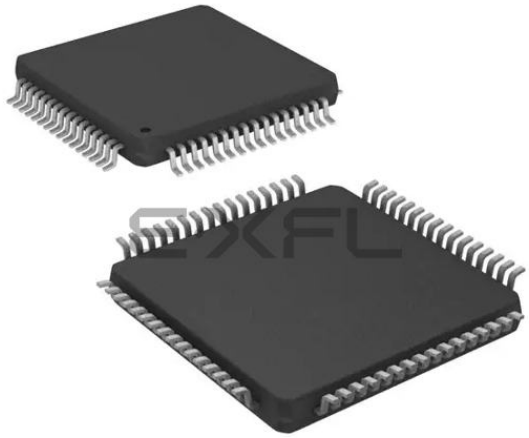


**Welcome to E-XFL.COM**

## 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 | 40MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 54 |
| Program Memory Size | 64KB (32K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 1K x 8 |
| RAM Size | 3936 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2V ~ 5.5V |
| Data Converters | A/D 12x10b |
| Oscillator Type | Internal |
| 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/pic18lf6622t-i-pt |

**TABLE 1-3: PIC18F6527/6622/6627/6722 PINOUT I/O DESCRIPTIONS (CONTINUED)**

| Pin Name | Pin Number | Pin Type | Buffer Type | Description |
|---|---|---|---|---|
| | TQFP | | | |
| | | | | PORTE is a bidirectional I/O port. |
| RE0/$\overline{RD}$/P2D | 2 | | | |
| RE0 | | I/O | ST | Digital I/O. |
| $\overline{RD}$ | | I | TTL | Read control for Parallel Slave Port. |
| P2D | | O | — | ECCP2 PWM output D. |
| RE1/$\overline{WR}$/P2C | 1 | | | |
| RE1 | | I/O | ST | Digital I/O. |
| $\overline{WR}$ | | I | TTL | Write control for Parallel Slave Port. |
| P2C | | O | — | ECCP2 PWM output C. |
| RE2/$\overline{CS}$/P2B | 64 | | | |
| RE2 | | I/O | ST | Digital I/O. |
| $\overline{CS}$ | | I | TTL | Chip select control for Parallel Slave Port. |
| P2B | | O | — | ECCP2 PWM output B. |
| RE3/P3C | 63 | | | |
| RE3 | | I/O | ST | Digital I/O. |
| P3C | | O | — | ECCP3 PWM output C. |
| RE4/P3B | 62 | | | |
| RE4 | | I/O | ST | Digital I/O. |
| P3B | | O | — | ECCP3 PWM output B. |
| RE5/P1C | 61 | | | |
| RE5 | | I/O | ST | Digital I/O. |
| P1C | | O | — | ECCP1 PWM output C. |
| RE6/P1B | 60 | | | |
| RE6 | | I/O | ST | Digital I/O. |
| P1B | | O | — | ECCP1 PWM output B. |
| RE7/ECCP2/P2A | 59 | | | |
| RE7 | | I/O | ST | Digital I/O. |
| ECCP2**(2)** | | I/O | ST | Enhanced Capture 2 input/Compare 2 output/ PWM 2 output. |
| P2A**(2)** | | O | — | ECCP2 PWM output A. |

**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 $I^2C$™ = $I^2C$/SMBus input buffer

**Note 1:** Default assignment for ECCP2 when Configuration bit, CCP2MX, is set.
**2:** Alternate assignment for ECCP2 when Configuration bit, CCP2MX, is cleared.

### 2.7.1 OSCILLATOR CONTROL REGISTER

The OSCCON register (Register 2-2) controls several aspects of the device clock's operation, both in full power operation and in power-managed modes.

The System Clock Select bits, SCS<1:0>, select the clock source. The available clock sources are the primary clock (defined by the FOSC<3:0> Configuration bits), the secondary clock (Timer1 oscillator) and the internal oscillator block. The clock source changes immediately after either of the SCS<1:0> bits are changed, following a brief clock transition interval. The SCS bits are reset on all forms of Reset.

The Internal Oscillator Frequency Select bits (IRCF<2:0>) select the frequency output of the internal oscillator block to drive the device clock. The choices are the INTRC source (31 kHz), the INTOSC source (8 MHz) or one of the frequencies derived from the INTOSC postscaler (31.25 kHz to 4 MHz). If the internal oscillator block is supplying the device clock, changing the states of these bits will have an immediate change on the internal oscillator's output. On device Resets, the default output frequency of the internal oscillator block is set at 1 MHz.

