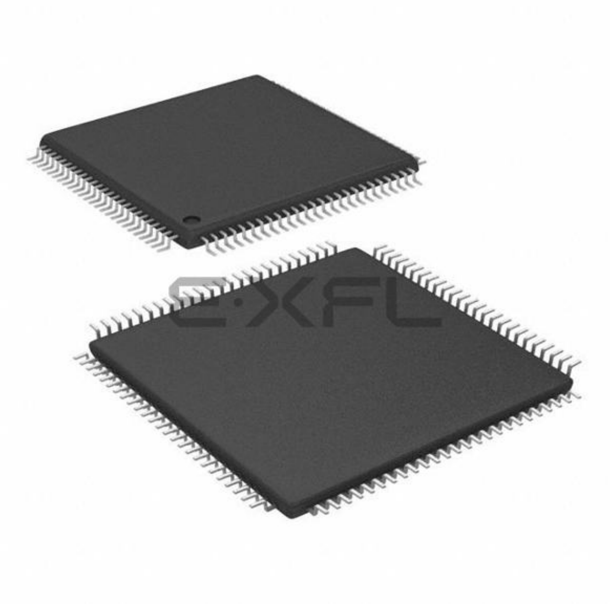


Welcome to E-XFL.COM

## 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 | Obsolete |
| Core Processor | PIC |
| Core Size | 16-Bit |
| Speed | 32MHz |
| Connectivity | I²C, IrDA, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, LVD, POR, PWM, WDT |
| Number of I/O | 85 |
| Program Memory Size | 64KB (22K x 24) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 16K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2V ~ 3.6V |
| Data Converters | A/D 16x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 100-TQFP |
| Supplier Device Package | 100-TQFP (14x14) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic24fj64ga110t-i-pf |

## TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

### Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

### Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

### Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

## 1.2 Other Special Features

- **Peripheral Pin Select:** The Peripheral Pin Select (PPS) feature allows most digital peripherals to be mapped over a fixed set of digital I/O pins. Users may independently map the input and/or output of any one of the many digital peripherals to any one of the I/O pins.
- **Communications:** The PIC24FJ256GA110 family incorporates a range of serial communication peripherals to handle a range of application requirements. There are three independent I$^2$C™ modules that support both Master and Slave modes of operation. Devices also have, through the Peripheral Pin Select (PPS) feature, four independent UARTs with built-in IrDA® encoder/decoders and three SPI modules.
- **Analog Features:** All members of the PIC24FJ256GA110 family include a 10-bit A/D Converter module and a triple comparator module. The A/D module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period, as well as faster sampling speeds. The comparator module includes three analog comparators that are configurable for a wide range of operations.
- **CTMU Interface:** In addition to their other analog features, members of the PIC24FJ256GA110 family include the brand new CTMU interface module. This provides a convenient method for precision time measurement and pulse generation, and can serve as an interface for capacitive sensors.
- **Parallel Master Port:** One of the general purpose I/O ports can be reconfigured for enhanced parallel data communications. In this mode, the port can be configured for both master and slave operations, and supports 8-bit transfers with up to 16 external address lines in Master modes.
- **Real-Time Clock/Calendar:** This module implements a full-featured clock and calendar with alarm functions in hardware, freeing up the timer resources and program memory space for the use of the core application.

## 1.3 Details on Individual Family Members

Devices in the PIC24FJ256GA110 family are available in 64-pin, 80-pin and 100-pin packages. The general block diagram for all devices is shown in Figure 1-1.

The devices are differentiated from each other in four ways:

1. Flash program memory (64 Kbytes for PIC24FJ64GA1 devices, 128 Kbytes for PIC24FJ128GA1 devices, 192 Kbytes for PIC24FJ192GA1 devices and 256 Kbytes for PIC24FJ256GA1 devices).
2. Available I/O pins and ports (53 pins on 6 ports for 64-pin devices, 69 pins on 7 ports for 80-pin devices and 85 pins on 7 ports for 100-pin devices).
3. Available Interrupt-on-Change Notification (ICN) inputs (same as the number of available I/O pins for all devices).
4. Available remappable pins (31 pins on 64-pin devices, 42 pins on 80-pin devices and 46 pins on 100-pin devices)

All other features for devices in this family are identical. These are summarized in Table 1-1.

A list of the pin features available on the PIC24FJ256GA110 family devices, sorted by function, is shown in Table 1-4. Note that this table shows the pin location of individual peripheral features and not how they are multiplexed on the same pin. This information is provided in the pinout diagrams in the beginning of this data sheet. Multiplexed features are sorted by the priority given to a feature, with the highest priority peripheral being listed first.

