



Welcome to [E-XFL.COM](https://www.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](#)"

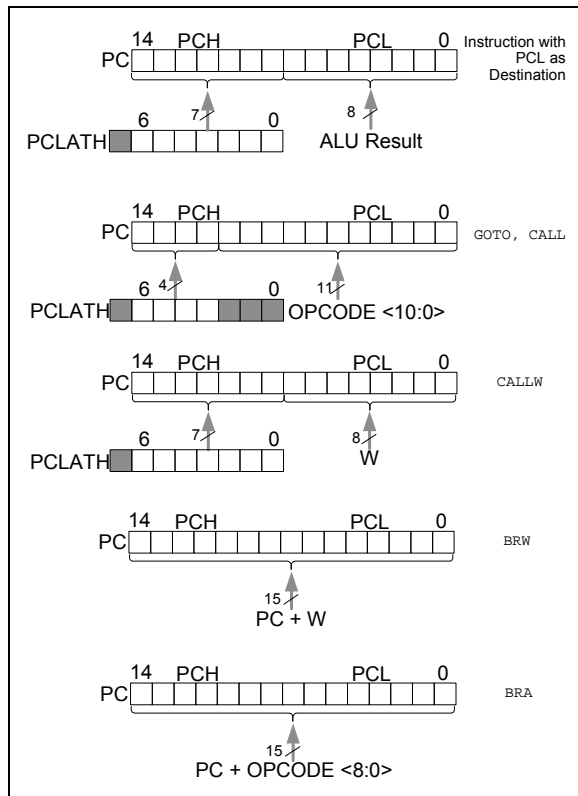
Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	32MHz
Connectivity	I ² C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LCD, POR, PWM, WDT
Number of I/O	54
Program Memory Size	28KB (16K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 17x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1947-e-pt

3.4 PCL and PCLATH

The Program Counter (PC) is 15-bit wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-4 shows the five situations for the loading of the PC.

FIGURE 3-4: LOADING OF PC IN DIFFERENT SITUATIONS



3.4.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper seven bits to the PCLATH register. When the lower eight bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.4.2 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (`ADDWF PCL`). When performing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to the Application Note AN556, *Implementing a Table Read* (DS00556).

3.4.3 COMPUTED FUNCTION CALLS

A computed function `CALL` allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function `CALL`, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the `CALL` instruction, the PCH<2:0> and PCL registers are loaded with the operand of the `CALL` instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The `CALLW` instruction enables computed calls by combining PCLATH and W to form the destination address. A computed `CALLW` is accomplished by loading the W register with the desired address and executing `CALLW`. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.4.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, `BRW` and `BRA`. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using `BRW`, load the W register with the desired unsigned address and execute `BRW`. The entire PC will be loaded with the address `PC + 1 + W`.

If using `BRA`, the entire PC will be loaded with `PC + 1 +`, the signed value of the operand of the `BRA` instruction.

