

实验一 3—8 译码器的实现

班级 软件 1801 姓名 肖云杰 学号 201826010113

一、实验目的

1. 熟悉译码器的工作原理
2. 熟悉 Quartus II 软件的基本操作，了解各种设计方法（原理图设计、文本设计、波形设计）

二、实验内容

1. 熟悉 QuartusII 软件的基本操作，了解各种设计输入方法（原理图设计、文本设计、波形设计）
2. 用 VHDL 语言和原理图，分别设计一个 3—8 译码器，最后仿真实验验证。

三、实验方法

1、实验方法

采用基于 FPGA 进行数字逻辑电路设计的方法。
采用的软件工具是 Quartus II。

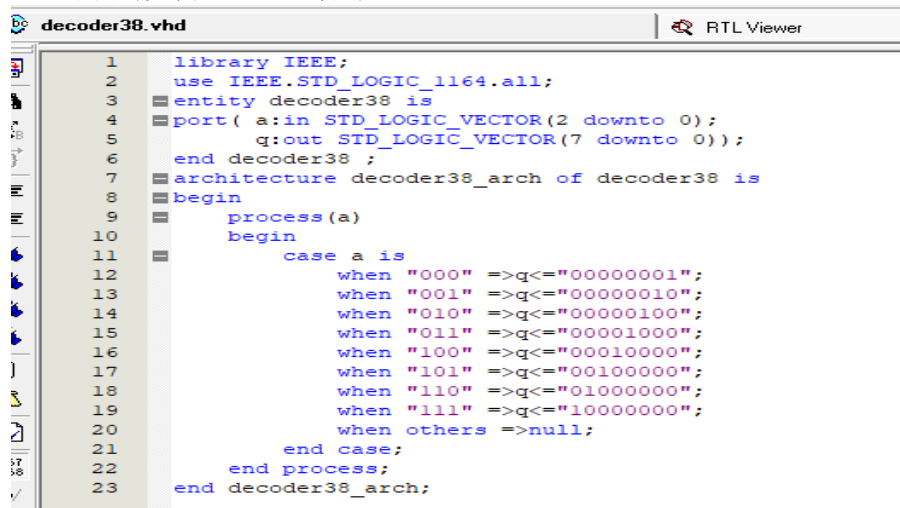
2、实验步骤

1. 新建，编写源代码。
 - (1).选择保存项和芯片类型：【File】-【new project wizard】-【next】（设置文件路径+设置 project name 为 decoder38）-【next】（设置文件名 decoder38.vhd—在【add】）-【properties】（type=AHDL）-【next】（family=Cyclone II; name=EP2C5T144C8）-【next】-【finish】
 - (2).新建：【file】-【new】-【OK】
2. 写好源代码，保存文件（decoder38.vhd）。
3. 编译与调试。确定源代码文件为当前工程文件，点击【processing】-【start compilation】进行文件编译，编译成功。
4. 波形仿真及验证。新建一个 vector waveform file。按照程序所述插入 in[2..0] out[7..0]节点（in 为输入节点，out 为输出节点）。（操作为：右击 -【insert】-【insert node or bus】-【node finder】（pins=all; 【list】）-【>>】-【ok】-【ok】）。任意设置 in 的输入波形...点击保存按钮保存,然后【start simulation】，出 output 的输出图。
5. 时序仿真或功能仿真。
6. 查看 RTL Viewer:【Tools】-【netlist viewer】-【RTL viewer】。

四、实验过程（3-8 译码器）

1、编译过程（VHDL 实现 3-8 译码器）

a) 源代码如图（VHDL 设计）



```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3  entity decoder38 is
4  port( a:in STD_LOGIC_VECTOR(2 downto 0);
5        q:out STD_LOGIC_VECTOR(7 downto 0));
6  end decoder38 ;
7  architecture decoder38_arch of decoder38 is
8  begin
9      process(a)
10     begin
11         case a is
12             when "000" =>q<="00000001";
13             when "001" =>q<="00000010";
14             when "010" =>q<="00000100";
15             when "011" =>q<="00001000";
16             when "100" =>q<="00010000";
17             when "101" =>q<="00100000";
18             when "110" =>q<="01000000";
19             when "111" =>q<="10000000";
20             when others =>null;
21         end case;
22     end process;
23 end decoder38_arch;
```

