



# Embedded Systems

# ARDUINO UYGULAMALAR

Dr. Cahit Karakuş, June-2019

# Kritik Altyapılar

Ekonominin temel taşları olan yatırımlar:

- Köprüler, havaalanları, limanlar, tren istasyonları ve hatları
- Sanayi Bölgeleri
- Finans
- Telekom
- Savunma Sanayi
- Boru hatları (Petrol, doğal gaz)
- Elektrik dağıtım şebekesi
- Elektrik üretim sahaları: Nükleer, Barajlar, Termik santraller, Rüzgar türbünleri, Güneş panelleri
- Fiziksel Güvenlik

- Terör
- Hatalar
- Bakım – onarım servis hizmetleri
- Devletler arası çatışmalar ve oyunlar
- En temel problem: Bilgisayar, internet
- Saldırı: Solucan, Hacker, Sahtekarlık
- Çevre kirliliği

# What is a Microcontroller

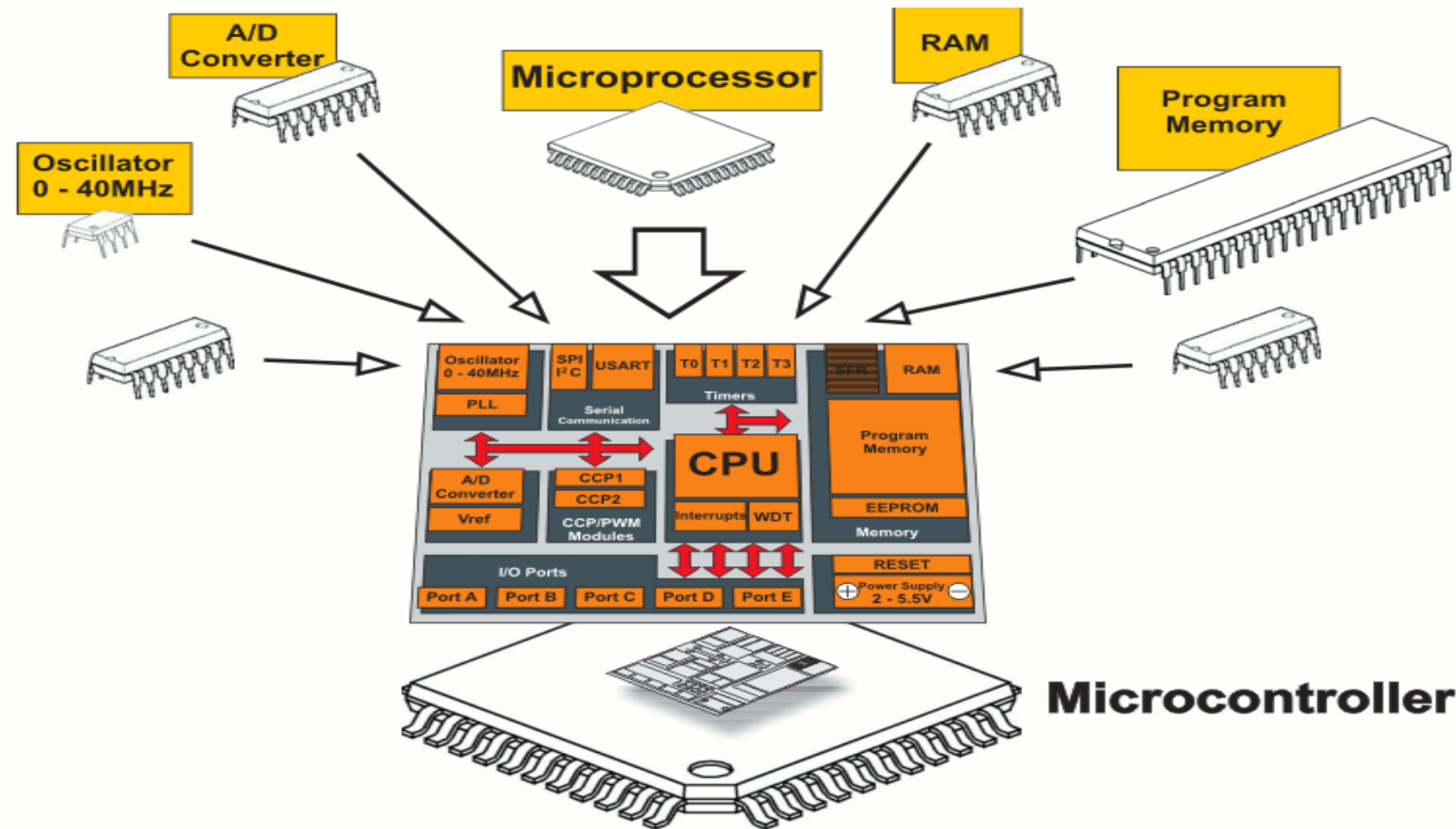


Fig. 0-1 Microcontroller versus Microprocessor

- A small computer on a single chip
  - containing a processor, memory, and input/output
- Typically "**embedded**" inside some device that they control
- A microcontroller is often small and low cost

# Gömülü Sistemler:

MP, Bellek (Ram, Rom), I/O  
Timing & Clock, Sistem Bus

- Yapılım Yöntemleri
- 1- Üzerinde (C++)
- 2- Kurulum yapılması.

Teh. Çip. (0/80)



Pazar.

Endüstri  
Kritik Altyapı.

## Projelendirme

Otomasyon alt yapı projelendirme -

Kurulum. (Enerji, Topraklama, haberleşme.)

Yönetim ekibi eğitim

Teknik destek (7/24)

Güvenlik  
(Yangın, Su baskını, Hırsızlık) → Temel Bilgileri.

Transdüser. (Algılayıcılar, detektörler, ölçerler)  
Veri toplayıcılar (US, nettedu)

→ Sinyal (Digital, Analog)

- Embedded sistem (I/O; Numbers of I/O Analog)

- Aktüatör (motor, piston, pnömatik, Elektrik-hareket. <sup>Digital</sup> Sürme sistemleri)

- Robot (Algılayıcı - Aktüatör)

- Enerji alması - Haberleşme (Kablosuz - 5G/4G - Telsiz, sesli)

Risk: Network security (CPU-Embedded)

Switch port K

Wi-Fi

# Sınav

- Ardunio Uno R3 Başlangıç Seti, 84 Parça
- Kit seti (RFID)
- Robotistan, Cimri.com

# Platforms

- Arduino
- Raspberry Pi
- Intel Galileo
- Adafruit
- SparkFun
- ARM mbed
- Particle
- Etc..

# Temel Kavramlar

# What to Get

- **Required:**

- Arduino (such as Uno)
- USB A-B (printer) cable
- Breadboard
- Hookup wire
- LED's
- Resistors
- Sensors
- Switches



- **Good Idea:**

- Capacitors
- Transistors
- DC motor/servo
- Relay

- **Advanced:**

- Soldering iron & solder
- Heat shrink tubing
- 9V battery adapter
- Bench power supply
- Multimeter
- Havya



# 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

Gömülü - 8+

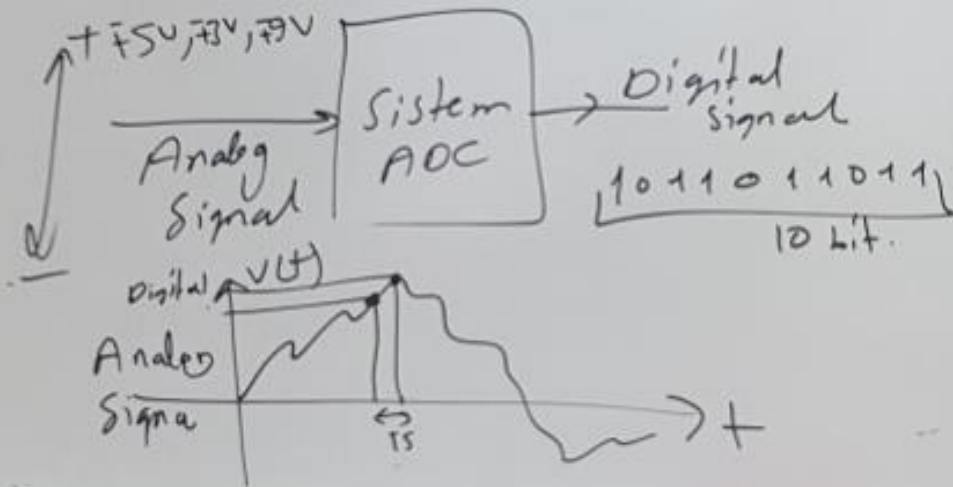
# 10 Bit ADC

Analog Sinyal.

Örnekleme frekansı,  $f_s \geq 2 * f_{max}$

↑  
Sinyaldeki  
max frekans

Örnekleme  
Aralığı,  $T_s = \frac{1}{f_s}$



$2^{10} = 1024$   
(0 ile 1023)  
aranda  
digital  
değer

$1024$  {  
00 0000 0000 = (0)<sub>d</sub>.  
:  
:  
:  
11 1111 1111 (7)<sub>d</sub>  
98 7654 3210 -indis

$$2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 1023$$

**WHAT IS ARDUINO?**

# Arduino

- Arduino, elektronik ve programlama ile deney yapmayı daha eğlenceli ve sezgisel hale getirmek için tasarlanmış açık kaynaklı bir fiziksel bilgi işlem platformudur.
- Arduino'nun kendine özgü, basitleştirilmiş bir programlama dili vardır ve çok sayıda önceden hazırlanmış sınav ve öğretici programlar mevcuttur. Rom belleğe yüklenmiştir.
- Arduino ile motorlar, sıcaklık sensörleri vb. gibi birçok küçük ölçekli sensörü ve aktüatörü kolayca keşfedebilirsiniz.
- Arduino ile çözüm olasılıkları sonsuzdur.

# What is Arduino?

- It's an open source electronics prototyping platform:
  - Open Source electronic prototyping platform based on flexible easy to use **hardware and software**.
  - **Open source**: resources that can be used, redistributed or rewritten free of charge, often software or hardware.
  - **Electronics**: technology which makes use of the controlled motion of electrons through different media.
  - **Prototyping**: an original form that can serve as a basis or standard for other things.
  - **Platform**: hardware architecture with software framework on which other software can run. Sanal medyada sosyaleşmeye yönelik paylaşım (Deneyim, Tecrübe, Bilgi birikimi)

# What is Arduino?

- Arduino kartı, fiziksel dünyadaki nesnelerin fiziksel davranışlarını algılamak ve yönetmek için programlanabilen bir mikrodenetleyici ya da mikroişlemci içerir.
- Arduino, sensörlere ve girişlere yanıt vererek LED'ler, motorlar ve ekranlar gibi çok çeşitli çıkışlarla karşılıklı etkileşime girebilir.
- Esnekliği ve düşük maliyeti nedeniyle Arduino, etkileşimli donanım projeleri oluşturmak isteyen tasarımcılar ve yaratıcı alanlar için çok popüler bir seçim haline geldi.
- Arduino, 2005 yılında İtalya'da Massimo Banzi tarafından mühendis olmayanların donanım projeleri oluşturmak için düşük maliyetli, basit bir araca erişmesinin bir yolu olarak tanıtıldı.
- Pano açık kaynak olduğundan, herkesin kendi panosunu üretmesine izin veren bir Creative Commons lisansı altında yayınlanır. Web'de arama yaparsanız, yüzlerce Arduino uyumlu klon ve varyasyon bulacaksınız, ancak yalnızca resmi kurulların adında Arduino vardır.

# Uzaktan motor alıřtırma senaryosu

- Uzaktan motor alıřtırılacak.
- ıkıř sinyali retilir (Dijital)
- ıkıř sinyali src karta gider. nk, kk sinyal ile byk sinyala kontrol edilecektir. (0 – 5 v aralıęı ile 220 volt kontrol edilecektir.) Gml sistem ıkıř sinyal ile sistem alıřtırılmaz. nk, akım ve gerilim seviyesi yeterli olamaz.
- Motor alıřtırılır. Peki, motorun alıřtıęını nasıl anlayacaksınız? Motorun enerji ekip ekmedięi, ısınıp ısınmadıęı, hareket retilip retilmedięi algılayıcılarla belirlenmelidir.

# Why Arduino?

- Her ne sebeple olursa olsun, Arduino mikrodenetleyicileri fiili standart haline geldi.
- Arduino mikrodenetleyicilerini kullanan birçok projeye sahiptir.
- Kullanım kolaylığı ve kullanışlılık arasındaki dengeyi sağlamaya çalışır.
- Programlama dilleri en büyük engel olarak görülüyor.Arduino C, C++'ın oldukça basitleştirilmiş bir versiyonudur.
- Ucuz (35 \$ perakende).



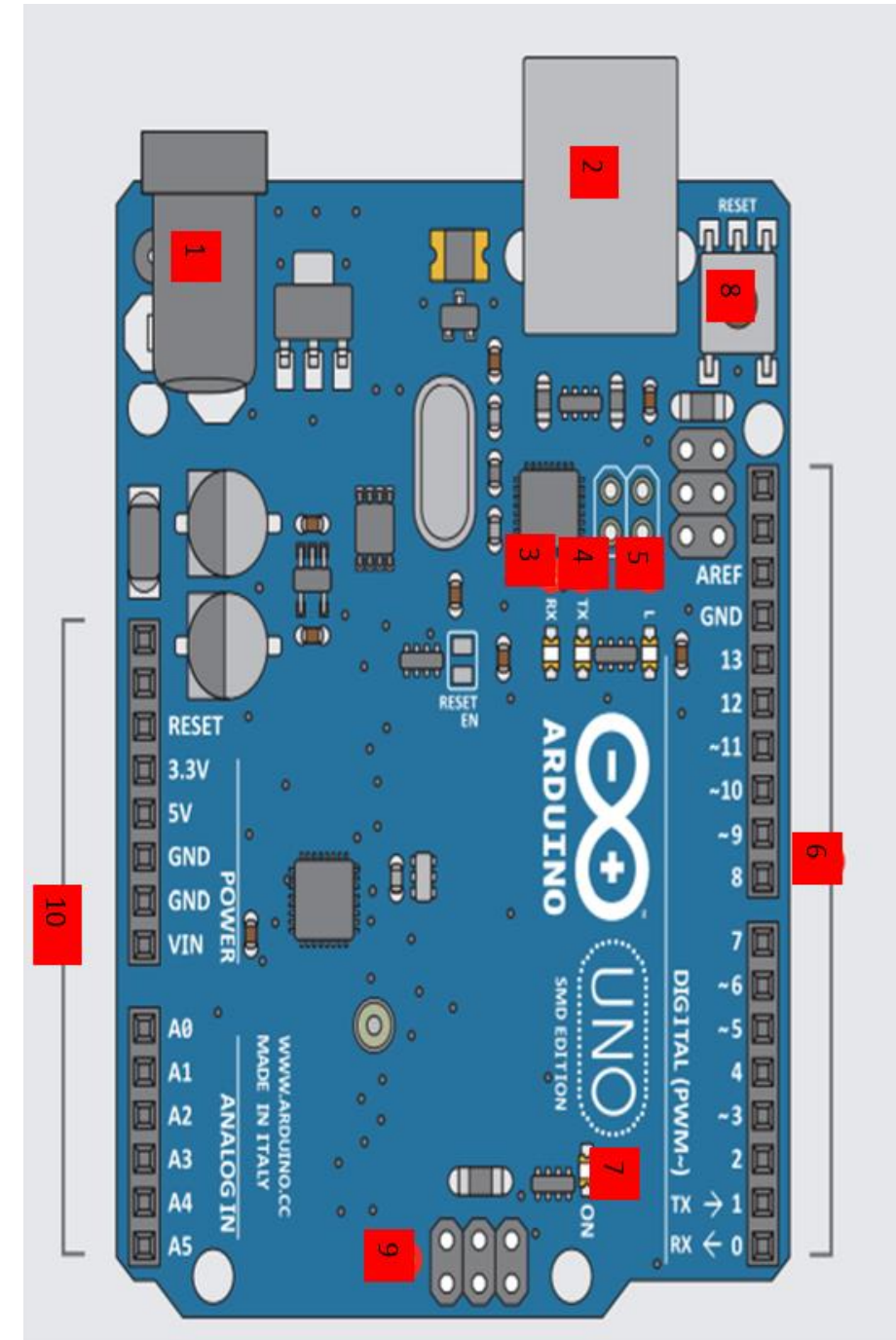
# ARDUINO

Arduino'nun özel kartları sayesinde;

- Motor, LCD ekran, röle, sensör gibi birçok ünite Kontroller, GSM ve internet üzerinden kolaylıkla kontrol edebilmektedir.
- Atmel AVR 8 bit mikrodenetleyici ve C++ tabanlı IDE tabanlı açık kaynaklı bir donanım platformu 300.000'den fazla pano üretildi
- Arduino Due, 32-bit ARM Cortex'e dayanmaktadır
- Tüm arduinolar aynı şekilde programlanır ve kullanılır.
- Sadece ihtiyacınıza göre farklı çeşitleri seçmeniz gerekiyor.

# Arduino Uno

- 1) Power In (Barrel Jack): Can be used with either a 9V or 12V wall-wart or battery
- 2) Power In (USB Port): Provides power and communicates with your board when plugged into your computer via USB.
- 3) LED (RX: Receiving): Lights up when the Arduino is receiving data (such as when being programmed).
- 4) LED (TX: Transmitting): Lights up when your Arduino is transmitting data (such as when running a program).
- 5) LED (Pin 13: Troubleshooting): LED is incorporated into your sketch to show if your program is running properly.
- 6) Pins (ARef, Ground, Digital, Rx, Tx): Used for inputs, outputs, power, and ground.
- 7) LED (Indicates Arduino is ON): Simple power indicator LED.
- 8) Reset button: Used to manually reset Arduino, which makes code restart.
- 9) ICSP Pins: For “in-circuit serial programming,” used if you want to bypass the bootloader.
- 10) Pins (Analog In, Power In, Ground, Power Out, Reset): Used for inputs, outputs, power, and ground.



# Advanced Projects

Whatever you like to control...

- Ultrasonic distance sensor
- Temperature sensor
- Liquid Crystal Displays (LCD)
- Servo motors
- Stepper motors
- Real-time clock (RTC)
- Bluetooth

# Hardware



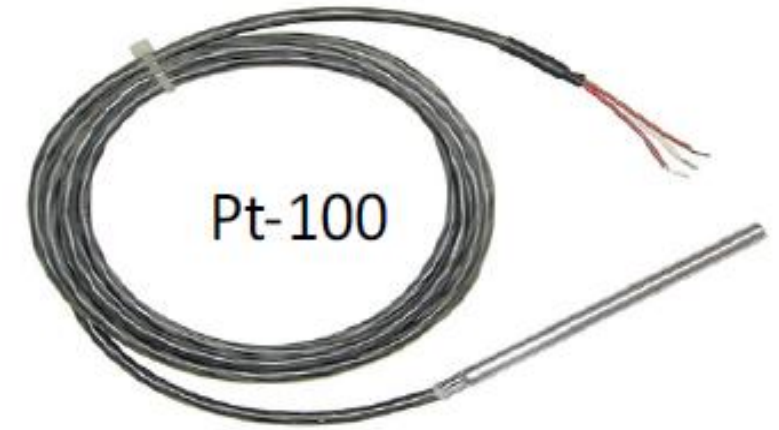
Arduino UNO Device



Breadboard

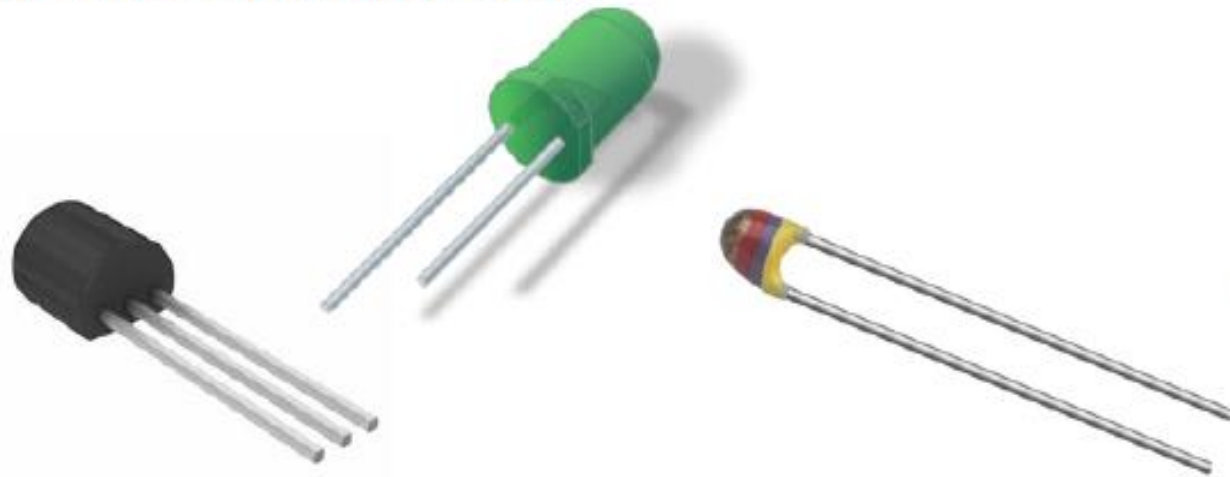


Tools



Pt-100

Sensors and Actuators



Pt-100 Transmitter



Multimeter

# **Main types of Arduino**

# ARDUINO

- Main types of Arduino:
  - *Uno*
  - *Nano*
  - *Micro*
  - *Mega*

# Different Varieties of Arduino



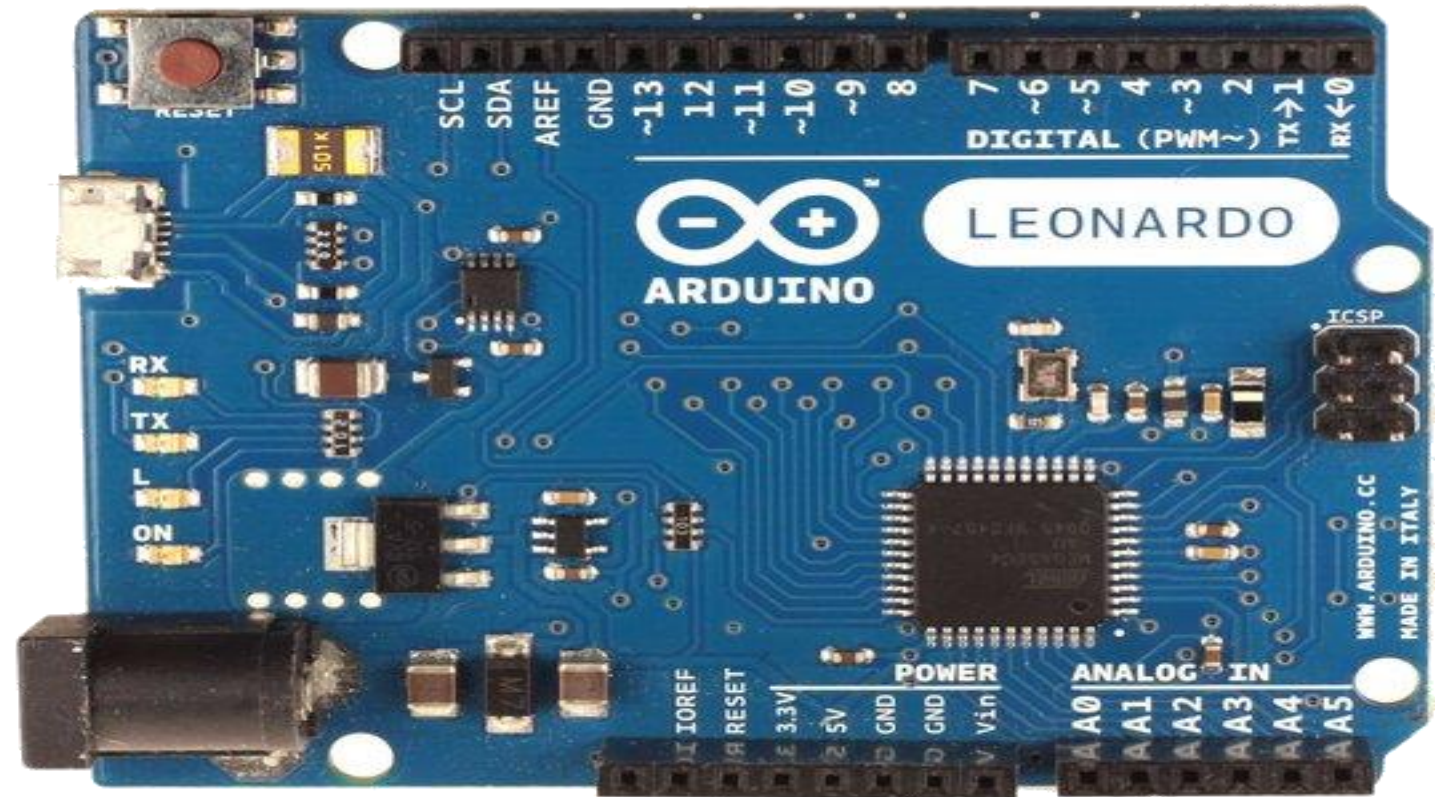
# Different Varieties of Arduino





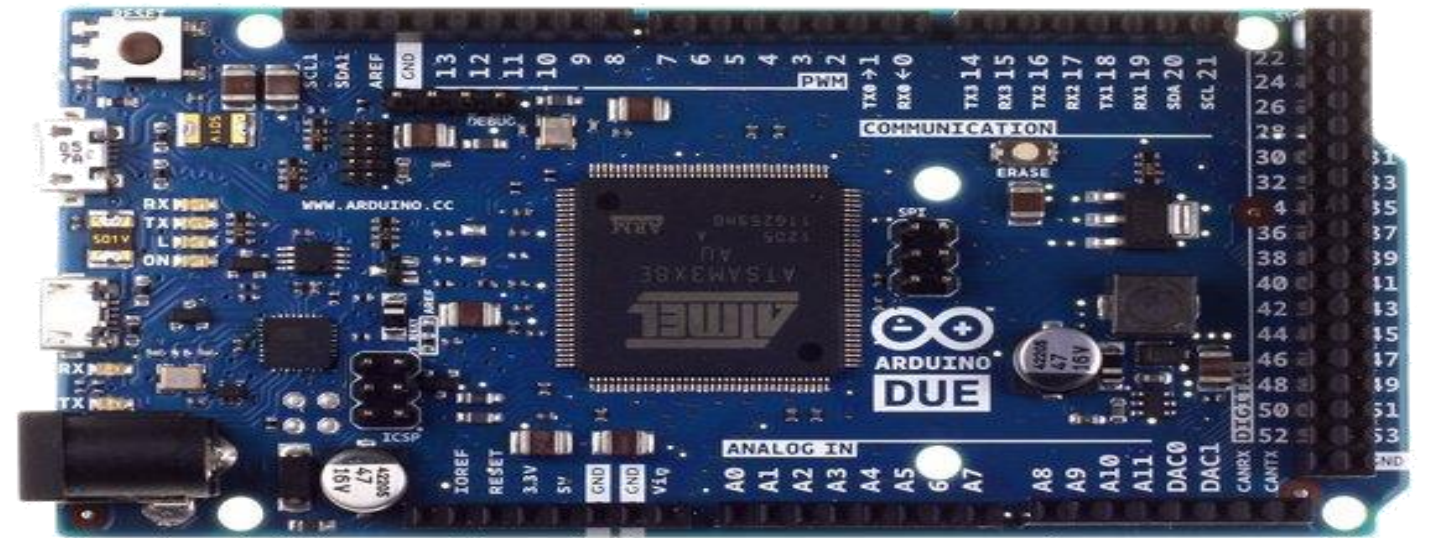
# Leonardo

- Compared to the Uno, a slight upgrade.
  - Built in USB compatibility
  - Bugs?
- Presents to PC as a mouse or keyboard