PIC16(L)F1946/1947

FIGURE 3-6: ACCESSING THE STACK EXAMPLE 2

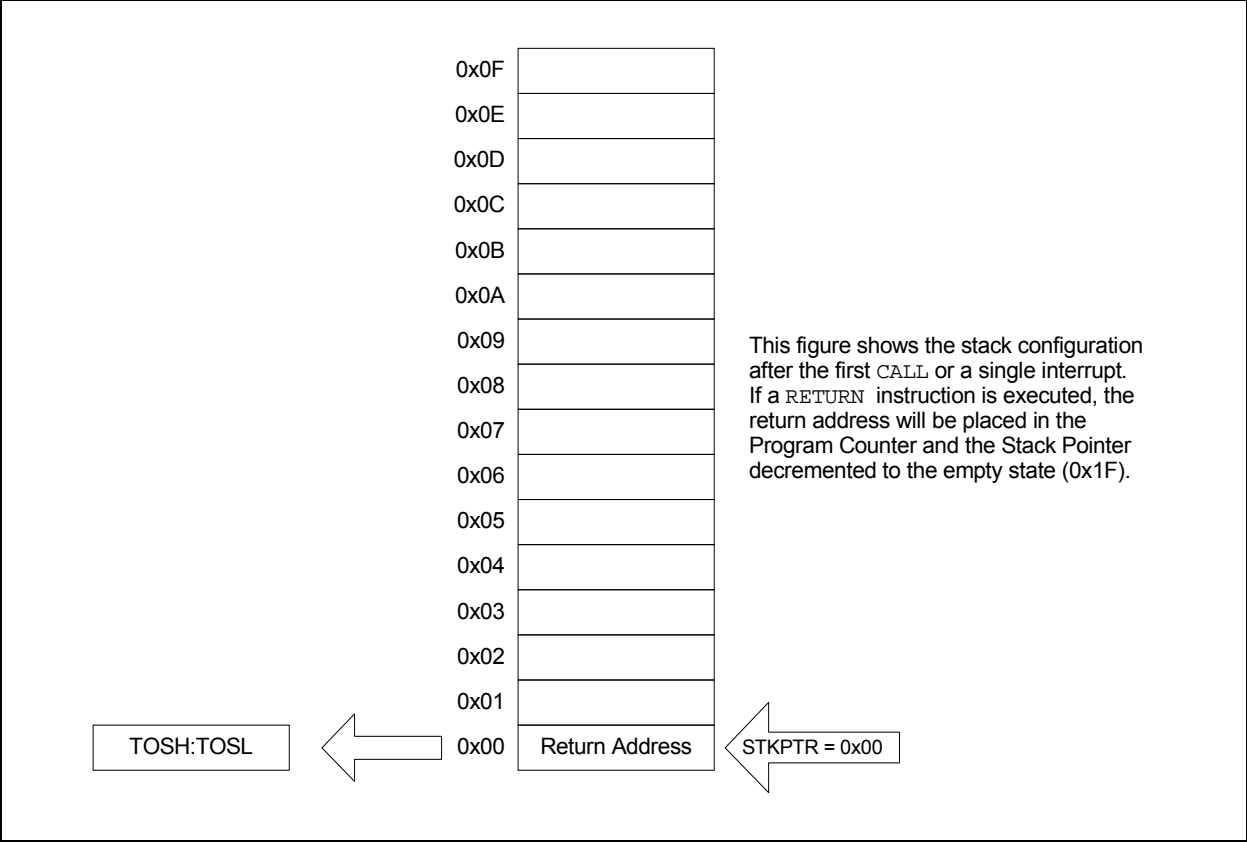


FIGURE 3-7: ACCESSING THE STACK EXAMPLE 3

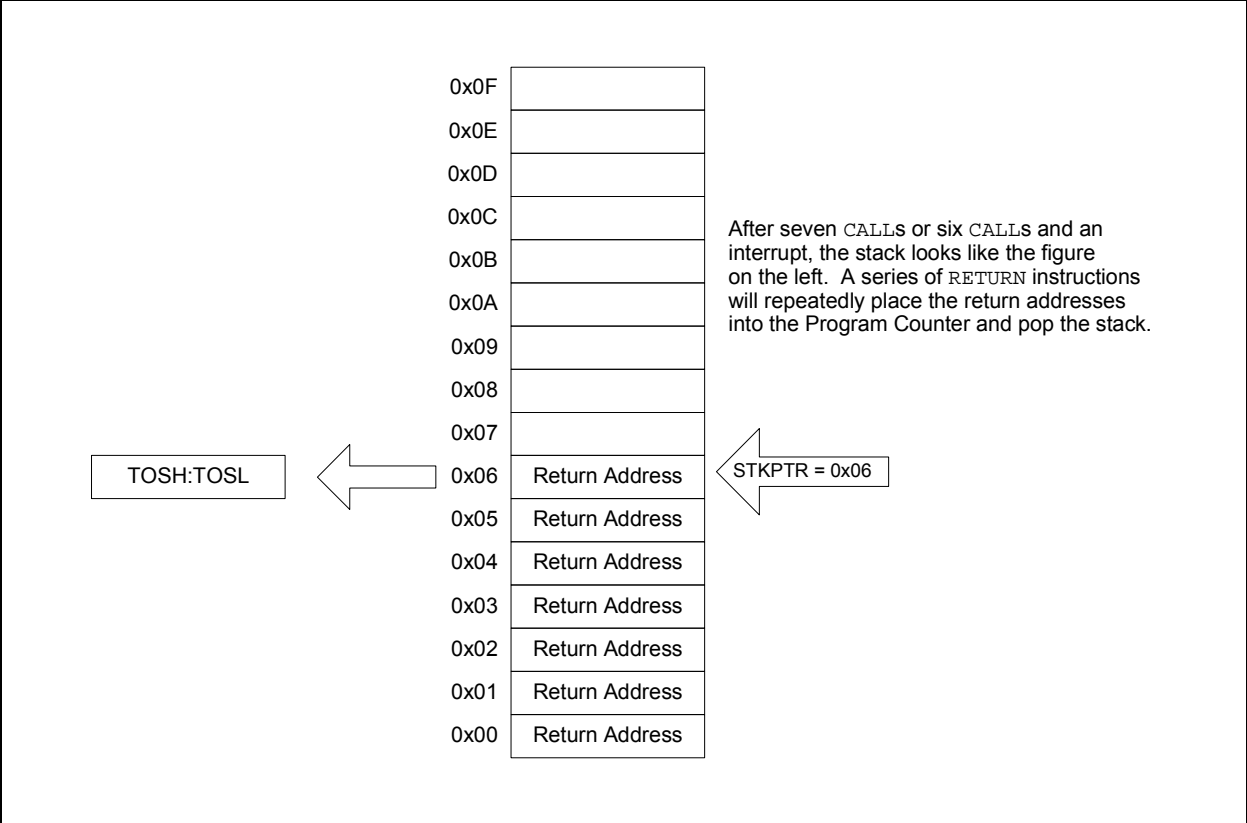
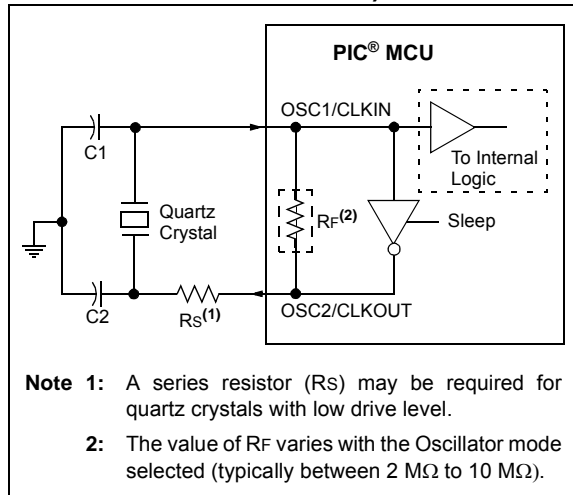


FIGURE 5-3: QUARTZ CRYSTAL OPERATION (LP, XT OR HS MODE)

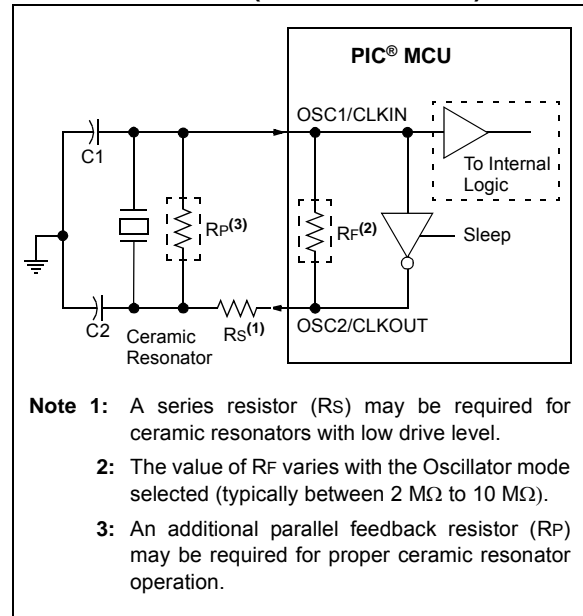


Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

- 2:** Always verify oscillator performance over the V_{DD} and temperature range that is expected for the application.
- 3:** For oscillator design assistance, reference the following Microchip Applications Notes:

- AN826, *Crystal Oscillator Basics and Crystal Selection for μ PIC[®] and PIC[®] Devices* (DS00826)
- AN849, *Basic PICmicro[®] Oscillator Design* (DS00849)
- AN943, *Practical PICmicro[®] Oscillator Analysis and Design* (DS00943)
- AN949, *Making Your Oscillator Work* (DS00949)

FIGURE 5-4: CERAMIC RESONATOR OPERATION (XT OR HS MODE)



5.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for LP, XT or HS modes, the Oscillator Start-up Timer (OST) counts 1024 oscillations from OSC1. This occurs following a Power-on Reset (POR) and when the Power-up Timer (PWRT) has expired (if configured), or a wake-up from Sleep. During this time, the program counter does not increment and program execution is suspended. The OST ensures that the oscillator circuit, using a quartz crystal resonator or ceramic resonator, has started and is providing a stable system clock to the oscillator module.

