**PRATICAL FOR IOT**

**Q1)Displaying Time over 4 Digit 7 Segment Display using Raspberry Pi  
CODE:**

**import sys**

**import**

**RPi.GPIO as GPIO import time, datetime**

**import tm1637**

**Display= tm1637.TM1637(23,24,tm1637.BRIGHT\_TYPICAL)**

**Display.Clear()**

**Display.SetBrightnes(1) while(True):**

**now = datetime.datetime.now()**

**hour = now.hour**

**minute=now.minute**

**second=now.second**

**currenttime=[int(hour/10),hour%10,int(minute/10),minute%10 ]**

**Display.Show(currenttime)**

**Display.ShowDoublepoint(second%2)**

**time.sleep(1)**

**Q2)Capturing Images & CLIPS with Raspberry Pi and Pi Camera**

**//Camera Image Capture**

from time import sleep

from picamera import PiCamera

camera = PiCamera()

camera.resolution = (1280, 720)

camera.start\_preview()

# Camera warm-up time

sleep(2)

camera.capture('/home/pi/Pictures/newImage.jpg')

camera.stop\_preview()

**//Camera Video**

from time import sleep

from picamera import PiCamera

camera = PiCamera()

camera.resolution = (1280, 720)

camera.start\_preview()

camera.start\_recording('/home/pi/video.h264')

sleep(10)

camera.stop\_recording()

camera.stop\_preview()

Q3) ***TELEGRAM***

import sys

import time

import random

import datetime

import telepot

import RPi.GPIO as GPIO

from telepot.loop import MessageLoop

red=23 # connect red led at pin 23 of raspberry pi

now=datetime.datetime.now()

GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)

GPIO.setup(red,GPIO.OUT)

GPIO.output(red,0)

def action(msg):

chat\_id = msg['chat']['id']

command = msg['text']

print ('Got command: %s' %command)

if 'On' in command:

message="Turn On"

message=message+" red"

GPIO.output(red,1)

bot.sendMessage(chat\_id,message)

if 'Off' in command:

message="Turn Off"

message=message+" red"

GPIO.output(red,0)

bot.sendMessage(chat\_id, message)

bot = telepot.Bot('626665131:AAHsNzQbqSj9GZ9-w2t4I') #paste your bot tokan here.

print(bot.getMe())

MessageLoop(bot,action).run\_as\_thread()

print ('I am listening...')

while 1:

time.sleep(10)  
  
Q4) ***BLINKING LED***

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BOARD) #set the Pin mode you will be working with

led1 = 29

led2 = 31

led3 = 33

led4 = 35

led5 = 36

led6 = 37

led7 = 38

led8 = 40

#setup the ledPin(i.e. GPIO22) as output

GPIO.setup(led1, GPIO.OUT)

GPIO.setup(led2, GPIO.OUT)

GPIO.setup(led3, GPIO.OUT)

GPIO.setup(led4, GPIO.OUT)

GPIO.setup(led5, GPIO.OUT)

GPIO.setup(led6, GPIO.OUT)

GPIO.setup(led7, GPIO.OUT)

GPIO.setup(led8, GPIO.OUT)

GPIO.output(led1, False)

GPIO.output(led2, False)

GPIO.output(led3, False)

GPIO.output(led4, False)

GPIO.output(led5, False)

GPIO.output(led6, False)

GPIO.output(led7, False)

GPIO.output(led8, False)

def ledpattern(ledVal1, ledVal2, ledVal3, ledVal4, ledVal5, ledVal6, ledVal7,ledVal8):

GPIO.output(led1, ledVal1)

GPIO.output(led2, ledVal2)

GPIO.output(led3, ledVal3)

GPIO.output(led4, ledVal4)

GPIO.output(led5, ledVal5)

GPIO.output(led6, ledVal6)

GPIO.output(led7, ledVal7)

GPIO.output(led8, ledVal8)

def patterOne():

for i in range (0, 3):

ledpattern(1, 0, 1, 0, 1, 0, 1, 0)

time.sleep(1)

ledpattern(0, 1, 0, 1, 0, 1, 0, 1)

time.sleep(1)

def patternTwo():

for i in range (0, 5):

ledpattern(1, 0, 0, 0, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 1, 0, 0, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 1, 0, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 1, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 1, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 0, 1, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 0, 0, 1, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 0, 0, 0, 1)

time.sleep(0.1)

def patternThree():

for i in range (0, 5):

ledpattern(0, 0, 0, 0, 0, 0, 0, 1)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 0, 0, 1, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 0, 1, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 0, 1, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 0, 1, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 0, 1, 0, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(0, 1, 0, 0, 0, 0, 0, 0)

