

Wi-Fi-based Wireless Sensors for Data Acquisition

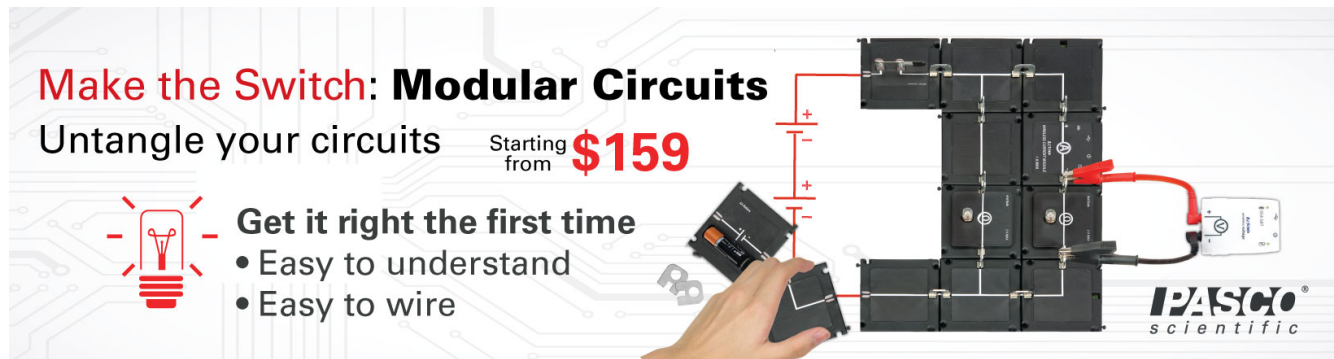
William C. Bensky

Citation: *The Physics Teacher* **56**, 393 (2018); doi: 10.1119/1.5051157

View online: <https://doi.org/10.1119/1.5051157>


View Table of Contents: <http://aapt.scitation.org/toc/pte/56/6>

Published by the [American Association of Physics Teachers](#)



Make the Switch: Modular Circuits

Untangle your circuits Starting from **\$159**

 **Get it right the first time**

- Easy to understand
- Easy to wire

PASCO
scientific

Wi-Fi-based Wireless Sensors for Data Acquisition

William C. Bensky, University of Southern California, Los Angeles, CA

Although mainly used by the hobby or “maker” crowd, the Arduino¹ microcontroller has made its way into all levels of education, including physics labs.² Perhaps the most compelling aspect of the Arduino in this regard is the ease with which it can be interfaced and acquire data from a sensor, using any one of its many analog inputs, SPI or I2C interfaces.³ In this paper we wish to introduce the reader to the Sparkfun “Thing,” which is shown in Fig. 1.⁴ The “Thing” is a board that is smaller and less expensive than an Arduino, while being very similar in its programming, analog input capabilities, and support for the SPI and I2C interfaces. But here’s the real uniqueness of this board: it creates its own Wi-Fi network, just like ones we’re all used to using (for example, by our home routers). In use, one simply connects to the Thing’s Wi-Fi network using a standard device (laptop, tablet, smartphone, Chromebook, etc.) and retrieves the sensor data using a web browser. Thus, no custom programming is required, and no special software is needed by the end user. Its small size and easy access allows for all kinds of embedded and/or portable sensor applications.

Below we will demonstrate a flexible Thing-based framework for creating a wireless sensor, whose output can be read using any standard Wi-Fi-capable device. Our methodology is all within the realm of the Arduino “ecosystem,” so it should be familiar to those readers with Arduino experience. We will illustrate the technique through the

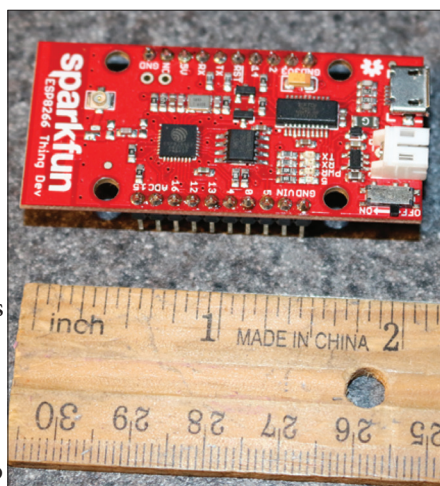


Fig. 1. The Sparkfun “Thing” development board.

