	Name	Address	Name	Address	Name
E7Fh	CANCON_RO4[(2)]	E5Fh	CANCON_RO6[(2)]	E3Fh	CANCON_RO8[(2)]	E1Fh	—[(4)]
E7Eh	CANSTAT_RO4[(2)]	E5Eh	CANSTAT_RO6[(2)]	E3Eh	CANSTAT_RO8[(2)]	E1Eh	—[(4)]
E7Dh	B5D7	E5Dh	B3D7	E3Dh	B1D7	E1Dh	—[(4)]
E7Ch	B5D6	E5Ch	B3D6	E3Ch	B1D6	E1Ch	—[(4)]
E7Bh	B5D5	E5Bh	B3D5	E3Bh	B1D5	E1Bh	—[(4)]
E7Ah	B5D4	E5Ah	B3D4	E3Ah	B1D4	E1Ah	—[(4)]
E79h	B5D3	E59h	B3D3	E39h	B1D3	E19h	—[(4)]
E78h	B5D2	E58h	B3D2	E38h	B1D2	E18h	—[(4)]
E77h	B5D1	E57h	B3D1	E37h	B1D1	E17h	—[(4)]
E76h	B5D0	E56h	B3D0	E36h	B1D0	E16h	—[(4)]
E75h	B5DLC	E55h	B3DLC	E35h	B1DLC	E15h	—[(4)]
E74h	B5EIDL	E54h	B3EIDL	E34h	B1EIDL	E14h	—[(4)]
E73h	B5EIDH	E53h	B3EIDH	E33h	B1EIDH	E13h	—[(4)]
E72h	B5SIDL	E52h	B3SIDL	E32h	B1SIDL	E12h	—[(4)]
E71h	B5SIDH	E51h	B3SIDH	E31h	B1SIDH	E11h	—[(4)]
E70h	B5CON	E50h	B3CON	E30h	B1CON	E10h	—[(4)]
E6Fh	CANCON_RO5	E4Fh	CANCON_RO7	E2Fh	CANCON_RO9	E0Fh	—[(4)]
E6Eh	CANSTAT_RO5	E4Eh	CANSTAT_RO7	E2Eh	CANSTAT_RO9	E0Eh	—[(4)]
E6Dh	B4D7	E4Dh	B2D7	E2Dh	B0D7	E0Dh	—[(4)]
E6Ch	B4D6	E4Ch	B2D6	E2Ch	B0D6	E0Ch	—[(4)]
E6Bh	B4D5	E4Bh	B2D5	E2Bh	B0D5	E0Bh	—[(4)]
E6Ah	B4D4	E4Ah	B2D4	E2Ah	B0D4	E0Ah	—[(4)]
E69h	B4D3	E49h	B2D3	E29h	B0D3	E09h	—[(4)]
E68h	B4D2	E48h	B2D2	E28h	B0D2	E08h	—[(4)]
E67h	B4D1	E47h	B2D1	E27h	B0D1	E07h	—[(4)]
E66h	B4D0	E46h	B2D0	E26h	B0D0	E06h	—[(4)]
E65h	B4DLC	E45h	B2DLC	E25h	B0DLC	E05h	—[(4)]
E64h	B4EIDL	E44h	B2EIDL	E24h	B0EIDL	E04h	—[(4)]
E63h	B4EIDH	E43h	B2EIDH	E23h	B0EIDH	E03h	—[(4)]
E62h	B4SIDL	E42h	B2SIDL	E22h	B0SIDL	E02h	—[(4)]
E61h	B4SIDH	E41h	B2SIDH	E21h	B0SIDH	E01h	—[(4)]
E60h	B4CON	E40h	B2CON	E20h	B0CON	E00h	—[(4)]

Note 1: Shaded registers are available in Access Bank low area, while the rest are available in Bank 15.
2: CANSTAT register is repeated in these locations to simplify application firmware. Unique names are given for each instance of the controller register due to the Microchip header file requirement.
3: These registers are not CAN registers.
4: Unimplemented registers are read as ‘0’.

