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, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 30x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f46k22-i-p

2.13 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM can detect oscillator failure any time after the Oscillator Start-up Timer (OST) has expired. The FSCM is enabled by setting the FCMEN bit in the CONFIG1H Configuration register. The FSCM is applicable to all external oscillator modes (LP, XT, HS, EC, RC and RCIO).

FIGURE 2-10:   FSCM BLOCK DIAGRAM
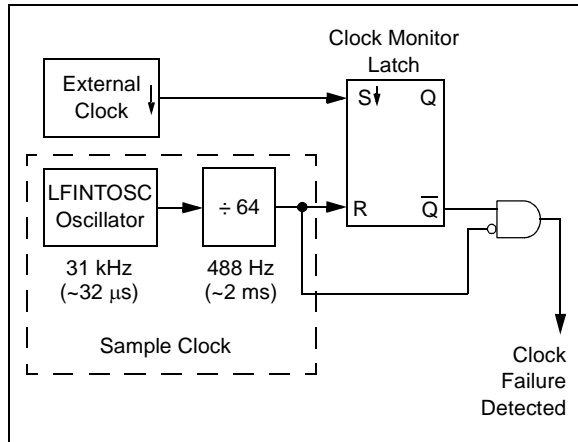
2.13.1   FAIL-SAFE DETECTION

The FSCM module detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64 (see Figure 2-10). Inside the fail detector block is a latch. The external clock sets the latch on each falling edge of the external clock. The sample clock clears the latch on each rising edge of the sample clock. A failure is detected when an entire half-cycle of the sample clock elapses before the primary clock goes low.

2.13.2   FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the device clock to an internal clock source and sets the bit flag OSCFIF of the PIR2 register. The OSCFIF flag will generate an interrupt if the OSCFIE bit of the PIE2 register is also set. The device firmware can then take steps to mitigate the problems that may arise from a failed clock. The system clock will continue to be sourced from the internal clock source until the device firmware successfully restarts the external oscillator and switches back to external operation. An automatic transition back to the failed clock source will not occur.

The internal clock source chosen by the FSCM is determined by the IRCF<2:0> bits of the OSCCON register. This allows the internal oscillator to be configured before a failure occurs.

2.13.3   FAIL-SAFE CONDITION CLEARING

The Fail-Safe condition is cleared by either one of the following:

• Any Reset
• By toggling the SCS1 bit of the OSCCON register

Both of these conditions restart the OST. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON. When the OST times out, the Fail-Safe condition is cleared and the device automatically switches over to the external clock source. The Fail-Safe condition need not be cleared before the OSCFIF flag is cleared.

2.13.4   RESET OR WAKE-UP FROM SLEEP

The FSCM is designed to detect an oscillator failure after the Oscillator Start-up Timer (OST) has expired. The OST is used after waking up from Sleep and after any type of Reset. The OST is not used with the EC or RC Clock modes so that the FSCM will be active as soon as the Reset or wake-up has completed.

> **Note:** Due to the wide range of oscillator start-up times, the Fail-Safe circuit is not active during oscillator start-up (i.e., after exiting Reset or Sleep). After an appropriate amount of time, the user should check the OSTS bit of the OSCCON register to verify the oscillator start-up and that the system clock switchover has successfully completed.

> **Note:** When the device is configured for Fail-Safe clock monitoring in either HS, XT, or LS Oscillator modes then the IESO config-uration bit should also be set so that the clock will automatically switch from the internal clock to the external oscillator when the OST times out.

FIGURE 4-5: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V$_{DD}$): CASE 2
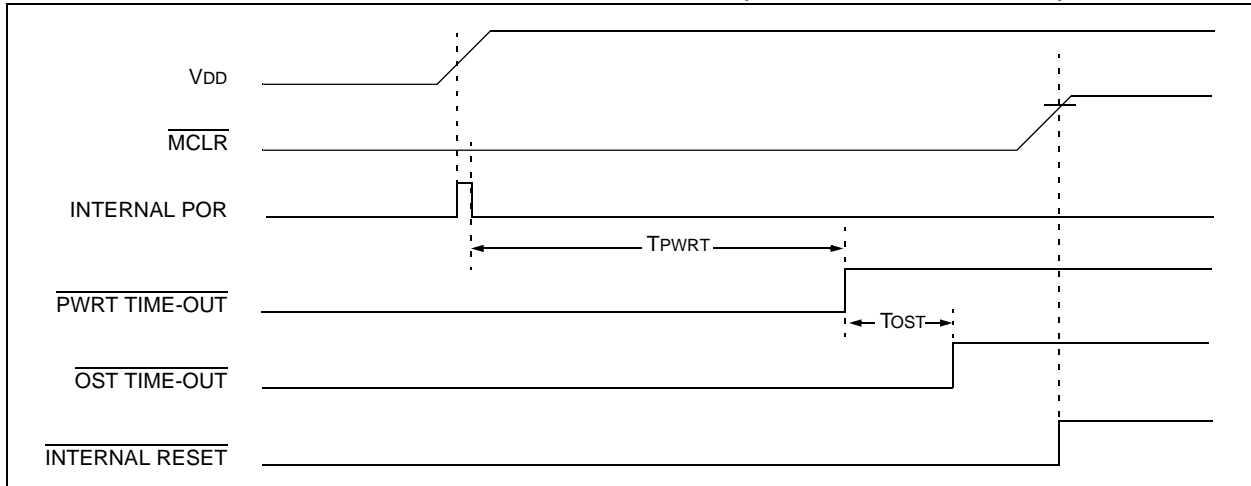

