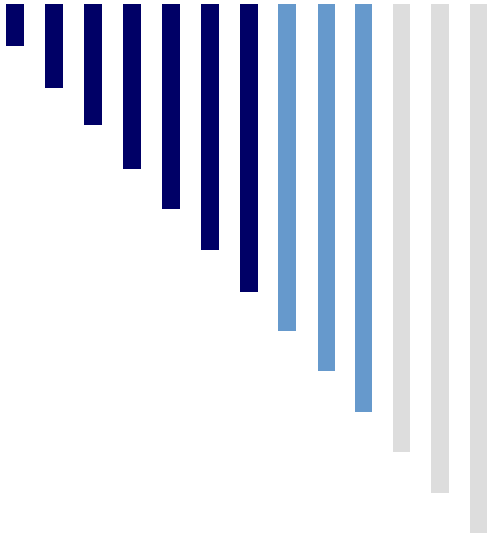


第3章

Quartus II应用向导

3.1 QuartusII应用一般流程

3.1.1 输入设计程序

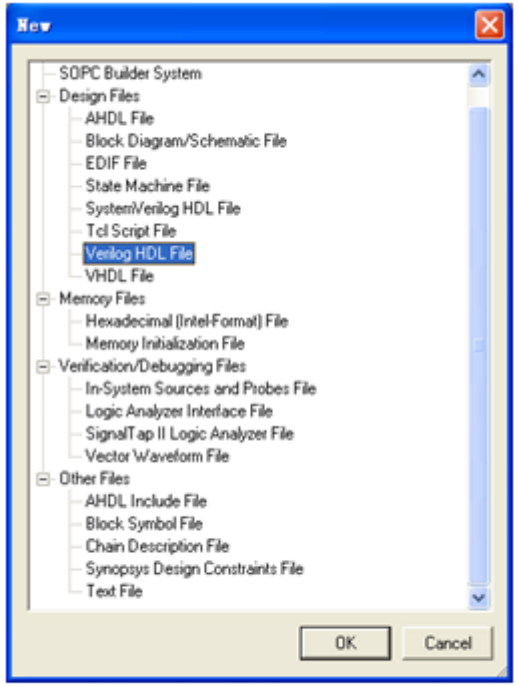


图 3-1　选择编辑文件类型

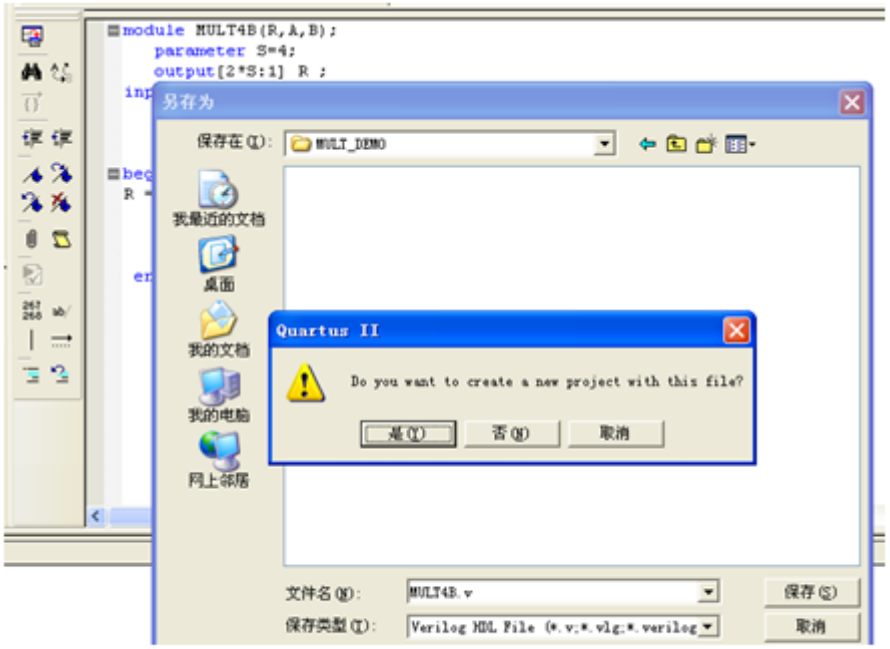


图 3-2　编辑输入源程序并存盘

3.1 QuartusII应用一般流程

3.1.2 创建本项目设计工程

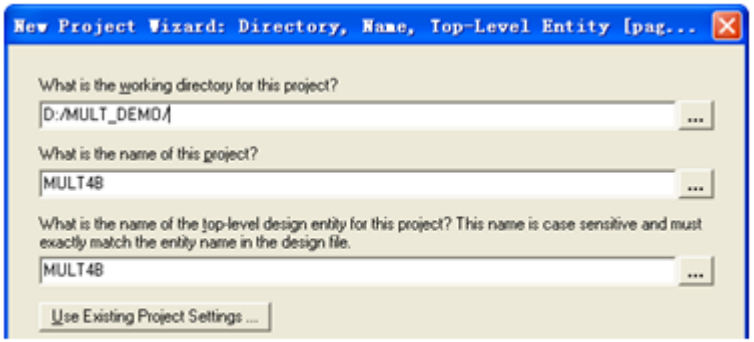


图 3-3　利用 New Project Wizard 创建工程 MULT4B

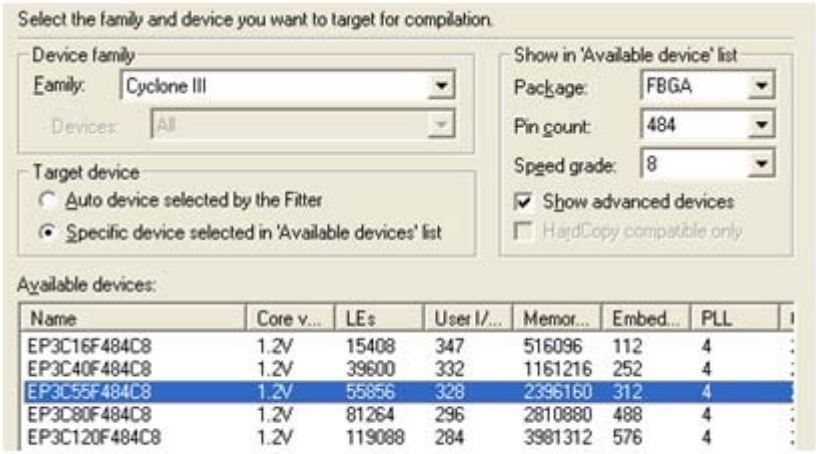


图 3-4　选择目标器件 EP3C55F484C8

3.1 QuartusII应用一般流程

3.1.3 设置约束项目

