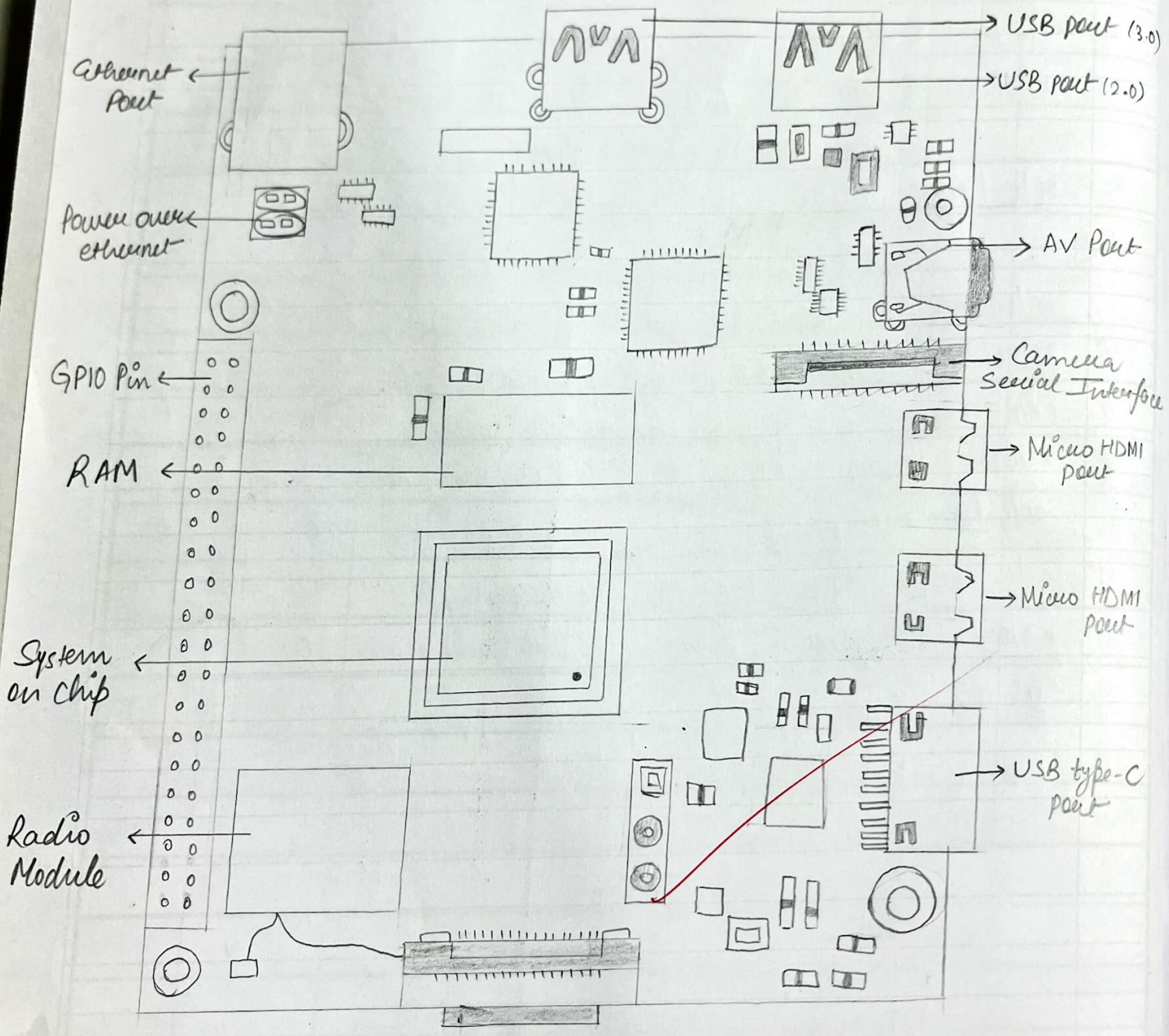
CONTENTS

Ex. No.	Date	Name of Experiment	Page	Remarks
1.	23/01/2023	To Study Raspberry Pi and Beagle Board (one block)	01	
2.	30/01/2023	To make LEDs blink in continous, increase and random order.	04	13/2
3.	06/02/2023	Controlling intensity of LED using PWM	07	
4.	13/02/2023	Interrupt Programming using GPIO Pins	12	20/2
5.	20/02/2023	Connecting Raspberry Pi to 7 Segment display	16	27/2
6.	27/02/2023	Interfacing PIR sensor with Raspberry Pi	19	
7.	06/03/2023	Interfacing with ultrasonic sensor	21	3/3
8.	13/03/2023	Installing & Connecting SQlite to Raspberry Pi	23	20/3
9.	20/03/2023	Interfacing with temperature sensor	26	27/3
10.	27/03/2023	Interfacing 16X2 LCD with raspberry Pi	28	3/4
11.	03/04/2023	Interfacing Raspberry Pi with smoke sensor	30	10/4/23
12.	10/04/2023	Creating web interface for Raspberry Pi	32	24/4

Completed

"Raspberry Pi"



Expt No. 01
Date: 23/01/2023

Page No. 01

Experiment No: 01

AIM: To study Raspberry Pi board and Beagle Bone Black Board.

OBSERVATIONS:

1. Raspberry Pi

Raspberry Pi consist of the following components:

i) System On chip (SOC)

Integrated circuit - which has both CPU and GPU of Raspberry Pi.

ii) RAM (Random Access Memory)

It is located close to the SOC. It is a volatile memory and is used for executing program.

iii) Radio Module

It is used for wireless communication. It handles both wifi and bluetooth connections.

Ports:

i) Universal Serial Bus (USB) ports

They are used to connect keyboard, mic, digital cams, etc.

ii) Ethernet Port

It uses an RJ45 connector and has two small LEDs to indicate status at the bottom.

iii) AV Port

It is a 3.5mm audio - visual Jack. It can be used for both audio and visual output.

It has a tip-ring-ring-sleeve (TRRS)

To study Raspberry Pi board and Beagle Bone Black Board.

Raspberry Pi consist of the following components:

i) System On Chip (SOC)
Integrated circuit which has both CPU and GPU of Raspberry Pi.

ii) RAM (Random Access Memory)
It is located close to the SOC. It is a volatile memory and is used for executing programs.

iii) Radio Module
It is used for wireless communication. It provides both wifi and bluetooth connections.

Ports:

i) Universal Serial Bus (USB) ports
They are used to connect keyboard, mouse, digital cams, etc.

ii) Ethernet Port
It uses an RJ45 connector and has two small LEDs to indicate status of the network.

iii) AV Port-
It is a 3.5mm audio-visual jack. It can be used for both audio and visual. It has a tip-ring-sleeve (TRRS)

"Beagle Bone Black Board"

iv) Camera Serial Interface (CSI)
It is a camera connector port specially designed for Raspberry Pi camera module.
v) Micro HDMI Ports
Used to connect displays to Raspberry Pi.
vi) USB type-C Port
It is a power port
vii) Display Serial Input (DSI)
It is used with Raspberry Pi's touch display.
viii) General Purpose Input Output (GPIO) Pins
A total of 40 pins arranged into 20 rows.
ix) Power Over Ethernet (POE) hat
It is used to take power from a network connection rather than the USB type-C port.

Under the board there is an SD-Card connector used to connect an SD card. It act as the storage of the Raspberry Pi. It has files, softwares and Raspberry Pi OS.

2. BeagleBone Black Board
A BeagleBone Black Board consist of the following components:

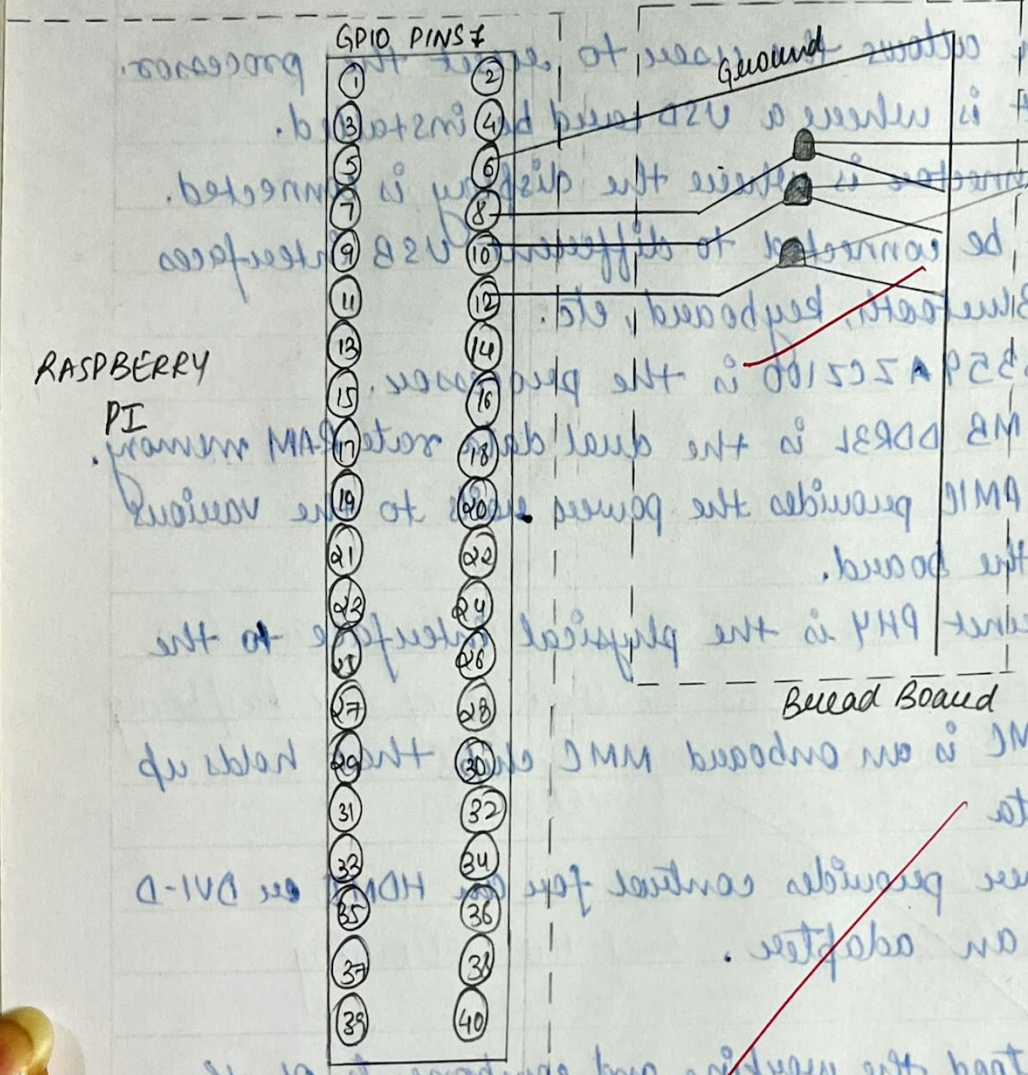
i) DC Power
It is the main DC input that accepts 5V power.
ii) Power Button
It alerts the processor to initiate the power down sequence
iii) 10/100 Ethernet is the connection to the LAN.

