1 INTRODUCTION

A 4-bit binary up/down counter counts sequence from 0000 to 1111 and 1111 to 0000. The circuit operation can be explained as follows:

- When select line of the multiplexer is zero, the counter is doing up count from 0 to 15 for every trailing edge of the clock.
- When select line of the multiplexer is one, the counter is doing down count from 15 to 0 for every trailing edge of the clock.

Here, we implemented an up-down counter using the 2:1 multiplexer and a 4-bit binary counter. Consider I_0 and I_1 are the input lines, S is the select line and m_o is the output line of multiplexer. If $S = 0$ then I_0 is selected otherwise I_1 is selected at the output.

Consider a 4-bit synchronous up-down counter. The output of MUX is connected to the UP/DOWN pin of the counter such that whenever I_0 is selected at the output of the MUX, the counter is doing up counting and when I_1 is selected at the output of MUX, the counter is doing down counting. If the reset pin of counter is set to 1, then count value is reset to zero.

2 VERILOG CODE

Following is the verilog implementation of 4-bit synchronous up-down counter:

```
module updowncountms(clk,rst,count,select,mout,I0,I1);
input clk, rst, select,I0,I1;
output reg mout;
output reg [3:0]count = 0;

always @ (select or I0 or I1)
begin
if(select == 1'b0)
mout <= I0;
else
mout <= I1;
end

always @ (clk)
begin
if(rst == 1)
count <= 0;
else
if(mout == I0)
count <= count + 1;
else
count <= count - 1;
end
endmodule
```

3 TEST BENCH CODE

Following is the test bench code for implementing 4-bit synchronous up-down counter:

```
module t_updownms;

        // Inputs
        reg clk;
        reg rst;
        reg select;
        reg I0;
        reg I1;

        // Outputs
        wire [3:0] count;
        wire mout;

        // Instantiate the Unit Under Test (UUT)
        updowncountms uut (
                .clk(clk),
                .rst(rst),
                .count(count),
                .select(select),
                .mout(mout),
                .I0(I0),
                .I1(I1)
        );

        initial begin
                // Initialize Inputs
                clk = 0;
                rst = 0;
                select = 0;
                I0 = 0;
                I1 = 1;

                // Wait 100 ns for global reset to finish
                #0.01;
                clk=1;
                #80;
                select = 1;rst=1;
                //#80;
                rst = 0;

                // Add stimulus here
        end
        always #5 clk = ~clk;

endmodule
```