**TABLE 1-3: DEVICE FEATURES FOR THE PIC24FJ256GA110 FAMILY: 100-PIN DEVICES**

| Features | PIC24FJ64GA110 | PIC24FJ128GA110 | PIC24FJ192GA110 | PIC24FJ256GA110 |
|---|---|---|---|---|
| Operating Frequency | DC – 32 MHz | | | |
| Program Memory (bytes) | 64K | 128K | 192K | 256K |
| Program Memory (instructions) | 22,016 | 44,032 | 67,072 | 87,552 |
| Data Memory (bytes) | 16,384 | | | |
| Interrupt Sources (soft vectors/NMI traps) | 66 (62/4) | | | |
| I/O Ports | Ports A, B, C, D, E, F, G | | | |
| Total I/O Pins | 85 | | | |
| Remappable Pins | 46 (32 I/O, 14 input only) | | | |
| Timers: | | | | |
| Total Number (16-bit) | 5(1) | | | |
| 32-Bit (from paired 16-bit timers) | 2 | | | |
| Input Capture Channels | 9(1) | | | |
| Output Compare/PWM Channels | 9(1) | | | |
| Input Change Notification Interrupt | 85 | | | |
| Serial Communications: | | | | |
| UART | 4(1) | | | |
| SPI (3-wire/4-wire) | 3(1) | | | |
| I²C™ | 3 | | | |
| Parallel Communications (PMP/PSP) | Yes | | | |
| JTAG Boundary Scan | Yes | | | |
| 10-Bit Analog-to-Digital Module (input channels) | 16 | | | |
| Analog Comparators | 3 | | | |
| CTMU Interface | Yes | | | |
| Resets (and delays) | POR, BOR, RESET Instruction, $\overline{\text{MCLR}}$, WDT; Illegal Opcode, REPEAT Instruction, Hardware Traps, Configuration Word Mismatch (PWRT, OST, PLL Lock) | | | |
| Instruction Set | 76 Base Instructions, Multiple Addressing Mode Variations | | | |
| Packages | 100-Pin TQFP | | | |

**Note 1:** Peripherals are accessible through remappable pins.

**TABLE 1-4: PIC24FJ256GA110 FAMILY PINOUT DESCRIPTIONS (CONTINUED)**

| Function | Pin Number | | | I/O | Input Buffer | Description |
|---|---|---|---|---|---|---|
| | 64-Pin TQFP, QFN | 80-Pin TQFP | 100-Pin TQFP | | | |
| CN0 | 48 | 60 | 74 | I | ST | Interrupt-on-Change Inputs. |
| CN1 | 47 | 59 | 73 | I | ST | |
| CN2 | 16 | 20 | 25 | I | ST | |
| CN3 | 15 | 19 | 24 | I | ST | |
| CN4 | 14 | 18 | 23 | I | ST | |
| CN5 | 13 | 17 | 22 | I | ST | |
| CN6 | 12 | 16 | 21 | I | ST | |
| CN7 | 11 | 15 | 20 | I | ST | |
| CN8 | 4 | 6 | 10 | I | ST | |
| CN9 | 5 | 7 | 11 | I | ST | |
| CN10 | 6 | 8 | 12 | I | ST | |
| CN11 | 8 | 10 | 14 | I | ST | |
| CN12 | 30 | 36 | 44 | I | ST | |
| CN13 | 52 | 66 | 81 | I | ST | |
| CN14 | 53 | 67 | 82 | I | ST | |
| CN15 | 54 | 68 | 83 | I | ST | |
| CN16 | 55 | 69 | 84 | I | ST | |
| CN17 | 31 | 39 | 49 | I | ST | |
| CN18 | 32 | 40 | 50 | I | ST | |
| CN19 | — | 65 | 80 | I | ST | |
| CN20 | — | 37 | 47 | I | ST | |
| CN21 | — | 38 | 48 | I | ST | |
| CN22 | 40 | 50 | 64 | I | ST | |
| CN23 | 39 | 49 | 63 | I | ST | |
| CN24 | 17 | 21 | 26 | I | ST | |
| CN25 | 18 | 22 | 27 | I | ST | |
| CN26 | 21 | 27 | 32 | I | ST | |
| CN27 | 22 | 28 | 33 | I | ST | |
| CN28 | 23 | 29 | 34 | I | ST | |
| CN29 | 24 | 30 | 35 | I | ST | |
| CN30 | 27 | 33 | 41 | I | ST | |
| CN31 | 28 | 34 | 42 | I | ST | |
| CN32 | 29 | 35 | 43 | I | ST | |
| CN33 | — | — | 17 | I | ST | |
| CN34 | — | — | 38 | I | ST | |
| CN35 | — | — | 58 | I | ST | |
| CN36 | — | — | 59 | I | ST | |
| CN37 | — | — | 60 | I | ST | |
| CN38 | — | — | 61 | I | ST | |
| CN39 | — | — | 91 | I | ST | |
| CN40 | — | — | 92 | I | ST | |
| CN41 | — | 23 | 28 | I | ST | |
| CN42 | — | 24 | 29 | I | ST | |

**Legend:** TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; $I^2C$™ = $I^2C$/SMBus input buffer

#### 6.2.1 POR AND LONG OSCILLATOR START-UP TIMES

The oscillator start-up circuitry and its associated delay timers are not linked to the device Reset delays that occur at power-up. Some crystal circuits (especially low-frequency crystals) will have a relatively long start-up time. Therefore, one or more of the following conditions is possible after $\overline{\text{SYSRST}}$ is released:

- The oscillator circuit has not begun to oscillate.
- The Oscillator Start-up Timer has not expired (if a crystal oscillator is used).
- The PLL has not achieved a lock (if PLL is used).

