Expt 1(i): To interface Buzzer with Raspberry Pi and write a program to ‘turn ON’ LED for 1 sec after every 2 seconds. # connect J8A PIN-3,4 TO J8 PIN-3,4

import time

import RPi.GPIO as gpio bz\_board\_pin=38 bz\_bcm\_pin=20 gpio.setwarnings(False) gpio.setmode(gpio.BOARD)

gpio.setup(bz\_board\_pin, gpio.OUT) def bzBeep(ontime=1,offtime=2): gpio.output(bz\_board\_pin, gpio.HIGH) time.sleep(ontime) # Wait for ontime gpio.output(bz\_board\_pin, gpio.LOW) time.sleep(offtime)

return try:

while True:

bzBeep()

except KeyboardInterrupt: gpio.output(bz\_board\_pin, gpio.LOW) gpio.cleanup()

exit

Expt 1(ii): To interface Push button Raspberry Pi and write a program to ‘turn ON’

LED when push button is pressed

# Connect J4A PIN-1,3,5 TO J4 PIN-1,3,5

import time

import RPi.GPIO as gpio gpio.setwarnings(False) gpio.setmode(gpio.BOARD) led1 = 31

switch1 = 35 gpio.setup(led1,gpio.OUT,initial=gpio.LOW)

gpio.setup(switch1,gpio.IN) # Configure switch1 as input pin

def ledOnOff(event):

if event==switch1:

if gpio.input(switch1)==gpio.LOW:

gpio.output(led1, gpio.HIGH)

else:

gpio.output(led1, gpio.LOW)

gpio.add\_event\_detect(switch1, gpio.BOTH , callback = ledOnOff, bouncetime = 10) try:

while(True): time.sleep(1)

except KeyboardInterrupt: gpio.cleanup()

#Expt 2(i): To interface DHT11 sensor with Raspberry Pi and write a program to print temperature and humidity readings. import adafruit\_dht

import board

from time import sleep

dhtDevice = adafruit\_dht.DHT11(board.D4,use\_pulseio=False)

def readDHT(retries=5):

while retries:

try:

H=dhtDevice.humidity T=dhtDevice.temperature

if isinstance(H,int) and isinstance(T,int):

return H,T

except RuntimeError as error: print(error.args[0])

sleep(1) retries -= 1 continue

except Exception as error: dhtDevice.exit()

raise error

return(None,None)

if name == ' main ':

try:

while True:

humidity, temperature = readDHT()

if temperature != None and humidity != None: print('Humidity = {}%'.format(humidity)) print('Temperature = {}\u00B0C'.format(temperature)) else:

print('Unable to read the DHT sensor') sleep(2)

except KeyboardInterrupt:

pass

# Expt 2(ii): To interface OLED with Raspberry Pi and write a program to print temperature and humidity readings on it

import pi4dht11

import Adafruit\_GPIO.SPI as SPI

import Adafruit\_SSD1306

from datetime import datetime

from PIL import Image

from PIL import ImageDraw from PIL import ImageFont import time

RST = None

DC = 23

SPI\_PORT = 0

SPI\_DEVICE = 0

disp = Adafruit\_SSD1306.SSD1306\_128\_64(rst=RST,i2c\_address=0x3C) disp.begin()

disp.clear() disp.display() width = disp.width

height = disp.height

image = Image.new('1', (width, height)) draw = ImageDraw.Draw(image)

draw.rectangle((0,0,width,height), outline=0, fill=0) padding = -2

top = padding

bottom = height-padding

x = 0

font = ImageFont.load\_default()

try:

while True:

draw.rectangle((0,0,width,height), outline=0, fill=0) h,t = pi4dht11.readDHT()

if h==None and t==None:

h=''

t=''

else:

h=str(h) t=str(t)

d=datetime.now()

da= d.strftime("%d/%m/%Y") ti=d.strftime("%H:%M:%S")

draw.text((x, top),"Humidity: "+ h+ "%",font=font, fill=255) draw.text((x, top+12), "Temperature: " + t + "\u00B0C", font=font, fill=255) draw.text((x, top+24), "Date: "+da, font=font, fill=255) draw.text((x, top+36), "Time: "+ti, font=font, fill=255) disp.image(image)

disp.display() time.sleep(2)

except KeyboardInterrupt:

pass

*# Expt 3: To interface motor using relay with Raspberry Pi and write a program to ‘turn ON’ motor when push button is pressed.*

import time

import RPi.GPIO as gpio gpio.setwarnings(False) gpio.setmode(gpio.BCM) relay1 = 16

switch1 = 19 gpio.setup(relay1,gpio.OUT,initial=gpio.LOW) gpio.setup(switch1,gpio.IN)

def toggleRelay(event):

f event==switch1:

if gpio.input(switch1)==gpio.LOW:

if gpio.input(relay1):

gpio.output(relay1, gpio.LOW)

else:

gpio.output(relay1, gpio.HIGH)

gpio.add\_event\_detect(switch1, gpio.BOTH , callback = toggleRelay, bouncetime = 35)

try:

while(True):

time.sleep(1)

except KeyboardInterrupt:

gpio.cleanup()



