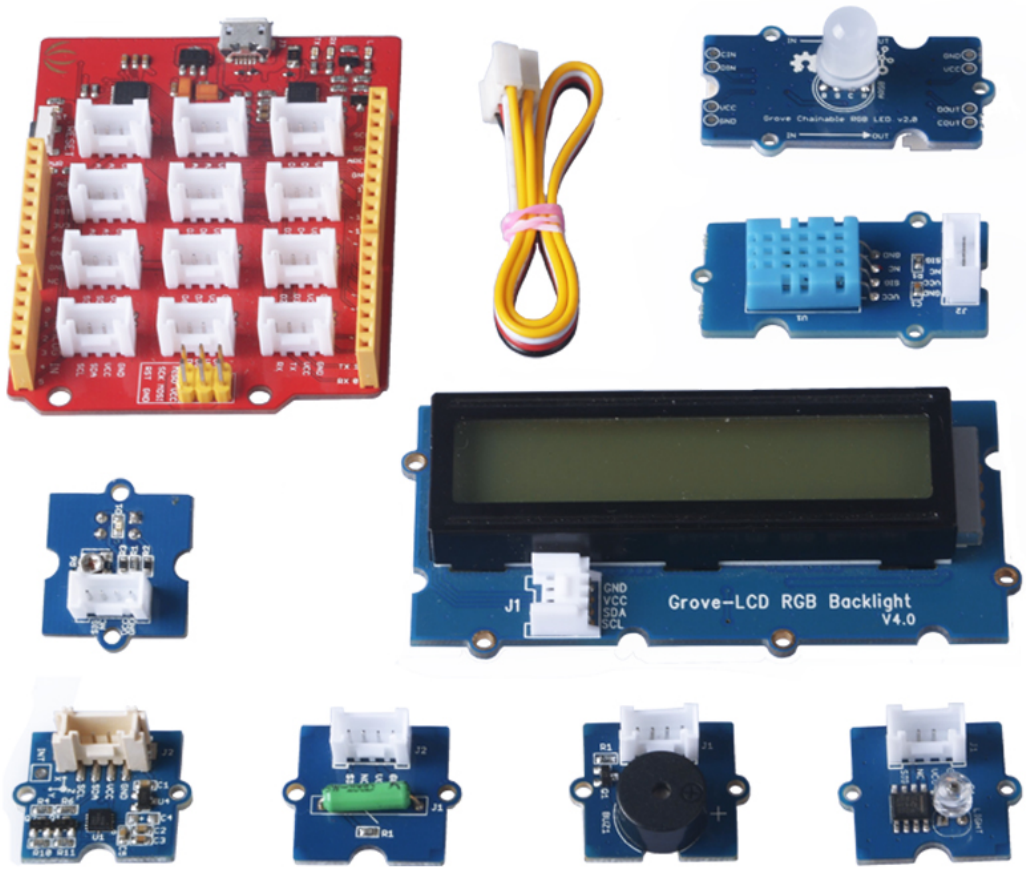
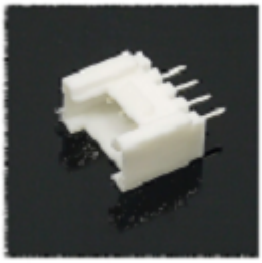
Grove Beginner Kit for Arduino SKU:110020171

GROVE SYSTEM



Grove is a modular prototyping system consist of a base unit and various modules with standardized connector. the base unit is generally a microprocessor which allows for communicates, processes and controls the input or output from the Grove modules. Every single Grove module typically addresses a single function, range from a simple button to a more complex heart rate sensor. the standardised Grove connector allows user to assemble Grove units with building block approach, compared to the jumper or solder based system it is much easier to assemble or disassemble, which simplifies the learning system for experimenting, building and prototyping. We also provide Grove to Pin Header Converter or Grove Base HAT available for variety developing platforms for those who wants to use grove sensor and actuator modules without Grove System Development Board.



Grove system users need to have at least some basic electronic knowledge background, otherwise you need go through this basic tutorial to learn some basic operations on the Grove system, the first part of this tutorial consists list of basic information on the components included in the starter kit, followed by the basic setup of the Arduino IDE for Seeeduino Lotus. Then, the 11 tutorial sessions provide the basic operation on each individual components in the starter kit and the applications by combine multiple modules together, which gives learner some insight and basic knowledge on hooking up and coding with the Grove system.

Grove Beginner Kit for Arduino

Grove Beginner Kit for Arduino contains one Seeeduino Lotus V1.1 Dev Board(Arduino Compatible) and 8 modules. The detailed information are listed below.

Development Board

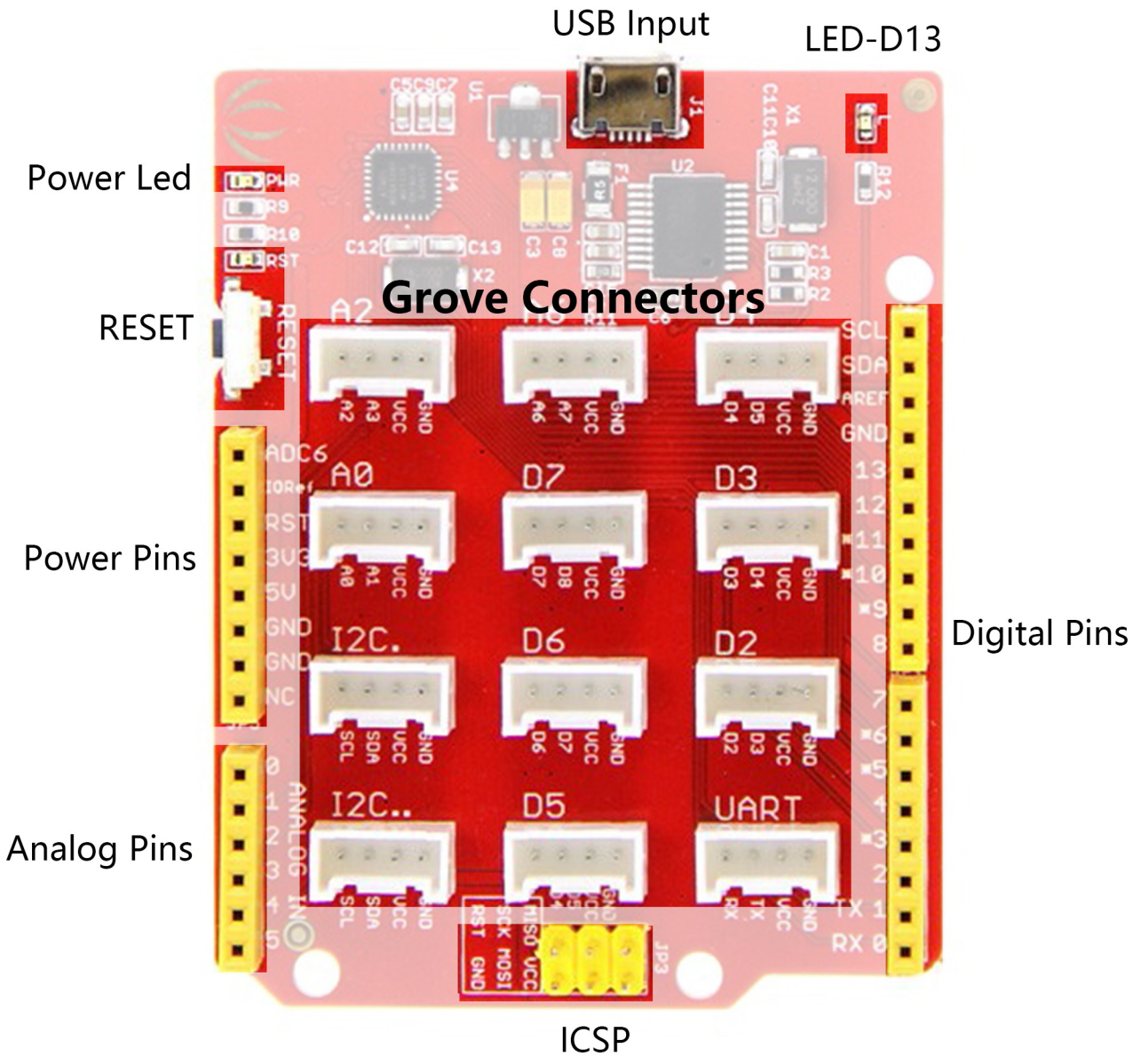
Seeeduino Lotus V1.1

Seeeduino Lotus is a development board with ATMEGA328 AVR microcontroller, it is the combination of Seeeduino and Grove Base Shield. It uses an Atmel ATmege328-MU microcontroller and CP2102N chip, ATmege328-MU is a high performance, low power AVR 8-bit Microcontroller, CP2102N is a USB to Serial converter chip which allows you to communicate the Seeeduino Lotus with computer by using a micro-USB cable. Seeeduino Lotus has 14 Digital input/output(6 can be used as PWM outputs) and 7 Analog input/output, a micro USB connection, an ICSP header, 12 Grove connector, a reset button.

Features

- Fully compatible with Arduino UNO
- ATmega328 microcontroller
- 2 on-board Grove connectors
- 14 Digital I/O Pins (6 PWM outputs)
- 6 Analog Inputs
- ISP Header
- Arduino UNO-R3 Shield Compatible
- Micro USB programming and power supply
- 5V Operating Voltage
- Supports Windows, Mac OS and Linux

Hardware



LED-D13: An LED is connected to D13 pin of the board. This can be used as an on-board LED indicator for programs/sketches.

USB Input: USB Port is used to connect the board to your PC for programming and for powering up. Micro USB is the a very common type of USB port, could be found with most Android phones, and other devices. You probably have dozens of these cables laying around your house.

Reset: This button is conveniently placed on the side to allow you to reset the Seeeduino board even when a shield is placed on top. whereas the button on the other Arduino board is placed on top which makes it hard to reach with shield attached.

Power Pins, Analog Pins & Digital Pins: These extra headers are available when you want to connect other none grove connector sensors and actuators, and especially the power headers are used when you want to power more sensors/devices.

Grove Connectors: Seeed Studio has a variety of sensors/devices that can make use of this Analog, Digital,I2C and UART connection. In addition, we sell independent Grove connectors to help you make our own sensor connections.

ICSP: This is the ICSP connection for the ATmega328P, it is located in the standard ICSP/SPI position for Arduino Uno, Due, Mega, and Leonardo compatible hardware. The SPI pins in this port: MISO, SCK, and MOSI, are also connected to digital pins 12, 13, and 11 respectively just like those of the Arduino Uno.

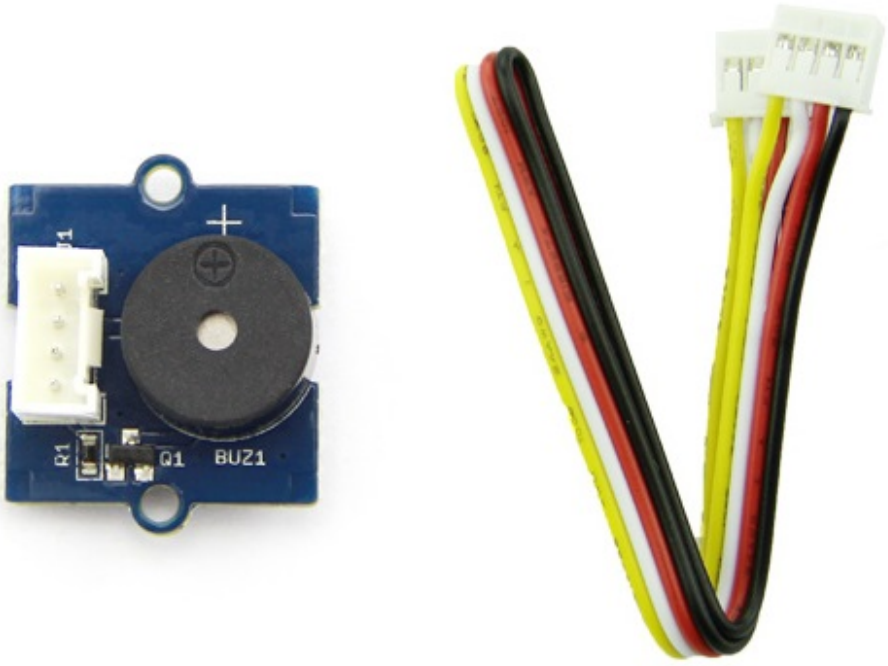
USB 2 Uart: Pinout of USB-2-Uart. These pads can be used to interact with other UART devices by putting the on-board ATmega328 in reset mode. This makes Seeeduino Lotus to be used as a USB2UART utility board.

Arduino UNO vs Seeeduino Lotus

	Seeeduino Lotus V1.1	Arduino Uno R3
Release Date	2018/03	2016/02
Microcontroller	ATMega328P	ATMega328P
Operating Voltage	5V	5V
Flash	32KB	32KB
SRAM	2KB	2KB
EEPROM	1KB	1KB
Power supply interface	Micro USB	USB, DC Port
Grove Connectors	12	None

Sensors

Grove - Buzzer



This module uses piezo buzzer as the main component, it can produce high pitch tone while it is connected to digital output and logic level set to High, otherwise it can produce various tones according to the frequencies

generated from the Analog PWM output that connected to it. (note: the frequency range that normal human ear can distinguish is between 20 Hz and 20kHz.)

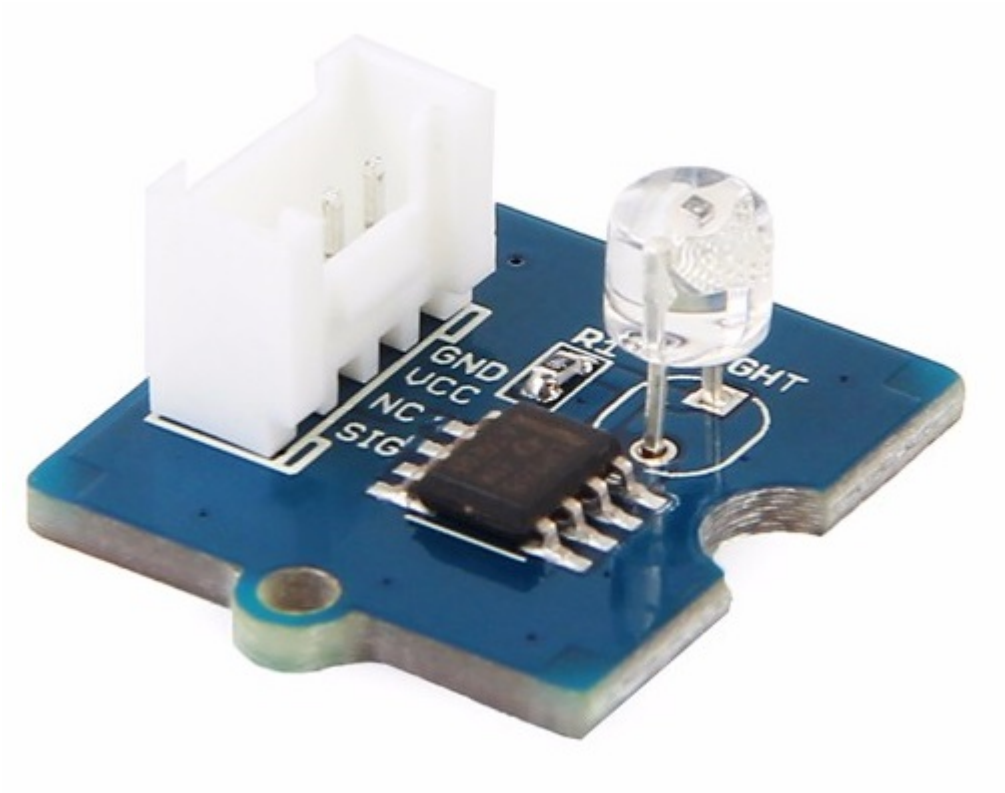
Grove - Tilt Switch

Grove-Tilt Switch is the equivalent of a button, and is used as a digital input. Inside the tilt switch is a pair of balls that make contact with the pins when the case is upright. Tilt the case over and the balls don't touch, thus not making a connection. It is wired to the SIG line, NC is not used on this Grove module.

Grove - Chainable RGB LED

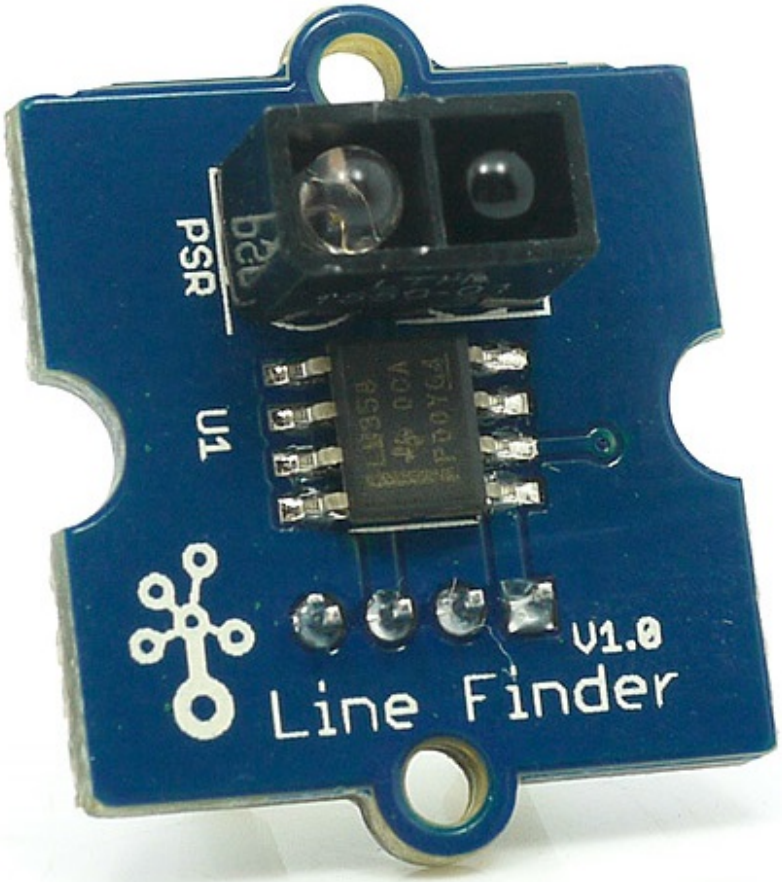
Grove - Chainable RGB LED is based on P9813 chip which is a full-color LED driver. It provides 3 constant-current drivers as well as modulated output of 256 shades of gray. It communicates with a MCU using 2-wire transmission (Data and Clock). This 2-wire transmission can be used to cascade additional Grove - Chainable RGB LED modules. The built-in clock regeneration enhances the transmission distance. This Grove module is suitable for any colorful LED based projects.

Grove - Light Sensor



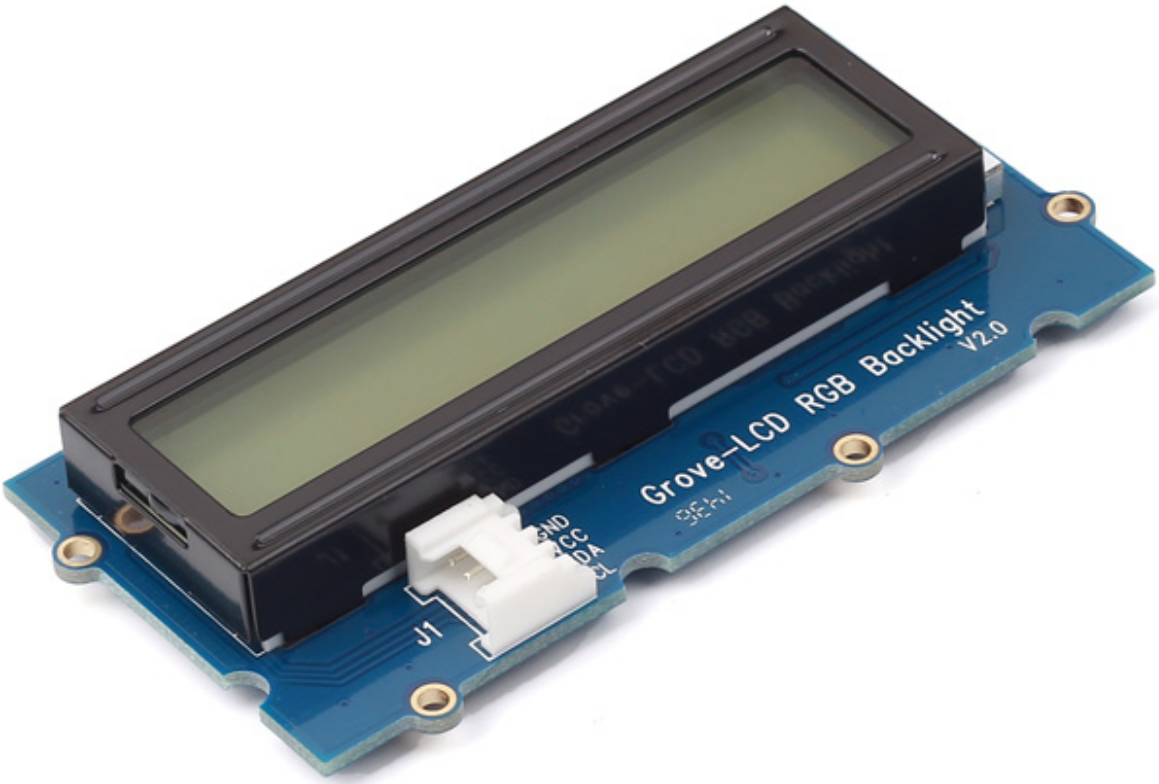
The Grove - Light sensor integrates a photo-resistor(light dependent resistor) to detect the intensity of light. The resistance of photo-resistor decreases when the intensity of light increases. A dual OpAmp chip LM358 on board produces voltage corresponding to intensity of light(i.e. based on resistance value). The output signal is analog value, the brighter the light is, the larger the value.

Grove - Line Finder



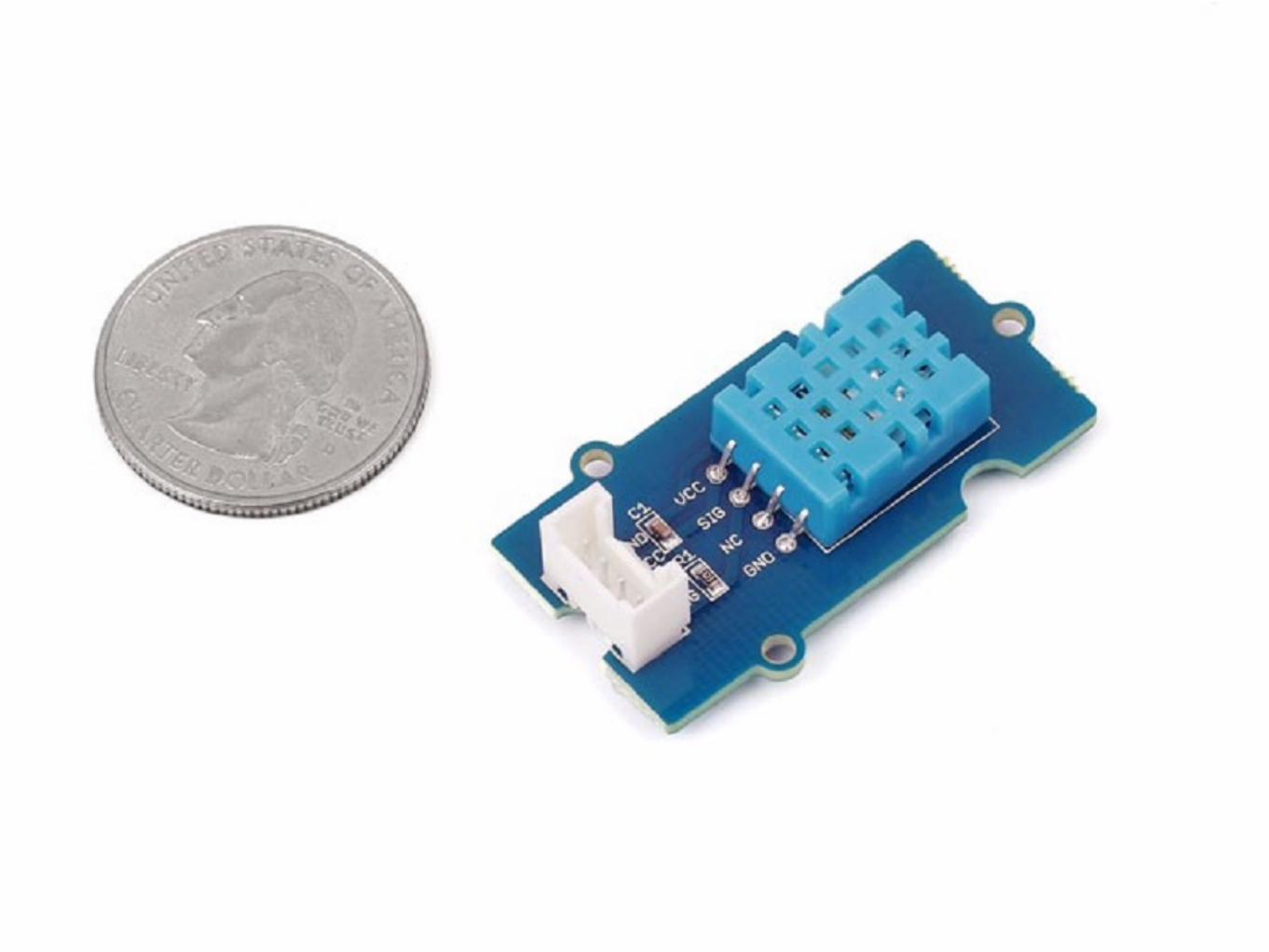
Grove-Line finder is designed for line-following robot. It has an IR emitting LED and an IR sensitive phototransistor. It can output digital signal to a microcontroller so that the robot can follow a black line on white background, or vice versa.

Grove - LCD RGB Backlight



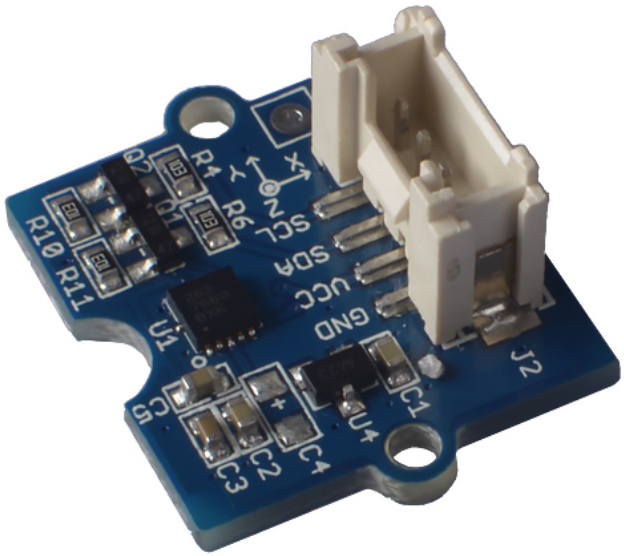
This Grove - LCD RGB Backlight module is a 16 Characters 2 Lines LCD display, it uses I2C bus interface to communicate with the development board, thus these will reduce the pin header from 10 to 2 which is very convenient for the Grove system. This LCD display module also supports customise characters, you can create and display heart symbol or stick-man on this LCD module through a simple coding configuration.

Grove - Temperature & Humidity Sensor(DHT11)



This Temperature & Humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It has excellent reliability and long term stability. Please note that this sensor will not work for temperatures below 0 degree.

Grove - 3-Axis Digital Accelerometer



3-Axis Digital Accelerometer is the key part in projects like orientation detection, gesture detection and Motion detection. This 3-Axis Digital Accelerometer(±1.5g) is based on Freescale's low power consumption module, MMA7660FC. It features up to 10,000g high shock survivability and configurable Samples per Second rate. For generous applications that don't require too large measurement range, this is a great choice because it's durable, energy saving and cost-efficient.

GETTING STARTED

Minimum Requirement

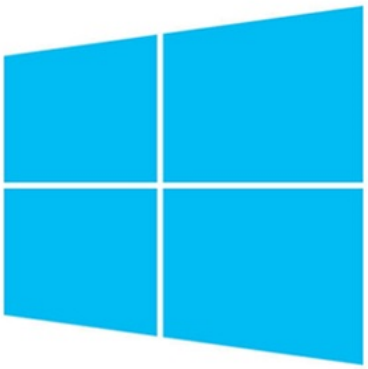
- Grove starter kit
- micro USB cable
- Computer with Arduino IDE

Basic Tutorial

Arduino IDE basic setup

Step 1. Install USB to Serial driver for Seeeduino Lotus V1.1

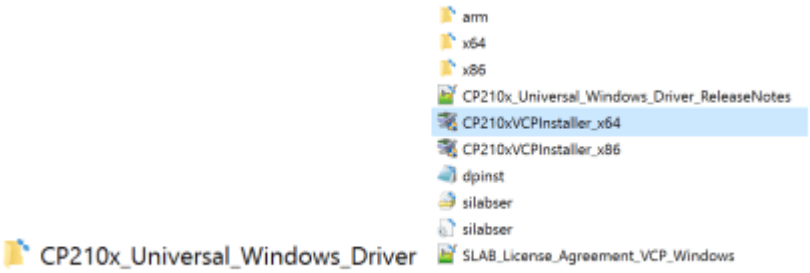
Seeeduino lotus Version 1.1 and above which adapts the CP2102N USB to serial chip, it adds support for majority operating systems include Windows, MacOS and Linux, please download and install the relevant driver for your operating system. Download links: Official Website: CP210x USB to UART Bridge VCP Drivers

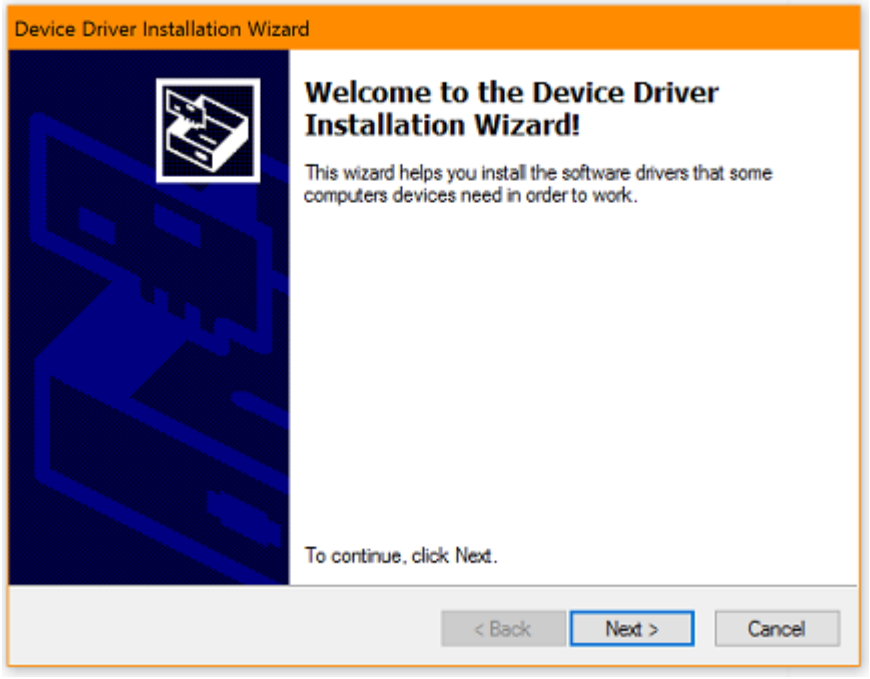


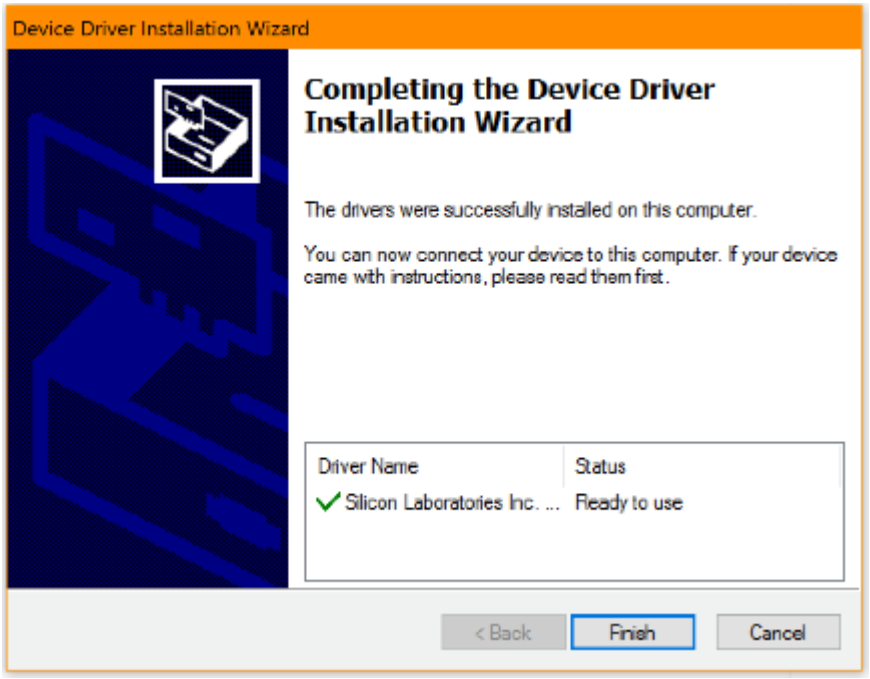
Install driver

Windows:

Extract/unzip the downloaded compressed driver file, open the extracted file and select install the relevant driver according to the bit of your operating system, in this case we selected 64bit, 32bits OS user should select _x86 file, follow the install wizard to install.