# Due

- Much faster processor, many more pins
- Operates on 3.3 volts
- Similar to the Mega



# ARDUINO MICRO

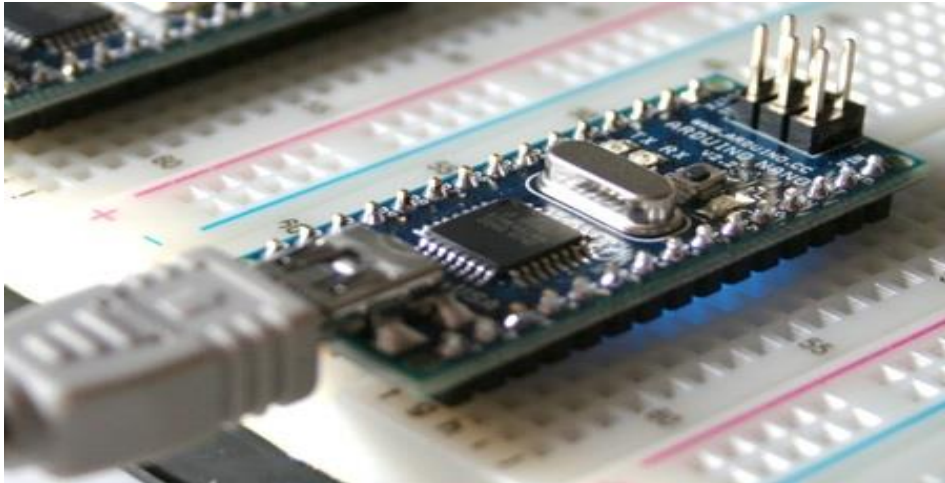


Can be mounted on breadboard.  
It is smaller in size.  
It also has a regulating circuit.  
Total 32 Input / Output Pins  
12 Analog Pins  
7 PWM pins



# ARDUINO NANO

Breadboard üzerine monte edilebilir.  
Boyut olarak **microdan daha küçük.**  
Üzerinde yine regüle devresi bulunur.  
Toplam 22 Giriş/Çıkış Pini  
8 Analog Pini  
6 PWM pini



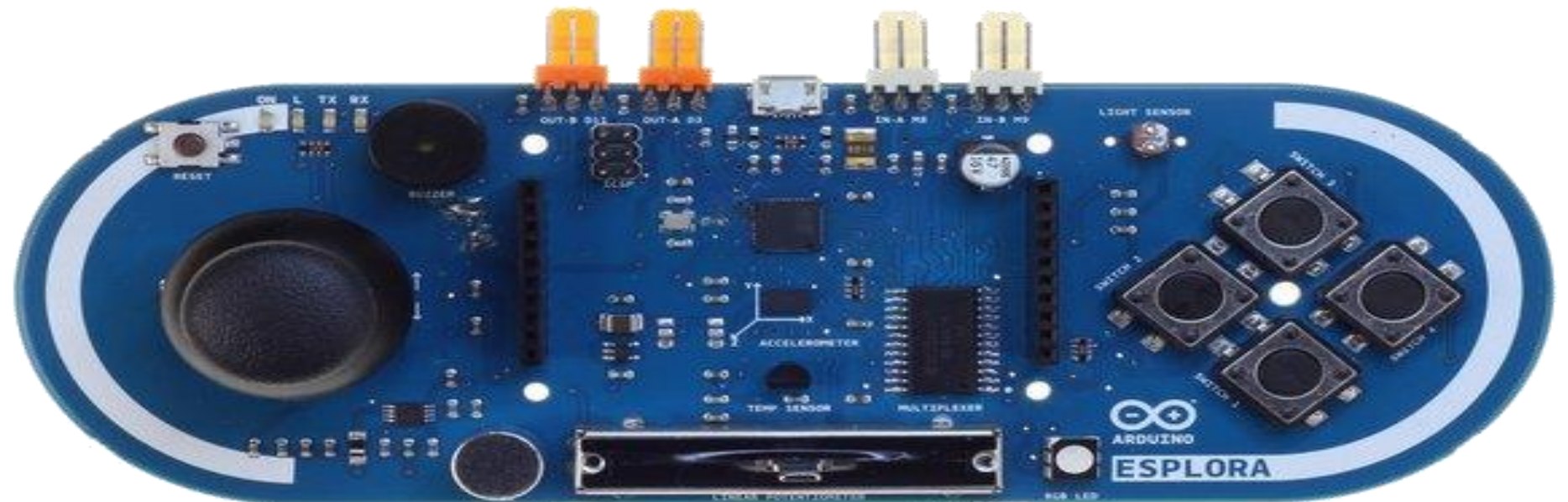
# LilyPad

- LilyPad is popular for clothing-based projects.

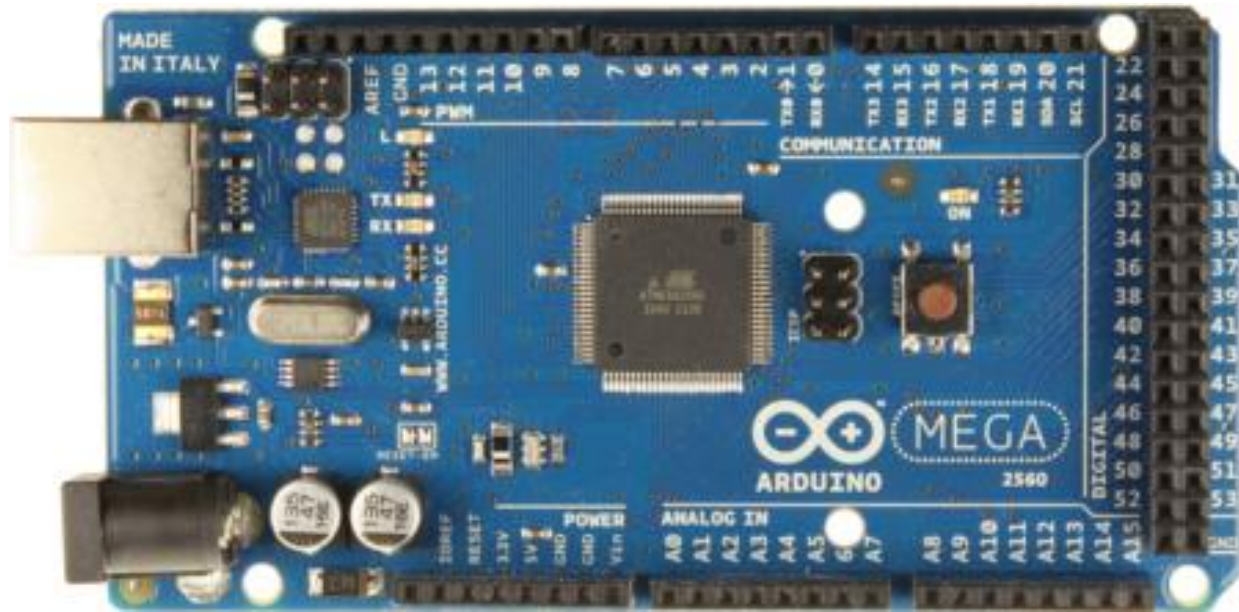


# Esplora

- Game controller
- Includes joystick, four buttons, linear potentiometer (slider), microphone, light sensor, temperature sensor, three-axis accelerometer.
- Not the standard set of IO pins.



# ARDUINO MEGA

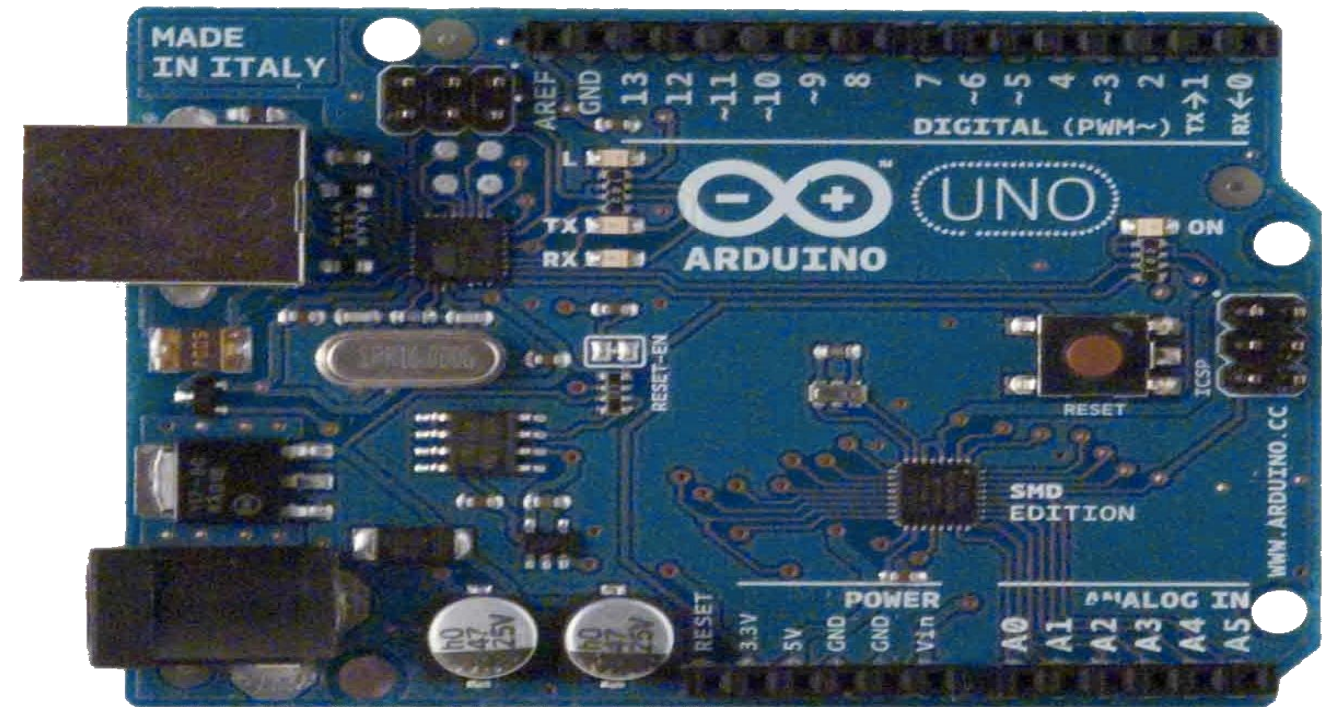


There are more pins than UNO.  
It is more preferred in places that require input /  
output connection.

Total 54 Inputs / Outputs  
15 PWM Pins  
16 Analog Pins

# Arduino Uno Close Up

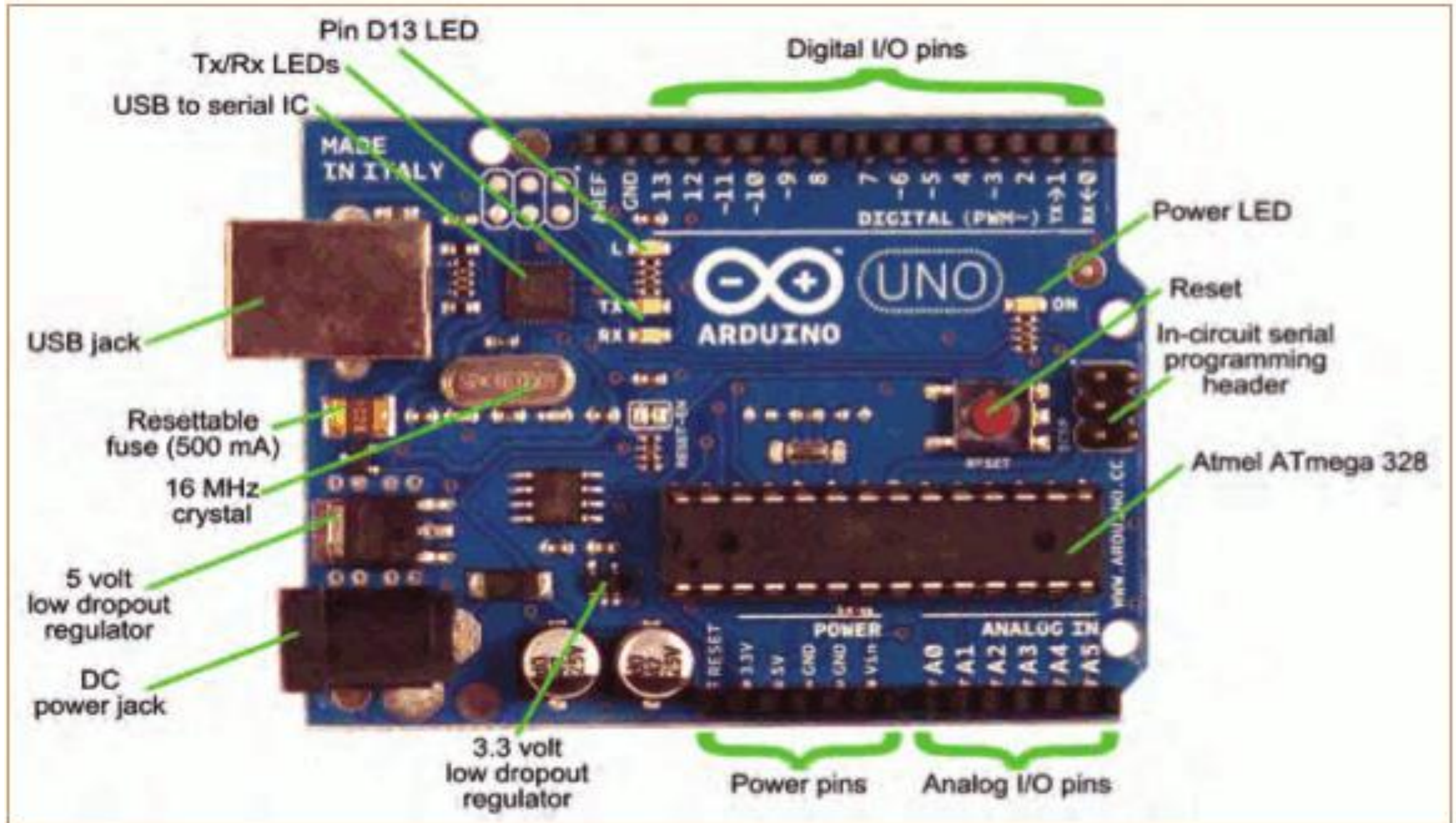
- The pins are in three groups:
  - Invented in 2010
  - 14 digital pins
  - 6 analog pins
  - power



