InnoVenture Electronics Bootcamp 2019

Exploring 'Smart' Electronic Systems

Eugene Ee
National University of Singapore
Engineering Design & Innovation Centre (EDIC)
engewhe@nus.edu.sg

Preliminaries

- *Hi, welcome to the Electronics Bootcamp 2019. Before we get started……*

- *Form a group of 2*

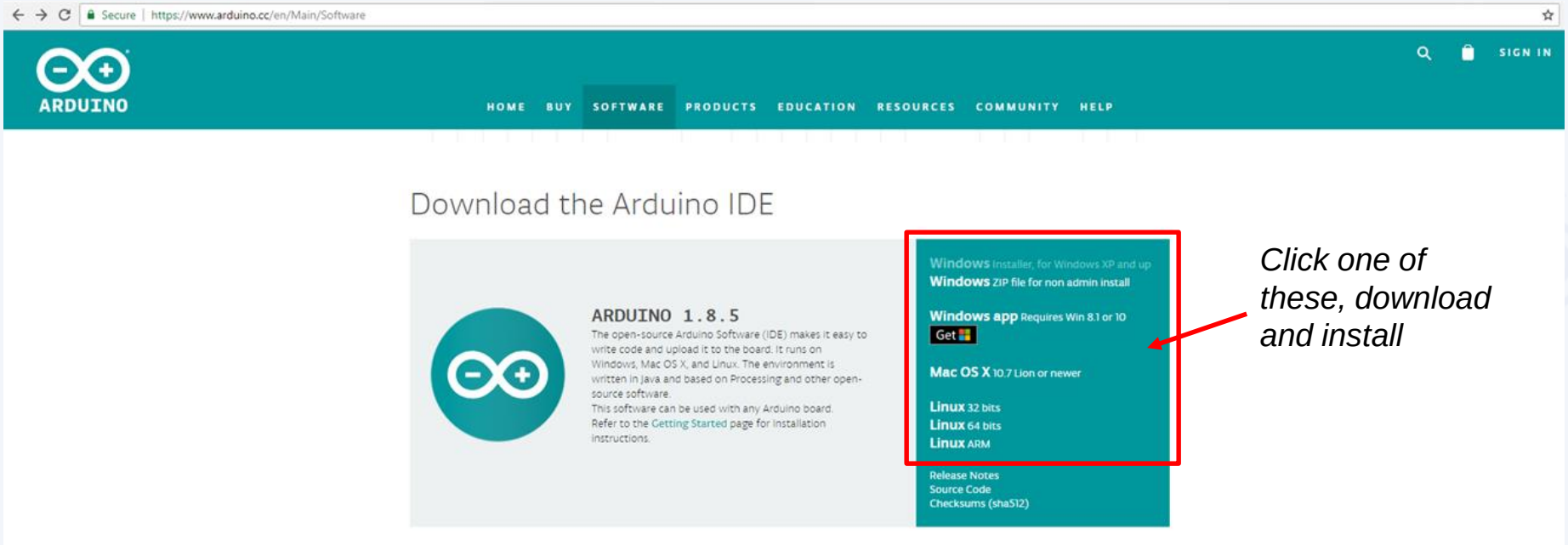
- *Each group should have the following:*
  1. *an Arduino,*
  2. *A breadboard + electronic components + tools,*
  3. *benchtop power supply,*
  4. *a desktop/laptop and*
  5. *some working space around you*

- *Download the Arduino software.*

Preliminaries

- *Download and install Arduino from:*

https://www.arduino.cc/en/Main/Software

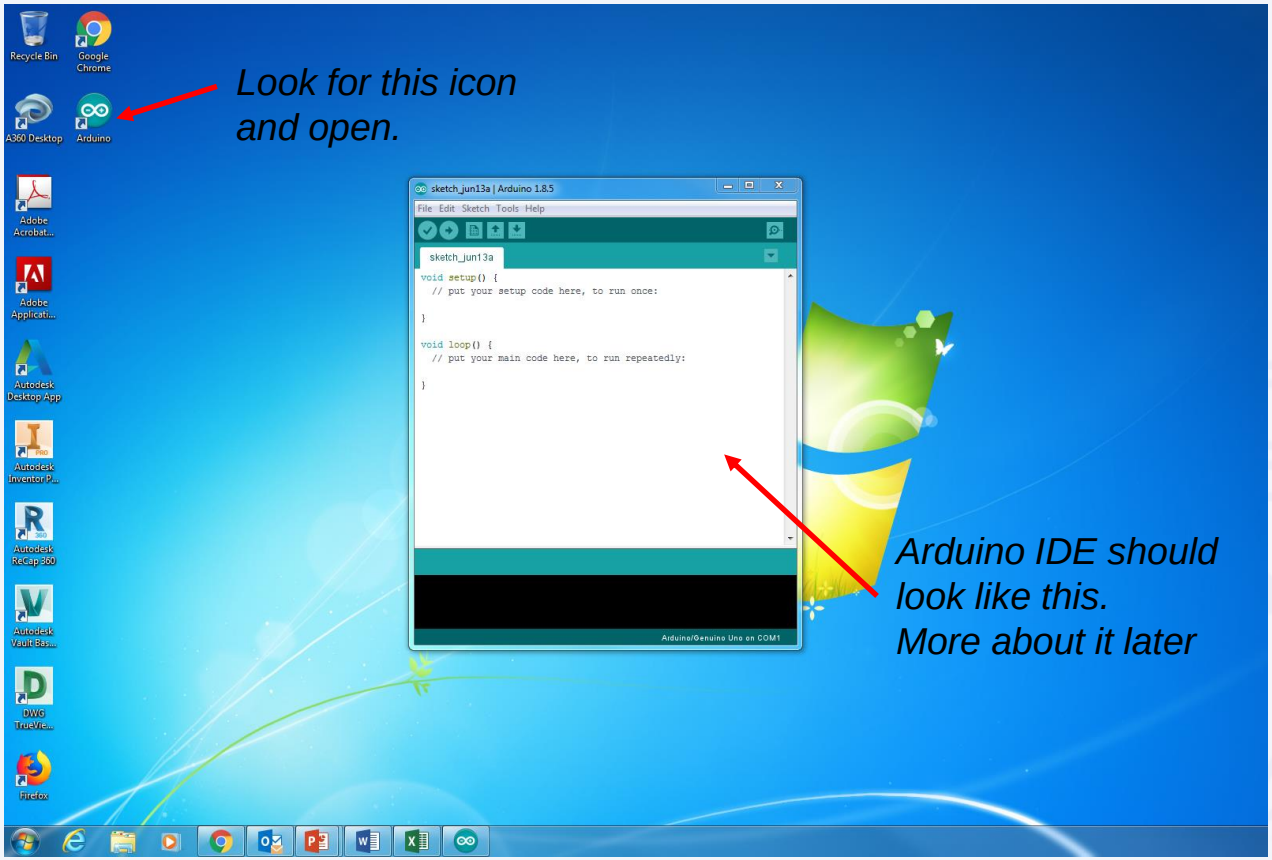


Click one of these, download and install

If you have not done so…

Preliminaries

- *You should see the Arduino IDE on your desktop*



Content

- ***Introduction to Electronics and Arduino***
- ***Basic Electronics***
- ***Sensors & Interfaces***
- ***Integration & Testing***

