FLINT REGIONAL SCIENCE & ENGINEERING FAIR

Measurements, Sensors and Data Logging Course

Week 2

Upcoming Weeks

- Office Hours
  - Monday Jan 25 @ 7:00 PM
  - Monday Feb 1 @ 7:00 PM

- Weekly Session
  - Thursday Jan 28 @ 7:00 PM
  - Thursday Feb 4 @ 7:00 PM

Lesson 2: Button

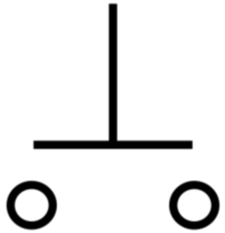
Use a pushbutton to change the state of the LED

Pushbutton Introduction

Lesson 2: Button

- What is a pushbutton?
  - A pushbutton is a momentary type of switch used as an input to an electrical system. When **closed** it allows an electrical current to flow through it. When **open** it prevents electrical current flow through it.
- Where are pushbuttons used?
  - Pushbuttons are used in many devices, from keyboards, cell phones, alarm clocks, industrial equipment, home appliances and much, much more.
  - Activity: find a device not listed above that uses a pushbutton.

Pushbutton Symbol



By Michel Bakni - Derived from files [1] and [2].(in English) (1993) 315-1975 - IEEE Standard American National Standard Canadian Standard Graphic Symbols for Electrical and Electronics Diagrams (Including Reference Designation Letters), IEEE, p. 59 DOI: 10.1109/IEEESTD.1993.93397. ISBN: 0738109479., CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=94264073

Example Pushbutton

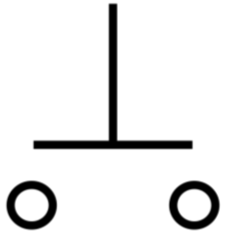
Image from https://www.digikey.com/en/products/detail/e-switch/TL2230EEF140/4029358

Pushbutton Introduction

Lesson 2: Button

- Recap
  - What are the Arduino read and write functions?
    - ____________ for reading an analog input
    - ____________ for reading a digital input
    - ____________ for a digital output
  - Pushbutton states (open, closed)
    - _________ completes the connection and allows current to flow
    - _________ does not allow current to flow

Pushbutton Symbol



By Michel Bakni - Derived from files [1] and [2].(in English) (1993) 315-1975 - IEEE Standard American National Standard Canadian Standard Graphic Symbols for Electrical and Electronics Diagrams (Including Reference Designation Letters), IEEE, p. 59 DOI: 10.1109/IEEESTD.1993.93397. ISBN: 0738109479., CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=94264073

Example Pushbutton

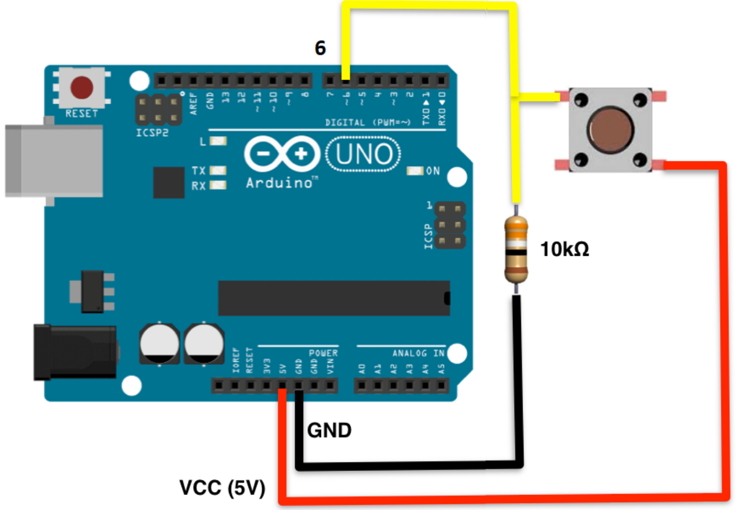
Image from https://www.digikey.com/en/products/detail/e-switch/TL2230EEF140/4029358

Pushbutton Introduction

Lesson 2: Button

- How do I use a pushbutton?
  - Follow the connection diagram to the right.
    - Grove Beginner’s Kit has already done this for you.
  - We then read the state of the input using the digitalRead function.

Example Pushbutton Connection



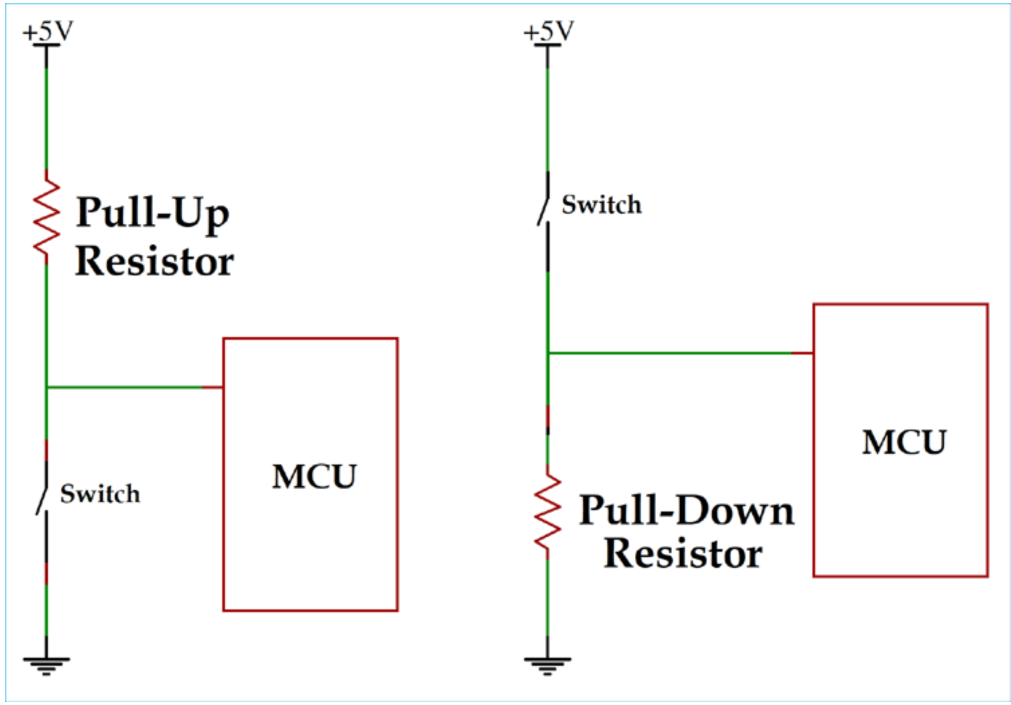
Modified from https://sites.google.com/site/ardunitydoc/getting-started/run-examples/push-button

- More Info:
  - https://www.allaboutcircuits.com/textbook/digital/chpt-4/switch-types/
  - https://en.wikipedia.org/wiki/Push-button

Pushbutton Introduction

Lesson 2: Button

- What happens if the circuit isn't completed?
  - If the circuit is not completed (to GND or +5V), the input will "float"
  - To prevent the value from floating, we use a resistor to pull up or pull down the input.
    - 10K Ohm is a common pull up / down resistor value
    - **Pull Up** = Resistor to Vcc (+5V)
    - **Pull Down** = Resistor to GND

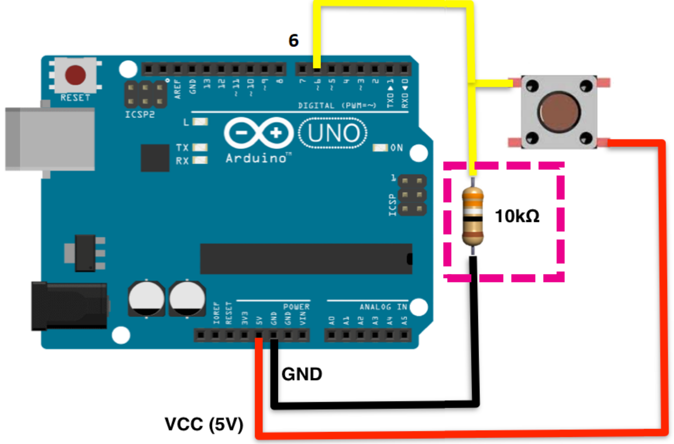


Pullup and Pulldown Resistors

Lesson 2: Button

- To prevent the value from floating, we use a resistor to pull up or pull down the input.
  - 10K Ohm is a common pull up / down resistor value
  - **Pull Up** = Resistor to Vcc (+5V)
  - **Pull Down** = Resistor to GND

Example Pushbutton Connection



Modified from https://sites.google.com/site/ardunitydoc/getting-started/run-examples/push-button

- More Info:
  - https://learn.sparkfun.com/tutorials/pull-up-resistors/all

Lesson 2 Hardware

Lesson 2: Button

- We can use the MCU on our Arduino to receive a digital reading of the state of a pushbutton.
  - Receiving a **HIGH** signal means the button is **pressed**.
  - Receiving a **LOW** signal means the button is **released**.
- What hardware will we need for this Lesson?
  - Grove LED Module on pin D4
  - Grove Button Module on pin D6
  - Seeeduino Lotus (Arduino Uno compatible board)

