Austin Wroblos

CIS-11

Final Project Write-Up

I started the project wanting to use c++ to get random numbers and hold an array of cards, but as I progressed through the project, I found that I needed to take a more simplistic approach to complete this assignment. I began by setting up the random number generator in c++ because I wasn't entirely sure how to get random numbers in ARM assembly. After that, I struggled with trying to get values passed from c++ into my assembly functions after some time and much-needed assistance in class I was able to get that working. Once I was able to get the values into my ASM functions the rest of the project seemed pretty straightforward from there. I've made blackjack in c++ and java before so it was interesting to see how I could get it done in assembly, and it was easily one of the most challenging. I feel like I have progressed my knowledge immensely since the beginning of this project and have learned a lot about segmentation faults. I ended up deciding just to take the integer value of the cards and add them to the desired total rather than holding an array of all the cards. The logic behind this is that there are 13 different cards you can get from A-K (Ace's can only count as 1 in my version of blackjack). The random number generator would return the value to my dealing function, which would see if the card was a ten or higher. If it was higher it would add 10 to the hand and if smaller then it would add the random number directly into the player's hand. The program begins by asking you to click the left button to start the game once it is pressed, the player is given instructions, and the initial cards are dealt. Once the player has received their first 2 cards the binary value of their hand is represented on the breadboard with LEDs and then a check for blackjack is run. Following this, there is just a loop seeing if the player wants to hit or stand using the digitalRead function in the wiringPi library with buttons on the breadboard, so each time you hit the value of the card is added to your hand, and the LEDs are reset to the correct value. If at anytime the player goes over 21 the RGB led on the board will turn from green to red. If the player stands, then the dealer’s turn function begins, and if the dealer's hand is less than sixteen, then the dealer will hit until his deck value is worth sixteen or greater. After this, the player and the dealers' deck values are compared, and then the winner is decided by who has a higher value but is less than or equal to twenty-one. Lastly, everything is done in assembly besides generating random numbers.

C++ line count: 30 with blank spaces

ARM assembly line count: 480 with blank spaces

Notes: I have provided a diagram of my wiring on the breadboard but wasn't able to find a T-type cobbler breakout board in the software I was using so I just aligned everything as if it were there.