COE3DQ5 – Lab #2 Report

Mengjia Li (400011362)
Nicole Wu (400089778)
lim14@mcmaster.ca
wuz78@mcmaster.ca
Sep 20th, 2019

a. Upper/lower case for LCD

To achieve using left shift (12) for capital letters and right shift (59) for lower case letters, within *PS2_controller* code adding a logic of comparing PS2 code decide the output *PS2_upper_lower*; while during this time *PS2_make_code* is set to 0 avoid inputting shift key to display. In top-level file, input this signal to *PS2_to_LCD_ROM* MIF file. A 16 1 bit shift register *PS2_upper_lower_reg* is created to match up with *data_reg* in order to display together at the second line.

b. Top and bottom-line display

LCD_position and *data_counter* are used to tracking the top line and bottom line display, respectively. Once one of them reach F, the change-line instruction is sent and the displayed mood will change.

c. Letter comparison & 7 segment display

In order to compare the first and last character in the second line with any character in the first line, two additional 16 8 bit buffer registers: *data_store, data_compare* and are created. in an *always_comb* logic *data_compare[0]* and *[15]* is compared with each 8-bit *data_store* in a for loop to decide whether *first_char/last_char* flag needs to be asserted or not, which further be used to set 7 segment display to "d".

d. 5 seconds delay of green LED/7 segment display/erase operation

In LCD FSM *LCD_line_detect* is assigned with *LCD_line* so it will be shifted one register wise. Another FSM LED_state is used to handle this function. Two states are basically used: S_LED_INIT and S_LED_DELAY. During S_LED_INIT state, the *flag* is set to 0 and 5 seconds counter-counter is not counting at all. When *LCD_line_detect* is HIGH and *LCD_line* is LOW (switching from 2nd line back to the 1st line) and both *LCD_position* (counting 1st line) and *data_counter* (counting 2nd line) is 0, LED_state goes from S_LED_INIT to S_LED_DELAY. In S_LED_DELAY state, the flag is set to 1 in this state. *counte*r will count up to 5. When it reaches 5, the *counter* is reset, the state now goes back to S_LED_INIT. All green LEDs and left most two 7 segments have ON/OFF based on this *flag* signal. To perform erase operation after 5 seconds of delay, another new state S_LCD_RESET is created in LCD FSM. During S_LCD_FINISH_CHANGE_LINE state, if *LCD_line* is HIGH then the state will enter S_LCD_RESET state. Within reset state, *LCD_erased* flag is set to 1. LCD signals are re-initialized to ensure it starts from the first line and the first character. Also, when LED_state is detected to be S_LED_INIT (*2'd0*), LCD_instruction will be assigned with *9'h001* to empty the display.

e. Push-button exception handling

The third FSM BUTTON_state is created for handling push-button cases. There are 5 states in total: S_IDLE, S_ONCE, S_TWO, S_THREE and S_DISPLAY. When it is in S_IDLE, *button_erase* flag is 0, *pb_detected[3:0]* signal is used to check if any of the buttons has been pressed once. if it holds true, a *push_count* counter will be added by 1, the state now goes to S_ONCE. At S_ONCE state, 4 if the statement is executed to check *pb_detected* signal again, while any of it is pressed, *push_count* signal is checked to decide whether FSM should proceed to S_TWO (the same button pressed twice) or goes back to S_IDLE, if the condition holds true, *push_count* will be added by 1 again. S_THREE is similar to S_TWO. While in S_DISPLAY state, *button_erase* flag is set to 1, the state goes back to S_IDLE. *button_erase* flag now is used in LCD FSM to erase the display immediately along with *LCD_erased* flag.