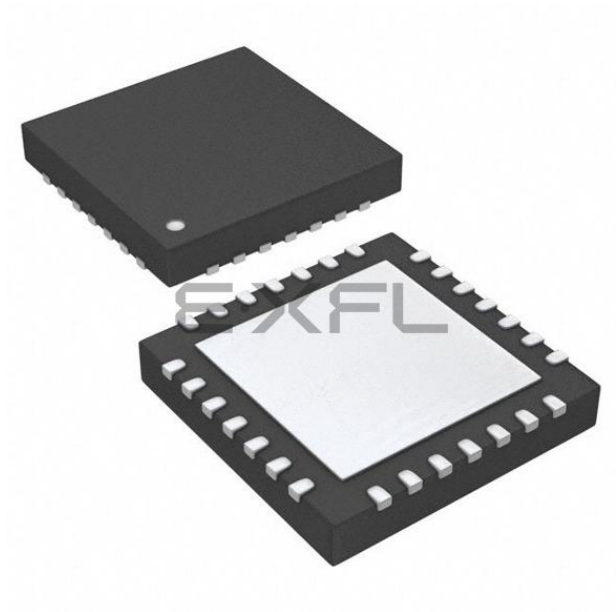


Welcome to E-XFL.COM

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	48MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf26j11t-i-ml

TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
$\overline{MCLR}$	18	18	I	ST	Master Clear (Reset) input; this is an active-low Reset to the device.
OSC1/CLKI/RA7	32	30			Oscillator crystal or external clock input.
OSC1			I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS. Main oscillator input connection.
CLKI			I	CMOS	External clock source input; always associated with pin function OSC1 (see related OSC1/CLKI pins).
RA7(1)			I/O	TTL	Digital I/O.
OSC2/CLKO/RA6	33	31			Oscillator crystal or clock output
OSC2			O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	—	Main oscillator feedback output connection in RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6(1)			I/O	TTL	Digital I/O.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); DIG = Digital output

Note 1: RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

4.6.9 DEEP SLEEP MODE REGISTERS

Deep Sleep mode registers are provided in Register 4-1 through Register 4-6.

REGISTER 4-1: DSCONH: DEEP SLEEP CONTROL HIGH BYTE REGISTER (BANKED F4Dh)

R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
DSEN(1)	—	—	—	—	(Reserved)	DSULPEN	RTCWDIS
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 **DSEN:** Deep Sleep Enable bit(1)
1 = Deep Sleep mode is entered on a SLEEP command
0 = Sleep mode is entered on a SLEEP command

bit 6-3 **Unimplemented:** Read as '0'

bit 2 **(Reserved):** Always write '0' to this bit

bit 1 **DSULPEN:** Ultra Low-Power Wake-up Module Enable bit
1 = ULPWU module is enabled in Deep Sleep
0 = ULPWU module is disabled in Deep Sleep

bit 0 **RTCWDIS:** RTCC Wake-up Disable bit
1 = Wake-up from RTCC is disabled
0 = Wake-up from RTCC is enabled

Note 1: In order to enter Deep Sleep, Sleep must be executed immediately after setting DSEN.

REGISTER 4-2: DSCONL: DEEP SLEEP CONTROL LOW BYTE REGISTER (BANKED F4Ch)

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0(1)	R/W-0(1)
—	—	—	—	—	ULPWDIS	DSBOR	RELEASE
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 7-3 **Unimplemented:** Read as '0'

bit 2 **ULPWDIS:** Ultra Low-Power Wake-up Disable bit
1 = ULPWU wake-up source is disabled
0 = ULPWU wake-up source is enabled (must also set DSULPEN = 1)

bit 1 **DSBOR:** Deep Sleep BOR Event Status bit
1 = DSBOREN was enabled and V_{DD} dropped below the DSBOR arming voltage during Deep Sleep, but did not fall below V_{DSBOR}
0 = DSBOREN was disabled or V_{DD} did not drop below the DSBOR arming voltage during Deep Sleep

bit 0 **RELEASE:** I/O Pin State Release bit
Upon waking from Deep Sleep, the I/O pins maintain their previous states. Clearing this bit will release the I/O pins and allow their respective TRIS and LAT bits to control their states.

Note 1: This is the value when V_{DD} is initially applied.

6.1.4.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 6-1) contains the Stack Pointer value, the STKFUL (Stack Full) and the STKUNF (Stack Underflow) status bits. The value of the Stack Pointer can be 0 through 31. The Stack Pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. On Reset, the Stack Pointer value will be zero. The user may read and write the Stack Pointer value. This feature can be used by a Real-Time Operating System (RTOS) for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a Power-on Reset (POR).

The action that takes place when the stack becomes full depends on the state of the Stack Overflow Reset Enable (STVREN) Configuration bit.

Refer to **Section 26.1 "Configuration Bits"** for device Configuration bits' description.

If STVREN is set (default), the 31st push will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31st push and the STKPTR will remain at 31.

When the stack has been popped enough times to unload the stack, the next pop will return zero to the PC and set the STKUNF bit, while the Stack Pointer remains at zero. The STKUNF bit will remain set until cleared by software or until a POR occurs.

