COE 4DS4 – Lab #2 Report

Group Number: 33
Moshiur Howlader / 001316948 / howlam@mcmaster.ca
Ryan Ganeshan / 001322407 / ganesr3@mcmaster.ca
January 17th, 2018

Exercise 1: This exercise involved modifying in-lab experiment 2. We first implemented a case statement that controlled sw[16] and [17] configurations. The code was arranged within a while loop, with a 1 second sleep period at the end to implement a 1Hz "polling" rate of the switches. **Case 0** (sw[17]) is implemented as follows. The value written to the GLED register is 1FF or 0 and the value is changed every second. Our **Case 1** (!sw[17] && sw[18]) was implemented by iterating through LED's 1-7, and then back to 7, and complementing the current bit AND the previous bit of the GLED data register, all within a *for l*oop. The *for* loop iterates every second. **Case 2** (!sw[17] & !sw[16]) is implemented by first setting the 8th LED on. Then the value that we write to the GLED is modified through many if-statements. Each LED- Dual-Switch pairing is assigned a frequency (extracted from its bit configuration), as well as a counter. We then implemented if statements to implement the toggling functions of the respective LED's. If an LED's frequency was 3 then we also checked the counter to see if 3 seconds has elapsed since the last toggle. Counters were reset after every toggle. Counters are incremented at the end of every loop iteration.

Exercise 2: The array that was initialized to store the 16 most recent samples were required to be signed. Otherwise, it was not possible to display negative inputs to the system. Switch_Val/ Switch_Val_BUF, and sw_16_val/sw_16_val_buf were signed and unsigned values used to hold the Switch 0 to 15 values, and Switch 16 value respectively. These were important conditions to control the flow of our program. Contents of the sample array were shifted up (remove array[15] element) everytime a change in Switch_Val occured. Basic loop logic was used to compute for max and second max. To find the longest strictly monotonic decreasing sequence, using the condition (array[i] > array[i+1]), we would loop through the array while updating three bookkeeping variables: beginIndex, endIndex, and longestLength for the monotonic sequence. To display the max and second max, we used the sprintf() function to convert our integer max and second-max to a string format, which would be compatible with the alt_up_character_lcd_string() function. The right justify logic onto the LCD was done using a set of if statements that checked the magnitudes of the max and second-max values in powers of 10 to determine the number of spaces to be shifted by. The negative sign was accounted for. Our max value and second max value algorithm would find the two highest values, not accounting for repeated values. If repeated values were specified to be avoided, we would included a condition to check against the value of max, in second max, and avoid considering this in our comparison.

Exercise 3: Two Interrupt Functions, SW_GRPA_interrupt and SW_GRPB_interrupt were implemented to handle the toggling of MSB (using a mask of 0x100) of Group A switches, and LSB (using mask of 0x1) of Group B switches. Three helper functions for interrupt A was implemented, one to find 4 LSB value, another to find 4 MSB values, and a third to check if these value were the same, triggering blinking of the GLED_8 for 3 seconds. For interrupt B, since coming into the interrupt function, it is given that a transition has occurred for the relevant interrupt switch. Knowing this, only other relevant condition to check was the current data value in the Switch_9's register, which was used to detect a rising or falling transition.