ICE 4071: Industrial Internet of Things (IIoT)

Arduino & Raspberry Pi

Dr. S. Meenatchisundaram

Email: meenasundar@gmail.com

What is an Arduino?

Open Source electronic prototyping **platform** based on flexible **easy to use** hardware and software.

What is an Arduino?

- ❑ Arduino is an open-source physical computing platform.
- ❑ It is a small microcontroller board with a USB plug.
- ❑ Based on a simple i/o board and a development environment that implements the Processing/writing language.
- ❑ Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer.
- ❑ Easy-to-use hardware and software.

What is an Arduino?

- It’s intended for students, artists, designers, hobbyists and anyone who tinker with technology.

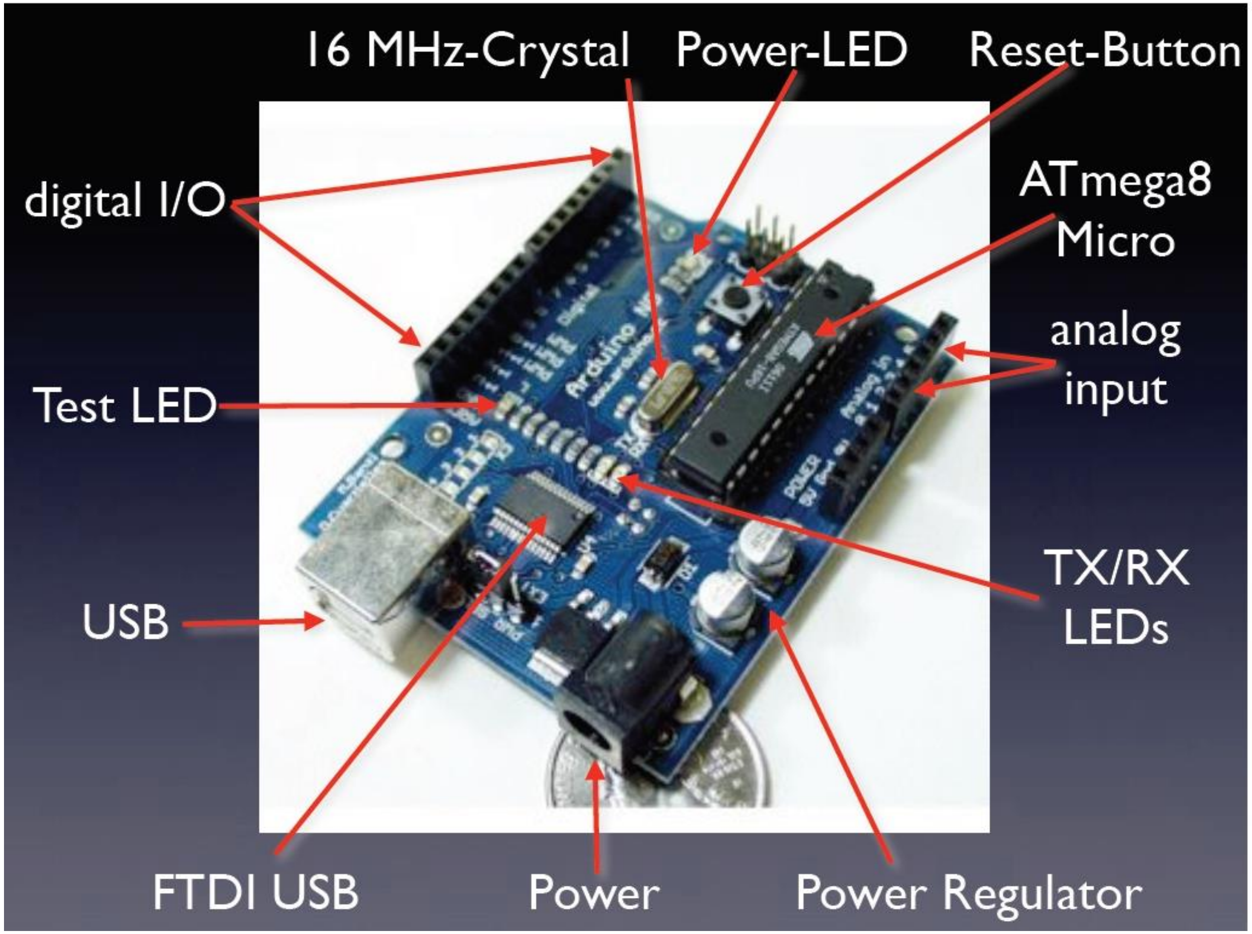
- It is programmed in Arduino Programming language(APL) similar to C/C++.

- Way more easy to program compared to other microcontroller packages.

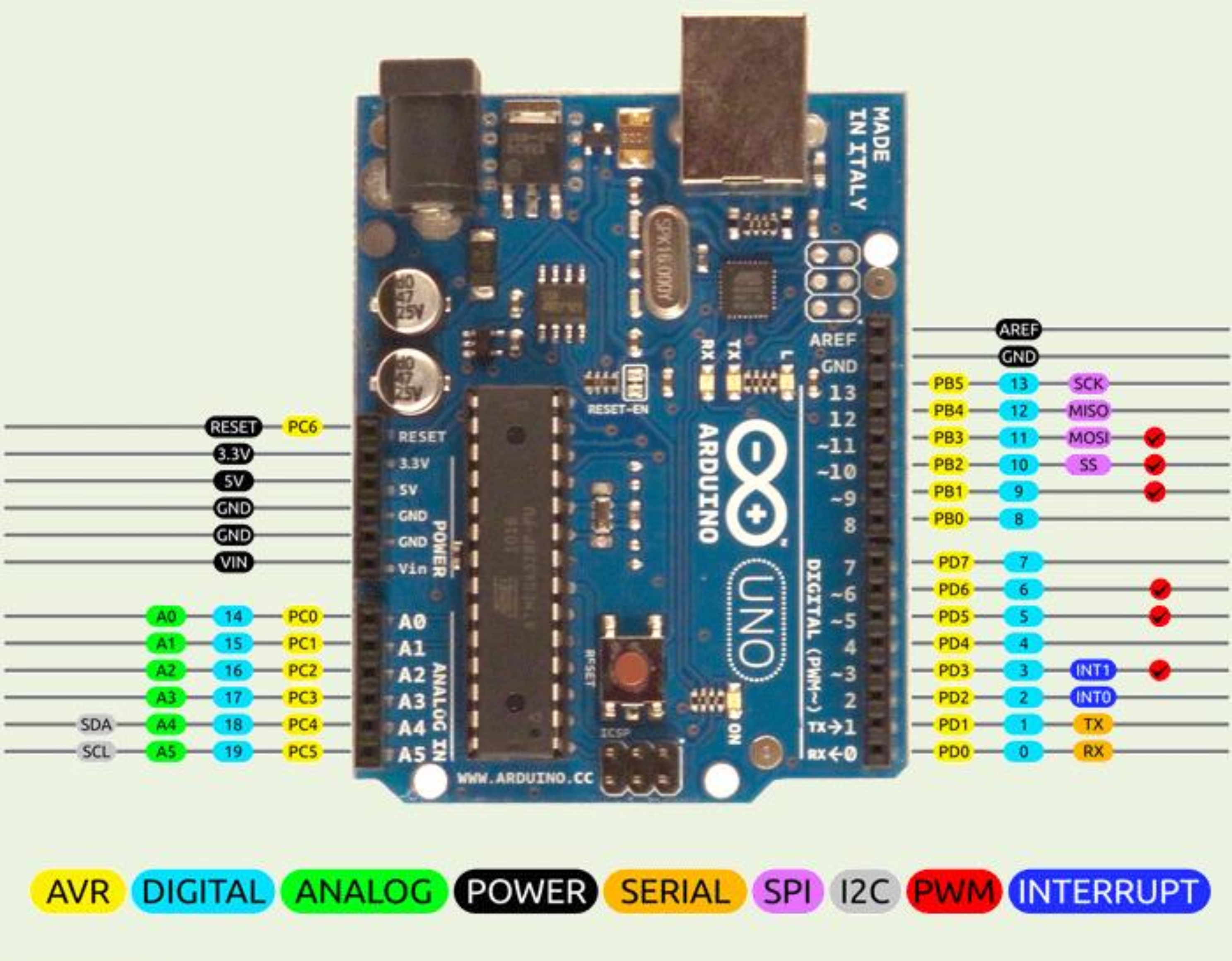
- The Arduino is a microcontroller development platform (not a microcontroller….)

