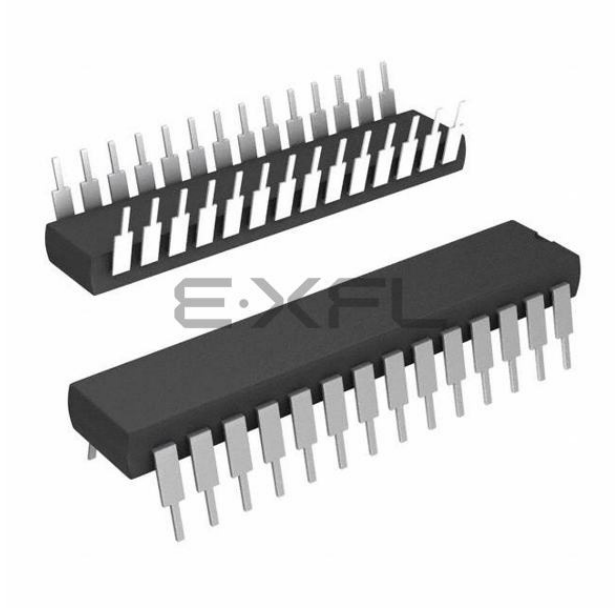


Welcome to E-XFL.COM

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	CANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	25
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)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f26k83-i-sp

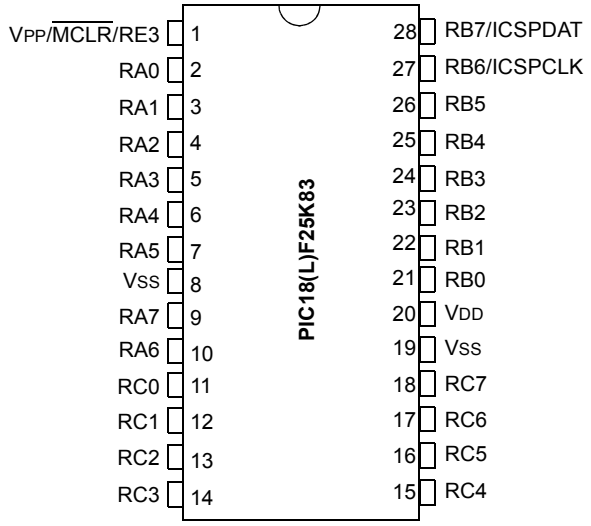
TABLE 2: PACKAGES

Device	SPDIP	SOIC	SSOP	UQFN	QFN
PIC18(L)F25K83	•	•	•	•	•
PIC18(L)F26K83	•	•	•	•	•

Note 1: Pin details are subject to change.

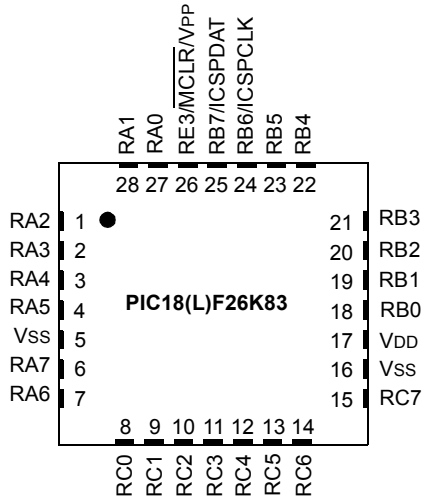
Pin Diagrams

28-pin SPDIP, SOIC, SSOP



Note: See Table 3 for location of all peripheral functions.

28-pin QFN (6x6x0.9mm), UQFN (4x4x0.5mm)



Note 1: See Table 3 for location of all peripheral functions.

2: It is recommended that the exposed bottom pad be connected to VSS, however it must not be the only VSS connection to the device.

4.5.5 STATUS REGISTER

The STATUS register, shown in Register 4-2, contains the arithmetic status of the ALU. As with any other SFR, it can be the operand for any instruction.

If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, the results of the instruction are not written; instead, the STATUS register is updated according to the instruction performed. Therefore, the result of an instruction with the STATUS register as its destination may be different than intended. As an example, `CLRF STATUS` will set the Z bit and leave the remaining Status bits unchanged (‘`0uuu u1uu`’).

It is recommended that only `BCF`, `BSF`, `SWAPF`, `MOVFF`, `MOVWF` and `MOVFFL` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions that do not affect Status bits, see the instruction set summaries in **Section 42.2 “Extended Instruction Set”** and Table 42-3.

Note:	The C and DC bits operate as the borrow and digit borrow bits, respectively, in subtraction.

4.5.6 CALL SHADOW REGISTER

When `CALL`, `CALLW`, `RCALL` instructions are used, the WREG, BSR and STATUS are automatically saved in hardware and can be accessed using the WREG_CSHAD, BSR_CSHAD and STATUS_CSHAD registers.

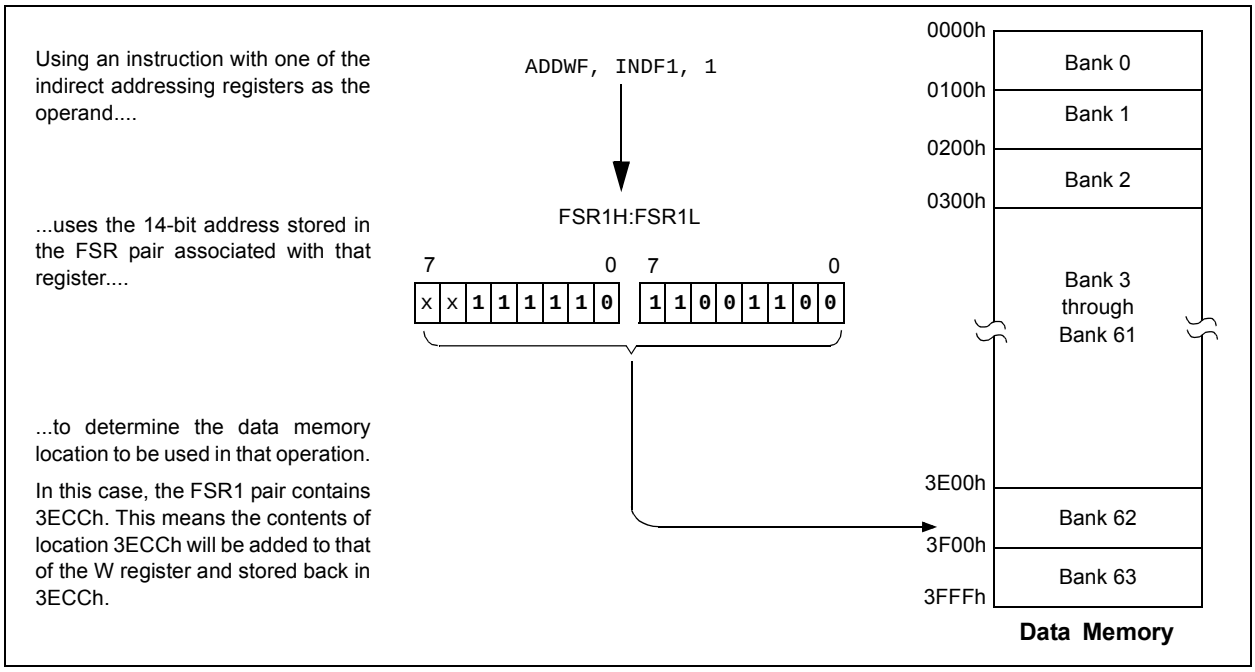
4.7.3.2 FSR Registers, POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers which cannot be directly read or written. Accessing these registers actually accesses the location to which the associated FSR register pair points, and also performs a specific action on the FSR value. They are:

- POSTDEC: accesses the location to which the FSR points, then automatically decrements the FSR by 1 afterwards
- POSTINC: accesses the location to which the FSR points, then automatically increments the FSR by 1 afterwards
- PREINC: automatically increments the FSR by 1, then uses the location to which the FSR points in the operation
- PLUSW: adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the location to which the result points in the operation.

In this context, accessing an INDF register uses the value in the associated FSR register without changing it. Similarly, accessing a PLUSW register gives the FSR value an offset by that in the W register; however, neither W nor the FSR is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR register.

FIGURE 4-6: INDIRECT ADDRESSING



5.2 Register Definitions: Configuration Words

REGISTER 5-1: CONFIGURATION WORD 1L (30 0000h)

U-1	R/W-1	R/W-1	R/W-1	U-1	R/W-1	R/W-1	R/W-1
—	RSTOSC<2:0>			—	FEXTOSC<2:0>		
bit 7							bit 0

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

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

bit 6-4 **RSTOSC<2:0>:** Power-up Default Value for COSC bits
- 111 = EXTOSC operating per FEXTOSC<2:0> bits
- 110 = HFINTOSC with HFFRQ = 4 MHz and CDIV = 4:1
- 101 = LFINTOSC
- 100 = SOSC
- 011 = Reserved
- 010 = EXTOSC with 4x PLL, with EXTOSC operating per FEXTOSC<2:0> bits
- 001 = Reserved
- 000 = HFINTOSC with HFFRQ = 64 MHz and CDIV = 1:1; resets COSC/NOSC to 3'b110

bit 3 **Unimplemented:** Read as '1'

bit 2-0 **FEXTOSC<2:0>:** FEXTOSC External Oscillator Mode Selection bits
- 111 = EC (External Clock) above 8 MHz; PFM set to high power
- 110 = EC (External Clock) for 500 kHz to 8 MHz; PFM set to medium power
- 101 = EC (External Clock) below 500 kHz; PFM set to low power
- 100 = Oscillator is not enabled
- 011 = Reserved (do not use)
- 010 = HS (crystal oscillator) above 8 MHz; PFM set to high power
- 001 = XT (crystal oscillator) above 500 kHz, below 8 MHz; PFM set to medium power
- 000 = LP (crystal oscillator) optimized for 32.768 kHz; PFM set to low power

6.11 Start-up Sequence

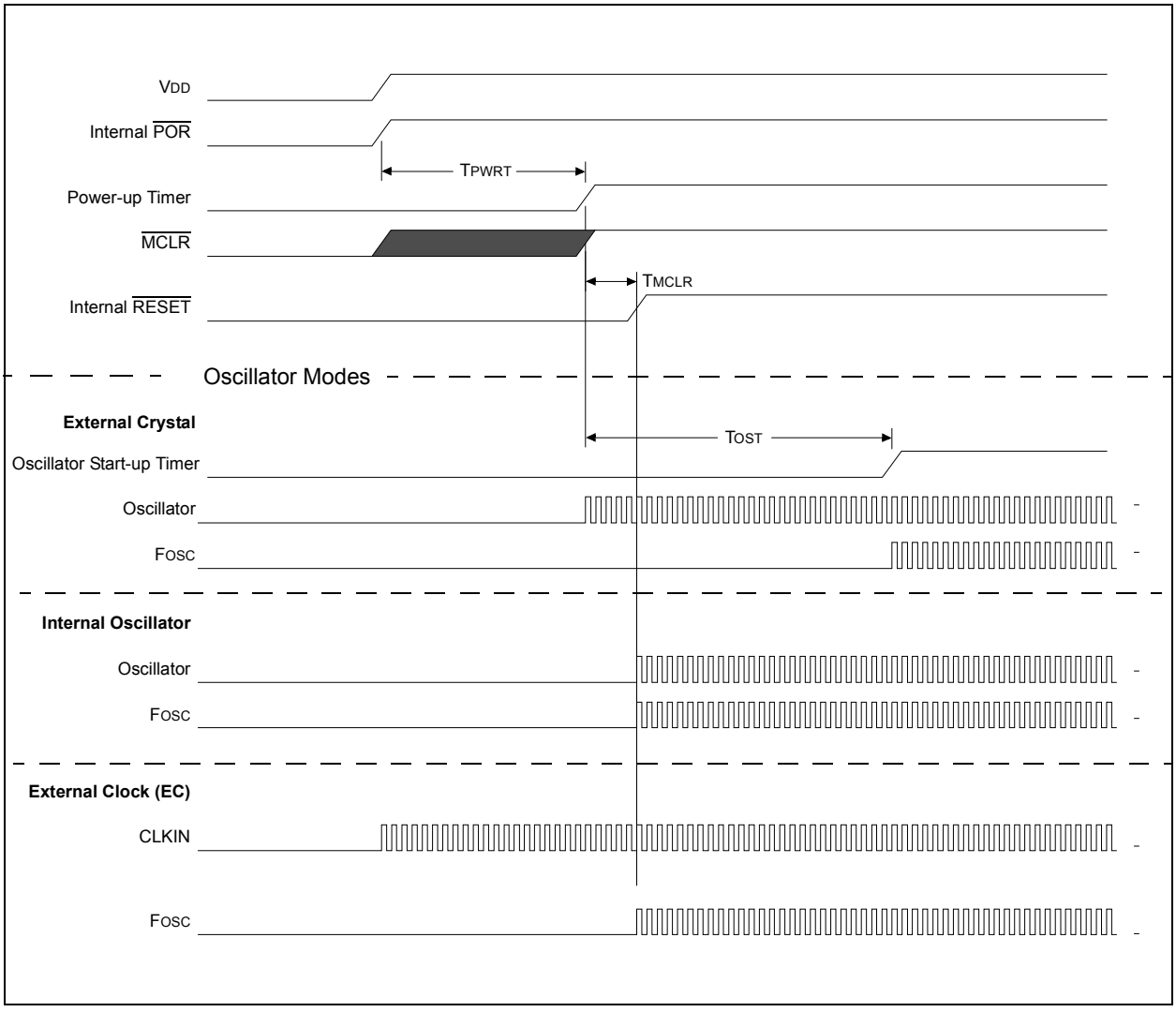
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. Oscillator start-up timer runs to completion (if required for selected oscillator source).
3. $\overline{MCLR}$ must be released (if enabled).