The device will not begin to execute code until a valid clock source has been released to the system. Therefore, the oscillator and PLL start-up delays must be considered when the Reset delay time must be known.

#### 6.2.2 FAIL-SAFE CLOCK MONITOR (FSCM) AND DEVICE RESETS

If the FSCM is enabled, it will begin to monitor the system clock source when $\overline{\text{SYSRST}}$ is released. If a valid clock source is not available at this time, the device will automatically switch to the FRC Oscillator and the user can switch to the desired crystal oscillator in the Trap Service Routine (TSR).

## 6.3 Special Function Register Reset States

Most of the Special Function Registers (SFRs) associated with the PIC24F CPU and peripherals are reset to a particular value at a device Reset. The SFRs are grouped by their peripheral or CPU function and their Reset values are specified in each section of this manual.

The Reset value for each SFR does not depend on the type of Reset with the exception of four registers. The Reset value for the Reset Control register, RCON, will depend on the type of device Reset. The Reset value for the Oscillator Control register, OSCCON, will depend on the type of Reset and the programmed values of the FNOSC bits in Flash Configuration Word 2 (CW2); see Table 6-2. The RCFGCAL and NVMCON registers are only affected by a POR.

**REGISTER 7-20: IPC3: INTERRUPT PRIORITY CONTROL REGISTER 3**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | AD1IP2 | AD1IP1 | AD1IP0 | — | U1TXIP2 | U1TXIP1 | U1TXIP0 |
| 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 15-7 **Unimplemented:** Read as ‘0’

bit 6-4 **AD1IP<2:0>:** A/D Conversion Complete Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **U1TXIP<2:0>:** UART1 Transmitter Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

### REGISTER 7-37: IPC22: INTERRUPT PRIORITY CONTROL REGISTER 22

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | SPI3IP2 | SPI3IP1 | SPI3IP0 | — | SPF3IP2 | SPF3IP1 | SPF3IP0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | U4TXIP2 | U4TXIP1 | U4TXIP0 | — | U4RXIP2 | U4RXIP1 | U4RXIP0 |
| 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 15 **Unimplemented:** Read as ‘0’

bit 14-12 **SPI3IP<2:0>:** SPI3 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as ‘0’

bit 10-8 **SPF3IP<2:0>:** SPI3 Fault Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 7 **Unimplemented:** Read as ‘0’

bit 6-4 **U4TXIP<2:0>:** UART4 Transmitter Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **U4RXIP<2:0>:** UART4 Receiver Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

## 9.0 POWER-SAVING FEATURES

> **Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the *"PIC24F Family Reference Manual"*, **Section 10. "Power-Saving Features"** (DS39698).

The PIC24FJ256GA110 family of devices provides the ability to manage power consumption by selectively managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. All PIC24F devices manage power consumption in four different ways:

- Clock frequency
- Instruction-based Sleep and Idle modes
- Software controlled Doze mode
- Selective peripheral control in software

Combinations of these methods can be used to selectively tailor an application's power consumption, while still maintaining critical application features, such as timing-sensitive communications.

### 9.1 Clock Frequency and Clock Switching

PIC24F devices allow for a wide range of clock frequencies to be selected under application control. If the system clock configuration is not locked, users can choose low-power or high-precision oscillators by simply changing the NOSC bits. The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 8.0 "Oscillator Configuration"**.

### 9.2 Instruction-Based Power-Saving Modes

PIC24F devices have two special power-saving modes that are entered through the execution of a special `PWRSAV` instruction. Sleep mode stops clock operation and halts all code execution; Idle mode halts the CPU and code execution, but allows peripheral modules to continue operation. The assembly syntax of the `PWRSAV` instruction is shown in Example 9-1.

Sleep and Idle modes can be exited as a result of an enabled interrupt, WDT time-out or a device Reset. When the device exits these modes, it is said to "wake-up".

#### 9.2.1 SLEEP MODE

Sleep mode has these features:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption will be reduced to a minimum provided that no I/O pin is sourcing current.
- The Fail-Safe Clock Monitor does not operate during Sleep mode since the system clock source is disabled.
- The LPRC clock will continue to run in Sleep mode if the WDT is enabled.
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode.
- Some device features or peripherals may continue to operate in Sleep mode. This includes items such as the input change notification on the I/O ports, or peripherals that use an external clock input. Any peripheral that requires the system clock source for its operation will be disabled in Sleep mode.

