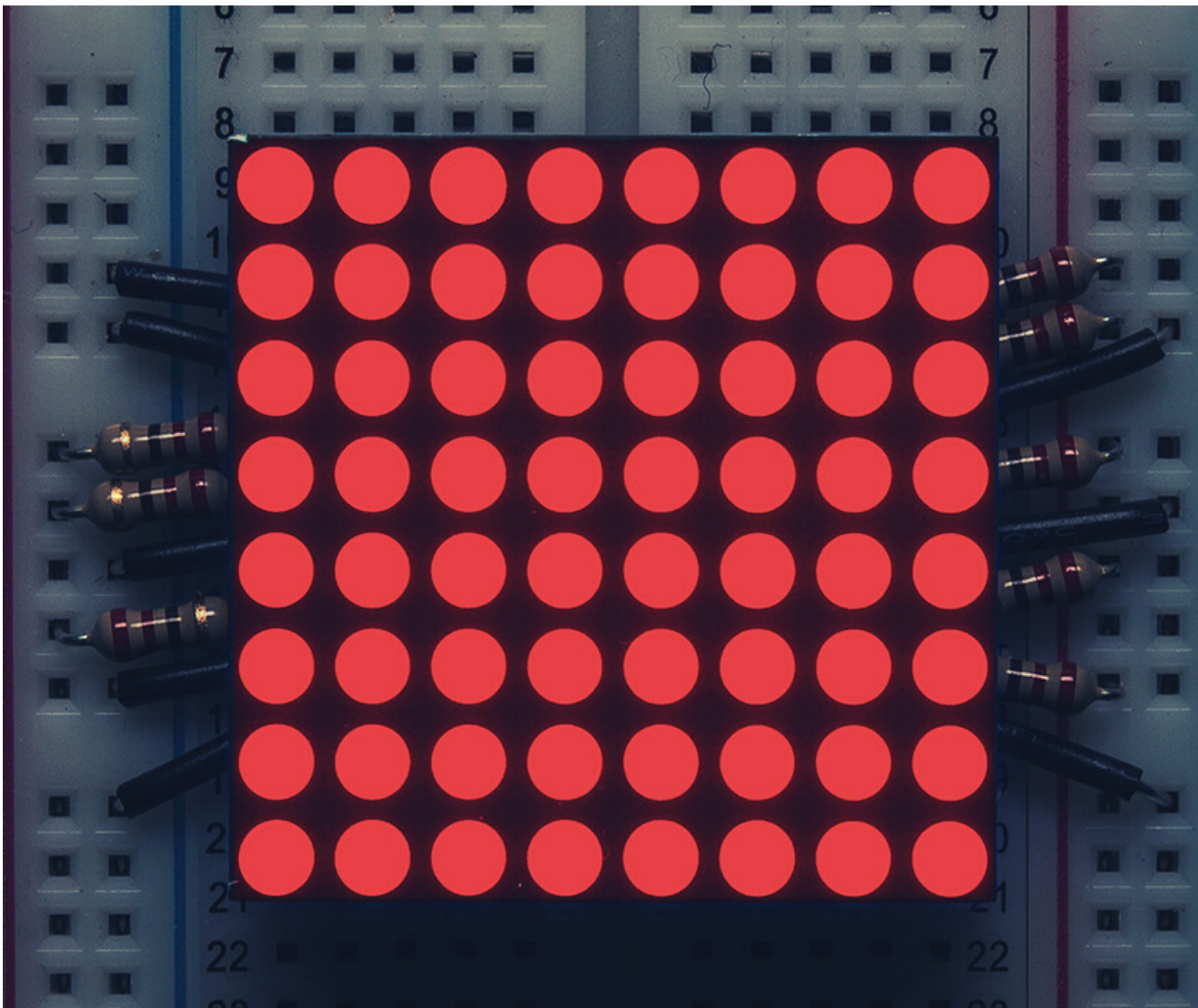
BATTLESHIP

ARDUINO CREATIVE INTERACTIVE PROJECTS
ASSIGNMENT 1

WALEED RAJA NOUREEN



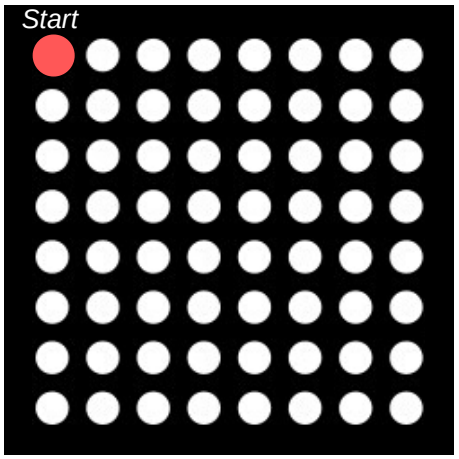
USERGUIDE

1. A user manual: Instructions on how to play and interact with the game.

The game starts with the Player (LED) on the top left-hand side. Where they can utilise navigation buttons to control their location throughout the 8 by 8 grid. There is a hidden ship randomly located within the grid taking 4 slots (4 LEDS) and the player will have to find the ship. By navigating through the grid, the user can select which one of location to target to see if they have found the ship. Once the user selects the give location, the display will tell whether they have found the ship or not. and display the target location hit. Once, the player has found all 4 spots of the ship, they will be indicated their victory or on the other hand, if they guess and fail 10 times, they will lose and will have to restart again.

KEY CONTROLS

Up *Down* *Confirm Choice* *Left* *Right*



2. A brief discussion of the design decisions taken

Throughout utilising Arduino and creating the battleship game, I utilised an 8x8 led matrix which required me to download additional command libraries to utilise the hardware. I tested out various led matrices to see which will be time efficient but also effective to which led me to the current one which I utilised within this project. This hardware required a byte array (containing 8 slots, 8 bits) defining which LED within each row should be turned on by giving it a state of 1. For example, if I wanted to turn on the LED on the right in the first row, I would generate a byte of 00000001 where the most significant bit is the left-hand side and the least is the right. At first, I attempted to generate each individual byte and call it individually which took a large amount of screen space during designing as seen in the program. Hence, I generated a function which iterates through the byte array instead of being manually calling it throughout the code. Instead, I just needed to generate it as a variable.

Furthermore, another problem that I faced through within the design process was the button controls. This issue caused the led to exceed limit of the LED matrix and keep continuing over. Hence, I needed to add a feature where the led will go to the other side and by changing the current x and y values, so they do not exceed the maximum. This required me to turn off the previous LED state to off to give the indication of movement.

Moreover, another issue that I came across was generating the random ship location. At the beginning, I utilised the random function to generate an X and Y value between the minimum and maximum of the led matrix. and plot them to the display. However, I came to notice that when restarting the program, the ship was located at the same coordinates every time. This transpired because the random function is only a pseudo random generator which utilises the same stream of pseudo-random numbers when the program is restarted.

Thus, I needed to implement the use of randomSeed function. which uses an arbitrary point in its random sequence. It is important for a sequence of values generated by random to differ each time. This is done by creating an analogue read from an empty pin. This gave me random values each time when the program started.

Other design decisions such as using Boolean states to give each correct LED chosen so they cannot use the same spot 4 times to win. As well as utilising auditory and visual features for users with impairment such as Piezo.

3. Breadboard layout sheet

This page contains the circuit diagram of my project showing the connection between each component on the Arduino board. The diagram was made in Fritzing.