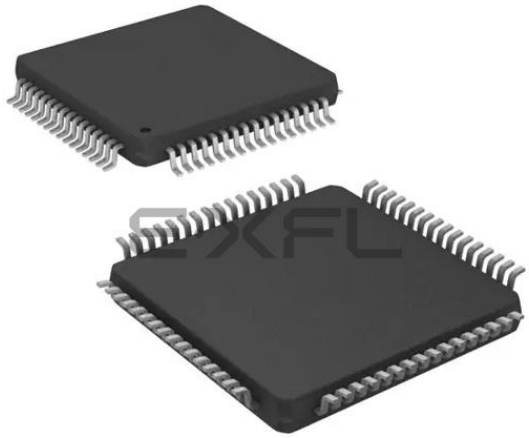


**Welcome to E-XFL.COM**

## 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 | 20MHz |
| Connectivity | I²C, SPI |
| Peripherals | Brown-out Detect/Reset, LCD, POR, PWM, WDT |
| Number of I/O | 25 |
| Program Memory Size | 7KB (4K x 14) |
| Program Memory Type | OTP |
| EEPROM Size | - |
| RAM Size | 176 x 8 |
| Voltage - Supply (Vcc/Vdd) | 4V ~ 5.5V |
| Data Converters | A/D 5x10b |
| Oscillator Type | External |
| Operating Temperature | -40°C ~ 85°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/pic16c925-i-pt |

**TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Details on page |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 1** | | | | | | | | | | | |
| 80h | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 | 26 |
| 81h | OPTION | RBPU | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 20 |
| 82h | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 25 |
| 83h | STATUS | IRP | RP1 | RP0 | TO | PD | Z | DC | C | 0001 1xxx | 19 |
| 84h | FSR | Indirect Data Memory Address Pointer | | | | | | | | xxxx xxxx | 26 |
| 85h | TRISA | — | — | PORTA Data Direction Register | | | | | | --11 1111 | 29 |
| 86h | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 31 |
| 87h | TRISC | — | — | PORTC Data Direction Register | | | | | | --11 1111 | 33 |
| 88h | TRISD | PORTD Data Direction Register | | | | | | | | 1111 1111 | 34 |
| 89h | TRISE | PORTE Data Direction Register | | | | | | | | 1111 1111 | 36 |
| 8Ah | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the PC | | | | | ---0 0000 | 25 |
| 8Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 21 |
| 8Ch | PIE1 | LCDIE | ADIE | — | — | SSPIE | CCP1IE | TMR2IE | TMR1IE | 00-- 0000 | 24 |
| 8Dh | — | Unimplemented | | | | | | | | — | — |
| 8Eh | PCON | — | — | — | — | — | — | POR | $\overline{BOR}$ | ---- --0- | 24 |
| 8Fh | — | Unimplemented | | | | | | | | — | — |
| 90h | — | Unimplemented | | | | | | | | — | — |
| 91h | — | Unimplemented | | | | | | | | — | — |
| 92h | PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 51 |
| 93h | SSPADD | Synchronous Serial Port ($I^2C$ mode) Address Register | | | | | | | | 0000 0000 | 69, 72 |
| 94h | SSPSTAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 59 |
| 95h | — | Unimplemented | | | | | | | | — | — |
| 96h | — | Unimplemented | | | | | | | | — | — |
| 97h | — | Unimplemented | | | | | | | | — | — |
| 98h | — | Unimplemented | | | | | | | | — | — |
| 99h | — | Unimplemented | | | | | | | | — | — |
| 9Ah | — | Unimplemented | | | | | | | | — | — |
| 9Bh | — | Unimplemented | | | | | | | | — | — |
| 9Ch | — | Unimplemented | | | | | | | | — | — |
| 9Dh | — | Unimplemented | | | | | | | | — | — |
| 9Eh | ADRESL | A/D Result Register Low | | | | | | | | xxxx xxxx | 79 |
| 9Fh | ADCON1 | — | — | — | — | — | PCFG2 | PCFG1 | PCFG0 | ---- -000 | 76 |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0'.
Shaded locations are unimplemented, read as '0'.

**Note 1:** These pixels do not display, but can be used as general purpose RAM.

## 4.0 I/O PORTS

Some pins for these ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

### 4.1 PORTA and TRISA Register

The RA4/T0CKI pin is a Schmitt Trigger input and an open drain output. All other RA port pins have TTL input levels and full CMOS output drivers. All RA pins have data direction bits (TRISA register), which can configure these pins as output or input.

Setting a bit in the TRISA register puts the corresponding output driver in a Hi-Impedance mode. Clearing a bit in the TRISA register puts the contents of the output latch on the selected pin.

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified, and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The other PORTA pins are multiplexed with analog inputs and the analog VREF input. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register1).

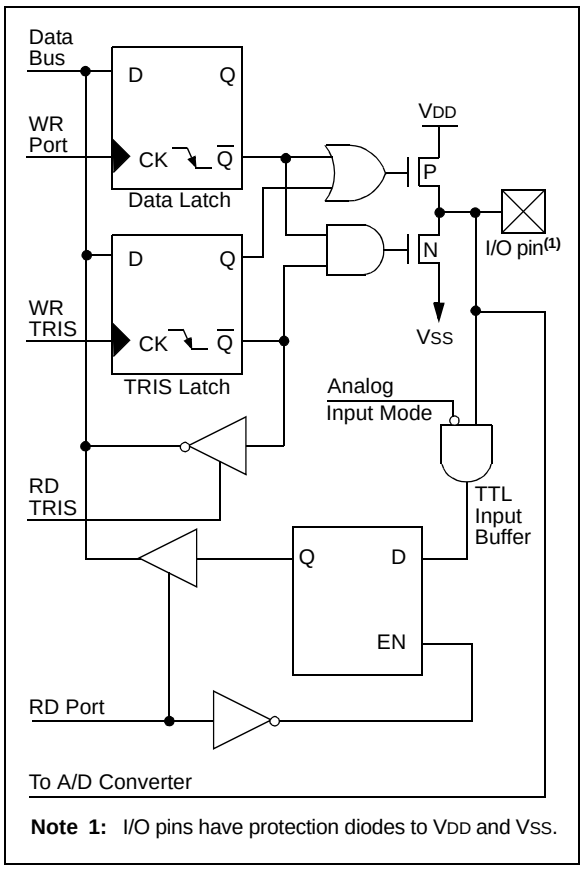
| **Note:** | On a Power-on Reset, these pins are configured as analog inputs and read as '0'. |
|---|---|

The TRISA register controls the direction of the RA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

