

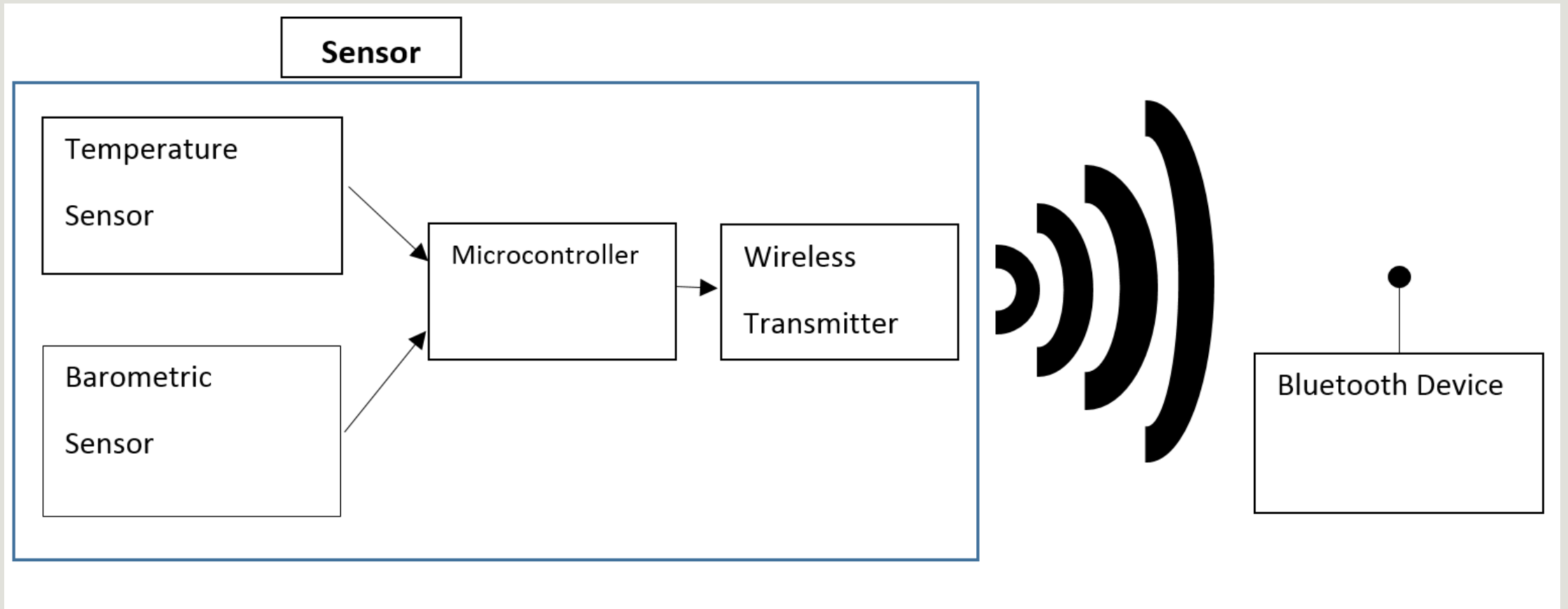
Bluetooth Temperature and Barometric Pressure Sensor

BY: TOM VANHOUDT

Initial Ideas and Goals

- The project started because of an interest in Bluetooth technology.
- Initial project designs:
 - Wireless, via Bluetooth communication
 - Sensor would be portable and battery powered
 - Temperature readings accurate to 0.5 degrees Celsius.
 - Pressure readings accurate to 0.02 millibars (1 bar = 750.06 mmHg)

Initial Design



Main Components

The three main components I used where:

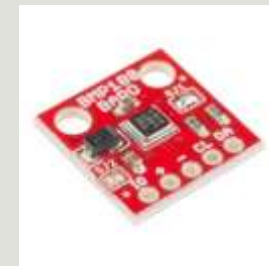
- Bluetooth Mate Silver
- Barometric Pressure Sensor – BMP180
- MSP430 Launchpad

Part	Cost
Bluetooth Mate Silver	\$24.95
Barometric Pressure Sensor – BMP180	\$9.95
MSP430 Launchpad	\$9.99