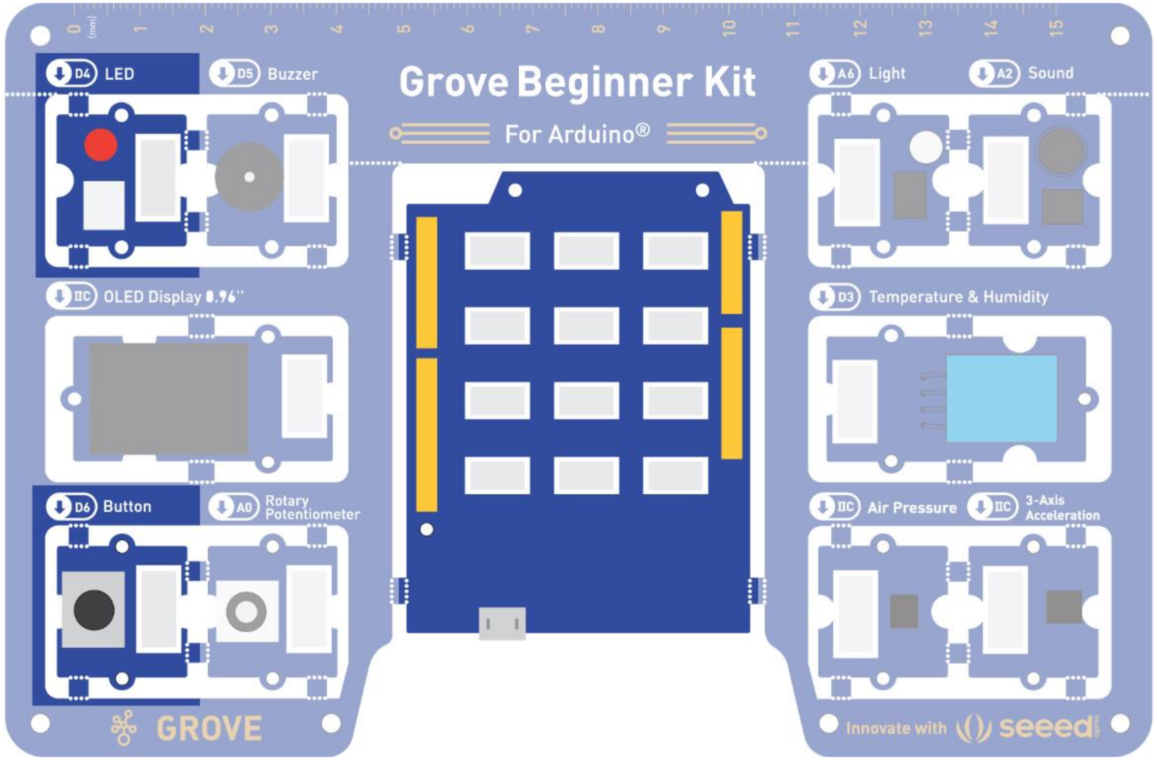


Image from https://files.seeedstudio.com/wiki/Grove-Beginner-Kit-For-Arduino/res/Grove-Beginner-Kit-For-ArduinoPDF.pdf

Open and Upload Sketch

Lesson 2: Button

1. Open Button Sketch
   a. **File → Sketchbook → CrashCourse_Jan → L2_Button**
2. Verify the sketch by clicking the Verify Button.
   a. The sketch should compile with no errors.
3. Upload the sketch to your Arduino by clicking the Upload Button.
   a. The sketch should re-compile, and then upload to your Arduino.
4. Watch the LED as you press the button.

Code Analysis – pinMode Function

Lesson 2: Button

```
pinMode(buttonPin, INPUT);
```

- Configures the buttonPin as an input.

- The pinMode function configures the specified pin to behave either as an input or an output.
- Syntax:

```
pinMode(pin, mode);
```

  - Pin: Arduino pin number to set the mode of
  - Mode: options are
    - `INPUT`, set pin as an input
    - `OUTPUT`, set pin as an output
    - `INPUT_PULLUP`, set pin as an input and enable a weak internal pullup resistor.
- More information:
  - https://www.arduino.cc/reference/en/language/functions/digital-io/pinmode/

Code Analysis – Variable Assignment

Lesson 2: Button

```
buttonValue = digitalRead(buttonPin);
```

- Assigns the state of read buttonPin to buttonValue.

- The assignment operator (**=**) puts whatever is on the right side of the equal sign into the variable on the left side.
- Syntax:

```
    variable = value;
```

- Variable: stores the value of the statement on the right side of the equals sign.
- Value: statement, function or equation whose value is to be stored in variable.

- More information:
  - https://www.arduino.cc/en/Reference/VariableDeclaration

Code Analysis – if…else if…else Conditionals

Lesson 2: Button

- The `if` statement checks a condition and executes the proceeding statement(s) if the condition is TRUE.
- The `else` statement executes if the previous `if` conditional evaluated as FALSE. The `else` statement is optional.
- The `else if` statement combines the `else` statement with the `if` statement. The `else if` statement is optional.
- Syntax:

```
if(condition1)
{
   // do this
}
else if(condition2) // OPTIONAL
{
   // do that
}
else // OPTIONAL
{
   // do something else
}
```

  – `conditionX` must evaluate to TRUE (not 0) or FALSE (0)
- More information:
  – https://www.arduino.cc/reference/en/language/structure/control-structure/if/
  – https://www.arduino.cc/reference/en/language/structure/control-structure/else/

```
if(buttonValue == HIGH)
{
  digitalWrite(ledPin, HIGH);
}
else
{
  digitalWrite(ledPin, LOW);
}
```

Example of an if…else conditional

Button Activities

Lesson 2: Button

- Activity 1
  - Change the LED to turn OFF if the button is pressed and turn ON when the button is released.
- Activity 2 (Bonus / Homework)
  - Keep the same function as the original W1L2_Button.ino sketch, without using a conditional statement.
- Activity 3 (Bonus / Homework)
  - Toggle the LED every time the button is pressed.

Lesson 3: Potentiometer

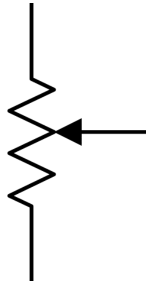
Use a potentiometer to change the brightness of the LED

Potentiometer Introduction

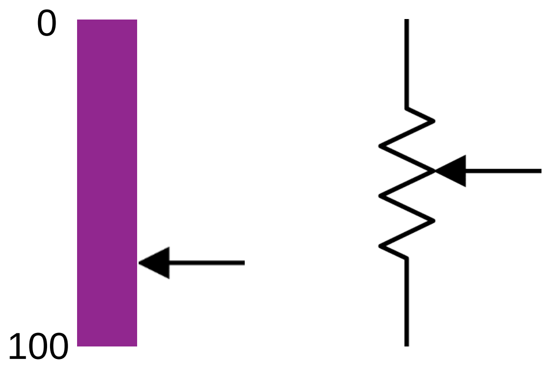
Lesson 3: Pot

- What is a potentiometer (pot)?
  - A pot is a type of variable resistor that has 3 terminals, two end terminals and a moveable wiper terminal.
  - Commonly used as position sensors.

Potentiometer Symbol



By Potentiometer with load.png: OmegatronDerivative work: DesbWit - This file was derived from: Potentiometer with load.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=20737313



Potentiometer Introduction

Lesson 3: Pot

- Where are pots used?
  - Pots are used in many devices, from volume knobs, industrial equipment, servos, home appliances, vehicles, and much, much more.
  - Activity: find a specific device that uses a pot.

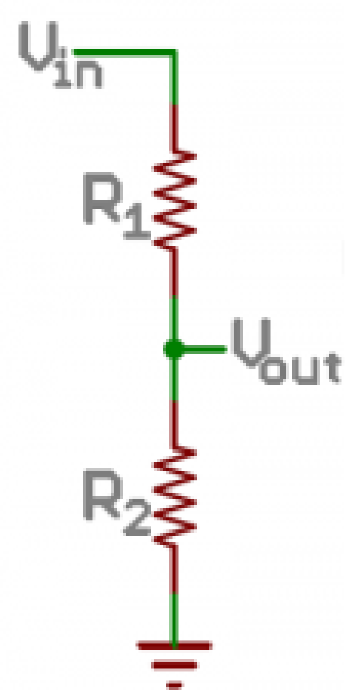
Example Potentiometers

By Junkyardsparkle - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=39291275

Voltage Divider

Lesson 3: Pot

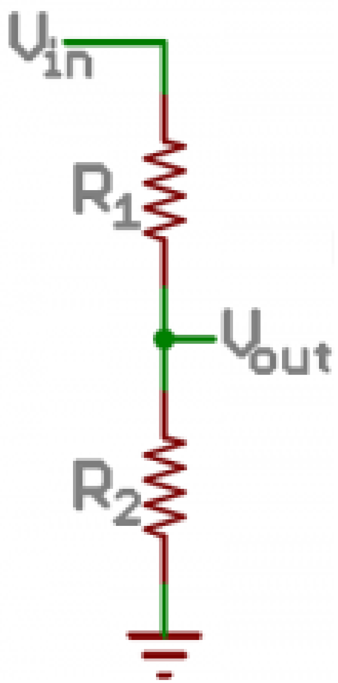
- What is a voltage divider?
  - Simple circuit which turns a large voltage into a smaller one.
    - **Vout = Vin** $* \frac{\boldsymbol{R2}}{\boldsymbol{R1+R2}}$
    - Vin = 5V, R1 = 50, R2 = 50
      - Vout = $5\mathrm{V} * \frac{50}{50+50}$ = 2.5V
    - Vin = 5V, R1 = 20, R2 = 80
      - Vout = $5\mathrm{V} * \frac{80}{20+80}$ = 4V
  - More Info:
    - https://learn.sparkfun.com/tutorials/voltage-dividers/all#:~:text=A%20voltage%20divider%20is%20a,most%20fundamental%20circuits%20in%20electronics



Voltage Divider

Lesson 3: Pot

- What is a voltage divider?
  - Simple circuit which turns a large voltage into a smaller one.
    - **Vout = Vin** $* \frac{\boldsymbol{R2}}{\boldsymbol{R1+R2}}$
    - Vin = 5V, R1 = 10, R2 = 90
      - Vout = ?
    - Vin = 5V, R1 = 180, R2 = 20
      - Vout = ?

