Typical Architecture of an Embedded System

Figure 1-2 shows a configuration diagram of a typical embedded system consisting of two main parts: embedded hardware and embedded software. The embedded hardware

primarily includes the processor, memory, bus, peripheral devices, I/O ports, and various controllers. The embedded software usually contains the embedded operating system and various applications.

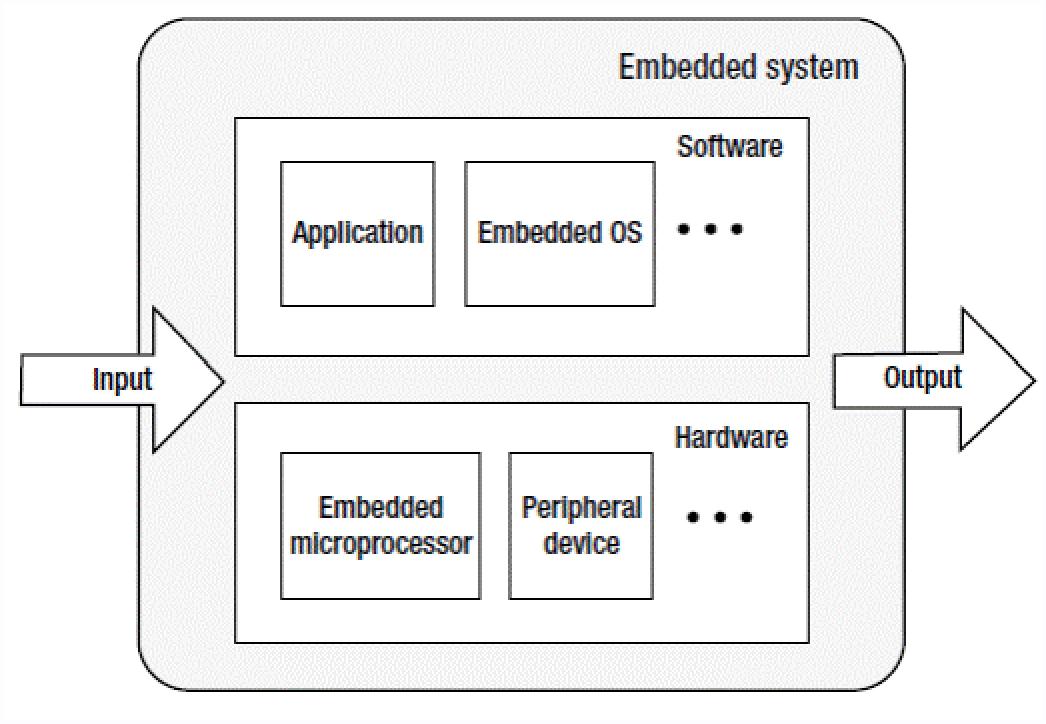


Figure 1-2. Basic architecture of an embedded system

Input and output are characteristics of any open system, and the embedded system is no exception. In the embedded system, the hardware and software often collaborate to deal with various input signals from the outside and output the processing results through some form. The input signal may be an ergonomic device (such as a keyboard, mouse, or touch screen) or the output of a sensor circuit in another embedded system. The output may be in the form of sound, light, electricity, or another analog signal, or a record or file for a database.

Typical Hardware Architecture

The basic computer system components—microprocessor, memory, and input and output modules—are interconnected by a system bus in order for all the parts to communicate and execute a program (see Figure 1-3).

