Homework 5:
777 For 7s Report

Submitted: December 13, 2017

Sabbir Ahmed

1 Description

This project required implementation of a slot machine simulation on the AVR Dragon.

To lure the user into playing the game, they are greeted by a welcoming "HELLO" screen on the LCD until the push button is pressed.

The player, who starts with 10 credits/earnings, will be playing a slot machine game where only A through F are represented. The slot machine has only three slots, so only words of three characters long can be made. The player will place a bet with a minimum of 1 and maximum of 9 to see if they can form a word. The player cannot bet more than their total earnings.

Once a betting amount is decided, the player will press the pushbutton to start spinning the first character, and a second pushbutton press should slow the spinning down until it stops. The next pushbutton press will activate the second slot, then slow the second slot until it stops, similar to the first, and the same process will follow for the last slot. If a word is formed that matches one in the dictionary stored in the device's memory, then multiply the bet by a predetermined value depending on the word that was spelled and add it to their total earnings.

The table below describes the various tiers of words and their multiplier value:

Multiplier	**Words**
1	BAC, CAF, DEF, FAB, FAE, FAC
2	ADC, CAD, DAE, DEB, FEB
3	CAB, FAD, FDA
4	ACE, BAD
5	BED, DAB, FED

If a word with a multiplier of 4 or 5 is spelled, then flash the LED on and off for 2 seconds to signify some kind of jackpot.

If no word is formed then subtract the bet from their earnings.

If the player has 0 earnings remaining, then sound the buzzer to signify that they have lost and have no more earnings to spend. A "LOSE" screen is displayed and reset is the only accepted input at this point.

A reset may occur at any point of the game and should bring the state of the device back to the initial "HELLO" screen.

2 Implementation

The project was developed with a bottom-up design. Functions with more frequent usage and higher priority were developed first, and later pieced together to contribute to higher level functionalities. Because of reasons later detailed in the Troubleshooting section, the entire source code had to be contained in a single `main.c` script.

2.1 Memory Management

The implementation emphasized the limited memory on the chip. Usage of strings were minimized, and menu strings were stored as constant `PROGMEM` variables for reuse. The program memory was also utilized to communicate with the LCD with `lcd_puts_P`.

3 Usage

The project depends on user inputs from the Butterfly joystick and push button. The program sits in a loop until the `PUSH` button is pressed. The user may press the `UP`, `DOWN`, `LEFT` or `RIGHT` joystick buttons to toggle between the screens and options as per the description. The inputs are interrupt driven, and the LCD alerts the user whenever necessary.

4 Testing and Troubleshooting

Debugging the functionalities of the program required extensive usage of hardware. This attribute resulted in numerous reprogramming of the AVR Dragon. Small changes, especially during experimentation of values of signals sent to external devices such as the servo, slowed down the development process.

The development process took place on a Linux machine since it was much more user-friendly and convenient that Atmel Studio. `avr- gcc` was used to compile the source code, and `avrdude` provided the AVR libraries. Once substantial progress was determined in the development, the source code would be pushed to the version control system to transfer to a ready-for-upload Windows machine to program the chip. Although this process proved to be a faster approach for development, it required all the changes to be contained on a single file for frequent reprogramming.

Most of the issues arose from the external devices.

4.1 JTAG Cable Issues

The JTAG cable from the kit does not work anymore. Temporary replacements were obtained from peers, which slowed down development further.

5 Code

The C scripts used for the implementation has been attached alongside the report.

5.1 main.c

The entire source code of the project.