Elective-1-- IoT&ES Lab Assignments

(Perform any two Lab assignments)

Laboratory Assignment-1

Title: Understanding the connectivity of Arduino with IR Sensor.

Objective: Write an application to detect obstacle and notify user using LEDS

Hardware/Software components:

1. Arduino UNO Board:
2. IR (Infrared) Sensor
3. LEDs, Bread Board, & Jumper wires, etc
4. Arduino IDE

1.Arduino Uno Board:

Introduction:

Arduino is an Italian open-source hardware and software company, project, and user community that designs and manufactures Single-Board Microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC BY-SA license, while the software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL),[1] permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.[2]

Arduino board designs use a variety of microcontrollers (Microchip ATmega328P, Atmega16U2). The ATmega328P serves as the brain of the board aand executes the uploaded code. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including USB on some models, which are also used for loading programs. The microcontrollers can be programmed using the C and Embedded C, using a standard API which is also known as the **Arduino Programming Language**, inspired by the Processing language and used with a modified version of the Processing IDE. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool developed in Go.

Following Figure1.1 shows the Arduino UNO Board with Input/output Pin diagram.

Pin0-13 are digital input/output pins.

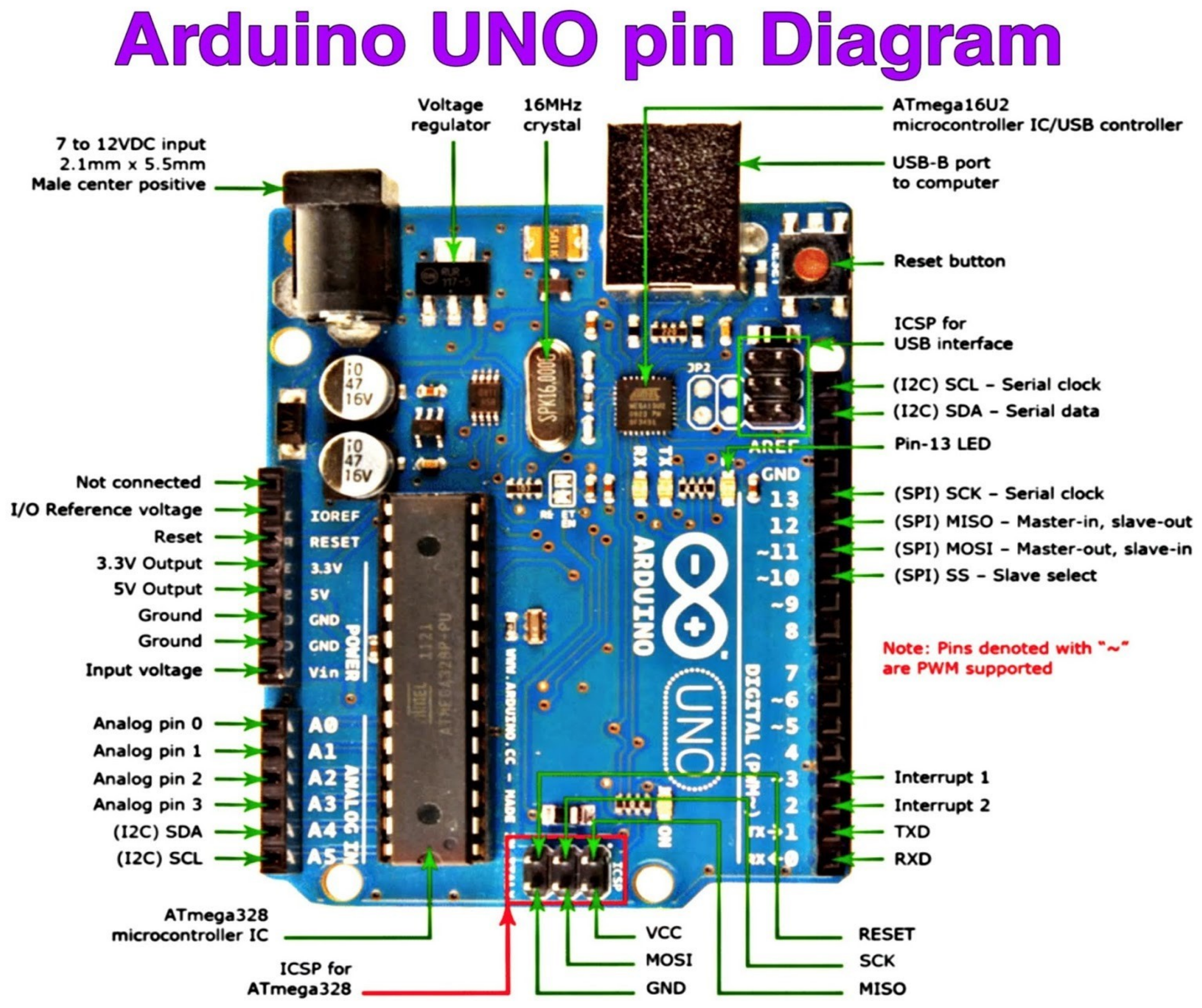


Figure1.1 Aruino Pin Diagram

