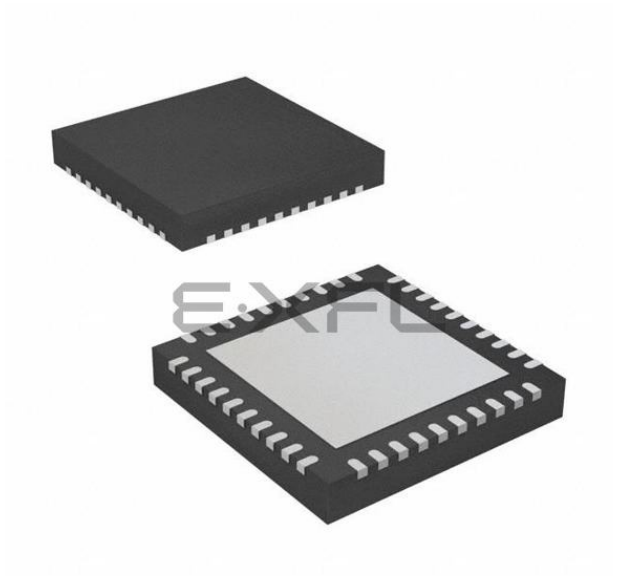


Welcome to E-XFL.COM

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	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	8K 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/pic18f47k42-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 →

			LSB = 1	LSB = 0	Word Address ↓
					000000h
					000002h
					000004h
					000006h
Instruction 1:	`MOVLW`	`055h`	0Fh	55h	000008h
Instruction 2:	`GOTO`	`0006h`	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	`MOVFF`	`123h, 456h`	C1h	23h	00000Eh
			F4h	56h	000010h
Instruction 4:	`MOVFFL`	`123h, 456h`	00h	60h	000012h
			F4h	8Ch	000014h
			F4h	56h	000016h
					000018h
					00001Ah

FIGURE 4-4: DATA MEMORY MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Bank	BSR<5:0>	Address addr<7:0>	PIC18(L)F45K42 PIC18(L)F55K42	PIC18(L)F26K42 PIC18(L)F46K42 PIC18(L)F56K42	PIC18(L)F27K42 PIC18(L)F47K42 PIC18(L)F57K42	Address addr<13:0>
Bank 0	00 0000	00h	Access RAM	Access RAM	Access RAM	0000h 005Fh
		FFh	GPR	GPR	GPR	0060h 00FFh
Bank 1	00 0001	00h FFh	GPR	GPR	GPR	0100h . . . 03FFh
Bank 2	00 0010	00h FFh				
Bank 3	00 0011	00h . . FFh				
Banks 4 to 7	00 0100 – 00 0111	00h . . FFh	GPR	GPR	GPR	0400h . . 07FFh
Banks 8 to 15	00 1000 – 00 1111	00h . . FFh	Unimplemented	GPR	GPR	0800h . . 0FFFh
Banks 16 to 31	01 0000 – 01 1111	00h . . FFh		Unimplemented		1000h . . 1FFFh
Banks 32 to 55	10 0000 – 11 0111	00h . . FFh			Unimplemented	2000h . . 37FFh
Banks 56 to 62	11 1000 – 11 1110	00h . . FFh	SFR	SFR	SFR	3800h . . 3EFFh
Bank 63	11 1111	00h . . FFh	SFR	SFR	SFR	3800h 3EFFh 3F60h 3FFFh

Virtual Bank

Access RAM	00h 5Fh
SFR	60h FFh

Preliminary

TABLE 4-6: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 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’.
Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented in PIC18(L)F26/27/45/46/47K42.

