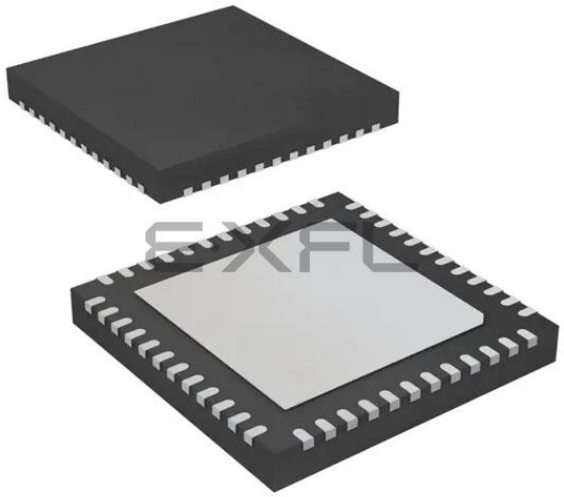


Welcome to E-XFL.COM

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, LVD, POR, PWM, WDT
Number of I/O	44
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 43x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-UFQFN Exposed Pad
Supplier Device Package	48-UQFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf55k42-i-mv

REGISTER 5-6: CONFIGURATION WORD 3H (30 0005h)

U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	WDTCCS<2:0>			WDTCWS<2:0>		
bit 7							bit 0

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

bit 7-6 **Unimplemented:** Read as '1'

bit 5-3 **WDTCCS<2:0>:** WDT Input Clock Selector bits

If WDTE<1:0> Fuses = 2'b00:

These bits are ignored.

Otherwise:

000 = WDT reference clock is the 31.0 kHz LFINTOSC
001 = WDT reference clock is the 31.25 kHz MFINTOSC
010 = WDT reference clock is SOSC
011 = Reserved (default to LFINTOSC)
•
•
110 = Reserved (default to LFINTOSC)
111 = Software control

bit 2-0 **WDTCWS<2:0>:** WDT Window Select bits

WDTCWS<2:0>	Window at POR			Software Control of Window	Keyed Access Required?
	Value	Window Delay Percent of Time	Window Opening Percent of Time		
000	000	87.5	12.5	No	Yes
001	001	75	25		
010	010	62.5	37.5		
011	011	50	50		
100	100	37.5	62.5		
101	101	25	75		
110	111	n/a	100		
111	111	n/a	100	Yes	No

5.7.1 MICROCHIP UNIQUE IDENTIFIER (MUI)

The PIC18(L)F26/27/45/46/47/55/56/57K42 devices are individually encoded during final manufacturing with a Microchip Unique Identifier, or MUI. The MUI cannot be user-erased. This feature allows for manufacturing traceability of Microchip Technology devices in applications where this is a required. It may also be used by the application manufacturer for a number of functions that require unverified unique identification, such as:

- Tracking the device
- Unique serial number

The MUI consists of six program words. When read together, these fields form a unique identifier. The MUI is stored in nine read-only locations, located between 3F0000h to 3F000Fh in the DIA space. Table 5-3 lists the addresses of the identifier words.

Note:	For applications that require verified unique identification, contact your Microchip Technology sales office to create a Serialized Quick Turn Programming^{SM} option.

5.7.2 EXTERNAL UNIQUE IDENTIFIER (EUI)

The EUI data is stored at locations 3F0010h to 3F0023h in the Program Memory region. This region is an optional space for placing application specific information. The data is coded per customer requirements during manufacturing.

Note:	Data is stored in this address range on receiving a request from the customer. The customer may contact the local sales representative, or Field Applications Engineer, and provide them the unique identifier information that is supposed to be stored in this region.

5.7.3 ANALOG-TO-DIGITAL CONVERSION DATA OF THE TEMPERATURE SENSOR

The purpose of the Temperature Sensor module is to provide a temperature-dependent voltage that can be measured by an analog module, see **Section 35.0 “Temperature Indicator Module”**.

The DIA table contains the internal ADC measurement values of the Temperature sensor for Low and High range at fixed points of reference. The values are measured during test and are unique to each device. The measurement data is stored in the DIA memory region as hexadecimal numbers corresponding to the ADC conversion result. The calibration data can be used to plot the approximate sensor output voltage, V_{TSENSE} vs. Temperature curve without having to make calibration measurements in the application. For more information on the operation of the Temperature Sensor, refer to **Section 35.0 “Temperature Indicator Module”**.

- **TSLR2**: Address 3F0026h to 3F0027h store the measurements for the low-range setting of the Temperature Sensor at V_{DD} = 3V.
- **TSHR2**: Address 3F002Ch to 3F002Dh store the measurements for the High Range setting of the Temperature Sensor at V_{DD} = 3V.
- The stored measurements are made by the device ADC using the internal V_{REF} = 2.048V.

REGISTER 9-29: IPR4: PERIPHERAL INTERRUPT PRIORITY REGISTER 4

R/W-1/1	R/W-1/1	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
CLC1IP	CWG1IP	NCO1IP	—	CCP1IP	TMR2IP	TMR1GIP	TMR1IP
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 **CLC1IP:** CLC1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **CWG1IP:** CWG1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **NCO1IP:** NCO1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **Unimplemented:** Read as '0'

bit 3 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **TMR2IP:** TMR2 Interrupt Priority bit
1 = High priority
0 = Low priority

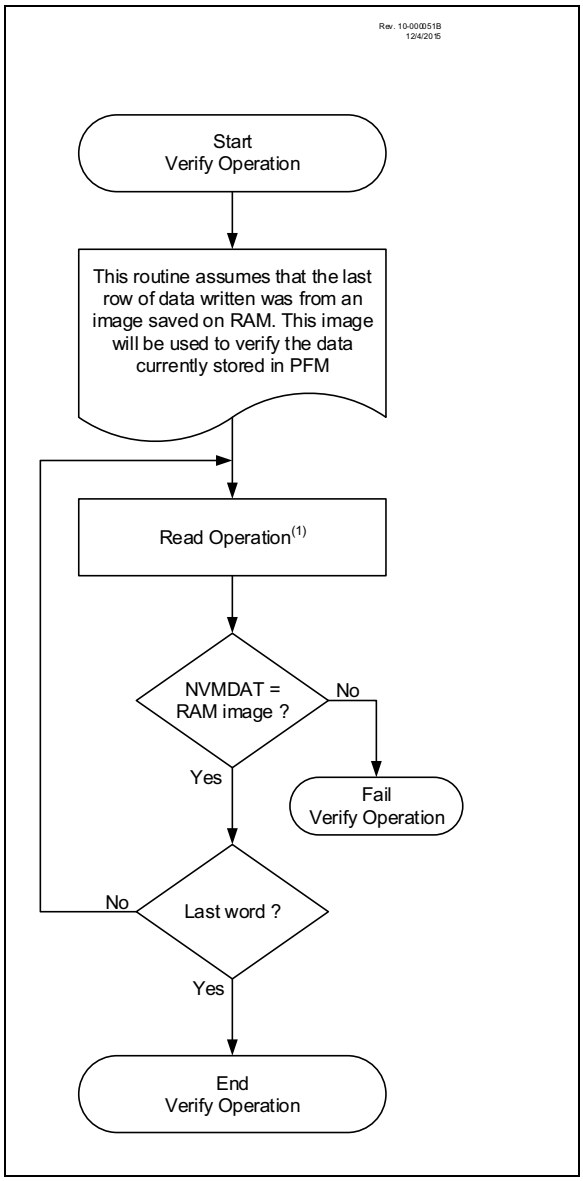
bit 1 **TMR1GIP:** TMR1 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Interrupt Priority bit
1 = High priority
0 = Low priority

13.1.6.2 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit. Since program memory is stored as a full page, the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

FIGURE 13-10: PROGRAM FLASH MEMORY VERIFY FLOWCHART



13.1.6.3 Unexpected Termination of Write Operation

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. If the write operation is interrupted by a MCLR Reset or a WDT Time-out Reset during normal operation, the WRERR bit will be set which the user can check to decide whether a rewrite of the location(s) is needed.