IR (Infrared) Sensor/Proximity sensor

IR Sensor works by emitting infrared signal/radiation and receiving of the signal when the signal bounces back from any obstacle. In other words, the IR Sensor works by continuously sending signal (in a direction) and continuously receive signal, if comes back by bouncing on any obstacle in the way. It can also be used for proximity detection. IR technology is used in a wide range of wireless applications which includes remote controls and sensing. The infrared part in the electromagnetic spectrum can be separated into three main regions: near IR, mid-IR & far IR. The wavelengths of these three regions vary based on the application. For the near IR region, the wavelength ranges from 700 nm- 1400 nm, the wavelength of the mid-IR region ranges from 1400 nm – 3000 nm & finally for the far IR region, the wavelength ranges from 3000 nm – 1 mm.

Figure1.2 shows the IR sensor module. The IR sensor module includes five essential parts like Emitter (IR-Tx), Receiver(IR-Rx), Indicator (output LED), OUTpin, VCC and GND. The pin configuration of the IR sensor module is discussed below.

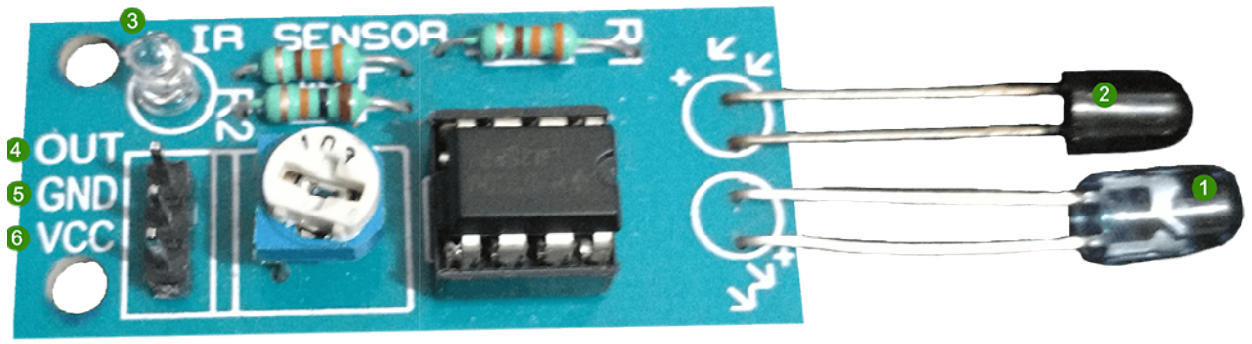


Figure1.2 Components: IR Sensor

1. **Emitter**: This component continuously emits the infrared signal
2. **Receiver**: It waits for the signal which is bounced back by obstacle
3. **Indicator**: On board LED to signal if obstacle is deducted by the sensor
4. **Output**: Could be used as Input for further processing of the signal
5. **Ground**: Ground/Negative point of the circuit
6. **Voltage**: Input 3.3V

