EGR 314 - Team 205

Final Report

Wearable Robotics: Watching Distance

Samir Strasser, Sai Srinivas Tatwik Meesala,

Noah Blevins, Jose Diaz

April 29, 2022

Arizona State University

EGR 314

Dr. Aukes

Table of Contents

Problem Definition

Problem Statement (Samir)

Stroke patients can not perceive depth as a result of their stroke. The team hopes to solve this issue by giving stroke patients a wearable device that will tell them how far away objects are.

Performance Specifications (Sai Srinivas Tatwik)

The performance specifications for the product are shown in Table 1. Performance specifications with actual values and rationale and plan can be seen in Appendix B.

Table 1. Performance Specifications for Watching Distance

Metric #	Metric	Unit
1	Max Distance Range	m
2	Weight	kg
3	Dimensions	mm x mm x mm
4	Angular Velocity	RPM
5	Voltage input	Volts
6	Screen Brightness	Lumens/Watt
7	Water Resistance	IP
8	Sensor Response time (max)	ms

Pls. see Appendix A for Instructor-defined requirements with actual values.

Design Concept (Samir, Tatwik, Noah)

Visually Engaging Representation

A computer-generated sketch with the device's components is shown in Figure 1.

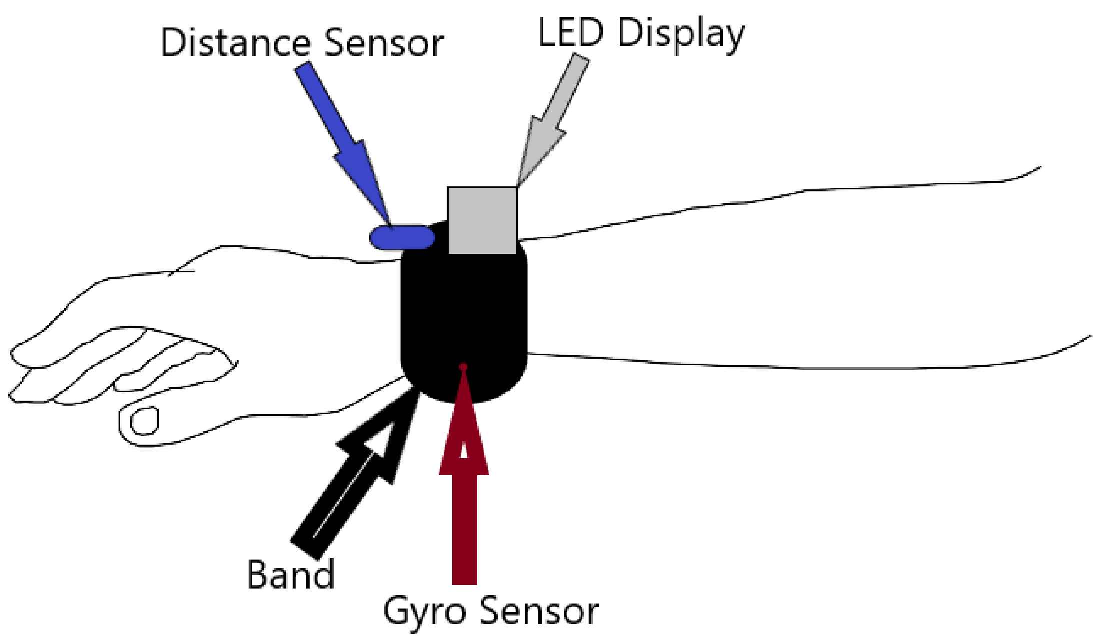


Figure 1. Visually Engaging Representation of Watching Distance

Visual Story-Based Representation

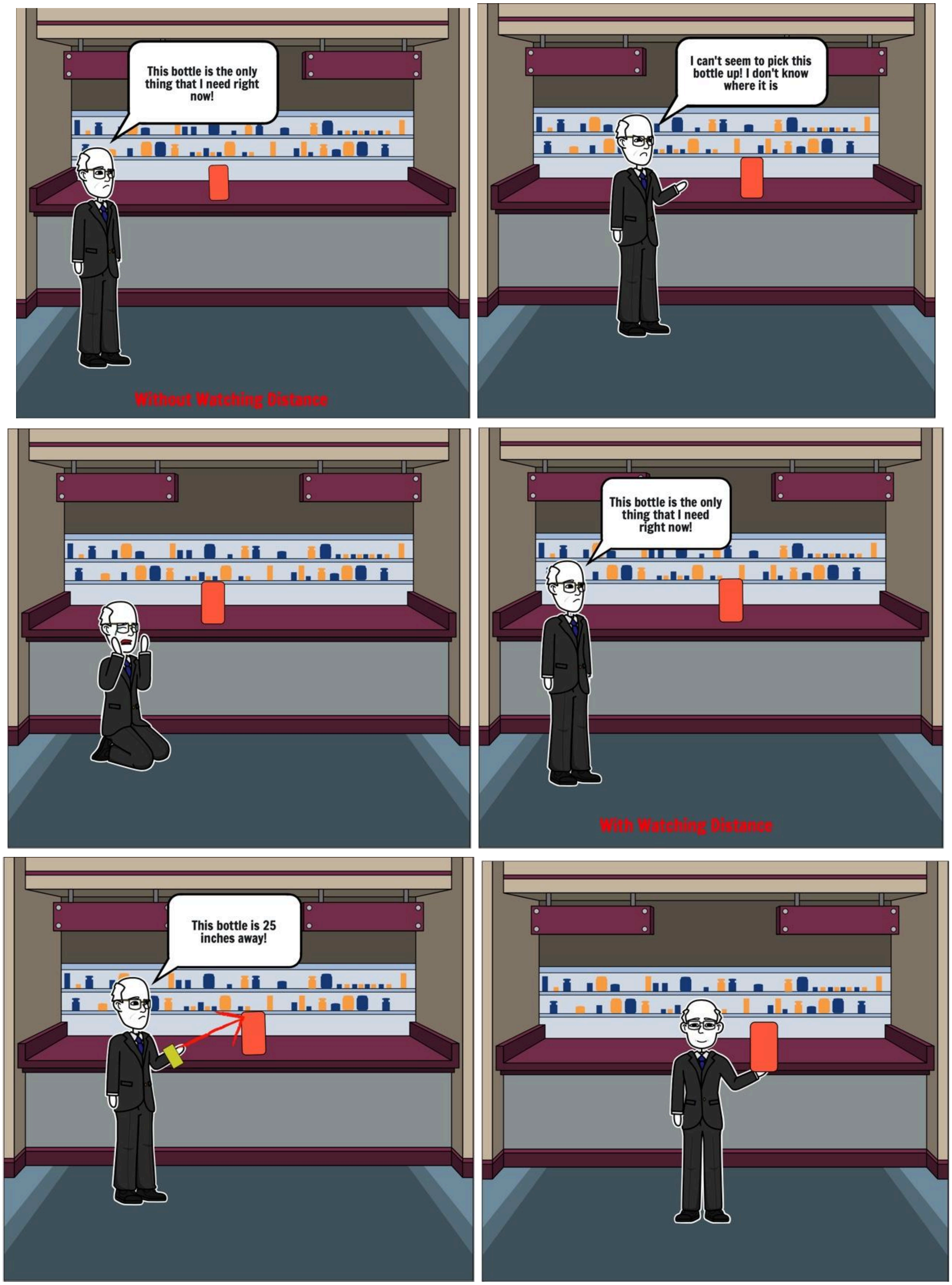
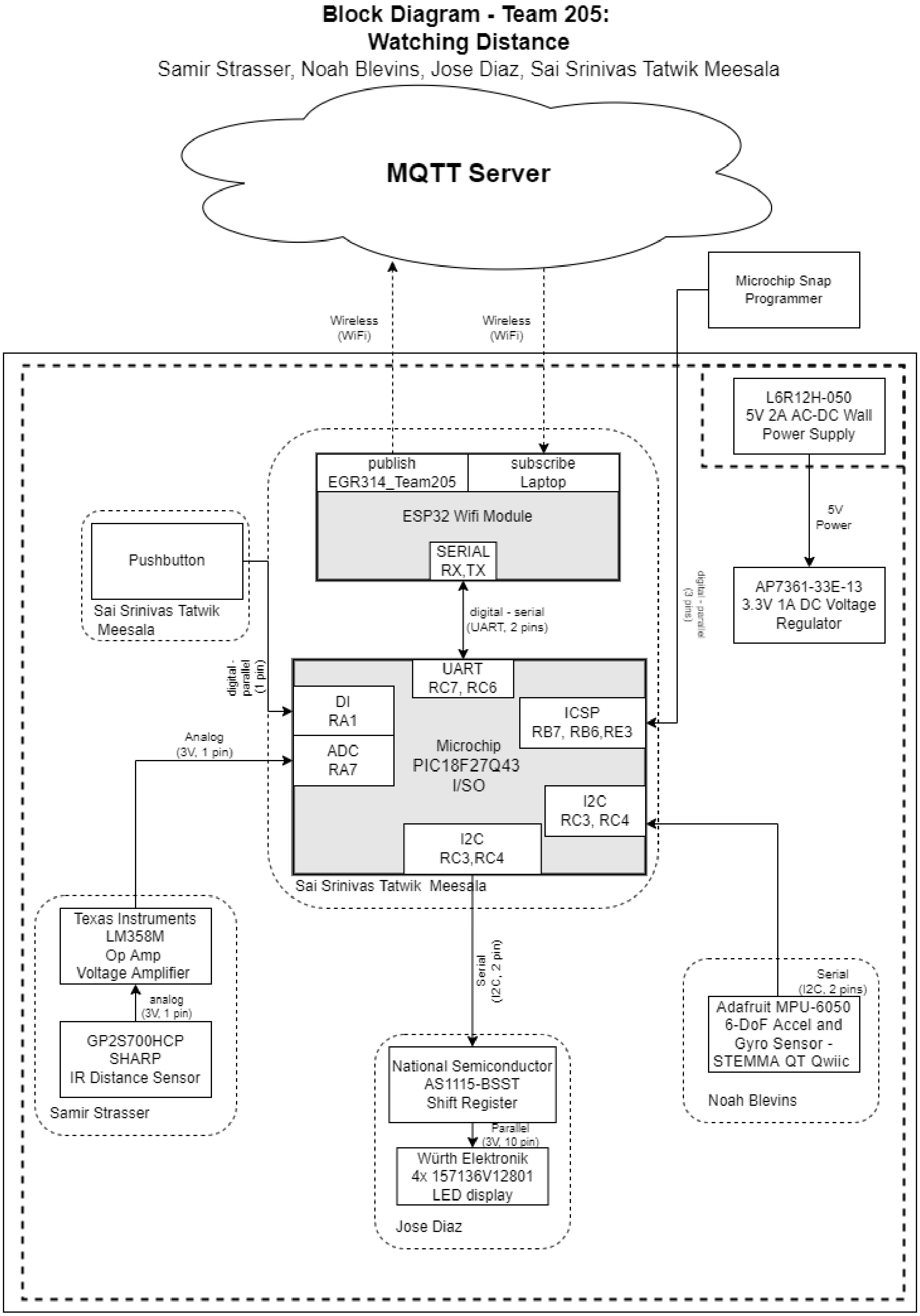


Figure 2. Comic Strip of Watching Distance

Physical Prototype

Figure 3. Physical Prototype of Watching Distance

Block Diagram (All)



Progress, Lessons Learned, and Version 2.0+

Figures 4 represent each team's printed circuit board (PCB) design with physical project photos. Both the front (left image) and back (right image) views are displayed. As shown, several modifications outside of the PCB design were necessary for the LED Display and Distance Measuring device of the system, including many fly wires, breakout board, and external components.

Project Photos

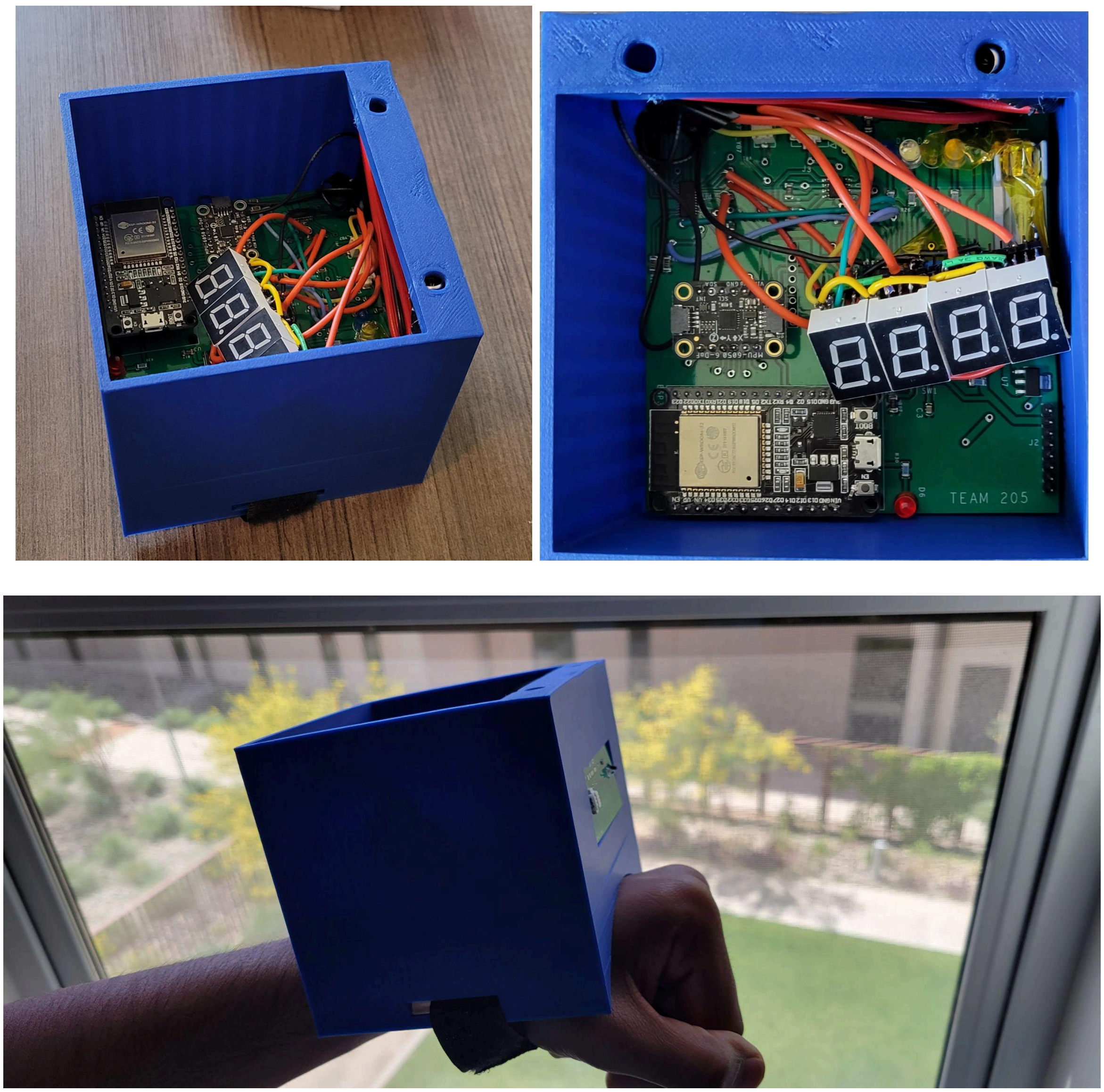


Figure 4. Final Physical Prototype

Hardware Design (Samir)

