Temperature Sensor Guide

By Team AJN

Last updated on April 2025

Tested on:

BeagleY-AI (Target): **Debian 12.8**
PC (Host): **Debian 12.8**

Guide Completion Requirements:

- Quick Start Guide
- I2C guide: ADC

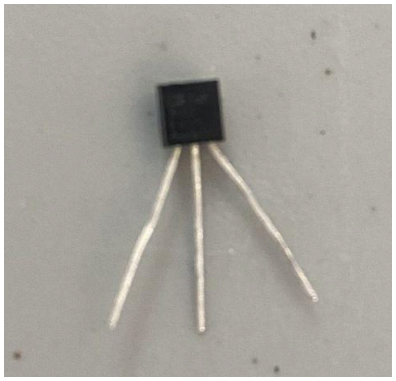
Contents

1: Introduction

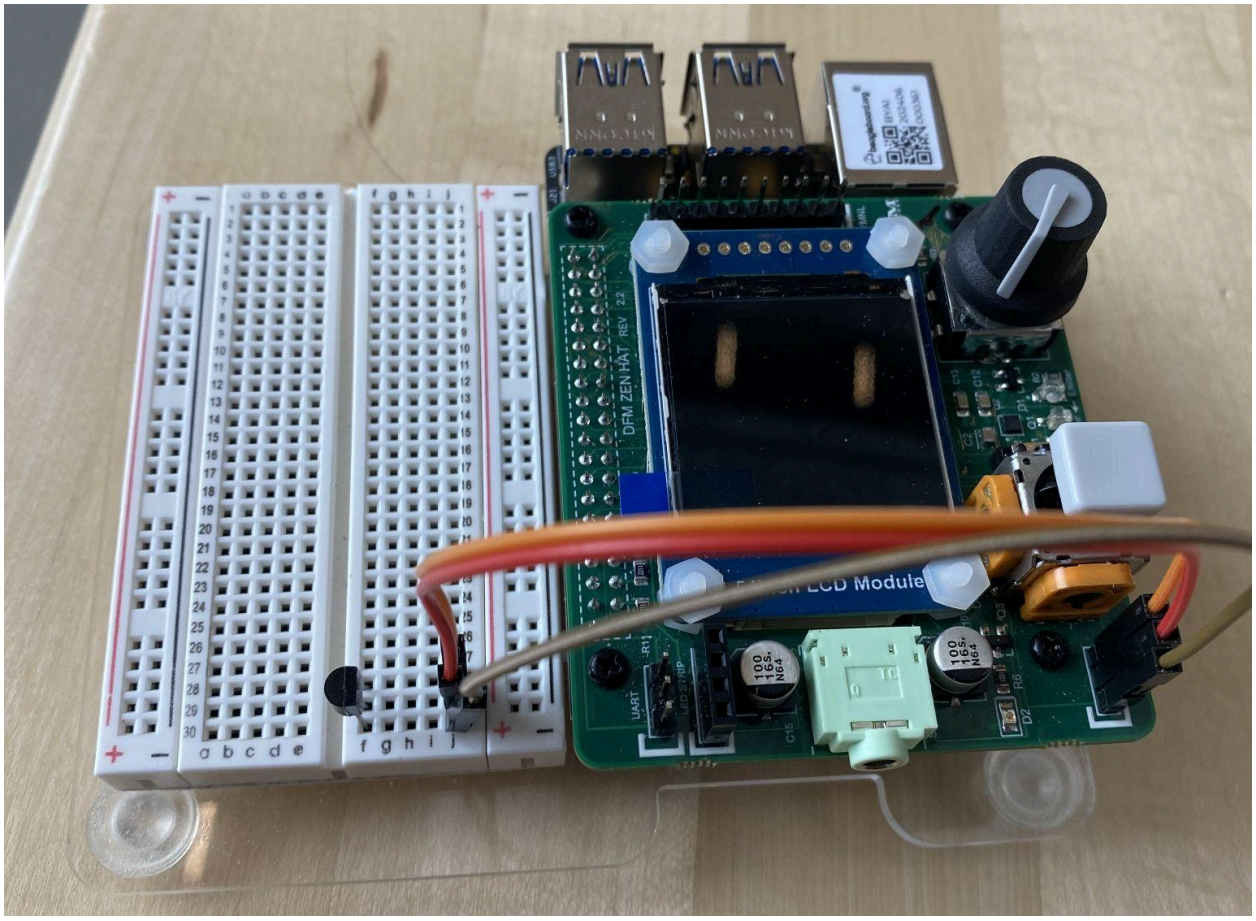
This document serves to guide users to set up their TMP36 temperature sensor to connect to their BeagleY-AI. Sample commands and code is provided in order to read from an I2C value. This guide is based on other TMP36 temperature sensor guides and expands upon the professor's official I2C guide, modified to be compatible with the BeagleY-AI.