**EXAMPLE 4-1: INITIALIZING PORTA**

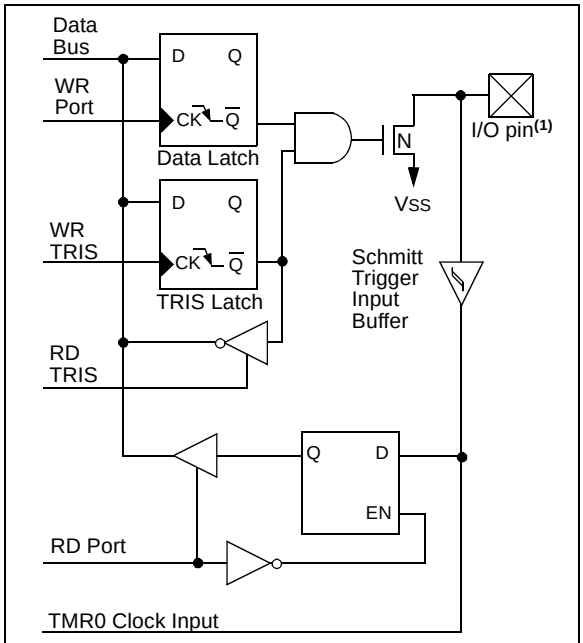
```
BCF     STATUS, RP0   ; Select Bank0
BCF     STATUS, RP1
CLRF    PORTA         ; Initialize PORTA
BSF     STATUS, RP0   ; Select Bank1
MOVLW   0xCF          ; Value used to
                      ; initialize data
                      ; direction
MOVWF   TRISA         ; Set RA<3:0> as inputs
                      ; RA<5:4> as outputs
                      ; RA<7:6> are always
                      ; read as '0'.
```

**FIGURE 4-1: BLOCK DIAGRAM OF PINS RA3:RA0 AND RA5**




**Note 1:** I/O pins have protection diodes to VDD and VSS.

**FIGURE 4-2: BLOCK DIAGRAM OF RA4/T0CKI PIN**




**Note 1:** I/O pin has protection diodes to VSS only.

## 4.6 PORTF and TRISF Register

PORTF is a digital input only port. Each pin is multiplexed with an LCD segment driver. These pins have Schmitt Trigger input buffers.

**Note 1:** On a Power-on Reset, these pins are configured as LCD segment drivers.

**2:** To configure the pins as a digital port, the corresponding bits in the LCDSE register must be cleared. Any bit set in the LCDSE register overrides any bit settings in the corresponding TRIS register.

**EXAMPLE 4-6: INITIALIZING PORTF**

```
BCF STATUS,  RP0    ;Select Bank2
BSF STATUS,  RP1    ;
BCF LCDSE,   SE16   ;Make all PORTF
BCF LCDSE,   SE12   ;digital inputs
```

**FIGURE 4-8: PORTF BLOCK DIAGRAM**

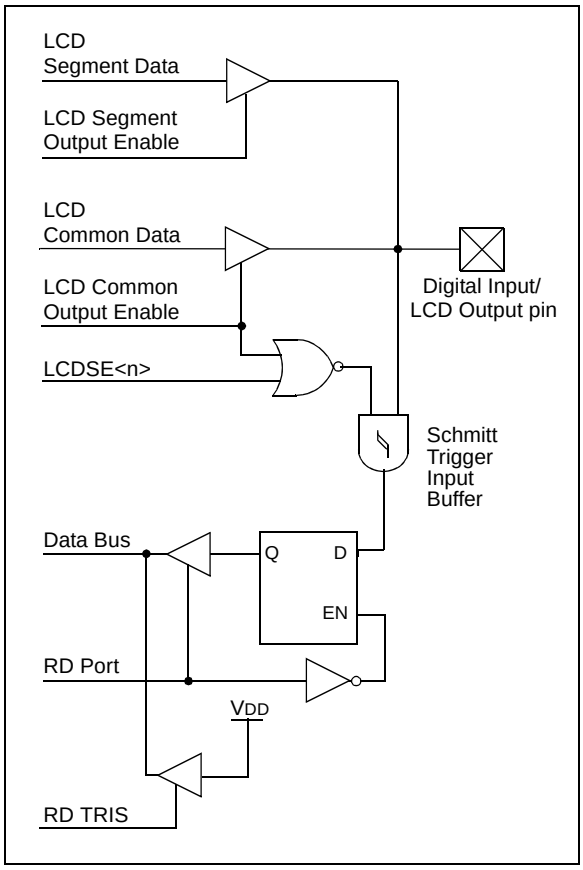



**TABLE 4-11: PORTF FUNCTIONS**

| Name | Bit# | Buffer Type | Function |
|---|---|---|---|
| RF0/SEG12 | bit0 | ST | Digital input or Segment Driver12. |
| RF1/SEG13 | bit1 | ST | Digital input or Segment Driver13. |
| RF2/SEG14 | bit2 | ST | Digital input or Segment Driver14. |
| RF3/SEG15 | bit3 | ST | Digital input or Segment Driver15. |
| RF4/SEG16 | bit4 | ST | Digital input or Segment Driver16. |
| RF5/SEG17 | bit5 | ST | Digital input or Segment Driver17. |
| RF6/SEG18 | bit6 | ST | Digital input or Segment Driver18. |
| RF7/SEG19 | bit7 | ST | Digital input or Segment Driver19. |

Legend: ST = Schmitt Trigger input

**TABLE 4-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 107h | PORTF | RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 | 0000 0000 | 0000 0000 |
| 187h | TRISF | PORTF Data Direction Control Register | | | | | | | | 1111 1111 | 1111 1111 |
| 10Dh | LCDSE | SE29 | SE27 | SE20 | SE16 | SE12 | SE9 | SE5 | SE0 | 1111 1111 | 1111 1111 |

Legend: Shaded cells are not used by PORTF.

## 4.8 I/O Programming Considerations

### 4.8.1 BI-DIRECTIONAL I/O PORTS

Any instruction which writes, operates internally as a read followed by a write operation. The BCF and BSF instructions, for example, read the register into the CPU, execute the bit operation and write the result back to the register. Caution must be used when these instructions are applied to a port with both inputs and outputs defined. For example, a BSF operation on bit5 of PORTB will cause all eight bits of PORTB to be read into the CPU. Then the BSF operation takes place on bit5 and PORTB is written to the output latches. If another bit of PORTB is used as a bi-directional I/O pin (e.g., bit0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and rewritten to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the input mode, no problem occurs. However, if bit0 is switched into output mode later on, the contents of the data latch may now be unknown.

Reading the port register reads the values of the port pins. Writing to the port register, writes the value to the port latch. When using read-modify-write instructions (e.g. BCF, BSF) on a port, the value of the port pins is read, the desired operation is done to this value, and this value is then written to the port latch.

Example 4-8 shows the effect of two sequential read-modify-write instructions on an I/O port. A pin actively outputting a Low or High should not be driven from external devices at the same time, in order to change the level on this pin ("wired-or", "wired-and"). The resulting high output currents may damage the chip.

