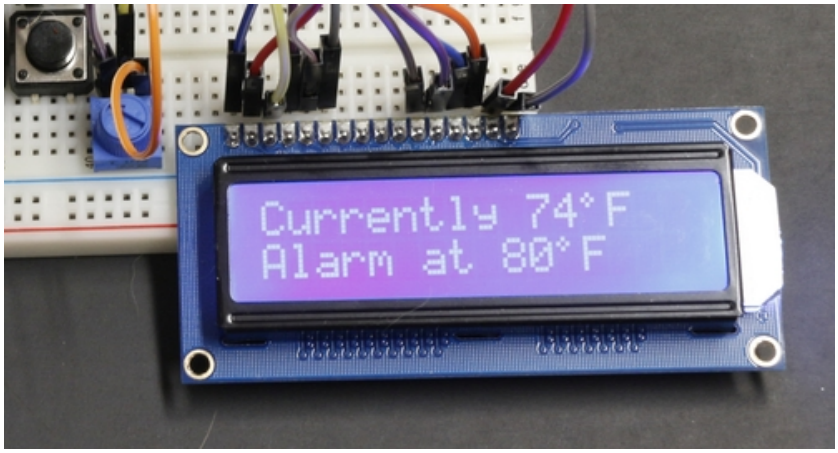
adafruit learning system

Adabox 001 Temperature Alarm

Created by Kirby Griese



Last updated on 2017-08-10 02:06:30 AM UTC

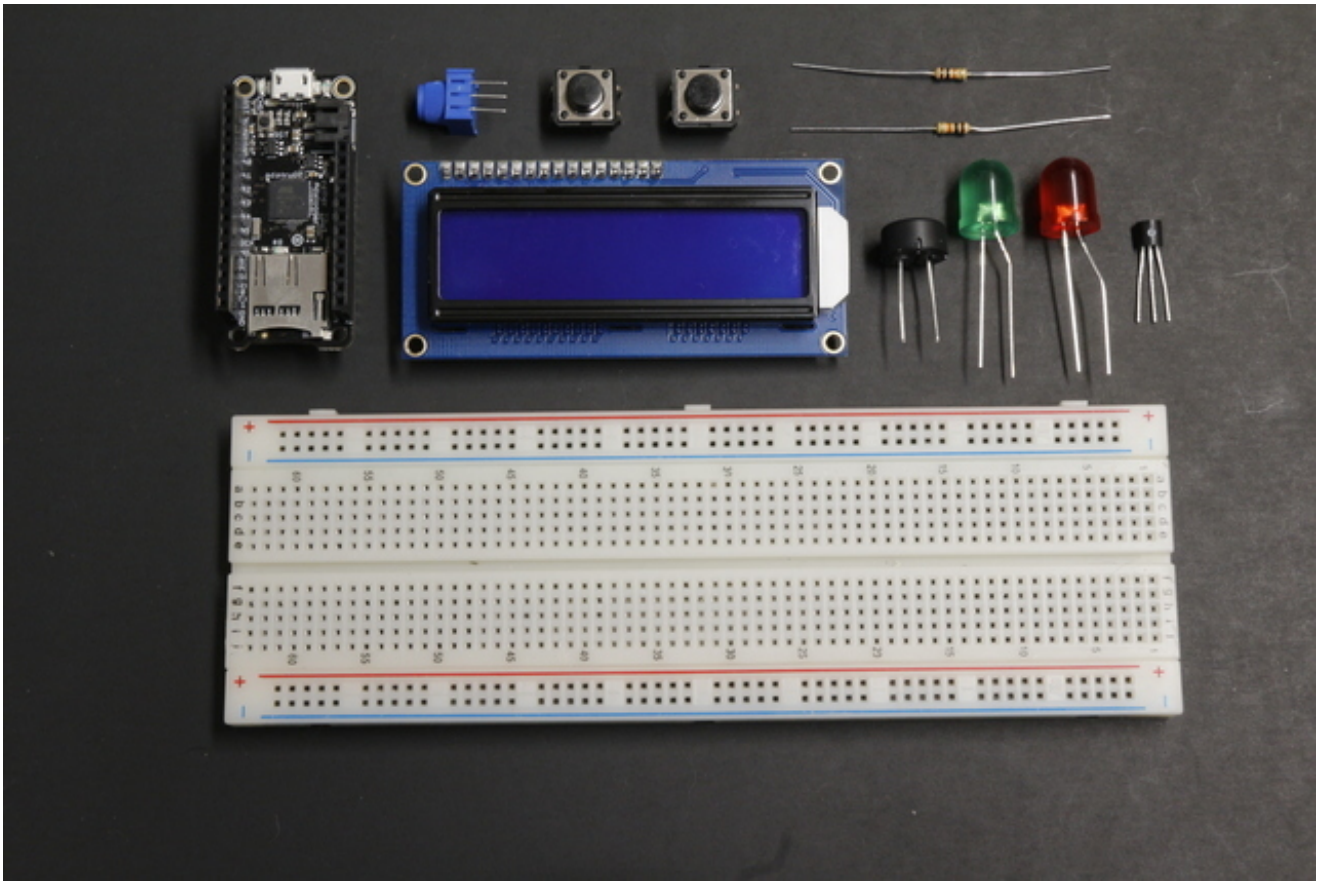
Guide Contents

Guide Contents 2
Overview 3
Parts Needed 3
Assembly 5
Assembled Standard LCD 16x2 + extras - White on Blue 5
16x2 LCD 6
TMP36 9
Alarm 13
LED Status 19
Going Further 25
DHT11 basic temperature-humidity sensor + extras 25

Overview

Now that you got Adabox 001 you are probably wondering what you can make with it. I always like making temperature monitors since it is fairly simple and can be used to interact with quite a few other items.

In this guide we are going to build a device to read the temperature and sound an alarm if the temperature is too high.



Parts Needed

- Adabox 001 (http://adafru.it/tUc) - This will include all of the following needed parts
- Adafruit Feather 32u4 Adalogger (http://adafru.it/tUd)
- 16x2 Standard LCD (http://adafru.it/tUe)
- Piezo Buzzer (http://adafru.it/dCD)
- Diffused Red 10mm LED (http://adafru.it/eby)
- Diffused Green 10mm LED (http://adafru.it/tUf)

- Two 12mm Tactile Switch Buttons (http://adafru.it/f75)
- TMP36 - Analog Temperature Sensor (http://adafru.it/165)
- Full sized breadboard (http://adafru.it/239)
- Two 560ohm Resistors

Assembly

The Temperature Monitor and Alarm will be assembled in four steps. This will let us verify that each part is setup correctly and works.

The first step will be setting up the 16x2 LCD Display, read on!