TABLE 4-11: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES BANK 56

38FFh	—	38DFh	—	38BFh	—	389Fh	IVTADU	387Fh	—	385Fh	—	383Fh	—	381Fh	—
38FEh	—	38DEh	—	38BEh	—	389Eh	IVTADH	387Eh	—	385Eh	—	383Eh	—	381Eh	—
38FDh	—	38DDh	—	38BDh	—	389Dh	IVTADL	387Dh	—	385Dh	—	383Dh	—	381Dh	—
38FCh	—	38DCh	—	38BCh	—	389Ch	—	387Ch	—	385Ch	—	383Ch	—	381Ch	—
38FBh	—	38DBh	—	38BBh	—	389Bh	—	387Bh	—	385Bh	—	383Bh	—	381Bh	—
38FAh	—	38DAh	—	38BAh	—	389Ah	—	387Ah	—	385Ah	—	383Ah	—	381Ah	—
38F9h	—	38D9h	—	38B9h	—	3899h	—	3879h	—	3859h	—	3839h	—	3819h	—
38F8h	—	38D8h	—	38B8h	—	3898h	—	3878h	—	3858h	—	3838h	—	3818h	—
38F7h	—	38D7h	—	38B7h	—	3897h	—	3877h	—	3857h	—	3837h	—	3817h	—
38F6h	—	38D6h	—	38B6h	—	3896h	—	3876h	—	3856h	—	3836h	—	3816h	—
38F5h	—	38D5h	—	38B5h	—	3895h	—	3875h	—	3855h	—	3835h	—	3815h	—
38F4h	—	38D4h	—	38B4h	—	3894h	—	3874h	—	3854h	—	3834h	—	3814h	—
38F3h	—	38D3h	—	38B3h	—	3893h	—	3873h	—	3853h	—	3833h	—	3813h	—
38F2h	—	38D2h	—	38B2h	—	3892h	—	3872h	—	3852h	—	3832h	—	3812h	—
38F1h	—	38D1h	—	38B1h	—	3891h	—	3871h	—	3851h	—	3831h	—	3811h	—
38F0h	—	38D0h	—	38B0h	—	3890h	PRODH_SHAD	3870h	—	3850h	—	3830h	—	3810h	—
38EFh	—	38CFh	—	38AFh	—	388Fh	PRODL_SHAD	386Fh	—	384Fh	—	382Fh	—	380Fh	—
38EEh	—	38CEh	—	38AEh	—	388Eh	FSR2H_SHAD	386Eh	—	384Eh	—	382Eh	—	380Eh	—
38EDh	—	38CDh	—	38ADh	—	388Dh	FSR2L_SHAD	386Dh	—	384Dh	—	382Dh	—	380Dh	—
38ECh	—	38CCh	—	38ACh	—	388Ch	FSR1H_SHAD	386Ch	—	384Ch	—	382Ch	—	380Ch	—
38EBh	—	38CBh	—	38ABh	—	388Bh	FSR1L_SHAD	386Bh	—	384Bh	—	382Bh	—	380Bh	—
38EAh	—	38CAh	—	38AAh	—	388Ah	FSR0H_SHAD	386Ah	—	384Ah	—	382Ah	—	380Ah	—
38E9h	—	38C9h	—	38A9h	—	3889h	FSR0L_SHAD	3869h	—	3849h	—	3829h	—	3809h	—
38E8h	—	38C8h	—	38A8h	—	3888h	PCLATU_SHAD	3868h	—	3848h	—	3828h	—	3808h	—
38E7h	—	38C7h	—	38A7h	—	3887h	PCLATH_SHAD	3867h	—	3847h	—	3827h	—	3807h	—
38E6h	—	38C6h	—	38A6h	—	3886h	BSR_SHAD	3866h	—	3846h	—	3826h	—	3806h	—
38E5h	—	38C5h	—	38A5h	—	3885h	WREG_SHAD	3865h	—	3845h	—	3825h	—	3805h	—
38E4h	—	38C4h	—	38A4h	—	3884h	STATUS_SHAD	3864h	—	3844h	—	3824h	—	3804h	—
38E3h	—	38C3h	—	38A3h	—	3883h	SHADCON	3863h	—	3843h	—	3823h	—	3803h	—
38E2h	—	38C2h	—	38A2h	—	3882h	BSR_CSHAD	3862h	—	3842h	—	3822h	—	3802h	—
38E1h	—	38C1h	—	38A1h	—	3881h	WREG_CSHAD	3861h	—	3841h	—	3821h	—	3801h	—
38E0h	—	38C0h	—	38A0h	—	3880h	STATUS_CSHAD	3860h	—	3840h	—	3820h	—	3800h	—

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

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

REGISTER 7-7: OSCEN: OSCILLATOR MANUAL ENABLE 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	U-0	U-0
EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—

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 **EXTOEN:** External Oscillator Manual Request Enable bit
1 = EXTOSC is explicitly enabled, operating as specified by FEXTOSC
0 = EXTOSC could be enabled by requesting peripheral

bit 6 **HFOEN:** HFINTOSC Oscillator Manual Request Enable bit
1 = HFINTOSC is explicitly enabled, operating as specified by OSCFRQ (Register 7-5)
0 = HFINTOSC could be enabled by requesting peripheral

bit 5 **MFOEN:** MFINTOSC (500 kHz/31.25 kHz) Oscillator Manual Request Enable bit (Derived from HFINTOSC)
1 = MFINTOSC is explicitly enabled
0 = MFINTOSC could be enabled by requesting peripheral

bit 4 **LFOEN:** LFINTOSC (31 kHz) Oscillator Manual Request Enable bit
1 = LFINTOSC is explicitly enabled
0 = LFINTOSC could be enabled by requesting peripheral

bit 3 **SOSCEN:** Secondary Oscillator Manual Request Enable bit
1 = Secondary Oscillator is explicitly enabled, operating as specified by SOSCPWR
0 = Secondary Oscillator could be enabled by requesting peripheral

bit 2 **ADOEN:** ADC Oscillator Manual Request Enable bit
1 = ADC oscillator is explicitly enabled
0 = ADC oscillator could be enabled by requesting peripheral

bit 1-0 **Unimplemented:** Read as ‘0’

9.5 Context Saving

The Interrupt controller supports a two-level deep context saving (Main routine context and Low ISR context). Refer to state machine shown in Figure 9-6 for details.

The Program Counter (PC) is saved on the dedicated device PC stack. CPU registers saved include STATUS, WREG, BSR, FSR0/1/2, PRODL/H and PCLATH/U.

After WREG has been saved to the context registers, the resolved vector number of the interrupt source to be serviced is copied into WREG. Context save and restore operation is completed by the interrupt controller based on current state of the interrupts and the order in which they were sent to the CPU.

Context save/restore works the same way in both states of MVECEN. When IPEN = 0, there is only one level interrupt active. Hence, only the main context is saved when an interrupt is received.

