SYNACORP
Your Strategic Solutions Provider

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
www.synacorp.my | Email: sales@synacorp.com.my

REF: B12-AIESMSM

Arduino IR Encoder Speed Motion Sensor Module

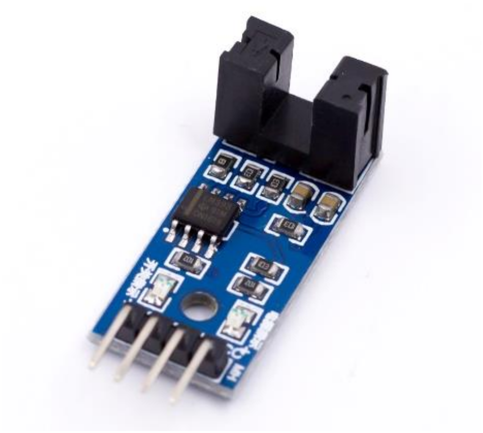


Diagram 1.0

Description

There are two columns visible in this sensor, and one of them has an IR diode. A phototransistor is located in the other column. Upon activating the sensor, a connection is established between these two columns. In other words, the phototransistor absorbs the infrared radiation that the infrared diode emits. When the sensor is not blocked, the LED will light up and the digital output will be in the LOW state. When the LED is off and the sensor is covered, the digital output will be in the HIGH state. The sensor's sensitivity can be changed using the on-board potentiometer. The sensor then sends a signal to us. In other words, we can obtain it in an analog or digital format. Additionally, the **LM393** comparator IC in this sensor module can provide digital readings.

Specifications

- Operating Voltage: 3.3V to 5V DC
- Signal Output: Digital
- Detection Method: Infrared (IR) reflection
- Output Type: Pulse signal
- Output Signal Frequency: Depending on the number of encoder pulses
- Detection Distance: Typically 1-10 mm (depends on the reflective surface and module design)
- Output Form: Digital level output
- Response Time: Fast, suitable for high-speed rotation detection
- Operating Temperature Range: -10°C to 70°C
- Dimensions: Varies by manufacturer, typically around 32mm x 14mm x 12mm

Pin Configuration

- VCC: Power supply (3.3V-5V)
- GND: Ground
- DO: Digital output signal (pulses)
- AO: Analog Output Signal (pulse)