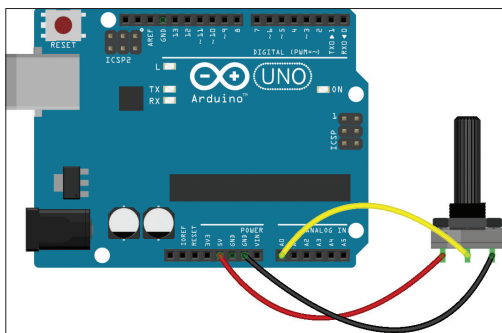


Fig. 2. A 10-k Ω potentiometer connected as a voltage divider to analog input “0” on the Arduino Uno.

development of a Wi-Fi-enabled gyroscope, and think the reader will be impressed with the ease with which a wireless sensor can be implemented.

Sensor “warm-up”

To those new to the Arduino ecosystem, we begin by illustrating a basic Arduino-based sensor. Suppose an Arduino Uno⁵ is connected to a 10-k Ω potentiometer⁶ as a voltage divider, as shown in Fig. 2.

We “power” the potentiometer by connecting its outer two leads to the Arduino’s regulated +5V and 0V supply, and the center lead to A0, which is “analog input zero (0)” on the Arduino. As the potentiometer shaft is turned, voltages between 0 and 5 V will appear on the center lead, and on the analog input of the Arduino. If the code in Fig. 3 is uploaded into the Arduino, the serial monitor can capture the analog values read (into variable x), and send them over the serial port (via the `Serial.println(x)` statement) to the host computer for display as shown in Fig. 4. Note the Arduino has a 10-bit analog to digital converter, with a 5V analog reference, so multiplying x by $5.0/1023$ would convert these values into actual voltages. Such numbers can naturally be exported for plotting, analysis, etc.⁷ As a sensor, the potentiometer could be used as a “smart pivot” for a physical pendulum for example. A similar voltage division can be made by replacing the potentiometer with a fixed resistor in series with another type of resistive sensor, such as a CdS photocell for measuring light,⁸ a thermistor for temperature,⁹ “flex sensor” for motion,¹⁰ or force sensor for force.¹¹

More sophisticated sensors that require more than a single analog output can interface to the Arduino using the SPI or I2C protocols,³ which are at their core serial protocols, but

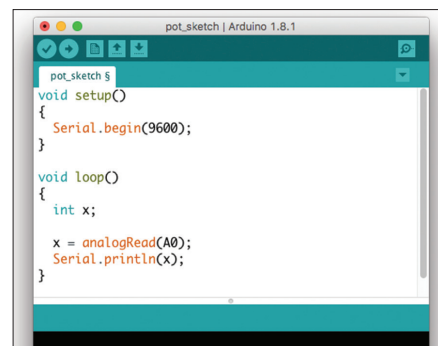


Fig. 3. Code that will read analog 0 and send the result to the serial port.

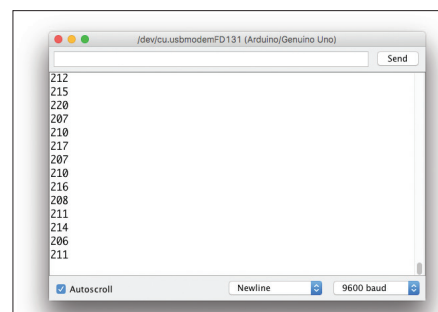


Fig. 4. Output of the code from Fig. 2, as seen on the host computer’s “Serial Monitor.” The numbers change as the potentiometer shaft in Fig. 1 is rotated.

allow for digital data packets (i.e., sensor readings) to be exchanged between the sensor and Arduino, and also allow for the connection of multiple sensors without using additional Arduino ports.¹² Such sensors often have readily available libraries and wiring diagrams, allowing one to use the sensors, without having to understand the SPI or I2C protocols.¹³

As development of one's sensor project may go, limitations will likely arise. The first is that the Arduino is typically tethered via a USB cable to a host computer, in order to receive power and to log sensor data. It is possible of course to run the Arduino on a battery or "wall wart" for power. For data logging, an SD (secure digital) card may be connected,¹⁴ but this removes the ability to see data arriving in real time, and one must always stop the data acquisition and/or have physical access to the Arduino in order to access the card. Ethernet shields are available for sending data out over a network, but this requires a substantial software suite on the server side to process incoming data.

Wireless sensors

Wireless sensors at the level of this work are not new. Here are three examples. The first is XBee technology, which works very well,^{15,16} but we found several drawbacks with it. In particular, the receiver must have custom software written to handle the incoming wireless data, including implementing an error-checking protocol. Also, although XBee boards will read and transmit the analog output from a sensor, there is no way to transmit the exact time the sample was taken. Lastly, XBee boards do not work directly with the I2C or SPI interface (eliminating the use of many sensors, unless an Arduino is added to the mix).

Second, there are Wi-Fi shields for the Arduino, but their cost has remained curiously high for many years now (~\$50), and it can be difficult to gain permission to use these on an enterprise Wi-Fi network (at a university, for example) for security reasons.

Third, there is Bluetooth, an option we have not embraced for a wireless sensor framework, primarily because in functionality, a typical microcontroller Bluetooth module merely replaces the usual computer-to-microcontroller (USB) cable. The end result is a (somewhat cumbersome) wireless sensor package requiring the sensor, Bluetooth module, and microcontroller on the transmitting side. On the receiving side, unless custom (Windows or macOS) software is written, a second microcontroller and Bluetooth module would be needed for data logging.

All told, we'd really prefer a tighter integration of the wireless link and the microcontroller functionality, hence the Sparkfun Thing.

Sparkfun Thing: Up and running

Dispensing then with the lackluster wireless options, we now move on to the Thing. To begin, we present a simple analog input for testing, similar to the potentiometer experiment from above, except we'll use a TMP36 thermometer,¹⁷ as

shown in Fig. 5. We note the ADC has a maximum input of 1 V (the TMP36 outputs a voltage in millivolts, that is the temperature in degrees Fahrenheit). Next, we power the Thing by plugging a USB cable between the board and host computer.

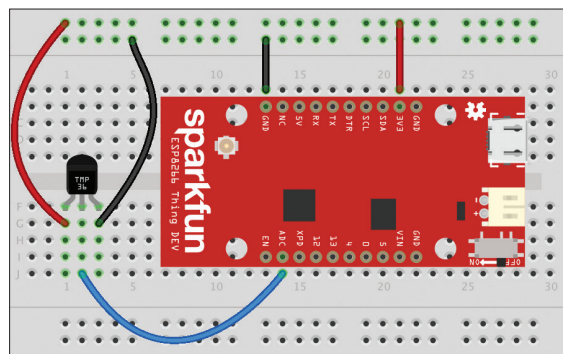


Fig. 5. The Sparkfun Thing wired to a TMP36 temperature sensor.

```
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <SPI.h>
#include <SparkFunLSM9DS1.h>

WiFiServer server(80);

void setup() {
  WiFi.mode(WIFI_AP);
  WiFi.softAP("wifi-temp", "physics1");
  server.begin();
}

void loop() {
  WiFiClient client = server.available();
  if (!client)
    return;

  String s = "HTTP/1.1 200 OK\r\n";
  s += "Content-Type: text/html\r\n\r\n";
  s += "<!DOCTYPE HTML>\r\n<html>\r\n";
  s += "The temperature is: ";
  s += String(analogRead(A0) * 100.0 / 1024.0);
  s += "</html>\n";

  client.print(s);
  client.flush();
}
```

Fig. 6. Complete code to read the temperature from the TMP36 and send it out over the Wi-Fi network called "wifi-temp" with password "physics1."

The Thing is programmed through the same Arduino software used above, but a software add-on tailored for this board must first be installed. We followed the directions in Ref. 18 without experiencing any problems. After selecting "Sparkfun ESP8266 Thing Dev" from the "Tools→Board" menu, and the appropriate serial port from the "Tools→Port" menu, we uploaded the 28 lines of code shown in Fig. 6 into the Thing. In this figure, the `WiFi.mode(WIFI_AP);` line is what sets the Thing into "router mode," as in broadcasting its own Wi-Fi network to which one may connect. The line `WiFi.softAP("wifi-temp", "physics1");` sets the network name (here "wifi-temp," short for "Wi-Fi temperature," with a password of "physics1.") (Note the password must be at least eight-characters in length.) Lastly, the `millis()` function allows us to timestamp the data samples.

