**8 bits Ring Counter (Behavioral Modeling)**

**Lab no# 05**

****

Spring 2022

CSE-308L Digital Systems Design lab

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Dr: Ma’am Madeha sheer**

**May** 22, 2022

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Objectives:**

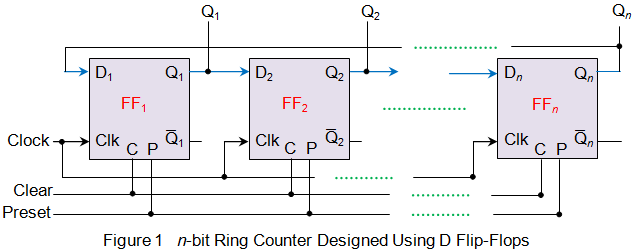
* To learn about the behavior of Ring counter.

**Ring Counter:**

Ring counter is a typical application of Shift register. Ring counter is almost same as the shift counter. The only change is that the output of the last flip-flop is connected to the input of the first flip-flop in case of ring counter but in case of shift resister it is taken as output. Except this all the other things are same.

No. of states in Ring counter = No. of flip-flop used

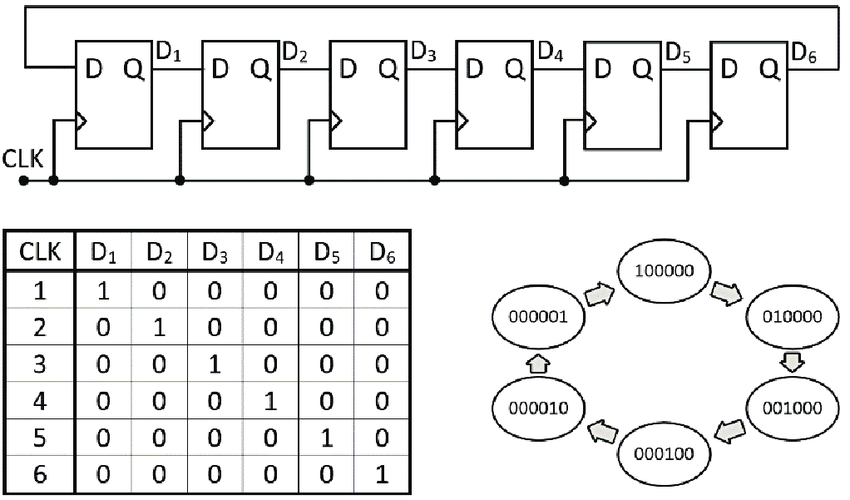
So, for designing 4-bit Ring counter we need 4 D flip flops.



**An n-bit Ring Counter**

In this diagram, we can see that the clock pulse (CLK) is applied to all the flip-flop simultaneously. Therefore, it is a Synchronous Counter.

Truth table for 6 bits Ring Counter:



Task**: Design an 8 bits Ring Counter.**

**Source Code:**

module Ring\_counter\_8bits(count,clk\_100Mhz,reset);

input clk\_100Mhz,reset;

output [7:0]count;

reg [7:0]count;

clk\_divider cd(clk\_1Mhz,clk\_100Mhz,reset);

always @(posedge clk\_1Mhz or negedge reset) //we can also keep clk for sensitivity only.

begin

if(~reset)

count<=8'b10000000;

else

begin

count<=count<<1;

count[0]<=count[7];

//unblocking assignments. in blocking case ring counter print 0 continuously.

end

end

endmodule

As sporton6 clock’s (v10) frequency is 100 MHz, which is not detectable. So we modeled clock divider and instantiated it in ring counter.

**Source Code:**

module clk\_divider(clk\_1Mhz,clk\_100Mhz,reset);

input clk\_100Mhz,reset;

output reg clk\_1Mhz;

reg [26:0]c; //10Mhz=27 bits..

always @(negedge clk\_100Mhz)

begin

if(~reset)

begin

c=0;

clk\_1Mhz=1'b1;

//we equate clk\_1Mhz to 1 in reset so to run always block of counter also to become reset.

end

else

begin

c=c+1'b1;

if(c==50000000)

begin

clk\_1Mhz=~clk\_1Mhz;

c=0;

end

end

end

endmodule

**Note:** we can further decrease frequency of clock by increasing the value of c in if statement and vice versa.

**UCF File:**

# PlanAhead Generated physical constraints

NET "count[7]" LOC = P15;

NET "count[6]" LOC = P16;

NET "count[5]" LOC = N15;

NET "count[4]" LOC = N16;

NET "count[3]" LOC = U17;

