Salah Alrefai

ECE 4220

05/15/2015

Final Project Report

**Abstract**

My project is an auto attendance program for instructors of a class or a meeting organizer. The main goal of this project is to put together what we learn from class in concepts and techniques to use in useful program. The idea of the project came from me and didn’t take it from somewhere else. The idea as I think to put a helpful solution in hand of instructors and meeting organizers. The project contain three programs, one is the main program, second is the python program to read from sensor and third is the user program. I used a raspberry pi as my embedded system and PIR motion sensor as my hardware component. I focus on in this lab to do fifo’s (named pipes) and threads. I learned a lot of concepts and techniques that can be used in purposing solutions and apply them to real life.

**Introduction**

My project is an auto attendance program for instructors of a class or a meeting organizer. The project goal is to solve the problem of attendance either in schools, universities or any one love to take attendance for different reasons and make taking attendance more easy, simple and, robust. Today’s methods of collection attendance may waste useful time of class as example and may make hard on students who came in late into the class. Those problems led me to the solution of my project. My solution is that the instructor will give time in seconds to the program that I implemented then the program have a clock to take attendance through the motion sensor, then the program will request information about the user, which should he have an app that will send his info automatically.

My objectives/goals:

Expand my horizon in embedded systems and work with different ones than we used for this class.

Learn new techniques in programing and components or concepts that associate with computing.

Try to include what we learned in class to useful usage of it and companied those programing techniques for better understanding the material.

**Background**

The idea of the project came up to me during classes that took my attendance as a student in the university. Since I know there are motion sensors that used in different applications in different appliances from smart houses to security for companies. I know the functionality that needed to automate the attendance through it. In my implementation, I kind have a client-server relationship between my programs. This relation comes from fifo (named pipe) and the way they are used in the program. Named pipe is a common technique used by computer engineers to overcome problems of communication between processes in computer. The main goal is making a program for the instructor to control and request information from student application and if info exist that will be sent back to the instructor page. This process is used in web pages in the browser and replicates the client-server relationship. Also, the way of implementing the project I looked at life examples to come up with this design. The idea behind the requesting information from the user program, I got it from classes that take attendance and when some instructors make student put their student ID in a paper to count for attendance. My project is an auto attendance for classes, meetings or anyone want to take attendance of the crowd coming to an event or meeting. My project will be uses in schools, universities and companies.