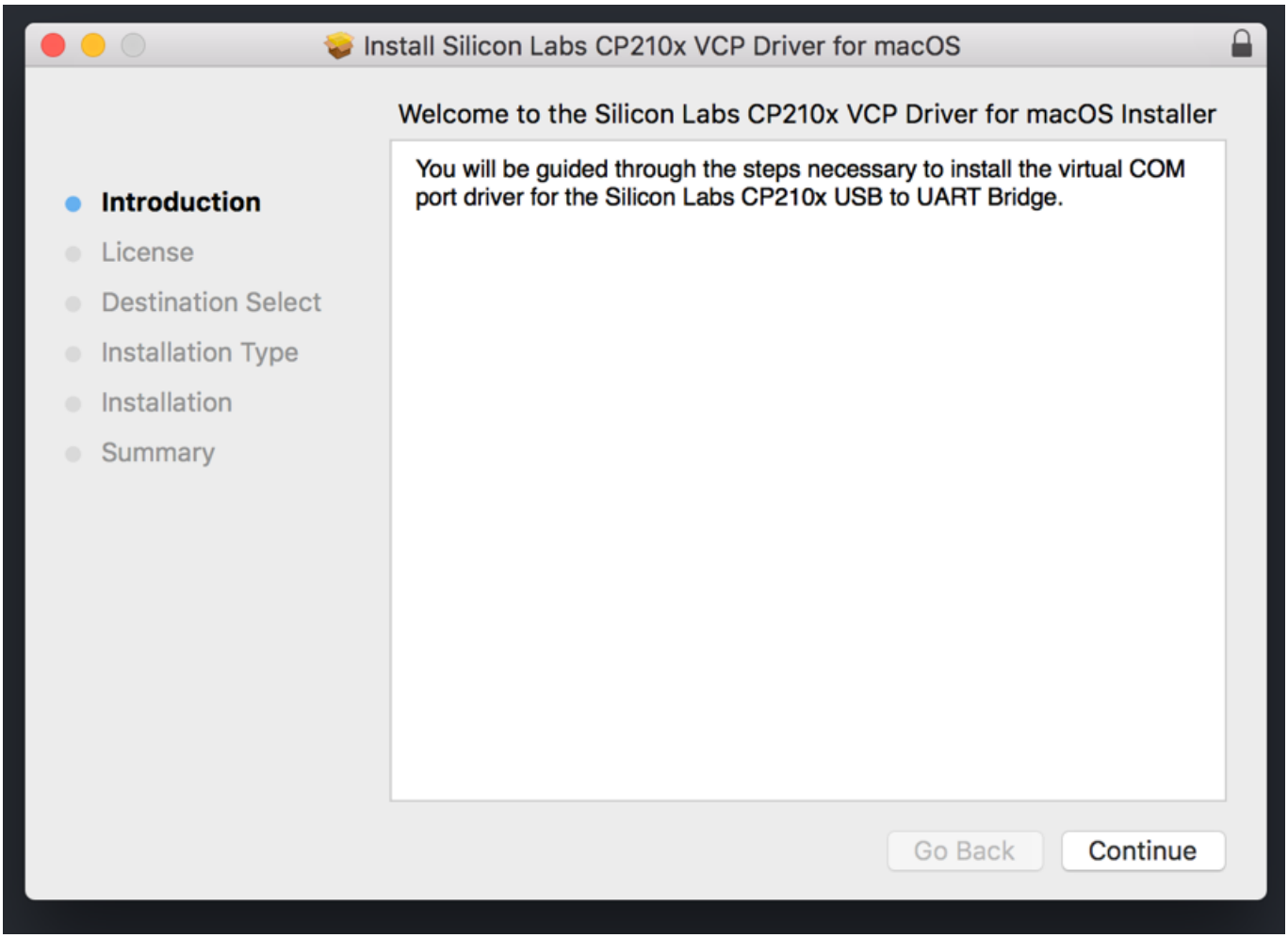






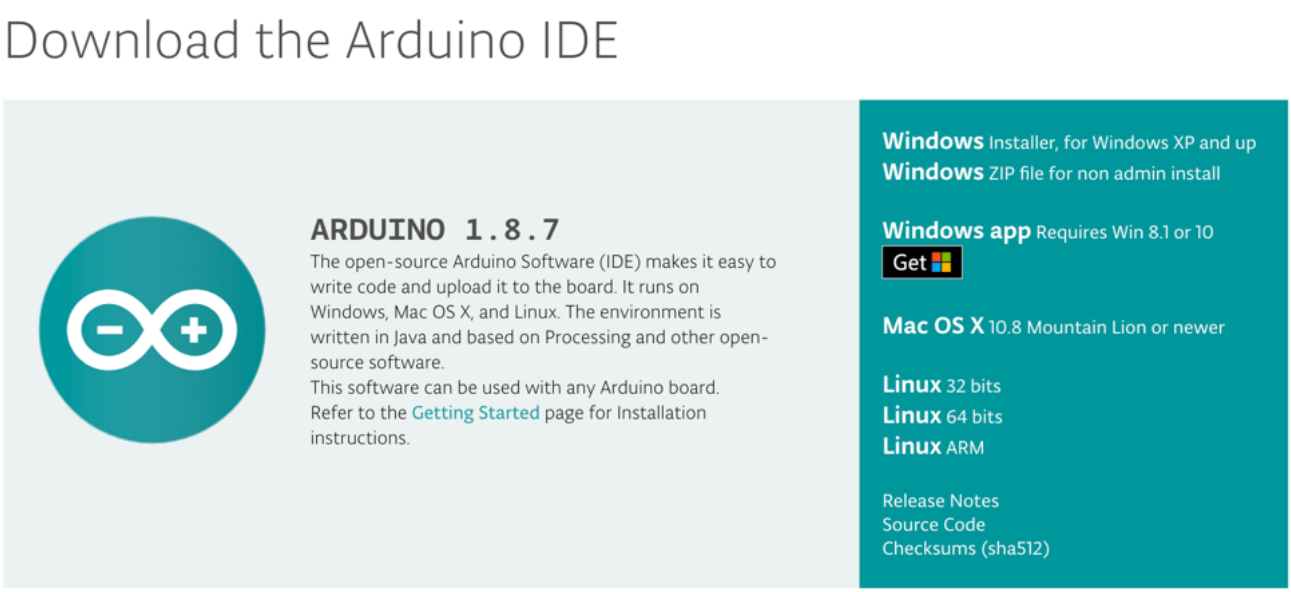
Mac OS:

Double click “Silicon Labs VCP Driver.pkg” file, and follow the setup wizard to install.



Step 2. Download and Install Arduino IDE

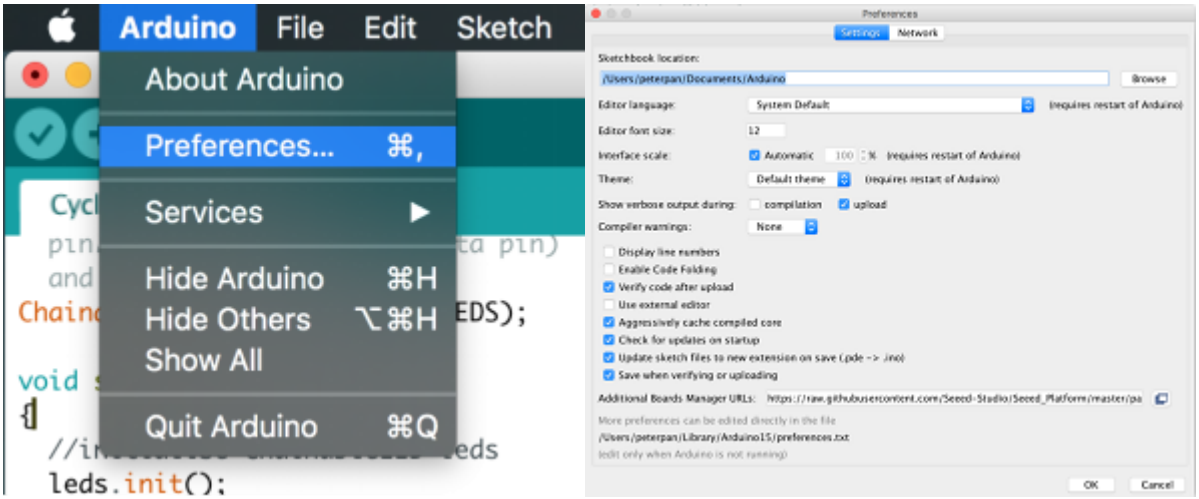
Please download and install the Arduino IDE according to your operating system.

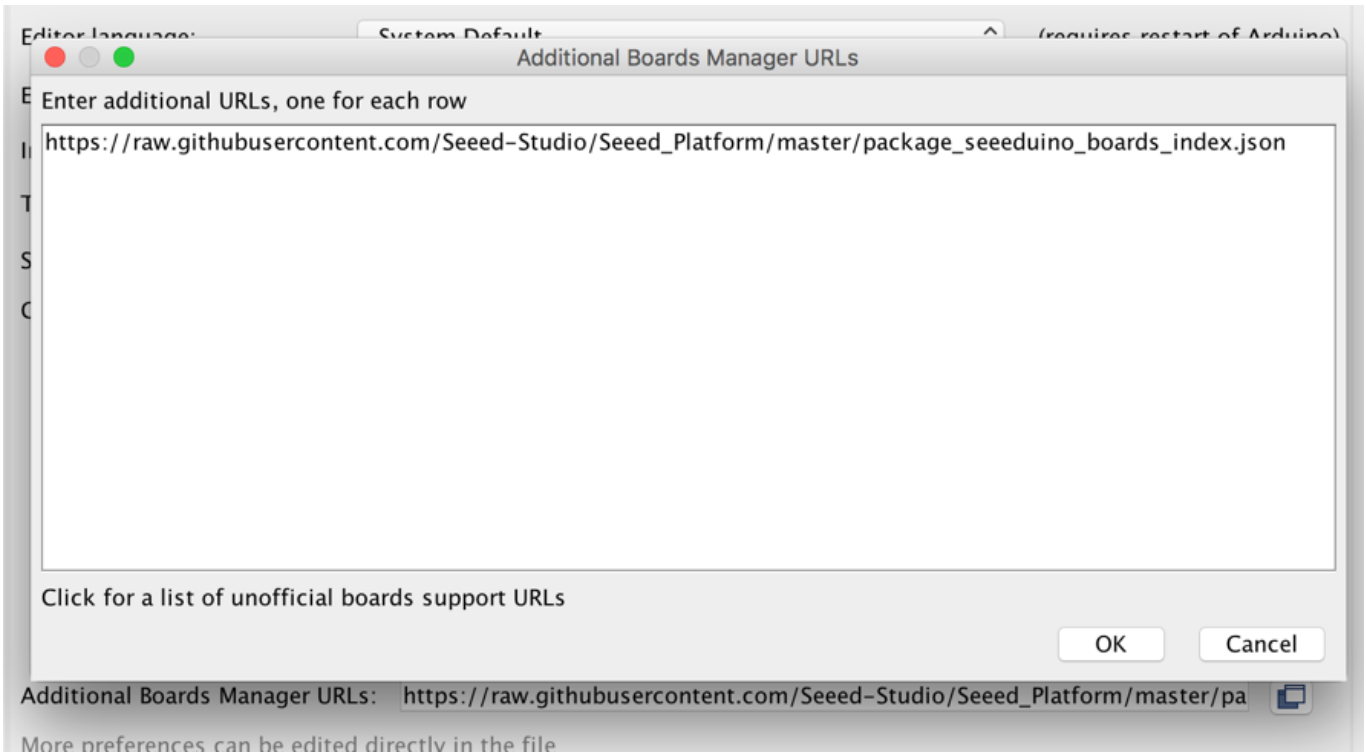


Step 3. Add library for Seeeduino Lotus

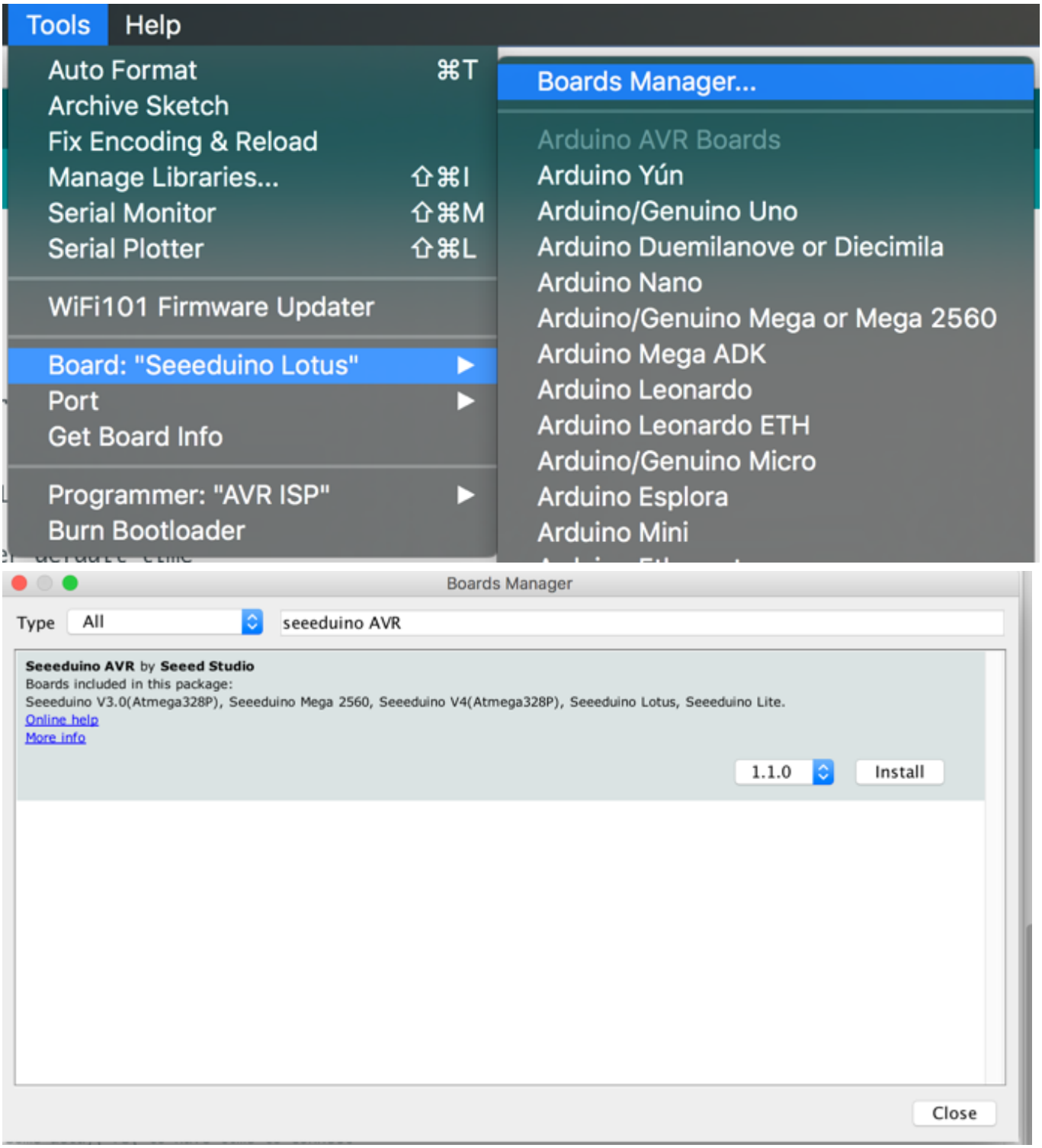
- Open Arduino | Preferences, from the preferences window under settings page find Additional Boards Manager URLS, copy & paste the Library URL into the text box, then press ok to take effect. Library URL:

https://raw.githubusercontent.com/Seeed-Studio/Seeed_Platform/master/package_seeeduino_boards_index.json

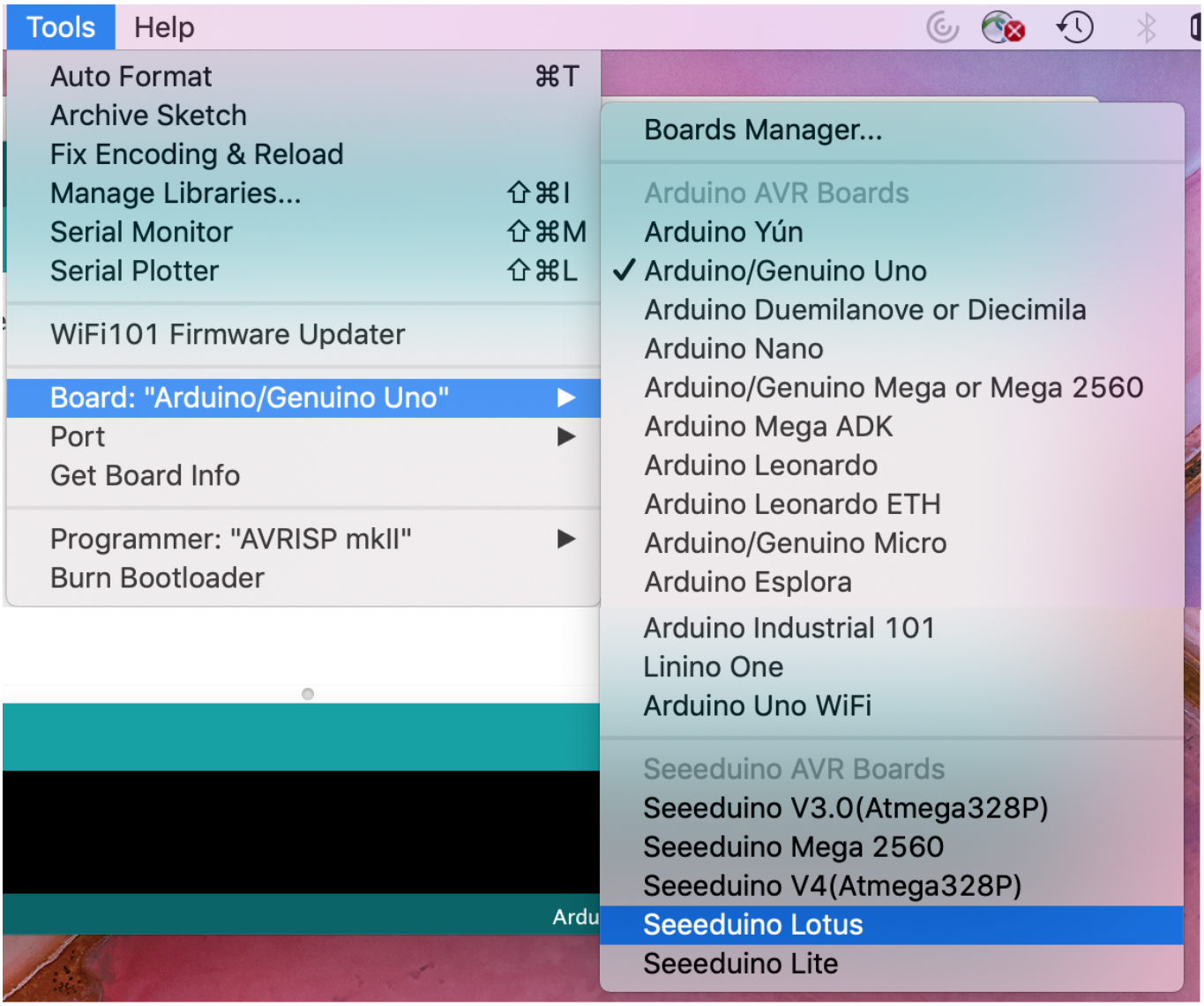




- Open Tools | Board: | Boards Manager, search for Seeeduino AVR and click install to install the Seeeduino AVR library. if you cannot see the Seeeduino AVR listed in the Boards Manager window, please repeat the first step and make sure the URL you entered is correct.

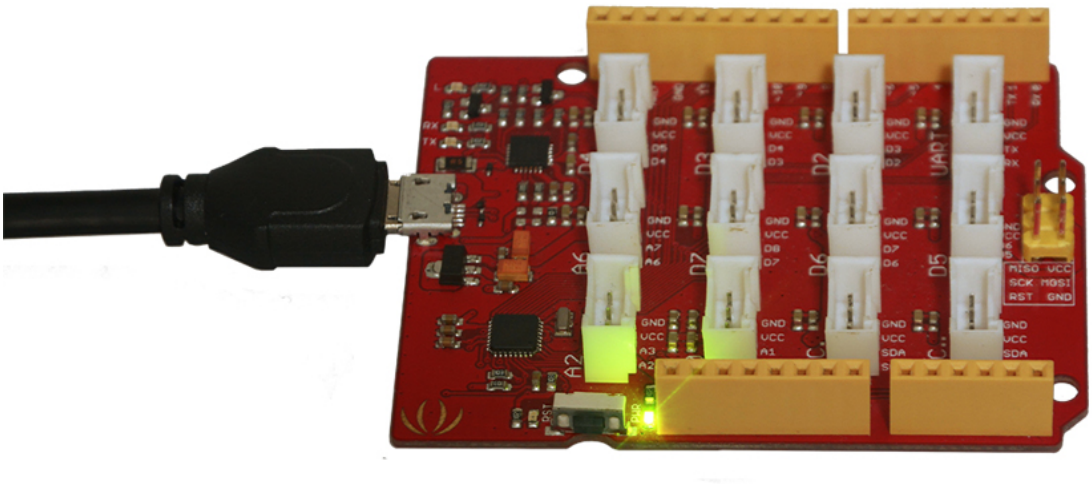


- Open Tools | Board: under the secondary list you should find Seeeduino AVR Boards section as shown, and please select the correct boards according to your development environment, in this tutorial we should select Seeeduino Lotus.



Step 4. Connect Seeeduino Lotus

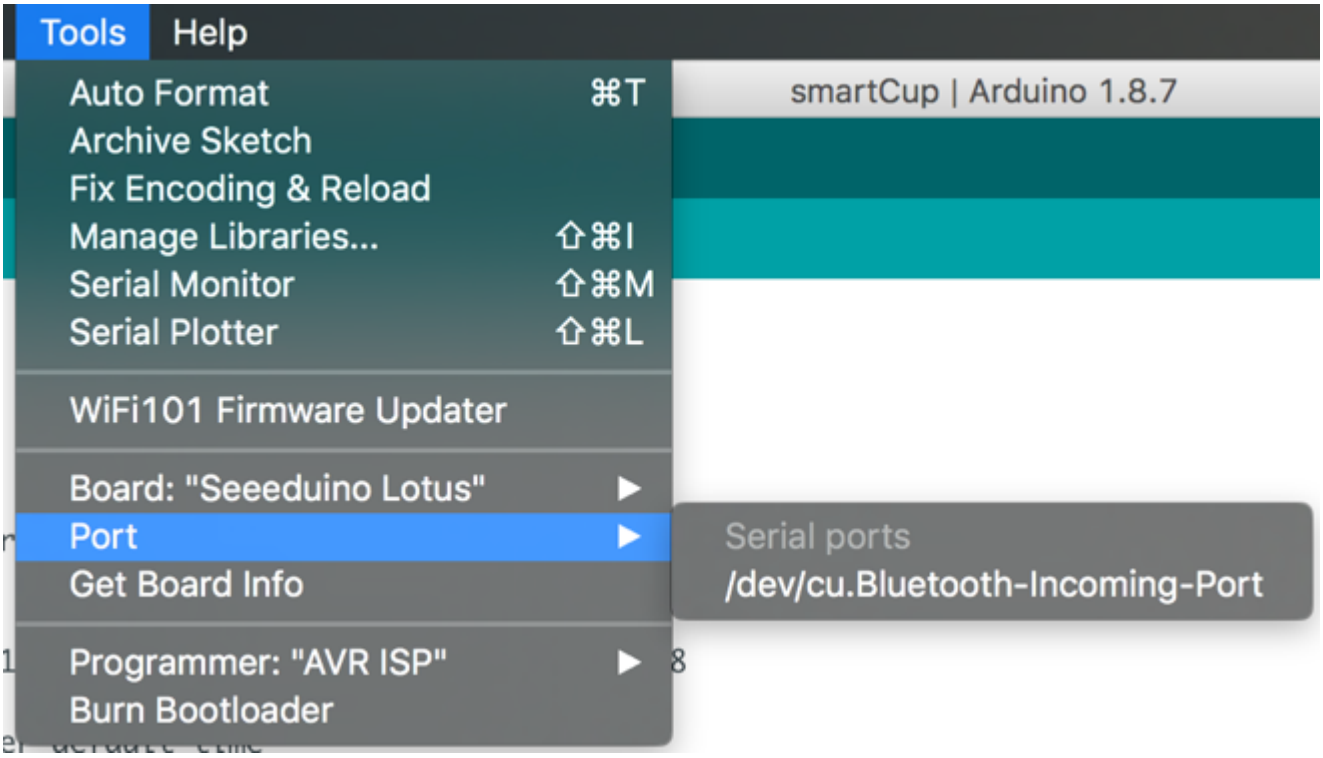
Please connect Seeeduino Lotus and computer through a Micro-USB cable, the green power LED on Seeeduino lotus should lit up.

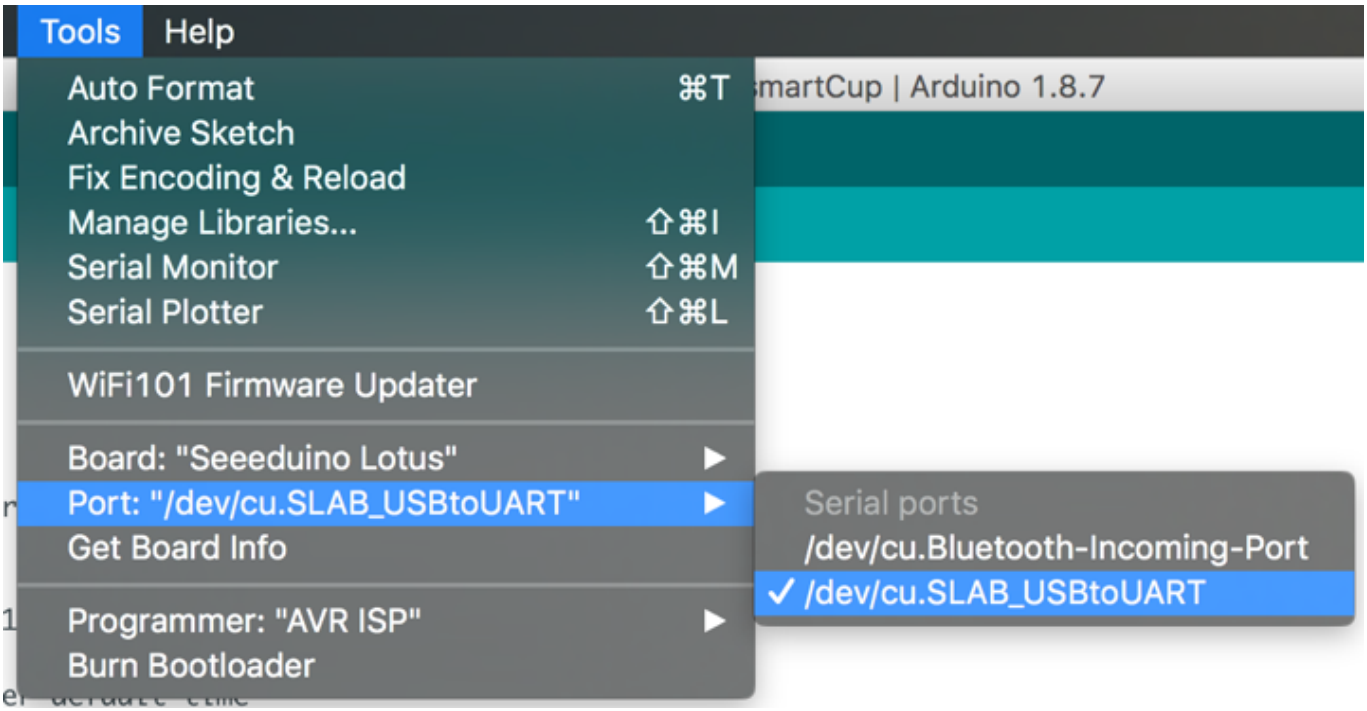


Step 5. Configure IDE for Seeeduino Lotus

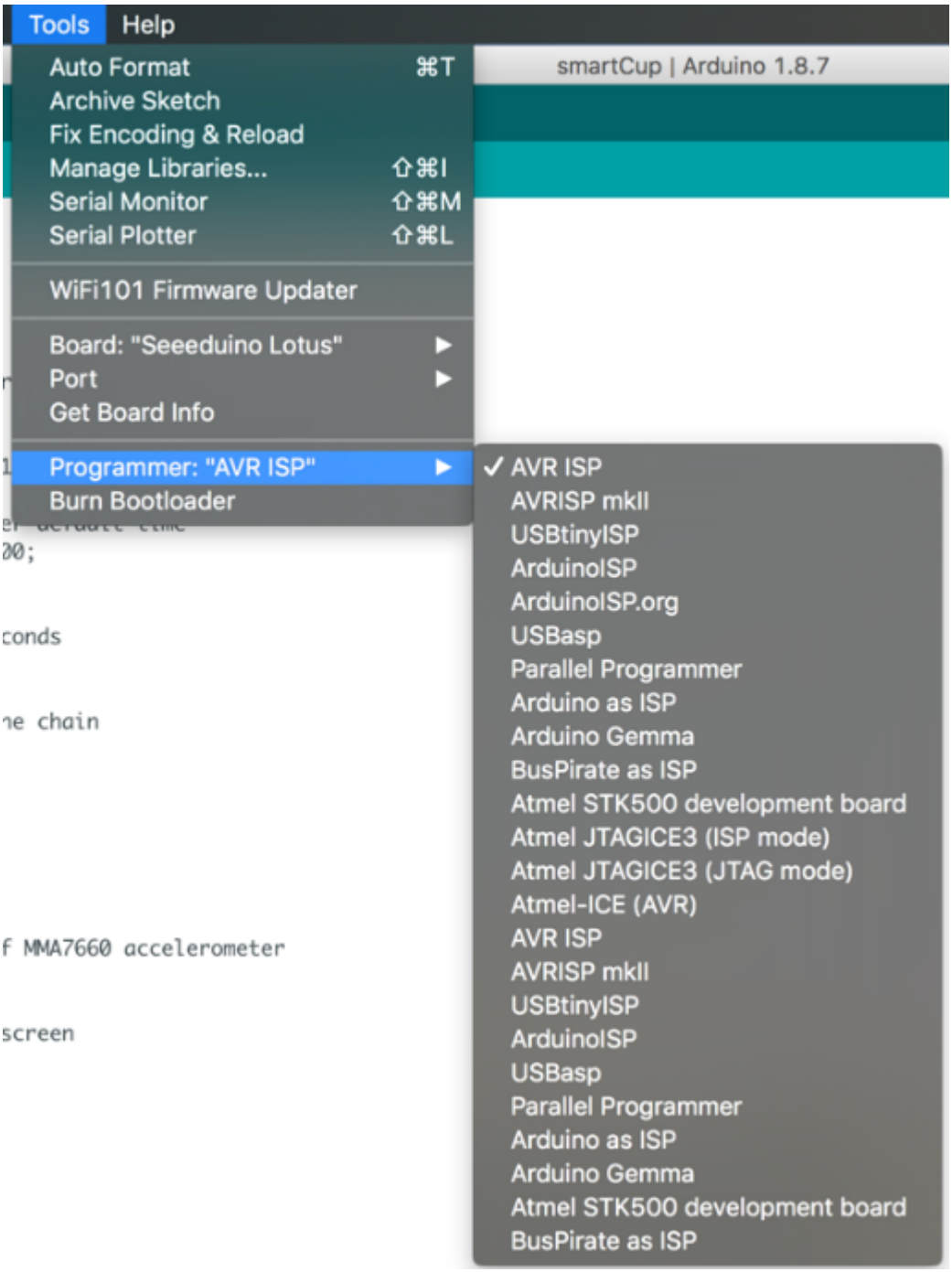
Please follow the steps as shown before, select “Seeeduino Lotus” under the Boards Manager.

Select the serial device of the Arduino board from the Tools | Serial Port menu. To find out the correct serial device, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port. The entry you selected should contain “SLAB_USB”.





Set “Tools | Programmer” as “AVR ISP”.

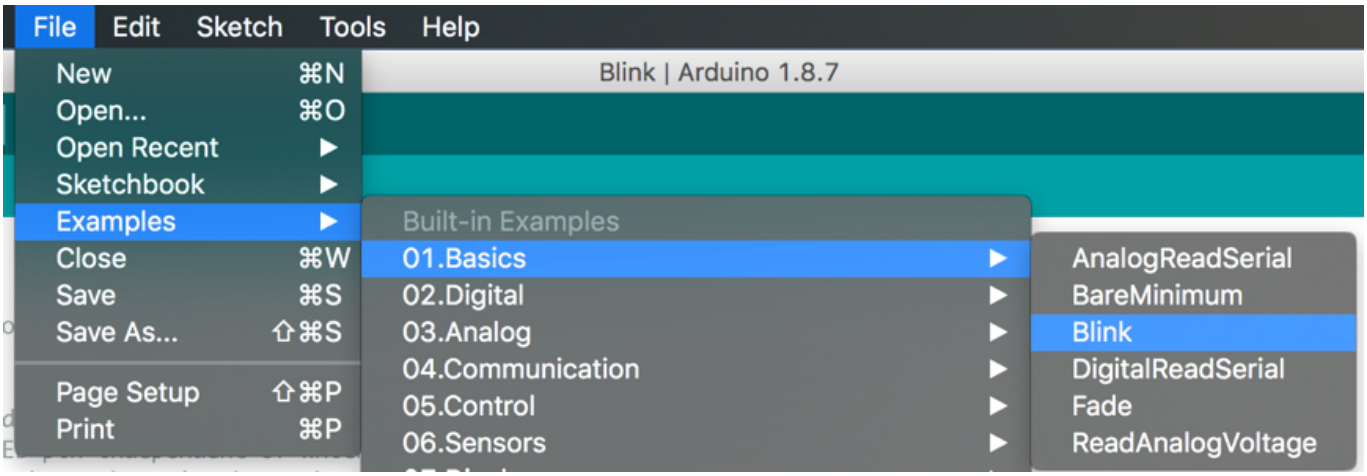


Blink Demo

After all the basic set up of Arduino IDE, we can now test the blink demo code on the Seeeduino Lotus development board. Note: You should complete the steps above to continue the following.

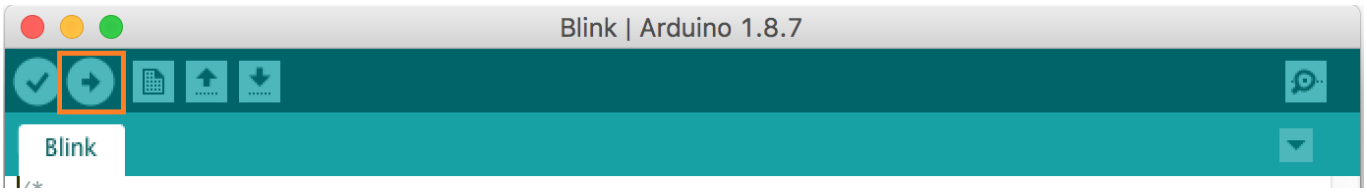
Select Blink Demo From Menu

Select File | Examples | 01 Basics | Blink from menu bar, the blink example code should appear in the new window.

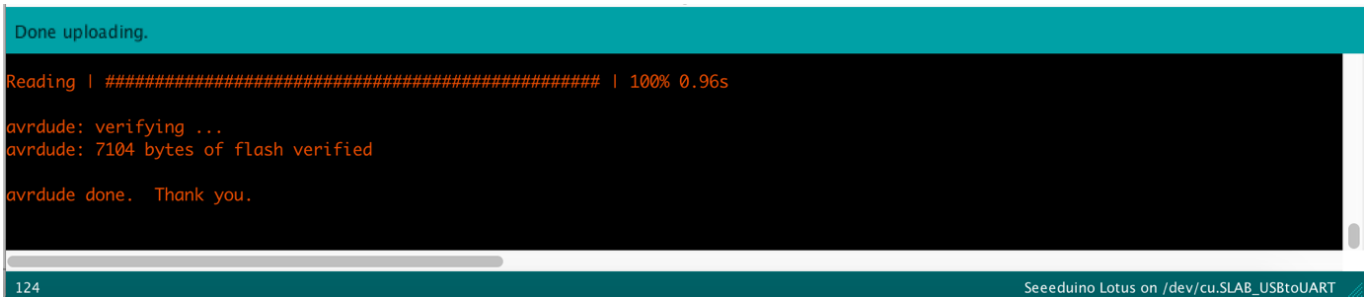


Upload code

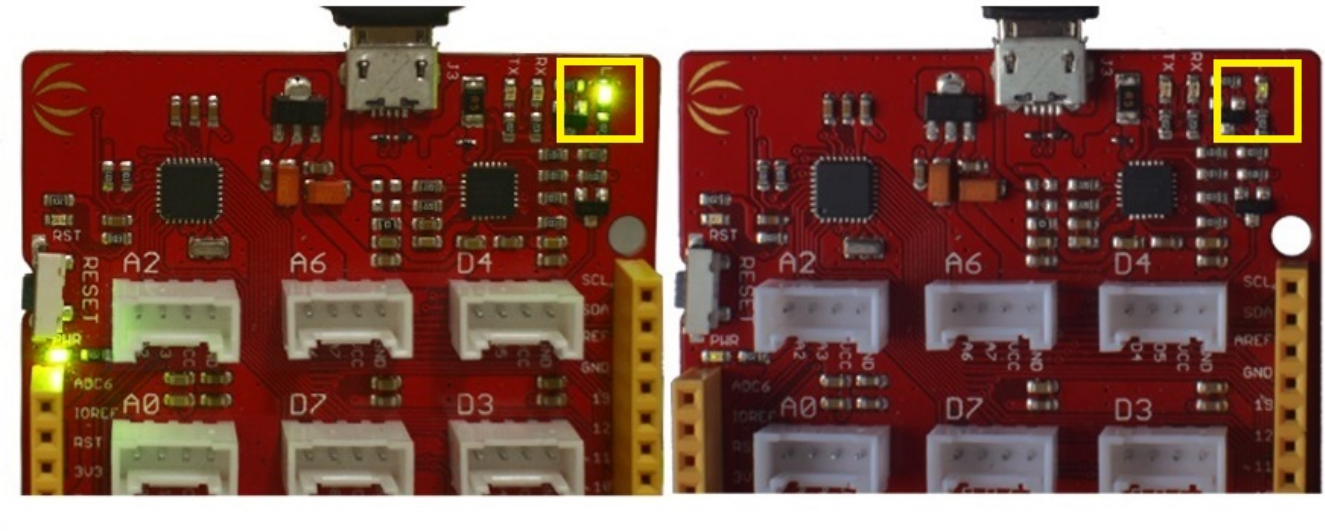
Please ensure the correct Board, Port and Programmer are selected under tools menu. Now we can upload the code into the Lotus dev board by press the right arrow icon on the top left corner of the IDE.



Once the code uploaded successfully, the text “avrdude done. Thank you.” should appear in the log window of the IDE.



Now you should see the built-in LED flashes in a one second interval.