Expt No.
Date: / /

iv) Serial debug is the serial debug port.
v) USB client is a mini USB connection to the PC that can also power the board.
vi) BOOT switch can be used to force a boot from SD card.
vii) There are four blue LEDs that can be used by the user.
viii) Reset Button allows the user to reset the processor.
ix) USD Soft Slot is where a USD card be installed.
x) Micro HDMI connector is where the display is connected.
xi) USB host can be connected to different USB interfaces such as Wi-fi, Bluetooth, keyboard, etc.
xii) Sitara AM3359AZCZ100 is the processor.
xiii) Micron 512MB DDR3L is the dual data rate RAM memory.
xiv) TPS65217C PMIC provides the power rails to the various components on the board.
xv) SMSC ethernet PHY is the physical interface to the network.
xvi) Micron eMMC is an onboard MMC chip that holds up to 2GB of data
xvii) HDMI framer provides control for an HDMI or DVI-D display with an adapter.

RESULT: Understood the working and components of the Raspberry Pi board and the BeagleBoneBlackBoard.

12/2



Interfacing LED with GPIO Pins Circuit Diagram

Expt No. 02
Date: 30/01/2023

Experiment No: 02

Interfacing LED with GPIO Pins

AIM: To make LEDs blink in continuous, reverse and random order using python and raspberry pi.

PROCEDURE:

* Connect keyboard and mouse to USB port and monitor to HDMI port.
* SD card already contains pre-installed OS
* Provide power supply and write program in the editor for all three orders.
* Attach the ground supply to 6th pin, voltage to 1st pin and input pins to 8th, 10th and 12th pin.

PROGRAM:

1. Serial Order / Continous Order

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO, initial = GPIO.LOW)
GPIO.setup(10, GPIO, initial = GPIO.LOW)
GPIO.setup(12, GPIO, initial = GPIO.LOW)
while True:
        GPIO.output(8, GPIO.HIGH)
        sleep(2)
        GPIO.output(10, GPIO.HIGH)
        sleep(2)
        GPIO.output(12, GPIO.HIGH)
        sleep(2)
```

Expt No.
Date: / /

Page No. 05

2. Reverse Order

```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO, initial = GPIO.LOW)
GPIO.setup(10, GPIO, initial = GPIO.HIGH)
GPIO.setup(12, GPIO, initial = GPIO.LOW)
while True:
    GPIO.output(8, GPIO.HIGH)
    sleep(2)
    GPIO.output(10, GPIO.HIGH)
    sleep(2)
    GPIO.output(12, GPIO.HIGH)
    sleep(2)
    GPIO.output(12, GPIO.LOW)
    GPIO.output(10, GPIO.LOW)
    GPIO.output(8, GPIO.LOW)
    sleep(2)
    GPIO.output(12, GPIO.HIGH)
    sleep(2)
    GPIO.output(10, GPIO.HIGH)
    sleep(2)
    GPIO.output(8, GPIO.HIGH)
    sleep(2)
    GPIO.output(12, GPIO.LOW)
    GPIO.output(10, GPIO.LOW)
    GPIO.output(8, GPIO.LOW)
    sleep(2)
```

Expt No.
Date: / /

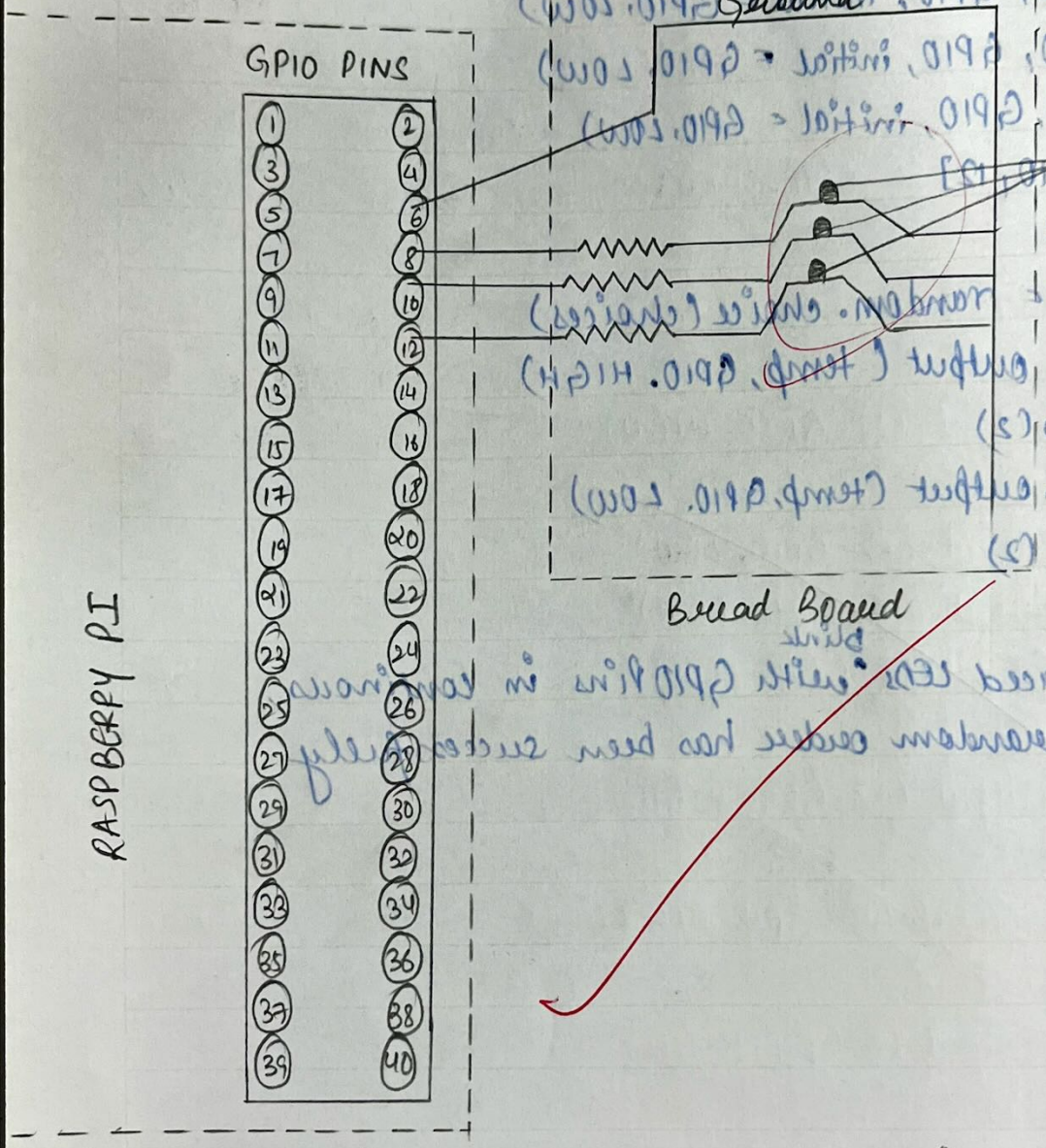
Page No. 06

3. Random Order

```
import RPi.GPIO as GPIO
import time
import random
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO, initial = GPIO.LOW)
GPIO.setup(10, GPIO, initial = GPIO.LOW)
GPIO.setup(12, GPIO, initial = GPIO.LOW)
choices = [8, 10, 12]
while True:
        temp = random.choice(choices)
        GPIO.output(temp, GPIO.HIGH)
        sleep(2)
        GPIO.output(temp, GPIO.LOW)
        sleep(2)
```

RESULT: Interfaced LEDs blink with GPIO Pins in continous, reverse and random order has been successfully implemented.

13/2



Controlling Intensity of LED using PWM
Circuit Diagram

Expt No.
Date: 06 / 02 / 2023

Experiment No. 03

Controlling intensity of LED light using Pulse Width Modulation

AIM: Controlling intensity of LED light using pulse width modulation and make any four patterns.

PROCEDURE:

1. Connect mouse and keyboard to USB ports of Raspberry Pi.
2. Connect Monitor to HDMI port of Raspberry Pi.
3. Provide power supply and write programs for four patterns.
4. Connect positive terminals of LEDs to GPIO 8, GPIO 10, GPIO 12 and negative terminals to ground (GPIO 6).

PROGRAMS:

1. LEDs intensity increasing & decreasing in a serial order

```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(10, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
P = GPIO.PWM(8, 100)
Q = GPIO.PWM(10, 100)
R = GPIO.PWM(12, 100)
P.start(0)
Q.start(0)
R.start(0)
```

