Abdullah Abdulqader

Professor Carl Michal

PHY319

April 6, 2017

Color

Abstract:

This paper reports the design, building process, and the difficulties encountered in the making of the project Color.

Introduction:

An arcade game is an entertainment machine typically installed in public places such as amusement parks. Many arcade games are electro-mechanical, meaning that players use mechanical devices to play an electronic game. The aim of this project was to build an electro-mechanical arcade game. This report discusses the game's concept, design, and the software.

Game Concept:

Color is a competitive game played by two players. Each player is given an LED strip containing 30 lit LEDs and is able to control it. Their objective is to race by turning off all 30 LEDs from the strip's top to its bottom, the first to meet this objective wins. Each lit LED can be red, yellow, blue, or green. Each player is given a set of buttons with the same four colors to control their strip. To turn off the top lit LED a player has to

click the button with the same color as the top lit LED. For example, if the top lit LED was blue, then a player would have to click on the blue button to turn it off.

Apparatus and Game Design:

The MSP430G2553 launchpad, WS2811 LED strips, and colored momentary switch buttons are the main equipment of this project. The WS2811 RGB LED strips are addressable, meaning it includes an IC chip in between each and every LED, making it possible to control each LED individually. Runs on 5V, about 18W per meter, it costs 35 CAD for a meter length with 60 LEDs. This strip is controlled by the MSP430 using the Serial Peripheral Interface (SPI) with Slave In Master Out (SIMO) signal. Three jumper wires were soldered to the +5V, Digital Input (Din), GND lines on the strip, then connected the Din to P1.7 (P1.7 is the only pin on the launchpad that is can communicate with SIMO signal) as shown in figure 1. The timing requirements for the WS2812 LEDs are very tight. As such, this requires a relatively high speed SPI clock (this uses ~5.3 MHz) and an even higher CPU clock speed.

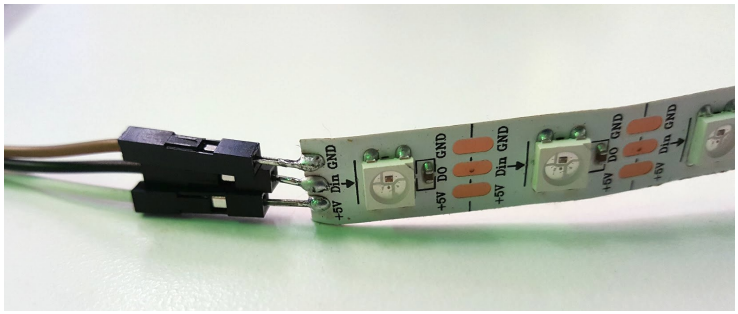

Figure 1: soldered jumper wires soldered to the LED strip lines.

I made controllers by mounting the colored buttons on small hand sized boxes, as shown in figure 2, so that players can easily control the strips. The pins P1.0 - P.3 on the launchpad are connected to the controllers.

Figure 2: controllers

To start the game, both players have to push the start button (mounted on the breadboard) connected to P1.4. For that to work, I had to make both launchpads communicate with each other: P1.5 was used to send out a signal to the other launchpad, and P1.6 was used to receive a signal from the other launchpad. P1.5 and P1.6 are used to communicate tell/know if a player is ready to start the game and if the player has won or lost the game.

