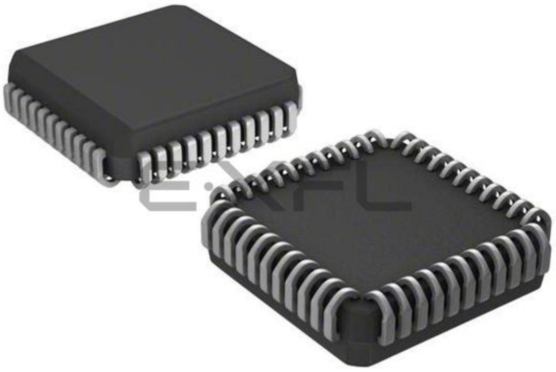


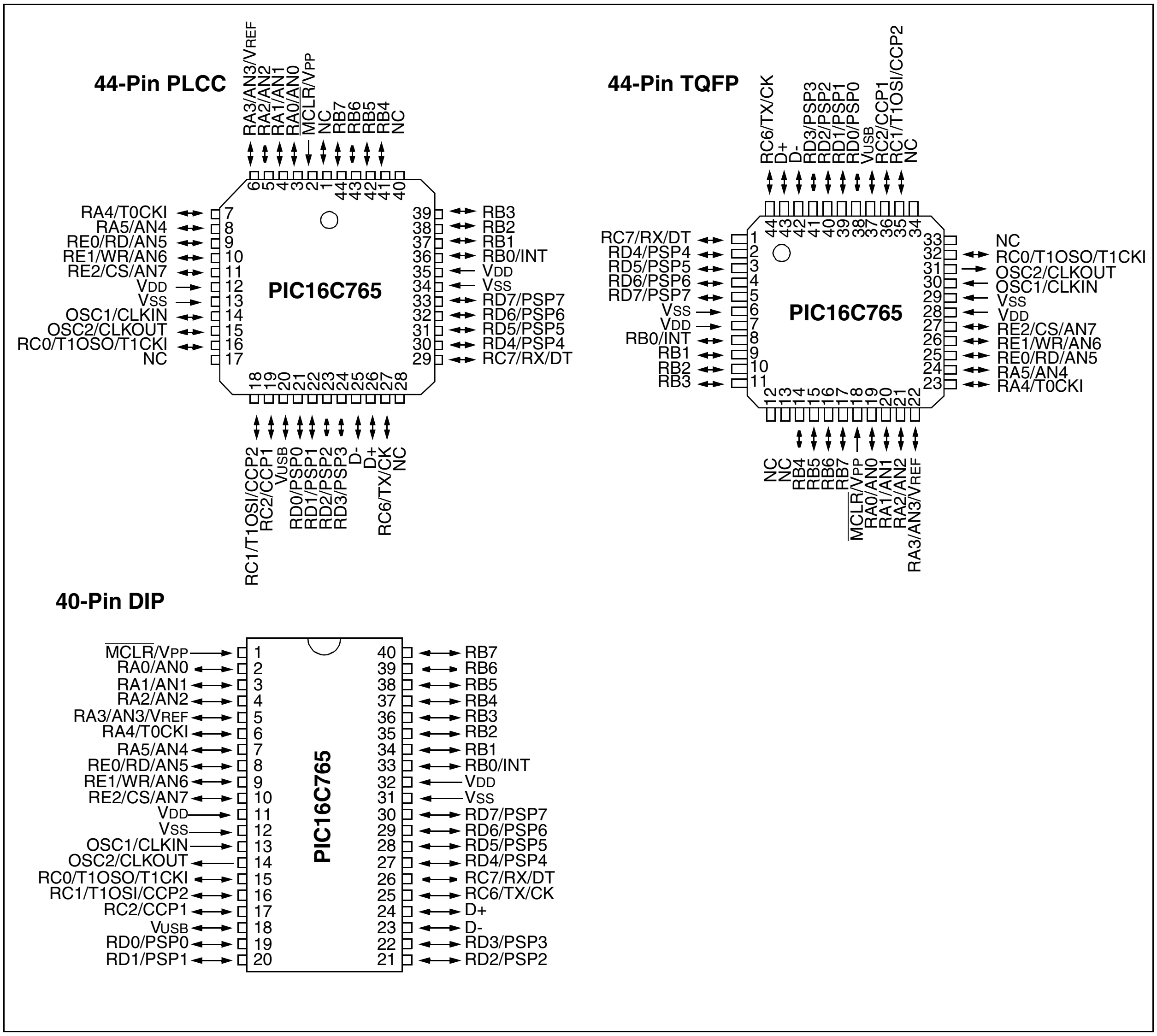
Welcome to E-XFL.COM

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	24MHz
Connectivity	SCI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	33
Program Memory Size	14KB (8K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	4.35V ~ 5.25V
Data Converters	A/D 8x8b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.59x16.59)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16c765-i-l



Key Features PICmicro™ Mid-Range Reference Manual (DS33023)	PIC16C745	PIC16C765
Operating Frequency	6 MHz or 24 MHz	6 MHz or 24 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Program Memory (14-bit words)	8K	8K
Data Memory (bytes)	256	256
Dual Port Ram	64	64
Interrupt Sources	11	12
I/O Ports	22 (Ports A, B, C)	33 (Ports A, B, C, D, E)
Timers	3	3
Capture/Compare/PWM modules	2	2
Analog-to-Digital Converter Module	5 channel x 8 bit	8 channel x 8 bit
Parallel Slave Port	—	Yes
Serial Communication	USB, USART/SCI	USB, USART/SCI
Brown-out Detect Reset	Yes	Yes

NOTES:

 Preliminary

1.0 GENERAL DESCRIPTION

The PIC16C745/765 devices are low cost, high-performance, CMOS, fully-static, 8-bit microcontrollers in the PIC16CXX mid-range family.

