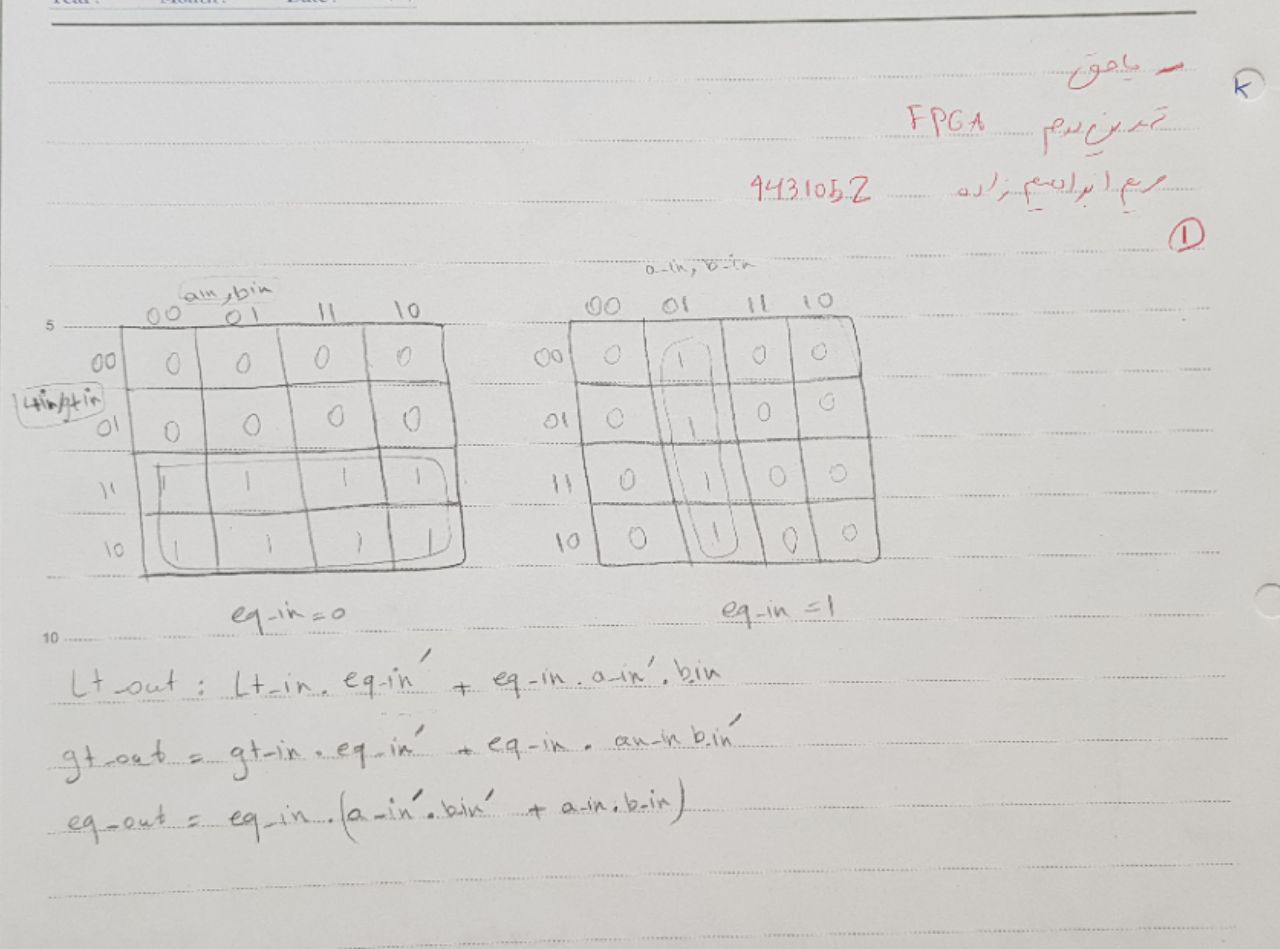
1.



2.

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity onebitcomparetor is

Port ( a\_in :in std\_logic;

b\_in : in std\_logic;

eq\_in : in std\_logic;

gt\_in : in std\_logic;

lt\_in : in std\_logic;

eq\_out : out std\_logic;

gt\_out : out std\_logic;

lt\_out : out std\_logic);

end onebitcomparetor;

architecture Behavioral of onebitcomparetor is

begin

eq\_out <= (eq\_in and(not a\_in) and (not b\_in)) or (eq\_in and a\_in and b\_in) ;

gt\_out <= (gt\_in) or (eq\_in and a\_in and (not b\_in));

lt\_out <= (lt\_in) or (eq\_in and b\_in and (not a\_in));

end Behavioral;

3.

