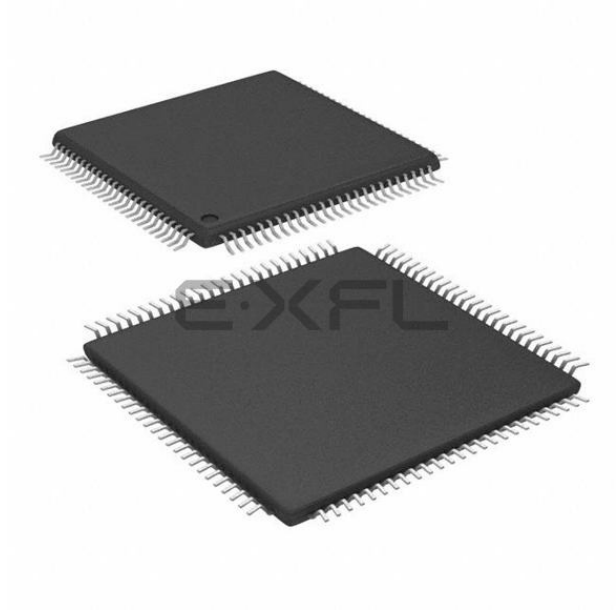


Welcome to E-XFL.COM

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	16-Bit
Speed	32MHz
Connectivity	I²C, PMP, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	85
Program Memory Size	192KB (65.5K 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 ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24fj192ga110-e-pt

TABLE 3-1: CPU CORE REGISTERS

Register(s) Name	Description
W0 through W15	Working Register Array
PC	23-Bit Program Counter
SR	ALU STATUS Register
SPLIM	Stack Pointer Limit Value Register
TBLPAG	Table Memory Page Address Register
PSVPAG	Program Space Visibility Page Address Register
RCOUNT	Repeat Loop Counter Register
CORCON	CPU Control Register

FIGURE 3-2: PROGRAMMER S MODEL

4.3.3 READING DATA FROM PROGRAM MEMORY USING PROGRAM SPACE VISIBILITY

The upper 32 Kbytes of data space may optionally be mapped into any 16K word page of the program space. This provides transparent access of stored constant data from the data space without the need to use special instructions (i.e., TBLRDL/H).

Program space access through the data space occurs if the Most Significant bit (MSb) of the data space EA is 1 and program space visibility is enabled by setting the PSV bit in the CPU Control (CORCON<2>) register. The location of the program memory space to be mapped into the data space is determined by the Program Space Visibility Page Address (PSVPAG) register. This 8-bit register defines any one of 256 possible pages of 16K words in program space. In effect, PSVPAG functions as the upper 8 bits of the program memory address, with the 15 bits of the EA functioning as the lower bits. Note that by incrementing the PC by 2 for each program memory word, the lower 15 bits of data space addresses directly map to the lower 15 bits in the corresponding program space addresses.

Data reads to this area add an additional cycle to the instruction being executed, since two program memory fetches are required.

Although each data space address, 8000h and higher, maps directly into a corresponding program memory address (see Figure 4-7), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits of any program space locations used as data should be programmed with 1111 1111 or 0000 0000 to force a NOP. This prevents possible issues should the area of code ever be accidentally executed.

Note:	PSV access is temporarily disabled during table reads/writes.