All PIC® microcontrollers employ an advanced RISC architecture. The PIC16C745/765 microcontroller family has enhanced core features, eight-level deep stack and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with the separate 8-bit wide data. The two stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches, which require two cycles. A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance.

The PIC16C745 device has 22 I/O pins. The PIC16C765 device has 33 I/O pins. Each device has 256 bytes of RAM. In addition, several peripheral features are available including: three timer/counters, two Capture/Compare/PWM modules and two serial ports. The Universal Serial Bus (USB 1.1) low speed peripheral provides bus communications. The Universal Synchronous Asynchronous Receiver Transmitter (USART) is also known as the Serial Communications Interface or SCI. Also, a 5-channel high-speed 8-bit A/D is provided on the PIC16C745, while the PIC16C765 offers 8 channels. The 8-bit resolution is ideally suited for applications requiring a low cost analog interface (e.g., thermostat control, pressure sensing, etc.).

The PIC16C745/765 devices have special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. There are 4 oscillator options, of which EC is for the external regulated clock source, E4 is for the external regulated clock source with the PLL enabled, HS is for the high speed crystals/resonators and H4 is for high speed crystals/resonators with the PLL enabled. The SLEEP (power-down) feature provides a power-saving mode. The user can wake-up the chip from SLEEP through several external and internal interrupts and RESETS.

A highly reliable Watchdog Timer (WDT), with a dedicated on-chip RC oscillator, provides protection against software lock-up, and also provides one way of waking the device from SLEEP.

A UV erasable CERDIP packaged version is ideal for code development, while the cost-effective One-Time-Programmable (OTP) version is suitable for production in any volume.

The PIC16C745/765 devices fit nicely in many applications ranging from security and remote sensors to appliance controls and automotives. The EPROM technology makes customization of application programs (data loggers, industrial controls, UPS) extremely fast and convenient. The small footprint packages make this microcontroller series perfect for all applications with space limitations. Low-cost, low-power, high-performance, ease of use and I/O flexibility make the PIC16C745/765 devices very versatile, even in areas where no microcontroller use has been considered before (e.g., timer functions, serial communication, capture and compare, PWM functions and coprocessor applications).

1.1 Family and Upward Compatibility

Users familiar with the PIC16C5X microcontroller family will realize that this is an enhanced version of the PIC16C5X architecture. Code written for the PIC16C5X can be easily ported to the PIC16C745/765 family of devices.

1.2 Development Support

PIC® devices are supported by the complete line of Microchip Development tools.

Please refer to Section 15.0 for more details about Microchip's development tools.

TABLE 4-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other resets (2)
Bank 3											
180h	INDF(3)	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	0000 0000
181h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
182h	PCL(3)	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
183h	STATUS(3)	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	000q quuu
184h	FSR(3)	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
185h	—	Unimplemented								—	—
186h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
187h	—	Unimplemented								—	—
188h	—	Unimplemented								—	—
189h	—	Unimplemented								—	—
18Ah	PCLATH(1,3)	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	---0 0000
18Bh	INTCON(3)	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
18Ch-18Fh	—	Unimplemented								—	—
190h	UIR	—	—	STALL	UIDLE	TOK_DNE	ACTIVITY	UERR	USB_RST	--00 0000	--00 0000
191h	UIE	—	—	STALL	UIDLE	TOK_DNE	ACTIVITY	UERR	USB_RST	--00 0000	--00 0000
192h	UEIR	BTS_ERR	OWN_ERR	WRT_ERR	BTO_ERR	DFN8	CRC16	CRC5	PID_ERR	0000 0000	0000 0000
193h	UEIE	BTS_ERR	OWN_ERR	WRT_ERR	BTO_ERR	DFN8	CRC16	CRC5	PID_ERR	0000 0000	0000 0000
194h	USTAT	—	—	—	ENDP1	ENDP0	IN	—	—	---x xx--	---u uu--
195h	UCTRL	—	—	SEO	PKT_DIS	DEV_ATT	RESUME	SUSPND	—	--x0 000-	--xq qqq-
196h	UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	-000 0000	-000 0000
197h	USWSTAT	SWSTAT7	SWSTAT6	SWSTAT5	SWSTAT4	SWSTAT3	SWSTAT2	SWSTAT1	SWSTAT0	0000 0000	0000 0000
198h	UEP0	—	—	—	—	EP_CTL_DIS	EP_OUT_EN	EP_IN_EN	EP_STALL	---- 0000	---- 0000
199h	UEP1	—	—	—	—	EP_CTL_DIS	EP_OUT_EN	EP_IN_EN	EP_STALL	---- 0000	---- 0000
19Ah	UEP2	—	—	—	—	EP_CTL_DIS	EP_OUT_EN	EP_IN_EN	EP_STALL	---- 0000	---- 0000
19Bh-19Fh	Reserved	Reserved, do not use.								0000 0000	0000 0000

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

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8> whose contents are transferred to the upper byte of the program counter.

2: Other (non power-up) RESETS include external RESET through $\overline{\text{MCLR}}$ and Watchdog Timer Reset.