**library IEEE;**

**use IEEE.STD\_LOGIC\_1164.ALL;**

**entity fourbitcomparetor is**

**Port (**

**a : in std\_logic\_vector(3 downto 0);**

**b : in std\_logic\_vector(3 downto 0);**

**eq\_in : in std\_logic;**

**gt\_in : in std\_logic;**

**lt\_in : in std\_logic;**

**greater : out std\_logic;**

**smaller : out std\_logic;**

**equal : out std\_logic**

**);**

**end fourbitcomparetor;**

**architecture Behavioral of fourbitcomparetor is**

**component onebitcomparetor is**

**Port ( a\_in :in std\_logic;**

**b\_in : in std\_logic;**

**eq\_in : in std\_logic;**

**gt\_in : in std\_logic;**

**lt\_in : in std\_logic;**

**eq\_out : out std\_logic;**

**gt\_out : out std\_logic;**

**lt\_out : out std\_logic);**

**end component onebitcomparetor;**

**signal sig : std\_logic\_vector(8 downto 0);**

**begin**

**u1: onebitcomparetor port map(a(3), b(3), eq\_in, gt\_in, lt\_in,sig(0),sig(1),sig(2));**

**u2: onebitcomparetor port map(a(2), b(2) ,sig(0),sig(1),sig(2),sig(3),sig(4),sig(5));**

**u3: onebitcomparetor port map(a(1), b(1) ,sig(3),sig(4),sig(5),sig(6),sig(7),sig(8));**

**u4: onebitcomparetor port map(a(0), b(0) ,sig(6),sig(7),sig(8),equal,greater,smaller);**

**end Behavioral;**

**nbit**

**library IEEE;**

**use IEEE.STD\_LOGIC\_1164.ALL;**

**--suppose n is 4**

**entity nbit is**

**generic (N : integer := 4);**

**Port ( a : in std\_logic\_vector(15 downto 0);**

**b : in std\_logic\_vector(15 downto 0);**

**greater : out std\_logic;**

**smaller : out std\_logic;**

**equal : out std\_logic**

**);**

**end nbit;**

**architecture Behavioral of nbit is**

**component fourbitcomparetor is**

**Port (**

**a : in std\_logic\_vector(3 downto 0);**

**b : in std\_logic\_vector(3 downto 0);**

**eq\_in : in std\_logic;**

**gt\_in : in std\_logic;**

**lt\_in : in std\_logic;**

**greater : out std\_logic;**

**smaller : out