Selecting a Micro

There are two family of microcontroller that are extremely popular among hobbyist. The "PIC" series from Microchip and "AVR" series from Atmel. Both these chips are wonder of modern microelectronics. PIC had ruled for a long time but now AVR is also getting in serious competition.

>>Speed and ease of use.

I prefer AVRs because one major reason. They are fast. When a PIC and an AVR is running with same frequency lets say 12 MHz, then the AVR is actually executing four times faster than the PIC ! Yes 4 times faster. This is because the PIC requires 4 cycle to perform a single execute cycle while the AVR execute most of the instruction in 1 clock cycle.

In addition, I like the AVR architecture because of its consistency. It makes using the most advanced feature of AVR very easy to use. These chips are easily available and they are cheap.

>>Free "C" compiler.

One more important thing, generally microcontroller programs are written in assembly language for efficiency. Which you may know is a very low level and unstructured language. Therefore, to achieve a small thing lots of code need to be written and the programmer cannot concentrate on program logic. This makes the things harder. However, mid to high end AVR mcus are powerful enough to support high level language such as C efficiently. To write programs in 'c' for AVR or any other microcontroller we need a c compiler for that MCU. Generally these compilers are priced so high that they are out of reach of hobbyist and small company. But fortunately for AVR mcus there is a very high quality 'c' compiler for free. It is GNU C compiler. It is extremely popular and it has a large user base. It is part of the free and open source software revolution that we are witnessing (like Linux, PHP, Apache, OpenOffice.org etc). Providing excellent software at no cost. They are the dedicated work of programmers around the globe.

Different AVRs

Now we have selected AVR as our choice of micro lets see what different AVRs are available. The popular micros of AVR family are

- AT TINY2313 [20 PIN, 2K Flash ,128 Byte RAM, 128 Bytes EEPROM]
- AT MEGA8 [28 PIN , 8K Flash , 1KB Ram, 512 Byte EEPROM]
- AT MEGA16 [40 PIN, 16K Flash, 1KB Ram, 512 Byte EEPROM]
- AT MEGA32 [40 PIN, 32K Flash, 2KB Ram, 1KB EEPROM]

You can choose any one of them according to your need. The concepts you learn from one MCU can be easily applied for any AVR provided that chip has that functionality. For Example you learned to use ADC (analogue to digital converter) on MEGA8 then you can easily use ADC on any AVR that has one (like MEGA16 or MEGA32).

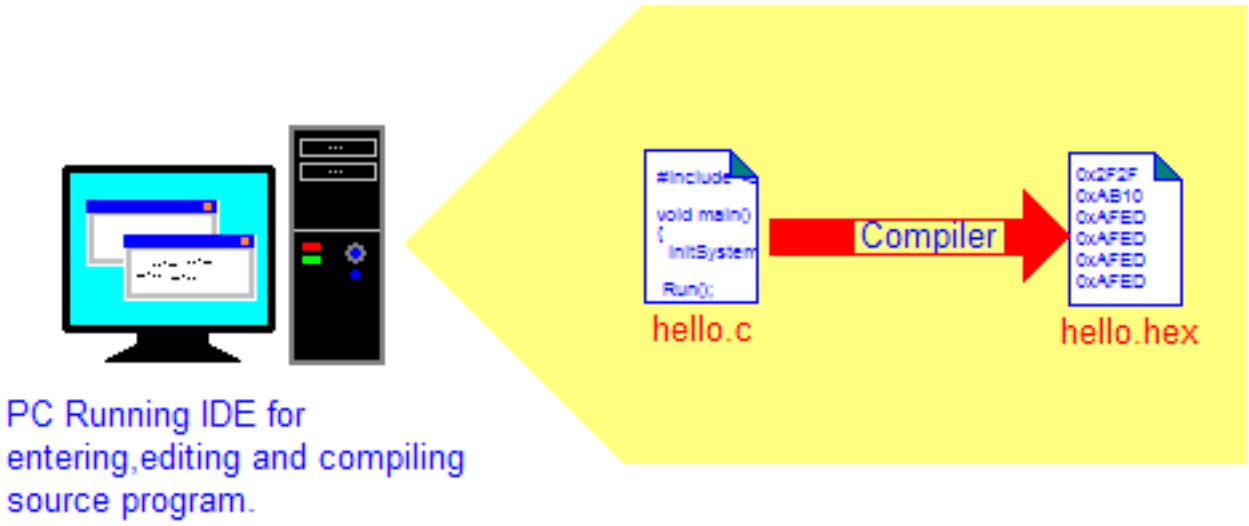
In this tutorial series we will be using AT MEGA8 because it has right combination of size, cost and features. It has lots of RAM and Flash to make moderate sized project with ease. Also it has many on chip peripherals like

