
ECE - 511-001

Fall 2017

People Tracker

29th November 2017

Cody Massey, Logan Robson, Ayo Raymond, Aaron Warrick

TABLE OF CONTENTS

OVERVIEW	3
MOTIVATION	3
SOLUTION	3
PARTS	4
PVS ffi	4°
CEF FMSCA	4°
Z HL	4°
IUVGMR	6
RESULTS	6
CONCLUSION	6
BIBLIOGRAPHY	7
APPENDIX	8
WANFLYMRQ	8°
SCUNQW	8°
VEKPCMEV	!
SEDCARXW	11

OVERVIEW

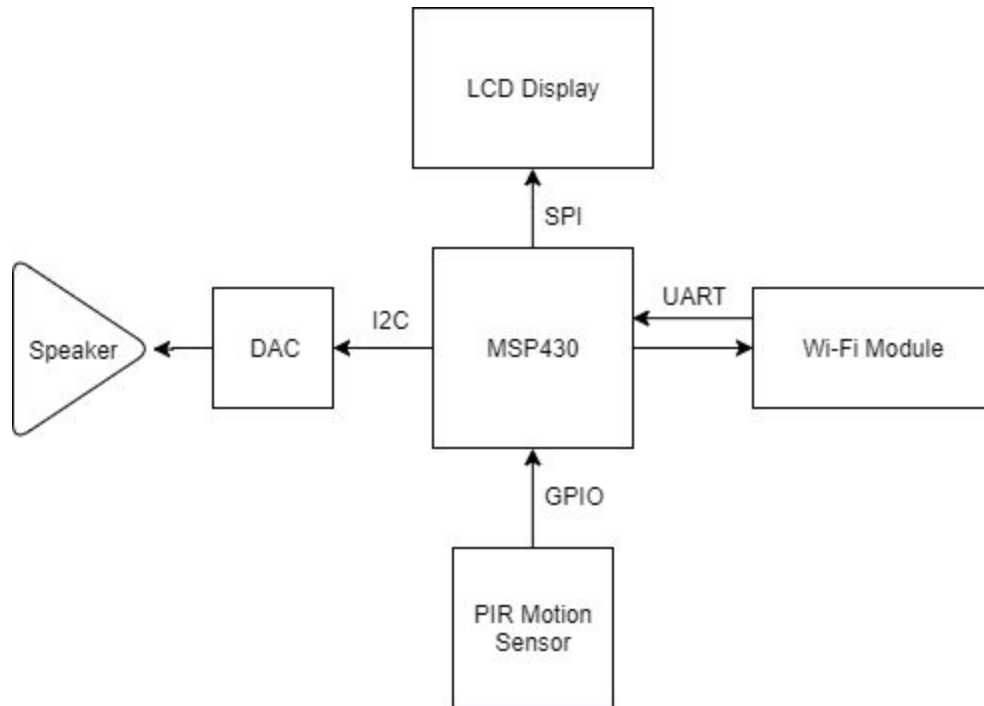
As students working on a group project, an issue that often occurs is finding an empty and quiet place to meet and discuss the project. There are many buildings on campus, which offer study rooms on a first come, first serve basis, and many group meetings are spent finding a place to meet, rather than discussing the project. Our project aims to provide a solution to this issue, by tracking the number of people in a particular room. To accomplish this, we designed a device which mounts to a doorway, and tracks the number of people in a room. The people count for a particular room can be accessed via an internet based interface, or via a printout on the device itself.

MOTIVATION

During the time of midterms and finals it is common to find rooms such as; the computer lab, or study hall reached to its capacity. Finding another room with space available can be a challenging task. The purpose of the People Counter device is to provide both local and remote monitoring of room capacity. This provides a convenient and reliable method to determine if a given space is full or not without actually entering the room.

SOLUTION

Our solution involves using many devices, including a microcontroller, which acts as the central component in our device, responsible for communicating with all other components in our device. Along with the microcontroller, we will also be using an LCD display, Wi-Fi module, digital-to-analog converter (DAC) and two passive infrared motion sensors (PIR). The LCD display will convey information about the room, including current room count and room capacity. The Wi-Fi module is responsible for providing remote access to the room information, and will allow users to view room count, and edit room capacity via an online interface. The PIR motion sensors are responsible for detecting people entering and exiting a room. Finally, the DAC is responsible for playing sounds, when a person is detected to have entered or exited a room, as well as when a room reaches capacity.