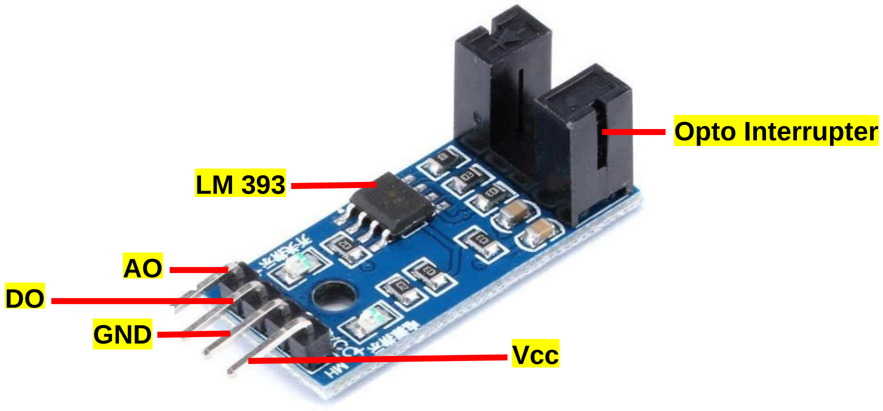


Diagram 1.1

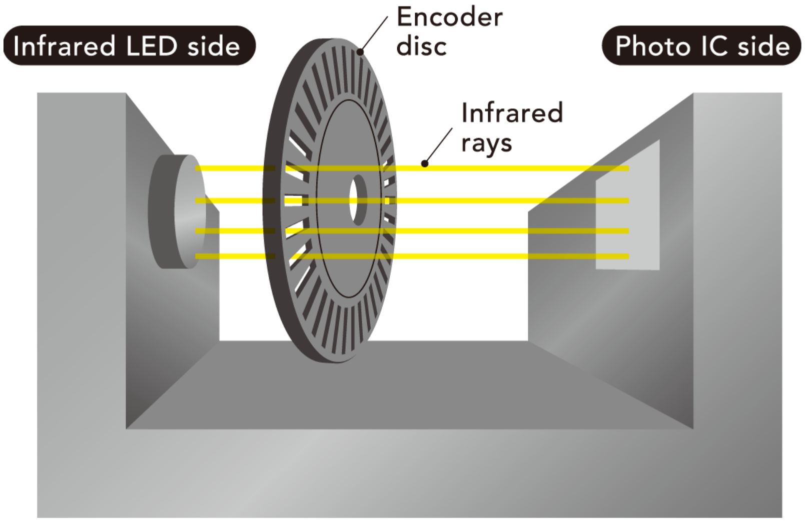
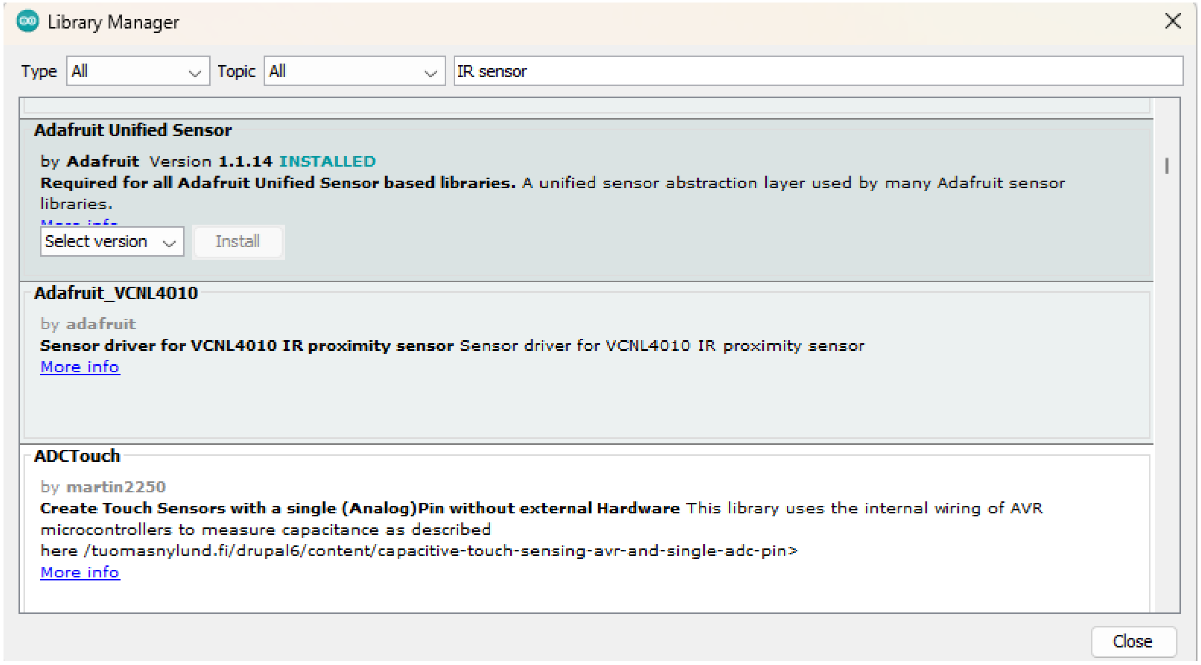


Diagram 1.2

Library

Install the **Adafruit Unified Sensor**.



IR sensor

Adafruit Unified Sensor

by **Adafruit** Version **1.1.14** **INSTALLED**
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
More info

Adafruit_VCNL4010

by **adafruit**
Sensor driver for VCNL4010 IR proximity sensor Sensor driver for VCNL4010 IR proximity sensor
More info

ADCTouch

by **martin2250**
Create Touch Sensors with a single (Analog)Pin without external Hardware This library uses the internal wiring of AVR microcontrollers to measure capacitance as described here /tuomasnylund.fi/drupal6/content/capacitive-touch-sensing-avr-and-single-adc-pin>
More info