The total time out will vary based on oscillator configuration and Power-up Timer configuration. See **Section 7.0 "Oscillator Module (with Fail-Safe Clock Monitor)"** for more information.

The Power-up Timer and oscillator start-up timer run independently of $\overline{MCLR}$ Reset. If $\overline{MCLR}$ is kept low long enough, the Power-up Timer and oscillator Start-up Timer will expire. Upon bringing $\overline{MCLR}$ high, the device will begin execution after 10 FOSC cycles (see Figure 6-4). This is useful for testing purposes or to synchronize more than one device operating in parallel.

FIGURE 6-4: RESET START-UP SEQUENCE



12.0 8x8 HARDWARE MULTIPLIER

12.1 Introduction

All PIC18 devices include an 8x8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 12-1.

12.2 Operation

Example 12-1 shows the instruction sequence for an 8x8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 12-2 shows the sequence to do an 8x8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 12-1: 8x8 UNSIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

EXAMPLE 12-2: 8x8 SIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG2
```

TABLE 12-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 40 MHz	@ 10 MHz	@ 4 MHz
8x8 unsigned	Without hardware multiply	13	69	4.3 μs	6.9 μs	27.6 μs	69 μs
	Hardware multiply	1	1	62.5 ns	100 ns	400 ns	1 μs
8x8 signed	Without hardware multiply	33	91	5.7 μs	9.1 μs	36.4 μs	91 μs
	Hardware multiply	6	6	375 ns	600 ns	2.4 μs	6 μs
16x16 unsigned	Without hardware multiply	21	242	15.1 μs	24.2 μs	96.8 μs	242 μs
	Hardware multiply	28	28	1.8 μs	2.8 μs	11.2 μs	28 μs
16x16 signed	Without hardware multiply	52	254	15.9 μs	25.4 μs	102.6 μs	254 μs
	Hardware multiply	35	40	2.5 μs	4.0 μs	16.0 μs	40 μs

EXAMPLE 13-4: WRITING TO PROGRAM FLASH MEMORY