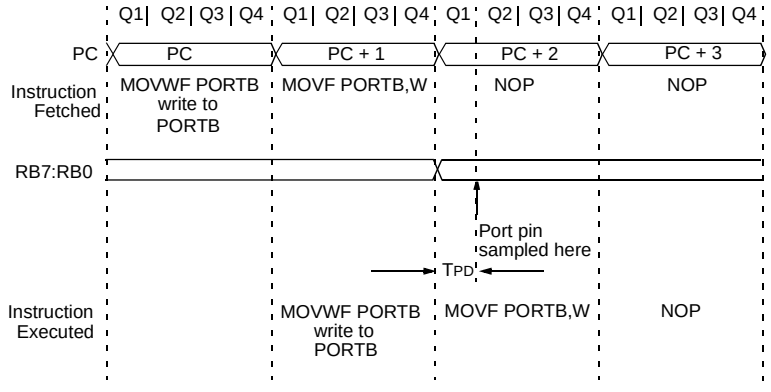
**EXAMPLE 4-8: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT**

```
;Initial PORT settings: PORTB<7:4> Inputs
;                       PORTB<3:0> Outputs
;PORTB<7:6> have external pull-ups and are
;not connected to other circuitry
;
;                    PORT latch   PORT pins
;                    ----------   ---------
  BCF PORTB,  7   ; 01pp pppp    11pp pppp
  BCF PORTB,  6   ; 10pp pppp    11pp pppp
  BCF STATUS, RP1 ; Select Bank1
  BSF STATUS, RP0 ;
  BCF TRISB,  7   ; 10pp pppp    11pp pppp
  BCF TRISB,  6   ; 10pp pppp    10pp pppp
;
;Note that the user may have expected the
;pin values to be 00pp ppp. The 2nd BCF
;caused RB7 to be latched as the pin value
;(high).
```

### 4.8.2 SUCCESSIVE OPERATIONS ON I/O PORTS

The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle (Figure 4-10). Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such to allow the pin voltage to stabilize (load dependent) before the next instruction, which causes that file to be read into the CPU, is executed. Otherwise, the previous state of that pin may be read into the CPU, rather than the new state. When in doubt, it is better to separate these instructions with a NOP, or another instruction not accessing this I/O port.

**FIGURE 4-10: SUCCESSIVE I/O OPERATION**




**Note:**

This example shows a write to PORTB followed by a read from PORTB.

Note that:

data setup time = (0.25TCY - TPD)

where TCY = instruction cycle
TPD = propagation delay

Therefore, at higher clock frequencies, a write followed by a read may be problematic.

## 6.4 Timer1 Oscillator

A crystal oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit T1OSCEN (T1CON<3>). The oscillator is a low power oscillator rated up to 200 kHz. It will continue to run during SLEEP. It is primarily intended for a 32 kHz crystal. Table 6-1 shows the capacitor selection for the Timer1 oscillator.

The Timer1 oscillator is identical to the LP oscillator. The user must provide a software time delay to ensure proper oscillator start-up.

**TABLE 6-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR**

| Osc Type | Freq | C1 | C2 |
|---|---|---|---|
| LP | 32 kHz | 33 pF | 33 pF |
| | 100 kHz | 15 pF | 15 pF |
| | 200 kHz | 15 pF | 15 pF |
| **These values are for design guidance only.** | | | |
| **Crystals Tested:** | | | |
| 32.768 kHz | Epson C-001R32.768K-A | | ± 20 PPM |
| 100 kHz | Epson C-2 100.00 KC-P | | ± 20 PPM |
| 200 kHz | STD XTL 200.000 kHz | | ± 20 PPM |

**Note 1:** Higher capacitance increases the stability of the oscillator but also increases the start-up time.

**2:** Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

## 6.5 Resetting Timer1 Using the CCP Trigger Output

If the CCP1 module is configured in Compare mode to generate a “special event trigger” (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1.

**Note:** The special event trigger from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode, to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L registers pair effectively become the period register for Timer1.

## 6.6 Resetting of Timer1 Register Pair (TMR1H:TMR1L)

TMR1H and TMR1L registers are not reset on a POR or any other RESET, except by the CCP1 special event trigger.

T1CON register is reset to 00h on a Power-on Reset. In any other RESET, the register is unaffected.

## 6.7 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

**TABLE 6-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh, 8Bh, 10Bh, 18Bh | INTCON | GIE | PEIE | TMR0IE | INTE | RBIE | TMR0IF | INTF | RBIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | LCDIF | ADIF | — | — | SSPIF | CCP1IF | TMR2IF | TMR1IF | 00-- 0000 | 00-- 0000 |
| 8Ch | PIE1 | LCDIE | ADIE | — | — | SSPIE | CCP1IE | TMR2IE | TMR1IE | 00-- 0000 | 00-- 0000 |
| 0Eh | TMR1L | Holding register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Fh | TMR1H | Holding register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10h | T1CON | — | — | T1CKPS1 | T1CKPS0 | T1OSCEN | T1SYNC | TMR1CS | TMR1ON | --00 0000 | --uu uuuu |

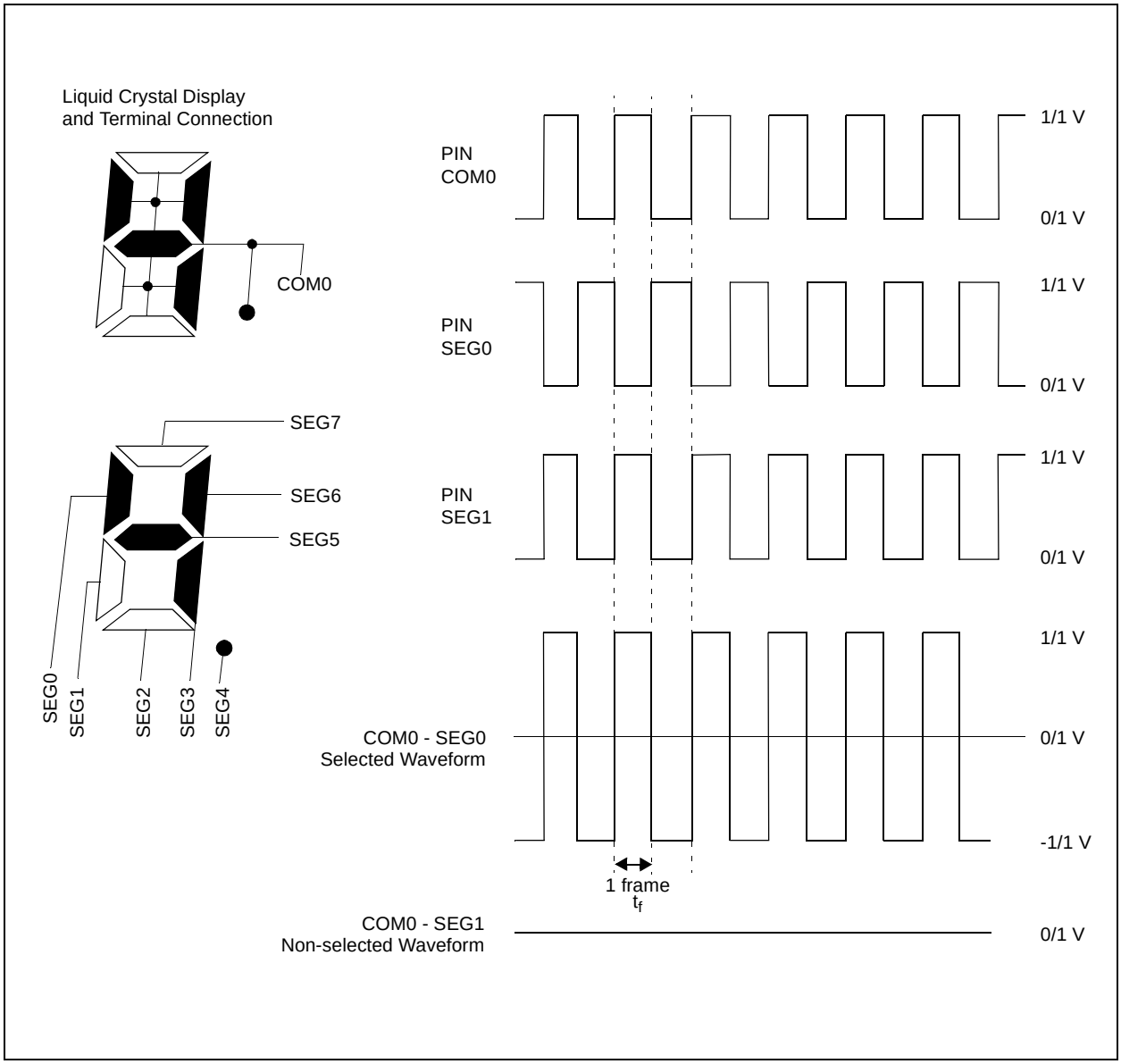
Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by theTimer1 module.

