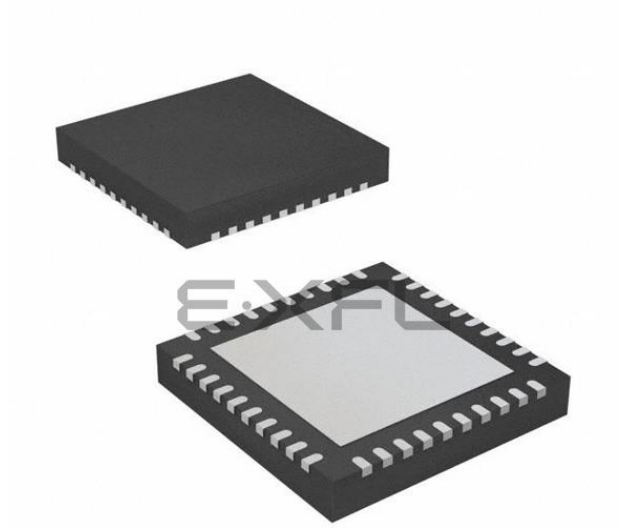


Welcome to E-XFL.COM

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	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°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/pic18f45k42t-i-mv

4.4.2 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 4.2.4 "Program Counter"**).

Figure 4-2 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 4-2 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 41.0 "Instruction Set Summary"** provides further details of the instruction set.

4.4.3 MULTI-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LFSR and two three-word instructions: MOVFFL and MOVSFL. In all cases, the second and the third 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 four 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 or third word is executed by itself, a NOP is executed instead. This is necessary for cases when the multi-word instruction is preceded by a conditional instruction that changes the PC. Example 4-4 shows how this works.

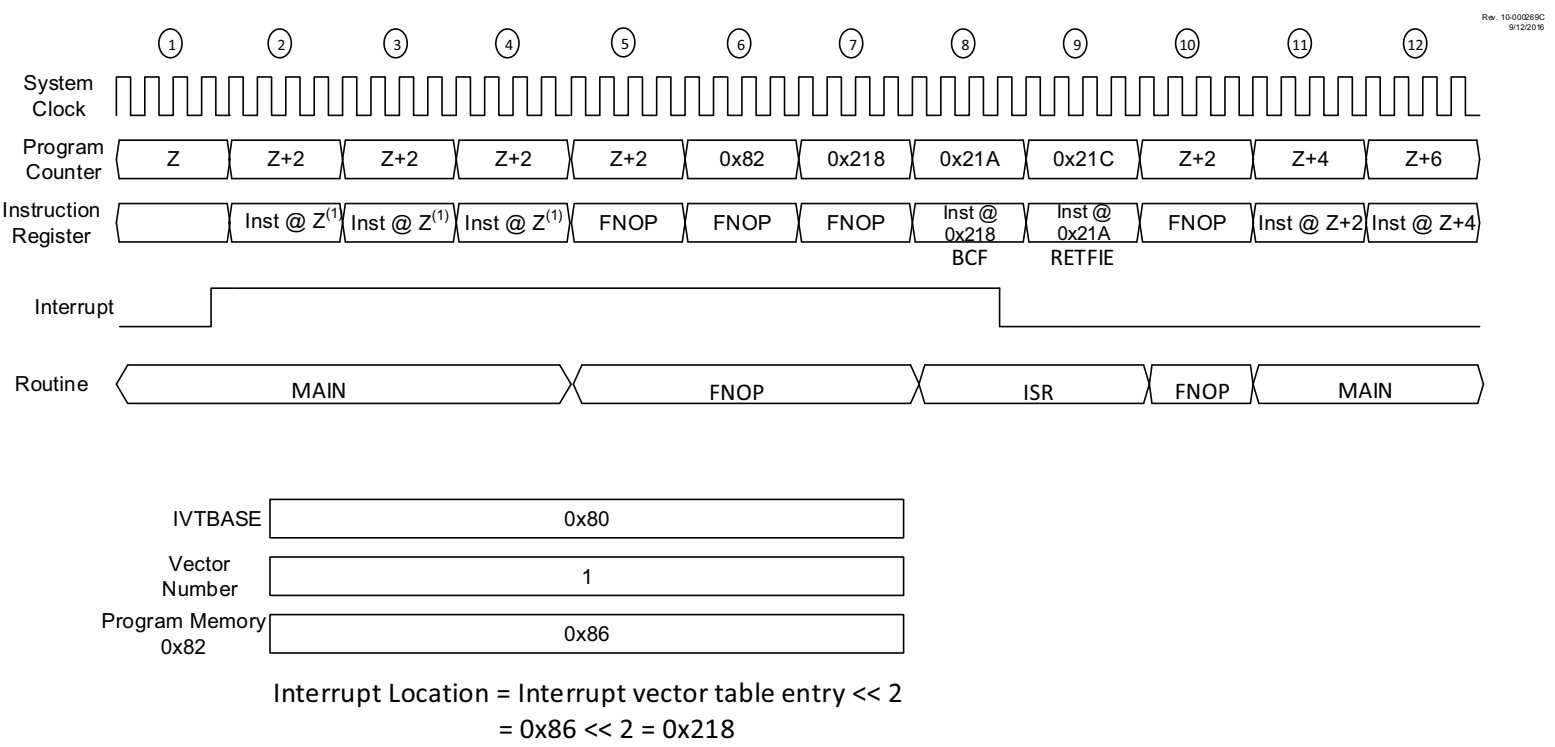
FIGURE 4-2: INSTRUCTIONS IN PROGRAM MEMORY