```
while True:
    for i in range(0,100,5)
        P.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(0,100,5)
        Q.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(0,100,5)
        R.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        P.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        Q.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        R.ChangeDutyCycle(i)
        time.sleep(0.1)
```

Observations: LED1, LED2, LED3' intensity goes in increasing order then again LED1, LED2, LED3 intensity goes in decreasing order.

2)
```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(false)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
```

Expt No.
Date: / /


```
GPIO.setup(10,GPIO.OUT)
GPIO.setup(12,GPIO.OUT)
P=GPIO.PWM(8,100)
Q=GPIO.PWM(10,100)
R=GPIO.PWM(12,100)
P.start(0)
Q.start(0)
R.start(0)

while True:
    for i in range(0,100,5)
        P.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        P.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(0,100,5)
        Q.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        Q.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(0,100,5)
        R.changeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        R.ChangeDutyCycle(i)
        time.sleep(0.1)
```

Observations: LED1, LED2, LED3 intensity first increases and then decreases in a serial order.

3)
```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(10, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
P = GPIO.PWM(8,100)
Q = GPIO.PWM(10,100)
R = GPIO.PWM(12,100)
P.start(0)
Q.start(0)
R.start(0)
while True:
    for i in range(0,100,5)
        P.ChangeDutyCycle(i)
        Q.ChangeDutyCycle(i)
        R.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100,0,-5)
        P.ChangeDutyCycle(i)
        Q.ChangeDutyCycle(i)
        R.ChangeDutyCycle(i)
        time.sleep(0.1)
```

Observations: Intensities of three LEDs increases and decreases simultaneously.

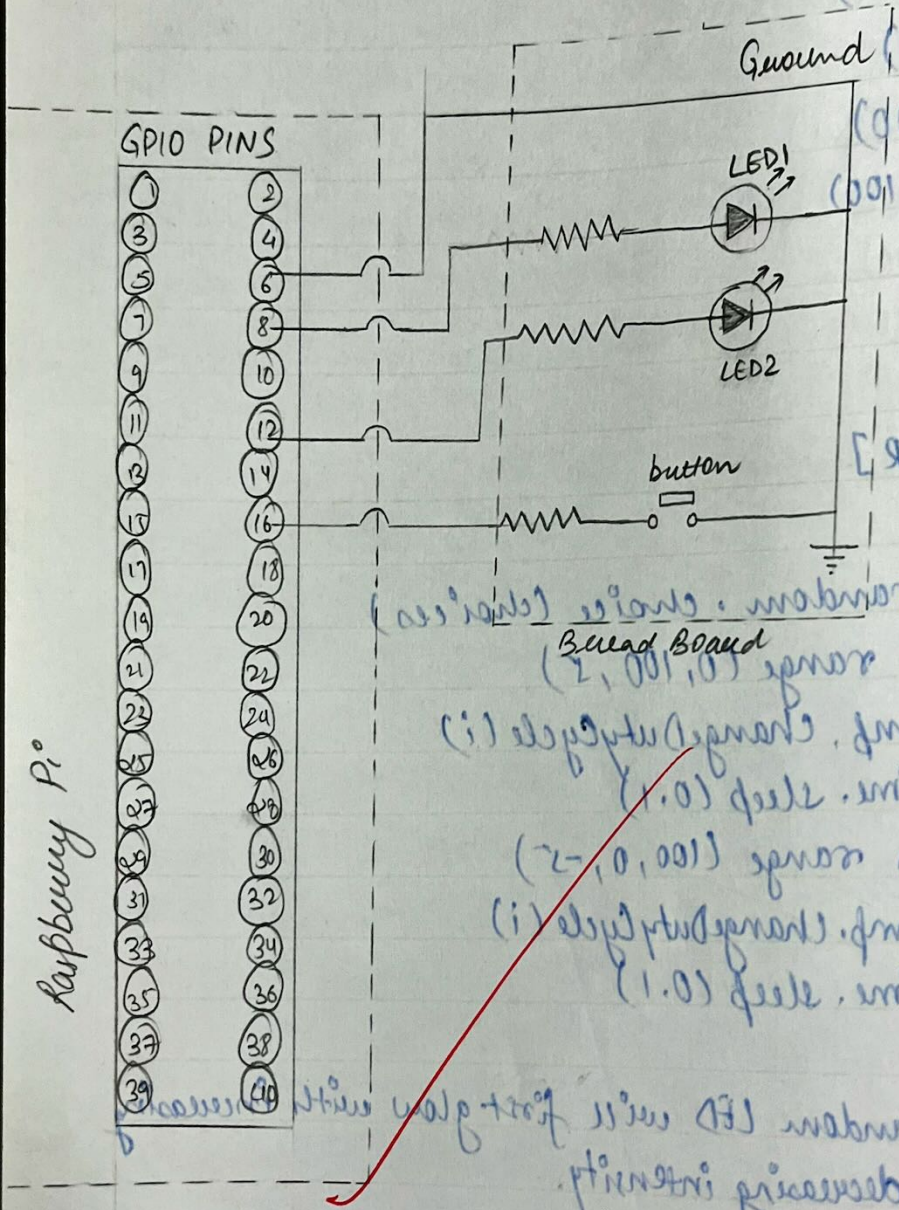
Expt No.
Date: / /

Page No. 11

4) import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(10, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
P = GPIO.PWM(8, 100)
Q = GPIO.PWM(10, 100)
R = GPIO.PWM(12, 100)
P.start(0)
Q.start(0)
R.start(0)
choices = [P, Q, R]
while True:
    temp = random.choice(choices)
    for i in range(0, 100, 5)
        temp.ChangeDutyCycle(i)
        time.sleep(0.1)
    for i in range(100, 0, -5)
        temp.ChangeDutyCycle(i)
        time.sleep(0.1)

Observations: Any random LED will first glow with increasing intensity and then decreasing intensity.

RESULT: Controlled intensity of LEDs using Pulse width modulation has been observed using four different patterns.



Interrupt Programming using GPIO Pins

Circuit Diagram:

Expt No. 04
Date: 13/02/2023

Experiment No. 04

Interrupt Programming using GPIO Pins

AIM: Using a button implement Interrupt Programming and generate patterns.

PROCEDURE:

1. Connect mouse and keyboard to USB ports of Raspberry Pi
2. Connect monitor to HDMI port of Raspberry Pi
3. Provide power supply and write python program for generating 3 patterns.
4. Connect positive terminals of LEDs to GPIO 8 and GPIO 12 button to GPIO 16 pin and negative terminals to ground (GPIO 6th pin)

PROGRAMS:

1. Switch ON and OFF together

```
import RPi.GPIO as GPIO
import time
BUTTON_PIN = 16
GPIO.setwarnings(False)
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT, initial = GPIO.LOW)
GPIO.setup(12, GPIO.OUT, initial = GPIO.LOW)
while True:
    GPIO.wait_for_edge(BUTTON_PIN, GPIO.RISING)
    GPIO.output(8, GPIO.HIGH)
    GPIO.output(12, GPIO.HIGH)
```

```
        time.sleep(1)
        GPIO.wait_for_edge(BUTTON_PIN, GPIO.RISING)
        GPIO.output(8, GPIO.LOW)
        GPIO.output(12, GPIO.LOW)
        time.sleep(1)
GPIO.cleanup()
```

OBSERVATION: when we press the button for the first time both led's light up together and once we press again both go down together.

2) PWM Pattern

```
import RPi.GPIO as GPIO
import time
BUTTON_PIN = 16
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down = GPIO.PUD_UP)
P = GPIO.PWM(8,100)
Q = GPIO.PWM(12,100)
while True:
    P.start(0)
    Q.start(100)
    GPIO.wait_for_edge(BUTTON_PIN, GPIO.FALLING)
    for x in range(0,100,1):
        P.changeDutyCycle(x)
        Q.changeDutyCycle(100-x)
```

```
        time.sleep(1)
    GPIO.wait_for_edge(BUTTON_PIN, GPIO.RISING)
    for x in range(100, 0, -1):
        P.ChangeDutyCycle(x)
        Q.changeDutyCycle(100-x)
        time.sleep(1)
GPIO.cleanup():
```

OBSERVATION: The intensity between both LEDs goes up alternatively as we switch on and off the button.

3. Pattern between 2 LED

```
import RPi.GPIO as GPIO
import time
BUTTON_PIN = 16
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(8, GPIO.OUT, initial = GPIO.LOW)
GPIO.setup(12, GPIO.OUT, initial = GPIO.LOW)
while True:
        GPIO.wait_for_edge(BUTTON_PIN, GPIO.RISING)
        GPIO.output(8, GPIO.HIGH)
        time.sleep(1)
        GPIO.wait_for_edge(BUTTON_PIN, GPIO.RISING)
        GPIO.output(12, GPIO.HIGH)
        time.sleep(1)
        GPIO.wait_for_edge(BUTTON_PIN, GPIO.FALLING)
        GPIO.output(12, GPIO.LOW)
```

Expt No.
Date: / /

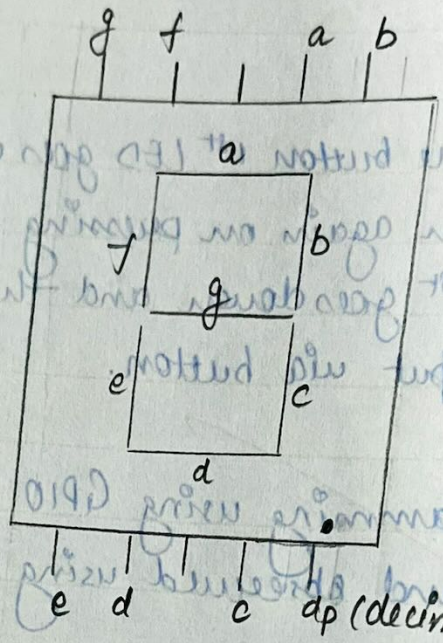
Page No. 15

