Grove - 1-Wire Thermocouple Amplifier(MAX31850K)

The Grove - 1-Wire Thermocouple Amplifier (MAX31850K) is a thermocouple-to-digital converters with 14-bit resolution and cold-junction compensation. This module is designed to be used in conjunction with a k-type thermocouple. The thermocouples have a much larger measurement range than thermistors. For example, this k-type thermocouple on our website has a measurement range of -50℃ to +600℃.

This module is based on the MAX31850K, which integrates amplifier, ADC and 64-bit ROM. Thanks to the 64-bit ROM, each device has a unique 64-bit serial code, which allows multiple units to function on the same 1-Wire bus. Therefore, it is simple to use one microcontroller (the master device) to monitor temperature from many thermocouples distributed over a large area.

Again, this module can't work alone, it must work with a k-type thermocouple, if you do not have one, you can consider Thermocouple Temperature Sensor K Type-1M in our bazaar.

Features

- Integrated Cold-Junction Compensation
- Wide Conversion Range: allow readings from -270℃ to +1768℃
- 14-Bit, 0.25℃ Resolution
- Will not work with any other kind of thermocouple except K type
- Detects Thermocouple Shorts to GND or VDD
- Detects Open Thermocouple

Attention

Although this module can convert from -270℃ to +1768℃, the temperature measurment range also limited by the thermocouple you use.

Specification

| Item	Value
Operating Voltage	3.3V/5V
Temperature Resolution	14 bits
Temperature Accuracy	± 2℃
Operating Temperature Range	-40℃ to +125℃
Allow Readings Range	-270℃ to +1768℃
Storage Temperature Range	-65℃ to +150℃
Input Jack	DIP Female Blue-2Pin
Output Interface	1-Wire bus

Applications

- Medical
- Appliances
- Industrial
- HVAC(Heating, Ventilation and Air Conditioning)

Hardware Overview

Pin Map
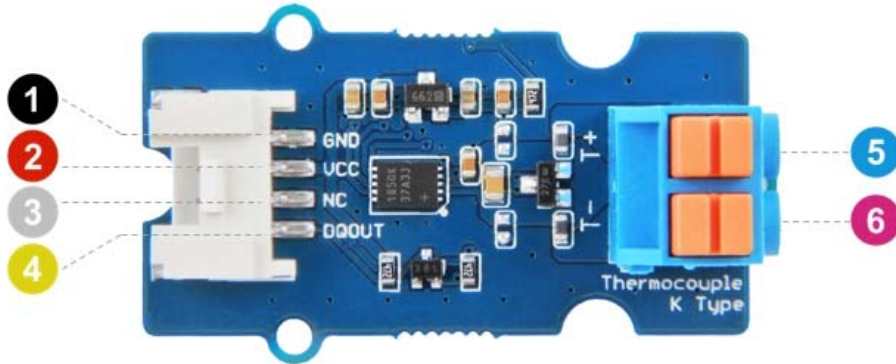
1. GND: connect this module to the system GND
2. VCC: you can use 5V or 3.3V for this module
3. NC: not connected
4. DQOUT: Data Input/Output. Open-drain 1-Wire interface pin.

