**SWAMI KESHVANAND INSTITUTE OFTECHNOLOGY, MANAGEMENT AND GRAMOTHAN, JAIPUR**

****

**Hands on Lab Guide**

**(Lab Manual)**

**INTERNET OF THINGS LAB**

**IV Year B.Tech VII SEM**

**(Course Code: 7CS4-21)**

**Session 2022-2023**

**Department of Computer Science &Engineering**

**SKIT, JAIPUR**

**Swami Keshvanand Institute of Technology,Management &Gramothan, Ramnagaria, Jagatpura, Jaipur-302017**

**LAB MANUAL**

**Internet of Things Lab (7CS4-21)**

**VERSION 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **AUTHOR / OWNER** | **REVIEWED BY** | **APPROVED BY** |
| **NAME** | Dr. Pankaj Dadheech,  Dr. Sarabjeet Singh, Dr. Yogendra Gupta,  Dr. Nilam Choudhary | Dr. Vinay Kanungo | Prof. (Dr.) Mukesh Kumar Gupta |
| **DESIGNATION** | Associate Professor/ Assistant Professor | Associate Professor | HOD (CSE) |
| **SIGNATURE & REVIEW DATE** |  |  |  |
| **SIGNATURE & REVIEW DATE** |  |  |
| **SIGNATURE & REVIEW DATE** |  |  |

|  |
| --- |
| Department of Computer Science & Engineering,  Jaipur – 302017, Rajasthan (INDIA)  E-Mail: hodcs@skit.ac.in, URL: [www.skit.ac.in](http://www.skit.ac.in) |

**GENERAL LABORATORY RULES**

**Responsibilities of Students**

Students are expected to follow some obvious rules of conduct:

**DO’s:**

* + Enter the lab on time and leave at proper time.
  + Wait for the previous class to leave before the next class enters.
  + Keep the bag outside in the respective racks.
  + Turn off the machine before leaving the lab.
  + Leave the labs at least as nice as you found them.
  + If you notice a problem with a piece of equipment (e.g. a computer doesn't respond) or the room in general (e.g. cooling, heating, lighting) please report it to lab assistant immediately.
  + Please be considerate of those around you, especially in terms of noise level. While labs are a natural place for conversations of all types, kindly keep the volume turned down.

**DONT’S:**

* + Do not misuse the equipment.
  + Do not adjust the heat or air conditioners. If you feel the temperature is not properly set, inform lab staff; we will attempt to maintain a balance that is healthy for people and machines.
  + Do not attempt to reboot the computer. Report the problems to lab assistant.
  + Do not remove or modify any software or file without permission.
  + Do not remove printers and machines from the network without the permission of lab assistant.
  + Do not monopolize equipment. If you're going to be away from your machine for more than 10 or 15 minutes, log out before leaving. This is both for the security of your account, and to ensure that others are able to use the lab resources while you are not.
  + Do not download or upload of MP3, JPG or MPEG files.
  + Playing games is not allowed in the lab.
  + No hardware including USB drives can be connected or disconnected in the labs without prior permission of the lab assistant.
  + Eatables are not allowed in the lab.
  + Don’t bring the mobile phones in the lab. If necessary then keep them in silence mode.

**INSTRUCTIONS**

**BEFORE ENTERING IN THE LAB**

* All the students are supposed to prepare the theory regarding the next experiment/ Program.
* Students are supposed to bring their lab records as per their lab schedule.
* Previous experiment/program should be written in the lab record.
* If applicable trace paper/graph paper must be pasted in lab record with proper labeling.
* All the students must follow the instructions, failing which he/she may not be allowed in the lab.

**WHILE WORKING IN THE LAB**

* Adhere to experimental schedule as instructed by the faculty.
* Get the previously performed experiment/ program signed by the faculty.
* Get the output of current experiment/program checked by the faculty in the lab copy.
* Each student should work on his/her assigned computer at each turn of the lab.
* Take responsibility of valuable accessories.
* If anyone is caught red-handed carrying any equipment of the lab, then he/she will have to face serious consequences.

**Marking/Assessment System**

**Total Marks -100**

Distribution of Marks - 60 (Sessional)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attendance | File Work | Performance | Viva | Total |
| 10 | 10 | 30 | 10 | 60 |

Distribution of Marks - 40 (End Term) Depends on Examiner

|  |  |  |  |
| --- | --- | --- | --- |
| File Work | Performance | Viva | Total |
| 10 | 20 | 10 | 40 |

RAJASTHAN TECHNICAL UNIVERSITY, KOTA

**Syllabus**

**IV Year-VII Semester: B.Tech. Computer Science and Engineering**

|  |  |  |  |
| --- | --- | --- | --- |
| **7CS4-21: Internet of Things Lab** | | | |
| **Credit: 2** | | **Max. Marks: 100(IA:60, ETE:40)** | |
| **0L+0T+4P** | | **End Term Exam: 2 Hours** | |
| **SN** | **List of Experiments** | |
| **1** | Start Raspberry Pi and try various Linix commands in command terminal window: ls, cd, touch, mv, rm, man, mkdir, rmdir, tar, gzip, cat, more, less, ps, sudo, cron, chown,chgrp, ping etc. | |
| **2** | Run some python programs on Pi like:   1. Read your name and print Hello message with name 2. Read two numbers and print their sum, difference, product and division. 3. Word and character count of a given string. 4. Area of a given shape (rectangle, triangle and circle) reading shape and appropriate values from standard input. | |
| **3** | Run some python programs on Pi like:   1. Print a name 'n' times, where name and n are read from standard input, using for and while loops. 2. Handle Divided by Zero Exception. 3. Print current time for 10 times with an interval of 10 seconds. 4. Read a file line by line and print the word count of each line. | |
| **4** | 1. Light an LED through Python program 2. Get input from two switches and switch on corresponding LEDs 3. Flash an LED at a given on time and off time cycle, where the two times are taken from a file. | |
| **5** | 1. Flash an LED based on cron output (acts as an alarm) 2. Switch on a relay at a given time using cron, where the relay's contact terminals are connected to a load. 3. Get the status of a bulb at a remote place (on the LAN) through web. | |
|  | The student should have hands on experience in using various sensors like temperature, humidity, smoke, light, etc. and should be able to use control web camera, network, and relays connected to the Pi. | |

**7CS4-21. INTERNET OF THINGS LAB**

**List of Experiments**

1. Start Raspberry Pi and try various Linux commands in command terminal window:  
   ls, cd, touch, mv, rm, man, mkdir, rmdir, tar, gzip, cat, more, less, ps, sudo, cron, chown,chgrp, ping etc.
2. Write Run some python programs on Pi like:
3. Read your name and print Hello message with name
4. Read two numbers and print their sum, difference, product and division.
5. Word and character count of a given string.
6. Area of a given shape (rectangle, triangle and circle) reading shape and appropriate values from standard input.
7. Run some python programs on Pi like:
8. Print a name 'n' times, where name and n are read from standard input, using for and while loops.
9. Handle Divided by Zero Exception.
10. Print current time for 10 times with an interval of 10 seconds.
11. Read a file line by line and print the word count of each line.
12. LED blinking using Raspberry pi.
13. LED bilking with push button using Raspberry pi.
14. Flash an LED at a given on time and off time cycle, where the two times are taken from a file.
15. Flash an LED based on cron output (acts as an Alarm).
16. Switch on a relay at a given time using cron, where the relay's contact terminals are

connected to a load.

1. Get the status of bulb at a remote place on the LAN through Web.
2. Interface Temperature & Humidity sensor using Raspberry pi and send the data over ThingSpeak cloud Infrastructure.
3. Design a system using Arduino to measure distance using Ultrasonic Sensor.
4. Design a system using Arduino to monitor Temperature and Humidity.

**Beyond Syllabus:**

1.Design a system using Arduino to monitor Soil Moisture of a plant on an IDE. If soil moisture is less, then LED should glow.

2.Design a system using Arduino that blinks an LED when you detect a motion in an area on an IDE.

3.Motion detection and blinking the light when the light intensity is low using Arduino.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Topic** | **Page Number** |
| 1 | Lab Plan | 11 |
| 2 | Lab Objective and Outcome | 12 |
| 3 | Experiment No 1 | 20 |
| 4 | Experiment No 2 | 26 |
| 5 | Experiment No 3 | 30 |
| 6 | Experiment No 4 | 33 |
| 7 | Experiment No 5 | 35 |
| 8 | Experiment No 6 | 37 |
| 9 | Experiment No 7 | 38 |
| 10 | Experiment No 8 | 42 |
| 11 | Experiment No 9 | 44 |
| 12 | Experiment No 10 | 47 |
| 13 | Experiment No 11 | 52 |
| 14 | Experiment No 12 | 55 |
| 15 | Beyond Syllabus | 58 |
| 20 | Viva Questions | 64 |
| 21 | References | 80 |

**LAB PLAN**

Total number of experiment 15

Total number of turns required 15

**Number of turns required for**

|  |  |  |
| --- | --- | --- |
| **Experiment Number** | **Turns** | **Scheduled Day** |
| Exp. 1 | 1 | Day 1 |
| Exp. 2 | 1 | Day 2 |
| Exp. 3 | 1 | Day 3 |
| Exp. 4 | 1 | Day 4 |
| Exp. 5 | 1 | Day 5 |
| Exp. 6 | 1 | Day 6 |
| Exp. 7 | 1 | Day 7 |
| Exp. 8 | 1 | Day 8 |
| Exp. 9 | 1 | Day 9 |
| Exp. 10 | 1 | Day 10 |
| Exp. 11 | 1 | Day 11 |
| Exp. 12 | 1 | Day 12 |
| Exp.13 | 1 | Day 13 |
| Exp. 14 | 1 | Day 14 |
| Exp. 15 | 1 | Day 15 |

**Distribution of Lab Hours:**

Attendance 05 minutes

Explanation of features of language 15 minutes

Explanation of experiment 15 minutes

Performance of experiment 70 minutes

Viva / Quiz / Queries 15 minutes

**Total 120 Minutes (2 Hrs.)**

**Lab Objectives and Outcome**

**Objectives**

The objective of this course is to impart necessary and practical knowledge of components of Internet of Things and develop skills required to build real-life IoT based projects.

* To introduce the terminology, technology and its applications.
* To introduce the concept of M2M (machine to machine) with necessary protocols.
* To introduce the Python Scripting Language which is used in many IoT devices.
* To introduce the Raspberry PI platform, that is widely used in IoT applications.
* To introduce the implementation of web based services on IoT devices.

