TI Designs
Wireless Thermocouple Sensor Transmitter DevPack for SensorTag

TI Designs

The TIDA-00650 reference design shows how to build a wireless thermocouple-based temperature transmitter. The design is in the DevPack form factor, which can be used in combination with TI's SensorTag. This allows realizing a wireless link with *Bluetooth*® Smart® or other wireless technologies (Zigbee®, Wi-Fi, sub 1GHz). The 24-bit ΔΣ sensor front-end used allows this design to operate over the –40°C to 85°C temperature range.

Design Resources

TIDA-00650	Design Folder
ADS1220	Product Folder
TS5A3159	Product Folder
SensorTag	Product Folder
MSP430FR5969	Product Folder

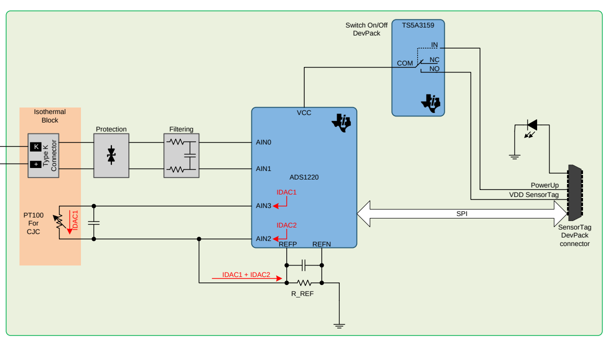
TI E2E™ Community — ASK Our E2E Experts

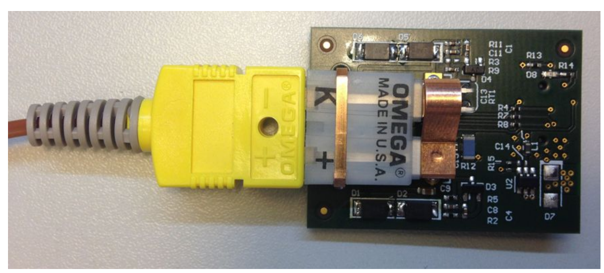
Design Features

- Wireless Thermocouple Sensor Transmitter
- 24-Bit ΔΣ Sensor AFE
- Bluetooth Smart Interface
- Type-K TC Sensor Implementation
- PT100 Sensor for CJC
- TC Temperature Error: 0.8°C Across TC and Ambient Temperature Range
- TC Temperature Range: –270°C to 1372°C
- Ambient Temperature Range: –40°C to 85°C

Featured Applications

- Isolated Sensors and Field Transmitters
- Factory Automation and Process Control
- Building Automation
- Portable Instrumentation





An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 System Description

1.1 System Introduction

This TI Design is a thermocouple-based sensor transmitter front-end in the DevPack form factor, which can be connected to the SimpleLink™ Bluetooth Smart/Multi-Standard SensorTag. With this wireless solution, a thermocouple temperature transmitter can be realized in an isolated fashion.

A standard Type-K thermocouple can directly be connected. Either the PT100 Sensor or the internal ADC temperature sensor acts as the cold-junction compensation (CJC). Both values are measured with an ADC and provide the digital converted signal to the SensorTag.

1.2 Key System Specification

Table 1. Key System Specifications

PARAMETER	SPECIFICATION AND FEATURES
Sensor type	Type-K thermocouple (all with firmware update)
Thermocouple temperature range	–200°C to 1372°C
CJC	RTD PT100 (optional: internal temperature sensor)
CJC temperature range	–40°C to 85°C
Power supply voltage range	Nominal: Coin cell from SensorTag Max limits (ADS1220): 2.3 to 5.5 V
Surge transient immunity	Designed to meet IEC 61000-4
Interface connector	DevPack connector for SensorTag

2 Block Diagram

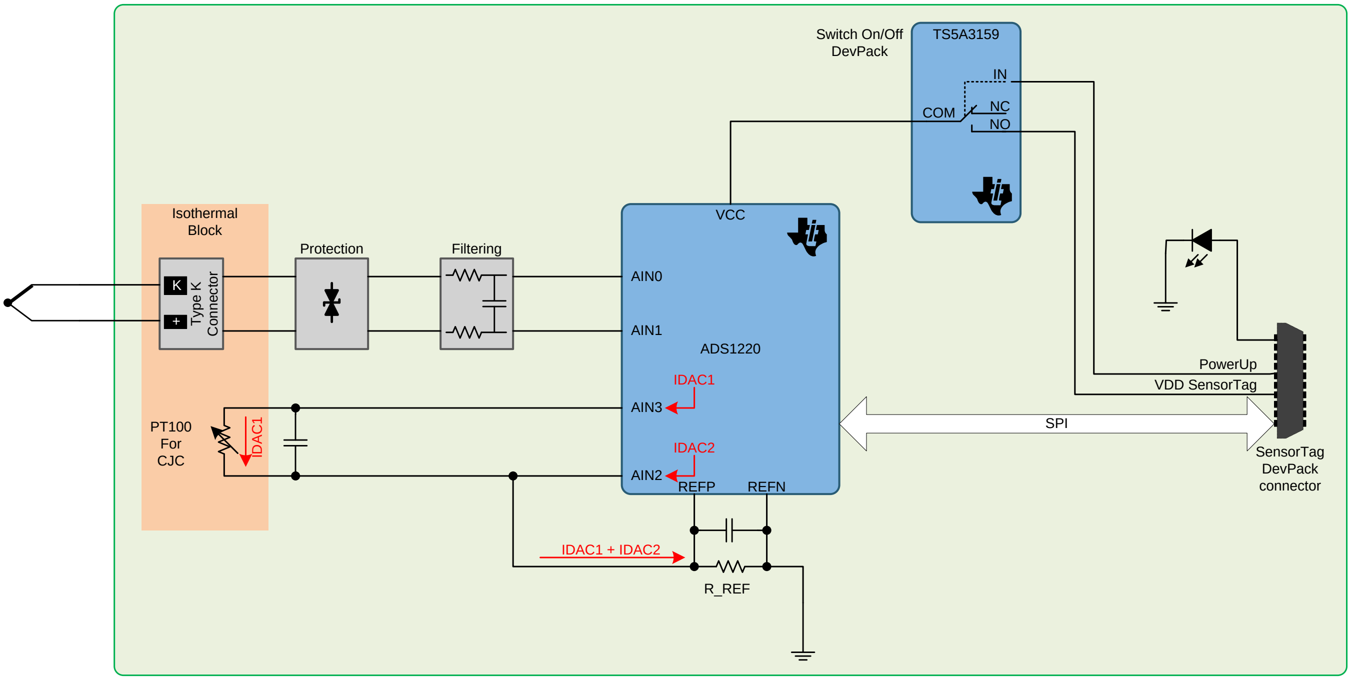


Figure 1. TIDA-00650 Block Diagram

2.1 Highlighted Products

2.1.1 ADS1220

The ADS1220 is a precision, 24-bit, analog-to-digital converter (ADC) that offers many integrated features to reduce system cost and component count in applications measuring small sensor signals. The device features two differential or four single-ended inputs through a flexible input multiplexer (MUX), a low-noise, programmable gain amplifier (PGA), two programmable excitation current sources, a voltage reference, an oscillator, a low-side switch, and a precision temperature sensor.

The device can perform conversions at data rates up to 2000 samples per second (SPS) with single-cycle settling. At 20 SPS, the digital filter offers simultaneous 50-Hz and 60-Hz rejection for noisy industrial applications. The internal PGA offers gains up to 128 V/V. This PGA makes the ADS1220 ideally-suited for applications measuring small sensor signals, such as resistance temperature detectors (RTDs), thermocouples, thermistors, and resistive bridge sensors. The device supports measurements of pseudo- or fully-differential signals when using the PGA. Alternatively, the device can be configured to bypass the internal PGA while still providing high input impedance and gains up to 4 V/V, allowing for single-ended measurements.

Power consumption is as low as 120 µA when operating in duty-cycle mode with the PGA disabled. The ADS1220 is offered in a leadless VQFN-16 or a TSSOP-16 package and is specified over a temperature range of –40°C to 125°C.

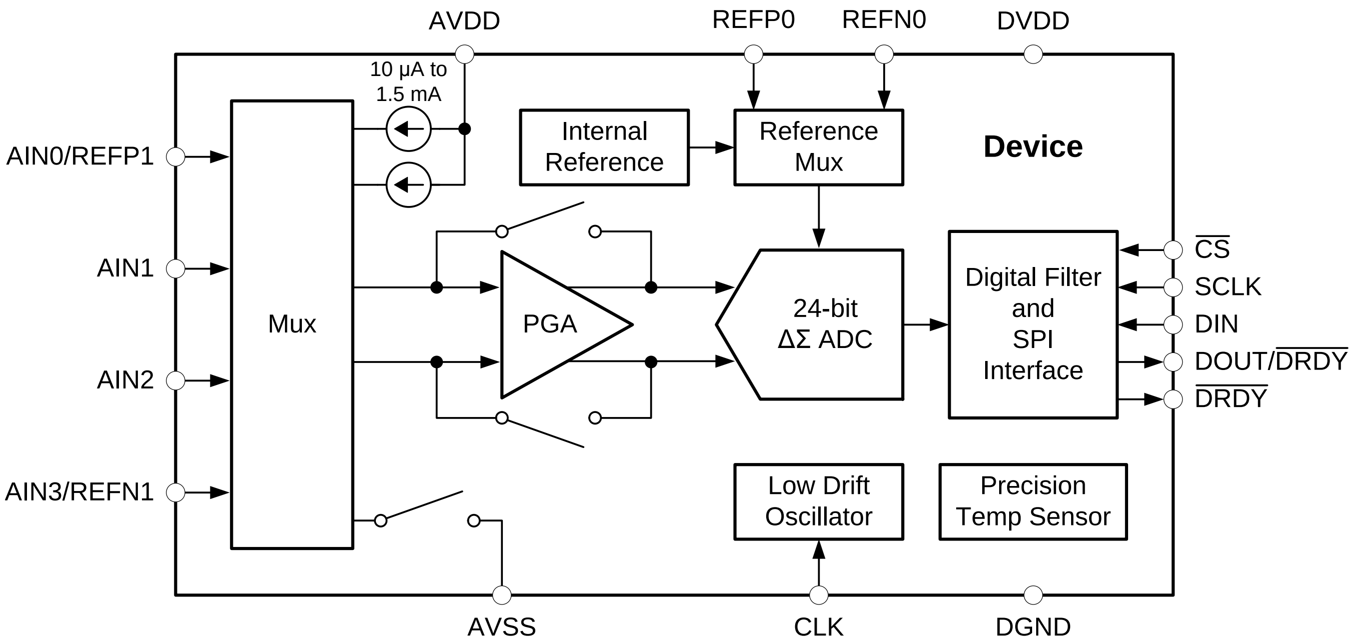


Figure 2. ADS1220 Functional Block Diagram

Features:

- Low current consumption: As low as 120 µA (typ) in Duty Cycle Mode
- Wide supply range: 2.3 to 5.5 V
- Programmable gain: 1 V/V to 128 V/V
- Programmable data rates: Up to 2 kSPS
- Up to 20-bit effective resolution
- Simultaneous 50-Hz and 60-Hz rejection at 20 SPS with single-cycle settling digital filter
- Two differential or four single-ended inputs
- Dual matched programmable current sources: 10 µA to 1.5 mA
- Internal 2.048-V reference: 5 ppm/°C (typ) drift
- Internal 2% accurate oscillator
- Internal temperature sensor: 0.5°C (typ) accuracy
- SPI-compatible interface (Mode 1)
- Package: 3.5×3.5×0.9-mm VQFN

2.1.2 TS5A3159

The TS5A3159 device is a single-pole double-throw (SPDT) analog switch that is designed to operate from 1.65 to 5.5 V. The device offers a low ON-state resistance and an excellent ON-state resistance matching, with the break-before-make feature to prevent signal distortion during the transferring of a signal from one channel to another. The device has excellent total harmonic distortion (THD) performance and consumes very low power. These features make this device suitable for portable audio applications.

