**DIGITAL DESIGN & COMPUTER ORGANIZATION LABORATORY**

**PROJECT REPORT**

**ON**

Design & implement a 3–bit Up/Down Counter.

SUBMITTED BY

**NAME SRN**

**4) Sai Sahana R K PES2UG19CS354**

**AUGUST – DECEMBER 2020**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**ELECTRONIC CITY CAMPUS,**

**BENGALURU – 560100, KARNATAKA, INDIA**

|  |  |  |
| --- | --- | --- |
| TABLE OF CONTENTS | | |
| Sl.No | TOPIC | PAGE No |
|  | ABSTRACT OF THE PROJECT | 1 |
|  | CIRCUIT DIAGRAM | 2 |
|  | MAIN VERILOG CODE | 4 |
|  | TEST BENCH FILE | 5 |
|  | SCREEN SHOTS OF THE OUTPUT | 6 |

**ABSTRACT OF THE PROJECT:**

Counters are used in many different applications. Some count up from zero and provide a change of output reaching a predetermined value; others count down from a present value to zero to provide an output state change.

However, some counters can operate in both up and down count mode, depending on the state of an up/down count mode input pin. They can be reversed at any point within their count sequence.

The 3-bit Up/Down counter is a bidirectional counter. Bi-direction counters are capable of counting in either the up direction or the down direction through any count sequence.

A simple three bit Up/Down synchronous counter can be built using flip flops giving a maximum count of zero(000), advancing through 001, 010 to seven(111) and back to zero(000) again.

**CIRCUIT DIAGRAM:**



**MAIN VERILOG CODE:**

module updown\_counter(input clk,reset,updown, output [2:0] counter);

reg [2:0] updowncounter;

always @(posedge clk or posedge reset)

begin

if(reset==1)

begin

updowncounter <= 3'b0;

end

else

if(updown==1)

begin

updowncounter <= updowncounter + 1'b1;

end

else

begin

updowncounter <= updowncounter - 1'b1;

end

end

assign counter = updowncounter;

endmodule

**TEST BENCH FILE:**

module testbench\_updown();

reg clk,reset,updown;

wire [2:0] counter;

updown\_counter fun(clk, reset, updown, counter);

initial begin

$dumpfile("counter.vcd");

$dumpvars(0,testbench\_updown);

end

initial begin

clk=0;

repeat(96)

#5 clk=~clk;

end

initial begin

reset=1;

updown=0;

#20;

reset=0;

#200

updown=1;

end

endmodule

**SCREEN SHOT OF THE OUTPUT:**



