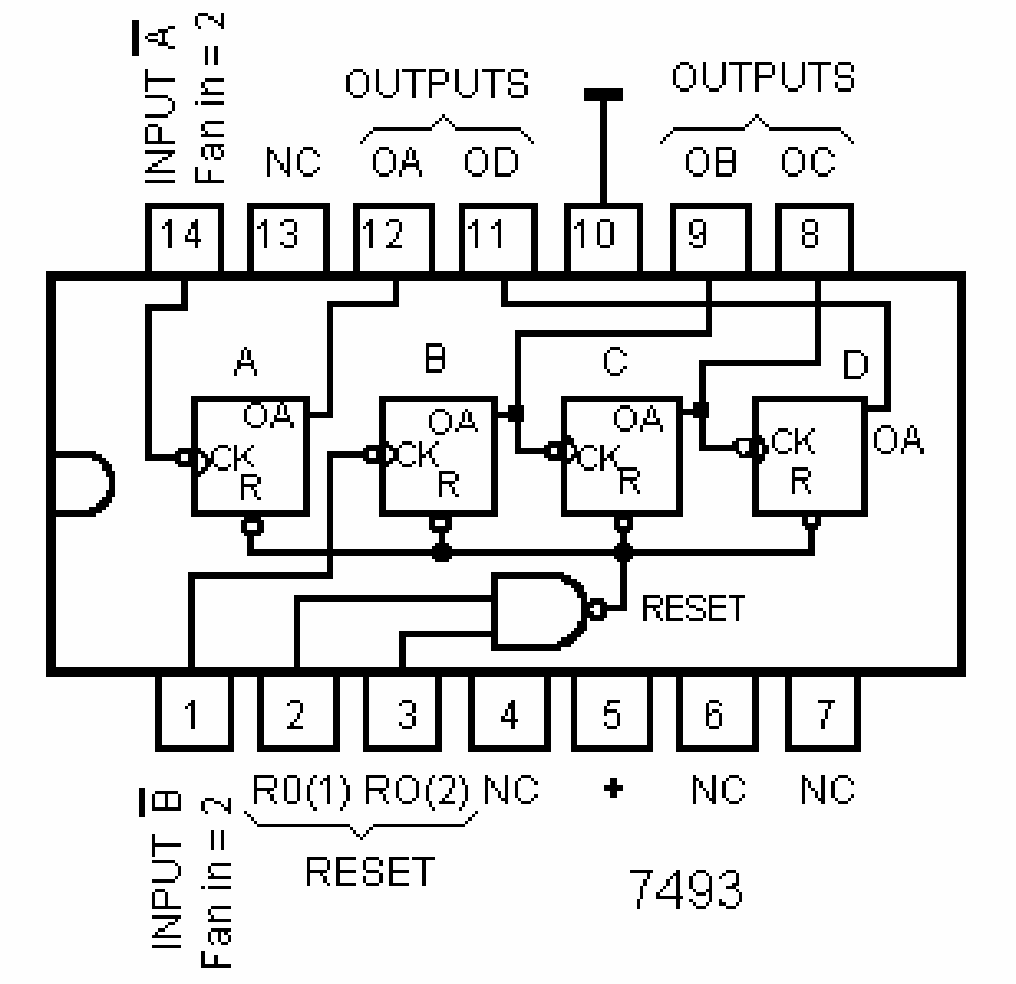
Name: Dhruv Maheshwari ID No: 2019A7PS0020U

**Experiment No. 10**

**Counters**

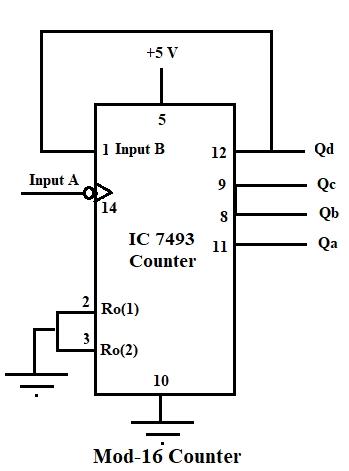
**Hardware runs**

**Components Required**

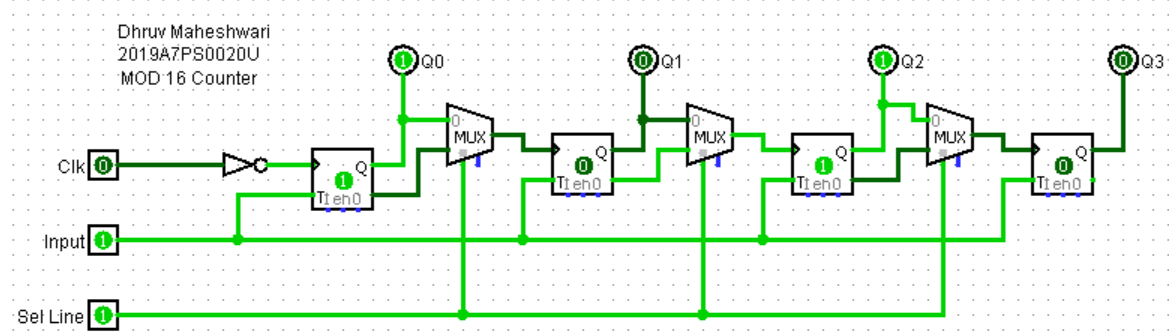
****

**7493-Binary Counter**

**Run 1: Mod -16 Counter (25 mins)**

IC 7493 is made to operate as a 4-bit counter by wiring the external terminal as shown below. Input A is connected to the pulser that generates the pulses which could either be a toggle switch or a pulse generator. Output QA is connected to input B. The reset pin R1 and R2 are connected to ground. 