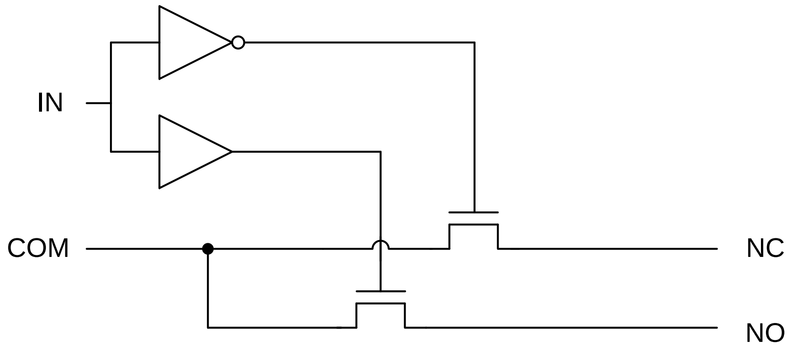


Figure 3. TS5A3159 Block Diagram

Features:

- Specified break-before-make switching
- Low ON-state resistance (1 Ω)
- Control inputs are 5-V tolerant
- Low charge injection
- Excellent ON-resistance matching
- Low total harmonic distortion
- 1.65-V to 5.5-V single-supply operation
- Latch-up performance exceeds 100 mA per JESD 78, class II
- ESD performance tested per JESD 22
  - 2000-V human-body model (A114-B, Class II)
  - 1000-V charged-device model (C101)

3 System Design Theory

The general thermocouple theory, as well as the protection and filtering of the ADS1220, is described in detail in the TI Design TIDA-00168. The few adjustments to this TI Design are described in this document and focuses on the test setup and the test results.

3.1 Thermocouple Channel (TC)

This TI Design uses a Type-K thermocouple for calculation and testing. According to ITS-90 [1], Type-K thermocouples are specified with polynomials from –270°C to 1372°C for a reference temperature (cold-junction temperature) of 0°C. The corresponding voltage range, based on the Seebeck-Effect [3] can be calculated either with the polynomials and the Seebeck coefficients [1] or with the help of available online calculators or look-up tables. With the online calculator [4], the voltage range for a Type-K thermocouple is shown in Table 2.

Table 2. Equivalent Type-K TC Voltage for Minimum and Maximum Temperature Range

TEMPERATURE (°C)	TYPE-K VOLTAGE FOR 0°C CJC (mV)
–270	–6.457738
1372	54.886364

Because the thermocouple does not require an excitation current, the internal IDAC1 and IDAC2 are not required, meaning a ratiometric measurement is not available. Therefore, the internal reference voltage V_{REF} of ADS1220 is used, which is 2.048 V.

3.1.1 PGA Gain for TC

To achieve the best resolution, use the entire ADC full-scale range (FSR); as a result, Equation 1 provides the PGA gain (PGA_{GAIN}) to get close to the FSR. With V_{REF} = 2.048 V and the maximum TC voltage of 54.886 mV:

$$PGA_{GAIN} = \frac{V_{REF}}{V_{TC,max}} = \frac{2.048\ V}{54.886\ mV} = 37.313\ V/V \quad (1)$$

The closest PGA gain setting without saturating the ADC is 32.

3.2 CJC

For CJC, the PT100 is used in a ratiometric measurement technique to eliminate errors from the excitation current. Therefore, an external precision resistor R_{REF} is placed across the ADCs REFP and REFN inputs to generate the reference voltage V_{REF} for the ADC. Figure 4 shows the simplified circuitry. The RTD is connected to AIN2 and AIN3 of the ADS1220. It is configured to source current IDAC1 at terminal AIN3 and IDAC2 at terminal AIN2. Both are set to 250 µA. IDAC1 develops across the PT100 element a voltage according to its resistor value, which depends on the temperature. The sum of IDAC1 and IDAC2 flows through R_{REF} and generates the external reference voltage for the ADC.

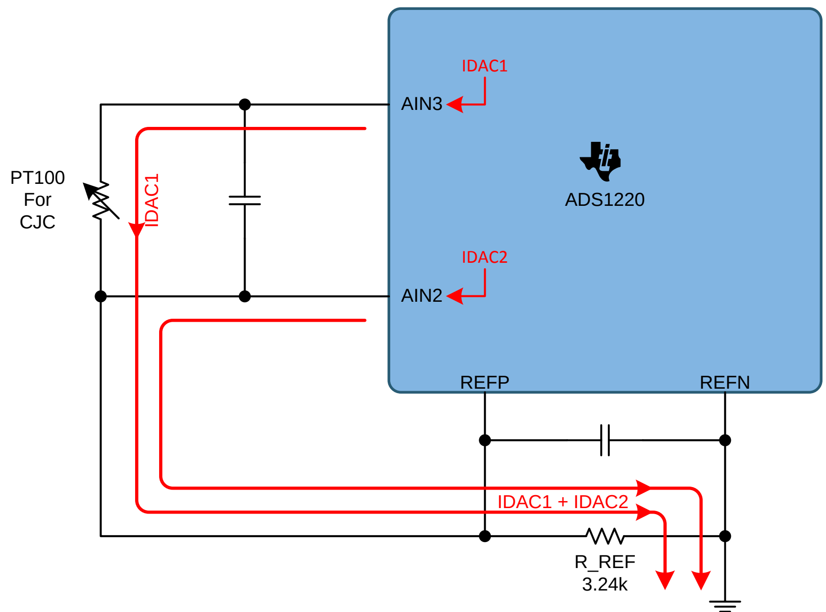


Figure 4. CJC Diagram

With Equation 2, Equation 3, and Equation 4, the resulting resistor value R_{PT100} can be calculated from the measured ADCCODE at terminals AIN3 and AIN2 and the know reference resistor R_{REF}. The excitation current of I_{DAC1} and I_{DAC2} cancel out. The accuracy is determined by R_{REF}; therefore, this reference resistor should be a precision resistor. In this TI Design, R_{REF} has a value of 3.24 kΩ with a tolerance of 0.1% and a temperature coefficient of 10 ppm/°C . The latter is important to have less variation across the ambient temperature.

$$V_{REF} = R_{REF} \times \left(I_{DAC1} + I_{DAC2}\right) = R_{REF} \times 2 \times I_{DAC1} \quad (2)$$

$$V_{PT100} = \frac{ADCCODE}{2^{23}} \times \frac{V_{REF}}{PGA_{GAIN}} = \frac{ADCCODE}{2^{23}} \times \frac{R_{REF} \times 2 \times I_{DAC1}}{PGA_{GAIN}} \quad (3)$$

$$R_{PT100} = \frac{V_{PT100}}{I_{DAC1}} = \frac{ADCCODE}{2^{23}} \times \frac{R_{REF} \times 2 \times I_{DAC1}}{PGA_{GAIN} \times I_{DAC1}} = \frac{ADCCODE}{2^{23}} \times \frac{R_{REF} \times 2}{PGA_{GAIN}} \quad (4)$$

In Equation 2, the current through R_{REF} given by I_{DAC1} + I_{DAC2} was replaced with 2 × IDAC1 because IDAC1 and IDAC2 always have the same value. There is obviously a small matching error between the two current sources. This error can be removed when only one excitation current (here: I_{DAC1}) is being used. Plus, it will reduce the current consumption. In this case, only I_{DAC1} develops the reference voltage across R_{REF}. This decreases the external voltage reference by a factor of 2 while keeping R_{REF} and I_{DAC1}, or it can be compensated by either using I_{DAC1} or R_{REF} twice.

Since the temperature of the cold junction is limited to the operating temperature of the design, the entire range of the PT100 sensor is not used. Even though this TI Design is a DevPack for the SensorTag for which the ambient temperature is specified from 0°C to 70°C, the following calculations are based on ambient temperature range from –40°C to 85°C.

From the resistor value of the PT100 sensor, the temperature can be derived with Callendar-van Dusen equations [1] or with the help of many available PT100 look-up tables.

Used PT100 temperature range in TIDA-00650: –40°C to 85°C

- –40°C → 84.2707 Ω
- 85°C → 132.8033 Ω

Extending the range slightly from 84 Ω (–40.68°C) to 134 Ω (88.14°C) gives the excitation current of I_{DAC1} = 250 µA a voltage range from 21 to 33.5 mV.

The RTD configuration in this design is according to the ADS1220 datasheet [2] and considered to be a pseudo-differential signal. The negative terminal of the PT100 sensor is set by $R_{REF} \times 2 \times I_{DAC1}$ = 3.24 kΩ × 2 × 250 µA = 1.62 V and stays at this potential. Thus, the positive leg of the PT100 sensor changes between 21 mV and 33.5 mV on top of the 1.62 V. With this information, the PGA gain setting of the ADS1220 can be calculated.

$$PGA_{GAIN} = \frac{V_{REF}}{V_{PT100,max}} = \frac{1.62\ V}{33.5\ mV} = 48.36\ V/V \quad (5)$$

which gives an available PGA_{GAIN} of 32 V/V. The resulting FSR is given by Equation 6:

$$FSR = \frac{\pm V_{REF}}{Gain} = \frac{\pm 1.62\ V}{32\ V/V} = \pm 50.625\ mV \quad (6)$$

3.2.1 Common-Mode Voltage Requirements for CJC Channel

Because the design uses the low-noise PGA of the ADS1220, verify that the PGA common-mode voltage requirements are met. In the ADS1220 datasheet [2], the following three equations are used to meet those requirements:

- $V_{CM(MIN)}$ ≥ AVSS + ¼(AVDD – AVSS)
- $V_{CM(MIN)}$ ≥ AVSS + 0.2 V + ½Gain × $V_{IN(MAX)}$
- $V_{CM(MAX)}$ ≤ AVDD – 0.2 V – ½Gain × $V_{IN(MAX)}$

With Equation 7 through Equation 9, the PGA common-mode voltage for the RTD channel is from 0.825 V to 2.564 V. As described in Section 3.2, the RTD channel is a pseudo-differential signal. The negative input is held at a constant voltage, which in this design is the external reference voltage, V_{REF} = 1.62 V. The positive input varies between 1.62 V + 0.021 V = 1.641 V and 1.62 V + 0.0335 V = 1.654 V, resulting in a common-mode voltage between those two voltages, which is within the calculated compliance range.

$$V_{CM,MIN} \geq 0\ V + \frac{1}{4} \times (3.3\ V - 0\ V) = 0.825\ V \quad (7)$$

$$V_{CM,MIN} \geq 0\ V + 0.2\ V + \frac{1}{2} \times 32 \times 33.5\ mV = 0.736\ V \quad (8)$$

$$V_{CM,MAX} \leq 3.3\ V - 0.2\ V - \frac{1}{2} \times 32 \times 33.5\ mV = 2.564\ V \quad (9)$$

