**Title : DOOR BELL NOTIFIER FOR HOME SECURITY SYSTEM USING BEAGLEBONE BLACK.**

**Group Members:**

1. Sneh Patel (201301132)
2. Saloni Jain (201301134)-Leader
3. Zeel Patel (201301135)
4. Devesh Singh (201301136)
5. Ankit Lakra (201301137)

**Deliverables:**

The purpose of this project is to make a doorbell notifier system using beaglebone black. This project works as follows:

1. At very first step, there will be a visitor who will stand in front of the door.The visitor can be either the person of interest or can be an intruder. This project is ask the authority to take an action for addressing the visitor either by allowing him to enter or by sending him back and asking him to come later or playing a siren to make neighbours aware of an intruder.
2. This is the first case when visitor is the whom we want to address. Webcam and Ultrasonic sensor Hc-sr04 are placed on the servomotor which is rotating 180 degrees. As soon as a visitor comes, Hc-sr04 will detect his/her presence and will supply low voltage to servo motor's control pin thereby, stopping it and asking webcam to take picture of the visitor. Now beaglebone will send the picture to the authority’s gmail account. And authority will reply back with “YES” which will open our electric door lock.
3. This is the second case when visitor himself/herself is the authority or some family member. A keypad is placed at the door which is set to a security password or pin. As soon as authority comes Hc-sr04 will detect his/her presence and will send the picture to the gmail of authority. Now authority can enter the pin which if correct will automatically open the door. If the visitor is not the authority he/she can make 3 attempts to try out pin after which siren will play if pin doesn’t matches.
4. This is the third case when we don’t want to attend the visitor may be due to some work or absence at home. In this case also Hc-sr04 will detect presence of visitor and webcam will take the picture. Then picture will be send to the authority. Now authority will revert back saying “NO x” where x=1,2,3,4. Here “No” is sent for not opening the lock and “x” is the already stored message in beaglebone black. These messages are stored by authority only like “I am busy, come later”. An audio will be played for the visitor depending on the number selected by authority.
5. This is the final case when the visitor is an intruder. As soon as Hc-sr04 detects his/her presence, servo motor will stop and webcam will take the picture. A window of 40 seconds will start as soon as Hc-sr04 detects the presence. If at the end of 40 seconds visitor neither presses the doorbell nor the number pad then at the end of 40 seconds , webcam will again take a picture and it will be send to the gmail account of authority after which a siren will start playing. Along with the picture , a message will be send "ALERT!!! INTRUDER IN FRONT OF YOUR HOUSE" to the gmail account of authority to make him/her alert.
6. The final listing is:
   1. Stopping servo motor on high of doorbell (when bell is pressed).
   2. Stopping servo motor on high of Hc-sr04 Ultrasonic range sensor.
   3. Taking picture using webcam and sending it to gmail account of authority.
   4. Reverting back to beaglebone by saying “YES” or “NO”.
   5. To play audio message stored in beaglebone black .
   6. Opening the door lock depending on “YES” or “NO”.

**Code:**

1. **Doorbell.**

import Adafruit\_BBIO.GPIO as GPIO

import time

import os

import sys

GPIO.setup("P9\_15", GPIO.IN)

#flag = 0

#print GPIO.input("P9\_15")

while True:

if GPIO.input("P9\_15")==1:

#print "1"

#flag = 1

sys.exit(1)

break

else:

sys.exit(0)

break

1. **Servo motor.**

import Adafruit\_BBIO.PWM as PWM  
 servoPin="P9\_22"  
 PWM.start(servoPin,2,50)  
 while(1):  
 desiredAngle=input("what angle do you want")  
 dutyCycle=1./18.\*desiredAngle + 2  
 PWM.set\_duty\_cycle(servoPin,dutyCycle)

1. **Ultrasonic Sensor Hc-sr04.**

import time  
 import Adafruit\_BBIO.GPIO as GPIO  
  
 #in cm  
 threshold = 180  
  
 # Define GPIO to use on Pi  
 GPIO\_TRIGGER = 'P9\_11'  
 GPIO\_ECHO = 'P9\_12'  
  
 # Set pins as output and input  
 GPIO.setup(GPIO\_TRIGGER,GPIO.OUT) # Trigger  
 GPIO.setup(GPIO\_ECHO,GPIO.IN) # Echo  
  
  
 while True:  
 # Set trigger to False (Low)  
 GPIO.output(GPIO\_TRIGGER, GPIO.LOW)  
  