In order to minimize latency between external oscillator start-up and code execution, the Two-Speed Clock Start-up mode can be selected (see **Section 5.4 "Two-Speed Clock Start-up Mode"**).

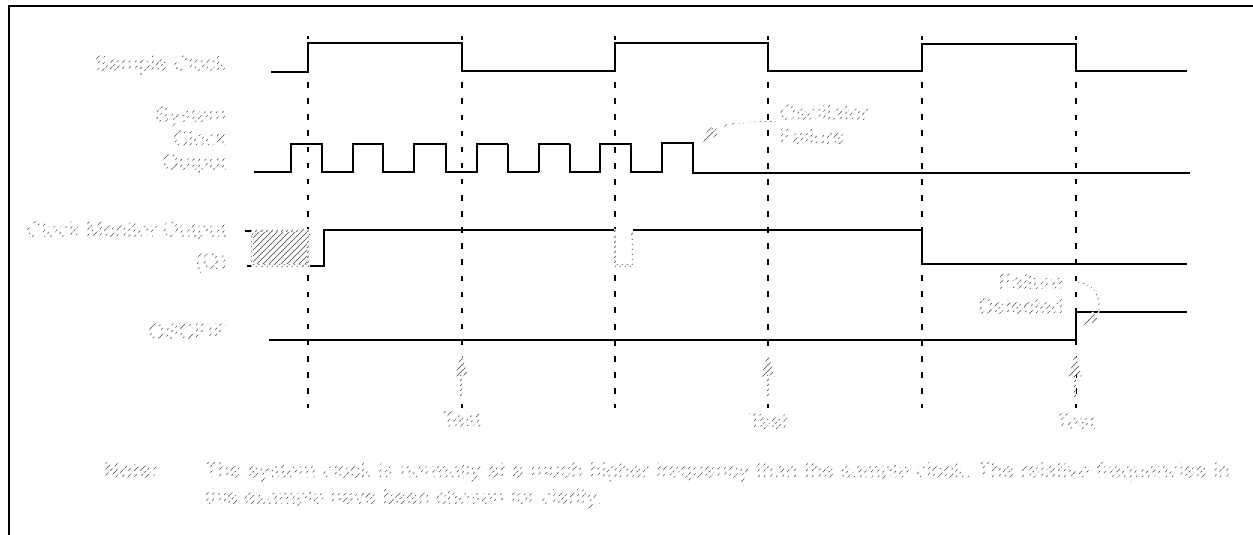
5.2.1.4 4x PLL

The oscillator module contains a 4x PLL that can be used with both external and internal clock sources to provide a system clock source. The input frequency for the 4x PLL must fall within specifications. See the PLL Clock Timing Specifications in **Section 30.0 "Electrical Specifications"**.

The 4x PLL may be enabled for use by one of two methods:

1. Program the PLEN bit in Configuration Words to a '1'.
2. Write the SPLLEN bit in the OSCCON register to a '1'. If the PLEN bit in Configuration Words is programmed to a '1', then the value of SPLLEN is ignored.

FIGURE 5-10: FSCM TIMING DIAGRAM



11.0 DATA EEPROM AND FLASH PROGRAM MEMORY CONTROL

The data EEPROM and Flash program memory are readable and writable during normal operation (full V_{DD} range). These memories are not directly mapped in the register file space. Instead, they are indirectly addressed through the Special Function Registers (SFRs). There are six SFRs used to access these memories:

- EECON1
- EECON2
- EEDATL
- EEDATH
- EEADRL
- EEADRH

When interfacing the data memory block, EEDATL holds the 8-bit data for read/write, and EEADRL holds the address of the EEDATL location being accessed. These devices have 256 bytes of data EEPROM with an address range from 0h to 0FFh.

When accessing the program memory block, the EEDATH:EEDATL register pair forms a 2-byte word that holds the 14-bit data for read/write, and the EEADRL and EEADRH registers form a 2-byte word that holds the 15-bit address of the program memory location being read.

The EEPROM data memory allows byte read and write. An EEPROM byte write automatically erases the location and writes the new data (erase before write).

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump rated to operate over the voltage range of the device for byte or word operations.

Depending on the setting of the Flash Program Memory Self Write Enable bits $WRT<1:0>$ of the Configuration Words, the device may or may not be able to write certain blocks of the program memory. However, reads from the program memory are always allowed.

When the device is code-protected, the device programmer can no longer access data or program memory. When code-protected, the CPU may continue to read and write the data EEPROM memory and Flash program memory.

11.1 EEADRL and EEADRH Registers

The EEADRH:EEADRL register pair can address up to a maximum of 256 bytes of data EEPROM or up to a maximum of 32K words of program memory.

When selecting a program address value, the MSB of the address is written to the EEADRH register and the LSB is written to the EEADRL register. When selecting a EEPROM address value, only the LSB of the address is written to the EEADRL register.

11.1.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for EE memory accesses.

Control bit EEP_{GD} determines if the access will be a program or data memory access. When clear, any subsequent operations will operate on the EEPROM memory. When set, any subsequent operations will operate on the program memory. On Reset, EEPROM is selected by default.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The $WREN$ bit, when set, will allow a write operation to occur. On power-up, the $WREN$ bit is clear. The $WRERR$ bit is set when a write operation is interrupted by a Reset during normal operation. In these situations, following Reset, the user can check the $WRERR$ bit and execute the appropriate error handling routine.