13.1.6.4 Protection Against Spurious Writes

A write sequence is valid only when both the following conditions are met, this prevents spurious writes which might lead to data corruption.

1. The WR bit is gated through the WREN bit. It is suggested to have the WREN bit cleared at all times except during memory writes. This prevents memory writes if the WR bit gets set accidentally.
2. The NVM unlock sequence must be performed each time before a write operation.

13.2 Device Information Area, Device Configuration Area, User ID, Device ID and Configuration Word Access

When REG<1:0> = 0b01 or 0b11 in the NVMCON1 register, the Device Information Area, the Device Configuration Area, the User IDs, Device ID/ Revision ID and Configuration Words can be accessed. Different access may exist for reads and writes (see Table 13-1).

13.2.1 Reading Access

The user can read from these blocks by setting the REG bits to 0b01 or 0b11. The user needs to load the address into the TBLPTR registers. Executing a TBLRD after that moves the byte pointed to the TABLAT register. The CPU operation is suspended during the read and resumes after. When read access is initiated on an address outside the parameters listed in Table 13-1, the TABLAT register is cleared, reading back ‘0’s.

13.2.2 Writing Access

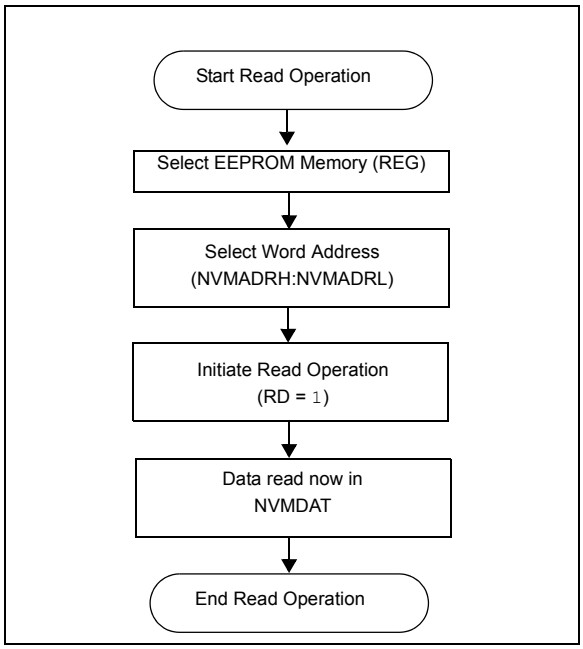
The WREN bit in NVMCON1 must be set to enable writes. This prevents accidental writes to the CONFIG words due to errant (unexpected) code execution. The WREN bit should be kept clear at all times, except when updating the CONFIG words. The WREN bit is not cleared by hardware. The WR bit will be inhibited from being set unless the WREN bit is set.

13.3.3 READING THE DATA EEPROM MEMORY

To read a data memory location, the user must write the address to the NVMADRL and NVMADRH register pair, clear REG<1:0> control bit in NVMCON1 register to access Data EEPROM locations and then set control bit, RD. The data is available on the very next instruction cycle; therefore, the NVMDAT register can be read by the next instruction. NVMDAT will hold this value until another read operation, or until it is written to by the user (during a write operation).

The basic process is shown in Example 13-5.

FIGURE 13-11: DATA EEPROM READ FLOWCHART



13.3.4 WRITING TO THE DATA EEPROM MEMORY

To write an EEPROM data location, the address must first be written to the NVMADRL and NVMADRH register pair and the data written to the NVMDAT register. The sequence in Example 13-6 must be followed to initiate the write cycle.

The write will not begin if NVM Unlock sequence, described in **Section 13.1.4 "NVM Unlock Sequence"**, is not exactly followed for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in NVMCON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution (i.e., runaway programs). The WREN bit should be kept clear at all times, except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, NVMCON1, NVMADRL, NVMADRH and NVMDAT cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. Both WR and WREN cannot be set with the same instruction.

After a write sequence has been initiated, clearing the WREN bit will not affect this write cycle. A single Data EEPROM word is written and the operation includes an implicit erase cycle for that word (it is not necessary to set FREE). CPU execution continues in parallel and at the completion of the write cycle, the WR bit is cleared in hardware and the NVM Interrupt Flag bit (NVMIF) is set. The user can either enable this interrupt or poll this bit. NVMIF must be cleared by software.

REGISTER 19-7: PMD6: PMD CONTROL REGISTER 6

U-0	U-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
—	—	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD
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 7-6 **Unimplemented:** Read as '0'

bit 5 **SMT1MD:** Disable SMT1 Module bit
1 = SMT1 module disabled
0 = SMT1 module enabled

bit 4 **CLC1MD:** Disable CLC4 Module bit
1 = CLC4 module disabled
0 = CLC4 module enabled

bit 3 **CLC3MD:** Disable CLC3 Module bit
1 = CLC3 module disabled
0 = CLC3 module enabled

bit 2 **CLC2MD:** Disable CLC2 Module bit
1 = CLC2 module disabled
0 = CLC2 module enabled