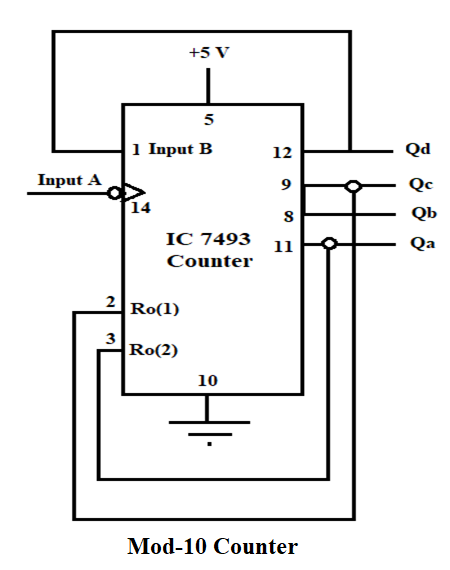
**Diagram**

****

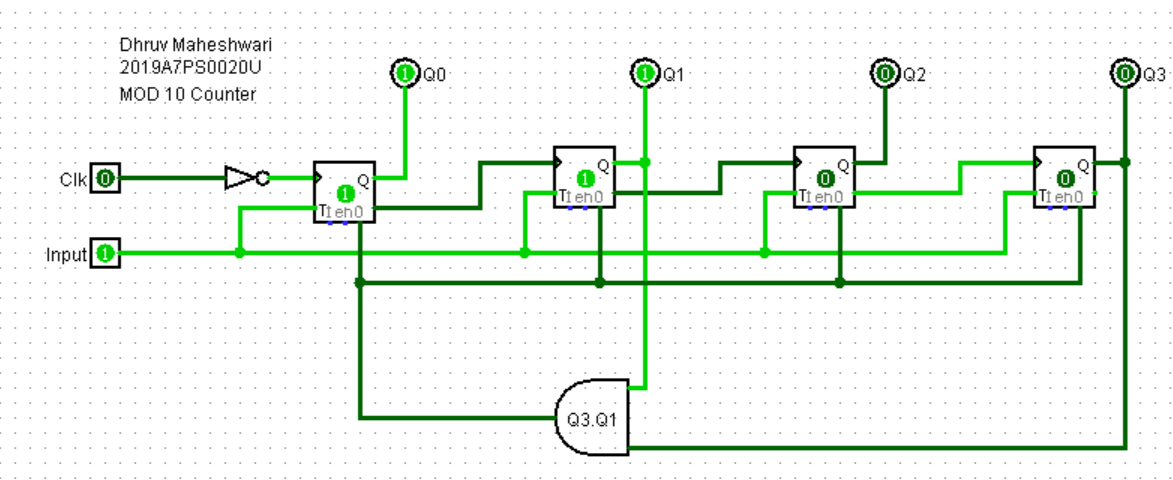
**Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **No. Of pulses(n)** | **Count- QDQCQBQA** | **No. Of pulses(n)** | **Count- QDQCQBQA** |
| 0 | 0000 | 9 | 1001 |
| 1 | 0001 | 10 | 1010 |
| 2 | 0010 | 11 | 1011 |
| 3 | 0011 | 12 | 1100 |
| 4 | 0100 | 13 | 1101 |
| 5 | 0101 | 14 | 1110 |
| 6 | 0110 | 15 | 1111 |
| 7 | 0111 | 16 | 0000 |
| 8 | 1000 | 17 | 0001 |

**Run 2: BCD counter (Mod 10) (20 mins)**

A BCD counter is one that counts from 0000 to 1001. The R1 and R2 are reset pins. Connecting both to 1 causes the counter to automatically reset to the 0000 state. This fact can be used to make IC 7493 to count from 0 to a variety of final counts without any external logic circuitry