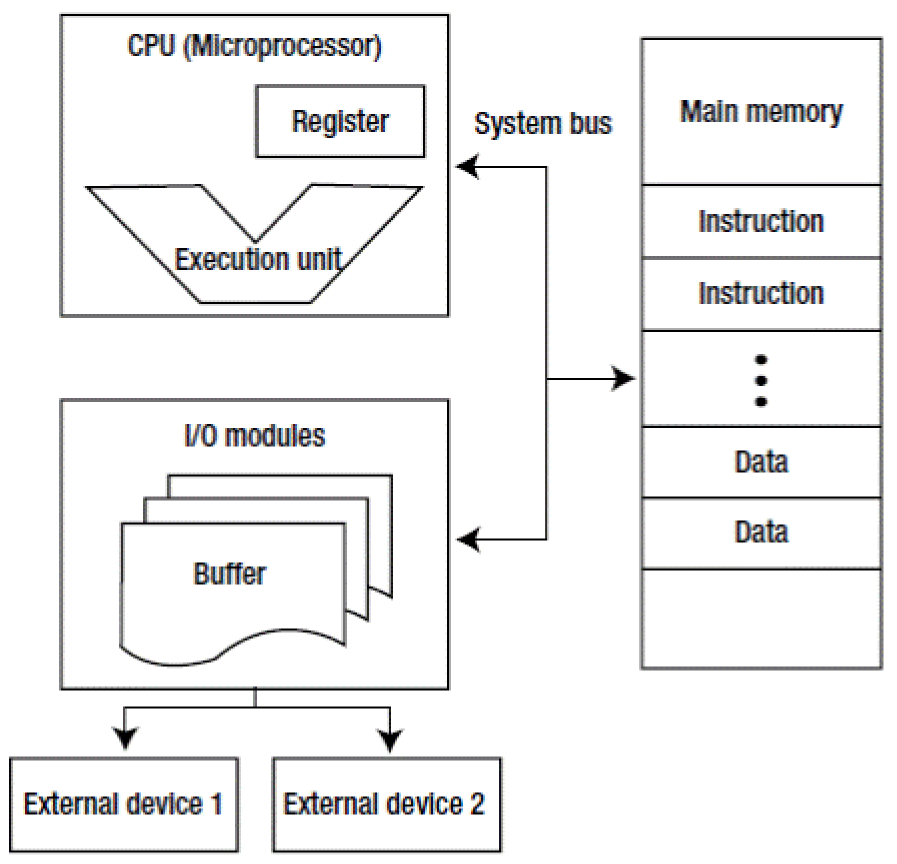


Figure 1-3. Computer architecture

In embedded systems, the microprocessor's role and function are usually the same as those of the CPU in a general-purpose computer: control computer operation, execute instructions, and process data. In many cases, the microprocessor in an embedded system is also called the CPU. Memory is used to store instructions and data. I/O modules are responsible for the data exchange between the processor, memory, and external devices. External devices include secondary storage devices (such as flash and hard disk), communications equipment, and terminal equipment. The system bus provides data

and controls signal communication and transmission for the processor, memory, and I/O modules.

There are basically two types of architecture that apply to embedded systems: Von Neumann architecture and Harvard architecture.

Von Neumann Architecture

Von Neumann architecture (also known as Princeton architecture) was first proposed by John von Neumann. The most important feature of this architecture is that the software and data use the same memory: that is, "The program is data, and the data is the program" (as shown in Figure 1-4).