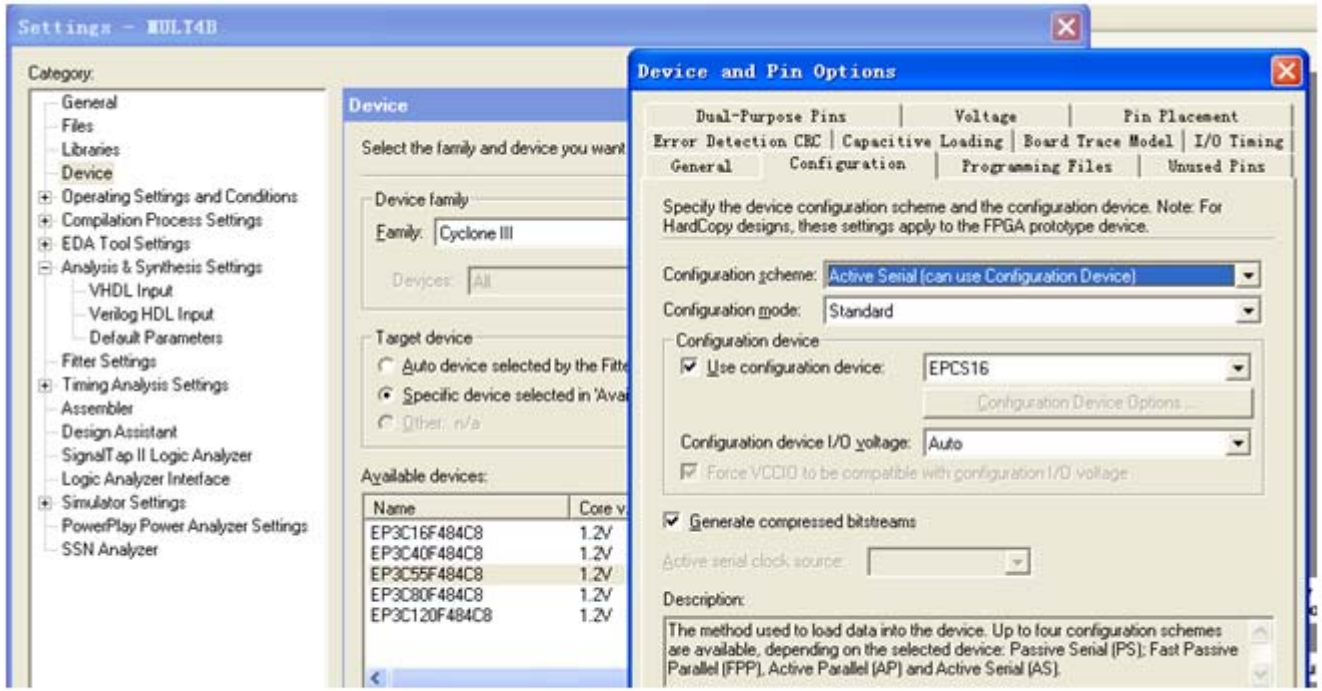


图 3-5　选择配置器件的工作方式

3.1 QuartusII应用一般流程

3.1.4 全程编译与逻辑综合

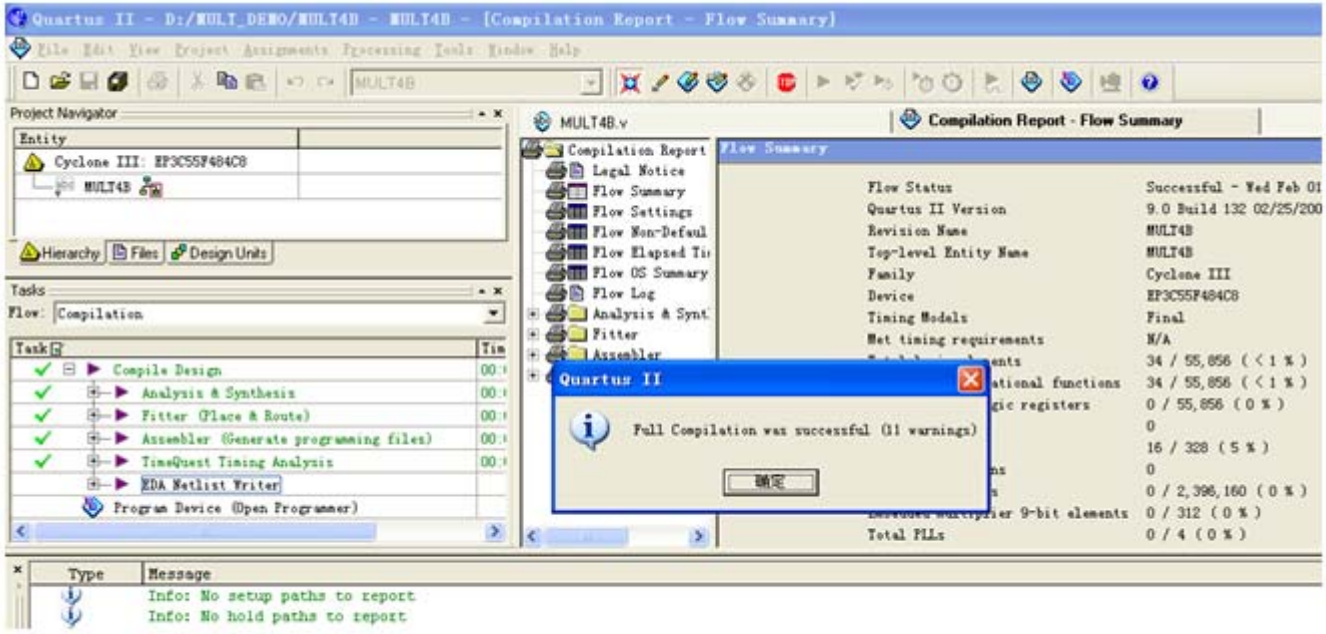


图 3-6 全程编译无错后的报告信息

3.1 QuartusII应用一般流程

3.1.5 测试设计项目

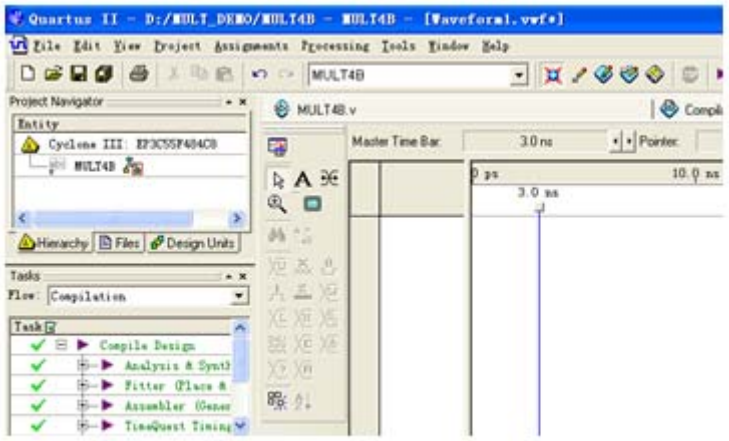


图 3-7 激励信号波形编辑器窗口

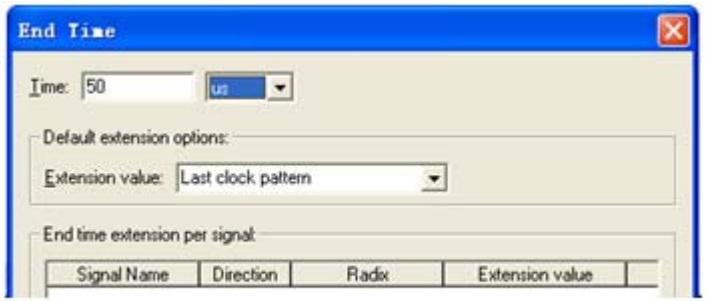


图 3-8 设置仿真时间长度

3.1 QuartusII应用一般流程

3.1.5 测试设计项目

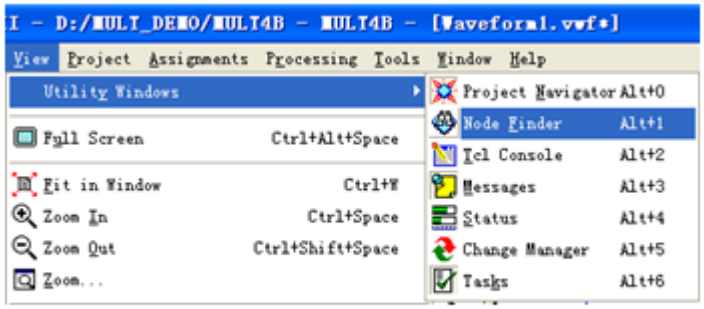


图 3-9 打开信号节点查询窗口

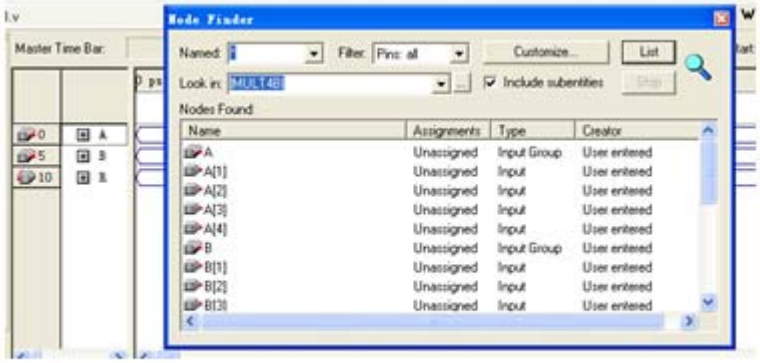


