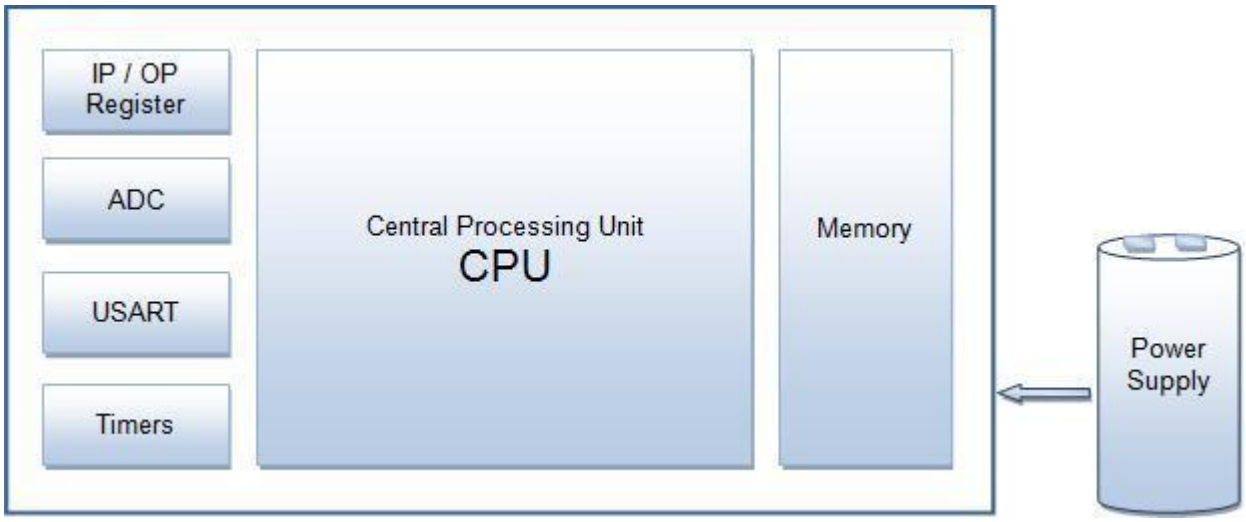
MICROCONTROLLER

Microcontroller can be termed as a system on chip computer which includes number of peripherals like RAM, EEPROM, Timers etc., required to perform some predefined task.



Does this mean that the microcontroller is another name for a computer…? The answer is NO!

The computer on one hand is designed to perform all the general purpose tasks on a single machine like you can use a computer to run a software to perform calculations or you can use a computer to store some multimedia file or to access internet through the browser, whereas the microcontrollers are meant to perform only the specific tasks, for e.g., switching the AC off automatically when room temperature drops to a certain defined limit and again turning it ON when temperature rises above the defined limit.

There are number of popular families of microcontrollers which are used in different applications as per their capability and feasibility to perform the desired task, most common of these are 8051, AVR and PIC microcontrollers. In this article we will introduce you with **AVR** family of microcontrollers.

History of AVR

AVR was developed in the year 1996 by Atmel Corporation. The architecture of AVR was developed by Alf-EgilBogen and VegardWollan. AVR derives its name from its developers and stands for **Alf-EgilBogen VegardWollan RISC microcontroller**, also known as **A**dvanced **V**irtual **R**ISC. The AT90S8515 was the first microcontroller which was based

on **AVR architecture** however the first microcontroller to hit the commercial market was AT90S1200 in the year 1997.

AVR microcontrollers are available in three categories:

1. **Tiny AVR** – Less memory, small size, suitable only for simpler applications

2. Mega AVR – These are the most popular ones having good amount of memory (up to 256 KB), higher number of inbuilt peripherals and suitable for moderate to complex applications.

3. Xmega AVR – Used commercially for complex applications, which require large program memory and high speed.

The following table compares the above mentioned AVR series of microcontrollers:

Series Name	Pins	Flash Memory	Special Feature
TinyAVR	6-32	0.5-8 KB	Small in size
MegaAVR	28-100	4-256KB	Extended peripherals
XmegaAVR	44-100	16-384KB	DMA , Event System included

What's special about AVR?

