

ARDUINO - FAST PROTOTYPING

Speaker - Ali Asgar I. Tashrifwala

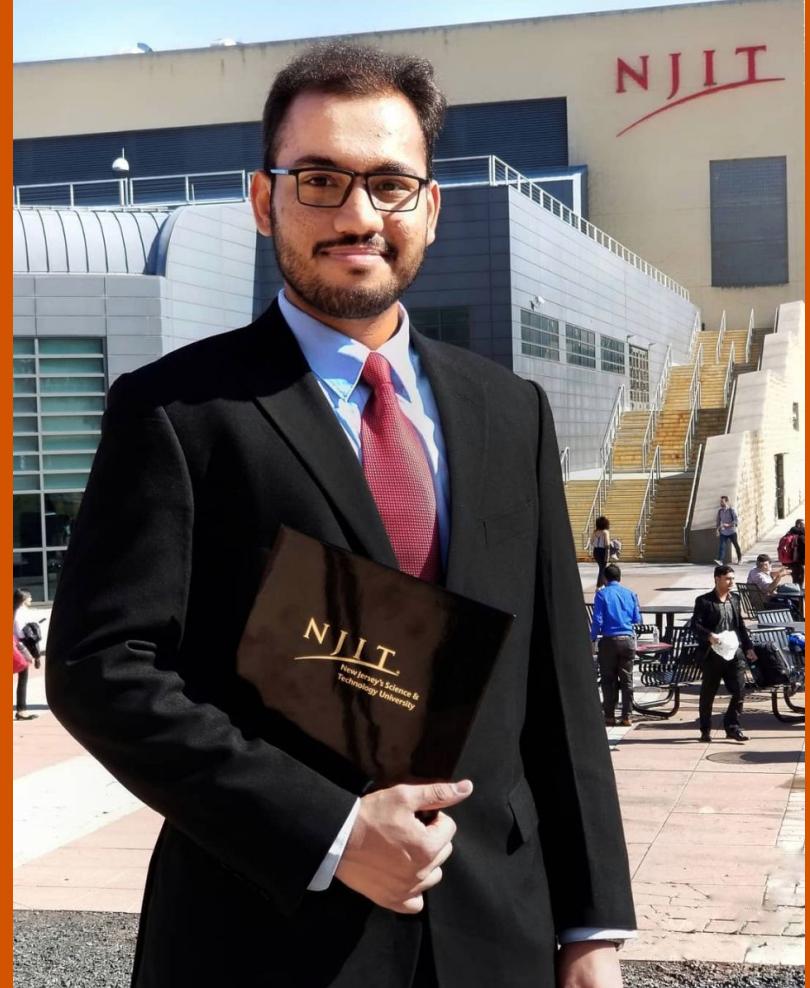
ARDUINO
DAY 2019

March 16TH
day.arduino.cc
#ArduinoD19

Ali Asgar I. Tashrifwala

About me:

- Research Assistant in Robotics & Data Laboratory (RADLab)
- Event Coordinator, NJIT IEEE Student Branch
- VP of Education, University Heights Toastmasters
- Multiple Hackathon Winner!



Previous Arduino Day Events: 2015, 2016, and 2017

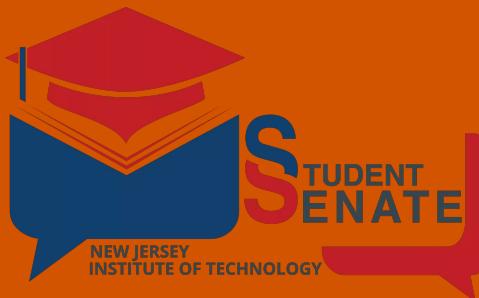
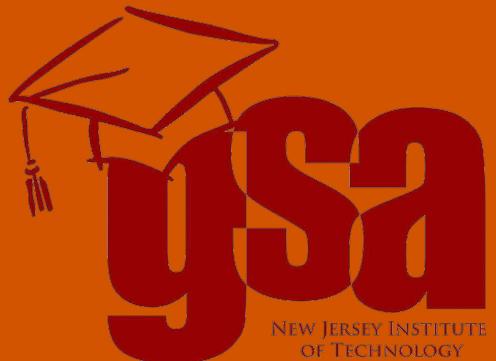




ORGANIZERS:

ARDUINO
DAY 2019

IEEE North Jersey Section



IEEE



Installation

- Go to - <https://www.arduino.cc/en/Main/Software>

Download the Arduino IDE

The screenshot shows the Arduino IDE download page. On the left, there's a large teal circular logo with a white infinity symbol containing a minus sign on the left and a plus sign on the right. To the right of the logo, the text "ARDUINO 1.8.8" is displayed in bold capital letters. Below this, a paragraph of text describes the software: "The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions." On the far right, there are download links for different operating systems: "Windows Installer, for Windows XP and up" (with a "Windows ZIP file for non admin install" link below it), "Windows app" (with a "Get" button), "Mac OS X 10.8 Mountain Lion or newer", "Linux 32 bits", "Linux 64 bits", and "Linux ARM". At the bottom, there are links for "Release Notes", "Source Code", and "Checksums (sha512)".

- Reference - Installing Arduino IDE - Sparkfun
<https://learn.sparkfun.com/tutorials/installing-arduino-ide/all>

What is Embedded System?

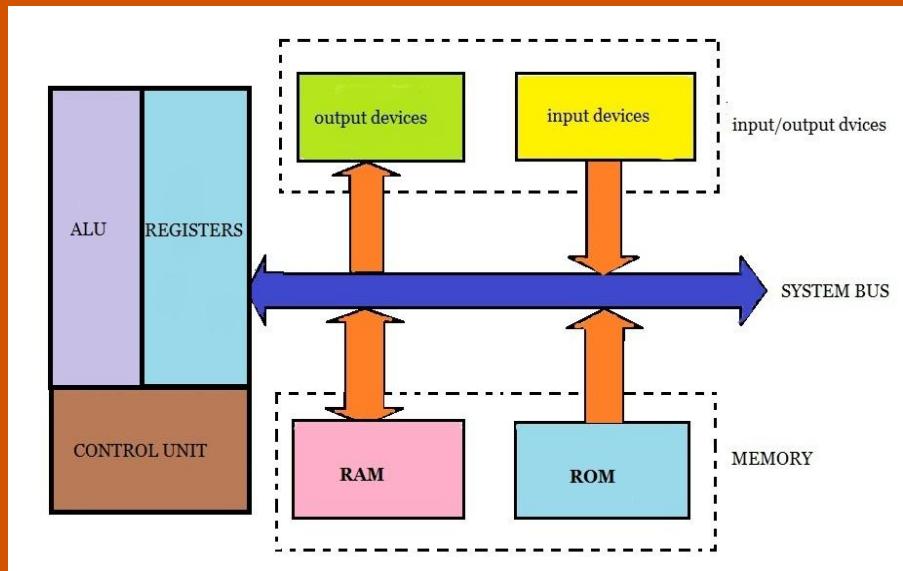
Embedded means "hidden inside". A branch of electronic systems where you see components (hardware) outside, but those components are controlled by set of instructions (software), which is hidden inside the components is called embedded systems.

SCOPE!



