EECE 345 Final Project
Camera Operated Gate Opening IoT System

Alexandro Ayala

12/15/2022

1. Product Definition

1.1. Summary

The goal of the project is to allow a person to cross through a gate without needing to touch anything to pass through. The gate opening and closing will be decided by authorized personnel viewing a camera stream on a site where the personnel will input a lock or unlock command and initiate the actual mechanism. This system creates a safe environment for animals and personnel by decreasing the surface contact of an area with high traffic and high risk of cross contamination.

1.2. Users

The users would be any business or home that requires a high level of security or as stated before a laboratory that needs to keep an area clean to avoid contamination.

1.3. Interfaces

The user who is trying to get through the gate simply needs to pass by the PIR sensor so that it detects movement. From there the person viewing the camera can decide to let the user in by simply typing “lock” or “unlock” into the submission field of the site the streaming is being hosted on.

2. Project Definition

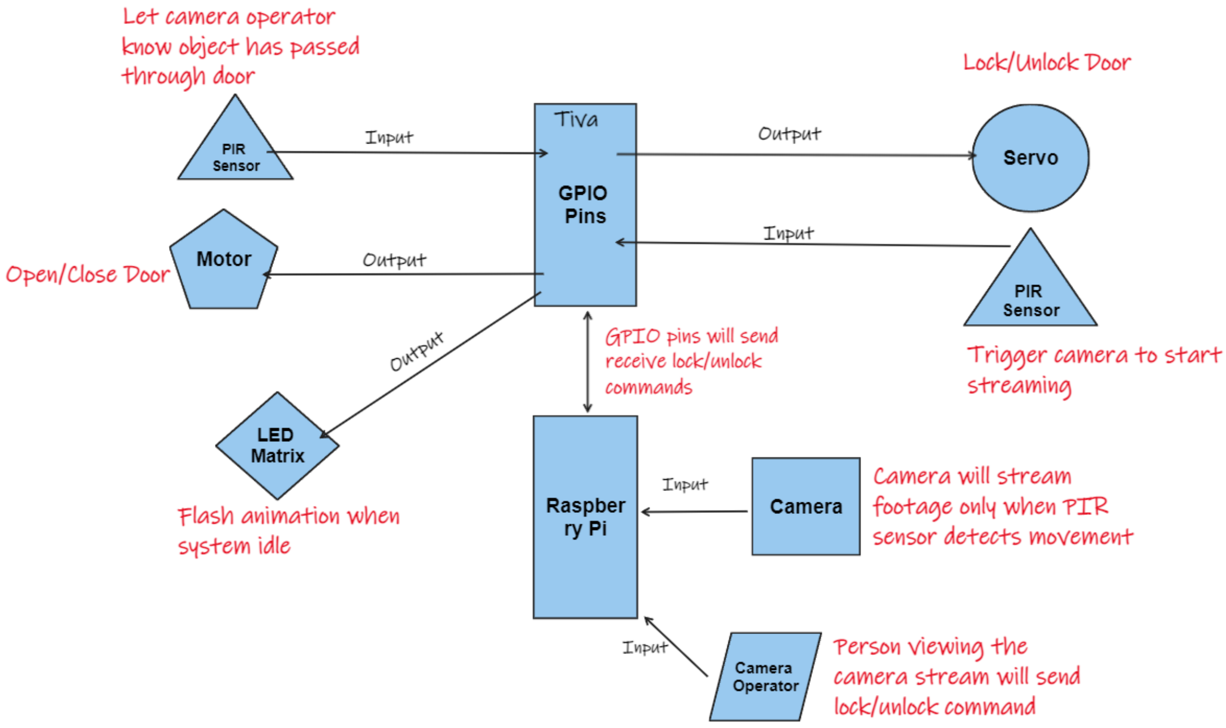
2.1. Summary

The embedded circuit is made up of a dc motor, a servo, two PIR sensors, and an LED matrix which is all connected to a microcontroller. The LED matrix simply plays an animation to let the person know the system is idle and hasn’t detected movement yet. The PIR sensors detect the movement. One of them is to notify the camera operator that a person is waiting for entry and the other is to let the operator know that the person has passed through the gate. The servo serves as a lock and unlock mechanism while the motor opens and closes the gate. The microcontroller is connected to a Raspberry Pi board via a couple GPIO pins. The Raspberry Pi has a camera connected to it which does the actual streaming. The microcontroller sends a signal to the Raspberry Pi to start or stop streaming and the Raspberry Pi sends a signal back to start the lock or unlock sequence.

2.2. Constraints and Limitations

For an actual gate, a large stepper motor would most likely be used compared to a simple DC motor. I don’t have a stepper motor and have never used one, so the DC motor was used instead. I would have like to be able to use two cameras so that the gate could operate from both sides and so that the camera operator can see everything.