```
        time.sleep(1)
        GPIO.wait_for_edge(BUTTON_PIN, GPIO.FALLING)
        GPIO.output(8, GPIO.LOW)
        time.sleep(1)
GPIO.cleanup()
```

OBSERVATION: On pressing the button 1st LED goes up again on pressing 2nd LED goes up then again on pressing 2nd LED goes down and after pressing 1st goes down and this pattern repeats as we give the input via button.

RESULT: Interrupt programming using GPIO Pins and button has been implemented and observed using three patterns.

sd/-



7 segment Display

Expt No. 05
Date: 20/02/2023

Page No. 16

Experiment No: 05

Connecting Raspberry Pi to 7 Segment Display

AIMS: i) Display numbers from 0 to 9 on the 7 segment display.
2) Simulating traffic light using LEDs and 7 segment display.

PROCEDURE:

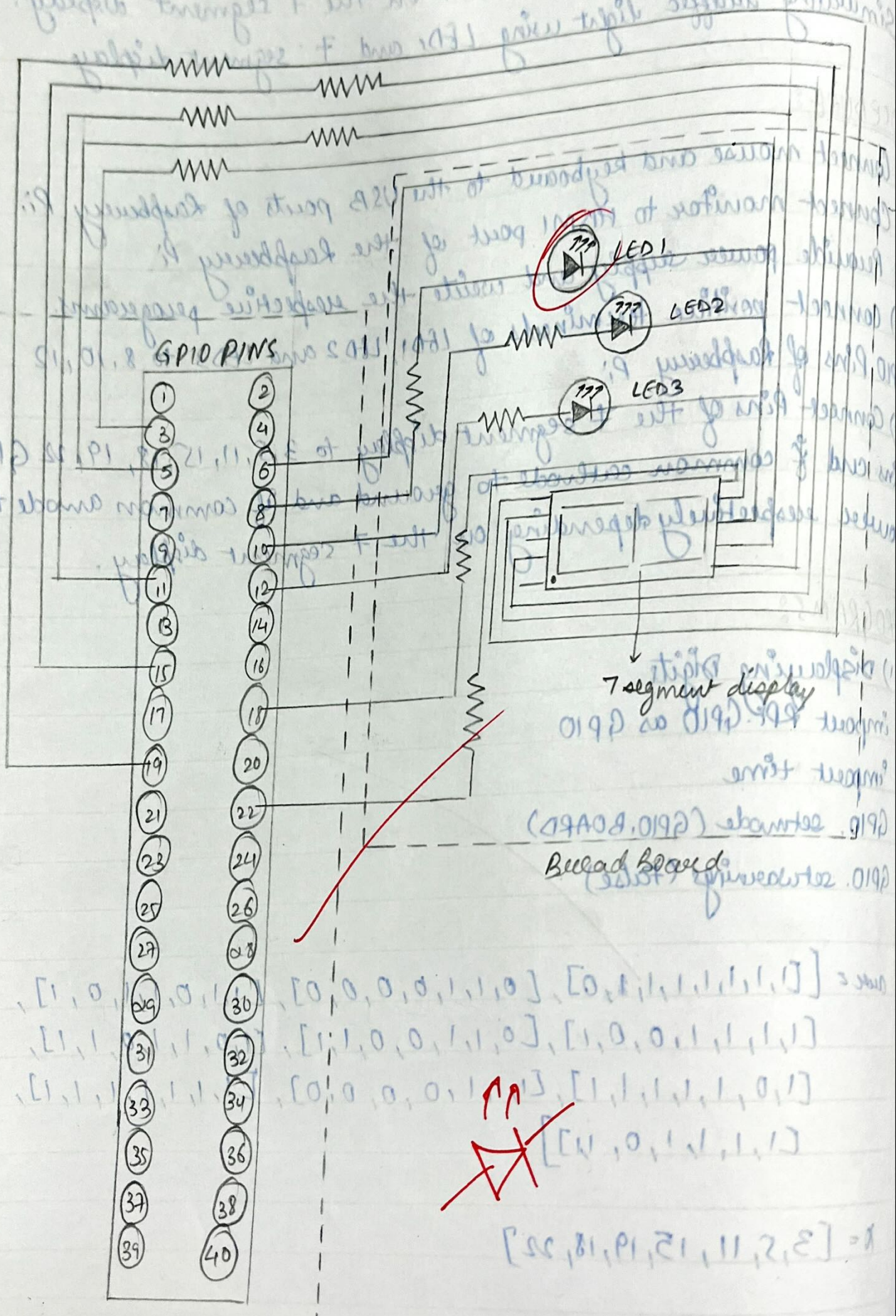
i) Connect mouse and keyboard to the USB ports of Raspberry Pi.
ii) Connect monitor to HDMI port of the Raspberry Pi
iii) Provide power supply and write the respective programs.
iv) Connect positive terminals of LED1, LED2 and LED3 to 8, 10, 12 GPIO Pins of Raspberry Pi
v) Connect Pins of the 7 segment display to 3, 5, 11, 15, 18, 19, 22 GPIO pins and if common cathode to ground and if common anode to power respectively depending on the 7 segment display.

PROGRAMS:

1) Displaying Digits

```
import RPI.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

num = [[1,1,1,1,1,1,0],[0,1,1,0,0,0,0],[1,1,0,1,1,0,1],
       [1,1,1,1,0,0,1],[0,1,1,0,0,1,1],[1,0,1,1,0,1,1],
       [1,0,1,1,1,1,1],[1,1,1,0,0,0,0],[1,1,1,1,1,1,1],
       [1,1,1,1,0,1,1]]

N = [3,5,11,15,19,18,22]
```

Circuit- Diagram



Expt No.
Date: / /

Page No. 17

```
for i in x:
    GPIO.setup(i, GPIO.OUT, initial = GPIO.LOW)
while True:
    for num in arr:
        for i in range(0,7):
            if (num[i]==1)
                GPIO.output(x[i], GPIO.HIGH)
        time.sleep(1.2)
        for i in range(0,7):
            GPIO.output(x[i], GPIO.LOW)
```

OBSERVATION: Digits from 0 to 9 are displayed on the 7 segment display.

2) Traffic light using 7 segment display + LED pattern

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

digits = [[1,1,1,1,1,1,0],[0,1,1,0,0,0,0],[1,1,0,1,1,0,1],
          [1,1,1,1,0,0,1],[0,1,1,0,0,1,1],[1,0,1,1,0,1,1],
          [1,0,1,1,1,1,1],[1,1,1,0,0,0,0,0],[1,1,1,1,1,1,1],
          [1,1,1,1,0,1,1]]
x = [3,5,11,15,19,18,22]

for i in x:
    GPIO.setup(i, GPIO.OUT, initial = GPIO.LOW)
```

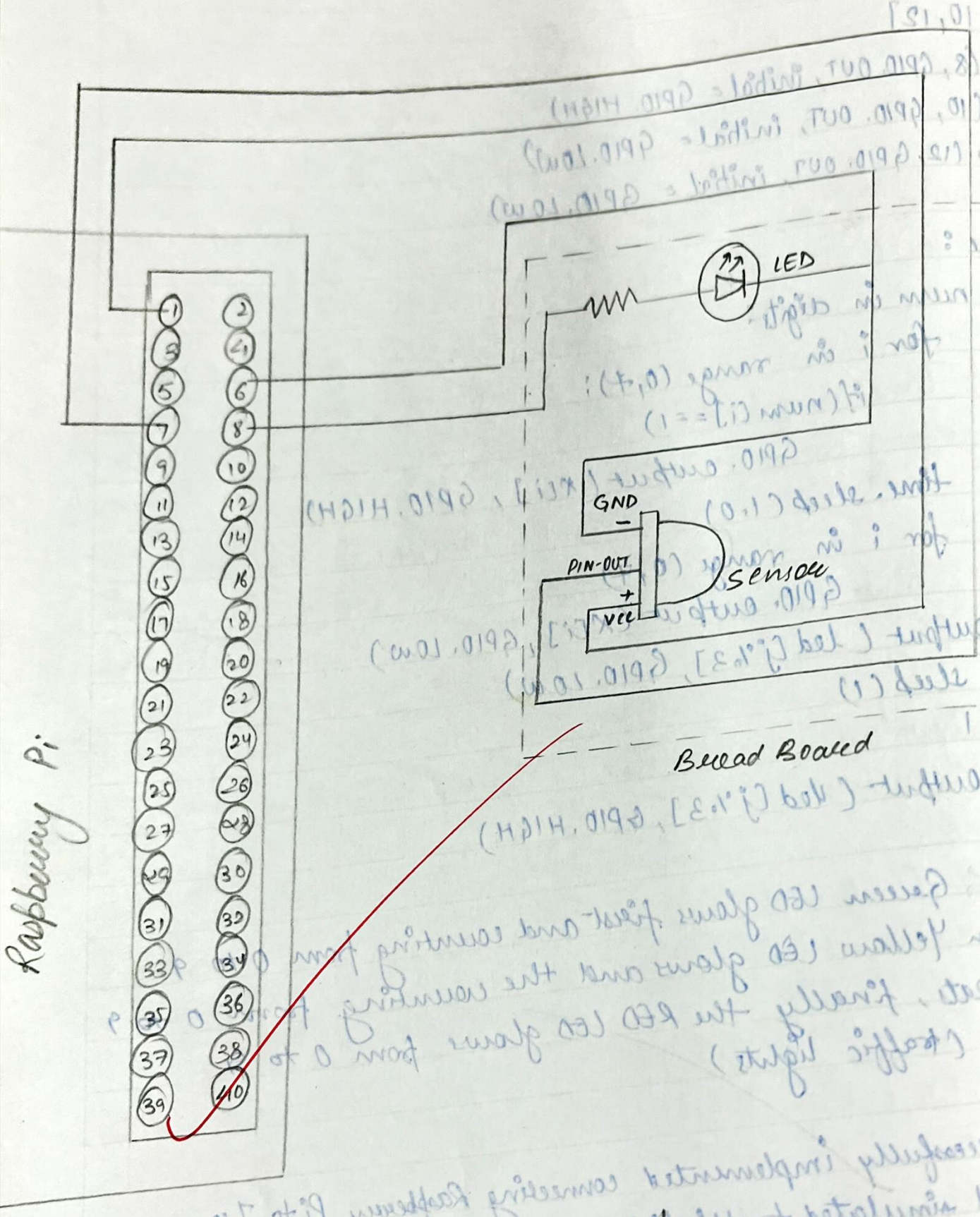
Expt No.
Date: / /


```
led = [8, 10, 12]
GPIO.setup(8, GPIO.OUT, initial = GPIO.HIGH)
GPIO.setup(10, GPIO.OUT, initial = GPIO.LOW)
GPIO.setup(12, GPIO.OUT, initial = GPIO.LOW)
j=0
while True:
    for num in digits:
        for i in range(0,7):
            if(num[i]==1)
                GPIO.output(x[i], GPIO.HIGH)
        time.sleep(1.0)
        for i in range(0,7)
            GPIO.output(x[i], GPIO.LOW)
    GPIO.output(led[j%3], GPIO.LOW)
    time.sleep(1)
    j=j+1
    GPIO.output(led[j%3], GPIO.HIGH)
```

