REAL-TIME EMBEDDED SYSTEM
EXPERIMENT NO. 1

REAL-TIME SCORE BOARD FOR A BASKETBALL GAME

Name: NAVARRO, ROD GERYK C.

Course/Section: CPE161P-4/C1

Group No.: N/A

Date of Performance: 12/10/2024
Date of Submission: 12/16/2024

CYREL O. MANLISES, PH.D.
Instructor

DISCUSSION

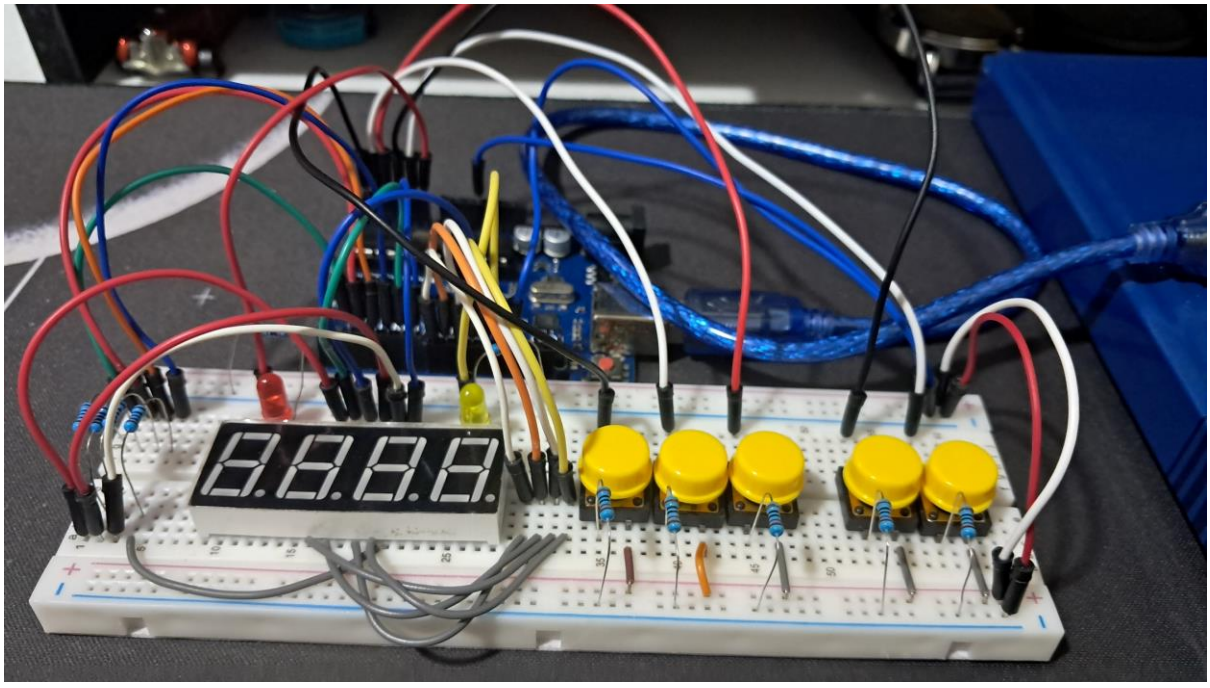

Figure 1: *The connection of the 4-digit SSD, buttons, and LEDs.*

I conducted the first part of the experiment during our scheduled class time. For the experiment, I used a 4-digit seven-segment display (SSD), buttons, and LEDs. The 4-digit SSD served as the scoreboard for two teams, with a maximum of two digits per team.

To start, I built the circuit using three buttons and a voltage divider made of 220-ohm resistors connected to one analog input. Later, I modified the setup to use five buttons: one for 3 points, one for 2 points, one for 1 point, one for switching teams, and one for decreasing the score. I removed the voltage divider from the circuit and decided to use the buttons for manually scoring points for the teams.

In addition, I also added two LEDs as indicators to show which team is currently being scored. Each team can score up to 99 points because the 4-digit SSD limits the maximum score to two digits per team.

Navarro_EXP1_RT.ino

```
1  const int sPins[7] = {6, 7, 8 ,9, 10, 11, 12};
2  const int dPins[4] = {2, 3, 4, 5};
3
4  const int switch1Pin = A5;
5  const int switch2Pin = A4;
6  const int switch3Pin = A3;
7  const int switch4Pin = A2;
8  const int switch5Pin = A1;
9
10 const int teamALED = A0;
11 const int teamBLED = 13;
12
13 int teamA = 0;
14 int teamB = 0;
15
16 bool adjustScore = true;
17 bool incrementMode = true;
18
19 unsigned long lastSwitch1PressTime = 0;
20 unsigned long lastSwitch2PressTime = 0;
21 unsigned long lastSwitch3PressTime = 0;
22 unsigned long lastSwitch4PressTime = 0;
23 unsigned long lastSwitch5PressTime = 0;
24
25 bool lastSwitch1State = HIGH;
26 bool lastSwitch2State = HIGH;
27 bool lastSwitch3State = HIGH;
28 bool lastSwitch4State = HIGH;
29 bool lastSwitch5State = HIGH;
30
31 byte digitCodes[10] = {
32   B00111111,
33   B00000110,
```

Figure 2: The initialization of variables and components.

