Shane Lester
Professor Eric Schweitzer
Embedded Systems
December 18th, 2018
Final Paper

How the project works:

This project is a tic tac toe arcade style video game booth. Its arcade booth casing is made out of Lego. It supports single player vs. the computer or multiplayer vs. a friend. Upon plugging it in, it displays a welcome message. After that, it asks to select a game mode (single or multiplayer). If multiplayer is selected, it will begin to commence gameplay. If single game mode is selected, it will commence gameplay. The computer will choose the option left over. Upon winning the game, the player who won will be outputted to the screen. If there is a tie, it will output that there is a tie. After a game finished, the first menu (select a game mode) is outputted to the user again. The system repeats from there.

To control menus and select pieces, a joystick (10k analog potentiometer) is used. To click options, an LED button is used. An LED matrix displays the output that the user sees.

The program is implemented in the C++ programming language using the Arduino IDE.

The programs 'main' function is in the file Arcade_Booth.ino. It is where all the modules are connected and the gameplay loop is. The files draw_board.cpp and draw_board.h are where the gameplay is physically drawn to the matrix. The module menu.cpp/ menu.h is where some of the games menus and pop up messages are displayed (others are embedded in the main function, because sometimes it made more sense when the joystick was involved) . The module game_logic.h/.cpp is where the game data and all game functionality is stored. Joystick.cpp / .h is where the joystick driver program is written. It allows the main function to interact with the joystick. The button is implemented in the main file with a few lines of code. Because it was so simple to implement, it was the only hardware input that didn't require a module. The only external libraries used was the driver program for the matrix. It is listed in the 'hardware components used' under the hardware component in which used it.

Hardware Components Used:

- Two 16 by 24 LED chainable matrices produced by Adafruit. They were used as a screen to display the tic tac toe game.
    - Note that this required an external library. It was a driver to interface between the programmer and the hardware. This library was downloaded from https://github.com/adafruit/HT1632 .
    - The part was bought at: https://www.adafruit.com/product/555?gclid=EAIaIQobChMIir6Yk-an3wIVAQOGCh3SggV2EAQYAyABEgKsF_D_BwE

- Elegoo Uno R3 produced by Elegoo via their super starter kit.
- Arduino Uno R3 shield produced by Elegoo via their super starter kit.
- 14 short male to male wires from Elegoo via their super starter kit.
    - There components were bought at :https://www.elegoo.com/product/elegoo-uno-project-super-starter-kit/
- Arcade style 30mm LED button produced by Adafruit. Used to select items throughout the experience.
    - This part was bought at https://www.adafruit.com/product/473
- Mini analog joystick – 2 10K potentiometers by Adafruit. Used to scroll through selections throughout experience.
    - This part was bought at https://www.adafruit.com/product/3102?gclid=EAIaIQobChMIgMrlp-un3wIVgh-GCh1TbghxEAQYASABEgJd__D_BwE
- 9V, 1A generic AC to DC power supply supplied by Tinkersphere. Used to provide power to the system.
    - This part was bought at : https://tinkersphere.com/arduino-compatible-components/142-9v-wall-power-adapter-arduino-compatible.html?gclid=EAIaIQobChMIjN__suun3wIVxCaGCh2powcHEAQYAiABEgKCffD_BwE
- Perma-Proto Half-sized Breadboard PCB produced by Adafruit. Used to solder male to male wires from the Elegoo Uno R3 to the Joystick.
    - This part was bought at: https://www.adafruit.com/product/1609?gclid=EAIaIQobChMIuvWR9YWo3wIVB4bICh1qIg2kEAQYAiABEgKI-PD_BwE
- Loose Lego bricks purchased at the Lego store. Used as a shell for the arcade booth. The bricks came in three types:
    - Blue 4 by 2 studs (the amount was two small containers from the Lego store). There containers are standardized and can easily be repurchased, however no link exists for them from an online outlet. Must be purchased at a Lego store location that sells loose bricks.
    - Black 6 by 2 studs (the amount was two large containers from the Lego store. There containers are standardized and can easily be repurchased, however no link exists for them from an online outlet. Must be purchased at a Lego store location that sells loose bricks.
    - One blue Lego baseplate 32 by 32 studs
        - Bought at: https://shop.lego.com/en-US/product/Blue-Baseplate-10714

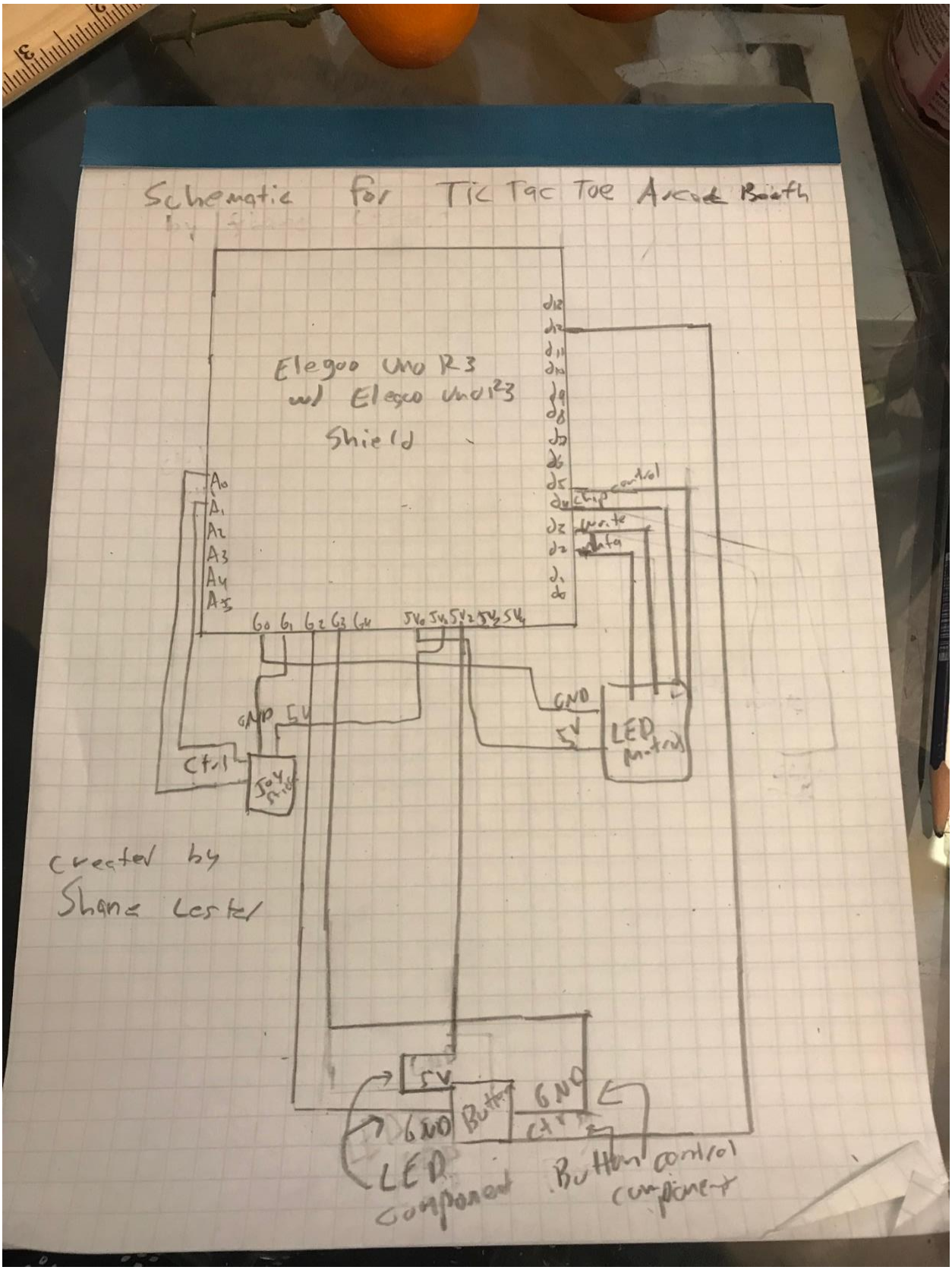
Deviations:

- I originally wanted to use 4 LED chainable matrices. I thought you could chain them on the side or on the bottom/top. It turns out you can only chain them on the side. Using 4 of them rather than two would have looked really bad.
- Initially I included two joysticks and two buttons. The reason I didn't include these in the system is because they looked really bad. The system is very small (especially because

the screen only included two matrices rather than 4) so adding an extra joystick and button turned out to look horrible.

- I initially wanted to use a 9v battery and an on/off switch. An arcade system is typically plugged into the wall, so it just looked and felt awkward using batteries. The system is also big and cumbersome, so batteries are unnecessary. Batteries are only used when transportation is important. It wouldn't ever make sense to not have it plugged in. As a consequence, the on/off switch was not necessary once the system was off batteries.
- Implementing difficulty levels required artificial intelligence. That wasn't implemented because I ran out of time for it. I researched the algorithm and had it half developed in Python, but I needed one more day to finish writing it in Python, rewrite it in C++, and then to implement it. With the Minimax algorithm, implementing difficulties becomes trivial (would only require a random mode and an AI mode, or always AI mode where they choose a random position some percentage of the time). The minimax algorithm was relatively simple to implement, but unfortunately, I ran out of time. Therefore, the system only has one difficulty, which is a random selection mode. However, iterating on the system to include a harder mode is very straightforward and would only require one more full day of development.
- Didn't implement computer being able to go first or second. That was just because I ran out of time. It would be simple to implement, just a menu. The game logic supports both the computer going first and second.
- Displaying statistics seemed overkill so I did not implement it. It would have been simple to implement, but I instead opted to allow the user to switch from single to multiplayer after each game. I implemented that instead purely because I thought it was cooler.

Schematic:


Schematic for Tic Tac Toe Arcade Booth
Elegoo Uno R3
w/ Elegoo Uno R3
Shield
GND
5V
LED matrix
5V
GND
Button
GND
5V
LED component
Button control component
created by
Shane Lester