Documentación Técnica

Sensor de distancia Time of Flight / vl53l0/1xv2

Nicolás Amaro Muñoz Read
Instituto de Innovación y Tecnología Aplicada

Descripción del vl53l0/1xv2

El sensor Time of Flight (ToF) es un tipo de sensor óptico que mide el tiempo que tarda un fotón, o pulso de luz, en viajar desde el sensor hasta un objeto y regresar.

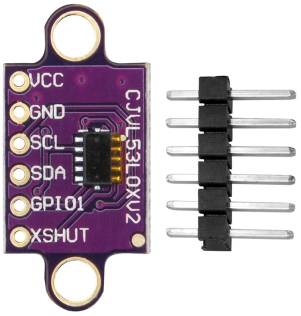
Específicamente el sensor a utilizar es el VL53L0/1XV2, fabricado por STMicroelectronics.

Características:

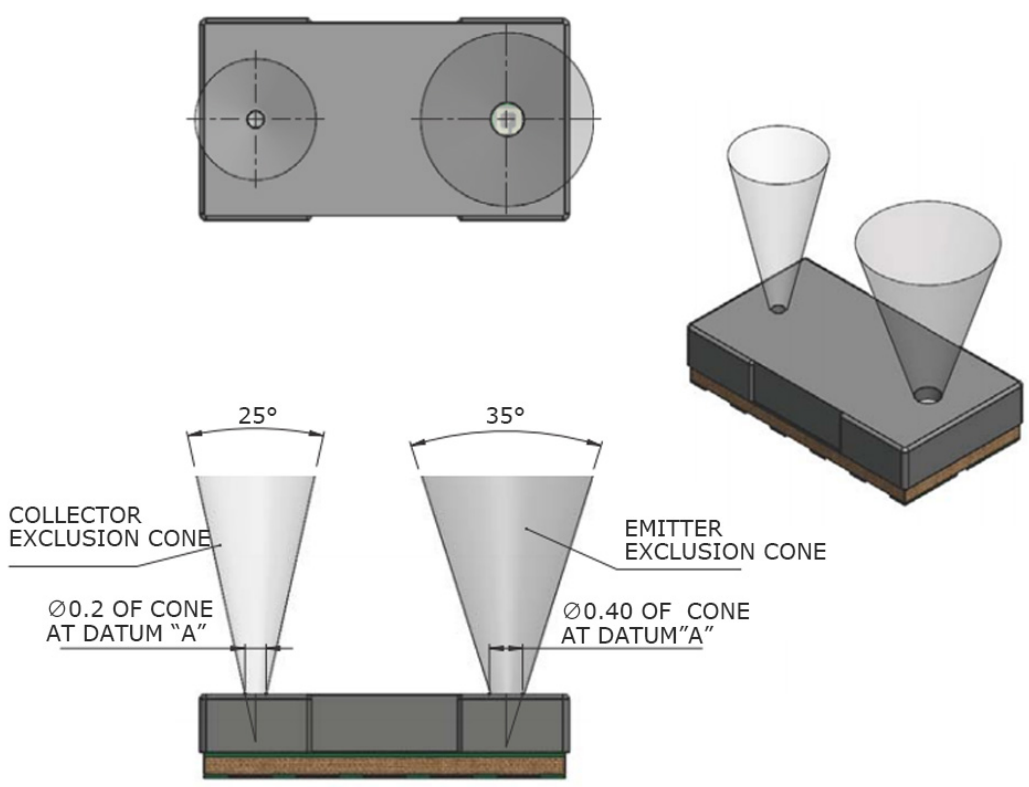
- Tiene una distancia de alcance máxima de 4 metros con una precisión de +/-5%.
- Tiene un tiempo de medición de 20 ms, permite una velocidad de actualización rápida.
- Puede funcionar con una fuente de alimentación de 2.6 V a 3.5 V.
- Se comunica con un microcontrolador a través de una interfaz I2C.
- Cuenta con un emisor láser de 940 nm VCSEL y un detector SPAD
- Tiene un ángulo de medición o FOV (Field of View) de 25°. Esto se traduce en un área de medición de 0.44 m de diámetro a una distancia de 1 m.

Datasheet:

VL53L1X Datasheet, PDF



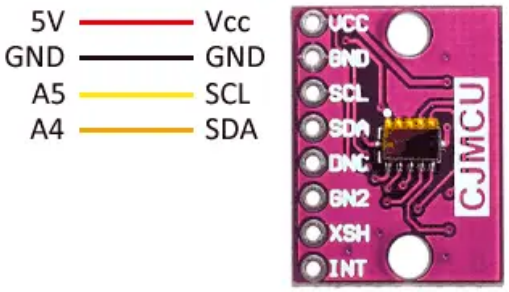
VL53L0 laser launch and receive angle

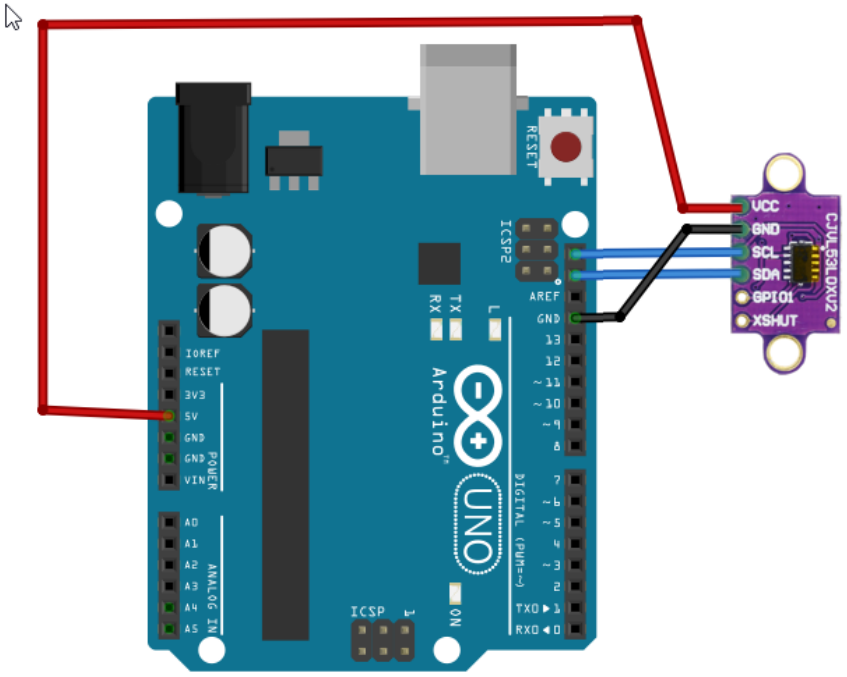


Pinout y coneccion

Para las conecciones con una placa Arduino UNO se utilizaran los pines: vin/vcc, gnd, scl, sda.

- Arduino Uno / vl53l0/1xv2
- 5v - vin/vcc #Algunas placas el pin vcc está como vin
- gnd - gnd
- scl - scl
- sda - sda





Esquema de código

Para programar el sensor ToF, es necesario tener instalada e importar la librería: “Adafruit_VL53L0X” o bien la librería “VL53L0X” que también puede funcionar.

```
#include "Adafruit_VL53L0X.h"

Adafruit_VL53L0X lox = Adafruit_VL53L0X();

void setup() {
  Serial.begin(9600);

  // Iniciar sensor
  Serial.println("VL53L0X test");
  if (!lox.begin()) {
    Serial.println(F("Error al iniciar VL53L0X"));
    while(1);
  }
}


void loop() {
  VL53L0X_RangingMeasurementData_t measure;

  Serial.print("Leyendo sensor... ");
  lox.rangingTest(&measure, false); // si se pasa true como parametro, muestra por puerto
serie datos de debug

  if (measure.RangeStatus != 4)
  {
    Serial.print("Distancia (mm): ");
   Serial.println(measure.RangeMilliMeter);
  }
  else
  {
    Serial.println("  Fuera de rango ");
  }

  delay(100);
}
```

Uso de dos o más sensores:

Tutorial conectar dos o más sensores:

https://www.youtube.com/watch?v=0glBk917HPg

Conectando dos o más sensores pueden causar interferencia por recibir y emitir pulsos al mismo tiempo, para evitar esto se pueden tomar algunas medidas como:

- Utilizar diferentes direcciones de I2C para cada sensor: los sensores VL53L0/1XV2 tienen una dirección I2C configurable, lo que significa que se pueden configurar diferentes direcciones para cada sensor. De esta manera, cada sensor puede comunicarse con el microcontrolador sin interferir con los demás sensores. Por último encender primero un sensor y luego el otro.

- Utilizar un multiplexor de bus I2C: un multiplexor de bus I2C permite conectar varios dispositivos I2C a un solo bus I2C, pero solo permite que uno de los dispositivos esté activo en un momento dado. De esta manera, se pueden conectar varios sensores VL53L0/1XV2 a un solo bus I2C sin que interfieran entre sí.

- Utilizar sensores con diferentes frecuencias de modulación: los sensores VL53L0/1XV2 pueden operar en diferentes frecuencias de modulación (10, 20, 33 y 50 MHz), lo que significa que se pueden utilizar diferentes frecuencias para cada sensor. De esta manera, se puede reducir la interferencia entre los sensores.

- Otra opción más rudimentaria es colocar un papel aluminio sobre el cono o agujero emisor y en el papel hacer un agujero un poco más pequeño que el diámetro del cono emisor para reducir la emisión de rayos infrarrojos. No es muy recomendable puesto que altera la precisión del sensor.

