SRM VALLIAMMAI ENGINEERING COLLEGE

(An Autonomous Institution)

SRM Nagar, Kattankulathur-603203

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

ACADEMIC YEAR: 2025-2026

ODD SEMESTER

LAB MANUAL

(REGULATION - 2023)

AD3564 – EMBEDDED SYSTEMS AND IoT LABORATORY

FIFTH SEMESTER

B.Tech – Artificial Intelligence and Data Science

Prepared By

R. VAISHNAVI , A.P (O.G) / AI&DS

AD3564 EMBEDDED SYSTEMS AND IoT LABORATORY LT P C

0 0 3 1.5

OBJECTIVES:

- To learn Assembly and Embedded C languages and perform basic operations.
- To learn the Arduino programming.
- To perform different communication methods with IoT devices.
- To learn Raspberry PI platform and python programming.
- To build a small low-cost embedded and IoT system using Arduino/Raspberry Pi/ open platform.

LIST OF EXPERIMENTS:

1. Write 8051 Assembly Language experiments using simulator.
2. Test data transfer between registers and memory.
3. Perform ALU operations.
4. Write Basic and arithmetic Programs Using Embedded C.
5. Introduction to Arduino platform and programming.
6. Explore different communication methods with IoT devices (Zigbee, GSM, Bluetooth).
7. Introduction to Raspberry PI platform and python programming.
8. Interfacing sensors with Raspberry PI.
9. Communicate between Arduino and Raspberry PI using any wireless medium.
10. Setup a cloud platform to log the data.
11. Log Data using Raspberry PI and upload to the cloud platform.
12. Design an IOT based system.

TOTAL: 45 PERIODS

LIST OF EQUIPMENTS: (30 Students per Batch)

8051 Simulator, Arduino Kit (IDE, UNO Board, USB Cable, Joystick Module, Zigbee Module, Bluetooth Module), Raspberry Pi Kit, Thonny IDE, IR Sensors and Ultra sensors

OUTCOMES:

At the end of this course, the students will be able to:

- Write embedded C programs.
- Design simple embedded applications.

- Compare the communication models in IoT.
- Write Python code for simple applications using Raspberry PI.
- Design IoT applications using Arduino/Raspberry Pi /open platform.

CO - PO - PSO MAPPING:

CO	PO												PSO			
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**	**3**	**4**
AD3564.1	2	2	-	-	-	-	-	-	-	-	-	-	3	-	1	-
AD3564.2	3	3	3	-	-	-	-	-	-	-	-	-	1	-	-	-
AD3564.3	2	2	2	-	-	-	-	-	-	-	-	-	3	-	-	2
AD3564.4	2	3	3	-	-	-	-	-	-	-	-	-	-	-	2	2
AD3564.5	3	-	3	-	-	-	-	-	-	-	-	-	2	-	3	-
Average	**2.4**	**2.5**	**2.5**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**2.0**	**-**	**2.0**	**2.0**

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. To afford the necessary background in the field of Artificial Intelligence and data Science to deal with engineering problems to excel as engineering professionals in industries.
2. To improve the qualities like creativity, leadership, teamwork and skill thus contributing towards the growth and development of society.
3. To develop ability among students towards innovation and entrepreneurship that caters to the needs of Industry and society.
4. To inculcate and attitude for life-long learning process through the use of Artificial Intelligence and Data Science sources.
5. To prepare then to be innovative and ethical leaders, both in their chosen profession and in other activities.

PROGRAMME OUTCOMES (POs)

After going through the four years of study, Bachelor of Technology in Artificial Intelligence and Data Science Graduates will exhibit ability to:

PO#	Graduate Attribute	Programme Outcomes
1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems.
2	Problem analysis	Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	Design/development of solutions	Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.
4	Conduct investigations of complex problems	Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5	Modern tool usage	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modelling to complex engineering activities, with an understanding of the limitations.
6	The engineer and society	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7	Environment and sustainability	Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings
10	Communication	Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions
11	Project management and finance	Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments
12	Life-long learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

PROGRAM SPECIFIC OUTCOMES (PSOs)

After the completion of Bachelor of Technology in Artificial Intelligence and Data Science programme the student will have following Program specific outcomes

1. Design and develop secured database applications with data analytical approaches of data preprocessing, optimization, visualization techniques and maintenance using state of the art methodologies based on ethical values.
2. Design and develop intelligent systems using computational principles, methods and systems for extracting knowledge from data to solve real time problems using advanced technologies and tools.
3. Design, plan and setting up the network that is helpful for contemporary business environments using latest software and hardware.
4. Planning and defining test activities by preparing test cases that can predict and correct errors ensuring a socially transformed product catering all technological needs.

COURSE OUTCOMES:

Course Name: AD3564 - EMBEDDED SYSTEMS AND IoT LABORATORY

Year of study: 2025 – 2026 (ODD SEM)

AD3564.1 Write embedded C programs.

AD3564.2 Design simple embedded applications.

AD3564.3 Compare the communication models in IoT.

AD3564.4 Write Python code for simple applications using Raspberry PI.

AD3564.5 Design IoT applications using Arduino/Raspberry Pi /open platform.

CO- PO MATRIX:

CO	PO											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
AD3564.1	2	2	-	-	-	-	-	-	-	-	-	-
AD3564.2	3	3	3	-	-	-	-	-	-	-	-	-
AD3564.3	2	2	2	-	-	-	-	-	-	-	-	-
AD3564.4	2	3	3	-	-	-	-	-	-	-	-	-
AD3564.5	3	-	3	-	-	-	-	-	-	-	-	-

CO -PO AVERAGE:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
AD3564	**2.4**	**2.5**	**2.5**	-	-	-	-	-	-	-	-	-

CO - PSO MATRIX:

CO	PSO1	PSO2	PSO3	PSO4
AD3564.1	3	-	1	-
AD3564.2	1	-	-	-
AD3564.3	3	-	-	2
AD3564.4	-	-	2	2
AD3564.5	2	-	3	-

CO - PSO AVERAGE:

CO	PSO1	PSO2	PSO3	PSO4
AD3564	**2.0**	**-**	**2.0**	**2.0**

EVALUATION PROCEDURE FOR EACH EXPERIMENT

S.No	Description	Mark
1.	Aim & Pre-Lab discussion	20
2.	Observation	30
3.	Conduction and Execution	30
4.	Output & Result	10
5.	Viva	10
Total		**100**

INTERNAL ASSESSMENT FOR LABORATORY

S.No	Description	Mark
1.	Conduction & Execution of Experiment	25
2.	Record	10
3.	Model Test	15
Total		**50**

TABLE OF CONTENTS

EXP NO:1

DATE:

Write 8051 Assembly Language experiments using simulator.

AIM:

To write 8051 assembly language experiments using simulator.

PROCEDURE:

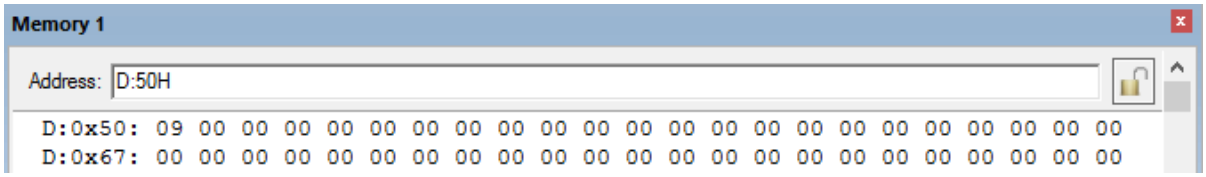
1. Create a new project, go to “Project” and close the current project “Close Project”.
2. Next Go to the Project New µVision Project and Create New Project Select Device for Target.
3. Select the device DS2250 or DS2251T or DS2252T.
4. Add Startup file Next go to “File” and click “New”.
5. Write a program on the editor window and save it with .asm extension.
6. Add this source file to Group and click on “Build Target” or F7.
7. Go to debugging mode to see the result of simulation by clicking Run or step run.
8. This program follows the above-mentioned steps to create and compile the project.
9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM 1:

//ADDITION

```
MOV A,#5
MOV R0,#4
ADD A,R0
MOV 50H,A
END
```

OUTPUT:

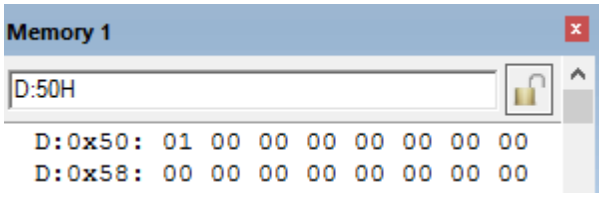


PROGRAM 2:

//SUBTRACTION

```
MOV A,#7
MOV R0,#6
SUBB A,R0
MOV 50H,A
END
```

OUTPUT:



RESULT:

Thus, the assembly language program for 8051 simulators has been executed successfully.

EXP NO: 2(a) **Test data transfer between registers to memory.**

DATE:

AIM:

To write and execute an Assembly language program to transfer data between registers and memory.

PROCEDURE:

1. Create a new project, go to “Project” and close the current project “Close Project”.

2. Next Go to the Project New µVision Project and Create a New Project Select Device for the Target.

3. Select the device DS2250 or DS2251T or DS2252T.

4. Add Startup file Next go to “File” and click “New”.

5. Write a program on the editor window and save it with .asm extension.

6. Add this source file to Group and click on “Build Target” or F7.

7. Go to debugging mode to see the result of the simulation by clicking Run or Step run.

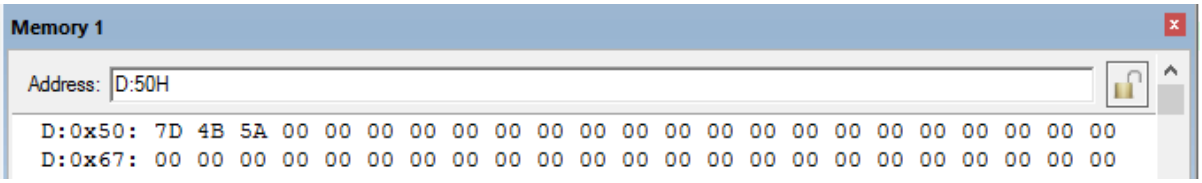
8. This program follows the above-mentioned steps to create and compile the project.

9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM:

```
MOV R0,#7DH

MOV R1,#4BH

MOV R2,#5AH

MOV 50H,R0

MOV 51H,R1

MOV 52H,R2

END
```

OUTPUT:



RESULT:

Thus, the assembly language program to Test data transfer between registers to memory.

EXP NO:2(b) **Test data transfer between memory to register.**

DATE:

AIM:

To write an assembly language program to test data transfer between memory and register.

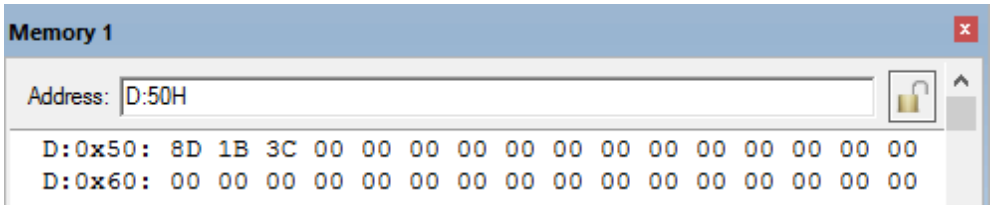
PROCEDURE:

1. Create a new project, go to “Project” and close the current project “Close Project”.
2. Next Go to the Project New μVision Project and Create a New Project Select Device for the Target.
3. Select the device DS2250 or DS2251T or DS2252T.
4. Add Startup file Next go to “File” and click “New”.
5. Write a program on the editor window and save it with .asm extension.
6. Add this source file to Group and click on “Build Target” or F7.
7. Go to debugging mode to see the result of the simulation by clicking Run or Step run.
8. This program follows the above-mentioned steps to create and compile the project.
9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM:

```
MOV 50H,#8DH

MOV 51H,#1BH

MOV 52H,#3CH

MOV R0,50H MOV

R1,51H MOV

R2,52H END
```

OUTPUT:



RESULT:

Thus, the assembly language program executes the data transfer between memory to register.

EXP NO:3

DATE:

Perform ALU operations.

AIM:

To write an assembly language program to perform ALU operations.

PROCEDURE:

1. Create a new project, go to “Project” and close the current project “Close Project”.

2. Next Go to the Project New μVision Project and Create a New Project Select Device for the Target.

3. Select the device DS2250 or DS2251T or DS2252T.

4. Add Startup file Next go to “File” and click “New”.

5. Write a program on the editor window and save it with .asm extension.

6. Add this source file to Group and click on “Build Target” or F7.

7. Go to debugging mode to see the result of the simulation by clicking Run or Step run.

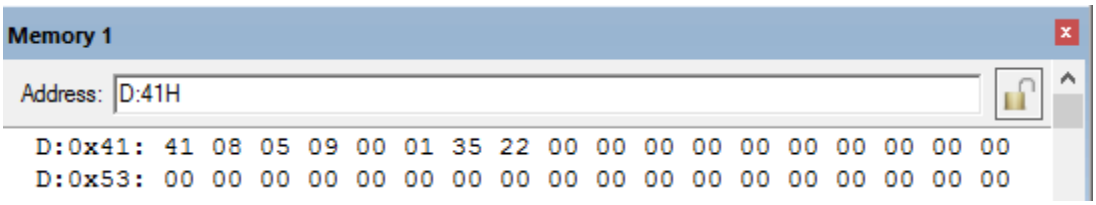
8. This program follows the above-mentioned steps to create and compile the project.