**Proposed method / System description / Implementation**

Figure flow chart

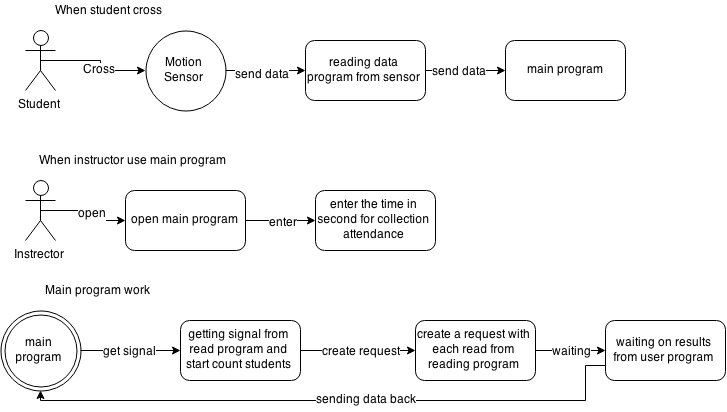
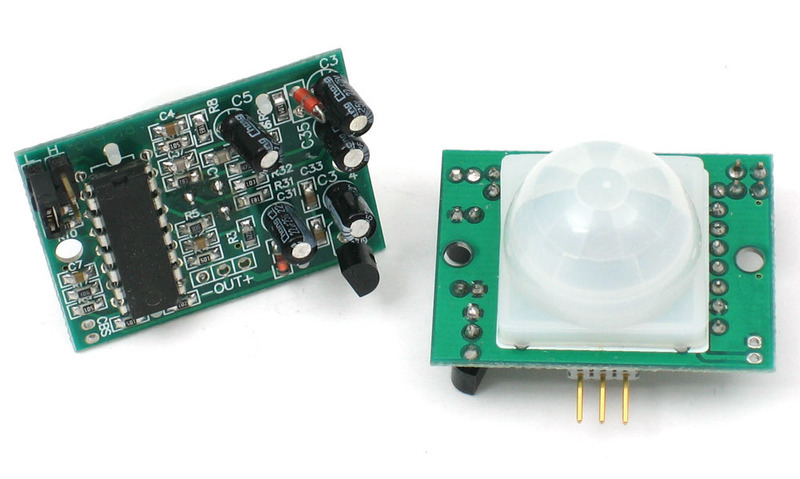
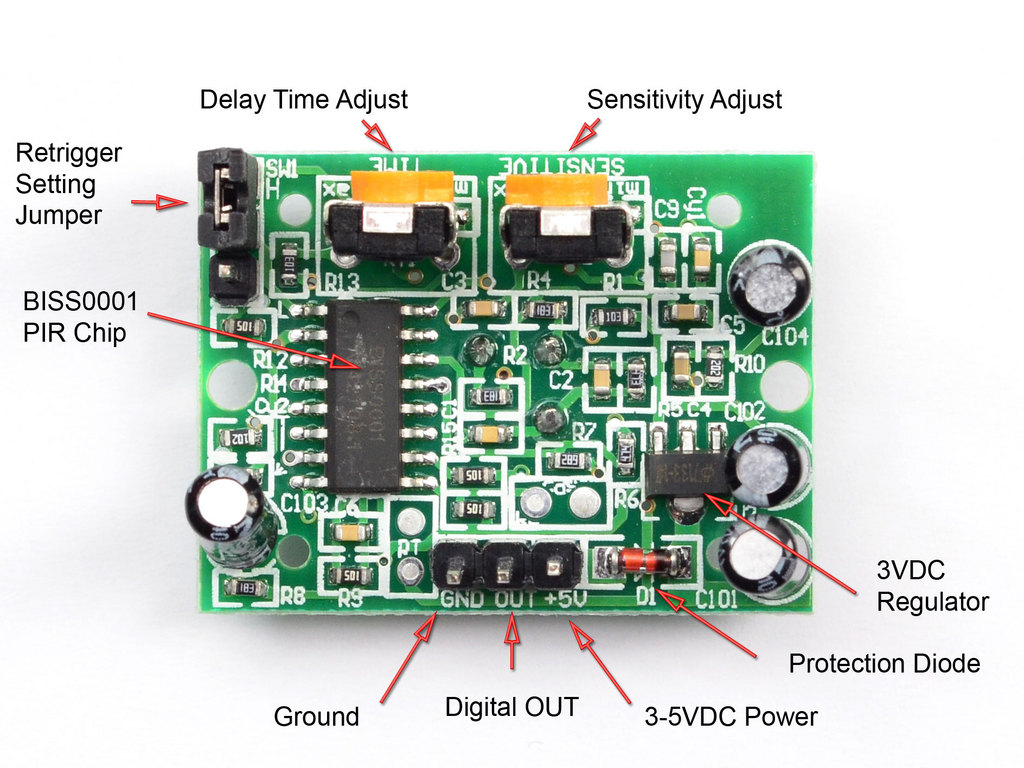
****