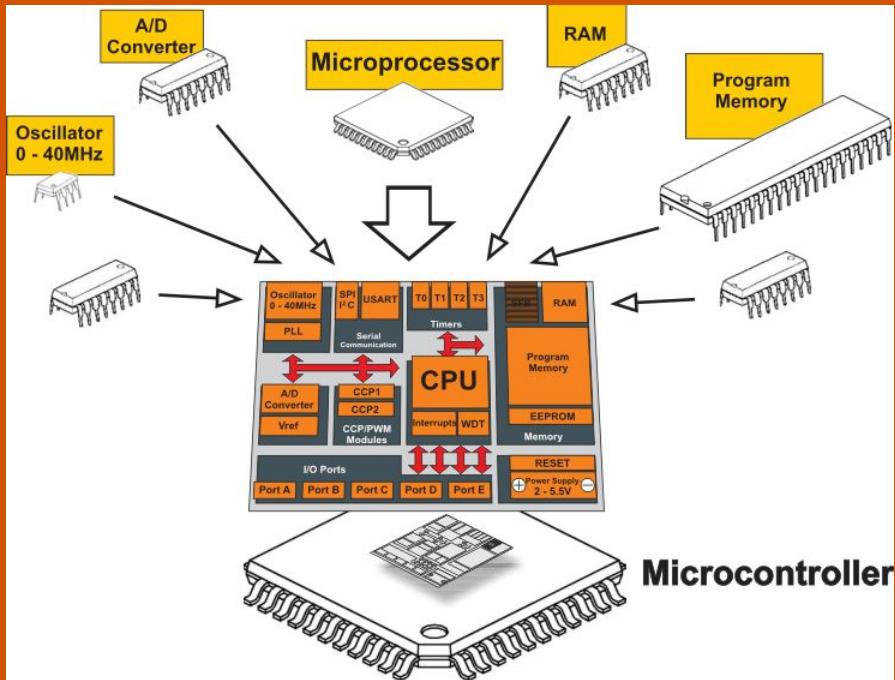
Microprocessors

8085 & 8086



Microcontrollers

8051



Assembly Language V/S Embedded C

```
1 CSEG  
2 PUBLIC main_  
3 PUBLIC puts_  
4 DSEG  
0000 48656C6C6F20576F 5 _3 DB 'Hello World!',0  
6 CSEG  
7  
0000 55 8 main_: PUSH BP  
0001 8BEC 9 MOV BP,SP  
0003 B80000 10 MOV AX,OFFSET _3  
0006 50 11 PUSH AX  
0007 E80000 12 CALL puts_  
000A 8BE5 13 MOV SP,BP  
000C 5D 14 POP BP  
000D C3 15 RET  
16 END
```

V/S

Link: <http://codepad.org/QOmLUb6y> [raw code | output | fork]

C, pasted 1 second ago:

```
1 #include <stdio.h>  
2  
3 int main()  
4 {  
5     printf("Hello World");  
6     return 0;  
7 }
```

Output:

```
1 Hello World
```

Overview

What is Arduino ?

- What is it used for?
- How to get started
- Demonstration

Questions are welcome at any time.

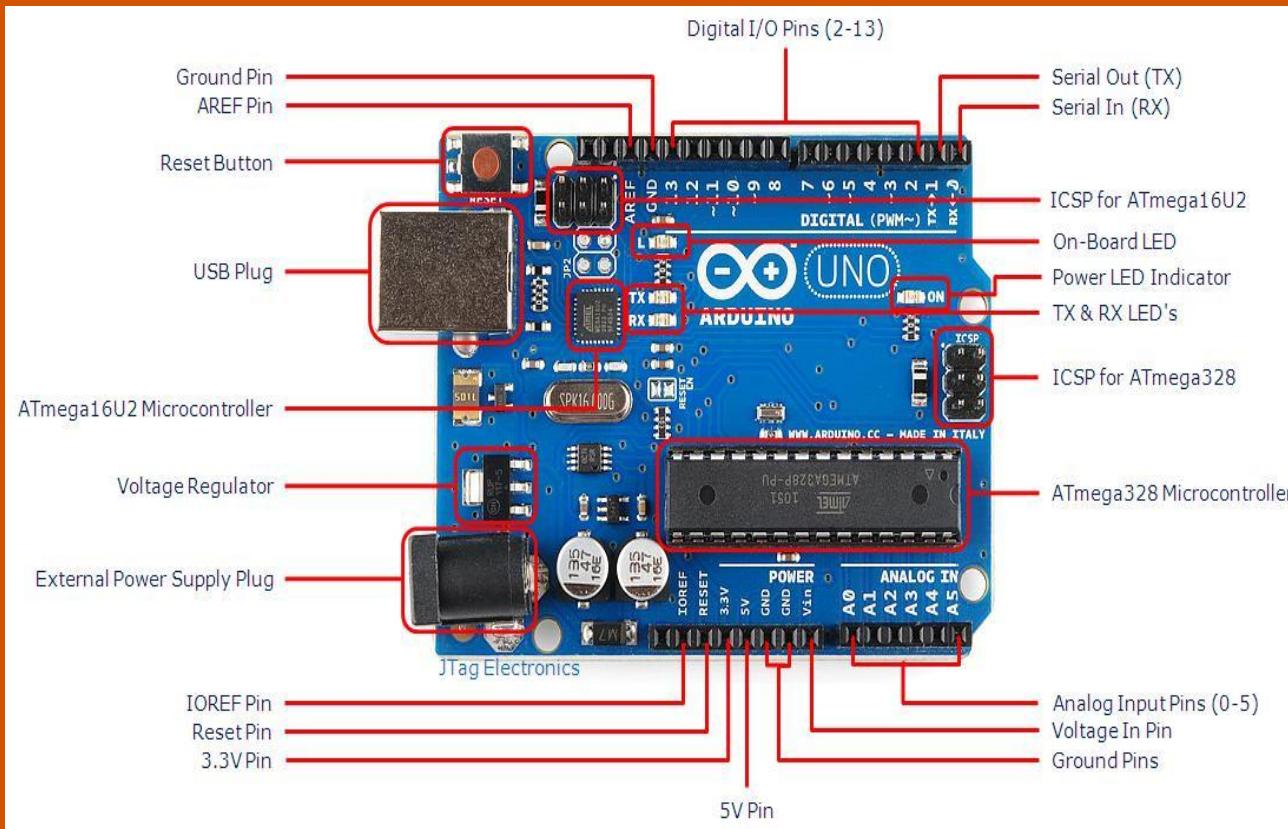
What is Arduino?

- *“Arduino is an open-source physical computing platform based on a simple i/o board and a development environment that implements the Processing / Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer.”* (www.arduino.cc, 2006)