**NOTES:**

**FIGURE 11-2: WAVEFORMS IN STATIC DRIVE**

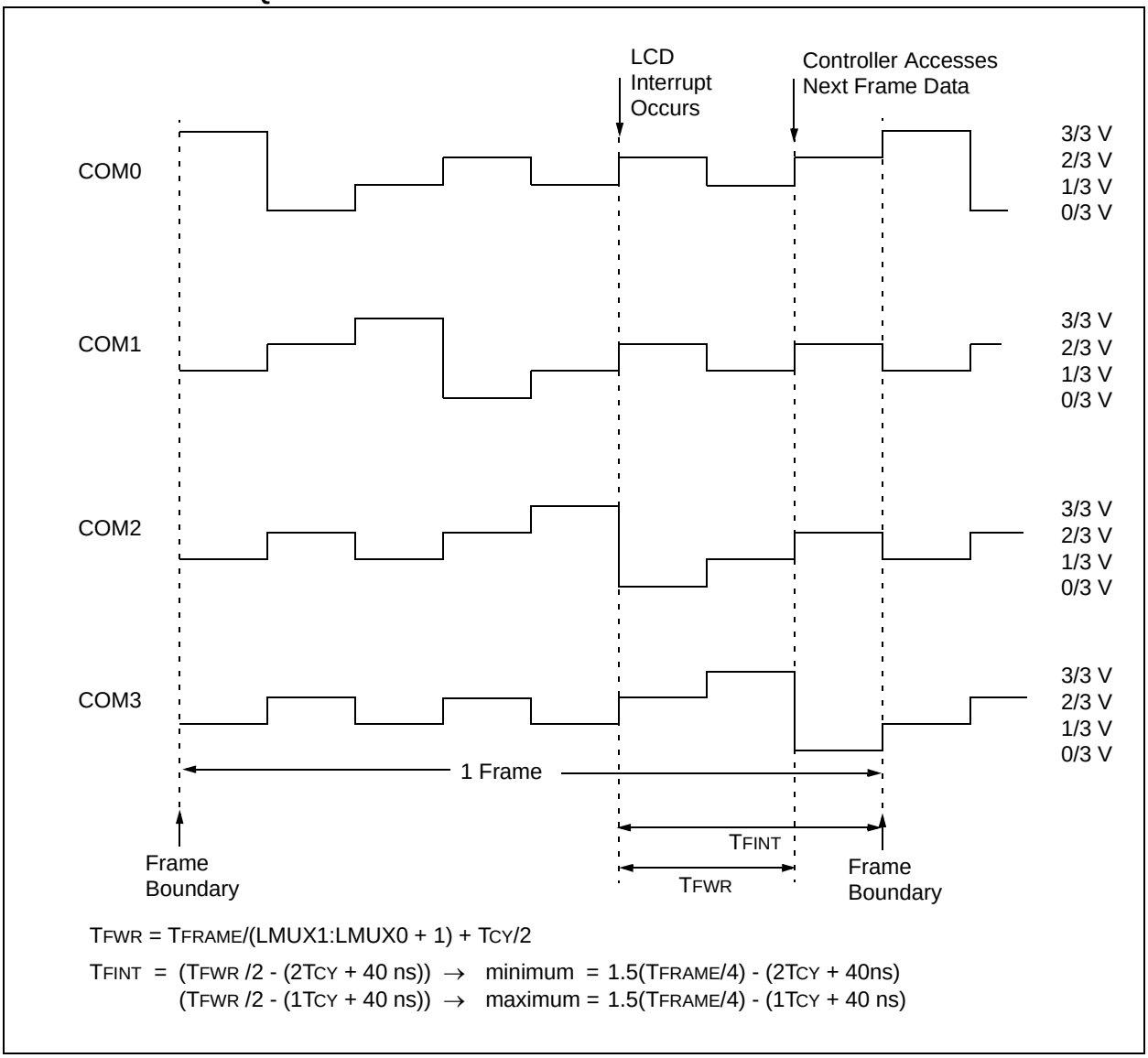


## 11.2 LCD Interrupts

The LCD timing generation provides an interrupt that defines the LCD frame timing. This interrupt can be used to coordinate the writing of the pixel data with the start of a new frame. Writing pixel data at the frame boundary allows a visually crisp transition of the image. This interrupt can also be used to synchronize external events to the LCD. For example, the interface to an external segment driver, such as a Microchip AY0438, can be synchronized for segment data update to the LCD frame.

A new frame is defined to begin at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary ($T_{FINT}$), as shown in Figure 11-7. The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins to access data after the interrupt ($T_{FWR}$). New data must be written within $T_{FWR}$, as this is when the LCD controller will begin to access the data for the next frame.

