CS-223 DIGITAL DESIGN PROJECT

CALCULATOR WITH FOUR OPERANDS

SENA KORKUT

SECTION-4

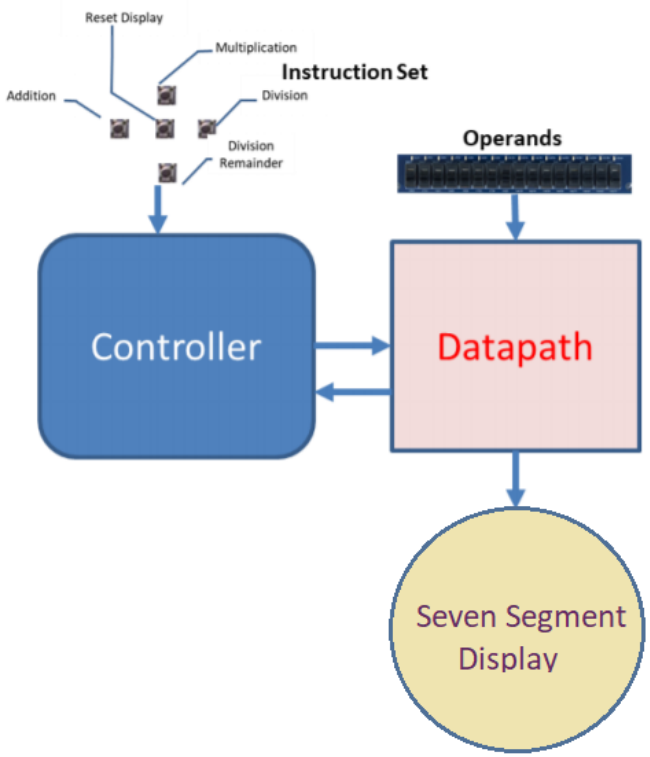
21703303

13/05/2019

INTRODUCTION:

The aim of this project is to implement a simple calculator to BASYS3. The calculator can operate four different calculations: adding, subtracting, multiplying and dividing and inputs of this calculator is 8-bit signed numbers which means the most significant bit of the binary number states the sign. In order to satisfy the requirements of the calculator, HLSM logic is used to determine separate states of calculator such as the type of the operand and the view of seven-segment display (result of the operation is blinking in every 500 milliseconds). To implement the calculator to hardware, BASYS3 is used. Switches and leds are representing two 8-bit inputs while buttons are executing the operations and resetting the result when pushed. The result is displayed with seven-segment display whereas default segments are displaying 0000. Results are displayed as hexadecimal numbers.

BLOCK DIAGRAM:



High Level State Machine of Top-Level Design:

- Inputs / Outputs and Local Variables of the System:

