ECE 4950 - Project 4 (12/10/18)

Customer Requirements and Final Design Parameters

Team 11: Logan Lewis, Anthony Sergent, Harrison Hart,
Nafis Reza, and Zach Swalhah

Executive Summary

The final project of ECE 4950 (Integrated Design I) required understanding of the procedures and topics reviewed over the first three projects in the course. This included a concise grasp on the use of the Quanser Q4 board and its ability to control the majority of the hardware components used in the lab. This, along with a strong understanding of MATLAB, Simulink, and the ability to communicate between digital and physical system components were all concepts that helped to produce the system used in the final project. The group applied the skills learned over the course of this class to build a physical system that utilized stepper motors, belts, a servo motor, and a DC motor. This system had the ability to move left, right, up, and down across the frame on which it was stationed. The purpose of this system was to simulate chess game scenarios, and the physical system had the ability to move to a certain location above a square on the board, lift a piece, and move it to a new location. There was code written in MATLAB that covered the chess game logic, the movement instructions, and the image processing. Also, a GUI was created that provided a straightforward way to use the system in its entirety by calling individual MATLAB functions, running their scripts, and communicating with the simulation file that controlled the movement of the physical system components. There were 4 specific tasks that the final project was meant to be able to carry out, so the design was built and tested to carry out these tasks. Over the course of working on the final project, the group learned many valuable concepts that can be carried on into ECE 4960, as well as future projects. These concepts included: image processing, communication processes between physical components and digital instructions, and implementation of various electrical circuits and motors.

Engineering Requirements

- Camera along with image processing set up to detect the different pieces in relation to one another and the sides of the board
- Change motor input to move the system to a desired location/ index above the board
- Based on the piece being moved, rules of movement for each chess piece is applied and a validity check is performed
- Find locations of both kings on every turn to determine if either king is in a checked position, or can potentially be checked, respectively
- The system analyzes the pieces and weights the possible movements to determine which move will give the "best" result based on the rules of chess
- The system should position in such a way that the simulated game clock (load cell) is pressed after a move is completed
- A GUI that is self-contained is created to provide options for each of the possible user input options to carry out the desired final project tasks
- The motor operation is efficient and the electronics used do not create excess noise
- The timing of the movement should be quick and efficient, ensuring the moves are completed correctly, while also taking into consideration the importance of speed
- The physical hardware components quickly respond to a user input, based on efficiently written code, and reduction in integrated system lag.
- The design does not have an excess of exposed wires or other potentially dangerous components in the open. Also, an E-stop was incorporated into the final system design
- The set-up is designed such that anyone could easily interact with and run the system to the full extent of its functionality
- Smooth/ closed-loop controlled motion was used with the inclusion of the DC motor on the z-axis, as the angle sent to the DC motor was variable and could be adjusted by the user at any time during operation of the simulation file

Design Details

Our design was inspired by 3D printer designs, with a rectangular frame, a crossbar in the middle of the frame that is able to move up and down along the Y-axis, and a subassembly that grabs the pieces able to move left and right on the crossbar. The functionality came from two stepper motors, the DC motor required, and a servo motor. The two stepper motors were NEMA 17 and worked simultaneously to control the X-axis and Y-axis movements using A4988 driver chips to control them. The connections made to the driver are displayed in figure 1 below.

