SIMPLE ARCADE GAME: SNAKE

Supervisors: Dr. Iman Elawady ; ENG. Michael B.Khani

Team STRAWBERRIES members:

▪ Enes Yurdatapan, 1810205079 * 1810205079@ogrenci.karabuk.edu.tr	▪ Kamoliddin Fatkhiddinov, 1910205506 1910205506@ogrenci.karabuk.edu.tr
▪ Ali Eren Ergün, 1810205041 1810205041@ogrenci.karabuk.edu.tr	▪ Ahmat Soumaine, 1810205023 1810205023@ogrenci.karabuk.edu.tr
▪ Ali Ramazan Taşdelen, 1910205003 alitasdelen@ogrenci.karabuk.edu.tr	▪ Sevgican Kılıç, 1810205078 1810205078@ogrenci.karabuk.edu.tr
▪ Batıkan Cımbıt, 1810205043 1810205043@ogrenci.karabuk.edu.tr	▪ Oğuzhan Portakal, 1810205048 1810205048@ogrenci.karabuk.edu.tr

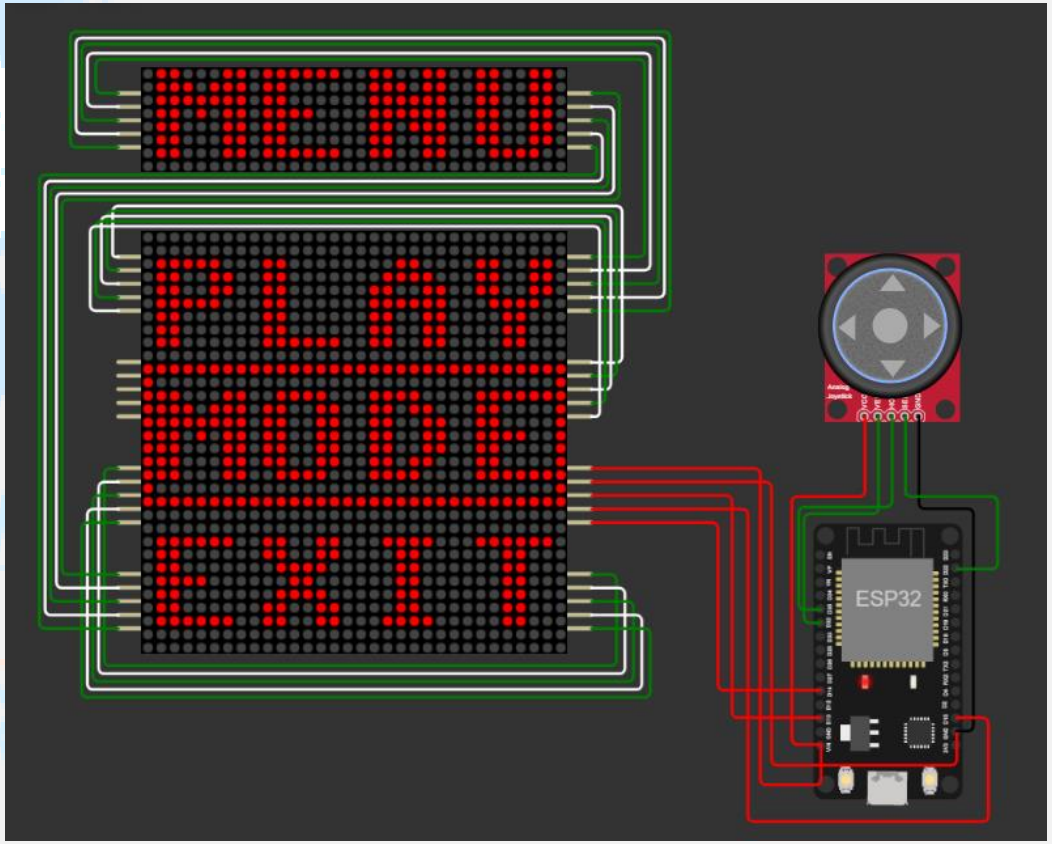
* Team leader and correspondent member

CONTENTS

INTRODUCTION