Additional power reductions can be achieved by disabling the on-chip voltage regulator whenever Sleep mode is invoked. This is done by clearing the PMSLP bit (RCON<8>). Disabling the regulator adds an additional delay of about 190 μs to the device wake-up time. It is recommended that applications not using the voltage regulator leave the PMSLP bit set. For additional details on the regulator and Sleep mode, see **Section 25.2.5 "Voltage Regulator Standby Mode"**.

The device will wake-up from Sleep mode on any of these events:

- On any interrupt source that is individually enabled
- On any form of device Reset
- On a WDT time-out

On wake-up from Sleep, the processor will restart with the same clock source that was active when Sleep mode was entered.

**EXAMPLE 9-1: `PWRSAV` INSTRUCTION SYNTAX**

```
PWRSAV    #0              ; Put the device into SLEEP mode
PWRSAV    #1              ; Put the device into IDLE mode
```

### 10.4.5 CONSIDERATIONS FOR PERIPHERAL PIN SELECTION

The ability to control Peripheral Pin Select options introduces several considerations into application design that could be overlooked. This is particularly true for several common peripherals that are available only as remappable peripherals.

The main consideration is that the Peripheral Pin Selects are not available on default pins in the device's default (Reset) state. Since all RPINRx registers reset to '111111' and all RPORx registers reset to '000000', all Peripheral Pin Select inputs are tied to VSS and all Peripheral Pin Select outputs are disconnected.

| | |
|---|---|
| **Note:** | In tying Peripheral Pin Select inputs to RP63, RP63 does not have to exist on a device for the registers to be reset to it. |

This situation requires the user to initialize the device with the proper peripheral configuration before any other application code is executed. Since the IOLOCK bit resets in the unlocked state, it is not necessary to execute the unlock sequence after the device has come out of Reset. For application safety, however, it is best to set IOLOCK and lock the configuration after writing to the control registers.

Because the unlock sequence is timing critical, it must be executed as an assembly language routine in the same manner as changes to the oscillator configuration. If the bulk of the application is written in C or another high-level language, the unlock sequence should be performed by writing in-line assembly.

Choosing the configuration requires the review of all Peripheral Pin Selects and their pin assignments, especially those that will not be used in the application. In all cases, unused pin-selectable peripherals should be disabled completely. Unused peripherals should have their inputs assigned to an unused RPn pin function. I/O pins with unused RPn functions should be configured with the null peripheral output.

The assignment of a peripheral to a particular pin does not automatically perform any other configuration of the pin's I/O circuitry. In theory, this means adding a pin-selectable output to a pin may mean inadvertently driving an existing peripheral input when the output is driven. Users must be familiar with the behavior of other fixed peripherals that share a remappable pin and know when to enable or disable them. To be safe, fixed digital peripherals that share the same pin should be disabled when not in use.

Along these lines, configuring a remappable pin for a specific peripheral does not automatically turn that feature on. The peripheral must be specifically configured for operation and enabled, as if it were tied to a fixed pin. Where this happens in the application code (immediately following device Reset and peripheral configuration or inside the main application routine) depends on the peripheral and its use in the application.

A final consideration is that Peripheral Pin Select functions neither override analog inputs, nor reconfigure pins with analog functions for digital I/O. If a pin is configured as an analog input on device Reset, it must be explicitly reconfigured as digital I/O when used with a Peripheral Pin Select.

Example 10-2 shows a configuration for bidirectional communication with flow control using UART1. The following input and output functions are used:

- Input Functions: U1RX, $\overline{\text{U1CTS}}$
- Output Functions: U1TX, $\overline{\text{U1RTS}}$

**EXAMPLE 10-2: CONFIGURING UART1 INPUT AND OUTPUT FUNCTIONS**

```
// Unlock Registers
__builtin_write_OSCCONL(OSCCON & 0xBF);

// Configure Input Functions (Table 9-1))
    // Assign U1RX To Pin RP0
    RPINR18bits.U1RXR = 0;

    // Assign U1CTS To Pin RP1
    RPINR18bits.U1CTSR = 1;

// Configure Output Functions (Table 9-2)
    // Assign U1TX To Pin RP2
    RPOR1bits.RP2R = 3;

    // Assign U1RTS To Pin RP3
    RPOR1bits.RP3R = 4;

// Lock Registers
__builtin_write_OSCCONL(OSCCON | 0x40);
```

### 10.4.6 PERIPHERAL PIN SELECT REGISTERS

The PIC24FJ256GA110 family of devices implements a total of 37 registers for remappable peripheral configuration:

- Input Remappable Peripheral Registers (21)
- Output Remappable Peripheral Registers (16)

> **Note:** Input and output register values can only be changed if IOLOCK (OSCCON<6>) = 0. See **Section 10.4.4.1 "Control Register Lock"** for a specific command sequence.

