Temperature & Humidity Sensor - HDC1080 - Trēo™ Module

Module Features

- Texas Instruments HDC1080
- RoHS Compliant
- Software Library
- NightShade Trēo™ Compatible
- Breakout Headers

HDC1080 Features

(from Texas Instruments)

- Relative Humidity Accuracy ±2% (Typ)
- Temperature Accuracy ±0.2°C (Typ)
- Excellent Stability at High Humidity
- 14-bit Measurement Resolution
- 100nA Sleep Current
- Average Supply Current:
  - 710nA @ 1sps, 11-bit RH Measurement
  - 1.3µA @ 1sps, 11-bit RH and Temperature Measurement

Applications

- HVAC
- Smart Thermostats and Room Monitors
- Handheld Meters
- Medical Devices
- Wireless Sensor

Trēo™ Compatibility

Electrical

Communication	I2C
Max Current, 3.3V	8mA
Max Current, 5V	0mA

Mechanical

- 25mm x 25mm Outline
- 20mm x 20mm Hole Pattern
- M2.5 Mounting Holes

Description

The HDC1080 Trēo™ Module is a Temperature & Humidity Sensor module that features Texas Instruments's HDC1080 Temperature & Humidity Sensor. This sensor consumes little power while providing high accuracy humidity and temperature measurements. It is also compatible with any 0.1in pin interfaces (Breadboards, Wires, Headers, and more). This module is a part of the NightShade Treo system, patent pending.

Table of Contents

1 Summary

The HDC1080 is initialized by calling the begin() method and then the temperature and humidity can be read with the readTemperature() and readHumidity() methods. The measurement settings can be adjusted with the other methods in the library. The internal heater function can be used to dry the humidity sensor after a period of high humidity or to test the temperature sensor.

2 What is Trēo™?

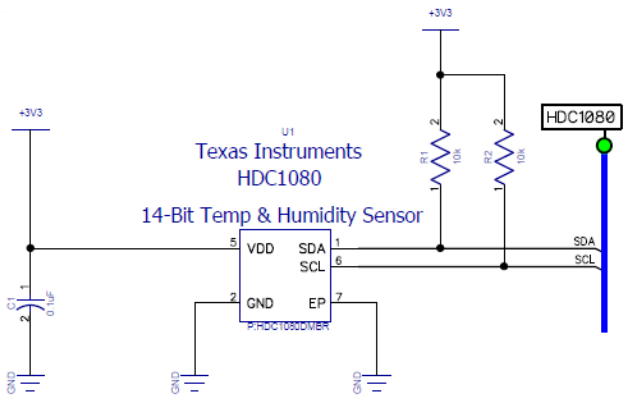
NightShade Trēo is a system of electronic modules that have standardized mechanical, electrical, and software interfaces. It provides you with a way to quickly develop electronic systems around microprocessor development boards. The grid attachment system, common connector/cabling, and extensive cross-platform software library allow you more time to focus on your application. Trēo is supported with detailed documentation and CAD models for each device.

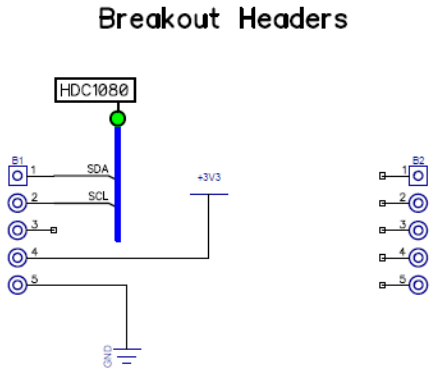
Learn more about Trēo here.

