

Assignment 1

Title: Study of Raspberry Pi, Beagle board, Arduino, and Microcontroller

Objective: To study different IoT hardware and their features.

1. Raspberry Pi.
2. Beagle board.
3. Arduino board.

Problem Statement: Study of different embedded systems Raspberry-Pi, Beagle board & Arduino board. Understanding the peripherals Raspberry-Pi /Beagle board.

Outcomes:

*Knowledge of the Raspberry-pi, Beagle Board & Arduino

Software Requirement & Hardware Requirement:

Raspberry Pi- Raspbian.

Arduino – arduinoOS.

Beagle bone - Angstrom

Boards of Raspberry Pi, arduinoOS, Angstrom

Theory:

What Is IoT?

The Internet of Things represents a general concept for the ability of network devices to sense and collect data from the world around us, and then share that data across the Internet where it can be processed and utilized for various interesting purposes.

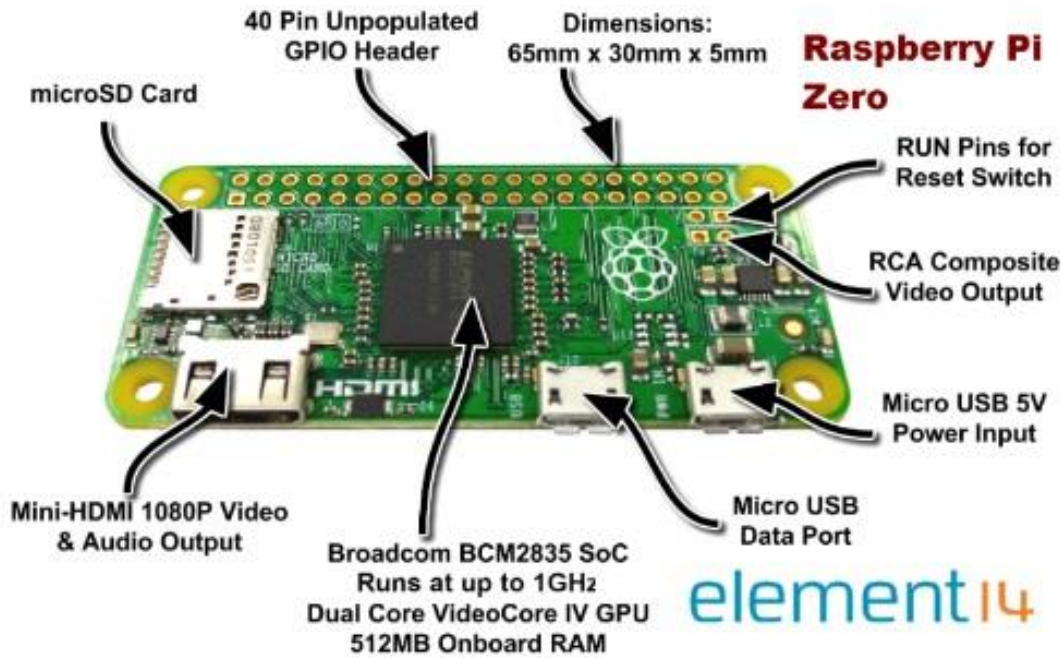
Some also use the term industrial Internet interchangeably with IoT. This refers primarily to commercial applications of IoT technology in the world of manufacturing. The Internet of Things is not limited to industrial applications, however.

Raspberry Pi:

A Raspberry Pi is a credit card-sized computer originally designed for education, inspired by the 1981 BBC Micro. Creator Eben Upton's goal was to create a low-cost device that would improve programming skills and hardware understanding at the pre-university level. But thanks to its small size and accessible price, it was quickly adopted by tinkerers, makers, and electronics enthusiasts for projects that require more than a basic microcontroller (such as Arduino devices).

The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that imply, at a low-power consumption level.

The Raspberry Pi was designed for the Linux operating system, and many Linux distributions now have a version optimized for the Raspberry Pi. Two of the most popular options are Raspbian, which is based on the Debian operating system, and Pidora, which is based on the Fedora operating system.



PIN	NAME		NAME	PIN
01	3.3V DC Power	Red	5V DC Power	02
03	GPIO02 (SDA1, I ² C)	Blue	5V DC Power	04
05	GPIO03 (SDL1, I ² C)	Blue	Ground	06
07	GPIO04 (GPCLK0)	Green	GPIO14 (TXD0, UART)	08
09	Ground	Black	GPIO15 (RXD0, UART)	10
11	GPIO17	Green	GPIO18 (PWM0)	12
13	GPIO27	Green	Ground	14
15	GPIO22	Green	GPIO23	16
17	3.3V DC Power	Red	GPIO24	18
19	GPIO10 (SP10_MOSI)	Purple	Ground	20
21	GPIO09 (SP10_MISO)	Purple	GPIO25	22
23	GPIO11 (SP10_CLK)	Purple	GPIO08 (SPI0_CE0_N)	24
25	Ground	Black	GPIO07 (SPI0_CE1_N)	26
27	GPIO00 (SDA0, I ² C)	Yellow	GPIO07 (SCL0, I ² C)	28
29	GPIO05	Green	Ground	30
31	GPIO06	Green	GPIO12 (PWM0)	32
33	GPIO13 (PWM1)	Green	Ground	34
35	GPIO19	Green	GPIO16	36
37	GPIO26	Green	GPIO20	38
39	Ground	Black	GPIO21	40

Fig 1: Raspberry-pi board and pin out

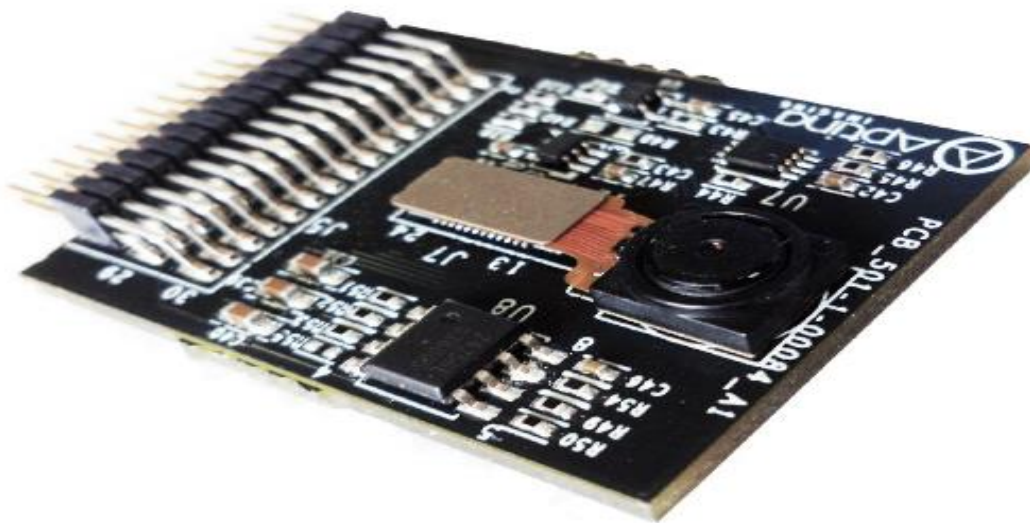
Features:

- CPU: Quad-core 64-bit ARM Cortex A53 clocked at 1.2 GHz
- GPU: 400MHz Video Core IV multimedia
- Memory: 1GB LPDDR2-900 SDRAM (i.e. 900MHz)
- USB ports: 4
- Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
- Network: 10/100Mbps Ethernet and 802.11n Wireless LAN
- Peripherals: 17 GPIO plus specific functions, and HAT ID bus

- Bluetooth: 4.1
- Power source: 5 V via Micro USB or GPIO header
- Size: 85.60mm × 56.5mm
- Weight: 45g (1.6 oz)

BEAGLEBONE:

The **Beagle Board** is a low-power open-source single-board computer produced by Texas Instruments in association with Digi-Key and Newark element14. The Beagle Board was also designed with open-source software development in mind and as a way of demonstrating the Texas Instrument's OMAP3530 system-on-a-chip. The board was developed by a small team of engineers as an educational board that could be used in colleges around the world to teach open source hardware and software capabilities. It is also sold to the public under the Creative Commons share-alike license. The board was designed using Cadence OrCAD for schematics and Cadence Allegro for PCB manufacturing; no simulation software was used. Beagle Bone's default operating system is Angstrom.



