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	25
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf24k42-i-ss

TABLE 4-5: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F24/25K42 DEVICES BANK 61

3DFFh	—	3DDFh	U2FIFO	3DBFh	—	3D9Fh	—	3D7Fh	—	3D5Fh	I2C2CON2	3D3Fh	—	3D1Fh	—
3DFEh	—	3DDEh	U2BRGH	3DBEh	—	3D9Eh	—	3D7Eh	—	3D5Eh	I2C2CON1	3D3Eh	—	3D1Eh	—
3DFDh	—	3DDDh	U2BRGL	3DBDh	—	3D9Dh	—	3D7Dh	—	3D5Dh	I2C2CON0	3D3Dh	—	3D1Dh	—
3DFCh	—	3DDCh	U2CON2	3DBCh	—	3D9Ch	—	3D7Ch	I2C1BTO	3D5Ch	I2C2ADR3	3D3Ch	—	3D1Ch	SPI1CLK
3DFBh	—	3DDBh	U2CON1	3DBBh	—	3D9Bh	—	3D7Bh	I2C1CLK	3D5Bh	I2C2ADR2	3D3Bh	—	3D1Bh	SPI1INTE
3DFAh	U1ERRIE	3DDAh	U2CON0	3DBAh	—	3D9Ah	—	3D7Ah	I2C1PIE	3D5Ah	I2C2ADR1	3D3Ah	—	3D1Ah	SPI1INTF
3DF9h	U1ERRIR	3DD9h	—	3DB9h	—	3D99h	—	3D79h	I2C1PIR	3D59h	I2C2ADR0	3D39h	—	3D19h	SPI1BAUD
3DF8h	U1UIR	3DD8h	U2P3L	3DB8h	—	3D98h	—	3D78h	I2C1STAT1	3D58h	I2C2ADB1	3D38h	—	3D18h	SPI1TWIDTH
3DF7h	U1FIFO	3DD7h	—	3DB7h	—	3D97h	—	3D77h	I2C1STAT0	3D57h	I2C2ADB0	3D37h	—	3D17h	SPI1STATUS
3DF6h	U1BRGH	3DD6h	U2P2L	3DB6h	—	3D96h	—	3D76h	I2C1ERR	3D56h	I2C2CNT	3D36h	—	3D16h	SPI1CON2
3DF5h	U1BRGL	3DD5h	—	3DB5h	—	3D95h	—	3D75h	I2C1CON2	3D55h	I2C2TXB	3D35h	—	3D15h	SPI1CON1
3DF4h	U1CON2	3DD4h	U2P1L	3DB4h	—	3D94h	—	3D74h	I2C1CON1	3D54h	I2C2RXB	3D34h	—	3D14h	SPI1CON0
3DF3h	U1CON1	3DD3h	—	3DB3h	—	3D93h	—	3D73h	I2C1CON0	3D53h	—	3D33h	—	3D13h	SPI1TCNTH
3DF2h	U1CON0	3DD2h	U2TXB	3DB2h	—	3D92h	—	3D72h	I2C1ADR3	3D52h	—	3D32h	—	3D12h	SPI1TCNTL
3DF1h	U1P3H	3DD1h	—	3DB1h	—	3D91h	—	3D71h	I2C1ADR2	3D51h	—	3D31h	—	3D11h	SPI1TXB
3DF0h	U1P3L	3DD0h	U2RXB	3DB0h	—	3D90h	—	3D70h	I2C1ADR1	3D50h	—	3D30h	—	3D10h	SPI1RXB
3DEFh	U1P2H	3DCFh	—	3DAFh	—	3D8Fh	—	3D6Fh	I2C1ADR0	3D4Fh	—	3D2Fh	—	3D0Fh	—
3DEEh	U1P2L	3DCEh	—	3DAEh	—	3D8Eh	—	3D6Eh	I2C1ADB1	3D4Eh	—	3D2Eh	—	3D0Eh	—
3DEDh	U1P1H	3DCDh	—	3DADh	—	3D8Dh	—	3D6Dh	I2C1ADB0	3D4Dh	—	3D2Dh	—	3D0Dh	—
3DECh	U1P1L	3DCCh	—	3DACh	—	3D8Ch	—	3D6Ch	I2C1CNT	3D4Ch	—	3D2Ch	—	3D0Ch	—
3DEBh	U1TXCHK	3DCBh	—	3DABh	—	3D8Bh	—	3D6Bh	I2C1TXB	3D4Bh	—	3D2Bh	—	3D0Bh	—
3DEAh	U1TXB	3DCAh	—	3DAAh	—	3D8Ah	—	3D6Ah	I2C1RXB	3D4Ah	—	3D2Ah	—	3D0Ah	—
3DE9h	U1RXCHK	3DC9h	—	3DA9h	—	3D89h	—	3D69h	—	3D49h	—	3D29h	—	3D09h	—
3DE8h	U1RXB	3DC8h	—	3DA8h	—	3D88h	—	3D68h	—	3D48h	—	3D28h	—	3D08h	—
3DE7h	—	3DC7h	—	3DA7h	—	3D87h	—	3D67h	—	3D47h	—	3D27h	—	3D07h	—
3DE6h	—	3DC6h	—	3DA6h	—	3D86h	—	3D66h	I2C2BTO	3D46h	—	3D26h	—	3D06h	—
3DE5h	—	3DC5h	—	3DA5h	—	3D85h	—	3D65h	I2C2CLK	3D45h	—	3D25h	—	3D05h	—
3DE4h	—	3DC4h	—	3DA4h	—	3D84h	—	3D64h	I2C2PIE	3D44h	—	3D24h	—	3D04h	—
3DE3h	—	3DC3h	—	3DA3h	—	3D83h	—	3D63h	I2C2PIR	3D43h	—	3D23h	—	3D03h	—
3DE2h	U2ERRIE	3DC2h	—	3DA2h	—	3D82h	—	3D62h	I2C2STAT1	3D42h	—	3D22h	—	3D02h	—
3DE1h	U2ERRIR	3DC1h	—	3DA1h	—	3D81h	—	3D61h	I2C2STAT0	3D41h	—	3D21h	—	3D01h	—
3DE0h	U2UIR	3DC0h	—	3DA0h	—	3D80h	—	3D60h	I2C2ERR	3D40h	—	3D20h	—	3D00h	—