Circuit: To detect obstacles

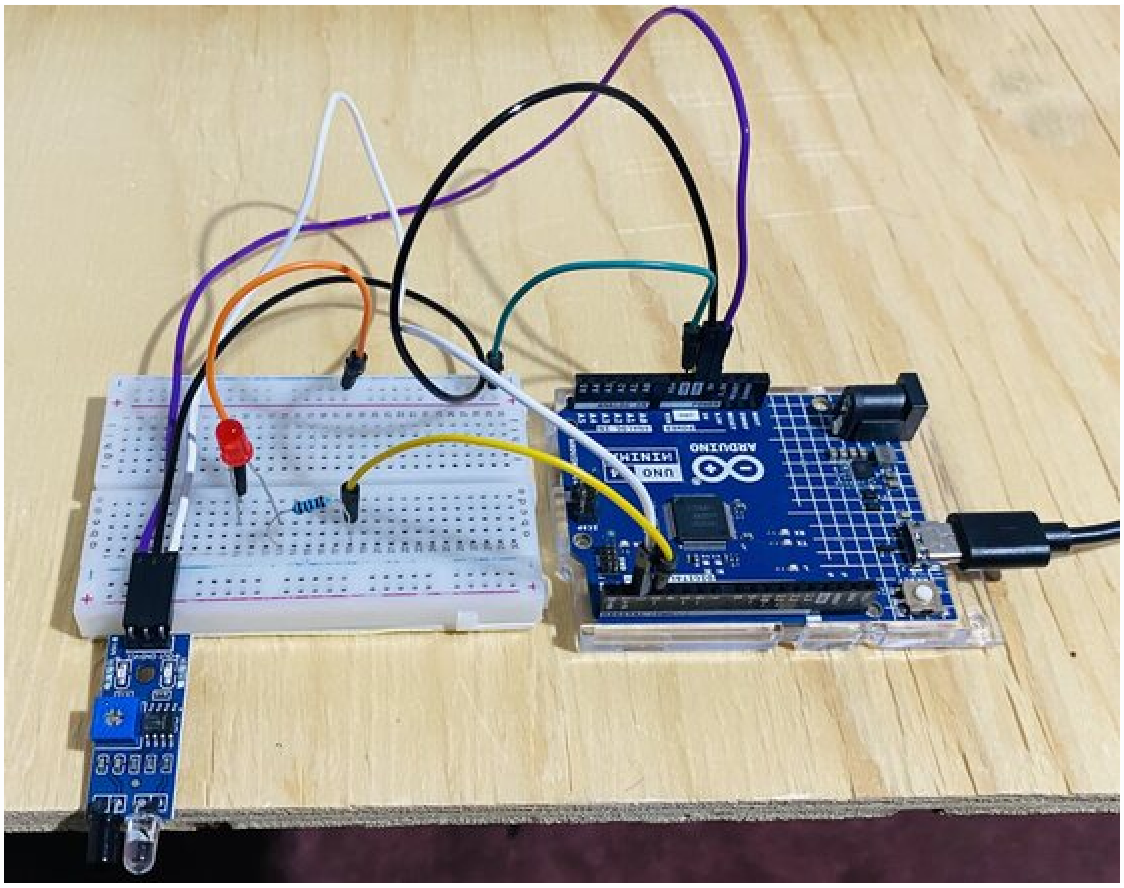
The circuit will be created which will turn on the LED when an obstacle is detected. And, as soon as the obstacle is removed from the way the LED will turn off. In order to achieve that follow the steps to create required circuit.

Part 1: Connecting IR Sensor

IR Sensor has 3 pins, viz VCC, GND and OUT.

We will use digital IO-4 for receiving input from the sensor and digital IO pin 13 for outputting object detected status through LED.

1. Connect digital IO-13 from the Arduino to LED through Breadboard
2. Connect OUT pin of the sensor to Arduino digital IO-4 through Breadboard
3. Connect GND pin of board to IR sensor module GND pin through breadboard
4. Connect VCC pin of Ardiuno board to VCC of the IR Sensor thr' Breadboard



Part 2: Connecting LED

Objective is to turn on the LED when obstacle is detected.

1. Connect +ve point of the LED (longer pin of the LED) to the Arduino digital Io-13 thr' Breadboard
2. Connect -ve point of the LED (smaller pin of the LED) to GND thr' the Breadboard

Part 3: Code to Connect IR Sensor I/P with LED status

```
Code is written & executed in embeddedC thr’ ArduinoIDE

int irSensorPin= 4; //Pin connected to IRsensor's OUT pin
int ledPin = 13;   //Pin connected to LED

void setup() {
pinModeirSensorPin, INPUT); //Set IRsensor pin as input

pinMode(ledPin, OUTPUT); //Set LEDpin as output

Serial.begin(9600);   //Initialize serial communication
for displaying message the COMM-Port on system
}

void loop() {
int sensorValue=digitalRead(irSensorPin); //Read the IR
sensor output

if(sensorValue == LOW) {//If the sensor detects an
obstacle (output is LOW for obstacle detection)

digitalWrite(ledPin, HIGH);   //Turn on the LED
    Serial.println("Obstacle detected! LED is ON.");
  }
else {
    digitalWrite(ledPin, LOW); // Turn off the LED

    Serial.println("No obstacle. LED is OFF.");
  }
delay(1000); //Small delay to avoid rapid state changes
}
```

On terminal it will start printing the status based on the conditions.

```
No Obstacle. LED is OFF!
No Obstacle. LED is OFF!
No Obstacle. LED is OFF!
Obstacle. Detected! LED is ON.
Obstacle. Detected! LED is ON.
```

Laboratory Assignment-2

Title: Understanding the connectivity of Arduino with DHT1 temperature Sensor.