9.5.1 ACCESSING SHADOW REGISTERS

The Interrupt controller automatically saves the context information in the shadow registers available in Bank 56. Both the saved context values (i.e., main routine and low ISR) can be accessed using the same set of shadow registers. By clearing the SHADLO bit in the SHADCON register (Register 9-43), the CPU register values saved for main routine context can accessed, and by setting the SHADLO bit of the CPU register, values saved for low ISR context can accessed. Low ISR context is automatically restored to the CPU registers upon exiting the high ISR. Similarly, the main context is automatically restored to the CPU registers upon exiting the low ISR.

The Shadow registers in Bank 56 are readable and writable, so if the user desires to modify the context, then the corresponding shadow register should be modified and the value will be restored when exiting the ISR. Depending on the user's application, other registers may also need to be saved.

REGISTER 10-2: CPUDOZE: DOZE AND IDLE REGISTER

R/W-0/0	R/W/HC/HS-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>		
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	HC = Bit is cleared by hardware; HS = Bit is set by hardware

bit 7 **IDLEN:** Idle Enable bit
`1` = A `SLEEP` instruction places the device into Idle mode
`0` = A `SLEEP` instruction places the device into Sleep mode

bit 6 **DOZEN:** Doze Enable bit[(1)]
`1` = Places the device into Doze mode
`0` = Places the device into Normal mode

bit 5 **ROI:** Recover-On-Interrupt bit[(1)]
`1` = Entering the Interrupt Service Routine (ISR) makes DOZEN = `0`
`0` = Entering the Interrupt Service Routine (ISR) does not change DOZEN

bit 4 **DOE:** Doze-On-Exit bit[(1)]
`1` = Exiting the Interrupt Service Routine (ISR) makes DOZEN = `1`
`0` = Exiting the Interrupt Service Routine (ISR) does not change DOZEN

bit 3 **Unimplemented:** Read as ‘`0`’

bit 2-0 **DOZE<2:0>:** Ratio of CPU Instruction Cycles to Peripheral Instruction Cycles
`111` =1:256
`110` =1:128
`101` =1:64
`100` =1:32
`011` =1:16
`010` =1:8
`001` =1:4
`000` =1:2

Note 1: Refer Table 10-1 for more details.

TABLE 10-2: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
VREGCON[(1)]	—	—	—	—	—	—	VREGPM	Reserved	176
CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>			177

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used in Power-Down mode.
Note 1: Not present in LF parts.

REGISTER 15-12: DMAxSCNTL: DMAx SOURCE COUNT LOW REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SCNT<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 **SCNT<7:0>:** Current Source Byte Count

REGISTER 15-13: DMAxSCNTH: DMAx SOURCE COUNT HIGH REGISTER

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	SCNT<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 **SCNT<11:8>:** Current Source Byte Count

REGISTER 15-14: DMAxDSAL: DMAx DESTINATION 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
DSA<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 **DSA<7:0>:** Destination Start Address bits

REGISTER 21-2: TxGCON: TIMERx GATE CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R-x	U-0	U-0
GE	GPOL	GTM	GSPM	GGO/$\overline{\text{DONE}}$	GVAL	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **GE:** Timerx Gate Enable bit
If TMRxON = 1:
1 = Timerx counting is controlled by the Timerx gate function
0 = Timerx is always counting
If TMRxON = 0:
This bit is ignored

bit 6 **GPOL:** Timerx Gate Polarity bit
1 = Timerx gate is active-high (Timerx counts when gate is high)
0 = Timerx gate is active-low (Timerx counts when gate is low)

bit 5 **GTM:** Timerx Gate Toggle Mode bit
1 = Timerx Gate Toggle mode is enabled
0 = Timerx Gate Toggle mode is disabled and Toggle flip-flop is cleared
Timerx Gate Flip Flop Toggles on every rising edge

bit 4 **GSPM:** Timerx Gate Single Pulse Mode bit
1 = Timerx Gate Single Pulse mode is enabled and is controlling Timerx gate)
0 = Timerx Gate Single Pulse mode is disabled

bit 3 **GGO/$\overline{\text{DONE}}$:** Timerx Gate Single Pulse Acquisition Status bit
1 = Timerx Gate Single Pulse Acquisition is ready, waiting for an edge
0 = Timerx Gate Single Pulse Acquisition has completed or has not been started.
This bit is automatically cleared when TxGSPM is cleared.

bit 2 **GVAL:** Timerx Gate Current State bit
Indicates the current state of the Timerx gate that could be provided to TMRxH:TMRxL
Unaffected by Timerx Gate Enable (TMRxGE)

bit 1-0 **Unimplemented:** Read as ‘0’