9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM:

```
//ADDITION
    MOV A,#20H
    ADD A,#21H
    MOV 41H,A
//SUBTRACTION
    MOV  A,#20H
    SUBB A,#18H
    MOV 42H,A
//MULTIPLICATION
    MOV A,#5H
    MOV B,#1H
    MUL AB
    MOV 43H,A
//DIVISION
    MOV A,#90H
    MOV B,#10H
    DIV AB
    MOV 44H,A
    MOV 45H,B
//AND
    MOV A,#5H
    MOV B,#1H
    ANL A,B
    MOV 46H,A
//OR
    MOV A,#25H
    MOV B,#15H
    ORL A,B
    MOV 47H,A
//XOR
    MOV A,#45H
    MOV B,#67H
    XRL A,B
    MOV 48H,A
    END
```

OUTPUT:



RESULT:

Thus, the assembly language program to perform an ALU operations has been executed successfully.

EXP NO:4(a)

Write Basic Programs Using Embedded C.

DATE:

AIM:

To write basic program using embedded c.

PROCEDURE:

1. Create a new project, go to “Project” and close the current project “Close Project”.

2. Next Go to the Project New μVision Project and Create a New Project Select Device for the Target.

3. Select the device DS2250 or DS2251T or DS2252T.

4. Add Startup file Next go to “File” and click “New”.

5. Write a program on the editor window and save it with .asm extension.

6. Add this source file to Group and click on “Build Target” or F7.

7. Go to debugging mode to see the result of the simulation by clicking Run or Step run.

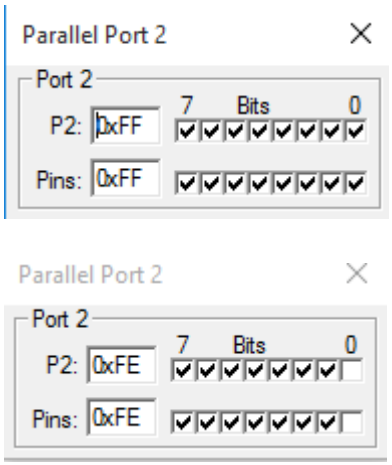
8. This program follows the above-mentioned steps to create and compile the project.

9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM:

```
#include<REG51.h> int
i,j;
sbit LED = P2^0; void
main()
{
while(1)
{
LED = 0;
for(j=0;j<10000;j++);
LED = 1;
for(j=0;j<10000;j++);
}
}
```

OUTPUT:



Note: Output Port 2 blinked when the program started running.

RESULT:

Thus, the basic embedded c program has been executed successfully.

EXP NO:4(b)

Write Arithmetic Program Using Embedded C.

DATE:

AIM:

To write arithmetic program using embedded c.

PROCEDURE:

1. Create a new project, go to “Project” and close the current project “Close Project”.

2. Next Go to the Project New μVision Project and Create a New Project Select Device for the Target.

3. Select the device DS2250 or DS2251T or DS2252T.

4. Add Start up file Next go to “File” and click “New”.

5. Write a program on the editor window and save it with .asm extension.

6. Add this source file to Group and click on “Build Target” or F7.

7. Go to debugging mode to see the result of the simulation by clicking Run or Step run.

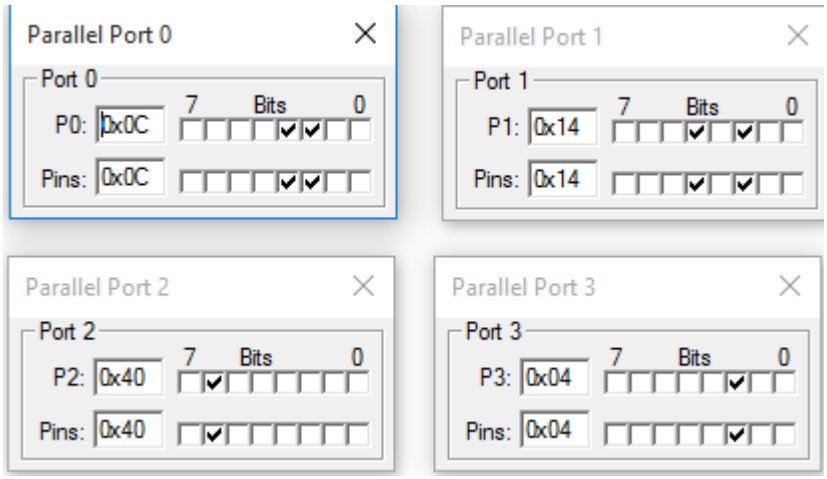
8. This program follows the above-mentioned steps to create and compile the project.

9. To see the output go to the memory window and enter the memory location like D:50H.

PROGRAM:

```
#include<REG51.H>
unsigned char a, b; void
main()
{ a=0x10;
b=0x04;
P0=a-b;
P1=a+b;
P2=a*b;
P3=a/b;
while(1);
}
```

OUTPUT:



RESULT:

Thus, the arithmetic program using embedded c has been executed successfully.

EXP NO:5 **Introduction to Arduino platform and programming**

DATE:

AIM:

To write a study of Arduino platform and programming installation steps

INTRODUCTION:

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. The word "uno" means "one" in Italian and was chosen to mark the initial release of Arduino Software.

FEATURES OF THE ARDUINO :

1. 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.

2. The board functions can be controlled by sending a set of instructions to the microcontroller on the board via Arduino IDE.

3. Arduino IDE uses a simplified version of C++, making it easier to learn to program.

4. Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Arduino IDE
(Integrated Development Environment)

Introduction:

The Arduino Software (IDE) is easy-to-use and is based on the Processing programming environment. The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

The Arduino Software (IDE) – contains:

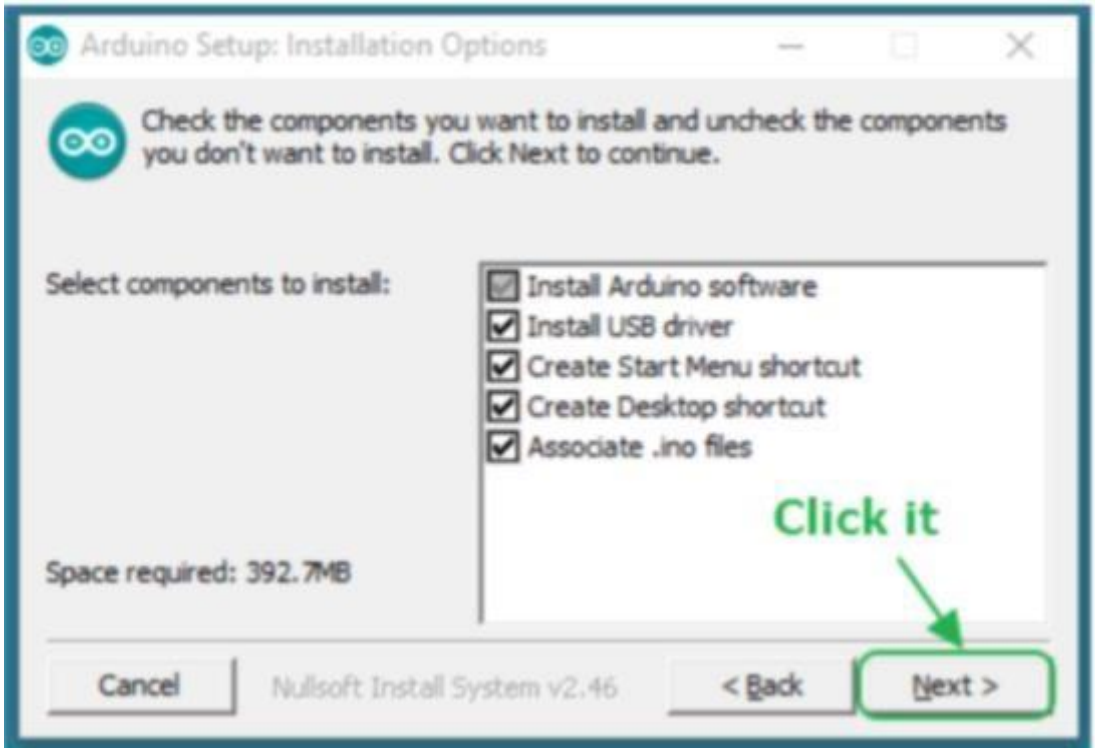
- A text editor for writing code
- A message area
- A text consoles
- A toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

Installation of Arduino Software (IDE)

Step1: Downloading

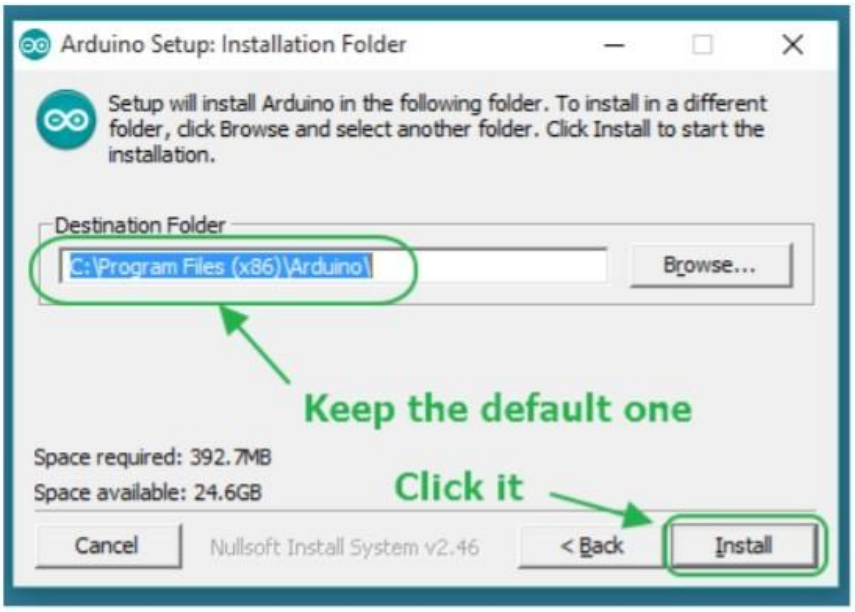
- To install the Arduino software, download this page:

http://arduino.cc/en/Main/Software and proceed with the installation by allowing the driver installation process.



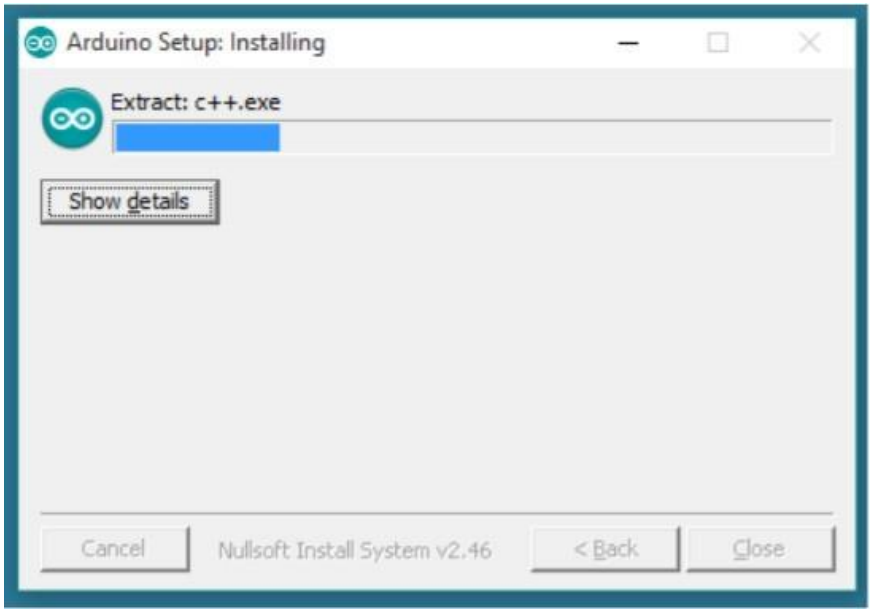
Step 2: Directory Installation

Choose the installation directory.



Step 3: Extraction of Files

- The process will extract and install all the required files to execute properly the Arduino Software (IDE)



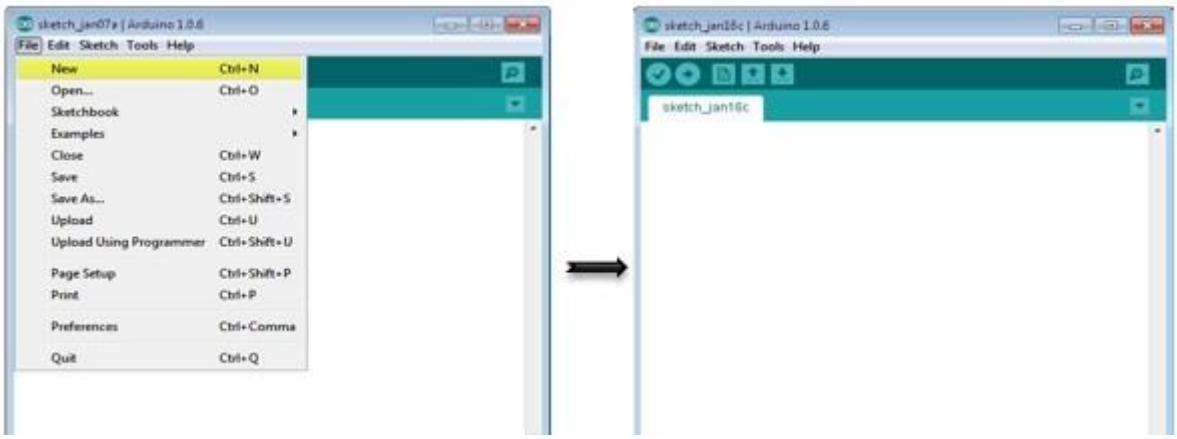
Step 4: Connecting the board

- The USB connection with the PC is necessary to program the board and not

just to power it up. The Uno and Mega automatically draw power from either the USB or an external power supply. Connect the board to the computer using the USB cable. The green power LED (labelled PWR) should go on.