Interrupt flag bit $EEIF$ of the $PIR2$ register is set when write is complete. It must be cleared in the software.

Reading $EECON2$ will read all '0's. The $EECON2$ register is used exclusively in the data EEPROM write sequence. To enable writes, a specific pattern must be written to $EECON2$.

PIC16(L)F1946/47

EXAMPLE 11-5: WRITING TO FLASH PROGRAM MEMORY

```
; This write routine assumes the following:
; 1. The 16 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
;    stored in little endian format
; 3. A valid starting address (the least significant bits = 000) is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
    BCF      INTCON,GIE      ; Disable ints so required sequences will execute properly
    BANKSEL  EEADRH         ; Bank 3
    MOVF     ADDRH,W         ; Load initial address
    MOVWF    EEADRH         ;
    MOVF     ADDRL,W        ;
    MOVWF    EEADRL         ;
    MOVLW    LOW DATA_ADDR  ; Load initial data address
    MOVWF    FSR0L          ;
    MOVLW    HIGH DATA_ADDR ; Load initial data address
    MOVWF    FSR0H          ;
    BSF      EECON1,EEPGD    ; Point to program memory
    BCF      EECON1,CFGSR    ; Not configuration space
    BSF      EECON1,WREN     ; Enable writes
    BSF      EECON1,LWLO     ; Only Load Write Latches

LOOP
    MOVIW    FSR0++          ; Load first data byte into lower
    MOVWF    EEDATL         ;
    MOVIW    FSR0++          ; Load second data byte into upper
    MOVWF    EEDATH         ;

    MOVF     EEADRL,W        ; Check if lower bits of address are '000'
    XORLW    0x07           ; Check if we're on the last of 8 addresses
    ANDLW    0x07           ;
    BTFSC    STATUS,Z        ; Exit if last of eight words,
    GOTO     START_WRITE     ;

    Required Sequence
    MOVLW    55h             ; Start of required write sequence:
    MOVWF    EECON2          ; Write 55h
    MOVLW    0AAh           ;
    MOVWF    EECON2          ; Write AAh
    BSF      EECON1,WR       ; Set WR bit to begin write
    NOP      ; Any instructions here are ignored as processor
    NOP      ; halts to begin write sequence
    NOP      ; Processor will stop here and wait for write to complete.

    ; After write processor continues with 3rd instruction.

    INCF     EEADRL,F        ; Still loading latches Increment address
    GOTO     LOOP           ; Write next latches

START_WRITE
    BCF      EECON1,LWLO     ; No more loading latches - Actually start Flash program
    ; memory write

    Required Sequence
    MOVLW    55h             ; Start of required write sequence:
    MOVWF    EECON2          ; Write 55h
    MOVLW    0AAh           ;
    MOVWF    EECON2          ; Write AAh
    BSF      EECON1,WR       ; Set WR bit to begin write
    NOP      ; Any instructions here are ignored as processor
    NOP      ; halts to begin write sequence
    NOP      ; Processor will stop here and wait for write complete.

    ; after write processor continues with 3rd instruction

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

REGISTER 12-9: WPUB: WEAK PULL-UP PORTB REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-0 **WPUB<7:0>**: Weak Pull-up Register bits
 1 = Pull-up enabled
 0 = Pull-up disabled

- Note 1:** Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
Note 2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 12-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	90
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	148
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	148
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	148
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	128
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	330
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	330
LCDSE4	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	330
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			188
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	128
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ $\overline{\text{DONE}}$	T1GVAL	T1GSS<1:0>		198
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	128
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	129

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTB.

18.0 COMPARATOR MODULE