time.sleep(0.1)

ledpattern(1, 0, 0, 0, 0, 0, 0, 0)

time.sleep(0.1)

def patternFour():

for i in range (0, 5):

ledpattern(0, 1, 1, 1, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 0, 1, 1, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 0, 1, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 0, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 0, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 1, 0, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 1, 1, 0, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 1, 1, 1, 0)

time.sleep(0.1)

def patternFive():

for i in range (0, 5):

ledpattern(1, 1, 1, 1, 1, 1, 1, 0)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 1, 1, 0, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 1, 0, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 1, 0, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 1, 0, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 1, 0, 1, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(1, 0, 1, 1, 1, 1, 1, 1)

time.sleep(0.1)

ledpattern(0, 1, 1, 1, 1, 1, 1, 1)

time.sleep(0.1)

try:

while True:

patterOne()

patternTwo()

patternThree()

patternFour()

patternFive()

finally:

#reset the GPIO Pins

GPIO.cleanup()  
  
  
***Q5 – RFID***

import RPi.GPIO as GPIO

import time

import serial

def read\_rfid ():

ser = serial.Serial ("/dev/ttyUSB0")

ser.baudrate = 9600

data = ser.read(12)

ser.close ()

return data

try:

while True:

id = read\_rfid ()

print (id)

if id=="40002C3ECB99":

print("Access Granted")

else:

print("Access Denied")

finally:

GPIO.cleanup()

***Q 6 – VISITORS MODULE***

import RPi.GPIO as gpio

import picamera

import time

m11=17

m12=27

led=5

buz=26

button=19

HIGH=1

LOW=0

gpio.setwarnings(False)

gpio.setmode(gpio.BCM)

gpio.setup(led, gpio.OUT)

gpio.setup(buz, gpio.OUT)

gpio.setup(m11, gpio.OUT)

gpio.setup(m12, gpio.OUT)

gpio.setup(button, gpio.IN)

gpio.output(led , 0)

gpio.output(buz , 0)

gpio.output(m11 , 0)

gpio.output(m12 , 0)

data=""

def capture\_image():

print("Please Wait..");

data= time.strftime("%d\_%b\_%Y\%H:%M:%S")

camera.start\_preview()

time.sleep(5)

print (data)

camera.capture('/home/pi/Desktop/Visitors/%s.jpg'%data)

camera.stop\_preview()

print("Image Captured Successfully")

time.sleep(2)

def gate():

print(" Welcome ")

gpio.output(m11, 1)

gpio.output(m12, 0)

time.sleep(1.5)

gpio.output(m11, 0)

gpio.output(m12, 0)

time.sleep(3)

gpio.output(m11, 0)

gpio.output(m12, 1)

time.sleep(1.5)

gpio.output(m11, 0)

gpio.output(m12, 0)

print(" Thank You ")

time.sleep(2)

print("Visitor Monitoring")

print(" Using RPI ")

time.sleep(3)

camera = picamera.PiCamera()

camera.rotation=180

camera.awb\_mode= 'auto'

camera.brightness=55

time.sleep(2)

while 1:

#d= time.strftime("%d %b %Y")

#t= time.strftime("%H:%M:%S")

#print("Time: %s"%t)

#print("Date:%s"%d)

print(" Please Press Button")

print(" to open the gate ")

gpio.output(led, 1)

if gpio.input(button)==0:

gpio.output(buz, 1)

gpio.output(led, 0)

time.sleep(0.5)

gpio.output(buz, 0)

capture\_image()

gate()

time.sleep(0.5)

***Q 7 – OSCILLOSCOPE***

import time

import matplotlib.pyplot as plt

#import numpy

from drawnow import \*

# Import the ADS1x15 module.

import Adafruit\_ADS1x15

# Create an ADS1115 ADC (16-bit) instance.

adc = Adafruit\_ADS1x15.ADS1115()

GAIN = 1

val = [ ]

cnt = 0

plt.ion()

# Start continuous ADC conversions on channel 0 using the previous gain value.

adc.start\_adc(0, gain=GAIN)

print('Reading ADS1x15 channel 0')

#create the figure function

def makeFig():

plt.ylim(-5000,5000)

plt.title('Osciloscope')

plt.grid(True)

plt.ylabel('ADC outputs')

plt.plot(val, 'ro-', label='Channel 0')

plt.legend(loc='lower right')

while (True):

# Read the last ADC conversion value and print it out.

value = adc.get\_last\_result()

print('Channel 0: {0}'.format(value))

# Sleep for half a second.

time.sleep(0.5)

val.append(int(value))

drawnow(makeFig)

plt.pause(.000001)

cnt = cnt+1

if(cnt>50):

val.pop(0)