**Course Outcomes**

After completion of this course, students will be able to –

|  |  |
| --- | --- |
| 7CS4-01.1 | **Understand** the basics of application areas of IOT |
| 7CS4-01.2 | **Explain** and **realize** the revolution of Internet in Mobile Devices, Cloud & Sensor Networks |
| 7CS4-01.3 | **Discuss** the **architectur**e, operation, and business benefits of an IoT solution |
| 7CS4-01.4 | **Examine** the potential business opportunities that IoT can uncover |
| 7CS4-01.5 | **Explore** the relationship between IoT, cloud computing, and big data and Identify how IoT differs from traditional data collection systems |

**Arduino**

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programed (referred to as a microcontroller) and ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are:

* Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
* You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
* Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board.
* Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
* Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

## 

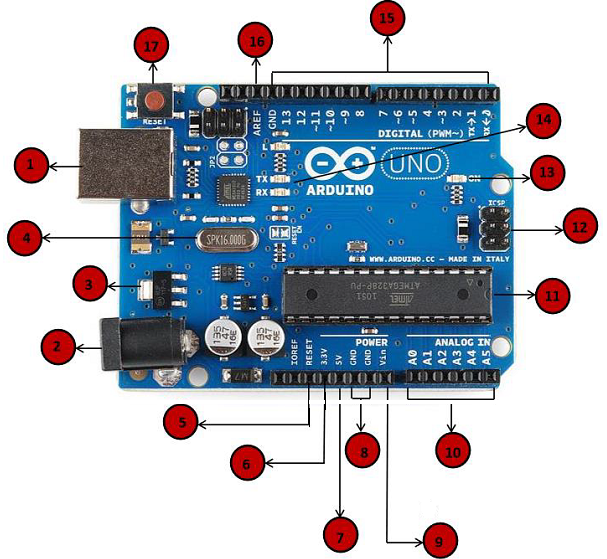
## Board Types:

Various kinds of Arduino boards are available depending on different microcontrollers used. However, all Arduino boards have one thing in common: they are programmed through the Arduino IDE. The differences are based on the number of inputs and outputs (the number of sensors, LEDs, and buttons you can use on a single board), speed, operating voltage, form factor etc. Some boards are designed to be embedded and have no programming interface (hardware), which you would need to buy separately. Some can run directly from a 3.7V battery, others need at least 5V. Here is a list of different Arduino boards available.

* **Arduino boards based on ATMEGA328 microcontroller**
* **Arduino boards based on ATMEGA32u4 microcontroller**
* **Arduino boards based on ATMEGA2560 microcontroller**
* **Arduino boards based on AT91SAM3X8E microcontroller**

**Components of Arduino Board**

The Arduino UNO board is used because it is the most popular board in the Arduino board family. In addition, it is the best board to get started with electronics and coding. Some boards look a bit different from the one given below, but most Arduinos have majority of these components in common.



**Fig: Arduino Uno Pin Diagram**

1. **Power USB:** Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).
2. **Power (Barrel Jack):** Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).
3. **Voltage Regulator:** The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.
4. **Crystal Oscillator:** The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.
5. **Arduino Reset:** Reset the Arduino board, i.e., start the program from the beginning. The UNO board can be reset in two ways. First, by using the reset button (17) on the board. Second, connect an external reset button to the Arduino pin labelled RESET (5).
6. **Pins (3.3, 5, GND, Vin):** 3.3V (6) − Supply 3.3 output volt, 5V (7) − Supply 5 output volt. Most of the componentsused with Arduino board works fine with 3.3 volt and 5 volt. GND (8)(Ground) − There are several GND pins on the Arduino, any of which can be used to ground your circuit.Vin (9) − This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.
7. **Analog pins:** The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.
8. **Main microcontroller:** Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.
9. **ICSP pin:** Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output.
10. **Power LED indicator:** This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.
11. **TX and RX LEDs:** On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.
12. **Digital I/O:** The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.
13. **AREF:** AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

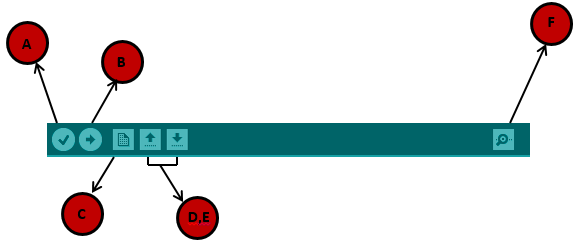
**Arduino IDE**

Arduino IDE is open source software that is used to program the Arduino controller board connected via USB cable. Following steps are required to set up an Arduino board.

1. **Power the board** by connecting it to a PC via USB cable. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port. Connect the Arduino board to your computer using the USB cable.
2. **Launch Arduino IDE:** After the Arduino IDE software is downloaded, unzip the folder. Inside the folder, find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.
3. **Open the first project**: Once the software starts, two options are available −

* Create a new project.
* Open an existing project example.
* To create a new project, select File → New.
* To open an existing project example, select File → Example

1. **Select the Arduino board:** To avoid any error while uploading your program to the board, select the correct Arduino board name, which matches with the board connected to your computer. Go to Tools → Board and select the Arduino Uno board.
2. **Select the serial port:** Select the serial device of the Arduino board. Go to Tools → Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.
3. **Upload the program to the board:** Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.



**Fig: IDE Toolbar**

A − Used to check if there is any compilation error.

B − Used to upload a program to the Arduino board.

C − Shortcut used to create a new sketch.

D − Used to directly open one of the example sketch.

E − Used to save your sketch.

F − Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

**Raspberry Pi**

Raspberry Pi, developed by Raspberry Pi Foundation in association with Broadcom, is a series of small single-board computers and perhaps the most inspiring computer available today. Earlier, the Raspberry Pi was used to teach basic computer science in schools but later, because of its low cost and open design, the model became far more popular than anticipated. It is widely used to make gaming devices, fitness gadgets, weather stations, and much more. But apart from that, it is used by thousands of people of all ages who want to take their first step in computer science.

**Generations and Models**

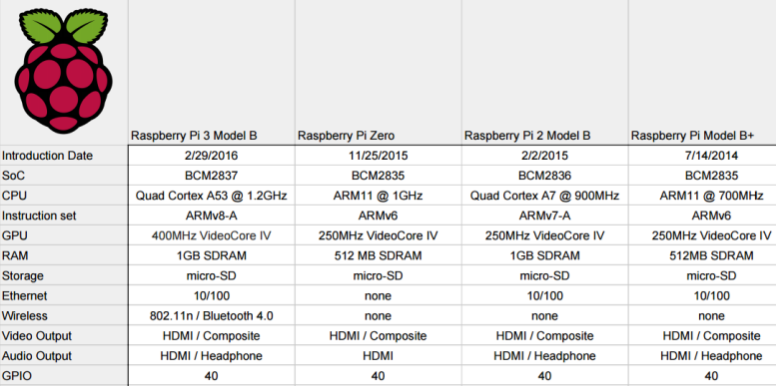
In 2012, the company launched the Raspberry Pi and the current generations of regular Raspberry Pi boards are Zero, 1, 2, 3, and 4. Generation 1 Raspberry Pi had the following four options:

* **Model A**
* **Model A +**
* **Model B**
* **Model B +**

Among these models, the Raspberry Pi B models are the original credit-card sized format. On the other hand, the Raspberry Pi A models has a smaller and more compact footprint and hence, these models have the reduced connectivity options. Raspberry Pi Zero models, which come with or without GPIO (general-purpose input output) headers installed, are the most compact of all the Raspberry Pi boards types.

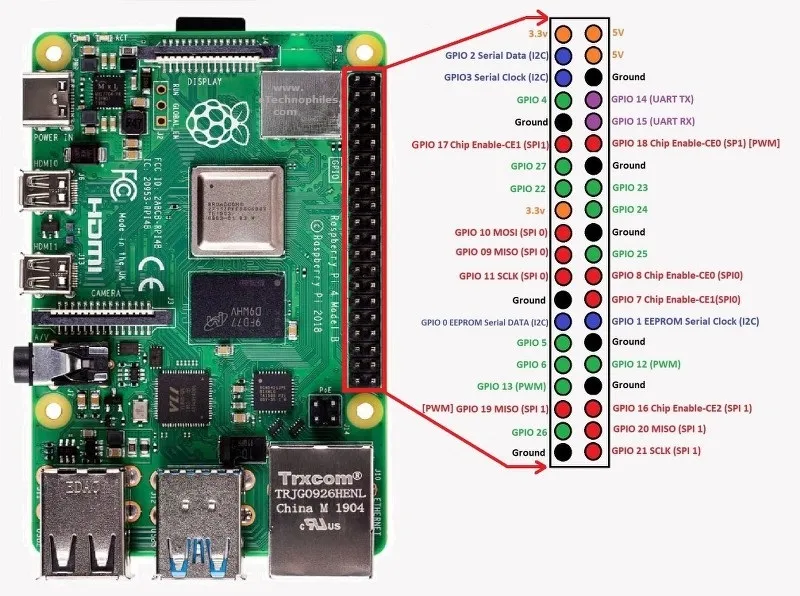
The best use of Raspberry Pi is to learn how a computer works. It is easy to learn how to make electronic projects or programs with it. It comes with two programming languages, **Scratch** and **Python**. Through GPIO (general purpose input output) pins, Raspberry Pi can be connected to other circuits to control the devices of choice.

**Specifications**



**Fig: Comparison of various models of Raspberry Pi**

**PIN Diagram**

****

**Fig: PIN Diagram of Raspberry Pi**

The basic set up for Raspberry Pi includes the following:

* HDMI Cable.
* Monitor.
* Keyboard.
* Mouse.
* 5 Volt Power Adapter.
* LAN Cable.
* Min- 2GB micro SD Card.

**Operating System**

The official supported OS are:

* Raspbian.
* NOOBS.

Some of the third party OS are:

* UBUNTU mate.
* Snappy Ubuntu core.
* Windows 10 core.
* Pinet.
* RISC OS.

**Experiment – 1**

**Aim:** Start Raspberry Pi and try various Linux commands in command terminal window:

ls, cd, touch, mv, rm, man, mkdir, rmdir, tar, gzip, cat, more, less, ps, sudo,cron, chown, chgrp, ping, chmod, curl, echo, exit, find, finger, free, grep, kill etc.

**Linux Commands:**

**1. ls:** It lists directory contents of files and directories.

- Display All Information About Files / Directories Using $ ls -l

- Open Last Edited File Using ls -t

- Display File Size in Human Readable Format Using ls –lh

**2. cd:** cd command in linux known as change directory command. It is used to change current working directory.

**Syntax:**

**$ cd [directory]**

**- cd /:** this command is used to change directory to the root directory

**- cd dir\_1/dir\_2/dir\_3:** This command is used to move inside a directory from a directory

**- cd.. :** This command is used to move to the parent directory of current directory or the directory one level up from the current directory.

**- cd “dir name”:** This command is used to navigate to a directory with white spaces.

**3. touch:** Used to create, change and modify timestamps of a file. It is used to create a file without any content.

**Touch command Syntax to create a new file:**

touch file\_name

**Touch command to create multiple files:**

touch File1\_name File2\_name File3\_name

**4. mv:** mv stands for move. mv is used to move one or more files or directories from one place to another in file system. It has two distinct functions.It renames a file or folder. It moves group of files to different directory.

**Syntax:**

**mv [Option] source destination**

**5. rm**: rm stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system

**Syntax:**

**rm [OPTION]... FILE...**

**6. man: man** command in Linux is used to display the user manual of any command

**Syntax:**

**$man [OPTION]... [COMMAND NAME]...**

**7. mkdir:** mkdir command in Linux allows the user to create directories

**Syntax:**

**$man [OPTION]... [COMMAND NAME]...**

**8. rmdir:** rmdir command in Linux allows the user to create directories

**Syntax:**

rmdir [-p] [-v | –verbose] [–ignore-fail-on-non-empty] directories …

**9. tar:** The Linux ‘tar’ stands for tape archive, is used to create Archive and extract the Archive files.

**Syntax:**

**tar [options] [archive-file] [file or directory to be archived]**

**10. gzip:** gzip command compresses files.

**Syntax:**

**gzip [Options] [filenames]**

**Example:**

$ gzip mydoc.txt]

**11. cat:** It reads data from the file and gives their content as output. It helps us to create, view and concatenate files.

**Syntax:**

$cat filename

**12. more:** Thiscommand is used to view the text files in the command prompt, displaying one screen at a time in case the file is large (For example log files). The more command also allows the user do scroll up and down through the page.

**Syntax:**

**more [-options] [-num] [+/pattern] [+linenum] [file\_name]**

∙ **[-options]**: any option that you want to use in order to change the way the file is displayed. Choose any one from the followings: (-d, -l, -f, -p, -c, -s, -u)

∙ **[-num]**: type the number of lines that you want to display per screen.∙ **[+/pattern]**: replace the pattern with any string that you want to find in the text file.