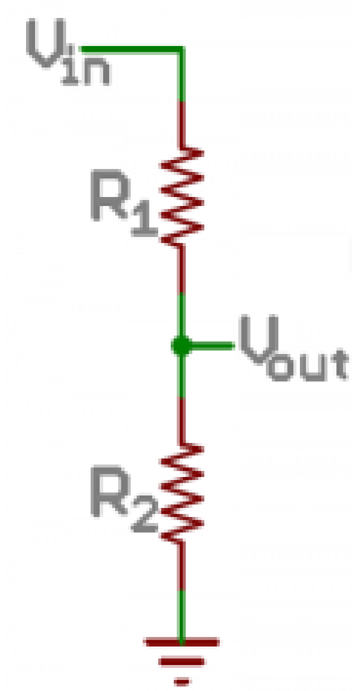


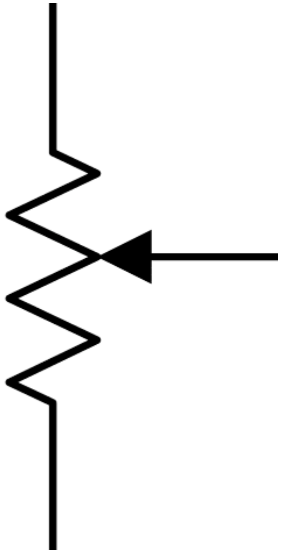
  - More Info:
    - https://learn.sparkfun.com/tutorials/voltage-dividers/all#:~:text=A%20voltage%20divider%20is%20a,most%20fundamental%20circuits%20in%20electronics

Potentiometer Introduction

Lesson 3: Pot

- Potentiometers are commonly used as adjustable voltage dividers.





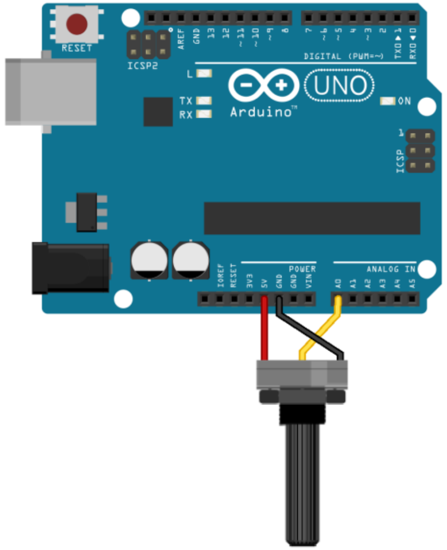
Potentiometer Introduction

Lesson 3: Pot

- How do I use a potentiometer?
  - Follow the connection diagram to the right. Your Grove Beginner’s Kit has already done this for you.
  - We then read the state of the input using the analogRead function.

- More Info:
  - https://en.wikipedia.org/wiki/Potentiometer
  - https://www.allaboutcircuits.com/textbook/direct-current/chpt-6/voltage-divider-circuits/

Example Potentiometer Connection



Modified from https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogInput

Potentiometer Introduction

Lesson 3: Pot

- How do I use a potentiometer?
  - Follow the connection diagram to the right. Your Grove Beginner's Kit has already done this for you.
  - We then read the state of the input using the analogRead function.

- More Info:
  - https://en.wikipedia.org/wiki/Potentiometer
  - https://www.allaboutcircuits.com/textbook/direct-current/chpt-6/voltage-divider-circuits/

Combining Analog, PWM, and LEDs

Lesson 3: Pot

- We can use the MCU on our Arduino to read the value of the pot and output a PWM signal to the LED to control the brightness.
  - Outputting a higher value is a larger duty cycle which means a brighter LED.
- What hardware will we need for this Lesson?
  - Grove LED Module on pin D4
  - Grove Rotary Potentiometer Module on pin A0
  - Seeeduino Lotus (Arduino Uno compatible board)

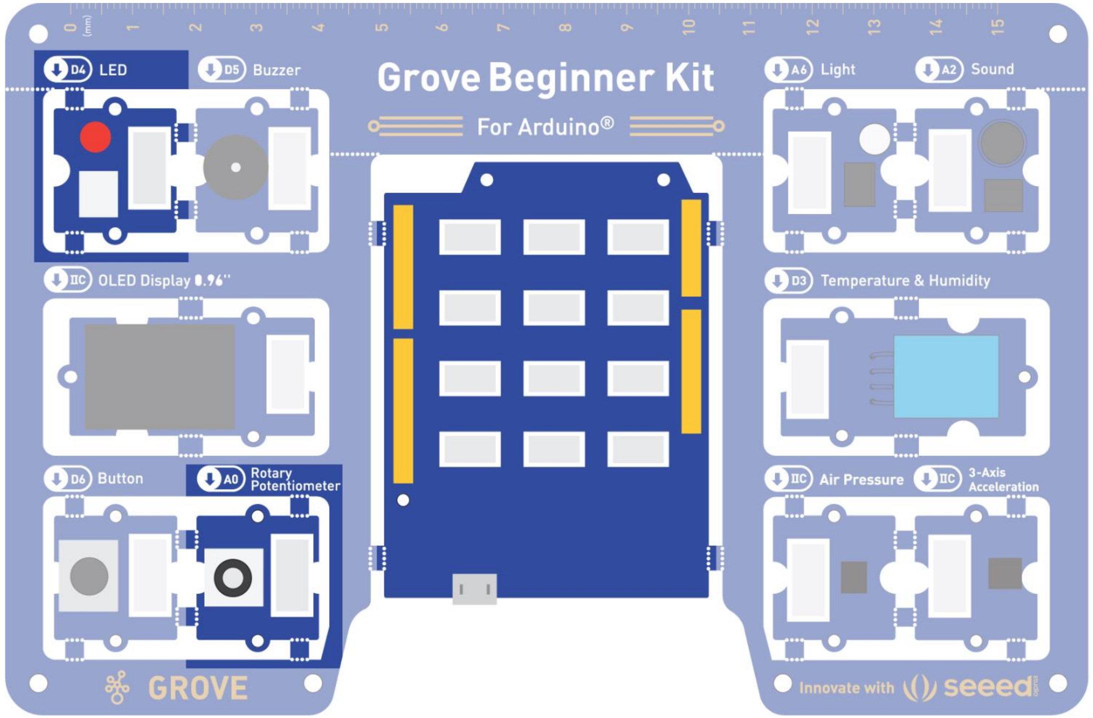


Image from https://files.seeedstudio.com/wiki/Grove-Beginner-Kit-For-Arduino/res/Grove-Beginner-Kit-For-ArduinoPDF.pdf

Open and Upload Sketch

Lesson 3: Pot

1. Open Pot Sketch
   a. **File → Sketchbook → CrashCourse_Jan → L3_Pot**
2. Verify the sketch by clicking the Verify Button.
   a. The sketch should compile with no errors.
3. Upload the sketch to your Arduino by clicking the Upload Button.
   a. The sketch should re-compile, and then upload to your Arduino.
4. Watch the LED as you rotate the potentiometer.

Code Analysis – delayMicroseconds Function

Lesson 3: Pot

```
delayMicroseconds(potValue);
```

- Wait for the number of microseconds (µs) stored in potValue.

- This function is similar to the delay function from Lesson 1, except it pauses by microseconds instead if milliseconds.
- There are 1000µs in 1ms and 1,000,000µs in 1s.
- Syntax:

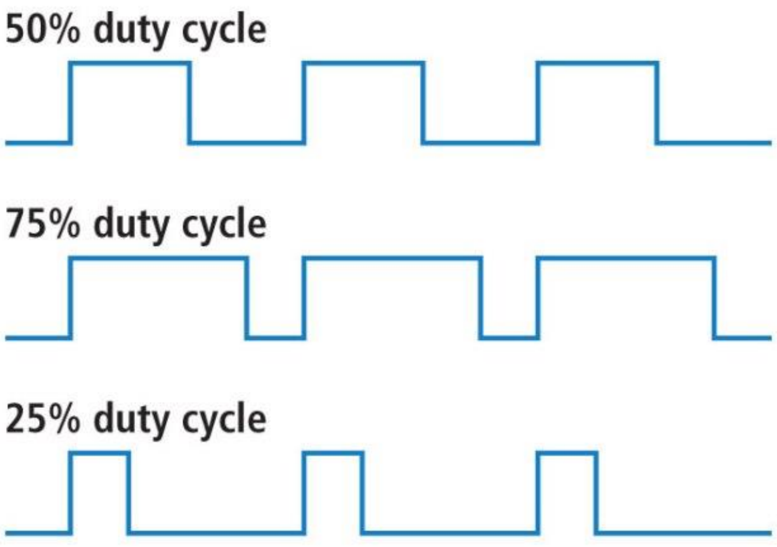
```
    delayMicroseconds(µs);
```

  - µs: number of microseconds (µs) to pause.
    - Data type is unsigned int with a range of 0 to 16,383µs (about 16 ms)
- More information:
  - https://www.arduino.cc/reference/en/language/functions/time/delaymicroseconds/

Pulse Width Modulation (PWM) Introduction

Lesson 3: Pot

- What is Pulse Width Modulation?
  - PWM is a type of digital signal that varies its value using the width of the pulse.
- It is easy to convert a PWM signal back to an analog signal with a low pass filter.
- We can use this to control the brightness of the LED.
- More Info:
  - https://en.wikipedia.org/wiki/Pulse-width_modulation
  - https://www.allaboutcircuits.com/textbook/semiconductors/chpt-11/pulse-width-modulation/
  - https://learn.sparkfun.com/tutorials/pulse-width-modulation/all