Introduction to Electronics and Arduino

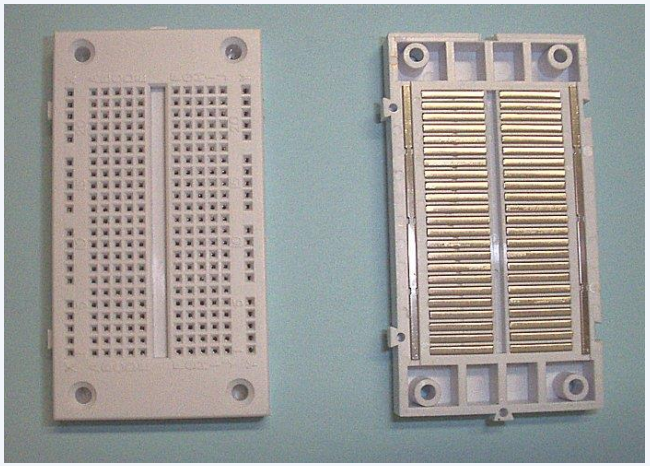
- *This section covers the fundamentals required to:*
  - *understand circuit diagrams*
  - *wire up electronics*
  - *get started with microcontrollers*

- *These are essential to build and design circuits in the later part of this bootcamp.*

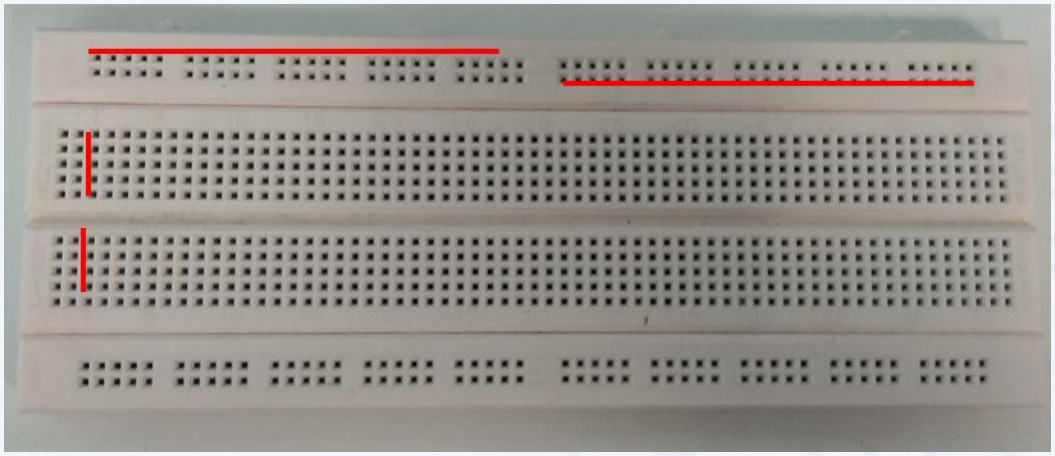
- *Don't hesitate to ask questions if you get stuck along the way.*

Introduction to Electronics and Arduino

- *Setting up the Breadboard*
  - *A breadboard is typically used to prototype first stage electronics.*
  - *It is common to use the side rails as power lines.*



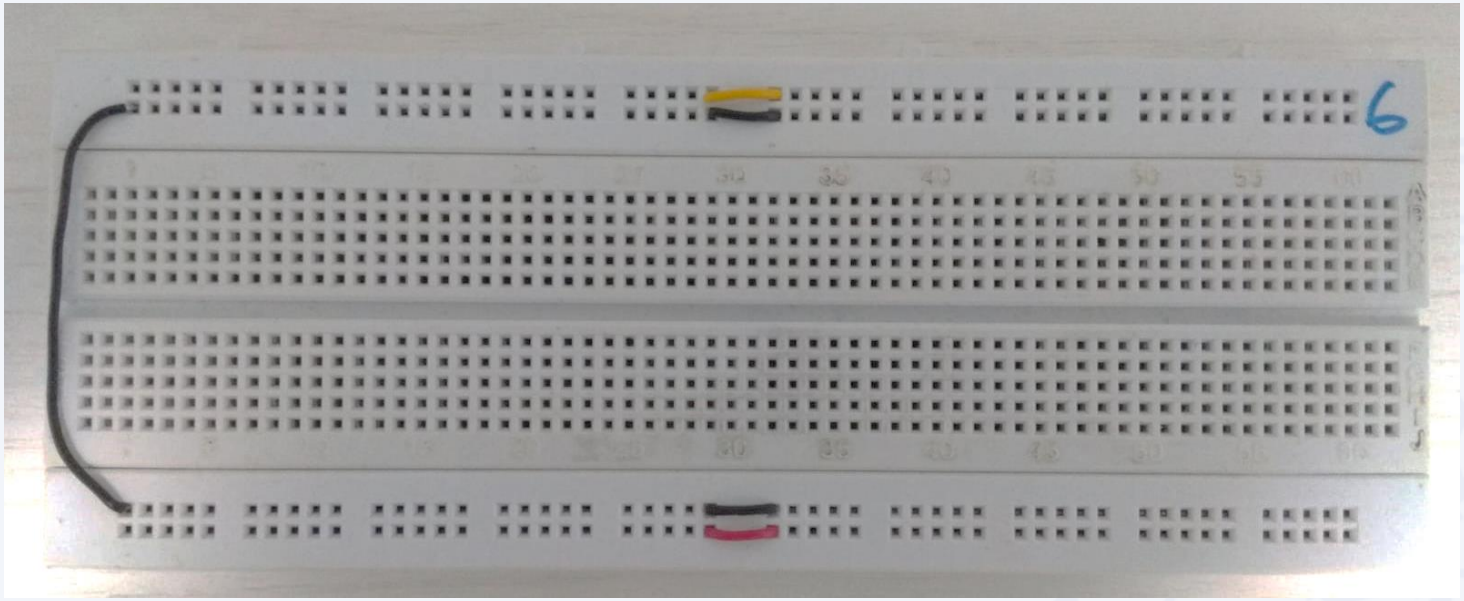
Inside a Breadboard



Breadboard Connections

Introduction to Electronics and Arduino

- *Setting Up the Breadboard*
  - *For this exercise we will need two rails of **common ground**, one **5V power** rail and one **3.3V power** rail.*
  - *These voltage levels **5V** and **3.3V** are common to most electronic systems.*

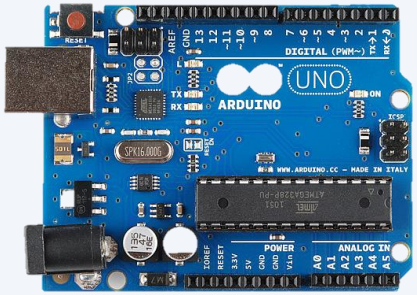


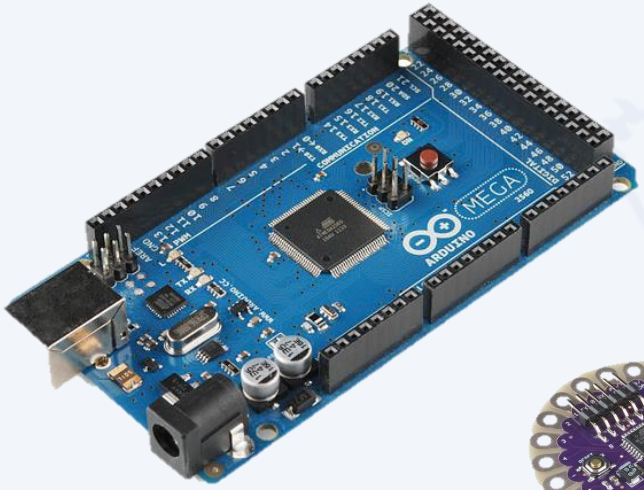
Setting Up a Breadboard

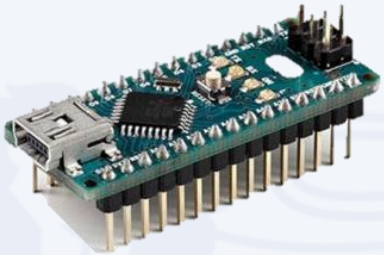
Introduction to Electronics and Arduino

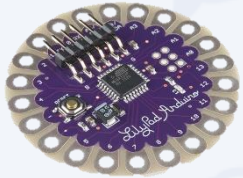
- *An Arduino is…*
  - *Open source electronics platform*
  - *Programmable microcontroller on a circuit board, hardware*
  - *Integrated Development Environment (IDE), software*

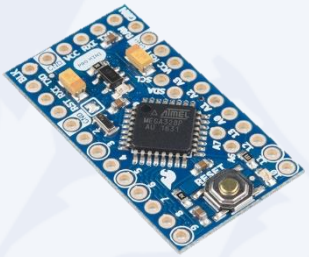
- *Variants:*





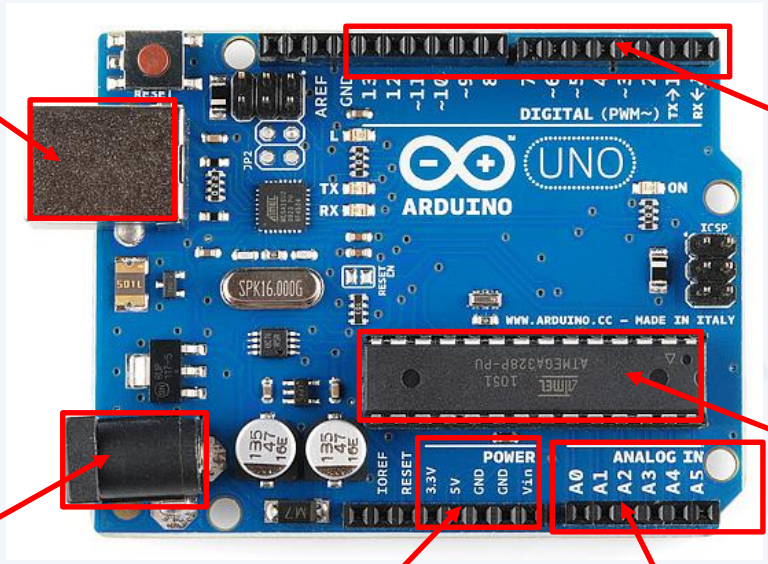






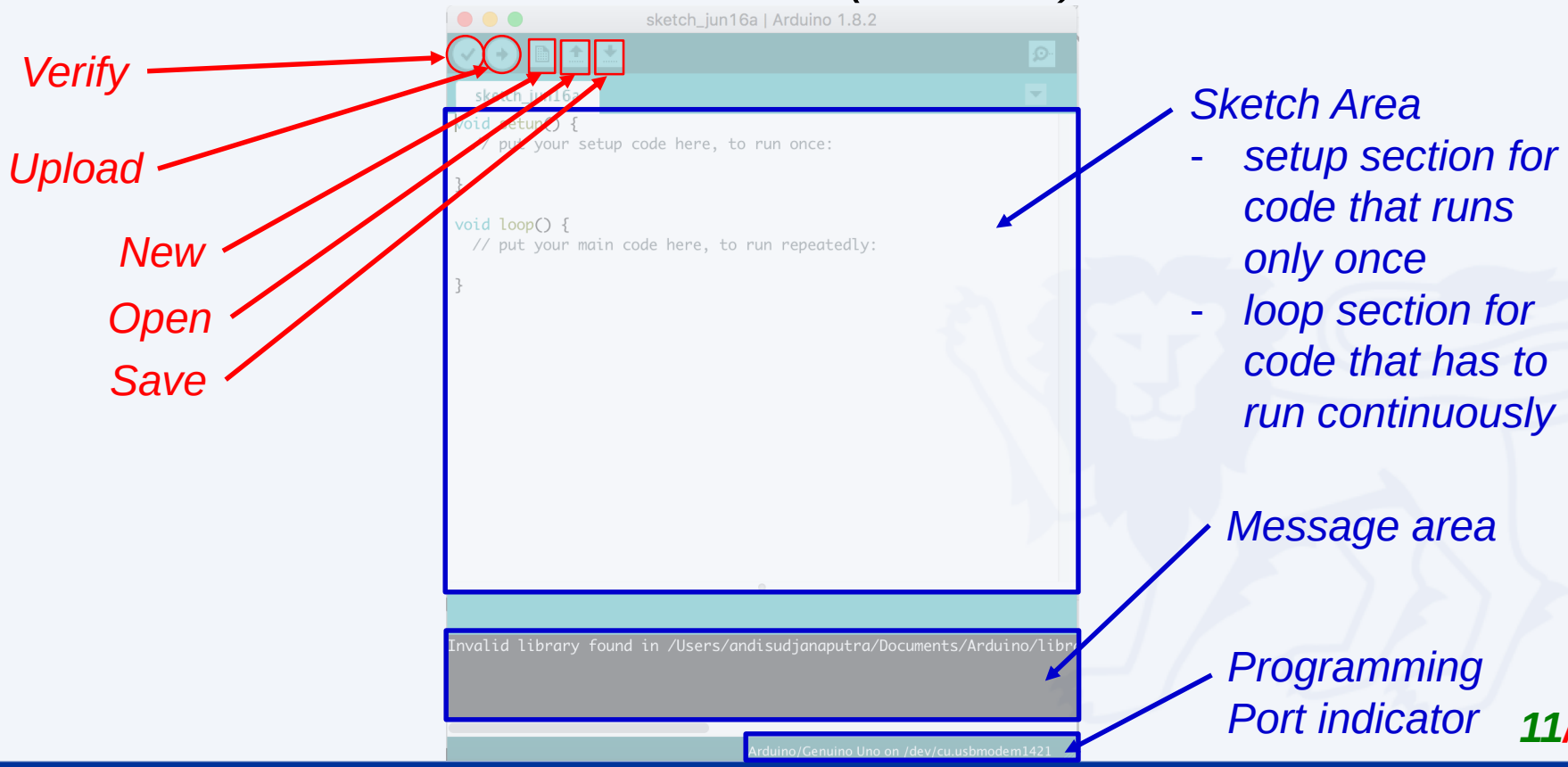
Introduction to Electronics and Arduino

- *Arduino Uno*
  - *Introduction to the Arduino's hardware*



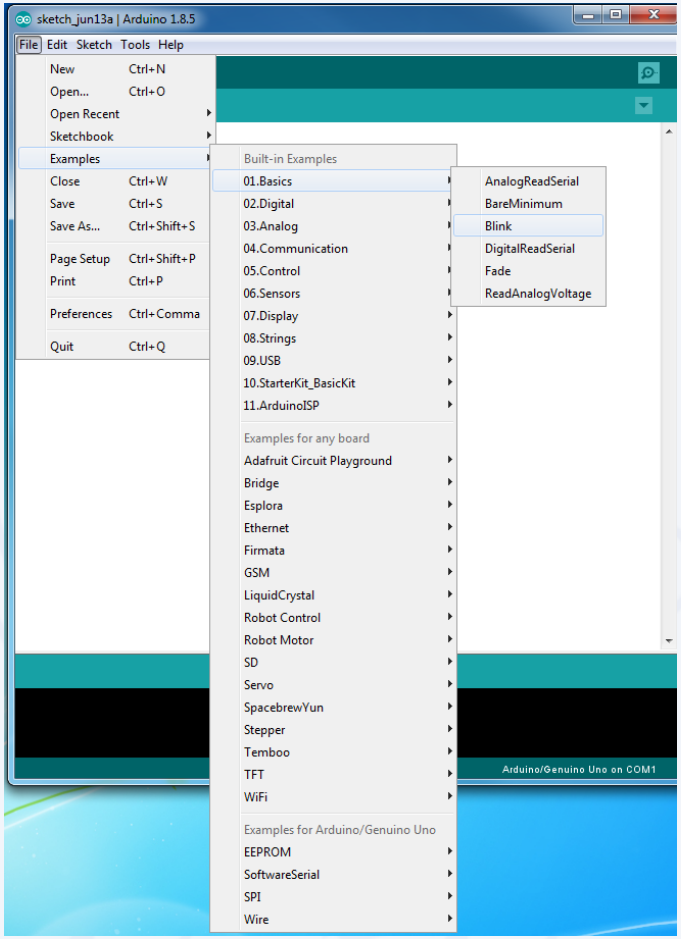
Introduction to Electronics and Arduino

- *Arduino Uno*
  - *Introduction to the Arduino IDE (software)*



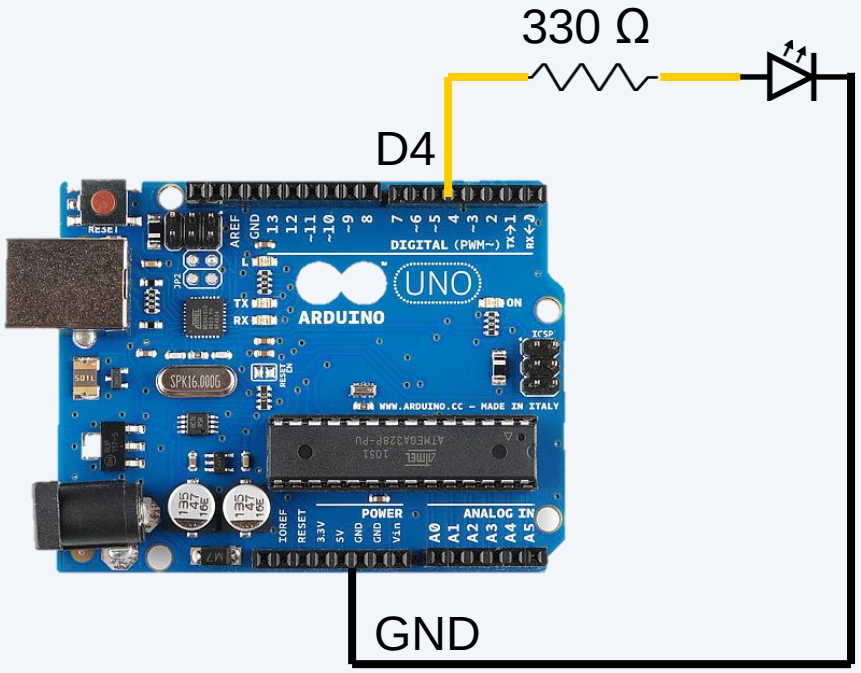
Introduction to Electronics and Arduino

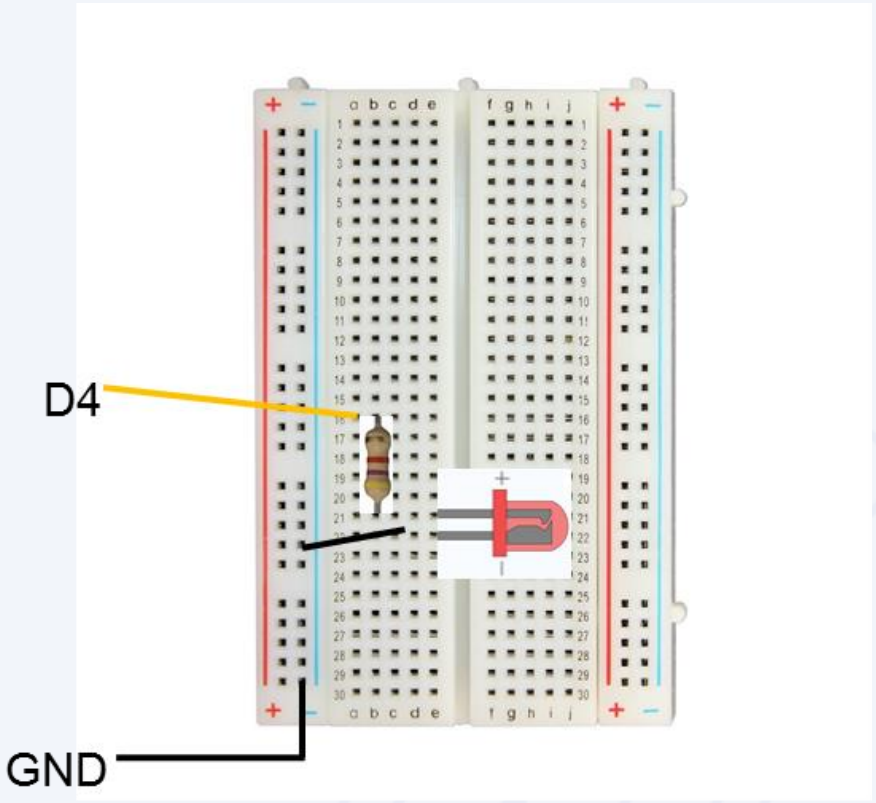
- *Making an LED blink*
  - *File > Examples > 01.Basics > Blink*
  - *Upload code.*
  - *This example turns the built in LED on for 1000 ms and then off for 1000 ms.*
  - *Other digital pins can be controlled in the same manner.*
  - *We will attempt to connect our own LED to digital pin 4 and control it in a similar way.*



Introduction to Electronics and Arduino

- *D4 Blink Circuit Hardware*





Introduction to Electronics and Arduino

- *D4 Blink Software and Syntax*

```
// Output
#define LEDpin 4

void setup() {
    pinMode(LEDpin, OUTPUT);
}

void loop() {
    digitalWrite(LEDpin, HIGH);
    delay(1000);
    digitalWrite(LEDpin, LOW);
    delay(1000);
}
```

Define the name of D4

Setup D4 as output

Repetitively, turn LED high for 1000ms, then low for 1000ms

Introduction to Electronics and Arduino

- D4 Blink Software and Syntax

Try it yourself: Make an LED connected to D8 blink
(a) at a faster rate
(b) Turn ON for 1.5s and turn OFF for 0.5s

For students who have prior knowledge or have completed the exercise in advance, try to blink an LED connected to D4 without the use of the delay function. Explain your solution to the class later.

Content

- ***Introduction to Electronics and Arduino*** 
- ***Basic Electronics***
- ***Sensors & Interfaces***
- ***Integration & Testing***

Basic Electronics

- *This section covers:*
  - *An introduction to electronic switches*
  - *Reading various inputs (digital & analog) using a microcontroller*
  - *Controlling a load with an electronic switch*
  - *Programming logic*

- *Note: Please keep all circuits built during the guided sessions, you may want to re-use them during the DIY session.*

Basic Electronics

- *Electronic Switches (Transistors)*
  - *Bipolar Junction Transistor (BJT)*
  - *Various types: NPN & PNP*
  - *Typically used as an electronic switch or an amplifier*

Circuit Symbol of BJT

BJT TO-92 Package

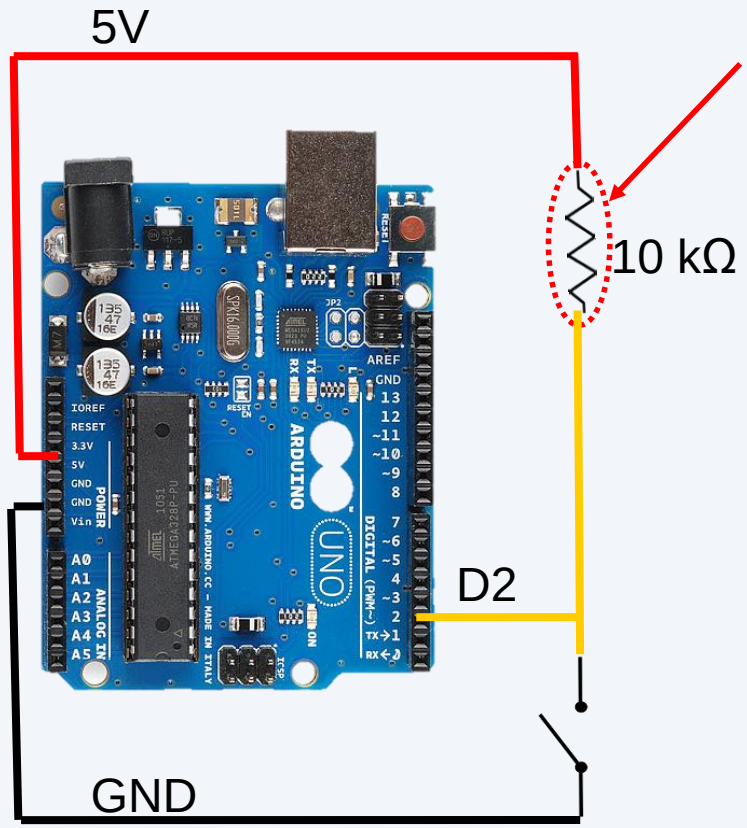
  - *Metal Oxide Semiconductor Field Effect Transistor (MOSFET)*
  - *Various types: nMOSFET, pMOSFET*
  - *Typically used as an electronic switch, analog signal & power amplifiers, motor controllers, data storage devices*

Circuit Symbol of MOSFET

MOSFET TO-220 Package

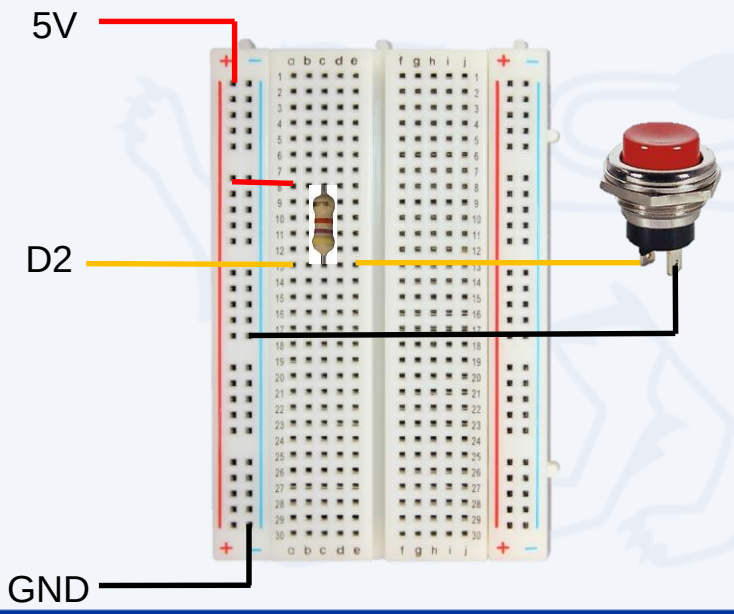
Basic Electronics

- *Reading a Digital Input from a Button (Hardware)*



Pull-up Resistor

- *Reads **HIGH** when button is **OPEN**.*
- *Prevents the input pin from reading an unknown state (floating).*
- *Limits the current flowing in the circuit.*



Basic Electronics

- *Reading a Digital Input from a Button (Software and Syntax)*

```
// Input
#define buttonPin 2

// Variable
int buttonState = 0;

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

void loop() {
    buttonState = digitalRead(buttonPin);
    Serial.println(buttonState);
    delay(1000);
}
```

Use D2 to read the button
Initialize the state of button to be low

Setup the serial communication over USB
Setup D2 as input

Read the button every second.
Show it on the serial monitor.

Basic Electronics

- *Making Decisions (Software and Syntax)*

```
void loop() {
     buttonState = digitalRead(buttonPin);
     if (buttonState == LOW) {
          digitalWrite(ledPin, HIGH);
     }
     else {
     digitalWrite(ledPin, LOW);
     }
}
```

Read the button.
If the button is pressed, switch on LED; if not switch it off.

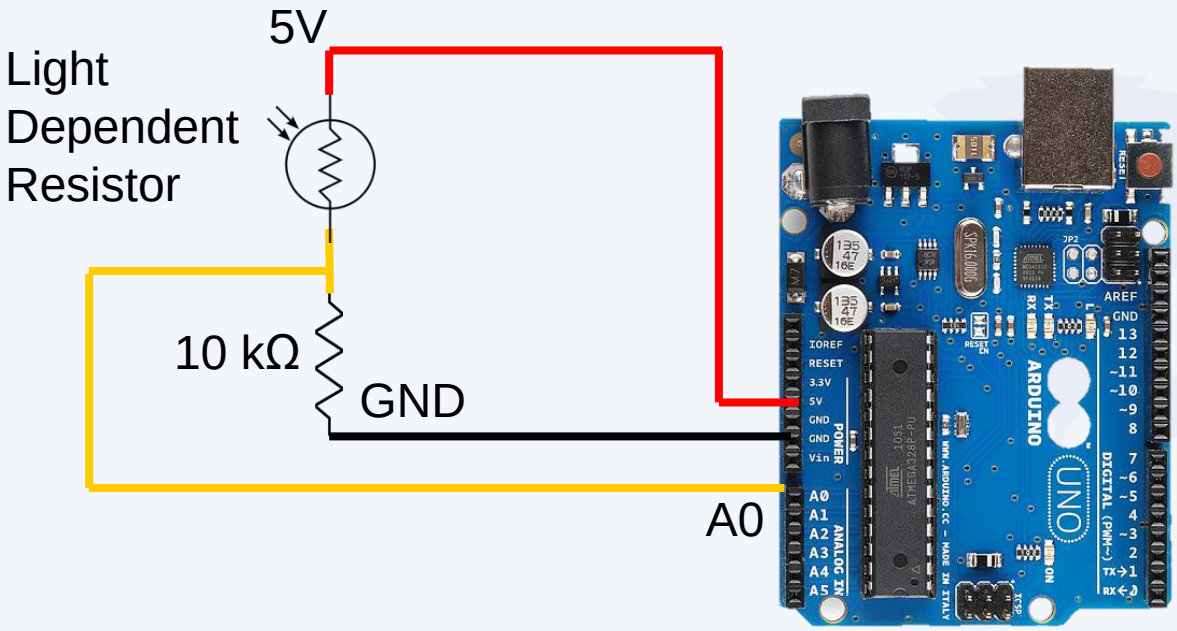
Remember a pull up resistor registers HIGH input when button is open and LOW input when button is pressed

Try it yourself: If else syntax is provided above. Using what you have learnt, make **D2 to read a button** that controls an **LED on D3**.

For students who have prior knowledge or have completed the exercise in advance, try the same exercise with **interrupts**. Use the internet to search for "Arduino interrupts" to get help if necessary.

Basic Electronics

- *Analog Read (Hardware)*
  - *Everything covered thus far is digital (HIGH or LOW state)*
  - *In real life, most quantities measured have more than 2 states*
  - *For the Arduino microcontoller to quantify things that are not binary, we make use of an **analogRead()** function*



Basic Electronics

- *Analog Read (Software and Syntax)*

```
// Variable
int ldrValue = 0;

void setup() {
    Serial.begin(9600);
}

void loop() {
    ldrValue = analogRead(A0);
    Serial.println(ldrValue);
    delay(1000);
}
```

Initialize a variable that stores the value of voltage at point 2

Read analog value at A0 and print it every second

Basic Electronics

- *Making Comparisons (Software and Syntax)*

```
void loop() {
     ldrValue = analogRead(A0);
     if (ldrValue < 500) {
          digitalWrite(ledPin, HIGH);
     }
     else {
     digitalWrite(ledPin, LOW);
     }
}
```

Read the button.
If the button is pressed, switch on LED; if not switch it off.

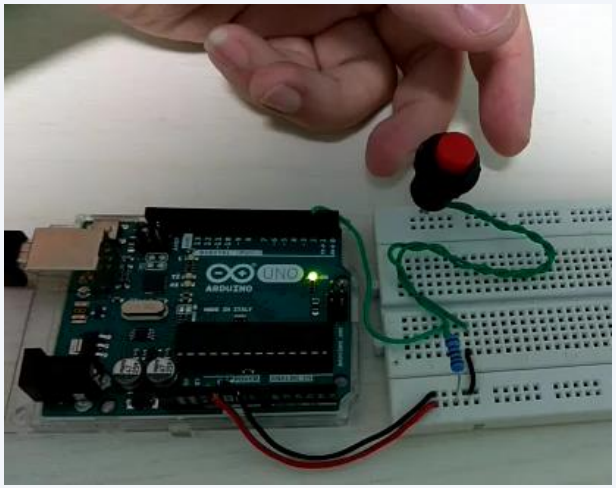
Remember a pull up resistor registers HIGH input when button is open and LOW input when button is pressed

Try it yourself: If else syntax is provided above. Using what you have learnt, use the **LDR** to turn on an **LED on D13** when it is dark and turn off the led when it is bright

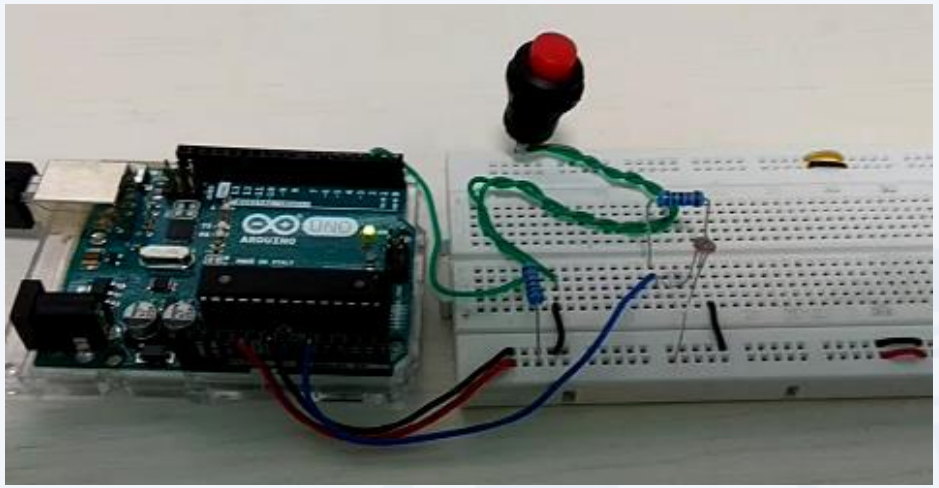
For students who have prior knowledge or have completed the exercise in advance, try the same exercise with **multiple brightness levels**. Adjust the LED brightness using PWM according to the brightness of the surroundings. Use the internet to search for "Arduino analogWrite" to get help if necessary.

Basic Electronics

- *At this stage, you should have a circuit that does either…*



Using digitalRead() on a Button

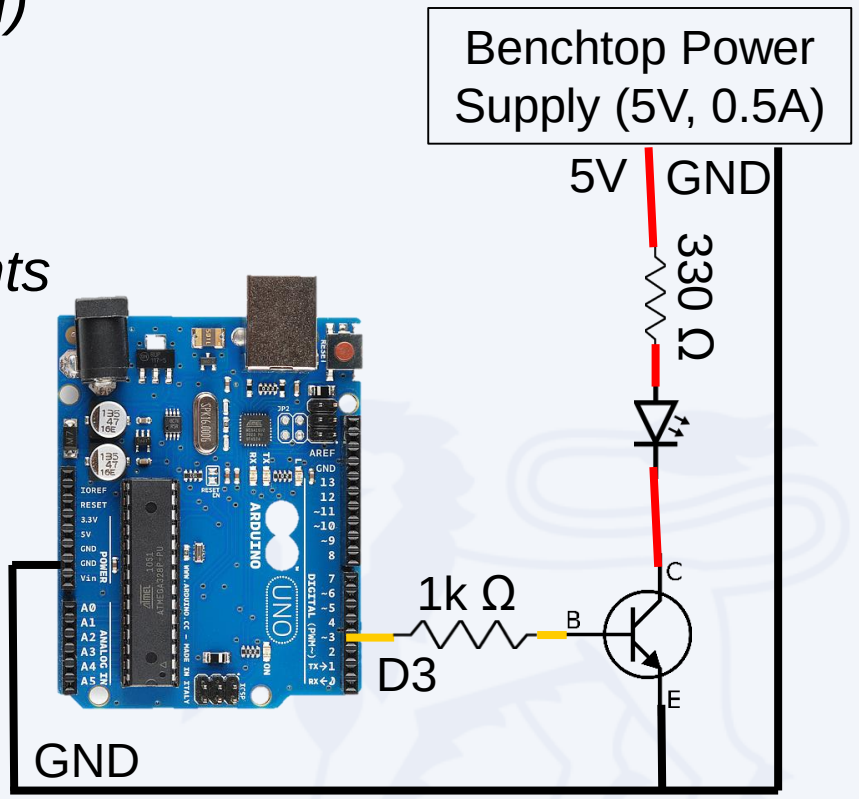


Using analogRead() on a Light Dependent Resistor

- *If not, please ask for help*

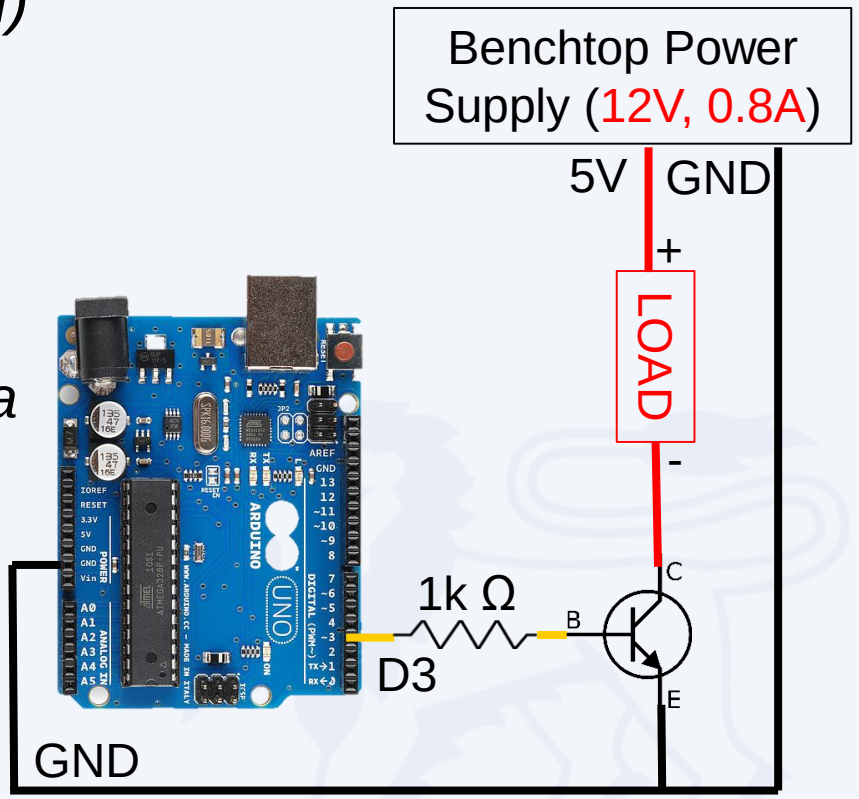
Basic Electronics

- *BJT as a Switch (Guided tutorial)*
  - *Set up circuit as shown*
  - *Search the internet for the BJT datasheet to find out which points are B, C and E*
  - *Program D3 to turn on for 1 second and off for 1 second (blink).*
  - *We could just use D3 to control the LED. Why use transistors?*



Basic Electronics

- *BJT as a Switch (Guided tutorial)*
  - *We could just use D3 to control the LED. Why use transistors?*
  - *Using a similar circuit, control a 12V Load (Fan/LED strip)*
  - *You may wish to control it with a button, a sensor, by timing a delay, etc.*



Content

- *Introduction to Electronics and Arduino* 
- *Basic Electronics* 
- ***Sensors & Interfaces***
- *Integration & Testing*

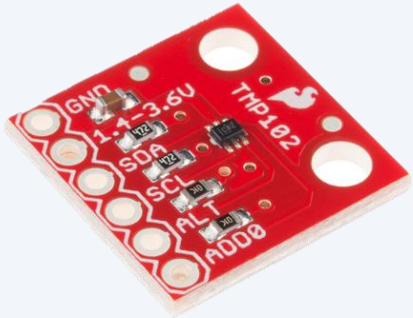
Sensors & Interfaces

- *This section covers:*
  - *An introduction to electronic sensors*
  - *Reading data from a temperature sensor with I^2C interface.*

- *Note: Please keep all circuits built during the guided sessions, you may want to re-use them during the DIY session.*

Sensors & Interfaces

- *Electronic Sensors*
  - *Available in the form of integrated circuits (ICs) or breakout boards*
  - *Typically used to collect data from surroundings*
  - *Can be used as a trigger for actuators*
  - *Temperature, humidity, pressure, colour, distance, light, etc…*



TMP102 Breakout Board on Sparkfun: https://www.sparkfun.com/

TMP102AIDRLR - Temperature Sensor IC, Open Drain, ± 1°C, -55 °C, 125 °C, SOT-563, 6 Pins

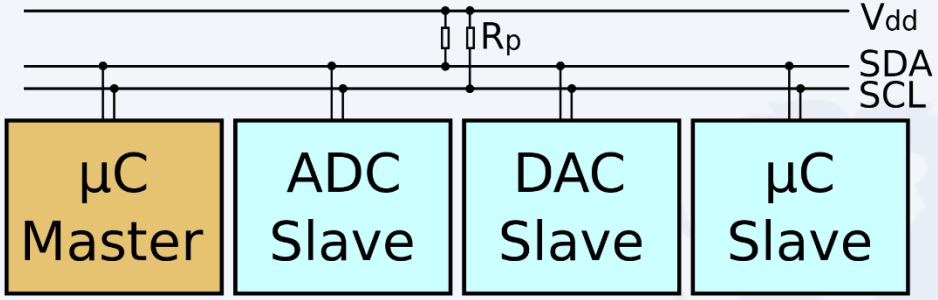
Manufacturer:	TEXAS INSTRUMENTS
Manufacturer Part No:	TMP102AIDRLR
Order Code:	2764803
Technical Datasheet:	TMP102AIDRLR Datasheet

See all Technical Docs

TMP102 IC on element14: https://sg.element14.com/

Sensors & Interfaces

- *Electronic Interfaces*
  - *Used as a means to communicate with the sensor*
  - *Eg. Universal Asynchronous Receiver/Transmitter (UART), Inter-integrated Circuit (I^2C) Protocol, Serial Peripheral Interface (SPI), etc…*

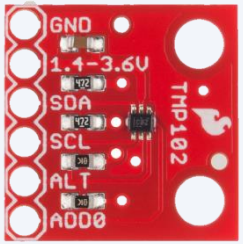


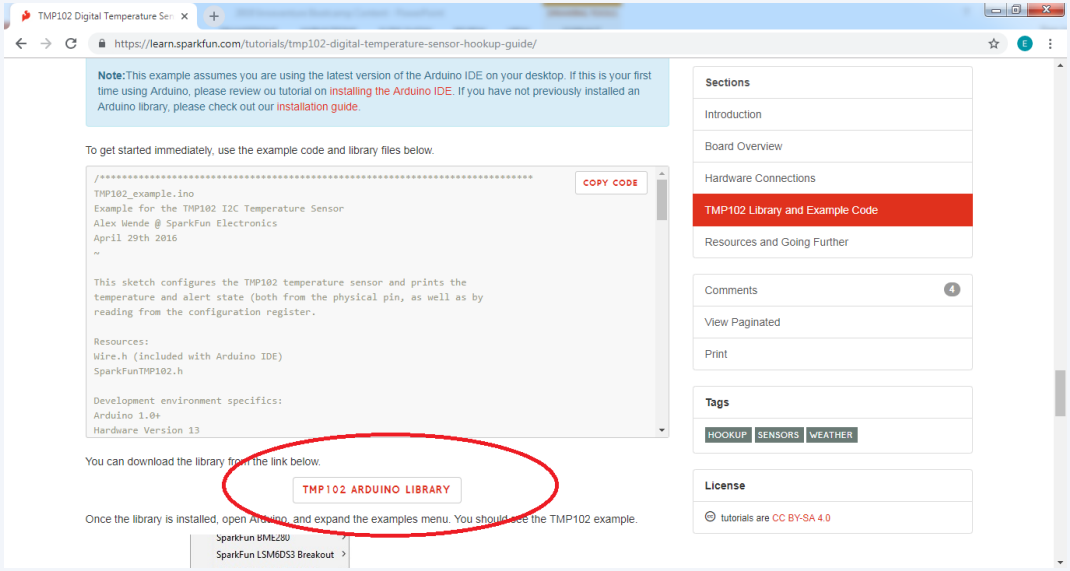
https://en.wikipedia.org/wiki/I^2C

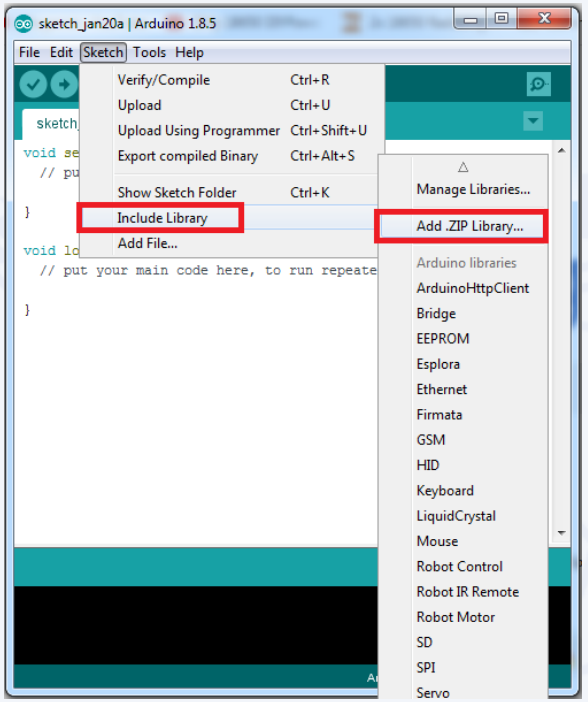
- *I^2C*
  - *Consists of Master and Slave devices*
  - *Bus architecture*
  - *2 wires, Serial Data Line (SDA) & Serial Clock Line (SCL) to perform communication with microcontroller*

Sensors & Interfaces

- *I²C Temperature Sensor (Hardware)*
  - *Read datasheet or tutorial (https://learn.sparkfun.com/tutorials/tmp102-digital-temperature-sensor-hookup-guide)*
  - *Download & Install Arduino library*

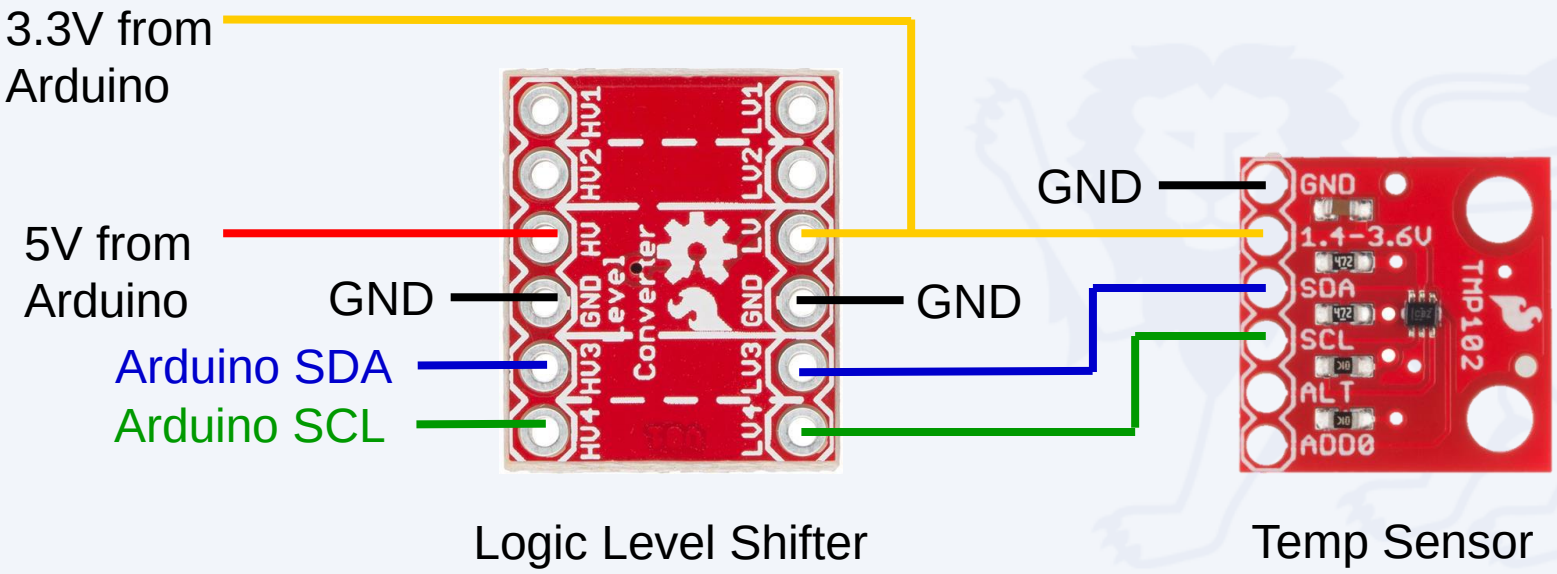






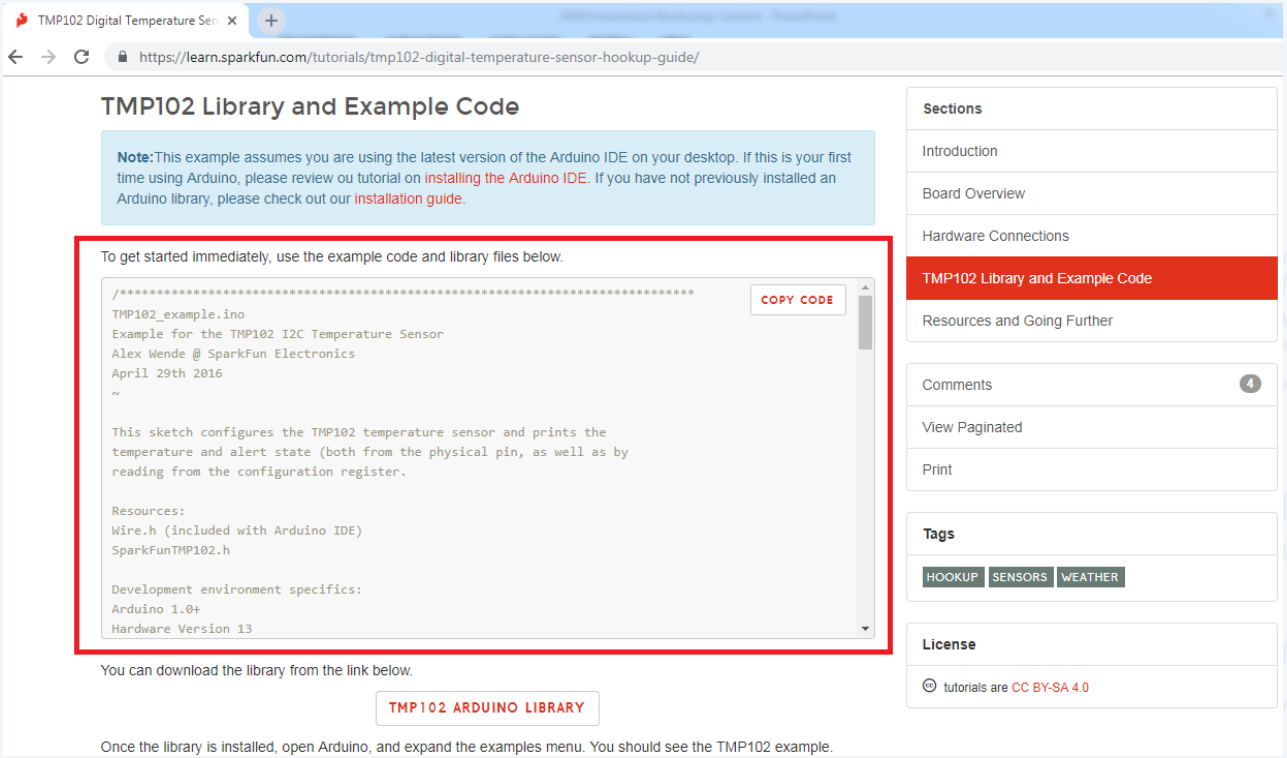
Sensors & Interfaces

- *I^2C Temperature Sensor (Hardware)*
  - *Notice that the TMP102 has operating voltage of 1.4V – 3.6V*
  - *Do not power it using the 5V supply! Use the 3.3V instead.*
  - *A logic level shifter has to be used for the 5V Arduino to communicate with the 3.3V temperature sensor*



Sensors & Interfaces

- *I^2C Temperature Sensor (Software)*
  - *Take a look and try to understand the example code.*
  - *Modify it to read the temperature of the room in degrees C*



Content

- ***Introduction to Electronics and Arduino*** 
- ***Basic Electronics*** 
- ***Sensors & Interfaces*** 
- ***Integration & Testing***

Integration & Testing

- *Integration & testing is an important part of every project.*
- *Using the hardware and software developed in the previous exercises, develop a smart system that is able to:*
  - *Sense the temperature and lighting conditions of a room*
  - *Automatically turn on the fan when it gets hot*
  - *Automatically turn on the LED lighting strip when it gets dark*
  - *Lights should be able to be controlled manually by a switch*
  - *Print and display the current status of the temperature, lighting, fan and lights on the serial monitor*

- *Note: If unsure of what to do, please ask.*
- *If you are done and would like to ask more questions, please feel free to do so.*

The End

Any Questions?