Program Memory Byte Locations →

Instruction 1: MOVLW 055h
Instruction 2: GOTO 0006h
Instruction 3: MOVFF 123h, 456h
Instruction 4: MOVFFL 123h, 456h

LSB = 1	LSB = 0	Word Address ↓
		000000h
		000002h
		000004h
		000006h
0Fh	55h	000008h
EFh	03h	00000Ah
F0h	00h	00000Ch
C1h	23h	00000Eh
F4h	56h	000010h
00h	60h	000012h
F4h	8Ch	000014h
F4h	56h	000016h
		000018h
		00001Ah

FIGURE 9-9: INTERRUPT TIMING DIAGRAM - THREE CYCLE INSTRUCTION



Note 1: Instruction @ Z is a Three-cycle instruction.

© 2016-2017 Microchip Technology Inc.

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

REGISTER 11-3: WDTPSL: WWDT PRESCALE SELECT LOW BYTE REGISTER (READ-ONLY)

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<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 **PSCNT<7:0>:** Prescale Select Low Byte bits(1)

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should not be read during normal operation.

REGISTER 11-4: WDTPSH: WWDT PRESCALE SELECT HIGH BYTE REGISTER (READ-ONLY)

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<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 **PSCNT<15:8>:** Prescale Select High Byte bits(1)

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should not be read during normal operation.

13.4 Register Definitions: Nonvolatile Memory

REGISTER 13-1: NVMCON1: NONVOLATILE MEMORY CONTROL 1 REGISTER

R/W-0/0	R/W-0/0	U-0	R/S/HC-0/0	R/W/HS-x/q	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
REG<1:0>		—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

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

bit 7-6 **REG<1:0>:** NVM Region Selection bit
10 =Access PFM Locations
x1 = Access User IDs, Configuration Bits, DIA, DCI, Rev ID and Device ID
00 = Access Data EEPROM Memory Locations

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

bit 4 **FREE:** Program Flash Memory Erase Enable bit(1)
1 = Performs an erase operation on the next WR command
0 = The next WR command performs a write operation

bit 3 **WRERR:** Write-Reset Error Flag bit(2,3,4)
1 = A write operation was interrupted by a Reset (hardware set),
or WR was written to 0b1 when an invalid address is accessed (Table 9-1, Table 13-1)
or WR was written to 0b1 when REG<1:0> and address do not point to the same region
or WR was written to 0b1 when a write-protected address is accessed (Table 9-2).
0 = All write operations have completed normally

bit 2 **WREN:** Program/Erase Enable bit
1 = Allows program/erase and refresh cycles
0 = Inhibits programming/erasing and user refresh of NVM

bit 1 **WR:** Write Control bit(5,6,7)
When REG points to a Data EEPROM Memory location:
1 = Initiates an erase/program cycle at the corresponding Data EEPROM Memory location
When REG points to a PFM location:
1 = Initiates the PFM write operation with data from the holding registers
0 = NVM program/erase operation is complete and inactive

bit 0 **RD:** Read Control bit(8)
1 = Initiates a read at address pointed by REG and NVMADR, and loads data into NVMDAT
0 = NVM read operation is complete and inactive

Note 1: This can only be used with PFM.
2: This bit is set when WR = 1 and clears when the internal programming timer expires or the write is completed successfully.
3: Bit must be cleared by the user; hardware will not clear this bit.
4: Bit may be written to '1' by the user in order to implement test sequences.
5: This bit can only be set by following the unlock sequence of **Section 13.1.4 "NVM Unlock Sequence"**.
6: Operations are self-timed and the WR bit is cleared by hardware when complete.
7: Once a write operation is initiated, setting this bit to zero will have no effect.
8: The bit can only be set in software. The bit is cleared by hardware when the operation is complete.

