Keshav Dasu

Project Report

High Level Description:

The game I created is called "Lock Pick." To play the player has three attempts to pick a lock successfully. The lock pins will be displayed on the top portion of the LCD screen as '|' characters and the bottom portion of the screen will show the current position of the players lock pick. When the player hits a button the current '|' character will either change to a "^" (symbolizing success) or the player will lose a life. If the player hits all the correct pins they will advance to the next level. In the earlier levels a clear distinctive sound is played, unlike the default sound played while scrolling, when the player scrolls over a loose pin. As the player progress the range the player can turn the potentiometer shrinks and the sound variation between the scroll sound and sound clue converge closer to the same frequency. The levels are indefinite you could keep playing for as long as you choose.

The main set of differences between my implemented version of the game and the original proposal are as following. I chose to rename my game, scrap the RGB component, removed the option to reset game, and to not return a life after a completing a level. The changes had no effect on my proposed "build upons". I decided the significance of the RGB part was unnecessary and returning a life back seemed to make the game to easy. However, all my proposed "build upons" are implemented in the game. They are as following; randomized auto generated levels, sound while scrolling as well as other sound effects, and if you scroll to fast you will lose a life.

Testing and Bug Report:

For testing I would use PORTC to display any variables whose behavior I found questionable. I created and integer to hexadecimal function and used that in conjunction with PORTC. Bugs that were found a fixed include, failure to go to next level, failure to auto generate a level and add a new pin pick, failure for scroll sound to play, and failure for menu and game over songs to play. There are no known bugs to me in my current version.

User Guide:

When you initially start the game it is best for you to turn the potentiometer to the upright position. To start the game and continue to next level press the button. To play the game use the potentiometer to scroll and the button to bump the "loose" pin.

Technologies and Components Used:

1. AVR Studio 5.1
2. Piezo Buzzer
3. LCD screen
4. Potentiometer
5. LED's
6. Button
7. ATmega1284 chip

YouTube Link:

https://www.youtube.com/watch?v=hmeTxPcyZ60

Source Files:

Kdasu001_Labproject.c: Contains the main loop as well as the state machines which are in accordance to the PES structured programming techniques.

Utilities.h: Contains all the useful functions given to us in lab that assist with communication to the hardware. (e.g. ADC_init(), set_PWM(), TimerOn(), GetBit(), etc..)

Helper_functions.h: Contains functions that assist the game logic as well as to debug the game.

Io.c/io.h: Given to us to use from ilearn for this class. Assists with communicating and initializing the LCD screen.