- It is the winner of “**worlds best interaction award 2012**” sponsored by Google

Arduino


16 MHz-Crystal
Power-LED
Reset-Button
digital I/O
ATmega8
Micro
analog
input
Test LED
USB
TX/RX
LEDs
FTDI USB
Power
Power Regulator

Pinout

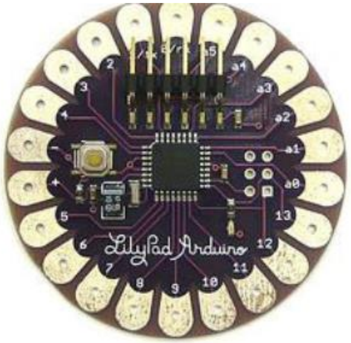


2014 by Bouni
Photo by Arduino.cc

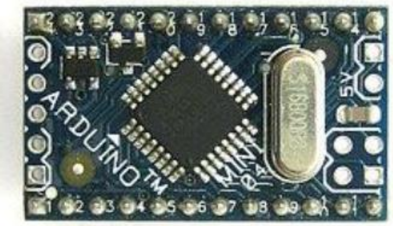
Different flavors!!!

- ❑ There are many versions of Arduino board. Versions differ by size, microcontroller, etc.

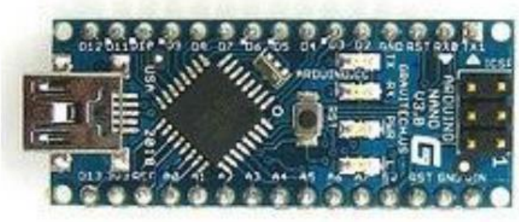
MEGA



LILYPAD



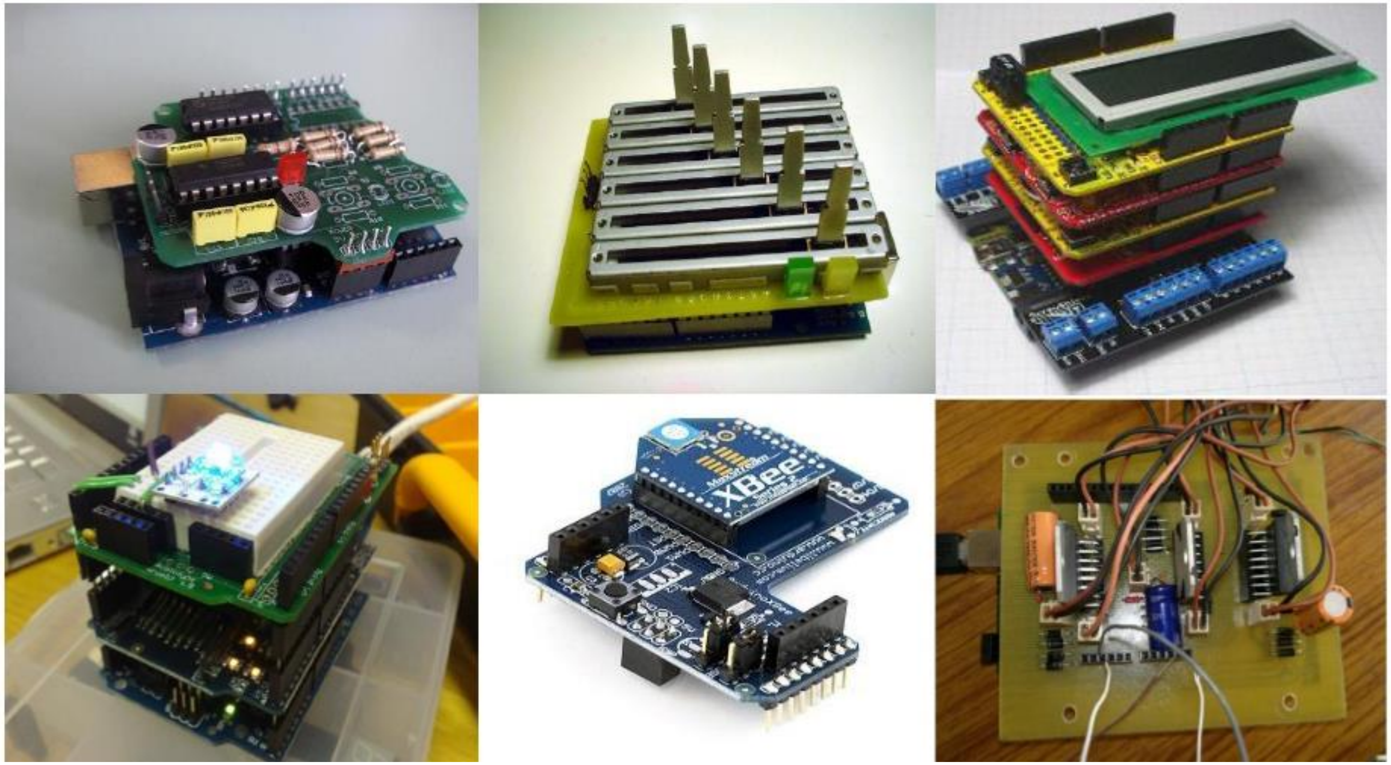
MINI



NANO 43mm x 18mm

Shields

- Printed circuit boards that sit atop an arduino
- Plug into the normally supplied pin-headers of arduino.
- Th
- Fo



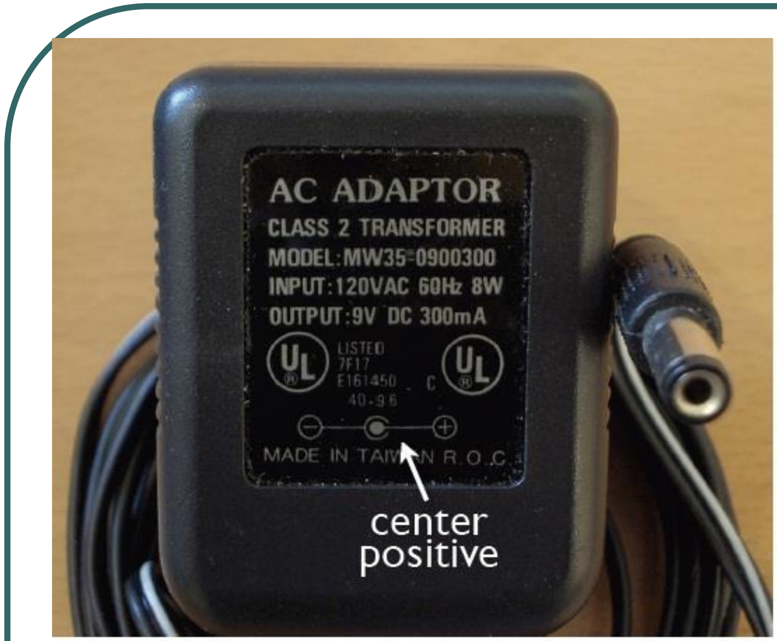
External power

- ❑ Should be between 9V and 12V DC.