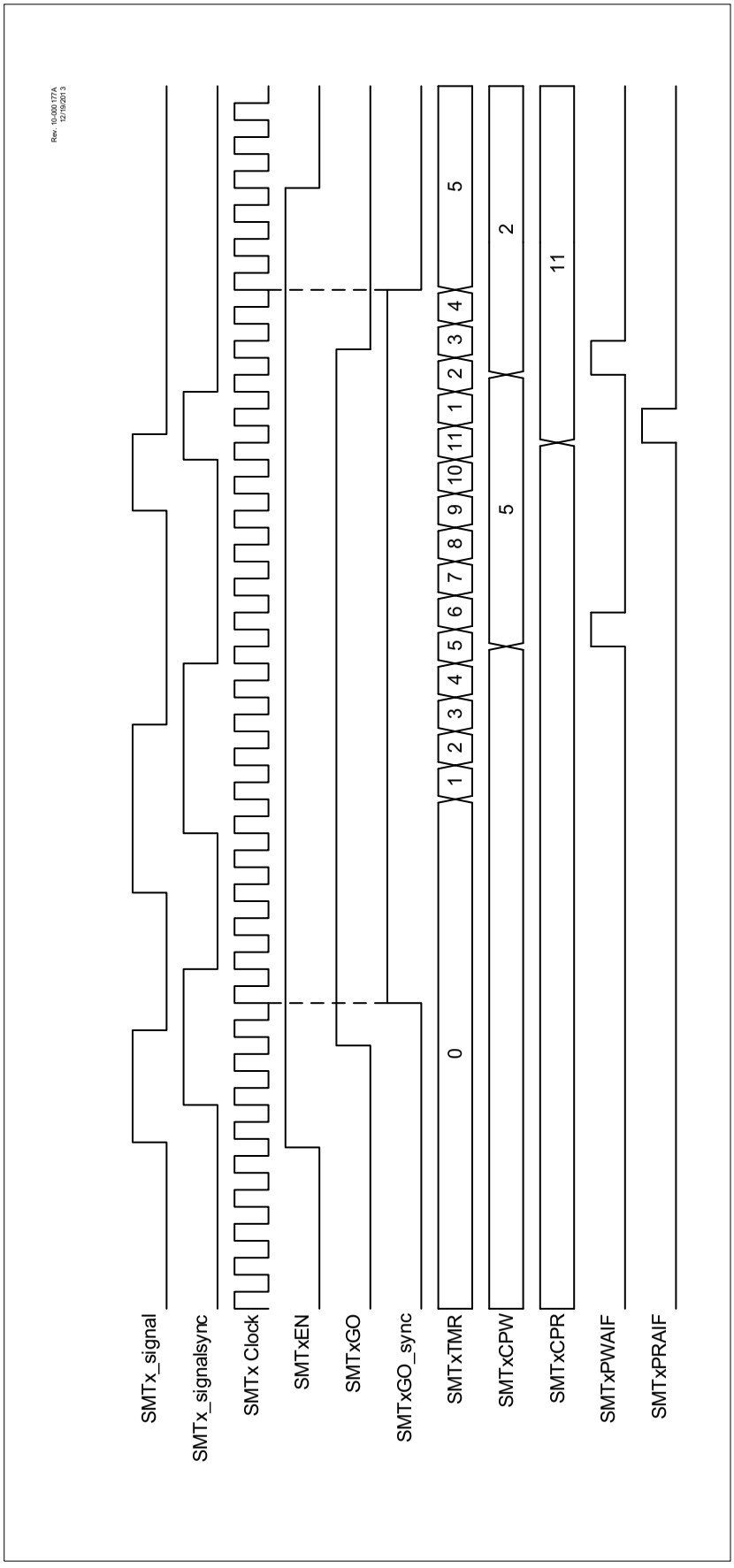
bit 1 **CLC1MD:** Disable CLC1 Module bit
1 = CLC1 module disabled
0 = CLC1 module enabled

bit 0 **DSMMD:** Disable Data Signal Modulator bit
1 = DSM module disabled
0 = DSM module enabled

25.6.2 GATED TIMER MODE

Gated Timer mode uses the SMTSIGx input to control whether or not the SMT1TMR will increment. Upon a falling edge of the external signal, the SMT1CPW register will update to the current value of the SMT1TMR. Example waveforms for both repeated and single acquisitions are provided in Figure 25-4 and Figure 25-5.

FIGURE 25-6: PERIOD AND DUTY-CYCLE REPEAT ACQUISITION MODE TIMING DIAGRAM



25.6.5 WINDOWED MEASURE MODE

This mode measures the window duration of the SMTWINx input of the SMT. It begins incrementing the timer on a rising edge of the SMTWINx input and updates the SMT1CPR register with the value of the timer and resets the timer on a second rising edge. See Figure 25-10 and Figure 25-11.

26.12 Operation During Sleep

The CWG module operates independently from the system clock and will continue to run during Sleep, provided that the clock and input sources selected remain active.

The HFINTOSC remains active during Sleep when all the following conditions are met:

- CWG module is enabled
- Input source is active
- HFINTOSC is selected as the clock source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as system clock and CWG clock, when the CWG is enabled and the input source is active, then the CPU will go idle during Sleep, but the HFINTOSC will remain active and the CWG will continue to operate. This will have a direct effect on the Sleep mode current.

26.13 Configuring the CWG

1. Ensure that the TRIS control bits corresponding to CWG outputs are set so that all are configured as inputs, ensuring that the outputs are inactive during setup. External hardware should ensure that pin levels are held to safe levels.
2. Clear the EN bit, if not already cleared.
3. Configure the MODE<2:0> bits of the CWGxCON0 register to set the output operating mode.
4. Configure the POLy bits of the CWGxCON1 register to set the output polarities.
5. Configure the ISM<4:0> bits of the CWGxISM register to select the data input source.
6. If a steering mode is selected, configure the STRx bits to select the desired output on the CWG outputs.
7. Configure the LSBD<1:0> and LSAC<1:0> bits of the CWGxASD0 register to select the auto-shutdown output override states (this is necessary even if not using auto-shutdown because start-up will be from a shutdown state).
8. If auto-restart is desired, set the REN bit of CWGxAS0.
9. If auto-shutdown is desired, configure the ASxE bits of the CWGxAS1 register to select the shutdown source.
10. Set the desired rising and falling dead-band times with the CWGxDBR and CWGxDBF registers.
11. Select the clock source in the CWGxCLKCON register.
12. Set the EN bit to enable the module.
13. Clear the TRIS bits that correspond to the CWG outputs to set them as outputs.

If auto-restart is to be used, set the REN bit and the SHUTDOWN bit will be cleared automatically. Otherwise, clear the SHUTDOWN bit in software to start the CWG.

28.1 NCO Operation

The NCO operates by repeatedly adding a fixed value to an accumulator. Additions occur at the input clock rate. The accumulator will overflow with a carry periodically, which is the raw NCO output (NCO_overflow). This effectively reduces the input clock by the ratio of the addition value to the maximum accumulator value. See Equation 28-1.

The NCO output can be further modified by stretching the pulse or toggling a flip-flop. The modified NCO output is then distributed internally to other peripherals and can be optionally output to a pin. The accumulator overflow also generates an interrupt (NCO_overflow).

The NCO period changes in discrete steps to create an average frequency. This output depends on the ability of the receiving circuit (i.e., CWG or external resonant converter circuitry) to average the NCO output to reduce uncertainty.

EQUATION 28-1: NCO OVERFLOW FREQUENCY

$$F_{OVERFLOW} = \frac{NCO\ Clock\ Frequency \times Increment\ Value}{2^{20}}$$

28.1.1 NCO CLOCK SOURCES

Clock sources available to the NCO include:

- FOSC
- HFINTOSC
- LFINTOSC
- MFINTOSC/4 (32 kHz)
- MFINTOSC (500 kHz)
- CLC1/2/3/4_out
- CLKREF
- SOSC

The NCO clock source is selected by configuring the N1CKS<2:0> bits in the NCO1CLK register.

28.1.2 ACCUMULATOR

The accumulator is a 20-bit register. Read and write access to the accumulator is available through three registers:

- NCO1ACCL
- NCO1ACCH
- NCO1ACCU

28.1.3 ADDER

The NCO Adder is a full adder, which operates independently from the source clock. The addition of the previous result and the increment value replaces the accumulator value on the rising edge of each input clock.