std\_logic;**

**equal : out std\_logic**

**);**

**end component fourbitcomparetor;**

**signal sig : std\_logic\_vector(14 downto 0):="000000000000001";**

**begin**

**g\_GENERATE\_FOR: for i in 0 to N-1 generate**

**m : fourbitcomparetor port map(**

**a => a((N-i)\*4 -1 downto (N-i)\*4 -1 - 3)**

**,b => b((N-i)\*4 -1 downto (N-i)\*4 -1 - 3)**

**,eq\_in => sig(i\*3)**

**,gt\_in => sig(i\*3+1)**

**,lt\_in => sig(i\*3+2)**

**,greater => sig(i\*3+3)**

**,smaller => sig(i\*3+4)**

**,equal => sig(i\*3+5)**

**);**

**end generate ;**

**greater <= sig(12);**

**smaller <= sig(13);**

**equal <= sig(14);**

**end Behavioral;**

**tetsbench**

**library IEEE;**

**use IEEE.STD\_LOGIC\_1164.ALL;**

**entity testbench is**

**-- Port ( );**

**end testbench;**

**architecture Behavioral of testbench is**

**component nbit is**

**generic (N : integer := 4);**

**Port ( a : in std\_logic\_vector(15 downto 0);**

**b : in std\_logic\_vector(15 downto 0);**

**greater : out std\_logic;**

**smaller : out std\_logic;**

**equal : out std\_logic);**

**end component nbit;**

**signal asig,bsig : std\_logic\_vector(15 downto 0):="0000000000000000";**

**signal gt,lt,eq : std\_logic;**

**begin**

**m : nbit port map(asig,bsig,gt,lt,eq);**

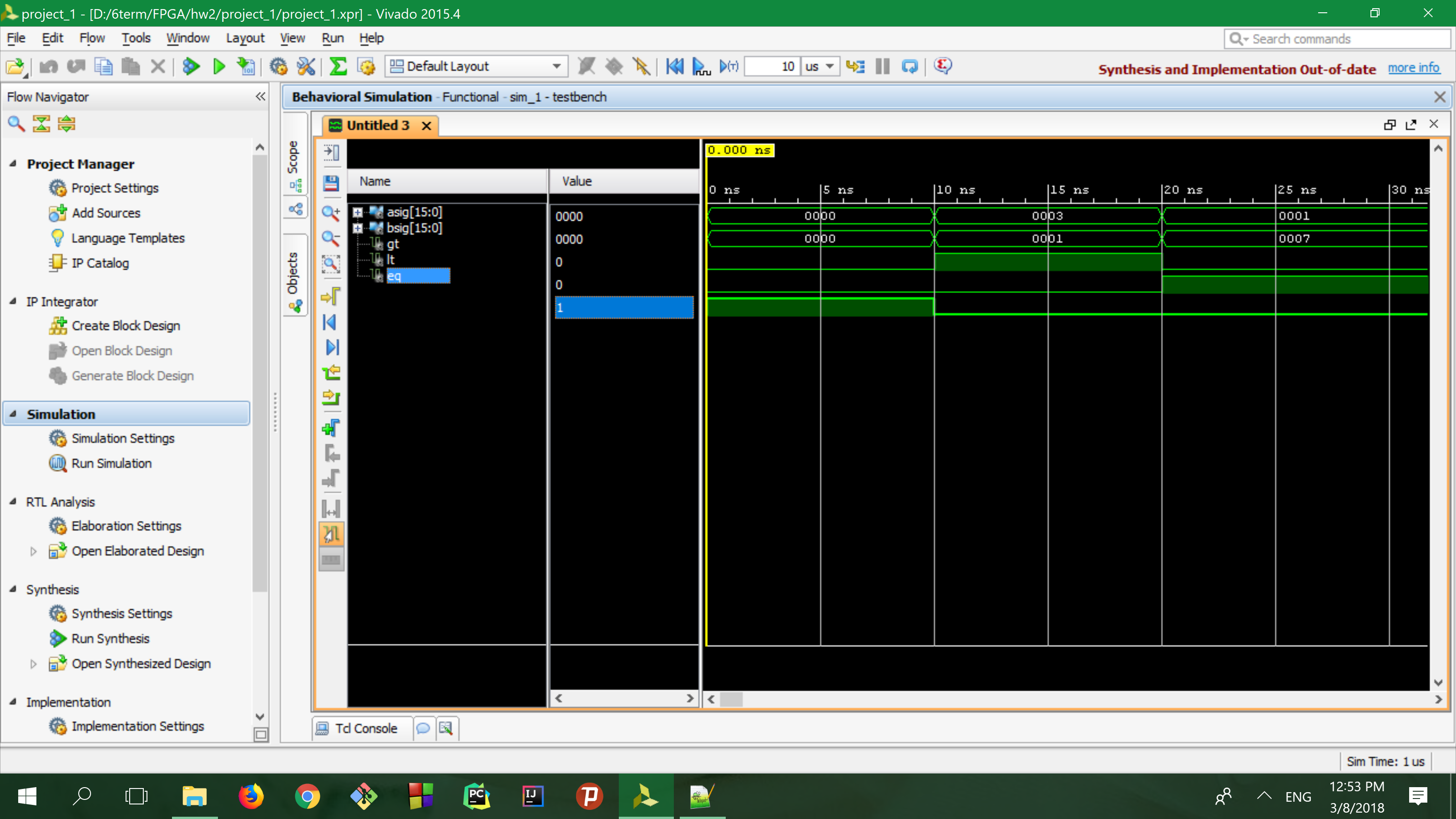
**asig <= "0000100000000011" after 10ns,**

**"0000000000000001" after 20ns;**

**bsig <= "0001000000000001" after 10ns,**

**"0000000000000111" after 20ns;**

**end Behavioral;**



4.

This is a mux.

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

Entity circuit is

port ( i0, i1, i2, i3, a, b : in std\_logic;

q : out std\_logic);

End circuit;

Architecture circuit of circuit is

signal not\_a : std\_logic;

signal not\_b : std\_logic;

signal m0,m1,m2,m3 : std\_logic;

Begin

not\_a <= not a;

not\_b <= not b;

m0 <= not\_a and not\_b and i0;

m1 <= not\_a and b and i1;

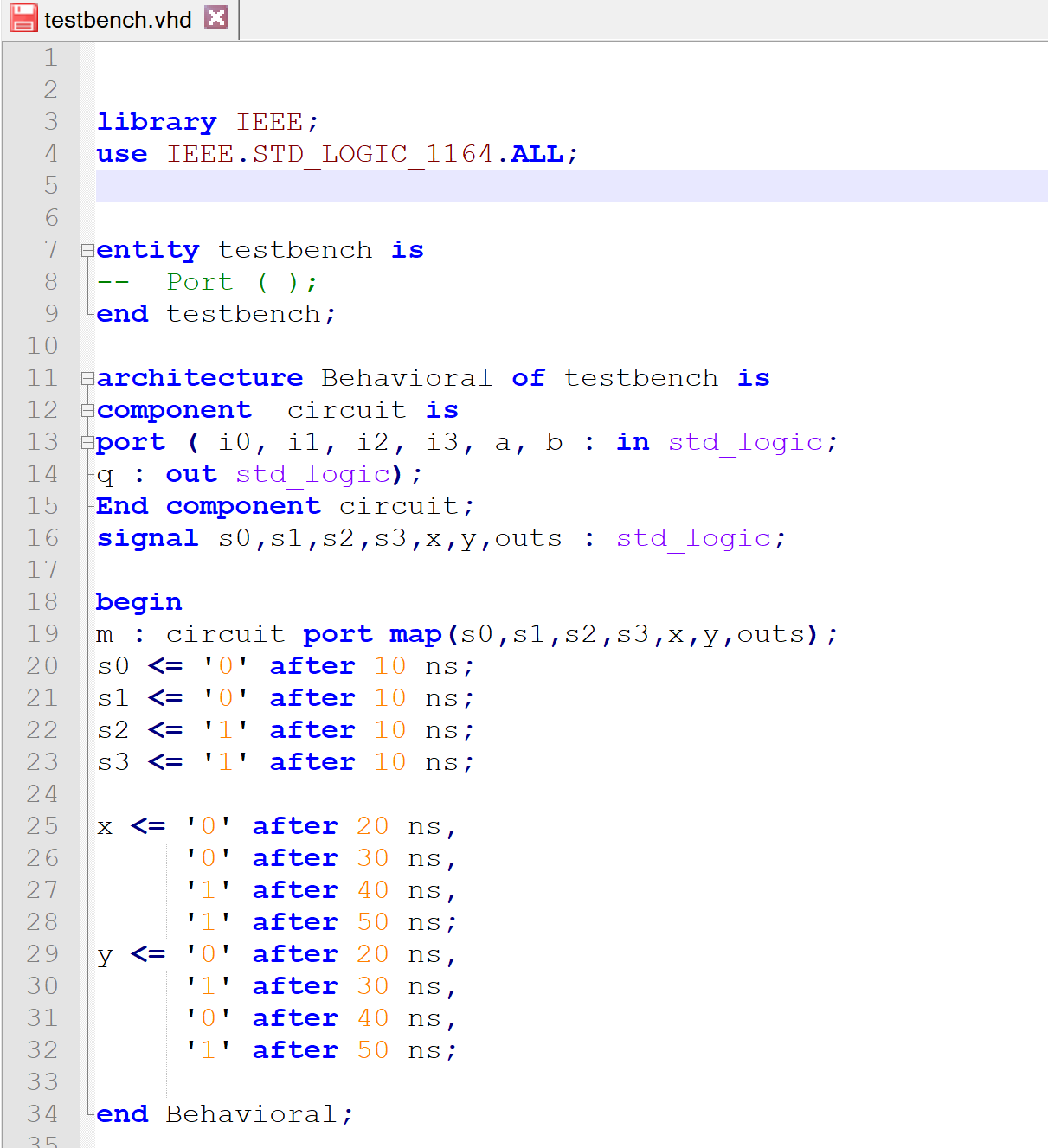
m2 <= a and not\_b and i2;

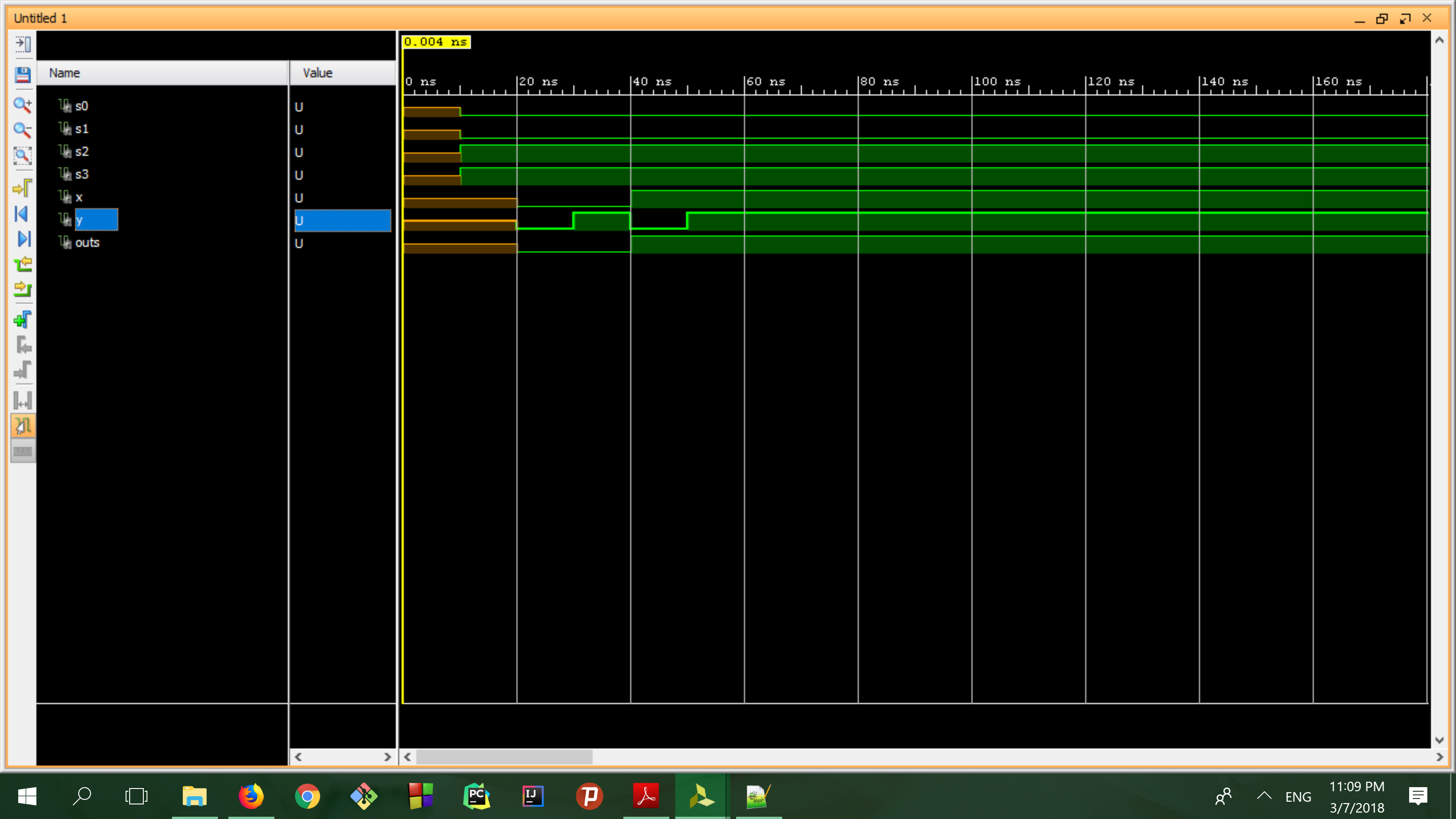
m3 <= a and b and i3;