After uploading the code, we used our iPhone to see what Wi-Fi networks were available and obtained the list in Fig. 7, where "wifi-temp" is indeed available ("longitude" is an actual Wi-Fi network). After connecting to this network, we used the web browser on the phone to go to the "site"

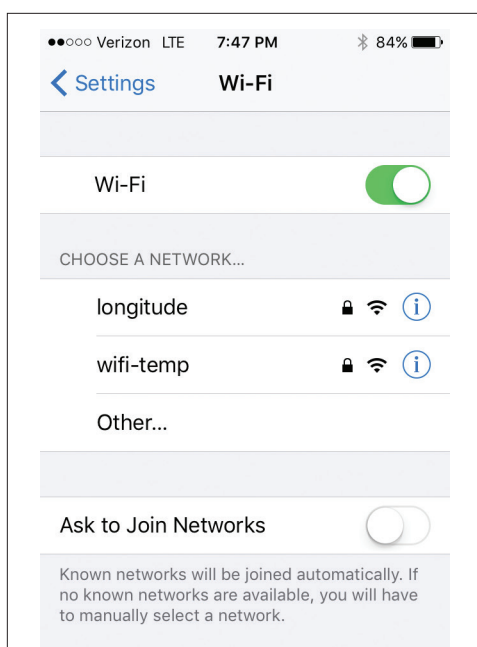


Fig. 7. Wi-Fi networks available when the Sparkfun Thing is running with the code from Fig. 6. We note the presence of a network called “wifi-temp.”

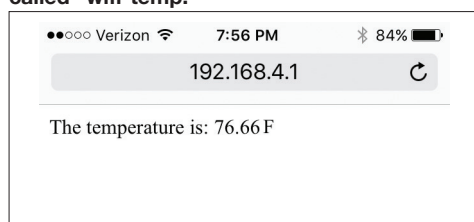


Fig. 8. Temperature read from the TMP36 sensor, in a webpage retrieved using the web browser on the iPhone.

`http://192.168.4.1`. Here we were greeted with the “page” that is the temperature read from the TMP36, shown in Fig. 8, as generated by the code in Fig. 6. We are able to take readings up to about 200 ft from the device, a range that is much longer than what Bluetooth would deliver. The maximum sample rate over the I2C bus seems to be about 200 Hz, which we determined by logging 1000 (time, sample) data points straight to the Thing’s internal memory. This rate appears to be an internal hardware limitation, having nothing to do with the Wi-Fi connectivity or connected receiving device.

Untethering the Thing from the host computer is a matter of powering it with a battery, keeping in mind that it can consume almost 200 mA. If using the Thing’s USB port, the “computer” end of the cable can be plugged into a portable battery pack used to charge phones, or into a standard (120- or 220-V) USB charger. For maximum portability, we recommend a lithium ion battery¹⁹ that will plug right into the white connector in Fig. 1. We also recommend the battery charger.²⁰

Lastly, we note two things about the URL used to communicate with the sensor, `http://192.168.4.1`. First, this is the default access IP address on the Thing’s

network, which may be changed using the `WiFi.config()` function. Second, this URL may be used to send commands to the Thing board, perhaps to change a sample rate, and/or start/stop acquisition, or even to toggle a remote relay on or off. For this, the access URL would resemble `http://192.168.4.1/relay/on`, parsed using the `lineClient.readStringUntil("\r");` in the Thing code.

Sparkfun Thing application: A wireless gyroscope

Our interest in a small portable wireless sensor was motivated by a project involving the measurement “the motion” of antiquarian turret clocks during a recent study abroad experience in London, England, for which the author was a teaching assistant, during the summer of 2017. That is, we wanted to attach a small gyroscope directly to a clock pendulum, or foliot bar, to capture a clock’s motion for further study. To this end, we outfitted the Thing with a “9dof” (nine degrees of freedom) board,²¹ which is an I2C sensor that reads angular speed, angular acceleration, and magnetic field, each along three perpendicular axes with 16 bits of resolution. The pins on the 9dof board fortuitously line up with pins GND, 3V3, 2, and 14 on the Thing (pins 2 and 14 are the dedicated I2C