3: These registers can be addressed from any bank.

4: The Parallel Slave Port (PORTD and PORTE) is not implemented on the PIC16C745, always maintain these bits clear.

4.5 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

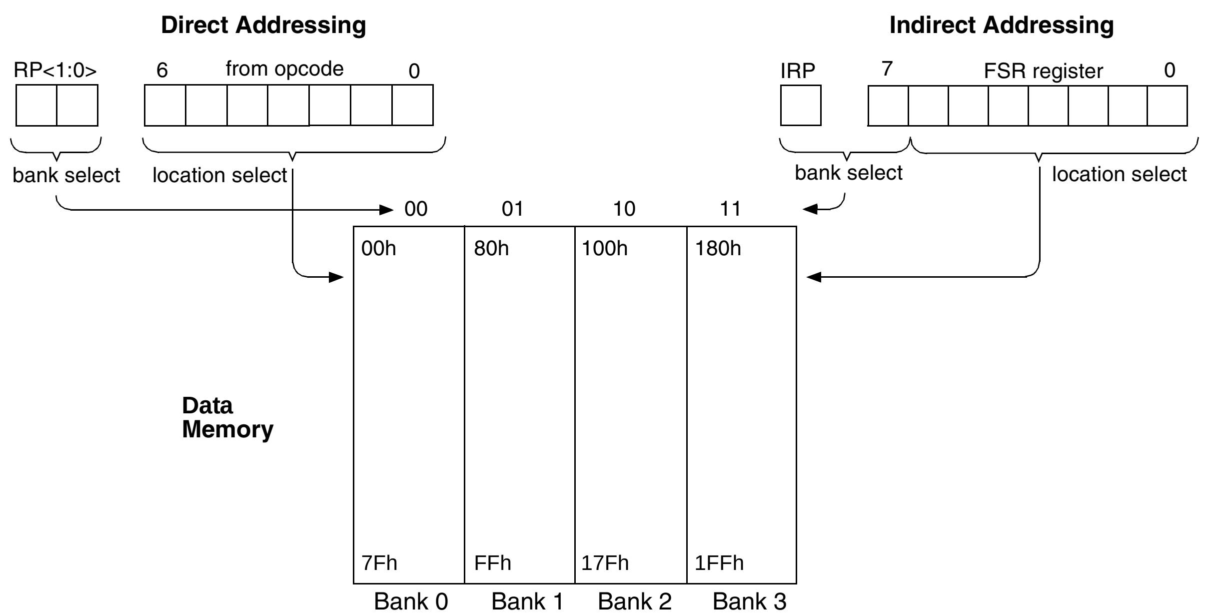
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself indirectly (FSR = '0') will read 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-4.

A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 4-2.

EXAMPLE 4-2: INDIRECT ADDRESSING

```
            movlw   0x20    ;initialize pointer
            movwf   FSR     ;to RAM
NEXT        clrf    INDF    ;clear INDF register
            incf    FSR,F   ;inc pointer
            btfss   FSR,4   ;all done?
            goto    NEXT    ;no clear next
CONTINUE
            :               ;yes continue
```

FIGURE 4-4: DIRECT/INDIRECT ADDRESSING



Note: For register file map detail see Figure 4-2.

 Preliminary

TABLE 5-1: PORTA FUNCTIONS

Name	Function	Input Type	Output Type	Description
RA0/AN0	RA0	ST	CMOS	Bi-directional I/O
	AN0	AN	—	A/D Input
RA1/AN1	RA1	ST	CMOS	Bi-directional I/O
	AN1	AN	—	A/D Input
RA2/AN2	RA2	ST	CMOS	Bi-directional I/O
	AN2	AN	—	A/D Input
RA3/AN3/VREF	RA3	ST	CMOS	Bi-directional I/O
	AN3	AN	—	A/D Input
	VREF	AN	—	A/D Positive Reference
RA4/T0CKI	RA4	ST	OD	Bi-directional I/O
	T0CKI	ST	—	Timer 0 Clock Input
RA5/AN4	RA5	ST		Bi-directional I/O
	AN4	AN	—	A/D Input

Legend: OD = open drain, ST = Schmitt Trigger

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other resets
05h	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	`--0x 0000`	`--0u 0000`
85h	TRISA	—	—	PORTA Data Direction Register						`--11 1111`	`--11 1111`
9Fh	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	`---- -000`	`---- -000`

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

6.0 TIMER0 MODULE

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt-on-overflow from FFh to 00h
- Edge select for external clock

Figure 6-1 is a block diagram of the Timer0 module and the prescaler shared with the WDT.

Additional information on the Timer0 module is available in the PIC Mid-Range MCU Family Reference Manual (DS33023).