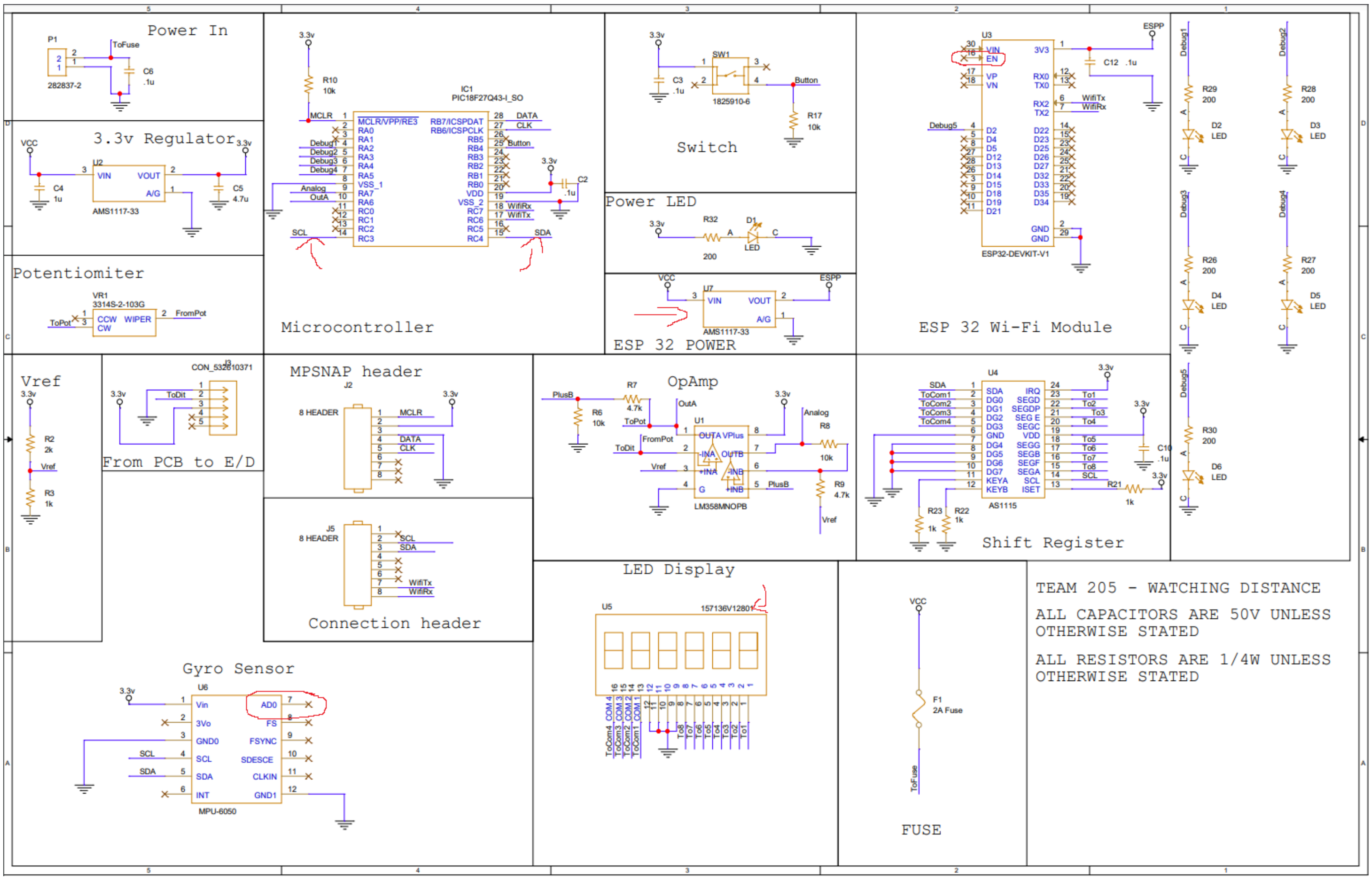


Figure 5. Main Board Schematic

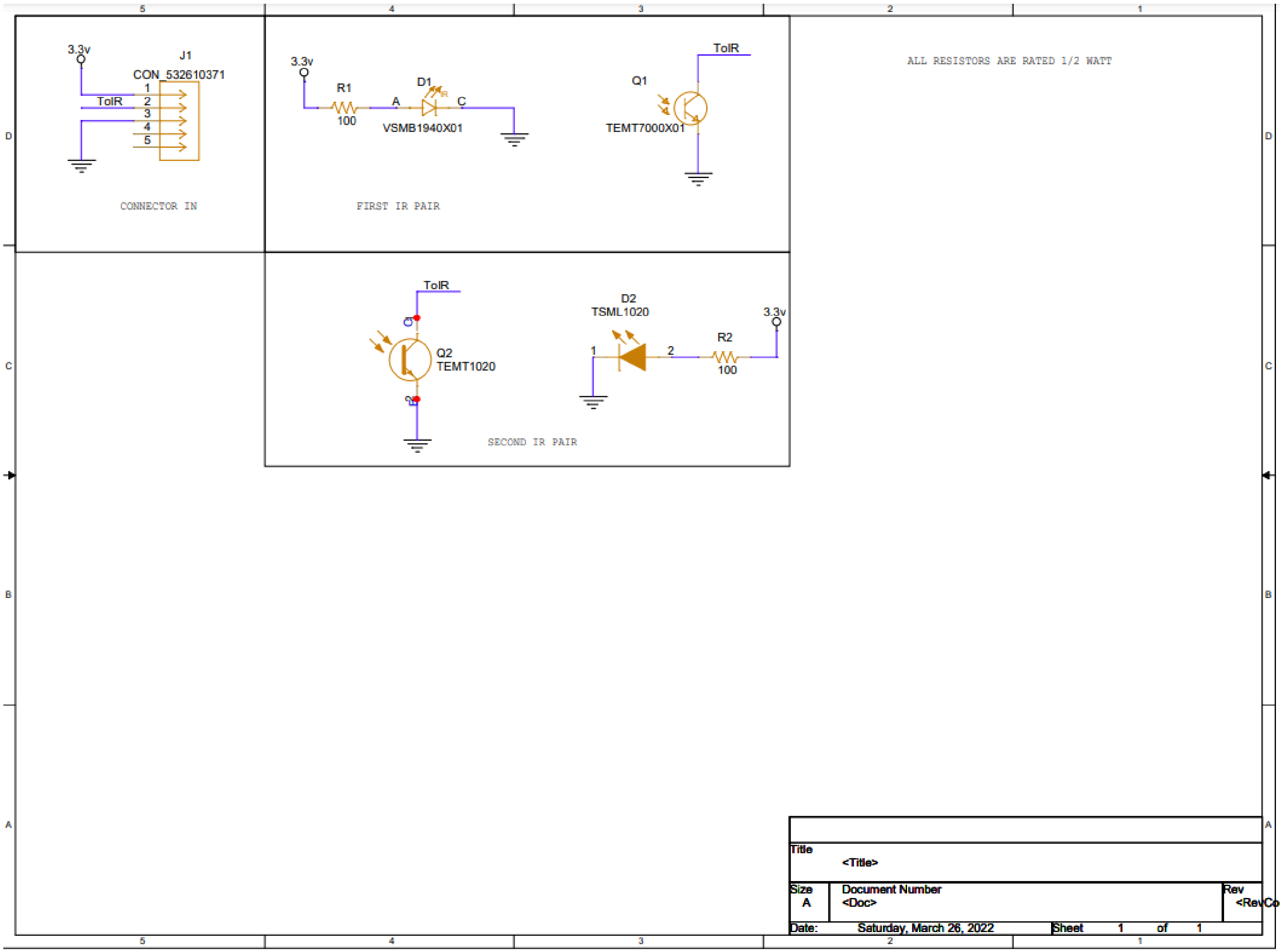


Figure 6. Breakout Board Schematic

Several modifications were made to the hardware design due to time constraints and component compatibility issues.

The five biggest changes to the team's hardware design were as follows:

1) Setting the AD0 pin on the Gyro Sensor to low
    a) Previously, AD0 was running high connected to 3.3V, all this does is change the address from 0x68 to 0x69, as the team only had one sensor there was no reason to have it set high. Now, the AD0 pin is not connected to anything and is marked as N/C.
2) Getting rid of the pull-up resistors on the SDA and SCL lines
    a) The gyro sensor had built-in pull-up resistors, meaning if more were added they would be in parallel and bring the total resistance down.
3) Team had to change the LED display
    a) The previous display had too little documentation and no useful information was available, so the team had to change to a new display that had sufficient documentation and was widely used.
4) The EN pin of ESP 32 needed no external connections.
    a) The EN pin is pulled high within the board; the team originally had a resistor connected to power on it but did not solder the resistor on as it was not needed.
5) Team had to add an extra 3.3v power supply as the display was changed
    a) Previously ESP 32 was powered by 5V. To keep all the subsystems under the same voltage rail, EPS 32 is powered by a 3.3V regulator to avoid any communication errors. The supply powered the ESP 32 alone as it was the device that drew the most current, about 500mA.

See Appendix G for the full PCB and Breakout Board PCB.

Version 2.0: Watching Distance Hardware Design

If version two were to be made of the hardware design, the team would change a few things. First off the team would have the correct footprint for the new LED display team is using. The team may also add a second breakout board that can house the display to run a flexible cable to it and angle it how as needed. This would make housing the device much easier and give us more options with case designs. Next, the team desires to print a housing specifically for the sensor. This would help to get rid of stray light coming into the sensor and messing up the readings. The team could also make the breakout board that houses the sensor components smaller, as this would make mounting it to an enclosure much easier. Presently, it is only big because the team needed room for 2 different sensors as the team did not know which one would work out best.

Now that the team has deeper knowledge, version 2 could be designed on a board that is smaller than .25in*.25in if needed. It would also be better to just use the chip of the ESP 32 so it is not taking up too much space on the board. Of course, this would be a hard task and only something one would do if this was a going-to-market product. If

Team did do this, however, it would save a lot of space on the board and make the watch more practical to wear. A big problem for the board is the fact that it is ~4in*4in meaning it is bigger than a phone. If Team could remake it with 2oz copper to get smaller traces, and use smaller parts then the team could turn this into a real watch. Another thing the team could do would be to make the MPU 6050 not be on a breakout board, this would require us to get soldering done elsewhere as the team does not have the capability to solder just the chip here at ASU. If the team did this however it would help to, again, make the board smaller. As all of our components work well, the biggest thing the team can do for this project is to make it smaller which would then make it more practical. Another way it can be made smaller is with surface mount LEDs, these can be as small as a resistor and just as bright.

Software Design (Sai Srinivas Tatwik Meesala)

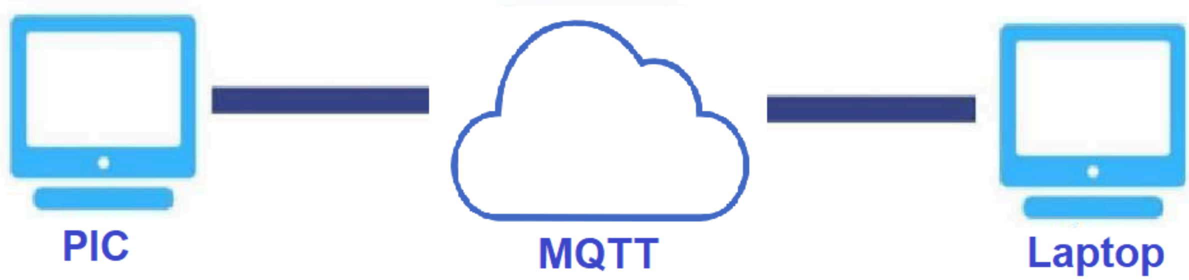


Figure 7 Final network diagram of Watching Distance

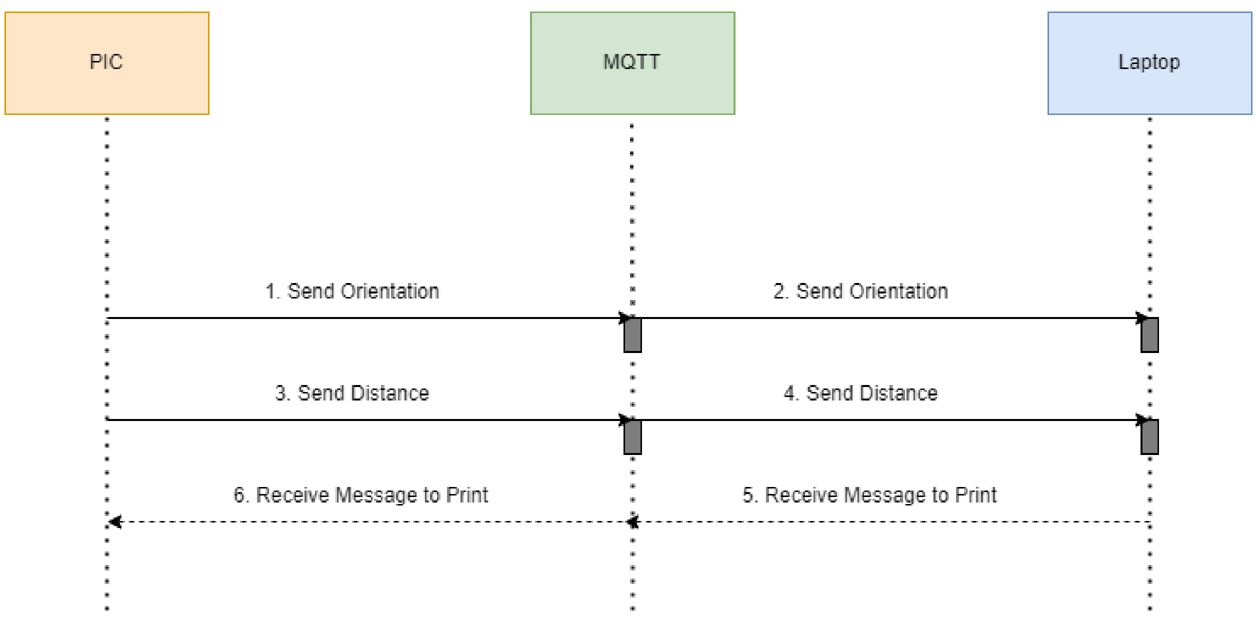


Figure 8 Final UML sequence diagram of Watching Distance

Software Design UML FlowChart

The changes that were made in the UML FlowChart are highlighted with a red marker in the figures below.

