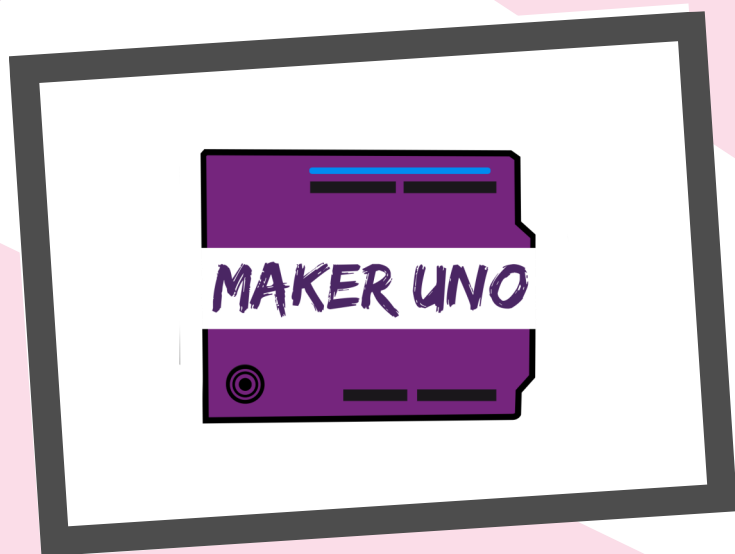


# Teacher Guidebook

Maker-UNO /

Arduino



**Written specifically based on the  
Form Two Technology Design  
Curriculum Specifications (DSKP)**

# PREFACE

This module is written specifically based on the Curriculum Specification (DSKP) for the Technology Design subject (RBT) in Form Two, published by the Malaysian Ministry of Education.

This module aims to strengthen and prepare the teachers with the right knowledge and skills to teach the electronic components of the syllabus. This module is written specifically for teachers with no prior knowledge in programming and using microcontroller.

This module is designed specifically for Maker UNO microcontroller but it also applicable for any microcontroller that are compatible with the Arduino UNO form factor, for instance, CT-UNO. This module also comes with a student module that can be used alongside with the teacher module.

This module was developed and written by teachers for teachers, a joint collaboration between Cytron Technologies and Arus Academy. Arus Academy is a social enterprise based in Alma, Bukit Mertajam, Malaysia. It was co-founded by four secondary school teachers with the mission to provide high quality and relevance for STEM education for the underprivileged children.

Arduino operates on the spirit and principle of open source. This module was developed with the same principle in mind. Users are encouraged to share, modify and redistribute all information contained in the module. All we are asking is to acknowledge and give the appropriate credits to the source material.

This module is protected under the Creative Commons Attribution -ShareAlike 4.0 International (CC BY-SA 4.0).





This document is licensed under the Attribution-ShareAlike 4.0 International Creative Commons mons.

**You are free to**

- **Share**– Copy and redistribute the material in any medium or format
- **Adapt** – remix, transform, and build upon the material for any purpose, even commercially.

**Under the following condition**

- **Attribution** – You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **Share Alike** – If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No Additional Restriction** – You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

**Publish Summary**

First published in Bahasa 24/01/2018

Second published in English 20/05/2018

– Revision 1: Fixed typos / translation errors based on readers' feedback

**For any information / correction / suggestions, please contact:**

[enquiry@arusacademy.org.my](mailto:enquiry@arusacademy.org.my)

# TABLE OF CONTENTS

SUGGESTED T&L TIME ALLOCATION	6
PREPARATION	7
UNIT 1– MICROCONTROLLER THEORY	8
1.1– WHAT IS MICROCONTROLLER AND ITS FUNCTION	9
1.2– PARTS IN A MICROCONTROLLER	12
1.3– BLOCK DIAGRAM	14
1.4– INTRODUCTION TO SCHEMATIC DIAGRAM	15
UNIT 2– OUTPUT	18
2.1– INTRODUCTION TO OUTPUT PROGRAMMING	19
2.2– OUTPUT DEVICES	30
2.3– INTRODUCTION TO OUTPUT CIRCUIT CONNECTION	34
2.4– OUTPUT CIRCUIT SIMULATION	42
UNIT 3– INPUT	49
3.1– INTRODUCTION TO INPUT PROGRAMMING	50
3.2– INPUT DEVICES	54
3.3– INTRODUCTION TO INPUT CIRCUIT CONNECTION	59
3.4– INPUT CIRCUIT SIMULATION	63





# TABLE OF CONTENTS

UNIT 4- COMBINATION OF OUTPUT DAN INPUT	68
4.1- INTRODUCTION TO CONDITIONAL PROGRAMMING STRUCTURE	69
4.2- OUTPUT AND INPUT CIRCUIT CONNECTION	76
4.3- OUTPUT AND INPUT CIRCUIT SIMULATION	78
4.4- PROJECT CREATION	83
APPENDIX 1-HOW TO PROGRAM WITH ANDROID SMART PHONE	84
APPENDIX 2-ADDITIONAL REFERENCE	85
STUDENT LEARNING MODULE	87
UNIT 1- MICROCONTROLLER	88
UNIT 2- OUTPUT	92
UNIT 3- INPUT	103
UNIT 4- INPUT AND OUTPUT	111
STUDENT LEARNING MODULE ANSWER SCHEME	
UNIT 1- MICROCONTROLLER	118
UNIT 2- OUTPUT	121
UNIT 3- INPUT	131
UNIT 4- INPUT AND OUTPUT	138



# SUGGESTED T&L TIME ALLOCATION

UNIT	TOPIC	TIME ALLOCATION	REMARKS
UNIT 1	MICROCONTROLLER THEORY	2 HOURS	CERTAIN TOPICS WILL REQUIRE THE USE OF LAPTOPS OR MOBILE PHONES
UNIT 2	OUTPUT	3 HOURS	
UNIT 3	INPUT	2 HOURS	
UNIT 4	COMBINATION OF OUTPUT / INPUT	3 HOURS	

# PREPARATION

## **Arduino Microcontroller**

- Teacher may purchase Maker-UNO sets on Cytron Web Store:  
<https://www.cytron.io/p-maker-uno>
- Teacher may also purchase Arduino sets from any online stores.

## **Arduino Software**

- Software may be downloaded on Arduino official website :  
<https://www.arduino.cc/en/Main/Software>

## **Printing the module**

- The teacher and student module is designed to be printed in A5 (half A4 page)

# UNIT 1

# MICROCONTROLLER

## LEARNING STANDARD

- 2.4.1 State the meaning of microcontroller and microprocessor
- 2.4.2 Explain parts within the microcontroller
- 2.4.3 Sketch a schematic diagram for a microcontroller

## ASSESSMENT STANDARD

- TIER 1 State the meaning and parts in a microcontroller
- TIER 2 Explain the function of each hardware in a microcontroller
- TIER 3 Sketch a schematic diagram using a microcontroller

## SUB-UNIT

UNIT 1.1	What is a microcontroller and a microprocessor?	LS: 2.4.1
UNIT 1.2	Parts in a microcontroller	LS:2.4.2
UNIT 1.3	Introduction to Block Diagrams	
UNIT 1.4	Introduction to schematic diagrams	LS: 2.4.3

## SUGGESTED T&L TIME BREAKDOWN

UNIT 1.1	30 MINUTES
UNIT 1.2	
UNIT 1.3	30 MINUTES
UNIT 1.4	60 MINUTES

## PREPARATION

- 1) Can be used along with Student Module

## UNIT 1.1

# WHAT IS A MICROCONTROLLER AND A MICROPROCESSOR?

### Learning Objective

In this unit, SWBAT state the meaning of a microcontroller as well as a microprocessor.

### Success Criteria:

Students are able to state the meaning of a microcontroller as well as a microprocessor orally or in writing.

A microcontroller is a programmable hardware. This programmable hardware uses a microprocessor as its central processing unit (CPU), random access memory (RAM), read only memory (ROM), as well as input and output (I/O) ports.

CPU, RAM, ROM, and I/O ports are located in a single chip.

Microprocessors only have CPUs within its chip. RAM, ROM, and I/O ports from the CPU.

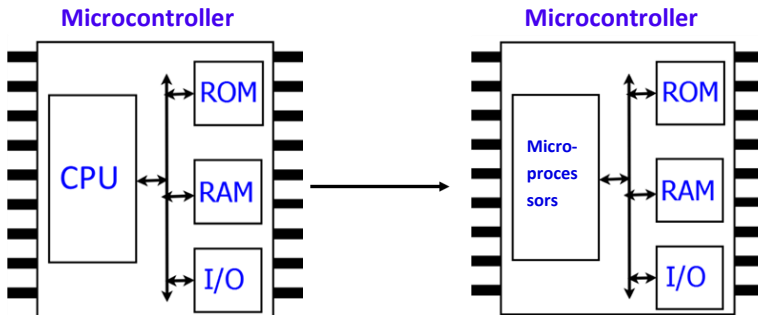


Diagram 1.1.(a):Diagram shows that the CPU is also a microprocessor.

Microcontrollers are frequently used in various devices around us. For example, in television sets, air-conditioners, and the modern car.

# UNIT 1.1

Microcontrollers receive inputs, processes the information received and produced the desired output accurately. A microcontroller can be considered small scale computer or a brain.

Comparisons could be made to Diagram 1.1(b). The microcontroller acts in the same way as the nervous system in our body. The nerve receptors sends information to the spinal cord to be processed in the brain. The brain will then send the appropriate response to the muscles.

A microcontroller can be given instructions through programming to execute specific tasks.

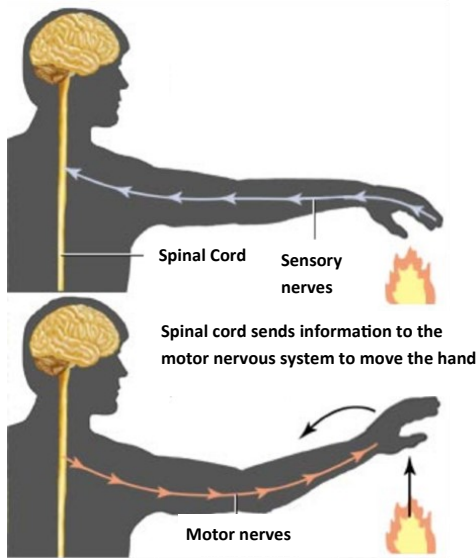


Diagram 1.1.(b): Diagram of the human nervous system.

After receiving instructions, the microcontroller will store its instructions without needing to be connected to the computer.

In conclusion, a microcontroller works according to Diagram 1.1(c) where it receives information from inputs and responds according to the program within its memory.



Diagram 1.1.(c): Diagram shows the parts in a microcontroller.

## UNIT 1.1

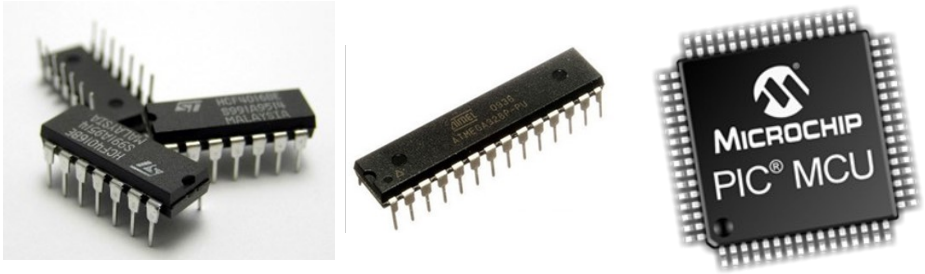


Diagram 1.1.(d): Jenis mikropengawal yang wujud

Some of the more commonly used Microcontroller Development Board:



Diagram 1.1.(e): Intel Edison



Diagram 1.1.(f): Arduino UNO

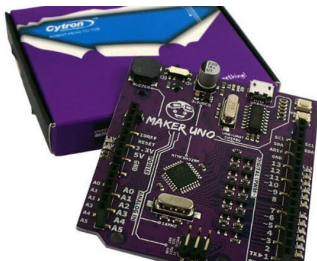


Diagram 1.1.(g): Maker UNO



Diagram 1.1.(h): Beaglebone Black

## UNIT 1.2

# COMPONENTS OF A MICROCONTROLLER

### Learning Objective

In this unit, SWBAT explain different components in a microcontroller.

### Success Criteria:

Students are able to state the different components of a microcontroller orally or in writing.

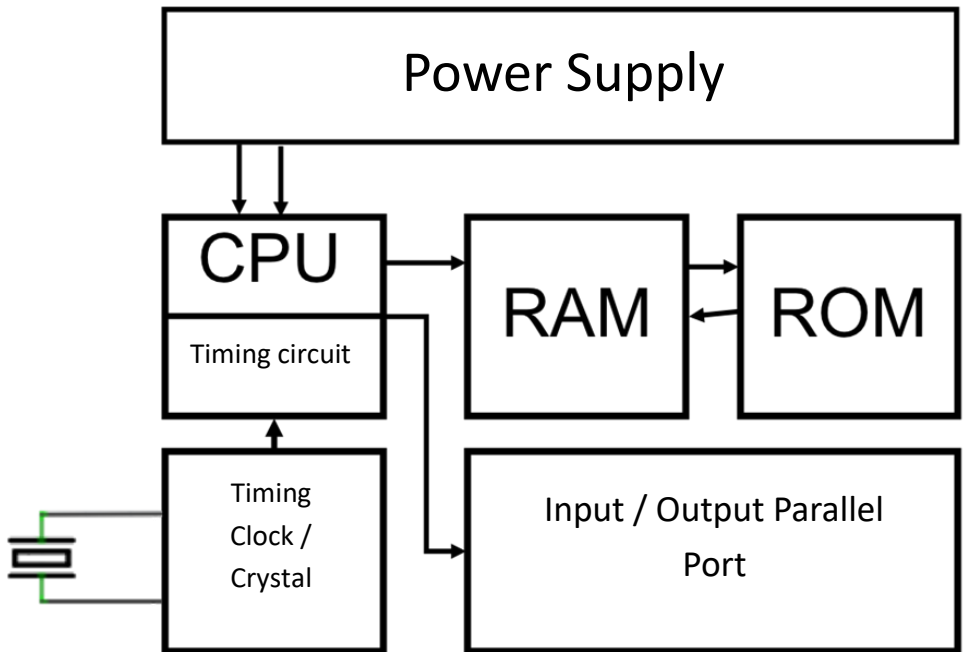


Diagram 1.2.(a): Diagram shows the different components in a microcontroller



# UNIT 1.2

Parts	Function
CPU	Receives information and instructions and processes inputs and outputs.
RAM & ROM	Memory space to store and execute instructions.
I/O Parallel Ports	Connects to input and output devices such as LED, motor, & sensors.  There are two types of Input and Output signals— analog and digital signals
Timing Circuit	Used to maintain the timing of the internal cycle of the microcontroller.
Timing Crystal	Produces constant oscillation frequency as reference for the timing circuit.
Power Supply	Provides electricity to the microcontroller.

## UNIT 1.3

# INTRODUCTION TO BLOCK DIAGRAMS

### Learning Objective

In this unit, SWBAT draw block diagrams

### Success Criteria:

Students are able to draw a block diagram which includes a microcontroller, power supply and input & output.

