THE UNIVERSITY of TULSA

Electrical and Computer Engineering

Rock, Paper, Scissors, Lizard, and Spock

Final Design Report

Noah Simpson
Will Ferguson
Zade Mahayni

December 14, 2022

Revision History

Date	Version	Author	Description
14DEC2022	1.0	Simpson N., Ferguson W., Mahayni Z.	Finished project

Customer Approval

Name	**Role**	**Signature of Approval**	**Approval Date**
Team Member	Engineer		15DEC2022
Team Member	Engineer		15DEC2022
Nathan Hutchins	Customer		15DEC2022

Contents

List of Figures

Listings

Abstract

In this project, we will be building a twist on the game of Rock Paper Scissors that adds two more options for an extra level of complexity. The goal of this project is to correctly simulate the game. In this game, two players choose an option for their turn. The five options are Lizard, Spock, Rock, Paper, and Scissors. Each option beats two other options. Scissors cuts Paper. Paper covers Rock. Rock crushes Lizard. Lizard poisons Spock. Spock smashes Scissors. Scissors decapitates Lizard. Lizard eats Paper. Paper disproves Spock. Spock vaporizes Rock. Rock crushes Scissors. There will be a timer displayed that gives each player five seconds to choose their option. Once a player chooses an option, the timer stops. If a player has not chosen an option after five seconds, their turn will be forfeited. The first to win three rounds will be the winner of the match. During the match, the win/loss record will be displayed on the LCD. When a player starts the program, they have the option to choose a multiplayer mode with another player, or a single-player mode where they play against an AI. The AI randomly selects one of the five options, and the outcome is determined the same way as in the multiplayer mode.

Nomenclature

ECO	Engineering Change Order
Nucleo	NUCLEO F411RE
AI	Artificial Intelligence
LCD	Liquid-Crystal Display

Chapter 1

Introduction

1.1 Introduction

Rock Paper Scissors is a classic game where two players use hand signals to make their selection based on what they think the opposition will choose. In the classic game, there are only three options to choose from. However, in this iteration of the game, we are increasing complexity by adding Lizard and Spock as options. Rock Paper Scissors Lizard Spock will allow the user to choose from five options, and each one of those options can beat two of the other options. This will allow for dynamic game play and deeper thought into making a selection.

1.2 Objective

Simulate a game of Rock Paper Scissors Lizard Spock. Each player has 5 seconds to choose an option, and after the timer runs out, it displays if you lose or win. The game is a best of 5. There will also be a single player option where you can play against an AI.

1.3 Customer

This game can be played by everyone over the age of 3.

1.4 Design Team

1.4.1 Engineering Co-Manager/Engineer/Programming Specialist

Noah Simpson is a computer engineering student at the University of Tulsa.

1.4.2 Engineering Co-Manager/Engineer/LCD Specialist

Will Ferguson is a computer engineering student at the University of Tulsa.

1.4.3 Engineering Co-Manager/Programmer/Keypad Specialist

Zade Mahayni is a computer engineering student at the University of Tulsa.

Chapter 2

Project Requirements

2.1 Hardware Requirements

2.1.1 NUCLEO F411RE

We shall store and run the game on the NUCLEO F411RE which is provided by The University of Tulsa. This board shall have the program uploaded to it, and the remaining hardware shall be attached to the board.

2.1.2 Keypad

The game shall use one or two keypads, but no more. The keypads will be used to make a selection. The keypads that shall be used are the 96BB2-056-R from Digi-Key Electronics. The single-player option shall require one keypad and the two-player option shall require two keypads, one for each player. The keypads have eight pins each, and each pin shall be connected to a specific port on the Nucleo.

2.1.3 LCD

The game shall use two LCD screens. The two LCDs that shall be used are the SparkFun 20x4 SerLCD. Each player will have a personal LCD screen that shall display the win/loss record, the countdown timer, and what their selection for the current round is.

2.2 Electrical Requirements

2.2.1 Electrical Power

The project shall run on 5 volts provided by the USB cable plugged into the Nucleo. That cable will be plugged into a computer or a wall outlet to deliver the power.

2.3 Software Requirements

2.3.1 STM32F411

The project shall be created using the STM32CubeIDE compiler.

2.4 Game Requirements

2.4.1 Must Resolve All Inputs Correctly

The game shall adhere to the rules of Rock Paper Scissors Lizard Spock.

- Scissors beats Paper and Lizard
- Paper beats Rock and Spock
- Rock beats Scissors and Lizard
- Lizard beats Spock and Paper
- Spock beats Rock and Scissors

2.4.2 Game Modes

- Single Player
  - The user plays against an AI that will randomly select an option.
- Multiplayer
  - The user plays against another user.

2.4.3 Timer

There shall be a five-second timer on the LCDs that terminates when a user selects an option. If no option is selected when the timer reaches zero, the user's turn will automatically be forfeited.

2.4.4 Win/Loss Record

There will be a win/loss record that is displayed on each LCD. In the event of a tie, the current round will be replayed. Once the winner of a round is determined, the record will update and will not include ties.

2.4.5 Victory/Defeat Screen

Once a player wins three times, the winner's LCD will display a victory screen, and the loser's LCD will display a defeat screen.

Chapter 3

Hardware Design

3.1 Mechanical Design

Both keypads are connected to the Nucleo. Both LCDs are connected to the Nucleo and share the same power which is connected to a breadboard from the Nucleo. See Figure 3.1 for the setup.

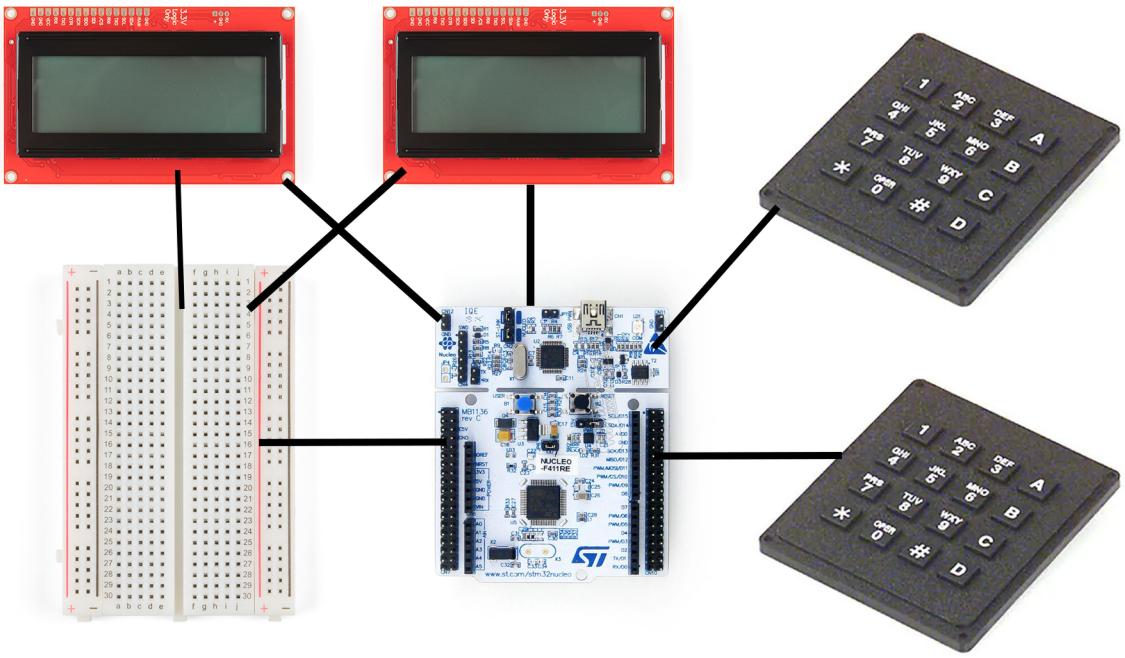

Figure 3.1: Hardware setup

3.2 Electrical System Design

The Nucleo receives power via USB from a source such as a laptop or power brick. The LCDs get 5V of power from the Nucleo by connecting to the breadboard.