BeagleBone Black Pinout

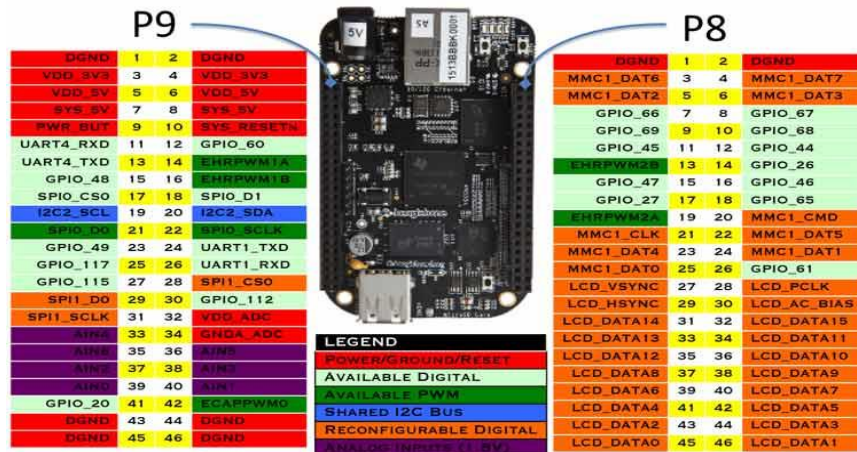


Fig 2: Beagle board and pin out

Features:

- AM3358 1GHz ARM® Cortex-A8 Processor.
- 4GB 8-bit eMMC Onboard Flash.
- 3D Graphics Accelerator.
- NEON Floating-Point Accelerator.
- 2x PRU 32-bit Microcontroller.
- Connectivity: USB Client for power and communications. USB Host. Ethernet. Micro HDMI. ...
- Software Compatibility. Angstrom Linux. Android. Ubuntu

Arduino:

Arduino is an open-source electronics platform based on easy-to-use hardware and software.

Arduino boards can read inputs – a light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, or publishing something online.

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming.

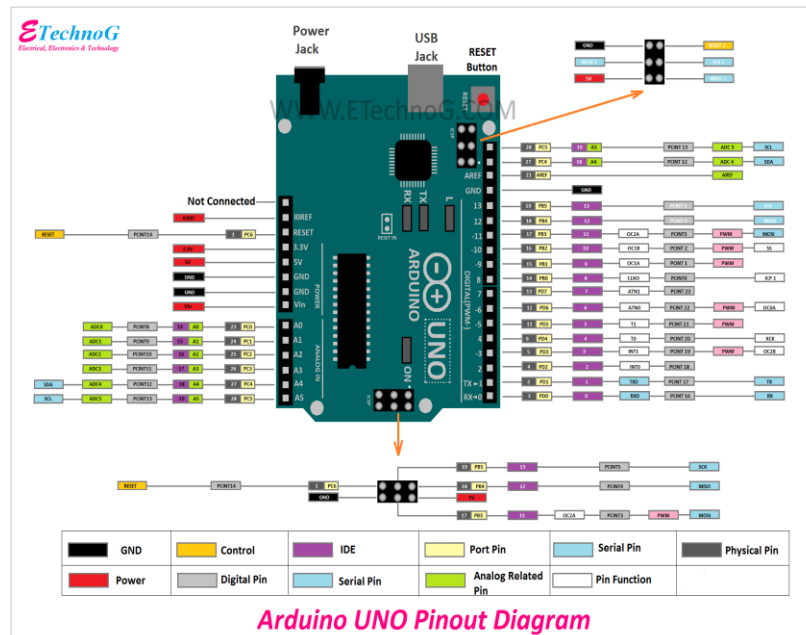
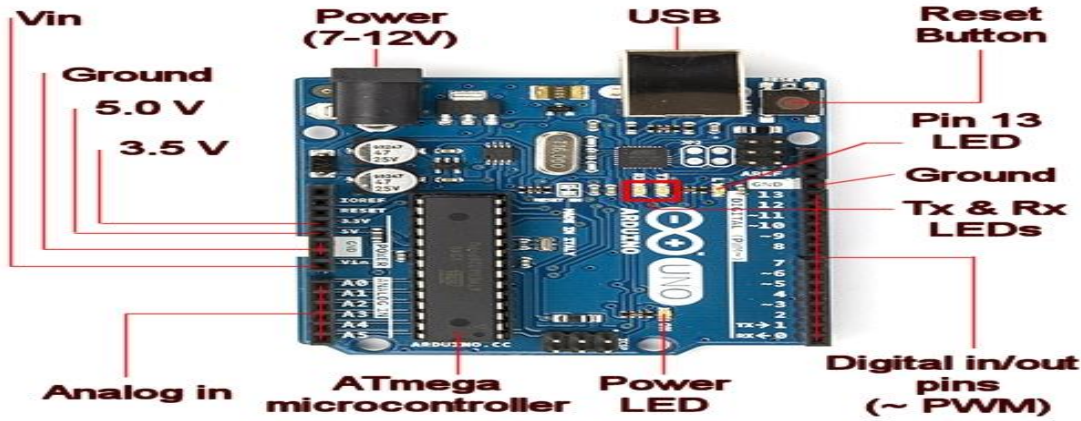


Fig 3: Arduino and pinout

Features of the Arduino UNO:

Microcontroller: ATmega328

Operating Voltage: 5V

Input Voltage (recommended): 7-12V

Input Voltage (limits): 6-20V

Digital I/O Pins: 14 (of which 6 provide PWM output)

Analog Input Pins: 6

DC Current per I/O Pin: 40 mA

DC Current for 3.3V Pin: 50 mA

Flash Memory: 32 KB of which 0.5 KB used by bootloader

SRAM: 2 KB (ATmega328)

EEPROM: 1 KB (ATmega328)

Clock Speed: 16 MHz

Applications of Arduino Technology:

- 1.The Obstacle Avoidance Robot Operated with Arduino.
- 2.Arduino based Controlling of Electrical Appliances using IR.
- 3.Arduino based Home Automation.
- 4.Underground Cable Fault Recognition using the Arduino Board.

Name	Arduino Uno	Raspberry Pi	BeagleBone
Model Tested	R3	Model B	Rev A5
Price	\$29.95	\$35	\$89
Size	2.95"x2.10"	3.37"x2.125"	3.4"x2.1"
Processor	ATMega 328	ARM11	ARM Cortex-A8
Clock Speed	16MHz	700MHz	700MHz
RAM	2KB	256MB	256MB
Flash	32KB	(SD Card)	4GB(microSD)
EEPROM	1KB		
Input Voltage	7-12v	5v	5v
Min Power	42mA (.3W)	700mA (3.5W)	170mA (.85W)
Digital GPIO	14	8	66
Analog Input	6 10-bit	N/A	7 12-bit
PWM	6		8
TWI/I2C	2	1	2
SPI	1	1	1
UART	1	1	5
Dev IDE	Arduino Tool	IDLE, Scratch, Squeak/Linux	Python, Scratch, Squeak, Cloud9/Linux
Ethernet	N/A	10/100	10/100
USB Master	N/A	2 USB 2.0	1 USB 2.0
Video Out	N/A	HDMI, Composite	N/A
Audio Output	N/A	HDMI, Analog	Analog

Conclusion:

Hence we studied the basic features and Pin configuration of Raspberry Pi, Beagle board, and Arduino.

FAQ's

1. Which processor is used in Raspberry Pi?
2. Which is the processor used in the Beagle board?
3. Which processor is used in Arduino?
4. How many GPIO pins are there in Raspberry Pi?
5. How many GPIO pins are there on the Beagle board?
6. How many GPIO pins are there in Arduino?
7. What are the features of Raspberry Pi?
8. What are the features of Beagle board?
9. What are the features of Arduino?
10. What is the clock speed of Raspberry Pi, Beagle board, and Arduino?

Assignment 2

Aim: Study of different operating systems for Raspberry Pi/Beagle board. Understanding the process of OS installation on Raspberry Pi/Beagle board

Objective:

- To understand The Functionalities of various single Boards embedded platforms fundamentals.
- To Develop a Comprehensive Approach towards Building small low cost embedded IOT systems.
- To implement the assignments based on sensory inputs.

Problem Statement: Study of different operating systems for Raspberry Pi/Beagle board. Understanding the process of OS installation on Raspberry Pi/Beagle board.

Outcomes:

- Design the minimum system for sensor-based applications.
- Solve the Problems related to primitive needs using IoT.
- Develop Full-fledged IoT application for Distributed environment.

Software and Hardware Requirements:

Raspberry-Pi /Beagle board circuit, basic peripherals like LEDS, buzzer.

