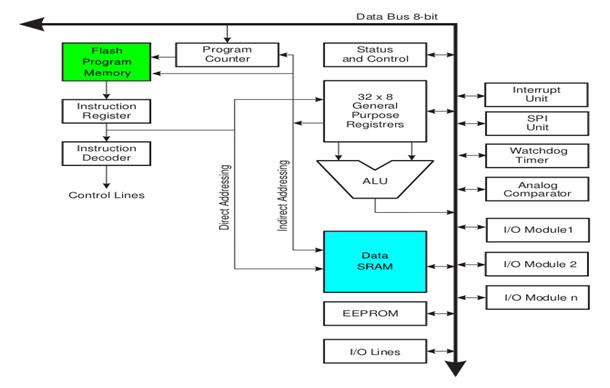
***Arduino – Basics and Design***

**Defining Arduino**

An Arduino is actually a microcontroller based kit which can be either used directly by purchasing from the vendor or can be made at home using the components, owing to its open source hardware feature. It is basically used in communications and in controlling or operating many devices. It was founded by Massimo Banzi and David Cuartielles in 2005.

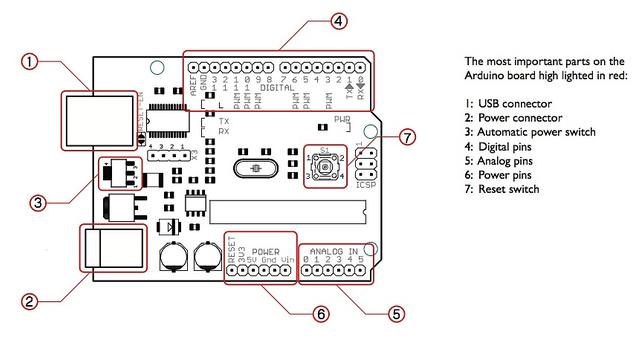
**Arduino Architecture:**

Arduino’s processor basically uses the Harvard architecture where the program code and program data have separate memory. It consists of two memories- Program memory and the data memory. The code is stored in the flash program memory, whereas the data is stored in the data memory. The Atmega328 has 32 KB of flash memory for storing code (of which 0.5 KB is used for the boot loader), 2 KB of SRAM and 1 KB of EEPROM and operates with a clock speed of 16MHz.

**Arduino Architecture**

**Arduino Pin Diagram**

A typical example of Arduino board is Arduino Uno. It consists of ATmega328- a 28 pin microcontroller.



Arduino Pin Diagram

Arduino Uno consists of 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button

**Power Jack**:  Arduino can be power either from the pc through a USB or through external source like adaptor or a battery. It can operate on a external supply of 7 to 12V. Power can be applied externally through the pin Vin or by giving voltage reference through the IORef pin.

**Digital Inputs**: It consists of 14 digital inputs/output pins, each of which provide or take up 40mA current. Some of them have special functions like pins 0 and 1, which act as Rx and Tx respectively , for serial communication, pins 2 and 3-which are external interrupts, pins 3,5,6,9,11 which provides pwm output and pin 13 where LED is connected.

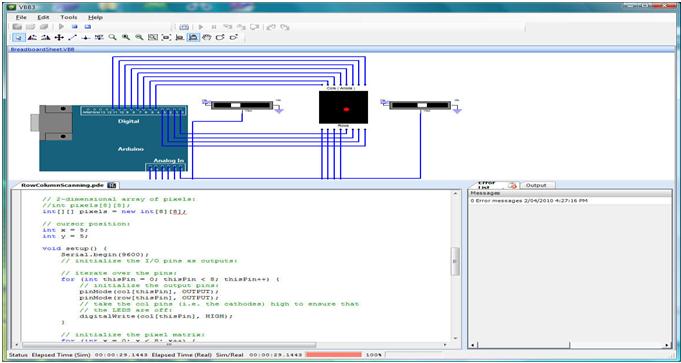
**Analog inputs**: It has 6 analog input/output pins, each providing a resolution of 10 bits.

**ARef**: It provides reference to the analog inputs

**Reset**: It resets the microcontroller when low.

**How to program an Arduino?**

The most important advantage with Arduino is the programs can be directly loaded to the device without requiring any hardware programmer to burn the program. This is done because of the presence of the 0.5KB of Bootloader which allows the program to be burned into the circuit. All we have to do is to download the Arduino software and writing the code.



The Arduino tool window consists of the toolbar with the buttons like verify, upload, new, open, save, serial monitor. It also consists of a text editor to write the code, a message area which displays the feedback like showing the errors, the text console which displays the output and a series of menus like the File, Edit, Tools menu.

**5 Steps to program an Arduino**

* Programs written in Arduino are known as sketches. A basic sketch consists of 3 parts

1. Declaration of Variables  
2. Initialization: It is written in the setup () function.  
3. Control code: It is written in the loop () function.

* The sketch is saved with .ino extension. Any operations like verifying, opening a sketch, saving a sketch can be done using the buttons on the toolbar or using the tool menu.
* The sketch should be stored in the sketchbook directory.
* Chose the proper board from the tools menu and the serial port numbers.
* Click on the upload button or chose upload from the tools menu. Thus the code is uploaded by the bootloader onto the microcontroller.

**Few of basic Adruino functions are:**

* **digitalRead**(pin): Reads the digital value at the given pin.
* **digitalWrite**(pin, value): Writes the digital value to the given pin.
* **pinMode**(pin, mode): Sets the pin to input or output mode.
* **analogRead**(pin): Reads and returns the value.
* **analogWrite**(pin, value): Writes the value to that pin.
* **serial.begin**(baud rate): Sets the beginning of serial communication by setting the bit rate.

**How to Design your own Arduino?**

We can also design our own Arduino by following the schematic given by the Arduino vendor and also available at the websites. All we need are the following components- A breadboard, a led, a power jack, a IC socket, a microcontroller, few resistors, 2 regulators, 2 capacitors.

* The IC socket and the power jack are mounted on the board.
* Add the 5v and 3.3v regulator circuits using the combinations of regulators and capacitors.
* Add proper power connections to the microcontroller pins.
* Connect the reset pin of the IC socket to a 10K resistor.
* Connect the crystal oscillators to pins 9 and 10
* Connect the led to the appropriate pin.
* Mount the female headers onto the board and connect them to the respective pins on the chip.
* Mount the row of 6 male headers, which can be used as an alternative to upload programs.
* Upload the program on the Microcontroller of the readymade Adruino and then pry it off and place back on the user kit.

**7 Reasons why Arduino is being preferred these days**

1. It is inexpensive
2. It comes with an open source hardware feature which enables users to develop their own kit using already available one as a reference source.
3. The Arduino software is compatible with all types of operating systems like Windows, Linux, and Macintosh etc.
4. It also comes with open source software feature which enables experienced software developers to use the Arduino code to merge with the existing programming language libraries and can be extended and modified.
5. It is easy to use for beginners.
6. We can develop an Arduino based project which can be completely stand alone or projects which involve direct communication with the software loaded in the computer.
7. It comes with an easy provision of connecting with the CPU of the computer using serial communication over USB as it contains built in power and reset circuitry.

So this is some basic idea regarding an Arduino. You can use it for many types of applications. For instance in applications involving controlling some actuators like motors, generators, based on the input from sensors.