Explanation of the blink code

void setup() is the setup function only runs one time when you press reset or power the board.

```
void setup() {
}
```

initialize digital pin LED_BUILTIN as an output.

```
pinMode(LED_BUILTIN, OUTPUT);
```

void loop() is the loop function runs over and over again forever.

```
void loop() {
}
```

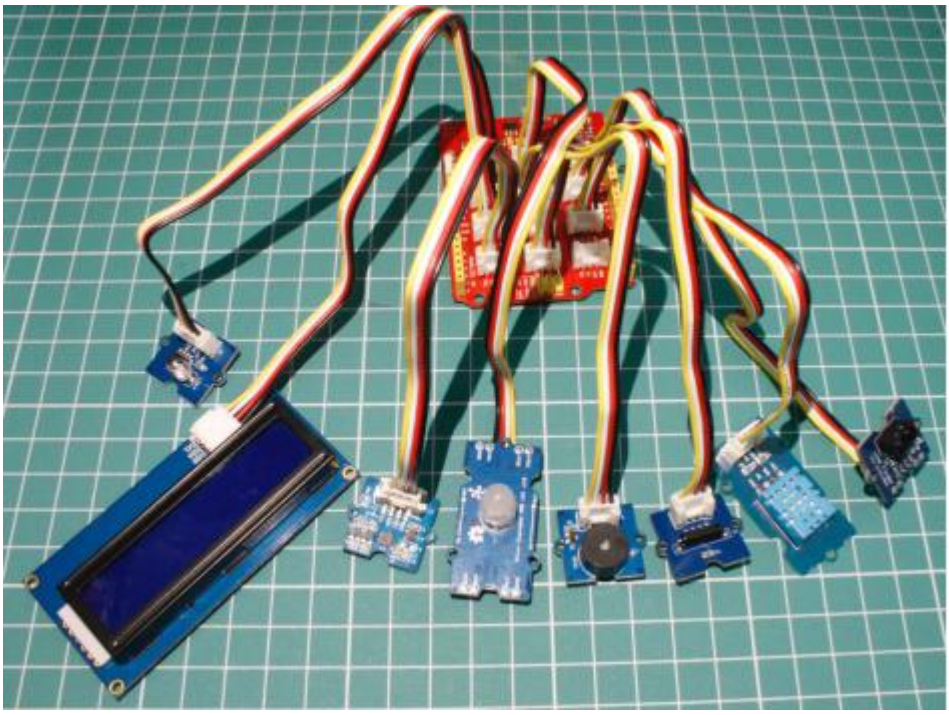
digitalWrite()is to set the LED_BUILTIN pin as voltage level HIGH, which means to turn on the LED. Similarly to turn off the LED just set the voltage level to LOW by changing the code HIGH to LOW.

```
digitalWrite(LED_BUILTIN, HIGH);
digitalWrite(LED_BUILTIN, LOW);
```

delay() means to pause the program, the number inside the bracket means the amount of time(in milliseconds) to pause(delay).

```
delay(1000);
```

Grove Starter Kit 10 Sessions Tutorial



Goal

This section consists of 10 tutorials, the tutorials can be divided into two parts, the first 8 sessions introduce the basic operation of each individual module from this starter kit, and the last 2 sessions uses example cases to show how the modules can be combined and applied in real life applications.

Prerequisite

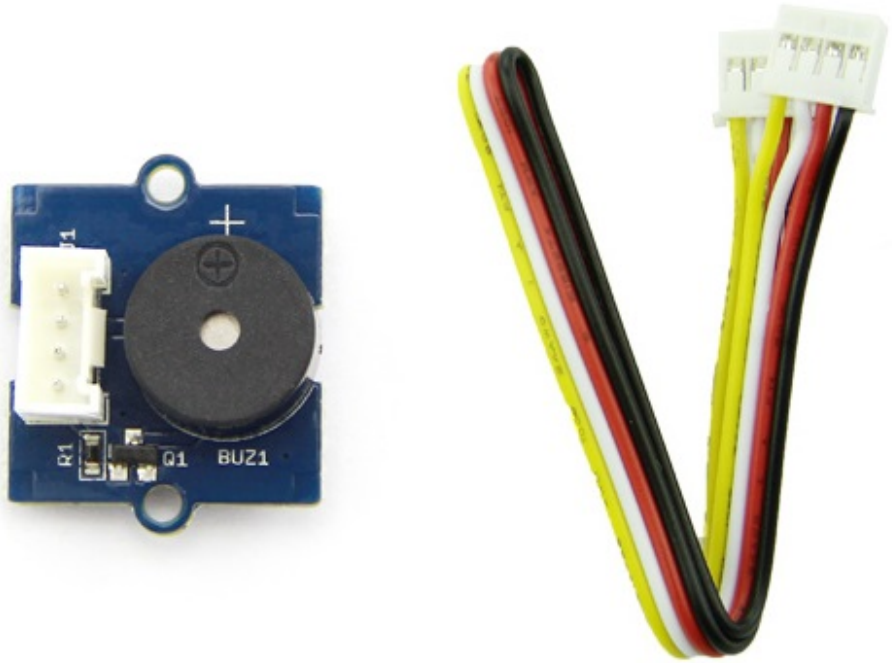
The fundamental knowledge on operate Seeeduino Lotus with Arduino IDE and the coding skill are essential to this tutorial. Therefore, please make sure you have completed the basic setup tutorial above and successfully installed the USB to serial driver on your operating system for Seeeduino Lotus, finished the Blink LED demo and ensure it is fully working with the Seeeduino Lotus board.

Learning outcome

- Be able to operate Arduino IDE to write code for Seeeduino Lotus V1.1 to drive modules from Grove Starter Kit.
- Be able to identify the type of modules include in this Kit and their applications.

- Be able to demonstrate each components of Grove Starter Kit and utilise the relevant module to your own projects after this tutorial

Session 1: Grove - Buzzer



Objective

Using buzzer to generate some noise and also setting specific frequency to produce some tones.

Key knowledge

- Buzzer module is an actuator.
- Use digital signal to make buzz noise
- Produce specific tone by setting frequency accordingly
- Use tone(pin, frequency, duration) function to make buzzer play music
- Learn how to use “for loop” in Arduino IDE

Hardware requirement

Self-prepare

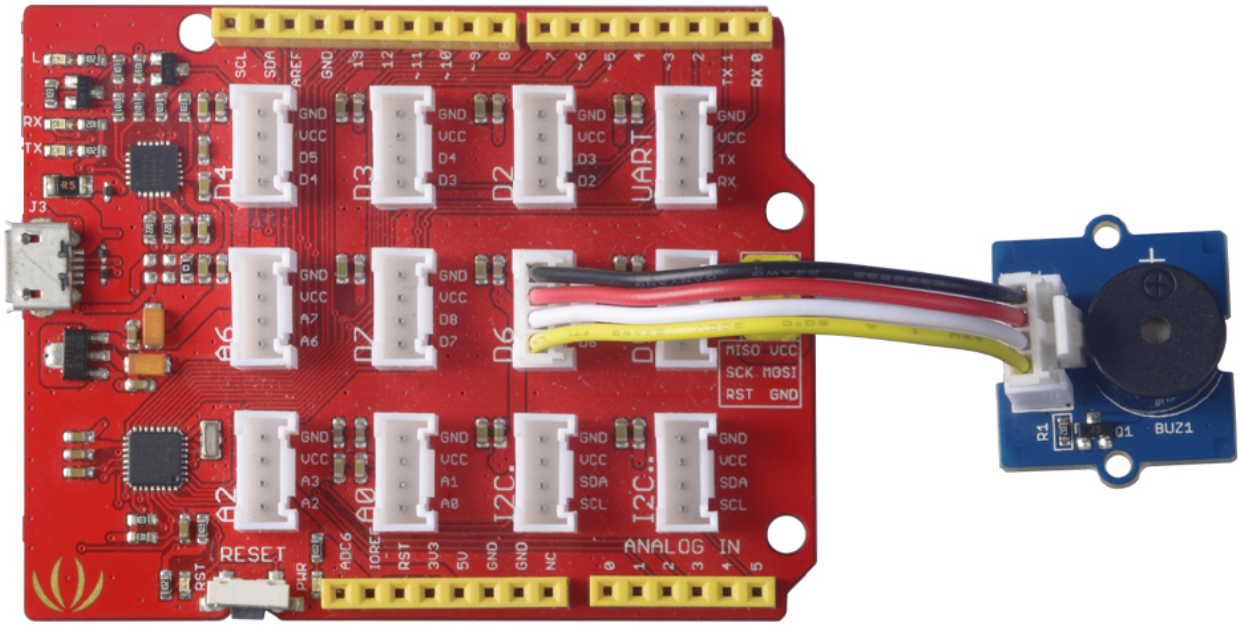
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – Buzzer module

Hardware connection

Step 1. Please use the Grove cable to connect Grove - Buzzer module to D6 port of Seeeduino Lotus



Step 2. Link Seeeduino Lotus with computer by a micro USB cable.

Software programming

Example 1:Use digital logic high/low to make the buzzer “buzz”

Step 1: Copy & Paste the following code into Arduino IDE

```
//assign buzzer as pin 6
#define buzzer 6

void setup()
{
  //set buzzer as output
  pinMode(buzzer, OUTPUT);
}

void loop()
{ //turn on buzzer(set logic level high)
  digitalWrite(buzzer, HIGH);
  //wait 1s
  delay(1000);
  //turn off buzzer(set logic level low)
  digitalWrite(buzzer, LOW);
  //wait 1s
  delay(1000);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

!!!Success When the code finishes upload, you can hear “buzz” with a 1 second gap in between sound.

Example 2:Use different frequency to make the buzzer generate different tone.

Step 1: copy & paste the following code into Arduino IDE

```
//assign buzzer as pin 6
#define buzzer 6

void setup()
{
  /* tone(pin, frequency, duration) */
  //set buzzer pin to play 264Hz for 300ms
  tone(buzzer, 262, 300);
  //wait 1s
  delay(1000);

  //set buzzer pin to play 297Hz for 300ms
  tone(buzzer, 294, 300);
  //wait 1s
  delay(1000);

  //set buzzer pin to play 330Hz for 300ms
  tone(buzzer, 330, 300);
  //wait 1s
  delay(1000);
}

void loop()
{
  // no need to repeat the tone.
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

!!!Success When the code finishes upload, you should hear the buzzer is making “Do、 Re、 Mi” sound.

Example 3:Use tone(pin, frequency, duration)function to make music from buzzer

Step 1: copy & paste the following code into Arduino IDE

```
// initalise the frequency of the notes
```

```
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_C4  262
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_G4  392
#define NOTE_C5  523

//assign buzzer as pin 6
#define buzzer 6

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_F4, NOTE_E4,
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_G4, NOTE_F4,
  NOTE_C4, NOTE_C4, NOTE_C5, NOTE_A4, NOTE_F4, NOTE_E4, NOTE_D4,
  NOTE_AS4, NOTE_AS4, NOTE_A4, NOTE_F4, NOTE_G4, NOTE_F4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  8, 8, 4, 4, 4, 2,
  8, 8, 4, 4, 4, 2,
  8, 8, 4, 4, 4, 4, 4,
  8, 8, 4, 4, 4, 2,
};

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0 ; thisNote < 25 ; thisNote++) {

    // to calculate the note duration, take one second divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000 / noteDurations[thisNote];
    tone(buzzer, melody[thisNote], noteDuration);

    // to distinguish the notes, set a minimum time between them.
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    noTone(buzzer);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

!!!success When the code finishes upload, you should hear a melody from buzzer, please what song is it.

Further Explore

Please checkout Brett Hagman's GitHub repo "Tone" to create tones and music.

Session 2: Grove - Tilt Switch

Objective Use tilt switch module to turn on/off the built-in LED on the Seeeduino Lotus, and also use the tilt switch to make the buzzer module from previous session to buzz.

Key knowledge

- Tilt Switch is a signal input module
- The operation on tilt switch
- uses digitalRead(pin) function get the input logic signal from tilt switch which is HIGH for switched on, and LOW for switched off.
- if(condition){}else{}function and comparison operator such as !=(not equal to), <(less than), <=(less than or equal to), ==(equal to), >(greater than) and >=(greater than or equal to).

Hardware requirement

Self-prepare

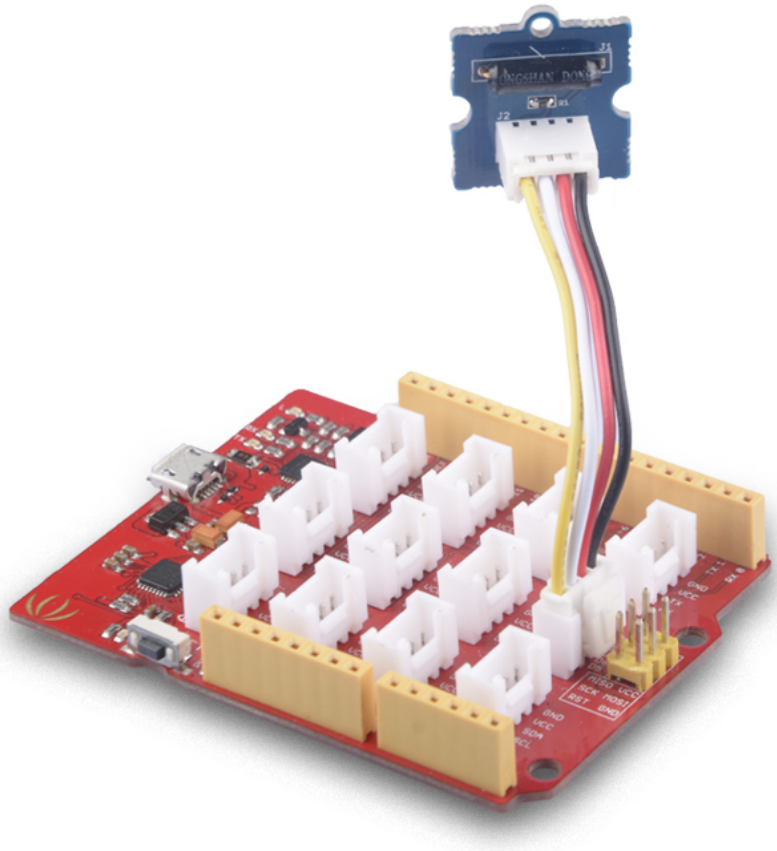
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – Tilt Switch
- Grove – Buzzer

Hardware connection

Step 1: Connect Grove – Tilt Switch to D5 port of Seeeduino Lotus.



Step 2: Link Seeeduino Lotus with computer by a micro USB cable

Software programming Example 1:Observe the tilt switch behaviour by using Serial Monitor

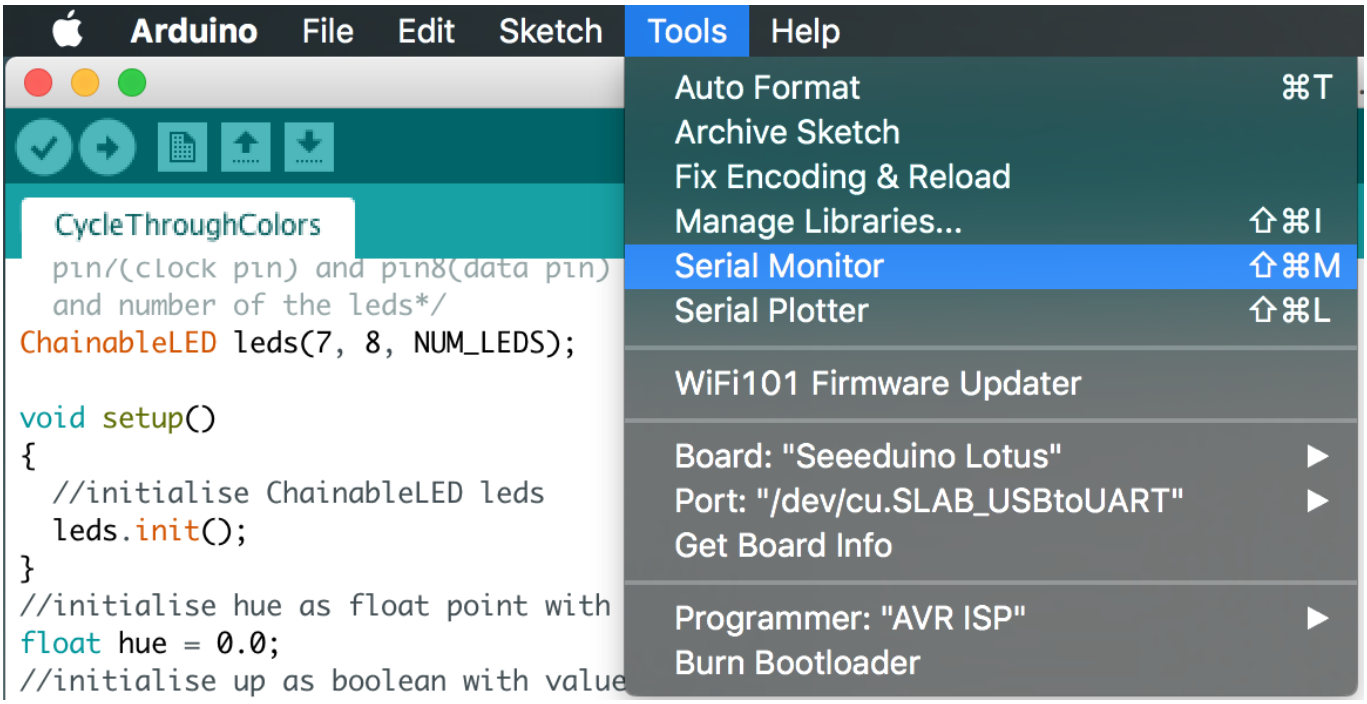
Step 1: copy & paste the following code into Arduino IDE

```
//assign name tiltswitchPin to pin 5
#define tiltswitchPin 5
//creates a integer variable called 'val' to store read value
int val;

void setup()
{
  //set pinMode of tiltswitchPin to input
  pinMode(tiltswitchPin, INPUT);
  // opens serial port, sets data rate to 9600 bps
  Serial.begin(9600);
}

void loop()
{ //read the tilt switch input
  val = digitalRead(tiltswitchPin);
  //display the tilt switch status, 1 is on, 0 is off.
  Serial.println(val);
}
```

Step 2: Upload code into Seeeduino Lotus

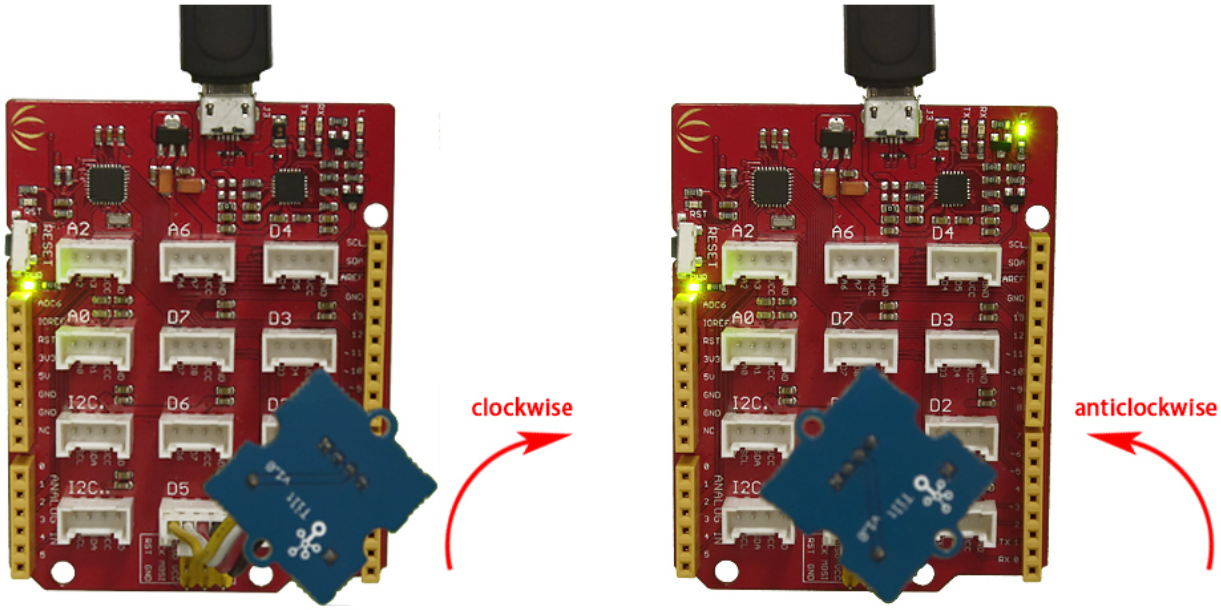
Step 3: Open Serial Monitor

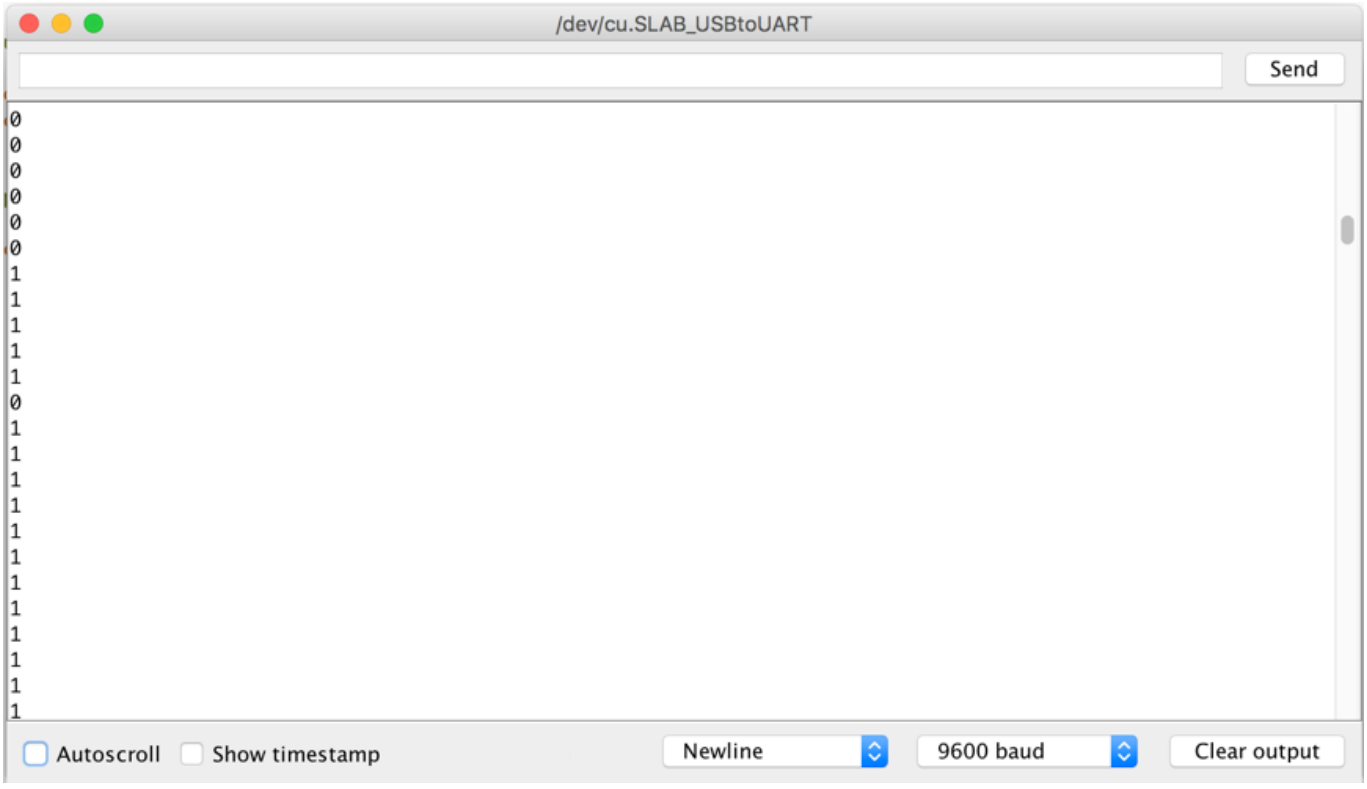
to open serial monitor please select Tools | Serial Monitor from menu bar, or simply click the magnifier icon on the tool bar. Note: Please wait for the code finish uploaded before open serial monitor.



Step 4: Observe result

Please tilt the tilt switch in both directions, you should see “1” or “0” displayed in serial monitor, now you can find the right orientation for tilt switch to turn on/off.





Example 2:Use tilt switch to turn on/off built-in LED

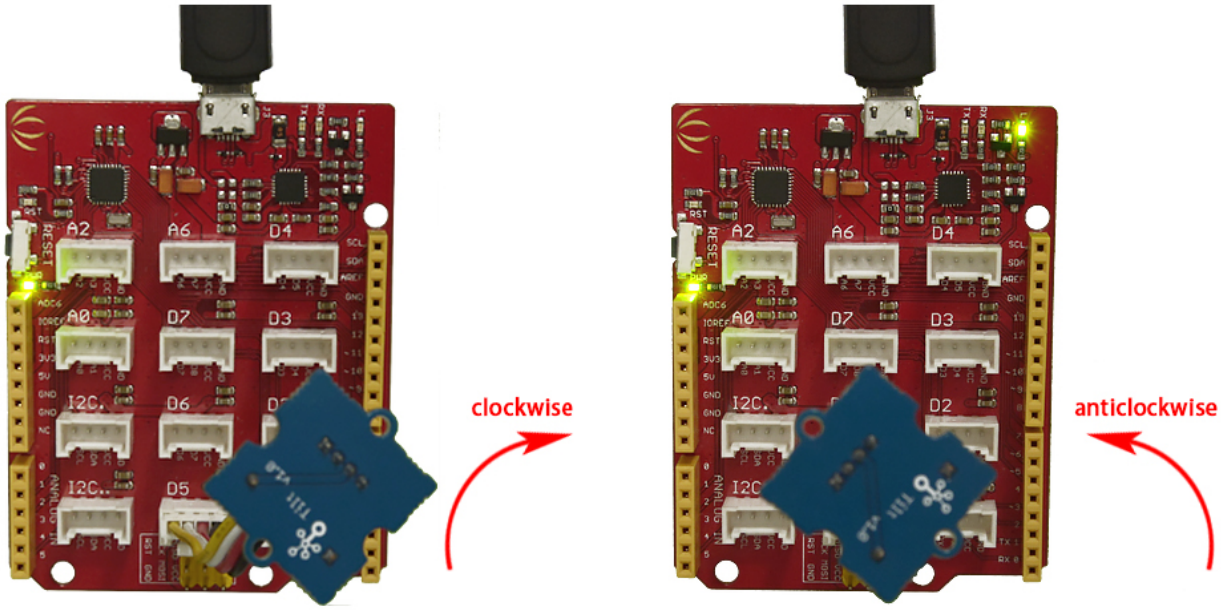
Step 1: copy & paste the following code into Arduino IDE

```
//set title of pin 5 as tiltSwitch
#define tiltSwitch 5

void setup()
{ //set pin 5(tilt switch) as input pin
  pinMode(tiltSwitch, INPUT);
  //set pin 13(Builtin LED) as output pin
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{ //read the status of tilt switch
  if (HIGH == digitalRead(tiltSwitch)) {
    /*
       if the logic level of tilt switch
       is high turn on the builtin LED
    */
    digitalWrite(LED_BUILTIN, HIGH);
  } else
  {
    //otherwise turn off the builtin LED
    digitalWrite(LED_BUILTIN, LOW);
  }
}
```

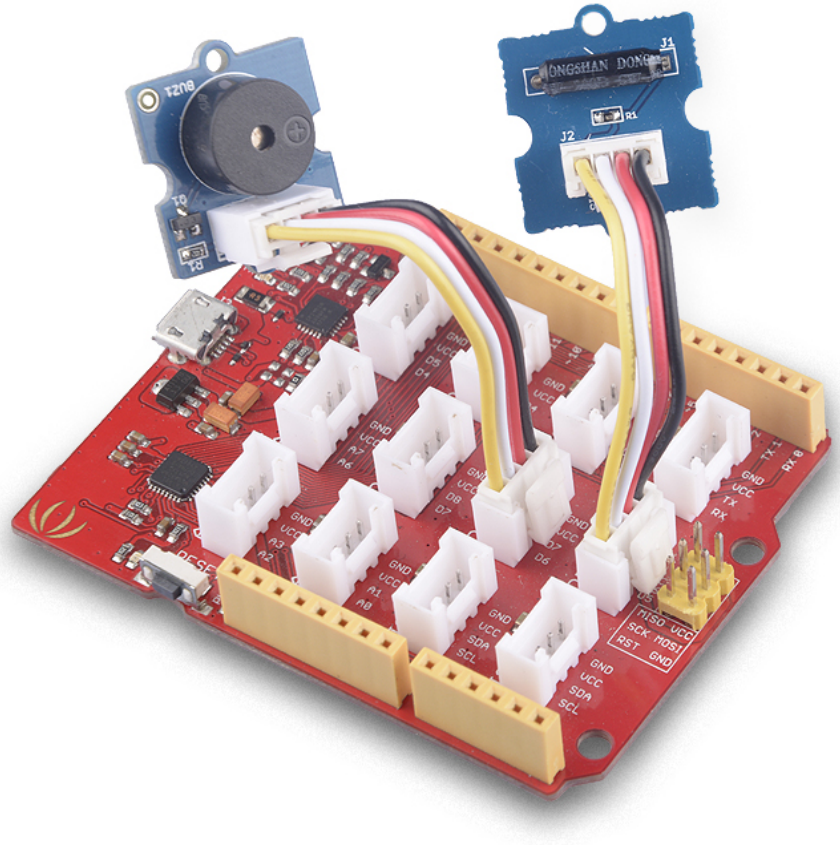
Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

!!!Success Now you should be able to turn on/off the built-in LED on the Seeeduino Lotus by tilting the tilt switch in the right orientation.



Example 3: Use tilt switch to pause and play ringtone from the buzzer Please connect Grove – Buzzer module to D6 port of Seeeduino Lotus



Step 1: copy & paste the following code into Arduino IDE

```
// initalise the frequency of the notes
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_C4  262
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_G4  392
#define NOTE_C5  523

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_F4, NOTE_E4,
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_G4, NOTE_F4,
  NOTE_C4, NOTE_C4, NOTE_C5, NOTE_A4, NOTE_F4, NOTE_E4, NOTE_D4,
  NOTE_AS4, NOTE_AS4, NOTE_A4, NOTE_F4, NOTE_G4, NOTE_F4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  8, 8, 4, 4, 4, 2,
  8, 8, 4, 4, 4, 2,
```

```
  8, 8, 4, 4, 4, 4, 4,
  8, 8, 4, 4, 4, 2,
};

//set title of pin 5 as tiltSwitch
#define tiltSwitch 5
//set title of pin 6 as buzzer
#define buzzer 6
// set variable currentNote to store latest note played
int currentNote;

void setup()
{
  //set pin 5(tilt switch) as input pin
  pinMode(tiltSwitch, INPUT);
}

void loop()
{
  /*read the status of tilt switch
if the logic level of tilt switch
is high, start play music */
  if (HIGH == digitalRead(tiltSwitch)) {

for (int thisNote = currentNote ; thisNote < 25 ; thisNote++) {
  // to calculate the note duration, take one second divided by the note type.
  //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
  int noteDuration = 1000 / noteDurations[thisNote];
  tone(buzzer, melody[thisNote], noteDuration);

  // to distinguish the notes, set a minimum time between them.
  int pauseBetweenNotes = noteDuration * 1.30;
  delay(pauseBetweenNotes);

  /*reset the currentNote to the 0
is the music is finished*/
  if (thisNote >= 24) {
currentNote = 0;
  }

  /*druing the music read the status
of tilt switch if the logic level
of tilt switch is LOW, stop play
music and store the previous played
tone and jump to next tone*/
  if (LOW == digitalRead(tiltSwitch)) {

//store the current note(thisNote) to currentNote
currentNote = thisNote;
//set the next note ready to play by increase currentNote by 1 increament
currentNote ++;
/*reset the currentNote to the beginning
  is the music is finished*/
if (currentNote >= 25)
```

```
  {
    //restart the music from beginning by reset the currentNote to 0,
    currentNote = 0;
  }
  //if the tilt switch is set to logic level low, stop play music
  break;
    }
  }
    }
  }
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

!!!Success Now you should be able to pause the ringtone by tilting the tilt switch to off position, and resume the music tone by tilting the tilt switch to on position.

Further Explore

After this session, you can attach the tilt switch sensor module to your toolbox lid, so when you lift the lid, triggers the tilt switch to turn on, then you can set a delay for a short period to activate buzzer to make some tones which reminds you the lid is still open, so you will not to forget to close the lid after you've finished using the toolbox.

Session 3: Grove – Chainable RGB LED

Objective

Use code to control the chainable RGB LED to show different color, and switch between colors by using tilt switch.

Key knowledge

- Grove – Chainable RGB LED is an actuator
- Import library for grove module
- Use setColorHSB() function to control the hue, saturation and brightness of the LED module
- Use setColorRGB() function to control the color and brightness of the LED module
- Use % (modulo) operation to find remainder, eg. 5%2=1, 9%3=0.
- Use switch(val)...case...; function

Hardware requirement

Self-prepare

- micro-USB cable
- computer with Arduino IDE and serial-to-USB driver installed

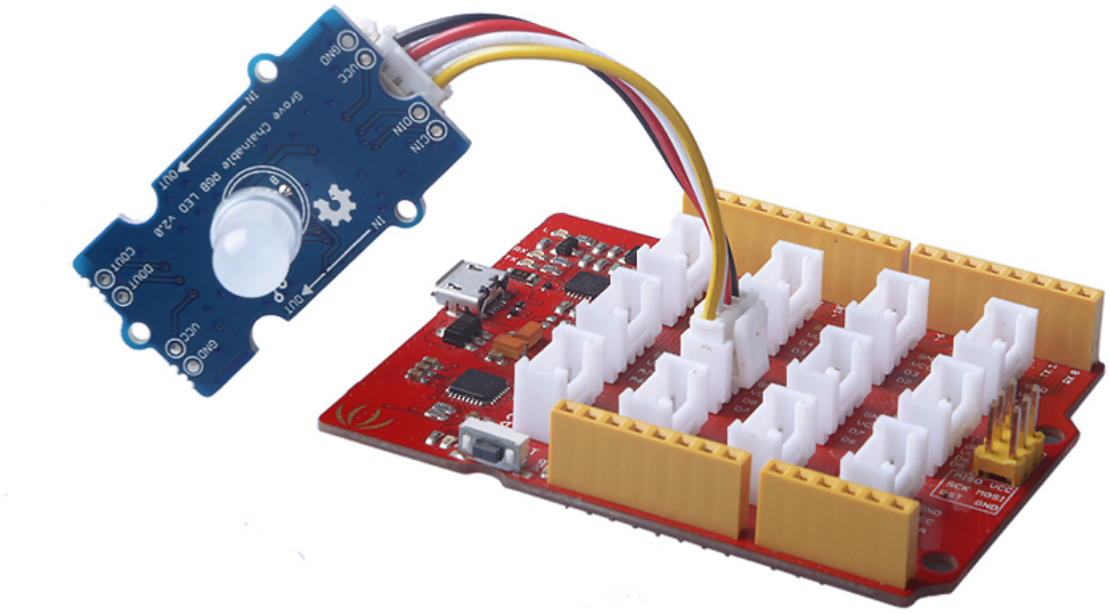
Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable

- Grove – Chainable RGB LED
- Grove – Tilt Switch
- Grove – Buzzer

Hardware connection

Step 1: Connect Grove – Chainable RGB LED to D7 port of Seeeduino Lotus, Note: please connect the G|V|DI|CI port from LED as shown below.



Step 2: Link Seeeduino Lotus with computer by a micro USB cable.