Pulse Width Modulation (PWM) Introduction

Lesson 3: Pot

- Two parts to Pulse Width Modulation
  - Duty Cycle (D), can can be calculated as follows:
    - $\mathrm{D} = \frac{t_H}{t_H + t_L}$
    - Where t_H is the time the signal is high,
    - And t_L is the time the signal is low.
    - Is measured as %
  - Frequency (Hz), how many times the cycle can occur in 1 second
    - $\mathrm{F} = \frac{1}{t_H + t_L}$

Pulse Width Modulation (PWM) Introduction

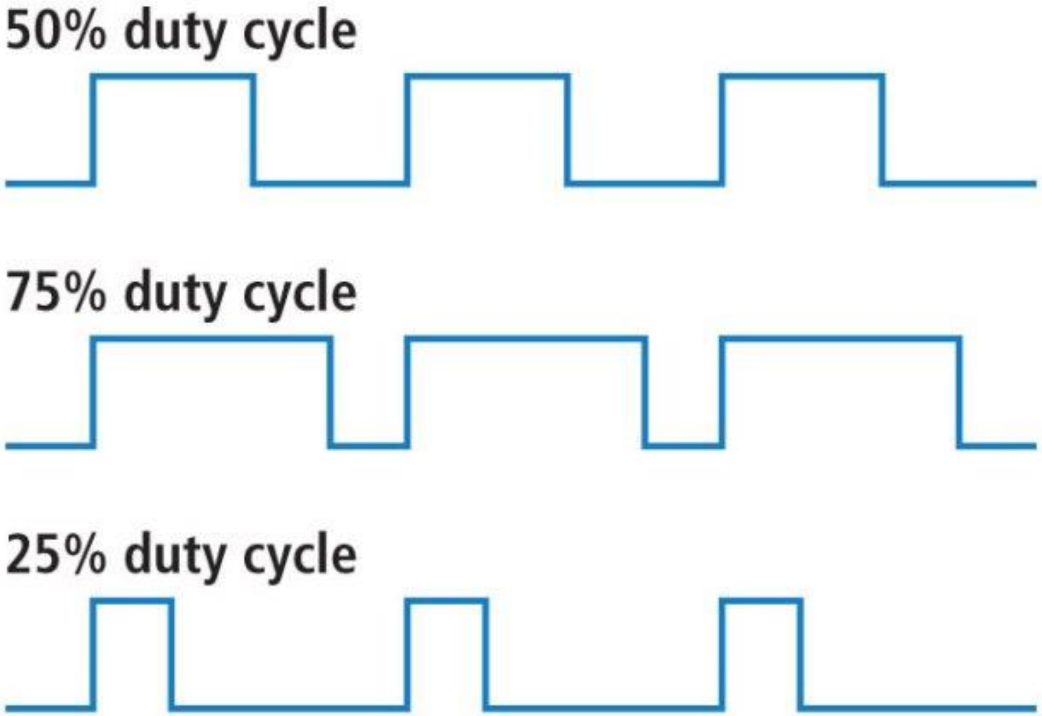
Lesson 3: Pot

- $\mathrm{D} = \frac{t_H}{t_H+t_L}$, $\mathrm{F} = \frac{1}{t_H+t_L}$

- t_H = 50ms, t_L = 50ms
  - $\mathrm{D} = \frac{50ms}{50ms+50ms}$ = 50%
  - $\mathrm{F} = \frac{1}{0.050s+0.050s}$ = 10Hz

- t_H = 25ms, t_L = 75ms
  - $\mathrm{D} = \frac{25ms}{25ms+75ms}$ = 25%
  - $\mathrm{F} = \frac{1}{0.025s+0.075s}$ = 10Hz

- t_H = 150ms, t_L = 50ms
  - $\mathrm{D} = \frac{ms}{ms+ms}$ = %
  - $\mathrm{F} = \frac{1}{s+s}$ = Hz



Pot Activities

Lesson 3: Pot

- Activity 1
  - Change the LED PWM to get brighter with a clockwise rotation of the potentiometer.
- Activity 2 (Bonus / Homework)
  - If the light sensor is on pin A6, modify the sketch to use the light sensor instead of the potentiometer.

Lesson 4: Light Sensor

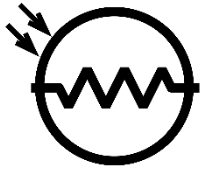
See the output of the light sensor in the Serial Monitor

Light Sensor Introduction

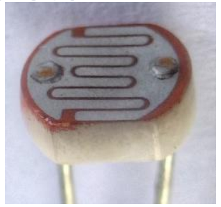
Lesson 4: Light Sensor

- What is a Light Sensor?
  - A light sensor is a type of device that changes a measurable electrical property based on the number (and type) of photons hitting it.
  - They come in many types but the main three for sensing applications are
    - **Photoresistors**: Resistance changes with light
    - **Photodiodes**: Photocurrent increases with light (this is also how a solar cell works)
    - **Phototransistors**: Amplified version of a photodiode.
- Where are light sensors used?
  - Occupancy sensors, daylight sensors, fiber optic communications, TVs (remote control receiver), cell phones, range finders, camera image sensors, etc.
  - Activity: Find a device not listed above that uses a light sensor.
- More information:
  - https://en.wikipedia.org/wiki/Photodetector
  - https://en.wikipedia.org/wiki/Photodiode
  - https://en.wikipedia.org/wiki/Photoresistor
  - https://www.seeedstudio.com/blog/2020/01/08/what-is-a-light-sensor-types-uses-arduino-guide/

Photoresistor

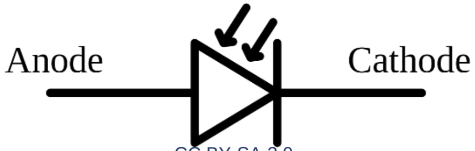


By User:FDominec et al. - File:Electrical_symbols_library.svg, CC0, https://commons.wikimedia.org/w/index.php?curid=49516462



By © Nevit Dilmen, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=30560805

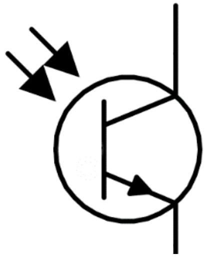
Photodiode



CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=755076

Copied from https://www.digikey.com/en/products/detail/w%C3%BCrth-elektronik/1540031EA4590/12366192

Phototransistor



By myself - WikiProject Wikipedia, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=32224508

Copied from https://www.digikey.com/en/products/detail/kingbright/WP7113P3C/7318904

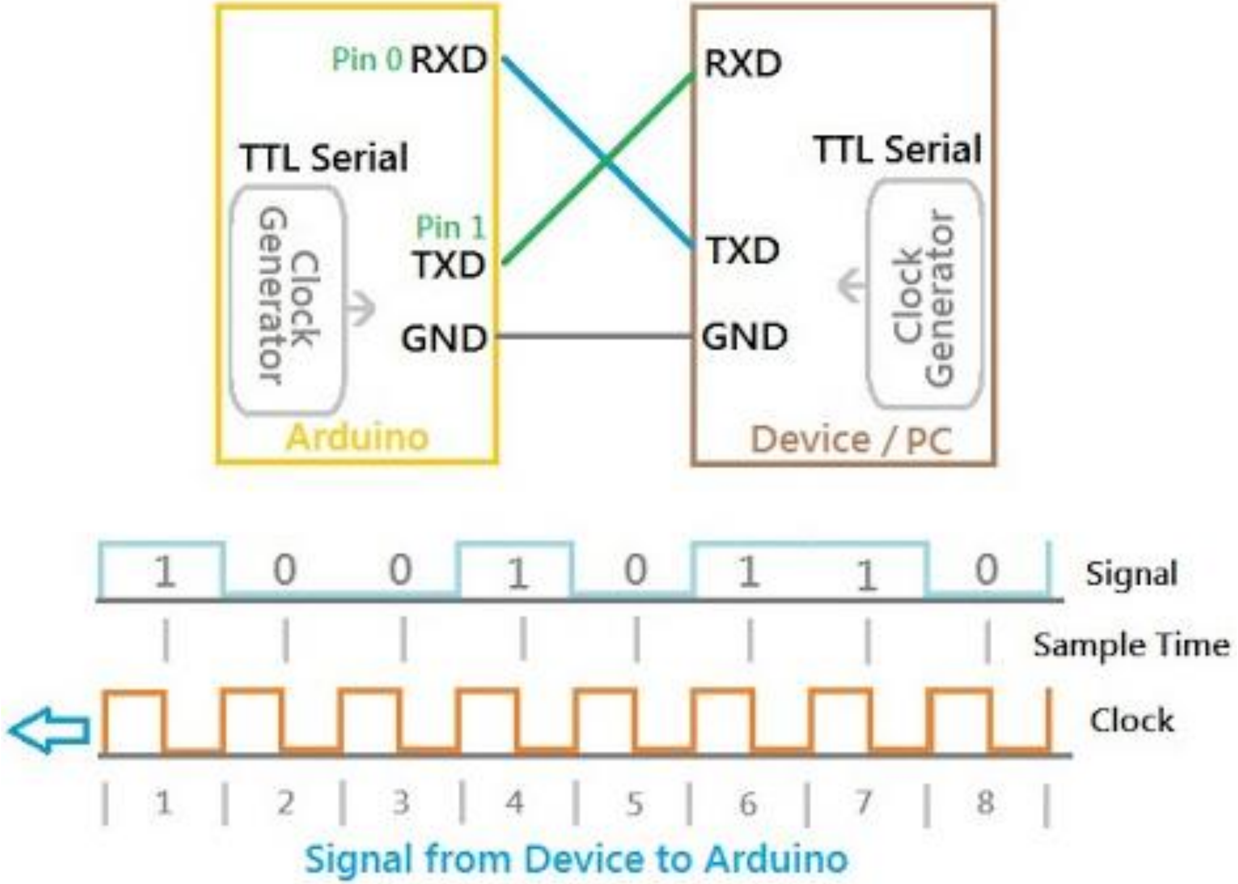
Serial Introduction

Lesson 4: Light Sensor

- What is Serial Communication?
  - A digital signal where data is sent one bit at a time over a single channel.
  - Serial communications include RS232, RS485, UART, USART, USB, Ethernet, CAN, I^{2}C, SPI, SATA, etc.
  - Serial (without descriptors) typically refers to RS-232 and related communication signaling standards (UART or USART for a microcontroller).
- Where are serial communications used?
  - Internet, computers, cell
- More information:
  - https://en.wikipedia.org/wiki/Serial_communication
  - https://www.codrey.com/embedded-systems/uart-serial-communication-rs232/

Serial Introduction

Lesson 4: Light Sensor



- More information:
  - http://elextutorial.com/learn-arduino/arduino-serial-communication-write-port-example-test-begin/

Lesson 4 Hardware

Lesson 4: Light Sensor

- What hardware will we need for this Lesson?
  - Grove Light Sensor Module on pin A6
  - Seeeduino Lotus (Arduino Uno compatible board)
    - The Arduino has the serial port hardware built into the device

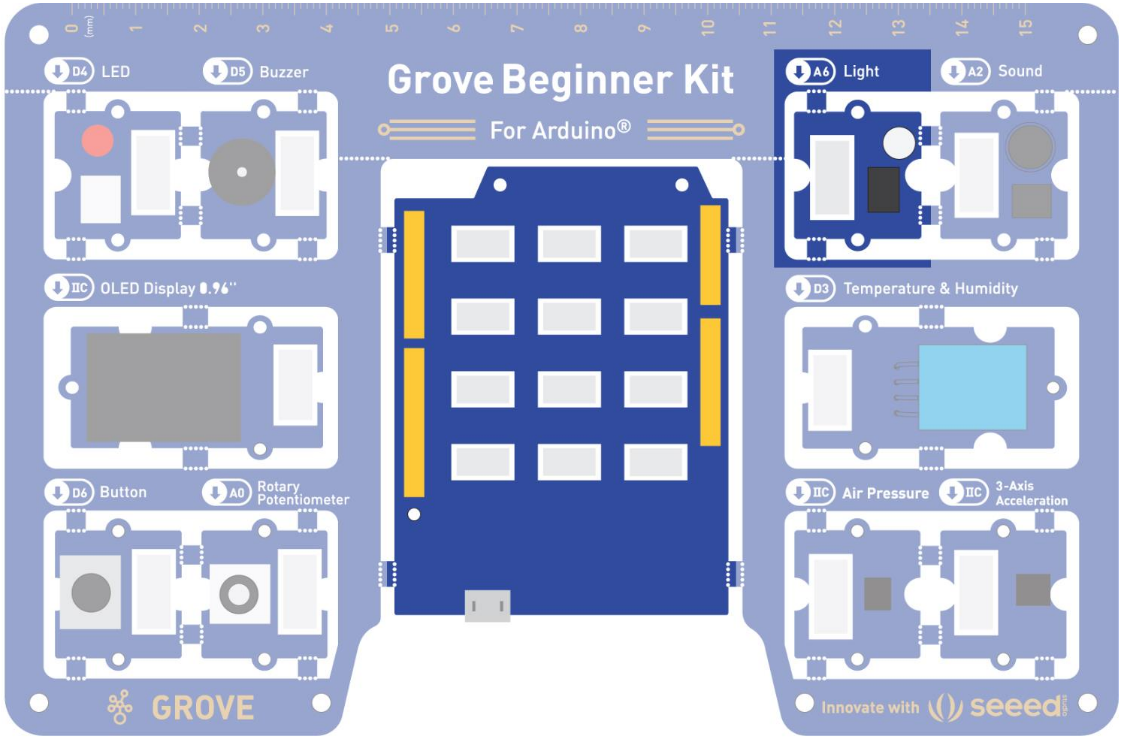
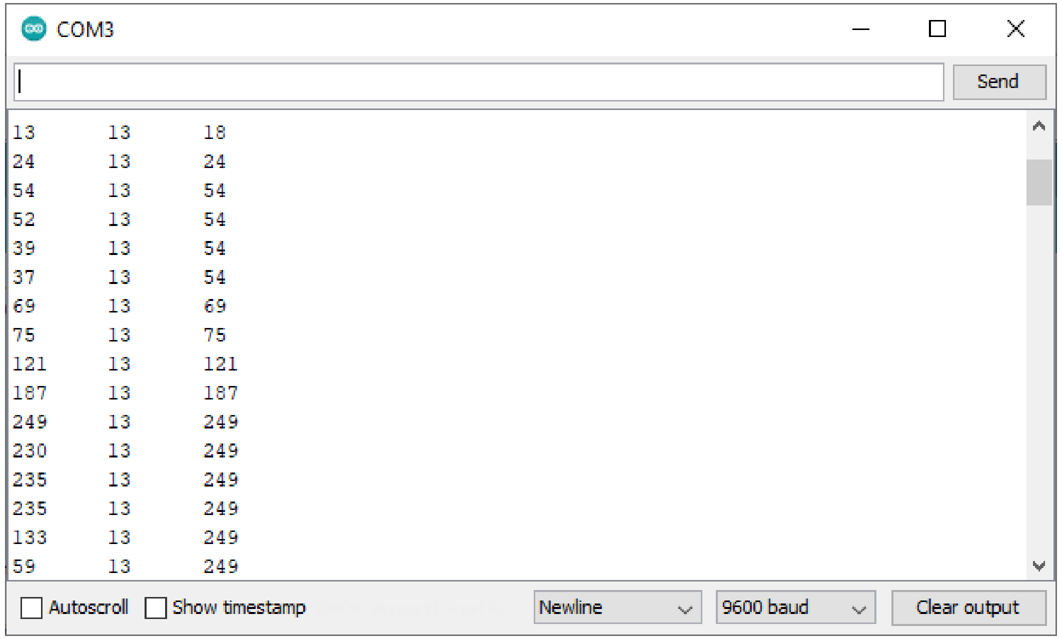


Image modified from https://files.seeedstudio.com/wiki/Grove-Beginner-Kit-For-Arduino/res/Grove-Beginner-Kit-For-ArduinoPDF.pdf

Open and Upload Sketch

Lesson 4: Light Sensor

1. Open Light_Serial Sketch
   a. **File → Sketchbook → CrashCourse_Jan → L4_Light_Serial.ino**
2. Verify the sketch by clicking the Verify Button.
   a. The sketch should compile with no errors.
3. Upload the sketch to your Arduino by clicking the Upload Button.
   a. The sketch should re-compile, and then upload to your Arduino.
4. Open the serial monitor.
   a. **Tools → Serial Monitor** (Ctrl+Shift+M)
5. Observe the output in the Serial Monitor

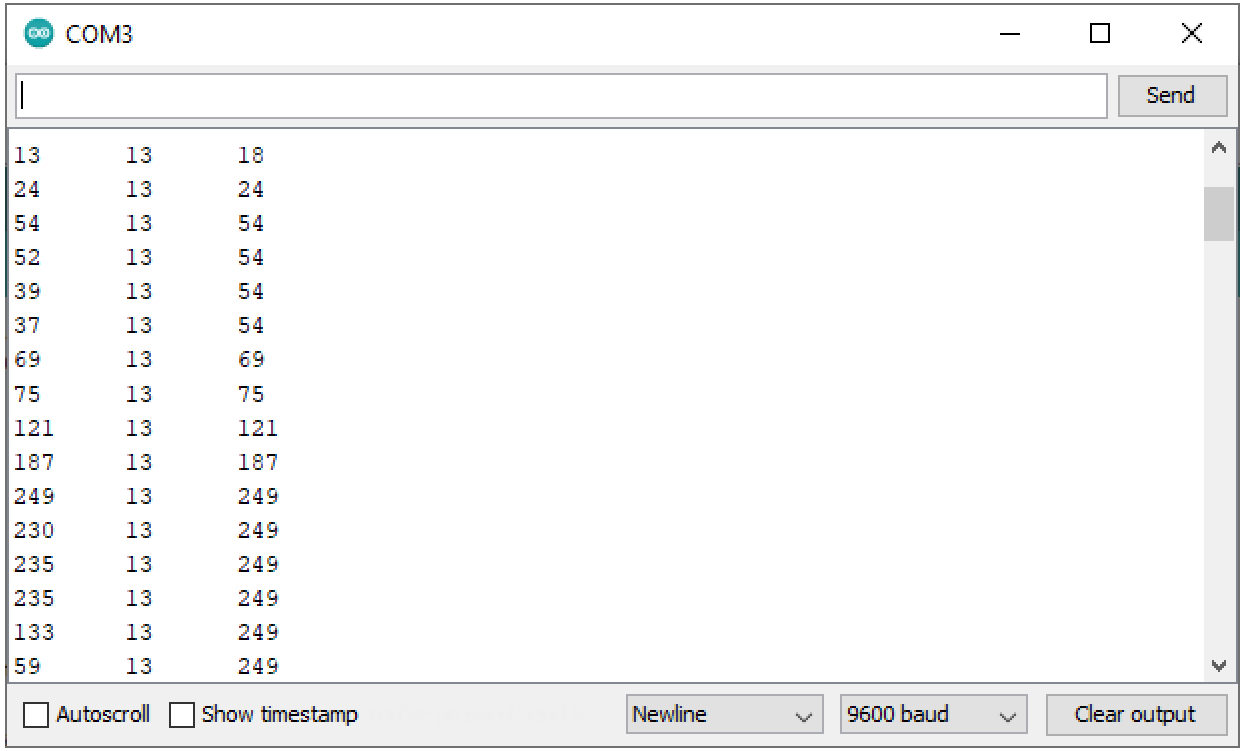


```
13      13      18
24      13      24
54      13      54
52      13      54
39      13      54
37      13      54
69      13      69
75      13      75
121     13      121
187     13      187
249     13      249
230     13      249
235     13      249
235     13      249
133     13      249
59      13      249
```