FIGURE 4-6: SLOW RISE TIME ($\overline{\text{MCLR}}$ TIED TO V$_{DD}$, V$_{DD}$ RISE > T$_{PWRT}$)
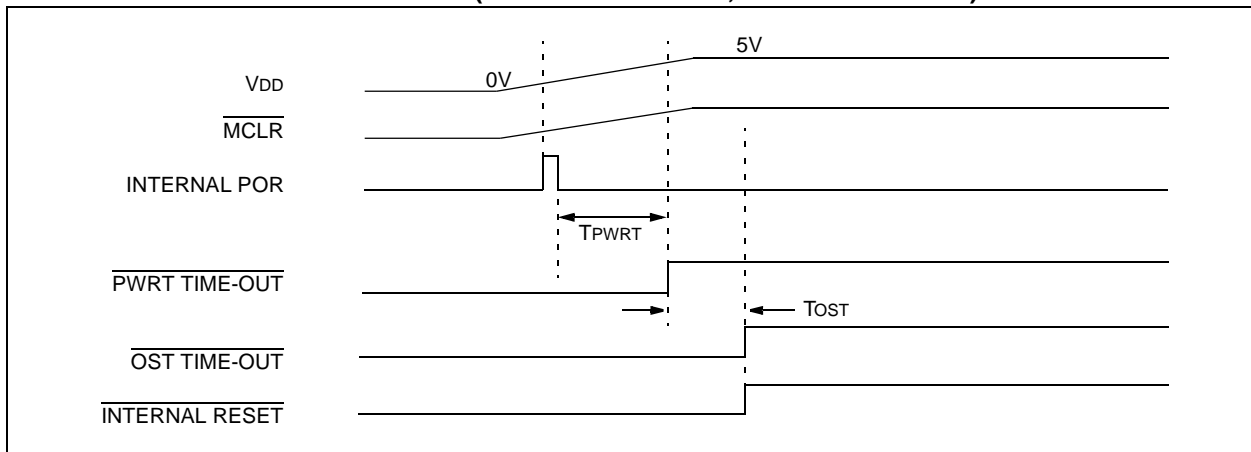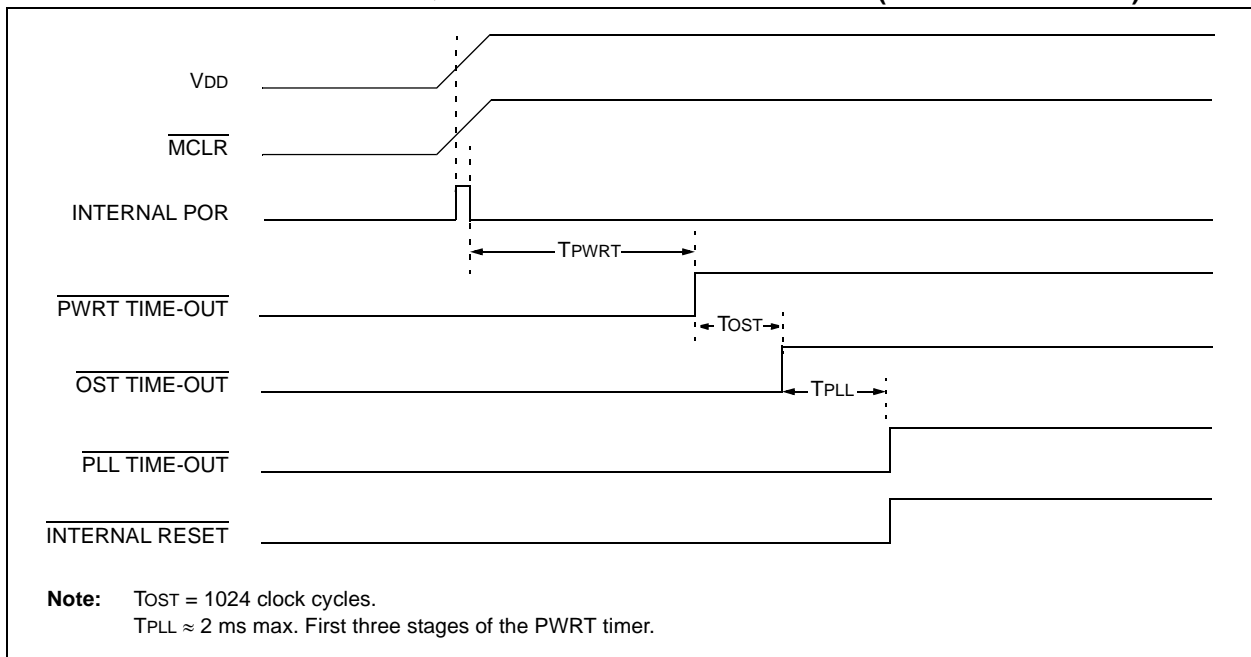

FIGURE 4-7: TIME-OUT SEQUENCE ON POR W/PLL ENABLED ($\overline{\text{MCLR}}$ TIED TO V$_{DD}$)

Note: T$_{OST}$ = 1024 clock cycles.
T$_{PLL}$ ≈ 2 ms max. First three stages of the PWRT timer.

PIC18(L)F2X/4XK22

5.3.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as either two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSb = 0). To maintain alignment with instruction boundaries, the PC increments in steps of two and the LSb will always read '0' (see **Section 5.1.1 "Program Counter"**).

Figure 5-4 shows an example of how instruction words are stored in the program memory.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction #2 in Figure 5-4 shows how the instruction GOTO 0006h is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 25.0 "Instruction Set Summary"** provides further details of the instruction set.

FIGURE 5-4: INSTRUCTIONS IN PROGRAM MEMORY

				LSB = 1	LSB = 0	Word Address ↓
		Program Memory Byte Locations →				000000h
						000002h
						000004h
						000006h
Instruction 1:	MOVLW	055h		0Fh	55h	000008h
Instruction 2:	GOTO	0006h		EFh	03h	00000Ah
				F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h		C1h	23h	00000Eh
				F4h	56h	000010h
						000012h
						000014h

5.3.4 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LSFR. In all cases, the second word of the instruction always has '1111' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence – immediately after the first word – the data in the second word is accessed and used by the instruction sequence.

If the first word is skipped for some reason and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 5-4 shows how this works.

> **Note:** See **Section 5.8 "PIC18 Instruction Execution and the Extended Instruction Set"** for information on two-word instructions in the extended instruction set.

EXAMPLE 5-4: TWO-WORD INSTRUCTIONS

CASE 1:

Object Code	Source Code
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; No, skip this word
1111 0100 0101 0110	; Execute this word as a NOP
0010 0100 0000 0000	ADDWF REG3 ; continue code

CASE 2:

Object Code	Source Code
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; Yes, execute this word
1111 0100 0101 0110	; 2nd word of instruction
0010 0100 0000 0000	ADDWF REG3 ; continue code

6.3 Register Definitions: Memory Control

REGISTER 6-1: EECON1: DATA EEPROM CONTROL 1 REGISTER

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit		
S = Bit can be set by software, but not cleared		U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7   **EEPGD:** Flash Program or Data EEPROM Memory Select bit
          1 = Access Flash program memory
          0 = Access data EEPROM memory

bit 6   **CFGS:** Flash Program/Data EEPROM or Configuration Select bit
          1 = Access Configuration registers
          0 = Access Flash program or data EEPROM memory

bit 5   **Unimplemented:** Read as '0'

bit 4   **FREE:** Flash Row (Block) Erase Enable bit
          1 = Erase the program memory block addressed by TBLPTR on the next WR command
              (cleared by completion of erase operation)
          0 = Perform write-only

bit 3   **WRERR:** Flash Program/Data EEPROM Error Flag bit[1]
          1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal
              operation, or an improper write attempt)
          0 = The write operation completed

bit 2   **WREN:** Flash Program/Data EEPROM Write Enable bit
          1 = Allows write cycles to Flash program/data EEPROM
          0 = Inhibits write cycles to Flash program/data EEPROM

bit 1   **WR:** Write Control bit
          1 = Initiates a data EEPROM erase/write cycle or a program memory erase cycle or write cycle.
              (The operation is self-timed and the bit is cleared by hardware once write is complete.
              The WR bit can only be set (not cleared) by software.)
          0 = Write cycle to the EEPROM is complete

bit 0   **RD:** Read Control bit
          1 = Initiates an EEPROM read (Read takes one cycle. RD is cleared by hardware. The RD bit can only
              be set (not cleared) by software. RD bit cannot be set when EEPGD = 1 or CFGS = 1.)
          0 = Does not initiate an EEPROM read

Note 1: When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the
error condition.

