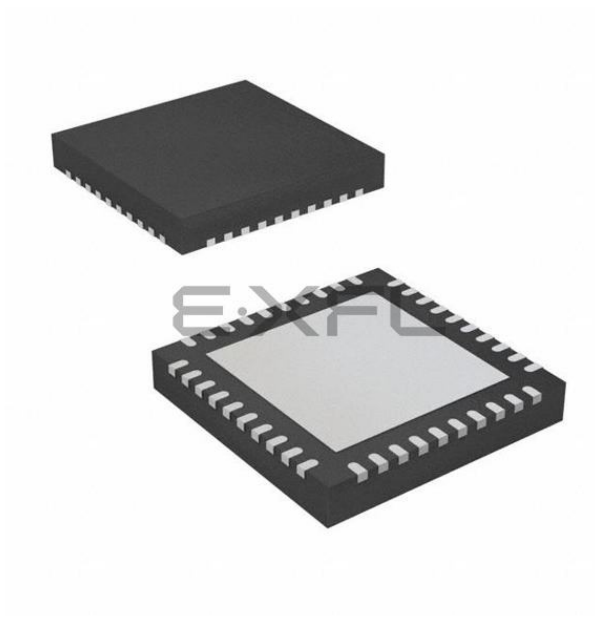


Welcome to E-XFL.COM

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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	40-UFQFN Exposed Pad
Supplier Device Package	40-UQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf46k42-e-mv

5.7.4 FIXED VOLTAGE REFERENCE DATA

The DIA stores measured FVR voltages for this device in mV for the different buffer settings of 1x, 2x or 4x at Program Memory locations 3F0030h to 3F003Bh. For more information on the FVR, refer to **Section 34.0 "Fixed Voltage Reference (FVR)"**.

- FVRA1X stores the value of ADC FVR1 Output voltage for 1x setting (in mV)
- FVRA2X stores the value of ADC FVR1 Output Voltage for 2x setting (in mV)
- FVRA4X stores the value of ADC FVR1 Output Voltage for 4x setting (in mV)
- FVRC1X stores the value of Comparator FVR2 output voltage for 2x setting (in mV)
- FVRC2X stores the value of Comparator FVR2 output voltage for 2x setting (in mV)
- FVRC4X stores the value of Comparator FVR2 output voltage for 4x setting (in mV)

5.8 Device Configuration Information

The Device Configuration Information (DCI) is a dedicated region in the Program memory space mapped from 3FFF00h to 3FFF09h. The data stored in these locations is read-only and cannot be erased.

Refer to Table 5-4: Device Configuration Information for PIC18(L)F26/27/45/55/46/47/56/57K42 for the complete DCI table address and description. The DCI holds information about the device which is useful for programming and Bootloader applications.

The erase size is the minimum erasable unit in the PFM, expressed as rows. The total device Flash memory capacity is (Row Size * Number of rows)

TABLE 5-4: DEVICE CONFIGURATION INFORMATION FOR PIC18(L)F26/27/45/55/46/47/56/57K42

ADDRESS	Name	DESCRIPTION	VALUE			UNITS
			PIC18(L)F45/55K42	PIC18(L)F26/46/56K42	PIC18(L)F27/47/57K42	
3F FF00h-3F FF01h	ERSIZ	Erase Row Size	64	64	64	Words
3F FF02h-3F FF03h	WLSIZ	Number of write latches per row	128	128	128	Bytes
3F FF04h-3F FF05h	URSIZ	Number of User Rows	256	512	1024	Rows
3F FF06h-3F FF07h	EESIZ	Data EEPROM memory size	256	1024	1024	Bytes
3F FF08h-3F FF09h	PCNT	Pin Count	40(1)/48	28/40(1)/48	28/40(1)/48	Pins

Note 1: Pin count of 40 is also used for 44-pin part.

7.2.2.3 Internal Oscillator Frequency Adjustment

The internal oscillator is factory-calibrated. This internal oscillator can be adjusted in software by writing to the OSCTUNE register (Register 7-3).

The default value of the OSCTUNE register is 00h. The value is a 6-bit two's complement number. A value of 1Fh will provide an adjustment to the maximum frequency. A value of 20h will provide an adjustment to the minimum frequency.

When the OSCTUNE register is modified, the oscillator frequency will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred.

OSCTUNE **does not affect** the LFINTOSC frequency. Operation of features that depend on the LFINTOSC clock source frequency, such as the Power-up Timer (PWRT), WWDT, Fail-Safe Clock Monitor (FSCM) and peripherals, are *not* affected by the change in frequency.

7.2.2.4 LFINTOSC

The Low-Frequency Internal Oscillator (LFINTOSC) is a factory-calibrated 31 kHz internal clock source.

The LFINTOSC is the frequency for the Power-up Timer (PWRT), Windowed Watchdog Timer (WWDT) and Fail-Safe Clock Monitor (FSCM).