Serial Monitor

Lesson 4: Light Sensor

- What is the Serial Monitor?
  - The Serial Monitor is a feature of the Arduino IDE that gives you a serial terminal to see what is being sent to the COM port and allows you to send stuff out of the COM port.
  - We use this for receiving data from the Arduino.
  - We can also use this to help us debug our sketches.



```
13      13      18
24      13      24
54      13      54
52      13      54
39      13      54
37      13      54
69      13      69
75      13      75
121     13      121
187     13      187
249     13      249
230     13      249
235     13      249
235     13      249
133     13      249
59      13      249
```

Code Analysis – `Serial` Functions

Lesson 4: Light Sensor

- `Serial.begin(9600);`
  - Start the Serial port at a 9600 baud
  - Put this function in the setup() function
  - Must call this function before using any other serial function
- `Serial.print(“string”);`
  - print a string or value to the serial port
- `Serial.println(“string”);`
  - same as print but add a new line character at the end of the string or value
- Special characters:
  - `‘\t’` is a Tab character
  - `‘\n’` is a New Line (some operating systems [⊞] use `“\r\n”`)
- More Information:
  - https://www.arduino.cc/reference/en/language/functions/communication/serial/
  - https://en.wikipedia.org/wiki/Control_character

Code Analysis – `min()` and `max()` Functions

Lesson 4: Light Sensor

```
min(valueA, valueB);
```

- Returns whichever value is lower

```
max(valueA, valueB);
```

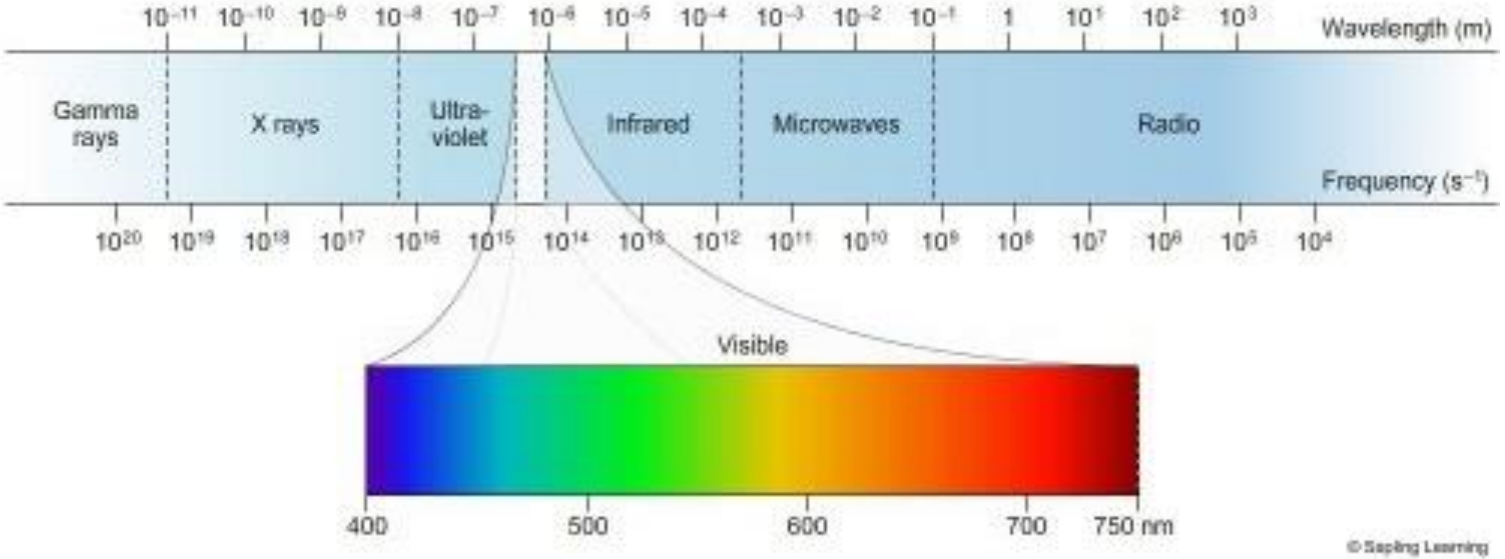
- Returns whichever value is higher

- More information:
  - https://www.arduino.cc/reference/en/language/functions/math/max/
  - https://www.arduino.cc/reference/en/language/functions/math/min/

Sensors & Applications

Sensors & Applications: Color Sensors

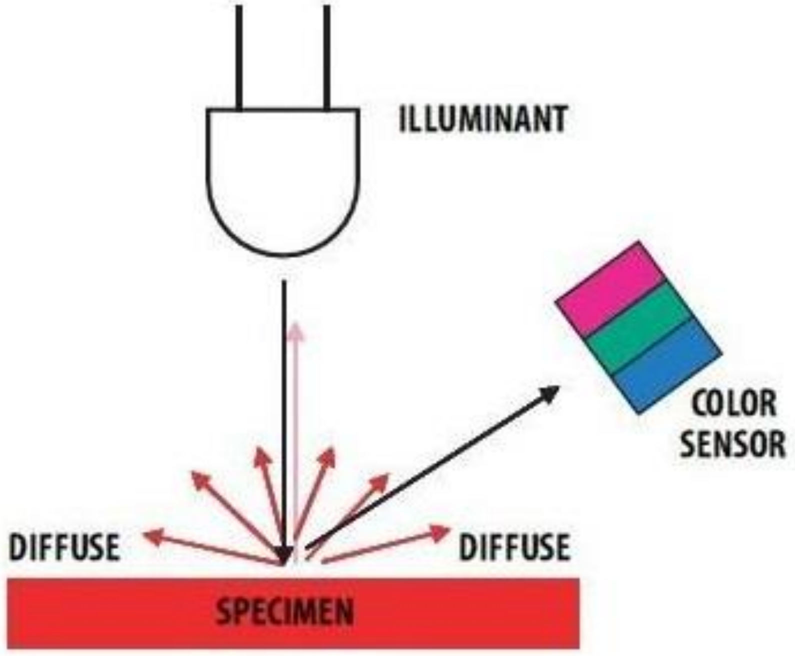
- Think of how Red, Green and Blue combine to make colors
  - Sensors have individual photodiodes that are sensitive to a frequency band of light (color) and measure the intensity of that frequency.
  - Filters are used to make the photodiodes sensitive to limited frequency bands
  - Data from the individual diodes is combined to create a color measurement.



© Sapling Learning

Sensors & Applications: Color Sensors

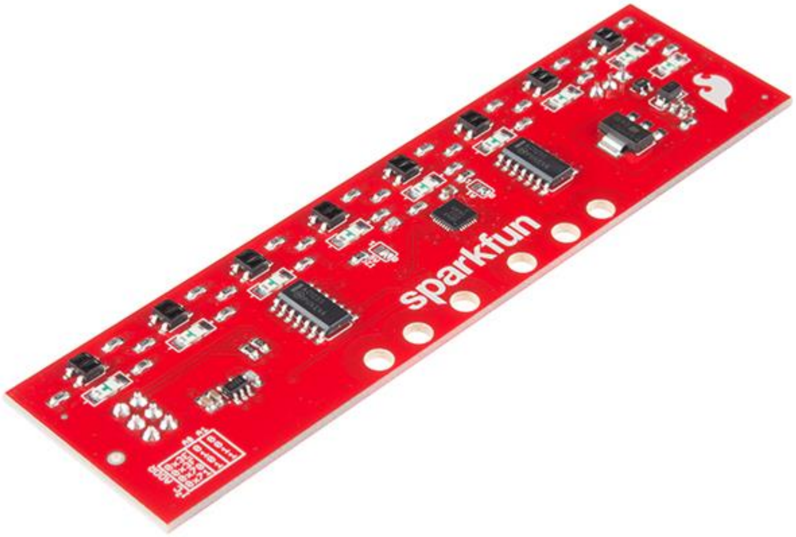
- Object Color Detection
  - Typical Application: light that shines out, reflected wavelengths are measured
  - Able to measure intensity of ambient light