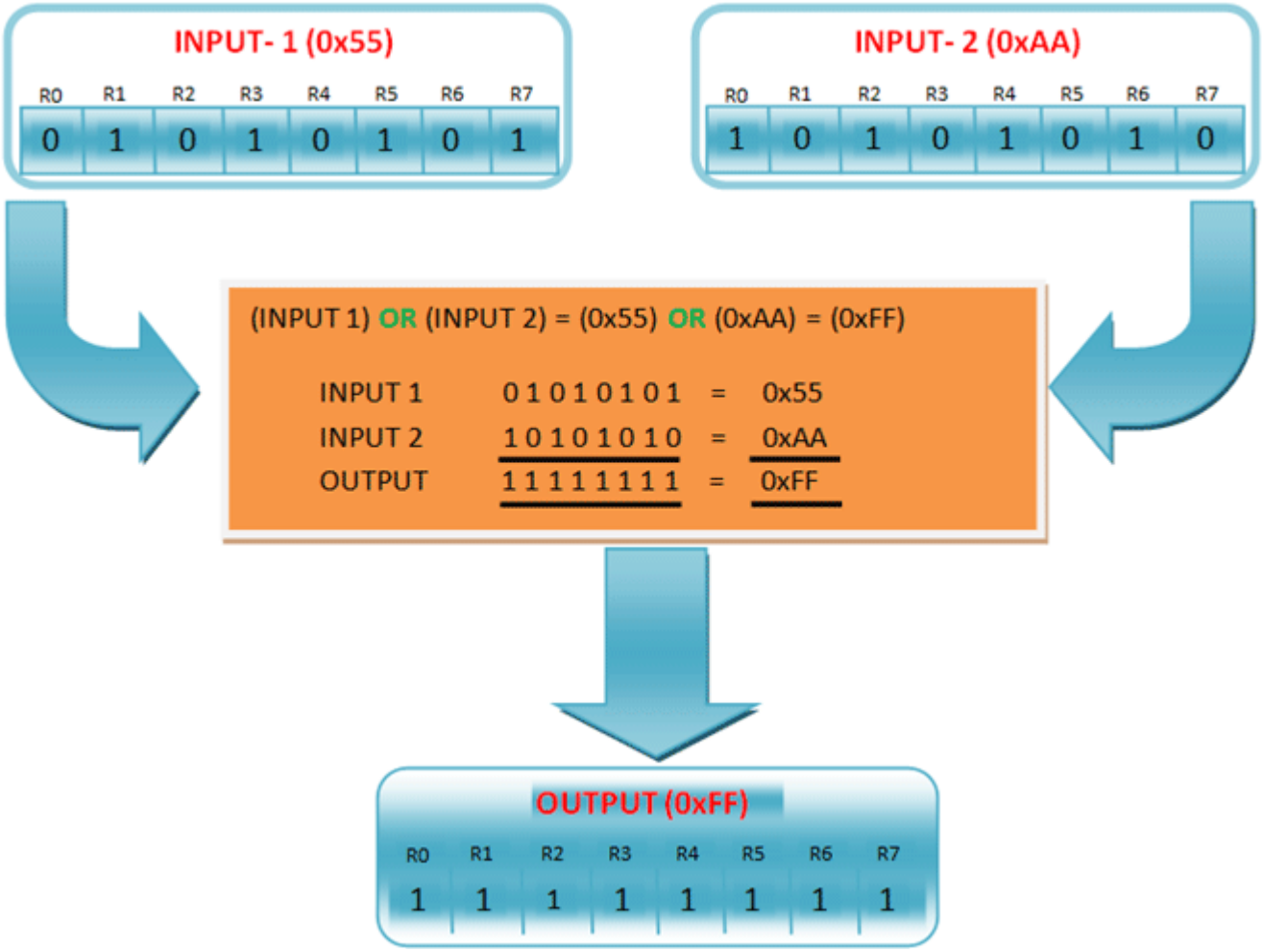
They are fast: **AVR microcontroller** executes most of the instructions in single execution cycle. AVRs are about 4 times faster than PICs; they consume less power and can be operated in different power saving modes. Let's do the comparison between the three most commonly used families of microcontrollers.