2.3. System Architecture



Let camera operator know object has passed through door

Lock/Unlock Door

Open/Close Door

GPIO pins will send receive lock/unlock commands

Trigger camera to start streaming

Flash animation when system idle

Camera will stream footage only when PIR sensor detects movement

Person viewing the camera stream will send lock/unlock command

2.4. Specifications

Metric Number	Need Number	Metric	Units	Marginal Value	Ideal Value
1	5	Low Stream Output Latency	ms	<50ms	<3ms
2	5	Low User Input Latency	ms	<10ms	<5ms
3	4	Compact Design	cm	<30cm	<15cm

2.5. Prototype Costs

Quantity	Description	Unit Cost
1	Tiva™ C Series TM4C123G LaunchPad Evaluation Kit	$22.60
2	PIR Motion Sensor	$.99
1	Raspberry Pi 4 Model B	$55
1	5V Servo Motor	$11.99
1	5V DC Motor	$14.36
1	8x8 LED Matrix	$1.99
		Total Cost $106.93

3. Project Design

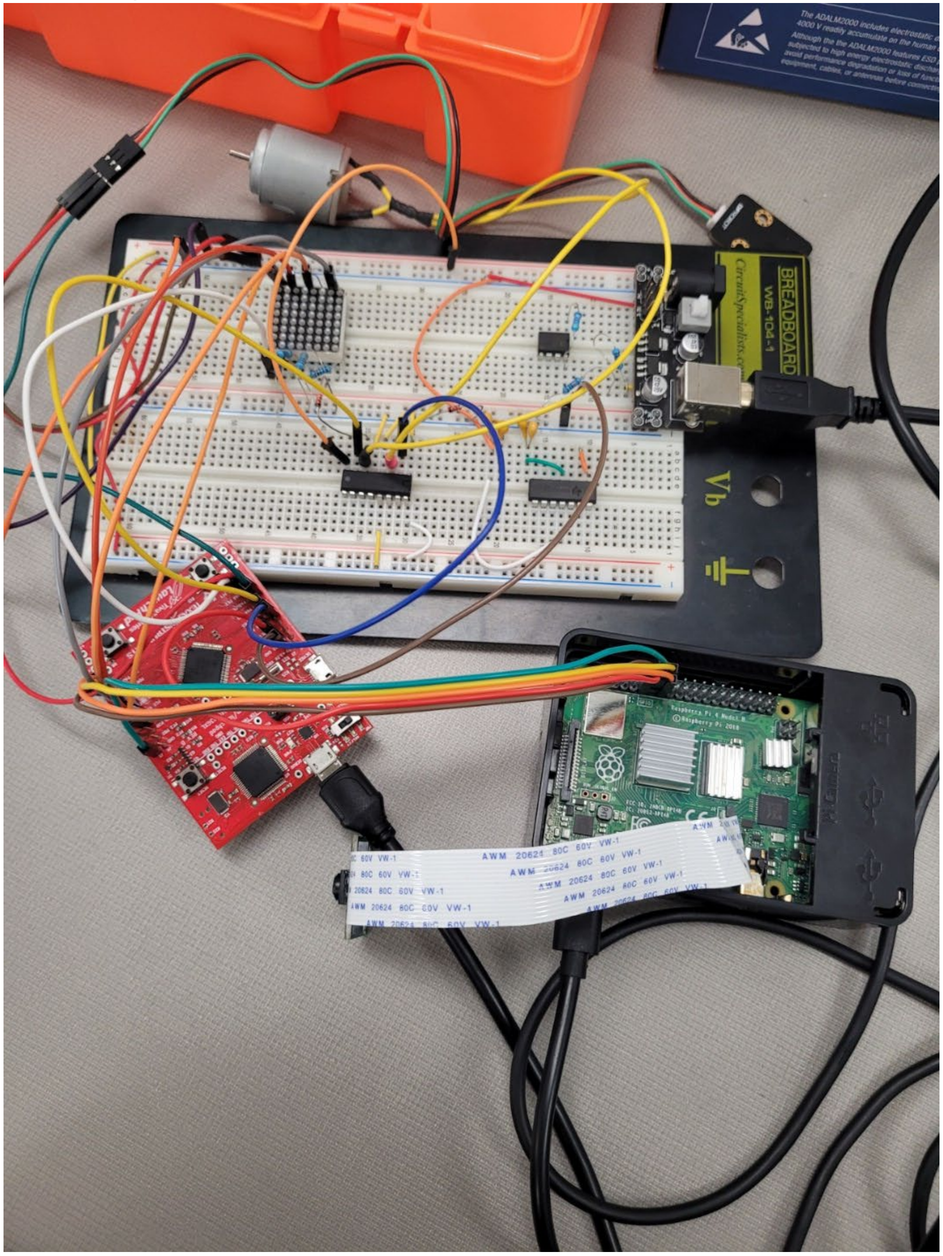
3.1. software components

Embedded C script for controlling the locking and unlocking mechanism and a Python script for creating a server and streaming the footage to a web page.