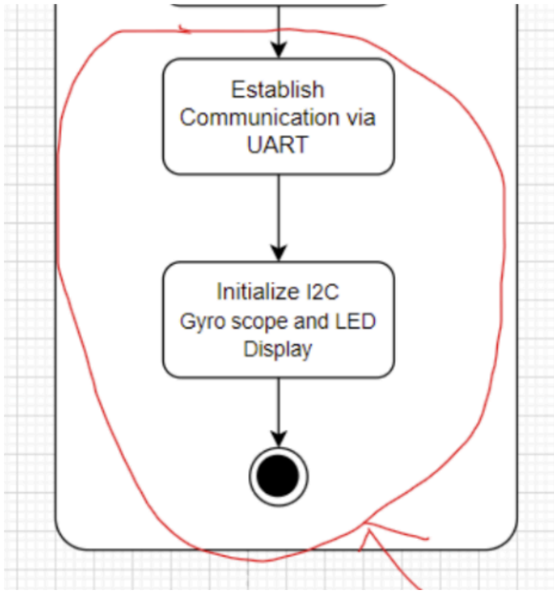


Figure 9 Updated System Initialization

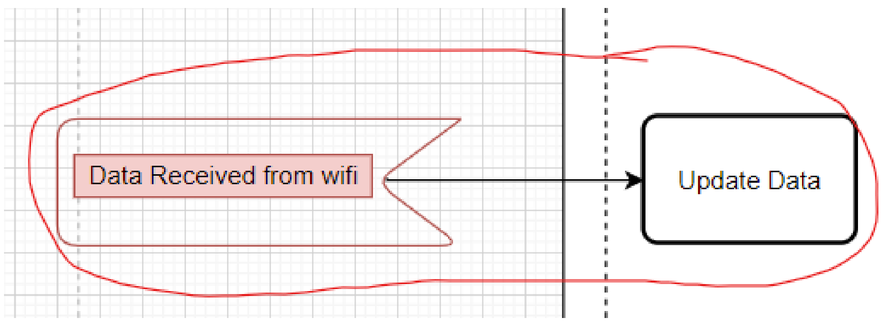


Figure 10 Updated Interrupt Service Routine (ISR)

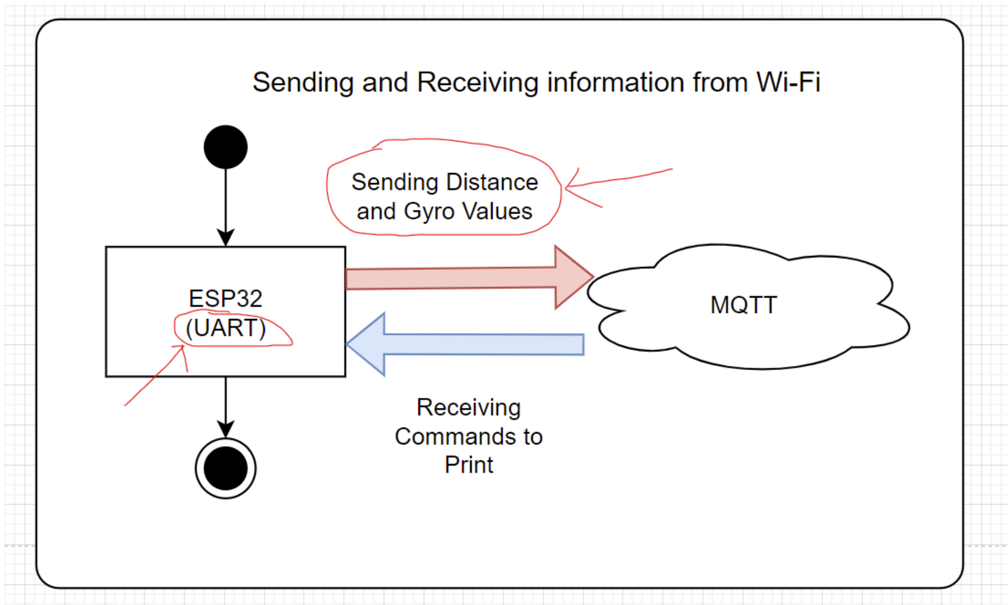


Figure 11 Updated Sending and Receiving information from Wifi

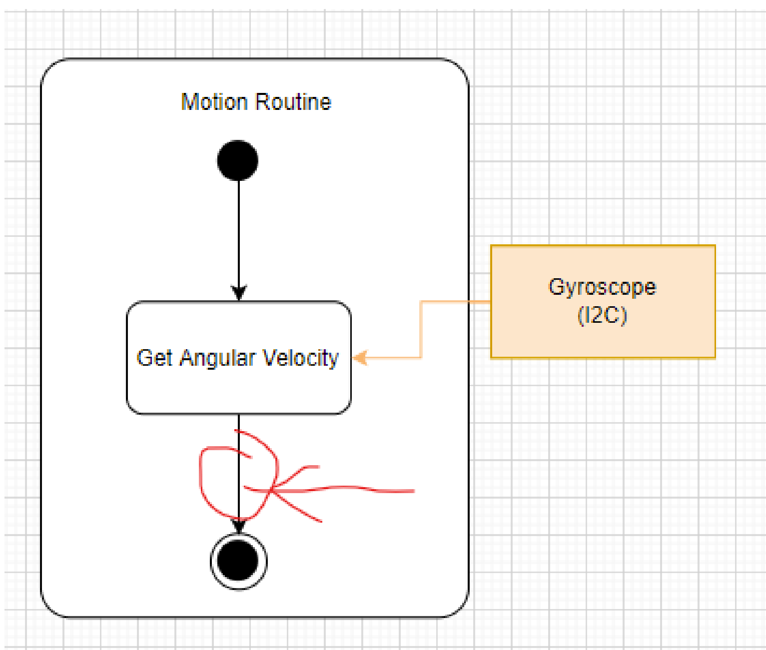


Figure 12 Updated Motion Routine

As with the Hardware Design of the system, several modifications were made to the software design due to time constraints and component compatibility issues. See Appendix J, Appendix K, Appendix L, Appendix M, and Appendix N for the overall project code and Appendix H for the MPLAB code and configurations used.

The five biggest changes to the team's hardware design were as follows:

- Updated System Initialization (see Figure 9)
    1. The team added blocks to initialize UART, Gyroscope, and LED Shift Register that operate on the I2C serial communication protocol. Not initializing the sensors and/or outputs will lead to no output data coming from or going to the subsystem.
- Team Changed changed Hardware Interrupt Service Routine to Software Service Routine (see Figure 10)
    2. The team was initially taught to use a push button to reset the entire system if something failed to function. Now, the team is using Software Service Routine to read the UART values that are sent from the ESP 32. Using the UART interrupt was easy and was also discussed in one of the assignments. To debug this team also added a code to toggle the led when each message is received.
- Sending and Receiving information from Wifi was updated (see Figure 11)
    3. The team initially desired to use UART as well as the I2C serial communication protocol for communication between ESP 32 and PIC Microcontroller. The team later changed it to only UART communication as I2C serial communication protocol was not possible to use.
    4. Team initially sent only Distance values to Wi-Fi, whereas now, the PIC microcontroller sends Distance as well as Gyroscope values to track the user's moment. Tracking both Distance and Gyroscope would be useful for the team to check if the sensors are properly calibrated or not.
- Updated Motion Routine (see Figure 12)
    5. The team previously desired to get Angular Velocity as well as Temperature readings MPU 6050 but the Team was only able to get Gyroscopic values due to software issues. The team was not able to initialize the Gyroscope to output Temperature readings.

Version 2.0: Watching Distance Software Design

Some improvements that could be made in future iterations of a design are:

The team would try to get temperature and acceleration measurements in x, y, and z-direction to monitor the user’s moments. If the team could get the temperature readings, the device can also get ambient temperature. The team can try to implement a complement and/or a Kalman filter to get precise moments and distance readings.

Currently, the measurements are very noisy and inconsistent. This would help the team to automatically read the distance when the hand is pointed toward the object.
The team is also planning to implement a power-saving mode when there is no motion detected for a long time.

The team may also implement an emergency mode if the user (patient) collapses and needs help. This information could be used to alert nearby hospitals. The team is also proposing to add bidirectional Bluetooth connectivity where the device could connect wirelessly to the user's mobile phone app.

Currently, all the calculations for analog sensors are located in main.c. The code would be much cleaner if there was a separate header and source file for it. If the ESP 32 is switched to I2C communication, the device could use a cheaper/less demanding version of the PIC and the device may not need any UART pins. Currently, there are many delays used in the code which prevent other functions from working. In 2.0, the team would design the code where everything would run seamlessly.

If in the future the team chooses to use a battery, a pull-up resistor could be added from the voltage source to display the battery percentage of the device.

Lessons Learned (All)

There are many things that the team learned throughout this project.

1. The team learned that I2C is difficult to get working and that it needs to be worked on constantly.
2. Make sure you have extra header pins on your PCB in case anything goes wrong.
3. Keep your final design specifications in mind during the creation of the project.
4. Read the datasheet for all components, there is often more than one datasheet for each component.
5. The team learned how to adapt the display, changing the component to better suit their needs.
6. The team learned the importance of interrupts as well as how to effectively input them into their future projects.
7. The team learned how to use SPI through the homework, and that SPI is probably better for a shift register.
8. The team learned how to put a silkscreen on a PCB with Cadence Allegro.
9. The team learned how to use Wifi to control the Rx/Tx Lines on the ESP 32.
10. The team learned that even if the datasheet does not have code examples they can be found online for popular components.

Recommendations for Future Students

There are numerous things the team would do differently now that they have experienced and met several challenges and inconveniences during the project.

1. Make sure the whole team understands the project and all the requirements. If there are any doubts about any part of the project make sure to get clarification.
2. Try to buy both surface mount and through-hole versions of all of the parts when it is available. This will let you prototype earlier and if your board has flaws you can work them out with a breadboard.
3. Make sure to not try and go outside the scope of the class. The class is only one semester-long, it may sound like a long time but it will finish very fast. Choose things that are easier than not.
4. Students are recommended to visit office hours if there are any doubts about the homework. Having a TA available will make the time it takes to finish things like homework easier and faster.
5. Always do things as early as possible. Make sure to start prototyping as soon as you can, solder as soon as you can, and do the homework early. Even if your subsystem is easy for you, your teammates might need help as well as it is worth it for your team's final success.

Appendix

The appendices will consist of instructor-defined requirements with actual values, performance specifications with actual values, the major component selection rationale, the power budget, bill of materials, a PCB layout, the top design, and the code used to complete the final product.

Appendix A - Instructor-defined requirements with actual values

#	Metric	Unit	Marginal Value	Ideal Value	Actual Values
0	Embedded (HW + SW) subsystem designed and used in the final device	#	1 by each team member	∞	1
1	Prototype budget ● Does not include development kits, components or PCBs from PRLTA, or components acquired for free	$	$60 / team member	< $60 / team member	$42.8
2	Programming language	programmed in C or C++			c
3	Hardware and/or software-driven interrupts	#	≥ 1	3	1
4	Power supply ● AC adapter, battery, or solar panel connected to a custom-designed voltage regulator circuit. ● Does not count as a subsystem	#	1	1	1
Microcontroller (µC) Subsystem (1 team member responsible for both #5 & #6)					
5	Microchip PIC microcontroller (µC) ● 8- or 16-bit, in-system programmable, MCC compatible surface mount IC ● No PIC16F887, PIC16F917, PIC18F47Q10, or dsPICs allowed	#	1	1	1
6	Wireless communications via ESP32 ● Must communicate bidirectionally with the PIC via a serial protocol (e.g., I^2C, UART)	#	1	1	1
Sensor + Actuator Subsystem(s) (1 team member per sensor/actuator)					
7	Serial sensor(s) read by a µC ● Must communicate with the PIC via a serial protocol (e.g., I^2C, SPI, UART) ● Must use custom software to read, filter (if necessary), and control µC output(s)	#	1	1	1
8	Analog sensor(s) read by a µC ● Must use custom-designed non-trivial hardware amplification and/or filtering ● Must use custom software to read, filter (if necessary), and control µC output(s)	#	1	1	1
9	Actuator(s) controlled by a µC	#	1	1	1

	• Must have a multi-pin digital interface, a custom-designed non-trivial interface circuit, or a filtered analog output • Must utilize complex custom software • Must use sensor data to control μC output(s)				
Additional Requirements (Graded during Live Demonstration)					
10	Functioning custom printed circuit board created in Cadence • No commercial PCBs or breadboards	#	≥ 1per team	4	1 with breakout board
11	Functioning during final demonstration	%	50% functioning (100 points)	100% functioning (200 points)	100%
12	Physical electromechanical prototype	#	≥ 1	# justified by design	1

Appendix B - Performance specifications with actual values

Performance Specifications

Metric #	Metric	Unit	Marginal Value	Ideal Value	Actual Value
1	Max Distance Range	m	<5	<7	0.5
2	Weight	kg	<1.3	<1	0.5
3	Dimensions	mm x mm x mm	<35x35x10	<40x40x15	40x40x15
4	Angular Velocity	RPM	3	5	4
5	Voltage input	Volts	5	5	5
6	Screen Brightness	Lumens/Watt	80	60	80
7	Water Resistance	IP	IPX2	IPX4	none
8	Sensor Response time (max)	ms	<100	<10	10

Rationale & Plan

Specification 1 - Max Distance Range:

Team 205 wants this device to not only work for picking up objects but also for walking around a room. Because of that, the further the sensor can go, the more helpful it will be. When the user points this device at a wall, Team 205 doesn't want them to get a maxed-out reading when Team 205 can help it. To measure this, Team 205, will specify a distance sensor that has ample range.