2: TMP36 setup guide

The TMP36 temperature sensor has 3 pins. Before setting up, it is essential to NOT connect the pins to the wrong wiring, as this risks both the temperature sensor and the Beagleboard (and potentially the user too!). For this guide, a breadboard is used to connect the sensor with the Beagleboard, with the first 3 rows being used. Images of the temperature sensor being connected with the Beagleboard are shown below to verify correct installation of the sensor.



Above image is the temperature sensor. The left pin connects to a 3.3V power source, the middle pin connects to an ACD pin, and the right pin connects to ground.



How to connect the sensor to the Beagleboard. Please note the direction of the sensor, where the flat side is facing towards the BeagleY-AI.

3: Linux command for readings

Assuming the user has set up I2C readings and is able to ssh into their Beagleboard, a command may be used to continuously sample temperature readings:

```
(byai)$ i2cset -y 1 0x48 1 0x83F2 w
```

If the user wishes to get temperature readings used a command, they may follow up with:

```
(byai)$ i2cget -y 1 0x48 0x00 w
```

Beware that the value printed is given in **least significant byte first** format, like other I2C readings. Note the following values outputted below:

```
(byai)$ i2cget -y 1 0x48 0x00 w
(byai)$ 0xfd18
(byai)$ i2cget -y 1 0x48 0x00 w
(byai)$ 0x0219
```

```
(byai)$ i2cget -y 1 0x48 0x00 w
(byai)$ 0xa119
```

If the user touches the sensor and the value changes, then it is a sign that the sensor works as intended. If the value does not change, check if the sensor is properly connected according to the images shown. The sensor may be facing the wrong direction. If i2cget is an unknown command, then check if i2c-tools is installed on the Beagleboard.

If the user wants to convert their readings into a temperature value in celsius, they may use these formulas:

raw value * 2.5 / 4095 * 3.3
The formula above converts a raw value into a voltage.

((voltage * 1000.0) - 500.0) / 10.0
The formula above converts the resulting voltage into a temperature in celsius.

4: Sample C code

Assuming the user has followed the I2C guide's code, they can get the following temperature readings by using the following code snippets:

```
uint16_t get_temperature_levels() {
   uint16_t raw_value = read_i2c_reg16(i2c_sensor_desc, REG_DATA);
   uint16_t value = ((raw_value & 0xFF) << 8) | ((raw_value & 0xFF00) >> 8);
   value  = value >> 4;
   return value;
}
```

get_temperature_levels() allows the user to retrieve the raw values of the sensor. Note that due to the value being least significant byte first, the value is adjusted accordingly.

```
#define A2C_VOLTAGE_REF_V 3.3
#define A2C_MAX_READING 4095

double get_temperature_in_celsius() {
   uint16_t raw_reading = get_temperature_levels();
   double voltage = ((double)raw_reading * 2.5 / A2C_MAX_READING) * A2C_VOLTAGE_REF_V;
   double temperature = ((voltage * 1000.0) - 500.0) / 10.0;

   return temperature;
}
```

get_temperature_in_celsius() allows the user to convert their raw values into temperature readings in celsius.

5: References

Previous TMP36 Student Guide:
https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt433/links/files/2022-student-howtos/TMP36TemperatureSensor.pdf

Brian Fraser's I2C Guide:
https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt433/guides/files_byai/I2CGuide.pdf