Assembled Standard LCD 16x2 + extras - White on Blue

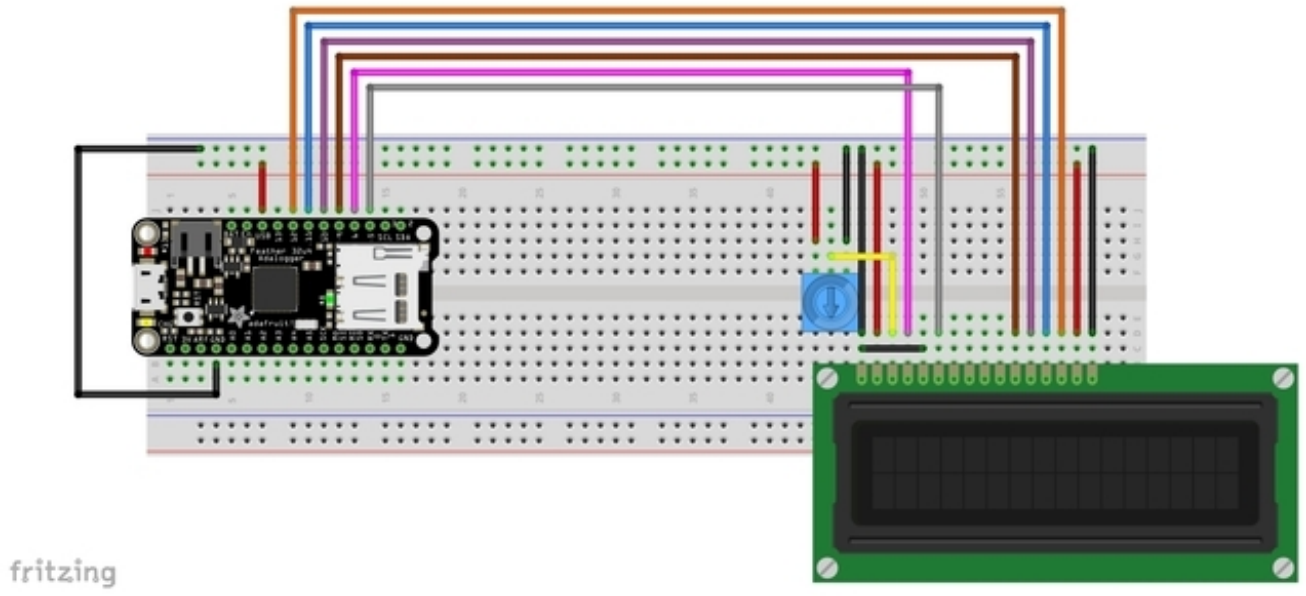
PRODUCT ID: 1447
Standard HD44780 LCDs are useful for creating standalone projects. This product is similar to our Standard LCD 16x2 display but comes with the header soldered on! 16 characters...
http://adafru.it/tUe
$10.95
IN STOCK

16x2 LCD

These are the connections needed to wire the 16x2 LCD Display to the feather. You can also follow the fritzing diagram.

LCD Connections

- **LCD #1 (leftmost)** to **Feather GND**
- **LCD 2** to **Feather USB**
- **LCD 3** to **Middle pin Potentiometer**
- **LCD 4** to **Feather 6**
- **LCD 5** to **Feather GND**
- **LCD 6** to **Feather 5**
- *skip 7, 8, 9, 10*
- **LCD 11** to **Feather 9**
- **LCD 12** to **Feather 10**
- **LCD 13** to **Feather11**
- **LCD 14** to **Feather 12**
- **LCD 15** to **Feather USB**
- **LCD 16 (rightmost)** to **Feather GND**



The potentiometer is used for adjusting the contrast. To connect that we will connect the

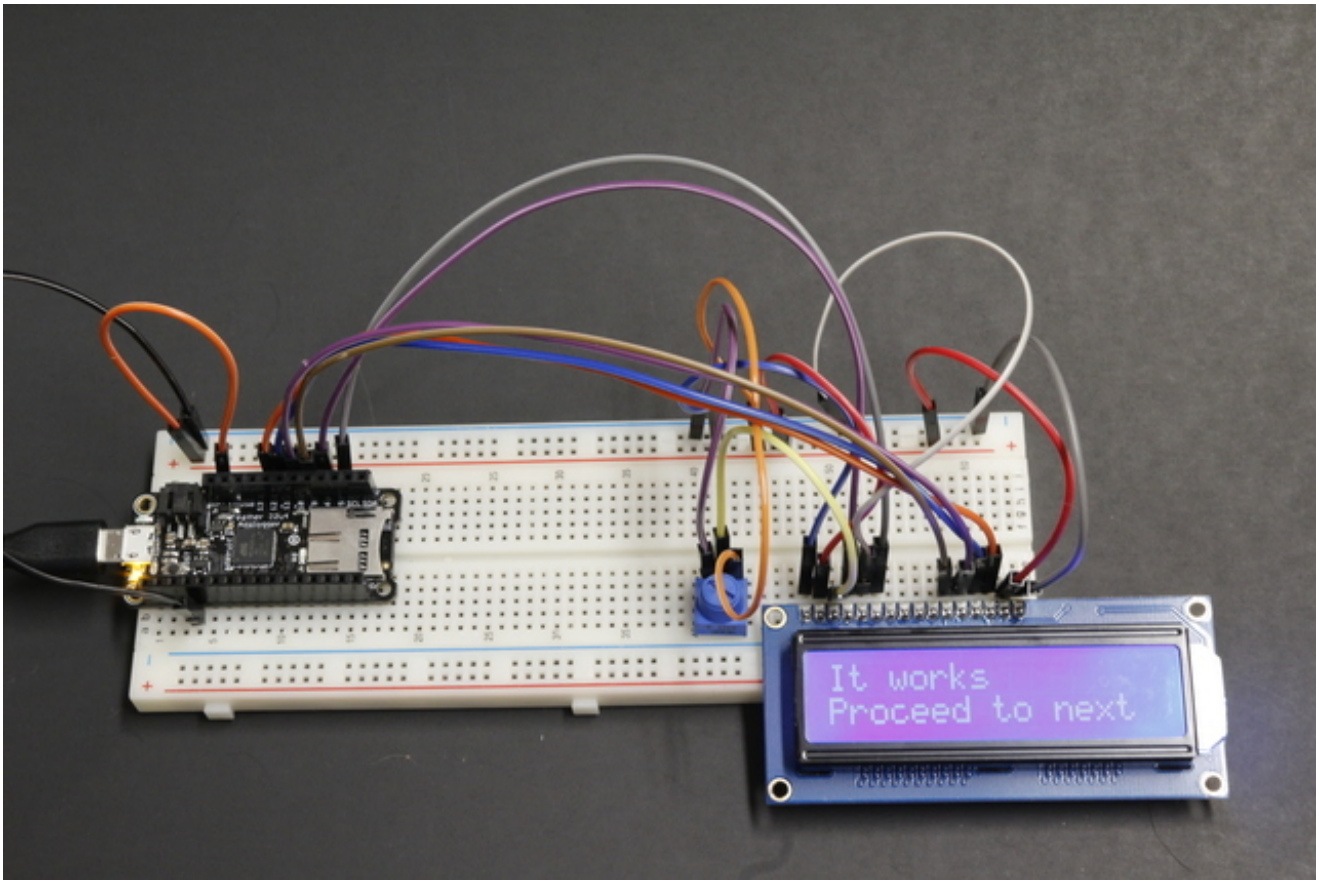
middle pin to **LCD 3**. Then one pin to **Feather USB** and the other pin to **Feather GND**. It does not matter which outer pin is power or ground.