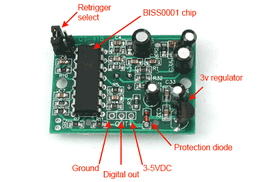
Figure visual workflow

The project contain three programs inside it, each program have a specific task to achieve. The first program is a program writing in python that open up a fifo (named pipe) and write to it a character, 1 that represent high signal that if the program detect a motion (reading from motion sensor) from motion sensor that comes from the pin, The program setup the pin 18 to read data from it. This high signal indicates that a student has pass (cross) motion sensor, this high signal sent to the main program. The second program is the main program that interacts with the instructor by asking him first to enter the number of seconds to allow the collection of attendance for the class. Also, the main program at first creates two fifo’s and open them for later usage. After the instructor input the main program create a thread to get data from the motion sensor from the fifo that is used for that purpose. The fifo in this thread is reading from python program that uses that thread and I did that because I was interesting of sending data from one programing language to other one and do operation on the input. The thread makes reading data simple and it running forever in the background waiting on an input from the sensor program so every person cross through that sensor within the time counter from the professor won’t interfere with each other, also we want the two tasks running in the same time. Inside the reading thread from fifo, if there is a read from fifo the program will count the students cross and update a variable for that purpose. In addition, within that specific thread and when the read happing, that thread will create a new request to wait on student information that be inputted from the user program. The new created thread is for requesting data from the second create fifo that retrieve data from user program. I didn’t this way so the three tasks running together within the time that been specified by the instructor which is the window for taking attendance. The last program is the user program that opens a fifo for writing info to it upon request in the created thread in main program. Also, it running forever in the background that takes input from the console, which is student ID and send it through fifo to the request thread in the main program. I did make it running forever because I want to show that only thread request will receive data and if there is none, there is no data going to thread.

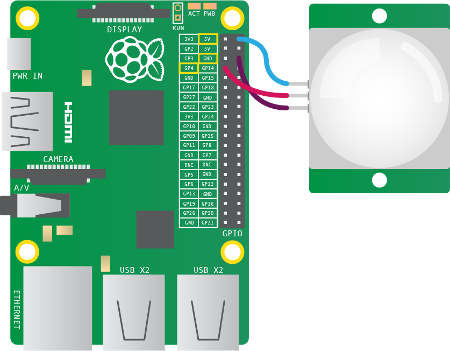
**Pyroelectric ("Passive") InfraRed Sensors**

****

****

****

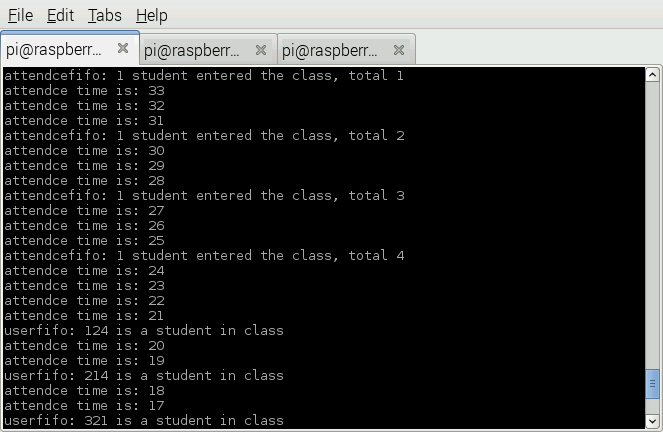
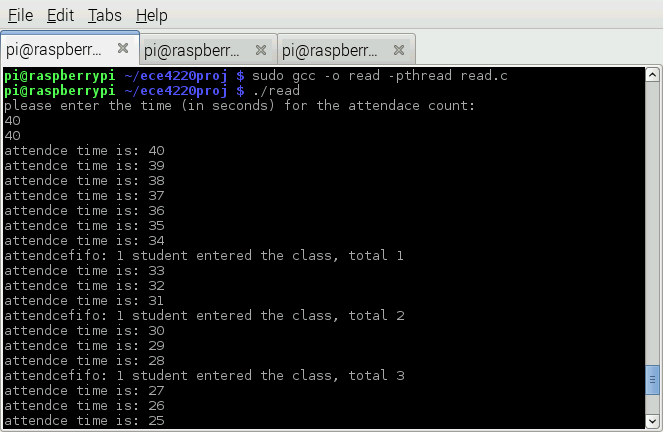
Reading PIR Sensors