Hi-Fi System Architecture Diagram

PARTS

MSP430

The main component in our design is the microcontroller. The microcontroller is responsible for communicating with the other components in the devices. The microcontroller we used is the MSP430FR6989, which we chose for a number of reasons with the foremost being its availability. The table below lists some of the other MSP430s we considered, as well as their memory capacities, clock speed, and price.

Model	RAM	Nonvolatile Mem.	Clock Speed	Price
FR6989	2KB	128KB	16MHz	\$17.99
FR5994	8KB	256KB	16MHz	\$17.64
F5529	8KB	128KB	25MHz	\$13.49

LCD DISPLAY

The purpose of the LCD is to display information of the capacity of the room onto a screen. We use SPI for communication; more specifically, the MOSI pin and clock pin. We chose the TFT Display ST7775 out of convenience because it was already on hand.

Model	Size	Resolution	Interface	Price
TFT Display ST7735	1.35" x 2.2" x 0.25"	160x128	SPI	\$9.95

DAC

The digital-to-analog converter (DAC) is responsible for playing sounds on events (person entering, exiting, capacity reached). We compared a number of DACs and ultimately chose the MCP4725 based on its resolution/price, and availability. The MCP4725 communicates with the MSP430 via I2C, and utilizes the MSP430 eUSCI module. The software for the MCP4725 uses the MSP430 DriverLibrary as a base, with functions to perform specific operations of the DAC built on top.

Model	Resolution	Interface	Price
MCP4901	8-bit	SPI	\$1.01
MCP4725	12-bit	I2C	\$1.01
MCP47A1	6-bit	I2C	\$0.60

WI-FI

The ESP-8266 essentially acts as a web server by providing data transmission from the MSP-430 FR6989 via Uart to a client or user on the network by utilizing TCP, HTTP, and FTP protocols. The information will be accessed through a user interface. The data displayed will include the quantity of people in a given space and state of the room i.e "space available" or "capacity reached".

Model	RAM	Size	Protocols	Voltage	Trans Time	Price
ESP8266	64KB	5x5mm	IPV4, TCP, HTTP, FTP	3.0V-3.6V	< 2ms	\$6.99

IR SENSOR

The IR sensor is responsible for tracking the number of people that walk into the room and out of the room. We use two sensors so that we can determine which direction a person is coming from to easily identify if they are entering or leaving. The sensor goes high when it detects movement and depending on which sensor triggered first, the internal counter is incremented or decremented. The sensor communicates over a GPIO pin. We chose this sensor out of convenience because we already had it.

Model	Range	Delay Time	Size	Interface	Price
HC-SR501	$\leq 100^\circ$, 7 m	5-200s (Adjustable)	32*24 mm	GPIO	\$4.99
HC-SR04	$\leq 15^\circ$, 2cm-400 cm	Negligible	45*20 mm	GPIO	\$12.00

RESULTS

We were able to complete the People Counter, but there existed a number of hiccups. Our device is successfully able to count the number of people that enter and exit a room, using the IR sensors, push the data onto a wifi server, and display the results onto the LCD. On top of that, when the limit of people is reached an alarm is sounded. The issues we faced were related to the IR sensor and the micro sd reader on the LCD. The IR sensor we used has a delay of roughly 3-5 seconds after it detects a person. This means that people counter will miss individuals that come in immediately after one another. For the sensor to work properly, individuals will need to come in over a spread of time, at 3-5 seconds apart at minimum. Another issue we faced, was the speed limitations of I2C and the MSP430, and the resulting restrictions of sounds we could therefore play via the DAC. These restrictions meant that any audio files we wanted to play on the device needed to be resampled at a low bitrate, to be played without distortion. Additionally, the DAC on-board the PCB proved difficult to debug, and we instead used the MCP4725 breakout board. Aside from these two issues, our system works how we expect it to.