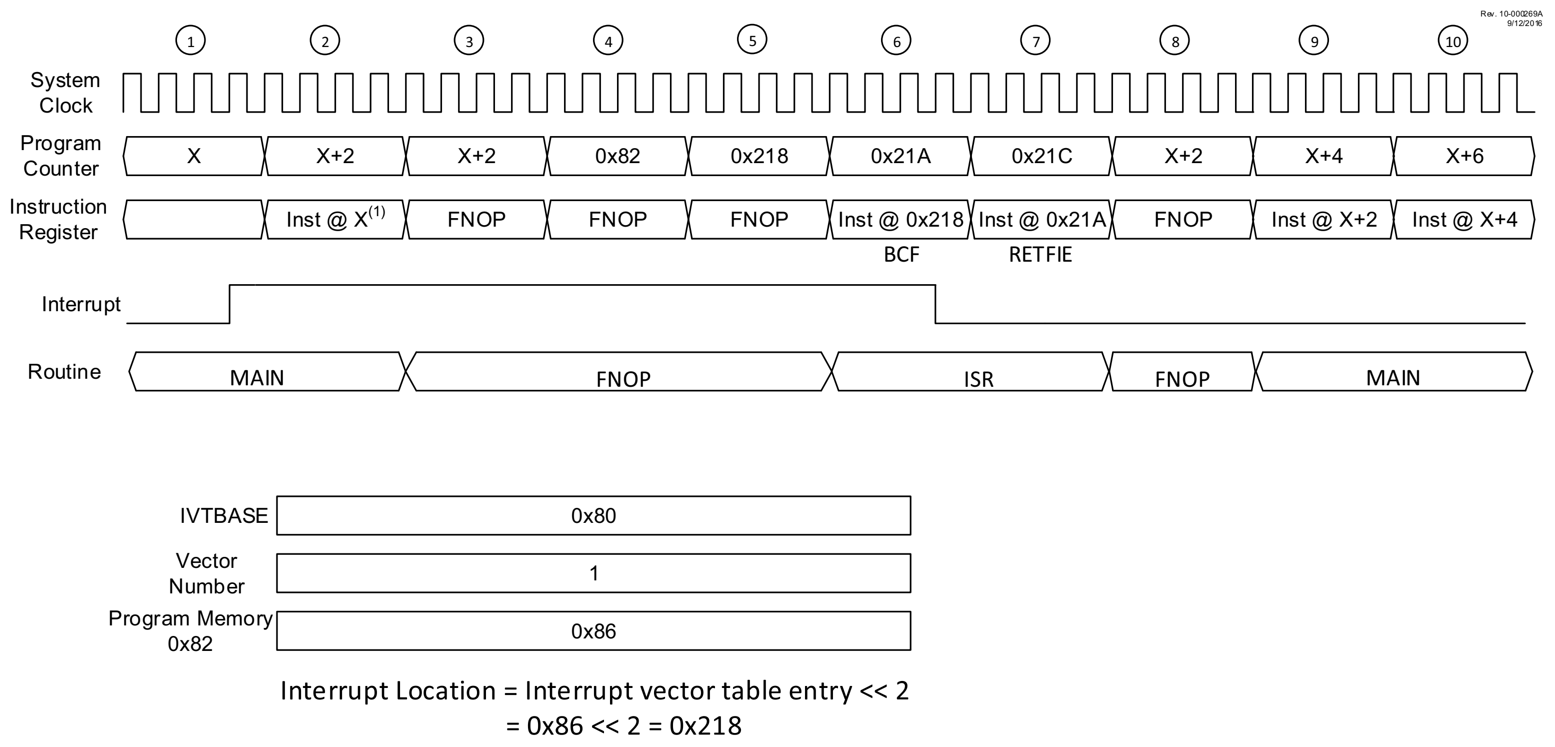
The LFINTOSC is enabled through one of the following methods:

- Programming the RSTOSC<2:0> bits of Configuration Word 1 to enable LFINTOSC.
- Write to the NOSC<2:0> bits of the OSCCON1 register during run-time. See **Section 7.3, Clock Switching** for more information.

7.2.2.5 ADCRC

The ADCRC is an oscillator dedicated to the ADC^2 module. The ADCRC oscillator can be manually enabled using the ADOEN bit of the OSCEN register. The ADCRC runs at a fixed frequency of 600 kHz. ADCRC is automatically enabled if it is selected as the clock source for the ADC^2 module.

FIGURE 9-7: INTERRUPT TIMING DIAGRAM - ONE CYCLE INSTRUCTION



Note 1: Instruction @ X is a One-cycle Instruction

REGISTER 9-5: PIR2: PERIPHERAL INTERRUPT REGISTER 2[(1)]

R-0/0	R-0/0	R-0/0	R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
I2C1RXIF[(2)]	SPI1IF[(3)]	SPI1TXIF[(4)]	SPI1RXIF[(4)]	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HS = Hardware set

bit 7 **I2C1RXIF:** I^{2}C1 Receive Interrupt Flag bit[(2)]
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 6 **SPI1IF:** SPI1 Interrupt Flag bit[(3)]
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 5 **SPI1TXIF:** SPI1 Transmit Interrupt Flag bit[(4)]
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 4 **SPI1RXIF:** SPI1 Receive Interrupt Flag bit[(4)]
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 3 **DMA1AIF:** DMA1 Abort Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 2 **DMA1ORIF:** DMA1 Overrun Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 1 **DMA1DCNTIF:** DMA1 Destination Count Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 0 **DMA1SCNTIF:** DMA1 Source Count Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

Note 1: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

2: I2CxTXIF and I2CxRXIF are read-only bits. To clear the interrupt condition, the CLRBF bit in I2CxSTAT1 register must be set.

3: SPIxIF is a read-only bit. To clear the interrupt condition, all bits in the SPIxINTF register must be cleared.

4: SPIxTXIF and SPIxRXIF are read-only bits and cannot be set/cleared by the software.

REGISTER 19-8: PMD7: PMD CONTROL REGISTER 7

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	DMA2MD	DMA1MD
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	q = Value depends on condition

bit 7-2	**Unimplemented:** Read as ‘0’
bit 1	**DMA2MD:** Disable DMA2 Module bit 1 = DMA2 module disabled 0 = DMA2 module enabled
bit 0	**DMA1MD:** Disable DMA1 Module bit 1 = DMA1 module disabled 0 = DMA1 module enabled

TABLE 19-1: SUMMARY OF REGISTERS ASSOCIATED WITH PERIPHERAL MODULE DISABLE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	290
PMD1	NCO1MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	291
PMD2	—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	292
PMD3	PWM8MD	PWM7MD	PWM6MD	PWM5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	293
PMD4	CWG3MD	CWG2MD	CWG1MD	—	—	—	—	—	294
PMD5	—	—	U2MD	U1MD	—	SPI1MD	I2C2MD	I2C1MD	295
PMD6	—	—	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD	295
PMD7	—	—	—	—	—	—	DMA2MD	DMA1MD	297

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by peripheral module disable.