5. T+: Thermocouple Input, positive pole.
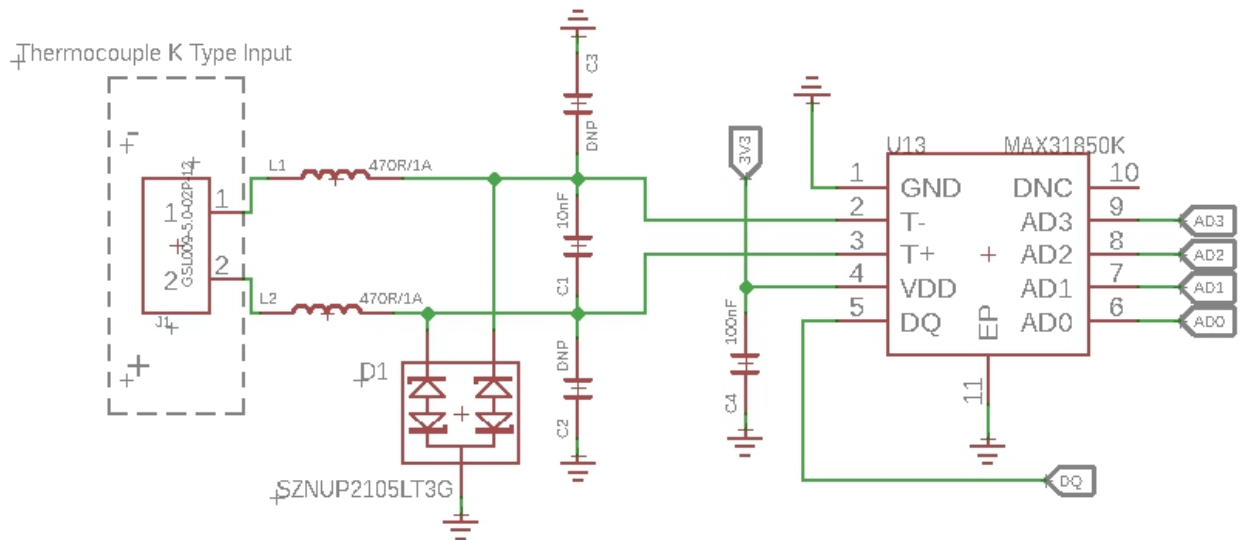6. T -: Thermocouple Input, negative pole
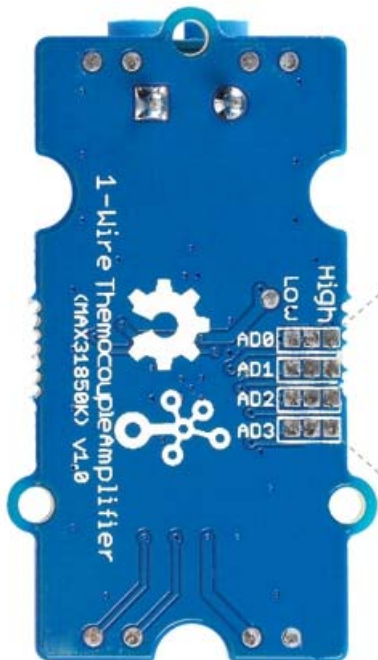
Four location address inputs simplify mapping of individual units to specific locations.

AD0(Least Significant Bit)

AD1

AD2

AD3(Most Significant Bit)

Default location address
0000

Schematic

**Input Jack**

Because of the small signal levels involved, we take a lot measures to filter the noise.

- **1--L1,L2** We use thermocouple up to 1 meter long. Such long wires can be regarded as antennas, which will receive spatial electric field interference and generate high frequency noise. So we use two inductances to filter the high frequency noise.
- **2--C1** It is strongly recommended by the chip manufacturer to add a 10nF ceramic surfacemount differential capacitor, placed across the T+ and T- pins, in order to filter noise on the thermocouple lines.
- **3--D1** We use the SZNUP2105LT3G DUAL BIDIRECTIONAL VOLTAGE SUPPRESSOR to protect this module from ESD(Electro-Static discharge).

Bi-directional level shifter circuit
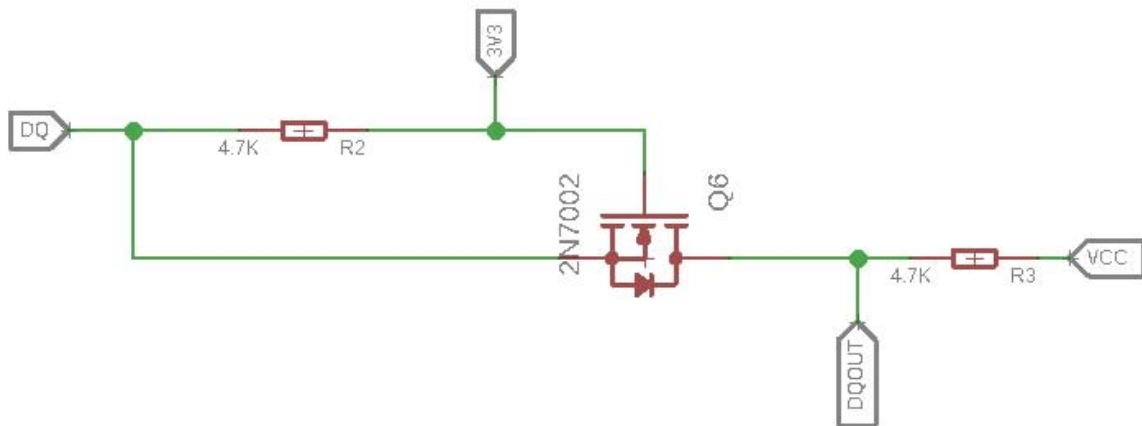
This is a typical Bi-directional level shifter circuit to connect two different voltage section. The left part, **DQ** pin of the MAX31850K use 3.3V, if the Arduino use 5V, this circuit will be needed. In the schematic above, **Q6** is N-Channel MOSFET 2N7002, which act as a bidirectional switch. In order to better understand this part, you can refer to the AN10441.

