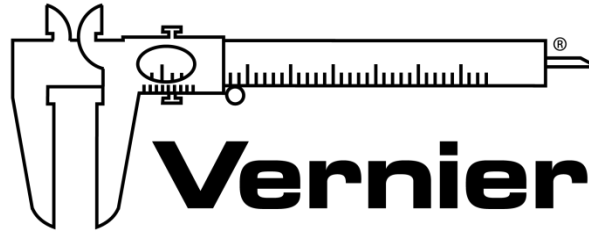


DIY Data Logging with Arduino



MEASURE. ANALYZE. LEARN.™

Vernier Software & Technology
www.vernier.com
888.837.6437

Josh Ence
jence@vernier.com

Dave Vernier
dvernier@vernier.com

NSTA National 2019
St. Louis, MO

DIY Data Logging With Arduino

NSTA 2019, St. Louis, MO



Sparkfun Redboard Arduino
(order code ARD-RED)



Vernier Interface Shield
(order code BT-ARD)

Lesson 1: Introduction to Arduino

- What is Arduino?
 - Shields, and the Vernier Interface Shield
 - Arduino website: www.arduino.cc
- Arduino programming, boards and ports
 - Terminology
 - Integrated Development Environment (IDE)
 - Sketch
 - Processing code
 - The "setup" and "loop" program structure
 - Compiling and uploading,
- How do Vernier sensors work with the Arduino?
 - Vernier website resources: www.vernier.com/engineering/arduino
 - Analog and digital signals
 - The VernierLib Library

Activity 1: Creating and modifying a simple sketch

Work in small groups to start taking data using a Dual-Range Force Sensor with the Arduino.

1. Connect the Vernier Interface Shield to the Arduino. Be careful in lining up the pins.
2. Connect a Dual-Range Force Sensor to the Analog 1 connector on the shield.
3. Set the range switch on the sensor to ± 10 N.
4. Open the Arduino Integrated Development Environment software. Then, open the AnalogReadSerial sketch by going to the File menu and choosing Examples/Basic/AnalogReadSerial.

Except for some of the comments, the entire sketch is listed below:


```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {

  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);

  // print out the value you read:
  Serial.println(sensorValue);

  // delay in between reads for stability
  delay(1);
}
```

5. Try out the AnalogReadSerial sketch by clicking on the  icon on the tool bar at the top of the screen. If everything is working properly, you should get messages at the bottom of the screen that the sketch has compiled, and has been uploaded, and the program should run.
6. To see if the sketch is working, click on "magnifying glass" icon at the top right of the toolbar or selecting Serial Monitor from the Tools menu. This should open a new window displaying the sensorValue. Notice how the sensorValue changes when you push and pull on the Force Sensor. If this is not working, let one of the workshop helpers know.
7. The Arduino IDE also has the ability to graph this data. Close the Serial Monitor window and choose "Serial Plotter" from the Tools menu to open a graph of the sensorValue. Try experimenting with the force sensor.
8. Modify the AnalogReadSerial sketch to slow it down. As written, it takes data very fast. This is controlled by the delay statement at the end of the sketch. The delay is in milliseconds and it is originally set to only 1 millisecond. Modify the sketch so it takes a reading every half second.

Lesson 2: Reading Calibrated Sensor Values with Arduino?

As you saw in the first activity, Arduino microcontrollers have the ability to read an analog voltage in the range of 0 to 5 volts and display the result as a number. When you experiment with the Force Sensor in Activity 1, you may have noticed that the sensorValue displayed on the Serial Monitor was around 500 with no force and increased when pushed on the sensor and decreased when you pulled on the sensor. It was always in the range of 0 to 1023.

Most Vernier analog sensors with BTA cables produce a voltage output between 0-5 V, so the Arduino can read them nicely. We have a Vernier library that handles the calibration of these sensors and makes these sensors very easy to use.

