MAX31865

Descripción

Optimizado para detectores de temperatura de resistencia de platino (RTD). Una resistencia externa establece la sensibilidad para la RTD que se utiliza y una precisión delta-sigma ADC convierte la relación de la resistencia de RTD a la resistencia de referencia en forma digital.

Características

- Conversión simple de la resistencia RTD de platino al valor digital
- Maneja RTD de platino de 100Ω a 1kΩ (PT100 a PT1000)
- Compatible con conexiones de sensores de 2, 3 y 4 hilos
- Interfaz compatible con SPI
- Resolución ADC de 15 bits; Temperatura nominal. Resolución de temperatura nominal 0.03125 ° C (varía según la no linealidad RTD)
- Precisión total en todas las condiciones de funcionamiento: 0.5 ° C (0.05% de la escala completa) como máximo
- Entradas Vref completamente diferenciales
- Tiempo de conversión de 21 ms (máx.)
- Sus entradas están protegidas contra fallas de sobre tensión grandes como ± 45V
- Detección de fallas (elemento RTD abierto, RTD cortocircuitado a voltaje fuera de rango, o corto a través de elemento RTD)

Para la detección precisa de la temperatura, nada supera a un RTD de platino. Los detectores de temperatura de resistencia (RTD) son sensores de temperatura que contienen una resistencia que cambia el valor de resistencia a medida que cambia su temperatura, básicamente un tipo de termistor. En este sensor, la resistencia es en realidad una pequeña tira de platino con una resistencia de 100 u 1000 ohmios a 0 ° C, de ahí el nombre PT100 / PT1000. En comparación con la mayoría de los termistores NTC / PTC, el tipo PT de RTD es mucho más estable y preciso.

Para obtener esa precisión del PT100x RTD, debe usarse un amplificador que esté diseñado para leer la baja resistencia. Mejor aún, tenga un amplificador que pueda ajustar automáticamente y compensar la resistencia de los cables de conexión. El MAX31865 maneja todas sus necesidades de RTD, e incluso puede compensar RTD de 3 o 4 cables para una mejor precisión. Conéctelo con cualquier microcontrolador a través de SPI.

Pines de Potencia

Vin: Pin de alimentación, el chip usa 3VDC (se incluye un regulador de voltaje)

3V3: Es la salida de 3.3V del regulador de voltaje, se puede tomar 100mA de esto si se desea

GND: Común

Pines de SPI

CLK: Entrada al chip, es el pulso que marca la sincronización. Con cada pulso de este reloj, se lee o se envía un bit.

SDO: Salida de datos en serie (Datos enviados desde el MAX31865 al controlador)

SDI: Salida de datos del Master(Controlador) y entrada de datos al Slave(MAX31865)

CS: Pin de selección de chip, estado LOW para iniciar una transmisión

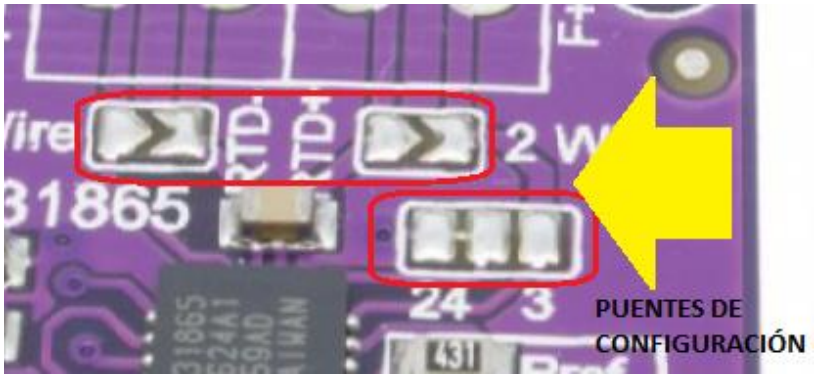
Si se desea conectar varios MAX31865 a un microcontrolador, se deben compartir los pines SDI, SDO, y SCK. Luego asignar a cada uno un pin CS único.

RDY: Salida activa en estado LOW de datos push-pull lista para datos. DRDY baja cuando hay un nuevo resultado de conversión disponible en el registro de datos. Cuando ocurre una operación de lectura de un registro de datos re resistencia RTD, DRDY regresa a alto.