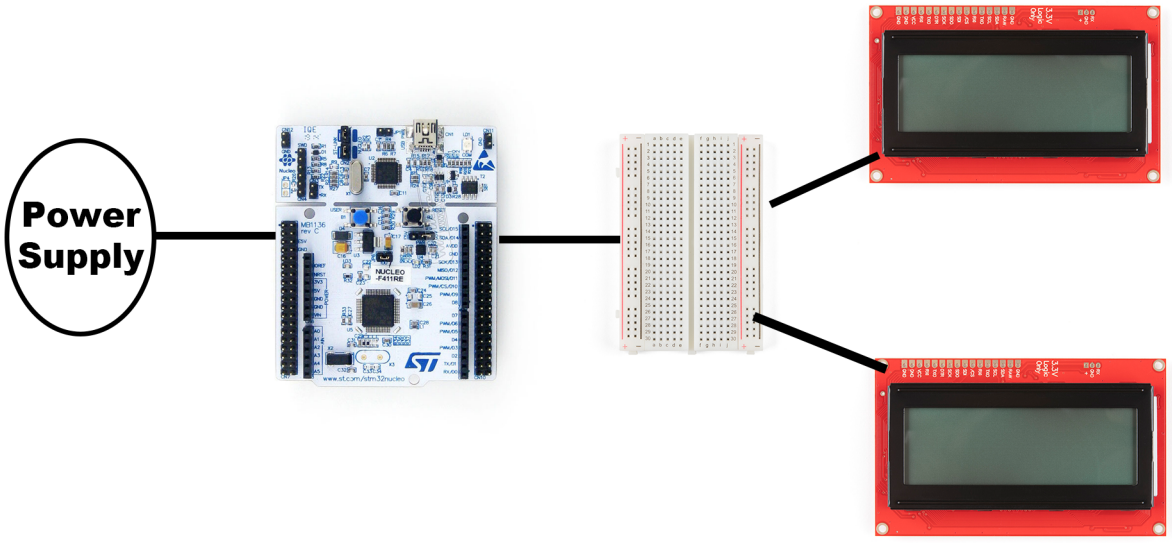

Figure 3.2: Power Overview

3.3 Electronic System Design

The LCDs have three wires that need to be connected: a power wire, a grounded wire, and a RX wire. The RX wire is connected to the TX port on the Nucleo. The keypads have eight pins each, with four of them being connected to the rows, and the other four pins being connected to the columns. These are connected to the Nucleo as shown in Figure 3.3 below.

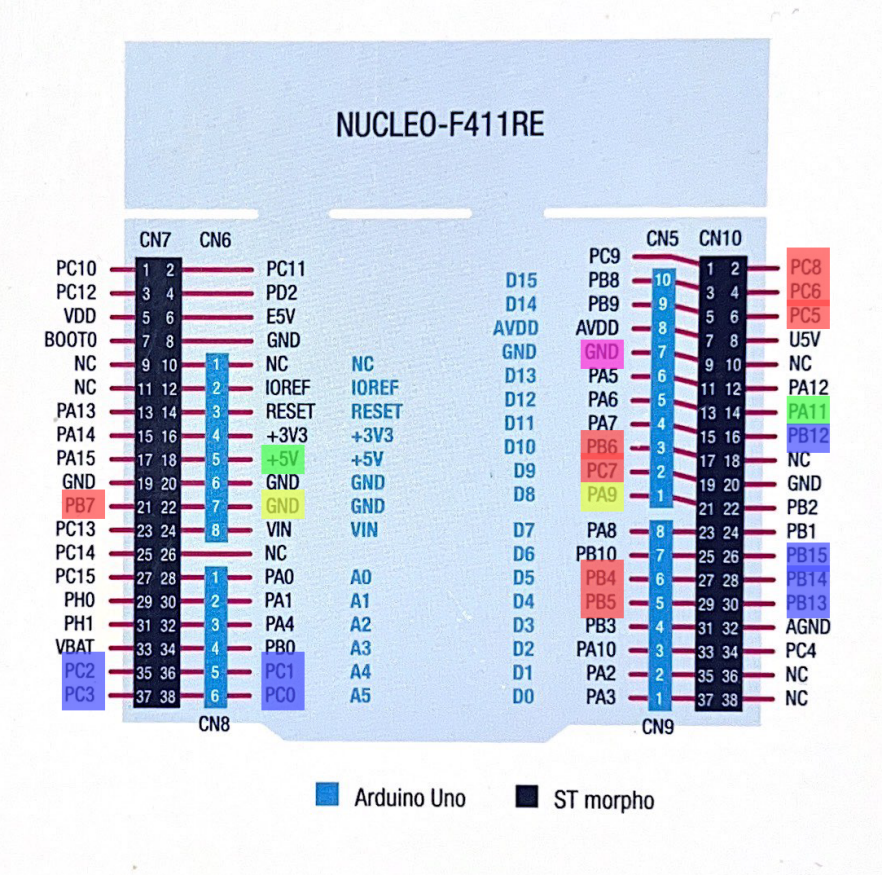

Figure 3.3: Nucleo Pinout

- **Yellow** - LCD 1
- **Pink** - LCD 2
- **Red** - Keypad 1
- **Blue** - Keypad 2
- **Green** - Breadboard

Chapter 4

Software Design

4.1 Introduction

We included several libraries in the main.c file as shown below.

```
#include "main.h"
#include "string.h"
#include "stdlib.h"
#include "stdbool.h"
#include "stdio.h"
#include "memory.h"
#include "stdint.h"
#include "stm32f4xx_hal_rcc.h"
#include "stm32f4xx_hal.h"
#include "stm32f4xx_hal_exti.h"
#include "stm32f4xx_hal_flash_ex.h"
#include "stm32f4xx_hal_flash.h"
#include "stm32f4xx_hal_gpio.h"
```

Listing 4.1: Libraries included

We also included the variables for the keypad in the main.c file. Since we were going to display many messages throughout the program, we created a variable for each message that would be displayed.

```
  char welcome_player_one[]="          Player One                    How many players?   (1
    or 2)";
  char welcome_player_two[]="          Player Two                    Wait for player one.";
  char player_one_wait_5[]="Game starts in 5    Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_wait_4[]="Game starts in 4    Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_wait_3[]="Game starts in 3    Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_wait_2[]="Game starts in 2    Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_wait_1[]="Game starts in 1    Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_choice_5[]="Make your choice!  5Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_choice_4[]="Make your choice!  4Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_choice_3[]="Make your choice!  3Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_choice_2[]="Make your choice!  2Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char player_one_choice_1[]="Make your choice!  1Rock(1) Paper(2)    Scissors(3)
    Lizard(4) Spock(5)";
  char please_wait_1[]="Please wait for     player 2";
  char please_wait_2[]="Player 1 is         selecting";
  char buffer[]="    Please Wait";
```

```
  char player1_choice_message[]="Player 1 chose:      ";
  char player_one_choice_final[]="                     ";
  char player2_choice_message[]="Player 2 chose:      ";
  char player_two_choice_final[]="                     ";
  char teststatement[]="hi mom";
  char winstatement[]="                          You Win!       ";
  char loststatement[]="                          DEFEATED! XD     ";
  char score[32]="                              0--0";
```

Listing 4.2: Possible Messages

We also created a function for each LCD that clears its screen as shown below.

```
void clearScreen1(void){
    uint8_t clearscreen1[] = {0x7C, 0x2D};
    HAL_UART_Transmit(&huart1, clearscreen1, sizeof(clearscreen1), HAL_MAX_DELAY);
}
void clearScreen2(void){
    uint8_t clearscreen2[] = {0x7C, 0x2D};
    HAL_UART_Transmit(&huart6, clearscreen2, sizeof(clearscreen2), HAL_MAX_DELAY);
}
```

Listing 4.3: Clear Screen Command

Figure 4.1 is the flowchart that gives a simple description of the process that the program runs through.