b)编译、调试过程

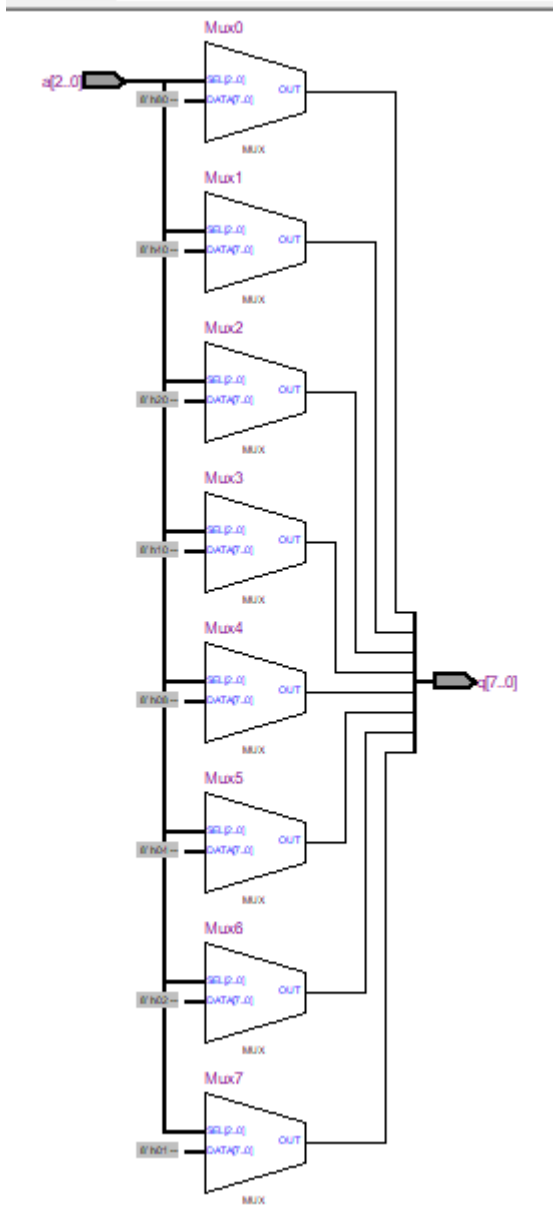
Warning: No exact pin location assignment(s) for 11 pins of 11 total pins

Warning: Found 8 output pins without output pin load capacitance assignment

Warning: The Reserve All Unused Pins setting has not been specified, and will default to 'As output driving ground'.

(以上三个警告，都是管脚未编制导致的)