Objective: Write an application to read the environment temperature. If temperature crosses a threshold value, generate alerts using LED

Hardware/Software components:

1. Arduino UNO Board:
2. IR (Infrared) Sensor
3. LEDs, Bread Board, & Jumper wires, etc
4. Arduino IDE

Temperature Sensor: DTH11/DTH12

The DHT11 and DHT22 sensors are used to measure temperature and relative humidity. These are very popular among makers and electronics hobbyists. Figure2.1 shows DHT11/DHT22 temperature Sensors.

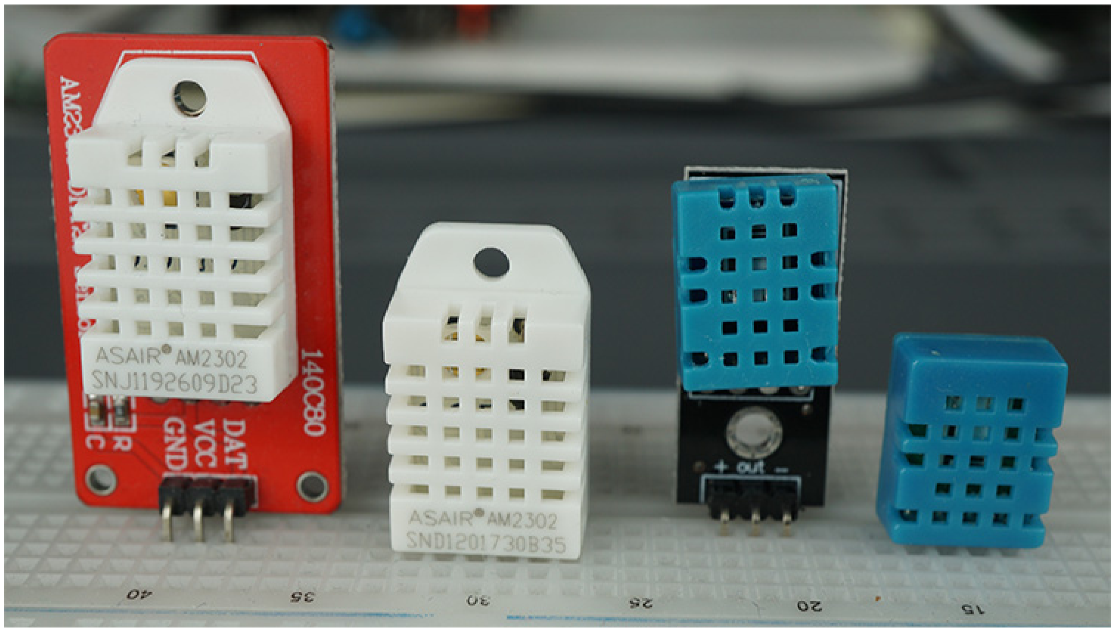
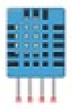
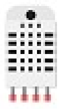


Figure2.1 DHT11/DHT22 temperature Sensors

These sensors contain a chip that does analog to digital conversion and spit out a digital signal with the temperature and humidity. This makes them very easy to use with any microcontroller.

DHT11 vs DHT22

The DHT11 and DHT22 are very similar but differ in their specifications. The following table compares some of the most important specifications of the DHT11 and DHT22 temperature and humidity sensors. For a more in-depth analysis of these sensors, please check the sensors' datasheet.

	DHT11	**DHT22**
		
Temperature range	0 to 50 ºC $^{+/-2\,ºC}$	-40 to 80 ºC $^{+/-0.5ºC}$
Humidity range	20 to 90% $^{+/-5\%}$	0 to 100% $^{+/-2\%}$
Resolution	Humidity: 1% Temperature: 1ºC	Humidity: 0.1% Temperature: 0.1ºC
Operating voltage	3 – 5.5 V DC	3 – 6 V DC
Current supply	0.5 – 2.5 mA	1 – 1.5 mA
Sampling period	1 second	2 seconds

The DHT22 sensor has a better resolution and a wider temperature and humidity measurement range. However, it is a bit more expensive, and you can only request readings with 2 seconds interval. The DHT11 has a smaller range and it's less accurate. However, you can request sensor readings every second. It's also a bit cheaper. Despite their differences, they work in a similar way, and you can use the same code to read temperature and humidity. You just need to select in the code the sensor type you're using.