**FIGURE 11-7: EXAMPLE WAVEFORMS AND INTERRUPT TIMING IN QUARTER-DUTY CYCLE DRIVE**




$T_{FWR} = T_{FRAME}/(LMUX1:LMUX0 + 1) + T_{CY}/2$

$T_{FINT} = (T_{FWR}/2 - (2T_{CY} + 40\ ns)) \rightarrow$ minimum $= 1.5(T_{FRAME}/4) - (2T_{CY} + 40ns)$

$(T_{FWR}/2 - (1T_{CY} + 40\ ns)) \rightarrow$ maximum $= 1.5(T_{FRAME}/4) - (1T_{CY} + 40\ ns)$

**TABLE 12-6: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

| Register | Power-on Reset | $\overline{\text{MCLR}}$ Resets WDT Reset | Wake-up via WDT or Interrupt |
|---|---|---|---|
| W | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF | N/A | N/A | N/A |
| TMR0 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCL | 0000h | 0000h | PC + 1(2) |
| STATUS | 0001 1xxx | 000q quuu(3) | uuuq quuu(3) |
| FSR | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTA | --0x 0000 | --0u 0000 | --uu uuuu |
| PORTB | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTC | --xx xxxx | --uu uuuu | --uu uuuu |
| PORTD | 0000 0000 | 0000 0000 | uuuu uuuu |
| PORTE | 0000 0000 | 0000 0000 | uuuu uuuu |
| PCLATH | ---0 0000 | ---0 0000 | ---u uuuu |
| INTCON | 0000 000x | 0000 000u | uuuu uuuu(1) |
| PIR1 | 00-- 0000 | 00-- 0000 | uu-- uuuu(1) |
| TMR1L | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR1H | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T1CON | --00 0000 | --uu uuuu | --uu uuuu |
| TMR2 | 0000 0000 | 0000 0000 | uuuu uuuu |
| T2CON | -000 0000 | -000 0000 | -uuu uuuu |
| SSPBUF | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| SSPCON | 0000 0000 | 0000 0000 | uuuu uuuu |
| CCPR1L | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCPR1H | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| CCP1CON | --00 0000 | --00 0000 | --uu uuuu |
| ADRES | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| ADCON0 | 0000 00-0 | 0000 00-0 | uuuu uu-u |
| OPTION_REG | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA | --11 1111 | --11 1111 | --uu uuuu |
| TRISB | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISC | --11 1111 | --11 1111 | --uu uuuu |
| TRISD | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISE | 1111 1111 | 1111 1111 | uuuu uuuu |
| PIE1 | 00-- 0000 | 00-- 0000 | uu-- uuuu |
| PCON | ---- --0- | ---- --u- | ---- --u- |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

**Note 1:** One or more bits in INTCON and/or PIR1 will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

**3:** See Table 12-5 for RESET value for specific condition.

## 12.5 Interrupts

The PIC16C925/926 family has nine sources of interrupt:

- External interrupt RB0/INT
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB7:RB4)
- A/D Interrupt
- TMR1 overflow interrupt
- TMR2 matches period interrupt
- CCP1 interrupt
- Synchronous serial port interrupt
- LCD module interrupt

The interrupt control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

| **Note:** | Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit, or the GIE bit. |
|---|---|

A global interrupt enable bit, GIE (INTCON<7>), enables (if set) all unmasked interrupts, or disables (if cleared) all interrupts. When bit GIE is enabled, and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt bits are set, regardless of the status of the GIE bit. The GIE bit is cleared on RESET.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine as well as sets the GIE bit, which re-enables interrupts.

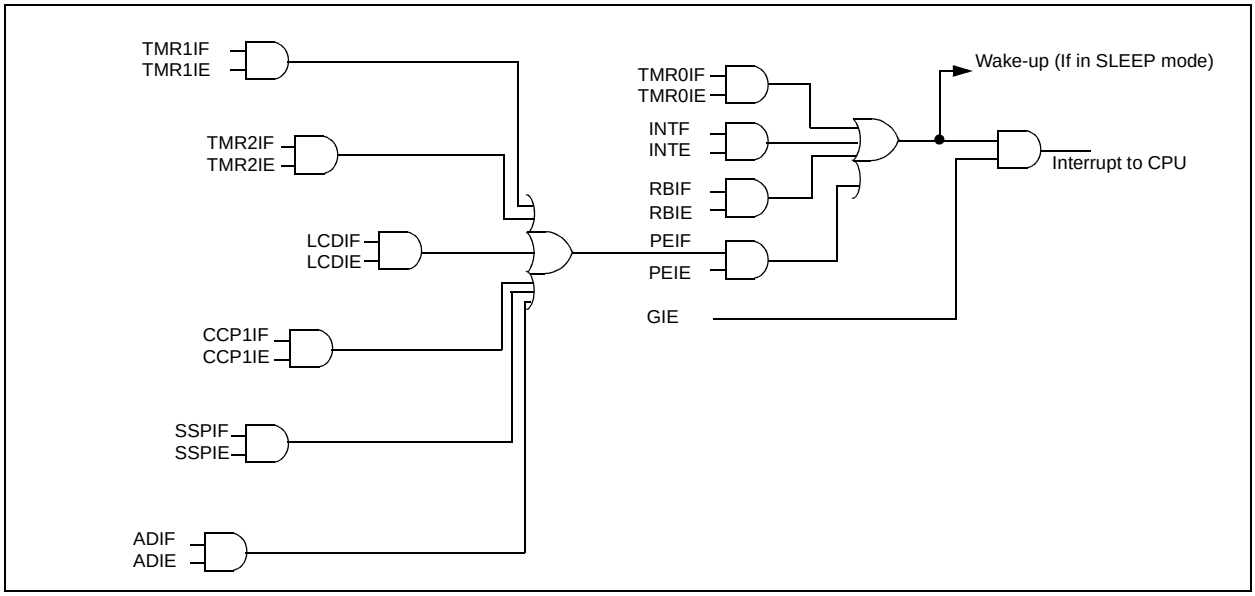
The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flags are contained in the special function register, PIR1. The corresponding interrupt enable bits are contained in special function register, PIE1, and the peripheral interrupt enable bit is contained in special function register, INTCON.

When an interrupt is responded to, the GIE bit is cleared to disable any further interrupts, the return address is pushed onto the stack and the PC is loaded with 0004h. Once in the Interrupt Service Routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

For external interrupt events, such as the RB0/INT pin or RB Port change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 12-11). The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit, PEIE bit, or the GIE bit.

**FIGURE 12-10: INTERRUPT LOGIC**



## 12.8 Power-down Mode (SLEEP)

Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{PD}$ bit (STATUS<3>) is cleared, the $\overline{TO}$ (STATUS<4>) bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had, before the SLEEP instruction was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, place all I/O pins at either $V_{DD}$, or $V_{SS}$, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D, disable external clocks. Pull all I/O pins that are hi-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input should also be at $V_{DD}$ or $V_{SS}$ for lowest current consumption. The contribution from on-chip pull-ups on PORTB should also be considered.

The $\overline{MCLR}$ pin must be at a logic high level ($V_{IHMC}$).

### 12.8.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

1. External RESET input on $\overline{MCLR}$ pin.
2. Watchdog Timer Wake-up (if WDT was enabled).
3. Interrupt from RB0/INT pin, RB port change, or peripheral interrupt.

