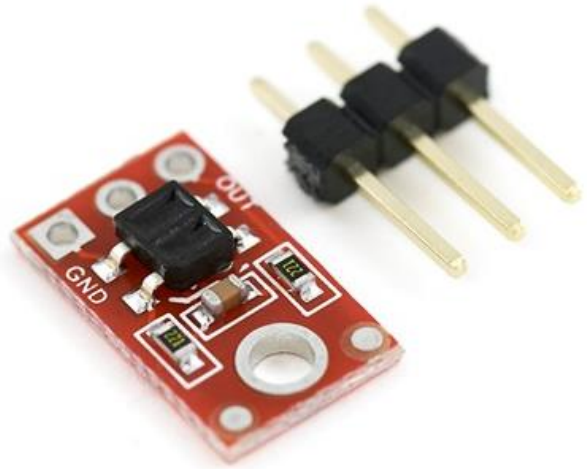
SYNACORP TECHNOLOGIES SDN. BHD. (1310487-K)
No.25 Lorong 1/SS3, Bandar Tasek Mutiara,
14120 Simpang Ampat, Penang, Malaysia.
T: +604.586.0026 F: +604.586.0026
WEBSITE: www.synacorp.my EMAIL: sales@synacorp.my

QTR-1A Reflectance Sensor Reading with Arduino



Introduction

The QTR-1A reflectance sensor carries a single infrared LED and phototransistor pair. The phototransistor is connected to a pull-up resistor to form a voltage divider that produces an analog voltage output between 0 V and VIN (which is typically 5 V) as a function of the reflected IR. Lower output voltage is an indication of greater reflection.

The LED current-limiting resistor is set to deliver approximately 17 mA to the LED when VIN is 5 V. The current requirement can be met by some microcontroller I/O lines, allowing the sensor to be powered up and down through an I/O line to conserve power.

This sensor was designed to be used with the board parallel to the surface being sensed. Because of its small size, multiple units can easily be arranged to fit various applications such as line sensing and proximity/edge detection. The reflectance measurement is output as an analog voltage.

Specifications

- Dimensions: 0.3" x 0.5" x 0.1" (without header pins installed)
- Operating voltage: 5.0 V
- Supply current: 25 mA
- Output format: analog voltage
- Output voltage range: 0 to supplied voltage
- Optimal sensing distance: 0.125" (3 mm)
- Maximum recommended sensing distance: 0.25" (6 mm)
- Weight without header pins: 0.008 oz (0.23 g)

Objective

The sensor will display the data in Serial Monitor tab.