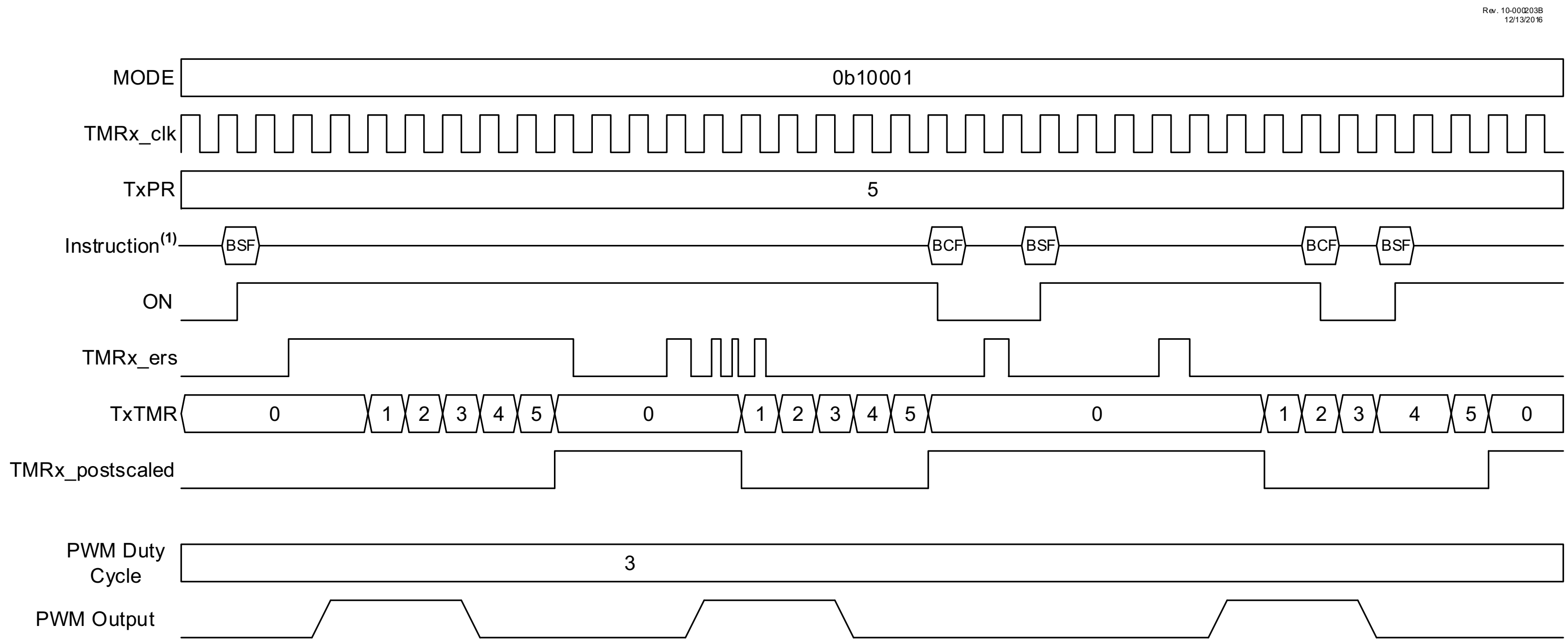
22.5.9 EDGE-TRIGGERED MONOSTABLE MODES

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

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

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

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



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

23.4.2 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for standard PWM operation:

1. Use the desired output pin RxyPPS control to select CCPx as the source and disable the CCPx pin output driver by setting the associated TRIS bit.
2. Load the T2PR register with the PWM period value.
3. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
4. Load the CCPRxL register, and the CCPRxH register with the PWM duty cycle value and configure the FMT bit of the CCPxCON register to set the proper register alignment.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note below.
   - Select the timer clock source to be as FOSC/4 using the T2CLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer prescale value.
   - Enable the Timer by setting the ON bit of the T2CON register.
6. Enable PWM output pin:
   - Wait until the Timer overflows and the TMR2IF bit of the PIR4 register is set. See Note below.
   - Enable the CCPx pin output driver by clearing the associated TRIS bit.

Note:	In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored.

23.4.3 TIMER2 TIMER RESOURCE

The PWM standard mode makes use of the 8-bit Timer2 timer resources to specify the PWM period.

23.4.4 PWM PERIOD

The PWM period is specified by the T2PR register of Timer2. The PWM period can be calculated using the formula of Equation 23-1.

EQUATION 23-1: PWM PERIOD

$$PWM\ Period = [(T2PR) + 1] \bullet 4 \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

Note 1: TOSC = 1/FOSC

When T2TMR is equal to T2PR, the following three events occur on the next increment cycle:

- T2TMR is cleared
- The CCPx pin is set. (Exception: If the PWM duty cycle = 0%, the pin will not be set.)
- The PWM duty cycle is transferred from the CCPRxL/H register pair into a 10-bit buffer.

