COE3DQ5-Lab #2 report

Group 6

Jingnan Chen (1073196) Piravinth Puvanesarajah (1055744)

chenj39@mcmaster.ca, puvanp@mcmaster.ca

Saturday, January 24, 2015

Exercise 1: in the while loop, we have a for loop and a counter to loop through all the switches to see if any of them are turned on. Then we check if any of Switch 16 or 17 is on. If so, then we check if Switch 17 is off, that means Switch 16 is on, we start start from the LSB to the MSB to do bitwise AND with (0x1<<i) to see if we get 0 out of it. If we get 0, we set the green led value to the i, which is the least significant switch that is off. If Switch 17 is on, we check if the switch value is equal to 0x3FFFF, if so, that means all the switches are turned on; otherwise we start from the LSB to the MSB to do bitwise AND with (0x1<<i) to see if we get 0 out of it. If so, that switch is the most significant switch that is off. After all of these are done, we check the number of switches that are turned on. If the number is greater than 9, then we do the bitwise OR between green led value and 0x1E0.

Exercise 2: in the while loop, we have a variable to store the previous Switch 16 value and we compare the previous flag with the current flag. If they are not the same, and if the Switch 16 value is from low to high, we find the two largest values in the previous 16 recorded values from a self-defined function. Also, there is a self-defined function to convert 16-bit signed values to string and prints the two largest values on LCD display. We set the column cursor to the 16 minus the length of the converted string to make the characters right justified. If the Switch 16 is from high to low, we print the entire 16 previous recorded values to the terminal with oldest value to latest value from left to right. Also we have a self-defined function to find the longest monotonic decreasing subsequence. We also have a variable to store previous switch value and compare with the current switch value. If any difference is detected, we shift the new value to the array.

Exercise 3: The first task that had to be accomplished in this exercise was to detect the interrupts of group A and group B. Once the interrupt had been detected and read, it had to be cleared so that it can be read again. Group A controls the amount in the machine. According to the switch that had been toggled (int_pos) we would increment the amount. Whenever the group A interrupt function is called, green LED 0 would be high showing that there is a transaction in progress. The next challenge was to make a function for group B with various cases within. The cases would correspond to the ticket destinations. Similar to group A, we once again check for interrupts, read the interrupts and clear the interrupt. With each case of int_pos, which holds what switch was toggled, in the group B function, we would call a different function called purchase. The purchase function deals with if the amount in the machine is greater than the ticket price or not. If the amount is greater than the ticket value, we continue with the transaction and display the balance on the seven segment. If the ticket value is not less than the amount in the machine, we light green LED 3 on for 1.5 seconds and keep green LED 0 on.