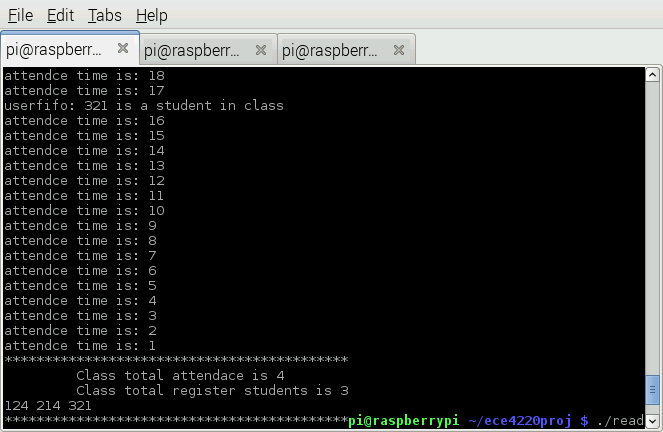
Connecting PIR sensors to a microcontroller is really simple. The PIR acts as a digital output so all you need to do is listen for the pin to flip high (detected) or low (not detected). Its likely that you'll want retriggering, so be sure to put the jumper in the H position. Power the PIR with 5V and connect ground to ground. Then connect the output to a digital pin. There is an example below. 

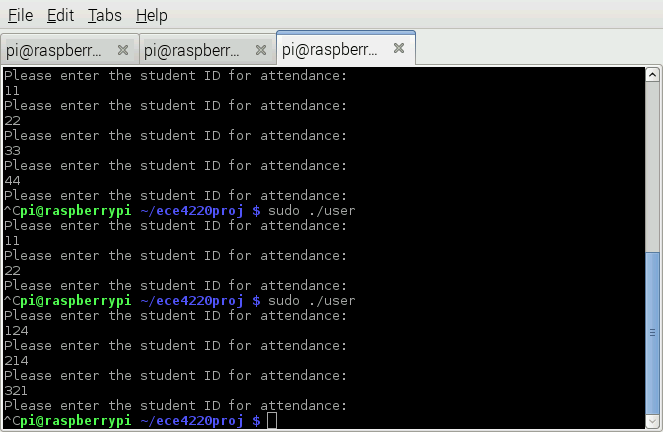
**Experiments and Results**

In this lab, there are three programs and two of them relay on created fifo by the main program. Testing for two programs that relay on main program was done by doing the main functionality of each program and test it, then tested with fifo’s by either read or write to fifo’s. The python program that responsible upon reading data from motion sensor by the pin (GPIO) interface was tested by real life test like put my hand next to the motion sensor and remove it quickly. The reading was to quick in this case due to the program do nothing besides reading the pin. Also, I tested the program with fifo opening and write to them and retrieve data from the main program and print it to console to check wither is correct or not. Checking the second program that relay on main program due to fifo was easy. Because the only thing the program do is get numbers from the user for the student ID, then writing it to fifo, I tested the writing an input to console and print it to the console from main program for checking that the writing operation is successful. The main program tests was little different due that have two creating fifo’s and two type of threads is been created. Testing the creating the fifo’s is easy just check them if they exist either by ls command console or see the file that they be created in and if they exist inside it. Also, checking the number that the instructor put it for the timer was easy through console and printing them. The first thread checking was by printing to the console due it runs and read high signals from the motion sensor if some one cross next the motion sensor. Lastly, checking the user info inputted by the student in the user program checked in printing also. I ran the programs a lot of time to check on the result. The division of the worked happened as working on getting data from the sensor, then main program: the timer for instructor, then first thread to get info through fifo if some one crossed next to the motion sensor and counting students, lastly, the thread that get data from the user program. Lastly, implementing the user program and test it.