CONCLUSION

From completing this project, we learned the necessity to complete and test individual components early on so that proper alternatives can be arranged. The issues we faced, especially with the IR sensor, were discovered very late in the semester because we waited to test the components. They were not project ending but they certainly could have been avoided. Regardless, our project as a whole was success.

BIBLIOGRAPHY

<http://www.ti.com/tool/msp430usbdevpack>

<http://www.ti.com/tool/MSPDRIVERLIB>

https://github.com/Esplt/Hardware/blob/master/mod.pretty/SJ-3523-SMT-TR_N.kicad_mod

https://github.com/Tinkerforge/kicad-libraries/blob/master/AB26TRQ.kicad_mod

<https://github.com/esp8266/Arduino>

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

<http://www.geeksforgeeks.org/implement-itoa/>

GMU ECE 447 Code provided code for LCD

Monk, Simon. **Programming Arduino: Getting Started with Sketches**. New York: McGraw-Hill, 2014. Print. °

°

°

°

°

°

°

°

°

°



TASK DIVISION

LCD - Ayo

WIFI - Aaron

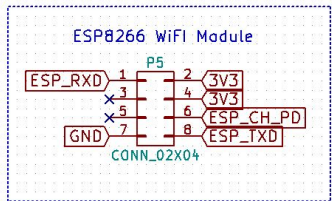
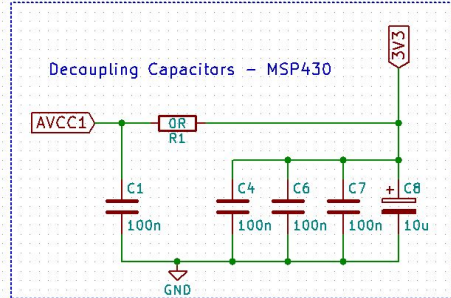
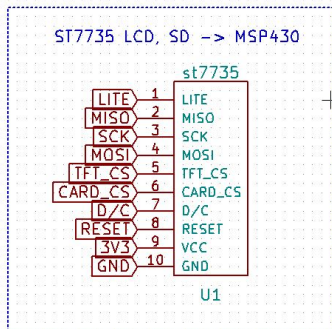
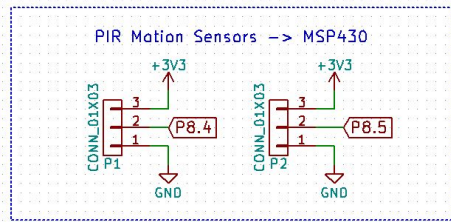
DAC - Logan

IR Sensor - Ayo/Cody

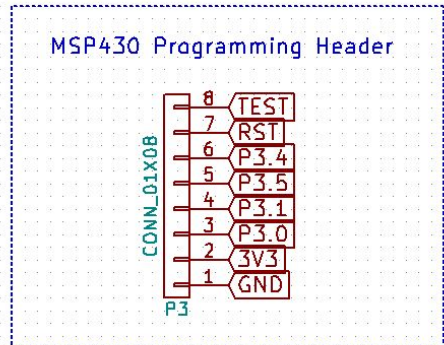
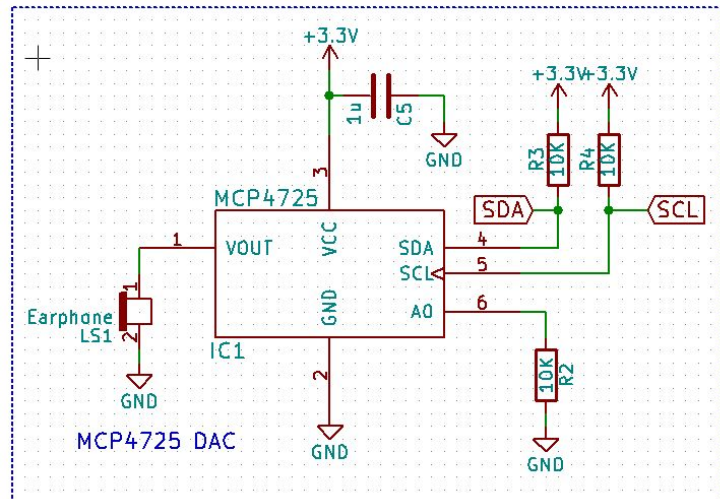
PART LIST