A: 8-bit (signed two's complement) input from leds corresponding to switches

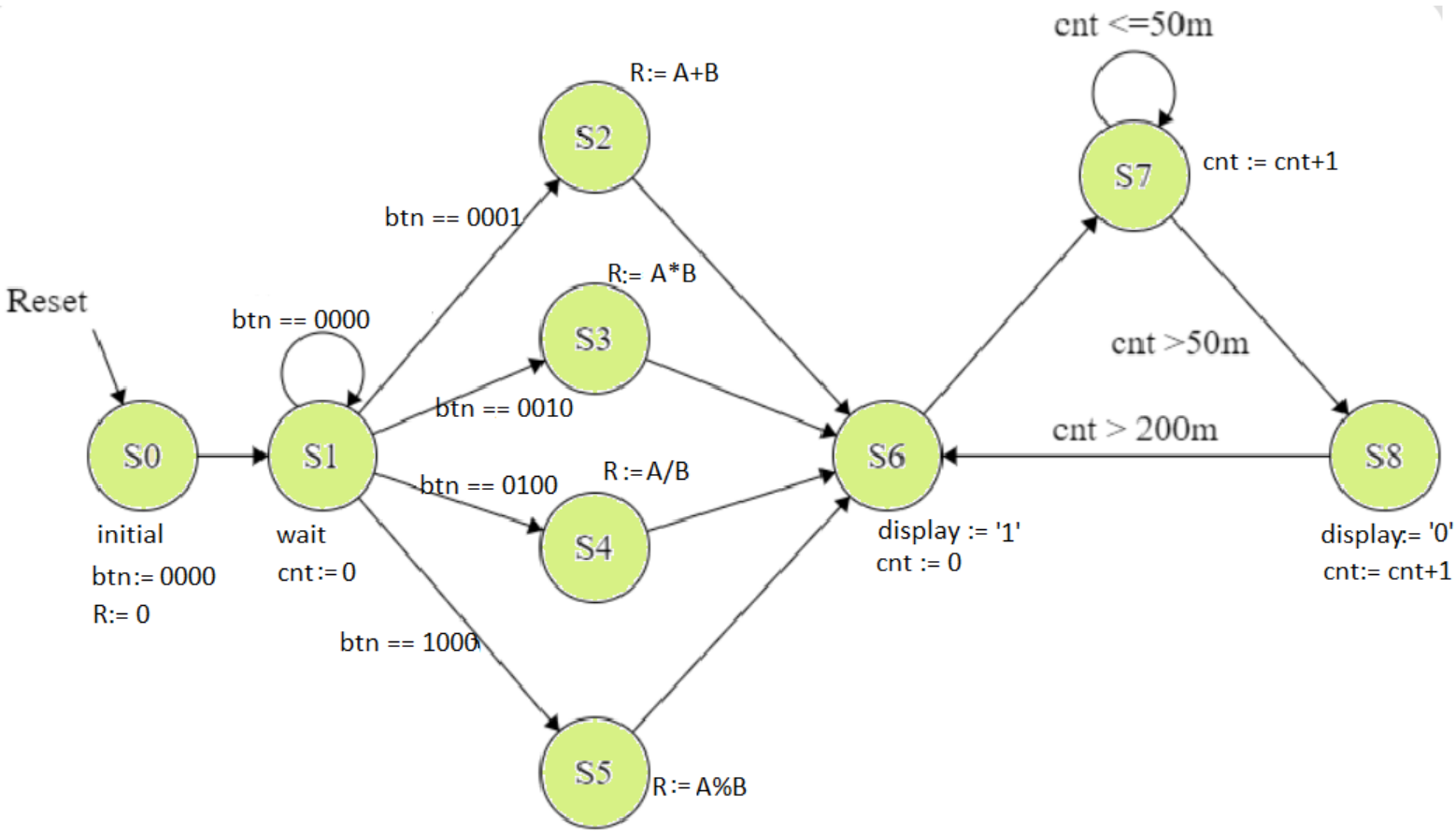
B: 8-bit (signed two's complement) input from leds corresponding to switches

btn: 4-bit input from buttons pushed

R: 16-bit output representing result

cnt: counter to count from 0 to 50 million and 200 million (for blinking system)

display: bit output to decide displaying the result



Datapath and Controller Block of HLSM:

- Inputs/Outputs of Datapath and Controller

clr: bit input for datapath/bit output from controller to clear up-counter

inc: bit input for datapath/bit output from controller to decide to increase

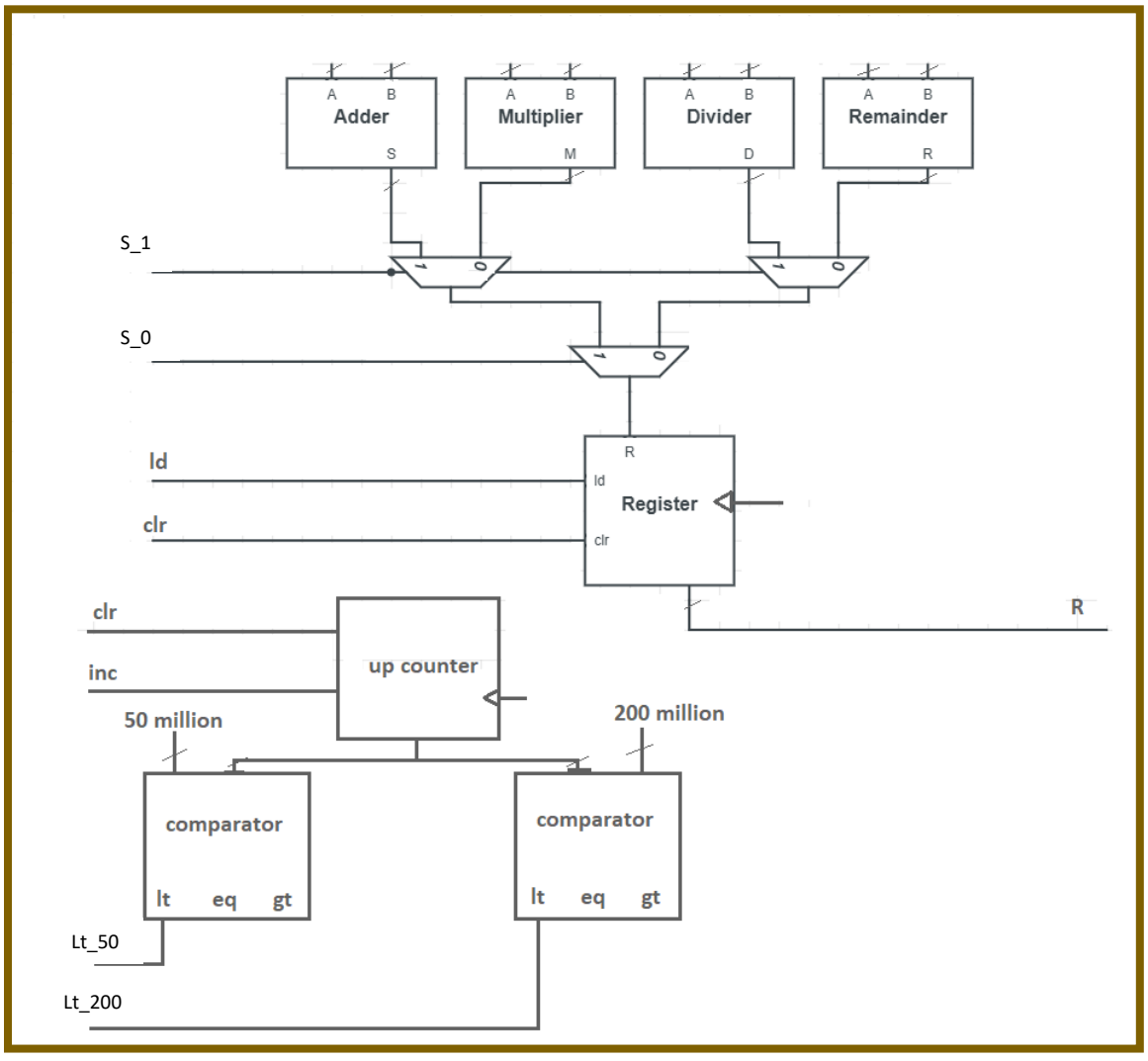
ld: bit input for datapath/ bit output from controller to load register

reg_clr: bit input for datapath/ bit output from controller to clear register

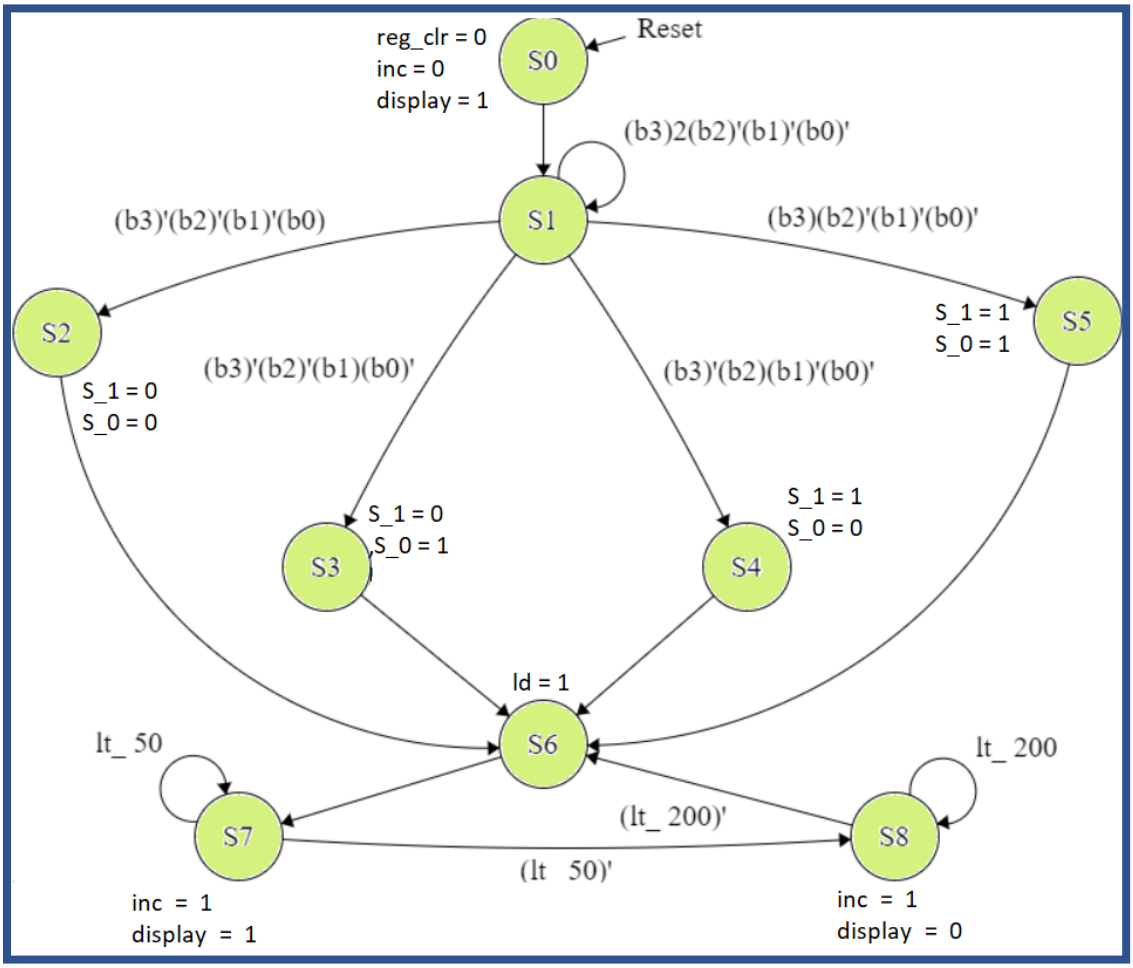
lt_50-lt_200 bit output from datapath/bit input for controller to decide increasing cnt

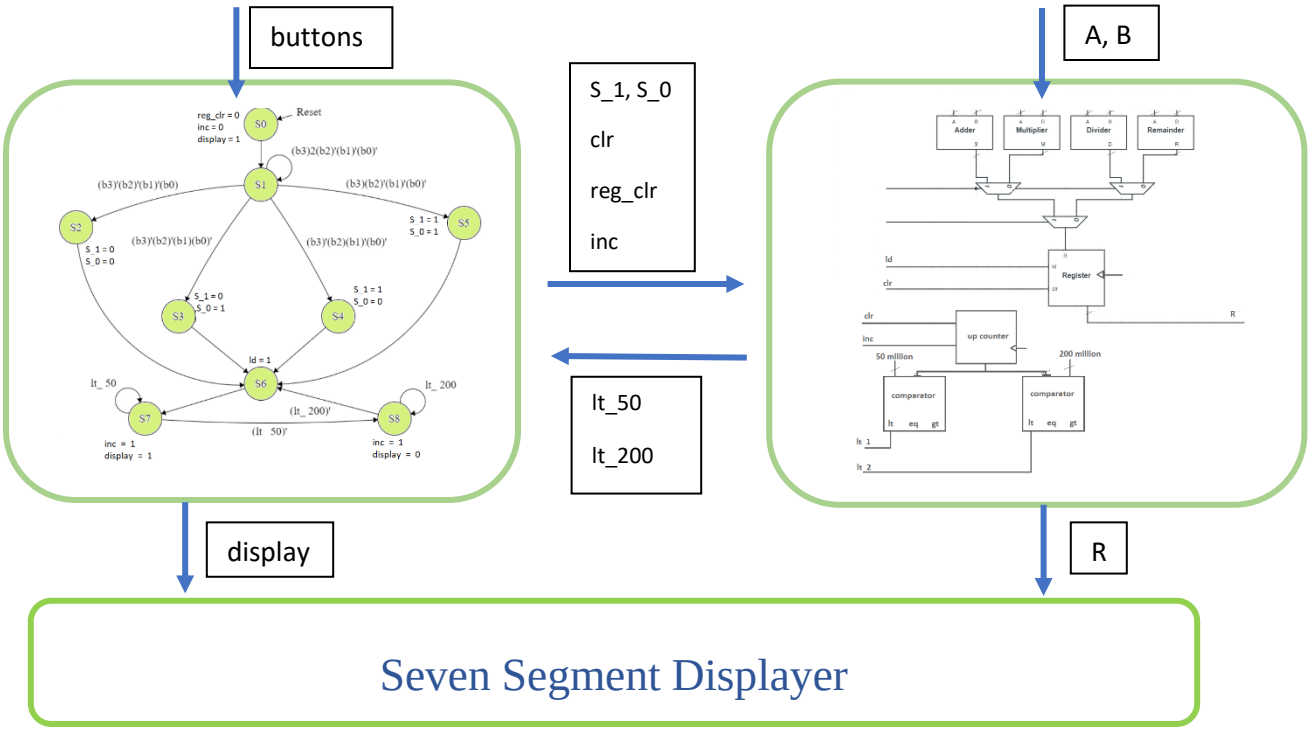
S: 2 bit input for datapath/ 2 bit output from controller to select operation

DATAPATH

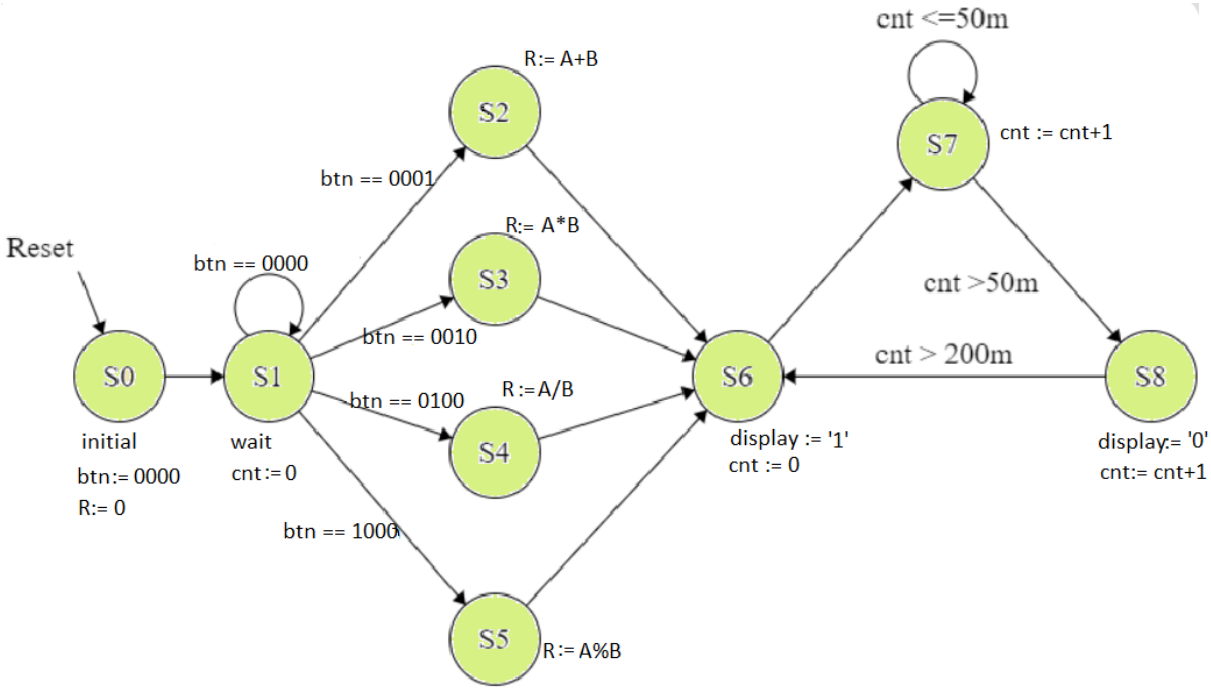


CONTROLLER


reg_clr = 0
inc = 0
display = 1
Reset
S0
(b3)2(b2)'(b1)'(b0)'
S1
(b3)'(b2)'(b1)'(b0)
(b3)(b2)'(b1)'(b0)'
S2
S5
S_1 = 1
S_0 = 1
S_1 = 0
S_0 = 0
(b3)'(b2)'(b1)(b0)'
(b3)'(b2)(b1)'(b0)'
S3
S4
S_1 = 0
S_0 = 1
S_1 = 1
S_0 = 0
ld = 1
S6
lt_ 50
lt_ 200
S7
S8
(lt_ 200)'
(lt 50)'
inc = 1
display = 1
inc = 1
display = 0


buttons
A, B
S_1, S_0
clr
reg_clr
inc
lt_50
lt_200
display
R
Seven Segment Displayer

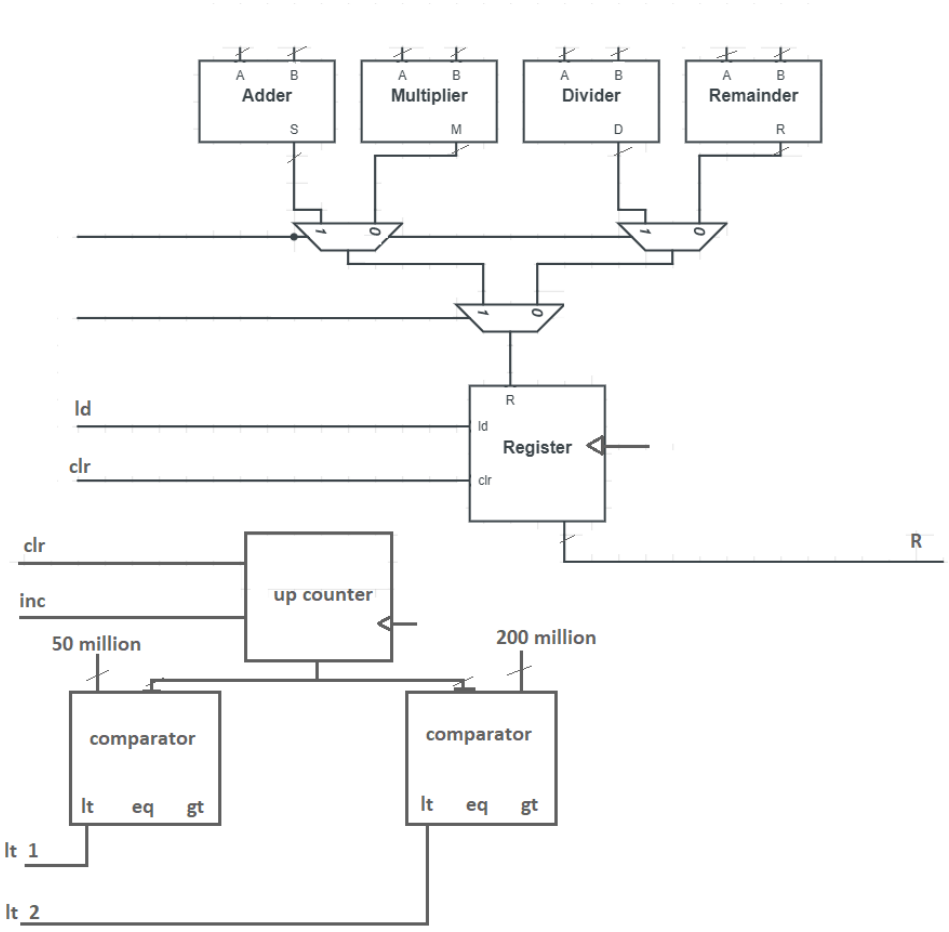
DETAILED EXPLANATION OF THE WORK:



The top-level design of the system is a HLSM. As seen above, the system starts with S0, the initial state. Also, when reset is pushed, the system should begin from the initial state. In the initial state, the local variables and result output are initialized. Without any condition, it goes to next state, S1. In this state, it waits for button signals. The next state is S1 when there is no button signal coming. When an input comes, it goes to next states. According to its inputs from buttons, S2, S3, S4 and S5 have different assignments for the result. If the fourth bit of the button is '1', it goes to S2, which assigns R (A+B), if the third bit of the button is '1', it goes to S3, which assigns R (A*B), if the second bit of the button is '1', it goes to S4, which assigns R (A/B) and lastly, if the first bit of the button is '1', it goes to S5, which assigns R (A%B). After all these states, it goes to S6, which is for displaying the result in seven segment display. Display variable is assigned to '1' in S6 and the counter is assigned to 0. Then, it goes to S7 and starts incrementing the counter. It counts until counter reaches 50 million and then goes to S8 to turn

off the seven-segment display. In S8, it counts until it reaches 200 million, since display must be off for 1500 milliseconds and on just for 500 milliseconds. From S8, it returns to S6 when counter reaches 200 million and clears the counter to 0, then repeats.

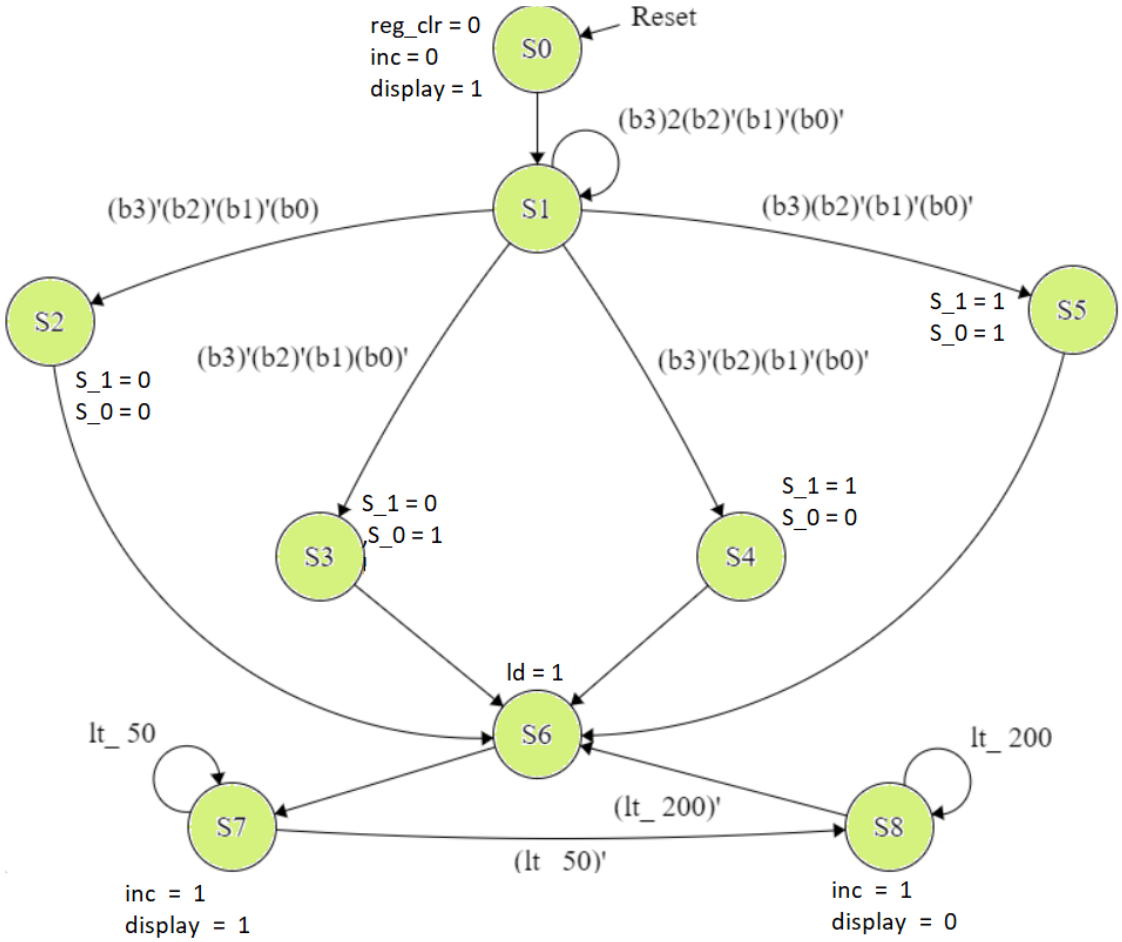
Description of Datapath:



The datapath of the system consists of different kind of components. The inputs coming from the switches are the inputs A and B of the datapath. After coming to the system, the inputs go to four different custom boxes. In SystemVerilog code of the boxes, adder, multiplier, divider and remainder are operated with operators. As outputs of this operations, the right operation is selected with the help of 2 bit select (S) input coming from controller. Choosing the right results with multiplexers, the results goes to register to be stored. It is stored according to the inputs

coming from controller. If the clr input is ‘1’, result is assigned to 0. If the load input is ‘1’, result is stored and else the stored value from before is kept. Besides keeping the result, varied circuit components are used to display blinking. An up-counter, which increments its local variable count in every increase signal where it is ‘1’ and resets the count to 0 when clr signal coming from controller is ‘1’. The output goes to two different comparators, one for checking if the coming input is less than, equal to, or greater than 50 million and the other one is for the same but for 200 million. In SystemVerilog module of comparator, it returns only less than (lt) output to controller. Two different outputs of two comparators go to controller for the state conditions in FSM in controller.

Description for Controller:



In controller part of HLSM, a finite state machine of the whole design is designed. The external inputs for controller are the inputs coming from buttons. As first state, S0, the inputs and local variables are initialized to ‘0’ and display is initialized to ‘1’, also when reset is pushed, it goes to S0. After S0, it goes to S1 where it waits for the button signal to come and returns to itself until some button is pushed. When some button is pushed, it goes to different states S2, S3, S4 and S5. According to the arranged signal outputs, signal 2’b00 is assigned for A+B. Signal 2’b01 is assigned for A*B, 2’b10 is assigned for A/B and 2’b11 assigned for A%B. In these states, the signal output to be sent to datapath is initialized. After initializing states, the next state is S6, where load input for register in datapath is assigned to ‘1’. From S6 to S7, S7 waits until the variable coming from datapath (lt_50) is false and it makes display variable ‘1’ and inc ‘1’. When the counter in up-counter from datapath is not less than 50 million, S7 goes to S8 where display will be ‘0’ and waits until the counter variable of up-counter from datapath is greater than 200 million. When it reaches 200 million, lt_200 becomes ‘0’ and S8 returns to S6, initializing clr to ‘1’.

During coding SystemVerilog modules, writing the components and connecting them together with the help of circuit design was easy as it was designed earlier by separating each condition and operation. However, there were so many errors when pushing the buttons and displaying the matching result, such as in every button pushed, the result was the summation of inputs regardless of the type of the button. In addition to this, at the beginning, the sum of the inputs was being displayed and it was changing even when the switches are changed without pushing the reset button. To solve these problems, some additional code blocks are implemented to the design modules which are not mentioned in datapath and controller. To trace the wrong part of the code and correct it, testbench for mux is used because the problem part was selecting the matching operator. Simulation did not any wrong outputs whereas in implemented hardware,

outputs of divider, multiplier and remainder were not correct. Trying different code blocks, when initializing select signal, it has been concluded that the problem was in controller when initializing select signals in different states. Moreover, when taking the negative numbers into account, some changes in operation modules and seven segment modules were necessary. Apart from the design, in code modules, there is a bit variable sign to control and decide if the A, B and result is negative or positive.

CONCLUSION:

For this project, a simple calculator that has four different operators has been implemented to hardware (BASYS3). For the operands, buttons of the board are used and for the 8-bit inputs, switches and leds are used. The result is displayed in seven segment display of BASYS3 and the result is blinking (open for 500 milliseconds and off for 1500 milliseconds). In order to solve the problem, high level state machine logic is used and for the circuit of HLSM, datapath and controller logic of the system is analyzed. Datapath circuit and FSM of the controller is turned into a SystemVerilog module with different circuit components inside of it. During studying the project, errors of SystemVerilog code (tracing the code to find the problem) and delay problems of the clock signal have caused some hardship. So much time has spent to understand the syntax of SystemVerilog and the error types. In some cases, there were problems in connecting inputs and outputs of different circuit components even if there is no error shown by the program.

REFERENCES:

- Seven Segment Display code from Unilica (given module for labs)
- DDCA chapter 4
- Project_report pdf file