Michael Li and Ranyodh Mandur
Due Date:March 22, 2017
Course Code: ICE 4MI
Teacher: Mr Webb

Capture the Flag Bot Proposal

<u>Guide Line</u>

Description

- Bot Design / Materials
- Detection Systems
  - Line
  - Wall
  - Flag
- Strategies

Material List

- Electrical Components
- Additions

Timeline

Drawings

Description:

The capture the flag bot is a robot designed to navigate a ring, collect a very specific object as the “flag” in the middle, and bring it back to its starting position.

Bot Design:
The bot will have a total of 2 layers, both made of half inch plastic sheets. The bottom layer will be used as our base. It will just have a cubical base. We plan on making it 6*6 inches. This layer will hold our microcontroller circuit, motor circuit, line detection, batteries and wall detection. The motors will be towards the back and completely at the sides of the bot. The bot will also be supported by a third point of contact to the ground from the back side that will extend roughly be half an inch behind it. There will be a total of 6 holes on the bottom layer, they will be located on the top right and left ends. These holes will be used for 2 line detection systems to help us navigate the platform better. The other 2 holes will be used for support foundations to the second layer. The first layer will also consist of a small ledge that comes off at a 30 degree angle from the middle area to help hold the flag. As for the second layer, it will roughly be 4.5*4.5 inches. Assuming the flag is still a cylindrical shape, we will make a half oval cut at the front. Doing so will allow the bot to pull in the flag to the centre of the bot, and rest it on the first layer along with the little ledge. The second layer will hold the servo motor, lcd and serial board. All given measurements are just estimate values, they may vary in the end.

Detection Systems: There will be 3 different types of detection in the Capture the Flag bot. The first one will be line detection, to prevent the bot from falling out of the platform. A wall detection to determine if the bot is close enough to the flag to drive the servo motors. And finally, a serial communications detection for both identifying the flag and identifying which of the starting ends is ours.

1. Line: The line detection works based around a super bring light emitting diode facing straight to the floor and a phototransistor. The floor has a black color, therefore most of the energy/brightness of the light is absorbed. But once the light is on a colorful or white surface, light reflects off the surface into the phototransistor. If the phototransistor detects light, it dumps all the current to ground instead of the specific pin on the chip. The chip will then be programmed based on whether the pin is a high or a low. And this program will communicate with the motors to reverse or go forward. We are considering having 2 line detectors as the acute angle from coming out will be difficult to navigate with a single line sensor.

2. Wall: The wall detection works by using a sharp analog distance sensor. The sharp sensor uses infrared detection and can determine how far away the object is, provided it is within 10 cm to 80 cm. The sensor outputs a voltage based on the distance. However since this voltage output is not linear, by breaking the input into smaller pieces, we will use piecewise linear approximation to determine the distance. (equation: distance = $\frac{\frac{6787}{voltage-3}-4}{5}$) This value would then undergo if statements as to whether or not the bot should power the servo motor.

3. Serial: The serial communication we use is asynchronous communication, meaning it will not have support from an external clock and we must set proper protocols on both receiving and transmitting ends. Basically, everything we want to find(base and flag) will send a signal with an infrared emitter with an ASCII letter encoded in it. For example the flag will emit a signal with the ASCII code for the letter “F”. We will confirm these signals and tell our bot, this is the flag or base. The bot will then go there and or acquire the object.

So an ideal run of the capture the flag bot would be: Get to the middle of the ring using line detection and avoid falling out. Once we get to the middle, serial will tell us which of the many objects is the flag. The would then bot approaches the confirmed flag, and once the wall detection confirms the distance is close enough, the bot will power the servo motors and acquire the flag. Finally, we will once again use serial communications to find where our starting point was, return to the start point without falling out of the platform with the help of line detection. All this would be done in the fastest way possible. End of routine.

Strategies:

We plan on making is adding a second line sensor to help navigation, we plan on having them towards the 2 front edges of the bot. Other than that, many of our current “strategies” were already covered. Such as we will ensure that the bot is heavy enough to hold the flag and not tip, it will have it’s wheels towards the back a bit more and further apart. In addition, we will add another point of contact in some way other than just the 2 wheels to ensure balance. The flag should also be held within the centre of the bot, so on the second layer, we will make oval shaped cut, so the clamp can bring it in. After that, we want to have a strong hold on the flag, so we will create a small ledge on the first layer to ensure even if the clamp loses grip, we will still have some control over the flag. Finally, we will coordinate the battery pack location and where the flag will be held to evenly spread the weight around the bot.

Materials List:

Resistors	100Ω	220Ω	10KΩ	4.7KΩ	56KΩ	Total
Quantity	1	3	3	1	1	9
Pricing ($)	0.01	0.1	0.01	0.01	0.01	
Total Price ($)	0.01	0.03	0.01	0.01	0.01	0.09

Capacitor	0.1uF	1000uF	1uF	15pF	Total
Quantity	6	2	4	2	
Pricing ($)	0.07	0.29	0.05	0.08	
Total Price ($)	0.42	0.58	0.20	0.16	1.63

Detection Systems	Detector LED	PN 168	Sharp GP2D12 Sensor	Red LED	IR Receiver	Total
Quantity	1	1	1	2	1	
Pricing ($)	0.14	0.37	11.75	0.13	1.46	
Total Pricing ($)	0.14	0.37	11.75	0.26	1.46	13.98

Motor System	Chip L293D + socket 16	SW SW-SPST	Motors/Wheels/ Brackets	Servo Motor	Total
Quantity	1	1	2	1	
Pricing ($)	3.48	0.99	7.97	6.78	
Total Pricing ($)	3.48	0.99	15.94	6.78	27.19