Note

In this section we only show you part of the schematic, for the full document please refer to the Resources

Assembling Drawing
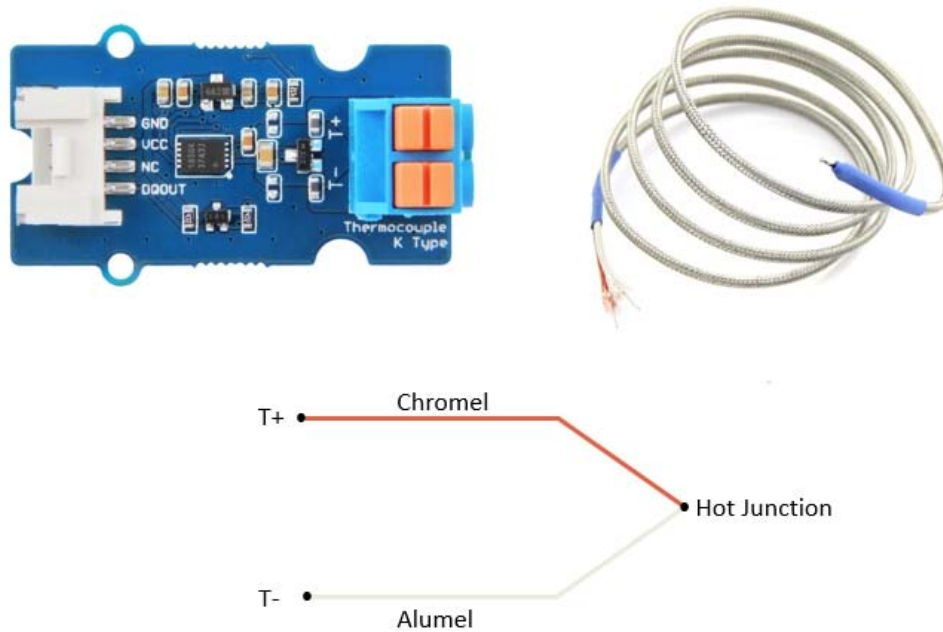
Tip

Please insert the **Red** wire of thermocouple into the **T+** port of the Grove - 1-Wire Themrocouple Amplifier (MAX31850K) , and the **White** wire into **T-** port. If you do not use the k-type thermocouple from our bazaar, the color may be different, please make sure the **+ -** with the seller.

Platforms Supported

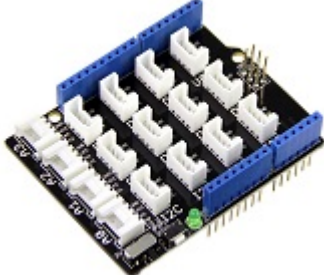| Arduino	Raspberry Pi	BeagleBone	Wio	LinkIt ONE