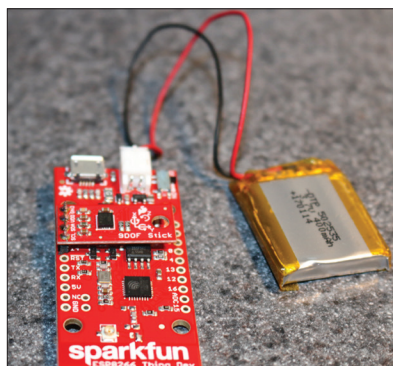


Fig. 9. Final assembly of our Thing-based gyroscope, with the lithium ion battery from Ref. 19. Note the “9dof” board is soldered directly onto the Thing board.

pins); thus, the 9dof board was soldered directly onto the Thing as shown in Fig. 9. This figure also shows the aforementioned lithium ion battery. We used a 3-D printer to print a small plastic case for the components, with a magnet embedded for quick attachment to the iron clock parts. Code to run the whole thing, as well as STL files for 3-D printing, is avail-

able by contacting the author. The total mass of the entire assembly is 26 g.

The device was tested by attaching it to three pendulums of different lengths in the lab (lengths of 1.3 m, 0.96 m, and 0.61 m). Data obtained for the 1.3-m pendulum are shown in Fig. 10, which shows the angular speed (in degrees/second, to $\pm 3\%$.) vs. time as the pendulum oscillated. To determine the period T , we found the time difference between each adjacent maxima and averaged them. We plot T^2 vs. length for all three pendulums in Fig. 11. The slope of the line fit is $3.96 (\pm 0.11) \text{ s}^2/\text{m}$, giving a value of $g = 9.95 (\pm 0.29) \text{ m/s}^2$. We are thus confident in the physical correctness of the gyroscope readings and in the overall wireless data throughput.

Lastly, before embarking on our clock-measuring work, we

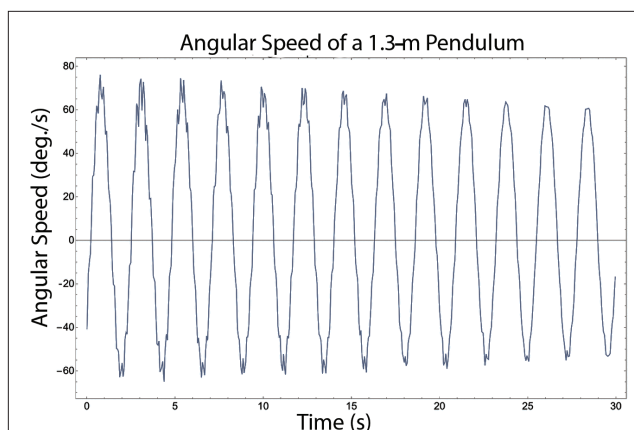


Fig. 10. Data taken when the wireless gyroscope in Fig. 9 was attached to a 1.3-m long pendulum.

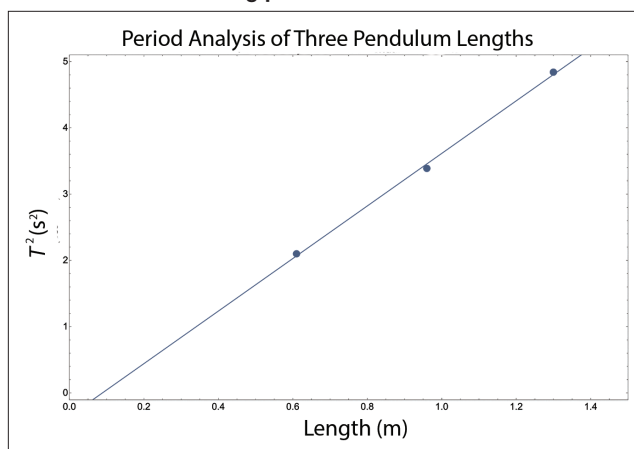


Fig. 11. Analysis of three pendulums. The slope of the line is $3.97 \text{ s}^2/\text{m}$, giving a value of $g = 9.95 \text{ m/s}^2$.