The following diagram shows an example of a block diagram:

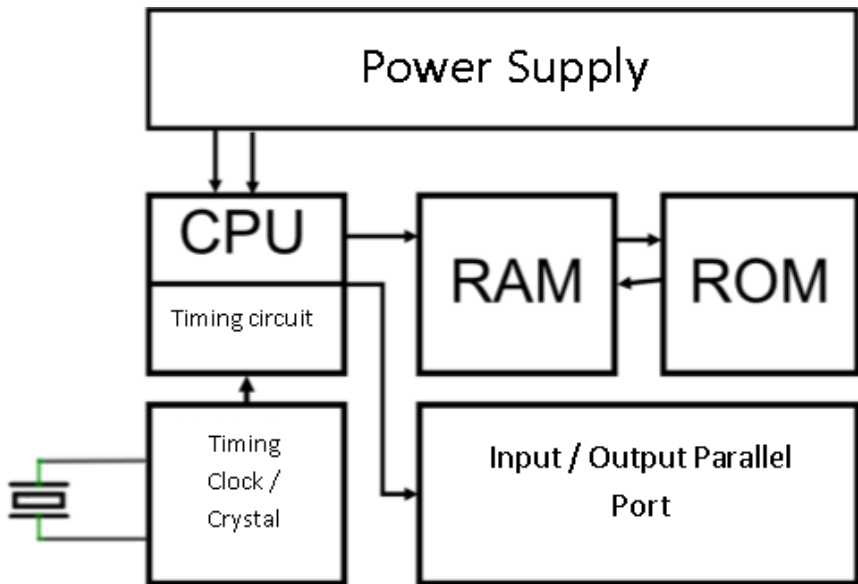


Diagram 1.3.(a): Block diagram for a microcontroller

## UNIT 1.4

# INTRODUCTION TO SCHEMATIC DIAGRAM

### **Learning Objective**

In this unit, SWBAT draw schematic diagrams.

### **Success Criteria:**

Students are able to draw schematic diagrams accurately based on drawing standards.



---

Guide to drawing standardized schematic diagrams.

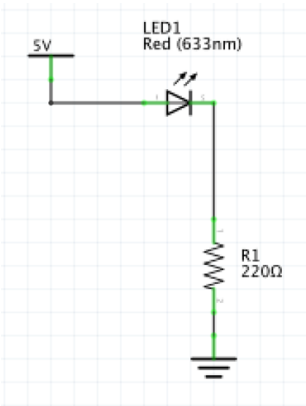
1. Ensure all lines drawn are straight.
2. Ensure all lines are not arrows.
3. Ensure all lines are drawn horizontally or vertically only.
4. Minimize drawing lines crossing each other to avoid confusion.
5. Use standard symbols to represent each component.
6. Ensure that all components in the schematic diagram are labeled.

# UNIT 1.4

Important elements in a schematic diagram.

Part	Function
 Ground (GND)	It's the negative terminal of a battery or power supply. Electronic circuits will generally end with GND.
 Power Supply 5V	Source of power. The starting point of circuits.

The following is an example of a schematic diagram of an electronic circuit.



Take note that the circuit fulfilled the condition of of a good circuit.

- ☑ Ensure all lines drawn are straight
- ☑ Lines drawn are not arrows
- ☑ All lines are drawn vertically or horizontally only
- ☑ No crossing of lines
- ☑ Use standarized symbols to represent each components in the circuit
- ☑ All components in the circuit are labelled

Diagram 1.4.(a): Schematic diagram

# UNIT 1.4

The following is the schematic symbol for a microcontroller.

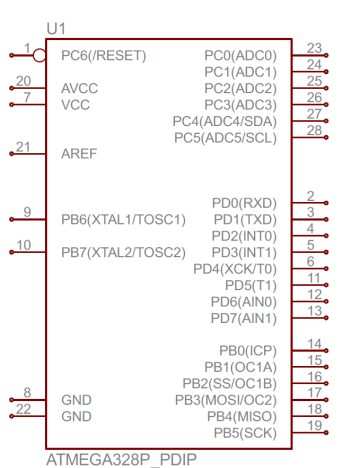


Diagram 1.4.(b): Schematic diagram for an ATMEGA328P chip. (The microcontroller for an Arduino UNO)

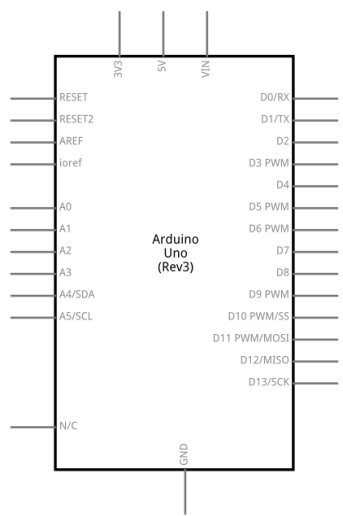


Diagram 1.4.(c): Schematic diagram for Arduino UNO development board.

The following diagram is a schematic diagram for an electronic circuit using a microcontroller.

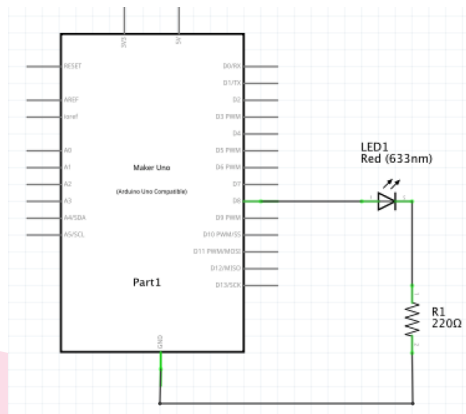


Diagram 1.4.(d): Schematic diagram for an LED circuit connected from pin D8 to GND.

# UNIT 2

# OUTPUT

## LEARNING STANDARD

- 2.4.4 Build functioning simulated circuit with dedicated software.
- 2.4.5 Connect input and output circuit on the microcontroller
- 2.4.6 Write simple program based on input and output circuit

## ASSESSMENT STANDARD

TIER 4 Test out functionality of a circuit that includes microcontroller.

## SUB-UNIT

UNIT 2.1	Introduction to Output circuit programming	SP: 2.4.6
UNIT 2.2	Types of output devices	
UNIT 2.3	Introduction to connecting output circuit	SP: 2.4.5
UNIT 2.4	Output circuit simulation	SP: 2.4.5

## RECOMMENDED T&L TIME ALLOCATION

UNIT 2.1	60 MINUTES
UNIT 2.2	60 MINUTES
UNIT 2.3	
UNIT 2.4	60 MINUTES

## PREPARATION

- 1) Ensure that there are enough Arduino / Maker UNO for each students.
- 2) Unit 2.1 requires the use of a computer or a smartphone to program the microcontroller. (Refers to Appendix 1)
- 3) Unit 2.4 requires computer and internet
- 4) Can be used along with Student Module

## UNIT 2.1

# INTRODUCTION TO OUTPUT CIRCUIT PROGRAMMING

### Learning Objective

In this unit, SWBAT write simple program to control output circuit.

### Success Criteria:

Students will be able to control built in LED in Maker UNO / Arduino

Microcontroller can control output circuit by sending signals to its output pin on the microcontroller. With this, any components that are connected to those pins can be controlled via programming.

In general, microcontrollers can be made to send out two different types of signals—analogue and digital signals. Before we write any program, we need to understand the difference between digital signals and analogue signals.

Digital signals has two state—ON (current flowing in the circuit) or OFF (no current in the circuit). By sending ON digital signal to pin 5, we can activate any electronic components that are connected to the pin. Likewise, by sending an OFF digital signal to pin 5, we will be able to deactivate / turn off any components that are connected to pin 5.

Analog signals, on the other hand, use more than two different signals. For Arduino, it has a range of 256 values—from 0 to 255. 0 means that there is no current in the circuit (voltage value of 0) and 255 is the maximum voltage in the circuit. The analog value (0 to 255) is directly proportional to the voltage values that will be sent to the output pin (for instance, 127 will send out 50% of the maximum voltage value).

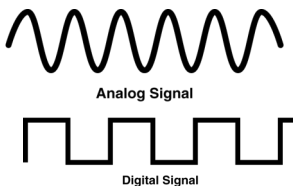


Diagram 2.1.(a): Comparison between digital and analog signals

In other words, if we connect an LED to an analog pin, 0 will turn off the pin, 127 will cause the LED to light up at 50% brightness whereas 255 will light it up at 100% brightness.

# UNIT 2.1

Microcontroller runs two main function—setup function and loop function. Setup function will be executed only once after the Arduino is turned on or reset, whereas, the loop function will be run continuously until the Arduino is turned off.

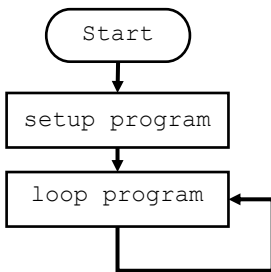


Diagram 2.1.(b): Flowchart for Arduino

We can use the Arduino IDE (Integrated Development Environment) which to program an Arduino. The software can be downloaded from Arduino website [www.arduino.cc](http://www.arduino.cc).

After opening the Arduino IDE software, it would be a good idea to display the line numbers for the code for readability of the code. To do that, click on File -> Preference, then select the Display Line Number option, then click OK.

You will find the user interface in the Arduino IDE to be similar to the one in Diagram 2.1(c). Notice how the two functions (setup and loop) is readily available for you to use.

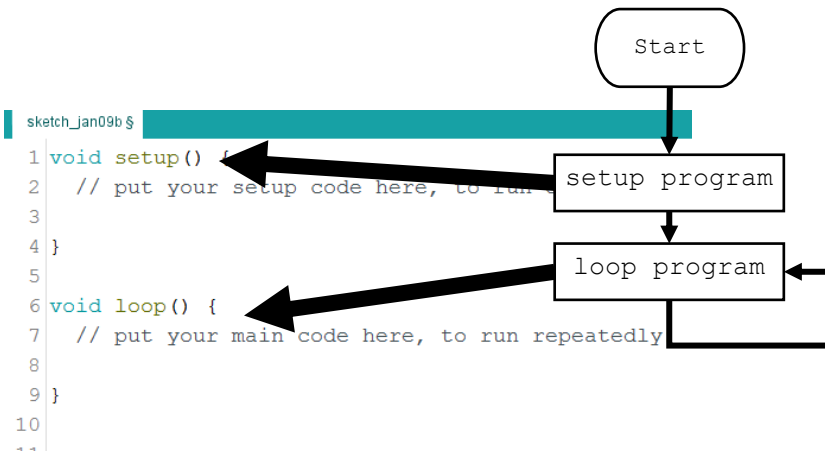


Diagram 2.1.(c): Comparison between flowchart and Arduino IDE programming



# UNIT 2.1

Steps to writing a good Arduino program

## Step 1–Set **pinMode** in **void setup()**

Identify and specify the pin numbers and the mode to be used—whether it is input mode or output mode.



## Step 2–Plan Your Program

Think about what do you wish to accomplish with the program. Sketch out flow chart if needed.



## Step 3–Write your Code in **void loop()**

Write your code in the loop function. Your code may involve sending output signal, receiving and reading input signal, as well as knowing what output signal to be sent out based on input signal.

Rules in Arduino programming:

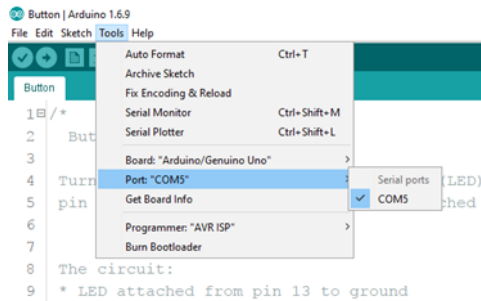
- Ensure that each line of code ends with the semicolon sign ;
- Make sure keywords are spelled correctly (**orange** / **blue**).
- () and {} comes in pair.

# UNIT 2.1

Steps to writing a good Arduino program

## Step 4—Connect Arduino to Computer with a USB Cable

Connect the Arduino board to the computer with a USB cable, then select the right port.



## Step 5—Press the upload icon (arrow) and save the file with an appropriate name

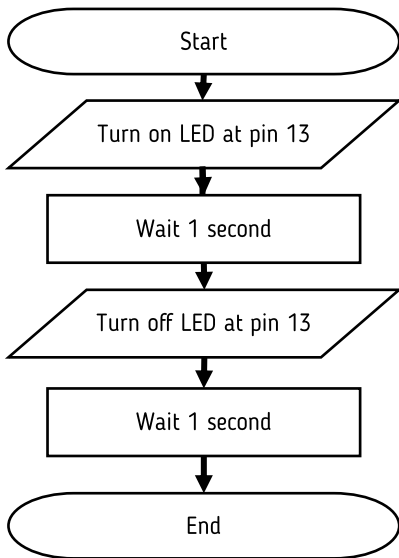


## UNIT 2.1


The code will remain in Arduino memory once the code is uploaded. It will remain there until the program is erased or written over.

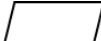
Before we write a program, we should plan the program with a flowchart.

We will be creating the following program:



This is a sequential program structure—program which does not contain any branches. Notice how there are different shapes being used to represent the different kinds of code.

 Represents the beginning or an end to a code

 Represents the input or output of an algorithm

 Represents steps / procedures in an algorithm

# UNIT 2.1

Here's the code for the flowchart above:

```
1 void setup() {  
2   pinMode(13,OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(13,HIGH);  
7   delay(1000);  
8   digitalWrite(13,LOW);  
9   delay(1000);  
10 }
```

Fungsi untuk setiap satu baris adalah seperti berikut:

Line	Programming Statement	Function
1	<code>void setup() {</code>	<p>State the beginning of the setup function. Setup function is used to define commonly to define the modes of the pins to be used.</p> <p>Notice how void and setup are in different colour—this indicates that they are special keywords in Arduino programming.</p> <p>The { symbole marks the beginning of the code for the set up function</p>

# UNIT 2.1

Line	Programming Statement	Function
2	<code>pinMode(13,OUTPUT);</code>	<p>Set the mode of the pin—whether it will be an input or an output pin.</p> <p>This line sets pin 13 to be an output pin</p>
3	<code>}</code>	<p>The <code>}</code> symbol marks the end of the setup function. Beginning from the <code>{</code> symbol on line 1, setup function includes only codes between <code>{</code> and <code>}</code>, which is line 1 to 3 only.</p>
5	<code>void loop() {</code>	<p>This line marks the beginning of the loop function. Loop function is the function that will be run indefinitely until the Arduino is turned off. The word void and loop are keywords too in Arduino—hence the different colour.</p> <p><code>{</code> symbol marks the beginning of the loop function</p>

## UNIT 2.1

Baris	Pernyataan atur cara	Fungsi
6	<code>digitalWrite(13, HIGH);</code>	<p>DigitalWrite sends digital signals to the specified pin. This line will send HIGH signal to pin 13, which will turn on any components connected to pin 13.</p> <p>Arduino is case sensitive—W for digitalWrite and HIGH needs to be written in capital letter—otherwise it will not be recognized as keywords and will instead be in other colour.</p>
7	<code>delay(1000)</code>	<p>Delay will wait for a specified amount of time. The value written in the parenthesis is measured in milliseconds. For instance, 1000 milliseconds is equivalent to 1 second.</p> <p>Thus, delay(1000) means that the system will halt and wait for 1 second.</p>
8	<code>digitalWrite(13, LOW)</code>	<p>This line will send LOW to pin 13. It will turn off / deactivate components that are connected to pin 13</p>

## UNIT 2.1

Baris	Pernyataan atur cara	Fungsi
9	<code>delay(1000)</code>	<p>We need to put another delay after turning off pin 13. Without this line, after line 8 is executed, the program flow will go back to line 6.</p> <p>Eventhough pin 13 is turned off, but the length of off time is too quick and is not detectable by naked eye—so to us, it will look as if it is not turned on.</p>
10	<code>}</code>	<p>The <code>}</code> symbol marks the end of the loop function. Beginning from the <code>{</code> symbol on line 5, loop function only includes lines between line 5 and line 10 only.</p>

With this, we can control pin 13 to turn on and off every 1 second. You will notice that the LED on the Arduino board will blink too. This is because pin 13 is also the pin for the built-in LED.