28.1.4 INCREMENT REGISTERS

The increment value is stored in three registers making up a 20-bit incrementer. In order of LSB to MSB they are:

- NCO1INCL
- NCO1INCH
- NCO1INCU

When the NCO module is enabled, the NCO1INCU and NCO1INCH registers should be written first, then the NCO1INCL register. Writing to the NCO1INCL register initiates the increment buffer registers to be loaded simultaneously on the second rising edge of the NCO_clk signal.

The registers are readable and writable. The increment registers are double-buffered to allow value changes to be made without first disabling the NCO module.

When the NCO module is disabled, the increment buffers are loaded immediately after a write to the increment registers.

Note: The increment buffer registers are not user-accessible.

31.17.1 AUTO-BAUD DETECT

The UART module supports automatic detection and calibration of the baud rate in the 8-bit asynchronous and LIN modes. However, setting ABDEN to start auto-baud detection is neither necessary, nor possible in LIN mode because that mode supports auto-baud detection automatically at the beginning of every data packet. Enabling auto-baud detect with the ABDEN bit applies to the asynchronous modes only.

Note:	In DALI Mode, ABDEN is ignored. The baud rate needs to be manually set to 1200 using the BRG registers.

When Auto-Baud Detect (ABD) is active, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RX signal, the RX signal is timing the BRG. The Baud Rate Generator is used to time the period of a received 55h (ASCII “U”), which is the Sync character for the LIN bus. The unique feature of this character is that it has five falling edges, including the Start bit edge, five rising edges including the Stop bit edge.

In 8-bit Asynchronous mode, setting the ABDEN bit in the UxCON0 register enables the auto-baud calibration sequence. The first falling edge of the RX input after ABDEN is set will start the auto-baud calibration sequence. While the ABD sequence takes place, the UART state machine is held in idle. On the first falling edge of the receive line, the UxBRG begins counting up using the BRG counter clock as shown in Figure 31-12. The fifth falling edge will occur on the RX pin at the beginning of the bit 7 period. At that time, an accumulated value totaling the proper BRG period is left in the UxBRGH, UxBRGL register pair, the ABDEN bit is automatically cleared and the ABDIF interrupt flag is set. ABDIF must be cleared by software.

RXIDL indicates that the sync input is active. RXIDL will go low on the first falling edge and go high on the fifth rising edge.

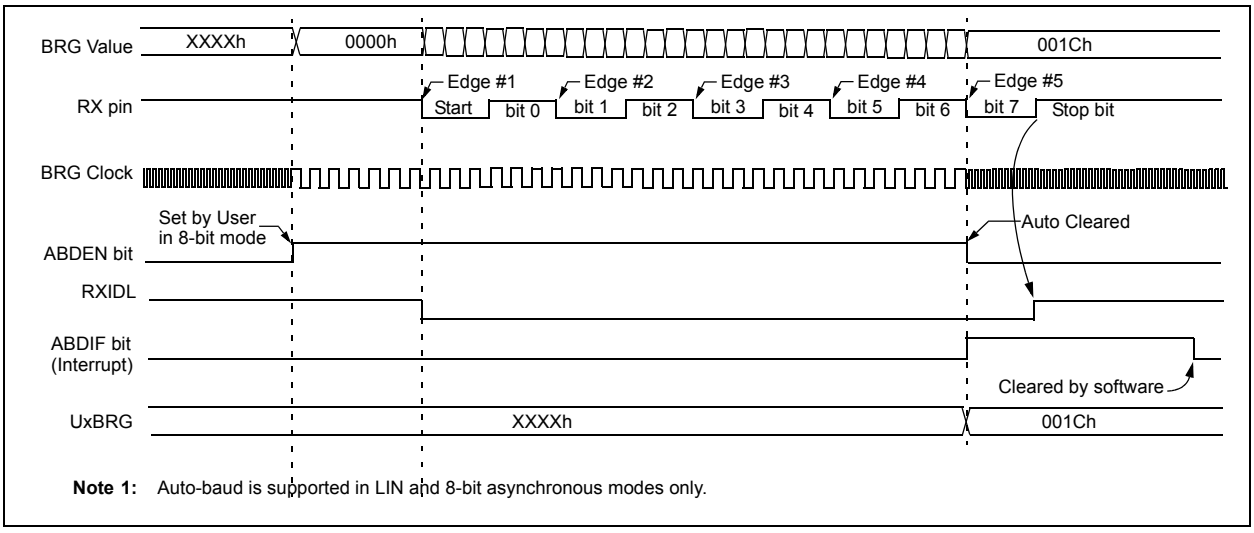
The BRG auto-baud clock is determined by the BRGS bit as shown in Table 31-2. During ABD, the internal BRG register is used as a 16-bit counter. However, the UxBRGH and UxBRGL registers retain the previous BRG value until the auto-baud process is successfully completed. While calibrating the baud rate period, the internal BRG register is clocked at 1/8th the BRG base clock rate. The resulting byte measurement is the average bit time when clocked at full speed and is transferred to the UxBRGH and UxBRGL registers when complete.

Note 1:	If the WUE bit is set with the ABDEN bit, auto-baud detection will occur on the byte following the Break character (see **Section 31.17.3 “Auto-Wake-up on Break”**).
2:	It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and UART baud rates are not possible.

TABLE 31-2: BRG COUNTER CLOCK RATES

BRGS	BRG Base Clock	BRG ABD Clock
1	FOSC/4	FOSC/32
0	FOSC/16	FOSC/128

FIGURE 31-12: AUTOMATIC BAUD RATE CALIBRATION



Note 1: Auto-baud is supported in LIN and 8-bit asynchronous modes only.

32.4 Transfer Counter

In all master modes, the transfer counter can be used to determine how many data transfers the SPI will send/receive. The transfer counter is comprised of the SPIxTCTH/L set of registers, and is also partially controlled by the SPIxTWIDTH register. The Transfer Counter has two primary modes, determined by the BMODE bit of the SPIxCON0 register. Each mode uses the SPIxTCTH/L and SPIxTWIDTH registers to determine the number and size of the transfers. In both modes, when the transfer counter reaches zero, the TCZIF interrupt flag is set.

Note:	When BMODE=1 in all master modes (and at all times in slave modes), the Transfer Counter will still decrement as transfers occur and can be used to count the number of messages sent/received, as well as to control SS(out) and to trigger TCZIF. Also when BMODE = 1, the SPIxTWIDTH register can be used in Master and Slave modes to determine the size of messages sent and received by the SPI, even if the Transfer Counter is not being actively used to control the number of messages being sent/received by the SPI module.

32.4.1 TOTAL BIT COUNT MODE (BMODE = 0)

In this mode, SPIxTCTH/L and SPIxTWIDTH are concatenated to determine the total number of bits to be transferred. These bits will be loaded from/into the transmit/receive FIFOs in 8-bit increments and the transfer counter will be decremented by eight until the total number of remaining bits is less than eight. If there are any remaining bits (SPIxTWIDTH ≠ 0), the transmit FIFO will send out one final message with any extra bits greater than the remainder ignored. The SPIxTWIDTH is the remaining bit count but the value does not change as it does for the SPIxTCT value. Similarly, the receiver will load a final byte into the receiver FIFO, and pad the extra bits with zeros. The LSBF bit of SPIxCON0 determines whether the Most Significant or Least Significant bits of this final byte are ignored/padded. For example, when LSBF = 0 and the final transfer contains only two bits then if the last byte sent was 5Fh then the RXB of the receiver will contain 40h which are the two MSbits of the final byte padded with zeros in the LSbits.