When a nominal output frequency of 31 kHz is selected (IRCF<2:0> = 000), users may choose which internal oscillator acts as the source. This is done with the INTSRC bit in the OSCTUNE register (OSCTUNE<7>). Setting this bit selects INTOSC as a 31.25 kHz clock source derived from the INTOSC postscaler. Clearing INTSRC selects INTRC (nominally 31 kHz) as the clock source and disables the INTOSC to reduce current consumption.

This option allows users to select the tunable and more precise INTOSC as a clock source, while maintaining power savings with a very low clock speed. Additionally, the INTOSC source will already be stable should a switch to a higher frequency be needed quickly. Regardless of the setting of INTSRC, INTRC always remains the clock source for features such as the Watchdog Timer and the Fail-Safe Clock Monitor.

The OSTS, IOFS and T1RUN bits indicate which clock source is currently providing the device clock. The OSTS bit indicates that the Oscillator Start-up Timer and PLL Start-up Timer (if enabled) have timed out and the primary clock is providing the device clock in primary clock modes. The IOFS bit indicates when the internal oscillator block has stabilized and is providing the device clock in RC Clock modes. The T1RUN bit (T1CON<6>) indicates when the Timer1 oscillator is providing the device clock in secondary clock modes. In power-managed modes, only one of these three bits will be set at any time. If none of these bits are set, the INTRC is providing the clock or the internal oscillator block has just started and is not yet stable.

The IDLEN bit controls whether the device goes into Sleep mode or one of the Idle modes when the SLEEP instruction is executed.

The use of the flag and control bits in the OSCCON register is discussed in more detail in **Section 3.0 "Power-Managed Modes"**.

| | |
|---|---|
| **Note 1:** | The Timer1 oscillator must be enabled to select the secondary clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON<3>). If the Timer1 oscillator is not enabled, then any attempt to select a secondary clock source will be ignored. |
| **2:** | It is recommended that the Timer1 oscillator be operating and stable before selecting the secondary clock source or a very long delay may occur while the Timer1 oscillator starts. |

### 2.7.2 OSCILLATOR TRANSITIONS

The PIC18F8722 family of devices contains circuitry to prevent clock "glitches" when switching between clock sources. A short pause in the device clock occurs during the clock switch. The length of this pause is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

Clock transitions are discussed in greater detail in **Section 3.1.2 "Entering Power-Managed Modes"**.

## 4.5 Device Reset Timers

The PIC18F8722 family of devices incorporates three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

### 4.5.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of the PIC18F8722 family of devices is an 11-bit counter which uses the INTRC source as the clock input. While the PWRT is counting, the device is held in Reset.

The power-up time delay depends on the INTRC clock and will vary from chip-to-chip due to temperature and process variation. See DC parameter 33 in Table 28-12 for details.

The PWRT is enabled by clearing the $\overline{\text{PWRTEN}}$ Configuration bit.

### 4.5.2 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (parameter 33, Table 28-12). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP, HS and HSPLL modes and only on Power-on Reset, or on exit from most power-managed modes.

### 4.5.3 PLL LOCK TIME-OUT

With the PLL enabled in its PLL mode, the time-out sequence following a Power-on Reset is slightly different from other oscillator modes. A separate timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL lock time-out ($T_{PLL}$) is typically 2 ms and follows the oscillator start-up time-out.

### 4.5.4 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

1. After the POR pulse has cleared, PWRT time-out is invoked (if enabled).
2. Then, the OST is activated.

The total time-out will vary based on oscillator configuration and the status of the PWRT. Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7 all depict time-out sequences on power-up, with the Power-up Timer enabled and the device operating in HS Oscillator mode. Figures 4-3 through 4-6 also apply to devices operating in XT or LP modes. For devices in RC mode and with the PWRT disabled, on the other hand, there will be no time-out at all.

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, all time-outs will expire. Bringing $\overline{\text{MCLR}}$ high will begin execution immediately (Figure 4-5). This is useful for testing purposes or to synchronize more than one PIC18F8722 family device operating in parallel.