```
                MOVLW     D'64'                 ; number of bytes in erase block
                MOVWF     COUNTER
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     CODE_ADDR_UPPER       ; Load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVF      TABLAT, W             ; get data
                MOVWF     POSTINC0              ; store data
                DECFSZ    COUNTER               ; done?
                BRA       READ_BLOCK            ; repeat
MODIFY_WORD
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     NEW_DATA_LOW          ; update buffer word
                MOVWF     POSTINC0
                MOVLW     NEW_DATA_HIGH
                MOVWF     INDF0
ERASE_BLOCK
                MOVLW     CODE_ADDR_UPPER       ; load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
                BCF       NVMCON1, REG0         ; point to Program Flash Memory
                BSF       NVMCON1, REG1         ; point to Program Flash Memory
                BSF       NVMCON1, WREN         ; enable write to memory
                BSF       NVMCON1, FREE         ; enable Erase operation
                BCF       INTCON0, GIE          ; disable interrupts
                MOVLW     55h
Required        MOVWF     NVMCON2               ; write 55h
Sequence        MOVLW     AAh
                MOVWF     NVMCON2               ; write 0AAh
                BSF       NVMCON1, WR           ; start erase (CPU stall)
                BSF       INTCON0, GIE          ; re-enable interrupts
                TBLRD*-                         ; dummy read decrement
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
WRITE_BUFFER_BACK
                MOVLW     BlockSize             ; number of bytes in holding register
                MOVWF     COUNTER
                MOVLW     D'64'/BlockSize       ; number of write blocks in 64 bytes
                MOVWF     COUNTER2
```

20.8 Register Definitions: Timer0 Control

REGISTER 20-1: T0CON0: TIMER0 CONTROL REGISTER 0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	MD16	OUTPS<3:0>			
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 **EN:** TMR0 Enable bit
1 = The module is enabled and operating
0 = The module is disabled and in the lowest power mode

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **OUT:** TMR0 Output bit (read-only)
TMR0 output bit

bit 4 **MD16:** TMR0 Operating as 16-Bit Timer Select bit
1 = TMR0 is a 16-bit timer
0 = TMR0 is an 8-bit timer

bit 3-0 **OUTPS<3:0>:** TMR0 Output Postscaler (Divider) Select bits
1111 = 1:16 Postscaler
1110 = 1:15 Postscaler
1101 = 1:14 Postscaler
1100 = 1:13 Postscaler
1011 = 1:12 Postscaler
1010 = 1:11 Postscaler
1001 = 1:10 Postscaler
1000 = 1:9 Postscaler
0111 = 1:8 Postscaler
0110 = 1:7 Postscaler
0101 = 1:6 Postscaler
0100 = 1:5 Postscaler
0011 = 1:4 Postscaler
0010 = 1:3 Postscaler
0001 = 1:2 Postscaler
0000 = 1:1 Postscaler

REGISTER 20-2: T0CON1: TIMER0 CONTROL REGISTER 1

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
CS<2:0>			ASYNC	CKPS<3:0>			