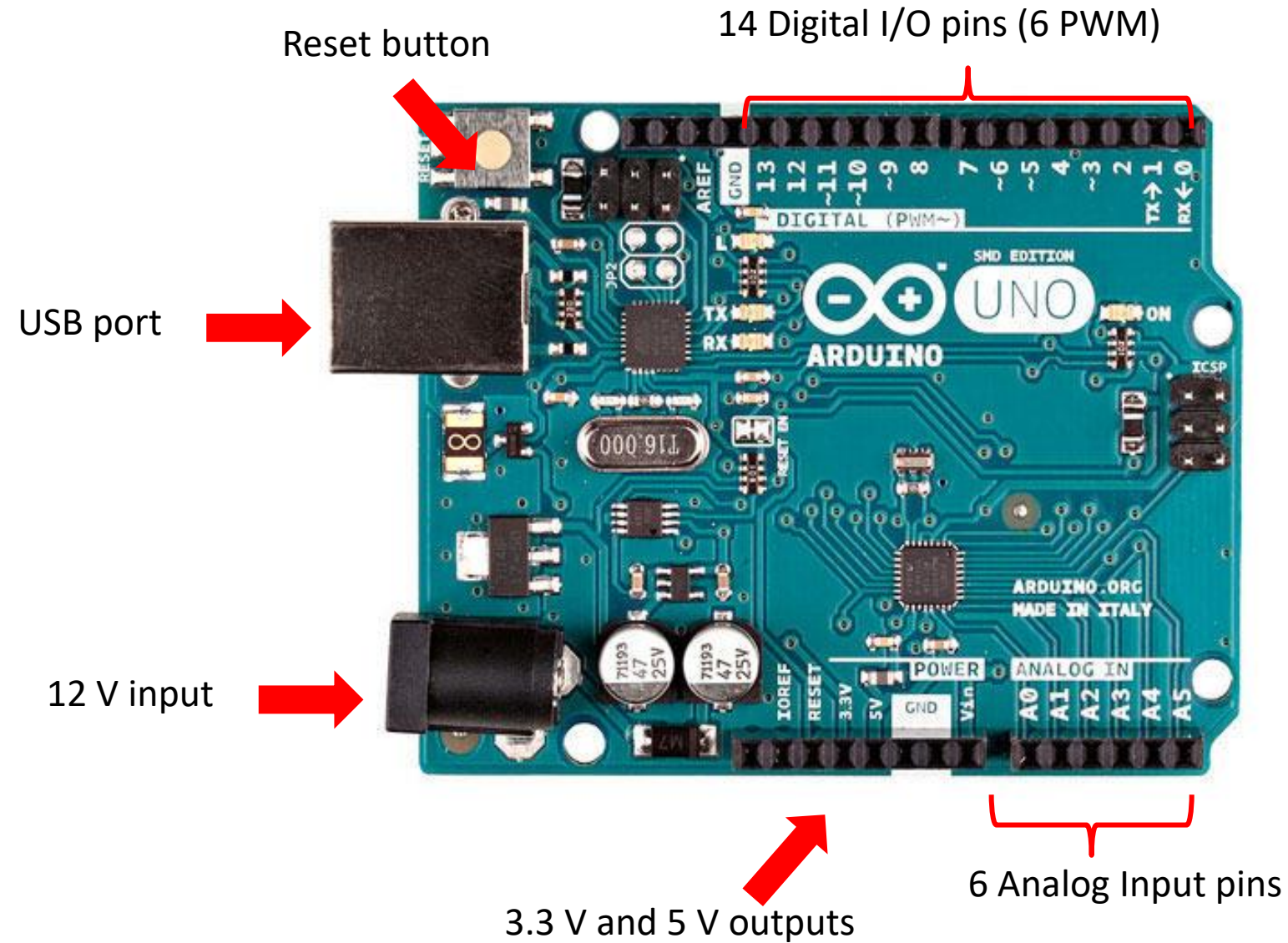


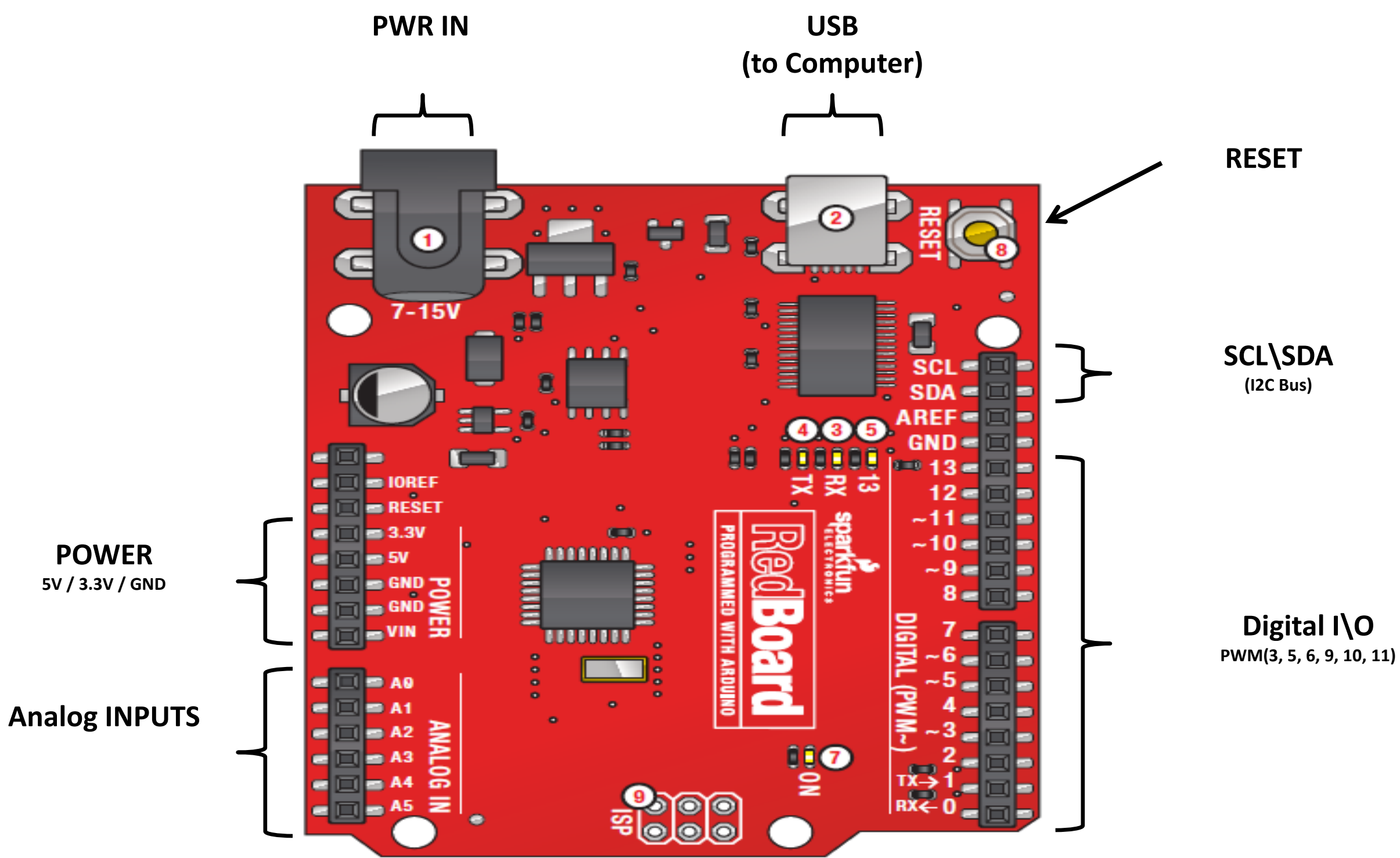
# Circuit Development Board

# The Arduino Development Board



# About Arduino Uno

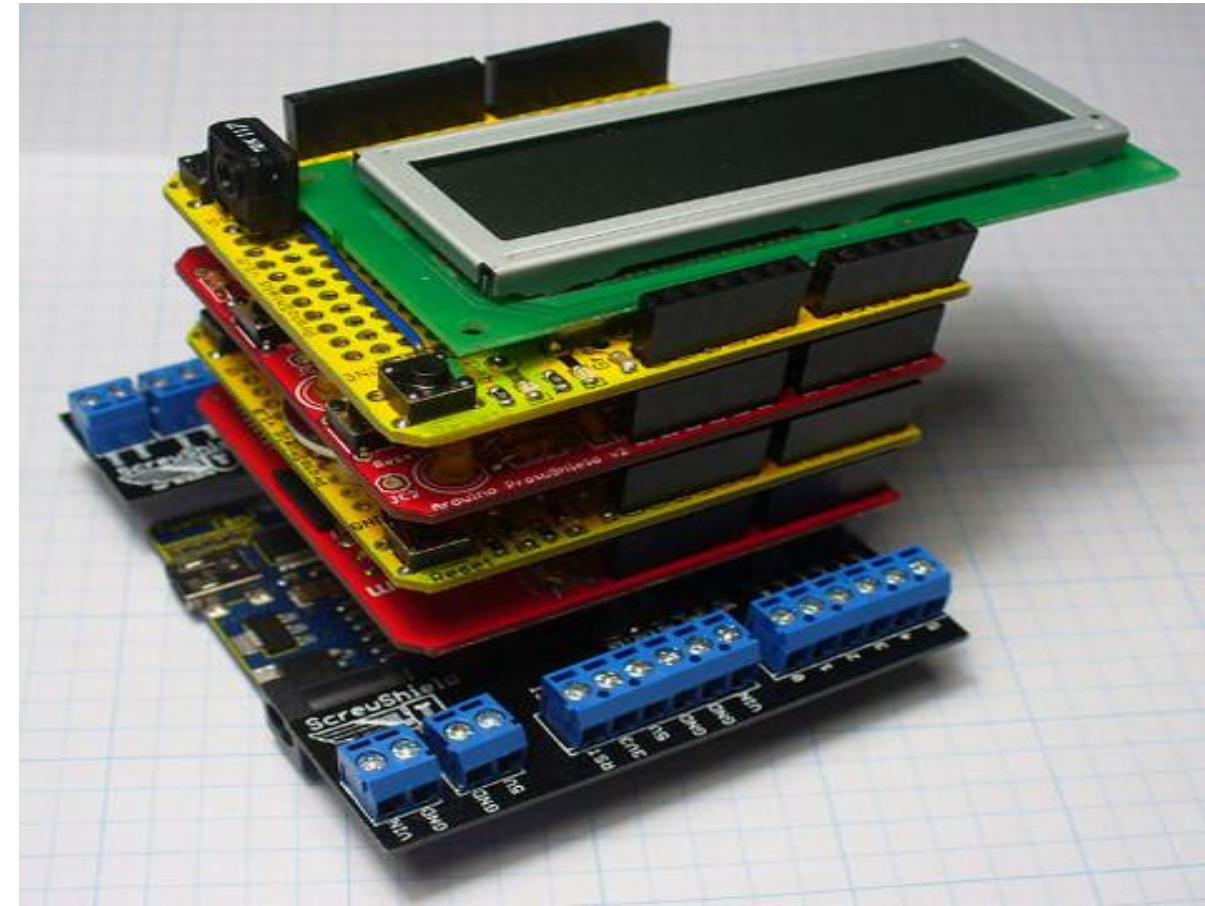
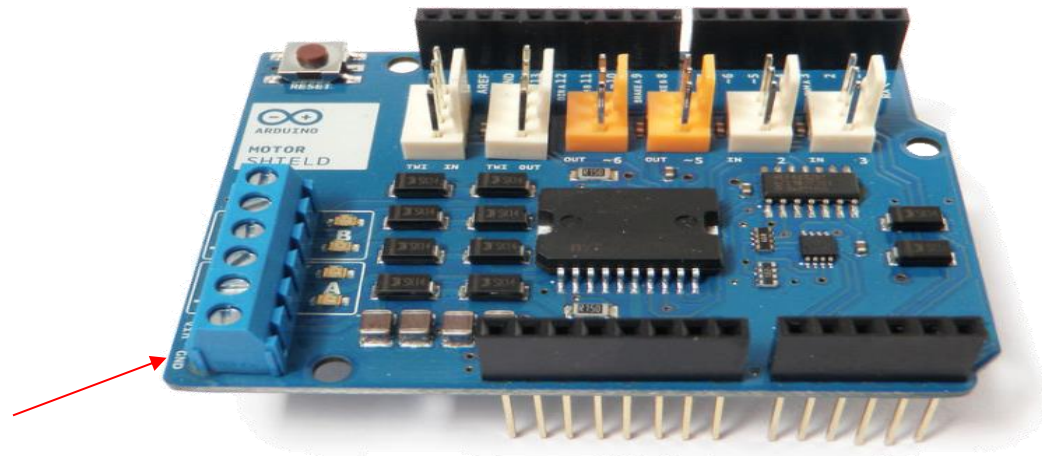




**“Extend the capabilities of an Arduino”**

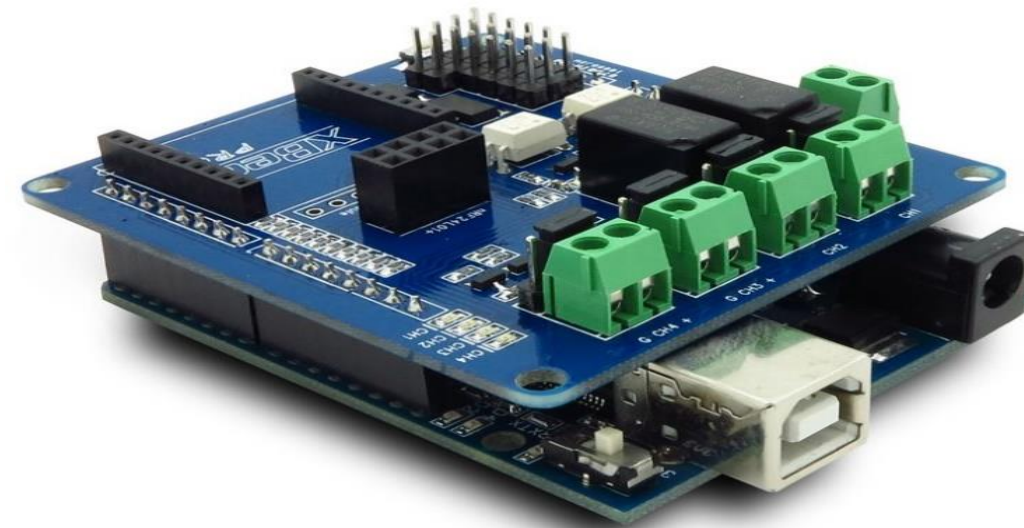
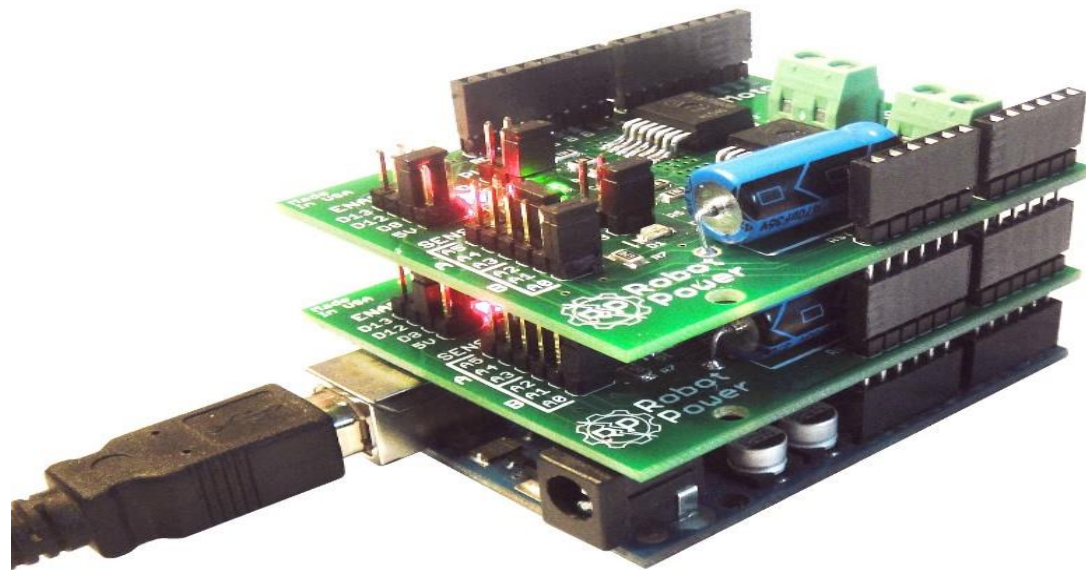
# Geliştirme Modüller

- Shields are circuit boards that plug into the top of an Arduino.
- They extend the capabilities of an Arduino.
- Examples:
  - Ethernet
  - GPS
  - Motor
  - Prototype
- [shieldlist.org](http://shieldlist.org)



# ARDUINO

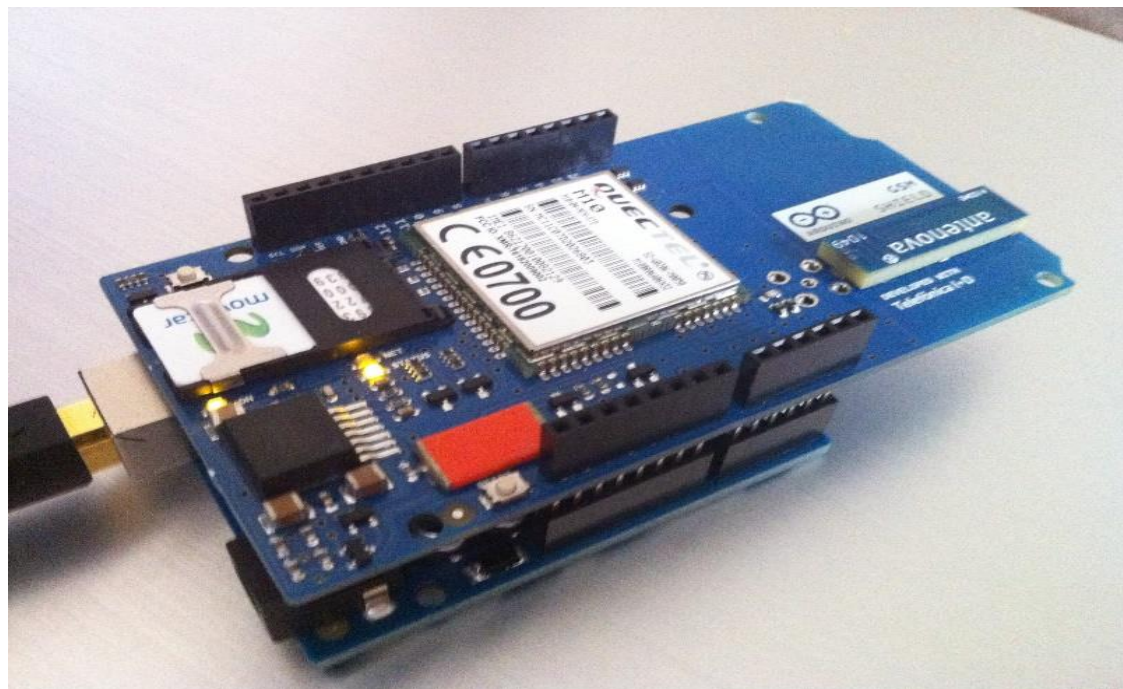
- The Arduino has a special card for each function. These cards are called shield.



DC Motor is controlled by installing Motor Shield to Arduino.

# ARDUINO

- Arduino is made special purpose by installing special shields according to the function to be used. You can create and install shields yourself.





# ARDUINO

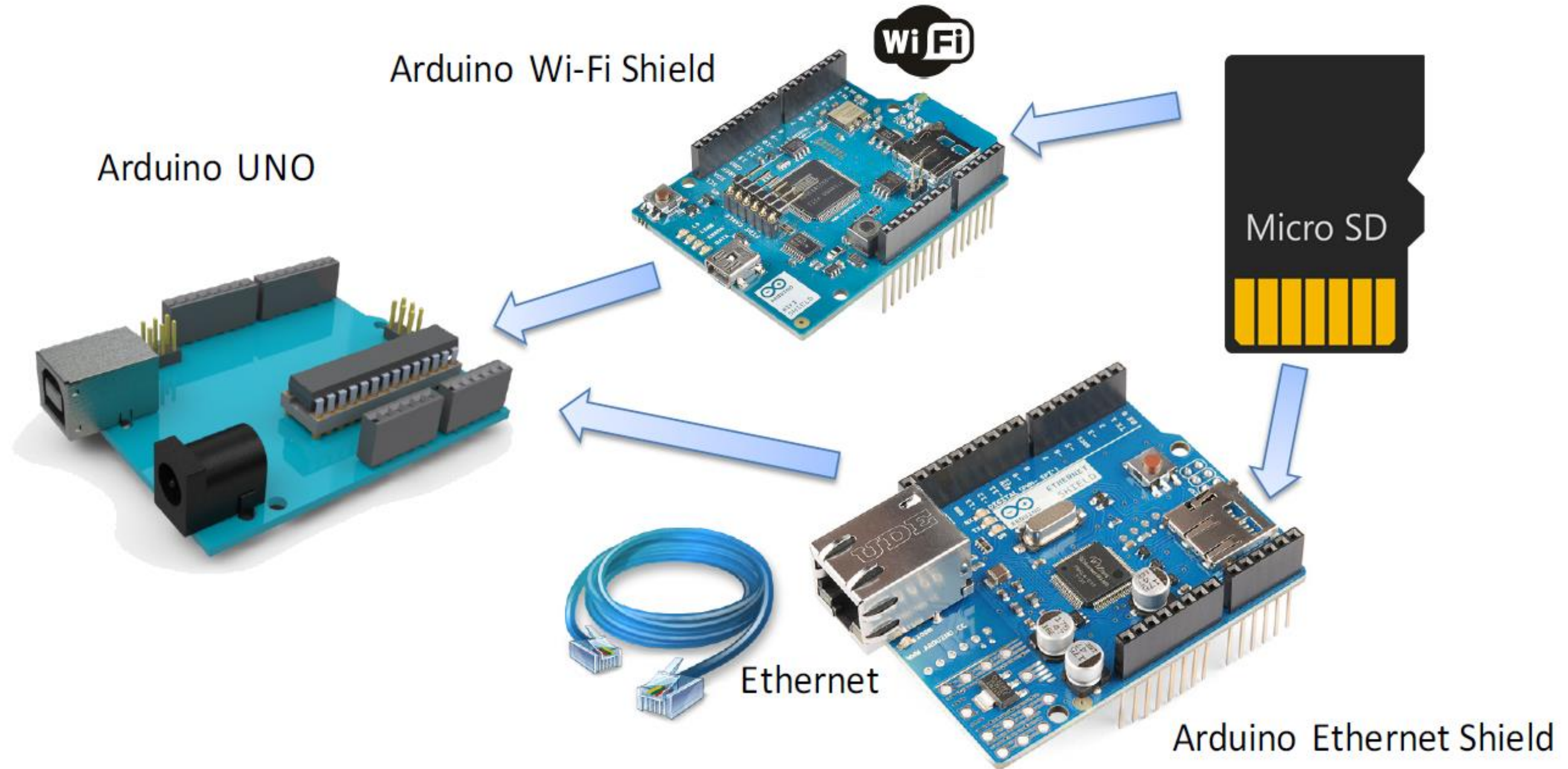


Thanks to GSM shield, an Arduino can be controlled by input and output pins via SMS



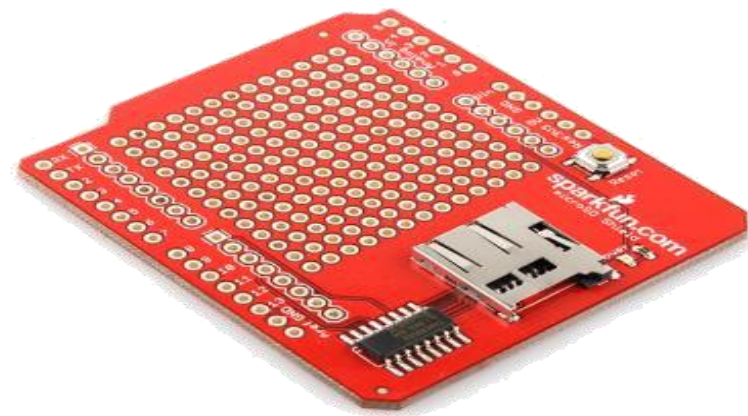
An arduino can be controled via the internet thanks to its Ethernet and LCD shield. Operations are also shown on the LCD.

# Arduino Ethernet/Wi-Fi Shield with SD Card

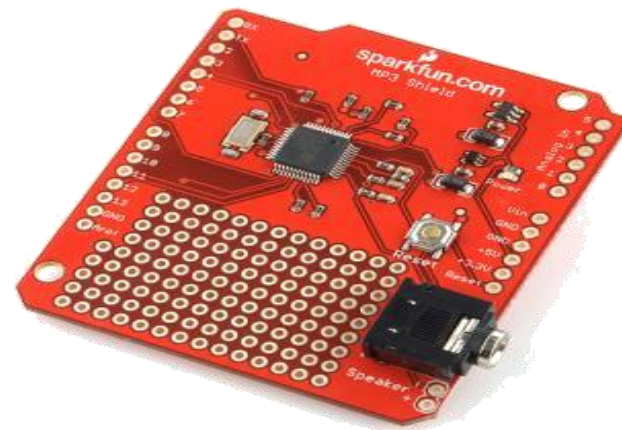


# Arduino Shields

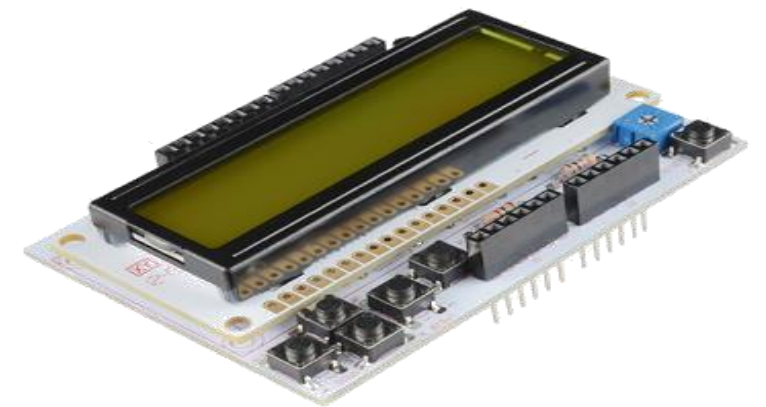
Micro SD










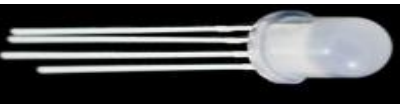
MP3 Trigger




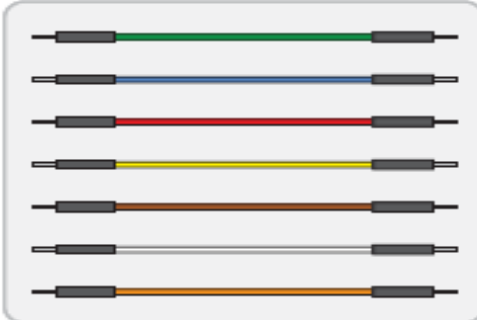
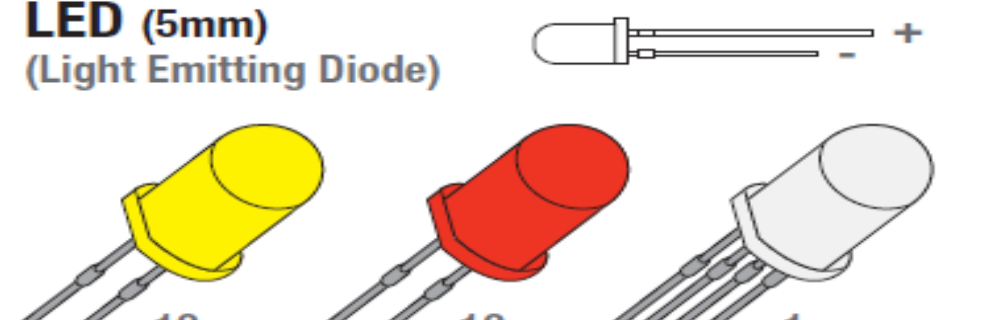




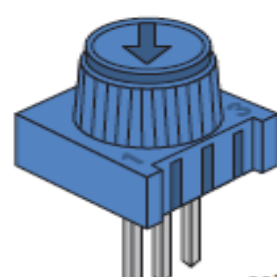

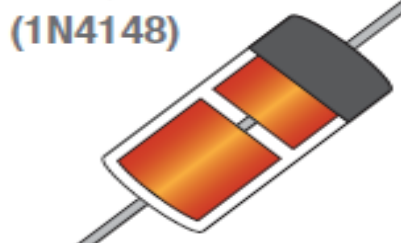
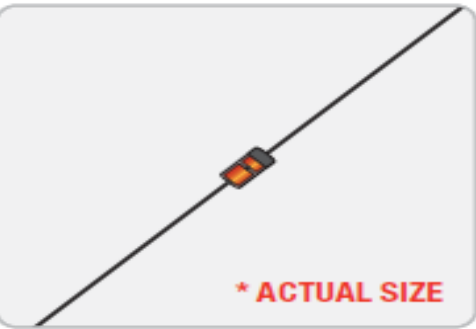
LCD



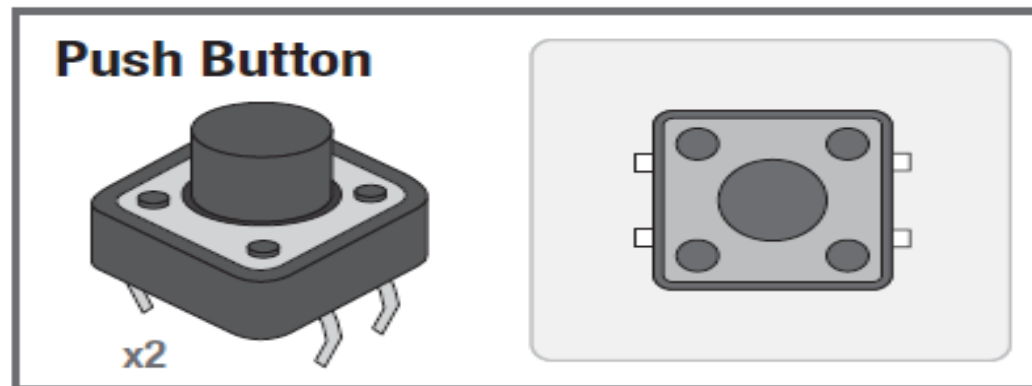
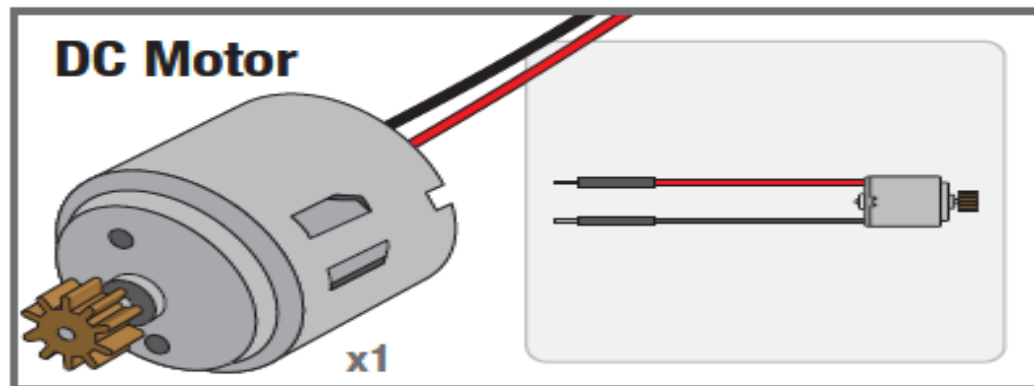
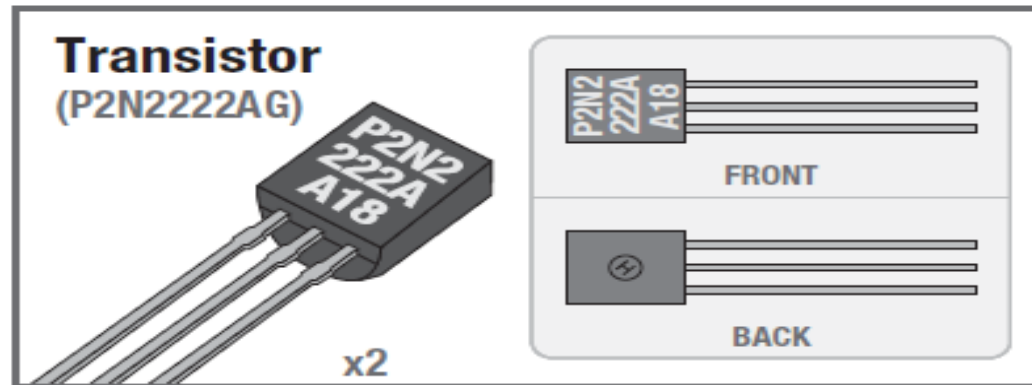
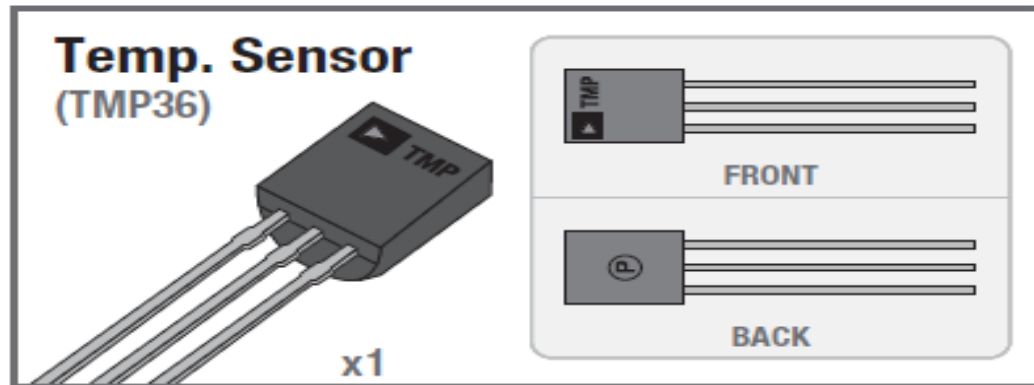
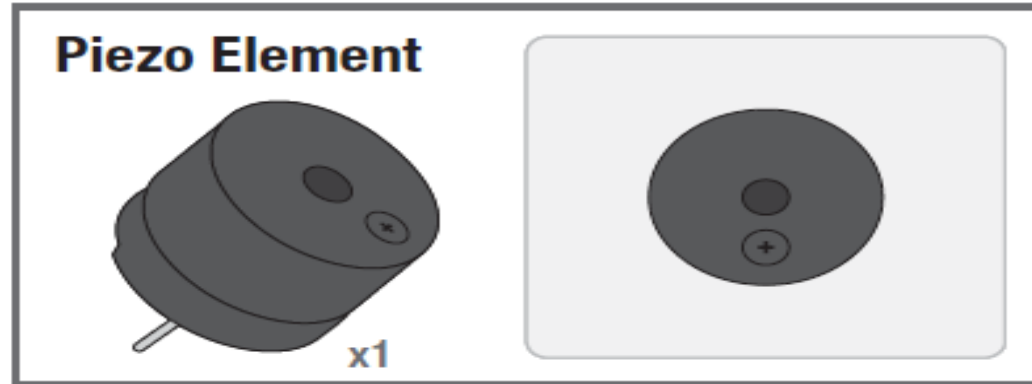
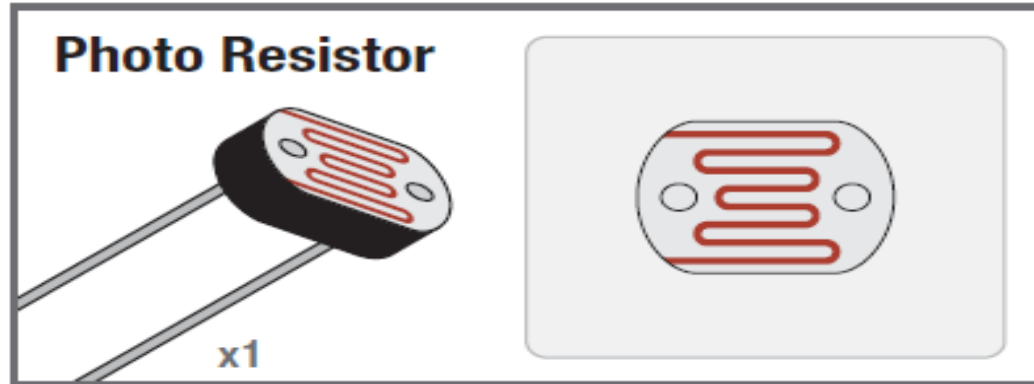
# SIK Components