**REGISTER 10-1: RPINR0: PERIPHERAL PIN SELECT INPUT REGISTER 0**

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | INT1R5 | INT1R4 | INT1R3 | INT1R2 | INT1R1 | INT1R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| 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 15-14 | **Unimplemented:** Read as '0' |
| bit 13-8 | **INT1R<5:0>:** Assign External Interrupt 1 (INT1) to Corresponding RPn or RPIn Pin bits |
| bit 7-0 | **Unimplemented:** Read as '0' |

**REGISTER 10-2: RPINR1: PERIPHERAL PIN SELECT INPUT REGISTER 1**

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | INT3R5 | INT3R4 | INT3R3 | INT3R2 | INT3R1 | INT3R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | INT2R5 | INT2R4 | INT2R3 | INT2R2 | INT2R1 | INT2R0 |
| 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 15-14 | **Unimplemented:** Read as '0' |
| bit 13-8 | **INT3R<5:0>:** Assign External Interrupt 3 (INT3) to Corresponding RPn or RPIn Pin bits |
| bit 7-6 | **Unimplemented:** Read as '0' |
| bit 5-0 | **INT2R<5:0>:** Assign External Interrupt 2 (INT2) to Corresponding RPn or RPIn Pin bits |

**REGISTER 10-5: RPINR4: PERIPHERAL PIN SELECT INPUT REGISTER 4**

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | T5CKR5 | T5CKR4 | T5CKR3 | T5CKR2 | T5CKR1 | T5CKR0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | T4CKR5 | T4CKR4 | T4CKR3 | T4CKR2 | T4CKR1 | T4CKR0 |
| 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 15-14 **Unimplemented:** Read as ‘0’
bit 13-8 **T5CKR<5:0>:** Assign Timer5 External Clock (T5CK) to Corresponding RPn or RPIn Pin bits
bit 7-6 **Unimplemented:** Read as ‘0’
bit 5-0 **T4CKR<5:0>:** Assign Timer4 External Clock (T4CK) to Corresponding RPn or RPIn Pin bits

**REGISTER 10-6: RPINR7: PERIPHERAL PIN SELECT INPUT REGISTER 7**

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | IC2R5 | IC2R4 | IC2R3 | IC2R2 | IC2R1 | IC2R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | IC1R5 | IC1R4 | IC1R3 | IC1R2 | IC1R1 | IC1R0 |
| 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 15-14 **Unimplemented:** Read as ‘0’
bit 13-8 **IC2R<5:0>:** Assign Input Capture 2 (IC2) to Corresponding RPn or RPIn Pin bits
bit 7-6 **Unimplemented:** Read as ‘0’
bit 5-0 **IC1R<5:0>:** Assign Input Capture 1 (IC1) to Corresponding RPn or RPIn Pin bits

### REGISTER 10-17: RPINR22: PERIPHERAL PIN SELECT INPUT REGISTER 22

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | SCK2R5 | SCK2R4 | SCK2R3 | SCK2R2 | SCK2R1 | SCK2R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | SDI2R5 | SDI2R4 | SDI2R3 | SDI2R2 | SDI2R1 | SDI2R0 |
| 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 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **SCK2R<5:0>:** Assign SPI2 Clock Input (SCK2IN) to Corresponding RPn or RPIn Pin bits

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **SDI2R<5:0>:** Assign SPI2 Data Input (SDI2) to Corresponding RPn or RPIn Pin bits

### REGISTER 10-18: RPINR23: PERIPHERAL PIN SELECT INPUT REGISTER 23

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | SS2R5 | SS2R4 | SS2R3 | SS2R2 | SS2R1 | SS2R0 |
| 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 15-6 **Unimplemented:** Read as ‘0’

bit 5-0 **SS2R<5:0>:** Assign SPI2 Slave Select Input (SS2IN) to Corresponding RPn or RPIn Pin bits

### REGISTER 10-19: RPINR27: PERIPHERAL PIN SELECT INPUT REGISTER 27

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | U4CTSR5 | U4CTSR4 | U4CTSR3 | U4CTSR2 | U4CTSR1 | U4CTSR0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | U4RXR5 | U4RXR4 | U4RXR3 | U4RXR2 | U4RXR1 | U4RXR0 |
| 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 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **U4CTSR<5:0>:** Assign UART4 Clear to Send ($\overline{\text{U4CTS}}$) to Corresponding RPn or RPIn Pin bits

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **U4RXR<5:0>:** Assign UART4 Receive (U4RX) to Corresponding RPn or RPIn Pin bits

### REGISTER 10-20: RPINR28: PERIPHERAL PIN SELECT INPUT REGISTER 28

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | SCK3R5 | SCK3R4 | SCK3R3 | SCK3R2 | SCK3R1 | SCK3R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | SDI3R5 | SDI3R4 | SDI3R3 | SDI3R2 | SDI3R1 | SDI3R0 |
| 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 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **SCK3R<5:0>:** Assign SPI3 Data Input (SCK3IN) to Corresponding RPn or RPIn Pin bits

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **SDI3R<5:0>:** Assign SPI3 Data Input (SDI3) to Corresponding RPn or RPIn Pin bits

### REGISTER 10-32: RPOR10: PERIPHERAL PIN SELECT OUTPUT REGISTER 10

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | RP21R5 | RP21R4 | RP21R3 | RP21R2 | RP21R1 | RP21R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | RP20R5 | RP20R4 | RP20R3 | RP20R2 | RP20R1 | RP20R0 |
| 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 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **RP21R<5:0>:** RP21 Output Pin Mapping bits
Peripheral output number n is assigned to pin, RP21 (see Table 10-3 for peripheral function numbers).

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **RP20R<5:0:>** RP20 Output Pin Mapping bits
Peripheral output number n is assigned to pin, RP20 (see Table 10-3 for peripheral function numbers).

### REGISTER 10-33: RPOR11: PERIPHERAL PIN SELECT OUTPUT REGISTER 11

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | RP23R5 | RP23R4 | RP23R3 | RP23R2 | RP23R1 | RP23R0 |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | RP22R5 | RP22R4 | RP22R3 | RP22R2 | RP22R1 | RP22R0 |
| 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 15-14 **Unimplemented:** Read as ‘0’

bit 13-8 **RP23R<5:0>:** RP23 Output Pin Mapping bits
Peripheral output number n is assigned to pin, RP23 (see Table 10-3 for peripheral function numbers).

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **RP22R<5:0>:** RP22 Output Pin Mapping bits
Peripheral output number n is assigned to pin, RP22 (see Table 10-3 for peripheral function numbers).

**NOTES:**

### REGISTER 14-1: OCxCON1: OUTPUT COMPARE x CONTROL 1 REGISTER

| U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | OCSIDL | OCTSEL2 | OCTSEL1 | OCTSEL0 | — | — |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | U-0 | U-0 | R/W-0, HCS | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ENFLT0 | — | — | OCFLT0 | TRIGMODE | OCM2(1) | OCM1(1) | OCM0(1) |
| bit 7 | | | | | | | bit 0 |

| Legend: | HCS = Hardware Clearable/Settable bit | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **OCSIDL:** Stop Output Compare x in Idle Mode Control bit
1 = Output Compare x halts in CPU Idle mode
0 = Output Compare x continues to operate in CPU Idle mode

bit 12-10 **OCTSEL<2:0>:** Output Compare x Timer Select bits
111 = Peripheral Clock ($F_{CY}$)
110 = Reserved
101 = Reserved
100 = Timer1
011 = Timer5
010 = Timer4
001 = Timer3
000 = Timer2

bit 9-8 **Unimplemented:** Read as '0'

bit 7 **ENFLT0:** Fault 0 Input Enable bit
1 = Fault 0 input is enabled
0 = Fault 0 input is disabled

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

bit 4 **OCFLT0:** PWM Fault Condition Status bit
1 = PWM Fault condition has occurred (cleared in HW only)
0 = No PWM Fault condition has occurred (this bit is only used when OCM<2:0> = 111)

bit 3 **TRIGMODE:** Trigger Status Mode Select bit
1 = TRIGSTAT (OCxCON2<6>) is cleared when OCxRS = OCxTMR or in software
0 = TRIGSTAT is only cleared by software

bit 2-0 **OCM<2:0>:** Output Compare x Mode Select bits(1)
111 = Center-Aligned PWM mode on OCx(2)
110 = Edge-Aligned PWM mode on OCx(2)
101 = Double Compare Continuous Pulse mode: initialize OCx pin low, toggle OCx state continuously on alternate matches of OCxR and OCxRS
100 = Double Compare Single-Shot mode: initialize OCx pin low, toggle OCx state on matches of OCxR and OCxRS for one cycle
011 = Single Compare Continuous Pulse mode: compare events continuously toggle OCx pin
010 = Single Compare Single-Shot mode: initialize OCx pin high, compare event forces OCx pin low
001 = Single Compare Single-Shot mode: initialize OCx pin low, compare event forces OCx pin high
000 = Output compare channel is disabled

**Note 1:** The OCx output must also be configured to an available RPn pin. For more information, see **Section 10.4 "Peripheral Pin Select"**.

**2:** OCFA pin controls OC1-OC4 channels; OCFB pin controls the OC5-OC9 channels. OCxR and OCxRS are double-buffered only in PWM modes.