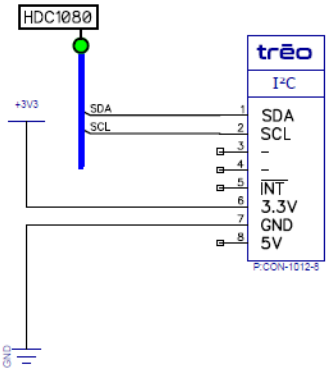
3 Electrical Characteristics

	Minimum	Nominal	Maximum
Voltages			
$V_{i/o}$ (SDA, SCL, INT)	-0.3V	-	3.6V
$V_{3.3V}$	3.1V	3.3V	3.5V
Humidity Measurement			
Response Time	-	15s	-
Sample Rate	154Hz	-	400Hz
Range	-16g	-	+16g
Precision	0.006%/LSB		0.392%/LSB
Error	-	±2%	-
Temperature Measurement			
Sample Rate	157Hz	-	274Hz
Range	-40°C	-	+125°C
Precision	0.010°C/LSB	-	0.081°C/LSB
Error	-	±0.2°C	±0.4°C
I2C Slave Address		0x40	
Operating Temperature	-25°C	-	+85°C

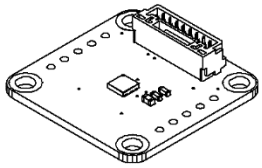
4 Electrical Schematic

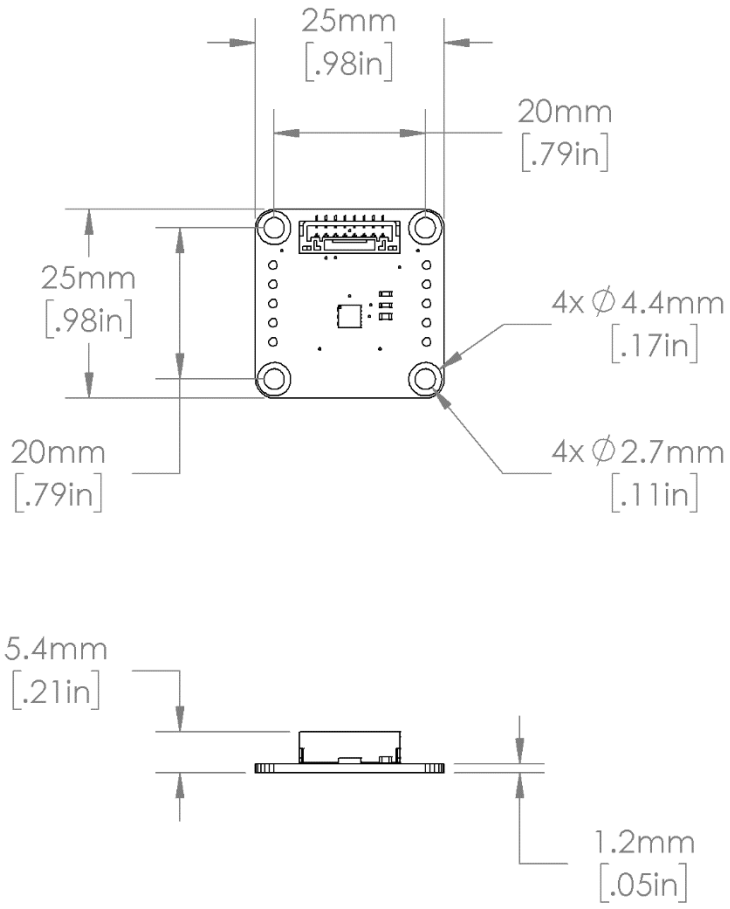






5 Mechanical Outline





6 Example Arduino Program

```
/*******************************************************
  HDC1080_TempHumidity - NightShade_Treo by NightShade Electronics

  This sketch demonstrates the functionality of the
  NightShade Trēo HDC1080 temperature and humidity sensing
  module. (NSE-1135-1) It prints the temperature and
  humidity values to Serial at 115200 baudrate.

  Created by Aaron D. Liebold
  on February 15, 2021

  Links:
  NightShade Trēo System: https://nightshade.net/treo
  Product Page: https://nightshade.net/product/treo-temp-humidity-sensor-hdc1080/

  Distributed under the MIT license
  Copyright (C) 2021 NightShade Electronics
  https://opensource.org/licenses/MIT
*******************************************************/

// Include NightShade Treo Library
#include <NightShade_Treo.h>

NightShade_Treo_HDC1080 sensor(1);

unsigned long lastHeater = 0, lastUpdate = 0;

void setup() {
  sensor.begin();
  Serial.begin(115200);

  sensor.setTemperatureResolution(0);
  sensor.setHumidityResolution(0);

  Serial.print("Device ID: 0x");
  Serial.print(sensor.readDeviceId(), HEX);
  Serial.print("\nManufacturer ID: 0x");
  Serial.print(sensor.readManufacturerId(), HEX);
  Serial.print("\nSerial ID: 0x");
  Serial.print((unsigned long) sensor.readSerialId(), HEX);
  Serial.print("\n");

  delay(1);
}

void loop() {
  if (millis() >= lastUpdate + 500) {
    lastUpdate = millis();
    sensor.acquireData(1, 1);
```

```
    Serial.print(sensor.readTemperature());
    Serial.print("C\t");
    Serial.print(sensor.readHumidity());
    Serial.print("%\n");
  }
}
```

7 Library Overview (C++ & Python)

C++ Class

NightShade_Treo_HDC1080 <classObject>();

Python Module

<classObject> = NightShade_Treo.HDC1080()

7.1 Constructors