c) RTL 视图



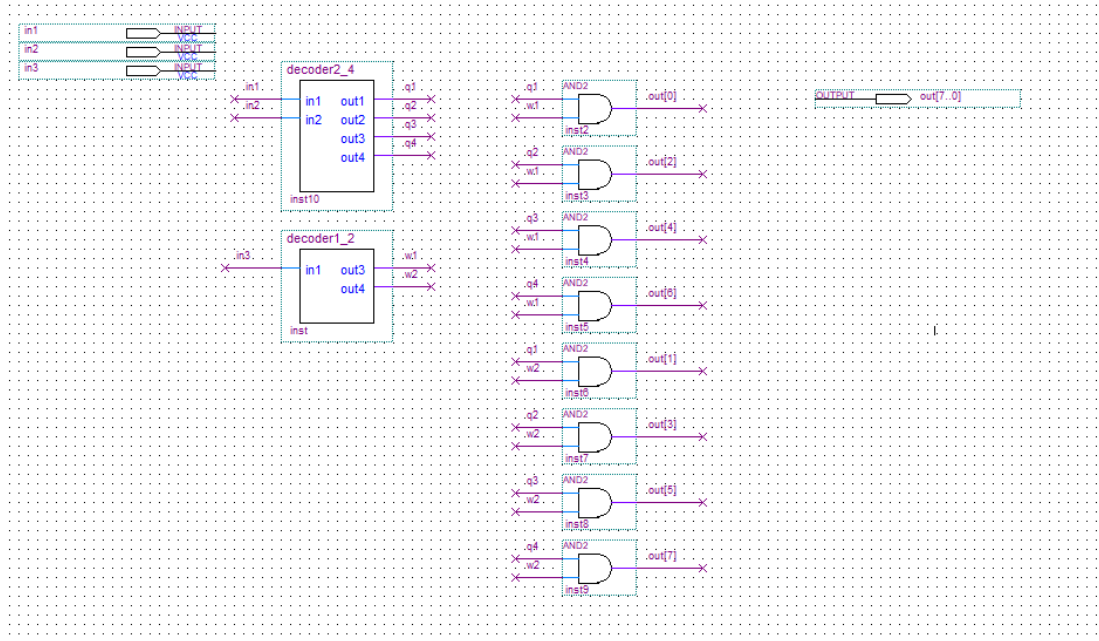
d)结果分析及结论

VHDL 文件（行为式描述）实现的 38 译码器编译实现基本完成，没有错误（除了管脚没有分配）

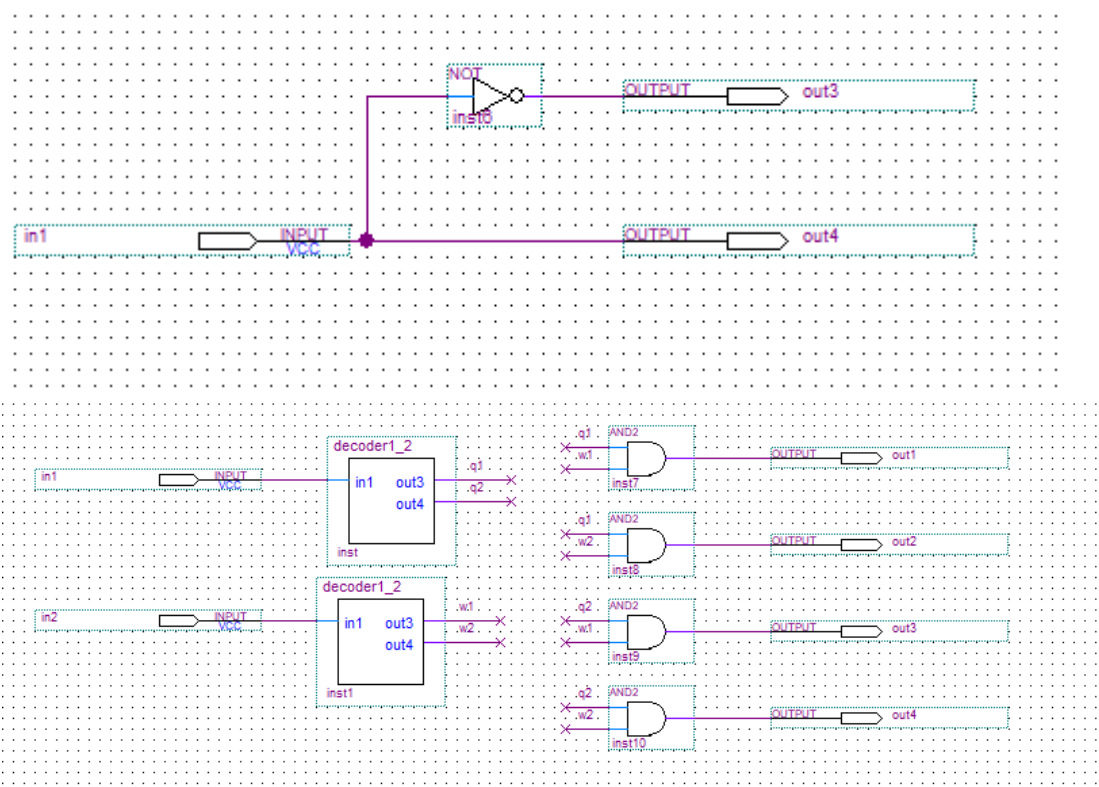
1、编译过程（原理图法实现 3-8 译码器）

a) 源代码如图（图像法）

顶层文件：



模块设计：



b)编译、调试过程

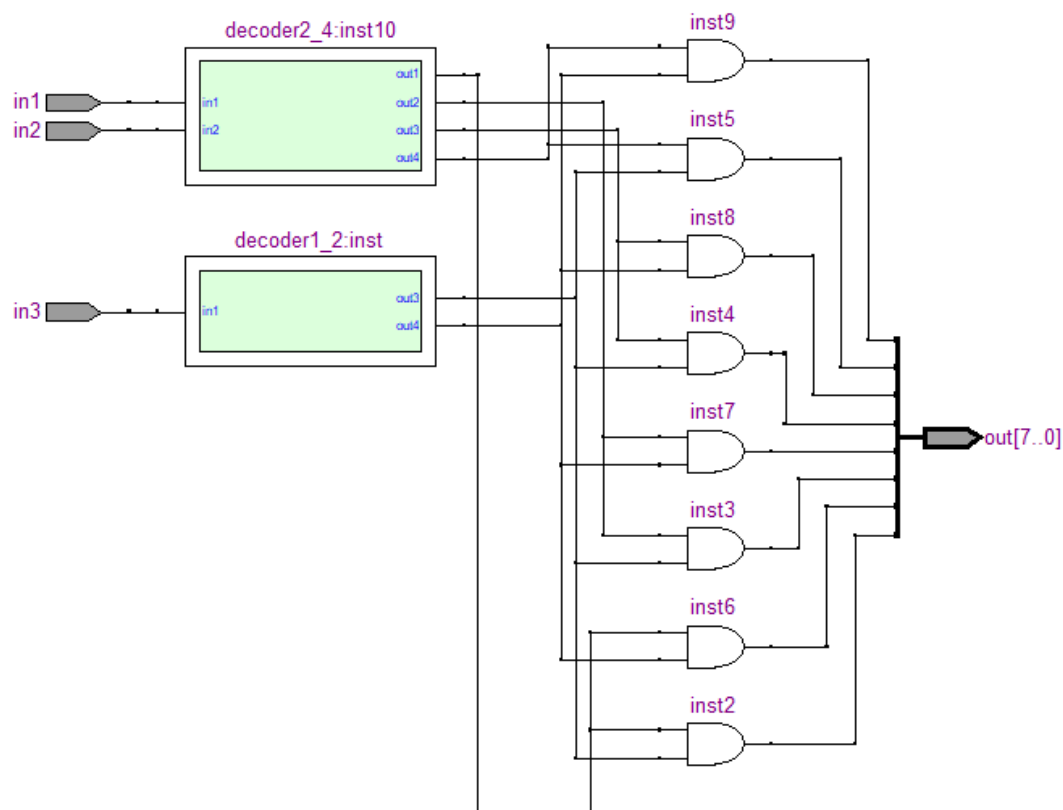
Warning: No exact pin location assignment(s) for 11 pins of 11 total pins

Warning: Found 8 output pins without output pin load capacitance assignment

Warning: The Reserve All Unused Pins setting has not been specified, and will default to 'As output driving ground'.