In this mode, the SPI master will only transmit messages when the SPIxTCT value is greater than zero, regardless of TXR and RXR settings. In Master Transmit mode, the transfer starts with the data write to the SPIxTXB register or the count value written to the SPIxTCTL register, which ever occurs last. In Master Receive-only mode, the transfer clocks start when the SPIxTCTL value is written. Transfer clocks are suspended when the receive FIFO is full and resume as the FIFO is read.

32.4.2 VARIABLE TRANSFER SIZE MODE (BMODE = 1)

In this mode, SPIxTWIDTH specifies the width of every individual piece of the data transfer in bits. SPIxTCTH/SPIxTCTL specifies the number of transfers of this bit length. If SPIxTWIDTH = 0, each piece is a full byte of data. If SPIxTWIDTH ≠ 0, then only the specified number of bits from the transmit FIFO are shifted out, with the unused bits ignored. Received data is padded with zeros in the unused bit areas when transfered into the receive FIFO. The LSBF bit of SPIxCON0 determines whether the Most Significant or Least Significant bits of the transfers are ignored/padded. In this mode, the transfer counter being zero only stops messages from being sent/received when in “Receive only” mode.

Note:	With BMODE = 1, it is possible for the transfer counter (SPIxTCTH/L) to decrement below zero, although when in “Receive only” Master mode, transfer clocks will cease when the transfer counter reaches zero.

FIGURE 32-9: CLOCKING DETAIL-MASTER MODE, CKE = 0, SMP = 1

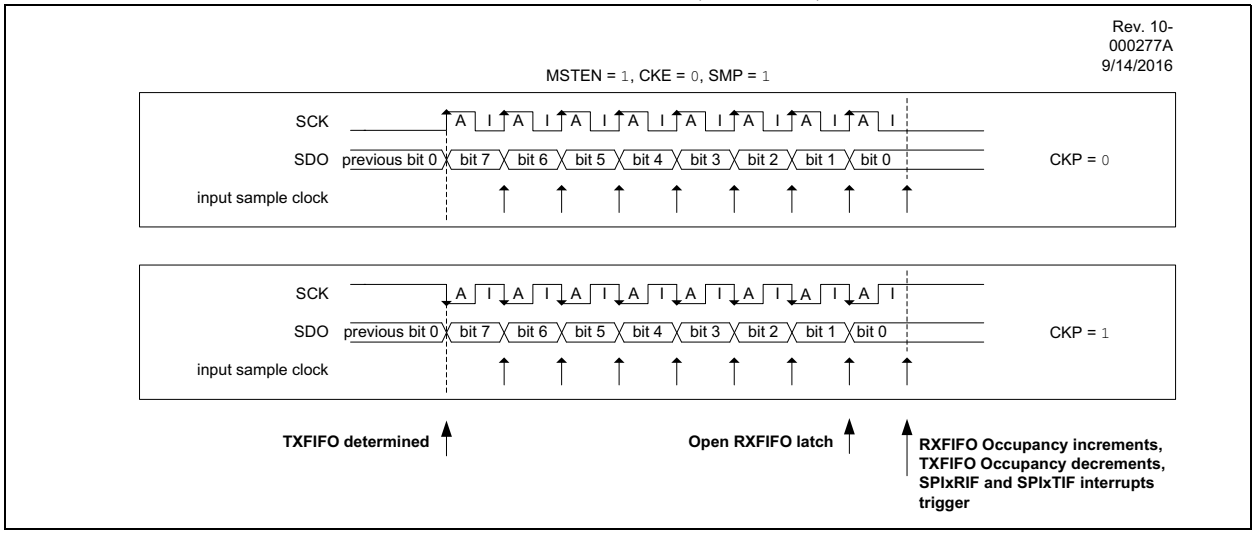
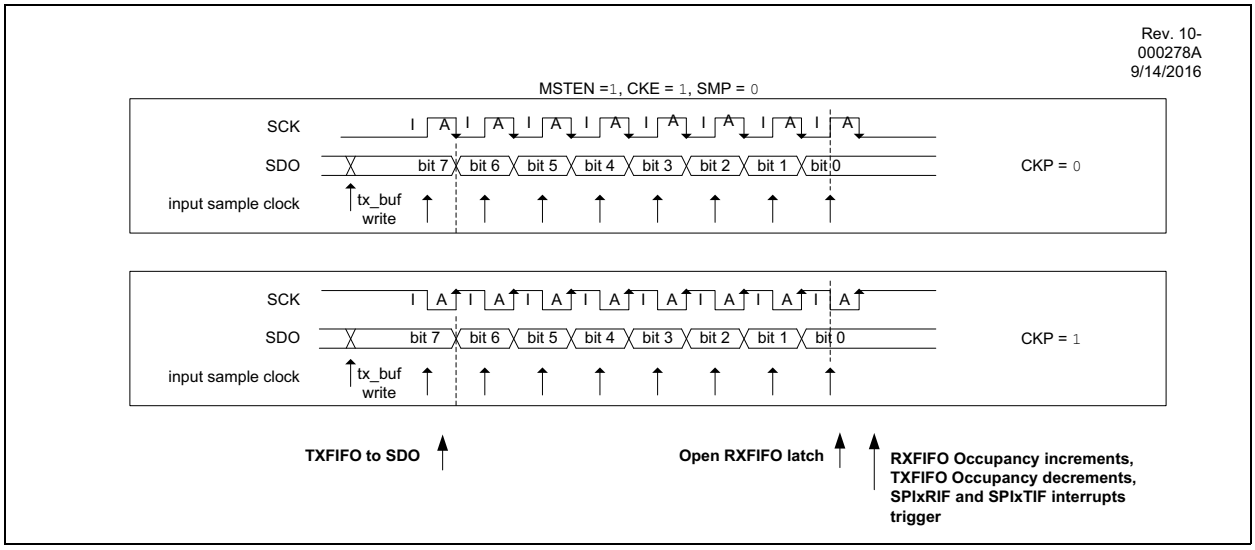


