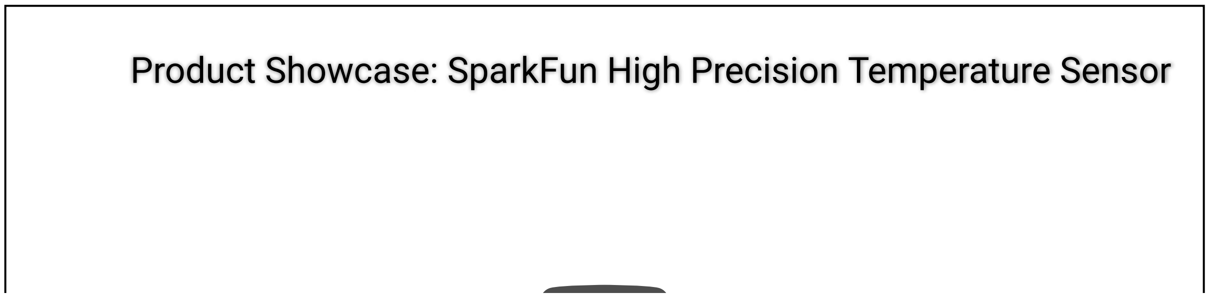
Qwiic TMP117 High Precision Digital Temperature Sensor Hookup Guide

Introduction

The TMP117 is a high precision, digital temperature sensor. What makes the TMP117 stand out is it's ability to be accurate down to ±0.1°C (from -20°C to 50°C). The measurements can also have a resolution of 0.0078°C! This is great for projects that require more stable temperature readings. There's also additional features that come with the TMP117. Some of these features include offsetting the temperature, entering low-power mode, and averaging the readings.

SparkFun High Precision Temperature Sensor - TMP117 (Qwiic)
SEN-15805


Product Showcase: SparkFun High Precision Temperature Sensor

Required Materials

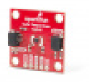
To follow along with this tutorial, you will need the following materials. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

Qwiic TMP117 High Precision Digital Temperature Sensor Wishlist SparkFun Wish List

Qwiic Cable - 100mm
PRT-14427

USB micro-B Cable - 6 Foot
CAB-10215
USB 2.0 type A to micro USB 5-pin. This is a new, smaller connector for USB devices. Micro USB connectors are a…

SparkFun RedBoard Qwiic
DEV-15123


SparkFun High Precision Temperature Sensor - TMP117 (Qwiic)
SEN-15805

Suggested Reading

Before continuing on with this tutorial, you may want to familiarize yourself with some of these topics if they're unfamiliar to you. If you aren't familiar with the Qwiic system, we recommend reading here for an overview.

Qwiic Connect System

If you aren't familiar with the following concepts, we recommend checking out these tutorials before continuing. Make sure to install the appropriate drivers before uploading code. In this case, we'll need to make sure that the CH340 drivers are installed for the RedBoard Qwiic.

I2C
An introduction to I2C, one of the main embedded communications protocols in use today.

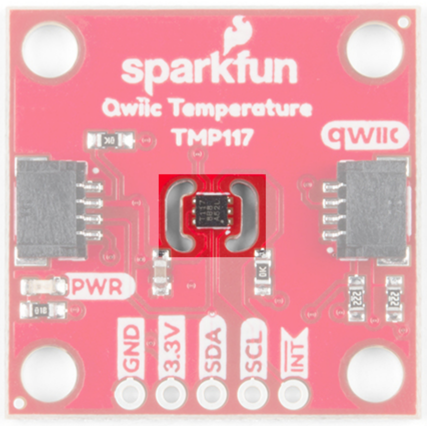
RedBoard Qwiic Hookup Guide
This tutorial covers the basic functionality of the RedBoard Qwiic. This tutorial also covers how to get started blinking an LED and using the Qwiic system.

How to Install CH340 Drivers
How to install CH340 drivers (if you need them) on Windows, Mac OS X, and Linux.

Hardware Overview

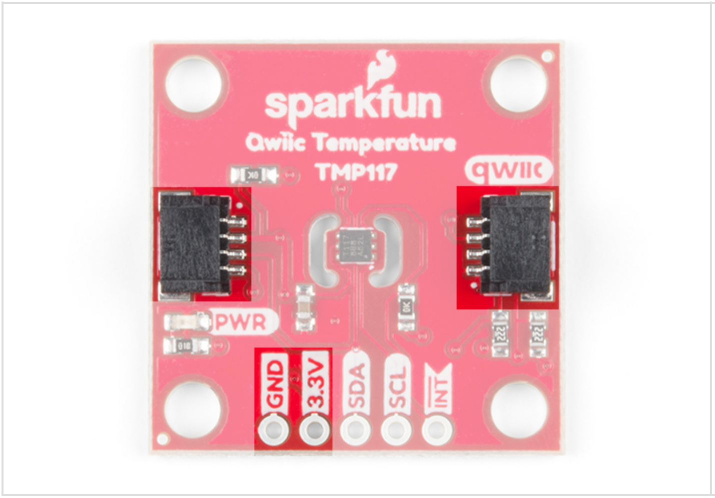
TMP117

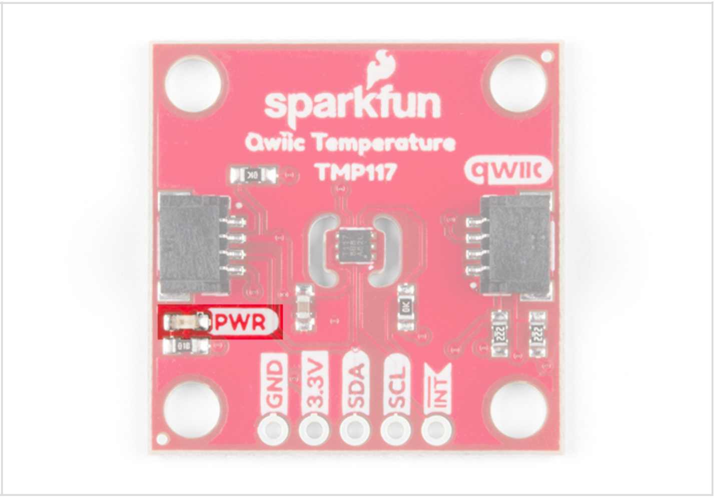
The TMP117 is located on a tiny, isolated island between two slots that are cut into the PCB. This minimizes heat generated from components on the PCB and any errors that may result when taking temperature readings.



Power

The sensor has a low power consumption and is operates between 1.8V to 5.5V. To power the sensor on the breakout board, it utilizes **3.3V** from the Qwiic connector. Depending on your application, you can also connect a power via the plated through holes for 3.3V and GND. The corresponding PWR LED will light up to indicate if the sensor is being powered.

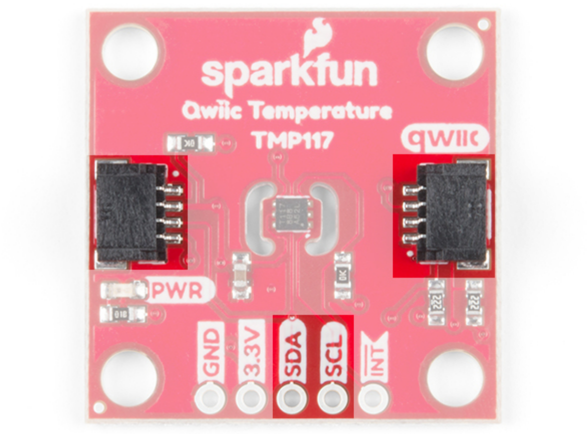




Note: For accurate readings, make sure to use the minimal supply voltage acceptable for the system.

I^{2}C Pins

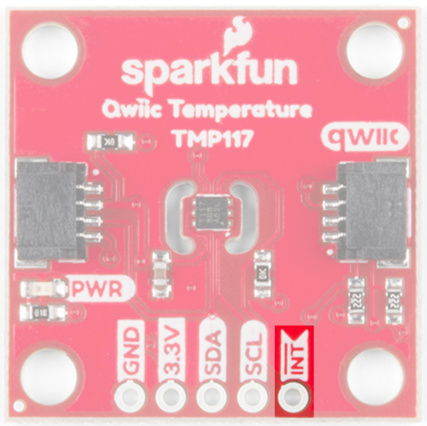
To communicate with the sensor, you will need an I^{2}C bus. The Qwiic system makes it easy to connect the TMP117 to your projects via the Qwiic connector. You can add a Qwiic cable between the sensor and development board to start experimenting with the sensor in your projects. Depending on your application, you can also connect to the I^{2}C pins via the plated through holes for SDA and SCL.



Note: For accurate readings, make sure to avoid heavy bypass traffic on the I^{2}C bus and use the highest available communication speed. That is, try to not add too many I^{2}C devices talking on the same bus or stretch your clock signal.

Interrupt Pin (a.k.a. Alert Pin)

The $\overline{\text{INT}}$ pin on the board is connected to the TMP117's "alert" pin. When the pin is active, it will be pulled LOW by default. If the TMP117's temperature limits are configured and the sensor exceeds the values, the alert pin will pulled LOW.



Jumpers

The board has a few jumper pads on the board. If you have not worked with jumper pads, make sure to check out the tutorial on "How to Work with Jumper Pads and PCB Traces" for more information.