	8051	PIC	AVR
SPEED	Slow	Moderate	Fast
MEMORY	Small	Large	Large
ARCHITECTURE	CISC	RISC	RISC
ADC	Not Present	Inbuilt	Inbuilt
Timers	Inbuilt	Inbuilt	Inbuilt
PWM Channels	Not Present	Inbuilt	Inbuilt

AVR is an 8-bit microcontroller belonging to the family of Reduced Instruction Set Computer (**RISC**). In RISC architecture the instruction set of the computer are not only fewer in number but also simpler and faster in operation. The other type of categorization is CISC (Complex

Instruction Set Computers). We will explore more on this when we will learn about the architecture of AVR microcontrollers in following section.

Let's see what this entire means. What is 8-bit? This means that the microcontroller is capable of transmitting and receiving 8-bit data. The input/output registers available are of 8-bits. The AVR families controllers have register based architecture which means that both the operands for an operation are stored in a register and the result of the operation is also stored in a register. Following figure shows a simple example performing OR operation between two input registers and storing the value in Output Register.



The CPU takes values from two input registers INPUT-1 and INPUT-2, performs the logical operation and stores the value into the OUTPUT register. All this happens in 1 execution cycle.

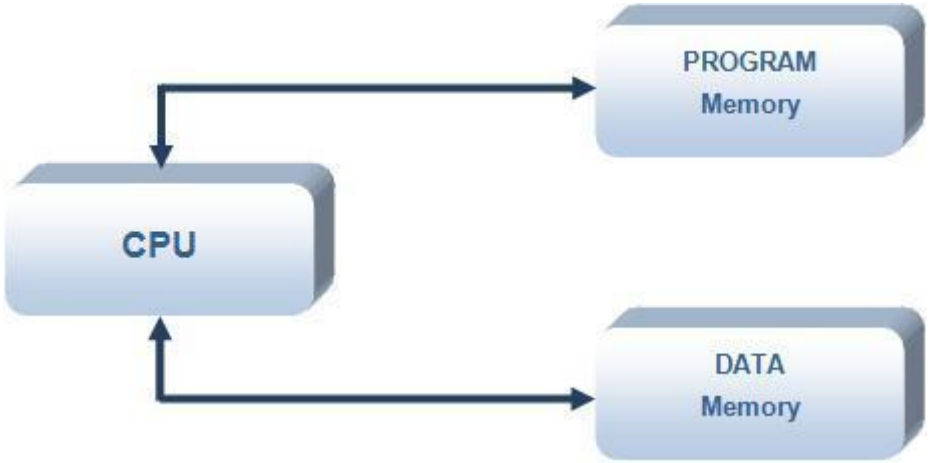
In our journey with the AVR we will be working on Atmega16 microcontroller, which is a 40-pin IC and belongs to the mega AVR category of AVR family. Some of the features of Atmega16 are:

- 16KB of Flash memory
- 1KB of SRAM
- 512 Bytes of EEPROM
- Available in 40-Pin DIP
- 8-Channel 10-bit ADC
- Two 8-bit Timers/Counters
- One 16-bit Timer/Counter
- 4 PWM Channels
- In System Programmer (ISP)
- Serial USART
- SPI Interface
- Digital to Analog Comparator.

Architecture of AVR

The AVR microcontrollers are based on the advanced RISC architecture and consist of 32 x 8-bit general purpose working registers. Within one single clock cycle, AVR can take inputs from two general purpose registers and put them to ALU for carrying out the requested operation, and transfer back the result to an arbitrary register. The ALU can perform arithmetic as well as logical operations over the inputs from the register or between the register and a constant. Single register operations like taking a complement can also be executed in ALU. We can see that AVR does not have any register like accumulator as in 8051 family of microcontrollers; the operations can be performed between any of the registers and can be stored in either of them.