**Diagram**

****

**Truth Table**

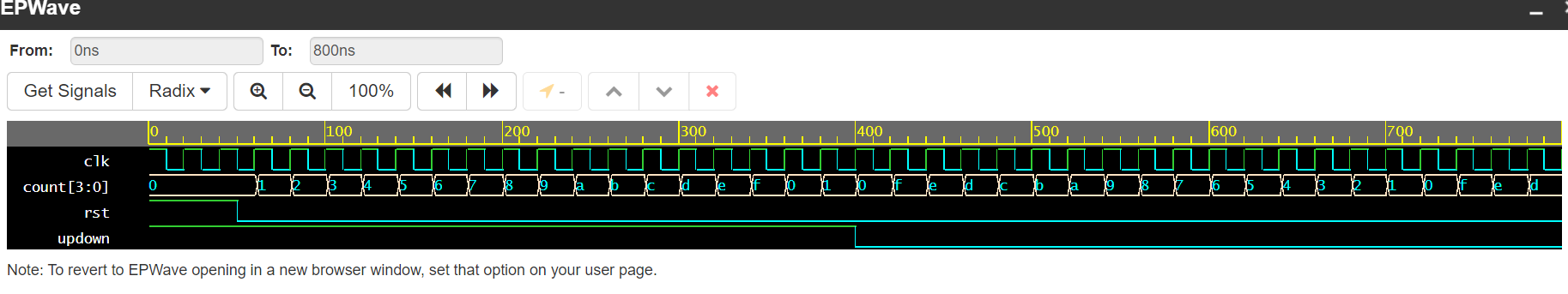
|  |  |  |  |
| --- | --- | --- | --- |
| **No. Of pulses(n)** | **Count- QDQCQBQA** | **No. Of pulses(n)** | **Count- QDQCQBQA** |
| 0 | 0000 | 6 | 0110 |
| 1 | 0001 | 7 | 0111 |
| 2 | 0010 | 8 | 1000 |
| 3 | 0011 | 9 | 1001 |
| 4 | 0100 | 10 | 0000 |
| 5 | 0101 | 11 | 0001 |

**Software runs**

**Run 3: (20 Mins)**

Write verilog code and testbench for 4-bit synchronous UP/DOWN counter. Code is given below for your reference.

<https://www.edaplayground.com/x/rPUh>



*module counter\_1 (input clk, rst, updown, output reg [3:0] count);*

*always @ (posedge clk, posedge rst) begin*

*if (rst)*

*count <= 4'b0000;*

*else if (updown)*

*count <= #1 count + 1'b1;*

*else*

*count <= #1 count - 1'b1;*

*end*

*endmodule*

*// Test Bench*

*module counter\_tb;*

*reg clk,rst,updown;*

*wire [3:0]count;*

*initial begin*

*clk = 1'b1;*

*repeat (80) # 10 clk = ~clk;*

*$stop;*

*end*

*initial begin*

*rst=1'b1;*

*repeat (1) # 50 rst = ~rst;*

*end*

*initial begin*

*#00 updown =1'b1;*

*#400 updown =1'b0;*

*end*

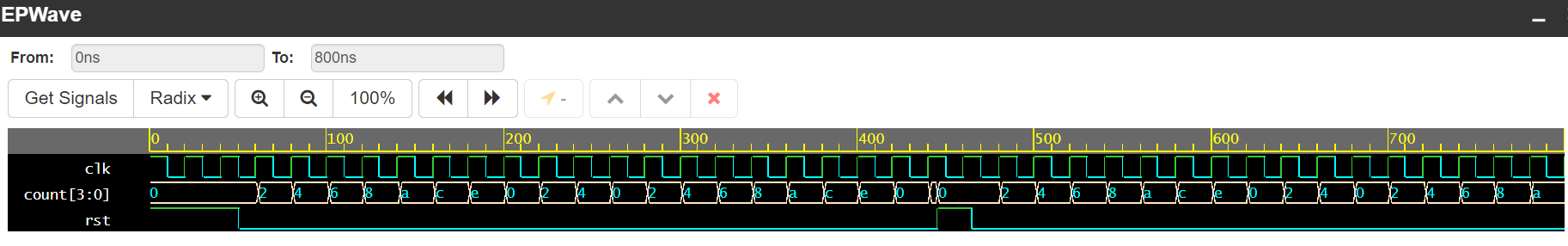
*counter\_1 ud (clk,rst,updown,count);*

*endmodule*

**Run 4: (20 Mins)**

Write Verilog code and testbench for even counter. Code is given for your reference.

<https://www.edaplayground.com/x/fBkp>



*module counter\_2 (input clk, rst, output reg [4:0] cnt);*

*always @ (posedge clk, posedge rst) begin*

*if (rst)*

*cnt <= 5'b00000;*

*else if (cnt == 5'd20)*

*cnt <= #1 5'b00000;*

*else*

*cnt <= #1 cnt + 5'b00010;*

*end*

*endmodule*

//Test Bench

*module counter\_tb;*

*reg clk,rst;*

*wire [3:0]count;*

*initial begin*

*clk = 1'b1;*

*repeat (80) # 10 clk = ~clk;*

*$stop;*

*end*

*initial begin*

*rst=1'b1;*

*repeat (1) # 50 rst = ~rst;*

*#395 rst = 1'b1;*

*#20 rst = 1'b0;*

*end*

*counter\_2 even (clk,rst,count);*

*endmodule*