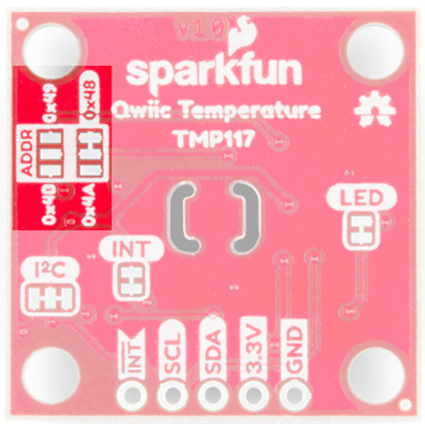
How to Work with Jumper Pads and PCB Traces

APRIL 2, 2018

Handling PCB jumper pads and traces is an essential skill. Learn how to cut a PCB trace, add a solder jumper between pads to reroute connections, and repair a trace with the green wire method if a trace is damaged.

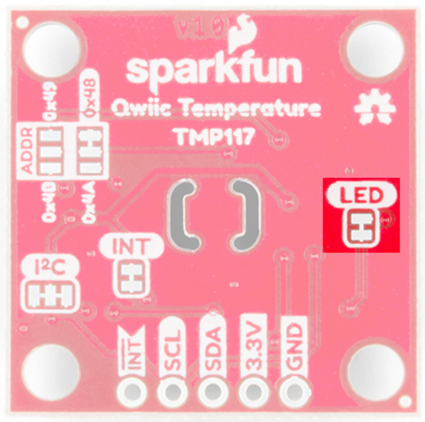
Address Select

The default address of the board is **0x48**. If you need to adjust the address of the sensor, you can cut the trace connecting to the default address and add a solder jumper to the respective pads to change the address to *0x49*, *0x4A*, or *0x4B*.



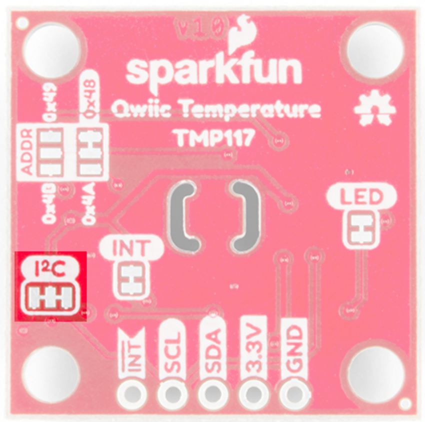
LED

If you need to disable the PWR LED to make the board inconspicuous, conserve more power, or ensure that you are minimizing any heat generated from the LED, you can cut the jumper connecting to the LED.

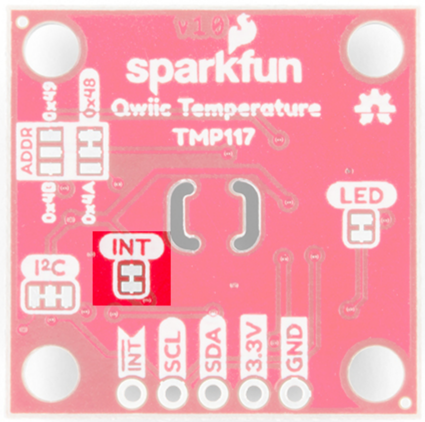


Pull-Up Resistors

The board also includes jumpers to disable the pull-up resistors on the I^{2}C bus line. If you are using a few I^{2}C devices on the same bus that already have pull-up resistors on their respective boards, you may want to cut the jumpers to disconnect. This is for special use cases when daisy chaining about 7x I^{2}C devices on the same bus. Keep in mind that you will want to avoid heavy bypass traffic on the I^{2}C bus if you are trying to take accurate readings.

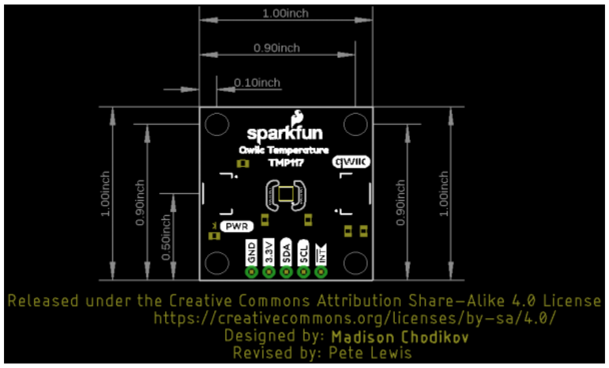


There is also a pull-up resistor on the $\overline{\text{INT}}$ pin if you need to disable the it as well.



Dimensions

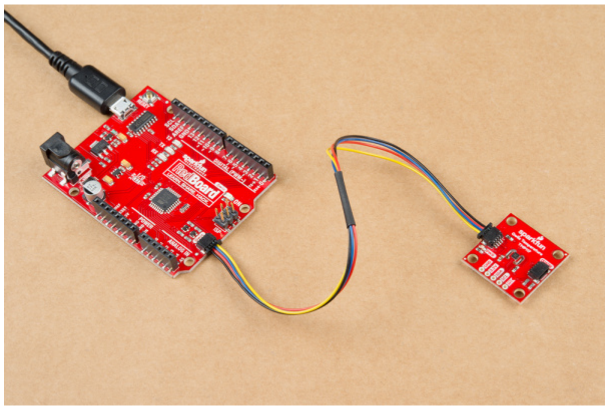
The board uses the standard Qwiic 1.0"x1.0" board size with four mounting holes.


