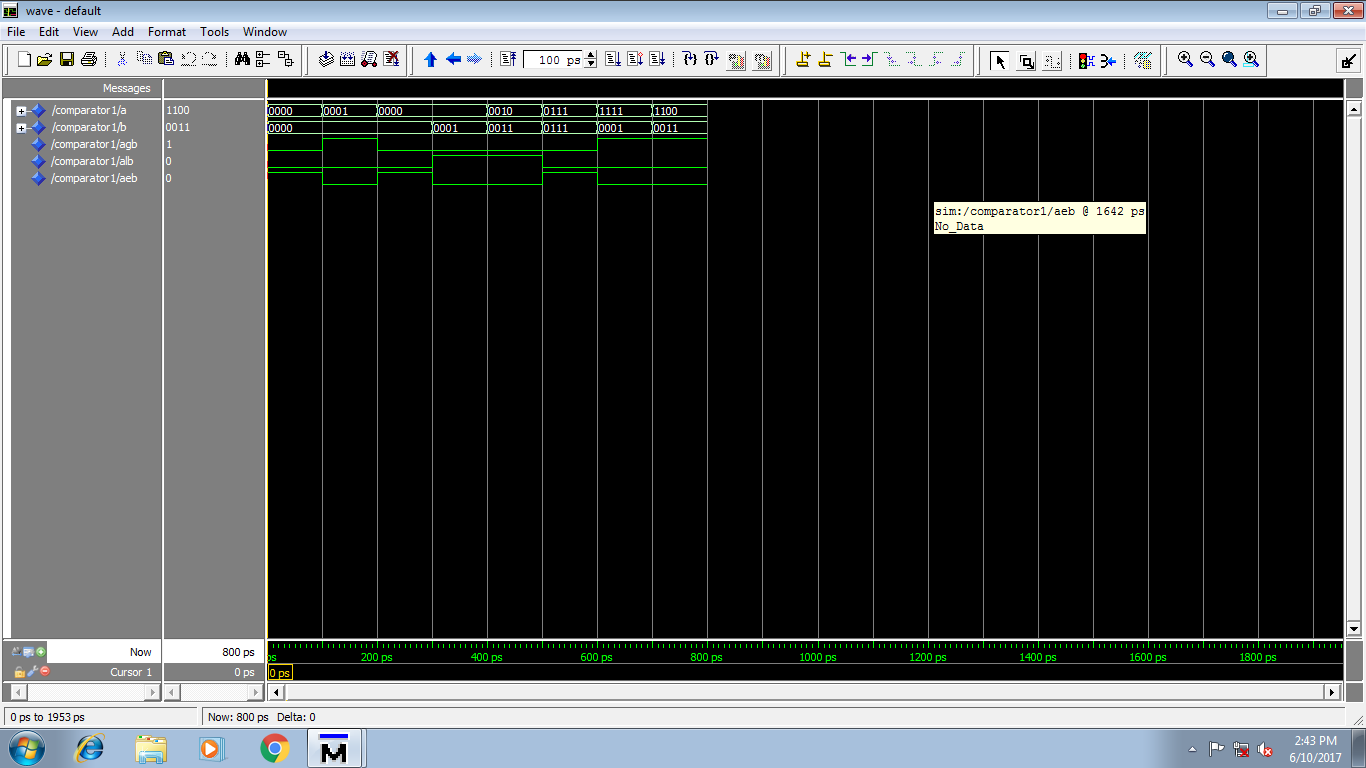
**4 bit Comparator Behavioral**



library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity comparator1 is

port(

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

agb : out std\_logic;

alb : out std\_logic;

aeb : out std\_logic

);

end comparator1;

architecture comp\_arch of comparator1 is

begin

process (a,b)

begin

if a>b then

agb <= '1';

alb <= '0';

aeb <= '0';

elsif a<b then

agb <= '0';

alb <= '1';

aeb <= '0';

else

aeb <= '1';

alb <= '0';