REGISTER 9-10:   PIE2: PERIPHERAL INTERRUPT ENABLE (FLAG) REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	C1IE	C2IE	EEIE	BCL1IE	HLVDIE	TMR3IE	CCP2IE
bit 7							bit 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

bit 7   **OSCFIE:** Oscillator Fail Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 6   **C1IE:** Comparator C1 Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 5   **C2IE:** Comparator C2 Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 4   **EEIE:** Data EEPROM/Flash Write Operation Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 3   **BCL1IE:** MSSP1 Bus Collision Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 2   **HLVDIE:** Low-Voltage Detect Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 1   **TMR3IE:** TMR3 Overflow Interrupt Enable bit
           1 = Enabled
           0 = Disabled

bit 0   **CCP2IE:** CCP2 Interrupt Enable bit
           1 = Enabled
           0 = Disabled

REGISTER 9-17: IPR4: PERIPHERAL INTERRUPT PRIORITY REGISTER 4

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	CCP5IP	CCP4IP	CCP3IP
bit 7							bit 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

bit 7-3   **Unimplemented:** Read as '0'

bit 2   **CCP5IP:** CCP5 Interrupt Priority bit
             1 = High priority
             0 = Low priority

bit 1   **CCP4IP:** CCP4 Interrupt Priority bit
             1 = High priority
             0 = Low priority

bit 0   **CCP3IP:** CCP3 Interrupt Priority bit
             1 = High priority
             0 = Low priority

REGISTER 9-18: IPR5: PERIPHERAL INTERRUPT PRIORITY REGISTER 5

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	TMR6IP	TMR5IP	TMR4IP
bit 7							bit 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

bit 7-3   **Unimplemented:** Read as '0'

bit 2   **TMR6IP:** TMR6 to PR6 Match Interrupt Priority bit
             1 = High priority
             0 = Low priority

bit 1   **TMR5IP:** TMR5 Overflow Interrupt Priority bit
             1 = High priority
             0 = Low priority

bit 0   **TMR4IP:** TMR4 to PR4 Match Interrupt Priority bit
             1 = High priority
             0 = Low priority

The I²C interface supports the following modes and features:

- Master mode
- Slave mode
- Byte NACKing (Slave mode)
- Limited Multi-master support
- 7-bit and 10-bit addressing
- Start and Stop interrupts
- Interrupt masking
- Clock stretching
- Bus collision detection
- General call address matching
- Address masking
- Address Hold and Data Hold modes
- Selectable SDAx hold times

Figure 15-2 is a block diagram of the I²C interface module in Master mode. Figure 15-3 is a diagram of the I²C interface module in Slave mode.

The PIC18(L)F2X/4XK22 has two MSSP modules, MSSP1 and MSSP2, each module operating independently from the other.

> **Note 1:** In devices with more than one MSSP module, it is very important to pay close attention to SSPxCONx register names. SSP1CON1 and SSP1CON2 registers control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.
>
> **2:** Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names, module I/O signals, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module when required.

FIGURE 15-2: MSSPx BLOCK DIAGRAM (I²C MASTER MODE)
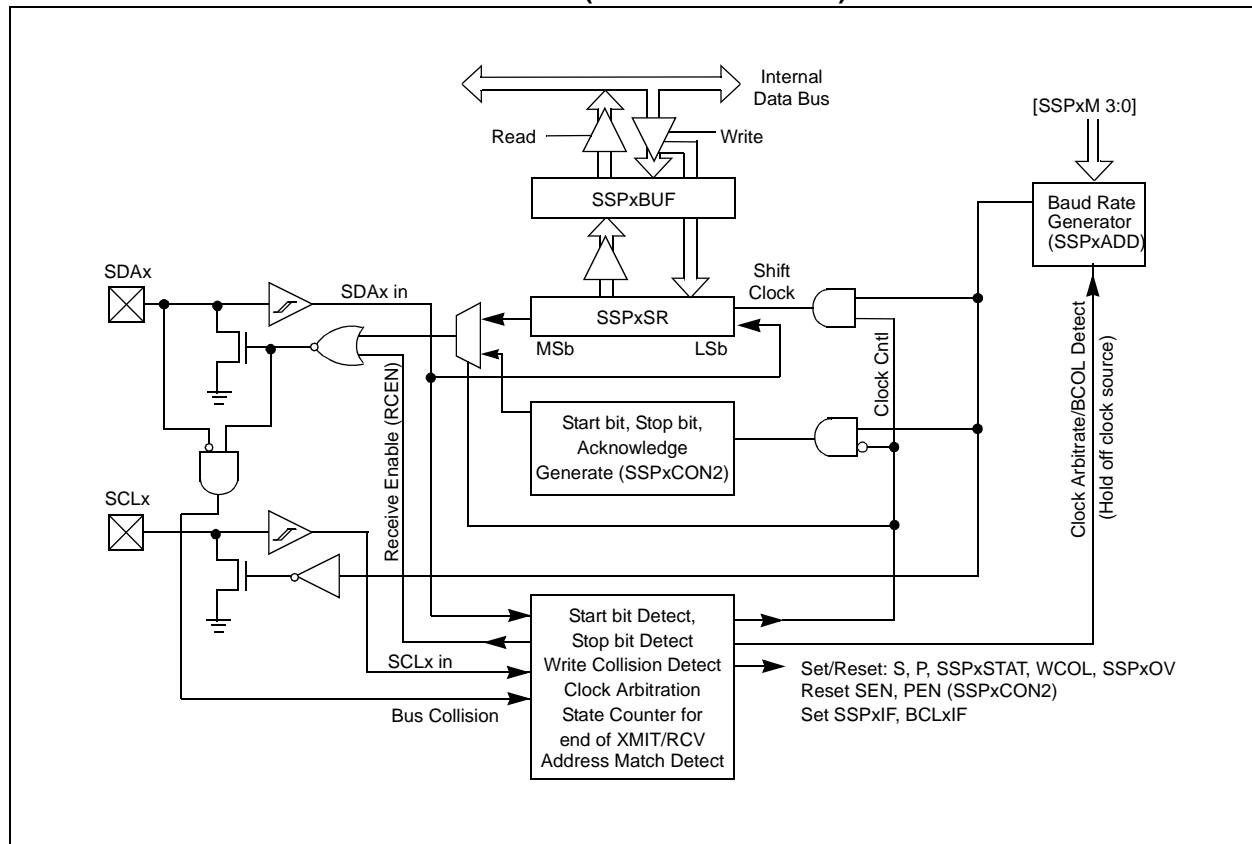
15.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCLx pulse for any transferred byte in I²C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDAx line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge ($\overline{ACK}$) is an active-low signal, pulling the SDAx line low indicated to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an $\overline{ACK}$ is placed in the ACKSTAT bit of the SSPxCON2 register.

Slave software, when the AHEN and DHEN bits are set, allow the user to set the $\overline{ACK}$ value sent back to the transmitter. The ACKDT bit of the SSPxCON2 register is set/cleared to determine the response.

Slave hardware will generate an $\overline{ACK}$ response if the AHEN and DHEN bits of the SSPxCON3 register are clear.

There are certain conditions where an $\overline{ACK}$ will not be sent by the slave. If the BF bit of the SSPxSTAT register or the SSPxOV bit of the SSPxCON1 register are set when a byte is received.