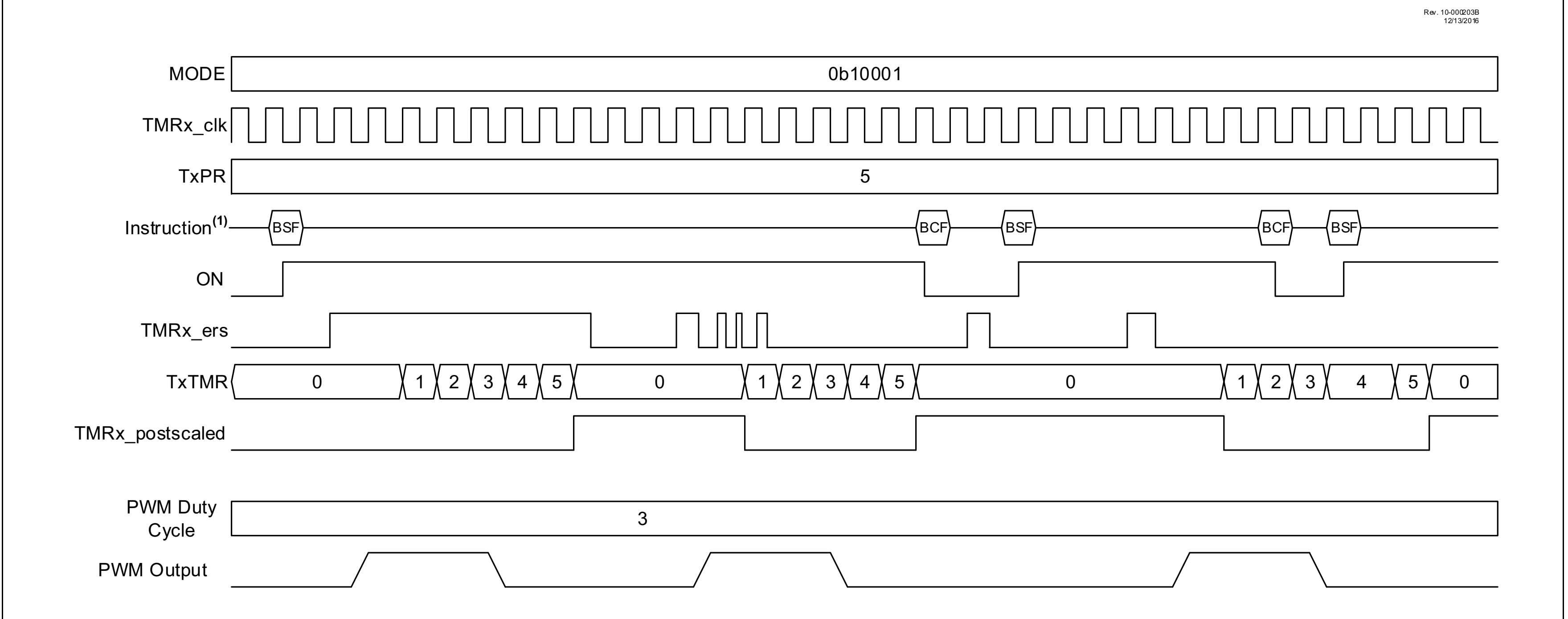
22.5.9 EDGE-TRIGGERED MONOSTABLE MODES

The Edge-Triggered Monostable modes start the timer on an edge from the external Reset signal input, after the ON bit is set, and stop incrementing the timer when the timer matches the T2PR period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = `10001`)
- Falling edge (MODE<4:0> = `10010`)
- Rising or Falling edge (MODE<4:0> = `10011`)

When an Edge-Triggered Monostable mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the T2PR value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

FIGURE 22-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = `10001`)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

FIGURE 25-9: HIGH AND LOW MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAM

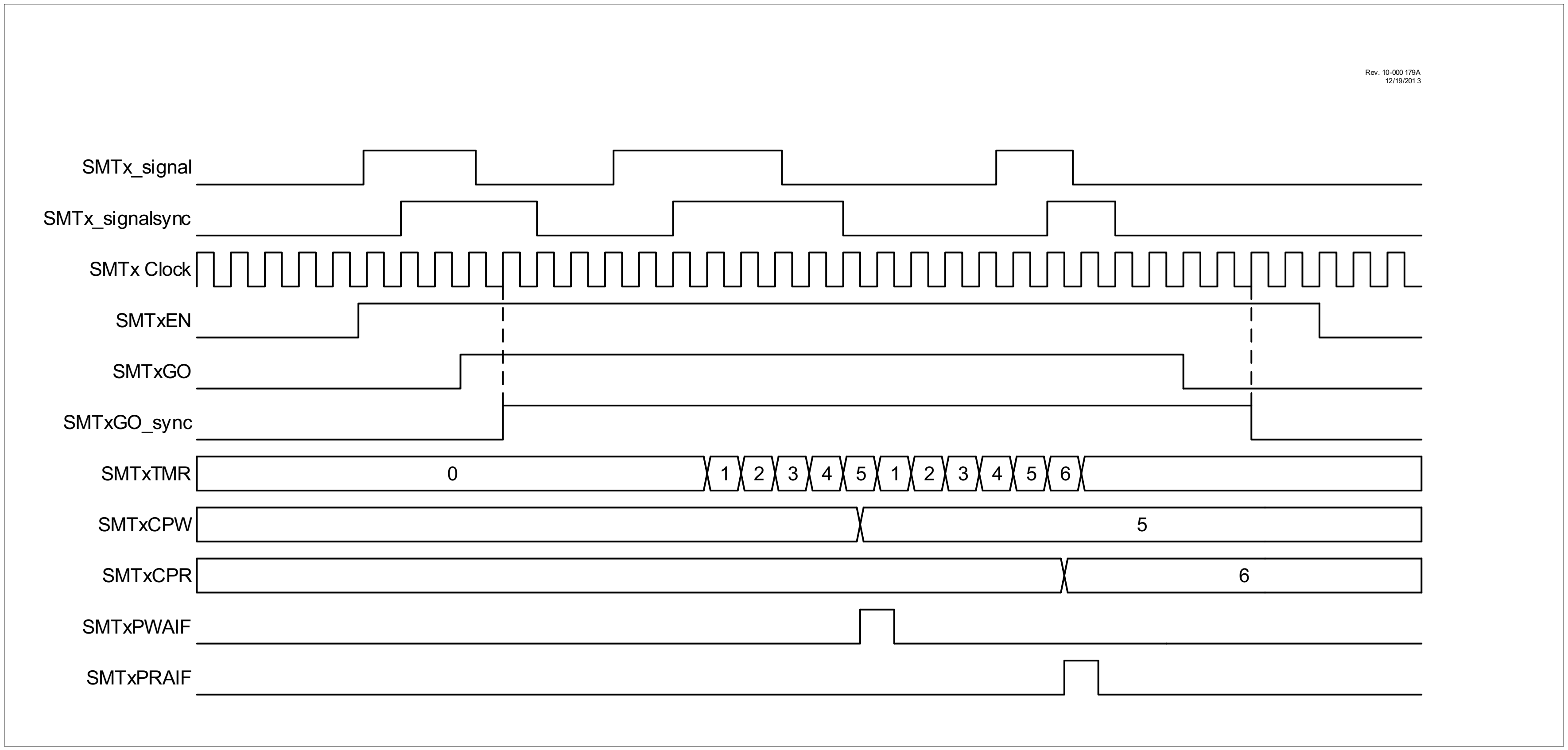


TABLE 30-2: MD1SRC SELECTION MUX CONNECTIONS

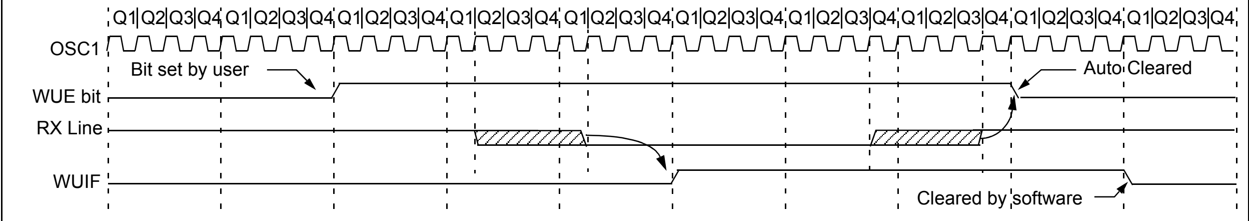
MS<4:0>		Connection
`1 1111` – `1 0111`	31-23	Reserved
`1 0110`	22	SPI1 SDO
`1 0101`	21	Reserved
`1 0100`	20	UART2 TX
`1 0011`	19	UART1 TX
`1 0010`	18	CLC4 OUT
`1 0001`	17	CLC3 OUT
`1 0000`	16	CLC2 OUT
`0 1111`	15	CLC1 OUT
`0 1110`	14	CMP2 OUT
`0 1101`	13	CMP1 OUT
`0 1100`	12	NCO1 OUT
`0 1011`	11	Reserved
`0 1010`	10	Reserved
`0 1001`	9	PWM8 OUT
`0 1000`	8	PWM7 OUT
`0 0111`	7	PWM6 OUT
`0 0110`	6	PWM5 OUT
`0 0101`	5	CCP4 OUT
`0 0100`	4	CCP3 OUT
`0 0011`	3	CCP2 OUT
`0 0010`	2	CCP1 OUT
`0 0001`	1	DSM1 BIT
`0 0000`	0	Pin selected by MDSRCPPS

TABLE 30-3: SUMMARY OF REGISTERS ASSOCIATED WITH DATA SIGNAL MODULATOR MODE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
MD1CON0	EN	—	OUT	OPOL	—	—	—	BIT	469
MD1CON1	—	—	CHPOL	CHSYNC	—	—	CLPOL	CLSYNC	470
MD1CARH	—	—	—	—	—	CHS<2:0>			471
MD1CARL	—	—	—	—	—	CLS<2:0>			471
MDSRC	—	—	—	—	SRCS<3:0>				472

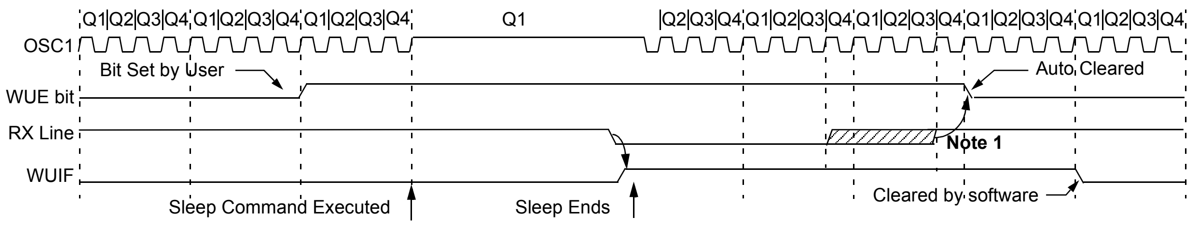
Legend: — = unimplemented, read as ‘0’. Shaded cells are not used in the Data Signal Modulator mode.