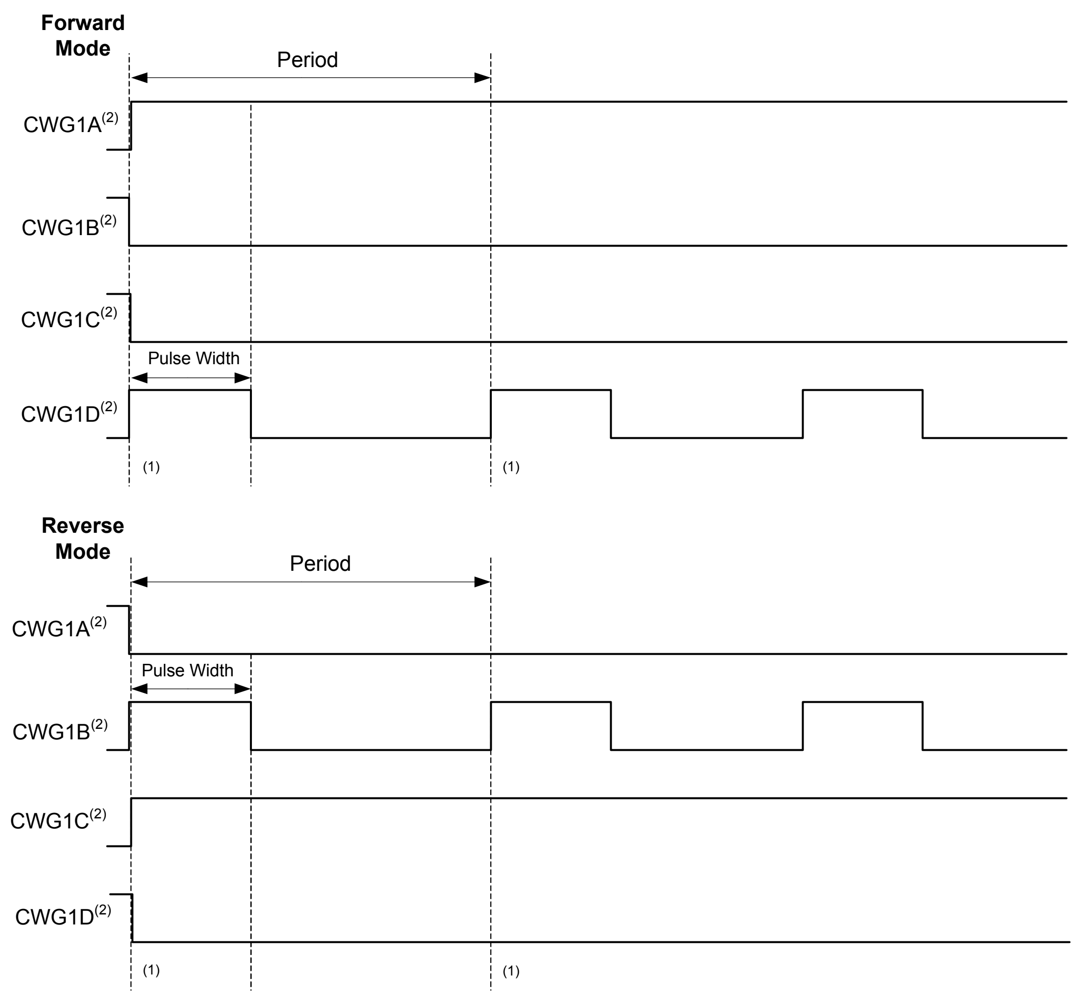
Note:	The Timer postscaler (see **Section 22.3 "External Reset Sources"**) is not used in the determination of the PWM frequency.

In Forward Full-Bridge mode (MODE<2:0> = 010), CWGxA is driven to its active state, CWGxB and CWGxC are driven to their inactive state, and CWGxD is modulated by the input signal, as shown in Figure 26-7.

In Reverse Full-Bridge mode (MODE<2:0> = 011), CWGxC is driven to its active state, CWGxA and CWGxD are driven to their inactive states, and CWGxB is modulated by the input signal, as shown in Figure 26-7.

In Full-Bridge mode, the dead-band period is used when there is a switch from forward to reverse or vice-versa. This dead-band control is described in **Section 26.6 "Dead-Band Control"**, with additional details in **Section 26.7 "Rising Edge and Reverse Dead Band"** and **Section 26.8 "Falling Edge and Forward Dead Band"**. Steering modes are not used with either of the Full-Bridge modes. The mode selection may be toggled between forward and reverse toggling the MODE<0> bit of the CWGxCON0 while keeping MODE<2:1> static, without disabling the CWG module.