When the module is addressed, after the 8th falling edge of SCLx on the bus, the ACKTIM bit of the SSPx-CON3 register is set. The ACKTIM bit indicates the acknowledge time of the active bus.

The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is enabled.

15.5 I²C Slave Mode Operation

The MSSPx Slave mode operates in one of four modes selected in the SSPxM bits of SSPxCON1 register. The modes can be divided into 7-bit and 10-bit Addressing mode. 10-bit Addressing modes operate the same as 7-bit with some additional overhead for handling the larger addresses.

Modes with Start and Stop bit interrupts operated the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

15.5.1 SLAVE MODE ADDRESSES

The SSPxADD register (Register 15-7) contains the Slave mode address. The first byte received after a Start or Restart condition is compared against the value stored in this register. If the byte matches, the value is loaded into the SSPxBUF register and an interrupt is generated. If the value does not match, the module goes Idle and no indication is given to the software that anything happened.

The SSPx Mask register (Register 15-6) affects the address matching process. See **Section 15.5.9 "SSPx Mask Register"** for more information.

15.5.1.1   I²C Slave 7-bit Addressing Mode

In 7-bit Addressing mode, the LSb of the received data byte is ignored when determining if there is an address match.

15.5.1.2   I²C Slave 10-bit Addressing Mode

In 10-bit Addressing mode, the first received byte is compared to the binary value of '1 1 1 1 0 A9 A8 0'. A9 and A8 are the two MSb of the 10-bit address and stored in bits 2 and 1 of the SSPxADD register.

After the acknowledge of the high byte the UA bit is set and SCLx is held low until the user updates SSPxADD with the low address. The low address byte is clocked in and all eight bits are compared to the low address value in SSPxADD. Even if there is not an address match; SSPxIF and UA are set, and SCLx is held low until SSPxADD is updated to receive a high byte again. When SSPxADD is updated the UA bit is cleared. This ensures the module is ready to receive the high address byte on the next communication.

A high and low address match as a write request is required at the start of all 10-bit addressing communication. A transmission can be initiated by issuing a Restart once the slave is addressed, and clocking in the high address with the R/$\overline{W}$ bit set. The slave hardware will then acknowledge the read request and prepare to clock out data. This is only valid for a slave after it has received a complete high and low address byte match.

15.6.5 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition (Figure 15-27) occurs when the RSEN bit of the SSPxCON2 register is programmed high and the master state machine is no longer active. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. SCLx is asserted low.

Following this, the RSEN bit of the SSPxCON2 register will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit of the SSPxSTAT register will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

Note 1: If RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

FIGURE 15-27:   REPEAT START CONDITION WAVEFORM
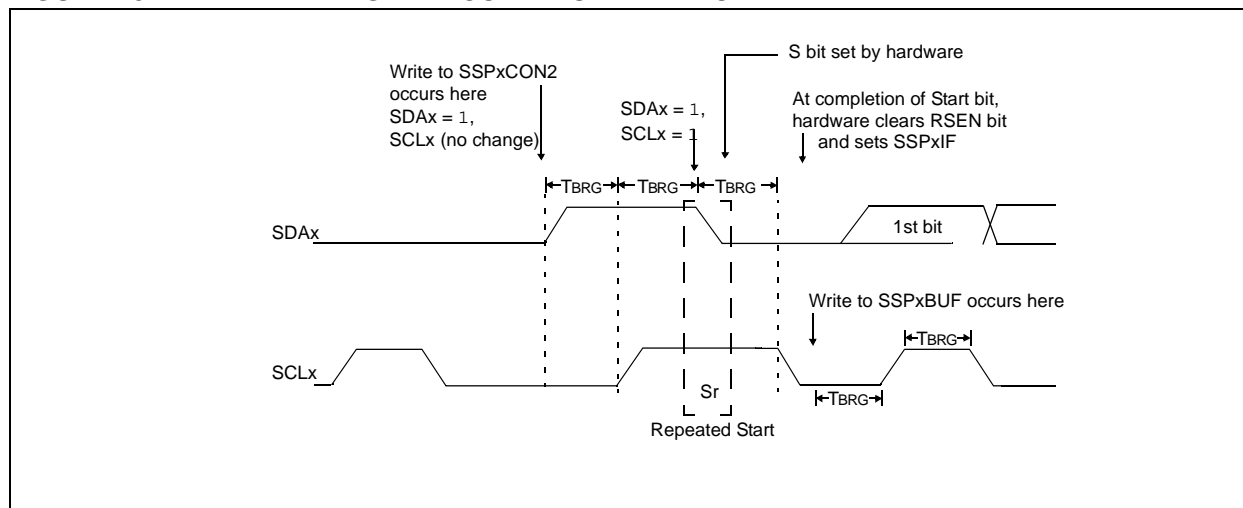
TABLE 15-2: REGISTERS ASSOCIATED WITH I²C OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	149
ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1[1]	ANSB0[1]	150
ANSELC	ANSC7	ANSC6	ANSC5	ANSC4	ANSC3	ANSC2	—	—	150
ANSELD	ANSD7	ANSD6	ANSD5	ANSD4	ANSD3	ANSD2	ANSD1[2]	ANSD0[2]	150
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	109
IPR1	—	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	121
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCL1IP	HLVDIP	TMR3IP	CCP2IP	122
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	CTMUIP	TMR5GIP	TMR3GIP	TMR1GIP	123
PIE1	—	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	117
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCL1IE	HLVDIE	TMR3IE	CCP2IE	118
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	CTMUIE	TMR5GIE	TMR3GIE	TMR1GIE	119
PIR1	—	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	112
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCL1IF	HLVDIF	TMR3IF	CCP2IF	113
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	CTMUIF	TMR5GIF	TMR3GIF	TMR1GIF	114
PMD1	MSSP2MD	MSSP1MD	—	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	53
SSP1ADD	SSP1 Address Register in I²C Slave mode. SSP1 Baud Rate Reload Register in I²C Master mode.								258
SSP1BUF	SSP1 Receive Buffer/Transmit Register								—
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				253
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	255
SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	256
SSP1MSK	SSP1 MASK Register bits								257
SSP1STAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	252
SSP2ADD	SSP2 Address Register in I²C Slave mode. SSP2 Baud Rate Reload Register in I²C Master mode.								258
SSP2BUF	SSP2 Receive Buffer/Transmit Register								—
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				253
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	255
SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	256
SSP2MSK	SSP1 MASK Register bits								257
SSP2STAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	252
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1[1]	TRISB0[1]	151
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	151
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1[2]	TRISD0[2]	151

Legend: Shaded bits are not used by the MSSPx in I²C mode.

Note 1: PIC18(L)F2XK22 devices.

2: PIC18(L)F4XK22 devices.

