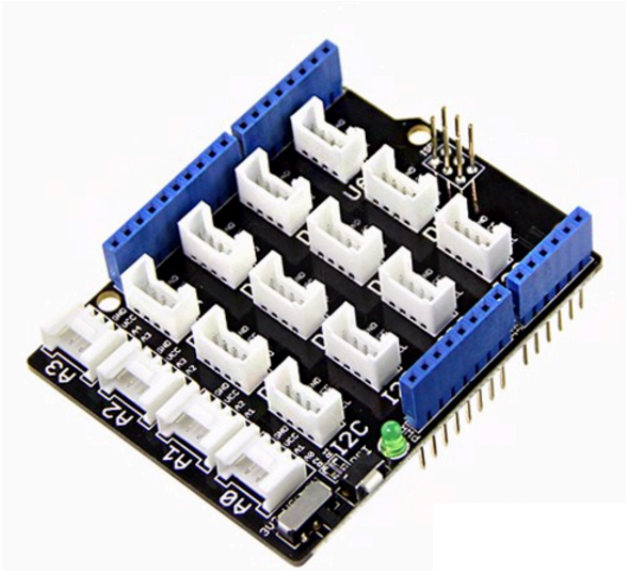
ARTIK IF board II

Arduino shields are modular circuit boards that can be plugged onto your Arduino boards or compatible base boards for extra functionality. With the new ARTIK IF board II, you can leverage the Arduino shields' capabilities in your project.

1. Grove Base Shield v2

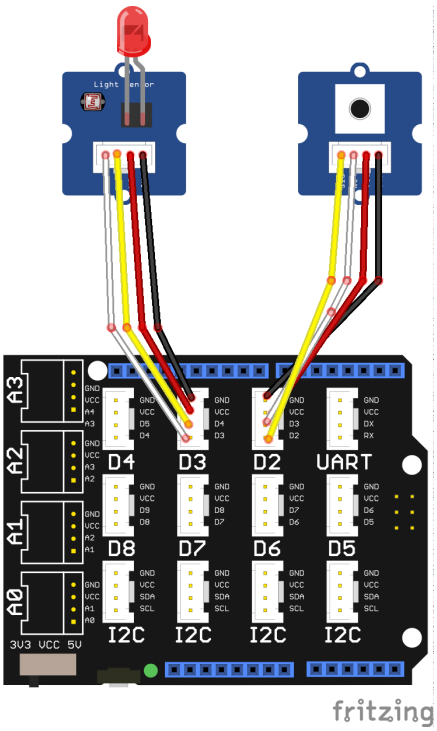
On ARTIK 530/710, you can attach a Grove Base Shield to your ARTIK IF board II. This will bring you easy-to-use Grove connectors including interfaces like analog inputs, I2C, digital I/Os and UART. Even if your project requires a lot of sensors, you don't have to mess up with bread board or jump wires. An integrated switch on the board allows you to select the working voltage at either 5V or 3.3V.



On ARTIK 530/710, you can use Arduino sketch to program the sensors. Arduino IDE setup and libArduino installation guide can be found here.

A Digital Grove Module Example

Here is an example of using button state to control the LED. Connect a Grove button to D2, and a Grove LED to D3.




```
int buttonPin = 2;
int LEDPin = 3;

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

// the loop function runs over and over again forever
void loop() {
  int state = digitalRead(buttonPin);
  digitalWrite(LEDPin, state);    // turn the LED on or off
  delay(1000);                    // wait for a second
}
```

An I2C Grove Module Example

To play with the I2C modules, I plugged a Grove 3-axis digital accelerometer into an I2C port on the base shield. Please follow the instruction here to install the additional libraries.

Once you have the library installed, navigate to Arduino IDE-> File -> Examples-> Accelerometer_MMA7660FC-master->MMA7660FC_Demo, change *Serial.print* in the

sketch to *DebugSerial.print* in order to show the debug output on the serial console, then compile it for ARTIK 530/710 platform.