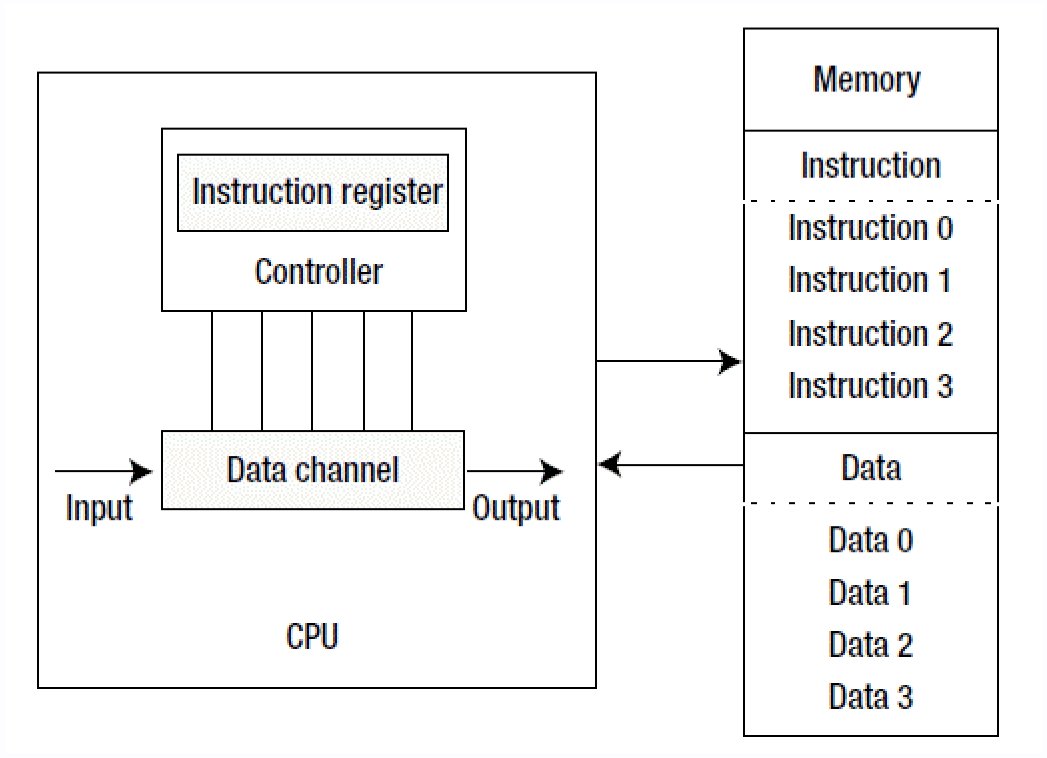


Figure 1-4. Von Neumann architecture

In the Von Neumann architecture, an instruction and data share the same bus. In this architecture, the transmission of information becomes the bottleneck of computer performance and affects the speed of data processing; so, it is often called the *Von Neumann bottleneck*. In reality, cache and branch-prediction technology can effectively solve this issue.

Harvard Architecture

The Harvard architecture was first named after the Harvard Mark I computer. Compared with the Von Neumann architecture, a Harvard architecture processor has two outstanding features. First, instructions and data are stored in two separate memory modules; instructions and data do not coexist in the same module. Second, two independent buses are used as dedicated communication paths between the CPU and memory; there is no connection between the two buses. The Harvard architecture is shown in Figure 1-5.