Comparators are used to interface analog circuits to a digital circuit by comparing two analog voltages and providing a digital indication of their relative magnitudes. Comparators are very useful mixed signal building blocks because they provide analog functionality independent of program execution. The analog comparator module includes the following features:

- Independent comparator control
- Programmable input selection
- Comparator output is available internally/externally
- Programmable output polarity
- Interrupt-on-change
- Wake-up from Sleep
- Programmable Speed/Power optimization
- PWM shutdown
- Programmable and fixed voltage reference

18.1 Comparator Overview

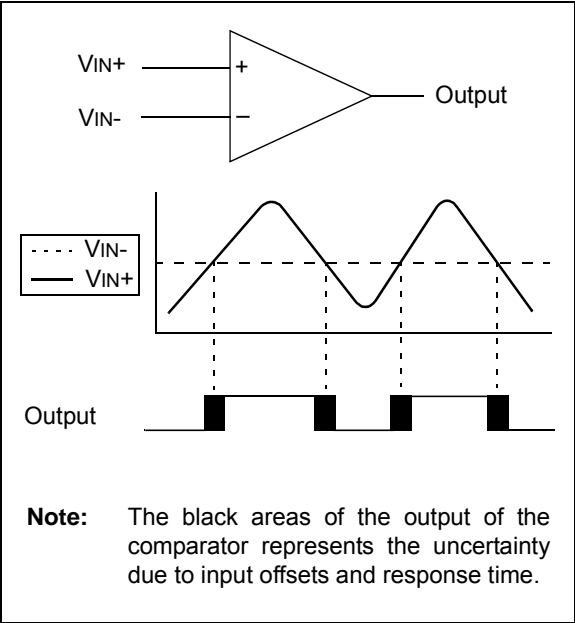
A single comparator is shown in Figure 18-1 along with the relationship between the analog input levels and the digital output. When the analog voltage at V_{IN+} is less than the analog voltage at V_{IN-} , the output of the comparator is a digital low level. When the analog voltage at V_{IN+} is greater than the analog voltage at V_{IN-} , the output of the comparator is a digital high level.

The comparators available for this device are located in Table 18-1.

TABLE 18-1: COMPARATOR AVAILABILITY PER DEVICE

Device	1	2	3
PIC16(L)F1946	•	•	•
PIC16(L)F1947	•	•	•

FIGURE 18-1: SINGLE COMPARATOR



PIC16(L)F1946/47

19.4 Register Definitions: SR Latch Control

REGISTER 19-2: SRCON0: SR LATCH CONTROL 0 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/S-0/0	R/S-0/0
SRLEN	SRCLK<2:0>			SRQEN	SRNQEN	SRPS	SRPR
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	S = Bit is set only

- bit 7 **SRLEN:** SR Latch Enable bit
1 = SR Latch is enabled
0 = SR Latch is disabled
- bit 6-4 **SRCLK<2:0>:** SR Latch Clock Divider bits
111 = Generates a 1 Fosc wide pulse every 512th Fosc cycle clock
110 = Generates a 1 Fosc wide pulse every 256th Fosc cycle clock
101 = Generates a 1 Fosc wide pulse every 128th Fosc cycle clock
100 = Generates a 1 Fosc wide pulse every 64th Fosc cycle clock
011 = Generates a 1 Fosc wide pulse every 32nd Fosc cycle clock
010 = Generates a 1 Fosc wide pulse every 16th Fosc cycle clock
001 = Generates a 1 Fosc wide pulse every 8th Fosc cycle clock
000 = Generates a 1 Fosc wide pulse every 4th Fosc cycle clock
- bit 3 **SRQEN:** SR Latch Q Output Enable bit
If SRLEN = 1:
1 = Q is present on the SRQ pin
0 = External Q output is disabled
If SRLEN = 0:
SR Latch is disabled
- bit 2 **SRNQEN:** SR Latch \overline{Q} Output Enable bit
If SRLEN = 1:
1 = \overline{Q} is present on the SRnQ pin
0 = External \overline{Q} output is disabled
If SRLEN = 0:
SR Latch is disabled
- bit 1 **SRPS:** Pulse Set Input of the SR Latch bit⁽¹⁾
1 = Pulse set input for 1 Q-clock period
0 = No effect on set input
- bit 0 **SRPR:** Pulse Reset Input of the SR Latch bit⁽¹⁾
1 = Pulse Reset input for 1 Q-clock period
0 = No effect on Reset input

Note 1: Set only, always reads back '0'.

PIC16(L)F1946/47

21.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter which is accessed through the TMR1H:TMR1L register pair. Writes to TMR1H or TMR1L directly update the counter.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1 is enabled by configuring the TMR1ON and TMR1GE bits in the T1CON and T1GCON registers, respectively. Table 21-1 displays the Timer1 enable selections.

TABLE 21-1: TIMER1 ENABLE SELECTIONS

TMR1ON	TMR1GE	Timer1 Operation
0	0	Off
0	1	Off
1	0	Always On
1	1	Count Enabled

21.2 Clock Source Selection

The TMR1CS<1:0> and T1OSCEN bits of the T1CON register are used to select the clock source for Timer1. Table 21-2 displays the clock source selections.

21.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, the TMR1H:TMR1L register pair will increment on multiples of Fosc as determined by the Timer1 prescaler.

When the Fosc internal clock source is selected, the Timer1 register value will increment by four counts every instruction clock cycle. Due to this condition, a 2 LSB error in resolution will occur when reading the Timer1 value. To utilize the full resolution of Timer1, an asynchronous input signal must be used to gate the Timer1 clock input.

The following asynchronous sources may be used:

- Asynchronous event on the T1G pin to Timer1 gate