- **Hardware Impacts:** According to the specifications of our team design may only detect objects up to 50 cm away from the sensors.
- **Software Impacts:** The code will need to be modified to detect movement that is short and long-range.
- **Mechanical Impacts:** No impact

Specification 2 - Weight:

Team 205 does not want this device to be too heavy as sometimes stroke patients have a weaker motor function in their limbs.

To measure this, Team 205 will weigh the device on a scale. The final weight of the included PCB sensors and shell may weigh up to 0.5 Kg.

- **Hardware Impacts:** Components inside the product will need to be light enough for the user to mount them on his/her wrist.
- **Software Impacts:** Components inside the product will need to be light enough for the user to mount them on his/her wrist.

- **Mechanical Impacts:** No impact.

Specification 3 - Dimensions:

Team 205 does not want the dimensions of this device to be too big, as it needs to be placed on a wrist, it needs to be sized accordingly.

To measure this, Team 205 will use a ruler and plan before Team 205 makes the device.

- **Hardware Impacts:** No impact.
- **Software Impacts:** No impact.
- **Mechanical Impacts:** To make the product fit comfortably on the wrist, the final dimensions of this device would be 40mmX40mmX15mm.

Specification 4 - Angular Velocity:

The gyroscope measures the angular velocities from all three dimensions (x,y, and z). This helps us to measure the orientation of the devices by comparing them to the previous orientation.

To measure this, Team 205 needs to find the device, calculate its noise, and plan before Team 205 makes the final device. Our device can measure the angular velocities up to 4 RPM.

- **Hardware Impacts:** No impact
- **Software Impacts:** The code will need to be modified to track motion in an environment.
- **Mechanical Impacts:** The sensor will be placed in the center of the device to get the most accurate readings.

Specification 5 - Voltage Input:

Team 205 wants the product to last an entire day, as the client will need it for their entire day. Because of this, the device will be connected to the wall power supply.

To measure this, Team 205, will use the power budget that Team 205 will create in the future. According to the schematic design, the PCB would require 5V of input voltage.

- **Hardware Impacts:** No impact
- **Software Impacts:** No impact
- **Mechanical Impacts:** No impact

Specification 6 - Screen Brightness:

Team 205 wants the device to be able to be seen in the sunlight, in case the client is outside, therefore Team 205 needs it to be bright enough to accommodate that.

Team 205 can measure this by spacing a bright enough led panel. With LED lights and the screen, the product may produce up to 80 Lumen/watt.

- **Hardware Impacts:** LED lights and screen on the face of the watch.
- **Software Impacts:** The code will need to be modified to glow the lights when motion is detected.
- **Mechanical Impacts:** No impact

Specification 7 - Water Resistance or Ingress protection

Team 205 does not want the clients to be out of luck when it is raining, therefore Team 205 wants to make sure that the electronics are housed in a way that rain and dust will not affect them.

Team 205 can measure this by either using the product in the rain or spraying it with water. Team 205 is also planning submersion or rainfall simulations. Since the project is still in its initial stage, our product would have no water or dust resistance.

- **Hardware Impacts:** No impact
- **Software Impacts:** No impact
- **Mechanical Impacts:** Waterproof and dustproof insulation will need to be used to ensure components remain dry.

Specification 8 - Sensor Response time (max)

Team 205 wants the time it takes for the distance to get to the led panel to be as short as possible. The shorter it is, the better the user experience will be.

Team 205 can measure this through software. The IR sensor and Gyroscope are connected to PIC via an analog interface and I2C respectively.

- **Hardware Impacts:** No Impact
- **Software Impacts:** Team 205 intends to use the ESP32 (wifi module) to connect to PIC via a UART connection. Team 205 is expecting a sensor max response time of about 10ms.
- **Mechanical Impacts:** No Impact

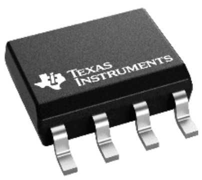
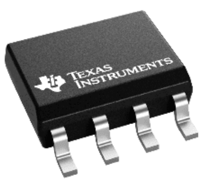
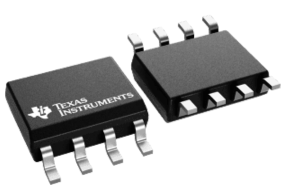
Appendix C - Major Component Selection Rationale

Major Component Selections

Several potential off-the-shelf/Custom electrical and mechanical components that could be used in the project were researched for the device's main subsystems:

IR Distance Sensor Components

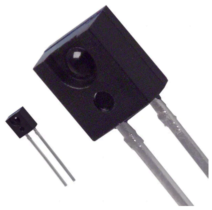
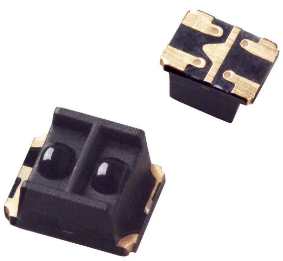
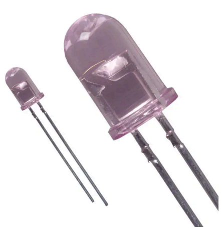
Table 1. Op-Amp for IR sensor

Solution	Pros	Cons
 LM358M [Link] $0.232	• From reviews, this is the best to use for team's application • Inexpensive and easy to use • Many Example Circuits • Reasonably sized	• Sometimes hard to find surface mount versions • Lower Hz Rating
 LM358-N [Link] $0.243	• Higher frequency op Amp • Inexpensive and easy to use • the Same size as the other op Amp	• Don't need this good of an op-amp • Needs a lot more voltage
 LM358B [Link] $0.240	• Higher frequency op Amp • Inexpensive and easy to use and mount	• Over the top for team's needs • Needs a lot of input voltage

Selection: The chosen Op-Amp is **LM358M**

Rationale: LM358M is the most commonly used, and has the best data sheets, making it easier to use.

Table 2. IR Distance sensor

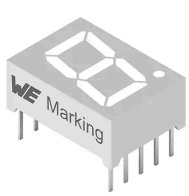
Solution	Pros	Cons
 QSE114 Link $.76	• All in one makes it easy to use • Very small range	• Not sure if an all in one would fall into the scope of this project • The documentation of datasheet is vague
 GP2S700HCP Link $1.76	• Surface Mount, Meaning the circuit that can be used can be the one team used in the homework • Small and compact, will be good for overall design quality	• Small range, maybe too small • Very small device, might be hard to solder
 QED223 Link $1.00  WP7113PD1BT/BD-P22 Link $0.86	• Both form Digikey, so they will get here at the same time • Separation makes the circuit more complex	• Datasheet only shows the size of components, nothing more • No reviews on Digikey

Selection: The chosen IR Sensor is GP2S700HCP

Rationale: The component is from a reputable source, it will make the circuit design easier, and will be smaller

LED Display Components

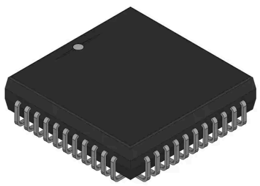
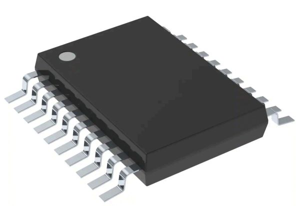
Table 3. LED Display

Solution	Pros	Cons
 LCD-A2X1C50TR Link $3.18	• Works at a 5V voltage • Simple pinout • Small (30mm x 26.17mm)	• Not surface mount • Only 2 full digits and a number 1 display.
 OD-358R Link $3.44	• Works at a voltage of 5V • Simple pinout • 3 full digits and a number 1 display with colons and low battery characters. • Good size for a wearable device (50.8mmx30.48mm)	• Not surface mount • Comes with plastic screen, won't be able to operate at high-temperature
 157136V12801 Link $0.98	• Works at voltages from 3.3 to 5V • Very Simple to use. • Works with the chosen controller.	• Hard to solder for the team's use. • Very large. • Need 4 of them

Selection: The Würth Elektronik 7-segment display 157136V12801.

Rationale: The 157136V12801 works with the controller the team is using and can be expanded as needed by adding more to the board.

Table 4. Shift Resistor for LED Display

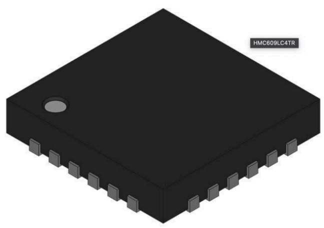
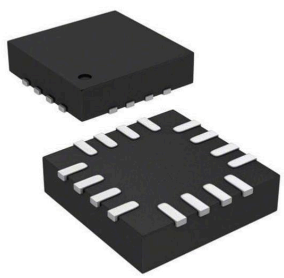
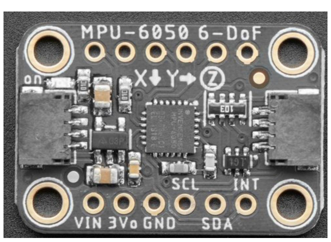
Solution	Pros	Cons
 MM5452V Link $4.92	• Can drive four 7-segment displays with decimal point • Uses SPI • Works with a voltage range of 3V to 10V	• Hard to acquire • Only sold in packs of 61 on most sites.
 AS1115-BSST Link $4.19	• Can drive eight 7-segment displays • Uses SPI as well as I2C communication • Works with voltage range 2.7V to 5.5V	• Can only display 0-9,-, and “HELP” and A-F • Complicated to use.
 SN74LS674DW Link $27.72	• Can drive multiple 7-segment displays at a time using the cathode. • Works with voltage range 4.5V to 5.5V	• Expensive • Can only be bought in packs of 25

Selection: The chosen Shift resistor is **AS1115-BSST.**

Rationale: The AS1115-BSST works with the display panel that was chosen. The circuit would not be as simple as using any of the other two options, but it is easy to get and not too expensive.

Gyro Sensor

Table 5. Gyro Sensor

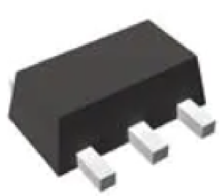
Solution	Pros	Cons
 FXAS21002CQR1547 Link $2.28	● 3 axis sensor ● Inexpensive ● High sensitivity	● Difficult to get ● Very small (4mmx4mm)
 A3G4250DTR Link $18.86	● Low Voltage ● 3 axis sensor ● Low noise rate density	● Small (4mm x 4mm) ● Expensive ● Needs to be connected to itself (see page 8 of the datasheet)
 Adafruit MPU-6050 6-DoF Accel and Gyro Sensor - STEMMA QT Qwiic Link $6.95	● Both accelerometer and gyro sensor ● Already on daughterboard ● Relatively inexpensive	● Last updated 2019 ● 7V logic

Selection: The chosen Gyro sensor is Adafruit **MPU-6050 6-DoF Accel and Gyro Sensor - STEMMA QT Qwiic.**

Rationale: This is the only IMU Team that was able to find one that is in stock and on a daughterboard, as well as having both an accelerometer and gyroscope sensor.

3.3V Regulator

Table 5. Voltage Regulator

Solution	Pros	Cons
 Diodes Incorporated AP7361-33E-13 Link $0.67	• Easy to mount, as it is big enough • Comes with footprint • Can output up to 1A	• Low max input voltage at 6V • Not recommended for new device
 BD33C0AHFP-CTR Link $2.21	• Max Input Voltage: 26.5 V • Operating temperature between -40 to 125 C • Can output up to 1A	• Too many complex pins for voltage regulator • Expensive
 Nisshinbo R1173H331B-T1-FE Link $1.25	• As it is big, it is easy to mount • Comes with footprint • Affordable • Can output up to 1A	• Expensive when compared with other competitors • Long factory lead time of 21 weeks

Selection: The chosen Voltage regulator is **AP7361-33E-13.**

Rationale: The AP7361-33E-13 outputs 3.3V with a 1A current. It can be shipped immediately and has a detailed datasheet.