REGISTER 14-9: CRCXORH: CRC XOR HIGH BYTE REGISTER

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
X<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 **X<15:8>**: XOR of Polynomial Term X^n Enable bits

REGISTER 14-10: CRCXORL: CRC XOR LOW BYTE REGISTER

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	U-1
X<7:1>							—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit
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 7-1 **X<7:1>**: XOR of Polynomial Term X^n Enable bits

bit 0 **Unimplemented:** Read as ‘1’

REGISTER 15-3: DMAxBUF: DMAx DATA BUFFER REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
BUF7	BUF6	BUF5	BUF4	BUF3	BUF2	BUF1	BUF0
bit 7							bit 0

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

bit 7-0 **BUF<7:0>:** DMA Internal Data Buffer bits
DMABUF<7:0>
These bits reflect the content of the internal data buffer the DMA peripheral uses to hold the data being moved from the source to destination.

REGISTER 15-4: DMAxSSAL: DMAx SOURCE START ADDRESS LOW 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
SSA<7:0>							
bit 7							bit 0

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

bit 7-0 **SSA<7:0>:** Source Start Address bits

REGISTER 15-5: DMAxSSAH: DMAx SOURCE START ADDRESS HIGH 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
SSA<15:8>							
bit 7							bit 0

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

bit 7-0 **SSA<15:8>:** Source Start Address bits

TABLE 15-3: SUMMARY OF REGISTERS ASSOCIATED WITH DMA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
DMAxCON0	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	248
DMAxCON1	DMODE<1:0>		DSTP	SMR<1:0>		SMODE<1:0>		SSTP	249
DMAxBUF	DBUF7	DBUF6	DBUF5	DBUF4	DBUF3	DBUF2	DBUF1	DBUF0	250
DMAxSSAL	SSA<7:0>								250
DMAxSSAH	SSA<15:8>								250
DMAxSSAU	—	—	SSA<21:16>						251
DMAxSPTRL	SPTR<7:0>								251
DMAxSPTRH	SPTR<15:8>								251
DMAxSPTRU	—	—	SPTR<21:16>						252
DMAxSSZL	SSZ<7:0>								252
DMAxSSZH	—	—	—	—	SSZ<11:8>				252
DMAxSCNTL	SCNT<7:0>								253
DMAxSCNTH	—	—	—	—	SCNT<11:8>				253
DMAxDSAL	DSA<7:0>								253
DMAxDSAH	DSA<15:8>								254
DMAxDPTRL	DPTR<7:0>								254
DMAxDPTRH	DPTR<15:8>								254
DMAxDSZL	DSZ<7:0>								255
DMAxDSZH	—	—	—	—	DSZ<11:8>				255
DMAxDCNTL	DCNT<7:0>								255
DMAxDCNTH	—	—	—	—	DCNT<11:8>				256
DMAxSIRQ	—	SIRQ<6:0>							256
DMAxAIRQ	—	AIRQ<6:0>							256

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by DMA.

EXAMPLE 16-2: INITIALIZING PORTE

```
CLRF    PORTE       ;Initialize PORTE by
                    ;clearing output
                    ;data latches
CLRF    LATE        ;Alternate method
                    ;to clear output
                    ;data latches
CLRF    ANSELE      ;Configure analog pins
                    ;for digital only
MOVLW   05h         ;Value used to
                    ;initialize data
                    ;direction
MOVWF   TRISE       ;Set RE<0> as input
                    ;RE<1> as output
                    ;RE<2> as input
```

16.3.2 PORTE ON 28-PIN DEVICES

For PIC18(L)F26/27K42 devices, PORTE is only available when Master Clear functionality is disabled (MCLRE = 0). In this case, PORTE is a single bit, input-only port comprised of RE3 only. The pin operates as previously described. RE3 in PORTE register is a read-only bit and will read '1' when MCLRE = 1 (i.e., Master Clear enabled).

16.3.3 RE3 WEAK PULL-UP

The port RE3 pin has an individually controlled weak internal pull-up. When set, the WPUE3 bit enables the RE3 pin pull-up. When the RE3 port pin is configured as $\overline{\text{MCLR}}$, (CONFIG2L, MCLRE = 1 and CONFIG4H, LVP = 0), or configured for Low-Voltage Programming, (MCLRE = x and LVP = 1), the pull-up is always enabled and the WPUE3 bit has no effect.