 # Allow module to settle  
 time.sleep(0.5)  
  
 # Send 10us pulse to trigger  
 GPIO.output(GPIO\_TRIGGER, GPIO.HIGH)  
 time.sleep(0.00001)  
 GPIO.output(GPIO\_TRIGGER, GPIO.LOW)  
 start = time.time()  
 print GPIO.input(GPIO\_ECHO)  
 while GPIO.input(GPIO\_ECHO)==0:  
 start = time.time()  
  
 print GPIO.input(GPIO\_ECHO)  
 while GPIO.input(GPIO\_ECHO)==1:  
 stop = time.time()  
  
 print start  
 print stop  
 # Calculate pulse length  
 elapsed = stop-start  
  
 # Distance pulse travelled in that time is time  
 # multiplied by the speed of sound (cm/s)  
 distance = elapsed \* 34000  
   
 # That was the distance there and back so halve the value  
 distance = distance / 2  
  
 print "Distance : %.1f" % distance  
  
 if distance<threshold:  
 break  
  
 # Reset GPIO settings  
 GPIO.cleanup()

1. **Webcam image.**

from cv2 import \*  
 # initialize the camera  
 cam = VideoCapture(0) # 0 -> index of camera  
 s, img = cam.read()  
 if s: # frame captured without any errors  
 namedWindow("cam-test",CV\_WINDOW\_AUTOSIZE)  
 imshow("cam-test",img)  
 # waitKey(0)  
 destroyWindow("cam-test")  
 imwrite("filename.png",img) #save image

1. **Send an email to gmail account of the authority.**

hostname = "smtp.gmail.com"  
 password = "ifqdxcimpptxwmul"  
 me = "bbb2daiict@gmail.com"  
 you = "ankitlakra137@gmail.com"  
   
 import smtplib  
 #from email.mime.text import MIMEText  
 from email.mime.image import MIMEImage  
 from email.mime.multipart import MIMEMultipart  
  
 msg = MIMEMultipart()  
   
 msg["Subject"] = "Test subject"  
 msg["From"] = me  
 msg["To"] = you  
 #msg.attach(MIMEText(file("gmail1.py").read())  
 fp = open('filename.png', 'rb')  
 img = MIMEImage(fp.read())  
 fp.close()  
 msg.attach(img)  
  
 s = smtplib.SMTP\_SSL(hostname)  
 s.login(me, password)  
 s.sendmail(me, [you], msg.as\_string())  
 s.quit()

1. **Reading the mail sent by the authority to open the door lock.**

#!/usr/bin/env python  
 #  
 # Very basic example of using Python and IMAP to iterate over emails in a  
 # gmail folder/label. This code is released into the public domain.  
 #  
 # RKI July 2013  
 # http://www.voidynullness.net/blog/2013/07/25/gmail-email-with-python-via-imap/  
 #  
 import sys  
 import imaplib  
 import getpass  
 import email  
 import email.header  
 import datetime  
  
 EMAIL\_ACCOUNT = "bbb2daiict@gmail.com"  
 EMAIL\_PASSWORD = "ifqdxcimpptxwmul"  
 EMAIL\_FOLDER = "INBOX"  
  
 prev\_count = 0  
 curr\_count = 0  
  
  
 def process\_mailbox(M):  
 """  
 Do something with emails messages in the folder.   
 For the sake of this example, print some headers.  
 """  
  
 rv, data = M.search(None, "ALL")  
 if rv != 'OK':  
 print "No messages found!"  
 return  
  
 global prev\_count  
 global curr\_count  
  
 for num in data[0].split():  
 curr\_count = curr\_count+1  
 #print prev\_count  
 #print curr\_count  
 if prev\_count<curr\_count:  
 num = curr\_count  
 rv, data = M.fetch(num, '(RFC822)')  
 if rv != 'OK':  
 print "ERROR getting message", num  
 return  
  
 msg = email.message\_from\_string(data[0][1])  
 decode = email.header.decode\_header(msg['Subject'])[0]  
 subject = unicode(decode[0])  
 if subject=="Yes":  
 sys.exit(1)  
 """  
 print 'Message %s: %s' % (num, subject)  
 print 'Raw Date:', msg['Date']  
 # Now convert to local date-time  
 date\_tuple = email.utils.parsedate\_tz(msg['Date'])  
 if date\_tuple:  
 local\_date = datetime.datetime.fromtimestamp(  
 email.utils.mktime\_tz(date\_tuple))  
 print "Local Date:", \  
 local\_date.strftime("%a, %d %b %Y %H:%M:%S")  
 """  
 elif subject=="No" or subject=="No 1":  
 sys.exit(2)  
 prev\_count = curr\_count  
 curr\_count = 0  
  
  
 while True:  
 M = imaplib.IMAP4\_SSL('imap.gmail.com')  
  
 try:  
 rv, data = M.login(EMAIL\_ACCOUNT, EMAIL\_PASSWORD)  
 except imaplib.IMAP4.error:  
 print "LOGIN FAILED!!! "  
 sys.exit(1)  
  