- ❑ Must be rated for a minimum of 250mA current output.

- ❑ Must have a 2.1mm power plug on the Arduino end.

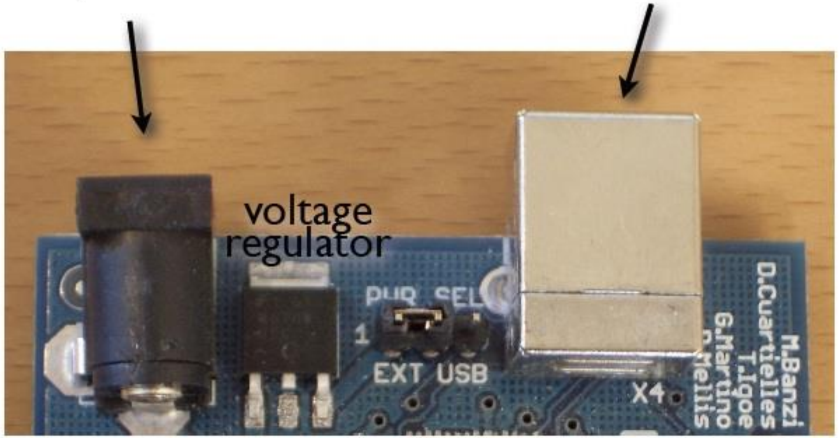
- ❑ The plug must be “centre positive”,that is,the middle pin of the plug has to be the + connection



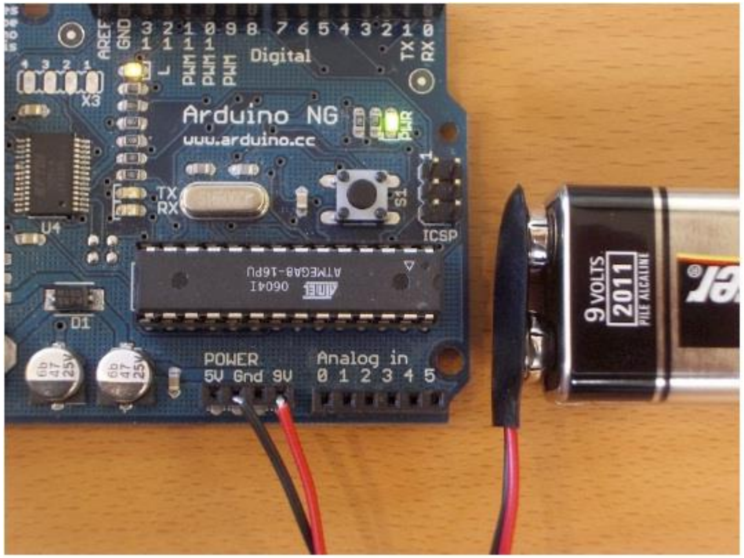
center positive

External power connector

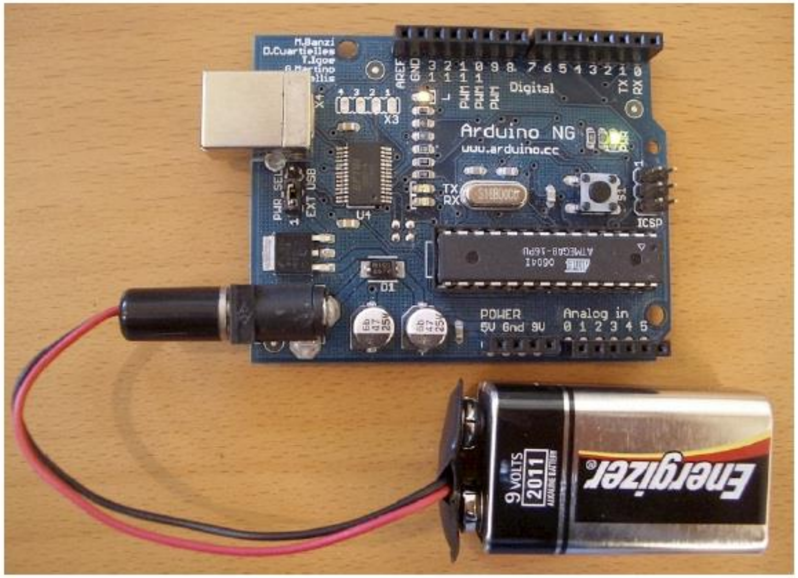
USB connector



voltage regulator



Be careful about polarity! And shorts!