could not resist testing the extreme portability of the device. To this end, we inserted the entire unit into a foam football, as shown in Fig. 12, and proceeded to drop it from the top of an 11-m (36-ft) high building. We note that the MEMS, cantilever-based accelerometer²² is really measuring a weight-to-net-force ratio, meaning an acceleration of 1.0 ($\pm 4\%$) is measured along an axis perpendicular to the ground, when the sensor is at rest. An acceleration of zero (or $0g$) is reported when the sensor is in free fall. In Fig. 13, we plot the net acceleration of the sensor vs. time. The net acceleration was computed via $a = (a_x^2 + a_y^2 + a_z^2)^{1/2}$, where the three components of a come directly from the sensor data. In Fig. 13, we note six regions of interest as follows. Region A is for the ball at rest at the top of the building, B is when it is in free fall for 1.5 s (the acceleration gets “small,” approaching $0.1g$ or so), C is a large acceleration upon hitting the ground, D is a 1.0-s free fall (again, small acceleration) after bouncing from the ground, E is the ball hitting and rolling on the ground after the bounce, and F is when the ball is at rest on the ground. We hoped for better $0g$ regions, but note this type of accelerometer (the cantilever) comes with inherent random noise and an offset, and is really best suited as a slow-motion orientation sensor, mostly used in smartphones and game controllers. In the inset of Fig. 13, we show a y-axis calibrated into units of accel-



Fig. 12. The Wi-Fi gyroscope inserted into a foam football.

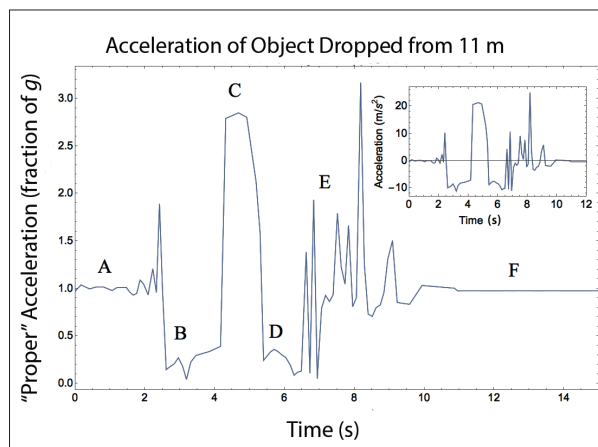


Fig. 13. Acceleration (fraction of g) of the ball in Fig. 12 when dropped from an 11-m building. A) at rest at top of building, B) in free fall for 1.5 s (near $0g$), C) acceleration upon hitting the ground, D) 1.0-s free fall after bounce from the ground, E) hitting and rolling on ground after bounce, F) at rest on ground. Inset: calibrated version.

eration. Our calibration mapped 1.0 from the accelerometer to 0.0 m/s^2 , 0.156 from the accelerometer to -9.8 m/s^2 , and 0.60 from the accelerometer into -4.9 m/s^2 (as measured by placing the device in an Atwood machine, with the counter mass at 1/3 of the device mass). The calibration showed a linear dependence of output from the device into m/s^2 , and was used to scale the y-axis of the inset of Fig. 13.

Conclusions

We have demonstrated a hardware and software framework for constructing a highly portable wireless sensor that uses standard Wi-Fi technology to output sensor data. This results in a sensor whose values can be retrieved using a web browser (or any TCP/IP-based communication) on any Wi-Fi-capable device. This largely eliminates the need for custom software development to communicate with the sensor, including the end user, who will not require any special software downloads or installs. For our needs, a wireless gyroscope was constructed for less than \$50, which was then tested by attaching it to a pendulum and determining a value of g . We also tested the ability of the device to measure ac-

celeration by embedding it into an object in various stages of free fall.

Although the Wi-Fi network created by the Thing board is a bona fide wireless network, it is 100% contained to the board itself, meaning it is completely isolated from any larger (enterprise) network. Thus is it “secure” in that it is not associated with an actual data network. For this reason, the board may find interesting uses in outreach around a school or university. For example, the board may be located in a display case, allowing passersby to connect to it with their mobile device, perhaps to read a sensor attached to a larger (research) apparatus to highlight ongoing work in a laboratory, to activate something in the display (an LED perhaps), or take part in a survey of some sort.