Name	Image	Type	Function	Notes
Push Button		Digital Input	Switch - Closes or opens circuit	Polarized, needs resistor
Trim potentiometer		Analog Input	Variable resistor	Also called a Trimpot.
Photoresistor		Analog Input	Light Dependent Resistor (LDR)	Resistance varies with light.
Relay		Digital Output	Switch driven by a small signal	Used to control larger voltages
Temp Sensor		Analog Input	Temp Dependent Resistor	
Flex Sensor		Analog Input	Variable resistor	
Soft Trimpot		Analog Input	Variable resistor	Careful of shorts
RGB LED		Dig & Analog Output	16,777,216 different colors	Ooh... So pretty.

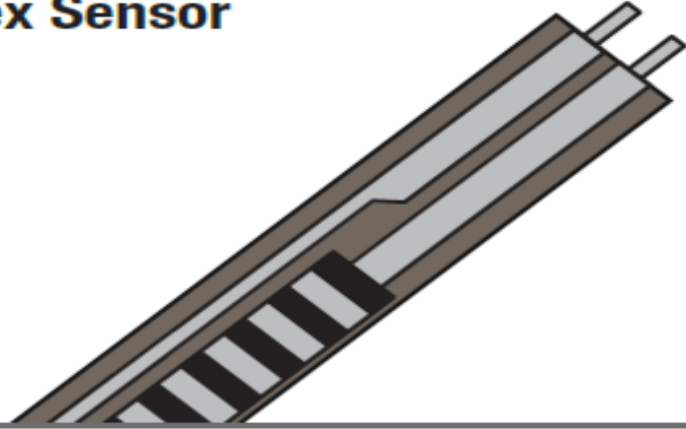
# SIK Components

<p><b>Jumper Wire</b> Various Colors</p>  <p>x30</p> 	<p><b>LED (5mm)</b> (Light Emitting Diode)</p>  <p>x10      x10      x1</p>
<p><b>330Ω Resistor</b></p>  <p>x25</p> 	<p><b>10KΩ Resistor</b></p>  <p>x25</p> 
<p><b>Potentiometer</b></p>  <p>x1</p> 	<p><b>Diode</b> (1N4148)</p>  <p>x2</p> 

# SIK Components

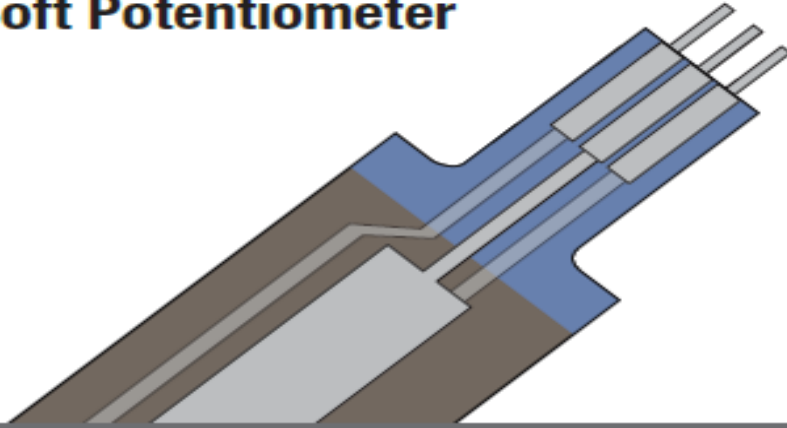


**Flex Sensor**



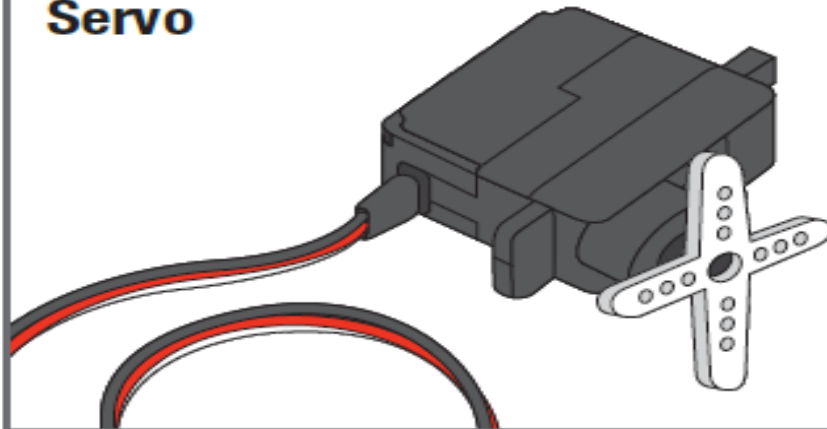
x1

**Soft Potentiometer**



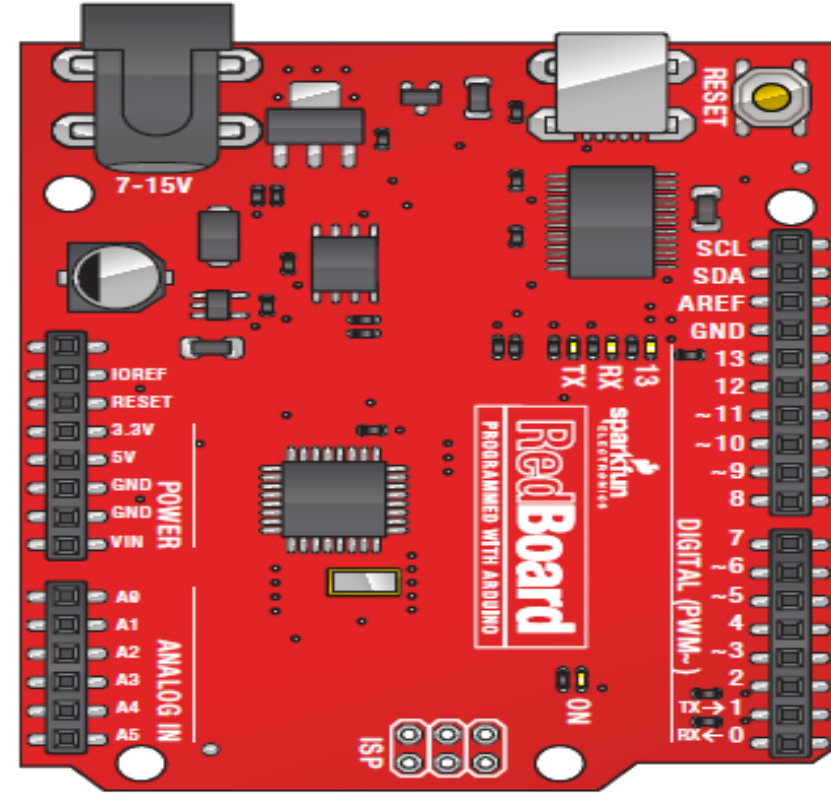
x1

**Servo**



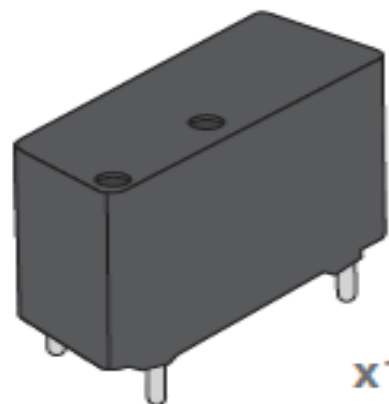
x1

**SparkFun RedBoard**



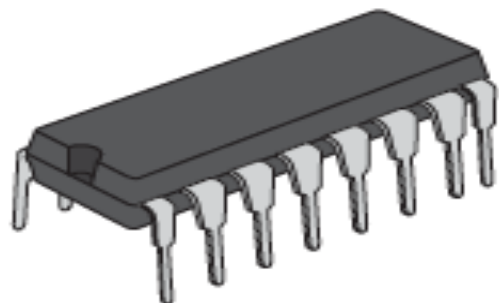
x1

## Relay



x1

## Integrated Circuit (IC)



x1

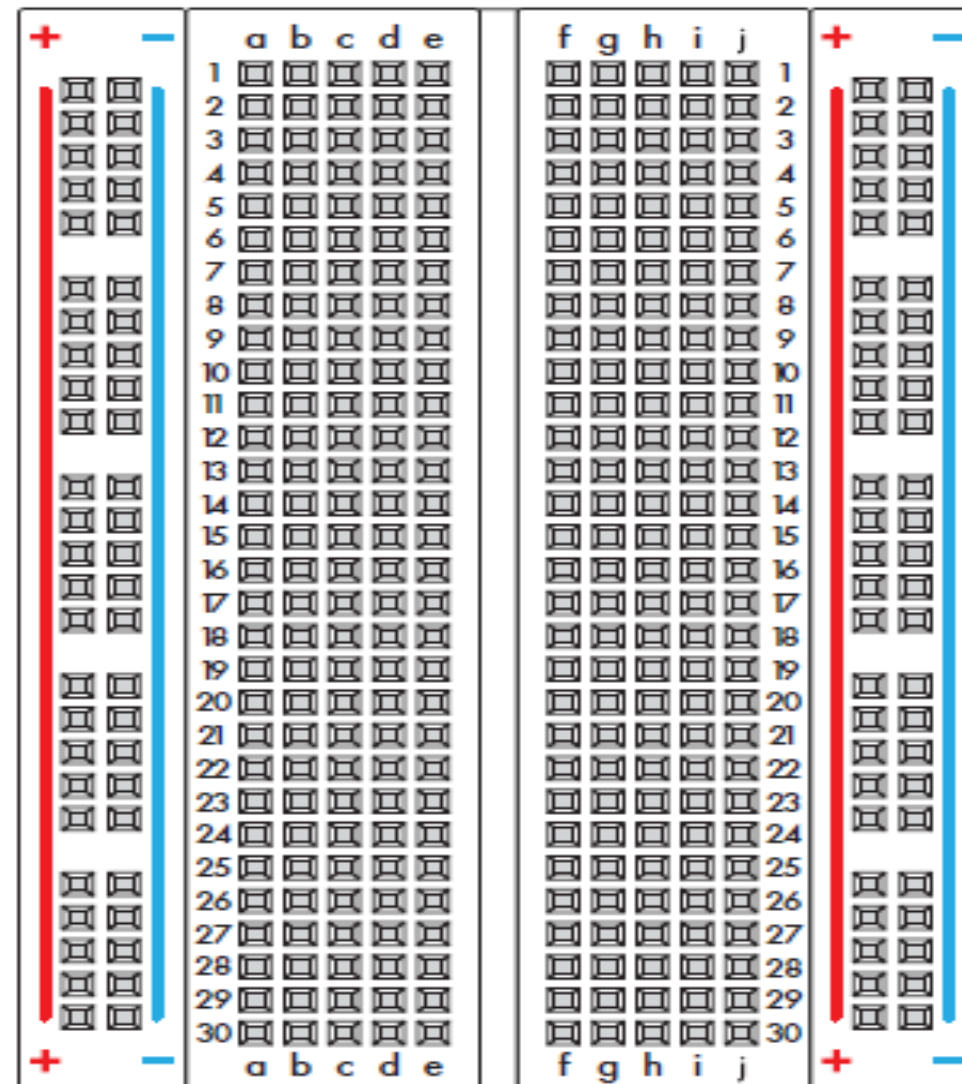
## LCD



x1

## Breadboard

Standard Solderless (Color may vary)



x1



# Arduino Software

# downLoad Arduino IDE

(Integrated Development Environment)

arduino.cc/en/main/software

# 1

## Download

Click on the “+” sign next to your appropriate computer operating system.

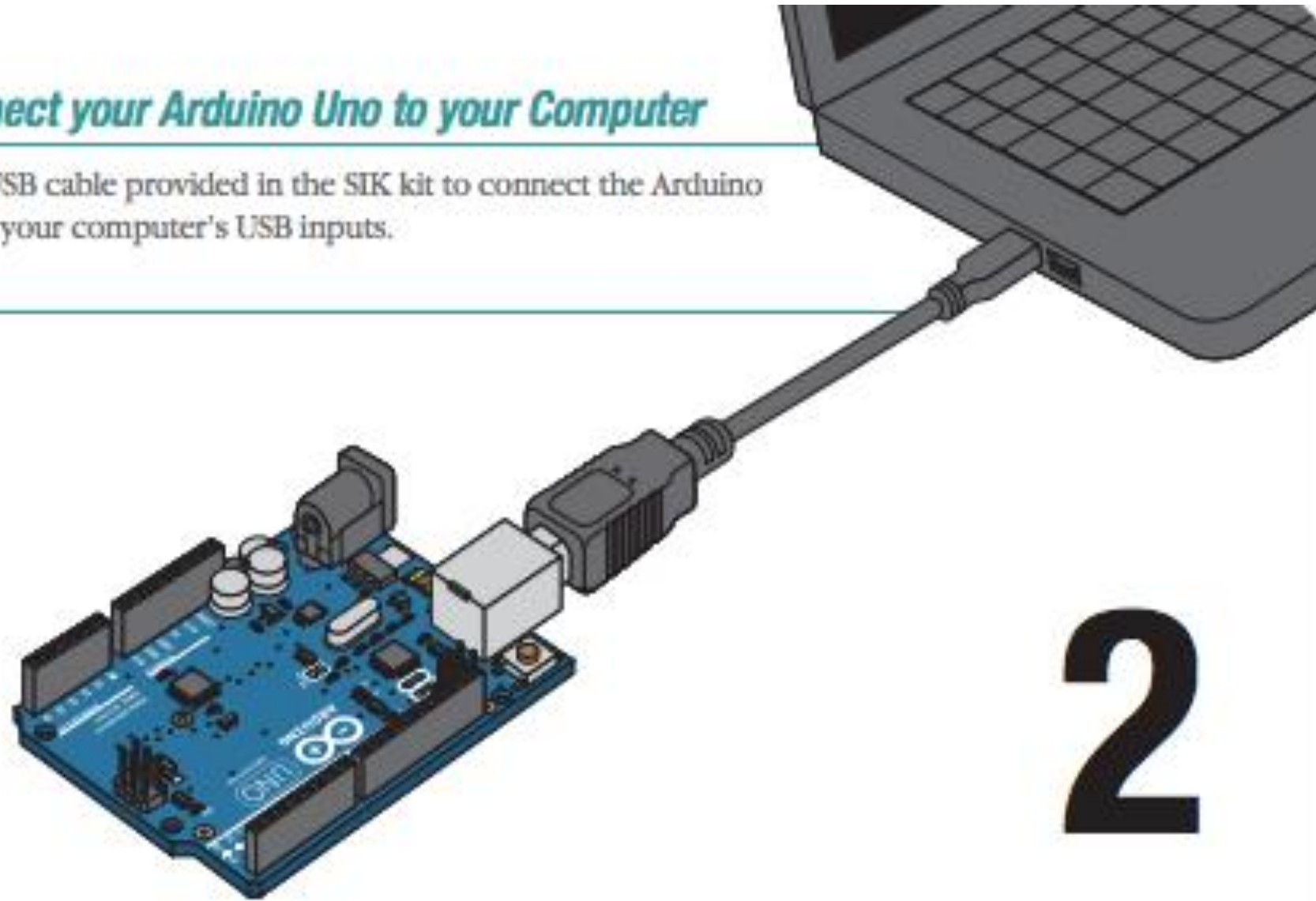
- + Windows
- + Mac OS X
- + Linux: 32 bit, 64 bit
- + source



# Connect RedBoard to your Computer

## *// Connect your Arduino Uno to your Computer*

Use the USB cable provided in the SIK kit to connect the Arduino to one of your computer's USB inputs.



2

# Getting Started

- Check out: <http://arduino.cc/en/Guide/HomePage>
  1. **Download & install the Arduino environment (IDE)**
  2. **Connect the board to your computer via the UBS cable**
  3. **If needed, install the drivers (not needed in lab)**
  4. **Launch the Arduino IDE**
  5. **Select your board**
  6. **Select your serial port**
  7. **Open the blink example**
  8. **Upload the program**

# Open up Arduino

- Hints:

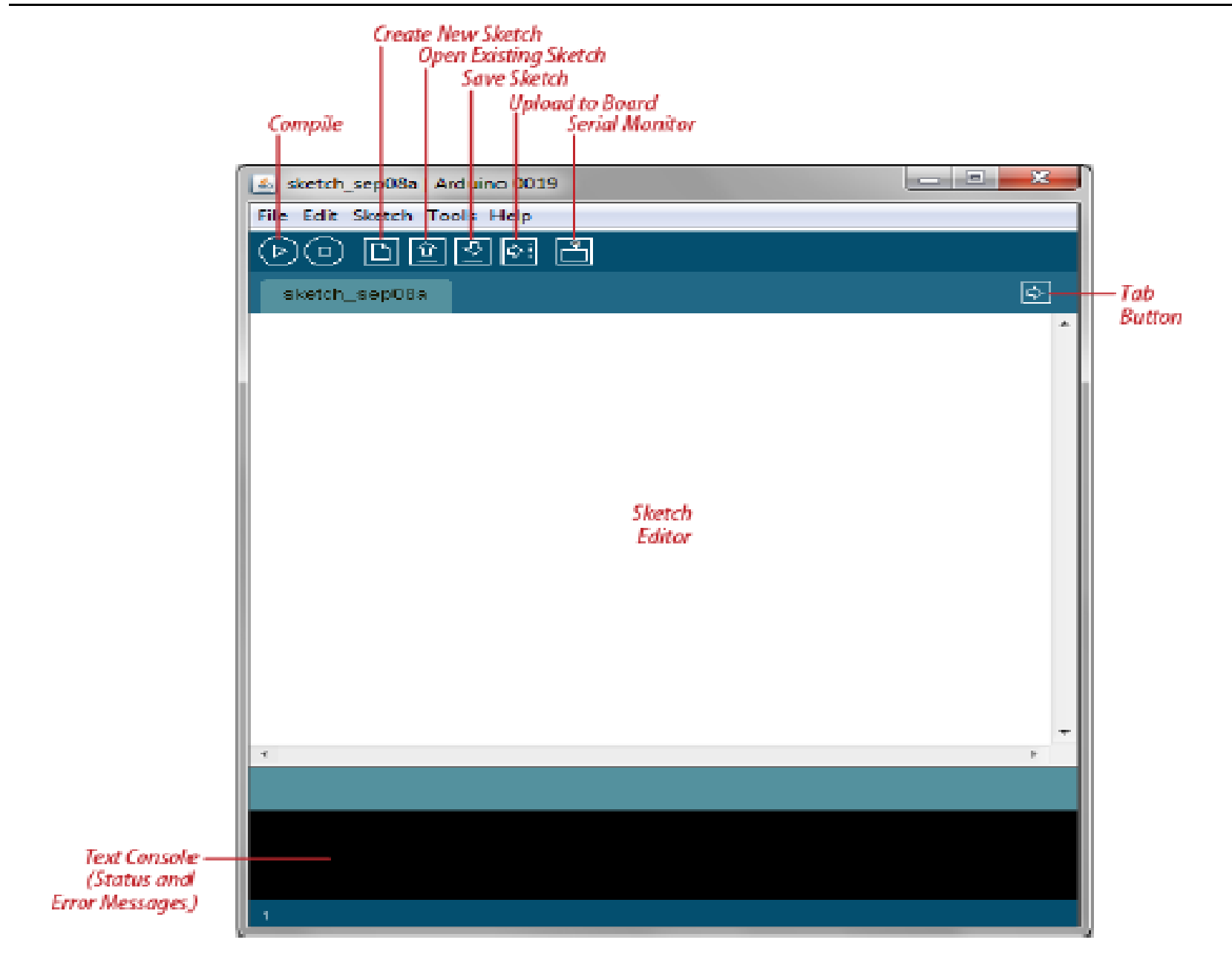
- **For PC Users →**

1. Let the installer copy and move the files to the appropriate locations, or
2. Create a folder under C:\Program Files (x86) called Arduino. Move the entire Arduino program folder here.