Main program





User program 

**Discussion and Conclusions**

The project wasn’t hard to do but using the raspberry pi and interface hardware component was kind of new to me. Also, I never program on raspberry pi any techniques or concepts learned from the class before so the results were amazing. I notice in the raspberry pi the level of compiler optimization was different than the TS2750, because I ran to problems with threads and I had to change a level of optimization there. Also, I realized that fifo’s are really important when interfacing with different programing languages or processes to establishes communication between them. I was hoping of using TCP/IP or UDP in my program as was a main portion of it, but since the problem with static IP in the lab and raspberry pi plus the time constrains I couldn’t do them. The interesting part of implementation that threads run very fast sometime and slow sometime, also forever loops are not a good solution in implementing, I notice that the thread that responsible upon reading from fifo that get reading from motion sensor are not accurate. I think that interrupts would be better solution than a forever loop but due to time constrains I couldn’t explore on that matter. The results that I obtain made sense of course due to cross path of what we learned in the lectures and labs. I were expecting when my hand goes near the motion sensor the thread will read from fifo that contain the reading from python program of data from the motion sensor and the thread count me as student entered the room. Upon that the thread request is been created and waiting on reading from fifo, that comes from user program that write student ID to fifo that inputted from the user. Lastly, the main program should print to console the total number of attendances of class, total number of students that gave their student IP and what’s their students ID’s, The result came like I expected. They were as expected, because of the way I structured the program to behaved. There wasn’t a lot of problems I encounter doing the project of these structure. Only problem was figuring out, why the threads behaved little crazy due to compiler optimization. I fix the problem by changing compiler optimization level. I did learn a good amount about using fifo’s between two different programing languages and they are helpful in many situations. The problem with my approach is the project is a simulation of what I entrained it to do due to TCP/IP, UDP static IP problem in the Internet of the university and raspberry pi. I would suggest that changing the structure little bit and depends on Internet service to do it instead of doing it the way I did it. May be using Bluetooth instead of Internet would be better. The problem with my approach, the motion sensor won’t handle a large amount of people getting into the room at ones and need array of motion sensors or another approach to solve the problem. Overall, it is useful project and the baby steps to solve the problem with different approach.

**Appendices**

**Code**

Main program

#include <sys/stat.h>

#include <unistd.h>

#include <fcntl.h>

#include <stdio.h>

#include <pthread.h>

void\* thread0();

void\* thread1();

typedef struct {

int studentID;

} students;

typedef struct {

int attendaced;

int total;

} classes;

//Global variables

int fifo;

char filename[] = "fifo.tmp";

char filename2[] = "fifo2.tmp";

classes class;

int atten = 0;

students student[50];

int main(int argc, char \*argv[]){

int timer\_atend = 0;

// Create FIFO

int s\_fifo = mkfifo(filename, S\_IRWXU);

if (s\_fifo != 0)

{

printf("mkfifo() error: %d\n", s\_fifo);

return -1;

}

// Create FIFO

int s\_fifo2 = mkfifo(filename2, S\_IRWXU);

if (s\_fifo2 != 0)

{

printf("mkfifo2() error: %d\n", s\_fifo2);

return -1;

}

//opening the fifo

fifo = open(filename, O\_RDONLY);

if (fifo < 0)

{

printf("open() error: %d\n", fifo);

return -1;

}

printf("please enter the time (in seconds) for the attendace count: \n");

scanf("%d", &timer\_atend);

printf("%d \n", timer\_atend);

pthread\_t tid;

pthread\_create(&tid, NULL, thread0, NULL);

int counter=0,i;

//count down the time for enabling attendce

for (i=timer\_atend; i>0; i--) {

printf("attendce time is: %d \n", i);

sleep(1);

}

// Close and delete FIFO

close(fifo);

unlink(filename);

unlink(filename2);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t Class total attendace is %d\n", class.total);

printf("\t Class total register students is %d\n", class.attendaced);

printf("\t");

for(i=0;i<atten;i++){

printf("%d ",student[i].studentID);

}

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

return 0;

}

//thread for reading from fifo, the students that passes motion sensor

void\* thread0(){

char arr2[3];

int counter =0;

pthread\_t tid;

while (1) {

//Read from FIFO

int s\_read = read(fifo, arr2, sizeof(arr2));

if (s\_read < 0)

{

printf("fprintf() error: %d\n", s\_read);

}

counter++;

printf("attendcefifo: %s student entered the class, total %d \n", arr2, counter);

class.total = counter;

pthread\_create(&tid, NULL, thread1, NULL);

sleep(1);

}

pthread\_exit(0);

}

//thread for reading the user data from user program

void\* thread1(){

int value;

//opening fifo

int fifo = open(filename2 , O\_RDWR);

//reading data from user program

read(fifo, &value, sizeof(value));

//print data

printf("userfifo: %d is a student in class\n", value);

student[atten].studentID = value;

class.attendaced = ++atten;

pthread\_exit(0);

}

User program

#include <sys/stat.h>

#include <unistd.h>

#include <fcntl.h>

#include <stdio.h>

int main(int argc, char \*argv[]){

int studentID;

