ECE 1895 Junior Design Fundamentals

Department of Electrical and Computer Engineering

University of Pittsburgh, Pittsburgh, PA, 15260

Project 2 Preliminary Proposal

Instructor:
Dr. Dickerson , dickerson@pitt.edu

Teaching Assistants:

Karey Stone , kjs165@pitt.edu

Madison Hodgson , madisonhodgson@pitt.edu

Wesley Saville , wms20@pitt.edu

Matthew Belding , mtb60@pitt.edu

Authors:

Alex Sleigher , aos15@pitt.edu

Dillon Hepler , dih41@pitt.edu

Isaiah Johnston , irj8@pitt.edu

Introduction

For our “Bop-It” project, we chose to implement a driving game called “Drive-It”, taking inspiration from the typical control scheme of F1 vehicles. Our design starts with a dashboard cluster inspired assembly for our components. The user will use a combination of at least three of the following inputs: a steering wheel, paddle shifter, horn, acceleration pedal, brake pedal, and an ignition switch.

To streamline our design process, the work will be split amongst the members of the group as follows:

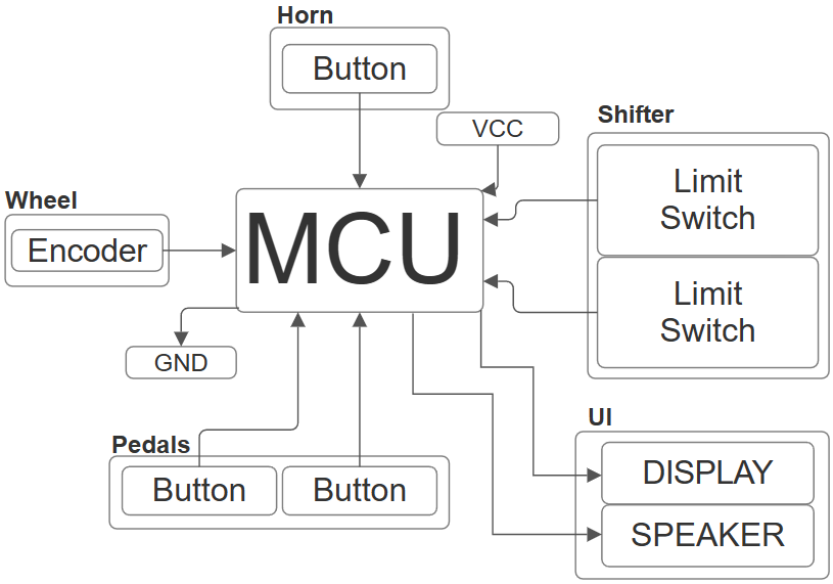
Dillon Hepler Software Design and Implementation
Isaiah Johnston Hardware and Enclosure Design
Alex Sleigher Project Lead; Project Design, Organization, and Compilation