Here is a very simple sample sketch using the VernierLib library. This sketch is in our examples folder and is named VernierLibDemoSimple.

```
#include "VernierLib.h"
VernierLib Vernier;

float sensorReading;

void setup(void)
{
    Serial.begin(9600);
    Vernier.autoID();           // this is the routine to do the
                               autoID
}

void loop()
{
    sensorReading=Vernier.readSensor();
    Serial.print(sensorReading);
    Serial.print(" ");
    Serial.println(Vernier.sensorUnits());
    delay(500);                //half a second
}
```

Activity 2 - Try out the Sketch to Read Force

1. Open the VernierLibDemoSimple example by choosing File > Examples > VernierLib > VernierLibDemoSimple
2. Try it out by clicking on the ➡ icon. If your Force Sensor is plugged in, you should see calibrated Force readings displayed in the Serial Monitor window.
3. Check the calibration of your system by using known masses.
4. Close the Serial Monitor window and open the Serial Plotter by choosing it from the Tools menu. Again, try out the Force Sensor.
5. Try switching the Range switch on the Force Sensor and then pressing the Reset button on the Arduino board to restart the program. How is the calibration?
6. Try a different BTA sensor such as a temperature sensor. How does the calibration seem?
7. Save this sketch on the desktop with a name you will recognize.

Extensions

If you finish early, here are two challenges you can try:

- Modify your sketch so that it takes data more quickly.
- Modify your sketch to turn on the LED on the Vernier shield, which is connected to Arduino digital pin 13, when the force is more than 5 Mewtons. You will need to add a statement like the one below in the loop section of the program:

```
if (sensorReading > 5) {  
    digitalWrite (13,HIGH);  
}
```

Lesson 3: Using the DCU in an Arduino sketch

The Vernier Digital Control Unit (DCU) can be connected to the Vernier Interface Shield and used to control DC electrical devices. One of the advantages of the DCU is that it can provide more power, for devices such as motors and lamps, than the Arduino alone can provide. With the DCU connected, you can control up to six different output lines.

We will use the Vernier library to simplify controlling the DCU with a line like this:

```
Vernier.DCU(X);
```

where X can be any number between 0 and 15. In this workshop, we will primarily use line D1 of the DCU and control it with the following Arduino code:

```
Vernier.DCU(1); //turns on line D1  
Vernier.DCU(0); //turns off line D1
```

The other lines of the DCU can be controlled in a similar way. To keep things simple, we recommend using only the first three DCU lines. You can control them and set them in any pattern using the following values for X. Notice that this is the basic binary counting pattern.

X	D1	D2	D3
0	Off	Off	Off
1	On	Off	Off
2	Off	On	Off
3	On	On	Off
4	Off	Off	On
5	On	Off	On
6	Off	On	On
7	On	On	On

Activity 3: Controlling things with the Arduino and the DCU

1. Connect the DCU into the Digital 2 port.
2. Wire the device (buzzer, lamp, etc) to DCU Line D1 and GND.
3. Modify your sketch you used in Activity 2 to turn on Line D1 when your sketch first runs and leave it on all the time. Remember that you can use these lines of code to control line D1:

```
Vernier.DCU(1); //turns on line D1  
Vernier.DCU(0); //turns off line D1
```

4. Remember the basic structure of all Arduino programs. Should your added lines be in the Setup or Loop part of the code?
5. Many interesting projects involve activating a device based on a sensor reading. To do this, you need to use an IF statement to compare the reading from the sensor, with a numeric value. For example, if you want the line to go on if (and only if) the Force is greater than 5 N, you would add this code:

```
if (sensorReading>5){  
    Vernier.DCU(1);  
} else {  
    Vernier.DCU(0);  
}
```

6. Add these lines to your sketch and give it a try. Would you add them in the setup or loop section of the code?
7. The Line D2 of the DCU can be controlled in a similar way with VernierLib library. Connect a second device and modify the sketch to control it. Can you make the second line go on if the force is greater than 7 N?

Lesson 4: Arduino demonstrations

In a 1.5-hour workshop, all we can do is introduce you to Arduino. We brought some other demonstrations to show the possibilities.

Activity 4: Try things!

Depending on your interests and time, you can:

- Use an Arduino sketch with a different sensor.
- Take a look at the other features of the VernierLib library. Read the pdf in the VernierLib folder.
- Experiment with one of our Arduino demonstrations.
- Construct and program your own project. We have lots of sensors and we have lots of buzzers, LEDs, lamps, and motors. Use supplies provided to develop your own Arduino engineering challenge.