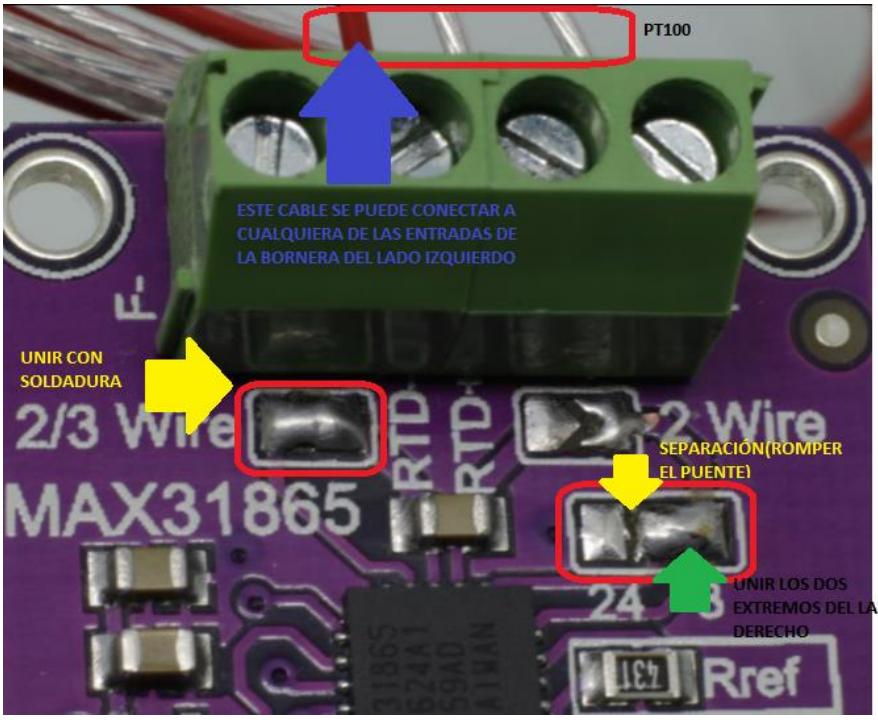
El sensor está cableado para el uso de RTD a 4 hilos, pero puede configurarse para 2 o 3 hilos muy fácilmente.

