# 实验二 多路复用器及控制信号产生逻辑的实现

班级 计科1808 姓名 张继伟 学号201808010829

一、实验目的

1．熟悉多路复用器以及模型机的工作原理。

2．学会使用 VHDL 语言设计多路复用器。

3．掌握 generic 的使用，能运用设计参数化多路复用器。

4．学会使用 VHDL 语言设计模型机控制信号产生逻辑。

二、实验内容

1、使用 VHDL 语言设计多路复用器。

2、掌握 generic 的使用，能运用设计参数化多路复用器。

3、使用 VHDL 语言设计模型机控制信号产生逻辑。

三、实验方法

1. （一）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。

1．多路复用器，又名多路选择器、多路开关

多路复用器是一个组合电路，它可以从多个输入中选择一个输入，并将信息

直接传输到输出。选择哪一条输入线由一组输入变量控制，它们被称为选择输入。

通常，2n 条输入线要 n 个选择输入，选择输入的位组合决定选择哪个输入线。

例如 n=1 的 2-1 多路复用器。这个复用器有两个信息输入 I0 和 I1，一个单独的

选择输入 S，电路的真值表如表 1 所示。

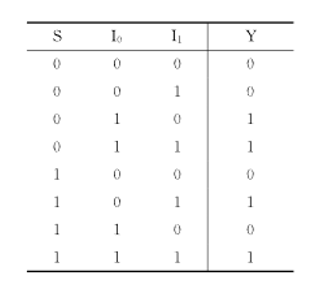


表 1 2-1 多路复用器真值表 分析真值表可知，如果选择输入 S=0，多路复用器输出为 I0 的值；

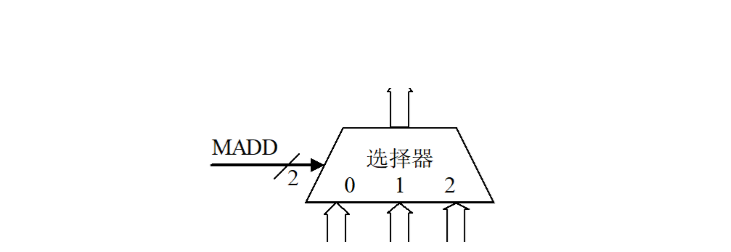
如果选择：

输入 S=1，多路开关输出 I1 的值。这样，S 不是选择输入 I0 就是选择输入 I1 到

输出 Y。通过这些讨论，可以看出，2-1 多路复用器输出 Y 的方程式为



2．8 重 3-1 多路复用器



3．运用 generic 设计参数化加法器，再调用定制为 4 位加法器。

a）参数化加法器

library ieee; use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity par\_add is generic(n:integer:=4);

port(a,b:in std\_logic\_vector(n downto 0);

s:out std\_logic\_vector(n downto 0);

c:out std\_logic);

end par\_add;

architecture exa of par\_add is signal t:std\_logic\_vector(n+1 downto 0);

begin

t<=('0'&a)+('0'&b);

s<=t(n downto 0);

c<=t(n+1); end exa;

b）设计参数化加法器，定制为 4 位加法器

library ieee; use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity add3 is

port(x,y:in std\_logic\_vector(3 downto 0);

cin:in std\_logic;

s:out std\_logic\_vector(3 downto 0);

cout:out std\_logic); end add3;

architecture exa of add3 is component par\_add is

generic(n:integer:=4);

port(a,b:in std\_logic\_vector(n downto 0);

s:out std\_logic\_vector(n downto 0);

c:out std\_logic); end component;

signal sum:std\_logic\_vector(3 downto 0);

signal mid:std\_logic\_vector(4 downto 0);

signal c3:std\_logic;

begin

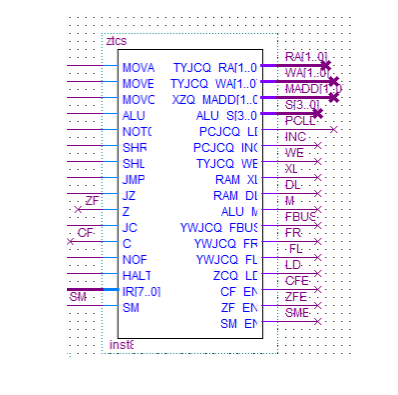
g0:par\_add generic map(n=>3)port map(a=>x,b=>y,s=>sum,c=>c3);

mid<=(c3&sum)+("0000"&cin);

s<=mid(3 downto 0);

cout<=mid(4); end exa;