Bateries	9V Battery	AA (1.5V) Battery	Battery Clips	Voltage Regulator	Total

Quantity	1	4	2	2	
Pricing ($)	3.2	0.49	0.63	0.55	
Total Pricing ($)	3.2	1.56	1.26	1.1	7.12

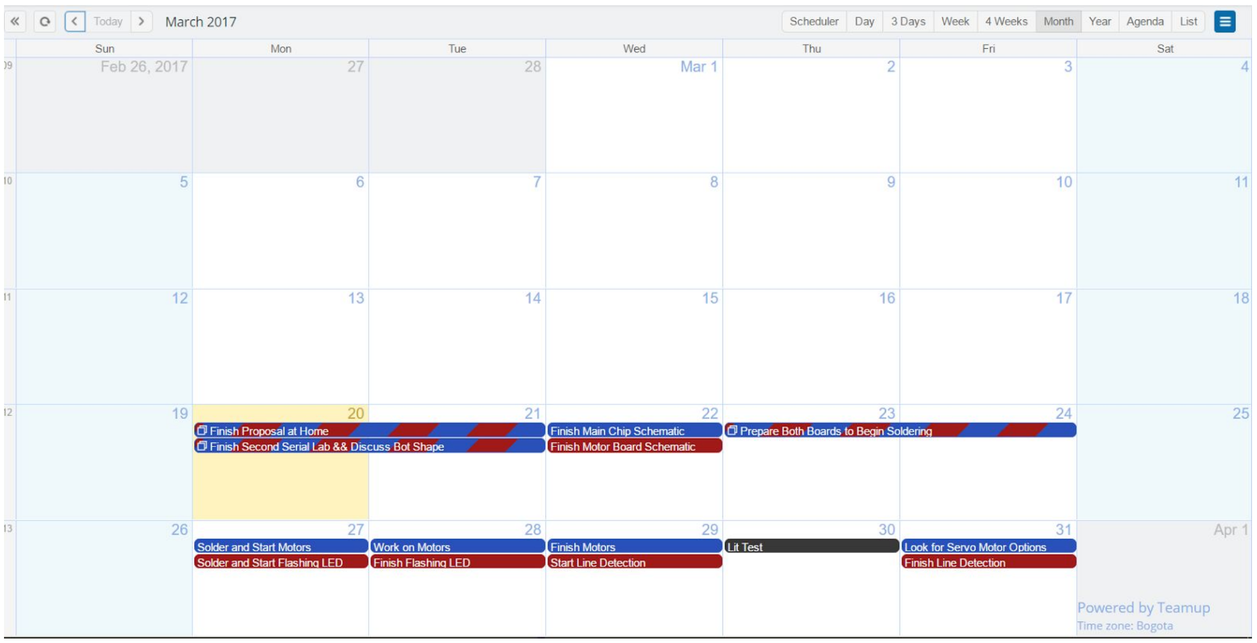
Main	PIC 16F877A + socket 40	DMC 16207 LCD	Potentiometer	Max 232 + socket 16	20 MHz Crystal	RS232 Adapter	74HC04N + socket 14	Total
Quantity	1	1	2	1	1	1	1	
Pricing ($)	8.93	9.83	2.39	1.52	0.22	3.15	1.22	
Total Pricing ($)	8.93	9.83	4.78	1.52	0.22	3.15	1.22	29.65

Additional Costs	Thin poly plastic material 6”*6”	Support Point	Total
Quantity	3	1	
Pricing ($)	4.99	0.50	
Total Pricing ($)	14.97	0.50	15.47

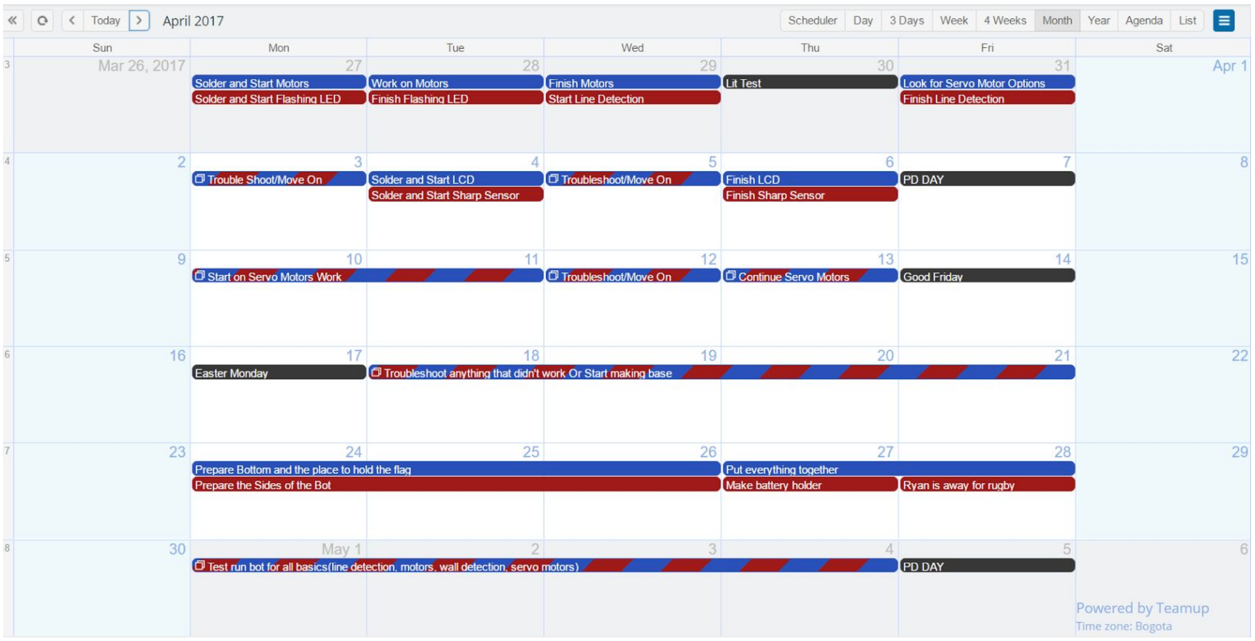
Total: $92.34

Calendar:

The following calendar will represent the schedule of the members of the capture the flag bot. Blue will represent Michael Li, and red will be Ryan Mendure. The mixed lines, blue and red represent both members collaborating or discussing a specific subject. The gray lines represent days where we will have trouble accessing the tech room due to different factors such as holidays or PD days. During those days, members of the group will try to finish what they can at home, ex finish writing the program on a .txt file and bringing it to school.



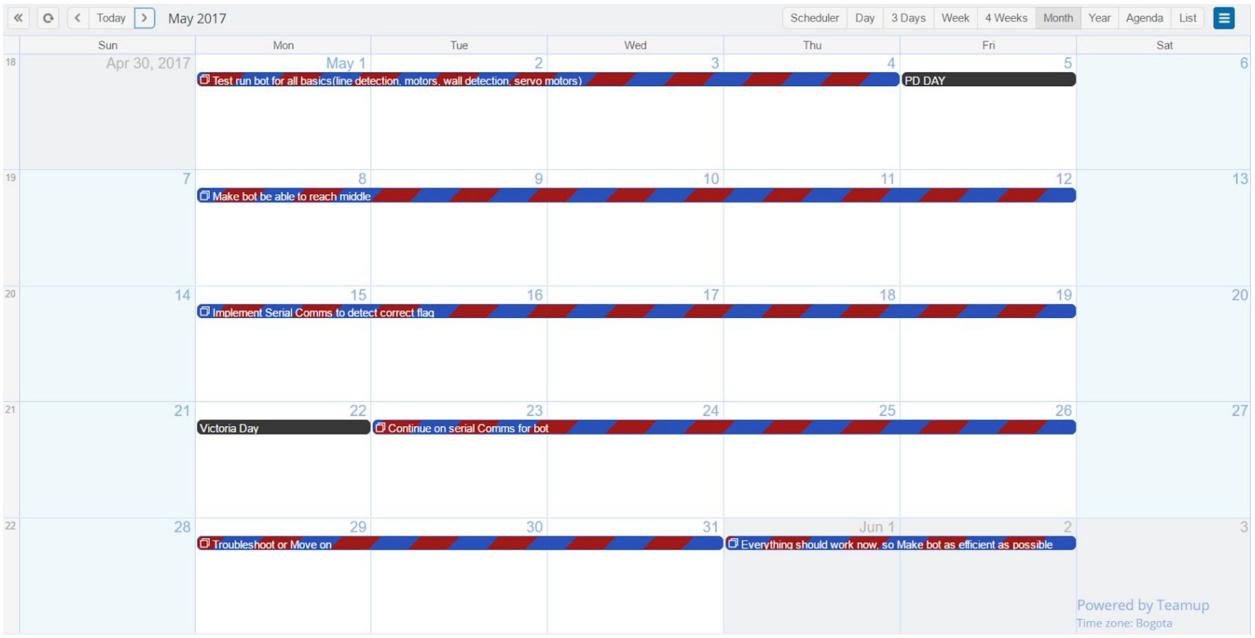
	Sun	Mon	Tue	Wed	Thu	Fri	Sat
09	Feb 26, 2017	27	28	Mar 1	2	3	4
10	5	6	7	8	9	10	11
11	12	13	14	15	16	17	18
12	19	20 Finish Proposal at Home Finish Second Serial Lab && Discuss Bot Shape	21	22 Finish Main Chip Schematic Finish Motor Board Schematic	23 Prepare Both Boards to Begin Soldering	24	25
13	26	27 Solder and Start Motors Solder and Start Flashing LED	28 Work on Motors Finish Flashing LED	29 Finish Motors Start Line Detection	30 Lit Test	31 Look for Servo Motor Options Finish Line Detection	Apr 1



« ⟳ ‹ Today › April 2017 — Scheduler | Day | 3 Days | Week | 4 Weeks | Month | Year | Agenda | List

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
3	Mar 26, 2017	27 Solder and Start Motors; Solder and Start Flashing LED	28 Work on Motors; Finish Flashing LED	29 Finish Motors; Start Line Detection	30 Lit Test	31 Look for Servo Motor Options; Finish Line Detection	Apr 1
4	2	3 Trouble Shoot/Move On	4 Solder and Start LCD; Solder and Start Sharp Sensor	5 Troubleshoot/Move On	6 Finish LCD; Finish Sharp Sensor	7 PD DAY	8
5	9	10 Start on Servo Motors Work	11	12 Troubleshoot/Move On	13 Continue Servo Motors	14 Good Friday	15
6	16	17 Easter Monday	18 Troubleshoot anything that didn't work Or Start making base	19	20	21	22
7	23	24 Prepare Bottom and the place to hold the flag; Prepare the Sides of the Bot	25	26	27 Put everything together; Make battery holder	28 Ryan is away for rugby	29
8	30	May 1 Test run bot for all basics(line detection, motors, wall detection, servo motors)	2	3	4	5 PD DAY	6