q <= m0 or m1 or m2 or m3;

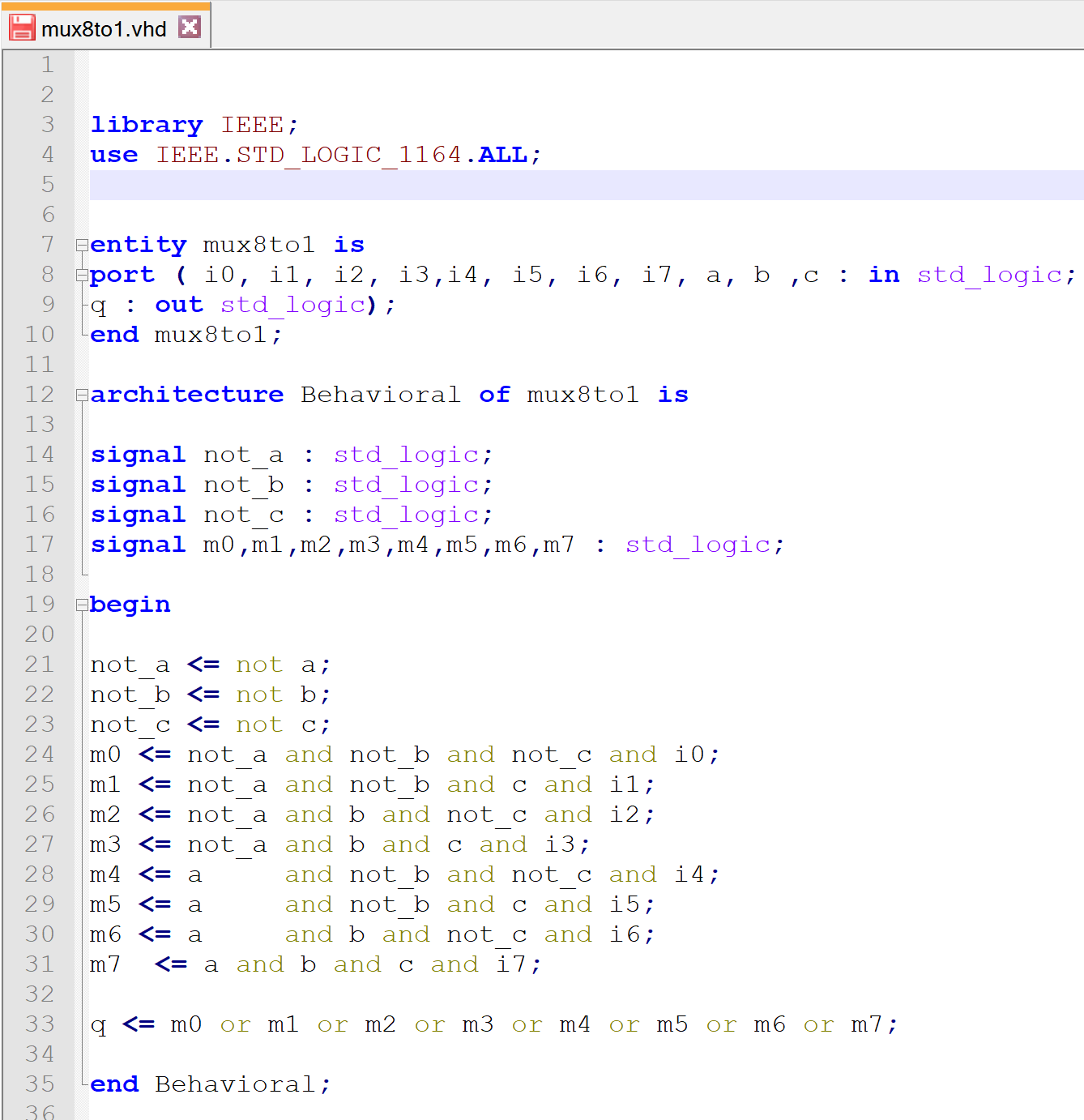
End circuit;

