

ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ 2

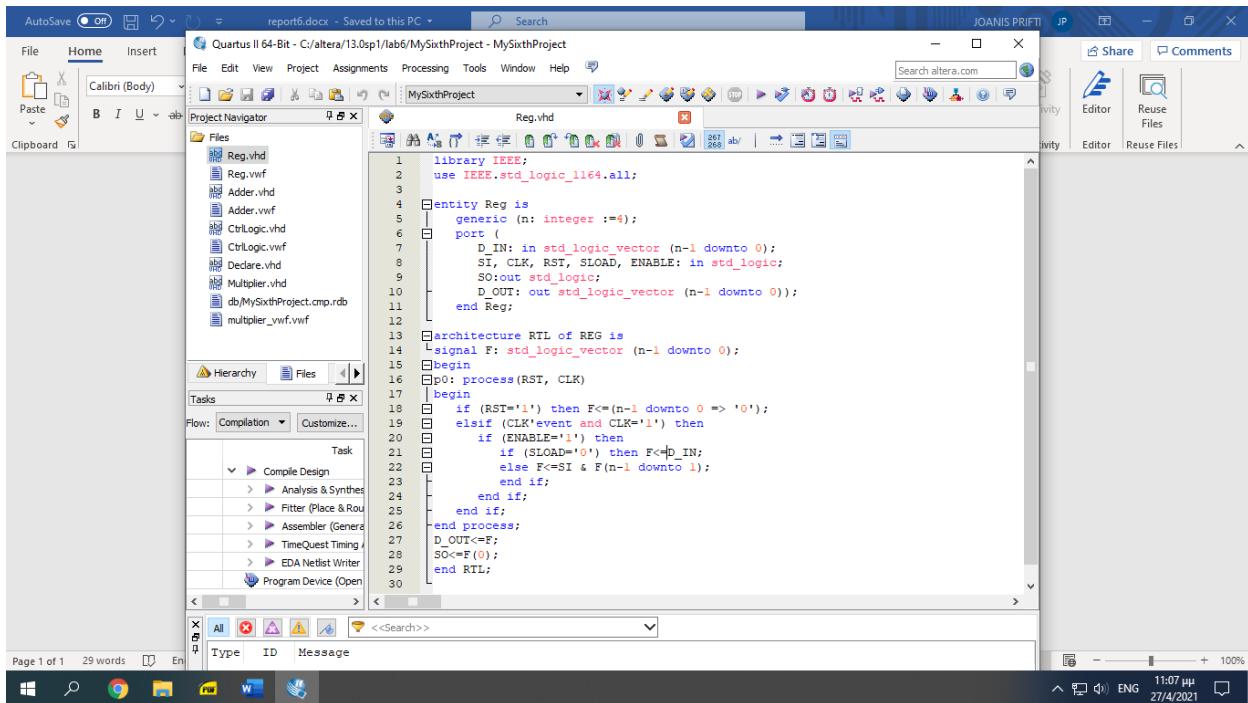
6η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

Γκότσης Βασίλης 3206

Πρίφτη Ιωάννης 3321

ΣΧΕΔΙΑΣΗ ΒΑΣΙΚΗΣ ΟΝΤΟΤΗΤΑΣ ΚΑΤΑΧΩΡΗΤΗ

Ακολουθώντας τις οδηγίες της εκφώνησης σχεδιάσαμε τον καταχωρητή των 4bit όπως φαίνεται στην παρακάτω εικόνα:

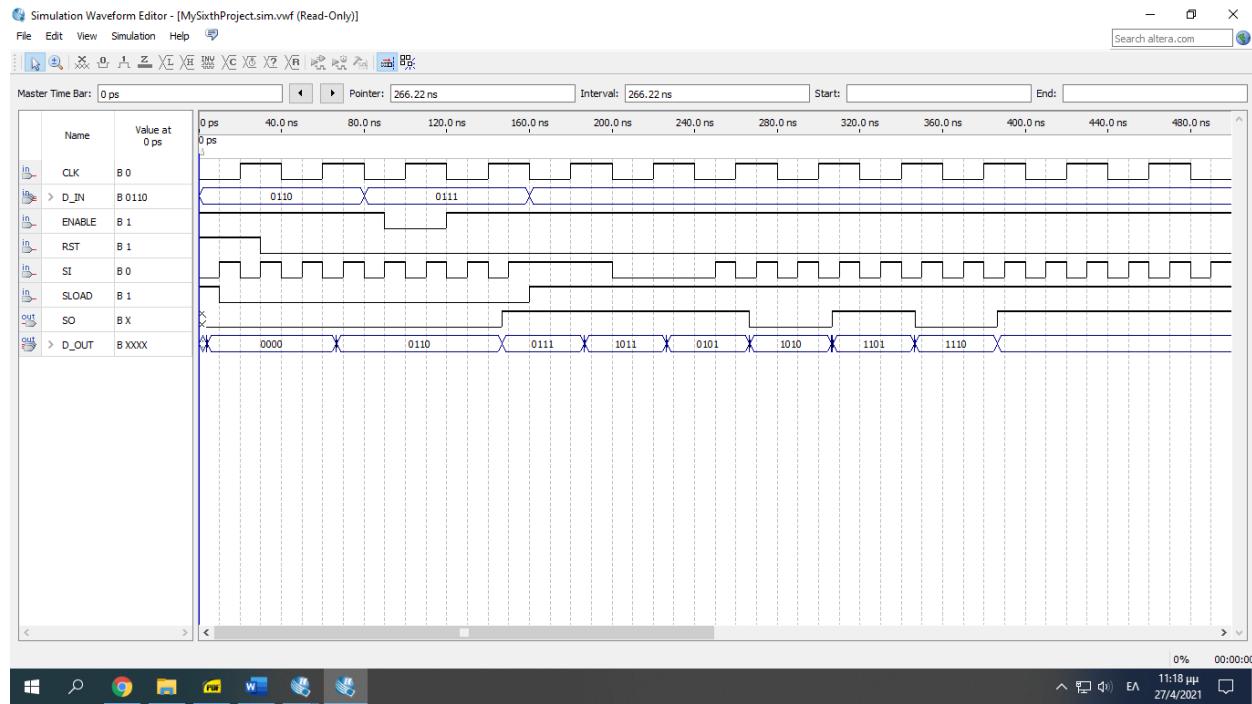


```
library IEEE;
use IEEE.std_logic_1164.all;

entity Reg is
    generic (n: integer :=4);
    port (
        D_IN: in std_logic_vector (n-1 downto 0);
        SI, CLK, RST, SLOAD, ENABLE: in std_logic;
        SO:out std_logic;
        D_OUT: out std_logic_vector (n-1 downto 0));
end Reg;

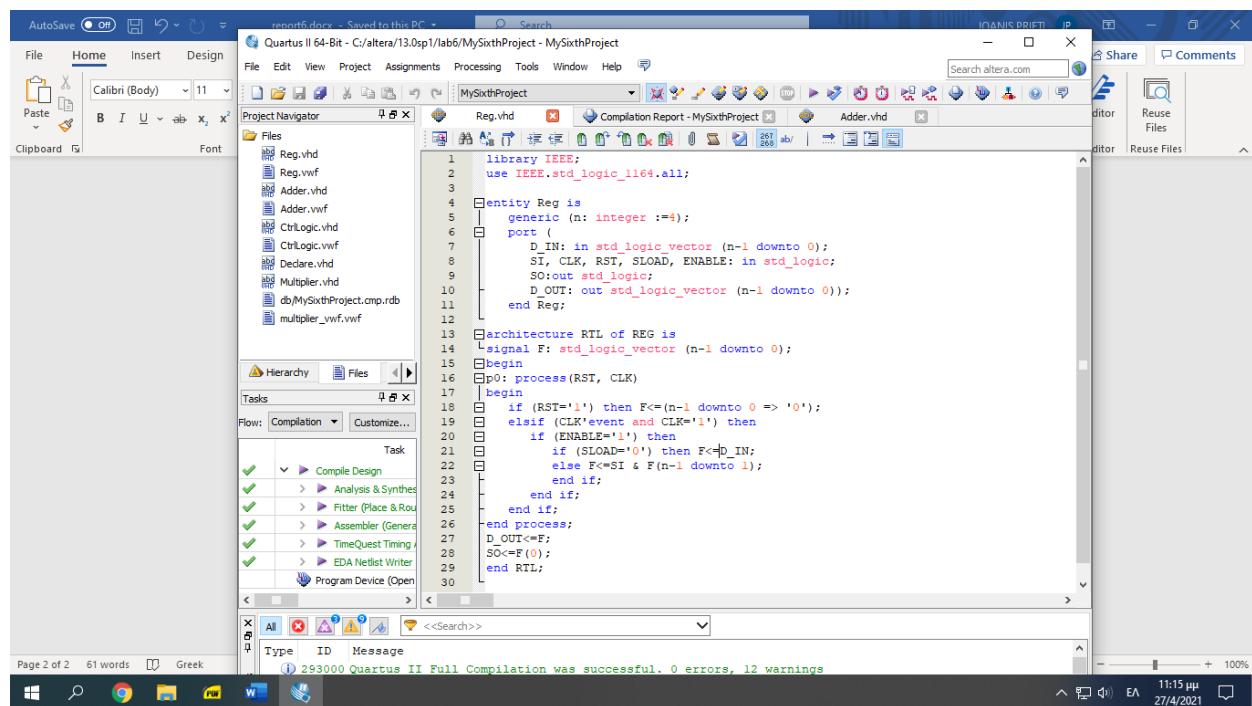
architecture RTL of REG is
signal F: std_logic_vector (n-1 downto 0);
begin
p0: process(RST, CLK)
begin
    if (RST='1') then F<=n-1 downto 0 => '0';
    elsif (CLK'event and CLK='1') then
        if (ENABLE='1') then
            if (SLOAD='0') then F<=D_IN;
            else F<=SI & F(n-1 downto 1);
        end if;
    end if;
end process;
D_OUT<=F;
SO<=F(0);
end RTL;
```

Στην συνέχεια εκτελώντας χρονική εξομοίωση το κύκλωμα λειτουργεί σωστά όπως φαίνεται στην παρακάτω εικόνα:

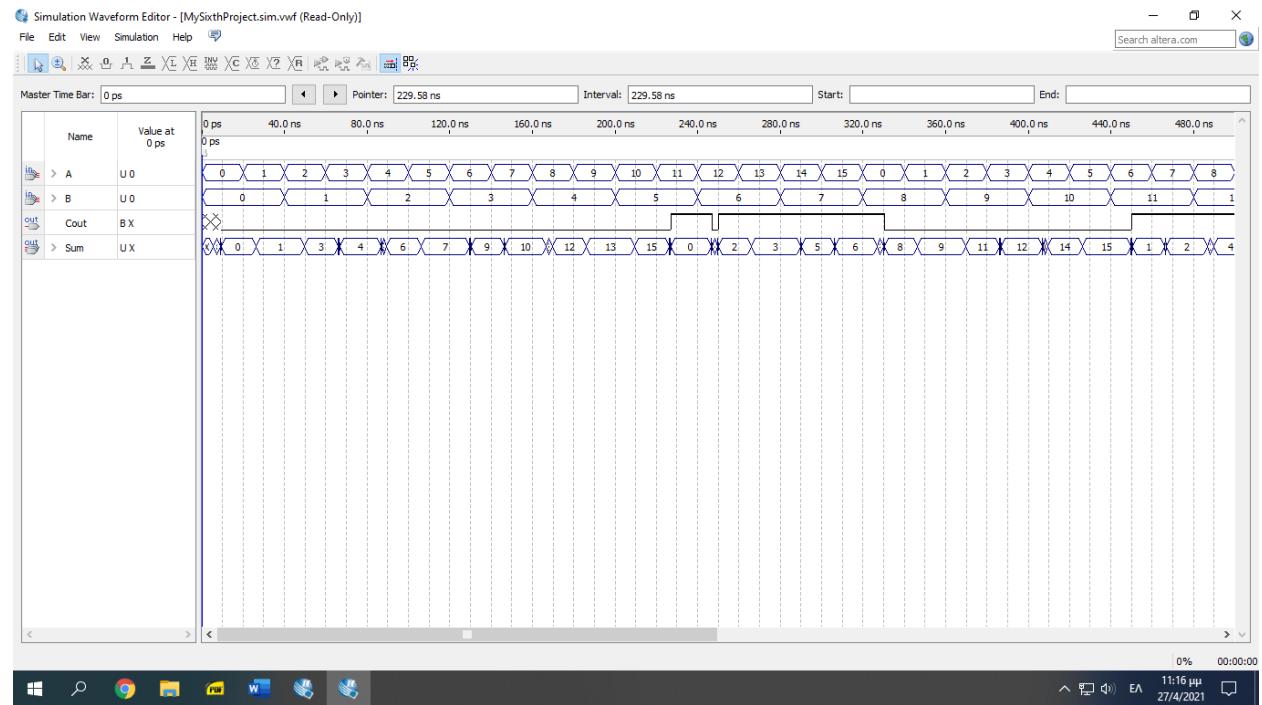


ΣΧΕΔΙΑΣΗ ΑΘΡΟΙΣΤΗ

Ακολουθώντας τις οδηγίες της εκφώνησης σχεδιάσαμε έναν αθροιστή 4bit όπως φαίνεται στην παρακάτω εικόνα:



Στη συνέχεια, εκτελώντας χρονική εξομοίωση το κύκλωμα λειτουργεί σωστά όπως φαίνεται στην παρακάτω εικόνα:



ΣΧΕΔΙΑΣΗ ΜΟΝΑΔΑΣ ΕΛΕΓΧΟΥ

Ακολουθώντας τα βήματα της εκφώνησης σχεδιάσαμε την μονάδα ελέγχου του πολλαπλασιαστή όπως φαίνεται στις παρακάτω εικόνες:

The screenshot shows the Quartus II 64-Bit interface with the project 'MySixthProject' open. The main window displays the VHDL source code for 'CtrlLogic.vhd'. The code defines an entity 'CtrlLogic' with a generic 'n' and a port containing Rst, CLK, SL_A, SL_B, SL_H, SL_L, SL_C, EN_A, EN_B, EN_H, EN_L, and EN_C. It includes an architecture 'RTL' with a process 'p0' that updates a signal 'count' based on Rst and CLK. The simulation window below shows a timeline with various logic levels for these signals over time.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use work.Declarations.all;

entity CtrlLogic is
    generic( n: integer:=4);
    port(
        Rst, CLK : in std_logic;
        SL_A, SL_B, SL_H, SL_L, SL_C : out std_logic;
        EN_A, EN_B, EN_H, EN_L, EN_C : out std_logic);
end CtrlLogic;

architecture RTL of CtrlLogic is
begin
    p0:process(Rst, CLK)
    begin
        if(Rst='1')then
            count<=n downto 0=>'0';
        elsif(CLK'event and CLK='1')then
            count<=count + '1';
        end if;
    end process;

```

This screenshot shows the same Quartus II session with the CtrlLogic.vhd code expanded. The code now includes a second process 'p1' that handles state transitions based on the 'LOAD', 'SHIFT', and 'FINISH' states. The simulation window shows the resulting logic levels for all output signals (SL_A, SL_B, SL_H, SL_L, SL_C, EN_A, EN_B, EN_H, EN_L, EN_C) over time, corresponding to the state changes defined in the process.

```
count<=count + '1';
    end if;
end process;

p1:process(Rst,CLK)
begin
    if(Rst='1')then state <= LOAD;
    elsif(CLK'event and CLK='1')then
        case state is
            when LOAD => if(conv_integer(count)=2*n-1)then state <= ADD: end if;
            when ADD => state <= SHIFT;
            when SHIFT => if(conv_integer(count)=4*n-1)then state <= FINISH:else state <= ADD: end if;
            when FINISH => null;
        end case;
    end if;
end process;

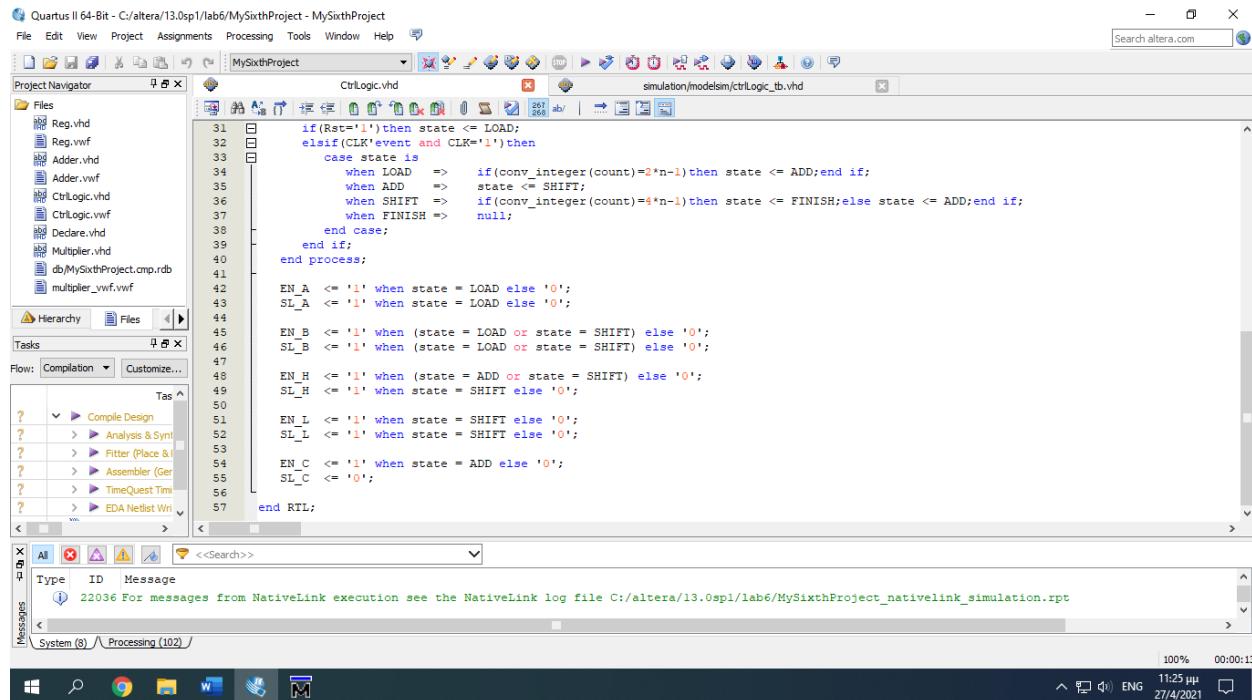
EN_A <= '1' when state = LOAD else '0';
SL_A <= '1' when state = LOAD else '0';

EN_B <= '1' when (state = LOAD or state = SHIFT) else '0';
SL_B <= '1' when (state = LOAD or state = SHIFT) else '0';

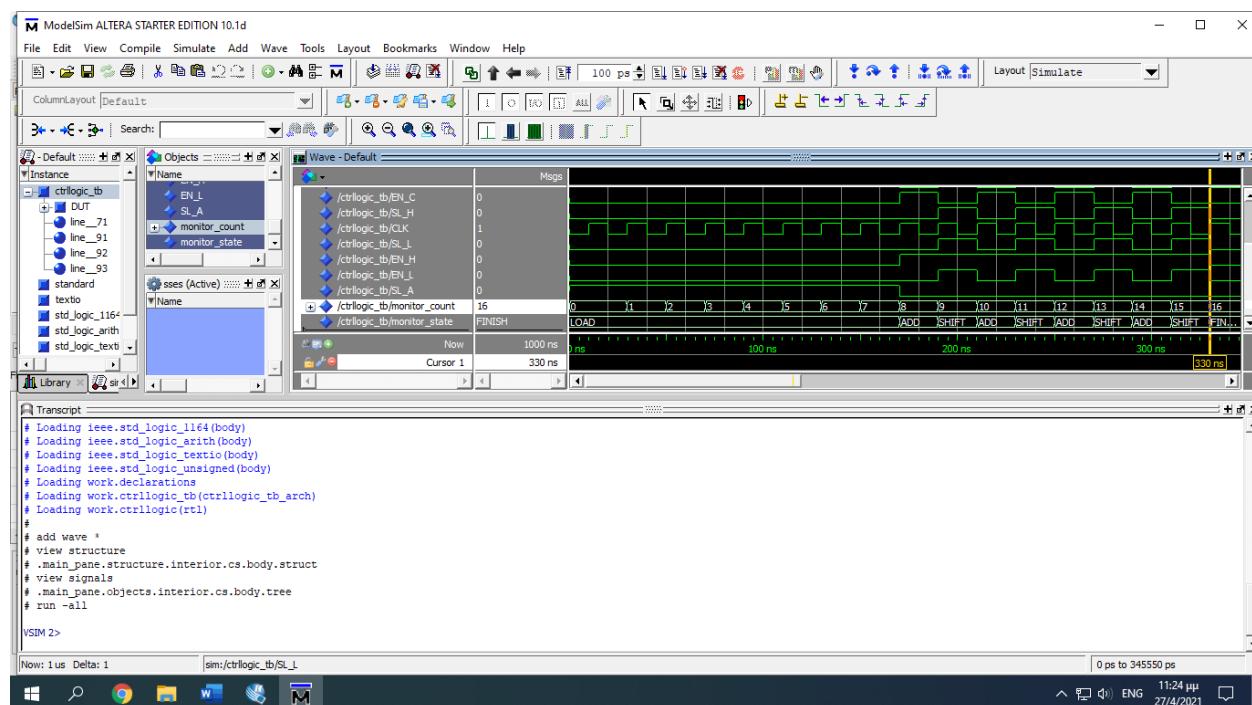
EN_H <= '1' when (state = ADD or state = SHIFT) else '0';
SL_H <= '1' when state = SHIFT else '0';

EN_L <= '1' when state = SHIFT else '0';

```



Στη συνέχεια, εκτελώντας RTL εξομοίωση φτιάξαμε το test-bench ctrlLogic_tb στο οποίο προσθέσαμε στις τελευταίες γραμμές του δύο νέα σήματα ώστε να μπορούμε να παρακολουθούμε σε ποια κατάσταση βρίσκεται και πόσοι χτύπου ρολογιού έχουν περάσει μέχρι εκείνη τη στιγμή, όπως φαίνεται στις παρακάτω εικόνες:



Όσον, αφορά την αναπαράσταση του μετρητή κύκλων ρολογιού(monitor_count) έχουμε χρησιμοποιήσει unsigned για να είναι ευδιάκριτο.

The screenshot shows the Quartus II 64-Bit software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, Help, and a search bar for altera.com. The main workspace displays the source code for 'CtrlLogic.vhd' under the 'MySixthProject' project. The code defines an entity 'ctrlLogic_tb' with a generic parameter 'n' set to 4, and an architecture 'ctrlLogic_tb_arch' that instantiates an 'ctrlLogic' component. The code uses IEEE standard libraries for logic and textio. The left sidebar shows the 'Project Navigator' with files like Reg.vhd, Adder.vhd, CtrlLogic.vhd, and simulation models. The bottom pane shows the 'Messages' window with a single warning message: '22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt'. The status bar at the bottom right shows the date as 27/4/2021.

The screenshot shows the Quartus II 64-Bit software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, Help, and a search bar for 'Search altera.com'. The main window displays a 'Project Navigator' on the left with files like Reg.vhd, Reg.vwf, Adder.vhd, Adder.vwf, CtrlLogic.vhd, CtrlLogic.vwf, Dclare.vhd, Multiplier.vhd, dbMySixthProject.cmp.rdb, and multiplier_vwf.wvf. The central workspace shows the VHDL code for 'CtrlLogic_tb_arch' of 'ctrlLogic_tb'. The code defines signals for SL_B, Rst, SL_C, EN_A, EN_B, EN_C, SL_H, and monitor_count, and a component declaration for 'CtrlLogic' with a generic parameter 'n'. The bottom status bar shows 'Ln 1 Col 1', 'VHDL File', '100%', '00:00:11', '11.25 μs', 'ENG', and the date '27/4/2021'.

```
15 L ARCHITECTURE ctrlLogic_tb_arch OF ctrlLogic_tb IS
16   SIGNAL SL_B : STD_LOGIC ;
17   SIGNAL Rst : STD_LOGIC ;
18   SIGNAL SL_C : STD_LOGIC ;
19   SIGNAL EN_A : STD_LOGIC ;
20   SIGNAL EN_B : STD_LOGIC ;
21   SIGNAL EN_C : STD_LOGIC ;
22   SIGNAL SL_H : STD_LOGIC ;
23   SIGNAL CLK : STD_LOGIC ;
24   SIGNAL SL_L : STD_LOGIC ;
25   SIGNAL EN_H : STD_LOGIC ;
26   SIGNAL EN_L : STD_LOGIC ;
27   SIGNAL SL_A : STD_LOGIC ;
28   SIGNAL monitor_count : std_logic_vector (n downto 0);
29   signal monitor_state : state_type;
30
31   COMPONENT CtrlLogic
32     GENERIC (
33       n : INTEGER );
34     PORT (
35       SL_B : out STD_LOGIC ;
36       Rst : in STD_LOGIC ;
37       SL_C : out STD_LOGIC ;
38       EN_A : out STD_LOGIC ;
39       EN_B : out STD_LOGIC ;
40       EN_C : out STD_LOGIC ;
41       SL_H : out STD_LOGIC ;
42       CLK : in STD_LOGIC ;
43       SL_L : out STD_LOGIC ;
44       monitor_count : in STD_LOGIC_VECTOR (n downto 0);
45       monitor_state : in state_type );
46   END COMPONENT CtrlLogic;
```

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator CtrlLogic.vhd Compilation Report - MySixthProject simulation/modelsim/ctrlLogic_tb.vhd

```

32   GENERIC (
33     n : INTEGER );
34   PORT (
35     SL_B : out STD_LOGIC ;
36     Rst : in STD_LOGIC ;
37     SL_C : out STD_LOGIC ;
38     EN_A : out STD_LOGIC ;
39     EN_B : out STD_LOGIC ;
40     EN_C : out STD_LOGIC ;
41     SL_H : out STD_LOGIC ;
42     CLK : in STD_LOGIC ;
43     SL_L : out STD_LOGIC ;
44     EN_H : out STD_LOGIC ;
45     EN_L : out STD_LOGIC ;
46     SL_A : out STD_LOGIC );
47   END COMPONENT ;
48 BEGIN
49   DUT : CtrlLogic
50   GENERIC MAP (
51     n => n )
52   PORT MAP (
53     SL_B => SL_B ,
54     Rst => Rst ,
55     SL_C => SL_C ,
56     EN_A => EN_A ,
57     EN_B => EN_B ,
58     EN_C => EN_C ,
59     SL_H => SL_H ,
60     CLK => CLK ,
61     SL_L => SL_L ,
62     EN_H => EN_H ,
63     EN_L => EN_L ,
64     SL_A => SL_A ) ;
65
66
67
68   -- "Clock Pattern" : dutyCycle = 50
69   -- Start Time = 0 ns, End Time = 1 us, Period = 20 ns
70   Process
71   Begin
72     clk <= '0' ;
73     wait for 10 ns ;
74   end

```

Type ID Message
22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

System (8) / Processing (102)

Ln 1 Col 1 VHDL File 100% 00:00:13
11:25 μs ENG 27/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator CtrlLogic.vhd Compilation Report - MySixthProject simulation/modelsim/ctrlLogic_tb.vhd

```

47   END COMPONENT ;
48 BEGIN
49   DUT : CtrlLogic
50   GENERIC MAP (
51     n => n )
52   PORT MAP (
53     SL_B => SL_B ,
54     Rst => Rst ,
55     SL_C => SL_C ,
56     EN_A => EN_A ,
57     EN_B => EN_B ,
58     EN_C => EN_C ,
59     SL_H => SL_H ,
60     CLK => CLK ,
61     SL_L => SL_L ,
62     EN_H => EN_H ,
63     EN_L => EN_L ,
64     SL_A => SL_A ) ;
65
66
67
68   -- "Clock Pattern" : dutyCycle = 50
69   -- Start Time = 0 ns, End Time = 1 us, Period = 20 ns
70   Process
71   Begin
72     clk <= '0' ;
73     wait for 10 ns ;
74   end

```

Type ID Message
22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

System (8) / Processing (102)

Ln 1 Col 1 VHDL File 100% 00:00:13
11:25 μs ENG 27/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator CtrlLogic.vhd Compilation Report - MySixthProject simulation/modelsim/ctrlLogic_tb.vhd

Search altera.com

CtrlLogic.vhd

```

69      -- Start Time = 0 ns, End Time = 1 us, Period = 20 ns
70      Process
71          Begin
72              clk <= '0';
73              wait for 10 ns;
74          -- 10 ns, single loop till start period.
75              for Z in 1 to 49
76                  loop
77                      clk <= '1';
78                      -- monitor_count <= << signal DUT.count : std_logic_vector (n downto 0) >>;
79                      wait for 10 ns;
80                      clk <= '0';
81                      wait for 10 ns;
82                  -- 990 ns, repeat pattern in loop.
83                  end loop;
84                  clk <= '1';
85                  -- monitor_count <= << signal DUT.count : std_logic_vector (n downto 0) >>;
86                  wait for 10 ns;
87                  -- dumped values till 1 us
88                  wait;
89              End Process;
90
91      Rst<='1','0' after 20 ns;
92      monitor_count <= << signal DUT.count : std_logic_vector (n downto 0) >>;
93      monitor_state <= << signal DUT.state : state_type >>;
94  END;
95

```

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

System (8) / Processing (102) /

Ln 1 Col 1 VHDL File 100% 00:00:13

1125 µs 27/4/2021

Επιτέλεον, το αρχείο με την δήλωση τις μεταβλητής καταστάσεων φαίνεται στην παρακάτω εικόνα:

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Declare.vhd

Search altera.com

Declare.vhd

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  package Declarations is
5      type state_type is (LOAD, ADD, SHIFT, FINISH);
6  end Declarations;
7
8

```

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

System (8) / Processing (102) /

Ln 1 Col 1 VHDL File 100% 00:00:13

1136 µs 27/4/2021

ΣΧΕΔΙΑΣΗ ΠΟΛΛΑΠΛΑΣΙΑΣΤΗ

Ακολουθώντας τα βήματα της εκφώνησης, δημιουργήσαμε ένα νέο αρχείο για την περιγραφή του πολλαπλασιαστή και τοποθετήσαμε εκεί τον κώδικα του όπως περιγράφεται. Το κύκλωμα φαίνεται στις παρακάτω εικόνες:

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use work.Declarations.all;

entity Multiplier is
    generic (n: integer :=4);
    port (
        Rst, CLK, SI : in std_logic;
        Low, High : out std_logic_vector (n-1 downto 0));
end Multiplier;

architecture RTL_Mul of Multiplier is
begin
    component CtrlLogic
        generic (n: integer := 4);
        port (
            Rst, CLK : in std_logic;
            SL_A, SL_B, SL_H, SL_L, SL_C : out std_logic;
            EN_A, EN_B, EN_H, EN_L, EN_C : out std_logic );
    end component;
    component Adder
        generic (n: integer :=4);
        port (
            A, B : in std_logic_vector (n-1 downto 0);
            SUM : out std_logic_vector (n-1 downto 0);
            COUT : out std_logic );
    end component;
end;

```

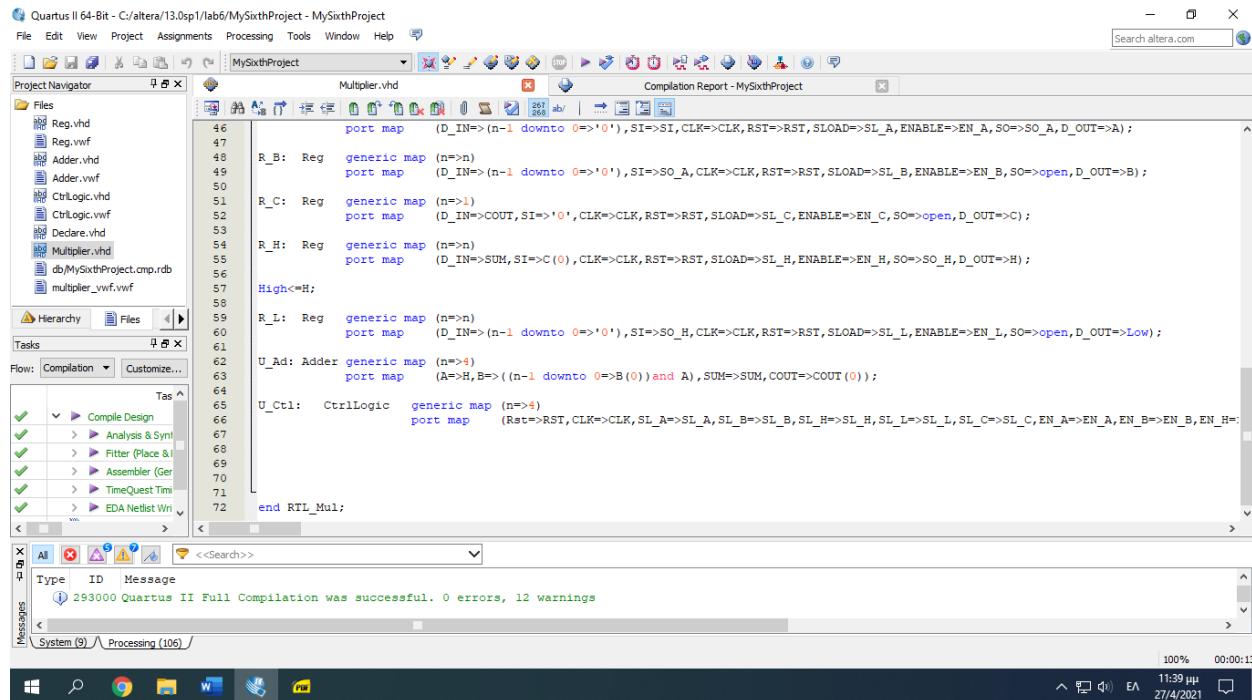
Messages pane shows: 293000 Quartus II Full Compilation was successful. 0 errors, 12 warnings.

```

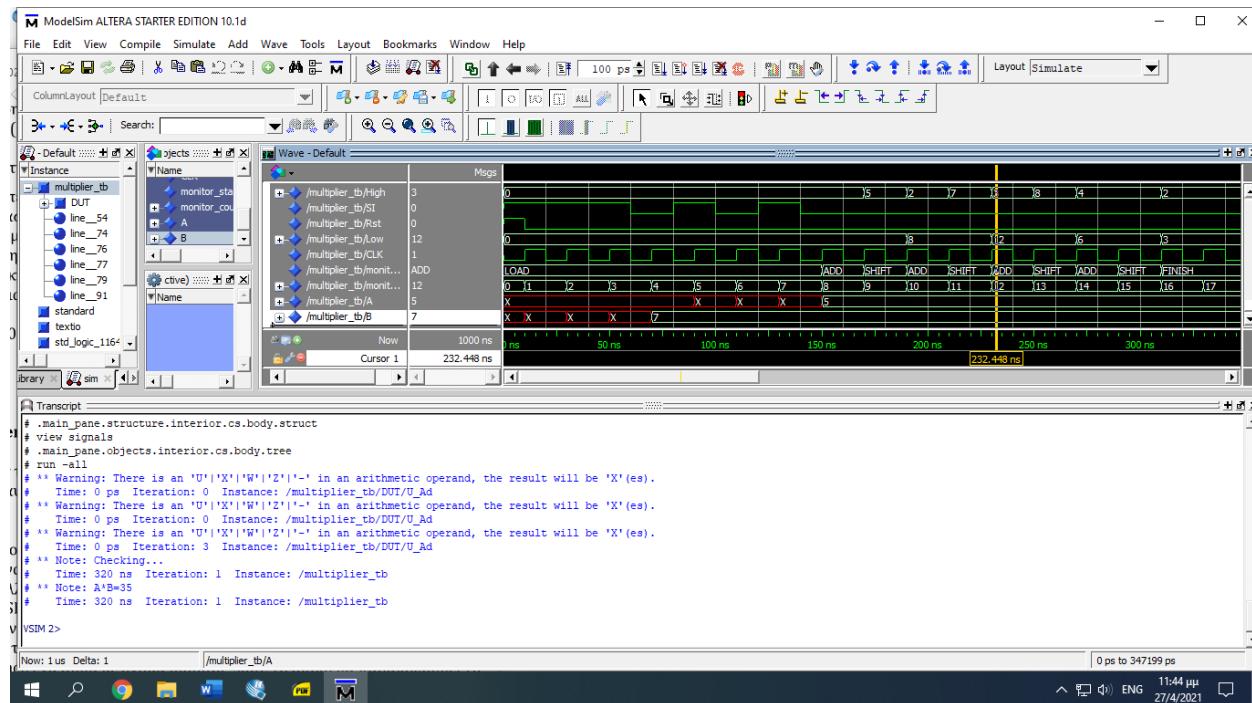
component Adder
    generic (n: integer :=4);
    port (
        A, B : in std_logic_vector (n-1 downto 0);
        SUM : out std_logic_vector (n-1 downto 0);
        COUT : out std_logic );
end component;
component Reg
    generic (n: integer:=4);
    port (
        D_in: in std_logic_vector (n-1 downto 0);
        SI, CLK, RST, SLOAD, ENABLE: in std_logic;
        SO: out std_logic;
        D_OUT: out std_logic_vector (n-1 downto 0));
end component;
signal SL_A, SL_B, SL_H, SL_L, SL_C, EN_A, EN_B, EN_H, EN_L, EN_C : std_logic;
signal SO_A, SO_H : std_logic;
signal A, B, SUM, H: std_logic_vector(n-1 downto 0);
signal C, COUT : std_logic_vector (0 downto 0);
begin
    R_A: Reg generic map (n=>n)
        port map (D_IN=>(n-1 downto 0=>'0'),SI=>SI,CLK=>CLK,RST=>RST,SLOAD=>SL_A,ENABLE=>EN_A,SO=>SO_A,D_OUT=>A);
    R_B: Reg generic map (n=>n)
        port map (D_IN=>(n-1 downto 0=>'0'),SI=>SO_A,CLK=>CLK,RST=>RST,SLOAD=>SL_B,ENABLE=>EN_B,SO=>open,D_OUT=>B);

```

Messages pane shows: 293000 Quartus II Full Compilation was successful. 0 errors, 12 warnings.



Εκτελώντας RTL εξομοίωση παρατηρούμε ότι το κύκλωμα λειτουργεί σωστά, όπως φαίνεται στην παρακάτω εικόνα:



Παρατηρούμε ότι στην κατάσταση finish έχει βρει το αποτέλεσμα του πολλαπλασιασμού το οποίο προκύπτει συνενώνοντας τα περιεχόμενα των καταχωρητών High & Low(0010 & 0011=00100011=35) το οποίο έχουμε βάλει να τυπώνεται και στο transcript. Έχουμε επιλέξει unsigned αναπαράσταση για όλα τα σήματα εκτός από τα High και Low για να είναι ευδιάκριτα τα αποτελέσματα.

Φυσικά όλα τα παραπάνω τα υλοποιήσαμε τροποποιώντας το test-bench multiplier_tb όπως φαίνεται στις παρακάτω εικόνες:

```

LIBRARY ieee;
LIBRARY std;
LIBRARY work;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_textio.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.all;
USE std.textio.all;
USE work.Declarations.all;

ENTITY multiplier_tb IS
  GENERIC (
    n : INTEGER := 4 );
END ;

ARCHITECTURE multiplier_tb_arch OF multiplier_tb IS
  SIGNAL High : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL SI : STD_LOGIC ;
  SIGNAL Rst : STD_LOGIC ;
  SIGNAL Low : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL CLK : STD_LOGIC ;
  SIGNAL monitor_state : state_type;
  SIGNAL monitor_count : std_logic_vector(n downto 0);
  SIGNAL A : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL B : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  COMPONENT Multiplier
    GENERIC (
      n : INTEGER );
    PORT (
      High : out STD_LOGIC_VECTOR (n - 1 downto 0) ;
      SI : in STD_LOGIC ;
      Rst : in STD_LOGIC ;
      Low : out STD_LOGIC_VECTOR (n - 1 downto 0) ;
      CLK : in STD_LOGIC );
  END COMPONENT ;
  BEGIN
    DUT : Multiplier
    GENERIC MAP (
      n => n )
    PORT MAP (
      High => High ,
      SI => SI ,
      Rst => Rst ,
      Low => Low ,
      CLK => CLK );
  END ;

```

```

LIBRARY ieee;
LIBRARY std;
LIBRARY work;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_textio.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.std_logic_unsigned.all;
USE ieee.numeric_std.all;
USE std.textio.all;
USE work.Declarations.all;

ENTITY multiplier_tb IS
  GENERIC (
    n : INTEGER := 4 );
END ;

ARCHITECTURE multiplier_tb_arch OF multiplier_tb IS
  SIGNAL High : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL SI : STD_LOGIC ;
  SIGNAL Rst : STD_LOGIC ;
  SIGNAL Low : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL CLK : STD_LOGIC ;
  SIGNAL monitor_state : state_type;
  SIGNAL monitor_count : std_logic_vector(n downto 0);
  SIGNAL A : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  SIGNAL B : STD_LOGIC_VECTOR (n - 1 downto 0) ;
  COMPONENT Multiplier
    GENERIC (
      n : INTEGER );
    PORT (
      High : out STD_LOGIC_VECTOR (n - 1 downto 0) ;
      SI : in STD_LOGIC ;
      Rst : in STD_LOGIC ;
      Low : out STD_LOGIC_VECTOR (n - 1 downto 0) ;
      CLK : in STD_LOGIC );
  END COMPONENT ;
  BEGIN
    DUT : Multiplier
    GENERIC MAP (
      n => n )
    PORT MAP (
      High => High ,
      SI => SI ,
      Rst => Rst ,
      Low => Low ,
      CLK => CLK );
  END ;

```

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Multiplier.vhd Compilation Report - MySixthProject simulation/modelsim/multiplier_tb.vhd

Files

- Reg.vhd
- Reg.vwf
- Adder.vhd
- Adder.vwf
- CtrlLogic.vhd
- CtrlLogic.vwf
- Dedare.vhd
- Multiplier.vhd
- db/MySixthProject.cmp.rdb
- multiplier_wvf.vwf

Hierarchy Files

Tasks

Flow: Compilation Customize...

Tags

- Compile Design
- Analysis & Synth
- Fitter (Place & Route)
- Assembler (Generate)
- TimeQuest Timing
- EDA Netlist Writing

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

System (13) / Processing (106) /

```

36      CLK : in STD_LOGIC );
37  END COMPONENT ;
38  BEGIN
39    DUT : Multiplier
40    GENERIC MAP (
41      n => n )
42    PORT MAP (
43      High => High ,
44      SI => SI ,
45      Rst => Rst ,
46      Low => Low ,
47      CLK => CLK ) ;
48
49
50  -- "Clock Pattern" : dutyCycle = 50
51  -- Start Time = 0 ns, End Time = 1 us, Period = 20 ns
52  Process
53  Begin
54    clk <='0';
55    Rst<='1';
56    wait for 10 ns;
57    -- 10 ns, single loop till start period.
58    for Z in 1 to 49
59    loop
60      clk <='1';
61      Rst<='0';
62      ....
63    end loop;
64    wait for 10 ns;
65    -- 10 ns, single loop till start period.
66    for Z in 1 to 49
67    loop
68      clk <='1';
69      Rst<='0';
70      ....
71    end loop;
72  End Process;
73
74  SI <'1' after 0 ns, '1' after 20 ns, '1' after 40 ns, '0' after 60 ns, '1' after 80 ns, '0' after 100 ns, '1'
75  after 120 ns, '0' after 140 ns;
76  monitor_state <= << signal DUT.U_Ctl.state : state_type >>;
77  monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
78
79  process(CLK,monitor_count,SI,A,B)
80  begin
81    if (CLK'event and CLK='1') then
82      if (conv_integer(monitor_count)<n) then
83        B(conv_integer(monitor_count))<=SI;
84      elsif ((conv_integer(monitor_count)=n) or conv_integer(monitor_count)>n) and (conv_integer(monitor_count)<2*n) then
85        A(conv_integer(monitor_count)-n)<=SI;
86      end if;
87    end if;
88  end process;

```

100% 00:00:13

Quartus II 64-Bit - C:/altera/13.0sp1/lab6/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Multiplier.vhd Compilation Report - MySixthProject simulation/modelsim/multiplier_tb.vhd

Files

- Reg.vhd
- Reg.vwf
- Adder.vhd
- Adder.vwf
- CtrlLogic.vhd
- CtrlLogic.vwf
- Dedare.vhd
- Multiplier.vhd
- db/MySixthProject.cmp.rdb
- multiplier_wvf.vwf

Hierarchy Files

Tasks

Flow: Compilation Customize...

Tags

- Compile Design
- Analysis & Synth
- Fitter (Place & Route)
- Assembler (Generate)
- TimeQuest Timing
- EDA Netlist Writing

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6/MySixthProject_nativelink_simulation.rpt

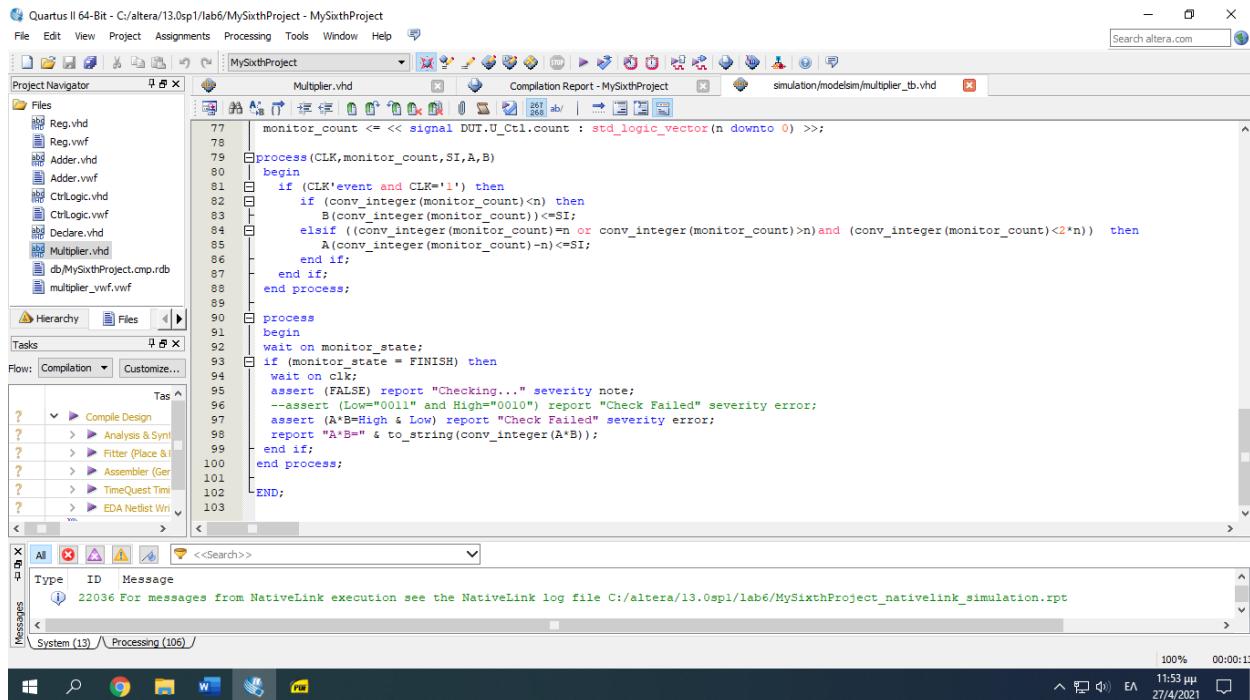
System (13) / Processing (106) /

```

60  loop
61    clk <='1';
62    Rst<='0';
63    wait for 10 ns;
64    clk <='0';
65    wait for 10 ns;
66    -- 990 ns, repeat pattern in loop.
67    end loop;
68    clk <='1';
69    wait for 10 ns;
70    -- dumped values till 1 us
71    wait;
72  End Process;
73
74  SI <'1' after 0 ns, '1' after 20 ns, '1' after 40 ns, '0' after 60 ns, '1' after 80 ns, '0' after 100 ns, '1'
75  after 120 ns, '0' after 140 ns;
76  monitor_state <= << signal DUT.U_Ctl.state : state_type >>;
77  monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
78
79  process(CLK,monitor_count,SI,A,B)
80  begin
81    if (CLK'event and CLK='1') then
82      if (conv_integer(monitor_count)<n) then
83        B(conv_integer(monitor_count))<=SI;
84      elsif ((conv_integer(monitor_count)=n) or conv_integer(monitor_count)>n) and (conv_integer(monitor_count)<2*n) then
85        A(conv_integer(monitor_count)-n)<=SI;
86      end if;
87    end if;
88  end process;

```

100% 00:00:13



Αρχικά, προσθέσαμε τα σήματα:

```

SIGNAL monitor_state : state_type;
SIGNAL monitor_count : std_logic_vector(n downto 0);
SIGNAL A : STD_LOGIC_VECTOR (n - 1 downto 0) ;
SIGNAL B : STD_LOGIC_VECTOR (n - 1 downto 0) ;

```

Όπου το monitor_state κρατάει την κατάσταση στην οποία βρίσκεται ο πολλαπλασιαστής, το σήμα monitor_count είναι ουσιαστικά ο μετρητής του ρολογιού, το σήμα A κρατάει τον αριθμό που θα φορτωθεί στον καταχωρητή A ενώ το σήμα B κρατάει τον αριθμό που θα φορτωθεί στον καταχωρητή B.

Στην Process που διαχειρίζεται το ρολόι έχουμε βάλει $Rst \leq '1'$; ώστε γίνει η αρχικοποίηση του κυκλώματος κατά την έναρξη του ρολογιού και στην συνέχεια έχουμε βάλει $Rst \leq '0'$; στην αρχή της for loop του ρολογιού ώστε να ξεκινήσει η φόρτωση των δεδομένων και επιπλέον να παραμείνει το Rst στην τιμή 0 διότι αν πάει στην τιμή 1 μπορεί να γίνει αρχικοποίηση του κυκλώματος πριν αυτό ολοκληρώσει τον υπολογισμό, γι' αυτό τον λόγο θέλουμε να παραμείνει 0.

Στην συνέχεια έχουμε βάλει:

$SI \leq '1'$ after 0 ns, '1' after 20 ns, '1' after 40 ns, '0' after 60 ns, '1' after 80 ns, '0' after 100 ns, '1' after 120 ns, '0' after 140 ns;

Ωστε να φορτωθούν σειριακά στην σειριακή είσοδο τα δεδομένα, δηλαδή να φορτωθούν τα δεδομένα και στους δύο καταχωρητές και πιο συγκεκριμένα για τον B θα πρέπει να ολισθήσουν πρώτα από τον A.

Όταν αυτή η διαδικασία ολοκληρωθεί (μετά από 2n κύκλους, όπου n=4 bit), ο καταχωρητής A θα έχει αποθηκευμένη την τιμή 7 και ο B την τιμή 5.

Έπειτα προσθέσαμε τα σήματα:

```
monitor_state <= << signal DUT.U_Ctl.state : state_type >>;  
monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
```

ώστε να μπορέσουμε να παρατηρούμε σε ποια κατάσταση βρίσκεται το κύκλωμα ανά πάσα στιγμή και σε ποιόν κύκλο ρολογιού βρίσκεται.

Στη συνέχεια φτιάξαμε μια process η οποία ελέγχει σύγχρονα(σε κάθε χτύπο ρολογιού) πόσοι κύκλοι έχουν περάσει και φορτώνει ανάλογα τα δεδομένα πρώτα στο σήμα B και μετά στο A. Ο λόγος που έγινε είναι για να μπορούμε στην συνέχεια να ελέγχουμε με αυτόματο αν το αποτέλεσμα για όποιους αριθμούς φορτωθούν στους καταχωρητές A και B είναι το αναμενόμενο($A*B=High \& Low$) ώστε να μην χρειάζεται να το βρίσουμε χειροκίνητα και να το ελέγχουμε.

Τέλος στην τελευταία process έχουμε τροποποιήσει την assert ώστε να κάνει τον αυτοματοποιημένο έλεγχο $A*B=High \& Low$ και εκτυπώνουμε με μια report το αναμενόμενο αποτέλεσμα στο transcript.

ΕΝΑΛΛΑΓΗ ΣΕ 8bit ΚΑΙ ΠΑΡΑΛΛΗΛΗ ΦΟΡΤΩΣΗ

-Στο κύκλωμα του καταχωρητή αλλάξαμε το η από 4 σε 8, ώστε να μπορεί ο καταχωρητής να αποθηκεύει αριθμούς των 8bit, το κύκλωμα φαίνεται στην παρακάτω εικόνα:

```

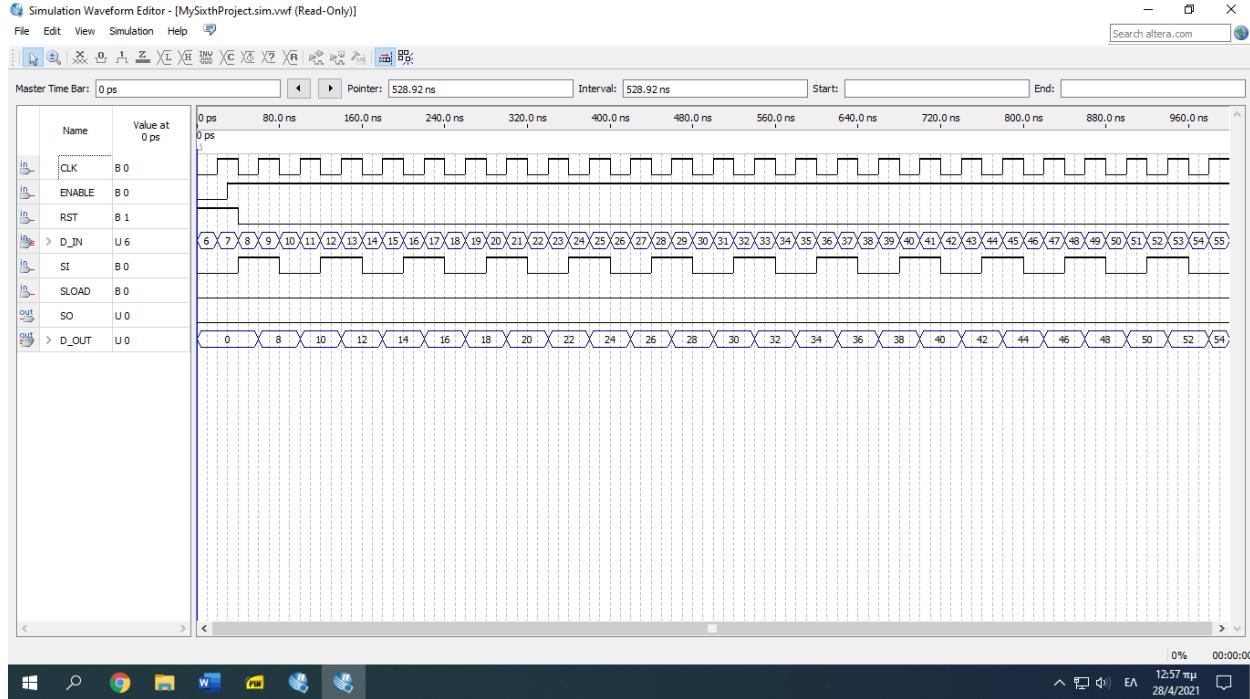
library IEEE;
use IEEE.std_logic_1164.all;

entity Reg is
generic (n: integer :=8);
port (
    D_IN: in std_logic_vector (n-1 downto 0);
    SI, CLR, RST, SLOAD, ENABLE: in std_logic;
    SO:out std_logic;
    D_OUT: out std_logic_vector (n-1 downto 0));
end Reg;

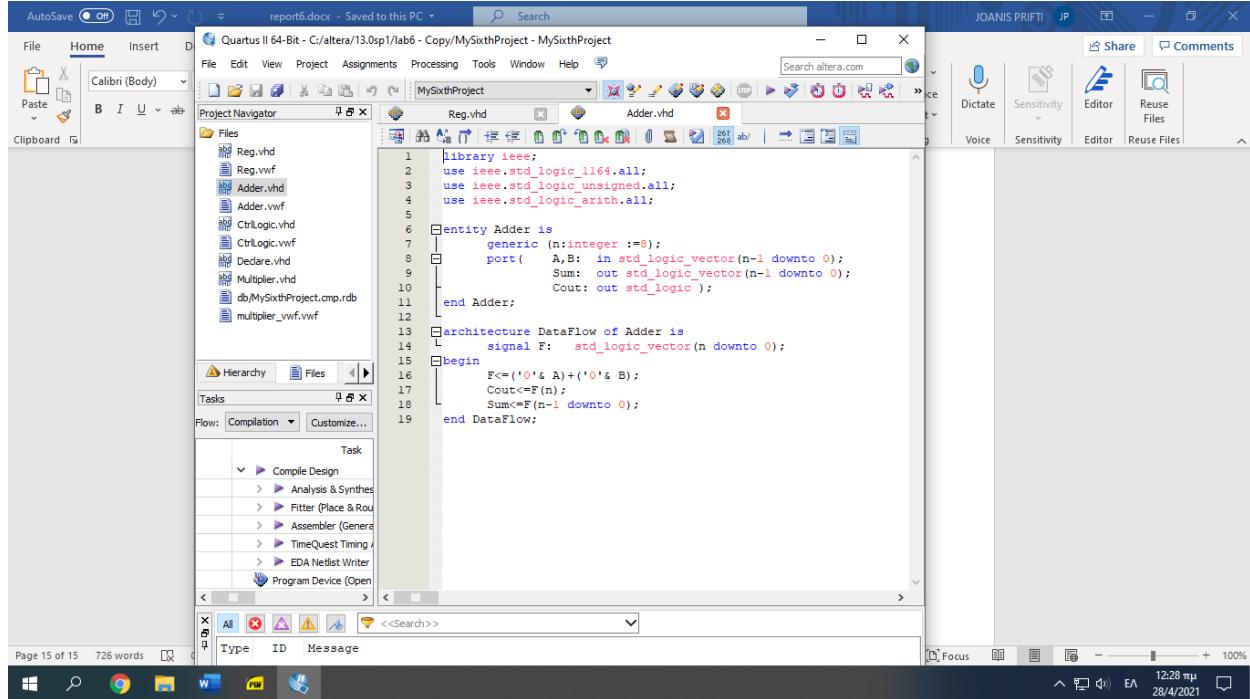
architecture RTL of REG is
signal F: std_logic_vector (n-1 downto 0);
begin
p0: process(RST, CLK)
begin
    if (RST='1') then F<=(n-1 downto 0 => '0');
    elsif (CLK'event and CLK='1') then
        if (ENABLE='1') then
            if (SLOAD='0') then F<=D_IN;
            else F=SI & F(n-1 downto 1);
            end if;
        end if;
    end process;
    D_OUT<=F;
    SO<=F(0);
end RTL;

```

Το κύκλωμα λειτουργεί σωστά όπως φαίνεται και από την παρακάτω στατική εξομοίωση:



-Στο κύκλωμα του αθροιστή αλλάξαμε το η από 4 σε 8bit ώστε να μπορεί να κάνει πράξεις(άθροισμα) με αριθμούς των 8bit, το κύκλωμα φαίνεται στην παρακάτω εικόνα:



The screenshot shows the Quartus II software interface with the project 'MySixthProject' open. The main window displays the VHDL code for an 8-bit adder:

```

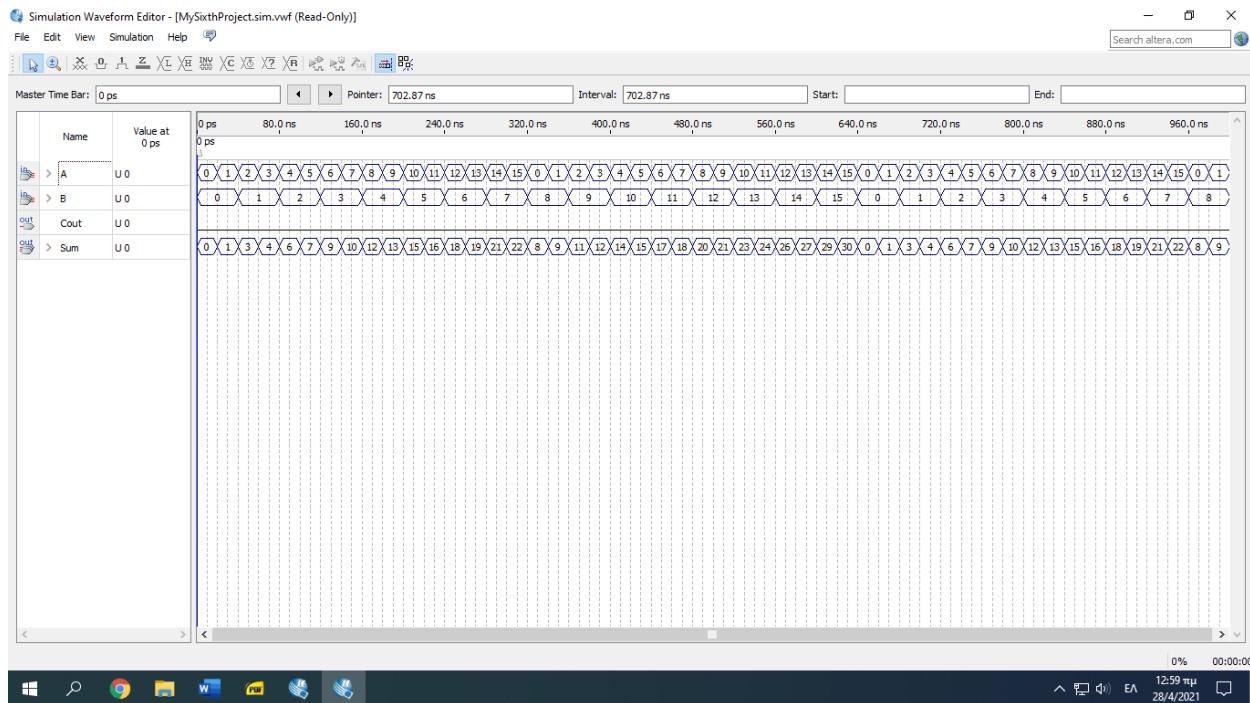
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity Adder is
    generic (n:integer :=8);
    port( A,B:  in std_logic_vector(n-1 downto 0);
          Sum: out std_logic_vector(n-1 downto 0);
          Cout: out std_logic );
end Adder;

architecture DataFlow of Adder is
begin
    F<=(*0'& A)+(*0'& B);
    Cout<=F(n);
    Sum<=F(n-1 downto 0);
end DataFlow;

```

Εκτελώντας στατική εξομοίωση το κύκλωμα λειτουργεί σωστά όπως φαίνεται στην παρακάτω εικόνα:



-Στο κύκλωμα της μονάδας ελέγχου αλλάξαμε το η από 4 σε 8bit ώστε να μπορεί να κάνει πράξεις με αριθμούς των 8bit.

-Η επόμενη αλλαγή έγινε όταν το κύκλωμα βρίσκεται στην κατάσταση LOAD να πηγαίνει στην επομένη μετά από έναν κύκλο διότι η φόρτωση των δεδομένων γίνεται παράλληλα, δηλαδή σε έναν κύκλο ρολογιού: when LOAD => if(conv_integer(count)=0)then state <= ADD;end if;

-Στη συνέχεια χρειάζεται 2* n κύκλους επιπλέον ώστε να ολοκληρώσει τον υπολογισμό, οπότε η αλλαγή έγινε στην κατάσταση SHIFT: when SHIFT => if(conv_integer(count)=2*n)then state <= FINISH;else state <= ADD;end if;

-Ο καταχωρητής A για να μπορέσει να φορτώσει παράλληλα δεδομένα θα πρέπει το σήμα SL_A να γίνει 0 για την τιμή 1 κάνει σειριακή ολίσθηση και παράλληλη φόρτωση: SL_A <= '0' when state = LOAD ;

-Αντίστοιχα και στον καταχωρητής B θα πρέπει το σήμα SL_B να γίνει 0 και επιπλέον αν βρεθούμε στην κατάσταση της ολίσθησης τότε θα πρέπει να γίνει 1: SL_B <= '0' when (state = LOAD) else '1' when (state = SHIFT);

Το κύκλωμα με τις αλλαγές φαίνεται στις παρακάτω εικόνες:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use work.Declarations.all;

entity CtrlLogic is
generic( n: integer:=8);
port(
    Rst, CLK :in std_logic;
    SL_A, SL_B, SL_H, SL_L, SL_C :out std_logic;
    EN_A, EN_B, EN_H, EN_L, EN_C :out std_logic);
end CtrlLogic;

architecture RTL of CtrlLogic is
begin
    p0:process(Rst, CLK)
    begin
        if(Rst='1')then
            count<=(n downto 0=>'0');
        elsif(CLK'event and CLK='1')then
            count<=count + '1';
        end if;
    end process;
end;
```

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Reg.vhd Adder.vhd CtrlLogic.vhd

```

19 begin
20   p0:process(Rst, CLK)
21   begin
22     if(Rst='1')then
23       count:=(n downto 0=>'0');
24     elsif(CLK'event and CLK='1')then
25       count:=count + '1';
26     end if;
27   end process;
28
29   p1:process(Rst,CLK)
30   begin
31     if(Rst='1')then state <= LOAD;
32     elsif(CLK'event and CLK='1')then
33       case state is
34         when LOAD => if(conv_integer(count)=0)then state <= ADD;end if;
35         when ADD => state <= SHIFT;
36         when SHIFT => if(conv_integer(count)=2*n)then state <= FINISH;else state <= ADD;end if;
37         when FINISH => null;
38       end case;
39     end if;
40   end process;
41
42   EN_A <= '1' when state = LOAD else '0';
43   SL_A <= '0' when state = LOAD ;
44
45   EN_B <= '1' when (state = LOAD or state = SHIFT) else '0';
46   SL_B <= '0' when (state = LOAD or state = SHIFT) else '0';
47
48   EN_H <= '1' when (state = ADD or state = SHIFT) else '0';
49   SL_H <= '1' when state = SHIFT else '0';
50
51   EN_L <= '1' when state = SHIFT else '0';
52   SL_L <= '1' when state = SHIFT else '0';
53
54   EN_C <= '1' when state = ADD else '0';
55   SL_C <= '0';
56
57 end RTL;

```

Tasks Compile Design Analysis & Synth Fitter (Place & Route) Assembler (Generate) TimeQuest Timing EDA Netlist Write

Messages All Type ID Message

System Processing /

Ln 22 Col 22 VHDL File 0% 00:00:00 12:41 πμ 28/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Reg.vhd Adder.vhd CtrlLogic.vhd

```

31 if(Rst='1')then state <= LOAD;
32 elsif(CLK'event and CLK='1')then
33   case state is
34     when LOAD => if(conv_integer(count)=0)then state <= ADD;end if;
35     when ADD => state <= SHIFT;
36     when SHIFT => if(conv_integer(count)=2*n)then state <= FINISH;else state <= ADD;end if;
37     when FINISH => null;
38   end case;
39 end if;
40 end process;
41
42 EN_A <= '1' when state = LOAD else '0';
43 SL_A <= '0' when state = LOAD ;
44
45 EN_B <= '1' when (state = LOAD or state = SHIFT) else '0';
46 SL_B <= '0' when (state = LOAD or state = SHIFT) else '0';
47
48 EN_H <= '1' when (state = ADD or state = SHIFT) else '0';
49 SL_H <= '1' when state = SHIFT else '0';
50
51 EN_L <= '1' when state = SHIFT else '0';
52 SL_L <= '1' when state = SHIFT else '0';
53
54 EN_C <= '1' when state = ADD else '0';
55 SL_C <= '0';
56
57 end RTL;

```

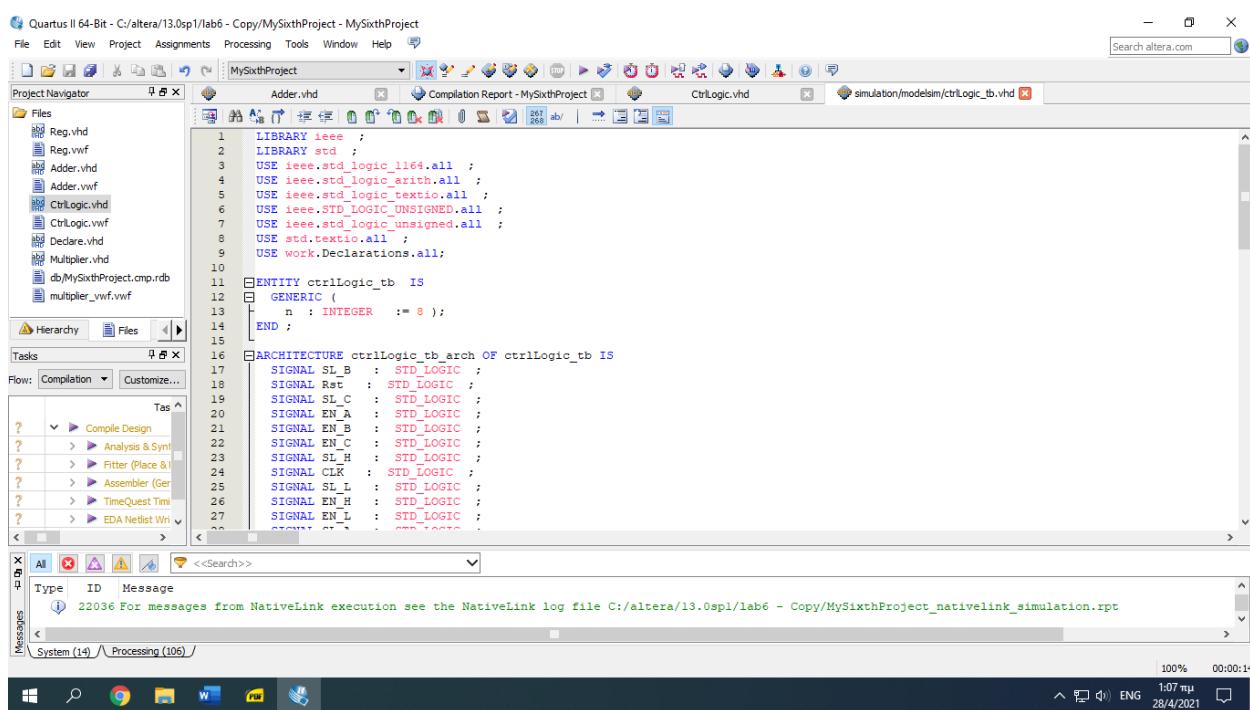
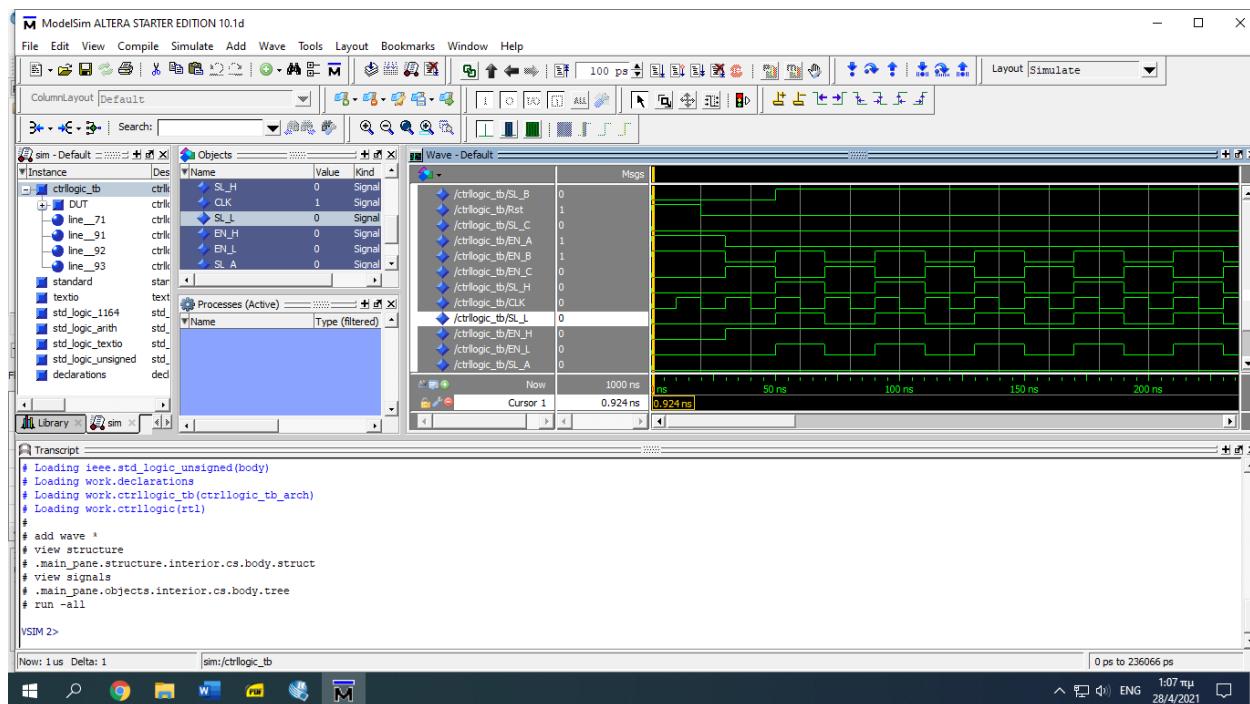
Tasks Compile Design Analysis & Synth Fitter (Place & Route) Assembler (Generate) TimeQuest Timing EDA Netlist Write

Messages All Type ID Message

System Processing /

Ln 22 Col 22 VHDL File 0% 00:00:00 12:41 πμ 28/4/2021

-Επιπλέον, αλλάξαμε στο test-bench ctrlLogic_tb την τιμή του n από 4 σε 8bit και εκτελέσαμε RTL εξομοίωση όπου το κύκλωμα λειτουργεί σωστά όπως φαίνεται στις παρακάτω εικόνες:



Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Adder.vhd CtrlLogic.vhd simulation/modelsim/ctrlLogic_tb.vhd

```

15
16 ARCHITECTURE ctrlLogic_tb_arch OF ctrlLogic_tb IS
17   SIGNAL SL_B : STD_LOGIC ;
18   SIGNAL Rst : STD_LOGIC ;
19   SIGNAL SL_C : STD_LOGIC ;
20   SIGNAL EN_A : STD_LOGIC ;
21   SIGNAL EN_B : STD_LOGIC ;
22   SIGNAL EN_C : STD_LOGIC ;
23   SIGNAL SL_H : STD_LOGIC ;
24   SIGNAL CLK : STD_LOGIC ;
25   SIGNAL SL_L : STD_LOGIC ;
26   SIGNAL EN_H : STD_LOGIC ;
27   SIGNAL EN_L : STD_LOGIC ;
28   SIGNAL SL_R : STD_LOGIC ;
29   SIGNAL monitor_count : STD_LOGIC_VECTOR (n downto 0);
30   signal monitor_state : state_type;
31
32 COMPONENT CtrlLogic
33   GENERIC (
34     n : INTEGER );
35   PORT (
36     SL_B : out STD_LOGIC ;
37     Rst : in STD_LOGIC ;
38     SL_C : out STD_LOGIC ;
39     EN_A : out STD_LOGIC ;
40     EN_B : out STD_LOGIC ;
41     EN_C : out STD_LOGIC ;
42     SL_H : out STD_LOGIC ;
43     CLK : in STD_LOGIC ;
44     SL_L : out STD_LOGIC ;
45     EN_H : out STD_LOGIC ;
46     EN_L : out STD_LOGIC ;
47     SL_A : out STD_LOGIC );
48
49 END COMPONENT ;
50 BEGIN
51   DUT : CtrlLogic
52   GENERIC MAP (
53     n => n )
54   PORT MAP (
55     SL_B => SL_B ,
56     Rst => Rst ,
57     SL_C => SL_C ,
58     EN_A => EN_A ,
59     EN_B => EN_B ,
60     EN_C => EN_C ,
61     SL_H => SL_H ,
62     CLK => CLK ,
63     SL_L => SL_L ,
64     EN_H => EN_H ,
65     EN_L => EN_L ,
66     SL_A => SL_A ) ;

```

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt

System (14) / Processing (106)

100% 00:00:14

Windows Taskbar: Search, Chrome, File Explorer, Start, Task View, Taskbar Buttons, Network, System Tray: 107 ms, ENG, 28/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Adder.vhd CtrlLogic.vhd simulation/modelsim/ctrlLogic_tb.vhd

```

39
40   EN_B : out STD_LOGIC ;
41   EN_C : out STD_LOGIC ;
42   SL_H : out STD_LOGIC ;
43   CLK : in STD_LOGIC ;
44   SL_L : out STD_LOGIC ;
45   EN_H : out STD_LOGIC ;
46   EN_L : out STD_LOGIC ;
47   SL_A : out STD_LOGIC );
48
49 END COMPONENT ;
50 BEGIN
51   DUT : CtrlLogic
52   GENERIC MAP (
53     n => n )
54   PORT MAP (
55     SL_B => SL_B ,
56     Rst => Rst ,
57     SL_C => SL_C ,
58     EN_A => EN_A ,
59     EN_B => EN_B ,
60     EN_C => EN_C ,
61     SL_H => SL_H ,
62     CLK => CLK ,
63     SL_L => SL_L ,
64     EN_H => EN_H ,
65     EN_L => EN_L ,
66     SL_A => SL_A ) ;

```

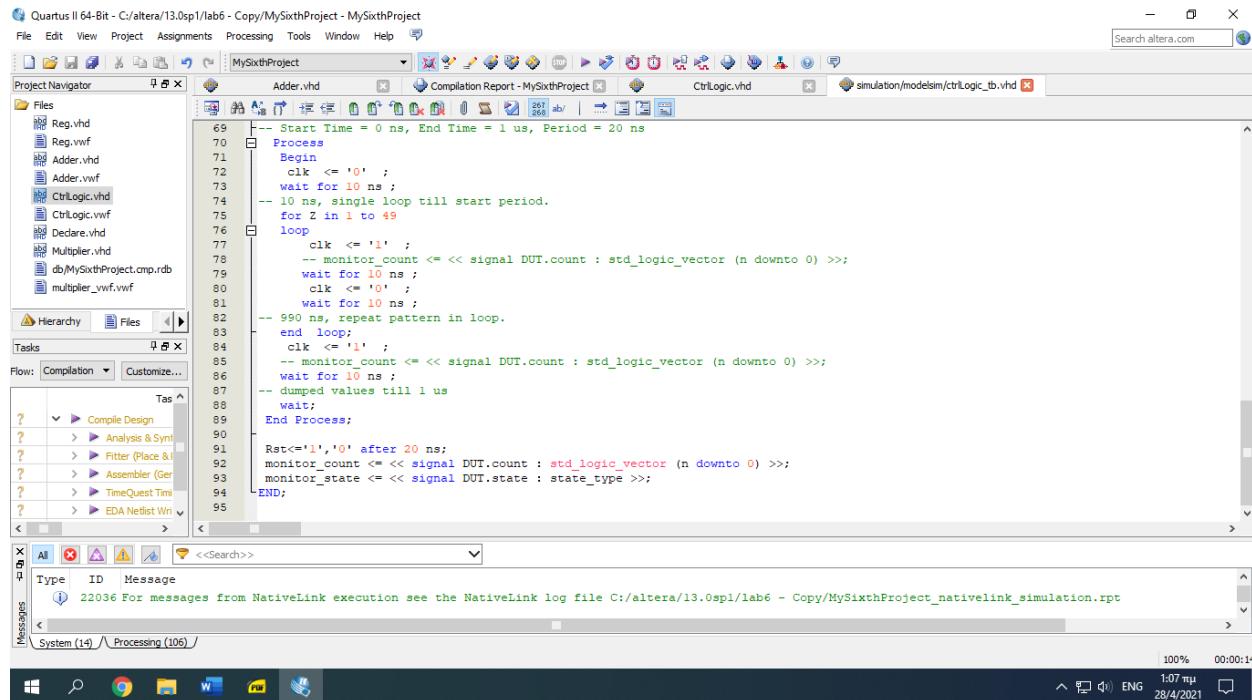
Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt

System (14) / Processing (106)

100% 00:00:14

Windows Taskbar: Search, Chrome, File Explorer, Start, Task View, Taskbar Buttons, Network, System Tray: 107 ms, ENG, 28/4/2021



-Στο κύκλωμα του πολλαπλασιαστή έχουμε προσθέσει στην οντότητα 2 επιπλέον εισόδους A_IN και B_IN που αποτελούν τις παράλληλες εισόδους των 8bit του πολλαπλασιαστή, όπως επίσης έχουμε αλλάξει και την τιμή του n από 4 σε 8bit.

-Στο component CtrlLogic έχουμε αλλάξει την τιμή του n από 4 σε 8bit, η οποία αλλαγή έχει γίνει και στα component Adder, component Reg.

-Στον καταχωρητή A έχουμε βάλει στην παράλληλη είσοδο του D_IN την παράλληλη είσοδο του πολλαπλασιαστή A_IN, όπως επίσης το SI στην τιμή 0(δεν χρησιμοποιείται).

-Στον καταχωρητή B έχουμε βάλει στην παράλληλη είσοδο του D_IN την παράλληλη είσοδο του πολλαπλασιαστή B_IN, όπως επίσης το SI στην τιμή 0(δεν χρησιμοποιείται).

-Στον αθροιστή έχουμε αλλάξει την τιμή του n από 4 σε 8bit.

Το κύκλωμα με τις αλλαγές φαίνεται στις παρακάτω εικόνες:

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Reg.vhd Adder.vhd CtrlLogic.vhd Multiplier.vhd

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
use work.Declarations.all;

entity Multiplier is
    generic (n: integer :=8);
    port (
        Rst, CLK, SI : in std_logic;
        Low, High : out std_logic_vector (n-1 downto 0);
        A_IN, B_IN : in std_logic_vector (n-1 downto 0));
end Multiplier;

architecture RTL_Mul of Multiplier is
begin
    component CtrlLogic
        generic (n: integer := 8 );
        port ( Rst, CLK : in std_logic;
                SL_A, SL_B, SL_H, SL_L, SL_C : out std_logic;
                EN_A, EN_B, EN_H, EN_L, EN_C : out std_logic );
    end component;
    component Adder
        generic (n: integer :=8);
        port ( A, B : in std_logic_vector (n-1 downto 0);
                SUM : out std_logic_vector (n-1 downto 0);
                COUT : out std_logic );
    end component;
    begin
        CtrlLogic: CtrlLogic
            generic map (n=>8)
            port map (Rst, CLK, SI, SL_A, SL_B, SL_H, SL_L, SL_C, EN_A, EN_B, EN_H, EN_L, EN_C);
        Adder: Adder
            generic map (n=>8)
            port map (A_IN, B_IN, SUM, COUT);
    end;
end;

```

Messages

Type ID Message

System Processing

Ln 1 Col 1 VHDL File 0% 00:00:00

1242 μs 28/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Reg.vhd Adder.vhd CtrlLogic.vhd Multiplier.vhd

```

component Adder
    generic (n: integer :=8);
    port ( A, B : in std_logic_vector (n-1 downto 0);
            SUM : out std_logic_vector (n-1 downto 0);
            COUT : out std_logic );
end component;
component Reg
    generic (n: integer:=8);
    port ( D_in: in std_logic_vector (n-1 downto 0);
            SI, CLK, RST, SLOAD, ENABLE: in std_logic;
            SO: out std_logic;
            D_OUT: out std_logic_vector (n-1 downto 0));
end component;
signal SL_A, SL_B, SL_H, SL_L, SL_C, EN_A, EN_B, EN_H, EN_L, EN_C : std_logic;
signal SO_A, SO_H : std_logic;
signal A, B, SUM, H: std_logic_vector(n-1 downto 0);
signal C, COUT : std_logic_vector (0 downto 0);
begin
    R_A: Reg generic map (n=>n)
        port map (D_IN=>A_IN,SI=>'0',CLK=>CLK,RST=>RST,SLOAD=>SL_A,ENABLE=>EN_A,SO=>open,D_OUT=>A);
    R_B: Reg generic map (n=>n)
        port map (D_IN=>B_IN,SI=>'0',CLK=>CLK,RST=>RST,SLOAD=>SL_B,ENABLE=>EN_B,SO=>open,D_OUT=>B);

```

Messages

Type ID Message

System Processing

Ln 1 Col 1 VHDL File 0% 00:00:00

1242 μs 28/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lbd6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator MySixthProject Reg.vhd Adder.vhd CtrlLogic.vhd Multiplier.vhd

Files Reg.vhd Reg.vwf Adder.vhd Adder.wvf CtrlLogic.vhd CtrlLogic.wvf Dclare.vhd Multiplier.vhd db/MySixthProject.cmp.rdb multiplier_vwf.vwf

Hierarchy Files Tasks Flow: Compilation Customize...

Tags

Compile Design

Analyze & Symb

Filter (Place &

Assembler (Gen

TimeQuest Timi

EDA Netlist Wri

47 port map (D_IN=>A_IN, SI=>'0', CLK=>CLK, RST=>RST, SLOAD=>SL_A, ENABLE=>EN_A, SO=>open, D_OUT=>A);
48 R_B: Reg generic map (n=>n)
49 port map (D_IN=>B_IN, SI=>'0', CLK=>CLK, RST=>RST, SLOAD=>SL_B, ENABLE=>EN_B, SO=>open, D_OUT=>B);
50
51 R_C: Reg generic map (n=>n)
52 port map (D_IN=>COUT, SI=>'0', CLK=>CLK, RST=>RST, SLOAD=>SL_C, ENABLE=>EN_C, SO=>open, D_OUT=>C);
53
54 R_H: Reg generic map (n=>n)
55 port map (D_IN=>SUM, SI=>C(0), CLK=>CLK, RST=>RST, SLOAD=>SL_H, ENABLE=>EN_H, SO=>SO_H, D_OUT=>H);
56
57 High<=H;
58
59 R_L: Reg generic map (n=>n)
60 port map (D_IN=>(n-1 downto 0=>'0'), SI=>SO_H, CLK=>CLK, RST=>RST, SLOAD=>SL_L, ENABLE=>EN_L, SO=>open, D_OUT=>Low);
61
62 U_Ad: Adder generic map (n=>8)
63 port map (A=>H, B=>((n-1 downto 0=>B(0)) and A), SUM=>SUM, COUT=>COUT(0));
64
65 U_Ctl: CtrlLogic generic map (n=>8)
66 port map (Rst=>RST, CLK=>CLK, SL_A=>SL_A, SL_B=>SL_B, SL_H=>SL_H, SL_L=>SL_L, SL_C=>SL_C, EN_A=>EN_A, EN_B=>EN_B, EN_H=>EN_H);
67
68
69
70
71
72
73 end RTL_Mul;

<<Search>>

Type ID Message

Messages

System Processing

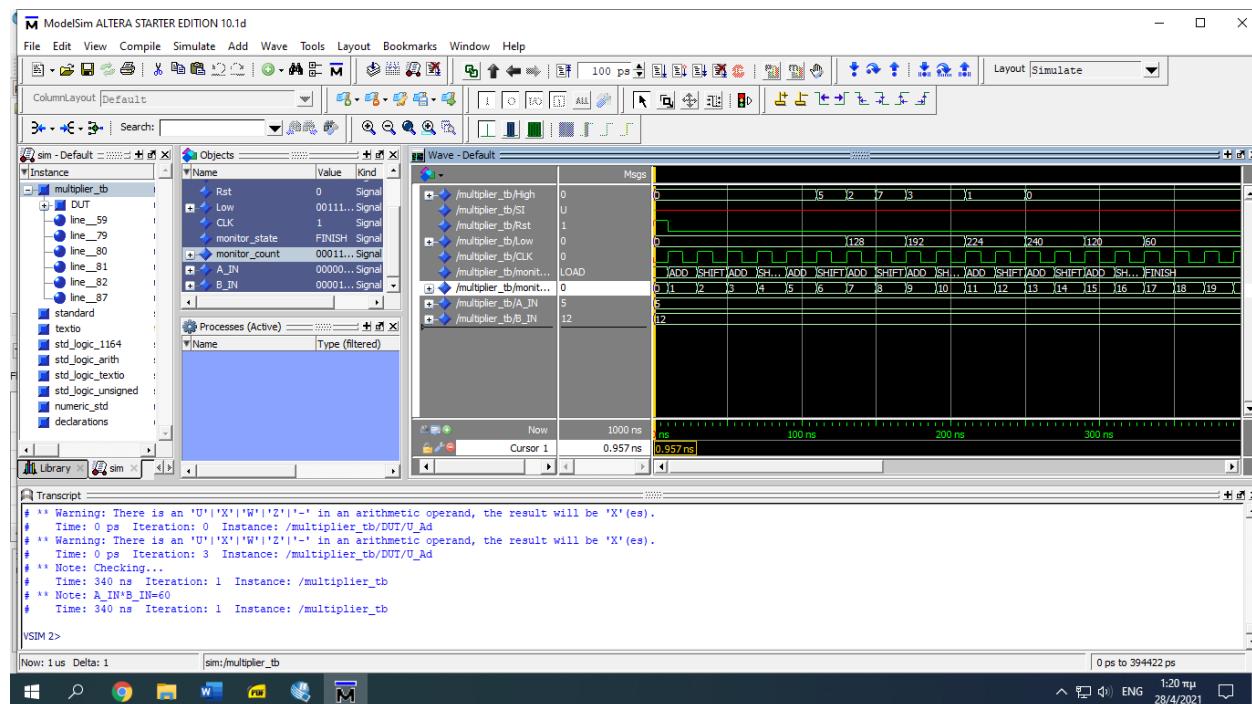
Ln 1 Col 1 VHDL File 0% 00:00:00

12:42 pm 28/4/2021

-Τέλος στο test-bench multiplier_tb αλλάξαμε την τιμή του n από 4 σε 8 bit, προσθέσαμε τα δύο σήματα A_IN και B_IN στην αρχιτεκτονική του πολλαπλασιαστή και στο COMPONENT Multiplier και τα συνδέσαμε στο port map.

-Επιπλέον αρχικοποιήσαμε τα σήματα A_IN και B_IN δίνοντάς τους τυχαίες τιμές των 8 bit και τέλος εκτυπώνουμε το αποτέλεσμα στο transcript

Εκτελέσαμε RTL εξομοίωση και το κύκλωμα λειτουργεί σωστά όπως φαίνεται στις παρακάτω εικόνες:



Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Search altera.com

Project Navigator simulation/modelsim/multiplier_tb.vhd Compilation Report - MySixthProject Multiplier.vhd

Files

- Reg.vhd
- Reg.vwf
- Adder.vhd
- Adder.vwf
- CtrlLogic.vhd
- CtrlLogic.vwf
- Dedare.vhd
- Multiplexer.vhd
- db/MySixthProject.cmp.rdb
- multiplier_vwf.vwf

Hierarchy Files

Tasks Flow: Compilation Customize...

Compile Design Analysis & Synthesis Filter (Place & Route) Assembler (Generate) TimeQuest Timing EDA Netlist Win

ENTITY multiplier_tb IS
 GENERIC (
 n : INTEGER := 8);
 END ;

 ARCHITECTURE multiplier_tb_arch OF multiplier_tb IS
 SIGNAL High : STD_LOGIC_VECTOR (n - 1 downto 0) ;
 SIGNAL SI : STD_LOGIC ;
 SIGNAL Ret : STD_LOGIC ;
 SIGNAL Low : STD_LOGIC_VECTOR (n - 1 downto 0) ;
 SIGNAL CLK : STD_LOGIC ;
 SIGNAL monitor_state : state_type;
 SIGNAL monitor_count : std_logic_vector(n downto 0);
 SIGNAL A_IN : STD_LOGIC_VECTOR (n - 1 downto 0) ;
 SIGNAL B_IN : STD_LOGIC_VECTOR (n - 1 downto 0) ;

Messages

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt

System (21) / Processing (111) /

100% 00:00:11 1:20 np ENG 28/4/2021

The screenshot shows the Quartus II 64-Bit software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, Help, and a search bar for altera.com. The left sidebar contains a Project Navigator with files like Reg.vhd, Reg.wvf, Adder.vhd, Adder.wvf, CtrlLogic.vhd, CtrlLogic.wvf, Dclare.vhd, Multiplier.vhd, and dbMySixthProject.cmp.rdb. Below it is a Hierarchy tree and a Tasks panel with compilation options. The main workspace displays a VHDL code editor for a multiplier_tb_arch architecture. The code defines signals for High, SI, Rst, Low, CLK, and various input and output vectors (A_IN, B_IN). It also includes a component declaration for a Multiplier with generic parameters n and m, and a port map section. The bottom pane shows a Messages window with a single entry: "22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt". The status bar at the bottom right shows the date and time as 1:20 pm, 28/4/2021.

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator simulation/modelsim/multiplier_tb.vhd Compilation Report - MySixthProject Multiplier.vhd

Search altera.com

Files

- Reg.vhd
- Reg.vwf
- Adder.vhd
- Adder.vwf
- CtrlLogic.vhd
- CtrlLogic.vwf
- Dedare.vhd
- Multipiler.vhd
- db/MySixthProject.cmp.rdb
- multiplier_wvf.wvf

Hierarchy Files

Tasks

Flow: Compilation Customize...

Tags

- Compile Design
- Analysis & Synthesis
- Fitter (Place & Route)
- Assembler (Generate)
- TimeQuest Timing
- EDA Netlist Writing

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt

System (21) / Processing (111) /

```

40     END COMPONENT ;
41 BEGIN
42     DUT : Multiplier
43     GENERIC MAP (
44         n => n )
45     PORT MAP (
46         High => High ,
47         SI => SI ,
48         Rst => Rst ,
49         Low => Low ,
50         CLK => CLK ,
51         A_IN => A_IN,
52         B_IN => B_IN );
53
54
55     -- "Clock Pattern" : dutyCycle = 50
56     -- Start Time = 0 ns, End Time = 1 us, Period = 20 ns
57     Process
58     Begin
59         clk <= '0';
60         Rst<='1';
61         wait for 10 ns;
62         -- 10 ns, single loop till start period.
63         for Z in 1 to 49
64             loop
65                 clk <= '1';
66             end loop;
67         end loop;
68     end Process;
69
70     A_IN<="00000101";
71     B_IN<="00001100";
72     monitor_state <= << signal DUT.U_Ctl.state : state_type >>;
73     monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
74
75     clk <= '1';
76     wait for 10 ns;
77     -- dumped values till 1 us
78     wait;
79     End Process;
80
81     A_IN<="00000101";
82     B_IN<="00001100";
83     monitor_state <= << signal DUT.U_Ctl.state : state_type >>;
84     monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
85
86     process
87     begin
88         wait on monitor_state;
89         if (monitor_state = FINISH) then
90             wait on clk;
91             assert (FALSE) report "Checking..." severity note;
92             --assert (Low&"0011" and High&"0010") report "Check Failed" severity error;
93             assert (A_IN&B_IN=High & Low) report "Check Failed" severity error;
94             report "A_IN&B_IN=" & to_string(conv_integer(A_IN&B_IN));
95         end if;
96     end process;
97
98     END;
99

```

100% 00:00:13

1:20 ms ENG 28/4/2021

Quartus II 64-Bit - C:/altera/13.0sp1/lab6 - Copy/MySixthProject - MySixthProject

File Edit View Project Assignments Processing Tools Window Help

Project Navigator simulation/modelsim/multiplier_tb.vhd Compilation Report - MySixthProject Multiplier.vhd

Search altera.com

Files

- Reg.vhd
- Reg.vwf
- Adder.vhd
- Adder.vwf
- CtrlLogic.vhd
- CtrlLogic.vwf
- Dedare.vhd
- Multipiler.vhd
- db/MySixthProject.cmp.rdb
- multiplier_wvf.wvf

Hierarchy Files

Tasks

Flow: Compilation Customize...

Tags

- Compile Design
- Analysis & Synthesis
- Fitter (Place & Route)
- Assembler (Generate)
- TimeQuest Timing
- EDA Netlist Writing

Messages

Type ID Message

22036 For messages from NativeLink execution see the NativeLink log file C:/altera/13.0sp1/lab6 - Copy/MySixthProject_nativelink_simulation.rpt

System (21) / Processing (111) /

```

73     clk <= '1';
74     wait for 10 ns;
75     -- dumped values till 1 us
76     wait;
77     End Process;
78
79     A_IN<="00000101";
80     B_IN<="00001100";
81     monitor_state <= << signal DUT.U_Ctl.state : state_type >>;
82     monitor_count <= << signal DUT.U_Ctl.count : std_logic_vector(n downto 0) >>;
83
84
85     process
86     begin
87         wait on monitor_state;
88         if (monitor_state = FINISH) then
89             wait on clk;
90             assert (FALSE) report "Checking..." severity note;
91             --assert (Low&"0011" and High&"0010") report "Check Failed" severity error;
92             assert (A_IN&B_IN=High & Low) report "Check Failed" severity error;
93             report "A_IN&B_IN=" & to_string(conv_integer(A_IN&B_IN));
94         end if;
95     end process;
96
97     END;
98

```

100% 00:00:13

1:20 ms ENG 28/4/2021