Wall Power Supply

Table 6. Voltage Regulator

Solution	Pros	Cons
 Kaga Electronics KTPS05-03315U-VI-P1 [Link] $9.61	• Comes with a Barrel Plug • Team 205 already has the component	• Outputs only 3.3V. The team needs 5V. • Max output current is only 1.5A • Too expensive
 Tri-Mag, LLC L6R12H-050 [Link] $7.7	• Comes with a Barrel Plug • Can output up to 2A with 5V • Team 205 already has the component	• Only 800 in stock • expensive
 Nisshinbo 5V 3A Charger Power Cord UL Listed 15W [Link] $15.99	• Comes with various Connector sizes • Can output up to 3A with 5V	• Very Expensive • Does not have official manufacturer's datasheet

Selection: The chosen Voltage regulator is **Tri-Mag, LLC L6R12H-050.**

Rationale: The AP7361-33E-13 outputs 5V with 2A current, which is sufficient for the team's needs. It can be shipped immediately and has a detailed datasheet. Team 205 already has the component

Appendix D - Microcontroller Selection Rationale

Project-Specific Requirements

PIC18F27Q43, PIC18F26Q43, PIC18F15Q40

1. Determine your project-specific requirements		**3. Look up specifications in the PIC datasheet**		
Design Considerations	**Team Project-Specific Requirements** from Problem Definition and Block Diagram	**PIC18F27Q43**	**PIC18F26Q43**	**PIC18F15Q40**
How many GPIO Pins?	10	25	25	18
Built-in Analog to Digital Converter? How many?	1	1	1	1
Built-in Hardware PWM? How many?	0	3	3	6
Built-in I2C? SPI? How many?	1 i2c, 2 SPI	I2C:1 SPI:2	I2C:1 SPI:2	I2C:1 SPI:2
Built-in UART? How many?	1	5	5	3
Other Required Built-In Features? *(optional)*	(None)			
Additional considerations specific to your project specifications *(optional)*	(None)			

Three Microcontrollers

2. Find 3 microcontrollers that meet your team project-specific requirements and find information on each				
Microcontroller Considerations	**Instructions**	**PIC Option 1**	**PIC Option 2**	**PIC Option 3**
Part Number	*Include the entire part number (leave off any letters at the end that specify the package type)*	**PIC18F27Q43**	**PIC18F26Q43**	**PIC18F15Q40**
Link (URL) to product page	*Do not paste links directly into the table. Instead, link them like this.*	**Link**	**Link**	**Link**
Links (URL) to Data Sheets		**Link**	**Link**	**Link**
Links (URL) to Application Notes	*Often provided by manufacturers to give you specific examples of how to use their products. Search for them in the search bar on the Microchip's website.*	**Link**	**Link**	**Link**
Links (URL) to Code Examples		**Link**	**Link**	**Link**
Links (URL) to External Resources	*Search on Google and YouTube for other resources for each specific microcontroller.*	**Link**	**Link**	**Link**
Production Unit Cost	*Find in the Microchip online store, or Digikey*	**$1.69**	**$1.49**	**$1.47**

Supply Voltage Range	*Find in the microcontroller datasheet*	**1.8 - 5.5V**	**1.8 - 5.5V**	**1.8 - 5.5V**
Absolute Maximum Current for entire IC	*Find in the microcontroller datasheet*	**250mA**	**250 mA**	**350 mA**
Maximum GPIO Pin Current (Source/Sink)	*Find in the microcontroller datasheet*	**50mA**	**50mA**	**50mA**
8-bit or 16-bit Architecture	*Find in the microcontroller datasheet*	**8-bit**	**8-bit**	**8-bit**
Available IC Packages / Footprints	*Find in the microcontroller datasheet. Choose a microcontroller with both surface mount and DIP/through-hole packages available. See Most Common Mistakes below for requirements to improve manufacturing reliability.*	Surface mount: PIC18F27Q43 I/SO DIP: PIC18F27Q43-I/SP **On this Website**	Surface mount: PIC18F26Q43-I/SS DIP: PIC18F26Q43 SP (Currently, production only available in Surface mount) **Link**	Surface mount: PIC18F15Q40-I/SS DIP: PIC18F15Q40-E/P **Link**
Supports External Interrupts?	*Find in the microcontroller datasheet*	**Yes**	**Yes**	**Yes**
In-System Programming Capability and Type	*Allows for programming the microcontroller without removing it from the PCB. Find in the microcontroller datasheet.*	**Yes, Serial**	**Yes, Serial**	**Yes, Serial**

Programming Hardware, Cost, and URL	*Find on the microcontroller product page*	**MpLab Snap - $31.00**	**MpLab Snap - $31.00**	**MpLab Snap - $31.00**
Works with MPLAB® X Integrated Development Environment (IDE)?	*Required. See Microchip Development Tools*	**Yes**	**Yes**	**Yes**
Works with Microchip Code Configurator?	*Required. Go to the MCC website, click the "Current Download" tab, and view the appendix in the release notes associated with the microcontroller you are considering.*	**Yes**	**Yes**	**Yes**

Find how each microcontroller meets your project-specific Requirements

When choosing microcontrollers team 205 made sure to make sure all of the microcontrollers were in the minimum range that was set above. Each controller gives the correct pins to control and get readings from a serial sensor, get readings from an analog input, as well as control an actuator, in team 205's case an extra serial output. All of the microcontrollers also had enough UART pins to be able to talk to the ESP 32 so that wifi communication would be possible. Team 205 also made sure they were all surface mountable and came with footprints we could import into Cadence. In terms of example code, team 205 always felt more was better, and all microcontrollers chosen have ample documentation. All three microcontroller families are available in both DIP/through-hole and surface mount packages.

Rankings

4. Write overall pros, cons, and rankings for the chosen microcontrollers				
Overall Pros	*Write at least 2 for each microcontroller*	**Only 28 pins, so it will be easy**	**Generous spacing between**	**Only 20 pins, easiest to**

		to solder **Hits all of our requirements** **Fastest Microcontroller out of its class**	**pins (.65mm BSC)** **Meets our requirements** **Only 28 pins, so it will be easy to solder**	**surface mount** **Meets all our requirements** **Comes in multiple package options**
Overall Cons	*Write at least 2 for each microcontroller*	**Low stock needs to be ordered soon.** **The PCB footprint was not on the official website.**	**Comparatively low data EEPROM** **Size and footprint was difficult to find (not on manufactures website)**	**Very low program memory out of all** **Almost no EEPROM memory**
Ranking	*1 = first, 2 = second, 3 = third*	**1**	**2**	**3**

Final Microcontroller

Final Microcontroller Choice: PIC18F27Q43 (Surface mount: PIC18F27Q43 I/SO)
Rationale: PIC18F27Q43 is easy to surface mount and has a program memory size of 128 KB. As mentioned in Project-Specific Requirements, it has 1 I2C, 2 SPI pins, and can support up to 5 UARTs. It had a supply voltage ranging between 1.8v and 5.5v. According to the team's rankings, PIC18F27Q43 was placed first. **PIC18F27Q43 I/SO** belongs to the PIC18F27Q43 family which is the surface-mount microcontroller that is used for the Team board

Appendix E - Power Budget

Team used two 3.3V regulators to power the PCB. The MP SNAP programmer and two 3.3V regulators are powered from the wall power supply (5V). The first voltage regulator powered microcontroller, IR Distance Sensor Diode, Opamp for IR Distance Sensor, ESP 32 Wifi Module, LED display, Gyro Sensor, and the Shift Register which total up to 639mA. The ESP 32 is powered with a second voltage regulator which totals to 500mA. The power supply for voltage regulators meets the safety margin of 25%. In total, the entire PCB used 1239mA of current. The wall power supply supplies about 2.2A which is more than the safety margin of 25%.

Project Name:	**Watching Distance**						
Team Member Names:	**Samir Strasser, Noah Blevins, Jose Diaz, Sai Srinivas Tatwik Meesala**						
Version:	1						
All Major Components	**Component Name**	**Part Number**	**Supply Voltage Range**	**#**	**Absolute Maximum Current (mA)**	**Total Current (mA)**	**Unit**
Microcontroller	Microchip PIC18F27Q43	PIC18F27Q43	3.3V	1	250	250	mA
IR Distance Sensor Diode	GP2S700HCP	GP2S700HCP	5V	1	50	50	mA
Opamp for IR Distance Sensor	Texas Instruments LM358	LM358	5V	1	0.3	0.3	mA
Wifi Module	ESP32 Wifi Module	ESP-WROOM-32	3.3V	1	500	500	mA
LED display	Orient Display OD-358R LCD Glass Panel	OD-358R	5V	1	0.01	0.01	mA

Gyro Sensor	Adafruit MPU-6050 6-DoF Accel and Gyro Sensor - STEMMA QT Qwiic	MPU-6050	3V-5V	1	3.6	3.6	mA
3.3V regulator	Diodes Incorporated AP7361-33E-13 Linear Voltage Regulator	AP7361-33E-13	5V	1	1000	1000	mA
Shift Register for LED display	National Semiconductor MM5452V	MM5452V	5V	1	0.1	0.1	mA
Programmer/Debugger	MPLAB® SNAP	PG164100	5V	1	0.1	0.1	mA
+5V Power Rail	**Component Name**	**Part Number**	**Supply Voltage Range**	**#**	**Absolute Maximum Current (mA)**	**Total Current (mA)**	**Unit**
Programmer/Debugger	MPLAB® SNAP	PG164100	5V	1	100	100	
3.3V regulator	Diodes Incorporated AP7361-33E-13 Linear Voltage Regulator	AP7361-33E-13	5V	2	1000	1139	mA
						0	mA
					Subtotal	1239	mA
					Safety Margin	25%	
					Total Current Required on +5V Rail	1548.75	mA

c2. Source Choice	5V Wall power supply	L6R12H-050	5V	1	2200	2200	mA
	Total Remaining Current Available on +5V Rail					651.25	mA
+3.3V Power Rail	**Component Name**	**Part Number**	**Supply Voltage Range**	**#**	**Absolute Maximum Current (mA)**	**Total Current (mA)**	**Unit**
LED display	Orient Display OD-358R LCD Glass Panel	OD-358R	5V	1	0.01	0.01	mA
Shift Register for LED display	National Semiconductor MM5452V	MM5452V	5V	1	335	335	mA
	Gyro Sensor	MPU-6050	3.3V	1	3.6	3.6	mA
	Microcontroller	PIC18F27Q43	3.3V	1	250	250	mA
IR Distance Sensor Diode	GP2S700HCP	GP2S700HCP	3.3V	1	50	50	
Op Amp for IR distance Sensor	Texas Instruments LM358	LM358	3.3V	1	0.3	0.3	mA
	Subtotal					638.91	mA
	Safety Margin					25%	
	Total Current Required on +3.3V Rail					798.6375	mA

c4. Regulator	+3.3V low-dropout regulator	AP7361-33E-13	+5V - 6V	1	1000	1000	mA
	Total Remaining Current Available on 3.3V Rail					201.3625	mA
+3.3V Power Rail	**Component Name**	**Part Number**	**Supply Voltage Range**	**#**	**Absolute Maximum Current (mA)**	**Total Current (mA)**	**Unit**
WiFi Module	ESP32 Wifi Module	ESP-WROOM-32	3.3V	1	500	500	mA
					Subtotal	500	mA
					Safety Margin	25%	
	Total Current Required on +3.3V Rail					625	mA
c4. Regulator	+3.3V low-dropout regulator	AP7361-33E-13	+5V - 6V	1	1000	1000	mA
	Total Remaining Current Available on 3.3V Rail					375	mA
External Power Source 1	**Component Name**	**Part Number**	**Supply Voltage**	**Output Voltage**	**Absolute Maximum**	**Total Current**	**Unit**

			Range		Current (mA)	(mA)	
Power Source 1 Selection	5V Wall power supply	L6R12H-050	110VAC	5V	2000	2000	mA
Power Rails Connected to External Power Source 1						0	mA
						0	mA
	+3.3V low-dropout regulator	KA78RM33RTF	+5V - 20V	1	1000	1000	mA
	Total Remaining Current Available on External Power Source 1					1000	mA
	Component Name	**Part Number**	**Supply Voltage Range**		**Capacity (mAh)**	**Required By Regulators**	
(Not using a battery)	Battery						
					Battery Life		hours

External Supply Voltage should be determined by the dropout voltage for the highest-voltage regulator (e.g., +14V for a +12V regulator).

If you have multiple units in your design (e.g., a base unit and remote unit) then you need a separate power budget for each unit

Appendix F - Bill of Materials

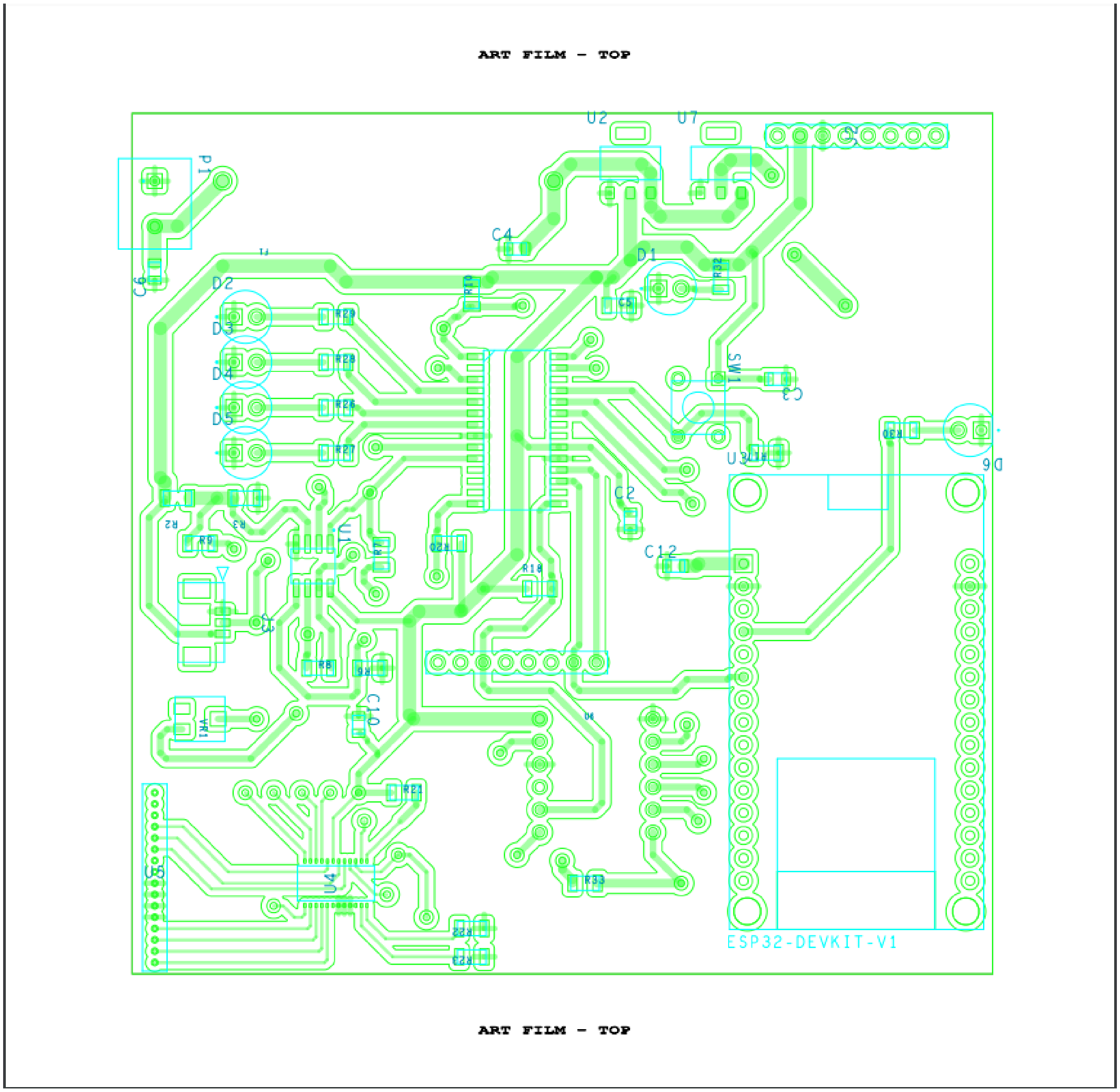
Watching distance: 18/2/22																	
Bill Of Materials	Revision : 1.0																
Team		205															
Name		Samir Strasser, Noah Blevins, Jose Diaz, Sai Srinivas Tatwik Meesala															
Date			2022-02-21														
Item	**Quantity**	**Schematic Reference Designators**	**Part**	**Part Name/Description**	**Unit Prototype Cost**	**Total Prototype Cost**	**Unit Production Cost**	**Total Production Cost**	**Manufacturer**	**Manufacturer Part #**	**Supplier**	**Supplier Part #**	**# Ordered**	**Date Ordered**	**# Received**	**Surplus**	**URL**

1	1	A1	E/D pair	IR Distance Sensor Diode	$1.76	$1.76	$1.76	$1.76	Sharp	GP2S700HCP	Digikey	1855-1020-1	7	2.7.2022	7	6	Link
2	7	C2,C3,C6,C7,C8,C9,C10	.1u			0		0							7	0	
3	1	C4	1u			0		0							1	0	
4	1	C5	4.7u			0		0							1	0	
5	6	D1,D2,D3,D4,D5,D6	LED			0		0							6	0	
6	1	IC1	PIC18F27Q43-I_SO	Microcontroller	$1.65	$1.65	1.65	1.65	Microchip	PIC18F27Q43-I_SO	Microchip	PIC18F27Q43-I_SO	7	2.7.2022	7	6	Link
7	2	J2,J5	8 HEADER	MPSNAP, and Headers	$20	$20	$20	$20	Microchip	PG164100	Microchip	PG164100			2	0	MpLab Snap
8	2	J3,J4	CON_532610371	5 pin jumper		0		0							2	0	
9	1	P1	282837-2	Power Jack		0		0							1	0	
10	1	R2	2k			0		0							1	0	
11	4	R3,R21,R22,R23	1k			0		0							4	0	
12	7	R4,R11,R24,R25,R	200			0		0							7	0	

		26,R27,															
		R28				0		0							0	0	
13	4	R6,R8,R10,R17	10k			0		0							4	0	
14	5	R7,R9,R18,R19,R20	4.7k			0		0							5	0	
15	1	SW1	1825910-6	Switch		0		0							1	0	
16	1	U1	LM358MNOPB	Opamp for IR Distance Sensor	$0.10	$0.10	$0.10	$0.10	Texas Instruments	LM358M	Texas Instruments	LM358M	5	2.7.2022	5	4	Link
17	1	U2	AMS1117-33	Voltage Regulator	$0.54	$0.54	$0.54	$0.54	Diodes Incorporated	AMS1117-33	Digikey		8	2.7.2022	8	7	Link
18	1	U3	ESP32-DEVKIT-V1	ESP32 Wifi Module	$10	$10	$10	$10	Espressif Systems	ESP32-DEVKIT-V1	Mouser	356-ESP32-DEVKIT-V1	1	2.7.2022	1	0	
19	1	U4	AS1115	Shift Register	$4.19	$4.19	$4.19	$4.19	austriamicrosystems AG	AS1115	Digikey	AS1115-BSSTTR	2	2.7.2022	2	1	Link
20	1	U5	OD-6010	LED display	$2.66	$2.66	$2.66	$2.66	Display	157136V12801	Digikey	2544-OD-6010	2	2.7.2022	2	1	Link
21	1	U6	MPU-6050	Gyro Sensor	$6.99	$6.99		0	Microchip	MPU-6050	Adafruit	3886	5	2.7.2022	5	4	Link

22	1	VR1	3314S-2-103G	Potentiometer	$1.90	$1.90	$1.90	$1.90	Bourns	Potentiometer	Mouser	652-3314S-2-103G	5	2.7.2022	5	4	
23	1	F1	Fuse	Fuse Holder					Littelfuse Inc.	03540101ZXGY							
				Total		$49.79		$42.80									