4 Getting Started: Hardware

4.1 Board Description

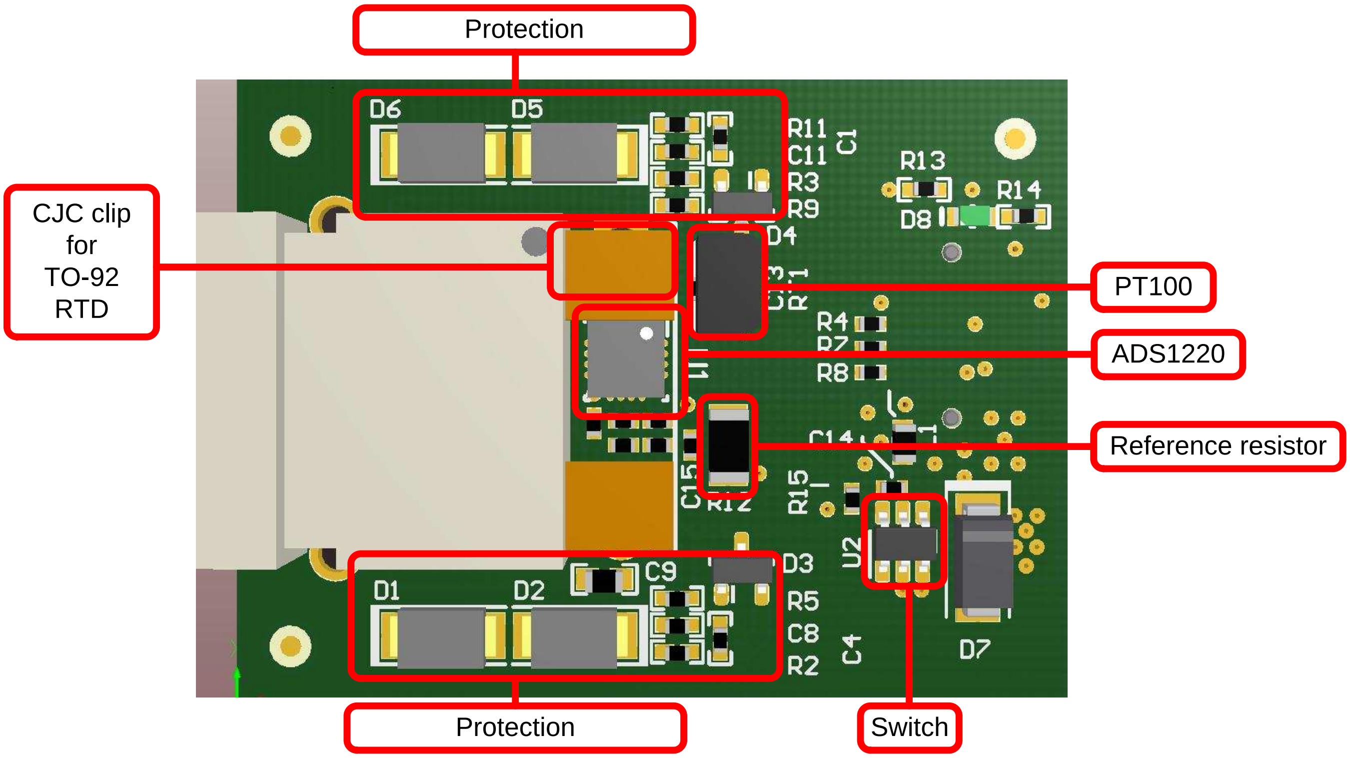


Figure 5. TIDA-00650 Board Overview

- CJC clip for TO-92 RTD
  - Mounts an RTD in a TO-92 package on to the cold junction
- Protection
  - Protection of the input towards the ADC
- PT100
  - RTD for measuring the cold-junction temperature
- ADS1220
  - Located directly at the cold-junction area (isothermal block)
  - Internal temperature sensor can be used for the CJC instead of PT100
- Reference resistor
  - Precision resistor to generate with IDAC of the ADC the external reference voltage for the ADC
- Switch
  - GPIO-controlled switched to power on or off the entire board

4.2 Hardware Setup Without SensorTag

The main purpose of this design is to use a DevPack in combination with the SensorTag. However, to evaluate and test the TI Design itself, leaving out the SensorTag is an easier approach. With a small adapter board, the TIDA-00650 can be easily connected to any MSP430™ LaunchPad™. Figure 6 shows the basic setup used to test and characterize the TIDA-00650. Programming the MSP430FR5969 LaunchPad was realized with Energia [6] and is explained in Section 5.2.

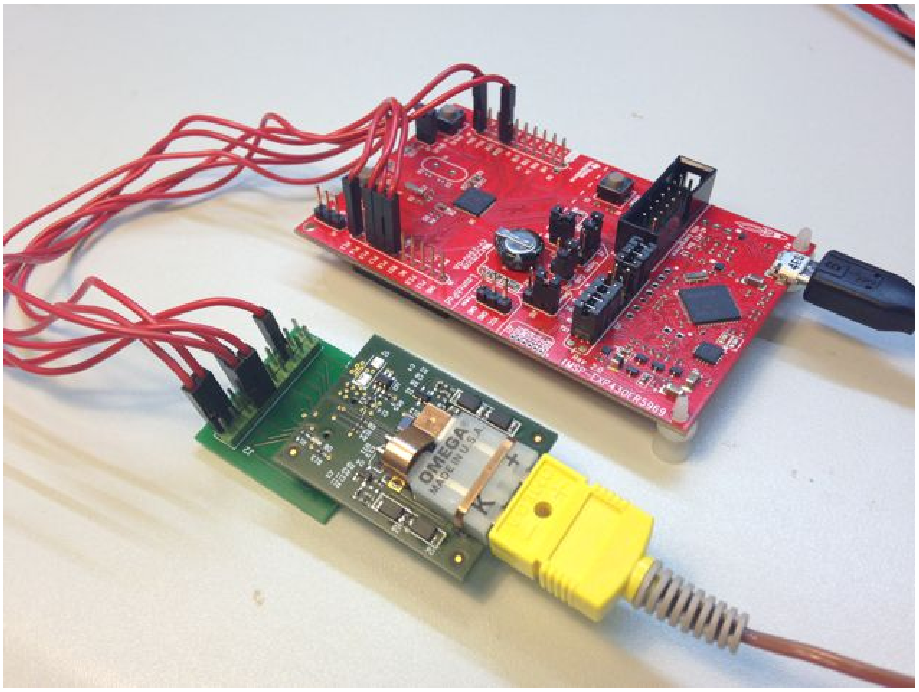


Figure 6. TIDA-00650 With MSP430FR5969 LaunchPad Setup

With the setup shown in Figure 6, the ADS1220 can easily be programmed and data read out on a PC. The MSP430 LaunchPad just acts as a bridge between USB and SPI as well as a control for the GPIOs. Table 3 is an overview of the connections between the TIDA-00650 and the LaunchPad.

Table 3. Connection Between TIDA-00650 and MSP430FR5969 LaunchPad Through SensorTag Adapter

TIDA-00650 (J2)	SENSORTAG ADAPTER BOARD (J1)	SENSORTAG ADAPTER BOARD (J2)	MSP430FR5969 LAUNCHPAD (J4, J5)	FUNCTION
2	1	2	VCC	VDD from SensorTag
1	2	1	GND	GND from SensorTag
13	14	13	10	PWR_UP_CHIP
19	20	19	11	LED
5	6	5	7	SCLK_ADC
17	18	17	13	nCS_ADC
12	11	12	15	DIN_ADC
14	13	14	14	DOUT_ADC

In addition to the connection between the TIDA-00650 and the LaunchPad, the LaunchPad must be connected to a PC through USB.

4.3 Firmware and Software

For a basic functional test, the Energia firmware "TIDA-00650_EnergiaCode_01" is used on the MSP430FR5969 LaunchPad. See Section 5 for more details on firmware and software.. The Python™ script "TIDA-00650_gettempint.py" runs on a PC to send the appropriate commands to the LaunchPad and displays the temperature in the console (Figure 7). The example shows five readings. The first value is the calculated thermocouple temperature in Celsius, and the second value is the read-out temperature of the internal temperature sensor of the ADS1220.

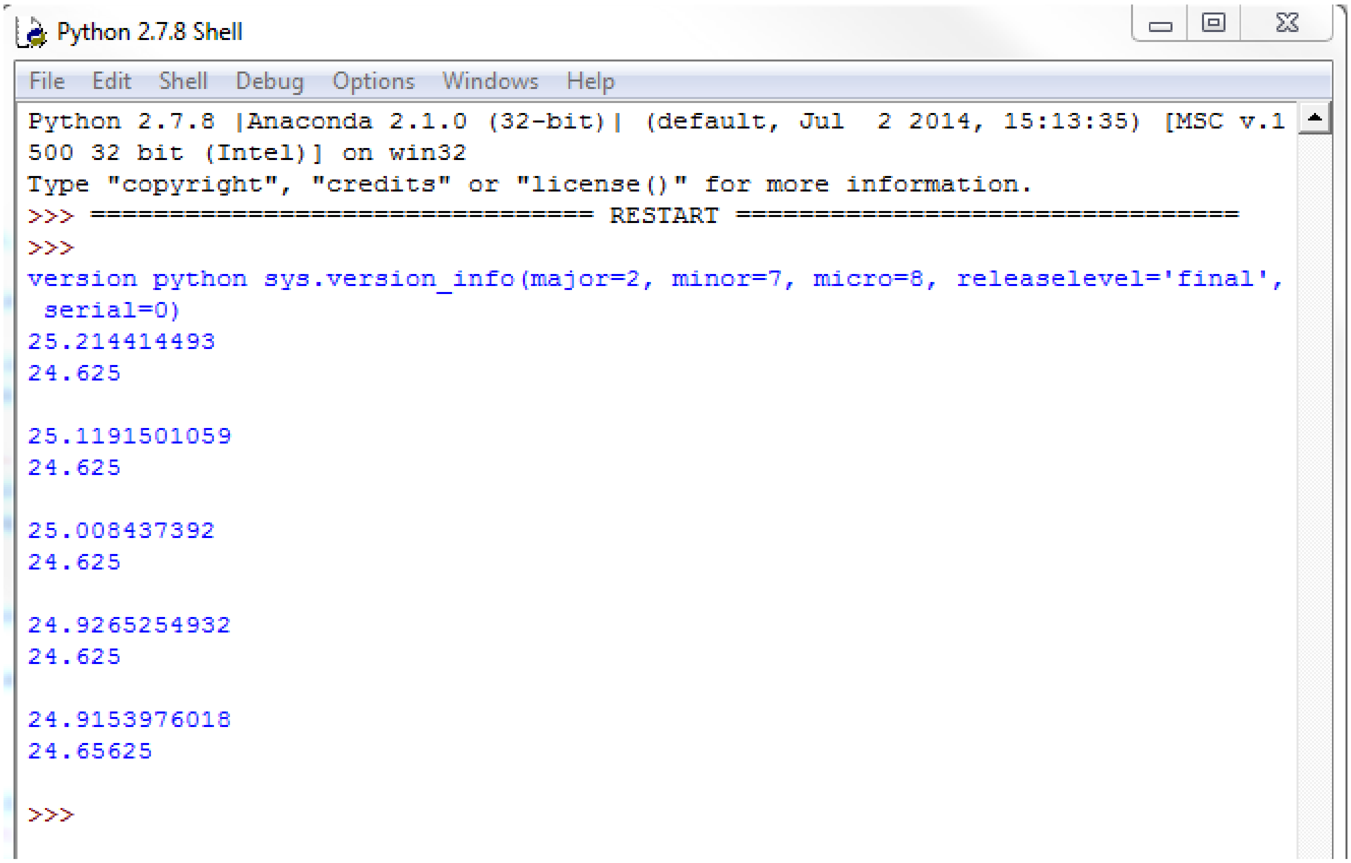


Figure 7. Python Console Output Displaying TC and CJC Temperature

5 Firmware and Software Description

5.1 ADS1220 Register Settings

Table 4 and Table 5 show the command definitions and configuration register map of the ADS1220, respectively. Find additional information in the ADS1220 datasheet [2]. In this TI Design, set the ADC into different modes to measure the thermocouple channel, the RTD channel, and the internal temperature sensor. The used configurations to test the TIDA-00650 are shown in Table 6 through Table 10.

Table 4. ADS1220 Command Definitions

COMMAND	DESCRIPTION	COMMAND BYTE[(1)]
RESET	Reset the device	0000 011x
START/SYNC	Start or restart conversions	0000 100x
POWERDOWN	Enter power-down mode	0000 001x
RDATA	Read data by command	0001 xxxx
RREG	Read *nn* registers starting at address *rr*	0010 *rrnn*
WREG	Write *nn* registers starting at address *rr*	0100 *rrnn*

(1) Operands: rr = Configuration register (00 to 11), nn = Number of bytes – 1 (00 to 11), and x = Don't care.