- C1 or C2 comparator input to Timer1 gate

21.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input T1CKI or the capacitive sensing oscillator signal. Either of these external clock sources can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

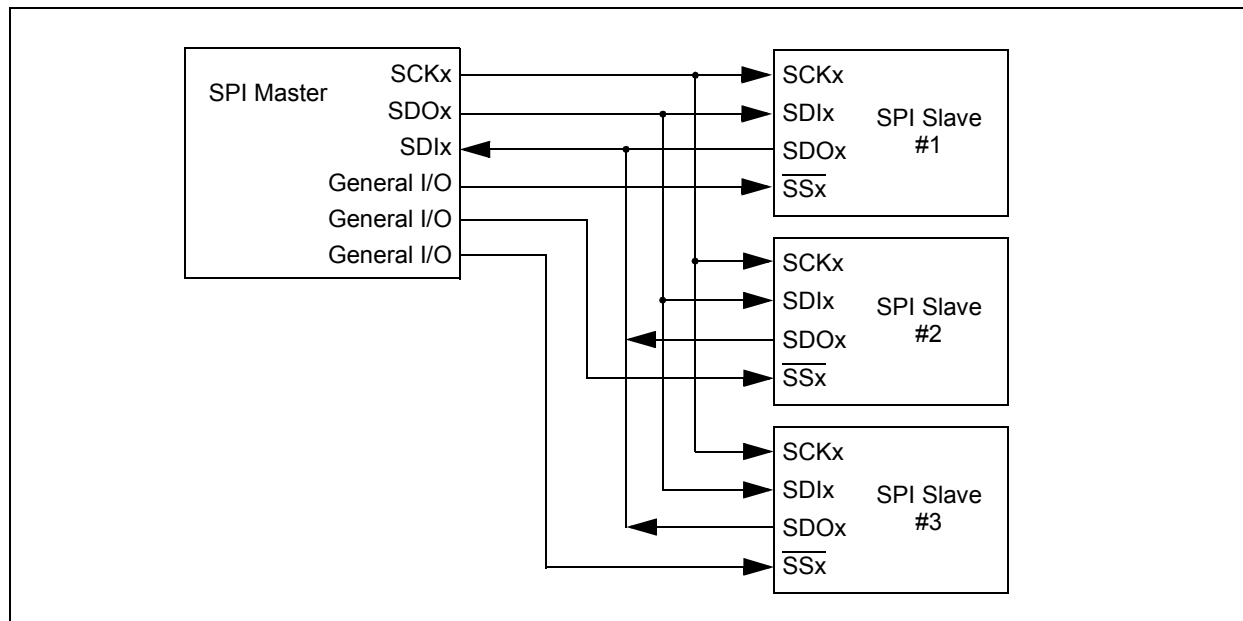
Note: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

- Timer1 enabled after POR
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = 0) when T1CKI is high then Timer1 is enabled (TMR1ON=1) when T1CKI is low.

TABLE 21-2: CLOCK SOURCE SELECTIONS

TMR1CS1	TMR1CS0	T1OSCEN	Clock Source
0	1	x	System Clock (Fosc)
0	0	x	Instruction Clock (Fosc/4)
1	1	x	Capacitive Sensing Oscillator
1	0	0	External Clocking on T1CKI Pin
1	0	1	Osc.Circuit On T1OSI/T1OSO Pins

FIGURE 24-4: SPI MASTER AND MULTIPLE SLAVE CONNECTION



24.2.1 SPI MODE REGISTERS

The MSSPx module has five registers for SPI mode operation. These are:

- MSSPx STATUS register (SSPxSTAT)
- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Control Register 3 (SSPxCON3)
- MSSPx Data Buffer register (SSPxBUF)
- MSSPx Address register (SSPxADD)
- MSSPx Shift register (SSPxSR)
(Not directly accessible)

SSPxCON1 and SSPxSTAT are the control and STATUS registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower six bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

In SPI master mode, SSPxADD can be loaded with a value used in the Baud Rate Generator. More information on the Baud Rate Generator is available in **Section 24.7 “Baud Rate Generator”**.

SSPxSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPxSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPxSR and SSPxBUF together create a buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

24.6.4 I²C MASTER MODE START CONDITION TIMING

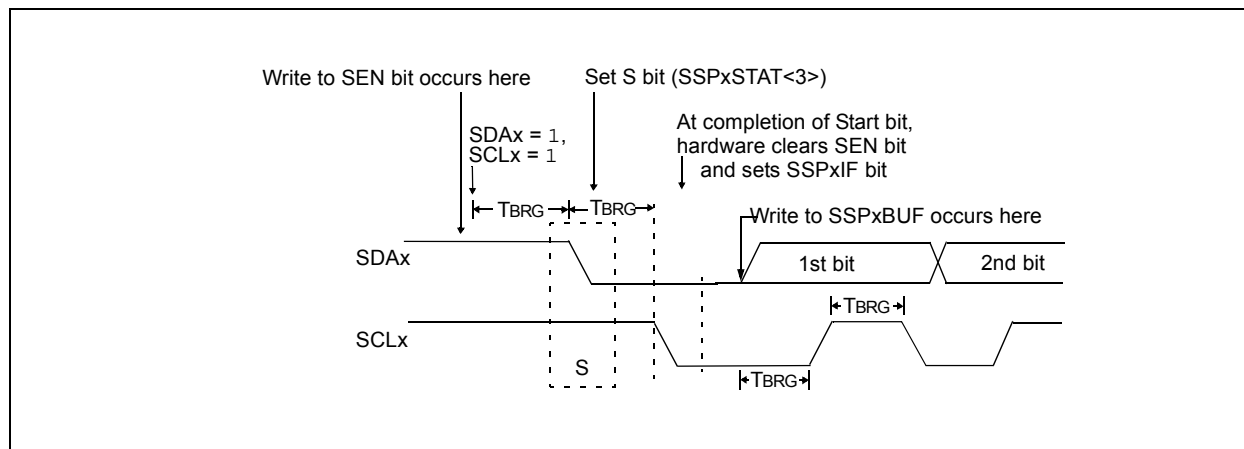
To initiate a Start condition, the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared

by hardware; the Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

Note 1: If, at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs. The Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

2: The Philips I²C Specification states that a bus collision cannot occur on a Start.

FIGURE 24-26: FIRST START BIT TIMING



26.2.2 CURRENT RANGES

The capacitive sensing oscillator can operate in one of seven different power modes. The power modes are separated into two ranges: the low range and the high range.

When the oscillator's low range is selected, the fixed internal voltage references of the capacitive sensing oscillator are being used. When the oscillator's high range is selected, the variable voltage references supplied by the FVR and DAC modules are being used. Selection between the voltage references is controlled by the CPSRM bit of the CPSCON0 register. See **Section 26.2.1 "Voltage Reference Modes"** for more information.

Within each range there are three distinct power modes: low, medium and high. Current consumption is dependent upon the range and mode selected. Selecting Power modes within each range is accomplished by configuring the CPSRNG <1:0> bits in the CPSCON0 register. See Table for proper power mode selection.

The remaining mode is a Noise Detection mode that resides within the high range. The Noise Detection mode is unique in that it disables the sinking and sourcing of current on the analog pin but leaves the rest of the oscillator circuitry active. This reduces the oscillation frequency on the analog pin to zero and also greatly reduces the current consumed by the oscillator module.

When noise is introduced onto the pin, the oscillator is driven at the frequency determined by the noise. This produces a detectable signal at the comparator output, indicating the presence of activity on the pin.

Figure 26-2 shows a more detailed drawing of the current sources and comparators associated with the oscillator.

TABLE 26-1: POWER MODE SELECTION

CPSRM	Range	CPSRNG<1:0>	Current Range ⁽¹⁾
1	High	00	Noise Detection
		01	Low
		10	Medium
		11	High
0	Low	00	Off
		01	Low
		10	Medium
		11	High

Note 1: See Power-Down Currents (IPD) in **Section 30.0 "Electrical Specifications"** for more information.

26.2.3 TIMER RESOURCES