Software programming

Add Library

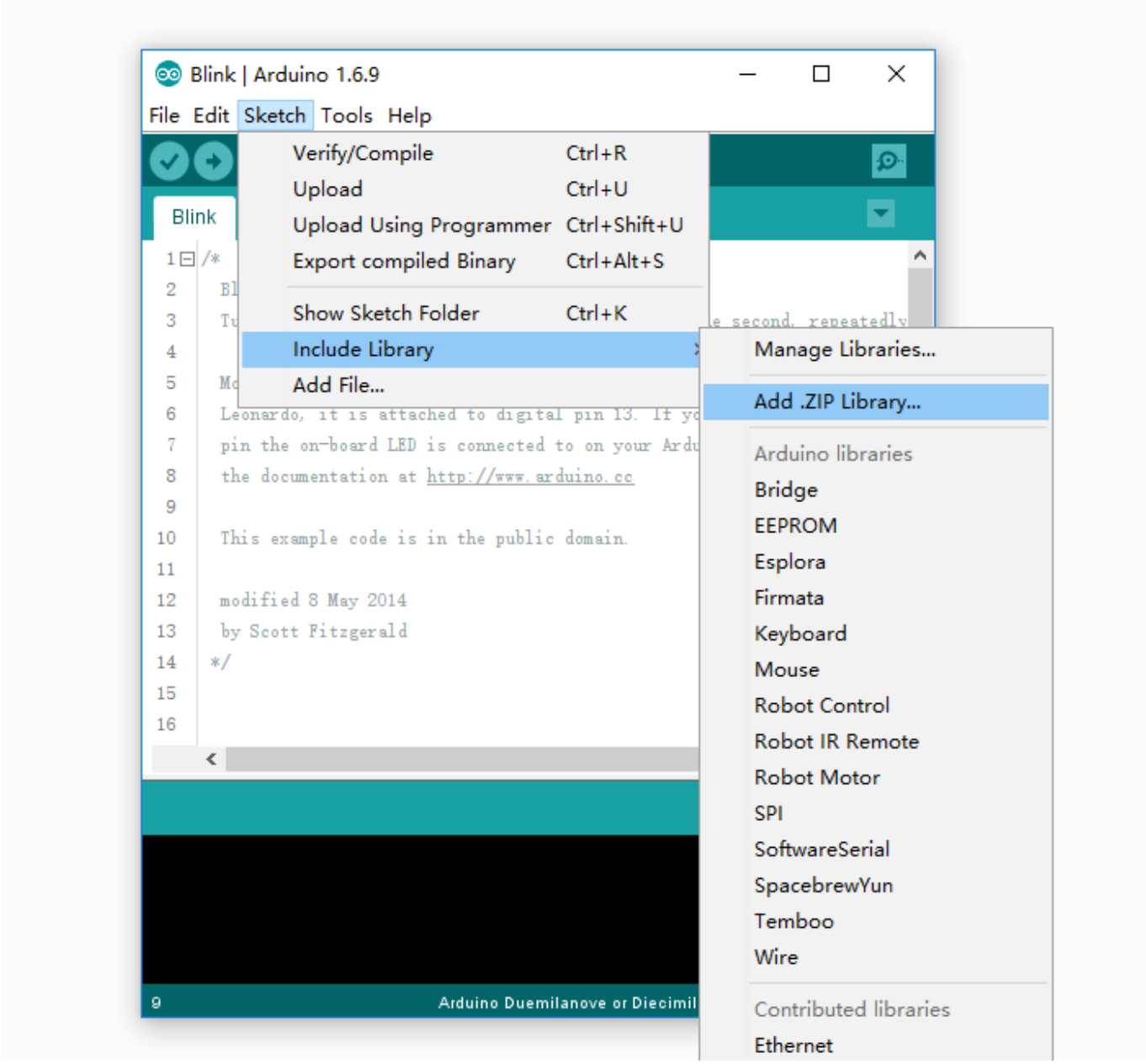
Steps below shows how to add library for Grove – Chainable RGB LED.

Step 1: Open Github repository from Library URL, and download the zip data

Please find “Clone or download | Download ZIP” from the Github page, you must choose Download ZIP only, and please remember the file path that you have downloaded and saved the zip file.

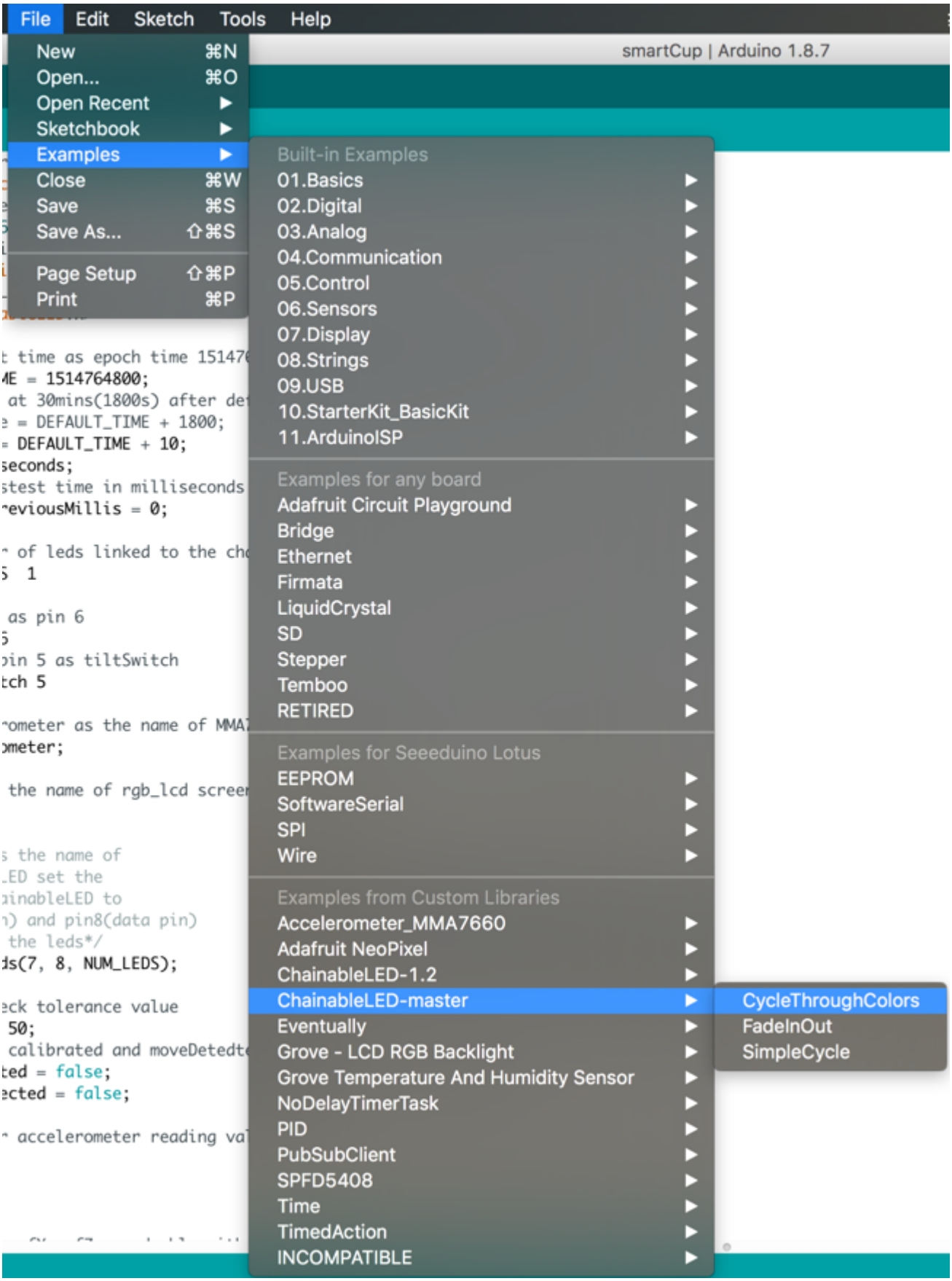
Step 2: Select the “include Library | Add .ZIP Library..

Please select open Sketch | Include Library | Add .ZIP Library..., in the new pop up window please select the zip file you have downloaded from last step, then click choose.



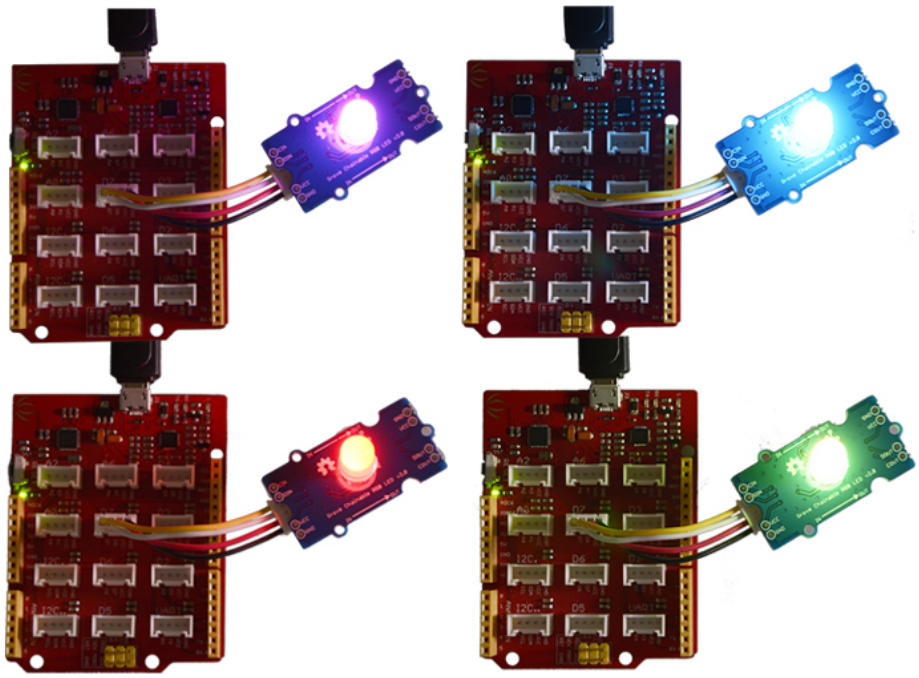
Step 3: Test if the library added successfully

Select and open File | Examples | ChainableLED-1.2 | CycleTroughColors



Upload Code: Click upload the code

!!!Success if the LED module is cycling trough different colors, then you know you have successfully load the Library.



Conclusion: Adding library method allows user easily add third party sensor drivers and libraries made by sensor module venders, for example, in this tutorial session, we used setColorRGB(byte led, byte red, byte green, byte blue) function is one of the function implement by Seeed Studio, this reduces user's development cost when they adapt new sensor module. You will need to add more libraries for the other grove modules later.

Example 1:Use setColorHSB function to change the LED color

Step 1: copy & paste the following code into Arduino IDE

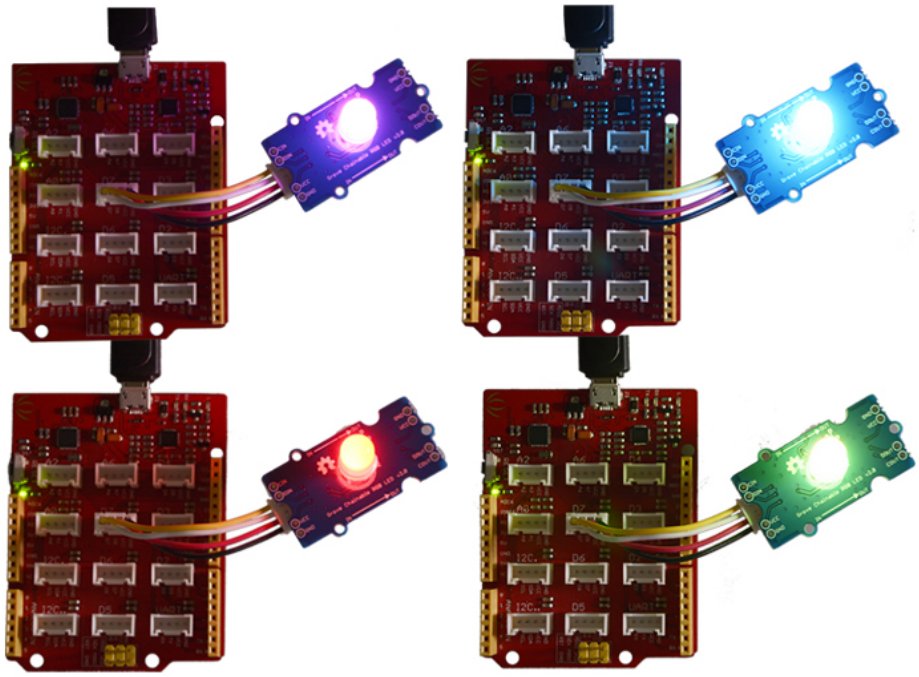
```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

void setup()
{
  //initialise ChainableLED leds
```

```
  leds.init();
}
//initialise hue as float point with value of 0.0
float hue = 0.0;
//initialise up as boolean with value of true
boolean up = true;

void loop()
{
  /*for loop is used for loop through
    each LED connected to the chain
    in this case there is only one LED
  */
  for (byte i = 0; i < NUM_LEDS; i++) {

    /*setColorHSB(byte led, float hue, float saturation, float brightness);
       in this case set the first and only chainableLED 0 with changing hue
       and full saturation and half brightness
    */
    leds.setColorHSB(i, hue, 1, 0.5);
    //    delay for 50ms for each color
    delay(50);

    /*if up is true increase
      hue at 0.025 interval
      otherwise decrease hue
      at 0.025 interval
    */
    if (up) {
      hue += 0.025;
    }
    else
    {
      hue -= 0.025;
    }
    /*
      if hue is greater than 1.0
       and up is true set up to false,
       otherwise if hue is less or
       equal to 0.0 and up is not
       ture(! means is not) set up
       to true
    */
    if (hue >= 1.0 && up)
    {
      up = false;
    }
    else if (hue <= 0.0 && !up)
    {
      up = true;
    }
  }
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

You should see the color of LED is changing according to the value of hue, which is increasing by 0.025 increment and when the hue value reaches 1, the hue value should reduce by 0.025 decrement until the value become 0, and each color should lit for 50 millisecond.



Example 2:Use setColorRGB function to change the color and brightness of LED

Step 1: copy & paste the following code into Arduino IDE

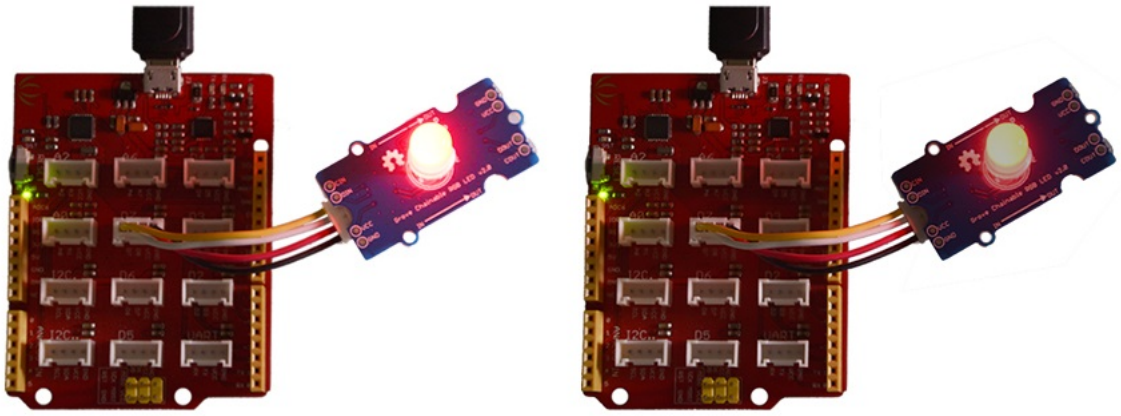
```
/*
   Example of using the ChainableRGB library for controlling a Grove RGB.
   This code fades in an out colors in a strip of leds.
*/

//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED set the
```

```
    pin of the ChainableLED to
    pin7(clock pin) and pin8(data pin)
    and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

void setup()
{
  //initialise ChainableLED leds
  leds.init();
}
//initialise power as byte with value of 0
byte power = 0;

void loop()
{
  /*for loop is used for loop through
    each LED connected to the chain
    in this case there is only one LED
  */
  for (byte i = 0; i < NUM_LEDS; i++)
  {
    /*
      % means modulo operation to find remainder
      eg 0 % 2 = 0, 1 % 2 = 1,  2 % 2 = 0...
      setColorRGB(byte led, byte red, byte green, byte blue);
      so in this case the even number of the LED chain
      will fading green color, odd number of the LED
      chain will fading red color, since we count the
      first LED as 0.
    */
    if (i % 2 == 0)
      //brighter red color from 0 to full power
      leds.setColorRGB(i, power, 0, 0);
    else
      //dimmer green color from full power to 0
      leds.setColorRGB(i, 0, 255 - power, 0);
  }
  //set power increment as 10
  power += 10;
  //light 0.5s for each brightness
  delay(500);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

you should see the red color LED increases the brightness in every 0.5s, because we only set the value for red variable in the setColorRGB(byte led, byte red, byte green ,byte blue) function.



If we change both red and green variables from this:

```
leds.setColorRGB(i, power, 0, 0);
```

to this:

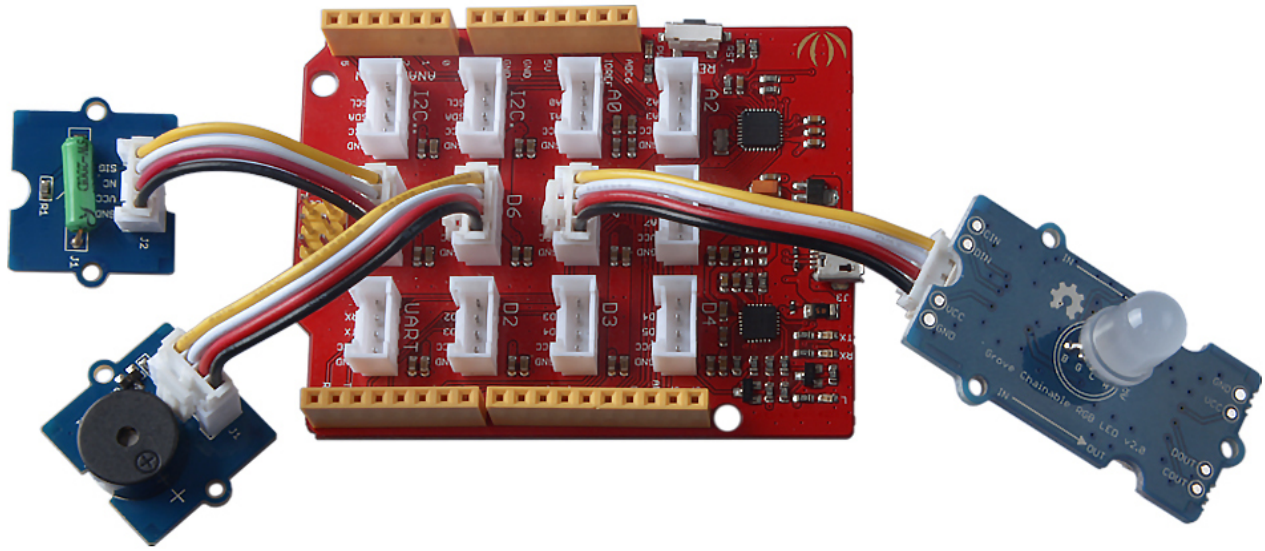
```
leds.setColorRGB(i, power, 255-power, 0);
```

please observe the difference.

Example 3:Use tilt switch to control the LED and Buzzer

Connect Grove – Tilt Switch to D5 port of Seeeduino Lotus.

Connect Grove – Buzzer module to D6 port of Seeeduino Lotus.



Step 1: copy & paste the following code into Arduino IDE

```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

// initalise the frequency of the notes
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_C4  262
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_G4  392
#define NOTE_C5  523

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_F4, NOTE_E4,
  NOTE_C4, NOTE_C4, NOTE_D4, NOTE_C4, NOTE_G4, NOTE_F4,
  NOTE_C4, NOTE_C4, NOTE_C5, NOTE_A4, NOTE_F4, NOTE_E4, NOTE_D4,
  NOTE_AS4, NOTE_AS4, NOTE_A4, NOTE_F4, NOTE_G4, NOTE_F4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
```

```
int noteDurations[] = {
  8, 8, 4, 4, 4, 2,
  8, 8, 4, 4, 4, 2,
  8, 8, 4, 4, 4, 4, 4,
  8, 8, 4, 4, 4, 2,
};

//set title of pin 5 as tiltSwitch
#define tiltSwitch 5
//set title of pin 6 as buzzer
#define buzzer 6

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

// set variable currentNote to store latest note played
int currentNote;
//initialise hue as float point with value of 0.0
float hue = 0.0;
//initialise up as boolean with value of true
boolean up = true;
//initialise power as byte with value of 0
byte power = 0;
//initialise color as integer with value of 0
int color = 0;

void setup()
{
  //set pin 5(tilt switch) as input pin
  pinMode(tiltSwitch, INPUT);
  //initialise ChainableLED leds
  leds.init();
}

void loop()
{
  /*read the status of tilt switch
    if the logic level of tilt switch
    is high, start play music */
  if (HIGH == digitalRead(tiltSwitch)) {

    for (int thisNote = currentNote ; thisNote < 25 ; thisNote++) {
      // to calculate the note duration, take one second divided by the note type.
      //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
      int noteDuration = 1000 / noteDurations[thisNote];
      tone(buzzer, melody[thisNote], noteDuration);

      // to distinguish the notes, set a minimum time between them.
      int pauseBetweenNotes = noteDuration * 1.30;
      delay(pauseBetweenNotes);
```

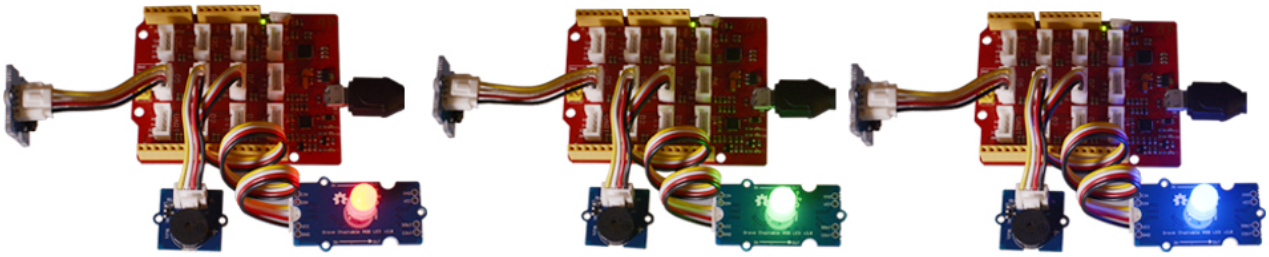
```
    /*reset the currentNote to the 0
      is the music is finished*/
    if (thisNote >= 24) {
      currentNote = 0;
    }

    /*set the LED to loop through
      different colors with different hue*/
    leds.setColorHSB(0, hue, 1, 0.5);

    /*if up is true increase
      hue at 0.025 interval
      otherwise decrease hue
      at 0.025 interval
    */
    if (up) {
      hue += 0.025;
    }
    else
    {
      hue -= 0.025;
    }
    /*if hue is greater than 1.0
       and up is true set up to false,
       otherwise if hue is less or
       equal to 0.0 and up is not
       ture(! means is not) set up
       to true
    */
    if (hue >= 1.0 && up)
    {
      up = false;
    }
    else if (hue <= 0.0 && !up)
    {
      up = true;
    }

    /*druing the music read the status
      of tilt switch if the logic level
      of tilt switch is LOW, stop play
      music and store the previous played
      tone and jump to next tone*/
    if (LOW == digitalRead(tiltSwitch)) {
      /* use switch...case to set the LED loop through three colors
         Red when color = 0 enters case 0
         Green when color = 1 enters case 1
         Blue when color = 2 enters case 2
         reset color to 0 if color is greater or equals 3
      */
      if (color >= 3) {
        color = 0;
      }
```

```
          switch (color) {
            case 0:
              //set LED to Red
              leds.setColorRGB(0, 255, 0, 0);
              break;
            case 1:
              //set LED to Green
              leds.setColorRGB(0, 0, 255, 0);
              break;
            case 2:
              //set LED to Blue
              leds.setColorRGB(0, 0, 0, 255);
              break;
          }
          //increase color by 1 increment everytime enter this condition
          color ++;

          //store the thisNote to currentNote
          currentNote = thisNote;
          //set the next note ready to play by increase currentNote by 1 increament
          currentNote ++;
          /*reset the currentNote to the beginning
            is the music is finished*/
          if (currentNote >= 25)
          {
            //restart the music from beginning by reset the currentNote to 0,
            currentNote = 0;
          }
          //if the tilt switch is set to logic level low, stop playing music
          break;
        }
      }
    }
  }
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

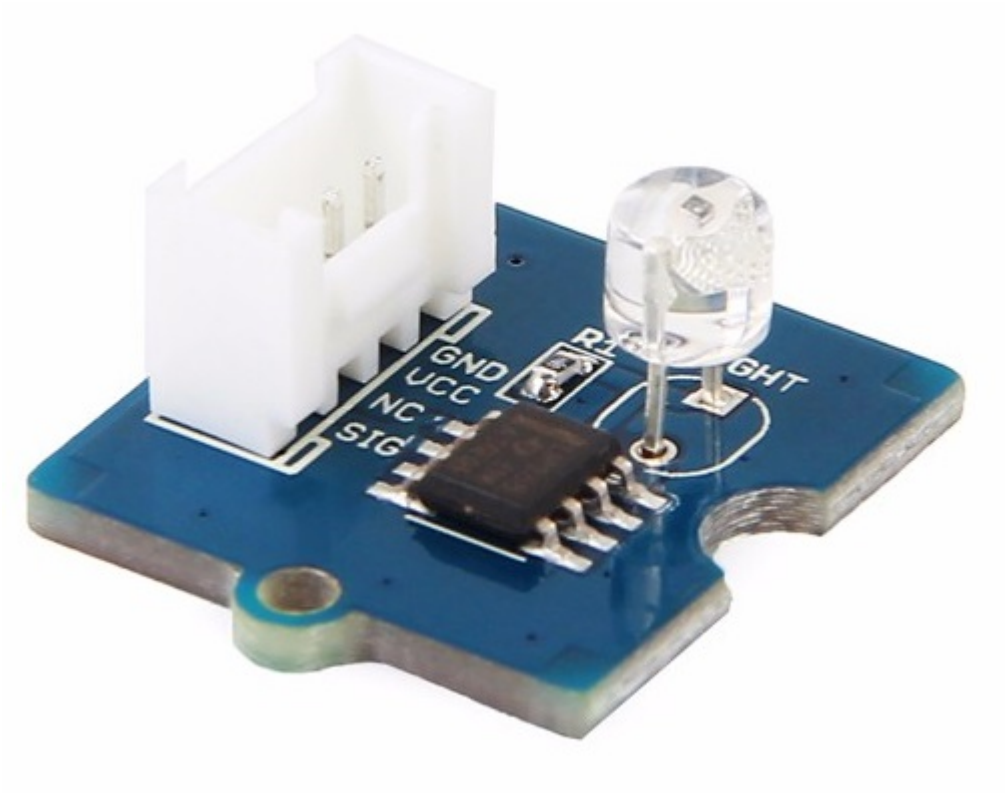
By tilting the tilt switch, you should see when the tilt switch is on, LED changes color along with the tone of buzzer changes, when the tilt switch is off, LED will cycle through Red, Green and Blue and the buzzer stops.



Further Explore

After this session, you can now combine the modules from the first three sessions and turn into a birthday gift box, where you can place the tilt switch sensor on the box lid, at the time the box is open, the tilt switch is triggered, then the buzzer starts play birthday song and the LED light starts to flash colorful lights.

Session 4: Grove - Light Sensor



Objective

Use light sensor module to turn on/off the chainable RGB LED module, and control the brightness of the LED according to the ambient light intensity.

Key knowledge

- Light Sensor module is an Analog signal input module
- use map(value, fromLow, fromHigh, toLow, toHigh)function to re-map number of Analog output from one range to another as more clear and practical.
- use light sensor as a light switch
- use light sensor to control the LED brightness by sense the ambient brightness

Hardware requirement

Self-prepare

- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

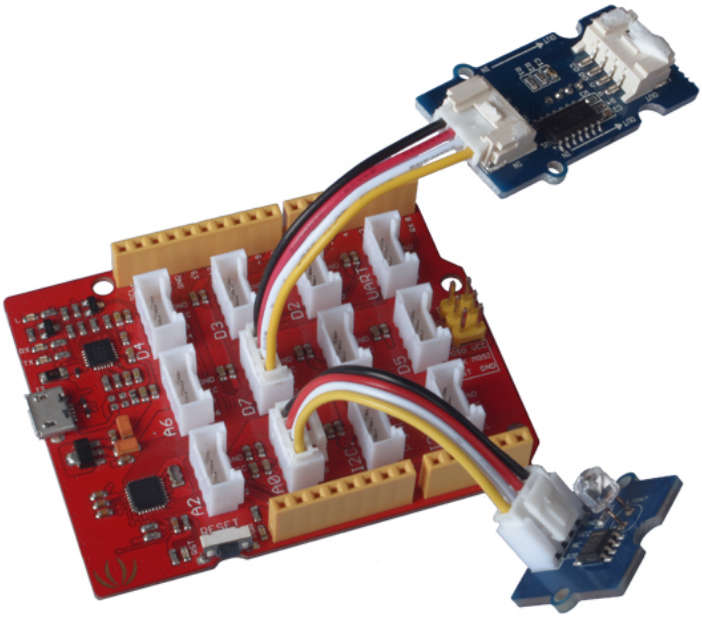
- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – Light Sensor
- Grove – Chainable RGB LED

Hardware connection

Step 1: Connect Grove – Light Sensor module to A0 port of Seeeduino Lotus

Step 2: Connect Grove – Chainable RGB LED to D7 port of Seeeduino Lotus

Step 3: Link Seeeduino Lotus with computer by a micro USB cable



Software programming

Example 1: Uses Light sensor to switch on/off LED by the brightness of the ambient light

Step 1: copy & paste the following code into Arduino IDE

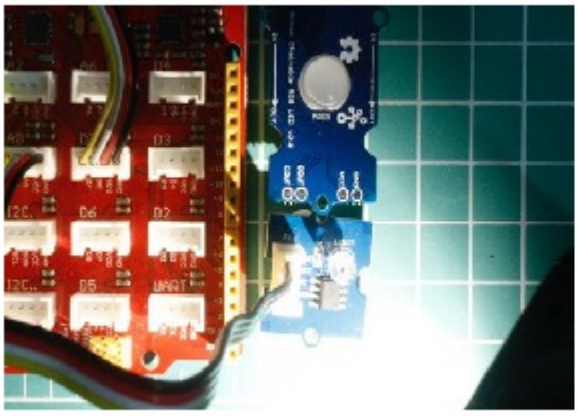
```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED, set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

//naming analog pin A0 as LightSensor
#define LightSensor A0

void setup()
```

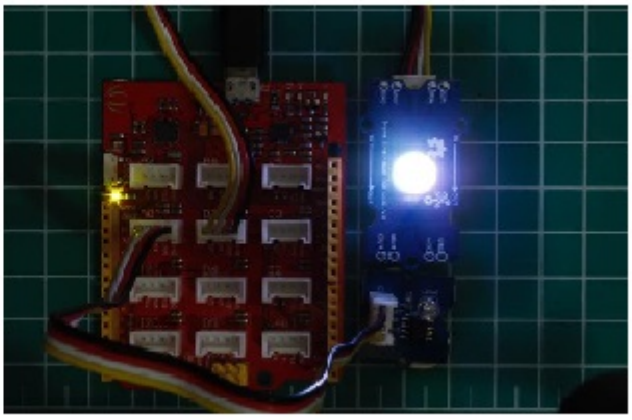
```
{
  //initialise ChainableLED leds
  leds.init();
}

void loop()
{
  //read the value of Light Sensor and store to value
  int value = analogRead(LightSensor);
  //if Sensor reading is less than 150 turn on LED
  if (value < 150) {
    //turn on LED
    leds.setColorRGB(0, 10, 10, 10);
    //delay for 1s
    delay(1000);
  } else
  {
    //turn off LED
    leds.setColorRGB(0, 0, 0, 0);
    //delay for 1s
    delay(1000);
  }
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

Note if the ambient light is to bright, you can use you hand to cover the light sensor module, then the LED should turn on. when the ambient light makes the reading value of light sensor go higher than 150, the LED should turn off.





Example 2: Uses light sensor to control the brightness of the LED

Step 1: copy & paste the following code into Arduino IDE

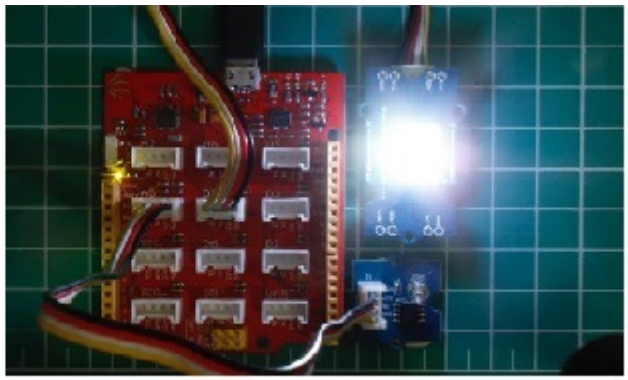
```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

#define LightSensor A0

void setup()
{
  //initialise ChainableLED leds
  leds.init();
  Serial.begin(9400);
}

//initialise hue as float point with value of 0.0
float hue = 0.0;
//initialise up as boolean with value of true
boolean up = true;

void loop()
{
  //read the value of Light Sensor and store to value
  int value = analogRead(LightSensor);
  /*map(value, fromLow, fromHigh, toLow, toHigh)
    Re-maps a number from one range to another
    In this case maping the analog value of light sensor
    ranging from 0-800 to 100-0, so when the brightness
    of surrounding enviroment is high so the sensor reading
    value is high, therefore the maped value should be opposite,
    so the birghtness LED should be dimmer.
    the brightness of chainable LED only accept float number, so
    we divide the maped value with 100.
  */
  float value_float = map(value, 0, 800, 50, 0) / 100.0;
  /*setColorHSB(byte led, float hue, float saturation, float brightness);
   * use the maped value(value_float) as brightness
  */
  leds.setColorHSB(0, hue, 1, value_float);
  delay(100);

  /*if up is true increase
    hue at 0.025 interval
    otherwise decrease hue
    at 0.025 interval
```

```
    */
    if (up) {
      hue += 0.025;
    }
    else
    {
      hue -= 0.025;
    }
    /*if hue is greater than 1.0
       and up is true set up to false,
       otherwise if hue is less or
       equal to 0.0 and up is not
       ture(! means is not) set up
       to true
    */
    if (hue >= 1.0 && up)
    {
      up = false;
    }
    else if (hue <= 0.0 && !up)
    {
      up = true;
    }
  }
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

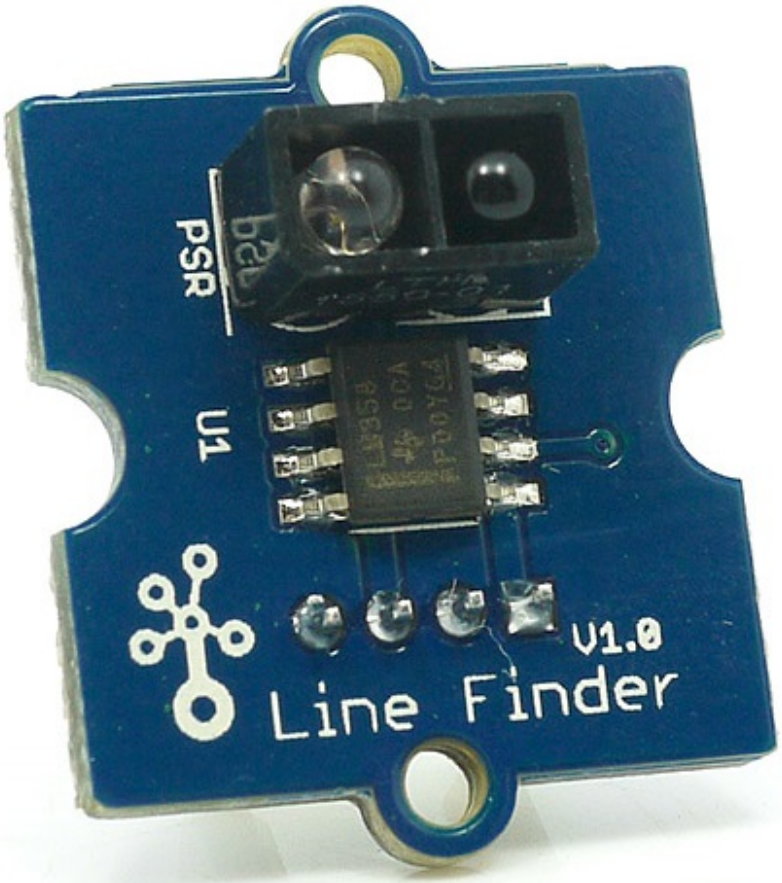
The brightness of the LED should decreases when the ambient brightness increases. When the ambient brightness decreases, the brightness of the LED should be increasing. as shown, the LED dims when there is a bright light shine on the light sensor, otherwise the LED is bright.



Further Explore

Now you can integrate this light sensor module into your corridor lighting system to control the brightness of the light, during the day time the light sensor detects the sun light then dim the brightness of the corridor light to low, which not only saves power, but also extend the lifespan of the globes.

Session 5: Grove - Line Finder



Objective

Uses line finder to detect black line, and control the color of the LED accordingly when the line finder detects black line or not.

Key knowledge

- Grove – Line Finder is a digital signal input module
- Revise on how to use Serial Monitor
- Uses signal input module to control Grove – Chainable RGB LED

Hardware requirement

Self-prepare

- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

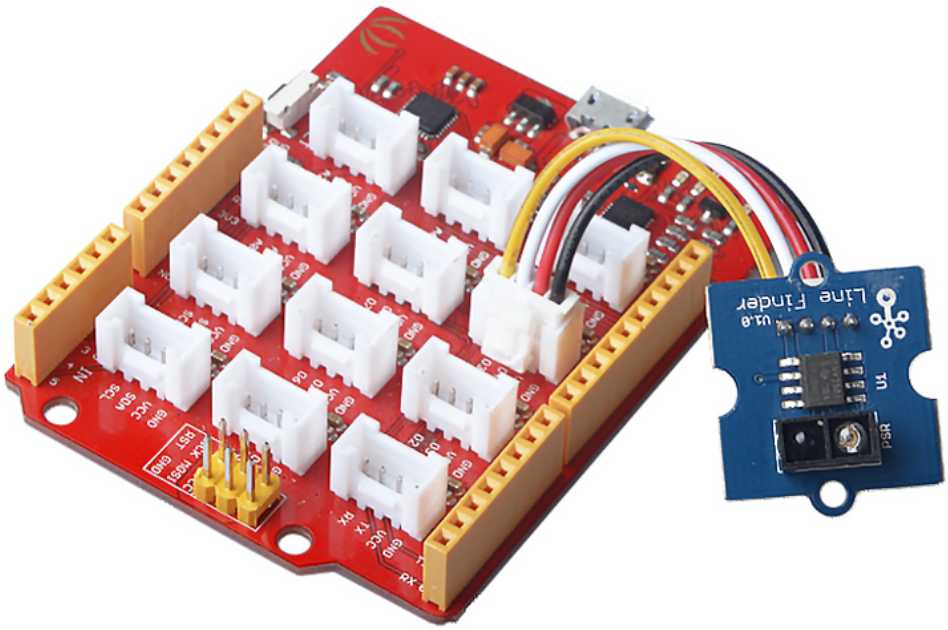
Included in the kit

- Grove – Line Finder
- Grove – Chainable RGB LED

Hardware connection

Step 1: Connect Grove – Line Finder module to D3 port of Seeeduino Lotus

Step 2: Link Seeeduino Lotus with computer by a micro USB cable



Software programming

Example 1:Uses Serial Monitor to display and test output signal from the line finder

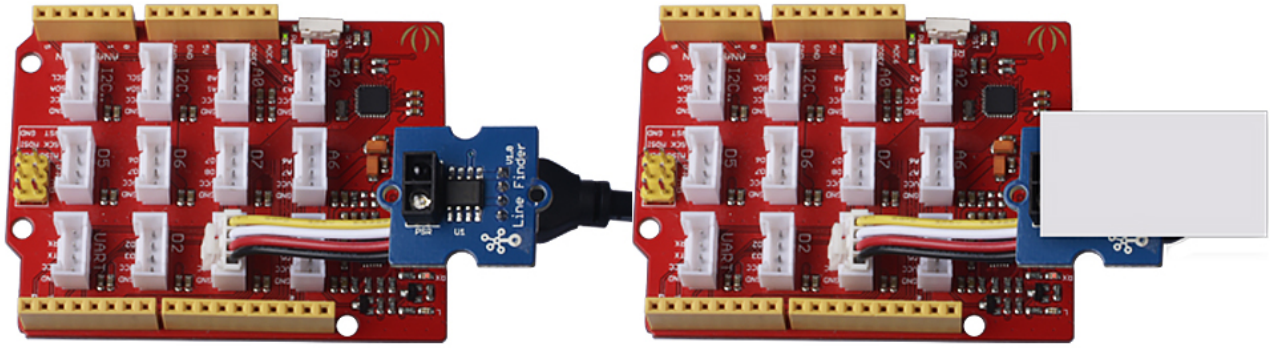
Step 1: copy & paste the following code into Arduino IDE

```
//naming pin3 as singalPin
#define signalPin 3

void setup() {
  // initialize the digital pin as an input:
  pinMode(signalPin, INPUT);
  // opens serial port, sets data rate to 9600 bps
  Serial.begin(9600);
}

void loop() {
  //read the line detector input
  int val = digitalRead(signalPin);

  //display the line detector status, 1 is black, 0 is white.
  Serial.println(val);
}
```

Step 2: Upload code into Seeeduino Lotus

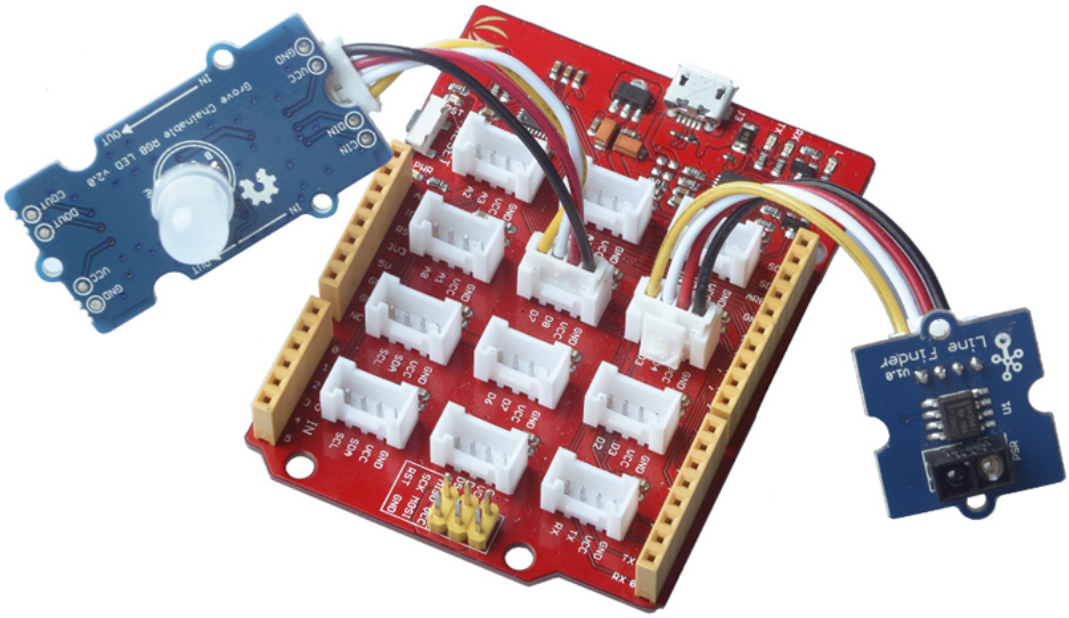
Step 3: Observe result

Note you should keep at least 5cm away for the detecting object with the line finder, to test the line finder you should tape a strip of black tape onto a white paper or tiles (or uses black objects). Now aim the line finder to the black object, the Serial Monitor should display 0, and if you move away the line finder from black object, the Serial Monitor should display 1.



Example 2:Uses Line detector to turn on or off Grove – Chainable RGB LED module.

Connect Grove – Chainable RGB LED to D7 port of Seeeduino Lotus



Step 1: copy & paste the following code into Arduino IDE

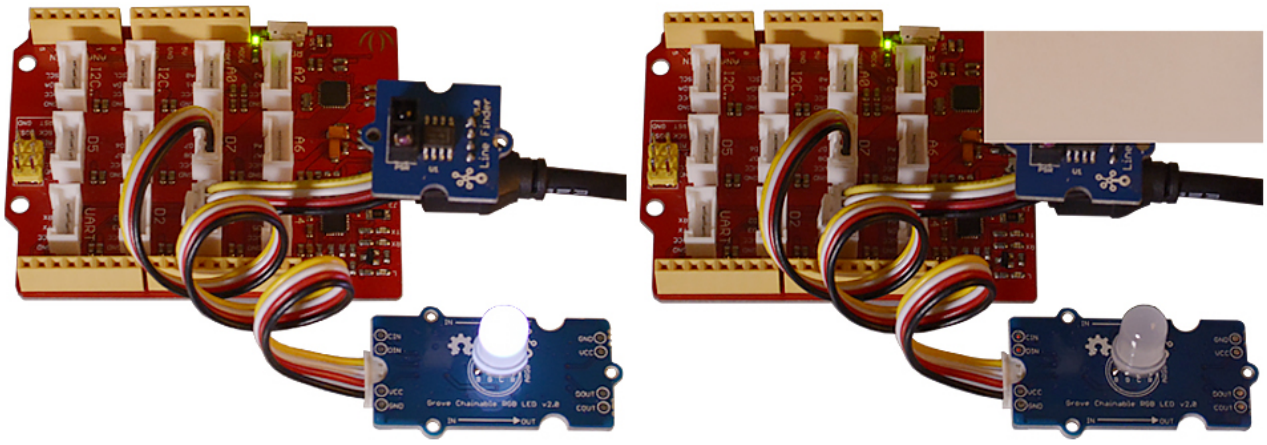
```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

//naming pin3 as lineFinder
#define lineFinder 3

void setup() {
  // initialize the digital pin as an input:
  pinMode(lineFinder, INPUT);
  //initialise ChainableLED leds
  leds.init();
}

void loop() {
  /*read the line detector input
```

```
     * if detected black(HIGH) turn on LED
    */
    if (HIGH == digitalRead(lineFinder))
    {
      //turn off LED
      leds.setColorRGB(0, 10, 10, 10);
    }

    /*read the line detector input
     * if reading Logic low turn off LED
    */
    if (LOW == digitalRead(lineFinder))
    {
      //turn off LED
      leds.setColorRGB(0, 0, 0, 0);
    }
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

You should see that if the line finder detects black line, the LED should turn off, otherwise the LED should lid up if the line finder cannot detects the black line.



Example 3:Uses line finder to control the LED to emit Red or Green

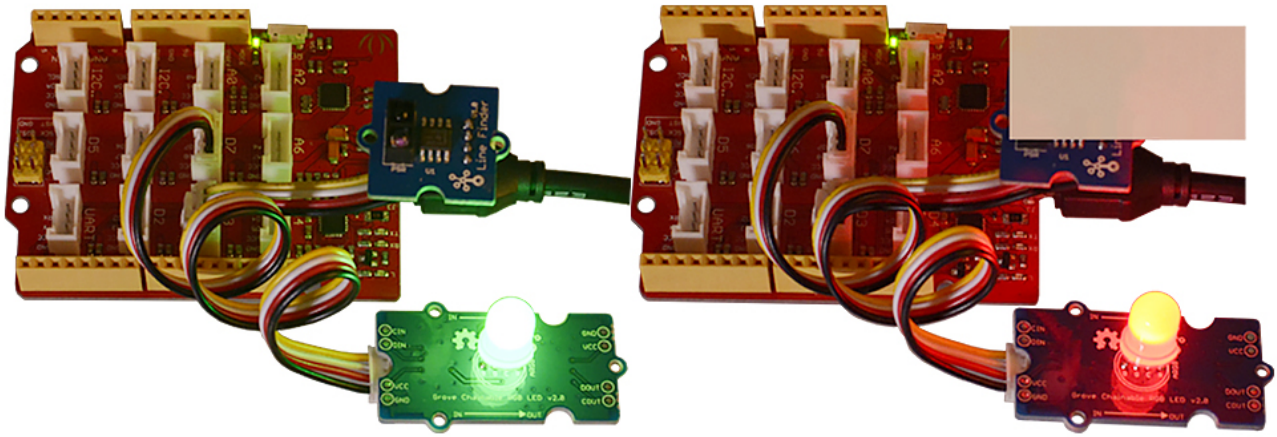
Step 1: copy & paste the following code into Arduino IDE

```
//add ChainableLED library to this project
#include <ChainableLED.h>

//set the number of leds linked to the chain
#define NUM_LEDS  1

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

//naming pin3 as lineFinder
#define lineFinder 3

void setup() {
  // initialize the digital pin as an input:
  pinMode(lineFinder, INPUT);
  //initialise ChainableLED leds
  leds.init();
}

void loop() {
  /*read the line detector input
   * if detected black(HIGH) set Green LED
  */
  if (HIGH == digitalRead(lineFinder))
  {
    //Green LED
    leds.setColorRGB(0, 0, 255, 0);
  }

  /*read the line detector input
   * if reading Logic low set Red LED
  */
  if (LOW == digitalRead(lineFinder))
  {
    //Red LED
    leds.setColorRGB(0, 255, 0, 0);;
  }
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

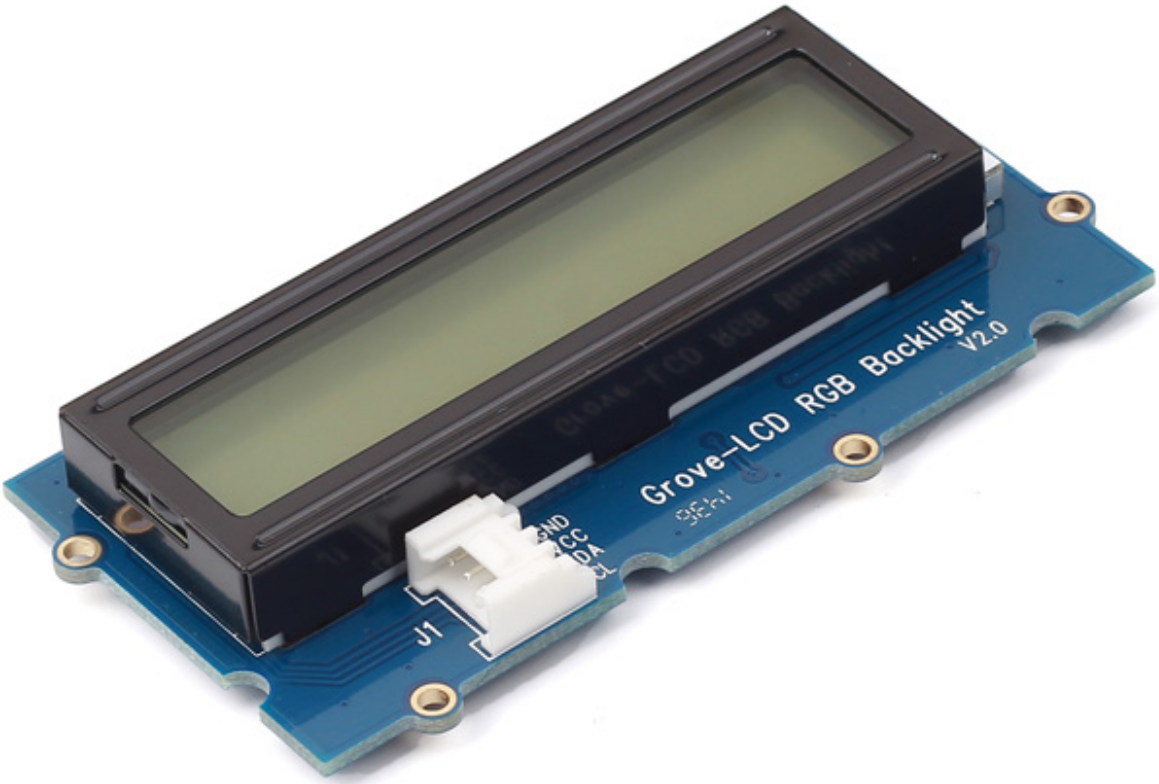
You should notice when the line finder detects black line, the LED will emit Red light, otherwise, if the line finder cannot spot the black line, the LED should emit Green light.



Further Explore

Now you can build your own line tracking car by using this line finder module and two motors with a motor driver(H-bridge), so when the line finder detects black line, active one side of the wheel motor, once the line finder is off the black line, stop the spinning side of the wheel motor, and active the other side of wheel motor, so car will drive along the black line with the front of the car constant turning left or right.

Session 6: Grove - LCD RGB Backlight



Objective

Uses Grove - LCD RGB Backlight screen to display “Hello World” and some custom characters.

Key knowledge

- Revise on how to add library
- Master in the positioning character and use binary code to generate custom character.
- Scrolling text displayed on the LCD screen
- Uses LCD built-in character code to display special characters, for example the degree sign “ ° ”

Hardware requirement

Self-prepare

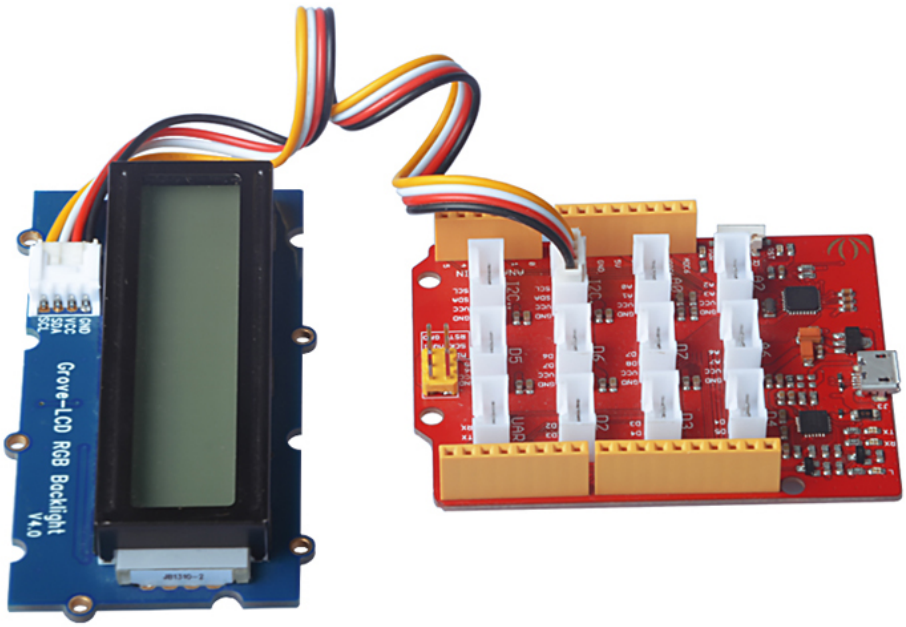
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove - LCD RGB Backlight

Hardware connection

Step 1: Connect Grove - LCD RGB Backlight module to I2C. port of Seeeduino Lotus note: it is the I2C port followed by one dot.



Step 2: Link Seeeduino Lotus with computer by a micro USB cable.

Software programming

Add Library

Add [Library](https://github.com/Seeed- Studio/Grove_LCD_RGB_Backlight/archive/master.zip) for Grove - LCD RGB Backlight Screen

Please follow the instructions in tutorial 3 on how to add library.

Example 1:Display Hello World

Step 1: copy & paste the following code into Arduino IDE

```
//include the rgb_lcd library
#include "rgb_lcd.h"
```

```
//assign name lcd to rgb_lcd
rgb_lcd lcd;

void setup()
{
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);

    // Print Hello, World! to the LCD.
    lcd.print("Hello, World!");
    delay(1000);
}

void loop()
{
    // set the cursor to column 0, line 1
    // (note: line 1 is the second row, since counting begins with 0):
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis()/1000);
    delay(100);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

You should see “Hello, World!” displayed in the first line and the a counting down timer in the second line.

Example 2: Display Seeed Studio logo and uses built-in character code to display text

Step 1: copy & paste the following code into Arduino IDE

```
//add LCD library
#include "rgb_lcd.h"

//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

/*draw customise character
  in this case is the seeed studio logo
*/
byte top_leftleft[8] = {
  B00000,
  B00000,
  B01000,
  B01000,
  B01000,
  B01100,
  B01100,
  B01110
};
byte top_midleft[8] = {
  B00001,
  B00010,
  B00010,
```

```
  B00110,
  B00110,
  B00100,
  B01100,
  B01100
};
byte top_midright[8] = {
  B10000,
  B01000,
  B01000,
  B01100,
  B01100,
  B00100,
  B00110,
  B00110
};
byte top_rightright[8] = {
  B00000,
  B00000,
  B00010,
  B00010,
  B00010,
  B00110,
  B00110,
  B01110
};
byte bot_leftleft[8] = {
  B00110,
  B00111,
  B00011,
  B00011,
  B00001,
  B00000,
  B00000,
  B00000
};
byte bot_midleft[8] = {
  B01100,
  B01110,
  B00110,
  B00110,
  B10011,
  B11011,
  B11111,
  B01111
};
byte bot_midright[8] = {
  B00110,
  B01110,
  B01100,
  B01100,
  B11001,
  B11011,
  B11111,
```

```
  B11110
};
byte bot_rightright[8] = {
  B01100,
  B11100,
  B11000,
  B11000,
  B10000,
  B00000,
  B00000,
  B00000
};

void setup()
{
  //initialise the lcd screen;
  // set up the lcd's number of columns and rows:
  lcd.begin(16, 2);

  /*create and assign numbers of
    each seeed studio logo elements
  */
  lcd.createChar(0, top_leftleft);
  lcd.createChar(1, top_midleft);
  lcd.createChar(2, top_midright);
  lcd.createChar(3, top_rightright);
  lcd.createChar(4, bot_leftleft);
  lcd.createChar(5, bot_midleft);
  lcd.createChar(6, bot_midright);
  lcd.createChar(7, bot_rightright);


  /* set the cursor to column 4, line 0
    note: line 0 is the first row, since counting begins with 0
    same rule apply to the column as well
  */
  lcd.setCursor(4, 0);
  //Print I and a space to the LCD at column 4, line 0
  lcd.print("I ");
  //set the cursor to column 6, line 0
  lcd.setCursor(6, 0);
  /* Print LOVE by using the LCD character lookup table
     note write() method is mentt to send raw bytes
     print() is mostly intended to format data as ascii.
     this is different way of display text on lcd.
  */
  //character 76 is L on lookup table
  lcd.write(76);
  //the hex number 0x4F(is 79) associates with O on lookup table
  lcd.write(0x4F);
  //character 86 is V on lookup table
  lcd.write(86);
  //character 69 is E on lookup table
  lcd.write(69);
```

```
    //set the cursor to column 10, line 0
    lcd.setCursor(10, 0);
    //Print a space and Grove to the LCD
    lcd.write(" Grove");
    //set the cursor to column 4, line 1
    lcd.setCursor(4, 1);
    //Print text Seeed Studio to the LCD
    lcd.print("Seeed Studio");

    //display seeed studio logo
    lcd.setCursor(0, 0);
    lcd.write((unsigned char)0);
    lcd.setCursor(1, 0);
    lcd.write(1);
    lcd.setCursor(2, 0);
    lcd.write(2);
    lcd.setCursor(3, 0);
    lcd.write(3);
    lcd.setCursor(0, 1);
    lcd.write(4);
    lcd.setCursor(1, 1);
    lcd.write(5);
    lcd.setCursor(2, 1);
    lcd.write(6);
    lcd.setCursor(3, 1);
    lcd.write(7);
}

void loop()
{

}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

You will see the Seeed Studio Logo displayed in the first 8 blocks, followed by the “I Love Grove” in the first line, and “Seeed Studio” in the second line.

Example 3:Scrolling text on LCD display

Step 1: copy & paste the following code into Arduino IDE

```
//add LCD library
#include "rgb_lcd.h"

//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

/*draw customise character
  in this case is the seeed studio logo
*/
byte top_leftleft[8] = {
  B00000,
  B00000,
  B01000,
  B01000,
  B01000,
  B01100,
  B01100,
  B01110
};
byte top_midleft[8] = {
  B00001,
  B00010,
```

```
  B00010,
  B00110,
  B00110,
  B00100,
  B01100,
  B01100
};
byte top_midright[8] = {
  B10000,
  B01000,
  B01000,
  B01100,
  B01100,
  B00100,
  B00110,
  B00110
};
byte top_rightright[8] = {
  B00000,
  B00000,
  B00010,
  B00010,
  B00010,
  B00110,
  B00110,
  B01110
};
byte bot_leftleft[8] = {
  B00110,
  B00111,
  B00011,
  B00011,
  B00001,
  B00000,
  B00000,
  B00000
};
byte bot_midleft[8] = {
  B01100,
  B01110,
  B00110,
  B00110,
  B10011,
  B11011,
  B11111,
  B01111
};
byte bot_midright[8] = {
  B00110,
  B01110,
  B01100,
  B01100,
  B11001,
  B11011,
```

```
  B11111,
  B11110
};
byte bot_rightright[8] = {
  B01100,
  B11100,
  B11000,
  B11000,
  B10000,
  B00000,
  B00000,
  B00000
};

void setup()
{
  //initialise the lcd screen;
  //set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  //wait for 1s
  delay(1000);
}

void loop()
{

  /*create and assign numbers of
    each seeed studio logo elements
  */
  lcd.createChar(0, top_leftleft);
  lcd.createChar(1, top_midleft);
  lcd.createChar(2, top_midright);
  lcd.createChar(3, top_rightright);
  lcd.createChar(4, bot_leftleft);
  lcd.createChar(5, bot_midleft);
  lcd.createChar(6, bot_midright);
  lcd.createChar(7, bot_rightright);

  /* set the cursor to column 4, line 0
    note: line 0 is the first row, since counting begins with 0
    same rule apply to the column as well
  */
  lcd.setCursor(4, 0);
  //Print I and a space to the LCD at column 4, line 0
  lcd.print("I ");
  //set the cursor to column 6, line 0
  lcd.setCursor(6, 0);
  /* Print LOVE by using the LCD character lookup table
     note write() method is mentt to send raw bytes
     print() is mostly intended to format data as ascii.
     this is different way of display text on lcd.
  */
  //character 76 is L on lookup table
  lcd.write(76);
```

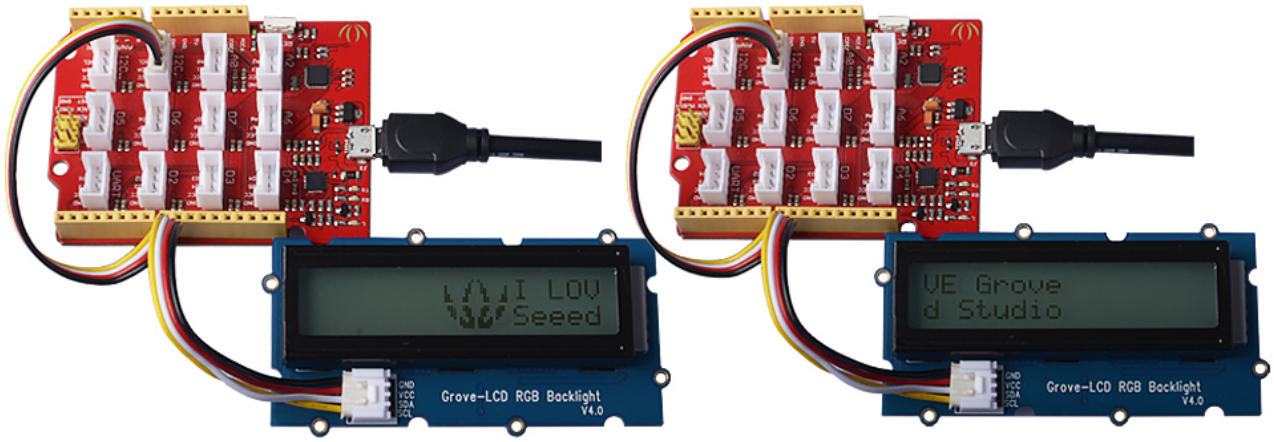
```
    //the hex number 0x4F(is 79) associates with O on lookup table
    lcd.write(0x4F);
    //character 86 is V on lookup table
    lcd.write(86);
    //character 69 is E on lookup table
    lcd.write(69);
    //set the cursor to column 10, line 0
    lcd.setCursor(10, 0);
    //Print a space and Grove to the LCD
    lcd.write(" Grove");
    //set the cursor to column 4, line 1
    lcd.setCursor(4, 1);
    //Print text Seeed Studio to the LCD
    lcd.print("Seeed Studio");

    //display seeed studio logo
    lcd.setCursor(0, 0);
    lcd.write((unsigned char)0);
    lcd.setCursor(1, 0);
    lcd.write(1);
    lcd.setCursor(2, 0);
    lcd.write(2);
    lcd.setCursor(3, 0);
    lcd.write(3);
    lcd.setCursor(0, 1);
    lcd.write(4);
    lcd.setCursor(1, 1);
    lcd.write(5);
    lcd.setCursor(2, 1);
    lcd.write(6);
    lcd.setCursor(3, 1);
    lcd.write(7);

    // scroll 16 positions (string length) to the left
    // to move it offscreen left:
    for (int positionCounter = 0; positionCounter < 16; positionCounter++) {
      // scroll one position left:
      lcd.scrollDisplayLeft();
      // wait a bit:
      delay(200);
    }

    // scroll 32 positions (string length + display length) to the right
    // to move it offscreen right:
    for (int positionCounter = 0; positionCounter < 32; positionCounter++) {
      // scroll one position right:
      lcd.scrollDisplayRight();
      // wait a bit:
      delay(200);
    }

    // scroll 16 positions (display length + string length) to the left
    // to move it back to center:
    for (int positionCounter = 0; positionCounter < 16; positionCounter++) {
```

```
      // scroll one position left:
      lcd.scrollDisplayLeft();
      // wait a bit:
      delay(200);
    }
  }
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

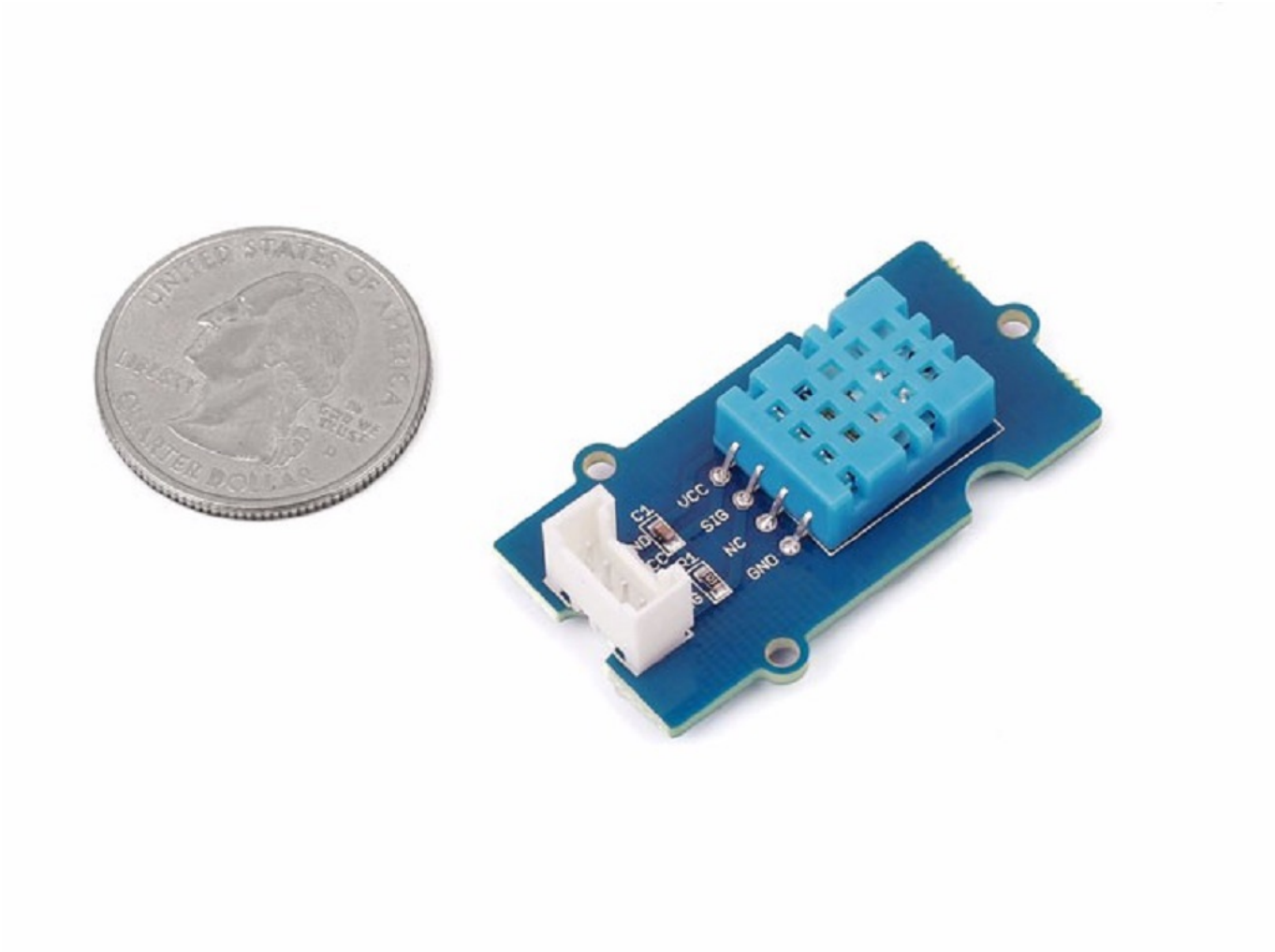
You should see the display scrolling text, firstly from right to left until all the text disappear at the end of the left side of screen, then the text will scroll back from left to right.



Further Explore

This website helps you to generate custom character for LCD screen driven by Arduino.

Session 7: Grove - Temperature & Humidity Sensor (DHT11)



Objective

Use DHT11 sensor module to detect the surrounding temperature and humidity, and display the output data from DHT11 to LCD Screen.

Key knowledge

- DHT11 is a digital sensor module
- revise on how to operate Serial Monitor and LCD Screen
- add DHT11 Library and initial setup for DHT11
- use Serial Monitor and LCD Screen to display data from DHT11 sensor

Hardware requirement

Self-prepare

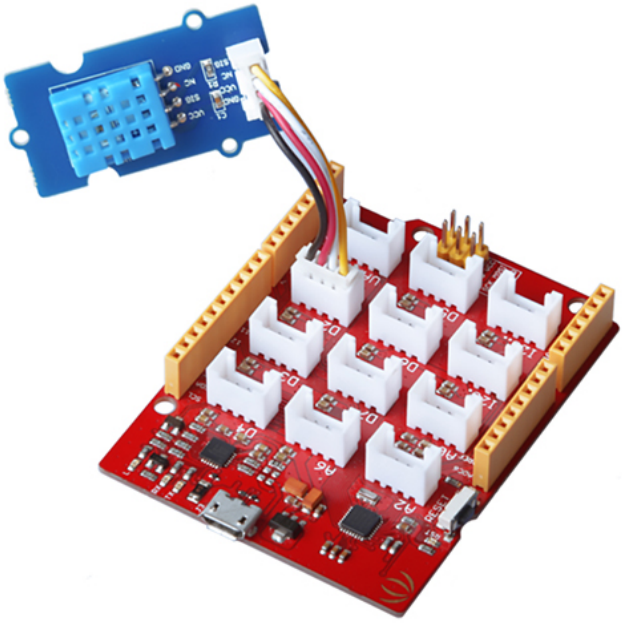
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – Temperature &Humidity Sensor(DHT11)

Hardware connection

Step 1: Connect Grove – Temperature &Humidity Sensor(DHT11) module to D2 port of Seeeduino Lotus.