图 3-10 向波形编辑器拖入信号节点

3.1 QuartusII应用一般流程

3.1.5 测试设计项目

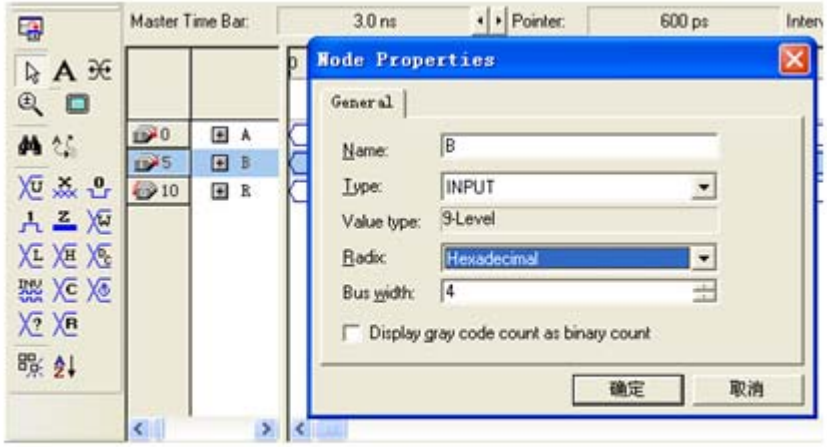


图 3-11 设置好的激励波形图，及选择总线数据格式

3.1 QuartusII应用一般流程

3.1.5 测试设计项目

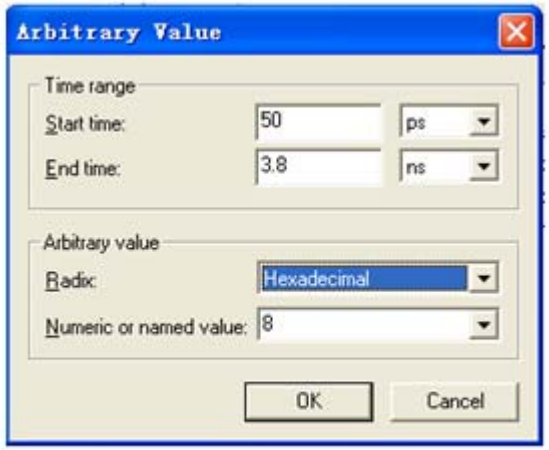


图 3-12 设置输入数据

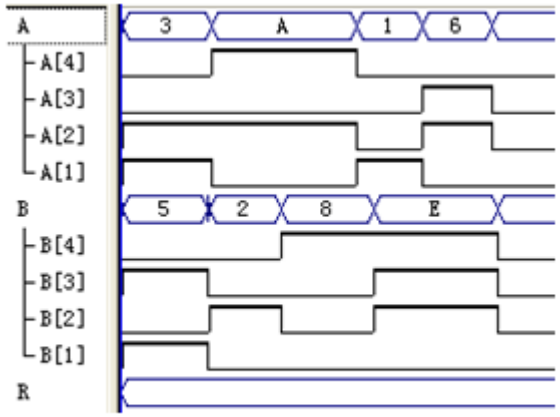


图 3-13 编辑好的 vwf 文件

3.1 QuartusII应用一般流程

3.1.5 测试设计项目

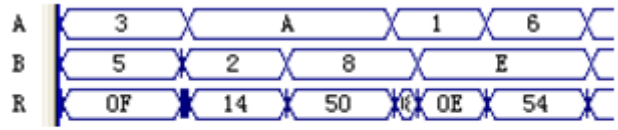


图 3-14 仿真波形输出(十六进制类型)

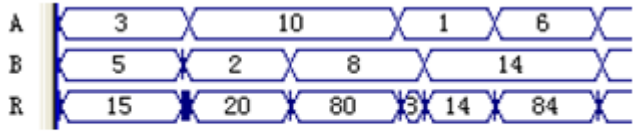


图 3-15 仿真波形输出(十进制类型)