Getting Started

Play With Arduino

Hardware

Materials required

Seeeduino V4.2	Base Shield	Grove - 1-Wire Thermocouple Amplifier x 2

k-type thermocouple x 2		Grove - I2C Hub

- **Step 1.** Insert the red wire of k-type thermocouple into **T+**, insert the light white wire of k-type thermocouple into **T-**
- **Step 2.** Connect the Grove - I2C Hub to the **D3** port of the Base Shield.
- **Step 3.** Plug the Grove - 1-Wire Thermocouple Amplifier A and B into the Grove - I2C Hub.
- **Step 4.** Plug Grove - Base Shield into Seeeduino.
- **Step 5.** Connect Seeeduino to PC via a USB cable.
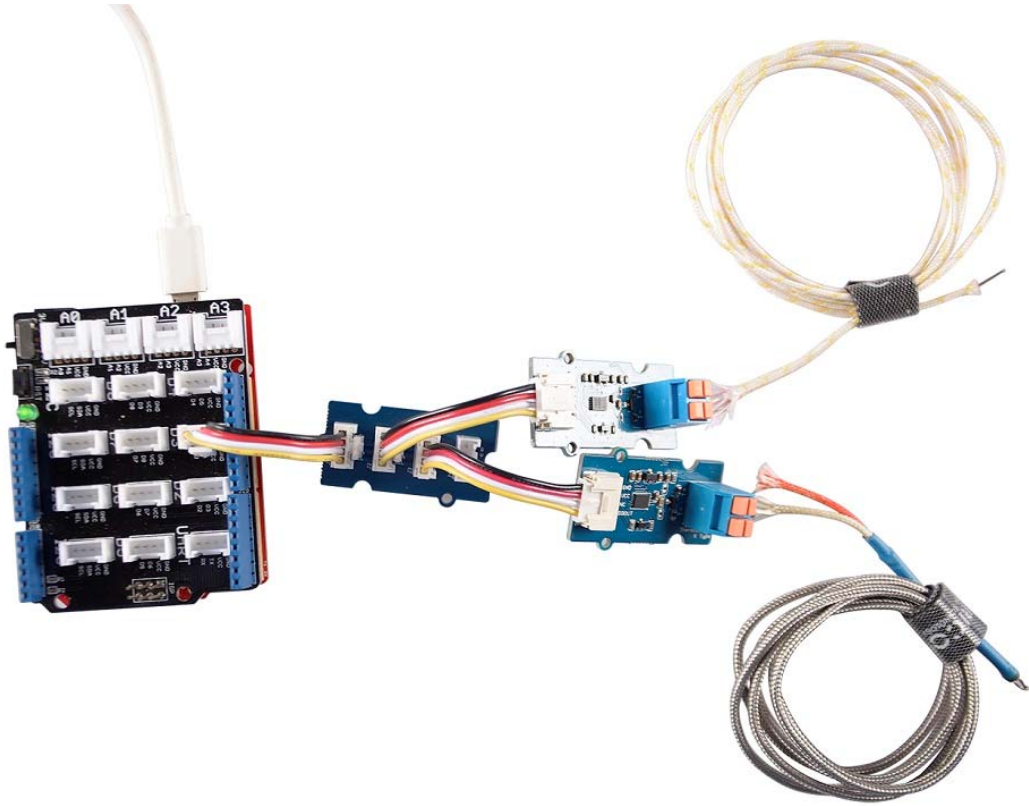
Seeeduino	Grove Cable	Grove - I2C Hub	Grove - 1-Wire Thermocouple Amplifier
GND	Black	G	GND
5V	Red	V	VCC
NC	White	SDA	NC
D3	Yellow	SCL	DQOUT

Software

- **Step 1.** Download the Seeed_MAX31850K Library from Github.
- **Step 2.** Refer to How to install library to install library for Arduino.
- **Step 3.** Unzip the library folder, follow the path to find and open **Multiple.ino** ---> **xxxx\Arduino\libraries\Seeed_MAX31850K-master\examples\Multiple**. xxxx is the path you installed your Arduino.

This folder **\Arduino\libraries\Seeed_MAX31850K-master\examples** contains two examples : **Multiple.ino** and **Simple.ino**.

Simple.ino--single mode(one host and one slave)

Multiple.ino--multiple mode(one host and multiple slaves,base on rom addressing-one wire search algorithm)

