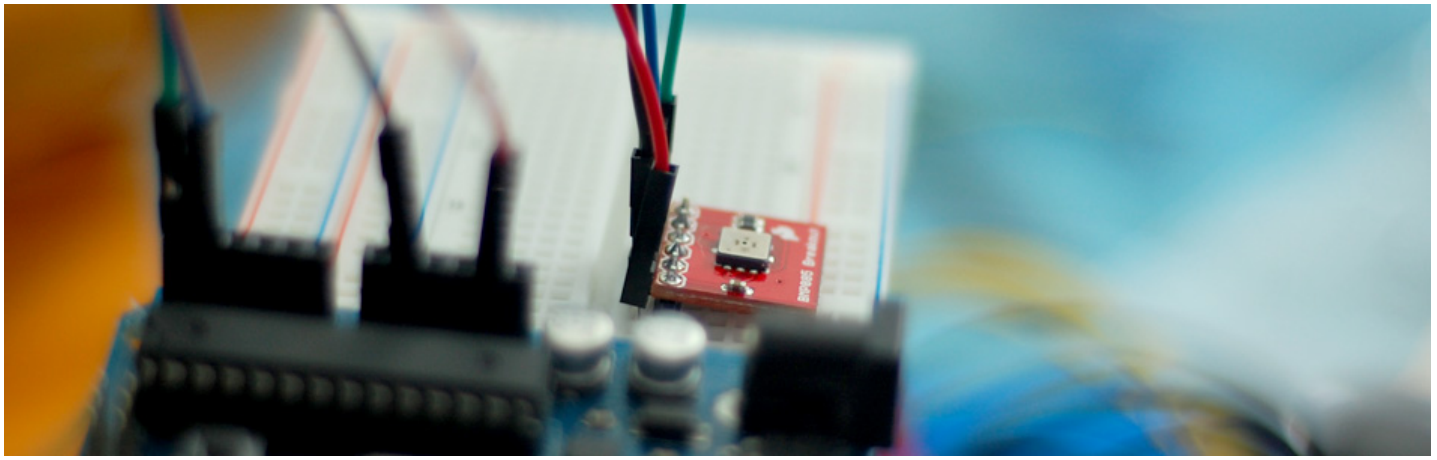
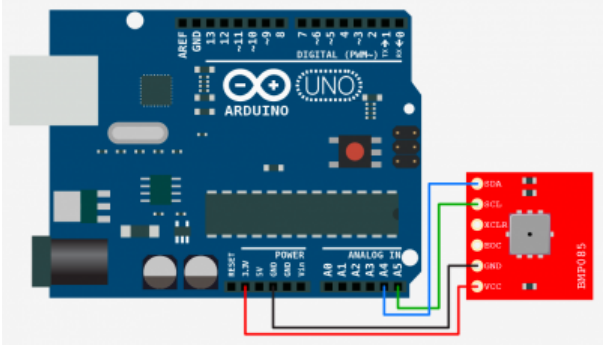
# Sensing Barometric Pressure | BMP085 + Arduino

Thursday, June 16 th , 2011



Light, location, temperature… What's next? Well, how about Barometric pressure? You know.. that thing that determines so much of our weather. Well the BMP085 Barometric Pressure sensor, available at SparkFun is a great little sensor capable of sensing such small changes in barometric pressure it can be used as a pretty precise altimeter as well. And, because no Barometric pressure sensor would be complete without a temperature reading, the BMP085 has an imbedded thermometer. It also looks cool!

For this example, the way we will be calculating altitude is imprecise and does not compensate for temperature or many other things that can contribute to differences. But, if you use it as a comparison against itself during a hour long period (barometric pressure also fluctuates throughout the day), you can get a pretty accurate difference reading.




**The reading differs from my weather report**

Im not going to get into why this is, but to make comparisons easier, your weather report uses a sea-level compensated reading, not an actual reading. This is an actual reading.

## Hooking it up

Hooking it up to your Arduino is pretty simple, the BMP085 is an I2C device. I2C is a 2-wire serial connection, so you just need to connect the BMP085 to power (3.3v) and ground, then the SDA (Data) and SCL (Clock) lines to your Arduino for communication. On your Arduino (everything but the mega) SDA is on analog pin 4, and SCL is on analog pin 5. On an Arduino Mega, SDA is digital 20, and SCL is digital 21.