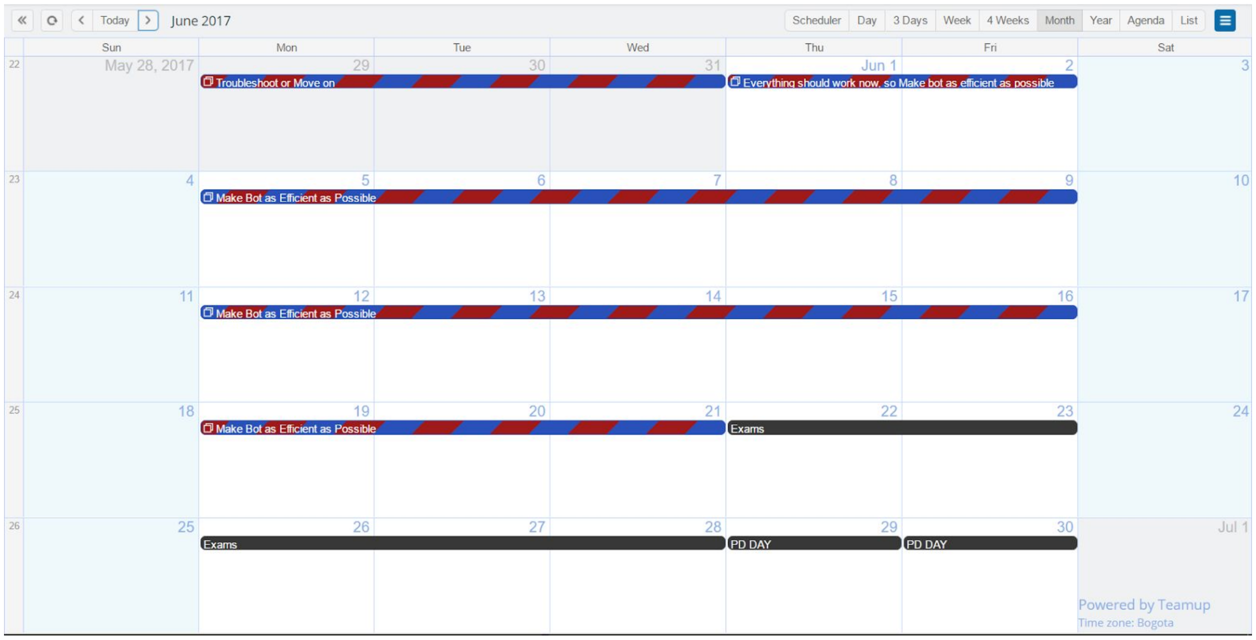
Powered by Teamup
Time zone: Bogota



« ⟳ ‹ Today › May 2017 — Scheduler | Day | 3 Days | Week | 4 Weeks | Month | Year | Agenda | List

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
18	Apr 30, 2017	May 1 Test run bot for all basics(line detection, motors, wall detection, servo motors)	2	3	4	5 PD DAY	6
19	7	8 Make bot be able to reach middle	9	10	11	12	13
20	14	15 Implement Serial Comms to detect correct flag	16	17	18	19	20
21	21	22 Victoria Day	23 Continue on serial Comms for bot	24	25	26	27
22	28	29 Troubleshoot or Move on	30	31	Jun 1 Everything should work now, so Make bot as efficient as possible	2	3

Powered by Teamup
Time zone: Bogota



Today June 2017

Scheduler Day 3 Days Week 4 Weeks Month Year Agenda List

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
22	May 28, 2017	29 Troubleshoot or Move on	30	31	Jun 1 Everything should work now, so Make bot as efficient as possible	2	3
23	4	5 Make Bot as Efficient as Possible	6	7	8	9	10
24	11	12 Make Bot as Efficient as Possible	13	14	15	16	17
25	18	19 Make Bot as Efficient as Possible	20	21	22 Exams	23	24
26	25	26 Exams	27	28	29 PD DAY	30 PD DAY	Jul 1

Powered by Teamup
Time zone: Bogota

15 June 2017

Capture The Flag Bot

Michael Li, Ryan Mandur

Robert Webb

Michael Li Reflection

To begin I'd like to say I don't think I've learned so much about something so much in only 4 months. To begin, everything actually seemed extremely overwhelming at the start of the course and even though we had a duotang it was very hard for me at least to understand it. However now I feel very comfortable with the information on it and I really wish I got to understand a lot of the specific detailed components on it earlier, such as MCLR on pin 1 acts like a reset switch, as well as RX and TX functions of each chip, and many other small pin functions I didn't even know existed. Secondly, understanding the analog and and digital signals, as well as how they communicate with the chip and the presets within the chip we have to make to ensure it is receiving the correct messages. Understanding the difference between the two and the corresponding components that use which types of communication. Thirdly, the major difference between capture the flag and fire fighter was the serial. I do not know if I could have figured out serial without the help of Gaurav and his labs. The whole idea of a message embedded in an infrared signal and then decoding it with a microcontroller was extremely overwhelming, but as we did the labs and started to understand what pulse, baud rates and parities are it slowly came together. Even though we never managed to get the serial working outside of our serial labs, it was very educational and I do feel like it was worth it. Alongside with serial, we also had to learn about controlling servo motors, the whole idea that the way we control it is by controlling the pulse width and manually creating a signal, Ryan did most of the research for this and I just implemented it in the program. Even though this was easier than the serial component, I think learning both of these make attempting the capture the flag so much more meaningful. In addition, I'd like to mention the lcd. Like everything else, this seemed fairly overwhelming at first as well, all the statements required for it to work and what they all mean. But as we kept trying and figured things out it really made sense, and being able to display something from one of the inputs detected from the chip itself was really rewarding. Before concluding, I'd also like to mention I am so much more comfortable with using the multimeter. I didn't even recognize some of the functions before and now I feel very comfortable testing voltages, resistance, capacitance,frequency and connections. To conclude, everything just clicked, even though there were some things that were identical to the sumobot we did last year, I feel like now everything just makes so much more sense and I'm not even sure how to describe it. Last year it just felt like I was following instructions and doing what I should be, but now I really feel like I understand what every component I added does and what the purpose of it is. Through some trouble with the line detection I understand what the 5.6K resistor does now and how increasing that actually increases the sensitivity of the bot, and all the capacitors we add to smooth out the current, all the grounds we tie and all the power we send in is because I understand why and how it works now, and It's not just those things, everything from the very first day when we got stuck on bootloader was a learning experience. The whole project was probably one of the most meaningful things I've done and I am actually fairly happy with the results even though we did not get a fully working capture the flag bot.

