메모(Technical)

Document No.: MJL-LD-AN-7
Author: 하태욱 [twha@mjl.com]
Version: 1.0
Date: 2000 년 8 월 8 일

Subject: **LPM_FUNCTION 을 사용한 디자인을 ModelSim 에서 Simulation 하기**

---

1. lpm_function 을 사용하여 256x8 DPRAM 만들기

MAX+plus II 또는 Quartus 에서 사용자는 본인이 필요로 하는 Component 를 MegaWizard Plug-in Manager 를 사용하여 만들 수 있다. 여기서는 간단한 Dual-Port RAM 을 만들어서 이것을 ModelSim 에서 Functional Simulation 및 Timing Simulation 을 하는 방법에 대해 기술하고자 한다.

MAX+plus II 에서 File → Megawizard Plug-in Manger 를 기동한다.

Figure 1 MegaWizard Plug-In Manager Window

새로운 Component 를 만들 것이므로 그림 1 에서 "**Create a new custom megafunction variation**"을 선택하고 Next Button 을 클릭한다.

아래 그림 2 와 같이 **lpm_ram_dp** 를 선택하고 출력 파일 형태를 VHDL 로 정의하고 저장될 디렉토리를 선택한 후 파일명을 넣는다. 여기서는 C:\mywork\modelsim\dp_ram 디렉토리에 test 라는 파일명을 넣는 것으로 한다. 사용자는 자기가 원하는 디렉토리를 정의하면 된다.

Next Button 을 눌러 다음 화면으로 진행하여 필요한 선택사항을 On 시킨다.

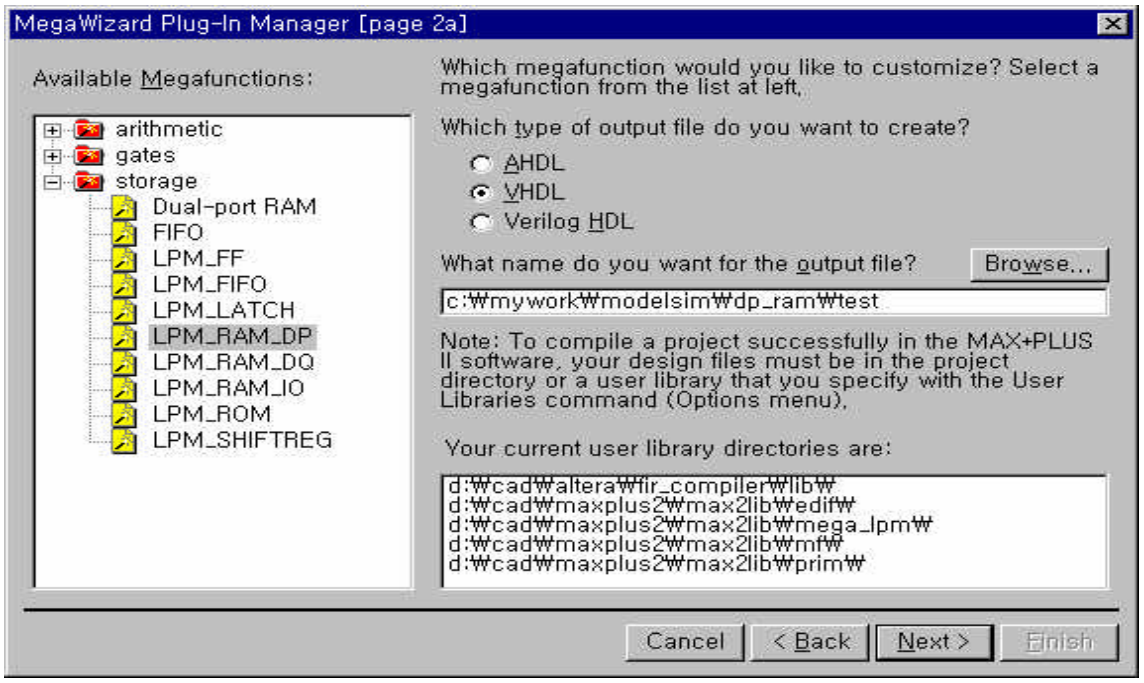


Figure 2 Select Megafunction Window

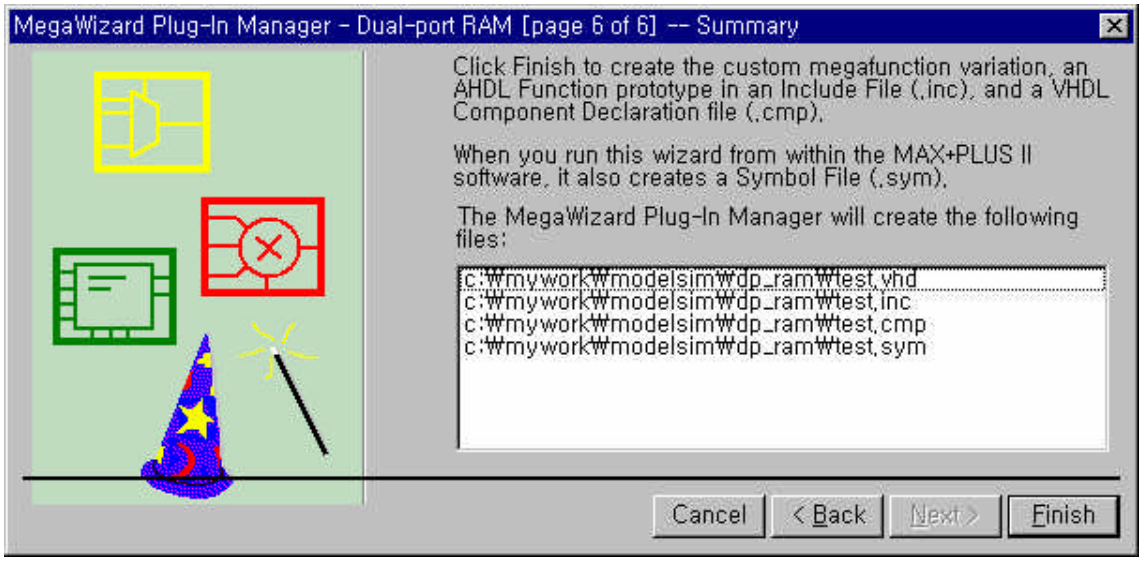


Figure 3 MegaWizard Summary Window

여기서는 모든 단계의 선택사항은 Default Option **을 선택하는 것으로 한다.**

Finish Button **을 클릭하여 결과를 확인한다.**

C: \ MYWORK \ MODELSIM \ DP_RAM **디렉토리에 그림 3 에서 보이는 파일들이 저장되어 있는지 확인한다.**

2. ModelSim 에서 256x8 DPRAM Function Simulation 하기

ModelSim 을 기동 시킨다. Top Menu 에서 File → Change Directory... 를 선택하여 test.vhd 파일이 있는 디렉토리로 이동한다. 여기서는 c: \ mywork \ modelsim \ dp_ram directory 를 선택한다.

Top Menu 에서 Design → Create a New Library... 를 선택하면 그림 4 와 같은 창이 나타난다. 여기에 work 라는 작업용 라이브러리을 만든다.

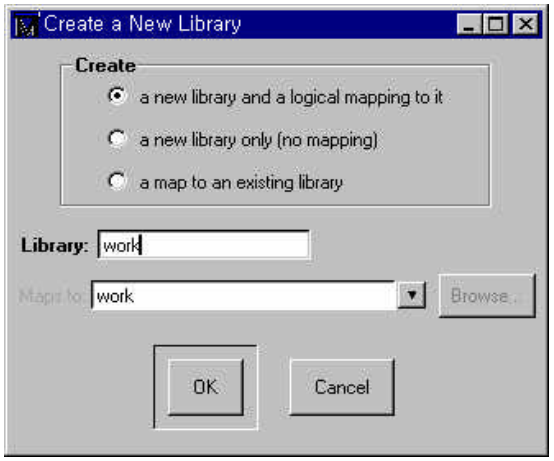


Figure 4 Create a New Library Window

여기서 한번 확인하고 넘어갈 사항이 있다. 현재 작업디렉토리에 modelsim.ini 파일이 있으면 지워버리길 바란다. 반드시 지울 필요는 없지만 어떤 라이브러리를 기존에 존재하는 라이브러리에 연결하고자 할 때 잘 안될 경우가 있는 데, 이럴 때는 반드시 작업 디렉토리에 있는 modelsim.ini 를 지워야 된다.

다음 단계는 Compile 단계이다. 그림 5 와 같이 행한다. Top Menu 에서 Design → Compile... 을 선택한다. Compile Window 에서 작업 디렉토리에 있는 test.vhd File 을 선택한다. 여기서는 c: \ mywork \ modelsim \ dp_ram \ test.vhd 를 선택한다.

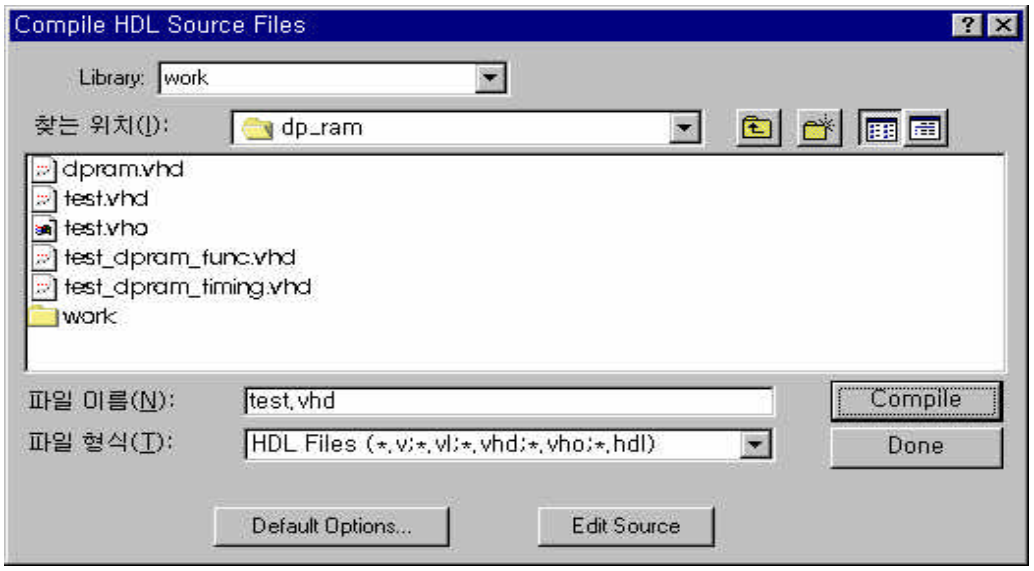


Figure 5 Compile HDL Source Files Window

Compile Button 을 클릭하여 test.vhd File 을 Compile 한다.

그 결과가 그림 6 과 같은 Error 가 있으면 **LPM LIBRARY** 를 만들어 주어야 한다.

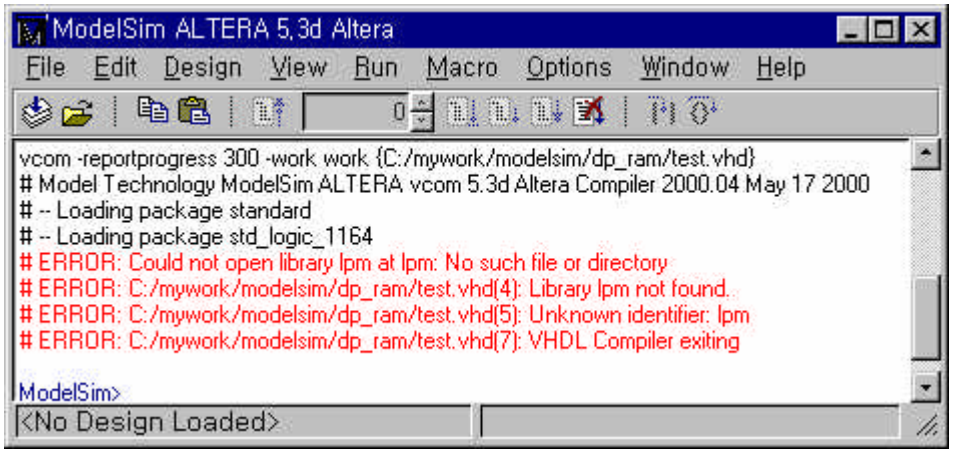
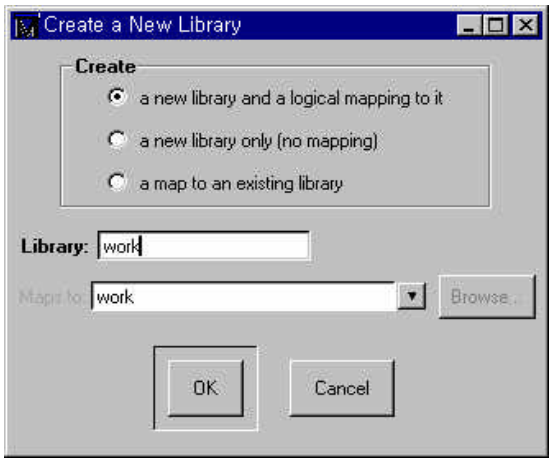


Figure 6 LPM Library 가 없을때 나오는 Error Message 창

ModelSim 용 LPM Library 를 만들어서 LPM Library 를 test.vhd 의 Compile 전에 정의를 해 주어야 만 ERROR 없이 Compile 이 수행된다. 여기서 몇 가지 개념을 잡고 넘어 가기로 한다.



- a new library and a logical mapping to it

이것은 현재 ModelSim 에서 선택한 디렉토리에 work 라는 디렉토리를 만들고 modelsim.ini 에 work library 가 현재 선택한 디렉토리밑에 만든 work directory 라는 정보를 저장한다. 여기서는 c: \ mywork \ modelsim \ dp_ram directory 에 work 라는 Sub-directory 를 만든다.

- a new library only [no mapping]

이것은 현재 ModelSim 에 선택한 디렉토리에 work 라는 디렉토리만 만든다.

- a map to an existing library

이것은 새로운 Library 를 만드는 것이 아니라 기존에 만든 라이브러리를 연결하여 사용할 때 이용한다.

LPM Library 는 한번 만들어서 여러 곳에서 사용할 수 있도록 하기 위해 여기서는 ModelSim 이 설치된 Directory 에 만들고 이것을 "**a map to an existing library**"를 이용하여 작업하는 방법을 사용하기로 한다. ModelSim Top Menu 에서 File → Change Directory… 를 선택하여 ModelSim 이 설치된 디렉토리로 이동한다. 여기에서는 d: \ cad \ modeltech_ae \ altera \ vhdl Directory 를 선택한다. ModelSim 5.3d Altera Edition Version 은 Altera 를 사용할 수 있도록 미리 필요한 Library 들을 미리 Compile 하여 Altera Directory 에 VHDL 과 Verilog Sub-directory 를 만들어 각각 저장 시켜 놓았다.

Top Menu 에서 Design → Create a New Library… 를 선택하여 그림 7 과 같이 작업한다.

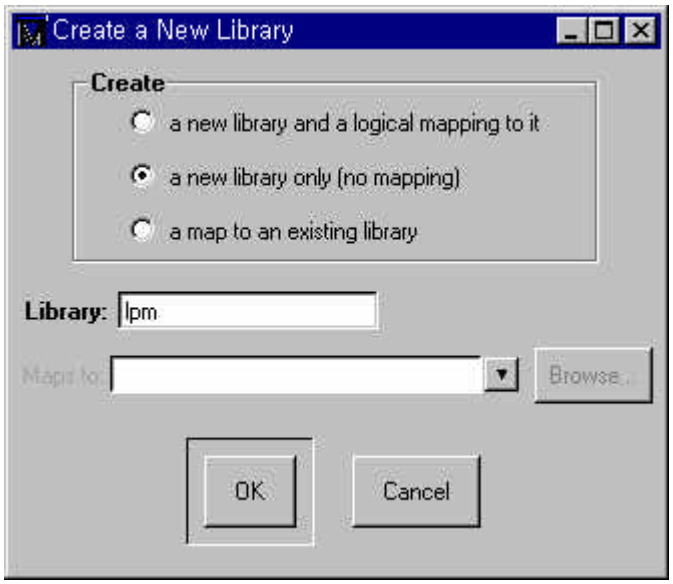


Figure 7 a new linrary only (no mapping) Window

LPM Directory 를 만들었으면 lpm_component 를 선언한 Source VHDL File 과 lpm_component 를 Modeling 한 Source VHDL File 을 ModelSim 에서 Compile 해야 한다.

MAX+plus II 는 <MAX+plus II 가 설치된 디렉토리> \ lpmsim Directory 에 220pack.vhd, 220model.vhd 두개의 파일이 lpm_component 의 Source Code 이다.

Quartus 는 <Quartus 가 설치된 디렉토리> \ eda \ sim_lib Directory 에 위에서 말한 두개의 Source Code 가 있다.

ModelSim Top Menu 에서 Design → Compile… 을 선택하여 220pack.vhd, 220model.vhd 를 Compile 한다. 반드시 220pack.vhd 를 먼저 Compile 하고 그 다음에 220model.vhd 를 Compile 한다.

그림 8 과 같이 작업을 한다. 그림 8 에서 Compile Button 을 클릭하기 전에 Source Code 가 1987 VHDL Version 으로 Coding 되어 있으므로 Default Option… Button 을 클릭하여 그림 9 와 같이 작업을 한다.

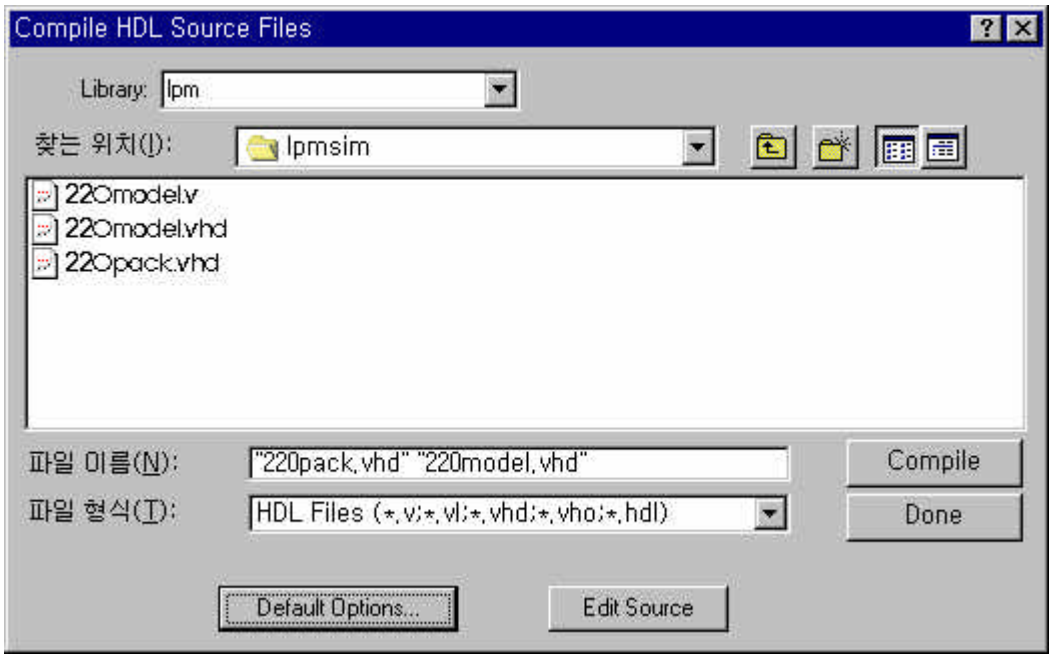


Figure 8 Compile HDL Source Files Window

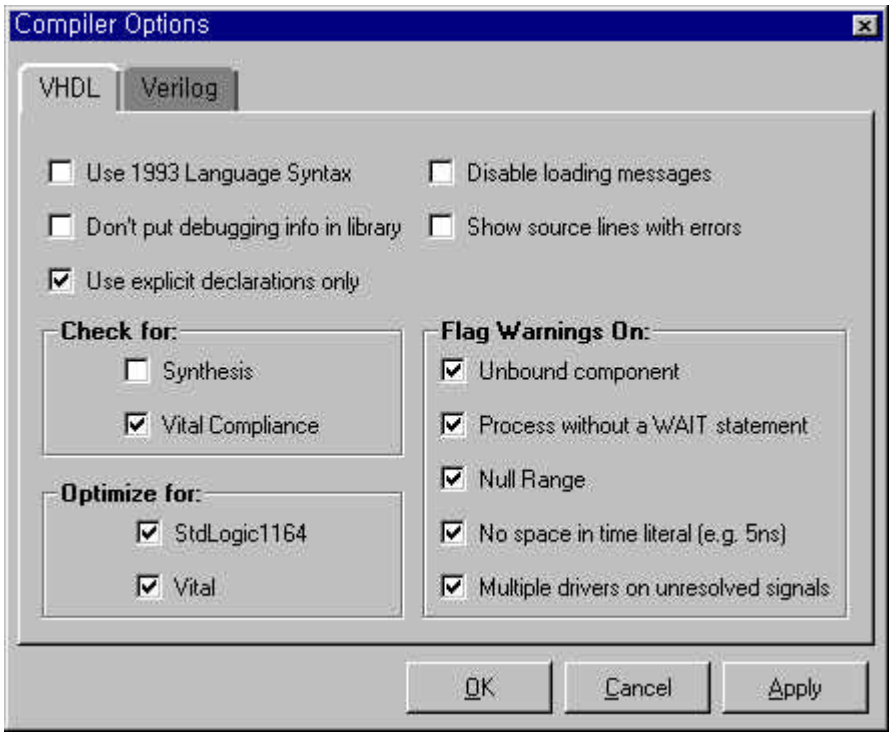


Figure 9 Compiler Options Window

그림 8 에서 Compile Button 을 눌러 Source File 이 이상 없이 Compile 되었는지 확인한다.

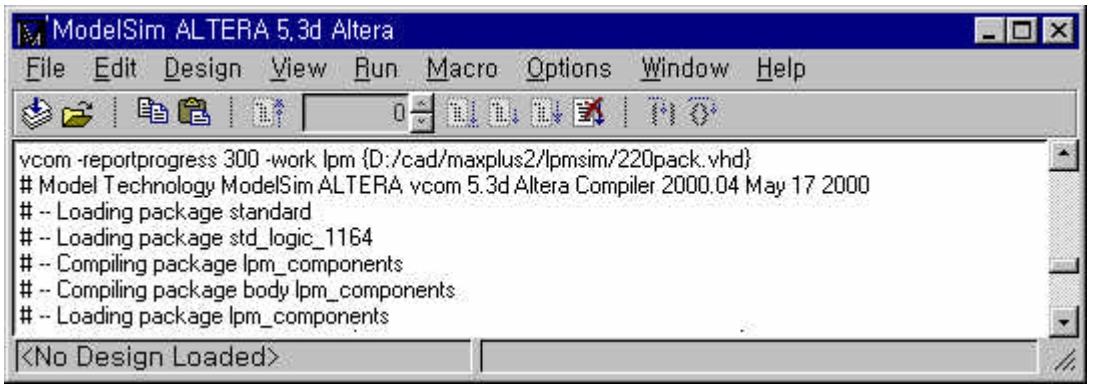


Figure 10 220PACK.VHD File Compile Result Window

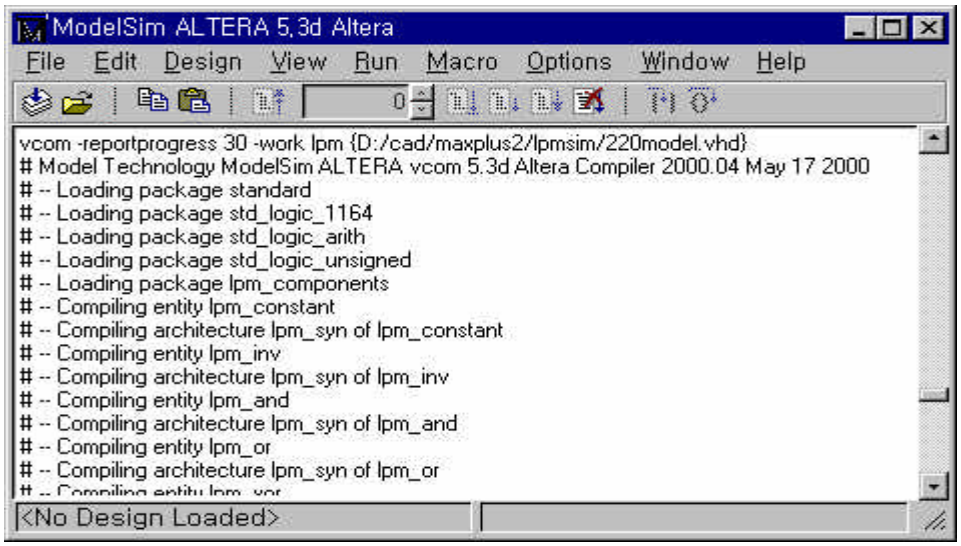


Figure 11 220MODEL.VHD File Compile Result Window

자 이제 LPM LIBRARY **도 만들었으니 본연의 일을 수행할 수가 있겠다.** ModelSim Top Menu **에서 그림** 12 **와 같이 하자. 이 방법은** File → Change Directory… **와 같은 일을 한다.**

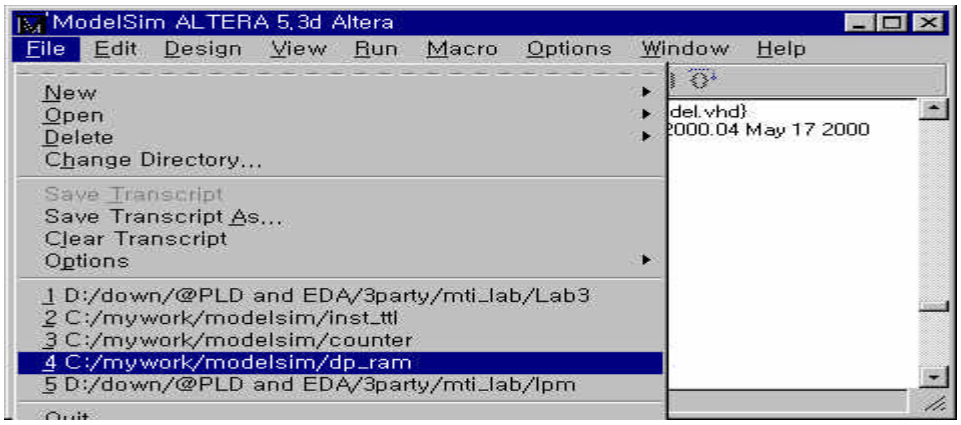


Figure 12 Change Directory... Window

ModelSim Top Down Menu 에서 Design → Browse Libraries... 를 선택한다. 그림 13 과 같이 LPM Library 가 정의 되어 있는지를 확인한다. 만일 LPM Library 가 정의가 되지 않았다면 Add Button 을 눌러 그림 14 와 같이 LPM Library 를 정의한다.

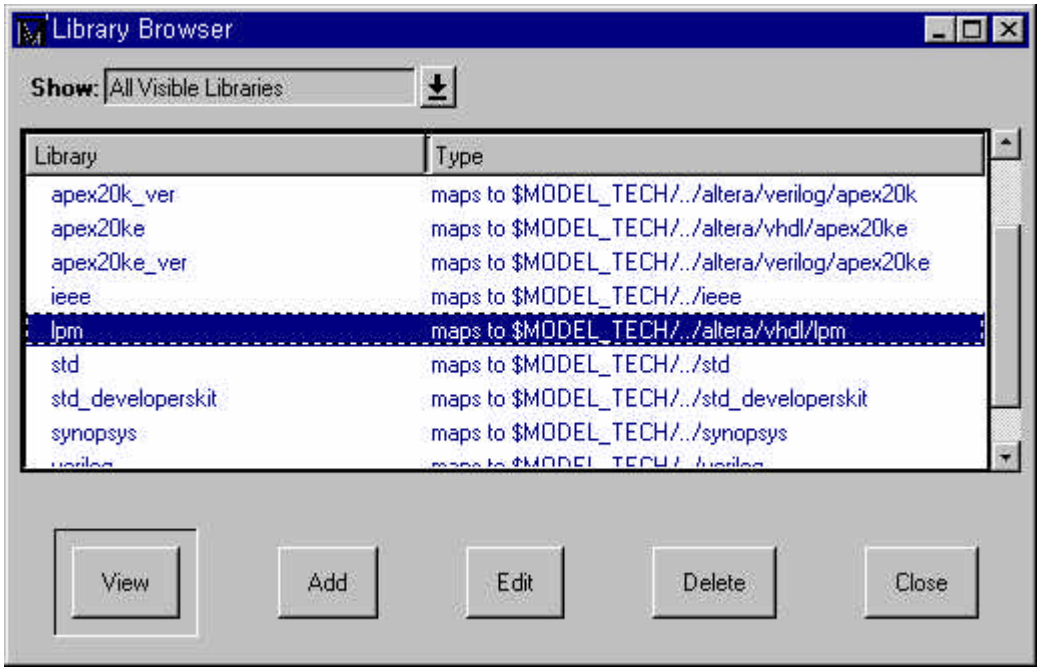


Figure 13 Library Browser Window

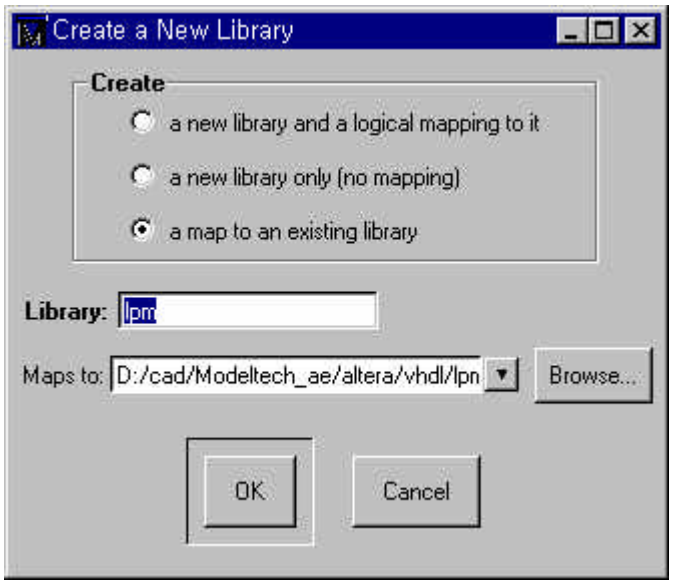


Figure 14 Create a New Library Window

a map to an existing library 를 선택한 후 Browse Button 을 눌러 전 단계에서 만든 LPM Library Directory Path 를 찾아서 넣어 준다. 이상 없이 진행이 되면 그림 13 과 같은 내용을 얻을 수 있다.

Design → Compile… Menu 를 선택하여 test.vhd 을 Compile 하자.

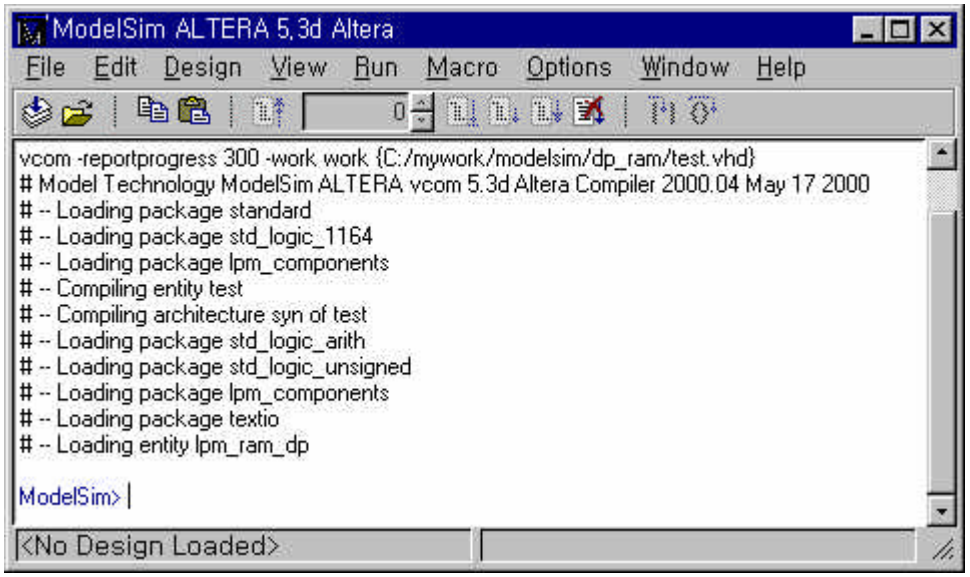


Figure 15 test.vhd 를 Compile 한 결과

그림 15 와 같은 결과를 얻어야 한다. 자 이제는 test.vhd 을 Functional Simulation 을 하기위한 Testbench 용 VHDL Code 를 작성하여야 한다. 다음 쪽에 Testbench File 의 내용을 적었다.

여기서는 이 파일의 이름을 test_dpram_func.vhd 라 한다.

ModelSim Top Down Menu 에서 Design → Compile… 을 선택하여 작성한 test_dpram_func.vhd File 을 Compile 한다. 이상없이 Compile 이 완료되면 Compile HDL Source Files Window 에 있는 Done Button 을 눌러 Compile Menu 를 빠져 나온다.

* Test_dpram_func.vhd 의 Source Code

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

--use std.TEXTIO.all;
use ieee.std_logic_TEXTIO.all;

entity test_dpram is
--   port (
--        );
end test_dpram;

architecture arch of test_dpram is

constant CYC_TIME : TIME := 50 ns;

component test
  port ( data : IN STD_LOGIC_VECTOR (7 DOWN 0);
         wraddress : IN STD_LOGIC_VECTOR (7 DOWN 0);
         rdaddress : IN STD_LOGIC_VECTOR (7 DOWN 0);
         wren : IN STD_LOGIC := '1';
         clock : IN STD_LOGIC ;
         q : OUT STD_LOGIC_VECTOR (7 DOWN 0));
end component;

signal clock, wren : std_logic;
signal datain : std_logic_vector(7 downto 0);
signal waddr, raddr, q : std_logic_vector(7 downto 0);

signal main_clk, reset : std_logic;
signal rd_clk : std_logic;
signal wr_clk : std_logic;
signal cnt_data : std_logic_vector(7 downto 0);

begin
U1: test port map ( data => datain,
                    wraddress => waddr,
                    rdaddress => raddr,
                    wren => wren,
                    clock => clock,
                    q => q);

wren <= '1';
clock <= main_clk;

main_clk_gen: process
  begin
    main_clk <= '0';
    wr_clk <= '0';
    wait for CYC_TIME/2;
    main_clk <= '1';
    wr_clk <= '1';
    wait for CYC_TIME/2;
end process;

rd_clk_gen: process
  begin
    rd_clk <= '0';
    wait for CYC_TIME;
    rd_clk <= '1';
    wait for CYC_TIME;
end process;

Reset_Gen: process
  begin
    if NOW = 0 ns then
      reset <= '0', '1' after 100 ns;
      wait for CYC_TIME * 2;
    else
      wait;
    end if;
end process;

CNTDATA_GEN: process(reset, main_clk)
  begin
    if reset = '0' then
      cnt_data <= (others=>'0');
    elsif main_clk'event and main_clk='1' then
      cnt_data <= cnt_data + '1';
    end if;
end process;

-- Data input from the counter value
Din_Gen: process(wr_clk,reset)
  begin
    if reset = '0' then
      datain <= (others=>'0');
    elsif wr_clk'event and wr_clk='0' then
      datain <= cnt_data;
    end if;
end process;

-- Write Address Generation
WADDR_GEN: process(reset,wr_clk)
  begin
    if reset = '0' then
      waddr <= (others=>'0');
    elsif wr_clk'event and wr_clk='0' then
      waddr <= waddr + '1';
    end if;
end process;

-- Read Address Generation
RADDR_GEN: process(reset,rd_clk)
  begin
    if reset = '0' then
      raddr <= (others=>'0');
    elsif rd_clk'event and rd_clk='0' then
      raddr <= raddr + '1';
    end if;
end process;

end arch;

configuration CONF_TEST_DP_RAM of TEST_DPRAM is
  for ARCH
    for U1:TEST use entity work.TEST(SYN);
    end for;
  end for;
end CONF_TEST_DP_RAM;
```

ModelSim 에서 Simulation 할 Design Loading 하기

ModelSim Top Menu 에서 Design → Load New Design… 을 선택하여 그림 16 과 같이 작업을 한다.

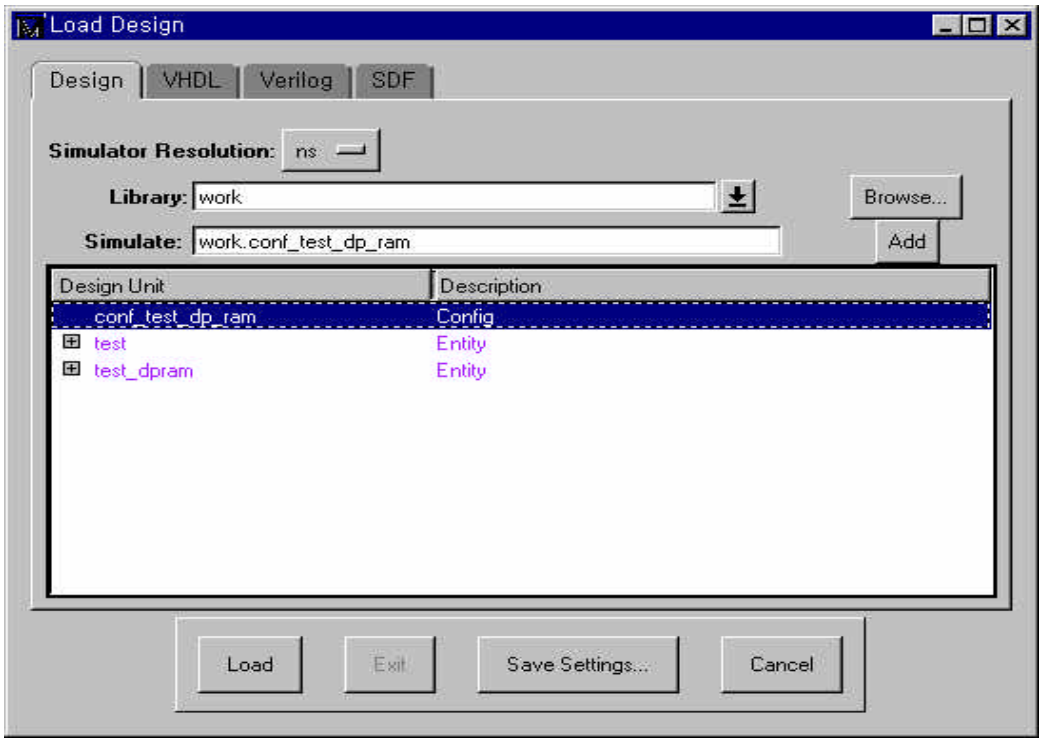


Figure 16 Load New Design Window

Load Button 을 눌러서 이상 없이 디자인이 Loading 되는 지를 그림 17 을 통해 확인하길 바란다.

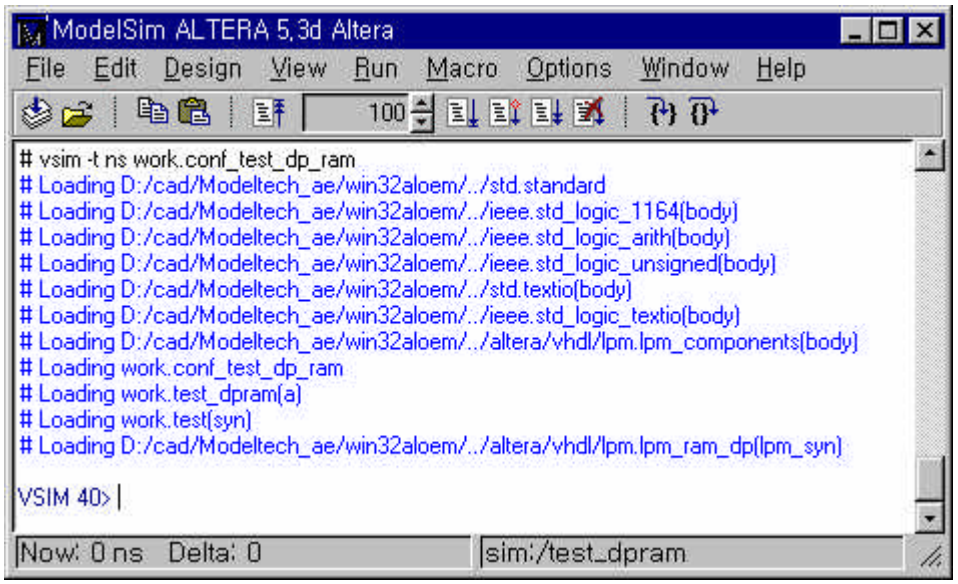


Figure 17 Design 이 Loading 된 결과 Window

그림 18 과 같이 Transcripts Window 에서 View Signals <enter>, View Wave <enter> 두개의 명령을 실행하여 화면에 Waveform Window 와 Signals Window 를 불러 들인다.

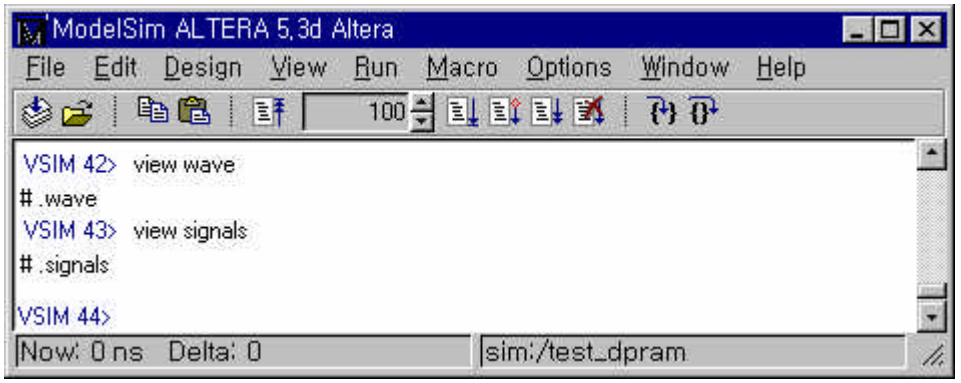


Figure 18 wave 와 signal window 를 활성화하는 화면

Signal Window 에서 그림 19 와 같이 View → Wave → Signals in Region 을 선택하여 모든 신호들을 Waveform Window 에서 볼 수 있도록 한다.

Figure 19 Signal Window

그림 20 에서와 같이 하여 Loading 된 디자인을 1000ns 까지 Simulation 을 수행한다.

Figure 20 1000ns Simulation Window

Wave Window 를 적당한 크기로 확대하여 원하는 결과가 나오는 지를 확인하다. 여기서는 초기치 값을 주지 않았다.

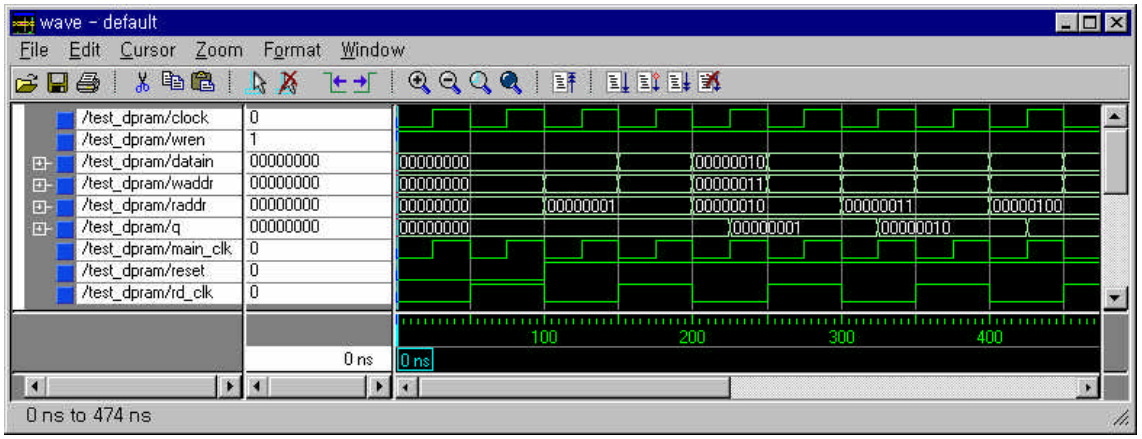


Figure 21 Functional Simulation Result Window

3. Timing Simulation 수행하기

Timing Simulation 용 Testbench File 만들기.

Function Simulation 에서 사용했던 Testbench File 을 Configuration 문을 수정하여 사용을 할 수가 있다. 다음과 같이 수정한다. U1 Instance 가 사용할 Architecture 는 \ EPF10K30ETC144-3 \ 이다.

```
configuration CONF_TEST_DP_RAM of TEST_DPRAM is
   for ARCH
      for U1:TEST use entity work.TEST(\EPF10K30ETC144-3\);
      end for;
   end for;
end CONF_TEST_DP_RAM;
```

여기서는 File Name 을 test_dpram_timing.vhd 로 정의한다.

MAX+plus II 에서 Gate-level VHDL Output 과 Delay Output 확인하기.

MAX+plus II 에서 test.vhd 를 Project 로 정의한 후 Assign → Device… 명령을 선택한 후 EPF10K30ETC144-3 을 선택한다.

Compile Process 를 활성화 시킨 후 Interfaces → VHDL Netlist Writer 를 On 시킨 후 Interfacs → VHDL Netlist Writer Settings… 를 선택하여 그림 22 와 같이 작업한다.

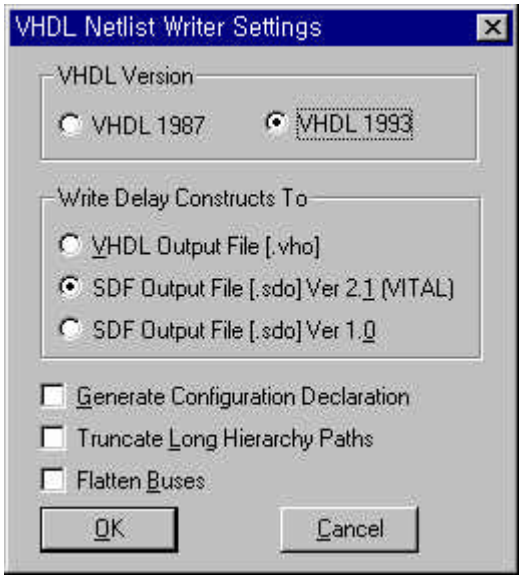


Figure 22 VHDL Netlist Writer Setting Window

Compile 을 실행한다. 작업 디렉토리에 test.vho 와 test.sdo 두개의 출력 파일이 있을 것이다. test.vho 는 test.vhd 파일을 Synthesis (논리합성) 한 후 그 정보를 가지고 EPF10K30E Device 에서 Place & Route 를 수행하고 그 결과 를 Gate-level 로 표현하여 VHDL File 로 만든 것을 의미하며 test.sdo 는 Standard Delay Format 에 맞게 Net 및 Logic Delay 를 표현한 것을 말한다. 이 두개의 File 이 있어야만 반드시 Timing Simulation 을 수행할 수가 있다.

ModelSim 에서 Design → Compile … 을 선택하여 test.vho 를 Compile 한다. 이상 없이 진행이 되면 work library 에 test.vho 에 대한 정보가 저장된다. 마찬가지로 test_dpram_timing.vhd 도 Compile 한다.

Design → Load New Design… 을 선택하여 그림 23, 그림 24 와 같이 작업을 한다.

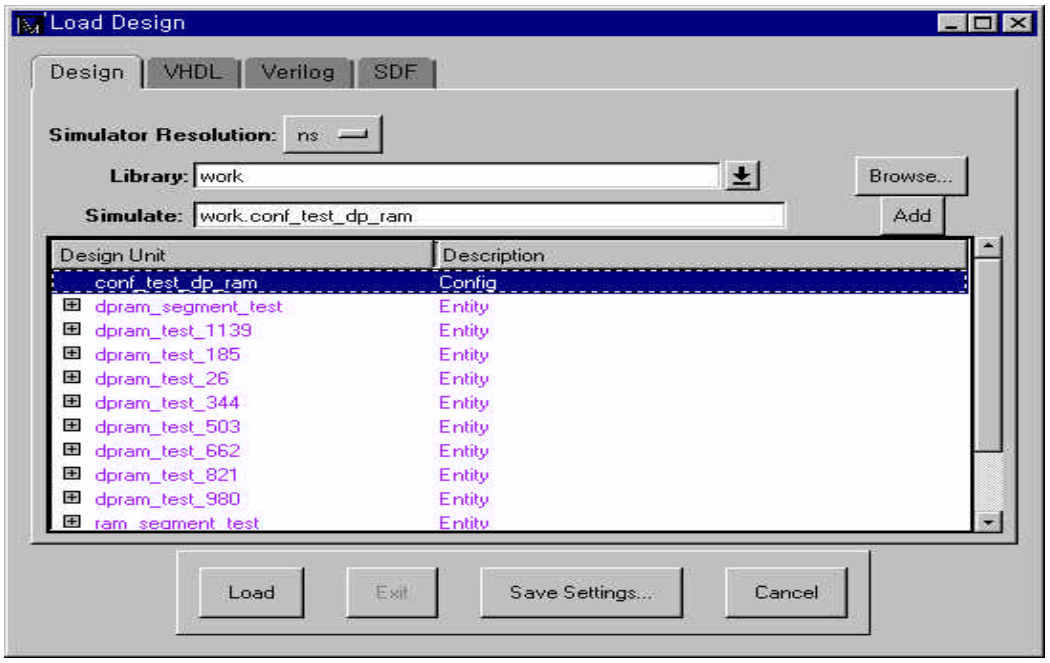


Figure 23 Load Design Window

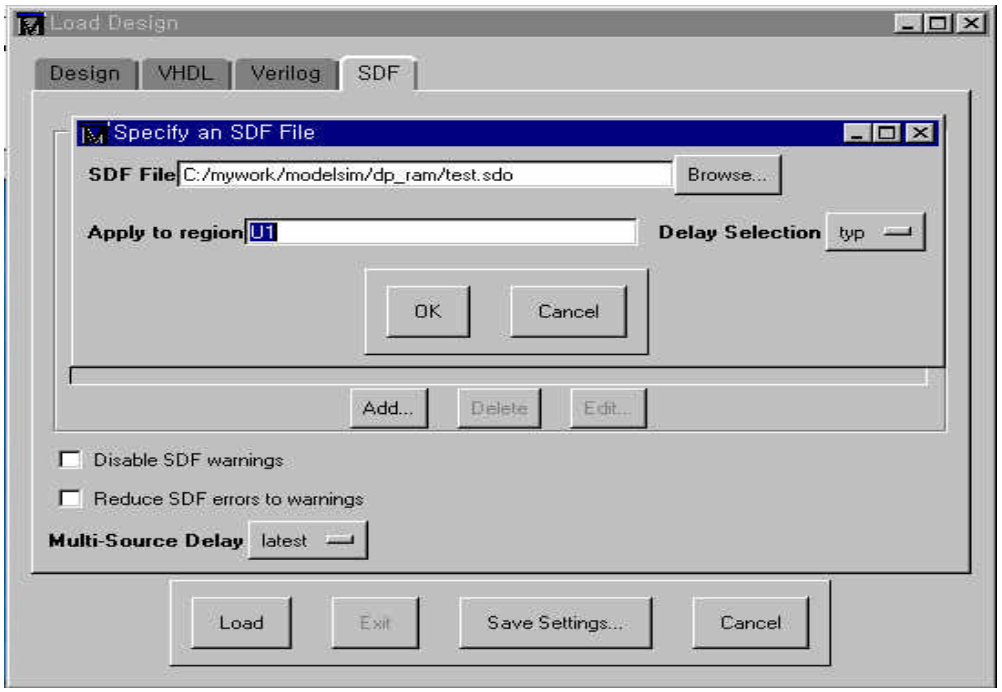


Figure 24 SDF Definition Window

위의 그림처럼 Add… Button 을 눌러 SDF File 이 있는 경로를 정의하고 Testbench File 에서 정의한 Instance Name 을 Apply to Region Field 에 정의한다. OK button 을 눌러 그 정보를 입력한 후 Load Button 을 눌러 디자인을 Loading 한다.

Functional Simulation 을 하는 경우와 같은 방법으로 나머지 과정을 수행한다.

그림 25 는 Timing Simulation 을 한 결과가 나타난 Waveform 이다.

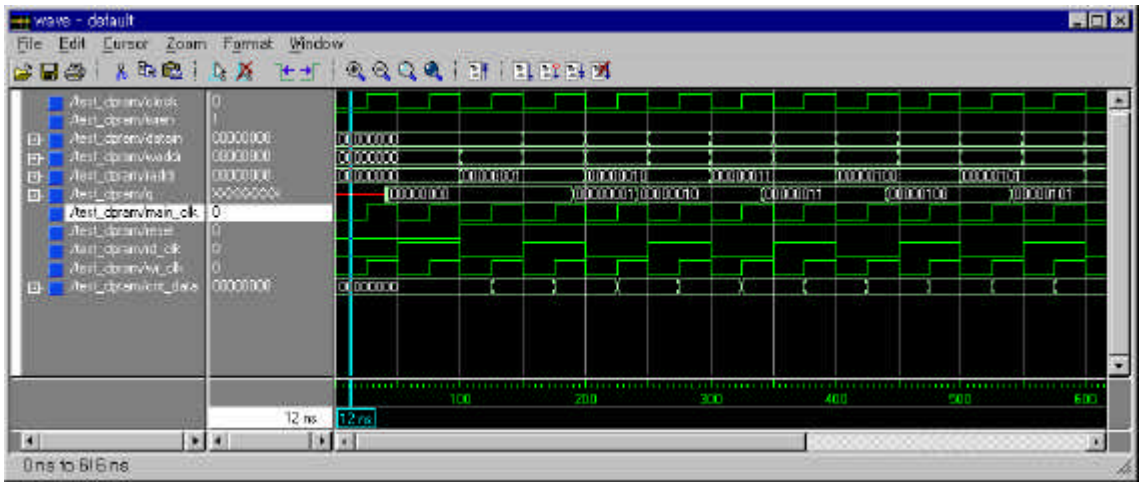


Figure 25 Result Waveform Window

Revision History

- 2000-7-13 - Ver 1.0: Initialize Release…
- 2000-8-8 - Ver 1.1: Text Modified by C.W.Yang…