1.00inch
0.90inch
0.10inch
1.00inch
0.90inch
0.50inch
0.90inch
1.00inch
sparkfun
Qwiic Temperature
TMP117
qwiic
PWR
GND
3.3V
SDA
SCL
INT
Released under the Creative Commons Attribution Share-Alike 4.0 License
https://creativecommons.org/licenses/by-sa/4.0/
Designed by: Madison Chodikov
Revised by: Pete Lewis

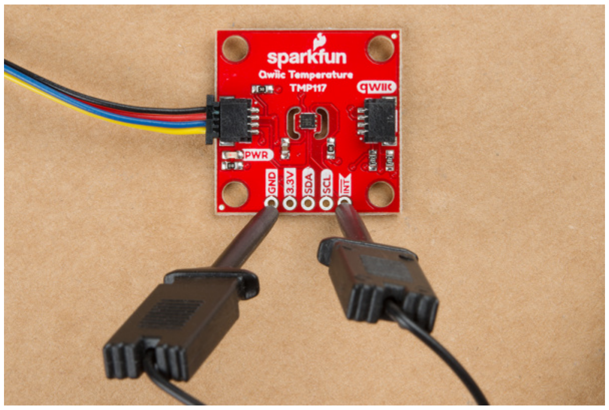
Note: For accurate readings, make sure to place the sensor horizontally and out of any airflow when mounting the sensor to an application.

Hardware Hookup

This board is an I^{2}C based board and so we've included a Qwiic connector on the breakout board. Hooking the sensor up is easy. Just plug one end of the Qwiic cable into the Qwiic TMP117 and the other to your development board. In this case, it's the RedBoard Qwiic. You'll be ready to upload a sketch and start reading temperature values. It seems like it's too easy too use, but that's why we made it that way!



If you need to temporarily access the pins via the standard breadboard spaced 0.1" PTH pads, you can connect some IC hooks to the through holes. In the image below, the interrupt pin needed to be measured with a multimeter so two IC hooks were connected to the board as a temporary connection. For a secure connection, you'll need to solder headers to the PTH pads.



Arduino Library

Note: If this is your first time using Arduino, please review our tutorial on installing the Arduino IDE. If you have not previously installed an Arduino library, please check out our installation guide.

SparkFun has written a library to control the Qwiic TMP117. You can obtain these libraries through the Arduino Library Manager. By searching for **TMP117**, you will get two results. Click on the one written by **SparkFun Electronics** and you should be able to install the latest version. If you prefer downloading the libraries manually you can grab them from the GitHub repository.

DOWNLOAD THE SPARKFUN TMP117 ARDUINO LIBRARY (ZIP)

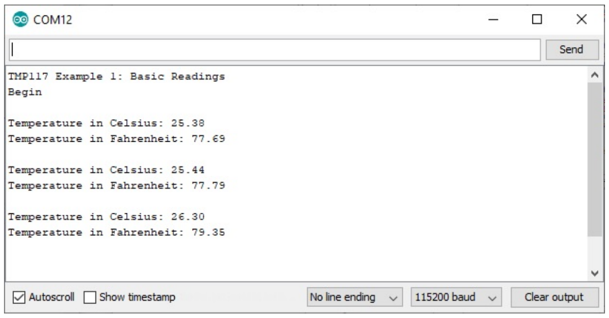
Arduino Example Code

Example 1: Basic Readings

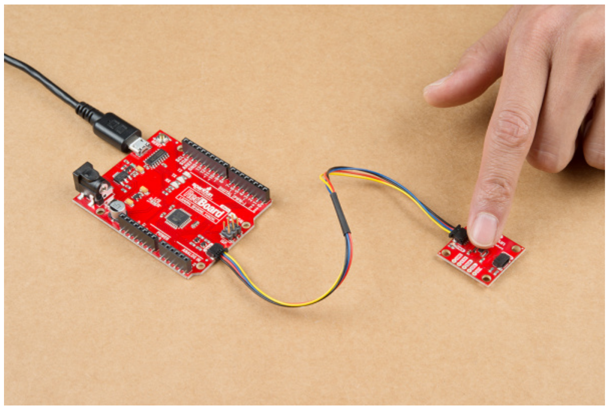
This example prints out the temperature in degrees Celsius and Fahrenheit. The beginning of the code is pretty much the same for the examples. The code initializes the I^{2}C bus to communicate with the TMP117 and serial UART to pass the data to our Arduino serial monitor. We'll want to set the bus to fast mode as recommended by the datasheet. Once initialized, we'll check to see if the address matches the TMP117. By default, this is **0x48**. If it matches, we'll continue running the sketch.