**TABLE 4-2: TIME-OUT IN VARIOUS SITUATIONS**

| Oscillator Configuration | Power-up[2] and Brown-out | | Exit from Power-Managed Mode |
|---|---|---|---|
| | $\overline{\text{PWRTEN}}$ = 0 | $\overline{\text{PWRTEN}}$ = 1 | |
| HSPLL | $T_{PWRT}$[1] + 1024 $T_{OSC}$ + $T_{PLL}$[2] | 1024 $T_{OSC}$ + $T_{PLL}$[2] | 1024 $T_{OSC}$ + $T_{PLL}$[2] |
| HS, XT, LP | $T_{PWRT}$[1] + 1024 $T_{OSC}$ | 1024 $T_{OSC}$ | 1024 $T_{OSC}$ |
| EC, ECIO | $T_{PWRT}$[1] | — | — |
| RC, RCIO | $T_{PWRT}$[1] | — | — |
| INTIO1, INTIO2 | $T_{PWRT}$[1] | — | — |

**Note 1:** See parameter 33, Table 28-12.
**2:** 2 ms is the nominal time required for the PLL to lock.

**TABLE 4-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)**

| Register | Applicable Devices | | | | Power-on Reset, Brown-out Reset | $\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets | Wake-up via WDT or Interrupt |
|---|---|---|---|---|---|---|---|
| FSR1H | 6X27 | 6X22 | 8X27 | 8X22 | ---- 0000 | ---- 0000 | ---- uuuu |
| FSR1L | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| BSR | 6X27 | 6X22 | 8X27 | 8X22 | ---- 0000 | ---- 0000 | ---- uuuu |
| INDF2 | 6X27 | 6X22 | 8X27 | 8X22 | N/A | N/A | N/A |
| POSTINC2 | 6X27 | 6X22 | 8X27 | 8X22 | N/A | N/A | N/A |
| POSTDEC2 | 6X27 | 6X22 | 8X27 | 8X22 | N/A | N/A | N/A |
| PREINC2 | 6X27 | 6X22 | 8X27 | 8X22 | N/A | N/A | N/A |
| PLUSW2 | 6X27 | 6X22 | 8X27 | 8X22 | N/A | N/A | N/A |
| FSR2H | 6X27 | 6X22 | 8X27 | 8X22 | ---- 0000 | ---- 0000 | ---- uuuu |
| FSR2L | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| STATUS | 6X27 | 6X22 | 8X27 | 8X22 | ---x xxxx | ---u uuuu | ---u uuuu |
| TMR0H | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |
| TMR0L | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T0CON | 6X27 | 6X22 | 8X27 | 8X22 | 1111 1111 | 1111 1111 | uuuu uuuu |
| OSCCON | 6X27 | 6X22 | 8X27 | 8X22 | 0100 q000 | 0100 q000 | uuuu uuqu |
| HLVDCON | 6X27 | 6X22 | 8X27 | 8X22 | 0-00 0101 | 0-00 0101 | u-uu uuuu |
| WDTCON | 6X27 | 6X22 | 8X27 | 8X22 | ---- ---0 | ---- ---0 | ---- ---u |
| RCON(4) | 6X27 | 6X22 | 8X27 | 8X22 | 0q-1 11q0 | 0q-q qquu | uq-u qquu |
| TMR1H | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| TMR1L | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| T1CON | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | u0uu uuuu | uuuu uuuu |
| TMR2 | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PR2 | 6X27 | 6X22 | 8X27 | 8X22 | 1111 1111 | uuuu uuuu | uuuu uuuu |
| T2CON | 6X27 | 6X22 | 8X27 | 8X22 | -000 0000 | -000 0000 | -uuu uuuu |
| SSP1BUF | 6X27 | 6X22 | 8X27 | 8X22 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| SSP1ADD | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1STAT | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1CON1 | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SSP1CON2 | 6X27 | 6X22 | 8X27 | 8X22 | 0000 0000 | 0000 0000 | uuuu uuuu |

**Legend:** u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.
Shaded cells indicate conditions do not apply for the designated device.

**Note 1:** One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

**2:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

**3:** When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

**4:** See Table 4-3 for Reset value for specific condition.

**5:** Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

## 6.4 Erasing Flash Program Memory

The minimum erase block is 32 words or 64 bytes. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the micro-controller itself, a block of 64 bytes of program memory is erased. The Most Significant 16 bits of the TBLPTR<21:6> point to the block being erased. TBLPTR<5:0> are ignored.

The EECON1 register commands the erase operation. The EEPGD bit must be set to point to the Flash program memory. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation.

For protection, the write initiate sequence for EECON2 must be used.