**FIGURE 18-2: LEGACY PARALLEL SLAVE PORT EXAMPLE**

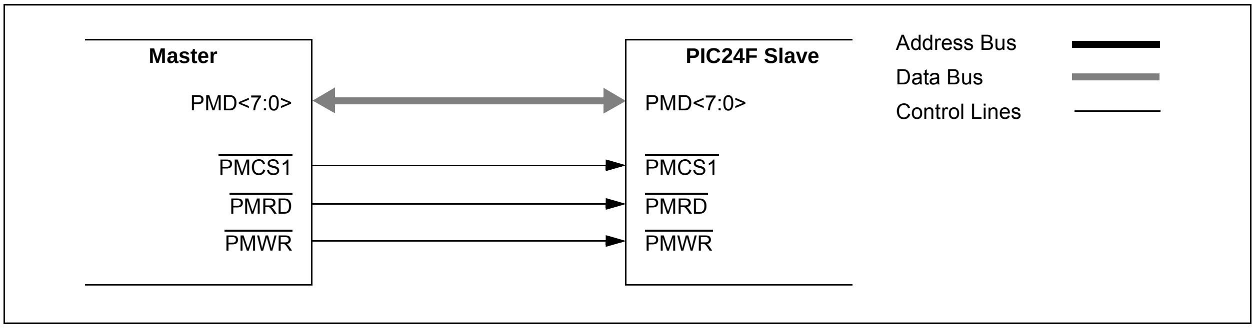



**FIGURE 18-3: ADDRESSABLE PARALLEL SLAVE PORT EXAMPLE**

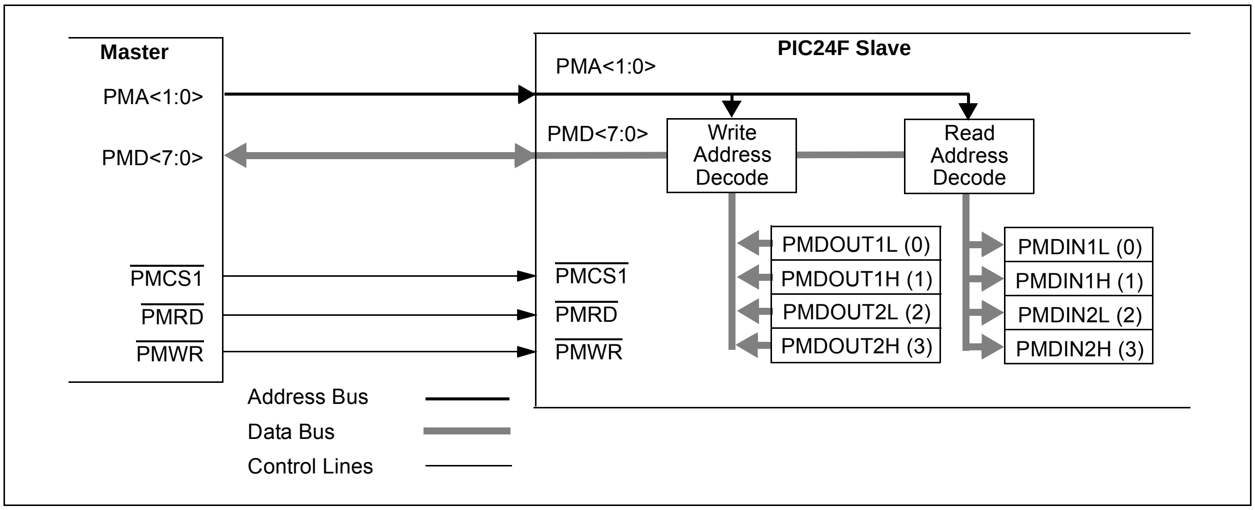
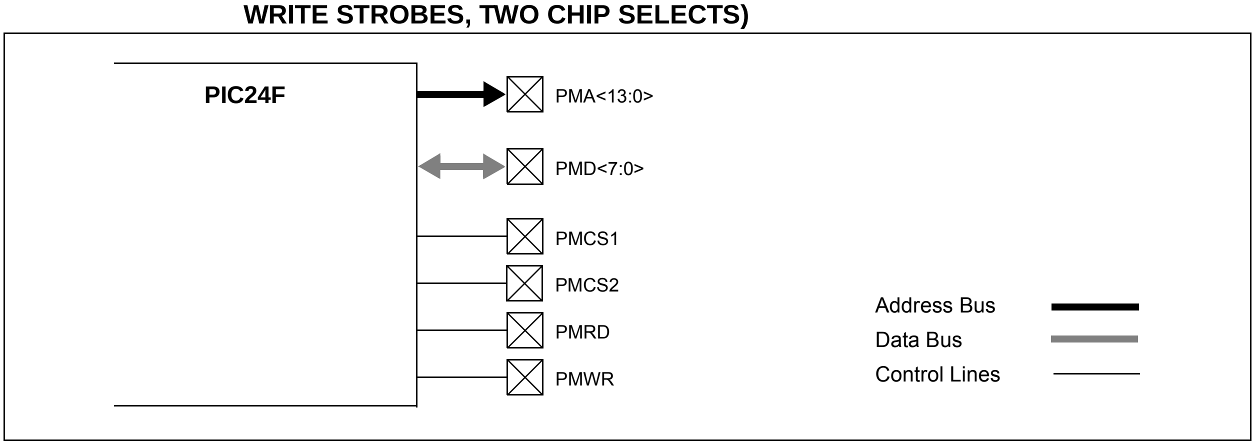



**TABLE 18-1: SLAVE MODE ADDRESS RESOLUTION**

| PMA<1:0> | Output Register (Buffer) | Input Register (Buffer) |
|---|---|---|
| 00 | PMDOUT1<7:0> (0) | PMDIN1<7:0> (0) |
| 01 | PMDOUT1<15:8> (1) | PMDIN1<15:8> (1) |
| 10 | PMDOUT2<7:0> (2) | PMDIN2<7:0> (2) |
| 11 | PMDOUT2<15:8> (3) | PMDIN2<15:8> (3) |