3.2. hardware components

A servo, dc motor, LED matrix, a camera, and 2 PIR sensors

3.3. final design with photos



4. Conclusions and recommendations.

5. Peer Evaluation, including a list of who did what in the project

Fill out the table to rank each team member in your group, skip if you worked alone.

Write the name of each of your group members in a separate column. For each person, indicate the extent to which you agree with the statement on the left, using a scale of 1-4 (1=strongly disagree; 2=disagree; 3=agree; 4=strongly agree).

Evaluation Criteria	Group member: Alexandro Ayala (only member)	Group member:
Attends group meetings regularly and arrives on time.	4	
Contributes meaningfully to group discussions.	4	
Completes group assignments on time.	4	
Prepares work in a quality manner.	4	
Demonstrates a cooperative and supportive attitude.	4	
Contributes significantly to the success of the project.	4	

6. Appendix

Embedded Code

```
#include <stdint.h>

#include <stdbool.h>

#include "driverlib/gpio.h"

#include "inc/hw_memmap.h"

#include "inc/hw_sysctl.h"

#include "inc/hw_types.h"

#include "driverlib/debug.h"

#include "driverlib/sysctl.h"

#include "SysTick.h"

#include "driverlib/pwm.h"
```

```
//#include "main.h"
#include "driverlib/pin_map.h"
#define GPIO_PORTF_LOCK_R       (*((volatile uint32_t *)0x40025520))
#define GPIO_PORTF_CR_R         (*((volatile uint32_t *)0x40025524))
#define GPIO_PORTF_PUR_R        (*((volatile uint32_t *)0x40025510))
static int32_t value12;
static int32_t value2;
static uint32_t i;
int main(void){
        SysCtlClockSet(SYSCTL_SYSDIV_12_5 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ | SYSCTL_OSC_MAIN);
        SysTick_Init(0xffffff, 0x05);       //enable systick with no interrupts

        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA); //our PIR sensors
        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);   //our LEDs and motor
        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);   //our LEDs
        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);   //our servo
        SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
        value12=0;
        //only reading 2 input
        GPIOPinTypeGPIOInput(GPIO_PORTA_BASE, GPIO_PIN_6 | GPIO_PIN_5);//ir sensor read
        GPIOPinTypeGPIOOutput(GPIO_PORTA_BASE, GPIO_PIN_3); // output to Rasberry PI
        GPIOPinTypeGPIOOutput(GPIO_PORTE_BASE, GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5);
        GPIOPinTypeGPIOOutput(GPIO_PORTC_BASE, GPIO_PIN_4 | GPIO_PIN_5 | GPIO_PIN_6 | GPIO_PIN_7);
        GPIOPinTypeGPIOOutput(GPIO_PORTD_BASE, GPIO_PIN_1 | GPIO_PIN_0);
        GPIOPinTypeGPIOInput(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_7);             //read from Rasberry PI

        SysTick_Wait(16777215); //add delay about 1 second for each delay
        SysTick_Wait(16777215);
        SysTick_Wait(16777215); //add delay
        SysTick_Wait(16777215); //add delay
        while(1){
                value12=GPIOPinRead(GPIO_PORTA_BASE, GPIO_PIN_5 | GPIO_PIN_6);
                i = 0;
                value2 = GPIOPinRead(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_7);
                if (value12 == 0x20){
                        GPIOPinWrite(GPIO_PORTA_BASE, GPIO_PIN_3, 0x08);
```

```
}
if (value2 == 0x40 ){
        while(i < 50){          //servo gate unlock
            GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_1, 0x02);
            SysTick_Wait(32000);
            GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_1, 0x00);
            SysTick_Wait(320000);
            i ++;
    }
        i = 0;
        GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_4, 0x10);//open door
        GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_5, 0x00);

        SysTick_Wait(16777215); //add delay
        SysTick_Wait(16777215); //add delay
        GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_4, 0x00);
        GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_5, 0x00);

        SysTick_Wait(16777215);
        SysTick_Wait(16777215);
        SysTick_Wait(16777215); //add delay to stop reading PIR
        SysTick_Wait(16777215);
}else if(value12 == 0x40){
        GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_0, 0x01);
}
        if(value2 == 0x80){
            GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_4, 0x00);
            GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_5, 0x20);//close door

            SysTick_Wait(16777215); //add delay
            SysTick_Wait(16777215); //add delay

            GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_4, 0x00);
            GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_5, 0x00);

            SysTick_Wait(16777215); //add delay
        i = 0;
        while(i < 50){//servo lock
            GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_1, 0x02);
            SysTick_Wait(16000);
            GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_1, 0x00);
            SysTick_Wait(320000);
            i ++;
        }

        SysTick_Wait(16777215); //add delay to stop reading PIR
        SysTick_Wait(16777215);
        SysTick_Wait(16777215); //add delay to stop reading PIR
        SysTick_Wait(16777215);

}else{//this is my led matrix animation
    //will play when idle
    GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_3, 0x08);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTE_BASE, GPIO_PIN_3, 0x00);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_4, 0x10);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_4, 0x00);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_5, 0x20);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_5, 0x00);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_6, 0x40);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_6, 0x00);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_7, 0x80);
    SysTick_Wait(2000000);
    GPIOPinWrite(GPIO_PORTC_BASE, GPIO_PIN_7, 0x00);
```

```
                        SysTick_Wait(2000000);
                }
        }
}
```