∙ **[+linenum]**: use the line number from where you want to start displaying the text content.

∙ **[file\_name]**: name of the file containing the text that you want to display on the screen.

**13. Less:** less command is linux utility which can be used to read contents of text file one page (one screen) per time. It has faster access because if file is large, it don’t access complete file, but access it page by page.

**Syntax:**

less filename

**14. ps**: It is usedfor viewing information related with the processes on a system which stands as abbreviation for **“Process Status”**

**Syntax –**

**ps [options]**

**15. sudo:** sudo (**S**uper **U**ser **DO**) command in Linux is generally used as a prefix of some command that only super user are allowed to run.

**16. cron:** automates the scheduled task at a predetermined time. It is adaemon process, which runs as a background process.

**Syntax:**

cron [-f] [-l] [-L loglevel]

**17. chown:** chown command is used to change the file Owner or group.

**Syntax:**

chown [OPTION]… [OWNER][: [GROUP]] FILE…

chown [OPTION]… –reference=RFILE FILE…

**18. chgrp**: chgrp command in Linux is used to change the group ownership of a file or directory.

**Syntax:**

Chgrp [OPTION]… GROUP FILE…

chgrp [OPTION]… –reference=RFILE FILE…

**19. ping:** PING (Packet Internet Groper) command is used to check the network connectivity between host and server/host.

**20. chmod:** The chmod command is used to change the access mode of a file. The name is an abbreviation of **change mode**.

**Syntax:**

chmod [reference][operator][mode] file...

**21. curl:** curlis a command line tool to transfer data to or from a server, using any of the

supported protocols

**Syntax:**

curl [options] [URL...]

**22. echo:** echo command in linux is used to display line of text/string that are passed as an argument .

**Syntax:**

**echo [option] [string]**

**23. exit: exit** command in linux is used to exit the shell where it is currently running. It takes one more parameter as *[N]* and exits the shell with a return of status *N*.

**Syntax:**

exit [n]

**24. find:** It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions.

**Syntax:**

$ find [where to start searching from]

[expression determines what to find] [-options] [what to find]

**25. Finger:** command is a user information lookup command which gives details of all the users logged in.

**Syntax:**

$finger[username]

**26. Free:** command which displays the total amount of free space available along with the amount of memory used and swap memory in the system, and also the buffers used by the kernel.

**Syntax:**

$free [OPTION]

**27. Grep:** The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern.

**Syntax:**

grep [options] pattern [files]

**28. Kill**: command in Linux (located in /bin/kill), is a built-in command which is used to terminate processes manually. Killcommand sends a signal to a process which terminates the process.

**Experiment – 2**

**Aim:** Run some python programs on Pi like:

* 1. Read your name and print Hello message with name.

**Python Program:**

name = input('What is your name?\n')

print ('Hello %s.' % name)

**Output:**

What is your name?

IoT Laboratory

Hello IoT Laboratory

* 1. Read two numbers and print their sum, difference, product and division.

**Python Program:**

num1 = int(input("Enter First Number: "))

num2 = int(input("Enter Second Number: "))

print("Enter which operation would you like to perform?")

ch = input("Enter any of these char for specific operation +,-,\*,/: ")

result = 0

if ch == '+':

result = num1 + num2

elif ch == '-':

result = num1 - num2

elif ch == '\*':

result = num1 \* num2

elif ch == '/':

result = num1 / num2

else:

print("Input character is not recognized!")

print(num1, ch , num2, ":", result)

**Output 1: Addition**

Enter First Number: 100

Enter Second Number: 5

Enter which operation would you like to perform?

Enter any of these char for specific operation +,-,\*, /: +

100 + 5: 105

**Output 2: Subtraction**

Enter First Number: 8

Enter Second Number: 7

Enter which operation would you like to perform?

Enter any of these char for specific operation +,-,\*, /: -

8 - 7: 1

**Output 3: Multiplication**

Enter First Number: 6

Enter Second Number: 8

Enter which operation would you like to perform?

Enter any of these char for specific operation +,-,\*, /: \*

6 \* 8: 48

**Output 4: Division**

Enter First Number: 20

Enter Second Number: 5

Enter which operation would you like to perform?

Enter any of these char for specific operation +,-,\*, /: /

* + - * 1. / 5 : 4.0
  1. Word and character count of a given string.

**Python Program:**

word\_count = 0

char\_count = 0

usr\_input = input("Enter a string : ")

split\_string = usr\_input.split()

word\_count = len(split\_string)

for word in split\_string:

char\_count += len(word)

print("Total words : {}".format(word\_count))

print("Total characters : {}".format(char\_count))

**Output:**

Enter a string: Hello, How are you?

Total words: 4

Total characters: 16

* 1. Area of a given shape (rectangle, triangle and circle) reading shape and appropriate values from standard input.

**Python Program:**

width = float(input('Please Enter the Width of a Rectangle: '))

height = float(input('Please Enter the Height of a Rectangle: '))

# calculate the area

Area = width \* height

# calculate the Perimeter

Perimeter = 2 \* (width + height)

print("\n Area of a Rectangle is: %.2f" %Area)

print(" Perimeter of Rectangle is: %.2f" %Perimeter)

**Output:**

Please Enter the Width of a Rectangle: 22

Please Enter the Height of a Rectangle: 5

Area of a Rectangle is: 110.00

Perimeter of Rectangle is: 54.00

**Python Program:**

# Python Program to find the area of triangle

# Three sides of the triangle a, b and c are provided by the user

a = float(input('Enter first side: '))

b = float(input('Enter second side: '))

c = float(input('Enter third side: '))

# calculate the semi-perimeter

s = (a + b + c) / 2

# calculate the area

area = (s\*(s-a)\*(s-b)\*(s-c)) \*\* 0.5

print('The area of the triangle is %0.2f' %area)

**Output:**

Enter first side: 3

Enter second side: 4

Enter third side: 5

The area of the triangle is 6.00

**Python Program:**

# Python Program to find Diameter, Circumference, and Area of a Circle

PI = 3.14

radius = float(input(' Please Enter the radius of a circle: '))

diameter = 2 \* radius

circumference = 2 \* PI \* radius

area = PI \* radius \* radius

print(" \n Diameter of a Circle = %.2f" %diameter)

print(" Circumference of a Circle = %.2f" %circumference)

print(" Area of a Circle = %.2f" %area)

**Output:**

Please Enter the radius of a circle: 5

Diameter of a Circle = 10.00

Circumference of a Circle = 31.40

Area of a Circle = 78.50

**Combined Python Program:**

#Area

print("Select one of the following:")

print("1. Rectangle\n2. Triangle\n3. Circle")

s=input("Enter your choice: ")

if s=='1':

x=int(input("Enter length:"))

y=int(input("Enter breadth:"))

print("Area={}".format(x\*y))

elif s=='2':

x=int(input("Enter base:"))

y=int(input("Enter height:"))

print("Area={}".format(0.5\*x\*y))

elif s=='3':

x=int(input("Enter radius:"))

print("Area={}".format(3.14\*x\*x))

else:

print("Enter a valid choice")

**Experiment – 3**

**Aim:** Run some python programs on Pi like:

* 1. Print a name 'n' times, where name and n are read from standard input, using for and while loops.

**Python Program (Using For Loop):**

i=1

print ("enter the name")

name=raw\_input()

print ("enter the no of time" )

num=raw\_input()

#print (type(num))

num=int(num)

for i in range(1,num+1):

print (i , name)

i=i+1

**Python Program (Using While Loop):**

print ("enter the name")

name=raw\_input()

print ("enter the no of time" )

num=raw\_input()

print (type(num))

num=int(num)

i=1

while(i<=num):

print (name)

i=i+1

**Python Program (Without Loop):**

def name(n):

if n != 0:

name(n-1)

print("Name")

name(10)

* 1. Handle Divided by Zero Exception.

**Python Program:**

print ("enter two no n1 and n2")

n1=raw\_input()

n2=raw\_input()

n1=int(n1)

n2=int(n2)

try:

div=n1/n2

print (div)

except ZeroDivisionError:

print ("zero division is handled")

print ("out of try catch block ")

**Output:**

Enter n1:10

Enter n2:0

zero division is handled

**#DivideByZero Exception**

x=int(input("First No:"))

y=int(input("Second No:"))

try:

print("x/y={}".format(x/y))

except Exception:

print("DivideByZero Exception")

* 1. Print current time for 10 times with an interval of 10 seconds.

**Python Program:**

import time

for i in range(1,11):

zz=time.asctime(time.localtime(time.time()))

zz=zz[11:19]

print (zz)

print (time.asctime(time.localtime(time.time())))

time.sleep(10)

**Python Program:**

#Current time 10 times

import datetime

import time

for i in range(0,10):

print(datetime.datetime.now().time())

time.sleep(10)

* 1. Read a file line by line and print the word count of each line.

**Python Program:**

file=open("eee.txt","r")

line=1

for i in file:

print ("print the line no=" , line , "and line is =" , i)

z=i.split()

print ("no of word in line =" , line ,"is = " , len(z))

line = line+1

**Python Program:**

# read line by line from file

L = ["Welcome to India", "Delhi is the capital\n", "Have fun\n"]

file1 = open('myfile.txt', 'w')

file1.writelines((L))

file1.close()

file1 = open('myfile.txt', 'r')

count = 0

while True:

count+=1

line = file1.readline()

if not line:

break

print("Word Count in Line{}: {}".format(count,

len(line.split())) )

file1.close()

**Experiment – 4**

**Aim:** LED blinking using Raspberry pi.

**Components:** Raspberry pi, LED, Resistor, Breadboard

**Explanation:**

The first step in this project is to design a simple LED circuit. Then we will make the LED circuit controllable from the Raspberry Pi by connecting the circuit to the general purpose input/output (GPIO) pins on the Raspberry Pi.

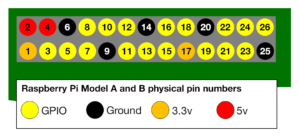
A simple LED circuit consists of a LED and resistor. The resistor is used to limit the current that is being drawn and is called a current limiting resistor. Without the resistor the LED would run at too high of a voltage, resulting in too much current being drawn which in turn would instantly burn the LED, and likely also the GPIO port on the Raspberry Pi.

To calculate the resistor value we need to examine the specifications of the LED. Specifically we need to find the forward voltage (VF) and the forward current (IF). A regular red LED has a forward voltage (VF) of 1.7V and forward current of 20mA (IF). Additionally we need to know the output voltage of the Raspberry Pi which is 3.3V.

