EDA技术实用教程

第4章

时序仿真与硬件实现

4.1 VHDL程序输入和编译

4.1.1 编辑和输入设计文件

【例 4-1】此程序后半部分与例 3-13 相同!

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
USE IEEE.STD_LOGIC_UNSIGNED.ALL ;
USE IEEE.STD_LOGIC_ARITH.ALL ;
ENTITY MULT4B IS
GENERIC (   S   : INTEGER := 4);
   PORT (   RX   : OUT STD_LOGIC_VECTOR(7 DOWNTO 0);
        AX, BX   : IN STD_LOGIC_VECTOR (3 DOWNTO 0) );
END ENTITY MULT4B;
ARCHITECTURE ONE OF MULT4B IS
   SIGNAL     A0 :  STD_LOGIC_VECTOR(2*S DOWNTO 1);
   SIGNAL      R  :  STD_LOGIC_VECTOR(2*S DOWNTO 1);
   SIGNAL  A, B  :  STD_LOGIC_VECTOR (S DOWNTO 1));
BEGIN
   RX <= R ;  A <= AX ; B <= BX ;
   A0 <= CONV_STD_LOGIC_VECTOR(0,S) & A;
 PROCESS (A, B)
  . . . . . .
```

4.1 VHDL程序输入和编译

4.1.1 编辑和输入设计文件

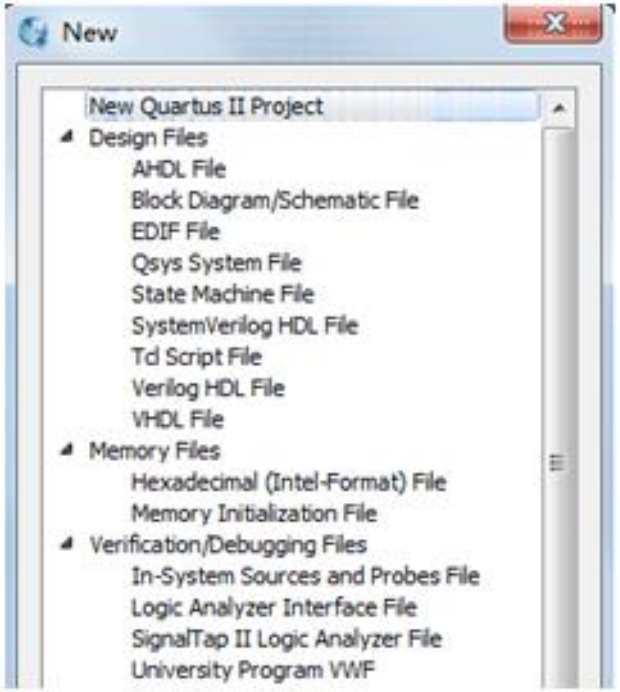

图 4-1　选择编辑文件类型

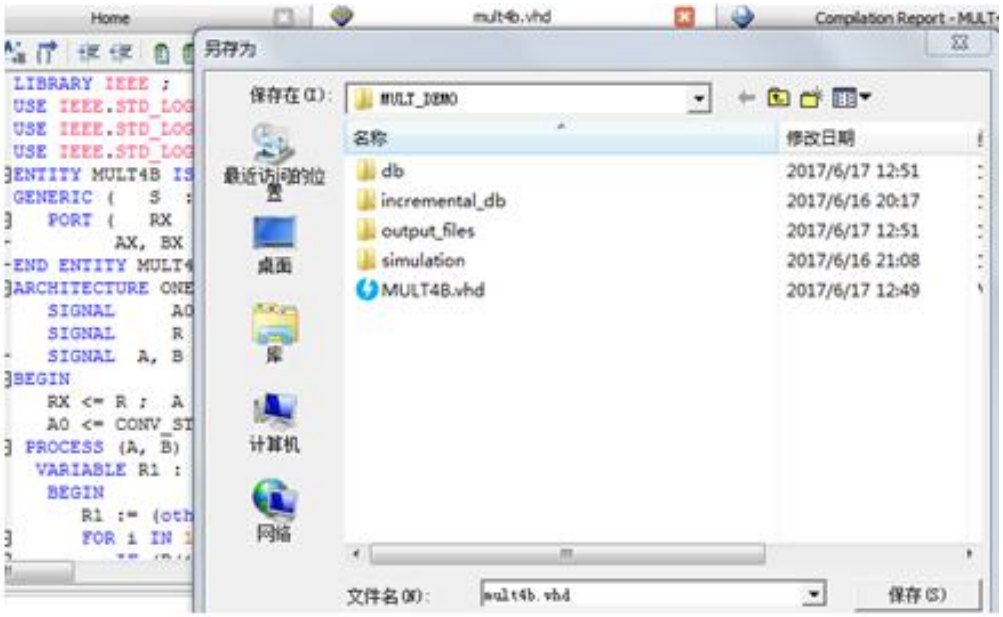

图 4-2　编辑输入源程序并存盘

4.1 VHDL程序输入和编译

4.1.2 创建工程

（1）打开并建立新工程管理窗口

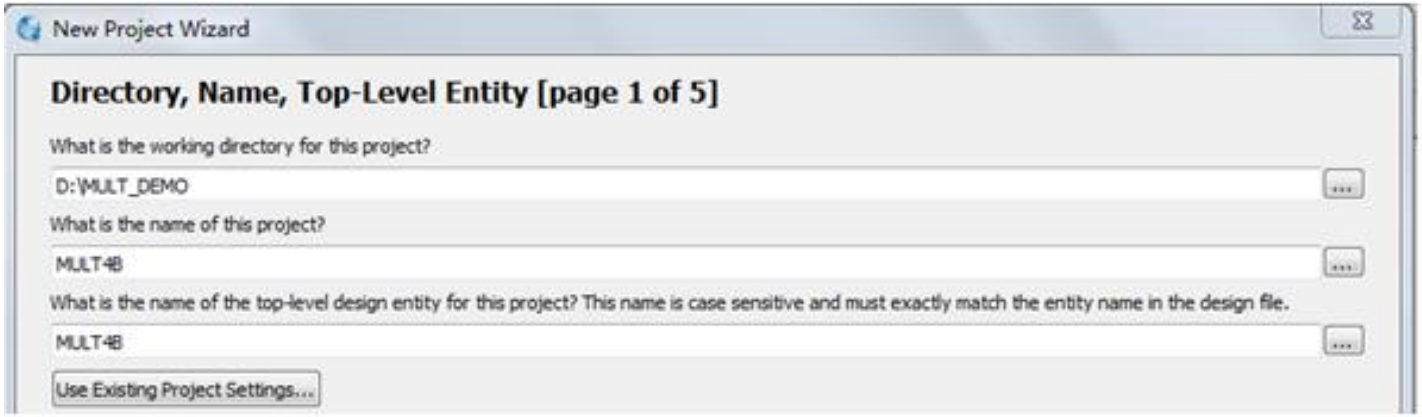

图 4-3　利用 New Project Wizard 创建工程 MULT4B

（2）将设计文件加入工程中

4.1 VHDL程序输入和编译

4.1.2 创建工程

（3）选择目标芯片

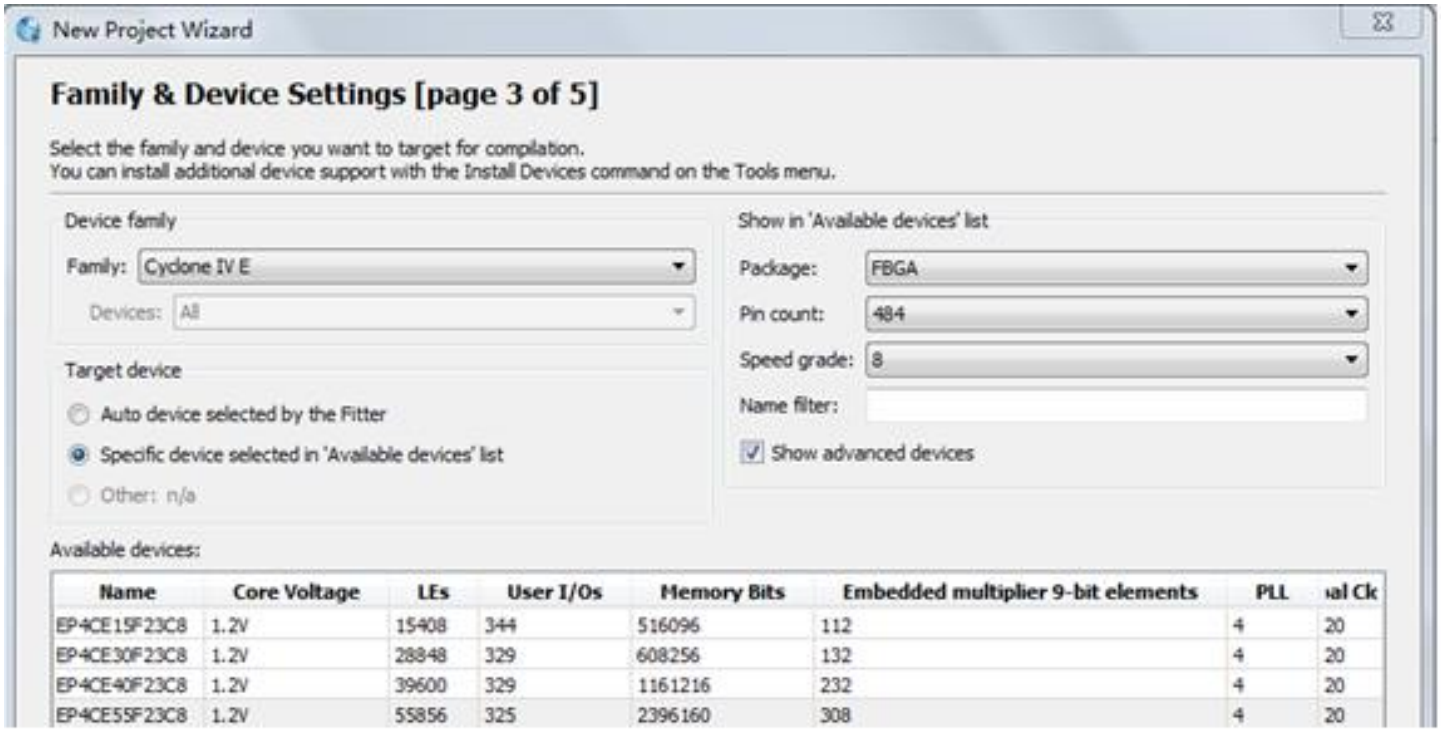

图 4-4　选择目标器件 EP4CE55F23C8

4.1 VHDL程序输入和编译

4.1.2 创建工程

（4）工具设置

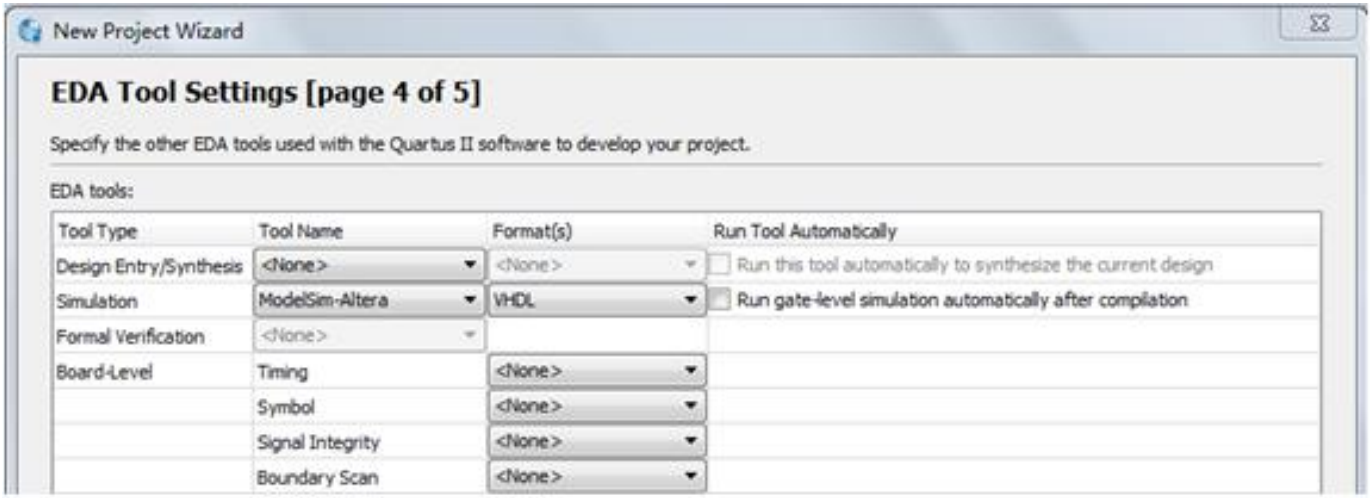

图 4-5　设计与验证工具软件选择

（5）结束设置

4.1 VHDL程序输入和编译

4.1.3 全程编译前约束项目设置

（1）选择编译约束条件

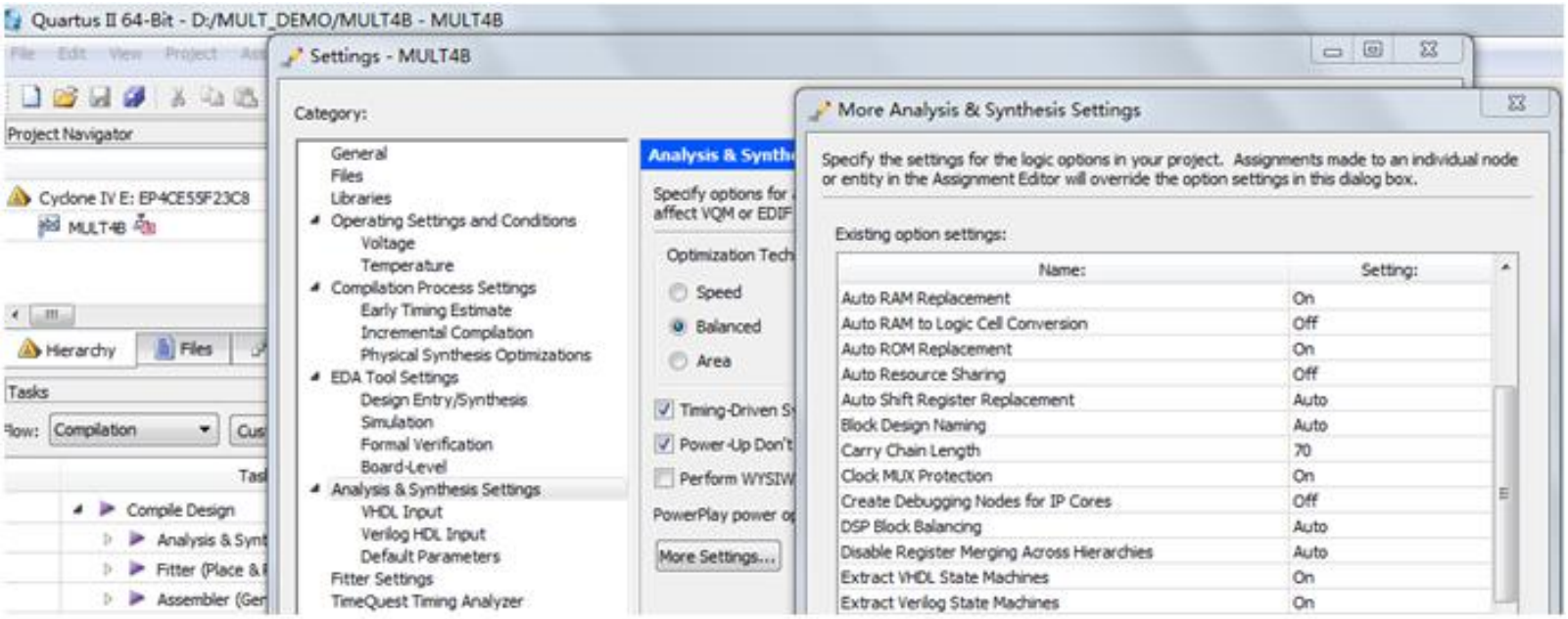

图 4-6　选择编译综合的工作方式

4.1 VHDL程序输入和编译

4.1.3 全程编译前约束项目设置

（2）选择目标芯片的其它控制项

（3）选择配置器件的工作方式

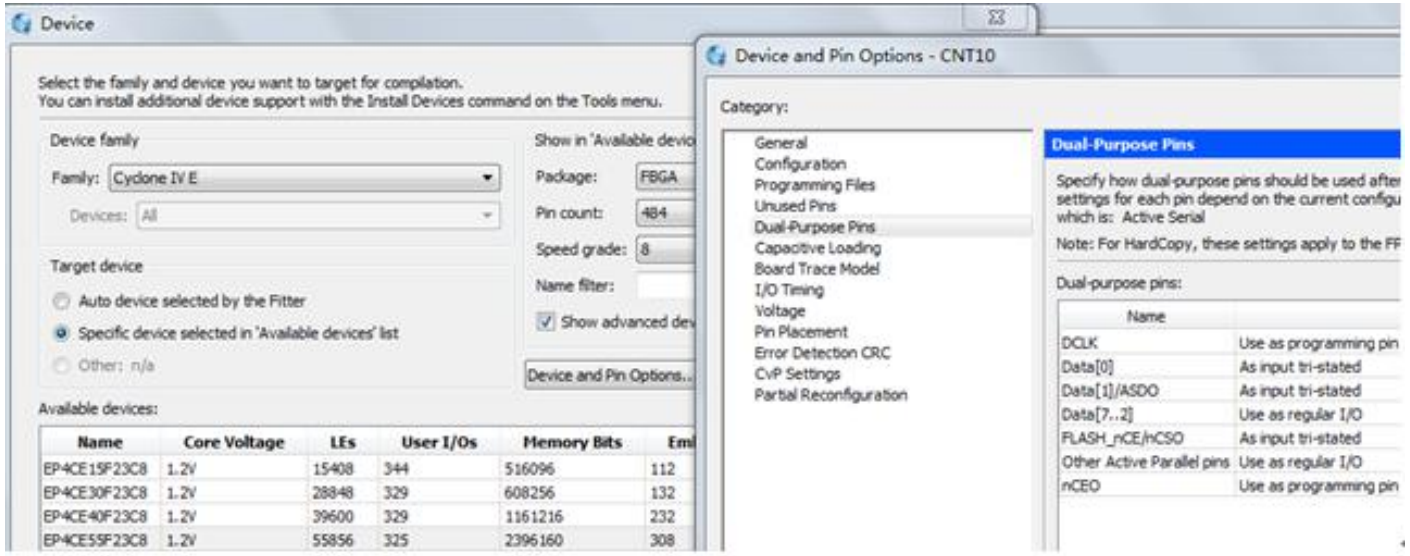

图 4-7　选择目标器件和工作方式

（4）选择目标器件引脚端口状态

4.1 VHDL程序输入和编译

4.1.4 全程综合与编译

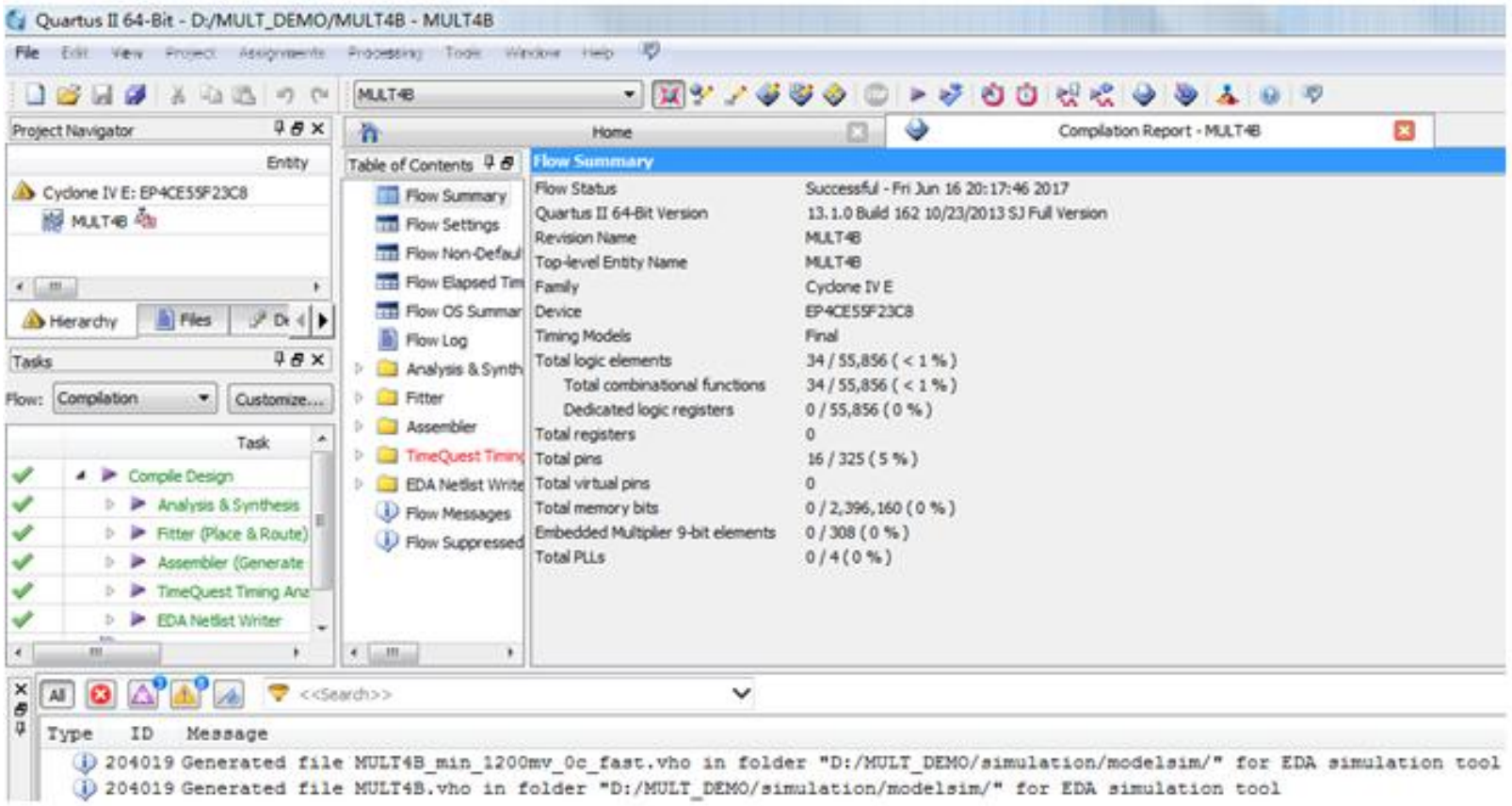

图 4-8　全程编译无错后的报告信息

4.1 VHDL程序输入和编译

4.1.5 RTL图观察器应用

4.2 仿真测试

（1）确认Quartus II中的仿真工具是否指向Modelsim所在路径

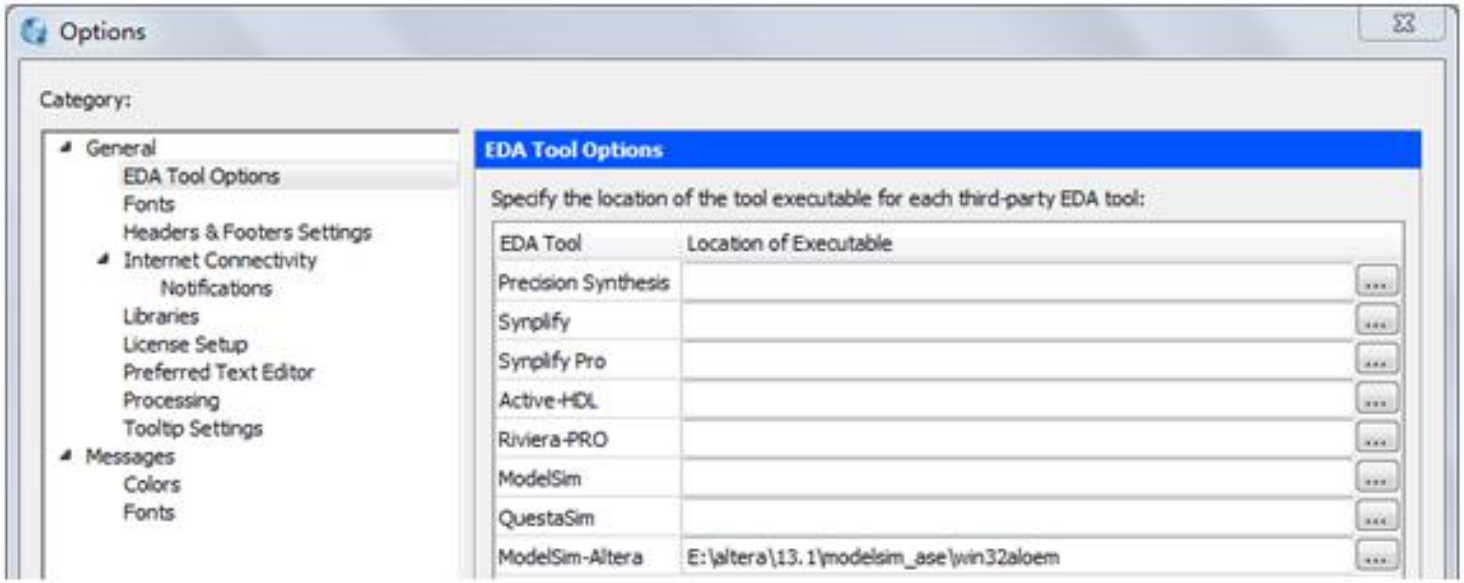

图 4-9 查看 Quartus 仿真工具指向 Modelsim 仿真软件的路径

4.2 仿真测试

（2）打开波形编辑器

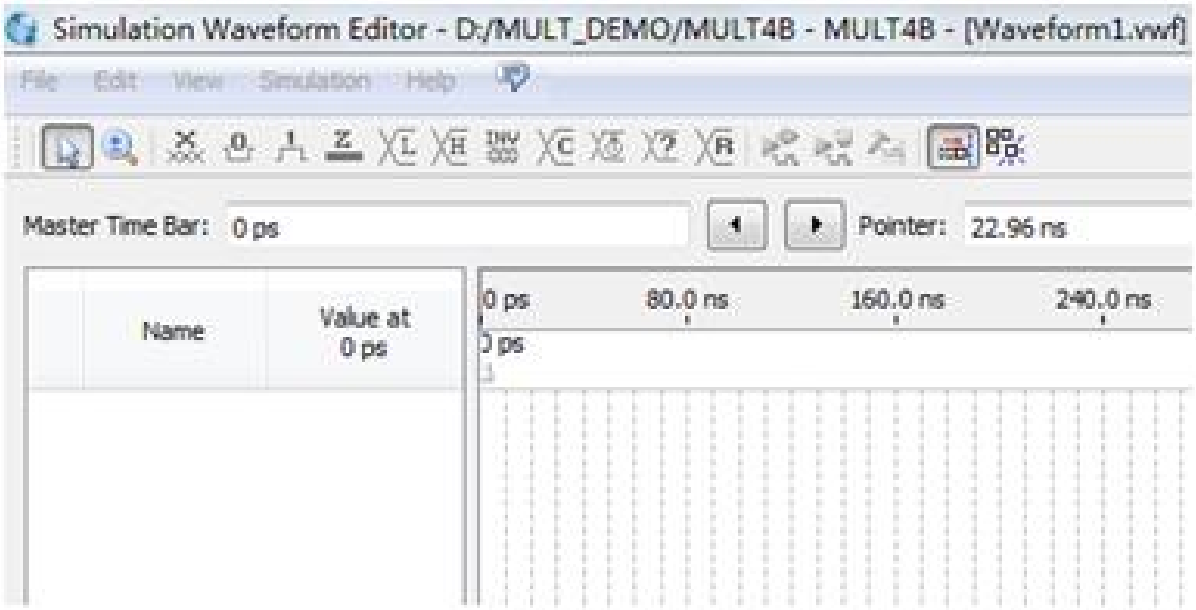

图 4-10 Vector Waveform File 文件编辑窗

4.2 仿真测试

（3）设置仿真时间区域

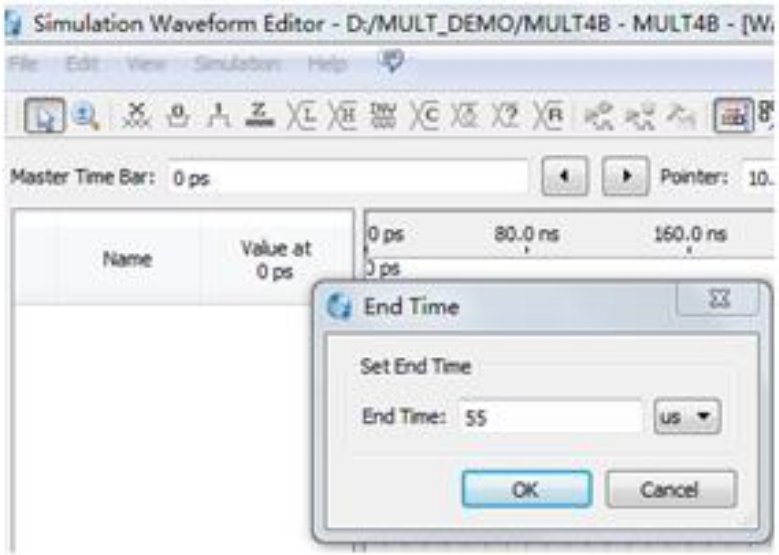

图 4-11 设置仿真时间长度

4.2 仿真测试

（4）波形文件存盘

（5）将工程**MULT4B**的端口信号节点选入波形编辑器中

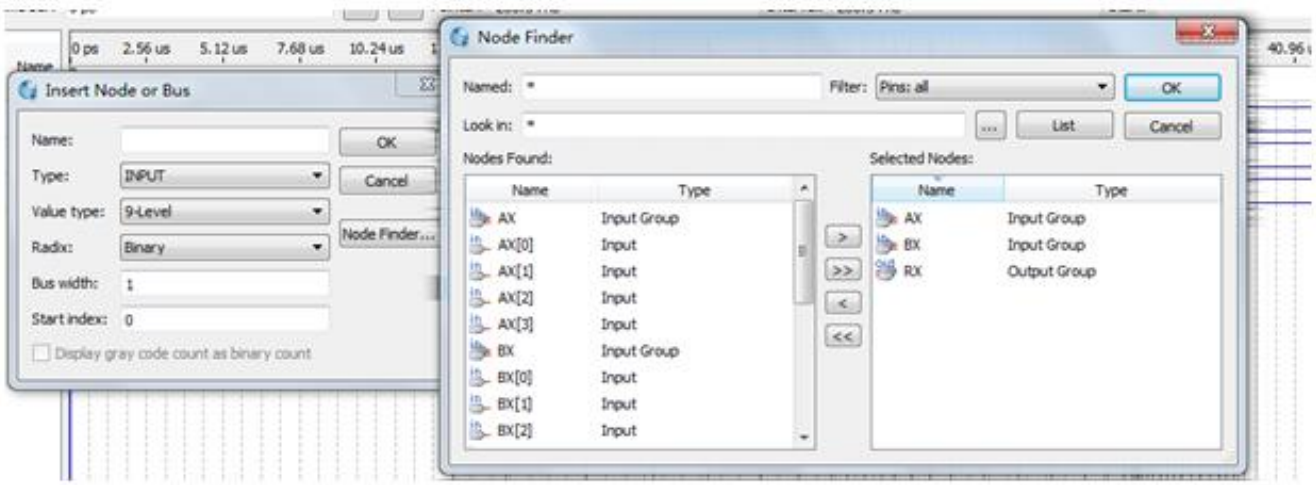

图 4-12 加入仿真需要的信号节点

4.2 仿真测试

（6）设置激励信号波形

图 4-13 设置矢量数据格式

图 4-14 设置输入数据

4.2 仿真测试

（7）图4-15是最后设置好的.vwf仿真激励波形文件图

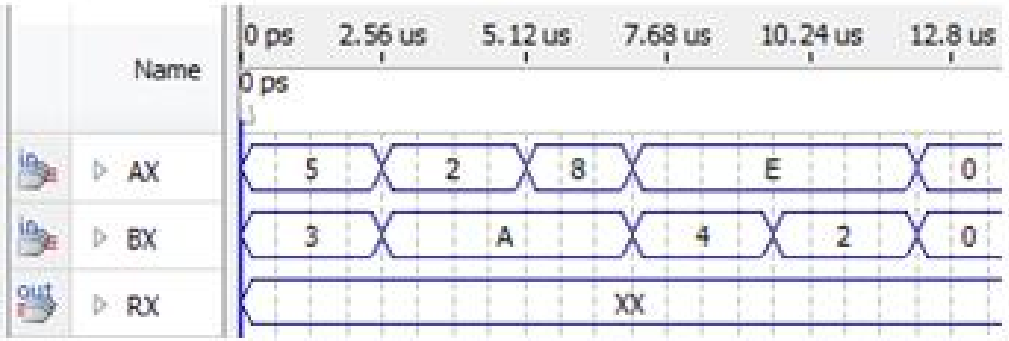

图 4-15 编辑好激励波形

（8）启动仿真器

4.2 仿真测试

（9）观察仿真结果

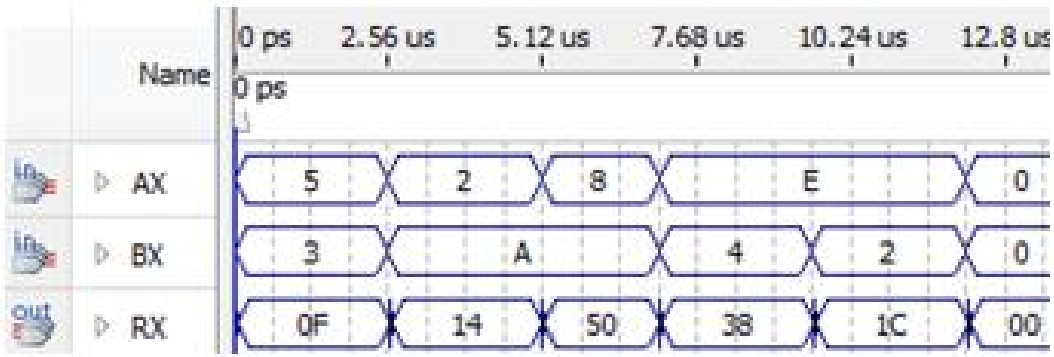

图 4-16 仿真输出的波形文件

4.3 引脚锁定与硬件测试

4.3.1 引脚锁定

表 4-1 基于 EP4CE55F23C8 FPGA 的引脚锁定情况

信号名	AX(3)	AX(2)	AX(1)	AX(0)	BX(3)	BX(2)	BX(1)	BX(0)
数据输入	键 1				键 2			
电路信号	PIO3	PIO2	PIO1	PIO0	PIO7	PIO6	PIO5	PIO4
FPGA 引脚	Y1	V1	R1	N1	AB6	Y7	AA6	AB3
信号名	RX(7)	RX(6)	RX(5)	RX(4)	RX(3)	RX(2)	RX(1)	RX(0)
数码显示	数码 6				数码 5			
电路信号	PIO23	PIO22	PIO21	PIO20	PIO19	PIO18	PIO17	PIO16
FPGA 引脚	AA4	AA5	Y2	AA1	V2	W1	R2	U1

4.3 引脚锁定与硬件测试

4.3.1 引脚锁定

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location
AX[3]	Input				PIN_V1
AX[2]	Input				PIN_Y2
AX[1]	Input				PIN_P4
AX[0]	Input				PIN_M5
BX[3]	Input				PIN_P3
BX[2]	Input				PIN_P5
BX[1]	Input				PIN_P2
BX[0]	Input				PIN_P1
RX[7]	Output				PIN_R3
RX[6]	Output				PIN_R4
RX[5]	Output				PIN_U1
RX[4]	Output				PIN_U2
RX[3]	Output				PIN_V2
RX[2]	Output				PIN_N5
RX[1]	Output				PIN_R1
RX[0]	Output				PIN_R2

图 4-17 编译完成后刚打开的 Pin Planner 窗

4.3 引脚锁定与硬件测试

4.3.1 引脚锁定

Node Name	Direction	Location
AX[3]	Input	PIN_Y1
AX[2]	Input	PIN_V1
AX[1]	Input	PIN_R1
AX[0]	Input	PIN_N1
BX[3]	Input	PIN_AB6
BX[2]	Input	PIN_Y7
BX[1]	Input	PIN_AA6
BX[0]	Input	PIN_AB3
RX[7]	Output	PIN_AA4
RX[6]	Output	PIN_AA5
RX[5]	Output	PIN_Y2
RX[4]	Output	PIN_AA1
RX[3]	Output	PIN_V2
RX[2]	Output	PIN_W1
RX[1]	Output	PIN_R2
RX[0]	Output	PIN_U1

图 4-18 引脚锁定完成后的情况

4.3 引脚锁定与硬件测试

4.3.2 编译文件下载

（1）打开编程窗和配置文件

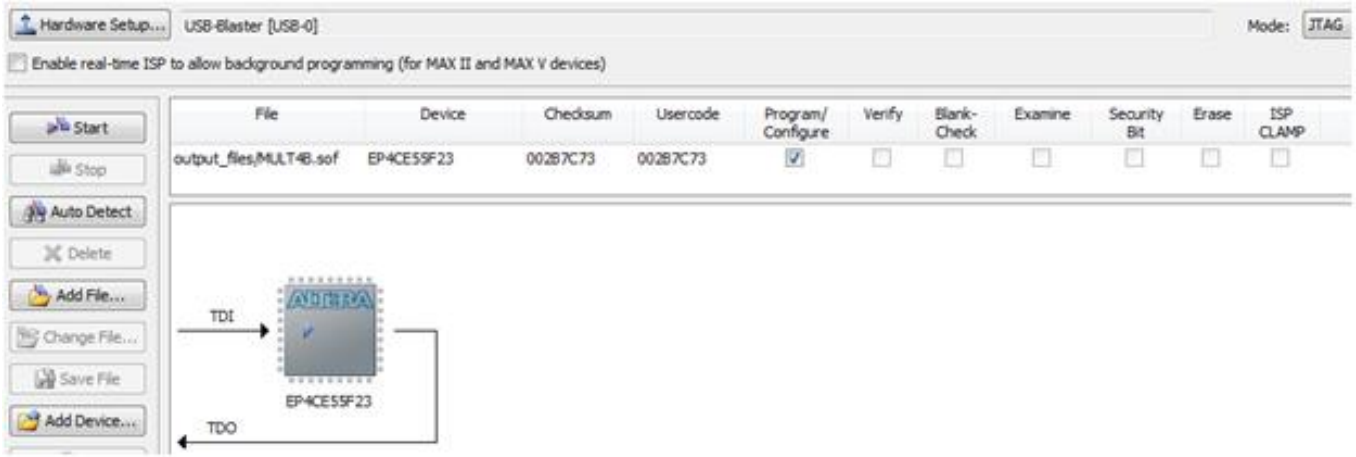

图 4-19 选择 JTAG 编程模式，将 SOF 文件载入 FPGA

4.3 引脚锁定与硬件测试

4.3.2 编译文件下载

（2）设置编程器

图 4-20 加入编程下载方式

（3）硬件测试

4.3 引脚锁定与硬件测试

4.3.3 JTAG间接编程模式

1. 将SOF文件转化为JTAG间接配置文件

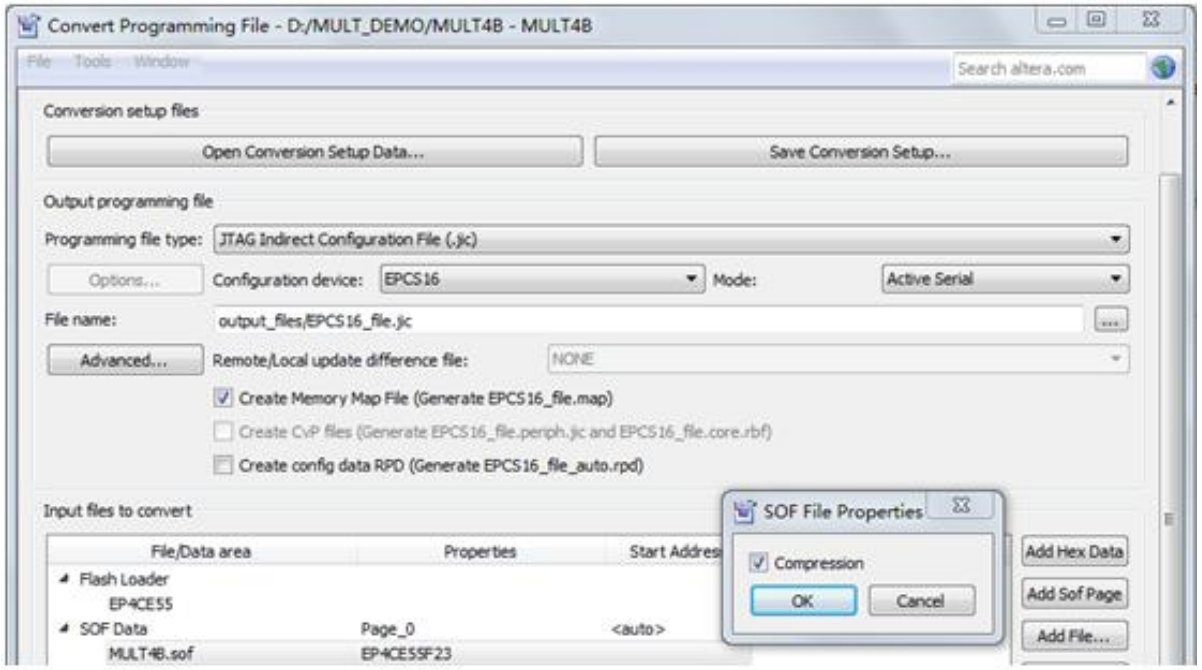

图 4-21 设定 JTAG 间接编程文件

4.3 引脚锁定与硬件测试

4.3.3 JTAG间接编程模式

2. 下载JTAG间接配置文件

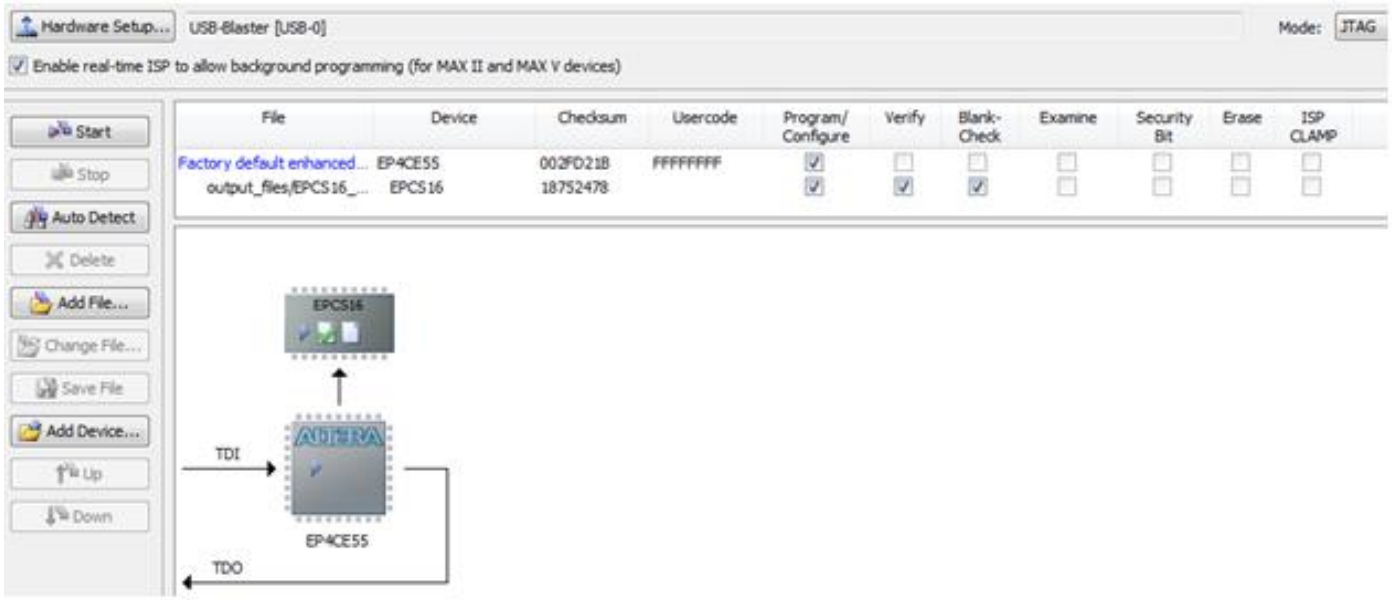

图 4-22 用 JTAG 模式将间接配置文件烧入配置器件 EPCS16 中

4.3 引脚锁定与硬件测试

4.3.4 USB-Blaster驱动程序安装方法

4.4 电路原理图设计流程

1．用原理图设计一个半加器

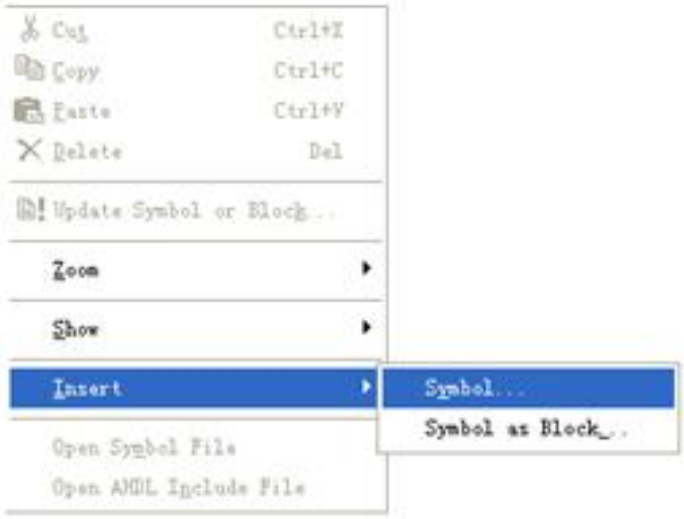

图 4-23 选择打开元件输入窗

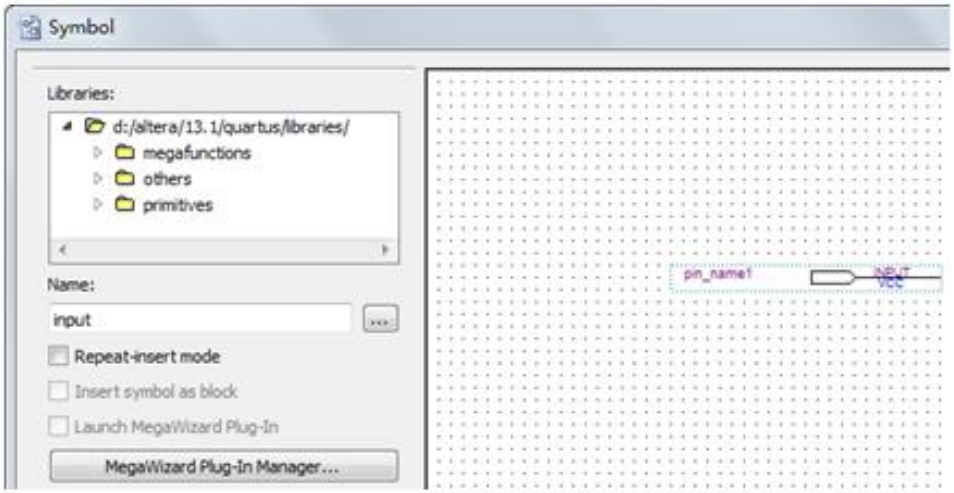

图 4-24 在元件输入对话框输入引脚

4.4 电路原理图设计流程

1. 用原理图设计一个半加器

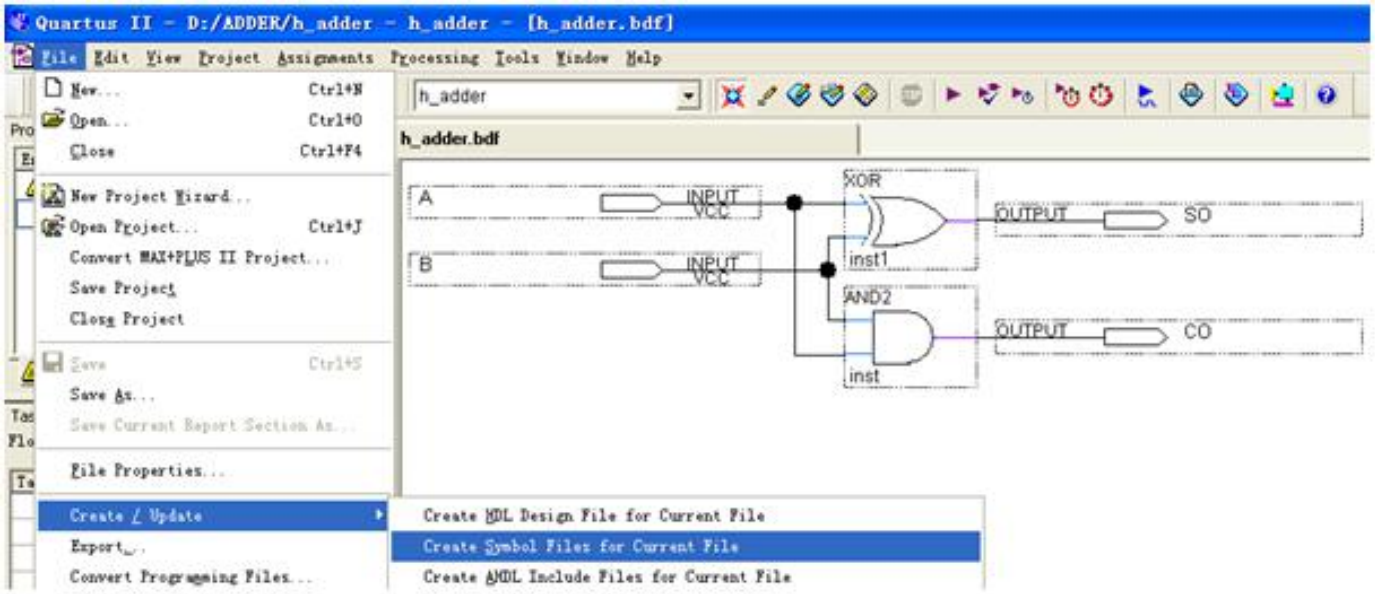

图 4-25 完成设计并将半加器封装成一个元件，以便在更高层设计中调用

4.4 电路原理图设计流程

2. 设计全加器顶层文件

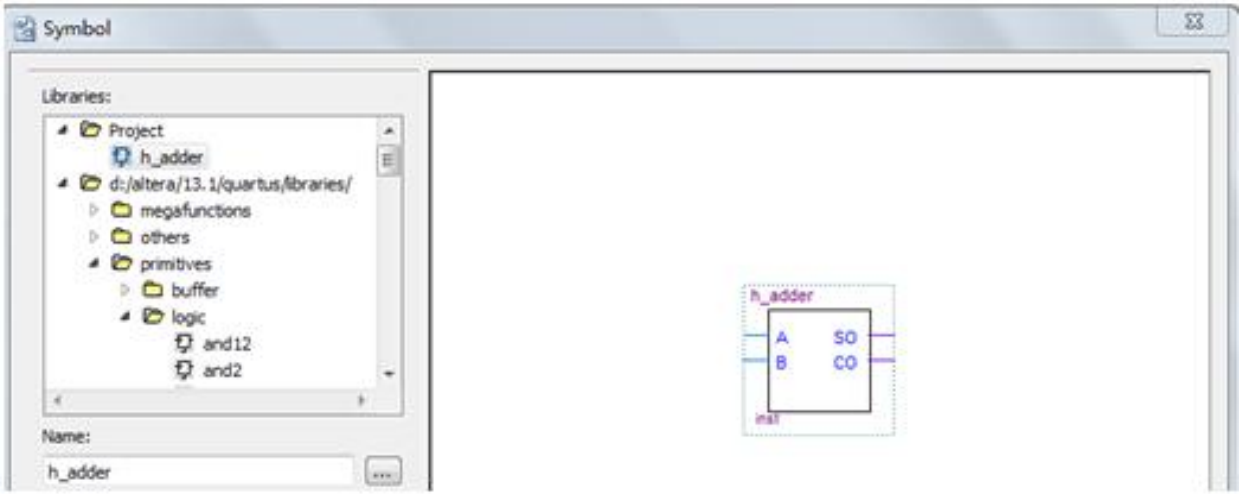

图 4-26 在 f_adder 工程下加入半加器原件

4.4 电路原理图设计流程

3．对设计项目进行时序仿真

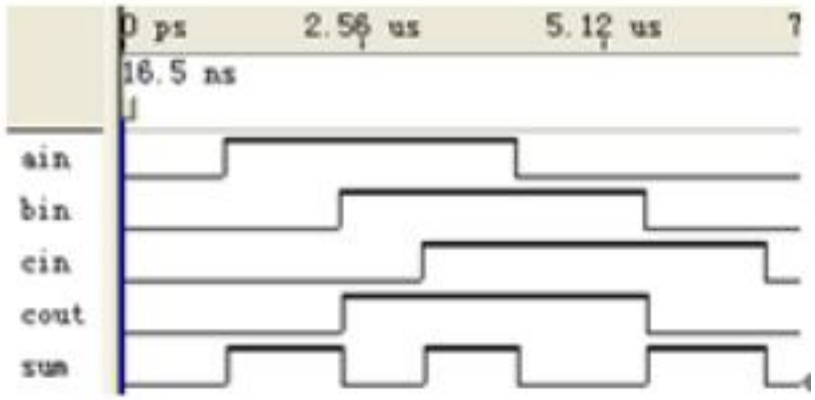

图 4-27 全加器的仿真波形

4.5 HDL版本设置及Analysis & Synthesis功能

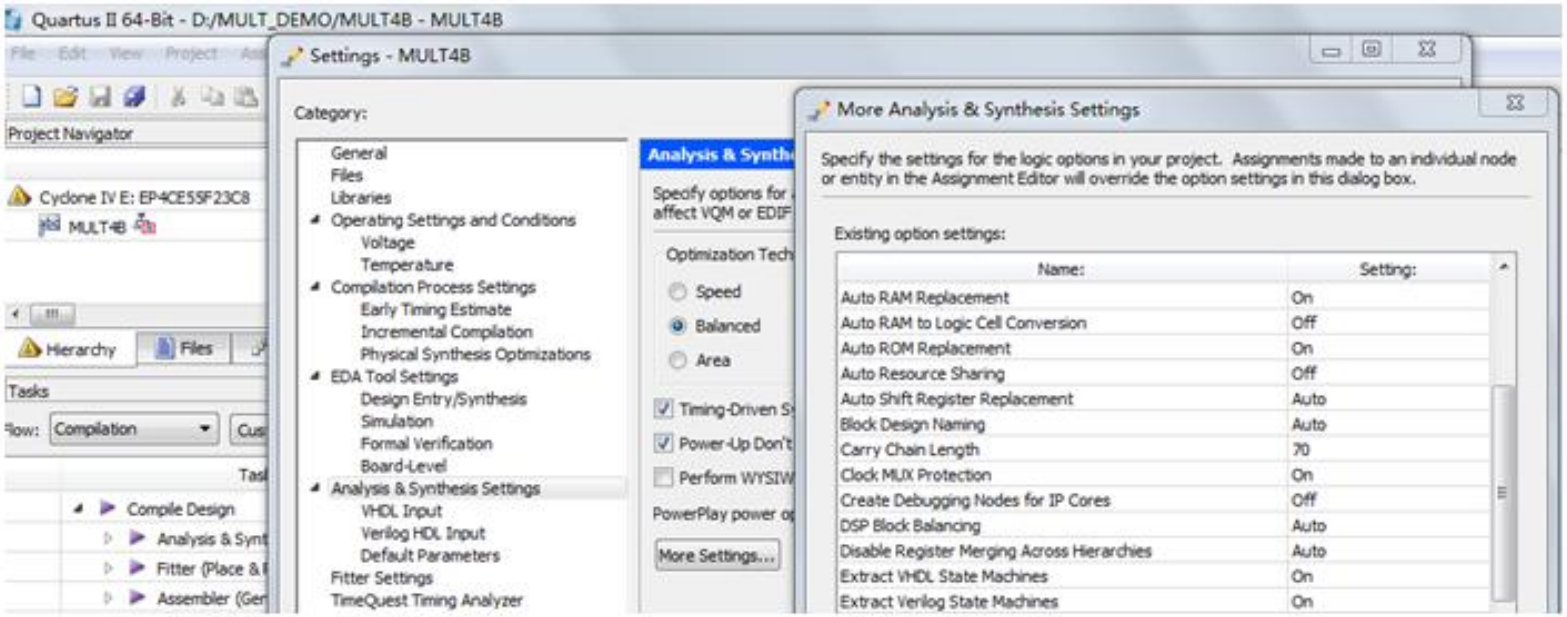

图 4-6　选择编译综合的工作方式

4.6 利用属性表述实现引脚锁定

【例 4-2】

```
ARCHITECTURE ONE OF MULT4B IS
  attribute chip_pin : string;  -- chip_pin被定义为字符串数据类型 string
  attribute chip_pin of AX : signal is "Y1,V1,R1,N1" ;
  attribute chip_pin of BX : signal is "AB6,Y7,AA6,AB3 ";
  attribute chip_pin of RX : signal is "AA4,AA5,Y2,AA1,V2,W1,R2,U1" ;
  ...
```

```
attribute chip_pin of EN : signal is "B6";
```

4.7 keep属性应用

【例 4-3】其它部分内容与例 3-9 相同

```
ARCHITECTURE fd1 OF f_adder IS
  . . .
SIGNAL net1,net2,net3 :  STD_LOGIC;
   attribute keep : boolean;      --由于“true”的类型是布尔类型boolean
   attribute keep  of net3 : signal is true;
BEGIN
```

4.7 keep属性应用

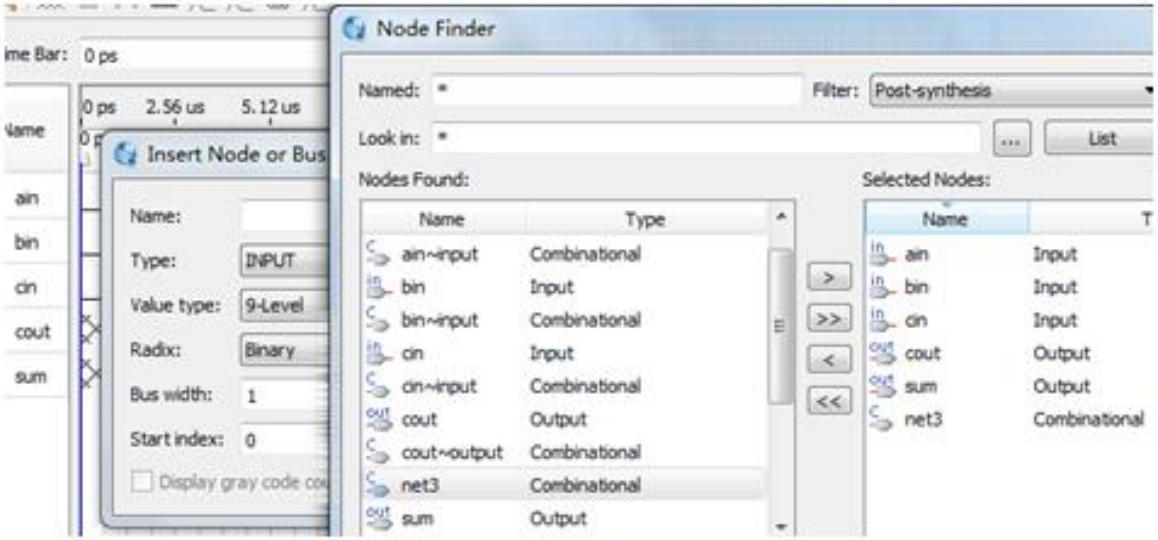

图 4-28　向仿真激励信号波形编辑窗调入信号 net3

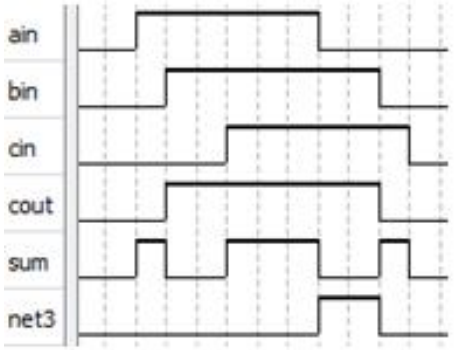

图 4-29 仿真波形

4.8 SignalProbe使用方法

1．按常规流程完成设计仿真和硬件测试

2．设置SignalProbe Pins

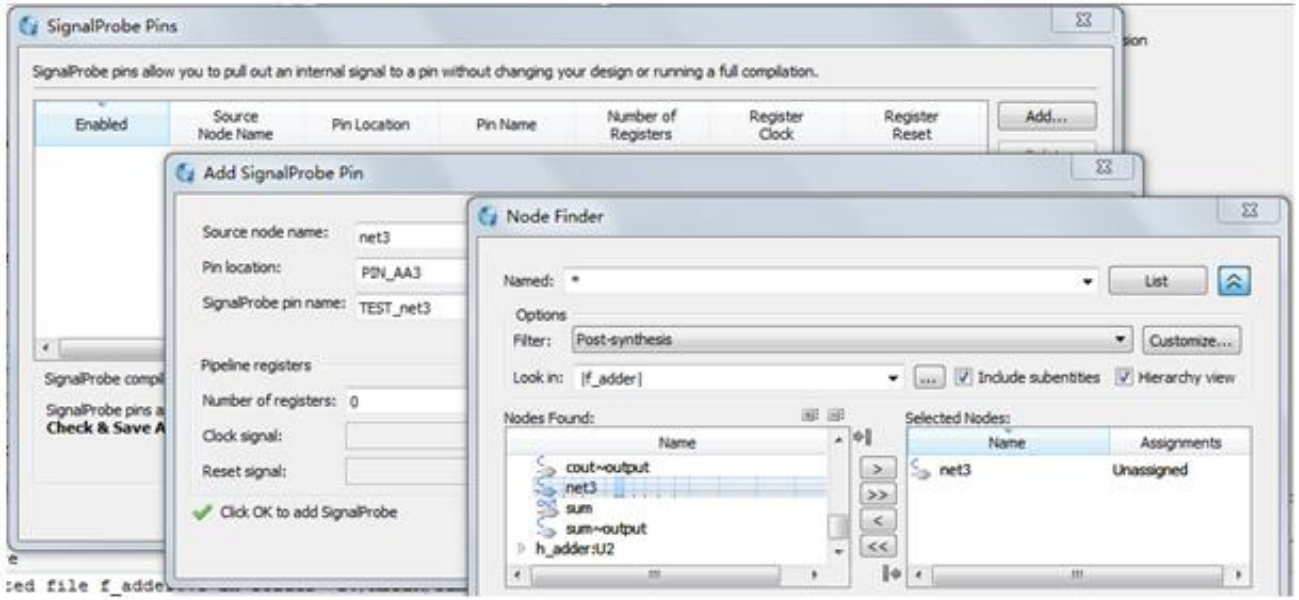

图 4-30 在 SignalProbe 对话框设置探测信号 net3

4.8 SignalProbe使用方法

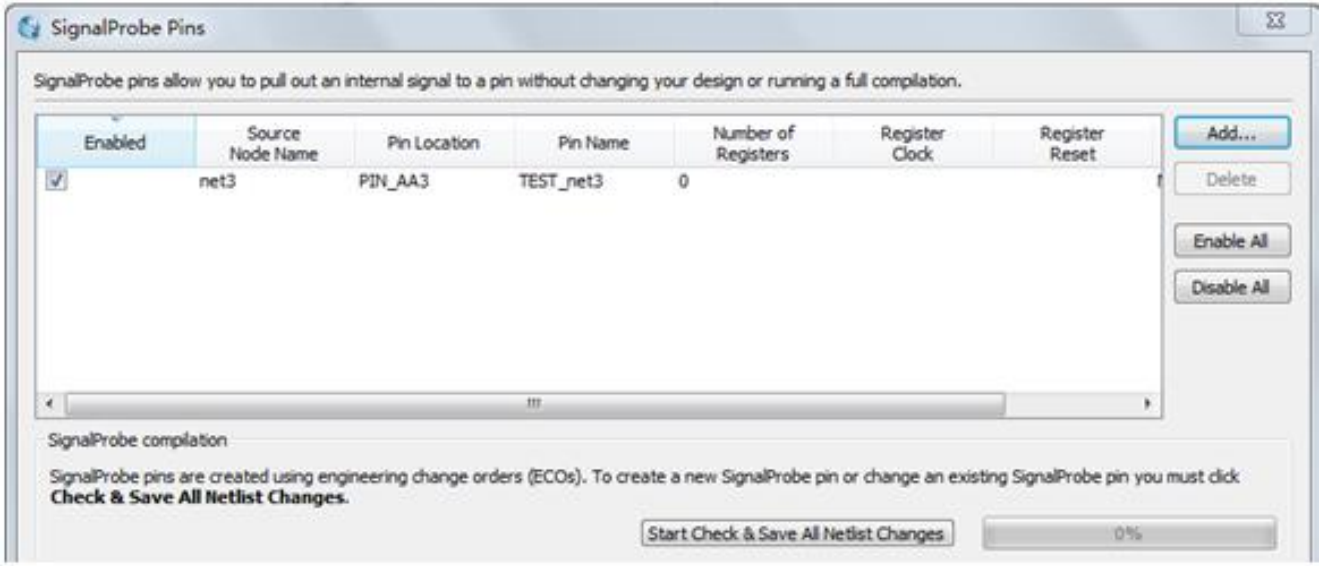

图 4-31　SignalProbe Pins 对话框设置情况

3．编译SignalProbe Pins测试信息并下载测试

实验与设计

4-1 多路选择器设计实验明

4-2 8位加法器设计实验

4-3 8位硬件乘法器设计实验

实验与设计

4-4 十六进制7段数码显示译码器设计

表 4-2 7 段译码器真值表

输入码	输出码	代表数据
0000	0111111	0
0001	0000110	1
0010	1011011	2
0011	1001111	3
0100	1100110	4
0101	1101101	5
0110	1111101	6
0111	0000111	7
1000	1111111	8
1001	1101111	9
1010	1110111	A
1011	1111100	B
1100	0111001	C
1101	1011110	D
1110	1111001	E
1111	1110001	F

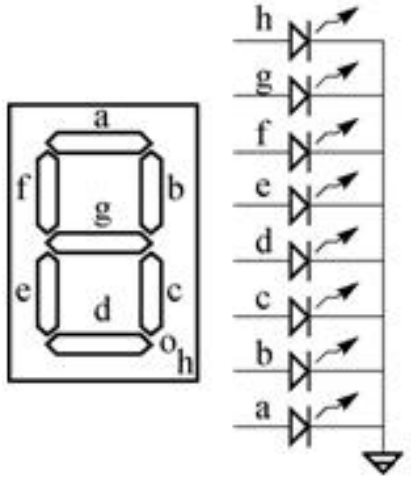

图 4-3 共阴数码管

【例 4-4】

```
LIBRARY IEEE ;
 USE IEEE.STD_LOGIC_1164.ALL ;
 ENTITY DECL7S IS
  PORT ( A   : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
      LED7S : OUT STD_LOGIC_VECTOR(6 DOWNTO 0)  ) ;
 END ;
 ARCHITECTURE one OF DECL7S IS
 BEGIN
  PROCESS( A )
  BEGIN
    CASE  A  IS
       WHEN "0000" =>  LED7S <= "0111111" ;
       WHEN "0001" =>  LED7S <= "0000110" ;
       WHEN "0010" =>  LED7S <= "1011011" ;
       WHEN "0011" =>  LED7S <= "1001111" ;
       WHEN "0100" =>  LED7S <= "1100110" ;
       WHEN "0101" =>  LED7S <= "1101101" ;
       WHEN "0110" =>  LED7S <= "1111101" ;
       WHEN "0111" =>  LED7S <= "0000111" ;
       WHEN "1000" =>  LED7S <= "1111111" ;
       WHEN "1001" =>  LED7S <= "1101111" ;
       WHEN "1010" =>  LED7S <= "1110111" ;
       WHEN "1011" =>  LED7S <= "1111100" ;
       WHEN "1100" =>  LED7S <= "0111001" ;
       WHEN "1101" =>  LED7S <= "1011110" ;
       WHEN "1110" =>  LED7S <= "1111001" ;
       WHEN "1111" =>  LED7S <= "1110001" ;
       WHEN OTHERS =>  NULL ;
   END CASE ;
  END PROCESS ;

END ;
```