**Slip 11:**

**Q. Write simple python programs on Pi like**

**a) Light an LED through Python program**

**b) Get input from two switches and switch on corresponding LEDs**

**c) Flash an LED at a given on time and off time cycle, where the two**

**times are taken from a file**

a) Light an LED through a Python program

STEP 1: Launch Proteus and create a new project

STEP 2: Write this code in main.py

import RPi.GPIO as GPIO

import time

# Set up GPIO mode

GPIO.setmode(GPIO.BCM) # Use BCM numbering

GPIO.setwarnings(False)

# Define the pin for the LED

led\_pin = 17 # Example GPIO pin

number # Set up the LED pin as output

GPIO.setup(led\_pin, GPIO.OUT)

# Turn on the LED

GPIO.output(led\_pin,

GPIO.HIGH) # Keep the LED

on for 5 seconds time.sleep(5)

Components Needed in the schematic capture:

1. Raspberry Pi model (e.g., Raspberry Pi 3B or 4B).

2. LED.

3. Wires for connections.

STEP 3: Search for LED in the library.

STEP 4: Connect the default terminal to one end of the LED

STEP 5: Connect ground terminal to other end

STEP 6: Double click on the default terminal and select GPIO 17

CONNECTIONS:

b) Get input from two switches and switch on corresponding LEDs

STEP 1: Write this code in main.py

import RPi.GPIO as GPIO

import time

# Set up GPIO mode

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

# Define the pins for switches and LEDs

switch1\_pin = 4 # GPIO pin for switch 1

switch2\_pin = 17 # GPIO pin for switch 2

led1\_pin = 18 # GPIO pin for LED 1

led2\_pin = 27 # GPIO pin for LED 2

# Set up pins

GPIO.setup(switch1\_pin, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

# Set switch pins as input

GPIO.setup(switch2\_pin, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

# Set switch pins as input

GPIO.setup(led1\_pin, GPIO.OUT) # Set LED pins as output

GPIO.setup(led2\_pin, GPIO.OUT)

while True:

button\_state1 = GPIO.input(switch1\_pin) # Read switch 1 state

button\_state2 = GPIO.input(switch2\_pin) # Read switch 2 state

# Read switches

if button\_state1 == GPIO.HIGH:

GPIO.output(led1\_pin, GPIO.HIGH) # Turn on LED1

else:

GPIO.output(led1\_pin, GPIO.LOW) # Turn off LED1

if button\_state2 == GPIO.HIGH:

GPIO.output(led2\_pin, GPIO.HIGH) # Turn on LED2

else:

GPIO.output(led2\_pin, GPIO.LOW) # Turn off LED2

time.sleep(3)

Components Needed in Proteus:

1. Raspberry Pi (e.g., Raspberry Pi 3B or 4B).

2. Two Push-Button Switches.

3. Two LEDs.

STEP 2: Search for LED in the library.

STEP 3: Connect the default terminal to one end of the LED and ground

terminal to the other end

STEP 4: Double click on the default terminal and select GPIO 18

STEP 5:Follow the same steps for LED 2 but select GPIO 27.

STEP 6: Search for Button in the library

STEP 7: Connect default terminal to one end and Ground to the other end

4

STEP 8: Double click on the default terminal and select GPIO 4

STEP 9: Follow the same steps for second button but select GPIO 17

CONNECTIONS:

c) Flash an LED at a given on-time and off-time cycle, where the times are

taken from a file

Step 1: Write this code in main.py

import RPi.GPIO as GPIO

import time

# Set up GPIO mode

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

# Define the pin for the LED

led\_pin = 4 # Example GPIO pin number

# Set up the LED pin as output

GPIO.setup(led\_pin, GPIO.OUT)

# Read on-time and off-time from a file

with open('file.txt', 'r') as f:

on\_time = float(f.readline().strip()) # Read and convert on-time to float

off\_time = float(f.readline().strip()) # Read and convert off-time to float

while True:

GPIO.output(led\_pin, GPIO.HIGH) # Turn on the LED

print(f"LED ON for {on\_time} seconds")

time.sleep(on\_time) # Keep it on for the on-time duration

GPIO.output(led\_pin, GPIO.LOW) # Turn off the LED

print(f"LED OFF for {off\_time} seconds")

time.sleep(off\_time) # Keep it off for the off-time duration

Step 2: Create a new file file.txt and add the following in the file.

1.0 #will stay on till 1.0

0.5 #will stay off till 0.5

Step 3: Search for LED in the library.

STEP 4: Connect the default terminal to one end of the LED and ground

terminal to the other end

STEP 5: Double click on the default terminal and select GPIO 04

CONNECTIONS:

Notes:

1. Ensure that the `RPi.GPIO` library is installed on your Raspberry Pi. If not,

install it using: sudo apt-get install python3-rpi.gpio

2. Make sure the GPIO pins are correctly wired with the LEDs and switches.

3. The programs use BCM pin numbering; adjust the pin numbers as needed

based on your Raspberry Pi model and wiring.

4. Handle GPIO pins carefully to avoid damage to the