Para el uso de 4 hilos

No hacer nada con los puentes

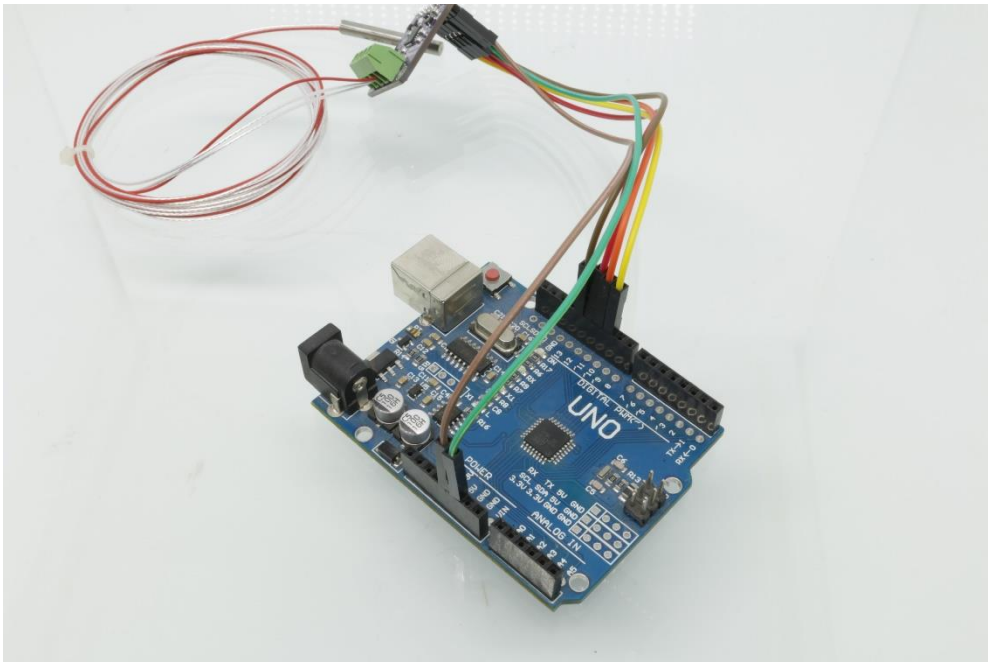


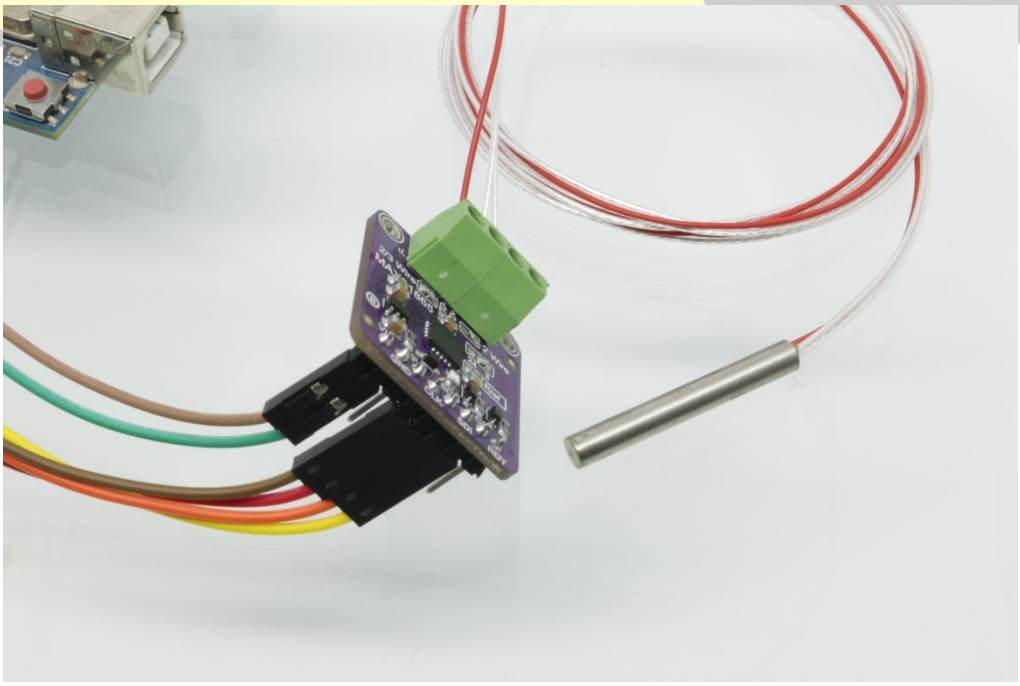
Para uso de 3 hilos



Para uso de 2 hilos

Conexiones





Código de prueba: Para probar el código descargar la librería Adafruit_MAX31865

```
#include <Adafruit_MAX31865.h>

// Use software SPI: CS, DI, DO, CLK
Adafruit_MAX31865 max = Adafruit_MAX31865(10, 11, 12, 13);
// use hardware SPI, just pass in the CS pin
//Adafruit_MAX31865 max = Adafruit_MAX31865(10);

// The value of the Rref resistor. Use 430.0!
#define RREF 430.0

void setup() {
  Serial.begin(115200);
  Serial.println("Adafruit MAX31865 PT100 Sensor Test!");

  max.begin(MAX31865_3WIRE);  // set to 2WIRE or 4WIRE as necessary
}

void loop() {
  uint16_t rtd = max.readRTD();

  Serial.print("RTD value: "); Serial.println(rtd);
  float ratio = rtd;
  ratio /= 32768;
  Serial.print("Ratio = "); Serial.println(ratio,8);
  Serial.print("Resistance = "); Serial.println(RREF*ratio,8);
  Serial.print("Temperature = "); Serial.println(max.temperature(100,
RREF));

  // Check and print any faults
  uint8_t fault = max.readFault();
  if (fault) {
    Serial.print("Fault 0x"); Serial.println(fault, HEX);
    if (fault & MAX31865_FAULT_HIGHTHRESH) {
      Serial.println("RTD High Threshold");
    }
    if (fault & MAX31865_FAULT_LOWTHRESH) {
      Serial.println("RTD Low Threshold");
    }
    if (fault & MAX31865_FAULT_REFINLOW) {
      Serial.println("REFIN- > 0.85 x Bias");
    }
    if (fault & MAX31865_FAULT_REFINHIGH) {
      Serial.println("REFIN- < 0.85 x Bias - FORCE- open");
    }
    if (fault & MAX31865_FAULT_RTDINLOW) {
      Serial.println("RTDIN- < 0.85 x Bias - FORCE- open");
    }
    if (fault & MAX31865_FAULT_OVUV) {
      Serial.println("Under/Over voltage");
    }
    max.clearFault();
  }
  Serial.println();
  delay(1000);
}
```

```
/*****************************************************************************
* Proyecto            : Medir temperatura con PT100 y MAX31865
* Placa               : Arduino UNO
* IDE                 : Arduino 1.8.5
* Autor               : Elvis Eliar Ipanaqué Villegas
* Fecha               : 06/11/2017
* Descripción  : Con el MAX31865 manejamos las necesidades de un PT100 y configuramos
las conexiones de 3 y 2 hilos
* Licencia BSD, todo este texto deberá ser incluido en cualquier distribución.
*****************************************************************************/
```