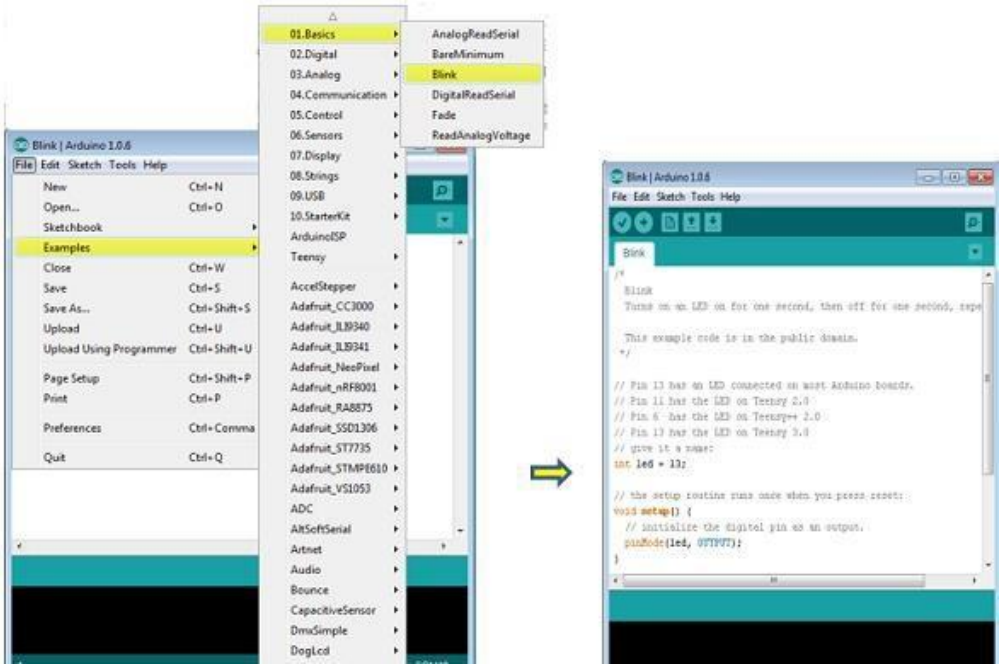
Step 5: Working on the new project

- Open the Arduino IDE software on your computer. Coding in the Arduino language will control your circuit.
- Open a new sketch File by clicking on New



Step 6: Working on an existing project

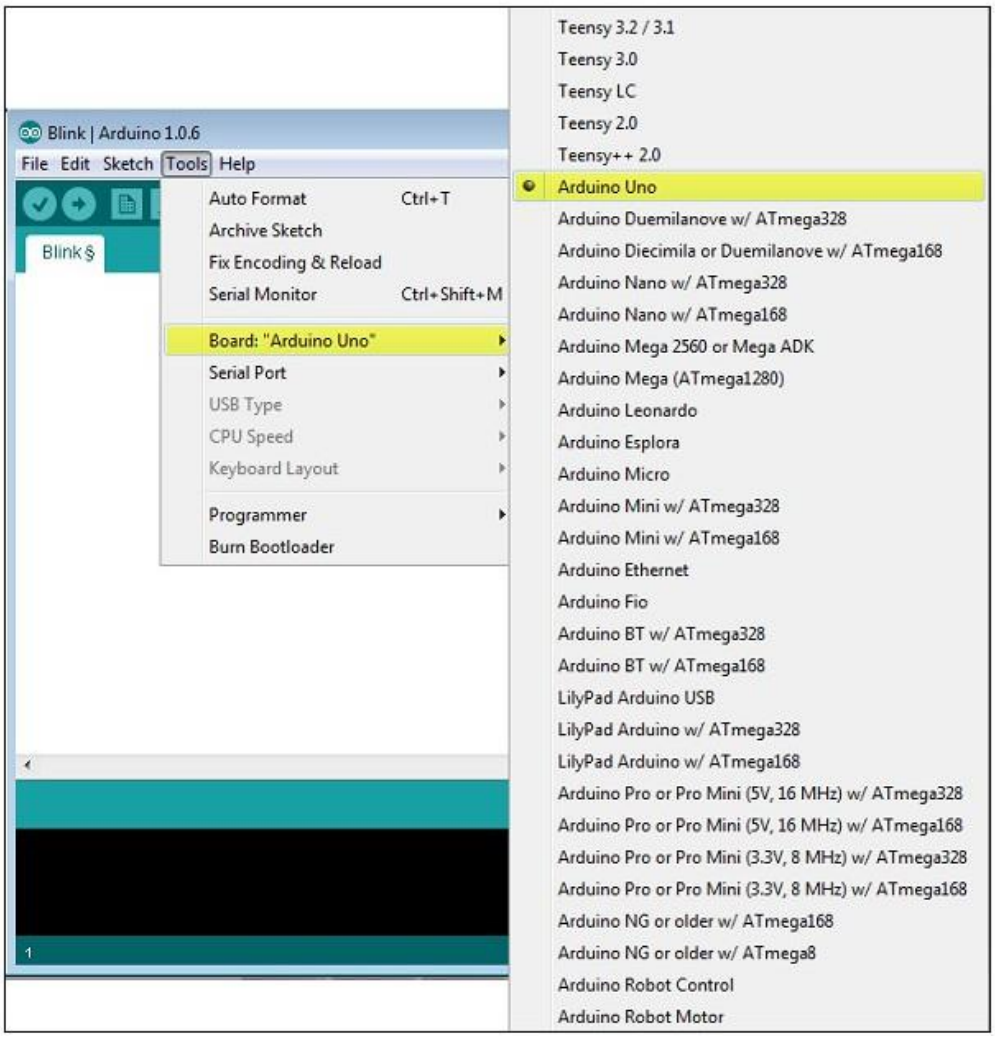
➢ To open an existing project example, select File → Example → Basics → Blink.



Step 7: Select your Arduino board.

➢ To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

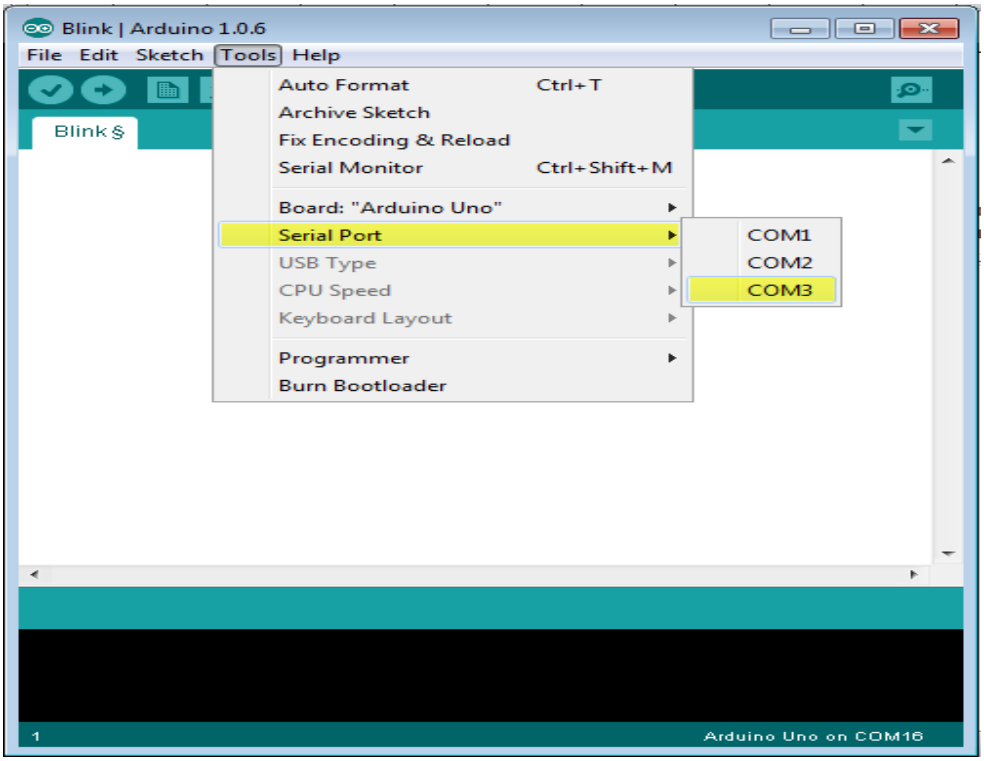
Go to Tools → Board and select your board.



Step 8: Select your 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.



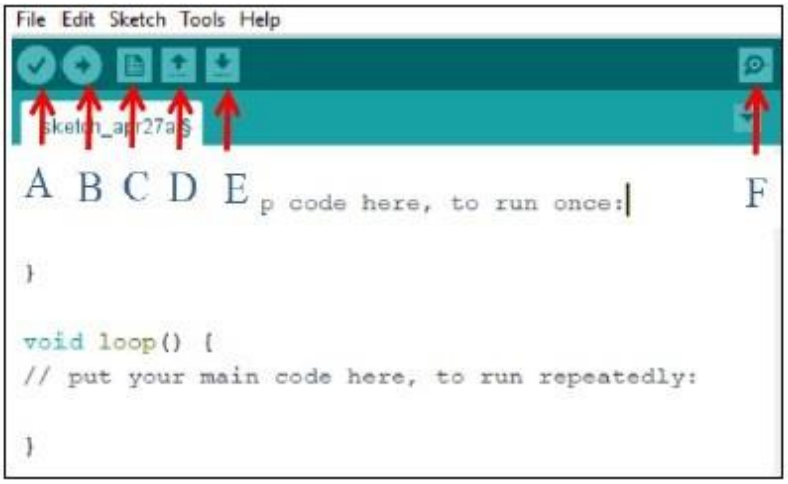
Step 9: Upload the program to your board.

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.

A	Verify
B	Upload
C	New
D	Open
E	Save
F	Serial Motor



RESULT:

Thus the study of arduino platform and programming installation steps and working on Arduino has been successfully executed.

EXP NO: 6

DATE

Different communication methods with IoT devices (Zigbee, GSM, Bluetooth)

AIM:

To Explore different communication methods with IoT devices (Zigbee, GSM, Bluetooth).

DIFFERENT COMMUNICATION METHODS:

IoT devices require reliable and efficient communication methods to transmit data and interact with other devices or systems. Here are three commonly used communication methods for IoT devices:

Zigbee:

Zigbee is a low-power wireless communication protocol designed for short-range communication between devices. It operates on the 2.4 GHz frequency band and supports mesh networking, allowing devices to communicate with each other through intermediate nodes. Zigbee is commonly used in home automation, industrial control, and smart energy applications.

GSM (Global System for Mobile Communications):

GSM is a widely used cellular network technology that enables IoT devices to connect to the internet using SIM cards. It operates on various frequency bands and provides wide coverage, making it suitable for applications that require long-range communication. GSM is commonly used in applications such as asset tracking, remote monitoring, and smart cities.

Bluetooth:

Bluetooth is a short-range wireless communication technology that operates on the 2.4 GHz frequency band. It is commonly used for connecting IoT devices to smartphones, tablets, and other nearby devices. Bluetooth Low Energy (BLE) is a power-efficient version of Bluetooth that is ideal for battery-powered IoT devices. Bluetooth is widely used in applications such as wearable devices, healthcare monitoring, and home automation.

Each communication method has its advantages and limitations, and the choice depends on the specific requirements of the IoT application. Factors to consider include range, power consumption, data rate, security, and interoperability with other devices or systems.

RESULT:

Thus different communication methods with IoT devices like Zigbee, GSM, Bluetooth has been explored successfully.

EXP NO: 6(a)

DATE

BLUETOOTH COMMUNICATION

AIM:

To write a program to control an LED using a Bluetooth module.