FIGURE 32-10: CLOCKING DETAIL-MASTER MODE, CKE = 1, SMP = 0



32.5.6.3 SCK Start-Up Delay

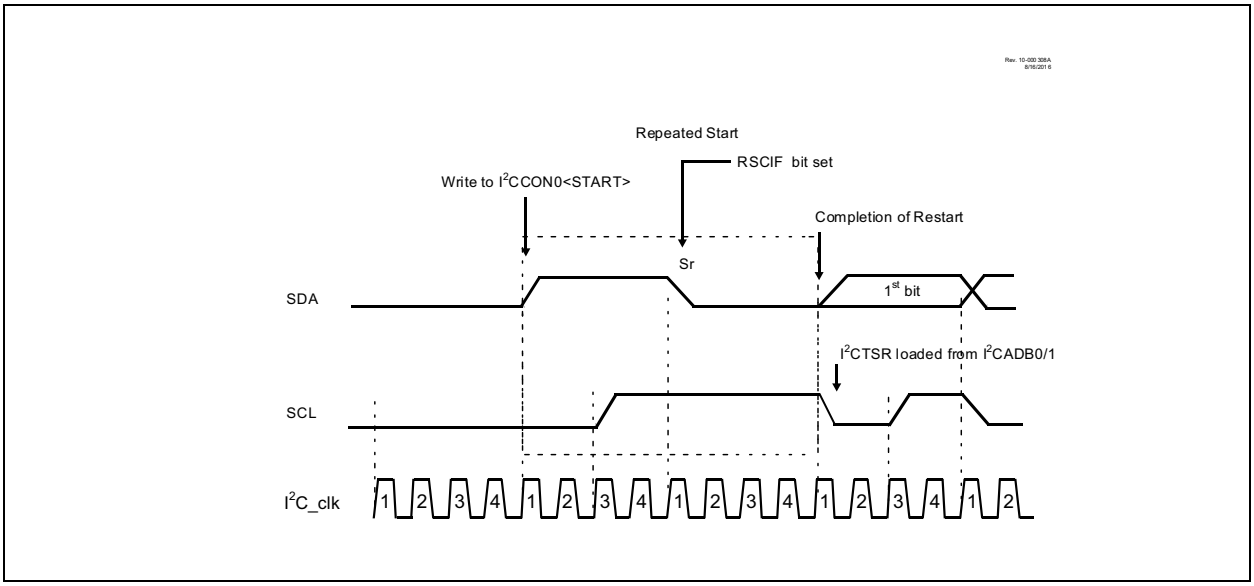
When starting an SPI data exchange, the master device sets the SS output (either through hardware or software) and then triggers the module to send data. These data triggers are synchronized to the clock selected by the SPIxCLK register before the first SCK pulse appears, usually requiring one or two clocks of the selected clock.

The SPI module includes synchronization delays on SCK generation specifically designed to ensure that the Slave Select output timing is correct, without requiring precision software timing loops.

When the value of the SPIxBAUD register is a small number (indicating higher SCK frequencies), the synchronization delay can be relatively long between setting SS and the first SCK. With larger values of SPIxBAUD (indicating lower SCK frequencies), this delay is much smaller and the first SCK can appear relatively quickly after SS is set.

By default, the SPI module inserts a ½ baud delay (half of the period of the clock selected by the SPIxCLK register) before the first SCK pulse. This allows for systems with a high SPIxBAUD value to have extra set-up time before the first clock. Setting the FST bit in SPIxCON1 removes this additional delay, allowing systems with low SPIxBAUD values (and thus, long synchronization delays) to forego this unnecessary extra delay.

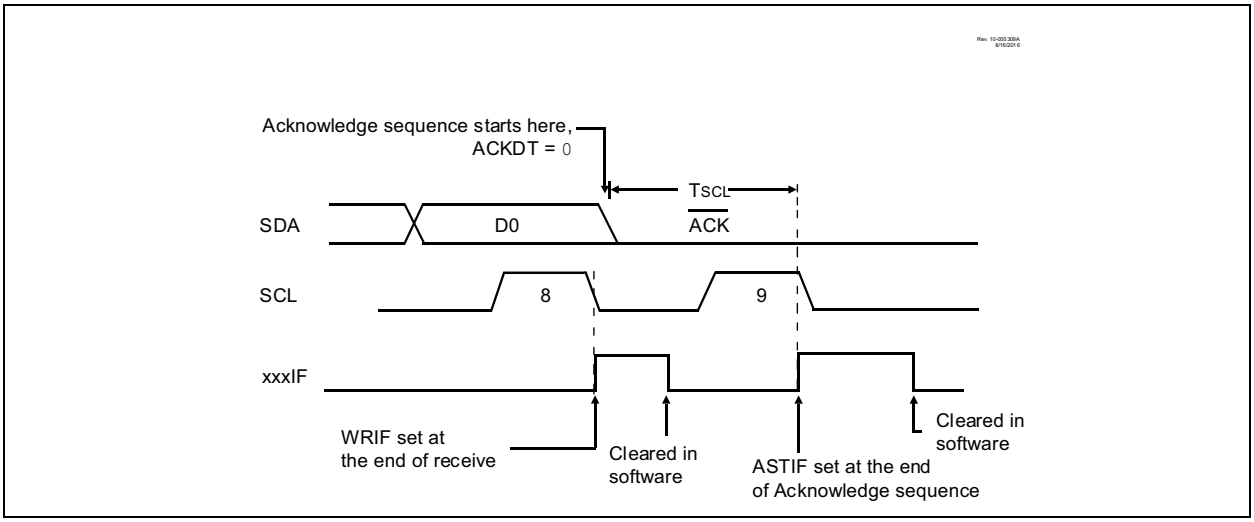
FIGURE 33-16: REPEATED START CONDITION TIMING



33.5.7 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled automatically following an address/data byte transmission. The SCL pin is pulled low and the contents of the Acknowledge Data bits (ACKDT/ACKCNT) are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The master then waits one clock period (TSCL) and the SCL pin is released high. When the SCL pin is sampled high (clock arbitration), the master counts another TSCL. The SCL pin is then pulled low. Figure 33-17 shows the timings for Acknowledge sequence.

FIGURE 33-17: ACKNOWLEDGE SEQUENCE TIMING



33.5.8 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of receive/transmit when I2CxCNT = 0. After the last byte of a receive/transmit sequence, the SCL line is held low. The master asserts the SDA line low. The SCL pin is then released high TSCL/2 later and is detected high. The SDA pin is then released. When the SDA pin transitions high while SCL is high, the PCIF bit of the I2CxIF register is set. Figure 33-18 shows the timings for a Stop condition.

FIGURE 39-2: LOW-VOLTAGE DETECT OPERATION (INTL = 1)