The power for the potentiometer and LCD are connected to USB on the Feather since that will provide 5V when powered via USB. The 3.3V output on the Feather can not power the display.

The following is the code to load on the Feather.

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(6, 5, 9, 10, 11, 12);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  lcd.print("It works");
  lcd.setCursor(0,1); //2nd line on display
  lcd.print("Proceed to next");
}

void loop() {

}
```

If everything is set correctly this is how it should look and what will be on the display. You may need to adjust the contrast potentiomer if you do not see the text.



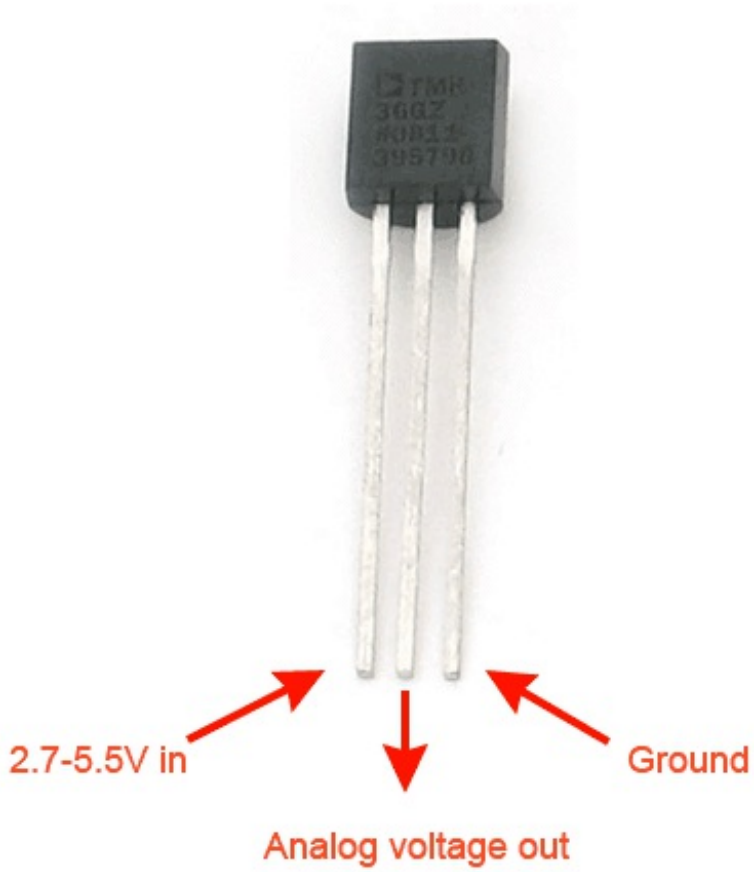
For more information on using the 16x2 LCD display with Arduino check out this guide.

Arduino Lesson 11. LCD Displays - Part 1
http://adafru.it/tUA

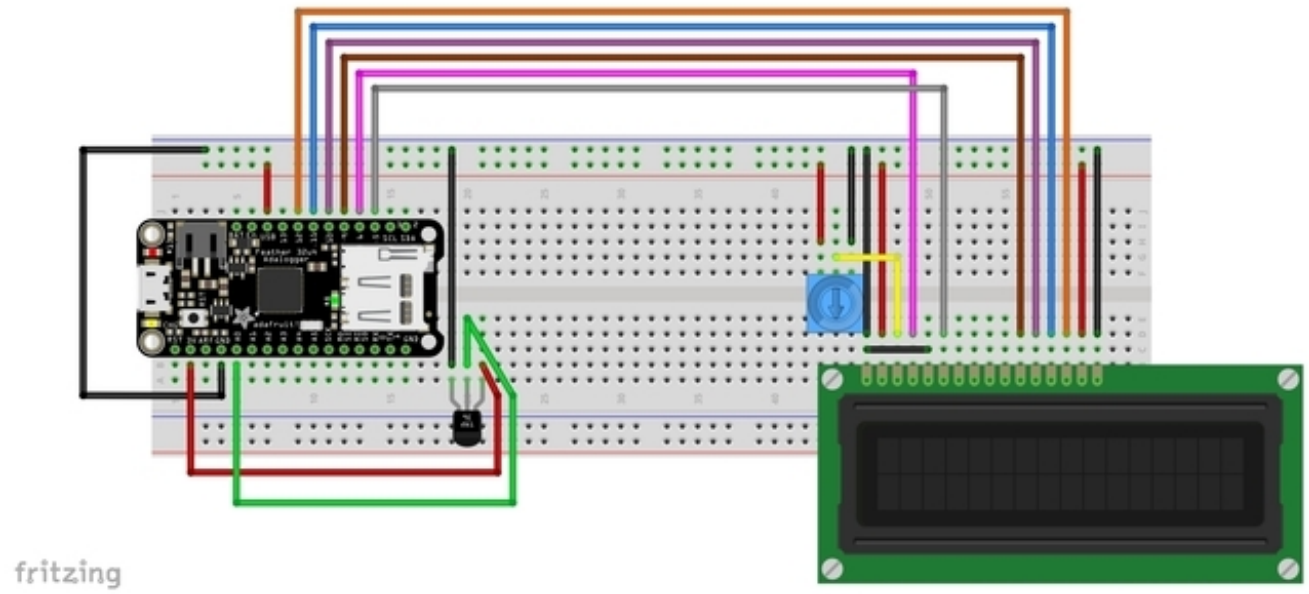
TMP36

Now that the display is working we can add the TMP36 to read the temperature.

There are 3 pins on the TMP36. The left pin is power, middle pin is analog voltage out and the right pin is ground.



We will connect the power pin to **3V** on the Feather. This way it will work when on battery powersince the USB pin will not be powered when on battery only. Analog voltage out will connect to **A0** on the Feather. Ground to **GND** on the Feather.



The code for this part is longer and completely replaces the code from the previous step. I have also included a degree symbol bitmap so it displays the temperature nicely. We can also choose to either display Celsius or Fahrenheit.

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(6, 5, 9, 10, 11, 12);

// Degree symbol bitmap
byte degree[8] = {
  B01000,
  B10100,
  B01000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
};

//TMP36 Pin Variables
const int sensorPin = A0;  //the analog pin the TMP36's Vout (sense) pin is connected to

//Set to 1 to display Celsius instead of Fahrenheit
int celsius = 0;

void setup() {
```

```
  //Create the degree symbol bitmap
  lcd.createChar(0, degree);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
}


void loop() {

  //Clear LCD
  lcd.clear();

  //Display Currently on the LCD
  lcd.print("Currently ") ;

  //getting the voltage reading from the temperature sensor
  int reading = analogRead(sensorPin);
  // converting that reading to voltage, for 3.3v arduino use 3.3
  float voltage = reading * 3.3;
  voltage /= 1024.0;

  float temperatureC = (voltage - 0.5) * 100 ;  //converting from 10 mv per degree with 500 mV offset
            //to degrees ((voltage - 500mV) times 100)

  if (celsius == 1) //If you set temperature as Celsius it will print Celsius values
  {
    //Round the temperature to a whole number
    float roundedTempC = round(temperatureC);

    // Display temperature in C
    lcd.print(roundedTempC, 0);
    lcd.write(byte(0)); //Degree symbol we created earlier
    lcd.print("C");
  }
  else //Display in Fahrenheit
  {
    //Convert from Celsius to Fahrenheit
    float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;

    //Round the temperature to a whole number
    float roundedTempF = round(temperatureF);

    // Display temperature in F
    lcd.print(roundedTempF, 0);
    lcd.write(byte(0)); //Degree symbol we created earlier
    lcd.print("F");
  }
  delay(1000);
}
```