also solves polarity concerns

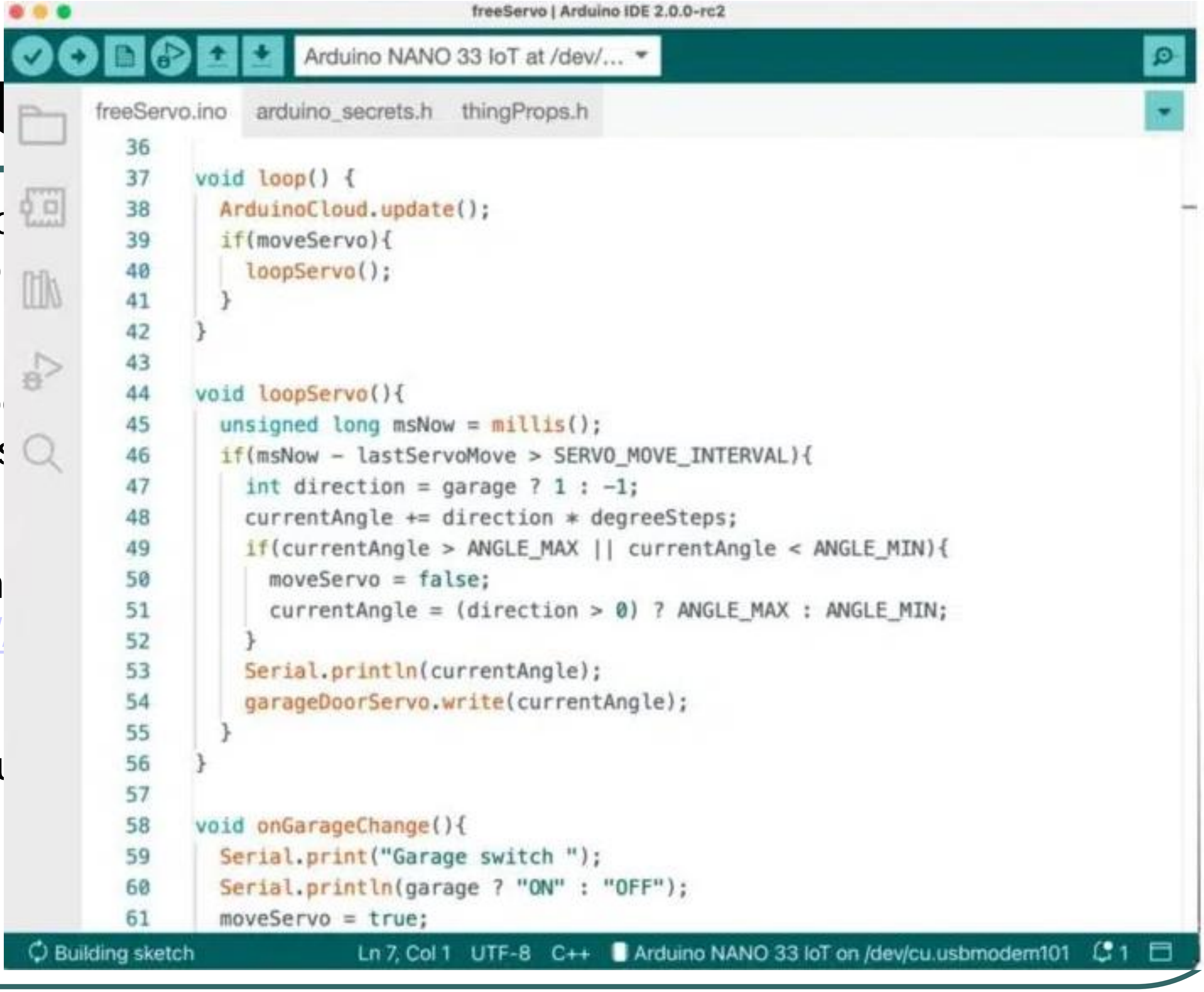
ARDU

❑ Arduino bo
is a versio

❑ Arduino lo
differences

❑ Arduino In
from http:/

❑ Currently u


freeServo | Arduino IDE 2.0.0-rc2
Arduino NANO 33 IoT at /dev/...
freeServo.ino
arduino_secrets.h
thingProps.h

```
void loop() {
  ArduinoCloud.update();
  if(moveServo){
    loopServo();
  }
}

void loopServo(){
  unsigned long msNow = millis();
  if(msNow - lastServoMove > SERVO_MOVE_INTERVAL){
    int direction = garage ? 1 : -1;
    currentAngle += direction * degreeSteps;
    if(currentAngle > ANGLE_MAX || currentAngle < ANGLE_MIN){
      moveServo = false;
      currentAngle = (direction > 0) ? ANGLE_MAX : ANGLE_MIN;
    }
    Serial.println(currentAngle);
    garageDoorServo.write(currentAngle);
  }
}

void onGarageChange(){
  Serial.print("Garage switch ");
  Serial.println(garage ? "ON" : "OFF");
  moveServo = true;
```

The Arduino IDE

- The arduino is programmed in C language.
- The language is very simple and provides many abstraction for simplicity of reading and writing powerful applications.
- It provides a serial monitor to see the serial data from the USB virtual COM port.
- Allows one click compiling, verification and burning of code onto the arduino.

Arduino Programming language v/s Processing

❑ Arduino has two reserved functions:

1. void setup()
2. void loop()

❑ There is no pop-up display window, hence void draw() is not special.

Loop() can be considered to do the same thing as draw() for the arduino.

❑ There are three types of variable in Arduino:

i. char

ii. int

iii. long

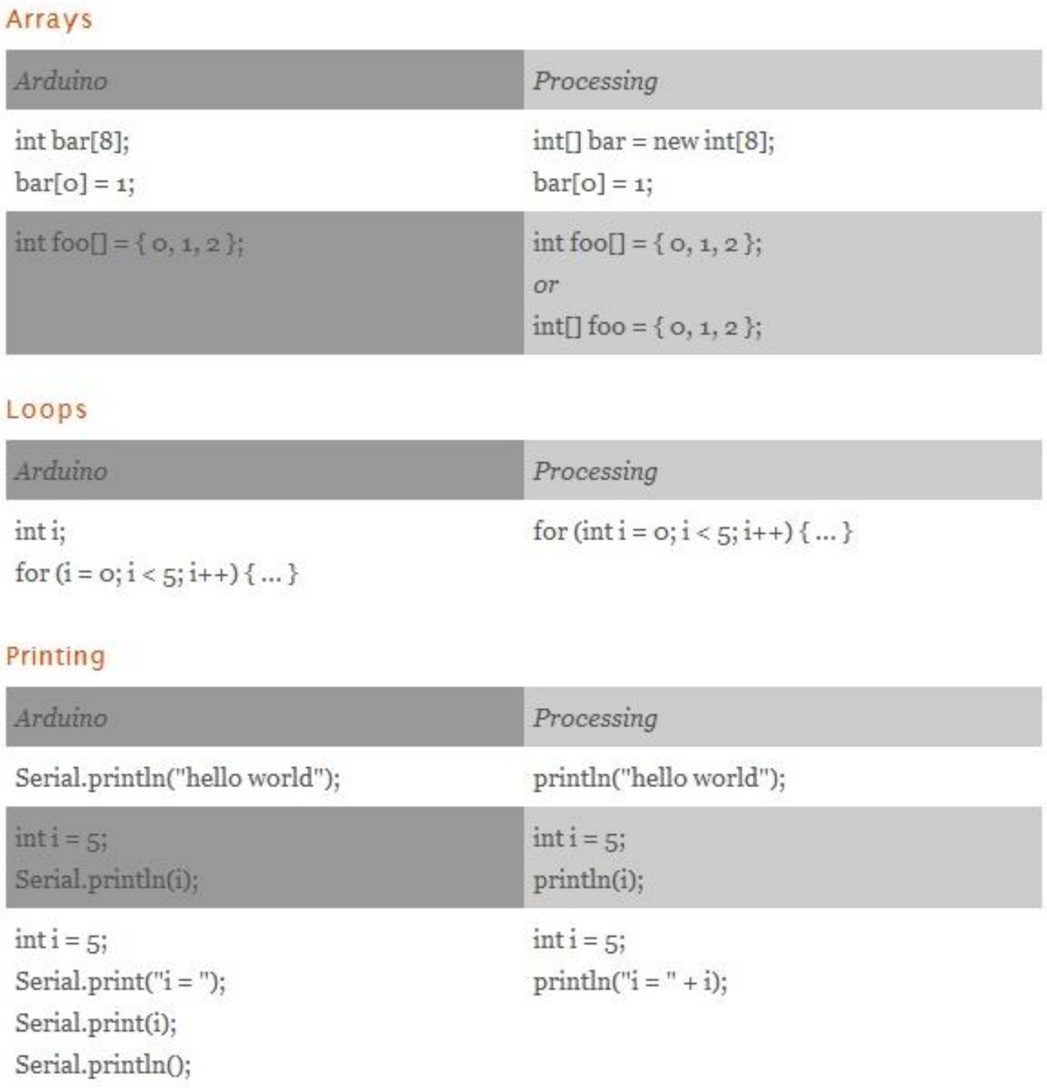
❑ Arduino has a few reserved constants, which do not need to be defined:

1. HIGH//5 volts
2. LOW//0 volts
3. INPUT//pin is input
4. OUTPUT//pin is output

❑ Conditional statements are the same as in Processing.

❑ Functions can be defined the same as in Processing

Arduino Programming language v/s Processing



Arrays

Arduino	*Processing*

int bar[8];
bar[0] = 1;
	int[] bar = new int[8];
	bar[0] = 1;

int foo[] = { 0, 1, 2 };	int foo[] = { 0, 1, 2 };
	or
	int[] foo = { 0, 1, 2 };

Loops

Arduino	*Processing*

int i;
for (i = 0; i < 5; i++) { ... }
	for (int i = 0; i < 5; i++) { ... }

Printing

Arduino	*Processing*
Serial.println("hello world");	println("hello world");

int i = 5;
Serial.println(i);
	int i = 5;
	println(i);

int i = 5;
Serial.print("i = ");
Serial.print(i);
Serial.println();
	int i = 5;
	println("i = " + i);

Steps in Arduino programming

- ❑ Open the IDE
- ❑ Write code and logic
- ❑ Click the verify/compile button to check your program for errors
- ❑ Attach the arduino via USB to the PC
- ❑ Install drivers if first time
- ❑ Setup serial port being used.
- ❑ Setup board which we need to program.
- ❑ Click upload code to send code to arduino.