Theory:

A. OS installation Steps on Raspberry Pi

1. Go to rasberry.org/download and download the Raspberry-Pi desktop pc ISO image file
2. Extract image file
3. Format 4GB SD card using an SD card Formatter
4. Using win32 Disk Manager Software write ISO image file to SD Card.
5. Put SD card on Raspberry Pi

B. OS installation Steps on Beagle board

1. Go to <http://beagleboard.org/latest-images/> and download the latest Debian image files
 2. Extract image file
 3. Format 4GB SD card using SD card Formatter
 4. Using win32 Disk Manager software write ISO image file to SD Card.
-

5. Put SD card on Beagle board

Conclusion: Thus we have studied different operating systems for Raspberry Pi/Beagle board and understand the process of OS installation on Raspberry Pi/Beagle board.

FAQ's

1. Which operating system is used in Raspberry Pi?
 2. Which operating system is used in the Beagle board?
 3. How you are going to install Raspberry Pi/Beagle board?
 4. What are the other operating systems that can be installed on Raspberry Pi/Beagle board?
 5. What is an SD card?
 6. What type of memory is used in SD cards?
 7. How to run Raspberry Pi in headless mode?
 8. List available models in Raspberry Pi
 9. List out Some popular companies that are working on IoT
 10. Define GPIO
-

Assignment 3

Aim: Study of Connectivity and configuration of Raspberry Pi/Beagle board/ Arduino circuit with basic peripherals, LEDS. It is understanding GPIO and its use in the program.

Objective:

- To understand The Functions of various single Board embedded platforms fundamentals.
- To Develop a Comprehensive Approach Towards Building small low-cost embedded IOT systems.
- To Implement the assignments based on sensory inputs.

Problem Statement: Study of Connectivity and configuration of Raspberry-Pi /Beagle board/ Arduino circuit with basic peripherals, LEDS. Understanding GPIO and its use in program.

Outcomes:

- Design the minimum system for sensor-based application.
- Solve the Problems related to the primitive needs using IoT.
- Develop Full Fledged IoT application for Distributed environment.

Software and Hardware Requirements:

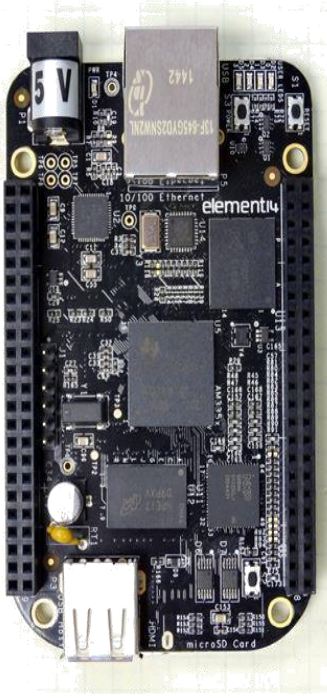
Raspberry-Pi /Beagle board/ Arduino circuit, basic peripherals like LEDS, buzzer.

Theory:

The Beagle bone has a large number of pins. There are two headers. Make sure you orient your Beagle bone in the same direction as mine in the picture, with the five volt plug on the top. In orientation, the pin header on the left is referred to as “P9” and the pin header on the right is referred to as “P8”. The legend in the diagram above shows the functions or the possible functions of the various pins. First, we have shaded in red the various 5V, 3.3V, 1.8V, and ground pins. Note that VDD_ADC is a 1.8 Volt supply and is used to provide a reference for Analog Read functions. The general-purpose GPIO pins have been shaded in green. Note some of these green pins can also be used for UART serial communication. If you want to simulate analog output, between 0 and 3.3 volts, you can use the PWM pins shaded in purple. The light blue pins can be used as analog in. Please note that the Analog In reads between 0 and 1.8 volts. You should not allow these pins to see higher voltages than 1.8 volts. When using these pins, use pins 32 and 34 as your voltage reference and ground, as pin 32 outputs a handy 1.8 volts. The pins shaded in light orange can be used for I2C. The dark orange pins are primarily used for LCD screen applications.

Beaglebone Black Pinout Diagram

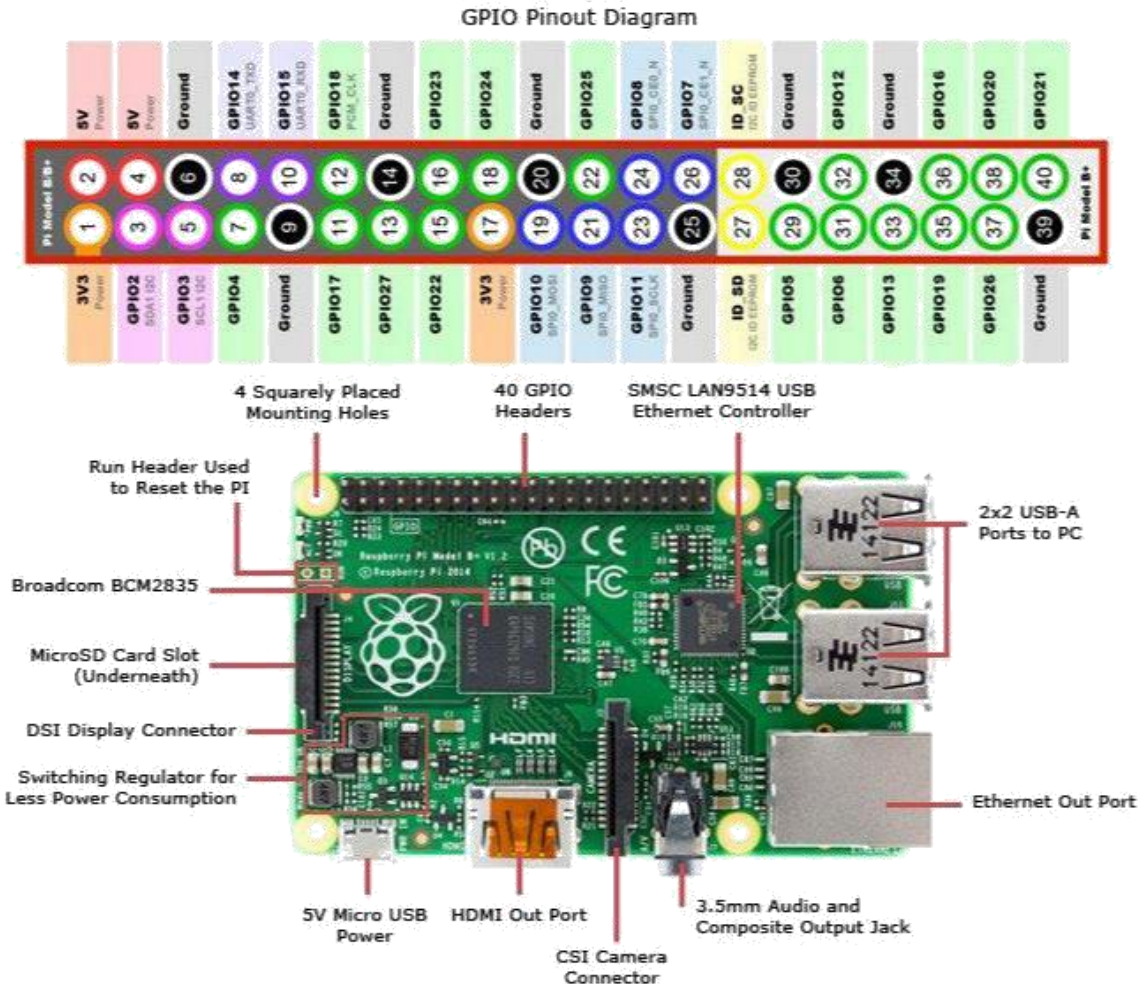
P9				P8			
Function	Physical Pins		Function	Function	Physical Pins		Function
DGND	1	2	DGND	DGND	1	2	DGND
VDD 3.3 V	3	4	VDD 3.3 V	MMC1_DAT6	3	4	MMC1_DAT7
VDD 5V	5	6	VDD 5V	MMC1_DAT2	5	6	MMC1_DAT3
SYS 5V	7	8	SYS 5V	GPIO_66	7	8	GPIO_67
PWR_BUT	9	10	SYS_RESET	GPIO_69	9	10	GPIO_68
UART4_RXD	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
UART4_TXD	13	14	EHRPWM1A	EHRPWM2B	13	14	GPIO_26
GPIO_48	15	16	EHRPWM1B	GPIO_47	15	16	GPIO_46
SPIO_CSO	17	18	SPIO_D1	GPIO_27	17	18	GPIO_65
I2C_SCL	19	20	I2C_SDA	EHRPWM2A	19	20	MMC1_CMD
SPIO_DO	21	22	SPIO_SLCK	MMC1_CLK	21	22	MMC1_DAT5
GPIO_49	23	24	UART1_TXD	MMC1_DAT4	23	24	MMC1_DAT1
GPIO_117	25	26	UART1_RXD	MMC1_DAT0	25	26	GPIO_61
GPIO_115	27	28	SP11_CSO	LCD_VSYNC	27	28	LCD_PCLK
SP11_DO	29	30	GPIO_112	LCD_HSYNC	29	30	LCD_AC_BIAS
SP11_SCLK	31	32	VDD_ADC	LCD_DATA14	31	32	LCD_DATA15
AIN4	33	34	GND_ADC	LCD_DATA13	33	34	LCD_DATA11
AIN6	35	36	AIN5	LCD_DATA12	35	36	LCD_DATA10
AIN2	37	38	AIN3	LCD_DATA8	37	38	LCD_DATA9
AIN0	39	40	AIN1	LCD_DATA6	39	40	LCD_DATA7
GPIO_20	41	42	ECAPWMO	LCD_DATA4	41	42	LCD_DATA5
DGND	43	44	DGND	LCD_DATA2	43	44	LCD_DATA3
DGND	45	46	DGND	LCD_DATA0	45	46	LCD_DATA1