If you want to change from Fahrenheit to Celsius change this section in the code.

```
//Set to 1 to display Celsius instead of Fahrenheit
int celsius = 0;
```

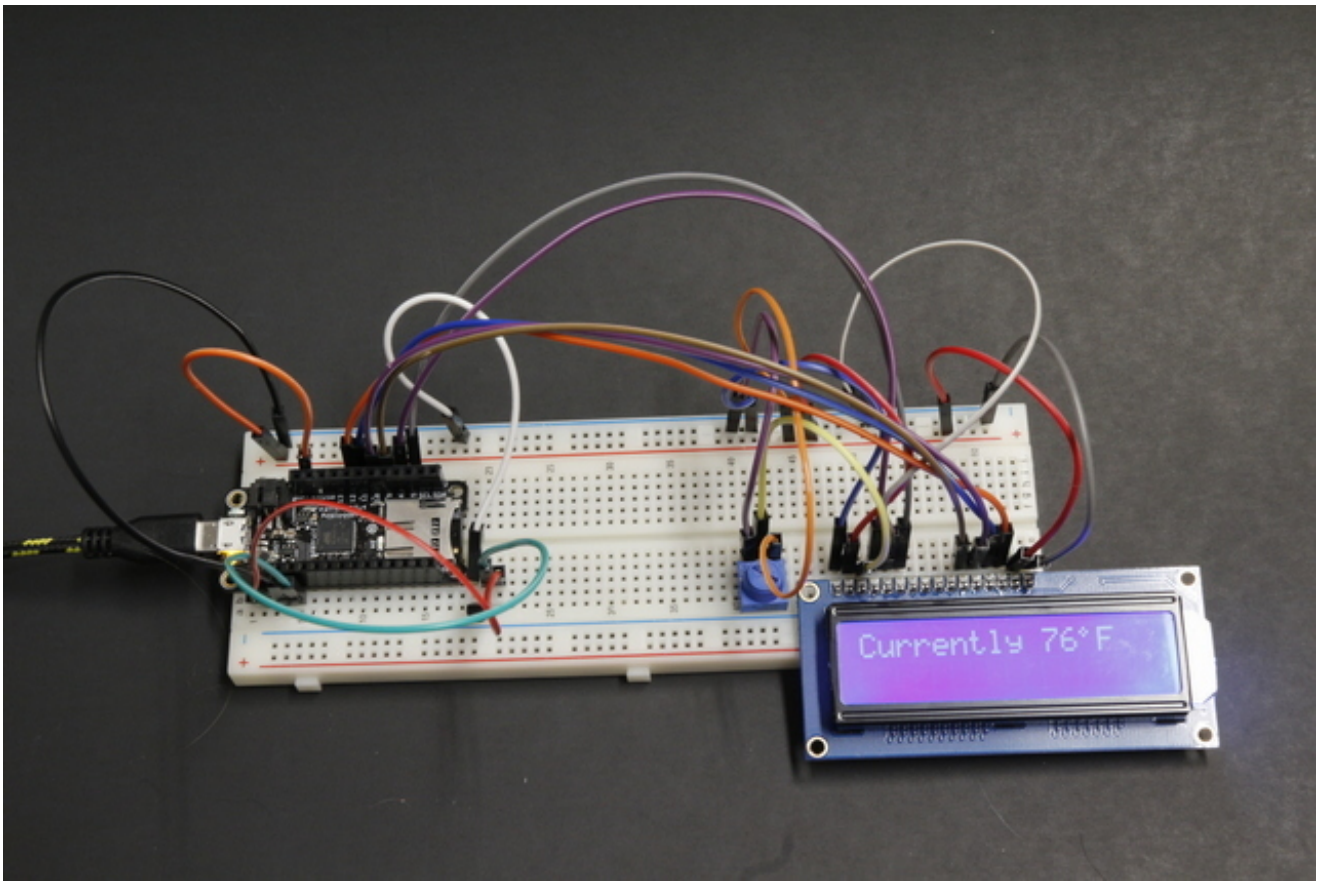
This way displays the temperature in Fahrenheit.

```
//Set to 1 to display Celsius instead of Fahrenheit
int celsius = 1;
```

With the value now changed to 1 it would display the temperature in Celsius. In the loop section of the program it checks this setting and then chooses which format to display

If the temperature reading seems abnormal, check that the connections are good. A bad ground connection could give inaccurate values.

Now you should have a good temperature reading on the display.



For more information on using the TMP36 check out this guide.