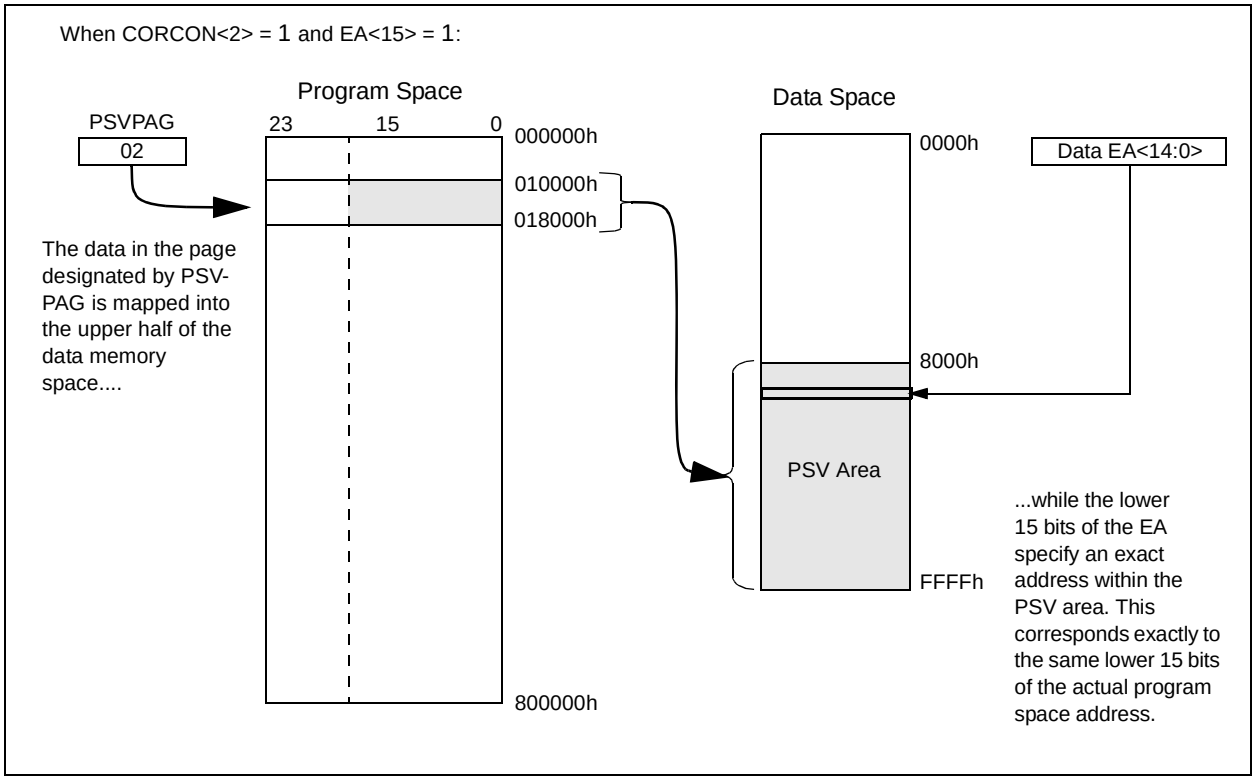
For operations that use PSV and are executed outside a REPEAT loop, the MOV and MOV.D instructions will require one instruction cycle in addition to the specified execution time. All other instructions will require two instruction cycles in addition to the specified execution time.

For operations that use PSV which are executed inside a REPEAT loop, there will be some instances that require two instruction cycles in addition to the specified execution time of the instruction:

- Execution in the first iteration
- Execution in the last iteration
- Execution prior to exiting the loop due to an interrupt
- Execution upon re-entering the loop after an interrupt is serviced

Any other iteration of the REPEAT loop will allow the instruction accessing data, using PSV, to execute in a single cycle.

FIGURE 4-7: PROGRAM SPACE VISIBILITY OPERATION



6.0 RESETS

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 7. Reset (DS39712).

The Reset module combines all Reset sources and controls the device Master Reset Signal, $\overline{SYSRST}$. The following is a list of device Reset sources:

- POR: Power-on Reset
- $\overline{MCLR}$: Pin Reset
- SWR: RESETInstruction
- WDT: Watchdog Timer Reset
- BOR: Brown-out Reset
- CM: Configuration Mismatch Reset
- TRAPR: Trap Conflict Reset
- IOPUWR: Illegal Opcode Reset
- UWR: Uninitialized W Register Reset

A simplified block diagram of the Reset module is shown in Figure 6-1.

Any active source of Reset will make the $\overline{SYSRST}$ signal active. Many registers associated with the CPU and peripherals are forced to a known Reset state. Most registers are unaffected by a Reset; their status is unknown on POR and unchanged by all other Resets.