NightShade_Treo_HDC1080(int port, uint8_t slaveAddress, uint32_t clockSpeed)

Creates a HDC1080 object.

Arguments:

port	Integer of the I2C port used (e.g. 0 = "/dev/i2c_0")
slaveAddress	7-bit slave address
clockSpeed	Desired clock speed for the bus

Returns:

Nothing

NightShade_Treo_HDC1080(int port)

Creates a HDC1080 object assuming the default slave address and clock speed.

Arguments:

port	Integer of the I2C port used. (e.g. 0 = "/dev/i2c_0")

Returns:

Nothing

7.2 Methods

reset()

Resets the module.

Arguments:

None

Returns:

Nothing

enableHeater(int enable)

Activates internal heater for temperature testing or drying humidity sensor.

Arguments:

uint8_t enable	true/false

Returns:

Nothing

setTemperatureResolution(uint8_t setting)

Set the resolution of the temperature measurement.

Arguments:

uint8_t setting	0: 14-bit measurement (6.35ms) 1: 11-bit measurement (3.65ms)

Returns:

Nothing

setHumidityResolution(uint8_t setting)

Set the resolution of the humidity measurement.

Arguments:

setting	0: 14-bit measurement (6.50ms) 1: 11-bit measurement (3.85ms) 2: 8-bit measurement (2.50ms)

Returns:

Nothing

acquireData(int readTemp, int readHumid)

Triggers a measurement of temperature and/or humidity and stores the result in a local buffer.

Arguments:

readTemp	true/false
readHumidity	true/false

Returns:

Nothing

readTemperatureRaw()

Returns the temperature value from the local buffer in the raw, 16-bit format presented in the IC. The value is interpreted as T,°C = (RawTemp)*165°C / 2^16 – 40°C.

Arguments:
None

Returns:
temperature (uint16_t)

readTemperature()

Returns the temperature value from the local buffer in degrees C as a float.

Arguments:
None

Returns:
Temperature, °C (float)

readHumidityRaw()

Returns the humidity value from the local buffer in the raw, 16-bit format presented in the IC. The value is interpreted as RH,% = (RawRH)*100% / 2^16.

Arguments:
None

Returns:
Relative Humidity (uint16_t)

readHumidity()

Returns the humidity value from the local buffer in %RH as a float.
Arguments:
None

Returns:
Relative Humidity, % (float)

readSerialId()

Returns the serial ID number from the IC.

Arguments:
None

Returns:
Serial ID (uint64_t)

readManufacturerId()

Returns the manufacturer ID number from the IC.

Arguments:
None

Returns:
Manufacturer ID (uint16_t)

readDeviceId()

Returns the device ID number from the IC.

Arguments:
None

Returns:
Device ID (uint16_t)