Bluetooth Mate Silver
Roving Networks



Barometric Pressure Sensor – BMP180
Bosch Sensortech



MSP430 Launchpad
Texas Instruments

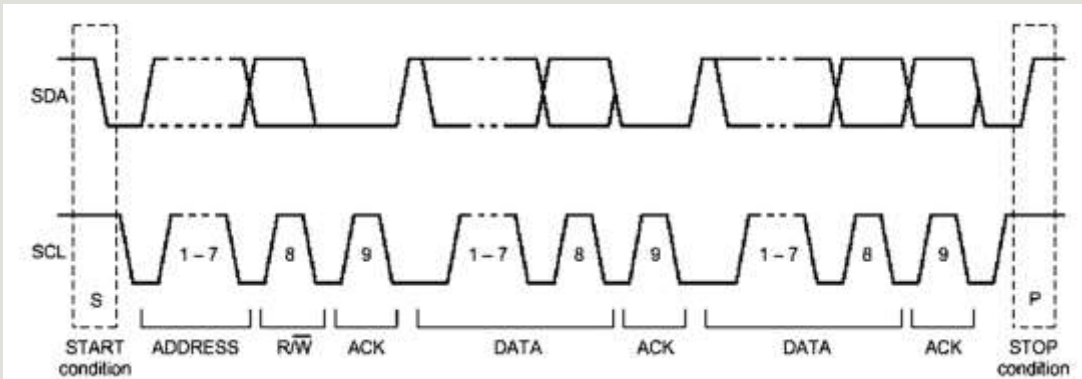


Steps Toward the Finished Project

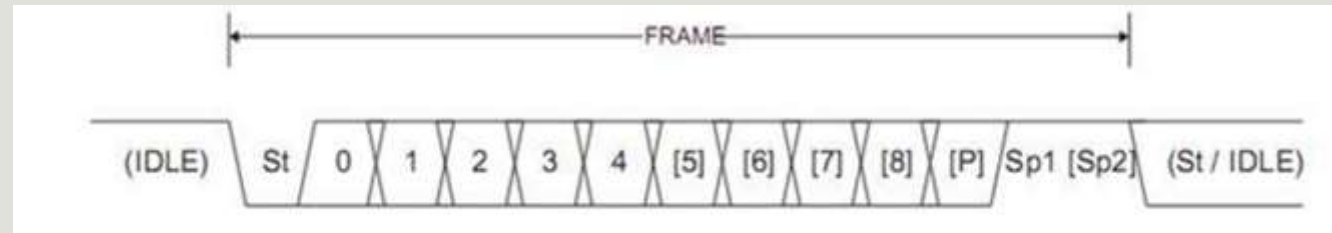
1. Determine what forms of communication to use between the sensor, microcontroller, and Bluetooth module.
2. Create code for microcontroller to grab data from the sensor and ship that data out to the Bluetooth Module.
3. Setup Bluetooth Module to be a pipeline that receives data, and immediately transmits that data.
4. Design printed circuit board, solder parts to board, and test.

Communication

I²C Protocol

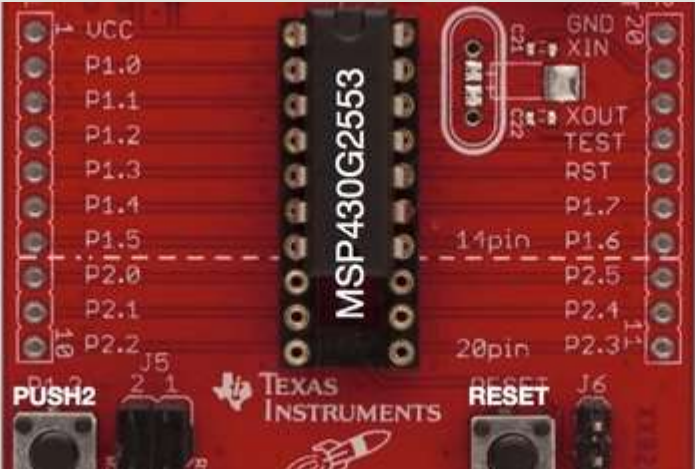


UART Serial Communication



MSP430 Pin Connections

+3.3V				1
RED_LED		A0	P1_0	2
	RX	A1	P1_1	3
	TX	A2	P1_2	4
		A3	P1_3	5
		A4	P1_4	6
PUSH2	SCK (B0)	A5	P1_5	7
	CS (B0)		P2_0	8
			P2_1	9
			P2_2	10



The image shows a Texas Instruments MSP430G2553 microcontroller on a red PCB. The chip is centrally located with a 14-pin header on the left and a 20-pin header on the right. Various components are visible: a red LED (RED_LED), two push buttons (PUSH2 and RESET), and several jumpers (J5, J6). The PCB is labeled with 'TEXAS INSTRUMENTS' and 'MSP430G2553'. Pin headers are labeled with P1.0 through P2.7 and A0 through A7. Power pins are labeled UCC and GND. Test pins are labeled XOUT, TEST, RST, and XIN. A 14-pin header is indicated by a dashed line, and a 20-pin header is indicated by a solid line.