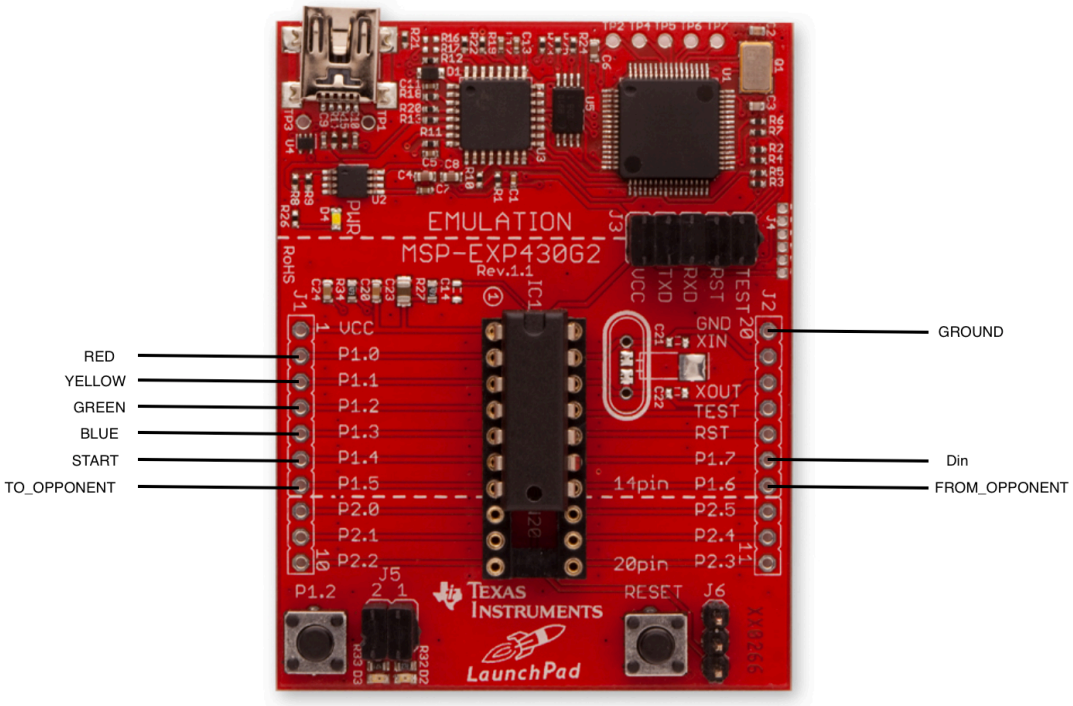

Figure 3: summary of all the connections to the launchpad.

To generate a random sequence of colored LEDS each time the players start the game, I have used the timer in the MSP430. When the MSP430 is connected to power, the timer would count from 0 - 255 repeatedly, when the player clicks the start button the timer stops and its value is then used in a function that will generate a random number:

```
char random(char x) {
  char number = ((x % 11) + (x % 3)) % 5;
  if (number == 4) {
    return random(x % 13);
  }
  return number;
}
```

The output of this function is not perfectly random, but it is suffice the needs of this project.

When a player presses a button that does not match with the color of the top lit LED, a penalty of approximately 2 seconds takes place on that player's strip where the strip will turn red and the player will not be able to race his opponent. Then, the strip return back to its state after the each penalty and the player can continue playing.

Once a player has finished the race, the winner's strip will display rainbow colors and the loser's will display a red color. To replay, players have to press the reset button on the launch pads.

Discussion:

Although most of this project has functioned as expected, it can be improved by the following:

- Using interrupts instead of polling: In the program I wrote, I used while loops to check for user input instead of interrupts. This is a waste of time and energy for the CPU.
- Using a latch (flip-flop) instead of a second launchpad: Although P1.7 is the only pin that can communicate a SIMO signal, a latch could have been used and connected to P1.7 to control both strips from a single launchpad.
- Measure the period of the input signals from the controllers: The input signals from the controllers were fluctuating and were not fully reliable. This could be been resolved by measuring how long each input signal is, then neglecting any small un-steady signals and keep the ones that are as long as a click takes.

Conclusion:

Although the golden age of arcade games have passed a long time ago, this project accomplished its goal of entertaining players with a nostalgic theme. Watching my friends play Color was a joyful experience for them and for myself.

References:

- Microprocessor family slau144j MSP430x2xx:
http://www.ti.com/lit/ug/slau144j/slau144j.pdf

- LaunchPad Experimenter board user guide (slau318):
http://www.ti.com/lit/ug/slau318g/slau318g.pdf

- MSP430G2553 Data sheet (slas735) :
http://www.ti.com/lit/ds/symlink/msp430g2553.pdf

- WS2812 Library:
https://github.com/mjmeli/MSP430-NeoPixel-WS2812-Library

Appendix:

The full step-by-step software development for this project can be viewed on Github: https://github.com/AbdullahAbdulqader/COLOR-ARCADE