Timer mode is selected by clearing bit T0CS (OPTION_REG<5>). In timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

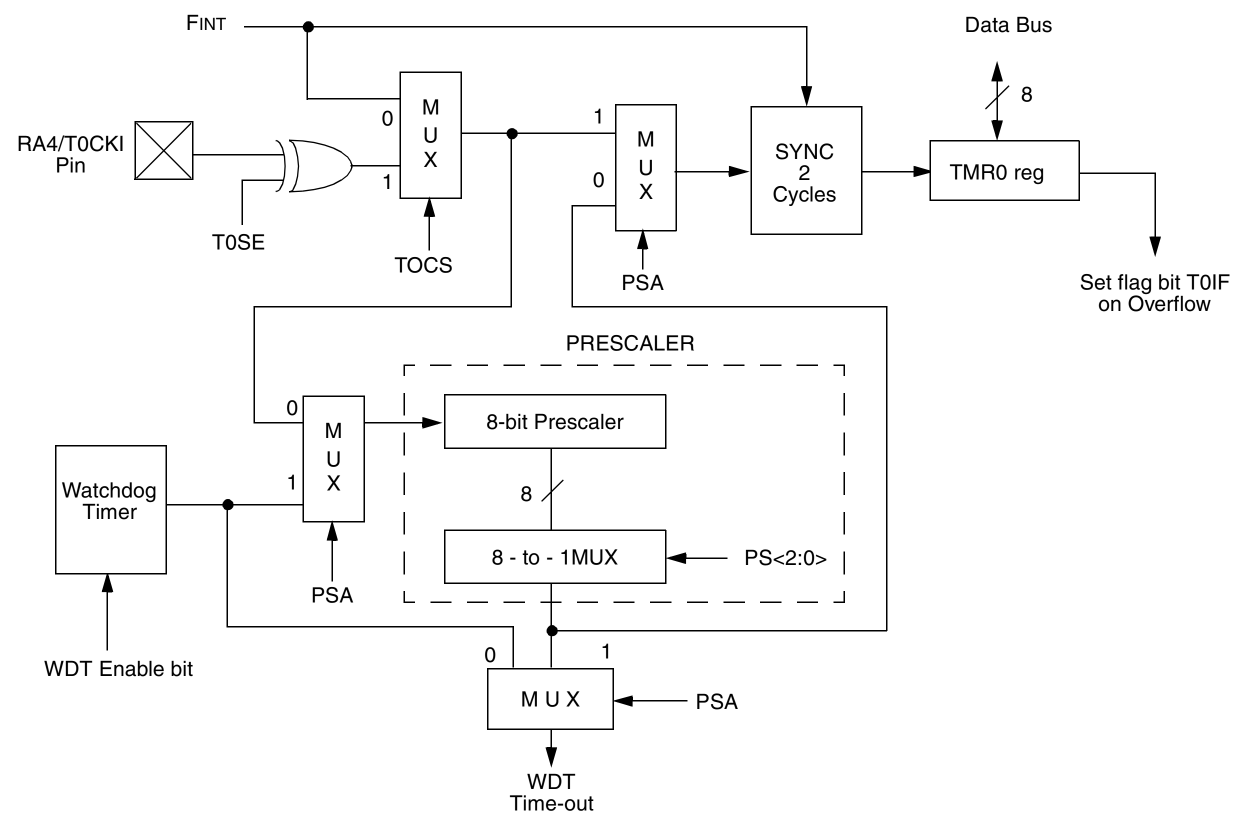
Counter mode is selected by setting bit T0CS (OPTION_REG<5>). In counter mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (OPTION_REG<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.2.

The prescaler is mutually exclusively shared between the Timer0 module and the watchdog timer. The prescaler is not readable or writable. Section 6.3 details the operation of the prescaler.

6.1 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit T0IF (INTCON<2>). The interrupt can be masked by clearing bit T0IE (INTCON<5>). Bit T0IF must be cleared in software by the Timer0 module interrupt service routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from SLEEP, since the timer is shut off during SLEEP.

FIGURE 6-1: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



Note: T0CS, T0SE, PSA, PS<2:0> are (OPTION_REG<5:0>).

9.0 CAPTURE/COMPARE/PWM MODULES

Each Capture/Compare/PWM (CCP) module contains a 16-bit register which can operate as a:

- 16-bit capture register
- 16-bit compare register
- PWM master/slave Duty Cycle register

Both the CCP1 and CCP2 modules are identical in operation, with the exception being the operation of the special event trigger. Table 9-1 and Table 9-2 show the resources and interactions of the CCP module(s). In the following sections, the operation of a CCP module is described with respect to CCP1. CCP2 operates the same as CCP1, except where noted.

CCP1 Module:

Capture/Compare/PWM Register1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. The special event trigger is generated by a compare match and will reset Timer1.

CCP2 Module:

Capture/Compare/PWM Register1 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. The special event trigger is generated by a compare match and will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Additional information on CCP modules is available in the PIC Mid-Range MCU Family Reference Manual (DS33023) and in “Using the CCP Modules” (AN594).

TABLE 9-1: CCP MODE - TIMER RESOURCES REQUIRED

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

TABLE 9-2: INTERACTION OF TWO CCP MODULES

CCPx Mode	CCPy Mode	Interaction
Capture	Capture	Same TMR1 time-base.
Capture	Compare	The compare should be configured for the special event trigger, which clears TMR1.
Compare	Compare	The compare(s) should be configured for the special event trigger, which clears TMR1.
PWM	PWM	The PWMs will have the same frequency and update rate (TMR2 interrupt).
PWM	Capture	None.
PWM	Compare	None.

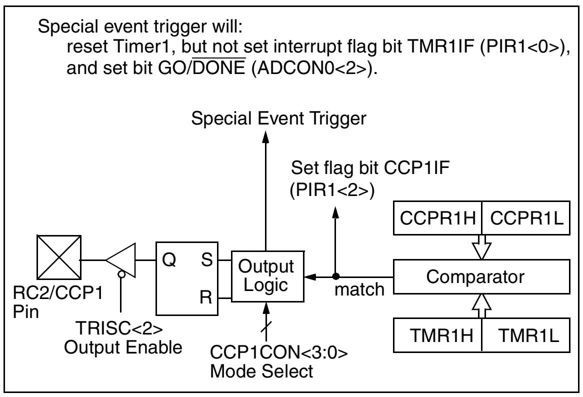
9.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the RC2/CCP1 pin is:

- Driven high
- Driven low
- Remains unchanged

The action on the pin is based on the value of control bits CCP1M<3:0> (CCP1CON<3:0>). At the same time, interrupt flag bit CCP1IF is set.