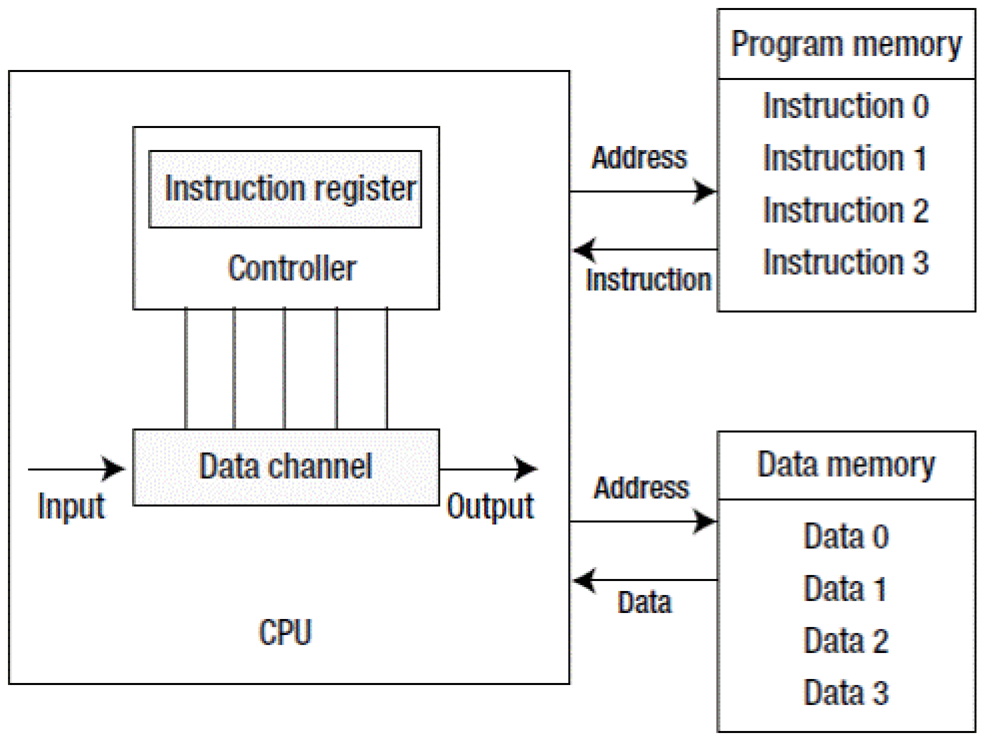


Figure 1-5. Harvard architecture

Because the Harvard architecture has separate program memory and data memory, it can provide greater data-memory bandwidth, making it the ideal choice for digital signal processing. Most systems designed for digital signal processing (DSP) adopt the Harvard architecture. The Von Neumann architecture features simple hardware design and flexible program and data storage and is usually the one chosen for general-purpose and most embedded systems.

To efficiently perform memory reads/writes, the processor is not directly connected to the main memory, but to the cache. Commonly, the only difference between the Harvard architecture and the Von Neumann architecture is single or dual L1 cache. In the Harvard architecture, the L1 cache is often divided into an instruction cache (I cache) and a data cache (D cache), but the Von Neumann architecture has a single cache.

PIC MICROCONTROLLER ARCHITECTURE

PIC MICROCONTROLLER ARCHITECTURE: PIC stands for Peripheral Interface Controller. PIC microcontroller was developed by microchip technology in 1993. It was developed for supporting PDP computers to control its peripheral devices and that's why it was named Peripheral Interface Controller.

PIC microcontrollers are of low cost, very fast and easy for the programming and execution of program. Their interfacing with other peripherals is also very easy. PIC Microcontrollers from Microchip Company are divided into 4 large families. In this PIC MICROCONTROLLER ARCHITECTURE article, I will explain step by step about PIC MICROCONTROLLER ARCHITECTURE and components used in pic microocntrollers. I recommend you to check a list of **Pic microcontroller project** here.

- First family: PIC10 (10FXXX) called Low End
- Second family: PIC12 (PIC12FXXX) called Mid-Range
- Third family: PIC16 (16FXXX)
- Fourth family: PIC 17/18 (18FXXX)