4．控制信号产生逻辑



1. 实验步骤
2. 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xor2）-【next】（设置文件名xor2.vhd—在【add】）-【properties】（type=AHDL）-【next】（family=FLEX10K；name=EPF10K10TI144-4）-【next】-【finish】

(2).新建：【file】-【new】（第二个AHDL File）-【OK】

2、写好源代码，保存文件（xor2.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a,b,c三个节点（a、b为输入节点，c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a,b的输入波形…点击保存按钮保存。（操作为：点击name（如：A））-右击-【value】-【clock】（如设置period=200；offset=0），同理设置name B（如120，,60），保存）。然后【start simulation】，出name C的输出图。

5、时序仿真或功能仿真。

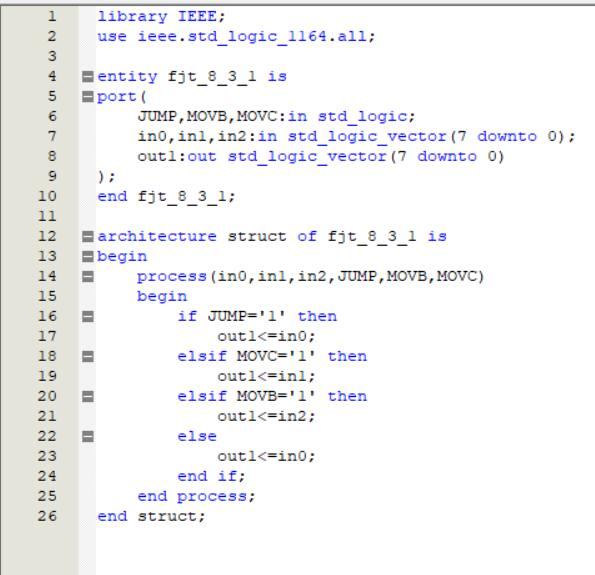
6、查看RTL Viewer:【Tools】-【netlist viewer】-【RTL viewer】。