FIGURE 9-2: COMPARE MODE OPERATION BLOCK DIAGRAM



9.2.1 CCP PIN CONFIGURATION

The user must configure the RC2/CCP1 pin as an output by clearing the TRISC<2> bit.

> **Note:** Clearing the CCP1CON register will force the RC2/CCP1 compare output latch to the default low level. This is not the data latch.

9.2.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

9.2.3 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt mode is chosen, the CCP1 pin is not affected. The CCPIF bit is set causing a CCP interrupt (if enabled).

9.2.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated, which may be used to initiate an action.

The special event trigger output of CCP1 resets the TMR1 register pair. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

The special event trigger output of CCP2 starts an A/D conversion (if the A/D module is on) and resets the TMR1 register pair and starts an A/D conversion (if the A/D module is enabled).

> **Note:** The special event trigger from the CCP1and CCP2 modules will not set interrupt flag bit TMR1IF (PIR1<0>).

9.3 PWM Mode (PWM)

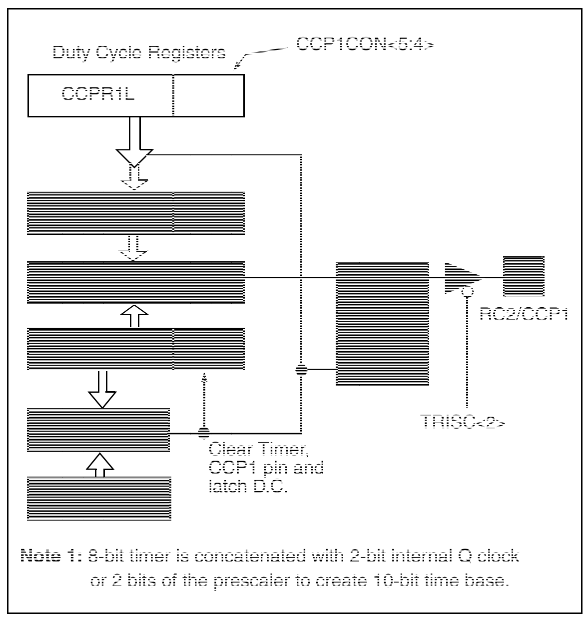
In pulse width modulation mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

> **Note:** Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 9-3 shows a simplified block diagram of the CCP module in PWM mode.

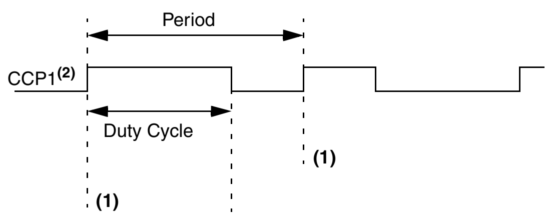
For a step by step procedure on how to set up the CCP module for PWM operation, see Section 9.3.3.

FIGURE 9-3: SIMPLIFIED PWM BLOCK DIAGRAM



A PWM output (Figure 9-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 9-4: PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.
2: Output signal is shown as asserted high.

9.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

$$\text{PWM period} = [(\text{PR2}) + 1] \bullet 4 \bullet \text{TOSC} \bullet (\text{TMR2 prescale value})$$

PWM frequency is defined as 1 / [PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see Section 8.1) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

9.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \bullet \text{Tosc} \bullet (\text{TMR2 prescale value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2 concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

Maximum PWM resolution (bits) for a given PWM frequency:

$$\text{Resolution} = \frac{\log\left(\frac{F_{INT}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

9.3.3 SET-UP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

10.0 UNIVERSAL SERIAL BUS

10.1 Overview

This section introduces a minimum amount of information on USB. If you already have basic knowledge of USB, you can safely skip this section. If terms like Enumeration, Endpoint, IN/OUT Transactions, Transfers and Low Speed/Full Speed are foreign to you, read on.

USB was developed to address the increased connectivity needs of PC's in the PC 2000 specification. There was a base requirement to increase the bandwidth and number of devices, which could be attached. Also desired were the ability for hot swapping, user friendly operation, robust communications and low cost. The primary promoters of USB are Intel, Compaq, Microsoft and NEC.

USB is implemented as a Tiered Star topology, with the host at the top, hubs in the middle, spreading out to the individual devices at the end. USB is limited to 127 devices on the bus, and the tree cannot be more than 6 levels deep.

USB is a host centric architecture. The host is always the master. Devices are not allowed to "speak" unless "spoken to" by the host.

Transfers take place at one of two speeds. Full Speed is 12 Mb/s and Low Speed is 1.5 Mb/s. Full Speed covers the middle ground of data intensive audio and compressed video applications, while low speed supports less data intensive applications.

10.1.1 TRANSFER PROTOCOLS

Full speed supports four transfer types: Isochronous, Bulk, Interrupt and Control. Low speed supports two transfer types: Interrupt and Control. The four transfer types are described below.

