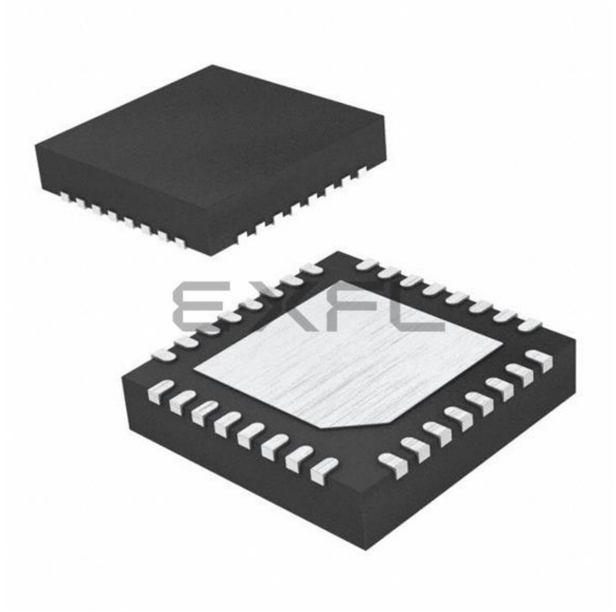
Microchip Technology - PIC18F25K80-E/MM Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

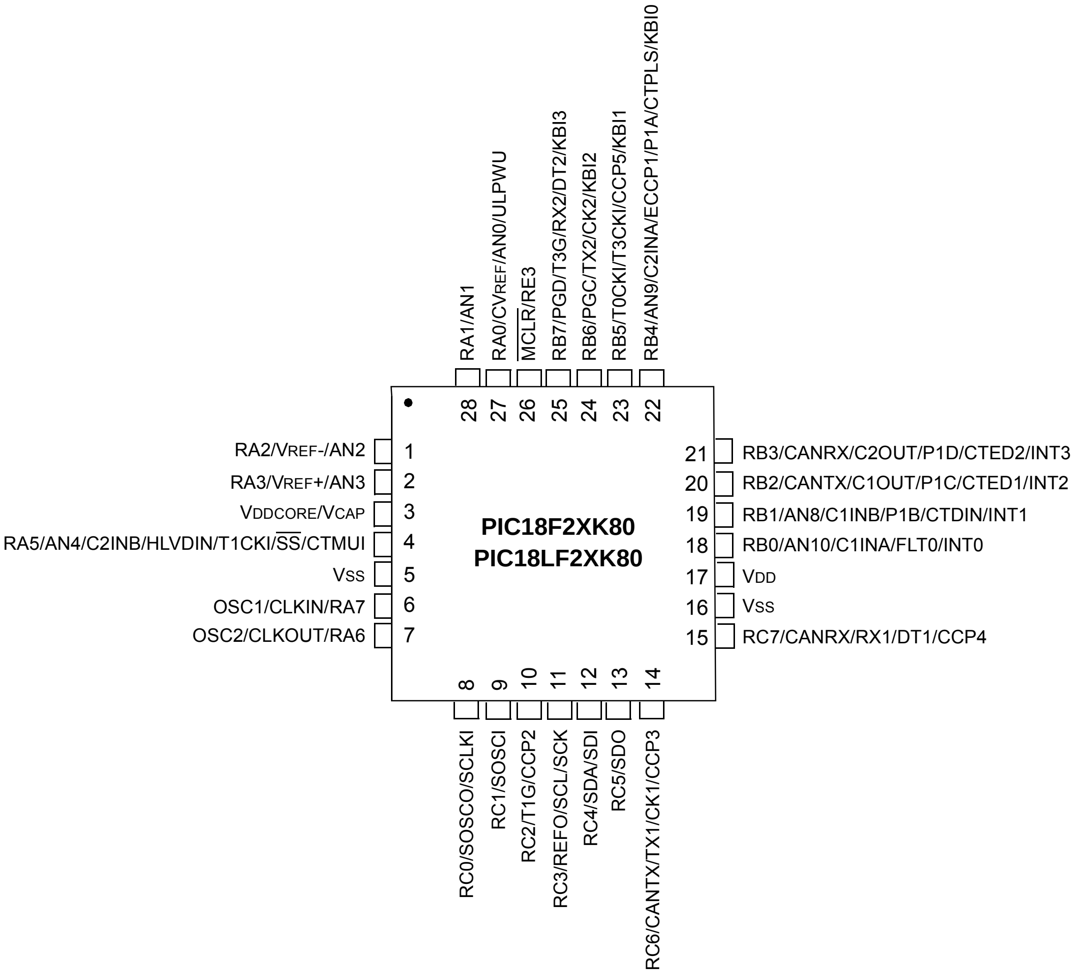
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	ECANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN-S (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25k80-e-mm

Pin Diagrams

28-Pin QFN(1)



Note 1: For the QFN package, it is recommended that the bottom pad be connected to VSS.

TABLE 1-6: PIC18F6XK80 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Num	Pin Type	Buffer Type	Description
RD6/P1C/PSP6	4			
RD6		I/O	ST/ CMOS	Digital I/O.
P1C		O	CMOS	Enhanced PWM1 Output C.
PSP6		I/O	ST/ CMOS	Parallel Slave Port data.
RD7/P1D/PSP7	5			
RD7		I/O	ST/ CMOS	Digital I/O.
P1D		O	CMOS	Enhanced PWM1 Output D.
PSP7		I/O	ST/ CMOS	Parallel Slave Port data.

Legend:				
	I^2C™ =	I^2C/SMBus input buffer	CMOS	= CMOS compatible input or output
	ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
	I	= Input	O	= Output
	P	= Power		

REGISTER 4-1: PMD2: PERIPHERAL MODULE DISABLE REGISTER 2

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	MODMD[1]	ECANMD	CMP2MD	CMP1MD
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 7-4 **Unimplemented:** Read as ‘0’

bit 3 **MODMD:** Modulator Output Module Disable bit[1]

1 = The modulator output module is disabled; all Modulator Output registers are held in Reset and are not writable

0 = The modulator output module is enabled

bit 2 **ECANMD:** Enhanced CAN Module Disable bit

1 = The Enhanced CAN module is disabled; all Enhanced CAN registers are held in Reset and are not writable

0 = The Enhanced CAN module is enabled

bit 1 **CMP2MD:** Comparator 2 Module Disable bit

1 = The Comparator 2 module is disabled; all Comparator 2 registers are held in Reset and are not writable

0 = The Comparator 2 module is enabled

bit 0 **CMP1MD:** Comparator 1 Module Disable bit

1 = The Comparator 1 module is disabled; all Comparator 1 registers are held in Reset and are not writable

0 = The Comparator 1 module is enabled

Note 1: This bit is only implemented on devices with 64 pins (PIC18F6XK80, PIC18LF6XK80).

TABLE 6-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F66K80 FAMILY (CONTINUED)

Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name
F3Fh	CANCON_RO0[5]	F0Fh	CANCON_RO3[5]	EDFh	CANCON_RO4[5]	EAFh	CANCON_RO7[5]	E7Fh	TXBIE[5]	E4Fh	RXF7EIDL[5]
F3Eh	CANSTAT_RO0[5]	F0Eh	CANSTAT_RO3[5]	EDEh	CANSTAT_RO4[5]	EAEh	CANSTAT_RO7[5]	E7Eh	BIE0[5]	E4Eh	RXF7EIDH[5]
F3Dh	RXB1D7[5]	F0Dh	TXB2D7[5]	EDDh	B5D7[5]	EADh	B2D7[5]	E7Dh	BSEL0[5]	E4Dh	RXF7SIDL[5]
F3Ch	RXB1D6[5]	F0Ch	TXB2D6[5]	EDCh	B5D6[5]	EACh	B2D6[5]	E7Ch	MSEL3[5]	E4Ch	RXF7SIDH[5]
F3Bh	RXB1D5[5]	F0Bh	TXB2D5[5]	EDBh	B5D5[5]	EABh	B2D5[5]	E7Bh	MSEL2[5]	E4Bh	RXF6EIDL[5]
F3Ah	RXB1D4[5]	F0Ah	TXB2D4[5]	EDAh	B5D4[5]	EAAh	B2D4[5]	E7Ah	MSEL1[5]	E4Ah	RXF6EIDH[5]
F39h	RXB1D3[5]	F09h	TXB2D3[5]	ED9h	B5D3[5]	EA9h	B2D3[5]	E79h	MSEL0[5]	E49h	RXF6SIDL[5]
F38h	RXB1D2[5]	F08h	TXB2D2[5]	ED8h	B5D2[5]	EA8h	B2D2[5]	E78h	RXFBCON7[5]	E48h	RXF6SIDH[5]
F37h	RXB1D1[5]	F07h	TXB2D1[5]	ED7h	B5D1[5]	EA7h	B2D1[5]	E77h	RXFBCON6[5]	E47h	RXFCON1[5]
F36h	RXB1D0[5]	F06h	TXB2D0[5]	ED6h	B5D0[5]	EA6h	B2D0[5]	E76h	RXFBCON5[5]	E46h	RXFCON0[5]
F35h	RXB1DLC[5]	F05h	TXB2DLC[5]	ED5h	B5DLC[5]	EA5h	B2DLC[5]	E75h	RXFBCON4[5]	E45h	BRGCON3[5]
F34h	RXB1EIDL[5]	F04h	TXB2EIDL[5]	ED4h	B5EIDL[5]	EA4h	B2EIDL[5]	E74h	RXFBCON3[5]	E44h	BRGCON2[5]
F33h	RXB1EIDH[5]	F03h	TXB2EIDH[5]	ED3h	B5EIDH[5]	EA3h	B2EIDH[5]	E73h	RXFBCON2[5]	E43h	BRGCON1[5]
F32h	RXB1SIDL[5]	F02h	TXB2SIDL[5]	ED2h	B5SIDL[5]	EA2h	B2SIDL[5]	E72h	RXFBCON1[5]	E42h	TXERRCNT[5]
F31h	RXB1SIDH[5]	F01h	TXB2SIDH[5]	ED1h	B5SIDH[5]	EA1h	B2SIDH[5]	E71h	RXFBCON0[5]	E41h	RXERRCNT[5]
F30h	RXB1CON[5]	F00h	TXB2CON[5]	ED0h	B5CON[5]	EA0h	B2CON[5]	E70h	SDFLC[5]		
F30h	RXB1CON[5]	EFFh	RXM1EIDL[5]	ECFh	CANCON_RO5[5]	E9Fh	CANCON_RO8[5]	E6Fh	RXF15EIDL[5]		
F2Fh	CANCON_RO1[5]	EFEh	RXM1EIDH[5]	ECEh	CANSTAT_RO5[5]	E9Eh	CANSTAT_RO8[5]	E6Eh	RXF15EIDH[5]		
F2Eh	CANSTAT_RO1[5]	EFDh	RXM1SIDL[5]	ECDh	B4D7[5]	E9Dh	B1D7[5]	E6Dh	RXF15SIDL[5]		
F2Dh	TXB0D7[5]	EFCh	RXM1SIDH[5]	ECCh	B4D6[5]	E9Ch	B1D6[5]	E6Ch	RXF15SIDH[5]		
F2Ch	TXB0D6[5]	EFBh	RXM0EIDL[5]	ECBh	B4D5[5]	E9Bh	B1D5[5]	E6Bh	RXF14EIDL[5]		
F2Bh	TXB0D5[5]	EFAh	RXM0EIDH[5]	ECAh	B4D4[5]	E9Ah	B1D4[5]	E6Ah	RXF14EIDH[5]		
F2Ah	TXB0D4[5]	EF9h	RXM0SIDL[5]	EC9h	B4D3[5]	E99h	B1D3[5]	E69h	RXF14SIDL[5]		
F29h	TXB0D3[5]	EF8h	RXM0SIDH[5]	EC8h	B4D2[5]	E98h	B1D2[5]	E68h	RXF14SIDH[5]		
F28h	TXB0D2[5]	EF7h	RXF5EIDL[5]	EC7h	B4D1[5]	E97h	B1D1[5]	E67h	RXF13EIDL[5]		
F27h	TXB0D1[5]	EF6h	RXF5EIDH[5]	EC6h	B4D0[5]	E96h	B1D0[5]	E66h	RXF13EIDH[5]		
F26h	TXB0D0[5]	EF5h	RXF5SIDL[5]	EC5h	B4DLC[5]	E95h	B1DLC[5]	E65h	RXF13SIDL[5]		
F25h	TXB0DLC[5]	EF4h	RXF5SIDH[5]	EC4h	B4EIDL[5]	E94h	B1EIDL[5]	E64h	RXF13SIDH[5]		
F24h	TXB0EIDL[5]	EF3h	RXF4EIDL[5]	EC3h	B4EIDH[5]	E93h	B1EIDH[5]	E63h	RXF12EIDL[5]		
F23h	TXB0EIDH[5]	EF2h	RXF4EIDH[5]	EC2h	B4SIDL[5]	E92h	B1SIDL[5]	E62h	RXF12EIDH[5]		
F22h	TXB0SIDL[5]	EF1h	RXF4SIDL[5]	EC1h	B4SIDH[5]	E91h	B1SIDH[5]	E61h	RXF12SIDL[5]		
F21h	TXB0SIDH[5]	EF0h	RXF4SIDH[5]	EC0h	B4CON[5]	E90h	B1CON[5]	E60h	RXF12SIDH[5]		
F20h	TXB0CON[5]	EEFh	RXF3EIDL[5]	EBFh	CANCON_RO6[5]	E90h	B1CON[5]	E5Fh	RXF11EIDL[5]		
F1Fh	CANCON_RO2[5]	EEEh	RXF3EIDH[5]	EBEh	CANSTAT_RO6[5]	E8Fh	CANCON_RO9[5]	E5Eh	RXF11EIDH[5]		
F1Eh	CANSTAT_RO2[5]	EEDh	RXF3SIDL[5]	EBDh	B3D7[5]	E8Eh	CANSTAT_RO9[5]	E5Dh	RXF11SIDL[5]		
F1Dh	TXB1D7[5]	EECh	RXF3SIDH[5]	EBCh	B3D6[5]	E8Dh	B0D7[5]	E5Ch	RXF11SIDH[5]		
F1Ch	TXB1D6[5]	EEBh	RXF2EIDL[5]	EBBh	B3D5[5]	E8Ch	B0D6[5]	E5Bh	RXF10EIDL[5]		
F1Bh	TXB1D5[5]	EEAh	RXF2EIDH[5]	EBAh	B3D4[5]	E8Bh	B0D5[5]	E5Ah	RXF10EIDH[5]		
F1Ah	TXB1D4[5]	EE9h	RXF2SIDL[5]	EB9h	B3D3[5]	E8Ah	B0D4[5]	E59h	RXF10SIDL[5]		
F19h	TXB1D3[5]	EE8h	RXF2SIDH[5]	EB8h	B3D2[5]	E89h	B0D3[5]	E58h	RXF10SIDH[5]		
F18h	TXB1D2[5]	EE7h	RXF1EIDL[5]	EB7h	B3D1[5]	E88h	B0D2[5]	E57h	RXF9EIDL[5]		
F17h	TXB1D1[5]	EE6h	RXF1EIDH[5]	EB6h	B3D0[5]	E87h	B0D1[5]	E56h	RXF9EIDH[5]		
F16h	TXB1D0[5]	EE5h	RXF1SIDL[5]	EB5h	B3DLC[5]	E86h	B0D0[5]	E55h	RXF9SIDL[5]		
F15h	TXB1DLC[5]	EE4h	RXF1SIDH[5]	EB4h	B3EIDL[5]	E85h	B0DLC[5]	E54h	RXF9SIDH[5]		
F14h	TXB1EIDL[5]	EE3h	RXF0EIDL[5]	EB3h	B3EIDH[5]	E84h	B0EIDL[5]	E53h	RXF8EIDL[5]		
F13h	TXB1EIDH[5]	EE2h	RXF0EIDH[5]	EB2h	B3SIDL[5]	E83h	B0EIDH[5]	E52h	RXF8EIDH[5]		
F12h	TXB1SIDL[5]	EE1h	RXF0SIDL[5]	EB1h	B3SIDH[5]	E82h	B0SIDL[5]	E51h	RXF8SIDL[5]		
F11h	TXB1SIDH[5]	EE0h	RXF0SIDH[5]	EB0h	B3CON[5]	E81h	B0SIDH[5]	E50h	RXF8SIDH[5]		
F10h	TXB1CON[5]					E80h	B0CON[5]				

Note
1: This is not a physical register.
2: Unimplemented registers are read as ‘0’.
3: This register is only available on devices with 64 pins.
4: This register is not available on devices with 28 pins.
5: Addresses, E41h through F5Fh, are also used by the SFRs, but are not part of the Access RAM. To access these registers, users must always load the proper BSR value.

8.6 Operation During Code-Protect

Data EEPROM memory has its own code-protect bits in Configuration Words. External read and write operations are disabled if code protection is enabled.

The microcontroller itself can both read and write to the internal data EEPROM regardless of the state of the code-protect Configuration bit. Refer to **Section 28.0 "Special Features of the CPU"** for additional information.

8.7 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been implemented. On power-up, the WREN bit is cleared. In addition, writes to the EEPROM are blocked during the Power-up Timer period (TPWRT, Parameter 33).

The write initiate sequence, and the WREN bit together, help prevent an accidental write during brown-out, power glitch or software malfunction.

8.8 Using the Data EEPROM

The data EEPROM is a high-endurance, byte-addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than Parameter D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 8-3.

> **Note:** If data EEPROM is only used to store constants and/or data that changes often, an array refresh is likely not required. See Parameter D124.

EXAMPLE 8-3: DATA EEPROM REFRESH ROUTINE

```
        CLRF    EEADR               ; Start at address 0
        CLRF    EEADRH              ;
        BCF     EECON1, CFGS        ; Set for memory
        BCF     EECON1, EEPGD       ; Set for Data EEPROM
        BCF     INTCON, GIE         ; Disable interrupts
        BSF     EECON1, WREN        ; Enable writes
LOOP                                ; Loop to refresh array
        BSF     EECON1, RD          ; Read current address
        MOVLW   55h                 ;
        MOVWF   EECON2              ; Write 55h
        MOVLW   0AAh                ;
        MOVWF   EECON2              ; Write 0AAh
        BSF     EECON1, WR          ; Set WR bit to begin write
        BTFSC   EECON1, WR          ; Wait for write to complete
        BRA     $-2
        INCFSZ  EEADR, F            ; Increment address
        BRA     LOOP                ; Not zero, do it again
        INCFSZ  EEADRH, F           ; Increment the high address
        BRA     LOOP                ; Not zero, do it again

        BCF     EECON1, WREN        ; Disable writes
        BSF     INTCON, GIE         ; Enable interrupts
```

10.0 INTERRUPTS

Members of the PIC18F66K80 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

The registers for controlling interrupt operation are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3, PIR4 and PIR5
- PIE1, PIE2, PIE3, PIE4 and PIE5
- IPR1, IPR2, IPR3, IPR4 and IPR5

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, interrupt sources have three bits to control their operation. They are:

- **Flag bit** – Indicating that an interrupt event occurred
- **Enable bit** – Enabling program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** – Specifying high priority or low priority

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits that enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate Global Interrupt Enable bit are set, the interrupt will vector immediately to address 0008h or 0018h, depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit that enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit that enables/disables all interrupt sources. All interrupts branch to address 0008h in Compatibility mode.

When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High-priority interrupt sources can interrupt a low-priority interrupt. Low-priority interrupts are not processed while high-priority interrupts are in progress.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine (ISR), the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used) that re-enables interrupts.

For external interrupt events, such as the INTx pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set regardless of the status of their corresponding enable bit or the GIE bit.

Note:	Do not use the `MOVFF` instruction to modify any of the Interrupt Control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

14.8.4 TIMER1 GATE SINGLE PULSE MODE

When Timer1 Gate Single Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer1 Gate Single Pulse mode is enabled by setting the T1GSPM bit (T1GCON<4>) and the T1GGO/$\overline{\text{T1DONE}}$ bit (T1GCON<3>). The Timer1 will be fully enabled on the next incrementing edge.

On the next trailing edge of the pulse, the T1GGO/$\overline{\text{T1DONE}}$ bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/$\overline{\text{T1DONE}}$ bit is once again set in software.

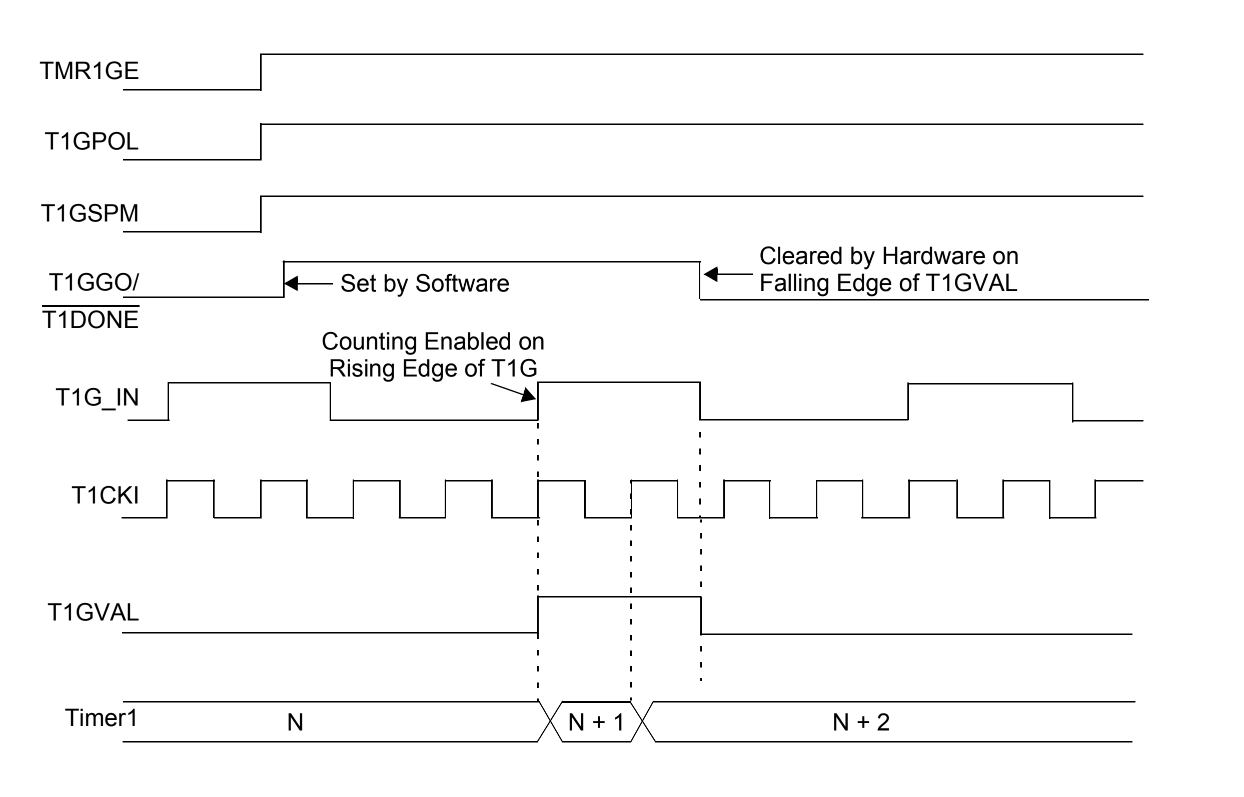
Clearing the T1GSPM bit of the T1GCON register will also clear the T1GGO/$\overline{\text{T1DONE}}$ bit. (For timing details, see Figure 14-6.)

Simultaneously enabling the Toggle and Single Pulse modes will permit both sections to work together. This allows the cycle times on the Timer1 gate source to be measured. (For timing details, see Figure 14-7.)

14.8.5 TIMER1 GATE VALUE STATUS

When the Timer1 gate value status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the T1GVAL bit (T1GCON<2>). This bit is valid even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

FIGURE 14-6: TIMER1 GATE SINGLE PULSE MODE



16.5 Timer3 Gates

Timer3 can be configured to count freely or the count can be enabled and disabled using the Timer3 gate circuitry. This is also referred to as the Timer3 gate count enable.

The Timer3 gate can also be driven by multiple selectable sources.

16.5.1 TIMER3 GATE COUNT ENABLE

The Timer3 Gate Enable mode is enabled by setting the TMR3GE bit (TxGCON<7>). The polarity of the Timer3 Gate Enable mode is configured using the T3GPOL bit (T3GCON<6>).

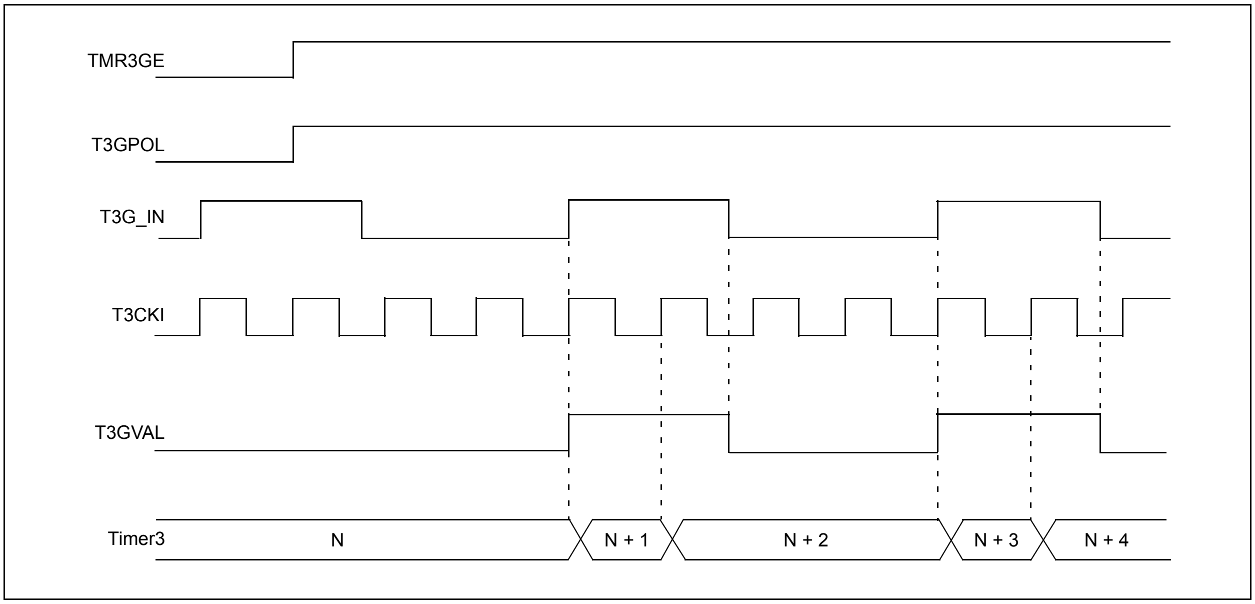
When Timer3 Gate Enable mode is enabled, Timer3 will increment on the rising edge of the Timer3 clock source. When Timer3 Gate Enable mode is disabled, no incrementing will occur and Timer3 will hold the current count. See Figure 16-2 for timing details.

TABLE 16-1: TIMER3 GATE ENABLE SELECTIONS

T3CLK[†]	T3GPOL (T3GCON<6>)	T3G Pin	Timer3 Operation
↑	0	0	Counts
↑	0	1	Holds Count
↑	1	0	Holds Count
↑	1	1	Counts

† The clock on which TMR3 is running. For more information, see T3CLK in Figure 16-1.

FIGURE 16-2: TIMER3 GATE COUNT ENABLE MODE



16.6 Timer3 Interrupt

The TMR3 register pair (TMR3H:TMR3L) increments from 0000h to FFFFh and overflows to 0000h. The Timer3 interrupt, if enabled, is generated on overflow and is latched in the interrupt flag bit, TMR3IF. Table 16-3 gives each module's flag bit.

This interrupt can be enabled or disabled by setting or clearing the TMR3IE bit. Table 16-3 displays each module's enable bit.

16.7 Resetting Timer3 Using the ECCP Special Event Trigger

If the ECCP modules are configured to use Timer3 and to generate a Special Event Trigger in Compare mode (CCP3M<3:0> = `1011`), this signal will reset Timer3. The trigger from ECCP will also start an A/D conversion if the A/D module is enabled (For more information, see **Section 20.3.4 "Special Event Trigger"**.)

The module must be configured as either a timer or synchronous counter to take advantage of this feature. When used this way, the CCPR3H:CCPR3L register pair effectively becomes a Period register for Timer3.

If Timer3 is running in Asynchronous Counter mode, the Reset operation may not work.

In the event that a write to Timer3 coincides with a Special Event Trigger from an ECCP module, the write will take precedence.

Note: The Special Event Triggers from the ECCPx module will only clear the TMR3 register's content, but not set the TMR3IF interrupt flag bit (PIR2<1>).

Note: The CCP and ECCP modules use Timers, 1 through 4, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRS register. For more details, see Register 20-2 and Register 19-2.

TABLE 16-3: REGISTERS ASSOCIATED WITH TIMER3 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR5	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF	TXB0IF	RXB1IF	RXB0IF
PIE5	IRXIE	WAKIE	ERRIE	TX2BIE	TXB1IE	TXB0IE	RXB1IE	RXB0IE
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
TMR3H	Timer3 Register High Byte							
TMR3L	Timer3 Register Low Byte							
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
OSCCON2	—	SOSCRUN	—	SOSCDRV	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

Legend: — = unimplemented, read as '`0`'. Shaded cells are not used by the Timer3 module.

21.3.1 REGISTERS

The MSSP module has four registers for SPI mode operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPSR and SSPBUF together, create a double-buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double-buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

REGISTER 21-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE$^{(1)}$	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF
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 **SMP:** Sample bit
SPI Master mode:
1 = Input data is sampled at the end of data output time
0 = Input data is sampled at the middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Select bit$^{(1)}$
1 = Transmit occurs on transition from active to Idle clock state
0 = Transmit occurs on transition from Idle to active clock state

bit 5 **D/$\overline{\text{A}}$:** Data/$\overline{\text{Address}}$ bit
Used in I^{2}C™ mode only.

bit 4 **P:** Stop bit
Used in I^{2}C mode only. This bit is cleared when the MSSP module is disabled; SSPEN is cleared.

bit 3 **S:** Start bit
Used in I^{2}C mode only.

bit 2 **R/$\overline{\text{W}}$:** Read/$\overline{\text{Write}}$ Information bit
Used in I^{2}C mode only.

bit 1 **UA:** Update Address bit
Used in I^{2}C mode only.

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
1 = Receive is complete, SSPBUF is full
0 = Receive is not complete, SSPBUF is empty

Note 1: Polarity of clock state is set by the CKP bit (SSPCON1<4>).

21.3.4 ENABLING SPI I/O

To enable the serial port, MSSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPCON registers and then set the SSPEN bit. This configures the SDI, SDO, SCK and $\overline{\text{SS}}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- SDI is automatically controlled by the SPI module
- SDO must have the TRISC<5> bit cleared
- SCK (Master mode) must have the TRISC<3> bit cleared
- SCK (Slave mode) must have the TRISC<3> bit set
- $\overline{\text{SS}}$ must have the TRISA<5> bit set

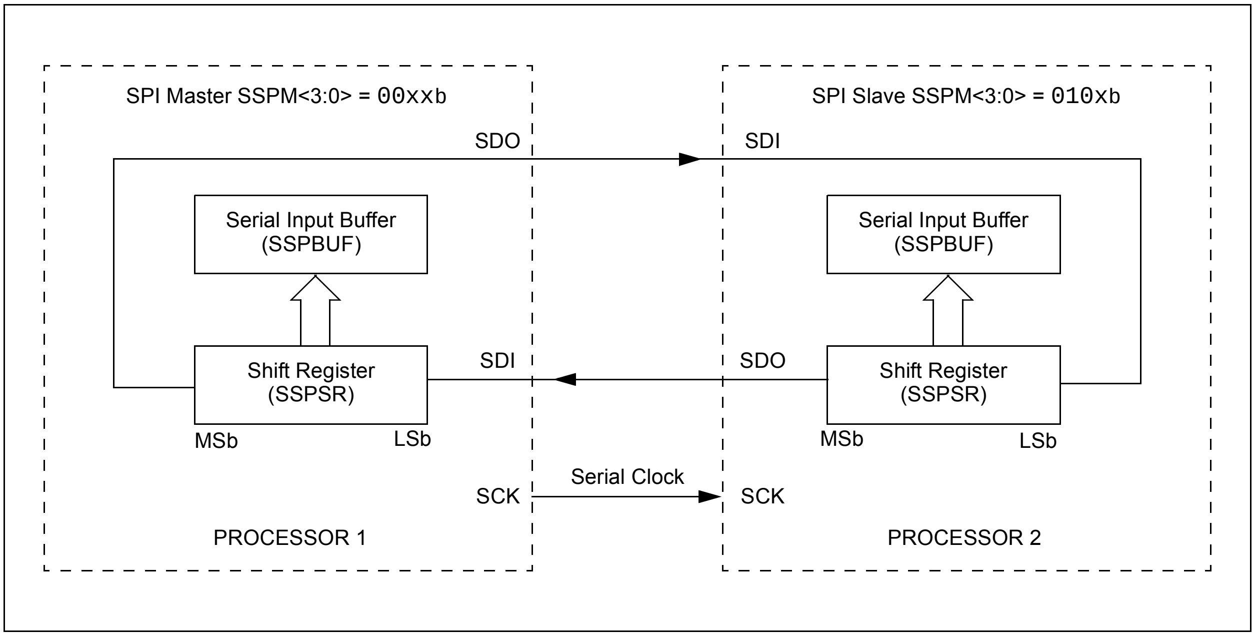
Any serial port function that is not desired may be overridden by programming the corresponding Data Direction (TRIS) register to the opposite value.

21.3.5 TYPICAL CONNECTION

Figure 21-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

FIGURE 21-2: SPI MASTER/SLAVE CONNECTION


SPI Master SSPM<3:0> = 00xxb
SPI Slave SSPM<3:0> = 010xb
SDO
SDI
Serial Input Buffer (SSPBUF)
Serial Input Buffer (SSPBUF)
Shift Register (SSPSR)
SDI
SDO
Shift Register (SSPSR)
MSb
LSb
MSb
LSb
Serial Clock
SCK
SCK
PROCESSOR 1
PROCESSOR 2

21.4.2 OPERATION

The MSSP module functions are enabled by setting the MSSP Enable bit, SSPEN (SSPCON1<5>).

The SSPCON1 register allows control of the I^{2}C operation. Four mode selection bits (SSPCON1<3:0>) allow one of the following I^{2}C modes to be selected:

- I^{2}C Master mode, clock
- I^{2}C Slave mode (7-bit address)
- I^{2}C Slave mode (10-bit address)
- I^{2}C Slave mode (7-bit address) with Start and Stop bit interrupts enabled
- I^{2}C Slave mode (10-bit address) with Start and Stop bit interrupts enabled
- I^{2}C Firmware Controlled Master mode, slave is Idle

Selection of any I^{2}C mode with the SSPEN bit set forces the SCL and SDA pins to be open-drain, provided these pins are programmed as inputs by setting the appropriate TRISC bit. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCL and SDA pins.

21.4.3 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISC<4:3> set). The MSSP module will override the input state with the output data when required (slave-transmitter).

The I^{2}C Slave mode hardware will always generate an interrupt on an address match. Address masking will allow the hardware to generate an interrupt for more than one address (up to 31 in 7-bit addressing and up to 63 in 10-bit addressing). Through the mode select bits, the user can also choose to interrupt on Start and Stop bits.

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{\text{ACK}}$) pulse and load the SSPBUF register with the received value currently in the SSPSR register.

Any combination of the following conditions will cause the MSSP module not to give this $\overline{\text{ACK}}$ pulse:

- The Buffer Full bit, BF (SSPSTAT<0>), was set before the transfer was received.
- The overflow bit, SSPOV (SSPCON1<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit, SSPIF, is set. The BF bit is cleared by reading the SSPBUF register, while bit, SSPOV, is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I^{2}C specification, as well as the requirement of the MSSP module, are shown in timing Parameter 100 and Parameter 101.

21.4.3.1 Addressing

Once the MSSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register, SSPSR<7:1>, is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

1. The SSPSR register value is loaded into the SSPBUF register.
2. The Buffer Full bit, BF, is set.
3. An $\overline{\text{ACK}}$ pulse is generated.
4. The MSSP Interrupt Flag bit, SSPIF, is set (and interrupt is generated, if enabled) on the falling edge of the ninth SCL pulse.

In 10-Bit Addressing mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. The R/$\overline{\text{W}}$ (SSPSTAT<2>) bit must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '`11110 A9 A8 0`', where '`A9`' and '`A8`' are the two MSbs of the address. The sequence of events for 10-bit addressing is as follows, with Steps 7 through 9 for the slave-transmitter:

1. Receive first (high) byte of address (bits, SSPIF, BF and UA, are set on address match).
2. Update the SSPADD register with second (low) byte of address (clears bit, UA, and releases the SCL line).
3. Read the SSPBUF register (clears bit, BF) and clear flag bit, SSPIF.
4. Receive second (low) byte of address (bits, SSPIF, BF and UA, are set).
5. Update the SSPADD register with the first (high) byte of address. If match releases SCL line, this will clear bit, UA.
6. Read the SSPBUF register (clears bit, BF) and clear flag bit SSPIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits, SSPIF and BF, are set).
9. Read the SSPBUF register (clears bit, BF) and clear flag bit, SSPIF.

FIGURE 21-16: I^{2}C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)

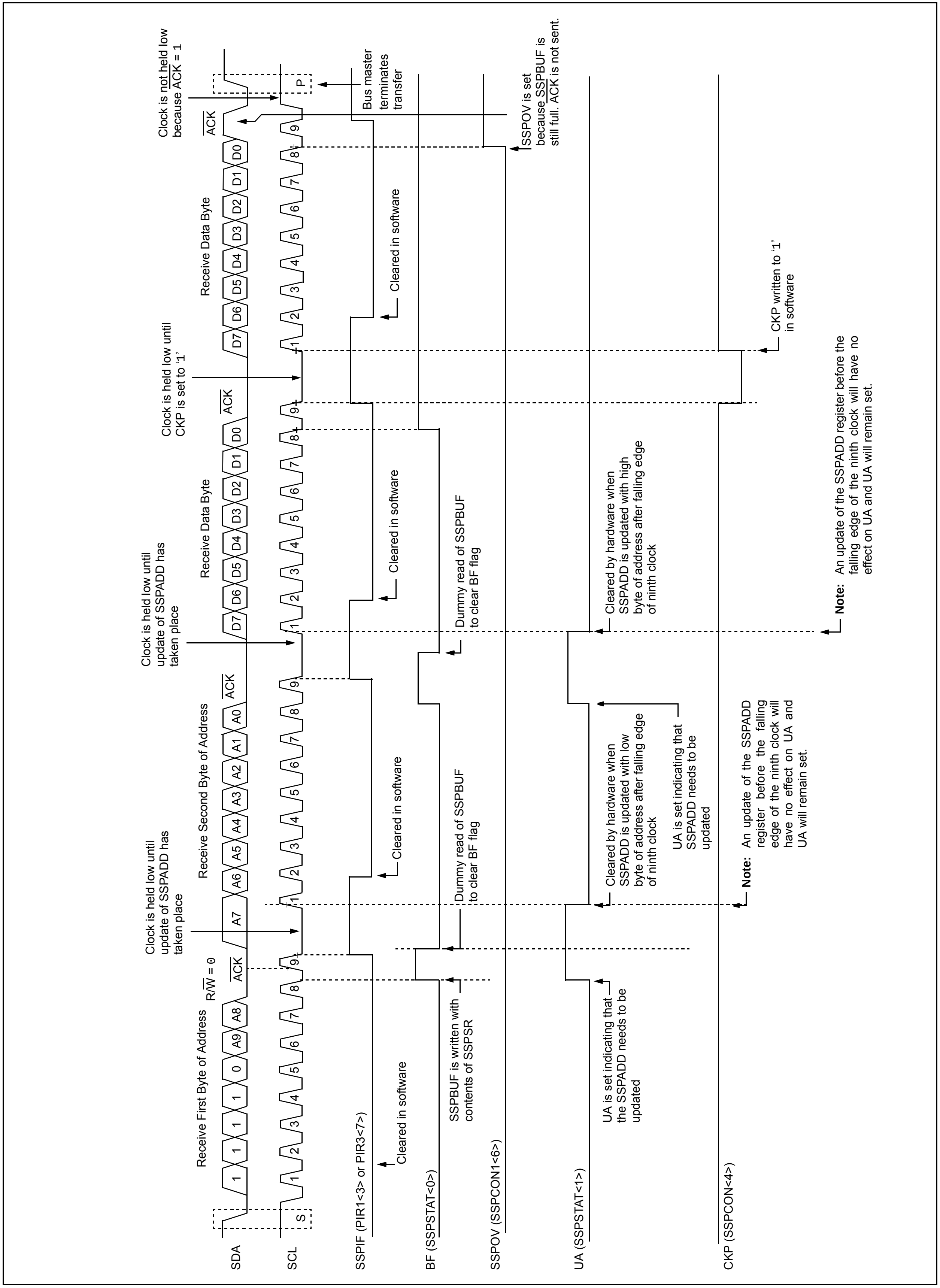

SDA
SCL
SSPIF (PIR1<3> or PIR3<7>)
BF (SSPSTAT<0>)
SSPOV (SSPCON1<6>)
UA (SSPSTAT<1>)
CKP (SSPCON<4>)
S
1 1 1 1 0 A9 A8 R/W = 0 ACK
1 2 3 4 5 6 7 8 9
Receive First Byte of Address
Cleared in software
SSPBUF is written with contents of SSPSR
UA is set indicating that the SSPADD needs to be updated
Clock is held low until update of SSPADD has taken place
A7 A6 A5 A4 A3 A2 A1 A0 ACK
1 2 3 4 5 6 7 8 9
Receive Second Byte of Address
Cleared in software
Dummy read of SSPBUF to clear BF flag
Cleared by hardware when SSPADD is updated with low byte of address after falling edge of ninth clock
UA is set indicating that SSPADD needs to be updated
Note: An update of the SSPADD register before the falling edge of the ninth clock will have no effect on UA and UA will remain set.
Clock is held low until update of SSPADD has taken place
D7 D6 D5 D4 D3 D2 D1 D0 ACK
1 2 3 4 5 6 7 8 9
Receive Data Byte
Cleared in software
Dummy read of SSPBUF to clear BF flag
Cleared by hardware when SSPADD is updated with high byte of address after falling edge of ninth clock
Note: An update of the SSPADD register before the falling edge of the ninth clock will have no effect on UA and UA will remain set.
Clock is held low until CKP is set to '1'
D7 D6 D5 D4 D3 D2 D1 D0 ACK
1 2 3 4 5 6 7 8 9
Receive Data Byte
Cleared in software
CKP written to '1' in software
SSPOV is set because SSPBUF is still full. ACK is not sent.
Clock is not held low because ACK = 1
Bus master terminates transfer
P

TABLE 22-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG1	EUSART1 Receive Register							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 Baud Rate Generator Register High Byte							
SPBRG1	EUSART1 Baud Rate Generator Register Low Byte							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG2	EUSART2 Receive Register							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 Baud Rate Generator Register High Byte							
SPBRG2	EUSART2 Baud Rate Generator Register Low Byte							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous master reception.

REGISTER 24-1: CMxCON: COMPARATOR CONTROL x REGISTER

R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
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 **CON:** Comparator Enable bit

1 = Comparator is enabled
0 = Comparator is disabled

bit 6 **COE:** Comparator Output Enable bit

1 = Comparator output is present on the CxOUT pin
0 = Comparator output is internal only

bit 5 **CPOL:** Comparator Output Polarity Select bit

1 = Comparator output is inverted
0 = Comparator output is not inverted

bit 4-3 **EVPOL<1:0>:** Interrupt Polarity Select bits

11 = Interrupt generation on any change of the output(1)
10 = Interrupt generation only on high-to-low transition of the output
01 = Interrupt generation only on low-to-high transition of the output
00 = Interrupt generation is disabled

bit 2 **CREF:** Comparator Reference Select bit (non-inverting input)

1 = Non-inverting input connects to internal CVREF voltage
0 = Non-inverting input connects to CxINA pin

bit 1-0 **CCH<1:0>:** Comparator Channel Select bits

11 = Inverting input of comparator connects to VBG
10 = Inverting input of comparator connects to C2INB pin(2)
01 = Inverting input of comparator connects to CxINC pin
00 = Inverting input of comparator connects to C1INB pin(2)

Note 1: The CMPxIF is automatically set any time this mode is selected and must be cleared by the application after the initial configuration.

2: Comparator 1 uses C2INB as an input to the inverting terminal. Comparator 2 uses C1INB as an input to the inverted terminal.

REGISTER 27-3: ECANCON: ENHANCED CAN CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
MDSEL1[(1)]	MDSEL0[(1)]	FIFOWM[(2)]	EWIN4	EWIN3	EWIN2	EWIN1	EWIN0
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 7-6 **MDSEL<1:0>:** Mode Select bits[(1)]

00 = Legacy mode (Mode 0, default)
01 = Enhanced Legacy mode (Mode 1)
10 = Enhanced FIFO mode (Mode 2)
11 = Reserved

bit 5 **FIFOWM:** FIFO High Water Mark bit[(2)]

1 = Will cause FIFO interrupt when one receive buffer remains
0 = Will cause FIFO interrupt when four receive buffers remain[(3)]

bit 4-0 **EWIN<4:0>:** Enhanced Window Address bits

These bits map the group of 16 banked CAN SFRs into Access Bank addresses, 0F60-0F6Dh. The exact group of registers to map is determined by the binary value of these bits.

Mode 0:
Unimplemented: Read as ‘0’

Mode 1, 2:
00000 = Acceptance Filters 0, 1, 2 and BRGCON2, 3
00001 = Acceptance Filters 3, 4, 5 and BRGCON1, CIOCON
00010 = Acceptance Filter Masks, Error and Interrupt Control
00011 = Transmit Buffer 0
00100 = Transmit Buffer 1
00101 = Transmit Buffer 2
00110 = Acceptance Filters 6, 7, 8
00111 = Acceptance Filters 9, 10, 11
01000 = Acceptance Filters 12, 13, 14
01001 = Acceptance Filter 15
01010-01110 = Reserved
01111 = RXINT0, RXINT1
10000 = Receive Buffer 0
10001 = Receive Buffer 1
10010 = TX/RX Buffer 0
10011 = TX/RX Buffer 1
10100 = TX/RX Buffer 2
10101 = TX/RX Buffer 3
10110 = TX/RX Buffer 4
10111 = TX/RX Buffer 5
11000-11111 = Reserved

Note 1: These bits can only be changed in Configuration mode. See Register 27-1 to change to Configuration mode.
2: This bit is used in Mode 2 only.
3: If FIFO is configured to contain four or less buffers, then the FIFO interrupt will trigger.

REGISTER 28-11: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0
bit 7							bit 0

Legend:	C = Clearable bit		
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 7-4 **Unimplemented:** Read as ‘0’

bit 3 **EBTR3:** Table Read Protection bit

1 = Block 3 is not protected from table reads executed in other blocks(1)
0 = Block 3 is protected from table reads executed in other blocks(1)

bit 2 **EBTR2**: Table Read Protection bit

1 = Block 2 is not protected from table reads executed in other blocks(1)
0 = Block 2 is protected from table reads executed in other blocks(1)

bit 1 **EBTR1:** Table Read Protection bit

1 = Block 1 is not protected from table reads executed in other blocks(1)
0 = Block 1 is protected from table reads executed in other blocks(1)

bit 0 **EBTR0:** Table Read Protection bit

1 = Block 0 is not protected from table reads executed in other blocks(1)
0 = Block 0 is protected from table reads executed in other blocks(1)

Note 1: For the memory size of the blocks, see Figure 28-6.

REGISTER 28-12: CONFIG7H: CONFIGURATION REGISTER 7 HIGH (BYTE ADDRESS 30000Dh)

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
bit 7							bit 0

Legend:	C = Clearable bit		
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 **Unimplemented:** Read as ‘0’

bit 6 **EBTRB:** Boot Block Table Read Protection bit

1 = Boot block is not protected from table reads executed in other blocks[(1)]
0 = Boot block is protected from table reads executed in other blocks[(1)]

bit 5-0 **Unimplemented:** Read as ‘0’

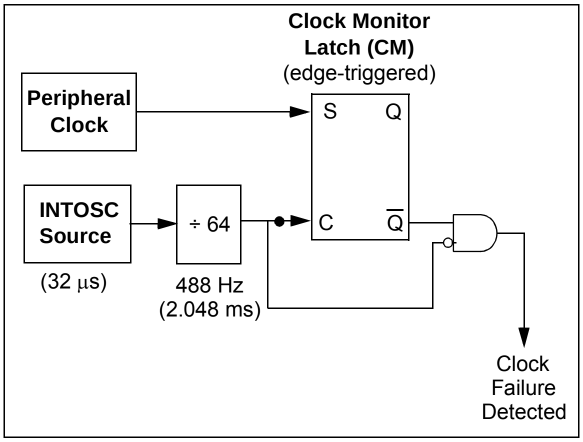
Note 1: For the memory size of the blocks, see Figure 28-6.

28.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the LF-INTOSC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 28-4) is accomplished by creating a sample clock signal, which is the output from the LF-INTOSC divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor (CM) latch. The CM is set on the falling edge of the device clock source, but cleared on the rising edge of the sample clock.

FIGURE 28-4: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 28-5). This causes the following:

- The FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>)
- The device clock source switches to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition)
- The WDT is reset

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing-sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 4.1.4 "Multiple Sleep Commands"** and **Section 28.4.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF<2:0> bits prior to entering Sleep mode.

The FSCM will detect only failures of the primary or secondary clock sources. If the internal oscillator block fails, no failure would be detected nor would any action be possible.

28.5.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTOSC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTOSC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF<2:0> bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

28.5.2 EXITING FAIL-SAFE OPERATION

The Fail-Safe condition is terminated by either a device Reset or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as the OST or PLL timer). The INTOSC multiplexer provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

TBLWT Table Write

Syntax: TBLWT (*; *+; *-; +*)

Operands: None

Operation:
if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR – No Change
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) + 1 → TBLPTR
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) – 1 → TBLPTR
if TBLWT+*,
(TBLPTR) + 1 → TBLPTR;
(TABLAT) → Holding Register

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 6.0 “Memory Organization”** for additional details on programming Flash memory.)

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0:	Least Significant Byte of Program Memory Word
TBLPTR[0] = 1:	Most Significant Byte of Program Memory Word

The `TBLWT` instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

TBLWT Table Write (Continued)

Example 1: `TBLWT *+;`

Before Instruction

TABLAT	=	55h
TBLPTR	=	00A356h
HOLDING REGISTER (00A356h)	=	FFh

After Instructions (table write completion)

TABLAT	=	55h
TBLPTR	=	00A357h
HOLDING REGISTER (00A356h)	=	55h

Example 2: `TBLWT +*;`

Before Instruction

TABLAT	=	34h
TBLPTR	=	01389Ah
HOLDING REGISTER (01389Ah)	=	FFh
HOLDING REGISTER (01389Bh)	=	FFh

After Instruction (table write completion)

TABLAT	=	34h
TBLPTR	=	01389Bh
HOLDING REGISTER (01389Ah)	=	FFh
HOLDING REGISTER (01389Bh)	=	34h