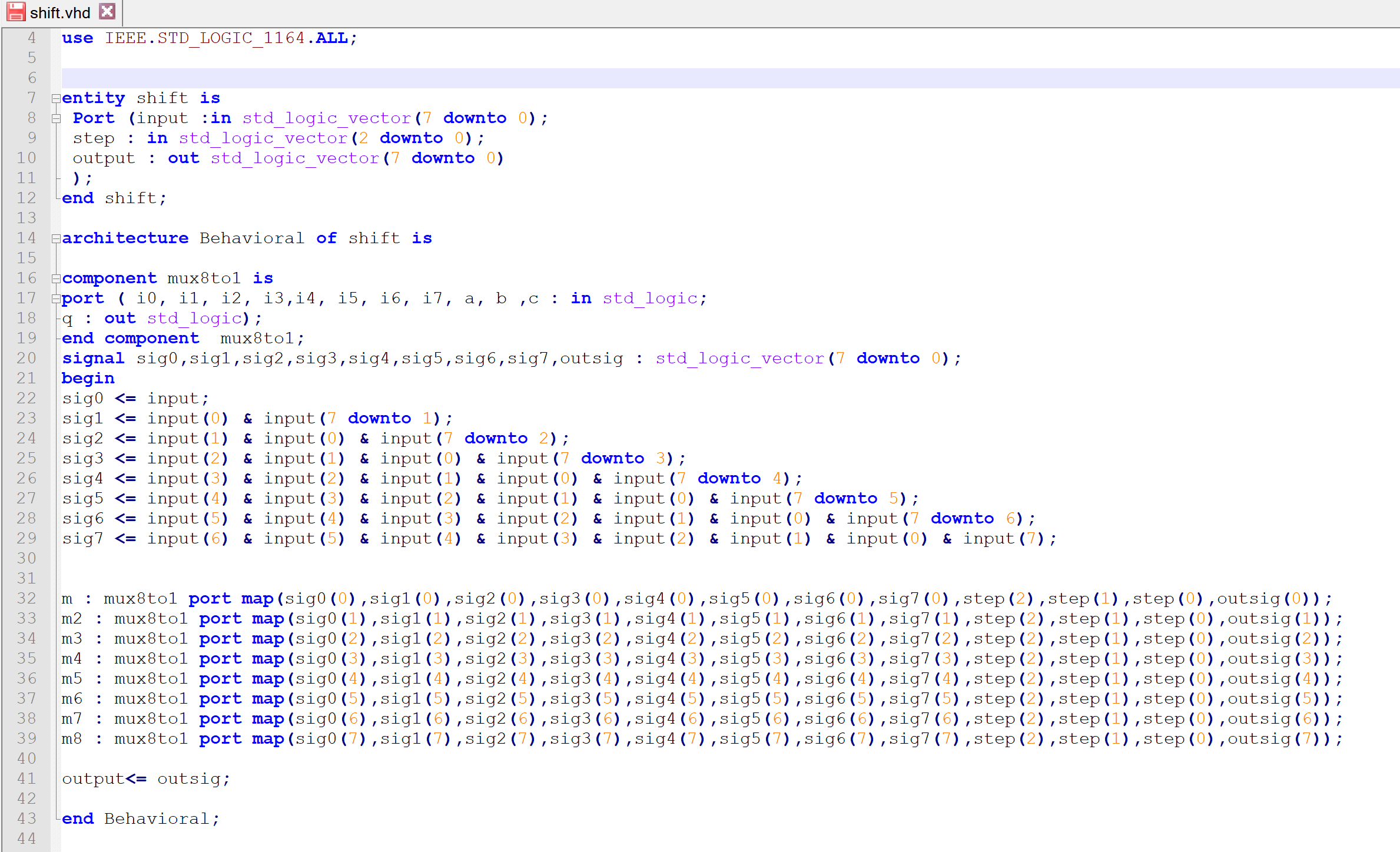
-- this module is a mux

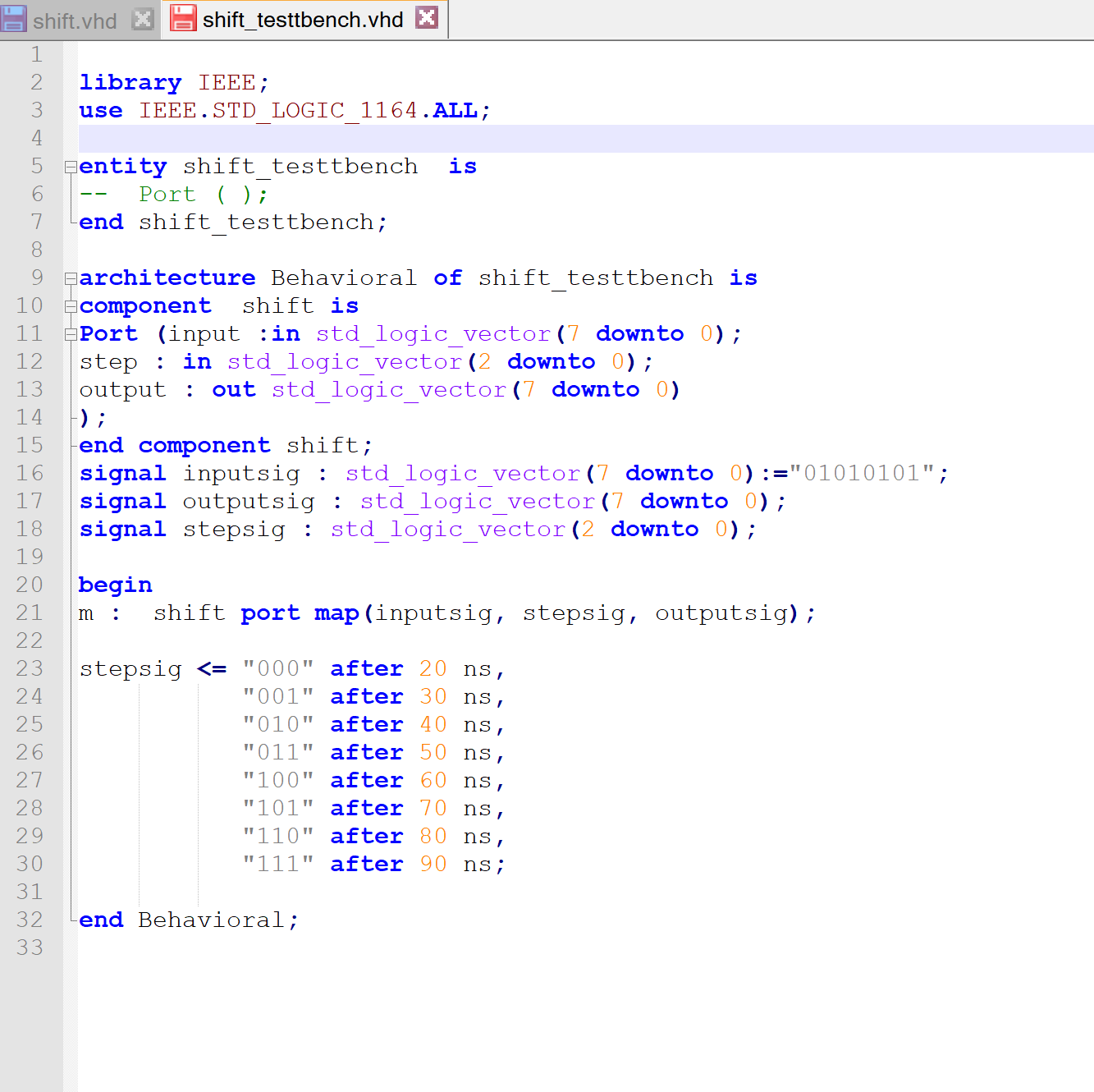


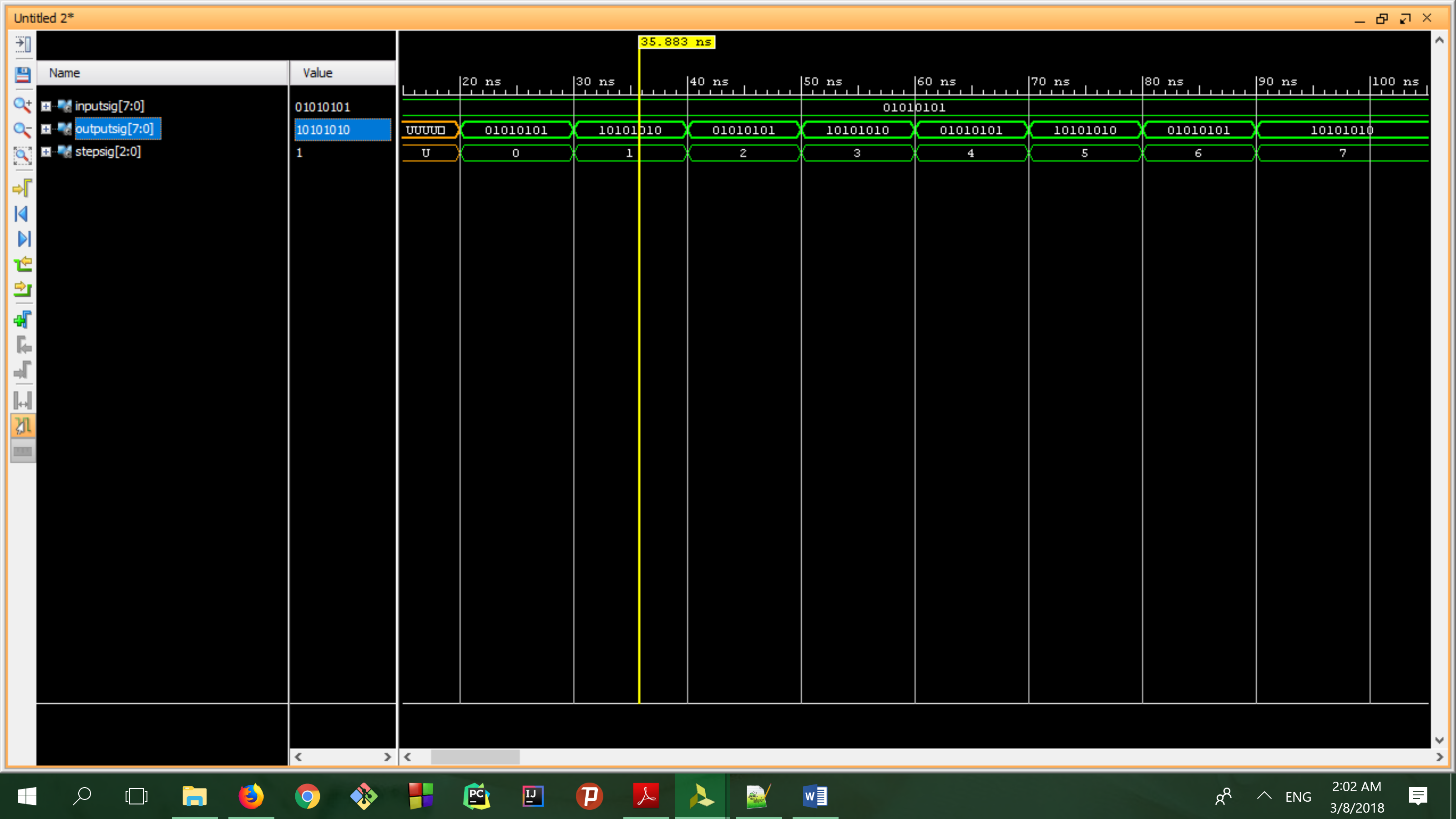


**5.**









**Emtiazi.**

**library IEEE;**

**use IEEE.STD\_LOGIC\_1164.ALL;**

**entity onehot is**

**Port (input :in STD\_LOGIC\_VECTOR(7 downto 0);**

**valid:out STD\_LOGIC;**

**Output:out STD\_LOGIC\_VECTOR(2 downto 0) );**

**end onehot;**

**architecture Behavioral of onehot is**

**signal s : std\_logic\_vector(5 downto 0);**

**begin**

**s(0) <= input(0) xor input(1);**

**s(1) <= input(2) xor input(3);**

**s(2) <= input(4) xor