LEGEND

- Power, Ground, Reset
- Digital Pins
- PWM Output
- 1.8 Volt Analog Inputs
- Shared I2C Bus
- Reconfigurable Digital

Raspberry Pi 3 Model B Pin Diagram



Basic Peripherals

1. LED

A light-emitting diode (LED) is a semiconductor device that emits light when an electric current is passed through it. Light is produced when the particles that carry the current (known as electrons and holes) combine together within the semiconductor material.

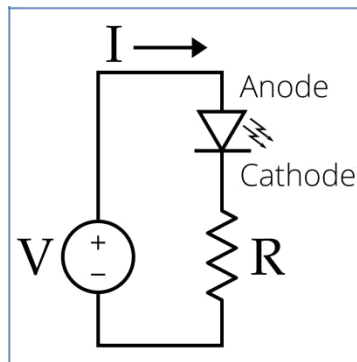
Since light is generated within the solid semiconductor material, LEDs are described as solid-state devices. The term solid-state lighting, which also encompasses organic LEDs (OLEDs), distinguishes this lighting technology from other sources that use heated filaments (incandescent and tungsten halogen lamps) or gas discharge (fluorescent lamps).

Program to Glow LED

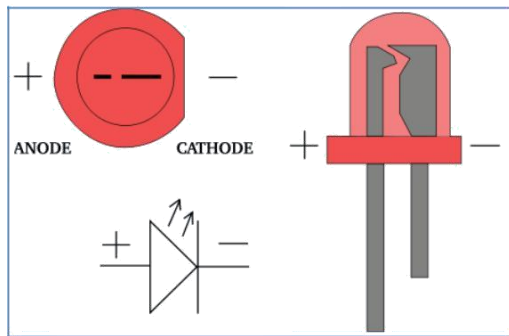
LED has two pins one is positive (long end) and one is negative (small end)

Connect the positive end to GPIO pin p8 10 of Beagle Bone or pin 17 of Raspberry Pi or pin 2 of Arduino using a jumper cable

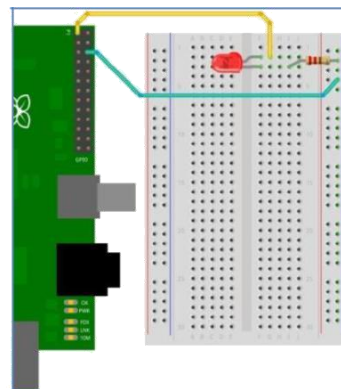
Connect the negative end to any GND pin on Beagle Bone or Raspberry Pi pin 2 of Arduino as shown in the diagram. Once the connection is done run code



LED Circuit Diagram



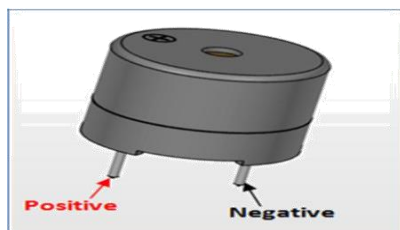
LED Symbol



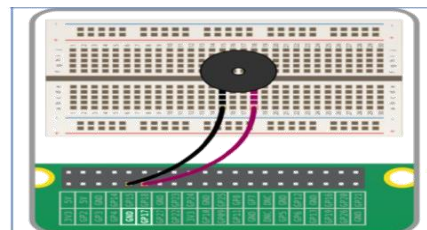
LED Connection with Raspberry pi

2. Buzzer

A buzzer is an electrical device is used to make a buzzing sound.



Buzzer



Buzzer Connected to Raspberry pi

Program to ON/OFF Buzzer

The buzzer has two pins one is positive (long end) and one is negative (small end) Connect the positive end to GPIO pin of Raspberry Pi/ Arduino using a jumper cable

Connect the negative end to any GND pin of Raspberry Pi/ Arduino as shown in the diagram. Once the connection is done run code.

Conclusion: Thus we have studied the connectivity and configuration of Raspberry Pi Raspberry Pi/Beagle board/ Arduino with basic peripherals, LEDS, and buzzer

FAQ'S

1. How many power pins and ground pins are there in Raspberry Pi
 2. List mostly used sensor types in IOT
 3. List mostly used actuator types in IOT
 4. What are the most common IOT applications?
 5. What are the functions used to read analog and digital data from a sensor in Arduino?
-