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Arduino IDE 2.0	1
2	Arduino UNO Development Board	1
3	Jumper Wires	few
4	Arduino USB Cable	1
5	HC-05 Bluetooth Module	1

PROCEDURE

1. Install Arduino IDE 2.0
2. Write Arduino Code
3. Connect Arduino UNO to PC via USB cable

4. Select Board & Port

- Board: Arduino UNO
- Port: Select the correct COM port from Tools > Port

5. Upload the Program

- Disconnect HC-05 before uploading (to avoid serial conflict)
- Upload the code
- Reconnect the HC-05 module after upload

6. Install a Bluetooth Terminal App on your smartphone (e.g., "Bluetooth Terminal" or "Arduino Bluetooth Controller" from Play Store)

7. Pair your Phone with HC-05

Default pairing code: 1234 or 0000

8. **Open Bluetooth Terminal App**

- Connect to HC-05
- Send characters like 1 to turn ON the LED and 0 to turn it OFF

CONNECTIONS:

Arduino UNO Pin	Bluetooth Module	Arduino Development Board
VCC	**5V**	**-**
GND	**GND**	**-**
2	**Tx**	**-**
3	**Rx**	**-**
4	**-**	**LED**

PROGRAM:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //rx,tx
void setup() {
mySerial.begin(9600);
Serial.begin(9600);
pinMode(4, OUTPUT);
}

void loop() {
if(mySerial.available()>0)
    {
        char data=mySerial.read();
        Serial.println(data);
        if(data=='1'){
        digitalWrite(4,HIGH);
        Serial.println("LED ON");
        }
        else if(data=='2'){
         digitalWrite(4,LOW);
         Serial.println("LED OFF");
        }
    }
}
```

RESULT:

Thus a program to control an LED using a Bluetooth module has been executed successfully.

EXP NO: 6(b)

DATE

ZIGBEE COMMUNICATION

AIM:

To write a program to control an LED using a Zigbee module.

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Arduino IDE 2.0	1
2	Arduino UNO Development Board	2
3	Jumper Wires	few
4	Arduino USB Cable	2
5	Zigbee Module	2

PROCEDURE

1. Configure the Zigbee Modules (using XCTU)

2. Wiring and Setup

3. Upload Arduino Code to Receiver

4. **Send Commands from Transmitter**

   Use **Serial Monitor** in XCTU to send:

   `1` → LED turns ON

   `0` → LED turns OFF

CONNECTIONS:

TRANSMITTER:

Arduino UNO Pin	Zigbee Module
VCC	**5V**
GND	**G**
2	**Tx**
3	**Rx**

PROGRAM:
TRANSMITTER SIDE:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //rx,tx
void setup() {
        mySerial.begin(9600);
        Serial.begin(9600);
}

void loop() {
        mySerial.write('A');
        Serial.println('A');
        delay(100);
        mySerial.write('B');
        Serial.println('B');
        delay(100);
}
```

CONNECTIONS:

RECEIVER:

Arduino UNO Pin	Zigbee Module	Arduino Development Board
-	5V	5V
-	G	GND
2	Tx	-
3	Rx	-
4	-	LED1

RECEIVER SIDE:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //rx,tx
void setup() {
        mySerial.begin(9600);
        Serial.begin(9600);
        pinMode(4, OUTPUT);
}

void loop() {
if(mySerial.available()>0)
{
          char data=mySerial.read();
          Serial.println(data);
          if(data=='A')
          digitalWrite(4,HIGH);
          else if(data=='B')
          digitalWrite(4,LOW);
}
}
```

RESULT:

Thus a program to control an LED using a Zigbee module has been executed successfully.

EXP NO:7(a)

DATE

INTRODUCTION TO THE RASPBERRY PI PLATFORM

Introduction to Raspberry Pi Pico W:

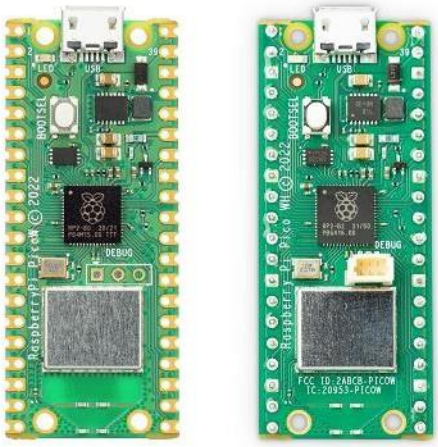
The Raspberry Pi Pico W is a compact and affordable microcontroller board developed by the Raspberry Pi Foundation. Building upon the success of the Raspberry Pi Pico, the Pico W variant brings wireless connectivity to the table, making it an even more versatile platform for embedded projects. In this article, we will provide a comprehensive overview of the Raspberry Pi Pico W, highlighting its key features and capabilities.

Features:

- RP2040 microcontroller with 2MB of flash memory
- On-board single-band 2.4GHz wireless interfaces (802.11n)
- Micro USB B port for power and data (and for reprogramming the flash)
- 40 pins 21mmx51mm ‘DIP’ style 1mm thick PCB with 0.1″ through-hole pins also with edge castellations
- Exposes 26 multi-function 3.3V general purpose I/O (GPIO)
- 23 GPIO are digital-only, with three also being ADC-capable
- Can be surface mounted as a module
- 3-pin ARM serial wire debug (SWD) port
- Simple yet highly flexible power supply architecture
- Various options for easily powering the unit from micro-USB, external supplies, or batteries
- High quality, low cost, high availability
- Comprehensive SDK, software examples, and documentation
- Dual-core Cortex M0+ at up to 133MHz
- On-chip PLL allows variable core frequency
- 264kByte multi-bank high-performance SRAM

Raspberry Pi Pico W:

The Raspberry Pi Pico W is based on the RP2040 microcontroller, which was designed by Raspberry Pi in-house. It combines a powerful ARM Cortex-M0+ processor with built-in Wi-Fi connectivity, opening up a range of possibilities for IoT projects, remote monitoring, and wireless communication. The Pico W retains the same form factor as the original Pico, making it compatible with existing Pico accessories and add-ons.



RP2040 Microcontroller:

At the core of the Raspberry Pi Pico W is the RP2040 microcontroller. It features a dual-core ARM Cortex-M0+ processor running at 133MHz, providing ample processing power for a wide range of applications. The microcontroller also includes 264KB of SRAM, which is essential for storing and manipulating data during runtime. Additionally, the RP2040 incorporates 2MB of onboard flash memory for program storage, ensuring sufficient space for your code and firmware.

Wireless Connectivity:

The standout feature of the Raspberry Pi Pico W is its built-in wireless connectivity. It includes an onboard Cypress CYW43455 Wi-Fi chip, which supports dual-band (2.4GHz and 5GHz) Wi-Fi 802.11b/g/n/ac. This allows the Pico W to seamlessly connect to wireless networks, communicate with other devices, and access online services. The wireless capability opens up new avenues for IoT projects, remote monitoring and control, and real-time data exchange.

GPIO and Peripherals:

Similar to the Raspberry Pi Pico, the Pico W offers a generous number of GPIO pins, providing flexibility for interfacing with external components and peripherals. It features 26 GPIO pins, of which 3 are analog inputs, and supports various protocols such as UART, SPI, I2C, and PWM. The Pico W also includes onboard LED indicators and a micro-USB port for power and data connectivity.

MicroPython and C/C++ Programming:

The Raspberry Pi Pico W can be programmed using MicroPython, a beginner-friendly programming language that allows for rapid prototyping and development. MicroPython provides a simplified syntax and high-level abstractions, making it easy for newcomers to get started. Additionally, the

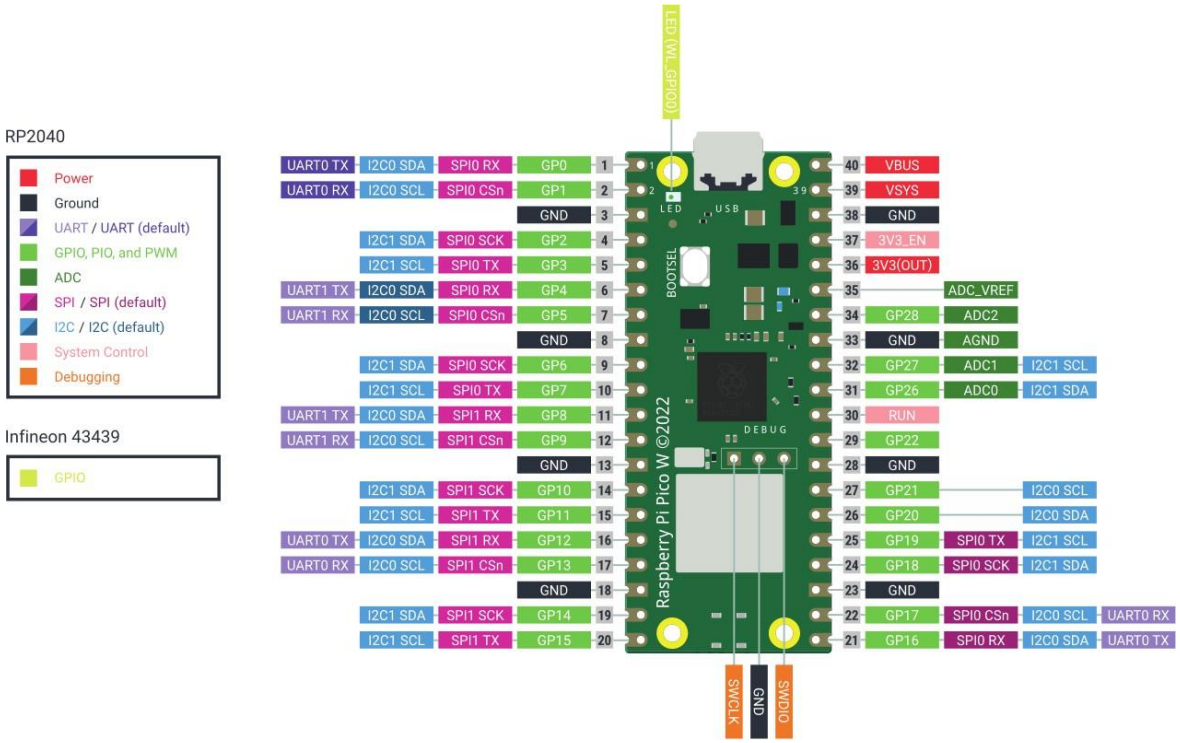
Pico W is compatible with C/C++ programming, allowing experienced developers to leverage the rich ecosystem of libraries and frameworks available.

Programmable Input/Output (PIO) State Machines:

One of the unique features of the RP2040 microcontroller is the inclusion of Programmable Input/Output (PIO) state machines. These state machines provide additional processing power and flexibility for handling real-time data and timing-critical applications. The PIO state machines can be programmed to interface with custom protocols, generate precise waveforms, and offload tasks from the main processor, enhancing the overall performance of the system.

Open-Source and Community Support

As with all Raspberry Pi products, the Pico W benefits from the vibrant and supportive Raspberry Pi community. Raspberry Pi provides extensive documentation, including datasheets, pinout diagrams, and programming guides, to assist developers in understanding the board's capabilities. The community offers forums, online tutorials, and project repositories, allowing users to seek help, share knowledge, and collaborate on innovative projects.



The Raspberry Pi Pico W brings wireless connectivity to the popular Raspberry Pi Pico microcontroller board. With its powerful RP2040 microcontroller, built-in Wi-Fi chip, extensive GPIO capabilities, and compatibility with MicroPython and C/C++ programming, the Pico W offers a versatile and affordable platform for a wide range of embedded projects. Whether you are a beginner or an experienced developer, the Raspberry Pi Pico W provides a user-friendly and flexible platform to bring your ideas to life and explore the exciting world of wireless IoT applications.

RESULT:

Thus the study about the Introduction to Raspberry PI platform has been successfully executed.

EXP NO: 7(b)

DATE

INTRODUCTION TO PYTHON PROGRAMMING

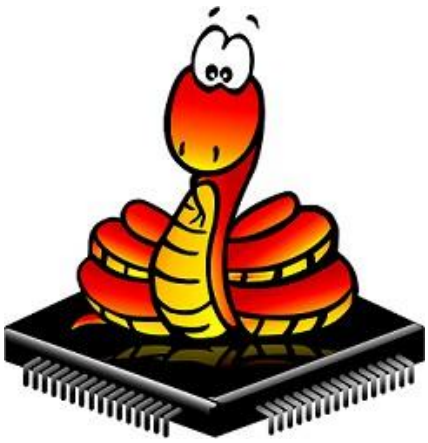
Getting Started with Thonny MicroPython (Python) IDE:

If you want to program your ESP32 and ESP8266 with MicroPython firmware, it's very handy to use an IDE. you'll have your first LED blinking using MicroPython and Thonny IDE.

What is MicroPython?

MicroPython is a Python 3 programming language re-implementation targeted for microcontrollers and embedded systems. MicroPython is very similar to regular Python. Apart from a few exceptions, the language features of Python are also available in MicroPython. The most significant difference between Python and MicroPython is that MicroPython was designed to work under constrained conditions.

Because of that, MicroPython does not come with the entire pack of standard libraries. It only includes a small subset of the Python standard libraries, but it includes modules to easily control and interact with the GPIOs, use Wi-Fi, and other communication protocols.



Thonny IDE:

Thonny is an open-source IDE which is used to write and upload MicroPython programs to different development boards such as Raspberry Pi Pico, ESP32, and ESP8266. It is extremely interactive and easy to learn IDE as much as it is known as the beginner-friendly IDE for new programmers. With the help of Thonny, it becomes very easy to code in Micropython as it has a built-in debugger that helps to find any error in the program by debugging the script line by line.

You can realize the popularity of Thonny IDE from this that it comes pre-installed in Raspian OS which is an operating system for a Raspberry Pi. It is available to install on r Windows, Linux, and Mac OS.

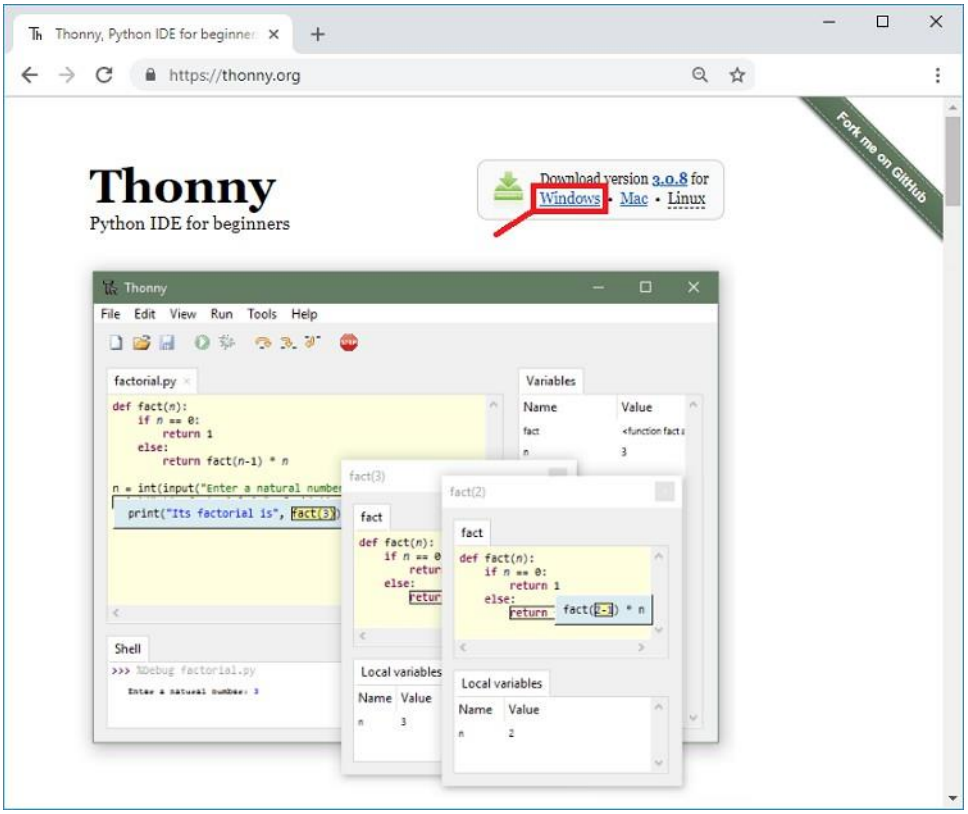
A) Installing Thonny IDE – Windows PC

Thonny IDE comes installed by default on Raspbian OS that is used with the Raspberry Pi board.

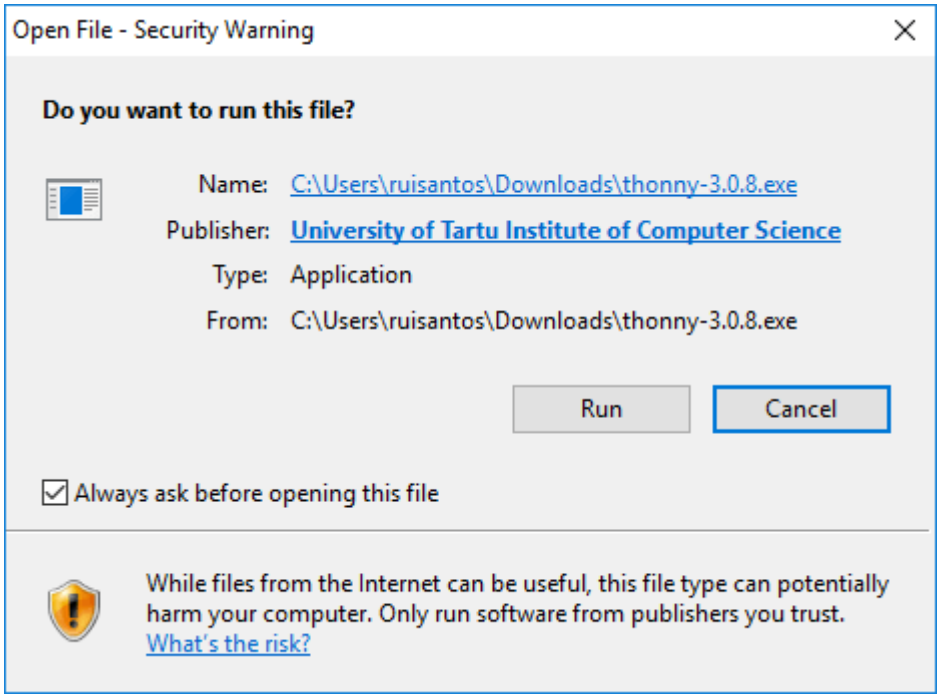
To install Thonny on your Windows PC, follow the next instructions:

1. Go to https://thonny.org

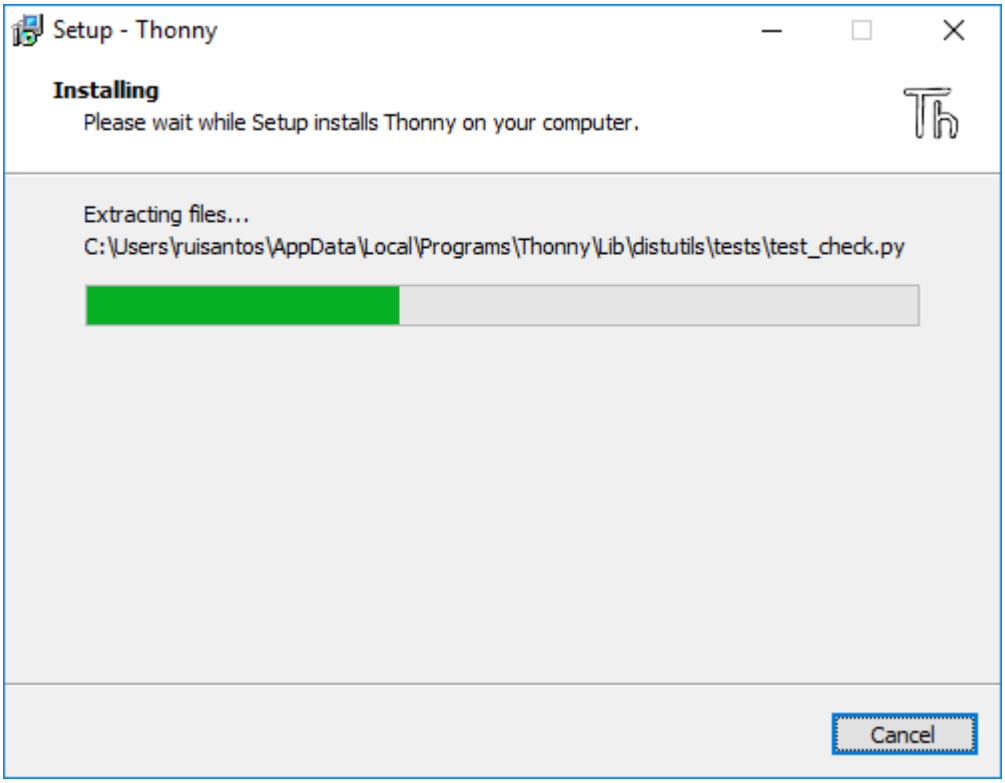
2. Download the version for Windows and wait a few seconds while it downloads.



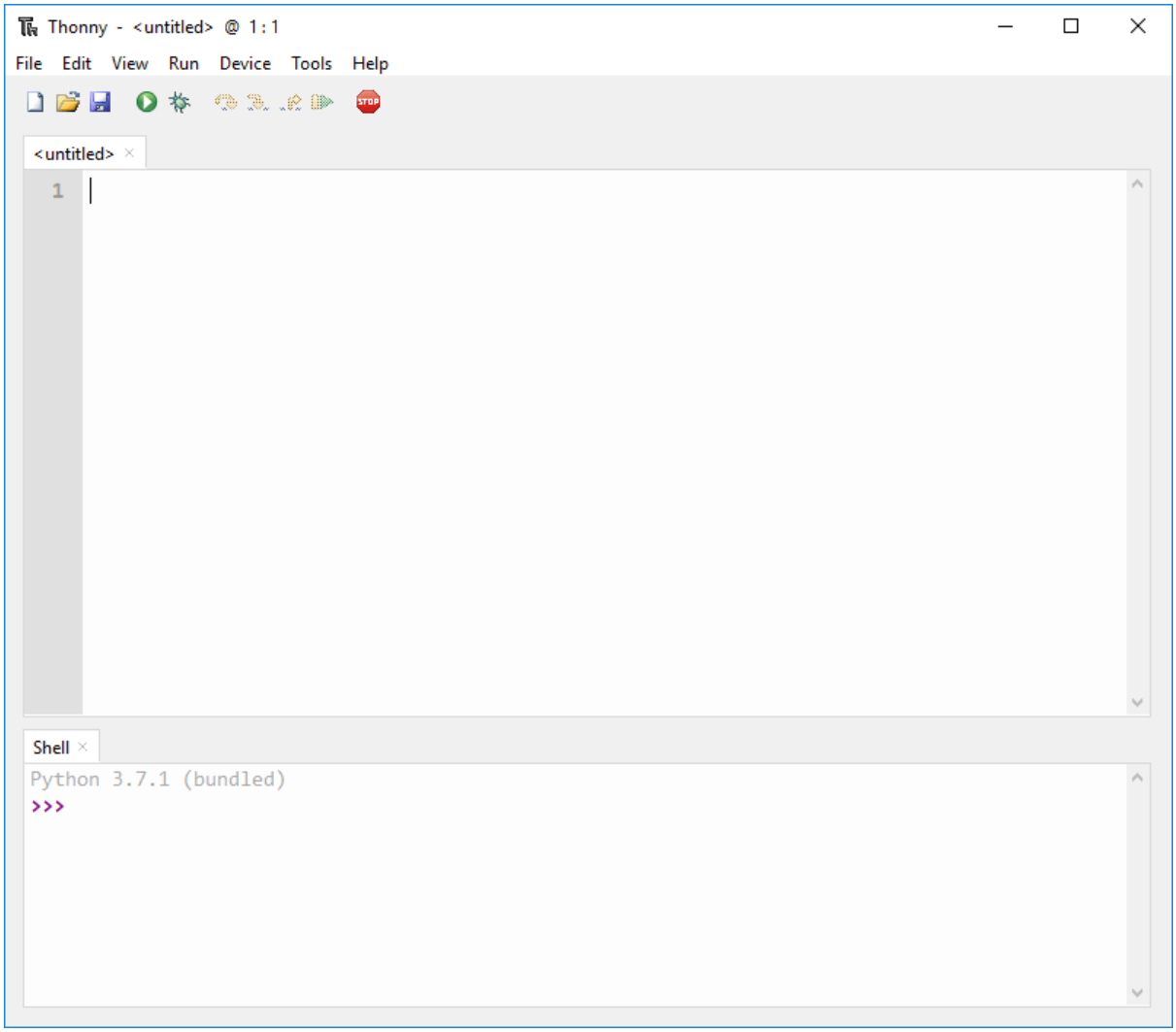
3. Run the *.exe* file.



4. Follow the installation wizard to complete the installation process. You just need to click "Next".



5. After completing the installation, open Thonny IDE. A window as shown in the following figure should open.


Thonny - <untitled> @ 1 : 1
File Edit View Run Device Tools Help
<untitled>
1
Shell
Python 3.7.1 (bundled)
>>>

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board
GP16	**LED**

PROGRAM

LED:

```
from machine import Pin
import time
LED = Pin(16, Pin.OUT)
while True:
    LED.value(1)
    time.sleep(1)
    LED.value(0)
    time.sleep(1)
```

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board (RGB)
GP16	**R**
GP17	**G**
GP18	**B**
GND	**COM**

RGB:

```
from machine import Pin
from time import sleep_ms,sleep
r=Pin(16,Pin.OUT)
y=Pin(17,Pin.OUT)
g=Pin(18,Pin.OUT)

while True:
    r.value(1)
    sleep_ms(1000)
    r.value(0)
    sleep_ms(1000)
    y.value(1)
    sleep(1)
    y.value(0)
    sleep(1)
    g.value(1)
    sleep(1)
    g.value(0)
    sleep(1)
```

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board
GP16	**LED**
GP15	**SW1**

SWITCH CONTROLLED LED:

```
from machine import Pin
from time import sleep
led=Pin(16,Pin.OUT)
sw=Pin(15,Pin.IN)
while True:
    bt=sw.value()
    if bt== True:
        print("LED ON")
        led.value(1)
        sleep(2)
        led.value (0)
        sleep(2)
        led.value (1)
        sleep(2)
        led.value(0)
        sleep(2)
    else:
        print("LED OFF")
    sleep(0.5)
```

RESULT:

Thus the study about Introduction to Python programming has been successfully executed.

EXP NO: 8

DATE

INTERFACING SENSORS WITH RASPBERRY PI

AIM:

To interface the IR sensor and Ultrasonic sensor with Raspberry Pico.

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Thonny IDE	1
2	Raspberry Pi Pico Development Board	1
3	Jumper Wires	few
4	Micro USB Cable	1
5	IR Sensor	1
6	Ultrasonic sensor	1

PROCEDURE

1. Install Thonny IDE
2. Connect the Sensors to the Pico
3. Write the MicroPython Code
4. Testing the Output

- Run the code in Thonny.
- Open the **Shell** window.
- Move your hand or an object in front of the IR and Ultrasonic sensors.

Observe:

"`IR: Obstacle Detected`" when an object is close to the IR sensor.

"`Ultrasonic Distance: XX.XX cm`" showing accurate distance readings.

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board	IR Sensor Module
GP16	**BUZZER**	**-**
GP15	**-**	**OUT**
-	**5V**	**VCC**
-	**GND**	**GND**

PROGRAM:

IR Sensor:

```
from machine import Pin
from time import sleep
buzzer=Pin(16,Pin.OUT)
ir=Pin(15,Pin.IN)
while True:
    ir_value=ir.value()
    if ir_value== True:
        print("Buzzer OFF")
        buzzer.value(0)
    else:
        print("Buzzer ON")
        buzzer.value (1)
    sleep(0.5)
```

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board	Ultrasonic Sensor Module
GP16	**BUZZER**	**-**
GP15	**-**	**ECHO**
GP14	**-**	**TRIG**
-	**5V**	**VCC**
-	**GND**	**GND**

ULTRASONIC SENSOR:

```
from machine import Pin, PWM
import utime
trigger = Pin(14, Pin.OUT)
echo = Pin(15, Pin.IN)
buzzer = Pin(16, Pin.OUT)

def measure_distance():
    trigger.low()
    utime.sleep_us(2)
    trigger.high()
    utime.sleep_us(5)
    trigger.low()
    while echo.value() == 0:
        signaloff = utime.ticks_us()
    while echo.value() == 1:
        signalon = utime.ticks_us()

    timepassed = signalon - signaloff
    distance = (timepassed * 0.0343) / 2
    return distance

 while True:
    dist = measure_distance()
    print(f"Distance : {dist} cm")
    if dist <= 10:
        buzzer.value(1)
        utime.sleep(0.01)
    else:
        buzzer.value(0)
        utime.sleep(0.01)
    utime.sleep(0.5)
```

RESULT:

Thus Interfacing sensors with Raspberry PI has been successfully executed.

EXP NO: 9

DATE

COMMUNICATE BETWEEN ARDUINO AND RASPBERRY PI

AIM:

To write and execute the program to Communicate between Arduino and Raspberry PI using any wireless medium (Bluetooth)

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Thonny IDE	1
2	Raspberry Pi Pico Development Board	1
3	Arduino Uno Development Board	1
4	Jumper Wires	few
5	Micro USB Cable	1
6	Bluetooth Module	2

PROCEDURE

1. Arduino Setup
2. Raspberry Pi Setup
3. Pair with HC-05 Bluetooth Module
4. Find Serial Port
5. Write Python Script on Raspberry Pi

CONNECTIONS:

Arduino UNO Pin	Arduino Development Board	Bluetooth Module
2	**-**	**Tx**
3	**-**	**Rx**
-	**GND**	**GND**
-	**5V**	**5V**

PROGRAM:

MASTER

ARDUINO:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //rx,tx
void setup() {
        mySerial.begin(9600);
}

void loop() {
        mySerial.write('A');
        delay(1000);
        mySerial.write('B');
        delay(1000);
}
```

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board	Bluetooth Module
GP16	**LED**	**-**
VCC	**-**	**+5V**
GND	**-**	**GND**
GP1	**-**	**Tx**
GP0	**-**	**Rx**

SLAVE

RASPBERRY PI PICO

```
from machine import Pin, UART
uart = UART(0, 9600)
led = Pin(16, Pin.OUT)

while True:
  if uart.any() > 0:
    data = uart.read()
    print(data)
    if "A" in data:
      led.value(1)
      print('LED on \n')
      uart.write('LED on \n')
    elif "B" in data:
      led.value(0)
      print('LED off \n')
      uart.write('LED off \n')
```

RESULT:

Thus the program to Communicate between Arduino and Raspberry PI using any wireless medium (Bluetooth) has been successfully executed.

EXP NO: 10

DATE

CLOUD PLATFORM TO LOG THE DATA

AIM:

To set up a cloud platform to log the data from IoT devices.

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No.	Software Requirements	Quantity
1	Blynk Platform	1

CLOUD PLATFORM-BLYNK:

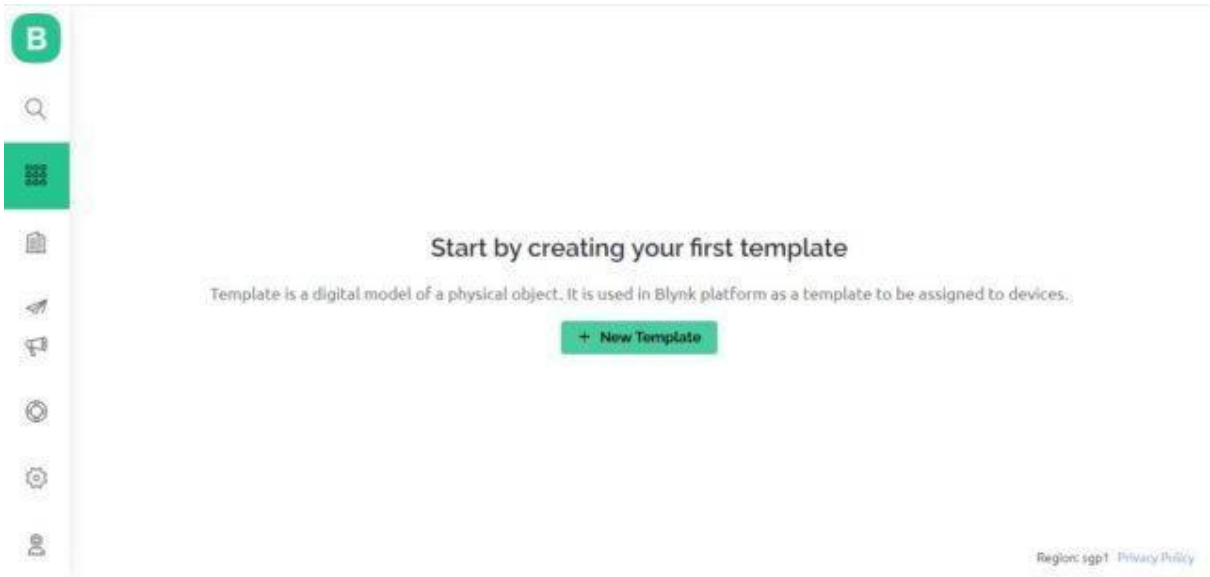
Blynk is a smart platform that allows users to create their Internet of Things applications without the need for coding or electronics knowledge. It is based on the idea of physical programming & provides a platform to create and control devices where users can connect physical devices to the Internet and control them using a mobile app.

Setting up Blynk 2.0 Application

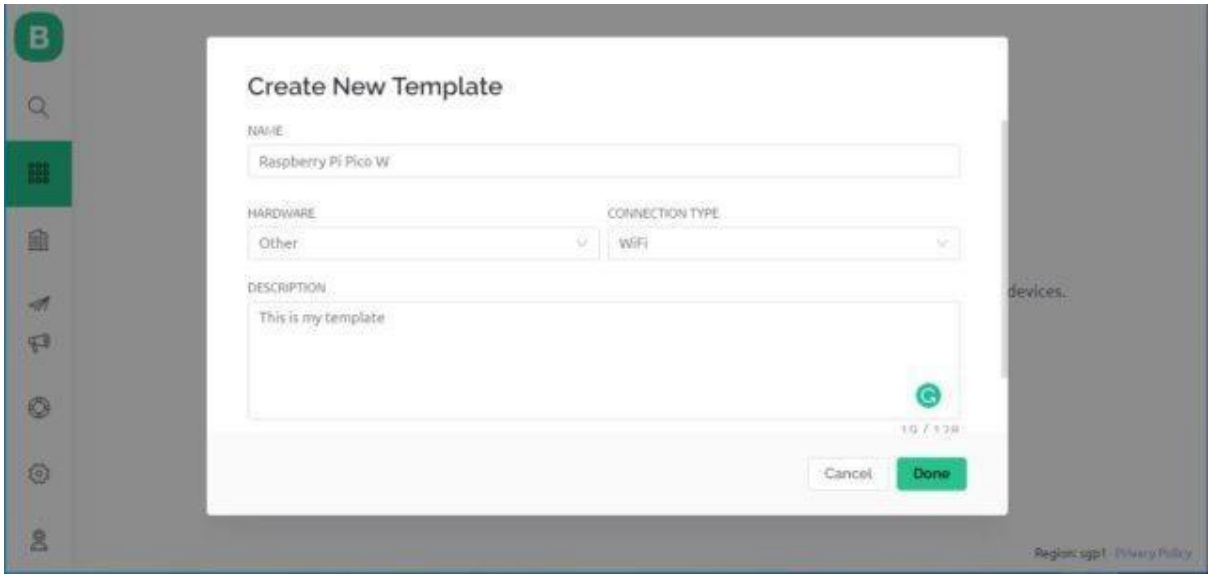
To control the LED using Blynk and Raspberry Pi Pico W, you need to create a Blynk project and set up a dashboard in the mobile or web application. Here's how you can set up the dashboard:

Step 1: Visit **blynk.cloud** and create a Blynk account on the Blynk website. Or you can simply sign in using the registered Email ID.

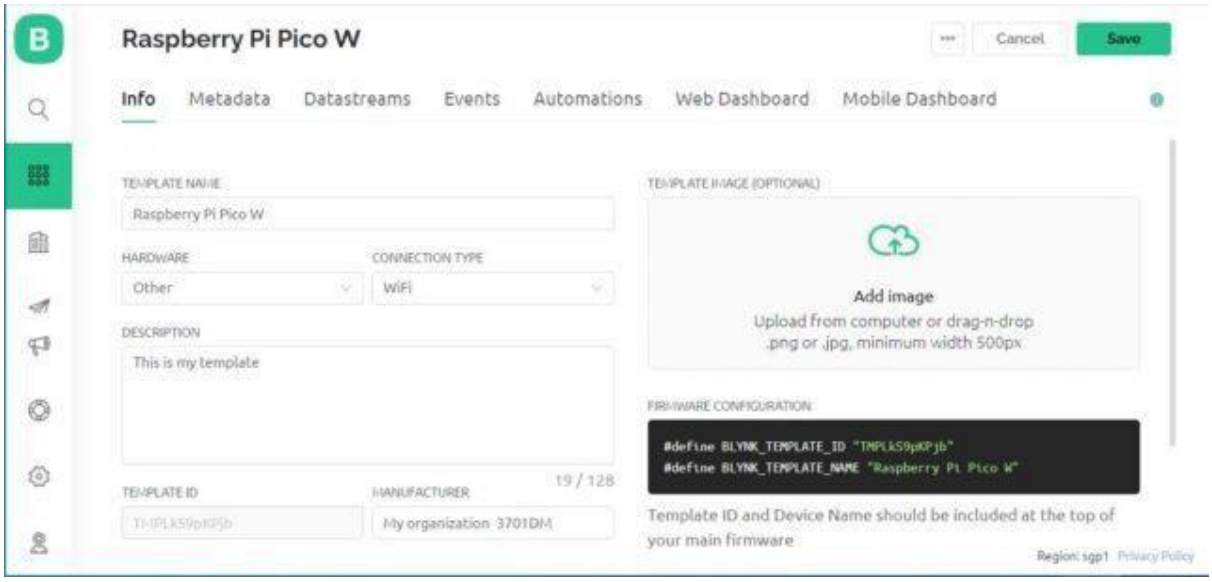
Step 2: Click on **+New Template**.



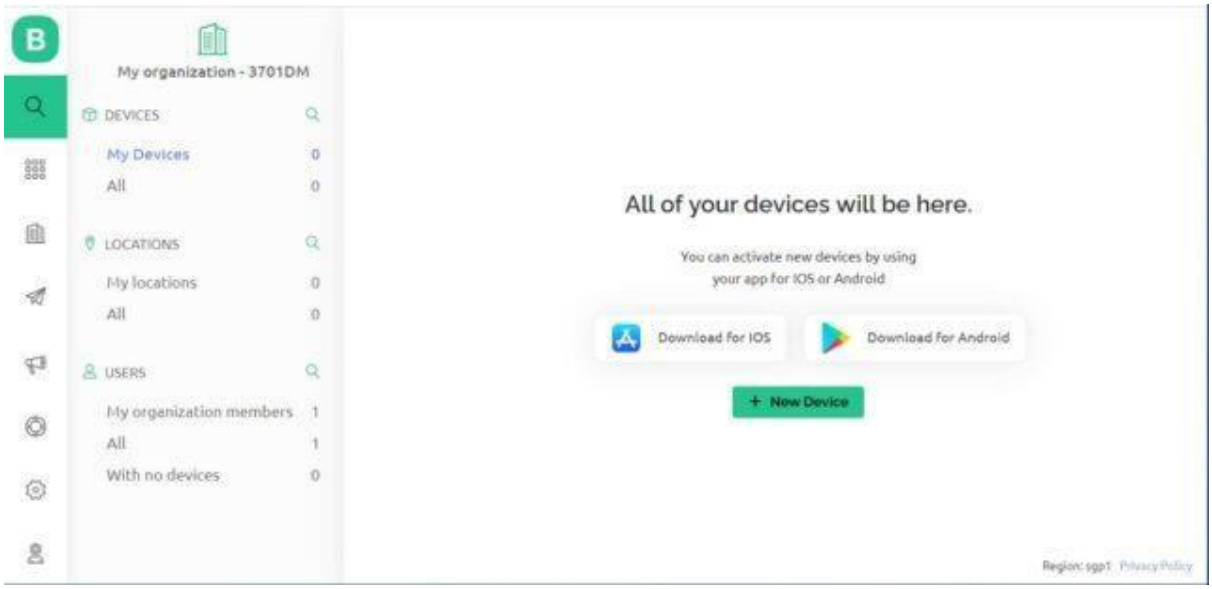
Step 3: Give any name to the Template such as Raspberry Pi Pico W. Select ‘Hardware Type’ as Other and ‘Connection Type’ as WiFi.