16.3.4 INTERRUPT-ON-CHANGE

The interrupt-on-change feature is available only on the RE3 pin of PORTE for all devices. If MCLRE = 1 or LVP = 1, RE3 port functionality is disabled and interrupt-on-change on RE3 is not available. For further details refer to **Section 18.0 "Interrupt-on-Change"**.

18.0 INTERRUPT-ON-CHANGE

PORTA, PORTB, PORTC and pin RE3 of PORTE can be configured to operate as Interrupt-on-Change (IOC) pins on PIC18(L)F26/27/45/46/47/55/56/57K42 family devices. An interrupt can be generated by detecting a signal that has either a rising edge or a falling edge. Any individual port pin, or combination of port pins, can be configured to generate an interrupt. The interrupt-on-change module has the following features:

- Interrupt-on-Change enable (Master Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

Figure 18-1 is a block diagram of the IOC module.

18.1 Enabling the Module

To allow individual port pins to generate an interrupt, the IOCIE bit of the PIEx register must be set. If the IOCIE bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

18.2 Individual Pin Configuration

For each port pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

18.3 Interrupt Flags

The IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits located in the IOCAF, IOCBF, IOCCF and IOCEF registers respectively, are status flags that correspond to the interrupt-on-change pins of the associated port. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the PIR0 register reflects the status of all IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits.

18.4 Clearing Interrupt Flags

The individual status flags, (IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits), can be cleared by resetting them to zero. If another edge is detected during this clearing operation, the associated status flag will be set at the end of the sequence, regardless of the value actually being written.

In order to ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits should be performed. The following sequence is an example of what should be performed.

EXAMPLE 18-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

```
MOVLW   0xff
XORWF   IOCAF, W
ANDWF   IOCAF, F
```

18.5 Operation in Sleep

The interrupt-on-change interrupt sequence will wake the device from Sleep mode, if the IOCIE bit is set.

If an edge is detected while in Sleep mode, the IOCxF register will be updated prior to the first instruction executed out of Sleep.

24.0 PULSE-WIDTH MODULATION (PWM)

The PWM module generates a pulse-width modulated signal determined by the duty cycle, period, and resolution that are configured by the following registers:

- TxPR
- TxCON
- PWMxDCH
- PWMxDCL
- PWMxCON

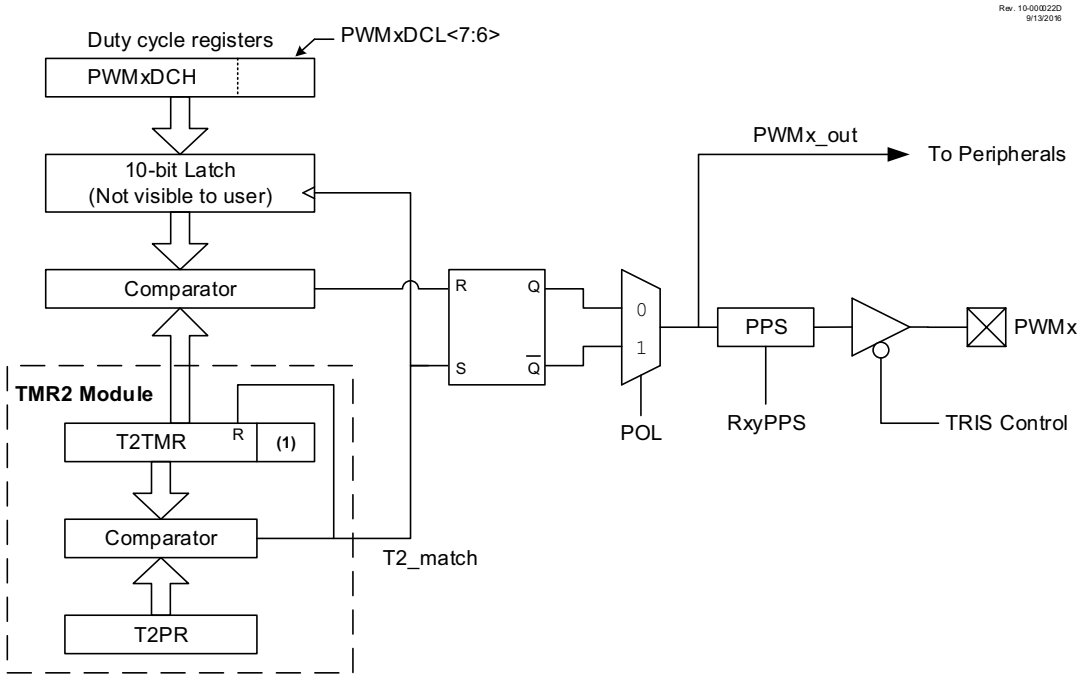
Note: The corresponding TRIS bit must be cleared to enable the PWM output on the PWMx pin.

Each PWM module can select the timer source that controls the module. Each module has an independent timer selection which can be accessed using the CCPTMRS1 register (Register 23-2). Please note that the PWM mode operation is described with respect to T2TMR in the following sections.

Figure 24-1 shows a simplified block diagram of PWM operation.

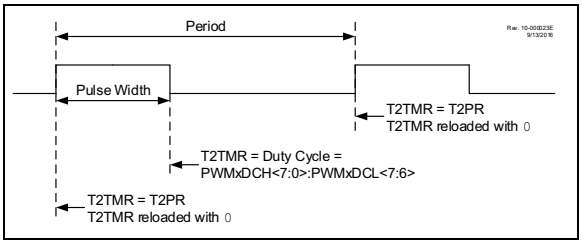
Figure 24-2 shows a typical waveform of the PWM signal.

FIGURE 24-1: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time-base.

FIGURE 24-2: PWM OUTPUT



For a step-by-step procedure on how to set up this module for PWM operation, refer to **Section 24.1.9 "Setup for PWM Operation using PWMx Pins"**.

REGISTER 31-6: UxUIR: UART GENERAL INTERRUPT REGISTER

R/S/W-0/0	R/S/W-0/0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
WUIF	ABDIF	—	—	—	ABDIE	—	—
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	S = Hardware set

bit 7 **WUIF:** Wake-up Interrupt bit
- 1 = Idle to non-idle transition on RX line detected when WUE is set. Also sets UxIF. (WUIF must be cleared by software to clear UxIF)
- 0 = WUE not enabled by software or no transition detected

bit 6 **ABDIF**: Auto-baud detect interrupt bit
- 1 = Auto-baud detection complete. Status shown in UxIF when ABDIE is set. (Must be cleared by software)
- 0 = Auto-baud not enabled or auto-baud enabled and auto-baud detection not complete

bit 5-3 **Unimplemented:** Read as '0'

bit 2 **ABDIE:** Auto-baud Detect Interrupt Enable bit
- 1 = ABDIF will set UxIF bit in PIRx register
- 0 = ABDIF will not set UxIF

bit 1-0 **Unimplemented:** Read as '0'

REGISTER 31-16: UxP3H: UART PARAMETER 3 HIGH REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P3<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 7-6 **Unimplemented:** Read as '0'

bit 0 **P3<8>:** Most Significant Bit of Parameter 3
DMX mode:
Most Significant bit of last address of receive block
Other modes:
Not used

REGISTER 31-17: UxP3L: UART PARAMETER 3 LOW 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
P3<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 **P3<7:0>:** Least Significant Bits of Parameter 3
DMX mode:
Least Significant Byte of last address of receive block
LIN Slave mode:
Number of data bytes to receive
Asynchronous Address mode:
Receiver address mask. Received address is XOR'd with UxP2L then AND'd with UxP3L
Match occurs when result is zero
Other modes:
Not used

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.