FIGURE 31-13: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION



Note 1: The **UART** remains in idle while the WUE bit is set.

FIGURE 31-14: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the automatic clearing of the WUE bit can occur while the `stposc` signal is still active. This sequence should not depend on the presence of Q clocks.

2: The **UART** remains in idle while the WUE bit is set.

31.18 Transmitting a Break

The UART module has the capability of sending either a fixed length Break period or a software timed Break period. The fixed length Break consists of a Start bit, followed by 12 ‘0’ bits and a Stop bit. The software timed Break is generated by setting and clearing the BRKOVR bit in the UxCON1 register.

To send the fixed length Break, set the SENDB and TXEN bits in the UxCON0 register. The Break sequence is then initiated by a write to UxTXB. The timed Break will occur first, followed by the character written to UxTXB that initiated the Break. The initiating character is typically the Sync character of the LIN specification.

SENB is disabled in the LIN and DMX modes because those modes generate the Break sequence automatically.

The SENDB bit is automatically reset by hardware after the Break Stop bit is complete.

The TXMTIF bit in the UxERRIR register indicates when the transmit operation is active or idle, just as it does during normal transmission. See Figure 31-15 for the timing of the Break sequence.

31.19 Receiving a Break

The UART has counters to detect when the RX input remains in the space state for an extended period of time. When this happens, the RXBKIF bit in the UxERRIR register is set.

A Break is detected when the RX input remains in the space state for 11 bit periods for asynchronous and LIN modes, and 23 bit periods for DMX mode.

The user can select to receive the Break interrupt as soon as the Break is detected or at the end of the Break, when the RX input returns to the Idle state. When the RXBIMD bit in the UxCON1 is ‘1’ then RXBKIF is set immediately upon Break detection. When RXBIMD is ‘0’ then RXBKIF is set when the RX input returns to the Idle state.

31.20 UART Operation During Sleep

The UART ceases to operate during Sleep. The safe way to wake the device from Sleep by a serial operation is to use the Wake-on-Break feature of the UART. See Section 31.17.3, Auto-Wake-up on Break

TABLE 31-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE UART

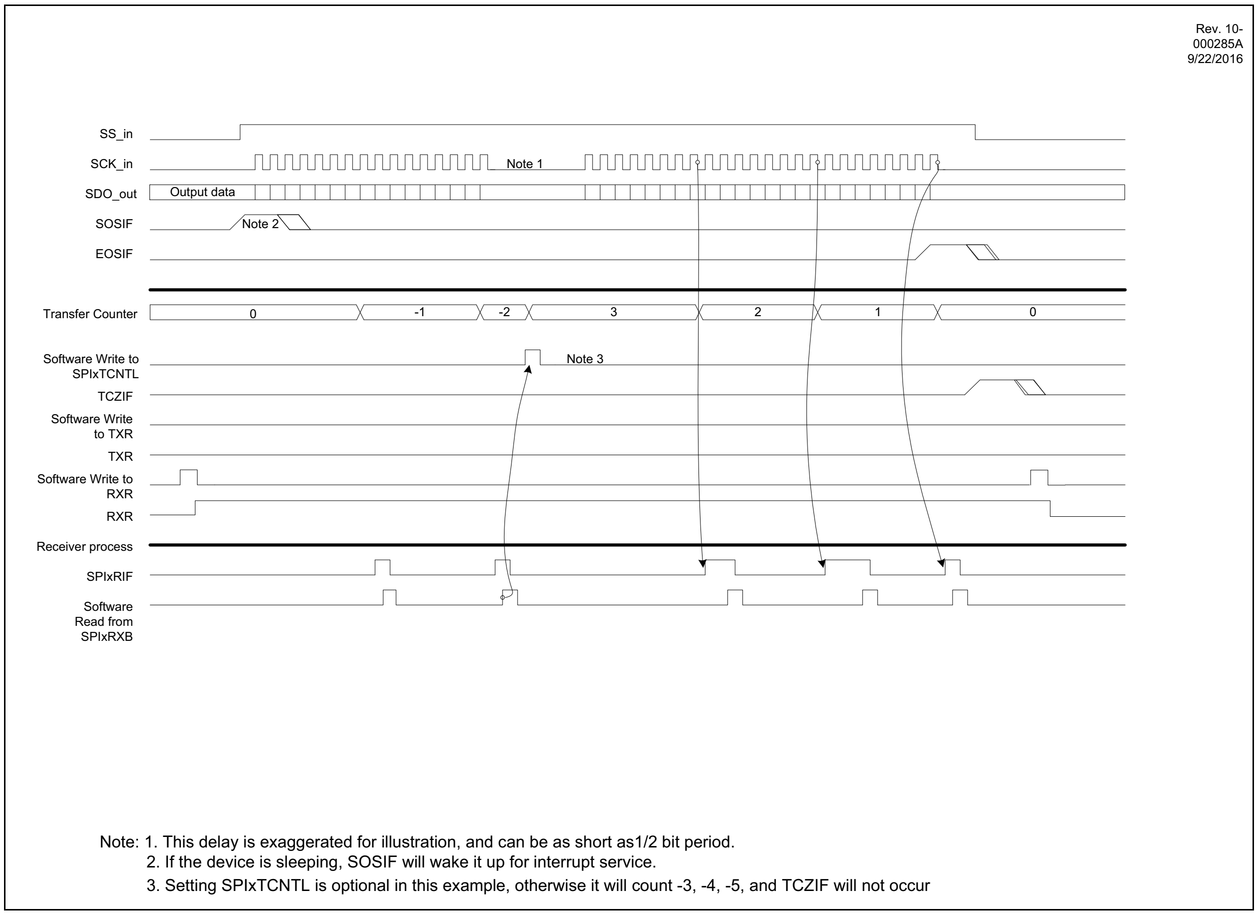
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
UxCON0	BRGS	ABDEN	TXEN	RXEN	MODE<3:0>				498
UxCON1	ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB	499
UxCON2	RUNOVF	RXPOL	STP<1:0>		C0EN	TXPOL	FLO<1:0>		500
UxERRIR	TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF	501
UxERRIE	TXMTIE	PERIE	ABDOVE	CERIE	FERIE;	RXBKIE	RXFOIE	TXCIE	502
UxUIR	WUIF	ABDIF	—	—	—	ABDIE	—	—	503
UxFIFO	TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF	504
UxBRGL	BRG<7:0>								505
UxBRGH	BRG<15:8>								505
UxRXB	RXB<7:0>								506
UxTXB	TXB<7:0>								506
UxP1H	—	—	—	—	—	—	—	P1<8>	507
UxP1L	P1<7:0>								507
UxP2H	—	—	—	—	—	—	—	P2<8>	508
UxP2L	P2<7:0>								508
UxP3H	—	—	—	—	—	—	—	P3<8>	509
UxP3L	P3<7:0>								509
UxTXCHK	TXCHK<7:0>								510
UxRXCHK	RXCHK<7:0>								510