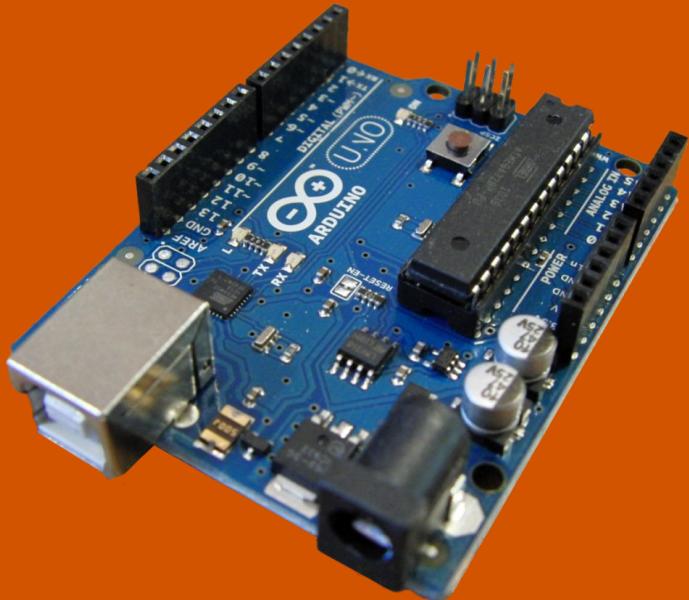
FAST PROTOTYPING

Meet Arduino Uno R3



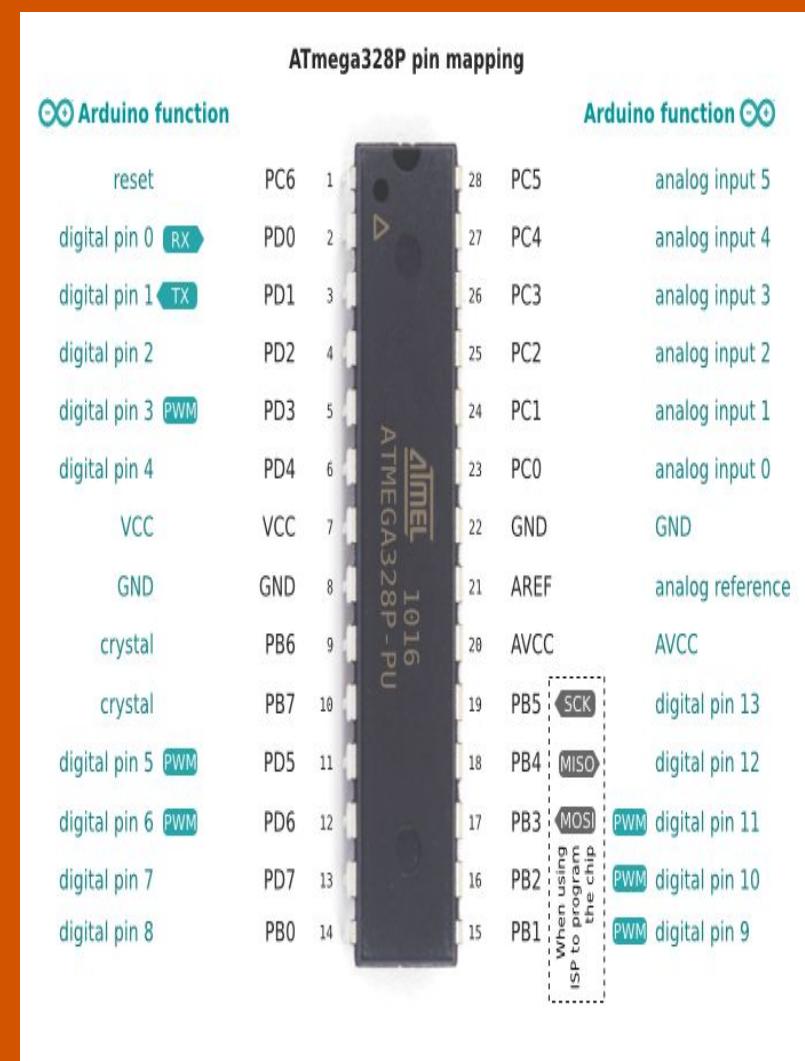
About Arduino Uno

- ATmega328 Microcontroller
- USB Plug & External Power Supply Plug
- Power Pins
- Input and Output Pins
- LED Indicators
- ICSP Headers
- ATmega16U2 Microcontroller Voltage regulator



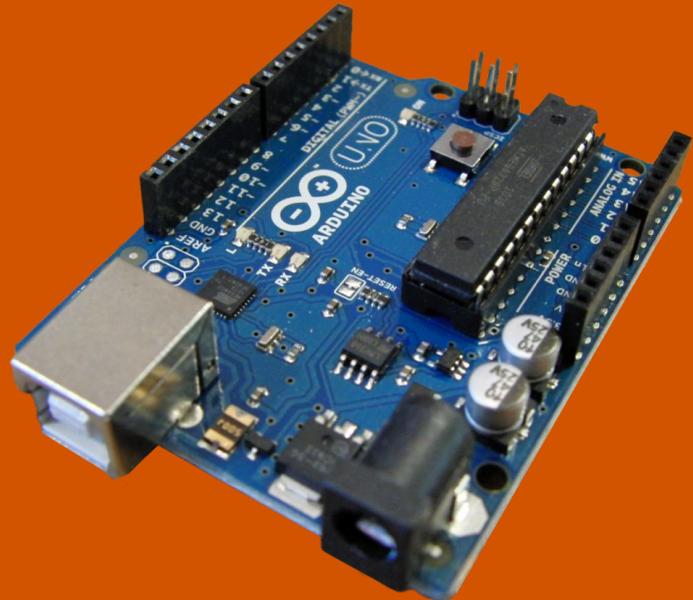
About Arduino Uno

- Operating Voltage 5V
- Input Voltage 7-12V
- Digital I/O Pins 14
- PWM output 6
- Analog Input Pins 6
- Flash Memory 32 KB
- SRAM 2 KB
- EEPROM 1 KB
- Clock Speed 16 MHz



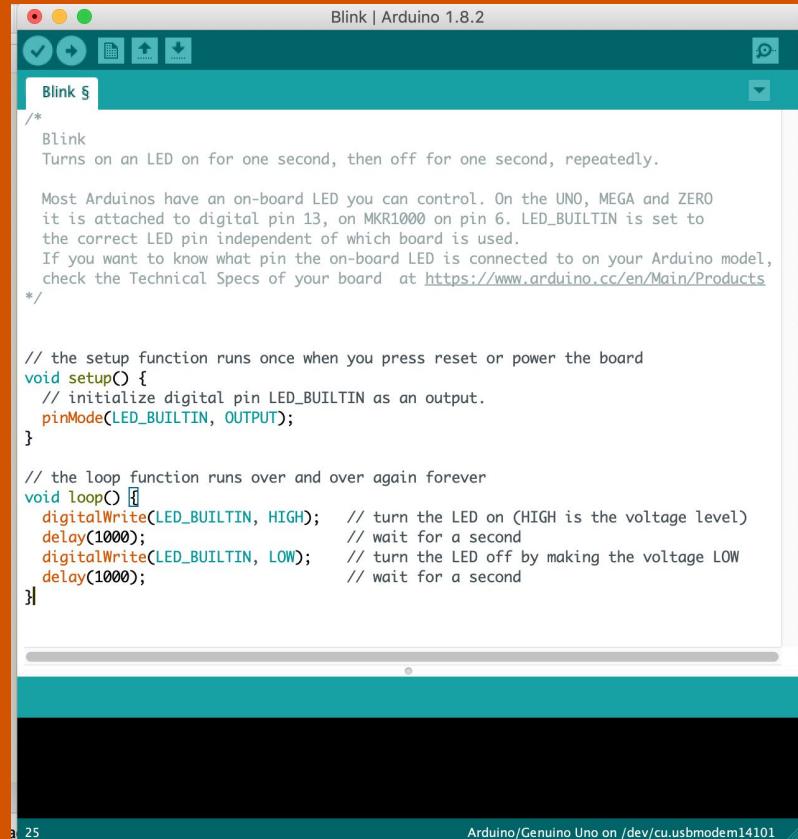
The “Power” of Arduino is in:

- Standard board design
- Wiring language
- Open Source



Arduino is a platform

- Also including an Integrated Development Environment (IDE) for programming.
- The language itself is based in C but is largely modeled upon the www.processing.org language.



The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1.8.2". The code editor displays the "Blink" sketch. The code is as follows:

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
 * it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
 * the correct LED pin independent of which board is used.
 * If you want to know what pin the on-board LED is connected to on your Arduino model,
 * check the Technical Specs of your board at https://www.arduino.cc/en/Main/Products
 */

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

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

The bottom of the screen shows a black window representing the Arduino Uno board.

What is it used for?

- Physical Computing projects / research
- Interactive Installations
- Rapid prototyping
- When you wish to move beyond the traditional Mouse, Keyboard and Monitor to develop novel and custom interactions in your project work.

What can it do?

Sensors (to sense stuff)

- Push buttons, touch pads, tilt switches.
- Variable resistors (eg. volume knob / sliders)
- Photoresistors (sensing light levels)
- Thermistors (temperature)
- Ultrasound (proximity range finder)

Actuators (to do stuff)

- Lights, LED's
- Motors
- Speakers
- Displays (LCD)

Why Arduino?

- It is Open Source, both in terms of Hardware and Software.
- It is cheap, the hardware can be built from components or a prefab board can be purchased for approx \$25.
- It can communicate with a computer via serial connection over USB.
- It can be powered from USB or standalone DC power.

Why Arduino?

- It can run standalone from a computer (chip is programmable) and it has memory (a small amount).
- It can work with both Digital and Analogue electronic signals. Sensors and Actuators.
- You can make cool stuff! Some people are even making simple robots, and we all know robots are just cool. ☺

How to get started

- You'll need a board of course, Arduino UNO R3 , along with the USB cable and DC power supplies.
- Read about, understand what you are working with and download the IDE: <http://www.arduino.cc>
- Mac, Windows and Penguin friendly versions available
- Then you are ready to plug it in!

Not so fast!

- It's important to note at this stage that Arduino's are electronic devices.
- This means you MUST consider electrical safety and understand the basics before diving straight in.
- The board itself doesn't operate at what would normally be considered dangerous Voltages or Amperage, but if in doubt at any stage of use you should seek more expert advice.

Basic Electrical knowledge

- A fantastic guide to electronics in theory, practice and of course safety is available as a PDF at:

<http://www.ibiblio.org/obp/electricCircuits/>

- What you want is Volume 1, DC circuits. This will help you greatly in understanding how to wire circuits when using sensors and actuators.

Getting up and running

- The power mode must be selected before you plug the board into anything.
- When powering from the USB cable (5 volts) the jumper should be closest to the USB input, for DC supply the jumper should be closest to the DC input.

Installation

- Go to - <https://www.arduino.cc/en/Main/Software>

Download the Arduino IDE

The screenshot shows the Arduino IDE download page. On the left, there's a large teal circular logo with a white infinity symbol containing a minus sign on the left and a plus sign on the right. To the right of the logo, the text "ARDUINO 1.8.8" is displayed in bold capital letters. Below this, a paragraph of text describes the software: "The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions." On the far right, there's a teal sidebar with download links:

- Windows** Installer, for Windows XP and up
Windows ZIP file for non admin install
- Windows app** Requires Win 8.1 or 10
[Get](#)
- Mac OS X** 10.8 Mountain Lion or newer
- Linux** 32 bits
- Linux** 64 bits
- Linux** ARM

At the bottom of the sidebar, there are links to "Release Notes", "Source Code", and "Checksums (sha512)".

