Micro-Processor/Micro-Controller

It's time to put a step into the world of autonomous robotics.

Now we get the data from the sensors and need to process it to get the desired output to complete the task. There are many Micro-Processors/Micro-Controllers we can use. Ex.-ATmega, Arduino etc. Here we are mentioning the basic Microcontroller

Microcontroller Components/Peripherals

Processor - The processor refers to the Central Processing Unit (CPU) of the microcontroller. It contains the Arithmetic Logic Unit (ALU), Control Unit, Instruction Decoder and some Special Registers (Stack Pointer, Status Register, Program Counter, etc.).

Volatile Memory - This is memory used by ht microcontroller for temporary data storage, system setup and peripherals configurations. Memory in this category includes SRAM and DRAM. AVR microcontrollers utilize SRAM.

Non-Volatile Memory - This is memory used by the microcontroller to store programs. Data can also be stored in this memory but the access time is much slower than that of RAM. Memory in this category includes ROM, PROM, EPROM, EEPROM and FLASH. The AVR microcontrollers utilize Flash for program storage, some AVR controllers contains a bit of EEPROM as well.

Timer Module - Most microcontrollers have at least one timer/counter peripheral. Timer/Counter modules are used to perform timing or counting operations in the controller. These include time stamping, measuring intervals, counting events, etc.

Interrupt Module - Interrupts enable the microcontroller to monitor certain events in the background while executing and application program and react to the event if necessary pausing the original program. This is all coordinated by the interrupt module.

Digital I/O Module - This module allows digital/logic communication with the microcontroller and the external world.

Analog I/O Modules - These modules are use to input/output analog information from/to the external world. Analog modules include Analog Comparators and Analog-to-Digital Converters.

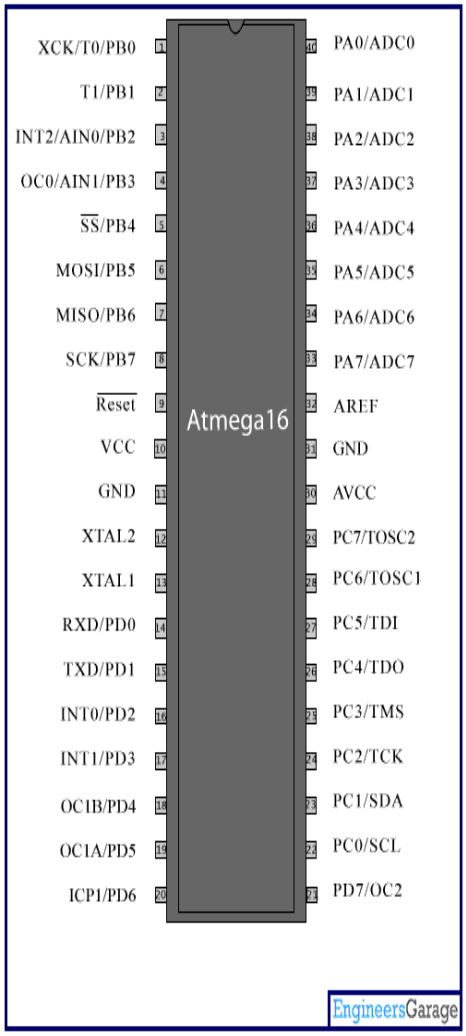
Serial Modules - These modules are used for serial communication with the external world.

ATmega:

 The ATmega is a low-power 8-bit microcontroller based on the AVR architecture.

 By executing powerful instructions in a single clock cycle, the ATmega achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

 This chip has fixed no. of I/O pins which can be used for different purposes.



 ATmega16 has 4 ports namely PORT A, PORT B, PORT C and PORT D.

 The respective pins are namely PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7. Each of these ports comprise of 8 pins.

 Ports are the means by which the micro controller accepts or gives out data.

 So now that you already know the function of 4x8=32 pins of the micro controller.

The rest of the eight pins are VCC, GND, RESET(bar), AVCC, AREF, XTAL1, XTAL2.

1. VCC and GND are the pins for the power supply of micro controller. ATmega16/32 works at 5V, exceeding this voltage at power supply might result in damaging of the IC.

2. The RESET(bar) pin, as it suggests is to reset the micro controller. However, it's an active low pin, i.e. It get's activated when you give it a low signal.

3. AVCC, is the Analog VCC. The micro controller has the feature of Analog to Digital converter. For this feature the micro controller requires an Analog circuit which needs to be given power externally. So, this analog circuit is powered by AVCC.

4. AREF, if the Analog Reference voltage pin. This is again used for ADC purposes.

5. The micro controller by default works at 1Mhz(internal frequency). We can connect an external crystal oscillator to generate higher frequencies and clock pulses. This external oscillator is connected across the XTAL pins (XTAL1 and XTAL2).

In ATmega16, the pins are capable to perform more than one task. For example, youcan see the pins PA0 (ADC0), PB5 (MOSI), PC2 (TCK), PD1 (TXD) etc. These are known as the Alternate Functions of the pins. The Alternate Functions of these pins become active only if you enable certain bits of some registers.

Registers:

Register is a small amount of storage available as part of a micro controller. Such registers are (typically) addressed by mechanisms other than main memory and can be accessed more quickly. These are used to configure the functionality of a micro controller. Since ATmega16/32 are 8-bit microcontrollers, all the registers in them are of 8-bits.

To implement timer through micro controller you have to code accordingly to use this feature. This purpose is fulfilled by registers. Each bit of these registers help in configuring the micro controller for the feature usage. How it's value is set can be referred from the datasheet. All the calculations in the micro controller are also done using 8-bit registers. The details of all the registers are mentioned in the Data sheet and can directly used while coding. The Programming of ATmega uses basic C/C++ to code and need to coded on platforms like AtmelStudio6. Popularly known as AVR Coding and need to be burnt to the ATmega using program burners like Robokits AVR Program Burner, Extreme Burner etc.

Links for Software's:

1) http://www.atmel.com/Microsite/atmel_studio6/compiler_editor.aspx

2) http://robokits-avr-usb-programmer.software.informer.com/2.0/

3) http://extremeelectronics.co.in/avr-tutorials/gui-software-for-usbasp-based-usb-avr-programmers/

Links for Data Sheets:

http://www.atmel.com/Images/Atmel-8154-8-bit-AVR-ATmega16A_Datasheet.pdf

Contact :

- MOHIT DHARIWAL +919933992777
- AYUSH KANWAR +918768702045