Appendix G - Final PCB ~ Main Board



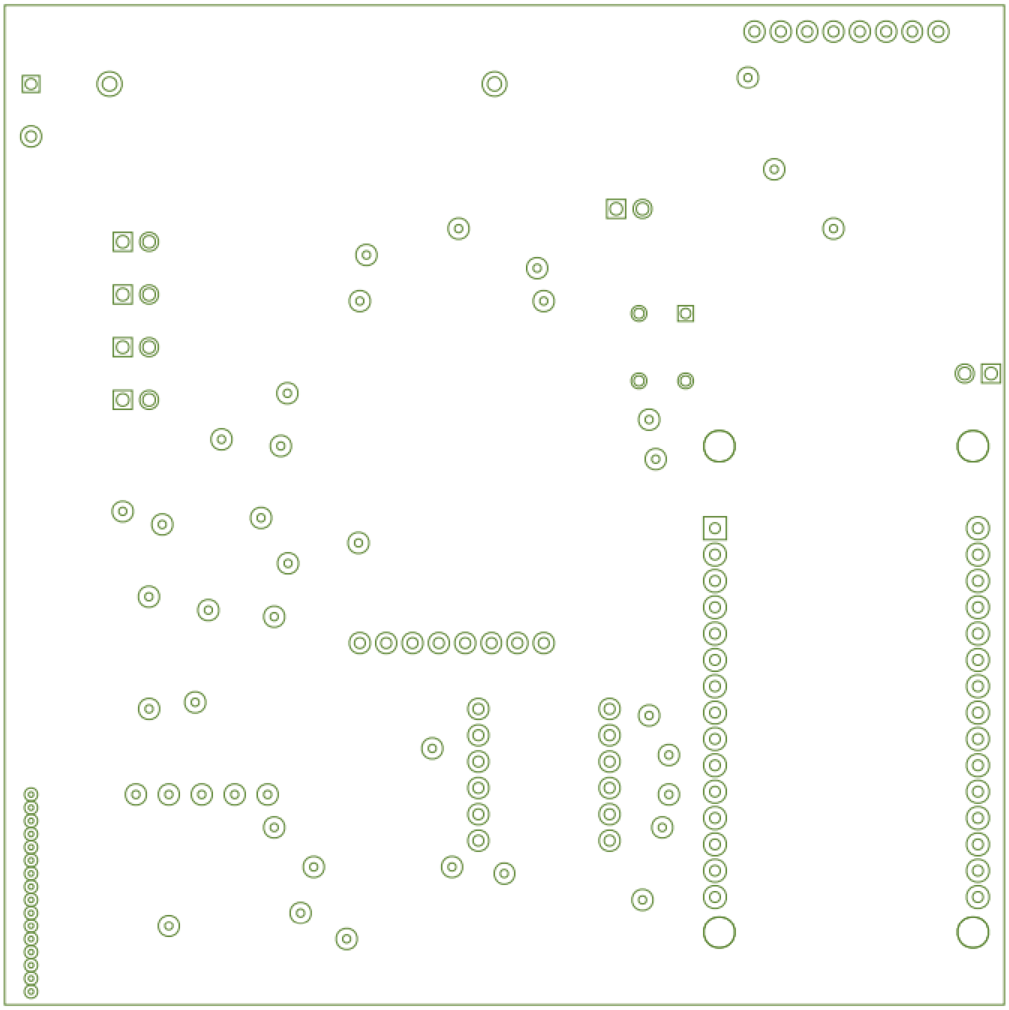
ART FILM - BOTTOM

ART FILM - BOTTOM

ART FILM – OUTLINE

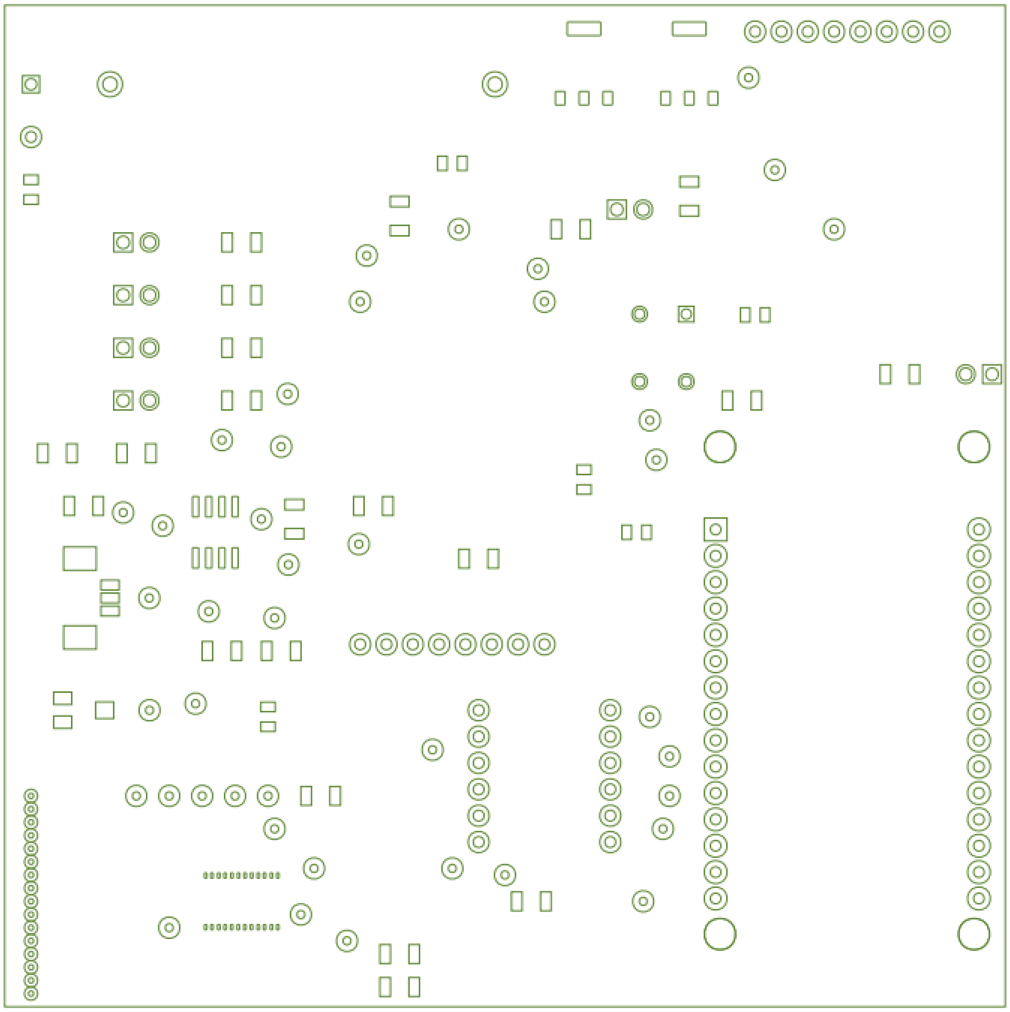
ART FILM – OUTLINE

ART FILM – SOLDERMASK_BOTTOM



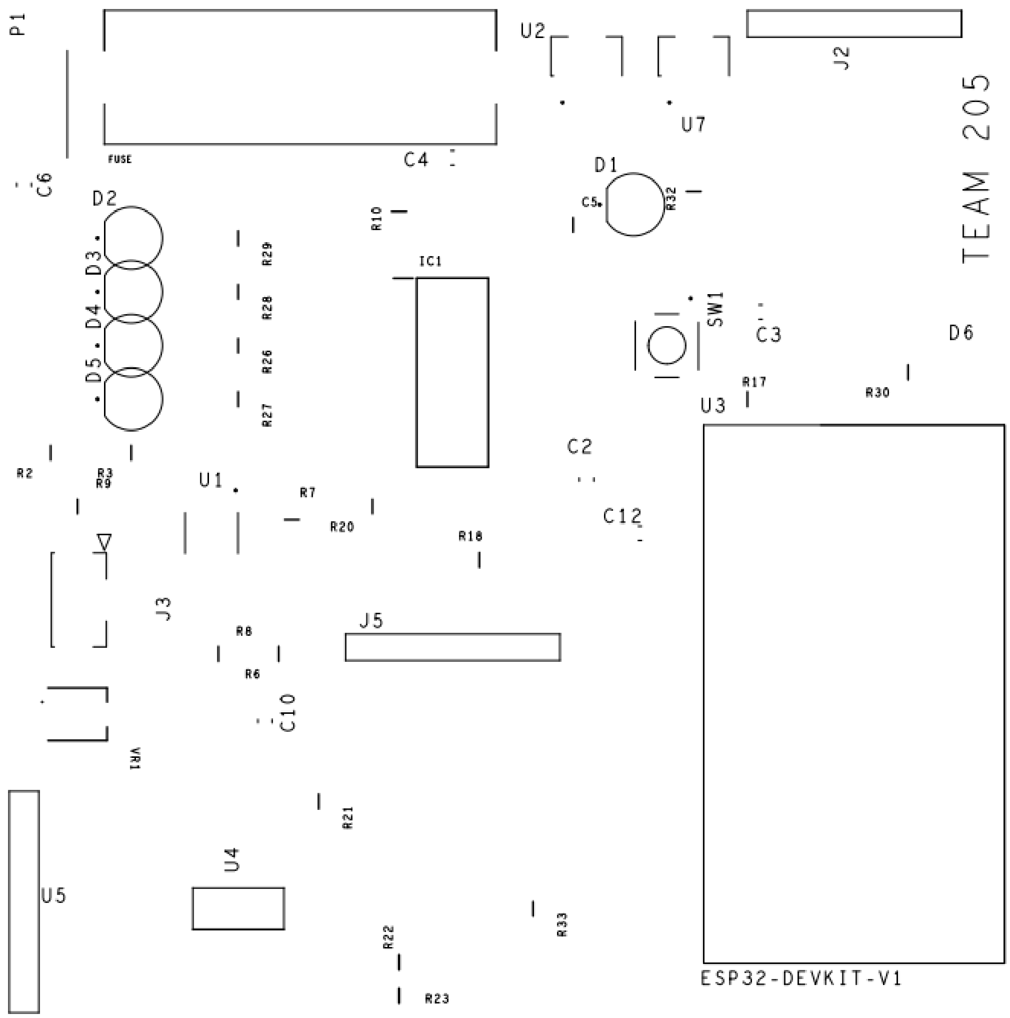
ART FILM – SOLDERMASK_BOTTOM

ART FILM - SOLDERMASK_TOP



ART FILM - SOLDERMASK_TOP

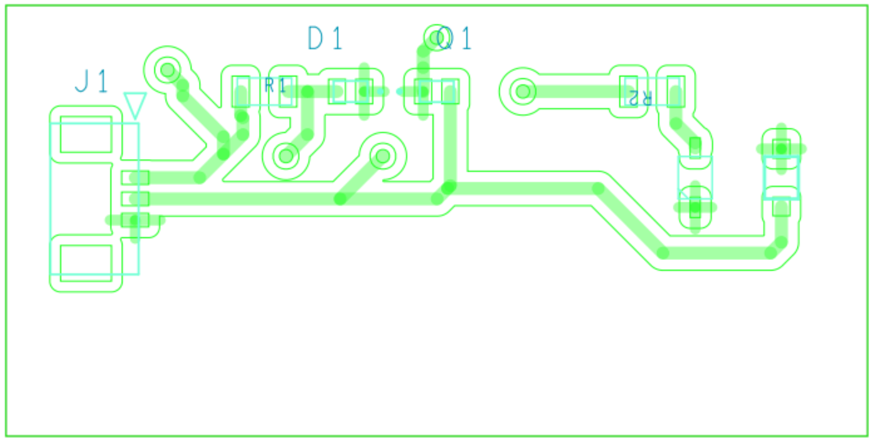
ART FILM - SILKSCREEN_TOP



ART FILM - SILKSCREEN_TOP

Appendix G - Final PCB ~ Breakout Board

ART FILM – TOP



ART FILM – TOP

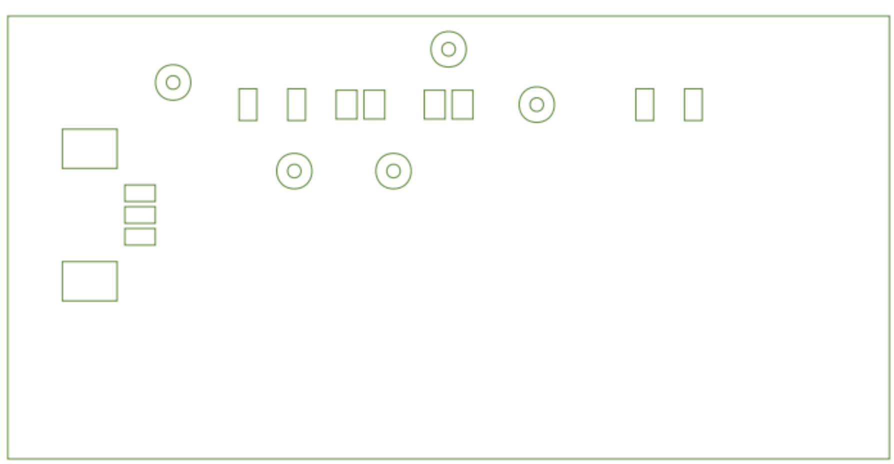
ART FILM – BOTTOM

ART FILM – BOTTOM

ART FILM – OUTLINE

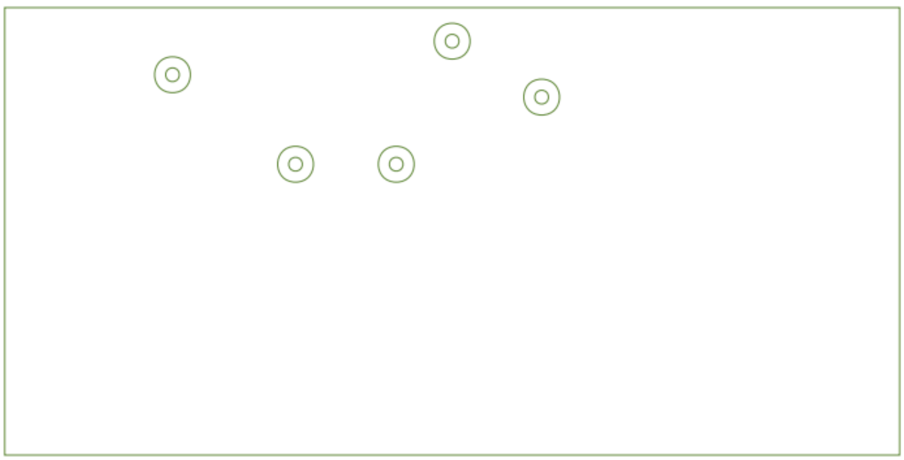
ART FILM – OUTLINE

ART FILM – SOLDERMASK_TOP



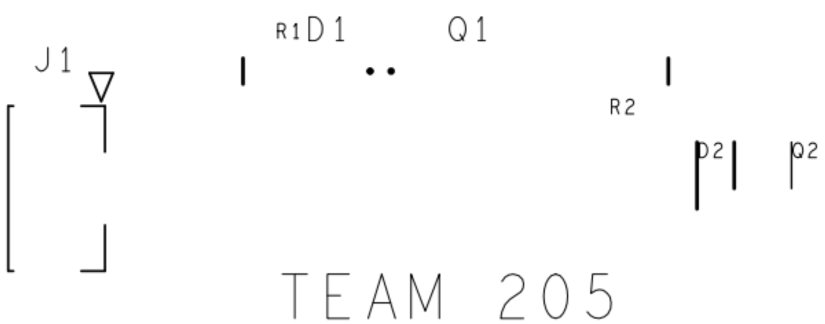
ART FILM – SOLDERMASK_TOP

ART FILM – SOLDERMASK_BOTTOM



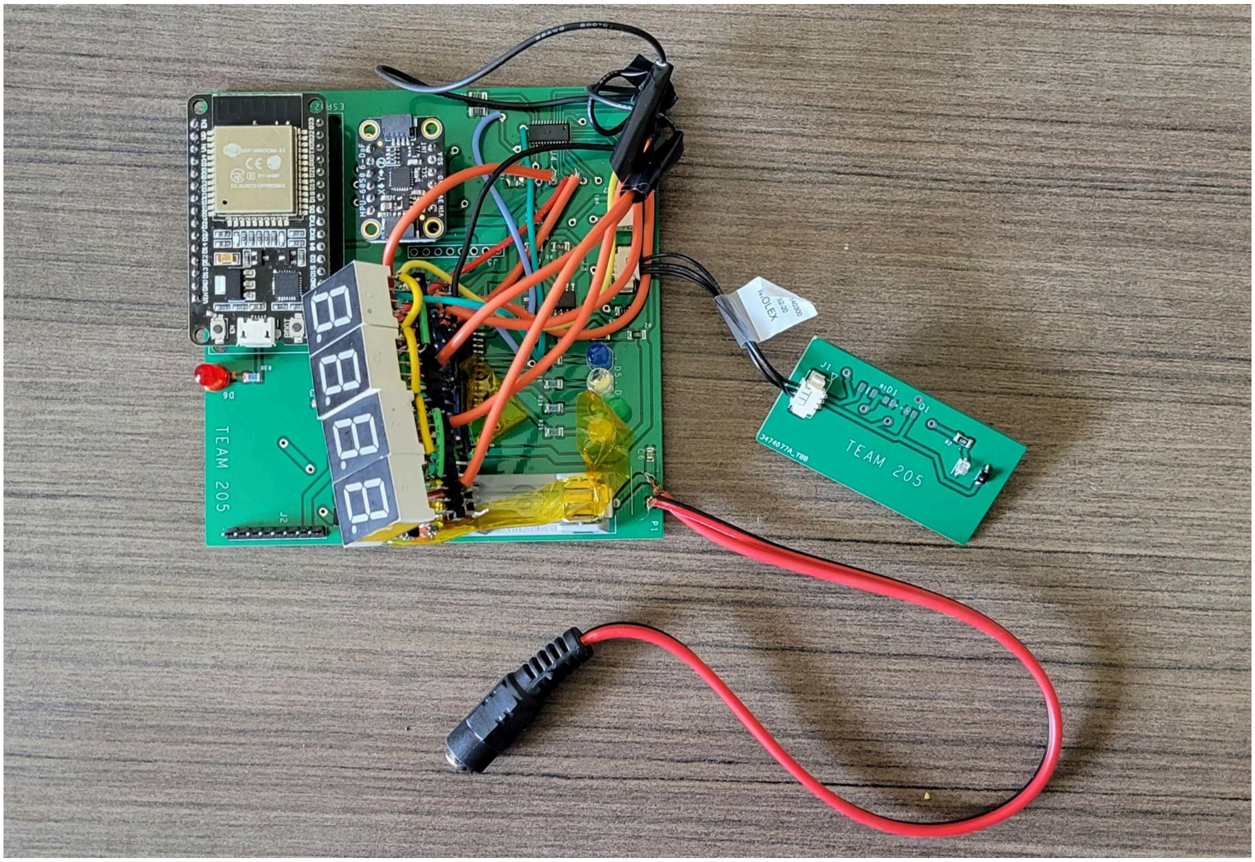
ART FILM – SOLDERMASK_BOTTOM

ART FILM – SILKSCREEN_TOP



ART FILM – SILKSCREEN_TOP

Appendix G - Final PCB: Photos of assembled PCBs



Appendix H - MCC configuration

Microcontroller IDE Project

The MPLAB® X project Top Design with all of the necessary components for the team's project is shown in Figure below. The pins and ports that will be used are shown in Figure below.

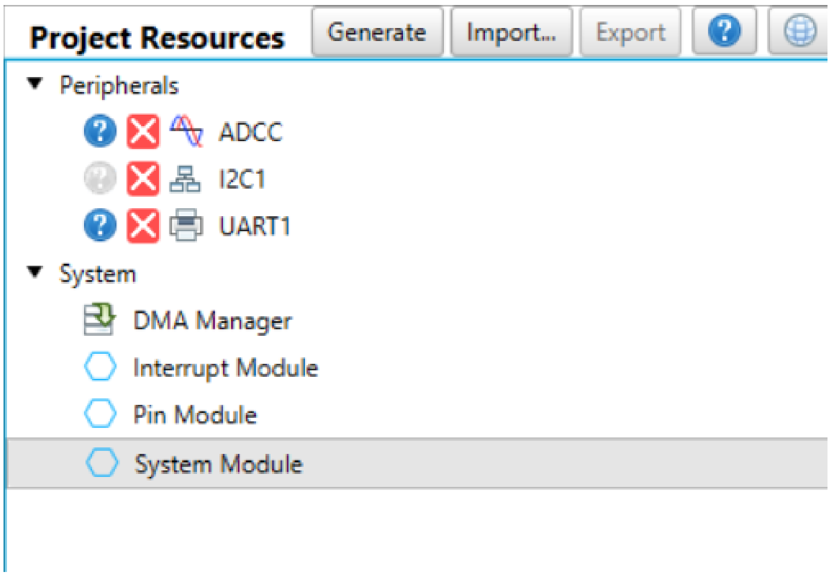


Figure MPLAB® X MCC Configured Project Resources - Flat View screenshot

Selected Package : SOIC28

Pin Name	Module	Function	Custom Name	Start High	Analog	Output	WPU	OD	IOC
RA2	Pin Module	GPIO	LED_Y			✓			none
RA3	Pin Module	GPIO	LED_G			✓			none
RA4	Pin Module	GPIO	LED_W			✓			none
RA5	Pin Module	GPIO	LED_B			✓			none
RA6	ADCC	ANA6	channel_ANA6		✓				none
RA7	ADCC	ANA7	channel_ANA7		✓				none
RB4	Pin Module	GPIO	SW						none
RC3	I2C1	SCL1				✓		✓	none
RC4	I2C1	SDA1				✓		✓	none
RC6	UART1	TX1			✓	✓			none
RC7	UART1	RX1							none