A long write is necessary for erasing the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

### 6.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

The sequence of events for erasing a block of internal program memory location is:

1. Load Table Pointer register with address of row being erased.
2. Set the EECON1 register for the erase operation:
   - set EEPGD bit to point to program memory;
   - clear the CFGS bit to access program memory;
   - set WREN bit to enable writes;
   - set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit. This will begin the row erase cycle.
7. The CPU will stall for duration of the erase for TIW (see parameter D133A).
8. Re-enable interrupts.

**EXAMPLE 6-2: ERASING A FLASH PROGRAM MEMORY ROW**

```
                    MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                    MOVWF   TBLPTRU                 ; address of the memory block
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
        ERASE_ROW
                    BSF     EECON1, EEPGD           ; point to Flash program memory
                    BCF     EECON1, CFGS            ; access Flash program memory
                    BSF     EECON1, WREN            ; enable write to memory
                    BSF     EECON1, FREE            ; enable Row Erase operation
                    BCF     INTCON, GIE             ; disable interrupts
    Required        MOVLW   55h
    Sequence        MOVWF   EECON2                  ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2                  ; write 0AAh
                    BSF     EECON1, WR              ; start erase (CPU stall)
                    BSF     INTCON, GIE             ; re-enable interrupts
```

## 10.0 INTERRUPTS

The PIC18F8722 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

There are ten registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3
- PIE1, PIE2, PIE3
- IPR1, IPR2, IPR3

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

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

- **Flag bit** to indicate that an interrupt event occurred
- **Enable bit** that allows program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** to select high priority or low priority

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

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

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

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

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

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

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

**REGISTER 10-2: INTCON2: INTERRUPT CONTROL REGISTER 2**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{RBPU}}$ | INTEDG0 | INTEDG1 | INTEDG2 | INTEDG3 | TMR0IP | INT3IP | RBIP |
| 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 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0:** External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 **INTEDG3:** External Interrupt 3 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **INT3IP:** INT3 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **RBIP:** RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

**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.

**REGISTER 10-3: INTCON3: INTERRUPT CONTROL REGISTER 3**

| 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.

**REGISTER 10-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3**

| R/W-0 | R/W-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| SSP2IP | BCL2IP | RC2IP | TX2IP | TMR4IP | CCP5IP | CCP4IP | CCP3IP |
| 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 **SSP2IP:** MSSP2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **BCL2IP:** MSSP2 Bus Collision Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC2IP:** EUSART2 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX2IP:** EUSART2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **TMR4IP:** TMR4 to PR4 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP5IP:** CCP5 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CCP4IP:** CCP4 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CCP3IP:** ECCP3 Interrupt Priority bit
1 = High priority
0 = Low priority

**TABLE 11-1: PORTA FUNCTIONS**

| Pin Name | Function | TRIS Setting | I/O | I/O Type | Description |
|---|---|---|---|---|---|
| RA0/AN0 | RA0 | 0 | O | DIG | LATA<0> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<0> data input; disabled when analog input enabled. |
| | AN0 | 1 | I | ANA | A/D input channel 0. Default input configuration on POR; does not affect digital output. |
| RA1/AN1 | RA1 | 0 | O | DIG | LATA<1> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<1> data input; disabled when analog input enabled. |
| | AN1 | 1 | I | ANA | A/D input channel 1. Default input configuration on POR; does not affect digital output. |
| RA2/AN2/VREF- | RA2 | 0 | O | DIG | LATA<2> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<2> data input. Disabled when analog functions enabled. |
| | AN2 | 1 | I | ANA | A/D input channel 2. Default input configuration on POR. |
| | VREF- | 1 | I | ANA | Comparator voltage reference low input and A/D voltage reference low input. |
| RA3/AN3/VREF+ | RA3 | 0 | O | DIG | LATA<3> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<3> data input; disabled when analog input enabled. |
| | AN3 | 1 | I | ANA | A/D input channel 3. Default input configuration on POR. |
| | VREF+ | 1 | I | ANA | Comparator voltage reference high input and A/D voltage reference high input. |
| RA4/T0CKI | RA4 | 0 | O | DIG | LATA<4> data output. |
| | | 1 | I | ST | PORTA<4> data input; default configuration on POR. |
| | T0CKI | x | I | ST | Timer0 clock input. |
| RA5/AN4/HLVDIN | RA5 | 0 | O | DIG | LATA<5> data output; not affected by analog input. |
| | | 1 | I | TTL | PORTA<5> data input; disabled when analog input enabled. |
| | AN4 | 1 | I | ANA | A/D input channel 4. Default configuration on POR. |
| | HLVDIN | 1 | I | ANA | High/Low-Voltage Detect external trip point input. |
| OSC2/CLKO/RA6 | OSC2 | x | O | ANA | Main oscillator feedback output connection (XT, HS, HSPLL and LP modes). |
| | CLKO | x | O | DIG | System cycle clock output (FOSC/4) in all oscillator modes except RC, INTIO7 and EC. |
| | RA6 | 0 | O | DIG | LATA<6> data output. Enabled in RCIO, INTIO2 and ECIO modes only. |
| | | 1 | I | TTL | PORTA<6> data input. Enabled in RCIO, INTIO2 and ECIO modes only. |
| OSC1/CLKI/RA7 | OSC1 | x | I | ANA | Main oscillator input connection. |
| | CLKI | x | I | ANA | Main clock input connection. |
| | RA7 | 0 | O | DIG | LATA<7> data output. Disabled in external oscillator modes. |
| | | 1 | I | TTL | PORTA<7> data input. Disabled in external oscillator modes. |