- **For Mac Users →**

1. Move the Arduino executable to the dock for ease of access.
2. Resist the temptation to run these from your desktop.

# Arduino IDE



# Using the Arduino IDE

Name of sketch

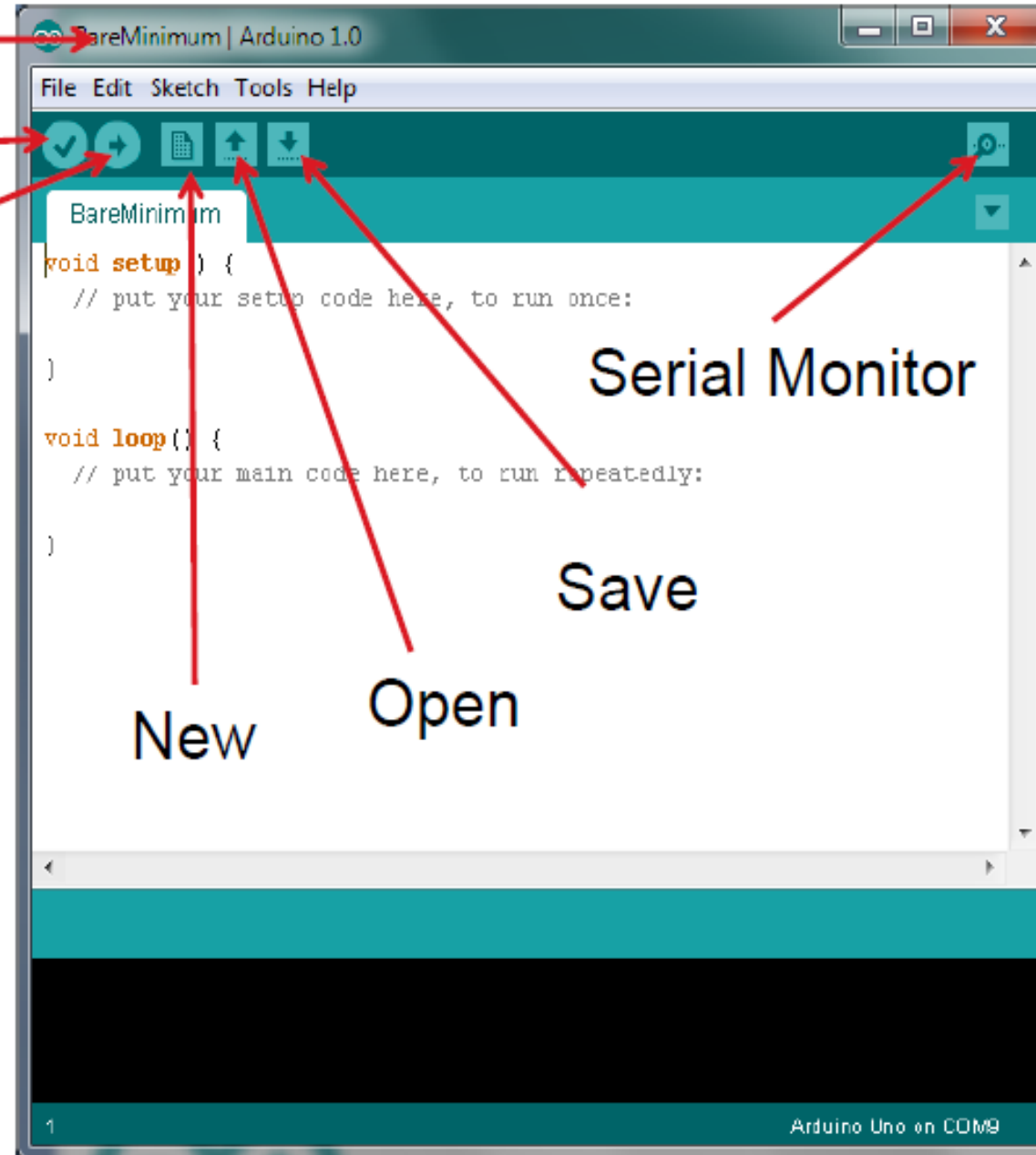
Compile sketch

Upload to board

Program area

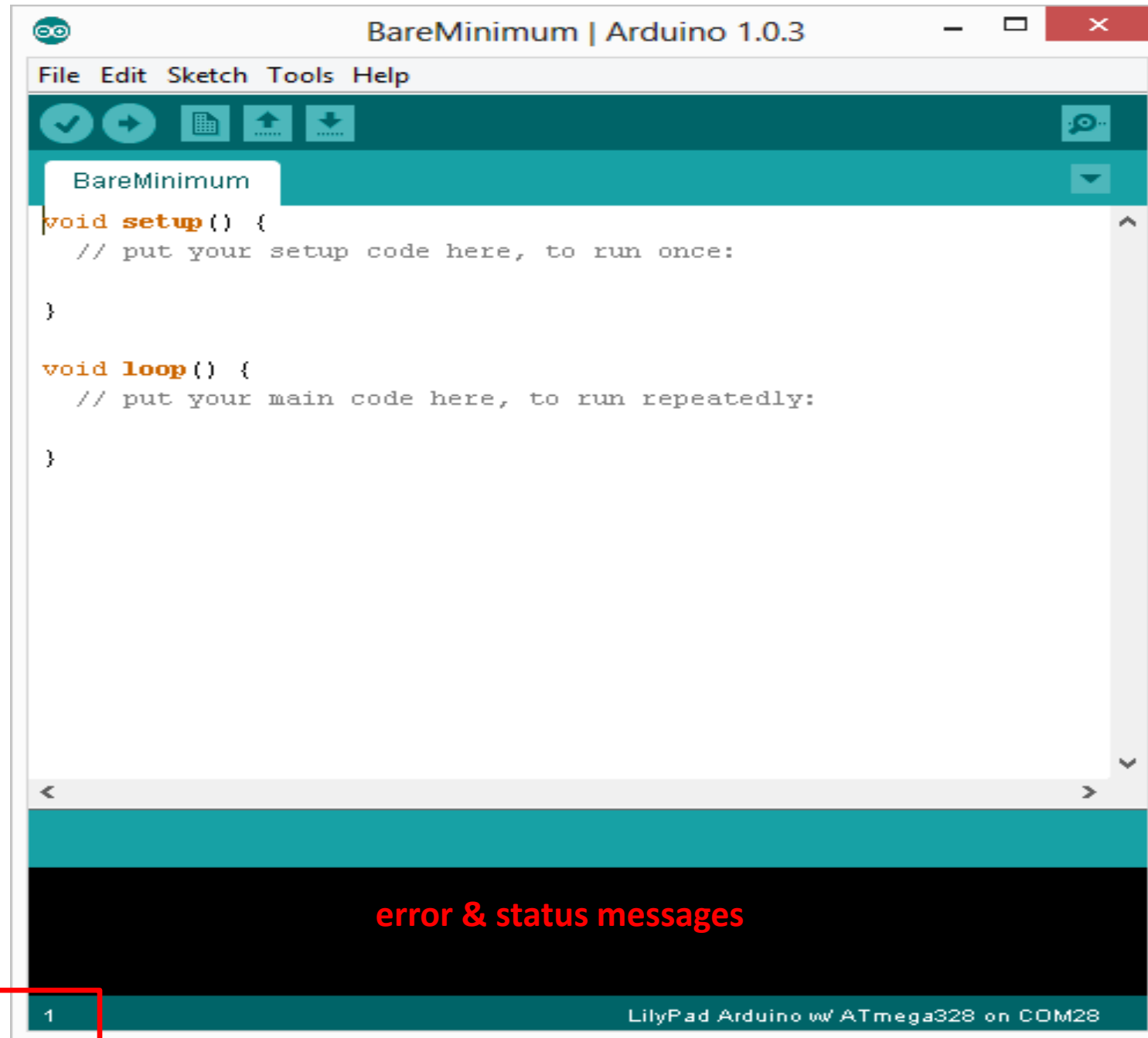


Messages /  
Errors



# Arduino

## Integrated Development Environment (IDE)

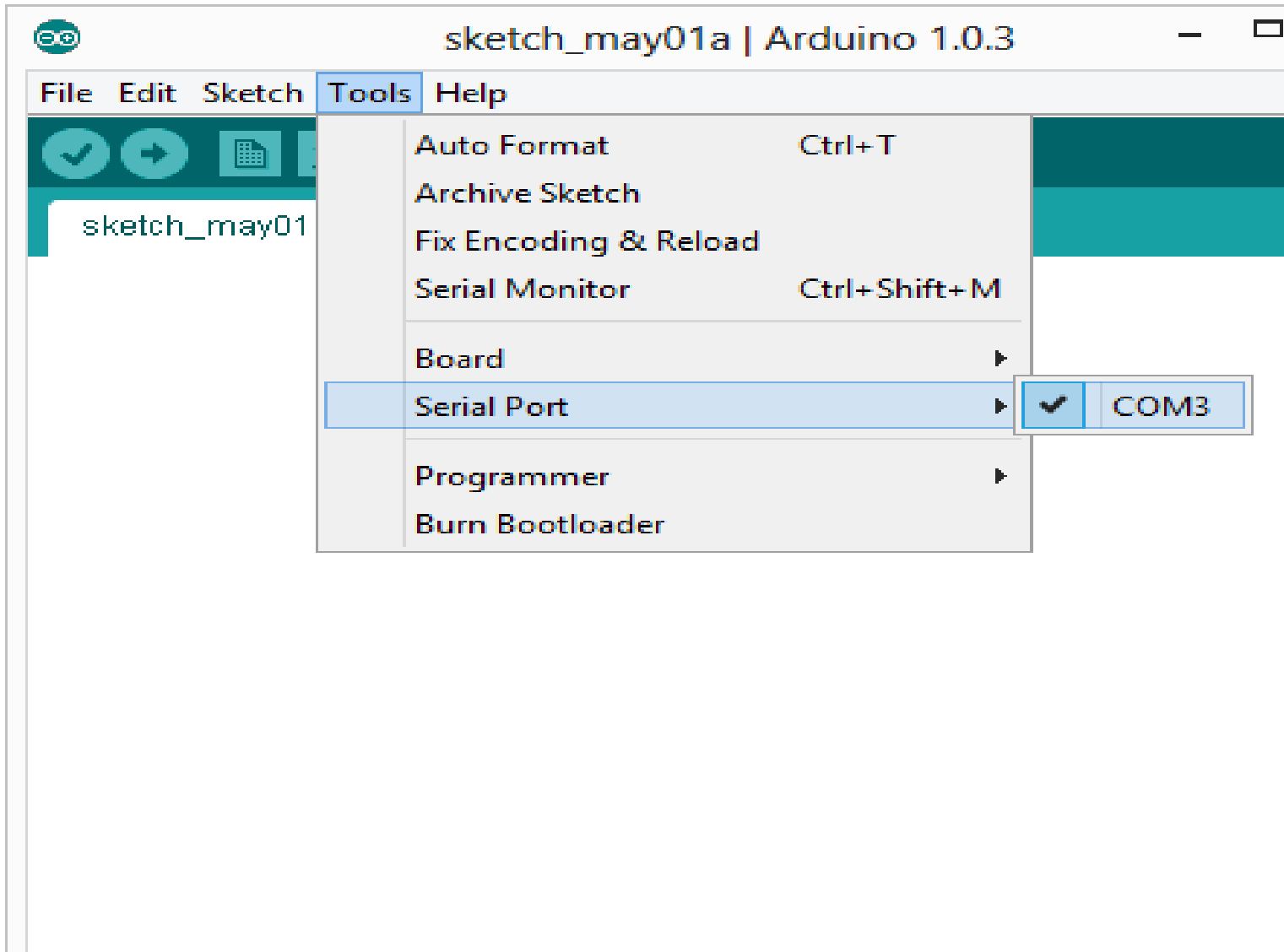


Two required functions / methods / routines:

```
void setup()  
{  
  // runs once  
}  
  
void loop()  
{  
  // repeats  
}
```



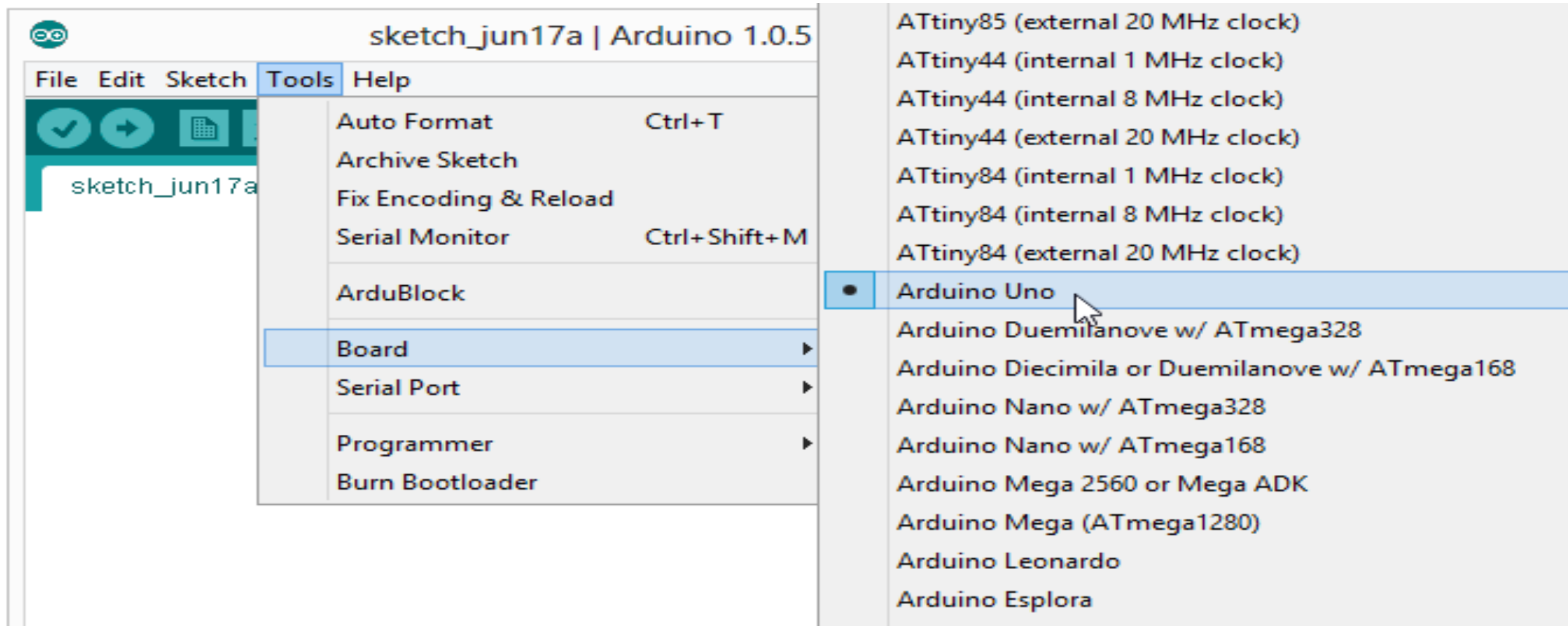
# Settings: Tools → Serial Port



- Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter.

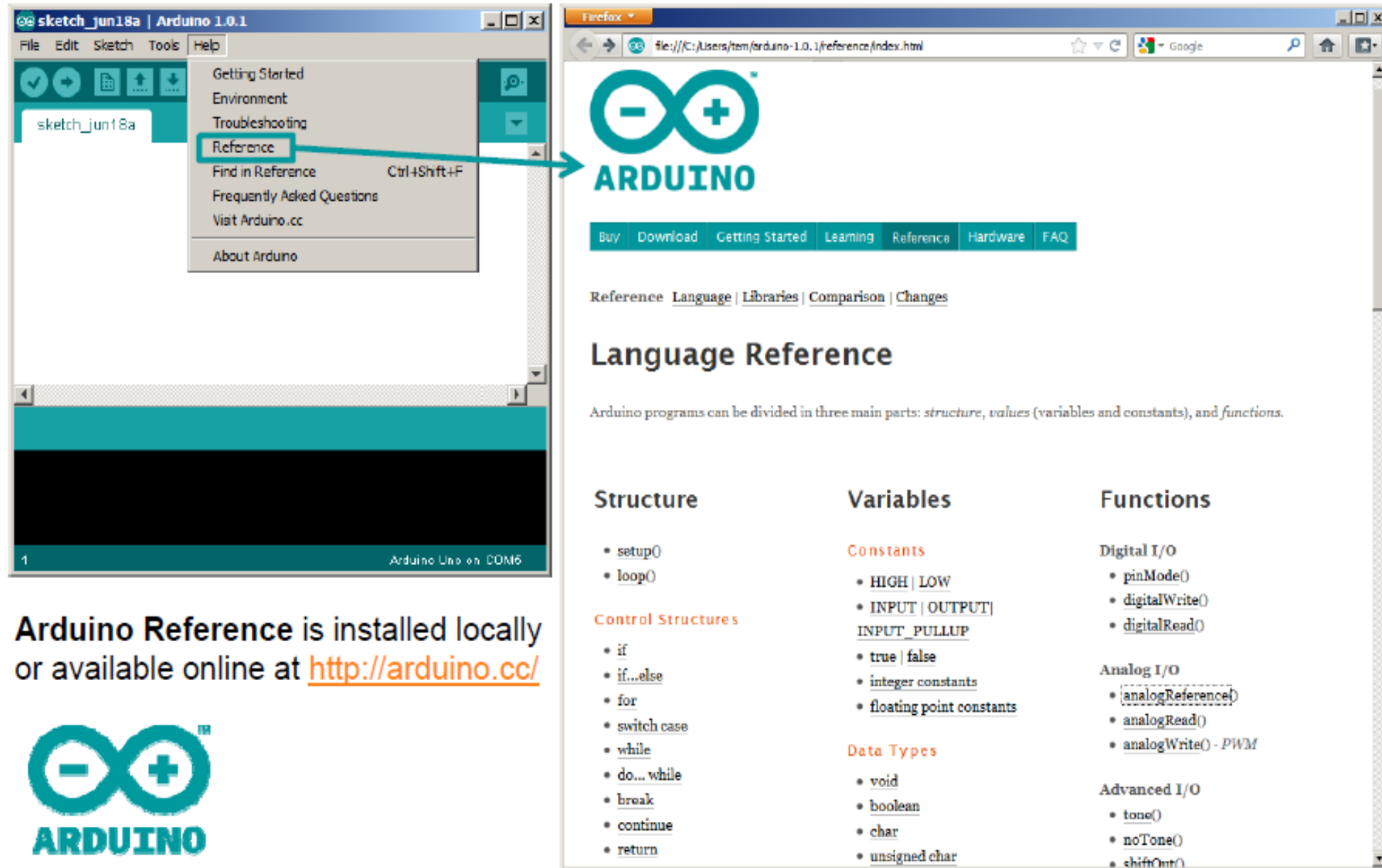
- Check to make sure that the drivers are properly installed.

# Settings: Tools → Board




- Next, double-check that the proper board is selected under the Tools→Board menu.

# Arduino Reference



The image shows two windows side-by-side. On the left is the Arduino IDE (version 1.0.1) with the 'Help' menu open and 'Reference' selected. On the right is a Firefox browser window displaying the local reference page at `file:///C:/Users/tem/arduino-1.0.1/reference/index.html`. The page features the Arduino logo, navigation links, and a 'Language Reference' section with sub-sections for Structure, Variables, and Functions.

Arduino Reference is installed locally or available online at <http://arduino.cc/>



The Arduino logo consists of a teal infinity symbol with a minus sign on the left and a plus sign on the right, with the word 'ARDUINO' in teal capital letters below it.

## Structure

- `setup()`
- `loop()`

### Control Structures

- `if`
- `if...else`
- `for`
- `switch case`
- `while`
- `do... while`
- `break`
- `continue`
- `return`

## Variables

### Constants

- `HIGH` | `LOW`
- `INPUT` | `OUTPUT`
- `INPUT_PULLUP`
- `true` | `false`
- `integer constants`
- `floating point constants`

### Data Types

- `void`
- `boolean`
- `char`
- `unsigned char`

## Functions

### Digital I/O

- `pinMode()`
- `digitalWrite()`
- `digitalRead()`

### Analog I/O

- `analogReference()`
- `analogRead()`
- `analogWrite()` - *PWM*

### Advanced I/O

- `tone()`
- `noTone()`
- `shiftOut()`

```
File Edit Sketch Tools Help
BareMinimum $
// Name of sketch
// Brief Description
// Date:
//

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

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

← ← ← comments

# Three commands to know...

- `pinMode` (pin, INPUT/OUTPUT);
- ex: `pinMode` (13, OUTPUT);
- `digitalWrite` (pin, HIGH/LOW);
- ex: `digitalWrite` (13, HIGH);
- `delay` (time\_ms);
- ex: `delay` (2500); // delay of 2.5 sec.
- **// NOTE: -> commands are CASE-sensitive**

# Arduino C Specific Functions

- **pinMode(*pin*, *mode*)**
  - Designates the specified pin for input or output
- **digitalWrite(*pin*, *value*)**
  - Sends a voltage level to the designated pin
- **digitalRead(*pin*)**
  - Reads the current voltage level from the designated pin
- **analog versions of above**
  - analogRead's range is 0 to 1023
- **serial commands**
  - print, println, write

# Terminology

“*sketch*” – a program you write to run on an Arduino board

“*pin*” – an input or output connected to something.  
e.g. output to an LED, input from a knob.

“*digital*” – value is either HIGH or LOW.  
(aka on/off, one/zero) e.g. switch state

“*analog*” – value ranges, usually from 0-255.  
e.g. LED brightness, motor speed, etc.

# 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

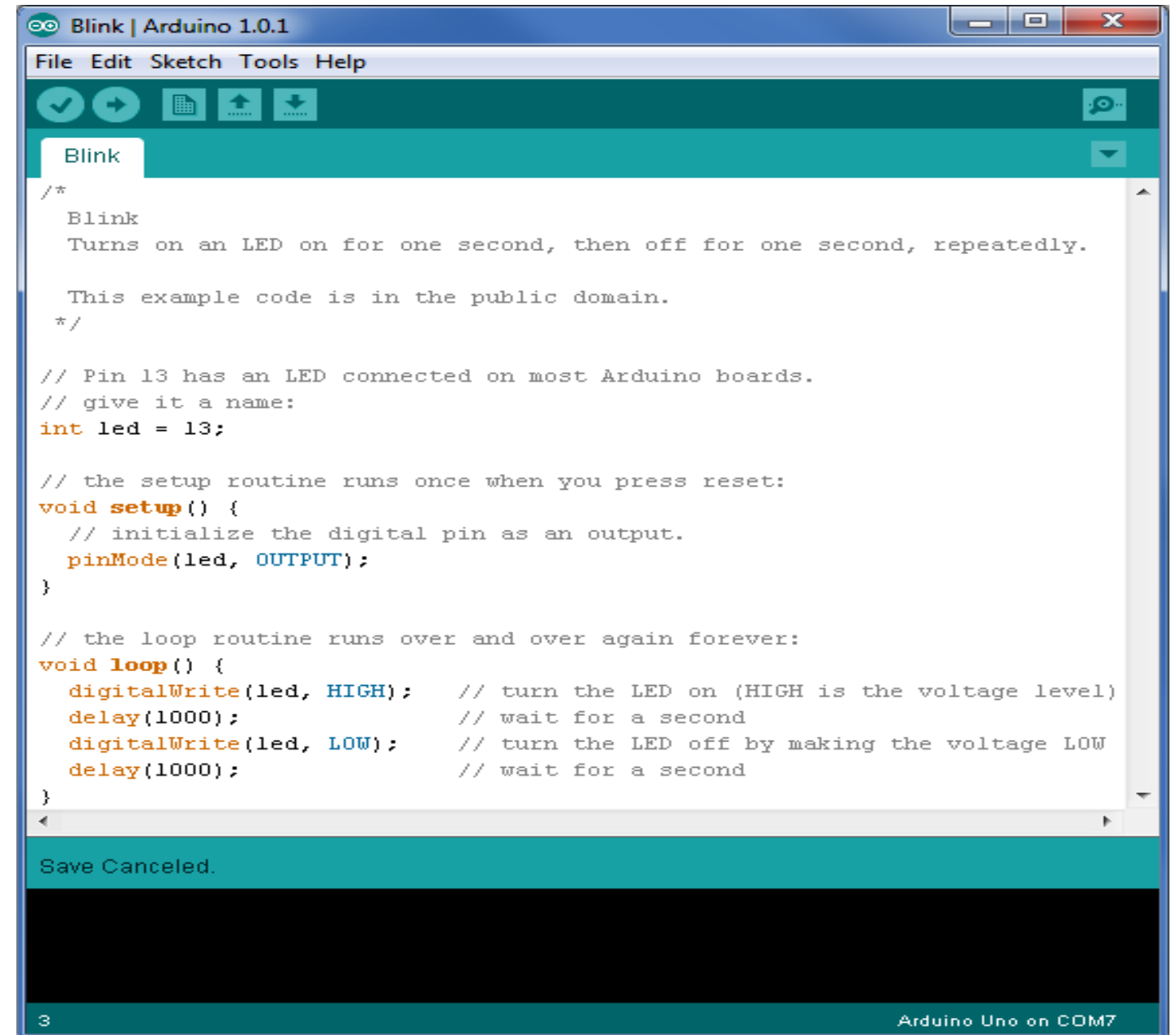
- A pin on arduino can be set as input or output by using pinMode function.
- `pinMode(13, OUTPUT); // sets pin 13 as output pin`
- `pinMode(13, INPUT); // sets pin 13 as input pin`

# 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`

# Compiler Features

- Numerous sample sketches are included in the compiler
- Located under File, Examples
- Once a sketch is written, it is uploaded by clicking on File, Upload, or by pressing <Ctrl> U



The screenshot shows the Arduino IDE interface with the 'Blink' sketch open. The code is as follows:

```
Blink | Arduino 1.0.1
File Edit Sketch Tools Help
Blink
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}

Save Canceled.
3 Arduino Uno on COM7
```

# Reading/Writing Analog Values

- `analogRead(A0); // used to read the analog value from the pin A0`
- `analogWrite(2,128);`

# Digital I/O

`pinMode(pin, mode)`

Sets pin to either INPUT or OUTPUT

`digitalRead(pin)`

Reads HIGH or LOW from a pin

`digitalWrite(pin, value)`

Writes HIGH or LOW to a pin

## Electronic stuff

Output pins can provide 40 mA of current

Writing HIGH to an input pin installs a 20K $\Omega$  pullup

# Arduino Timing

- `delay (ms)`
  - Pauses for a few milliseconds
- `delayMicroseconds (us)`
  - Pauses for a few microseconds
- More commands: [arduino.cc/en/Reference/HomePage](http://arduino.cc/en/Reference/HomePage)

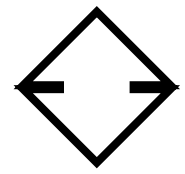
**“Serial Communication”**



# Serial Communication

**Method used to transfer data between two devices.**

Data passes between the computer and Arduino through the USB cable. Data is transmitted as zeros ('0') and ones ('1') sequentially.



Arduino dedicates Digital I/O pin # 0 to receiving and Digital I/O pin #1 to transmit.



# Serial Monitor & analogRead()



