Microcontrollers for Embedded Systems

Microprocessors and microcontrollers are widely used in embedded system products. As Embedded product uses a microprocessor (or controller) to do one task and one task only. A printer is an example of embedded system since the processor inside it performs one task only; namely, getting the data and printing it. Contrast this with the Pentium based PC. A PC can be used for any number of applications such as word processor, video game player, internet terminal etc.

Choosing a microcontroller

There are four major 8-bit microcontrollers. They are Motorola's 6811, Intel's 8051, Zilog's Z8 and PIC 16X from Microchip technology. Each of the above microcontrollers have a unique instruction set and register set; therefore, they are not compatible with each other. Programs written for one will not run on the others. There are also 16-bit and 32-bit microcontrollers made by various chip makers.

With all these different microcontrollers, what criteria do designers consider in choosing one?

Three criteria in choosing microcontrollers are as follows:

1. Meeting the computing needs of the task at hand efficiently and cost effectively. In analysing the needs of the micro-controller based projects, we must first see whether an 8-bit, 16-bit, 32-bit microcontroller can best handle the computing needs of the task most effectively. Some other considerations in this category are:

- Speed: What is the highest speed that the micro-controller supports? Speed of operation is mainly determined by the number of clock ticks required per instruction cycle and the maximum clock frequency supported by the controller. Speed of operation is usually expressed in terms of MIPS.
- Power Consumption: This is especially critical for battery powered products. Does the controller support idle and power down modes to reduce power consumption? It is a crucial factor since high power requirements and high power dissipation requires bulky power supplies and cooling equipment.
- The amount of RAM and ROM on chip. Does the controller support sufficient code memory space to hold the compiled hex code? Does the controller support sufficient internal data memory space (on chip RAM) to hold runtime variables and data structures?
- The number of I/O pins and the timer on the chip. Does it support all the peripherals required by the application like serial interface, parallel interface? Does it satisfy the general I/O port requirements by the applications? Does it have sufficient number of timers/counters? Does it have built-in ADC/DAC hardware in case of signal processing applications?
- How easy it is to upgrade to higher performance or lower power consumption versions?
- Cost per unit: This is important in terms of the final cost of the product in which a microcontroller is used. For example, there are microcontrollers that cost 50 cents per unit when purchased 100,000 units at a time.

2. The second criterion in choosing a microcontroller is how easy it is to develop products around it. Key considerations include the availability of the assembler, debugger, a code-efficient C language compiler, emulator, technical support and both in-house and outside expertise.

3. The third criterion in choosing a microcontroller is its ready availability in needed quantities both now and in the future.Usually deals with the *Lead time* - which is the time elapsed between purchase order approval and the supply of the product.Lead time should be short enough so as to reduce the product development time which also effects the time to market.