Cómo cambiar dirección I2C: https://www.youtube.com/watch?v=RRQASevYK3g

En este caso, para conectar y utilizar dos sensores simultáneamente, a cada uno le puse puertos I2C diferentes y un muy breve delay entre ellos. Es decir que primero enciendo el sensor 1 y lo apago, recibo el dato y lo almaceno y luego enciendo el sensor 2 y lo apago, y así sucesivamente. Para ello tuve que conectar el Xshout de cada sensor a la Arduino Uno.

Primero hay que hacer una conexión inicial de la Arduino Uno a una protoboard. Se deben conectar los pines A4 y A5, 5V y GND, y los pines 6 y 7 en caso de utilizar solo dos sensores, y luego conectar para cada uno de los sensores de la siguiente manera:

A4 - SDA
A5 - SCL
5V - VIN
Pin 7- Xshout

Para el segundo sensor se utilizará el pin 6 conectado su respectivo Xshout. Se aplican los pasos de la misma forma en caso de querer agregar más sensores.

Por último para realizar la prueba y corroborar que no existe interferencia entre los dos sensores, se puede utilizar el ejemplo de la librería Adafruit con el nombre vl5310x_dual.

```
#include "Adafruit_VL53L0X.h"

// address we will assign if dual sensor is present
#define LOX1_ADDRESS 0x30
#define LOX2_ADDRESS 0x31

// set the pins to shutdown
#define SHT_LOX1 7
#define SHT_LOX2 6

// objects for the vl5310x
Adafruit_VL53L0X lox1 = Adafruit_VL53L0X();
Adafruit_VL53L0X lox2 = Adafruit_VL53L0X();

// this holds the measurement
VL53L0X_RangingMeasurementData_t measure1;
VL53L0X_RangingMeasurementData_t measure2;

/*
    Reset all sensors by setting all of their XSHUT pins low for delay(10), then set all XSHUT high to bring out of reset
    Keep sensor #1 awake by keeping XSHUT pin high
    Put all other sensors into shutdown by pulling XSHUT pins low
    Initialize sensor #1 with lox.begin(new_i2c_address) Pick any number but 0x29 and it must be under 0x7F. Going with 0x30 to 0x3F is probably OK.
    Keep sensor #1 awake, and now bring sensor #2 out of reset by setting its XSHUT pin high.
    Initialize sensor #2 with lox.begin(new_i2c_address) Pick any number but 0x29 and whatever you set the first sensor to
 */
void setID() {
  // all reset
  digitalWrite(SHT_LOX1, LOW);
  digitalWrite(SHT_LOX2, LOW);
  delay(10);
  // all unreset
  digitalWrite(SHT_LOX1, HIGH);
  digitalWrite(SHT_LOX2, HIGH);
  delay(10);

  // activating LOX1 and resetting LOX2
  digitalWrite(SHT_LOX1, HIGH);
  digitalWrite(SHT_LOX2, LOW);

  // initing LOX1
```

```
  if(!lox1.begin(LOX1_ADDRESS)) {
    Serial.println(F("Failed to boot first VL53L0X"));
    while(1);
  }
  delay(10);

  // activating LOX2
  digitalWrite(SHT_LOX2, HIGH);
  delay(10);

  //initing LOX2
  if(!lox2.begin(LOX2_ADDRESS)) {
    Serial.println(F("Failed to boot second VL53L0X"));
    while(1);
  }
}

void read_dual_sensors() {

  lox1.rangingTest(&measure1, false); // pass in 'true' to get debug data printout!
  lox2.rangingTest(&measure2, false); // pass in 'true' to get debug data printout!

  // print sensor one reading
  Serial.print(F("1: "));
  if(measure1.RangeStatus != 4) {     // if not out of range
    Serial.print(measure1.RangeMilliMeter);
  } else {
    Serial.print(F("Out of range"));
  }

  Serial.print(F(" "));

  // print sensor two reading
  Serial.print(F("2: "));
  if(measure2.RangeStatus != 4) {
    Serial.print(measure2.RangeMilliMeter);
  } else {
    Serial.print(F("Out of range"));
  }

  Serial.println();
}

void setup() {
  Serial.begin(9600);

  // wait until serial port opens for native USB devices
  while (! Serial) { delay(1); }

  pinMode(SHT_LOX1, OUTPUT);
  pinMode(SHT_LOX2, OUTPUT);

  Serial.println(F("Shutdown pins inited..."));
```

```
  digitalWrite(SHT_LOX1, LOW);
  digitalWrite(SHT_LOX2, LOW);

  Serial.println(F("Both in reset mode...(pins are low)"));


  Serial.println(F("Starting..."));
  setID();

}

void loop() {

  read_dual_sensors();
  delay(100);
}
```

Links:

https://www.waveshare.com/w/upload/0/01/VL53L0X-Distance-Sensor-User-Manual-en.pdf

https://osoyoo.com/2019/04/21/arduino-lesson-vl53l0x-time-of-flight-distance-sensor/

https://community.st.com/s/question/0D50X0000BD3xA4SQJ/adding-multiple-vl53l0x-sensors