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the UART module.

32.6.2 SLAVE MODE RECEIVE OPTIONS

The RXR bit controls the nature of receptions in slave mode. When RXR is set, the SDI input data will be stored in the RXFIFO if it is not full. If the RXFIFO is full, the RXOIF bit will be set to indicate an RXFIFO overflow error and the data is discarded. When RXR is cleared, all received data will be ignored and not stored in the RXFIFO (although it may still be used for transmission if TXFIFO is empty). Figure 32-11 shows a typical slave mode communication, showing a case where the master writes two then three bytes, showing interrupts as well as the behavior of the transfer counter in slave mode (see **Section 32.4.3 "Transfer Counter in Slave mode"** for more details on t**Section 32.8 "SPI Interrupts"**he transfer counter in slave mode as well as Section X.8 for more information on interrupts).

FIGURE 32-11: SPI SLAVE MODE OPERATION – INTERRUPT-DRIVEN, MASTER WRITES 2+3 BYTES



Note: 1. This delay is exaggerated for illustration, and can be as short as1/2 bit period.
2. If the device is sleeping, SOSIF will wake it up for interrupt service.
3. Setting SPIxTCNTL is optional in this example, otherwise it will count -3, -4, -5, and TCZIF will not occur

REGISTER 32-10: SPIxSTATUS: SPI STATUS REGISTER

R/C/HS-0/0	U-0	R-1/1	U-0	R/C/HS-0/0	S-0/0	U-0	R-0/0
TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
		C = Clearable bit
		S = Settable bit
		HS = Bit can be set by hardware

bit 7	**TXWE**: Transmit Buffer Write Error bit 1 = SPIxTxB was written while TxFIFO was full 0 = No error has occurred
bit 6	**Unimplemented**: Read as ‘0’
bit 5	**TXBE**: Transmit Buffer Empty bit (read-only) 1 = Transmit buffer TxFIFO is empty 0 = Transmit buffer is not empty
bit 4	**Unimplemented**: Read as ‘0’
bit 3	**RXRE**: Receive Buffer Read Error bit 1 = SPIxRB was read while RxFIFO was empty 0 = No error has occurred
bit 2	**CLRBF**: Clear Buffer Control bit (write only) 1 = Reset the receive and transmit buffers, making both buffers empty 0 = Take no action
bit 1	**Unimplemented**: Read as ‘0’
bit 0	**RXBF**: Receive Buffer Full bit (read-only) 1 = Receive buffer is full 0 = Receive buffer is not full

REGISTER 32-11: SPIxRxB: SPI READ BUFFER REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’

bit 7-0	**RXB<7:0>**: Receiver Buffer bits (read-only) If RX buffer is not empty: Contains the top-most byte of RXFIFO, and reading this register will remove the top-most byte RXFIFO and decrease the occupancy of the RXFIFO If RX buffer is empty: Reading this register will read as ‘0’, leave the occupancy unchanged, and set the RXRE bit of SPIxSTATUS

33.5.10 MASTER RECEPTION IN 7-BIT ADDRESSING MODE

This section describes the sequence of events for the I^2C module configured as an I^2C master in 7-bit Addressing mode and is receiving data. Figure 33-20 is used as a visual reference for this description.

1. Master software loads slave address in I2CxADB1 with R/W bit = d and number of bytes to be received in one sequence in I2CxCNT register.
2. Master hardware waits for BFRE bit to be set; then shifts out start and address with R/W = `1`.
3. Master sends out the 9th SCL pulse for ACK, master hardware clocks in ACK from Slave
4. If ABD = `0`; i.e., Address buffers are enabled

If NACK, master hardware sends Stop or sets MDR (if RSEN = `1`) and waits for user software to write to S bit for restart.

If ABD = `1`; i.e., Address buffers are disabled

If NACK, master hardware sends Stop or sets MDR (if RSEN = `1`) and waits for user software to load the new address into I2CxTXB. Software writes to the S bit are ignored in this case.

5. If ACK, master hardware receives 7-bits of data into the shift register.
6. If the receive buffer is full (i.e., RXBF = `1`), clock is stretched on 7th falling SCL edge.
7. Master software must read previous data out of I2CxRXB to clear RXBF.
8. Master hardware receives 8th bit of data into the shift register and loads it into I2CxRXB, sets I2CxRXIF and RXBF bits. I2CxCNT is decremented.
9. If I2CxCNT! = `0`, master hardware clocks out ACKDT as ACK value to slave. If I2CxCNT = `0`, master hardware clocks out ACKCNT as ACK value to slave. It is up to the user to set the values of ACKDT and ACKCNT correctly. If the user does not set ACKCNT to '`1`', the master hardware will never send a NACK when I2CxCNT becomes zero. Since a NACK was not seen on the bus, the master hardware will also not assert a Stop condition.
10. Go to step 4.