This particular example will check to see if the TMP117 measured and averaged the temperature readings. If there is data ready, the TMP117 will notify us from the configuration register. When ready, we'll read the temperature registers and output it out to the Arduino serial monitor.

If you have not already, open the example up from your Arduino menu and clicking on the example: **File** > **Examples** > **SparkFun_High_Precision_Temperature_Sensor_TMP117_Qwiic** > **Example1_BasicReadings**. Select your board (in this case **Arduino/Genuino Uno** and COM port that the board enumerated to. Hit the upload button. Open the serial monitor at **115200** baud to start reading the temperature readings. You should see something like the output below.


```
TMP117 Example 1: Basic Readings
Begin

Temperature in Celsius: 25.38
Temperature in Fahrenheit: 77.69

Temperature in Celsius: 25.44
Temperature in Fahrenheit: 77.79

Temperature in Celsius: 26.30
Temperature in Fahrenheit: 79.35
```

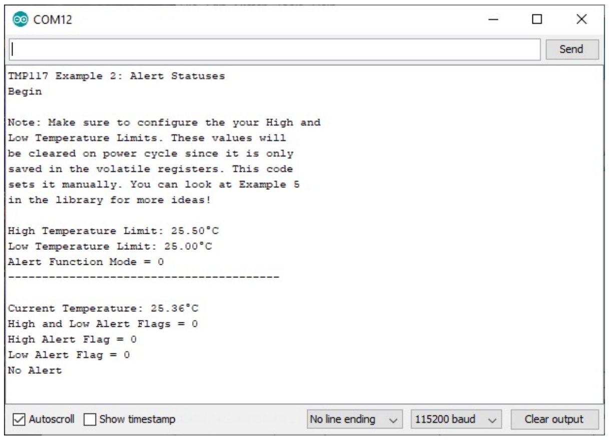
Try heating the sensor with your finger or lightly breathe some air across the sensor to watch the temperature values change! The third reading from the output shows the temperature rising after placing my finger on the temperature sensor. Remember, the sensor is taking a few readings and averaging them together before we are able see the output. Make sure to be patient with the output. If you need to output data faster, you can configure the conversion averaging times in example 6



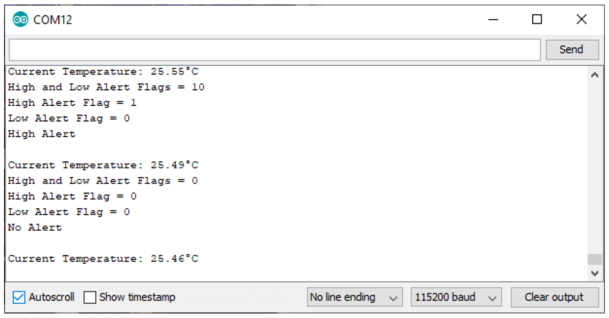
Example 2: Alert Status

This example configures the serial I^{2}C and UART like the first example. We'll set alert function mode and temperature limits in volatile memory. When ready, we'll check to see if the temperature exceeds the high or low limits. By default, the alert function mode is set to alert mode. We set it again just in case. If we above the high limit, the high alert flag will be raised and we will output `high alert`.. If we are below the low limit, the high alert flag will be raised and we will output the `low alert`. Otherwise, we will have message indicating that there is `no alert`.

If you have not already, open the example up from the Arduino menu and clicking on the example: **File** > **Examples** > **SparkFun_High_Precision_Temperature_Sensor_TMP117_Qwiic** > **Example2_AlertStatuses**. Select your board (in this case **Arduino/Genuino Uno** and COM port that the board enumerated to. Hit the upload button. Open the serial monitor at **115200** baud. You should see something like the output below.

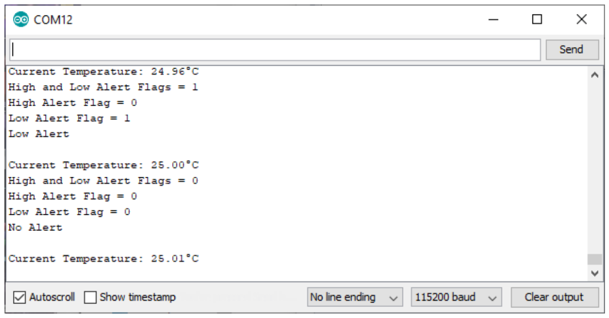

```
TMP117 Example 2: Alert Statuses
Begin

Note: Make sure to configure the your High and
Low Temperature Limits. These values will
be cleared on power cycle since it is only
saved in the volatile registers. This code
sets it manually. You can look at Example 5
in the library for more ideas!

High Temperature Limit: 25.50°C
Low Temperature Limit: 25.00°C
Alert Function Mode = 0
----------------------------------------

Current Temperature: 25.36°C
High and Low Alert Flags = 0
High Alert Flag = 0
Low Alert Flag = 0
No Alert
```