```
sketch_apr02a | Arduino 1.0.3
File Edit Sketch Tools Help
sketch_apr02a $
// analogRead() & Serial.print()
//
//
int sensorValue = 0;
int sensorPin = A0;

void setup()
{
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

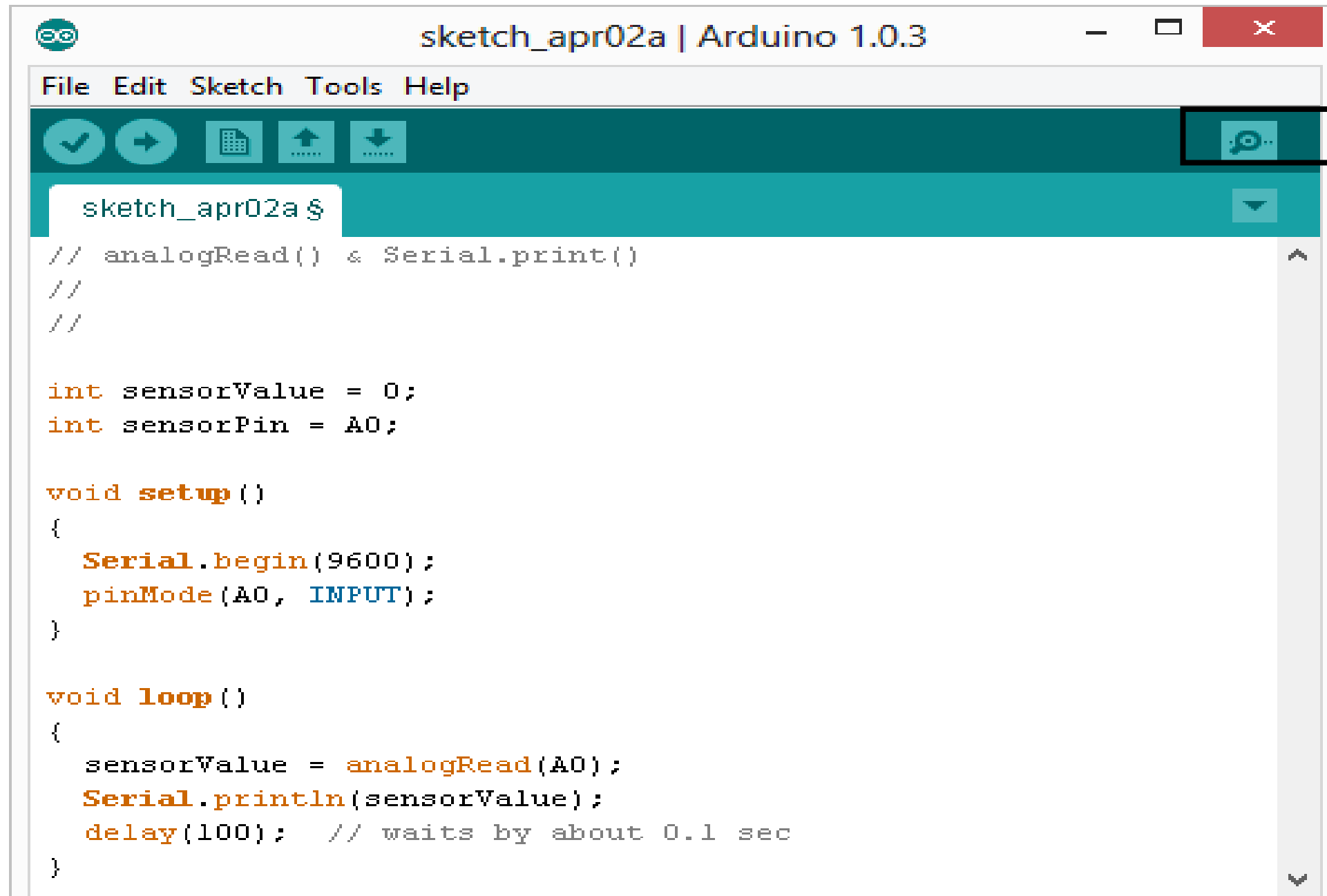
void loop()
{
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(100); // waits by about 0.1 sec
}
```

Initializes the Serial Communication

9600 baud data rate

prints data to serial bus

# Serial Monitor & analogRead()

The image shows a screenshot of the Arduino IDE interface. The window title is "sketch\_apr02a | Arduino 1.0.3". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with several icons. The Serial Monitor icon, which is a speech bubble with a magnifying glass, is highlighted with a black rectangular box. An arrow points from the text "Opens up a Serial Terminal Window" to this icon. The main area of the IDE contains a sketch with the following code:

```
sketch_apr02a $  
// analogRead() & Serial.print()  
//  
//  
  
int sensorValue = 0;  
int sensorPin = A0;  
  
void setup()  
{  
  Serial.begin(9600);  
  pinMode(A0, INPUT);  
}  
  
void loop()  
{  
  sensorValue = analogRead(A0);  
  Serial.println(sensorValue);  
  delay(100); // waits by about 0.1 sec  
}
```

Opens up a Serial Terminal Window

# Additional Serial Communication

## Sending a Message

```
void loop ( )  
{  
  Serial.print("Hands on ") ;  
  Serial.print("Learning ") ;  
  • Serial.println("is Fun!!!") ;  
}
```

BareMinimum | Arduino 1.0.5

File Edit Sketch Tools Help

COM24

Send

```
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands on Learning is Fun!!!  
Hands o
```

Autoscroll      No line ending      9600 baud

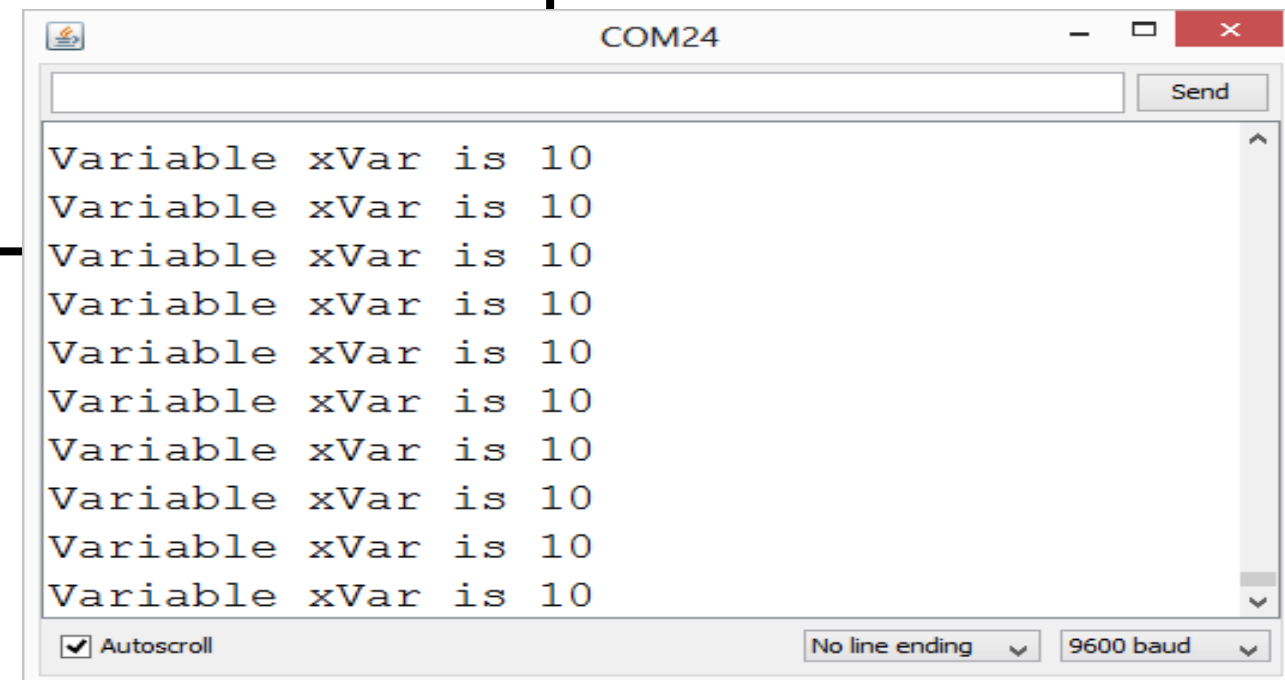
Done uploading.

Binary sketch size: 1,980 bytes (of a 32,256 byte maximum)

3      Arduino Uno on COM24

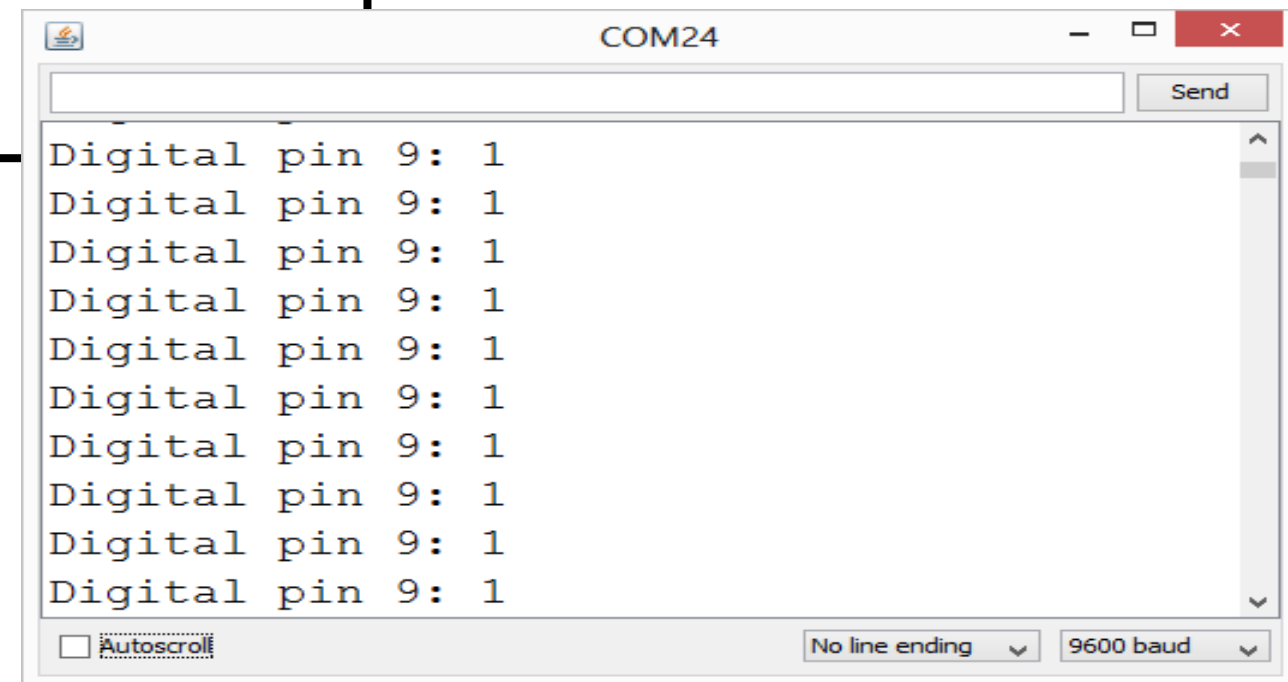
# Serial Communication: Serial Debugging

```
void loop()  
{  
  int xVar = 10;  
  Serial.print ( "Variable xVar is " ) ;  
  Serial.println ( xVar ) ;  
}
```



# Serial Communication: Serial Troubleshooting

```
void loop ( )  
{  
  Serial.print ("Digital pin 9: ");  
  Serial.println (digitalRead(9));  
}
```

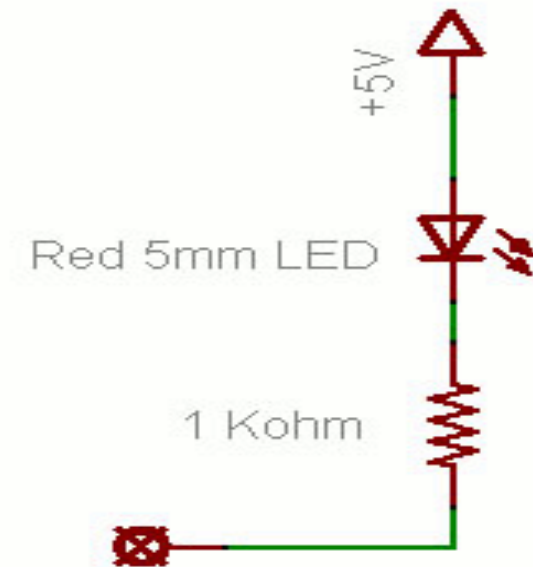
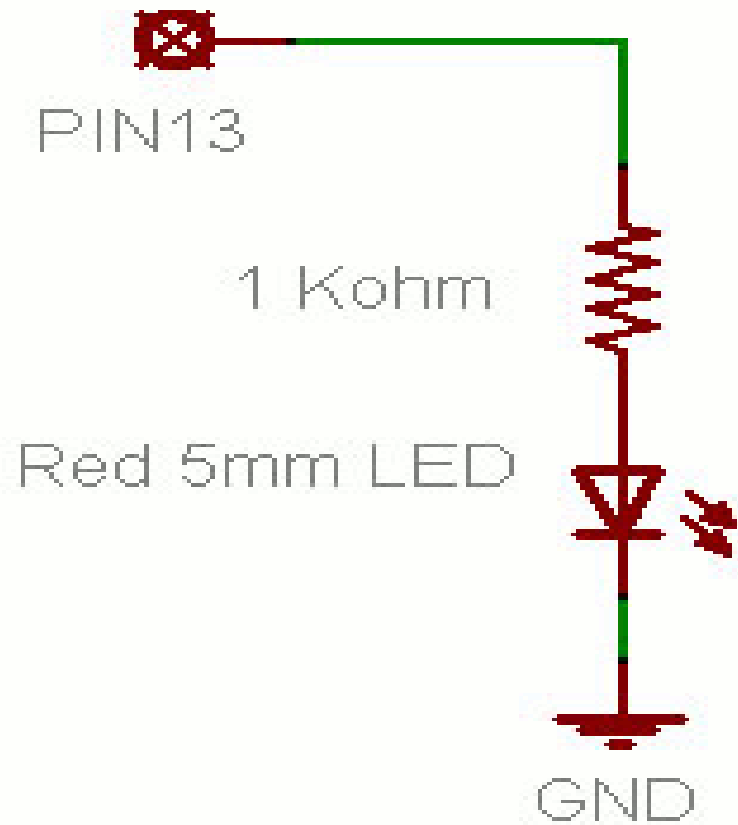


**“Arduino  
and  
Start Programming”**



# Putting It Together

- Complete the sketch (program) below.
- What output will be generated by this program?
- What if the schematic were changed?



```
void loop() // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(500); // waits for a second
  digitalWrite(ledPin, LOW); // sets the LED off
  delay(500); // waits for a second
}
```

# 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);
```

```
}
```

# Basic Arduino program- LED

```
int d1 = 1;
int d2 = 2;
int tipkalo = 3;
void setup() {
    // put your setup code here, to run once:
    pinMode(d1, OUTPUT);
    pinMode(d2, OUTPUT);
    pinMode(tipkalo, INPUT);
}

void loop() {
    // put your main code here, to run repeatedly:
    int a = digitalRead(tipkalo);
    if (a == HIGH) {
        stisnuto();
    }
    else {
        otpusteno();
    }
}
```

```
void stisnuto() {
    digitalWrite(d1, HIGH);
    digitalWrite(d2, LOW);
    delay(1000);
    digitalWrite(d2, HIGH);
    digitalWrite(d1, LOW);
    delay(1000);
}

void otpusteno() {
    digitalWrite(d1, HIGH);
    digitalWrite(d2, HIGH);
    delay(50);
    digitalWrite(d2, LOW);
    digitalWrite(d1, LOW);
    delay(50);
}
```

## Explanation of the code-declaring pins

```
int d1 = 1;  
int d2 = 2;  
int tipkalo = 3;
```

- We determine and declare pins which we will be using in our program

# Input/Output

- In the void setup we determine our Inputs and Outputs
- We chose for d1 and d2 to be the outputs while we chose “tipkalo” to be the input

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(d1, OUTPUT);  
  pinMode(d2, OUTPUT);  
  pinMode(tipkalo, INPUT);  
}
```

# Void loop & IF-ELSE loop

```
void loop() {  
  // put your main code here, to run repeatedly:  
  int a = digitalRead(tipkalo);  
  if (a == HIGH) {  
    stisnuto();  
  }  
  else {  
    otpusteno();  
  }  
}
```

- In the void loop we write the program which we want to run forever, in other words infinitely(in loops).
- In the code we are checking if the “tipkalo” is pressed or not.
- After that, we have a IF-ELSE loop with 2 subprograms, one for the case in which “tipkalo” is pressed and one for the case in which “tipkalo” is not pressed.

# Subprogram

```
void stisnuto() {  
    digitalWrite(d1, HIGH);  
    digitalWrite(d2, LOW);  
    delay(1000);  
    digitalWrite(d2, HIGH);  
    digitalWrite(d1, LOW);  
    delay(1000);  
}
```

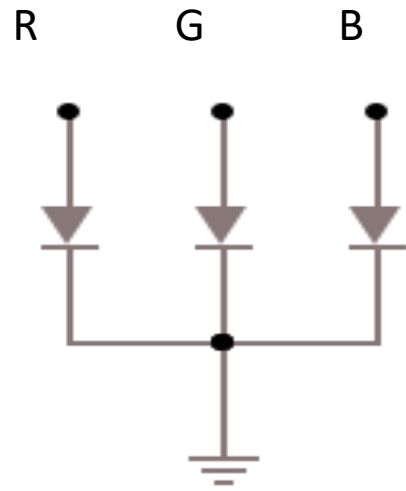
- If the “tipkalo” is pressed the LEDs will alternately light up.
- The delay is that which makes the LED blink, it puts the LED in a state of doing nothing and hence the light turns off making the illusion of blinking. We chose the delay to be 1000 microseconds, in other words 1 second.

# Subprogram

```
void otpusteno() {  
    digitalWrite(d1, HIGH);  
    digitalWrite(d2, HIGH);  
    delay(50);  
    digitalWrite(d2, LOW);  
    digitalWrite(d1, LOW);  
    delay(50);  
}
```

- If the “tipkalo” is not pressed both LEDs will blink at the same time
- They will blink in the span of 50ms.

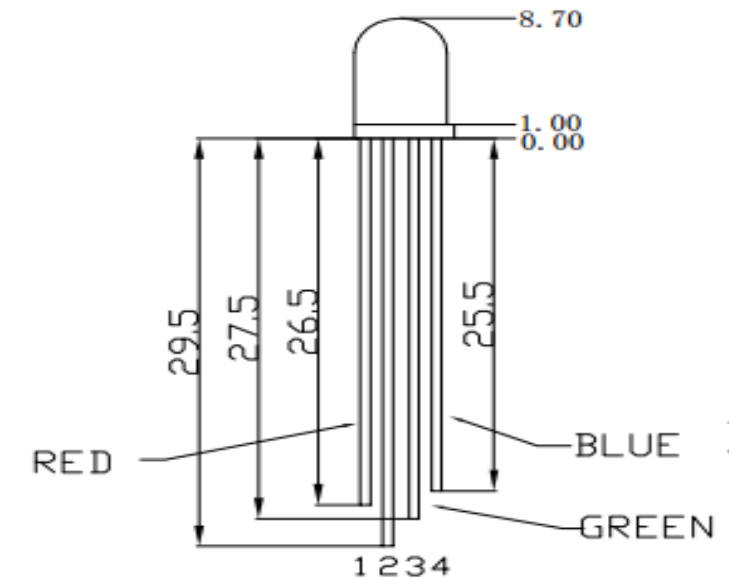




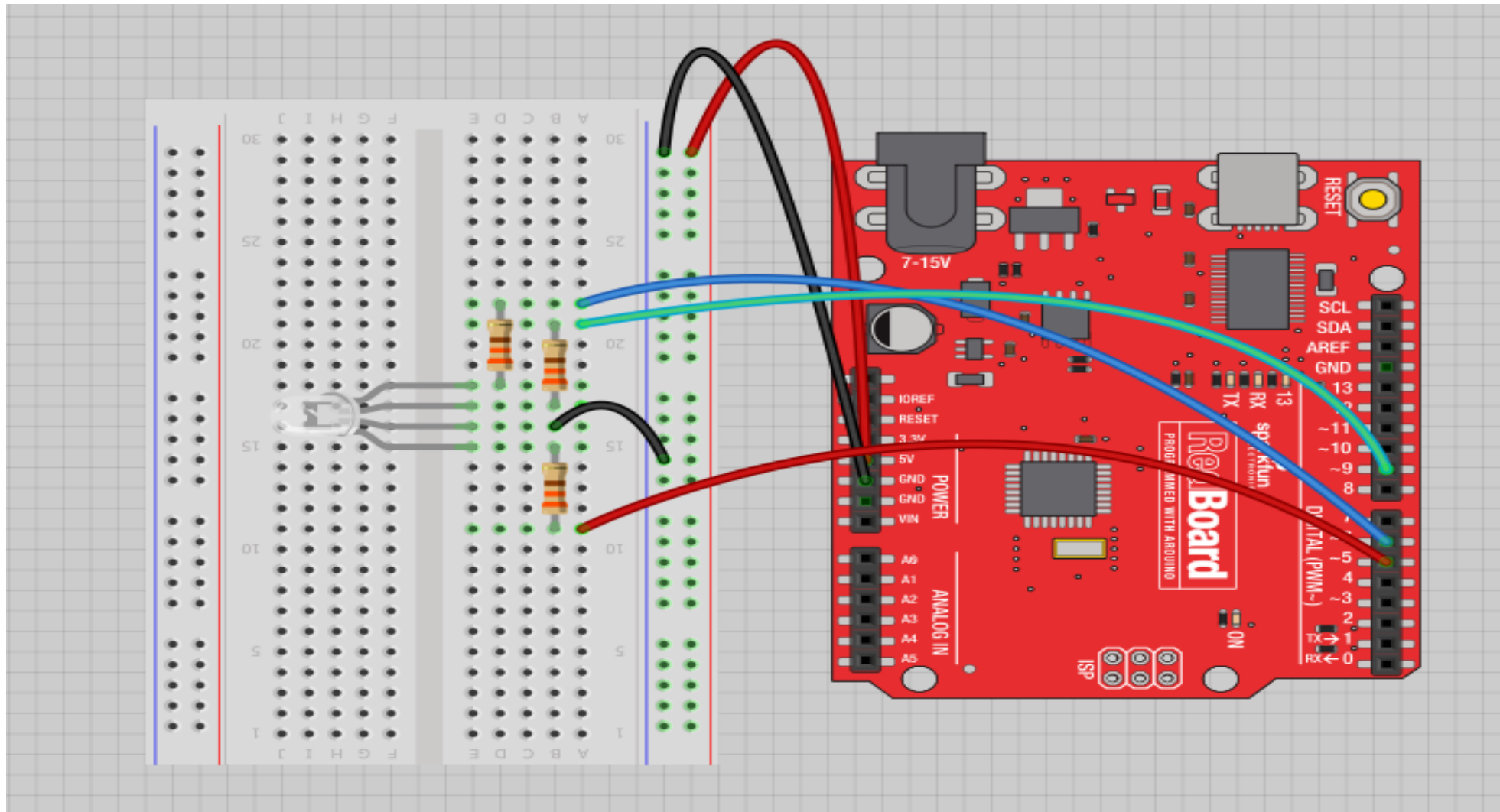
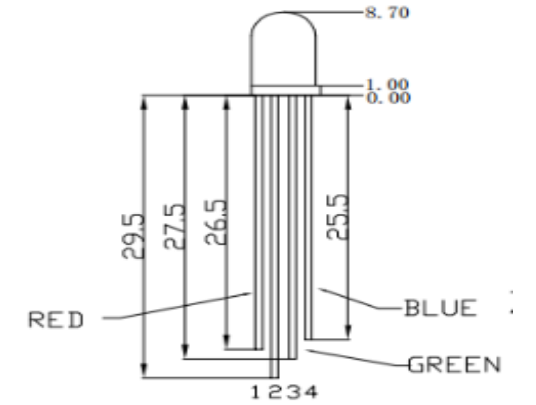
## Color Mixing Tri-color LED



- In the SIK, this is a standard – Common Cathode LED
- This means the negative side of the LED is all tied to Ground.



# RGB LED

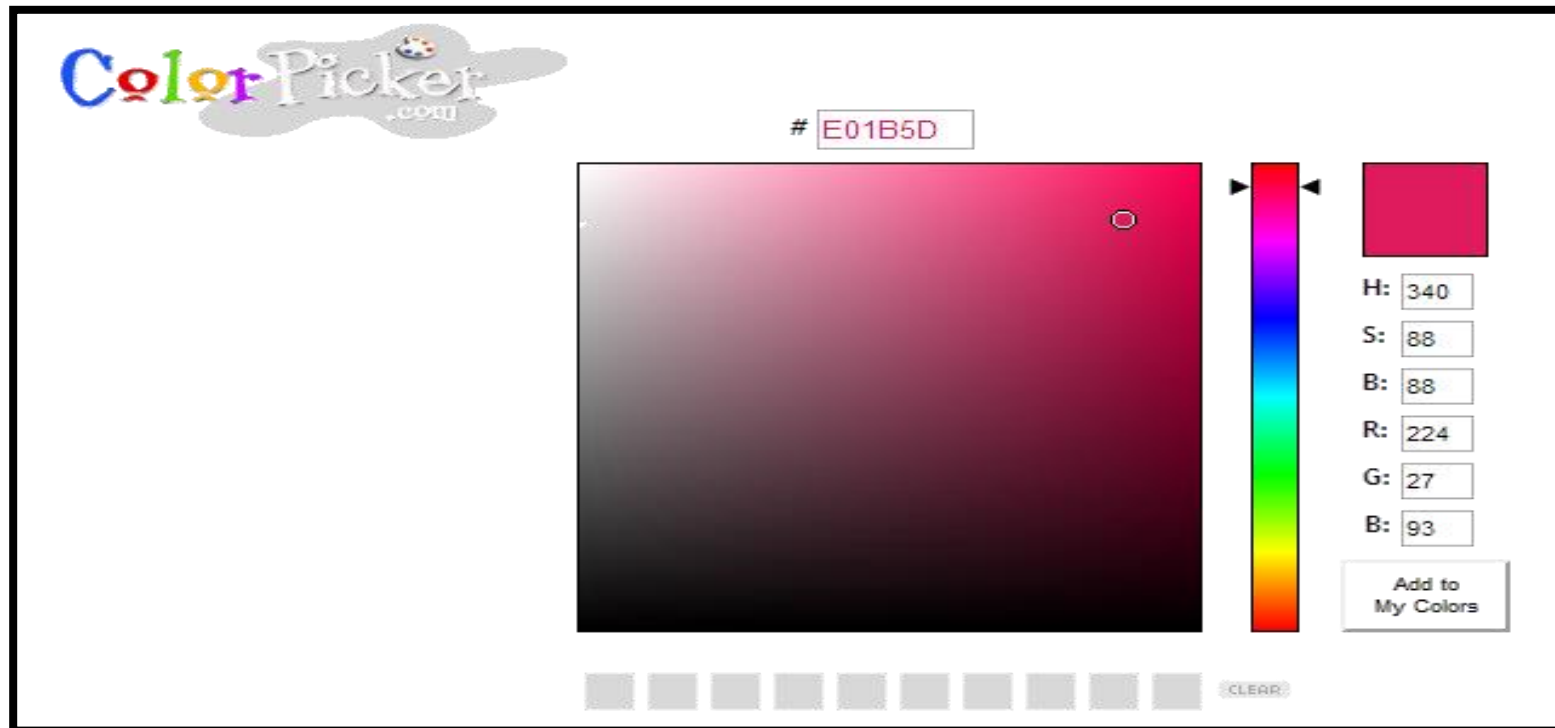


- Note: The longest leg of the RGB LED is the Common Cathode. This goes to GND.

Use pins 5, 6, & 9

# How many unique colors can you create?

$$\begin{aligned} \# \text{ of unique colors} &= 256 \cdot 256 \cdot 256 \\ &= 16,777,216 \text{ colors!} \end{aligned}$$



Play around with this with the `analogWrite()` command.

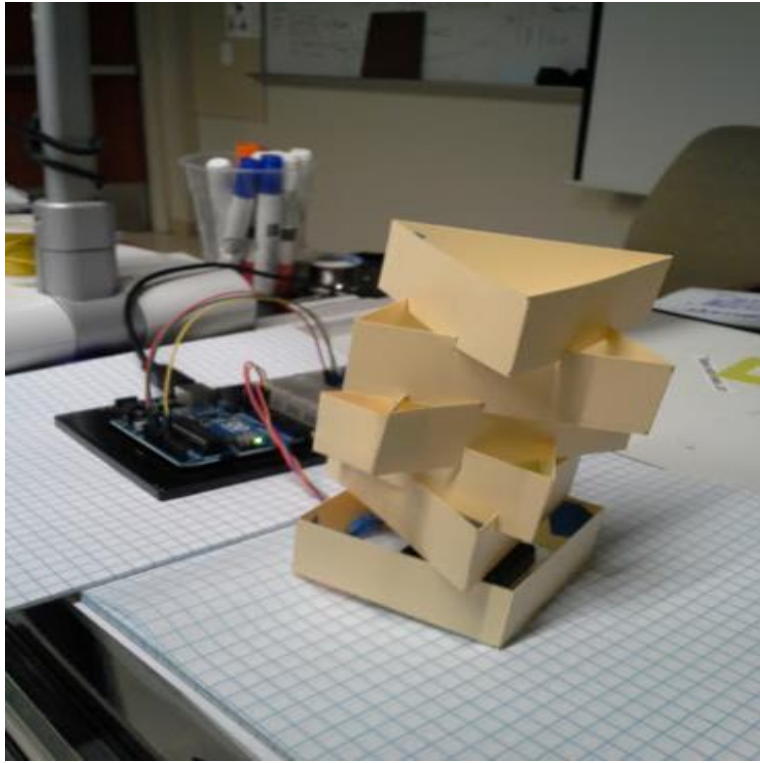
# RGB LED Color Mixing