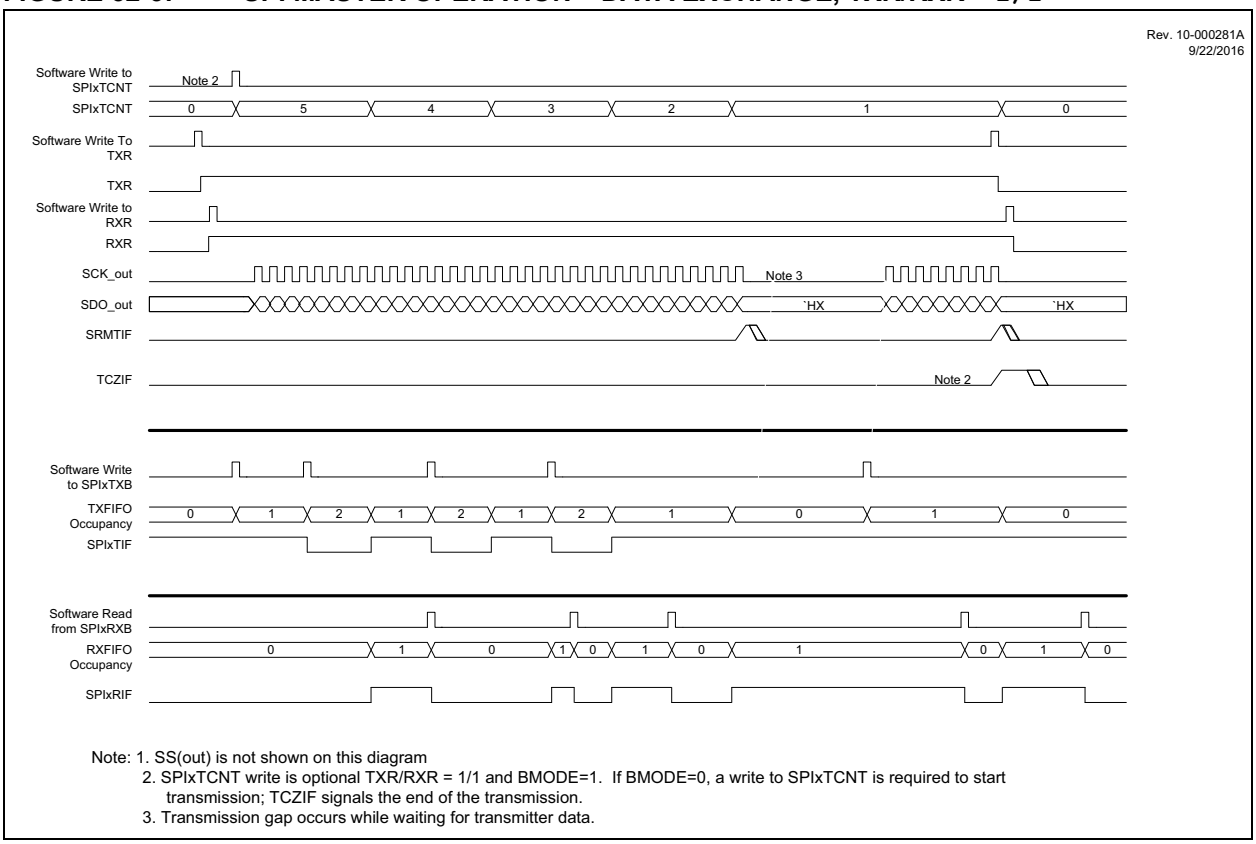
32.5.1 FULL DUPLEX MODE

When both TXR and RXR are set, the SPI master is in Full Duplex mode. In this mode, data transfer triggering is affected by the BMODE bit of SPIxCON0.

When BMODE = 1, data transfers will occur whenever both the RXFIFO is not full and there is data present in the TXFIFO. In practice, as long as the RXFIFO is not full, data will be transmitted/received as soon as the SPIxTxB register is written to, matching functionality of SPI (MSSP) modules on older 8-bit Microchip devices. The SPIxTCNT will decrement with each transfer. However, when SPIxTCNT is zero the next transfer is not inhibited and the corresponding SPIxTCNT decrement will cause the count to roll over to the maximum value. Figure 32-3 shows an example of a communication using this mode.

When BMODE = 0, the transfer counter (SPIxTCNTH/SPIxTCNTL) must also be written to before transfers will occur, and transfers will cease when the transfer counter reaches '0'. For example, if SPIxTXB is written twice and then SPIxTCTL is written with '3' then the transfer will start with the SPIxTCTL write. The two bytes in the TXFIFO will be sent after which the transfer will suspend until the third and last byte is written to SPIxTXB.

FIGURE 32-3: SPI MASTER OPERATION – DATA EXCHANGE, TXR/RXR = 1/1



Note: 1. SS(out) is not shown on this diagram
2. SPIxTCNT write is optional TXR/RXR = 1/1 and BMODE=1. If BMODE=0, a write to SPIxTCNT is required to start transmission; TCZIF signals the end of the transmission.
3. Transmission gap occurs while waiting for transmitter data.

There are four main operations based on the direction of the data being shared during I²C communication.

- Master Transmit (master is transmitting data to a slave)
- Master Receive (master is receiving data from a slave)
- Slave Transmit (slave is transmitting data to a master)
- Slave Receive (slave is receiving data from the master)

To begin any I²C communication, the master device sends out a Start bit followed by the address byte of the slave it intends to communicate with. This is followed by a single Read/Write bit, which determines whether the master intends to transmit to or receive data from the slave device.

If the requested slave exists on the bus, it will respond with an Acknowledge bit, otherwise known as an ACK. The master then continues to shift data in or out of the slave until it terminates the message with a Stop.

Further details about the I²C module are discussed in the section below.

33.3 I²C Mode Operation

All I²C communication is 8-bit data and 1-bit acknowledge and shifted out MSb first. The user can control the interaction between the software and the module using several control registers and interrupt flags. Two pins, SDA and SCL, are exercised by the module to communicate with other external I²C devices.