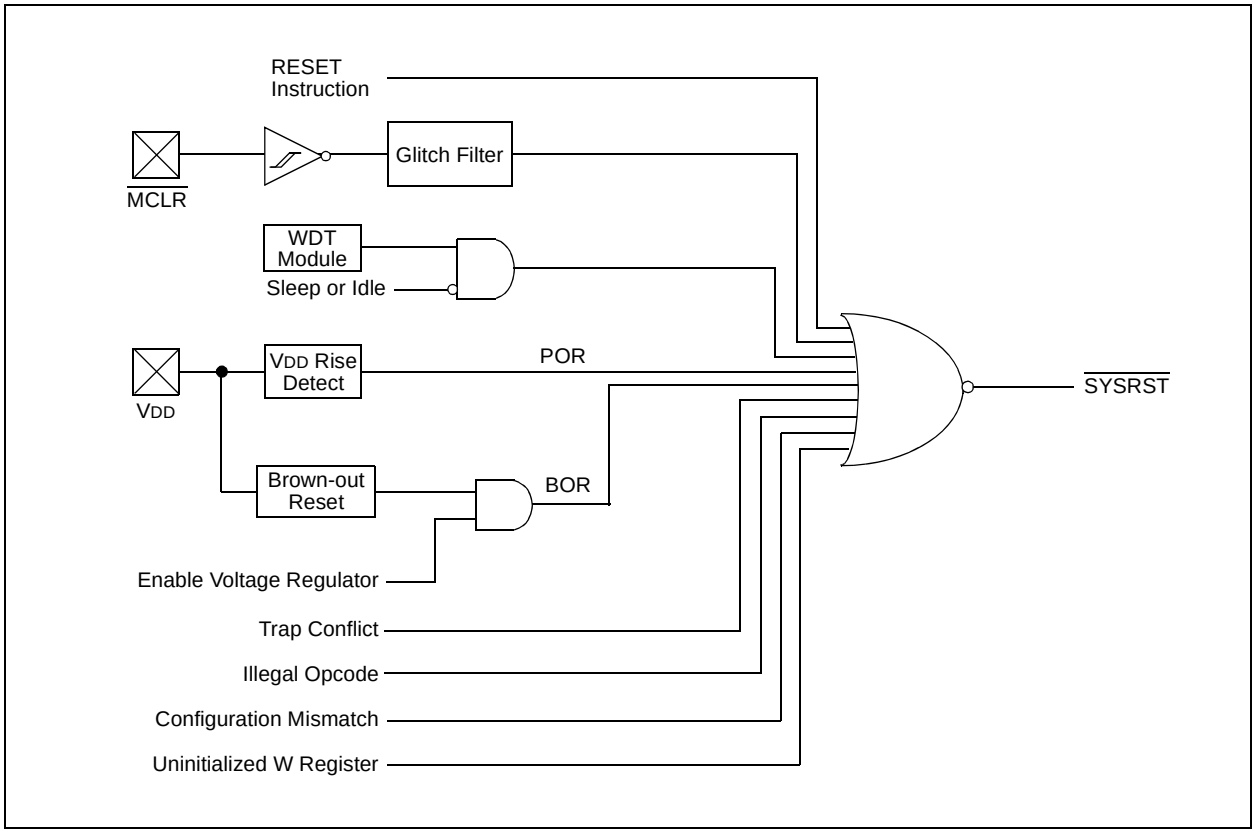
Note:	Refer to the specific peripheral or CPU section of this manual for register Reset states.

All types of device Reset will set a corresponding status bit in the RCON register to indicate the type of Reset (see Register 6-1). A Power-on Reset will clear all bits except for the BOR and POR bits (RCON<1:0>) which are set. The user may set or clear any bit at any time during code execution. The RCON bits only serve as status bits. Setting a particular Reset status bit in software will not cause a device Reset to occur.

The RCON register also has other bits associated with the Watchdog Timer and device power-saving states. The function of these bits is discussed in other sections of this data sheet.

Note:	The status bits in the RCON register should be cleared after they are read so that the next RCON register value after a device Reset will be meaningful.

FIGURE 6-1: RESET SYSTEM BLOCK DIAGRAM



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 7-18: IPC1: INTERRUPT PRIORITY CONTROL REGISTER 1

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—
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 **T2IP<2:0>:** Timer2 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 **OC2IP<2:0>:** Output Compare Channel 2 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 **IC2IP<2:0>:** Input Capture Channel 2 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3-0 **Unimplemented:** Read as ‘0’

REGISTER 7-23: IPC6: INTERRUPT PRIORITY CONTROL REGISTER 6

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—
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 **T4IP<2:0>:** Timer4 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 **OC4IP<2:0>:** Output Compare Channel 4 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 **OC3IP<2:0>:** Output Compare Channel 3 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 3-0 **Unimplemented:** Read as ‘0’