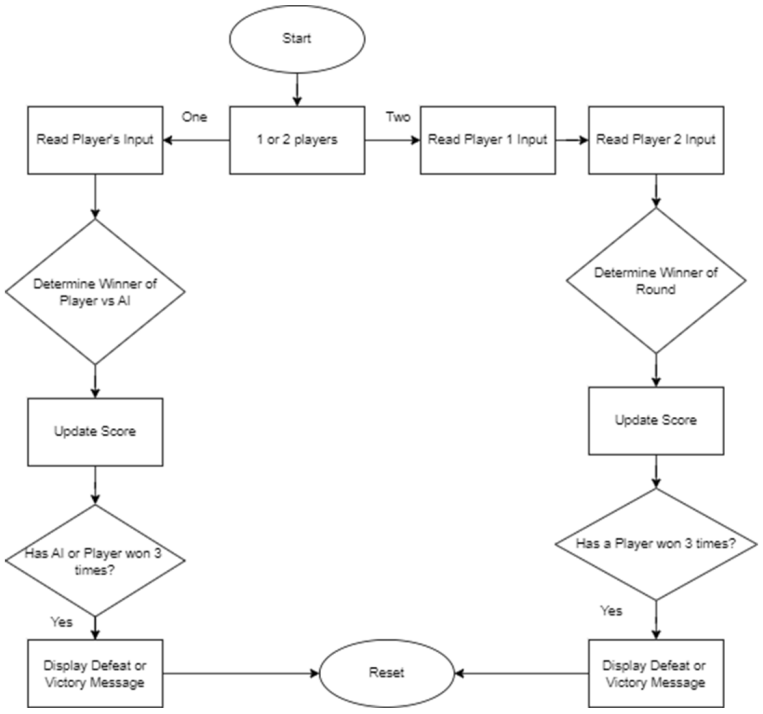

Figure 4.1: Small Flowchart

To make the keypad functional, we used the code from Dr. Hutchins' GitHub. We copied this into our interrupt file. We then created a second version of each variable. We then copied the original function from the first keypad, and changed all the variables to the second versions in order to make the second keypad functional without interrupting the first keypad.

```
void EXTI9_5_IRQHandler(void)
{
  /* USER CODE BEGIN EXTI9_5_IRQn 0 */
    key = 0;
    for(uint8_t c=0 ; c<4 ; c++){
        for(uint8_t i=0 ; i<4 ; i++){
            HAL_GPIO_WritePin((GPIO_TypeDef*)_KEYPAD_COLUMN_GPIO_PORT[i],
    _KEYPAD_COLUMN_GPIO_PIN[i], GPIO_PIN_SET);
        }
        HAL_GPIO_WritePin((GPIO_TypeDef*)_KEYPAD_COLUMN_GPIO_PORT[c],
    _KEYPAD_COLUMN_GPIO_PIN[c], GPIO_PIN_RESET);
        for(uint8_t r=0 ; r<4 ; r++){
            if(HAL_GPIO_ReadPin((GPIO_TypeDef*)_KEYPAD_ROW_GPIO_PORT[r],
    _KEYPAD_ROW_GPIO_PIN[r]) == GPIO_PIN_RESET){
                if(HAL_GPIO_ReadPin((GPIO_TypeDef*)_KEYPAD_ROW_GPIO_PORT[r],
    _KEYPAD_ROW_GPIO_PIN[r]) == GPIO_PIN_RESET){
                    key |= 1<<c;
                    key |= 1<<(r+8);
                    while(HAL_GPIO_ReadPin((GPIO_TypeDef*)_KEYPAD_ROW_GPIO_PORT[r],
    _KEYPAD_ROW_GPIO_PIN[r]) == GPIO_PIN_RESET){

                    }
                }
            }
        }
    }


    keyChar = KeyPadGetChar(key);
    key = 0;

    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_RESET);
```

Listing 4.4: Keypad interrupt code

The main block of code consists of many if-statements. It begins by displaying a welcome message and waiting for the player's decision on which mode they want to play. If the player selects the single player mode, the program jumps to the single player portion. In this block, the program reads the player's input and then goes to a function that generates a random number. This random number is read as the AI's "input." The program then jumps to a series of if-statements that decide who won the round. After the choices and scores are displayed, it goes to the next round and repeats the process. After either the player or the AI has won three rounds, a victory or defeat screen is displayed depending on who has reached 3 victories first. Then it goes back to the welcome screen.

```
      welcome:
      clearScreen1();
      clearScreen2();

      HAL_UART_Transmit(&huart1, welcome_player_one, sizeof(welcome_player_one)-1,
    HAL_MAX_DELAY);
      HAL_UART_Transmit(&huart6, welcome_player_two, sizeof(welcome_player_two)-1,
    HAL_MAX_DELAY);
      player1_score=0;
      player2_score=0;
      start:
      if (keyChar == 0){
          tim++;
          goto start;
      }
      else if (keyChar == '1'){
          clearScreen1();
          keyChar=0;
```

```
            goto one_player;
        }
        else if (keyChar == '2'){
                clearScreen1();
                clearScreen2();
                keyChar=0;
                keyChar2=0;
                goto two_player;
        }
        else if (keyChar != 0){
            keyChar=0;
            goto start;
        }
```

Listing 4.5: Welcome Code

```
if(player1_score < 3 && player2_score < 3){
      HAL_UART_Transmit(&huart1, player_one_wait_5, sizeof(player_one_wait_5)-1,
    HAL_MAX_DELAY);
      HAL_Delay(1000);
      clearScreen1();
      HAL_UART_Transmit(&huart1, player_one_wait_4, sizeof(player_one_wait_4)-1,
    HAL_MAX_DELAY);
      HAL_Delay(1000);
      clearScreen1();
```

Listing 4.6: Beginning of Player 1 Countdown

The code below shows a series of if statements nested within a single if statement. If player 1's keyChar is equal to 1, then that means that they have selected 'rock' as their choice. The nested if statements are the possible choices for the AI, and the winner's score will be updated based on if the AI's choice defeats 'rock' or not.

```
clearScreen1();
      if (keyChar == '1'){
          if (rand_number == 1){
              sprintf(score, "                              %d--%d", player1_score,
    player2_score);
              HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
              HAL_Delay(3000);
              keyChar = 0;
          }
          else if (rand_number ==2){
              player2_score++;
              sprintf(score, "                              %d--%d", player1_score,
    player2_score);
              HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
              HAL_Delay(3000);
              keyChar = 0;
          }
          else if (rand_number ==3){
              player1_score++;
                        sprintf(score, "                                %d--%d", player1_score
    , player2_score);
                        HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
                        HAL_Delay(3000);
                        keyChar = 0;
          }
          else if (rand_number ==4){
              player1_score++;
                        sprintf(score, "                                %d--%d", player1_score
    , player2_score);
                        HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
                        HAL_Delay(3000);
                        keyChar = 0;
          }
          else if (rand_number ==5){
              player2_score++;
```

```
                        sprintf(score, "                          %d--%d", player1_score
    , player2_score);
                        HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
                        HAL_Delay(3000);
                        keyChar = 0;
            }
        }
```

Listing 4.7: Example of logic for determining winner

If the player chooses the two player mode, the program jumps to reading the input of the first player. It then jumps to a block of code that reads the second player's input. This input from player 2 is defined as keyChar2. After both inputs have been submitted, it jumps to a block of code that decides who the winner is. It then displayed the choices of both players and then the updated score. Once a player has won three times, a defeat and victory screen are displayed.

```
if (keyChar2 == '1'){
                sprintf(player_two_choice_final, "Rock                ");
            }
            else if (keyChar2 == '2'){
                sprintf(player_two_choice_final, "Paper               ");
            }
            else if (keyChar2 == '3'){
                    sprintf(player_two_choice_final, "Scissors            ");
                }
            else if (keyChar2 == '4'){
                    sprintf(player_two_choice_final, "Lizard              ");
                }
            else if (keyChar2 == '5'){
                    sprintf(player_two_choice_final, "Spock               ");
                }
            else{
                keyChar2=0;
              sprintf(player_two_choice_final, "                    ");
               no_input_2 = 1;
            }
```

