UNIVERSITY OF NEVADA LAS VEGAS. DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING LABORATORIES.

Class:	CPE-300L		Semester:	Spring 2020
Points		Document author:	**Brysen Kokubun**	
		Author's email:	**Brysenkokubun96@gmail.com**	
		Document topic:	**Prelab 6**	
Instructor's comments:				

Introduction / Theory of operation

- For this lab experiment we will be learning how to implement the general datapath and its components. The components for our general datapath is a multiplexer, a Register files, an ALU, a shifter, and a tristate buffer.

Prelab main content

1. Understand the objective of the summation algorithm and explain how the algorithm works.

- The summation algorithm starts with the sum = 0 and takes in some input n. If the value of n is != 0 then add the current sum value and the current n value. Next decrement the n value and check if it is != 0. Keep decrementing the n value until the condition is met. Once the condition is met then output the value of the sum.

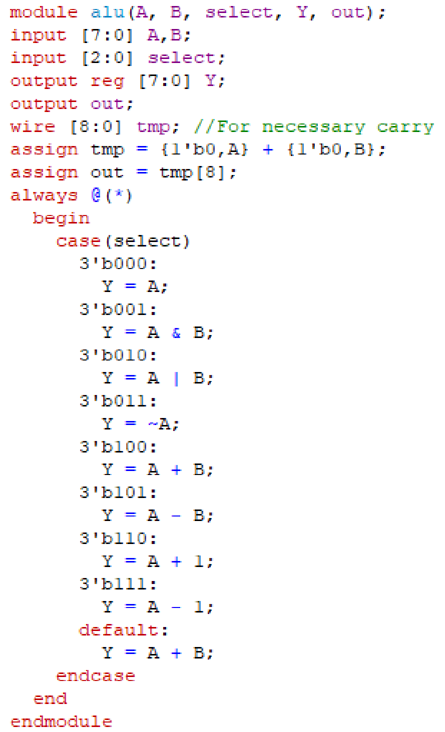
2. Identify the components in the general datapath from Fig 2 and explain the use of each component with respect to the summation algorithm?

- The MUX decides if it is a new input n or feedback data from the GDP, the feedback data will not stop until the condition of n = 0 is met.
- The OR gate gives a signal to the control unit and verifies if the n = 0 or n != 0.
- The Reg File is composed of 4 8-bit registers that can write/store, and read data, the data within the Reg File is the accumulated sum.
- The ALU is where the arithmetic logic/operations on the data are performed. The ALU will decrement n by 1 and sums the current sum value with the current n value.
- The shifter will pass through the data.
- The tristate buffer prevents the output to be enabled until n = 0. Once n = 0 then the tristate buffer will allow the data to be outputted.

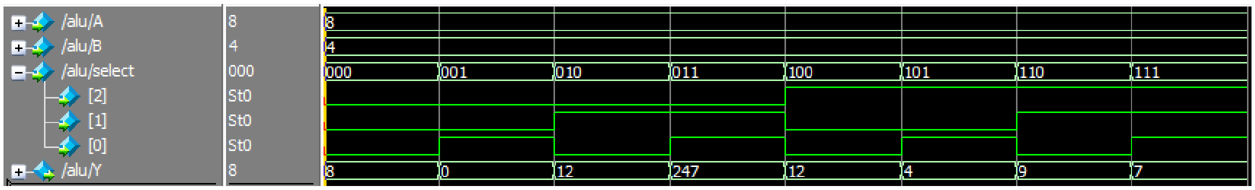
3. What are all the input and output signals shown in Fig2 associated with each block, explain all of them.

- IE: Input Enable for the MUX indicates if there is new data or if it is feedback data from the GDP
- WE: Write Enable for the Reg File allows the data to be written to a register
- WA: Write Register for the Reg File has 2 signals 1 and 0 each combination allows to select which of the 4 registers to write to
- RAE: Read Enable Register for register A, this allows data in register A to be read into the ALU
- RAA: Read Register for A, has 2 signals 1 and 0 to indicate which register of A to read from
- RBE: Read Enable Register for register B, this allows data in register B to be read into the ALU
- RBA: Read Register for B, has 2 signals 1 and 0 to indicate which register of B to read from
- ALU: Arithmetic Logic Unit, where logic/computation is made, 3 selection signals justify the ALU's operation.
- SH: Shifter, 2 signals that will justify the shifters operation
- OE: Output Enable, 1 signal that will either cause the data to be outputted or feedback through GDP

4. Write a Verilog code for the ALU described in Fig 2b and verify using the Modelsim.



```
module alu(A, B, select, Y, out);
input [7:0] A,B;
input [2:0] select;
output reg [7:0] Y;
output out;
wire [8:0] tmp; //For necessary carry
assign tmp = {1'b0,A} + {1'b0,B};
assign out = tmp[8];
always @(*)
  begin
    case(select)
      3'b000:
        Y = A;
      3'b001:
        Y = A & B;
      3'b010:
        Y = A | B;
      3'b011:
        Y = ~A;
      3'b100:
        Y = A + B;
      3'b101:
        Y = A - B;
      3'b110:
        Y = A + 1;
      3'b111:
        Y = A - 1;
      default:
        Y = A + B;
    endcase
  end
endmodule
```



/alu/A
/alu/B
/alu/select
[2]
[1]
[0]
/alu/Y
8
4
000
St0
St0
St0
8
8
4
000
001
010
011
100
101
110
111
8
0
12
247
12
4
9
7