- Reference - Installing Arduino IDE - Sparkfun
<https://learn.sparkfun.com/tutorials/installing-arduino-ide/all>

Board Type

Sketch Tools Help 

Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⌘M
Serial Plotter ⌘L
WiFi101 Firmware Updater

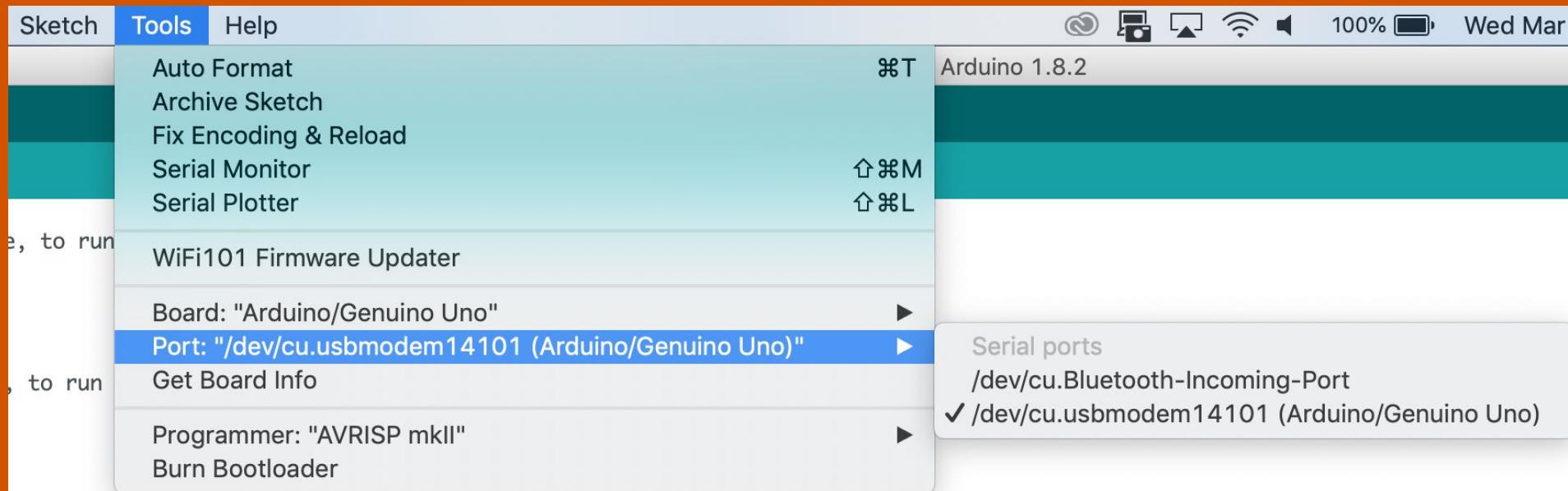
Board: "Arduino/Genuino Uno" ▶ **Arduino/Genuino Uno**

Port ▶
Get Board Info

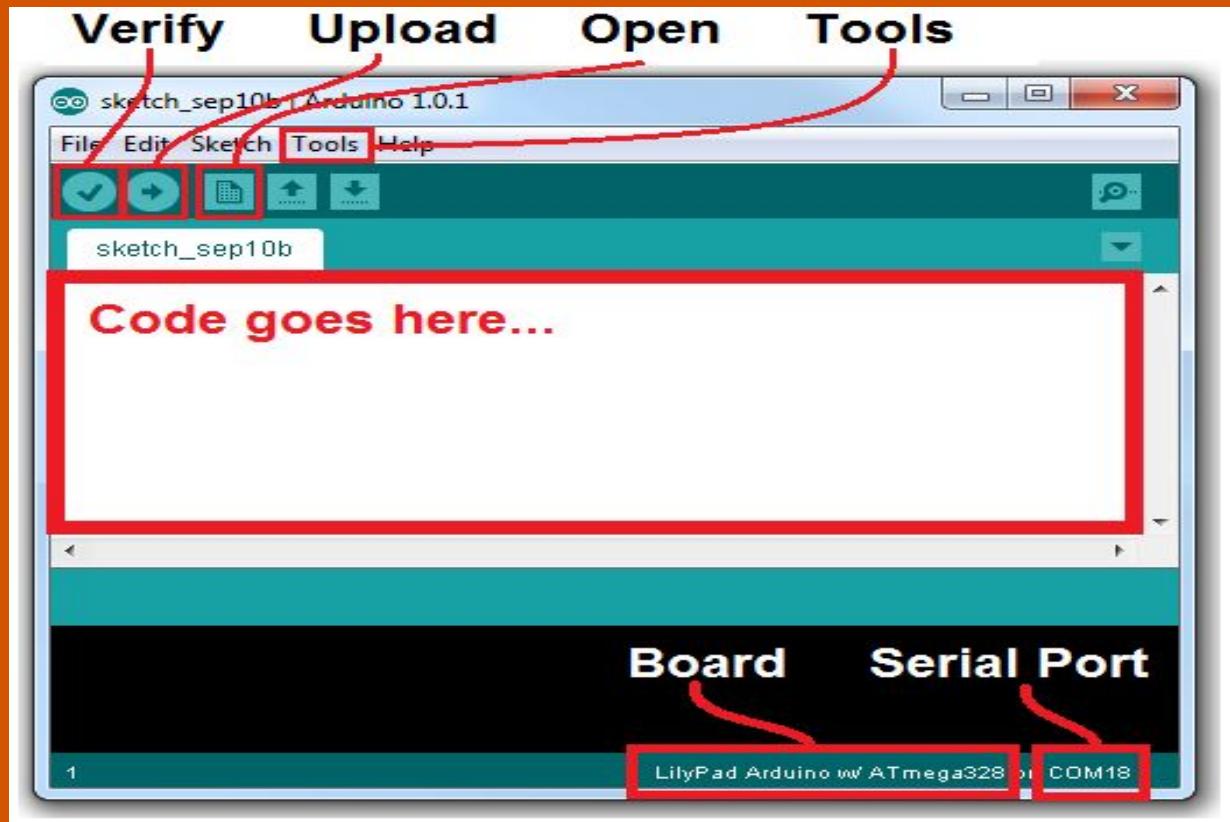
Programmer: "AVRISP mkII" ▶
Burn Bootloader

Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora

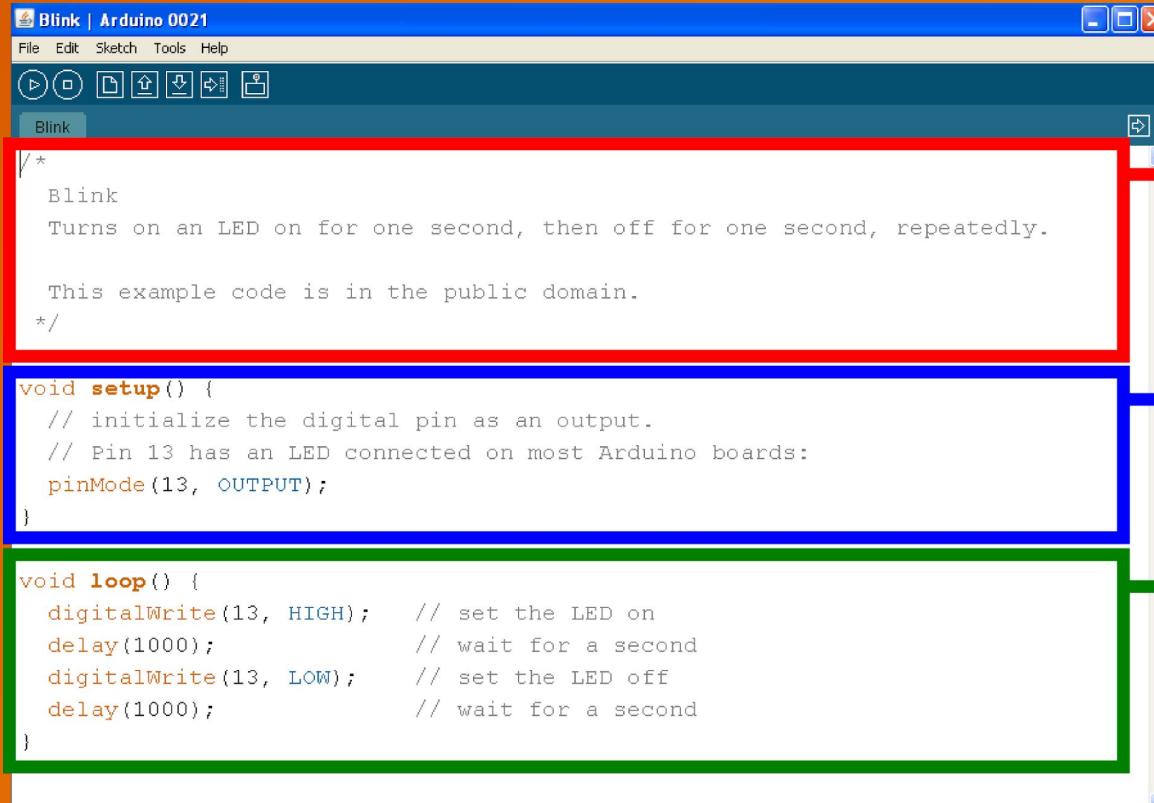
Serial Port / COM Port



The Environment



Parts of the Sketch



The image shows the Arduino IDE interface with a sketch titled "Blink". The code is divided into three main sections: comments/explanation, setup, and loop.

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */

void setup() {
    // initialize the digital pin as an output.
    // Pin 13 has an LED connected on most Arduino boards:
    pinMode(13, OUTPUT);
}

void loop() {
    digitalWrite(13, HIGH);      // set the LED on
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // set the LED off
    delay(1000);                // wait for a second
}
```

Comments /
Explaining
the game

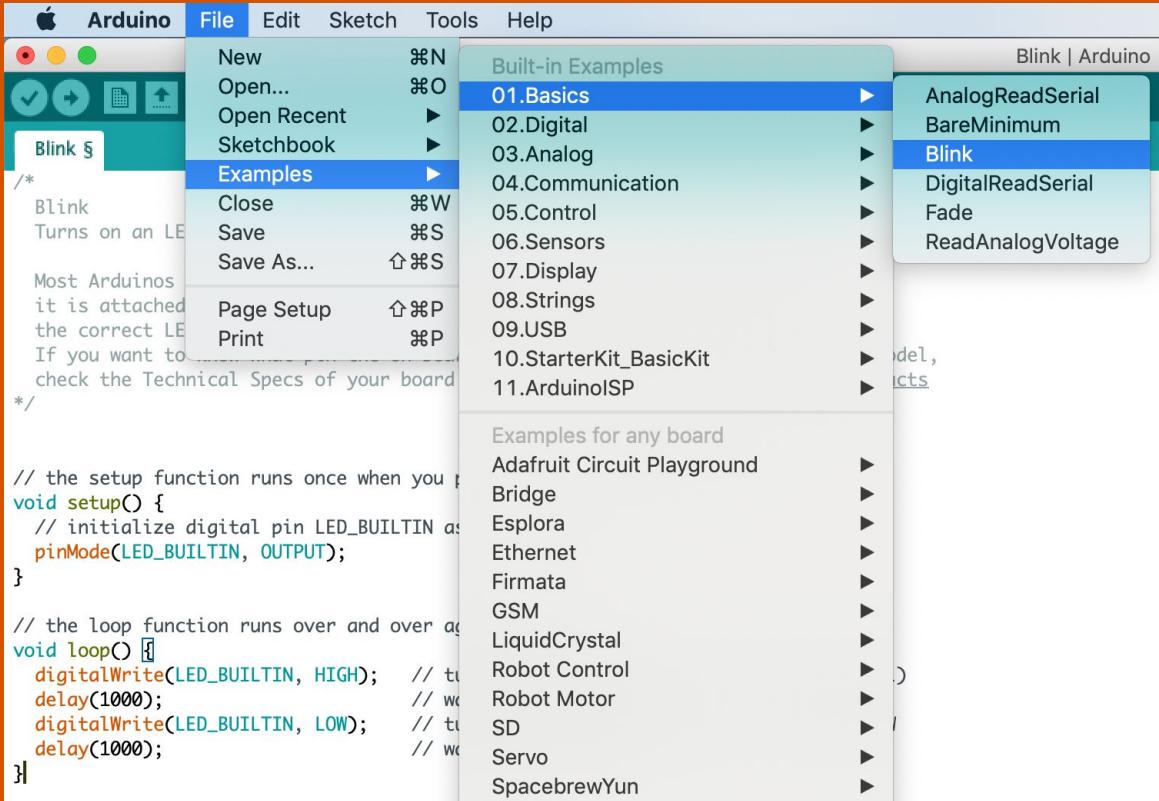
Setup /
Stretching or
tying shoes

Loop /
Playing the
game

Demonstration

- It's time for a simple demonstration.

BLINK PROGRAM



The screenshot shows the Arduino IDE interface on a Mac OS X system. The title bar says "Arduino". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The "File" menu has options like "New", "Open...", "Open Recent", "Sketchbook", "Examples", "Close", "Save", "Save As...", "Page Setup", and "Print". The "Examples" option is currently selected. A sub-menu titled "Built-in Examples" lists examples grouped by category: 01.Basics (Blink, Digital, Analog, Communication, Control, Sensors, Display, Strings, USB), 02.Digital, 03.Analog, 04.Communication, 05.Control, 06.Sensors, 07.Display, 08.Strings, 09.USB, 10.StarterKit_BasicKit, and 11.ArduinoISP. The "Blink" example is highlighted. To the right of the main menu, there is a vertical sidebar with links to other examples: AnalogReadSerial, BareMinimum, Blink (which is also highlighted in blue), DigitalReadSerial, Fade, and ReadAnalogVoltage. The main code editor window displays the "Blink" sketch:

```
/*
 * Blink
 * Turns on an LED
 * Most Arduinos
 * it is attached
 * to the correct LED
 * If you want to
 * check the Technical Specs of your board
 */

// the setup function runs once when you press
// the reset button or power the board
void setup() {
    // initialize digital pin LED_BUILTIN as
    // an output
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
    delay(1000);                      // wait for a second
    digitalWrite(LED_BUILTIN, LOW);     // turn the LED off (LOW is the ground level)
    delay(1000);                      // wait for a second
}
```

Variable Scope

Where you declare your variables matters



```
Blink $  
/*  
Blink  
Turns on an LED on for one second, then off for one second, repeatedly.  
  
This example code is in the public domain.  
*/  
const int variable1 = 1;  
int variable2 = 2;  
  
void setup() {  
    int variable3 = 3;  
  
    // initialize the digital pin as an output  
    // Pin 13 has an LED connected on most Arduino Boards  
    pinMode(13, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(13, HIGH); // set the LED on  
    delay(1000); // wait for a second  
    digitalWrite(13, LOW); // set the LED off  
    delay(1000); // wait for a second  
}
```

Constant / Read only
Variable available
anywhere
Variable available only
in this function,
between curly brackets

Setup void setup() {}

- The **setup** function comes before the **loop** function and is necessary for all Arduino sketches

```
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}
```

```
Setup void setup ()  
{ pinMode (13, OUTPUT); }
```

**Outputs are declare in setup, this is done by using the
pinMode function**

- This particular example declares digital pin # 13 as an output, remember to use CAPS

```
void setup() {  
  // initialize the digital pin as an output.  
  // pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}
```

Setup void setup () { Serial.begin();}

Serial communication also begins in setup

- This particular example declares Serial communication at a baud rate of 9600. More on Serial later...

```
void setup() {
    // initialize the digital pin as an output.
    // Pin 13 has an LED connected on most Arduino boards:
    pinMode(13, OUTPUT);
    Serial.begin(9600);
}
```

Setup, Internal Pullup Resistors

void setup () { digitalWrite (12, HIGH); }

You can also create internal pullup resistors in setup, to do so
digitalWrite the pin HIGH

- This takes the place of the pullup resistors currently on your circuit 7 buttons

```
void setup() {
    // initialize the digital pin as an output.
    // Pin 13 has an LED connected on most Arduino boards:
    pinMode(13, OUTPUT);
    Serial.begin(9600);
    digitalWrite(12, HIGH);
}
```

Conditional Statements

if (this is true) { do this; }

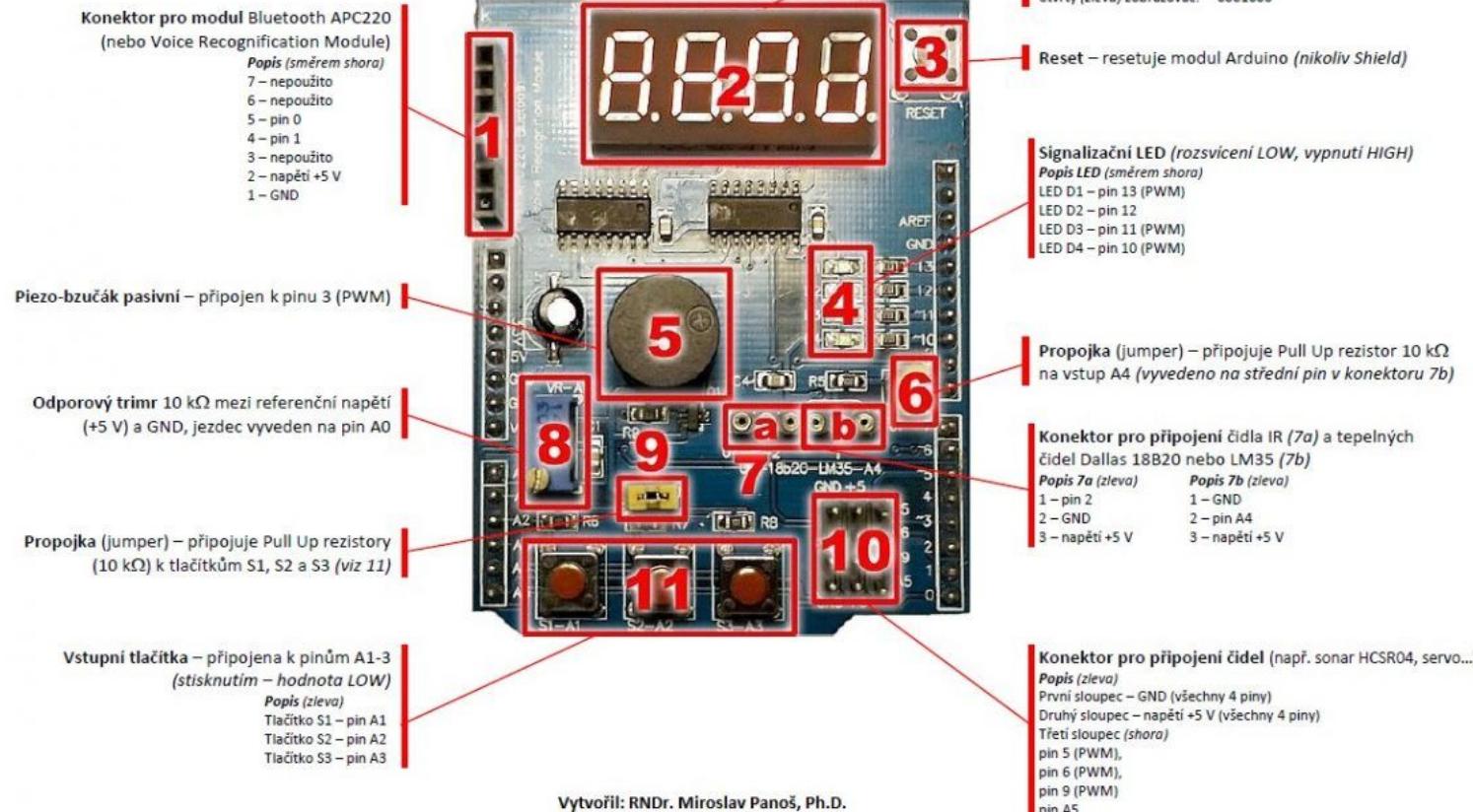
```
void loop() {
    // read the state of the pushbutton value:
    buttonState = digitalRead(buttonPin);

    // check if the pushbutton is pressed:
    // if it is, the buttonState is HIGH:
    if (buttonState == HIGH) {
        // turn LED on:
        digitalWrite(ledPin, HIGH);
    }
    else {
        // turn LED off:
        digitalWrite(ledPin, LOW);
    }
}
```

Conditional inside
parenthesis,
uses ==, <=, >= or !
you can also nest
using && or ||

Arduino Multi-function Shield

(Funduino výukový multi-shield do škol pro Arduino)



Comments

- Comments can be anywhere
Comments created with // or /* and */
Comments do not affect code

Operators

The equals sign

- `=` is used to assign a value
- `==` is used to compare values
- And & Or
- `&&` is “and”
- `||` is “or”

Declaring Variables

- Boolean: **boolean variableName;**
- Integer: **int variableName;**
- Character: **char variableName;**
- String: **stringName [];**

Assigning Variables

- Boolean: **variableName = true;**
- or **variableName = false;**

- Integer: **variableName = 32767;**
- or **variableName = -32768;**

- Character: **variableName = 'A';**
- or **stringName = "BLAH!!"**

Basic Repetition

Loop

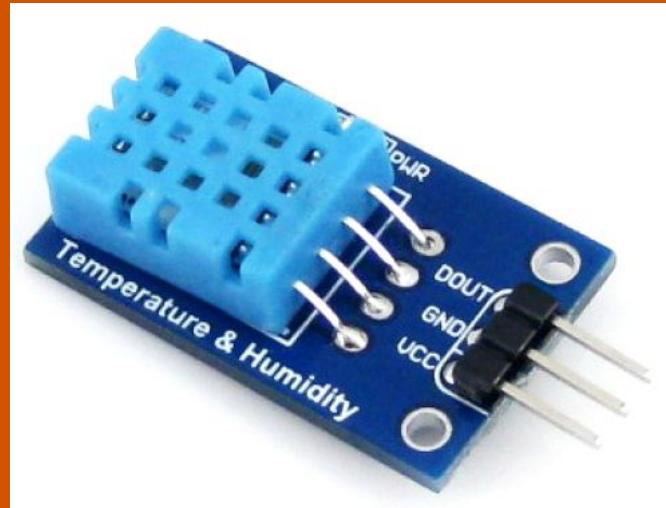
For

While

Temperature & Humidity Sensor (DHT11)

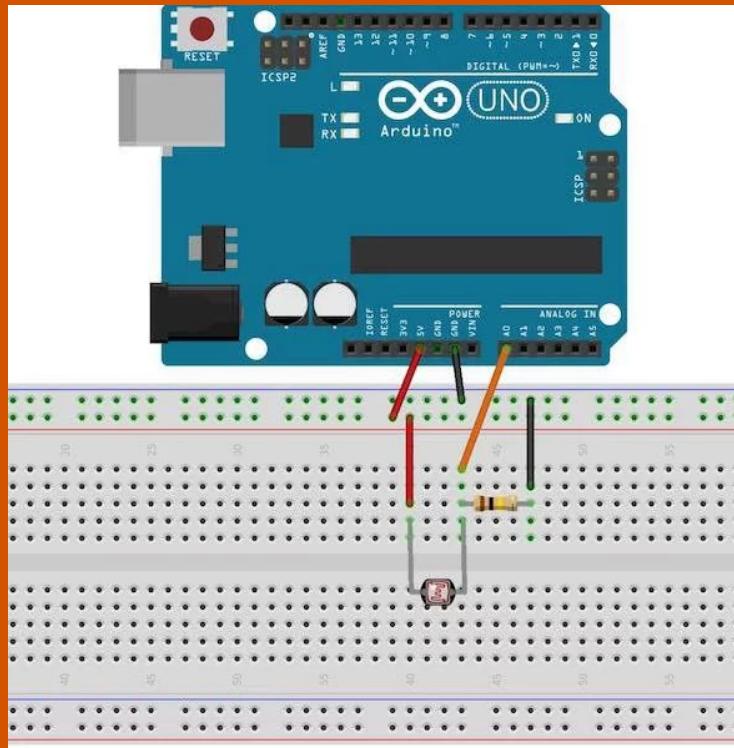
Specifications:

- Temperature
 - Resolution : 1°C
 - Accuracy : ±2°C
 - Measuring range : 0°C ~ 50°C
- Humidity
 - Resolution : 1%RH
 - Accuracy : ±5%RH (0~50°C)
 - Measuring range : 20%RH ~ 90%RH
(25°C)
- Operating voltage : 3.3V ~ 5.5 V
- Recommended storage condition
 - Temperature : 10°C ~40°C
 - Humidity : 60%RH or below



Light Dependent Resistor (LDR)

High Resistance Photons reduce the resistance.



Buzzer

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short).

Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

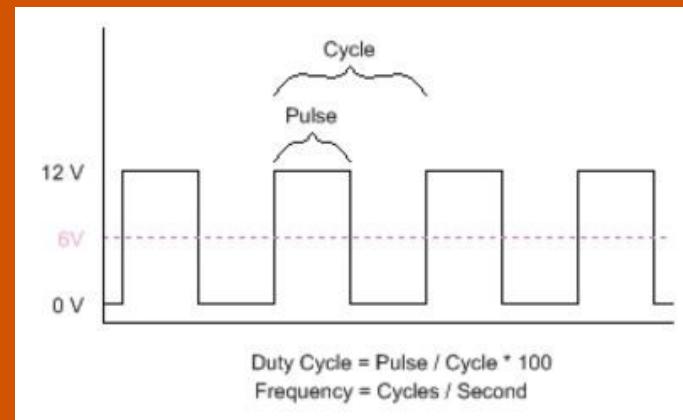


PWM

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means.

The duration of "on time" is called the pulse width.

To get varying analog values, you change, or modulate, that pulse width. controlling the brightness of the LED.



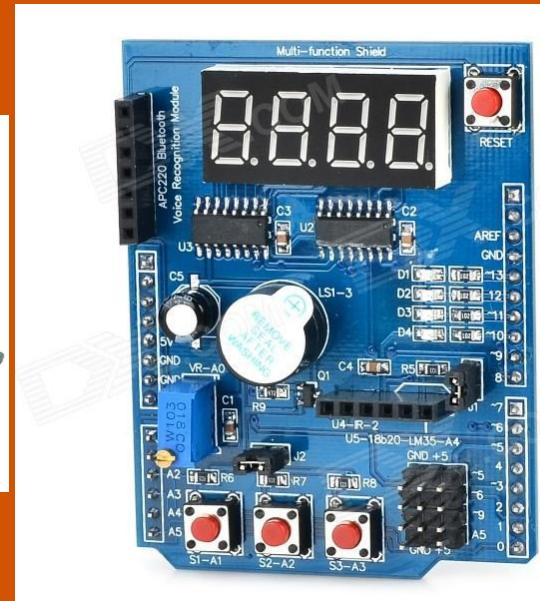
Ultrasonic Sensor (HC-SR04)

This is the HC-SR04 ultrasonic distance sensor. This economical sensor provides 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm.

Each HC-SR04 module includes an ultrasonic transmitter, a receiver and a control circuit.



Project - Car Parking Assistant



Internet of Things IoT

[Channels](#)[Apps](#)[Community](#)[Support ▾](#)[Commercial Use](#)[How to Buy](#)[Sign In](#)[Sign Up](#)

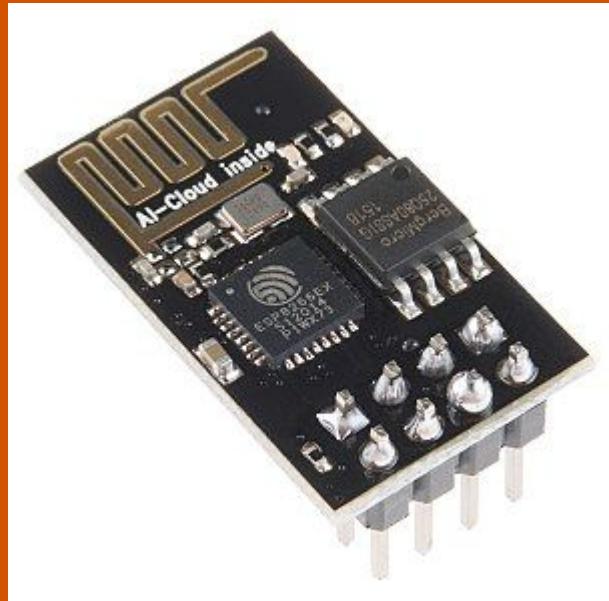
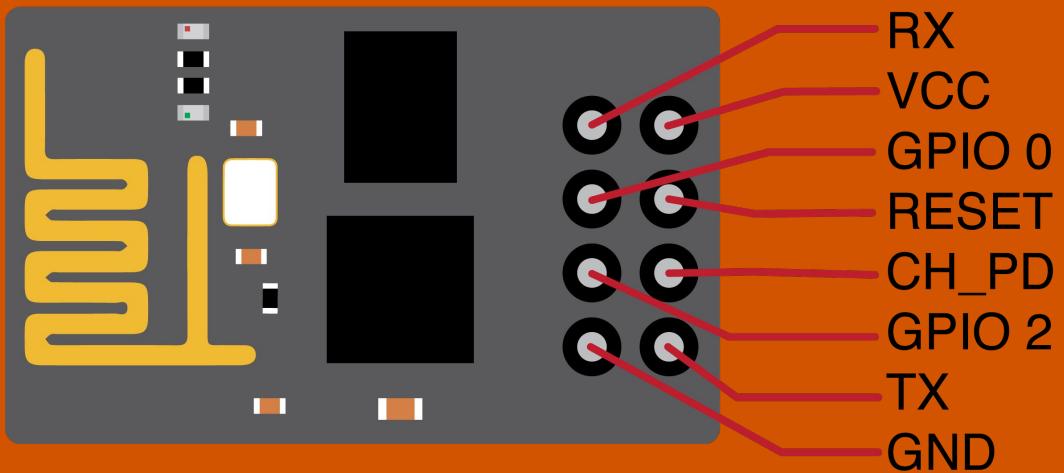
Understand Your Things

The open IoT platform with MATLAB analytics.