Listing 4.8: Assigning value from player 2 choice

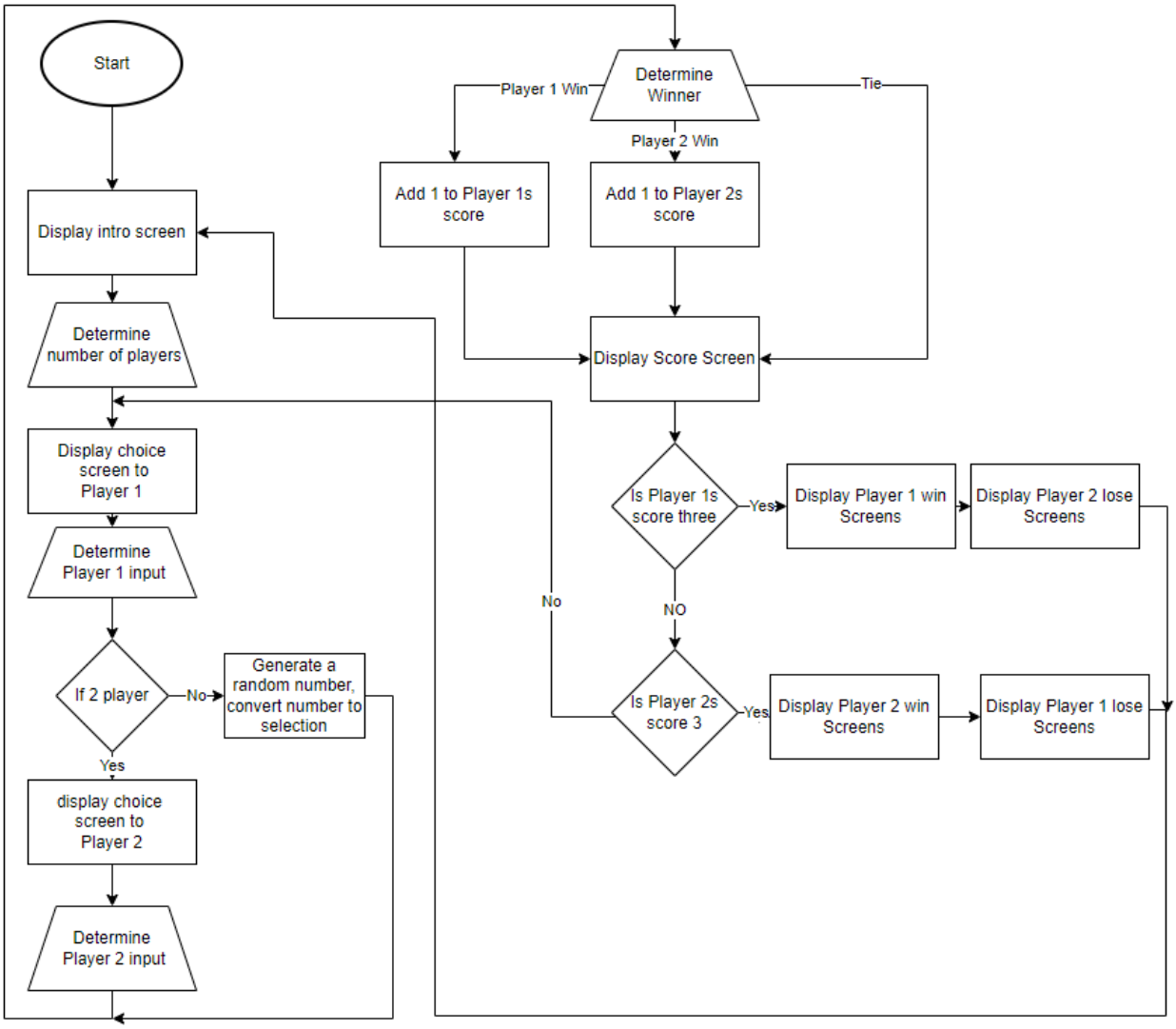

Figure 4.2: Large Flowchart

Chapter 5

Testing Requirements

5.1 Hardware Testing Requirements

5.1.1 The device shall recover from power loss

When the device is unplugged from a power source and then plugged back in, the game starts from the beginning.

5.1.2 Nucleo

This requirement was fulfilled by building the code onto our Nucleo and running it.

5.1.3 Keypad x2

This requirement was fulfilled by connecting the keypads to the Nucleo as shown in the Hardware Design section.

5.1.4 LCD x2

This requirement was fulfilled by connecting the LCD's to the Nucleo as shown earlier in the Hardware Design section.

5.2 Electrical Testing Requirements

5.2.1 Power Supply

This requirement was fulfilled by connecting the Nucleo to a laptop.

5.3 Software Testing Requirements

5.3.1 STM32CubeIDE

This requirement was fulfilled by using the STM32CubeIDE compiler to create our program.

5.4 Game Testing Requirements

5.4.1 Must Resolve All Inputs Correctly

This requirement was fulfilled by using a block of if-statements that determines the winner of a round depending on the inputs of the players. See Listing 4.8 for an example.

5.4.2 Game Modes

This requirement was fulfilled by prompting the player to decide which game mode to play on the welcome screen, and then going to the appropriate portion of code depending on what the user has chosen.

5.4.3 Timer

This requirement was fulfilled by displaying a 5 second timer on the top right of the LCD that counts down.

5.4.4 Win/Loss Record

This requirement was fulfilled by displaying the score at the end of each round.

5.4.5 Victory/Defeat Screen

This requirement was fulfilled by displaying a victory and defeat message once a player won three rounds.

Chapter 6

Users Guide

6.1 Introduction

This is a turn-based game of Rock Paper Scissors Lizard Spock. The first player chooses whether to play against on their own or against a second player. In both modes, the players have 5 seconds to choose an option. If no option is chosen after 5 seconds, it will count as forfeiting a round. However, if both players choose no option, the round will not be counted. The match ends when a player has won three rounds, and they will be shown a victory or defeat message depending on their result. The score is displayed after every round. Ties have no effect on the score.

6.2 Single-Player Mode

If the single player mode is chosen, the user will be put against an AI that randomly chooses an option every round.

6.3 Two-Player Mode

If the user chooses the two player mode, the game will be played in a turn-based process. The first player has five seconds to choose their option. Once an option has been chosen, the second player chooses their option. The options selected will not be displayed until both turns have passed. Make sure to hide your keypad while selecting so the second player does not cheat! It then updates the score depending on who won, and will repeat this process until someone has won three games.

6.4 Controls

6.4.1 Welcome Screen Controls

- 1 - Single-Player Mode
- 2 - Two-Player Mode

6.4.2 In-Game Controls

Pressing this number on the keypad when prompted on the screen will make that your choice.

- 1 - Rock
- 2 - Paper
- 3 - Scissors
- 4 - Lizard
- 5 - Spock

Chapter 7

Lessons Learned

In this project we learned several things such as how to use multiple devices with the Nucleo board as well as how to create a random number generator. The random number generator was done by having a variable named 'tim' that increases rapidly at the beginning of the game until the user selects an option. Once the option is selected, a seed is set by using the srand funtion. Then, a variable called 'rand_number' is set with the rand function. After each round of gameplay, the variable 'tim' is increased by one to change the seed so that the same number is not being constantly chosen.

```
1  if (keyChar == 0){
2          tim++;
3          goto start;
4      }
```

Listing 7.1: Seed set with 'tim'

```
1  results:
2      tim++;
3      srand(tim);
4      rand_number = (rand() % (upper-lower+1 ))+lower;
```

Listing 7.2: Random number chosen

Appendix A

Source Code

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2022 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "string.h"
#include "stdlib.h"
#include "stdbool.h"
#include "stdio.h"
#include "memory.h"
#include "stdint.h"
#include "stm32f4xx_hal_rcc.h"
#include "stm32f4xx_hal.h"
#include "stm32f4xx_hal_exti.h"
#include "stm32f4xx_hal_flash_ex.h"
#include "stm32f4xx_hal_flash.h"
#include "stm32f4xx_hal_gpio.h"
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

```

```
/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
UART_HandleTypeDef huart1;
UART_HandleTypeDef huart2;
UART_HandleTypeDef huart6;

/* USER CODE BEGIN PV */
uint16_t key = 0;
char keyChar = 0;
uint16_t key2 = 0;
char keyChar2 = 0;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART2_UART_Init(void);
static void MX_USART1_UART_Init(void);
static void MX_USART6_UART_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

//goto begin;
  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_USART2_UART_Init();
  MX_USART1_UART_Init();
  MX_USART6_UART_Init();
  /* USER CODE BEGIN 2 */
//  char FirstNumber[20];
//  char SecondNumber[20];
//  char tmpFirstNumber[20];
//  char tmpSecondNumber[20];
//  char limit[32];
  void clearScreen1(void);
```

```
  void clearScreen2(void);
  char sendMessage[32];

//  char notvalid[32];
//  for(int i = 0; i < 20; i++){
//    FirstNumber[i] = '\0';
//    SecondNumber[i] = '\0';
//    tmpFirstNumber[i] = '\0';
//    tmpSecondNumber[i] = '\0';
//  }
//  clearScreen();
//
//  HAL_UART_Transmit(&huart1, (unsigned char *)"Enter number 1:\r\n", 17, HAL_MAX_DELAY);
  begin:
  clearScreen1();

  char welcome_player_one[]="          Player One                    How many players?   (1 or 2)";
  char welcome_player_two[]="          Player Two                    Wait for player one.";
  char player_one_wait_5[]="Game starts in 5    Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_wait_4[]="Game starts in 4    Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_wait_3[]="Game starts in 3    Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_wait_2[]="Game starts in 2    Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_wait_1[]="Game starts in 1    Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_choice_5[]="Make your choice!  5Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_choice_4[]="Make your choice!  4Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_choice_3[]="Make your choice!  3Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_choice_2[]="Make your choice!  2Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char player_one_choice_1[]="Make your choice!  1Rock(1) Paper(2)    Scissors(3) Lizard(4) Spock(5)";
  char please_wait_1[]="Please wait for     player 2";
  char please_wait_2[]="Player 1 is         selecting";
  char buffer[]="    Please Wait";
  char player1_choice_message[]="Player 1 chose:     ";
  char player_one_choice_final[]="                    ";
  char player2_choice_message[]="Player 2 chose:     ";
  char player_two_choice_final[]="                    ";
  char teststatement[]="hi mom";
  char winstatement[]="                        You Win!      ";
  char loststatement[]="                        DEFEATED! XD    ";
  char score[32]="                           0--0";
  int game_counter = 0;
  int no_input_1 = 0;
  int no_input_2 = 0;
  int rand_number;
  int lower = 1;
  int upper = 5;
  int player1_score=0;
  int player2_score=0;
  int tim=0;
//goto welcome;
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {

          HAL_Delay(3000);
```

```
175
176       welcome:
177       clearScreen1();
178       clearScreen2();
179
180       HAL_UART_Transmit(&huart1, welcome_player_one, sizeof(welcome_player_one)-1,
      HAL_MAX_DELAY);
181       HAL_UART_Transmit(&huart6, welcome_player_two, sizeof(welcome_player_two)-1,
      HAL_MAX_DELAY);
182       player1_score=0;
183       player2_score=0;
184       start:
185       if (keyChar == 0){
186           tim++;
187           goto start;
188       }
189       else if (keyChar == '1'){
190           clearScreen1();
191           keyChar=0;
192           goto one_player;
193       }
194       else if (keyChar == '2'){
195               clearScreen1();
196               clearScreen2();
197               keyChar=0;
198               keyChar2=0;
199               goto two_player;
200       }
201       else if (keyChar != 0){
202           keyChar=0;
203           goto start;
204       }
205
206
207
208
209       one_player:
210
211       if(player1_score < 3 && player2_score < 3){
212       HAL_UART_Transmit(&huart1, player_one_wait_5, sizeof(player_one_wait_5)-1,
      HAL_MAX_DELAY);
213       HAL_Delay(1000);
214       clearScreen1();
215       HAL_UART_Transmit(&huart1, player_one_wait_4, sizeof(player_one_wait_4)-1,
      HAL_MAX_DELAY);
216       HAL_Delay(1000);
217       clearScreen1();
218       HAL_UART_Transmit(&huart1, player_one_wait_3, sizeof(player_one_wait_3)-1,
      HAL_MAX_DELAY);
219       HAL_Delay(1000);
220       clearScreen1();
221       HAL_UART_Transmit(&huart1, player_one_wait_2, sizeof(player_one_wait_2)-1,
      HAL_MAX_DELAY);
222       HAL_Delay(1000);
223       clearScreen1();
224       HAL_UART_Transmit(&huart1, player_one_wait_1, sizeof(player_one_wait_1)-1,
      HAL_MAX_DELAY);
225       HAL_Delay(1000);
226       clearScreen1();
227
228
229       HAL_UART_Transmit(&huart1, player_one_choice_5, sizeof(player_one_choice_5)-1,
      HAL_MAX_DELAY);
230       HAL_Delay(1000);
231       clearScreen1();
232       if(keyChar !=0){
233           goto pleasewait1;
234       }
```

```
235       HAL_UART_Transmit(&huart1, player_one_choice_4, sizeof(player_one_choice_4)-1,
    HAL_MAX_DELAY);
236       HAL_Delay(1000);
237       clearScreen1();
238       if(keyChar !=0){
239               goto pleasewait1;
240           }
241       HAL_UART_Transmit(&huart1, player_one_choice_3, sizeof(player_one_choice_3)-1,
    HAL_MAX_DELAY);
242       HAL_Delay(1000);
243       clearScreen1();
244       if(keyChar !=0){
245               goto pleasewait1;
246           }
247       HAL_UART_Transmit(&huart1, player_one_choice_2, sizeof(player_one_choice_2)-1,
    HAL_MAX_DELAY);
248       HAL_Delay(1000);
249       clearScreen1();
250       if(keyChar !=0){
251               goto pleasewait1;
252           }
253       HAL_UART_Transmit(&huart1, player_one_choice_1, sizeof(player_one_choice_1)-1,
    HAL_MAX_DELAY);
254       HAL_Delay(1000);
255       clearScreen1();
256       goto pleasewait1;
257
258
259
260       pleasewait1:
261       clearScreen1();
262       HAL_UART_Transmit(&huart1, please_wait_1, sizeof(please_wait_1)-1, HAL_MAX_DELAY);
263       if (keyChar == '1'){
264           sprintf(player_one_choice_final, "Rock                ");
265       }
266       else if (keyChar == '2'){
267           sprintf(player_one_choice_final, "Paper               ");
268       }
269       else if (keyChar == '3'){
270               sprintf(player_one_choice_final, "Scissors            ");
271           }
272       else if (keyChar == '4'){
273               sprintf(player_one_choice_final, "Lizard              ");
274           }
275       else if (keyChar == '5'){
276               sprintf(player_one_choice_final, "Spock               ");
277           }
278       else{
279           keyChar=0;
280           sprintf(player_one_choice_final, "                    ");
281           no_input_1 = 1;
282       }
283       HAL_Delay(5000);
284       clearScreen1();
285       HAL_Delay(100);
286       goto results;
287
288       results:
289       tim++;
290       srand(tim);
291       rand_number = (rand() % (upper-lower+1 ))+lower;
292 //rand_number = 3;
293
294       if (rand_number == 1){
295               sprintf(player_two_choice_final, "Rock                ");
296           }
297           else if (rand_number == 2){
298               sprintf(player_two_choice_final, "Paper               ");
```

```
299          }
300          else if (rand_number == 3){
301                  sprintf(player_two_choice_final, "Scissors            ");
302              }
303          else if (rand_number == 4){
304                  sprintf(player_two_choice_final, "Lizard              ");
305              }
306          else if (rand_number == 5){
307                  sprintf(player_two_choice_final, "Spock               ");
308              }
309          else{
310
311             no_input_2 = 1;
312          }
313
314
315      HAL_UART_Transmit(&huart1, player1_choice_message, sizeof(player1_choice_message)-1,
    HAL_MAX_DELAY);
316      HAL_UART_Transmit(&huart1, player_one_choice_final, sizeof(player_one_choice_final)-1,
     HAL_MAX_DELAY);
317      HAL_UART_Transmit(&huart1, player2_choice_message, sizeof(player2_choice_message)-1,
    HAL_MAX_DELAY);
318          HAL_UART_Transmit(&huart1, player_two_choice_final, sizeof(player_two_choice_final
    )-1, HAL_MAX_DELAY);
319
320
321      HAL_Delay(5000);
322
323
324 clearScreen1();
325      if (keyChar == '1'){
326          if (rand_number == 1){
327              sprintf(score, "                          %d--%d", player1_score,
    player2_score);
328              HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
329              HAL_Delay(3000);
330              keyChar = 0;
331          }
332          else if (rand_number ==2){
333              player2_score++;
334              sprintf(score, "                          %d--%d", player1_score,
    player2_score);
335              HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
336              HAL_Delay(3000);
337              keyChar = 0;
338          }
339          else if (rand_number ==3){
340              player1_score++;
341                          sprintf(score, "                          %d--%d", player1_score
    , player2_score);
342                          HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
343                          HAL_Delay(3000);
344                          keyChar = 0;
345          }
346          else if (rand_number ==4){
347              player1_score++;
348                          sprintf(score, "                          %d--%d", player1_score
    , player2_score);
349                          HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
350                          HAL_Delay(3000);
351                          keyChar = 0;
352          }
353          else if (rand_number ==5){
354              player2_score++;
355                          sprintf(score, "                          %d--%d", player1_score
    , player2_score);
356                          HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
357                          HAL_Delay(3000);
```

```
358                         keyChar = 0;
359         }
360       }
361
362       else if (keyChar == '2'){
363               if (rand_number == 1){
364               player1_score++;
365               sprintf(score, "                                %d--%d", player1_score,
    player2_score);
366               HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
367               HAL_Delay(3000);
368               keyChar = 0;
369               }
370               else if (rand_number ==2){
371                sprintf(score, "                                %d--%d", player1_score,
    player2_score);
372                HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
373                HAL_Delay(3000);
374                keyChar = 0;
375               }
376               else if (rand_number ==3){
377                 player2_score++;
378                             sprintf(score, "                                %d--%d",
    player1_score, player2_score);
379                             HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
380                             HAL_Delay(3000);
381                             keyChar = 0;
382                   }
383               else if (rand_number ==4){
384                 player2_score++;
385                             sprintf(score, "                                %d--%d",
    player1_score, player2_score);
386                             HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
387                             HAL_Delay(3000);
388                             keyChar = 0;
389                   }
390               else if (rand_number ==5){
391                 player1_score++;
392                 sprintf(score, "                                %d--%d", player1_score,
    player2_score);
393               HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
394               HAL_Delay(3000);
395               keyChar = 0;
396                     }
397       }
398     else if (keyChar == '3'){
399                  if (rand_number == 1){
400                     player2_score++;
401                                 sprintf(score, "                                %d--%d",
    player1_score, player2_score);
402                                 HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
403                                 HAL_Delay(3000);
404                                 keyChar = 0;
405                  }
406                  else if (rand_number ==2){
407                     player1_score++;
408                                 sprintf(score, "                                %d--%d",
    player1_score, player2_score);
409                                 HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
410                                 HAL_Delay(3000);
411                                 keyChar = 0;
412                  }
413                  else if (rand_number ==3){
414                    sprintf(score, "                                %d--%d", player1_score,
```