Python Code

```
# Web streaming example
# Source code from the official PiCamera package
# http://picamera.readthedocs.io/en/latest/recipes2.html#web-streaming

import io
import picamera
import logging
import socketserver
from threading import Condition
from http import server
import RPi.GPIO as GPIO
from flask import Flask, request, render_template, url_for
import threading
import cgi

PAGE="""\
<html>
<head>
<title>Raspberry Pi - Surveillance Camera</title>
</head>
<body>
<center><h1>Raspberry Pi Surveillance Camera</h1></center>
<center><img src="stream.mjpg" width="640" height="480"></center>
<center><form method="POST" enctype="multipart/form-data" action="/">
   <input type="text" name="unlock/lock" placeholder="unlock/lock/stop">
   <input type="submit" value="Submit"></form></center>
</body>
</html>
"""
#######
#initialize GPIO
```

```
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(15, GPIO.OUT) #this will connect to the Tiva and tell it to unlock
GPIO.setup(16, GPIO.OUT) # this will tell tiva to lock
GPIO.setup(13, GPIO.IN)  #this will be PIR sensor 1 for turning on system
GPIO.setup(11, GPIO.IN) #this will be PIR 2 for turning off system
#######

class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()
    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            # New frame, copy the existing buffer's content and notify all
            # clients it's available
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                self.condition.notify_all()
            self.buffer.seek(0)
        return self.buffer.write(buf)

class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type', 'multipart/x-mixed-replace; boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type', 'image/jpeg')
                    self.send_header('Content-Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
            except Exception as e:
                logging.warning(
                    'Removed streaming client %s: %s',
                    self.client_address, str(e))
        else:
            self.send_error(404)
            self.end_headers()
    def do_POST(self):
        if self.path.endswith('/'):
            ctype, pdict = cgi.parse_header(self.headers.get('Content-Type'))
            pdict['boundary'] = bytes(pdict['boundary'], "utf-8")
            content_len = int(self.headers.get('Content-length'))
            pdict['CONTENT-LENGTH'] = content_len

            if ctype == 'multipart/form-data':
                fields = cgi.parse_multipart(self.rfile, pdict)
                new_val = fields.get('unlock/lock')
            self.send_response(301)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Location', '/index.html')
            self.end_headers()
            if(new_val[0] == "unlock"): #this section is reading the user inputs
                print(new_val)
                GPIO.output(15, True)
            elif(new_val[0] == "lock"):
                print(new_val)
                GPIO.output(16, True)
                camera.stop_recording()
            elif(new_val[0] == "stop"):
                camera.stop_recording()
            else:
                print("Error: Invalid Answer")
            #self.wfile.write(encode(new_val))

class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True

####might not actually need this section
app = Flask(__name__)

@app.route('/', methods = ['POST', 'GET'])
def index():
    if request.method == 'POST':
        if request.form['unlock/lock'] == 'unlock':
            print("Gate Unlocked")
        elif request.form['unlock/lock'] == 'lock':
            print("Gate Locked")
        else:
            pass
    return render_template('index.html');
###
```

```
#if (GPIO.input(13)): #if PIR sensor 1 is triggered
with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    if(GPIO.input(13)):
        print("Object Detected")
        output = StreamingOutput()
    #Uncomment the next line to change your Pi's Camera rotation (in degrees)
        camera.rotation = 180

        camera.start_recording(output, format='mjpeg')

        address = ('', 8000)
        server = StreamingServer(address, StreamingHandler)
        server.serve_forever()
        '''threading.Thread(None, server.run).start()
    #Unlock, Lock = buttonpress()
        #my_form_post()
        '''
    elif (GPIO.input(11)):
        print("Object has passed")
```