**Legend:** PWR = Power Supply, O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST= Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

**TABLE 11-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---|---|---|---|---|---|---|---|---|---|
| PORTA | RA7(1) | RA6(1) | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 61 |
| LATA | LATA7(1) | LATA6(1) | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | 60 |
| TRISA | TRISA7(1) | TRISA6(1) | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 60 |
| ADCON1 | — | — | VCFG1 | VCFG0 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 59 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used by PORTA.

**Note 1:** RA<7:6> and their associated latch and data direction bits are enabled as I/O pins based on oscillator configuration; otherwise, they are read as '0'.

## 11.4 PORTD, TRISD and LATD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISD. Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATD) is also memory mapped. Read-modify-write operations on the LATD register read and write the latched output value for PORTD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

| **Note:** | On a Power-on Reset, these pins are configured as digital inputs. |
|---|---|

In 80-pin devices, PORTD is multiplexed with the system bus as part of the external memory interface. I/O port and other functions are only available when the interface is disabled by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTD is the low-order byte of the multiplexed address/data bus (AD<7:0>). The TRISD bits are also overridden.

PORTD can also be configured to function as an 8-bit wide parallel microprocessor port by setting the PSPMODE control bit (PSPCON<4>). In this mode, parallel port data takes priority over other digital I/O (but not the external memory interface). When the parallel port is active, the input buffers are TTL. For more information, refer to **Section 11.10 "Parallel Slave Port"**.

**EXAMPLE 11-4: INITIALIZING PORTD**

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

## 13.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Reset on CCP Special Event Trigger
- Device clock status flag (T1RUN)

A simplified block diagram of the Timer1 module is shown in Figure 13-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 13-2.

The module incorporates its own low-power oscillator to provide an additional clocking option. The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components and code overhead.

Timer1 is controlled through the T1CON Control register (Register 13-1). It also contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

**REGISTER 13-1: T1CON: TIMER1 CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON |
| 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 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 **T1RUN:** Timer1 System Clock Status bit
1 = Device clock is derived from Timer1 oscillator
0 = Device clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T13CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

## 21.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor ($C_{HOLD}$) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 21-3. The source impedance ($R_S$) and the internal sampling switch ($R_{SS}$) impedance directly affect the time required to charge the capacitor $C_{HOLD}$. The sampling switch ($R_{SS}$) impedance varies over the device voltage ($V_{DD}$). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ.** After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

| **Note:** | When the conversion is started, the holding capacitor is disconnected from the input pin. |
|---|---|

To calculate the minimum acquisition time, Equation 21-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 21-3 shows the calculation of the minimum required acquisition time $T_{ACQ}$. This calculation is based on the following application system assumptions:

| | | |
|---|---|---|
| $C_{HOLD}$ | = | 25 pF |
| Rs | = | 2.5 kΩ |
| Conversion Error | ≤ | 1/2 LSb |
| $V_{DD}$ | = | 5V → Rss = 2 kΩ |
| Temperature | = | 85°C (system max.) |

**EQUATION 21-1: ACQUISITION TIME**

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

**EQUATION 21-2: A/D MINIMUM CHARGING TIME**

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2048)$$

**EQUATION 21-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME**

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

$T_{AMP} = 0.2\ \mu s$

$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$
$(85°C - 25°C)(0.02\ \mu s/°C)$
$1.2\ \mu s$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, $T_{COFF}$ = 0 ms.