## Code

The code for this is largely copied and based off of Jim Lindblom's example from SparkFun (Thanks Jim!). I changed a few things so it is just easier for you to use the readings in calculations, and I added the altitude reading as well. **If you know of a better way to calculate altitude, please let us know so we can include it!**

The code simply outputs the temperature, pressure, and altitude in the serial terminal.

Copy Code

```
//Arduino 1.0+ Only
//Arduino 1.0+ Only

/*Based largely on code by  Jim Lindblom

 Get pressure, altitude, and temperature from the BMP085.
 Serial.print it out at 9600 baud to serial monitor.
 */

#include <Wire.h>

#define BMP085_ADDRESS 0x77  // I2C address of BMP085

const unsigned char OSS = 0;  // Oversampling Setting

// Calibration values
int ac1;
int ac2;
int ac3;
unsigned int ac4;
unsigned int ac5;
unsigned int ac6;
int b1;
int b2;
int mb;
int mc;
int md;

// b5 is calculated in bmp085GetTemperature(...), this variable is also used in bmp085GetPressure(...)
// so ...Temperature(...) must be called before ...Pressure(...).
long b5;

void setup(){
  Serial.begin(9600);
  Wire.begin();

  bmp085Calibration();
}

void loop()
{
  float temperature = bmp085GetTemperature(bmp085ReadUT()); //MUST be called first
  float pressure = bmp085GetPressure(bmp085ReadUP());
  float atm = pressure / 101325; // "standard atmosphere"
  float altitude = calcAltitude(pressure); //Uncompensated caculation - in Meters

  Serial.print("Temperature: ");
  Serial.print(temperature, 2); //display 2 decimal places
  Serial.println("deg C");

  Serial.print("Pressure: ");
  Serial.print(pressure, 0); //whole number only.
  Serial.println(" Pa");

  Serial.print("Standard Atmosphere: ");
  Serial.println(atm, 4); //display 4 decimal places

  Serial.print("Altitude: ");
  Serial.print(altitude, 2); //display 2 decimal places
  Serial.println(" M");

  Serial.println();//line break

  delay(1000); //wait a second and get values again.
}

// Stores all of the bmp085's calibration values into global variables
// Calibration values are required to calculate temp and pressure
// This function should be called at the beginning of the program
void bmp085Calibration()
```

```
{
  ac1 = bmp085ReadInt(0xAA);
  ac2 = bmp085ReadInt(0xAC);
  ac3 = bmp085ReadInt(0xAE);
  ac4 = bmp085ReadInt(0xB0);
  ac5 = bmp085ReadInt(0xB2);
  ac6 = bmp085ReadInt(0xB4);
  b1 = bmp085ReadInt(0xB6);
  b2 = bmp085ReadInt(0xB8);
  mb = bmp085ReadInt(0xBA);
  mc = bmp085ReadInt(0xBC);
  md = bmp085ReadInt(0xBE);
}

// Calculate temperature in deg C
float bmp085GetTemperature(unsigned int ut){
  long x1, x2;

  x1 = (((long)ut - (long)ac6)*(long)ac5) >> 15;
  x2 = ((long)mc << 11)/(x1 + md);
  b5 = x1 + x2;

  float temp = ((b5 + 8)>>4);
  temp = temp /10;

  return temp;
}

// Calculate pressure given up
// calibration values must be known
// b5 is also required so bmp085GetTemperature(...) must be called first.
// Value returned will be pressure in units of Pa.
long bmp085GetPressure(unsigned long up){
  long x1, x2, x3, b3, b6, p;
  unsigned long b4, b7;

  b6 = b5 - 4000;
  // Calculate B3
  x1 = (b2 * (b6 * b6)>>12)>>11;
  x2 = (ac2 * b6)>>11;
  x3 = x1 + x2;
  b3 = (((((long)ac1)*4 + x3)<<OSS) + 2)>>2;

  // Calculate B4
  x1 = (ac3 * b6)>>13;
  x2 = (b1 * ((b6 * b6)>>12))>>16;
  x3 = ((x1 + x2) + 2)>>2;
  b4 = (ac4 * (unsigned long)(x3 + 32768))>>15;

  b7 = ((unsigned long)(up - b3) * (50000>>OSS));