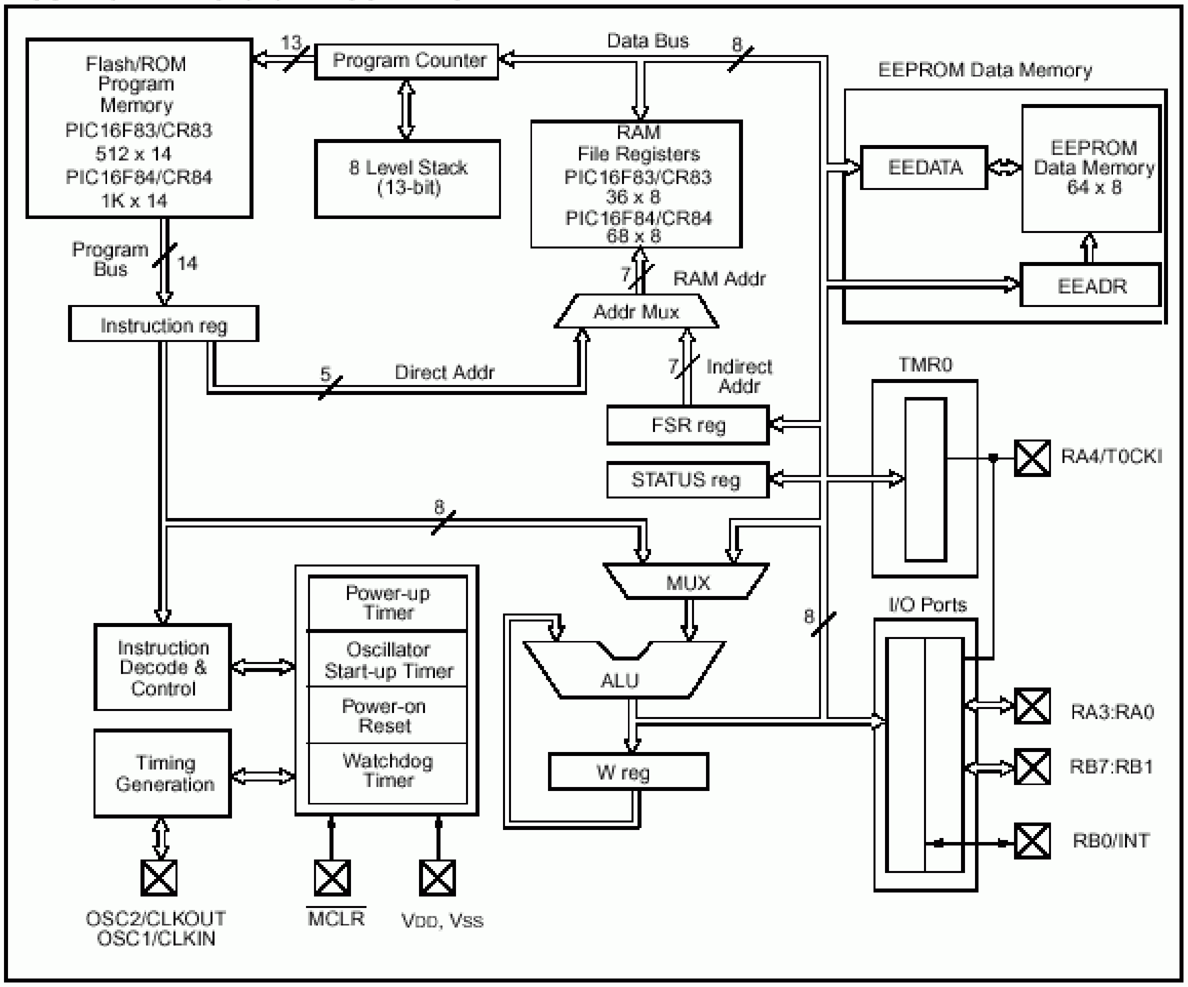
Each family has a variety of components along with built in special features. It offers a lot of memory sizes and pin packages and different clock ratings.

ARCHITECTURE:

PIC Microcontroller architecture is based on Harvard architecture and supports RISC architecture (Reduced Instruction Set Computer). PIC microcontroller architecture consists of memory organization (ram, rom, stack), CPU, timers, counter, ADC, DAC, serial communication, CCP module and I/O ports. PIC microcontroller also supports the protocols like CAN, SPI, UART for interfacing with other peripherals.

PIC MICROCONTROLLER ARCHITECTURE block diagram

FIGURE 3-1: PIC16F8X BLOCK DIAGRAM



1. **CPU (Central Processing Unit):**

PIC microcontroller's CPU consists of

- Arithmetic logic unit (ALU)
- Memory unit (MU)
- Control unit (CU)
- Accumulator

ALU is used for arithmetic operations and for logical decisions. Memory is used for storing the instructions after processing. Control unit is used to control the internal and external peripherals which are connected to the CPU and accumulator is used for storing the results.

2. **MEMORY ORGANIZATION:**

PIC microcontroller memory module consists of mainly 3 types of memories:

- **PROGRAM MEMORY:**

It contains the written program after we burned it in microcontroller. Program Counter executes commands stored in the program memory, one after the other. Pic microcontroller can have 8K words x 14 bits of Flash program memory that can be electrically erased and reprogrammed. Whenever we burn program into the micro, we erase an old program and write a new one.