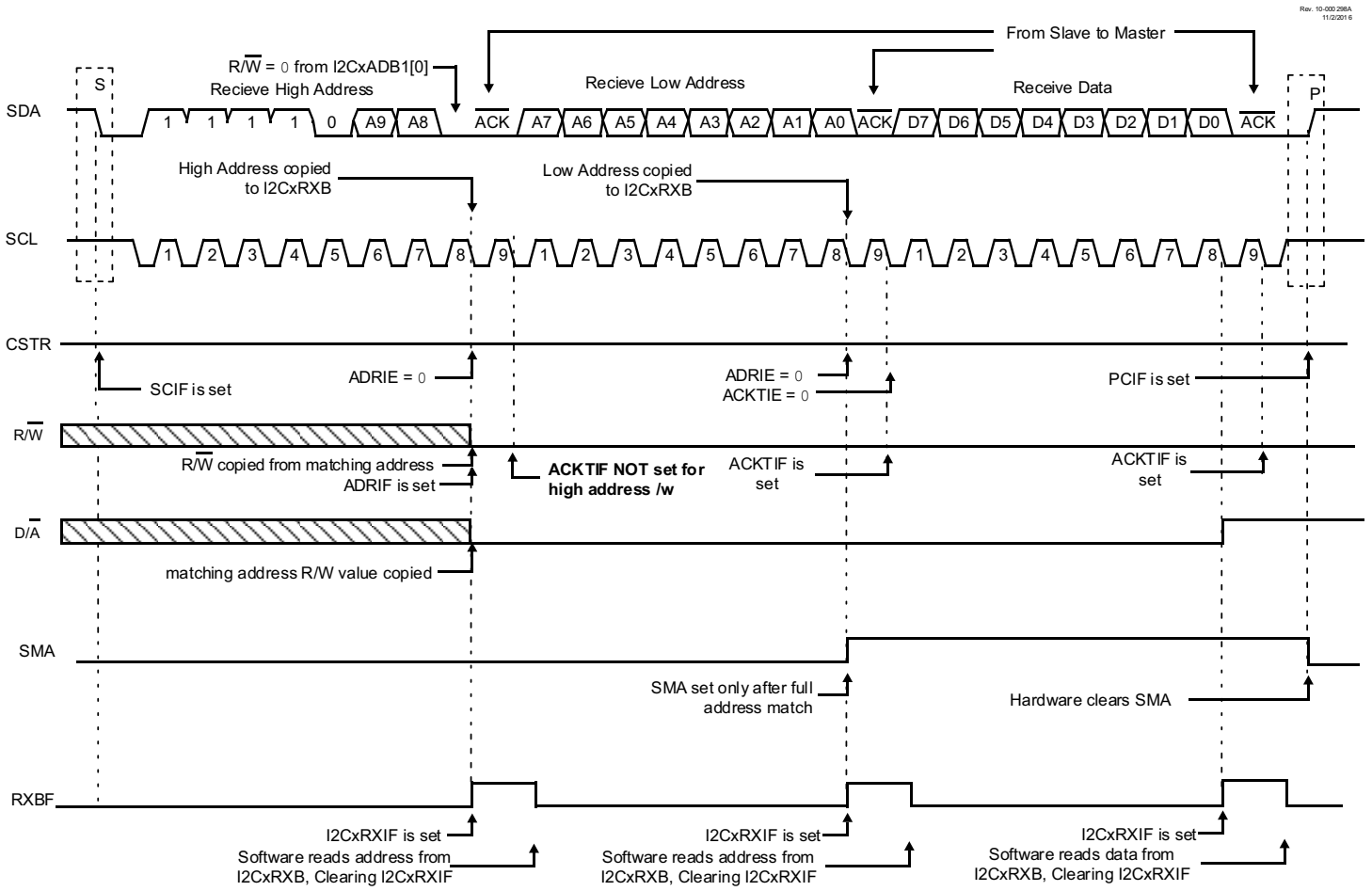
33.3.1 DEFINITION OF I²C TERMINOLOGY

The I²C communication protocol terminologies are defined for reference below in Table 33-1. These terminologies are used throughout this document. Table 33-1 has been adapted from the Phillips I²C specification.

TABLE 33-1: I²C BUS TERMS

TERM	Description
Transmitter	The device which shifts data out onto the bus
Receiver	The device which shifts data in from the bus
Master	The device that initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by the master
Multi-master	A bus with more than one device that can initiate data transfers
Arbitration	Procedure to ensure that only one master at a time controls the bus. Winning arbitration ensures that the message is not corrupted
Synchronization	Procedure to synchronize the clocks of two or more devices on the bus.
Idle	No master is controlling the bus, and both SDA and SCL lines are high
Active	Any time one or more master devices are controlling the bus
Addressed Slave	Slave device that has received a matching address and is actively being clocked by a master
Matching Address	Address byte that is clocked into a slave that matches the value stored in I2CxADR
Write Request	Slave receives a matching address with R/W bit clear and is ready to clock in data
Read Request	Master sends an address byte with the R/W bit set, indicating that it wishes to clock data out of the Slave. This data is the next and all following bytes until a Restart or Stop.
Clock Stretching	When a device on the bus holds SCL low to stall communication
Bus Collision	Any time the SDA line is sampled low by the module while it is outputting and expected high state.
Bus Timeout	Any time the I2CBTOISM input transitions high, the I²C module is reset and the module goes idle.