REGISTER 15-7: SSPxADD: MSSPx ADDRESS AND BAUD RATE REGISTER (I²C MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
			ADD<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	

Master mode:

bit 7-0   **ADD<7:0>:** Baud Rate Clock Divider bits
SCLx pin clock period = ((ADD<7:0> + 1) *4)/F$_{OSC}$

10-Bit Slave mode — Most Significant Address byte:

bit 7-3   **Not used:** Unused for Most Significant Address byte. Bit state of this register is a "don't care". Bit pattern sent by master is fixed by I²C specification and must be equal to '11110'. However, those bits are compared by hardware and are not affected by the value in this register.

bit 2-1   **ADD<2:1>:** Two Most Significant bits of 10-bit address

bit 0   **Not used:** Unused in this mode. Bit state is a "don't care".

10-Bit Slave mode — Least Significant Address byte:

bit 7-0   **ADD<7:0>:** Eight Least Significant bits of 10-bit address

7-Bit Slave mode:

bit 7-1   **ADD<7:1>:** 7-bit address

bit 0   **Not used:** Unused in this mode. Bit state is a "don't care".

16.5 EUSART Synchronous Mode

Synchronous serial communications are typically used in systems with a single master and one or more slaves. The master device contains the necessary circuitry for baud rate generation and supplies the clock for all devices in the system. Slave devices can take advantage of the master clock by eliminating the internal clock generation circuitry.

There are two signal lines in Synchronous mode: a bidirectional data line and a clock line. Slaves use the external clock supplied by the master to shift the serial data into and out of their respective receive and transmit shift registers. Since the data line is bidirectional, synchronous operation is half-duplex only. Half-duplex refers to the fact that master and slave devices can receive and transmit data but not both simultaneously. The EUSART can operate as either a master or slave device.

Start and Stop bits are not used in synchronous transmissions.

16.5.1 SYNCHRONOUS MASTER MODE

The following bits are used to configure the EUSART for Synchronous Master operation:

- SYNC = 1
- CSRC = 1
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXSTAx register configures the device for synchronous operation. Setting the CSRC bit of the TXSTAx register configures the device as a master. Clearing the SREN and CREN bits of the RCSTAx register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCSTAx register enables the EUSART. If the RXx/DTx or TXx/CKx pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

The TRIS bits corresponding to the RXx/DTx and TXx/CKx pins should be set.

16.5.1.1 Master Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a master transmits the clock on the TXx/CKx line. The TXx/CKx pin output driver is automatically enabled when the EUSART is configured for synchronous transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One clock cycle is generated for each data bit. Only as many clock cycles are generated as there are data bits.

16.5.1.2 Clock Polarity

A clock polarity option is provided for Microwire compatibility. Clock polarity is selected with the CKTXP bit of the BAUDCONx register. Setting the CKTXP bit sets the clock Idle state as high. When the CKTXP bit is set, the data changes on the falling edge of each clock and is sampled on the rising edge of each clock. Clearing the CKTXP bit sets the Idle state as low. When the CKTXP bit is cleared, the data changes on the rising edge of each clock and is sampled on the falling edge of each clock.

16.5.1.3 Synchronous Master Transmission

Data is transferred out of the device on the RXx/DTx pin. The RXx/DTx and TXx/CKx pin output drivers are automatically enabled when the EUSART is configured for synchronous master transmit operation.

A transmission is initiated by writing a character to the TXREGx register. If the TSR still contains all or part of a previous character the new character data is held in the TXREGx until the last bit of the previous character has been transmitted. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXREGx is immediately transferred to the TSR. The transmission of the character commences immediately following the transfer of the data to the TSR from the TXREGx.

Each data bit changes on the leading edge of the master clock and remains valid until the subsequent leading clock edge.

> **Note:** The TSR register is not mapped in data memory, so it is not available to the user.

16.5.1.4 Data Polarity

The polarity of the transmit and receive data can be controlled with the DTRXP bit of the BAUDCONx register. The default state of this bit is '0' which selects high true transmit and receive data. Setting the DTRXP bit to '1' will invert the data resulting in low true transmit and receive data.

20.0 SR LATCH

The module consists of a single SR latch with multiple Set and Reset inputs as well as separate latch outputs. The SR latch module includes the following features:

- Programmable input selection
- SR latch output is available internally/externally
- Selectable Q and $\overline{Q}$ output
- Firmware Set and Reset

The SR latch can be used in a variety of analog applications, including oscillator circuits, one-shot circuit, hysteretic controllers, and analog timing applications.

20.1 Latch Operation

The latch is a Set-Reset latch that does not depend on a clock source. Each of the Set and Reset inputs are active-high. The latch can be set or reset by:

- Software control (SRPS and SRPR bits)
- Comparator C1 output (sync_C1OUT)
- Comparator C2 output (sync_C2OUT)
- SRI Pin
- Programmable clock (DIVSRCLK)

The SRPS and the SRPR bits of the SRCON0 register may be used to set or reset the SR latch, respectively. The latch is Reset-dominant. Therefore, if both Set and Reset inputs are high, the latch will go to the Reset state. Both the SRPS and SRPR bits are self resetting which means that a single write to either of the bits is all that is necessary to complete a latch Set or Reset operation.

The output from Comparator C1 or C2 can be used as the Set or Reset inputs of the SR latch. The output of either Comparator can be synchronized to the Timer1 clock source. See **Section 18.0 "Comparator Module"** and **Section 12.0 "Timer1/3/5 Module with Gate Control"** for more information.

An external source on the SRI pin can be used as the Set or Reset inputs of the SR latch.

An internal clock source, DIVSRCLK, is available and it can periodically set or reset the SR latch. The SRCLK<2:0> bits in the SRCON0 register are used to select the clock source period. The SRSCKE and SRRCKE bits of the SRCON1 register enable the clock source to set or reset the SR latch, respectively.

20.2 Latch Output

The SRQEN and SRNQEN bits of the SRCON0 register control the Q and $\overline{Q}$ latch outputs. Both of the SR latch outputs may be directly output to I/O pins at the same time. Control is determined by the state of bits SRQEN and SRNQEN in the SRCON0 register.

The applicable TRIS bit of the corresponding port must be cleared to enable the port pin output driver.

20.3 DIVSRCLK Clock Generation

The DIVSRCLK clock signal is generated from the peripheral clock which is pre-scaled by a value determined by the SRCLK<2:0> bits. See Figure 20-2 and Table 20-1 for additional detail.

20.4 Effects of a Reset

Upon any device Reset, the SR latch is not initialized, and the SRQ and SRNQ outputs are unknown. The user's firmware is responsible to initialize the latch output before enabling it to the output pins.

23.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

The PIC18(L)F2X/4XK22 devices have a High/Low-Voltage Detect module (HLVD). This is a programmable circuit that sets both a device voltage trip point and the direction of change from that point. If the device experiences an excursion past the trip point in that direction, an interrupt flag is set. If the interrupt is enabled, the program execution branches to the interrupt vector address and the software responds to the interrupt.

The High/Low-Voltage Detect Control register (Register 23-1) completely controls the operation of the HLVD module. This allows the circuitry to be "turned off" by the user under software control, which minimizes the current consumption for the device.

The module's block diagram is shown in Figure 23-1.