- **DATA MEMORY:**

It is a RAM type which is used to store the data temporarily in its registers. The RAM memory is classified into banks. Each bank extends up to 7Fh (128 bytes). Number of banks may vary depending on the microcontroller. PIC16F84 has only two banks. Banks contain Special Function Registers (SFR) and General Purpose Registers (GPR). The lower locations of each bank are reserved for the Special Function Registers and upper locations are for General Purpose Registers.

General Purpose Registers (GPR):
These registers don't have any special function. These are used for general purpose for multiplying, addition or subtraction and then storing the results in other registers. CPU can easily access the data in these registers.

Special Function Registers (SFR):
These registers are used for special purposes and they cannot be used as normal registers. Their function is set at the time of manufacturing. They perform the function assigned to them and user cannot change the function of SFR. Three important SFRs for programming are:

STATUS register:	It changes the bank
PORT registers:	It assigns logic values 0 or 1 to the ports
TRIS registers:	It is a data direction register for input and output

- **DATA EEPROM:**

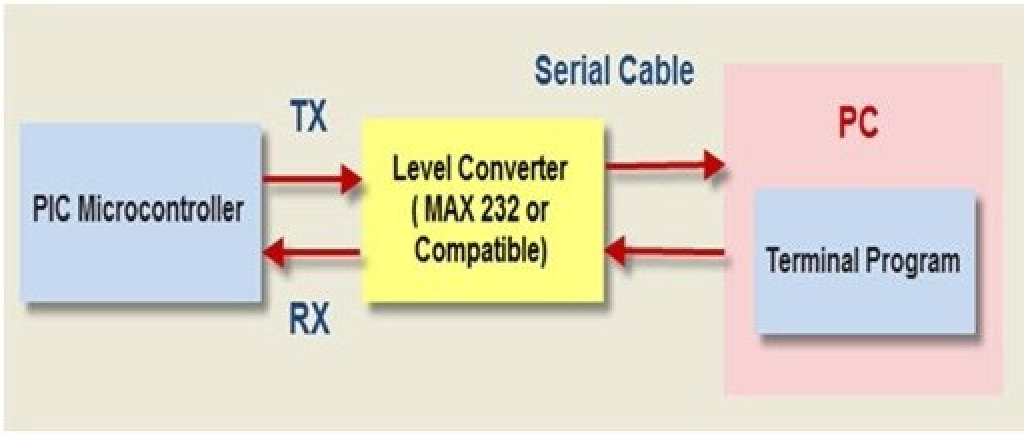
This memory allows storing the variables as a result of burning the written program. It is readable and writable during normal operation (over the full VDD range). This memory is not directly mapped in the register file. It is indirectly addressed through the SFRs. There are six SFRs which are used to read and write to this memory (EECON1, EECON2, EEDATA, EEDATH, EEADR, EEADRH).

3. **SERIAL COMMUNICATION:**

The transfer of one bit of data at time consecutively over a communication channel is called Serial Communication. There are three protocols of serial communication:

- **USART:** It stands for Universal synchronous and Asynchronous Receiver and Transmitter which provides a serial communication in two devices. In this protocol data is transmitted and received bit by bit through a single wire according to the clock pulses. To send and receive data serially the PIC microcontroller has two pins TXD and RXD.

- **SPI Protocol:** SPI stands for Serial Peripheral Interface. It is used to send data between PIC microcontrollers and other peripherals like sensors, shift registers and SD cards. Three wire SPI communications is supported in PIC microcontroller between two devices on a common clock source. SPI protocol has greater data handling capability than that of the USART.**I2C Protocol:** I2C stands for Inter Integrated Circuit, and this protocol is used to connect low speed devices like microcontrollers, EEPROMS and A/D converters. PIC microcontroller support two wire Interface or I2C communication between two devices which can work as both Master and Slave device.



Serial Communication

4. **INTERRUPTS:**

There are 20 internal interrupts and three external interrupt sources in PIC microcontrollers which are related with different peripherals like ADC, USART, Timers, and CCP etc.