$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2047)\ \mu s$
$-(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ln(0.0004883)\ \mu s$
$1.05\ \mu s$

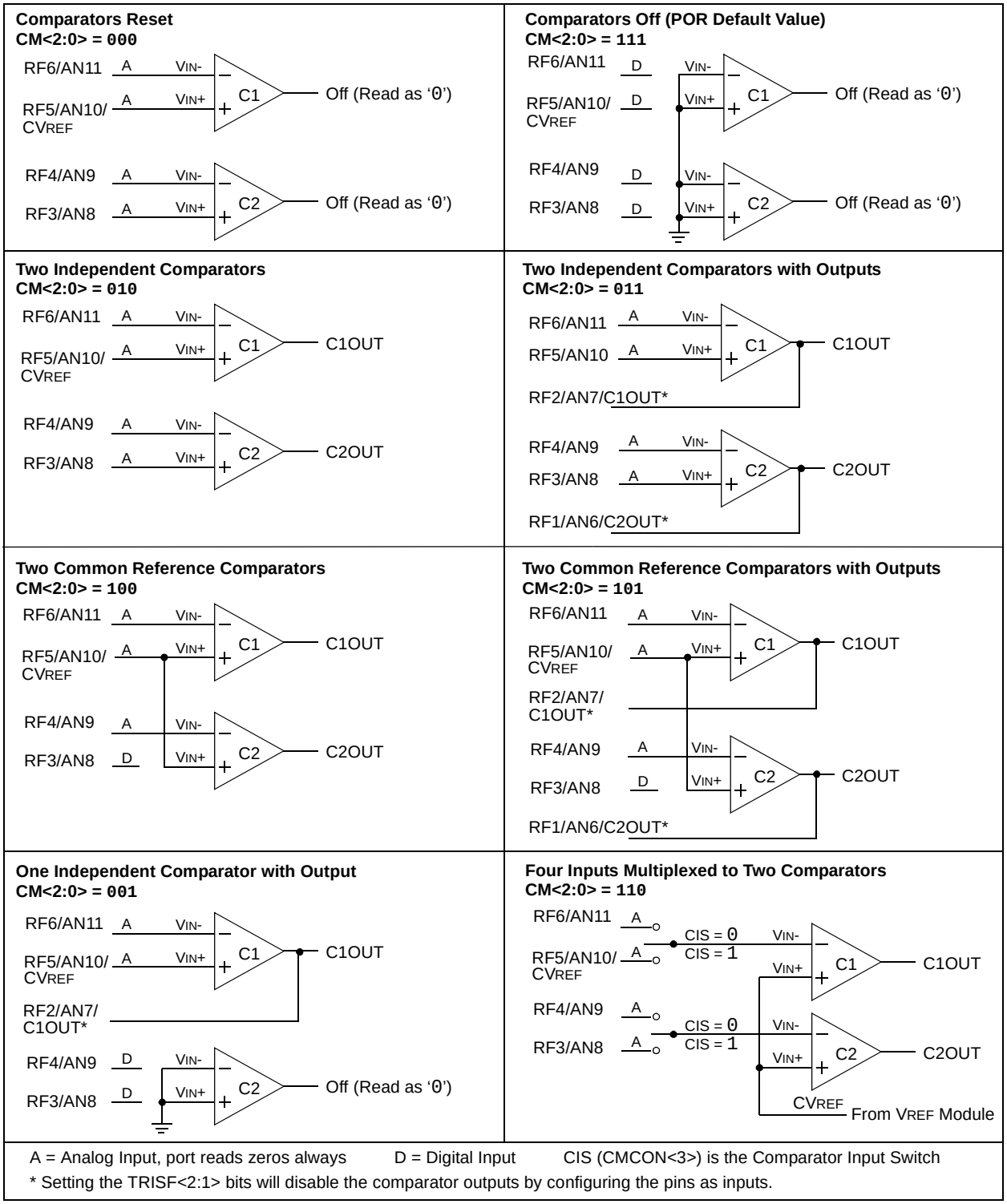
$T_{ACQ} = 0.2\ \mu s + 1\ \mu s + 1.2\ \mu s$
$2.4\ \mu s$

## 22.1 Comparator Configuration

There are eight modes of operation for the comparators, shown in Figure 22-1. Bits CM<2:0> of the CMCON register are used to select these modes. The TRISF register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in **Section 28.0 "Electrical Characteristics"**.

| **Note:** | Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur. |
|---|---|

**FIGURE 22-1: COMPARATOR I/O OPERATING MODES**




A = Analog Input, port reads zeros always D = Digital Input CIS (CMCON<3>) is the Comparator Input Switch

* Setting the TRISF<2:1> bits will disable the comparator outputs by configuring the pins as inputs.