Circuit diagram

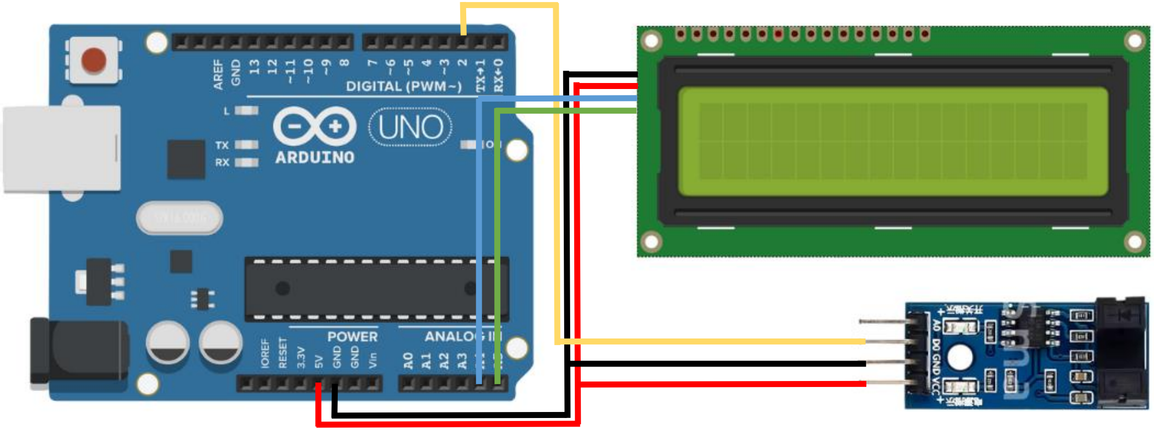
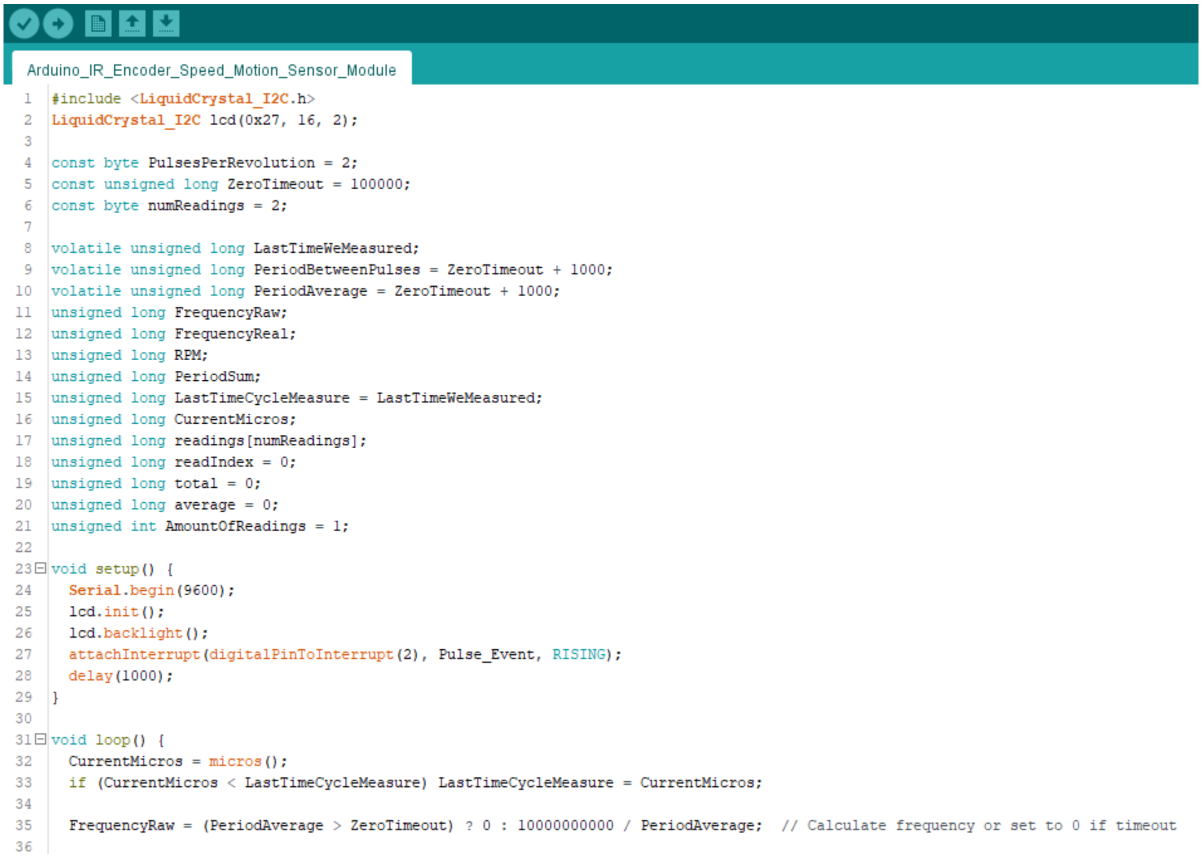
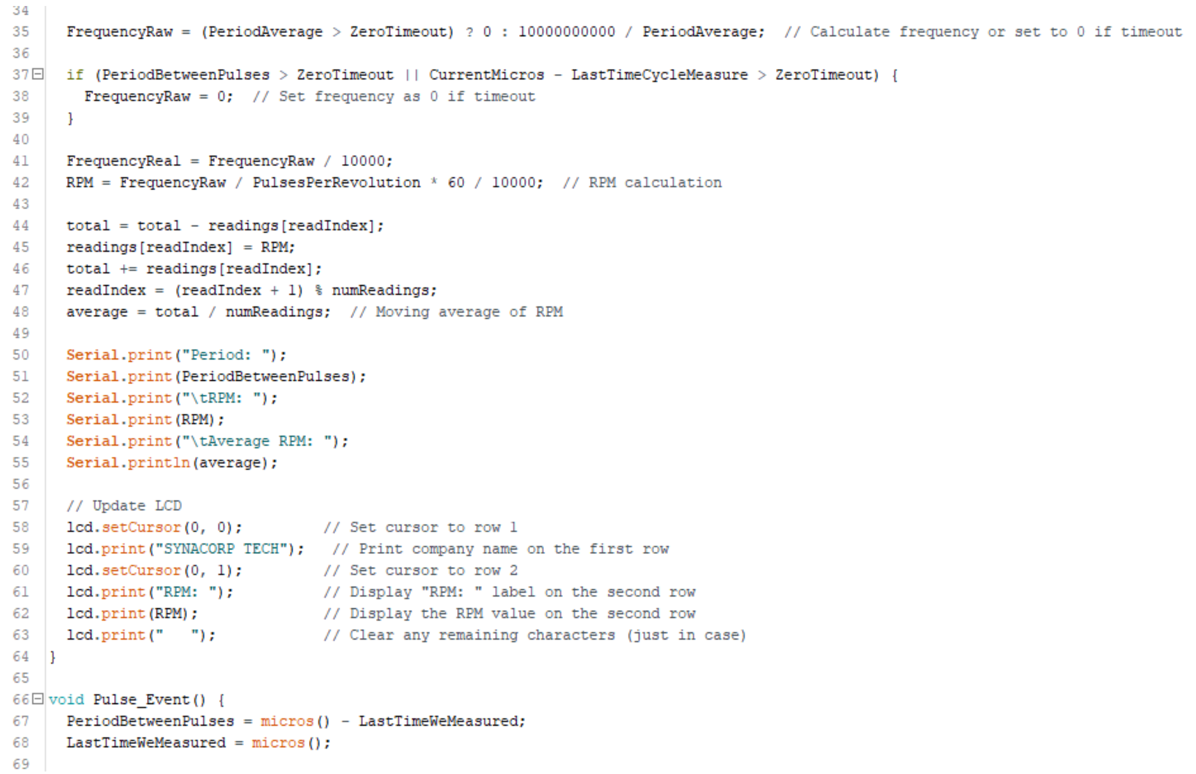