```
player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                        }
                else if (rand_number ==4){
                   player1_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                        }
                else if (rand_number ==5){
                   player2_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                        }
          }
    else if (keyChar == '4'){
                    if (rand_number == 1){
                   player2_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                    }
                    else if (rand_number ==2){
                   player1_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                    }
                    else if (rand_number ==3){
                   player2_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                           }
                    else if (rand_number ==4){
                  sprintf(score, "                              %d--%d", player1_score,
player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                           }
                    else if (rand_number ==5){
                   player1_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
```

```
                                keyChar = 0;
                            }
                }
      else if (keyChar == '5'){
                    if (rand_number == 1){
                    player1_score++;
                                sprintf(score, "                          %d--%d",
    player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                    }
                    else if (rand_number ==2){
                    player2_score++;
                                sprintf(score, "                          %d--%d",
    player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                    }
                    else if (rand_number ==3){
                    player1_score++;
                                sprintf(score, "                          %d--%d",
    player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                            }
                    else if (rand_number ==4){
                    player2_score++;
                                sprintf(score, "                          %d--%d",
    player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                            }
                    else if (rand_number ==5){
                   sprintf(score, "                          %d--%d", player1_score,
    player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                            }
                }
      else{
          player2_score++;
          sprintf(score, "                          %d--%d", player1_score, player2_score)
    ;
          HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
          HAL_Delay(3000);
          keyChar = 0;
      }

      clearScreen1();
      HAL_Delay(100);

game_counter++;



      goto one_player;

  }
```

```
524
525
526 if(player1_score > player2_score){
527     HAL_UART_Transmit(&huart1, winstatement, sizeof(winstatement), HAL_MAX_DELAY);
528     HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
529     HAL_Delay(7000);
530 }
531 else{
532     HAL_UART_Transmit(&huart1, loststatement, sizeof(loststatement), HAL_MAX_DELAY);
533     HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
534     HAL_Delay(7000);
535 }
536 goto welcome;
537
538
539
540
541
542
543
544       two_player:
545
546       if(player1_score < 3 && player2_score < 3){
547           HAL_UART_Transmit(&huart6, please_wait_2, sizeof(please_wait_2)-1, HAL_MAX_DELAY);
548           HAL_UART_Transmit(&huart1, player_one_wait_5, sizeof(player_one_wait_5)-1,
    HAL_MAX_DELAY);
549           HAL_Delay(1000);
550           clearScreen1();
551           HAL_UART_Transmit(&huart1, player_one_wait_4, sizeof(player_one_wait_4)-1,
    HAL_MAX_DELAY);
552           HAL_Delay(1000);
553           clearScreen1();
554           HAL_UART_Transmit(&huart1, player_one_wait_3, sizeof(player_one_wait_3)-1,
    HAL_MAX_DELAY);
555           HAL_Delay(1000);
556           clearScreen1();
557           HAL_UART_Transmit(&huart1, player_one_wait_2, sizeof(player_one_wait_2)-1,
    HAL_MAX_DELAY);
558           HAL_Delay(1000);
559           clearScreen1();
560           HAL_UART_Transmit(&huart1, player_one_wait_1, sizeof(player_one_wait_1)-1,
    HAL_MAX_DELAY);
561           HAL_Delay(1000);
562           clearScreen1();
563
564
565           HAL_UART_Transmit(&huart1, player_one_choice_5, sizeof(player_one_choice_5)-1,
    HAL_MAX_DELAY);
566           HAL_Delay(1000);
567           clearScreen1();
568           if(keyChar !=0){
569               goto pleasewait2;
570           }
571           HAL_UART_Transmit(&huart1, player_one_choice_4, sizeof(player_one_choice_4)-1,
    HAL_MAX_DELAY);
572           HAL_Delay(1000);
573           clearScreen1();
574           if(keyChar !=0){
575                   goto pleasewait2;
576               }
577           HAL_UART_Transmit(&huart1, player_one_choice_3, sizeof(player_one_choice_3)-1,
    HAL_MAX_DELAY);
578           HAL_Delay(1000);
579           clearScreen1();
580           if(keyChar !=0){
581                   goto pleasewait2;
582               }
583           HAL_UART_Transmit(&huart1, player_one_choice_2, sizeof(player_one_choice_2)-1,
```

```
    HAL_MAX_DELAY);
584         HAL_Delay(1000);
585         clearScreen1();
586         if(keyChar !=0){
587                 goto pleasewait2;
588             }
589         HAL_UART_Transmit(&huart1, player_one_choice_1, sizeof(player_one_choice_1)-1,
    HAL_MAX_DELAY);
590         HAL_Delay(1000);
591         clearScreen1();
592         goto pleasewait2;
593
594
595
596         pleasewait2:
597         clearScreen1();
598         HAL_UART_Transmit(&huart1, please_wait_1, sizeof(please_wait_1)-1, HAL_MAX_DELAY);
599         if (keyChar == '1'){
600             sprintf(player_one_choice_final, "Rock                ");
601         }
602         else if (keyChar == '2'){
603             sprintf(player_one_choice_final, "Paper               ");
604         }
605         else if (keyChar == '3'){
606                 sprintf(player_one_choice_final, "Scissors            ");
607             }
608         else if (keyChar == '4'){
609                 sprintf(player_one_choice_final, "Lizard              ");
610             }
611         else if (keyChar == '5'){
612                 sprintf(player_one_choice_final, "Spock               ");
613             }
614         else{
615             keyChar=0;
616             sprintf(player_one_choice_final, "                    ");
617             no_input_1 = 1;
618         }
619         clearScreen2();
620         HAL_Delay(100);
621        HAL_UART_Transmit(&huart6, player_one_wait_5, sizeof(player_one_wait_5)-1,
    HAL_MAX_DELAY);
622                 HAL_Delay(1000);
623                 clearScreen2();
624                 HAL_UART_Transmit(&huart6, player_one_wait_4, sizeof(player_one_wait_4)-1,
     HAL_MAX_DELAY);
625                 HAL_Delay(1000);
626                 clearScreen2();
627                 HAL_UART_Transmit(&huart6, player_one_wait_3, sizeof(player_one_wait_3)-1,
     HAL_MAX_DELAY);
628                 HAL_Delay(1000);
629                 clearScreen2();
630                 HAL_UART_Transmit(&huart6, player_one_wait_2, sizeof(player_one_wait_2)-1,
     HAL_MAX_DELAY);
631                 HAL_Delay(1000);
632                 clearScreen2();
633                 HAL_UART_Transmit(&huart6, player_one_wait_1, sizeof(player_one_wait_1)-1,
     HAL_MAX_DELAY);
634                 HAL_Delay(1000);
635                 clearScreen2();
636
637
638                 HAL_UART_Transmit(&huart6, player_one_choice_5, sizeof(player_one_choice_5
    )-1, HAL_MAX_DELAY);
639                 HAL_Delay(1000);
640                 clearScreen2();
641                 if(keyChar2 !=0){
642                     goto pleasewait3;
643                 }
```

```
644                 HAL_UART_Transmit(&huart6, player_one_choice_4, sizeof(player_one_choice_4
    )-1, HAL_MAX_DELAY);
645                 HAL_Delay(1000);
646                 clearScreen2();
647                 if(keyChar2 !=0){
648                         goto pleasewait3;
649                     }
650                 HAL_UART_Transmit(&huart6, player_one_choice_3, sizeof(player_one_choice_3
    )-1, HAL_MAX_DELAY);
651                 HAL_Delay(1000);
652                 clearScreen2();
653                 if(keyChar2 !=0){
654                         goto pleasewait3;
655                     }
656                 HAL_UART_Transmit(&huart6, player_one_choice_2, sizeof(player_one_choice_2
    )-1, HAL_MAX_DELAY);
657                 HAL_Delay(1000);
658                 clearScreen2();
659                 if(keyChar2 !=0){
660                         goto pleasewait3;
661                     }
662                 HAL_UART_Transmit(&huart6, player_one_choice_1, sizeof(player_one_choice_1
    )-1, HAL_MAX_DELAY);
663                 HAL_Delay(1000);
664                 clearScreen2();
665                 goto pleasewait3;
666 
667                 pleasewait3:
668                 clearScreen1();
669                 clearScreen2();
670                 HAL_Delay(100);
671                 HAL_UART_Transmit(&huart1, buffer, sizeof(buffer)-1, HAL_MAX_DELAY);
672                 HAL_UART_Transmit(&huart6, buffer, sizeof(buffer)-1, HAL_MAX_DELAY);
673                 HAL_Delay(2000);
674 
675       clearScreen2();
676       clearScreen1();
677       HAL_Delay(100);
678       goto results2;
679 
680       results2:
681 
682       if (keyChar2 == '1'){
683               sprintf(player_two_choice_final, "Rock                ");
684           }
685           else if (keyChar2 == '2'){
686               sprintf(player_two_choice_final, "Paper               ");
687           }
688           else if (keyChar2 == '3'){
689                   sprintf(player_two_choice_final, "Scissors            ");
690               }
691           else if (keyChar2 == '4'){
692                   sprintf(player_two_choice_final, "Lizard              ");
693               }
694           else if (keyChar2 == '5'){
695                   sprintf(player_two_choice_final, "Spock               ");
696               }
697           else{
698               keyChar2=0;
699             sprintf(player_two_choice_final, "                    ");
700              no_input_2 = 1;
701           }
702 
703 
704       HAL_UART_Transmit(&huart1, player1_choice_message, sizeof(player1_choice_message)
    -1, HAL_MAX_DELAY);
705       HAL_UART_Transmit(&huart1, player_one_choice_final, sizeof(player_one_choice_final
    )-1, HAL_MAX_DELAY);
```