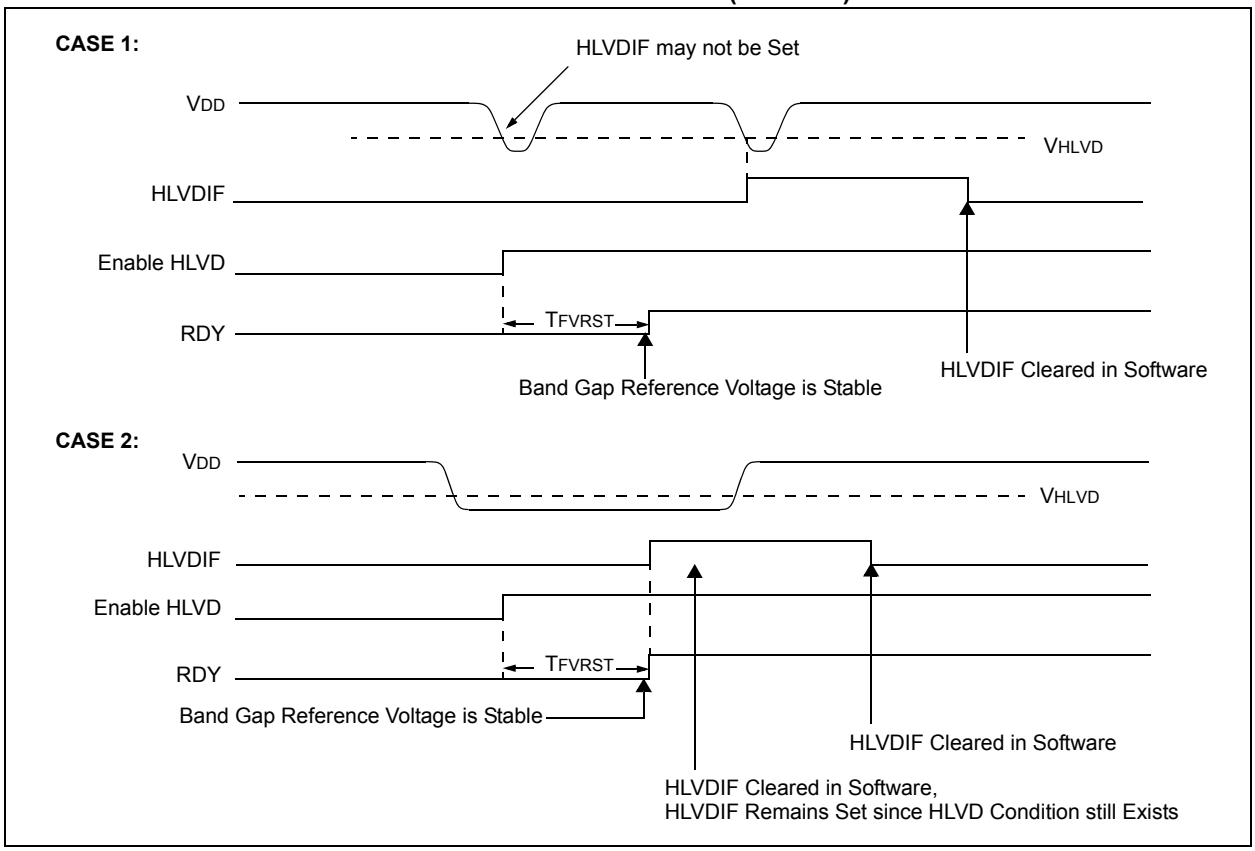
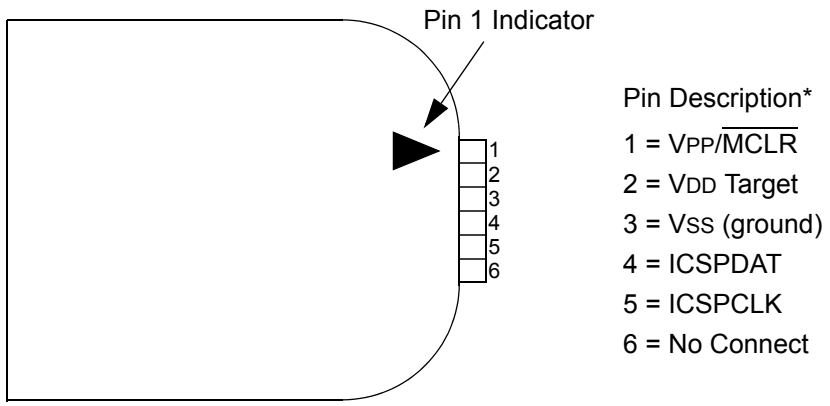
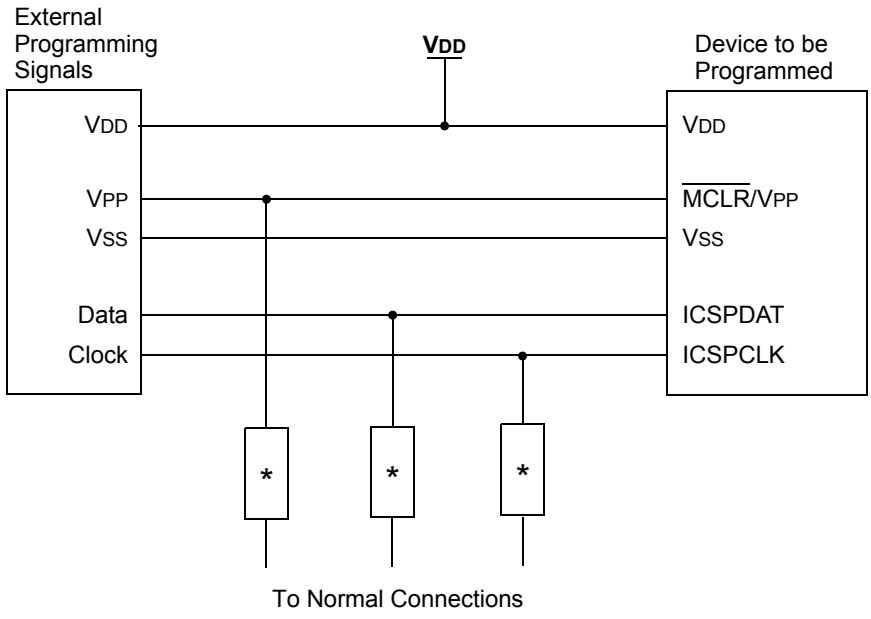


FIGURE 40-2: PICkit™ PROGRAMMER STYLE CONNECTOR INTERFACE



* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

FIGURE 40-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING



* Isolation devices (as required).

TABLE 41-1: OPCODE FIELD DESCRIPTIONS (CONTINUED)

Field	Description
< >	Register bit field
∈	In the set of
italics	User defined term (font is courier)

FIGURE 41-1: General Format for Instructions (1/2)

