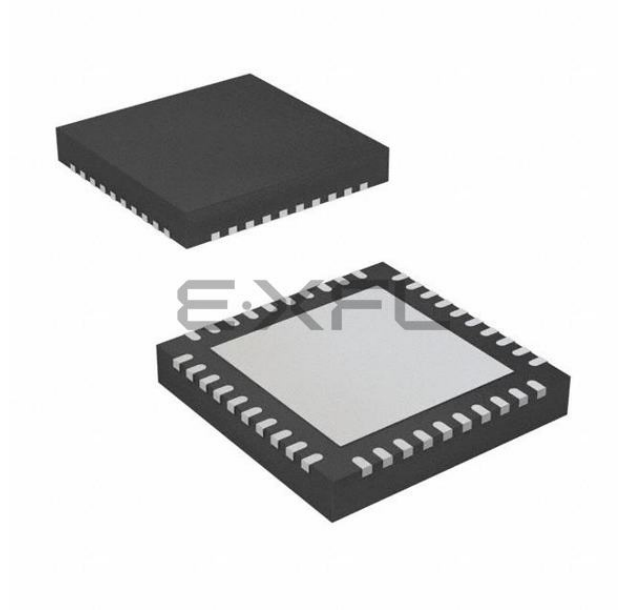


Welcome to E-XFL.COM

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	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
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/pic18lf46k42-i-mv

TABLE 4-5: SPECIAL FUNCTION REGISTER MAP FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES BANK 62

3EFFh	ADCLK	3EDFh	ADLTHH	3EBFh	CM1PCH	3E9Fh	—	3E7Fh	—	3E5Fh	—	3E3Fh	—	3E1Fh	—
3EFEh	ADACT	3EDEh	ADLTHL	3EBEh	CM1NCH	3E9Eh	DAC1CON0	3E7Eh	—	3E5Eh	—	3E3Eh	—	3E1Eh	—
3EFDh	ADREF	3EDDh	—	3EBDh	CM1CON1	3E9Dh	—	3E7Dh	—	3E5Dh	—	3E3Dh	—	3E1Dh	—
3EFCh	ADSTAT	3EDCh	—	3EBCh	CM1CON0	3E9Ch	DAC1CON1	3E7Ch	—	3E5Ch	—	3E3Ch	—	3E1Ch	—
3EFBh	ADCON3	3EDBh	—	3EBBh	CM2PCH	3E9Bh	—	3E7Bh	—	3E5Bh	—	3E3Bh	—	3E1Bh	—
3EFAh	ADCON2	3EDAh	—	3EBAh	CM2NCH	3E9Ah	—	3E7Ah	—	3E5Ah	—	3E3Ah	—	3E1Ah	—
3EF9h	ADCON1	3ED9h	—	3EB9h	CM2CON1	3E99h	—	3E79h	—	3E59h	—	3E39h	—	3E19h	—
3EF8h	ADCON0	3ED8h	—	3EB8h	CM2CON0	3E98h	—	3E78h	—	3E58h	—	3E38h	—	3E18h	—
3EF7h	ADPREH	3ED7h	ADCP	3EB7h	—	3E97h	—	3E77h	—	3E57h	—	3E37h	—	3E17h	—
3EF6h	ADPREL	3ED6h	—	3EB6h	—	3E96h	—	3E76h	—	3E56h	—	3E36h	—	3E16h	—
3EF5h	ADCAP	3ED5h	—	3EB5h	—	3E95h	—	3E75h	—	3E55h	—	3E35h	—	3E15h	—
3EF4h	ADACQH	3ED4h	—	3EB4h	—	3E94h	—	3E74h	—	3E54h	—	3E34h	—	3E14h	—
3EF3h	ADACQL	3ED3h	—	3EB3h	—	3E93h	—	3E73h	—	3E53h	—	3E33h	—	3E13h	—
2EF2h	—	3ED2h	—	3EB2h	—	3E92h	—	3E72h	—	3E52h	—	3E32h	—	3E12h	—
3EF1h	ADPCH	3ED1h	—	3EB1h	—	3E91h	—	3E71h	—	3E51h	—	3E31h	—	3E11h	—
3EF0h	ADRESH	3ED0h	—	3EB0h	—	3E90h	—	3E70h	—	3E50h	—	3E30h	—	3E10h	—
3EEFh	ADRESL	3ECFh	—	3EAFh	—	3E8Fh	—	3E6Fh	—	3E4Fh	—	3E2Fh	—	3E0Fh	—
3EEEh	ADPREVH	3ECEh	—	3EAEh	—	3E8Eh	—	3E6Eh	—	3E4Eh	—	3E2Eh	—	3E0Eh	—
3EEDh	ADPREVL	3ECDh	—	3EADh	—	3E8Dh	—	3E6Dh	—	3E4Dh	—	3E2Dh	—	3E0Dh	—
3EECh	ADRPT	3ECCh	—	3EACh	—	3E8Ch	—	3E6Ch	—	3E4Ch	—	3E2Ch	—	3E0Ch	—
3EEBh	ADCNT	3ECBh	—	3EABh	—	3E8Bh	—	3E6Bh	—	3E4Bh	—	3E2Bh	—	3E0Bh	—
3EEAh	ADACCU	3ECAh	HLVDCON1	3EAAh	—	3E8Ah	—	3E6Ah	—	3E4Ah	—	3E2Ah	—	3E0Ah	—
3EE9h	ADACCH	3EC9h	HLVDCON0	3EA9h	—	3E89h	—	3E69h	—	3E49h	—	3E29h	—	3E09h	—
3EE8h	ADACCL	3EC8h	—	3EA8h	—	3E88h	—	3E68h	—	3E48h	—	3E28h	—	3E08h	—
3EE7h	ADFLTRH	3EC7h	—	3EA7h	—	3E87h	—	3E67h	—	3E47h	—	3E27h	—	3E07h	—
3EE6h	ADFLTRL	3EC6h	—	3EA6h	—	3E86h	—	3E66h	—	3E46h	—	3E26h	—	3E06h	—
3EE5h	ADSTPTH	3EC5h	—	3EA5h	—	3E85h	—	3E65h	—	3E45h	—	3E25h	—	3E05h	—
3EE4h	ADSTPTL	3EC4h	—	3EA4h	—	3E84h	—	3E64h	—	3E44h	—	3E24h	—	3E04h	—
3EE3h	ADERRH	3EC3h	ZCDCON	3EA3h	—	3E83h	—	3E63h	—	3E43h	—	3E23h	—	3E03h	—
3EE2h	ADERRL	3EC2h	—	3EA2h	—	3E82h	—	3E62h	—	3E42h	—	3E22h	—	3E02h	—
3EE1h	ADUTHH	3EC1h	FVRCON	3EA1h	—	3E81h	—	3E61h	—	3E41h	—	3E21h	—	3E01h	—
3EE0h	ADUTHL	3EC0h	CMOUT	3EA0h	—	3E80h	—	3E60h	—	3E40h	—	3E20h	—	3E00h	—