```
706         HAL_UART_Transmit(&huart1, player2_choice_message, sizeof(player2_choice_message)
    -1, HAL_MAX_DELAY);
707             HAL_UART_Transmit(&huart1, player_two_choice_final, sizeof(
    player_two_choice_final)-1, HAL_MAX_DELAY);
708           HAL_UART_Transmit(&huart6, player1_choice_message, sizeof(player1_choice_message
    )-1, HAL_MAX_DELAY);
709                     HAL_UART_Transmit(&huart6, player_one_choice_final, sizeof(
    player_one_choice_final)-1, HAL_MAX_DELAY);
710                     HAL_UART_Transmit(&huart6, player2_choice_message, sizeof(
    player2_choice_message)-1, HAL_MAX_DELAY);
711                         HAL_UART_Transmit(&huart6, player_two_choice_final, sizeof(
    player_two_choice_final)-1, HAL_MAX_DELAY);
712
713
714         HAL_Delay(5000);
715
716
717      clearScreen1();
718      clearScreen2();
719         if (keyChar == '1'){
720             if (keyChar2 == '1'){
721                sprintf(score, "                              %d--%d", player1_score,
    player2_score);
722                 HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
723                HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
724                 HAL_Delay(3000);
725                 keyChar = 0;
726                keyChar2 = 0;
727             }
728             else if (keyChar2 == '2'){
729                 player2_score++;
730                 sprintf(score, "                               %d--%d", player1_score,
    player2_score);
731                 HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
732                HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
733                 HAL_Delay(3000);
734                 keyChar = 0;
735                 keyChar2 = 0;
736             }
737             else if (keyChar2 == '3'){
738                 player1_score++;
739                             sprintf(score, "                                %d--%d",
    player1_score, player2_score);
740                             HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY
    );
741                           HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
742                             HAL_Delay(3000);
743                             keyChar = 0;
744                           keyChar2 = 0;
745             }
746             else if (keyChar2 == '4'){
747                 player1_score++;
748                             sprintf(score, "                                %d--%d",
    player1_score, player2_score);
749                             HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY
    );
750                           HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
751                             HAL_Delay(3000);
752                             keyChar = 0;
753                           keyChar2 = 0;
754             }
755             else if (keyChar2 == '5'){
756                 player2_score++;
757                             sprintf(score, "                                %d--%d",
    player1_score, player2_score);
758                             HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY
    );
759                           HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
```

```
                            HAL_Delay(3000);
                            keyChar = 0;
                          keyChar2 = 0;

          }
          else{
              player1_score++;
             sprintf(score, "                              %d--%d", player1_score,
player2_score);
                                          HAL_UART_Transmit(&huart1, score, sizeof(score
), HAL_MAX_DELAY);
                                        HAL_UART_Transmit(&huart6, score, sizeof(score),
 HAL_MAX_DELAY);
                                          HAL_Delay(3000);
                                          keyChar = 0;
                                        keyChar2 = 0;
          }
      }

      else if (keyChar == '2'){
              if (keyChar2 == '1'){
              player1_score++;
              sprintf(score, "                              %d--%d", player1_score,
player2_score);
              HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
            HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
              HAL_Delay(3000);
              keyChar = 0;
            keyChar2 = 0;
              }
              else if (keyChar2 == '2'){
               sprintf(score, "                              %d--%d", player1_score,
player2_score);
               HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
            HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
               HAL_Delay(3000);
               keyChar = 0;
            keyChar2 = 0;
              }
              else if (keyChar2 == '3'){
                player2_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                            HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                            keyChar2 = 0;
                      }
              else if (keyChar2 == '4'){
                player2_score++;
                                sprintf(score, "                              %d--%d",
player1_score, player2_score);
                                HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                            HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                HAL_Delay(3000);
                                keyChar = 0;
                            keyChar2 = 0;
                      }
              else if (keyChar2 == '5'){
                player1_score++;
                sprintf(score, "                              %d--%d", player1_score,
player2_score);
                HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
```

```
                    HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
                    HAL_Delay(3000);
                    keyChar = 0;
                    keyChar2 = 0;
                          }
                  else{
                                player1_score++;
                               sprintf(score, "                              %d--%d",
    player1_score, player2_score);
                                                                HAL_UART_Transmit(&huart1,
    score, sizeof(score), HAL_MAX_DELAY);
                                                               HAL_UART_Transmit(&huart6, score
    , sizeof(score), HAL_MAX_DELAY);
                                                                HAL_Delay(3000);
                                                                keyChar = 0;
                                                               keyChar2 = 0;
                            }
              }
         else if (keyChar == '3'){
                    if (keyChar2 == '1'){
                      player2_score++;
                                        sprintf(score, "                              %d--%d
    ", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                       HAL_UART_Transmit(&huart6, score, sizeof(score),
    HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                       keyChar2 = 0;
                    }
                    else if (keyChar2 == '2'){
                      player1_score++;
                                        sprintf(score, "                              %d--%d
    ", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                       HAL_UART_Transmit(&huart6, score, sizeof(score),
    HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                       keyChar2 = 0;
                    }
                    else if (keyChar2 == '3'){
                      sprintf(score, "                                %d--%d", player1_score,
    player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                       HAL_UART_Transmit(&huart6, score, sizeof(score),
    HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                       keyChar2 = 0;
                            }
                    else if (keyChar2 == '4'){
                      player1_score++;
                                        sprintf(score, "                              %d--%d
    ", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
    HAL_MAX_DELAY);
                                       HAL_UART_Transmit(&huart6, score, sizeof(score),
    HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                       keyChar2 = 0;
                            }
                    else if (keyChar2 == '5'){
                      player2_score++;
```

```
                                        sprintf(score, "                          %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                              }
                  else{
                                               player1_score++;
                                             sprintf(score, "
 %d--%d", player1_score, player2_score);
                                              HAL_UART_Transmit(&huart1, score, sizeof(
score), HAL_MAX_DELAY);
                                                HAL_UART_Transmit(&huart6, score, sizeof
(score), HAL_MAX_DELAY);
                                          HAL_Delay(3000);
                                                    keyChar = 0;
                                          keyChar2 = 0;
                                            }
                  }
          else if (keyChar == '4'){
                            if (keyChar2 == '1'){
                        player2_score++;
                                        sprintf(score, "                          %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                            }
                            else if (keyChar2 == '2'){
                        player1_score++;
                                        sprintf(score, "                          %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                            }
                            else if (keyChar2 == '3'){
                        player2_score++;
                                        sprintf(score, "                          %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                                  }
                            else if (keyChar2 == '4'){
                        sprintf(score, "                            %d--%d", player1_score,
player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
```

```
                                        keyChar = 0;
                                      keyChar2 = 0;
                                }
                          else if (keyChar2 == '5'){
                        player1_score++;
                                        sprintf(score, "                                %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                                }
                     else{
                                                    player1_score++;
                                                  sprintf(score, "
      %d--%d", player1_score, player2_score);
                             HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY
);
                           HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
                                     HAL_Delay(3000);
                                     keyChar = 0;
                                       keyChar2 = 0;
                                                }
                    }
       else if (keyChar == '5'){
                          if (keyChar2 == '1'){
                        player2_score++;
                                        sprintf(score, "                                %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                          }
                          else if (keyChar2 == '2'){
                        player1_score++;
                                        sprintf(score, "                                %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                          }
                          else if (keyChar2 == '3'){
                        player1_score++;
                                        sprintf(score, "                                %d--%d
", player1_score, player2_score);
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                        HAL_Delay(3000);
                                        keyChar = 0;
                                      keyChar2 = 0;
                                }
                          else if (keyChar2 == '4'){
                        player2_score++;
                                        sprintf(score, "                                %d--%d
", player1_score, player2_score);
```