Legend: Unimplemented data memory locations and registers, read as '0'.

REGISTER 5-8: CONFIGURATION WORD 4H (30 0007h)

U-1	U-1	R/W-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	LVP	—	$\overline{\text{WRTSAF}}$ (1)	$\overline{\text{WRTD}}$ (1)	$\overline{\text{WRTC}}$ (1)	$\overline{\text{WRTB}}$ (1)
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 **LVP:** Low Voltage Programming Enable bit
1 = Low-voltage programming enabled. $\overline{\text{MCLR}}$/VPP pin function is $\overline{\text{MCLR}}$. MCLRE (Register 5-3) is ignored.
0 = HV on $\overline{\text{MCLR}}$/VPP must be used for programming.

Note: The LVP bit cannot be written (to zero) while operating from the LVP programming interface. The purpose of this rule is to prevent the user from dropping out of LVP mode while programming from LVP mode, or accidentally eliminating LVP mode from the configuration state

bit 4 **Unimplemented:** Read as '1'

bit 3 **$\overline{\text{WRTSAF}}$:** Storage Area Flash (SAF) write protection bit
1 = SAF not write-protected
0 = SAF write-protected

Note: Unimplemented if SAF is not present and only applicable if $\overline{\text{SAFEN}}$ = 0

bit 2 **$\overline{\text{WRTD}}$:** Data EEPROM write protection bit
1 = Data EEPROM not write-protected
0 = Data EEPROM write-protected

bit 1 **$\overline{\text{WRTC}}$:** Configuration Register write protection bit
1 = Configuration Register not write-protected
0 =Configuration Register write-protected

bit 0 **$\overline{\text{WRTB}}$:** Boot Block write protection bit
1 = Boot Block not write-protected
0 = Boot Block write-protected

Note: Only applicable if $\overline{\text{BBEN}}$ = 0

Note 1: Once protection is enabled through ICSP or a self write, it can only be reset through a bulk erase.

6.0 DEVICE INFORMATION AREA

The Device Information Area (DIA) is a dedicated region in the Program memory space. The DIA contains the calibration data for the internal temperature indicator module, stores the Microchip Unique Identifier words and the Fixed Voltage Reference voltage readings measured in mV.

The complete DIA table is shown in Table 6-1: Device Information Area, followed by a description of each region and its functionality. The data is mapped from 3F0000h to 3F003Fh in the PIC18(L)F2X/4X/5XK42 family. These locations are read-only and cannot be erased or modified by the user. The data is programmed into the device during manufacturing.

TABLE 6-1: DEVICE INFORMATION AREA

Address Range	Name of Region	Standard Device Information
3F0000h-3F000Bh	MUI0	Microchip Unique Identifier (6 Words)
	MUI1	
	MUI2	
	MUI3	
	MUI4	
	MUI5	
3F000Ch-3F000Fh	MUI6	Unassigned (2 Words)
	MUI7	
3F0010h-3F0023h	EUI0	Optional External Unique Identifier (10 Words)
	EUI1	
	EUI2	
	EUI3	
	EUI4	
	EUI5	
	EUI6	
	EUI7	
	EUI8	
	EUI9	
3F0024h-3F0025h	TSLR1	Unassigned (1 Word)
3F0026h-3F0027h	TSLR2	Temperature Indicator ADC reading at @ 90°C (low range setting)
3F0028h-3F0029h	TSLR3	Unassigned (1 word)
3F002Ah-3F002Bh	TSHR1	Unassigned (1 Word)
3F002Ch-3F002Dh	TSHR2	Temperature Indicator ADC reading at @ 90°C (high range setting)
3F002Eh-3F002Fh	TSHR3	Unassigned (1 Word)
3F0030h-3F0031h	FVRA1X	ADC FVR1 Output voltage for 1x setting (in mV)
3F0032h-3F0033h	FVRA2X	ADC FVR1 Output Voltage for 2x setting (in mV)
3F0034h-3F0035h	FVRA4X	ADC FVR1 Output Voltage for 4x setting (in mV)
3F0036h-3F0037h	FVRC1X	Comparator FVR2 output voltage for 1x setting (in mV)
3F0038h-3F0039h	FVRC2X	Comparator FVR2 output voltage for 2x setting (in mV)
3F003Ah-3F003Bh	FVRC4X	Comparator FVR2 output voltage for 4x setting (in mV)
3F003Ch-3F00Fh		Unassigned (34 Words)

REGISTER 11-5: PIR2: PERIPHERAL INTERRUPT REGISTER 2

R-0/0	R-0/0	R-0/0	R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
I2C1RXIF	SPI1IF	SPI1TXIF	SPI1RXIF	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF
bit 7							bit 0

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

bit 7 **I2C1RXIF:** I^2C1 Receive Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 6 **SPI1IF:** SPI1 Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 5 **SPI1TXIF:** SPI1 Transmit Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

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

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

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

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

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

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

REGISTER 11-17: PIE3: PERIPHERAL INTERRUPT Enable REGISTER 3

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
TMR0IE	U1IE	U1EIE	U1TXIE	U1RXIE	I2C1EIE	I2C1IE	I2C1TXIE
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 **TMR0IE:** TMR0 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **U1IE:** UART1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **U1EIE:** UART1 Framing Error Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **U1TXIE:** UART1 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **U1RXIE:** UART1 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **I2C1EIE:** I^2C1 Error Interrupt Enable bit
1 = Enabled
0 = Disabled

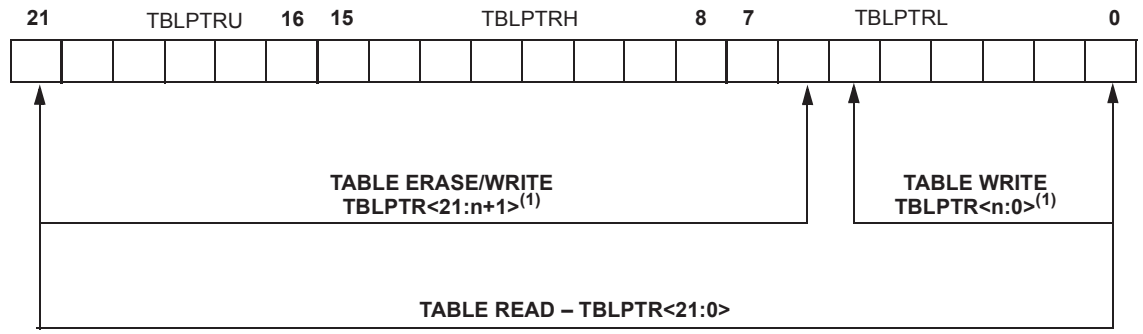
bit 1 **I2C1IE:** I^2C1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **I2C1TXIE:** I^2C1 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

TABLE 15-3: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

Example	Operation on Table Pointer
TBLRD* TBLWT*	TBLPTR is not modified
TBLRD*+ TBLWT*+	TBLPTR is incremented after the read/write
TBLRD*- TBLWT*-	TBLPTR is decremented after the read/write
TBLRD+* TBLWT+*	TBLPTR is incremented before the read/write

FIGURE 15-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



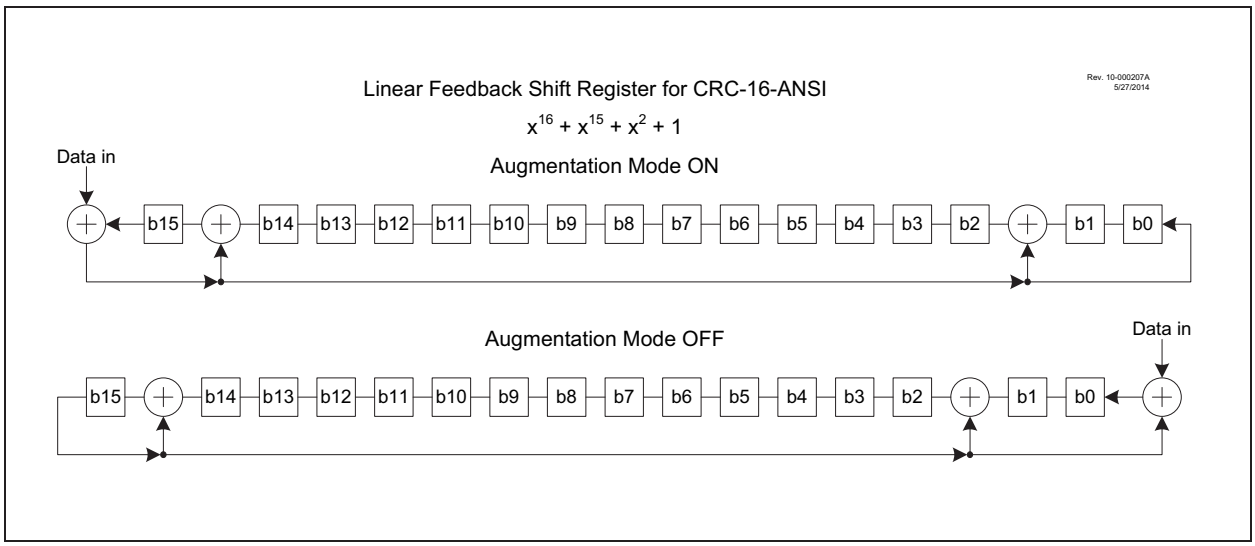
Note 1: Refer to Table 15-2 for the row size values.