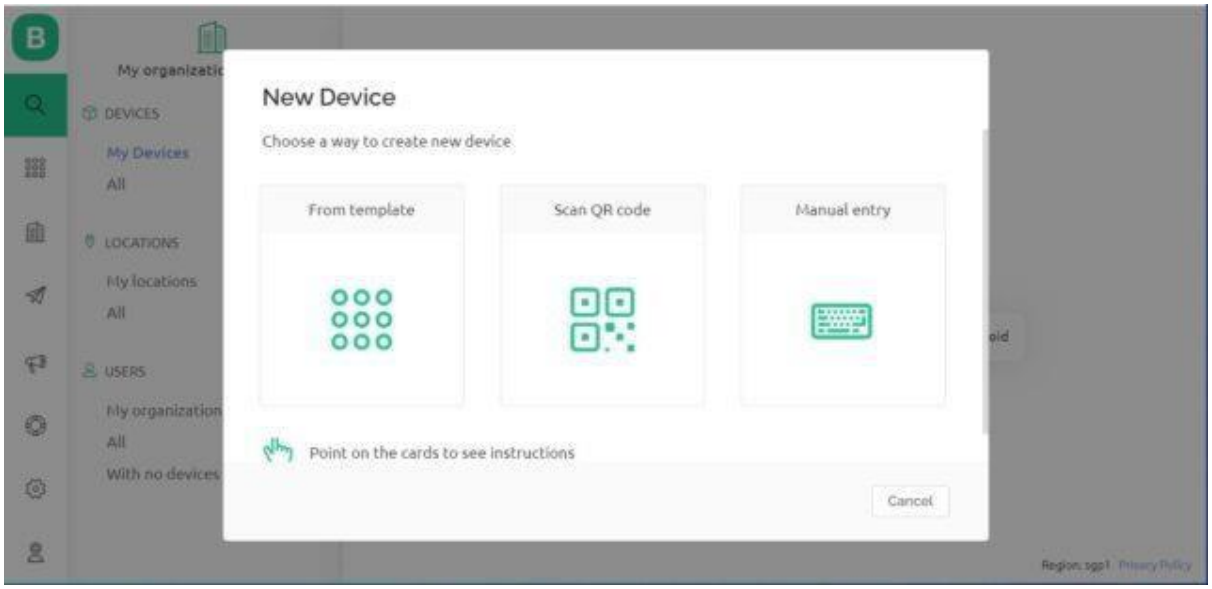
So a template will be created now.



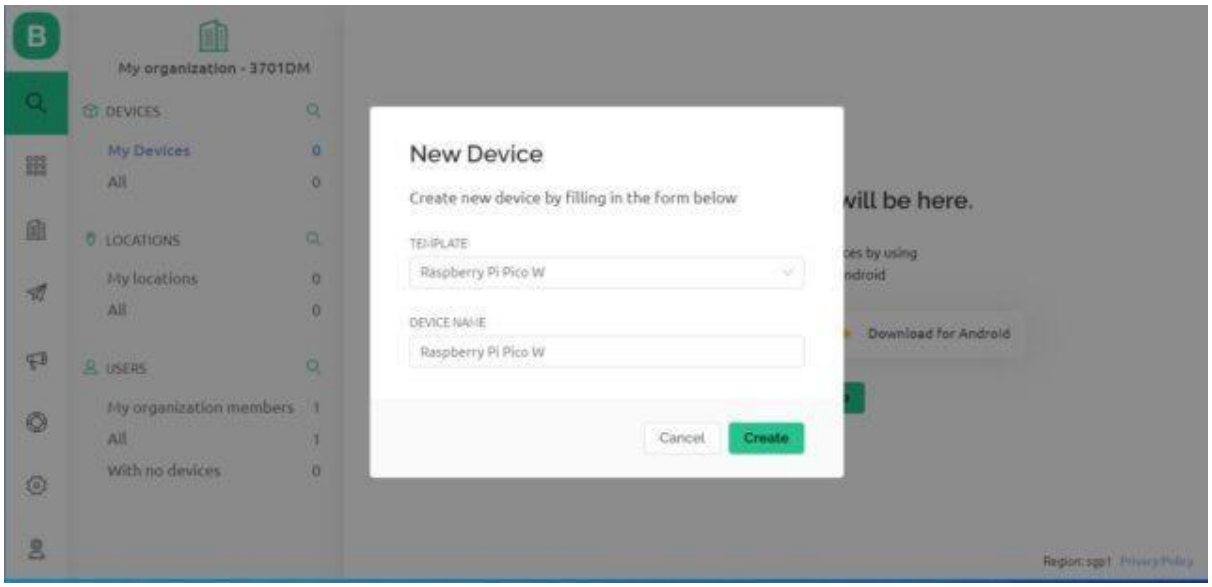
Step 4: Now we need to add a ‘New Device’ now.



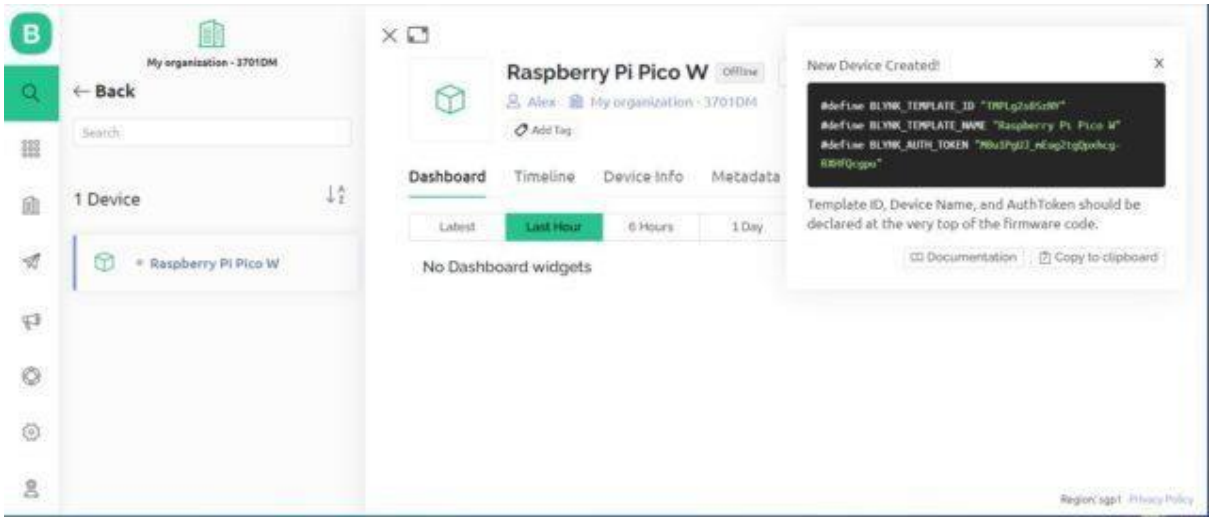
Select a New Device from ‘Template’.



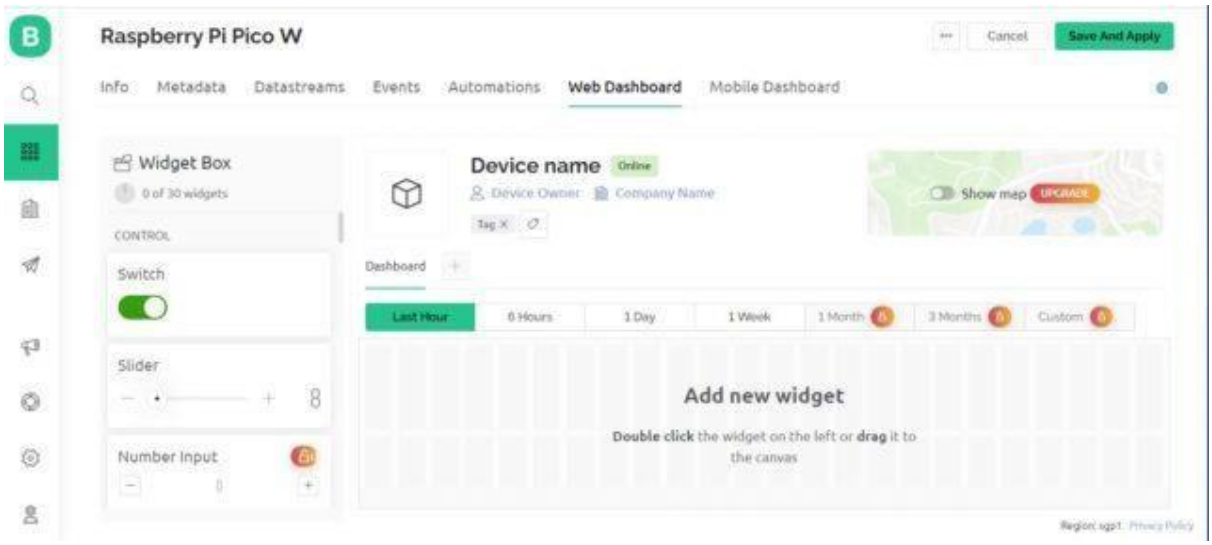
Select the device from a template that you created earlier and also give any name to the device. Click on Create.



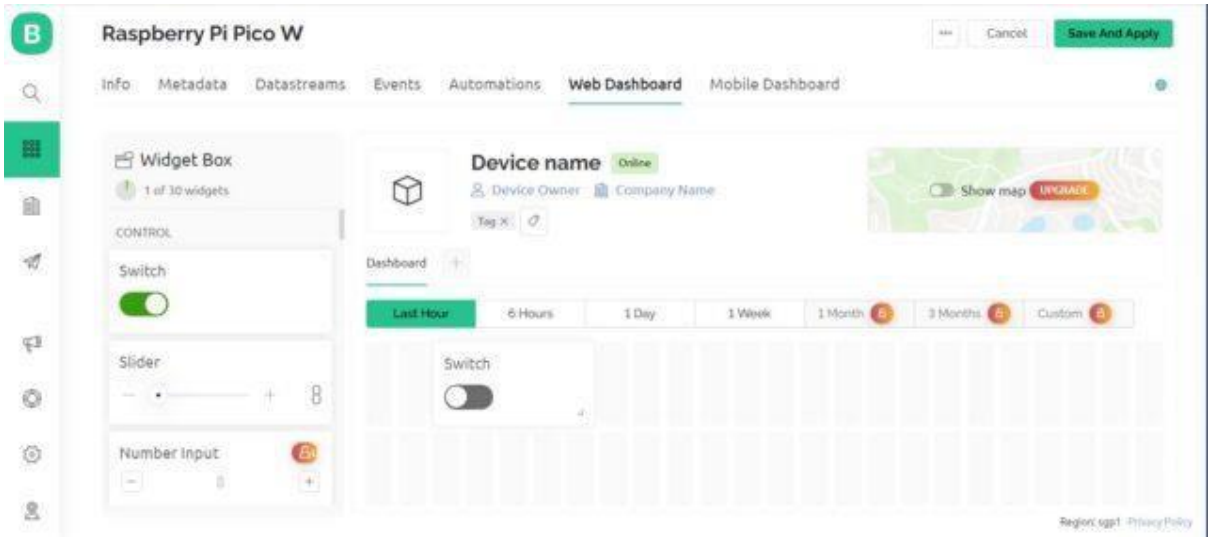
A new device will be created. You will find the Blynk Authentication Token Here. Copy it as it is necessary for the code.



Step 5: Now go to the dashboard and select ‘Web Dashboard’.



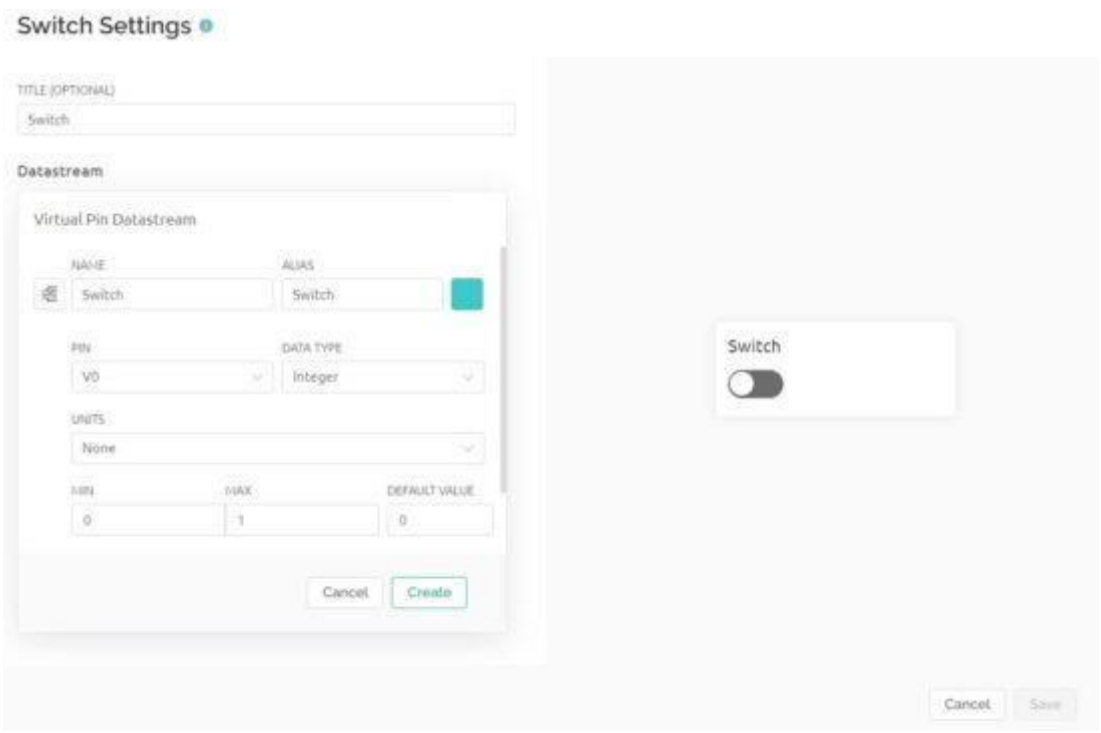
From the widget box drag a switch and place it on the dashboard screen.