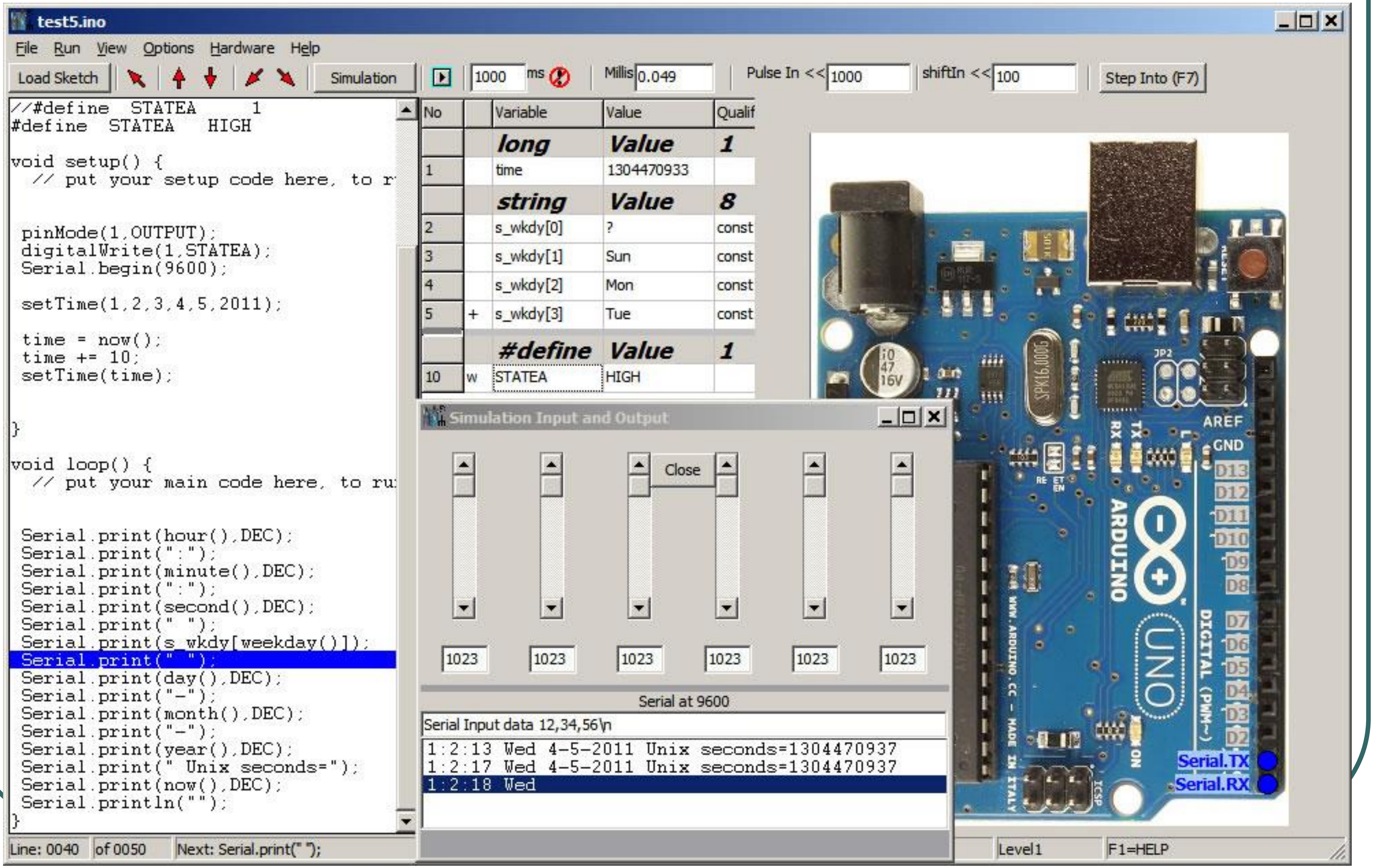
Arduino - Simulator

❑ "**simulator for Arduino v0.95**" is the simulator software to make virtual implementation of the Arduino.

❑ The benefits and features are:

1. The ability to teach and demonstrate the inner workings of an Arduino sketch
2. Test out a sketch without the hardware, or prior to purchasing hardware
3. Debug a sketch
4. Demonstrate a project to a potential customer
5. Develop a complicated sketch faster than using the hardware

Simulator for Arduino v0.95



1000 ms

Millis 0.049

Pulse In << 1000

shiftIn << 100

```
//#define  STATEA     1
#define  STATEA   HIGH

void setup() {
  // put your setup code here, to r

 pinMode(1,OUTPUT);
 digitalWrite(1,STATEA);
 Serial.begin(9600);

 setTime(1,2,3,4,5,2011);

 time = now();
 time += 10;
 setTime(time);


}

void loop() {
  // put your main code here, to ru


 Serial.print(hour(),DEC);
 Serial.print(":");
 Serial.print(minute(),DEC);
 Serial.print(":");
 Serial.print(second(),DEC);
 Serial.print(" ");
 Serial.print(s_wkdy[weekday()]);
 Serial.print(" ");
 Serial.print(day(),DEC);
 Serial.print("-");
 Serial.print(month(),DEC);
 Serial.print("-");
 Serial.print(year(),DEC);
 Serial.print(" Unix seconds=");
 Serial.print(now(),DEC);
 Serial.println("");
}
```

No		Variable	Value	Qualif
		long	***Value***	***1***
1		time	1304470933	
		string	***Value***	***8***
2		s_wkdy[0]	?	const
3		s_wkdy[1]	Sun	const
4		s_wkdy[2]	Mon	const
5	+	s_wkdy[3]	Tue	const
		#define	***Value***	***1***
10	w	STATEA	HIGH	

1023 1023 1023 1023 1023 1023

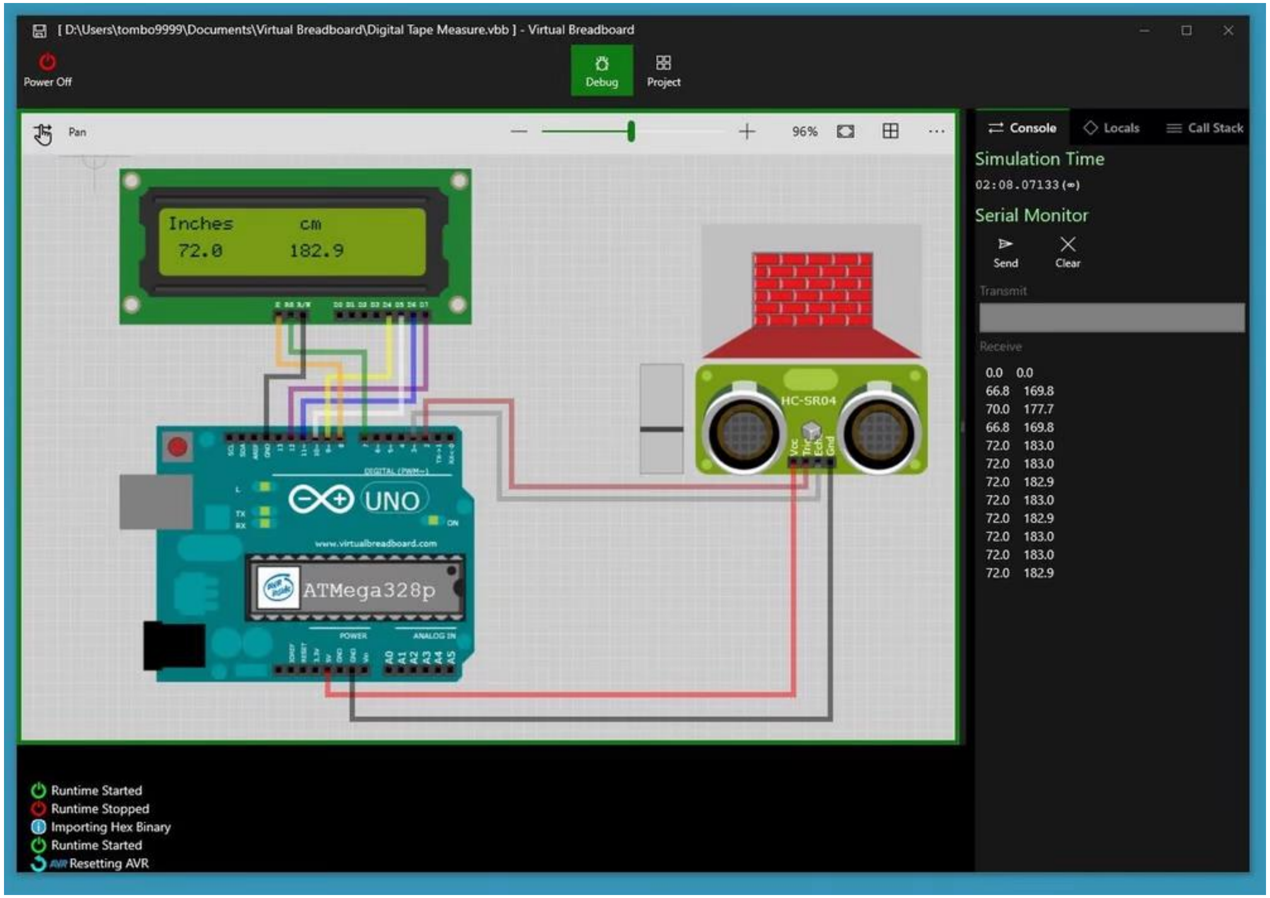
Serial at 9600

Serial Input data 12,34,56\n

```
1:2:13 Wed 4-5-2011 Unix seconds=1304470937
1:2:17 Wed 4-5-2011 Unix seconds=1304470937
1:2:18 Wed
```

Line: 0040 | of 0050 | Next: Serial.print(" "); | Level1 | F1=HELP

Simulator for Arduino



[D:\Users\tombo9999\Documents\Virtual Breadboard\Digital Tape Measure.vbb] - Virtual Breadboard

Simulation Time

```
02:08.07133 (∞)
```

Serial Monitor

Transmit

Receive

```
0.0   0.0
66.8   169.8
70.0   177.7
66.8   169.8
72.0   183.0
72.0   183.0
72.0   182.9
72.0   183.0
72.0   182.9
72.0   183.0
72.0   183.0
72.0   182.9
```

```
Inches      cm
 72.0      182.9
```

```
Runtime Started
Runtime Stopped
Importing Hex Binary
Runtime Started
AVR Resetting AVR
```

Why Arduino?

- ❑ It is Open Source,both in terms of **Hardware** and **Software.**
- ❑ It is cheap,(about $20,the cost of going out for pizza)
- ❑ USB connectivity(MacBooks don’t have serial ports)
- ❑ More powerful than a BASIC stamp(it costs around $180)
- ❑ Simple and easy to use by someone without formal electronics training. Editing and rewriting is often easier than writing from scratch.

Getting started with Programming

Bare minimum code

```
void setup() {
    // put your setup code here, to run
  once:
}


void loop() {
    // put your main code here, to run
  repeatedly:
}
```

Bare minimum code

setup : It is called only when the Arduino is powered on or reset. It is used to initialize variables and pin modes

loop : The loop functions runs continuously till the device is powered off. The main logic of the code goes here. Similar to while (1) for micro-controller programming.

PinMode

A pin on arduino can be set as input or output by using pinMode function.

```
pinMode(13, OUTPUT); // sets pin 13 as output pin
```

```
pinMode(13, INPUT); // sets pin 13 as input pin
```

Reading/writing digital values

```
digitalWrite(13, LOW); // Makes the output voltage on pin 13 , 0V
```

```
digitalWrite(13, HIGH); // Makes the output voltage on pin 13 , 5V
```

```
int buttonState = digitalRead(2); // reads the value of pin 2 in buttonState
```

Analog to Digital Conversion

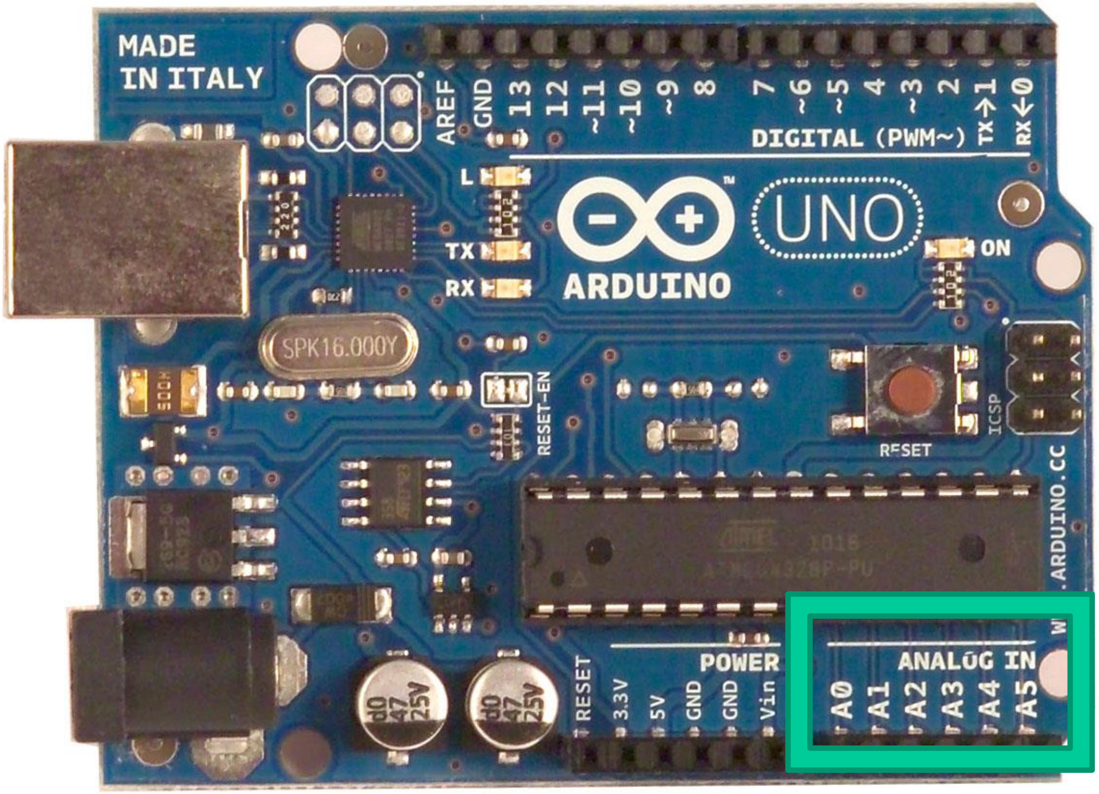
What is analog ?