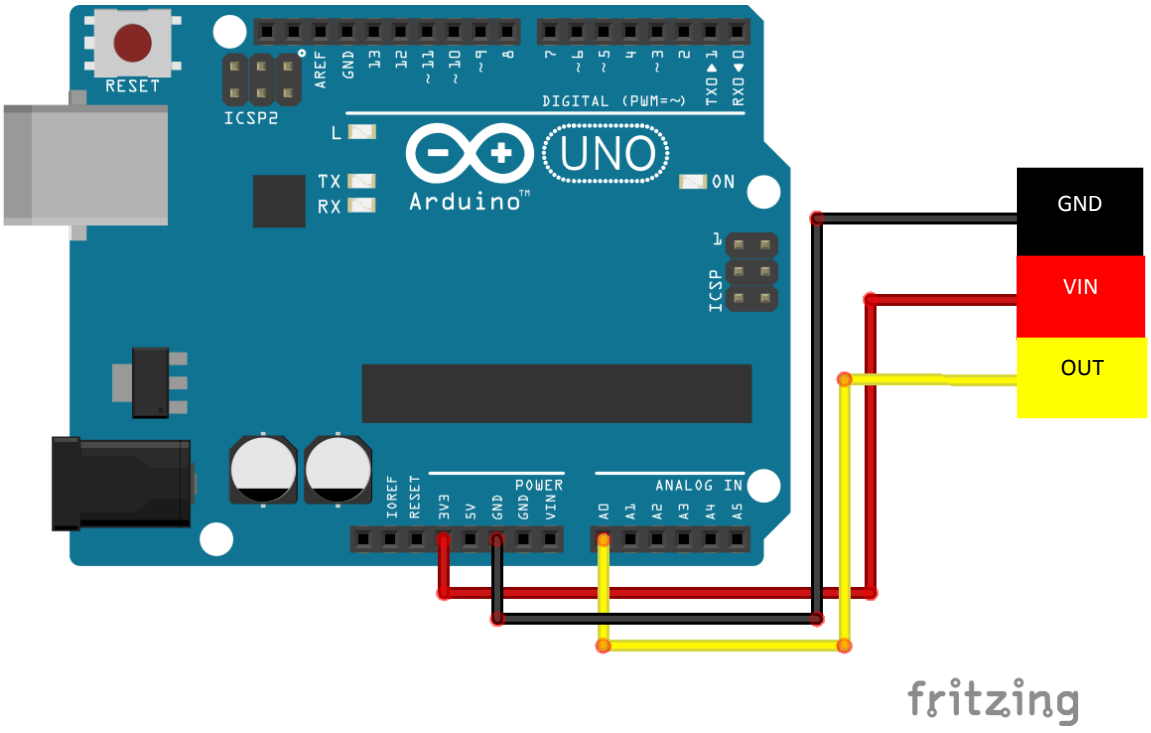
Conponents Needed

- Arduino UNO (Nano and Mega)
- QTR-1A Reflectance Sensor
- Jumper Wires

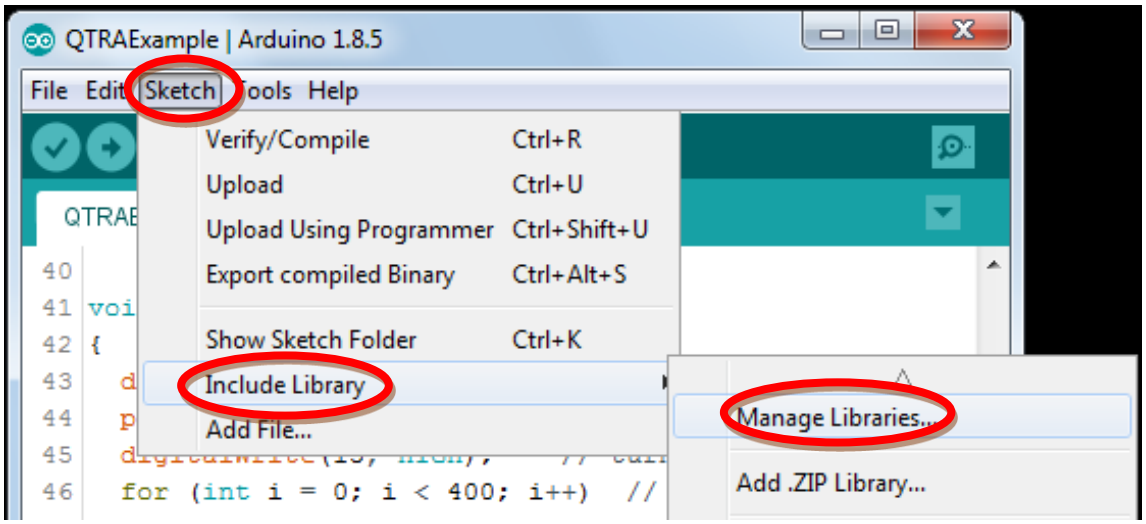
Procedures

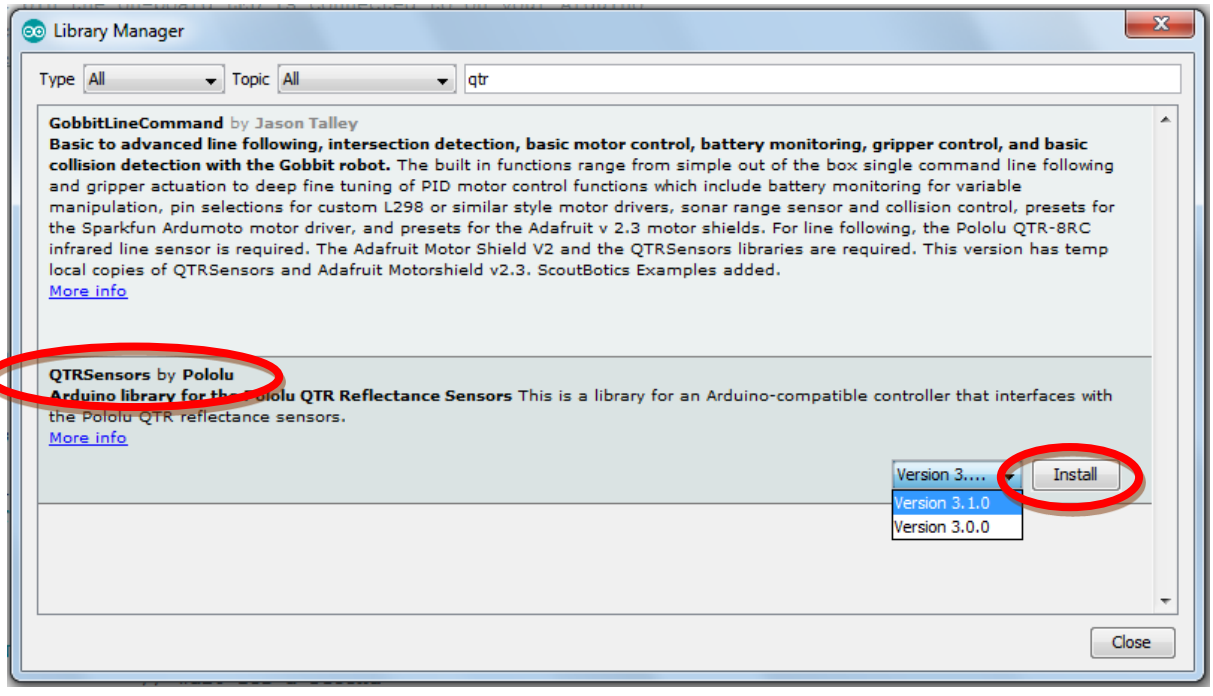
1. Connect the QTR-1A Reflectance Sensor with Arduino UNO based on picture and table provided below.