**FIGURE 18-4: MASTER MODE, DEMULTIPLEXED ADDRESSING (SEPARATE READ AND WRITE STROBES, TWO CHIP SELECTS)**



## 27.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

## 27.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

## 27.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

## 27.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

## 27.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

## 27.7 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC® DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB C Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

## 27.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

## 27.9 MPLAB ICD 3 In-Circuit Debugger System

MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost effective high-speed hardware debugger/programmer for Microchip Flash Digital Signal Controller (DSC) and microcontroller (MCU) devices. It debugs and programs PIC® Flash microcontrollers and dsPIC® DSCs with the powerful, yet easy-to-use graphical user interface of MPLAB Integrated Development Environment (IDE).

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

## 27.10 PICkit 3 In-Circuit Debugger/ Programmer and PICkit 3 Debug Express

The MPLAB PICkit 3 allows debugging and programming of PIC® and dsPIC® Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB Integrated Development Environment (IDE). The MPLAB PICkit 3 is connected to the design engineer's PC using a full speed USB interface and can be connected to the target via an Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the reset line to implement in-circuit debugging and In-Circuit Serial Programming™.

The PICkit 3 Debug Express include the PICkit 3, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

**FIGURE 28-5: EXTERNAL CLOCK TIMING**

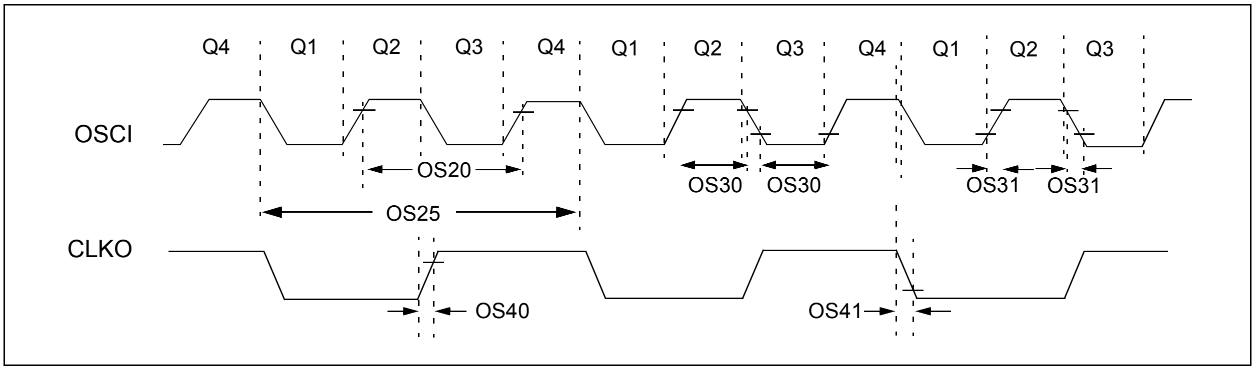



**TABLE 28-13: EXTERNAL CLOCK TIMING REQUIREMENTS**

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.50 to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial<br>-40°C ≤ TA ≤ +125°C for Extended | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym** | **Characteristic** | **Min** | **Typ(1)** | **Max** | **Units** | **Conditions** |
| OS10 | FOSC | External CLKI Frequency (external clocks allowed only in EC mode) | DC<br>4 | —<br>— | 32<br>8 | MHz<br>MHz | EC<br>ECPLL |
| | | Oscillator Frequency | 3<br>4<br>10<br>31 | —<br>—<br>—<br>— | 10<br>8<br>32<br>33 | MHz<br>MHz<br>MHz<br>kHz | XT<br>XTPLL<br>HS<br>SOSC |
| OS20 | TOSC | TOSC = 1/FOSC | — | — | — | — | See Parameter OS10 for FOSC value |
| OS25 | TCY | Instruction Cycle Time(2) | 62.5 | — | DC | ns | |
| OS30 | TosL, TosH | External Clock in (OSCI) High or Low Time | 0.45 x TOSC | — | — | ns | EC |
| OS31 | TosR, TosF | External Clock in (OSCI) Rise or Fall Time | — | — | 20 | ns | EC |
| OS40 | TckR | CLKO Rise Time(3) | — | 6 | 10 | ns | |
| OS41 | TckF | CLKO Fall Time(3) | — | 6 | 10 | ns | |

**Note 1:** Data in “Typ” column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**2:** Instruction cycle period (TCY) equals two times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “Min.” values with an external clock applied to the OSCI/CLKI pin. When an external clock input is used, the “Max.” cycle time limit is “DC” (no clock) for all devices.

**3:** Measurements are taken in EC mode. The CLKO signal is measured on the OSCO pin. CLKO is low for the Q1-Q2 period (1/2 TCY) and high for the Q3-Q4 period (1/2 TCY).