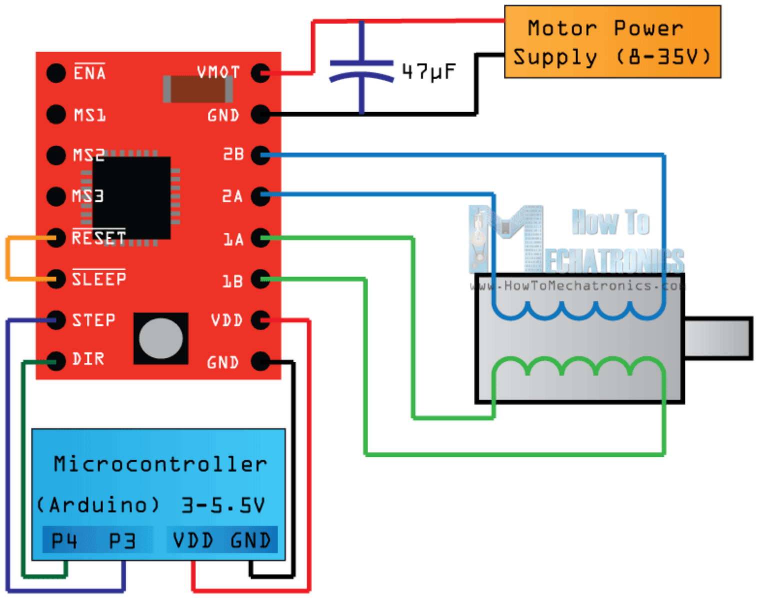

Motor Power Supply (8-35V)
47µF
ENA
MS1
MS2
MS3
RESET
SLEEP
STEP
DIR
VMOT
GND
2B
2A
1A
1B
VDD
GND
How To Mechatronics
www.HowToMechatronics.com
Microcontroller (Arduino) 3-5.5V
P4
P3
VDD
GND

Figure 1 (Stepper motor driver connection diagram)

For the microcontroller in figure 1, our quanser board was used instead and the digital outputs were used to send a constant value for the direction input and a square wave for the step input of the A4988 driver. The Nema 17 stepper motors were positioned in the top left and right corners of the rectangular frame and moved our grabbing subsystem with a belt in an H formation. When the motors rotate simultaneously in opposite directions, they are able to control the Y-axis motion. When the motors rotate in the same direction, they control the X-axis motion. This movement is demonstrated in Figures 2 and 3 below.

(images from: https://www.jjrobots.com/air-hockey-robot-evo-how/)

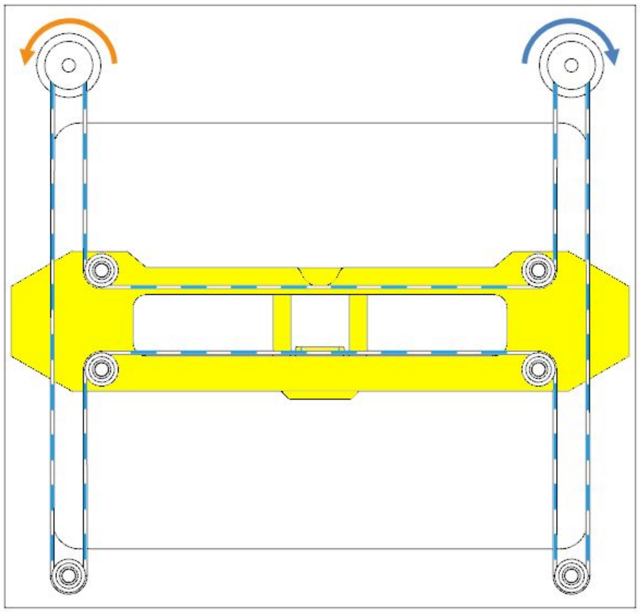


Figure 2 (Y - axis motion)

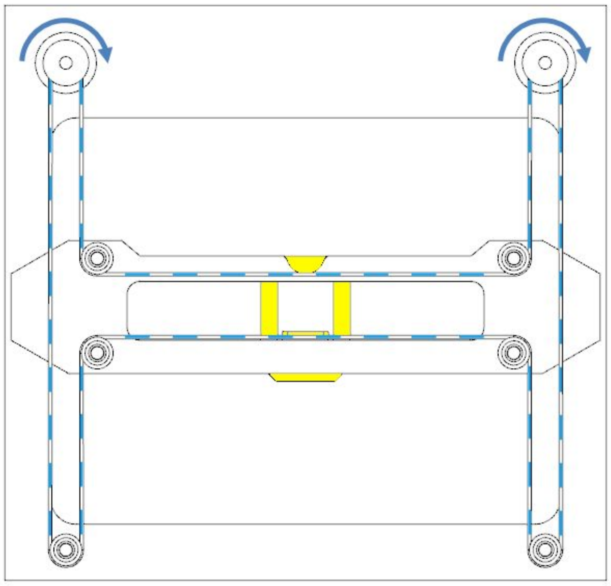


Figure 3 (X - axis motion)

As indicated by Figure 2, the crossbar moves down along the Y-axis when the motors rotate toward each other and up when the motors rotate away from each other. On the X-axis, the subassembly containing our Z-axis movement moves to the right when the stepper motors rotate counter-clockwise, and left when they move clockwise. Our Z-axis movement that allowed the claw to move up and down to pick up the pieces was powered by the Tohoku DC motor provided to us. We used a lead screw mechanism with the DC motor. Lead screws are long threaded rods used to convert rotational movement to linear movement. The lead screw was attached to the DC motor's shaft with a coupler. When the DC motor turns clockwise, the lead screw would turn clockwise resulting in the attached claw moving upward, and vice versa for counter clockwise. The DC motor sits on a 3D printed platform, as seen in figure 4, with its shaft pointed downward, attached via coupler to the lead screw. Parallel to the lead screw is a support rod to ensure that the claw does not rotate with the lead screw. The support rod is straight so that the claw bracket can smoothly move up and down along it. The bracket holding the DC motor is attached to bearings that sit on the parallel crossbeams similar to the Y direction support beams. A full model can be seen below in figures 5-7 of our systems design. This model does not include the GT2 belt, DC motor, or servo claw.