1	C1 -	100n : Capacitors_SMD:C_0603_HandSoldering
2	C2 -	10p : Capacitors_SMD:C_0603_HandSoldering
3	C3 -	10p : Capacitors_SMD:C_0603_HandSoldering
4	C4 -	100n : Capacitors_SMD:C_0603_HandSoldering
5	C5 -	1u : Capacitors_SMD:C_0603_HandSoldering
6	C6 -	100n : Capacitors_SMD:C_0603_HandSoldering
7	C7 -	100n : Capacitors_SMD:C_0603_HandSoldering
8	C8 -	10u : Capacitors_SMD:C_0805_HandSoldering
9	C9 -	10u : Capacitors_SMD:C_0805_HandSoldering
10	C10 -	10u : Capacitors_SMD:C_0805_HandSoldering
11	IC1 -	MCP4725 : TO_SOT_Packages_SMD:SOT-23-6
12	LS1 -	Earphone : headphone:SJ-3523-SMT-TR_N
13	P1 -	CONN_01X03 : Pin_Headers:Pin_Header_Straight_1x03_Pitch2.54mm
14	P2 -	CONN_01X03 : Pin_Headers:Pin_Header_Straight_1x03_Pitch2.54mm
15	P3 -	CONN_01X08 : Pin_Headers:Pin_Header_Straight_1x08_Pitch2.54mm
16	P4 -	CONN_01X02 : Pin_Headers:Pin_Header_Straight_1x02_Pitch2.54mm
17	P5 -	CONN_02X04 : Pin_Headers:Pin_Header_Straight_2x04_Pitch2.54mm
18	P7 -	CONN_01X02 : Pin_Headers:Pin_Header_Straight_1x02_Pitch2.54mm
19	P8 -	CONN_02X06 : Pin_Headers:Pin_Header_Straight_2x06_Pitch2.54mm
20	R1 -	0R : Resistors_SMD:R_0805_HandSoldering
21	R2 -	10K : Resistors_SMD:R_0603_HandSoldering
22	R3 -	10K : Resistors_SMD:R_0603_HandSoldering
23	R4 -	10K : Resistors_SMD:R_0603_HandSoldering
24	U1 -	st7735 : Pin_Headers:Pin_Header_Straight_1x10_Pitch2.54mm
25	U2 -	LM1117-3.3 : TO_SOT_Packages_SMD:SOT-223
26	U4 -	msp430fr6989 : Housings_QFP:LQFP-100_14x14mm_Pitch0.5mm
27	Y1 -	32.768kHz : footprints:AB26TRQ

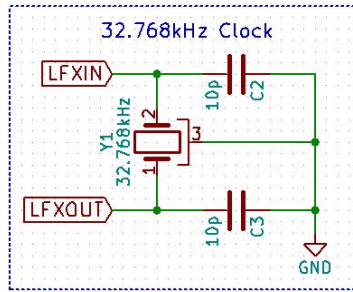
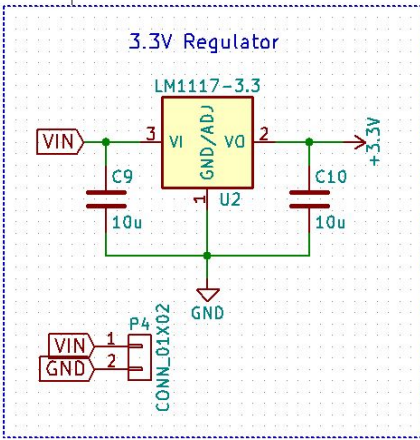
SCHEMATICS