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 **CS<2:0>:** Timer0 Clock Source Select bits
111 = CLC1
110 = SOSC
101 = MFINTOSC (500 kHz)
100 = LFINTOSC
011 = HFINTOSC
010 = FOSC/4
001 = Pin selected by T0CKIPPS (Inverted)
000 = Pin selected by T0CKIPPS (Non-inverted)

bit 4 **ASYNC:** TMR0 Input Asynchronization Enable bit
1 = The input to the TMR0 counter is not synchronized to system clocks
0 = The input to the TMR0 counter is synchronized to FOSC/4

bit 3-0 **CKPS<3:0>:** Prescaler Rate Select bit
1111 = 1:32768
1110 = 1:16384
1101 = 1:8192
1100 = 1:4096
1011 = 1:2048
1010 = 1:1024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

22.0 TIMER2/4/6 MODULE

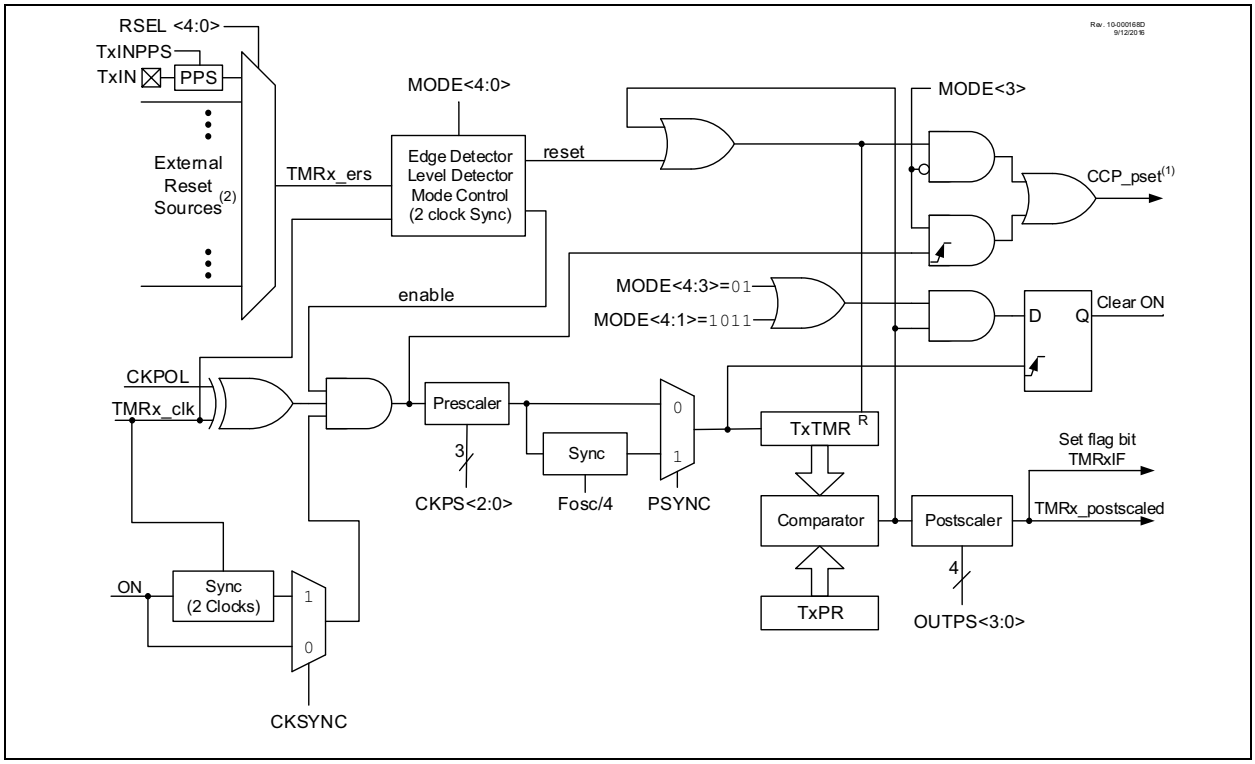
The Timer2/4/6 modules are 8-bit timers that can operate as free-running period counters or in conjunction with external signals that control start, run, freeze, and reset operation in One-Shot and Monostable modes of operation. Sophisticated waveform control such as pulse density modulation are possible by combining the operation of these timers with other internal peripherals such as the comparators and CCP modules. Features of the timer include:

- 8-bit timer register
- 8-bit period register
- Selectable external hardware timer resets
- Programmable prescaler (1:1 to 1:128)
- Programmable postscaler (1:1 to 1:16)
- Selectable synchronous/asynchronous operation
- Alternate clock sources
- Interrupt on period
- Three modes of operation:
  - Free Running Period
  - One-Shot
  - Monostable

See Figure 22-1 for a block diagram of Timer2. See Figure 22-2 for the clock source block diagram.

Note:	Three identical Timer2 modules are implemented on this device. The timers are named Timer2, Timer4, and Timer6. All references to Timer2 apply as well to Timer4 and Timer6. All references to T2PR apply as well to T4PR and T6PR.

FIGURE 22-1: TIMER2 BLOCK DIAGRAM



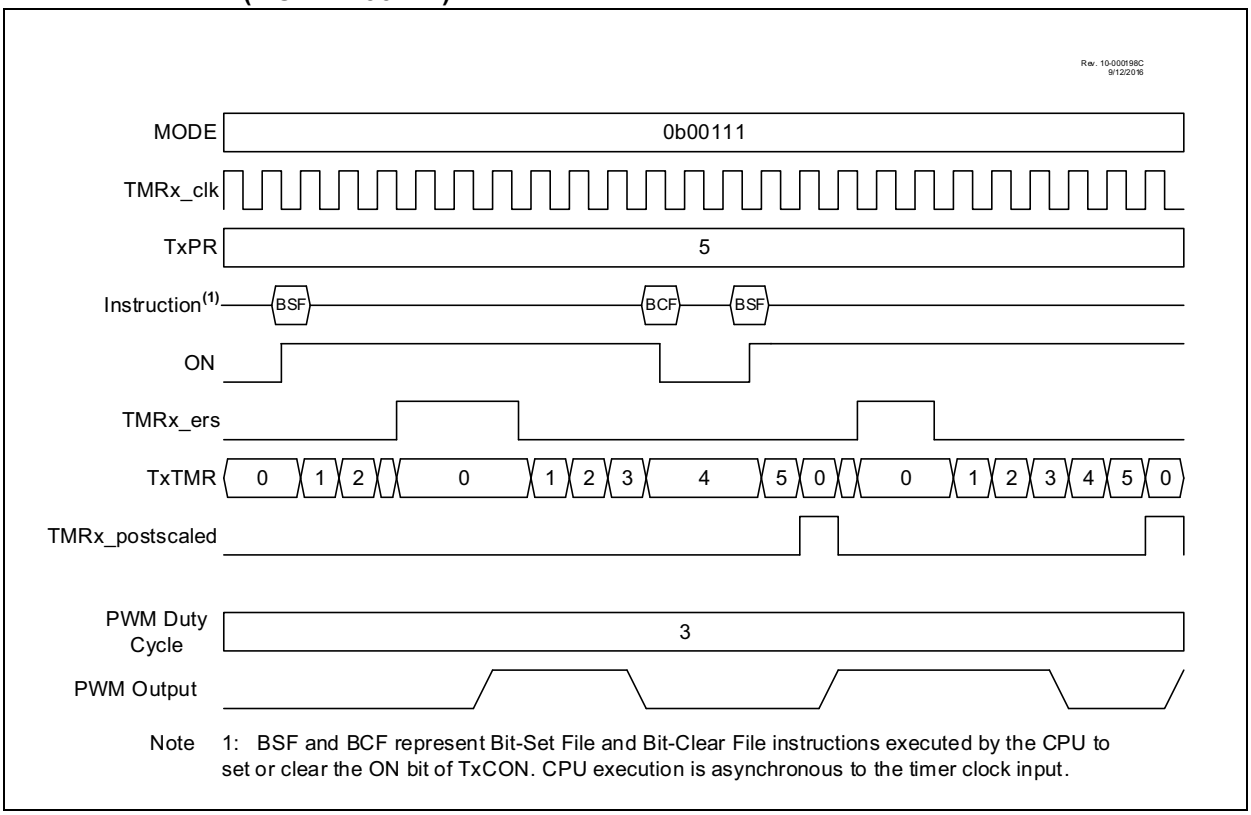
22.5.4 LEVEL-TRIGGERED HARDWARE LIMIT MODE

In the level triggered Hardware Limit Timer modes the counter is reset by high or low levels of the external signal TMR2_ers, as shown in Figure 22-7. Selecting MODE<4:0> = 00110 will cause the timer to reset on a low level external signal. Selecting MODE<4:0> = 00111 will cause the timer to reset on a high level external signal. In the example, the counter is reset while TMR2_ers = 1. ON is controlled by BSF and BCF instructions. When ON=0 the external signal is ignored.

When the CCP uses the timer as the PWM time base then the PWM output will be set high when the timer starts counting and then set low only when the timer count matches the CCPRx value. The timer is reset when either the timer count matches the T2PR value or two clock periods after the external Reset signal goes true and stays true.

The timer starts counting, and the PWM output is set high, on either the clock following the T2PR match or two clocks after the external Reset signal relinquishes the Reset. The PWM output will remain high until the timer counts up to match the CCPRx pulse width value. If the external Reset signal goes true while the PWM output is high then the PWM output will remain high until the Reset signal is released allowing the timer to count up to match the CCPRx value.