16.3 CRC Polynomial Implementation

Any polynomial can be used.The polynomial and accumulator sizes are determined by the PLEN<3:0> bits. For an n-bit accumulator, PLEN = n-1 and the corresponding polynomial is n+1 bits. Therefore the accumulator can be any size up to 16 bits with a corresponding polynomial up to 17 bits. The MSb and LSb of the polynomial are always '1' which is forced by hardware. All polynomial bits between the MSb and LSb are specified by the CRCXOR registers. For example, when using CRC-16-ANSI, the polynomial is defined as $X^{16}+X^{15}+X^2+1$.

The X^{16} and X^0 = 1 terms are the MSb and LSb controlled by hardware. The X^{15} and X^2 terms are specified by setting the corresponding CRCXOR<15:0> bits with the value of '0x8004'. The actual value is '0x8005' because the hardware sets the LSb to 1. However, the LSb of the CRCXORL register is unimplemented and always reads as '0'. Refer to Example 16-1.

EXAMPLE 16-2: CRC LFSR EXAMPLE



16.4 CRC Data Sources

Data can be input to the CRC module in two ways:

- User data using the CRCDAT registers (CRCDATH and CRCDATL)
- Program memory using the Program Memory Scanner

To set the number of bits of data, up to 16 bits, the DLEN bits of CRCCON1 must be set accordingly. Only data bits in CRCDAT registers up to DLEN will be used, other data bits in CRCDAT registers will be ignored.

Data is moved into the CRCSHIFT as an intermediate to calculate the check value located in the CRCACC registers.

The SHIFTM bit is used to determine the bit order of the data being shifted into the accumulator. If SHIFTM is not set, the data will be shifted in MSb first (Big Endian). The value of DLEN will determine the MSb. If SHIFTM bit is set, the data will be shifted into the accumulator in reversed order, LSb first (Little Endian).

The CRC module can be seeded with an initial value by setting the CRCACC<15:0> registers to the appropriate value before beginning the CRC.

16.4.1 CRC FROM USER DATA

To use the CRC module on data input from the user, the user must write the data to the CRCDAT registers. The data from the CRCDAT registers will be latched into the shift registers on any write to the CRCDATL register.

16.4.2 CRC FROM FLASH

To use the CRC module on data located in Program memory, the user can initialize the Program Memory Scanner as defined in **Section 16.8, Scanner Module Overview**.

TABLE 16-3: SUMMARY OF REGISTERS ASSOCIATED WITH CRC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CRCACCH	ACC<15:8>								223
CRCACCL	ACC<7:0>								224
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	222
CRCCON1	DLEN<3:0>				PLEN<3:0>				222
CRCDATH	DATA<15:8>								223
CRCDATL	DATA<7:0>								223
CRCSHIFTH	SHIFT<15:8>								224
CRCSHIFTL	SHIFT<7:0>								224
CRCXORH	X<15:8>								225
CRCXORL	X<7:1>							—	225
SCANCON0	EN	TRIGEN	SGO	—	—	MREG	BURSTMD	BUSY	226
SCANHADRU	—	—	HADR<21:16>						228
SCANHADRH	HADR<15:8>								229
SCANHADRL	HADR<7:0>								229
SCANLADRU	—	—	LADR<21:16>						227
SCANLADRH	LADR<15:8>								227
SCANLADRL	LADR<7:0>								228
SCANTRIG	—	—	—	—	TSEL<3:0>				230

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for the CRC module.

Depending on the priority of the DMA with respect to CPU execution (Refer to **Section 3.2 "Memory Access Scheme"** for more information), the DMA Controller can move data through two methods:

- Stalling the CPU execution until it has completed its transfers (DMA has higher priority over the CPU in this mode of operation)
- Utilizing unused CPU cycles for DMA transfers (CPU has higher priority over the DMA in this mode of operation). Unused CPU cycles are referred to as bubbles which are instruction cycles available for use by the DMA to perform read and write operations. In this way, the effective bandwidth for handling data is increased; at the same time, DMA operations can proceed without causing a processor stall.