Figure Pins and Ports

Appendix I - Main C Code

```
#include "mcc_generated_files/mcc.h"
#include "mcc_generated_files/examples/i2c1_master_example.h"
#include "mcc_generated_files/i2c1_master.h"
#include "MPU.h"
#include "Display.h"

/*
                        Main application
 */

void MPU_Setup()
{
    i2c1_address_t a = 0x68;
    I2C1_Write1ByteRegister(a,SMPLRT_DIV,0x07);
    __delay_ms(1);
    I2C1_Write1ByteRegister(a,PWR_MGMT_1,0x01);
    __delay_ms(1);
    I2C1_Write1ByteRegister(a,CONFIG,0x00);
    __delay_ms(1);
    I2C1_Write1ByteRegister(a,ACCEL_CONFIG,0x00);
    __delay_ms(1);
    I2C1_Write1ByteRegister(a,GYRO_CONFIG,0x18);
    __delay_ms(1);
    I2C1_Write1ByteRegister(a,INT_ENABLE,0x01);
    __delay_ms(1);
}
```

```
float MPU_Read()
{
    i2c1_address_t ad = 0x68;
    char buffer[40];
    int Ax,AxL,Ay,AyL,Az,AzL,TL,TH,Gx,Gy,Gz;
    float AX, AY,AYL, AZ,AZL, tl,th, GX, GY, GZ,AXL;
    float accD = 16384.0;
    //I2C1_Write1ByteRegister(ad, ACCEL_XOUT_H);

    //Acceleration
    Ax = (int)I2C1_Read1ByteRegister(ad, ACCEL_XOUT_H);
    AxL = (int) I2C1_Read1ByteRegister(ad, ACCEL_XOUT_L);
    Ay = (int)I2C1_Read1ByteRegister(ad, ACCEL_YOUT_H);
    AyL = (int) I2C1_Read1ByteRegister(ad, ACCEL_YOUT_L);
    Az = (int)I2C1_Read1ByteRegister(ad, ACCEL_ZOUT_H);
    AzL = (int) I2C1_Read1ByteRegister(ad, ACCEL_ZOUT_L);
    AX = (float)Ax/accD;
    AXL = (float)AxL/accD;
//    printf("AX_H: %f",AX);
//    printf("     AX_L: %f \n\r",AXL);
//    printf("AY_H: %f",AY);
//    printf("     AY_L: %f \n\r",AYL);
//    printf("AZ_H: %f",AZ);
//    printf("     AZ_L: %f \n\r",AZL);

    //Temp
    TL = (int) I2C1_Read1ByteRegister(ad, TEMP_OUT_L);
```

```
    TH = (int) I2C1_Read1ByteRegister(ad, TEMP_OUT_H);
    tl = (float) (TL/340.00)+36.53;
    th = (float) (TH/340.00)+36.53;
   // printf("T_H: %f",th);
    //printf("     T_L: %f \n\r",tl);

    //Gyro
    Gx = (int)I2C1_Read1ByteRegister(ad, GYRO_XOUT_H);
    Gy = (int)I2C1_Read1ByteRegister(ad, GYRO_YOUT_H);
    Gz = (int)I2C1_Read1ByteRegister(ad, GYRO_ZOUT_H);
    GX = (float)Gx/131.0;
    GY = (float)Gy/131.0;
    GZ = (float)Gz/131.0;
    printf("GX: %f \n\r",GX);
    //printf("GY: %f \n\r",GY);
    //printf("GZ: %f \n\r",GZ);

    return GX;
}



float Conversion(float x)
{
    float y =  (float) (0.0066575727*(x*x) + -0.5089235578*x + 9.856333104);
    return y*100;
}
```

```
float ADC_Read()
{
    float a,b,c,d; //float values to read ADC,D is avg
    adcc_channel_t c1, c2, c3; //used to read ADC
    __delay_ms(5);
    c1 = ADCC_GetSingleConversion(channel_ANA6);
    a = (float) c1 - 33;
    __delay_ms(5);
    c2 = ADCC_GetSingleConversion(channel_ANA6);
    b = (float) c2 - 33;
    __delay_ms(5);
    c3 = ADCC_GetSingleConversion(channel_ANA6);
    c = (float) c3 - 33;

    d = (a+b+c)/3; //find AVG
    return d;
}

uint8_t rxData = 'D';
void Data_Eusart(void)
{
      UART1_Receive_ISR();

if(UART1_is_rx_ready())
        {
            rxData = (uint8_t) UART1_Read();
        }
```

```
LED_W_Toggle();


}
void main(void)
{
    // Initialize the device
    SYSTEM_Initialize();
    MPU_Setup();
    Display_Init();
    INTERRUPT_GlobalInterruptEnable();
//    INTERRUPT_PeripheralInterruptEnable();
    UART1_SetRxInterruptHandler(Data_Eusart);

   // printf("hi2 \n\r");
    i2c1_address_t address = 0x68;
    uint8_t GYROX = 0x41;

    // Enable the Global Interrupts
    //INTERRUPT_GlobalInterruptEnable();

    // Enable the Peripheral Interrupts
    //INTERRUPT_PeripheralInterruptEnable();



    // If using interrupts in PIC18 High/Low Priority Mode you need to enable the Global High and Low Interrupts
    // If using interrupts in PIC Mid-Range Compatibility Mode you need to enable the Global Interrupts
```

```
// Use the following macros to:

// Enable the Global Interrupts
//INTERRUPT_GlobalInterruptEnable();

// Disable the Global Interrupts
//INTERRUPT_GlobalInterruptDisable();


adcc_channel_t conv;
float j;
int k;
float value = 0;
uint8_t b = 0;
I2C1_Open(address);
float GYROTHINGGY;



while (1)
{
   GYROTHINGGY = MPU_Read();
   //__delay_ms(10);
   if(GYROTHINGGY < 1)
   {
      LED_Y_SetHigh();
      Display_Blank();
```

```
    __delay_ms(1000);
}
else
{
    LED_Y_SetLow();
    j = ADC_Read();
    printf("ADC: %3.2f",j);
    __delay_ms(1);
    if(j>39)
    {
        j = 40;
    }
    k = (int) Conversion(j);

    if(k<500)
    {
        LED_G_SetHigh();
    }
    else
    {
        LED_G_SetLow();
    }
    printf("      Adjusted: %d",k);
    printf("      RX: %c\n\r",rxData);



    if(rxData == 'D')
```

```
            {
                Display_Digits(k);
            }
            else
            {
                Display_Help();
                __delay_ms(1000);
                rxData = 'D';
            }
            __delay_ms(500);
        }
    }


}
```

Appendix J - MPU Header Code

```
#ifndef XC_HEADER_TEMPLATE_H
#define      XC_HEADER_TEMPLATE_H

#include <xc.h> // include processor files - each processor file is guarded.
#define XG_OFFS_TC       0x00
#define YG_OFFS_TC       0x01
#define ZG_OFFS_TC       0x02
#define X_FINE_GAIN      0x03
#define Y_FINE_GAIN      0x04
#define Z_FINE_GAIN      0x05
#define XA_OFFS_H        0x06
#define XA_OFFS_L_TC     0x07
#define YA_OFFS_H        0x08
#define YA_OFFS_L_TC     0x09
#define ZA_OFFS_H        0x0A
#define ZA_OFFS_L_TC     0x0B
#define XG_OFFS_USRH     0x13
#define XG_OFFS_USRL     0x14
#define YG_OFFS_USRH     0x15
#define YG_OFFS_USRL     0x16
#define ZG_OFFS_USRH     0x17
#define ZG_OFFS_USRL     0x18
#define SMPLRT_DIV       0x19
#define CONFIG           0x1A
#define GYRO_CONFIG      0x1B
#define ACCEL_CONFIG     0x1C
```

```
#define FF_THR            0x1D
#define FF_DUR            0x1E
#define MOT_THR           0x1F
#define MOT_DUR           0x20
#define ZRMOT_THR         0x21
#define ZRMOT_DUR         0x22
#define FIFO_EN           0x23
#define I2C_MST_CTRL      0x24
#define I2C_SLV0_ADDR     0x25
#define I2C_SLV0_REG      0x26
#define I2C_SLV0_CTRL     0x27
#define I2C_SLV1_ADDR     0x28
#define I2C_SLV1_REG      0x29
#define I2C_SLV1_CTRL     0x2A
#define I2C_SLV2_ADDR     0x2B
#define I2C_SLV2_REG      0x2C
#define I2C_SLV2_CTRL     0x2D
#define I2C_SLV3_ADDR     0x2E
#define I2C_SLV3_REG      0x2F
#define I2C_SLV3_CTRL     0x30
#define I2C_SLV4_ADDR     0x31
#define I2C_SLV4_REG      0x32
#define I2C_SLV4_DO       0x33
#define I2C_SLV4_CTRL     0x34
#define I2C_SLV4_DI       0x35
#define I2C_MST_STATUS    0x36
#define INT_PIN_CFG       0x37
```

```
#define INT_ENABLE        0x38
#define DMP_INT_STATUS    0x39
#define INT_STATUS        0x3A
#define ACCEL_XOUT_H      0x3B
#define ACCEL_XOUT_L      0x3C
#define ACCEL_YOUT_H      0x3D
#define ACCEL_YOUT_L      0x3E
#define ACCEL_ZOUT_H      0x3F
#define ACCEL_ZOUT_L      0x40
#define TEMP_OUT_H        0x41
#define TEMP_OUT_L        0x42
#define GYRO_XOUT_H       0x43
#define GYRO_XOUT_L       0x44
#define GYRO_YOUT_H       0x45
#define GYRO_YOUT_L       0x46
#define GYRO_ZOUT_H       0x47
#define GYRO_ZOUT_L       0x48
#define EXT_SENS_DATA_00  0x49
#define EXT_SENS_DATA_01  0x4A
#define EXT_SENS_DATA_02  0x4B
#define EXT_SENS_DATA_03  0x4C
#define EXT_SENS_DATA_04  0x4D
#define EXT_SENS_DATA_05  0x4E
#define EXT_SENS_DATA_06  0x4F
#define EXT_SENS_DATA_07  0x50
#define EXT_SENS_DATA_08  0x51
#define EXT_SENS_DATA_09  0x52
```

```
#define EXT_SENS_DATA_10   0x53
#define EXT_SENS_DATA_11   0x54
#define EXT_SENS_DATA_12   0x55
#define EXT_SENS_DATA_13   0x56
#define EXT_SENS_DATA_14   0x57
#define EXT_SENS_DATA_15   0x58
#define EXT_SENS_DATA_16   0x59
#define EXT_SENS_DATA_17   0x5A
#define EXT_SENS_DATA_18   0x5B
#define EXT_SENS_DATA_19   0x5C
#define EXT_SENS_DATA_20   0x5D
#define EXT_SENS_DATA_21   0x5E
#define EXT_SENS_DATA_22   0x5F
#define EXT_SENS_DATA_23   0x60
#define MOT_DETECT_STATUS  0x61
#define I2C_SLV0_DO        0x63
#define I2C_SLV1_DO        0x64
#define I2C_SLV2_DO        0x65
#define I2C_SLV3_DO        0x66
#define I2C_MST_DELAY_CTRL 0x67
#define SIGNAL_PATH_RESET  0x68
#define MOT_DETECT_CTRL    0x69
#define USER_CTRL          0x6A
#define PWR_MGMT_1         0x6B
#define PWR_MGMT_2         0x6C
#define BANK_SEL           0x6D
#define MEM_START_ADDR     0x6E
```

```
#define MEM_R_W          0x6F
#define DMP_CFG_1        0x70
#define DMP_CFG_2        0x71
#define FIFO_COUNTH      0x72
#define FIFO_COUNTL      0x73
#define FIFO_R_W         0x74
#define WHO_AM_I         0x75
#ifdef  __cplusplus
extern "C" {
#endif /* __cplusplus */

   // TODO If C++ is being used, regular C code needs function names to have C
   // linkage so the functions can be used by the c code.

#ifdef  __cplusplus
}
#endif /* __cplusplus */

#endif /* XC_HEADER_TEMPLATE_H */
```

Appendix K - Display Header Code

```
#ifndef DISPLAY_H
#define      DISPLAY_H

#include "mcc_generated_files/i2c1_master.h"
#include "mcc_generated_files/examples/i2c1_master_example.h"

i2c1_address_t Display = 0x00;

#ifdef  __cplusplus
extern "C" {
#endif




#ifdef  __cplusplus
}
#endif

#endif /* DISPLAY_H */

#define one



void Display_Init(void);
void Display_Write1Digit(int Digit,int Number);
void Display_Digits(int Numbers);
```

```
void Display_Help(void);
void Display_Blank(void);
void Display_Off(void);
void Display_On(void);
```

Appendix L - Display Source Code

```
#include <builtins.h>
#include "mcc_generated_files/mcc.h"
#include "Display.h"

void Display_Init(){
    int digit;
    I2C1_Write1ByteRegister(Display, 0x0C,0x01);

  // Default global intensity is minimal. Set to 50 %.
  I2C1_Write1ByteRegister(Display, 0x0A, 0x0f);

  // Set scan limit to display all digits.
  I2C1_Write1ByteRegister(Display, 0x0B, 0x07);

  // Set all digits to "no decode".
  I2C1_Write1ByteRegister(Display, 0x09, 0x00);

  for(digit=0; digit<=7; digit++) {
    I2C1_Write1ByteRegister(Display, digit+1, 0x60);   // turn all segments ON
    //__delay_ms(1000);

  // Set all digits to font decode.
  I2C1_Write1ByteRegister(Display, 0x09, 0xFF);
  }
}
```

```
void Display_Write1Digit(int Digit,int Number){
     I2C1_Write1ByteRegister(Display, Digit, Number);
}
void Display_Digits(int Number){
   int Num4;
   int Num3;
   int Num2;
   int Num1;
   int Temporary;
   //print digit 4
   Num4 = Number/1000;
   Temporary = Number %1000;
   Display_Write1Digit(1,Num4);
   //print digit 3
   Num3 = Temporary/100;
   Temporary = Temporary %100;
   Display_Write1Digit(2,Num3);
   //print digit 2
      Num2 = Temporary/10;
   Temporary = Temporary %10;
   Display_Write1Digit(3,Num2);
   //print digit 1
      Num1 = Temporary/1;
   Display_Write1Digit(4,Num1);

}
```

```
void Display_Help()
{
   Display_Write1Digit(1,12);
   Display_Write1Digit(2,11);
   Display_Write1Digit(3,13);
   Display_Write1Digit(4,14);
}

void Display_Blank()
{
   Display_Write1Digit(1,10);
   Display_Write1Digit(2,10);
   Display_Write1Digit(3,10);
   Display_Write1Digit(4,10);
}

void Display_Off()
{
   I2C1_Write1ByteRegister(Display, 0x0C,0x80);
}

void Display_On()
{
   I2C1_Write1ByteRegister(Display, 0x0C,0x01);
}
```

Appendix M - MQTT topic table

	Team #	205	
	Team Name	Watching Distance	
	Topic	egr314/team205/PCB	egr314/team205/laptop
	Device Publishing	Board	Laptop
	Device Subscribing	Laptop	Board
	Value Separator	;	;
Value 1	**Name**	pitch	numberOut
	Description	y axis rotation	Data to print
	Units	degrees	8-bit
	Minimum Value	0	0
	Maximum Value	360	255
	Type	float	int
	Format String	f	i
	Unique Identifier	y	j
Value 2	**Name**	AnalogIn	n/a
	Description	Values from the IR sensor	
	Units	8-bit	
	Minimum Value	0	
	Maximum Value	255	
	Type	int	
	Format String	i	
	Unique Identifier	s	
Value 3	**Name**	numberOut	
	Description	Values to print	
	Units	m	
	Minimum Value	0	
	Maximum Value	255	
	Type	int	
	Format String	i	
	Unique Identifier	j	
	Combined Example	240	100

Appendix N - Thonny Python Coding

```
from mqtt_async import MQTTClient, config
import uasyncio as asyncio
import time
from machine import UART
import logging
logging.basicConfig(level=logging.DEBUG)

MAXTX = 4

# Change the following configs to suit your environment
TOPIC_PUB = 'EGR314/Team205/BOARD'
TOPIC_SUB = 'EGR314/Team205/LAPTOP'
config.server = 'egr314.ddns.net' # can also be a hostname
config.ssid     = 'photon'
config.wifi_pw  = 'particle'

uart = UART(2, 9600,tx=17,rx=16)
uart.init(9600, bits=8, parity=None, stop=1,flow=0) # init with given parameters

async def receiver():
    b = b''
    sreader = asyncio.StreamReader(uart)
    while True:
        res = await sreader.read(1)
        if res==b'\r':
            await client.publish(TOPIC_PUB, b, qos=1)
```

```
            print('published', b)
            b = b''
        else:
            b+=res

def callback(topic, msg, retained, qos):
    print('callback',topic, msg, retained, qos)
    while (not not msg):

        uart.write(msg[:MAXTX])
        time.sleep(.01)
        msg = msg[MAXTX:]

    uart.write('\r\n')
    time.sleep(.01)

async def conn_callback(client): await client.subscribe(TOPIC_SUB, 1)

async def main(client):
    await client.connect()
    asyncio.create_task(receiver())
    while True:
        await asyncio.sleep(1)

config.subs_cb = callback
config.connect_coro = conn_callback
```

```
client = MQTTClient(config)

loop = asyncio.get_event_loop()

loop.run_until_complete(main(client))
```