We can then calculate the resistor size needed to limit the current to the LED’s maximum forward current (IF) using ohm’s law like this:

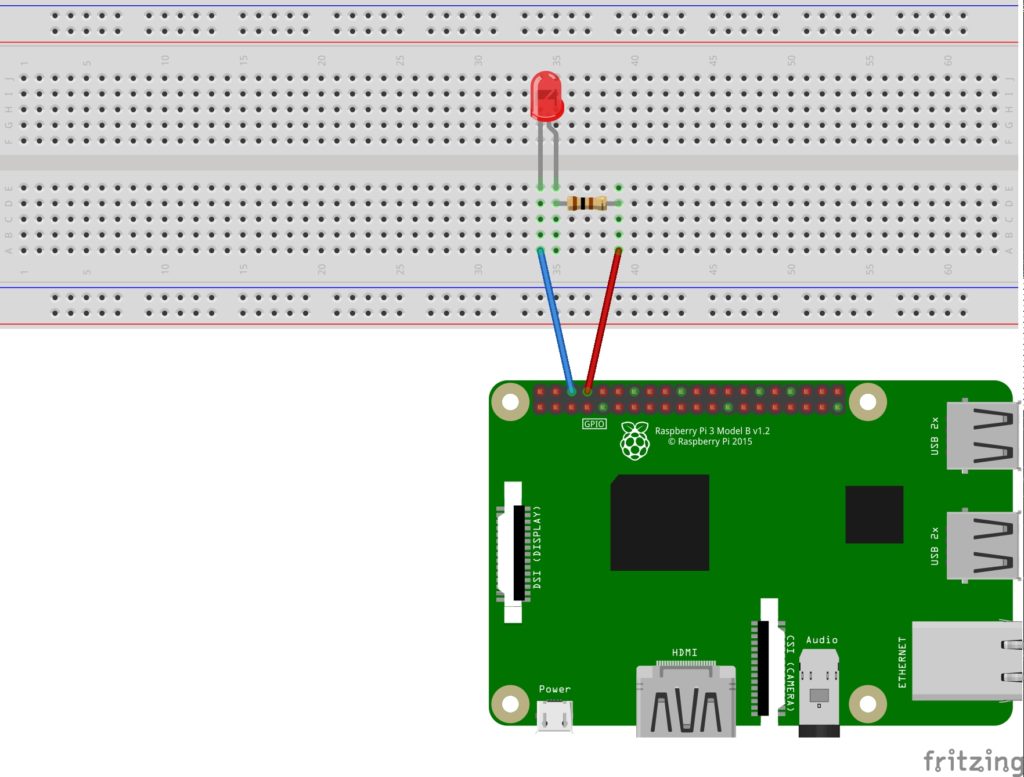
R=V/I= (3.3–VF)/IF= (3.3–1.7)/ (20mA) =80Ω

Unfortunately 80 ohm is not a standard size of a resistor. To solve this we can either combine multiple resistors, or round up to a standard size. In this case we would round up to 100 ohm. When hooking up the circuit, note the polarity of the LED. The LED has a long and a short lead. The long lead is the positive side also called the anode, the short lead is the negative side called the cathode. The long should be connected to the resistor and the short lead should be connected to ground via the blue jumper wire and pin 6 on the Raspberry Pi as shown on the diagram. To find the pin number refer to this diagram showing the physical pin numbers on the Raspberry Pi.



**Fig: Physical Pin Diagram of Raspberry Pi**

**Circuit:**



**Fig: Circuit Diagram**

**Code:**

importRPi.GPIOas GPIO # Import Raspberry Pi GPIO library

from *time* import sleep # Import the sleep function from the time module

GPIO.setwarnings(False) # Ignore warning for now

GPIO.setmode(GPIO.BOARD) # Use physical pin numbering

GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW) # Set pin 8 to be an output pin and set initial value to low (off)

whileTrue: # Run forever

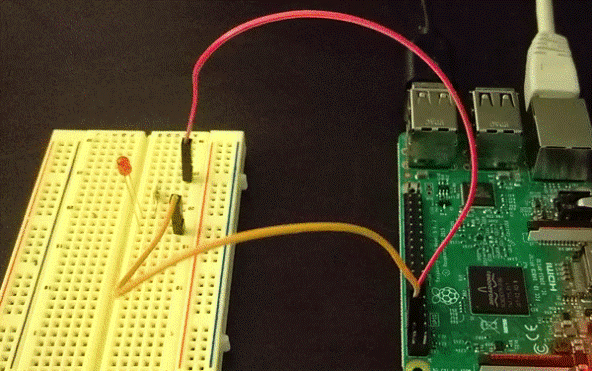
GPIO.output(8, GPIO.HIGH) # Turn on

sleep (1) # Sleep for 1 second

GPIO.output(8, GPIO.LOW) # Turn off

sleep (1) # Sleep for 1 second

**Output:**

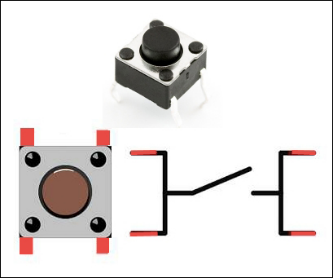


**Experiment 5**

**Aim:** LED blinking with push-button using Raspberry pi.

**Components:** Raspberry pi, LED, Resistor, Breadboard, Push-button

**Circuit:**



**Fig: Push-button**

**Code:**

importRPi.GPIOas GPIO

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BOARD)

GPIO.setup(16, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(8, GPIO.OUT,initial=GPIO.LOW)

While True:

If GPIO.input(16) == GPIO.LOW

print(“Button pushed”)

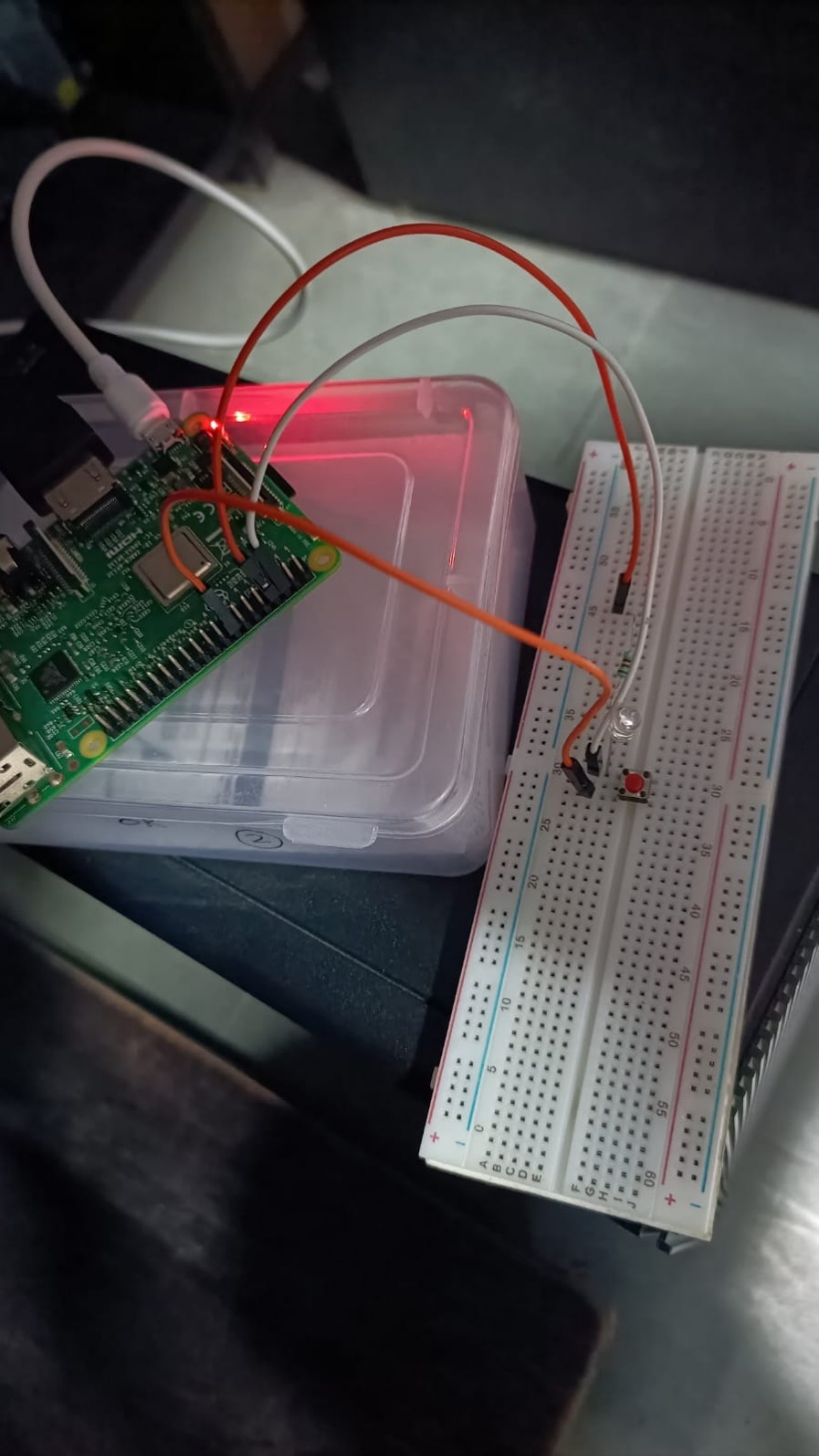
GPIO.output(8,GPIO.HIGH)

else:

print(“Button released”)

GPIO.output(8,GPIO.LOW)

**Output**



**Fig: Output of LED Blinking**

**Experiment 6**

**Aim:** Flash an LED at a given on time and off time cycle, where the two times are taken from a file.

**Components:** Raspberry pi, LED, Resistor

**Code:**

import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library

from time import sleep # Import the sleep function from the time module

GPIO.setwarnings(False) # Ignore warning for now

GPIO.setmode(GPIO.BOARD) # Use physical pin numbering

GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW) # Set pin 8 to be an output pin and set initial value to low (off)

f=open("delay.txt","r")

on\_time=int(f.readline())

off\_time=int(f.readline())

while True:

GPIO.output(8, GPIO.HIGH) # Turn on

sleep(on\_time) #

GPIO.output(8, GPIO.LOW) # Turn off

sleep(off\_time) #

**Experiment – 7**

**Aim:** Flash an LED based on cron output (acts as an Alarm).

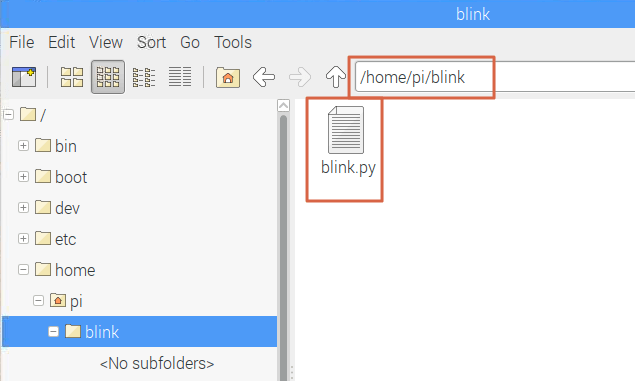
**Components:** Raspberry pi, LED, Resistor.

**Explaination:**

Unlike [Arduino](https://www.factoryforward.com/product/arduino-uno-r3-clone/), the [Raspberry Pi](https://www.factoryforward.com/product/raspberry-pi-3-india/) won’t run the code until it is executed manually. We need a scheduler to run the python code when the Pi is powered up or rebooted. Crontab (cron table) is scheduler used in Linux to schedule a specific task at a specific time. It is mostly used by Linux system Administrators to automate his daily tasks like scheduling backups and deleting old files, etc., Here we will be using it to launch the code during the startup, so we will be creating a shell script to navigate to the python code first and then launch it during startup using crontab.

Step 1:

First, create a ‘**blink**’ folder and then create an empty ‘**blink.py**’ file. It should be inside the path /home/pi/blink. (The /home/pi/ is the default user files path).



**Python code: blink.py**

import RPi.GPIO as GPIO

import time

LedPin = 11    # pin11

def setup():

  GPIO.setmode(GPIO.BOARD)       # Numbers GPIOs by physical location

  GPIO.setup(LedPin, GPIO.OUT)   # Set LedPin's mode is output

  GPIO.output(LedPin, GPIO.HIGH) # Turn ON led

def blink():

  while True:

    GPIO.output(LedPin, GPIO.HIGH)  # led on

    time.sleep(1)

    GPIO.output(LedPin, GPIO.LOW) # led off

    time.sleep(1)

def destroy():

  GPIO.output(LedPin, GPIO.LOW)   # led off

  GPIO.cleanup()                  # Release resource

if \_\_name\_\_ == '\_\_main\_\_':     # Program start from here

  setup()

  try:

    blink()

  except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the child program destroy() will be  executed.

    destroy()

**Step 2:**

Open the terminal and navigate to the ‘blink’ folder by using the following command

**cd blink**

Create a launcher script by typing the command in the terminal.