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 5-5: CONFIGURATION WORD 3L (30 0004h)

U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	WDTE<1:0>		WDTCPS<4:0>				
bit 7							bit 0

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

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

bit 6-5 **WDTE<1:0>:** WDT Operating Mode bits
00 = WDT is disabled, SWDTEN is ignored
01 = WDT is enabled/disabled by the SWDTEN bit in WDTCON0
10 = WDT is enabled while Sleep = 0, suspended when Sleep = 1; SWDTEN is ignored
11 = WDT is enabled regardless of Sleep; SWDTEN is ignored

bit 4-0 **WDTCPS<4:0>:** WDT Period Select bits

WDTCPS<4:0>	WDTPS at POR				Software Control of WDTPS?
	Value	Divider Ratio		Typical Time-out (FIN = 31 kHz)	
00000	00000	1:32	2^5	1 ms	No
00001	00001	1:64	2^6	2 ms	
00010	00010	1:128	2^7	4 ms	
00011	00011	1:256	2^8	8 ms	
00100	00100	1:512	2^9	16 ms	
00101	00101	1:1024	2^{10}	32 ms	
00110	00110	1:2048	2^{11}	64 ms	
00111	00111	1:4096	2^{12}	128 ms	
01000	01000	1:8192	2^{13}	256 ms	
01001	01001	1:16384	2^{14}	512 ms	
01010	01010	1:32768	2^{15}	1s	
01011	01011	1:65536	2^{16}	2s	
01100	01100	1:131072	2^{17}	4s	
01101	01101	1:262144	2^{18}	8s	
01110	01110	1:524299	2^{19}	16s	
01111	01111	1:1048576	2^{20}	32s	
10000	10000	1:2097152	2^{21}	64s	
10001	10001	1:4194304	2^{22}	128s	
10010	10010	1:8388608	2^{23}	256s	
10011 ... 11110	10011 ... 11110	1:32	2^5	1 ms	No
11111	01011	1:65536	2^{16}	2s	Yes

EXAMPLE 9-3: SETTING UP VECTORED INTERRUPTS USING MPASM

```
ISR_TMR0:  CODE       0x8C0                 ; ISR code at 0x08C0 in PFM
       BANKSEL        PIR0                  ; Select bank for PIR0
       BCF            PIR3, TMR0IF          ; Clear TMR0IF
       BTG            LATC, 0, ACCESS       ; Code to execute in ISR
       RETFIE         1                     ; Return from ISR

InterruptInit:
       BANKSEL        INTCON0               ; Select bank for INTCON0
       BSF            INTCON0, GIEH         ; Enable high priority interrupts
       BSF            INTCON0, GIEL         ; Enable low priority interrupts
       BSF            INTCON0, IPEN_INTCON0 ; Enable interrupt priority

       BANKSEL        PIE0                  ; Select bank for PIE0
       BSF            PIE3, TMR0IE          ; Enable TMR0 interrupt
       BSF            PIE4, TMR1IE          ; Enable TMR1 interrupt

       BCF            IPR3, TMR0IP          ; Make TMR0 interrupt low priority
       RETURN         1

VectorTableInit:
       ; Set IVTBASE (optional - default is 0x000008)
       MOVLW          0x00                  ; This is optional
       MOVWF          IVTBASEU, ACCESS      ; If not included, then the
       MOVLW          0x40                  ; hardware default value of
       MOVWF          IVTBASEH, ACCESS      ; 0x0008 will be taken.
       MOVLW          0x08
       MOVWF          IVTBASEL, ACCESS

       ; TMR0 vector at IVTBASE + 2*(TMR0 vector number i.e. 31) = 0x4046
       MOVLW          0x00                  ; Load TBLPTR with the
       MOVWF          TBLPTRU, ACCESS       ; PFM memory location to be
       MOVLW          0x40                  ; written to.
       MOVWF          TBLPTRH, ACCESS
       MOVLW          0x46
       MOVWF          TBLPTRL, ACCESS

       ; Write the contents of TMR0 vector location
       ; ISR_TMR0_ADDRESS >> 2 = 0x08C0 >> 2 = 0x0230
       MOVLW          0x30                  ; Low byte first
       MOVWF          TABLAT, ACCESS
       TBLWT*+                              ; Write to temp table latch

       MOVLW          0x02                  ; High byte next
       MOVWF          TABLAT, ACCESS
       TBLWT*+                              ; Write to temp table latch

       ; Write to PFM now using NVMCON
       BANKSEL        NVMCON1               ; Select bank for NVMCON1
       MOVLW          0x84                  ; Setting to write to PFM
       MOVWF          NVMCON1

       MOVLW          0x55                  ; Required unlock sequence
       MOVWF          NVMCON2
       MOVLW          0xAA
       MOVWF          NVMCON2
       BSF            NVMCON1, WR           ; Start writing to PFM

       BTFSC          NVMCON1, WR           ; Wait for write to complete
       GOTO           $-2

       RETURN         1
```

10.4 Register Definitions: Voltage Regulator Control

REGISTER 10-1: VREGCON: VOLTAGE REGULATOR CONTROL REGISTER(1)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1
—	—	—	—	—	—	VREGPM	Reserved
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 7-2 **Unimplemented:** Read as '0'

bit 1 **VREGPM:** Voltage Regulator Power Mode Selection bit
- 1 = Low-Power Sleep mode enabled in Sleep(2)
  Draws lowest current in Sleep, slower wake-up
- 0 = Normal Power mode enabled in Sleep(2)
  Draws higher current in Sleep, faster wake-up

bit 0 **Reserved:** Read as '1'. Maintain this bit set.

Note 1: Not present in LF parts.

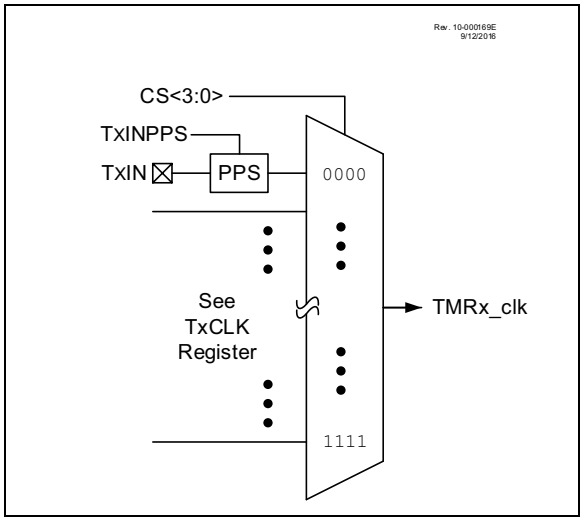
2: See **Section 44.0 "Electrical Specifications"**.

TABLE 14-2: 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>								219
CRCACCL	ACC<7:0>								220
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	218
CRCCON1	DLEN<3:0>				PLEN<3:0>				218
CRCDATH	DATA<15:8>								219
CRCDATL	DATA<7:0>								219
CRCSHIFTH	SHIFT<15:8>								220
CRCSHIFTL	SHIFT<7:0>								220
CRCXORH	X<15:8>								221
CRCXORL	X<7:1>							—	221
SCANCON0	EN	TRIGEN	SGO	—	—	MREG	BURSTMD	BUSY	222
SCANHADRU	—	—	HADR<21:16>						224
SCANHADRH	HADR<15:8>								225
SCANHADRL	HADR<7:0>								225
SCANLADRU	—	—	LADR<21:16>						223
SCANLADRH	LADR<15:8>								223
SCANLADRL	LADR<7:0>								224
SCANTRIG	—	—	—	—	TSEL<3:0>				226

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

FIGURE 22-2: TIMER2 CLOCK SOURCE BLOCK DIAGRAM



22.1 Timer2 Operation

Timer2 operates in three major modes:

- Free Running Period
- One-Shot
- Monostable

Within each mode there are several options for starting, stopping, and reset. Table 22-1 lists the options.

In all modes the T2TMR count register is incremented on the rising edge of the clock signal from the programmable prescaler. When T2TMR equals T2PR then a high level is output to the postscaler counter. T2TMR is cleared on the next clock input.

An external signal from hardware can also be configured to gate the timer operation or force a T2TMR count Reset. In gate modes, the counter stops when the gate is disabled and resumes when the gate is enabled. In Reset modes the T2TMR count is reset on either the level or edge from the external source.

The T2TMR and T2PR registers are both directly readable and writable. The T2TMR register is cleared and the T2PR register initializes to FFh on any device Reset. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the T2TMR register
- a write to the TxCON register
- any device Reset
- External Reset Source event that resets the timer.

Note:	T2TMR is not cleared when TxCON is written.

22.1.1 FREE RUNNING PERIOD MODE

The value of T2TMR is compared to that of the Period register, T2PR, on each clock cycle. When the two values match, the comparator resets the value of T2TMR to 00h on the next cycle and increments the output postscaler counter. When the postscaler count equals the value in the OUTPS bits of the TxCON register, then a one clock period wide pulse occurs on the T2TMR_postscaled output, and the postscaler count is cleared.

22.1.2 ONE-SHOT MODE

The One-Shot mode is identical to the Free Running Period mode except that the ON bit is cleared and the timer is stopped when T2TMR matches T2PR and will not restart until the T2ON bit is cycled off and on. Postscaler OUTPS values other than 0 are meaningless in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

22.1.3 MONOSTABLE MODE

Monostable modes are similar to One-Shot modes except that the ON bit is not cleared and the timer can be restarted by an external Reset event.

22.2 Timer2 Output

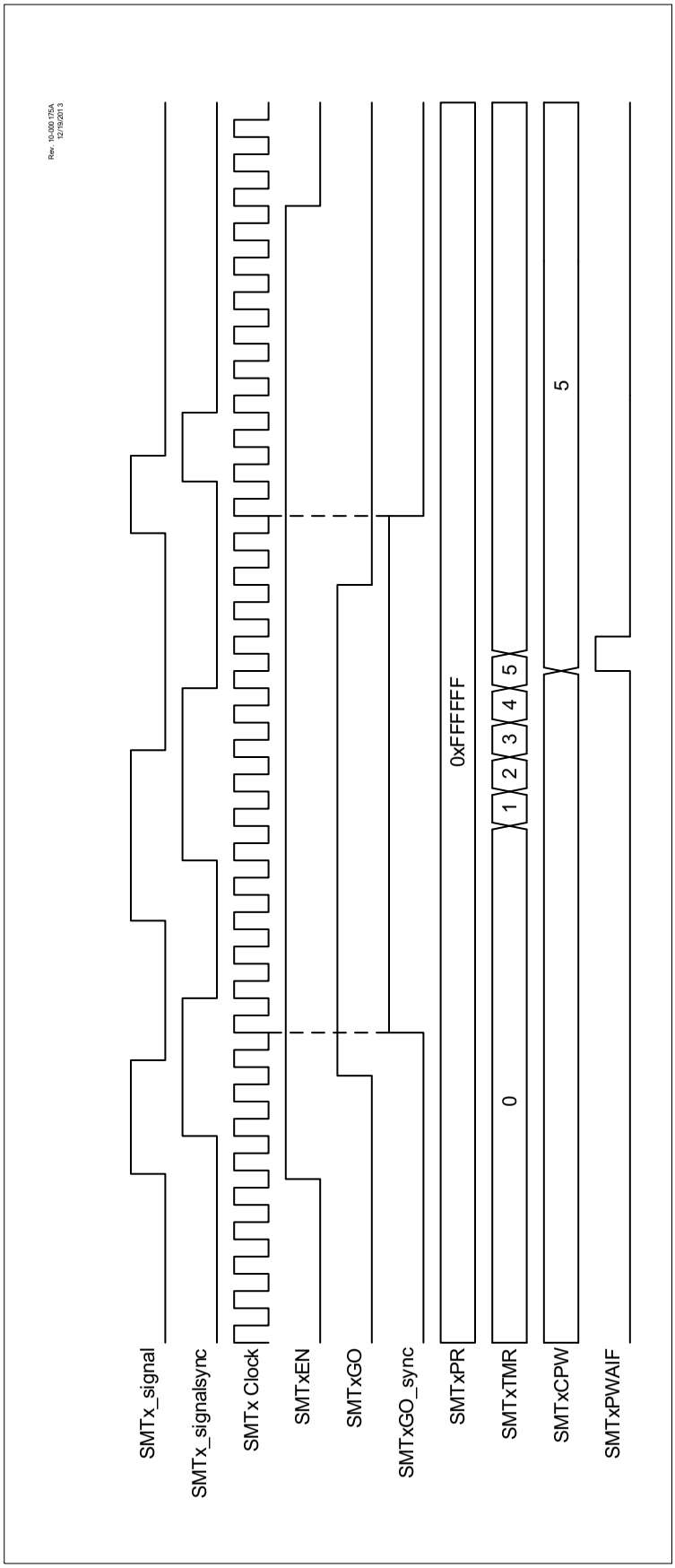
The Timer2 module's primary output is T2TMR_postscaled, which pulses for a single T2TMR_clk period when the postscaler counter matches the value in the OUTPS bits of the TxCON register. The T2PR postscaler is incremented each time the T2TMR value matches the T2PR value. This signal can be selected as an input to several other input modules.