- https://wiki.seeedstudio.com/Grove-I2C_Color_Sensor/

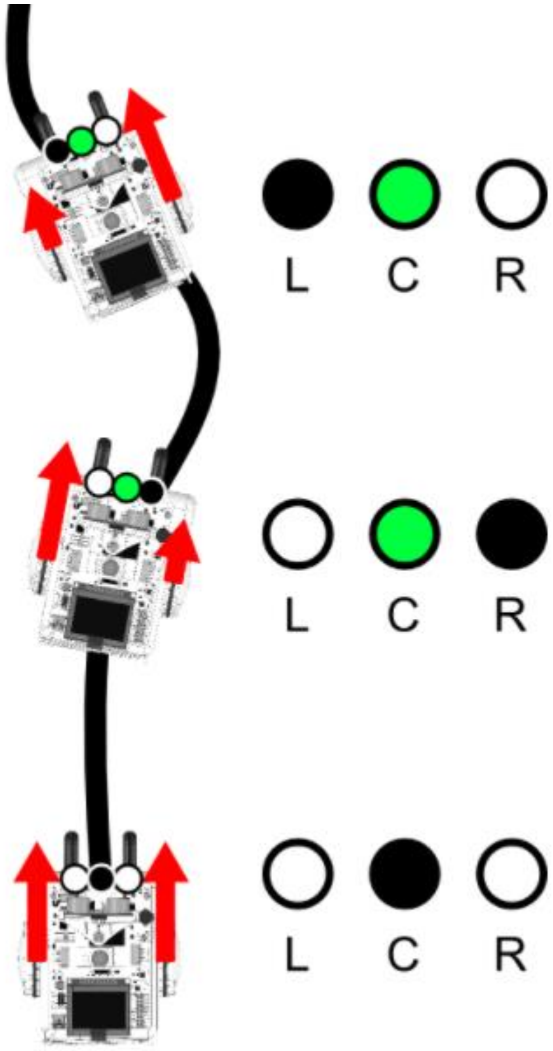
Sensors & Applications – Line Following Sensors

- Typically utilize IR (InfraRed) sensors
  - IR sensor consists of an LED and phototransistor
    - LED emits an IR light (humans an unable to see this)
    - Phototransistor is measuring IR light that is reflected back
      - White surface: reflects light back to the phototransistor
      - Black surface: absorbs light



Sensors & Applications – Line Following Sensors

- In-Use



Wrap-Up

Next Week

- Debug Sketch *W2_Debug*
  - Figure out why the program is not compiling.
- Challenge Sketch *W2_Chal*
  - Start with the sketch outline and write a program that:
    - Turns on the LED only when the potentiometer input is above 2.5V and display the potentiometer ADC input on the serial monitor
      - Hint: What is the ADC input for 2.5V
- Office Hours @ 7:00 PM Monday
  - Same Zoom call information
  - Will go through Debug and Challenge sketches

Lesson 5: Microphone

See the output of the microphone in the Serial Plotter

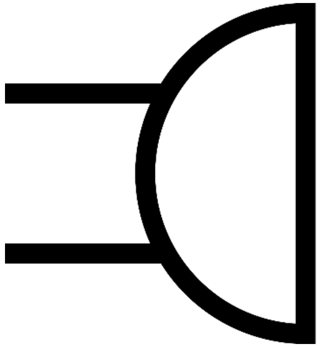
At-home activity

Microphone Introduction

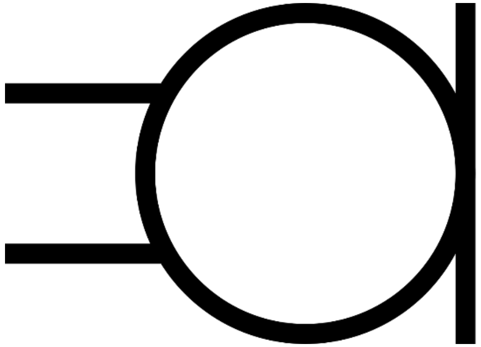
Lesson 5: Microphone

- What is a microphone?
  - A microphone is a transducer that converts sound wave to an electrical signal.
  - Microphones are used to record music and voice, but also used for scientific analysis.
- Where are microphones used?
  - Audio recording, cell phones, walkie-talkie, computers, sonar, presence detection, knock detection, etc.
- How do we use the Microphone?
  - Microphones must be amplified or conditioned before we can use the signal. We can then read the analog signal with the ADC in the microcontroller.
- More Information:
  - https://en.wikipedia.org/wiki/Microphone

Microphone Electrical Symbols



By ErikBuer - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=50257723



By ErikBuer - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=50257685

Microphone

Downloaded from https://www.digikey.com/en/products/detail/cui-devices/CMA-6542PF/1869980

Lesson 5 Hardware

Lesson 5: Microphone

- What hardware will we need for this Lesson?
  - Grove Sound Module on pin A2
  - Seeeduino Lotus (Arduino Uno compatible board)
    - The Arduino has the serial port hardware built into the device

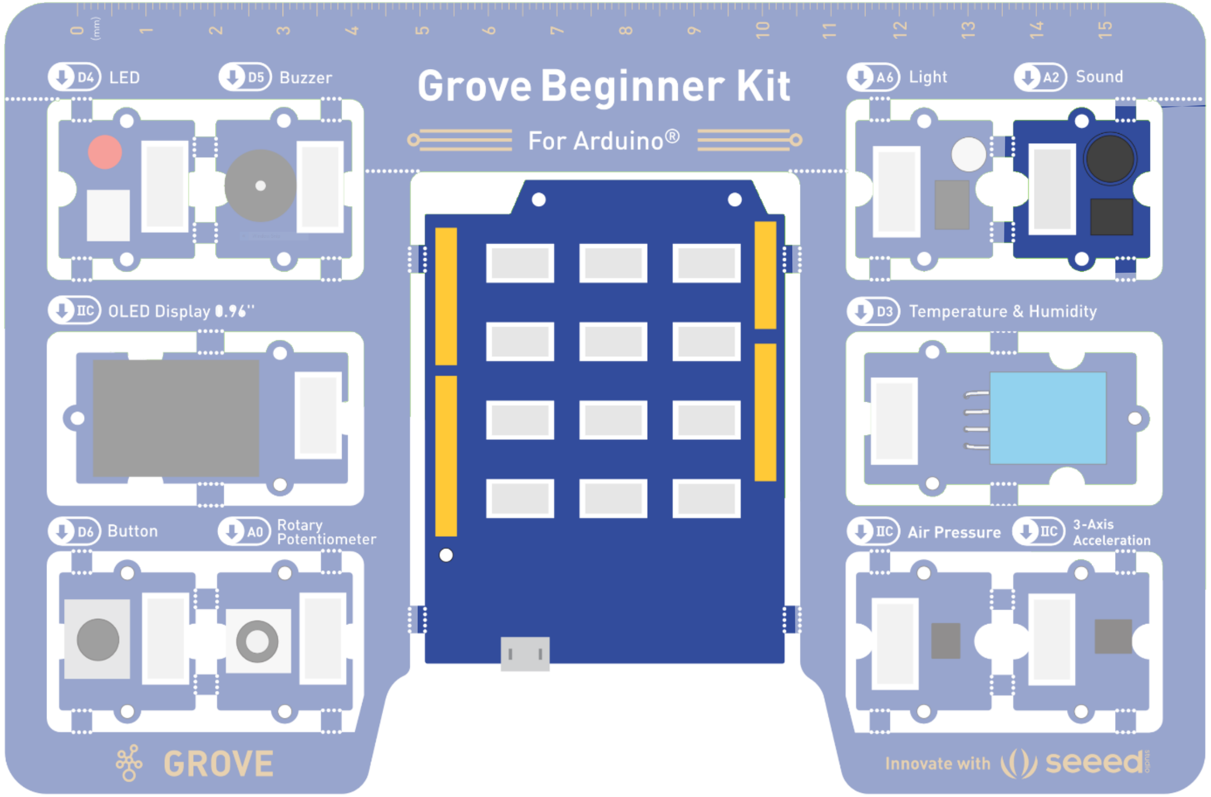


Image modified from https://files.seeedstudio.com/wiki/Grove-Beginner-Kit-For-Arduino/res/Grove-Beginner-Kit-For-ArduinoPDF.pdf