(以上三个警告，都是管脚未编制导致的)

c) RTL 视图



d) 结果分析及结论

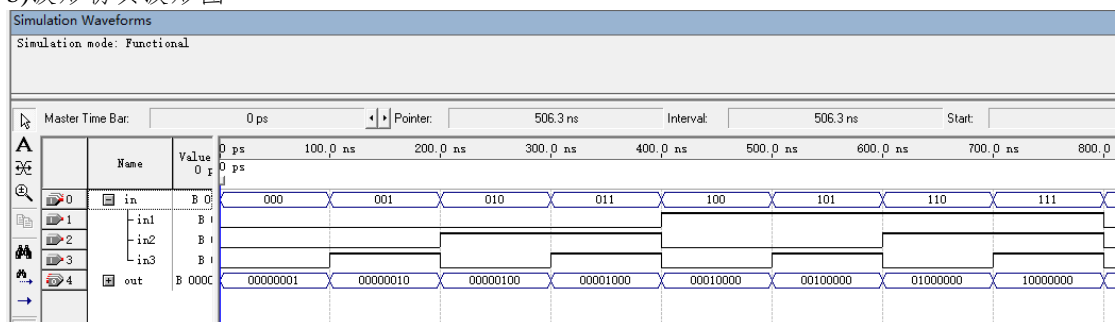
原理图描述实现的 38 译码器编译实现基本完成，没有错误（除了管脚没有分配）

2、波形仿真（3-8 译码器）

1) 波形仿真过程

波形仿真及验证，新建波形仿真文件【File】-【New】（选择Vector Waveform File），并且分别设置【End Time】为2us、【Grid Size】为100ns。按照程序所述插入in, out节点（in为输入节点，out为输出节点）。（操作为：双击引脚区-【Node Finder】（Pins=all; 【List】）-【>>】-【OK】-【OK】）。设置in的输入波形为周期型。

b) 波形仿真波形图



c) 结果分析及结论

0-100ns 输入 000 输出 00000001
 100-200ns 输入 001 输出 00000010
 200-300ns 输入 010 输出 00000100
 300-400ns 输入 011 输出 00001000
 400-500ns 输入 100 输出 00010000

500-600ns 输入 101 输出 00100000
 600-700ns 输入 110 输出 01000000
 700-800ns 输入 111 输出 10000000

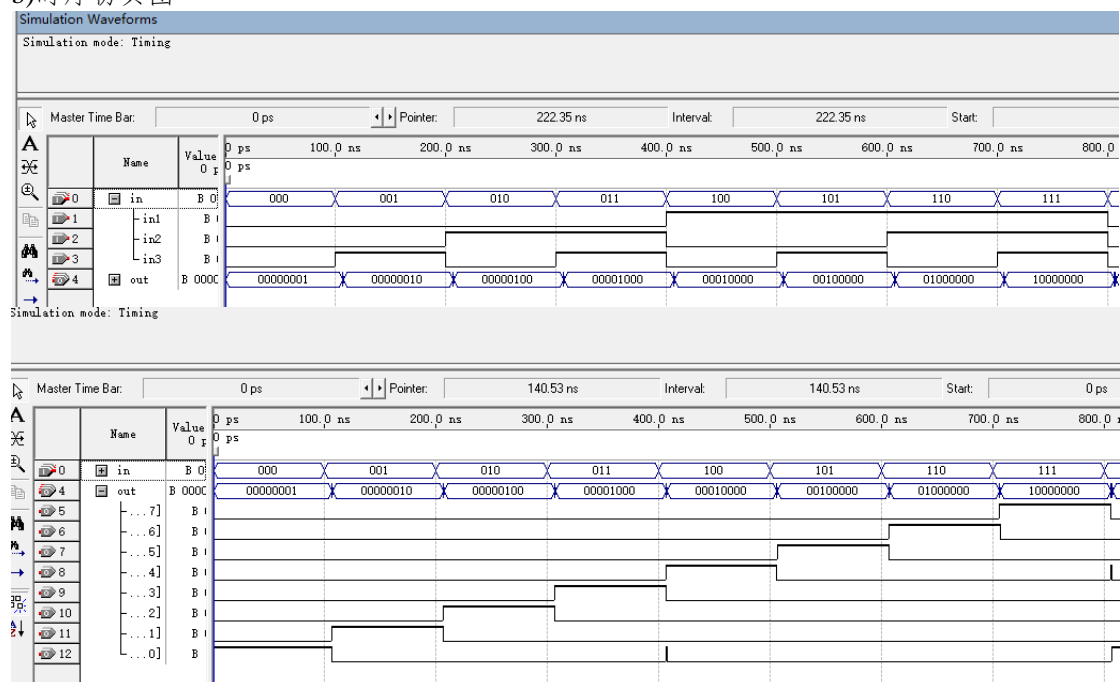
译码 输出正确

3、 时序仿真

a) 时序仿真过程

【Assignments】-【Setting】-【Simulator Settings】-【Timing】-【OK】-工具栏中【Start Simulation】，得到【Simulation Report】

b) 时序仿真图



b) 结果分析及结论

通过第一幅图可以看出：信号改变后，译码时存在延时，延时时间大约 11ns；

通过第二幅图，即放大后的波形可以看出：不仅信号存在延时，各个与门的状态的改变也不是同步的，会有其中一个的状态先发生改变，另一个才会发生改变。

五、实验过程（指令译码器）

1、编译过程（指令译码器）

A) 源代码如图（图像法）

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity instr is
5      port (
6          EN : in std_logic;
7          IR : in std_logic_vector(0 to 7);
8          MOVA, MOVNB, MOVNC, ADD, SUB, ORO, NOTO, RSR, RSL, JMP, JZ, JC, INO, OUTO, NOP, HALT : out std_logic
9      );
10 end entity instr;
11
12 architecture instr_op of instr is
13     signal command : std_logic_vector(3 downto 0);
14     signal R1, R2 : std_logic_vector(1 downto 0);
15 begin
16     command <= IR(0 to 3);
17     R1 <= IR(4 to 5);
18     R2 <= IR(6 to 7);
19     process(IR)
20     begin
21         MOVA <= '0';
22         MOVNB <= '0';
23         MOVNC <= '0';
24         ADD <= '0';
25         SUB <= '0';
26         ORO <= '0';
27         NOTO <= '0';
28         RSR <= '0';
29         RSL <= '0';
30         JMP <= '0';
31         JZ <= '0';
32         JC <= '0';
33         INO <= '0';
34         OUTO <= '0';
35         NOP <= '0';
36         HALT <= '0';
37         if EN = '0' then
38             if command = "1111" then
39                 if R1 = "11" then
40                     MOVNB <= '1';
41                 elsif R2 = "11" then
42                     MOVNC <= '1';
43                 else
44                     MOVA <= '1';
45                 end if;
46             elsif command = "1001" then
47                 ADD <= '1';
48             elsif command = "0110" then
49                 SUB <= '1';
50             elsif command = "1011" then