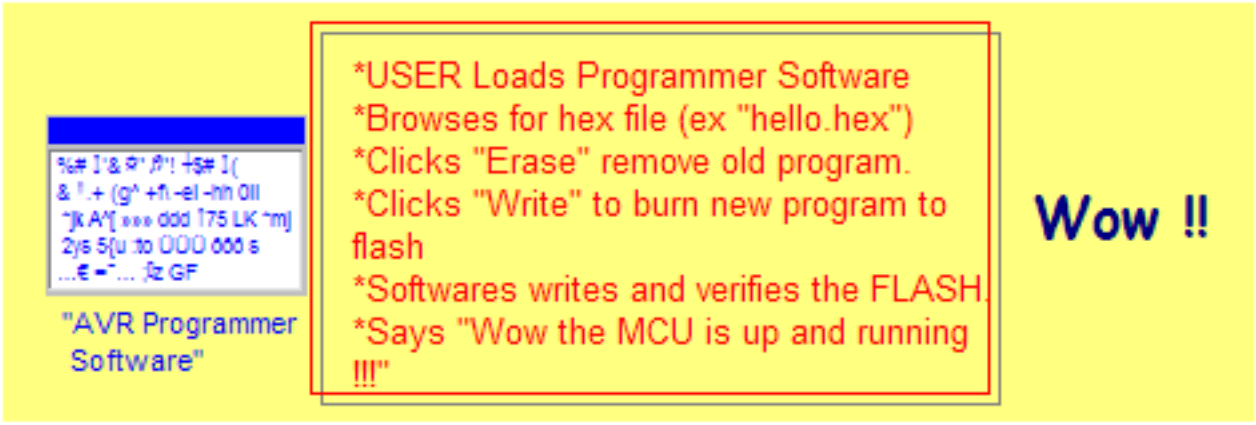
- Timers with PWM.
- Serial Communication
  - USART for connection with PC and other micros
  - SPI can be used for connection with other chips like DataFlash, Graphical LCD, MemoryCards.
  - TWI
- Analogue to Digital Converter.

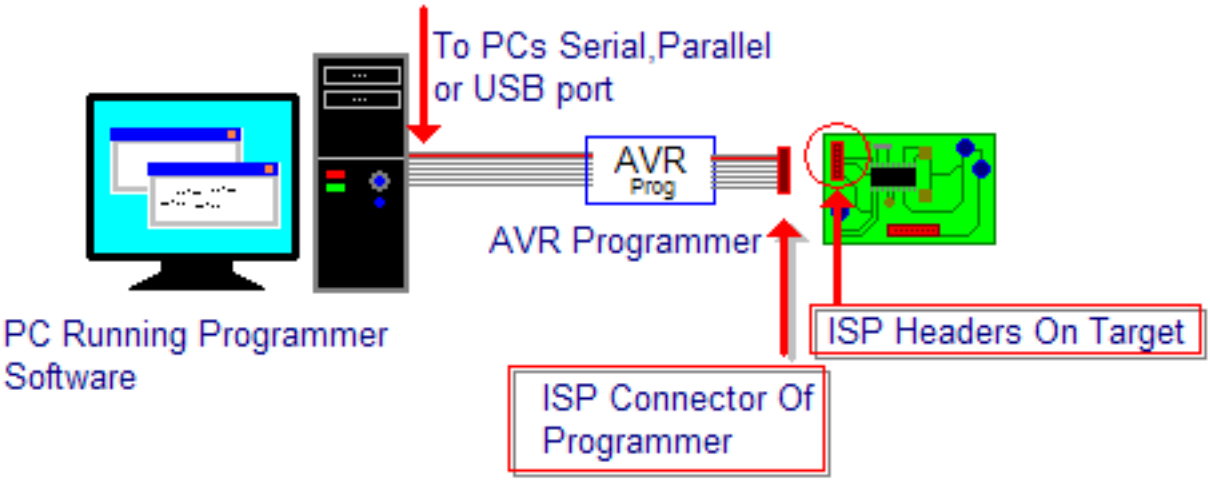
If you don't understand the above features no need to worry. Each will be having it own dedicated tutorials. Just understand they are integrated inside the chip to make your life easier.

Tools Required

The development process of AVR or any other MCU looks like this







Therefore, the tools you need are

Hardware

- A PC running Win98 or Better.
- An In-System programmer (ISP): This is the device which connects your mcu to your PC .You can easily make a serial or parallel port programmer. You can also make a USB programmer (little complicated) or buy one.
- A Target Board: It is nothing but you project which has an ISP header so that programmer can be connected and detached easily. In most simple case, it has a MCU with its power supply, a few basic connections, and ISP header.
- Last but not the least an AVR MCU.

Software

- A C compiler (free)

This will compile your program which is written in a high-level language to machine language. After successful compilation and build you will get a file with .hex which you have to burn into the MCU's flash

- A programmer software free (free)
  This software will help you burn the above mentioned hex file to mcu flash.

That is all for now.

Whats next

In next tutorial, I will tell you how to make a simple programmer, basic AVR board, write and compile your first program, and then burn it to MCU to watch it gooooo…