Table 5. ADS1220 Configuration Register Map

REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
00h	MUX[3:0]				GAIN[2:0]			PGA_BYPASS
01h	DR[2:0]			MODE[1:0]		CM	TS	BCS
02h	VREF[1:0]		50/60[1:0]		PSW	IDAC[2:0]		
03h	I1MUX[2:0]			I2MUX[2:0]			DRDYM	RESERVED

Five different configuration settings are pre-defined:

1. CONFIG TC1 (Table 6)
   - Configures the MUX of the ADC to measure the TC
     – AINP is connected to AIN0
     – AINN is connected to AIN1
   - PGA gain is set to 32
   - Internal reference voltage is used for VREF
   - IDAC1 and IDAC2 are disabled

Table 6. ADS1220 Configuration for Reading TC

CONFIG TC1: THERMOCOUPLE MEASUREMENT									
REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	COMMENT
00h	0	0	0	0	1	0	1	0	AINP: AIN0; AINN: AIN1; Gain: 32; PGA bypass: Off
01h	0	0	0	0	0	0	0	0	DR: 20 SPS; Normal mode; Single shot; Int temp: Off
02h	0	0	1	0	0	0	0	0	VREF: Internal; FIR: 50 Hz; IDAC: Off
03h	0	0	0	0	0	0	0	0	IDAC1: Disabled; IDAC2: Disabled

2. CONFIG TC2 (Table 7)
   - Configures the MUX of the ADC to measure the TC, but with changed polarity for optional chopping
     – AINP is connected to AIN1
     – AINN is connected to AIN0
   - PGA gain is set to 32
   - Internal reference voltage is used for VREF
   - IDAC1 and IDAC2 are disabled

Table 7. ADS1220 Configuration for Reading TC Swapped

CONFIG TC2: THERMOCOUPLE MEASUREMENT									
REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	COMMENT
00h	0	1	1	0	1	0	1	0	AINP: AIN1; AINN: AIN0; Gain: 32; PGA bypass: Off
01h	0	0	0	0	0	0	0	0	DR: 20 SPS; Normal mode; Single shot; Int temp: Off
02h	0	0	1	0	0	0	0	0	VREF: Internal; FIR: 50 Hz; IDAC: Off
03h	0	0	0	0	0	0	0	0	IDAC1: Disabled; IDAC2: Disabled

3. CONFIG RTD1 (Table 8)
   - Configures the MUX of the ADC to measure the RTD channel
     – AINP is connected to AIN2
     – AINN is connected to AIN3
   - PGA gain is set to 32
   - External reference voltage is used for V_{REF} (IDAC1 + IDAC2 across RREF)
   - IDAC1 and IDAC2 are each set to 250 µA

Table 8. ADS1220 Configuration for Reading RTD Channel

CONFIG RTD1: RTD MEASUREMENT									
REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	COMMENT
00h	0	1	0	1	1	0	1	0	AINP: AIN2; AINN: AIN3; Gain: 32; PGA bypass: Off
01h	0	0	0	0	0	0	0	0	DR: 20 SPS; Normal mode; Single shot; Int temp: Off
02h	0	1	1	0	0	1	0	0	VREF: External; FIR: 50 Hz; IDAC: 250 µA
03h	0	1	1	1	0	0	0	0	IDAC1: AIN2; IDAC2: AIN3

4. CONFIG RTD2 (Table 9)
   - Configures the MUX of the ADC to measure the RTD channel, but with changed polarity for optional chopping
     – AINP is connected to AIN3
     – AINN is connected to AIN2
   - PGA gain is set to 32
   - External reference voltage is used for VREF (IDAC1 + IDAC2 across RREF)
   - IDAC1 and IDAC2 are each set to 250 µA

Table 9. ADS1220 Configuration for Reading RTD Channel Swapped

CONFIG RTD2: RTD MEASUREMENT									
REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	COMMENT
00h	0	1	1	1	1	0	1	0	AINP: AIN3; AINN: AIN2; Gain: 32; PGA bypass: Off
01h	0	0	0	0	0	0	0	0	DR: 20 SPS; Normal mode; Single shot; Int temp: Off
02h	0	1	1	0	0	1	0	0	VREF: External; FIR: 50 Hz; IDAC: 250 µA
03h	0	1	1	1	0	0	0	0	IDAC1: AIN2; IDAC2: AIN3

5. CONFIG INTTEMP (Table 10)
   - BIT 1 of register 01h is set to 1, which enables the reading of the internal temperature sensor.
   - In this mode, all required settings are made automatically

Table 10. ADS1220 Configuration for Reading Internal Temperature

CONFIG INTTEMP: INTERNAL TEMPERATURE SENSOR									
REGISTER (HEX)	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	COMMENT
00h	0	0	0	0	0	0	0	0	Default
01h	0	0	0	0	0	0	1	0	Internal temp sensor activated
02h	0	0	0	0	0	1	0	0	Default
03h	0	0	0	0	0	0	0	0	Default

5.2 *MSP430FR5969 LaunchPad With Energia*

The MSP430FR5969 LaunchPad is used to realize the communication between the Python script running on the PC and the TIDA-00650, specifically the ADS1220. The LaunchPad provides the link between the PC's USB port (virtual COM Port) and the SPI of the ADS1220. In addition, the different configurations listed in Section 5.1 are pre-programmed in the MSP430. The programming has been realized with Energia.

5.3 Energia Script

The following script shows the code for the MSP430FR5969 LaunchPad realized with Energia. The MSP430 is used to wait for pre-defined commands through the serial port (USB virtual COM Port). Depending on the command being sent, the ADS1220 is configured according to the description in Section 5.1. The read back values from the ADS1220 are then sent back to the PC.

```
#include <SPI.h>
const int ADS1220_CS = 13;             // Define GPIO for ADS1220 CS
const int LED = 11;                    // Define GPIO to control LED
const int power_up=10;         // Define GPIO to Power up the board

unsigned char byte1;
unsigned char byte2;
unsigned char byte3;
signed long result;
signed long ADCcode;
float inttemp;
float RTD;
float Volt;
float data[5];

int incomingByte[5];// = 0;            // for incoming serial data

void setup()
{
  pinMode(ADS1220_CS, OUTPUT);         // Define ADS1220 CS GPIO as output
  pinMode(LED, OUTPUT);        // Define LED GPIO as output
  pinMode(power_up, OUTPUT);           // Define Power up GPIO as output
  SPI.begin();
  SPI.setDataMode(SPI_MODE1);          // SPI mode 1 config
  SPI.setBitOrder(MSBFIRST);   // SPI Bitorder set to MSB first
  SPI.setClockDivider(128);            // Define SPI clock frequency
  Serial.begin(9600);                  // opens serial port, sets data rate to 9600 bps
  digitalWrite(power_up,HIGH);         // Set GPIO power-up High
  LED_blink(200);                      // Call function LED_blink
}

void loop()
{
  incomingByte[0]=0;
  incomingByte[1]=0;
  incomingByte[2]=0;
  incomingByte[3]=0;
  incomingByte[4]=0;

  while (Serial.available() == 0)      // Wait until a serial command is available
  {
    delay(10);
  }
```

```
    incomingByte[0] = Serial.read();

    switch(incomingByte[0])           // Check for the incoming command
    {
      //CONFFIG TC1            // Call Function tc1 when command is "q"
      case(113): //q
      LED_blink(50);
      tc1();
      break;

      //CONFFIG TC2            // Call Function tc2 when command is "w"
      case(119): //w
      LED_blink(50);
      tc2();
      break;

      //CONFFIG RTD1           // Call Function rtd1 when command is "e"
      case(101): //e
      LED_blink(50);
      rtd1();
      break;

      //CONFFIG RTD2           // Call Function rtd2 when command is "r"
      case(114): //r
      LED_blink(50);
      rtd2();
      break;

      //CONFFIG INTTEMP        // Call Function temp when command is "t"
      case(116): //t
      LED_blink(50);
      temp();
      break;

      //CONFFIG SHORTINPUT     // Call Function shortinput when command is "y"
      case(121): //y
      LED_blink(50);
      shortinput();
      break;

      //GETDATA                // Call Function getdata when command is "u"
      case(117): //u
      LED_blink(50);
      getdata();
      break;
    }
  }
```

```
void getdata()
{
    digitalWrite(ADS1220_CS,LOW);             //CS low
    delay(10);
    //Single Shot Acquisition Mode
    SPI.transfer(0b00001000);                         //transfer start/sync command
    delay(500);  //wait until data ready
    byte1=SPI.transfer(0b00000000);           //Data MSB
    byte2=SPI.transfer(0b00000000);           //
    byte3=SPI.transfer(0b00000000);           //Data LSB
    ADCcode = (((long)byte1<<24) + ((long)byte2<<16) + ((long)byte3<<8))/256;
    Serial.println(ADCcode);
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}

void shortinput()
{
    digitalWrite(ADS1220_CS,LOW);             //CS low
    delay(10);
    SPI.transfer(0b00000110);                  //ADC RESET
    //CONFIG 0
    SPI.transfer(0x40);                               // Write Config Register 0
    SPI.transfer(0b11101010);                 // MUX: shorted; Gain: 32; PGAbypass: 0
    //CONFIG 1
    SPI.transfer(0x44);                               // Write Config Register 1
    SPI.transfer(0x00);
    //CONFIG 2
    SPI.transfer(0x48);                               // Write Config Register 2
    SPI.transfer(0b00100000);                 // VREF: INT; 50/60Hz: 50; PSW: 0; IDAC; OFF
    //CONFIG 3
    SPI.transfer(0x4C);                               // Write Config Register 2
    SPI.transfer(0b00000000);                 // IDAC1: OFF; IDAC2: OFF; DRDY: default
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}

void tc1()
{
    digitalWrite(ADS1220_CS,LOW);             //CS low
    delay(10);
    SPI.transfer(0b00000110);                 //ADC RESET
    //CONFIG 0
    SPI.transfer(0x40);                               // Write Config Register 0
    SPI.transfer(0b00001010);                 // MUX: AIN0,AIN1; Gain: 32; PGAbypass: 0
    //CONFIG 1
    SPI.transfer(0x44);                               // Write Config Register 1
    SPI.transfer(0x00);
    //CONFIG 2
    SPI.transfer(0x48);                               // Write Config Register 2
    SPI.transfer(0b00100000);                 // VREF: INT Rref; 50/60Hz: 50; PSW: 0; IDAC; off
    //CONFIG 3
    SPI.transfer(0x4C);                               // Write Config Register 2
    SPI.transfer(0b00000000);                 // IDAC1: off; IDAC2: off; DRDY: default
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}
```

```
void tc2()
{
    digitalWrite(ADS1220_CS,LOW);            //CS low
    delay(10);
    SPI.transfer(0b00000110);                //ADC RESET
    //CONFIG 0
    SPI.transfer(0x40);                              // Write Config Register 0
    SPI.transfer(0b01101010);                // MUX: AIN1,AIN0; Gain: 32; PGAbypass: 0
    //CONFIG 1
    SPI.transfer(0x44);                              // Write Config Register 1
    SPI.transfer(0x00);
    //CONFIG 2
    SPI.transfer(0x48);                              // Write Config Register 2
    SPI.transfer(0b00100000);                // VREF: INT Rref; 50/60Hz: 50; PSW: 0; IDAC; off
    //CONFIG 3
    SPI.transfer(0x4C);                              // Write Config Register 2
    SPI.transfer(0b00000000);                // IDAC1: off; IDAC2: off; DRDY: default
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}

void rtd1()
{
    digitalWrite(ADS1220_CS,LOW);            //CS low
    delay(10);
    SPI.transfer(0b00000110);                //ADC RESET
    //CONFIG 0
    SPI.transfer(0x40);                              // Write Config Register 0
    SPI.transfer(0b01011010);                // MUX: AIN2,AIN3; Gain: 32; PGAbypass: 0
    //CONFIG 1
    SPI.transfer(0x44);                              // Write Config Register 1
    SPI.transfer(0x00);
    //CONFIG 2
    SPI.transfer(0x48);                              // Write Config Register 2
    SPI.transfer(0b01100100);                // VREF: EXT Rref; 50/60Hz: 50; PSW: 0; IDAC; 250uA
    //CONFIG 3
    SPI.transfer(0x4C);                              // Write Config Register 2
    SPI.transfer(0b01110000);                // IDAC1: AIN2; IDAC2: AIN3; DRDY: default
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}

void rtd2()
{
    digitalWrite(ADS1220_CS,LOW);            //CS low
    delay(10);
    SPI.transfer(0b00000110);                //ADC RESET
    //CONFIG 0
    SPI.transfer(0x40);                              // Write Config Register 0
    SPI.transfer(0b01111010);                // MUX: AIN3,AIN2; Gain: 32; PGAbypass: 0
    //CONFIG 1
    SPI.transfer(0x44);                              // Write Config Register 1
    SPI.transfer(0x00);
    //CONFIG 2
    SPI.transfer(0x48);                              // Write Config Register 2
    SPI.transfer(0b01100100);                // VREF: EXT Rref; 50/60Hz: 50; PSW: 0; IDAC; 250uA
    //CONFIG 3
    SPI.transfer(0x4C);                              // Write Config Register 2
    SPI.transfer(0b01110000);                // IDAC1: AIN2; IDAC2: AIN3; DRDY: default
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}
```

```
void temp()
{
    digitalWrite(ADS1220_CS,LOW);            //CS low
    delay(10);
    SPI.transfer(0b00000110);                //ADC RESET
    SPI.transfer(0x44);
    SPI.transfer(0x02);
    delay(10);
    digitalWrite(ADS1220_CS,HIGH);
}

void LED_blink(int td)
{
  digitalWrite(LED ,HIGH);
  delay(td);
  digitalWrite(LED ,LOW);
  delay(td);
  digitalWrite(LED ,HIGH);
  delay(td);
  digitalWrite(LED ,LOW);
  delay(td);
  digitalWrite(LED ,HIGH);
  delay(td);
  digitalWrite(LED ,LOW);
  delay(td);
  digitalWrite(LED ,HIGH);
  delay(td);
  digitalWrite(LED ,LOW);
  delay(td);
}
```