23.1 Register - HLVD Control

REGISTER 23-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER

R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL<3:0>			
bit 7							bit 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

bit 7 **VDIRMAG:** Voltage Direction Magnitude Select bit

`1` = Event occurs when voltage equals or exceeds trip point (HLVDL<3:0>)
`0` = Event occurs when voltage equals or falls below trip point (HLVDL<3:0>)

bit 6 **BGVST:** Band Gap Reference Voltages Stable Status Flag bit

`1` = Internal band gap voltage references are stable
`0` = Internal band gap voltage reference is not stable

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit

`1` = Indicates that the voltage detect logic will generate the interrupt flag at the specified voltage range
`0` = Indicates that the voltage detect logic will not generate the interrupt flag at the specified voltage range and the HLVD interrupt should not be enabled

bit 4 **HLVDEN:** High/Low-Voltage Detect Power Enable bit

`1` = HLVD enabled
`0` = HLVD disabled

bit 3-0 **HLVDL<3:0>:** Voltage Detection Level bits[1]

`1111` = External analog input is used (input comes from the HLVDIN pin)
`1110` = Maximum setting
.
.
.
`0000` = Minimum setting

Note 1: See Table 27-5 for specifications.

BTFSC	Bit Test File, Skip if Clear
Syntax:	BTFSC f, b {,a}
Operands:	$0 \le f \le 255$ $0 \le b \le 7$ $a \in [0,1]$
Operation:	skip if (f<b>) = 0
Status Affected:	None

Encoding:

1011	bbba	ffff	ffff

Description:	If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction. 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 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    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
    PC   =   address (HERE)
After Instruction
    If FLAG<1> =   0;
        PC   =   address (TRUE)
    If FLAG<1> =   1;
        PC   =   address (FALSE)

BTFSS	Bit Test File, Skip if Set
Syntax:	BTFSS f, b {,a}
Operands:	$0 \le f \le 255$ $0 \le b < 7$ $a \in [0,1]$
Operation:	skip if (f<b>) = 1
Status Affected:	None

Encoding:

1010	bbba	ffff	ffff

Description:	If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction. 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 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    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
    PC   =   address (HERE)
After Instruction
    If FLAG<1> =   0;
        PC   =   address (FALSE)
    If FLAG<1> =   1;
        PC   =   address (TRUE)

CPFSGT	Compare f with W, skip if f > W
Syntax:	CPFSGT f {,a}
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	(f) – (W), skip if (f) > (W) (unsigned comparison)
Status Affected:	None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.
If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.
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 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      CPFSGT REG, 0
NGREATER  :
GREATER   :
```
Before Instruction
PC   =   Address (HERE)
W   =   ?
After Instruction
If REG   >   W;
    PC   =   Address (GREATER)
If REG   ≤   W;
    PC   =   Address (NGREATER)

CPFSLT	Compare f with W, skip if f < W
Syntax:	CPFSLT f {,a}
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	(f) – (W), skip if (f) < (W) (unsigned comparison)
Status Affected:	None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.
If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

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      CPFSLT REG, 1
NLESS     :
LESS      :
```
Before Instruction
PC   =   Address (HERE)
W   =   ?
After Instruction
If REG   <   W;
PC   =   Address (LESS)
If REG   ≥   W;
PC   =   Address (NLESS)

MOVLW — Move literal to W

Syntax:	MOVLW k
Operands:	$0 \leq k \leq 255$
Operation:	$k \rightarrow W$
Status Affected:	None
Encoding:	

0000	1110	kkkk	kkkk

Description:	The 8-bit literal 'k' is loaded into W.
Words:	1
Cycles:	1

Q Cycle Activity:

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

Example: MOVLW   5Ah

  After Instruction

    W   =   5Ah

MOVWF — Move W to f

Syntax:	MOVWF f {,a}
Operands:	$0 \leq f \leq 255$ $a \in [0,1]$
Operation:	$(W) \rightarrow f$
Status Affected:	None
Encoding:	

0110	111a	ffff	ffff

Description:	Move data from W to register 'f'. Location 'f' can be anywhere in the 256-byte bank. 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 $\leq$ 95 (5Fh). See **Section 25.2.3 "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 register 'f'