36.2.5 AUTO-CONVERSION TRIGGER

The auto-conversion trigger allows periodic ADC measurements without software intervention. When a rising edge of the selected source occurs, the GO bit is set by hardware.

The auto-conversion trigger source is selected by the ADACT register.

Using the auto-conversion trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met. See Register 36-33 for auto-conversion sources.

36.2.6 ADC CONVERSION PROCEDURE (BASIC MODE)

This is an example procedure for using the ADC to perform an analog-to-digital conversion:

1. Configure Port:
   - Disable pin output driver (Refer to the TRISx register)
   - Configure pin as analog (Refer to the ANSELx register)
2. Configure the ADC module:
   - Select ADC conversion clock
   - Select voltage reference
   - Select ADC input channel
   - Precharge and acquisition
   - Turn on ADC module
3. Configure ADC interrupt (optional):
   - Clear ADC interrupt flag
   - Enable ADC interrupt
   - Enable global interrupt[1]
4. If ADACQ = 0, software must wait the required acquisition time[2].
5. Start conversion by setting the GO bit.
6. Wait for ADC conversion to complete by one of the following:
   - Polling the GO bit
   - Polling the ADIF bit
   - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

2: Refer to **Section 36.3 "ADC Acquisition Requirements"**.

EXAMPLE 36-1: ADC CONVERSION

```
/*This code block configures the ADC
for polling, VDD and VSS references, FRC
oscillator and AN0 input.
Conversion start & polling for completion
are included.
 */
void main() {
    //System Initialize
    initializeSystem();

    //Setup ADC
    ADCON0bits.FM = 1; //right justify
    ADCON0bits.CS = 1; //FRC Clock
    ADPCH = 0x00; //RA0 is Analog channel
    TRISAbits.TRISA0 = 1; //Set RA0 to input
    ANSELAbits.ANSELA0 = 1; //Set RA0 to analog
    ADCON0bits.ON = 1; //Turn ADC On

    while (1) {
        ADCON0bits.GO = 1; //Start conversion
        while (ADCON0bits.GO); //Wait for conversion done
        resultHigh = ADRESH; //Read result
        resultLow = ADRESL; //Read result
    }
}
```

REGISTER 36-24: ADACCU: ADC ACCUMULATOR REGISTER UPPER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-x/x	R/W-x/x
—	—	—	—	—	—	ACC<17:16>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-2	**Unimplemented**: Read as ‘0’
bit 1-0	**ACC<17:16>**: ADC Accumulator MSB. Upper two bits of accumulator value. See Table 36-2 for more details.

REGISTER 36-25: ADACCH: ADC ACCUMULATOR REGISTER HIGH

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ACC<15:8>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**ACC<15:8>**: ADC Accumulator middle bits. Middle eight bits of accumulator value. See Table 36-2 for more details.

REGISTER 36-26: ADACCL: ADC ACCUMULATOR REGISTER LOW

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ACC<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**ACC<7:0>**: ADC Accumulator LSB. Lower eight bits of accumulator value. See Table 36-2 for more details.

REGISTER 36-29: ADERRH: ADC SETPOINT ERROR REGISTER HIGH

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ERR<15:8>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **ERR<15:8>**: ADC Setpoint Error MSB. Upper byte of ADC Setpoint Error. Setpoint Error calculation is determined by CALC bits of ADCON3, see Register 36-4 for more details.

REGISTER 36-30: ADERRL: ADC SETPOINT ERROR LOW BYTE REGISTER

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ERR<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **ERR<7:0>**: ADC Setpoint Error LSB. Lower byte of ADC Setpoint Error calculation is determined by CALC bits of ADCON3, see Register 36-4 for more details.

REGISTER 36-31: ADLTHH: ADC LOWER THRESHOLD HIGH BYTE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LTH<15:8>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **LTH<15:8>**: ADC Lower Threshold MSB. LTH and UTH are compared with ERR to set the ADUTHR and ADLTHR bits of ADSTAT. Depending on the setting of ADTMD, an interrupt may be triggered by the results of this comparison.

RETURN — Return from Subroutine

Syntax:	RETURN {s}
Operands:	s ∈ [0,1]
Operation:	(TOS) → PC, if s = `1` (WS) → W, (STATUSS) → Status, (BSRS) → BSR, PCLATU, PCLATH are unchanged
Status Affected:	None
Encoding:	`0000` `0000` `0001` `001s`
Description:	Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If 's'= `1`, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, Status and BSR. If 's' = `0`, no update of these registers occurs (default).
Words:	1
Cycles:	2

Q Cycle Activity:

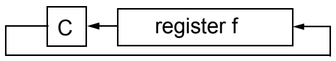
Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: `RETURN`

After Instruction:
PC = TOS

RLCF — Rotate Left f through Carry

Syntax:	RLCF f {,d {,a}}
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f<n>) → dest<n + 1>, (f<7>) → C, (C) → dest<0>
Status Affected:	C, N, Z
Encoding:	`0011` `01da` `ffff` `ffff`
Description:	The contents of register 'f' are rotated one bit to the left through the CARRY flag. If 'd' is '`0`', the result is placed in W. If 'd' is '`1`', the result is stored back in register 'f' (default). If 'a' is '`0`', the Access Bank is selected. If 'a' is '`1`', the BSR is used to select the GPR bank. If 'a' is '`0`' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `RLCF REG, 0, 0`