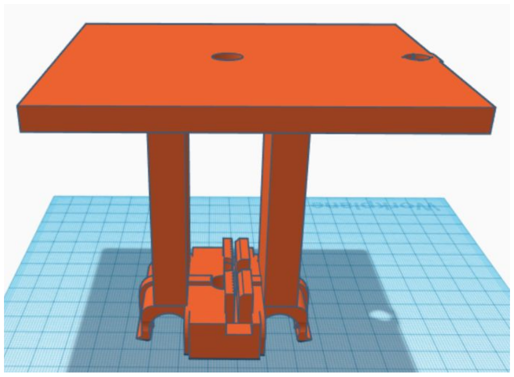


Figure 4 (DC motor bracket)

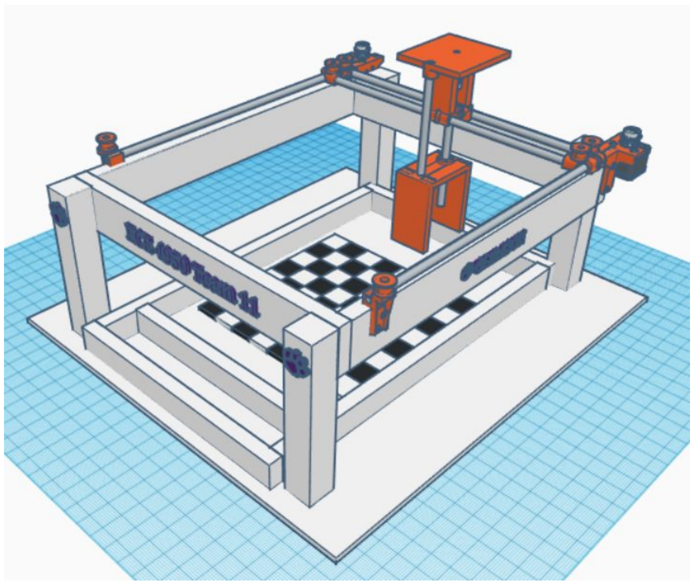


Figure 5 (System model 1)

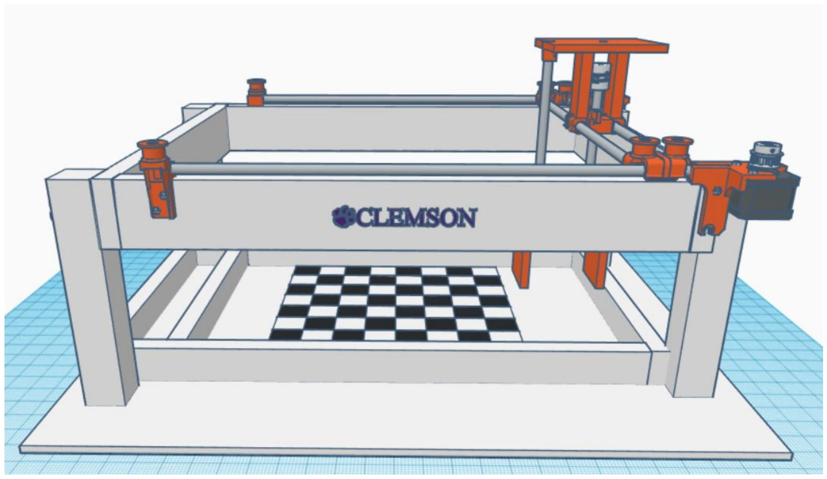


Figure 6 (System model 2)

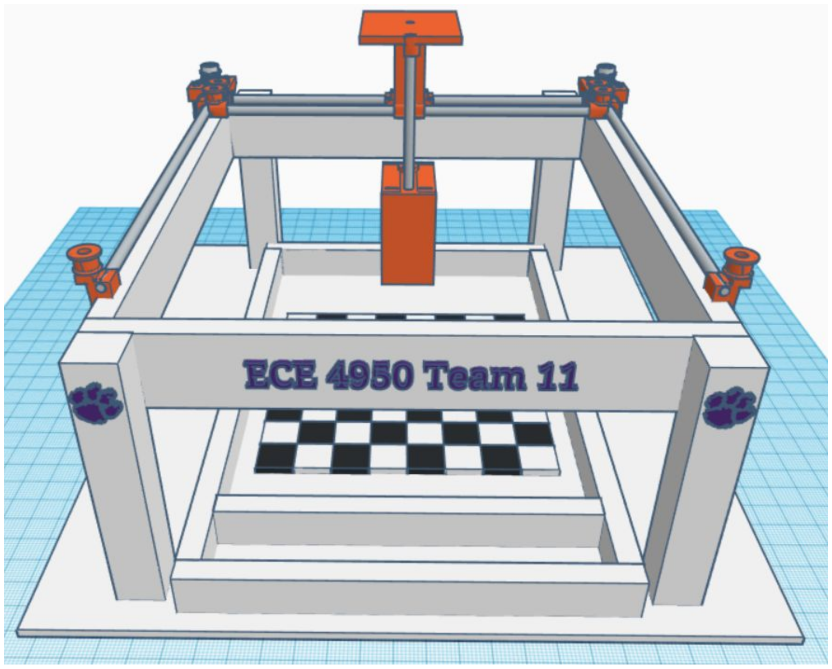


Figure 7 (System model 3)

In addition to the frame with the belts and all of the motors, a separate wooden support was built and positioned beside the main frame. This support housed the webcam used for the image processing portion of the project. The camera was positioned in such a way that the entire board was visible in the camera's view. Also, there was enough room above the board but within the frame so that the claw unit could be positioned out of the view field of the camera. Adhesive LED lights were attached to the underside of the camera support to be used in conjunction with the camera to effectively eliminate shadows from the pieces on the board. These lights were attached to a power block that was plugged into the AC power outlets to keep the LEDs on while the system was in operation.

The system operated through use of communication between the physical hardware and the software code that was written specifically to carry out the tasks presented in this final project. This code was broken into three major categories, each dealing with a separate concept that all came together to run the system as a whole. To have the entirety of the hardware working simultaneously, our group had to first ensure that the individual components were running as expected and could be controlled via code segments.

So, the first category of the coding involved the software that communicated directly to the simulation file to cause the motors, servos, and other hardware to respond in a variety of ways. Using a simulink file, four sections were created that each held component blocks for all of the responsive hardware components. These included: the DC Tohoku motor (used for z-axis movement), both stepper motors (used for x and y-axis movement), a servo (opening and closing the claw), and an E-stop switch. This simulation file held all of the initial values for each component's default settings and was compiled and ran prior to attempting any of the tasks for the final project system as a whole. The DC motor simulation was controlled via the same simulation model used in project 3, where only the degree input as a constant was changed during the z-axis movement. The stepper motors utilized digital outputs from the Q4 board and had pulse waveforms and constant inputs. The amplitude and the period of the pulses could be manipulated to start and stop the stepper motors and change their speed. The constant block was either a 1 or a 0, indicating either clockwise or counterclockwise rotation, respectively. Also, the Q4 board counter was used to operate the servo claw, with two constant inputs. One input was switched between a high and low value to open and close the claw. The E-stop part of the simulation file used an analog input from the Q4 board with 5 volts passing through it, if the E-stop was pressed, a MATLAB function output a 1, which triggered a Stop block that ended the simulation. Each of these component values were changed in a MATLAB function called "move_piece.m", which set each parameter in the simulation file so that the claw unit would position over a board index, lower to the piece, close the claw, raise back up, position over a new board index, and finally drop and release the piece.

Another important code category was the game logic. The game logic was broken into valid movement functions for each chess piece type and each of the four tasks that the system needed to carry out for the final project. The valid movement functions took current piece locations, checked their piece type, and determined whether a requested move was valid or not. The four tasks, moving a single piece, eliminating an enemy piece, checking the opponent king, and avoiding check each used these valid movement functions and called "move_piece.m" to communicate with the physical system components. Also, the image processing code provided the gamestate structure, which gave the locations, colors, and shapes of each piece present on the board.