Example: MOVWF   REG, 0

  Before Instruction

    W   =   4Fh
    REG   =   FFh
  After Instruction

    W   =   4Fh
    REG   =   4Fh

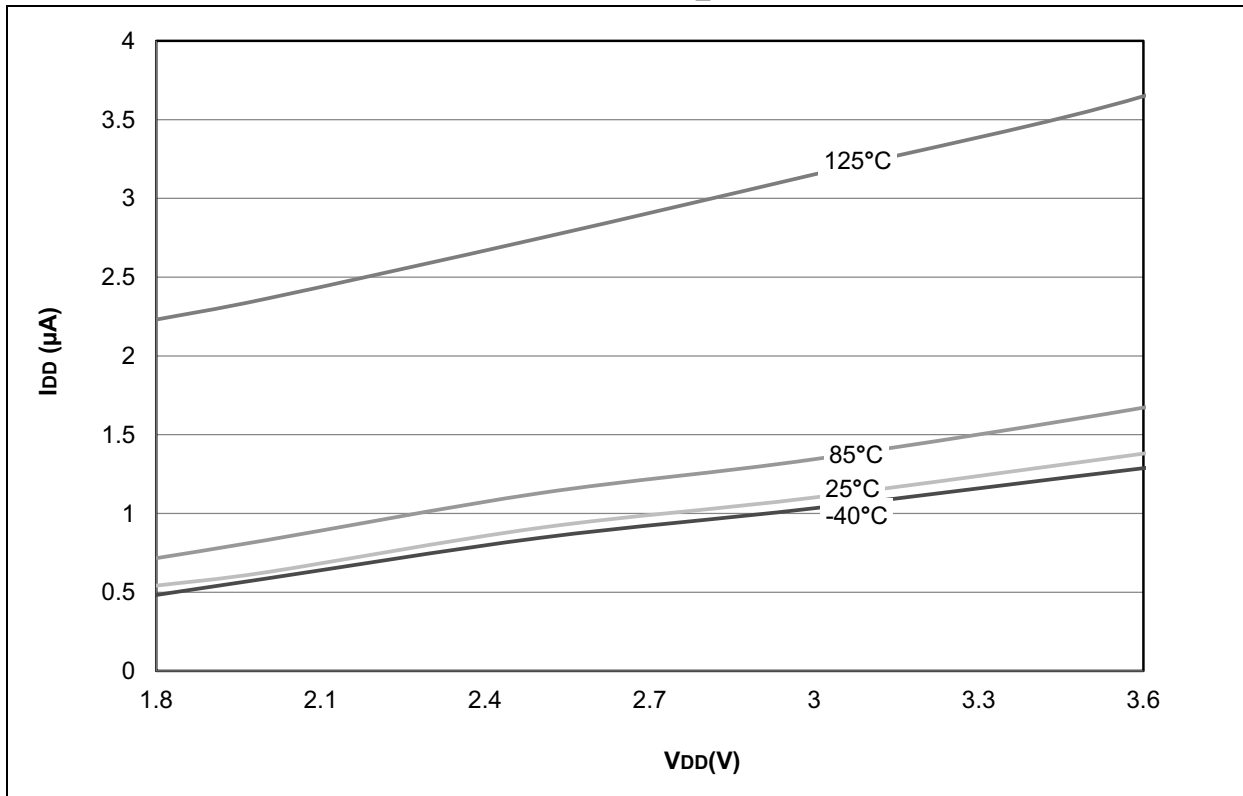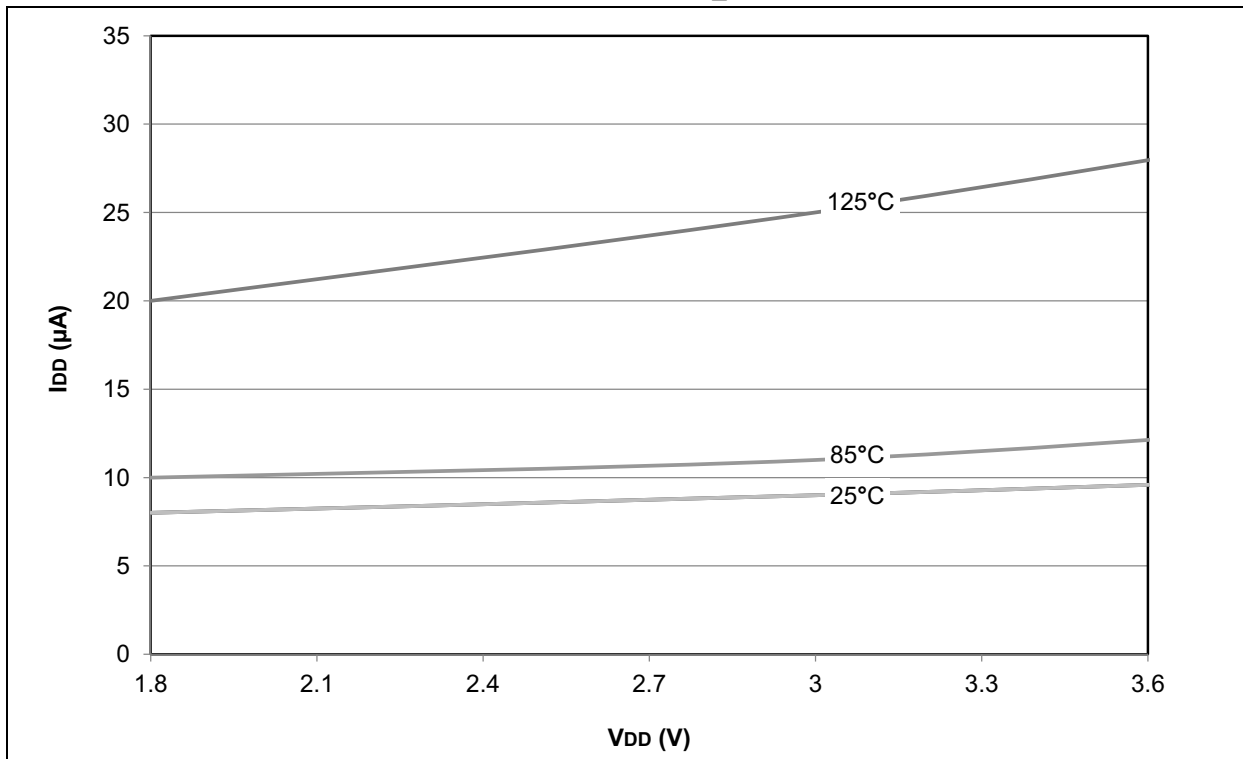**FIGURE 28-34:** **PIC18LF2X/4XK22 TYPICAL I_DD: RC_IDLE LF-INTOSC 31 kHz**

FIGURE 28-35: **PIC18LF2X/4XK22 MAXIMUM I_DD: RC_IDLE LF-INTOSC 31 kHz**

FIGURE 28-48: **PIC18LF2X/4XK22 TYPICAL I**DD**: PRI_RUN EC MEDIUM POWER**
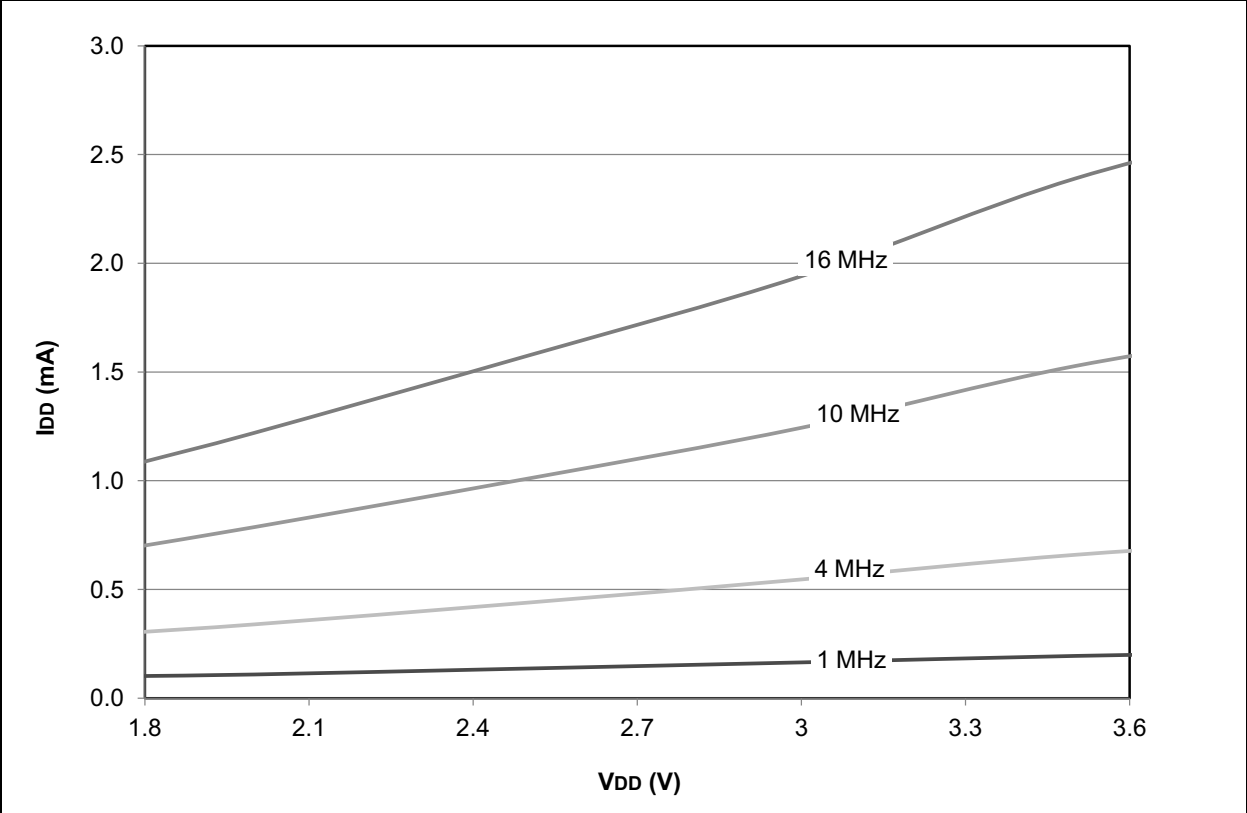