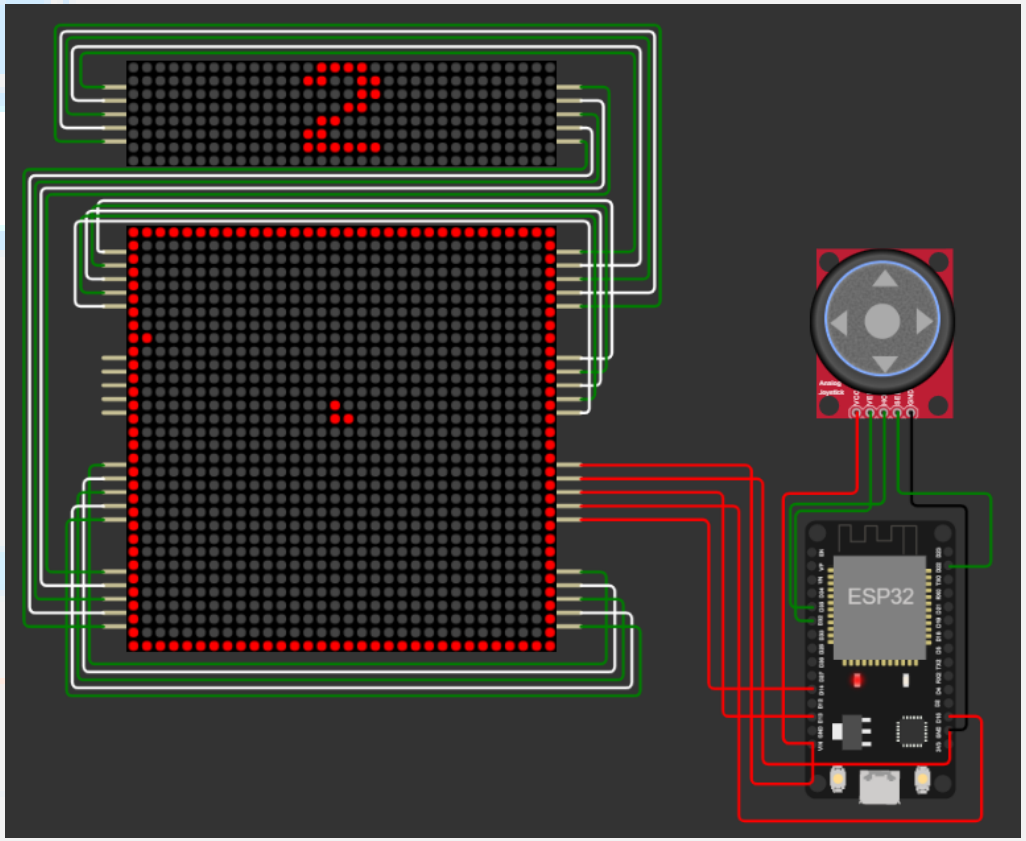
INTRODUCTION

- **Summary**
  - Due to the increase in the demand for the entertainment sector and mobile devices, it is possible to see that the sales in this sector have been at the highest point recently.
  - One of the most important points of winning in this sector is to be able to promote the product very well and to produce it cheaply.



INTRODUCTION

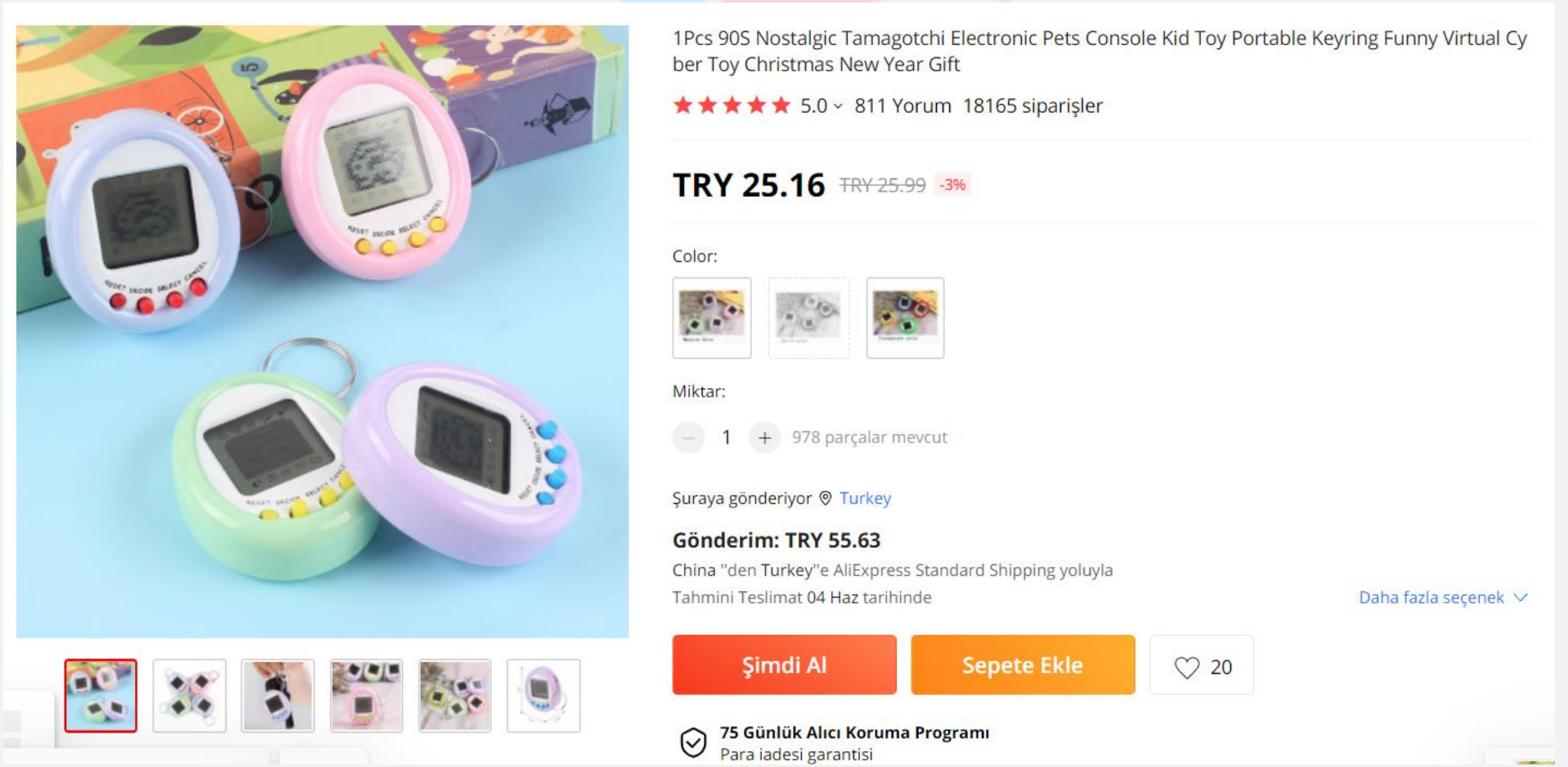
- **Goals**
  - We aimed to integrate traditional game which is an indispensable childhood enjoyment that people always remember, in a modern way.
  - We wanted to design it not only as a toy for children, but also for adults as a way of entertainment while waiting in line in anywhere or on a long journey.



