**Using case statement design function ALU that takes 4-bit input ‘A’ & ‘B’ & 3-bit input signal ‘select’ & gives 5-bit output ‘out’.The ALU implement the following function based on 3-bit input ‘select’.Ignore any overflow or underflow bits.**

|  |  |
| --- | --- |
| **Select Signal** | **Functions** |
| **3'b000** | **out=a** |
| **3'b001** | **out=a+b** |
| **3'b010** | **out=a-b** |
| **3'b011** | **out=a/b** |
| **3'b100** | **out=a%b** |
| **3'b101** | **out=a<<1** |
| **3'b110** | **out=a>>1** |
| **3'b111** | **out=a>b** |

**Code**

* **Design**

module mux(out,a,b,select);

input [3:0] a,b;

input [2:0] select;

output [4:0] out;

reg [4:0] out;

always @(a or b or select)

begin

case({select})

3'b000: out=a;

3'b001: out=a+b;

3'b010: out=a-b;

3'b011: out=a/b;

3'b100: out=a%b;

3'b101: out=a<<1;

3'b110: out=a>>1;

3'b111: out=a>b;

endcase

end

endmodule

* **TestBench**

module Baig;

reg [3:0] a,b;

reg [2:0] select;

wire [4:0] out;

mux m(out,a,b,select);

initial

begin

a=4'b1001;b=4'b0011;select=3'b000;

#30

select=3'b001;

#30

select=3'b010;

#30

select=3'b011;

#30

select=3'b100;

#30

select=3'b101;

#30

select=3'b110;

#30

select=3'b111;

#30

$finish;

end

endmodule

**Output**