Timer2 is also used by the CCP module for pulse generation in PWM mode. Both the actual T2TMR value as well as other internal signals are sent to the CCP module to properly clock both the period and pulse width of the PWM signal. See **Section 23.0 "Capture/Compare/PWM Module"** for more details on setting up Timer2 for use with the CCP, as well as the timing diagrams in **Section 22.5 "Operation Examples"** for examples of how the varying Timer2 modes affect CCP PWM output.

22.3 External Reset Sources

In addition to the clock source, the Timer2 also takes in an external Reset source. This external Reset source is selected for Timer2, Timer4, and Timer6 with the T2RST, T4RST, and T6RST registers, respectively. This source can control starting and stopping of the timer, as well as resetting the timer, depending on which mode the timer is in. The mode of the timer is controlled by the MODE bits of the T2HLT register. Edge Triggered modes require six Timer clock periods between external triggers. Level Triggered modes require the triggering level to be at least three Timer clock periods long. External triggers are ignored while in Debug Freeze mode.

FIGURE 25-5: GATED TIMER MODE SINGLE ACQUISITION TIMING DIAGRAM



26.3 Clock Source

The clock source is used to drive the dead-band timing circuits. The CWG module allows the following clock sources to be selected:

- FOSC (system clock)
- HFINTOSC

When the HFINTOSC is selected, the HFINTOSC will be kept running during Sleep. Therefore, CWG modes requiring dead band can operate in Sleep, provided that the CWG data input is also active during Sleep.The clock sources are selected using the CS bit of the CWGxCLKCON register (Register 26-3). The system clock FOSC, is disabled in Sleep and thus dead-band control cannot be used.

26.4 Selectable Input Sources

The CWG generates the output waveforms from the following input sources:

TABLE 26-1: SELECTABLE INPUT SOURCES

Source Peripheral	Signal Name	ISM<2:0>
CWGxPPS	Pin selected by CWGxPPS	000
CCP1	CCP1 Output	001
CCP2	CCP2 Output	010
PWM3	PWM3 Output	011
PWM4	PWM4 Output	100
CMP1	Comparator 1 Output	101
CMP2	Comparator 2 Output	110
DSM	Data signal modulator output	111

The input sources are selected using the IS<4:0> bits in the CWGxISM register (Register 26-4).

26.5 Output Control

26.5.1 CWG OUTPUTS

Each CWG output can be routed to a Peripheral Pin Select (PPS) output via the RxyPPS register (see **Section 17.0 "Peripheral Pin Select (PPS) Module"**).

26.5.2 POLARITY CONTROL

The polarity of each CWG output can be selected independently. When the output polarity bit is set, the corresponding output is active-high. Clearing the output polarity bit configures the corresponding output as active-low. However, polarity does not affect the override levels. Output polarity is selected with the POLy bits of the CWGxCON1. Auto-shutdown and steering options are unaffected by polarity.

26.6 Dead-Band Control

The dead-band control provides non-overlapping PWM signals to prevent shoot-through current in PWM switches. Dead-band operation is employed for Half-Bridge and Full-Bridge modes. The CWG contains two 6-bit dead-band counters. One is used for the rising edge of the input source control in Half-Bridge mode or for reverse dead-band Full-Bridge mode. The other is used for the falling edge of the input source control in Half-Bridge mode or for forward dead band in Full-Bridge mode.

Dead band is timed by counting CWG clock periods from zero up to the value in the rising or falling dead-band counter registers. See CWGxDBR and CWGxDBF registers, respectively.

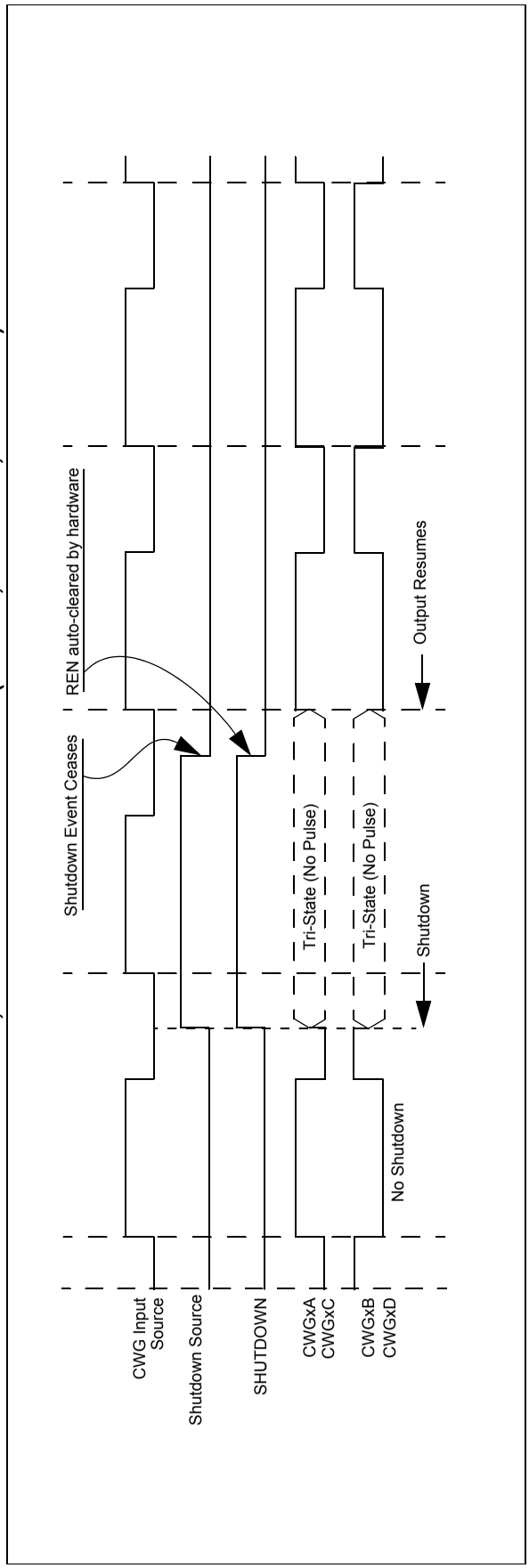
26.6.1 DEAD-BAND FUNCTIONALITY IN HALF-BRIDGE MODE

In Half-Bridge mode, the dead-band counters dictate the delay between the falling edge of the normal output and the rising edge of the inverted output. This can be seen in Figure 26-2.

26.6.2 DEAD-BAND FUNCTIONALITY IN FULL-BRIDGE MODE

In Full-Bridge mode, the dead-band counters are used when undergoing a direction change. The MODE<0> bit of the CWGxCON0 register can be set or cleared while the CWG is running, allowing for changes from Forward to Reverse mode. The CWGxA and CWGxC signals will change immediately upon the first rising input edge following a direction change, but the modulated signals (CWGxB or CWGxD, depending on the direction of the change) will experience a delay dictated by the dead-band counters.

FIGURE 26-16: SHUTDOWN FUNCTIONALITY, AUTO-RESTART ENABLED (REN = 1, LSAC = 01, LSBD = 01)



27.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) module provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 32 input signals and, through the use of configurable gates, reduces the 32 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

The output can be directed internally to peripherals and to an output pin.

There are four CLC modules available on this device - CLC1, CLC2, CLC3 and CLC4.

Note:	The CLC1, CLC2, CLC3 and CLC4 are four separate module instances of the same CLC module design. Throughout this section, the lower case 'x' in register names is a generic reference to the CLC number (which should be substituted with 1, 2, 3, or 4 during code development). For example, the control register is generically described in this chapter as CLCxCON, but the actual device registers are CLC1CON, CLC2CON, CLC3CON and CLC4CON.

Refer to Figure 27-1 for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset

TABLE 27-1: CLCx DATA INPUT SELECTION

DyS<5:0> Value	CLCx Input Source
111111 [63]	Reserved
.	
.	
.	
110100 [52]	Reserved
110011 [51]	CWG3B_out
110010 [50]	CWG3A_out
110001 [49]	CWG2B_out
110000 [48]	CWG2A_out
101111 [47]	CWG1B_out
101110 [46]	CWG1A_out
101101 [45]	SS1
101100 [44]	SCK1
101011 [43]	SDO1
101010 [42]	Reserved
101001 [41]	UART2_tx_out
101000 [40]	UART1_tx_out
100111 [39]	CLC4_out
100110 [38]	CLC3_out
100101 [37]	CLC2_out
100100 [36]	CLC1_out
100011 [35]	DSM1_out
100010 [34]	IOC_flag
100001 [33]	ZCD_out
100000 [32]	CMP2_out
011111 [31]	CMP1_out
011110 [30]	NCO1_out
011101 [29]	Reserved
011100 [28]	Reserved
011011 [27]	PWM8_out
011010 [26]	PWM7_out
011001 [25]	PWM6_out
011000 [24]	PWM5_out
010111 [23]	CCP4_out
010110 [22]	CCP3_out
010101 [21]	CCP2_out
010100 [20]	CCP1 _out
010011 [19]	SMT1_out
010010 [18]	TMR6_out
010001 [17]	TMR5 _overflow
010000 [16]	TMR4 _out
001111 [15]	TMR3 _overflow

TABLE 27-1: CLCx DATA INPUT SELECTION (CONTINUED)

DyS<5:0> Value	CLCx Input Source
001110 [14]	TMR2 _out
001101 [13]	TMR1 _overflow
001100 [12]	TMR0 _overflow
001011 [11]	CLKR _out
001010 [10]	ADCRC
001001 [9]	SOSC
001000 [8]	MFINTOSC (32 kHz)
000111 [7]	MFINTOSC (500 kHz)
000110 [6]	LFINTOSC
000101 [5]	HFINTOSC
000100 [4]	FOSC
000011 [3]	CLCIN3PPS
000010 [2]	CLCIN2PPS
000001 [1]	CLCIN1PPS
000000 [0]	CLCIN0PPS

REGISTER 31-14: UxP2H: UART PARAMETER 2 HIGH REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P2<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 **P2<8>:** Most Significant Bit of Parameter 2
DMX mode:
Most Significant bit of first address of receive block
DALI mode:
Most Significant bit of number of half-bit periods of idle time in Forward Frame detection threshold
Other modes:
Not used

REGISTER 31-15: UxP2L: UART PARAMETER 2 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
P2<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 **P2<7:0>:** Least Significant Bits of Parameter 2
DMX mode:
Least Significant Byte of first address of receive block
LIN Slave mode:
Number of data bytes to transmit
DALI mode:
Least Significant Byte of number of half-bit periods of idle time in Forward Frame detection threshold
Asynchronous Address mode:
Receiver address
Other modes:
Not used

32.3.3 TRANSMIT AND RECEIVE FIFOS

The transmission and reception of data from the SPI module is handled by two FIFOs, one for reception and one for transmission (addressed by the SFRs SPIxRXB and SPIxTXB, respectively.). The TXFIFO is written by software and is read by the SPI module to shift the data onto the SDO pin. The RXFIFO is written by the SPI module as it shifts in the data from the SDI pin and is read by software. Setting the CLRBF bit of SPIxSTATUS resets the occupancy for both FIFOs, emptying both buffers. The FIFOs are also reset by disabling the SPI module.

Note:	TXFIFO occupancy and RXFIFO occupancy simply refer to the number of bytes that are currently being stored in each FIFO. These values are used in this chapter to illustrate the function of these FIFOs and are not directly accessible through software.

The SPIxRXB register addresses the receive FIFO and is read-only. Reading from this register will read from the first FIFO location that was written to by hardware and decrease the RXFIFO occupancy. If the FIFO is empty, reading from this register will instead return a value of zero and set the RXRE (Receive Buffer Read Error) bit of the SPIxSTATUS register. The RXRE bit must then be cleared in software in order to properly reflect the status of the read error. When RXFIFO is full, the RXBF bit of the SPIxSTATUS register will be set. When the device receives data on the SDI pin, the receive FIFO may be written to by hardware and the occupancy increased, depending on the mode and receiver settings, as summarized in Table 32-1.

The SPIxTXB register addresses the transmit FIFO and is write-only. Writing to the register will write to the first empty FIFO location and increase the occupancy. If the FIFO is full, writing to this register will not affect the data and will set the TXWE bit of the SPIxSTATUS register. When the TXFIFO is empty, the TXBE bit of SPIxSTATUS will be set. When a data transfer occurs, data may be read from the first FIFO location written to and the occupancy decreases, depending on mode and transmitter settings, as summarized in Table 32-1 and **Section 32.6.1 "Slave Mode Transmit options"**.