The code begins by including the necessary variables for the 4-digit seven segment display, buttons, and LEDs. I also create the variable for team A and team B.

Navarro_EXP1_RT.ino

```
  B01101111,
};

void setup(){
  for (int i = 0; i < 7; i++){
    pinMode(sPins[i], OUTPUT);
  }

  for(int i = 0; i < 4; i++){
    pinMode(dPins[i], OUTPUT);
    digitalWrite(dPins[i], HIGH);
  }

  pinMode(switch1Pin, INPUT);
  pinMode(switch2Pin, INPUT);
  pinMode(switch3Pin, INPUT);
  pinMode(switch4Pin, INPUT);
  pinMode(switch5Pin, INPUT);

  pinMode(teamALED, OUTPUT);
  pinMode(teamBLED, OUTPUT);


  displayScore(teamA, teamB);


}

void loop(){

bool switch1State = digitalRead(switch1Pin);
bool switch2State = digitalRead(switch2Pin);
bool switch3State = digitalRead(switch3Pin);
```

Figure 3: The setup of the components.

So, after the initialization of the code is where I set the void setup (), and void loop (). In the void setup() is where I set all the components.

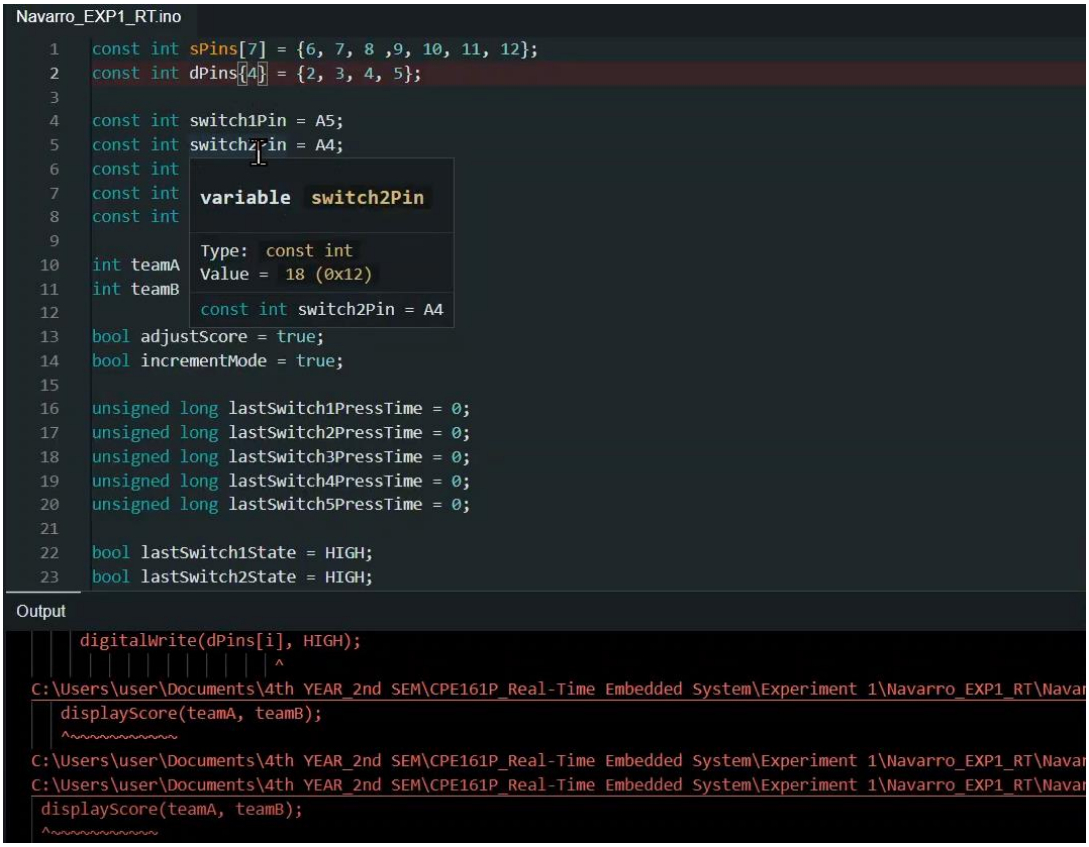

Figure 4: *Fixing the error in code.*

When I first tested my code, I encountered an error, but I quickly fixed it. The problem was caused by a missing semicolon and a missing function. I resolved it by adding the semicolon after the sPins variable and creating a function for displayScore. This wasn't the only error I faced while working on the code, but most of the issues were similar, and I was able to fix them right away.