四、实验过程

一、8 重 3-1 多路复用器

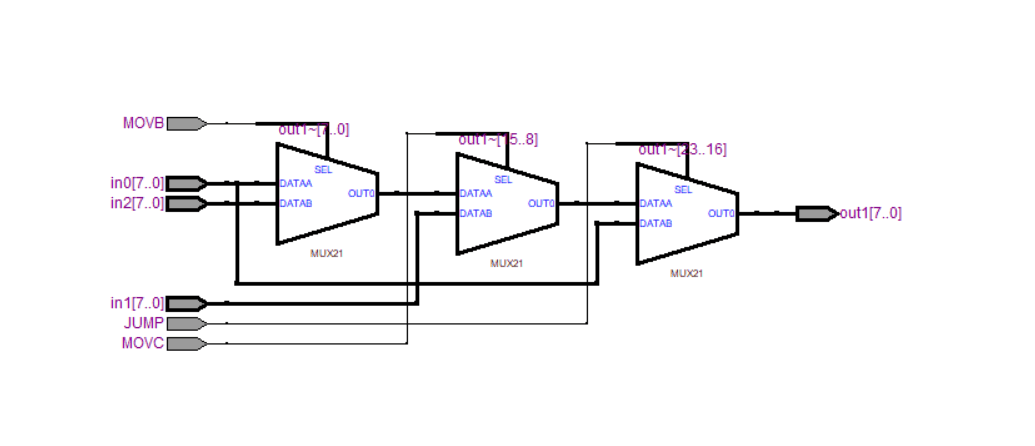
1. 编译过程

a）源代码如图（VHDL设计）



b)编译、调试过程

c) RTL视图



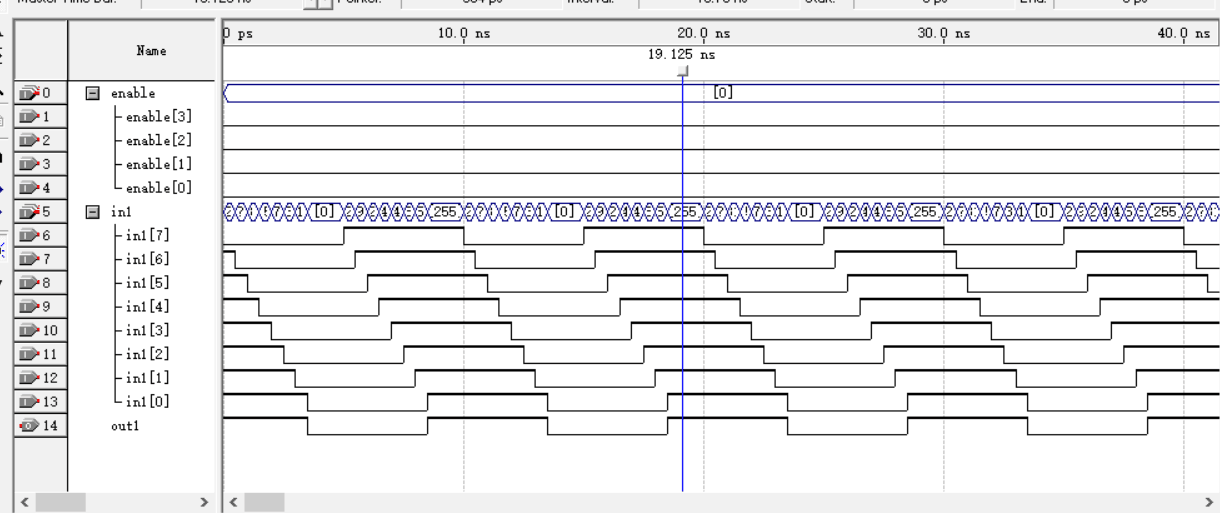
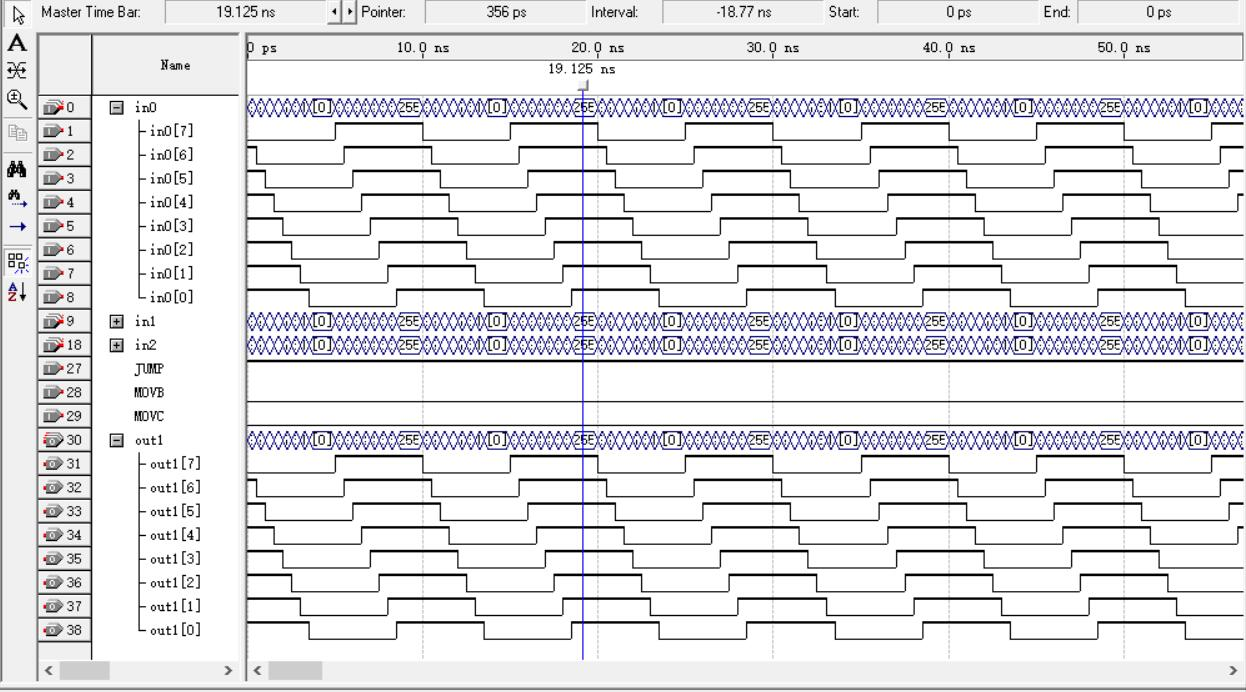
d)结果分析及结论