17.4 DMA Interface

The DMA module transfers data from the source to the destination one byte at a time, this smallest data movement is called a DMA data transaction. A DMA Message refers to one or more DMA data transactions.

Each DMA data transaction consists of two separate actions:

- Reading the Source Address Memory and storing the value in the DMA Buffer register
- Writing the contents of the DMA Buffer register to the Destination Address Memory

Note:	DMA data movement is a two-cycle operation.

The XIP bit (DMAxCON0 register) is a status bit to indicate whether or not the data in the DMAxBUF register has been written to the destination address. If the bit is set then data is waiting to be written to the destination. If clear it means that either data has been written to the destination or that no source read has occurred.

The DMA has read access to PFM, Data EEPROM, and SFR/GPR space, and write access to SFR/GPR space. Based on these memory access capabilities, the DMA can support the following memory transactions:

TABLE 17-1: DMA MEMORY ACCESS

Read Source	Write Destination
Program Flash Memory	GPR
Program Flash Memory	SFR
Data EE	GPR
Data EE	SFR
GPR	GPR
SFR	GPR
GPR	SFR
SFR	SFR

Even though the DMA module has access to all memory and peripherals that are also available to the CPU, it is recommended that the DMA does not access any register that is part of the System arbitration. The DMA, as a system arbitration client should not be read or written by itself or by another DMA instantiation.

The following sections discuss the various control interfaces required for DMA data transfers.

17.4.1 DMA ADDRESSING

The start addresses for the source read and destination write operations are set using the DMAxSSA <21:0> and DMAxDSA <15:0> registers, respectively.

When the DMA Message transfers are in progress, the DMAxSPTR <21:0> and DMAxDPTR <15:0> registers contain the current address pointers for each source read and destination write operation, these registers are modified after each transaction based on the Address mode selection bits.

The SMODE and DMODE bits in the DMAxCON1 control register determine the address modes of operation by controlling how the DMAxSPTR <21:0> and DMAxDPTR <15:0> bits are updated after every DMA data transaction combination (Figure 17-2).

Each address can be separately configured to:

- Remain unchanged
- Increment by 1
- Decrement by 1

REGISTER 17-18: DMAxDSZL – DMAx DESTINATION SIZE 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
DSZ<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 **DSZ<7:0>:** Destination Message Size bits

REGISTER 17-19: DMAxDSZH – DMAx DESTINATION SIZE HIGH REGISTER

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	DSZ<11:8>			
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 7-4 **Unimplemented:** Read as '0'

bit 3-0 **DSZ<11:8>:** Destination Message Size bits

REGISTER 17-20: DMAxDCNTL – DMAx DESTINATION COUNT LOW REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DCNT<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 **DCNT<7:0>:** Current Destination Byte Count