RELATED WORKS

RELATED WORKS

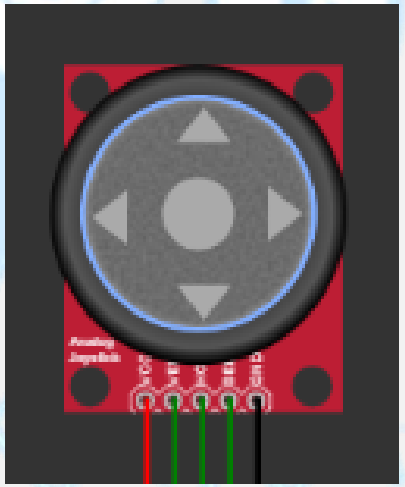
RELATED WORKS



PROJECT DESIGN

PROJECT DESIGN

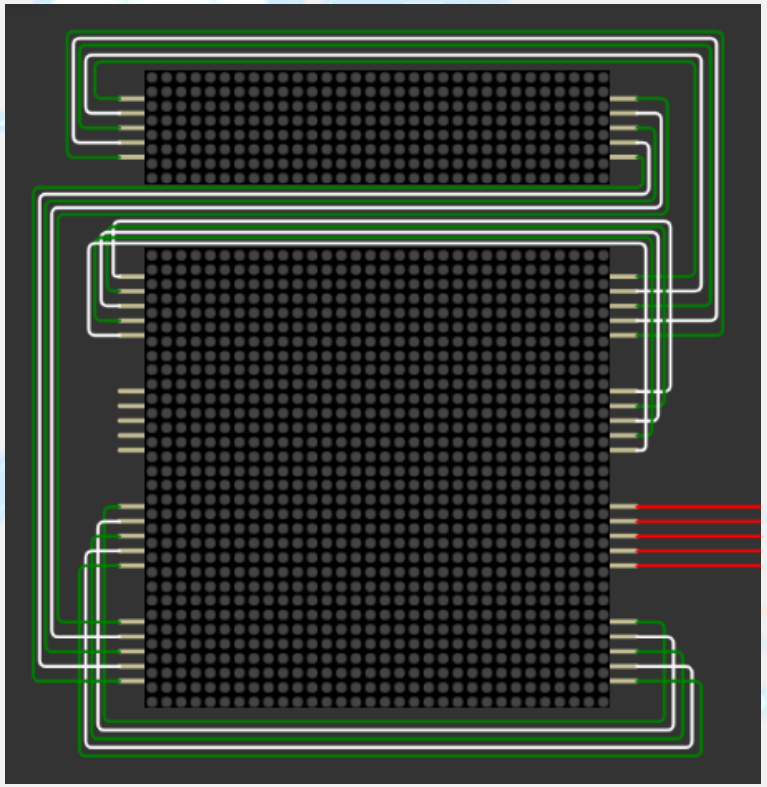
Analog Joystick



To control the game used Analog Joystick as input hardware

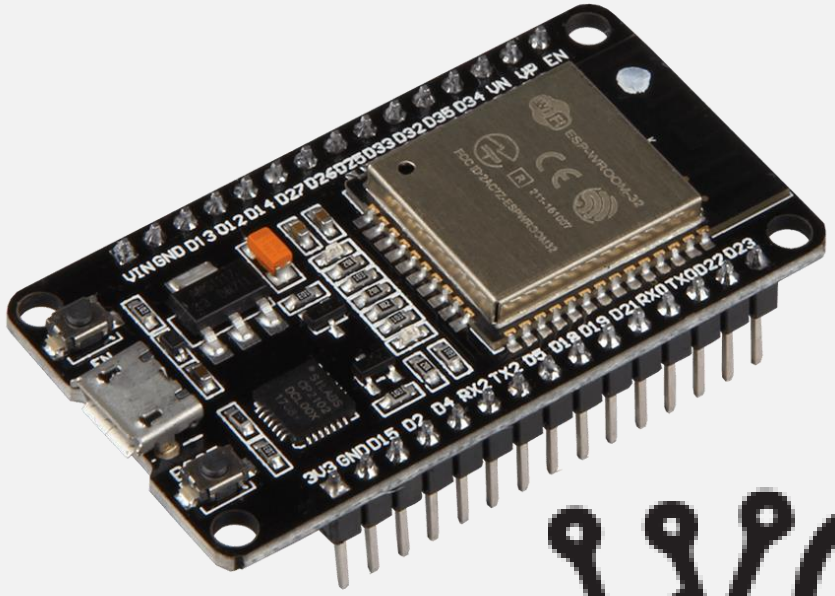
PROJECT DESIGN

8x8 LED Dot Matrix (MAX7219)

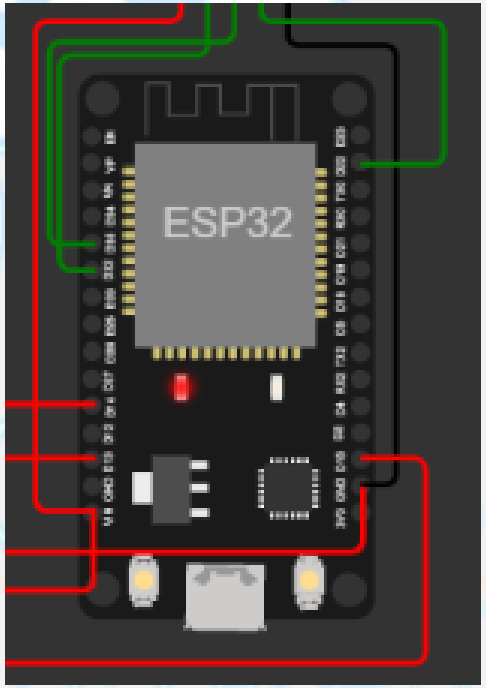


To display the game used 20 of 8x8 matrices connected to each other as output hardware

PROJECT DESIGN



ESP32



To control the game used ESP32 which connected input and output hardware

SOFTWARE

SOFTWARE

- **MAX7219 DRIVER**
  - To drive and use a custom display we found opensource driver MAX7219. But unfortunately, driver didn't work properly (e.g. driver displaying pixels upside down).
  - We changed some functions inside the library to make it suitable for our project.

```
1  from Max7219 import Max7219
2  from machine import Pin,SPI,ADC
3
4  spi = SPI(1, baudrate=40000000)
5  screen = Max7219(32,40, spi, Pin(15))
6  xAxis = ADC(Pin(35))
7  yAxis = ADC(Pin(32))
8  SW = Pin(22,Pin.IN, Pin.PULL_UP)
```

```
80      def show(self):
81          """Update display"""
82          # Write line per line on the matrices
83          for line in range(8):
84              self.cs(0)
85
86              for matrix in range(self.nb_matrices):
87                  # Guess where the matrix is placed
88                  row, col = divmod(matrix, self.cols)
89                  # Compute where the data starts
90                  if not self.rotate_180:
91                      offset = 8 * self.cols - row * self.cols * 8
92                      index = col + line * self.cols + offset
93
94                  else:
95                      offset = 8 * self.cols - row * self.cols * 8 - 1
96                      index = self.cols * (8 - line) - col + offset
97
98                  self.spi.write(bytearray([_DIGIT_0 + line, self.buffer[index]]))
99              self.cs(1)
```

SOFTWARE

- **GAME ENGINE**
  - We wrote code for Game Engine from scratch.
  - Tried to improve code maintenance.
  - Bunch of functions like Random Food and Obstacle Manager, Display and Mode Functions used together in a single Game Engine.

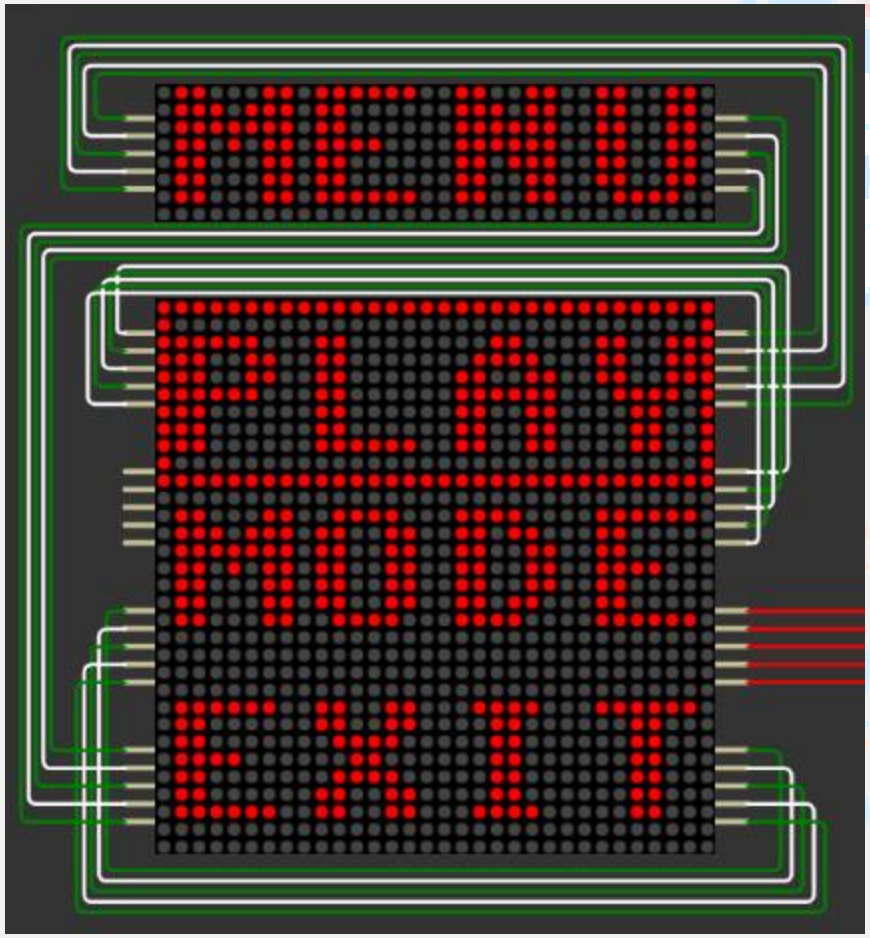
```
222         snake_List.append(snake_Head)
223         if len(snake_List) > Length_of_snake:
224           del snake_List[0]
225
226         for x in snake_List[:-1]:
227           if x == snake_Head:
228             game_close = True
229
230         self.our_snake(snake_List)
231         self.score_board(Length_of_snake - 1)
232
233
234         if x1 == foodx and y1 == foody:
235           foodx = random.randrange(0,31)
236           foody = random.randrange(0,31)
237           Length_of_snake += 1
238
239         self.screen.show()
240         self.screen.fill(0)
241
242     def draw_obstacle(self,line):
243       self.screen.line(line[0],line[1],line[2],line[3],1)
244
245     def random_obstacle_generator(self):
246       x1 = random.randrange(0,31)
247       y1 = random.randrange(0,31)
248
249       if random.randrange(0,2) == 0:
250         x2 = x1+5
251         y2 = y1
252         if x2>31:
```