20					GROUND
19	P2_6				XIN
18	P2_7				XOUT
17					TEST
16					RESET
15	P1_7	A7	SDA	MOSI (B0)	
14	P1_6	A6	SCL	MISO (B0)	GREEN_LED
13	P2_5				
12	P2_4				
11	P2_3				

Code

```
// Start a temperature measurement:
// If request is successful, the number of ms to wait is returned.
// If request is unsuccessful, 0 is returned.

status = pressure.startTemperature();
if (status != 0)
{
    // Wait for the measurement to complete:
    delay(status);

    // Retrieve the completed temperature measurement:
    // Note that the measurement is stored in the variable T.
    // Function returns 1 if successful, 0 if failure.

    status = pressure.getTemperature(T);
    if (status != 0)
    {
        // Print out the measurement:
        Serial.print("temperature: ");
        Serial.print(T,2);
        Serial.print(" deg C, ");
        Serial.print((9.0/5.0)*T+32.0,2);
        Serial.println(" deg F");
    }
}
```

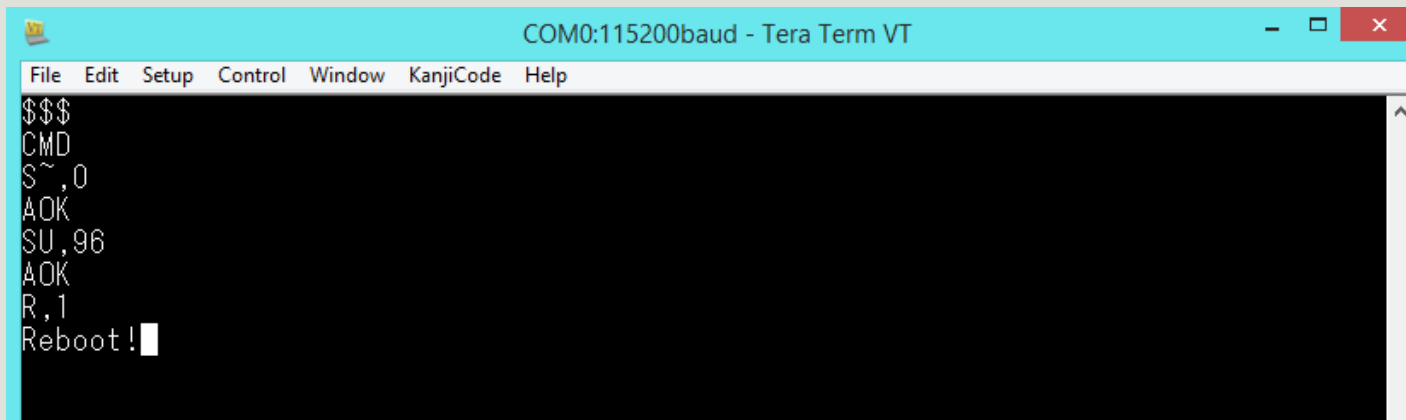
```
// Start a pressure measurement:
// The parameter is the oversampling setting, from 0 to 3 (highest res, longest wait).
// If request is successful, the number of ms to wait is returned.
// If request is unsuccessful, 0 is returned.

status = pressure.startPressure(3);
if (status != 0)
{
    // Wait for the measurement to complete:
    delay(status);

    // Retrieve the completed pressure measurement:
    // Note that the measurement is stored in the variable P.
    // Note also that the function requires the previous temperature measurement (T).
    // (If temperature is stable, you can do one temperature measurement for a number of pressure measurements.)
    // Function returns 1 if successful, 0 if failure.

    status = pressure.getPressure(P,T);
    if (status != 0)
    {
        // Print out the measurement:
        Serial.print("absolute pressure: ");
        Serial.print(P,2);
        Serial.print(" mb, ");
        Serial.print(P*0.0295333727,2);
        Serial.println(" inHg");
        Serial.println("\n");
    }
}
```