External $\overline{MCLR}$ Reset will cause a device RESET. All other events are considered a continuation of program execution and cause a “wake-up”. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of device RESET. The $\overline{PD}$ bit, which is set on power-up is cleared when SLEEP is invoked. The $\overline{TO}$ bit is cleared if a WDT time-out occurred (and caused wake-up).

The following peripheral interrupts can wake the device from SLEEP:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. SSP (START/STOP) bit detect interrupt.
3. SSP transmit or receive in Slave mode (SPI/$I^2C$).
4. CCP Capture mode interrupt.
5. A/D conversion (when A/D clock source is RC).
6. Special event trigger (Timer1 in Asynchronous mode using an external clock).
7. LCD module.

Other peripherals can not generate interrupts since during SLEEP, no on-chip Q clocks are present.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

### 12.8.2 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT postscaler will not be cleared, the $\overline{TO}$ bit will not be set and $\overline{PD}$ bits will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake-up from SLEEP. The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

**ANDLW** **AND Literal with W**

Syntax: [ *label* ] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. (k) → (W)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 11 | 1001 | kkkk | kkkk |

Description: The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process data | Write to W |

Example `ANDLW 0x5F`

Before Instruction:
W = 0xA3
After Instruction:
W = 0x03

**ANDWF** **AND W with f**

Syntax: [ *label* ] ANDWF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W).AND. (f) → (destination)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0101 | dfff | ffff |

Description: AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | Write to destination |

Example `ANDWF FSR, 1`

Before Instruction:
W = 0x17
FSR = 0xC2
After Instruction
W = 0x17
FSR = 0x02

**BCF** **Bit Clear f**

Syntax: [ *label* ] BCF f [,b]

Operands: $0 \le f \le 127$
$0 \le b \le 7$

Operation: $0 \rightarrow$ (f<b>)

Status Affected: None

Encoding:

| 01 | 00bb | bfff | ffff |
|---|---|---|---|

Description: Bit 'b' in register 'f' is cleared.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | Write register 'f' |

Example
```
BCF     FLAG_REG, 7
```
Before Instruction:
FLAG_REG = 0xC7
After Instruction:
FLAG_REG = 0x47

**BSF** **Bit Set f**

Syntax: [ *label* ] BSF f [,b]

Operands: $0 \le f \le 127$
$0 \le b \le 7$

Operation: $1 \rightarrow$ (f<b>)

Status Affected: None

Encoding:

| 01 | 01bb | bfff | ffff |
|---|---|---|---|

Description: Bit 'b' in register 'f' is set.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | Write register 'f' |

Example
```
BSF     FLAG_REG,   7
```
Before Instruction:
FLAG_REG = 0x0A
After Instruction:
FLAG_REG = 0x8A

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

Syntax: [ *label* ] BTFSC f [,b]

Operands: $0 \le f \le 127$
$0 \le b \le 7$

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

Status Affected: None

Encoding:

| 01 | 10bb | bfff | ffff |
|---|---|---|---|

Description: If bit 'b' in register 'f' is '1', then the next instruction is executed.
If bit 'b' in register 'f' is '0', then the next instruction is discarded, and a NOP is executed instead, making this a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | No Operation |

If Skip: (2nd Cycle)

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No Operation | No Operation | No Operation | No Operation |

Example
```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```
Before Instruction:
PC = address HERE
After Instruction:
if FLAG<1> = 0,
PC = address TRUE
if FLAG<1> = 1,
PC = address FALSE

**GOTO** **Unconditional Branch**

Syntax: [ *label* ] GOTO k

Operands: $0 \le k \le 2047$

Operation: $k \rightarrow PC<10:0>$
$PCLATH<4:3> \rightarrow PC<12:11>$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 10 | 1kkk | kkkk | kkkk |

Description: GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

| | Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|---|
| 1st Cycle | Decode | Read literal 'k' | Process data | Write to PC |
| 2nd Cycle | No Operation | No Operation | No Operation | No Operation |

Example GOTO THERE

After Instruction:
PC = Address THERE

**INCF** **Increment f**

Syntax: [ *label* ] INCF f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) + 1 → (destination)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1010 | dfff | ffff |

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | Write to destination |

Example INCF CNT, 1

Before Instruction:
CNT = 0xFF
Z = 0

After Instruction:
CNT = 0x00
Z = 1

**INCFSZ** **Increment f, Skip if 0**

Syntax: [ *label* ] INCFSZ f [,d]

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) + 1 → (destination), skip if result = 0

Status Affected: None

Encoding:

| 00 | 1111 | dfff | ffff |
|---|---|---|---|

Description: The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2TCY instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process data | Write to destination |

If Skip: (2nd Cycle)

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No Operation | No Operation | No Operation | No Operation |

Example

```
HERE    INCFSZ CNT,   1
        GOTO   LOOP
CONTINUE •
•
•
```

Before Instruction:
PC = address HERE

After Instruction:
CNT = CNT + 1
if CNT = 0,
PC = address CONTINUE
if CNT ≠ 0,
PC = address HERE +1

**IORLW** **Inclusive OR Literal with W**

Syntax: [ *label* ] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → (W)

Status Affected: Z

Encoding:

| 11 | 1000 | kkkk | kkkk |
|---|---|---|---|

Description: The contents of the W register is OR'ed with the eight-bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process data | Write to W |

Example

