**VHDL Code for MOD N Counter**

library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity modN\_counter is

generic(n: integer := 128);

port(

clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

dout : out STD\_LOGIC\_VECTOR(6 downto 0)

);

end modN\_counter;

architecture A\_modN of modN\_counter is

begin

counter : process (clk,reset) is

variable m : integer range 0 to (n-1) := 0;

begin

if (reset='1') then

m := 0;

--elsif (rising\_edge (clk)) then

elsif(clk'event and clk='1') then

m := m + 1;

end if;

if (m=(n)) then

m := 0;

end if;

dout<= conv\_std\_logic\_vector (m,7);

end process counter;

end A\_modN;

**VHDL Test Bench**

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--USE ieee.numeric\_std.ALL;

ENTITY MOD\_TB IS

END MOD\_TB;

ARCHITECTURE behavior OF MOD\_TB IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT modN\_counter

PORT(

clk : IN std\_logic;

reset : IN std\_logic;

dout : OUT std\_logic\_vector(6 downto 0)

);

END COMPONENT;

--Inputs

signal clk : std\_logic := '0';

signal reset : std\_logic := '0';

--Outputs

signal dout : std\_logic\_vector(6 downto 0);

-- Clock period definitions

constant clk\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: modN\_counter PORT MAP (

clk => clk,

reset => reset,

dout => dout

);

-- Clock process definitions

clk\_process :process

begin

clk <= '0';

wait for clk\_period/2;

clk <= '1';

wait for clk\_period/2;

end process;

-- Stimulus process

stim\_proc: process

begin

-- hold reset state for 100 ns.

--wait for 100 ns;

reset<='1';

wait for 10 ns;

reset<='0';

-- insert stimulus here

wait;

end process;

END;

**Implementation on Spartan 6**

**MOD 128**

library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity modN\_counter is

generic(n: integer := 128);

port(

clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

dout : out STD\_LOGIC\_VECTOR(6 downto 0)

);

end modN\_counter;

architecture A\_modN of modN\_counter is

signal temp:std\_logic\_vector(22 downto 0);

signal sclk:std\_logic;

begin

p11: process(clk,reset)

begin

if(reset='1')then temp<=(others=>'0');

elsif(clk'event and clk='1') then

temp<= temp + 1;

end if;

end process;

sclk<=temp(22);

counter : process (sclk,reset) is

variable m : integer range 0 to (n-1) := 0;

begin

if (reset='1') then

m := 0;

--elsif (rising\_edge (clk)) then

elsif(sclk'event and sclk='1') then

m := m + 1;

end if;

if (m=(n)) then

m := 0;

end if;

dout<= conv\_std\_logic\_vector (m,7);

end process counter;

end A\_modN;

**Implementation Constraint File**

NET "reset" LOC = "E4";

NET "CLK" LOC ="L15";

NET "dout<0>" LOC = "U18"; # Bank = 1, Pin name = IO\_L52N\_M1DQ15, Sch name = LD0

NET "dout<1>" LOC = "M14"; # Bank = 1, Pin name = IO\_L53P, Sch name = LD1

NET "dout<2>" LOC = "N14"; # Bank = 1, Pin name = IO\_L53N\_VREF, Sch name = LD2

NET "dout<3>" LOC = "L14"; # Bank = 1, Pin name = IO\_L61P, Sch name = LD3

NET "dout<4>" LOC = "M13"; # Bank = 1, Pin name = IO\_L61N, Sch name = LD4

NET "dout<5>" LOC = "D4"; # Bank = 1, Pin name = IO\_L61P, Sch name = LD3

NET "dout<6>" LOC = "P16"; # Bank = 1, Pin name = IO\_L61P, Sch name = LD3

#NET "dout<7>" LOC = "N12";