FIGURE 22-7: LEVEL TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00111)



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.

REGISTER 22-6: TxHLT: TIMERx HARDWARE LIMIT CONTROL 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
PSYNC	CKPOL	CKSYNC	MODE<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 **PSYNC:** Timerx Prescaler Synchronization Enable bit(1, 2)
1 = TxTMR Prescaler Output is synchronized to Fosc/4
0 = TxTMR Prescaler Output is not synchronized to Fosc/4

bit 6 **CKPOL:** Timerx Clock Polarity Selection bit(3)
1 = Falling edge of input clock clocks timer/prescaler
0 = Rising edge of input clock clocks timer/prescaler

bit 5 **CKSYNC:** Timerx Clock Synchronization Enable bit(4, 5)
1 = ON register bit is synchronized to T2TMR_clk input
0 = ON register bit is not synchronized to T2TMR_clk input

bit 4-0 **MODE<4:0>:** Timerx Control Mode Selection bits(6, 7)
See Table 22-1 for all operating modes.

Note 1: Setting this bit ensures that reading TxTMR will return a valid data value.
2: When this bit is '1', Timer2 cannot operate in Sleep mode.
3: CKPOL should not be changed while ON = 1.
4: Setting this bit ensures glitch-free operation when the ON is enabled or disabled.
5: When this bit is set then the timer operation will be delayed by two TxTMR input clocks after the ON bit is set.
6: Unless otherwise indicated, all modes start upon ON = 1 and stop upon ON = 0 (stops occur without affecting the value of TxTMR).
7: When TxTMR = TxPR, the next clock clears TxTMR, regardless of the operating mode.

FIGURE 26-1: SIMPLIFIED CWG BLOCK DIAGRAM (HALF-BRIDGE MODE, MODE<2:0> = 100)

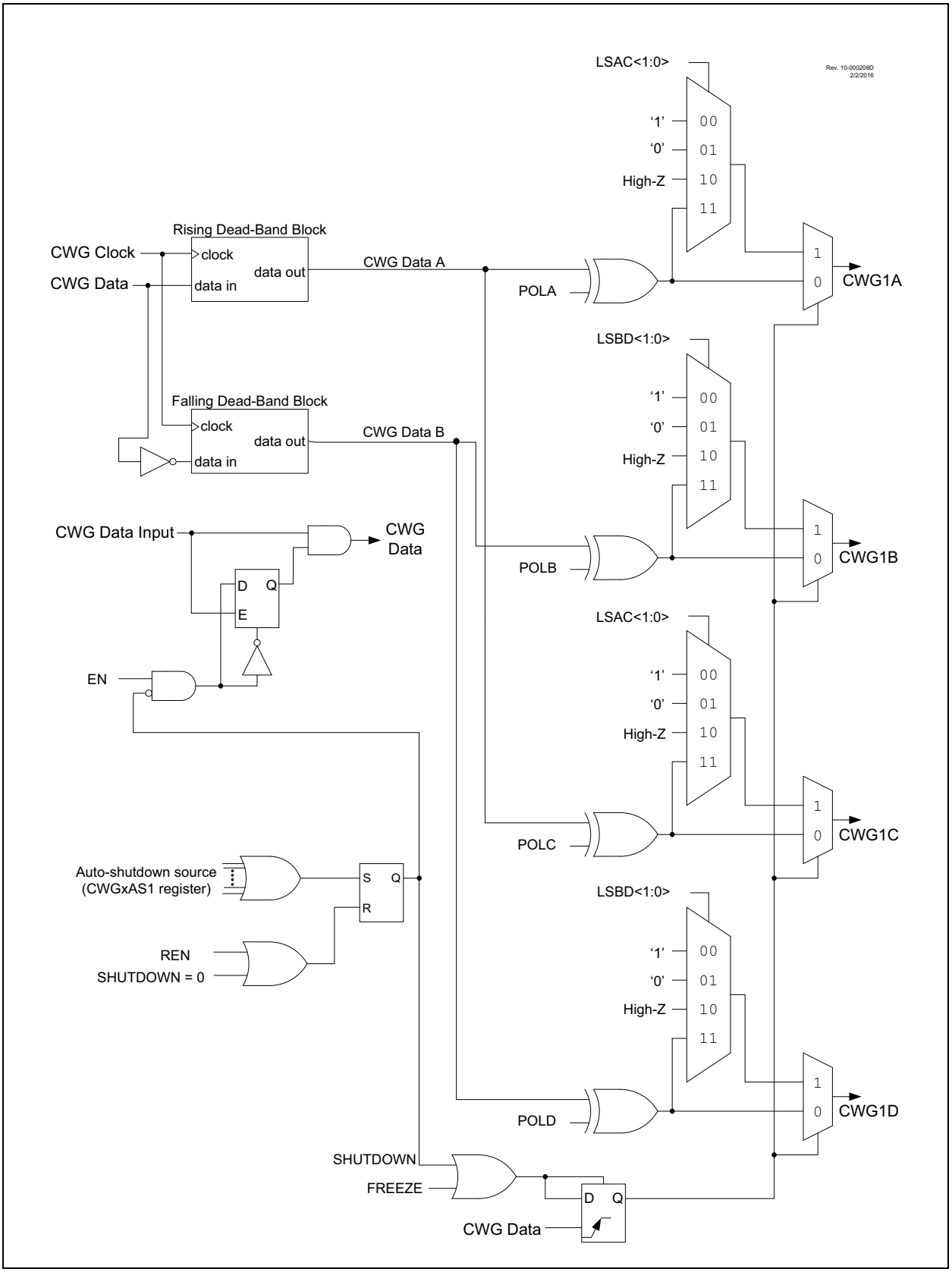


FIGURE 32-12: TRADITIONAL SPI DAISY – CHAIN CONNECTION

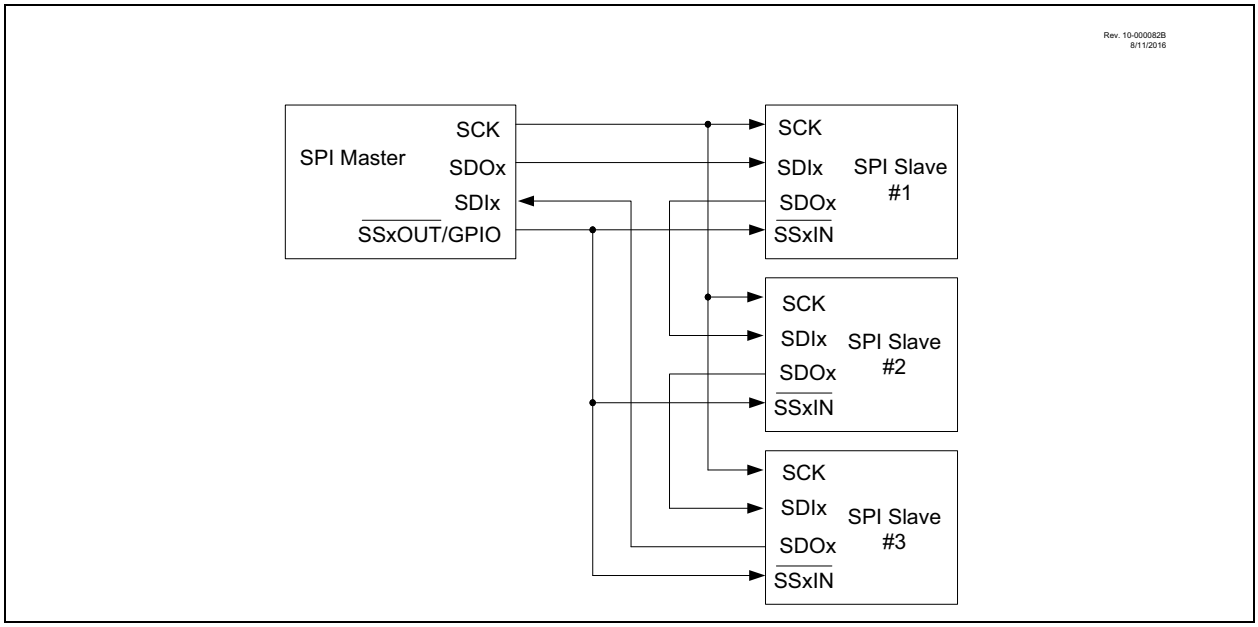


FIGURE 32-13: SPI DAISY-CHAIN CONNECTION WITH CHAINED SCK

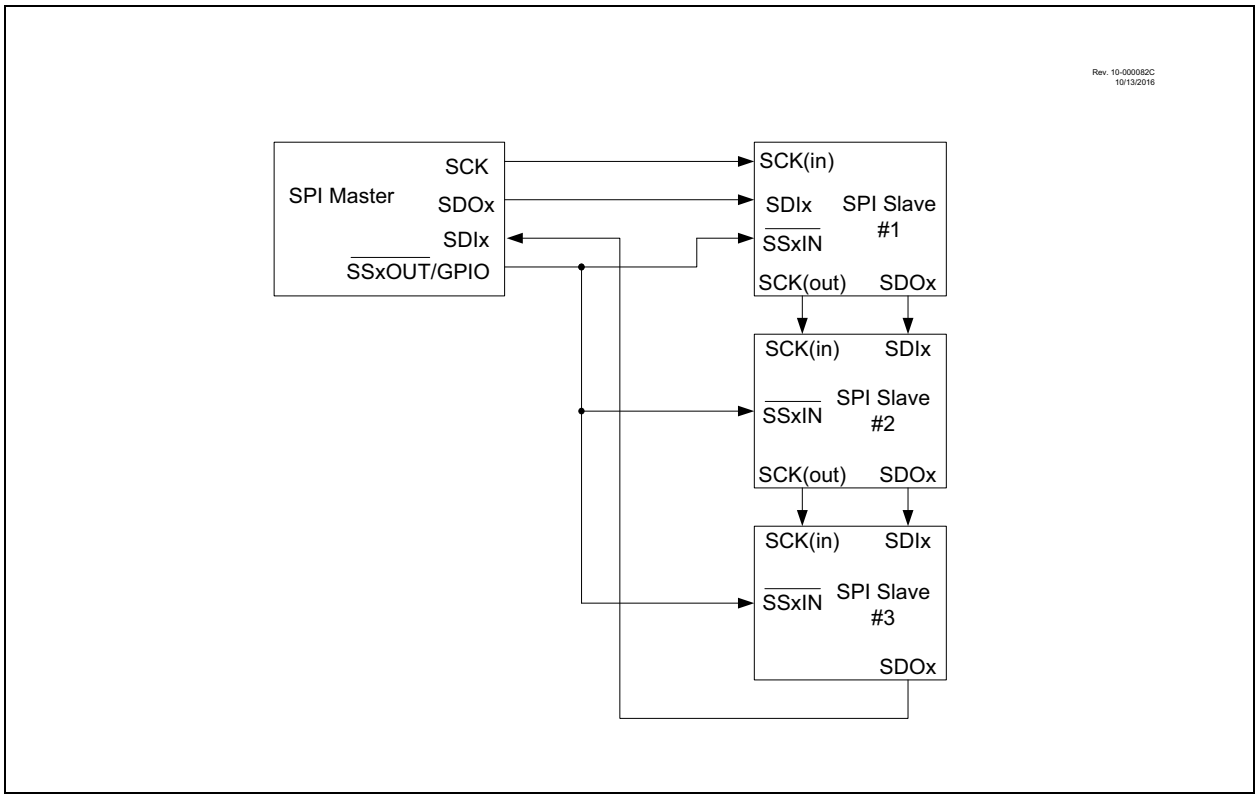


TABLE 33-18: SUMMARY OF REGISTERS FOR I²C 8-BIT MACRO

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
I2CxBTO	—	—	—	—	—	BTO<2:0>			567
I2CxCLK	—	—	—	—	—	CLK<2:0>			566
I2CxPIE	CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE	573
I2CxPIR	CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF	572
I2CxERR	—	BTOIF	BCLIF	NACKIF	—	BTOIE	BCLIE	NACKIE	570
I2CxSTAT0	BFRE	SMA	MMA	R	D	—	—	—	568
I2CxSTAT1	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	569
I2CxCON0	EN	RSEN	S	CSTR	MDR	MODE<2:0>			562
I2CxCON1	ACKCNT	ACKDT	ACKSTAT	ACKT	—	RXOV	TXU	CSD	564
I2CxCON2	ACNT	GCEN	FME	ADB	SDAHT<3:2>		BFRET<1:0>		565
I2CxADR0	ADR<7:0>								574
I2CxADR1	ADR<7:1>							—	575
I2CxADR2	ADR<7:0>								576
I2CxADR3	ADR<7:1>							—	577
I2CxADB0	ADB<7:0>								578
I2CxADB1	ADB<7:0>								579
I2CxCNT	CNT<7:0>								571
I2CxRXB	RXB<7:0>								—
I2CxTXB	TXB<7:0>								—

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the I²C module.

EXAMPLE 34-3: TRANSMITTING A CAN MESSAGE USING BANKED METHOD