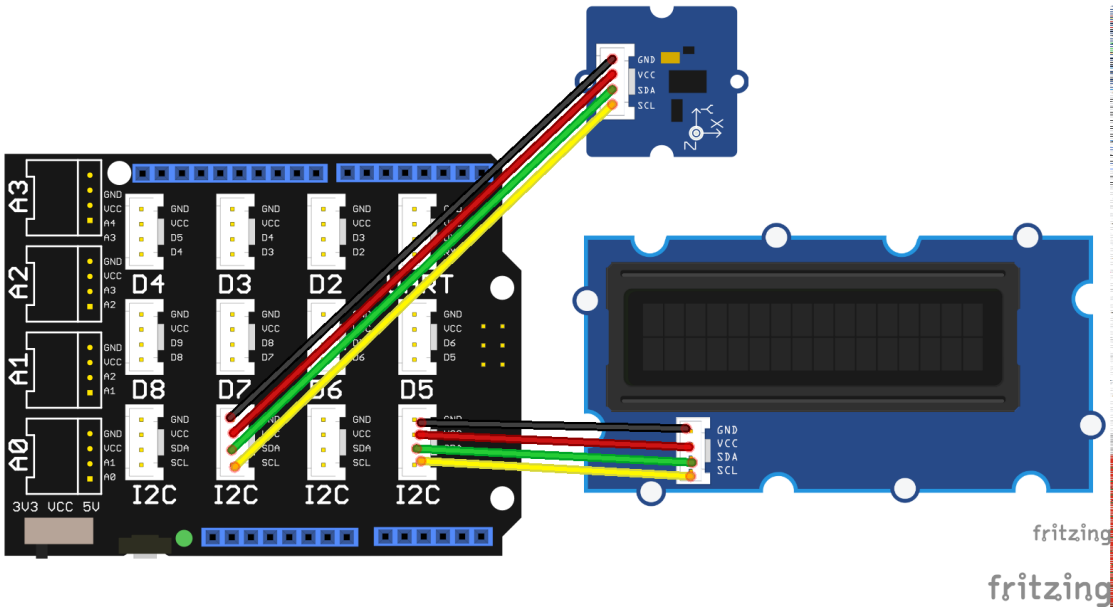
Run the compiled sketch, and you should be able to see x, y, z axis output from the accelerometer. Change the angle of your accelerometer will result in different outputs. The outputs consist of two parts: raw data and 3-axis acceleration info converted into the unit of gravity, “g”.

```
*************
x = 53
y = 61
z = 16
accleration of X/Y/Z:
2.52 g
2.90 g
0.76 g
*************
x = 0
y = 58
z = 45
accleration of X/Y/Z:
0.00 g
2.76 g
2.14 g
*************
x = 11
y = 52
z = 49
accleration of X/Y/Z:
0.52 g
2.48 g
2.33 g
*************
x = 47
y = 9
z = 7
accleration of X/Y/Z:
2.24 g
0.43 g
0.33 g
*************
```

To make our project a little more fun, we will use the accelerometer raw data to calculate the tilt angle between the surface of the accelerometer and the world XY plane, then display the angle on an I2C Grove LCD module.

I used the Grove –LCD RGB BackLight for the display part. First, follow the steps on Seeed wiki to install the Arduino library for the LCD module. Plug the LCD module into another I2C pin, and the sketch below basically uses MMA760 accelerometer to collect X, Y, Z data every 3 seconds, convert the raw data to a tilt angle and display it on the LCD module.

Tips: When you use LCD RGB BackLight, please turn the switch on your Base Shield to 5V position.

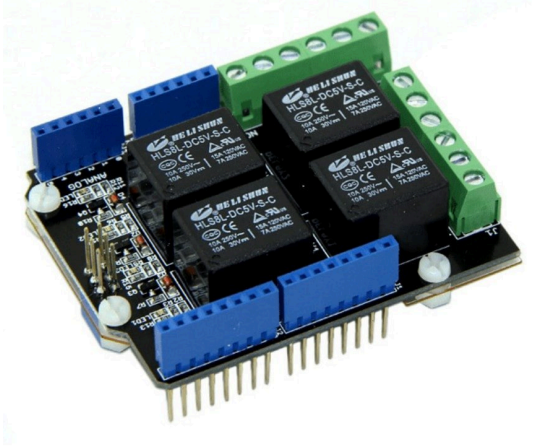


Here is our final sketch,

```
#include <Wire.h>
#include <math.h>
#include "MMA7660.h"
#include "rgb_lcd.h"

MMA7660 accelemeter;
rgb_lcd lcd;
const int colorR = 0;
const int colorG = 0;
const int colorB = 255;

void setup()
{
      accelemeter.init();
      Serial.begin(115200);
      lcd.begin(16, 2);
      //Set LCD background to blue
      lcd.setRGB(colorR, colorG, colorB);

      delay(1000);
```

```
}

void loop()
{
      int8_t x;
      int8_t y;
      int8_t z;
      float ax,ay,az;
      accelemeter.getXYZ(&x,&y,&z);

      double normal = sqrt(x * x + y * y + z * z);
      double z_normal = z / normal;

      //Calculate the inclination
      double angle = acos(z_normal) * 180 / M_PI;
      DebugSerial.print(angle);

      lcd.setCursor(0, 1);
      lcd.print("tilt=");
      lcd.print(angle);
      lcd.print((char)223);

      delay(3000);
}
```

2. **Relay Shield v3.0**



Thinking about to control a high voltage device by using ARTIK? Like turning on your coffee machine over the Internet? You can do this by using the relay shield. Relays are a key component in a lot of home projects, because they allow you to turn on and off higher voltage circuits (Of course, you can use relays with low power circuits too).