REGISTER 16-1: I2CxCON: I2Cx CONTROL REGISTER (CONTINUED)

bit 5 **ACKDT:** Acknowledge Data bit (When operating as I^2C master. Applicable during master receive.)
Value that will be transmitted when the software initiates an Acknowledge sequence.
1 = Sends NACK during Acknowledge
0 = Sends ACK during Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit (When operating as I^2C master. Applicable during master receive.)
1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit. Hardware clear at end of master Acknowledge sequence.
0 = Acknowledge sequence not in progress

bit 3 **RCEN:** Receive Enable bit (when operating as I^2C master)
1 = Enables Receive mode for I^2C. Hardware clear at end of eighth bit of master receive data byte.
0 = Receives sequence not in progress

bit 2 **PEN:** Stop Condition Enable bit (when operating as I^2C master)
1 = Initiates Stop condition on SDAx and SCLx pins. Hardware clear at end of master Stop sequence.
0 = Stop condition not in progress

bit 1 **RSEN:** Repeated Start Condition Enabled bit (when operating as I^2C master)
1 = Initiates Repeated Start condition on SDAx and SCLx pins. Hardware clear at end of master Repeated Start sequence.
0 = Repeated Start condition not in progress

bit 0 **SEN:** Start Condition Enabled bit (when operating as I^2C master)
1 = Initiates Start condition on SDAx and SCLx pins. Hardware clear at end of master Start sequence.
0 = Start condition not in progress

REGISTER 16-2: I2CxSTAT: I2Cx STATUS REGISTER (CONTINUED)

bit 4 **P:** Stop bit
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

bit 3 **S:** Start bit
1 = Indicates that a Start (or Repeated Start) bit has been detected last
0 = Start bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit (when operating as I^2C slave)
1 = Read – indicates data transfer is output from slave
0 = Write – indicates data transfer is input to slave
Hardware set or clear after reception of I^2C device address byte.

bit 1 **RBF:** Receive Buffer Full Status bit
1 = Receive complete, I2CxRCV is full
0 = Receive not complete, I2CxRCV is empty
Hardware set when I2CxRCV is written with received byte. Hardware clear when software reads I2CxRCV.

bit 0 **TBF:** Transmit Buffer Full Status bit
1 = Transmit in progress, I2CxTRN is full
0 = Transmit complete, I2CxTRN is empty
Hardware set when software writes I2CxTRN. Hardware clear at completion of data transmission.

19.3 Alarm

- Configurable from half second to one year
- Enabled using the ALRMEN bit (ALCFGRPT<15>, Register 19-3)
- One-time alarm and repeat alarm options available

19.3.1 CONFIGURING THE ALARM

The alarm feature is enabled using the ALRMEN bit. This bit is cleared when an alarm is issued. Writes to ALRMVAL should only take place when ALRMEN = 0.

As shown in Figure 19-2, the interval selection of the alarm is configured through the AMASK bits (ALCFGRPT<13:10>). These bits determine which and how many digits of the alarm must match the clock value for the alarm to occur.

The alarm can also be configured to repeat based on a preconfigured interval. The amount of times this occurs once the alarm is enabled is stored in the ARPT bits, ARPT<7:0> (ALCFGRPT<7:0>). When the value of the ARPT bits equals 00h and the CHIME bit (ALCFGRPT<14>) is cleared, the repeat function is disabled and only a single alarm will occur. The alarm can be repeated up to 255 times by loading ARPT<7:0> with FFh.

After each alarm is issued, the value of the ARPT bits is decremented by one. Once the value has reached 00h, the alarm will be issued one last time, after which the ALRMEN bit will be cleared automatically and the alarm will turn off.