```
    ; Need to transmit Standard Identifier message 123h using TXB0 buffer.
    ; To successfully transmit, CAN module must be either in Normal or Loopback mode.
    ; TXB0 buffer is not in access bank. And since we want banked method, we need to make sure
    ; that correct bank is selected.
    BANKSEL TXB0CON                 ; One BANKSEL in beginning will make sure that we are
                                    ; in correct bank for rest of the buffer access.
    ; Now load transmit data into TXB0 buffer.
    MOVLW   MY_DATA_BYTE1           ; Load first data byte into buffer
    MOVWF   TXB0D0                  ; Compiler will automatically set “BANKED” bit
    ; Load rest of data bytes - up to 8 bytes into TXB0 buffer.
    ...
    ; Load message identifier
    MOVLW   60H                     ; Load SID2:SID0, EXIDE = 0
    MOVWF   TXB0SIDL
    MOVLW   24H                     ; Load SID10:SID3
    MOVWF   TXB0SIDH
    ; No need to load TXB0EIDL:TXB0EIDH, as we are transmitting Standard Identifier Message only.

    ; Now that all data bytes are loaded, mark it for transmission.
    MOVLW   B’00001000’             ; Normal priority; Request transmission
    MOVWF   TXB0CON

    ; If required, wait for message to get transmitted
    BTFSC   TXB0CON, TXREQ          ; Is it transmitted?
    BRA     $-2                     ; No. Continue to wait...