## UNIT 2.1

You can also use `analogWrite` to send out analog signals. Not all Arduino pins are capable of producing analog signals, only those pins which have the ~ symbol on the board. The analog pins are pin 3,5,6,7, 9, 10 and 11

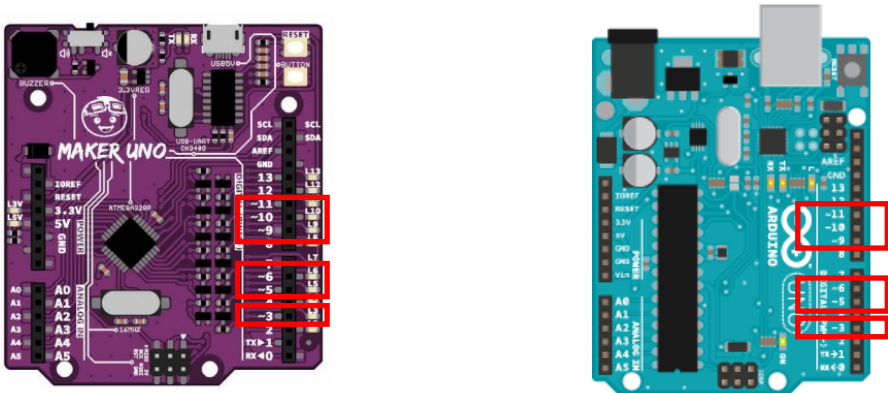


Diagram 2.1.(d): Location of analog pins (also known as PWM pins)

`analogWrite` works as follow:

**`analogWrite(pin, value)`** whereby value ranges from 0 to 255

For instance,

- `analogWrite(9, 255)` will send 255 which is the maximum voltage to the pin 9. Any components that are connected to pin 9 will receive maximum voltage. If there's any LED at pin 9, it will be turned on with 100% brightness.
- `analogWrite(9, 127)` will send 127 (half of 255) to pin 9. Any components that are connected to pin 9 will receive 50% of maximum voltage. If there's any LED at pin 9, it will be turned on with 50% brightness.
- `analogWrite(9, 0)` will send 0 to pin 9. Any components that are connected to pin 9 will receive 0 voltage. If there's any LED at pin 9, it will not be turned on at all.



# UNIT 2.1

**The following section is meant for Maker UNO only. For other variants of Arduino, you will need to connect LEDs to the circuit.**

Maker UNO has LED on every digital pin. You can write programming to control LED on every pin.

For example:

```
digitalWrite(9, LOW);  
digitalWrite(8, HIGH);  
delay(1000);  
digitalWrite(9, HIGH);  
digitalWrite(8, LOW);  
delay(1000);
```

This program will cause the LED on pin 9 and pin 8 to blink alternately.

# UNIT 2.2

## OUTPUT DEVICES

### Learning Objective

In this unit, SWBAT learn how to use various output devices

### Success Criteria:

Students will be able to list down names and functions of output devices.

Output devices will execute signals given by the pins that they are connected to, to the Arduino. Arduino can provide 5v to turn on the devices connected to each pin. The commonly used output devices in project making are, light emitting diode (LED) and buzzer. These output devices can be turned on without external power supply. There are also other output devices that require additional power supply, such as direct current motor 12v, LED light 12v and others.

## Resistor

- The higher the resistor value, the higher the resistance for the current to flow through.
- Resistors are important to prevent devices from burning out or to act as voltage divider.
- Resistor values can be read by looking at the colour band on the resistor.
- The symbol for resistor on a schematic diagram is as follow:



www.resistorguide.com

	Color	Significant figures			Multiply	Tolerance (%)	Temp. Coeff. (ppm/K)	Fail Rate (%)
Bad	black	0	0	0	x 1		250 (U)	
Beer	brown	1	1	1	x 10	1 (F)	100 (S)	1
Rots	red	2	2	2	x 100	2 (G)	50 (R)	0.1
Our	orange	3	3	3	x 1K		15 (P)	0.01
Young	yellow	4	4	4	x 10K		25 (O)	0.001
Guts	green	5	5	5	x 100K	0.5 (D)	20 (Z)	
But	blue	6	6	6	x 1M	0.25 (C)	10 (Z)	
Vodka	violet	7	7	7	x 10M	0.1 (B)	5 (M)	
Goes	grey	8	8	8	x 100M	0.05 (A)	1 (K)	
Well	white	9	9	9	x 1G			
Get	gold				x 0.1	5 (J)		
Some	silver				x 0.01	10 (K)		
Now!	none					20 (M)		

3m digit only for 5 and 6 bands

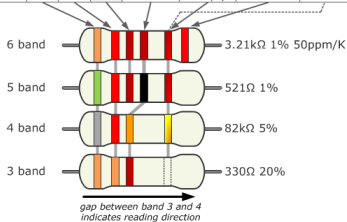


Diagram 2.2.(b): 10,000 Ohm resistor

Diagram 2.2.(a): Ways to read resistor values  
(Source: Resistorguide.com)

# UNIT 2.2

## Light Emitting Diodes (LED)

- Light Emitting Diodes are diodes which emit light when current passes through it.
- LED can be used as decorative lights and must be used alongside with a resistor to prevent it from burning.
- LED has polarity—it has a positive side and a negative side.
- There is a built-in LED at every pin on the Arduino
- LED symbol is as follow:

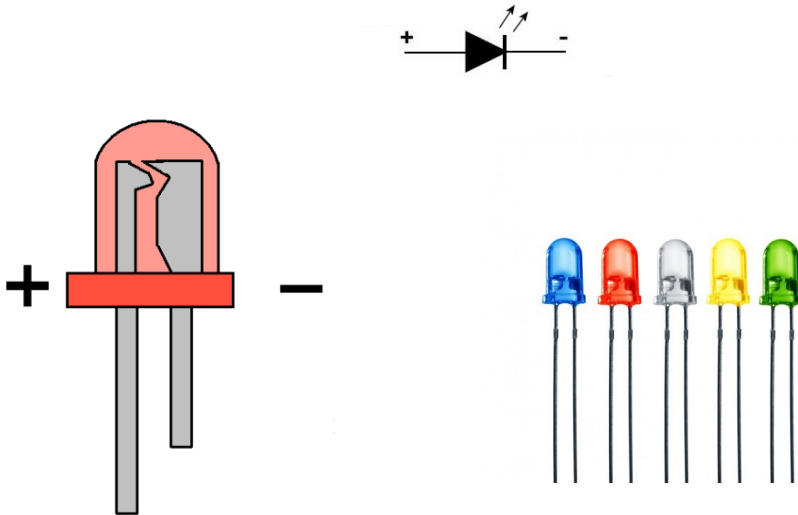


Diagram 2.2.(d): LEDs

Diagram 2.2.(c): How to identify the positive and negative leg of the LED

# UNIT 2.2

## Buzzer

- Buzzers work by turning electric signals to sound.
- Like the LED, buzzers have polarity too.
- Buzzer can be used as alarm or to play melodies.
- Buzzer symbol is as follow:



Diagram 2.2.(e): Various types of buzzer

There is a built-in buzzer on the Maker UNO. This buzzer is connected to pin 8 and can be turned on by a sliding switch. The buzzer needs to be activated by sliding the switch before it can be used.

Slider switch to activate buzzer

Buzzer

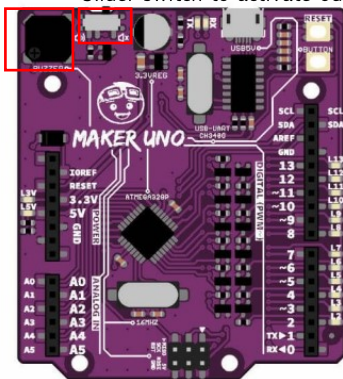


Diagram 2.2.(f): Buzzer and slider switch location on the buzzer

# UNIT 2.2

## Direct Current (DC) Motor

- There are different types of motor which has different voltage requirement such as 3v, 6v, 9v, 12v etc.
- The DC motor shaft will turn as current passes through it.
- In general, motor needs to be connected with a driver, or with a transistor and diode as it requires a high amount of current.
- Driver connected to the motor enables us to control the direction of the motor rotation—either clockwise or anti clockwise.
- If you are using the standard Maker UNO learning kit, there is a 3V DC motor included in the learning kit. Maker UNO cannot turn on any motor that requires a voltage of more than 5V.
- Motor needs to be used with a transistor because Arduino is not capable of sending out enough current to turn on a motor.
- The schematic symbol of a DC motor is as follow:



Diagram 2.2.(g): Different types of motor

## UNIT 2.3

# INTRODUCTION TO CONNECTING OUTPUT CIRCUIT

### Learning Objective

In this unit, SWBAT read schematic diagram and connect circuit based on the schematic diagram.

### Success Criteria:

Students will be able to connect at least 1 circuit based on the schematic diagram

---

We can connect output circuit by using jumper wires and breadboard. Breadboard are prototype board that allows us to test circuit by connecting jumper wires to it, all without using any soldering.

Before we begin, let us understand more about the following components:

### Jumper Wires

- Jumper wires are used to connect different components on / to the microcontroller.
- Unofficially, red jumper wires are used for connections going into power supply where as the black jumper wires are used for connections going into ground.

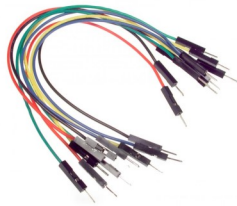
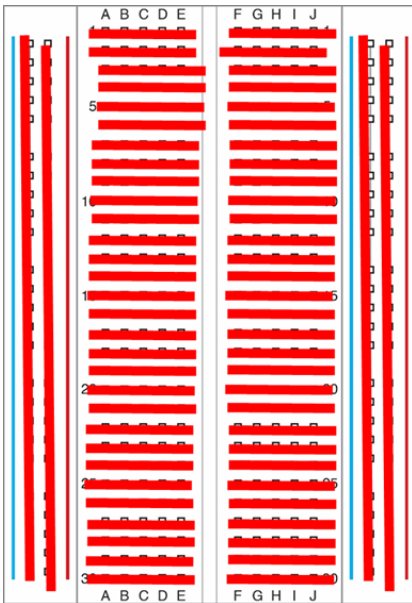


Diagram 2.3.(a): Jumper Wires

# UNIT 2.3

## Breadboard

- Breadboards are used to test and produce prototype circuit without the use of soldering.
- The holes are connected along the red lines as shown in Diagram 2.3(a). The components or wires will be connected if they are on the same line.



# UNIT 2.3

## Connecting Circuit onto Breadboard

### LED Connection

The following circuits show simple series and parallel circuit with the use of 3V battery and breadboard. The diagram shows the schematic diagram and the connection on the breadboard.

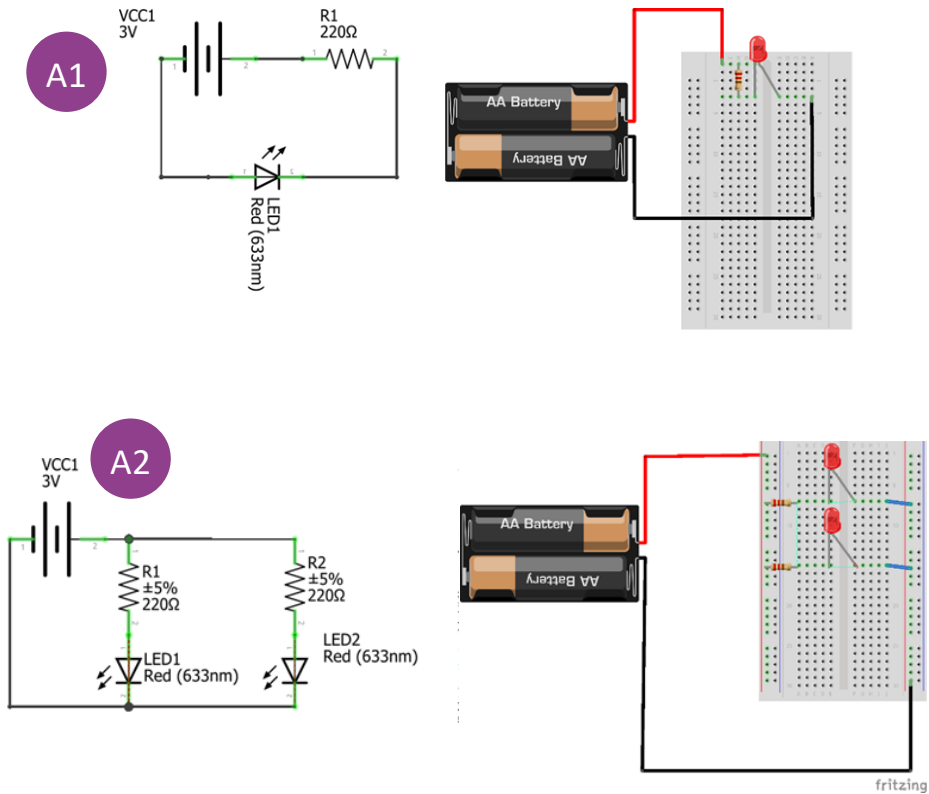
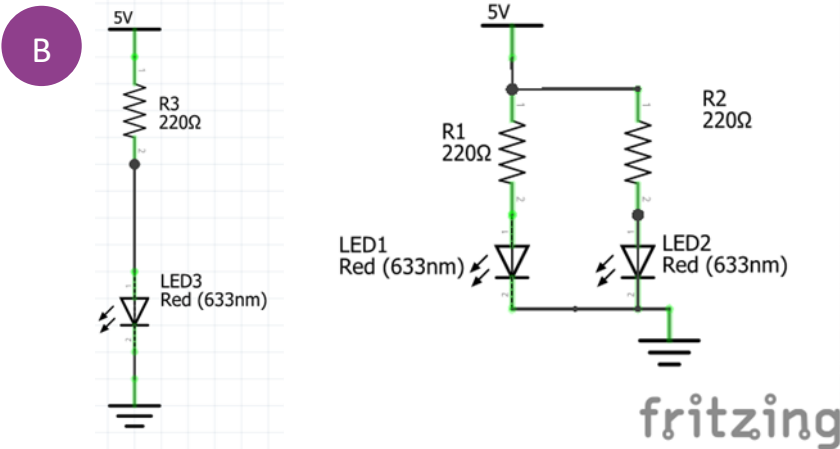


Diagram 2.2.(a): Circuit A1 dan A2  
(Top) Serial Circuit  
(Bottom) Parallel Circuit

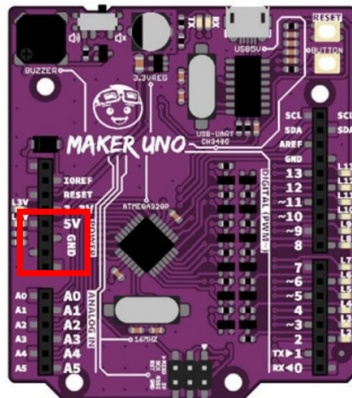


## UNIT 2.3

Let's replace the power source with a 5V power source:



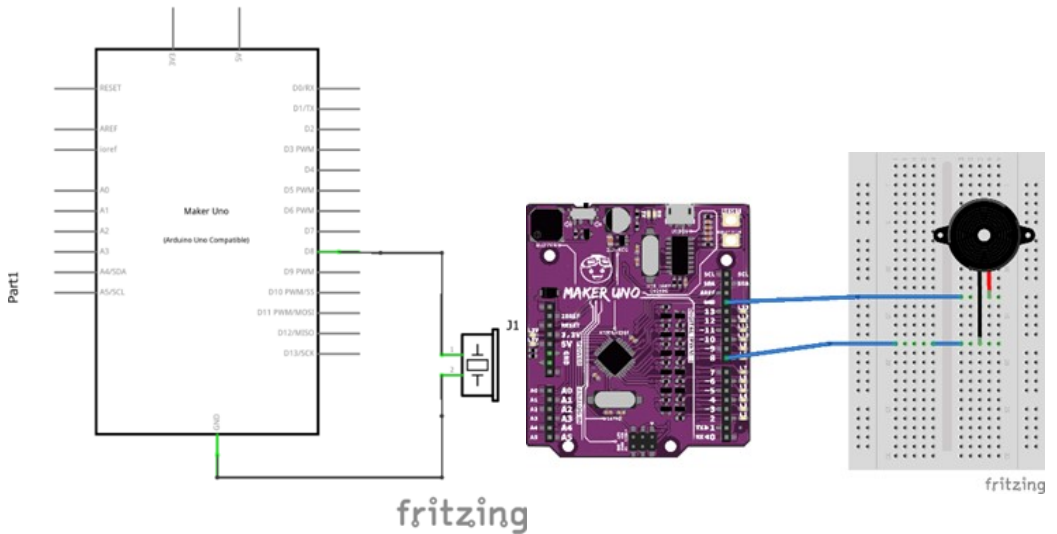
Arduino or Maker UNO is capable of providing 5V voltage like battery with its 5V pin (+) and GND (-) pin



# UNIT 2.3

## Buzzer

Buzzer needs to be connected according to its polarity similar like the LED (red to pin and black to ground). The following circuit is an example of buzzer connection to the Arduino. For Maker UNO, here is a built in buzzer located at pin 8. The buzzer needs to be activated by pushing the slider switch before usage.



# UNIT 2.3

## Programming for Buzzer:

```

1 void setup() {
2   // put your setup code here
3   pinMode(8, OUTPUT);
4 }
5
6 void loop() {
7   // put your main code here
8   tone(8, 165, 500);
9   delay(1000);
10  tone(8, 175, 500);
11  delay(1000);
12  tone(8, 196, 500);
13  delay(1000);
14 }
15
16

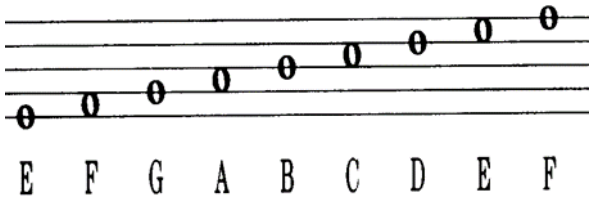
```

Line	Programming Statement	Function
1	tone (8, 165, 500)	<p>The buzzer will play sound at 165 Hz which is the E3 notes in music at pin 8 for 500 milliseconds.</p> <p><b>tone</b> is a special function that is used to control buzzer. It takes in 3 parameters that are pin numbers, sound frequencies to be played and the duration of the sound to be played.</p>

# UNIT 2.3

Tone is capable of producing musical notes. The following list shows a list of music notes (3rd octave) as well as its frequencies.

Music Notes	Frequencies
E	165
F	175
G	196
A	220
B	247
C	262
D	294
E	330
F	349

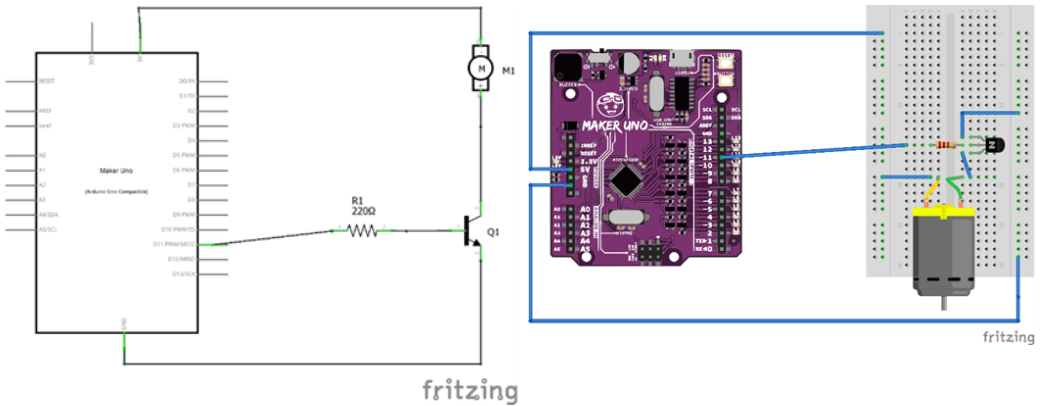


Visit <https://www.arduino.cc/en/Tutorial/ToneMelody?from=Tutorial.Tone> for a full list of music notes (Octave 1 to Octave 8)

# UNIT 2.3

## Direct Current (DC) Motor

DC Motor needs to be connected together with transistor—this is because motor requires a higher current to function and this current cannot be provided by the output pin on the Arduino. The circuit connection is as follows:



To overcome this problem, we need the transistor to function as a switch to control the motor. To turn on the motor, we will need to send the signals to pin 11 and then it will “open” the transistor to allow current to flow to the motor.

The programming to turn on the motor is as follows:

```
sketch_jan03a $
1 void setup() {
2   // put your setup code here,
3   pinMode(11, OUTPUT);
4 }
5
6 void loop() {
7   // put your main code here,
8   digitalWrite(11, HIGH);
9 }
```

## UNIT 2.4

# OUTPUT CIRCUIT SIMULATION

### Learning Objective

In this unit, SWBAT simulate output circuit using dedicated software.

### Success Criteria:

Students will be able to simulate at least 1 circuit along with the code simulation

We will be using Tinkerlab, a free software, to simulate output circuit

Weblink: [www.tinkerlab.com](http://www.tinkerlab.com)

### Steps to access Tinkerlab:

1) Register an account on Tinkercad

**TINKERCAD** FOR... FEATURES GALLERY COMMUNITY LEARN TEACH

SIGN IN **SIGN UP**

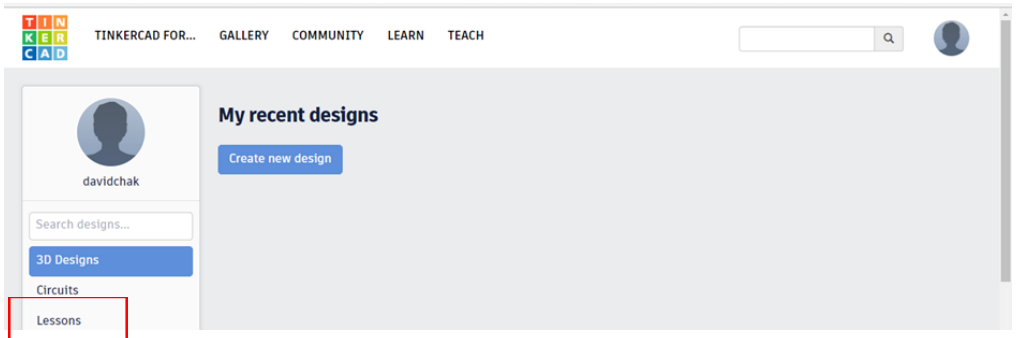
**Tinkercad is a simple, online 3D design and 3D printing app for everyone.**

Tinkercad is used by designers, hobbyists, teachers, and kids, to make toys, prototypes, home decor, Minecraft models, jewelry – the list is truly endless!

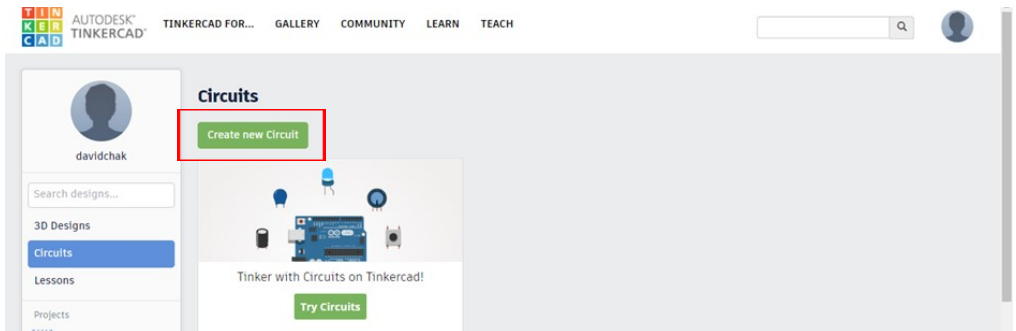
[Start Tinkering now](#)

# UNIT 2.4

2) Log in and click “Circuits”



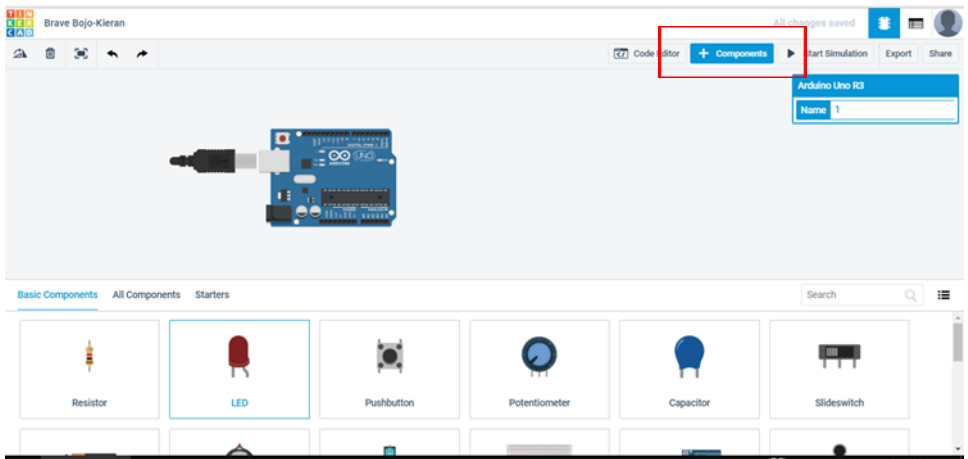
3) Click “Create New Circuit”



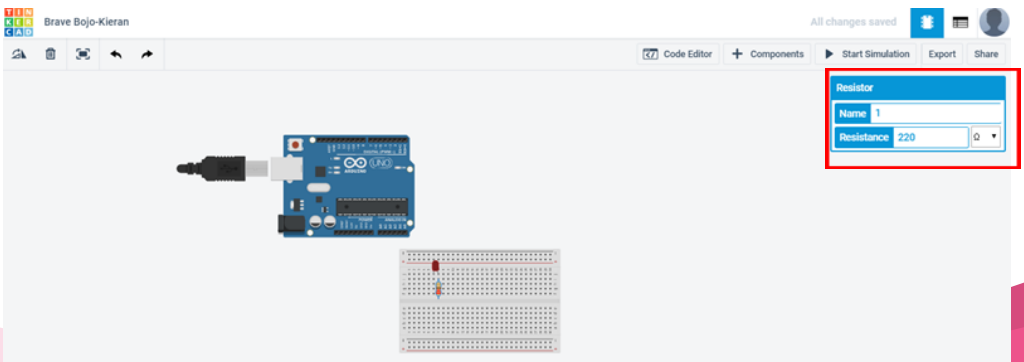
# UNIT 2.4

## Simulate circuit involving 1 LED connected to pin 5

- 1) Add LED, resistor, Arduino and breadboard by selecting “+ Components”



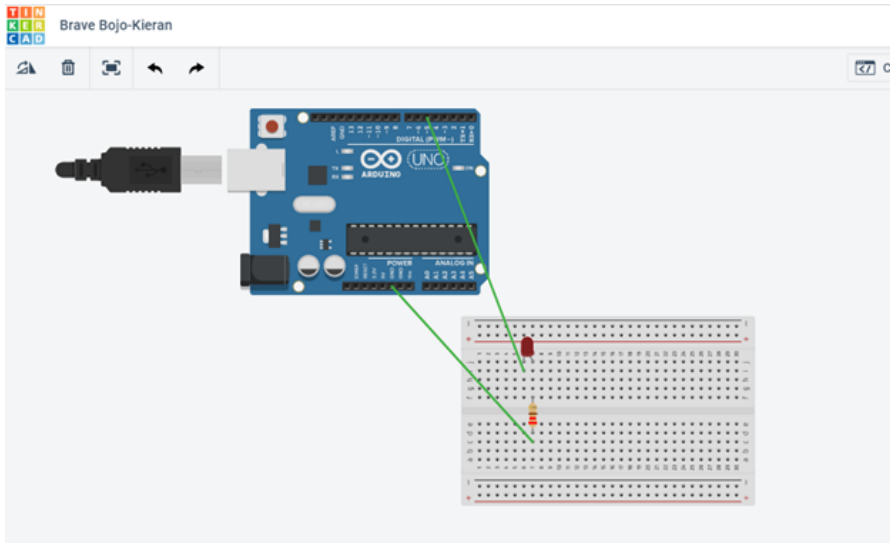
- 2) Make sure the resistor values are correct by clicking the resistor once and changing its resistance value.



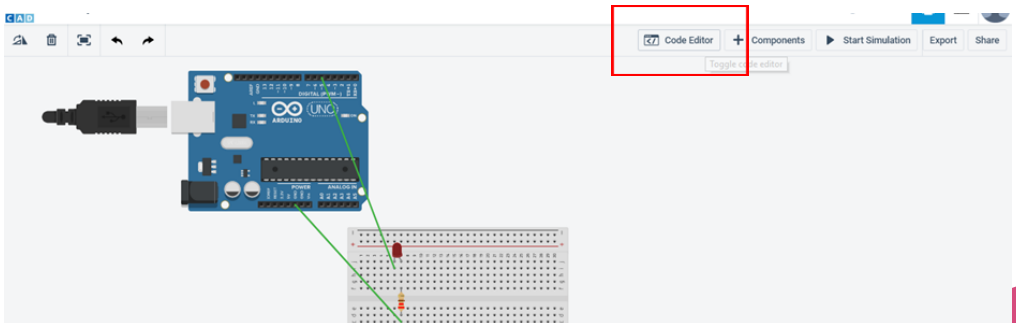


## UNIT 2.4

3) Connect devices on the breadboard with “drag and drop”



4) Write the code in “Code Editor”



## UNIT 2.4

5) Drag and drop the relevant blocks to create codes that will blink the LED at pin 5.

The screenshot shows the Arduino IDE interface. At the top, the user is logged in as 'Brave Bojo-Kieran' and 'All changes saved'. The main workspace displays a circuit diagram of an Arduino Uno R3 connected to a breadboard. A red LED is connected to pin 5 of the Arduino. Below the workspace, the 'Block' tab is active, showing a list of blocks on the left and a workspace on the right. The workspace contains three blocks: 'set LED to HIGH', 'set pin 5 to HIGH', and 'set pin 5 to HIGH'. The 'Upload & Run' button is visible. On the right, the 'Code Editor' tab is active, showing the following code:

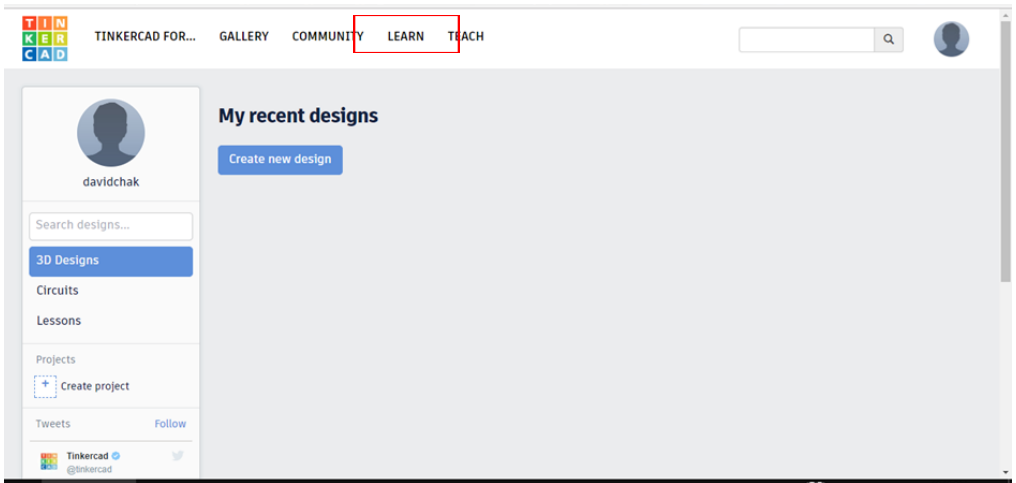
```
1 void setup()
2 {
3   pinMode(5, OUTPUT);
4 }
5
6 void loop()
7 {
8   digitalWrite(5, HIGH);
9   delay(1000); // Wait for 1000 millisecond(s)
10  digitalWrite(5, HIGH);
11  delay(1000); // Wait for 1000 millisecond(s)
12 }
```

6) Once the codes have been arranged, click “Upload and Run”. Try to simulate output circuit with more than 1 LED, buzzer or with a motor. For more detailed guide, you may refer to the video guide or step-by-step tutorials.

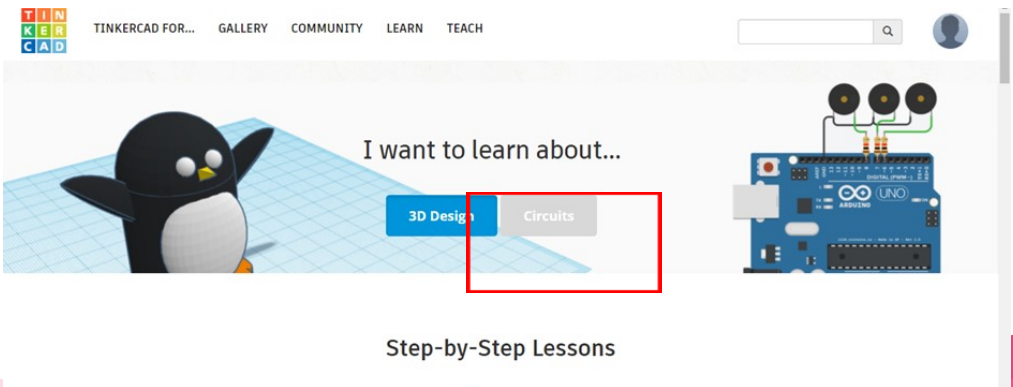
# UNIT 2.4

## Steps to access Tinkerlab tutorial

1) Log into your account and click “LEARN”

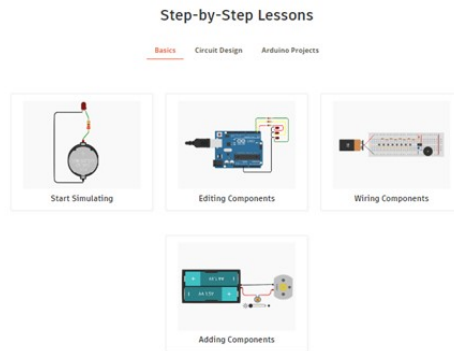


2) Click “Circuits”



# UNIT 2.4

3) You may follow the step-by-step lessons to learn how to use Tinkerlab or ways to use Arduino / other electronic components



3) You may also learn through video tutorial.



**Circuits Keyboard Shortcuts**

General shortcuts		Moving object(s)	
+ <b>C</b>	Copy object(s)	+ <b>M</b> or  + <b>V</b>	Move one unit
+ <b>X</b>	Cut object(s)	+ <b>M</b> + <b>↑</b> or <b>↓</b> or <b>←</b> or <b>→</b>	Move 1/20 unit
+ <b>V</b>	Paste object(s)	+ <b>R</b>	Rotate Clockwise
+ <b>Z</b>	Undo action(s)	+ <b>R</b> + <b>⇧</b>	Rotate Counterclockwise
+ <b>Y</b>	Re-do action(s)	Keyboard + mouse shortcuts (press and hold keyboard button, then move mouse)	
+ <b>A</b>	Deselect		
+ <b>I</b>	Invert selection	+ left mouse button	Select multiple object(s)

# UNIT 3

# INPUT

## LEARNING STANDARD

- 2.4.4 Build functioning simulated circuit with dedicated software.
- 2.4.5 Connect input and output circuit on the microcontroller
- 2.4.6 Write simple program based on input and output circuit

## ASSESSMENT STANDARD

TIER 4 Test out functionality of a circuit that includes microcontroller.

## SUB-UNIT

UNIT 3.1	Introduction to Input circuit programming	SP: 2.4.6
UNIT 3.2	Types of input devices	
UNIT 3.3	Introduction to connecting input circuit	SP: 2.4.5
UNIT 3.4	Input circuit simulation	

## RECOMMENDED T&L TIME ALLOCATION

UNIT 3.1	60 MINUTES
UNIT 3.2	30 MINUTES
UNIT 3.3	
UNIT 3.4	30 MINUTES

## PREPARATION

- 1) Ensure that there are enough Arduino / Maker UNO for each students.
- 2) Unit 3.1 requires the use of a computer or a smartphone to program the microcontroller. (Refers to Appendix 1)
- 3) Unit 3.4 requires computer and internet
- 4) Can be used along with Student Module

# UNIT 3.1

## INTRODUCTION TO INPUT CIRCUIT PROGRAMMING

### Learning Objective

In this unit, SWBAT write simple programming to control input circuit

### Success Criteria:

Students are able to write programming to read and display input values from the circuit.

Microcontroller can receive signals through its pins. Sensors that connected to the microcontroller can send digital and signals to the microcontroller

Digital signals have two types of values- 1 (On) or 0 (Off). All the number pins (3-13) can receive digital signal with the program command `digitalRead`. Examples of input devices that use digital signals are devices which has two states, such as switch (pressed / not pressed), rain sensor (has water / no water), etc.

Analog signals, on the other hand, contain more than two values, it ranges from 0 to 1023. Analog signals can only be read from the analog input pins (pin A0 to A5) by using the program command `analogRead`. Examples of analog output devices are sensors that sense a range of information, such as potentiometer (value changes with the knob turning), light sensors (very dark to very bright), microphone, etc.

Before we use any input devices, we need to set the input pins to be **INPUT** mode in the setup function:

```
sketch_jan07a$
```

```
1 void setup() {  
2   // put your setup code h  
3   pinMode(4, INPUT);  
4 }  
5
```

\*Replace 4 with any pin numbers used in the circuit.

## UNIT 3.1

There is a built-in switch at pin 2 for the Maker UNO. To use the pin, we will need to set the mode as `INPUT_PULLUP`

```
sketch_jan07a $  
1 void setup() {  
2   // put your setup code here,  
3   pinMode(2, INPUT_PULLUP);  
4 }  
5
```

To use the values, we need to use `analogRead` or `digitalRead` to read the value and then store it in a variable:

```
6 void loop() {  
7   // put your main code here,  
8   int x = digitalRead(2);  
9   int y = analogRead(A2);  
10 }
```

Line 8 reads digital signal from pin 2 and keep it in a variable x where as line 9 reads an analog signal from pin A2 and store it in a variable called y.

We can also open serial communication with the computer so we can retrieve and display the value read for debugging purpose. To do so, we need to begin the communication channel between the computer and Arduino with the code `Serial.begin(9600)` :

```
1 void setup() {  
2   // put your setup code  
3   pinMode(A0, INPUT);  
4   Serial.begin(9600);  
5 }
```

9600 is the data communication baudrate between the Arduino and the computer. The higher the value, the more information can be sent from Arduino to computer in a second. The typical baudrate used for Arduino-Computer communication is 9600.

## UNIT 3.1

Then, we can display the values from the input devices by using `Serial.println` (l is the small letter L)

```
1 void setup() {  
2  
3   pinMode(A0, INPUT);  
4   Serial.begin(9600);  
5 }  
6  
7 void loop() {  
8  
9   int x = analogRead(A2);  
10  Serial.println(x);  
11 }
```

Once the code has been uploaded, we can open Serial Monitor (CTRL+SHIFT+M) or press the magnifying glass symbol to see the values read by Arduino.





## UNIT 3.1

The following code reads the value from the built-in switch (for Maker UNO) and display the value. This example does not require any circuit on breadboard for Maker UNO, but it requires an external pushbutton connection for other Arduino models. The pushbutton would need to be connected to pin 2 and the the pin be set to `pinMode(2, INPUT)`.

sketch\_jan07a\$

```
1 void setup() {  
2     // put your setup code here,  
3     pinMode(2, INPUT_PULLUP);  
4     Serial.begin(9600);  
5 }  
6  
7 void loop() {  
8     // put your main code here,  
9     int x = digitalRead(2);  
10    Serial.println(x);  
11 }  
12  
13
```

# UNIT 3.2

## INPUT DEVICES

### Learning Objective

In this unit, SWBAT learn different types of input devices

### Success Criteria:

Students are able to list and state the name and function of input devices

Input devices will read value that is detected from the surrounding environment and send the information read through the pins on Arduino. All the pins can be used to read digital signals with the program command `digitalRead`.

Analog signals can only be detected by using the analog pins, which are pin A0 to A5. There are different input devices that can be used such as switch, potentiometer, light dependent resistors (light sensor), temperature sensor, human sensor (PIR Sensor), etc.

### Switch

- There are different types of switches such as microswitch, toggle switch, and pushbuttons.
- Although there are multiple types of switches on the market, all of them function with the same principle that is whether or not the circuit is complete.
- For more information about the different types of switch, , you may visit <https://learn.sparkfun.com/tutorials/switch-basics> or <https://electronicsclub.info/switches.htm> for more information
- Here are some of the more commonly seen switches:
- Pushbuttons
  - ⇒ The switch will be “ON” for as long as it is pressed. When it is not pressed, it will return to OFF state.



Diagram 3.1.(a): Pushbuttons

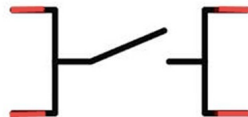


Diagram 3.1.(b): Schematic symbol for pushbutton

# UNIT 3.2

## Switch

- Toggle Switch  
⇒ SPST type - *Single Pole, Single Throw*



SPST has two terminals and the connection determines whether the internal is complete or incomplete. SPST connection requires pull-up / pull-down circuit to get a clearer reading. It will stay in ON or OFF mode based on the location of the knob.

- ⇒ SPDT type - *Single Pole, Double Throw*

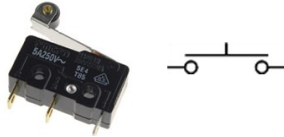


SPDT switch has 3 terminals—common, NO (Normally Open) and NC (Normal Close). SPDT switch do not require pull-up / pull-down circuit. It will stay in ON or OFF mode based on the knob location

# UNIT 3.2

## Switch

- Microswitch



Microswitch reacts to small movement or light pressure. For instance, the light switch in our refrigerator. When the fridge is open (pressure lifted off from the microswitch), the lights will be turned on.

## Potentiometer

- Potentiometer is a resistor that does not have a fixed resistance value. The resistance values can be changed by turning the knob.
- This resistance change can be read by Arduino via its analog input pins.



Diagram 3.1.(c): Potentiometer



Diagram 3.1.(d): Schematic symbol for potentiometer

# UNIT 3.2

## Light Sensor / Light Dependent Resistor

- The resistance value changes with the brightness of the surrounding environment—the brighter it is, the higher the value to be read by Arduino



Diagram 3.1.(e): Light Dependent Resistor

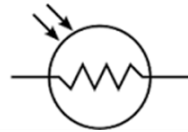


Diagram 3.1.(f): Schematic symbol for LDR

## Temperature Sensors

- There are different types of temperature sensor available in the market—the most common being LM35.
- LM35 produces input in the form of voltage read—we need to further process this input to get the temperature value measured in celcius.
- The calculation steps can be obtained here:  
<https://tutorial.cytron.io/2017/07/13/getting-started-temperature-sensor-celsius-sn-lm35dz/>

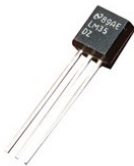


Diagram 3.1.(g): LM35

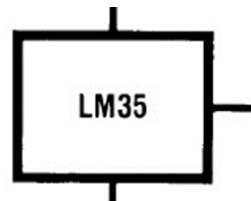


Diagram 3.1.(h): LM35 schematic symbol—this is an example of symbol for integrated circuit (IC)

# UNIT 3.2

## PIR Sensor

- Human movement can be detected by PIR Sensor (Passive Infrared) which detects infrared emitted by human.



Diagram 3.1.(e): PIR Sensor

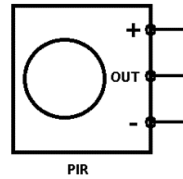


Diagram 3.1.(f): Schematic symbol for PIR Sensor

## UNIT 3.3

# INTRODUCTION TO INPUT CIRCUIT CONNECTION

### Learning Objective

In this unit, SWBAT read schematic diagram and connect circuit based on the schematic diagram.

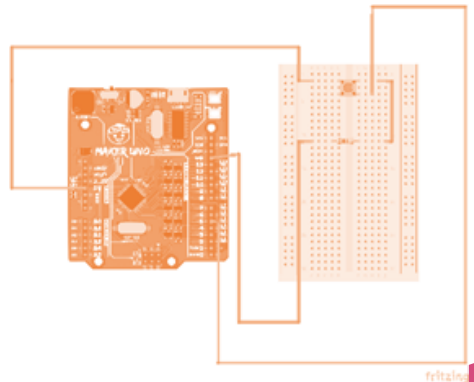
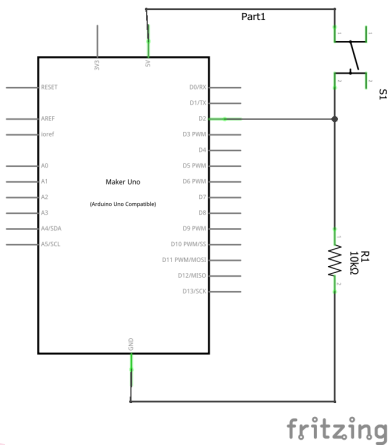
### Success Criteria:

Students are able to create at least 1 circuit based on schematic diagram

Like the output circuit, connecting input circuit uses jumper wires and breadboard as well.

### Pushbutton

- Maker UNO has a push button connected to pin 2. To use that, set the pinMode as **INPUT\_PULLUP** in setup.
- Here's the schematic and diagram to connect the pushbutton:



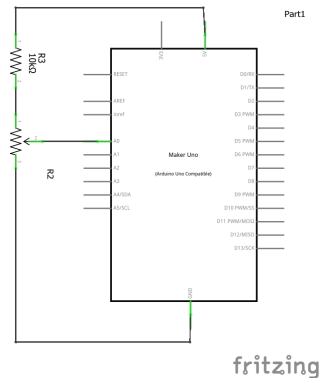
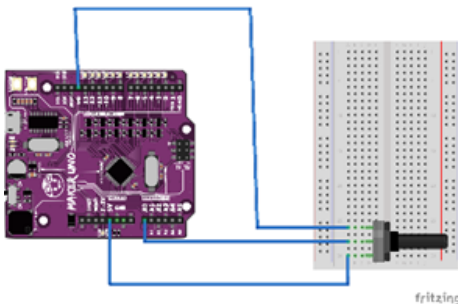
# UNIT 3.3

## Pushbutton

- We use **digitalRead** to read the value as this is a digital input device.
- 10k resistor is needed to create a pull down circuit. Pull down circuit ensures the accuracy of values read by pulling down all floating charges to the ground when the push button is not pressed.
- For more information on pull up and pull down circuit, visit <https://learn.sparkfun.com/tutorials/pull-up-resistors> dan [http://www.resistorguide.com/pull-up-resistor\\_pull-down-resistor/](http://www.resistorguide.com/pull-up-resistor_pull-down-resistor/)

## Potentiometer

- Potentiometer connection does not have any polarity because it is a resistor and resistor has no polarity.
- The left and right side of the potentiometer is connected to 5v or earth where as the middle pin is connected to the intended pin.
- Potentiometer is type of analog input device. For analog device, we need to use pin A0 to A5 and the program command **analogRead** to read its value

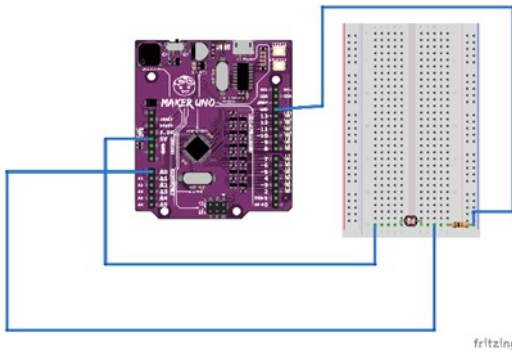




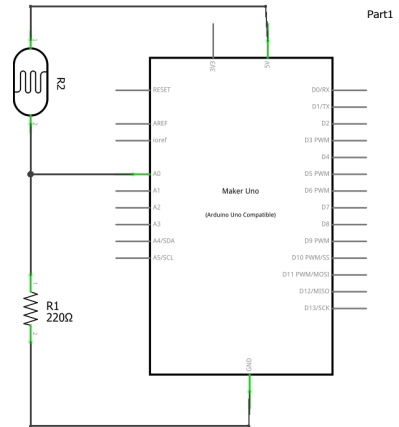
# UNIT 3.3

## Light Dependent Resistor

- Like potentiometer, light dependent resistor (LDR) has no polarity
- Similar to the connection of push button, LDR requires a pull down circuit to get a clean reading
- LDR is an analog input device, thus we connect it using pin A0-A5 and will be using the programming command `analogRead` to read its value.



fritzing

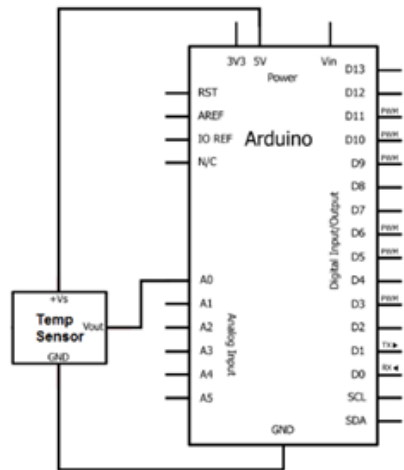
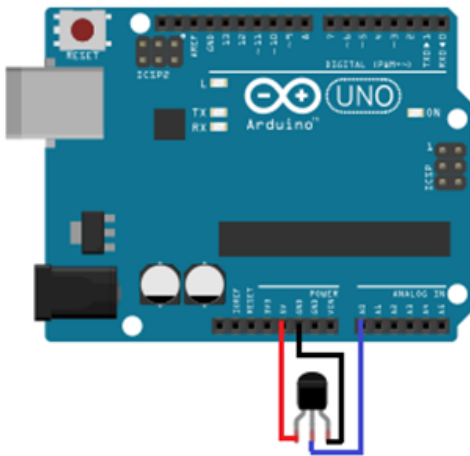


fritzing

# UNIT 3.3

## Temperature Sensor

- Temperature sensor LM35 connection is similar to potentiometer where the left and right terminal is connected to 5V and ground; the middle terminal to the pin.
- LM35 is an analog input device. Thus, we use pin A0-A5 and programming command `analogRead` to read its value.



## UNIT 3.4

# INPUT CIRCUIT SIMULATION

### Learning Objective

In this unit, SWBAT simulate input circuit on dedicated software.

### Success Criteria:

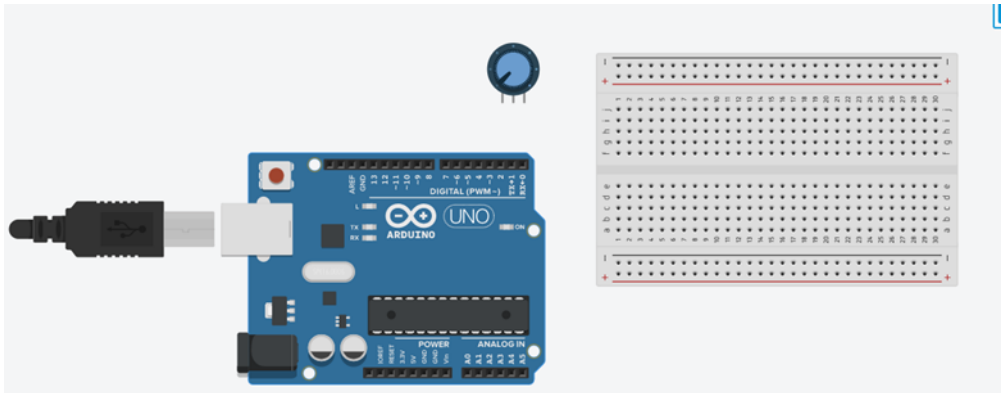
Students will be able to simulate at least 1 circuit simulation with programming.

We can use Tinkercad, a free web application to simulate input circuit

Web link for tinkercad: [www.tinkercad.com](http://www.tinkercad.com)

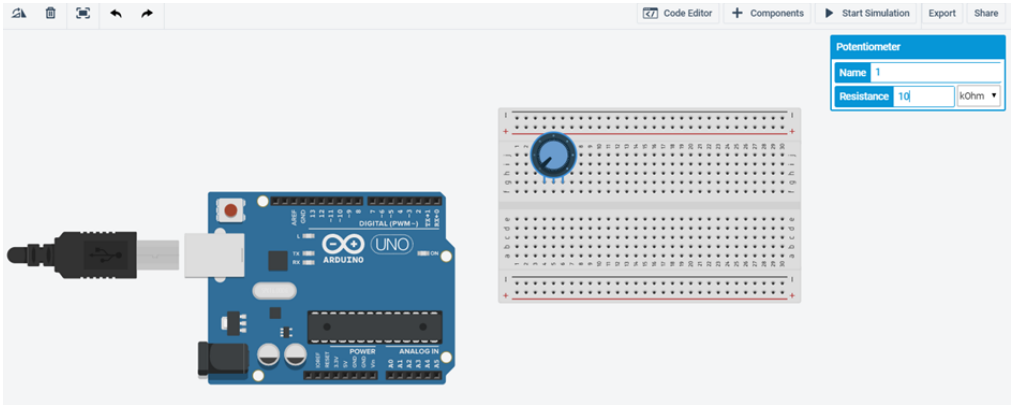
### Simulate potentiometer circuit

- 1) Add potentiometer, Arduino and breadboard by clicking on “+ Components”

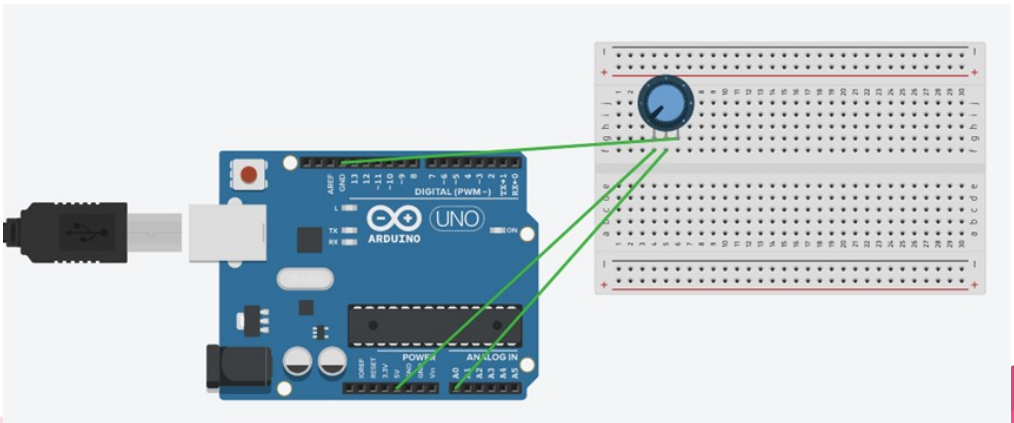


## UNIT 3.4

2) Make sure that the potentiometer resistance value is set correctly by clicking on it and changing its value to either 10k or 100k.

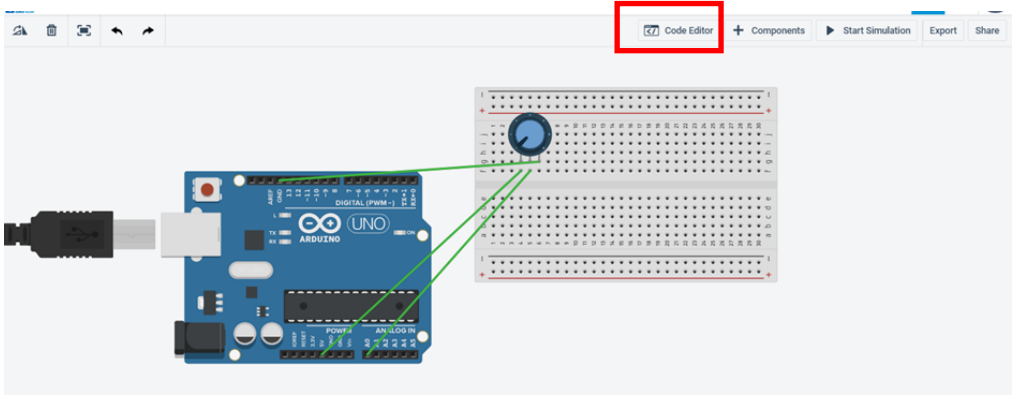


3) Drag and drop the components on the breadboard to connect all the components



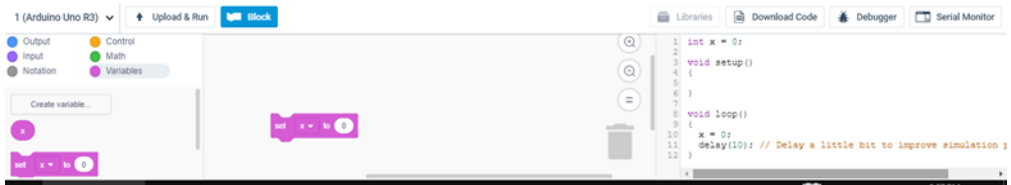
# UNIT 3.4

4) Write the programming in “Code Editor”



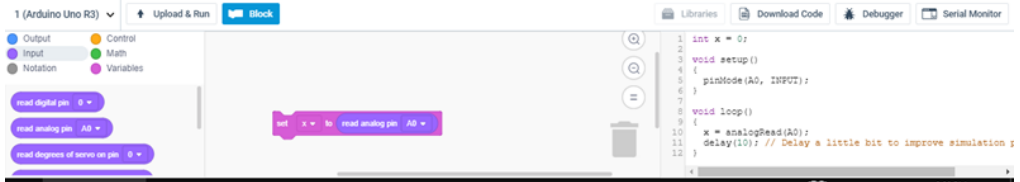
5) Drag and drop the relevant blocks to produce programming to read the value of potentiometer connected to A0 and displaying them

a) Create a new variable

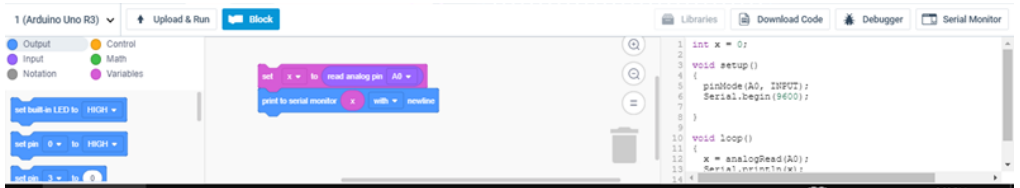


# UNIT 3.4

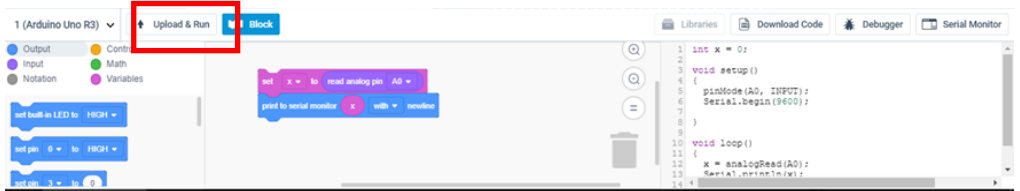
b) Set the value of the variable to values read from pin A0



c) Display the value read with Serial communication (combine Output blocks and Variables blocks)

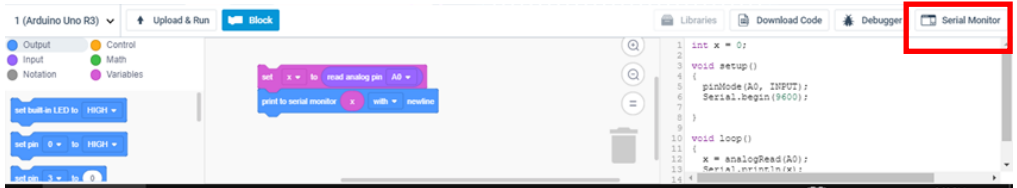


6) Click “Upload and Run” once the codes are arranged accordingly

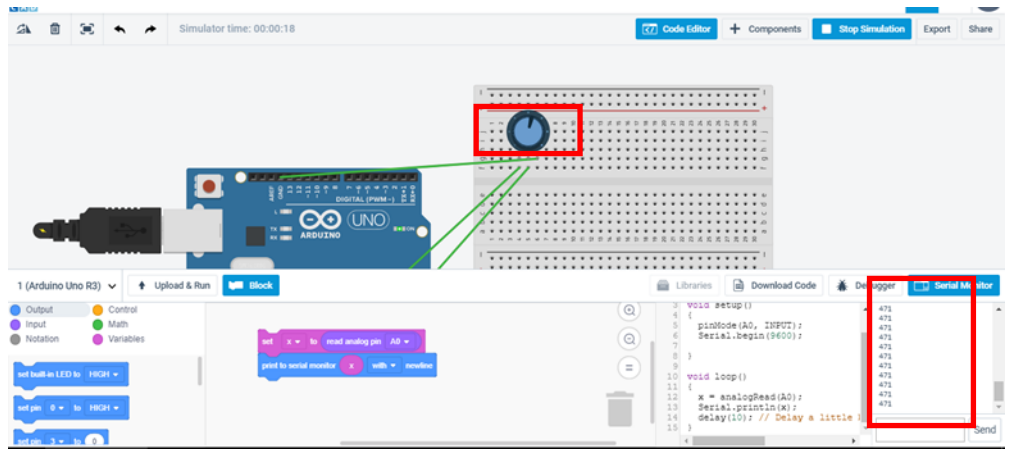


# UNIT 3.4

7) Serial monitor can be simulated by clicking on “Serial Monitor” button.



8) When we turn the knob, the values on the Serial Monitor will be changed as well.



# UNIT 4

## COMBINATION OF INPUT AND OUTPUT

### LEARNING STANDARD

2.4.4 Build functioning simulated circuit with dedicated software.

2.4.5 Connect input and output circuit on the microcontroller

2.4.6 Write simple program based on input and output circuit

2.4.7 Test and evaluate the function of the electronic circuit

2.4.8 Recommend improvement on electronic circuit.

### ASSESSMENT STANDARD

TIER 4 Test out functionality of a circuit that includes microcontroller.

TIER 5 Justify programming control structure for input and output to solve a problem.

TIER 6 Construct working microcontroller circuit.

### SUB-UNIT

UNIT 4.1	Introduction to conditional programming structure in Arduino	SP: 2.4.6
UNIT 4.2	Combine input and output circuit	SP: 2.4.7
UNIT 4.3	Simulate a combination of input and output circuit	SP 2.4.5
UNIT 4.4	Project Creation	SP: 2.4.7 / 2.4.8

### RECOMMENDED T&L TIME ALLOCATION

UNIT 4.1	60 MINUTES
UNIT 4.2	30 MINUTES
UNIT 4.3	
UNIT 4.4	90 MINUTES

### PREPARATION

- 1) Ensure that there are enough Arduino / Maker UNO for each students.
- 2) Unit 4.1 requires the use of a computer or a smartphone to program the microcontroller. (Refers to Appendix 1)
- 3) Unit 4.3 requires computer and internet
- 4) Can be used along with Student Module



## UNIT 4.1

# INTRODUCTION TO CONDITIONAL PROGRAMMING STRUCTURE

### Learning Objective

In this unit, SWBAT write programming that will produce different output based in the input received.

### Success Criteria:

Student will be able to turn on built in LED by pressing pushbuttons (built-in or connected externally)

We can combine input and output to create projects that will respond based on its interaction with the environment.

Recall that microcontroller functions as a brain to feed out different output (responses) based on the input (information) received.

We can receive information through input devices, and compare it with the desired value. If the value read is a match to the desired input, microcontroller can be programmed to produce the intended output. This is call a conditional programming structure. Conditional programming structure can be visualized in flowchart as follow:

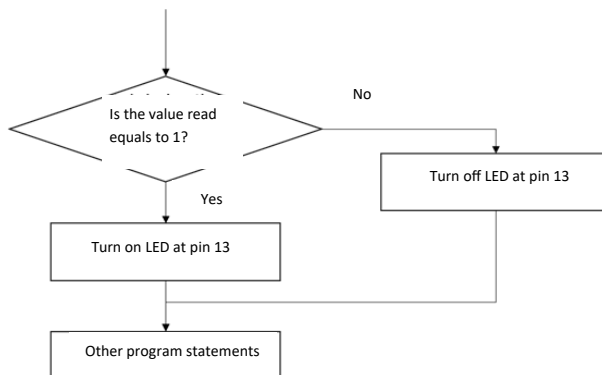


Diagram 4.1.(a): Flowchart for conditional programming structure

## UNIT 4.1

The program for the flowchart shown in Diagram 4.1(a) is as follow:

```
6 void loop() {  
7   // put your main code here,  
8   int x = digitalRead(2);  
9   if (x == 1){  
10    digitalWrite(13, HIGH);  
11  } else {  
12    digitalWrite(13, LOW);  
13  }  
14 }
```

Comparison operators are used to compare two different values and will produce the boolean result True if the comparison is valid. There are 6 comparison operators in Arduino programming:

Comparison Operators	Meaning	Example
==	Equals to	<pre>x = digitalRead(5); if (x == 1){     digitalWrite(13, HIGH); } else {     digitalWrite(13, LOW); }</pre> <p>Read the value x (pushbutton connected to pin 5). If x is 1 (push button is pressed), thus send high digital signal to pin 13 to turn on the LED. Or else, send low digital signal to pin 13 to turn off the LED.</p>

# UNIT 4.1

Comparison Operators	Meaning	Example
<b>!=</b>	Not equals to	<pre>x = digitalRead(5); if (x != 1){   digitalWrite(13, HIGH); } else {   digitalWrite(13, LOW); }</pre> <p>Read the value x (pushbutton connected to pin 5). If x is not 1 (push button is NOT pressed), thus send high digital signal to pin 13 to turn on the LED. Or else, send low digital signal to pin 13 to turn off the LED.</p>
<b>&gt;=</b>	Bigger than or equals to	<pre>x = analogRead(A0); if (x &gt;= 100){   digitalWrite(13, HIGH); } else {   digitalWrite(13, LOW); }</pre> <p>Read x (potentiometer connected to pin A0). If the value read is 100 or more than 100, send HIGH digital signal to turn on the LED at pin 13. Or else, send LOW digital signal to turn off the LED at pin 13.</p>

# UNIT 4.1

Comparison Operators	Meaning	Example
>	Bigger than	<pre>x = analogRead(A0); if (x &gt; 100){   digitalWrite(13, HIGH); } else {   digitalWrite(13, LOW); }</pre> <p>Read x (potentiometer connected to pin A0). If the value read is more than 100 (not including 100), send HIGH digital signal to turn on the LED at pin 13. Or else, send LOW digital signal to turn off the LED at pin 13.</p>
<=	Smaller than or equals to	<pre>x = analogRead(A0); if (x &lt;= 100){   digitalWrite(13, HIGH); } else {   digitalWrite(13, LOW); }</pre> <p>Read x (potentiometer connected to pin A0). If the value read is 100 or less than 100, send HIGH digital signal to turn on the LED at pin 13. Or else, send LOW digital signal to turn off the LED at pin 13.</p>

# UNIT 4.1

Comparison Operators	Meaning	Example
<	Smaller than	<pre>x = analogRead(A0); if (x &lt; 100){   digitalWrite(13, HIGH); } else {   digitalWrite(13, LOW); }</pre> <p>Read x (potentiometer connected to pin A0). If the value read is less than 100 (not including 100), send HIGH digital signal to turn on the LED at pin 13. Or else, send LOW digital signal to turn off the LED at pin 13.</p>

To check for value between two limits, we need to use the logic operator AND.

For example, the value between 500 to 700 (501->699) can be detected with the following codes:

```
6 void loop() {
7   // put your main code here,
8   int x = analogRead(A2);
9   if (x > 500 && x < 700) {
10    digitalWrite(13, HIGH);
11  } else {
12    digitalWrite(13, LOW);
13  }
14 }
```

(The symbol for AND is &&)

## UNIT 4.1

Example 1: (Only valid for Maker UNO)

The following codes read the pushbutton connected to pin 2 (built-in for Maker UNO), display that value, and if the pushbutton is pressed down, turn on the LED at pin 13. Or else, turn off the LED.

(For other Arduino, you will need to connect a push button to pin 2, and change the pinMode for pin 2 from `INPUT_PULLUP` ke `INPUT`)

```
sketch_jan07a$  
1 void setup() {  
2  
3     pinMode(2, INPUT_PULLUP);  
4     pinMode(13, OUTPUT);  
5     Serial.begin(9600);  
6 }  
7  
8 void loop() {  
9  
10    int x = digitalRead(2);  
11    Serial.println(x);  
12    if (x == 1){  
13        digitalWrite(13, HIGH);  
14    } else {  
15        digitalWrite(13, LOW);  
16    }  
17 }
```

# UNIT 4.1

Example 2: (Requires LDR connected to pin A0)

This program reads the value of light dependent resistor (LDR) connected to pin A0, display it and produce 2 kinds of output based on ambient light. If the surrounding environment is dark (value read is 400 or less than 400), the led at pin 13 will turn on. If the surrounding environment is bright (value read is more than 400), the led on pin 13 will be turned off.

```
sketch_jan07a$  
1 void setup() {  
2  
3     pinMode(A0, INPUT);  
4     pinMode(13, OUTPUT);  
5     Serial.begin(9600);  
6 }  
7  
8 void loop() {  
9  
10    int x = analogRead(A0);  
11    Serial.println(x);  
12    if(x <= 400){  
13        digitalWrite(13, HIGH);  
14    } else {  
15        digitalWrite(13, LOW);  
16    }  
17 }
```

## UNIT 4.2

# COMBINATION OF INPUT AND OUTPUT CIRCUIT.

### Learning Objective

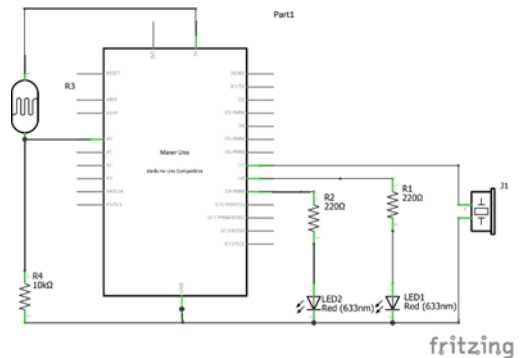
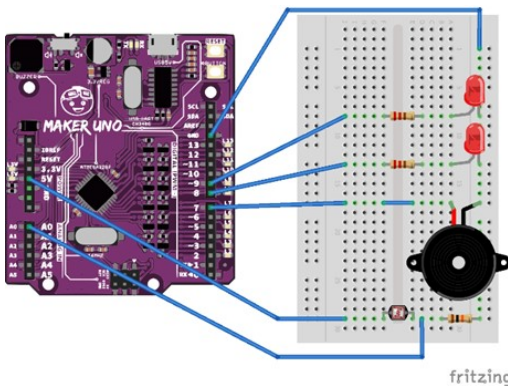
In this unit, SWBAT read schematic diagram and connect the circuit based on schematic diagram.

### Success Criteria:

Students can produce at least 1 circuit based on schematic given.

The combined circuit of input and output is no different than connecting them separately. Here are some examples of schematic that involves a combination of input and output devices.

### Circuit with LDR, Buzzer and LED

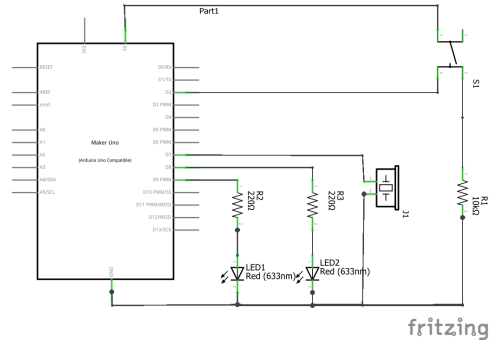
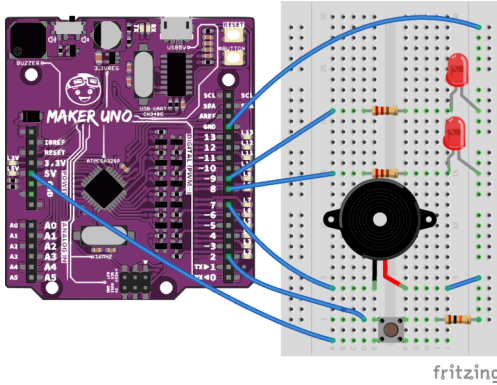


With this circuit we can create a simple music box—when the box is open (brightness increase), the microcontroller will play music and turn on the LEDs.



# UNIT 4.2

## Circuit with Pushbutton, LED and Buzzer



With this circuit, we can create a music that is activated by the pushbutton. When the box is open (pushbutton is no longer pressed), the microcontroller will play music and turn on the LED. Alternately, we can also write program that will turn on the music box by pressing the button.

## UNIT 4.3

# INPUT AND OUTPUT SIMULATION

### Learning Objective

In this unit, SWBAT simulate input and output circuit on dedicated software.