input(5);**

**s(3) <= input(6) xor input(7);**

**s(4) <= s(0) xor s(1);**

**s(5) <= s(2) xor s(3);**

**Valid <= s(4) xor s(5);**

**Output(2) <= input(7) or input(6) or input(5) or input(4);**

**Output(1) <= input(7) or input(6) or input(3) or input(2);**

**Output(0) <= input(7) or input(5) or input(3) or input(1);**

**end Behavioral;**

**test bench**

**library IEEE;**

**use IEEE.STD\_LOGIC\_1164.ALL;**

**entity testbench is**

**-- Port ( );**

**end testbench;**

**architecture Behavioral of testbench is**

**component onehot is**

**Port (input :in STD\_LOGIC\_VECTOR(7 downto 0);**

**valid:out STD\_LOGIC;**

**Output:out STD\_LOGIC\_VECTOR(2 downto 0) );**

**end component;**

**signal inputs :STD\_LOGIC\_VECTOR(7 downto 0);**

**signal valids :STD\_LOGIC;**

**signal outputs :STD\_LOGIC\_VECTOR(2 downto 0);**

**begin**

**m:onehot port map(inputs, valids ,outputs);**

**inputs <= "00000001" after 10 ns,**

**"00000010" after 20 ns,**

**"00000100" after 30 ns,**

**"00001000" after 40 ns,**

**"00010000" after 50 ns,**

**"00100000" after 60 ns,**

**"01000000" after 70 ns,**

**"10000000" after 80 ns,**

**"10000110" after 90 ns;**

**end Behavioral;**