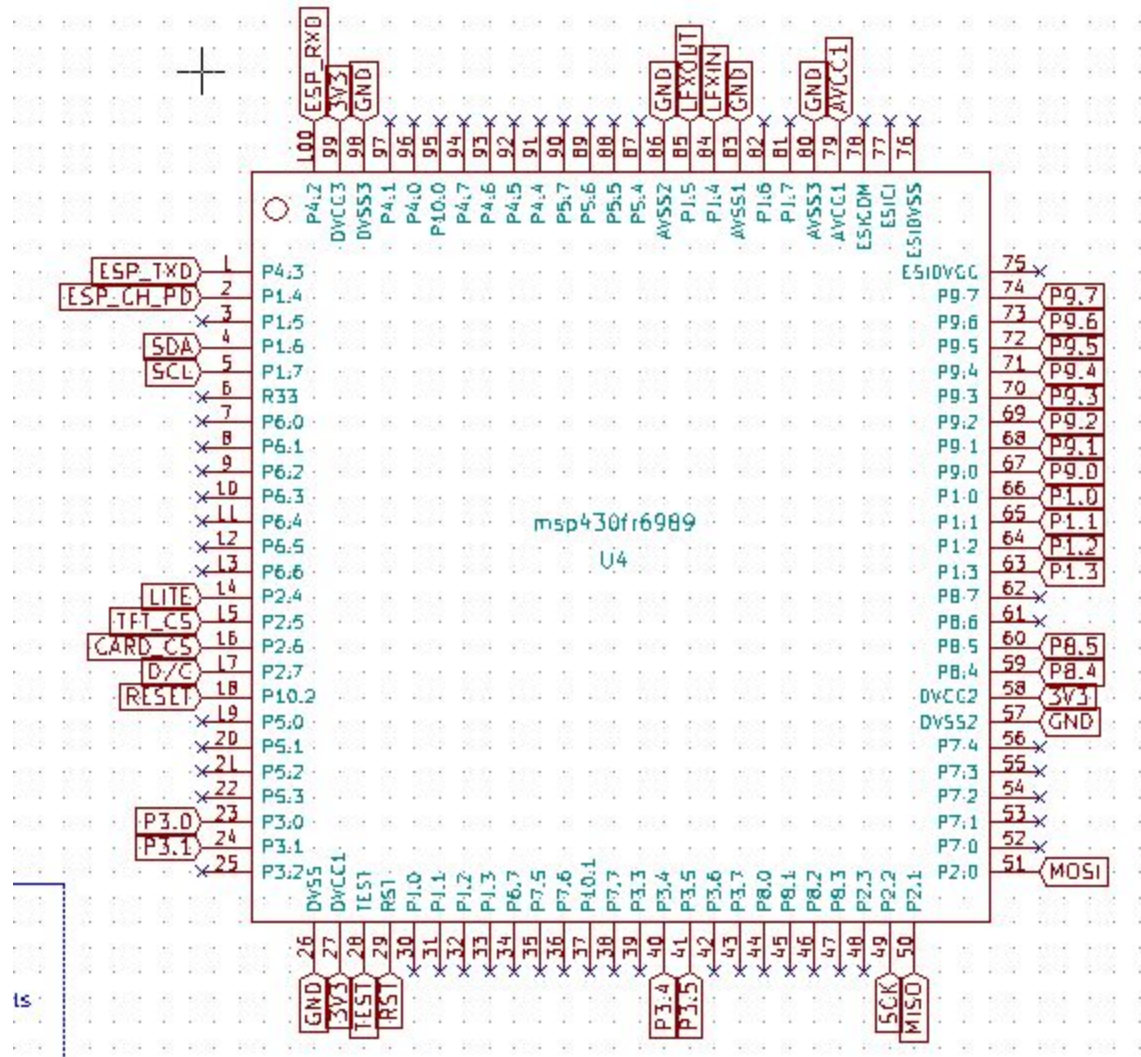
Hi ur / Homtop kod sagom'la'pim'aonncators'usb'abr'SIU'mollen'sensors' sagom'la'odpim'aonncator'abr'CEF°
 blspky' sagom'la'abr'bcacouplinf' a/p'ators'abr'PVSfifi' sagom'la'odpim'aonncators'abr'G/Sz/11 Z lln'mobuc



Hi ur'fi' Qod'to'irif'igt' sagom'la'abr'PESfif/EFCE' sagom'la'odpim'gc^bcs'abr'PVSfifi' prof'r'mninf' ^'rb°
 bc`uff'lnf°



Hi uc'fl Qod'to'rif'gt sagm'tla'od'fif'y'rcf uk'tor'ala'u't sagm'tla'od'f'i'j K(aryst'kabr'PV'Sf'i'i' °



Hi uc'f: Vagm'tla'od'p'm'aom'call'ens'on'PV'Sf'i'i' °