FIGURE 26-7: EXAMPLE OF FULL-BRIDGE OUTPUT



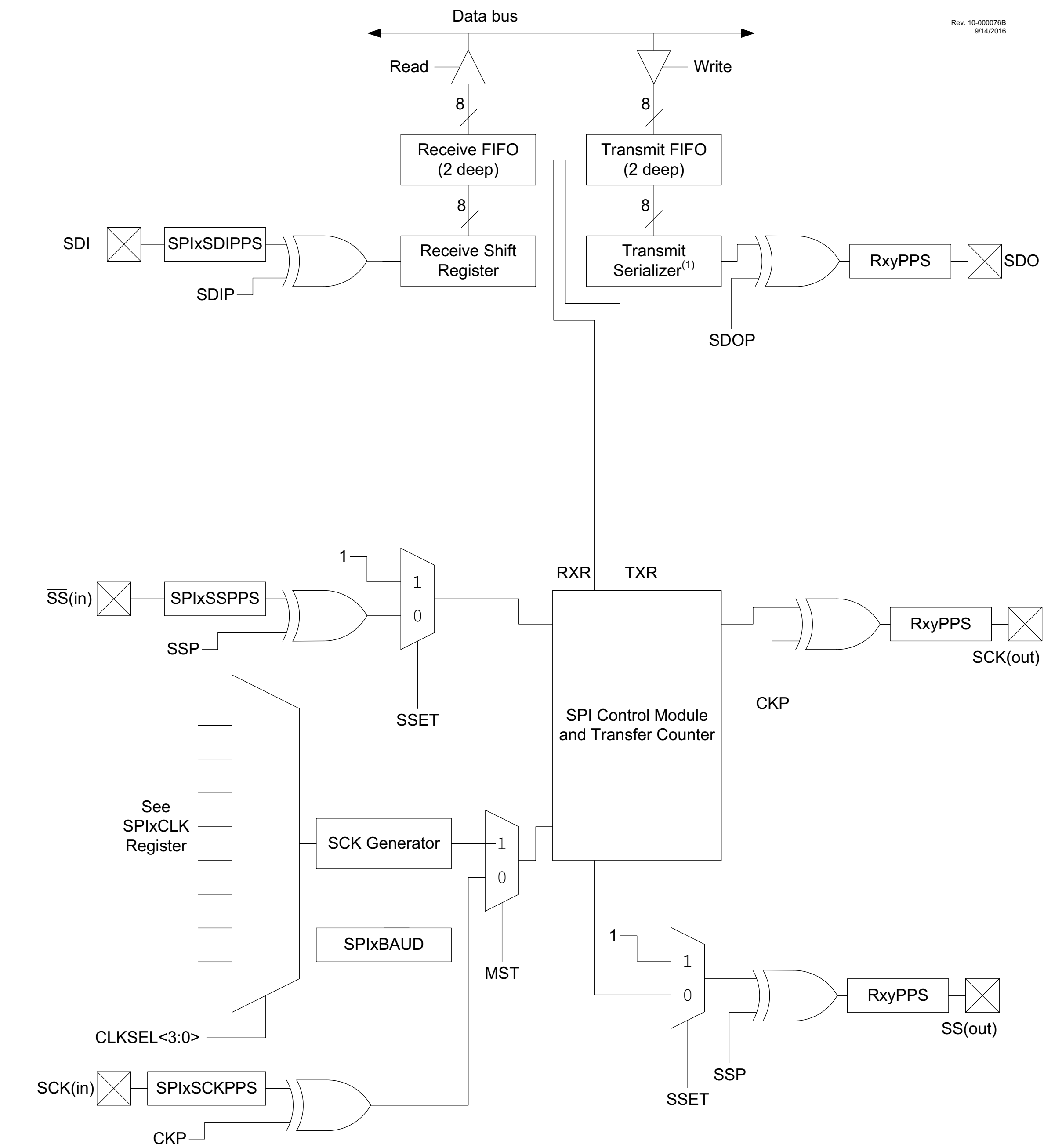
Note 1: A rising CWG data input creates a rising event on the modulated output.
2: Output signals shown as active-high; all POLy bits are clear.

TABLE 26-2: SUMMARY OF REGISTERS ASSOCIATED WITH CWG

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CWGxCON0	EN	LD	—	—	—	MODE<2:0>			424
CWGxCON1	—	—	IN	—	POLD	POLC	POLB	POLA	425
CWGxCLK	—	—	—	—	—	—	—	CS	426
CWGxISM	—	—	—	ISM<4:0>					427
CWGxSTR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	428
CWGxAS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—	429
CWGxAS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	430
CWGxDBR	—	—	DBR<5:0>						431
CWGxDBF	—	—	DBF<5:0>						431

Legend: – = unimplemented locations read as ‘0’. Shaded cells are not used by CWG.

FIGURE 32-1: SPI MODULE SIMPLIFIED BLOCK DIAGRAM



Note 1: If TXR=1 and the transmit FIFO is empty, the previous value of the receive shift register will be sent to the transmit serializer.

REGISTER 36-35: ADACT: ADC AUTO CONVERSION TRIGGER CONTROL REGISTER

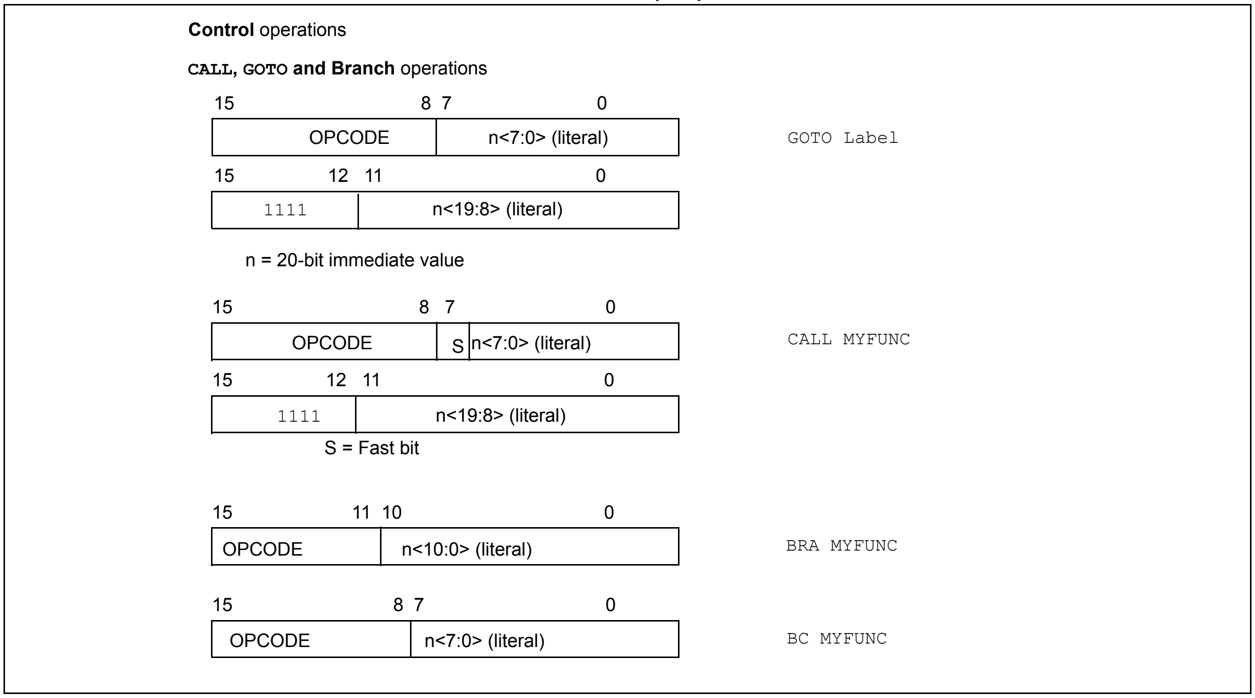
U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ACT<4:0>				
bit 7							bit 0