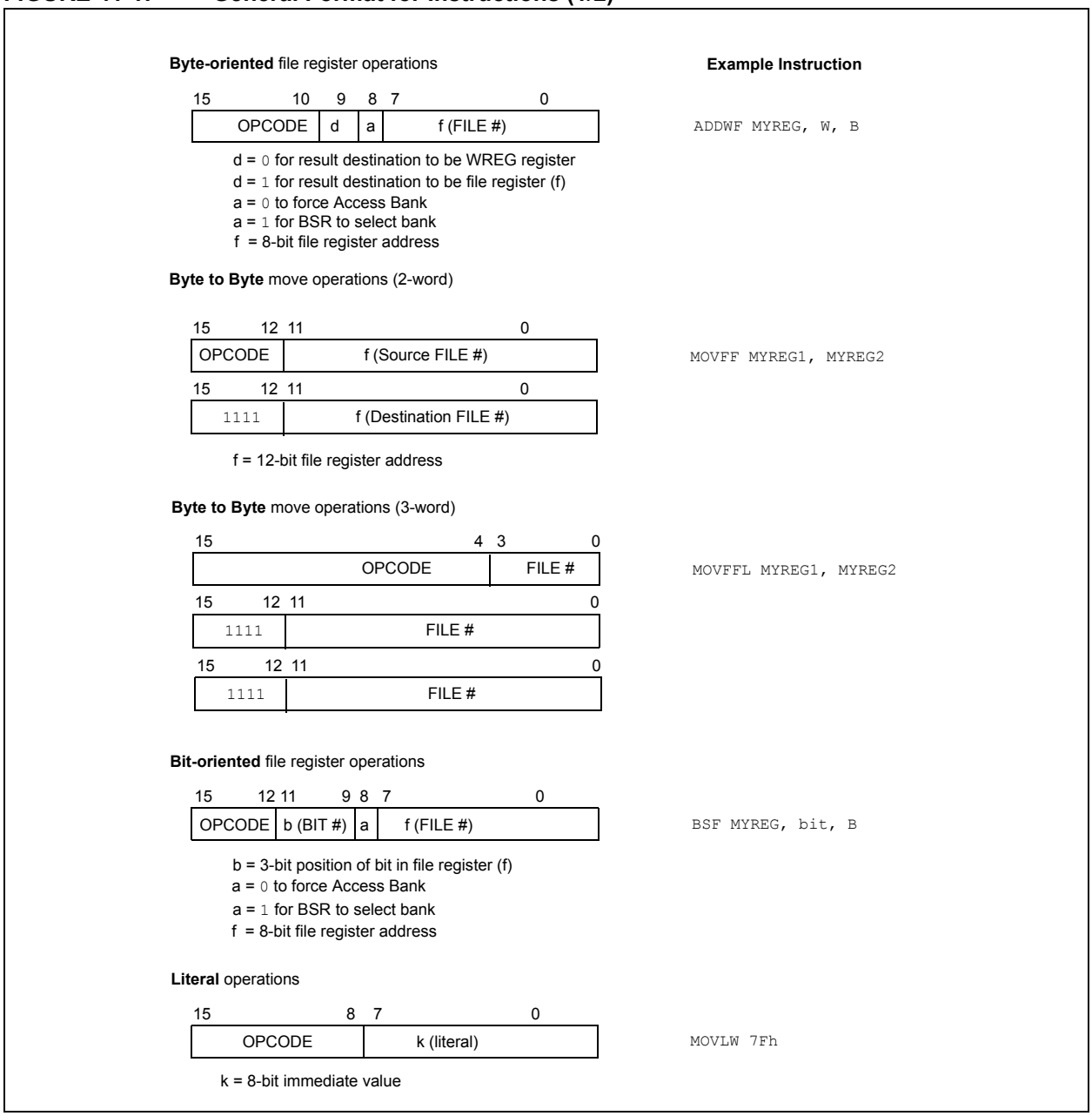


TABLE 41-2: INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb	LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER INSTRUCTIONS							
ADDWF	f, d ,a	Add WREG and f	1	0010 01da	ffff ffff	C, DC, Z, OV, N	
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010 00da	ffff ffff	C, DC, Z, OV, N	
ANDWF	f, d, a	AND WREG with f	1	0001 01da	ffff ffff	Z, N	
CLRF	f, a	Clear f	1	0110 101a	ffff ffff	Z	
COMF	f, d, a	Complement f	1	0001 11da	ffff ffff	Z, N	
DECF	f, d, a	Decrement f	1	0000 01da	ffff ffff	C, DC, Z, OV, N	
INCF	f, d, a	Increment f	1	0010 10da	ffff ffff	C, DC, Z, OV, N	
IORWF	f, d, a	Inclusive OR WREG with f	1	0001 00da	ffff ffff	Z, N	
MOVF	f, d, a	Move f to WREG or f	1	0101 00da	ffff ffff	Z, N	
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 ffff 1111 ffff	ffff ffff ffff ffff	None	2, 3
MOVFFL	f_s, f_d	Move f_s (source) to g (full destination) f_d (full destination)3rd word	3	0000 0000 1111 ffff 1111 gggg	0110 ffff ffgg gggg gggg	None	2
MOVWF	f, a	Move WREG to f	1	0110 111a	ffff ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000 001a	ffff ffff	None	
NEGF	f, a	Negate f	1	0110 110a	ffff ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011 01da	ffff ffff	C, Z, N	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100 01da	ffff ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011 00da	ffff ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100 00da	ffff ffff	Z, N	
SETF	f, a	Set f	1	0110 100a	ffff ffff	None	
SUBFWB	f, d, a	Subtract f from WREG with borrow	1	0101 01da	ffff ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101 11da	ffff ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Subtract WREG from f with borrow	1	0101 10da	ffff ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap nibbles in f	1	0011 10da	ffff ffff	None	
XORWF	f, d, a	Exclusive OR WREG with f	1	0001 10da	ffff ffff	Z, N	
BYTE-ORIENTED SKIP INSTRUCTIONS							
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110 001a	ffff ffff	None	1
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110 010a	ffff ffff	None	1
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110 000a	ffff ffff	None	1
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da	ffff ffff	None	1
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da	ffff ffff	None	1
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da	ffff ffff	None	1
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da	ffff ffff	None	1
TSTFSZ	f, a	Test f, skip if 0	1 (2 or 3)	0110 011a	ffff ffff	None	1
BIT-ORIENTED FILE REGISTER INSTRUCTIONS							
BCF	f, b, a	Bit Clear f	1	1001 bbba	ffff ffff	None	
BSF	f, b, a	Bit Set f	1	1000 bbba	ffff ffff	None	
BTG	f, d, a	Bit Toggle f	1	0111 bbba	ffff ffff	None	
BIT-ORIENTED SKIP INSTRUCTIONS							
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011 bbba	ffff ffff	None	1
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010 bbba	ffff ffff	None	1

Note 1: If Program Counter (PC) is modified or a conditional test is true, the instruction requires an additional cycle. The extra cycle is executed as a NOP.

2: Some instructions are multi word instructions. The second/third words of these instructions will be decoded 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.

3: f_s and f_d do not cover the full memory range. 2 MSBs of bank selection are forced to ‘b00 to limit the range of these instructions to lower 4k addressing space.

SUBWF — Subtract W from f

Syntax: SUBWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – (W) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored 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 $f \le 95$ (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 1:
```
SUBWF   REG, 1, 0
```
Before Instruction
REG = 3
W = 2
C = ?
After Instruction
REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2:
```
SUBWF   REG, 0, 0
```
Before Instruction
REG = 2
W = 2
C = ?
After Instruction
REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3:
```
SUBWF   REG, 1, 0
```
Before Instruction
REG = 1
W = 2
C = ?
After Instruction
REG = FFh ;(2's complement)
W = 2
C = 0 ; result is negative
Z = 0
N = 1

SUBWFB — Subtract W from f with Borrow

Syntax: SUBWFB f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – (W) – $(\overline{C})$ → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	10da	ffff	ffff

Description: Subtract W and the CARRY flag (borrow) from register 'f' (2's complement method). If 'd' is '0', the result is stored 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 $f \le 95$ (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 1:
```
SUBWFB  REG, 1, 0
```
Before Instruction
REG = 19h (0001 1001)
W = 0Dh (0000 1101)
C = 1
After Instruction
REG = 0Ch (0000 1100)
W = 0Dh (0000 1101)
C = 1
Z = 0
N = 0 ; result is positive

Example 2:
```
SUBWFB  REG, 0, 0
```
Before Instruction
REG = 1Bh (0001 1011)
W = 1Ah (0001 1010)
C = 0
After Instruction
REG = 1Bh (0001 1011)
W = 00h
C = 1
Z = 1 ; result is zero
N = 0

Example 3:
```
SUBWFB  REG, 1, 0
```
Before Instruction
REG = 03h (0000 0011)
W = 0Eh (0000 1110)
C = 1
After Instruction
REG = F5h (1111 0101)
; [2's comp]
W = 0Eh (0000 1110)
C = 0
Z = 0
N = 1 ; result is negative