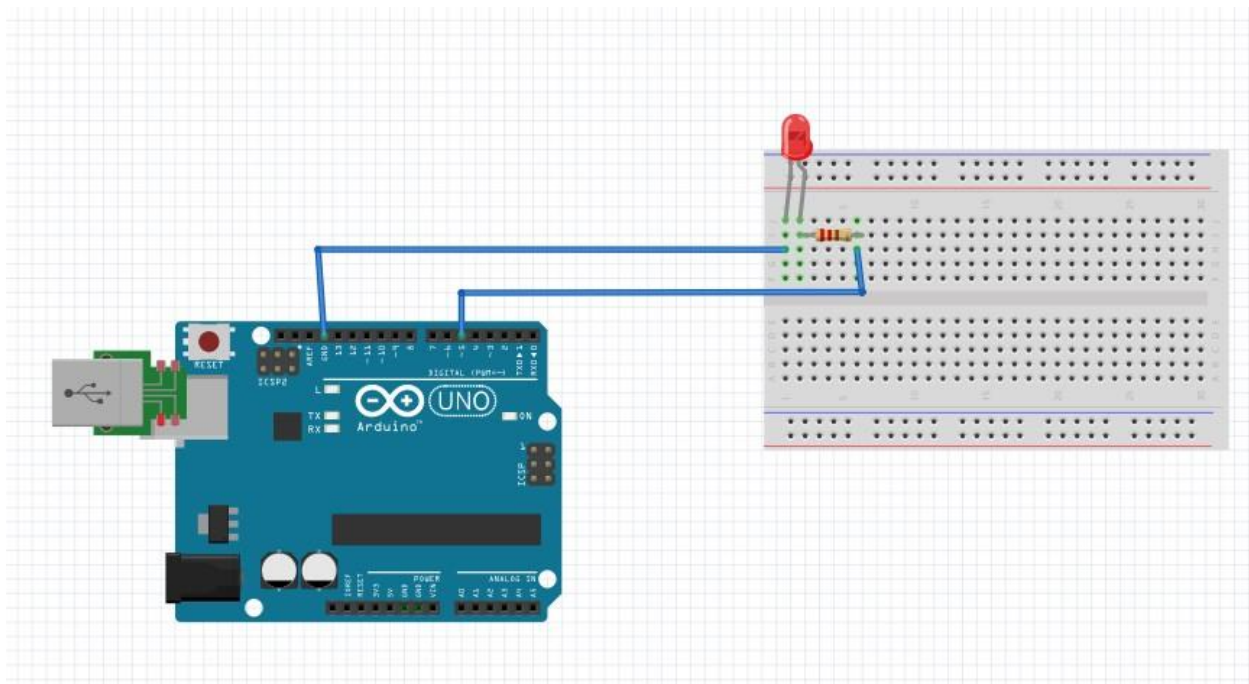
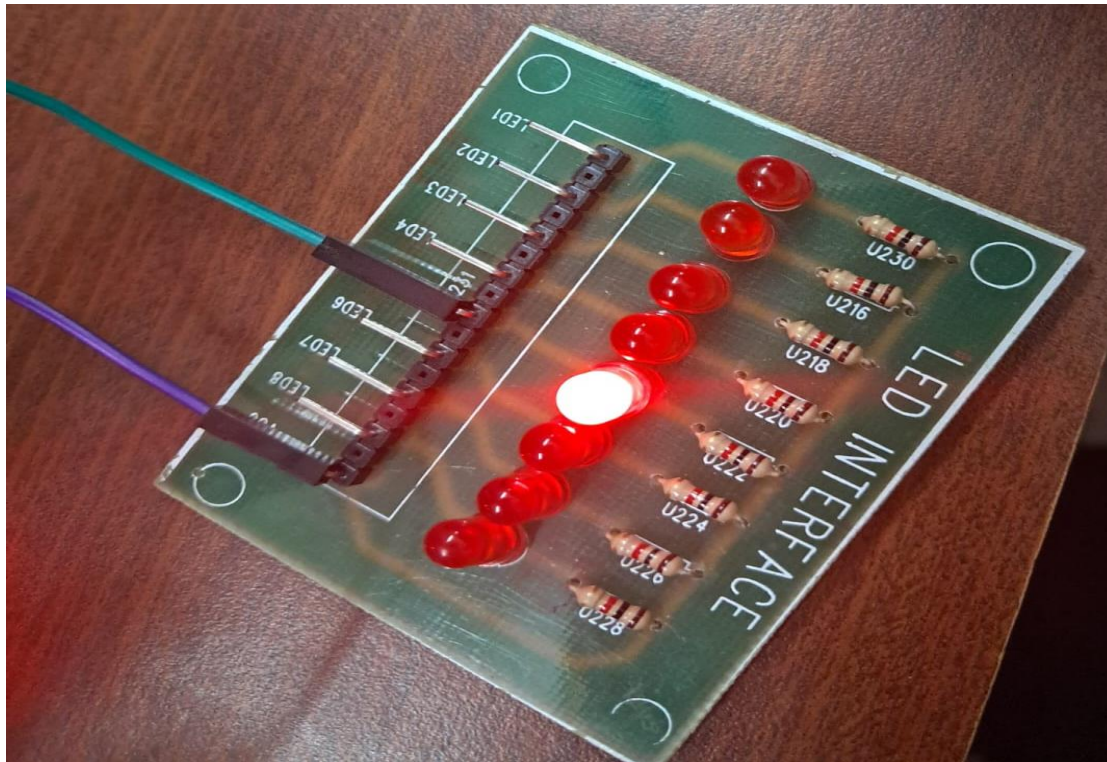
INPUT:-

```
#define led_pin 12

void setup() {
    // put your setup code here, to run once:
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    // put your main code here, to run repeatedly:
    digitalWrite(led_pin, HIGH);
    delay(1000);
    digitalWrite(led_pin, LOW);
    delay(500);
}
```

Output-



Assignment 4

Aim: Understanding the connectivity of Raspberry Pi/Beagle board/ Arduino circuit with IR sensor. Write an application to detect obstacles and notify users using LEDs.

Objective:

- To understand The Functions of various single Board embedded platforms fundamentals.
- To Develop a Comprehensive Approach Towards Building small low-cost embedded IOT systems.
- To Implement the assignments based on sensory inputs.

Problem Statement: Understanding the connectivity of Raspberry Pi/Beagle board circuit with IR sensor. Write an application to detect obstacles and notify users using LEDs.

Outcomes:

- Design the minimum system for sensor-based applications.
- Solve the Problems related to primitive needs using IoT.
- Develop Full-fledged IoT application for Distributed environment.

Software and Hardware Requirements:

Raspberry-Pi /Beagle board/Arduino circuit, basic peripherals like LEDS, buzzer.

Theory:

IR Sensor

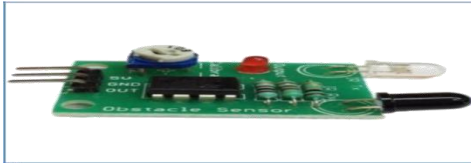
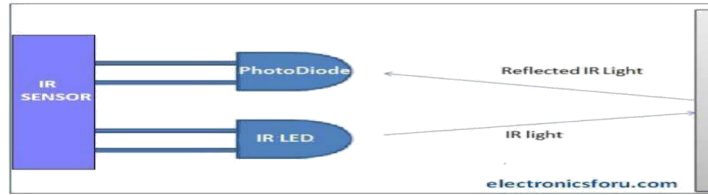
An IR sensor is a device that detects IR radiation falling on it. Proximity sensors (used in touchscreen phones and edge-avoiding robots), contrast sensors (used in line following robots), and obstruction counters/sensors (used for counting goods and in burglar alarms) are some applications involving IR sensors.

Principle of Working

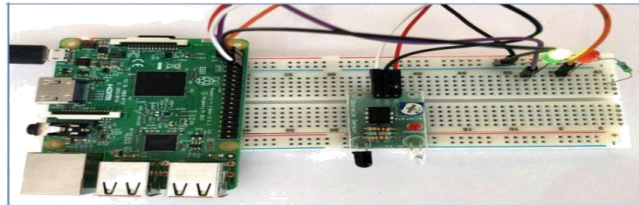
An IR sensor consists of two parts, the emitter circuit and the receiver circuit. This is collectively known as a photo-coupler or an optocoupler.

The emitter is an IR LED and the detector is an IR photodiode. The IR photodiode is sensitive to the IR light emitted by an IR LED. The photo-diode's resistance and output voltage change in proportion to the IR light received. This is the underlying working principle of the IR sensor.

The type of incidence can be direct incidence or indirect incidence. In direct incidence, the IR LED is placed in front of a photodiode with no obstacle in between. In indirect incidence, both the diodes are placed side by side with an opaque object in front of the sensor. The light from the IR LED hits the opaque surface and reflects back to the photodiode.



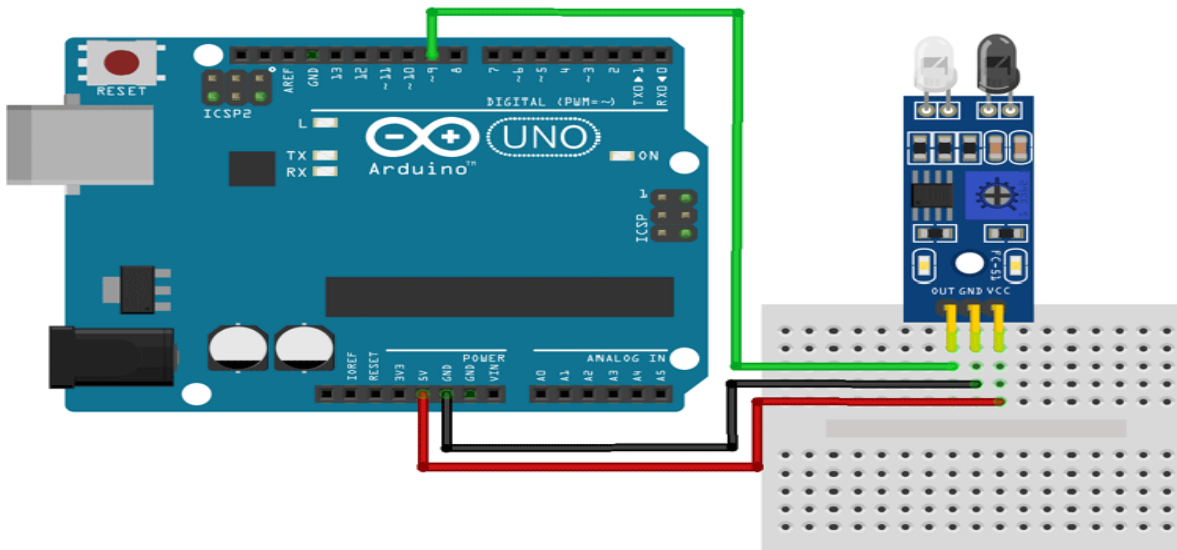
IR Sensor



IR connected to Raspberry pi

Connection

1. Connect IR and LED to the Arduino pin diagram using a jumper cable.
2. Write code to detect objects nearer to the IR sensor and notify using LED
3. Run code



Conclusion: Thus we have studied the connectivity of the Raspberry Pi/Beagle board/ Arduino circuit with an IR sensor and an application to detect obstacles and notify users using LEDs.

FAQ'S

1. How raspberry pi different from a desktop computer?
2. What is the use of GPIO PINS?
3. What are the common commands used in the raspberry pi?
4. How obstacles are detected using IR sensor?

Input-

```
const int irPin = 10; // Pin connected to the IR sensor output
```

```
const int ledPin = 9; // Pin connected to the LED
```

```
void setup() {
```

```
  // Initialize the serial monitor
```

```
  Serial.begin(9600);
```

```
  // Set the LED pin as OUTPUT
```

```
  pinMode(ledPin, OUTPUT);
```

```
  // Set the IR pin as INPUT
```

```
  pinMode(irPin, INPUT);
```

```
}
```

```
void loop() {
```

```
  // Read the value from the IR sensor
```

```
  int irValue = digitalRead(irPin);
```

```
  // Print the value to the serial monitor
```

```
  Serial.println(irValue);
```

```
  // If the IR sensor detects a signal (HIGH), turn on the LED
```

```
  if (irValue == HIGH) {
```

```
    digitalWrite(ledPin, HIGH); // Turn LED on
```

```
  } else {
```

```
    digitalWrite(ledPin, LOW); // Turn LED off
```

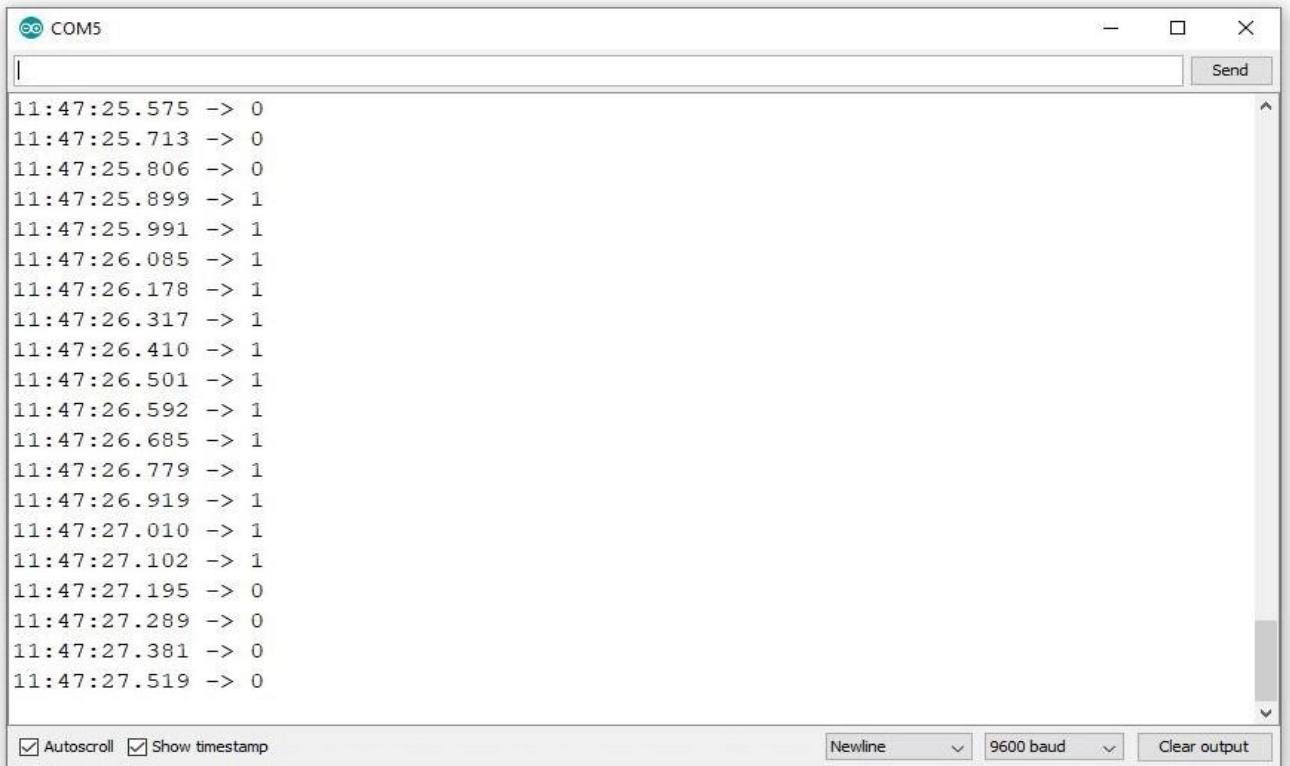
```
  }
```

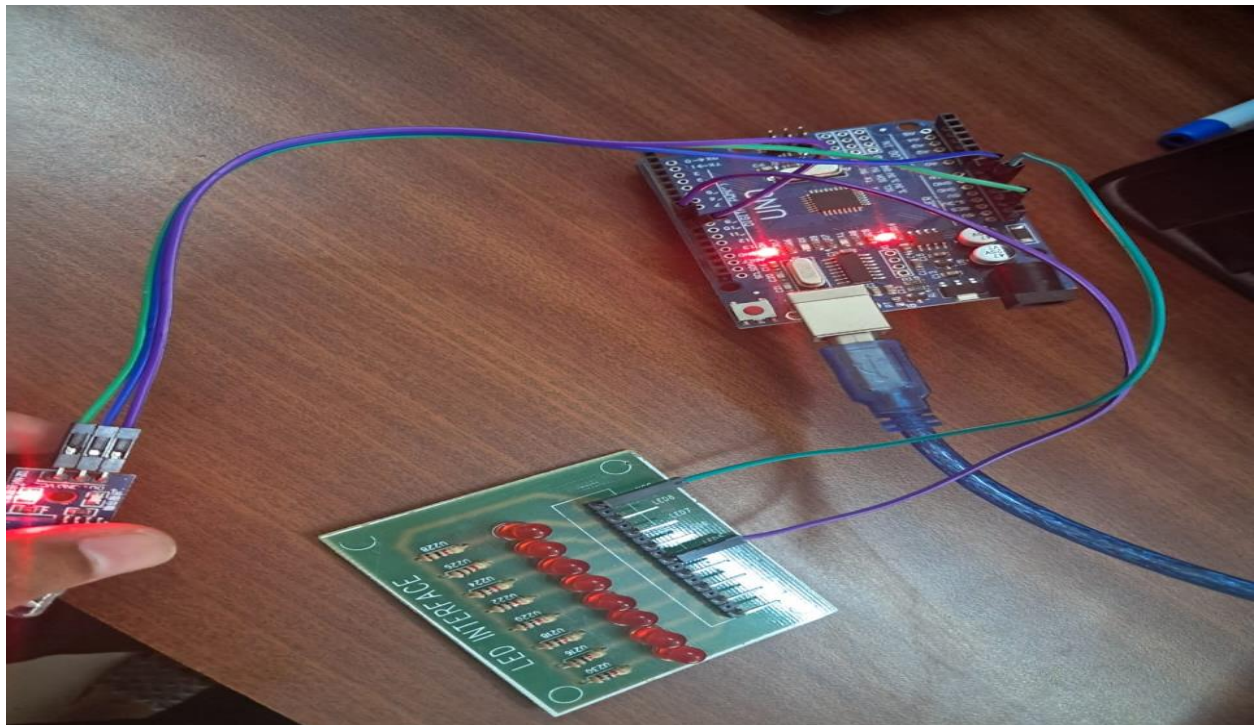
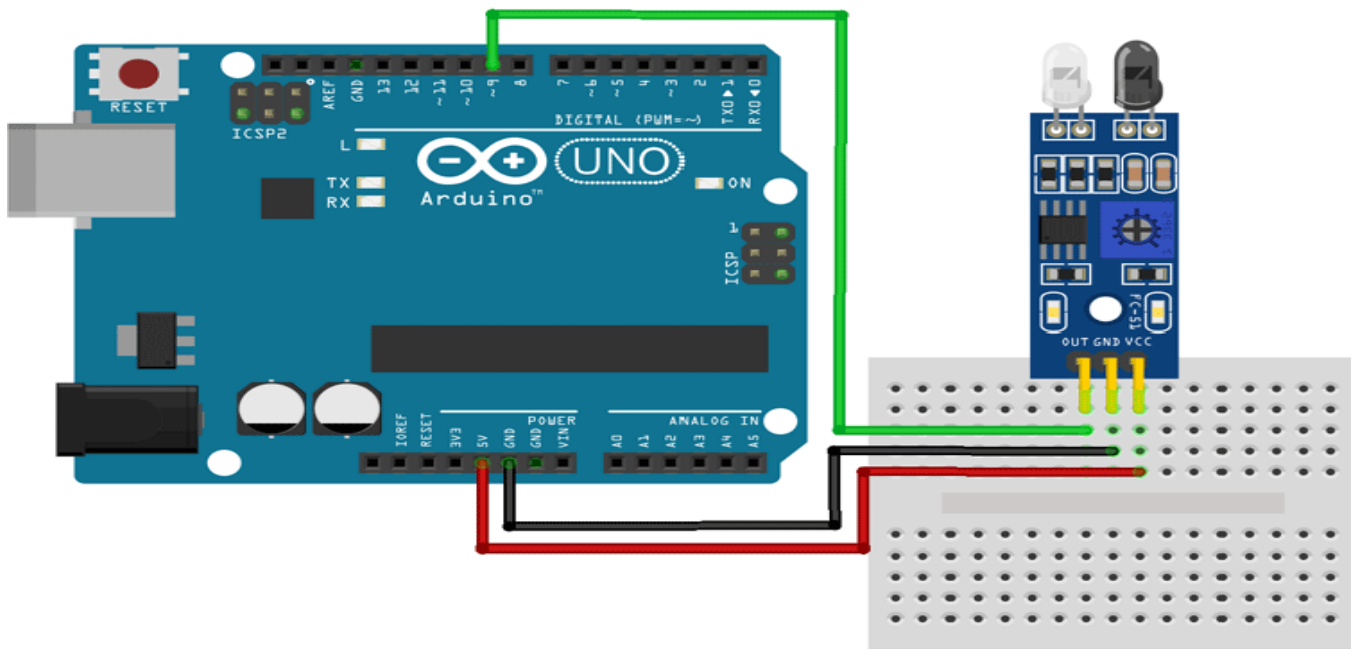
```
  // Small delay for stability
```

```
  delay(100);
```

```
}
```