We also had several issues along the way, the most disturbing one was probably the breadboarding the bootloader. We spent about 2 weeks on that and to this day we're still not 100% sure why it didn't work but apparently it was our chip and breadboard that was goofed up. So after switching chips and a breadboard everything worked fine. Other than that I was also stuck on line detection for a little bit. Somehow our voltages were actually increasing from 5 to 5.3 to the motors and they were having seizures. I didn't know why because everything worked fine separately but just when I added line it would mess up. So I thought it was because the current wasn't smooth enough so I added capacitors but that just created more issues. Also, since I thought I just copied and pasted the code from individual components that worked fine, I just automatically assumed it was a hardware issue and never bothered to look back on the program. In the end, it was a program issue, I might overwrote the tris statement for the c ports and after we fixed that 0 to a 1, everything was solved. There were the small issues we ran into serial such as mismatched baud rates or incorrectly tuned infrared frequencies but those were fairly simple fixes compared to bootloader. In the end, there was much more that didn't work ideally. The serial never worked on our bot and we don't really know why, and since we only had 2 batteries, 1 to motors and the other to literally everything else, whenever we powered the servo, the voltage to every component would drop from 5 to 3. This wouldn't be a problem early in the program but as it got further and more variables were being kept track of, the lcd and wall detection would get messed, such as the wall detection would only be able to reach a max of 28 cm and the lcd would display random values. We never had time to make another power board and we didn't know how to deal with this problem through software, so it was also left untended.

I would do so much differently if i got to do this again. Multiple sensors would be one of the major ones. Having multiple lines and not just a single line sensor to detect the island in the middle. The way we have it is we can get in the ring no issue, but once we start to try and navigate outside of the line it becomes an issue. Otherwise, I would like to say also figure out serial earlier, but we couldn't have. If we really wanted to do things in the order the marks go as in 90% is grabbing an object without serial, we had to wait until we could get the wall and servo and to work until we work on serial. But at that point it was a bit late for some things such as when we realized we need another battery or power board, we could have hustled more throughout the course such as coming in during lunch more to work. Also plan things out much much better. When I was designing the boards, I constantly forgot about power or a connection and everything became really sketchy. We had to make a power board only to realize we forgot power for some other things and made a second power board, we made sketchy line cuts and solder shorts to fix connections, so better planning would also be on top of my list.

I understand this is a fairly new project and we are kind of navigating through it at sort of a testing phase, but there were a lot of things that weren't settled from the beginning. Such as an actual flag, or how many objects are in the middle or how home base was gonna work. So many things changed in the last 2 weeks and many groups started to panic and realize they are actually incapable of finishing because they did not realize some challenges of the project even existed.

But that is also on us tho, we didn't communicate clearly and misunderstood certain aspects to the routine.

For any upcoming grade 12s, I would say just plan like never before. Don't edit things on the run, things don't have to work exactly as your design doc, but have a very strong idea as to how things are working before the last month and a half to avoid last minute realizations such as you need another board, but not enough time to make it. As well as all of the things I mentioned I would fix if I got to redo this project. Also, even though I am devastated I didn't actually finish my bot, I am still so happy I chose to do the capture the flag. The firefighter is definitely a challenge and I have no idea how I'd even approach it or get to room 4, but the experience of playing with serial in capture the flag and exploring different components is very rewarding itself.

Ryan Reflection

When I first began this class back in January, the only impression of the course I had was from my friends Gaurav Alex and Jacob Dinka. I thought going into this course that I was one hundred percent sure I was not going to do the capture the flag robot because I saw the stress it caused those guys and it scared me. I still remember going into the tech room at lunch during first semester and looking at Gaurav's bot and thinking : "wow I will never be able to build something like that, it's so complicated, this tech course is on another level". But I still had high hopes for my bot because I had my main man michael working with me. When I first looked through the duotang I felt a huge wave of anxiety wash over me, when I read the schematic for the bootloader this course had become real and my nightmares from Gaurav's bot had become real. Despite my initial fear and lack of understanding, Michael and I were able to persevere to create a working bot that can make it to the middle of the ring and capture the flag and kind of bring it back home.Although the bot isn't fully functional i am still very proud of it. I learned so much in such a short time it's incredible. But this did not come easy.