Table 23-3 shows the relation between the clock generated by the PLL and the frequency error from jitter (measured jitter-induced error of 2%, Gaussian distribution, within 3 standard deviations), as a percentage of the nominal clock frequency.

This is clearly smaller than the expected drift of a crystal oscillator, typically specified at 100 ppm or 0.01%. If we add jitter to oscillator drift, we have a total frequency drift of 0.0132%. The total oscillator frequency errors for common clock frequencies and bit rates, including both drift and jitter, are shown in Table 23-4.

TABLE 23-3: FREQUENCY ERROR FROM JITTER AT VARIOUS PLL GENERATED CLOCK SPEEDS

PLL Output	P_{jitter}	T_{jitter}	Frequency Error at Various Nominal Bit Times (Bit Rates)			
			8 µs (125 Kb/s)	4 µs (250 Kb/s)	2 µs (500 Kb/s)	1 µs (1 Mb/s)
40 MHz	0.5 ns	1 ns	0.00125%	0.00250%	0.005%	0.01%
24 MHz	0.83 ns	1.67 ns	0.00209%	0.00418%	0.008%	0.017%
16 MHz	1.25 ns	2.5 ns	0.00313%	0.00625%	0.013%	0.025%

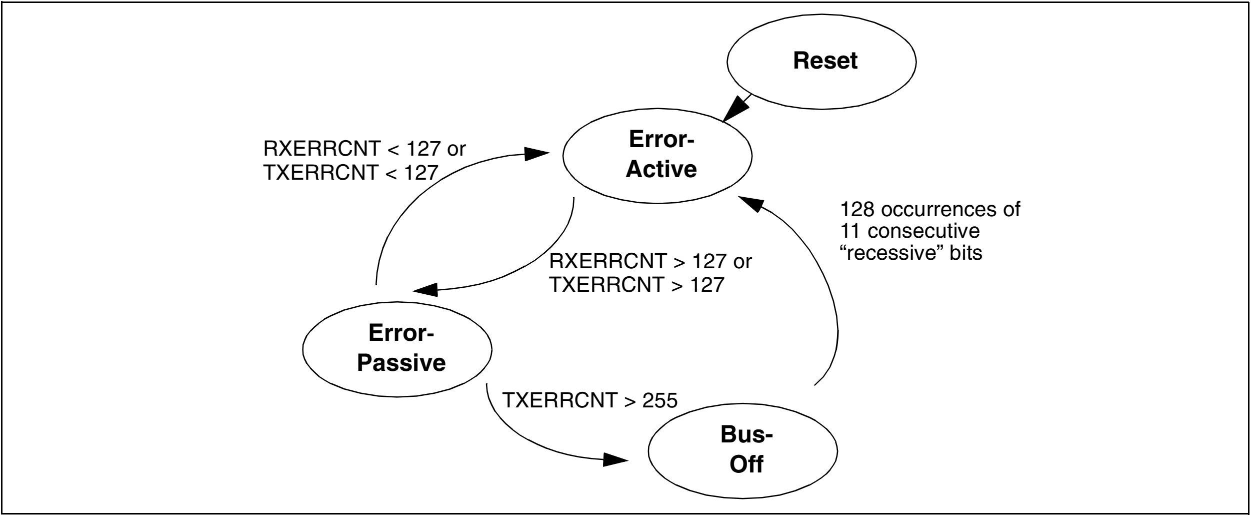
TABLE 23-4: TOTAL FREQUENCY ERROR AT VARIOUS PLL GENERATED CLOCK SPEEDS (100 PPM OSCILLATOR DRIFT, INCLUDING ERROR FROM JITTER)

Nominal PLL Output	Frequency Error at Various Nominal Bit Times (Bit Rates)			
	8 µs (125 Kb/s)	4 µs (250 Kb/s)	2 µs (500 Kb/s)	1 µs (1 Mb/s)
40 MHz	0.01125%	0.01250%	0.015%	0.02%
24 MHz	0.01209%	0.01418%	0.018%	0.027%
16 MHz	0.01313%	0.01625%	0.023%	0.035%

The PIC18F2585/2680/4585/4680 devices are error-active if both error counters are below the error-passive limit of 128. They are error-passive if at least one of the error counters equals or exceeds 128. They go to bus-off if the transmit error counter equals or exceeds the bus-off limit of 256. The devices remain in this state until the bus-off recovery sequence is received. The bus-off recovery sequence consists of 128 occurrences of 11 consecutive recessive bits (see Figure 23-8). Note that the CAN module, after going bus-off, will recover back to error-active without any intervention by the MCU if the bus remains Idle for 128 x 11 bit times. If this is not desired, the error Interrupt Service Routine should address this. The current Error mode of the CAN module can be read by the MCU via the COMSTAT register.