Output-





Assignment 5

Title: Connectivity of Raspberry-Pi /Beagle/ Arduino board circuit with DHT11 temperature sensor

Problem Statement: Understanding the connectivity of Raspberry Pi/Beagle board/Arduino circuit with temperature sensor. Write an application to read the environment temperature. If the temperature crosses a threshold value, the application indicates user using LEDs/ Buzzer.

Objectives: To understand the connectivity of Raspberry-Pi /Beagle board/ Arduino circuit with DHT11 temperature sensor

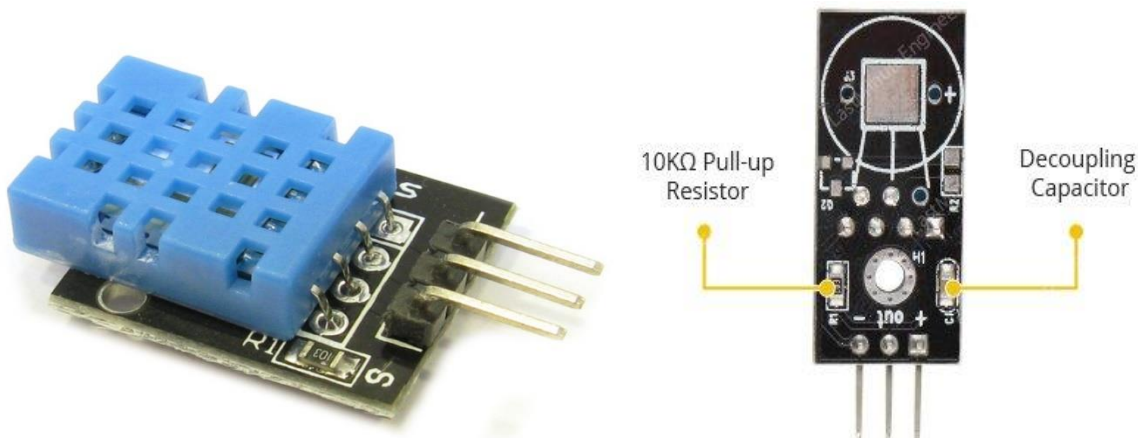
Theory:

Arduino UNO

Arduino UNO is a microcontroller board based on the **ATmega328P**. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

DHT11

DHT11 can measure temperature from 0°C to 50°C with a $\pm 2.0^\circ\text{C}$ accuracy, and humidity from 20 to 80% with a 5% accuracy. DHT11 has a sampling rate of 1Hz, which means it can provide new data once every second.

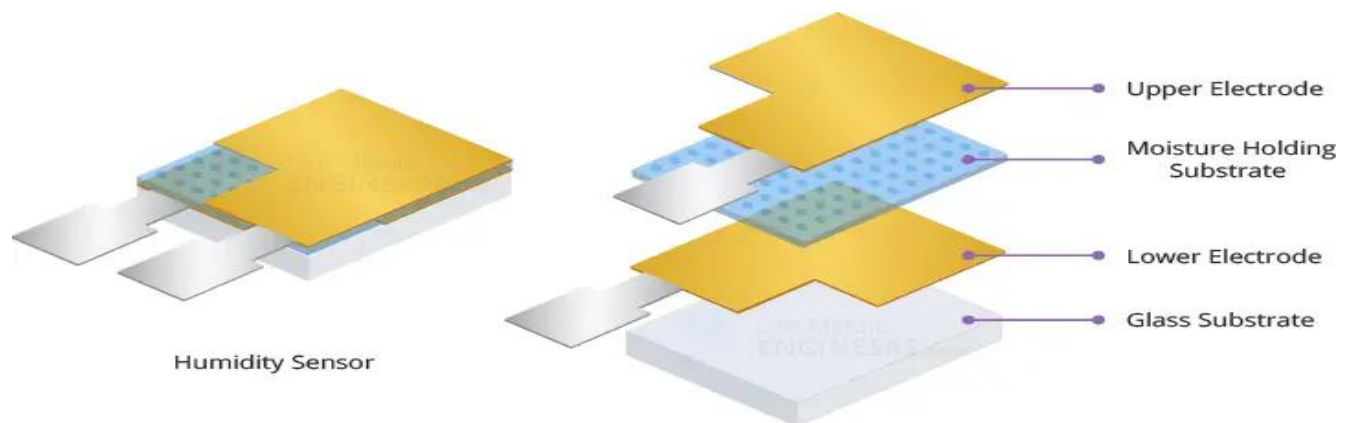


DHT11 sensors typically require an external 10K pull-up resistor on the output pin for proper communication between the sensor and the Arduino. However, because the module already includes a pull-up resistor, you do not need to add one. The module also includes a decoupling capacitor for filtering power supply noise. If you remove the sensor's casing, you will find an NTC thermistor and a humidity-sensing component inside. The humidity sensing component has

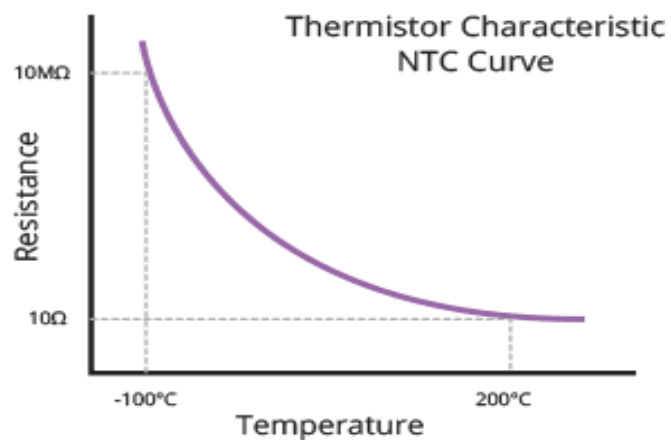
two electrodes with a moisture-holding substrate (usually a salt or conductive plastic polymer) in between. As the humidity rises, the substrate absorbs water vapor, resulting in the release of ions and a decrease in the resistance between the two electrodes.

This change in resistance is proportional to the humidity, which can be measured to estimate relative humidity.

DHT11 also includes a NTC thermistor for measuring temperature. A thermistor is a type of resistor whose resistance varies with temperature.