```
IORLW   0x35
```

Before Instruction:
W = 0x9A

After Instruction:
W = 0xBF
Z = 0

## 15.2 DC Characteristics: PIC16C925/926 (Commercial, Industrial) PIC16LC925/926 (Commercial, Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>0°C ≤ TA ≤ +70°C for commercial<br>Operating voltage VDD range as described in DC spec | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | | **Input Low Voltage** | | | | | |
| | VIL | I/O ports | | | | | |
| D030 | | with TTL buffer | VSS | — | 0.15VDD | V | For entire VDD range |
| | | | Vss | — | 0.8V | V | 4.5V ≤ VDD ≤ 5.5V |
| D031 | | with Schmitt Trigger buffer | VSS | — | 0.2VDD | V | |
| D032 | | MCLR, OSC1 (in RC mode) | VSS | — | 0.2VDD | V | |
| D033 | | OSC1 (in XT, HS and LP) | VSS | — | 0.3VDD | V | **(Note 1)** |
| | | **Input High Voltage** | | | | | |
| | VIH | I/O ports | | — | | | |
| D040 | | with TTL buffer | 2.0 | — | VDD | V | 4.5V ≤ VDD ≤ 5.5V |
| D040A | | | 0.25VDD + 0.8V | — | VDD | V | For entire VDD range |
| D041 | | with Schmitt Trigger buffer | 0.8VDD | — | VDD | V | |
| D042 | | MCLR | 0.8VDD | — | VDD | V | |
| D042A | | OSC1 (XT, HS and LP) | 0.7VDD | — | VDD | V | **(Note 1)** |
| D043 | | OSC1 (in RC mode) | 0.9VDD | — | VDD | V | |
| D070 | IPURB | **PORTB Weak Pull-up Current** | 50 | 250 | 400 | µA | VDD = 5V, VPIN = VSS |
| | | **Input Leakage Current (Notes 2, 3)** | | | | | |
| D060 | IIL | I/O ports | — | — | ±1.0 | µA | Vss ≤ VPIN ≤ VDD, Pin at hi-Z |
| D061 | | MCLR, RA4/T0CKI | — | — | ±5 | µA | Vss ≤ VPIN ≤ VDD |
| D063 | | OSC1 | — | — | ±5 | µA | Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration |
| | | **Output Low Voltage** | | | | | |
| D080 | VOL | I/O ports | — | — | 0.6 | V | IOL = 4.0 mA, VDD = 4.5V |
| D083 | | OSC2/CLKOUT (RC osc mode) | — | — | 0.6 | V | IOL = 1.6 mA, VDD = 4.5V |
| | | **Output High Voltage** | | | | | |
| D090 | VOH | I/O ports **(Note 3)** | VDD - 0.7 | — | — | V | IOH = -3.0 mA, VDD = 4.5V |
| D092 | | OSC2/CLKOUT (RC osc mode) | VDD - 0.7 | — | — | V | IOH = -1.3 mA, VDD = 4.5V |
| | | **Capacitive Loading Specs on Output Pins** | | | | | |
| D100 | COSC2 | OSC2 pin | — | — | 15 | pF | In XT, HS and LP modes when external clock is used to drive OSC1. |
| D101 | CIO | All I/O pins and OSC2 (in RC) | — | — | 50 | pF | |
| D102 | CB | SCL, SDA in $I^2C$ mode | — | — | 400 | pF | |
| D150 | VDD | **Open Drain High Voltage** | — | — | 8.5 | V | RA4 pin |

† Data in "Typ" column is at 5 V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C925/926 be driven with external clock in RC mode.

**2:** The leakage current on the MCLR pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**3:** Negative current is defined as current sourced by the pin.



**FIGURE 15-12: SPI SLAVE MODE TIMING (CKE = 0)**

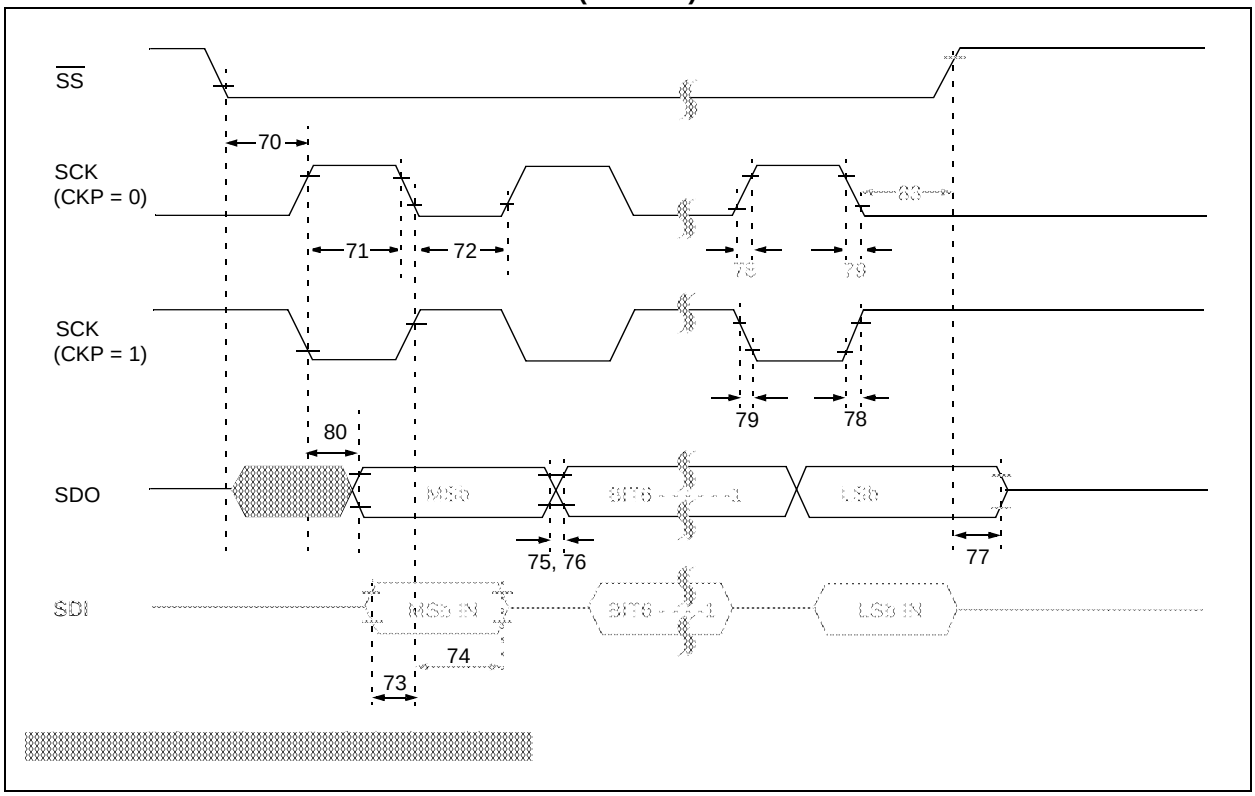



**FIGURE 15-13: SPI SLAVE MODE TIMING (CKE = 1)**

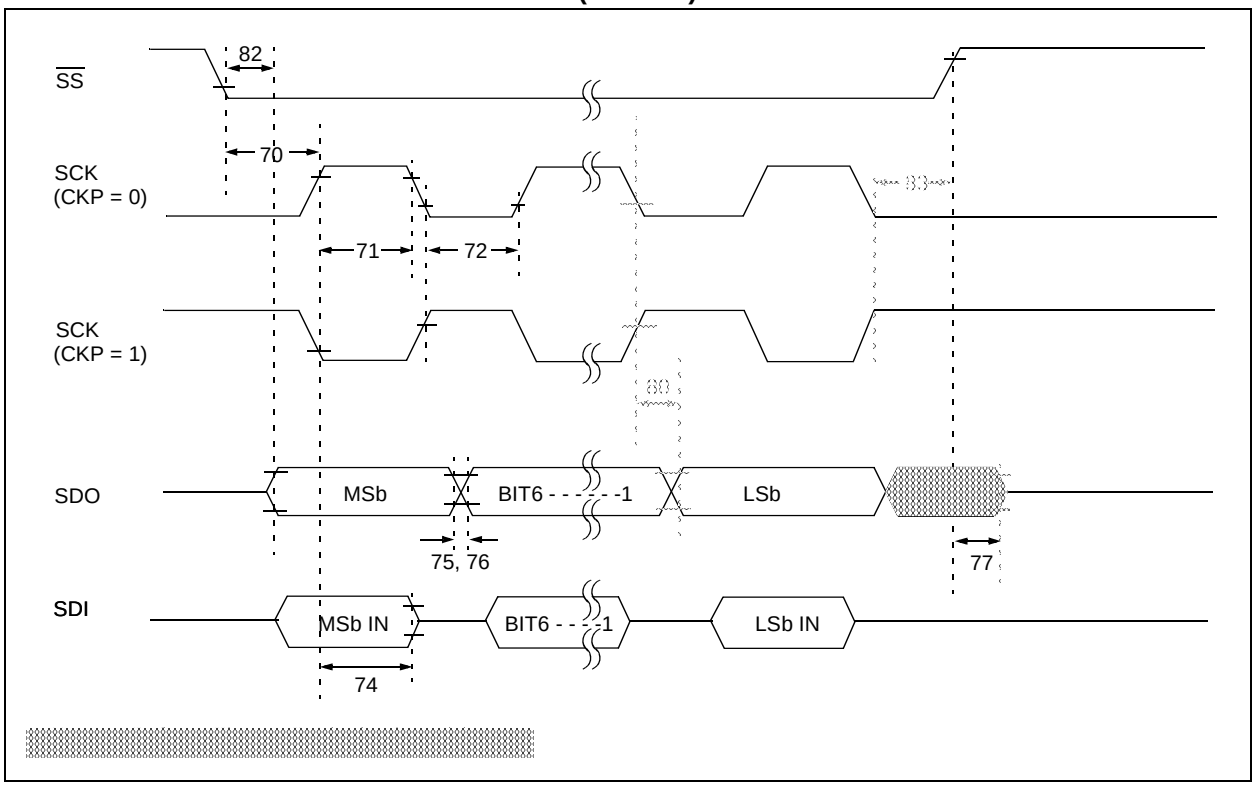