There were many obstacles that Michael and I had to hurdle over during these past four months. Most infamous of these was breadboarding the bootloader. This was probably the most frustrating and heartbreaking part of the entire course (especially because it was the very first circuit for our bot). Michael and I worked tirelessly for two weeks coming in and lunch and staying after school trying to figure out what was wrong with our bootloader. However, at the end of those two weeks we overcame the problems and came out with a working bootloader and we learned a lot about troubleshooting and especially learned how to use a multimeter and what all its various functions do. Those weren't the last of our struggles though, the next big obstacles came when we tried to combine all the different labs and circuits we made into a single bot controlled by our microcontroller. Our motors stopped working, we had to make two separate power boards just to accommodate all the various motors and such that our bot could actually function without constantly shutting down or just not moving at all. On top of that, we had to figure out a way to connect the TSOP infrared receiver to our microcontroller so that the signal produced by it wouldn't interfere with bootloading because they are both connected to the RX pin of our chip. We ended up making it so we could cut power to the TSOP so that when we needed to load a new program, all that was needed to be done was to flip a switch on or off. Finally, the last hurdle that we sadly weren't able to overcome was serial communication. Although we understood how serial communication worked and how it should look on our bot, implementing it was another story. There were many problems we had, such as other components interfering with the communication, not enough power, and not getting the proper message.

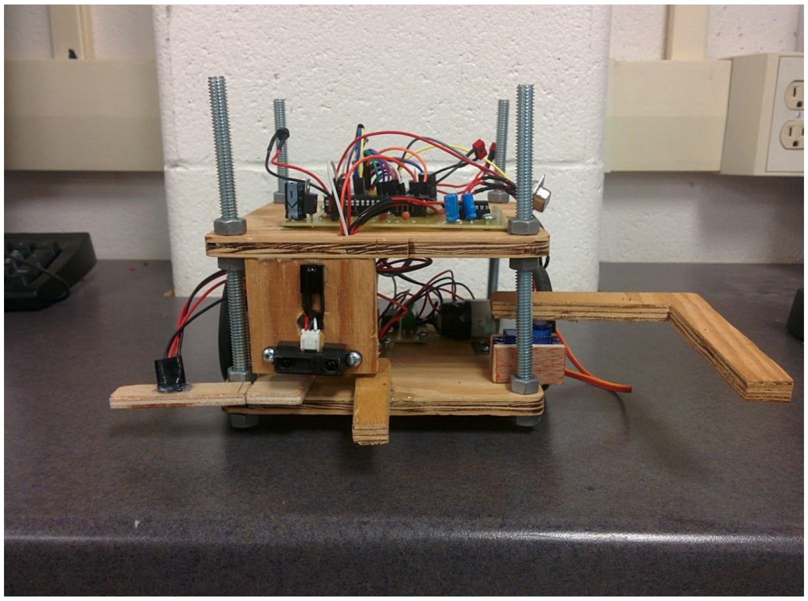
Many of our issues Michael and I worked through together(except for the last two weeks sorry Michael) as a team, but there were times where I just left Michael out to dry. Of these issues the biggest were the LCD, wall sensor, and parts of the line follower program. Even now I am still fuzzy on these aspects of the project.

Now onto what i'd improve on this course for next year. The biggest thing i'd change about this course is the emphasis on serial communication, touching more on how servos work, how to build a working claw, telling the students that simultaneously bootloading and receiving on the TSOP is not possible, power supply management, modularity, and of course having a working flag to capture. The labs we did for serial helped a lot but you should add in a portion where you actually implement serial using the TSOP and the PIC16f18XXX to display a string on the LCD. Servos were really confusing at first but I was able to grasp how they work pretty well, but this wasn't the case for most of the class, I would do a mini lesson on how to program servos and how to manipulate them using different signals. The claw is weird and for me at least hard to imagine how it works, doing a mini lesson on that would also be helpful. Probably one of the bigger issues is power management. Because there is three different motors running simultaneously and an LCD constantly outputting data, a plain 9V battery won't cut it for the power requirements of the system it would be extremely helpful if you did a lesson on good power supply management and how to calculate the amount of current needed for the circuits students may want to make. Also please just have a working flag for the next year class. Also have well defined marks for different things the capture the flag robot can do.(i say this because it was pretty vague during the semester and people were still generally confused at the end of the year on what an 80,85 and 95 bot were supposed to do)

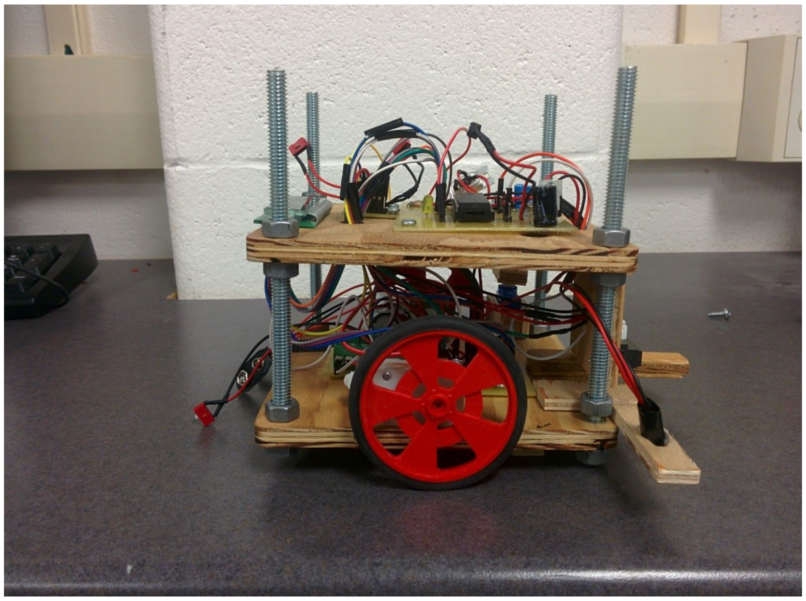
Although the bot isn't fully functional I am still extremely proud of what Michael and I were able to accomplish together and the amount I learned about electronics and how different components interact with each other. I loved this course and the independence and power it gives to the student to just work on what they want without a teacher hovering over them. Because this course is so unique and offers an experience you cannot find anywhere else in the school I strongly recommend younger students at our school to take it because it teaches them a lot.This course is the most stressful and rewarding course I have taken in all my years at SJAM