AVR follows Harvard Architecture format in which the processor is equipped with separate memories and buses for Program and the Data information. Here while an instruction is being executed, the next instruction is pre-fetched from the program memory.



Since AVR can perform single cycle execution, it means that AVR can execute 1 million instructions per second if cycle frequency is 1MHz. The higher is the operating frequency of the controller, the higher will be its processing speed. We need to optimize the power consumption with processing speed and hence need to select the operating frequency accordingly.

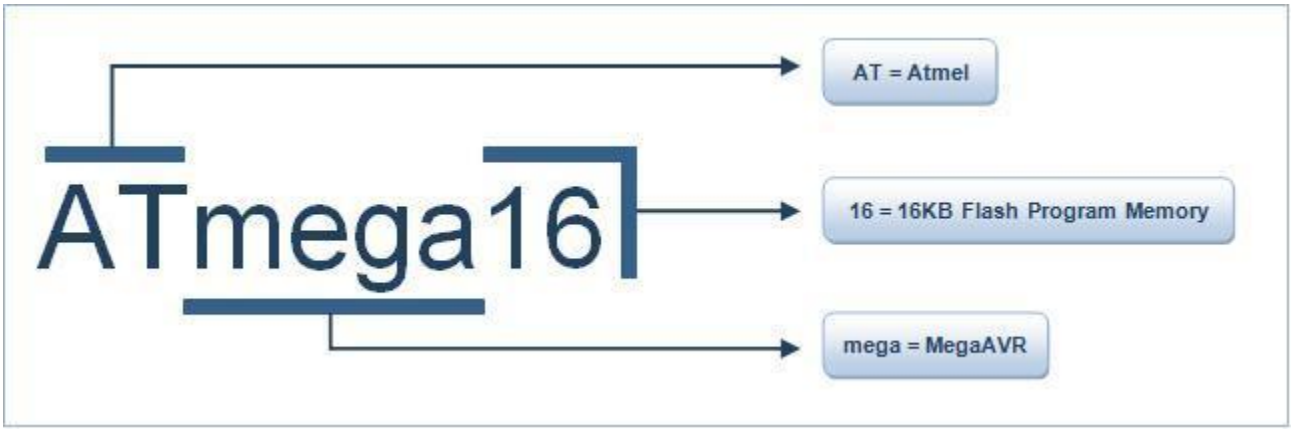
There are two flavors for Atmega16 microcontroller:

1. **Atmega16**:- Operating frequency range is 0 – 16 MHz
2. **Atmega16L**:- Operating frequency range is 0 – 8 MHz

If we are using a crystal of 8 MHz = 8 x 10^6 Hertz = 8 Million cycles, then AVR can execute 8 million instructions.

Naming Convention.!

The **AT** refers to Atmel the manufacturer, **Mega** means that the microcontroller belong to Mega AVR category, **16** signifies the memory of the controller, which is 16KB.



Architecture Diagram: Atmega16

Following points explain the building blocks of **Atmega16 architecture**:

· **I/O Ports:** Atmega16 has four (PORTA, PORTB, PORTC and PORTD) **8-bit** input-output ports.

· **Internal Calibrated Oscillator:** Atmega16 is equipped with an internal oscillator for driving its clock. By default Atmega16 is set to operate at internal calibrated oscillator of 1 MHz The maximum frequency of internal oscillator is 8Mhz. Alternatively, ATmega16 can be operated using an external crystal oscillator with a maximum frequency of 16MHz. In this case you need to modify the fuse bits. (Fuse Bits will be explained in a separate tutorial).

. **ADC Interface:** Atmega16 is equipped with an 8 channel ADC (**Analog to Digital Converter**) with a resolution of 10-bits. ADC reads the analog input for e.g., a sensor input and converts it into digital information which is understandable by the microcontroller.

·**Timers/Counters:** Atmega16 consists of two 8-bit and one 16-bit timer/counter. Timers are useful for generating precision actions for e.g., creating time delays between two operations.

·**Watchdog Timer:** Watchdog timer is present with internal oscillator. Watchdog timer continuously monitors and resets the controller if the code gets stuck at any execution action for more than a defined time interval.

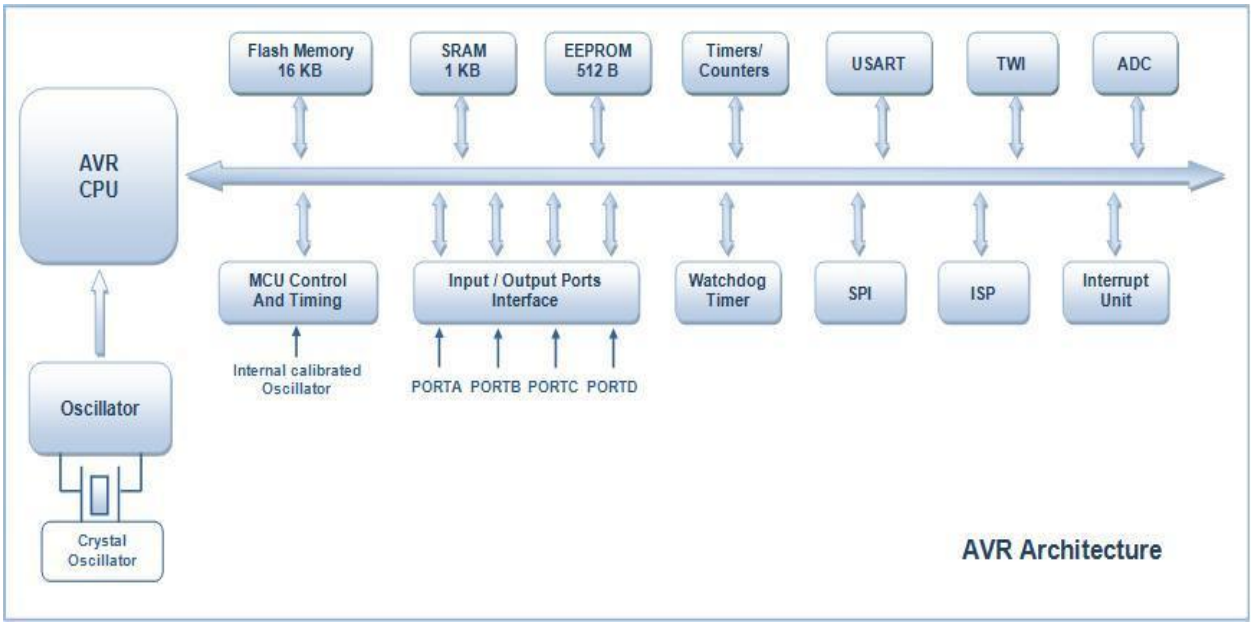
· **Interrupts:** Atmega16 consists of 21 interrupt sources out of which four are external. The remaining are internal interrupts which support the peripherals like USART, ADC, and Timers etc.

.**USART: Universal Synchronous and Asynchronous Receiver and Transmitter** interface is available for interfacing with external device capable of communicating serially (data transmission bit by bit).

·**General Purpose Registers:** Atmega16 is equipped with 32 general purpose registers which are coupled directly with the Arithmetic Logical Unit (ALU) of CPU.

· **ISP:** AVR family of controllers have **In System Programmable** Flash Memory which can be programmed without removing the IC from the circuit, ISP allows to reprogram the controller while it is in the application circuit.

.**DAC:** Atmega16 is also equipped with a **Digital to Analog Converter** (DAC) interface which can be used for reverse action performed by ADC. DAC can be used when there is a need of converting a digital signal to analog signal.



. **Memory:** Atmega16 consist of three different memory sections:

1. Flash EEPROM: Flash EEPROM or simple flash memory is used to store the program dumped or burnt by the user on to the microcontroller. It can be easily erased electrically as a single unit. Flash memory is non-volatile i.e., it retains the program even if the power is cut-off. Atmega16 is available with 16KB of in system programmable Flash EEPROM.

