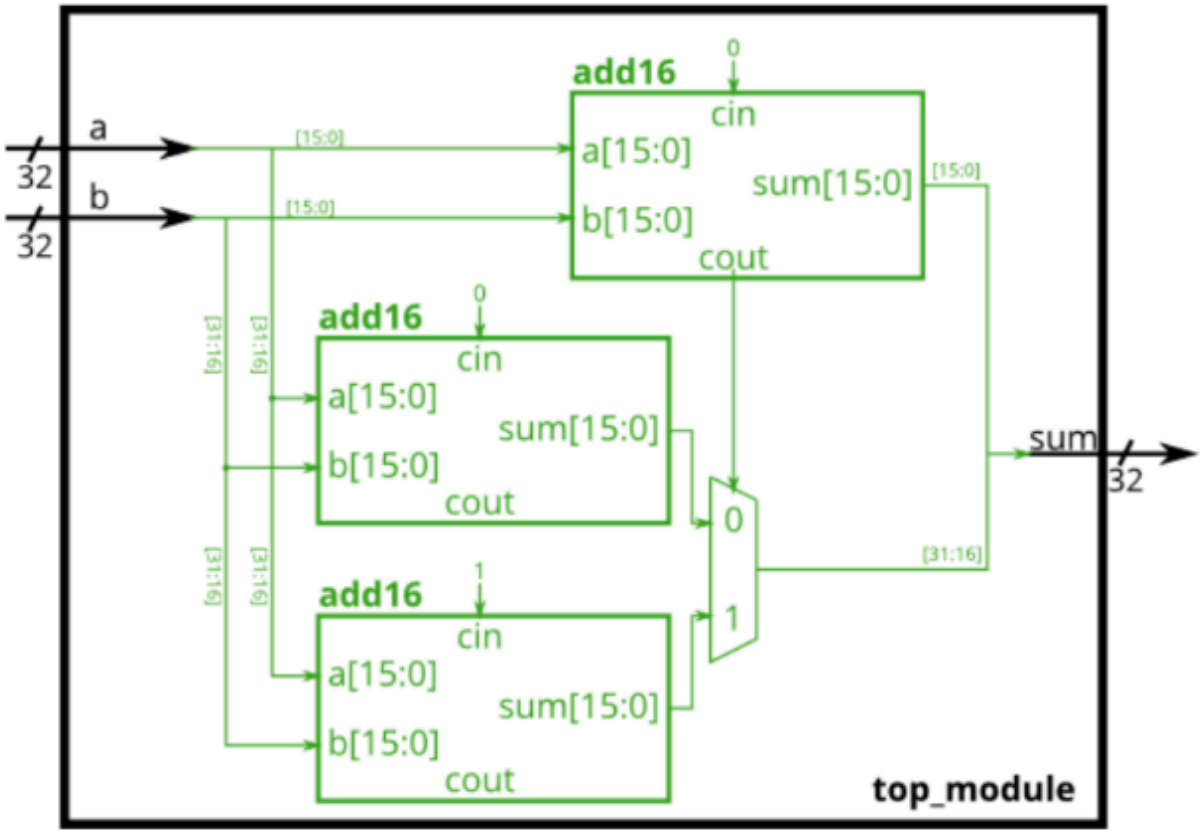
KARTHICK RAJA B
126004121

Given:

In this exercise, you have to design a module add16 as the previous exercise, which adds two 16-bit numbers with carry-in and produces a carry-out and 16-bit sum. You must instantiate three of these to build the carry-select adder, using your own 16-bit 2-to-1 multiplexer.



Verilog code:

```
module RTL_Day_9(a,b,sum);
input [31:0] a,b;
output[31:0] sum;
wire w1,w2,w3;
wire[31:0] sum1,sum2;
add16 a1 (a[15:0],b[15:0],0,sum[15:0],w1);
add16 a2 (a[31:16],b[31:16],0,sum1[31:16],w2);
add16 a3 (a[31:16],b[31:16],1,sum2[31:16],w3);
```

```
assign sum[31:16] = w1 ? sum2[31:16] : sum1[31:16];
endmodule
```

Testbench code:

```
module test_bench_tb9 ();
reg [31:0] a,b;
wire [31:0] sum;

RTL_Day_9 s1 (a,b,sum);
initial
begin

a={32{1'b1}}; b={32{1'b0}};   #100;
a={32{1'b1}}; b={32{1'b1}};   #100;
a={32{1'b1}}; b={16{2'b10}};  #100;
a={16{2'b01}}; b={32{1'b0}};

end
endmodule
```

Functional Simulation: