Laboratory 6: Calculator [8 Bit]

1 INTRODUCTION

In this lab you will be creating an 8 bit calculator with basic addition, subtraction, multiply, and divide operations. The calculator will also have a MR and MS button which allow one to save the present result into memory and to retrieve values from memory. You are encouraged to use packages for this lab and to start building up your own IP library. The technical requirements for this lab are listed below.

1. Shall employ the 4 pushbuttons as 'execute', 'ms', 'mr', and 'reset' moving from left to right.
    a. Might have to do a reset inversion in the top level due to pushbutton polarity.
2. Shall use the 8 right-most switches for the second input to the calculator.
3. The first input into the calculator shall come from either the working register or the save register.
4. Shall implement an 8 bit wide memory that contains the working register and save register. Only really need two rows, however you might have to create a 4x8 bit memory since the address line wants to be a std_logic_vector [at least 2 bits] instead of std_logic.

address	reg description
00	working register
01	save register
10	----
11	-----

5. Shall use the 2 left-most switches to select the desired operation
    a. [0:0] add [0:1] sub [1:0] multiply [1:1] divide
6. The working register shall be displayed on 3 seven segment LEDs in decimal form.
7. The working register shall only be updated upon a 'mr' or 'execute' button push.
8. Pressing the 'ms' button shall save the present working register into the save register.
9. Pressing the 'mr' button shall load the save register into the working register.
10. Calculator inputs shall both be 8 bit and the calculator output shall be 8 bit.
11. Shall operate with unsigned base 256 numbers. Ex. 0 - 1 = 255
12. All inputs shall be synchronized with the 50 MHz clock.
13. Shall display the present state via LEDs.

2 PRE-LAB [BLOCK DIAGRAM]

You shall create an order of operations list as well as a block diagram with a state transition diagram accurately mapping out your design. Also bring a printout of the block diagram to your lab class for an open discussion. Make sure to include proper synchronization, edge detection, and signal names in your diagram as well as a clear notation indicating the bit width of various busses and whether a block is a component or process. A state transition diagram as well as an order of operations list is also required.

3 SIMULATION

Create a simulation that tests out the use case shown in the posted video.

4 HARDWARE

Target your design onto the DE1 SoC and receive a signoff. To save you time, I posted my compile.tcl file that lists the pin assignments that I used.

5 DELIVERABLES

To receive full credit for this lab one must hand in the below items no later than 168 hrs [7 days] after the start of one's lab session. Signoffs can be obtained after the due date as long as the time stamp of the code is from before the deadline.

- ☐ Hard copy of this document.
- ☐ Hard copy of new src files [no tabs and print from notepad++ with 'show symbols' on].

6 SIGNOFFS NAME:_____

Category	Initials	Date	Points
Block Diagram			/20
Simulation			/30
Demonstration			/40
Deliverable			/10
Final Grade			/100