```
• int redPin = 5;
• int greenPin = 6;
• int bluePin = 9;

• void setup()
• {
•     pinMode(redPin, OUTPUT);
•     pinMode(greenPin, OUTPUT);
•     pinMode(bluePin, OUTPUT);
• }
```

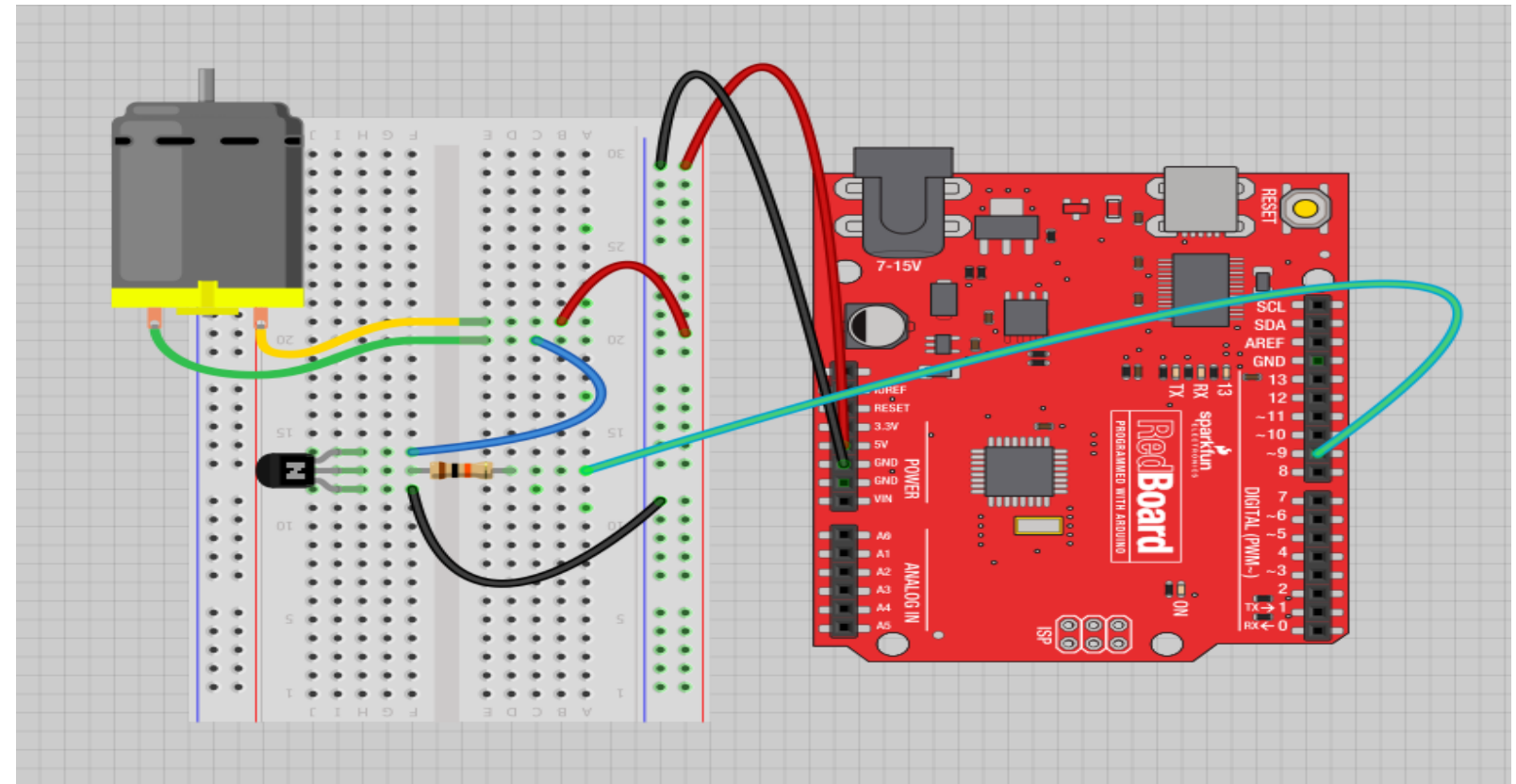
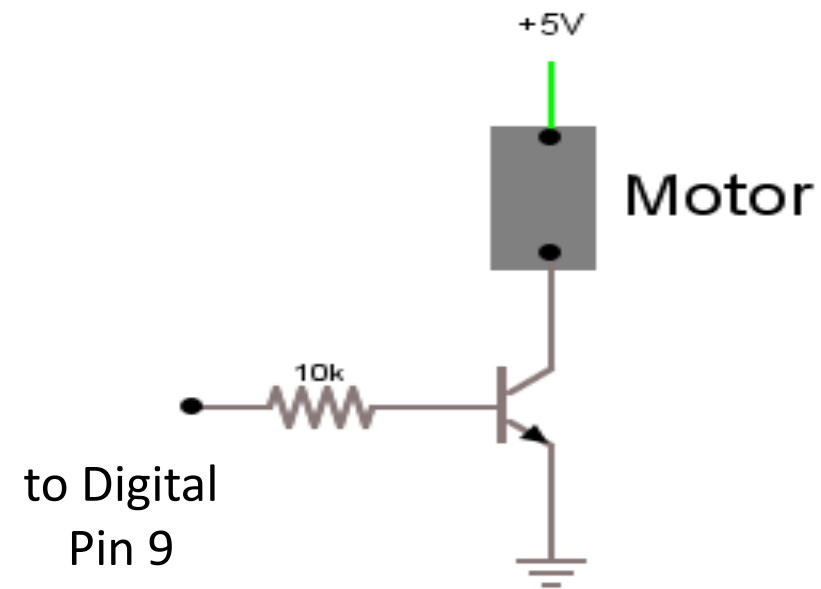
```
• void loop()
• {
•     analogWrite(redPin, 255);
•     analogWrite (greenPin, 255);
•     analogWrite (bluePin, 255);
• }
```

# Project: Mood Lamp / Light Sculpture

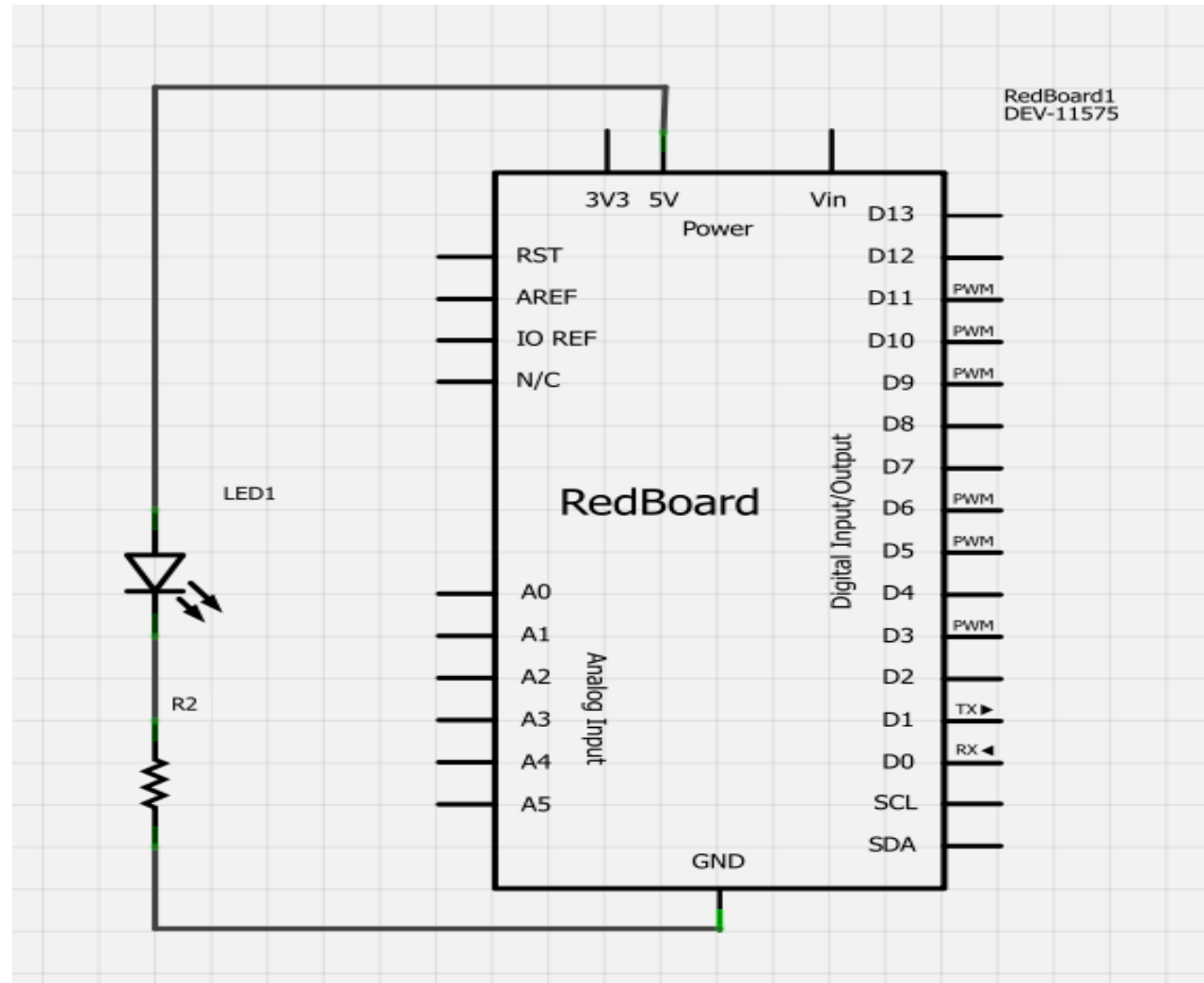


# Driving Motors or other High Current Loads

- NPN Transistor (Common Emitter “Amplifier” Circuit)



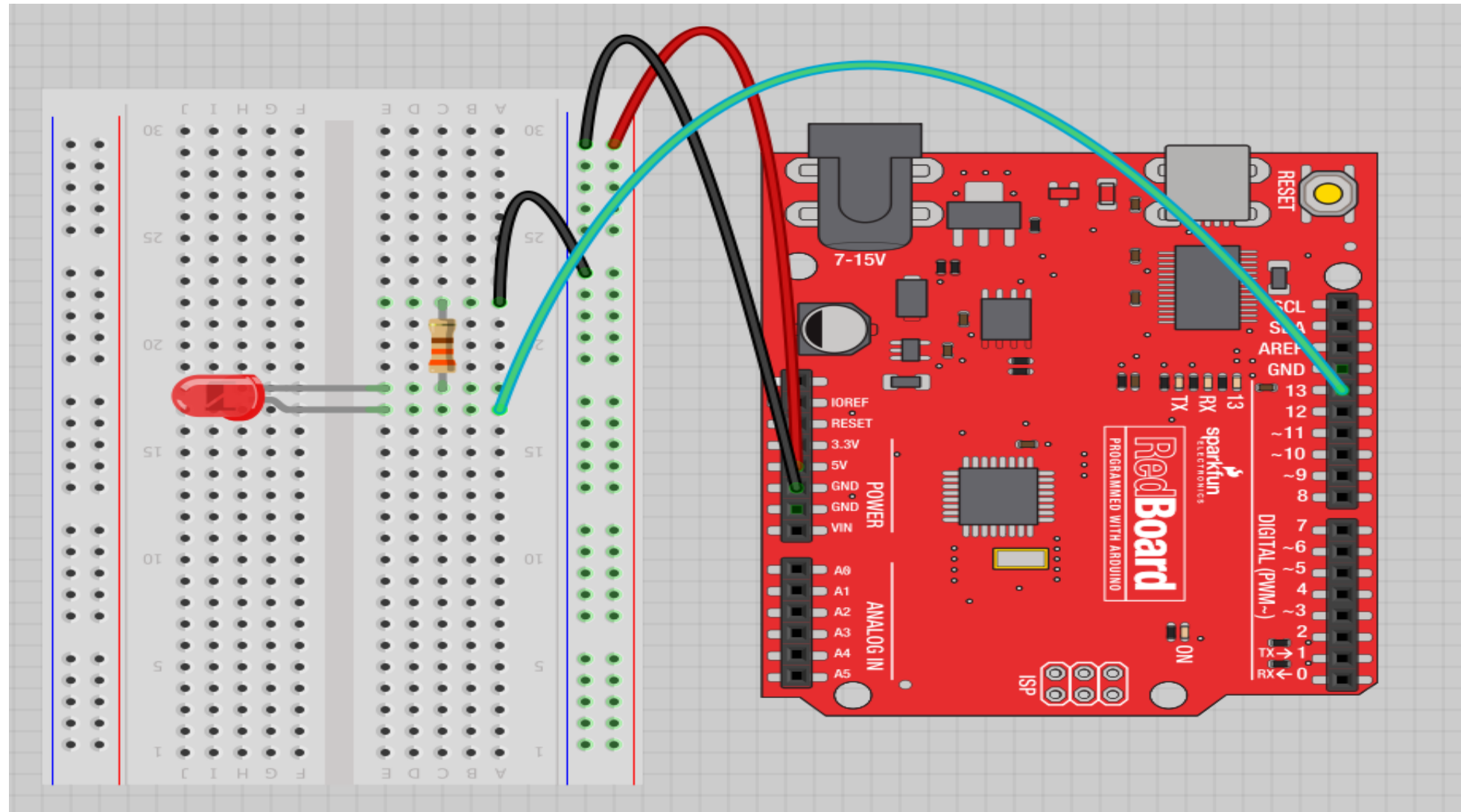
# Using the Breadboard to built a simple circuit



- Use the breadboard to wire up a single LED with a 330 Ohm Resistor (Orange-Orange-Brown).

Note: the longer leg on the LED is the positive leg and the shorter leg is the negative

# Wiring Diagram



Move the green wire from the power bus to pin 13 (or any other Digital I/O pin on the Arduino board).



# A few simple challenges

## Let's make LED#13 blink!

- **Challenge 1a** – blink with a 200 ms second interval.
- **Challenge 1b** – blink to mimic a heartbeat
- **Challenge 1c** – find the fastest blink that the human eye can still detect...
  - 1 ms delay? 2 ms delay? 3 ms delay???
  -

## Try adding other LEDs

Can you blink two, three, or four LEDs?

(Hint: Each LED will need it's own  $330\Omega$  resistor.)

Generate your own morse code flashing

How about → Knight Rider? Disco? Police Light?

# Programming Concepts: Variables

```
ProtonapProMiniExample2 $
```

```
// Comments go here
// Written by:  Joesephine Jones
// Date:  April 12, 2013

int sensorValue;
int ledPin;

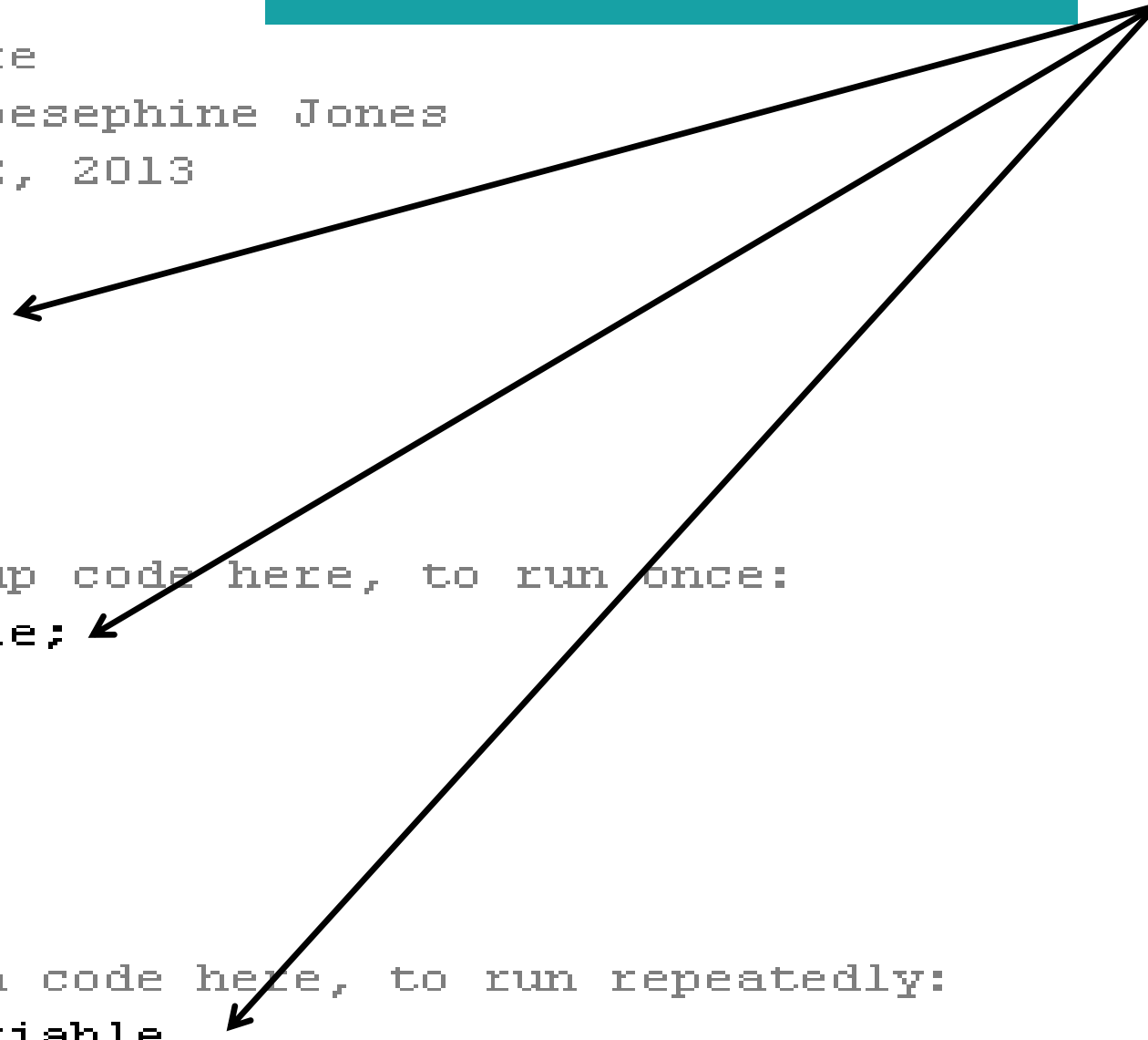
void setup()
{
  // put your setup code here, to run once:
  int setupVariable;

}

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

Variable Scope

- *Global*
- *---*
- *Function-level*

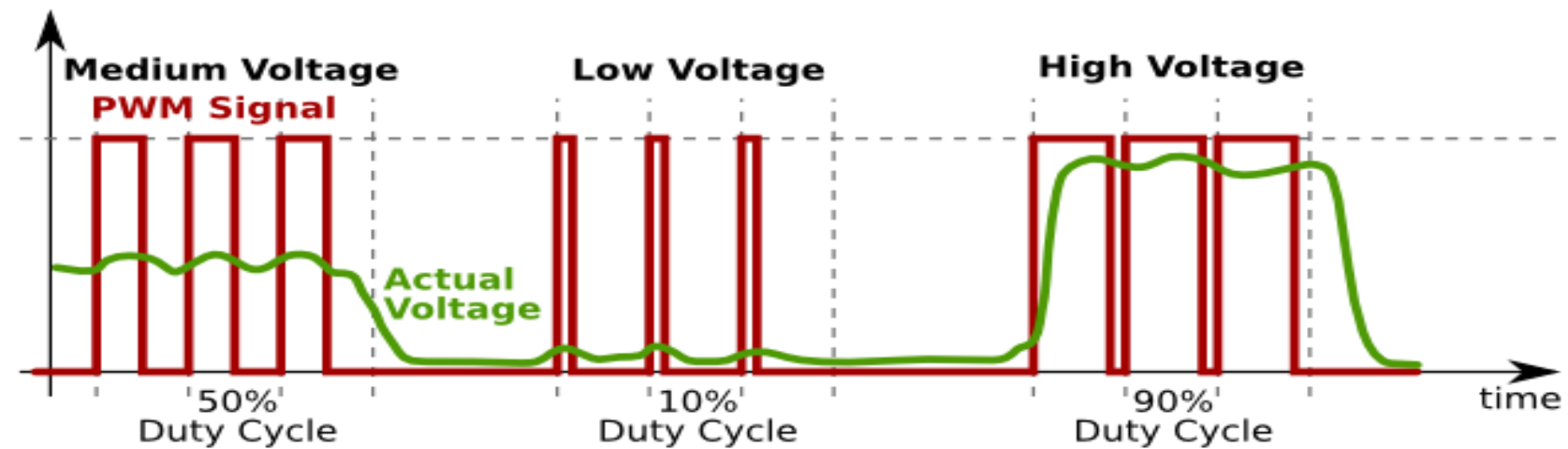


# Fading in and Fading Out (Analog or Digital?)

- A few pins on the Arduino allow for us to modify the output to mimic an analog signal.
- This is done by a technique called:
- Pulse Width Modulation (PWM)

# Pulse Width Modulation (PWM)

To create an analog signal, the microcontroller uses a technique called PWM. By varying the duty cycle, we can mimic an “average” analog voltage.