Step 2: Link Seeeduino Lotus with computer by a micro USB cable

Software programming

Add [Library](https://github.com/Seeed-Studio/Grove_Temperature_And_Humidity_Sensor/archive/master.zip)

Example 1: Uses Serial Monitor to monitor the surrounding temperature and humidity

Step 1: copy & paste the following code into Arduino IDE

```
//add DHT sensor library
#include <DHT.h>

//set digital pin2 as DHTPIN
#define DHTPIN 2
//set the sensor type as DHT 11
#define DHTTYPE DHT11

/*assign dht as the name of DHT sensor
  set the sensor pin as DHTPIN(pin2),
  set the sensor type as DHTTYPE(DHT11)
*/
```

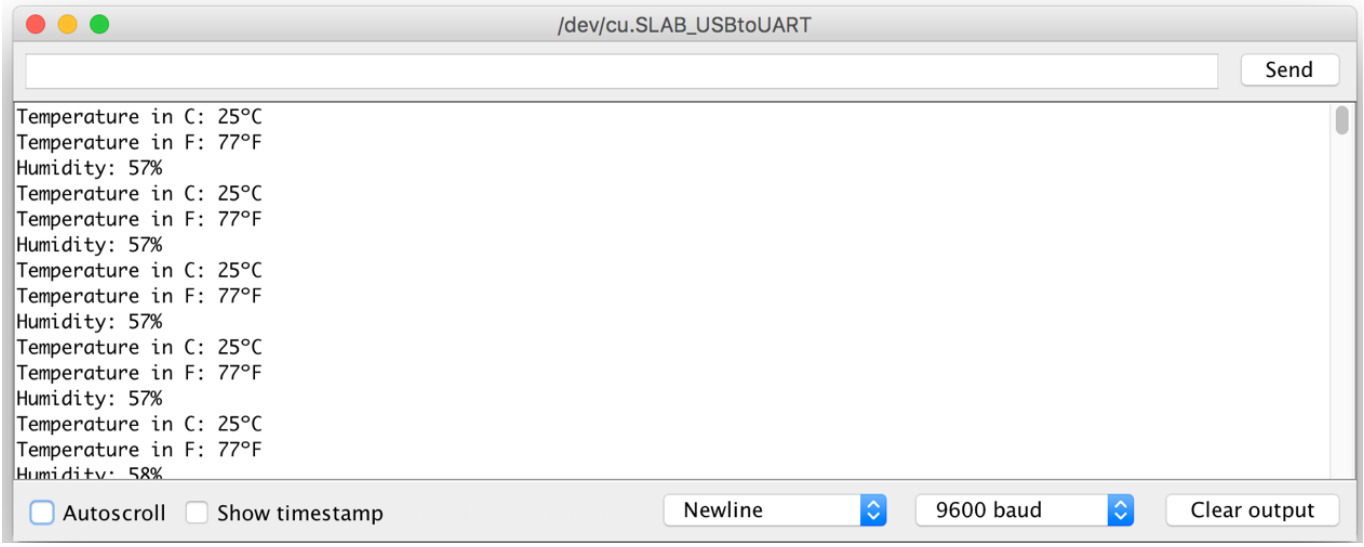
```
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  //initialise the dht sensor
  dht.begin();
  // opens serial port, sets data rate to 9600 bps
  Serial.begin(9600);
  //wait for 2s to initialise the board
  delay(2000);
}

void loop() {
  //store the humidity value to h
  int h = dht.readHumidity();
  //store the temperature value to t(in Celsius)
  int t = dht.readTemperature();
  //store the converted temperature value in fahrenheit to f
  int f = dht.convertCtoF(t);
  //display the title Temperature in C:
  Serial.print("Temperature in C: ");
  //display the temperature value t
  Serial.print(t);
  /* note the difference Serial.print()
     and Serial.println(),
     Serial.print() print the data in the same line
     Serial.println() print the data on the new line
     display the temperature unit ºC and change new line
  */
  Serial.println("ºC");
  //display the title Temperature in F:
  Serial.print("Temperature in F: ");
  //display the temperature value f
  Serial.print(f);
  //display the temperature unit ºF and change new line
  Serial.println("ºF");
  //display the title Humidity:
  Serial.print("Humidity: ");
  //display the Humidity value h
  Serial.print(h);
  //display the % sign
  Serial.println("%");
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Open Serial Monitor

Step 4: Observe result

You should see a similar text of temperature and humidity data display in serial monitor as shown below.



Example 2: Uses LCD screen to display data from DHT11 sensor

Firstly Connect Grove - LCD RGB Backlight module to I2C. port of Seeeduino Lotus note: it is the I2C port followed by one dot.

Step 1: copy & paste the following code into Arduino IDE

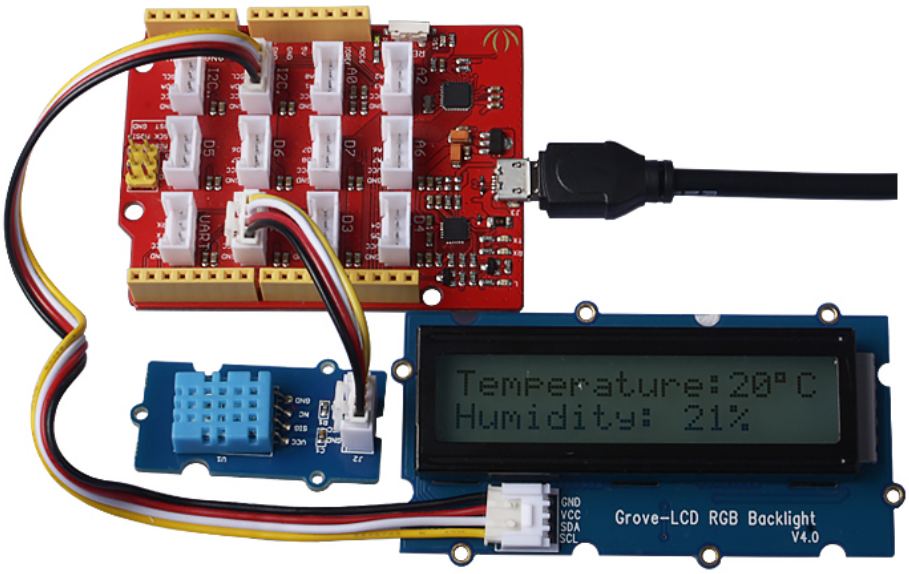
```
//add DHT sensor library
#include <DHT.h>
//add LCD library
#include <rgb_lcd.h>

//set digital pin2 as DHTPIN
#define DHTPIN 2
//set the sensor type as DHT 11
#define DHTTYPE DHT11

/*assign dht as the name of DHT sensor
  set the sensor pin as DHTPIN(pin2),
  set the sensor type as DHTTYPE(DHT11)
*/
DHT dht(DHTPIN, DHTTYPE);

//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

void setup() {
  //initialise the dht sensor
  dht.begin();
  //initialise the lcd screen;
  //set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  //wait for 2s
  delay(2000);
```

```
}

void loop() {
  //store the humidity value to h
  int h = dht.readHumidity();
  //store the temperature value to t(in Celsius)
  int t = dht.readTemperature();

  //set the LCD cursor to column 0, line 0
  lcd.setCursor(0, 0);
  //Print text temperature: to the LCD
  lcd.print("Temperature:");
  //set the LCD cursor to column 12, line 0
  lcd.setCursor(12, 0);
  //Print temperature value t to the LCD
  lcd.print(t);
  //set the LCD cursor to column 14, line 0
  lcd.setCursor(14, 0);
  //Print temperature º is character 223 on lookup table
  lcd.write(223);
  //Print C to the LCD
  lcd.print("C");

  //set the LCD cursor to column 0, line 1
  lcd.setCursor(0, 1);
  //Print text Humidity: to the LCD
  lcd.print("Humidity: ");
  //set the LCD cursor to column 10, line 1
  lcd.setCursor(10, 1);
  //Print humidity value h to the LCD
  lcd.print(h);
  //set the LCD cursor to column 12, line 1
  lcd.setCursor(12, 1);
  //Print sign % to the LCD
  lcd.print("%");
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

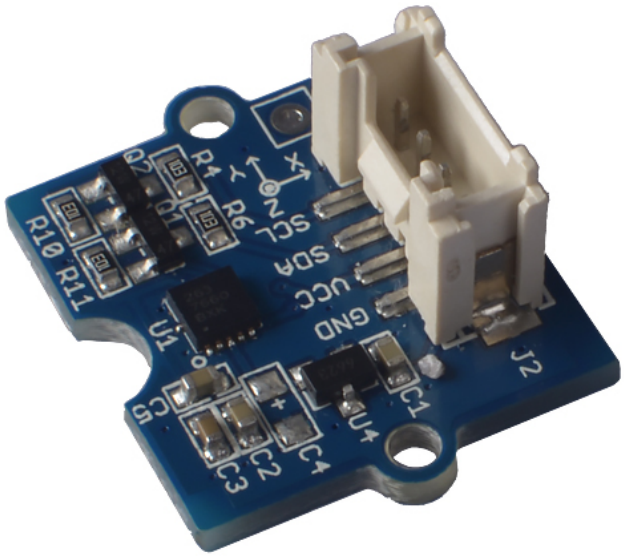
you can see the current room temperature and humidity display on the LCD screen.



Further Explore

After this session, you can build your own weather station by using DHT11 sensor and the Grove - LCD RGB Backlight display.

Session 8: Grove - 3-Axis Digital Accelerometer



Objective

Learn how to operate 3-axis digital accelerometer by observe the output data displayed on the LCD screen.

Key knowledge

- Uses serial monitor to show the offset and acceleration value from the 3-axis accelerometer, discover the relations between the output data and the position of the 3-axis accelerometer
- Uses the offset data from 3-axis accelerometer to calculate pitch and roll value, and observe the data change relative to the position of the 3-axis accelerometer.
- Learn how to use tilt switch to flip in between pages of the LCD screen, so the massive data from 3-axis accelerometer could be displayed clearly.

Hardware requirement

Self-prepare

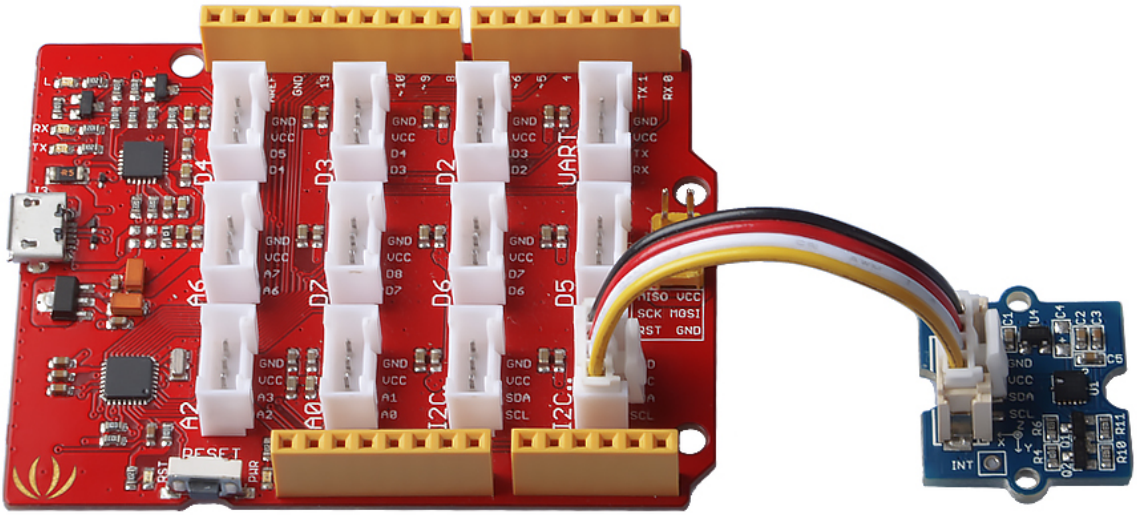
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – 3-Axis Digital Accelerometer
- Grove - LCD RGB Backlight
- Grove – Tilt Switch

Hardware connection

Step 1: Connect Grove – 3-Axis Digital Accelerometer to I2C.. port of Seeeduino Lotus note: it is the I2C port followed by two dots.



Step 2: Link Seeeduino Lotus with computer by a micro USB cable.

Software programming

Add Library

Please add the [3-axis accelerometer driver Library](https://github.com/Seeed-Studio/Accelerometer_MMA7660/archive/master.zip) into Arduino IDE

Example 1:Uses Serial Monitor to show the output data from the 3-axis accelerometer

Step 1: copy & paste the following code into Arduino IDE

```
//add accelemeter library
#include "MMA7660.h"

//assign accelemeter as the name of MMA7660 accelemeter
MMA7660 accelemeter;

void setup()
{
  //initialise the accelemeter
```

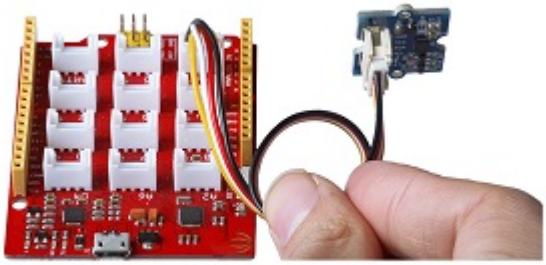
```
  accelemeter.init();
  // opens serial port, sets data rate to 9600 bps
  Serial.begin(9600);
}
void loop()
{
  //initialise x, y, z as int8_t
  int8_t x;
  int8_t y;
  int8_t z;
  //initialise ax, ay, az as float
  float ax, ay, az;
  //get x y z offset value from accelemeter
  accelemeter.getXYZ(&x, &y, &z);
  //display title x =
  Serial.print("x = ");
  //display value of x
  Serial.println(x);
  //display title y =
  Serial.print("y = ");
  //display value of y
  Serial.println(y);
  //display title z =
  Serial.print("z = ");
  //display value of z
  Serial.println(z);

  //get ax ay az acceleration value from accelemeter
  accelemeter.getAcceleration(&ax, &ay, &az);
  //display title accleration of X/Y/Z:
  Serial.println("accleration of X/Y/Z: ");
  //display value of ax
  Serial.print(ax);
  //display unit g
  Serial.println(" g");
  //display value of ay
  Serial.print(ay);
  //display unit g
  Serial.println(" g");
  //display value of az
  Serial.print(az);
  //display unit g
  Serial.println(" g");
  //display ************ as divider to make thing prettier
  Serial.println("**************");
  //wait for 0.5s
  delay(500);
}
```

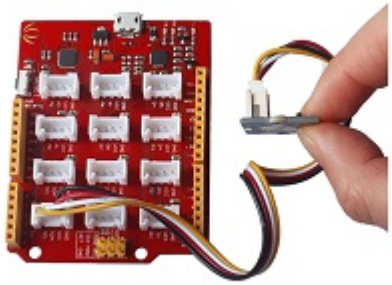
Step 2: Upload code into Seeeduino Lotus

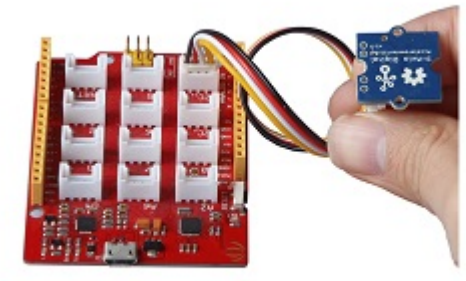
Step 3: Open Serial Monitor

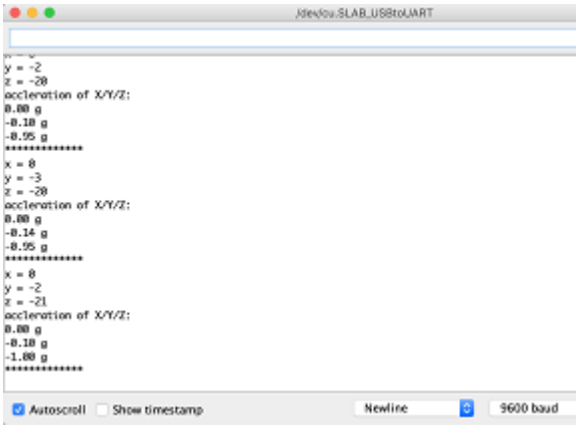
Step 4: Observe result

Please notice the data changes by positioning the 3-axis accelerometer according to pictures below.



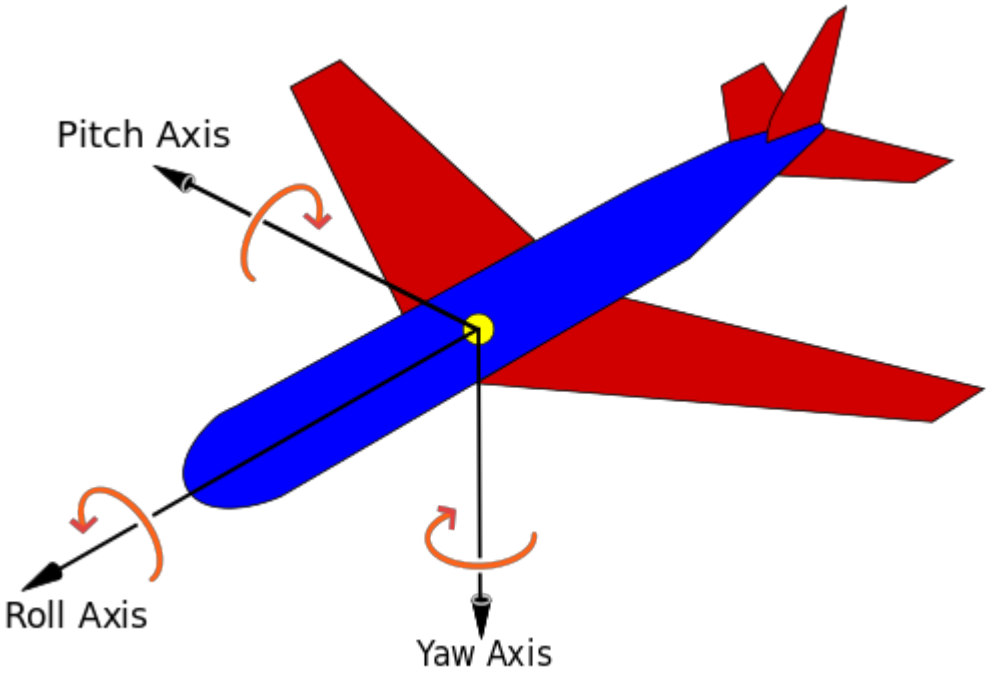






Example 2:Uses data from 3-axis accelerometer to calculate the value of Pitch and Roll

Background Knowledge:



as shown in above figure:

Pitch (Transverse axis) θ: it has its origin at the centre of gravity and is directed to the right, parallel to a line drawn from wingtip to wingtip. Motion about this axis is called pitch. A positive pitching motion raises the nose of the aircraft and lowers the tail. The elevators are the primary control of pitch. (Aircraft principal axes 2018)

Yaw (Vertical axis) ψ: it has its origin at the centre of gravity and is directed towards the bottom of the aircraft, perpendicular to the wings and to the fuselage reference line. Motion about this axis is called yaw. A positive yawing motion moves the nose of the aircraft to the right. The rudder is the primary control of yaw. (Aircraft principal axes 2018)

Roll (Longitudinal axis) Φ: it has its origin at the centre of gravity and is directed forward, parallel to the fuselage reference line. Motion about this axis is called roll. An angular displacement about this axis is called bank.[3] A positive rolling motion lifts the left wing and lowers the right wing. (Aircraft principal axes 2018)

Step 1: copy & paste the following code into Arduino IDE

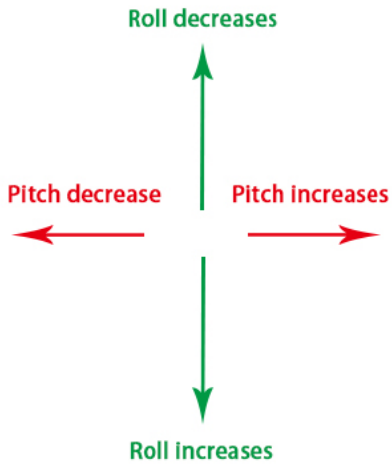
```
#include <Wire.h>
//add accelemeter library
#include "MMA7660.h"

//assign accelemeter as the name of MMA7660 accelemeter
MMA7660 accelemeter;

//set value 0.5 to alpha
const float alpha = 0.5;

//initialise fXg, fYg, fZg as double with value of 0
double fXg = 0;
```

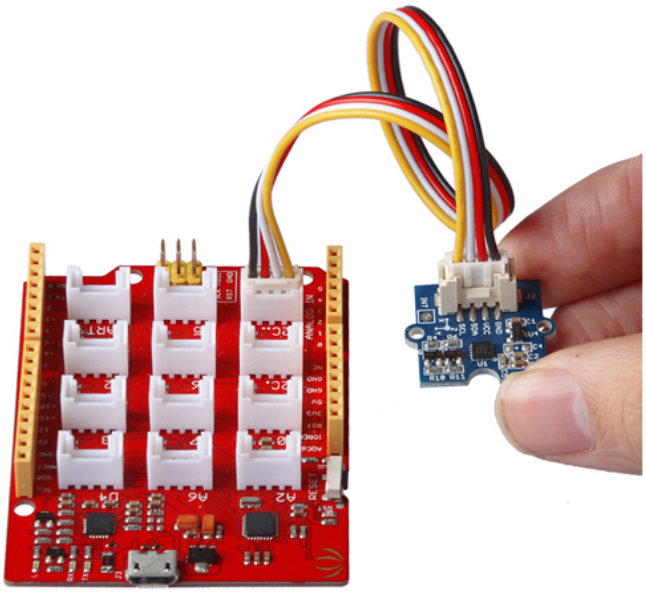
```
double fYg = 0;
double fZg = 0;
//initialise pitch and roll as double
double pitch, roll;

void setup()
{
  //initialise the accelemeter
  accelemeter.init();
  // opens serial port, sets data rate to 9600 bps
  Serial.begin(9600);
}
void loop()
{
  //initialise x, y, z as int8_t
  int8_t x;
  int8_t y;
  int8_t z;
  //get x y z offset value from accelemeter
  accelemeter.getXYZ(&x, &y, &z);

  //Low Pass Filter to reduce the noise
  fXg = x * alpha + (fXg * (1.0 - alpha));
  fYg = y * alpha + (fYg * (1.0 - alpha));
  fZg = z * alpha + (fZg * (1.0 - alpha));

  //Roll & Pitch Equations
  roll  = (atan2(-fYg, fZg) * 180.0) / M_PI;
  pitch = (atan2(fXg, sqrt(fYg * fYg + fZg * fZg)) * 180.0) / M_PI;
  //display title roll =
  Serial.print("roll = ");
  //display the roll value
  Serial.println(roll);
  //display title pitch =
  Serial.print("pitch = ");
  //display the pitch value
  Serial.println(pitch);
  delay(500);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

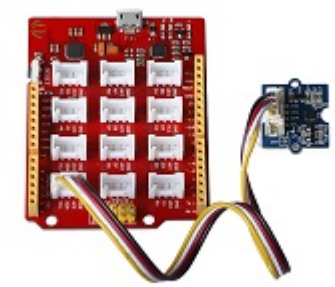
Please place the 3-axis accelerometer on a levelled surface as shown below.

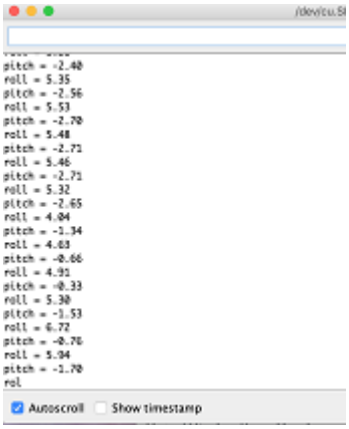


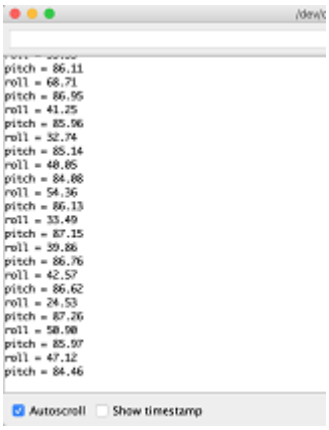


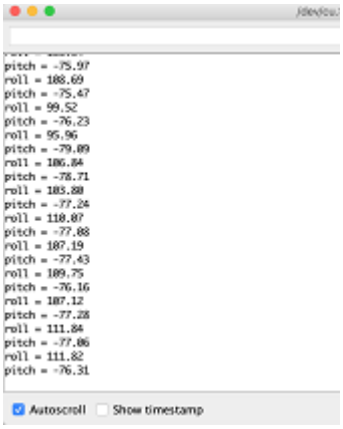
Observe the Roll data Please flip up/down the 3-axis accelerometer according to the green arrows in above figure, now the value of roll should increase when you flip down, decrease when flip up, in addition, the value of roll is positive when flip down at origin(placed parallel to water levelled), negative when flip up.

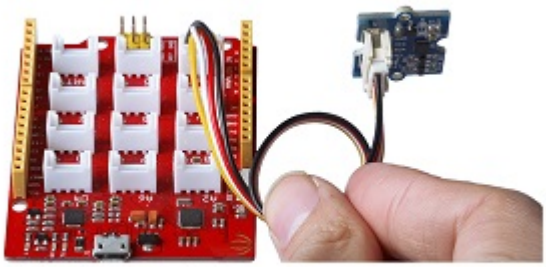
Observe the Pitch data Please tilt left/right the 3-axis accelerometer according to the red arrows in above figure, now the value of pitch should increase when you tilt right, decrease when tilt left, in addition, the value of pitch is positive when tilt right relative to origin(placed parallel to water levelled), negative when tilt left.

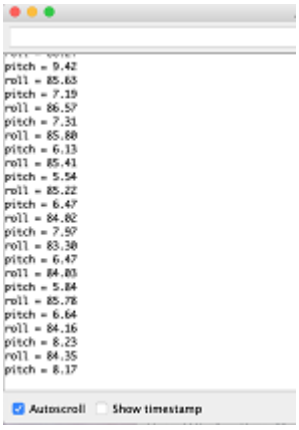


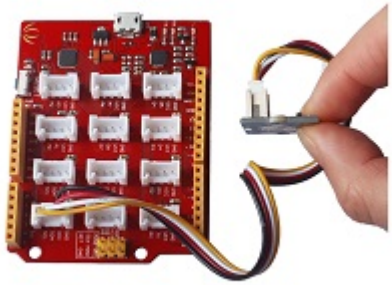


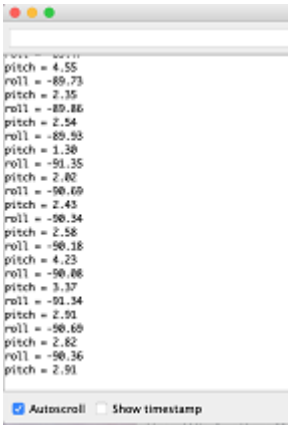








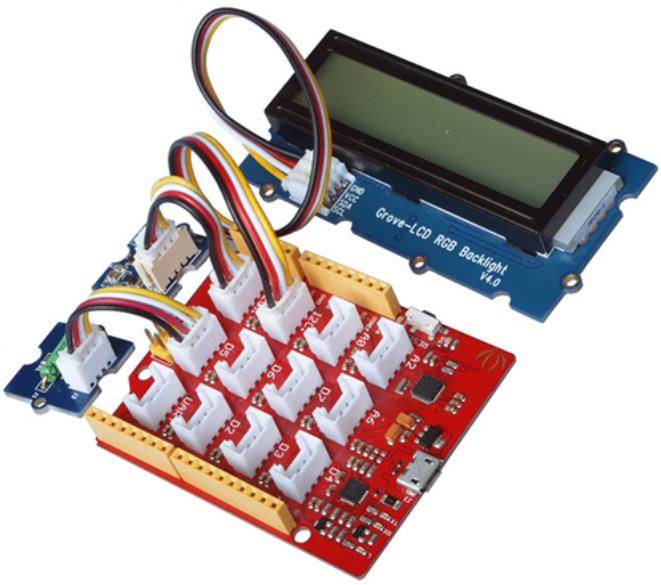




Now we can use 3-axis accelerometer to control or correct the direction of air plane or cars, by covert the pitch and roll data to control signal to control the actuator, the same technology is applied to the auto screen adjustment on mobile phone when you tilt the phone from portrait to landscape.

Example 3: Uses LCD screen to display the output data from 3-axis accelerometer

Please connect Grove – Tilt Switch to D5 port of Seeeduino Lotus, and connect Grove - LCD RGB Backlight module to I2C. port of Seeeduino Lotus, NOTE: it is the I2C port followed by one dot.



Here we use the tilt switch to flip the page on LCD screen to show different sets of data, when tilt switch is ON the LCD screen will display data of X, Y, Z Roll and Pitch, When tilt switch is at OFF position the LCD screen will display the acceleration data of aX, aY, aZ in each direction relatively.

Step 1: copy & paste the following code into Arduino IDE

```
//add accelemeter library
#include "MMA7660.h"
//add LCD library
#include <rgb_lcd.h>


//assign name tiltswitchPin to pin 5
#define tiltswitchPin 5

//assign accelemeter as the name of MMA7660 accelemeter
MMA7660 accelemeter;
//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

//set value 0.5 to alpha
const float alpha = 0.5;
```

```
//initialise fXg, fYg, fZg as double with value of 0
double fXg = 0;
double fYg = 0;
double fZg = 0;
//initialise pitch and roll as double

void setup()
{
  //initialise the accelemeter
  accelemeter.init();
  //initialise the lcd screen;
  //set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  //set pinMode of tiltswitchPin to input
  pinMode(tiltswitchPin, INPUT);
  //wait for 2s
  delay(2000);

}
void loop()
{
  /*if tilt switch is on display X, Y, Z, Roll and Pitch data
    if tilt switch is off display X, Y, Z acceleration data
  */
  if (HIGH == digitalRead(tiltswitchPin))
  {
    //initialise x, y, z as int8_t, pitch and roll as double
    int8_t x;
    int8_t y;
    int8_t z;
    double roll;
    double pitch;
    //get x y z offset value from accelemeter
    accelemeter.getXYZ(&x, &y, &z);

    //Low Pass Filter to reduce the noise
    fXg = x * alpha + (fXg * (1.0 - alpha));
    fYg = y * alpha + (fYg * (1.0 - alpha));
    fZg = z * alpha + (fZg * (1.0 - alpha));

    //Roll & Pitch Equations
    roll  = (atan2(-fYg, fZg) * 180.0) / M_PI;
    pitch = (atan2(fXg, sqrt(fYg * fYg + fZg * fZg)) * 180.0) / M_PI;
    //reset the lcd screen
    lcd.clear();
    //set the LCD cursor to column 0, line 0
    lcd.setCursor(0, 0);
    //display text x:
    lcd.print("x:");
    //display value of x
    lcd.print(x);
    //set the LCD cursor to column 5, line 0
    lcd.setCursor(5, 0);
    //display text y:
```

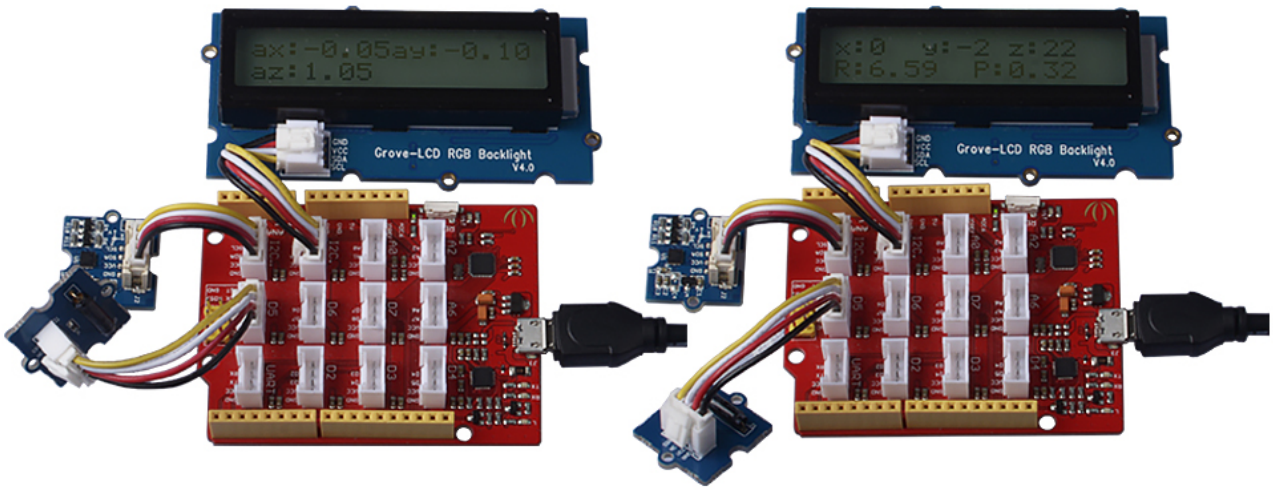
```
      lcd.print("y:");
      //display value of y
      lcd.print(y);
      //set the LCD cursor to column 10, line 0
      lcd.setCursor(10, 0);
      //display text z:
      lcd.print("z:");
      //display value of z
      lcd.print(z);

      //set the LCD cursor to column 0, line 1
      lcd.setCursor(0, 1);
      //display text R:
      lcd.print("R:");
      //display value of roll
      lcd.print(roll);
      //set the LCD cursor to column 8, line 1
      lcd.setCursor(8, 1);
      //display text P:
      lcd.print("P:");
      //display value of pitch
      lcd.print(pitch);
    } else
    {
      //initialise ax, ay, az as float
      float ax, ay, az;
      //get ax ay az acceleration value from accelemeter
      accelemeter.getAcceleration(&ax, &ay, &az);
      //reset the lcd screen
      lcd.clear();
      //set the LCD cursor to column 0, line 0
      lcd.setCursor(0, 0);
      //display text ax:
      lcd.print("ax:");
      //display value of ax
      lcd.print(ax);
      //set the LCD cursor to column 8, line 0
      lcd.setCursor(8, 0);
      //display text ay:
      lcd.print("ay:");
      //display value of ay
      lcd.print(ay);
      //set the LCD cursor to column 0, line 1
      lcd.setCursor(0, 1);
      //display text az:
      lcd.print("az:");
      //display value of az
      lcd.print(az);
    }
    //wait 0.5s
    delay(500);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

Firstly, please test if the tilt switch is altering the page of the LCD screen. Then you can rotate the 3-axis accelerometer around to observe the data change according to the rotations, get yourself familiar with the output data associate to the orientations of the 3-axis accelerometer.

Display the velocity, pitch and roll when tilt switch is on/off:



Further Explore

After play around with digital accelerometer module, you can imagine that accelerometer is one of the most important module can be found in rocket guidance system along with other modules such as GPS and gyros etc.. accelerometer is also used in mobile phones to detect if your phone is in portrait mode or landscape mode, so the screen can be tilted and adjust accordingly.

Session 9: Smart Garden

Objective

To make a Smart Garden sensor and reminder system by combining Grove starter kit modules

Key knowledge

- learn how to combine multiple modules into one application

- learn how to code for multiple devices in Arduino IDE
- adopt multiple modules to detect and analysis the planting environment, improving the logical think skills

Use case analysis

Sensor module

Use DHT11 module to monitor the surrounding environment of the plant, uses light sensor to detect the surrounding light intensity.

Actuator module

Uses buzzer to make different tones and LCD to notify different warnings message:

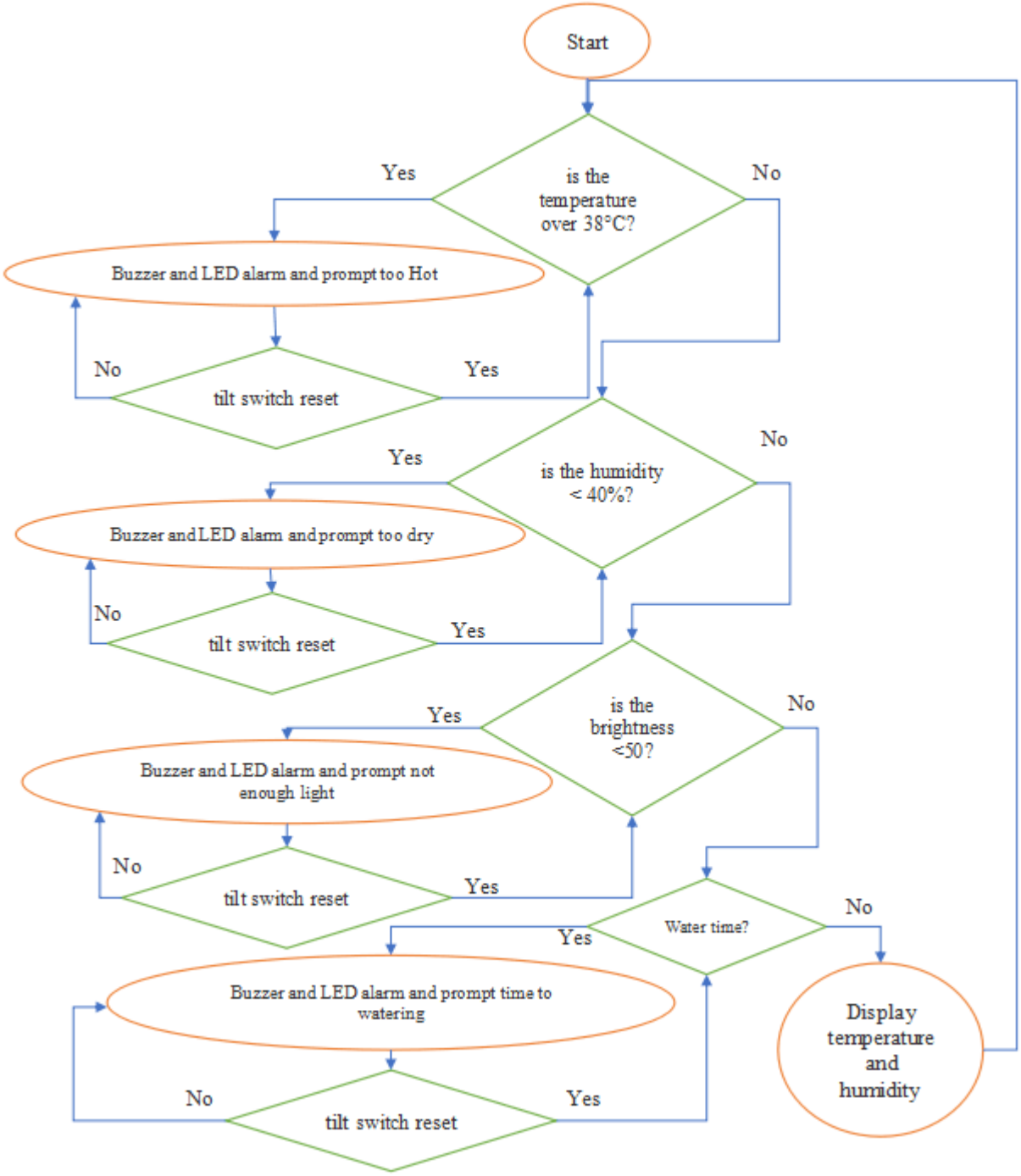
- warning 1: the surrounding temperature is higher than 38°C
- warning 2: the surrounding humidity is lower than 40%
- warning 3: Light intensity is lower than 50
- warning 4: remind user to watering plant

Uses LCD screen display:

- state 1: Show temperature
- state 2: Show humidity
- state 3: remind user to watering plant

Use tilt switch to reset warnings.