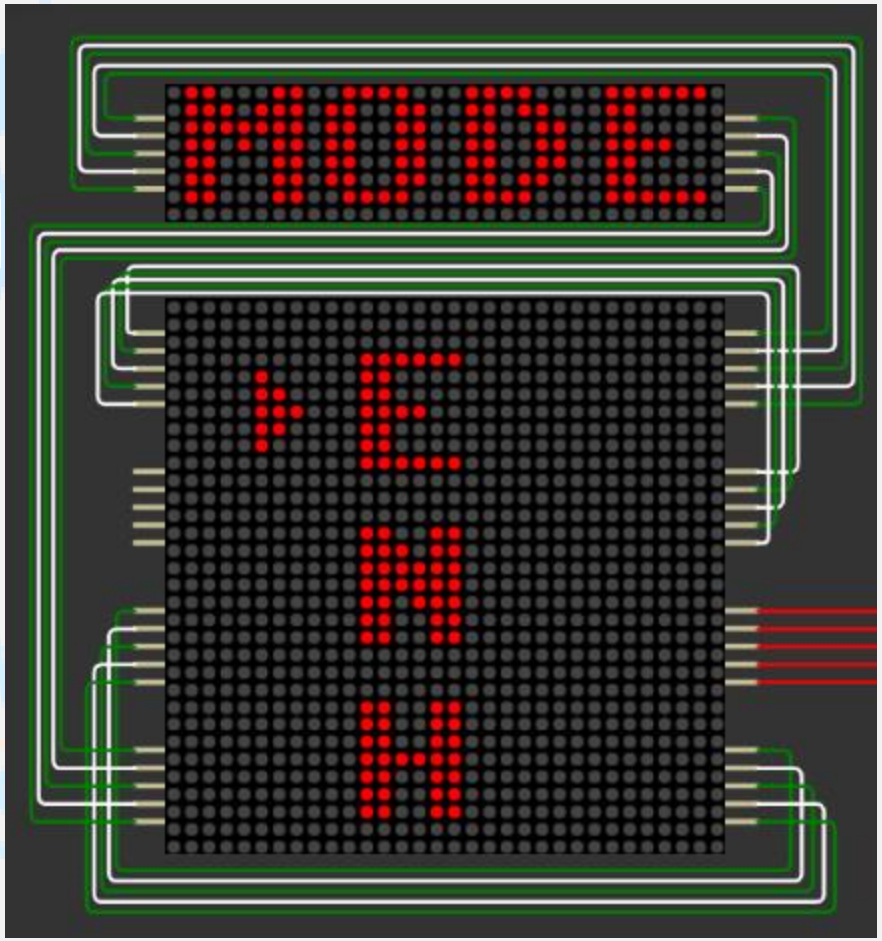
SOFTWARE

- **GAME MENU**
  - To select game difficulty and play we used the menu.
  - For future usage it can be also implement to select another games (e.g. Tetris, Pong etc.)

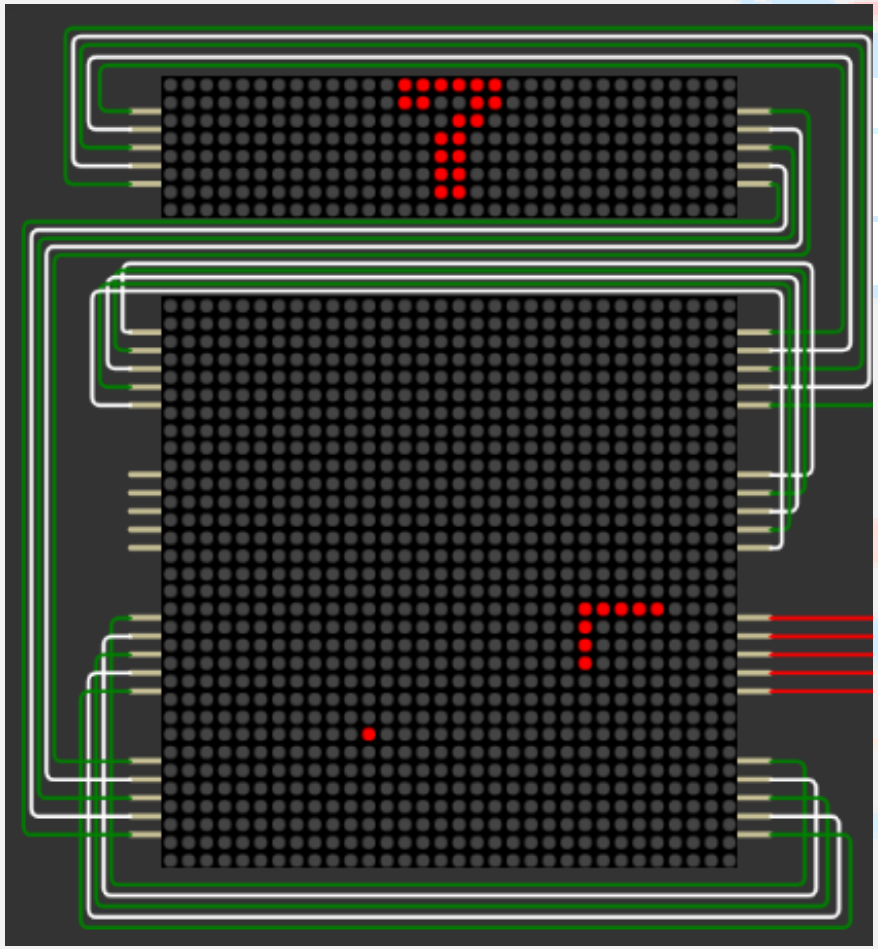
```
class Menu:
  def __init__(self,screen):
    self.screen = screen
    self.menu_index = 0
    self.mode_index = 0
    self.state = ""
    self.arrow = [[0,0],[0,1],[0,2],[0,3],[0,4],[1,1],[1,2],[1,3],[2,2]]

  def get_joystick_state(self):
    xRef = xAxis.read_u16()
    yRef = yAxis.read_u16()

    if yRef == 0:
      self.state = "down"
    if yRef == 65535:
      self.state = "up"
    if xRef == 0:
      self.state = "right"
    if xRef == 65535:
      self.state = "left"

  def display_menu_default(self):
    self.screen.fill(0)
    self.screen.text("MENU",0,32,1)
    self.screen.text("PLAY",0,2,1)
    self.screen.text("MODE",0,12,1)
    self.screen.text("EXIT",0,23,1)

  def display_mode_default(self):
    self.screen.fill(0)
```