Final Product Photos

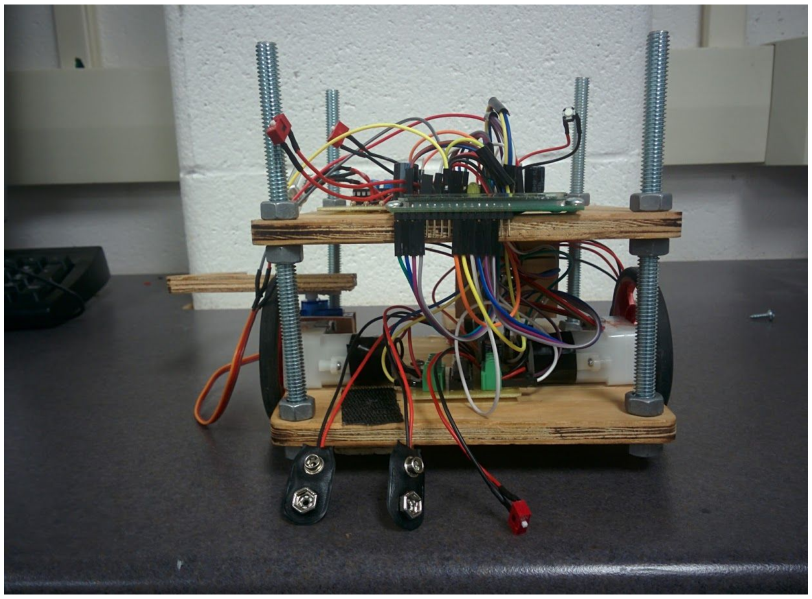
Front View



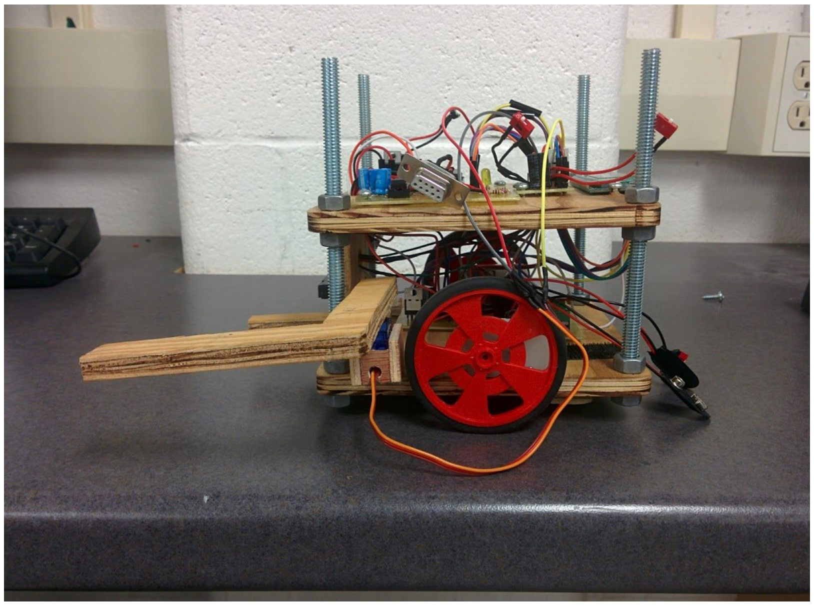
Right View



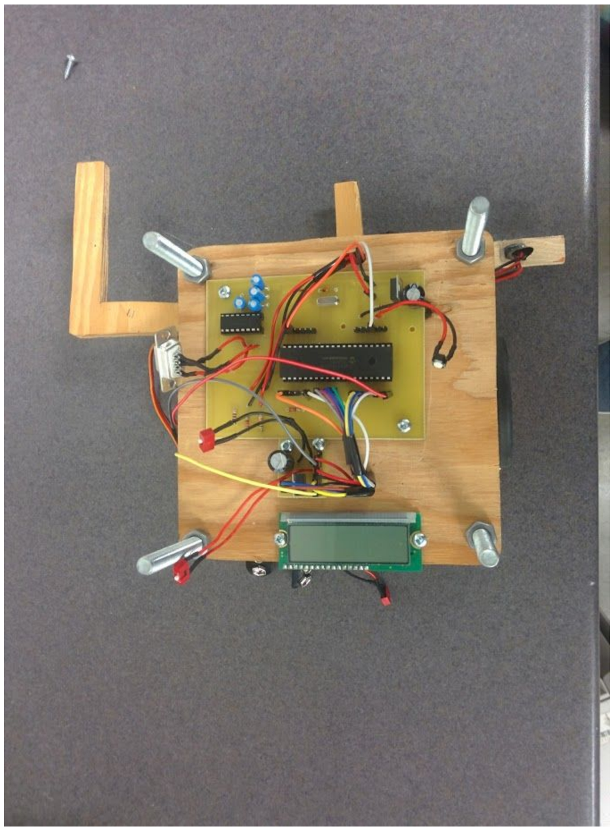
Back View



Left View



Top View



Program

```
‘required import for serial
INCLUDE "modedefs.bas"

DEFINE Loader_Used 1
DEFINE OSC 20
DEFINE LCD_DREG          PORTD
DEFINE LCD_DBIT  4
DEFINE LCD_RSREG PORTD
DEFINE LCD_RSBIT 0
DEFINE LCD_EREG PORTD
DEFINE LCD_EBIT  2
DEFINE LCD_BITS  4
DEFINE LCD_LINES         2
DEFINE LCD_COMMANDUS       1500
DEFINE LCD_DATAUS      50

DEFINE ADC_BITS     8
DEFINE ADC_CLOCK    3
DEFINE ADC_SAMPLEUS 50


TRISA = %11111111
TRISB = %00000000
TRISC = %00000001

LED          var PORTB.0
trans        var PORTC.0
rx           var PORTC.7

‘Variables
pastHome     var byte
wallD        var byte
r            var byte
i            var byte
counter      var byte
VAL          var word
```

```
counter = 0
pastHome = 0

MAIN:
ADCON1 = 0
ADCIN 2, R

‘stores the value received from serial into val
serin rx,N2400,1000, ERROR, ["peaches"], val
lcdout  $FE,1,#val

‘Update wall detection
wallD = 0
wallD = ((6787/(r-3))-4)/5

‘Bug with wall, randomly jumps to 14
if wallD = 14 then
   wallD = 55
endif

LCDOUT $FE,$C0 + 14, DEC wallD

'get to island
if pastHome = 0 then
   ‘go forwards until finds white line, then go to phase 2, where we follow the line

   LCDOUT $FE, $C0, "Phase 1"
   'forwards
   PORTB = %00100100

   If trans = 0 then
      pastHome = 1                ‘allows phase 2

      'pause
      PORTB = %00000001
      pause 100

      'reverse
      PORTB = %00010011
```

```
        pause 350

        'left
        PORTB = %00100001
        pause 650

        'forward
        PORTB = %00100101
        pause 200

    endif
endif

'line follow to centre
if pastHome = 1 then
    LCDOUT $FE, $C0, "Phase 2"
    if trans = 0 then
        gosub foundLine
    else
        gosub normal
    endif

    ‘While line following, if there is something less than 13cm in front of bot, go to capture phase
    if wallD < 13 then
        pastHome = 2     ‘allows phase 3
    endif
endif

'capture
if pastHome = 2 then
    ‘Powers servo and captures flag, executed once, then go to find home phase
    high LED
    ‘Power servo
    PORTB = %10000000
    LCDOUT $FE, $80, "Capturing"
    LCDOUT $FE, $C0, "Phase 3"
    pause 150
    PORTB = %00100100        ‘forward
    pause 250