 #print rv, data  
  
 rv, mailboxes = M.list()  
 """  
 if rv == 'OK':  
 print "Mailboxes:"  
 print mailboxes  
 """  
  
 rv, data = M.select(EMAIL\_FOLDER)  
 if rv == 'OK':  
 #print "Processing mailbox...\n"  
 process\_mailbox(M)  
 M.close()  
 else:  
 print "ERROR: Unable to open mailbox ", rv  
  
 M.logout()

1. **Alert mail sent by the beaglebone black to the authority in case of an intruder to alert the authority.**

hostname = "smtp.gmail.com"  
 password = "ifqdxcimpptxwmul"  
 me = "bbb2daiict@gmail.com"  
 you = "ankitlakra137@gmail.com"  
  
 import smtplib  
 #from email.mime.text import MIMEText  
 from email.mime.image import MIMEImage  
 from email.mime.multipart import MIMEMultipart  
  
 msg = MIMEMultipart()  
  
 msg["Subject"] = "ALERT!!! INTRUDER IN FRONT OF YOUR HOUSE"  
 msg["From"] = me  
 msg["To"] = you  
 #msg.attach(MIMEText(file("gmail1.py").read())  
 fp = open('filename.png', 'rb')  
 img = MIMEImage(fp.read())  
 fp.close()  
 msg.attach(img)  
  
 s = smtplib.SMTP\_SSL(hostname)  
 s.login(me, password)  
 s.sendmail(me, [you], msg.as\_string())  
 s.quit()

1. **Alarm in case of intruder.**

import subprocess  
 import time  
  
 subprocess.call("aplay -D sysdefault:CARD=Device siren.wav", shell=True)  
 time.sleep(7)

1. **To play an already stored audio message.**
2. import subprocess  
    import time  
     
    subprocess.call("aplay -D sysdefault:CARD=Device 1.wav", shell=True)  
    time.sleep(7)

**b.**import subprocess

import time

subprocess.call("aplay -D sysdefault:CARD=Device 2.wav", shell=True)

1. **Keypad.**

import Adafruit\_BBIO.GPIO as GPIO

import time

import sys

key = '134'

count = 3

flag = 0

trial = 0

input = list()

print "ENTER 3 digit pin: "

start = time.time()

while True:

r0=-1

r1=-1

r2=-1

r3=-1

c0=-1

c1=-1

c2=-1

GPIO.setup("P8\_11", GPIO.OUT)

GPIO.output("P8\_11", GPIO.LOW)

GPIO.setup("P8\_12", GPIO.OUT)

GPIO.output("P8\_12", GPIO.LOW)

GPIO.setup("P8\_13", GPIO.OUT)

GPIO.output("P8\_13", GPIO.LOW)

GPIO.setup("P8\_7", GPIO.IN, GPIO.PUD\_DOWN)

GPIO.setup("P8\_8", GPIO.IN, GPIO.PUD\_DOWN)

GPIO.setup("P8\_9", GPIO.IN, GPIO.PUD\_DOWN)

GPIO.setup("P8\_10", GPIO.IN, GPIO.PUD\_DOWN)

r0 = GPIO.input("P8\_7")

r1 = GPIO.input("P8\_8")

r2 = GPIO.input("P8\_9")

r3 = GPIO.input("P8\_10")

# print r0

# time.sleep(1)

GPIO.setup("P8\_11", GPIO.IN, GPIO.PUD\_DOWN)

GPIO.setup("P8\_12", GPIO.IN, GPIO.PUD\_DOWN)

GPIO.setup("P8\_13", GPIO.IN, GPIO.PUD\_DOWN)

if r0 == 0:

GPIO.setup("P8\_7", GPIO.OUT)

GPIO.output("P8\_7", GPIO.HIGH)

c0 = GPIO.input("P8\_11")

if c0 == 1:

print "1"

input.append(1)

flag = flag+1

time.sleep(1)

# break

c1 = GPIO.input("P8\_12")

if c1 == 1:

print "2"

input.append(2)

flag = flag+1

time.sleep(1)

# break

c2 = GPIO.input("P8\_13")

if c2 == 1:

print "3"

input.append(3)

flag = flag+1

time.sleep(1)

# break

if r1 == 0:

GPIO.setup("P8\_8", GPIO.OUT)

GPIO.output("P8\_8", GPIO.HIGH)

c0 = GPIO.input("P8\_11")

if c0 == 1:

print "4"

input.append(4)

flag = flag+1

time.sleep(1)

# break

c1 = GPIO.input("P8\_12")

if c1 == 1:

print "5"

input.append(5)

flag = flag+1

time.sleep(1)