OBSERVATION: Green LED glows first and counting from 0 to 9 starts then Yellow LED glows and the counting from 0 to 9 starts repeats, finally the RED LED glows from 0 to 9 counting. (traffic lights)

RESULT: Successfully implemented connecting Raspberry Pi to 7 segment display and simulated traffic light signals.

27/12



Circuit-Diagram

Expt No. 06
Date: 27/02/2023

Experiment No. 06

Interfacing PIR sensor with Raspberry Pi

AIM: To interface PIR (motion) sensor with Raspberry Pi to detect motion.

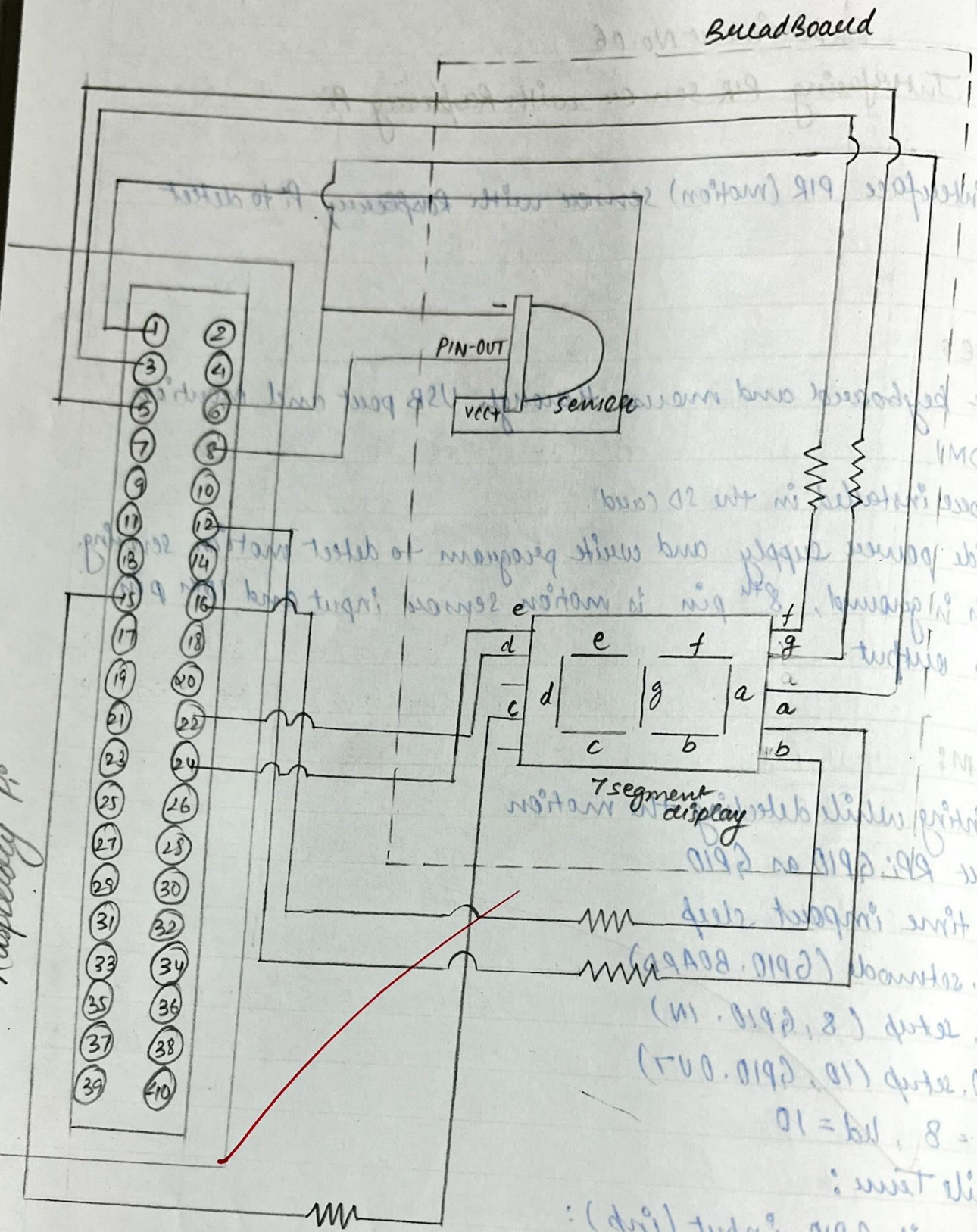
PROCEDURE:

* Connect keyboard and mouse through USB port and monitor using HDMI.
* OS is pre installed in the SD card.
* Provide power supply and write program to detect motion sensing.
* 6th pin is ground, 8th pin is motion sensor input and 10th pin is LED output.

PROGRAM:

1. Printing while detecting the motion

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8,GPIO.IN)
GPIO.setup(10,GPIO.OUT)
inp = 8, led = 10
while True:
    if GPIO.input(inp):
        GPIO.output(led, GPIO.HIGH)
        print("moving")
    else:
        GPIO.output(led, GPIO.LOW)
        print("Not moving")
sleep(1)
```



Circuit Diagram - 2

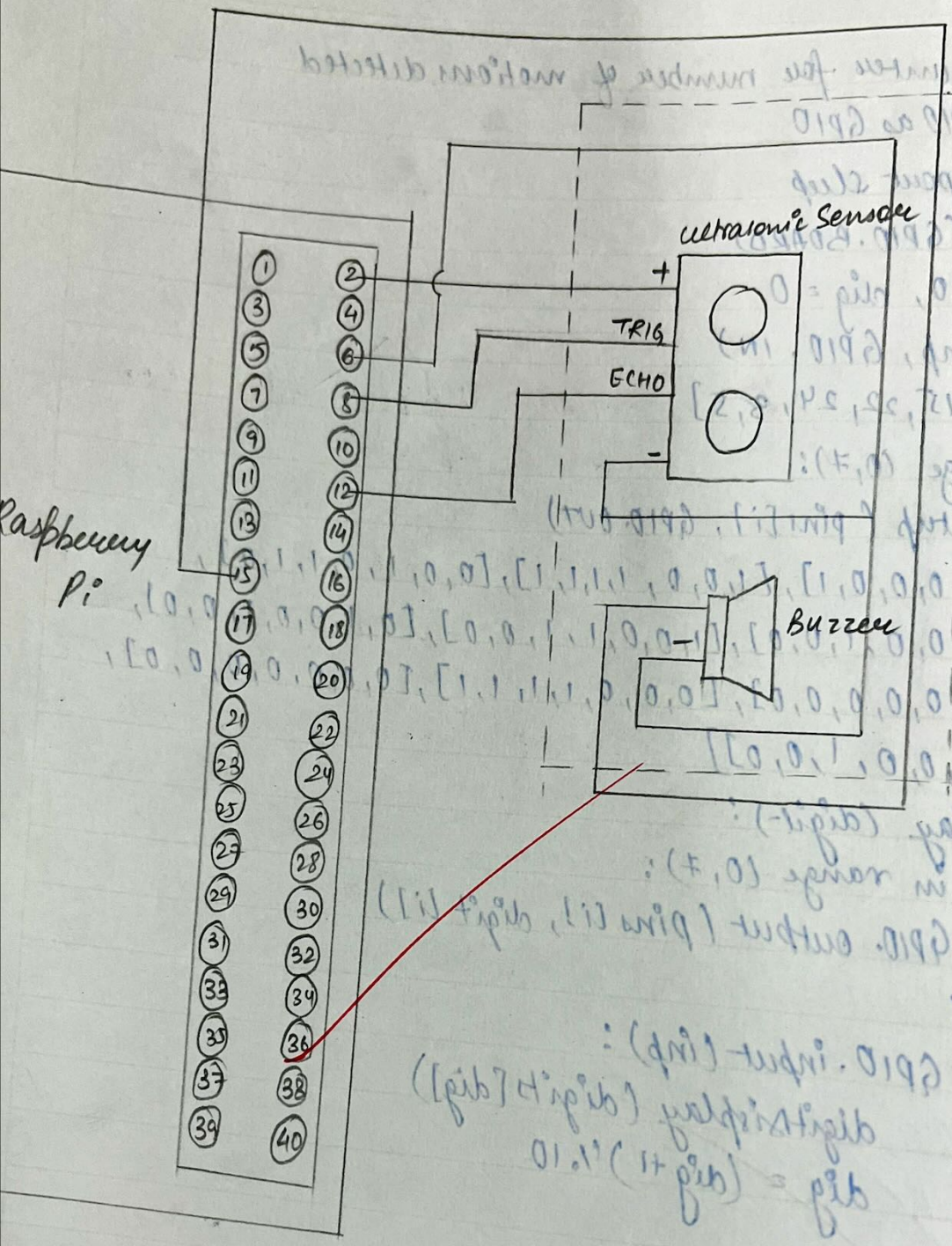
Expt No.
Date: / /

Page No. 20

2. 7 segment counter for number of motions detected

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
inp = 8, led = 10, dig = 0
GPIO.setup(inp, GPIO.IN)
pins = [12, 16, 15, 22, 24, 3, 5]
for i in range(0,7):
    GPIO.setup(pins[i], GPIO.OUT)
digits = [[0,0,0,0,0,0,1],[1,0,0,1,1,1,1],[0,0,1,0,1,1,0],
    [0,0,0,0,1,0,0],[1,0,0,1,1,0,0],[0,1,0,0,1,0,0],
    [0,1,0,0,0,0,0],[0,0,0,1,1,1,1],[0,0,0,0,0,0,0],
    [0,0,0,0,1,0,0]]
def digitDisplay(digit):
    for i in range(0,7):
        GPIO.output(pins[i], digit[i])
while True:
    if GPIO.input(inp):
        digitDisplay(digits[dig])
        dig = (dig+1)%10
sleep(1)
```