The current temperature reading at the time was `25.36°C`. This is within the boundary of our temperature limits that we set so our alert flag is `0`. You can adjust the limits as necessary but we made the temperature window narrow to easily test the functionality using the heat from our body. When we are outside the boundary limits, the alert flags will be set to `1` with their respective limits. As the temperature changes and is within our boundaries, the flag will be set back to `0`.



```
Current Temperature: 25.55°C
High and Low Alert Flags = 10
High Alert Flag = 1
Low Alert Flag = 0
High Alert

Current Temperature: 25.49°C
High and Low Alert Flags = 0
High Alert Flag = 0
Low Alert Flag = 0
No Alert

Current Temperature: 25.46°C
```

High Alert Flag Raised to 1

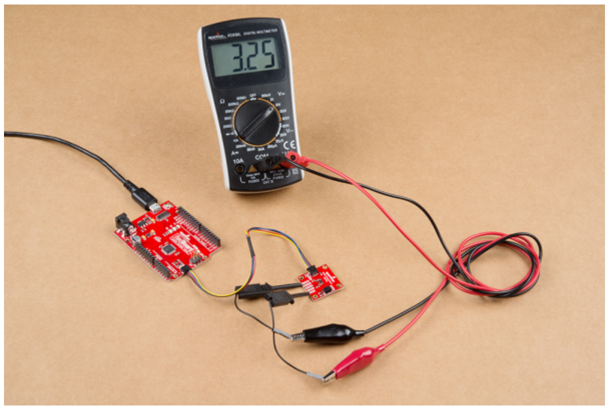


```
Current Temperature: 24.96°C
High and Low Alert Flags = 1
High Alert Flag = 0
Low Alert Flag = 1
Low Alert

Current Temperature: 25.00°C
High and Low Alert Flags = 0
High Alert Flag = 0
Low Alert Flag = 0
No Alert

Current Temperature: 25.01°C
```

Low Alert Flag Raised to 1

Click on images for a closer view of the outputs.

Note: The example code checks for the alert flag from the TMP117's configuration register. As the alert flag is raised, the alert pin will also be active. If you have a multimeter connected to the $\overline{\text{INT}}$ and GND pins, you will notice that it will change from a HIGH (~3.3V) to a LOW (0V) whenever the the high or low alert flags are raised as well.



If you look at *Figure 18 of the Alert Mode Timing Diagram* on page 15 of the datasheet, the serial output and the voltage of the pin matches what we expect from the TMP117!

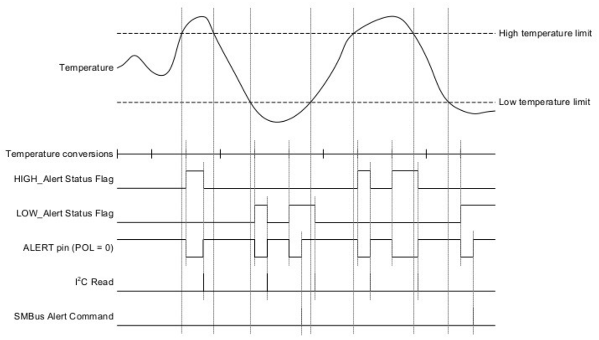
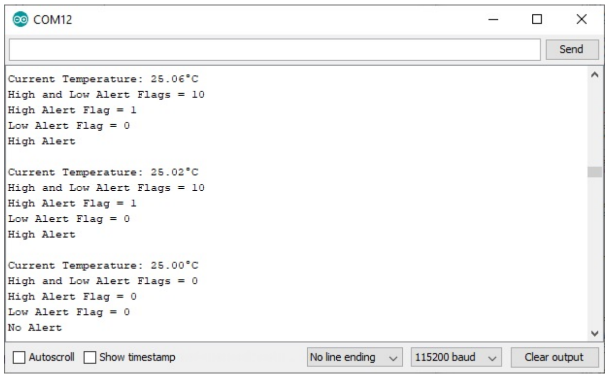


Figure 18. Alert Mode Timing Diagram

The example above shows what happens when we are in "alert mode." You'll notice that the flags and pin resets itself every time there is a read of the configuration register. If you modify the line that sets the alert function mode to "therm mode", the the flags behave slightly differently.

```
//.
//. somewhere in Example 2's setup(){} function
//.

  /*Note: Set to alert or therm mode. To change, simply adjust
  add or remove a `//` to the line.*/
  //sensor.setAlertFunctionMode(0);//set to alert mode
  sensor.setAlertFunctionMode(1);//set to therm mode
```

If the current temperature exceeds the high temperature limit, you'll notice that the alert flag will stay at 1 until the temperature is below our low temperature limit.



```
Current Temperature: 25.06°C
High and Low Alert Flags = 10
High Alert Flag = 1
Low Alert Flag = 0
High Alert

Current Temperature: 25.02°C
High and Low Alert Flags = 10
High Alert Flag = 1
Low Alert Flag = 0
High Alert

Current Temperature: 25.00°C
High and Low Alert Flags = 0
High Alert Flag = 0
Low Alert Flag = 0
No Alert
```

Note: If you look at *Figure 19 of the Therm Mode Timing Diagram* on page 16 of the datasheet, the serial output and the voltage of the pin matches what we expect from the TMP117!