Arduino UNO	QTR-1A
A2	OUT
5V	VIN
GND	GND



2. Open Arduino IDE and instal QTRSensors Library by clicking on ***Sketch*** > ***Include Library*** > ***Manage Libraries….*** . The Library Manager popup will appear, **search for QTRSensors and click install.**



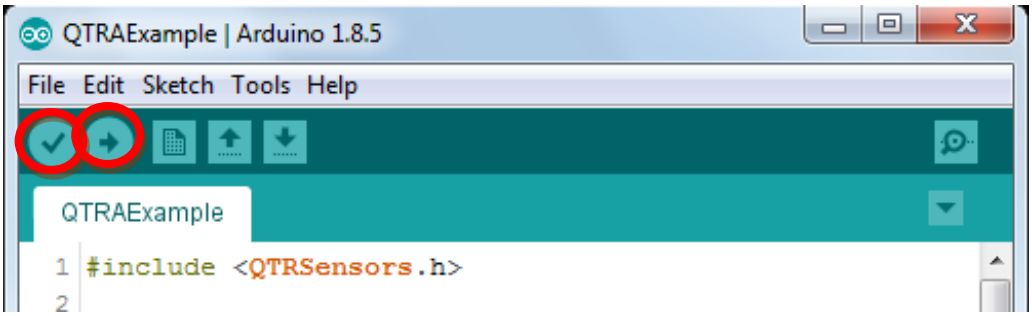


3. Still in **Arduino IDE** copy the following given program code into Arduino IDE.

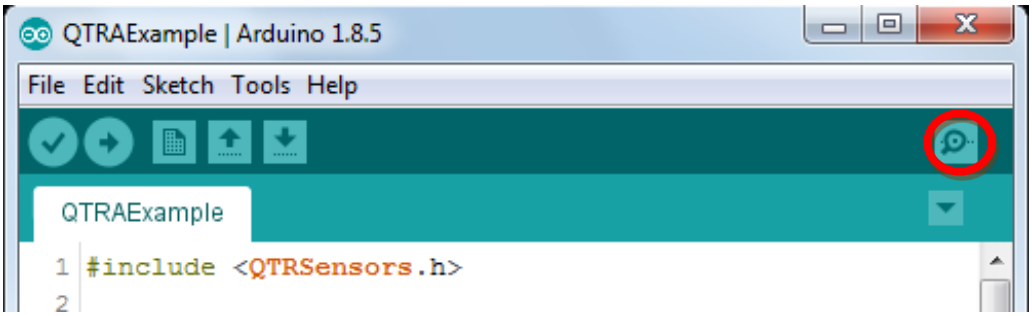
```
#include <QTRSensors.h>

// This example is designed for use with six QTR-1A sensors or the first six sensors of a
// QTR-8A module.  These reflectance sensors should be connected to analog inputs 0 to 5.
// The QTR-8A's emitter control pin (LEDON) can optionally be connected to digital pin 2,
// or you can leave it disconnected and change the EMITTER_PIN #define below from 2 to
// QTR_NO_EMITTER_PIN.
// The main loop of the example reads the calibrated sensor values and uses them to
// estimate the position of a line.  You can test this by taping a piece of 3/4" black
// electrical tape to a piece of white paper and sliding the sensor across it.  It
// prints the sensor values to the serial monitor as numbers from 0 (maximum reflectance)
// to 1000 (minimum reflectance) followed by the estimated location of the line as a number
// from 0 to 5000.  1000 means the line is directly under sensor 1, 2000 means directly
// under sensor 2, etc.  0 means the line is directly under sensor 0 or was last seen by
// sensor 0 before being lost.  5000 means the line is directly under sensor 5 or was
// last seen by sensor 5 before being lost.


#define NUM_SENSORS             1  // number of sensors used
#define NUM_SAMPLES_PER_SENSOR  4  // average 4 analog samples per sensor reading
#define EMITTER_PIN             2  // emitter is controlled by digital pin 2

// sensors 0 through 5 are connected to analog inputs 0 through 5, respectively
QTRSensorsAnalog qtra((unsigned char[]) {0, 1, 2, 3, 4, 5},
  NUM_SENSORS, NUM_SAMPLES_PER_SENSOR, EMITTER_PIN);
unsigned int sensorValues[NUM_SENSORS];