**nano launcher.sh**

**Step 3:**

Now the editor will get launched. Copy and paste the following code to the editor. This shell script will navigate to the blink code.

#! /bin/sh

# launcher.sh

# navigate to home directory, then to this directory, and then execute python script, then back home

cd /

cd home/pi/blink

sudo python blink.py

cd /

Now press Ctrl – X and the press Y to Save. It will ask for filename leave it as ‘launcher.sh’ itself and press enter.

**Step 4:**

We want to make this launcher.sh script executable. Type the following command for it.

chmod 755 launcher.sh

You can test the launcher script by typing the following command if you want to test.

sh launcher.sh

**Step 5:**

Create a log directory for capturing logs incase of any errors. Navigate back to home directory and create a log directory there.

cd

mkdir logs

**Step 6:**

We need to schedule the launcher script to run during startup itself. Open the **crontab**by using the command.

sudo crontab –e

Once the crontab window opens add this line at the end of the file.

50 9 \* \* 4 python3 /home/rasp-pi/blink/blink.py

#Please refer https//:crontab.guru

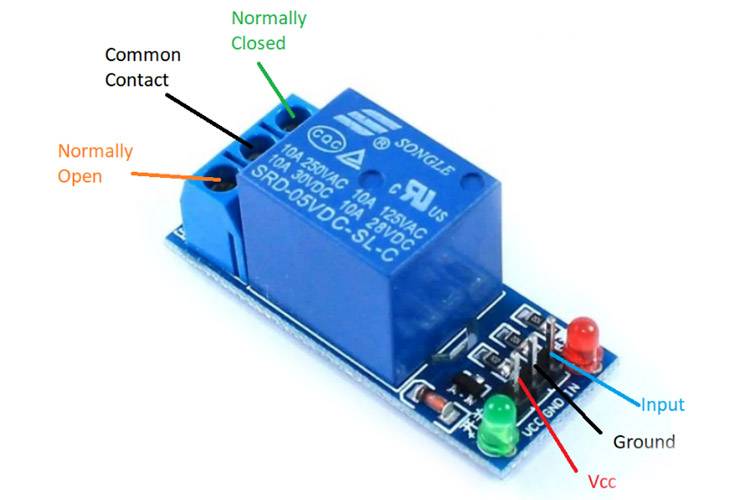
# Now Press CTRL+X , to save type Y, remain the file name as it is press enter

**Experiment – 8**

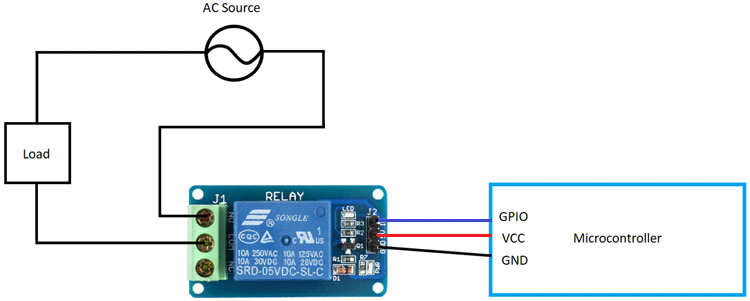
**Aim:** Switch on a relay at a given time using cron, where the relays contact terminals are connect to load.

**Components:** Raspberry Pi, Relay, Bulb.

**Explanation:** Relay is an electro-mechanical device which acts as a switch. DC electrical current is used to energize the relay coil which opens or closes the contact switches. Internal circuit of a single channel 5V relay consists of normally open contacts, normally closed contacts and a coil.



**Relay Module**



**Relay Connection**

**Code:**

importRPi.GPIOas GPIO # Import Raspberry Pi GPIO library

from *time* import sleep # Import the sleep function from the time module

GPIO.setwarnings(False) # Ignore warning for now

GPIO.setmode(GPIO.BOARD) # Use physical pin numbering

GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW) # Set pin 8 to be an output pin and set initial value to low (off)

whileTrue: # Run forever

GPIO.output(8, GPIO.HIGH) # Turn on

sleep (5)

GPIO.output(8, GPIO.LOW) # Turn off

sleep (5)

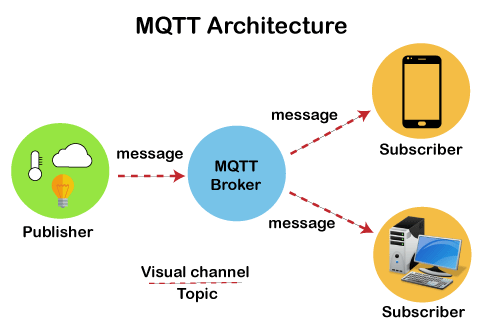
**Experiment -9**

**Aim**: Get the status of bulb at a remote place on the LAN through Web.

**Explanation:**

**MQTT Protocol**

MQTT stands for Message Queuing Telemetry Transport. MQTT is a machine to machine internet of things connectivity protocol. It is an extremely lightweight and publish-subscribe messaging transport protocol. This protocol is useful for the connection with the remote location where the bandwidth is a premium. These characteristics make it useful in various situations, including constant environment such as for communication machine to machine and internet of things contexts. It is a publish and subscribe system where we can publish and receive the messages as a client. It makes it easy for communication between multiple devices. It is a simple messaging protocol designed for the constrained devices and with low bandwidth, so it's a perfect solution for the internet of things applications.



Now we will look at the architecture of MQTT. To understand it more clearly, we will look at the example. Suppose a device has a temperature sensor and wants to send the rating to the server or the broker. If the phone or desktop application wishes to receive this temperature value on the other side, then there will be two things that happened. The publisher first defines the topic; for example, the temperature then publishes the message, i.e., the temperature's value. After publishing the message, the phone or the desktop application on the other side will subscribe to the topic, i.e., temperature and then receive the published message, i.e., the value of the temperature. The server or the broker's role is to deliver the published message to the phone or the desktop application.

import paho client:

sudo pip install paho-mqtt

sudo pip3 install paho-mqtt

**Code:**

from time import sleep

import os,sys

import RPi.GPIO as GPIO

import paho.mqtt.client as paho

#from urllib.parse import urlparse

import urllib.parse as urlparse

#from urllib.parse import urlparse

#import urlparse

GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)

LED\_PIN = 8 #define LED pin

GPIO.setup(LED\_PIN,GPIO.OUT) # Set pin function as output

GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW)

def on\_connect(self, mosq, obj, rc):

self.subscribe("led", 0)

def on\_message(mosq, obj, msg):

print(msg.topic + " " + str(msg.qos) + " " + str(msg.payload))

print(msg.payload)

if(msg.payload == b'on'):

print ("LED on")

GPIO.output(LED\_PIN,GPIO.HIGH) #LED ON

else:

print ("LED off")

GPIO.output(LED\_PIN,GPIO.LOW) # LED OFF

def on\_publish(mosq, obj, mid):

print("mid: " + str(mid))

def on\_subscribe(mosq, obj, mid, granted\_qos):

print("Subscribed: " + str(mid) + " " + str(granted\_qos))

mqttc = paho.Client() # object declaration

# Assign event callbacks

mqttc.on\_message = on\_message # called as callback

mqttc.on\_connect = on\_connect

mqttc.on\_publish = on\_publish

mqttc.on\_subscribe = on\_subscribe

#url\_str = os.environ.get('CLOUDMQTT\_URL', 'tcp://broker.emqx.io:1883') # pass broker addr e.g. "tcp://iot.eclipse.org"

#url\_str = os.environ.get('CLOUDMQTT\_URL', 'tcp://broker.hivemq.com:1883')

url\_str = os.environ.get('CLOUDMQTT\_URL', 'tcp://broker.emqx.io:1883')

url = urlparse.urlparse(url\_str)

mqttc.connect(url.hostname, url.port)

rc = 0

while True:

while rc == 0:

import time

rc = mqttc.loop()

#time.sleep(0.5)

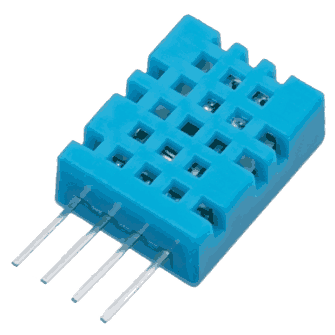
print("rc: " + str(rc))

**Experiment – 10**

**Aim:** Interface Temperature & Humidity sensor using Raspberry pi and send the data over ThingSpeak cloud Infrastructure.

**Components:** Raspberry pi, DHT11/DHT22 sensor.

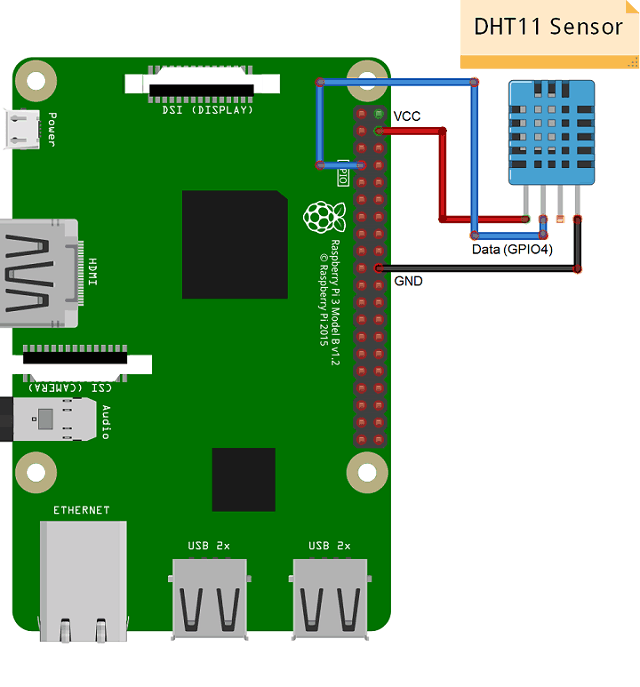
**Explanation:**



**DHT11 Sensor**

* DHT11 sensor measures humidity and temperature values serially over a single wire.
* It can measure relative humidity in percentage (20 to 90% RH) and temperature in degree Celsius in the range of 0 to 50°C.
* It has 4 pins; one of which is used for data communication in serial form.
* Pulses of different TON and TOFF are decoded as logic 1 or logic 0 or start pulse or end of a frame.

**Circuit:**



**Code:**

* Here, we are going to interface DHT11 sensor with Raspberry Pi 3 and display Humidity and Temperature on terminal.
* We will be using the DHT Sensor Python library by Adafruit from GitHub. The Adafruit Python DHT Sensor library is created to read the Humidity and Temperature on raspberry Pi or Beaglebone Black. It is developed for DHT series sensors like DHT11, DHT22 or AM2302.
* <https://github.com/adafruit/Adafruit_Python_DHT/archive/master.zip>

Extract the library and install it in the same root directory of downloaded library by executing following command,

sudo python setup.py install

* Once the library and its dependencies has been installed, open the example sketch named simple test from the library kept in examples folder.
* In this code, raspberry Pi reads Humidity and Temperature from DHT11 sensor and prints them on terminal. But, it read and display the value only once. So, here we made change in the program to print value continuously.