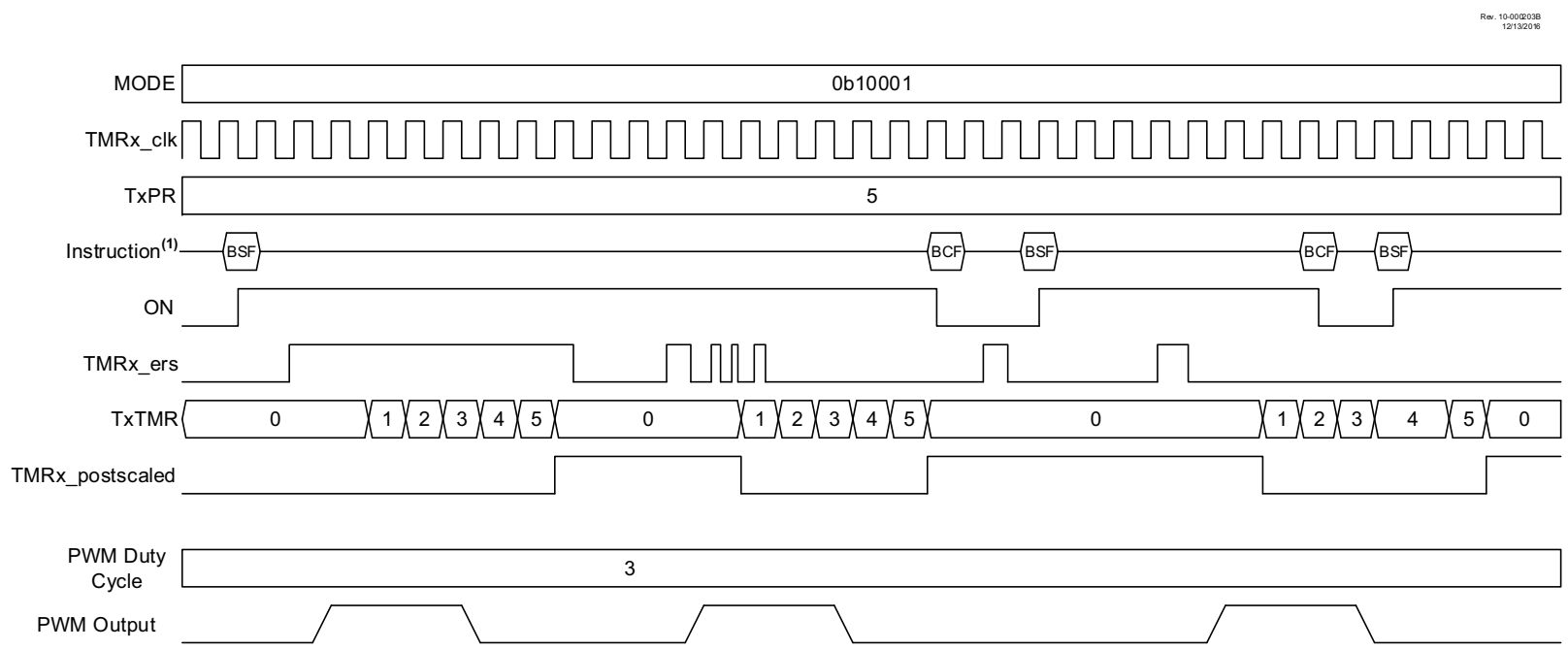
24.5.9 EDGE-TRIGGERED MONOSTABLE MODES

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

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

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

FIGURE 24-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = 10001)



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

27.6.2 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 27-4 and Figure 27-5.

REGISTER 34-2: SPIxINTE: SPI INTERRUPT ENABLE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0
SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—
bit 7							bit 0

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

bit 7 **SRMTIE**: Shift Register Empty Interrupt Enable bit
1 = Enables the Shift Register Empty Interrupt
0 = Disables the Shift Register Empty Interrupt

bit 6 **TCZIE**: Transfer Counter is Zero Interrupt Enable bit
1 = Enables the Transfer Counter is Zero Interrupt
0 = Disables the Transfer Counter is Zero Interrupt

bit 5 **SOSIE**: Start of Slave Select Interrupt Enable bit
1 = Enables the Start of Slave Select Interrupt
0 = Disables the Start of Slave Select Interrupt

bit 4 **EOSIE**: End of Slave Select Interrupt Enable bit
1 = Enables the End of Slave Select Interrupt
0 = Disables the End of Slave Select Interrupt

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

bit 2 **RXOIE**: Receiver Overflow Interrupt Enable bit
1 = Enables the Receiver Overflow Interrupt
0 = Disables the Receiver Overflow Interrupt

bit 1 **TXUIE**: Transmitter Underflow Interrupt Enable bit
1 = Enables the Transmitter Underflow Interrupt
0 = Disables the Transmitter Underflow Interrupt

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

REGISTER 34-3: SPIxTCNTL – SPI TRANSFER COUNTER LSB 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
TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
bit 7							bit 0

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

bit 7-0 TCNT<7:0>
BMODE = 0:
Bits 10-3 of the Transfer Counter, counting the total number of bits to transfer
BMODE = 1:
Bits 7-0 of the Transfer Counter, counting the total number of bytes to transfer

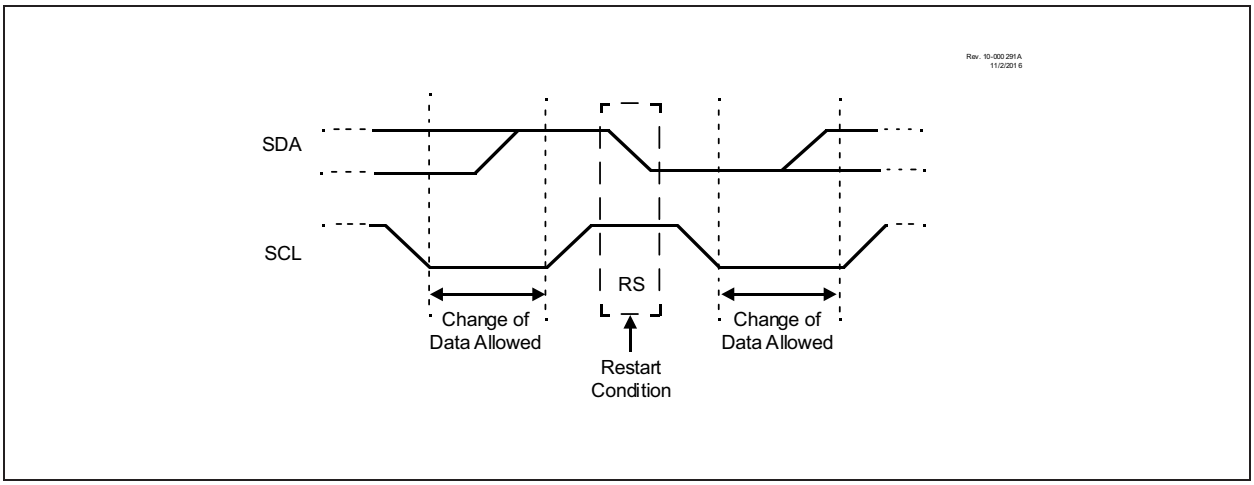
Note: This register should not be written to while a transfer is in progress (BUSY bit of SPIxCON2 is set).