DHT Pinout

DHT sensors have four pins as shown in the following Figure2.2. However, if you get your DHT sensor in a breakout board, it comes with only three pins and with an internal pull-up resistor on pin 2.

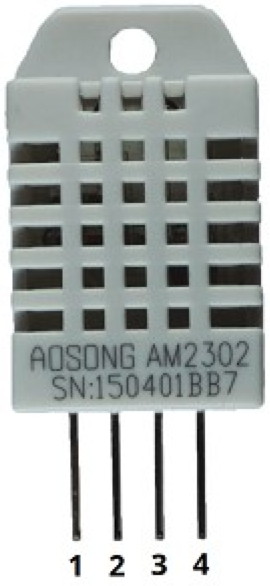


Figure2.2 DHT11 Pin diagram

The following table shows the DHT22 and DHT11 pinout. When the sensor is facing you, pin numbering starts at 1 from left to right

DHT pin	**Connect to**
1	3.3V
2	Any digital GPIO; also connect a 4.7k Ohm pull-up resistor
3	Don’t connect
4	GND

Circuit diagram: The Connection of DHT11 with Aruino is shown in Figure2.3

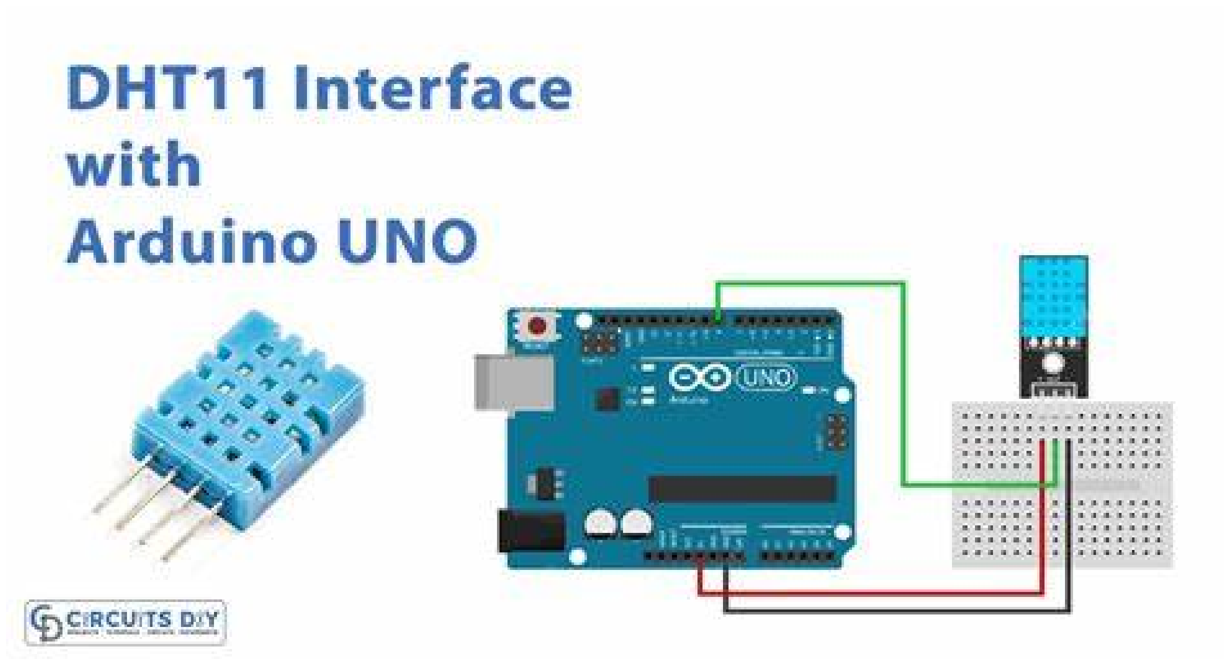


Figure2.3 Circuit connection of DHT11 with Arduino board

Code for Arduino DHT11/DHT22

The following script gets temperature from the DHT sensors and prints the readings on the COM portl.

```
#include "DHT.h"
#define DHTPIN 2//Pin where the DHT sensor is connected
#define DHTTYPE DHT11 //DHT11 or DHT22 sensor type
#define LEDPIN 13     //Pin where the LED is connected

DHT dht(DHTPIN, DHTTYPE);

void setup() {
Serial.begin(9600); //Initialize the serial communication
dht.begin();  // Initialize the DHT sensor
pinMode(LEDPIN, OUTPUT); // Set the LED pin as output
}
```

```
void loop() {

// Read temperature values
float temperature = dht.readTemperature(); //Celsius

  // Check if reading is valid
if (isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

// Print the readings to the serial monitor
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println("°C");


//Turn on LED if the temperature is above 35°C,
otherwise turn it off

if (temperature > 35) {
   digitalWrite(LEDPIN, HIGH); // Turn on LED
   Serial.println("LED ON: Temperature is above 35°C");
  }
else {
    digitalWrite(LEDPIN, LOW);  // Turn off LED
    Serial.println("LED OFF: Temperature is below or
    equal to 35°C");
  }
  //Wait 2 seconds before the next reading

delay(2000);
}
```

Output will display on terminal as:

```
Terminal:
Temp=28.3⁰C
Temp=28.3⁰C,
Temp=28.3⁰C,
…..
```

Laboratory Assignment-III

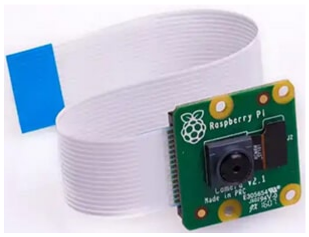
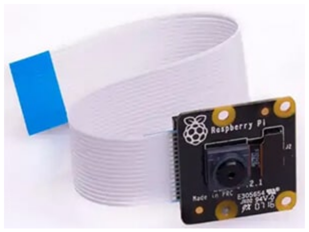
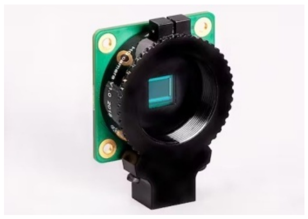
Title: Understanding and connectivity of RaspberryPi board with camera.
Objective: Write an application to capture and store the image

Hardware Components:

- Raspberry Pi board
- Raspberry Pi Camera Module sensor
- Breadboard & Jumper wires

Raspberry Pi Camera Module
Unlike USB cameras on a computer, the number of available products for Raspberry Pi is quite limited, but there is still a choice to make between these main options as given in Table3.1:

Tale 3.1 Comparison of Camera modules

	Official Raspberry Pi Camera Module	**Raspberry Pi NoIR Camera Module**	**Raspberry Pi HQ Camera**
			
Specs	8 Megapixel 3280 x 2464 1080p	8 Megapixel 3280 x 2464 1080p30 Infrared Night Vision	12.3 Megapixel Lens required 1080p

The Raspberry Pi Foundation offers three camera models:

- The first one was the first available and has been updated in 2016 (v2). **It's a high-quality camera with an 8-megapixel sensor that allows you to get HD pictures (3280 × 2464 pixels) and videos (1080p max)**. This model is compatible with any Raspberry Pi model and easy to install on Raspberry Pi OS (we'll see that later).

- **The second one (NoIR) is almost the same. The only difference is the ability to take infrared photographs.** The price is also a bit higher, but if you have a few extra dollars available, take this one. It's often used for security cameras or to take photos in a low-light environment.
- Also, the Raspberry Pi Foundation released a new high-quality camera model more recently. It's like a tiny reflex camera, and you can put additional lenses on it.