    ; Message is transmitted.
```

42.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set of the PIC18(L)F25/26K83 family of devices. This includes the MPLAB C18 C compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default Configuration bits for that device. The default setting for the XINST Configuration bit is '0', disabling the extended instruction set and Indexed Literal Offset Addressing mode. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option, or dialog box within the environment, that allows the user to configure the language tool and its settings for the project
- A command line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

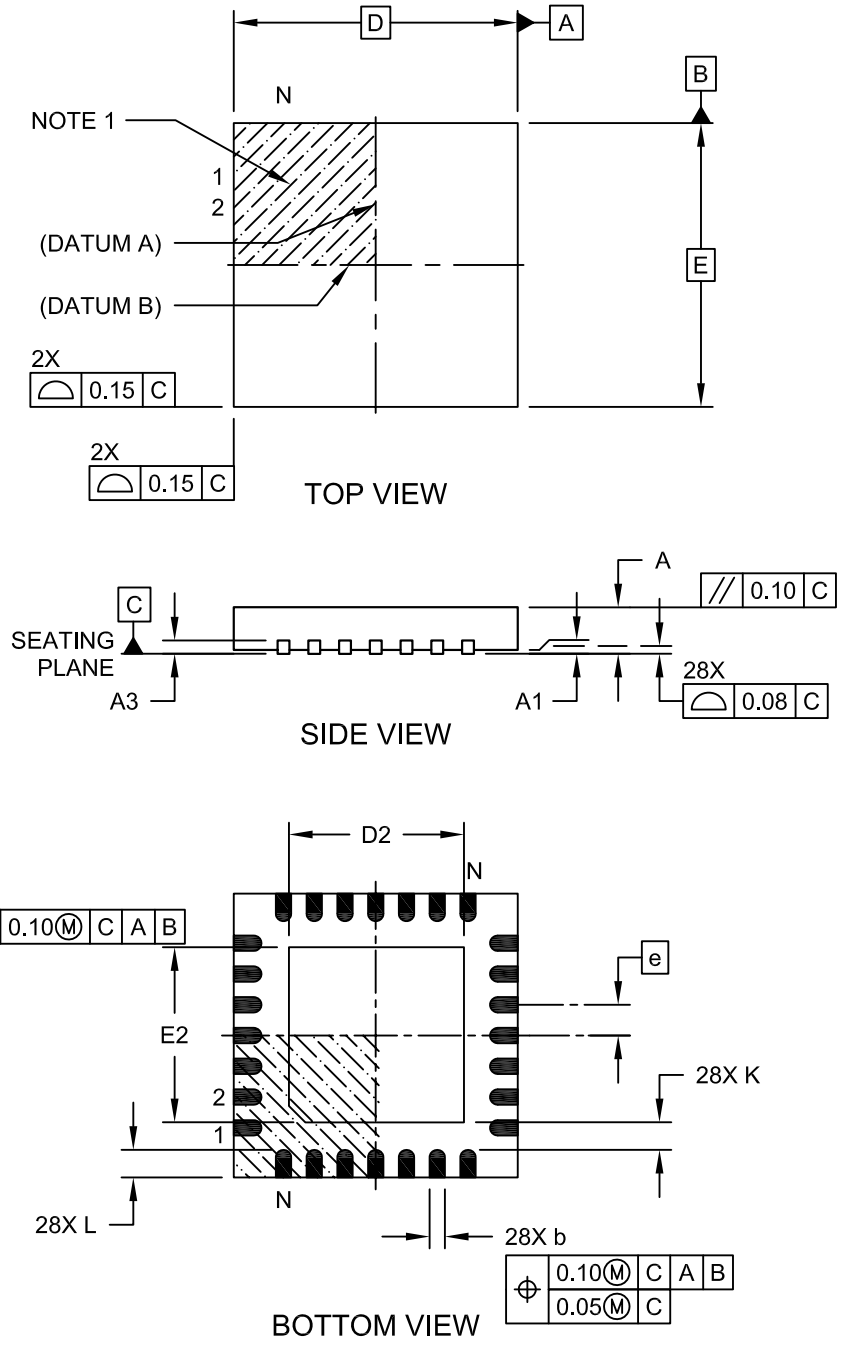
TABLE 43-1: REGISTER FILE SUMMARY FOR PIC18(L)F25/26K83 DEVICES (CONTINUED)

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3FCEh	PORTE	—	—	—	—	RE3	—	—	—	252
3FCDh	—	Unimplemented								—
3FCCh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	252
3FCBh	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	252
3FCAh	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	252
3FC9h - 3FC5h	—	Unimplemented								—
3FC4h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	253
3FC3h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	253
3FC2h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	253
3FC1h - 3FBDh	—	Unimplemented								—
3FBCh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	254
3FBBh	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	254
3FBAh	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	254
3FB9h	T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				287
3FB8h	T0CON0	EN	—	OUT	MD16	OUTPS				286
3FB7h	TMR0H	TMR0H								288
3FB6h	TMR0L	TMR0L								288
3FB5h	T1CLK	CS								300
3FB4h	T1GATE	GSS								301
3FB3h	T1GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3FB2h	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	323
3FB1h	TMR1H	TMR1H								302
3FB0h	TMR1L	TMR1L								302
3FAFh	T2RST	—	—	—	RSEL					321
3FAEh	T2CLK	—	—	—	—	CS				300
3FADh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE					324
3FACh	T2CON	ON	CKPS			OUTPS				298
3FABh	T2PR	PR2								322
3FAAh	T2TMR	TMR2								322
3FA9h	T3CLK	CS								300
3FA8h	T3GATE	GSS								301
3FA7h	T3GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3FA6h	T3CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	323
3FA5h	TMR3H	TMR3H								302
3FA4h	TMR3L	TMR3L								302
3FA3h	T4RST	—	—	—	RSEL					321
3FA2h	T4CLK	—	—	—	—	CS				320
3FA1h	T4HLT	PSYNC	CKPOL	CKSYNC	MODE					324
3FA0h	T4CON	ON	CKPS			OUTPS				323
3F9Fh	T4PR	PR4								322
3F9Eh	T4TMR	TMR4								322
3F9Dh	T5CLK	CS								320
3F9Ch	T5GATE	GSS								301
3F9Bh	T5GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3F9Ah	T5CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	323
3F99h	TMR5H	TMR5H								302
3F98h	TMR5L	TMR5L								302
3F97h	T6RST	—	—	—	RSEL					321

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not present in LF devices.

28-Lead Plastic Quad Flat, No Lead Package (ML) - 6x6 mm Body [QFN]
With 0.55 mm Terminal Length

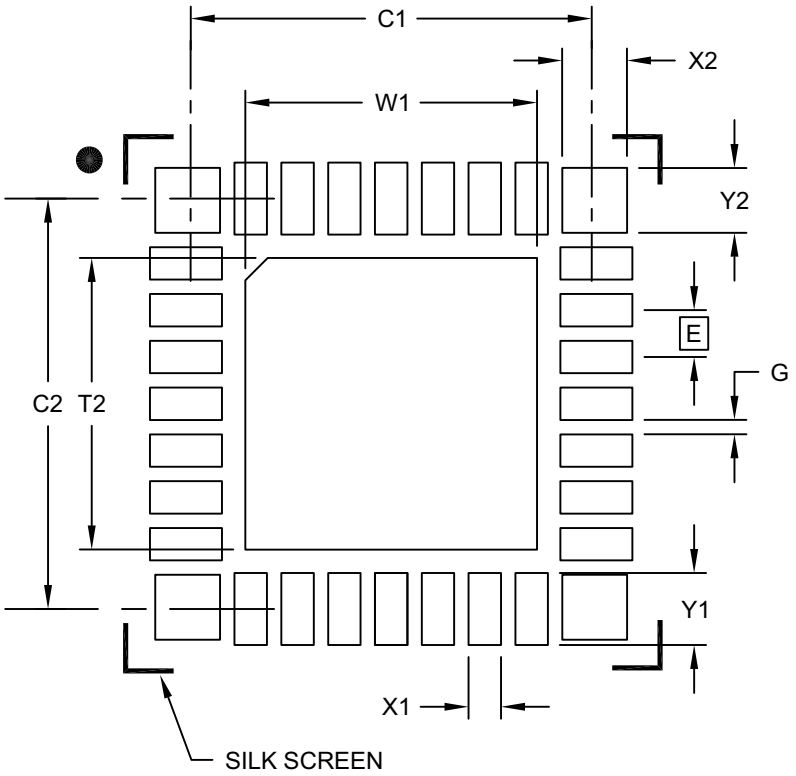
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-105C Sheet 1 of 2

28-Lead Plastic Quad Flat, No Lead Package (MX) - 6x6 mm Body [UQFN] With 0.60mm Contact Length And Corner Anchors

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W1			4.05
Optional Center Pad Length	T2			4.05
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.00
Corner Pad Width (X4)	X2			0.90
Corner Pad Length (X4)	Y2			0.90
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

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

Microchip Technology Drawing No. C04-2209B