Additionally, there is an Error State Warning flag bit, EWARN, which is set if at least one of the error counters equals or exceeds the error warning limit of 96. EWARN is reset if both error counters are less than the error warning limit.

FIGURE 23-8: ERROR MODES STATE DIAGRAM



23.15 CAN Interrupts

The module has several sources of interrupts. Each of these interrupts can be individually enabled or disabled. The PIR3 register contains interrupt flags. The PIE3 register contains the enables for the 8 main interrupts. A special set of read-only bits in the CANSTAT register, the ICODE bits, can be used in combination with a jump table for efficient handling of interrupts.

All interrupts have one source, with the exception of the error interrupt and buffer interrupts in Mode 1 and 2. Any of the error interrupt sources can set the error interrupt flag. The source of the error interrupt can be determined by reading the Communication Status register, COMSTAT. In Mode 1 and 2, there are two interrupt enable/disable and flag bits – one for all transmit buffers and the other for all receive buffers.

The interrupts can be broken up into two categories: receive and transmit interrupts.

The receive related interrupts are:

- Receive Interrupts
- Wake-up Interrupt
- Receiver Overrun Interrupt
- Receiver Warning Interrupt
- Receiver Error-Passive Interrupt

The transmit related interrupts are:

- Transmit Interrupts
- Transmitter Warning Interrupt
- Transmitter Error-Passive Interrupt
- Bus-Off Interrupt

REGISTER 24-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

bit 7	**IESO:** Internal/External Oscillator Switchover bit `1` = Oscillator Switchover mode enabled `0` = Oscillator Switchover mode disabled
bit 6	**FCMEN:** Fail-Safe Clock Monitor Enable bit `1` = Fail-Safe Clock Monitor enabled `0` = Fail-Safe Clock Monitor disabled
bit 5-4	**Unimplemented:** Read as '`0`'
bit 3-0	**FOSC3:FOSC0:** Oscillator Selection bits `11xx` = External RC oscillator, CLKO function on RA6 `101x` = External RC oscillator, CLKO function on RA6 `1001` = Internal oscillator block, CLKO function on RA6, port function on RA7 `1000` = Internal oscillator block, port function on RA6 and RA7 `0111` = External RC oscillator, port function on RA6 `0110` = HS oscillator, PLL enabled (Clock Frequency = 4 x FOSC1) `0101` = EC oscillator, port function on RA6 `0100` = EC oscillator, CLKO function on RA6 `0011` = External RC oscillator, CLKO function on RA6 `0010` = HS oscillator `0001` = XT oscillator `0000` = LP oscillator

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		u = Unchanged from programmed state

TABLE 25-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
LITERAL OPERATIONS									
ADDLW	k	Add literal and WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	AND literal with WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	Inclusive OR literal with WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	Move literal (12-bit) 2nd word	2	1110	1110	00ff	kkkk	None	
		to FSR(f) 1st word		1111	0000	kkkk	kkkk		
MOVLB	k	Move literal to BSR<3:0>	1	0000	0001	0000	kkkk	None	
MOVLW	k	Move literal to WREG	1	0000	1110	kkkk	kkkk	None	
MULLW	k	Multiply literal with WREG	1	0000	1101	kkkk	kkkk	None	
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	
SUBLW	k	Subtract WREG from literal	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR literal with WREG	1	0000	1010	kkkk	kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS									
TBLRD*		Table Read	2	0000	0000	0000	1000	None	
TBLRD*+		Table Read with post-increment		0000	0000	0000	1001	None	
TBLRD*-		Table Read with post-decrement		0000	0000	0000	1010	None	
TBLRD+*		Table Read with pre-increment		0000	0000	0000	1011	None	
TBLWT*		Table Write	2	0000	0000	0000	1100	None	5
TBLWT*+		Table Write with post-increment		0000	0000	0000	1101	None	5
TBLWT*-		Table Write with post-decrement		0000	0000	0000	1110	None	5
TBLWT+*		Table Write with pre-increment		0000	0000	0000	1111	None	5

Note 1: When a Port register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is ‘`1`’ for a pin configured as input and is driven low by an external device, the data will be written back with a ‘`0`’.