FIGURE 33-11: I^2C SLAVE, 10-BIT ADDRESS, RECEPTION WITH STOP (ADB = 1)



© 2016-2017 Microchip Technology Inc.

REGISTER 33-12: I2CxADR0: I²C ADDRESS 0 REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
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 HC = Hardware clear

bit 7-0 **ADR<7-0>:** Address 1 bits

MODE<2:0> = 00x | 11x - 7-bit Slave/Multi-Master Modes

ADR0<7:1>: 7-bit Slave Address

ADR0<0>: Unused in this mode; bit state is a don't care

MODE<2:0> = 01x - 10-bit Slave Modes

ADR0<7:0>: Eight Least Significant bits of 10-bit address 0

36.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIRx register. The ADC Interrupt Enable is the ADIE bit in the PIEx register. The ADIF bit must be cleared in software.

Note 1:	The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.
2:	The ADC operates during Sleep only when the FRC oscillator is selected.

This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake up from Sleep and resume in-line code execution, the ADIE bit of the PIEx register and the GIE bits of the INTCON0 register must both be set. If all these bits are set, the execution will switch to the Interrupt Service Routine.

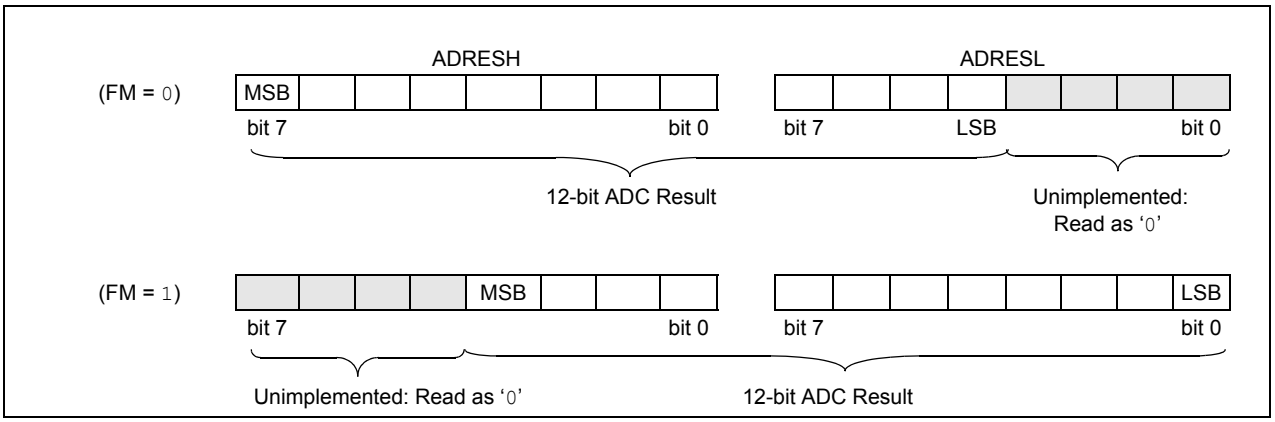
36.1.6 RESULT FORMATTING

The 12-bit ADC conversion result can be supplied in two formats, left justified or right justified. The FM bits of the ADCON0 register controls the output format.

Figure 36-3 shows the two output formats.

Writes to the ADRES register pair are always right justified regardless of the selected format mode. Therefore, data read after writing to ADRES when ADFRM0 = 0 will be shifted left four places.

FIGURE 36-3: 12-BIT ADC CONVERSION RESULT FORMAT



IORWF **Inclusive OR W with f**

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

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

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed 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: `IORWF RESULT, 0, 1`

Before Instruction
RESULT = 13h
W = 91h
After Instruction
RESULT = 13h
W = 93h

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 16383$

Operation: k → FSRf

Status Affected: None

Encoding:

1110 1111	1110 0000	$00k_{13}k$ k_7kkk	kkkk kkkk

Description: The 14-bit literal 'k' is loaded into the File Select Register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction
FSR2H = 03h
FSR2L = ABh