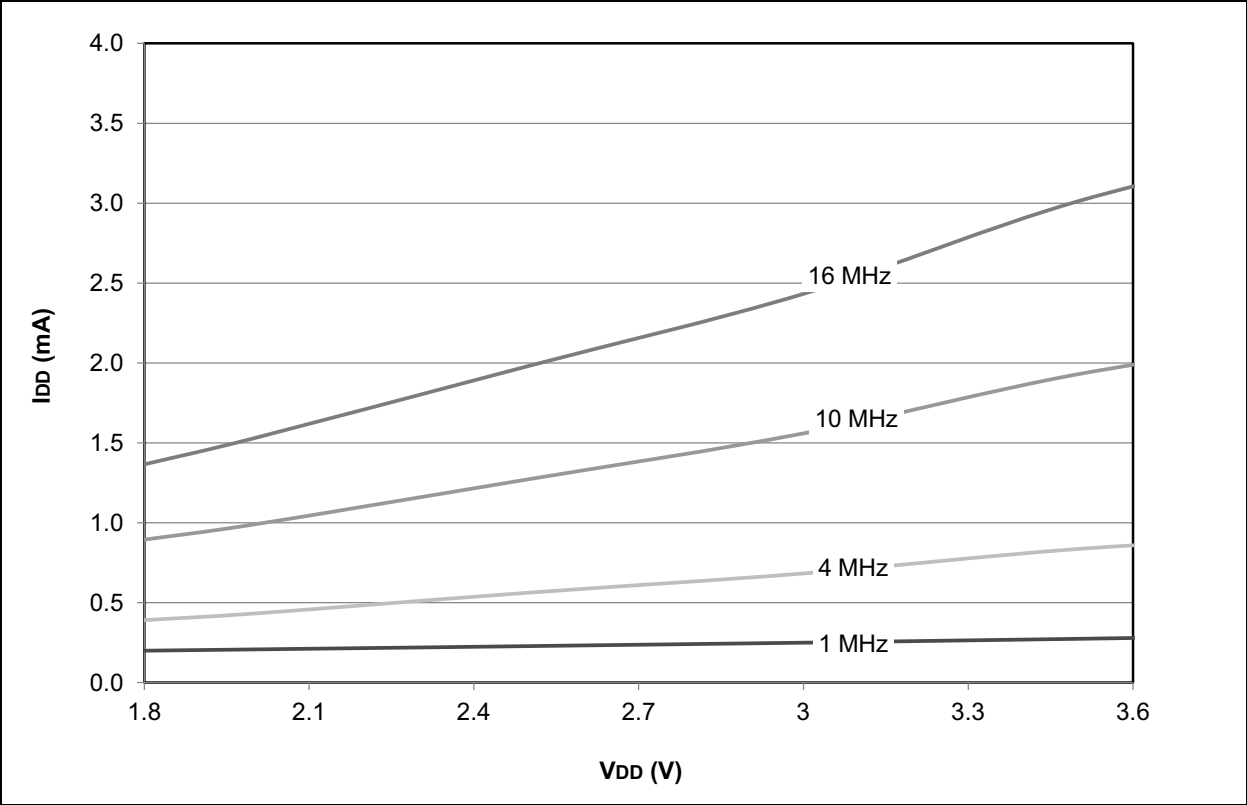**FIGURE 28-49:** **PIC18LF2X/4XK22 MAXIMUM I**DD**: PRI_RUN EC MEDIUM POWER**

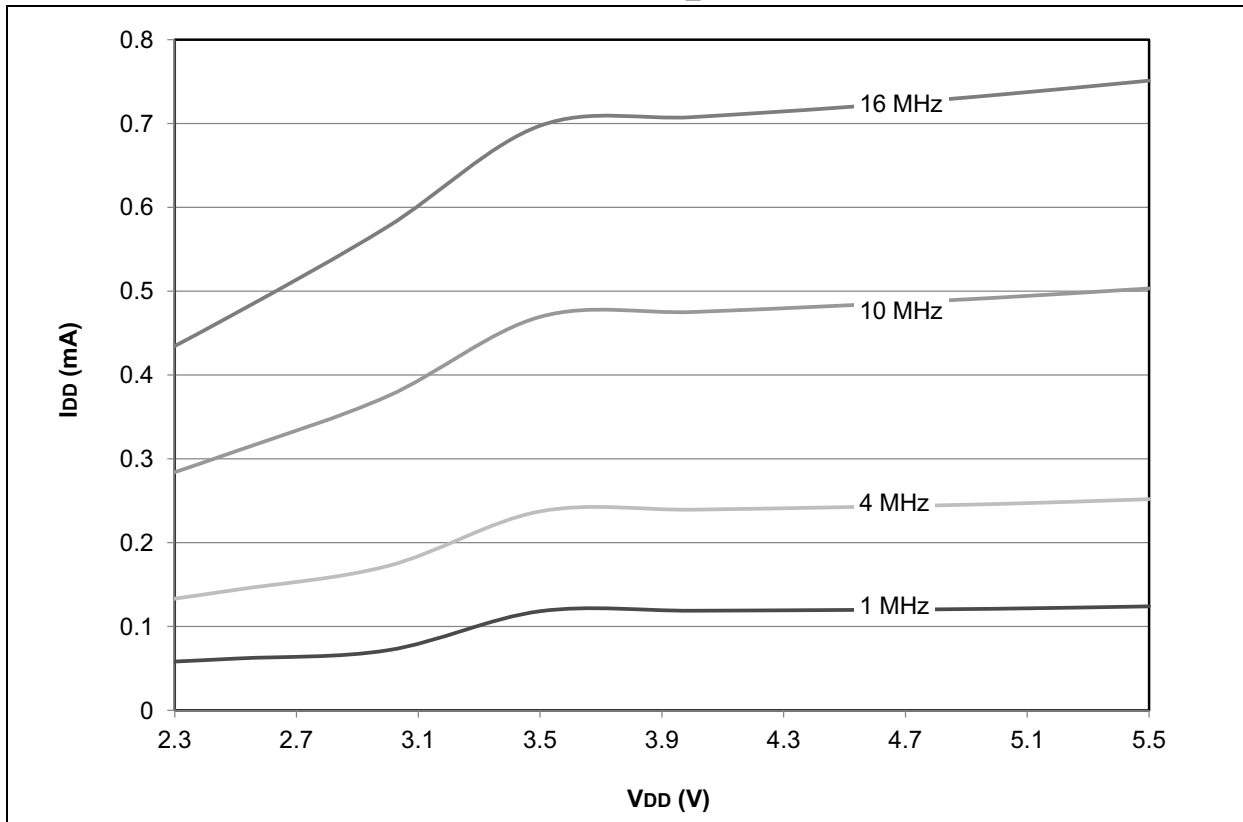**FIGURE 28-62:** **PIC18F2X/4XK22 TYPICAL I_{DD}: PRI_IDLE EC MEDIUM POWER**

FIGURE 28-63: **PIC18F2X/4XK22 MAXIMUM I_{DD}: PRI_IDLE EC MEDIUM POWER**