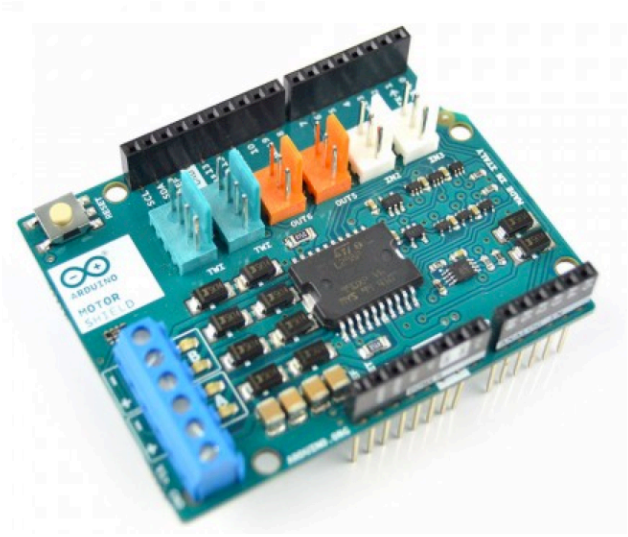
Seeed Studio Relay Shield v3.0 packs 4 sets of relays onto a single board. Please refer to Seeed wiki page for more technical details about this shield.

Be careful when you handle the circuits, as this is like leaving the cap off an electrical outlet.

The 4 Relay Shield packs 4 sets of relays onto a single shield. Arduino digital pins 4-7 are used for relays 4, 3, 2, 1 respectively. The sketch below simply turns on and off the relays one by one every other second.

```
unsigned char relayPin[4] = {4,5,6,7};

void setup()
{
  int i;
  for(i = 0; i < 4; i++)
  {
    pinMode(relayPin[i],OUTPUT);
  }
}

void loop()
{
  int i=0;
  for(i = 0; i < 4; i++)
  {
    digitalWrite(relayPin[i],HIGH);
  }
  delay(1000);

  for(i = 0; i < 4; i++)
  {
    digitalWrite(relayPin[i],LOW);
  }
  delay(1000);

}
```

3. Arduino Motor Shield v3.0



The Arduino Motor Shield allows you to control motor direction and speed. The shield has two channels, which enables it to control two DC motors or a stepper motor.

You can use the pins on your ARTIK to select a motor channel, specify a motor direction, set motor speed and start or stop the motor.

Pin breakdown is as follows:

Function	Channel A	Channel B
Direction	Digital 12	Digital 13
Speed (PWM)	Digital 3	Digital 11
Braking	Digital 9	Digital 8

As indicated in the table, we can use Pin 12(Channel A) and Pin 13(Channel B) to control motor's direction. To drive the motor forward, these pins need to be brought High. To drive the motor backward, these pins need to be driven Low.

The speed of motors can be controlled by Pin 3 (Channel A) and Pin 11(Channel B) using PWM signals. The speed needs to be defined as a number between 0 to 255, with 255 for full speed.

You can also apply a brake to the motors. The brake is controlled by Pin 8 (Channel A) and Pin 9(Channel B). Setting the pins to LOW disengages the brakes, or setting them to HIGH engages the brakes.

Another feature of the Motor shield is its capability to determine the amount of current the motor is drawing.

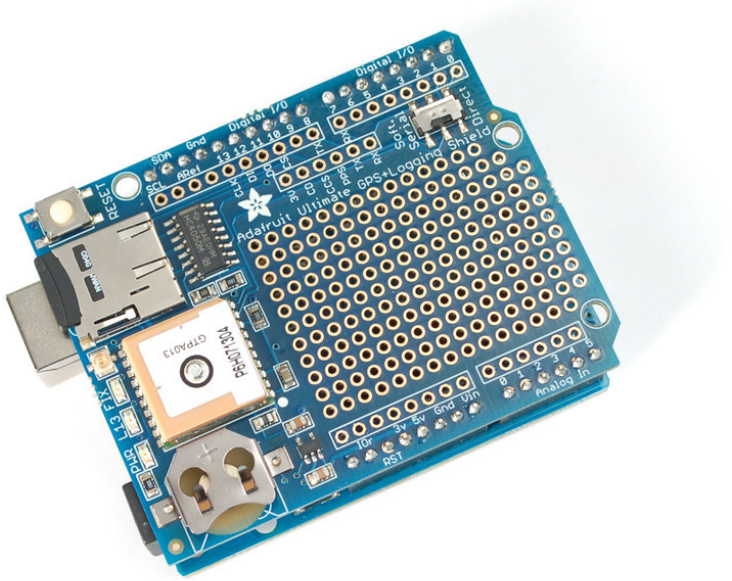
Here is a sketch to spin the motor on Channel B at full speed for 5 seconds:

```
void setup() {
 //Set up Channel B
  pinMode(13, OUTPUT); //Initiates Direction pin for Channel B
  pinMode(8, OUTPUT); //Initiates Brake pin for Channel B
}

void loop(){
  digitalWrite(13, HIGH); //Sets Forward direction for Channel B
  digitalWrite(8, LOW);    //Disengages Brake for Channel B
  analogWrite(11, 255);//Spins the motor on Channel B at full speed
  delay(5000);

  digitalWrite(8, HIGH); //Engages Brake for Channel B
  delay(1000);
}
```