2: If this instruction is executed on the TMR0 register (and where applicable, ‘d’ = `1`), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

TSTFSZ — Test f, Skip if 0

Syntax:	TSTFSZ f {,a}
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	skip if f = `0`
Status Affected:	None

Encoding:

0110	011a	ffff	ffff

Description: If ‘f’ = `0`, the next instruction fetched during the current instruction execution is discarded and a `NOP` is executed, making this a two-cycle instruction.

If ‘a’ is ‘`0`’, the Access Bank is selected. If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank (default).

If ‘a’ is ‘`0`’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 25.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1(2)

Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE     TSTFSZ  CNT, 1
NZERO    :
ZERO     :
```

Before Instruction
```
    PC       =   Address (HERE)
```
After Instruction
```
    If CNT   =   00h,
    PC       =   Address (ZERO)
    If CNT   ≠   00h,
    PC       =   Address (NZERO)
```

XORLW — Exclusive OR Literal with W

Syntax:	XORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .XOR. k $\rightarrow$ W
Status Affected:	N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal ‘k’. The result is placed in W.

Words: 1

Cycles: 1

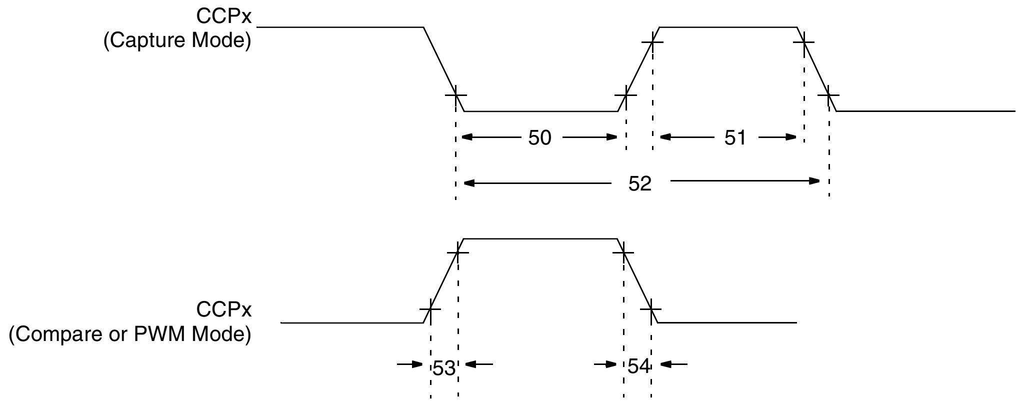
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: `XORLW   0AFh`

Before Instruction
```
    W    =   B5h
```
After Instruction
```
    W    =   1Ah
```

FIGURE 27-10: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)



Note: Refer to Figure 27-4 for load conditions.

TABLE 27-12: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)

Param No.	Sym	Characteristic			Min	Max	Units	Conditions
50	TCCL	CCPx Input Low Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	20	—	ns	VDD = 2.0V
51	TCCH	CCPx Input High Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	20	—	ns	VDD = 2.0V
52	TCCP	CCPx Input Period			$\frac{3\,T_{CY} + 40}{N}$	—	ns	N = prescale value (1,4 or 16)
53	TCCR	CCPx Output Fall Time		PIC18**F**XXXX	—	25	ns	
				PIC18**LF**XXXX	—	45	ns	VDD = 2.0V
54	TCCF	CCPx Output Fall Time		PIC18**F**XXXX	—	25	ns	
				PIC18**LF**XXXX	—	45	ns	VDD = 2.0V