GAME PHOTOS

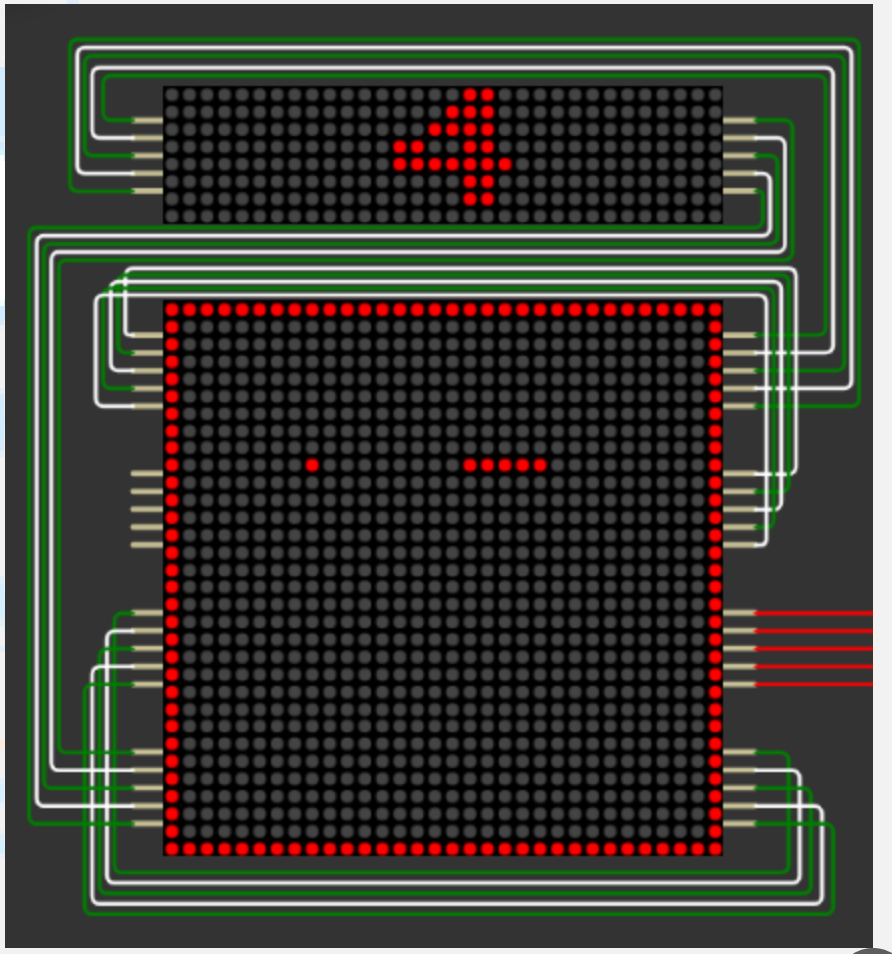




GAME MODES

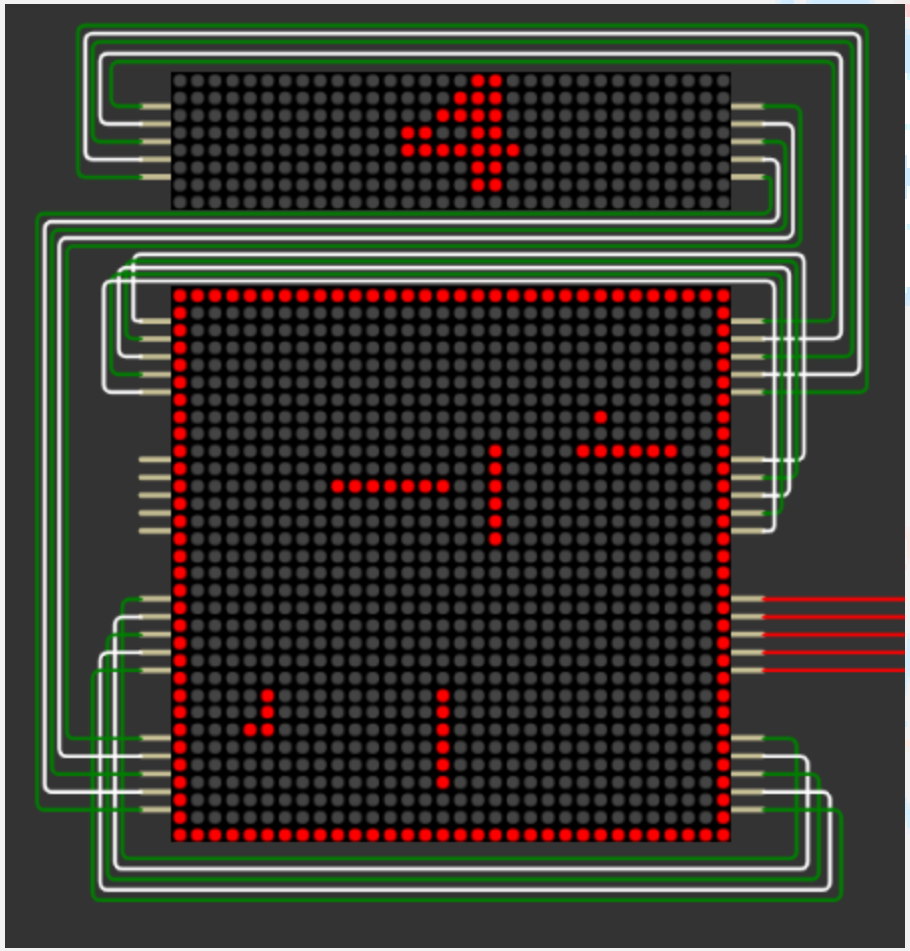


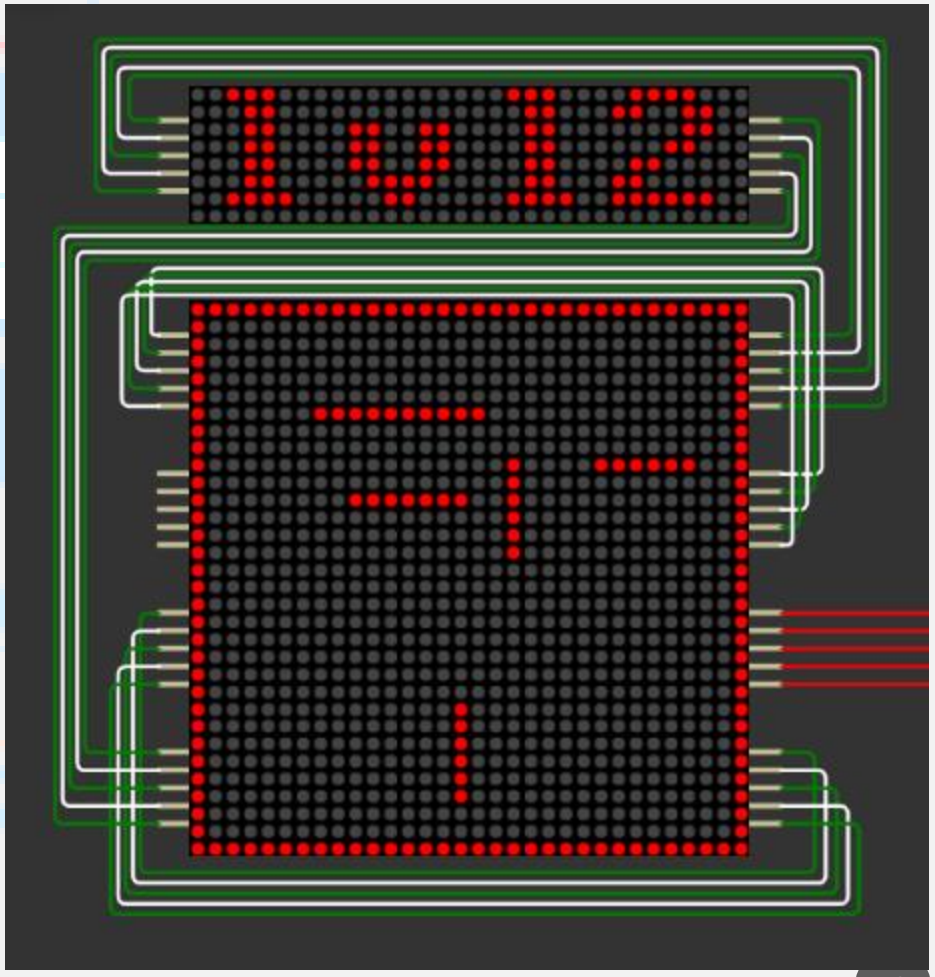
EASY MODE



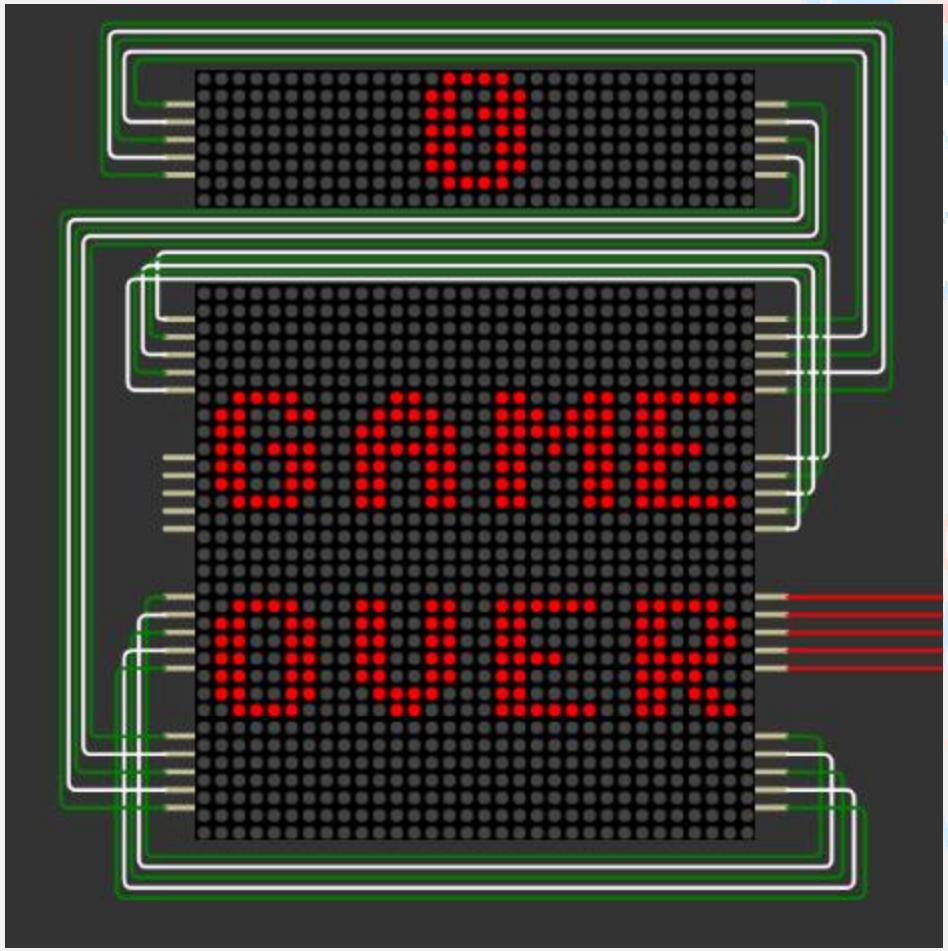
NORMAL MODE

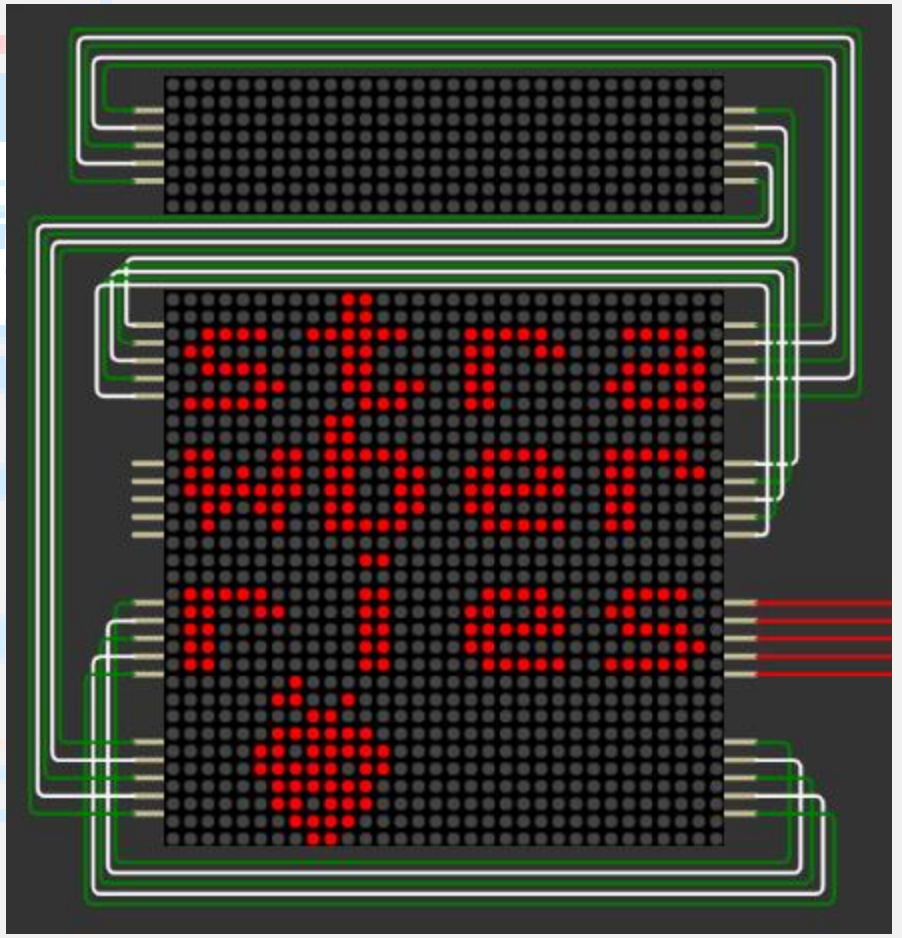
HARD MODE





GAME PHOTOS





CONCLUSION

CONCLUSION

- Prototype was used for testing trade-offs and consistency of product.
- As a result of the our project simulated in Wokwi environment we came to the conclusion of a physical product is plausible.
- Most of project time used for software part was advantage of using a virtual environment and developing the game.
- Due of simulation platform and python environment we can't solve problems related to speed of the snake.(real baud rate differs from simulated one)

References

- Arduino Simulator: Uno, Mega, ESP32, FastLED, LCD1602, Servo, Raspberry Pi Pico, Sensors. Designed for makers, by makers. https://wokwi.com/projects/328974406829212243
- https://github.com/csdexter/MAX7219
- https://github.com/coding-world/max7219
- https://github.com/adafruit/Adafruit_CircuitPython_MAX7219
- https://www.w3schools.com/python/

THANKS FOR LISTENING