CSE 316 - Project Report

Project Name: Arcade Shooting Game

Submitted by:

1705073 Istiak Bin Mahmod

1705086 Shafaet Zaman

Supervisor:

Mahjabin Nahar

Lecturer

Department of Computer Science & Engineering

Bangladesh University of Engineering & Technology

July 30, 2021

Contents

1 Introduction

For our 316 Project we have tried to simulate an arcade shooting game. The motivation behind this project was that we all have played shooting games when we were child in a retro console type device. Now with our knowledge of Microprocessor and Microcontroller taught in CSE 315 course we have tried to build that game to see how the things work together. For this project we have used *ATMEL Studio 7 & Proteus 8* software to build our code and simulate the game.

2 Components

Software:

1. Proteus 8
2. ATMEL Studio 7

Equipment:

- ATmega32

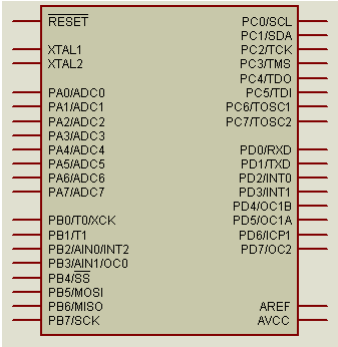

Figure 1: ATmega32

- 3 to 8 line decoder (74LS138) (5 pieces)

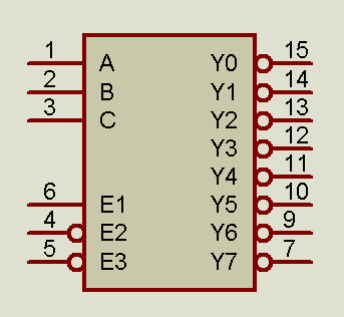

Figure 2: 3-8 Line Decoder

- 8 x 8 red LED Matrix (4 pieces)

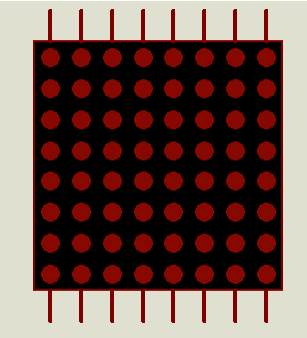

Figure 3: 8x8 LED Matrix

- LCD Display (LM016L)

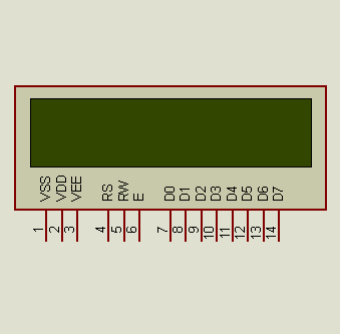

Figure 4: LCD Display

- NOT Gate (7404)

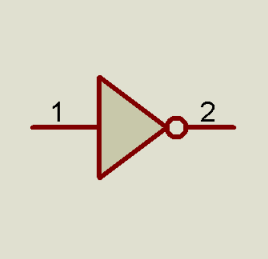

Figure 5: Not Gate

- Button

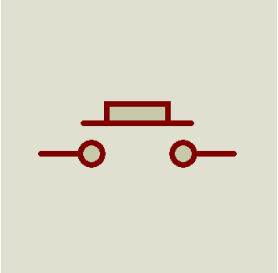

Figure 6: Button

- Resistance

Figure 7: Resistance

In this project we didn't have to use any sensors and actuators.
We have used 4 LED matrices to show our gameplay. The LED matrices were multiplexed by using decoders and not gates.
LCD display was used to show the score and game status.
Button were used to control the game (Movement and Fire).

3 Circuit Diagram

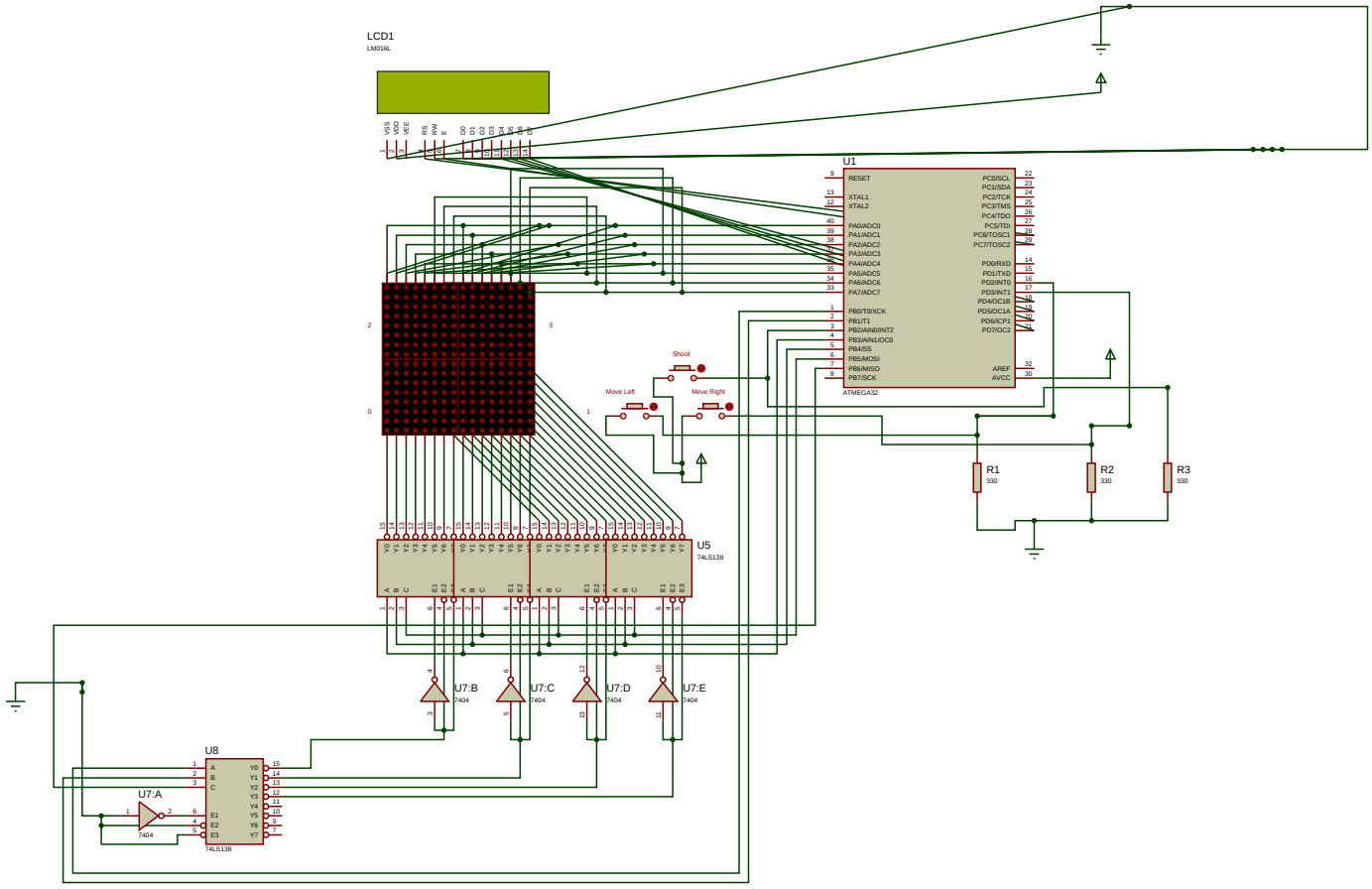

Figure 8: Circuit Diagram

4 GamePlay

We have tried to keep the gameplay and game logic simple for the ease of the player.
There will be a ship, from where the player can shoot fires to the enemy bullets which are coming from above continuously.

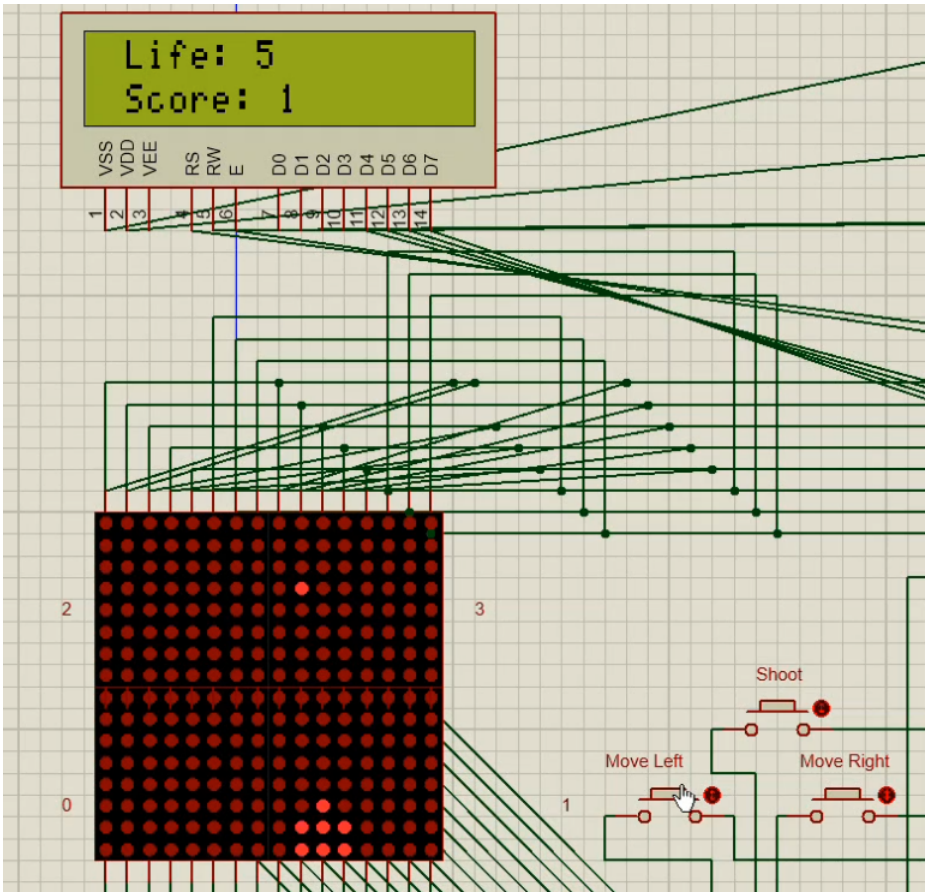

Figure 9: Gameplay

4.1 Movement

Player can move left and right using Move Left and Move Right buttons across the whole LED matrix. Player can move to suitable position to avoid collision with the bullets so that he doesn't have to lose life.

4.2 Shoot

Player can shoot bullets as he/she wants. A player can fire 3 bullets at a time.

4.3 Scoring

If the player bullet hits the enemy bullet, player will gain a score and vice versa for the enemy bullet. As game score increases, the difficulty of the gameplay will also be harder. At first only one bullet at a time will fall from above, but as gamescore increases player will see more bullets coming fro above and they will also come faster towards the ship.

4.4 Element of surprise

At certain score points the player will see that, bombs as well as bullets will come from above. It takes only one bullet to hit the bomb, but it has to hit the bomb right in the middle. If player can hit the bomb, he will gain 2 point and if the bomb hits the player he will lose two lives.

4.5 Ending

Player can score as much as he/she wants until he has finished all 5 lives he/she was given. There will be a nice animation at the begin and at the end of the game.

5 Conclusion

Yet the gameplay seems so simple, we faced some challenges while building the circuit as well as in the coding part as logic section the heart of a game.

In the circuit, first of all it was a challenge to determine whether we have to use one Atmega32 or more than one. Then we decided to use one ATmega32. Then it was the question of how more than one LED matrices can be enlighten at a time. Because in our 316 offline we were taught to use just one LED matrix. To solve the problem, we have used 5 decoders. Among them one decoder is directly connected to the Atmega32 and the other 4 decoders are connected to 4 LED matrices. According to the code and rule of decoder the main decoder can control the enabling or disabling part of the other 4 decoders. Thus we were able to use the LED matrices as we wanted. To see if each of the LED's were working we used a function called ShowLED(). To implement the function we had to make a 16 x 16 matrix and some addressing and logical stuffs in our code.

Now the game logic challenge. Though it was simple movement and shoot. The main challenge was to show all the LEDs simulataneously so that the gameplay looks smooth. We have used buttons to control the game and each of the buttons did call an interrupt to the ATMega32. As it was the matter of interrupt, we had to code the delays accordingly after some trial and error approach.

We have used C plus plus language for the code and have divided every part like bullets, firing, bombing etc in different classes.

Still some issues like certain blinking problem with the LED still remains, which we couldn't fix.

After all, the gameplay looks smooth and enjoyable and a player won't get bored and certainly can feel the nostalgia.