- **Isochronous Transfers**, meaning equal time, guarantee a fixed amount of data at a fixed rate. This mode trades off guaranteed data accuracy for guaranteed timeliness. Data validity is not checked because there isn't time to re-send bad packets anyway and the consequences of bad data are not catastrophic.
- **Bulk Transfers** are the converse of Isochronous. Data accuracy is guaranteed, but timeliness is not.
- **Interrupt Transfers** are designed to communicate with devices which have a moderate data rate requirement. Human Interface Devices like keyboards are but one example. For Interrupt Transfers, the key is the desire to transfer data at regular intervals. USB periodically polls these devices at a fixed rate to see if there is data to transfer.
- **Control Transfers** are used for configuration purposes.

10.1.2 FRAMES

Information communicated on the bus is grouped in a format called Frames. Each Frame is 1 ms in duration and is composed of multiple transfers. Each transfer type can be repeated more than once within a frame.

10.1.3 POWER

Power has always been a concern with any device. With USB, 5 volt power is now available directly from the bus. Devices may be self-powered or bus-powered. Self-powered devices will draw power from a wall adapter or power brick. On the other hand, bus-powered devices will draw power directly from the USB bus itself. There are limits to how much power can be drawn from the USB bus. Power is expressed in terms of "unit loads" ($\leq$100 mA). All devices, including Hubs, are guaranteed at least 1 unit load (low power), but must negotiate with the host for up to 5 unit loads (high power). If the host determines that the bus as currently configured cannot support a device's request for more unit loads, the device will be denied the extra unit loads and must remain in a low power configuration.

10.1.4 END POINTS

At the lowest level, each device controls one or more endpoints. An endpoint can be thought of as a virtual port. Endpoints are used to communicate with a device's functions. Each endpoint is a source or sink of data. Endpoints have both an In and Out direction associated with it. Each device must implement endpoint 0 to support Control Transfers for configuration. There are a maximum of 15 endpoints available for use by each full speed device and 6 endpoints for each slow speed device. Remember that the bus is host centric, so In/Out is with respect to the host and not the device.

10.1.5 ENUMERATION

Prior to communicating on the bus, the host must see that a new device has been connected and then go through an "enumeration process". This process allows the host to ask the device to introduce itself, and negotiate performance parameters, such as power consumption, transfer protocol and polling rate. The enumeration process is initiated by the host when it detects that a new device has attached itself to the bus. This takes place completely in the background from the application process.

10.1.6 DESCRIPTORS

The USB specification requires a number of different descriptors to provide information necessary to identify a device, specify its endpoints, and each endpoint's function. The five general categories of descriptors are Device, Configuration, Interface, End Point and String.

10.5 USB Register Map

The USB Control Registers, Buffer Descriptors and Buffers are located in Bank 3.

10.5.1 CONTROL AND STATUS REGISTERS

The USB module is controlled by 7 registers, plus those that control each endpoint and endpoint/direction buffer.

10.5.1.1 USB Interrupt Register (UIR)

The USB Interrupt Status Register (UIR) contains flag bits for each of the interrupt sources within the USB. Each of these bits are qualified with their respective interrupt enable bits (see the Interrupt Enable Register UIE). All bits of the register are logically OR'ed together to form a single interrupt source for the microprocessor interrupt found in PIR1 (USBIF). Once an interrupt bit has been set, it must be cleared by writing a zero.

REGISTER 10-1: USB INTERRUPT FLAGS REGISTER (UIR: 190h)

U-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
—	—	STALL	UIDLE	TOK_DNE	ACTIVITY	UERR	USB_RST
bit7							bit0

R = Readable bit
C = Clearable bit
U = Unimplemented bit, read as ‘0’
-n = Value at POR reset

bit 7-6: **Unimplemented:** Read as '0'

bit 5: **STALL:** A STALL handshake was sent by the SIE

bit 4: **UIDLE:** This bit is set if the USB has detected a constant idle on the USB bus signals for 3 ms. The idle timer is reset by activity on the USB bus. Once a IDLE condition has been detected, the user may wish to place the USB module in SUSPEND by setting the SUSPEND bit in the UCTRL register.

bit 3: **TOK_DNE:** This bit is set when the current token being processed is complete. The microprocessor should immediately read the USTAT register to determine the Endpoint number and direction used for this token. Clearing this bit causes the USTAT register to be cleared or the USTAT holding register to be loaded into the STAT register if another token has been processed.

bit 2: **ACTIVITY:** Activity on the D+/D- lines will cause the SIE to set this bit. Typically this bit is unmasked following detection of SLEEP. Users must enable the activity interrupt in the USB Interrupt Register (UIE: 191h) prior to entering suspend.

bit 1: **UERR:** This bit is set when any of the error conditions within the ERR_STAT register has occurred. The MCU must then read the ERR_STAT register to determine the source of the error.

bit 0: **USB_RST:** This bit is set when the USB has decoded a valid USB Reset. This will inform the MCU to write 00h into the address register and enable endpoint 0. USB_RST is set once a USB Reset has been detected for 2.5 microseconds. It will not be asserted again until the USB Reset condition has been removed, and then reasserted.

Note 1: Bits can only be modified when UCTRL.SUSPND = 0.

11.3 USART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner, i.e., transmission and reception do not occur at the same time. When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the RC6/TX/CK and RC7/RX/DT I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTA<7>).