# break

c2 = GPIO.input("P8\_13")

if c2 == 1:

print "6"

input.append(6)

flag = flag+1

time.sleep(1)

# break

if r2 == 0:

GPIO.setup("P8\_9", GPIO.OUT)

GPIO.output("P8\_9", GPIO.HIGH)

c0 = GPIO.input("P8\_11")

if c0 == 1:

print "7"

input.append(7)

flag = flag+1

time.sleep(1)

# break

c1 = GPIO.input("P8\_12")

if c1 == 1:

print "8"

input.append(8)

flag = flag+1

time.sleep(1)

# break

c2 = GPIO.input("P8\_13")

if c2 == 1:

print "9"

input.append(9)

flag = flag+1

time.sleep(1)

# break

if r3 == 0:

GPIO.setup("P8\_10", GPIO.OUT)

GPIO.output("P8\_10", GPIO.HIGH)

c0 = GPIO.input("P8\_11")

if c0 == 1:

print "\*"

input.append('\*')

flag = flag+1

time.sleep(1)

# break

c1 = GPIO.input("P8\_12")

if c1 == 1:

print "0"

input.append(0)

flag = flag+1

time.sleep(1)

# break

c2 = GPIO.input("P8\_13")

if c2 == 1:

print "#"

input.append('#')

flag = flag+1

time.sleep(1)

# break

if flag == count:

if input == map(int,key):

print "DOOR OPEN!"

sys.exit(1)

else:

print "WRONG PIN!"

sys.exit(0)

end = time.time()

elapsed = end - start

if elapsed >= 6:

print "Keypad Time out"

sys.exit(0)

1. **Main code to run the above codes.(integrated code)**

import subprocess

import time

import os

import signal

#import doorbell as doorbell

#import subprocess

import math

door = "1"

while True:

print "System Starting"

key\_ret = -1 # 1 = correct; 3 = wrong after 3 attempts

mail\_ret = 0 # 10 = yes, (0,1,2) = no

doorbell\_ret = 0

alarm\_flag = 0

elapsed = 0

#subprocess.call("python servo2.py", shell=True) #calling servo

#subprocess.call("python New.py", shell=True) #Ultrasonic range finder HC-SR04

#subprocess.call("python servo21.py", shell=True) #calling servo to STOP it with 0 degree input

start = time.time()

subprocess.call("python image.py", shell=True) #Calling webcam to take a pic

subprocess.call("python sendmail.py", shell=True) #sending e-mail to user

while True:

end = time.time()

elapsed = end - start

doorbell\_ret = subprocess.call("sudo python doorbell.py", shell=True) #calling doorbell

if elapsed< 40.0 and doorbell\_ret == 1: #This block represents a person ringing doorbell

mail\_ret = subprocess.call("sudo python readmail.py", shell=True) # 1 = yes, 2 = no

if mail\_ret == 10: #User sent 'Yes', which logically implies to open the door

print "WELCOME" #doorlatch opened

#subprocess.call("python servo.py", shell=True) #using servo to open the door

print "Please close the door in 30 seconds"

time.sleep(30)

#subprocess.call("python servo.py", shell=True) #using servo to close the door

#doorlatch closed

break

elif mail\_ret == 0: ##User sent 'No 0', which logically implies to NOT open the door

print "DOOR NOT OPEN" #and audio

break

elif mail\_ret == 1: ##User sent 'No 1', which logically implies to NOT open the door and play message one

print "DOOR NOT OPEN" #and audio

subprocess.call("python message.py", shell=True)

sleep(10)

break

elif mail\_ret == 2: ##User sent 'No 2', which logically implies to NOT open the door

print "DOOR NOT OPEN" #and audio

subprocess.call("python message2.py", shell=True)

sleep(10)

break

if elapsed >= 40 and doorbell\_ret == 0: #this block represents a theif

print"ALARM" #play alarm

subprocess.call("python image.py", shell=True)

if alarm\_flag != 1:

print "CHANGE THE PLUG!!!!"

time.sleep(20)

subprocess.call("python alarm.py &", shell=True)

alarm\_flag = 1

subprocess.call("python sendalertmail.py", shell=True)

sleep(15)

break

if key\_ret == -1:

key\_ret = subprocess.call("sudo python keypad.py", shell=True)

if elapsed < 40 and key\_ret == 1: #This block represents a person entering pin on keypad

print"WELCOME through Keypad entry"

#subprocess.call("python servo.py", shell=True) #using servo to open the door

print "Please close the door in 30 seconds"

time.sleep(30)

#subprocess.call("python servo.py", shell=True) #using servo to close the door

break

**Block Diagram / Pin Diagram:**