It is continuous range of voltage values (not just 0 or 5V)

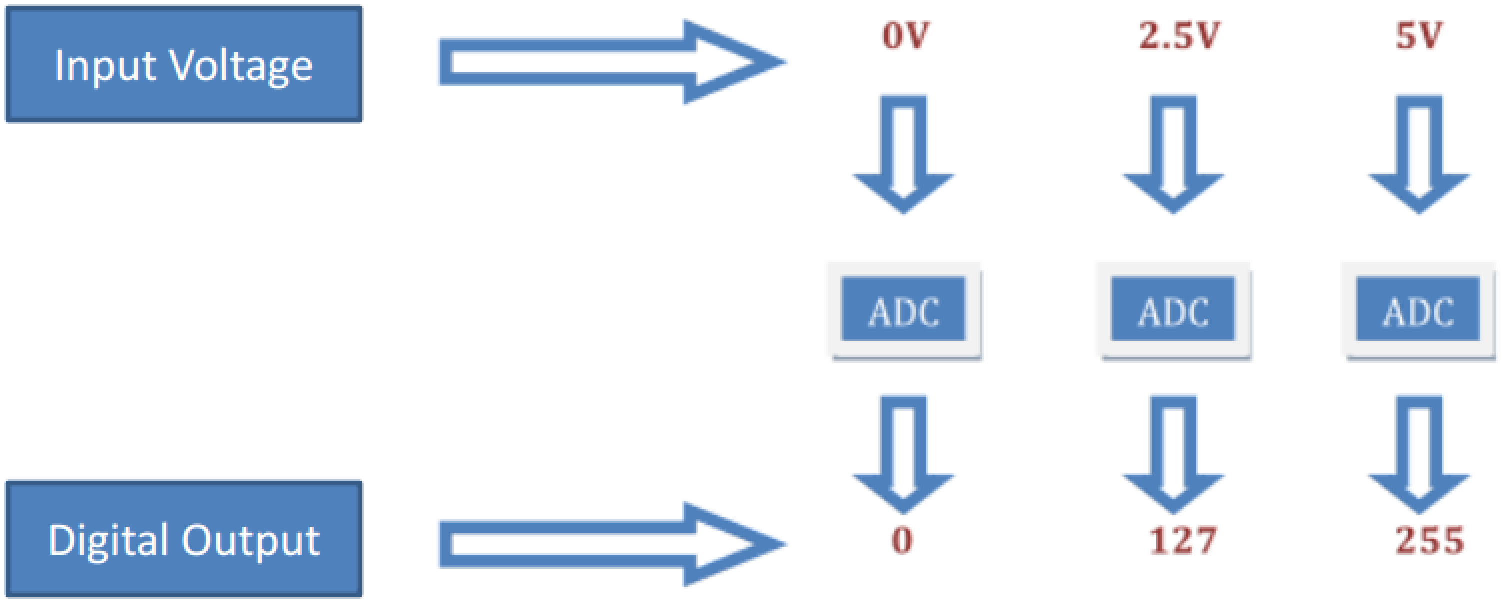
Why convert to digital ?

Because our microcontroller only understands digital.

ADC in Arduino Uno

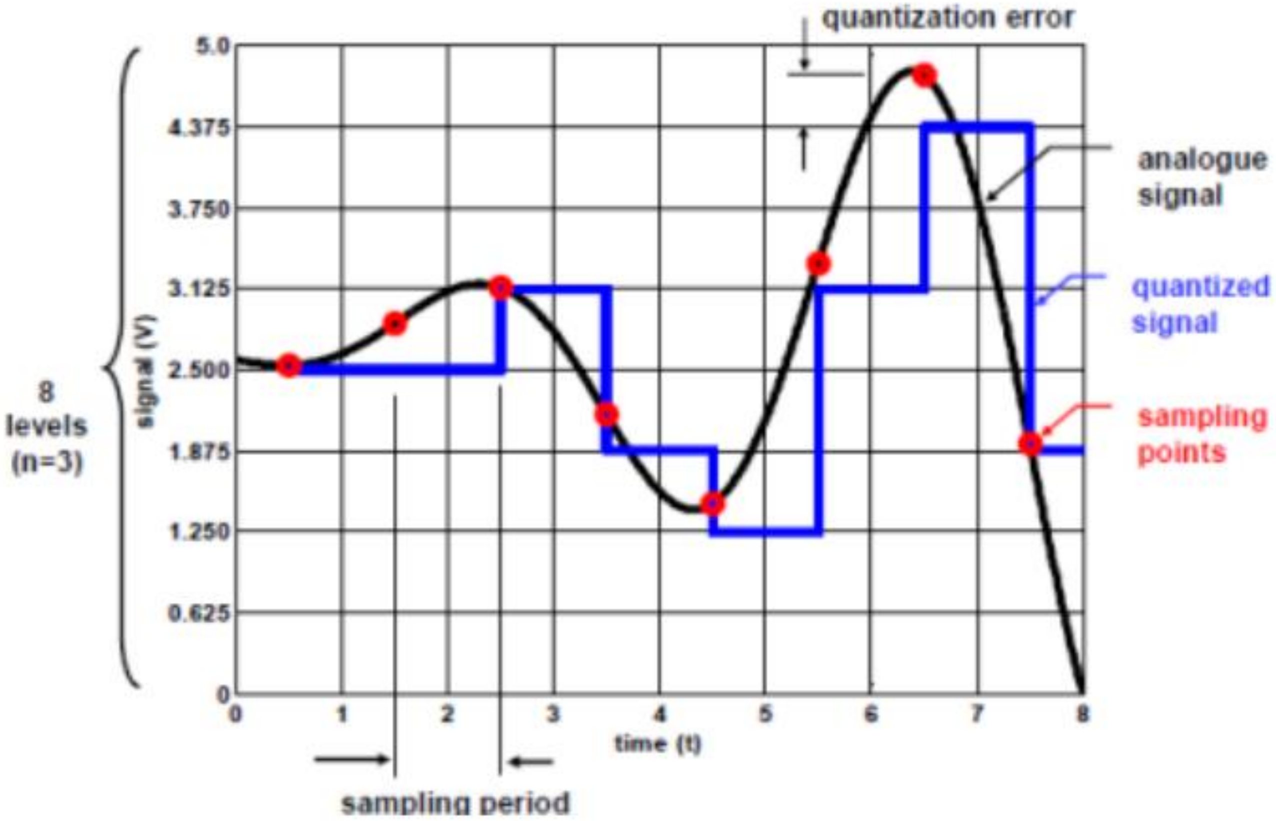


Converting Analog Value to Digital


Input Voltage

Digital Output

Quantization the signal



ADC in Arduino

The Arduino Uno board contains 6 pins for ADC

10-bit analog to digital converter

This means that it will map input voltages between 0 and 5 volts into integer values between 0 and 1023