**TABLE 15-11: A/D CONVERTER CHARACTERISTICS: PIC16C925/926 (COMMERCIAL, INDUSTRIAL) PIC16LC925/926 (COMMERCIAL, INDUSTRIAL)**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| A01 | NR | Resolution | | — | — | 10-bits | bit | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A02 | EABS | Total Absolute error | | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A03 | EIL | Integral linearity error | | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A04 | EDL | Differential linearity error | | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A05 | EFS | Full scale error | | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A06 | EOFF | Offset error | | — | — | < ± 2 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A07 | EGN | Gain error | | — | — | < ± 1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A10 | — | Monotonicity | | — | guaranteed | — | — | VSS ≤ VAIN ≤ VREF |
| A20 | VREF | Reference voltage | | AVDD - 2.5V | — | AVDD + 0.3 | V | |
| A25 | VAIN | Analog input voltage | | VSS - 0.3 | — | VREF + 0.3 | V | |
| A30 | ZAIN | Recommended impedance of analog voltage source | | — | — | 10.0 | kΩ | |
| A40 | IAD | A/D conversion current (VDD) | PIC16C925/926 | — | 220 | — | μA | Average current consumption when A/D is on. **(Note 1)** |
| | | | PIC16**LC**925/926 | — | 90 | — | μA | |
| A50 | IREF | VREF input current **(Note 2)** | | 10 | — | 1000 | μA | During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD. |
| | | | | — | — | 10 | μA | During A/D Conversion cycle |

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

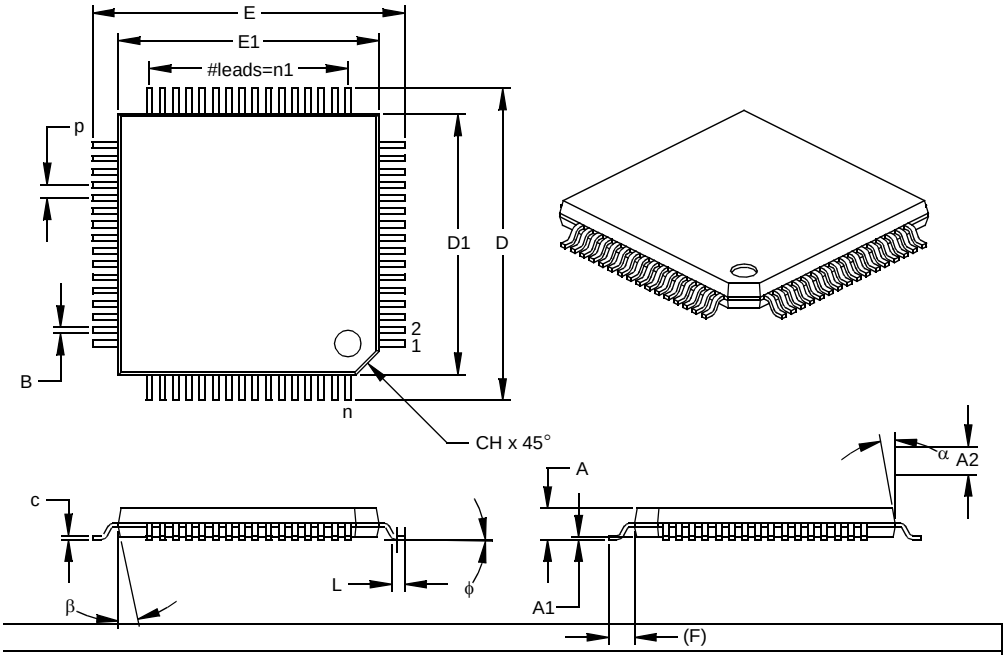
**Note 1:** When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

**2:** VREF current is from RA3 pin or VDD pin, whichever is selected as reference input.

## 17.2 Package Details

### 64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)

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




| Units | INCHES | | | MILLIMETERS* | | |
|---|---|---|---|---|---|---|
| Dimension Limits | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 64 | | | 64 | |
| Pitch | p | | .020 | | | 0.50 | 1.20 |
| Pins per Side | n1 | | 16 | | | 16 | 1.05 |
| Overall Height | A | .039 | .043 | .047 | 1.00 | 1.10 | 0.25 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 0.75 |
| Standoff § | A1 | .002 | .006 | .010 | 0.05 | 0.15 | |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 7 |
| Footprint (Reference) | (F) | | .039 | | | 1.00 | 12.25 |
| Foot Angle | φ | 0 | 3.5 | 7 | 0 | 3.5 | 12.25 |
| Overall Width | E | .463 | .472 | .482 | 11.75 | 12.00 | 10.10 |
| Overall Length | D | .463 | .472 | .482 | 11.75 | 12.00 | 10.10 |
| Molded Package Width | E1 | .390 | .394 | .398 | 9.90 | 10.00 | 0.23 |
| Molded Package Length | D1 | .390 | .394 | .398 | 9.90 | 10.00 | 0.27 |
| Lead Thickness | c | .005 | .007 | .009 | 0.13 | 0.18 | 1.14 |
| Lead Width | B | .007 | .009 | .011 | 0.17 | 0.22 | 15 |
| Pin 1 Corner Chamfer | CH | .025 | .035 | .045 | 0.64 | 0.89 | 15 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | |

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-026
Drawing No. C04-085