Open and Upload Sketch

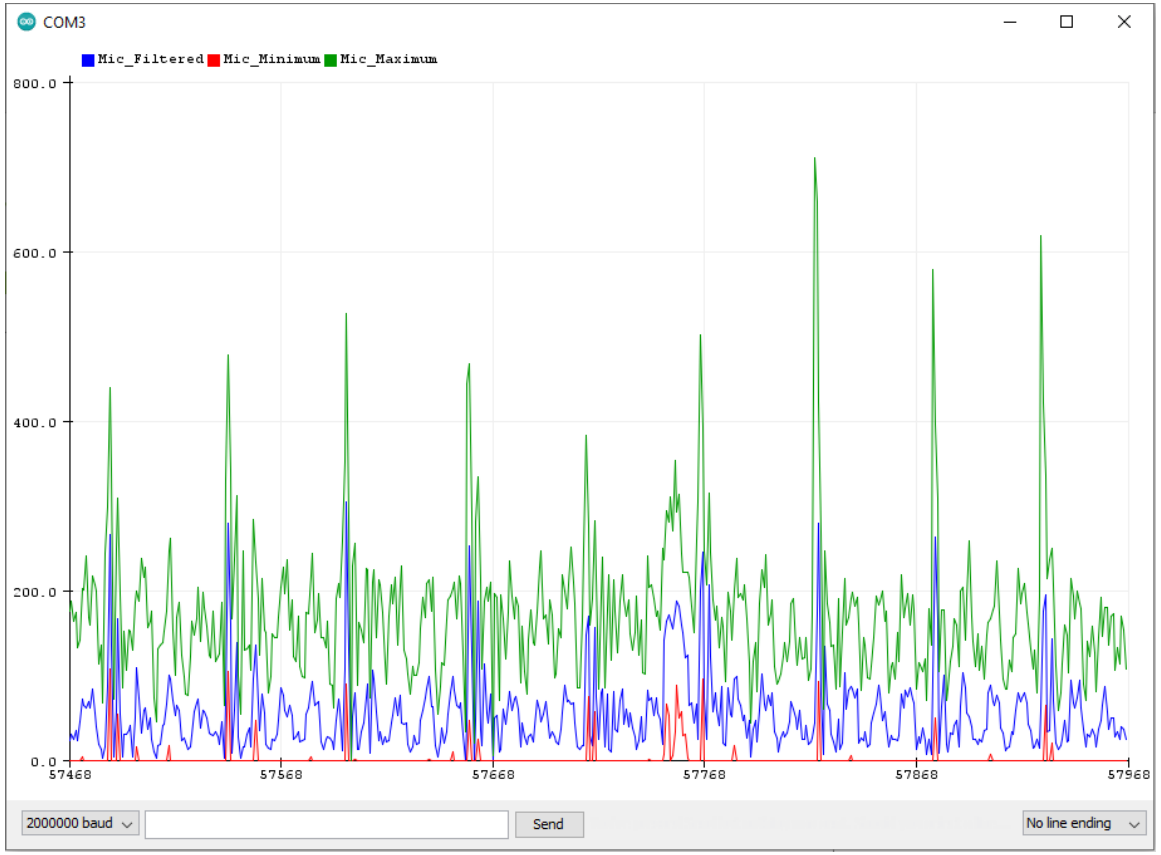
Lesson 5: Microphone

1. Open Microphone Sketch
   a. **File → Sketchbook → FRSEF_Crash_Course → Week_2 → W2L5_Microphone.ino**
2. Verify the sketch by clicking the Verify Button.
   a. The sketch should compile with no errors.
3. Upload the sketch to your Arduino by clicking the Upload Button.
   a. The sketch should re-compile, and then upload to your Arduino.
4. Open the serial monitor.
   a. **Tools → Serial Plotter** (Ctrl+Shift+L)
5. Observe the output in the Serial Plotter

Serial Plotter

Lesson 5: Microphone

- What is the Serial Plotter?
  - The Serial Plotter is a feature of the Arduino IDE that gives you a graphical representation of what is being sent to the COM port.
  - We use this for receiving data from the Arduino.
  - The serial plotter will display up to 500 consecutive sample periods.



2000000 baud

No line ending

- More Information:
  - https://arduinogetstarted.com/tutorials/arduino-serial-plotter

Serial Plotter

Lesson 5: Microphone

- How do we use the serial plotter?
  - Optionally we start off with a header using the syntax:

`Serial.println(“header_1 header_2”);`

    - We can add more headers by separating them with a space
  - To display the values, we use the `Serial.print()` and `Serial.println()` functions to send values to the Serial Plotter similar to how we sent values to the Serial Monitor.
  - Each value in a sample period should be separated by tab `‘/t’` character. Each new sample period should be separated by a newline character or using the `Serial.println()` function.

Example Serial Plotter Code

```
void setup()
{
  Serial.begin(9600);
  Serial.println(“header1 header2”);
}

void loop()
{
  // get values to display
  int val1 = analogRead(A0);
  int val2 = analogRead(A2);
  Serial.print(val1);
  Serial.print(‘/t’);
  Serial.println(val2);
}
```

Code Analysis – C++ Bitshift Operators

Lesson 5: Microphone

```
for(unsigned int i = 0; i < 1<<filterConstant; i++)
```

- Shift binary 1 to the left by **filterConstant** bits
  - Equivalent to $2^{\texttt{filterConstant}}$

<< Bitshift Left `0b0001 << 2 = 0b0100`

- Shift the binary value on the left by the number of bits on the right

>> Bitshift Right `0b0101 >> 1 = 0b0010`

- Shift the binary value on the right by the number of bits on the left

```
A << 1 → A * 2
B << 2 → B * 4
C << 3 → C * 8

D >> 1 → D / 2
E >> 2 → E / 4
F >> 3 → F / 8
```

- Leading or trailing digits get dropped, and new digits are 0
- Often used to efficiently multiply (<<) or divide (>>) by powers of 2
- More Information:
  - https://beginnersbook.com/2017/08/cpp-operators/
  - https://www.arduino.cc/reference/en/

Code Analysis – C++ Comparison Operators

Lesson 5: Microphone

```
for(unsigned int i = 0; i < 1<<filterConstant; i++)
```

- Is **i** less than $2^{\texttt{filterConstant}}$?

==	Equal To	→ TRUE if the left side is **equal to** the right side
!=	Not Equal To	→ TRUE if the left side is **not equal to** the right side
<	Less Than	→ TRUE if the left side is **less than** the right side
<=	Less Than or Equal To	→ TRUE if the left side is **less than or equal to** the right side
>	Greater Than	→ TRUE if the left side is **greater than** the right side
>=	Greater Than or Equal To	→ TRUE if the left side is **greater than or equal to** the right side

- More Information:
  - https://beginnersbook.com/2017/08/cpp-operators/
  - https://www.arduino.cc/reference/en/

Code Analysis – C++ Arithmetic Operators

Lesson 5: Microphone

+	Addition	A = 1 + 2	→	A = 3
-	Subtraction	B = 3 - 1	→	B = 2
*	Multiplication	C = 2 * 4	→	C = 8
/	Division	D = 6 / 3	→	D = 2
%	Modulo (remainder)	E = 7 % 4	→	E = 3

- More Information:
  - https://beginnersbook.com/2017/08/cpp-operators/
  - https://www.arduino.cc/reference/en/

Code Analysis – C++ Auto-increment and Auto-decrement Operators

Lesson 5: Microphone

```
for(unsigned int i = 0; i < 1<<filterConstant; i++)
```

- Increment **`i`** by 1 at the end of the for loop.

++ Auto-increment `i++` → `i = i + 1`

- Increments the value of a variable by 1

-- Auto-decrement `j--` → `j = j - 1`

- Decrements the value of a variable by 1

- More Information:
  - https://beginnersbook.com/2017/08/cpp-operators/
  - https://www.arduino.cc/reference/en/

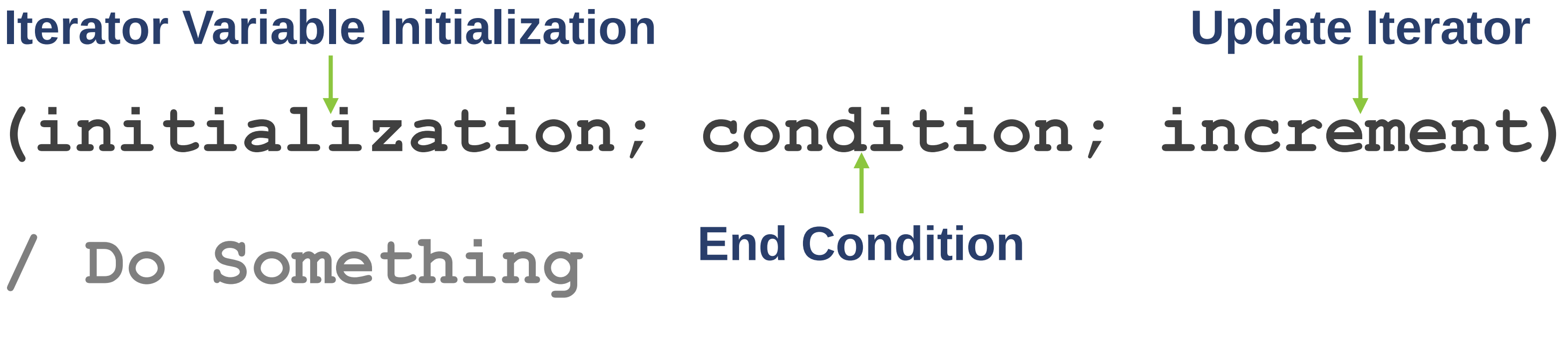
Code Analysis – for Loop

Lesson 5: Microphone

```
for(unsigned int i = 0; i < 1<<filterConstant; i++)
{/* Do Something */}
```

  - Repeat code inside the curly braces $2^{\texttt{filterConstant}}$ times

- for() Loops are used to repeat code that appears between its curly braces
- Syntax:



```
for(initialization; condition; increment)
{
   // Do Something
}
```

- More Information:
  - https://www.arduino.cc/reference/en/language/structure/control-structure/for/
  - https://beginnersbook.com/2017/08/cpp-for-loop/

Code Analysis – C++ Assignment Operators

Lesson 5: Microphone

```
micValueLong += micValue;
```

- Add **micValue** and **micValueLong** then store the result in **micValueLong**

=	Equals	Assigns value of right side to the left side
+=	Plus Equals	A += 2 → A = A + 2
-=	Minus Equals	B -= 3 → B = B - 3
***=**	Multiplication	C *= 4 → C = C * 4
/=	Division	D /= 5 → D = D / 5
%=	Modulo (remainder)	E %= 6 → E = E % 6

- More Information:
  - https://beginnersbook.com/2017/08/cpp-operators/
  - https://www.arduino.cc/reference/en/

Code Analysis – Averaging Filter

Lesson 5: Microphone

- What is a filter?
  - A filter is used to remove an unwanted component of a signal.
  - For sensor measurements a **low pass filter** is often used to reduce noise or some high frequency component.
  - There are many different types of filters, and numerous ways to implement filters.
- What is averaging?
  - Averaging is taking the mean value of a signal over the sampling period.
- More Information:
  - https://en.wikipedia.org/wiki/Filter_(signal_processing)
  - https://en.wikipedia.org/wiki/Average
  - https://www.mathsisfun.com/mean.html

```
// Average filter
int average;
int sumSamples = 0;
for(int i = 0; I < numSamples; i++)
{
 sumSamples += analogRead(A2);
}
average = sumSamples / numSamples;
```