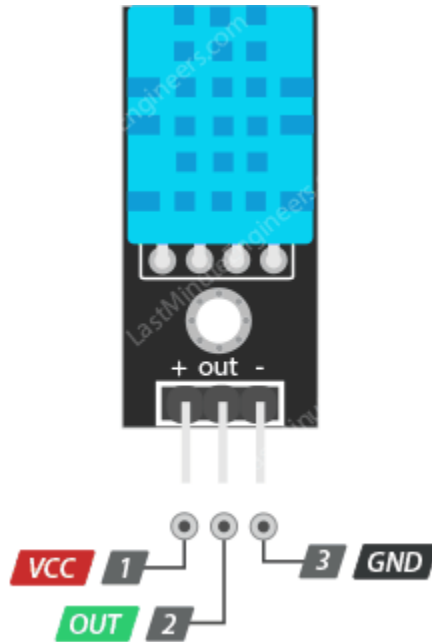


Technically, all resistors are thermistors in the sense that their resistance changes slightly with temperature, but this change is typically very small and difficult to measure. Thermistors are designed so that their resistance changes dramatically with temperature (by 100 ohms or more per degree). The term “NTC” stands for “Negative Temperature Coefficient,” which means that resistance decreases as temperature rises.



The sensor also includes an 8-bit SOIC-14 packaged IC. This IC measures and processes the analog signal using stored calibration coefficients, converts the analog signal to digital, and outputs a digital signal containing the temperature and humidity.

The DHT11 module is relatively simple to connect. There are only three pins:



+ (VCC) pin provides power to the sensor. Even though the supply voltage of the module ranges from 3.3V to 5.5V, a 5V supply is recommended. With a 5V power supply, the sensor can be placed up to 20 meters away. With 3.3V supply voltage, the sensor can be placed just 1 meter away; otherwise, the line voltage drop will cause measurement errors.

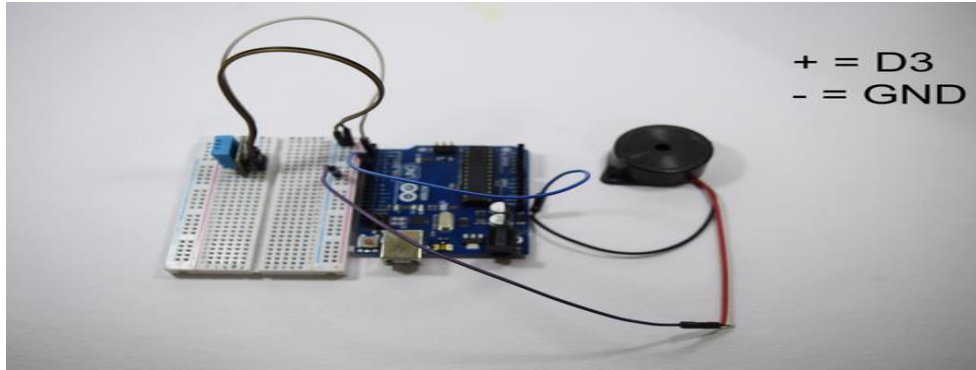
Out pin is used for communication between the sensor and the microcontroller.

Connection DHT11 Module to Arduino

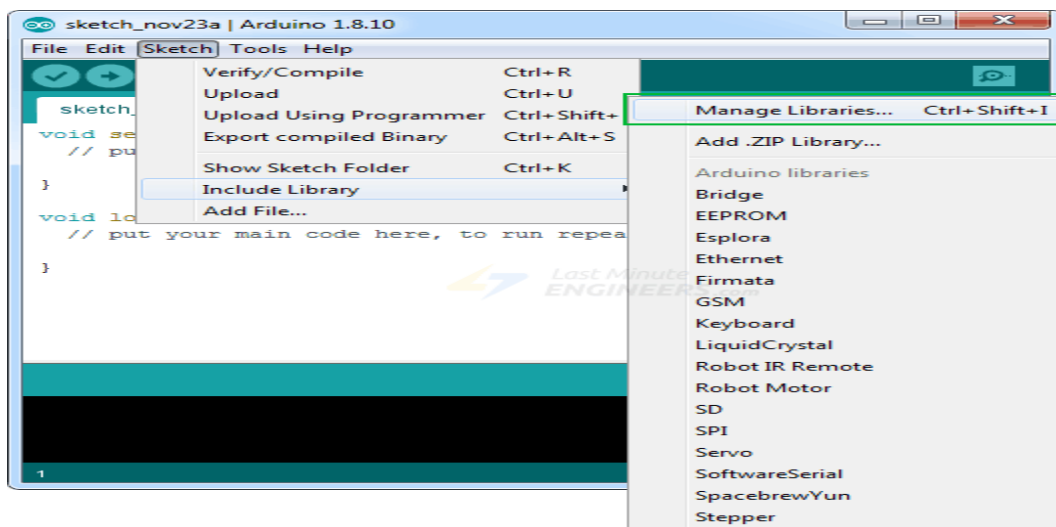
Connections are relatively simple. Begin by connecting the + (VCC) pin to the Arduino's 5V output and the – (GND) pin to ground. Finally, connect the Out pin to digital pin #8.

Installing DHT library

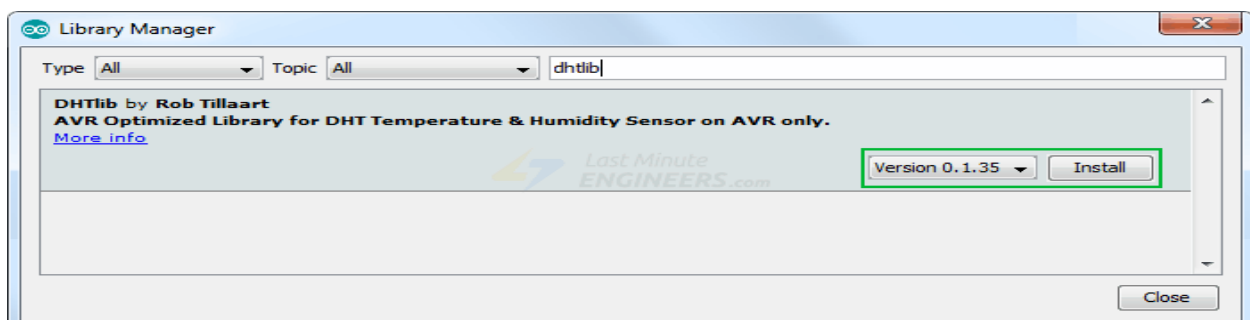
The DHT sensors has their own proprietary single-wire data transfer protocol. This protocol requires precise timing. We don't have to worry too much about this, though, because we'll be using the [DHTlib library](#), which handles almost everything.



To install the library, navigate to Sketch > Include Library > Manage Libraries... Wait for the Library Manager to download the libraries index and update the list of installed libraries.



Filter your search by entering 'dhtlib'. There should only be a single entry. Click on that and then choose Install.



After uploading the sketch on Arduino output is seen on the serial monitor and the serial plotter.

Conclusion: Thus we have studied the connectivity of the Arduino circuit with a DHT11 temperature sensor and an application to detect the temperature crossing the threshold is notify users using a buzzer.

FAQ'S

1. What is the max and min temperature range of DHT11?
 2. What is the max and min Humidity range of DHT11?
 3. What is the sampling rate of DHT11?
 4. How to include DHT libraries in Arduino?
-

Input-

```
#include <dht.h>
#define inPin 2 // Pin number for DHT11 data
#define buzzerPin 10 // Pin number for the buzzer
Dht DHT;
Const float tempThreshold = 37.0; // Temperature threshold in Celsius
Const float humidityThreshold = 70.0; // Humidity threshold in percentage
Void setup() {
  Serial.begin(9600);
  pinMode(buzzerPin, OUTPUT); // Set buzzer pin as output
  digitalWrite(buzzerPin, LOW); // Ensure the buzzer is off initially
}
Void loop() {
  Int readData = DHT.read11(inPin);
  Float t = DHT.temperature; // Read temperature
  Float h = DHT.humidity; // Read humidity
  Serial.print("Temperature = ");
  Serial.print(t);

  Serial.print("Â°C | ");
  Serial.print((t * 9.0) / 5.0 + 32.0); // Convert Celsius to Fahrenheit
  Serial.println("Â°F ");

  Serial.print("Humidity = ");
  Serial.print(h);
  Serial.println("% ");

                                     // Send data for plotting

  Serial.print("T:");
  Serial.print(t);
  Serial.print(",H:");
```

```
Serial.println(h);  
// Check if temperature or humidity exceeds thresholds  
If (t > tempThreshold || h > humidityThreshold) {  
  Tone(buzzerPin, 1000); // Activate buzzer at 1000 Hz  
  Delay(200); // Sound for 200 milliseconds  
  noTone(buzzerPin); // Turn off buzzer  
  Serial.println("Alert! Threshold exceeded!");  
}  
Delay(1000); // Wait one second before next reading  
}
```

Output-