Diagram 1.3

Coding

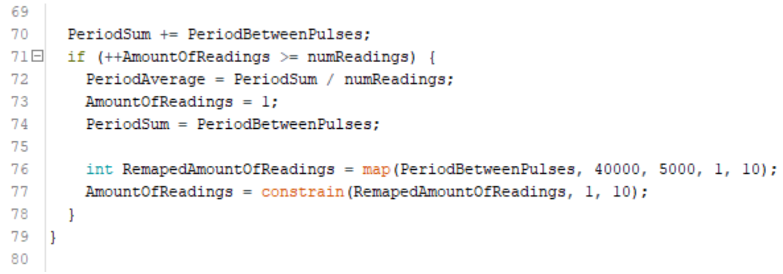


```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

const byte PulsesPerRevolution = 2;
const unsigned long ZeroTimeout = 100000;
const byte numReadings = 2;

volatile unsigned long LastTimeWeMeasured;
volatile unsigned long PeriodBetweenPulses = ZeroTimeout + 1000;
volatile unsigned long PeriodAverage = ZeroTimeout + 1000;
unsigned long FrequencyRaw;
unsigned long FrequencyReal;
unsigned long RPM;
unsigned long PeriodSum;
unsigned long LastTimeCycleMeasure = LastTimeWeMeasured;
unsigned long CurrentMicros;
unsigned long readings[numReadings];
unsigned long readIndex = 0;
unsigned long total = 0;
unsigned long average = 0;
unsigned int AmountOfReadings = 1;

void setup() {
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
  attachInterrupt(digitalPinToInterrupt(2), Pulse_Event, RISING);
  delay(1000);
}

void loop() {
  CurrentMicros = micros();
  if (CurrentMicros < LastTimeCycleMeasure) LastTimeCycleMeasure = CurrentMicros;

  FrequencyRaw = (PeriodAverage > ZeroTimeout) ? 0 : 10000000000 / PeriodAverage;  // Calculate frequency or set to 0 if timeout