35.3.7 RESTART CONDITION

A Restart is valid any time that a Stop would be valid. A master can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the slave that a Start would, resetting all slave logic and preparing it to clock in an address. The master may want to address the same or another slave. Figure 35-4 shows the waveform for a Restart condition.

In 10-bit Addressing Slave mode a Restart is required for the master to clock data out of the addressed slave. Once a slave has been fully addressed, matching both high and low address bytes (SMA = 1), the master can issue a Restart and the high address byte with the R/W bit set. The slave logic will then hold the clock and prepare to clock out data.

FIGURE 35-4: RESTART CONDITION



35.3.8 ACKNOWLEDGE SEQUENCE

The ninth SCL pulse for any transferred byte in I^2C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDA line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge (ACK) is an active-low signal, pulling the SDA line low indicates to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an ACK is placed in the ACKSTAT bit of the I2CxCON1 register. The ACKSTAT bit is cleared when the receiving device sends an Acknowledge and is set when the receiving device does not Acknowledge. A slave sends an Acknowledge when it has recognized its address. When in a mode that is receiving data, the ACK data being sent to the transmitter depends on the value of I2CxCNT register. ACKDT is the value sent when I2CxCNT! = 0. When I2CxCNT = 0, the ACKCNT value is used instead.

In Slave mode, if the ADRIE or WRIE bits are set, clock stretching is initiated when there is an address match or when there is an attempt to write to slave. This allows the user to set the ACK value sent back to the transmitter. The ACKDT bit of the I2CxCON1 register is set/cleared to determine the response. Slave hardware will generate an ACK response if the ADRIE or WRIE bits are clear.

Certain conditions will cause a not-ACK (NACK) to be sent automatically. If any of the RXRE, TXRE, RXO, or TXU bits is set, the hardware response is forced to NACK. All subsequent responses from the device for address matches or data will be a NACK response.

35.3.9 BUS TIME-OUT

The I2CxBTO register can be used to select the time-out source for the module. The I^2C module is reset when the selected bus time out signal goes high. This feature is useful for SMBus and PMBus™ compatibility.

For example, Timer2 can be selected as the bus time-out source and configured to count when the SCL pin is low. If the timer runs over before the SCL pin transitioned high, the timer-out pulse will reset the module.

Note:	The bus time-out source should produce a rising edge.

If the module is configured as a slave and a BTO event occurs when the slave is active, i.e. the SMA bit is set, the module is immediately reset. The SMA and CSTR bits are also cleared, and the BTOIF bit is set.

38.2.6 AUTO-CONVERSION TRIGGER

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

The Auto-conversion Trigger source is selected by the ADACT register.

Using the Auto-conversion Trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met. See Register 38-33 for auto-conversion sources.

38.2.7 ADC CONVERSION PROCEDURE (BASIC MODE)

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

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

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

2: Refer to **Section 38.3 "ADC Acquisition Requirements"**.

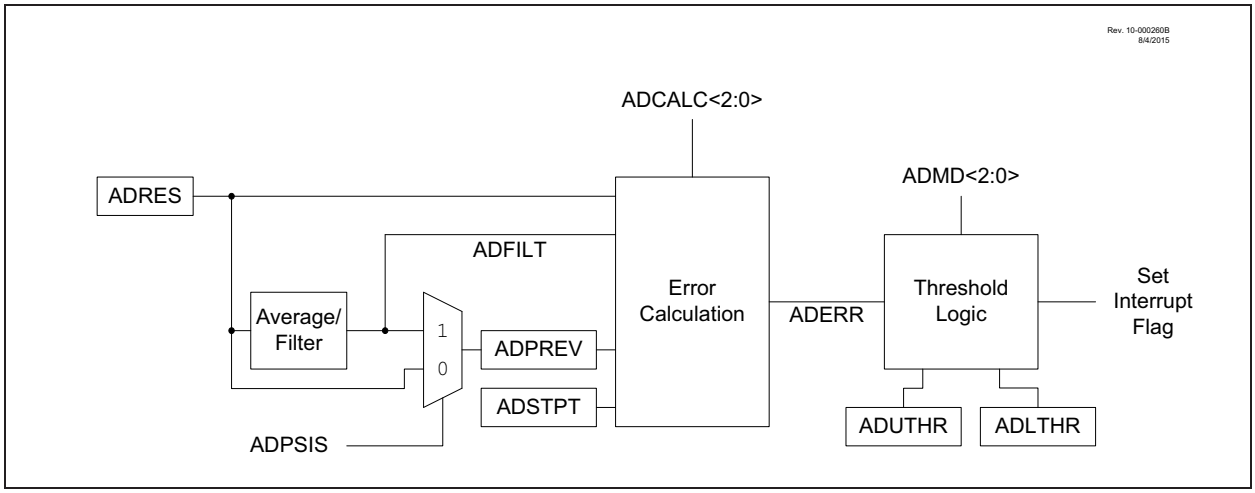
EXAMPLE 38-1: ADC CONVERSION

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

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

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

38.6 Computation Operation

The ADC module hardware is equipped with post conversion computation features. These features provide data post-processing functions that can be operated on the ADC conversion result, including digital filtering/averaging and threshold comparison functions.