11.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 11-1. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer register TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one Tcycle), the TXREG is empty and interrupt bit TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit TRMT (TXSTA<1>) shows the status of the TSR register. TRMT is a read only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the CK line. Data out is stable around the falling edge of the synchronous clock (Figure 11-6). The transmission can also be started by first loading the TXREG register and then setting bit TXEN (Figure 11-7). This is advantageous when slow baud rates are selected, since the BRG is kept in RESET when bits TXEN, CREN and SREN are clear. Setting enable bit TXEN will start the BRG, creating a shift clock immediately. Normally, when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing enable bit TXEN, during a transmission, will cause the transmission to be aborted and will reset the transmitter. The DT and CK pins will revert to hi-impedance. If either bit CREN or bit SREN is set during a transmission, the transmission is aborted and the DT pin reverts to a hi-impedance state (for a reception). The CK pin will remain an output if bit CSRC is set (internal clock). The transmitter logic, however, is not reset, although it is disconnected from the pins. In order to reset the transmitter, the user has to clear bit TXEN. If bit SREN is set (to interrupt an on-going transmission and receive a single word), then after the single word is received, bit SREN will be cleared and the serial port will revert back to transmitting, since bit TXEN is still set. The DT line will immediately switch from hi-impedance receive mode to transmit and start driving. To avoid this, bit TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit should be set and the ninth bit should be written to bit TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG can result in an immediate transfer of the data to the TSR register (if the TSR is empty). If the TSR was empty and the TXREG was written before writing the “new” TX9D, the “present” value of bit TX9D is loaded.

Steps to follow when setting up a Synchronous Master Transmission:

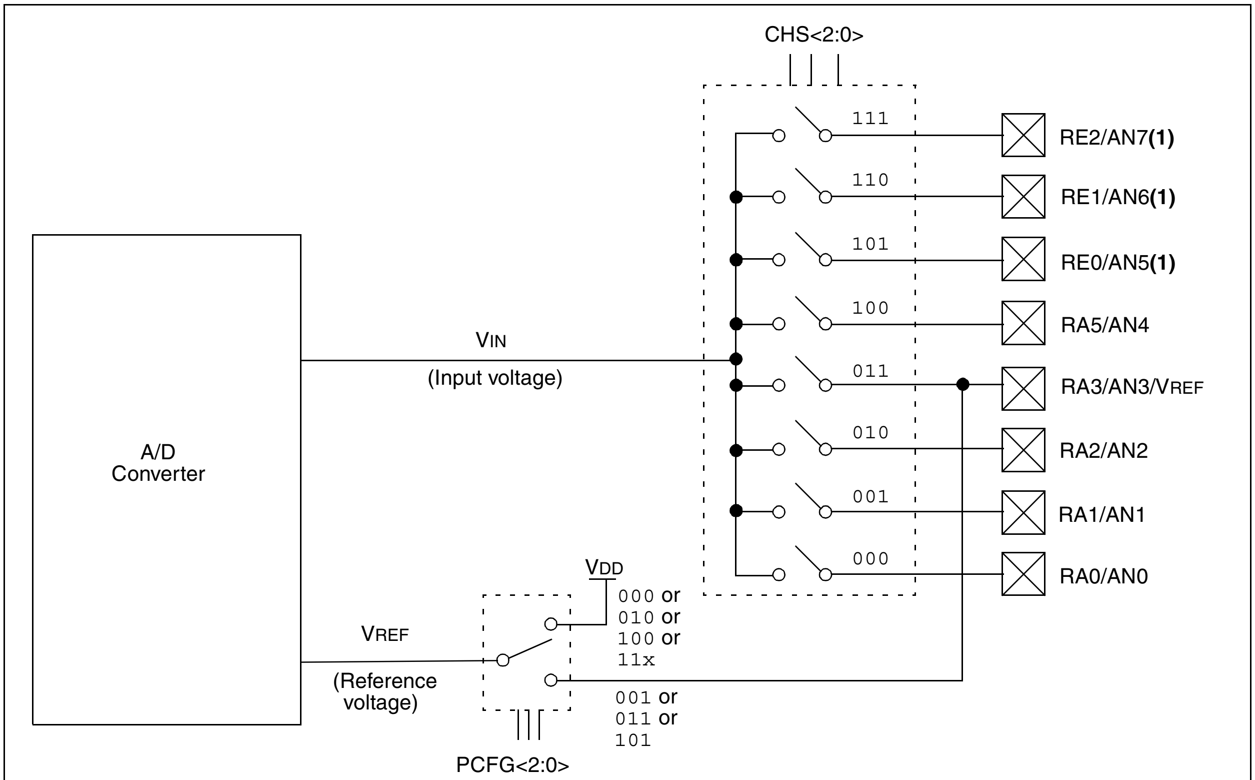
1. Initialize the SPBRG register for the appropriate baud rate (Section 11.1).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.

The following steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
   - Configure analog pins / voltage reference / and digital I/O (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D conversion clock (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR
   - Waiting for the A/D interrupt
6. Read A/D result register (ADRES), clear bit ADIF if required.
7. For next conversion, go to step 1 or step 2 as required. The A/D conversion time per bit is defined as T_{AD}. A minimum wait of $2T_{AD}$ is required before next acquisition starts.

FIGURE 12-1: A/D BLOCK DIAGRAM



Note 1: Not available on PIC16C745.