Figure3.1 shows connection of camera to raspberry Pi Board

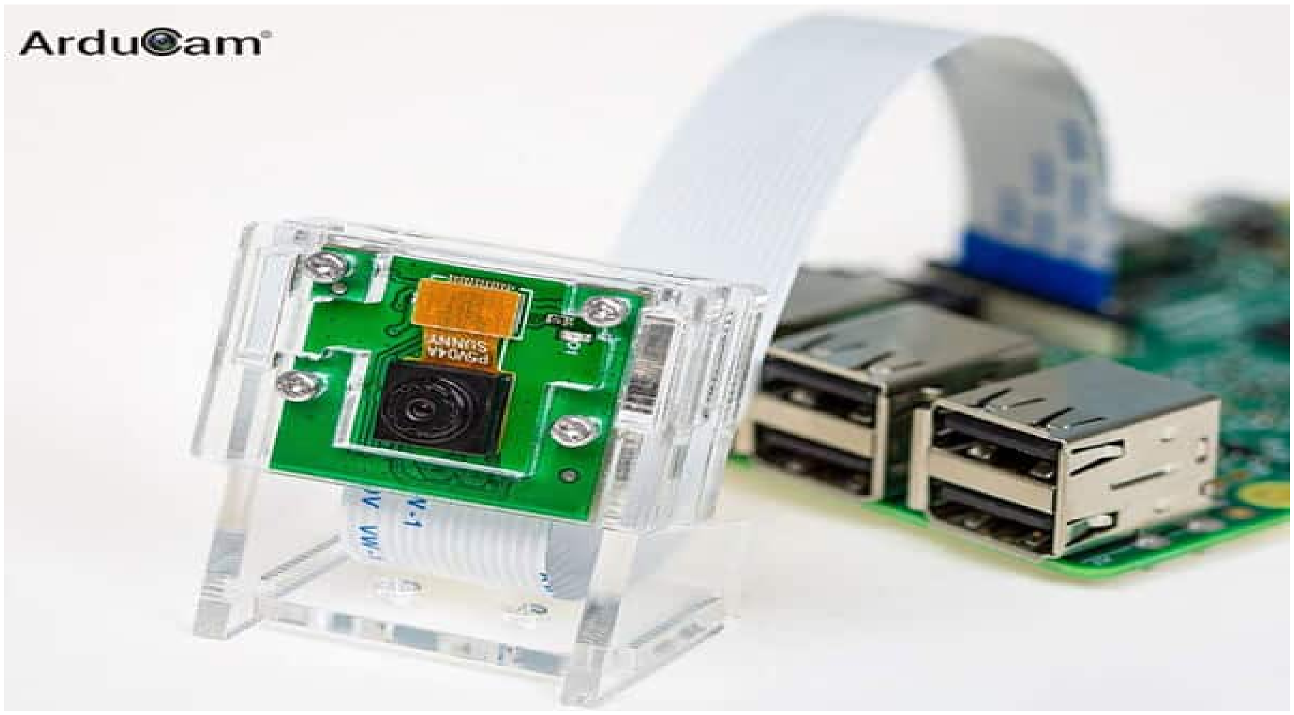


Figure3.1 Camera connection to Raspberry Pi board

Software requirement

Raspberry Pi OS (formerly known as Raspbian) is the recommended operating system to install and use a camera on Raspberry Pi.

Hardware installation

- **Take the Raspberry Pi board in hand.**
  Unplug all cables, the Raspberry Pi must be turned off and disconnected from the power supply.

- **Find the camera port on the Raspberry Pi** (between the HDMI and audio ports).
  You'll find it easily because it's the only one that fits the cable width, and it should be written "CAMERA" on the main board.
- Before plugging in the cable, you may need to remove the plastic film and **lightly pull the black plastic**.
- **Plug the cable and push the black plastic to hold the cable inside**.
  Make sure to align both connectors on the same side (cable connectors on the HDMI port side, blue side of the ribbon towards the USB ports):

  Figure3.2 shows camera Port on Raspberry Pi Board

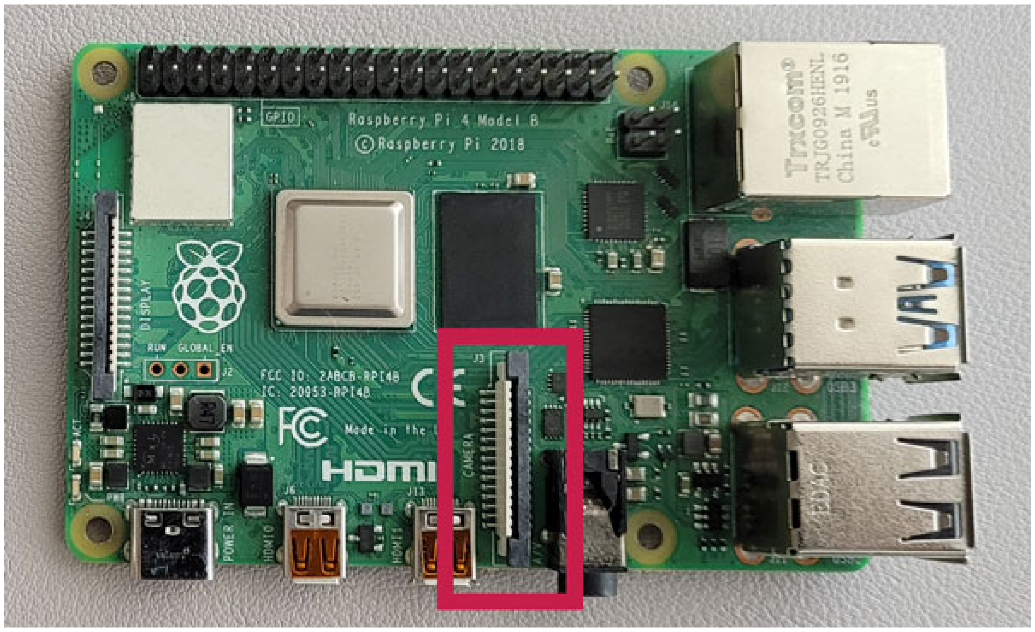


Figure3.2 Camera Port on raspberry Pi Board

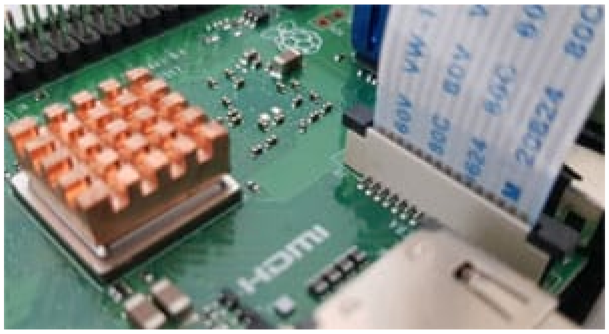


Figure3.3 Camera connector to raspberry Pi Board

Commands to Interact with the Raspberry Pi Camera

libcamera is an open-source software library aimed at supporting camera systems directly from the Linux operating system on Arm processors. Proprietary code running on the Broadcom GPU is minimised. For more information about libcamera see the libcamera website.

libcamera-hello is the equivalent of a "hello world" application for the camera. It starts the camera, displays a preview window, and does nothing else. For example:

libcamera-hello

For more parameters, please use the --help parameter to browse:

libcamera-hello --help

Take a photo

You can try is to simply take a picture of the image seen by the camera.

"`libcamera-still`" is the corresponding command on Raspberry Pi OS

To use it, the basic command line is:

```
libcamera-still -o image.jpg
```

With -o you define the target file name (where the pictures will be saved). It's possible to use a file name including the path, for example:

```
libcamera-still -o ~/Pictures/mypicture.jpg
```

Use the -h parameter to display all the possible options for this command:

```
libcamera-still -h
```

Here are a few interesting options you can try:

```
--width arg (=0) Set the output image width (0 = use default value)
--height arg (=0) Set the output image height (0 = use default value)
-t [ --timeout ] arg (=5000) Time (in ms) for which program runs
-o [ --output ] arg Set the output file name
-n [ --nopreview ] =arg(=1) Do not show a preview window-p [ --preview
] arg (=0,0,0,0) Set the preview window dimensions, given as
x,y,width,height e.g. 0,0,640,480
-f [ --fullscreen ] =arg(=1) Use a fullscreen preview window
--qt-preview =arg(=1) Use Qt-based preview window (WARNING: causes
heavy CPU load, fullscreen not
supported)
--rotation arg (=0) Request an image rotation, 0 or 180
--brightness arg (=0) Adjust the brightness of the output images, in
the range -1.0 to 1.0
--contrast arg (=1) Adjust the contrast of the output image, where 1.0
= normal contrast
--saturation arg (=1) Adjust the colour saturation of the output,
where 1.0 = normal and 0.0 =
greyscale
-q [ --quality ] arg (=93) Set the JPEG quality parameter
```

Record a video

To record a video, the command is almost the same.
"libcamera-vid" is the corresponding command name

So like for pictures, to record a video use:
```
libcamera-vid -o video.h264
```

Control the camera with Python

Python is the main programming language on Raspberry Pi. And the good news is that everything is available by default to use Python directly on Raspberry Pi OS (editors, compilers, basic libraries, ...).

Create your first script:

Install the Python camera library if needed (installed by default on Desktop):
```
sudo apt install python3-picamera
```

Python Script:
```
import time
import picamera
with picamera.PiCamera() as camera:
camera.start_preview()
```

```
time.sleep(5)
camera.capture('/home/pi/Images/python2.jpg')
camera.stop_preview()
```

```
Accessing the Raspberry Pi Camera with OpenCV
```

```
# import the necessary packages
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import cv2

# Initialize the camera and grab to the camera capture

camera = PiCamera()
rawCapture = PiRGBArray(camera)

# allow the camera to warmup
time.sleep(0.1)

# grab an image from the camera

camera.capture(rawCapture, format="bgr")
image = rawCapture.array

# display the image on screen and wait for a keypress
cv2.imshow("Image", image)
cv2.waitKey(0)
```