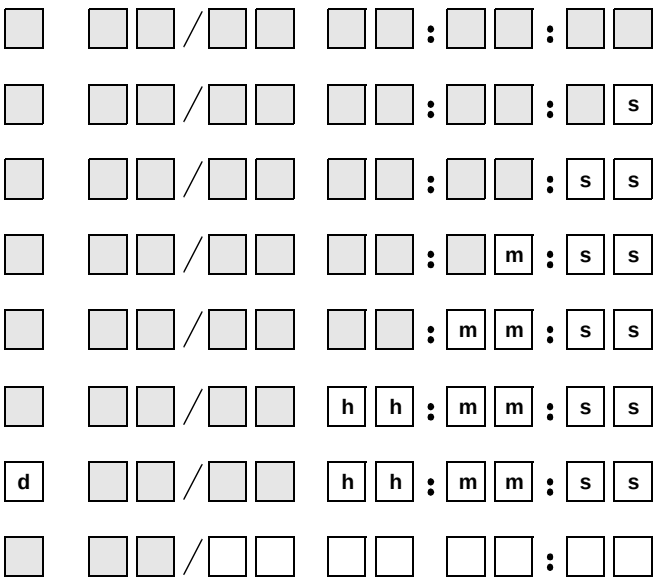
Indefinite repetition of the alarm can occur if the CHIME bit = 1. Instead of the alarm being disabled when the value of the ARPT bits reaches 00h, it rolls over to FFh and continues counting indefinitely while CHIME is set.

19.3.2 ALARM INTERRUPT

At every alarm event, an interrupt is generated. In addition, an alarm pulse output is provided that operates at half the frequency of the alarm. This output is completely synchronous to the RTCC clock and can be used as a trigger clock to other peripherals.

Note:	Changing any of the registers, other then the RCFGCAL and ALCFGRPT registers, and the CHIME bit while the alarm is enabled (ALRMEN = 1), can result in a false alarm event leading to a false alarm interrupt. To avoid a false alarm event, the timer and alarm values should only be changed while the alarm is disabled (ALRMEN = 0). It is recommended that the ALCFGRPT register and CHIME bit be changed when RTCSYNC = 0.

FIGURE 19-2: ALARM MASK SETTINGS



Note 1: Annually, except when configured for February 29.

24.0 CHARGE TIME MEASUREMENT UNIT (CTMU)

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 11. "Charge Time Measurement Unit (CTMU)"** (DS39724).

The Charge Time Measurement Unit is a flexible analog module that provides accurate differential time measurement between pulse sources, as well as asynchronous pulse generation. Its key features include:

- Four edge input trigger sources
- Polarity control for each edge source
- Control of edge sequence
- Control of response to edges
- Time measurement resolution of 1 nanosecond
- Accurate current source suitable for capacitive measurement

Together with other on-chip analog modules, the CTMU can be used to precisely measure time, measure capacitance, measure relative changes in capacitance or generate output pulses that are independent of the system clock. The CTMU module is ideal for interfacing with capacitive-based sensors.

The CTMU is controlled through two registers: CTMUCON and CTMUICON. CTMUCON enables the module and controls edge source selection, edge source polarity selection, and edge sequencing. The CTMUICON register controls the selection and trim of the current source.

24.1 Measuring Capacitance

The CTMU module measures capacitance by generating an output pulse, with a width equal to the time, between edge events on two separate input channels. The pulse edge events to both input channels can be selected from four sources: two internal peripheral modules (OC1 and Timer1) and two external pins (CTEDG1 and CTEDG2). This pulse is used with the module's precision current source to calculate capacitance according to the relationship

$$I = C \bullet \frac{dV}{dT}$$

For capacitance measurements, the A/D Converter samples an external capacitor (C_{APP}) on one of its input channels after the CTMU output's pulse. A Precision Resistor (R_{PR}) provides current source calibration on a second A/D channel. After the pulse ends, the converter determines the voltage on the capacitor. The actual calculation of capacitance is performed in software by the application.

Figure 24-1 shows the external connections used for capacitance measurements, and how the CTMU and A/D modules are related in this application. This example also shows the edge events coming from Timer1, but other configurations using external edge sources are possible. A detailed discussion on measuring capacitance and time with the CTMU module is provided in the *"PIC24F Family Reference Manual"*.

FIGURE 24-1: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR CAPACITANCE MEASUREMENT