Bluetooth Module Setup



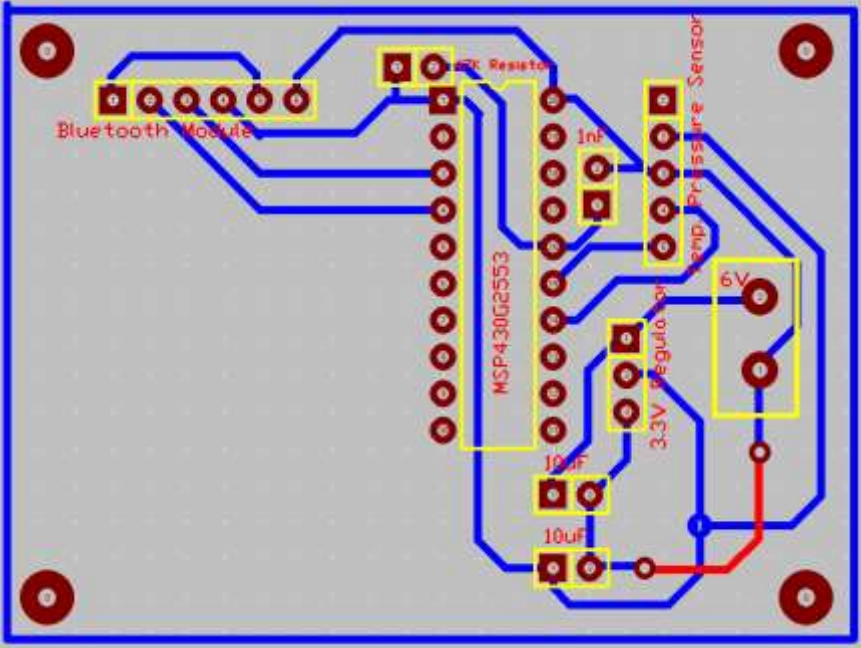
```
COM0:115200baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
$$$
CMD
S~,0
AOK
SU,96
AOK
R,1
Reboot!
```

- \$\$\$ - Enters command mode to allow user to change settings.
- CMD - Bluetooth module enters command mode
- AOK - Bluetooth module responds that command was accepted.
- S~,0 - Sets Bluetooth module to the Serial Port Profile (SPP), acts as serial pipeline.
- SU,96 - Change the baud rate of the Bluetooth module.
- R,1 - Reboots the Bluetooth module.

FTDI Breakout Board - 5V



Final Design and Results



Final Design and Results (cont'd)

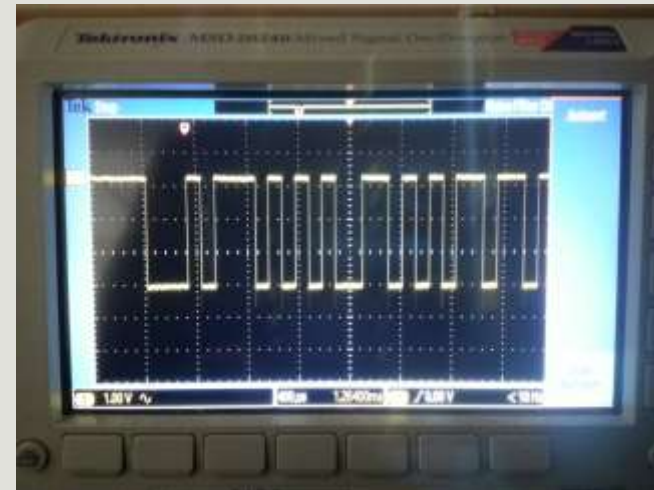
I²C Protocol
Serial Data Line



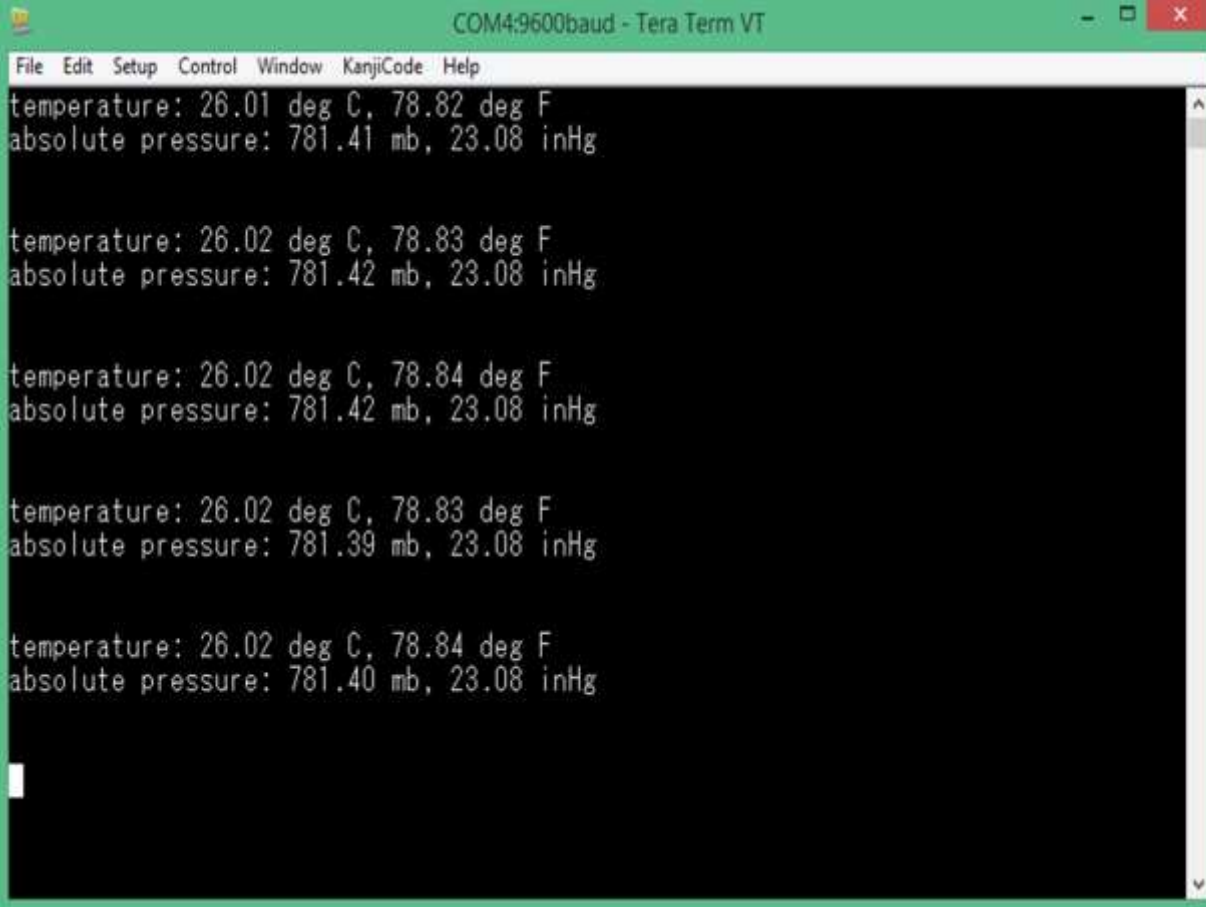
Clock Line



UART Serial Communication



Final Design and Results (cont'd)

A screenshot of a terminal window titled "COM4:9600baud - Tera Term VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", "KanjiCode", and "Help". The terminal displays five lines of sensor data, each consisting of two lines: "temperature: [value] deg C, [value] deg F" and "absolute pressure: [value] mb, [value] inHg". The data values are: 26.01 deg C, 78.82 deg F, 781.41 mb, 23.08 inHg; 26.02 deg C, 78.83 deg F, 781.42 mb, 23.08 inHg; 26.02 deg C, 78.84 deg F, 781.42 mb, 23.08 inHg; 26.02 deg C, 78.83 deg F, 781.39 mb, 23.08 inHg; and 26.02 deg C, 78.84 deg F, 781.40 mb, 23.08 inHg. A white cursor is visible at the bottom left of the terminal area.

```
COM4:9600baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
temperature: 26.01 deg C, 78.82 deg F
absolute pressure: 781.41 mb, 23.08 inHg

temperature: 26.02 deg C, 78.83 deg F
absolute pressure: 781.42 mb, 23.08 inHg

temperature: 26.02 deg C, 78.84 deg F
absolute pressure: 781.42 mb, 23.08 inHg

temperature: 26.02 deg C, 78.83 deg F
absolute pressure: 781.39 mb, 23.08 inHg

temperature: 26.02 deg C, 78.84 deg F
absolute pressure: 781.40 mb, 23.08 inHg
```

Future Considerations

- I found that, when my sensor was connected and transmitting data, the circuit pulled between 30 and 35 mA of current. I used 4 AA batteries to power my circuit, with each battery holding about 2800 mAh of electrical charge. The approximate battery life of my design is a little over 3 days, when connected and transmitting continuously. To increase my battery life, I could program my microcontroller to go into “sleep mode” and conserve energy. I could also increase the time between sensor readings.
- My circuit and battery pack are not enclosed in a box. To make my design easier to handle, I could place the circuit in a box, but that would require some calibration of the sensor.

Conclusion

- I accomplished my goal of learning more about Bluetooth communication.
- I acquired the temperature and barometric pressure accuracy goals.
- I designed a portable temperature and barometric pressure sensor.



Questions