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 7-5 **Unimplemented**: Read as ‘0’

bit 4-0 **ADACT<4:0>**: Auto-Conversion Trigger Select Bits
11111 = Reserved, do not use
•
•
•
11110 = Reserved, do not use
11101 = Software write to ADPCH
11100 = Reserved, do not use
11011 = Software read of ADRESH
11010 = Software read of ADERRH
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = Logical OR of all Interrupt-on-change Interrupt Flags
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = PWM8_out
10000 = PWM7_out
01111 = PWM6_out
01110 = PWM5_out
01101 = CCP4_trigger
01100 = CCP3_trigger
01011 = CCP2_trigger
01010 = CCP1_trigger
01001 = SMT1_trigger
01000 = TMR6_postscaled
00111 = TMR5_overflow
00110 = TMR4_postscaled
00101 = TMR3_overflow
00100 = TMR2_postscaled
00011 = TMR1_overflow
00010 = TMR0_overflow
00001 = Pin selected by ADACTPPS
00000 = External Trigger Disabled

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



CPFSEQ — Compare f with W, skip if f = W

Syntax:	CPFSEQ f {,a}
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(f) – (W), skip if (f) = (W) (unsigned comparison)
Status Affected:	None
Encoding:	`0110` `001a` `ffff` `ffff`
Description:	Compares the contents of data memory location ‘f’ to the contents of W by performing an unsigned subtraction. If ‘f’ = W, then the fetched instruction is discarded and a `NOP` is executed instead, making this a 2-cycle instruction. If ‘a’ is ‘`0`’, the Access Bank is selected. If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank. 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(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      CPFSEQ REG, 0
NEQUAL    :
EQUAL     :
```

Before Instruction
- PC Address = `HERE`
- W = ?
- REG = ?

After Instruction
- If REG = W;
  - PC = Address (`EQUAL`)
- If REG ≠ W;
  - PC = Address (`NEQUAL`)

CPFSGT — Compare f with W, skip if f > W

Syntax:	CPFSGT f {,a}
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(f) – (W), skip if (f) > (W) (unsigned comparison)
Status Affected:	None
Encoding:	`0110` `010a` `ffff` `ffff`
Description:	Compares the contents of data memory location ‘f’ to the contents of the W by performing an unsigned subtraction. If the contents of ‘f’ are greater than the contents of WREG, then the fetched instruction is discarded and a `NOP` is executed instead, making this a 2-cycle instruction. If ‘a’ is ‘`0`’, the Access Bank is selected. If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank. If ‘a’ is ‘`0`’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 41.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:

```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```

Before Instruction
- PC = Address (`HERE`)
- W = ?

After Instruction
- If REG > W;
  - PC = Address (`GREATER`)
- If REG ≤ W;
  - PC = Address (`NGREATER`)