```



  FrequencyRaw = (PeriodAverage > ZeroTimeout) ? 0 : 10000000000 / PeriodAverage; // Calculate frequency or set to 0 if timeout

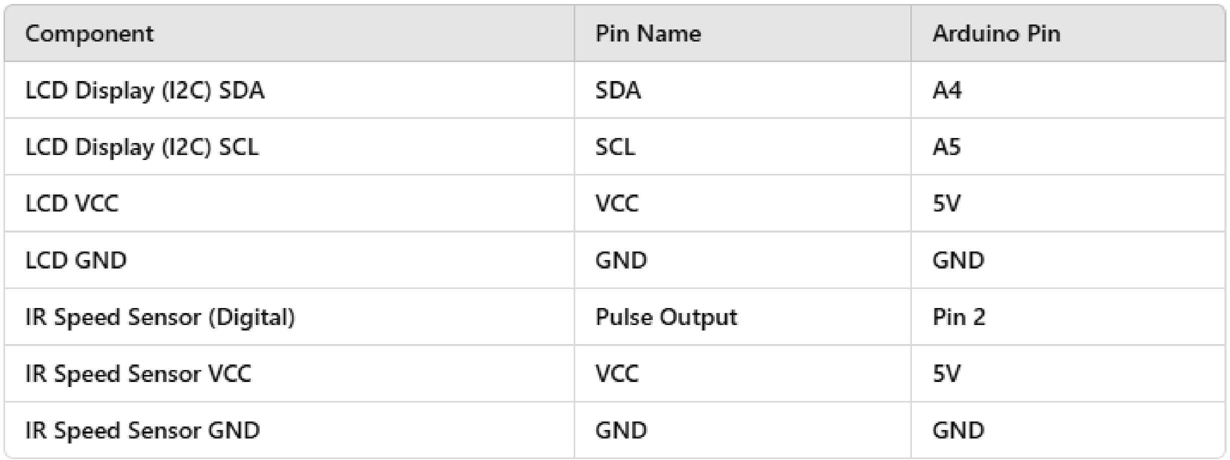
```

  if (PeriodBetweenPulses > ZeroTimeout || CurrentMicros - LastTimeCycleMeasure > ZeroTimeout) {
    FrequencyRaw = 0;  // Set frequency as 0 if timeout
  }

  FrequencyReal = FrequencyRaw / 10000;
  RPM = FrequencyRaw / PulsesPerRevolution * 60 / 10000;  // RPM calculation

  total = total - readings[readIndex];
  readings[readIndex] = RPM;
  total += readings[readIndex];
  readIndex = (readIndex + 1) % numReadings;
  average = total / numReadings;  // Moving average of RPM

  Serial.print("Period: ");
  Serial.print(PeriodBetweenPulses);
  Serial.print("\tRPM: ");
  Serial.print(RPM);
  Serial.print("\tAverage RPM: ");
  Serial.println(average);

  // Update LCD
  lcd.setCursor(0, 0);         // Set cursor to row 1
  lcd.print("SYNACORP TECH");   // Print company name on the first row
  lcd.setCursor(0, 1);         // Set cursor to row 2
  lcd.print("RPM: ");          // Display "RPM: " label on the second row
  lcd.print(RPM);              // Display the RPM value on the second row
  lcd.print("   ");            // Clear any remaining characters (just in case)
}

void Pulse_Event() {
  PeriodBetweenPulses = micros() - LastTimeWeMeasured;
  LastTimeWeMeasured = micros();

```



```

  PeriodSum += PeriodBetweenPulses;
  if (++AmountOfReadings >= numReadings) {
    PeriodAverage = PeriodSum / numReadings;
    AmountOfReadings = 1;
    PeriodSum = PeriodBetweenPulses;

    int RemapedAmountOfReadings = map(PeriodBetweenPulses, 40000, 5000, 1, 10);
    AmountOfReadings = constrain(RemapedAmountOfReadings, 1, 10);
  }
}

```

- End of coding-

Wiring



Component	Pin Name	Arduino Pin
LCD Display (I2C) SDA	SDA	A4
LCD Display (I2C) SCL	SCL	A5
LCD VCC	VCC	5V
LCD GND	GND	GND
IR Speed Sensor (Digital)	Pulse Output	Pin 2
IR Speed Sensor VCC	VCC	5V
IR Speed Sensor GND	GND	GND