```
                                        HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                    HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_Delay(3000);
                                      keyChar = 0;
                                    keyChar2 = 0;
                                }
                          else if (keyChar2 == '5'){
                        sprintf(score, "                              %d--%d", player1_score,
player2_score);
                                      HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                                    HAL_UART_Transmit(&huart6, score, sizeof(score),
HAL_MAX_DELAY);
                                      HAL_Delay(3000);
                                      keyChar = 0;
                                    keyChar2 = 0;
                                }
                    else{
                                                player1_score++;
                                               sprintf(score, "
      %d--%d", player1_score, player2_score);
                                  HAL_UART_Transmit(&huart1, score, sizeof(score),
HAL_MAX_DELAY);
                  HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
                   HAL_Delay(3000);
                      keyChar = 0;
                      keyChar2 = 0;
                                           }
                    }
        else{
            if (keyChar2 == '1' || keyChar2 == '2' || keyChar2 == '3' || keyChar2 == '4'
|| keyChar2 == '5'){
                player2_score++;

            sprintf(score, "                              %d--%d", player1_score,
player2_score);
            HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
           HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
            HAL_Delay(3000);
            keyChar = 0;
            keyChar2=0;
            }
            else{
               sprintf(score, "                              %d--%d", player1_score,
player2_score);
               HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
              HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
               HAL_Delay(3000);
               keyChar = 0;
               keyChar2=0;
            }
        }

        clearScreen1();
        clearScreen2();
        HAL_Delay(100);

     game_counter++;



        goto two_player;

      }


```

```
    if(player1_score > player2_score){
       HAL_UART_Transmit(&huart1, winstatement, sizeof(winstatement)-1, HAL_MAX_DELAY);
       HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
       HAL_UART_Transmit(&huart6, loststatement, sizeof(loststatement), HAL_MAX_DELAY);
                HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
       HAL_Delay(7000);
    }
    else{
       HAL_UART_Transmit(&huart1, loststatement, sizeof(loststatement), HAL_MAX_DELAY);
        HAL_UART_Transmit(&huart1, score, sizeof(score), HAL_MAX_DELAY);
        HAL_UART_Transmit(&huart6, winstatement, sizeof(winstatement)-1, HAL_MAX_DELAY);
               HAL_UART_Transmit(&huart6, score, sizeof(score), HAL_MAX_DELAY);
        HAL_Delay(7000);
    }
    goto welcome;




    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
  RCC_OscInitStruct.PLL.PLLM = 16;
  RCC_OscInitStruct.PLL.PLLN = 336;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV4;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
```

```
  {
    Error_Handler();
  }
}

/**
  * @brief USART1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_USART1_UART_Init(void)
{

  /* USER CODE BEGIN USART1_Init 0 */

  /* USER CODE END USART1_Init 0 */

  /* USER CODE BEGIN USART1_Init 1 */

  /* USER CODE END USART1_Init 1 */
  huart1.Instance = USART1;
  huart1.Init.BaudRate = 9600;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_UART_Init(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN USART1_Init 2 */

  /* USER CODE END USART1_Init 2 */

}

/**
  * @brief USART2 Initialization Function
  * @param None
  * @retval None
  */
static void MX_USART2_UART_Init(void)
{

  /* USER CODE BEGIN USART2_Init 0 */

  /* USER CODE END USART2_Init 0 */

  /* USER CODE BEGIN USART2_Init 1 */

  /* USER CODE END USART2_Init 1 */
  huart2.Instance = USART2;
  huart2.Init.BaudRate = 115200;
  huart2.Init.WordLength = UART_WORDLENGTH_8B;
  huart2.Init.StopBits = UART_STOPBITS_1;
  huart2.Init.Parity = UART_PARITY_NONE;
  huart2.Init.Mode = UART_MODE_TX_RX;
  huart2.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart2.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_UART_Init(&huart2) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN USART2_Init 2 */

  /* USER CODE END USART2_Init 2 */
```

```
1166
1167 }
1168
1169 /**
1170   * @brief USART6 Initialization Function
1171   * @param None
1172   * @retval None
1173   */
1174 static void MX_USART6_UART_Init(void)
1175 {
1176
1177   /* USER CODE BEGIN USART6_Init 0 */
1178
1179   /* USER CODE END USART6_Init 0 */
1180
1181   /* USER CODE BEGIN USART6_Init 1 */
1182
1183   /* USER CODE END USART6_Init 1 */
1184   huart6.Instance = USART6;
1185   huart6.Init.BaudRate = 9600;
1186   huart6.Init.WordLength = UART_WORDLENGTH_8B;
1187   huart6.Init.StopBits = UART_STOPBITS_1;
1188   huart6.Init.Parity = UART_PARITY_NONE;
1189   huart6.Init.Mode = UART_MODE_TX_RX;
1190   huart6.Init.HwFlowCtl = UART_HWCONTROL_NONE;
1191   huart6.Init.OverSampling = UART_OVERSAMPLING_16;
1192   if (HAL_UART_Init(&huart6) != HAL_OK)
1193   {
1194     Error_Handler();
1195   }
1196   /* USER CODE BEGIN USART6_Init 2 */
1197
1198   /* USER CODE END USART6_Init 2 */
1199
1200 }
1201
1202 /**
1203   * @brief GPIO Initialization Function
1204   * @param None
1205   * @retval None
1206   */
1207 static void MX_GPIO_Init(void)
1208 {
1209   GPIO_InitTypeDef GPIO_InitStruct = {0};
1210
1211   /* GPIO Ports Clock Enable */
1212   __HAL_RCC_GPIOC_CLK_ENABLE();
1213   __HAL_RCC_GPIOH_CLK_ENABLE();
1214   __HAL_RCC_GPIOA_CLK_ENABLE();
1215   __HAL_RCC_GPIOB_CLK_ENABLE();
1216
1217   /*Configure GPIO pin Output Level */
1218   HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, GPIO_PIN_RESET);
1219
1220   /*Configure GPIO pin Output Level */
1221   HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_RESET);
1222
1223   /*Configure GPIO pin Output Level */
1224   HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_RESET);
1225
1226   /*Configure GPIO pins : PC0 PC1 PC2 PC3 */
1227   GPIO_InitStruct.Pin = GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3;
1228   GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
1229   GPIO_InitStruct.Pull = GPIO_PULLDOWN;
1230   GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
1231   HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
1232
1233   /*Configure GPIO pin : LD2_Pin */
```

```
  GPIO_InitStruct.Pin = LD2_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LD2_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pins : PC5 PC6 PC7 PC8 */
  GPIO_InitStruct.Pin = GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7|GPIO_PIN_8;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

  /*Configure GPIO pins : PB12 PB13 PB14 PB15 */
  GPIO_InitStruct.Pin = GPIO_PIN_12|GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

  /*Configure GPIO pins : PB4 PB5 PB6 PB7 */
  GPIO_InitStruct.Pin = GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

  /* EXTI interrupt init*/
  HAL_NVIC_SetPriority(EXTI9_5_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(EXTI9_5_IRQn);

  HAL_NVIC_SetPriority(EXTI15_10_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(EXTI15_10_IRQn);

}

/* USER CODE BEGIN 4 */
void clearScreen1(void){
    uint8_t clearscreen1[] = {0x7C, 0x2D};
    HAL_UART_Transmit(&huart1, clearscreen1, sizeof(clearscreen1), HAL_MAX_DELAY);
}
void clearScreen2(void){
    uint8_t clearscreen2[] = {0x7C, 0x2D};
    HAL_UART_Transmit(&huart6, clearscreen2, sizeof(clearscreen2), HAL_MAX_DELAY);
}

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
```

```
1302   */
1303 void assert_failed(uint8_t *file, uint32_t line)
1304 {
1305   /* USER CODE BEGIN 6 */
1306   /* User can add his own implementation to report the file name and line number,
1307      ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
1308   /* USER CODE END 6 */
1309 }
1310 #endif /* USE_FULL_ASSERT */
```

Listing A.1: Source Code

Appendix B

Bill of Materials

B.1 Cost breakdown

NUCLEO F411RE	$12.74
LCD Screen (x2) (@ $29.95/ea)	$59.90
Keypad (x2) (@ $35.46/ea)	$70.92
Breadboard	$4.20
TOTAL	**$141.76**

Appendix C

Rating

This game is rated G for everyone 3 and up.