Finally, The GUI coding was completed so that the user could easily operate the entire system by simply clicking buttons and typing in piece types and alignments to be associated with the colored shapes on the board. The GUI called the corresponding MATLAB functions that went along with the task that was being run, including taking the picture of the board to run the image processing. Also, a status window displayed status messages for each action that the GUI could perform to provide feedback to the user.

Analysis of Final Prototype Performance

Customer Requirement	How well our system met expectations
Determining type/ location of chess pieces on the game board	Consistently identified the index locations pieces. The image processing accurately displayed the shapes and colors of the pieces, with only a few instances of error.
Move a specific piece based on user input	The system moved pieces well, as lifting, transferring, and placing them into their new locations based on the index entered by the user, or via the game logic was accurate.
Determine the validity of requested moves	The game logic code correctly displayed move validity for each of the cases tested as a status message on the GUI.
Check the opposing king/ move user king away from a checked position	Our system was able to put the opponent king in check for each test case performed. The fourth task, making a move to eliminate the check on the user's king, did not work for every case, but for a few simple scenarios the correct moves were executed.
Determine the "best" move	This requirement was well fulfilled as the system could potentially avoid user check for some cases, put the opponent king in check, and eliminate pieces as necessary.
Hit button to stop the chess game clock after every move	The movement of the system could fulfill this requirement.
GUI / User Friendly	A GUI was created that allowed the user to control the entire system by typing and clicking buttons. Also, information was accessible that described how to use the GUI.
Quiet	The stepper motors created vibration along with the wooden frame that caused the system to bit a bit noisy, but not overly loud.
Efficient	Our system consistently completed the moves that it was meant to carry out, without performing unnecessary actions.
Fast	The movement was not very fast, this sacrifice was made so that the system would be more accurate.
Safe	The system did not have dangerous components, the wires were well laid out, and there was an E-stop.
Smooth/ closed-loop controlled motion	This was achieved with the DC motor for the z-axis movement, as the user could type in a variable height for the claw to raise at any time during the operation of the simulation.

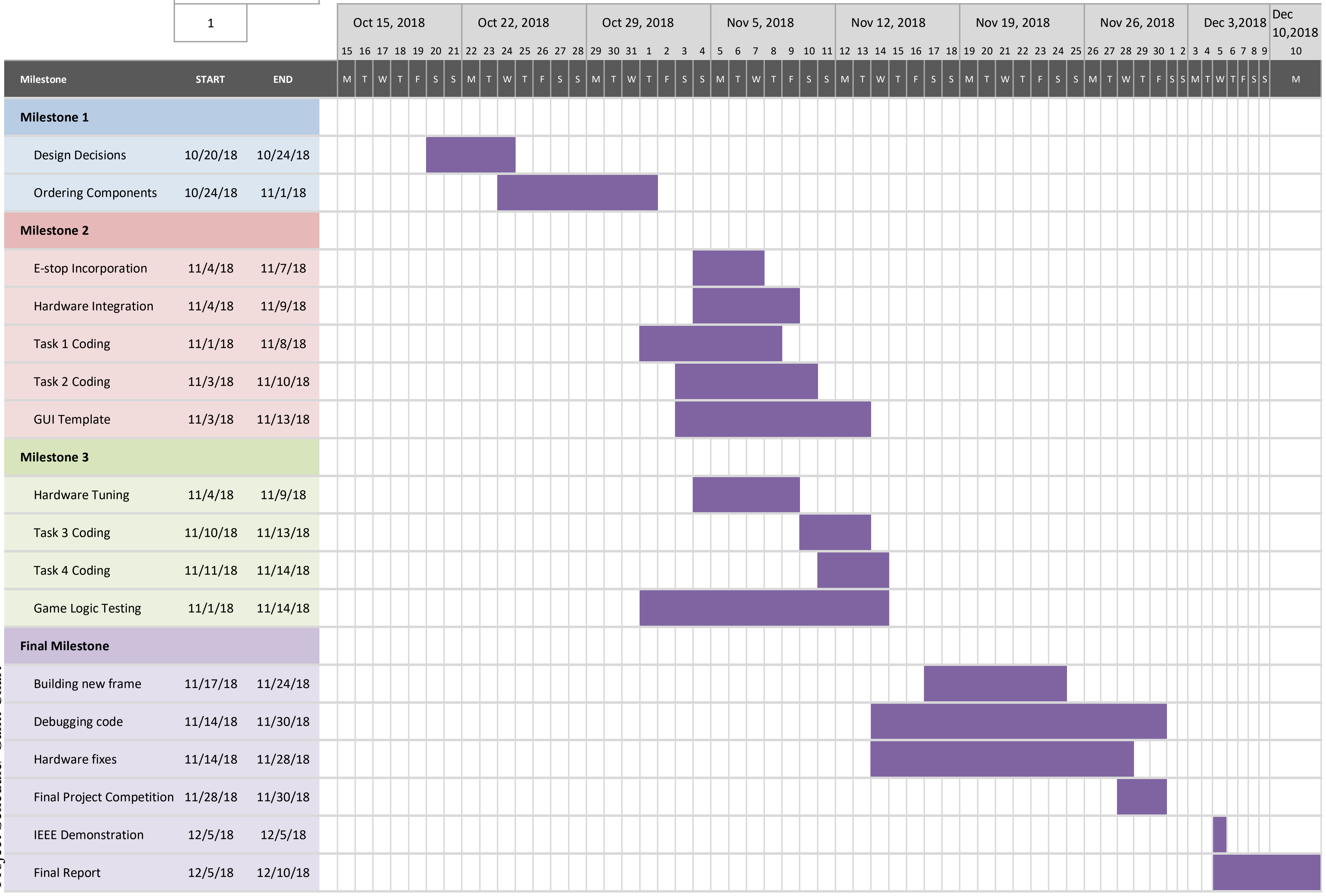
Project 4 (Final Project)