5.4 Python Script

For the simple setup described in Section 5.3, the following Python script is running on a PC.

```
#########################################
# DESCRIPTION                           #
#########################################
# TIDA-00650
# Test Script #1
# Configures ADS1220 to read Thermocouple
# Configures ADS1220 to read internal Temp Sensor
# Uses Python Module "thermocouples" to calculate
# the temperature
# NOTE: This script does not use PT100
#########################################

import datetime,time
import serial
from thermocouples_reference import thermocouples
import bitstring

typeK = thermocouples['K']
ser=serial.Serial('COM45',timeout=0)
ser.close()
ser.open()

for x in range(5):
    ## Config TC2
    ser.write('w')                              # Command to MSP430 to configure ADS1220 in TC2
Mode
    ## GET DATA
    ser.write('u')                              # Command to MSP430 to read out the captured data
    while ser.inWaiting() == 0:                 # Wait for completing the datatransfer
        pass
    tc2=float(ser.readline())                   # Store the data in variable "tc2"
    ## Config TEMP
    ser.write('t')                              # Command to MSP430 to configure ADS1220 in
Internal Temp Mode
    ## GET DATA
    ser.write('u')                              # Command to MSP430 to read out the captured data
    while ser.inWaiting() == 0:                 # Wait for completing the datatransfer
        pass
    temp=float(ser.readline())                  # Store the data in variable "temp"
    s = bitstring.pack('int:24',int(temp))      # Generate a bitstring
    c = s[:14]                                  # extract the 14Bits out of the 24Bits
    t = c.int*0.03125                           # calculate the temperature based on the internal
Temp Sensor

    adcvoltage = (tc2/(2**23))*(2.048/32)       #calculated ADC voltage from tc2
    tctemp = typeK.inverse_CmV(adcvoltage*1000, Tref=t)    #calculated TC temp from tc2 for
Tref=internal Temp sensor
    print tctemp                                # Print on concole thermocouple temperature
    print t                                     # Print on console CJC temperature (internal Temp
Sensor)
    print(' ')
ser.close()
```

6 Test Setup

The entire test setup is automated. A Python [5] script controls all measurement instruments as well as the configuration register of the ADC with the help of the MSP430FR5969 LaunchPad, programmed with Energia [6]. Figure 8 shows the test setup used to characterize the TIDA-00650.

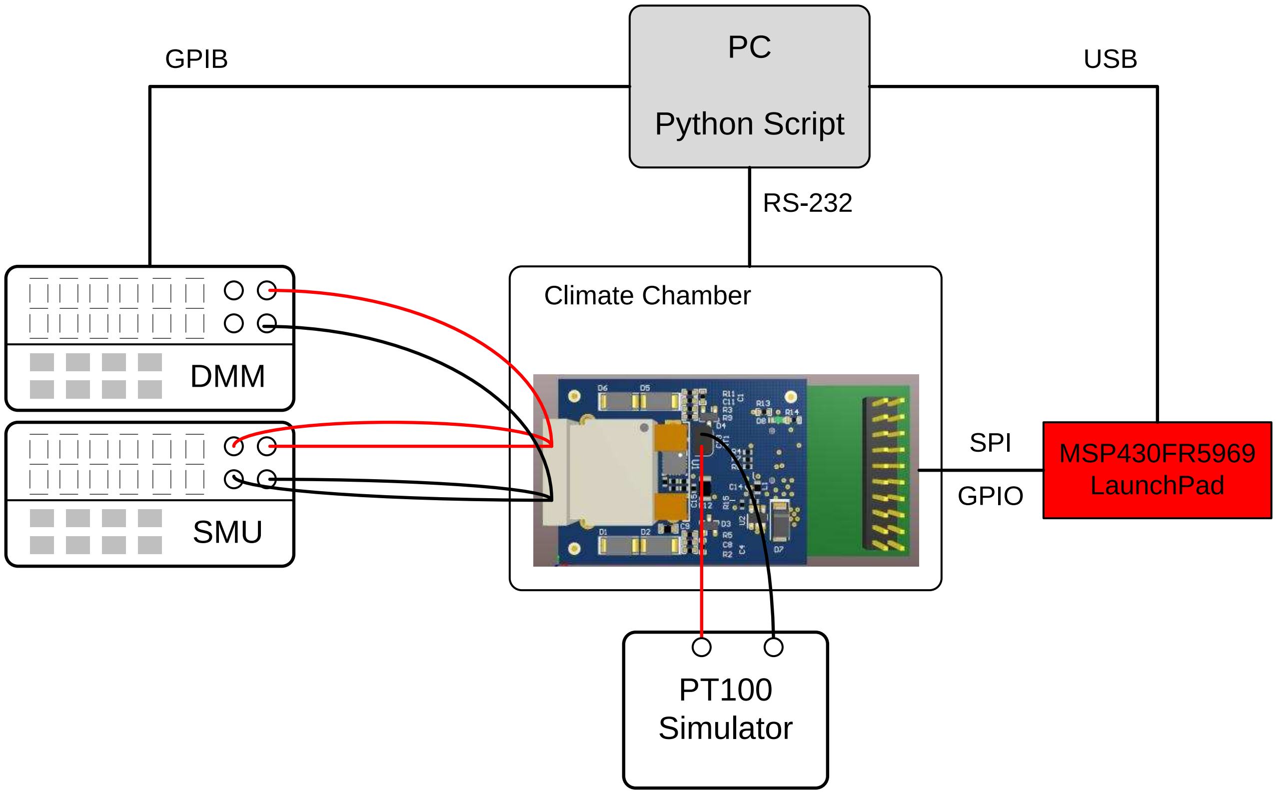


Figure 8. Test Setup

Instruments:

- Keysight Dual Source Meter Unit (SMU): B2912A
- HP 8.5 Digit Digital Multimeter (DMM): 3458A
- Time Electronics: PT100 Simulator
- Climate Chamber
- MSP430FR5969 LaunchPad
- PC

With the SMU the Type-K equivalent thermovoltage can be generated and provided to the input of the design. The PT100 Simulator has a set of high precision resistor values representing 23 fixed temperatures in the range from –200°C to 800°C. This allows testing the two channels independently. With the 8.5-digit DMM, the PT100 Simulator resistors are measured up front to calculate with the exact values. In addition, the DMM measures the provided thermocouple voltage directly at the connector.

The converted ADC raw data are sent through the LaunchPad at the top the Python script on the PC where all data are stored.

For further readings on the theoretical part of the thermocouple, see the TI Design TIDA-00168.

7 Test Results

7.1 PT100 Simulator

For the different tests, a PT100 simulator provides high precision resistors at the RTD channel. The RTD channel measures the CJC; therefore, resistor values representing the temperature range from –50°C to 125°C are sufficient. The PT100 simulator is connected with long leads to the PCB, which have to be taken into account as well. A separate measurement of the actual PT100 simulator resistor values and the resulting equivalent temperature have been performed prior the actual board characterization. Table 11 shows the measured results. The highlighted values have been used for further calculations.

Table 11. PT100 Simulator

PT100 SIMULATOR TEMPERATURE (°C)	PT100 SIMULATOR RESISTOR VALUE (R)	PT100 SIMULATOR MEASURED RESISTOR VALUE (R)	EQUIVALENT TEMPERATURE FROM MEASURED PT100 RESISTOR VALUE (°C)	TEMPERATURE ERROR (°C)	RESISTOR ERROR (R)
–50	80.306282	80.6509886	–49.13186760	0.868132398	0.3447067
–20	92.159898	92.4676791	–19.21715314	0.782846856	0.3077807
-10	96.085879	96.4005896	–9.19706791	0.802932091	0.3147106
0	100.000000	100.3435740	0.87920231	0.879202314	0.3435740
10	103.902525	104.2424849	10.87253189	0.872531894	0.3399599
20	107.793500	108.1364338	20.88278288	0.882782881	0.3429338
30	111.672925	112.0142825	30.88134539	0.881345394	0.3413575
40	115.540800	115.8702743	40.85320509	0.853205095	0.3294743
50	119.397125	119.7203596	50.83955618	0.839556181	0.3232346
60	123.241900	123.6163923	60.97563759	0.975637589	0.3744923
80	130.896800	131.2386250	80.89591277	0.895912769	0.3418250
100	138.505500	138.9175985	101.08670820	1.086708234	0.4120985
150	157.325125	157.7162725	151.04740460	1.047404600	0.3911475

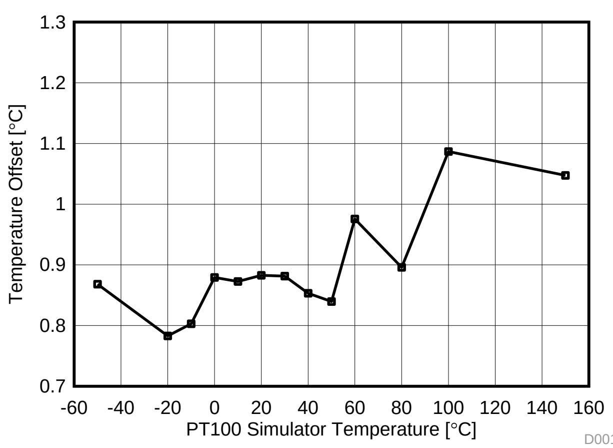


Figure 9. PT100 Simulator Equivalent Temperature Offset Including Test Wire

7.2 TC

For the TC test, the SMU provides the equivalent thermocouple voltages for a CJC reference of 0°C. In the Python code, the captured ADC codes are converted into the voltage and the equivalent thermocouple temperature. The differences in the thermocouple voltages are shown in the following graphs over different ambient temperatures with gain and offset calibration.

Figure 10 shows the temperature error across the provided thermocouple temperature from –200°C to 1200°C. The tests were done at different ambient temperatures (–40°C, –20°C, 0°C, 25°C, 50°C, 70°C, and 85°C). Prior to the test run, a gain calibration has been done at room temperature. Also, the offset error is eliminated because the test software has implemented chopping.

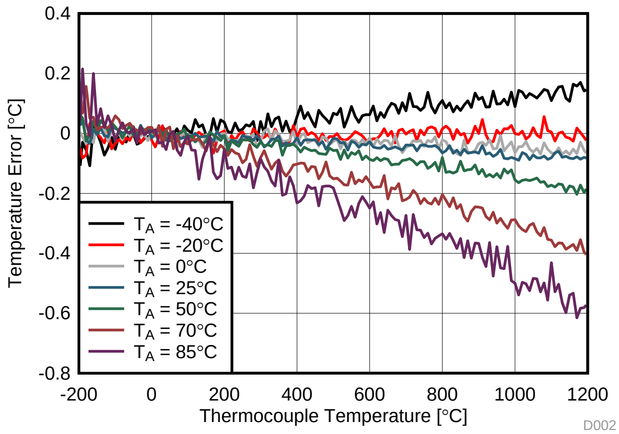


Figure 10. Thermocouple Temperature Error for All Tested Ambient Temperature With Gain and Offset Calibration

Since the TIDA-00650 design is made as an attachment to the SensorTag, which is specified for an ambient temperature from 0°C to 70°C, the overall error of the TC can be minimized compared to an operating temperature range from –40°C to 85°C. Figure 11 and Figure 12 indicate an improvement of around 0.2°C.

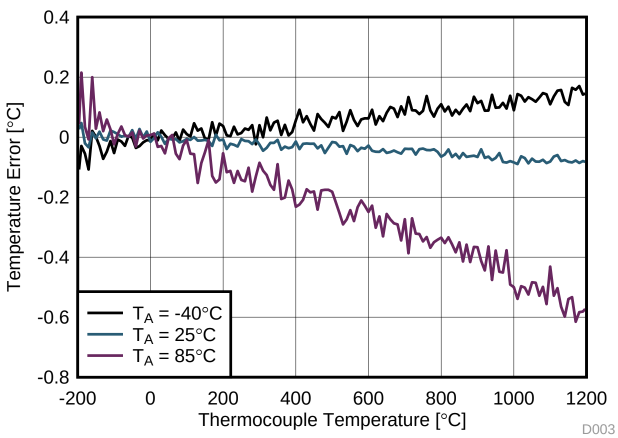


Figure 11. Thermocouple Temperature Error (T_A = –40°C to 85°C)

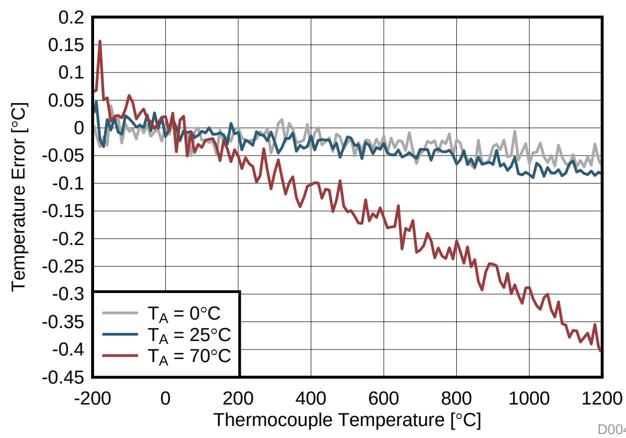


Figure 12. Thermocouple Temperature Error (T_A = 0°C to 70°C)

Figure 13 to Figure 17 are histogram plots of the TC. In all cases, the input voltage is 50 mV, and 1000 captures are taken for different ambient temperatures.

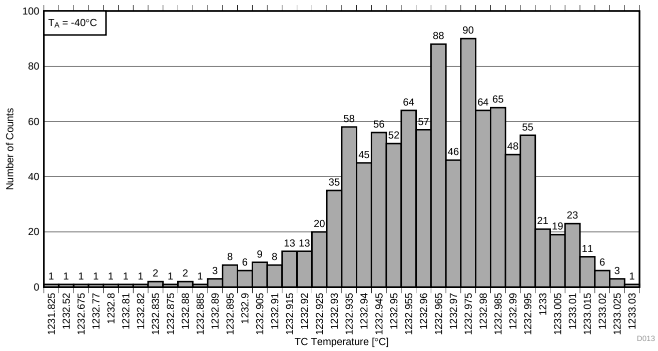


Figure 13. Thermocouple V_{IN} = 50 mV at T_A = –40°C; Min/Max = 1.189

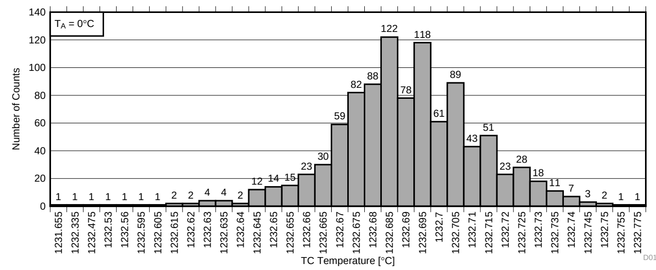


Figure 14. Thermocouple V_{IN} = 50 mV at T_A = 0°C; Min/Max = 1.118

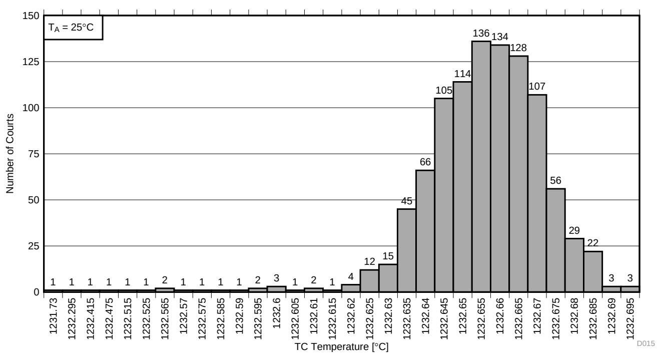


Figure 15. Thermocouple V_{IN} = 50 mV at T_A = 25°C; Min/Max = 0.966

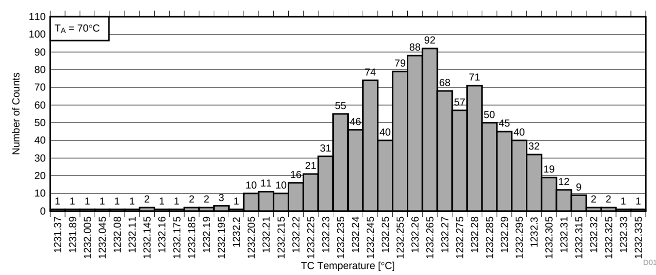


Figure 16. Thermocouple V_{IN} = 50 mV at T_A = 70°C; Min/Max = 0.962

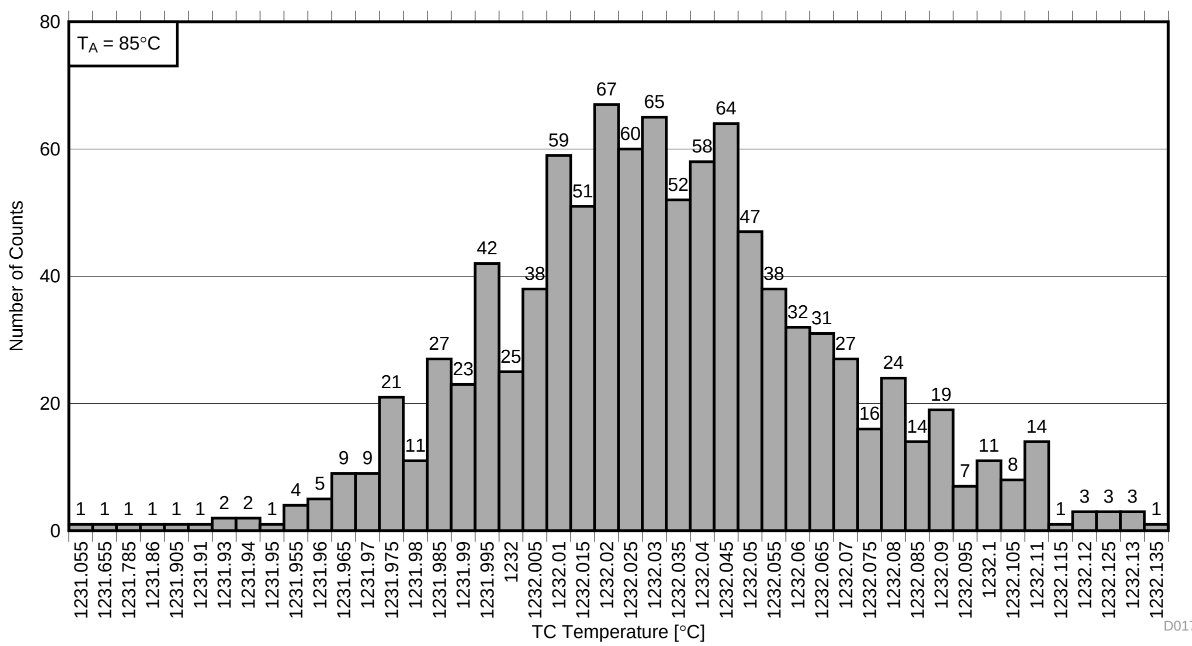


Figure 17. Thermocouple V_{IN} = 50 mV at T_A = 85°C; Min/Max = 1.084

7.3 RTD Channel

Table 11 shows that for a set PT100 simulator temperature of 0°C, the equivalent PT100 resistor must be 100 Ω. For the used PT100 simulator including the lead resistance, the resistor value is 100.34 Ω, resulting in an equivalent temperature of 0.88°C. Taken this into account, the temperature error of the RTD channel across ambient temperature is shown in Figure 18.

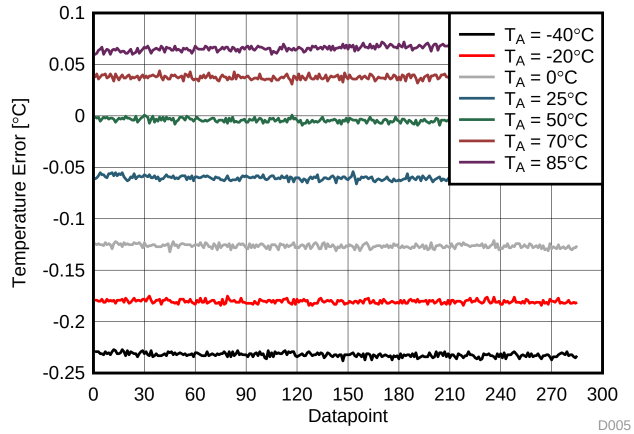


Figure 18. RTD Channel Temperature Error for PT100 = 100 Ω Across Ambient Temperature

The variations between an ambient temperature of –40°C to 85°C are approximately 0.3°C. Also comparing the range –40°C to 85°C with the limited ambient temperature range of 0°C to 70°C as shown already in Section 7.2, further improvements can be achieved (Figure 19 and Figure 20). In this case, the RTD temperature error is 0.2°C.

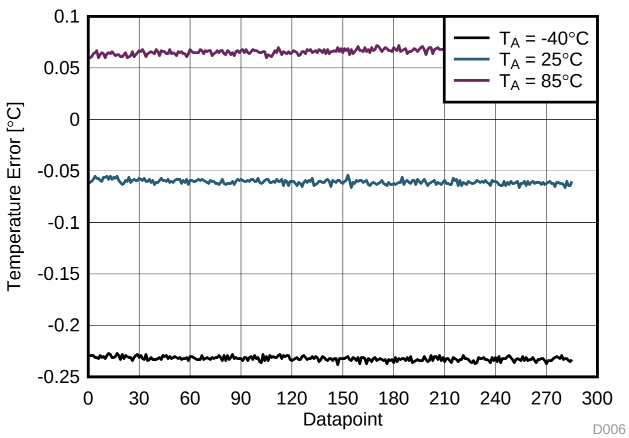


Figure 19. RTD Channel Temperature Error for PT100 = 100 Ω at –40°C, 25°C, and 85°C

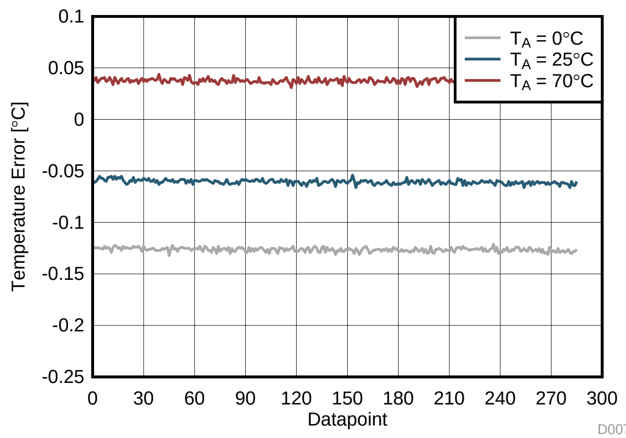


Figure 20. RTD Channel Temperature Error for PT100 = 100 Ω at 0°C, 25°C, and 70°C

From the previous plots where around 280 data points were taken, Figure 21 to Figure 25 show the corresponding histograms for different ambient temperatures and a PT100 simulator setting of 0°C.