*# Expt\_5: To interface Bluetooth with Raspberry Pi and write a program to turn LED ON/OFF when '1'/'0' is received from smartphone using Bluetooth.*

import RPi.GPIO as gpio

import time

from datetime import datetime gpio.setwarnings(False) gpio.setmode(gpio.BCM)

led1 = 6 gpio.setup(led1,gpio.OUT,initial=gpio.HIGH) import serial s=serial.Serial("/dev/rfcomm0",9600,timeout=1) commands=('1','0') # Valid commands

try:

while True: inp=s.readline().decode('utf8').strip().lower() print(inp)

if inp in commands:

if inp=='1':

gpio.output(led1, gpio.HIGH) print("LED is Turned ON")

s.write('LED is Turned ON\n'.encode('ascii'))

if inp=='0':

gpio.output(led1, gpio.LOW) print("LED is Turned OFF")

s.write('LED is Turned OFF\n'.encode('ascii'))

else: time.sleep(1)

except KeyboardInterrupt:

pass

#Expt 4: To interface Bluetooth with Raspberry Pi and write a program to send sensor data to smartphone using Bluetooth.

import pi4dht11

import time

from datetime import datetime

import serial s=serial.Serial("/dev/rfcomm0",9600,timeout=1) import Adafruit\_GPIO.SPI as SPI

import Adafruit\_MCP3008 SPI\_PORT = 0

SPI\_DEVICE = 0

mcp = Adafruit\_MCP3008.MCP3008(spi=SPI.SpiDev(SPI\_PORT, SPI\_DEVICE))

def send\_bluetooth\_dht():

h,t = pi4dht11.readDHT() time=datetime.now().strftime('%H:%M:%S') if h != None or t != None:

s.write("Temparature at {} is {}\n".format(time,t).encode('ascii')) s.write("Humidity at {} is {}\n".format(time,h).encode('ascii')) print('Humidity = {} ; Temeprature = {}'.format(h,t))

else:

s.write('Unable to read from sensor at time {}\n'.format(time).encode('ascii')) print('Unable to red from sensor at time {}'.format(time))

def send\_bluetooth\_light():

lightvalue=mcp.read\_adc(1) time=datetime.now().strftime('%H:%M:%S')

s.write('Light value at {}: {}\n'.format(time,lightvalue).encode('ascii')) print('Light value at {}: {}'.format(time,lightvalue)) commands=('light','dht','')

try:

while True: inp=s.readline().decode('utf8').strip().lower() print(inp)

if inp in commands:

if inp=='dht':

send\_bluetooth\_dht()

if inp=='light':

send\_bluetooth\_light()

else:

print('Invalid command word')

s.write('Invalid command word\n'.encode('ascii')) time.sleep(2)

except KeyboardInterrupt:

pass

#Expt 6: Write a program on Raspberry Pi to upload temperature and humidity data to thingspeak cloud.

import pi4dht11 import urllib.request import time

WRITE\_API\_KEY = "UEEPPRM8KV7LPLUB"

baseurl = f"https://api.thingspeak.com/update?api\_key=

{WRITE\_API\_KEY}" H,T = pi4dht11.readDHT()

*1*if H != None and T != None:

try:

f = urllib.request.urlopen(baseurl + f'&field1={T}&field2={H}') f.close()

print(f'Humidity = {H}%') print(f'Temperature = {T}\u00B0C')

print("Uploaded to Thingspeak Successfully")

except:

print('Not successful in uploading to Thingspeak.com...Exiting..')

else: print("Sensor reading error occured")

# Expt 7: Write a program on Raspberry Pi to retrieve temperature and humidity data from thingspeak cloud

# Connect J8A PIN-3,4 TO J8 PIN-3,4 for Buzzer Connection

import json

from gpiozero import Buzzer

import time

from urllib import request

from datetime import datetime bz=Buzzer(20)

def datetime\_from\_utc\_to\_local(utc\_datetime):

now\_timestamp = time.time()

offset = datetime.fromtimestamp(now\_timestamp) - datetime.utcfromtimestamp(now\_timestamp)

dt = utc\_datetime + offset date1 = dt.strftime('%d-%m-%Y') time1 = dt.strftime('%H:%M:%S') return(date1,time1)

READ\_API\_KEY='W8BJHXHG4WFBV8KJ' # Modify READ\_API\_KEY CHANNEL\_ID='2412318'

url = f"<http://api.thingspeak.com/channels/>{CHANNEL\_ID}/feeds/last.json? api\_key= {READ\_API\_KEY}"

connection = request.urlopen(url) response = connection.read() data = json.loads(response) temperature = data['field1'] humidity = data['field2'] timeStamp = data['created\_at'] print(timeStamp)

ts = datetime.fromisoformat(timeStamp[:-1]).astimezone()#timezone.utc) Date, Time = datetime\_from\_utc\_to\_local(ts)

print(f'Date: {Date}\nTime: {Time}\nTemperature: \

{temperature}\u00B0C\nHumidity: {humidity}%')

if eval(temperature) < 20: bz.beep(n=1)

elif eval(temperature) < 30: bz.beep(n=2)

elif eval(temperature) < 40: bz.beep(n=3)

else: bz.beep(n=4) time.sleep(8) bz.close()