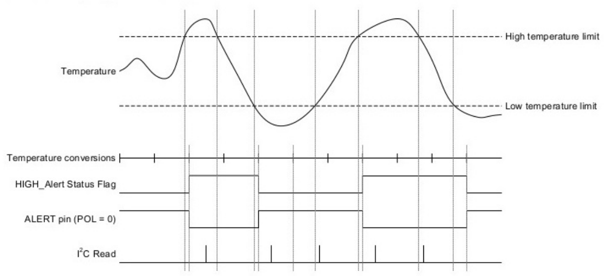
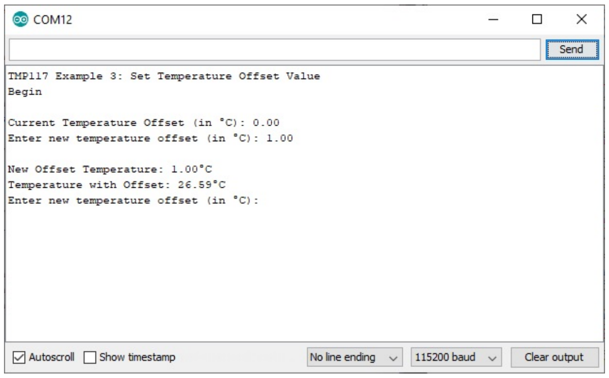


Figure 19. Therm Mode Timing Diagram

Example 3: Set Offset Temperature

In this example, you can adjust the offset temperature of the TMP117. This is useful if you need to adjust the temperature readings based on your system in case the location you are measuring is warmer or colder than the surrounding environment.

If you have not already, open the example up from the Arduino menu and clicking on the example: **File** > **Examples** > **SparkFun_High_Precision_Temperature_Sensor_TMP117_Qwiic** > **Example3_SetOffsetTemperatureValue**. Select your board (in this case **Arduino/Genuino Uno** and COM port that the board enumerated to. Hit the upload button. Open the serial monitor at **115200** baud. You will be asked to enter a temperature offset. After entering a value and hitting your `ENTER` key, the TMP117 will save the offset in its volatile memory.

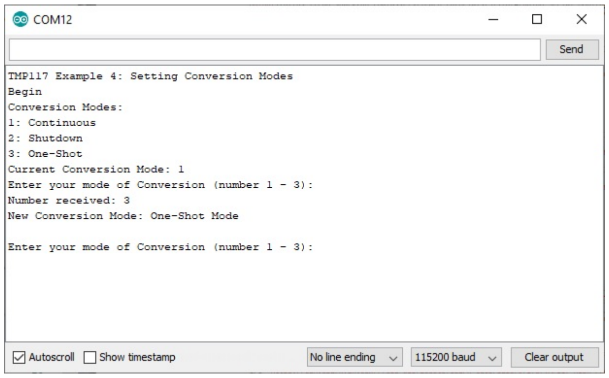


```
TMP117 Example 3: Set Temperature Offset Value
Begin

Current Temperature Offset (in °C): 0.00
Enter new temperature offset (in °C): 1.00

New Offset Temperature: 1.00°C
Temperature with Offset: 26.59°C
Enter new temperature offset (in °C):
```

Example 4: Set Conversion Mode

You can change the continuous conversion mode. The default is in "continuous conversion mode". You can set it **one-shot** or **shutdown mode** for low power applications. In one-shot mode, the TMP117 will take one temperature reading. After one temperature reading, the sensor will enter low power shutdown mode. When the TMP117 is set to shutdown mode, all temperature conversions are aborted and the TMP117 will enter low power shutdown mode.

If you have not already, open the example up from the Arduino menu and clicking on the example: **File** > **Examples** > **SparkFun_High_Precision_Temperature_Sensor_TMP117_Qwiic** > **Example4_SetConversionMode**. Select your board (in this case **Arduino/Genuino Uno** and COM port that the board enumerated to. Hit the upload button. Open the serial monitor at **115200** baud. You will be asked to enter a temperature offset. After entering a value and hitting your `ENTER` key, the TMP117 will save the offset in its volatile memory.



```
TMP117 Example 4: Setting Conversion Modes
Begin
Conversion Modes:
1: Continuous
2: Shutdown
3: One-Shot
Current Conversion Mode: 1
Enter your mode of Conversion (number 1 - 3):
Number received: 3
New Conversion Mode: One-Shot Mode

Enter your mode of Conversion (number 1 - 3):
```

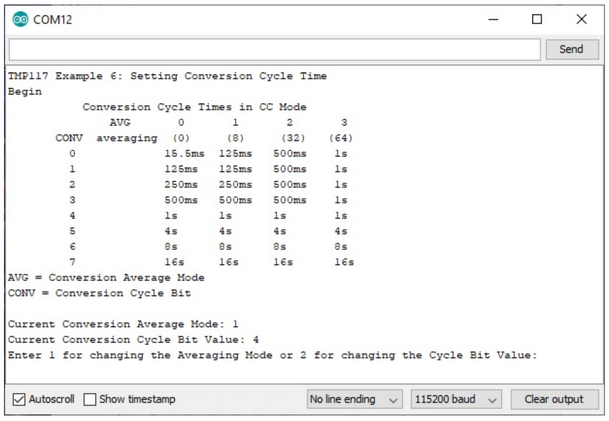
Example 6: Set Conversion Cycle Time

Note: We're going to skip ahead to example 6 in the library. Example 5 provides a prompt to set the alert mode and temperature mode. Since we know how to do that from example 2, we'll go over the next example.

In this example, you can adjust the conversion cycle bit and the conversion averaging mode. You can reduce the amount of noise by adjusting these values. By **decreasing** the cycle average mode and conversion cycle bit, you will get faster readings but the data will start to become more noisy. By **increasing** the conversion average mode and conversion cycle bit, you will get more stable temperature readings. You will get a slower output because it is averaging the several data points together. However, the readings will be smoother. As stated in the datasheet on page 33 under 8.2.2.1 Noise and Averaging, you'll want higher averaging numbers whenever your "*system environment is noisy (such as when measuring air flow temperatures, power supply fluctuations, intensive communication on a serial bus...).*" Here's a table of the conversion cycle times from page 27 of the datasheet.

	AVG = 0 (0 Conversions)	**AVG = 1 (8 Conversions)**	**AVG = 2 (32 Conversions)**	**AVG = 3 (64 Conversions)**
CONV = 0	15.5 ms	125 ms	500 ms	1 sec
CONV = 1	125 ms	125 ms	500 ms	1 sec
CONV = 2	250 ms	250 ms	500 ms	1 sec
CONV = 3	500 ms	500 ms	500 ms	1 sec
CONV = 4	1 sec	1 sec	1 sec	1 sec
CONV = 5	4 sec	4 sec	4 sec	4 sec
CONV = 6	8 sec	8 sec	8 sec	8 sec
CONV = 7	16 sec	16 sec	16 sec	16 sec

If you have not already, open the example up from the Arduino menu and clicking on the example: **File** > **Examples** > **SparkFun_High_Precision_Temperature_Sensor_TMP117_Qwiic** > **Example6_SetConversionCycleTime**. Select your board (in this case **Arduino/Genuino Uno** and COM port that the board enumerated to. Hit the upload button. Open the serial monitor at **115200** baud.



```
TMP117 Example 6: Setting Conversion Cycle Time
Begin
          Conversion Cycle Times in CC Mode
              AVG       0       1       2       3
       CONV  averaging  (0)     (8)     (32)    (64)
         0              15.5ms  125ms   500ms   1s
         1              125ms   125ms   500ms   1s
         2              250ms   250ms   500ms   1s
         3              500ms   500ms   500ms   1s
         4              1s      1s      1s      1s
         5              4s      4s      4s      4s
         6              8s      8s      8s      8s
         7              16s     16s     16s     16s
AVG = Conversion Average Mode
CONV = Conversion Cycle Bit

Current Conversion Average Mode: 1
Current Conversion Cycle Bit Value: 4
Enter 1 for changing the Averaging Mode or 2 for changing the Cycle Bit Value:
```

More Examples

We only went over 5 of the examples. There's a few more in the library. Make sure to read through the description and comments in the code before trying out the additional examples that were not explained earlier!

GITHUB: SPARKFUN_TMP117_ARDUINO_LIBRARY > EXAMPLES

Resources and Going Further

Now that you've successfully got your Qwiic TMP117 up and running, it's time to incorporate it into your own project! For more information, check out the resources below:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PNG)
- Datasheet (PDF)
- Arduino Library
- GitHub Product Repo
- SFE Product Showcase

Need some inspiration for your next project? Check out some of these related tutorials using temperature sensors:

MAX31855K Thermocouple Breakout Hookup Guide

Learn how to take readings with a k-type thermocouple using the MAX31855K cold-junction-compensated k-type thermocouple-to-digital converter.

LilyPad Temperature Sensor Hookup Guide

How to hook up the LilyPad Temperature Sensor as well as some project ideas and example code.

TMP102 Digital Temperature Sensor Hookup Guide

How to connect and use the SparkFun Digital Temperature Sensor Breakout - TMP102 with an Arduino.

SparkFun Inventor's Kit Experiment Guide - v4.1

The SparkFun Inventor's Kit (SIK) Experiment Guide contains all of the information needed to build all five projects, encompassing 16 circuits, in the latest version of the kit, v4.1.