RESULT: In this experiment we successfully detected motion using PIR sensor interfaced with Raspberry Pi.



Circuit Diagram

Expt No. 07
Date: 06/03/2023

Page No. 21

Experiment No. 07

Interfacing with ultrasonic sensor

AIM: To find the distance between the ultrasonic sensor and the obstacle.

PROCEDURE:

* Connect keyboard and mouse through USB port and monitor using HDMI.
* OS is preinstalled in the SD card
* Provide power supply and write program to find the distance
* 6th pin is ground, 8th is trigger, 10th pin is buzzer, 12th pin is echo and 8th pin is also LED.

PROGRAM:

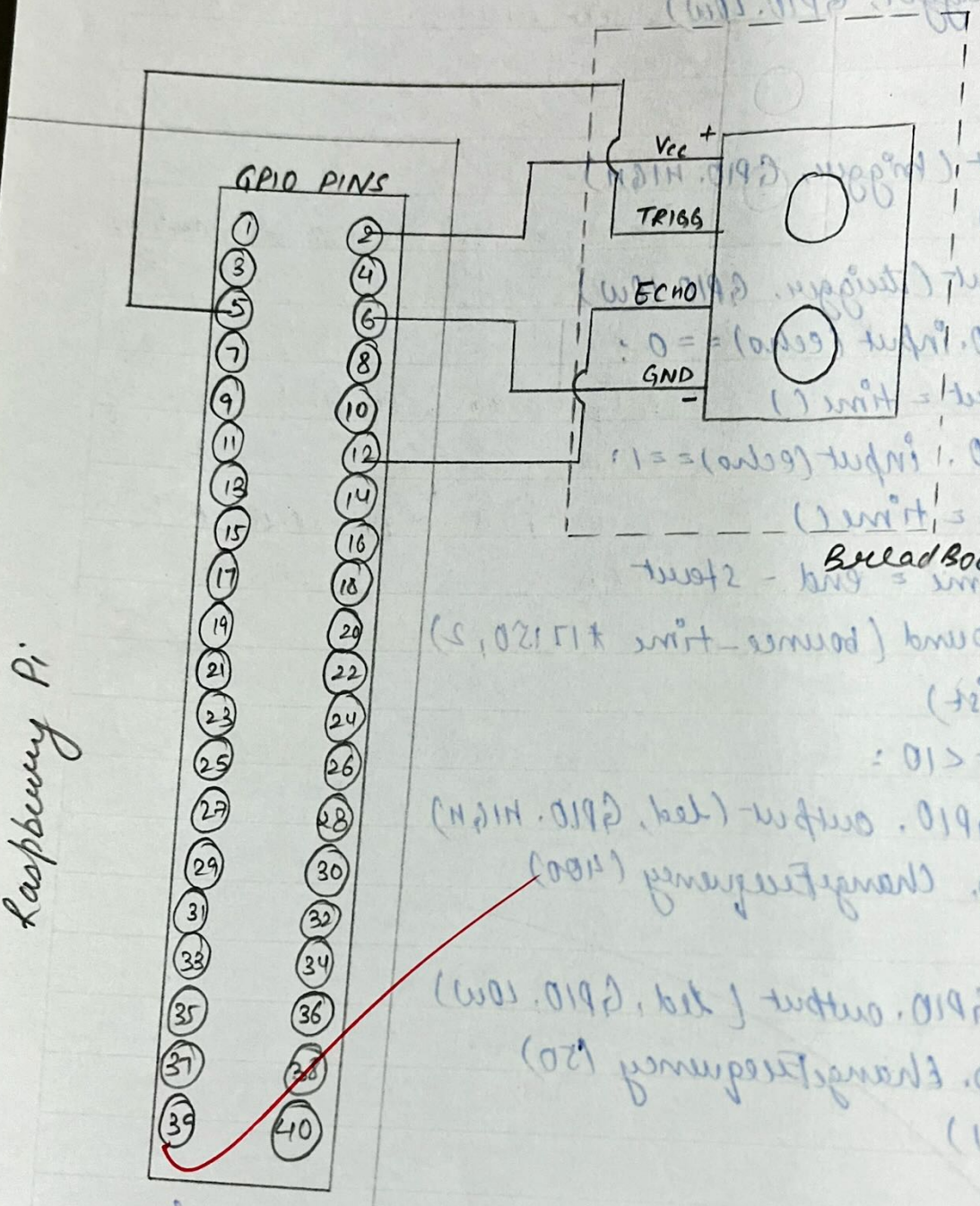
1. To find and print distance and also alert using LED and buzzer if distance is less than 10.

```
import RPi.GPIO as GPIO
from time import sleep, time
trigger = 5
led = 8
buzzer = 10
echo = 12
GPIO.setmode(GPIO.BOARD)
GPIO.setup(trigger, GPIO.OUT)
GPIO.setup(led, GPIO.OUT)
GPIO.setup(buzzer, GPIO.OUT)
GPIO.setup(echo, GPIO.OUT)
```

```
p = GPIO.PWM(led, 100)
p.start(50)
GPIO.output(trigger, GPIO.LOW)

while True:
    GPIO.output(trigger, GPIO.HIGH)
    sleep(0.01)
    GPIO.output(trigger, GPIO.LOW)
    while GPIO.input(echo)==0:
        start = time()
    while GPIO.input(echo)==1:
        end = time()
    bounce_time = end - start
    dist = round(bounce_time * 17150, 2)
    print(dist)
    if dist<10:
        GPIO.output(led, GPIO.HIGH)
        p.ChangeFrequency(400)
    else:
        GPIO.output(led, GPIO.LOW)
        p.ChangeFrequency(50)
    sleep(1)
```

12/3

RESULT: In this experiment, we successfully found the distance between ultrasonic sensor and the obstacle and altered the LEDs and altered the buzzer at a certain distance threshold.



Circuit Diagram

Expt No.
Date: 13/03/2023

EXPERIMENT NO. 08

Installing and Connecting SQLite in Raspberry Pi

AIM: To install and connect SQLite in Raspberry Pi and store the data from an ultrasonic sensor to the database.

PROCEDURE:

* Connect mouse and keyboard to USB ports of the Raspberry Pi.
* connect monitor to HDMI port of Raspberry Pi
* Connect to the internet using LAN or wifi. Provide power supply.
* Update the OS and other softwares using these 3 commands in the terminal:

```
$ sudo apt update
$ sudo apt-get update
$ sudo apt full-upgrade
```

* We might have to set system time before proceeding using the command

```
$ sudo date -s 'Mon Mar 13 14:45:00 IST 2023'
```

* Install Sqlite3 on Raspberry Pi using the command

```
$ sudo apt install sqlite3
```

* create a database and tables inside the database using the commands

```
$ sqlite3 sensor.db
$ create table distance(id int primary key auto increement,
                        dist real);
```

* Write the python program to connect to the database and store values in the same location. Run the program.

Expt No.
Date: / /

PROGRAMS:

1. To store the value into distance table.

```
import sqlite3
conn = sqlite3.connect('sensor.db')
c = conn.cursor()
c.execute("INSERT INTO distance (dist) values (10)")
c.commit()
```

OBSERVATION: the value gets inserted into the table. This can be verified by running "SELECT * FROM distance" in the sqlite terminal.