Note:	Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset, as the contents of the SFRs are not affected.

6.1.4.3 PUSH and POP Instructions

Since the Top-of-Stack is readable and writable, the ability to push values onto the stack and pull values off the stack, without disturbing normal program execution is necessary. The PIC18 instruction set includes two instructions, PUSH and POP, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

The PUSH instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack.

The POP instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

REGISTER 6-1: STKPTR: STACK POINTER REGISTER (ACCESS FFCh)

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL(1)	STKUNF(1)	—	SP4	SP3	SP2	SP1	SP0
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 **STKFUL:** Stack Full Flag bit(1)
1 = Stack became full or overflowed
0 = Stack has not become full or overflowed

bit 6 **STKUNF:** Stack Underflow Flag bit(1)
1 = Stack underflow occurred
0 = Stack underflow did not occur

bit 5 **Unimplemented:** Read as '0'

bit 4-0 **SP<4:0>:** Stack Pointer Location bits

Note 1: Bits 7 and 6 are cleared by user software or by a POR.

REGISTER 9-3: INTCON3: INTERRUPT CONTROL REGISTER 3 (ACCESS FF0h)

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
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 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **INT3IE:** INT3 External Interrupt Enable bit
1 = Enables the INT3 external interrupt
0 = Disables the INT3 external interrupt

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

bit 2 **INT3IF:** INT3 External Interrupt Flag bit
1 = The INT3 external interrupt occurred (must be cleared in software)
0 = The INT3 external interrupt did not occur

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt did not occur

Note: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

10.7.3.1 Input Mapping

The inputs of the PPS options are mapped on the basis of the peripheral; that is, a control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping (see Register 10-6 through Register 10-20). Each register contains a 5-bit field, which is associated with one of the pin selectable peripherals. Programming a given peripheral's bit field with an appropriate 5-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any of the bit fields corresponds to the maximum number of peripheral pin selections supported by the device.

TABLE 10-13: SELECTABLE INPUT SOURCES (MAPS INPUT TO FUNCTION)(1)

Input Name	Function Name	Register	Configuration Bits
External Interrupt 1	INT1	RPINR1	INTR1R<4:0>
External Interrupt 2	INT2	RPINR2	INTR2R<4:0>
External Interrupt 3	INT3	RPINR3	INTR3R<4:0>
Timer0 External Clock Input	T0CKI	RPINR4	T0CKR<4:0>
Timer3 External Clock Input	T3CKI	RPINR6	T3CKR<4:0>
Input Capture 1	CCP1	RPINR7	IC1R<4:0>
Input Capture 2	CCP2	RPINR8	IC2R<4:0>
Timer1 Gate Input	T1G	RPINR12	T1GR<4:0>
Timer3 Gate Input	T3G	RPINR13	T3GR<4:0>
EUSART2 Asynchronous Receive/Synchronous Receive	RX2/DT2	RPINR16	RX2DT2R<4:0>
EUSART2 Asynchronous Clock Input	CK2	RPINR17	CK2R<4:0>
SPI2 Data Input	SDI2	RPINR21	SDI2R<4:0>
SPI2 Clock Input	SCK2IN	RPINR22	SCK2R<4:0>
SPI2 Slave Select Input	SS2IN	RPINR23	SS2R<4:0>
PWM Fault Input	FLT0	RPINR24	OCFAR<4:0>

Note 1: Unless otherwise noted, all inputs use the Schmitt Trigger input buffers.

11.1 Module Registers

The PMP module has a total of 14 Special Function Registers (SFRs) for its operation, plus one additional register to set configuration options. Of these, eight registers are used for control and six are used for PMP data transfer.

11.1.1 CONTROL REGISTERS

The eight PMP Control registers are:

- PMCONH and PMCONL
- PMMODEH and PMMODEL
- PMSTATL and PMSTATH
- PMEH and PMEL

The PMCON registers (Register 11-1 and Register 11-2) control basic module operations, including turning the module on or off. They also configure address multiplexing and control strobe configuration.

The PMMODE registers (Register 11-3 and Register 11-4) configure the various Master and Slave modes, the data width and interrupt generation.

The PMEH and PMEL registers (Register 11-5 and Register 11-6) configure the module's operation at the hardware (I/O pin) level.

The PMSTAT registers (Register 11-5 and Register 11-6) provide status flags for the module's input and output buffers, depending on the operating mode.