Reading/Writing Analog Values

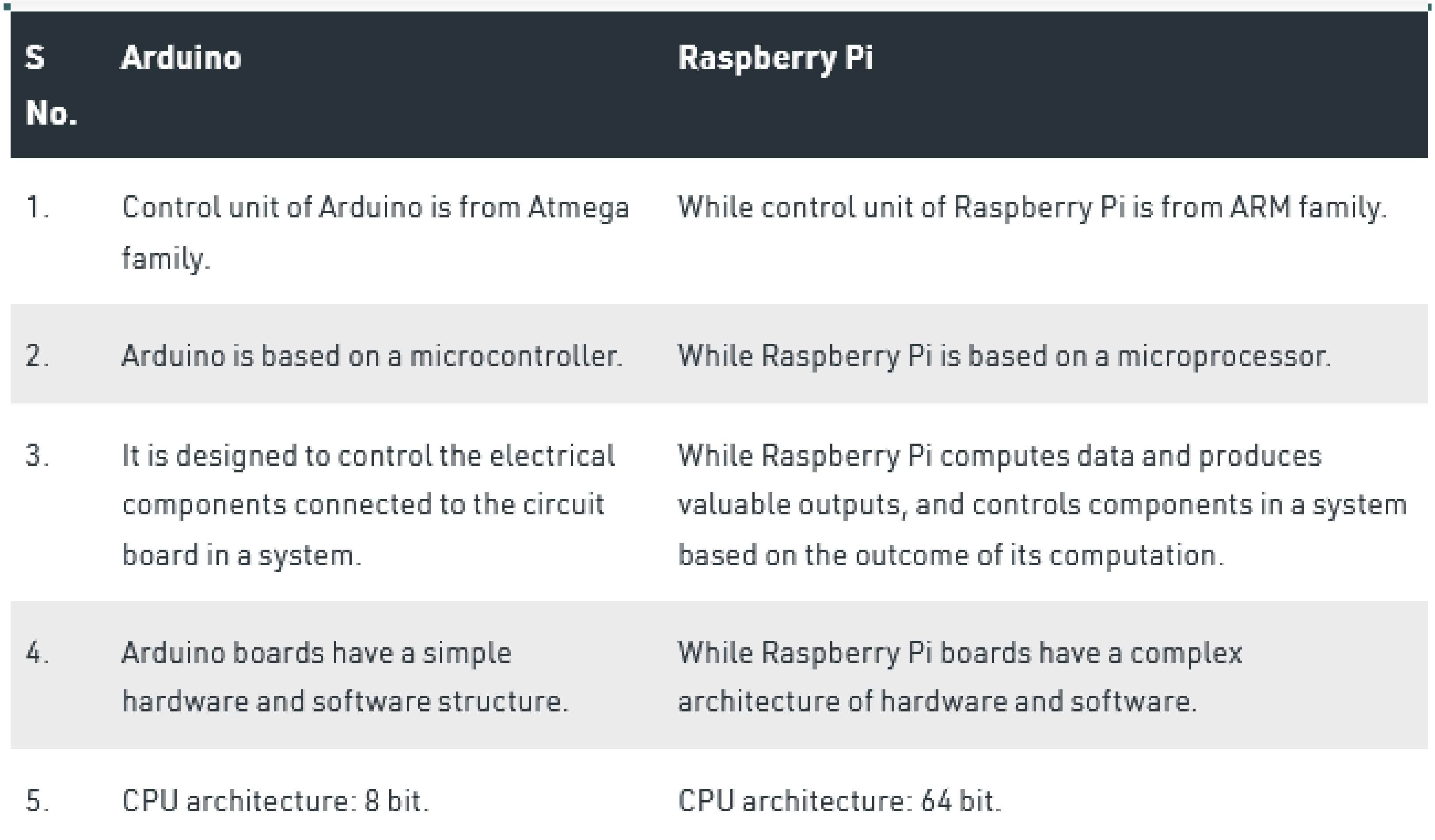
analogRead(A0); // used to read the analog value from the pin A0

analogWrite(2,128);

ADC Example

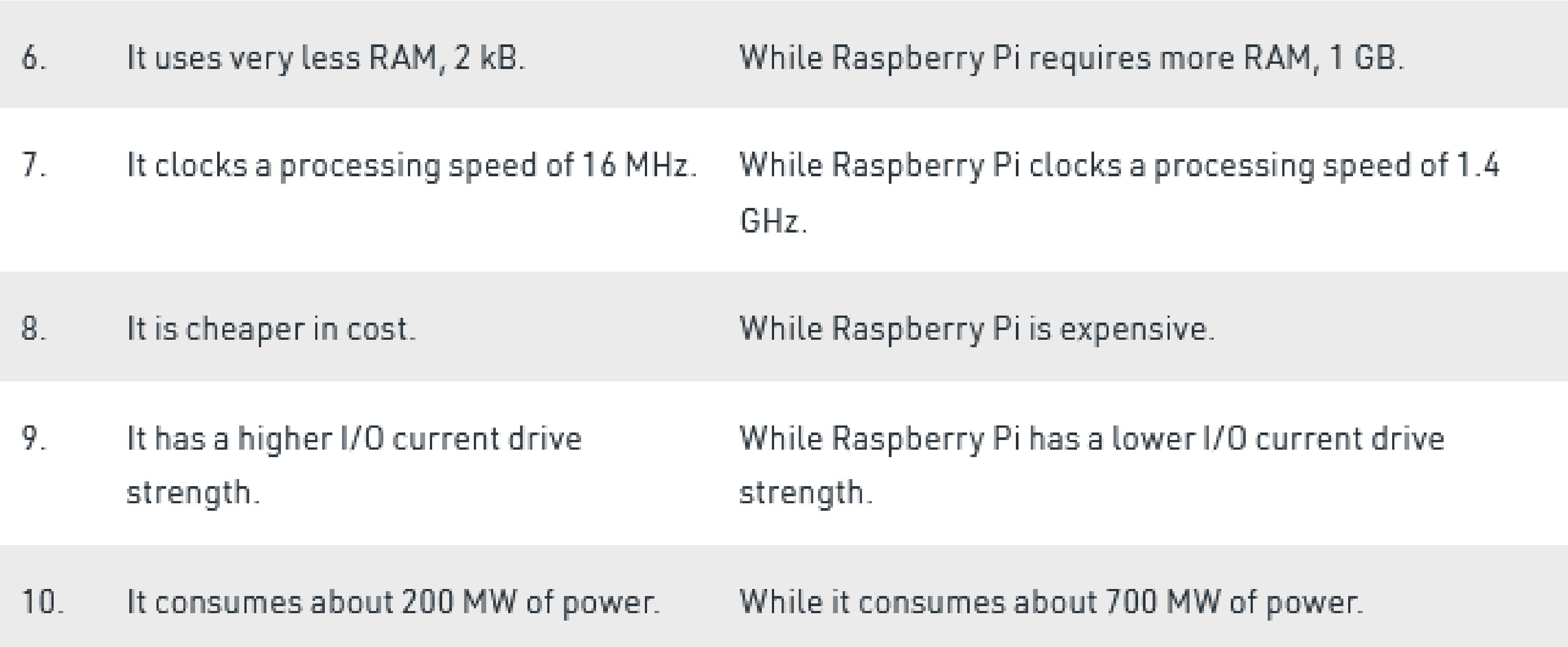
```
// These constants won't change.  They're used to give names to the pins used:
        const int analogInPin = A0;  // Analog input pin that the potentiometer is attached to
        const int analogOutPin = 9; // Analog output pin that the LED is attached to

        int sensorValue = 0;        // value read from the pot
        int outputValue = 0;        // value output to the PWM (analog out)

        void setup() {
          // initialize serial communications at 9600 bps:
          Serial.begin(9600);
        }

        void loop() {
          // read the analog in value:
          sensorValue = analogRead(analogInPin);
          // map it to the range of the analog out:
          outputValue = map(sensorValue, 0, 1023, 0, 255);
          // change the analog out value:
          analogWrite(analogOutPin, outputValue);

          // print the results to the serial monitor:
          Serial.print("sensor = " );
          Serial.print(sensorValue);
          Serial.print("\t output = ");
          Serial.println(outputValue);

          // wait 2 milliseconds before the next loop
          // for the analog-to-digital converter to settle
          // after the last reading:
          delay(2);
        }
```

Arduino Vs Raspberry Pi



S No.	Arduino	Raspberry Pi
1.	Control unit of Arduino is from Atmega family.	While control unit of Raspberry Pi is from ARM family.
2.	Arduino is based on a microcontroller.	While Raspberry Pi is based on a microprocessor.
3.	It is designed to control the electrical components connected to the circuit board in a system.	While Raspberry Pi computes data and produces valuable outputs, and controls components in a system based on the outcome of its computation.
4.	Arduino boards have a simple hardware and software structure.	While Raspberry Pi boards have a complex architecture of hardware and software.
5.	CPU architecture: 8 bit.	CPU architecture: 64 bit.

Arduino Vs Raspberry Pi



6.	It uses very less RAM, 2 kB.	While Raspberry Pi requires more RAM, 1 GB.
7.	It clocks a processing speed of 16 MHz.	While Raspberry Pi clocks a processing speed of 1.4 GHz.
8.	It is cheaper in cost.	While Raspberry Pi is expensive.
9.	It has a higher I/O current drive strength.	While Raspberry Pi has a lower I/O current drive strength.
10.	It consumes about 200 MW of power.	While it consumes about 700 MW of power.