32.3.4 LSB VS. MSB-FIRST OPERATION

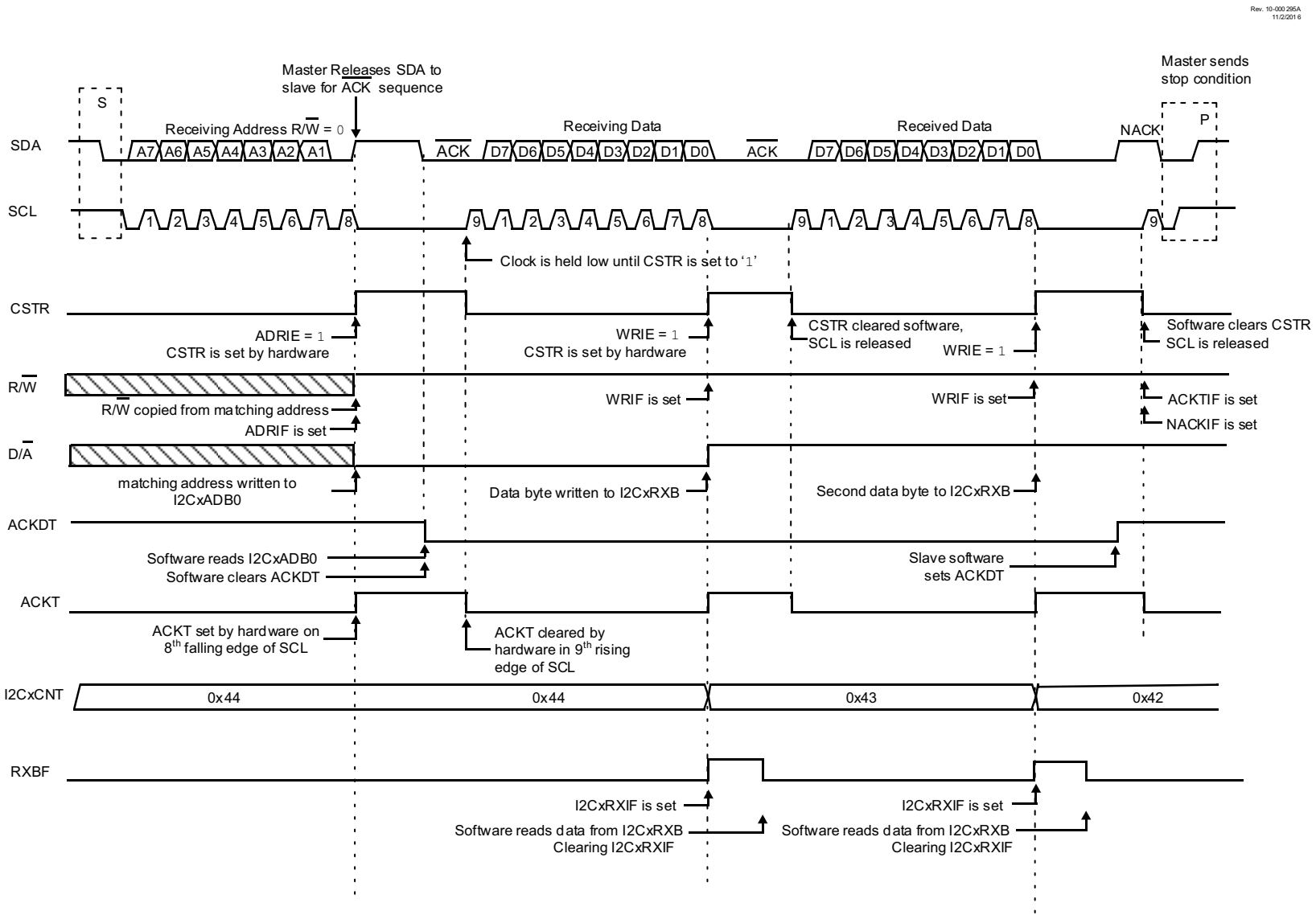
Typically, SPI communication is output Most-Significant bit first, but some devices/buses may not conform to this standard. In this case, the LSBF bit may be used to alter the order in which bits are shifted out during the data exchange. In both Master and Slave mode, the LSBF bit of SPIxCON0 controls if data is shifted MSb or LSb first. Clearing the bit (default) configures the data to transfer MSb first, which is traditional SPI operation, while setting the bit configures the data to transfer LSb first.

32.3.5 INPUT AND OUTPUT POLARITY BITS

SPIxCON1 has three bits that control the polarity of the SPI inputs and outputs. The SDIP bit controls the polarity of the SDI input, the SDOP bit controls the polarity of the SDO output, and the SSP bit controls the polarity of both the slave $\overline{SS}$ input and the master SS output. For all three bits, when the bit is clear, the input or output is active-high, and when the bit is set, the input or output is active-low. When the EN bit of SPIxCON0 is cleared, SS(out) and SCK(out) both revert to the inactive state dictated by their polarity bits. The SDO output state when the EN bit of SPIxCON0 is cleared is determined by several factors.

- When the associated TRIS bit for the SDO pin is cleared, and the SPI goes idle after a transmission, the SDO output will remain at the last bit level. The SDO pin will revert to the Idle state if EN is cleared.
- When the associated TRIS bit for the SDO pin is set, behavior varies in Slave and Master mode.
  - In Slave mode, the SDO pin tri-states when:
  - Slave Select is inactive,
  - the EN bit of SPIxCON0 is cleared, or when
  - the TXR bit of SPIxCON2 is cleared.
  - In Master mode, the SDO pin tri-states when TXR = 0. When TXR = 1 and the SPI goes idle after a transmission, the SDO output will remain at the last bit level. The SDO pin will revert to the Idle state if EN is cleared.

FIGURE 33-8: I²C SLAVE, 7-BIT ADDRESS, RECEPTION NO I2CxCNT (ACKTIE = 0, ADRIE = 1, WRIE = 1)



33.5 I²C Master Mode

Master mode is enabled by setting and clearing the appropriate Mode<2:0> bits in I2CxCON and then by setting the I2CEN bit. Master mode of operation is supported by interrupt generation on buffer full (RXIF), buffer empty (TXIF), and the detection of the Start, Restart, and Stop conditions. The Stop (P), Restart (RS) and Start (S) bits are cleared from a Reset or when the I²C module is disabled. Control of the I²C bus is asserted when the BFRE bit of I2CSTAT0 is set.

33.5.1 I²C MASTER MODE OPERATION

The master device generates all of the serial clock pulses and the Start, Restart, and Stop conditions. A transfer is ended with a Stop condition or with a Restart condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released, and MMA bit will stay set signifying that the Master module is still active.

The steps to initiate a transaction depends on the setting of the address buffer disable bit (ABD) of the I2CxCON2 register.

- ABD = 0 (Address buffers are enabled)

In this case, the master module will use the address stored in the address buffer registers (I2CxADB0/1) to initiate communication with a slave device. User software needs to set the Start bit (S) in the I2CxCON0 register to start communication. This is valid for both 7-bit and 10-bit Addressing modes.

- ABD = 1 (Address buffers are disabled)

In this case, the slave address is transmitted through the transmit buffer and the contents of the address buffers are ignored. User software needs to write the slave address to the transmit buffer (I2CxTXB) to initiate communication. Writing to the Start bit is ignored in this mode. This is valid for both 7-bit and 10-bit Addressing modes.