    PORTB = %00000100        ‘right
    pause 400
    PORTB = %00100000        ‘left
    pause 150

    PORTB = %00100100        ‘forward
    pause 325
    PORTB = %00000000        ‘stop

    for i = 0 to 50                ‘send message to close claw
        high LED
        pulsout PORTB.7, 500
        pause 19
    next i

    pastHome = 3         ‘allows phase 4
endif



'find island again
if pastHome = 3 then
    ‘Spins until finds something, then goes to it
    LCDOUT $FE, $C0, "Phase 4"
    PORTB = %00000100        ‘keep turning right

    If wallD> 50 then
        pastHome = 4     ‘allows phase 5
    endif

endif

'navigate back to island
if pastHome = 4 then
    ‘Just goes straight until it finds the island, then line follows again
    LCDOUT $FE, $C0, "Phase 5"
    'forwards
    PORTB = %00100100

    If trans = 0 then
```

```
        pastHome = 5      ‘allows phase 6
    endif
endif

'line follow
if pastHome = 5 then
    ‘ line follows until finds home base, then turns into home
    LCDOUT $FE, $C0, "Phase 6"
    LCDOUT $FE, $C0+10, dec counter
    pulsout PORTB.7, 500
    if trans = 0 then
        gosub foundLine
    else
        gosub normal
    endif

    if counter > 3 then
        if wallD > 15 then
            if wallD < 20 then
                pastHome = 6     ‘allows phase 7
            endif
        endif
    endif


endif

'found home
if pastHome= 6 then
     ‘Enters home and then stops
    LCDOUT $FE, $80, "                "
    LCDOUT $FE, $C0, "Phase 7"

    PORTB = %00100100       ‘forward
    pause 650
    PORTB = %00100000       ‘left
    pause 700
    PORTB = %00100100       ‘forward
    pause 200
    high led
    pause 500
    pastHome = 7          ‘allows phase 8
endif

‘drop flag
if pastHome = 7 then
    ‘End of program, flag is delivered and bot stops
    PORTB = %00000000        ‘stop motors completely
    high led
    pause 500
    low led
    pause 500
endif
GOTO MAIN
‘End of Main


‘Line following subroutines

‘When it’s not on white
normal:
    low led
    'right
    PORTB = %00000100


    LCDOUT $FE, $80, "Turning"
    RETURN

‘When it’s on a line
foundLine:
    'left
    PORTB = %00100001

    LCDOUT $FE, $80, "On Line"
    pause 200

    'forwards
```

```
   PORTB = %00100101
   pause 50

   counter = counter + 1

   return

ERROR:
   lcdout $FE,1,"ERROR"
   pause 1000
```

Michael Li: ____________________

Ryan Mandur: ________________