## 25.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTOSC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is LP, XT, HS or HSPLL (crystal-based modes). Other sources do not require an OST start-up delay; for these, Two-Speed Start-up should be disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer after a Power-on Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits IRCF<2:0> immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF2:0> bits prior to entering Sleep mode.

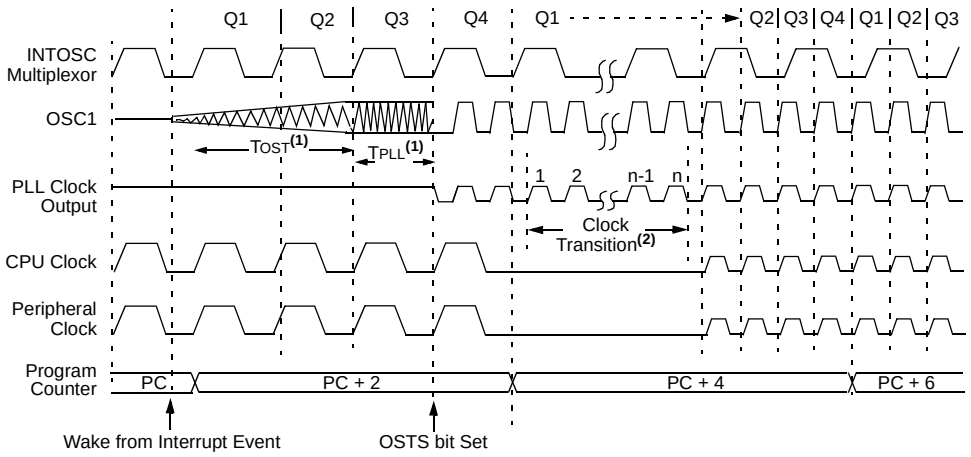
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

### 25.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTOSC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including multiple SLEEP instructions (refer to **Section 3.1.4 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS<1:0> bit settings or issue SLEEP instructions before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