Before Instruction
REG = `1110 0110`
C = 0
After Instruction
REG = `1110 0110`
`W` = `1100 1100`
C = 1

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3FA1h	T4HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3FA0h	T4CON	ON	CKPS			OUTPS				338
3F9Fh	T4PR	PR4								337
3F9Eh	T4TMR	TMR4								337
3F9Dh	T5CLK	CS								335
3F9Ch	T5GATE	GSS								316
3F9Bh	T5GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3F9Ah	T5CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	338
3F99h	TMR5H	TMR5H								317
3F98h	TMR5L	TMR5L								317
3F97h	T6RST	—	—	—	RSEL					336
3F96h	T6CLK	—	—	—	—	CS				315
3F95h	T6HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3F94h	T6CON	ON	CKPS			OUTPS				338
3F93h	T6PR	PR6								337
3F92h	T6TMR	TMR6								337
3F91h - 3F80h	—	Unimplemented								
3F7Fh	CCP1CAP	CTS								352
3F7Eh	CCP1CON	EN	—	OUT	FMT	MODE				350
3F7Dh	CCPR1H	RH								353
3F7Ch	CCPR1L	RL								352
3F7Bh	CCP2CAP	CTS								352
3F7Ah	CCP2CON	EN	—	OUT	FMT	MODE				350
3F79h	CCPR2H	RH								353
3F78h	CCPR2L	RL								352
3F77h	CCP3CAP	CTS								352
3F76h	CCP3CON	EN	—	OUT	FMT	MODE				350
3F75h	CCPR3H	RH								353
3F74h	CCPR3L	RL								352
3F73h	CCP4CAP	CTS								352
3F72h	CCP4CON	EN	—	OUT	FMT	MODE				350
3F71h	CCPR4H	RH								353
3F70h	CCPR4L	RL								352
3F6Fh	—	Unimplemented								
3F6Eh	PWM5CON	EN	—	OUT	POL	—	—	—	—	358
3F6Dh	PWM5DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	360
3F6Dh	PWM5DCH	DC8								360
3F6Ch	PWM5DCL	DC1	DC0	—	—	—	—	—	—	360
3F6Ch	PWM5DCL	DC		—	—	—	—	—	—	360
3F6Bh	—	Unimplemented								
3F6Ah	PWM6CON	EN	—	OUT	POL	—	—	—	—	358
3F69h	PWM6DCH	DC9		DC7	DC6	DC5	DC4	DC3	DC2	360
3F69h	PWM6DCH	DC								360
3F68h	PWM6DCL	DC1	DC0	—	—	—	—	—	—	360
3F68h	PWM6DCL	DC		—	—	—	—	—	—	360
3F67h	—	Unimplemented								

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition

Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.
4: Unimplemented in PIC18(L)F45/55K42.

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3BFEh	DMA1AIRQ	—	AIRQ							256
3BFDh	DMA1CON1	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	249
3BFCh	DMA1CON0	DMODE		DSTP	SMR		SMODE		SSTP	248
3BFBh	DMA1SSAU	—	—	SSA						251
3BFAh	DMA1SSAH	SSA								250
3BF9h	DMA1SSAL	SSA								250
3BF8h	DMA1SSZH	—	—	—	—	SSZ				252
3BF7h	DMA1SSZL	SSZ								252
3BF6h	DMA1SPTRU	—	—	SPTR						252
3BF5h	DMA1SPTRH	SPTR								251
3BF4h	DMA1SPTRL	SPTR								251
3BF3h	DMA1SCNTH	—	—	—	—	SCNT				253
3BF2h	DMA1SCNTL	SCNT								253
3BF1h	DMA1DSAH	DSA								254
3BF0h	DMA1DSAL	SSA								253
3BEFh	DMA1DSZH	—	—	—	—	DSZ				255
3BEEh	DMA1DSZL	DSZ								255
3BEDh	DMA1DPTRH	DPTR								254
3BECh	DMA1DPTRL	DPTR								254
3BEBh	DMA1DCNTH	—	—	—	—	DCNT				256
3BEAh	DMA1DCNTL	DCNT								255
3BE9h	DMA1BUF	BUF								250
3BE8h - 3BE0h	—	Unimplemented								
3BDFh	DMA2SIRQ	—	SIRQ							256
3BDEh	DMA2AIRQ	—	AIRQ							256
3BDDh	DMA2CON1	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	249
3BDCh	DMA2CON0	DMODE		DSTP	SMR		SMODE		SSTP	248
3BDBh	DMA2SSAU	—	—	SSA						251
3BDAh	DMA2SSAH	SSA								250
3BD9h	DMA2SSAL	SSA								250
3BD8h	DMA2SSZH	—	—	—	—	SSZ				252
3BD7h	DMA2SSZL	SSZ								252
3BD6h	DMA2SPTRU	—	—	SPTR						252
3BD5h	DMA2SPTRH	SPTR								251
3BD4h	DMA2SPTRL	SPTR								251
3BD3h	DMA2SCNTH	—	—	—	—	SCNT				253
3BD2h	DMA2SCNTL	SCNT								253
3BD1h	DMA2DSAH	DSA								254
3BD0h	DMA2DSAL	SSA								253
3BCFh	DMA2DSZH	—	—	—	—	DSZ				255
3BCEh	DMA2DSZL	DSZ								255
3BCDh	DMA2DPTRH	DPTR								254
3BCCh	DMA2DPTRL	DPTR								254
3BCBh	DMA2DCNTH	—	—	—	—	DCNT				256
3BCAh	DMA2DCNTL	DCNT								255
3BC9h	DMA2BUF	BUF								250

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.
4: Unimplemented in PIC18(L)F45/55K42.

43.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

43.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

43.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

43.9 PICkit 3 In-Circuit Debugger/ Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

43.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.