**Digital Computer Design Laboratory**

**#ALU**

**1-** Create the following files **fulladder.sv**, **NbitFulladder.sv**, **MUX2.sv** and **MUX4.sv**.

module fulladder(input logic a, b, cin,

output logic s, cout);

assign s=a^b^cin;

assign cout= a&b|b&cin|cin&a;

endmodule

**fulladder.sv**

module NbitFulladder( input logic [7:0] a, b,

input logic cin,

output logic [7:0] s,

output logic cout );

wire [6:0] c;

fulladder i\_0 (a[0],b[0],cin,s[0],c[0]);

fulladder i\_1 (a[1],b[1],c[0],s[1],c[1]);

fulladder i\_2 (a[2],b[2],c[1],s[2],c[2]);

fulladder i\_3 (a[3],b[3],c[2],s[3],c[3]);

fulladder i\_4 (a[4],b[4],c[3],s[4],c[4]);

fulladder i\_5 (a[5],b[5],c[4],s[5],c[5]);

fulladder i\_6 (a[6],b[6],c[5],s[6],c[6]);

fulladder i\_7 (a[7],b[7],c[6],s[7],cout);

endmodule

**NbitFulladder.sv**

module mux2(input logic [3:0] d0, d1,

input logic s,

output logic [3:0] y);

assign y = s ? d1 : d0;

endmodule

**MUX2.sv**

module mux4(input logic [3:0] d0, d1, d2, d3,

input logic [1:0] s,

output logic [3:0] y);

assign y = s[1] ?

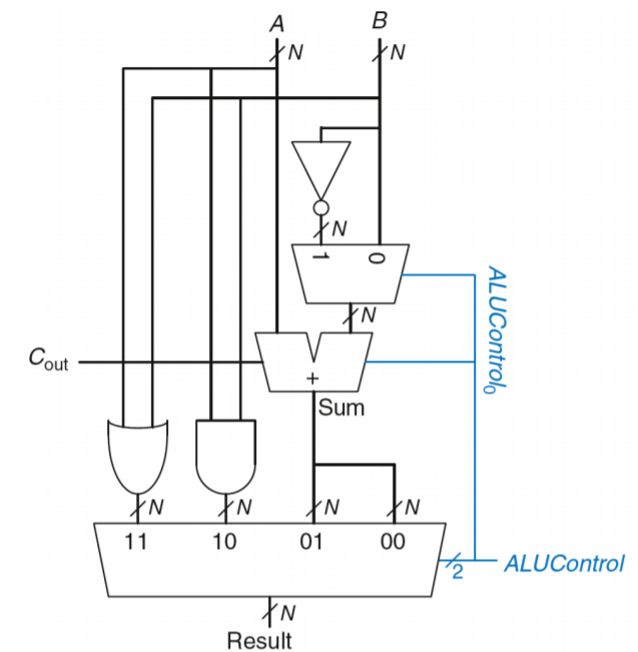
(s[0] ? d3 : d2):

(s[0] ? d1 : d0);

endmodule

**MUX4.sv**

**2-** Create 8-bit ALU module given below using **NbitFulladder**, **MUX2** and **MUX4** modules and create a **testbench** for this module.



|  |  |
| --- | --- |
|  | Function |
| 00 | **Add** |
| 01 | **Subtract** |
| 10 | **AND** |
| 11 | **OR** |