3.1.6 RTL图观察器应用

3.2 硬件功能验证及FPGA开发

3.2.1 引脚锁定

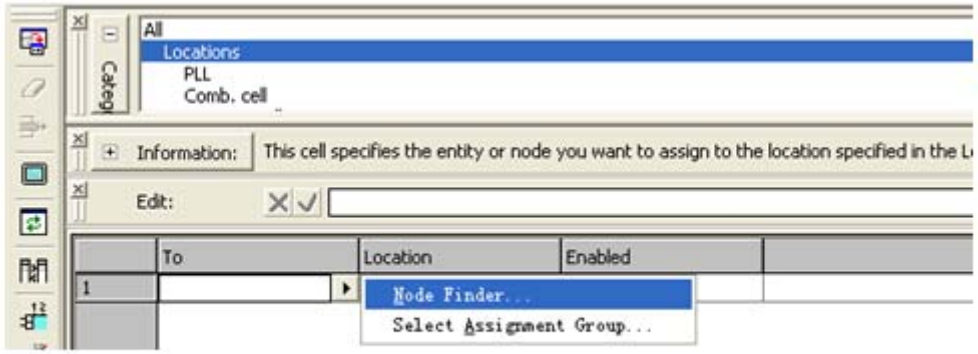


图 3-16 利用 Assignment Editor 编辑器锁定 FPGA 引脚

3.2 硬件功能验证及FPGA开发

3.2.1 引脚锁定

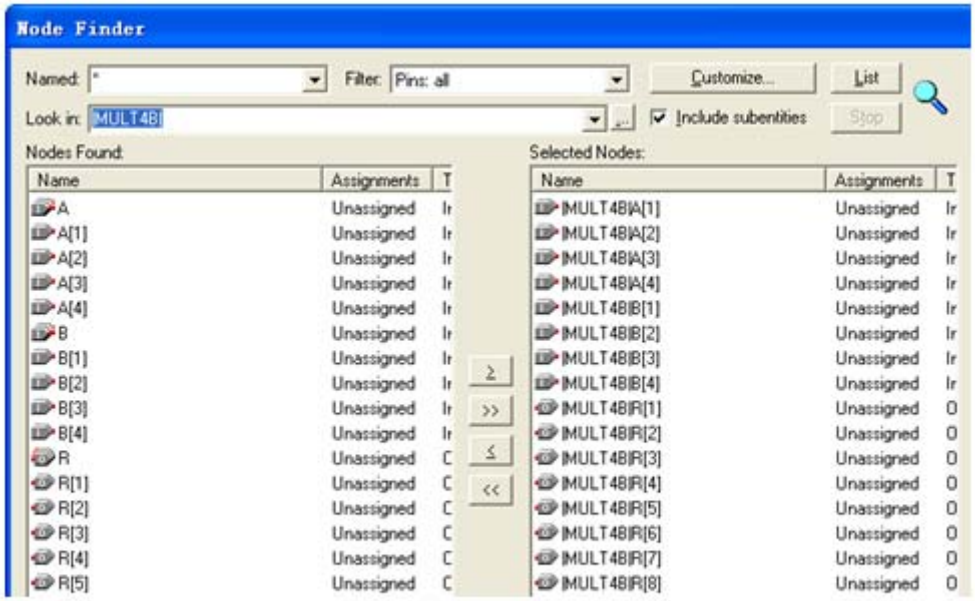


图 3-17 利用 Node Finder 工具选择需要锁定引脚的信号

图 3-18 引脚锁定窗口

3.2 硬件功能验证及FPGA开发

3.2.2 编译文件下载

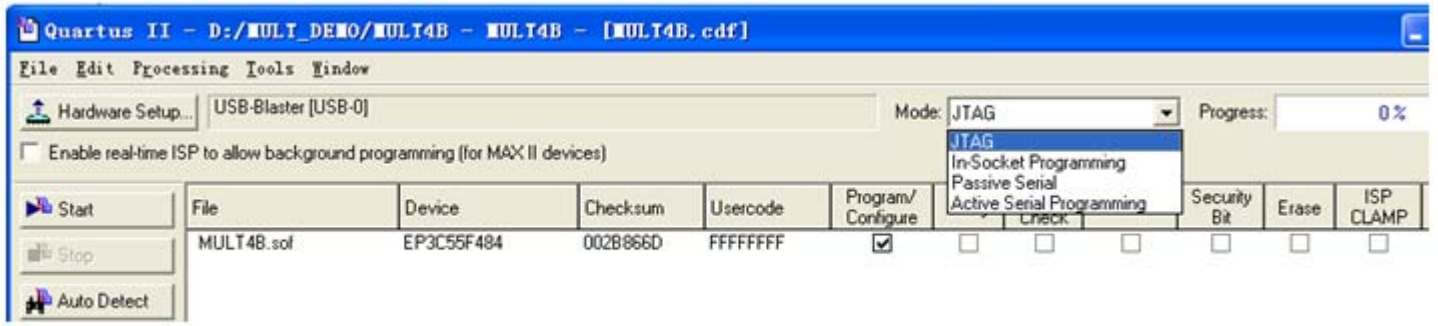


图 3-19 选择JTAG编程模式

3.2 硬件功能验证及FPGA开发

3.2.2 编译文件下载

图 3-20 加入编程下载方式

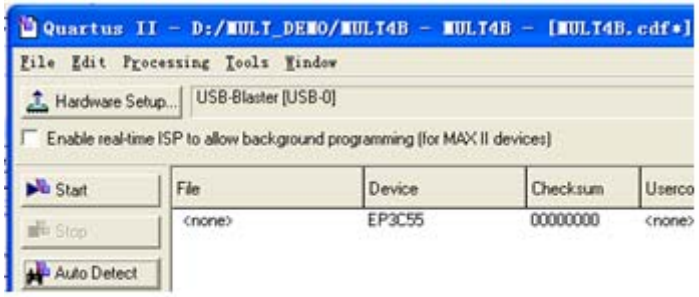


图 3-21 检测实验板上的 FPGA 器件

3.2 硬件功能验证及FPGA开发

3.2.3 AS直接编程模式

3.2.4 JTAG间接编程模式

1. 将SOF文件转化为JTAG间接配置文件。

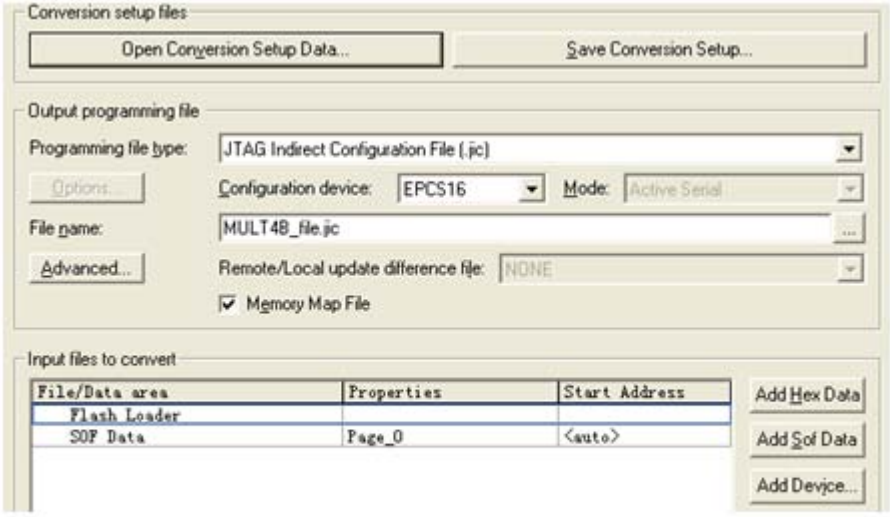


图 3-22 设定 JTAG 间接编程文件

3.2 硬件功能验证及FPGA开发

3.2.4 JTAG间接编程模式

1. 将SOF文件转化为JTAG间接配置文件。

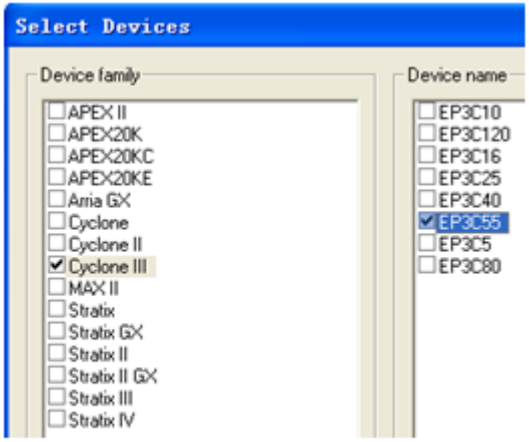


图 3-23 选择目标器件 EP3C55

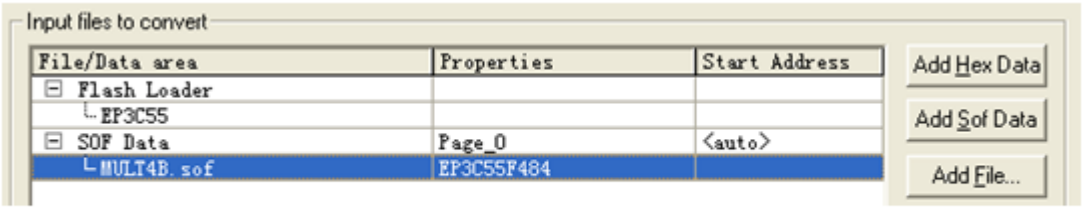


图 3-24 加入 SOF 文件

3.2 硬件功能验证及FPGA开发

3.2.4 JTAG间接编程模式

2. 下载**JTAG**间接配置文件。

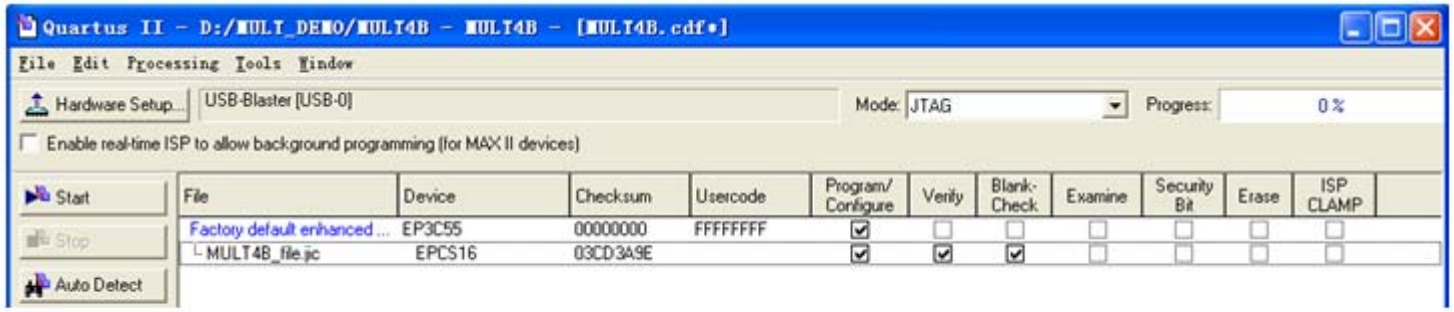


图 3-25 用 JTAG 模式对配置器件 EPCS16 进行编程

3.2.5 USB-Blaster编程配置器件使用方法

3.3 电路原理图设计流程

1. 为本项工程设计建立文件夹

2. 建立原理图文件工程和仿真

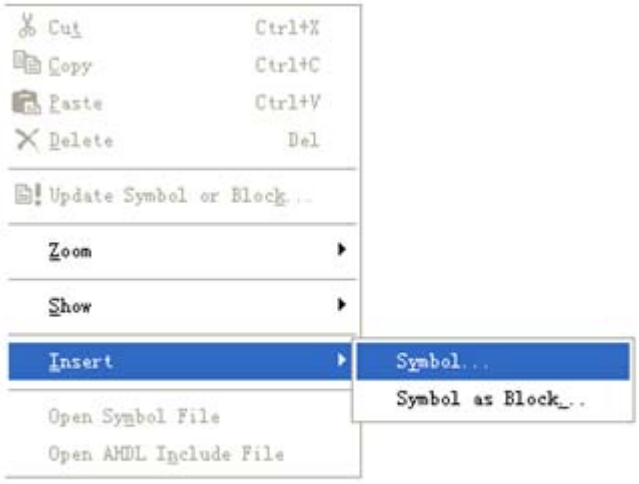


图 3-26 选择打开元件输入窗

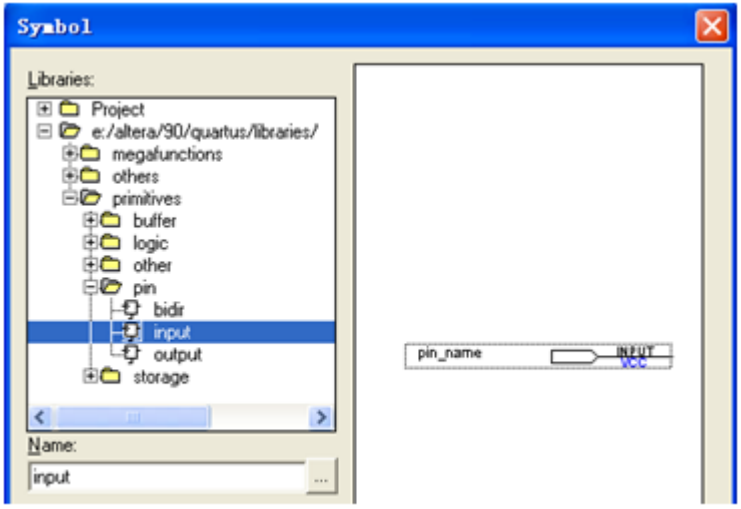


图 3-27 在元件输入对话框输入引脚

3.3 电路原理图设计流程

2. 建立原理图文件工程和仿真

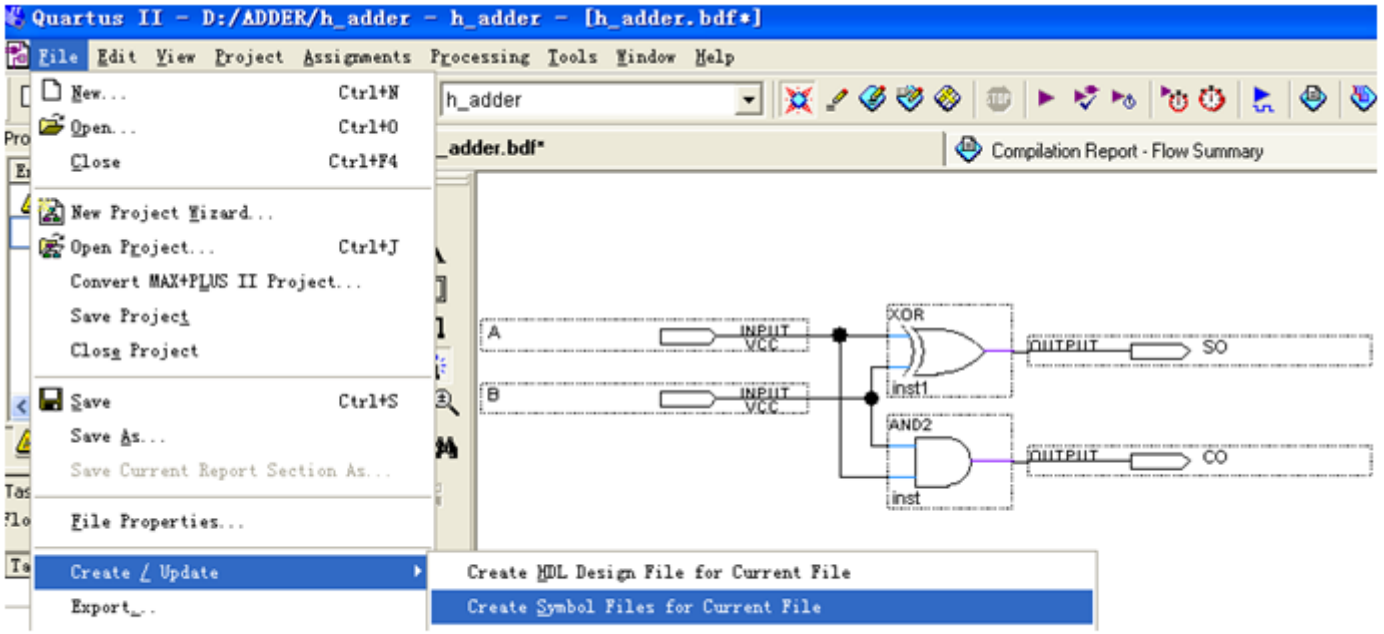


图 3-28 将半加器封装成一个元件，以便在更高层设计中调用

3.3 电路原理图设计流程

3. 将设计项目设置成可调用的元件

4. 设计全加器顶层文件

图 3-29 全加器 f_adder.bdf 工程设置

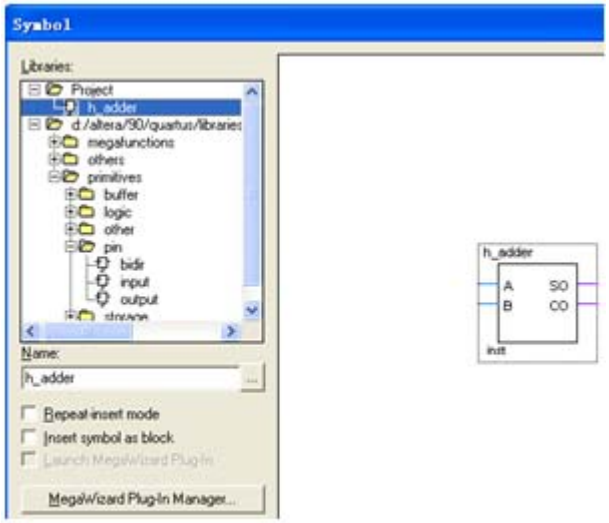


图 3-30 在 f_adder 工程下加入半加器

3.3 电路原理图设计流程

5. 对设计项目进行时序仿真

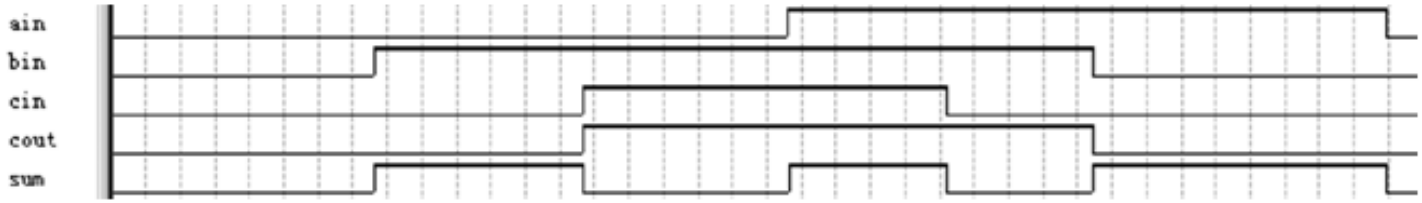


图 3-31 全加器工程 f_adder 的仿真波形

6. 硬件测试

3.4 HDL版本设置及Analysis & Synthesis功能

作为**Quartus II**的编译模块之一，**Settings**对话框的**Analysis & Synthesis**包括**QuartusII Integrated Synthesis**集成综合器，完全支持**VHDL**和**Verilog HDL**，并提供控制综合过程的（约束）选项。

当建立数据库时，**Analysis & Synthesis**的分析阶段将检查工程的逻辑完整性和一致性，并检查边界连接和语法错误。

Analysis & Synthesis 还能使用多种算法来减少逻辑资源的耗用，删除冗余逻辑以及尽可能有效地利用器件体系结构，优化逻辑资源的利用率。

3.5 利用属性表述实现引脚锁定

【例 3-1】此例硬件测试对应基于 EP3C55 的 55F+系统

```
module MULT4B(R,A,B);
  input  [4:1] A  /* synthesis chip_pin="B6,A6,B7,A7" */;
  input  [4:1] B  /* synthesis chip_pin="K21,H22,J21,A5" */;
  output [8:1] R  /* synthesis chip_pin="E3,G3,D8,B2,A3,A4,C6,H21" */;
  reg [8:1] TA, R ;   reg [4:1] TB;
  always @(A or B)  begin
    R = 0 ; TA = A ;  TB = B ;
     repeat(4) begin  if(TB[1]) begin R=R+TA; end
          TA = TA<<1; TB = TB>>1;   end
   end
endmodule
```

3.6 keep属性应用

【例 3-2】

```
module ff_adder(ain,bin,cin,cout,sum);
    output cout,sum ;
    input ain,bin,cin ;
    (* synthesis, keep *)  wire  net1 ;
    wire  net2,net3 ;
    h_adder  U1(  ain, bin, net1, net2);
    h_adder  U2(.A(net1), .SO(sum), .B(cin),.CO(net3) );
         or  U3(cout, net2, net3);
endmodule
```

3.6 keep属性应用

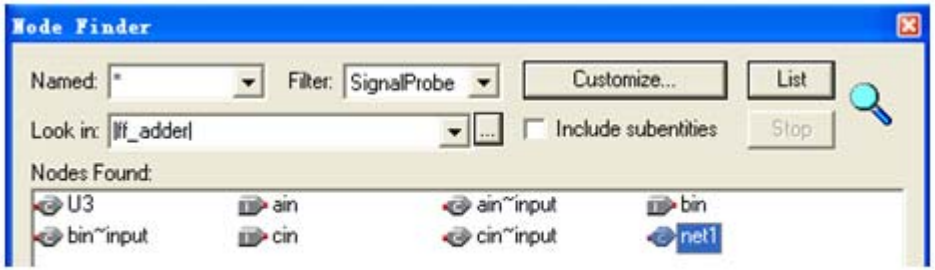


图 3-32 加入仿真测试信号 net1

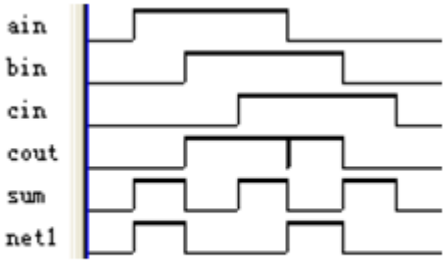


图 3-33 例 3-2 的仿真波形

3.7 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 3-35 ECO 文件编译成功

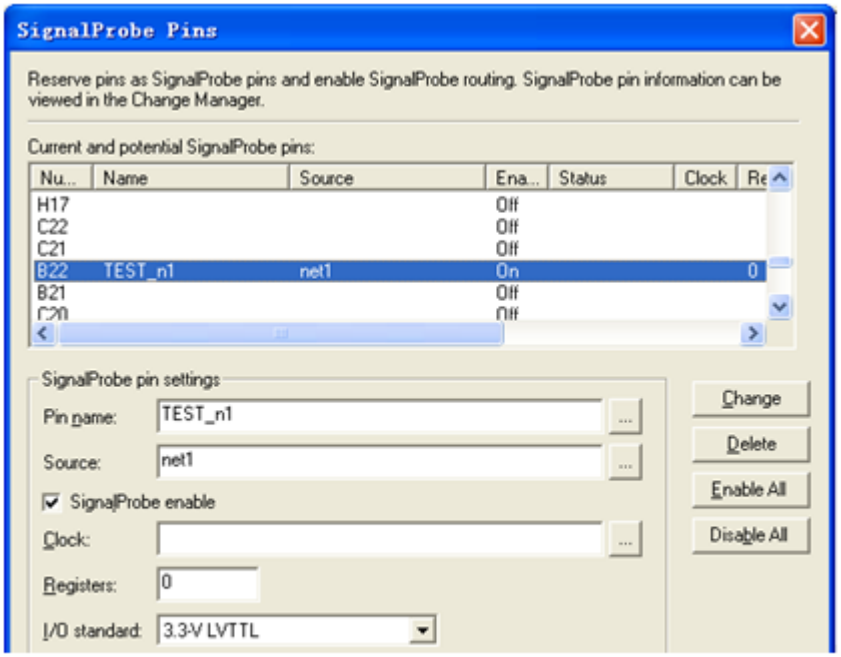


图 3-34 在 SignalProbe 对话框设置探测信号 net1

3. 编译SignalProbe Pins测试信息并下载测试

3.8 宏模块逻辑功能查询

```
FUNCTION 74138 (g1, g2an, g2bn, c, b, a)
   RETURNS (y0n, y1n, y2n, y3n, y4n, y5n, y6n, y7n);
```

Inputs						Outputs							
Enable			Select										
G1	G2*		C	B	A	Y0N	Y1N	Y2N	Y3N	Y4N	Y5N	Y6N	Y7N
X	H		X	X	X	H	H	H	H	H	H	H	H
L	X		X	X	X	H	H	H	H	H	H	H	H
H	L		L	L	L	L	H	H	H	H	H	H	H
H	L		L	L	H	H	L	H	H	H	H	H	H
H	L		L	H	L	H	H	L	H	H	H	H	H
H	L		L	H	H	H	H	H	L	H	H	H	H
H	L		H	L	L	H	H	H	H	L	H	H	H
H	L		H	L	H	H	H	H	H	H	L	H	H
H	L		H	H	L	H	H	H	H	H	H	L	H
H	L		H	H	H	H	H	H	H	H	H	H	L

* G2 = G2AN + G2BN

图 3-36　74138 真值表

习　题

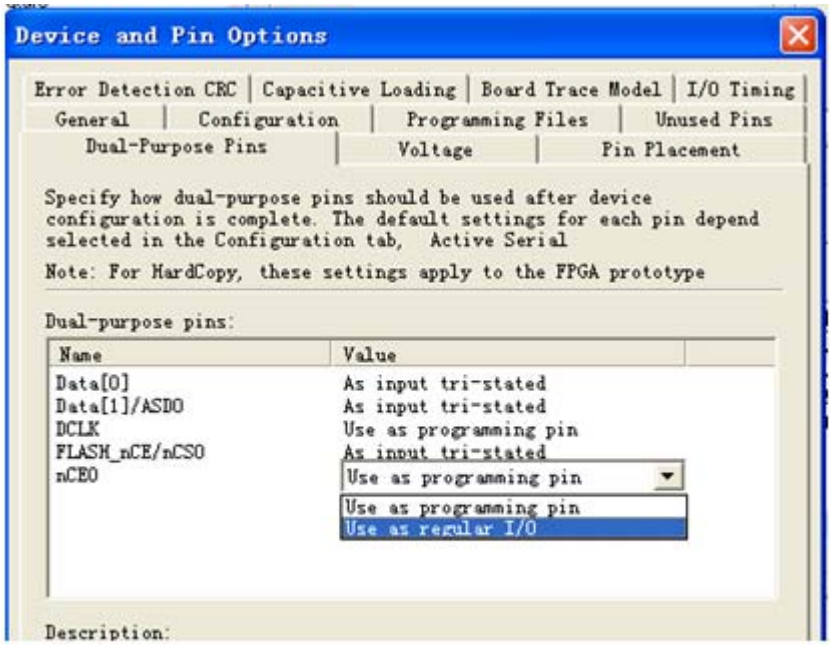
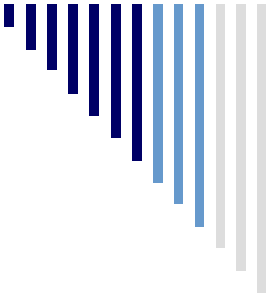


图 3-37　指定 nCEO 脚为普通 I/O pin



EDA实验

3-1 多路选择器设计实验

3-2 **8**位加法器设计实验

EDA实验

3-3 十六进制7段数码显示译码器设计

【例 3-3】

```
module DECL7S (A,LED7S) ;
  input [3:0] A;  output [6:0] LED7S ;   reg [6:0] LED7S ;
  always @ (A)
  case(A)
   4'b0000 :  LED7S <= 7'b0111111 ;
   4'b0001 :  LED7S <= 7'b0000110 ;
   4'b0010 :  LED7S <= 7'b1011011 ;
   4'b0011 :  LED7S <= 7'b1001111 ;
   4'b0100 :  LED7S <= 7'b1100110 ;
   4'b0101 :  LED7S <= 7'b1101101 ;
   4'b0110 :  LED7S <= 7'b1111101 ;
   4'b0111 :  LED7S <= 7'b0000111 ;
   4'b1000 :  LED7S <= 7'b1111111 ;
   4'b1001 :  LED7S <= 7'b1101111 ;
   4'b1010 :  LED7S <= 7'b1110111 ;
   4'b1011 :  LED7S <= 7'b1111100 ;
   4'b1100 :  LED7S <= 7'b0111001 ;
   4'b1101 :  LED7S <= 7'b1011110 ;
   4'b1110 :  LED7S <= 7'b1111001 ;
   4'b1111 :  LED7S <= 7'b1110001 ;
  default :  LED7S <= 7'b0111111 ;
  endcase
  endmodule
```

EDA实验

3-3 十六进制7段数码显示译码器设计

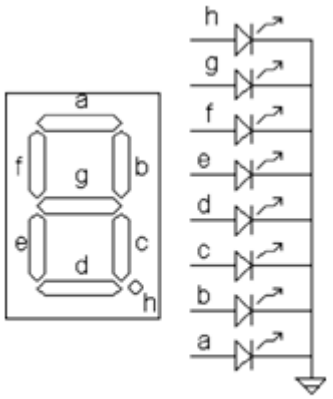


图 3-38 共阴数码管

表 3-1 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