2. Byte Addressable EEPROM: This is also a nonvolatile memory used to store data like values of certain variables. Atmega16 has 512 bytes of EEPROM; this memory can be useful for storing the lock code if we are designing an application like electronic door lock.

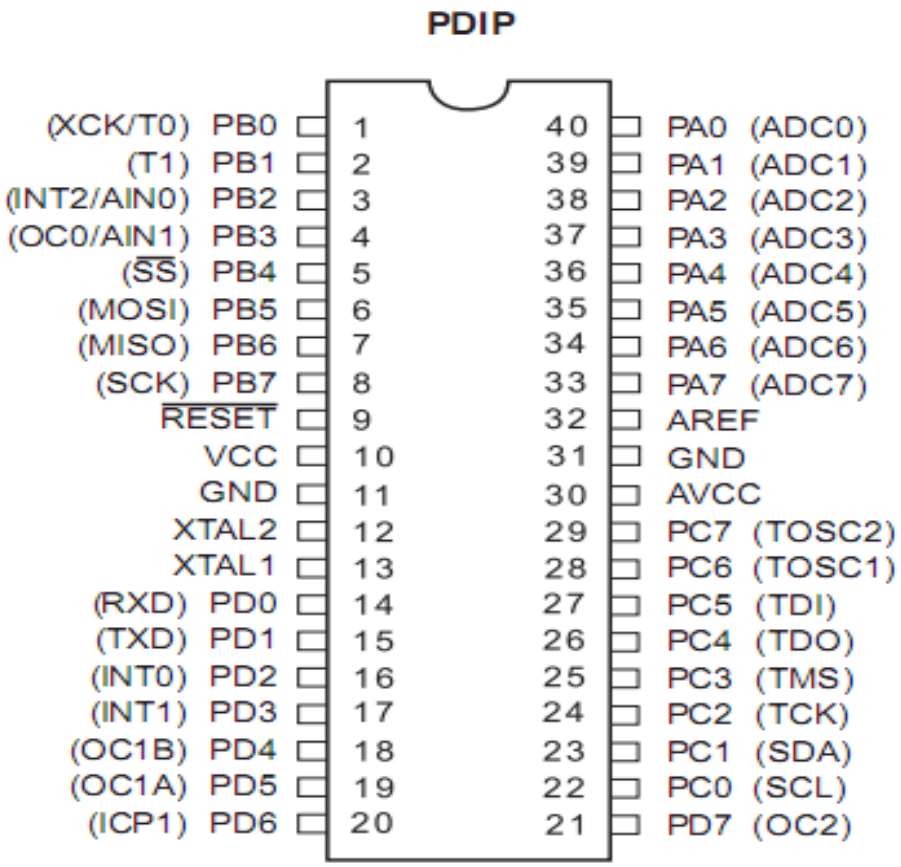
3. SRAM: Static Random Access Memory, this is the volatile memory of microcontroller i.e., data is lost as soon as power is turned off. Atmega16 is equipped with 1KB of internal SRAM. A small portion of SRAM is set aside for general purpose registers used by CPU and some for the peripheral subsystems of the microcontroller.

· **SPI: Serial Peripheral Interface**, SPI port is used for serial communication between two devices on a common clock source. The data transmission rate of SPI is more than that of USART.

· **TWI: Two Wire Interface** (TWI) can be used to set up a network of devices, many devices can be connected over TWI interface forming a network, the devices can simultaneously transmit and receive and have their own unique address.

PIN DIAGRAM :

Pinout ATmega16



Pin Descriptions

VCC Digital supply voltage.

GND Ground.

Port A (PA7...PA0) Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7...PB0) Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C (PC7...PC0) Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5 (TDI), PC3 (TMS) and PC2(TCK) will be activated even if a reset occurs.

Port D (PD7...PD0) Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both

a	high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when reset condition becomes active, even if the clock is not running.
RESET	Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.
XTAL1:	Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
XTAL2:	Output from the inverting Oscillator amplifier.
AVCC:	AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.
AREF:	AREF is the analog reference pin for the A/D Converter.