Verilog code :-

```
module multiply ( product, a, b);

input [31:0] a, b ;

output reg [63:0] product ;

reg [31:0] abs_a = 0;

reg [31:0] abs_b = 0;

integer i;

always @( a or b )

begin

abs_a = 0;

abs_b = 0;

product = 0;

//unsigned multiplication

if (a[31] | b[31] == 0)

begin

for(i=0; i<32; i=i+1)

if( a[i] == 1'b1 ) product = product + ( b << i );

end

//signed multiplication

if (a[31] | b[31] == 1)

begin

abs_a[30:0] = a[30:0] ;

abs_b[30:0] = b[30:0] ;

product = 0;

Â Â

for(i=0; i<32; i=i+1) begin

if( abs_a[i] == 1'b1 )

begin

product = product + ( abs_b << i );

end

end

product = -product;

end

end

endmodule
```

I simulated this code to verify its successful operation, and the code successfully passed in simulation, i am pasting the test bench i wrote and resulting waveform for u..

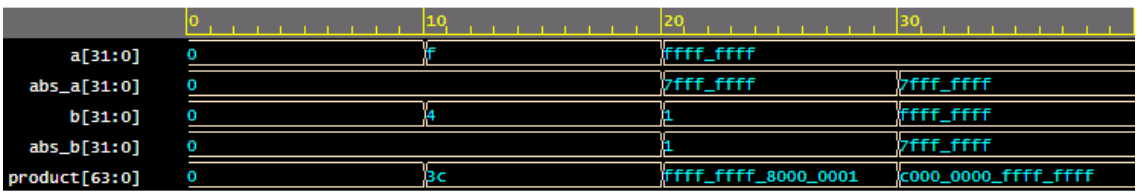
Testbench :-

```
// single line comment

/* multi
```

```
line
comment
*/
module multiply_tb ();
reg [31:0] a, b ;
wire [63:0] product ;
Â Â
multiply m ( product, a, b);
Â Â
initial begin
$dumpfile("d.vcd");
$dumpvars(1, m);
Â Â
a = 0; b = 0;
#10 a = 32'b 0000_0000_0000_0000_0000_0000_0000_1111 ;
b = 32'b 0000_0000_0000_0000_0000_0000_0000_0100 ;
/*
a = f(hex) and b = 4(hex)
thus, product = ( 15 * 4)decimal = (60)decimal = (3c)hex
*/
Â Â
//multiplication of a signed number with unsigned number
#10 a = 32'b 1111_1111_1111_1111_1111_1111_1111_1111; // a signed number
b = 32'b 0000_0000_0000_0000_0000_0000_0000_0001 ;
Â Â
//multiplication of a signed number with another signed number
#10 b = 32'b 1111_1111_1111_1111_1111_1111_1111_1111; // a signed number
a = 32'b 1111_1111_1111_1111_1111_1111_1111_1111; // a signed number
Â Â
#10 $finish;
end
endmodule
```

Waveform :



If u have any query do comment below.

Hope u find my answer helpful. If u do please upvote this.

Keep Chegging !

0 10 20 30 ffff_ffff fff_ffff 0 a[31:0] abs_a[31:0) b[31:0] abs_b[31:0] product[63:0] fff_ffff fffffff 11 0 fff_ffff 0 30 fffffff_8000_0001 C000_0000_ffff_ffff