**Counter\_mod**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.std\_logic\_unsigned.all;

entity counter\_mod is

generic(modu : positive := 3);

Port ( clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

m : in STD\_LOGIC\_VECTOR (modu downto 0);

q : out STD\_LOGIC\_VECTOR (modu downto 0));

end counter\_mod;

architecture Behavioral of counter\_mod is

signal next\_state, output : std\_logic\_vector(modu downto 0) := x"0";

begin

process(clk, reset, m)

begin

if(rising\_edge(clk)) then

if(reset = '1') then

output <= x"0";

next\_state <= x"1";

elsif(next\_state = m-1) then

output <= next\_state;

next\_state <= x"0";

else

output <= next\_state;

next\_state <= next\_state + x"1";

end if;

end if;

end process;

q <= output;

end Behavioral;

**Counter\_mod\_tb**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.std\_logic\_unsigned.all;

use work.all;

entity counter\_tb is

end counter\_tb;

architecture Behavioral of counter\_tb is

signal clk\_tb, r : std\_logic;

signal max, q\_tb : std\_logic\_vector(3 downto 0);

begin

p0 : entity work.counter\_mod(behavioral) port map (clk\_tb, r, max, q\_tb);

process

begin

wait for 25ns;

max <= "1100";

r <= '0';

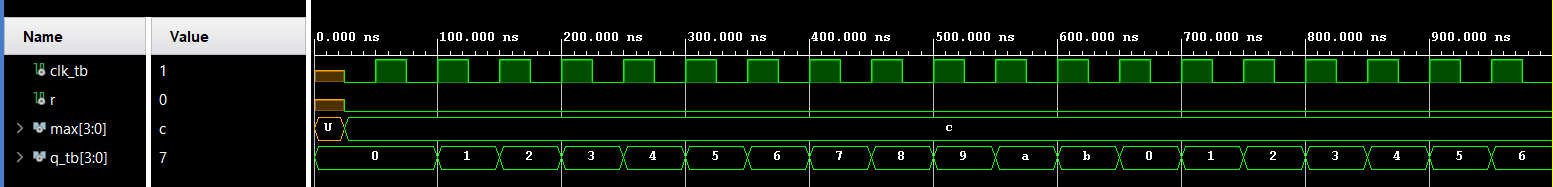
clk\_tb <= '0';

wait for 25ns;

clk\_tb <= '1';

end process;

end Behavioral;



**Counter simulation**

**Counter\_with\_floor**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.std\_logic\_unsigned.all;

entity counter\_with\_floor is

generic(modu : positive := 3);

Port ( clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

m : in STD\_LOGIC\_VECTOR (modu downto 0);

n : in STD\_LOGIC\_VECTOR (modu downto 0);

q : out STD\_LOGIC\_VECTOR (modu downto 0));

end counter\_with\_floor;

architecture Behavioral of counter\_with\_floor is

signal next\_state, output : std\_logic\_vector(modu downto 0) := x"0";

begin

process(clk, reset, m, n)

begin

if(rising\_edge(clk)) then

if(reset = '1') then

output <= n;

next\_state <= n + 1;

elsif(next\_state = m-1) then

output <= next\_state;

next\_state <= n;

elsif(next\_state = x"0") then

output <= n;

next\_state <= n + 1;

else

output <= next\_state;

next\_state <= next\_state + x"1";

end if;

end if;

end process;

q <= output;

end Behavioral;

**Counter\_with\_floor\_tb**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.std\_logic\_unsigned.all;

use work.all;

entity counter\_with\_floor\_tb is

end counter\_with\_floor\_tb;

architecture Behavioral of counter\_with\_floor\_tb is

signal clk\_tb, r : std\_logic;

signal max, min, q\_tb : std\_logic\_vector(3 downto 0);

begin

p1 : entity work.counter\_with\_floor(behavioral) port map (clk\_tb, r, max, min, q\_tb);

process

begin

wait for 25ns;

max <= "1100";

min <= "0110";

r <= '0';

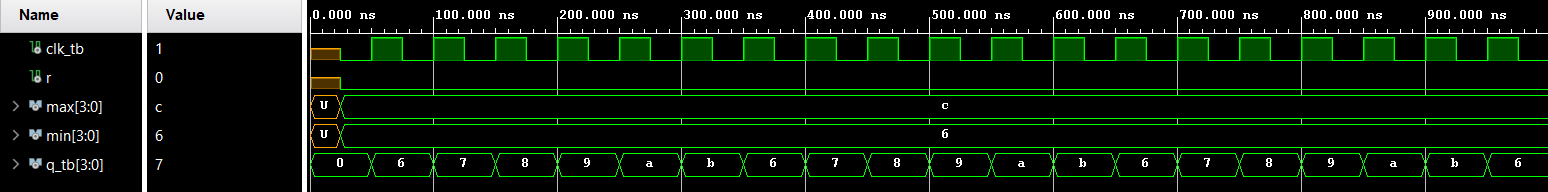
clk\_tb <= '0';

wait for 25ns;

clk\_tb <= '1';

end process;

end Behavioral;



**Floored Counter Simulation**