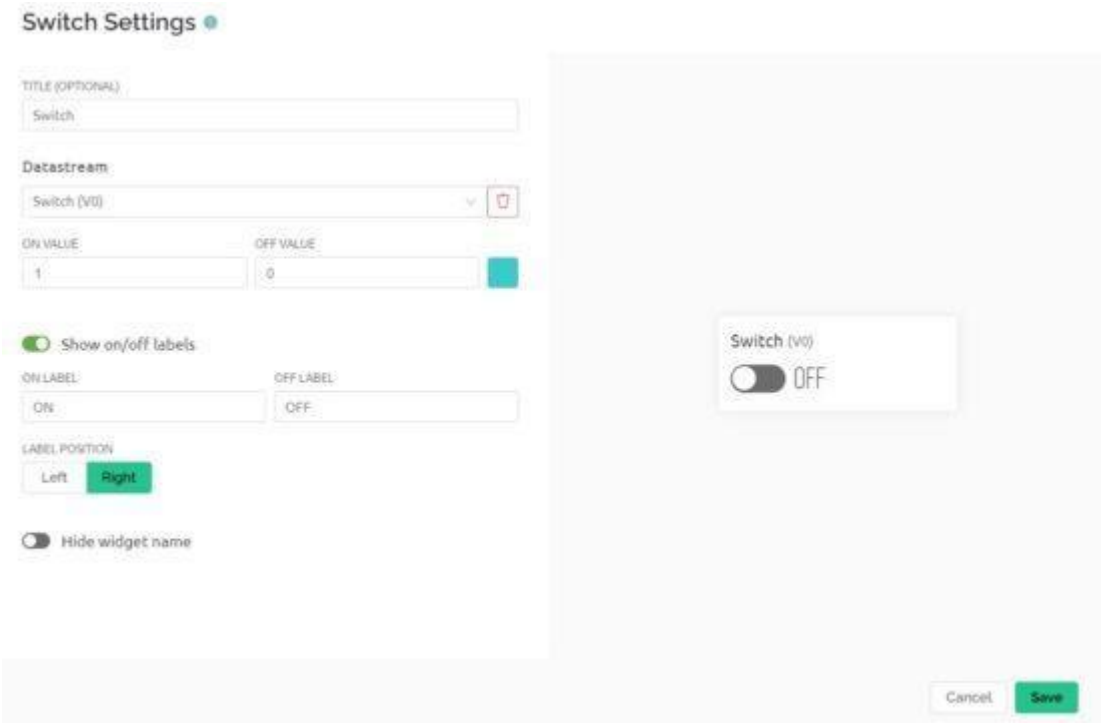
Step 6:

On the switch board click on Settings and here you need to set up the Switch. Give any title to it and Create Datastream as Virtual Pin.

Configure the switch settings as per the image below and click on create.



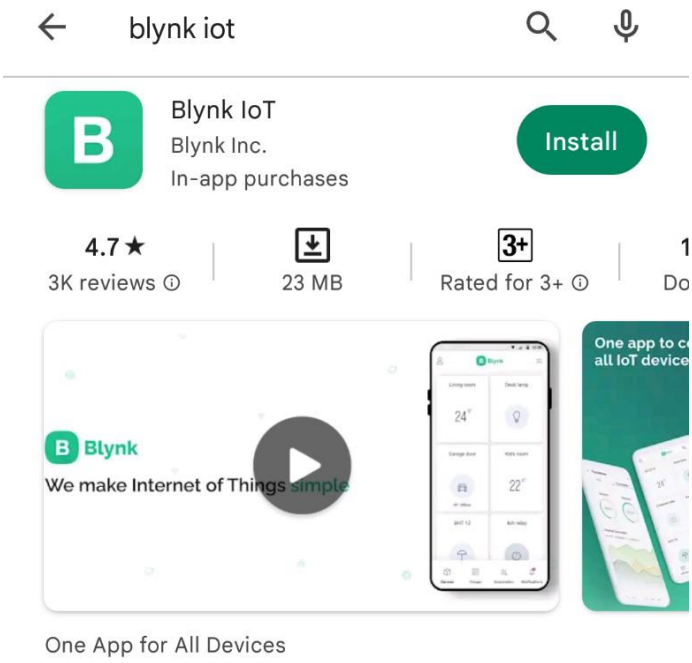
Configure the final steps again.



With this Blynk dashboard set up, you can now proceed to program the Raspberry Pi Pico W board to control the LED.

Step 7:

To control the LED with a mobile App or Mobile Dashboard, you also need to setup the Mobile Phone Dashboard. The process is similarly explained above.



Install the Blynk app on your smartphone The Blynk app is available for iOS and Android. Download and install the app on your smartphone. then need to set up both the Mobile App and the Mobile Dashboard in order to control the LED with a mobile device. The process is explained above.

1. Open Google Play Store App on an android phone
2. Open Blynk.App
3. Log In to your account (using the same email and password)
4. Switch to Developer Mode
5. Find the "**Raspberry Pi Pico Pico W**" template we created on the web and tap on it
6. Tap on the "**Raspberry Pi Pico Pico W**" template (this template automatically comes because we created it on our dashboard).
7. tap on plus icon on the left-right side of the window
8. Add one button Switch
9. Now We Successfully Created an android template
10. it will work similarly to a web dashboard template

RESULT:

Thus the cloud platform set up to log the data from IoT devices has been successfully executed.

EXP NO: 11

DATE

Log Data using Raspberry PI and upload it to the cloud platform

AIM:

To write and execute the program Log Data using Raspberry PI and upload it to the cloud platform

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Thonny IDE	1
2	Raspberry Pi Pico Development Board	few
3	Jumper Wires	1
4	Micro USB Cable	1

PROCEDURE

1. Boot up Raspberry Pi (with internet)

2. Open Terminal and run:

sudo apt update

sudo apt install python3-pip

pip3 install adafruit-circuitpython-dht

sudo apt install libgpiod2

3. Python Code to Read Sensor and Upload Data
Create a ThingSpeak account:

- Go to https://thingspeak.com
- Sign in → Create a new channel → Enable fields like *Temperature*, *Humidity*
- Go to **API Keys** tab and note your **Write API Key**

CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board	LCD Module
-	5V	VCC
-	GND	GND
GP0	-	SDA
GP1	-	SCL

PROGRAM:

```
from machine import Pin, I2C, ADC
from utime import sleep_ms
from pico_i2c_lcd import I2cLcd
import time
import network
import BlynkLib

adc = machine.ADC(4)
i2c=I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)
I2C_ADDR=i2c.scan()[0]
lcd=I2cLcd(i2c,I2C_ADDR,2,16)

wlan = network.WLAN()
wlan.active(True)
wlan.connect("Wifi_Username","Wifi_Password")

BLYNK_AUTH = 'Your_Token'

# connect the network
wait = 10
while wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    wait -= 1
    print('waiting for connection...')
    time.sleep(1)

# Handle connection error
if wlan.status() != 3:
    raise RuntimeError('network connection failed')
else:
```

```
    print('connected')
    ip=wlan.ifconfig()[0]
    print('IP: ', ip)

"Connection to Blynk"
# Initialize Blynk
blynk = BlynkLib.Blynk(BLYNK_AUTH)

lcd.clear()

while True:
    ADC_voltage = adc.read_u16() * (3.3 / (65536))
    temperature_celcius = 27 - (ADC_voltage - 0.706)/0.001721
    temp_fahrenheit=32+(1.8*temperature_celcius)
    print("Temperature in C: {}".format(temperature_celcius))
    print("Temperature in F: {}".format(temp_fahrenheit))

    lcd.move_to(0,0)
    lcd.putstr("Temp:")
    lcd.putstr(str(round(temperature_celcius,2)))
    lcd.putstr("C ")
    lcd.move_to(0,1)
    lcd.putstr("Temp:")
    lcd.putstr(str(round(temp_fahrenheit,2)))
    lcd.putstr("F")
    time.sleep(5)

    blynk.virtual_write(3, temperature_celcius)
    blynk.virtual_write(4, temp_fahrenheit)
    blynk.log_event(temperature_celcius)

    blynk.run()

    time.sleep(5)
```

RESULT:

Thus the program Log Data using Raspberry PI and the study about uploading it to the cloud Platform has been executed successfully.

EXP NO: 12

DATE

Design an IOT-based system

AIM:

To design a Smart Home Automation IOT-based system

HARDWARE & SOFTWARE TOOLS REQUIRED:

S.No	Hardware & Software Requirements	Quantity
1	Thonny IDE	1
2	Raspberry Pi Pico Development Board	few
3	Jumper Wires	1
4	Micro USB Cable	1
5	LED or Relay	1

PROCEDURE

1.Download and install Thonny IDE: https://thonny.org
2. Connect Raspberry Pi Pico via USB.
3.In Thonny:

Go to **Run → Select Interpreter**

Choose: MicroPython (Raspberry Pi Pico)

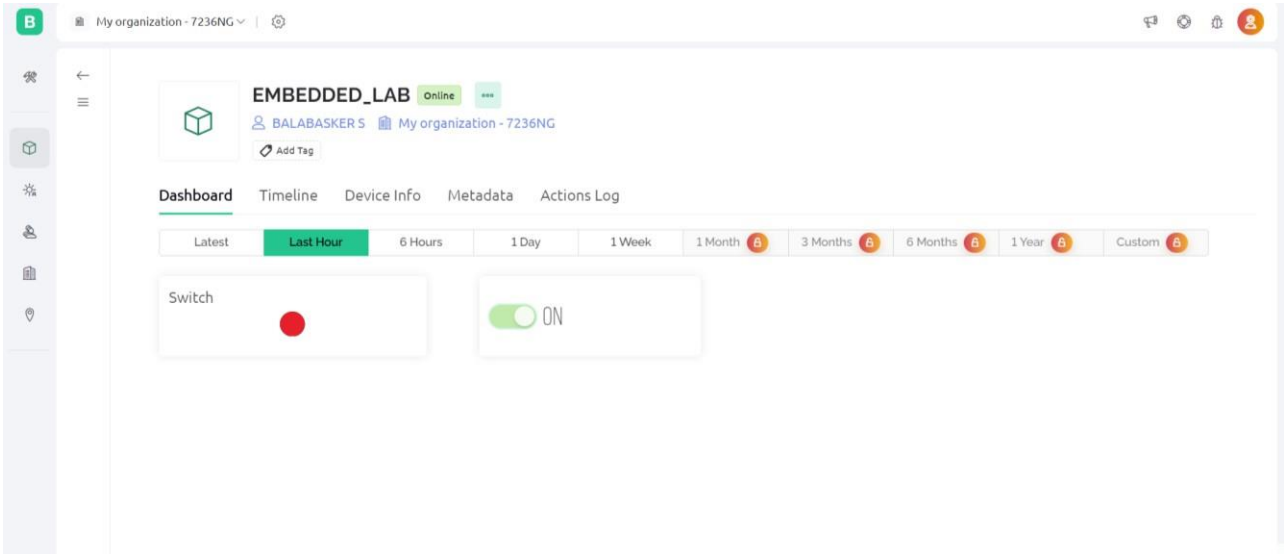
Port: Auto-detected

4.Connect Output Device (LED or Relay)

(If Using Pico W) Connect to WiFi

5.Control LED via Web (Example using MicroPython Web Server)

6.Test the Smart Home Control



CONNECTIONS:

Raspberry Pi Pico Pin	Raspberry Pi Pico Development Board
GP16	LED 1

PROGRAM:

```
import time
import network
import BlynkLib
from machine import Pin
led=Pin(16, Pin.OUT)

wlan = network.WLAN()
wlan.active(True)
wlan.connect("Wifi_Username","Wifi_Password")
BLYNK_AUTH = 'Your_Token'

# connect the network
wait = 10
while wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    wait -= 1
    print('waiting for connection...')
    time.sleep(1)

# Handle connection error
if wlan.status() != 3:
    raise RuntimeError('network connection failed')
else:
    print('connected')
    ip=wlan.ifconfig()[0]
    print('IP: ', ip)

"Connection to Blynk"
# Initialize Blynk
blynk = BlynkLib.Blynk(BLYNK_AUTH)

# Register virtual pin handler
@blynk.on("V0") #virtual pin V0
def v0_write_handler(value): #read the value
    if int(value[0]) == 1:
        led.value(1) #turn the led on
    else:
```

```
    led.value(0)     #turn the led off while True:
blynk.run()
```

RESULT:

Thus the Smart Home Automation design using IoT-based system has been executed successfully.