33.5.1.1 Master Transmitter

In Master Transmitter mode, the first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/W) bit. In the case of master transmitter, the R/W bit will be logic '0'. Serial data is transmitted eight bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

33.5.1.2 Master Receiver

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received eight bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of the transmission.

33.5.2 MASTER CLOCK SOURCE AND ARBITRATION

The I²C module clock source is selected by the I2CxCLK register. The I²C Clock provides the SCL output clock for Master mode and is used by the Bus Free timer. The I²C clock can be sourced from several peripherals.

33.5.3 BUS FREE TIME

In Master modes, the BFRE bit of the I2CxSTAT0 register gives an indication of the bus idle status. The master hardware cannot assert a Start condition until this bit is set by the hardware. This prevents the master from colliding with other masters that may already be talking on the bus. The BFRET<1:0> bits of I2CxCON1 allow selection of 8 to 64 pulses of the I²C clock input before asserting the BFRE bit. The BFRET bits are used to ensure that the I²C module always follows the minimum Stop Hold Time. The I²C timing requirements are listed in the electrical specifications chapter.

Note:	I²C clock is not required to have a 50% duty cycle.

33.5.4 MASTER CLOCK TIMING

The clock generation in the I²C module can be configured using the Fast Mode Enable (FME) bit of the I2CxCON2 register. This bit controls the number of times the SCL pin is sampled before the master hardware drives it.

33.5.4.1 Clock Timing with FME = 0

One Tscl, consists of five clocks of the I²C clock input. The first clock is used to drive SCL low, the third releases SCL high. The fourth and fifth clocks are used to detect if the SCL pin is, in fact, high or being stretched by a slave.

If a slave is clock stretching, the hardware waits; checking SCL on each successive I²C clock, proceeding only after detecting SCL high. Figure 33-13 shows the clock synthesis timing when FME = 0.

41.0 INSTRUCTION SET SUMMARY

PIC18(L)F26/27/45/46/47/55/56/57K42 devices incorporate the standard set of PIC18 core instructions, as well as an extended set of instructions, for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

41.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC® MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two-program memory locations and two that require three-program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 41-3 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 41-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator ‘f’ specifies which file register is to be used by the instruction. The destination designator ‘d’ specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator ‘b’ selects the number of the bit affected by the operation, while the file register designator ‘f’ represents the number of the file in which the bit is located.

The literal instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The control instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the four MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 41-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 41-3, lists the standard instructions recognized by the Microchip Assembler (MPASM™).

Section 41.1.1 “Standard Instruction Set” provides a description of each instruction.

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh).
See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

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

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

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

BTFSS **Bit Test File, Skip if Set**

Syntax: BTFSS f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh).
See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

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

Q Cycle Activity:

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

If skip:

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

If skip and followed by 2-word instruction:

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

Example:
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

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

BTG **Bit Toggle f**

Syntax: BTG f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit 'b' in data memory location 'f' is inverted.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `BTG PORTC, 4, 0`

Before Instruction:
PORTC = 0111 0101 [75h]
After Instruction:
PORTC = 0110 0101 [65h]

BOV **Branch if Overflow**

Syntax: BOV n

Operands: $-128 \le n \le 127$

Operation: if OVERFLOW bit is '1'
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the OVERFLOW bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BOV Jump`

Before Instruction
PC = address (HERE)
After Instruction
If OVERFLOW = 1;
PC = address (Jump)
If OVERFLOW = 0;
PC = address (HERE + 2)

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
3A16h	RC6PPS	—	—	—	RC6PPS4	RC6PPS3	RC6PPS2	RC6PPS1	RC6PPS0	280
3A15h	RC5PPS	—	—	—	RC5PPS4	RC5PPS3	RC5PPS2	RC5PPS1	RC5PPS0	280
3A14h	RC4PPS	—	—	—	RC4PPS4	RC4PPS3	RC4PPS2	RC4PPS1	RC4PPS0	280
3A13h	RC3PPS	—	—	—	RC3PPS4	RC3PPS3	RC3PPS2	RC3PPS1	RC3PPS0	280
3A12h	RC2PPS	—	—	—	RC2PPS4	RC2PPS3	RC2PPS2	RC2PPS1	RC2PPS0	280
3A11h	RC1PPS	—	—	—	RC1PPS4	RC1PPS3	RC1PPS2	RC1PPS1	RC1PPS0	280
3A10h	RC0PPS	—	—	—	RC0PPS4	RC0PPS3	RC0PPS2	RC0PPS1	RC0PPS0	280
3A0Fh	RB7PPS	—	—	—	RB7PPS4	RB7PPS3	RB7PPS2	RB7PPS1	RB7PPS0	280
3A0Eh	RB6PPS	—	—	—	RB6PPS4	RB6PPS3	RB6PPS2	RB6PPS1	RB6PPS0	280
3A0Dh	RB5PPS	—	—	—	RB5PPS4	RB5PPS3	RB5PPS2	RB5PPS1	RB5PPS0	280
3A0Ch	RB4PPS	—	—	—	RB4PPS4	RB4PPS3	RB4PPS2	RB4PPS1	RB4PPS0	280
3A0Bh	RB3PPS	—	—	—	RB3PPS4	RB3PPS3	RB3PPS2	RB3PPS1	RB3PPS0	280
3A0Ah	RB2PPS	—	—	—	RB2PPS4	RB2PPS3	RB2PPS2	RB2PPS1	RB2PPS0	280
3A09h	RB1PPS	—	—	—	RB1PPS4	RB1PPS3	RB1PPS2	RB1PPS1	RB1PPS0	280
3A08h	RB0PPS	—	—	—	RB0PPS4	RB0PPS3	RB0PPS2	RB0PPS1	RB0PPS0	280
3A07h	RA7PPS	—	—	—	RA7PPS4	RA7PPS3	RA7PPS2	RA7PPS1	RA7PPS0	280
3A06h	RA6PPS	—	—	—	RA6PPS4	RA6PPS3	RA6PPS2	RA6PPS1	RA6PPS0	280
3A05h	RA5PPS	—	—	—	RA5PPS4	RA5PPS3	RA5PPS2	RA5PPS1	RA5PPS0	280
3A04h	RA4PPS	—	—	—	RA4PPS4	RA4PPS3	RA4PPS2	RA4PPS1	RA4PPS0	280
3A03h	RA3PPS	—	—	—	RA3PPS4	RA3PPS3	RA3PPS2	RA3PPS1	RA3PPS0	280
3A02h	RA2PPS	—	—	—	RA2PPS4	RA2PPS3	RA2PPS2	RA2PPS1	RA2PPS0	280
3A01h	RA1PPS	—	—	—	RA1PPS4	RA1PPS3	RA1PPS2	RA1PPS1	RA1PPS0	280
3A00h	RA0PPS	—	—	—	RA0PPS4	RA0PPS3	RA0PPS2	RA0PPS1	RA0PPS0	280
39FFh - 39F8h	—	Unimplemented								
39F7h	SCANPR	—	—	—	—	—	PR			31
39F6h - 39F5h	—	Unimplemented								
39F4h	DMA2PR	—	—	—	—	—	PR			31
39F3h	DMA1PR	—	—	—	—	—	PR			30
39F2h	MAINPR	—	—	—	—	—	PR			30
39F1h	ISRPR	—	—	—	—	—	PR			30
39F0h	—	Unimplemented								
39EFh	PRLOCK	—	—	—	—	—	—	—	PRLOCKED	31
39EEh - 39E7h	—	Unimplemented								
39E6h	NVMCON2	NVMCON2								211
39E5h	NVMCON1	REG		—	FREE	WRERR	WREN	WR	RD	210
39E4h	—	Unimplemented								
39E3h	NVMDAT	DAT								212
39E2h	—	Unimplemented								
39E1h	NVMADRH(4)	—	—	—	—	—	—	ADR		211
39E0h	NVMADRL	ADR								211
39DFh	OSCFRQ	—	—	—	—	FRQ				107
39DEh	OSCTUNE	—	—	TUN						108
39DDh	OSCEN	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—	109
39DCh	OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR	106
39DBh	OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	105