To measure the change in frequency of the capacitive sensing oscillator, a fixed time base is required. For the period of the fixed time base, the capacitive sensing oscillator is used to clock either Timer0 or Timer1. The frequency of the capacitive sensing oscillator is equal to the number of counts in the timer divided by the period of the fixed time base.

26.2.4 FIXED TIME BASE

To measure the frequency of the capacitive sensing oscillator, a fixed time base is required. Any timer resource or software loop can be used to establish the fixed time base. It is up to the end user to determine the method in which the fixed time base is generated.

Note: The fixed time base can not be generated by the timer resource that the capacitive sensing oscillator is clocking.

26.2.4.1 Timer0

To select Timer0 as the timer resource for the CPS module:

- Set the T0XCS bit of the CPSCON0 register.
- Clear the TMR0CS bit of the OPTION_REG register.

When Timer0 is chosen as the timer resource, the capacitive sensing oscillator will be the clock source for Timer0. Refer to **Section 20.0 "Timer0 Module"** for additional information.

REGISTER 27-2: LCDPS: LCD PHASE REGISTER

R/W-0/0	R/W-0/0	R-0/0	R-0/0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-1/1
WFT	BIASMD	LCDA	WA	LP<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	C = Only clearable bit

bit 7	WFT: Waveform Type bit 1 = Type-B phase changes on each frame boundary 0 = Type-A phase changes within each common type
bit 6	BIASMD: Bias Mode Select bit When LMUX<1:0> = 00: 0 = Static Bias mode (do not set this bit to '1') When LMUX<1:0> = 01: 1 = 1/2 Bias mode 0 = 1/3 Bias mode When LMUX<1:0> = 10: 1 = 1/2 Bias mode 0 = 1/3 Bias mode When LMUX<1:0> = 11: 0 = 1/3 Bias mode (do not set this bit to '1')
bit 5	LCDA: LCD Active Status bit 1 = LCD driver module is active 0 = LCD driver module is inactive
bit 4	WA: LCD Write Allow Status bit 1 = Writing to the LCDDATAN registers is allowed 0 = Writing to the LCDDATAN registers is not allowed
bit 3-0	LP<3:0>: LCD Prescaler Selection bits 1111 = 1:16 1110 = 1:15 1101 = 1:14 1100 = 1:13 1011 = 1:12 1010 = 1:11 1001 = 1:10 1000 = 1:9 0111 = 1:8 0110 = 1:7 0101 = 1:6 0100 = 1:5 0011 = 1:4 0010 = 1:3 0001 = 1:2 0000 = 1:1

BCF	Bit Clear f
Syntax:	[<i>label</i>] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$0 \rightarrow (f)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

BTFSC	Bit Test f, Skip if Clear
Syntax:	[<i>label</i>] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if (f) = 0
Status Affected:	None
Description:	If bit 'b' in register 'f' is '1', the next instruction is executed. If bit 'b', in register 'f', is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction.

BRA	Relative Branch
Syntax:	[<i>label</i>] BRA label [<i>label</i>] BRA \$+k
Operands:	$-256 \leq \text{label} - \text{PC} + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	$(\text{PC}) + 1 + k \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the signed 9-bit literal 'k' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + k$. This instruction is a 2-cycle instruction. This branch has a limited range.

BTFSS	Bit Test f, Skip if Set
Syntax:	[<i>label</i>] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if (f) = 1
Status Affected:	None
Description:	If bit 'b' in register 'f' is '0', the next instruction is executed. If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

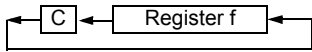
BRW	Relative Branch with W
Syntax:	[<i>label</i>] BRW
Operands:	None
Operation:	$(\text{PC}) + (W) \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + (W)$. This instruction is a 2-cycle instruction.

BSF	Bit Set f
Syntax:	[<i>label</i>] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$1 \rightarrow (f)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

RETFIE	Return from Interrupt
Syntax:	[<i>label</i>] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None
Description:	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.
Words:	1
Cycles:	2
<u>Example:</u>	RETFIE After Interrupt PC = TOS GIE = 1

RETLW	Return with literal in W
Syntax:	[<i>label</i>] RETLW <i>k</i>
Operands:	$0 \leq k \leq 255$
Operation:	$k \rightarrow (W)$; TOS → PC
Status Affected:	None
Description:	The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.
