ModelSim 사용법

1. ModelSim-Altera를 이용한 Function/RTL 시뮬레이션

1.1. 테스트벤치를 사용하지 않는 명령어 기반 시뮬레이션

1.1.1. 시뮬레이션을 위한 하드웨어

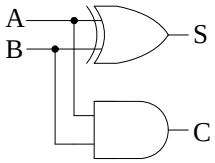


그림 1. 반가산기

1.1.2. 작업 디렉토리

- **File - Change Directory**를 클릭하여 작업 디렉토리 지정.

1.1.3. 소스 파일 작성

- 모델심 편집기나 기타 편집기 가능
- **New - Source - Verilog**
- **그림 3과 같이 반가산기 코드 작성**

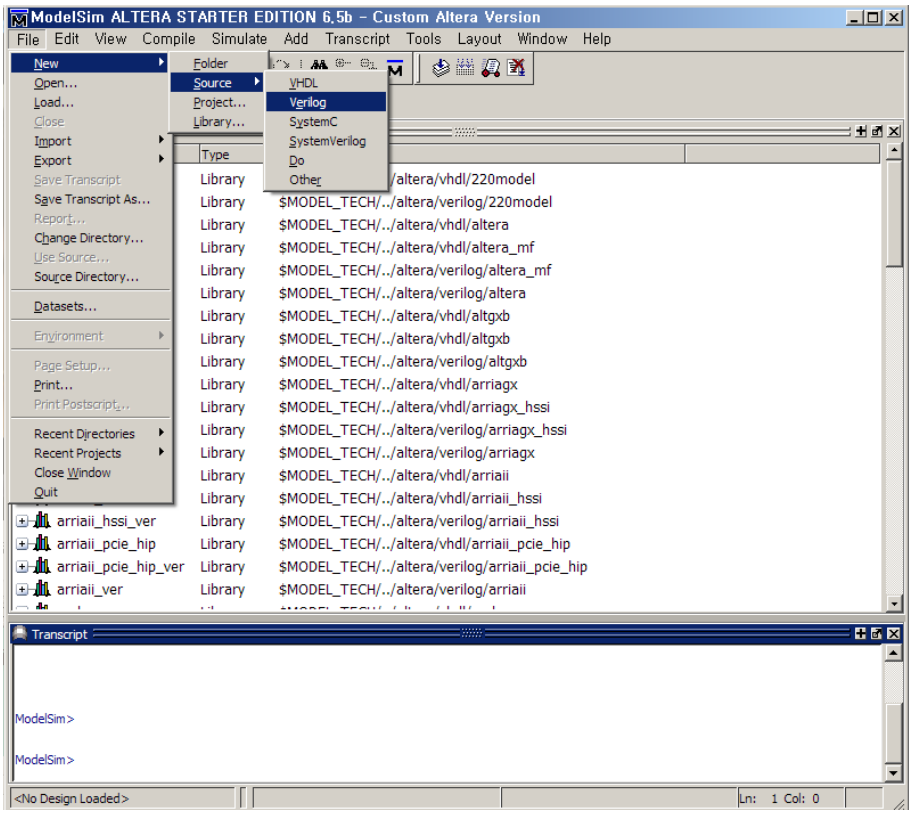


그림 2. 소스 파일 생성

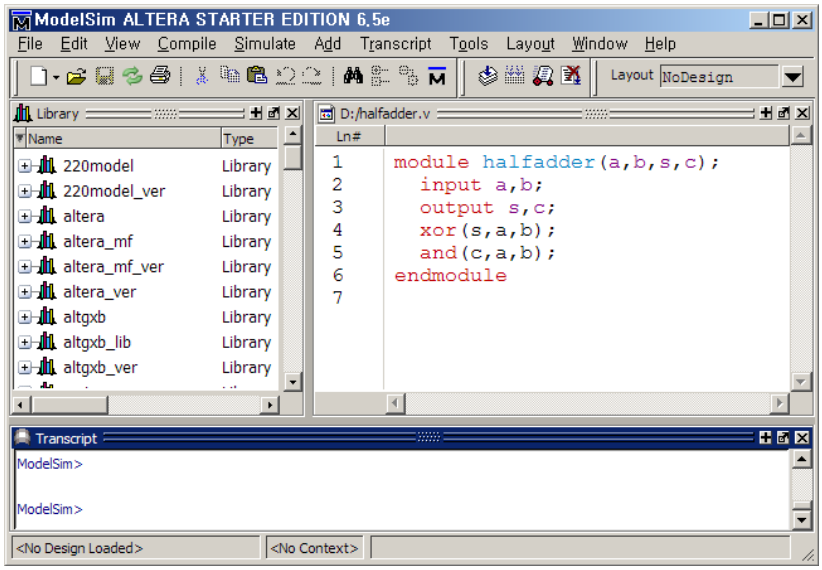


그림 3. 반가산기 소스 코드

1.1.4. 라이브러리 생성 및 소스 파일 컴파일

(a) 라이브러리 생성

모델심은 하드웨어 소스 파일을 시뮬레이션하기 위해 라이브러리를 요구한다. vlib은 라이브러리 생성 명령어로써, 수행 후 그림 4와 같이 작업 디렉토리에 work가 생성된다.

Command: vlib <라이브러리 이름>

```
vlib work
```

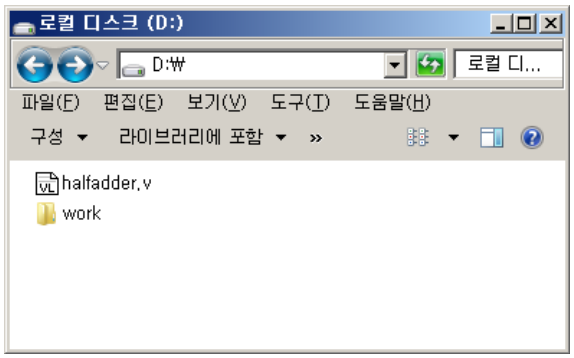


그림 4. 라이브러리 생성

(b) 소스 파일 컴파일

소스 파일 생성 후, 콘솔 창을 이용하여 컴파일과 시뮬레이션을 수행한다. 본 예제는 반가산기로써 하나의 halfadder.v 파일만 존재하기 때문에 아래와 같은 명령어로 콘솔에서 컴파일을 수행한다. 만일 여러 개의 파일들이 있다면 모든 파일에 대한 컴파일이 필요하다.

Command: vlog <소스 파일>

```
vlog halfadder.v
```

1.1.5. 시뮬레이션

(a) VSIM 시뮬레이터를 호출하여 모듈에 존재하는 모든 오브젝트(input, output, reg, wire)들을 그림 5와 같이 오브젝트 window에 적재한다. 최상위 모듈을 적재하면 하위 모듈은 자동으로 적재된다.

- 모듈 이름: 소스 파일의 이름(여러 개의 모듈이 존재하는 경우, 최상위 모듈 이름)

Command: vsim <모듈 이름>

```
vsim halfadder
```

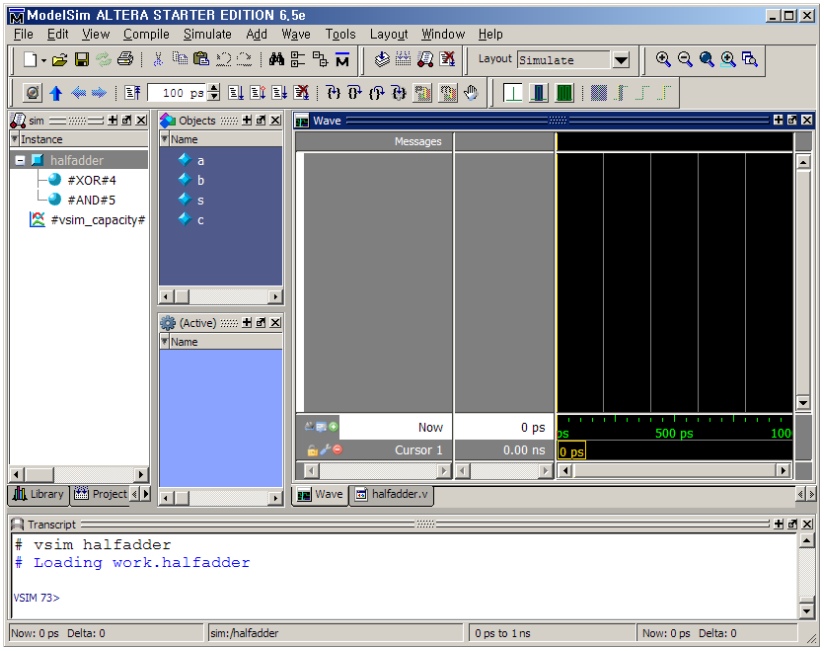


그림 5. VSIM 시뮬레이터를 통한 오브젝트 적재

(b) 오브젝트 윈도우에서 실제 시뮬레이션이 필요한 오브젝트들을 Wave 윈도우에 시그널로 추가한다. 아래 add 명령어는 모듈에 포함된 모든 오브젝트들을 추가하는 것이며, 특정 오브젝트만 추가할 경우 add wave halfadder/s와 같이 명시적으로 이름을 기재한다. 그림 6은 반가산기의 모든 오브젝트들을 추가한 경우이다.

Command: add wave <오브젝트 이름>

```
add wave halfadder/*
```

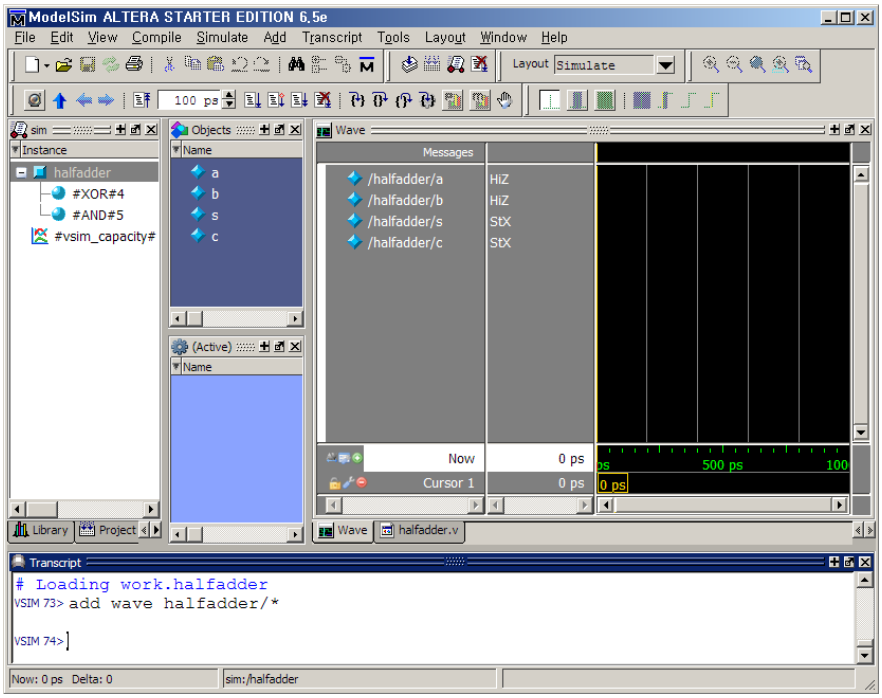


그림 6. Wave 윈도우에 시그널 추가

(c) Wave 윈도우의 시그널 값 설정. 테스트벤치 파일을 사용하지 않기 때문에 각 입력 시그널에 대해서 시간 별로 입력 값 설정이 필요하다. 시간 뒤에 단위를 지정하지 않으면, 기본적으로 ps 단위가 된다. 출력은 입력 값들에 의해 결정된다.

Command: force <오브젝트 name> <값1> <시간1>, <값2> <시간2>, … , <값n> <시간n>

force halfadder/a(또는 a) 0 0, 1 10, 0 20

force halfadder/b 0 0, 0 10, 1 20

(d) 시뮬레이션 실행. run 명령을 종료 시간과 함께 기술하여 시뮬레이션을 시작한다. 그림 7은 100ps까지 시뮬레이션이 실행된 결과이다.

Command: run <종료 시간>

run 100

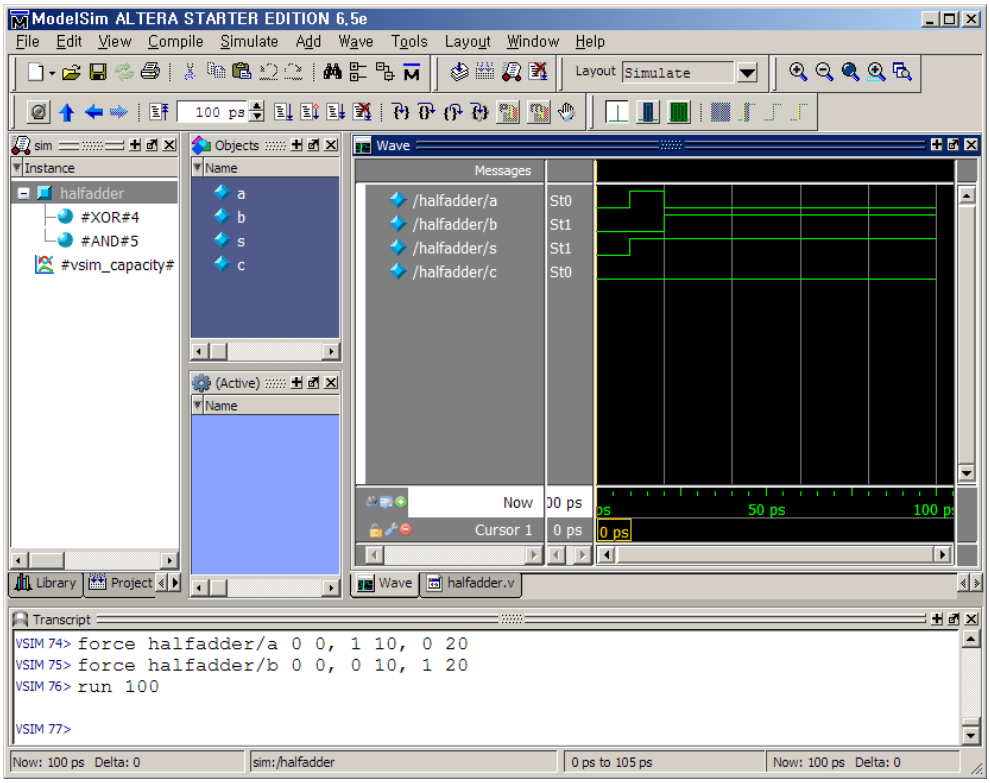


그림 7. 시뮬레이션 실행

(e) 시뮬레이션 재실행

Command: restart

(f) 시뮬레이션 종료

Command: quit -sim

1.2. 테스트벤치를 사용하지 않는 GUI 기반 시뮬레이션

1.2.1. 프로젝트 및 소스 파일 생성

(a) 작업 디렉토리 지정 1.1.2와 동일

(b) 프로젝트 생성

메뉴 **New - Project** 선택. 그림 8의 다이얼로그 박스에서, 프로젝트 이름과 Default Library Name 작성. 본 예제에서는 반가산기를 생성하기 위해, 프로젝트 이름을 halfadder로 작성하고 라이브러리는 기본 work를 사용. OK 버튼을 클릭.

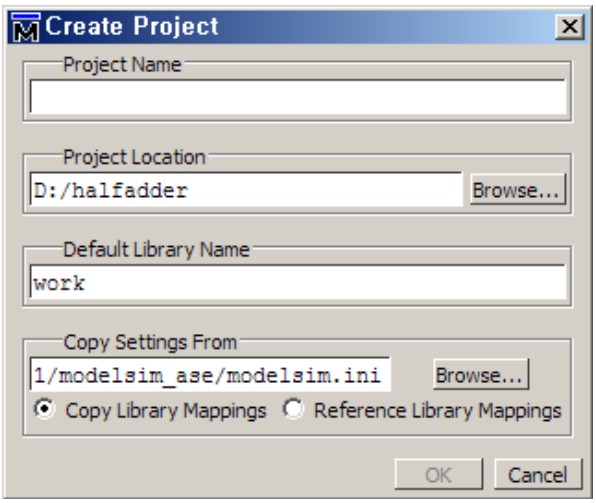
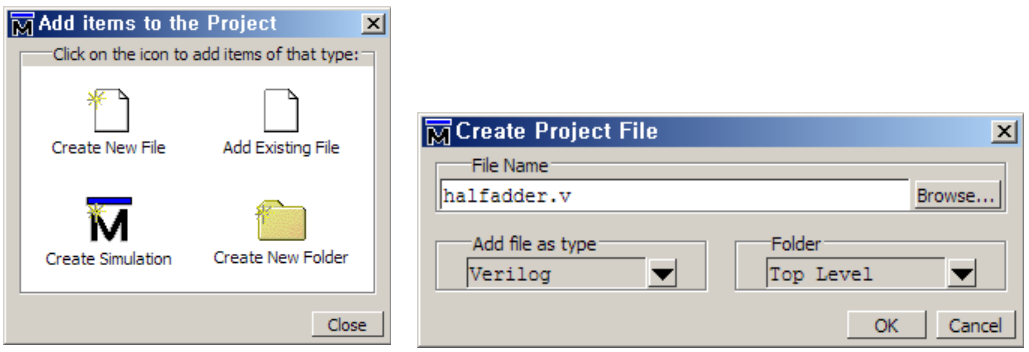


그림 8. 프로젝트 생성

(c) 소스 파일(설계 모듈) 생성

소스 파일이 있는 경우: 그림 9(a)에서 Add Existing File 선택하여 작업 디렉토리에서 소스 파일 선택.

소스 파일이 없는 경우: Create New File 선택. 그림 9(b)에서 파일 이름을 halfadder(또는 halfadder.v)로 기재하고 파일 타입은 Verilog를 선택. 완료프로젝트 생성 결과하면 그림 10과 같이 빈 소스 파일을 포함하는 프로젝트 생성.



(a) (b)

그림 9. 프로젝트에 소스 파일 추가

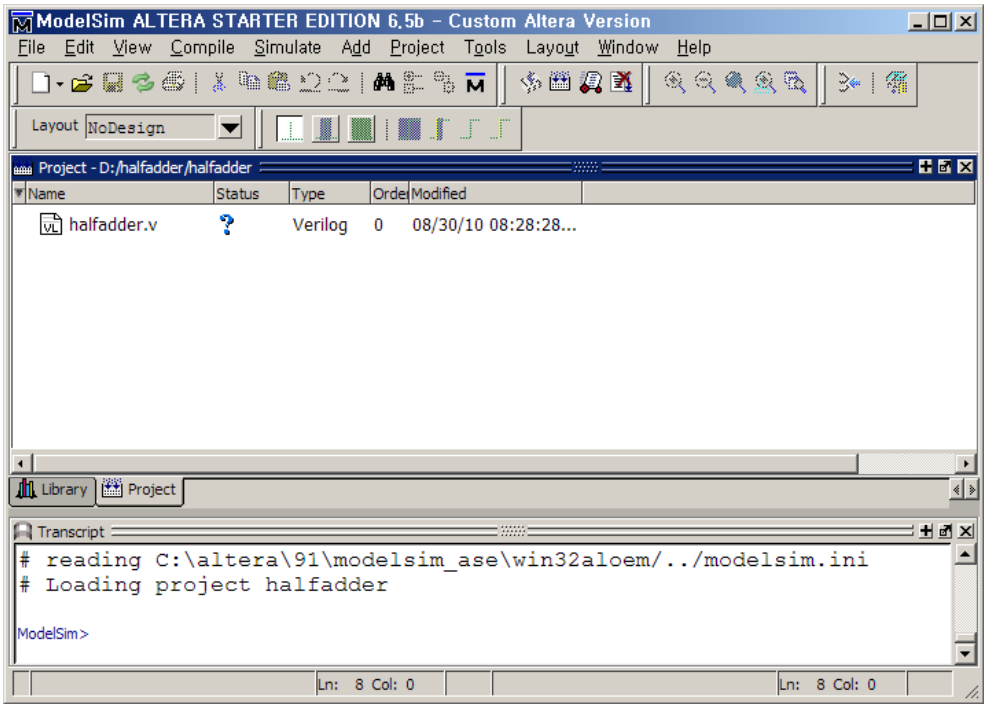


그림 10. 프로젝트에 파일 추가

(d) 소스 파일 작성.

- 파일 이름에서 마우스 오른쪽 버튼을 클릭하여 Edit 선택. 프로젝트 오른쪽에 Edit 윈도우 생성. 그림 3과 같이 반가산기 코드 작성.

1.2.2. 소스 파일 컴파일

작성된 파일을 저장하고 그림 11에 1에 표시된 컴파일 아이콘 클릭 (또는 메뉴에서 **Compile - Compile All** 선택).

1.2.3. 시뮬레이션

(a) 컴파일이 완료 후, 그림 11의 2에 표시된 시뮬레이션 아이콘 클릭 (또는 메뉴에서 **Simulate - Start Simulation** 선택). 그림 12의 Start Simulation 다이얼로그 박스에서 work 라이브러리를 확장하여 halfadder 모듈을 선택하고 OK 버튼 클릭하면 그림 13과 같이 시뮬레이션을 위한 오브젝트 적재 작업이 완료된다.

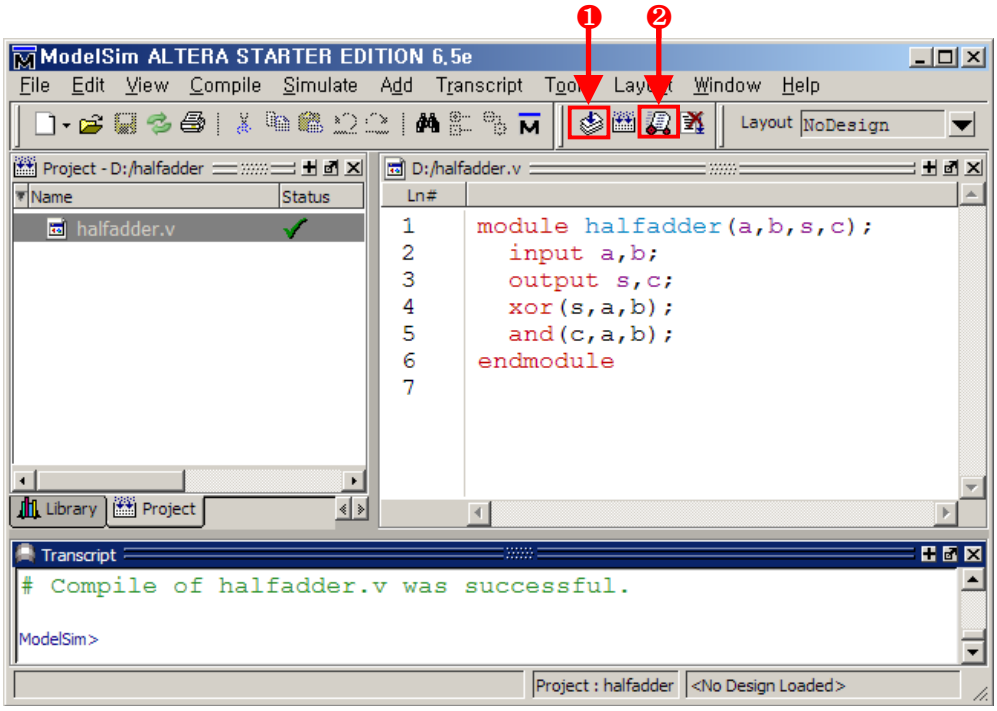


그림 11. 컴파일 및 시뮬레이션

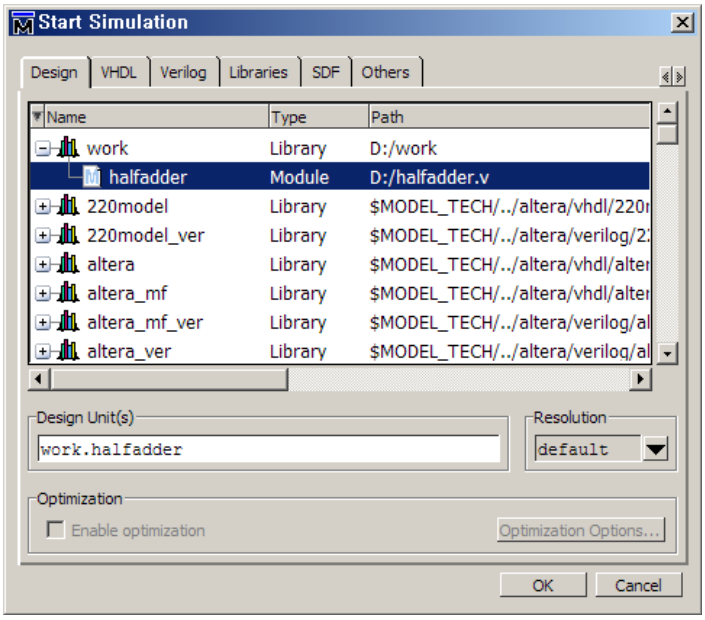


그림 12. Start Simulation

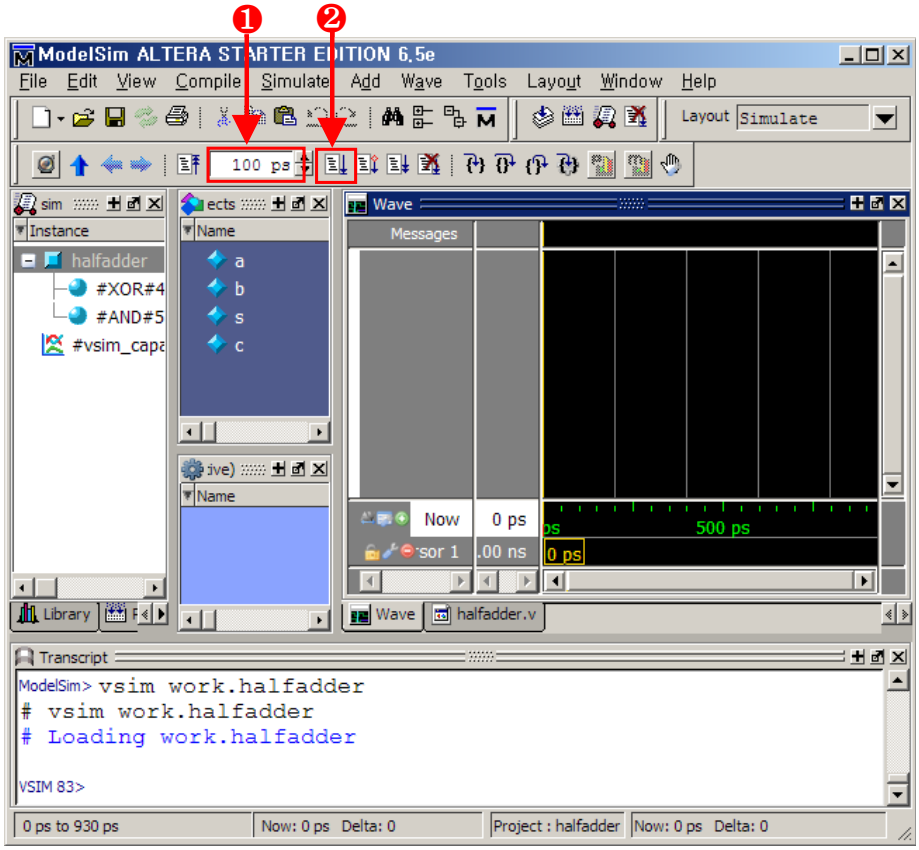


그림 13. VSIM 시뮬레이터를 통한 오브젝트 적재

(b) 시뮬레이션이 필요한 오브젝트들을 드래그하여 Wave 윈도우에 시그널로 추가.

(c) ***force*** 명령어를 통해 각 입력 시그널에 값 설정. 그림 13에서 시뮬레이션 종료 시간 설정(1번) 후, 시뮬레이션 시작(2번).

1.3. 테스트벤치를 사용한 GUI기반 Function 시뮬레이션

- 작업 디렉토리와 프로젝트 생성은 1.2.1의 (a),(b)와 같은 순서로 수행
- 1.2.1의 (c)에서 반가산기와 그림 14의 테스트벤치 소스 파일이 (tb_halfadder.v) 존재.

```
module tb_halfadder;
    reg a, b;
    wire s, c;

    initial begin
      a = 1'b0; b = 1'b0;
      #10 a = 1'b1;
      #10 a = 1'b0; b = 1'b1;
    end
    halfadder u1 (
      .a(a),
      .b(b),
      .s(s),
      .c(c)
    );
endmodule
```

그림 14. 반가산기 테스트벤치 파일

- 그림 9에서 Add Existing File을 선택하여 halfadder.v와 tb_halfadder.v 파일을 추가.

1.3.1. 소스 파일 컴파일

- 그림 15와 같이 반가산기와 테스트벤치 소스 파일이 추가되었으면 메뉴에서 **Compile - Compile All**

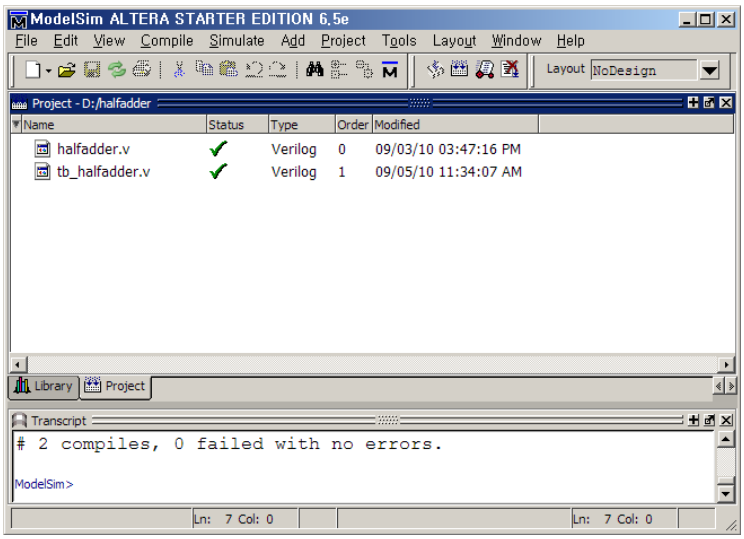


그림 15. 소스 파일 컴파일

1.3.2. 시뮬레이션

(a) **Simulate - Start Simulate**(또는 시뮬레이션 아이콘) 선택

(b) 그림 12의 Start Simulation 다이얼로그 박스가 나타나면 work 라이브러리 하위에 halfadder와 tb_halfadder가 포함된 것을 볼 수 있다. 명령어 기반 시뮬레이션과 달리 테스트벤치 파일을 클릭하고 OK 버튼 클릭. 그림 16과 같이 최상위 모듈 tb_halfadder가 하위에 반가산기를 인스턴스로 포함한다. 시뮬레이션에 필요한 오브젝트들을 wave 윈도우로 드래그한 후 그림 16의 run 아이콘(1번)을 클릭하면 시뮬레이션이 시작된다. 시뮬레이션은 아이콘 왼쪽에 표시된 시간에 종료된다.

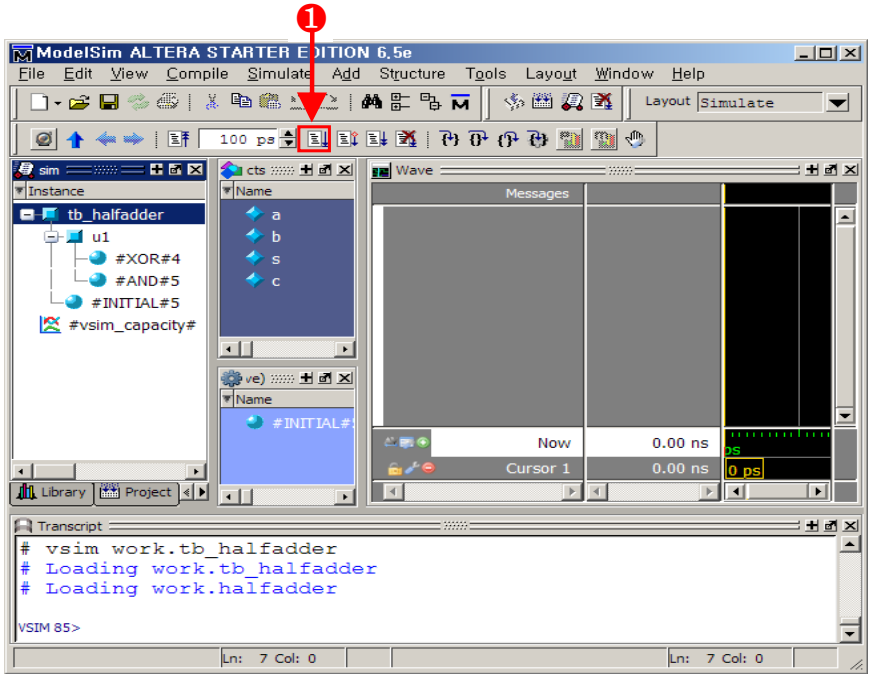


그림 16. VSIM 시뮬레이터를 통한 오브젝트 적재

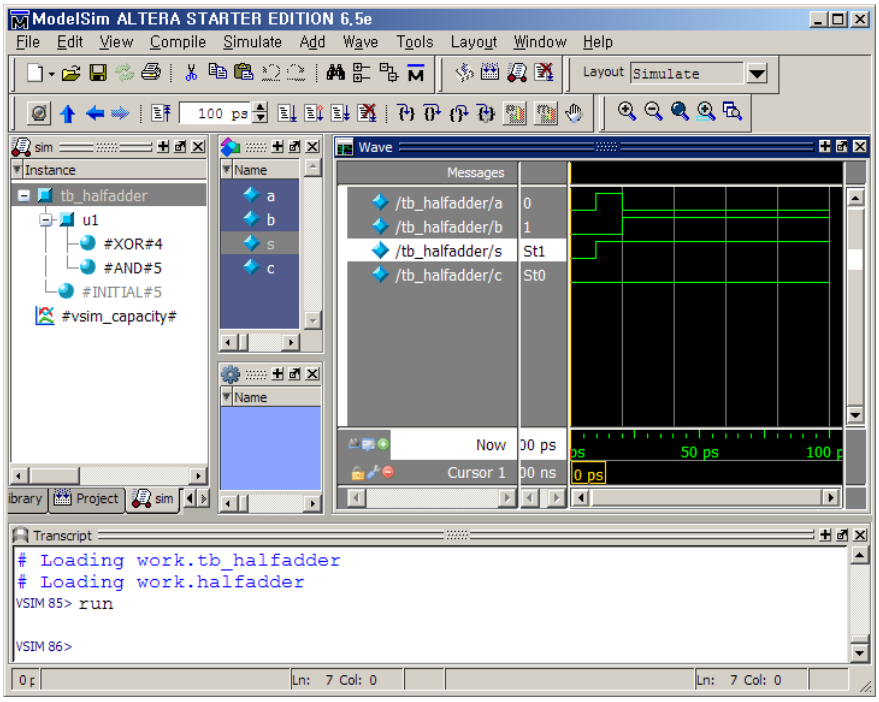


그림 17. 시뮬레이션 결과

2. ModelSim-Altera를 이용한 Gate-Level Timing 시뮬레이션

2.1. 설계 파일

Counter

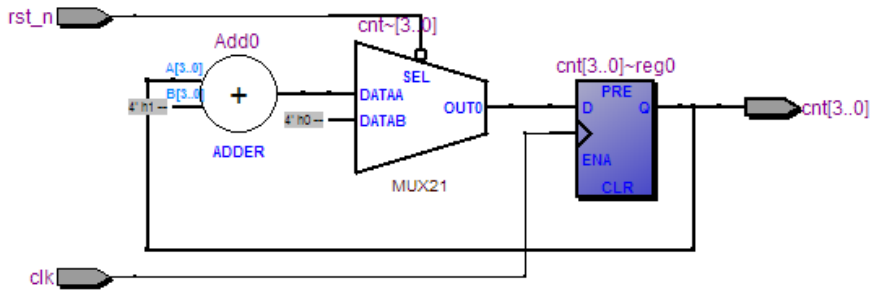


그림 18. 카운터 블록도

2.2. Gate-Level Timing 시뮬레이션에 필요한 파일

1) Netlist 파일: *.vo
2) Standard Delay Format Output File(SDF): *.sdo

2.3. Gate-Level Timing 시뮬레이션을 위한 Quartus II 프로젝트 생성

1) 프로젝트 생성
   - **File - New Project Wizard**

3) 디렉토리 지정 및 프로젝트 이름(Top-Level Entity와 동일) 입력

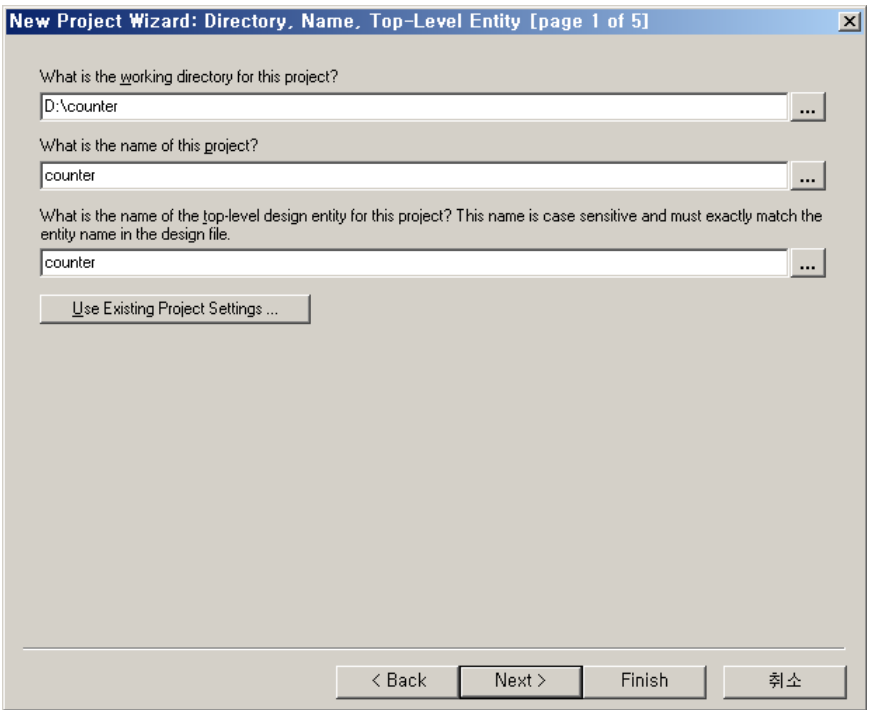


그림 19. 프로젝트 생성 (1/5)

4) 소스 파일이 존재하는 경우
- 그림 20에서 Browse하여 존재하는 소스 파일을 프로젝트에 추가

5) 소스 파일이 없는 경우
- 그림 20에서 Next 클릭

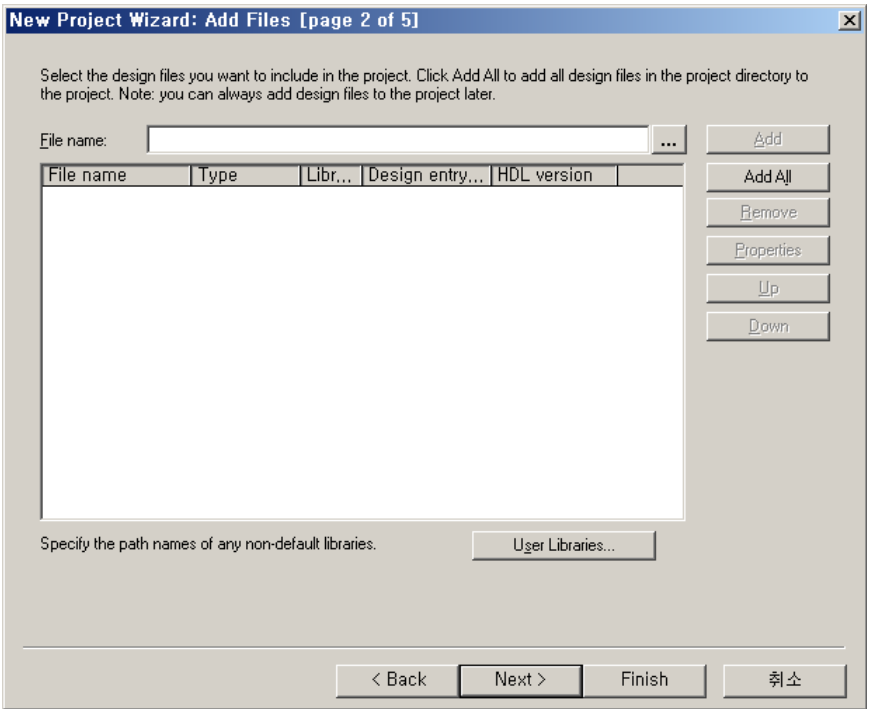


그림 20. 프로젝트 생성 (2/5)

6) 사용할 디바이스 선택 (Family: Cyclone II, Specific Name: EP2C35F672C8)

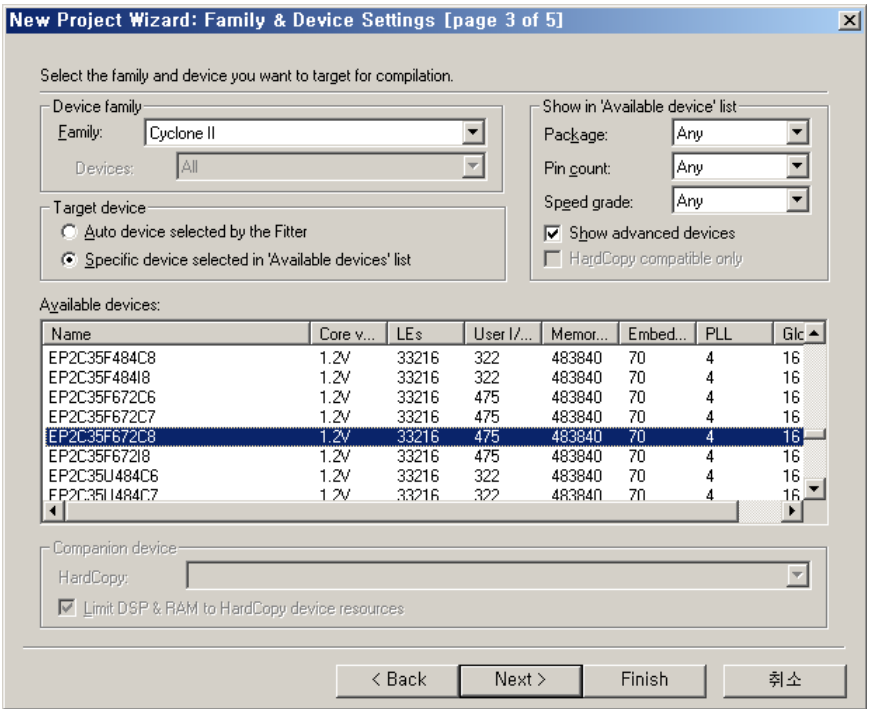


그림 21. 프로젝트 생성 (3/5)

7) Gate-Level Timing 시뮬레이션을 위한 Tool과 HDL 타입 설정

- Tool name: ModelSim-Altera
- Format: Verilog HDL

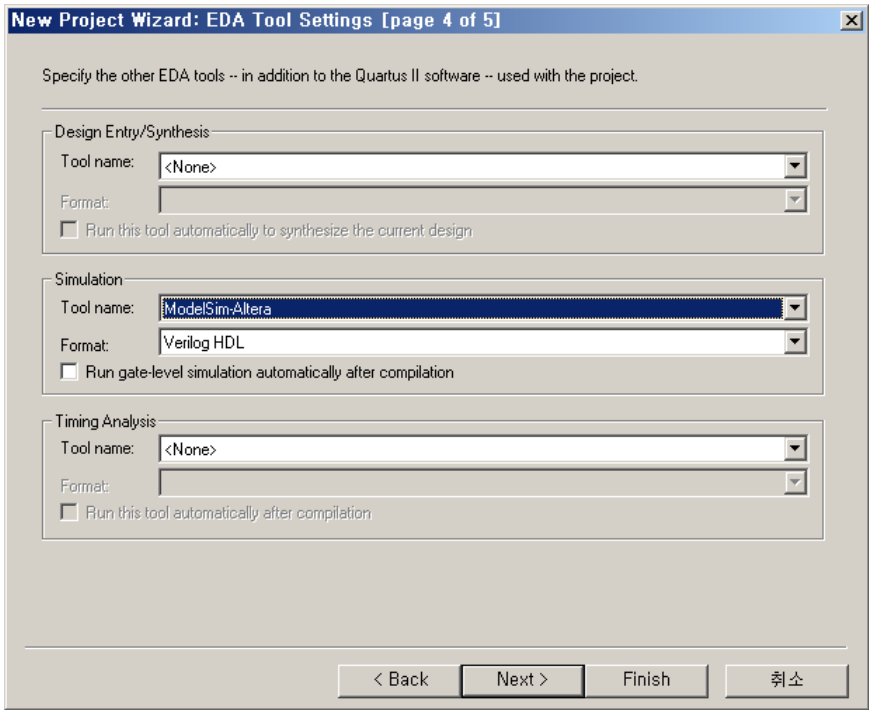


그림 22. 프로젝트 생성 (4/5)

8) 카운터 소스 파일 생성

- **File - New**
- Design File - Verilog HDL File 선택

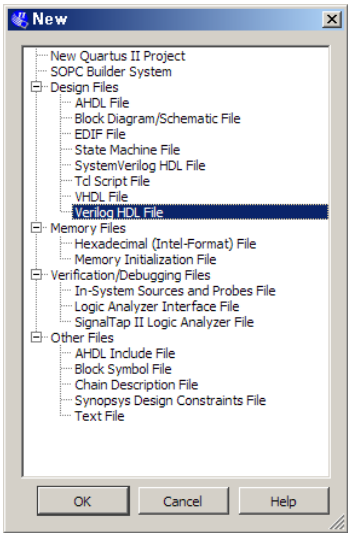


그림 23. 소스 파일 생성

2.4. Timing 시뮬레이션을 위한 Netlist 파일 생성

- Netlist 파일 생성을 위한 Quartus II 프로젝트 설정

1) 프로젝트 설정

- **Assignments - Settings**
- Category에서 Simulation 선택. 그림 24에서 Tool name과 EDA Netlist Writer settings 내용 확인

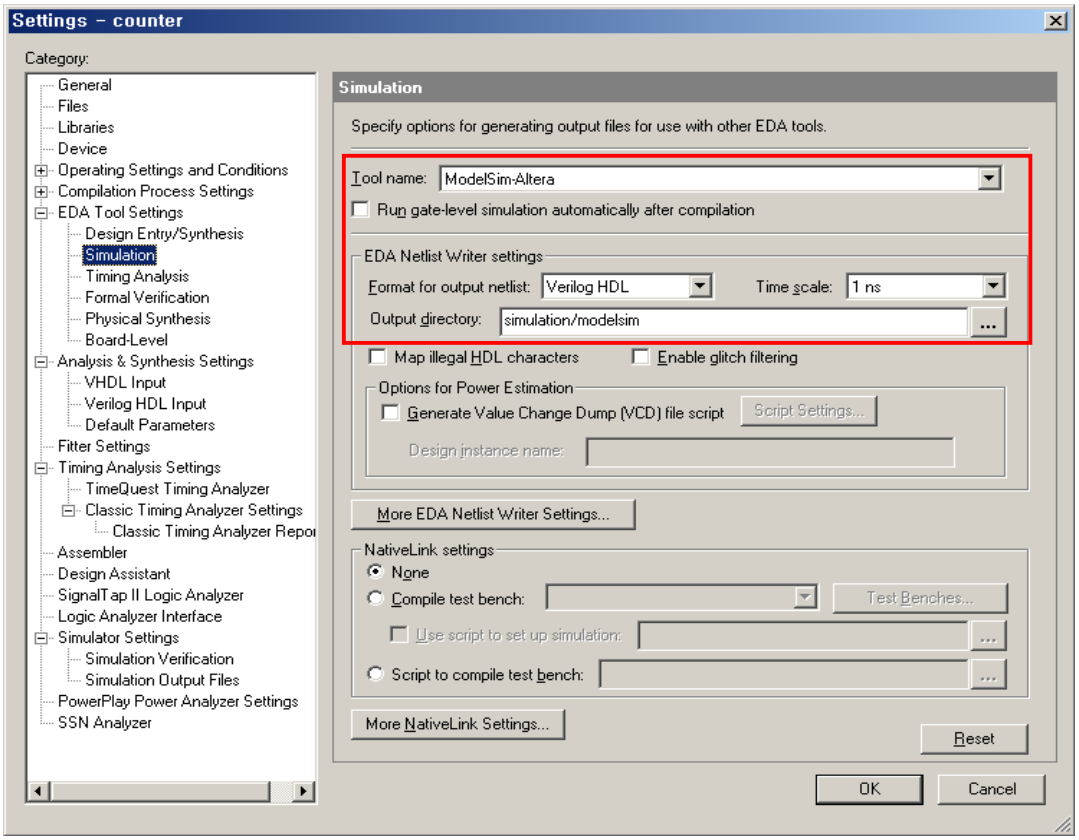


그림 24. Netlist 파일 저장 위치 지정

9) NetiveLink settings에서 테스트벤치 설정(이전에 테스트벤치 파일 작성이 완료되어야 함)

- 그림 25에서 Test Benches 클릭
- 그림 26에서 New 선택
- 그림 27과 같이 테스트벤치 이름과 Top Level Module 이름 작성
- 테스트벤치에 사용된 인스턴스 이름 작성
- 시뮬레이션 종료 시간 설정
- 테스트벤치 파일 추가 (Browse 사용)

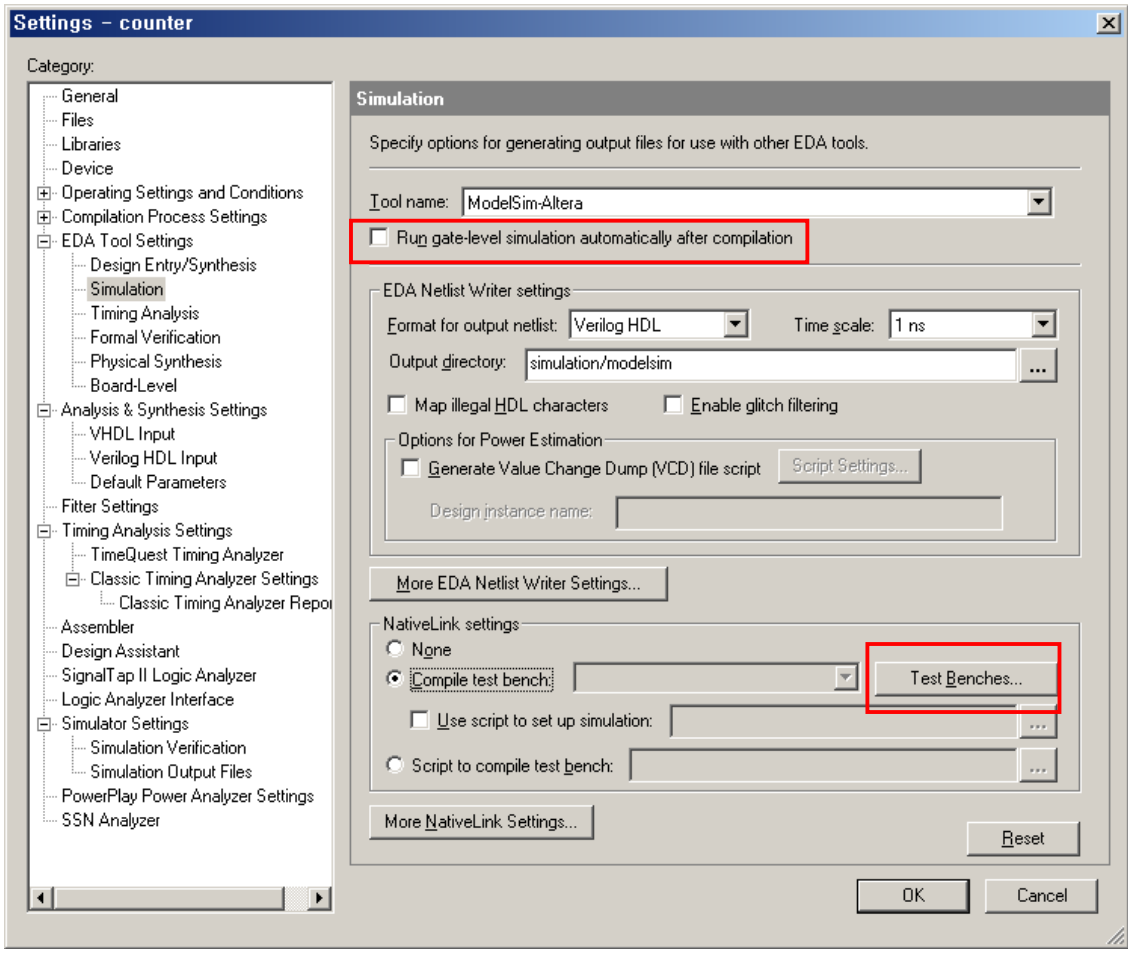


그림 25. 테스트벤치 컴파일

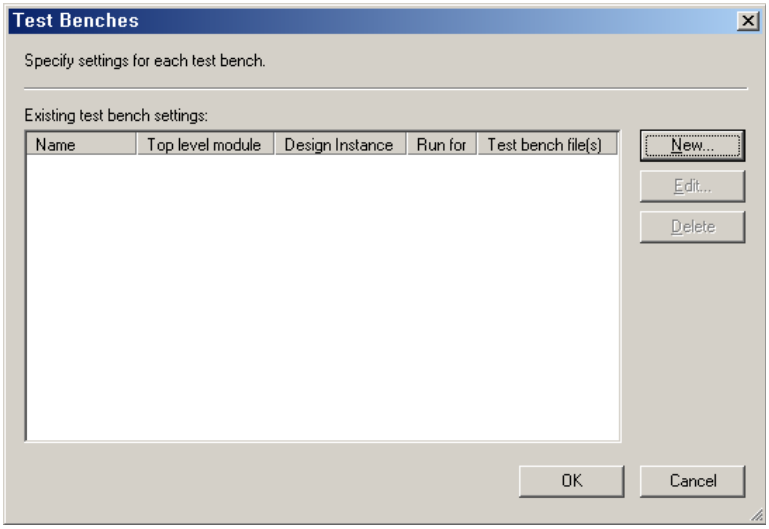


그림 26. 테스트벤치 추가

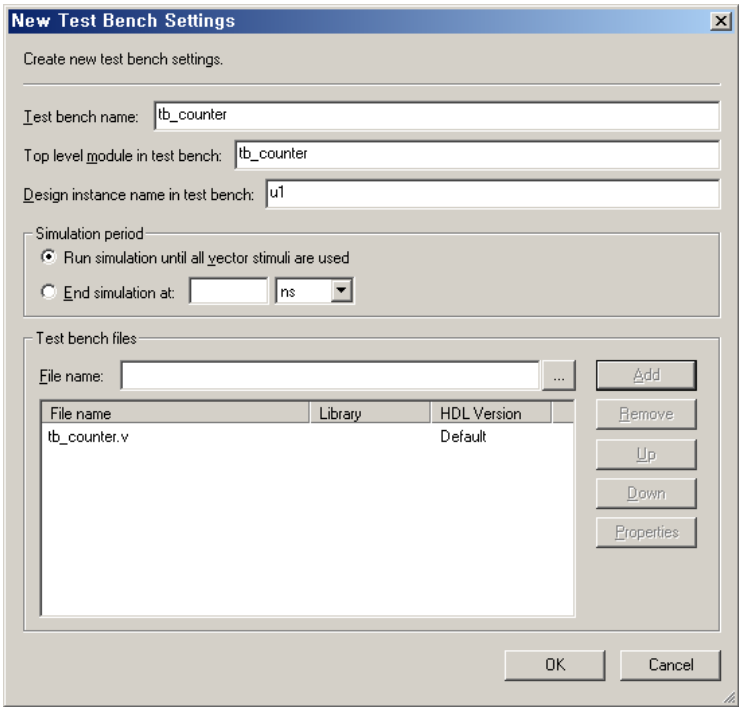


그림 27. 테스트벤치 설정

10) ModelSim-Altera 경로 설정

- **Tools - Option**
- 그림 28과 같이 EDA Tool Options에서 ModelSim-Altera가 설치된 경로 확인

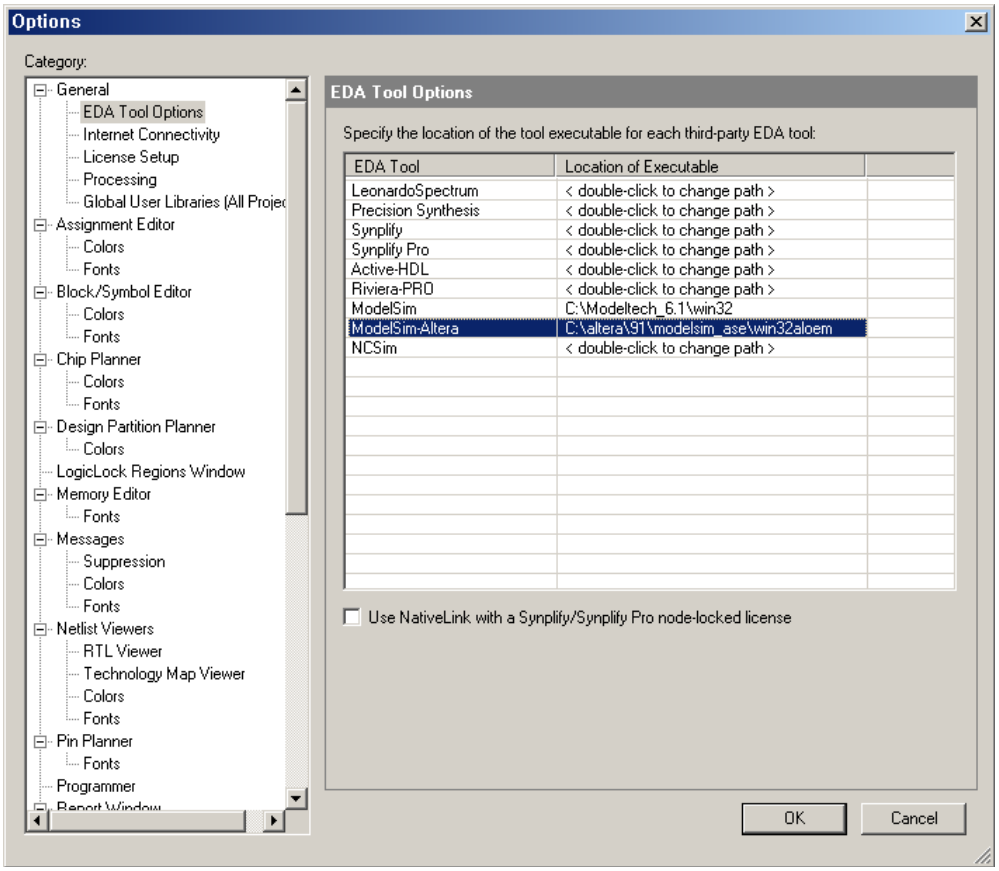


그림 28. ModelSim-Altera 경로 설정

11) 프로젝트 컴파일

- 그림 25에서 Run gate-level simulation automatically after compilation을 클릭하면 컴파일 완료 후 자동으로 ModelSim-Altera로 시뮬레이션 된다.

12) 생성된 Netlist 파일(counter.vo)과 SDF(counter_v.sdo) 확인

- 다음 디렉토리에 저장: **"Quartus Ⅱ 프로젝트 디렉토리₩simulation₩modelsim"**

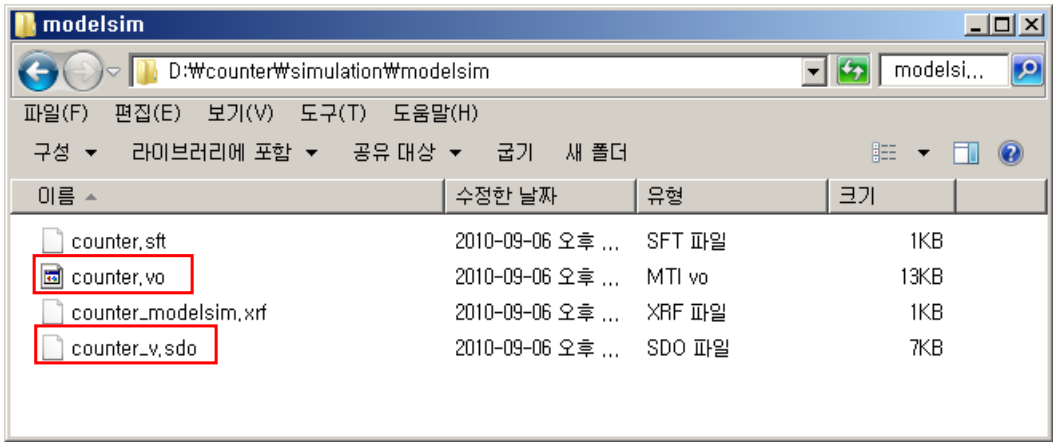


그림 29. 생성된 Netlist와 SDF 파일

2.5. ModelSim-Altera 프로젝트 생성

- Run gate-level simulation automatically after compilation을 사용하지 않고 ModelSim-Altera를 독립적으로 실행하여 시뮬레이션.

1) 작업 디렉토리 변경(counter.vo, counter_v.sdo가 생성된 디렉토리)
2) work 라이브러리 생성
   - **File - New - Library**

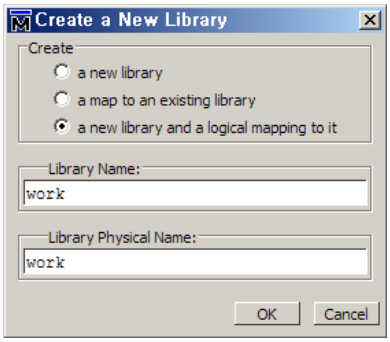


그림 30. 라이브러리 생성

3) 프로젝트 생성
   - **File - New - Project**

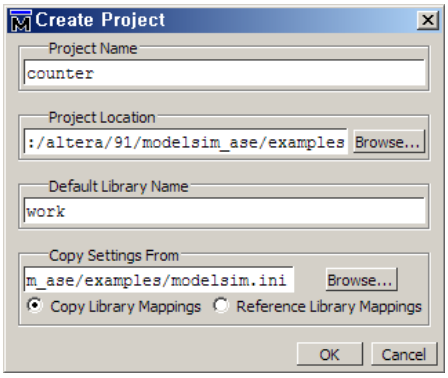


그림 31. 프로젝트 생성

4) 그림 32의 Add Existing File으로 파일 추가
   - **"Quartus Ⅱ 프로젝트 디렉토리₩simulation₩modelsim"**의 Netlist 파일(counter.vo).
   - **"Quartus Ⅱ 프로젝트 디렉토리"**의 테스트벤치 파일(tb_counter.v).

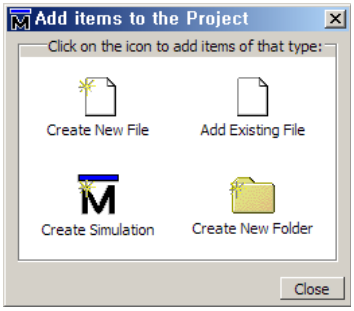


그림 32. 소스 파일 추가

2.6. 컴파일

1) 프로젝트에 추가된 Netlist와 테스트벤치 컴파일
2) 두 개의 파일 사이에 종속성이 있는 경우, 컴파일 순서는 시뮬레이션에 영향을 미친다. 때문에 종속성이 명확하지 않으면 그림 33의 프로젝트 탭에서 오른쪽 마우스 버튼을 클릭하여 그림 34의 Auto Generate를 사용한다.

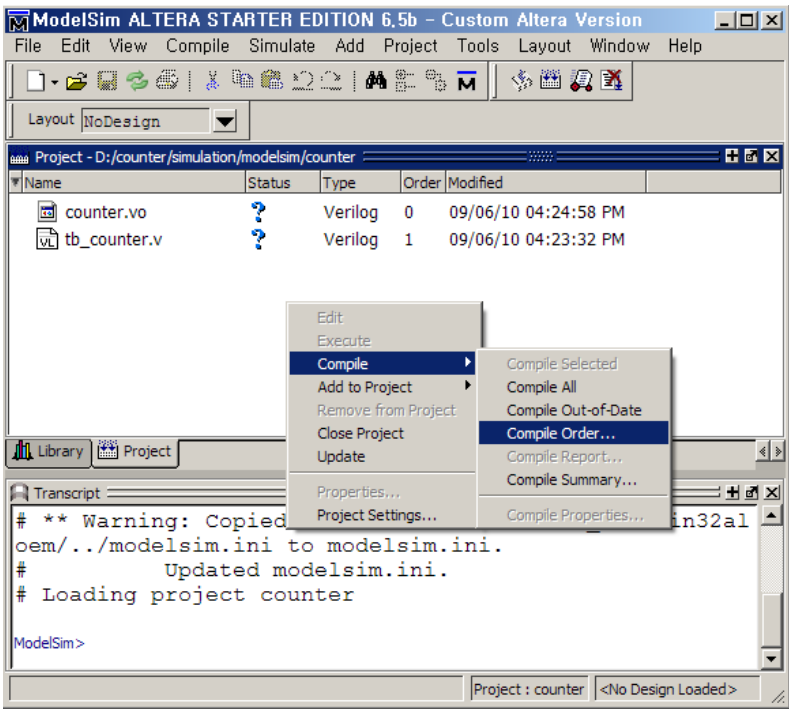


그림 33. 컴파일

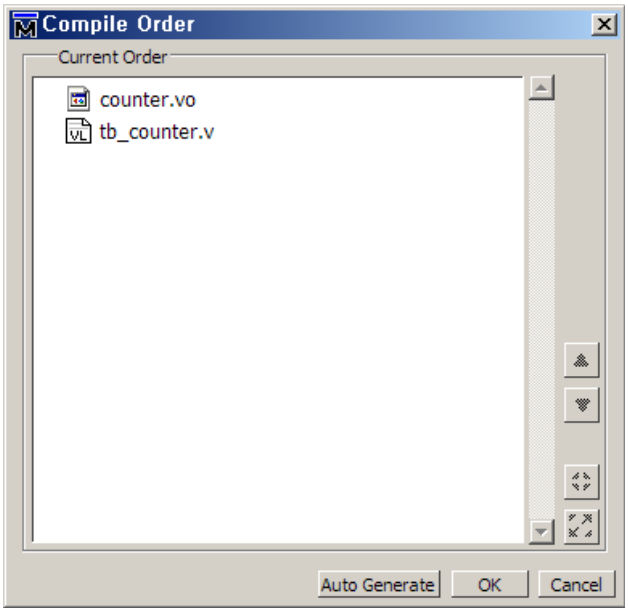


그림 34. 컴파일 순서

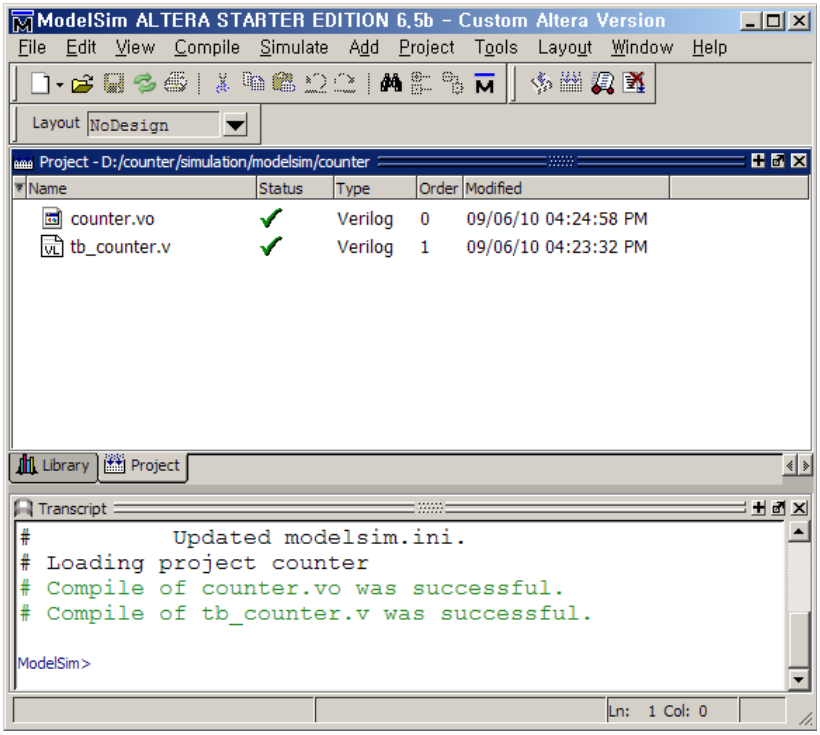


그림 35. 컴파일 완료

2.7. 시뮬레이션

1) **Simulate - Start Simulate**
2) 그림 36의 Library 탭 클릭, Search Libraries (-L)에서 사용하는 FPGA 디바이스 라이브러리를 선택. 제공되는 라이브러리는 VHDL과 Verilog 버전 존재. Verilog는 다음과 같이 "디바이스이름_ver" 라이브러리를 사용한다. 예를 들어, CycloneII는 cycloneii_ver를 추가한다.

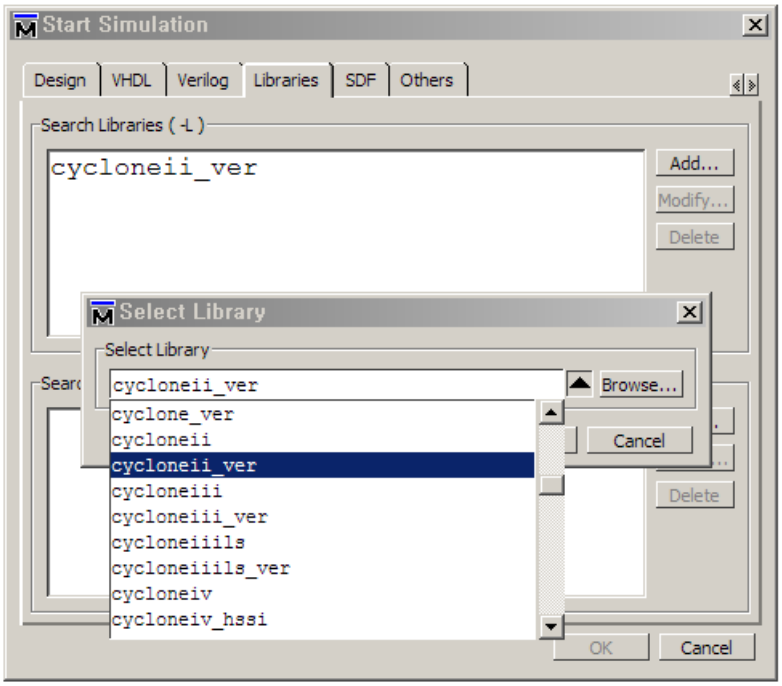


그림 36. 디바이스 라이브러리 추가

3) 라이브러리가 추가되면 Design 탭을 클릭하여 그림 37과 같이 테스트벤치를 선택한다. OK 버튼을 누르면 테스트벤치 모듈 내에 존재하는 모든 오브젝트들이 적재된다.

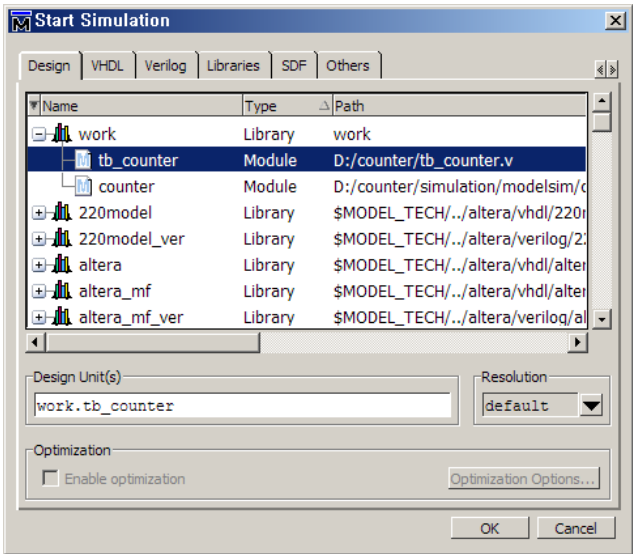


그림 37. 시뮬레이션

4) 시뮬레이션이 필요한 오브젝트들을 wave 창으로 드래그한다.
5) 그림 13의 종료시간(1번)을 설정하고 run 아이콘(2번)을 클릭하거나 run <종료시간> 명령어로 시뮬레이션을 시작할 수 있다.
6) Netlist와 SDF 파일을 이용하여 카운터에 대한 Gate-Level Timing 시뮬레이션 결과는 그림 38과 같다.

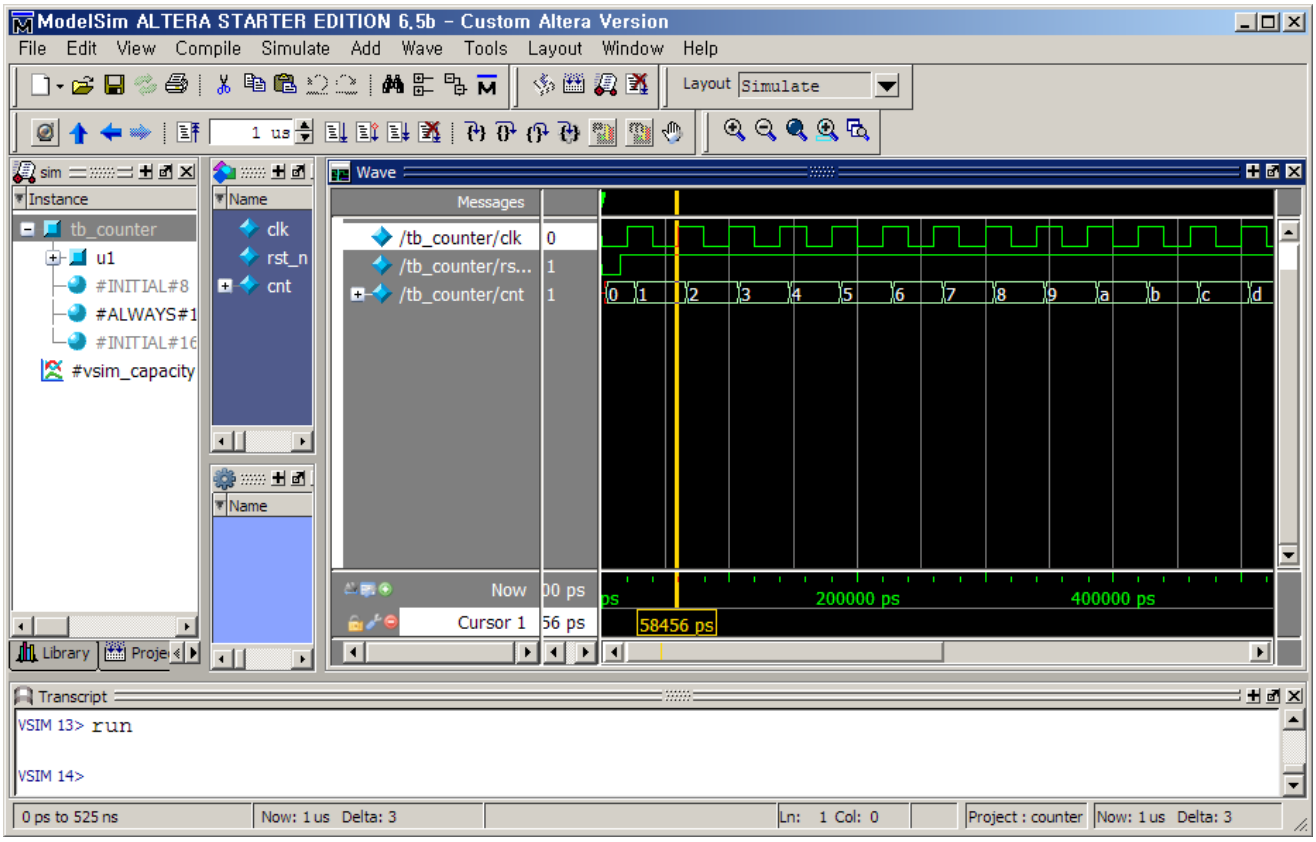


그림 38. 카운터에 대한 Gate-Level 시뮬레이션