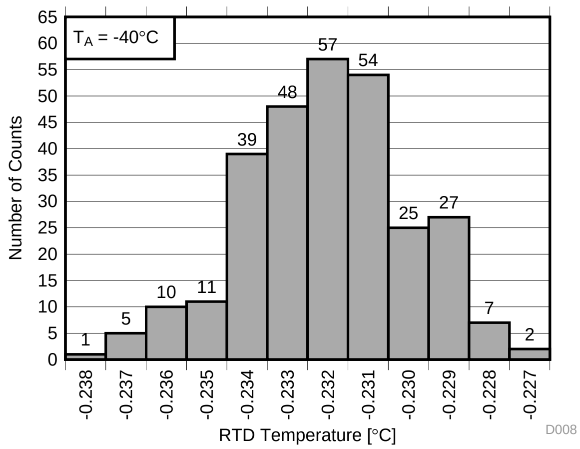


Figure 21. RTD Channel Histogram (280 Data Points) for PT100 = 100 Ω at –40°C

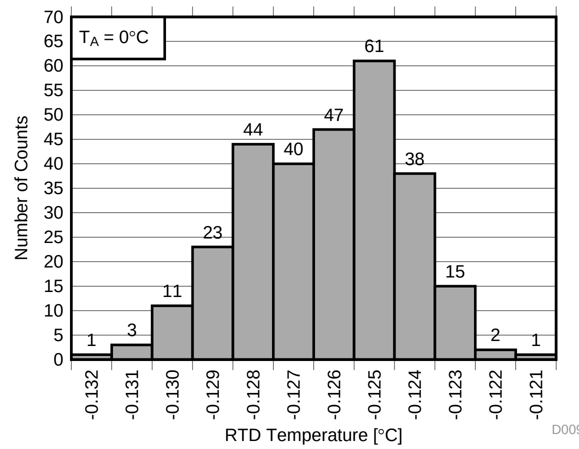


Figure 22. RTD Channel Histogram (280 Data Points) for PT100 = 100 Ω at 0°C

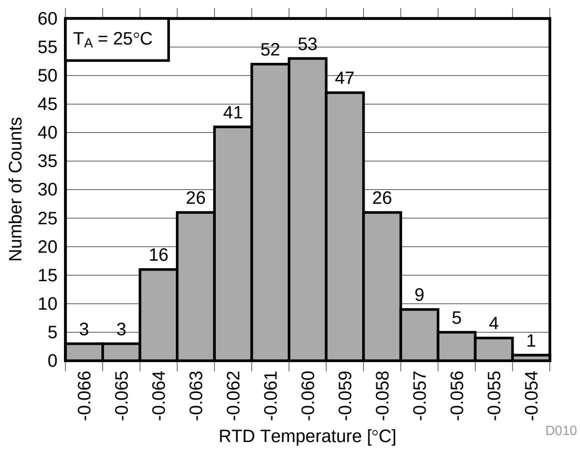


Figure 23. RTD Channel Histogram (280 Data Points) for PT100 = 100 Ω at 25°C

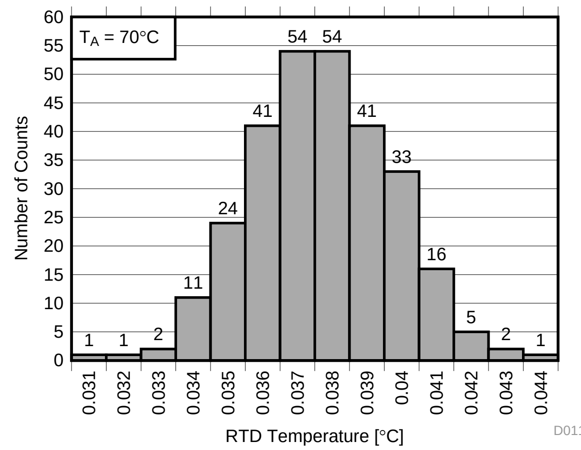


Figure 24. RTD Channel Histogram (280 Data Points) for PT100 = 100 Ω at 70°C

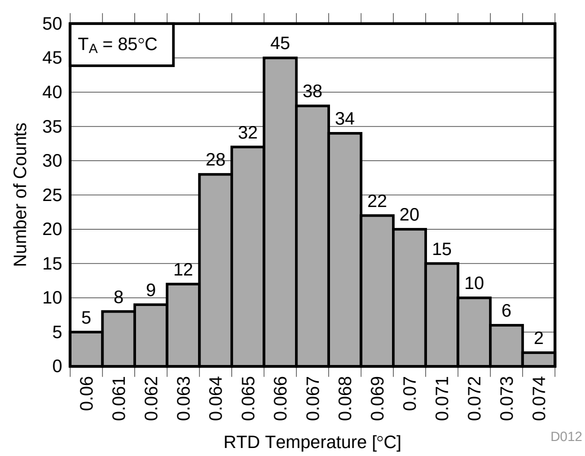


Figure 25. RTD Channel Histogram (280 Data Points) for PT100 = 100 Ω at 85°C

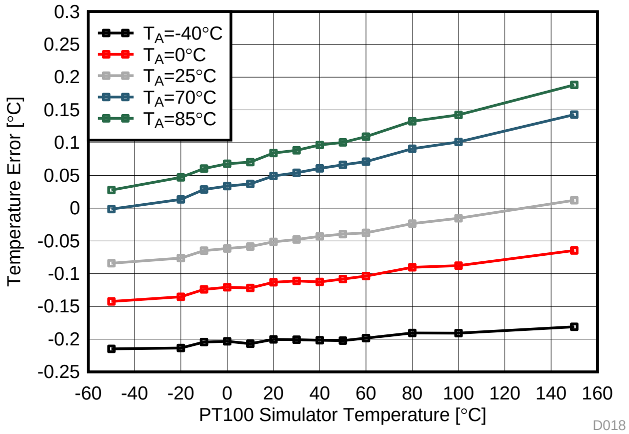


Figure 26. RTD Channel Temperature Error for Different PT100 Simulator Settings Across Ambient Temperature

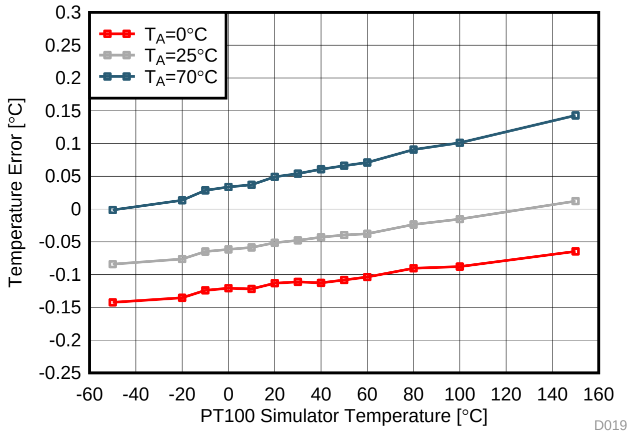


Figure 27. RTD Channel Temperature Error for Different PT100 Simulator Settings at 0°C, 25°C, and 70°C

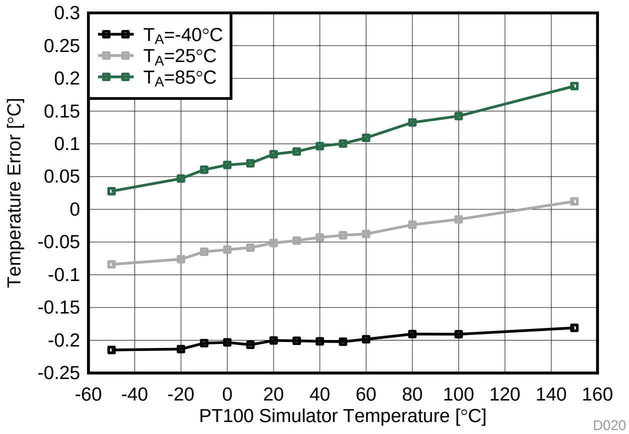


Figure 28. RTD Channel Temperature Error for Different PT100 Simulator Settings at –40°C, 25°C, and 85°C

8 Design Files

8.1 Schematics

To download the schematics, see the design files at TIDA-00650.

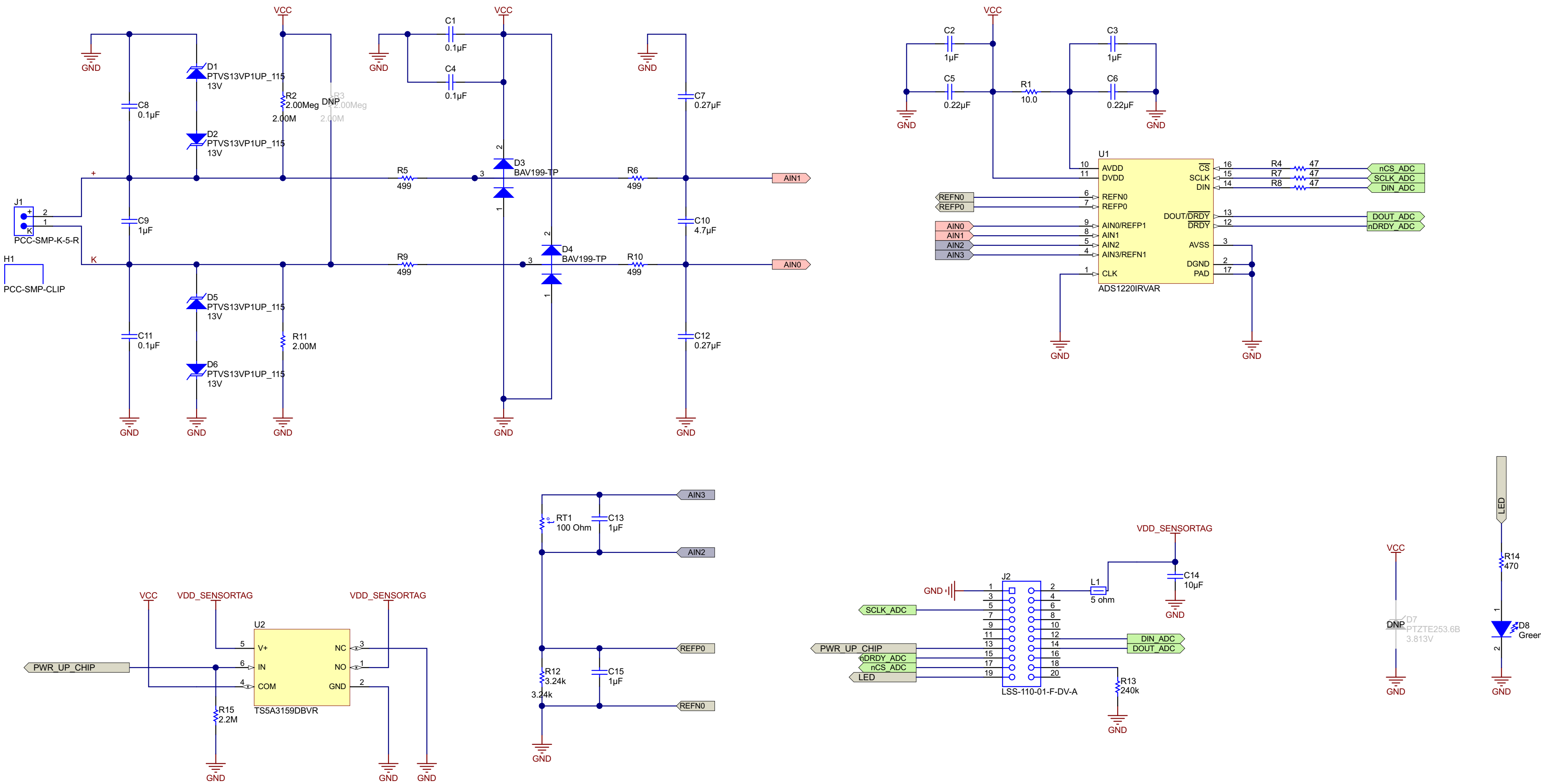


Figure 29. TIDA-00650 Schematic

8.2 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDA-00650.