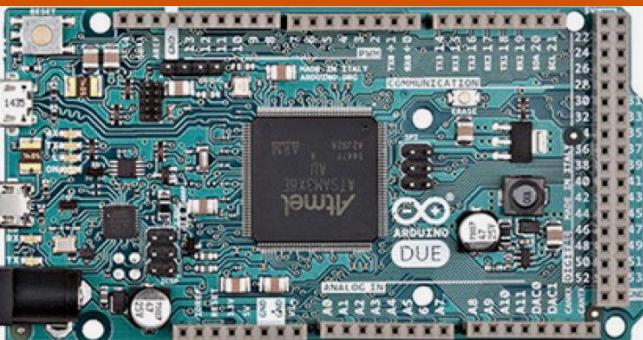
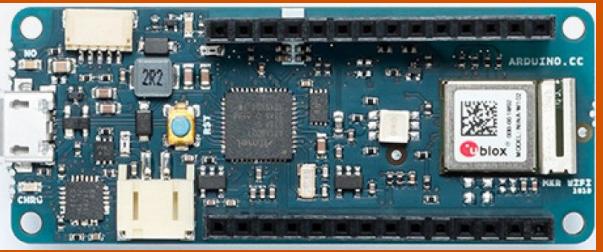
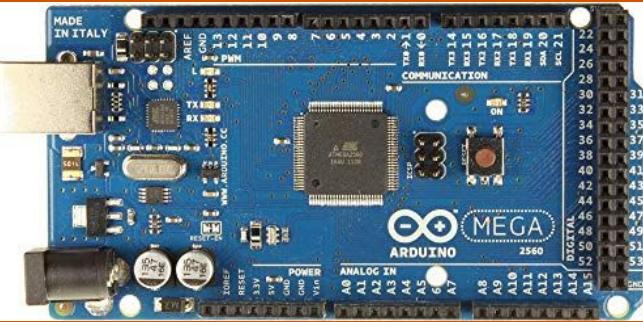
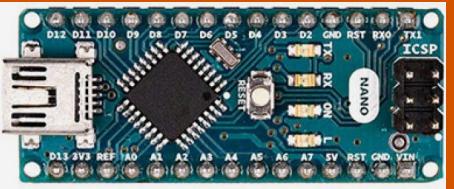
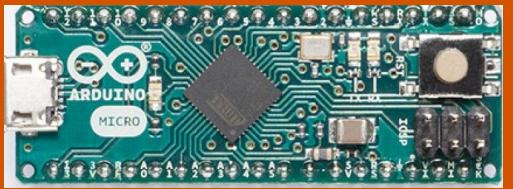
[Get Started For Free](#)[Learn More](#)

Internet of Things IoT

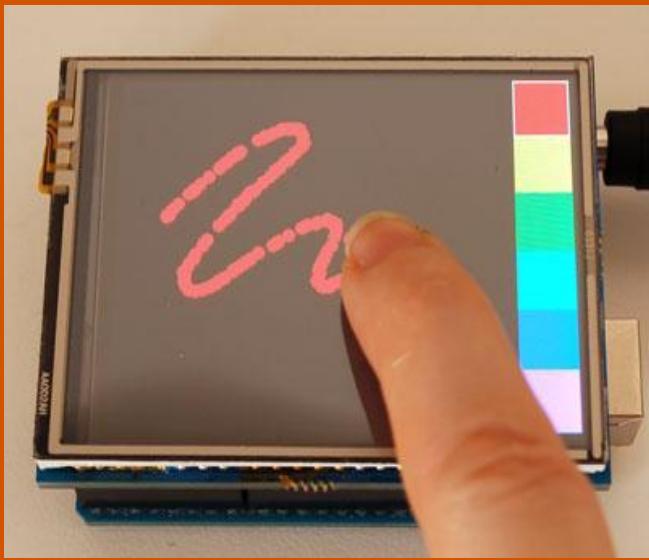
The ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability.



Arduino I/O Boards



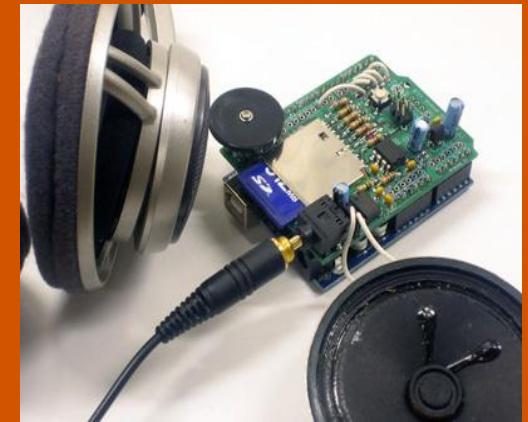
Shields



Touchscreen
Shield



Datalogging
Shield

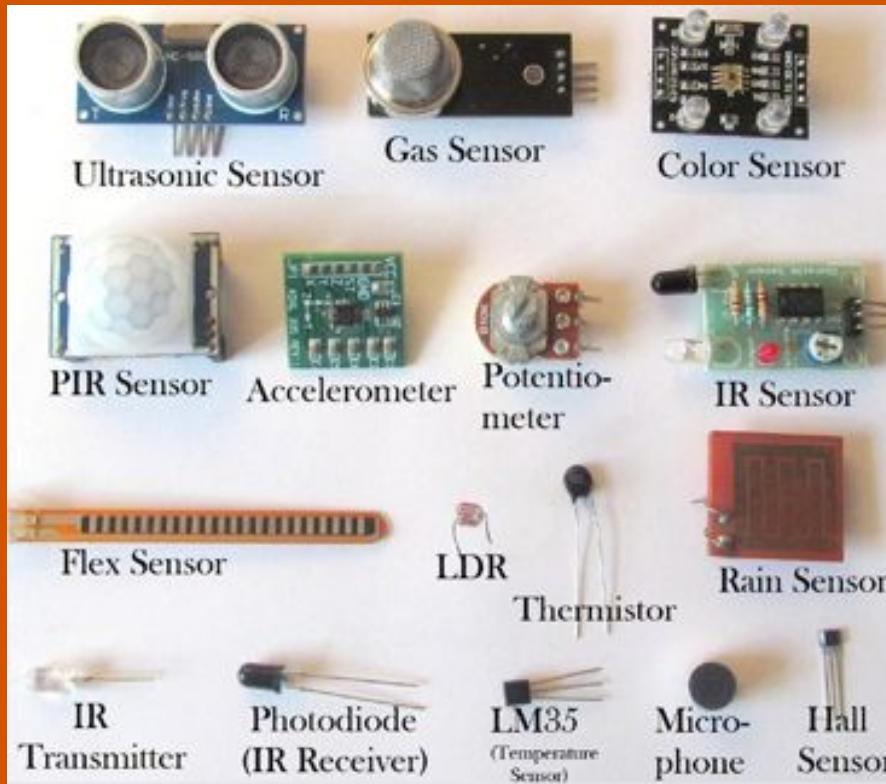


Wave Shield

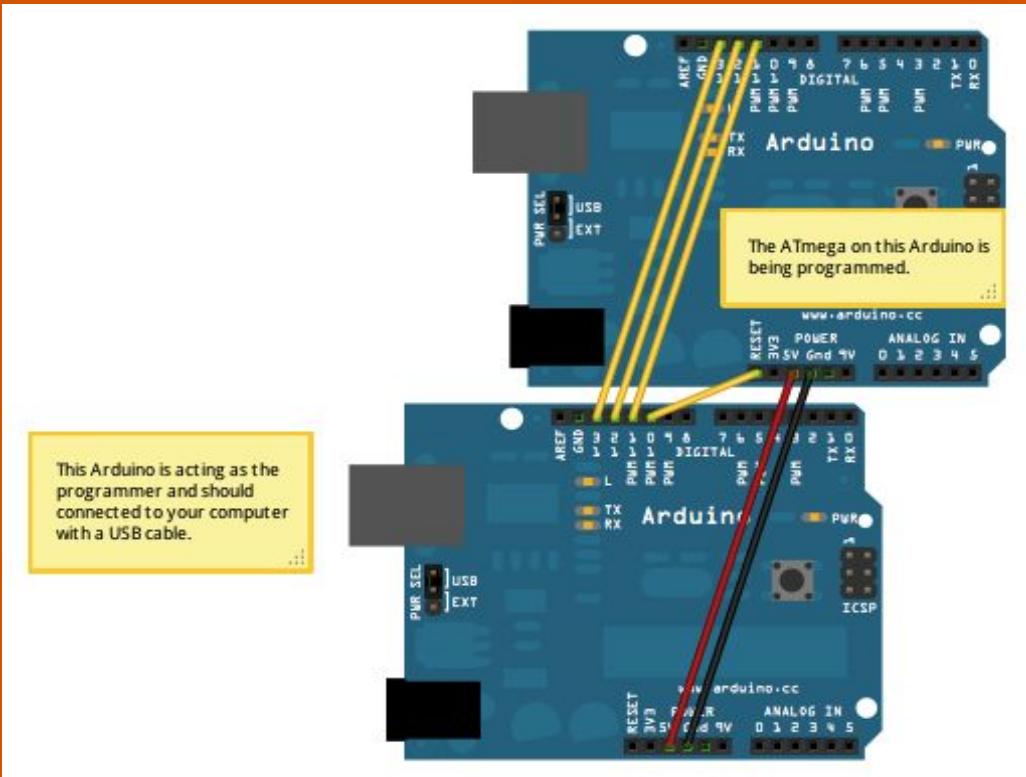
Even more shields!



Sensors



Bootloading



Arduino board to burn a bootloader onto an AVR

- Select the items in the Tools > Board and Serial Port menus that correspond to the board you are using as the programmer (not the board being programmed).
- Upload the ArduinoISP sketch.
- Wire your Arduino board to the target as shown in the diagram below. (Note for the Arduino Uno: you'll need to add a 10 uF capacitor between reset and ground.)

Arduino board to burn a bootloader onto an AVR

- Select the item in the Tools > Board menu that corresponds to the board on which you want to burn the bootloader (not the board that you're using as the programmer).
- Select the Arduino as ISP in the Tools>Programmer menu.
- Use the Burn Bootloader command.

Questions?

Connect with me on -

<https://www.linkedin.com/in/ali-asgar/>

Endorse me for -

- *Arduino*
- *Embedded Systems*
- *Public Speaking*



How good? - Highly skilled

How do you know? - Worked together directly on the same team or project

The screenshot shows a LinkedIn profile for Ali Asgar Tashrifwala. At the top is a circular profile picture of a man with a beard, wearing a blue shirt and tie, looking towards the camera. Behind him are several computer monitors displaying code and data. Below the picture is the name "Ali Asgar Tashrifwala". Underneath the name is a summary: "Research Assistant at RADLab. Embedded Systems | Robotics | Drones | Linux | PCB Design | IoT | C". It also mentions "Greater New York City Area". At the bottom of the profile section are two buttons: "Add profile section ▾" and "More...". To the right of the profile are several social media links and icons: "NJIT" (New Jersey Institute of Technology) with a graduation cap icon, "See contact info" with a person icon, and "See connections (500+)" with a people icon. A small pencil icon is also visible near the top right of the profile area.

Ali Asgar I. Tashrifwala
ai229@njit.edu
[linkedin.com/in/ali-asgar/](https://www.linkedin.com/in/ali-asgar/)

Thank you !

March 16TH
day.arduino.cc
#ArduinoD19