13.5 Time-out in Various Situations

TABLE 13-3: RESET TIME-OUTS

Oscillator Configuration	POR		BOR		Wake-up from SLEEP
	$\overline{PWRTE}$ = 0	$\overline{PWRTE}$ = 1	$\overline{PWRTE}$ = 0	$\overline{PWRTE}$ = 1	
HS	TPWRT + 1024•TOSC	1024•TOSC	TPWRT + 1024•TOSC	1024•TOSC	1024•TOSC
H4	TPWRT + TPLLRT + 1024•TOSC	TPLLRT + 1024•TOSC	TPWRT + TPLLRT + 1024•TOSC	TPLLRT + 1024•TOSC	TPLLRT + 1024•TOSC
EC	TPWRT	0	TPWRT	0	0
E4	TPWRT + TPLLRT	TPLLRT	TPWRT + TPLLRT	TPLLRT	TPLLRT

TABLE 13-4: STATUS BITS AND THEIR SIGNIFICANCE

$\overline{POR}$	$\overline{BOR}$	$\overline{TO}$	$\overline{PD}$	
0	x	1	1	Power-on Reset
0	x	0	x	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	x	x	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
1	0	1	1	Brown-out Reset
1	1	0	1	WDT Reset
1	1	0	0	WDT Wake-up
1	1	u	u	$\overline{MCLR}$ Reset during normal operation
1	1	1	0	$\overline{MCLR}$ Reset during SLEEP or Interrupt Wake-up from SLEEP

TABLE 13-5: RESET CONDITION FOR SPECIAL REGISTERS

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0x
$\overline{MCLR}$ Reset during normal operation	000h	000u uuuu	---- --uu
$\overline{MCLR}$ Reset during SLEEP	000h	0001 0uuu	---- --uu
WDT Reset	000h	0000 1uuu	---- --uu
WDT Wake-up	PC + 1	uuu0 0uuu	---- --uu
Brown-out Reset	000h	000x xuuu	---- --u0
Interrupt Wake-up from SLEEP	PC + 1(1)	uuu1 0uuu	---- --uu

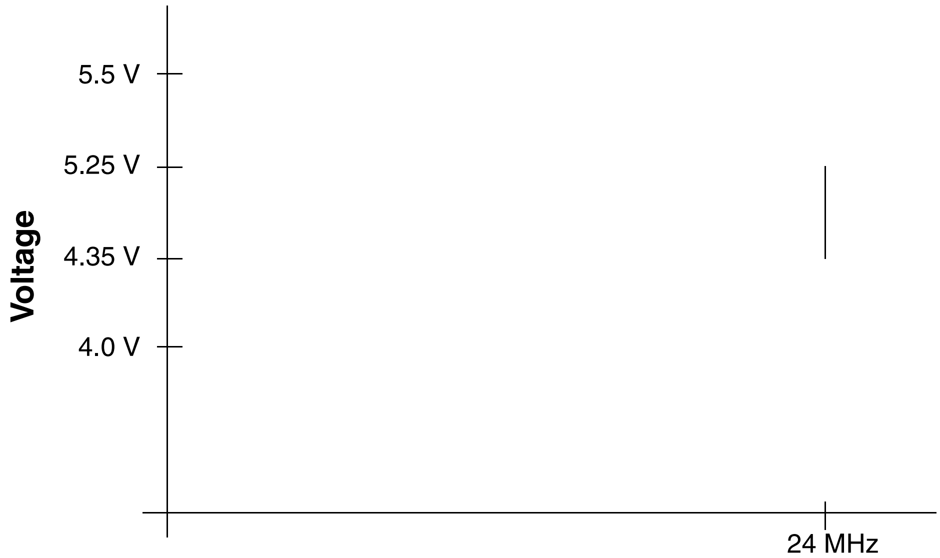
Legend: u = unchanged, x = unknown, - = unimplemented bit read as '0'.
Note 1: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

TABLE 13-6: REGISTERS ASSOCIATED WITH RESETS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
03h, 83h, 103h, 183h	Status	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	$\overline{BOR}$	---- --qq	---- --uu

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'.

FIGURE 16-1: VALID OPERATING REGIONS, FREQUENCY ON F_{INT}, -40°C ≤ TA ≤ +85°C



 Preliminary

16.3 AC (Timing) Characteristics

16.3.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{\text{RD}}$
cs	$\overline{\text{CS}}$	rw	$\overline{\text{RD}}$ or $\overline{\text{WR}}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{\text{SS}}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{\text{MCLR}}$	wr	$\overline{\text{WR}}$

Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance

16.3.2 TIMING CONDITIONS

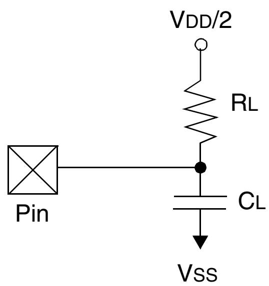
The temperature and voltages specified in Table 16-1 apply to all timing specifications unless otherwise noted. Figure 16-2 specifies the load conditions for the timing specifications.

TABLE 16-1: TEMPERATURE AND VOLTAGE SPECIFICATIONS - AC

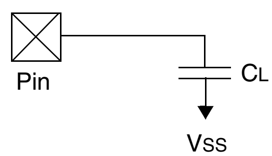
AC CHARACTERISTICS	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C $\leq T_A \leq$ +85°C for industrial Operating voltage V_{DD} range as described in DC spec Section 16.1 and Section 16.2.

FIGURE 16-2: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS

Load condition 1



Load condition 2



R_L = 464Ω

C_L = 50 pF for all pins except OSC2/CLKOUT, but including D and E[(1)] outputs as ports

C_L = 15 pF for OSC2 output

Note 1: PIC16C765 only.

NOTES: