



VHDL Simulation Synthesis

- Synopsys Tool -

System ASIC Design Lab.

:

jcho@asiclab.inchon.ac.kr

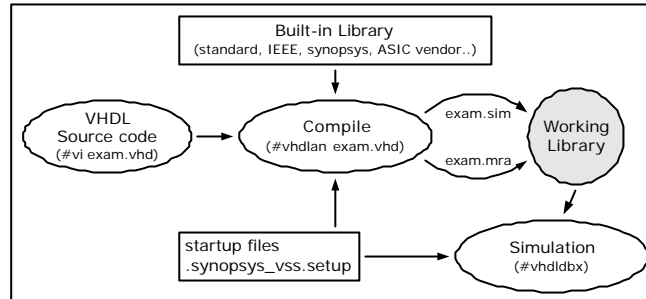


- ✍ -I
- ✍ -1 : 1-Bit Full Adder Simulation
 - ✍ Synopsys Simulation Simulation Tool
- ✍ -2 : 1-Bit Full Adder Synthesis
 - ✍ Synopsys Synthesis Synthesis Tool
- ✍ -3 : 1-Bit Serial Adder
 - ✍ Synopsys Tool
- ✍ -4 : 3-tap FIR Filter
 - ✍
- ✍ -5 : Counter 7-Segment Display
 - ✍
- ✍ -II
- ✍ -6 : Traffic Light Controller
 - ✍ Digital System RTL
 - ✍ VHDL
 - ✍ Synopsys Tool
 - ✍ Max+PlusII
 - ✍ FPGA



Simulation

☞ Synopsys VSS Simulator



- ☞ VHDL compile : \$ vhdlan {vhd_source_file_name}
- ☞ Built-in Library
 - ☞ standard, IEEE, synopsys, ASIC vendor, your own...
- ☞ Working Library
 - ☞ VHDL code compile (*.sim, *.mra)
- ☞ Simulation : \$ vhldbx
- ☞ setup file
 - ☞ .synopsys_vss.setup



VSS Simulator (1)

☞ .synopsys_vss.setup file

```

☞ Working_Library_Name Working_Library_Directory User가
☞ Working_Library_Name : Compile Symbolic Name
☞ Working_Library_Directory : Working Library Directory Path
  
```

```

TIMEBASE = NS
WAVEFORM = waves
WORK_____ > Working_Library_Name
Working_Library_Name_____ : Working_Library_Directory
EDITCMD = xterm -geom 92x60 -T Synopsys-Editor -e vi
  
```

☞

```

☞ User Home Directory sim_work directory Unix
  
```

```
$ mkdir ~/sim_work
```

☞ .synopsys_vss.setup file

```

TIMEBASE = NS
WAVEFORM = waves
WORK_____ > sim_work
sim_work _____ : ~/sim_work
EDITCMD = xterm -geom 92x60 -T Synopsys-Editor -e vi
  
```



-1 : Full Adder

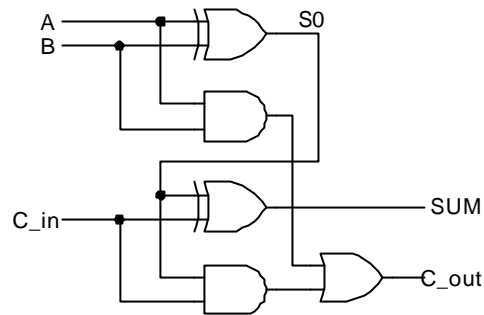
Data flow

VHDL

✍ Data Flow

```

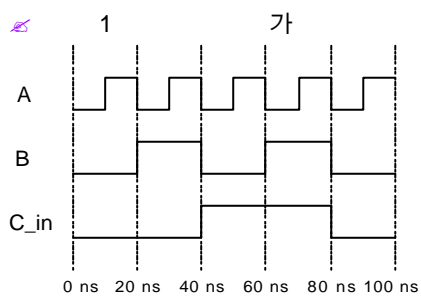
library IEEE;
use IEEE.Std_logic_1164.all;
entity fa is
  port (
    A : in std_logic;
    B : in std_logic;
    C_in : in std_logic;
    C_out: out std_logic;
    Sum : out std_logic);
end fa;
architecture rtl of fa is
  signal S0 : std_logic;
begin
  S0 <= A xor B;
  Sum <= S0 xor C_in;
  C_out <= (A and B) or (S0 and C_in);
end rtl;
  
```



-1 : Stimulus Vector

VHDL

✍



✍ Stimulus Vector

```

library IEEE;
use IEEE.Std_logic_1164.all;
entity vec is
  port (A, B, C_in : out std_logic);
end vec;
--
architecture rtl of vec is
begin
  
```

```

process
begin
  A <= '0'; B <= '0'; C_in <= '0';
  wait for 10 ns;
  A <= '1'; B <= '0'; C_in <= '0';
  wait for 10 ns;
  A <= '0'; B <= '1'; C_in <= '0';
  wait for 10 ns;
  A <= '1'; B <= '1'; C_in <= '0';
  wait for 10 ns;
  A <= '0'; B <= '0'; C_in <= '1';
  wait for 10 ns;
  A <= '1'; B <= '0'; C_in <= '1';
  wait for 10 ns;
  A <= '0'; B <= '1'; C_in <= '1';
  wait for 10 ns;
  A <= '1'; B <= '1'; C_in <= '1';
  wait for 10 ns;
end process;
  
```

```

end rtl;
  
```


-1 : Compile **File**

Compile Error ~/sim_work Directory

단일기

항변 편집기 선택사항 도움말

```

[yhlee@ulitina:~/home4/yhlee/IDEC/VHDL/FA] ls -asF
총 18      2 ../          2 tb_fa.vhd
2 ./      2 fa.vhd       2 vec.vhd
[yhlee@ulitina:~/home4/yhlee/IDEC/VHDL/FA] ls -asF ~/sim_work
총 182     18 FA.sim      2 VEC.nra
2 ./      14 FA_RTL.sim  8 VEC.sim
6 ../     2 TB_FA.nra   18 VEC_RTL.sim
8 CONF.sim 8 TB_FA.sim   2 sparc085/
2 FA.nra  28 TB_FA_RTL.sim
[yhlee@ulitina:~/home4/yhlee/IDEC/VHDL/FA]

```

[종료]

Univ. of Incheon System ASIC Design Lab. Synopsys VHDL NO-11

-1 : Test Bench Compile **Loading**

Simulation Engine Test Bench VHDL Loading

\$ vhldbx & Window가

Loading Window

Working Library

Design

Simulation

Vhdlbxb - Select Simulator Arguments

Library	Design
STH_WORK	TOP
SYNOPSYS	FA_RTL
DEFAULT	TB_FA_RTL
ISARE	VEC_RTL
CSOOMP	
GTECH	
NI	
IDEE ASIC	

Design: STH_WORK_CONF

Time Units: NS

Arguments:

[OK] [Cancel]

Simulation Configuration Entity_Architecture_name (configuration)

Univ. of Incheon System ASIC Design Lab. Synopsys VHDL NO-12

-1 : Test Bench Compile

Loading

☞ Loading

Window

The screenshot shows the Synopsys VHDL Debugger interface. The main window displays the VHDL source code for a test bench. The code includes library declarations, entity declarations, and component instantiations. A blue arrow points from the text 'VHDL Top Model Source code' to the source code window. Below the source code, there is a command line area where the command 'cd U1' is entered. A blue circle highlights this command line, with a blue arrow pointing to the text 'Command line'. The interface also shows a toolbar with buttons for 'Stop at', 'Clear', 'Trace', and 'Event Bpt.'.



-1 : VHDL Debugger Command

☞

Command

(1)

☞ **trace ***

☞ source code

☞ **cd label : cd U1**

☞ sub-block

☞ **trace signal_name : trace s0**

☞

☞ **cd ..**

☞ block

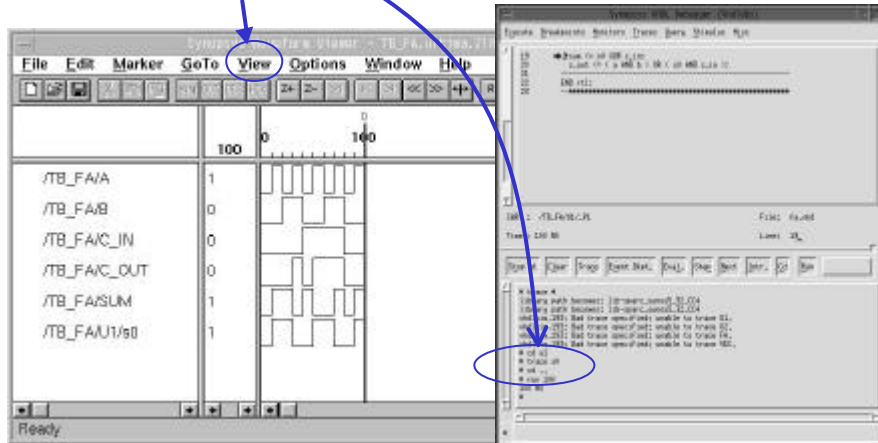
The screenshot shows a list of blocks in the Synopsys VHDL Debugger. The list includes: /TB_FA/A, /TB_FA/B, /TB_FA/C_IN, /TB_FA/C_OUT, /TB_FA/D000, /TB_FA/D001, and /TB_FA/D002. Blue arrows point from the text 'sub-block' and 'block' to the corresponding entries in the list.

The screenshot shows the Synopsys VHDL Debugger interface with the source code window displaying the VHDL code. The command line area shows the command 'cd U1' entered. The interface also shows a toolbar with buttons for 'Stop at', 'Clear', 'Trace', 'Event Bpt.', 'Eval.', 'Stop', 'Step', 'Intr.', 'Cd', and 'Run'.



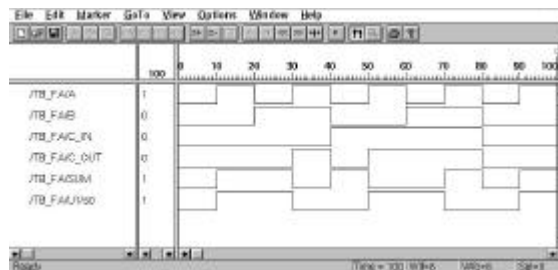
-1 : Simulation Engine

- ☞ Command (2)
- ☞ **run 100**
 - ☞ Simulation Engine 100 ns Simulation
- ☞ waveform scale
 - ☞ >tool-bar menu View sub-menu



-1 : Simulation Waveform

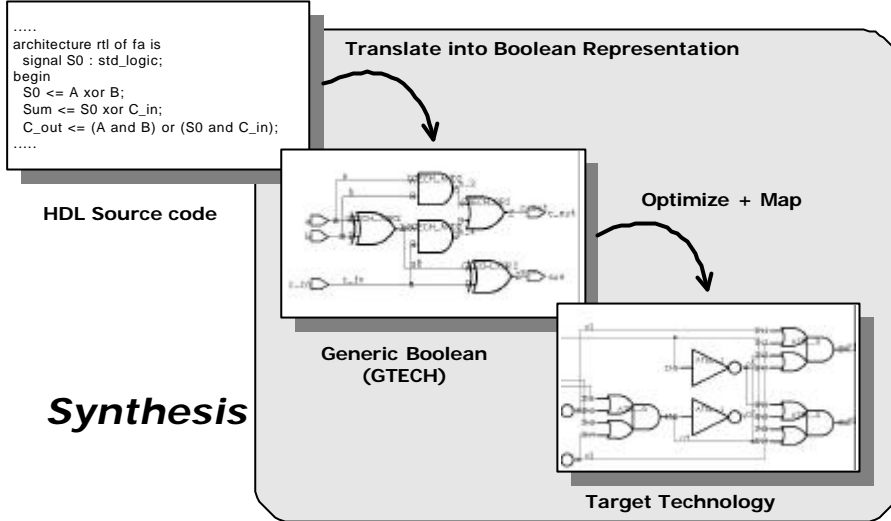
- ☞ Full Fit ()
- ☞ Zoom In ()
- ☞ Zoom Out ()
- ☞ Zoom Fit ()



-2 : VHDL

(Logic Synthesis) ?

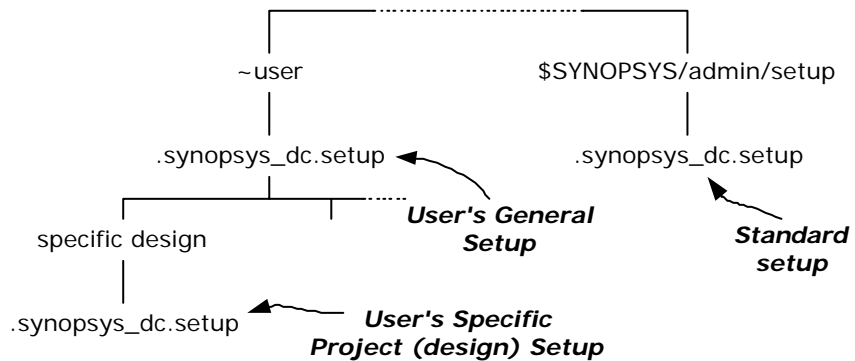
Synthesis = Translation + Optimization + Mapping



Synopsys Synthesis Tool

File

- Design Compiler Setup File : **.synopsys_dc.setup**
- Setup File 가
- Setup File : Synopsys Tool Default Setup File .
- (account) :
- Directory : Directory Design Model .



Synopsys Synthesis Tool

☞ .synopsys_dc.setup File

☞ 가 : Target Library ALTERA FLEX10K

```

#* From the System Variable Group *#
#*search_path = { /home2/maxplus2/synopsys/library/alt_syn/flex0000/lib /home2/synopsys/libraries/sy
n /home2/maxplus2/synopsys/library/alt_nf };*#
search_path = { /tools2/max8.3/synopsys/library/alt_syn/flex10k/lib #
               /tools/synopsys/libraries/syn #
               /tools2/max8.3/synopsys/library/alt_nf #
               /tools2/max8.3/max21b/mega_ipa #
               /tools2/max8.3/simlib/concept/alt_ipn };

target_library = {flex10k.db};
symbol_library = {altera.sdb};
link_library = {"*" flex10k.db};

define_design_lib altera -path #MAXPLUS/synopsys/library/alt_nf/lib
define_design_lib DM_FLEX10K -path #MAXPLUS/synopsys/library/alt_syn/flex10k/lib/dm_flex10k
define_design_lib DM_FLEX10K_FPGA -path #MAXPLUS/synopsys/library/alt_syn/flex10k/lib/dm_flex10k_fpga

adifout_netlist_only = "true";
adifout_power_and_ground_representation = "net";
adifout_no_array = "false";
adifout_power_net_name = "VDD";
    
```

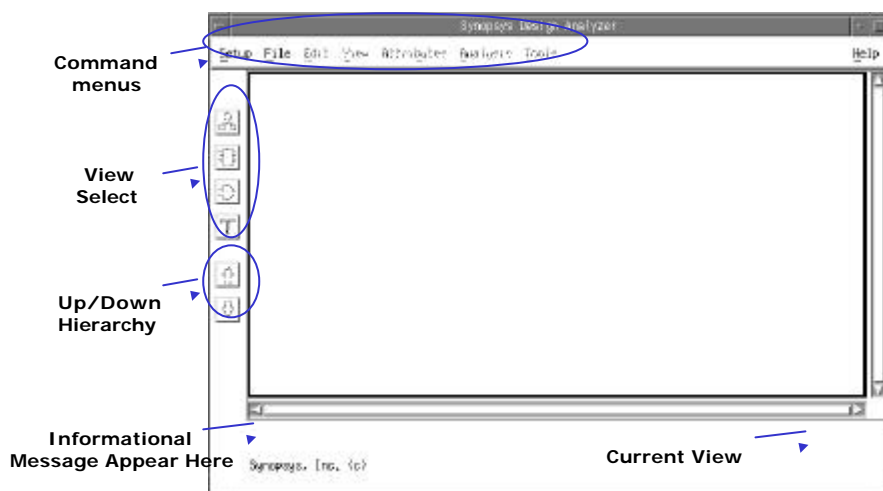


-2 : Design Analyzer Window

☞ Design Analysis Tool Loading Unix Command

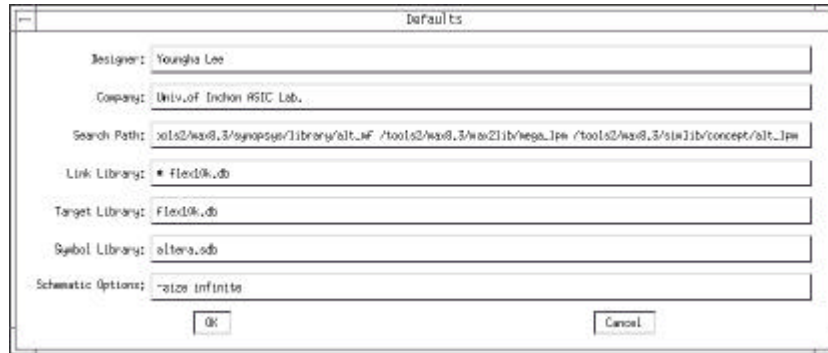
☞ \$ **design_analyzer &**

Window가



-2 : Check Design Compiler Setup

- ✍ Design Analysis Command Menu Setup Defaults..
- ✍ >Setup > Defaults Window가 Call
- ✍ .synopsys_dc.setup File Loading

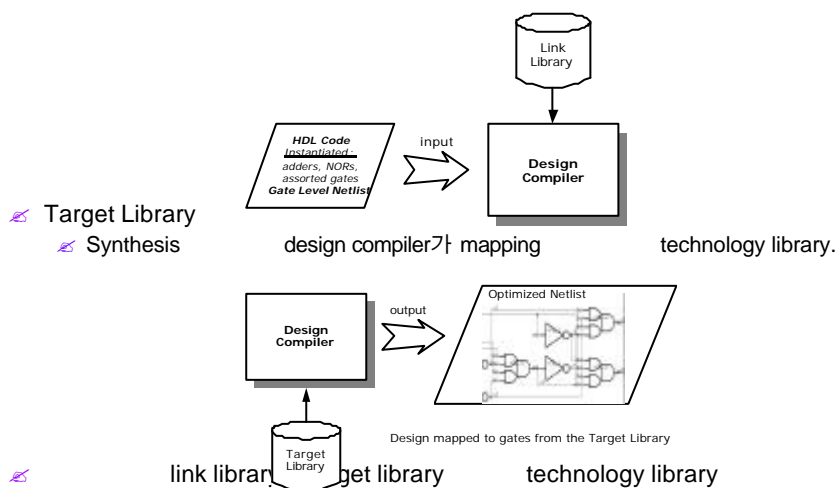


- ✍
- ✍ Link Library
- ✍ Target Library
- ✍ Symbol Library



-2 : Technology Library

- ✍ Link Library
- ✍ Optimization Design Compiler cells wireloads, technology library.



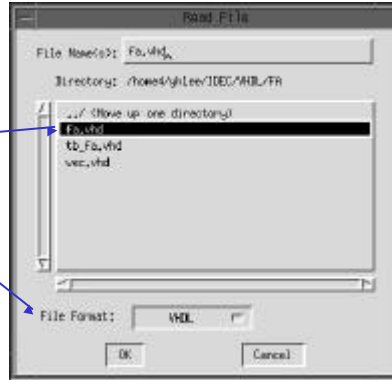
-2 : Design Compiler File (1)

- ☞ Design Analysis Command Menu File Read..
- ☞ >File > Read Window 가
- ☞ File Name File Format , OK

Reading Design File

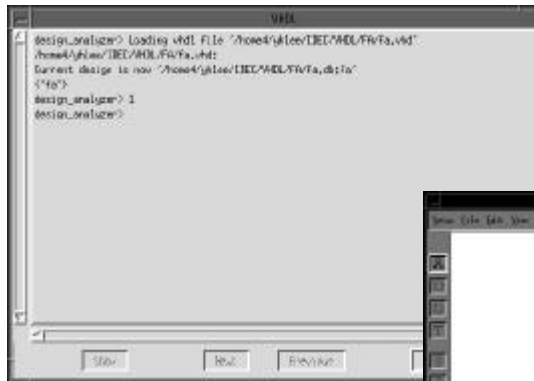
Supports many different formats

Format	Description
db	Synopsys Internal Format
edif	Electronics Design Interchange format
equation	Synopsys Equation Format
lsi	LSI Logic Corporation Format
mif	Mentor Intermediate Format
pla	Berkeley (Expresso) PLA Format
st	Synopsys State Table Format
tdl	Tegas Design Language Format
verilog	Cadence Design Systems Inc. HDL
vhdl	IEEE Standard VHDL
xnf	Xilinx Netlist Format



-2 : Design Compiler File (2)

- ☞ File Read Window 가



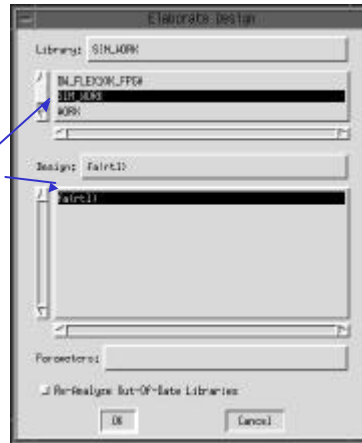
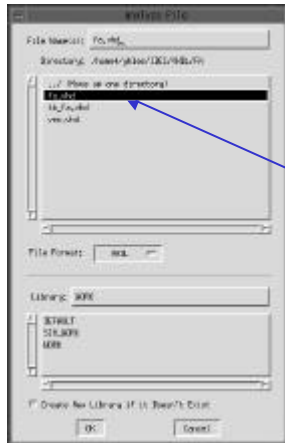
- ☞ Error가 Window
- ☞ Symbol VHDL Code



-2 : HDL Files

Translate

- ✍ File Read
 - ✍ Analyze Elaborate
- ✍ Design Analysis Command Menu
 - ✍ >File > Analyze
 - ✍ >File > Elaborate

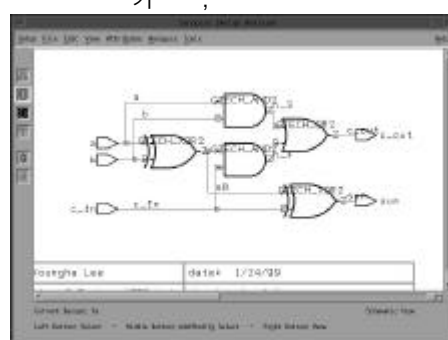


-2 : HDL Files

Translate

- ✍ Translate
 - ✍ Symbol Window

- ✍ Symbol
 - ✍ Symbol Double Click
 - ✍ Target Library

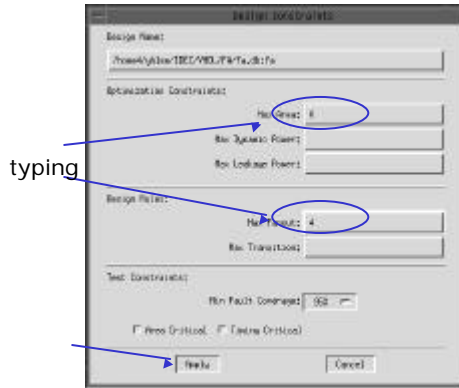


-2 : Optimization Constraints

Optimization Design Block

>Attributes > Optimization Constraints > Design Constraints

Window



typing

Optimization Constraints

Max.Area Area 0

Vendor Technology Library Target

e.g. : 2-input-NAND-gate, inverters, transistors, or square mils, etc.

Design Rule Constraints

Max Fanout

Optimization Fanout



-2 : Optimization

Optimization Design Block

>Tools > Design Optimization

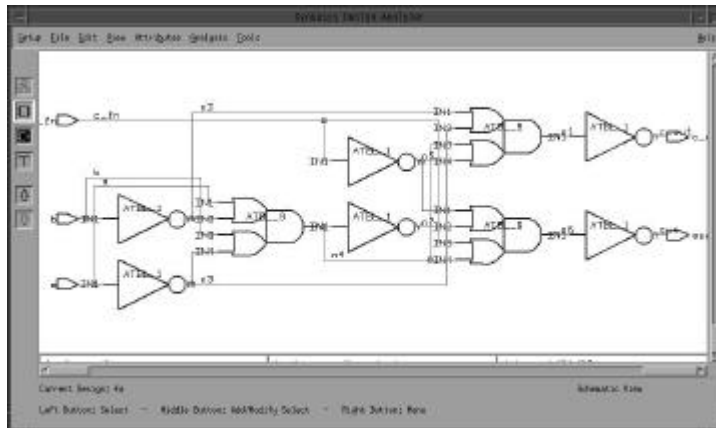
Design Optimization Window "OK"

Optimization Window 가



-2 : Optimization

- ☞ Symbol Target Library
- ☞ Symbol Double Click
- ☞ VHDL Modeling
 - ☞ Behavioral Modeling :
 - ☞ Structural Modeling : Component Symbol



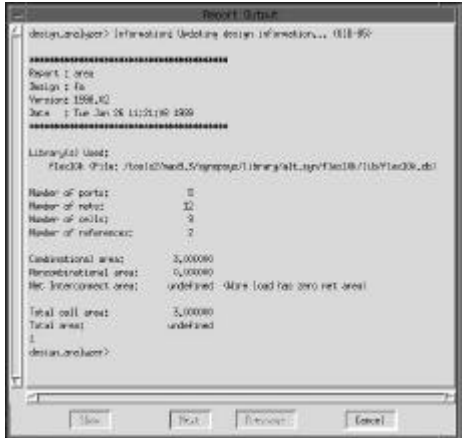
-2 : Optimization Report (1)

- ☞ Cell , Delay Critical Path
- ☞ Command Menu >Analysis>Report Window 71



-2 : Optimization Report (2)

Area Report File
Window Area



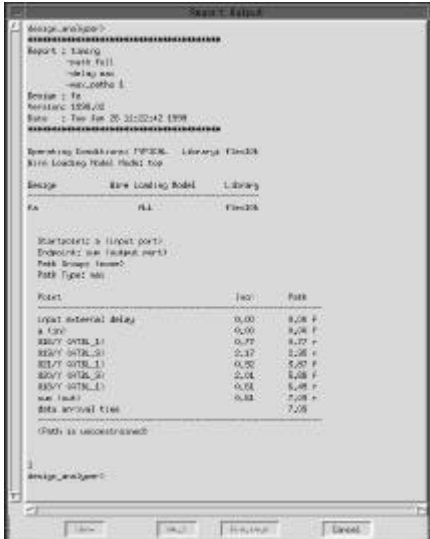
```

design_analyzer> Information: Updating design information... 0018-95
-----
Report : area
Design : fa
Version: 1508_02
Date   : Tue Jan 26 11:21:00 1998
-----
Library(s) Used:
  Flib208: 0file: /tools/ncad.5/synopsys/lib/alt_syn/flib208/flib208.dbi

Number of ports:      0
Number of nodes:     12
Number of cells:      3
Number of references: 2

Conditional area:    3,00000
Hierarchical area:   0,00000
Net Interconnect area: undefined - Wire load has zero net area
Total cell area:     3,00000
Total area:          undefined
design_analyzer>
                    
```

Delay Report File
Window Timing



```

design_analyzer>
-----
Report : Timing
Path Full
Delay Max
--_path: 1
Design : fa
Version: 1508_02
Date   : Tue Jan 26 11:22:42 1998
-----
Timing tool: Flib208 Library: Flib208
Wire Load: Model: Model: top
Design: Wire Load: Model: Library:
fa ALL Flib208

Startpoint: a (input port)
Endpoint: m (output port)
Peak Driver: fa:m0
Path Type: 0

Path:
-----
Port: Two: Path:
Output internal delay: 0,00 0,00 F
a (in): 0,00 0,00 F
0007/0126_1: 0,77 0,77 +
0007/0126_0: 2,17 2,08 +
0017/0126_1: 0,20 0,20 F
0017/0126_0: 2,04 0,08 F
0007/0126_1: 0,04 0,08 F
m (out): 0,04 0,08 +
Data arrival time: 7,28
Path is unconstrained
design_analyzer>
                    
```

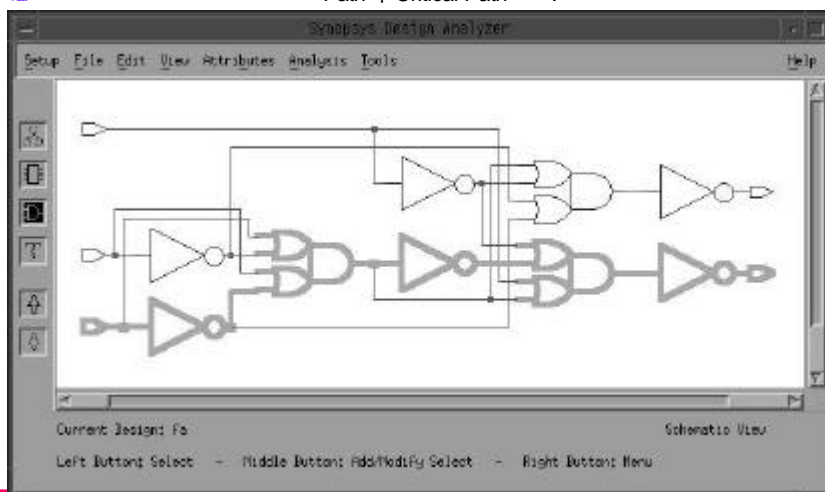
-2 : Optimization Critical Path

Path Command Menu
>Analysis>Highlight>Critical Path

Timing Delay가 가

Path가 Critical Path

Path Window 가



-2 :

☞ Synthesis

☞ ALTERA Library

edif

file

file



☞ Recommendation : Always use the save ALL Designs in Hierarchy option.



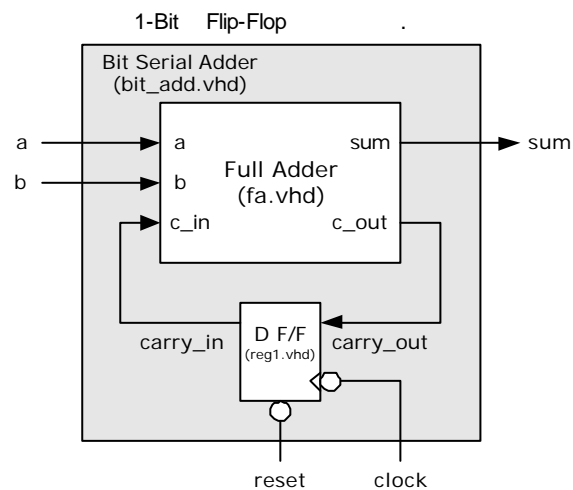
-3 : 가

☞ 1 VHDL

Full Adder

N-Bit 가

☞ carry



-3 : 가 VHDL

```

1-Bit Flip-Flop VHDL
library IEEE; use IEEE.std_logic_1164.all;
entity reg1 is
  port ( reset, clock: in std_logic;
        d_in      : in std_logic;
        d_out     : out std_logic);
end reg1;
architecture rtl of reg1 is
begin
  process ( reset, clock, d_in )
  begin
    if reset = '0' then
      d_out <= '0';
    elsif clock = '0' and clock'event then
      d_out <= d_in;
    end if;
  end process;
end rtl;

```

```

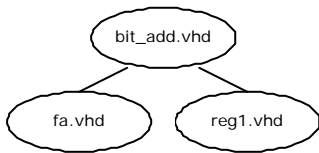
가 VHDL
library IEEE; use IEEE.std_logic_1164.all;
entity bit_add is
  port ( a, b, reset, clock : in std_logic;
        sum : out std_logic);
end bit_add;
architecture rtl of bit_add is
  component fa
  port ( a, b, c_in : in std_logic;
        sum, c_out : out std_logic);
  end component;
  component reg1
  port ( reset, clock, d_in : in std_logic;
        d_out : out std_logic);
  end component;
  for u0 : fa use entity work.f a(rtl);
  for u1 : reg1 use entity work.reg1(rtl);
  signal carry_in, carry_out : std_logic;
begin
  u0 : fa port map (a, b, carry_in, sum, carry_out);
  u1 : reg1 port map (reset, clock, carry_out, carry_in);
end rtl;

```

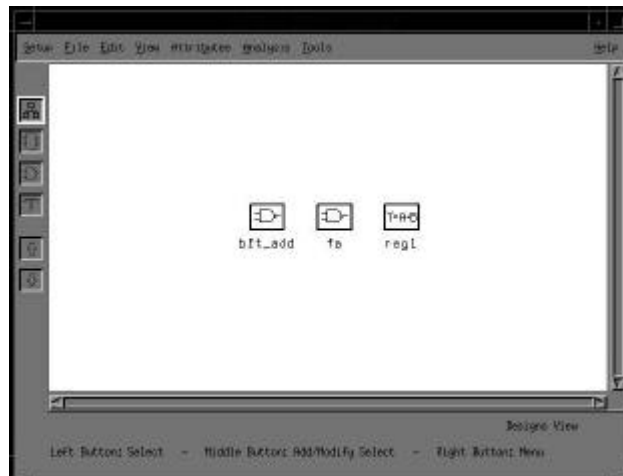


-3 : 가 File Read

- 1
- >File>Read>fa.vhd
- >File>Read>reg1.vhd
- >File>Read>bit_add.vhd



3가 VHDL Code File Read
 VHDL Code File Read



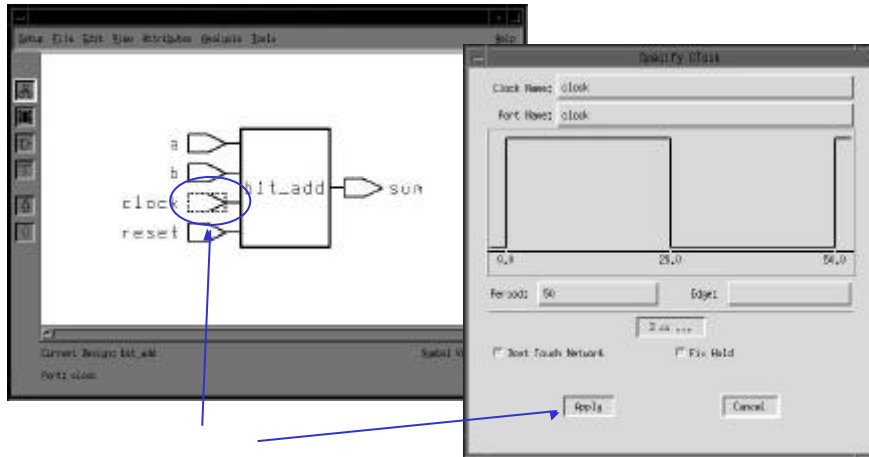
-3 : Clock

✎ Clock attributes

✎ Design Analyzer clock

✎ Design Analysis Command Menu Attributes Clock, Specify

✎ **>Attributes > Clock > Specify**



-3 : Design Constraint

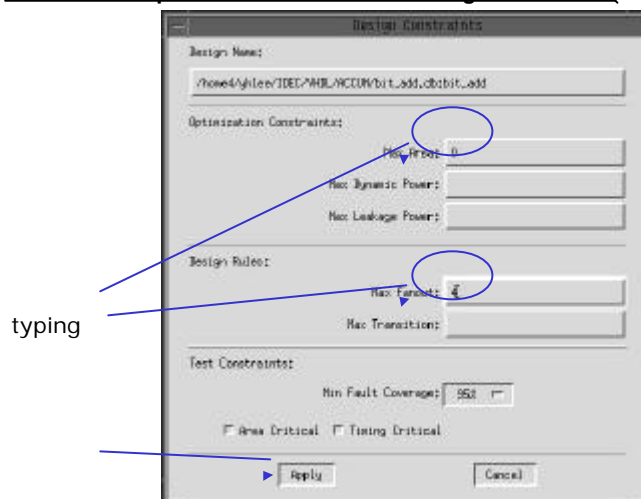
✎ Optimization

Optimization

Design Rule

✎ symbol

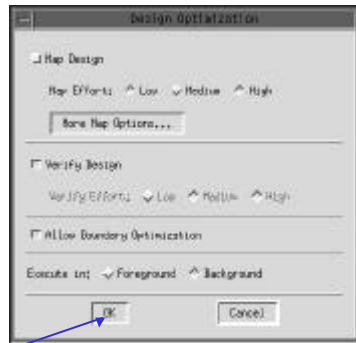
✎ **>Attributes > Optimization Constraints > Design Constraints**



-3 : Optimization

Optimization Design Block

>Tools > Design Optimization



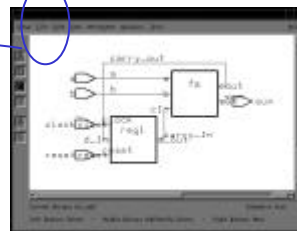
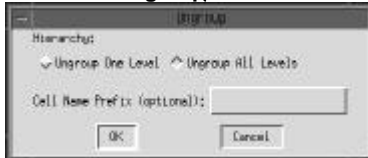
- Design Optimization Window "OK"
- Optimization Window 가
- Cell Delay
- Report File



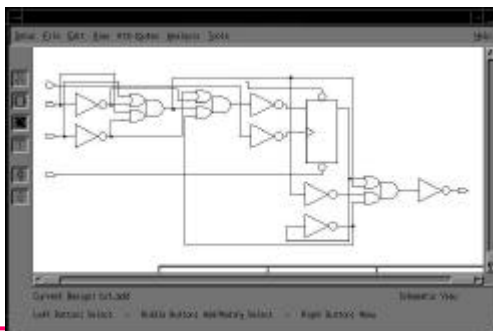
-3 : 1-Level

1-Level

>Edit > Ungroup



1-Level



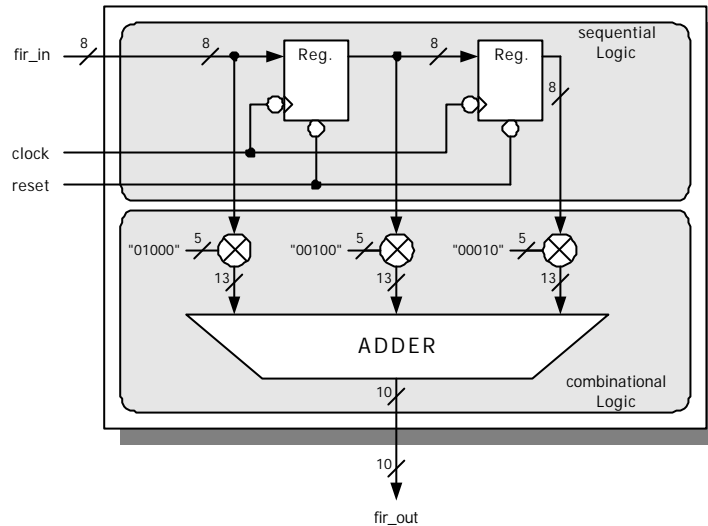
-4 : 3-Tap FIR Filter RTL

3-Tap FIR Filter

RTL

VHDL

, Simulation Synthesis



-4 : 3-Tap FIR Filter VHDL

```

library IEEE; use IEEE.std_logic_1164.all;
use IEEE.std_logic_signed.all;
entity fir is
  port ( fir_in : in std_logic_vector(7 downto 0);
        clock : in std_logic;
        reset : in std_logic;
        fir_out : out std_logic_vector(9 downto 0));
end fir;
--
architecture rtl of fir is
  signal t_1, t_2 : std_logic_vector(7 downto 0);
-- filter
  constant a1 : std_logic_vector(4 downto 0) :=
    "01000";
  constant a2 : std_logic_vector(4 downto 0) :=
    "00100";
  constant a3 : std_logic_vector(4 downto 0) :=
    "00010";
begin
  --filter register
  process(clock,reset) begin
    if reset = '0' then
      t_1 <= (others => '0');
      t_2 <= (others => '0');
    elsif clock'event and clock = '0' then
      t_1 <= fir_in;
      t_2 <= t_1;
    end if;
  end process;
--filter
  process(fir_in, t_1, t_2)
    variable b_0, b_1, b_2, tmp :
      std_logic_vector(12 downto 0);
  begin
    b_0 := a1 * fir_in;  b_1 := a2 * t_1;
    b_2 := a3 * t_2;  tmp := b_0 + b_1 + b_2;
    fir_out <= tmp(12 downto 3);
  end process;
end rtl;

```



-4 : FIR Filter

Test Bench VHDL

```
library IEEE; use IEEE.std_logic_1164.all;
entity tb_fir is end tb_fir;
--
architecture rtl of tb_fir is
--
  component fir
  port ( fir_in  : in std_logic_vector(7 downto 0);
        clock   : in std_logic;
        reset   : in std_logic;
        fir_out : out std_logic_vector(9 downto 0));
  end component;
--
  for u0: fir use entity work.fir(rtl);
--
  signal fir_in  : std_logic_vector(7 downto 0);
  signal clock, reset : std_logic := '0';
  signal fir_out : std_logic_vector(9 downto 0);
--
begin
  u0: fir port map ( fir_in, clock, reset, fir_out);
```

```
clock <= not clock after 20 ns;
--
reset <= '1' after 10 ns;
--
process
begin
  fir_in <= "00010000", "00101110" after 100 ns,
           "00110001" after 200 ns,
           "01000111" after 300 ns,
           "00001010" after 400 ns,
           "00011100" after 500 ns,
           "00100001" after 600 ns,
           "00000111" after 700 ns;

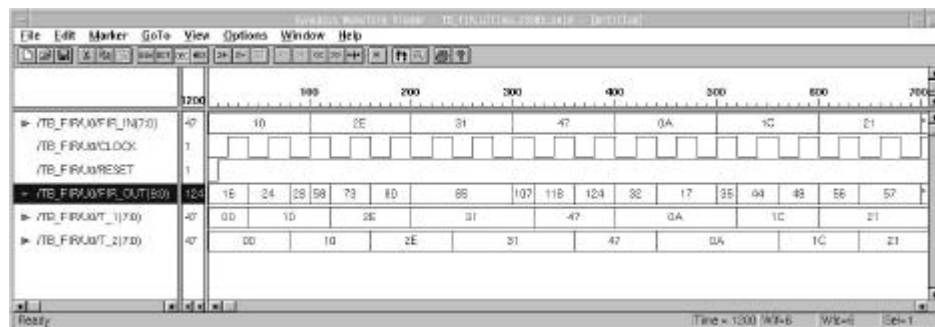
  wait for 800 ns;
end process;
end rtl;
--
configuration conf of tb_fir is
  for rtl end for;
end conf;
```



-4 : Synopsys Simulator

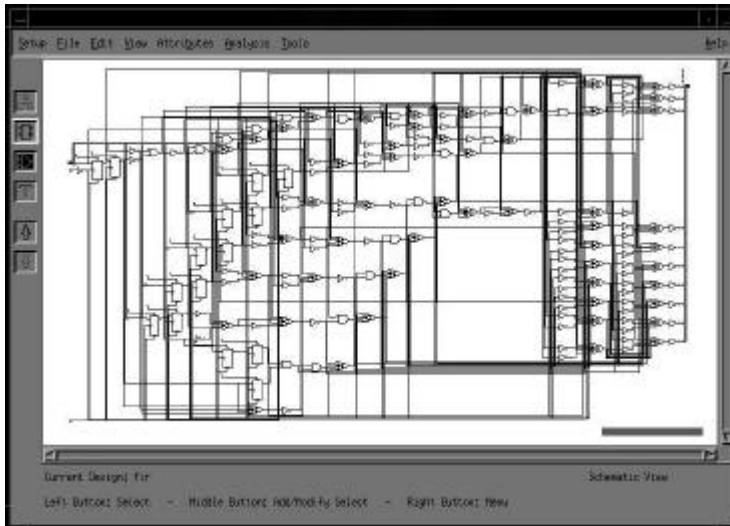
VHDL

Waveform



-4 : Synopsys Synthesis Tool

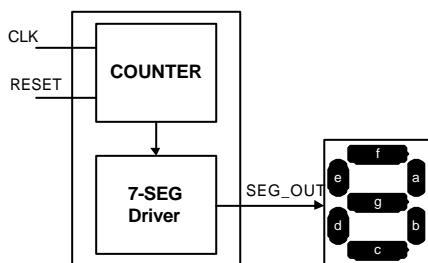
Report File



-5 : Synthesis Tool

Simulation Tool

10 Counter
10 Counter Seven-Segment



10 Counter

RESE T	CLK	CNT_OU T(n)	CNT_OUT(n+1)
1	X	X	"0000"
0	?	"1001"	"0000"
0	?	"1001"	CNT_OUT(n) + "1"

Segment

CNT_OUT	SEG_OUT
"0000"	"1111110"
"0001"	"1100000"
"0010"	"1011011"
"0011"	"1110011"
"0100"	"1100101"
"0101"	"0110111"
"0110"	"0111111"
"0111"	"1100110"
"1000"	"1111111"
"1001"	"1110111"



-5 : Behavioral Level VHDL

Counter VHDL

```
library IEEE; use IEEE.Std_logic_1164.all;
use IEEE.Std_logic_unsigned."+";
entity Counter is
  port ( RESET : in std_logic;
        Clk : in std_logic;
        CNT_out : out std_logic_vector(3 downto 0));
end Counter;
architecture Counter_Arch of Counter is
  signal CNT : std_logic_vector(3 downto 0);
begin
  process(RESET, CLK)
  begin
    if RESET = '1' then
      -- same expression of CNT <= "0000";
      CNT <= (others => '0');
    elsif Clk'event and Clk = '1' then
      if CNT = "1001" then
        CNT <= (others => '0');
      else
        CNT <= CNT + '1';
      end if;
    end if;
  end process;
  CNT_out <= CNT;
end Counter_Arch;
```

Segment VHDL

```
library IEEE; use IEEE.Std_logic_1164.all;
entity Segment is
  port ( Seg_IN : in std_logic_vector(3 downto 0);
        Seg_OUT : out std_logic_vector(6 downto 0));
end Segment;
architecture Segment_Arch of Segment is
begin
  process(Seg_IN)
  begin
    case Seg_IN is
      when "0000" => Seg_OUT <= "1111110";
      when "0001" => Seg_OUT <= "1100000";
      when "0010" => Seg_OUT <= "1011011";
      when "0011" => Seg_OUT <= "1110011";
      when "0100" => Seg_OUT <= "1100101";
      when "0101" => Seg_OUT <= "0110111";
      when "0110" => Seg_OUT <= "0111111";
      when "0111" => Seg_OUT <= "1100110";
      when "1000" => Seg_OUT <= "1111111";
      when "1001" => Seg_OUT <= "1110111";
      when others => Seg_OUT <= "0000000";
    end case;
  end process;
end Segment_Arch;
```



-5 : Structural Level Test Bench VHDL

CNT_SEG VHDL

```
library ieee; use ieee.std_logic_1164.all;
entity CNT_SEG is
  port ( RESET : in std_logic;
        Clk : in std_logic;
        Seg_out : out std_logic_vector(6 downto 0));
end CNT_SEG;
architecture CNT_SEG_Arch of CNT_SEG is
  component Counter
    port ( RESET : in std_logic;
          Clk : in std_logic;
          CNT_out : out std_logic_vector(3 downto 0));
  end component;
  component Segment
    port ( Seg_IN : in std_logic_vector(3 downto 0);
          Seg_OUT : out std_logic_vector(6 downto 0));
  end component;
  signal CNT_out : std_logic_vector(3 downto 0);
  for CNT : Counter
    use entity work.Counter(Counter_Arch);
  for Seg : Segment
    use entity work.Segment(Segment_Arch);
  begin
    CNT : Counter port map
      (RESET=> RESET, Clk=> Clk, CNT_out=> CNT_out);
    Seg : Segment port map
      (Seg_IN=> CNT_out, Seg_OUT=> Seg_OUT);
  end CNT_SEG_Arch;
  configuration CNT_SEG_C of CNT_SEG is
    for CNT_SEG_Arch
      end for;
  end CNT_SEG_C;
```

Test-Bench VHDL

```
library IEEE; use IEEE.Std_logic_1164.all;
entity TB_CNT_SEG is
end TB_CNT_SEG;
--
architecture TB_CNT_SEG_Arch of TB_CNT_SEG is
  component CNT_SEG
    port ( RESET : in std_logic;
          Clk : in std_logic;
          Seg_out : out std_logic_vector(6 downto 0));
  end component;
  signal RESET : std_logic := '1';
  signal Clk : std_logic := '1';
  signal Seg_out : std_logic_vector(6 downto 0);
begin
  RESET <= '0' after 5 ns;
  Clk <= not Clk after 10 ns;
  Counter : CNT_SEG port map (
    RESET => RESET,
    Clk => Clk,
    Seg_out => Seg_out);
end TB_CNT_SEG_Arch;
--
configuration TB_CNT_SEG_C of TB_CNT_SEG is
  for TB_CNT_SEG_Arch
  end for;
end TB_CNT_SEG_C;
```



-5 : VSS Simulator

```

VHDL Source Code      Analysis
$ vhdlan Counter.vhd
$ vhdlan Segment.vhd
$ vhdlan CNT_SEG.vhd
$ vhdlan TB_CNT_SEG.vhd
File_Name.scr          , Script File      . ( : Comp.scr)
Debugging Tool Loading Simulation Library Design Model
$ vhdldb &
Design Model SIM_WORK.TB_CNT_SEG_C
Waveform
Waveform
trace *
trace /TB_CNT_SEG/Counter/CNT_out
run 300
Script File
File_Name.scr
Command include Comp.scr
Waveform

```

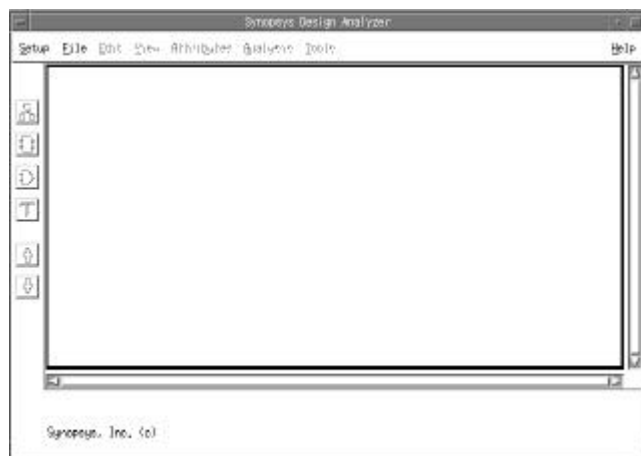


-5 : Design Analyzer Window

```

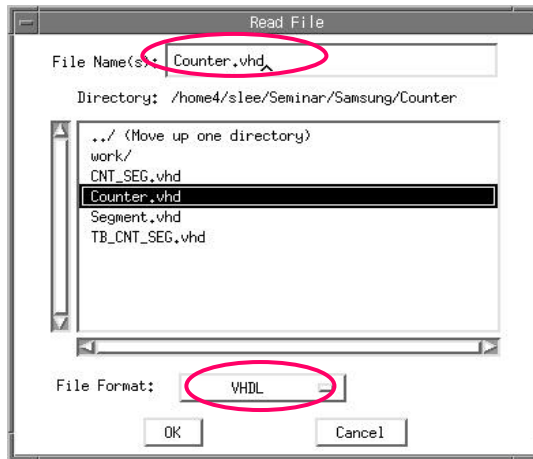
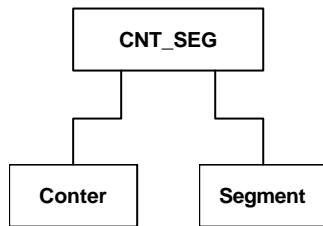
Design Analysis      Tool Loading
$ design_analyzer &
Window가 Call

```



-5 : VHDL File Read

- ⌘ VHDL Source File Read
- ⌘ Window **File -> Read** Window가 Call
- ⌘ Window Bottom-Up , VHDL Code Read
- ⌘ Counter.vhd
- ⌘ Segment.vhd
- ⌘ CNT_SEG.vhd



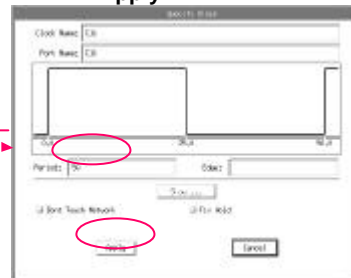
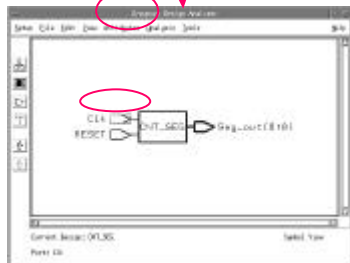
-5 : Clock

- ⌘ Clock

1. CNT_SEG Double

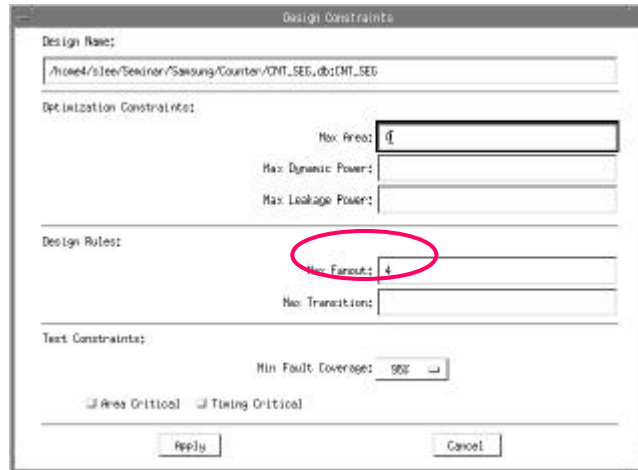
2. Clk

3. Attribute -> clock -> Specify Apply



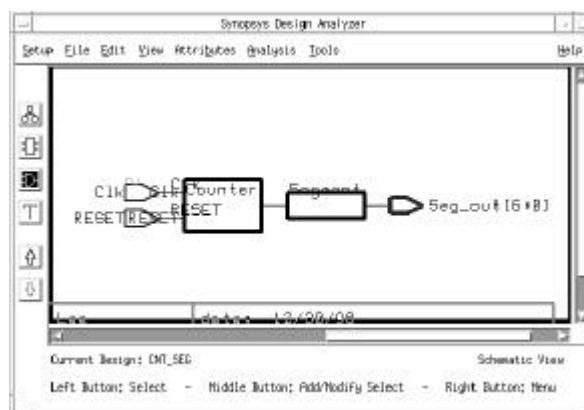
-5 : Design Constraint

- ✍ Optimization Design Rule
- ✍ symbol
- ✍ Attributes -> Optimization Constraints -> Design Constraints



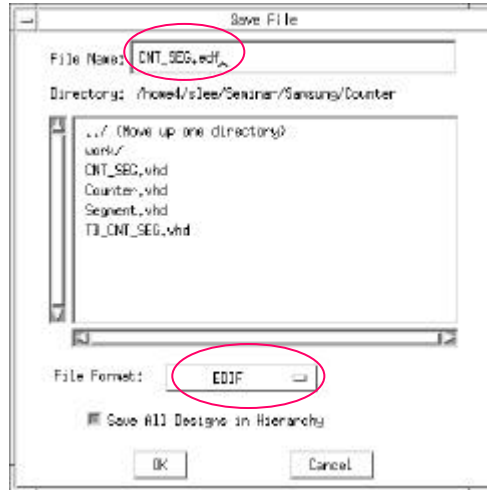
-5 :

- ✍
- ✍ Block CNT_SEG
- ✍ Level Component
- ✍ Unit Double Click



-5 : Physical Design EDIF File

- ✍ EDIF File
- ✍ File -> Save as File Format **EDIF**
- ✍ File Name **CNT_SEG.edf**

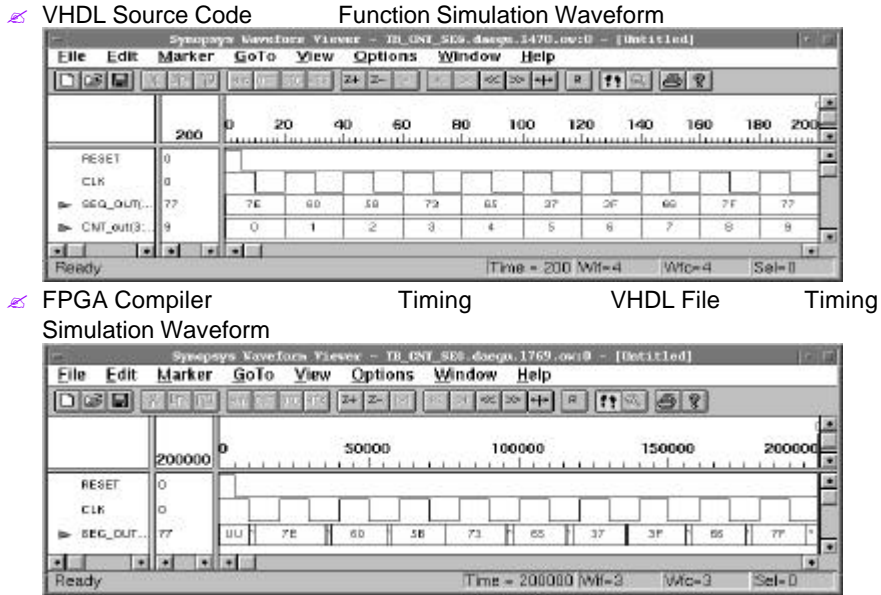


-5 : FPGA Compiler Physical Design

- ✍ MAX+PLUSII
- ✍ \$ maxplus2&
- ✍ File -> Project -> Name
CNT_SEG.edf
- ✍ MAX+PLUSII -> Compile
- ✍ Interface -> Edif Netlist Reader
Settings Synopsys
- ✍ Interface -> VHDL Netlist Writer
VHDL File **Project**
Name.vho Timing
- ✍ Assign -> Device
 - ✍ Family FLEX8000
 - ✍ Devices EPF8282LC84-3
- ✍ Compiler Window Start

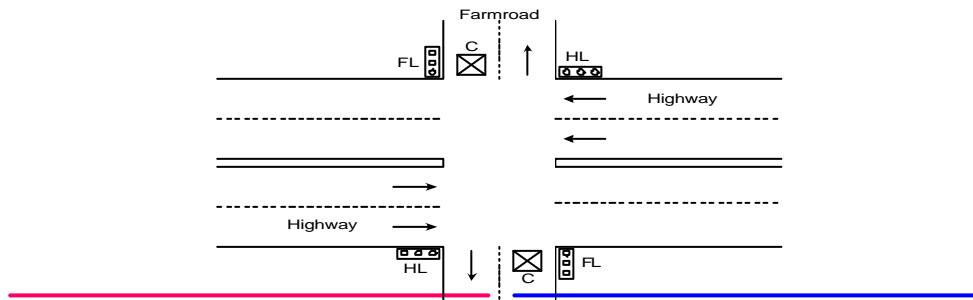


-5 : Timing Simulation



-6 : Traffic Light Controller

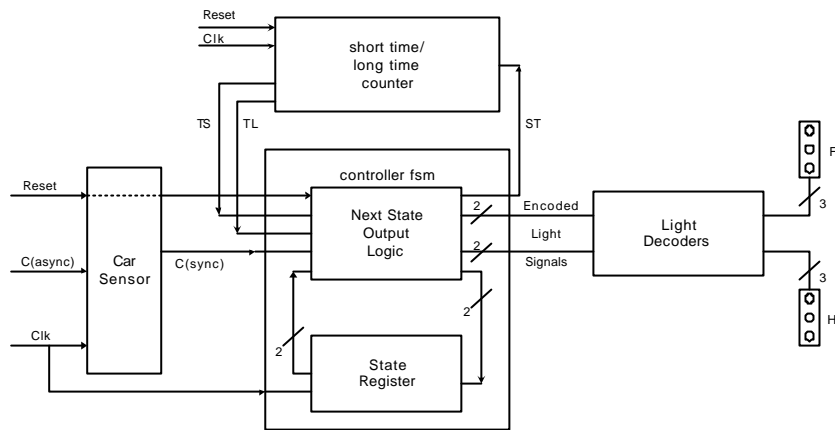
- ✎ Word Description
- ✎ Highway Farm road가
 - ✎ Farm Road 가 Car
 - ✎ Highway Green TL Car
 - ✎ Highway Yellow Mode , Highway Yellow Mode
 - ✎ Highway Yellow Mode TS , Highway Red Mode
 - ✎ Farm Road Green Mode TL Car Farm Road Yellow Mode
 - ✎ Farm Road Yellow Mode TS Highway Green Mode
 - ✎ TL (16 CLK)*(CLK 30) , TS (1 CLK)*(CLK 30)



-6 : Traffic Light Controller Block

Block Diagram

Module : Modular Design
I/O Pin

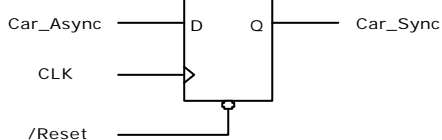


-6 : Traffic Light Controller

Car Sensor

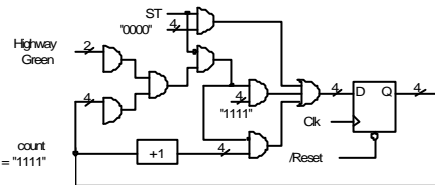
Car-Async
CLK

/Reset	Car_Async	CLK	Car_Sync
0	x	x	0
1	0		0
1	1		1



Short Time 4- Long Time Counter

/Reset	ST	count	count
0	x	x	"0000"
1	1	x	"0000"
1	0	Highway Green	"1111"
			count + '1'



count	TL	TS
"1111"	1	0
"0000"	0	1
0	0	0

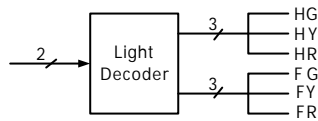


-6 : Traffic Light Controller

Light Decoders

- 47†
- High-Way Farm-Road
47†
Decoder

H0(F0)	H1(F1)	HG(FG)	HY(FY)	HR(FR)	Light
0	0	1	0	0	Green
0	1	0	1	0	Yellow
1	0	0	0	1	Red
1	1	Not Allowed			

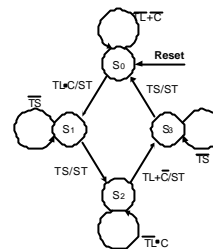


Traffic Light State Controller

- 47†

State Table State Diagram

Reset	TS	TL	C	Current State	Next State	ST (Next state)
0	x	x	x	S0	S0	0
1	x	0	x	S0	S0	0
1	x	x	0	S0	S0	0
1	x	1	1	S0	S1	1
1	0	x	x	S1	S1	0
1	1	x	x	S1	S2	1
1	x	0	1	S2	S2	0
1	x	1	x	S2	S3	1
1	x	x	0	S2	S3	1
1	0	x	x	S3	S3	0
1	1	x	x	S3	S0	1



-6 : Traffic Light Controller VHDL

VHDL

```

library IEEE; use IEEE.std_logic_1164.ALL;
use IEEE.std_logic_unsigned."+";
entity TRAFFIC is
port (RESET, CLK, C_ASYNC : in std_logic;
      HG, HY, HR, FG, FY, FR : out std_logic);
end TRAFFIC;
architecture RTL of TRAFFIC is
type STATE is (S0, S1, S2, S3);
type LIGHT is (GREEN, YELLOW, RED);
signal C_STATE, N_STATE : STATE;
signal H, F : LIGHT;
signal C_SYNC, ST, TS, TL : std_logic;
signal COUNT:std_logic_vector(3 downto 0);
begin
A: block begin
process (RESET, CLK) begin
if RESET = '0' then
C <= '0';
elsif CLK = '1' and CLK'event then
C <= C_ASYNC;
end if;
end process;

```

```

--Timer counter
-- HG count
process (RESET, CLK) begin
if RESET='0' then
COUNT <= "0000";
elsif CLK = '1' and CLK'event then
if ST = '1' then
COUNT <= "0000";
elsif C_STATE=S0 and COUNT="1111" then
COUNT <= COUNT; --NULL
else
COUNT <= COUNT + '1';
end if;
end if;
end process;
--Long time, short time
TS<='1' when COUNT="0000" else '0';
TL<='1' when COUNT="1111" else '0';
--High-way ON/OFF
HG<='1' when H=GREEN else '0';
HY<='1' when H=YELLOW else '0';
HR<='1' when H=RED else '0';

```



-6 : Traffic Light Controller VHDL

```

--Farm road      ON/OFF
FG<='1' when F=GREEN else '0';
FY<='1' when F=YELLOW else '0';
FR<='1' when F=RED else '0';
end block;
--
B: block begin
process (RESET, CLK) begin
if RESET = '0' then
C_STATE <= S0;
elsif CLK = '1' and CLK'event then
C_STATE <= N_STATE;
end if;
end process;
--
timer
process (C_STATE, TS, TL, C_SYNC) begin
case C_STATE is
when S0 => H <= GREEN; F <= RED;
if (TL = '1') and (C_SYNC = '1') then
N_STATE <= S1; ST <= '1';
else
N_STATE <= S0; ST <= '0';
end if;
when S1 => H <= YELLOW; F <= RED;
if TS = '1' then
N_STATE <= S2; ST <= '1';
else
N_STATE <= S1; ST <= '0';
end if;
when S2 => H <= RED; F <= GREEN;
if (TL = '1') or (C_SYNC = '0') then
N_STATE <= S3; ST <= '1';
else
N_STATE <= S2; ST <= '0';
end if;
when S3 => H <= RED; F <= YELLOW;
if TS = '1' then
N_STATE <= S0; ST <= '1';
else
N_STATE <= S3; ST <= '0';
end if;
end case;
end process;
end block;
end RTL;

```



-6 : Traffic Light Controller Test Bench

```

Simulation      Test Bench
Stimulus Vector
Traffic Light Controller
library IEEE; use IEEE.std_logic_1164.ALL;
--Empty entity
entity TB_TRAFFIC is end TB_TRAFFIC;
architecture RTL of TB_TRAFFIC is
--Test component
component TRAFFIC
port (RESET,CLK,C_AYNC : in std_logic;
      HG,HY,HR,FG,FY,FR : out std_logic);
end component;
--
signal RESET : std_logic;
signal CLK : std_logic := '0';
signal C_AYNC : std_logic;
signal HG,HY,HR,FG,FY,FR : std_logic;
--component configuration
for U0 : TRAFFIC use entity work.TRAFFIC(RTL);
begin
--
vector
CLK <= not CLK after 15 sec;
vector
RESET <= '0', '1' after 1 sec;
--Random vector
process begin
C_AYNC <= '0';
wait for 10 min;
C_AYNC <= '1';
wait for 5 min;
C_AYNC <= '0';
wait for 5 min;
C_AYNC <= '1';
wait for 5 min;
end process;
--
Pin Wire
U0 : TRAFFIC
port map (RESET, CLK, C_AYNC, HG, HY,
          HR, FG, FY, FR);
end RTL;

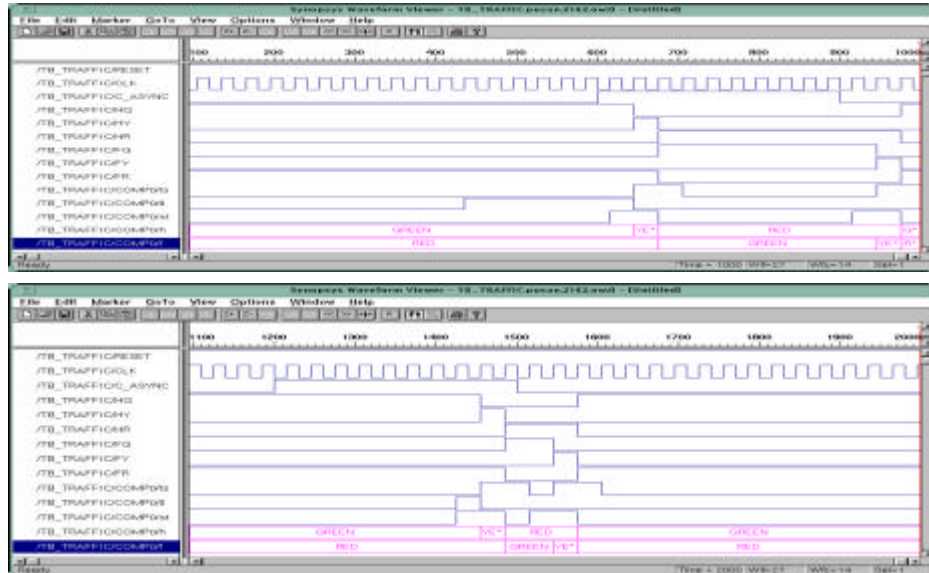
```



-6 : Traffic Light Controller VHDL

Test Bench

Synopsys Simulation



-6 : Traffic Light Controller

Script File

```
read -f vhd traffic.vhd
```

```
group -hdl_block A
```

```
group -hdl_block B
```

```
create_clock clk
```

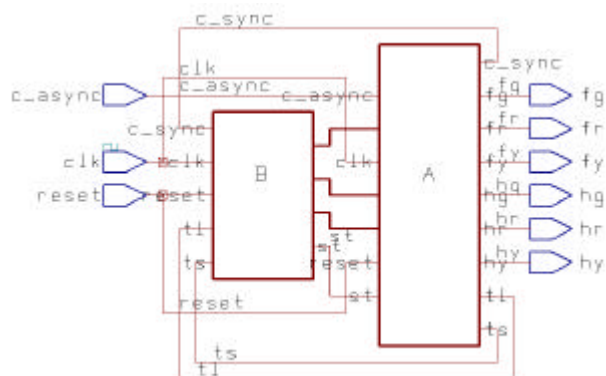
```
max_delay 0 -to all_outputs()
```

```
compile -map_effort high
```

```
create_schematic
```

```
write -f db
```

```
write -f edif -hierarchy -o traffic.edif
```



2 Sub-Block

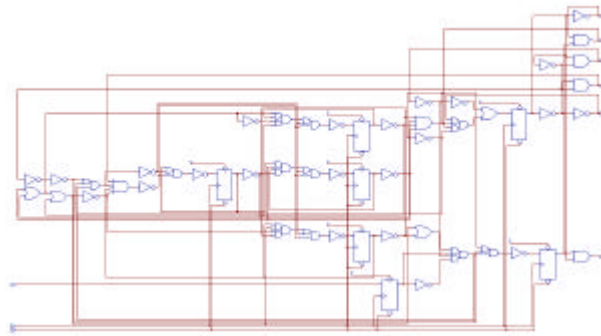
Block Label

VHDL Block Label



-6 : Traffic Light Controller 1-Level

```
1-Level  
Script File  
read -f vhd traffic.vhd  
create_clockclk  
max_delay 0 -to all_outputs()  
compile -map_effort high  
create_schematic  
write -f db  
write -f edif -hierarchy -o traffic.edf
```

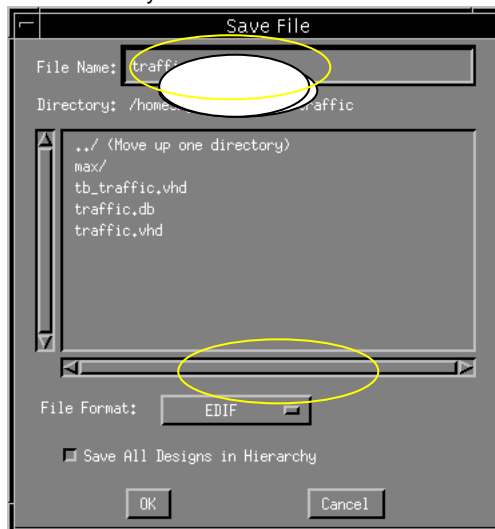


7 Flip-Flop Random Logic



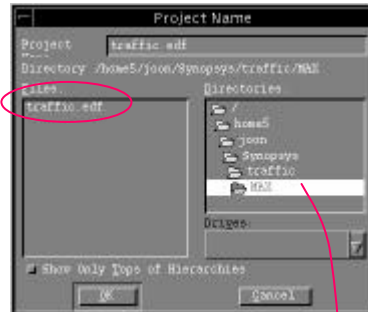
-6 : Traffic Light Controller VHDL FPGA

Synthesis (NO-33)
ALTERA Library FPGA edif

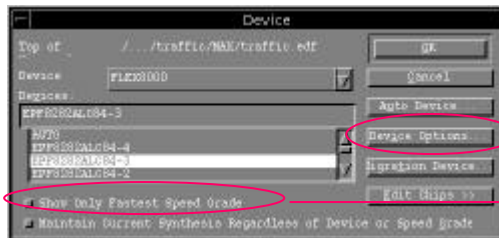


-6 : Traffic Light Controller VHDL FPGA

- ✂ directory MAX Sub-directory
- ✂ traffic.edf MAX directory
- ✂ MAX directory Max+plus
- ✂ : max2win&
- ✂ Project
 - ✂ File -> Project -> Name traffic.edf
- ✂ Device
 - ✂ Device FLEX 8000
 - ✂ EPF8282LC84 -3
 - ✂ Assign -> Device EPF8282LC84-3

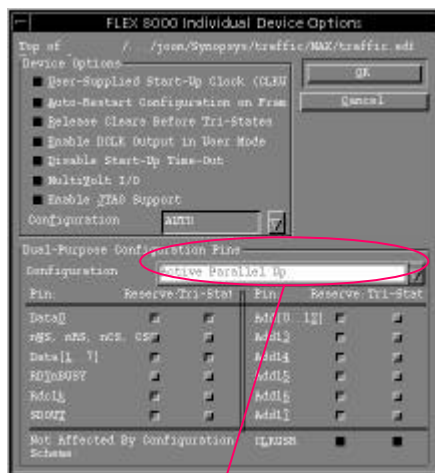


MAX
directory

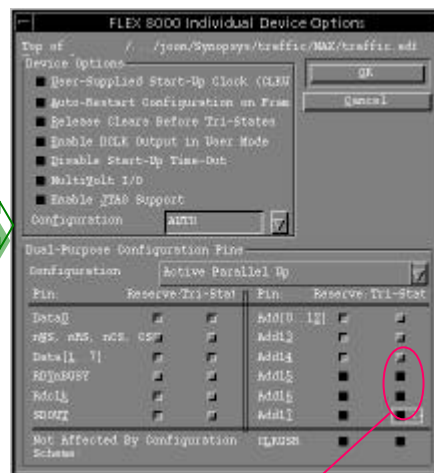


-6 : Traffic Light Controller VHDL FPGA

- ✂ Device Options



✂ Configuration Active Parallel Up



✂ Add15, 16, 17



-6 : Traffic Light Controller VHDL FPGA

Compiler

Max+plus II -> Compiler

Processing -> Preserve All Node Name Synonyms

(1 bit Node Name)

Interfaces -> Edif Netlist Reader Settings



Vendor Synopsys

Interfaces -> VHDL Netlist Writer Settings



Interfaces -> VHDL Netlist Writer

(Gate Delay Timing Simulation .vho)



-6 : Traffic Light Controller VHDL FPGA

Start

Compiler



error가

error가

rpt

compile

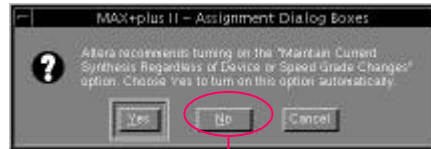
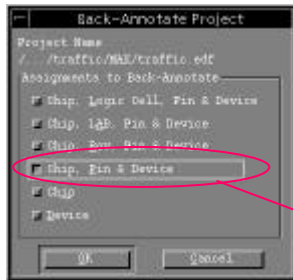
가



-6 : Traffic Light Controller VHDL FPGA

Pin Re-Fitting

- ✍ rpt , compile 가
- ✍ FPGA
- ✍ 가
- ✍ Max+plus II -> Compiler , Assign -> Back-Annotate Project



check OK NO ()

- ✍ Hierarchy Display -> acf
- ✍ FPGA Chip 가
- ✍ , acf In, Out port

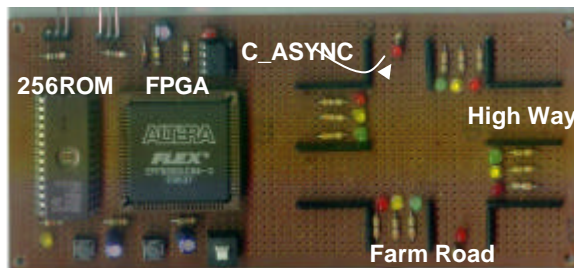


-6 : Traffic Light Controller VHDL FPGA

- ✍ compiler
 - ✍ Max+plus II
 - ✍ MAX directory traffic.hex
 - ✍ 27C256 ROM Download
 - ✍ FPGA
- ```

| C_ASYNC : INPUT_PIN = 73
| CLK : INPUT_PIN = 12;
| RESET : INPUT_PIN = 31;
| FG : OUTPUT_PIN = 30;
| FR : OUTPUT_PIN = 56;
| FY : OUTPUT_PIN = 25;
| HG : OUTPUT_PIN = 24;
| HR : OUTPUT_PIN = 28;
| HY : OUTPUT_PIN = 16;

```



# -6 : FPGA Prototyping

## Traffic Light Controller

## FPGA

## Board

```

75 77 79 81 83 01 03 05 07 09 11
74 76 78 80 82 84 02 04 06 08 10 13 12
72 73 15 14
70 71 17 16
68 69 19 18
66 67 21 20
64 65 23 22
62 63 25 24
60 61 27 26
58 59 29 28
56 57 31 30
54 55 52 50 48 46 44 42 40 38 36 34 32
53 51 49 47 45 43 41 39 37 35 33

```

EPF8282ALC84-2