2. To measure distances using ultrasonic sensor and store the readings into the database.

```
import RPi.GPIO as GPIO
import sqlite3
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(5, GPIO.OUT, initial = GPIO.LOW)
GPIO.setup(12, GPIO.IN)
conn = sqlite3.connect("sensor.db")
c = conn.cursor()
while True:
    time.sleep(1)
    GPIO.output(5, GPIO.HIGH)
    time.sleep(0.01)
    GPIO.output(5, GPIO.LOW)
    start = end = time.time()
```

id	dist
1	23.5
2	23.8
3	24.5
4	24.7

Sample database output

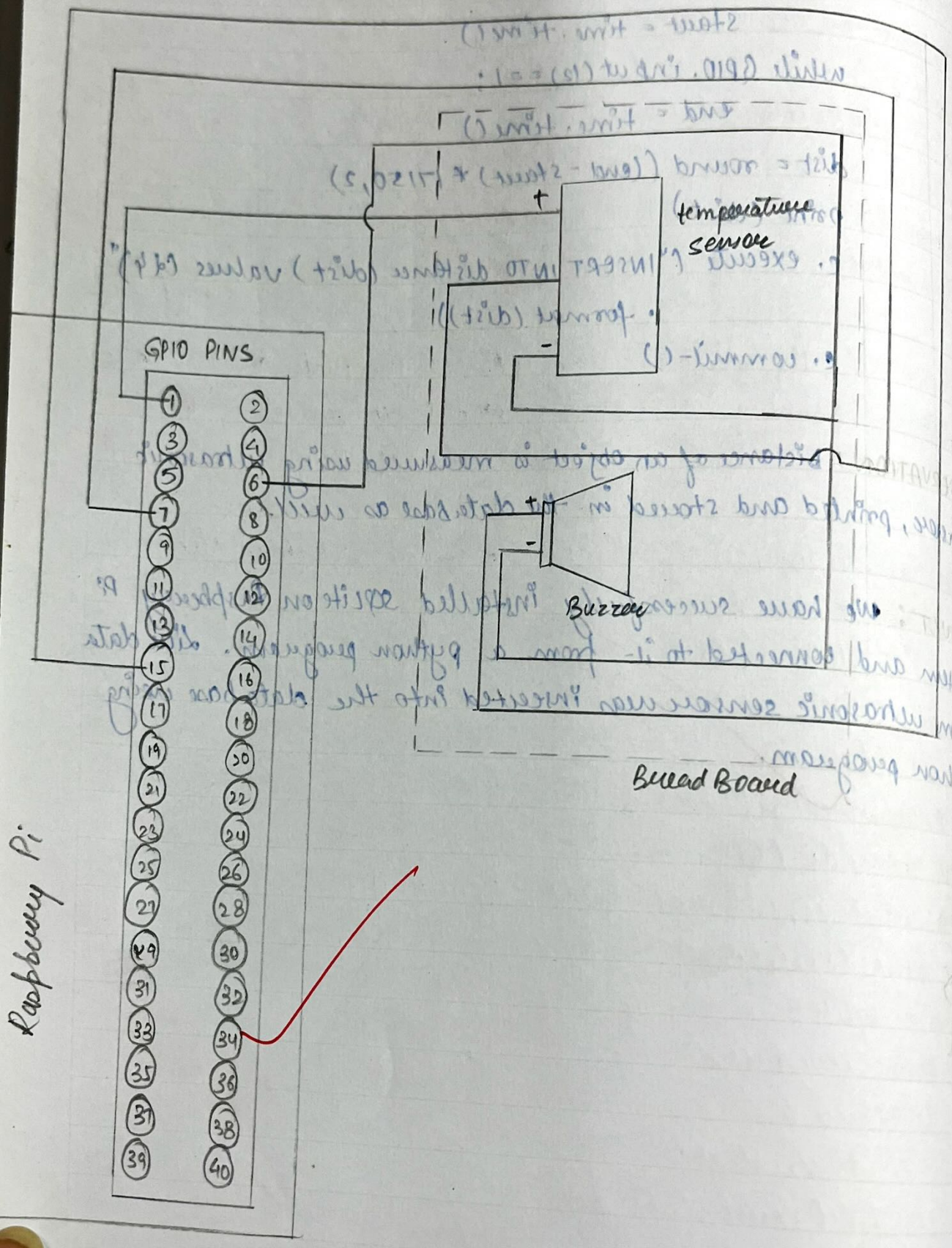
Sqlite db ?

```
while GPIO.input(12)==0:
        start = time.time()
while GPIO.input(12)==1:
        end = time.time()
dist = round((end-start)*17150,2)
print(dist)
c.execute("INSERT INTO distance(dist) values(%s)"
        .format(dist))
c.commit()
```

OBSERVATION: Distance of an object is measured using ultrasonic sensors, printed and stored in the database as well.

RESULT: We have successfully installed SQLite on Raspberry Pi system and connected to it from a python program. Live data from ultrasonic sensor was inserted into the database using python program.

20/5



Circuit-Diagram

Expt No. 09
Date: 20/03/2023

Experiment No: 09

Interfacing with temperature sensor

Aim: To detect temperature using sensor and store it in a database and activate a buzzer after a threshold.

PROCEDURE:

* Connect keyboard and mouse through USB port and monitor using HDMI cable.
* OS is preinstalled in the SD card
* Provide power supply and write the program to find the temperature and humidity using the connected temperature sensor, store the values in the database and activate the buzzer above a certain temperature.
* 6th pin is ground, GPIO4pin is temperature sensor input and 5th pin is buzzer input.

COMMANDS:

The following commands are executed before writing the program for creating the database:

```
$ sqlite3 temp.db
>> create table temperature (id integer primary key auto-
   increment, cels real);
>>.tables
```

PROGRAM:

```
import board
import adafruit_dht
import time
```

id	temp_f	temp_c	humidity
1	36	30	0.7
2	87.3	31	0.8
3	89.6	32	0.75

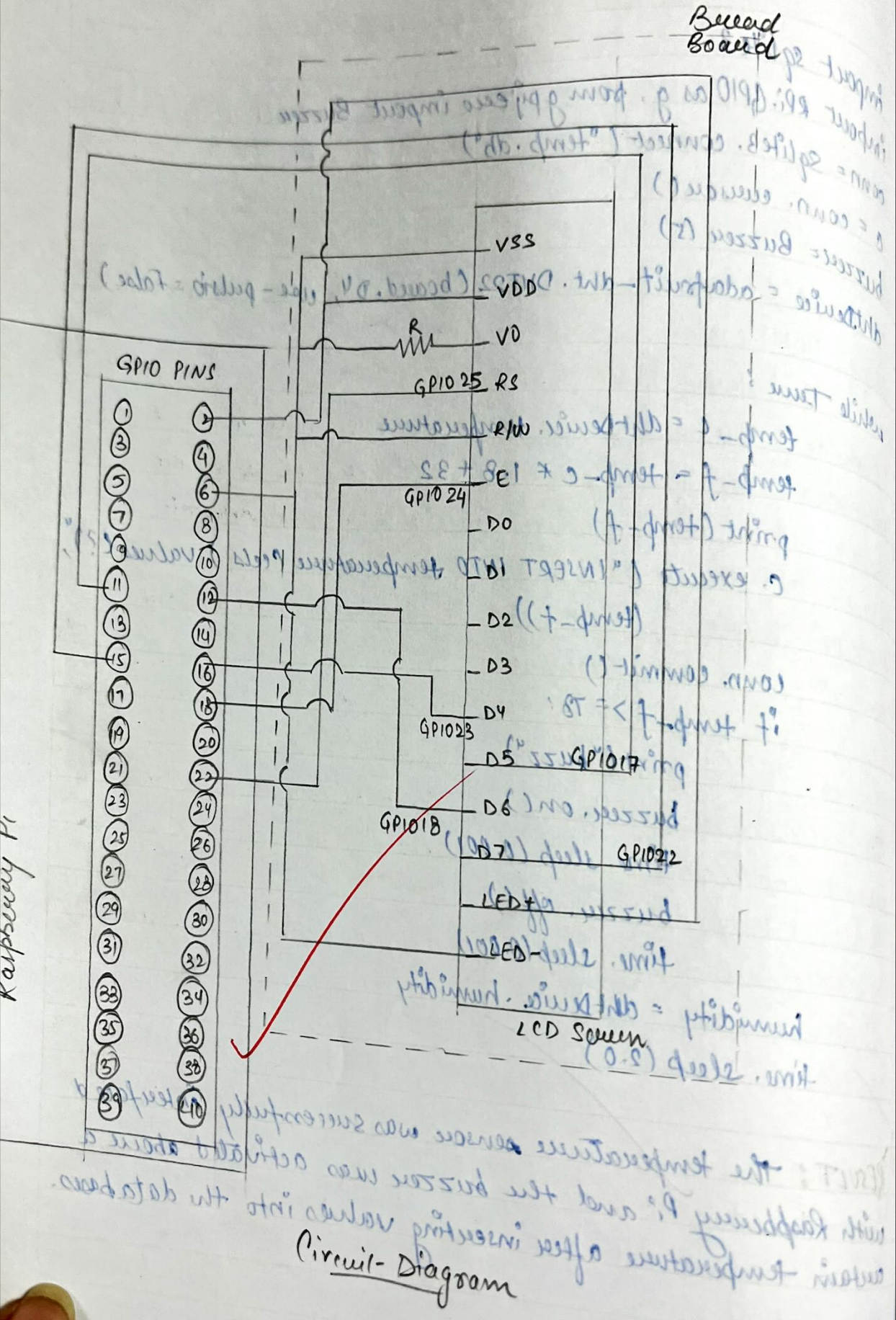
Sample database output

Expt No.
Date: / /

Page No. 27

```
import sqlite3
import RPi.GPIO as g. from gpiozero import Buzzer
conn = sqlite3.connect("temp.db")
c = conn.cursor()
buzzer = Buzzer(5)
dhtDevice = adafruit_dht.DHT22(board.D4, use_pulsio=False)

while True:
    temp_c = dhtDevice.temperature
    temp_f = temp_c * 1.8 + 32
    print(temp_f)
    c.execute("INSERT INTO temperature(cels) values(?)",
              (temp_f))
    conn.commit()
    if temp_f >= 78:
        print("buzz")
        buzzer.on()
        time.sleep(0.001)
        buzzer.off()
        time.sleep(0.001)
    humidity = dhtDevice.humidity
    time.sleep(2.0)
```

RESULT: The temperature sensor was successfully interfaced with Raspberry Pi and the buzzer was activated above a certain temperature after inserting values into the database.



Circuit-Diagram

Expt No. 10
Date: 27/03/2023

Page No. 28

Experiment No. 10

Interfacing 16X2 LCD with Raspberry Pi

AIM: Connecting Raspberry Pi to 16X2 LCD and writing program to display messages on the screen.