ECE 4950
Group 11

SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Sat, 10/20/2018

1



Milestone	START	END
Milestone 1		
Design Decisions	10/20/18	10/24/18
Ordering Components	10/24/18	11/1/18
Milestone 2		
E-stop Incorporation	11/4/18	11/7/18
Hardware Integration	11/4/18	11/9/18
Task 1 Coding	11/1/18	11/8/18
Task 2 Coding	11/3/18	11/10/18
GUI Template	11/3/18	11/13/18
Milestone 3		
Hardware Tuning	11/4/18	11/9/18
Task 3 Coding	11/10/18	11/13/18
Task 4 Coding	11/11/18	11/14/18
Game Logic Testing	11/1/18	11/14/18
Final Milestone		
Building new frame	11/17/18	11/24/18
Debugging code	11/14/18	11/30/18
Hardware fixes	11/14/18	11/28/18
Final Project Competition	11/28/18	11/30/18
IEEE Demonstration	12/5/18	12/5/18
Final Report	12/5/18	12/10/18

Project Schedule/ Gantt Chart

ECE 4950 Project 4 – Customer Requirements and Final Design Parameters

Use the guidelines below to complete your report and add at the end of your report.
Group Member Last Names: Lewis, Swalhah, Sergent, Hart, Reza

Score	Pts		Performance Indicators
	5	**General Format - Professional Looking Document/Preparation (whole document)** a) Fonts, margins (11pt, times new roman, single spaced. 1" margins on all sides). b) Spelling and grammar are correct c) Layout of pictures – all figures need numbers and captions and must be referenced in the text d) Follows the page limitations below. e) References. Use IEEE reference format. f) This grading sheet is included as the final page.	g.1
	0	**Page 1: Title, Group Name, Group Members, and Date** **Executive Summary** (1 concise, well-written paragraph) Provide an overview of this project. Briefly describe what you did and what you learned.	g.1
	5	**Page 2: Engineering Requirements** (<1 page) Bulleted list of Final Design Engineering Requirements	e.1
	10	**Pages: 3-7: Design Details (<5 pages)** Describe a system that can be built including System Architecture and System Integration based on the Engineering Requirements. Do not include data sheets or software code.	e.2
	10	**Page 8: Analysis of Final Prototype Performance** (<1 page) Did it succeed or fail to meet customer requirements? What went wrong and what happened in the design process to allow this problem? Make a table of the customer requirements and address how well your design met these expectations.	e.3 e.4 i.1
	5	**Page 9: Project Schedule/Gantt Chart** (<1 page) Create a schedule (Gantt chart) that shows the tasks and schedule for your project. Start from the very beginning of your project and extend to the end (completing final report and presentation).	k.2
		Page 10 This grading sheet is included as the final page.	
	50	Laboratory demonstration of your prototype (evaluated by instructor and TAs). Evaluator will manipulate the interface and evaluate how well the system provides the timing and display functions (i.e. how well does the closed loop control work). Is it well built? Neat wiring? (.6 * the prototype evaluation score)	g.2
	15	Rating by reviewers during competition	g.2