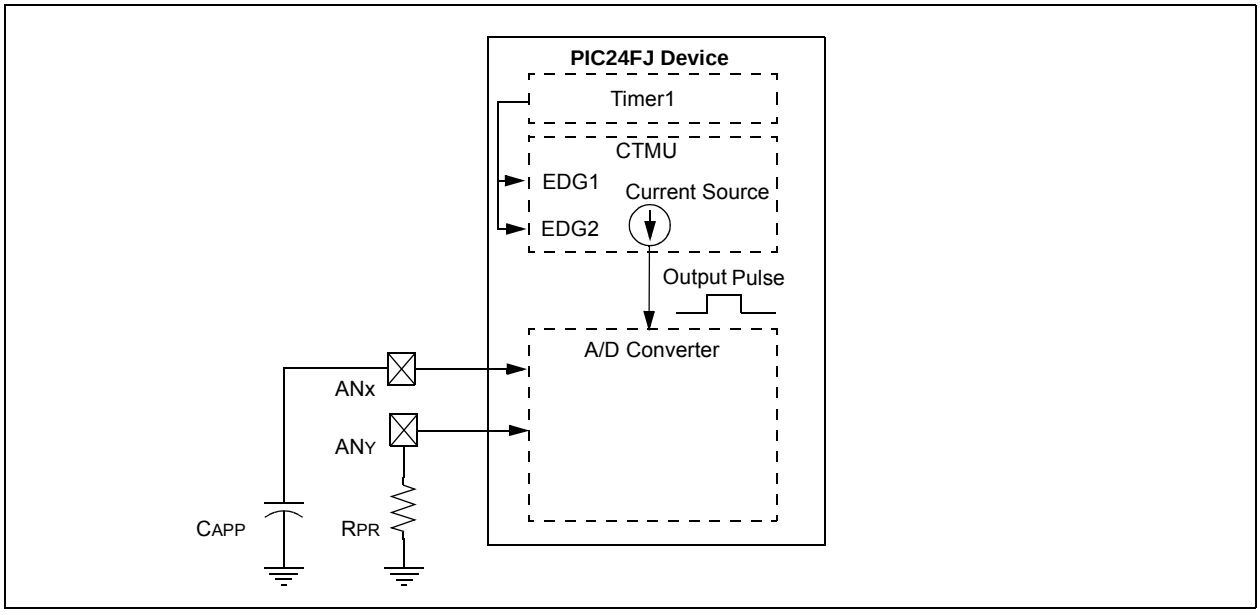


TABLE 26-1: SYMBOLS USED IN OPCODE DESCRIPTIONS

Field	Description
#text	Means literal defined by “text ”
(text)	Means “content of text ”
[text]	Means “the location addressed by text ”
{ }	Optional field or operation
<n:m>	Register bit field
.b	Byte mode selection
.d	Double-Word mode selection
.S	Shadow register select
.w	Word mode selection (default)
bit4	4-bit bit selection field (used in word addressed instructions) ∈ {0...15}
C, DC, N, OV, Z	MCU Status bits: Carry, Digit Carry, Negative, Overflow, Sticky Zero
Expr	Absolute address, label or expression (resolved by the linker)
f	File register address ∈ {0000h...1FFFh}
lit1	1-bit unsigned literal ∈ {0,1}
lit4	4-bit unsigned literal ∈ {0...15}
lit5	5-bit unsigned literal ∈ {0...31}
lit8	8-bit unsigned literal ∈ {0...255}
lit10	10-bit unsigned literal ∈ {0...255} for Byte mode, {0:1023} for Word mode
lit14	14-bit unsigned literal ∈ {0...16384}
lit16	16-bit unsigned literal ∈ {0...65535}
lit23	23-bit unsigned literal ∈ {0...8388608}; LSB must be ‘0’
None	Field does not require an entry, may be blank
PC	Program Counter
Slit10	10-bit signed literal ∈ {-512...511}
Slit16	16-bit signed literal ∈ {-32768...32767}
Slit6	6-bit signed literal ∈ {-16...16}
Wb	Base W register ∈ {W0..W15}
Wd	Destination W register ∈ { Wd, [Wd], [Wd++], [Wd--], [++Wd], [--Wd] }
Wdo	Destination W register ∈ { Wnd, [Wnd], [Wnd++], [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] }
Wm,Wn	Dividend, Divisor working register pair (direct addressing)
Wn	One of 16 working registers ∈ {W0..W15}
Wnd	One of 16 destination working registers ∈ {W0..W15}
Wns	One of 16 source working registers ∈ {W0..W15}
WREG	W0 (working register used in file register instructions)
Ws	Source W register ∈ { Ws, [Ws], [Ws++], [Ws--], [++Ws], [--Ws] }
Wso	Source W register ∈ { Wns, [Wns], [Wns++], [Wns--], [++Wns], [--Wns], [Wns+Wb] }