**Note:**

* Assign proper sensor type to the sensor variable in this library. Here, we are using DHT11 sensor.

sensor = Adafruit\_DHT.DHT11

* If anyone is using sensor DHT22 then we need to assign **Adafruit\_DHT.DHT22**to the sensor variable shown above.
* Then assign pin no. to which DHT sensor’s data pin is connected. Here, data out of DHT11 sensor is connected to GPIO4. As shown in above interfacing diagram.

**sudo apt-get install http.client**

#Code:

import Adafruit\_DHT

import http.client

import urllib

import time

key = "CKUNX8CN1DHO6MO0" # Put your API Key here

# Sensor should be set to Adafruit\_DHT.DHT11,

# Adafruit\_DHT.DHT22, or Adafruit\_DHT.AM2302.

sensor = Adafruit\_DHT.AM2302

# Example using a Raspberry Pi with DHT sensor

# connected to GPIO4.

pin = 4

def thermometer():

while True:

#Calculate CPU temperature of Raspberry Pi in Degrees C

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

#print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

else:

print('Failed to get reading. Try again!')

params = urllib.parse.urlencode({'field1': temperature, 'field2': humidity, 'key':key })

headers = {"Content-typZZe": "application/x-www-form-urlencoded","Accept": "text/plain"}

conn = http.client.HTTPConnection("api.thingspeak.com:80")

try:

conn.request("POST", "/update", params, headers)

response = conn.getresponse()

#print (temp)

print (response.status, response.reason)

data = response.read()

conn.close()

except:

print ("connection failed")

break

if \_\_name\_\_ == "\_\_main\_\_":

while True:

thermometer()

**ThingSpeak Cloud:**

ThingSpeak is an open IoT platform for monitoring the data online. In ThingSpeak, the data in the channel can be set as private or public according to the choice. ThingSpeak takes minimum of 15 seconds to update the readings. It’s a great and very easy to use platform for building IOT projects.

The first step in building any IoT project using Raspberry Pi is to upload the data to any cloud server using Raspberry Pi. In this simplest Raspberry Pi IOT project, ThingSpeak is used as a cloud server to store the data. Here Raspberry Pi will read DHT11/DHT22 sensor data and send it to ThingSpeak. The sensed data can be monitored from anywhere in the world via internet. This will be useful for some application where Pi is running for long time at some remote place and CPU temperature needs to be monitored.

**Step 1: Signup for ThingSpeak**

For creating the channel on ThingSpeak you first need to sign up on ThingSpeak. In case if you already have account on ThingSpeak just sign in using your id and password.

For creating your account go to [www.thingspeak.com](http://www.thingspeak.com)

**Step 2: Create a Channel for Your Data**

Once Signing is done, create a new channel by clicking “New Channel” button

**Step 3: Getting API Key in ThingSpeak**

To send data to ThingSpeak, a unique API key is needed, which will be used later in the python code to upload the CPU data onto ThingSpeak Website.

Click on “API Keys” button to get the unique API key for uploading the CPU data.

**Step 4:  Python Code for Raspberry Pi**

Run the script that is mentioned on previous page

**Step 5: Check ThingSpeak site for Data Logging**

After completing these steps, open the channel and observe that the DHT sensor data is getting updated into ThingSpeak website.

**Experiment –11**

**Aim:** To measure distance using Ultrasonic Sensor.

**Components:** ARDUINO UNO, Ultrasonic Sensor

**Explanation:**

Ultrasonic Sensors, particularly HC-SR04 Ultrasonic Sensor, are very popular among electronic hobbyists and are frequently used in a variety of projects like Obstacle Avoiding Robot, Distance Measurement, Proximity Detection and so forth. In this project, we will learn about HC-SR04 Ultrasonic and see how to interface one with Raspberry Pi.



We will now see how to measure the distance of an object using HC-SR04 Ultrasonic Sensor. In order to send the 40 KHz Ultrasound, the TRIG Pin of the Ultrasonic Sensor must be held HIGH for a minimum duration of 10µS.After this, the Ultrasonic Transmitter, will transmits a burst of 8-pulses of ultrasound at 40 KHz. Immediately, the control circuit in the sensor will change the state of the ECHO pin to HIGH. This pin stays HIGH until the ultrasound hits an object and returns to the Ultrasonic Receiver. Based on the Time for which the Echo Pin stays HIGH, you can calculate the distance between the sensor and the object. For example, if we calculated the time for which ECHO is HIGH as 588µS, then you can calculate the distance with the help of the speed of sound, which is equal to 340m/s.

Distance = Velocity of Sound / (Time/2) = 340m/s / (588µS /2) = 10cm.

Physical Connections:-

In breadboard connect ultrasonic sensor

Back side pins of sensor:- ( in breadboard vertical connections( in between) are there)

1) VCC to 5v in Arduino board

2) Trig to pin 13 of digital pin of Arduino

3) Echo to pin 12 of digital pin of Arduino

4) Gnd to Gnd of Arduino board

Connect USB from computer to Arduino

**Code:**

const int trigPin = 13; const int echoPin = 12; void setup() {

Serial.begin(9600);} void loop()

{

long duration, inches, cm;

pinMode(trigPin, OUTPUT);

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

pinMode(echoPin, INPUT);

duration = pulseIn(echoPin, HIGH);

inches = microsecondsToInches(duration);

cm = microsecondsToCentimeters(duration);

Serial.print(inches);

Serial.print("in, ");

Serial.print(cm);

Serial.print("cm");

Serial.println();

delay(1000);

}

long microsecondsToInches(long microseconds)

{return microseconds / 74 / 2;

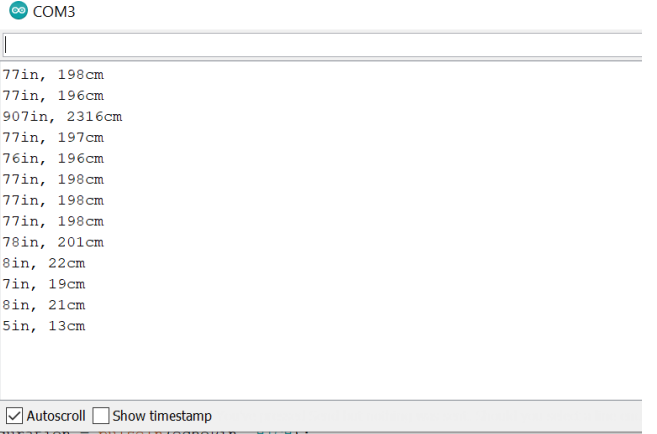
}

long microsecondsToCentimeters(long microseconds)

{return microseconds / 29 / 2;}

Verify your code in Arduino IDE that is sketch. Upload your code in Arduino IDE. After connecting, all done. Press Ctrl + shift + M. You can see output window

**Output:**



**Experiment – 12**

**Aim:** Design a system using Arduino to monitor Temperature and Humidity.

**Hardware Library:** ARDUINO UNO

**Components:** Temperature and Humidity sensor, ARDUINO UNO

**Theory:**

**DHT11 Sensor:**

The DHT11 is a basic, low cost digital temperature and humidity sensor.

∙ DHT11 is a single wire digital humidity and temperature sensor, which provides humidity and temperature values serially with one-wire protocol.

∙ DHT11 sensor provides relative humidity value in percentage (20 to 90% RH) and temperature values in degree Celsius (0 to 50 °C).

∙ DHT11 sensor uses resistive humidity measurement component, and NTC temperature measurement component.

**Explanation:**

Physical Connections:-

In breadboard connect DHT11 sensor

Back side pins of sensor: - (in breadboard vertical connections (in between) are there)

1) VCC to 5v in Arduino board

2) DATA to pin 4 of digital pin of Arduino

3) Gnd to Gnd of Arduino board

Connect USB from laptop to Arduino

**Code:**

#include <dht11.h>

#define DHT11PIN 4

dht11 DHT11;

void setup()

{

Serial.begin(9600);

}

void loop()

{

Serial.println();

int chk = DHT11.read(DHT11PIN);

Serial.print("Humidity (%): ");

Serial.println((float)DHT11.humidity, 2);

Serial.print("Temperature (C): "); Serial.println((float)DHT11.temperature, 2);

delay(2000); }

**Beyond Syllabus**

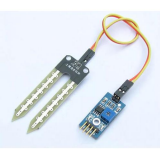
**Experiment – 1**

**Aim:** Design a system using Arduino to monitor Soil Moisture of a plant on an IDE. If soil moisture is less, then LED should glow.

**Hardware Library:** ARDUINO UNO

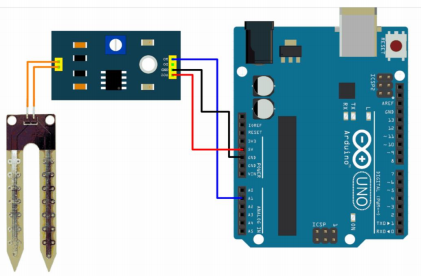
**Components:** Soil Moisture Sensor, Arduino UNO, Jumping wires, LED

**Theory:**

****

Soil moisture sensors measure the volumetric water content in soil. Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighing of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content.

**Connections:**

****

**Code:**

int sensorPin = A0;

int sensorValue;

int limit = 300;

void setup() {

Serial.begin(9600);

pinMode(13, OUTPUT);

}

void loop() {

sensorValue = analogRead(sensorPin);

Serial.println("Analog Value : ");

Serial.println(sensorValue);

if (sensorValue<limit) {

digitalWrite(13, HIGH);

}

else {

digitalWrite(13, LOW);

}

delay(1000);

}

**Output:**

****

**Experiment – 2**

**Aim:** Design a system using Arduino that blinks an LED when you detect a motion in an area on an IDE. Write working, connections, code and expected output.

**Hardware Library:** ARDUINO UNO

**Components:** PIR Sensor, Arduino UNO, Jumping wires, LED

**Theory:**

****

"Passive Infra-Red" sensor is a "Pyroelectric IR Sensor" which generates energy when exposed to heat. Everything emits some low level of radiation, the hotter the object is, the more radiation is emitted. When a human or an animal (with IR radiation wavelength of 9.4µMeter) approaches the sensors range the sensor detects the heat in the form of infrared radiation.

The sensor only detects the energy emitted by other objects and don’t produce any, that's why the sensor is called a PIR or "Passive Infra-Red" sensor. These sensors are small, cheap, rugged, low power and very easy to use.

**Connections:**

With Arduino,

∙ Connect the VCC of the PIR sensor to the 5v pin of Arduino.

∙Then connect the output pin to D13

∙ GND to the Ground pin of the Arduino.

∙ Connect the LED to the D2 pin of the Arduino.

∙ Now upload the code and test

**Code:**

int LED = 13; // the pin that the LED is attached to int PIR = 2; // the pin that the sensor is attached to

void setup() {

pinMode(LED, OUTPUT); // initialize LED as an output

pinMode(PIR, INPUT); // initialize sensor as an input Serial.begin(9600); // initialize serial

}

void loop(){

if (digitalRead(PIR) == HIGH) { // check if the sensor is HIGH digitalWrite(LED, HIGH); // turn LED ON

Serial.println("Motion detected!");

delay(100); // delay 100 milliseconds }

else {

digitalWrite(LED, LOW); // turn LED OFF

Serial.println("Motion stopped!");

delay(100); // delay 100 milliseconds }

}

**Output:**

Motion detected

Motion detected

Motion detected

Motion detected

Motion stopped

Motion stopped

Motion detected

Motion detected

Motion detected

Motion stopped

**Experiment – 3**

**Aim:** Motion detection and blinking the light when the light intensity is low using Arduino.

**Hardware Library:** ARDUINO UNO

**Components:** PIR Sensor, LDR Sensor, Arduino UNO, Jumping wires, LED

**Theory:**

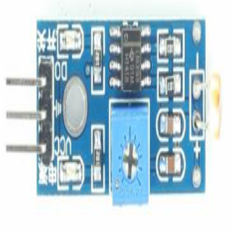
**PIR Sensor**

****

"Passive Infra-Red" sensor is a "Pyroelectric IR Sensor" which generates energy when exposed to heat. Everything emits some low level of radiation, the hotter the object is, the more radiation is emitted. When a human or an animal (with IR radiation wavelength of 9.4µMeter) approaches the sensors range the sensor detects the heat in the form of infrared radiation.