5. **I/O PORTS:**

Let us take PIC16 series, it consists of five ports, such as Port A, Port B, Port C, Port D and Port E.

- **Port A:**This port is 7-bit wide and can be used for both input and output. The status of TRISA register decided whether it is used as input or output port.
- **Port B:**It is an 8-bit port. This port also can be used as input and output. Moreover in input mode four of its bits are variable according to the interrupt signals.
- **Port C:**It is also an 8-bit port and can be used as both input and output port which is determined by the status of the TRISC register.
- **Port D:**This 8-bit port, unlike Port A, B and C is not an input/output port, but is used as acts as a slave port for the connection to

the microprocessor When in I/O mode Port D all pins should have Schmitt Trigger buffers.

- **Port E:**It is a 3-bit port which is used as the additional feature of the control signals to the A/D converter.

6. CCP MODULE:

A CCP module works in the following three modes:

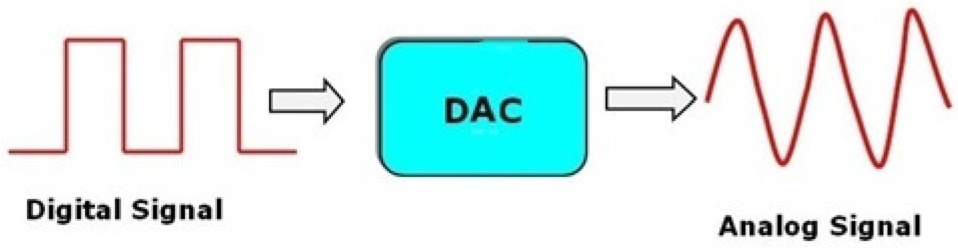
- **Capture Mode:** In this mode time is captured when a signal is arrived, or we can say that, when the CCP pin goes high it captures the value of the Timer1.
- **Compare Mode:** It works same as an analog comparator, which means that when timer 1's value reaches some reference value it will give an output signal.
- **PWM Mode:** This mode provides a 10 bit resolution pulse and duty cycle that is programmable.

7. Timers:

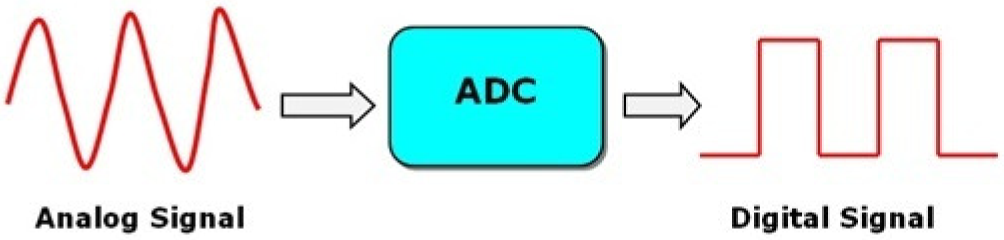
Timers and counters are important as timers can tell the time and count. PIC microcontroller can have up to four timers (depending upon the family) Timer0, Timer1, Timer2 and Timer3. Timer0 and Timer2 are of 8-bits while the Timer1 and Timer3 are of 16-bits, which can also be used as a counter. These timers work according to the selected modes.

8. D/A CONVERTER:

There are no **analog outputs in PIC Microcontroller.** To get analog output we have to use external Digital-to-Analog Converter (DAC). It can convert 8 bits of digital number from the eight digital outputs of PIC microcontroller.



9. A/D CONVERTER:



It converts the analog voltage levels to digital voltage values. In PIC Microcontroller, ADC has 8-channels and has resolution of 10-bit, which means that if we have to convert an analog voltage between 0V to 5V the converter will divide it to 2^10 levels (1024 levels). The special function registers ADCON0 and ADCON1 control the operation of ADC. The converter stores the lower 8 bits in ADRESL register and the upper bits in the ADRESH register. Reference voltage of 5V is required for the operation of the converter.