void setup()
{
  delay(500);
  pinMode(13, OUTPUT);
  digitalWrite(13, HIGH);    // turn on Arduino's LED to indicate we are in calibration mode
  for (int i = 0; i < 400; i++)  // make the calibration take about 10 seconds
  {
    qtra.calibrate();       // reads all sensors 10 times at 2.5 ms per six sensors (i.e. ~25 ms per call)
  }
  digitalWrite(13, LOW);     // turn off Arduino's LED to indicate we are through with calibration

  // print the calibration minimum values measured when emitters were on
  Serial.begin(9600);
  for (int i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(qtra.calibratedMinimumOn[i]);
    Serial.print(' ');
  }
  Serial.println();

  // print the calibration maximum values measured when emitters were on
  for (int i = 0; i < NUM_SENSORS; i++)
```

```
  {
    Serial.print(qtra.calibratedMaximumOn[i]);
    Serial.print(' ');
  }
  Serial.println();
  Serial.println();
  delay(1000);
}


void loop()
{
  // read calibrated sensor values and obtain a measure of the line position from 0 to 5000
  // To get raw sensor values, call:
  //  qtra.read(sensorValues); instead of unsigned int position = qtra.readLine(sensorValues);
  unsigned int position = qtra.readLine(sensorValues);

  // print the sensor values as numbers from 0 to 1000, where 0 means maximum reflectance and
  // 1000 means minimum reflectance, followed by the line position
  for (unsigned char i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(sensorValues[i]);
    Serial.print('\t');
  }
  //Serial.println(); // uncomment this line if you are using raw values
  Serial.println(position); // comment this line out if you are using raw values

  delay(250);
}
```

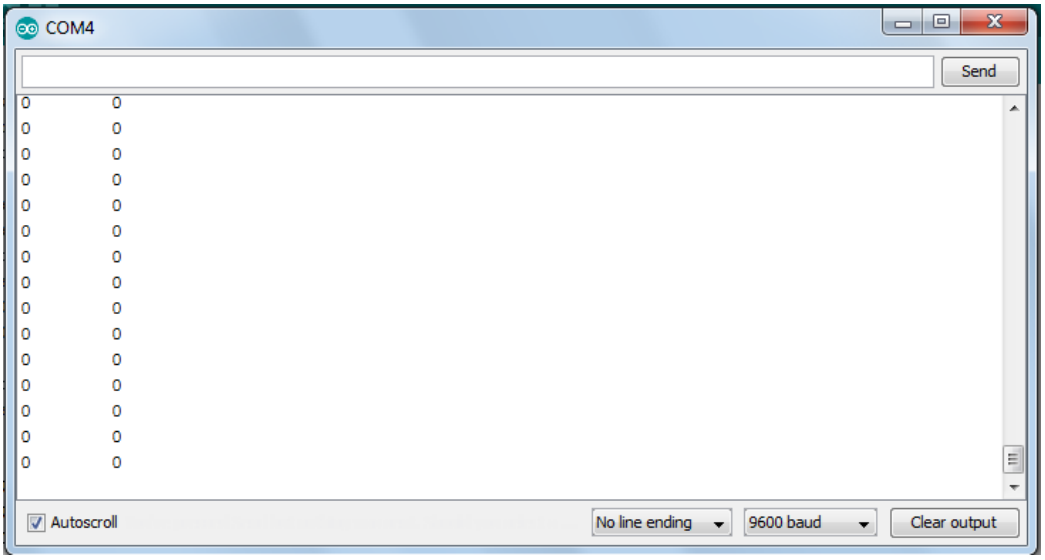
4. Connect Arduino UNO to PC click on **Verify** and then click **Upload** to upload the program sketch to the Arduino.



5. Done! Open **Serial Monitor** tab to see the QTR-1A Sensor reading.



- When sensor placed on white paper



- When sensor placed on black tape