Flowchart



Hardware requirement

Self-prepare

```
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed
- DIY acrylic frame
```

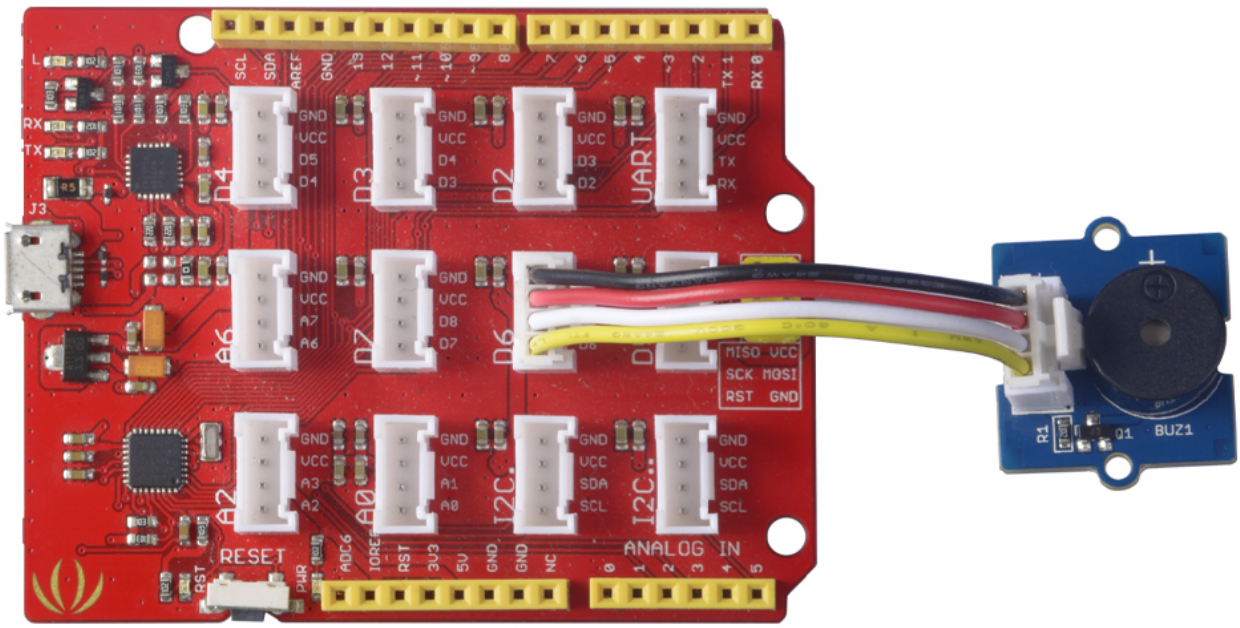
Included in the kit

```
- Seeeduino Lotus V1.1 development board
- Grove cable
```

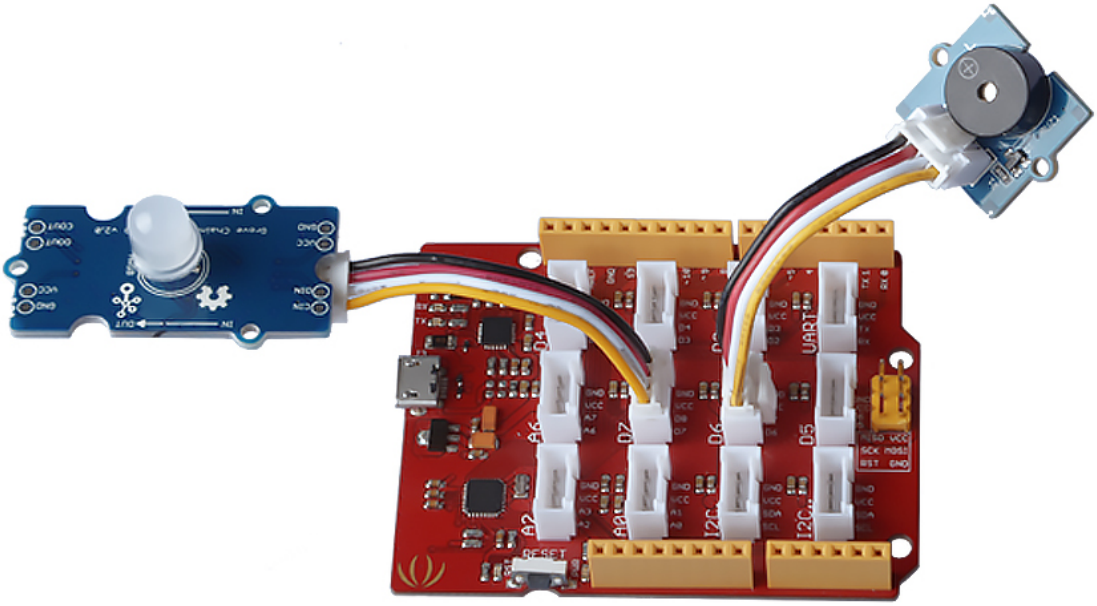
```
- Grove – Buzzer
- Grove – Chainable RGB LED
- Grove – Light Sensor
- Grove - LCD RGB Backlight
- Grove – Temperature &Humidity Sensor(DHT11)
- Grove – Tilt Switch
```

Hardware connection

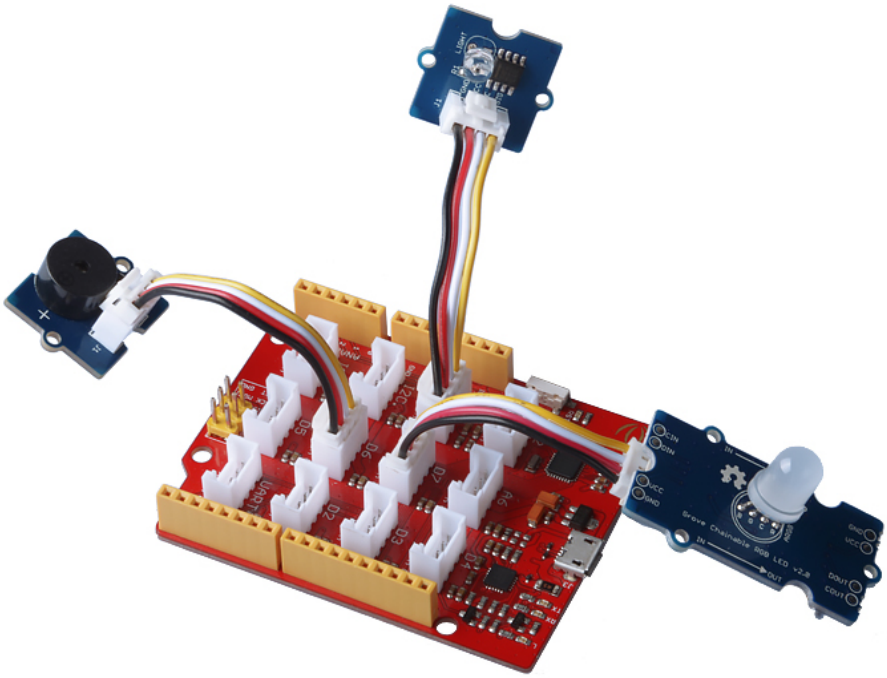
Step 1: Connect Grove – Buzzer module to D6 port of Seeeduino Lotus



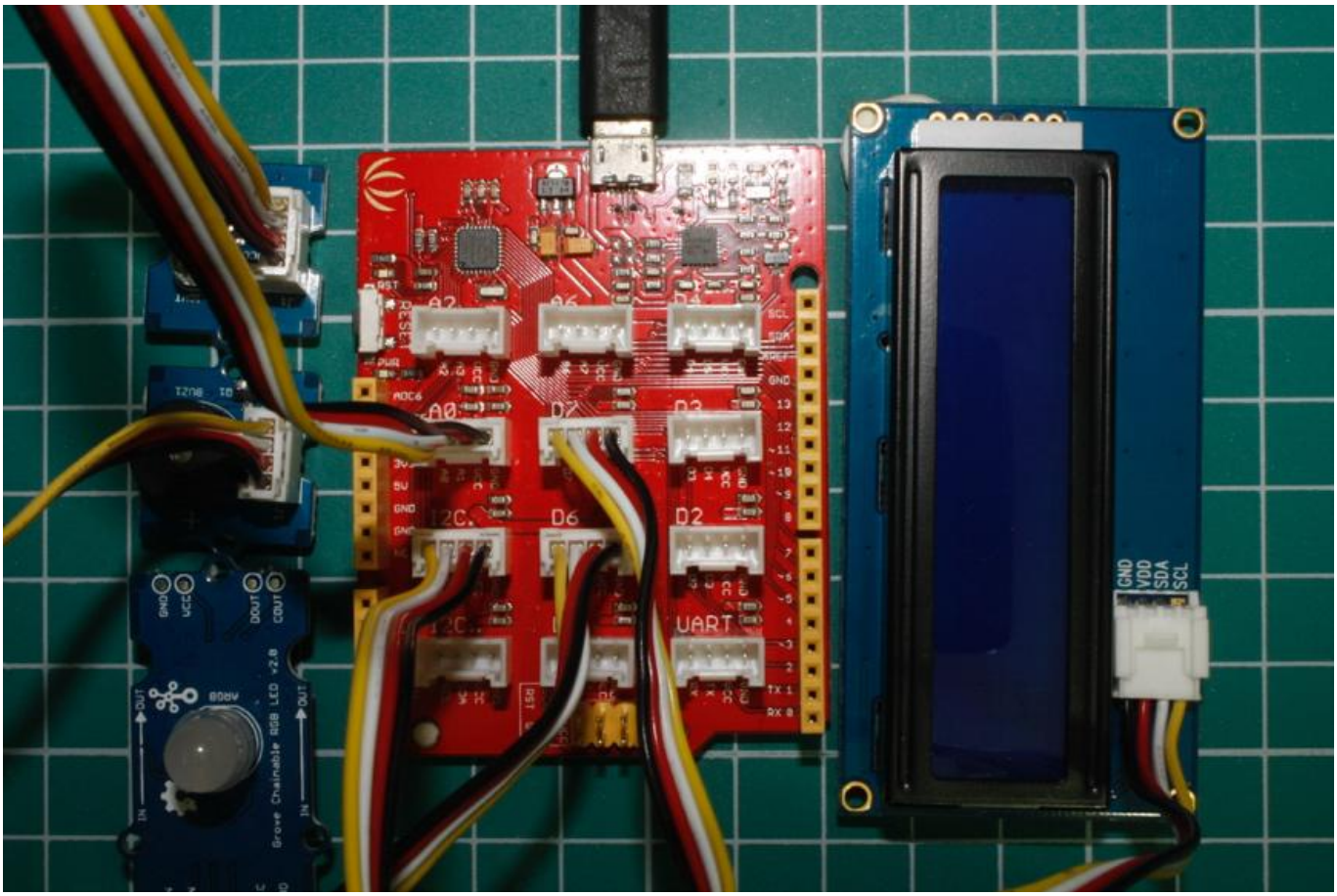
Connect Grove – Chainable RGB LED to D7 port of Seeeduino Lotus



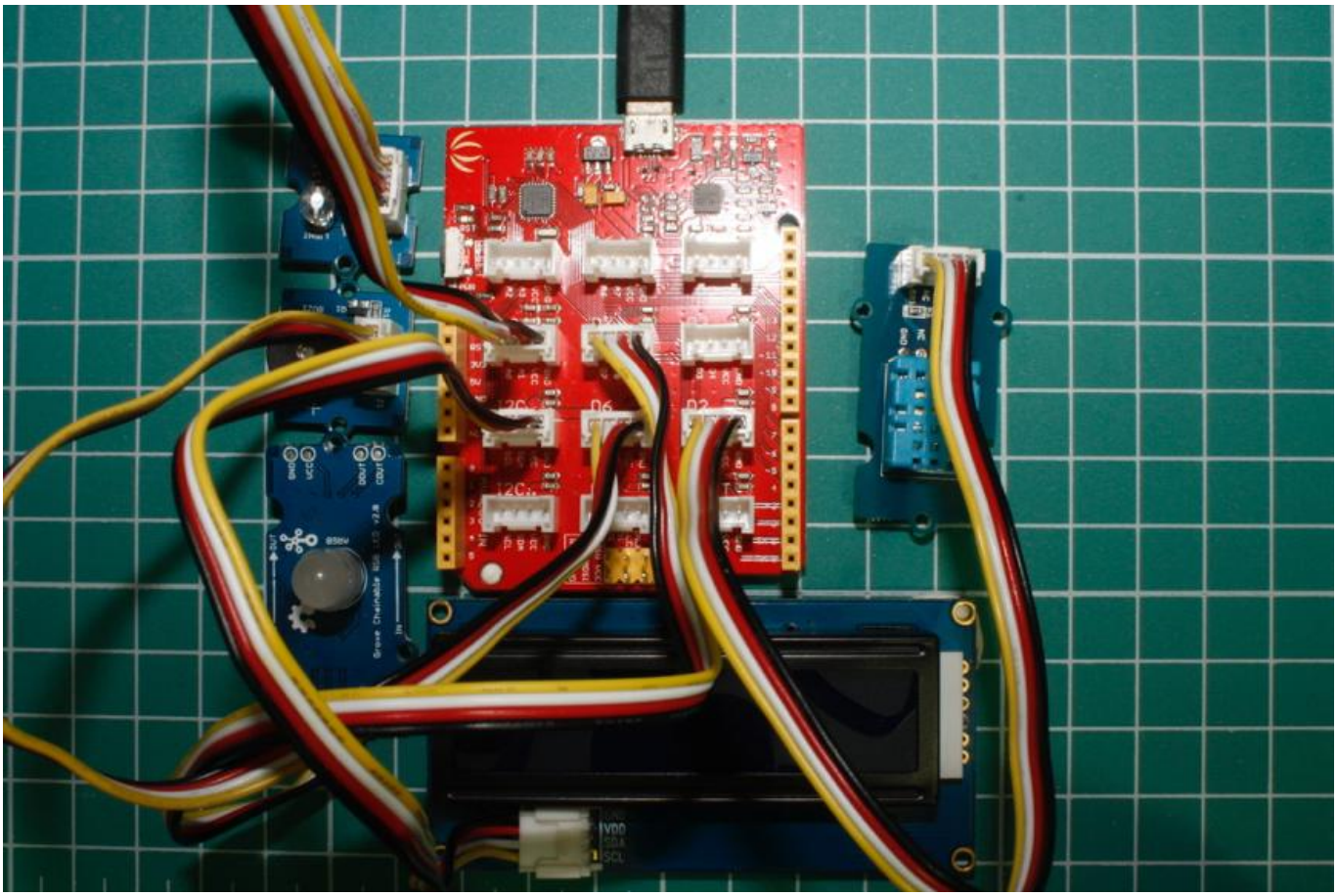
Connect Grove – Light Sensor module to A0 port of Seeeduino Lotus



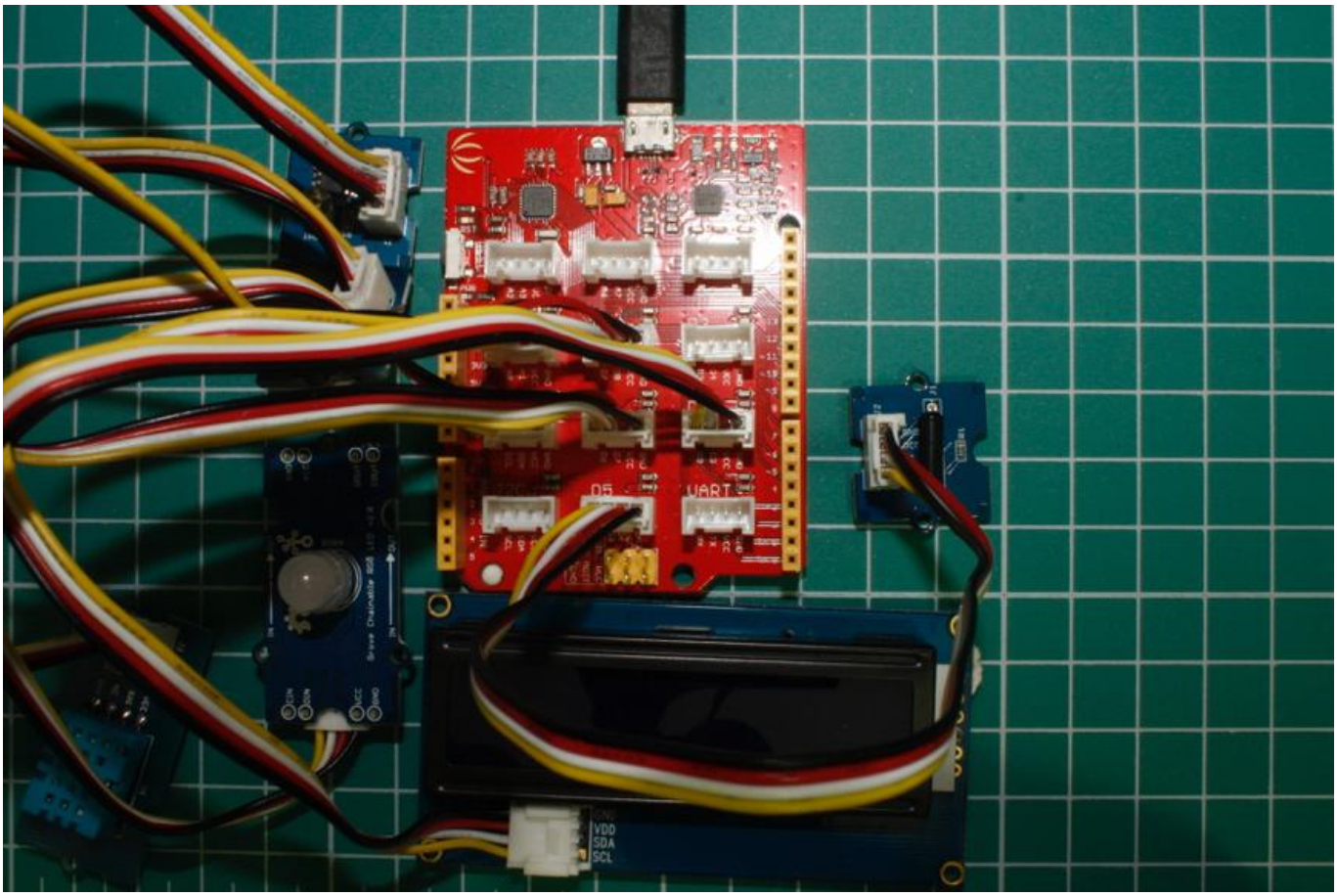
Connect Grove - LCD RGB Backlight module to I2C. port of Seeeduino Lotus note: it is the I2C port followed by one dot.



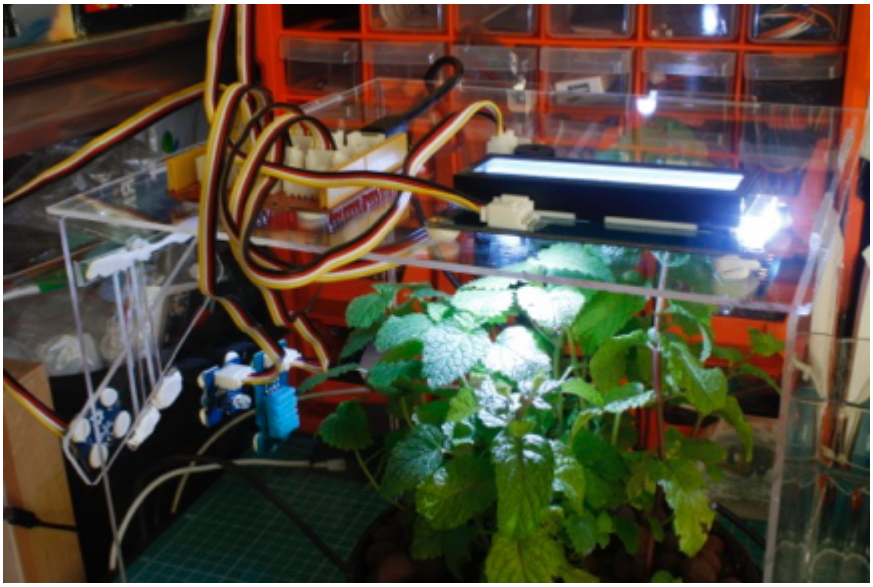
Connect Grove – Temperature &Humidity Sensor(DHT11) module to D2 port of Seeeduino Lotus.

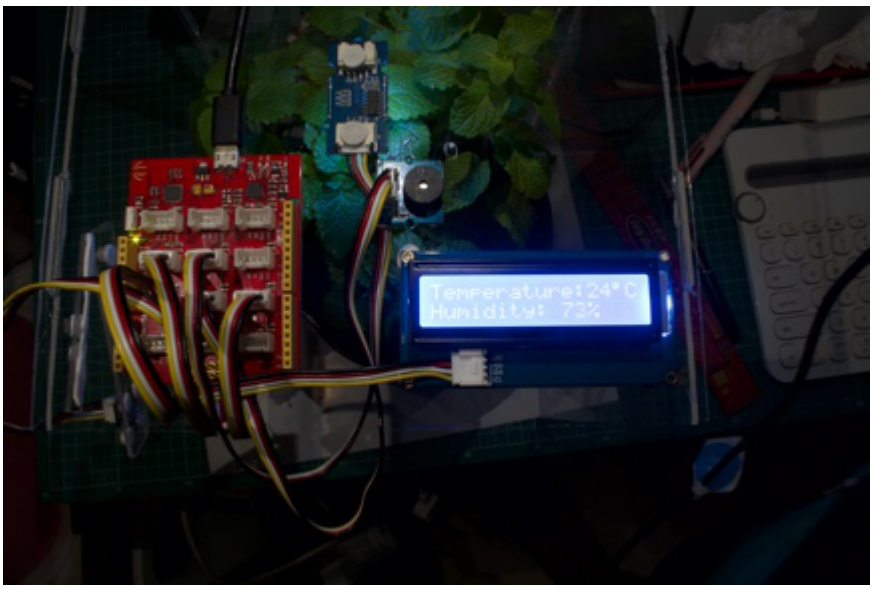


Connect Grove – Tilt Switch to D5 port of Seeeduino Lotus.



Fix all the components together on the DIY Acrylic Frame





Step 2: Link Seeeduino Lotus with computer by a micro USB cable.

Software programming

Step 1: Please add the TimeLib Library into Arduino IDE For more information please visit Arduino TimeLib tutorial

Step 2: copy & paste the following code into Arduino IDE

```
//add DHT sensor library
#include <DHT.h>
//add LCD library
#include <rgb_lcd.h>
//add ChainableLED library to this project
#include <ChainableLED.h>
//add Timelib library
#include <TimeLib.h>

//assign default time as epoch time 1514764800 which is 00:00:00 Jan 1 2018
long DEFAULT_TIME = 1514764800;
long waterTime = DEFAULT_TIME + 86400;

//set the number of leds linked to the chain
#define NUM_LEDS  1

//assign LightSensor as A0
#define LightSensor A0
//set digital pin2 as DHTPIN
#define DHTPIN 2
//set title of pin 5 as tiltSwitch
#define tiltSwitch 5
//assign buzzer as pin 6
#define buzzer 6

//set the sensor type as DHT 11
#define DHTTYPE DHT11

/*assign dht as the name of DHT sensor
  set the sensor pin as DHTPIN(pin2),
  set the sensor type as DHTTYPE(DHT11)
*/
DHT dht(DHTPIN, DHTTYPE);

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

void setup()
{
  //
  setTime(DEFAULT_TIME);
  //initialise the dht sensor
  dht.begin();
```

```
  //initialise the lcd screen;
  //set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  //initialise ChainableLED leds
  leds.init();
  //set pin 5(tilt switch) as input pin
  pinMode(tiltSwitch, INPUT);
  delay(1000);
}
int mode = 0;
void loop()
{
  //-------------DHT--------------------
  //store the humidity value to h
  int h = dht.readHumidity();

  //store the temperature value to t(in Celsius)
  int t = dht.readTemperature();

  int value = analogRead(LightSensor);
  float value_float = map(value, 0, 800, 50, 0) / 100.0;

  leds.setColorHSB(0, 0, 0, value_float);

  //initialise mode to 0, then set to case 0;

  //temperature exceed 38 degrees, then set to case 1;
  if (t > 38) {
    mode = 1;
  }
  //Humidity is less than 40 %, then set to case 2;
  if (h < 40)
  {
    mode = 2;
  }
  //LightSensor reading value is less than 50, then set to case 3;
  if (value < 50)
  {
    mode = 3;
  }
  //current time is greate or equals to waterTime(24 hour ahead), then set to case
4;
  if (now() >= waterTime  ) {
    mode = 4;
  }

  switch (mode) {
    case 0:
      //set the LCD cursor to column 0, line 0
      lcd.clear();
      lcd.setCursor(0, 0);
      //Print text temperature: to the LCD
      lcd.print("Temperature:");
      //set the LCD cursor to column 12, line 0
```

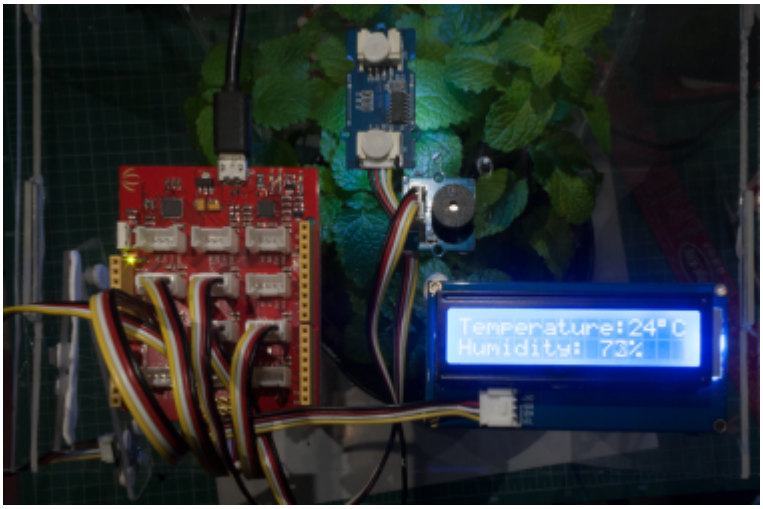
```
        lcd.setCursor(12, 0);
        //Print temperature value t to the LCD
        lcd.print(t);
        //set the LCD cursor to column 14, line 0
        lcd.setCursor(14, 0);
        //Print temperature º is character 223 on lookup table
        lcd.write(223);
        //Print C to the LCD
        lcd.print("C");

        //set the LCD cursor to column 0, line 1
        lcd.setCursor(0, 1);
        //Print text Humidity: to the LCD
        lcd.print("Humidity: ");
        //set the LCD cursor to column 10, line 1
        lcd.setCursor(10, 1);
        //Print humidity value h to the LCD
        lcd.print(h);
        //set the LCD cursor to column 12, line 1
        lcd.setCursor(12, 1);
        //Print sign % to the LCD
        lcd.print("%");
        break;
      case 1:
        tone(buzzer, 262, 300);
        leds.setColorRGB(0, 255, 0, 0);
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Temperature too ");
        lcd.setCursor(0, 1);
        lcd.print("High!!");
        if (HIGH == digitalRead(tiltSwitch))
        {
          leds.setColorRGB(0, 0, 0, 0);
          mode = 0;
        }
        break;
      case 2:
        tone(buzzer, 294, 300);
        leds.setColorRGB(0, 255, 0, 0);
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Warning! Too Dry");
        if (HIGH == digitalRead(tiltSwitch))
        {
          leds.setColorRGB(0, 0, 0, 0);
          mode = 0;
        }
        break;
      case 3:
        tone(buzzer, 330, 300);
        leds.setColorRGB(0, 255, 0, 0);
        lcd.clear();
        lcd.setCursor(0, 0);
```

```
        lcd.print("Not Enough Light");
        lcd.setCursor(0, 1);
        lcd.print("Check the LED..");
        if (HIGH == digitalRead(tiltSwitch))
        {
          leds.setColorRGB(0, 0, 0, 0);
          mode = 0;
        }
        break;
      case 4:
        tone(buzzer, 349, 300);
        leds.setColorRGB(0, 255, 0, 0);
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Time to water");
        lcd.setCursor(0, 1);
        lcd.print("the plants");
        if (HIGH == digitalRead(tiltSwitch))
        {
          waterTime = now() + 86400;
          mode = 0;
        }
        break;

    }
  }
```

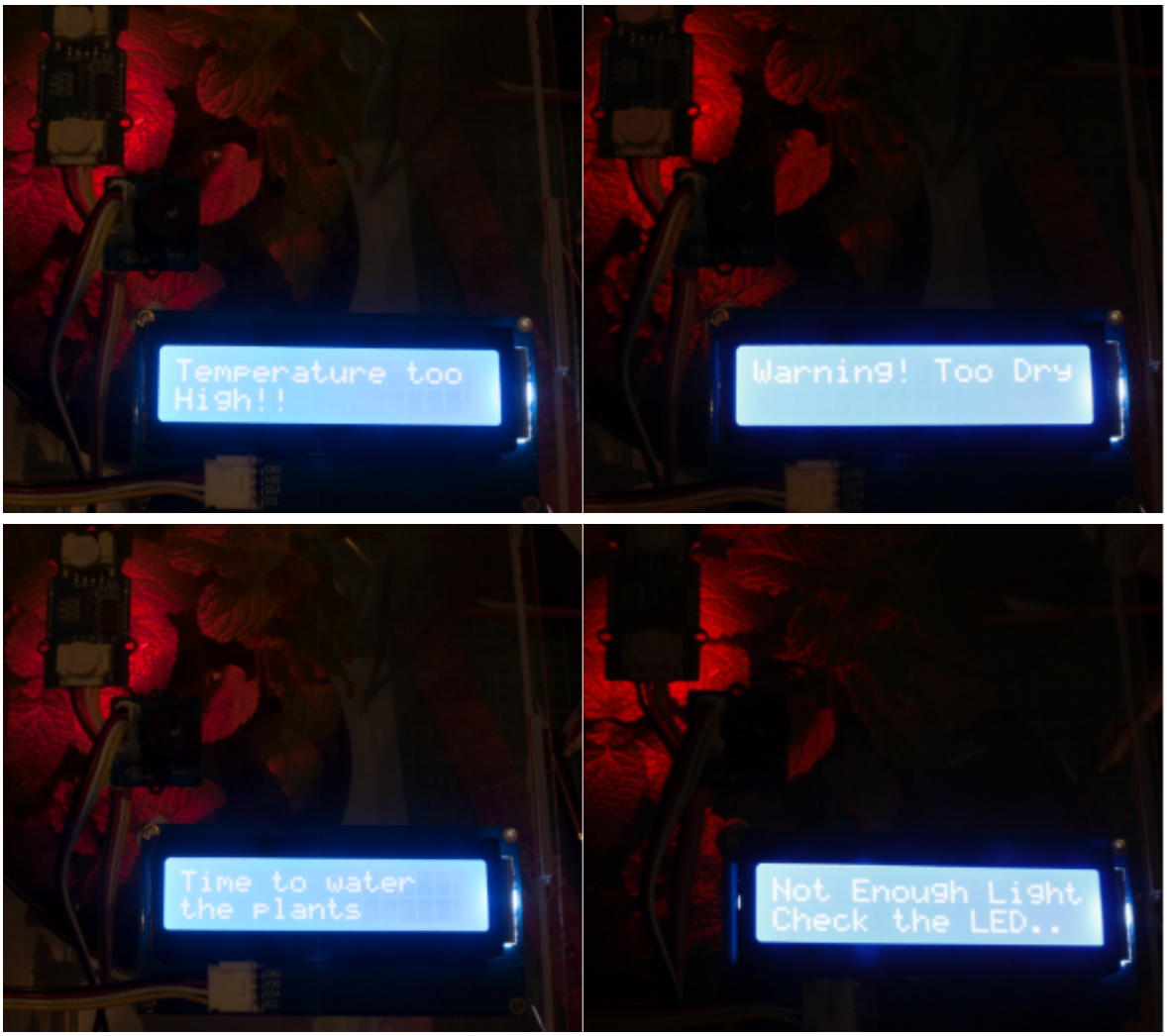
Step 3: Upload code into Seeeduino Lotus

Step 4: Observe result

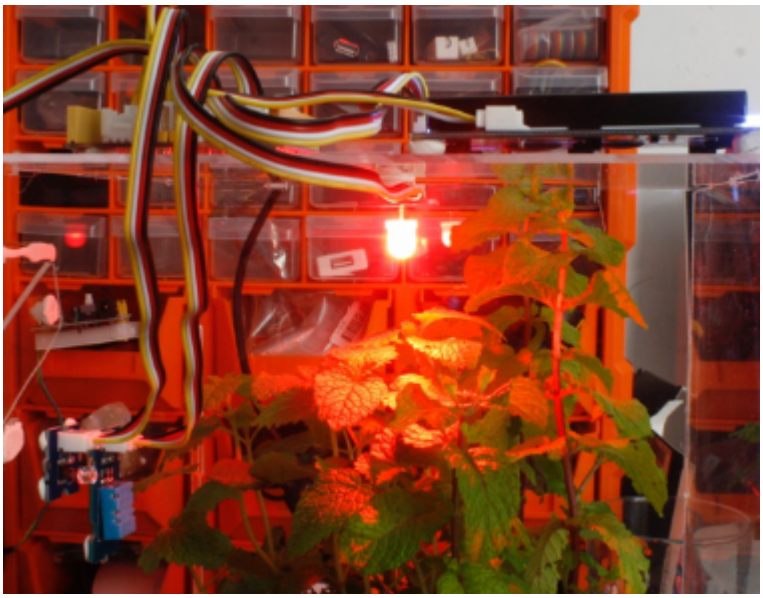
Under normal condition, the LED shines white light and the LCD screen shows temperature and humidity.



4 warning states

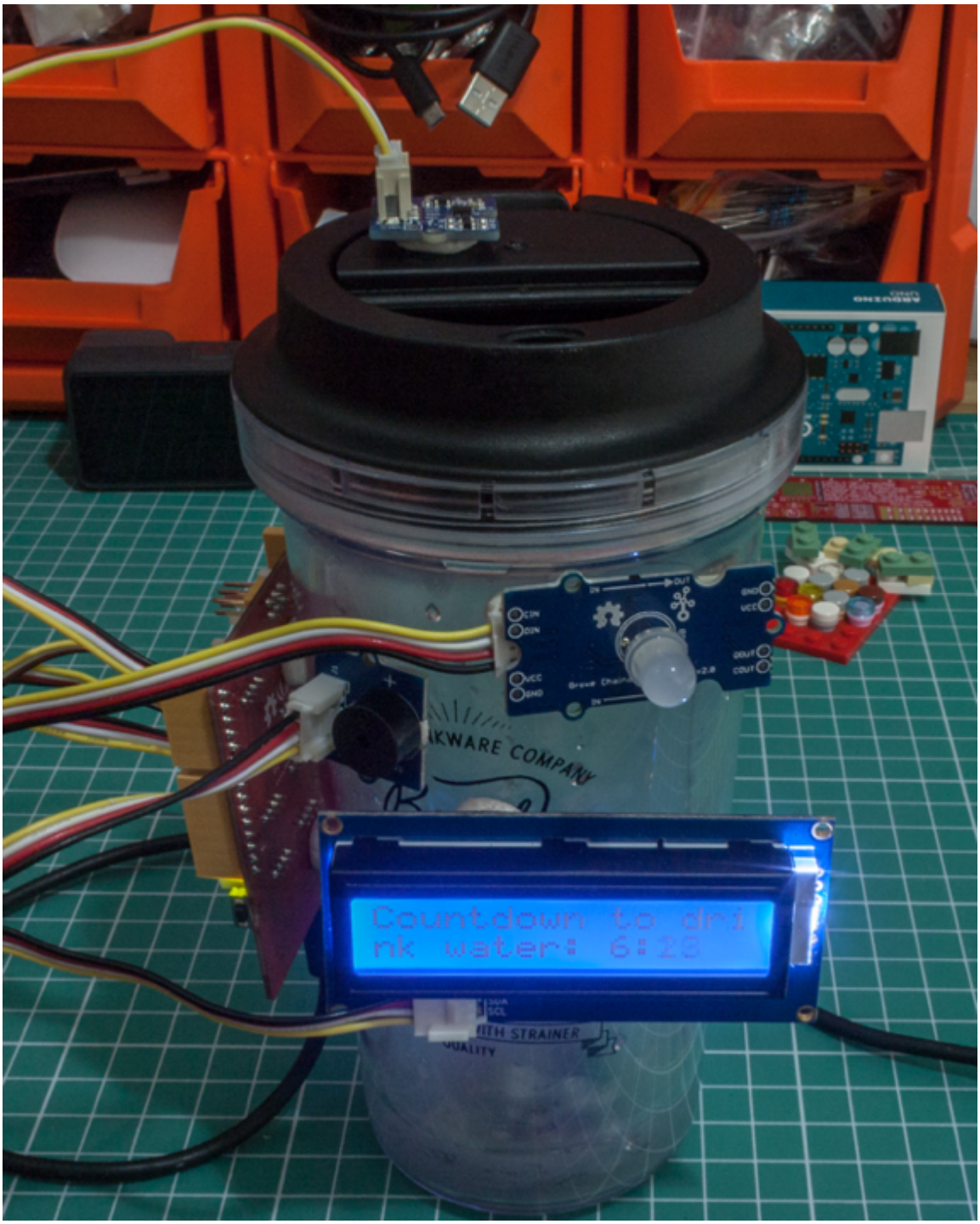


Red LED warning



Reset warnings by using tilt switch

Session 10: Smart Cup



Objective

Make a smart cup by using buzzer, RGB LED, 3-Axis Accelerometer and LCD screen, it will remind user to drink water at a certain period of time, the smart cup can also detect if the user consume some water, the LCD screen will display a counting down time to remind user when will be the next time to drink water.