NET "count[2]" LOC = U18;

NET "count[1]" LOC = T17;

NET "count[0]" LOC = T18;

NET "clk\_100Mhz" LOC = V10;

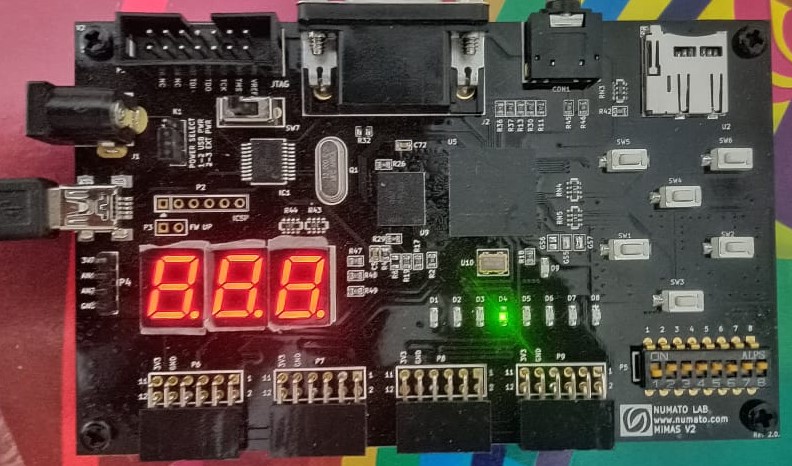
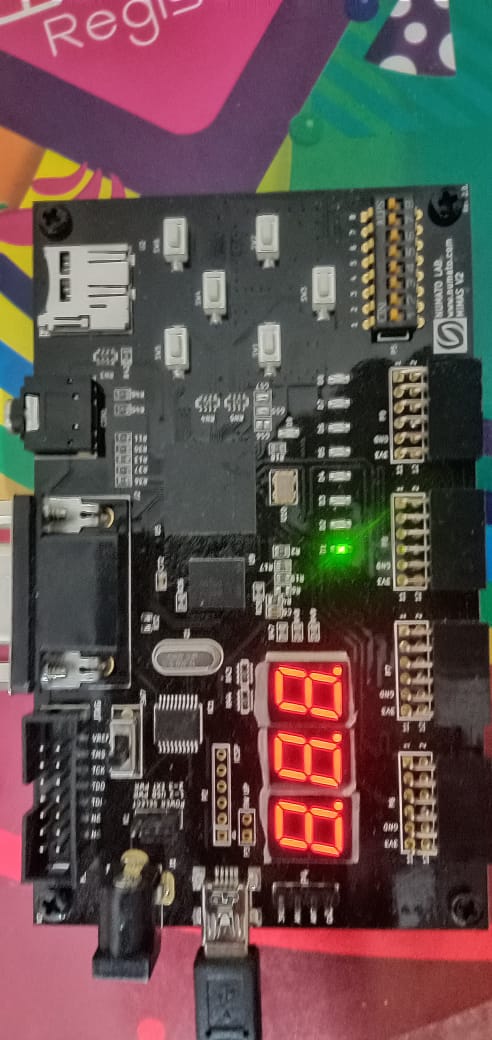
NET "reset" LOC = F17;

# PlanAhead Generated IO constraints

NET "clk\_100Mhz" PULLUP;

NET "reset" PULLUP;

**Output:**



When Reset=1 the counter stop counting and set to 8’b1000000. Dip switch 1 is reset input (the dip switches are active low).

**8 bits Ripple Counter in ModelSim:**

**Source Code:**

module rc (clk, init, count);

input clk, init;

output [7:0] count;

reg [7:0] count;

always @(negedge init or posedge clk)

begin

if (init)

count <= 8'b10000000;

else

begin

count <= count << 1; //Shift Left (Fill with Zero)

count[0] <= count[7];

//Compare blocking assignment (=) VS non-blocking assignment (<=)

end

end

endmodule

**Test bench:**

module test\_rc (clock, reset, out);

output clock, reset;

output [7:0] out;

reg clock;

reg reset;

always

#5 clock = ~clock; //toggle clk every 5 time units

initial

begin

clock = 1'b1; //set clk to 1

reset = 1'b1;

#12

reset = 1'b0;

#300 $finish; //terminate the simulation

end

initial

$monitor($time, ": Reset=%b ,Output = %b",reset,out);

endmodule

**Top Level:**

module top\_rc;

wire clk, init;

wire [7:0] count;

rc rc0 (clk, init, count);

test\_rc t\_rc0 (clk, init, count);

endmodule

**Output:**