  if (b7 < 0x80000000)
    p = (b7<<1)/b4;
  else
    p = (b7/b4)<<1;

  x1 = (p>>8) * (p>>8);
  x1 = (x1 * 3038)>>16;
  x2 = (-7357 * p)>>16;
  p += (x1 + x2 + 3791)>>4;

  long temp = p;
  return temp;
}

// Read 1 byte from the BMP085 at 'address'
char bmp085Read(unsigned char address)
{
  unsigned char data;

  Wire.beginTransmission(BMP085_ADDRESS);
  Wire.write(address);
  Wire.endTransmission();

  Wire.requestFrom(BMP085_ADDRESS, 1);
  while(!Wire.available())
    ;

  return Wire.read();
}

// Read 2 bytes from the BMP085
// First byte will be from 'address'
// Second byte will be from 'address'+1
int bmp085ReadInt(unsigned char address)
{
  unsigned char msb, lsb;
```

```
  Wire.beginTransmission(BMP085_ADDRESS);
  Wire.write(address);
  Wire.endTransmission();

  Wire.requestFrom(BMP085_ADDRESS, 2);
  while(Wire.available()<2)
    ;
  msb = Wire.read();
  lsb = Wire.read();

  return (int) msb<<8 | lsb;
}

// Read the uncompensated temperature value
unsigned int bmp085ReadUT(){
  unsigned int ut;

  // Write 0x2E into Register 0xF4
  // This requests a temperature reading
  Wire.beginTransmission(BMP085_ADDRESS);
  Wire.write(0xF4);
  Wire.write(0x2E);
  Wire.endTransmission();

  // Wait at least 4.5ms
  delay(5);

  // Read two bytes from registers 0xF6 and 0xF7
  ut = bmp085ReadInt(0xF6);
  return ut;
}

// Read the uncompensated pressure value
unsigned long bmp085ReadUP(){

  unsigned char msb, lsb, xlsb;
  unsigned long up = 0;

  // Write 0x34+(OSS<<6) into register 0xF4
  // Request a pressure reading w/ oversampling setting
  Wire.beginTransmission(BMP085_ADDRESS);
  Wire.write(0xF4);
  Wire.write(0x34 + (OSS<<6));
  Wire.endTransmission();

  // Wait for conversion, delay time dependent on OSS
  delay(2 + (3<<OSS));

  // Read register 0xF6 (MSB), 0xF7 (LSB), and 0xF8 (XLSB)
  msb = bmp085Read(0xF6);
  lsb = bmp085Read(0xF7);
  xlsb = bmp085Read(0xF8);

  up = (((unsigned long) msb << 16) | ((unsigned long) lsb << 8) | (unsigned long) xlsb) >> (8-OSS);

  return up;
}

void writeRegister(int deviceAddress, byte address, byte val) {
  Wire.beginTransmission(deviceAddress); // start transmission to device
  Wire.write(address);       // send register address
  Wire.write(val);         // send value to write
  Wire.endTransmission();     // end transmission
}

int readRegister(int deviceAddress, byte address){

  int v;
  Wire.beginTransmission(deviceAddress);
  Wire.write(address); // register to read
  Wire.endTransmission();

  Wire.requestFrom(deviceAddress, 1); // read a byte

  while(!Wire.available()) {
    // waiting
  }

  v = Wire.read();
  return v;
}

float calcAltitude(float pressure){

  float A = pressure/101325;
  float B = 1/5.25588;
  float C = pow(A,B);
```

```
  C = 1 - C;
  C = C /0.0000225577;

  return C;
}
```



 4 Comments

## Archives

- October 2013
- November 2012
- August 2012
- April 2012
- March 2012
- February 2012
- January 2012
- October 2011
- September 2011
- August 2011
- July 2011
- June 2011
- May 2011
- April 2011
- March 2011
- February 2011
- January 2011
- December 2010
- November 2010
- October 2010
- September 2010

## Categories

- funded projects
- Promotions
- Review
- Site
- Tutorials
- User Contributed Articles
- User Project

## about

- Content available under:
  Attribution-Share Alike 3.0 Unported
- Proudly powered by WordPress, **Free WordPress Themes, and Search Marketing**