Navarro_EXP1_RT.ino

```
67  }
68
69  void loop(){
70
71  bool switch1State = digitalRead(switch1Pin);
72  bool switch2State = digitalRead(switch2Pin);
73  bool switch3State = digitalRead(switch3Pin);
74  bool switch4State = digitalRead(switch4Pin);
75  bool switch5State = digitalRead(switch5Pin);
76
77  if(switch1State != lastSwitch1State){
78  if(switch1State != LOW){
79    if(incrementMode){
80      if(adjustScore){
81        incrementAThree();
82      }
83      else{
84        incrementBThree();
85      }
86    }
87  }
88
89  lastSwitch1State = switch1State;
90
91  }
92
93  if(switch2State != lastSwitch2State){
94  if(switch2State != LOW){
95    if(incrementMode){
96      if(adjustScore){
97        incrementATwo();
98      }
99      else{
100       incrementBTwo();
```

Figure 5: Display outputs for the digits and segments of teams, and function of each buttons as well as the LEDs.

In the loop() function, I set up all the conditions for the five buttons. The first button (on the left) adds 3 points to the score, the second button adds 2 points, and the third button adds 1 point. There is also a button to switch between teams, so the user can choose which team's score to increase or decrease. The maximum score for each team is 99. I also created specific functions to handle the button actions and the display.

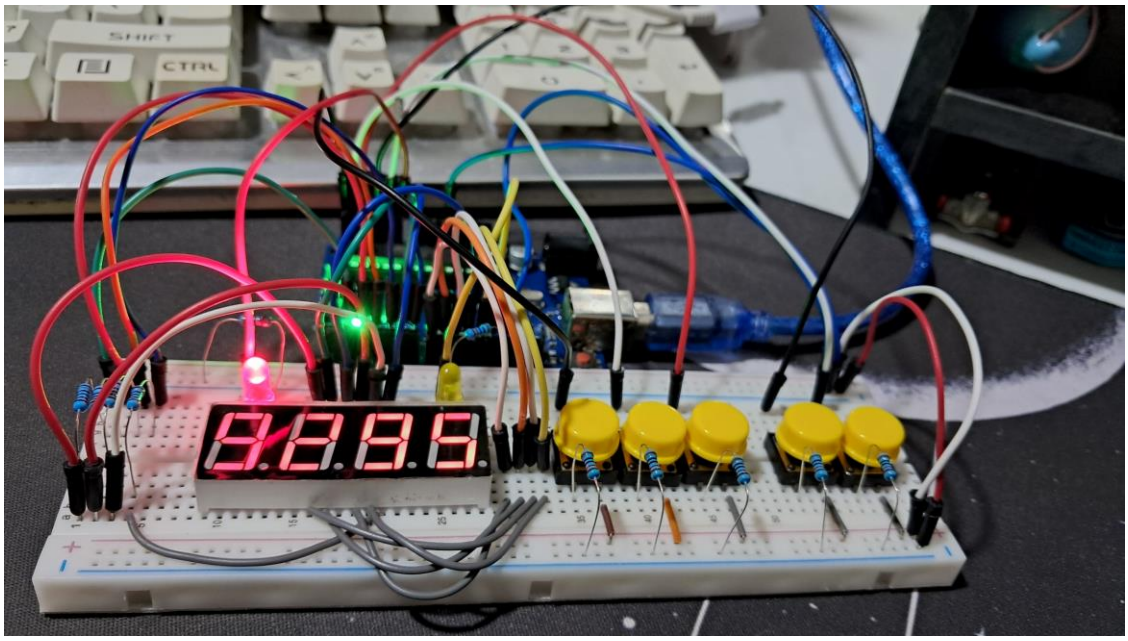

Figure 6: Working Prototype.

This is the final working prototype for this experiment that successfully meets all the requirements.

CONCLUSION

In this experiment I, I successfully built a working prototype of a scoreboard system using a 4-digit seven-segment display (SSD), buttons, and LEDs. The system was designed to keep track of scores for two teams, with each team able to score up to 99 points. Initially, I created the circuit with three buttons and a voltage divider, but later I modified it to use five buttons for more precise scoring functionality, including adding points, switching teams, and decreasing scores. The addition of two LEDs provided clear indicators for the active team being scored.

During the coding process, I encountered several errors, such as missing semicolons and functions, but I resolved them by debugging step-by-step. These challenges helped me improve my problem-solving skills and better understand how to manage code and circuit components. The final prototype met all the requirements, functioning as intended with all buttons, LEDs, and the SSD displaying scores accurately. This experiment taught me the importance of careful circuit design, proper coding, and iterative testing to achieve a successful outcome.