4. [Adafruit Ultimate GPS Logger Shield]

Thinking about creating geocaching/geofencing projects? I chose Adafruit Ultimate GPS Logger Shield as it provides Direct Connection ability.

The shield doesn't have pre-assembled headers, which gives you the flexibility to choose between 0.1" male headers or Arduino Shield stacking headers. By following the instruction, I am able to assemble my shield quickly.



The GPS Logger Shield supports Direct Connect and Soft Serial Connect modes, and we will use the Direct Connect mode for the project.

Direct Connection with Jumpers on ARTIK 530/710

To make the GPS Logger Shield be compatible with ARTIK 530/710, we need to enable direct connection with jumpers. It is a trick to connect the output of the GPS serial TTL UART directly to the usb-serial converter chip on an ARTIK. Connect a wire from the TX pad to digital 0 and a wire from the RX pad to digital 1, as shown in the photo below.

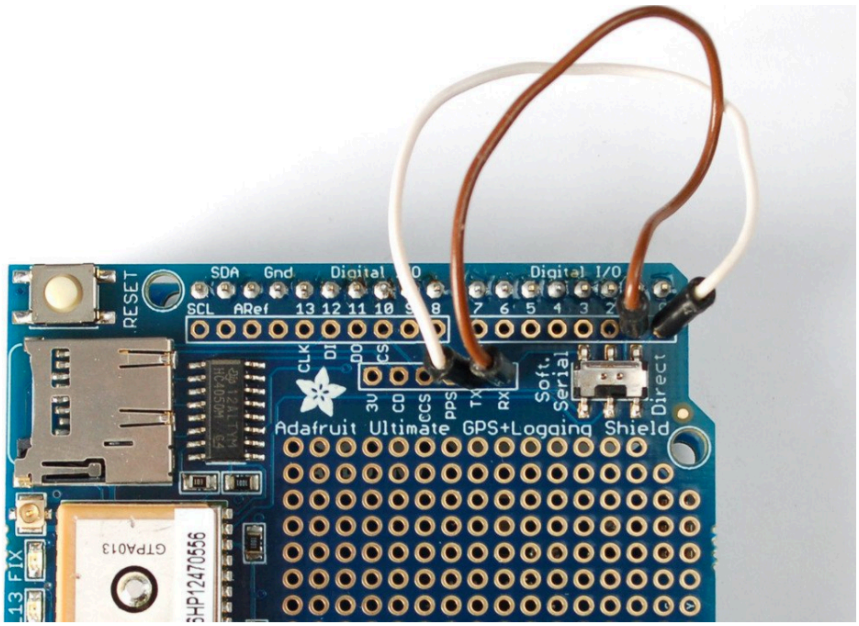


Photo taken from https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield/direct-connect

Library and examples of the shield are available on Github. Once you have them installed, try the sketch below

```
#include <Adafruit_GPS.h>
HardwareSerial* mySerial = &Serial1;
Adafruit_GPS GPS(mySerial);

void setup()
{
  DebugSerial.begin(115200);
  DebugSerial.println("Adafruit GPS library basic test!");
  GPS.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);//1Hzupdate rate
  GPS.sendCommand(PGCMD_ANTENNA);
  delay(1000);
  mySerial->println(PMTK_Q_RELEASE);
}

void loop()
```

```
{
   GPS.read();
   if (GPS.newNMEAreceived()) {
      if (!GPS.parse(GPS.lastNMEA()))
         return;
   }

   if (GPS.fix) {
      DebugSerial.print("Location): ");
      DebugSerial.print(GPS.latitudeDegrees, 4);
      DebugSerial.print(", ");
      DebugSerial.println(GPS.longitudeDegrees, 4);

      DebugSerial.print("Speed: ");
     DebugSerial.println(GPS.speed);
      DebugSerial.print("Angle: ");
     DebugSerial.println(GPS.angle);
      DebugSerial.print("Altitude: ");
     DebugSerial.println(GPS.altitude);
   }
}
```

Run the compiled sketch and you should be able to see the location output.