**FIGURE 25-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)**




**Note 1:** TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

**2:** Clock transition typically occurs within 2-4 TOSC.

## 25.5 Program Verification and Code Protection

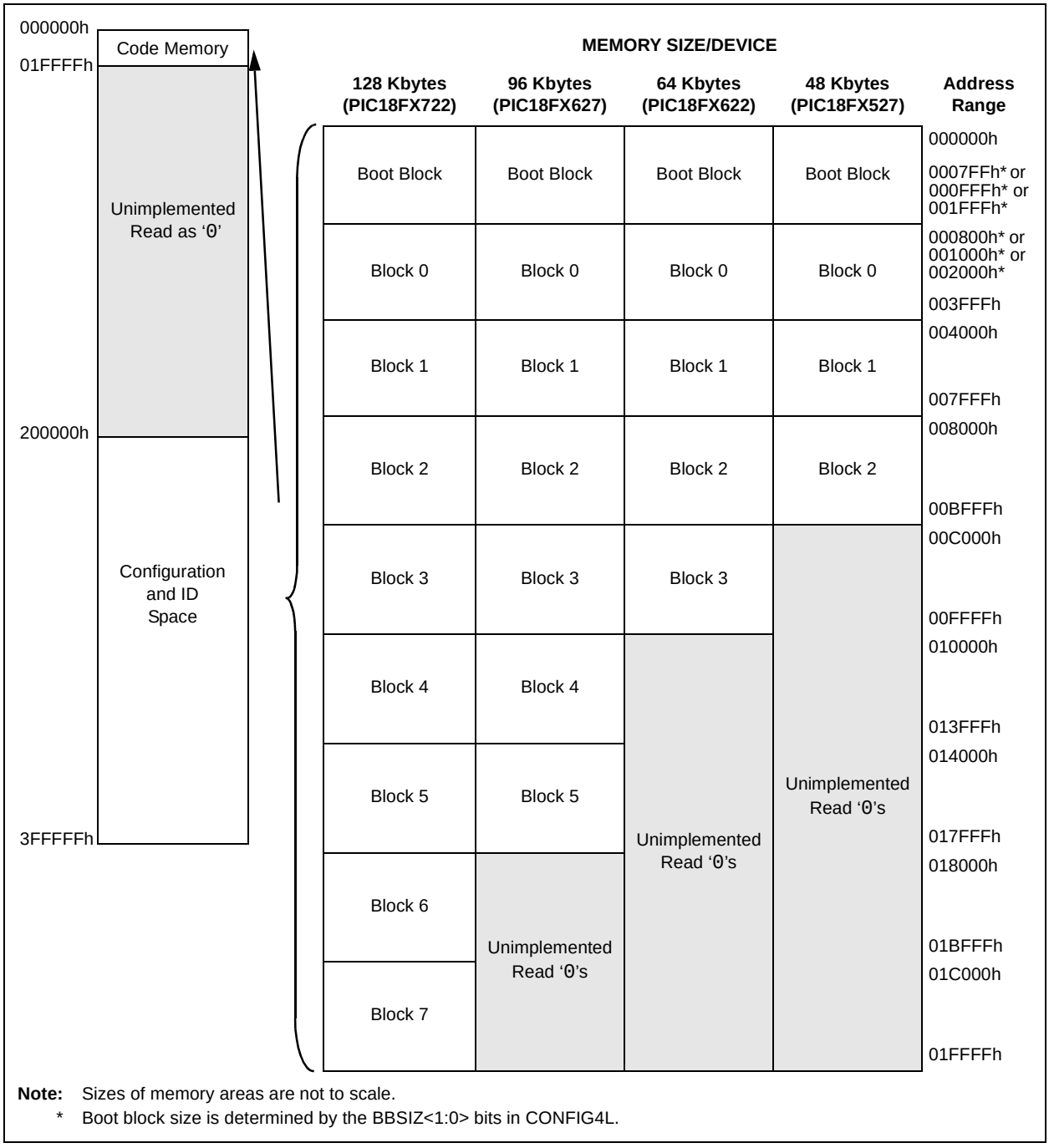
The user program memory is divided into four blocks for PIC18F6527/8527 devices, five blocks for PIC18F6622/8622 devices, six blocks for PIC18F6627/8627 devices and eight blocks for PIC18F6722/8722 devices. One of these is a boot block of 2, 4 or 8 Kbytes. The remainder of the memory is divided into blocks on binary boundaries.

Each of the blocks has three code protection bits associated with them. They are:

- Code-Protect bit (CPn)
- Write-Protect bit (WRTn)
- External Block Table Read bit (EBTRn)

Figure 25-5 shows the program memory organization for 48, 64, 96 and 128-Kbyte devices and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 25-3.

**FIGURE 25-5: CODE-PROTECTED PROGRAM MEMORY FOR THE PIC18F8722 FAMILY**




**Note:** Sizes of memory areas are not to scale.

* Boot block size is determined by the BBSIZ<1:0> bits in CONFIG4L.

**MOVFF** **Move f to f**

Syntax: MOVFF $f_s$,$f_d$

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

| | | | | |
|---|---|---|---|---|
| 1st word (source) | 1100 | ffff | ffff | ffffs |
| 2nd word (destin.) | 1111 | ffff | ffff | ffffd |

Description: The contents of source register '$f_s$' are moved to destination register '$f_d$'. Location of source '$f_s$' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination '$f_d$' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

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

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' (src) | Process Data | No operation |
| Decode | No operation<br>No dummy read | No operation | Write register 'f' (dest) |

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 33h
REG2 = 11h
After Instruction
REG1 = 33h
REG2 = 33h

**MOVLB** **Move Literal to Low Nibble in BSR**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0001 | kkkk | kkkk |

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0' regardless of the value of $k_7$:$k_4$.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process Data | Write literal 'k' to BSR |

Example: MOVLB 5

Before Instruction
BSR Register = 02h
After Instruction
BSR Register = 05h

### 26.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

| **Note:** | Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely. |
|---|---|

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing (**Section 5.5.1 "Indexed Addressing with Literal Offset"**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank (a = 0) or in a GPR bank designated by the BSR (a = 1). When the extended instruction set is enabled and a = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward-compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 26.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset Addressing mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind that, when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset Addressing mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

#### 26.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument 'f' in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within the brackets, will generate an error in the MPASM Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled), when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument 'd' functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command line option, /y, or the PE directive in the source listing.

### 26.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18F8722 family, it is very important to consider the type of code. A large, re-entrant application that is written in C and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

## W



## X