References

1. See Arduino, <http://www.arduino.cc>.
2. There are too many papers to list, but here are a few: F. Bouquet and J. Bobroff, “Project-based physics labs using low-cost open-source hardware,” *Am. J. Phys.* **85**, 216 (Feb. 2017); Jed Brody and Max Brown, “Transient heat conduction in a heat fin,” *Am. J. Phys.* **85**, 582 (July 2017); Jen-Feng Hsua, Shonali Dhingra, and Brian D’Ursob, “Design and construction of a cost-efficient Arduino-based mirror galvanometer system for scanning optical microscopy,” *Am. J. Phys.* **85**, 68 (Dec. 2016); Mike McCaughey, “An Arduino-based magnetometer,” *Phys. Teach.* **55**, 274 (April 2017); Calin Galeriu, “An Arduino-controlled photogate,” *Phys. Teach.* **51**, 156 (Feb. 2013); Daniel Nichols, “Arduino-based data acquisition into Excel, LabVIEW, and MATLAB,” *Phys. Teach.* **55**, 226–227 (April 2017).
3. See Serial Peripheral Interface Bus, Wikipedia, https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus; I2C, Wikipedia, <https://en.wikipedia.org/wiki/I2C>; and I2C, Sparkfun, <https://learn.sparkfun.com/tutorials/i2c>.
4. Sparkfun ESP8266 Thing, <https://www.sparkfun.com/products/13231>.
5. For Arduino Uno purchasing considerations, see <https://www.sparkfun.com/products/11021>.
6. 10k Audio Taper Pot, 6 mm shaft, <http://www.allelectronics.com/item/natp-10k/10k-audio-taper-pot-6mm-shaft/1.html>.
7. We note that version 1.8.1 and higher of the Arduino software has a “Serial plotter” built in as well.
8. Mini photocell, <https://www.sparkfun.com/products/9088>.
9. Thermistor (10k), <https://www.sparkfun.com/products/250>.
10. Flex sensor, <https://www.sparkfun.com/products/8606>.
11. Force sensitive resistor, <https://www.sparkfun.com/products/9376>.
12. See Patrick McDougall and Eric Ayars, “Two dimensional heat flow apparatus,” *Am. J. Phys.* **82**, 620–623 (June 2014) for a two-dimensional array of temperature probes all connected to a single Arduino.
13. See the “Arduino Library” and “Hookup Guide” links here: <https://www.sparkfun.com/products/13314>.
14. See SD Library, Arduino, <https://www.arduino.cc/en/Reference/SD>.
15. See XBee Buying Guide, SparkFun, https://www.sparkfun.com/pages/xbee_guide.
16. Eric Ayars and Estella Lai “Using XBee transducers for wireless data collection,” *Am. J. Phys.* **78**, 778–781 (July 2010).
17. Temperature Sensor - TMP 36, <https://www.sparkfun.com/products/10988>.
18. ESP8266 Thing Hookup Guide, <https://learn.sparkfun.com/tutorials/esp8266-thing-hookup-guide/installing-the-esp8266-arduino-addon>.
19. Lithium Ion Battery - 400 mAh, <https://www.sparkfun.com/products/13851>.
20. Sparkfun LiPo Charger Basic - Micro-USB, <https://www.sparkfun.com/products/10217>.
21. Sparkfun 9DoF Sensor Stick, <https://www.sparkfun.com/products/13944>.
22. See iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D magnetometer, <http://www.st.com/content/ccc/resource/technical/document/datasheet/1e/3f/2a/d6/25/eb/48/46/DM00103319.pdf/files/DM00103319.pdf/jcr:content/translations/en.DM00103319.pdf>.

Will Bensky is an undergraduate aerospace engineering student. He is interested in sensors and reusable rockets, and is currently working on long-range UAVs at the NASA Jet Propulsion Laboratory.
wbensky@gmail.com

**Concerned about
the safety of your students?**

**Safety
in Physics Education**

**Best
Seller!**



Promote safety awareness and encourage safe habits with this essential manual. Appropriate for elementary to advanced undergraduate laboratories.

Members \$9.50 • Nonmembers \$11.50
order online: www.aapt.org/store or **call:** 301-209-3333