PROCEDURE:

* Connect keyboard and mouse through USB port and monitor using HDMI cord.
* OS is preinstalled in the SD card.
* Provide power supply and write the python program.
* Connect GPIO pins 26, 19, 25, 24, 22, 27, ~~20~~ LCD pins RS, EN, D4, D5, D6, D7 respectively.

PROGRAM:

```
import time
import board
import digitalio
import adafruit_character_lcd.character_lcd as
characterlcd

lcd_column = 16
lcd_row = 2

lcd_rs = digitalio.DigitalInOut(board.D26)
lcd_en = digitalio.DigitalInOut(board.D19)
lcd_d7 = digitalio.DigitalInOut(board.D27)
lcd_d6 = digitalio.DigitalInOut(board.D22)
lcd_d5 = digitalio.DigitalInOut(board.D24)
lcd_d4 = digitalio.DigitalInOut(board.D25)
```

lcd_backlight = digitalio.DigitalInOut(board.D4)

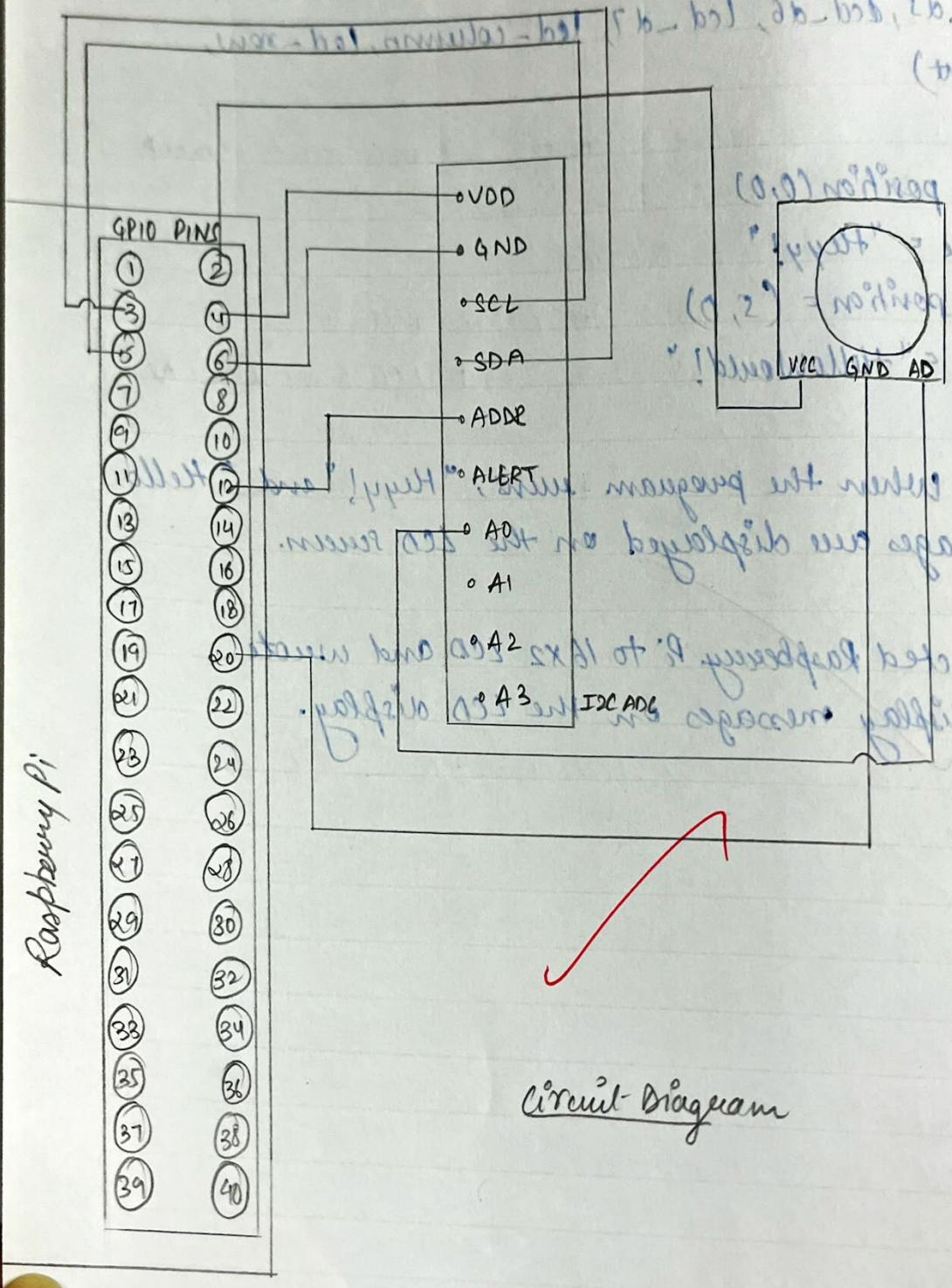
lcd = characterlcd.Character_LCD_Mono(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7, lcd_columns, lcd_rows, lcd_backlight)

lcd.cursor_position(0,0)
lcd.message = "Hey!"
lcd.cursor_position = (5,0)
lcd.message = "HelloWorld!"

OBSERVATION: When the program runs, "Hey!" and "Hello World!" messages are displayed on the LCD screen.

RESULT: Connected Raspberry Pi to 16x2 LCD and wrote program to display messages on the LCD display.

3/4



Circuit Diagram

Expt No. 11
Date: 3/4/2023

Page No. 30

Experiment No. 11

Interfacing Raspberry Pi with smoke sensor

AIM: Connecting Raspberry Pi to gas sensor and write program to detect smoke from Incense stick.

PROCEDURE:

* Connect mouse and keyboard to USB ports of Raspberry Pi
* Connect monitor to HDMI port of Raspberry Pi
* OS is preinstalled in the SD card
* Provide power supply and write program.
* Make circuit connections according to the circuit-diagram.

PROGRAM

→ Commands to be given on the terminal

- sudo raspi-config
* Go to interfacing options
* Select I2C
* choose yes to enable ARM I2C interface
* check /etc/modules
- sudo nano /etc/modules
* To ensure that eof has this line: i2c-dev
- sudo nano /boot/config.txt
* Uncomment dtparam = i2c_arm = on
- sudo apt-get install i2c-tools
  sudo reboot
  sudo i2c detect -y 1
* output should show 48 (40 rows, 8 columns)
- sudo pip3 install --upgrade setuptools
  pip3 install RPi.GPIO adafruit-blinka

32752	0.51177
32760	0.51180
32765	0.5185

Sample Output

sudo pip3 install adafruit-circuitpython-ads1x15

Python Code:

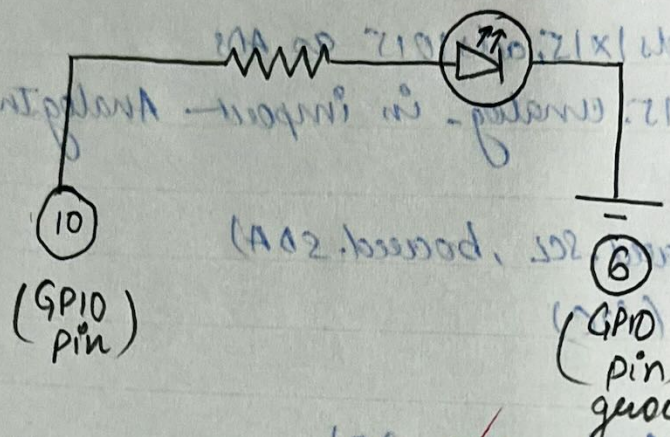
```
import time
import board
import busio
import adafruit_ads1x15.ads1015 as ADS
from adafruit_ads1x15.analog_in import AnalogIn

i2c = busio.I2C(board.SCL, board.SDA)
ads = ADS.ADS1015(i2c)
ads.gain = 8
chan = AnalogIn(ads, ADS.P0)
print("{:>5}\t{:>5}".format('raw', 'v'))
while True:
    print("{:>5}\t{:>5.3f}".format(chan.value, chan.voltage))
    time.sleep(0.5)
```

OBSERVATION:

As soon as the insense stick is placed near the detector the values are printed on screen (with increasing value as stick brought closer to sensor and vice-versa).

RESULT: Connected Raspberry Pi to smoke detector and wrote program to detect smoke.

16/4/23



Circuit Diagram

Expt No. 12
Date: 16/04/2023

Page No. 32

Experiment No. 12
Creating web interface for Raspberry Pi

Aim: To create web interface for raspberry Pi using node-red and toggling a LED using the interface.

Materials Required: Raspberry Pi, keyboard & mouse with USB cable, HDMI cable, USB-type C plug, breadboard, wires, LED, monitor, resistor.

Initial Steps:

* Connect the mouse and keyboard to the raspberry Pi using the USB ports. Then connect the USB type C plug to the USB power port of raspberry Pi. Connect plug with power socket.
* Connect the HDMI cable to the raspberry Pi at the micro HDMI port & connect the other end into the monitor.
* Turn the raspberry Pi on and open the terminal on the monitor.
* Now create the connections as shown in the circuit-diagram. Then follow the following instructions in the raspberry OS.

Procedure:

* Install nodeJS and node-red through the terminal.
* Start node-red by typing "node-red" in terminal.
* Open the node-red interface in browser using local host port 1880.

Expt No.
Date: / /

* install dashboard by going to Menu → manage pallate → "install" tab → dashboard.env → install
* create the web UI
  - Under dashboard on right side, create group & tab.
  - Drag and drop a switch from pallete and configure it.
  - Drag and drop GPIO from pallete and configure it.
* click deploy and open 127.0.0.1:1880/ui
* Connect LED and try controlling it through the UI.

Result: We are able to control and toggle the LED using the switch node in node red interface.