Or, you can just click the icon ⬜ in upper right corner of the code block to copy the following code into a new sketch in the Arduino IDE.

```
1  #include <OneWire.h>
2  #include <DallasTemperature.h>
3
4  // Data wire is plugged into port 2 on the Arduino
5  #define ONE_WIRE_BUS 3
6  #define TEMP_RESOLUTION 9
7
8  #define MAX_NUM_OF_DEVICE  10
9
10 // Setup a oneWire instance to communicate with any OneWire devices (not
11 just Maxim/Dallas temperature ICs)
12 OneWire oneWire(ONE_WIRE_BUS);
13
14 // Pass our oneWire reference to Dallas Temperature.
15 DallasTemperature sensors(&oneWire);
16
17 // arrays to hold device addresses
18 DeviceAddress Device_add[MAX_NUM_OF_DEVICE];
19 DeviceAddress insideThermometer, outsideThermometer;
20
21 void setup(void)
22 {
23   // start serial port
24   Serial.begin(115200);
25   Serial.println("Dallas Temperature IC Control Library Demo");
26
27   // Start up the library
28   sensors.begin();
29
30   // locate devices on the bus
31   Serial.print("Locating devices...");
32   Serial.print("Found ");
33   Serial.print(sensors.getDeviceCount(), DEC);
34   Serial.println(" devices.");
35
36   Serial.print("Parasite power is: ");
37   if (sensors.isParasitePowerMode()) Serial.println("ON");
38   else Serial.println("OFF");
39
40   for(int i=0;i<sensors.getDeviceCount();i++)
41   {
42     if(!sensors.getAddress(Device_add[i],i))
43     {
44       Serial.println("Find device error!!");
45     }
46     else
47     {
48       Serial.print("Device [");
```

```
49       Serial.print(i);
50       Serial.print("] addr =");
51       printAddress(Device_add[i]);
52     }
53     Serial.println(" ");
54   }
55
56 }
57
58 // function to print a device address
59 void printAddress(DeviceAddress deviceAddress)
60 {
61   for (uint8_t i = 0; i < 8; i++)
62   {
63     // zero pad the address if necessary
64     if (deviceAddress[i] < 16) Serial.print("0");
65     Serial.print(deviceAddress[i], HEX);
66   }
67 }
68
69 // function to print the temperature for a device
70 void printTemperature(DeviceAddress deviceAddress)
71 {
72   float tempC = sensors.getTempC(deviceAddress);
73   Serial.print("Temp C: ");
74   Serial.print(tempC);
75   Serial.print(" Temp F: ");
76   Serial.print(DallasTemperature::toFahrenheit(tempC));
77 }
78
79 // function to print a device's resolution
80 void printResolution(DeviceAddress deviceAddress)
81 {
82   Serial.print("Resolution: ");
83   Serial.print(sensors.getResolution(deviceAddress));
84   Serial.println();
85 }
86
87 // main function to print information about a device
88 void printData(DeviceAddress deviceAddress)
89 {
90   Serial.print("Device Address: ");
91   printAddress(deviceAddress);
92   Serial.print(" ");
93   printTemperature(deviceAddress);
94   Serial.println();
95 }
96
97 void loop(void)
98 {
99   // call sensors.requestTemperatures() to issue a global temperature
100  // request to all devices on the bus
101  Serial.print("Requesting temperatures...");
102  sensors.requestTemperatures();
103  Serial.println("DONE");
104
105  for(int i=0;i<sensors.getDeviceCount();i++)
```

```
106  {
107      printData(Device_add[i]);
108  }
109
    }
```

- **Step 4.** Upload the demo. If you do not know how to upload the code, please check How to upload code.
- **Step 5.** Open the **Serial Monitor** of Arduino IDE by click **Tool-> Serial Monitor**. Or tap the Ctrl+Shift+M key at the same time. Then set the baud rate **115200**.

Success

If every thing goes well, you will get the result.

```
 1 DONE
 2 Device Address: 3B23211800ing temperatures... 77.00
 3 Requesting temperatures...DONE
 4 Device Address: 3B2321180000005C Temp C: 25.00 Temp F: 77.00
 5 Requesting temperatures...Dallas Temperature IC Control Library Demo
 6 Locating devices...Found 2 devices.
 7 Parasite power is: OFF
 8 Device [0] addr =3B4C965D06D80C98
 9 Device [1] addr =3B2321180000005C
10 Requesting temperatures...DONE
11 Device Address: 3B4C965D06D80C98 Temp C: 26.25 Temp F: 79.25
12 Device Address: 3B2321180000005C Temp C: 25.25 Temp F: 77.45
13 Requesting temperatures...DONE
14 Device Address: 3B4C965D06D80C98 Temp C: 26.25 Temp F: 79.25
15 Device Address: 3B2321180000005C Temp C: 25.00 Temp F: 77.00
16 Requesting temperatures...DONE
17 Device Address: 3B4C965D06D80C98 Temp C: 26.25 Temp F: 79.25
18 Device Address: 3B2321180000005C Temp C: 25.25 Temp F: 77.45
19 Requesting temperatures...DONE
20 Device Address: 3B4C965D06D80C98 Temp C: 26.00 Temp F: 78.80
21 Device Address: 3B2321180000005C Temp C: 25.25 Temp F: 77.45
22 Requesting temperatures...DONE
23 Device Address: 3B4C965D06D80C98 Temp C: 26.00 Temp F: 78.80
24 Device Address: 3B2321180000005C
```

Tech Support

Please do not hesitate to submit the issue into our forum.