Key knowledge

- revise the TimeLib Library on setting and control time with Lotus.
- revise display and scrolling text on LCD screen
- revise if...else...and switch...case...operation with || (or) and &&(and) logic operator.
- further examining the applications of pitch and roll value reading from 3-Axis Accelerometer.
- use BlinkWithoutDelay example method to avoid using Delay function which prevent delay function to miss up the system timer.

- learn how to make and call customized function, the return result could be Boolean(trun/false), or a value of the variable by using return X.

Use case analysis

Sensor module

By comparing the pitch and roll data from 3-axis accelerometer readings to detect if the bottle is tilted or not, therefore it recognises if the user is drinking water or not. if the bottle is tilted, the next step is detecting if the bottle has been put back on the table, once the bottle is on the table, the pitch and roll data from 3-axis accelerometer will calibrate the maximum and minimum value for comparing.

Actuator module

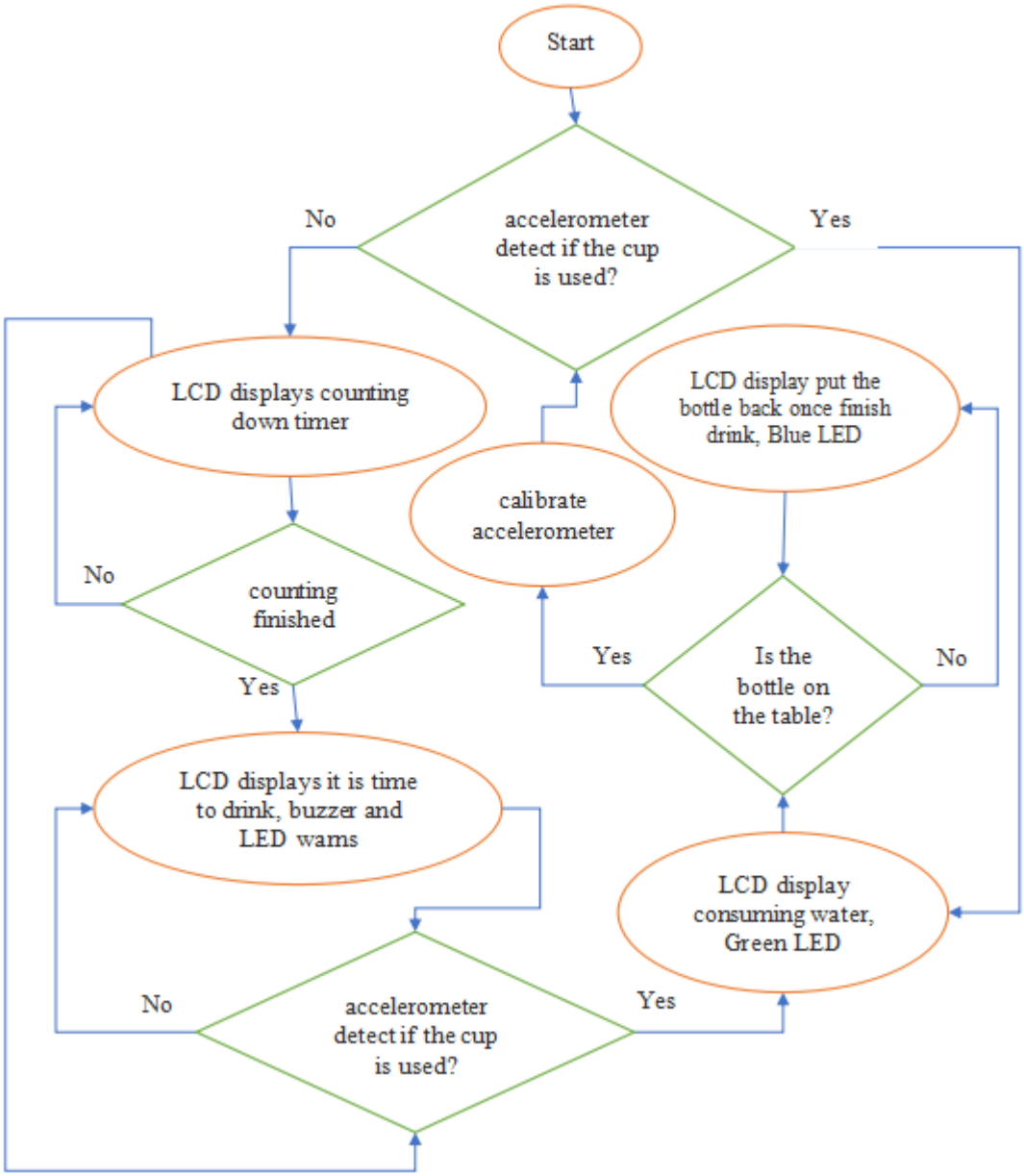
Uses buzzer to make different tones to remind different state:

- state 1: when timer counting down 30min finished, the buzzer will buzz to remind user to drink water
- state 2: the buzzer will buzz if the bottle is not stay still on the table.

Uses LCD screen display

- state 1: counting down timer
- state 2: remind user to drink water
- state 3: congratulate the user for drink water
- state 4: tell user to put back water once finished drinking

Flowchart



Hardware requirement

Self-prepare

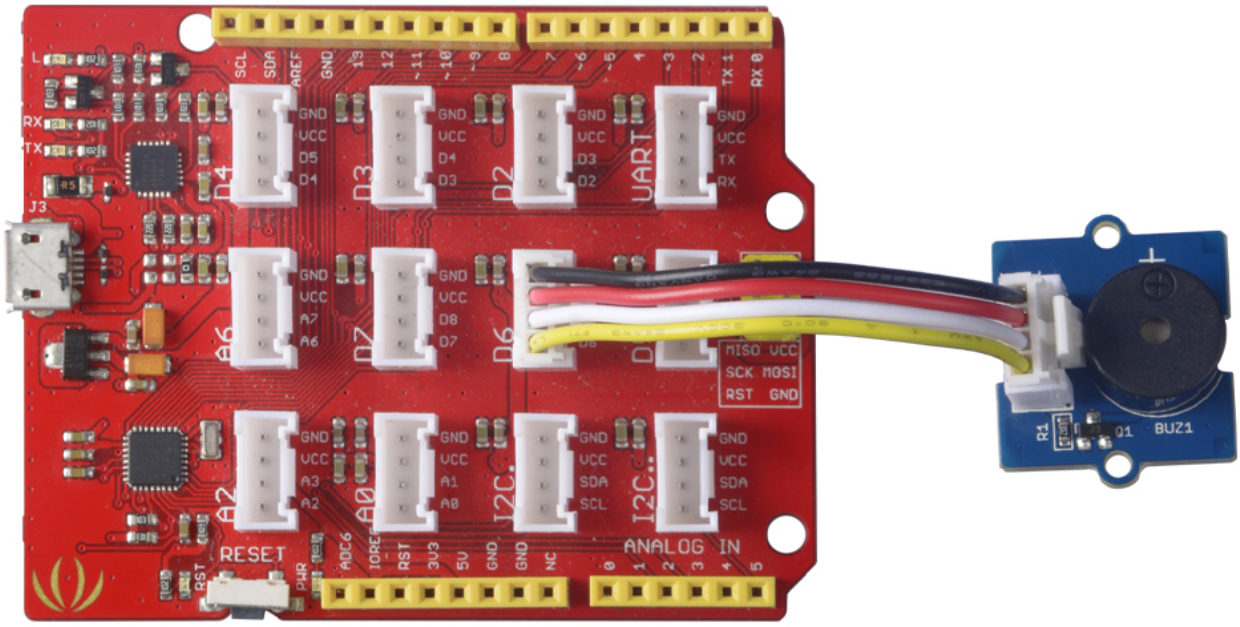
- micro-USB cable
- a computer with Arduino IDE and serial-to-USB driver installed

Included in the kit

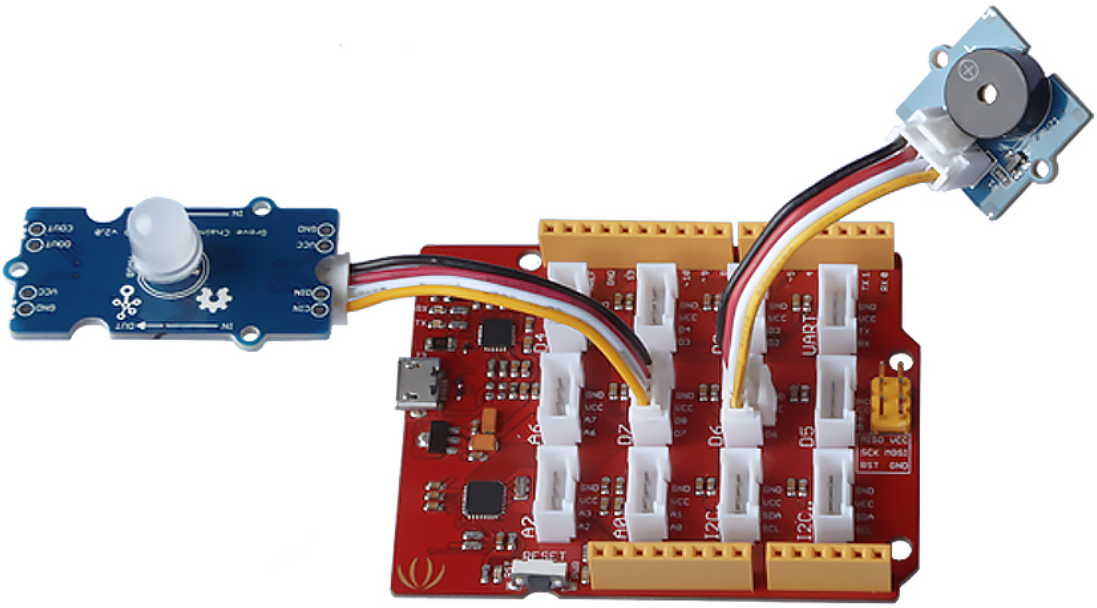
- Seeeduino Lotus V1.1 development board
- Grove cable
- Grove – Buzzer
- Grove – LED Chainable RGB LED
- Grove - LCD RGB Backlight
- Grove – 3-Axis Digital Accelerometer

Hardware connection

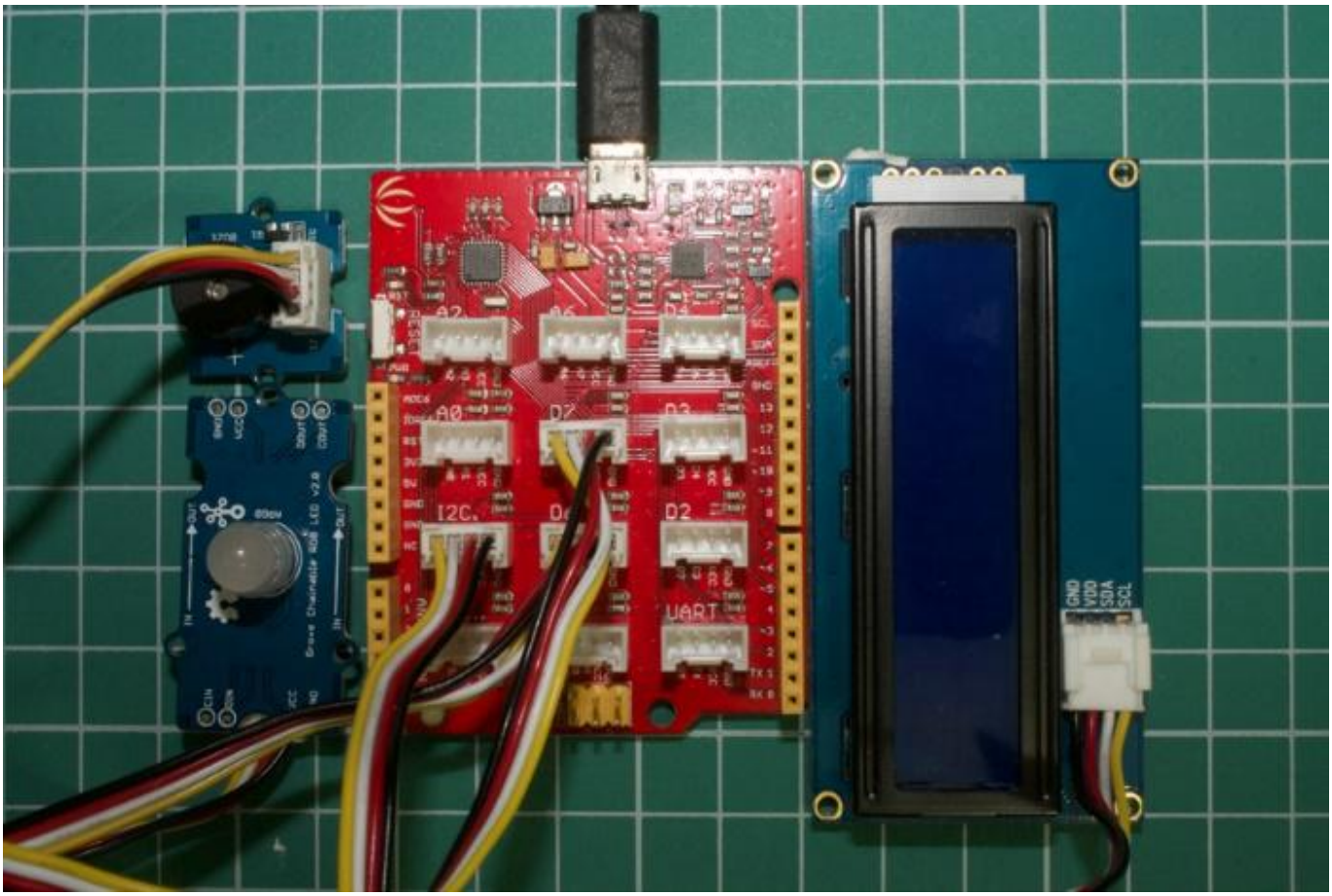
Step 1: Connect Grove - Buzzer module to D6 port of Seeeduino Lotus



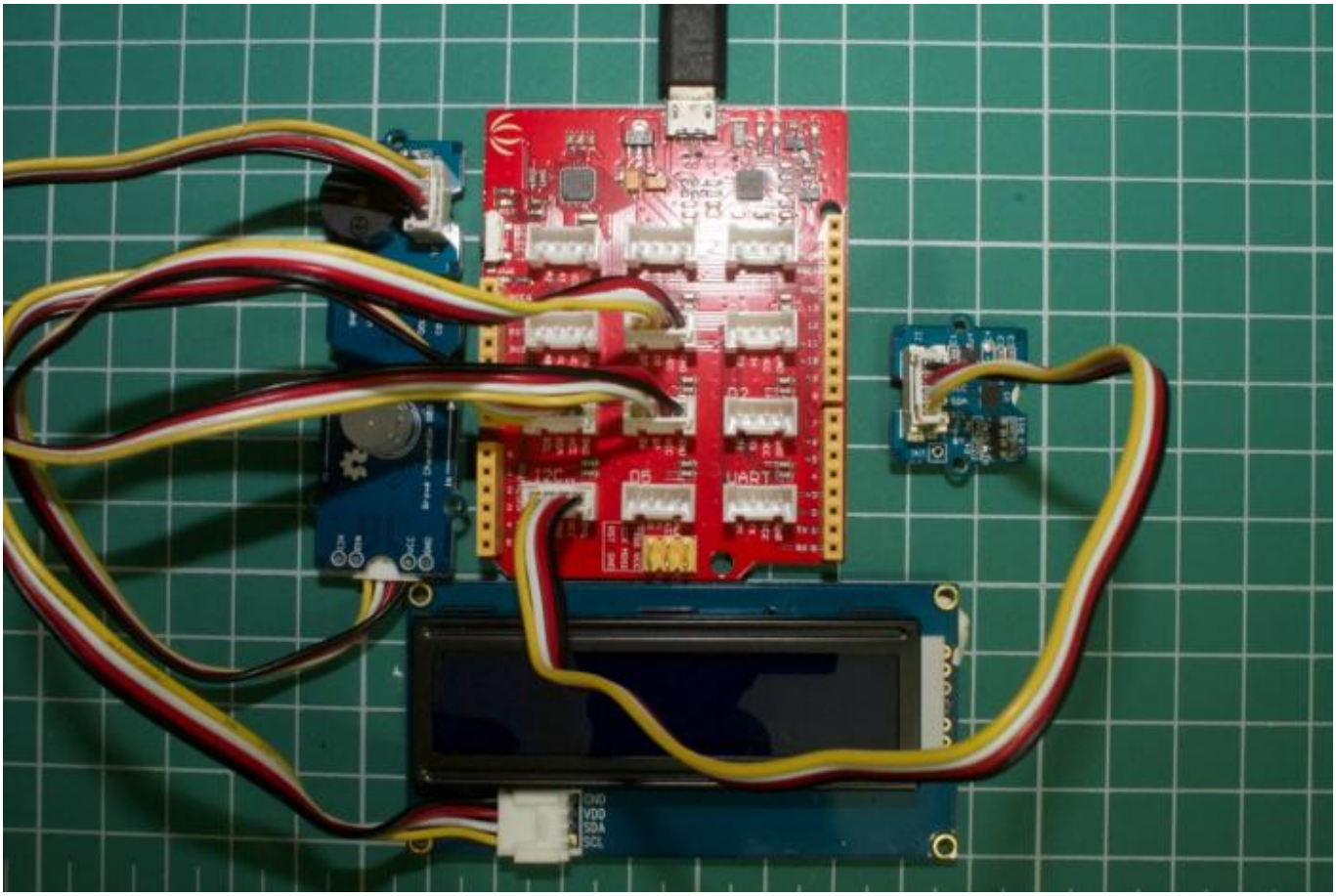
Connect Grove – Chainable RGB LED to D7 port of Seeeduino Lotus



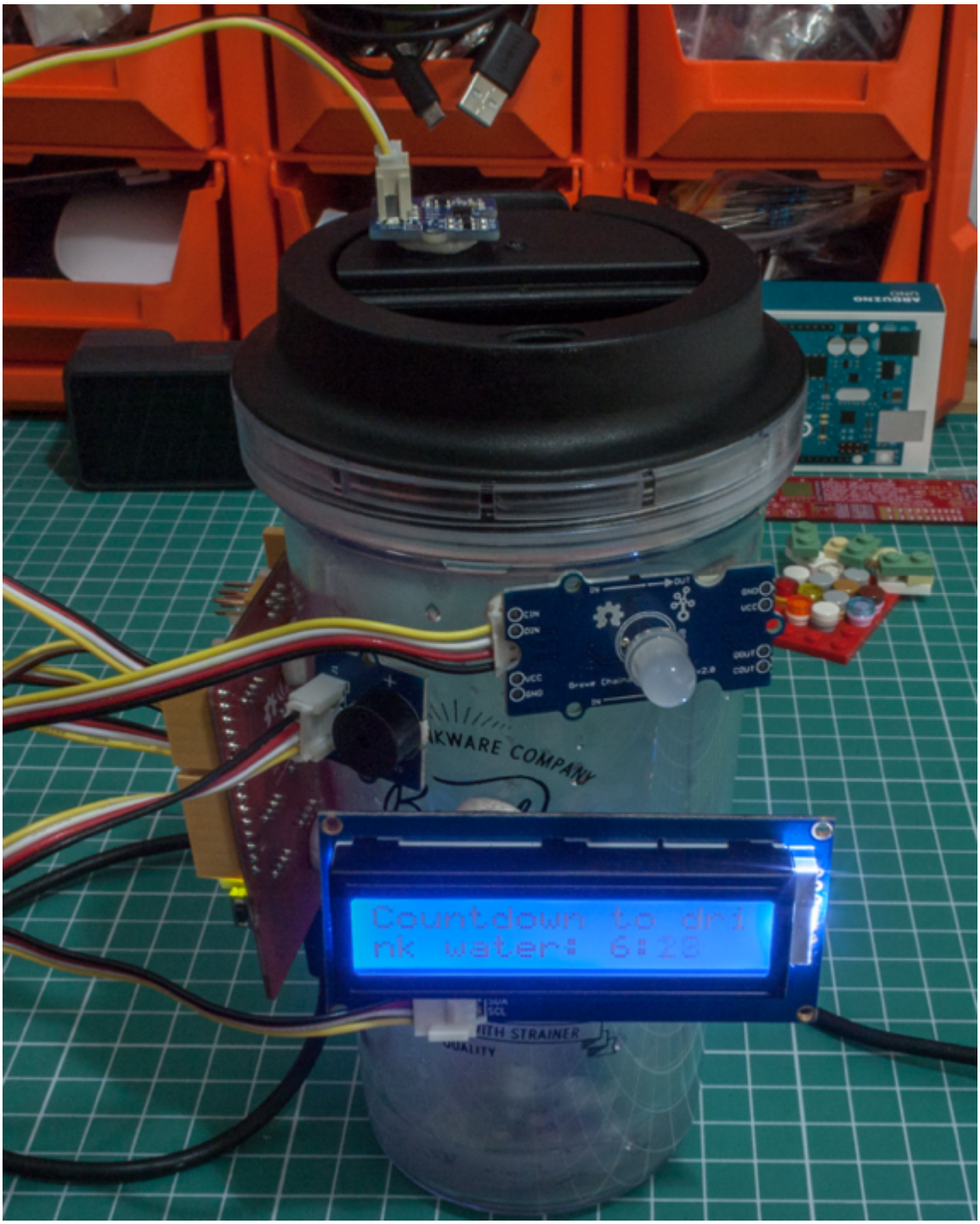
Connect Grove - LCD RGB Backlight module to I2C. port of Seeeduino Lotus note: it is the I2C port followed by one dot.



Connect Grove – 3-Axis Digital Accelerometer to I2C port of Seeeduino Lotus note: it is the I2C port followed by two dots.



Fix all the components together on a cup.



Software programming

Step 1: Please add the TimeLib Library into Arduino IDE

Copy & paste the following code into Arduino IDE

```
//add LCD library
#include <rgb_lcd.h>
//add accelerometer library
#include "MMA7660.h"
//add Timelib library
```

```
#include <TimeLib.h>
//add ChainableLED library to this project
#include <ChainableLED.h>

//assign default time as epoch time 1514764800 which is 00:00:00 Jan 1 2018
long DEFAULT_TIME = 1514764800;
//set drinkTime at 30mins(1800s) after default time
//long drinkTime = DEFAULT_TIME + 1800;
long drinkTime = DEFAULT_TIME + 10;
int lastDrink, seconds;
//will store lastest time in milliseconds
unsigned long previousMillis = 0;

//set the number of leds linked to the chain
#define NUM_LEDS  1

//assign buzzer as pin 6
#define buzzer 6
//set title of pin 5 as tiltSwitch
#define tiltSwitch 5

//assign accelerometer as the name of MMA7660 accelerometer
MMA7660 accelerometer;

//assign lcd as the name of rgb_lcd screen
rgb_lcd lcd;

/*assign leds as the name of
  the ChainableLED set the
  pin of the ChainableLED to
  pin7(clock pin) and pin8(data pin)
  and number of the leds*/
ChainableLED leds(7, 8, NUM_LEDS);

//set motion check tolerance value
int tolerance = 50;
//initalise the calibrated and moveDetedted as false
boolean calibrated = false;
boolean moveDetected = false;

//set int8_t for accelerometer reading value x, y, z
int8_t x;
int8_t y;
int8_t z;

//initialise fXg, fYg, fZg as double with value of 0
double fXg = 0;
double fYg = 0;
double fZg = 0;
//initialise pitch and roll as double
double p, r;

//Accelerometer limits
double rMin; //Minimum roll Value
```

```
double rMax; //Maximum roll Value
double rVal; //Current roll Value

double pMin; //Minimum pitch Value
double pMax; //Maximum pitch Value
double pVal; //Current pitch Value

//set value 0.5 to alpha for low pass filter tolerance
const float alpha = 0.5;

//iinitialise mode to set the default switch case to first(count from 0)
int mode = 0;

void setup()
{
  //set the system time to 00:00:00 Jan 1 2018
  setTime(DEFAULT_TIME);
  //initialise the accelerometer
  accelerometer.init();
  //initialise ChainableLED leds
  leds.init();
  //initialise the lcd screen;
  //set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  //set pin 5(tilt switch) as input pin
  pinMode(tiltSwitch, INPUT);
  //calibrate the accelerometer for at the begining
  calibrateAccel();
  //wait for 2
  delay(2000);
}

//setup accelerometer reading function output mapped value of roll and pitch
void Accel() {
  accelerometer.getXYZ(&x, &y, &z);

  //Low Pass Filter to reduce the noise
  fXg = x * alpha + (fXg * (1.0 - alpha));
  fYg = y * alpha + (fYg * (1.0 - alpha));
  fZg = z * alpha + (fZg * (1.0 - alpha));

  r  = (atan2(-fYg, fZg) * 180.0) / M_PI;
  p = (atan2(fXg, sqrt(fYg * fYg + fZg * fZg)) * 180.0) / M_PI;
  r = map(r, -90, 90, 0, 180);
  p = map(p, -90, 90, 0, 180);
  return r;
  return p;
}

//setup function for calibrate the accelerometer
void calibrateAccel() {
  //reset moveDetected to false
  moveDetected = false;
```

```
  //call accelerometer reading funtion
  Accel();

  //assign the reading of roll and pitch
  rVal = r;
  rMin = rVal;
  rMax = rVal;

  pVal = p;
  pMin = pVal;
  pMax = pVal;

  //calibrate the Accelerometer
  for (int i = 0; i < 50; i++) {
    //call accelerometer reading funtion
    Accel();
    /*--calibrate roll---*/
    //assign the reading of roll to rVal
    rVal = r;
    //evaluate if the new reading is greater than stored Maximum value
    if (rVal > rMax) {
      //if new reading value is greater save new value to rMax
      rMax = rVal;
      //evaluate if the new reading is less than stored Minimum value
    } else if (rVal < rMin) {
      //if new reading value is less save new value to rMin
      rMin = rVal;
    }

    /*--calibrate pitch---*/
    //assign the reading of pitch to pVal
    pVal = p;
    //evaluate if the new reading is greater than stored Maximum value
    if (pVal > pMax) {
      //if new reading value is greater save new value to pMax
      pMax = pVal;
      //evaluate if the new reading is less than stored Minimum value
    } else if (pVal < pMin) {
      //if new reading value is less save new value to pMin
      pMin = pVal;
    }
    //Delay 10ms between readings
    delay(10);
  }
  //set the calibrated to true
  calibrated = true;
}

//drinking function check if the bottle is tilting output ture/false
boolean drinking() {
  //initialise tilting as false
  boolean tilting = false;
  //reading from accelerometer
  Accel();
```

```
  rVal = r;
  pVal = p;
  /*evaluate if new roll value is greater than the maximum value or
     less than the minimum value saved previously.
     || means or
     if rolling is happening then set tilting to ture
     if pitch is happening then set tilting to ture
  */
  if (rVal > (rMax + tolerance) || rVal < (rMin - tolerance)) {
    tilting = true;
  }

  if (pVal > (pMax + tolerance) || pVal < (pMin - tolerance)) {
    tilting = true;
  }
  //output tilting
  return tilting;
}

//mothin function
void Motion() {
  //don't check for movement until recalibrated again
  calibrated = false;
}

void loop()
{
  /*evaluate if current time is greate or equals
    to drinkTime(30mins ahead), then switch to case 1;
    its time to drink
  */
  if (now() >= drinkTime  ) {
    //switch to case 1
    mode = 1;
  }
  //evaluate if the accelerometer is calibrated
  if (calibrated) {
    //evaluate if the bottle is tilted
    if (drinking()) {
      //switch to case 2
      mode = 2;
      //set moveDetected to true
      moveDetected = true;
    }
  }
  //evaluate if the moveDetected is true
  if (moveDetected) {
    //call motion function
    Motion();
  }
  //save current time in millisecond
  unsigned long currentMillis = millis();
  switch (mode) {
```

```
      /*Case 0:
        mode to display countdonw time if nothing happened
      */
      case 0:
        //minutes to drink water
        lastDrink = (drinkTime - now()) / 60;
        //seconds to drink water
        seconds = (drinkTime - now()) % 60;

        leds.setColorHSB(0, 0, 0, 0);

        /*refesh the LCD for 1s without using delay, refer
           to Example "BlinkWithoutDelay", so the system
           won't stop and wait
        */
        if (currentMillis - previousMillis >= 1000) {
          // save the last time you refreshed the LCD
          previousMillis = currentMillis;
          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print("Countdown to dri");
          lcd.setCursor(0, 1);
          lcd.print("nk water: ");
          lcd.setCursor(10, 1);
          lcd.print(lastDrink);
          lcd.print(":");
          lcd.print(seconds);
        }
        break;
      /*Case 1:
         reached 30mins time to drink some water
         with buzzer alarm and LCD display time
         to drink some water
      */
      case 1:
        tone(buzzer, 262, 300);
        leds.setColorRGB(0, 255, 0, 0);
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Time to drink");
        lcd.setCursor(0, 1);
        lcd.print("Some water");
        break;
      /*Case 2:
         detect if the wate bottle is tilted
         therefore user is drinking some water
         and recalibrate the sensor(accelerometer)
         once the bottle has been put on a flat
         surface if the bottle is still tilted or
         not sitting flat(accelerometer reading
         is not around 90 degrees), enter case 3
         detected the bottle is resting still enter
         to case 0 and reset the drink time to 30mins
         ahead
```

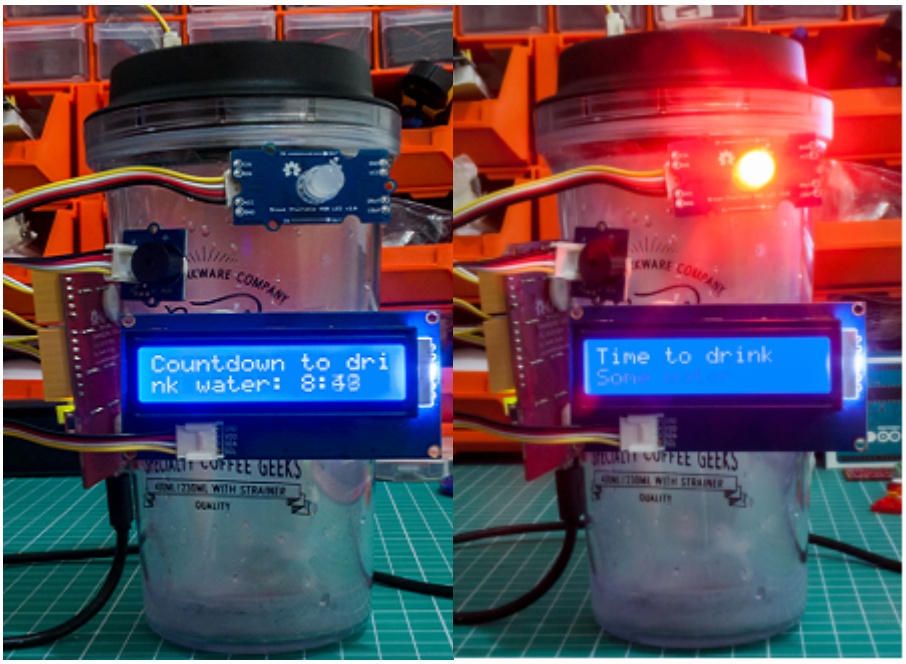
```
    */
    case 2:
      //stop buzzer
      noTone(buzzer);
      //update drinkTime
      drinkTime = now() + 1800;
      leds.setColorRGB(0, 0, 255, 0);
      //display message
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Well Done remind");
      lcd.setCursor(0, 1);
      lcd.print("you in 30mins");
      //wait for 5s for user to drink
      delay(5000);
      //reading accelerometer value
      Accel();
      //evaluate if the bottle is resting on flat
      if (r > 80 && r < 100 && p > 80 && p < 100) {
        //evaluate if the accelerometer calibrated
        if (!calibrated) {
          //calibrate accelerometer
          calibrateAccel();
        }
        else
        { //switch to mode 0
          mode = 0;
          //update drinkTime
          drinkTime = now() + 1800;
          leds.setColorRGB(0, 0, 0, 0);
        }
      }
      else
      { //if bottle is not resting on flat switch to mode 3
        mode = 3;
        leds.setColorRGB(0, 0, 0, 0);
      }
      break;
    /*case 3
       if the bottle is not resting on flat surface,
       display message with scrolling "plaase put
       down water bottle when finished!", then check
       if the bottle is resting still, if so, recalibrate
       accelerometer and once recalibrated switch back to
       case 0 and reset drink time to 30mins ahead
    */
    case 3:
      //update drinkTime
      drinkTime = now() + 1800;

      leds.setColorRGB(0, 0, 0, 255);
      //display message with autoscroll
      lcd.clear();
      lcd.setCursor(0, 0);
```

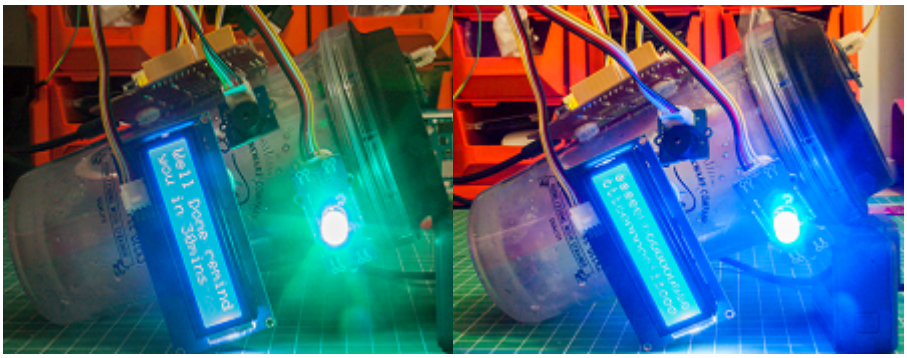
```
        lcd.print("Please put down water");
        lcd.setCursor(0, 1);
        lcd.print("bottle when finished!");
        for (int positionCounter = 0; positionCounter < 5; positionCounter++) {
          // scroll one position left:
          lcd.scrollDisplayLeft();
          // wait a bit:
          delay(200);
        }
        for (int positionCounter = 0; positionCounter < 5; positionCounter++) {
          // scroll one position right:
          lcd.scrollDisplayRight();
          // wait a bit:
          delay(200);
        }
        for (int positionCounter = 0; positionCounter < 5; positionCounter++) {
          // scroll one position left:
          lcd.scrollDisplayLeft();
          // wait a bit:
          delay(200);
        }

        //reading accelerometer value
        Accel();
        //evaluate if the bottle is resting on flat
        if (r > 80 && r < 100 && p > 80 && p < 100) {
          //evaluate if the accelerometer calibrated
          if (!calibrated) {
            //calibrate accelerometer
            calibrateAccel();
          }
          else
          { //switch to mode 0
            mode = 0;
            //update drinkTime
            drinkTime = now() + 1800;
            leds.setColorRGB(0, 0, 0, 0);
          }
        }
        break;
    }
    delay(1);
}
```

Step 2: Upload code into Seeeduino Lotus

Step 3: Observe result

The 4 states of the smart cup





REFERENCE

Aircraft principal axes. Accessed November 27, 2018. https://en.wikipedia.org/wiki/Aircraft_principal_axes.

APPENDIX

All the coding in this document is Available on the Github.

TTECH SUPPORT

Please do not hesitate to submit the issue into our forum or drop mail to techsupport@seeed.cc.