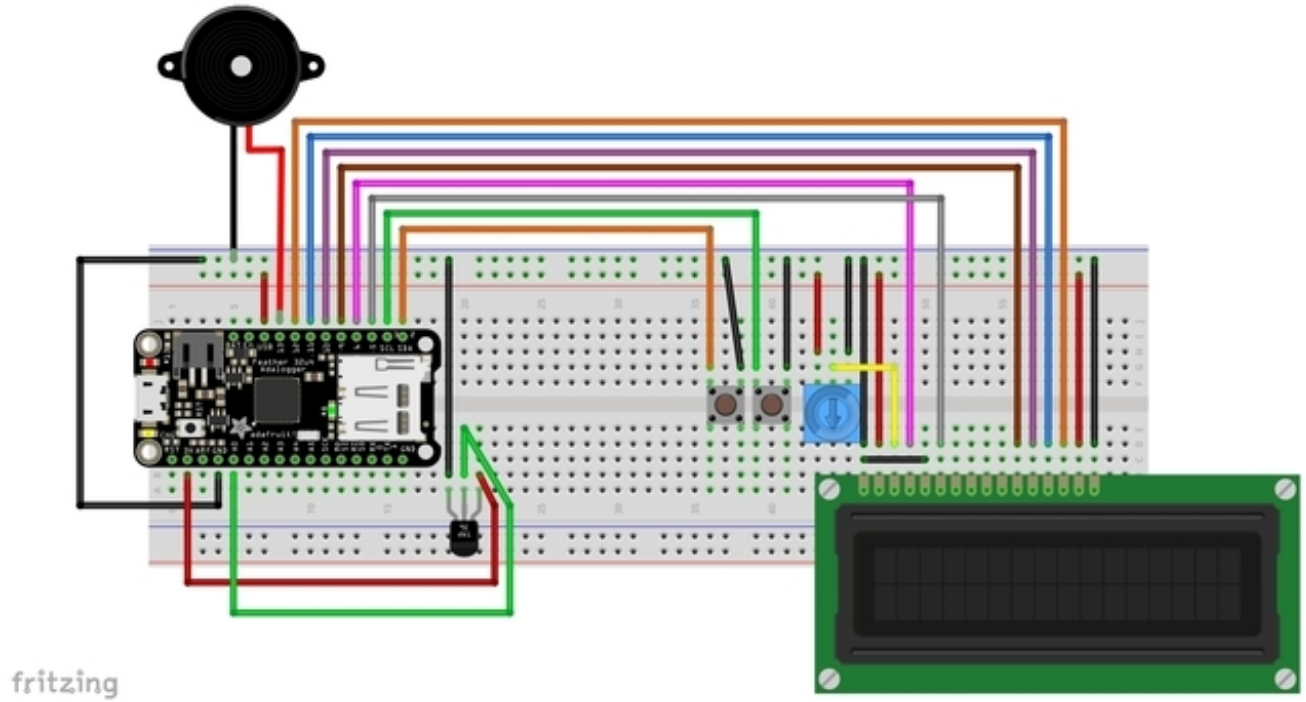
[TMP36 Temperature Sensor](http://adafru.it/tUB)
http://adafru.it/tUB

Alarm

To add alarm functionality we will be using two tactile switch buttons and a piezo buzzer.

The button to lower the alarm value will be wired to pin **2** on the feather. The raise alarm value button will be wired to pin **3** on the feather. Each button will have the other leg wired to **ground** as seen in the fritzing diagram below.

For the Piezo Buzzer one wire will go to **ground** and the other to pin **13** on the feather.



The code is mostly the same from previous step except for a few additions.

- The buttons and what pin they are assigned to.
- A starting value for alarmTemp. This can be changed to any value.
- Buzzer as an output and the pin it is connected to.
- We also add functions to run when a button is pressed.
- An if statement that checks if the alarm has been reached and should alarm

```
// include the library code:
#include <LiquidCrystal.h>
#include <math.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(6, 5, 9, 10, 11, 12);
```

```
// Degree symbol bitmap
byte degree[8] = {
  B01000,
  B10100,
  B01000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
};

//TMP36 Pin Variables
const int sensorPin = A0; //the analog pin the TMP36's Vout (sense) pin is connected to

//Pins for Tactile buttons used for raising or lowering alarm temperature
const int lowerAlarmTemp = 2;
const int raiseAlarmTemp = 3;

//Alarm value, change to what starting value should be.
int alarmTemp = 80;

//Buzzer Pin
const int buzzer = 13;
//frequency out
int freq;

//Set to 1 to display Celsius instead of Fahrenheit
int celsius = 0;

void setup() {
  //Create the degree symbol bitmap
  lcd.createChar(0, degree);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  //Set tactile buttons as inputs, use of pullup means we dont need external resistor
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);

  // Set buzzer pin as output
  pinMode(13, OUTPUT);
}

//Function that runs when button pressed to lower alarm temperature
void lowerAlarm()
{

  static unsigned long last_interrupt_time = 0;
```

```
  unsigned long interrupt_time = millis();
  // If interrupts come faster than 400ms, assume it's a bounce and ignore
  if (interrupt_time - last_interrupt_time > 400)
  {
    alarmTemp = alarmTemp - 1; //Lower alarmTemp by one
  }
  last_interrupt_time = interrupt_time;

}

//Function that runs when button pressed to raise alarm temperature
void raiseAlarm()
{
  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
  // If interrupts come faster than 400ms, assume it's a bounce and ignore
  if (interrupt_time - last_interrupt_time > 400)
  {
    alarmTemp = alarmTemp + 1; //Raise alarmTemp by one
  }
  last_interrupt_time = interrupt_time;
}

void loop() {

  //Interupt that when lowerAlarmTemp button is pressed runs lowerAlarm function
  attachInterrupt(digitalPinToInterrupt(lowerAlarmTemp), lowerAlarm, FALLING);
  //Interupt that when raiseAlarmTemp button is pressed runs raiseAlarm function
  attachInterrupt(digitalPinToInterrupt(raiseAlarmTemp), raiseAlarm, FALLING);

  //Clear LCD
  lcd.clear();

  //Display Currently on the LCD
  lcd.print("Currently ") ;

  //getting the voltage reading from the temperature sensor
  int reading = analogRead(sensorPin);
  // converting that reading to voltage, for 3.3v arduino use 3.3
  float voltage = reading * 3.3;
  voltage /= 1024.0;

  float temperatureC = (voltage - 0.5) * 100 ;  //converting from 10 mv per degree with 500 mV offset
  //to degrees ((voltage - 500mV) times 100)

  if (celsius == 1) //If you set temperature as Celsius it will print Celsius values
  {
    //Round the temperature to a whole number
    float roundedTempC = round(temperatureC);

    // Display temperature in C
```

```
  lcd.print(roundedTempC, 0);
  lcd.write(byte(0)); //Degree symbol we created earlier
  lcd.print("C");

  // Display alarm Temp
  lcd.setCursor(0, 1);
  lcd.print("Alarm at ");
  lcd.print(alarmTemp);
  lcd.write(byte(0));
  lcd.print("C");

  //Check if temperature is equal or greater than alarmTemp
  if (roundedTempC >= alarmTemp)
  {
    tone(buzzer, 200); // Play alarm tone
    delay(400);
    noTone(buzzer);
  }
  else
  {
    noTone(buzzer); //Make sure alarm is off
  }
}
else //Display in Fahrenheit
{
  //Convert from Celsius to Fahrenheit
  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;

  //Round the temperature to a whole number
  float roundedTempF = round(temperatureF);

  // Display temperature in F
  lcd.print(roundedTempF, 0);
  lcd.write(byte(0)); //Degree symbol we created earlier
  lcd.print("F");

  // Display alarm Temp
  lcd.setCursor(0, 1);
  lcd.print("Alarm at ");
  lcd.print(alarmTemp);
  lcd.write(byte(0));
  lcd.print("F");

  //Check if temperature is equal or greater than alarmTemp
  if (roundedTempF >= alarmTemp)
  {
    tone(buzzer, 200); // Play alarm tone
    delay(400);
    noTone(buzzer);
  }
  else
```

```
    {
      noTone(buzzer); //Make sure alarm is off
    }
  }
  delay(1000);
}
```

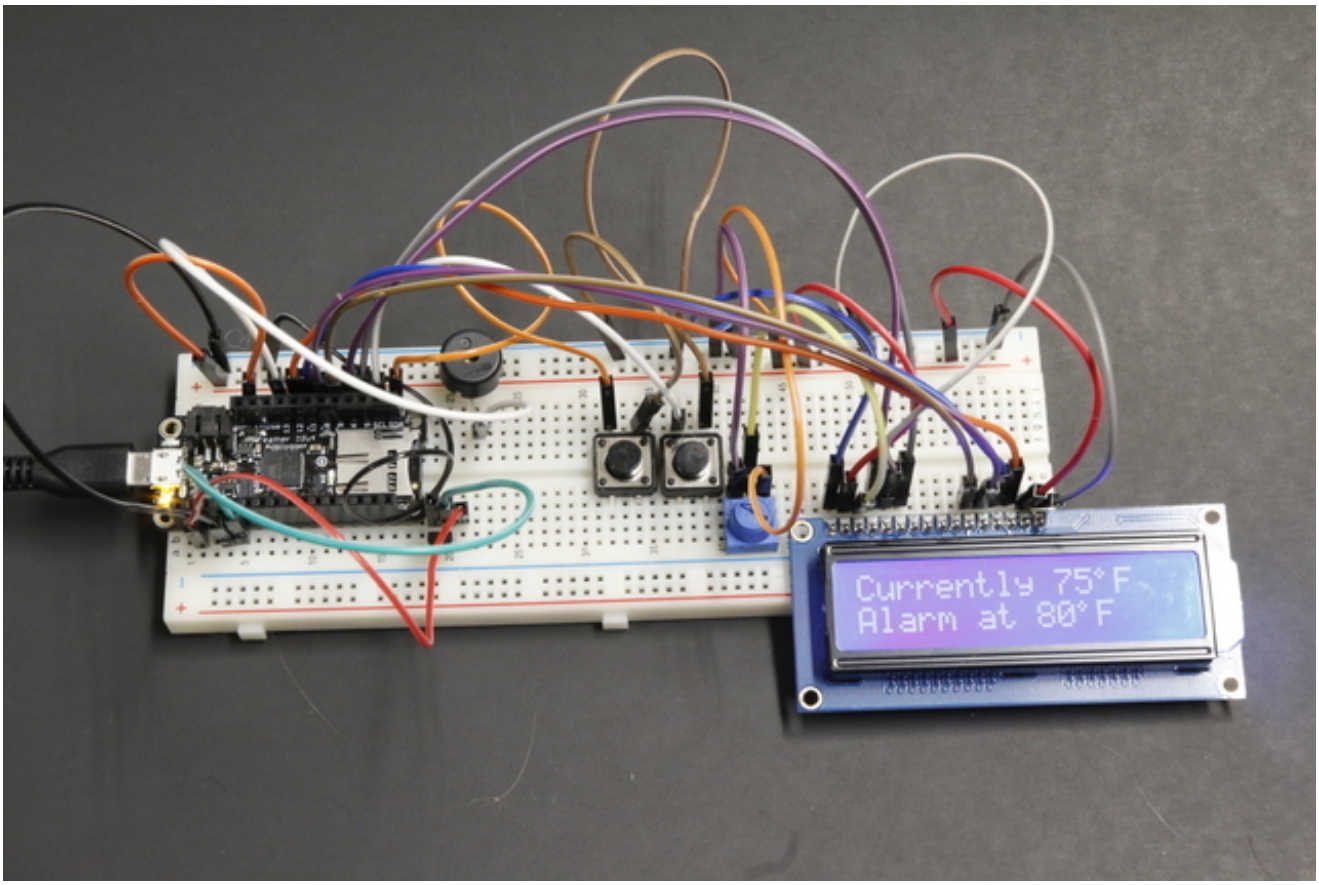
Currently the alarm is set to alarm if the temperature is greater than or equal to the set value. To change to alarm if it is at or below the set value we would look for these two individual lines in the code.

```
if (roundedTempC >= alarmTemp)
if (roundedTempF >= alarmTemp)
```

We would then change them both to less than or equal

```
if (roundedTempC <= alarmTemp)
if (roundedTempF <= alarmTemp)
```

If everything is good your wiring should look like this and the display should be showing the temperature and the alarm value.



For more information on using tactile switches check out this guide.

[Arduino Lesson 6. Digital Inputs]
http://adafru.it/tUC

To learn more about using a piezo buzzer check out this guide.

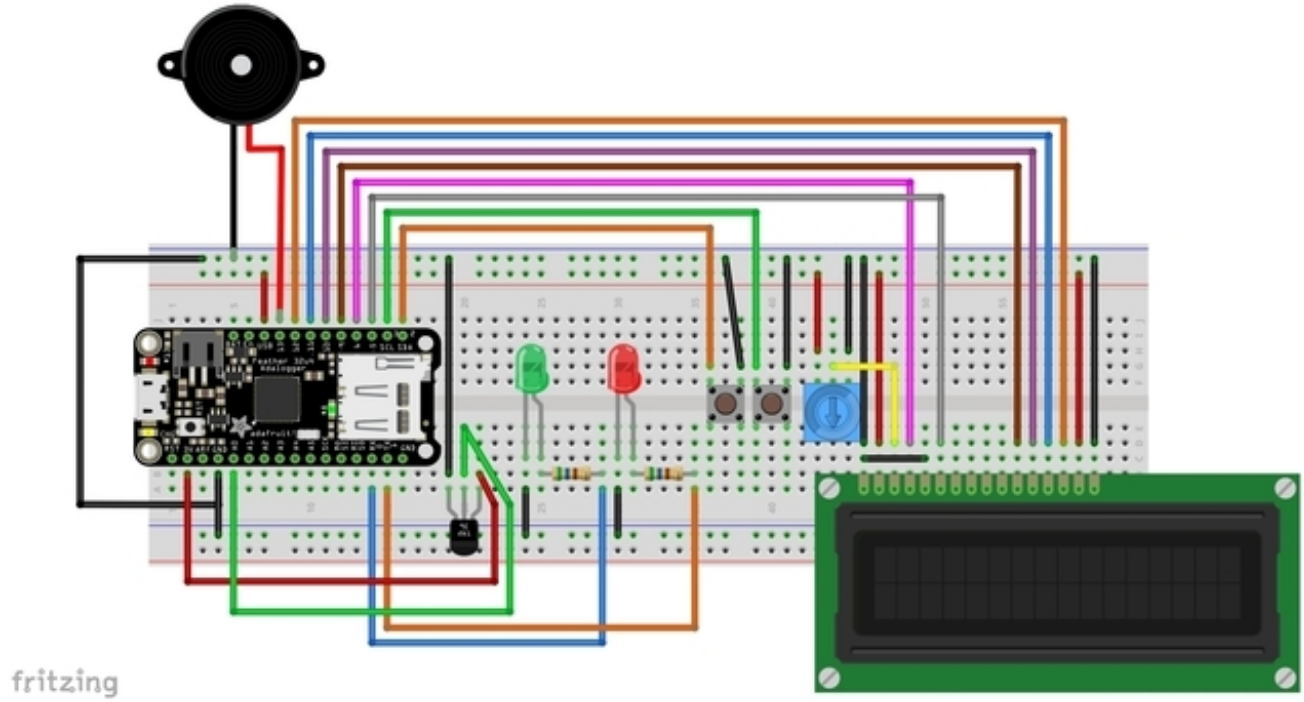
[Arduino Lesson 10. Making Sounds]
http://adafru.it/tUD

LED Status

Now we will add a Green LED and a Red LED as status indicators. We will also need two 560 ohm resistors.

For the Green LED we will wire the cathode(-) to**ground**. The cathode is the shorter length lead. The anode(+) will connect to the 560 Ohm resistor. The other end of the resistor will go to pin **0** on the feather.

The Red LED will be mostly similiar but connected to pin**1** on the Feather.



We will be using the code from previous step but adding the parts for the LED's

- The LED's and what pin they are assigned to
- Set the pins the LED's are on as outputs
- Turning on red LED if alarming
- Turning on green LED if not alarming

```
// include the library code:
#include <LiquidCrystal.h>
#include <math.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(6, 5, 9, 10, 11, 12);
```

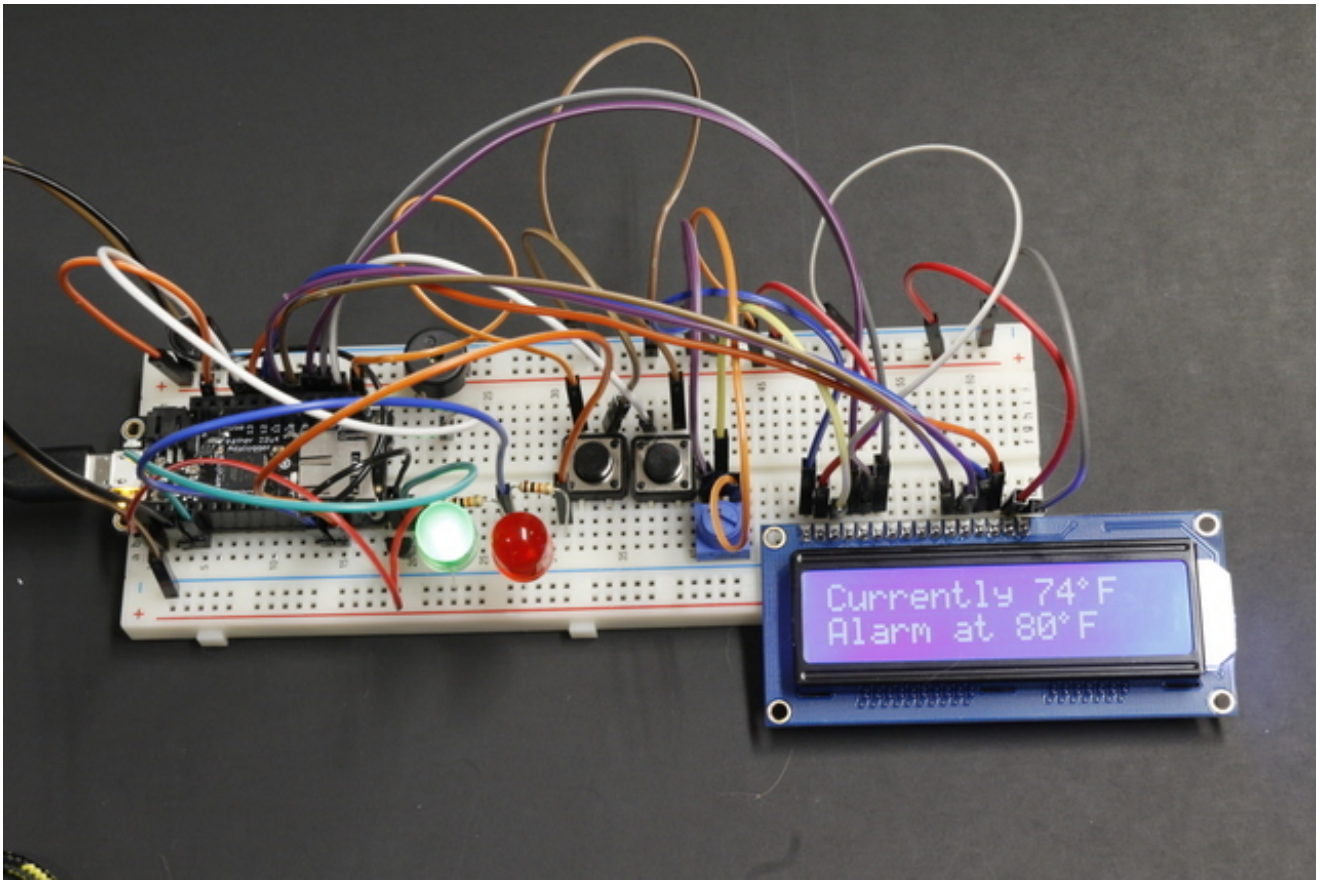
```
// Degree symbol bitmap
byte degree[8] = {
  B01000,
  B10100,
  B01000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
};

//TMP36 Pin Variables
const int sensorPin = A0; //the analog pin the TMP36's Vout (sense) pin is connected to

//Pins for Tactile buttons used for raising or lowering alarm temperature
const int lowerAlarmTemp = 2;
const int raiseAlarmTemp = 3;

//Alarm value, change to what starting value should be.
int alarmTemp = 80;

//Buzzer Pin
const int buzzer = 13;
//frequency out
int freq;

//Red and Green status LED's
const int redLed = 1;
const int greenLed = 0;

//Set to 1 to display Celsius instead of Fahrenheit
int celsius = 0;

void setup() {
  //Create the degree symbol bitmap
  lcd.createChar(0, degree);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  //Set tactile buttons as inputs, use of pullup means we dont need external resistor
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);

  // Set buzzer pin as output
  pinMode(13, OUTPUT);

  //Set LED's as output
  pinMode(0, OUTPUT);
```

```
  pinMode(1, OUTPUT);
}

//Function that runs when button pressed to lower alarm temperature
void lowerAlarm()
{

  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
  // If interrupts come faster than 400ms, assume it's a bounce and ignore
  if (interrupt_time - last_interrupt_time > 400)
  {
    alarmTemp = alarmTemp - 1; //Lower alarmTemp by one
  }
  last_interrupt_time = interrupt_time;

}

//Function that runs when button pressed to raise alarm temperature
void raiseAlarm()
{
  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
  // If interrupts come faster than 400ms, assume it's a bounce and ignore
  if (interrupt_time - last_interrupt_time > 400)
  {
    alarmTemp = alarmTemp + 1; //Raise alarmTemp by one
  }
  last_interrupt_time = interrupt_time;
}

void loop() {

  //Interupt that when lowerAlarmTemp button is pressed runs lowerAlarm function
  attachInterrupt(digitalPinToInterrupt(lowerAlarmTemp), lowerAlarm, FALLING);
  //Interupt that when raiseAlarmTemp button is pressed runs raiseAlarm function
  attachInterrupt(digitalPinToInterrupt(raiseAlarmTemp), raiseAlarm, FALLING);

  //Clear LCD
  lcd.clear();

  //Display Currently on the LCD
  lcd.print("Currently ") ;

  //getting the voltage reading from the temperature sensor
  int reading = analogRead(sensorPin);
  // converting that reading to voltage, for 3.3v arduino use 3.3
  float voltage = reading * 3.3;
  voltage /= 1024.0;

  float temperatureC = (voltage - 0.5) * 100 ;  //converting from 10 mv per degree with 500 mV offset
```

```
//to degrees ((voltage - 500mV) times 100)

if (celsius == 1) //If you set temperature as Celsius it will print Celsius values
{
  //Round the temperature to a whole number
  float roundedTempC = round(temperatureC);

  // Display temperature in C
  lcd.print(roundedTempC, 0);
  lcd.write(byte(0)); //Degree symbol we created earlier
  lcd.print("C");

  // Display alarm Temp
  lcd.setCursor(0, 1);
  lcd.print("Alarm at ");
  lcd.print(alarmTemp);
  lcd.write(byte(0));
  lcd.print("C");

  //Check if temperature is equal or greater than alarmTemp
  if (roundedTempC >= alarmTemp)
  {
    tone(buzzer, 200); // Play alarm tone
    delay(400);
    noTone(buzzer);
    digitalWrite(redLed, HIGH); //Turn red LED on
    digitalWrite(greenLed, LOW); //Turn green LED off
  }
  else
  {
    noTone(buzzer); //Make sure alarm is off
    digitalWrite(redLed, LOW); //Turn red LED off
    digitalWrite(greenLed, HIGH); //Turn green LED on
  }
}
else //Display in Fahrenheit
{
  //Convert from Celsius to Fahrenheit
  float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;

  //Round the temperature to a whole number
  float roundedTempF = round(temperatureF);

  // Display temperature in F
  lcd.print(roundedTempF, 0);
  lcd.write(byte(0)); //Degree symbol we created earlier
  lcd.print("F");

  // Display alarm Temp
  lcd.setCursor(0, 1);
  lcd.print("Alarm at ");
```

```
    lcd.print(alarmTemp);
    lcd.write(byte(0));
    lcd.print("F");

    //Check if temperature is equal or greater than alarmTemp
    if (roundedTempF >= alarmTemp)
    {
      tone(buzzer, 200); // Play alarm tone
      delay(400);
      noTone(buzzer);
      digitalWrite(redLed, HIGH); //Turn red LED on
      digitalWrite(greenLed, LOW); //Turn green LED off
    }
    else
    {
      noTone(buzzer); //Make sure alarm is off
      digitalWrite(redLed, LOW); //Turn red LED off
      digitalWrite(greenLed, HIGH); //Turn green LED on
    }
  }
  delay(1000);
}
```

After loading the new code on the Feather it should look like this picture and the green LED should be illuminated if below the alarm level. Otherwise the red LED should be on.



For more information on using LEDs with Arduino check out this guide.

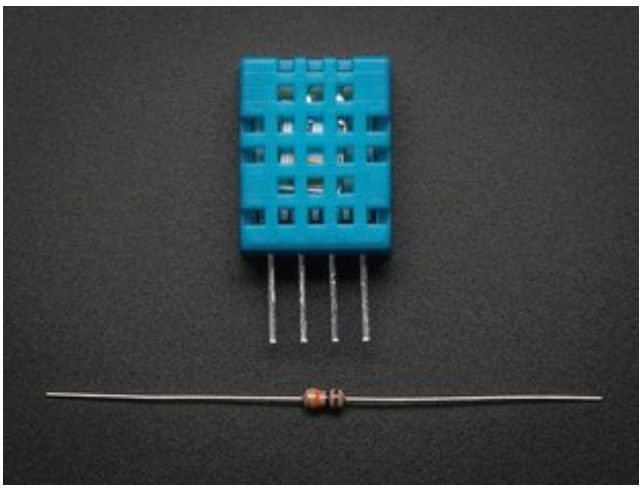
Arduino Lesson 2. LEDs
http://adafru.it/tUE

Going Further

Since Adabox 001 came with a Feather Adalogger we can also save the temperature data to the SD card. The guide for the Feather Adalogger explains how to use to the SD card and has an example program.

Using the SD Card - Feather Adalogger
http://adafru.it/tUF

Another option is to replace the TMP36 with a DHT11 to also read humidity.



DHT11 basic temperature-humidity sensor + extras

PRODUCT ID: 386
The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital...
http://adafru.it/f6N
$5.00
IN STOCK