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

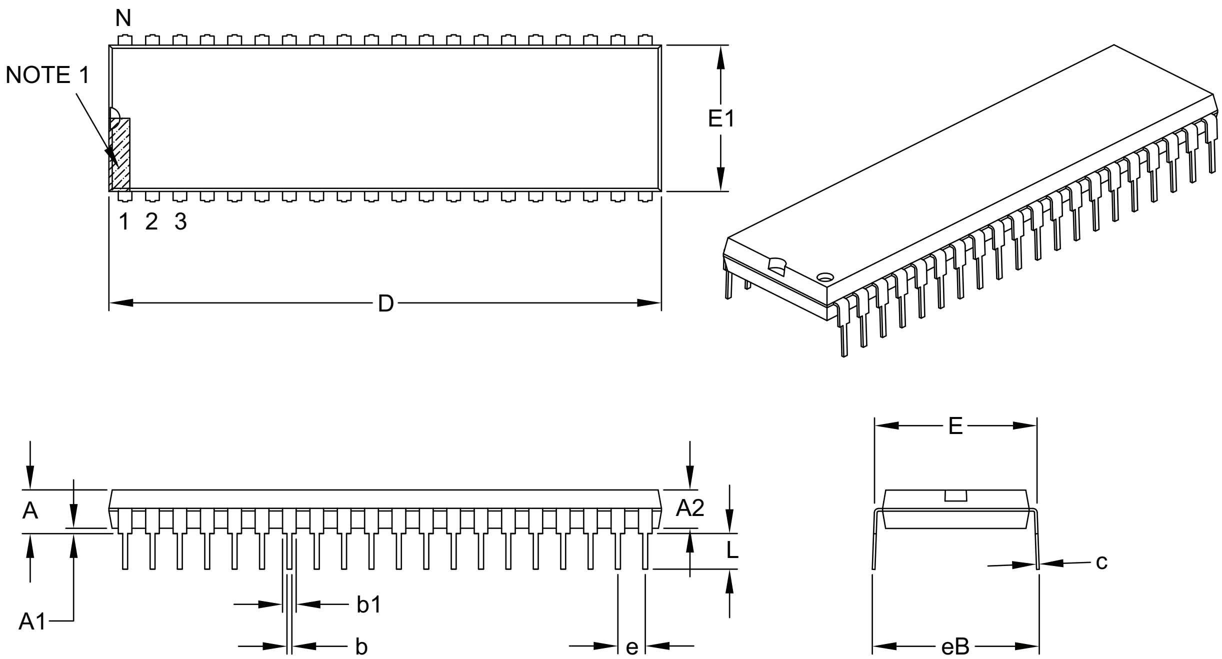
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3FD1h - 3FD0h	—	Unimplemented								
3FCFh	PORTF(3)	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	263
3FCEh	PORTE	—	—	—	—	RE0	RE2(2)	RE1(2)	RE1(2)	263
3FCDh	PORTD(2)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	263
3FCCh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	263
3FCBh	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	263
3FCAh	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	263
3FC9h - 3FC8h	—	Unimplemented								
3FB7h	TRISF(3)	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	264
3FB6h	TRISE(2)	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	264
3FB5h	TRISD(2)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	264
3FC4h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	264
3FC3h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	264
3FC2h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	264
3FC1h - 3FC0h	—	Unimplemented								
3FBFh	LATF(3)	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	265
3FBEh	LATE(2)	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	265
3FBDh	LATD(2)	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	265
3FBCh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	265
3FBBh	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	265
3FBAh	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	265
3FB9h	T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				302
3FB8h	T0CON0	EN	—	OUT	MD16	OUTPS				301
3FB7h	TMR0H	TMR0H								303
3FB6h	TMR0L	TMR0L								303
3FB5h	T1CLK	CS								315
3FB4h	T1GATE	GSS								316
3FB3h	T1GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3FB2h	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	338
3FB1h	TMR1H	TMR1H								317
3FB0h	TMR1L	TMR1L								317
3FAFh	T2RST	—	—	—	RSEL					336
3FAEh	T2CLK	—	—	—	—	CS				315
3FADh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3FACh	T2CON	ON	CKPS			OUTPS				313
3FABh	T2PR	PR2								337
3FAAh	T2TMR	TMR2								337
3FA9h	T3CLK	CS								315
3FA8h	T3GATE	GSS								316
3FA7h	T3GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3FA6h	T3CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	338
3FA5h	TMR3H	TMR3H								317
3FA4h	TMR3L	TMR3L								317
3FA3h	T4RST	—	—	—	RSEL					336
3FA2h	T4CLK	—	—	—	—	CS				335

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

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

40-Lead Plastic Dual In-Line (P) – 600 mil Body [PDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



		Units	INCHES	
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N		40	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.250
Molded Package Thickness	A2	.125	–	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.590	–	.625
Molded Package Width	E1	.485	–	.580
Overall Length	D	1.980	–	2.095
Tip to Seating Plane	L	.115	–	.200
Lead Thickness	c	.008	–	.015
Upper Lead Width	b1	.030	–	.070
Lower Lead Width	b	.014	–	.023
Overall Row Spacing §	eB	–	–	.700

Notes:

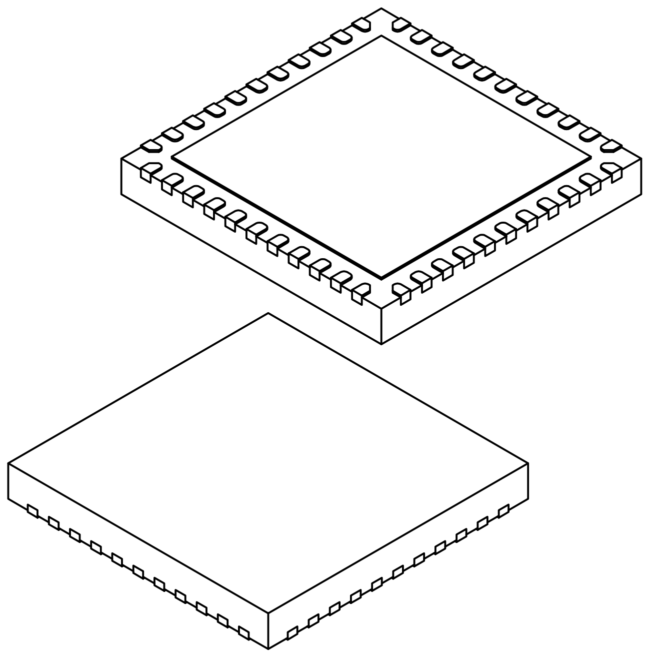
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-016B

44-Lead Plastic Quad Flat, No Lead Package (ML) - 8x8 mm Body [QFN or VQFN]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.25	6.45	6.60
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.25	6.45	6.60
Terminal Width	b	0.20	0.30	0.35
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103D Sheet 2 of 2