Table 12. BOM

ITEM	DESIGNATOR	QTY	VALUE	PARTNUMBER	MANUFACTURER	DESCRIPTION	PACKAGE REFERENCE
1	!PCB1	1		TIDA-00650	Any	Printed Circuit Board	
2	C1, C4, C8, C11	4	0.1uF	C1005X7R1H104K050BB	TDK	CAP, CERM, 0.1 µF, 50 V, +/- 10%, X7R, 0402	0402
3	C2, C3, C13, C15	4	1uF	GRM155R61A105KE15D	MuRata	CAP, CERM, 1 µF, 10 V, +/- 10%, X5R, 0402	0402
4	C5, C6	2	0.22uF	GRM155R60J224KE01D	MuRata	CAP, CERM, 0.22 µF, 6.3 V, +/- 10%, X5R, 0402	0402
5	C7, C12	2	0.27uF	GRM155R60J274KE01D	MuRata	CAP, CERM, 0.27 µF, 6.3 V, +/- 10%, X5R, 0402	0402
6	C9	1	1uF	UMK107AB7105KA-T	Taiyo Yuden	CAP, CERM, 1 µF, 50 V, +/- 10%, X7R, 0603	0603
7	C10	1	4.7uF	C1005X5R1A475K050BC	TDK	CAP, CERM, 4.7 µF, 10 V, +/- 10%, X5R, 0402	0402
8	C14	1	10uF	GRM155R61A106ME11	MuRata	CAP, CERM, 10 µF, 10 V, +/- 20%, X5R, 0402	0402
9	D1, D2, D5, D6	4	13V	PTVS13VP1UP_115	NXP Semiconductor	Diode, TVS, Uni, 13 V, 600 W, SOD-128	SOD-128
10	D3, D4	2	70V	BAV199-TP	Micro Commercial Components	Diode, Switching, 70 V, 0.215 A, SOT-23	SOT-23
11	D8	1	Green	LG L29K-G2J1-24-Z	OSRAM	LED, Green, SMD	1.7x0.65x0.8mm
12	H1	1		PCC-SMP-CLIP	Omega	Mounting bracket for PCC-SMP	
13	J1	1		PCC-SMP-K-5-R	Omega	Circuit Board Thermocouple Connector, Miniature Size, Type K, Horizontal installation, TH	20.8x6.3x15.5 mm
14	J2	1		LSS-110-01-F-DV-A	Samtec	Terminal/Socket Combo, 25mil, 10x2, Gold, SMT	Terminal Socket, 10x2, Gold, SMT
15	L1	1	5 ohm	BLM18BB050SN1D	MuRata	Ferrite Bead, 5 ohm @ 100 MHz, 0.7 A, 0603	0603
16	R1	1	10.0	CRCW040210R0FKED	Vishay-Dale	RES, 10.0, 1%, 0.063 W, 0402	0402
17	R2, R11	2	2.00Meg	CRCW04022M00FKED	Vishay-Dale	RES, 2.00 M, 1%, 0.063 W, 0402	0402
18	R4, R7, R8	3	47	CRCW040247R0JNED	Vishay-Dale	RES, 47, 5%, 0.063 W, 0402	0402
19	R5, R6, R9, R10	4	499	CRCW0402499RFKED	Vishay-Dale	RES, 499, 1%, 0.063 W, 0402	0402
20	R12	1	3.24k	PFC-W1206R-12-3241-B	TT Electronics/IRC	RES, 3.24 k, 0.1%, 0.333 W, 1206	1206
21	R13	1	240k	CRCW0402240KFKED	Vishay-Dale	RES, 240 k, 1%, 0.063 W, 0402	0402

Table 12. BOM (continued)

ITEM	DESIGNATOR	QTY	VALUE	PARTNUMBER	MANUFACTURER	DESCRIPTION	PACKAGE REFERENCE
22	R14	1	470	ERJ-2RKF4700X	Panasonic	RES, 470, 1%, 0.1 W, 0402	0402
23	R15	1	2.2Meg	CRCW04022M20JNED	Vishay-Dale	RES, 2.2 M, 5%, 0.063 W, 0402	0402
24	RT1	1	100 Ohm	32209210	Heraeus Sensor Technology	Temperature Sensor, 100ohm, TC=+3850ppm, TO-92, TH	TO-92, 2pin
25	U1	1		ADS1220IRVAR	Texas Instruments	Low-Power, Low-Noise, 24-Bit Analog-to-Digital Converter for Small Signal Sensors, RVA0016A	RVA0016A
26	U2	1		TS5A3159DBVR	Texas Instruments	1 ohm SPDT Analog Switch, DBV0006A	DBV0006A
27	D7	0	3.813V	PTZTE253.6B	Rohm	Diode, Zener, 3.813 V, 1 W, SMA	SMA
28	FID1, FID2, FID3	0		N/A	N/A	Fiducial mark. There is nothing to buy or mount.	N/A
29	LBL1	0		THT-14-423-10	Brady	Thermal Transfer Printable Labels, 0.650" W x 0.200" H - 10,000 per roll	PCB Label 0.650"H x 0.200"W
30	R3	0	2.00Meg	CRCW04022M00FKED	Vishay-Dale	RES, 2.00 M, 1%, 0.063 W, 0402	0402

8.3 Layout Prints

To download the layer plots, see the design files at TIDA-00650.

8.4 Altium Project

To download the Altium project files, see the design files at TIDA-00650.

8.5 Layout Guidelines

8.5.1 ADS1220

Since the main and only critical component on this design is the ADS1220, please refer to its datasheet [2] for a detailed description, layout recommendations, and example layout.

8.5.2 CJC

For proper CJC, place the temperature sensor at the isothermal block. The TIDA-00650 uses a PT100 sensor element in a TO-92 equivalent package, which can be mounted onto the dedicated clip of the Type-K thermocouple connector. Therefore, place the PT100 element to the connector accordingly. Figure 30 illustrates the placement of the PT100 element. Note that during the actual assembly of the RTD, the package must be placed on top of the thermocouple connector. See Figure 31 for an example of how to mount the RTD to the connector.

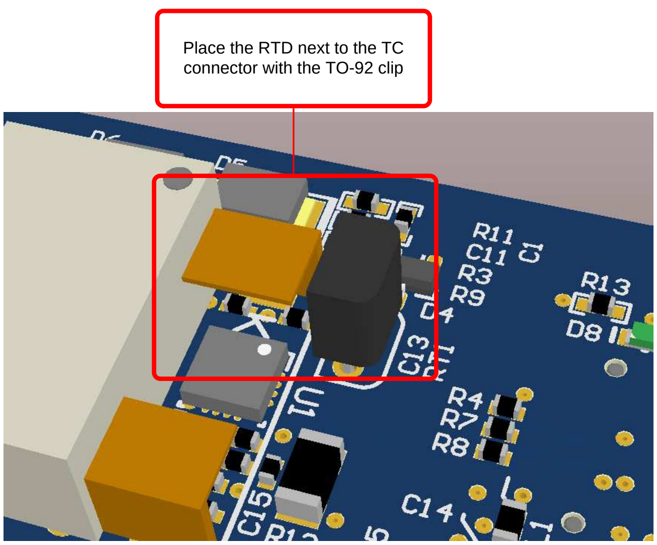


Figure 30. CJC Layout Requirements

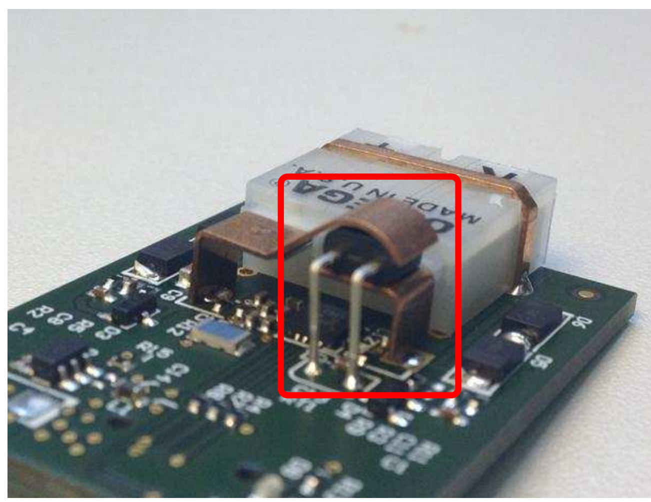


Figure 31. Assembly of PT100 on TC Connector Clip

8.6 Gerber Files

To download the Gerber files, see the design files at TIDA-00650.

8.7 Assembly Drawings

To download the assembly drawings, see the design files at TIDA-00650.

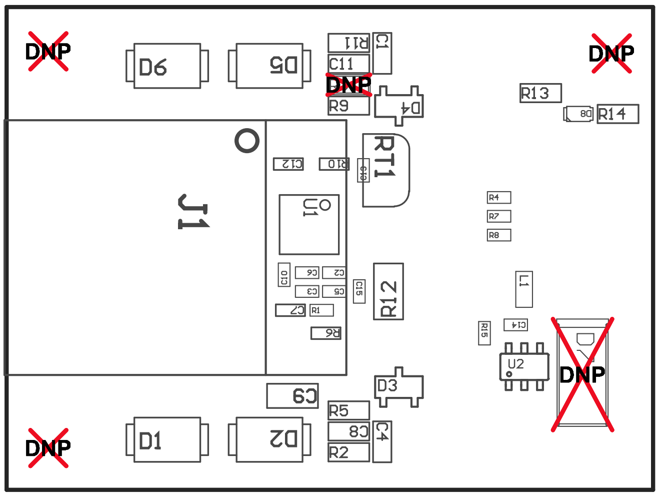


Figure 32. Top Layer Assembly Drawing

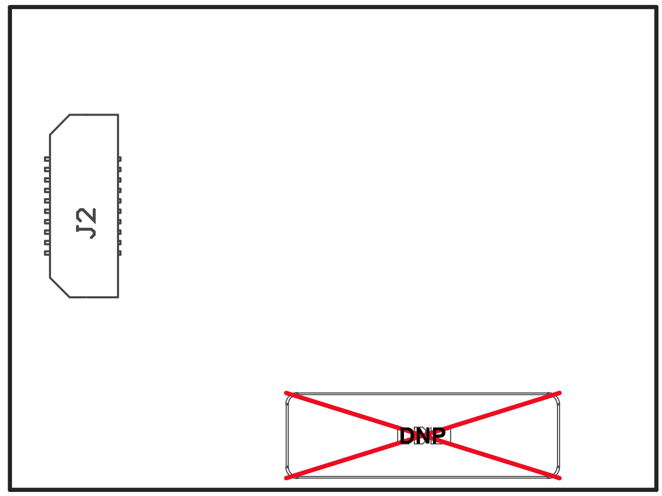


Figure 33. Bottom Layer Assembly Drawing

9 Software Files

To download the software files, see the design files at TIDA-00650.

10 References

1. International Bureau of Weights and Measures, *TECHNIQUES FOR APPROXIMATING THE INTERNATIONAL TEMPERATURE SCALE OF 1990* (http://www.bipm.org/utils/common/pdf/ITS-90/ITS-90-Techniques-for-Approximating.pdf)
2. Texas Instruments, *ADS1220 Low-Power, Low-Noise, 24-Bit ADC for Small Signal Sensors*, ADS1220 Datasheet (SBAS501)
3. Wikipedia, *Thermoelectric effect* (https://en.wikipedia.org/wiki/Thermoelectric_effect#Seebeck_effect)
4. Fluke Calibration, *Thermocouple Table Voltage Calculator* (http://us.flukecal.com/Thermocouple-Table-Voltage-Calculator)
5. Python Programming Language, (https://www.python.org/)
6. Energia, *Prototyping Software to Make Things Easy* (http://energia.nu/)
7. Texas Instruments, WEBENCH® Design Center, (http://www.ti.com/webench)
8. Texas Instruments, E2E Community (http://e2e.ti.com/)

11 About the Author

ALEXANDER WEILER is a systems engineer at Texas Instruments, where he is responsible for developing reference design solutions for the industrial segment. Alexander brings to this role his expertise in high-speed digital, low-noise analog, and RF system-level designs. Alexander earned his diploma in electrical engineering (Dipl.-Ing. (FH)) from the University of Applied Science in Karlsruhe, Germany.

Revision History

Changes from Original (September 2015) to A Revision **Page**

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265