Words:	1
Cycles:	2
<u>Example:</u>	CALL TABLE;W contains table ;offset value • ;W now has table value • • ADDWF PC ;W = offset RETLW k1 ;Begin table RETLW k2 ; • • • RETLW kn ; End of table
TABLE	
	Before Instruction W = 0x07 After Instruction W = value of k8

RETURN	Return from Subroutine
Syntax:	[<i>label</i>] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RLF	Rotate Left f through Carry
Syntax:	[<i>label</i>] RLF <i>f</i> , <i>d</i>
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Description:	The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is stored back in register 'f'.
	
Words:	1
Cycles:	1
<u>Example:</u>	RLF REG1,0 Before Instruction REG1 = 1110 0110 C = 0 After Instruction REG1 = 1110 0110 W = 1100 1100 C = 1

PIC16(L)F1946/47

30.3 DC Characteristics: PIC16(L)F1946/47-I/E (Industrial, Extended)

PIC16LF1946/47			Standard Operating Conditions (unless otherwise stated)				
			Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
PIC16F1946/47			Standard Operating Conditions (unless otherwise stated)				
			Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	Device Characteristics	Min.	Typ†	Max.	Units	Conditions	
						VDD	Note
	Supply Current (IDD) ^(1, 2)						
D009	LDO Regulator	—	350	—	μA	—	HS, EC OR HFINTOSC Clock modes with VCAP pin disabled
		—	30	—	μA	—	
		—	5	—	μA	—	LP/LFINTOSC Clock mode or Sleep (requires FVR and BOR to be disabled)
D010		—	5.0	11	μA	1.8	Fosc = 32 kHz
		—	6.0	13	μA	3.0	LP Oscillator mode (Note 4), -40°C ≤ TA ≤ +85°C
D010		—	24	53	μA	1.8	Fosc = 32 kHz
		—	30	58	μA	3.0	LP Oscillator mode (Note 4, 5), -40°C ≤ TA ≤ +85°C
		—	32	63	μA	5.0	
D010A		—	7.0	23	μA	1.8	Fosc = 32 kHz
		—	9.0	27	μA	3.0	LP Oscillator mode (Note 4) -40°C ≤ TA ≤ +125°C
D010A		—	24	68	μA	1.8	Fosc = 32 kHz
		—	30	88	μA	3.0	LP Oscillator mode (Note 4, 5) -40°C ≤ TA ≤ +125°C
		—	32	95	μA	5.0	
D011		—	60	105	μA	1.8	Fosc = 1 MHz
		—	120	190	μA	3.0	XT Oscillator mode
D011		—	95	130	μA	1.8	Fosc = 1 MHz
		—	170	220	μA	3.0	XT Oscillator mode (Note 5)
		—	190	270	μA	5.0	
D012		—	160	300	μA	1.8	Fosc = 4 MHz
		—	300	500	μA	3.0	XT Oscillator mode
D012		—	200	330	μA	1.8	Fosc = 4 MHz
		—	300	500	μA	3.0	XT Oscillator mode (Note 5)
		—	400	650	μA	5.0	

- Note 1:** The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; MCLR = VDD; WDT disabled.
- Note 2:** The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.
- Note 3:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2 \text{ REXT}$ (mA) with REXT in kΩ.
- Note 4:** FVR and BOR are disabled.
- Note 5:** 0.1 μF capacitor on VCAP (RF0).
- Note 6:** 8 MHz crystal oscillator with 4x PLL enabled.

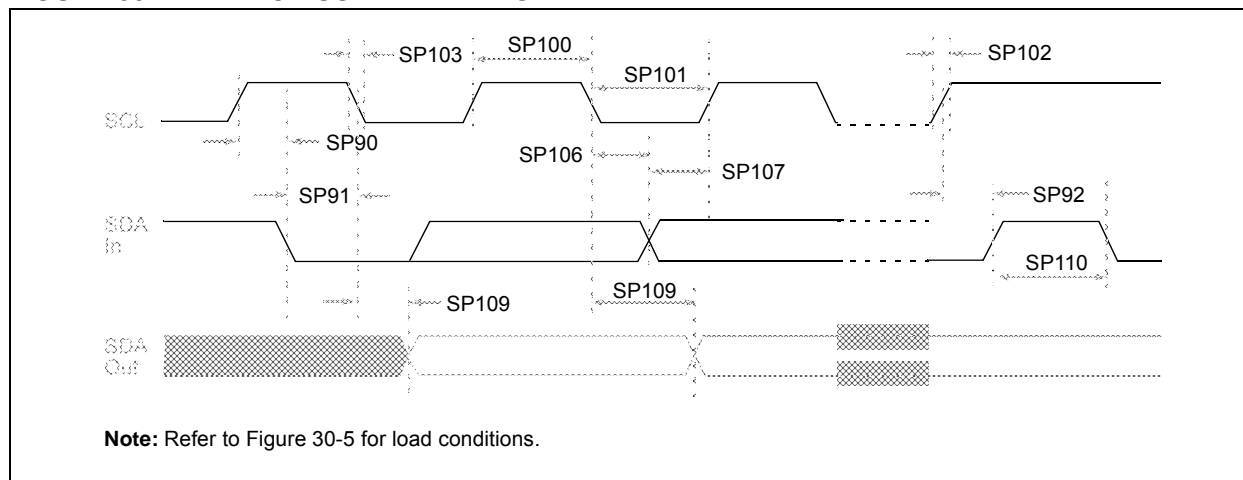
PIC16(L)F1946/47

TABLE 30-15: I²C BUS START/STOP BITS REQUIREMENTS

Param No.	Symbol	Characteristic		Min.	Typ	Max.	Units	Conditions
SP90*	TSU:STA	Start condition	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
		Setup time	400 kHz mode	600	—	—		
SP91*	THD:STA	Start condition	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
		Hold time	400 kHz mode	600	—	—		
SP92*	TSU:STO	Stop condition	100 kHz mode	4700	—	—	ns	
		Setup time	400 kHz mode	600	—	—		
SP93	THD:STO	Stop condition	100 kHz mode	4000	—	—	ns	
		Hold time	400 kHz mode	600	—	—		

* These parameters are characterized but not tested.

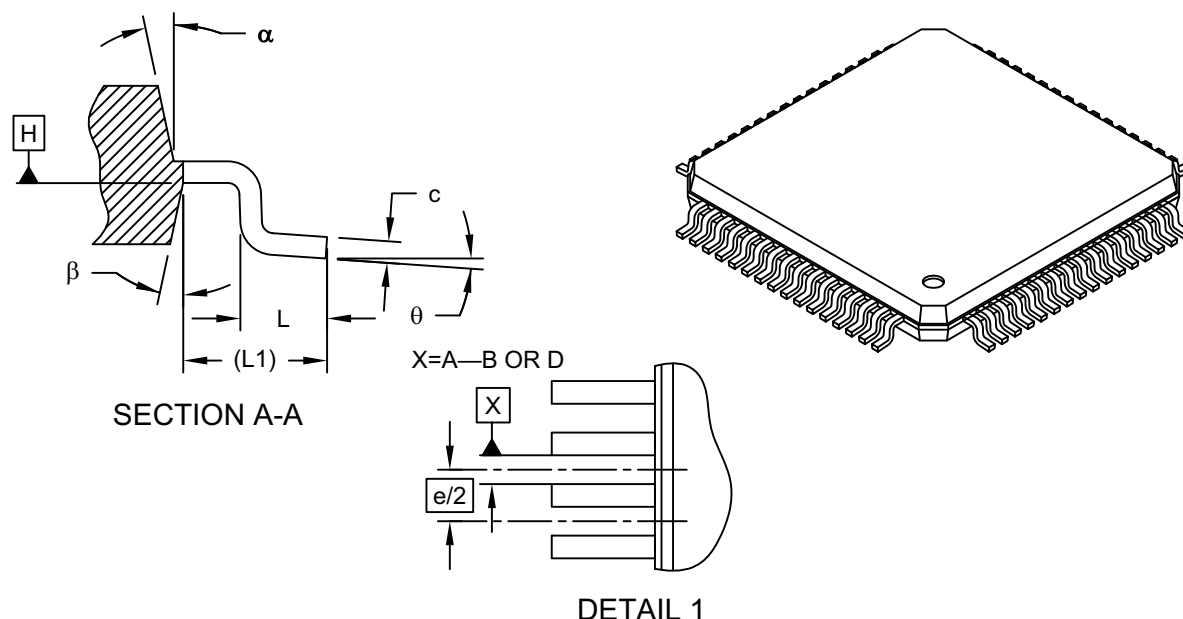
FIGURE 30-21: I²C BUS DATA TIMING



PIC16(L)F1946/47

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

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



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N	64		
Lead Pitch	e	0.50 BSC		
Overall Height	A	-	-	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	-	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1	1.00 REF		
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E	12.00 BSC		
Overall Length	D	12.00 BSC		
Molded Package Width	E1	10.00 BSC		
Molded Package Length	D1	10.00 BSC		
Lead Thickness	c	0.09	-	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Chamfers at corners are optional; size may vary.
- Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
- Dimensioning and tolerancing per ASME Y14.5M

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

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

Microchip Technology Drawing C04-085C Sheet 2 of 2