FIGURE 38-10: COMPUTATIONAL FEATURES SIMPLIFIED BLOCK DIAGRAM



The operation of the ADC computational features is controlled by ADMD <2:0> bits in the ADCON2 register.

The module can be operated in one of five modes:

• **Basic**: In this mode, ADC conversion occurs on single (ADDSEN = 0) or double (ADDSEN = 1) samples. ADIF is set after all the conversion are complete.

• **Accumulate**: With each trigger, the ADC conversion result is added to accumulator and CNT increments. ADIF is set after each conversion. ADTIF is set according to the calculation mode.

• **Average**: With each trigger, the ADC conversion result is added to the accumulator. When the RPT number of samples have been accumulated, a threshold test is performed. Upon the next trigger, the accumulator is cleared. For the subsequent tests, additional RPT samples are required to be accumulated.

• **Burst Average**: At the trigger, the accumulator is cleared. The ADC conversion results are then collected repetitively until RPT samples are accumulated and finally the threshold is tested.

• **Low-Pass Filter (LPF)**: With each trigger, the ADC conversion result is sent through a filter. When RPT samples have occurred, a threshold test is performed. Every trigger after that the ADC conversion result is sent through the filter and another threshold test is performed.

The five modes are summarized in Table 38-2 below.

REGISTER 38-8: ADPCH: ADC POSITIVE CHANNEL SELECTION REGISTER

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
—	—	ADPCH<5:0>					
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 7-6 **Unimplemented**: Read as '0'

bit 5-0 **ADPCH<5:0>**: ADC Positive Input Channel Selection bits
111111 = FVR Buffer 2(2)
111110 = FVR Buffer 1(2)
111101 = DAC1 output(1)
111100 = Temperature Indicator(3)
111011 = Vss (Analog Ground)
111010 = Reserved. No channel connected.
•
•
•
011000 = Reserved. No channel connected.
010111 = ANC7
010110 = ANC6
010101 = ANC5
010100 = ANC4
010011 = ANC3
010010 = ANC2
010001 = ANC1
010000 = ANC0
001111 = ANB7
001110 = ANB6
001101 = ANB5
001100 = ANB4
001011 = ANB3
001010 = ANB2
001001 = ANB1
001000 = ANB0
000111 = ANA7
000110 = ANA6
000101 = ANA5
000100 = ANA4
000011 = ANA3
000010 = ANA2
000001 = ANA1
000000 = ANA0

Note 1: See **Section 39.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** for more information.
2: See **Section 36.0 "Fixed Voltage Reference (FVR)"** for more information.
3: See **Section 37.0 "Temperature Indicator Module"** for more information.

REGISTER 38-20: ADRESH: ADC RESULT REGISTER HIGH, FM = 1

U-0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	—	—	ADRES<11:8>			
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 7-4 **Reserved**

bit 3-0 **ADRES<11:8>**: ADC Sample Result bits. Upper four bits of 12-bit conversion result.

REGISTER 38-21: ADRESL: ADC RESULT REGISTER LOW, FM = 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<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 **ADRES<7:0>**: ADC Result Register bits. Lower eight bits of 12-bit conversion result.

TSTFSZ **Test f, skip if 0**

Syntax: TSTFSZ f {,a}

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

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a 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 43.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 00h,
PC = Address (NZERO)

XORLW **Exclusive OR literal with W**

Syntax: XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

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

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example:
```
XORLW   0AFh
```
Before Instruction
W = B5h
After Instruction
W = 1Ah

TABLE 46-8: INTERNAL OSCILLATOR PARAMETERS(1)

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
OS50	FHFOSC	Precision Calibrated HFINTOSC Frequency	—	4 8 12 16 48 64	—	MHz	(**Note 2**)
OS51	FHFOSCLP	Low-Power Optimized HFINTOSC Frequency	0.93 1.86	1 2	1.07 2.14	MHz MHz	
OS53*	FLFOSC	Internal LFINTOSC Frequency	—	31	—	kHz	
OS54*	THFOSCST	HFINTOSC Wake-up from Sleep Start-up Time	— —	11 50	20 —	µs µs	VREGPM = 0 VREGPM = 1
OS56	TLFOSCST	LFINTOSC Wake-up from Sleep Start-up Time	—	0.2	—	ms	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: To ensure these oscillator frequency tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 µF and 0.01 µF values in parallel are recommended.

2: See Figure 46-6: Precision Calibrated HFINTOSC Frequency Accuracy Over Device VDD and Temperature.

FIGURE 46-6: PRECISION CALIBRATED HFINTOSC FREQUENCY ACCURACY OVER DEVICE VDD AND TEMPERATURE