# Fading

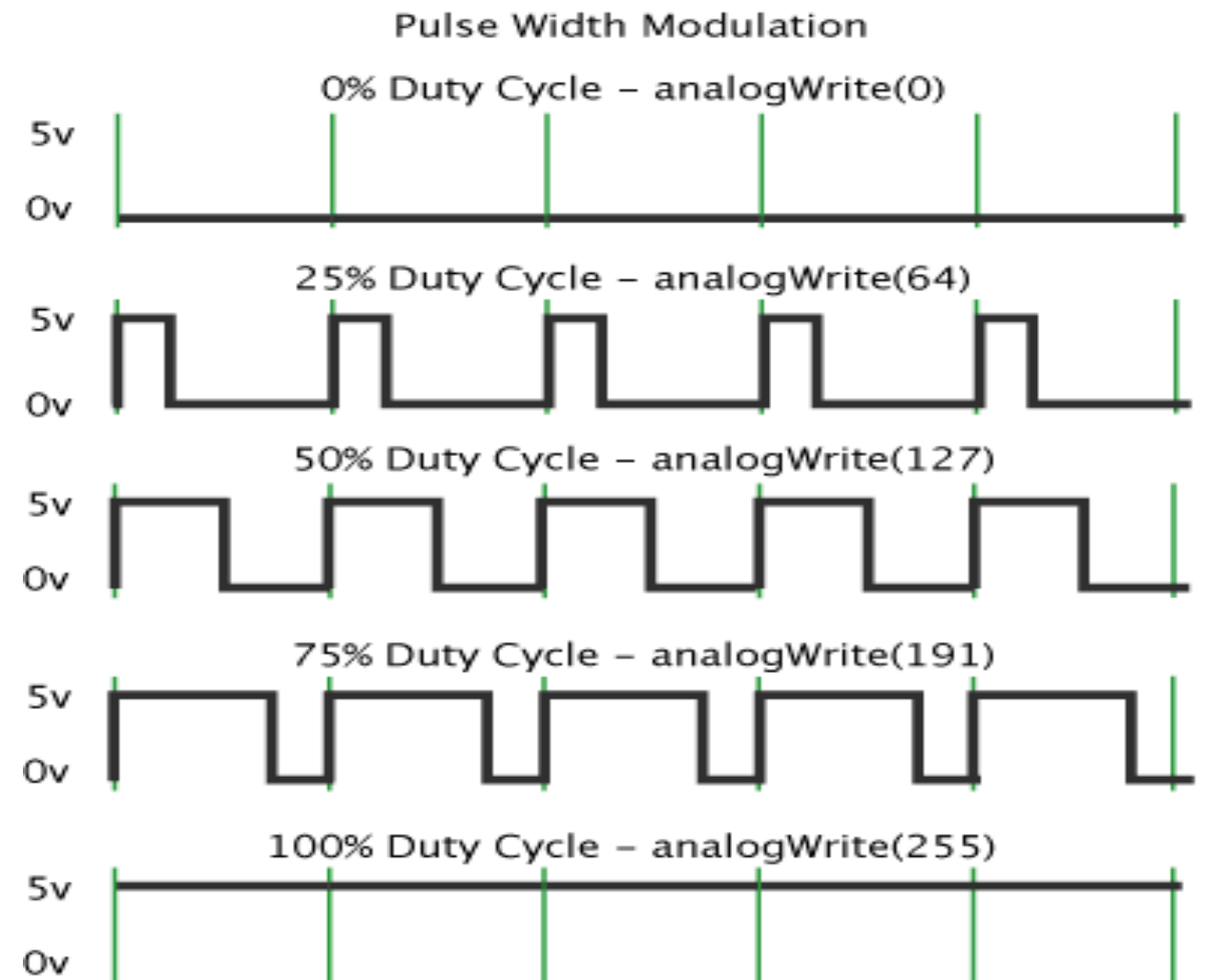
```
• analogWrite(pin, val);
```

•

• **pin** – refers to the OUTPUT pin (limited to pins 3, 5, 6, 9, 10, 11.) – denoted by a ~ symbol

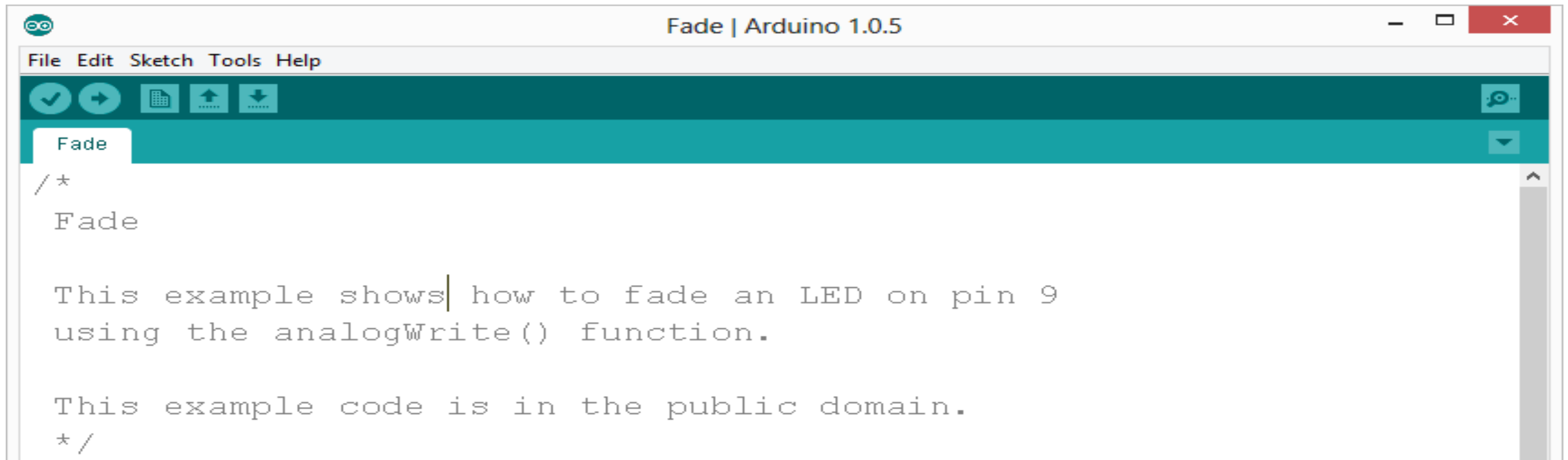
• **val** – 8 bit value (0 – 255).

• 0 => 0V | 255 => 5V



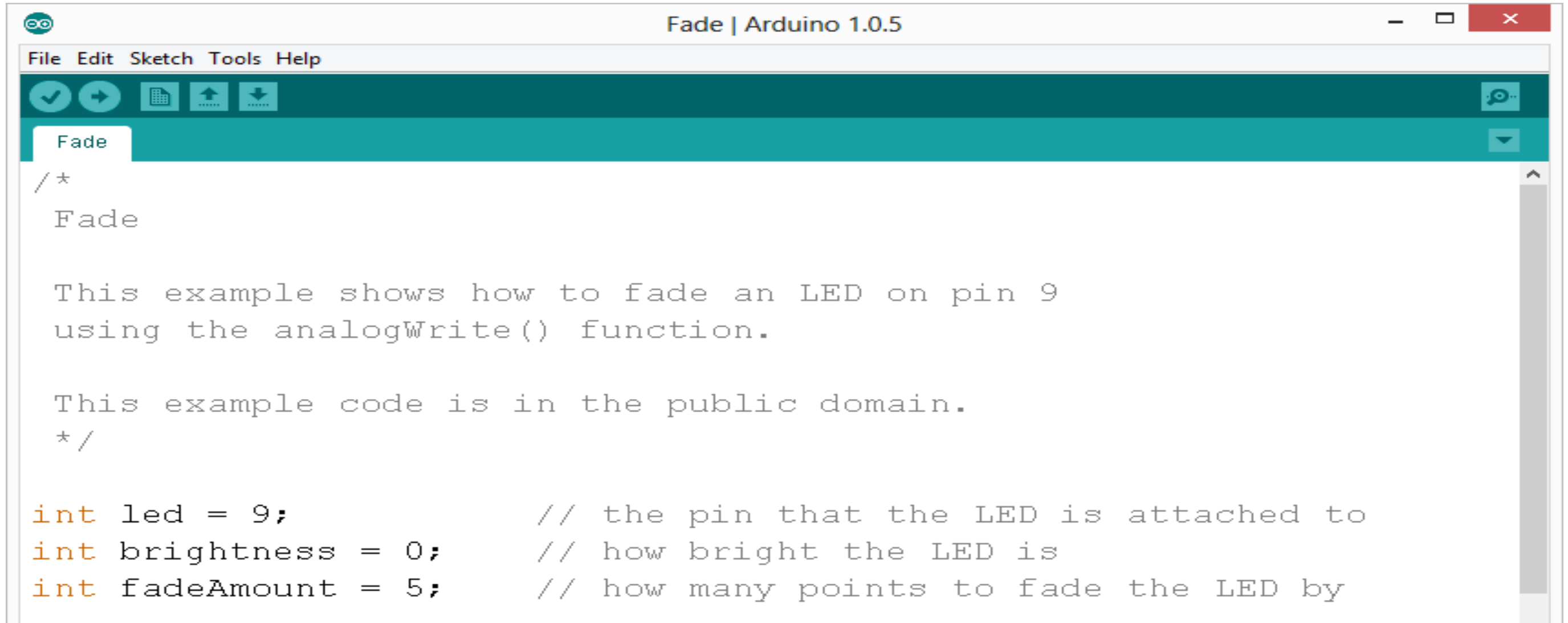
# Move one of your LED pins over to Pin 9

- In Arduino, open up:
- File → Examples → 01.Basics → Fade



```
/*  
Fade  
  
This example shows how to fade an LED on pin 9  
using the analogWrite() function.  
  
This example code is in the public domain.  
*/
```

# Fade - Code Review

A screenshot of the Arduino IDE interface. The window title is "Fade | Arduino 1.0.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checkmark, play, grid, upload, and download. A tab labeled "Fade" is active. The code editor contains the following text:

```
/*  
Fade  
  
This example shows how to fade an LED on pin 9  
using the analogWrite() function.  
  
This example code is in the public domain.  
*/  
  
int led = 9;           // the pin that the LED is attached to  
int brightness = 0;   // how bright the LED is  
int fadeAmount = 5;   // how many points to fade the LED by
```



# Fade - Code Review

```
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

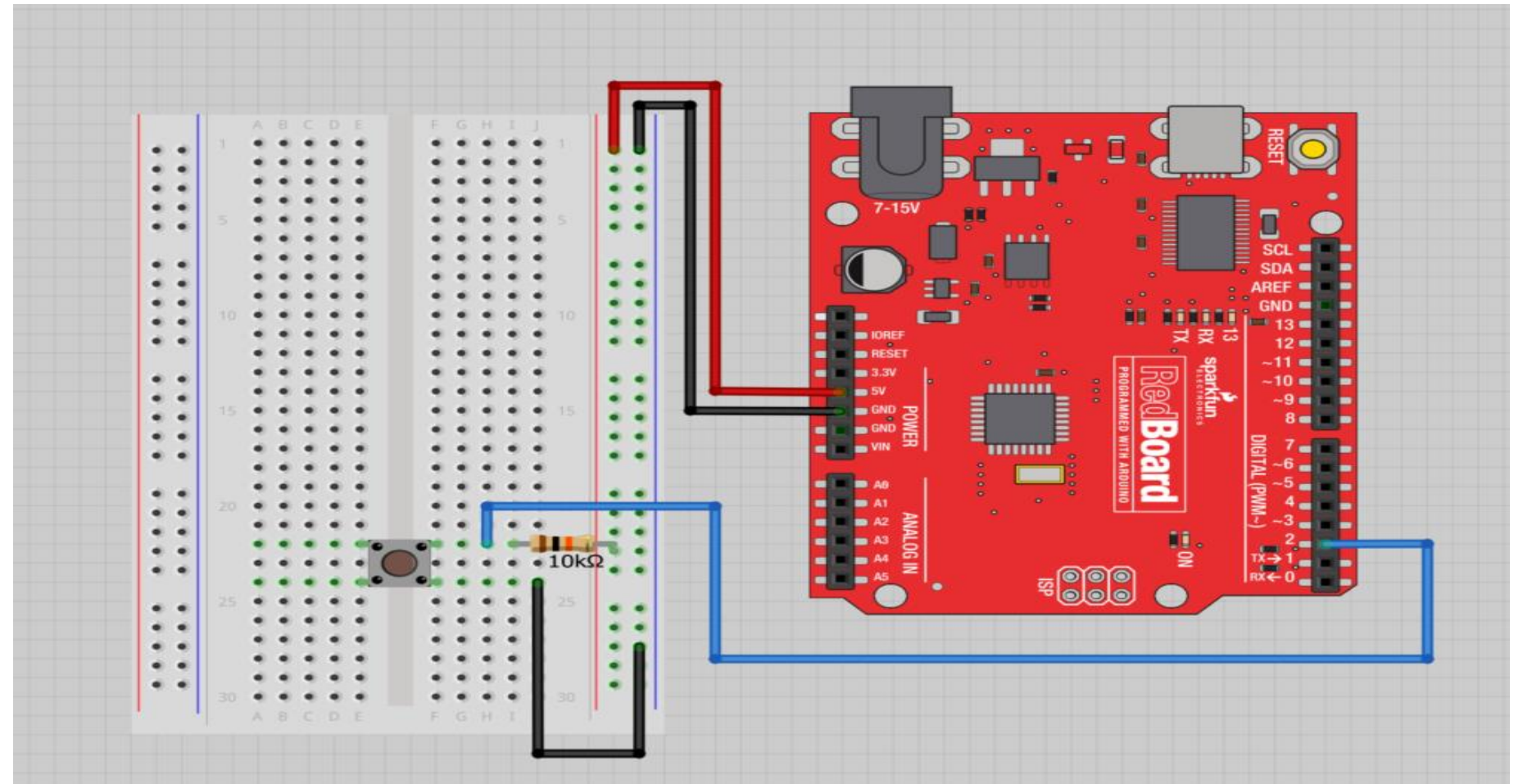
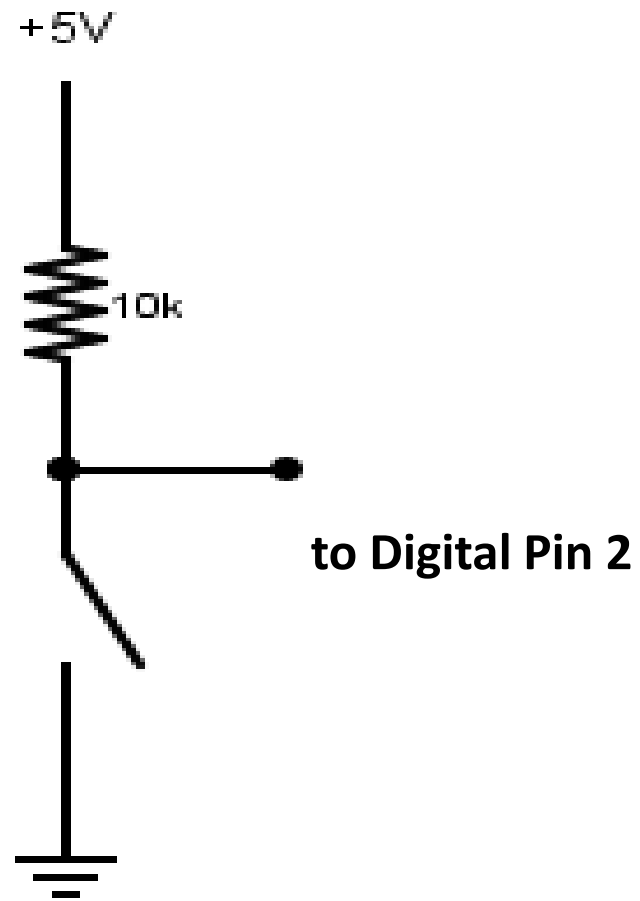
  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

# Project# 2 -- Fading

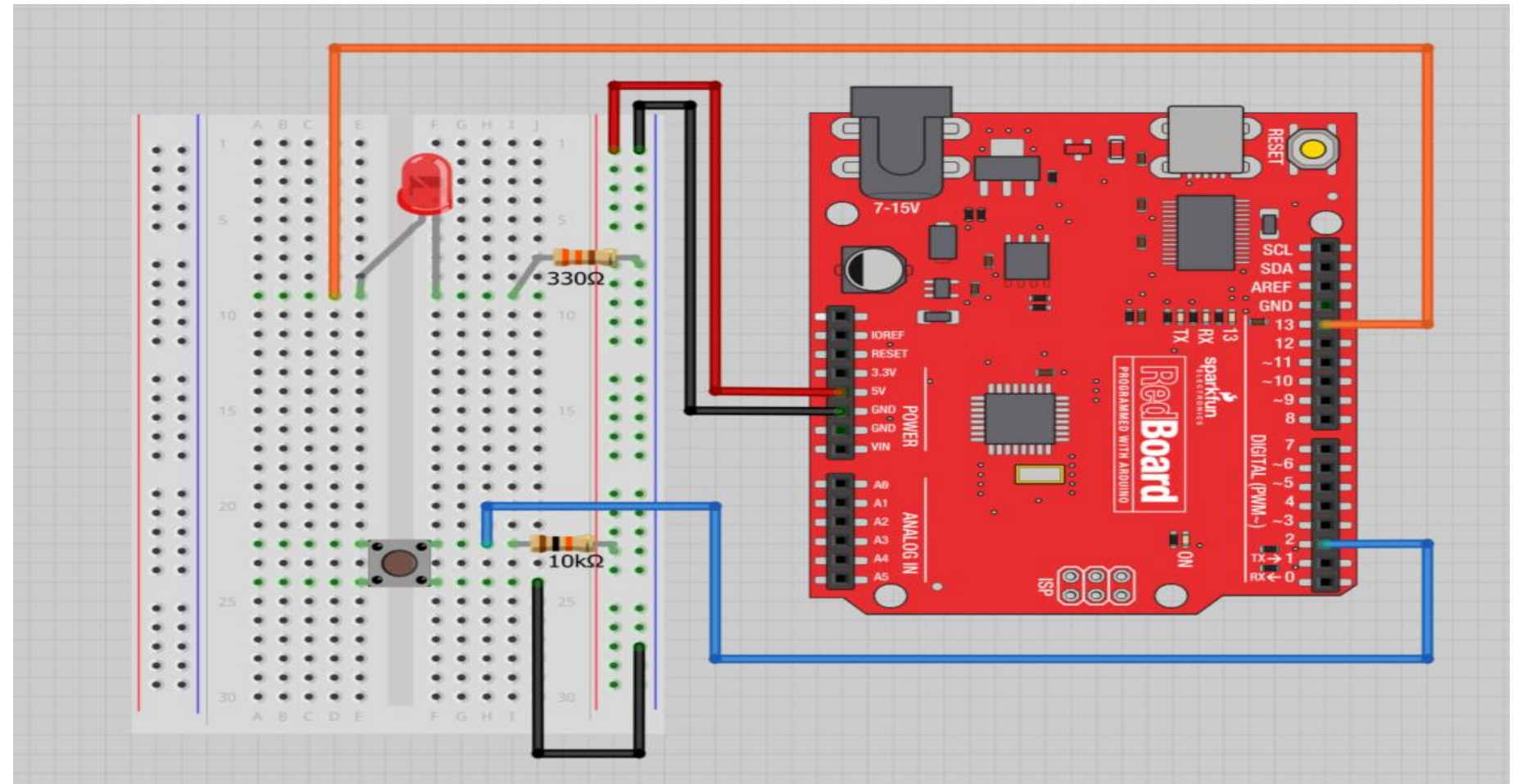
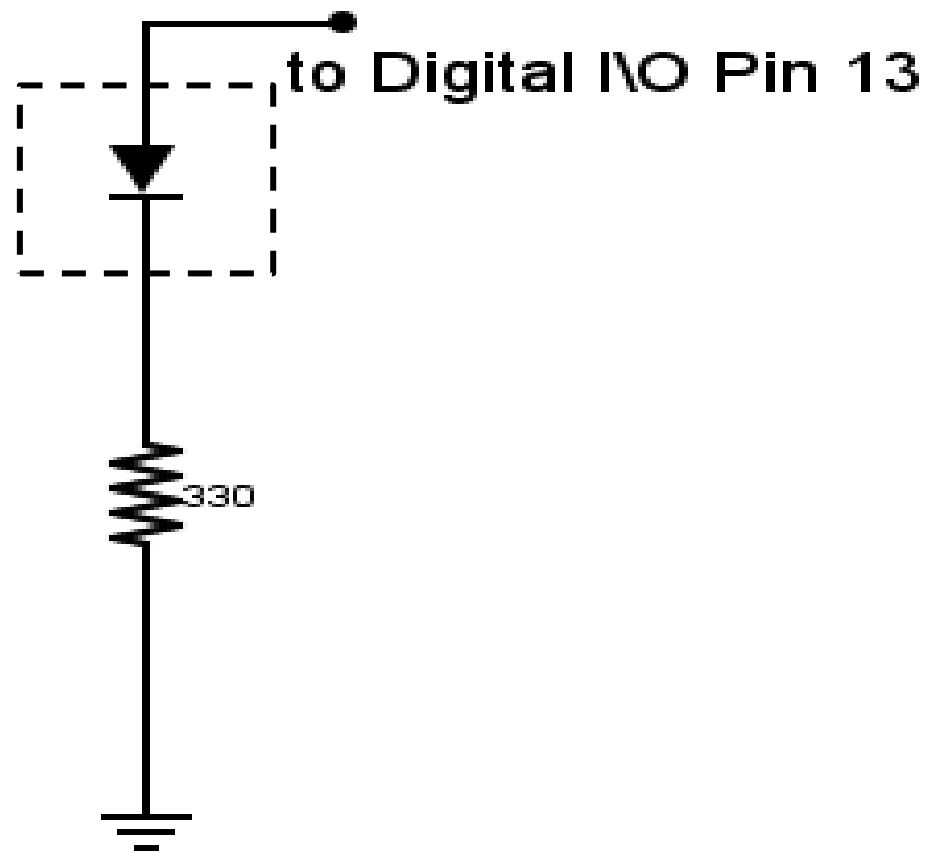
- **Challenge 2a** – Change the rate of the fading in and out. There are at least two different ways to do this – can you figure them out?
- **Challenge 2b** – Use 2 (or more) LEDs – so that one fades in as the other one fades out.

# Digital Sensors (Switches) Pull-up Resistor [\(circuit\)](#)



# Digital Sensors (Switches)

## Add an indicator LED to Pin 13



**“Competitors” to the Arduino**

# “Competitors” to the Arduino

- **PIC controller**
  - Microcontroller programmed with C or assembler
- **Alternatives to the Arduino line**
  - Pinguino – PIC controller
  - MSP430 – Texas Instruments; \$4.30
  - Others: customs, Teensy, etc.
- **Netduino**
- **Computers**
  - Raspberry Pi
  - BeagleBones – TI; has computer and controller

# Netduino

- Microcontroller and development tools created by Microsoft to work with the .NET Micro Framework.
- VASTLY better development environment.
  - [visualmicro.com](http://visualmicro.com)
  - Other alternatives
- Differences
  - Pins on a Netduino are 3.3V (not 5).
  - Netduinos have a much faster processor.
  - 60K of RAM (versus an Uno's 2K).
- Largely compatible with the Arduino, but it is not a drop-in replacement (can fry it).

# Raspberry Pi

- **Low end computer, not a controller**
- **Uses Debian Linux**
  - Arch Linux ARM, Fedora, FreeBSD, Slackware...
- **Programmed with Python**
  - BBC BASIC, C, Perl
- **As it is a computer and not a controller, its role in these projects is different.**
- **Hierarchy: computers control controllers, controllers control hardware.**



# Usage Notes

- A lot of slides are adopted from the presentations and documents published on internet by experts who know the subject very well.
- I would like to thank who prepared slides and documents.
- Also, these slides are made publicly available on the web for anyone to use
- If you choose to use them, I ask that you alert me of any mistakes which were made and allow me the option of incorporating such changes (with an acknowledgment) in my set of slides.

Sincerely,

Dr. Cahit Karakuş

**cahitkarakus@gmail.com**