Description

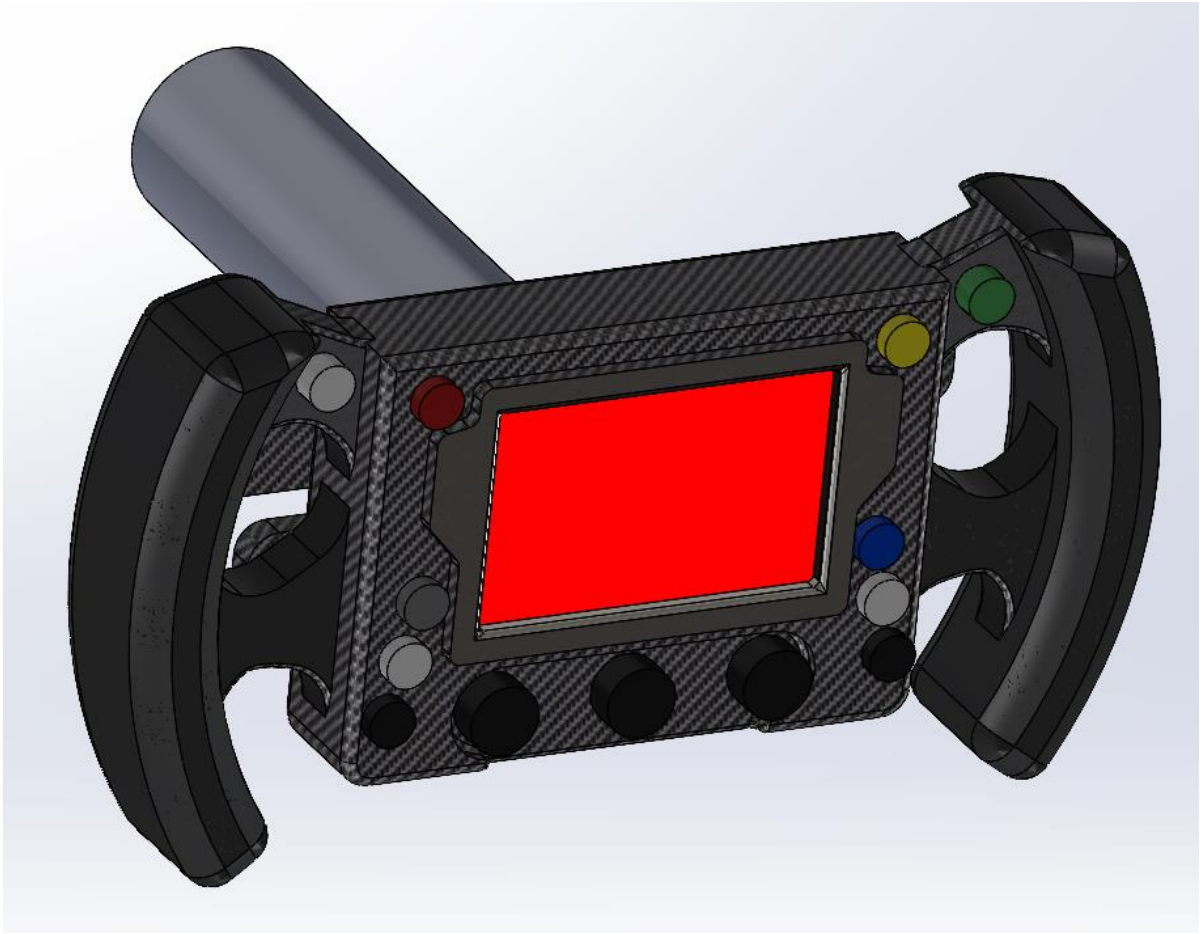
The ignition will be used to turn the toy on and off. This can be implemented by having a key be inserted into a slot and turned clockwise to activate a limit-switch which turns the device on and starts the game, and inversely operated to turn the device off. The device will command the user to perform one of multiple different actions. Starting with steering, the toy will command the used to “steer-it”. The user must then move the steering wheel within the accepted time frame to earn a point. This will be implemented using an encoder to measure the turning of the wheel. The horn will simply be controlled with a button, where the user must press it within a pre-determined time to earn a point and activate a speaker to simulate a car horn. The other forms of user input will be to accelerate and decelerate using gas and break pedals to earn points. The pedals will utilize angular positioning to determine the acceleration force. For the shifter, when prompted the user must activate the appropriate paddle shifter to earn points; these will be constructed with limit switches. If the user fails to follow any of the directions prompted, a negative consequence will occur, specifically the end of the game.

With each successful input, the user will be given a point and the time allowed to complete subsequent action will decrease. The game will conclude at 99 points, with a final time frame of 0.55 seconds. With each successful attempt, the time frame for task completion will decrease by 0.025 seconds, making the game difficult-yet possible to complete. There will also be a display to show the current score, along with other useful information.

Block Diagram



Conceptual Design



CREDIT: F1 Steering Wheel, Ibrahim Bakar, grabcad.com/library/formula-1-steering-wheel-1

Software Pseudocode

```
//include necessary libraries
//Define pins for inputs and outputs
// Define LED pin for score indication or maybe noise
// Define initial time to complete action in milliseconds
#define initial_time 5000
// Define percentage reduction factor for time after each successful action
#define time_reduction 0.97

// Function prototypes
void initialize();
void checkButtons();
void startGame();
void performAction();
void displayScore();
void gameOver();

// Global variables
volatile uint8_t score = 0;
volatile uint16_t timeToCompleteAction = intial_time;

int main() {
   initialize();
   while (1) {
      checkButtons();
   }
   return 0;
}
void initialize() {
   // Set input/output pins
Horn=Pin1;
Button=Pin2;
Brake=Pin3;
Steer=Pin4;
LED=Pin5;
 // Enable pull-up resistors for input pins
}

void checkButtons() {
if button == on, start game;
   }
}

void startGame() {
   score = 0;
   timeToCompleteAction = initial_time;

   while (1) {
      performAction();
   }
```

```
}

void performAction() {
   uint8_t randomAction = rand() % 3; // Generate a random number between 0 and 2

   new_delay(timeToCompleteAction);
            score++;
            timeToCompleteAction *=time_reduction;
         } else {
            gameOver();
         }
         break;
      case 1:
         if horn pressed:
            score++;
            timeToCompleteAction *=time_reduction;
         } else {
            gameOver();
         }
         break;
      case 2:
        if brake used:
            score++;
            timeToCompleteAction *=time_reduction;
         } else {
            gameOver();
         }
         break;

      case 3:
        if steer used:
            score++;
            timeToCompleteAction *=time_reduction;
         } else {
            gameOver();
         }
         break;

   }
   displayScore();
}

void displayScore() {
   //Implement score counting (Perhaps LEDS counting up in binary or a visual LED matrix counting up)
}

void gameOver() {
if wrong action pressed or run out of time:
   // Display final score
   displayScore();
```

```
    // make it maybe reset with a button pressed.
    while (1) {
        // Do nothing, game over state
    }
}
```