51                 ORO <= '1';
52             elsif command = "0101" then
53                 NOTO <= '1';
54             elsif command = "1010" then
55                 if R2 = "00" then
56                     RSR <= '1';
57                 else
58                     RSL <= '1';
59                 end if;
60             elsif command = "0001" then
61                 if R2 = "00" then
62                     JMP <= '1';
63                 elsif R2 = "01" then
64                     JZ <= '1';
65                 elsif R2 = "10" then
66                     JC <= '1';
67                 end if;
68             elsif command = "0010" then
69                 INO <= '1';
70             elsif command = "0100" then
71                 OUTO <= '1';
72             elsif command = "0111" then
73                 NOP <= '1';
74             elsif command = "1000" then
75                 HALT <= '1';
76             end if;
77         end if;
78     end process;
79 end architecture instr_op;

```

B) 编译、调试过程

Warning: No exact pin location assignment(s) for 25 pins of 25 total pins

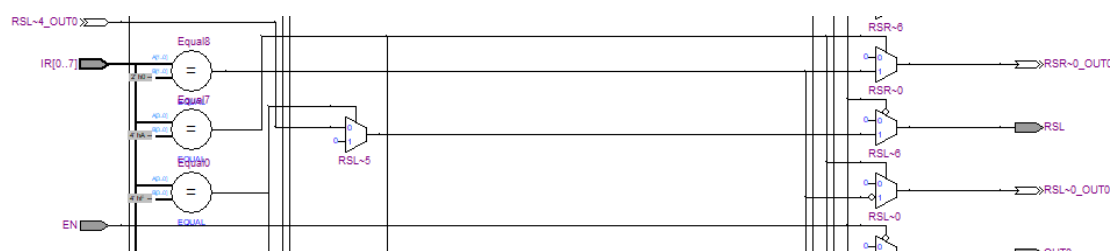
Warning: Some pins have incomplete I/O assignments. Refer to the I/O Assignment Warnings report for details

Warning: Found 16 output pins without output pin load capacitance assignment

Warning: The Reserve All Unused Pins setting has not been specified, and will default to 'As output driving ground'.

(以上四个警告，都是管脚未编制导致的)

C) RTL 原理图(部分)



D) 结果分析及结论

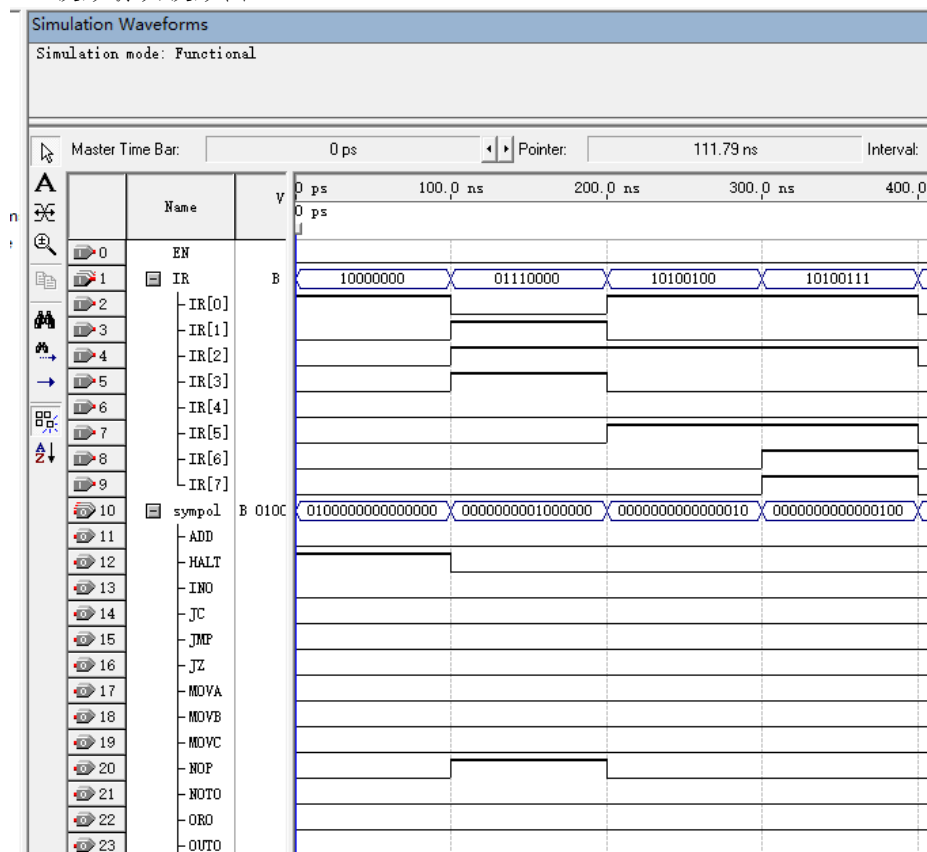
VHDL 文件实现的指令译码器编译实现基本完成，没有错误（除了管脚没有分配）

2、波形仿真（指令译码器）

A) 波形仿真过程

波形仿真及验证，新建波形仿真文件【File】-【New】（选择Vector Waveform File），并且分别设置【End Time】为2us、【Grid Size】为100ns。按照程序所述插入EN, IR, 16种操作节点（EN, IR为输入节点，操作为输出节点）。（操作为：双击引脚区-【Node Finder】（Pins=all; 【List】）-【>>】-【OK】-【OK】）。

B) 波形仿真波形图



C) 结果分析及结论

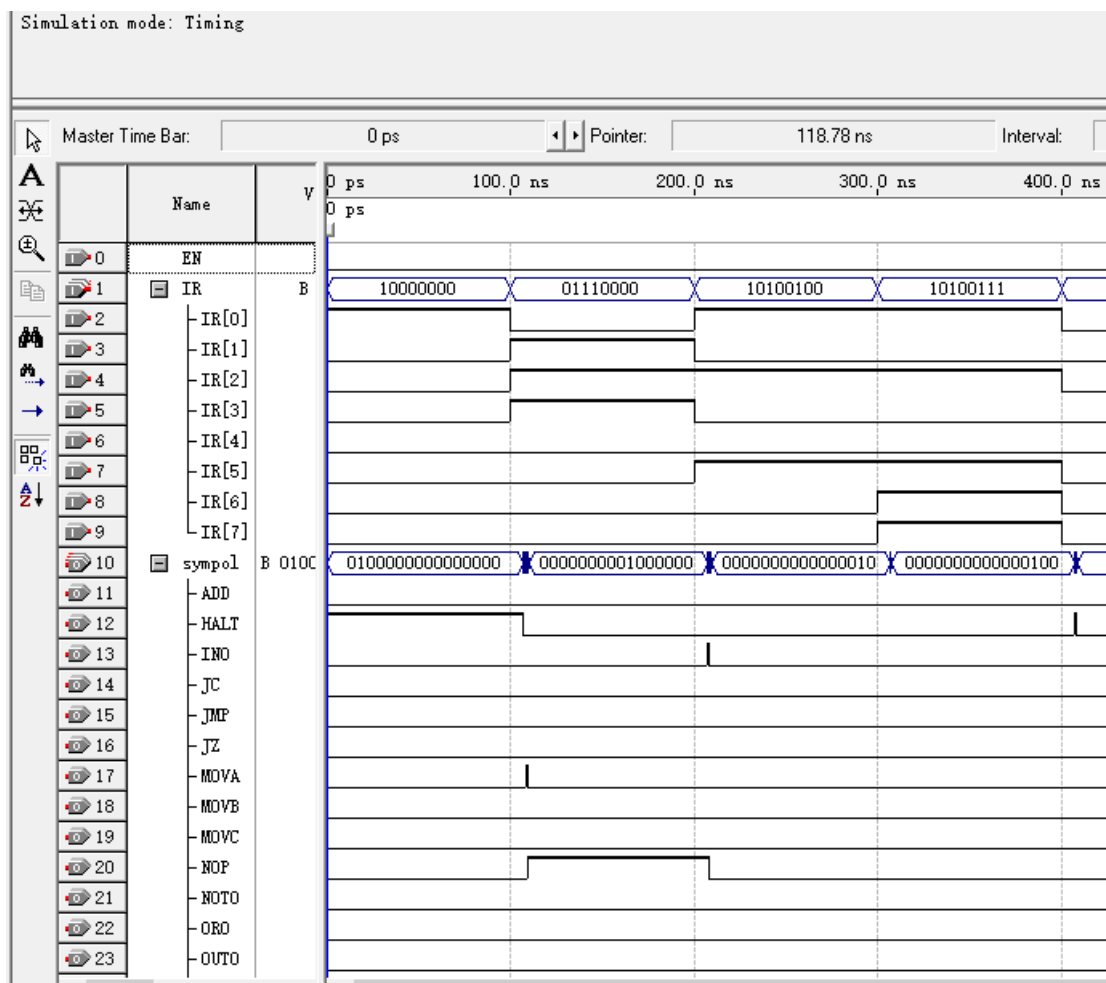
分析功能，按照功能表，功能正确实现

3、时序仿真（指令译码器）

a) 时序仿真过程

【Assignments】-【Setting】-【Simulator Settings】-【Timing】-【OK】-工具栏中【Start Simulation】，得到【Simulation Report】

b) 时序仿真图



c) 结果分析及结论

信号改变后，译码时存在延时，延时时间大约 11ns；功能实现正确

六、实验结论

1、思考题

1) 在日常生活中，我们哪些场所会用到译码器

- 译码器实现逻辑函数
- 译码器进行灯光控制
- 译码器可以构成数据分配器
- 译码器可以用来显示终端信号

2) 总结 VHDL 语言描述译码器电路的方法和常用语句

行为式语言描述，完全把所有的情况列出来，根据 case when else，赋值给输出语句，最后输出

VHDL 语言常用语句

- 1: 实体声明语句 entity XX is
- 2: 变量赋值语句 X:=
- 3: 信号赋值语句 X<=
- 4: architecture(结构体) architecture 结构体名 of 实体名 is
- 5: 进程体语句 process
- 6: 库的使用语句 library XXX
- 7: When else(并行语句)
- 8: if then else(顺序语句)

3) 比较原理图方式和 VHDL 方式设计组合逻辑的方法，步骤和优缺点

原理图方式：

首先确定组合逻辑电路的结构，然后根据结构画出顶层设计图，再根据顶层设计图，设计相应子模块的功能，画出原理图的子模块，将子模块添加到顶层设计图，完成顶层设计图，进行编译，仿真。

优点：

直观可见

设计比较简单的组合逻辑电路比较简单

缺点：

当组合逻辑电路比较复杂，原理图画的时候，也会非常复杂

画原理图可能会比较麻烦

VHDL 方式：

首先设计实体，写出输入和输出的信号，再根据实体，准备相应想要实现的功能，写出功能的结构体块，进行编译，仿真。

优点：

语法统一，即了解了结构之后，可以很容易写出

哪里错误了，会有严格的报错信息，可以根据信息 DEBUG

缺点：

需要学习新的语法，了解新的东西

对于新手，语法结构比较难

2、实验总结与实验心得

在本次实验中，开始由于什么都不懂，缺少了很多细节，不知道怎么处理。

如：

- a) 没有对输出设置初始值
- b) 错误的定义顶层文件名
- c) 造成的不能编译
- d) 标错端口造成的输出无序。
- e) 没有了解并行和顺序的区别，导致报错
- f) 不了解 vector(向量)的 downto 和 to，导致错误

通过报错信息的 debug 和询问同学，还是成功的完成了本次实验。

实验心得

做实验的时候，遇到 ERROR，不要被 ERROR 打败，吸取经验教训，积极看报错的代码行和代码块，合理使用 百度 和 谷歌 等，搜索引擎，找到自己的错误，吸取经验。

在真正做实验之前，与其从零开始，不如首先进行实验预习，比如这次实验，有 VHDL 语法和提前了解过 Quartus II 的人，就有明显的优势。

保持兴趣，锻炼自己的逻辑思维和代码能力。