//opening the fifo

int fifo = open("fifo2.tmp", O\_RDWR);

while(1){

printf("Please enter the student ID for attendance: \n");

//get student number

scanf("%d", &studentID);

//write it to the fifo

if(write(fifo, &studentID, sizeof(studentID)) != sizeof(studentID)) {

printf("write error");

}

}

return 0;

}

Sensor program

import time

import RPIO

# setup the pin to use

pir = 18

RPIO.setup(pir, RPIO.IN)

# open the fifo to wite to

filename = "fifo.tmp"

counter = 0

while True:

# Block until writting finishes...

with open(filename, 'w') as f:

#sensor detect motion

if RPIO.input(pir):

#writting data to fifo

data = f.write('1')

time.sleep(2)

time.sleep(1)

**Project Proposal**

**Attendance Tracker**

The project that I am proposing is an attendance tracker. In large classes or meetings people need to collect attendance of class or meeting for many different reasons but calling each person in name in large meeting or class will take a lot of time and effort or pass on a paper that people write their names will take time and the papers might be lost. I am proposing a automated way that give the person that take the attendance a free hand experience and the attenders to have a less effort and problems to contribute their attendance. I choose this project because class or meetings systems are necessary in this modern age and it solves small problems but effective ones. Also, it makes meetings or classes easier to attend and contribute to it. My project is using a motion sensor in the door that will count each person that go into the class or meeting then it will send over a network or Bluetooth to microcontroller that sends requests of confirmation to the same number of people that get into class or meeting to their mobile devices, those people will get notify by program that there is a attendance and open up the app so the person can save his info and send it back to the microcontroller that will save it and upload it to a web service or program for the guy that is taking attendance. There are different approaches for this problem using a full mobile app or a web service that will do the same thing the problem with them is not full automated process and every time the user should visit this app or website to confirm attending and the guy to take attendance has to setup the attendance every time. What I am proposing is a system fully exists in class or meeting room and will serve the same purpose with nearly full-automated process.

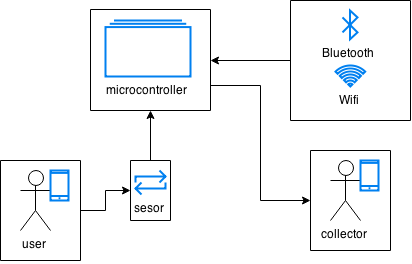


Figure 3 - general structure

Figure 4 - Flow Chart

**Goals:**

* Short goals
  + The motion sensor detect and microcontroller count people
  + The microcontroller send the confirmation request to devices
  + Make a new way and automated to attendance in classes/meetings
* Long goals
  + The project to serve a new way of interacting with meetings and classes
* Final
  + Serve a project can provide the freedom in attendance and interaction in classes/meetings

**Benefits and Outcomes:**

The project aiming to solve the problem of taking attendance in large meetings and classes, I am trying to automate the process and make it more efficient. Solving this problem makes class’s teachers focus more time on class and meetings organizers focus more time on the meeting.

I am hoping as an outcome of this project learns how to use network and Bluetooth requests from different devices. Also, learn how to build rapid prototype project for demonstrating for funds rising or company proposal idea. Lastly, I hope to place my knowledge into a useful project.

**Constrains and specifications:**

The project constrains are the sensor can count one person at the time, if two person go through the door sensor won’t detect that second person and consider them one. If the room is really large the Bluetooth chip won’t apple to find the devices near by due to distance limitation.

The project specifications are:

* Embedded Device is TS-7250 (maybe other)
* Using motion sensor as hardware component
* Bluetooth chip or Ethernet or Wi-Fi chip
* Multi tasking/threading
* Network communication
* Client-Server Model
* Interfacing to peripheral device (maybe)
* I might use the FairCom database

**Milestone:**

Table 1 - timeline

|  |  |
| --- | --- |
| Task | Date |
| Sensor interface / sensor app | **04/15/2015** |
| Microcontroller app | **04/22/2015** |
| Mobile app | **04/29/2015** |
| Collector app (mobile or web) | **05/04/2015** |