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

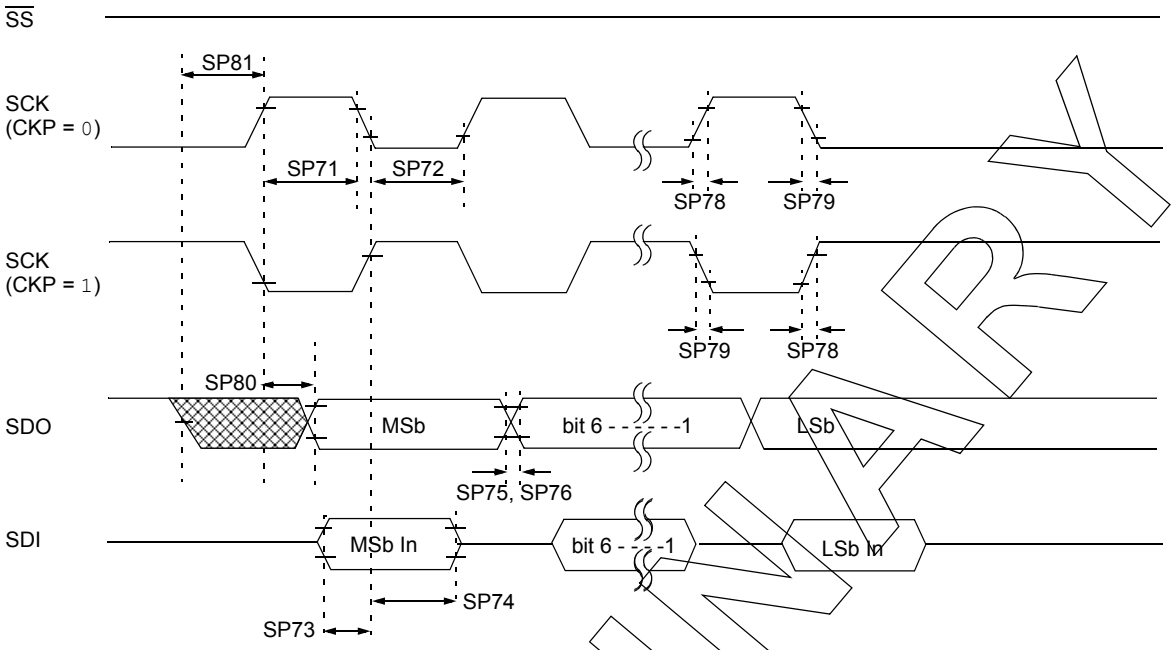
Note 1: Unimplemented in LF devices.

2: Unimplemented in PIC18(L)F26/27K42.

3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.

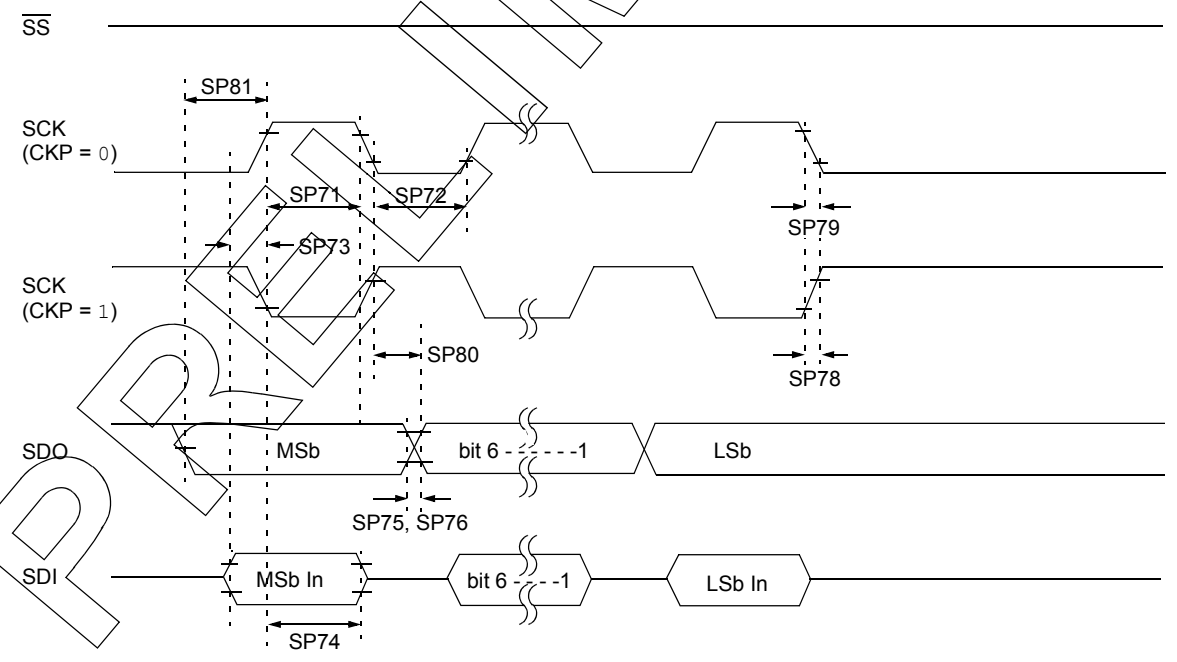
4: Unimplemented in PIC18(L)F45/55K42.

FIGURE 44-14: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)



Note: Refer to Figure 44-4 for load conditions.

FIGURE 44-15: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)



Note: Refer to Figure 44-4 for load conditions.