The sensor only detects the energy emitted by other objects and don't produce any, that's why the sensor is called a PIR or “Passive Infra-Red" sensor. These sensors are small, cheap, rugged, low-power and very easy to use.

**LDR Sensor**

****

The LDR Sensor Module is used to detect the presence of light / measuring the intensity of light. The output of the module goes high in the presence of light and it becomes low in the absence of light. The sensitivity of the signal detection can be adjusted using potentiometer.

**Connections:**

With Arduino,

∙ Connect the VCC of the PIR and LDR Sensor to the 5v pin of Arduino.

∙ Then connect the output pin of PIR to D13

∙ And connect the output pin of LDR to D4

∙ Connect the GND of PIR and LDR Sensor to the GND pin of the Arduino.

∙ Connect the LED to the D2 pin of the Arduino.

∙ Now upload the code and test

**Code:**

#define LED 2 // The pin that the LED is attached to

#define LDR 4 // The pin that the LDR Sensor is attached to

#define PIR 13 // The pin that the PIR sensor is attached to

void setup() {

pinMode(LED, OUTPUT); // initalize LED as an output

pinMode(PIR, INPUT); // initialize PIR sensor as an input

pinMode(LDR, INPUT); // initialize LDR sensor as an input

Serial.begin(9600); // initialize serial

}

void loop(){

if(digitalRead(PIR) == HIGH)

{

Serial.println("Motion detected!");

delay(100); // delay 100 milliseconds

if(digitalRead(LDR)<512) //for dark condition!

{

Serial.println("Blinking LED since Dark");

digitalWrite(LED, HIGH);

delay(500);

digitalWrite(LED, LOW);

}

}

else

{

Serial.println("Motion stopped!");

delay(100); // delay 100 milliseconds

}

}

**Output:**

Motion detected!

Motion detected!

Motion detected!

Blinking LED since Dark

Motion detected!

Blinking LED since Dark

Motion stopped!

Motion stopped!

Motion detected!

Motion detected!

Blinking LED since Dark

Motion detected!

Motion stopped!

**Viva Questions**

**1) What is the Internet of Things (IoT)?**

Internet of Things (IoT) is a network of physical objects or people called "things" that are embedded with software, electronics, network, and sensors that allow these objects to collect and exchange data. The goal of IoT is to extend to internet connectivity from standard devices like computer, mobile, tablet to relatively dumb devices like a toaster.

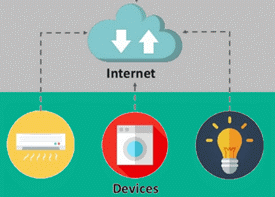
**2) Explain Raspberry Pi**

Raspberry Pi is a computer which is capable of doing all the operations like a conventional computer. It has other features such as onboard WiFi, GPIO pins, and Bluetooth in order to communicate with external things.

**3) How to run Raspberry pi in headless mode?**

Raspberry pi in headless mode can be run by using SSH. The latest operating system has an inbuilt VNC server that is installed for taking remote desktop on Raspberry Pi.

**4) What are the fundamental components of IoT?**

Linux Beginner Tutorial[](https://www.guru99.com/images/2/iot-interview-questions-1.png)

The four fundamental components of an IoT system are:

* **Sensors/Devices:** Sensors or devices are a key component that helps you to collect live data from the surrounding environment. All this data may have various levels of complexities. It could be a simple temperature monitoring sensor, or it may be in the form of the video feed.
* **Connectivity:** All the collected data is sent to a cloud infrastructure. The sensors should be connected to the cloud using various mediums of communications. These communication mediums include mobile or satellite networks, Bluetooth, WI-FI, WAN, etc.
* **Data Processing:** Once that data is collected, and it gets to the cloud, the software product performs processing on the gathered data. This process can be just checking the temperature, reading on devices like AC or heaters. However, it can sometimes also be very complex, like identifying objects, using computer vision on video.
* **User Interface:**The information needs to be available to the end-user in some way, which can be achieved by triggering alarms on their phones or sending them notification through email or text message. The user sometimes might need an interface which actively checks their IoT system.

**5) What is the difference between IoT and IIoT?**

The difference between IoT and IIoT is:

|  |  |
| --- | --- |
| **IoT** | **IIoT** |
| The full form of IoT is the Internet of Things. | The full form of IIoT is the Industrial Internet of Things. |
| A service model is human-centric. | A service model is machine-centric. |
| It supports customer-oriented applications. | It supports industry-oriented applications. |
| Communication transportation is done through wireless devices. | Communication transportation is done through both wired and wireless devices. |
| The quality of data is medium to high. | The quality of data is high to very high. |
| Criticality is not severe. | Criticality is severe. |

**6) List layers of IoT protocol stack**

Layers of IoT protocol stack are:

1) Sensing and information,

2) Network connectivity,

3) Information processing layer,

4) Application layer.

**7) What are the disadvantages of IoT?**

The disadvantages of IoT are:

* **Security:** IoT technology creates an ecosystem of connected devices. However, during this process, the system may offer little authentication control despite sufficient cyber security measures.
* **Privacy:** The use of IoT, exposes a substantial amount of personal data, in extreme detail, without the user's active participation. This creates lots of privacy issues.
* **Flexibility:** There is a huge concern regarding the flexibility of an IoT system. It is mainly regarding integrating with another system as there are many diverse systems involved in the process.
* **Complexity:** The design of the IoT system is also quite complicated. Moreover, it's deployment and maintenance also not very easy.
* **Compliance:** IoT has its own set of rules and regulations. However, because of its complexity, the task of compliance is quite challenging.

**8) Define Arduino**

Arduino is a free electronics platform having easy to use hardware and software. It has a microcontroller capable of reading input from sensors to control the motors programmatically.

**9) List mostly used sensors types in IoT**

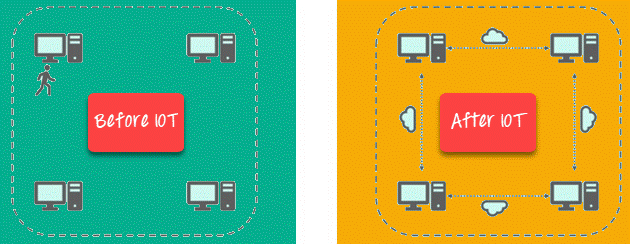
Mostly used sensor types in IoT are:

* Smoke sensor
* Temperature sensors
* Pressure sensor
* Motion detection sensors
* Gas sensor
* Proximity sensor
* IR sensors

**10) Mention the basic difference between IoT and sensor businesses?**

A sensor business does not need an active internet connection to work. Internet of Things requires a control side to work.

**11) What are the advantages of IoT?**

[](https://www.guru99.com/images/2/iot-interview-questions-2.png)

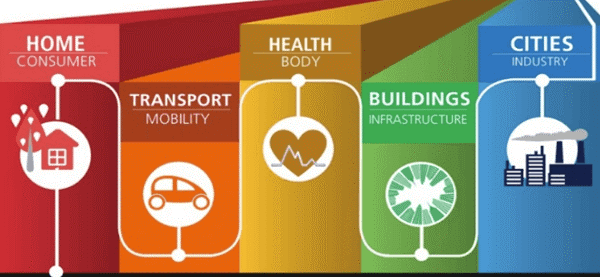
Key benefits of IoT technology are as follows:

* **Technical Optimization:** IoT technology helps a lot in improving techniques and making them better. For example, with IoT, a manufacturer is able to collect data from various car sensors. The manufacturer analyses them to improve its design and make them more efficient.
* **Improved Data Collection:** Traditional data collection has its limitations and its design for passive use. IoT facilitates immediate action on data.
* **Reduced Waste:** IoT offers real-time information leading to effective decision making & management of resources. For example, if a manufacturer finds an issue in multiple car engines, he can track the manufacturing plan of those engines and solves this issue with the manufacturing belt.
* **Improved Customer Engagement:** IoT allows you to improve customer experience by detecting problems and improving the process.

**12) What is Bluegiga APX4 protocol?**

The Bluegiga APX4 is a solution that supports both the WiFi and BLE platform, and it is based on a 450MHz ARM9 processor.

**13) What are the most common IoT applications?**

[](https://www.guru99.com/images/2/iot-interview-questions-3.png)

The most common IoT applications are:

* **Smart Thermostats:** Helps you to save resources on heating bills by knowing your usage patterns.
* **Connected Cars:** IoT helps automobile companies handle billing, parking, insurance, and other related stuff automatically.
* **Activity Trackers:** Helps you to capture heart rate patterns, calorie expenditure, activity levels, and skin temperature on your wrist.
* **Smart Outlets:** Remotely turn any device on or off. It also allows you to track a device's energy level and get custom notifications directly into your smart phone.
* **Parking Sensors:** IoT technology helps users to identify the real-time availability of parking spaces on their phones.
* **Connect Health:** The concept of a connected healthcare system facilitates real-time health monitoring and patient care. It helps in improved medical decision-making based on patient data.

**14) What is Pulse Width Modulation?**

PWM or Pulse Width Modulation is a variation of how much time the signal is high in an analog fashion. The signal can be high or low and the user can even change the proportion of the time.

**15) Mention applications of PWM in IoT**

Applications of PWM in IoT are controlling the speed of DC motor, controlling the direction of a servo moto, Dimming LED, etc.

**16) List available wireless communications boards available in Raspberry Pi?**

Wireless communications boards available in Raspberry Pi are 1) WiFi and 2) BLE/Bluetooth.

**17) What are the functions used to read analog and digital data from a sensor in Arduino?**

Functions used to read analog and digital data from a sensor in Arduino are: digitalRead() and digitalWrite().

**18) What is Bluetooth Low Energy?**

Bluetooth Low Energy is a wireless PAN (Personal Area Network) technology. It uses less power to transmit long-distance over a short distance.

**19) Define Micro Python**

MicroPython is a Python implementation, which includes a small subset of its standard library. It can be optimized to run on the ModeMCU microcontroller.

**20) List available models in Raspberry Pi**

Models of Raspberry Pi are:

* Raspberry Pi 1 Model B
* Raspberry Pi 1 Model B+
* Raspberry Pi 1 Model A
* Raspberry Pi Zero
* Raspberry Pi 3 Model B
* Raspberry Pi 1model A+
* Raspberry Pi Zero W
* Raspberry Pi 2

**21) What are the challenges of IoT?**

Important challenges of IoT are:

* Insufficient testing and updating.
* Concern regarding data security and privacy.
* Software complexity.
* Data volumes and interpretation.
* Integration with AI and automation.
* Devices require a constant power supply which is difficult.
* Interaction and short-range communication.

**22) Mention some of the commonly used water sensors**

The commonly used water sensors are:

* Turbidity sensor.
* Total organic carbon sensor.
* pH sensor.
* Conductivity sensor.