REGISTER 11-1: PMCONH: PARALLEL PORT CONTROL REGISTER HIGH BYTE (BANKED F5Fh)(1)

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPEN	—	—	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN
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 **PMPEN:** Parallel Master Port Enable bit
1 = PMP enabled
0 = PMP disabled, no off-chip access performed

bit 6-5 **Unimplemented:** Read as '0'

bit 4-3 **ADRMUX<1:0>:** Address/Data Multiplexing Selection bits
11 = Reserved
10 = All 16 bits of address are multiplexed on PMD<7:0> pins
01 = Lower 8 bits of address are multiplexed on PMD<7:0> pins (only eight bits of address are available in this mode)
00 = Address and data appear on separate pins (only eight bits of address are available in this mode)

bit 2 **PTBEEN:** Byte Enable Port Enable bit (16-Bit Master mode)
1 = PMBE port enabled
0 = PMBE port disabled

bit 1 **PTWREN:** Write Enable Strobe Port Enable bit
1 = PMWR/PMENB port enabled
0 = PMWR/PMENB port disabled

bit 0 **PTRDEN:** Read/Write Strobe Port Enable bit
1 = PMRD/PMWR port enabled
0 = PMRD/PMWR port disabled

Note 1: This register is only available in 44-pin devices.

11.3 MASTER PORT MODES

In its Master modes, the PMP module provides an 8-bit data bus, up to 16 bits of address, and all the necessary control signals to operate a variety of external parallel devices, such as memory devices, peripherals and slave microcontrollers. To use the PMP as a master, the module must be enabled (PMPEN = 1) and the mode must be set to one of the two possible Master modes (PMMODEH<1:0> = 10 or 11).

Because there are a number of parallel devices with a variety of control methods, the PMP module is designed to be extremely flexible to accommodate a range of configurations. Some of these features include:

- 8-Bit and 16-Bit Data modes on an 8-bit data bus
- Configurable address/data multiplexing
- Up to two chip select lines
- Up to 16 selectable address lines
- Address auto-increment and auto-decrement
- Selectable polarity on all control lines
- Configurable Wait states at different stages of the read/write cycle

11.3.1 PMP AND I/O PIN CONTROL

Multiple control bits are used to configure the presence or absence of control and address signals in the module. These bits are PTBEEN, PTWREN, PTRDEN and PTEN<15:0>. They give the user the ability to conserve pins for other functions and allow flexibility to control the external address. When any one of these bits is set, the associated function is present on its associated pin; when clear, the associated pin reverts to its defined I/O port function.

Setting a PTENx bit will enable the associated pin as an address pin and drive the corresponding data contained in the PMADDR register. Clearing a PTENx bit will force the pin to revert to its original I/O function.

For the pins configured as chip select (PMCS) with the corresponding PTENx bit set, the PTEN0 and PTEN1 bits will also control the PMALL and PMALH signals. When multiplexing is used, the associated address latch signals should be enabled.

11.3.2 READ/WRITE CONTROL

The PMP module supports two distinct read/write signaling methods. In Master Mode 1, read and write strobes are combined into a single control line, PMRD/PMWR. A second control line, PMENB, determines when a read or write action is to be taken. In Master Mode 2, separate read and write strobes (PMRD and PMWR) are supplied on separate pins.

All control signals (PMRD, PMWR, PMBE, PMENB, PMAL and PMCS) can be individually configured as either positive or negative polarity. Configuration is controlled by separate bits in the PMCONL register. Note that the polarity of control signals that share the same output pin (for example, PMWR and PMENB) are controlled by the same bit; the configuration depends on which Master Port mode is being used.

11.3.3 DATA WIDTH

The PMP supports data widths of both 8 bits and 16 bits. The data width is selected by the MODE16 bit (PMMODEH<2>). Because the data path into and out of the module is only 8 bits wide, 16-bit operations are always handled in a multiplexed fashion, with the Least Significant Byte (LSB) of data being presented first. To differentiate data bytes, the byte enable control strobe, PMBE, is used to signal when the Most Significant Byte (MSB) of data is being presented on the data lines.

11.3.4 ADDRESS MULTIPLEXING

In either of the Master modes (PMMODEH<1:0> = 1x), the user can configure the address bus to be multiplexed together with the data bus. This is accomplished by using the ADRMUX<1:0> bits (PMCONH<4:3>). There are three address multiplexing modes available; typical pinout configurations for these modes are displayed in Figure 11-9, Figure 11-10 and Figure 11-11.

In Demultiplexed mode (PMCONH<4:3> = 00), data and address information are completely separated. Data bits are presented on PMD<7:0> and address bits are presented on PMADDRH<6:0> and PMADDRL<7:0>.

In Partially Multiplexed mode (PMCONH<4:3> = 01), the lower eight bits of the address are multiplexed with the data pins on PMD<7:0>. The upper eight bits of address are unaffected and are presented on PMADDRH<6:0>. The PMA0 pin is used as an address latch, and presents the address latch low enable strobe (PMALL). The read and write sequences are extended by a complete CPU cycle during which the address is presented on the PMD<7:0> pins.

In Fully Multiplexed mode (PMCONH<4:3> = 10), the entire 16 bits of the address are multiplexed with the data pins on PMD<7:0>. The PMA0 and PMA1 pins are used to present address latch low enable (PMALL) and address latch high enable (PMALH) strobes, respectively. The read and write sequences are extended by two complete CPU cycles. During the first cycle, the lower eight bits of the address are presented on the PMD<7:0> pins with the PMALL strobe active. During the second cycle, the upper eight bits of the address are presented on the PMD<7:0> pins with the PMALH strobe active. In the event the upper address bits are configured as chip select pins, the corresponding address bits are automatically forced to '0'.

13.8.3 TIMER1 GATE TOGGLE MODE

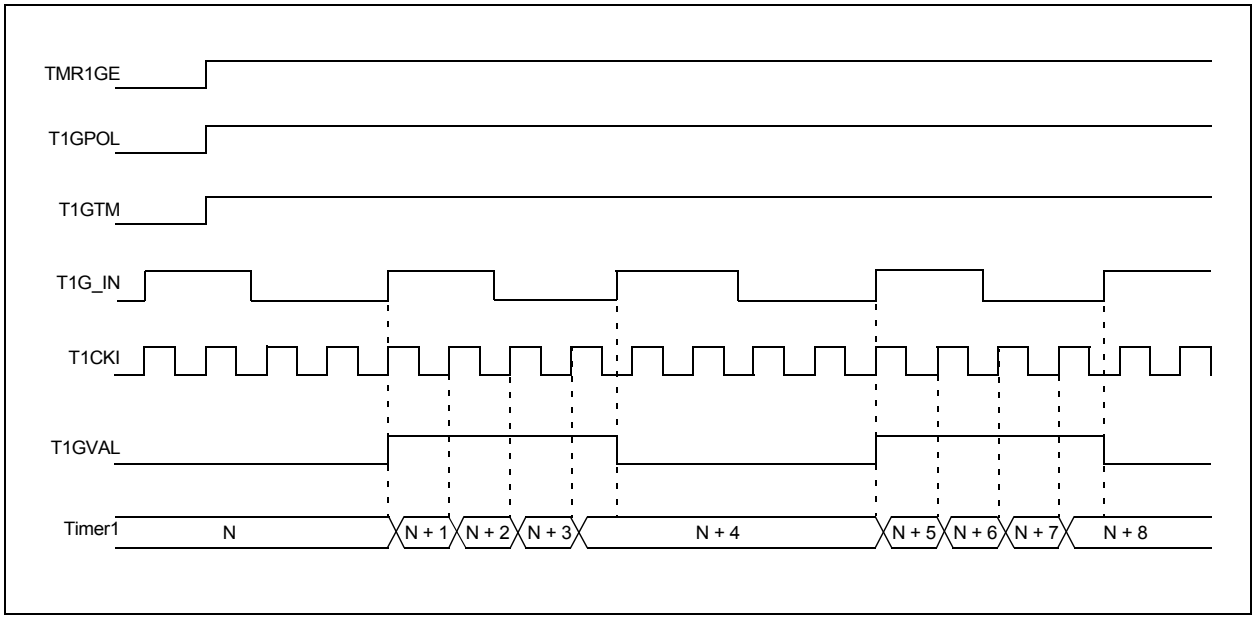
When Timer1 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer1 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See Figure 13-5 for timing details.

The T1GVAL bit will indicate when the Toggled mode is active and the timer is counting.

The Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When the T1GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

FIGURE 13-5: TIMER1 GATE TOGGLE MODE



17.1.3 ALRMVALH AND ALRMVALL REGISTER MAPPINGS

REGISTER 17-14: ALRMMNTH: ALARM MONTH VALUE REGISTER (ACCESS F8Fh, PTR 10b)(1)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
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 7-5 **Unimplemented:** Read as '0'

bit 4 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit bit
Contains a value of 0 or 1.

bit 3-0 **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-15: ALRMDAY: ALARM DAY VALUE REGISTER (ACCESS F8Eh, PTR 10b)(1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
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 **Unimplemented:** Read as '0'

bit 5-4 **DAYTEN<1:0>:** Binary Coded Decimal Value of Day's Tens Digit bits
Contains a value from 0 to 3.

bit 3-0 **DAYONE<3:0>:** Binary Coded Decimal Value of Day's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

When the RXINC bit is set, the RXADDR register will automatically increment after each received byte. Automatic receive address increment can be disabled by clearing the RXINC bit. If RXINC is disabled in Full-Duplex or Half-Duplex Receive modes, all incoming data bytes on SDI2 will overwrite the same memory location pointed to by the RXADDR register. After the SPI DMA transaction has completed, the last received byte will reside in the memory location pointed to by the RXADDR register.

The SPI DMA module can be used for either half-duplex receive only communication, half-duplex transmit only communication or full-duplex simultaneous transmit and receive operations. All modes are available for both SPI master and SPI slave configurations. The DUPLEX0 and DUPLEX1 bits can be used to select the desired operating mode.

The behavior of the DLYINTEN bit varies greatly depending on the SPI operating mode. For example behavior for each of the modes, see Figure 19-3 through Figure 19-6.

SPI Slave mode, DLYINTEN = 1: In this mode, an SSP2IF interrupt will be generated during a transfer if the time between successful byte transmission events is longer than the value set by the DLYCYC<3:0> bits in the DMACON2 register. This interrupt allows slave firmware to know that the master device is taking an unusually large amount of time between byte transmissions. For example, this information may be useful for implementing application-defined communication protocols involving time-outs if the bus remains Idle for too long. When DLYINTEN = 1, the DLYLVL<3:0> interrupts occur normally according to the selected setting.

SPI Slave mode, DLYINTEN = 0: In this mode, the time-out based interrupt is disabled. No additional SSP2IF interrupt events will be generated by the SPI DMA module, other than those indicated by the INTLVL<3:0> bits in the DMACON2 register. In this mode, always set DLYCYC<3:0> = 0000.

SPI Master mode, DLYINTEN = 0: The DLYCYC<3:0> bits in the DMACON2 register determine the amount of additional inter-byte delay, which is added by the SPI DMA module during a transfer. The Master mode $\overline{SS2}$ output feature may be used.

SPI Master mode, DLYINTEN = 1: The amount of hardware overhead is slightly reduced in this mode, and the minimum inter-byte delay is 8 TCY for FOSC/4, 9 TCY for FOSC/16 and 15 TCY for FOSC/64. This mode can potentially be used to obtain slightly higher effective SPI bandwidth. In this mode, the $\overline{SS2}$ control feature cannot be used, and should always be disabled (DMACON1<7:6> = 00). Additionally, the interrupt generating hardware (used in Slave mode) remains active. To avoid extraneous SSP2IF interrupt events, set the DMACON2 delay bits, DLYCYC<3:0> = 1111, and ensure that the SPI serial clock rate is no slower than FOSC/64.

In SPI Master modes, the DMAEN bit is used to enable the SPI DMA module and to initiate an SPI DMA transaction. After user firmware sets the DMAEN bit, the DMA hardware will begin transmitting and/or receiving data bytes according to the configuration used. In SPI Slave modes, setting the DMAEN bit will finish the initialization steps needed to prepare the SPI DMA module for communication (which must still be initiated by the master device).

To avoid possible data corruption, once the DMAEN bit is set, user firmware should not attempt to modify any of the MSSP2 or SPI DMA related registers, with the exception of the INTLVL bits in the DMACON2 register.

If user firmware wants to halt an ongoing DMA transaction, the DMAEN bit can be manually cleared by the firmware. Clearing the DMAEN bit while a byte is currently being transmitted will not immediately halt the byte in progress. Instead, any byte currently in progress will be completed before the MSSP2 and SPI DMA modules go back to their Idle conditions. If user firmware clears the DMAEN bit, the TXADDR, RXADDR and DMABC registers will no longer update, and the DMA module will no longer make any additional read or writes to SRAM; therefore, state information can be lost.

22.0 COMPARATOR MODULE

The analog comparator module contains two comparators that can be independently configured in a variety of ways. The inputs can be selected from the analog inputs and two internal voltage references. The digital outputs are available at the pin level and can also be read through the control register. Multiple output and interrupt event generation is also available. Figure 22-1 provides a generic single comparator from the module.

Key features of the module are:

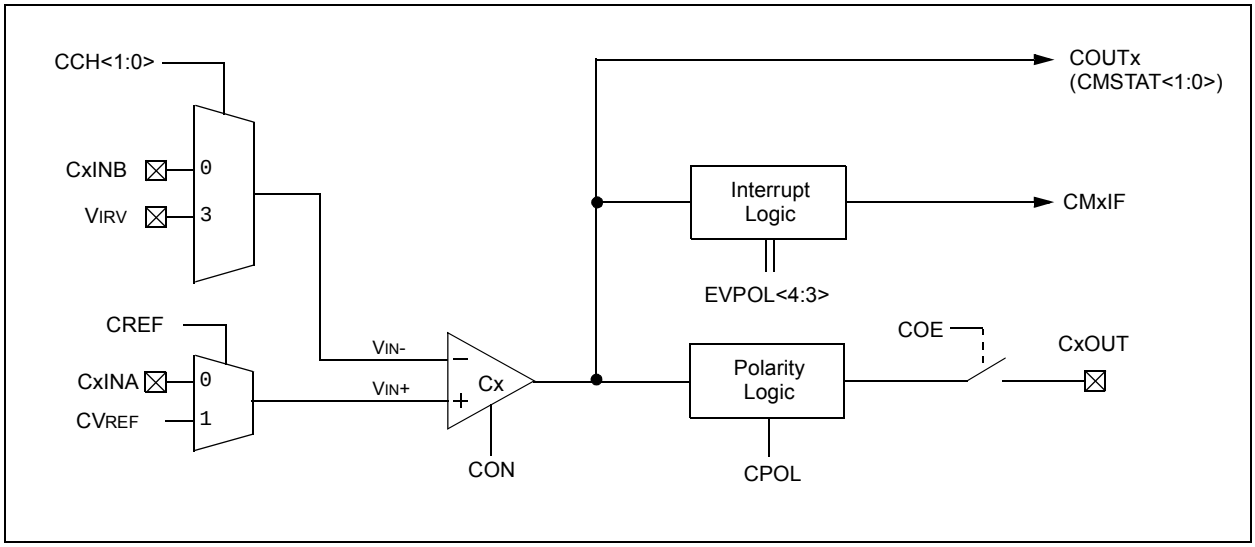
- Independent comparator control
- Programmable input configuration
- Output to both pin and register levels
- Programmable output polarity
- Independent interrupt generation for each comparator with configurable interrupt-on-change

22.1 Registers

The CMxCON registers (Register 22-1) select the input and output configuration for each comparator, as well as the settings for interrupt generation.

The CMSTAT register (Register 22-2) provides the output results of the comparators. The bits in this register are read-only.

FIGURE 22-1: COMPARATOR SIMPLIFIED BLOCK DIAGRAM



22.6 Comparator Interrupts

The comparator interrupt flag is set whenever any of the following occurs:

- Low-to-high transition of the comparator output
- High-to-low transition of the comparator output
- Any change in the comparator output

The comparator interrupt selection is done by the EVPOL<1:0> bits in the CMxCON register (CMxCON<4:3>).

In order to provide maximum flexibility, the output of the comparator may be inverted using the CPOL bit in the CMxCON register (CMxCON<5>). This is functionally identical to reversing the inverting and non-inverting inputs of the comparator for a particular mode.

An interrupt is generated on the low-to-high or high-to-low transition of the comparator output. This mode of interrupt generation is dependent on EVPOL<1:0> in the CMxCON register. When EVPOL<1:0> = 01 or 10, the interrupt is generated on a low-to-high or high-to-low transition of the comparator output. Once the interrupt is generated, it is required to clear the interrupt flag by software.

When EVPOL<1:0> = 11, the comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMSTAT<1:0>, to determine the actual change that occurred. The CMxIF bits (PIR2<6:5>) are the Comparator Interrupt Flags. The CMxIF bits must be reset by clearing them. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

Table 22-2 provides the interrupt generation corresponding to comparator input voltages and EVPOL bit settings.

Both the CMxIE bits (PIE2<6:5>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMxIF bits will still be set if an interrupt condition occurs.

Figure 22-1 provides a simplified diagram of the interrupt section.

TABLE 22-2: COMPARATOR INTERRUPT GENERATION

CPOL	EVPOL<1:0>	Comparator Input Change	COUTx Transition	Interrupt Generated
0	00	$V_{IN+} > V_{IN-}$	Low-to-High	No
		$V_{IN+} < V_{IN-}$	High-to-Low	No
	01	$V_{IN+} > V_{IN-}$	Low-to-High	Yes
		$V_{IN+} < V_{IN-}$	High-to-Low	No
	10	$V_{IN+} > V_{IN-}$	Low-to-High	No
		$V_{IN+} < V_{IN-}$	High-to-Low	Yes
	11	$V_{IN+} > V_{IN-}$	Low-to-High	Yes
		$V_{IN+} < V_{IN-}$	High-to-Low	Yes
1	00	$V_{IN+} > V_{IN-}$	High-to-Low	No
		$V_{IN+} < V_{IN-}$	Low-to-High	No
	01	$V_{IN+} > V_{IN-}$	High-to-Low	No
		$V_{IN+} < V_{IN-}$	Low-to-High	Yes
	10	$V_{IN+} > V_{IN-}$	High-to-Low	Yes
		$V_{IN+} < V_{IN-}$	Low-to-High	No
	11	$V_{IN+} > V_{IN-}$	High-to-Low	Yes
		$V_{IN+} < V_{IN-}$	Low-to-High	Yes

EXAMPLE 25-2: CURRENT CALIBRATION ROUTINE

```
#include <p18cxxx.h>

#define COUNT 500                           //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define RCAL .027                           //R value is 4200000 (4.2M)
                                            //scaled so that result is in
                                            //1/100th of uA
#define ADSCALE 1023                        //for unsigned conversion 10 sig bits
#define ADREF 3.3                           //Vdd connected to A/D Vr+

int main(void)
{
    int i;
    int j = 0;                              //index for loop
    unsigned int Vread = 0;
    double VTot = 0;
    float Vavg=0, Vcal=0, CTMUISrc = 0;     //float values stored for calcs

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONHbits.CTMUEN = 1;                    //Enable the CTMU
CTMUCONLbits.EDG1STAT = 0;                  // Set Edge status bits to zero
CTMUCONLbits.EDG2STAT = 0;
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1;           //drain charge on the circuit
        DELAY;                              //wait 125us
        CTMUCONHbits.IDISSEN = 0;           //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;          //Begin charging the circuit
                                            //using CTMU current source
        DELAY;                              //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;          //Stop charging circuit

        PIR1bits.ADIF = 0;                  //make sure A/D Int not set
        ADCON0bits.GO=1;                    //and begin A/D conv.
        while(!PIR1bits.ADIF);              //Wait for A/D convert complete

        Vread = ADRES;                      //Get the value from the A/D
        PIR1bits.ADIF = 0;                  //Clear A/D Interrupt Flag
        VTot += Vread;                      //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);            //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;                   //CTMUISrc is in 1/100ths of uA

}
```

25.3.2 CAPACITANCE CALIBRATION

There is a small amount of capacitance from the internal A/D Converter sample capacitor as well as stray capacitance from the circuit board traces and pads that affect the precision of capacitance measurements. A measurement of the stray capacitance can be taken by making sure the desired capacitance to be measured has been removed. The measurement is then performed using the following steps:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT (= 1).
3. Wait for a fixed delay of time t.
4. Clear EDG1STAT.
5. Perform an A/D conversion.
6. Calculate the stray and A/D sample capacitances:

$$C_{OFFSET} = C_{STRAY} + C_{AD} = (I \cdot t)/V$$

where I is known from the current source measurement step, t is a fixed delay and V is measured by performing an A/D conversion.

This measured value is then stored and used for calculations of time measurement or subtracted for capacitance measurement. For calibration, it is expected that the capacitance of $C_{STRAY} + C_{AD}$ is approximately known. C_{AD} is approximately 4 pF.

An iterative process may need to be used to adjust the time, t, that the circuit is charged to obtain a reasonable voltage reading from the A/D Converter. The value of t may be determined by setting C_{OFFSET} to a theoretical value, then solving for t. For example, if C_{STRAY} is theoretically calculated to be 11 pF, and V is expected to be 70% of V_{DD}, or 2.31V, then t would be:

$$(4\ pF + 11\ pF) \bullet 2.31V/0.55\ \mu A$$

or 63 μs.

See Example 25-3 for a typical routine for CTMU capacitance calibration.

TABLE 26-1: MAPPING OF THE FLASH CONFIGURATION WORDS TO THE CONFIGURATION REGISTERS

Configuration Register (Volatile)	Configuration Register Address	Flash Configuration Byte Address
CONFIG1L	300000h	XXXF8h
CONFIG1H	300001h	XXXF9h
CONFIG2L	300002h	XXXFAh
CONFIG2H	300003h	XXXFBh
CONFIG3L	300004h	XXXFCh
CONFIG3H	300005h	XXXFDh
CONFIG4L	300006h	XXXFEh
CONFIG4H	300007h	XXXFFh

TABLE 26-2: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprog. Value(1)
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN	111- ---1
300001h	CONFIG1H	—(2)	—(2)	—(2)	—(2)	—	CP0	—	—	1111 -1--
300002h	CONFIG2L	IESO	FCMEN	—	LPT1OSC	T1DIG	FOSC2	FOSC1	FOSC0	11-1 1111
300003h	CONFIG2H	—(2)	—(2)	—(2)	—(2)	WDTPS3	WDTPS2	WDTPS1	WDTPS0	1111 1111
300004h	CONFIG3L	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0	DSWDTEN	DSBOREN	RTCOSC	DSWDTOSC	1111 1111
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	MSSPMSK	—	—	IOL1WAY	1111 1--1
300006h	CONFIG4L	WPCFG	WPEND	WPFP5	WPFP4	WPFP3	WPFP2	WPFP1	WPFP0	1111 1111
300007h	CONFIG4H	—(2)	—(2)	—(2)	—(2)	—	—	—	WPDIS	1111 ---1
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxx0 0000(3)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0100 00xx(3)

Legend: x = unknown, u = unchanged, - = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: Values reflect the unprogrammed state as received from the factory and following Power-on Resets. In all other Reset states, the configuration bytes maintain their previously programmed states.

2: The value of these bits in program memory should always be programmed to '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

3: See Register 26-9 and Register 26-10 for DEVID values. These registers are read-only and cannot be programmed by the user.

REGISTER 26-11: WDTCON: WATCHDOG TIMER CONTROL REGISTER (ACCESS FC0h)

R/W-1	R-x	R-x	U-0	R-0	R/W-0	R/W-0	R/W-0
REGSLP(2)	LVDSTAT(2)	ULPLVL	—	DS(2)	ULPEN	ULPSINK	SWDTEN(1)
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 **REGSLP:** Voltage Regulator Low-Power Operation Enable bit(2)
1 = On-chip regulator enters low-power operation when device enters Sleep mode
0 = On-chip regulator is active even in Sleep mode

bit 6 **LVDSTAT:** Low-Voltage Detect Status bit(2)
1 = V_{DDCORE} > 2.45V nominal
0 = V_{DDCORE} < 2.45V nominal

bit 5 **ULPLVL:** Ultra Low-Power Wake-up Output bit (not valid unless ULPEN = 1)
1 = Voltage on RA0 > ~0.5V
0 = Voltage on RA0 < ~0.5V

bit 4 **Unimplemented**: Read as '0'

bit 3 **DS:** Deep Sleep Wake-up Status bit (used in conjunction with RCON, POR and BOR bits to determine Reset source)(2)
1 = If the last exit from POR was caused by a normal wake-up from Deep Sleep
0 = If the last exit from POR was a result of hard cycling V_{DD}, or if the Deep Sleep BOR was enabled and detected, a (V_{DD} < V_{DSBOR}) and (V_{DD} < V_{POR}) condition

bit 2 **ULPEN:** Ultra Low-Power Wake-up Module Enable bit
1 = Ultra Low-Power Wake-up module is enabled; ULPLVL bit indicates comparator output
0 = Ultra Low-Power Wake-up module is disabled

bit 1 **ULPSINK:** Ultra Low-Power Wake-up Current Sink Enable bit
1 = Ultra Low-Power Wake-up current sink is enabled (if ULPEN = 1)
0 = Ultra Low-Power Wake-up current sink is disabled

bit 0 **SWDTEN:** Software Controlled Watchdog Timer Enable bit(1)
1 = Watchdog Timer is on
0 = Watchdog Timer is off

Note 1: This bit has no effect if the Configuration bit, WDTEN, is enabled.
2: Not available on devices where the on-chip voltage regulator is disabled ("LF" devices).

TABLE 26-3: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
RCON	IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	70
WDTCON	REGSLP	LVDSTAT	ULPLVL	—	DS	ULPEN	ULPSINK	SWDTEN	70

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Watchdog Timer.

BNC **Branch if Not Carry**

Syntax: BNC n

Operands: -128 ≤ n ≤ 127

Operation: if Carry bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the Carry bit is '0', 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 two-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 BNC Jump

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

BNN **Branch if Not Negative**

Syntax: BNN n

Operands: -128 ≤ n ≤ 127

Operation: if Negative bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the Negative bit is '0', 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 two-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 BNN Jump

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

CALLW **Subroutine Call using WREG**

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

0000	0000	0001	0100

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched.

Unlike CALL, there is no option to update W, STATUS or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	Push PC to stack	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE    CALLW
```

Before Instruction
PC = address (HERE)
PCLATH = 10h
PCLATU = 00h
W = 06h

After Instruction
PC = 001006h
TOS = address (HERE + 2)
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF **Move Indexed to f**

Syntax: MOVSF [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 4095$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (source)	1110	1011	0zzz	$zzzz_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s', in the first word, to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh).

The MOVSF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

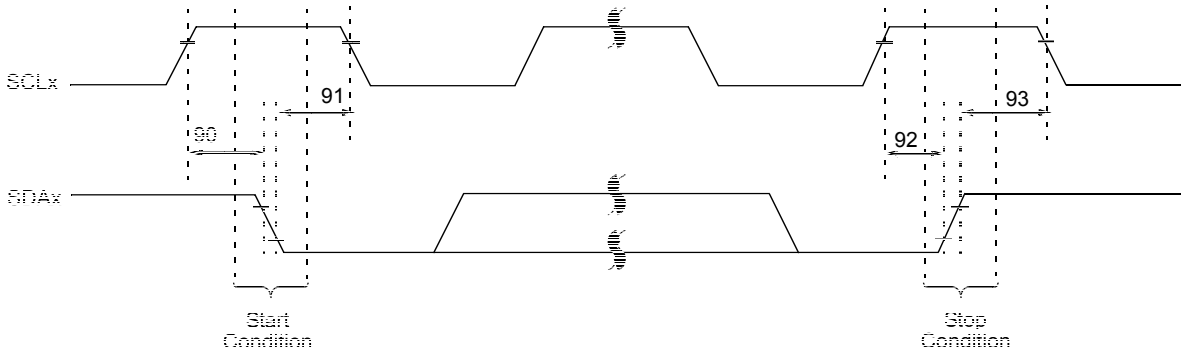
Example:

```
MOVSF   [0x05], REG2
```

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h

After Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 33h

FIGURE 29-17: I²C™ BUS START/STOP BITS TIMING

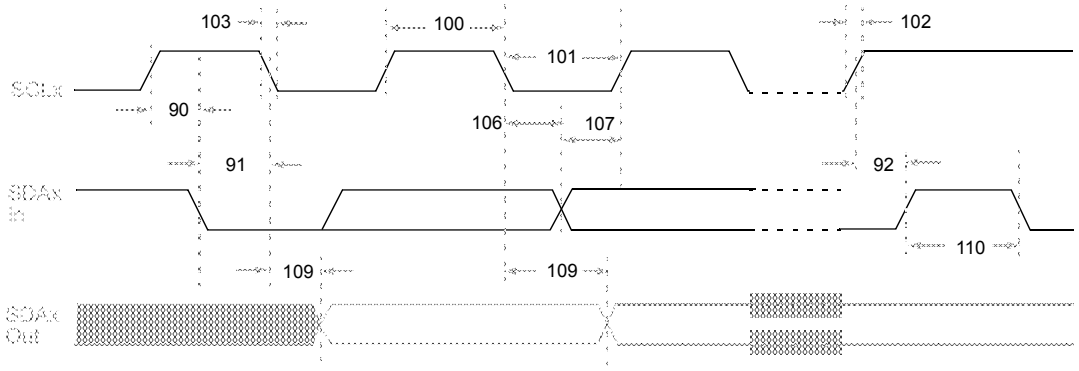


Note: Refer to Figure 29-4 for load conditions.

TABLE 29-24: I²C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4700	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4700	—	ns	
			400 kHz mode	600	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	ns	
			400 kHz mode	600	—		

FIGURE 29-18: I²C™ BUS DATA TIMING



Note: Refer to Figure 29-4 for load conditions.