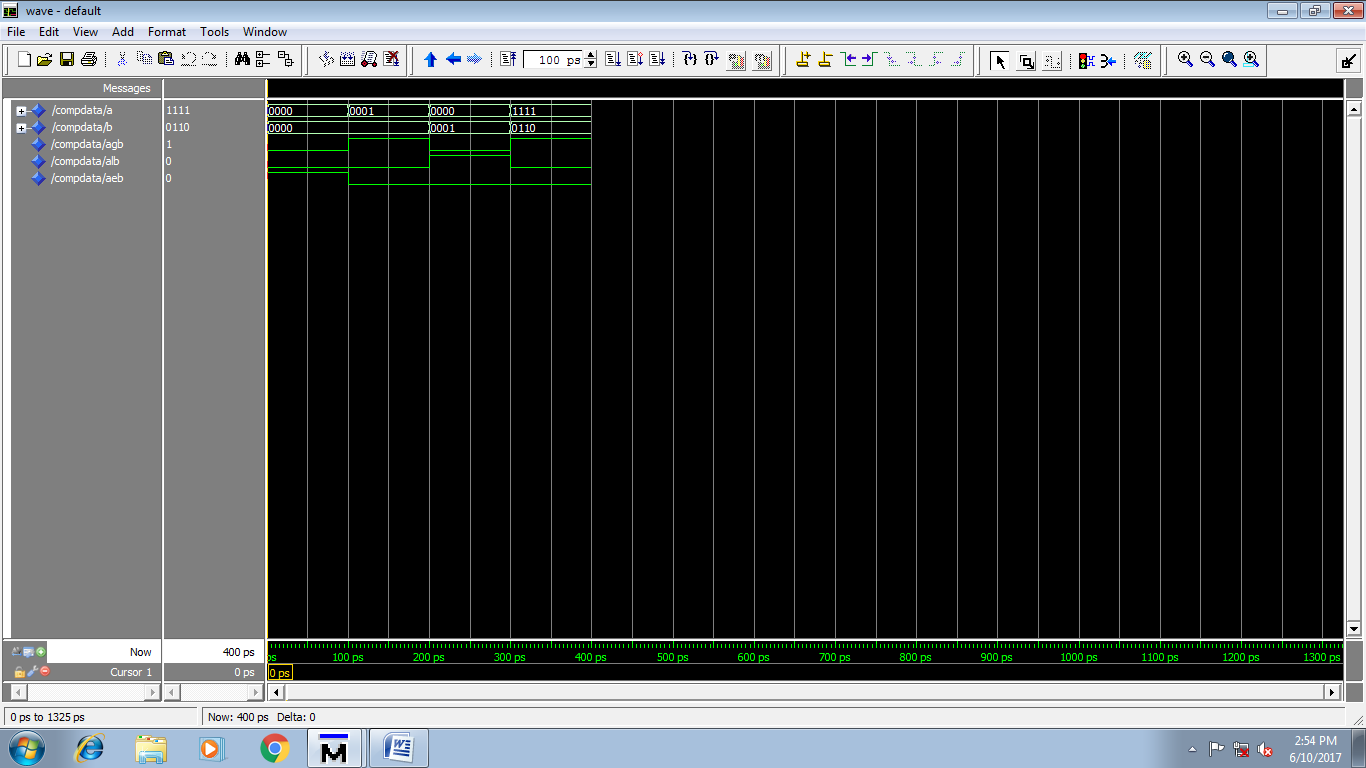
agb <= '0';

end if;

end process;

end comp\_arch;

**4 bit Comparator DataFlow**



library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity CompData is

port(

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

agb : out std\_logic;

alb : out std\_logic;

aeb : out std\_logic

);

end CompData;

architecture data\_arch of CompData is

begin

agb <= '1' when (a>b)

else '0';

alb <= '1' when (a<b)

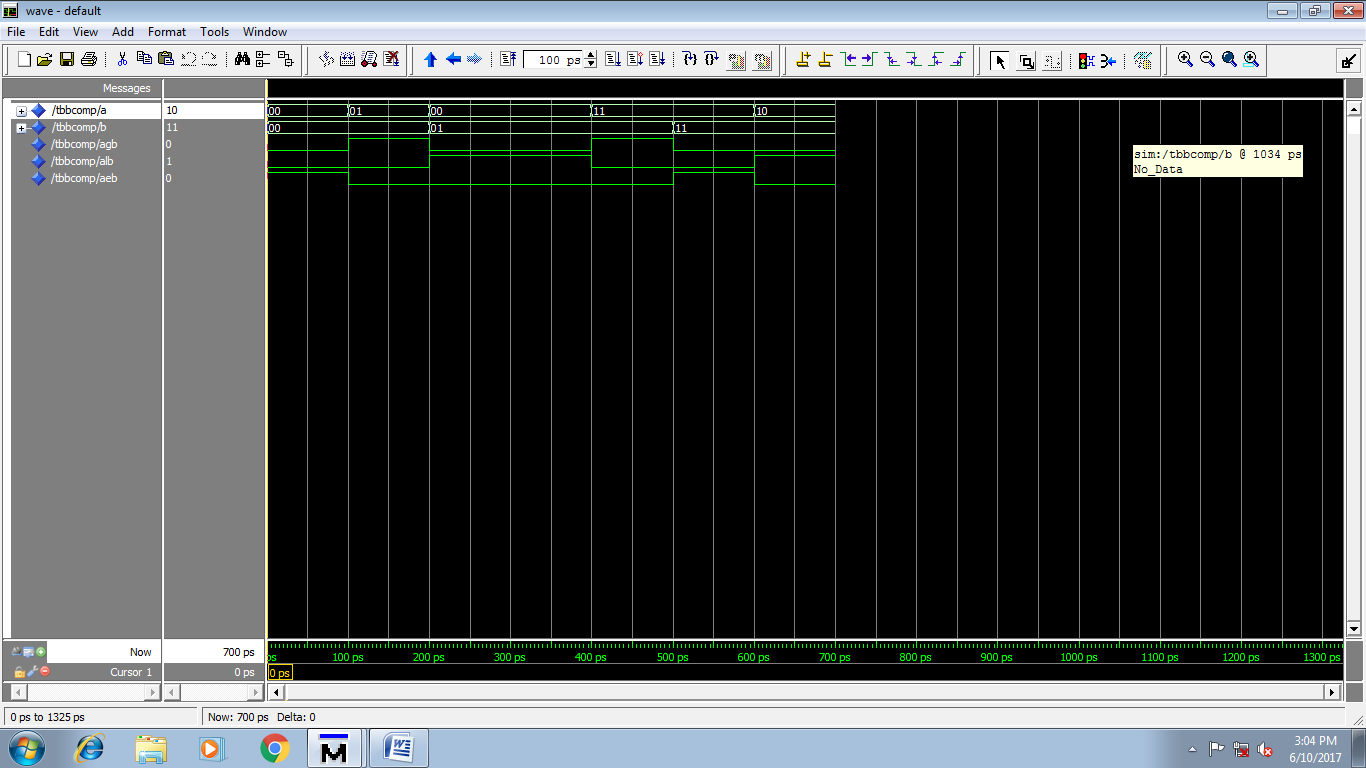
else '0';

aeb <= '1' when (a=b)

else '0';

end data\_arch;

**2-BIT Comparator Behavorial**



library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity TBBComp is

port(

a,b : in std\_logic\_vector(1 downto 0);

agb : out std\_logic;

alb : out std\_logic;

aeb : out std\_logic

);

end TBBComp;

architecture comp\_arch2 of TBBComp is

begin

process (a,b)

begin

if a>b then

agb <= '1';

alb <= '0';

aeb <= '0';

elsif a<b then

agb <= '0';

alb <= '1';

aeb <= '0';

else

aeb <= '1';

alb <= '0';

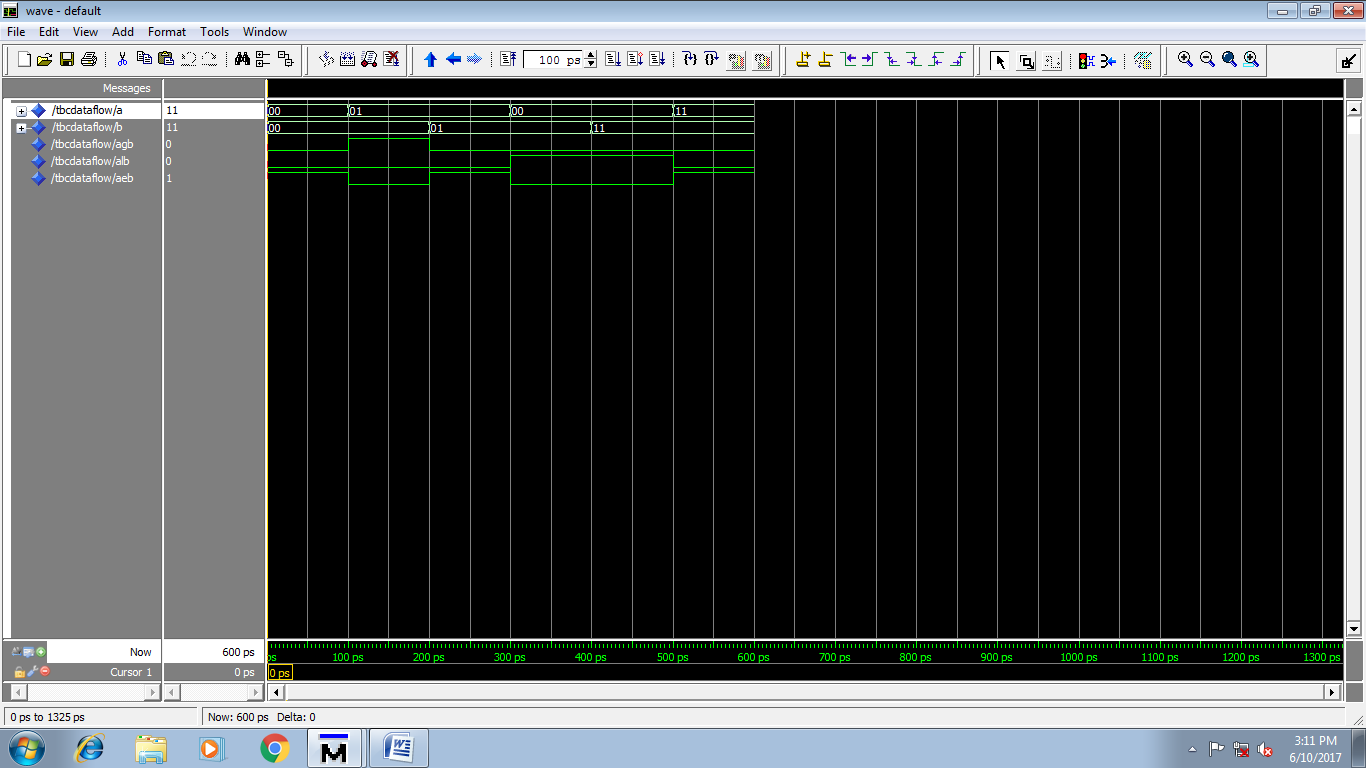
agb <= '0';

end if;

end process;

end comp\_arch2;

**2 bit comparator using DataFlow**



library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity TBCDataflow is

port(

a,b : in std\_logic\_vector(1 downto 0);

agb : out std\_logic;

alb : out std\_logic;

aeb : out std\_logic

);

end TBCDataflow;

architecture data\_arch4 of TBCDataflow is

begin

agb <= '1' when (a>b)

else '0';

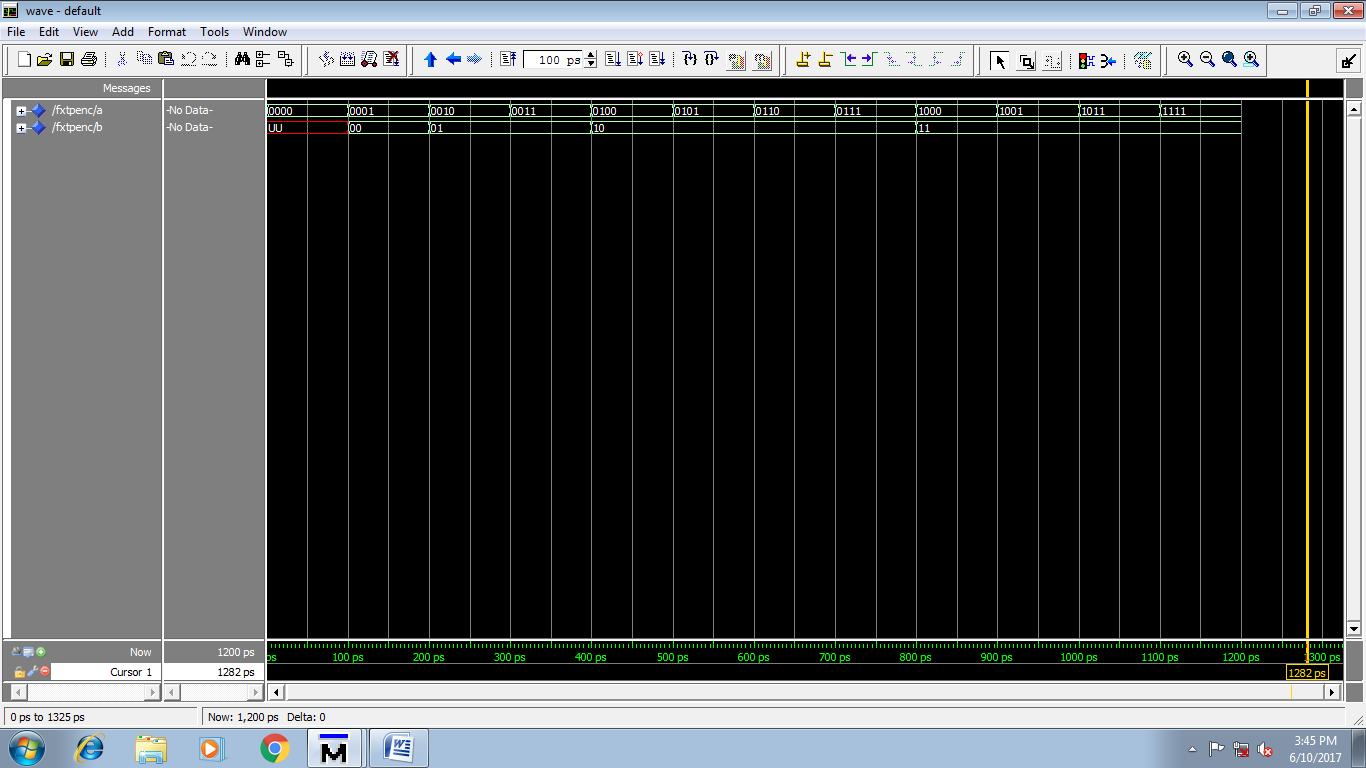
alb <= '1' when (a<b)

else '0';

aeb <= '1' when (a=b)

else '0';

end data\_arch4;

**Priority Encoder 4:2**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity FXTPEnc is

port(

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

b : out std\_logic\_vector(1 downto 0)

);

end FXTPEnc;

architecture data\_arch5 of FXTPEnc is

begin

process (a)

begin

if (a="0001") then

b <= "00";

elsif (a = "0010") then

b<= "01";

elsif (a>"0010" and a<"0100") then

b <= "01";

elsif (a = "0100") then

b<= "10";

elsif (a>"0100" and a<"1000") then

b <= "10";

elsif (a="1000" or a>"1000") then

b <= "11";

end if;

end process;

end data\_arch5;

**Priority Encoder 8:3**

**Incomplete code:**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity EXTPEnc is

port(

a : in std\_logic\_vector(7 downto 0);

b : out std\_logic\_vector(2 downto 0)

);

end EXTPEnc;

architecture data\_arch6 of EXTPEnc is

begin

process (a)

begin

if (a="00000001") then

b <= "000";

elsif (a = "00000010") then

b<= "001";

elsif (a>"00000010" and a<"00000100") then

b <= "001";

---c

elsif (a = "00000100") then

b<= "001";

elsif (a>"00000100" and a<"00001000") then

b <= "010";

---c

elsif (a = "00000010") then

b<= "010";

elsif (a>"00000010" and a<"00000100") then

b <= "010";

---

elsif (a = "00000010") then

b<= "011";

elsif (a>"00000010" and a<"00000100") then

b <= "011";

---

elsif (a = "00000010") then

b<= "100";

elsif (a>"00000010" and a<"00000100") then

b <= "100";

---

elsif (a = "00000010") then

b<= "101";

elsif (a>"00000010" and a<"00000100") then

b <= "101";

---

elsif (a = "00000010") then

b<= "110";

elsif (a>"00000010" and a<"00000100") then

b <= "110";

---

elsif (a = "00000010") then

b<= "111";

elsif (a>"00000010" and a<"00000100") then

b <= "111";

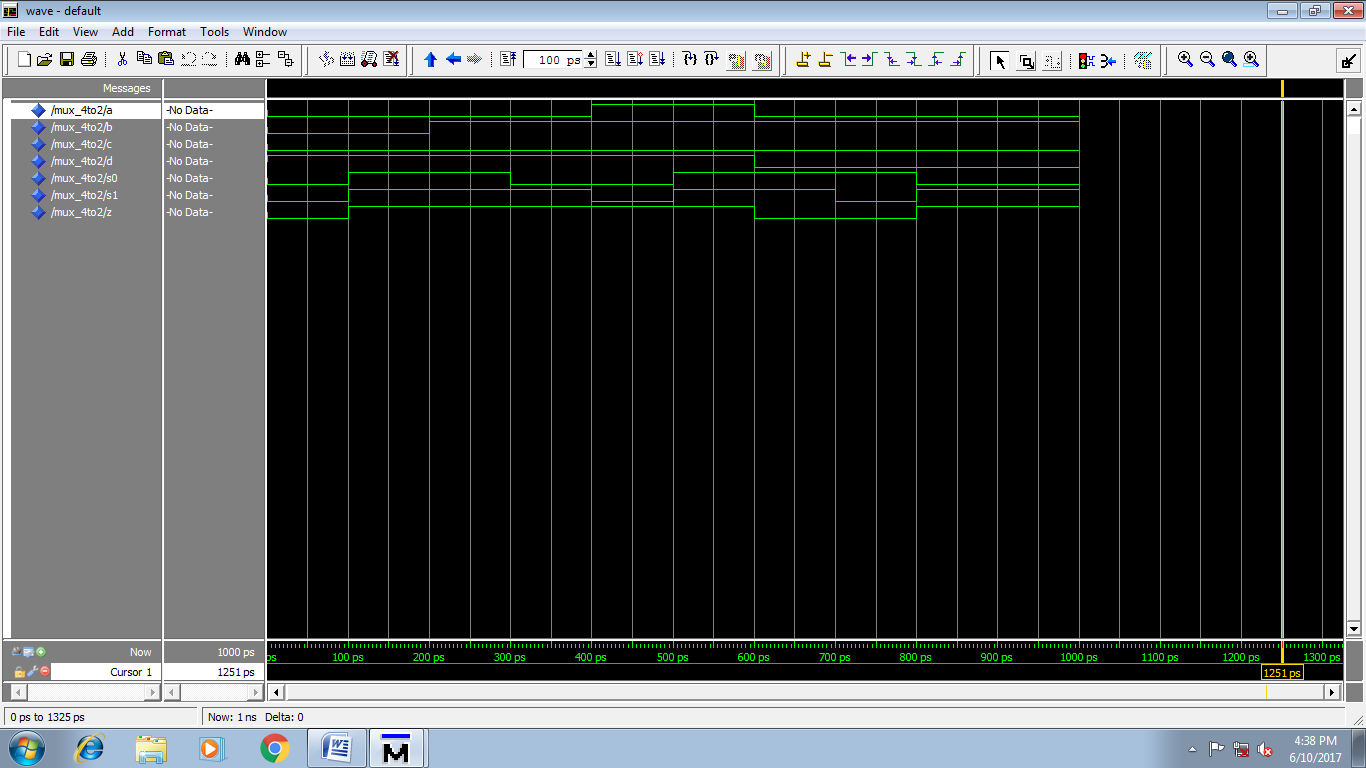
---

end if;

end process;

end data\_arch6;

**MUX 4:1**

****

library ieee;

use ieee.std\_logic\_1164.all;

--use ieee.std\_logic\_arith.all;

--use ieee.std\_logic\_unsigned.all;

entity MUX\_4to2 is

port(

A,B,C,D,S0,S1 : in std\_logic;

Z : out std\_logic

);

end MUX\_4to2;

architecture bhv\_MUX of MUX\_4to2 is

begin

process(A,B,C,D,S0,S1)

begin

if(S0 = '0' and S1 = '0') then

Z<=A;

elsif(S0='0' and S1='1') then

Z<=B;

elsif(S0='1' and S1='0') then

Z<=C;

elsif(S0='1' and S1='1') then

Z<=D;

end if;

end process;

end bhv\_MUX;

MUX 8:3

library ieee;

use ieee.std\_logic\_1164.all;

--use ieee.std\_logic\_arith.all;

--use ieee.std\_logic\_unsigned.all;

entity MUX\_4to2 is

port(

A,B,C,D,E,F,G,H,S0,S1,S2 : in std\_logic;

Z : out std\_logic

);

end MUX\_4to2;

architecture bhv\_MUX of MUX\_4to2 is

begin

process(A,B,C,D,S0,S1)

begin

if(S0 = '0' and S1 = '0' and S2='0') then

Z<=A;

elsif(S0='0' and S1='0' and S2='1') then

Z<=B;

elsif(S0='0' and S1='1' and S2='0') then

Z<=C;

elsif(S0='0' and S1='1' and S2='1') then

Z<=D;

elsif(S0 = '1' and S1 = '0' and S2='0') then

Z<=E;

elsif(S0='1' and S1='0' and S2='1') then

Z<=F;

elsif(S0='1' and S1='1' and S2='0') then

Z<=G;

elsif(S0='1' and S1='1' and S2='1') then

Z<=H;

end if;

end process;

end bhv\_MUX;

Please Run this code and find the output.