### Success Criteria:

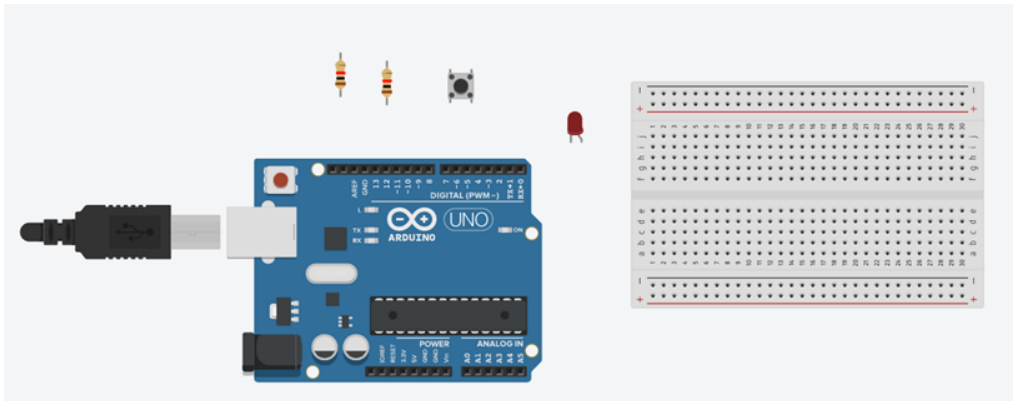
Students can simulate at least 1 circuit with its programming.

We can use Tinkercad, a free web application to simulate input circuit

Web link for tinkercad: [www.tinkercad.com](http://www.tinkercad.com)

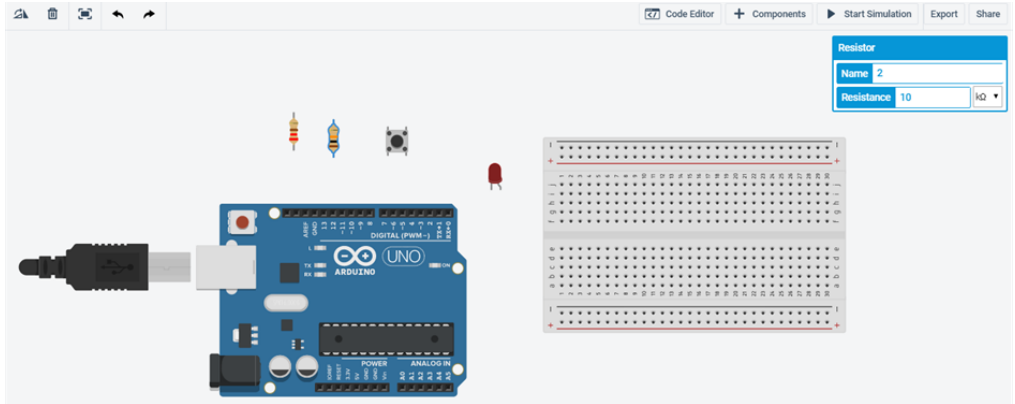
**Circuit that involves pushbutton and LED. When the pushbutton is pressed, the LED will turn on.**

1) Add in the pushbutton, LED, Arduino and breadboard by clicking on “+ Components”

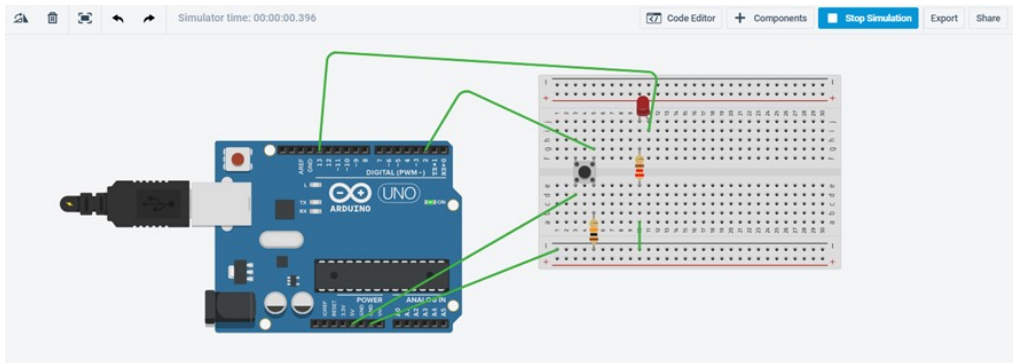


## UNIT 4.3

2) Make sure that the resistance value for the resistor is correct by clicking on the resistor and setting their values to 220 Ohm and 10k Ohm

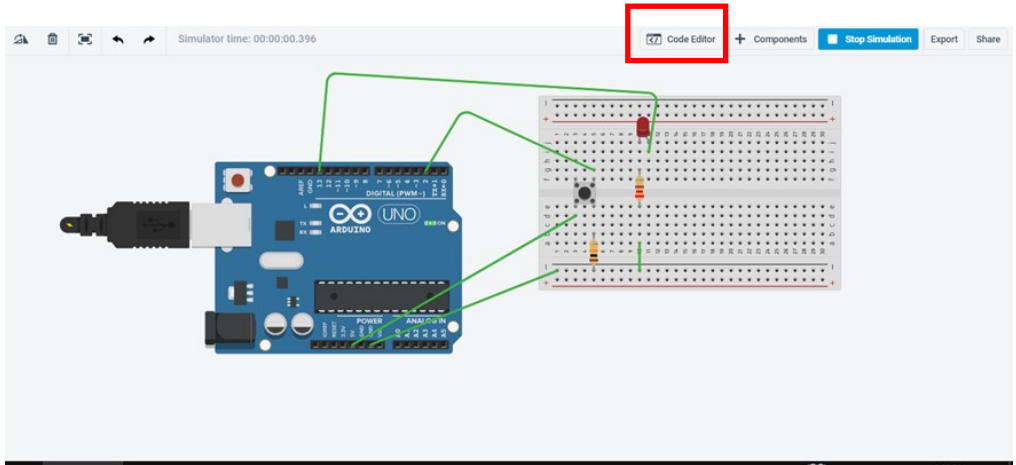


3) Drag and drop the components onto the breadboard to create the circuit



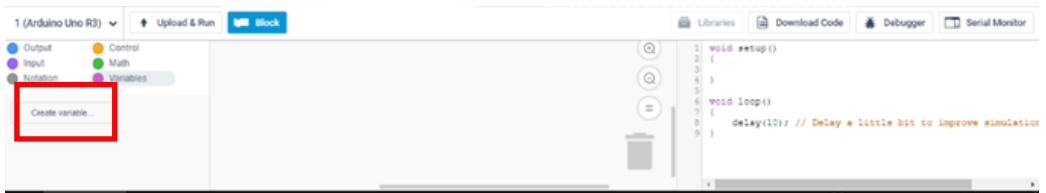
# UNIT 4.3

4) Program the circuit by clicking on “Code Editor”



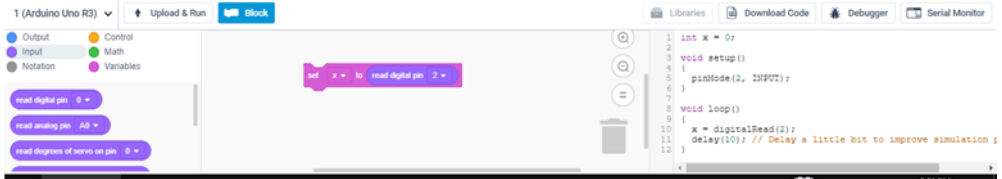
5) Drag and drop the relevant blocks to create the program

a) Create a new variable

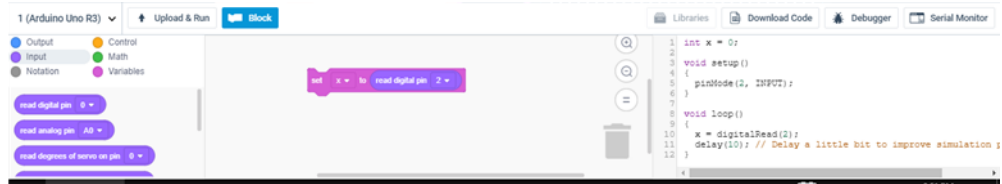


# UNIT 4.3

b) Set the value of the variable to value read from pin 2.



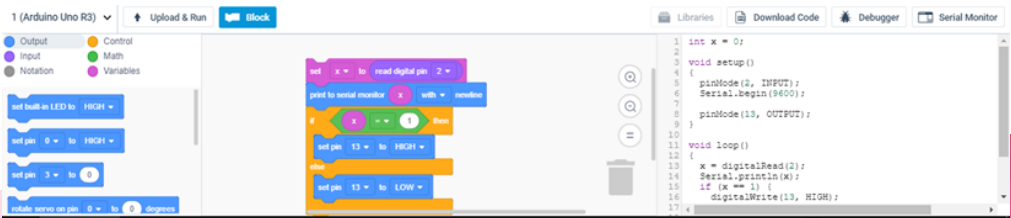
c) Display the value read with serial communication by combining output block and variable block.



d) Create the conditional control structure by selecting the “IF...THEN...ELSE” block from the control blocks. Then, use the Math and Variable blocks to create the following combination:



e) Drag the block to turn on and turn off the LED at pin 13 into the conditional control structure

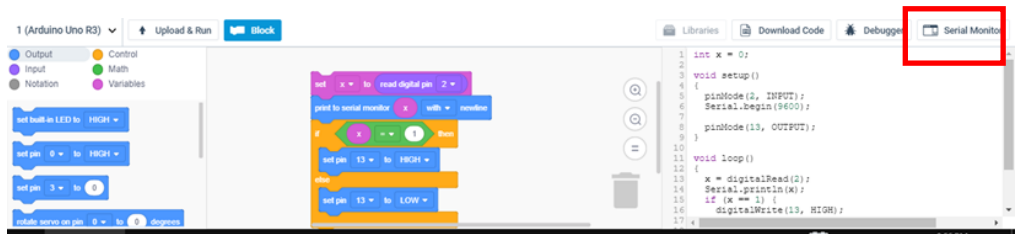


# UNIT 4.3

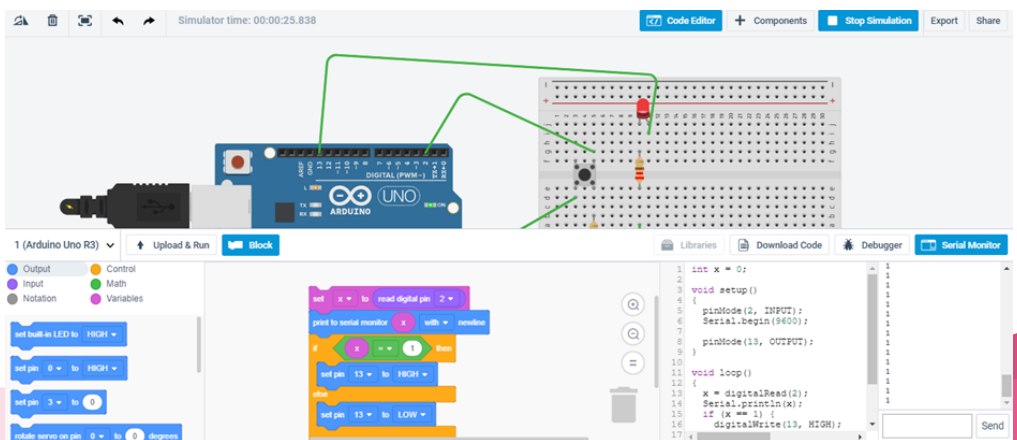
6) Click “Upload and Run”



7) Serial Monitor can be simulated by clicking on “Serial Monitor”



8) The LED will turn on when you click on the push button. The value in the Serial Monitor will be changed too.



## UNIT 4.4

# PROJECT CREATION

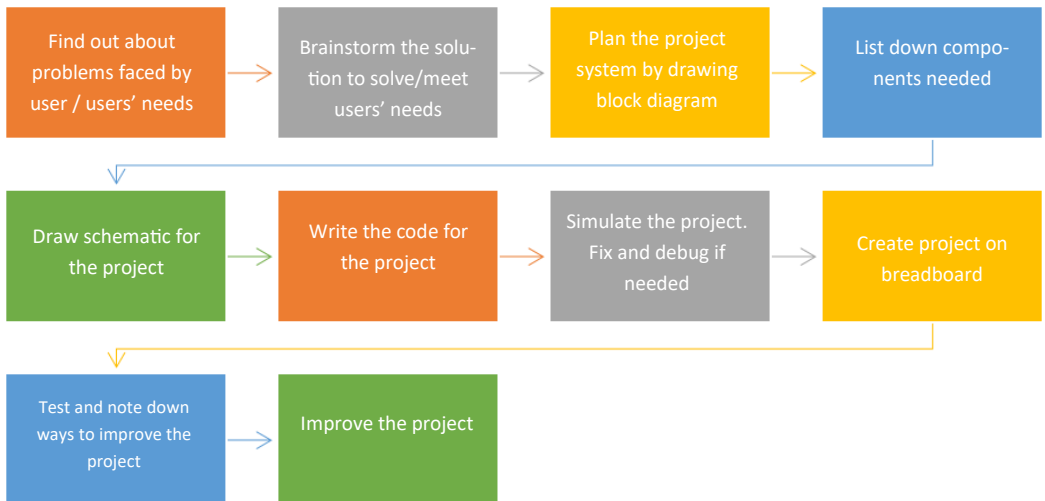
### Learning Objective

In this unit, SWBAT produce simple project that involves the use of input and output.

### Success Criteria:

Student can create project that uses input and output, simulate and document the project creation.

Here are the step-by-step guide to brainstorm, plan, simulate, and create a project:



# Appendix I

## PROGRAMMING ARDUINO WITH ANDROID SMARTPHONES

Step-by-step:

1. Determine whether your phone is capable of On-The-Go (OTG) function
  - Search on internet, phone settings, or use the USB OTG CHECKER (downloadable from Google Playstore<sup>1</sup> to check if your phone is capable of OTG function
  - Activate OTG if needed
2. Download Arduino Droid<sup>2</sup> from Google Playstore
3. Write the intended program
4. Click on lightning button to compile.
5. Select *Settings*→ *Board Type* → *Arduino*  
→ *Uno CH340G (Maker UNO) OR Uno (Arduino)*.



6. Connect the microcontroller to the smartphone by using USB and OTG cable.
7. Click on the upload icon to send the Arduino program to the microcontroller.



<sup>1</sup><https://play.google.com/store/apps/details?id=com.app.usbotgchecker&hl=en>

<sup>2</sup><https://play.google.com/store/apps/details?id=name.antonsmirnov.android.arduinoroid2&hl=en>



# Appendix 2

## Reference / Additional Information

---

### **For more information about Arduino:**

- <https://www.arduino.cc/en/Guide/HomePage>
- <https://www.arduino.cc/en/Tutorial/HomePage>
- <http://forefront.io/a/beginners-guide-to-arduino/>
- <https://www.makerspaces.com/arduino-uno-tutorial-beginners/>

### **Check out our video tutorials:**

- [www.arusacademy.org.my](http://www.arusacademy.org.my)

### **To get some ideas on what Arduino project is possible**

- <https://www.makerspaces.com/15-simple-arduino-uno-breadboard-projects/>
- <http://www.instructables.com/id/Arduino-Projects/>

### **Draw schematic by using the Fritzing software**

- <http://fritzing.org/home/>