**23) Differentiate between Arduino and Raspberry pi**

|  |  |
| --- | --- |
| **Arduino** | **Raspberry pi** |
| Arduino is an open, programmable USB microcontroller. | It can execute one program at a time. |
| Raspberry pi is a credit card size computer. | Users can run more than one program at a time. |

**24) What are mostly used IoT protocols?**

The mostly used IoT protocols are:

* XMPP
* AMQP
* Very Simple Control Protocol (VSCP)
* Data Distribution Service (DDS)
* MQTT protocol
* WiFi
* Simple Text Oriented Messaging Protocol(STOMP)
* Zigbee

**25) What are IoT publishers?**

IoT Publishers are sensors that send real-time data to intermediate devices or middleware.

**26) What is a library in Arduino?**

Arduino library is a collection of code that is already written for controlling module or sensor.

**27) Mention some of the wearable Arduino boards**

Wearable Arduino boards are:

* Lilypad Arduino main board.
* Lilypad Arduino simple.
* Lilypad Arduino simple snap.
* Lilypad Arduino USB.

**28) What is replication?**

Replication is the act of syncing data between two or more servers.

**29) What is IoT Thingworx?**

Thingworx is a platform for the fast development and deployment of connected devices. It is a collection of integrated IoT development tools that support analysis, production, property, and alternative aspects of IoT development.

**30) What is Salesforce IoT Cloud?**

The Salesforce IoT Cloud is an online platform for storing and processing IoT information. It is an assortment of various application development elements, which are called lightning. This program gathers information from websites, devices, customers, and partners. It then triggers actions for period responses.

**31) Explain IoT GE-PREDIX**

GE or General Electric Predix is the software for the information assortment from industrial instruments. It offers a PaaS which allows users performance management and operation optimization facility. It connects instrumentation, people, and information in an exceedingly conventional technique.

**32) List out Some popular companies is working on IoT.**

Popular companies working on IoT are: 1) Philips, 2) LG, 3) Google, 4) Apple and 5) Samsung.

**33) What are various types are of CAN Frame?**

Various types of CAN frames are:

1) Data frame.

2) Request frame.

3) Error frame.

4) Overload frame.

**34) What is the main difference between floating CPU and fixed-point CPU?**

Floating CPU can take floating value directly, whereas fixed CPU is converted to integer format. Thereby, it leads to the loss of some resolution.

**35) Define GPIO.**

GPIO is a programmable pin that can be used to control input or output pins programmatically.

**36) Explain Android things.**

Android things is an Android-based OS that is built for embedded devices.

**37) What is the aim of airflow sensors?**

The main aim of airflow sensors is to measure the air level in the soil. This sensor enables one to measure it dynamically, from one location, or multiple locations of the garden.

**38) Mention suitable databases for IoT.**

Suitable databases for IoT are:

* influx DB
* Apache Cassandra
* RethinkDB
* MongoDB
* Sqlite

**39) Why use the scheduler in RTOS?**

Scheduler in RTOS is used for switching one task to another.

**40) Mention real-time usage of Raspberry pi.**

* Home automation
* Portable web server
* manipulating the robots
* Internet radio

**41) Define IoT Contiki.**

IoT Contiki is software that targets explicitly little devices connected with the Internet. It is used with process power bandwidth, power, and restricted memory. Contiki helps for the management of programs, resources, processes, communication, and memory.

**42) What is data in IoT?**

Data in IoT refers to the information that is collected by the installed devices at any building.

**43) List majorly used IoT controllers by industries.**

Majorly used IoT controllers by industries are: 1) Siemens IoT 2020 and 2) Arduino.

**44) What is a crystal oscillator?**

A crystal oscillator is the main part of the microprocessor. It executes every single pulse one instruction in CPU.

**45) What is the importance of the Internet of Everything?**

Internet of Everything is important because:

* It brings together people, processes, things, and data to make network connections valuable and relevant.
* It converts the information into actions to create new capabilities and opportunities for businesses.

**46) What is WSN?**

The full form of WSN is Wireless Sensor Network. It is a network of notes, design to observe and to study physical parameters of the application.

**47) What is Zigbee?**

Zigbee is the same like Bluetooth. It used in a complex system for low power operation, robustness, and high security.

**48) What is Z-Wave?**

Z-Wave is an IoT technology that uses low power RF communication. It is designed for home automation products like lamp controllers and sensors.

**49) How to install a new library in Arduino?**

A new library in Arduino can be installed by selecting the library from the sketch option in Toolbar.

**50) What is MQTT?**

The full form of MQTT is Message Queue Telemetry Transport Protocol. It is a messaging protocol that is used for tracking devices in IoT.

**51) Name some important IoT hardware**

IoT hardware includes varieties of devices like router, bridge, sensor, etc.

**52) What are the operating systems supported by Pi?**

Operating systems supported by Pi are:

* Raspbian
* Open ELEC (Open Embedded Linux Entertainment center)
* RISC OS
* Lakka
* OSMC (Open Source Media Centre)
* Windows IoT Core

**53) How to reduce the size of the sketch?**

Reducing the size of the sketch is can be reduced by removing unwanted libraries from the code and make code short and simple.

**54) What are the various types of antennas designed for IoT devices?**

Various types of antennas designed for IoT devices are:

* Chip Antenna
* PCB Antenna
* Wire Antenna
* Proprietary Antenna
* Whip Antenna

**55) What is the difference between M2M and IoT?**

The difference between M2M and IoT is:

|  |  |
| --- | --- |
| **M2M** | **IoT** |
| Communication is done within embedded software at the client site. | Communication is done for grand-scale projects. |
| It uses isolated systems of devices having the same standards. | It uses integrated devices, applications, and data across varying standards. |
| M2M offers limited scalability options. | IoT is inherently more scalable. |
| A cellular network or wired network is used for device connectivity. | It uses an active Internet connection for device connectivity. |
| Machines can communicate with one machine at a time. | Many machines can communicate with each other over the Internet. |

**56) What are the features of influxDB?**

Features of influxDB are:

* Provides support of visualization tools.
* Works with distributed time-series database.
* It does not have any external dependencies.

**57) How to program Arduino?**

Programmers can use the Arduino IDE in order to write an Arduino program. Developers can also use Node.js Johny five-module in order to control Arduino.

**58) What are IoT testing tools?**

IoT testing tools can be divided into hardware and software:

* **IoT testing software:**Tcpdump and Wireshark.
* **Hardware for IoT testing:** JTAG Dongle, Digital Storage Oscilloscope, and Software Defined Radio.

**59) How to store the high-volume file into Arduino?**

A specification called Gridfs can be used for storing high volume file into Arduino.

**60) Mention IoT softwares.**

IoT softwares are: 1) Blockchain, 2) windows IoT, 3) Predix, 4) Microsoft Azure, 5) Bluemix, and 6) Node-RED.

**61) What is Shodan?**

Shodan is an IOT testing tool that can be used to discover which of your devices are connected to the Internet. It allows you to keep track of all the computers which are directly accessible from the Internet.

**62) What is a thing in IoT?**

IOT thing is an item having an embedded and connected computing device.

**63) What is Thermocouple?**

A Thermocouple is a device which consists of two different conductors joined together to form an electrical junction.

**64) Mention some examples of MEMS sensor.**

* MPU6050- Gyroscope
* ADXL345
* piezoelectric sensor
* Accelerometer

**65) What are IoT test approaches?**

IoT test approaches are: 1) Usability, 2) IoT Security, 3) Connectivity, 4) Performance, 5) Compatibility Testing, 6) Pilot Testing, 7) Regulatory Testing, and 8) Upgrade testing.

**66) What is sharding?**

Sharding is a method to split data into collections and stored in machines.

**67) List hardware prototypes used in IoT**

Hardware Prototypes used in IoT are 1) Raspberry Pi, 2) ARM Cortex Family, and 3) Arduino.

**68) What is IoT Testing?**

IoT testing is a type of testing to check IoT devices. Today there is an increasing need to deliver better and faster services. There is a huge demand to access, create, use, and share data from any device. The thrust is to provide greater insight and control over various interconnected IoT devices. Hence, the IoT testing framework is important.

**69) What are the types of IoT?**

There are two types of IoT:

* **Internet of Things:** It creates a business that uses gadgets to perform a task.
* **Industrial Internet of Things:** It creates business in the industry like agriculture.

**70) What is Thingful?**

Thingful is a search engine for the Internet of Things. It allows secure interoperability between millions of IoT objects via the Internet. This IOT testing tool also controls how data is used and empowers to take more decisive and valuable decisions.

**71) What are interrupts in Arduino?**

Interrupts enable specific tasks to process in the background and are enabled by default. Its main job is to ensure the device processor responds fastly to essential events.

**72) What is Asset Tracking?**

Asset Tracking or Asset management is the process of keeping track of physical assets and information.

**73) What are the risks associated with the IOE Internet of Everything?**

Risks associated with IOE are 1) Privacy, 2) Security, 3) Network congestion, and 4) Electricity consumption at the peaks.

**74) What is the basic difference between the IoT network and Wireless Sensor Network?**

Wireless Sensor Network things connected to the wireless network and gather some monitoring environment or data. IoT contains a combination of:

* WSN
* Internet
* Cloud Storage
* web or mobile application

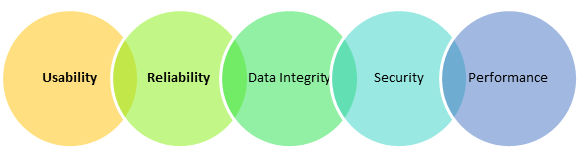
**75) What is the importance of the network in IoT?**

The network is the main part of the IoT. It is responsible for providing a practical and smart system that makes strong infrastructure. The network offers scalability to help devices coordinate with other lines with the Internet.

**76) What is the connection between IoT and sensors in the commercial enterprise?**

Sensors may be used in devices that are not net-connected, while devices need to be connected to the Net with IoT. Yet, sensing is a part of IoT, even if the device is not connected to the Net.

**77) Explain the types of testing in IoT?**

[](https://www.guru99.com/images/1/041318_0543_InternetofT1.png)

IoT devices testing types are:

* **Usability Testing:**There are so many devices of different shape and form factors are used by the users. Moreover, the perception also varies from one user to others. That's why checking the usability of the system is very important in IoT testing.
* **Compatibility Testing:**There are lots of devices that can be connected through the IoT system. These devices have varied software and hardware configuration. Therefore, a possible combination is huge. As a result, checking the compatibility in the IoT system is important.
* **Reliability and Scalability Testing:**Reliability and Scalability is important for building an IoT test environment which involves a simulation of sensors by utilizing virtualization tools and technologies.
* **Data Integrity Testing:**It's important to check the Data integrity in IoT testing as it requires a large amount of data and its application.
* **Security testing:**In the IoT environment, many users are accessing a massive amount of data. Thus, it is important to validate user via authentication, have data privacy controls as part of security testing.
* **Performance Testing:**Performance testing is important to create a strategic approach for developing and implementing an IoT testing plan.

**References:**

**Text Books:**

1. Vijay Madisetti, ArshdeepBahga, “Internet of Things-A Hands on Approach”, University Press, 2015, ISBN: 9788173719547.
2. Dr. SRN Reddy, RachitThukral and Manasi Mishra, “Introduction to Internet of Things: A practical Approach”, ETI Labs.
3. Pethuru Raj and Anupama C. Raman, “The Internet of Things: Enabling Technologies, Platforms, and Use Cases”, CRC Press.

**Reference Books:**

1. Matt Richardson & Shawn Wallace, “Getting Started with Raspberry Pi”, O'Reilly (SPD), 2014, ISBN: 9789350239759.
2. Jeeva Jose, “Internet of Things”, Khanna Publishing House, Delhi.
3. Adrian McEwen, “Designing the Internet of Things”, Wiley.
4. Raj Kamal, “Internet of Things: Architecture and Design”, McGraw Hill.
5. Cuno Pfister, “Getting Started with the Internet of Things”, O Reilly Media.