CENG 3006 INTRODUCTION TO EMBEDDED SYSTEMS SNAKE GAME

Gizem PESEN 170709050

Thursday 6th May, 2021

Abstract

Main goal in this project is to design a fun snake game that can be controlled with a joystick. This is a project draft, the program has been successfully written in C language, it is simulated by using buttons instead of joysticks.

Note : You can click the blue links to go to the specified location.

Contents

.

1 Introduction

Snake Game is the common name for a video game concept where the player maneuvers a line which grows in length, with the line itself being a primary obstracle. [1] The concept originated in the 1967 arcada game Blockade[2] Snake game became the favorite game of all children of an era. One of the reasons why the snake game is so popular; Nokia was putting the game on a led matrix with Arduino and let the player control snake with a joystick.

2 Project Details

It is planned on Project proporsal to use 8*8 Led Matrix display for displaying the snake and its food dot, Joystick for giving directions and starting the game , and finally an Arduino Uno to control the whole process. In this Simulation part, Except Joystick, We stayed true to everything we planned in Project Proporsal.In the final part, the joystick will be used in the project, but while simulating, such a way has been followed since there is no joystick in the simulation programs.

All codes are written in C language on Arduino Ide [13] and successfully simulated as planned using the Proteus 8 program.

3 Implementation

3.1 Libraries

3.1.1 LedControl

LedControl is a library for the MAX7221 and MAX7219 Led display drivers. Only a few components are needed to control 64 Leds or a 7-segment display.[4] The library can be added from the add libraries in Arduino IDE section, as well as we added it a zip file from github. [5]

3.1.2 MD MAX72XX

It is added this library a zip file from github. [3]

3.2 Functions

3.2.1 setup()

As seen in the code below, we can adjust or clear the light from the **Led Control** library. This is for opening setting brightness and clearing things.

Listing 1: LedControl

```
//setup the chip
lc.shutdown(0,false); //wake it
lc.setIntensity(0,15); //set brightness
lc.clearDisplay(0); //clear it
```

.

Listing 2: Buttons

```
#define UP 2
#define RIGHT 3
#define DOWN 4
#define LEFT 5
#define START 6

  pinMode(RIGHT, INPUT_PULLUP);
```

The keys connected to Arduino are defined in the code, the direction of the snake can be adjusted with the **defined direction keys**.For using the keys it is defined **pinMode** used.The picture below shows how all buttons are connected to Arduino. The reason that the 2,3,4,5,6 defined in the code matches this way is because they are connected in this way.

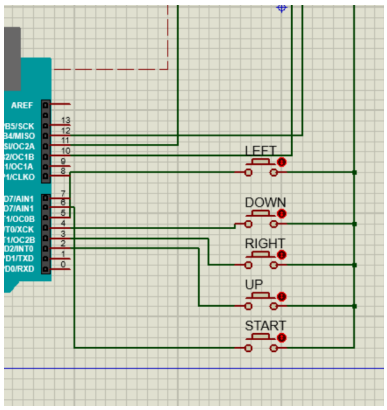

Figure 1: Buttons and It is connection with Arduino

Listing 3: For loop

```
for (int i = snake_len; i < START_LEN; i++){
    extend_snake();
}
new_apple(); //create new apple function
```

In this for loop we call **extend snake** function. With this part we can increase the **lenght of snake**.

3.2.2 loop()

At first we define **gamestart** false as boolean. This makes the code aware of whether the game has started or not. The **text Start will scroll** on the screen until the user presses the start button.

Listing 4: Start Message

```
bool gamestart = false;
  while(digitalRead(START)){
    scrollMessage(scrollTextStart); //to show start message on the dot matrix
    }
```

Listing 5: While Loop

```
    gamestart = true;
    while(gamestart){
        if (am_i_dead()){
          scrollMessage(scrollText); //it shows game over
          gamestart = false;
          snake_len = 1; //starting snake lenght
          snake[0][0] = random(0,8); //start snake random place
          snake[0][1] = random(0,8);
```

Within this **while loop** in the code, there is an **am I dead** function. Whether the snake died within the cycle is questioned.

3.2.3 am i dead()

As can be understood from the name **Am I dead**, it is the function called in the **loop** section, where we understand that the snake has not died according to its size.

Listing 6: Am I dead

```
bool am_i_dead(){
    for (int i = 0; i < snake_len-2; i++){
        if (snake[i][0] == snake[snake_len-1][0] && snake[i][1] ==
            snake[snake_len-1][1]){
           return true;
        }
    }
    return false;
}
```

3.2.4 check eaten apple()

Did the snake eat the apple or not? This is a question that can be queried with an if statement. Therefore, if is used in check eaten apple function. If the apple is eaten, the length will be increased with Extend Snake and a new apple will be created with the New Apple function.

Listing 7: Check Eaten Apple

```
void check_eaten_apple(){
    if (snake[snake_len-1][0] == apple[0] && snake[snake_len-1][1] == apple[1]){
        //eaten an apple
       extend_snake(); //call extend snake
       new_apple(); //call new apple
    }
}
```

.

3.2.5 new apple()

With the new apple function, a new apple is formed randomly.

Listing 8: New Apple

```
void new_apple() {
     apple[0] = random(0,8); //new apple in random place
     apple[1] = random(0,8);
}
```

3.2.6 move snake()

With the Move snake function, the snake is able to move forward.Arrays have been used to drive the snake forward.

Listing 9: Move Snake

```
void move_snake(){
    for (int i = 0; i < snake_len-1; i++){
        snake[i][0] = snake[i+1][0];
        snake[i][1] = snake[i+1][1];
    }
    snake[snake_len-1][0] = (snake[snake_len - 2][0] + vx + 8) % 8;
    snake[snake_len-1][1] = (snake[snake_len - 2][1] + vy + 8) % 8;
}
```

3.2.7 extend snake()

Extend Snake function has been used to lengthen the length of the snake using Array.

Listing 10: Extend Snake

```
void extend_snake(){
    snake[snake_len][0] = (snake[snake_len - 1][0] + vx + 8) % 8;
    snake[snake_len][1] = (snake[snake_len - 1][1] + vy + 8) % 8;
    snake_len++;
}
```

4 Simulation

4.1 Proteus 8 Set Up

In this project, proteus 8 program was used, considering that it is suitable for use. [6] [7] [8] One of the differences between Proteus and Simule Ide is the use of a library for Arduino access in the Proteus program. We had to research it to Simulate with Arduino Uno. .

4.2 System Kits On Simulation

- Arduino Uno

  A library for Arduino Uno simulation has been added to this program. [9]

- Matrix 8*8 Red

  It is recommended to use Red Matrix because first I tried with blue one and it didn't show the game clearly.

- Max 7219

- Buttons (Left, Down, Right, Up , Start)

watch [DEMO]

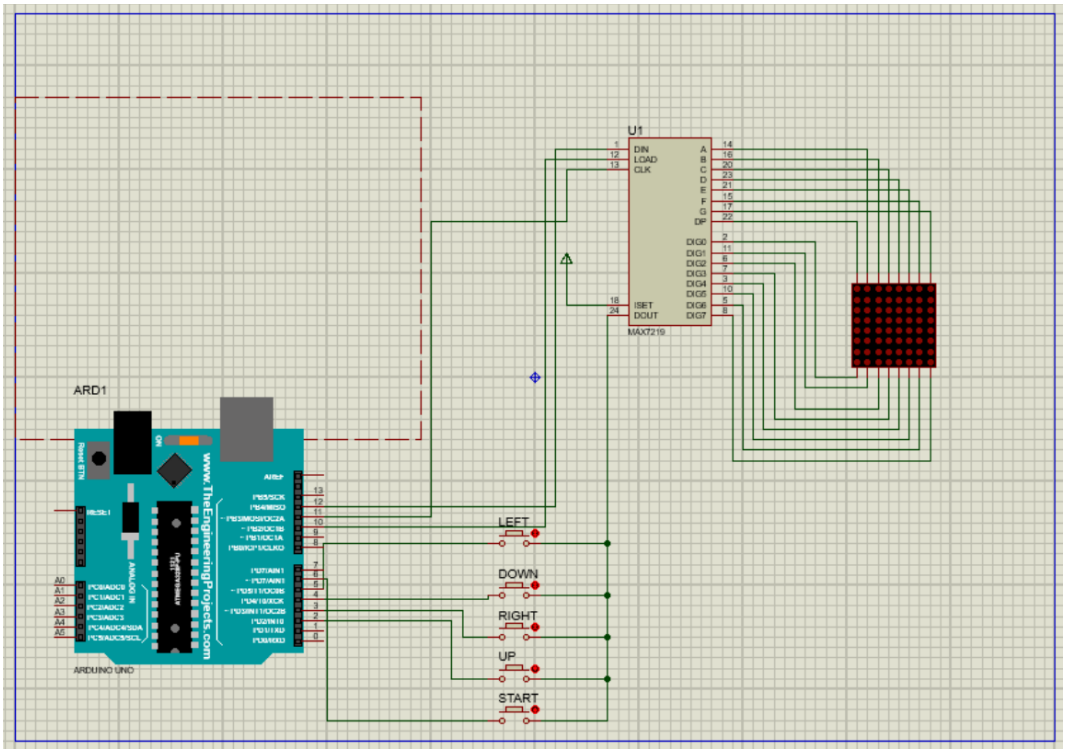

Figure 2: Simulation

4.3 Connection

It is connected between Led Matrix and max7219 . To connect them watched videos and searched whole things. [10] [11] . [12] .

.

5 References

[1] Gerard Goggin (2010),Global Mobile Media,Taylor Francis, p.101 , retrieved April 7, 2011
[2] Rusel DeMaria Johnny L. Wilson (2003). High score!: the illustrated history of electronic games (2 ed.). McGraw-Hill Professional. p. 24. ISBN 0-07-223172-6. Retrieved April 7, 2011.
[3] https://github.com/MajicDesigns/MD_MAX72XX
[4]http://wayoda.github.io/LedControl/
[5]https://github.com/wayoda/LedControl
[6]https://www.youtube.com/watch?v=0mckp3UdVmc
[7]https://www.youtube.com/watch?v=1yl9lSKIXkM
[8]https://maker.pro/arduino/projects/how-to-simulate-arduino-projects-using-proteus
[9]https://youtu.be/eeRzhR1pfnY
[10] https://circuitdigest.com/microcontroller-projects/arduino-snake-game-using-8x8-led-matrix
[11] http://ucboyutbilgi.blogspot.com/2018/10/gunler-bugun-size-arduino-ile.html
[12]https://www.youtube.com/watch?v=Ko1ErQx622M&t=232s