只要掌握了3-1多路复用器的原理，八重3-1多路复用器采用数据流描述相比结构描述能极大的简化代码长度。

1. 波形仿真

a)波形仿真过程（详见实验步骤）

b)波形仿真波形图



c)结果分析及结论

当输入为 00 时输出 i0

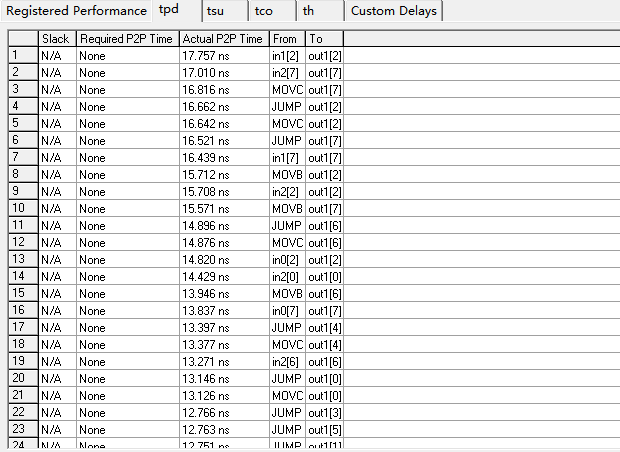
当输入为 01 时输出 i1

当输入为 10 时输出 i2

1. 时序仿真
2. 时序仿真过程

做好上述步骤后，编译【classic timing analysis】-在compilation report中选择【timing analysis】-【tpd】（引脚到引脚的延时）

b)时序仿真图



1. 结果分析及结论

当输入为 00 时输出 i0

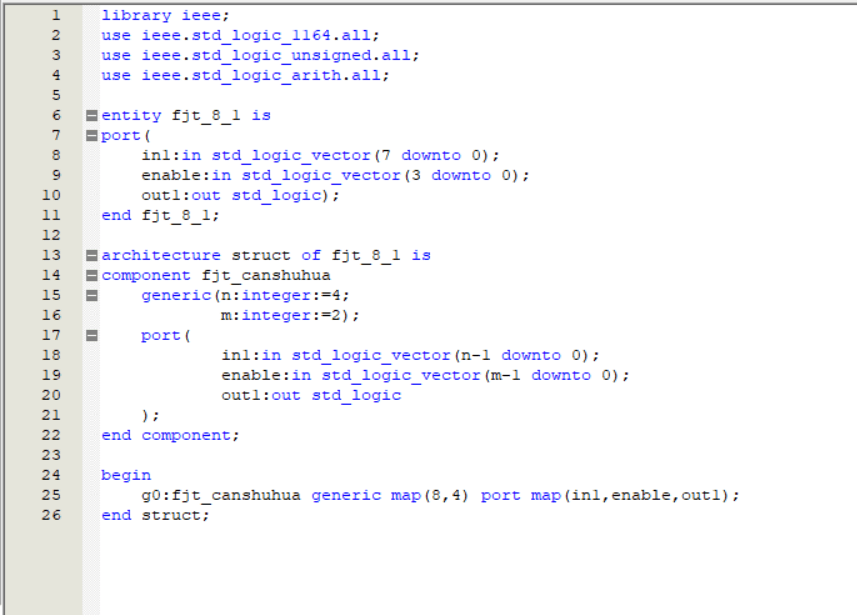
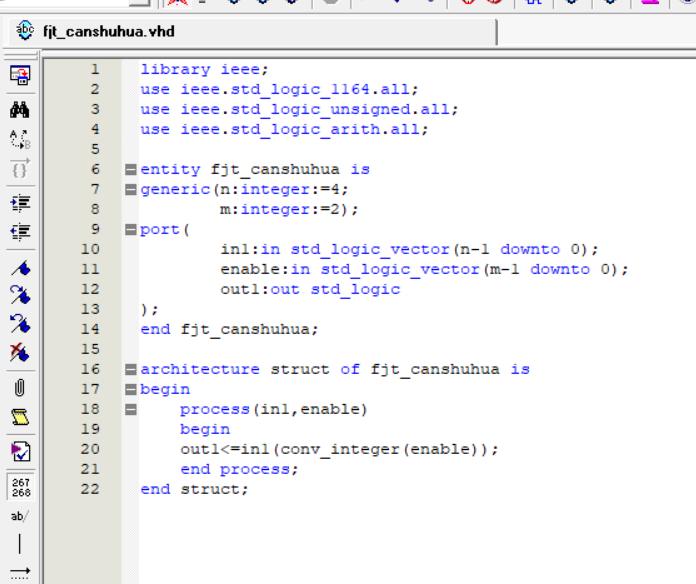
当输入为 01 时输出 i1

当输入为 10 时输出 i2

二、**参数化多路复用器定制为 8-1 多路复用器**

1. 编译过程

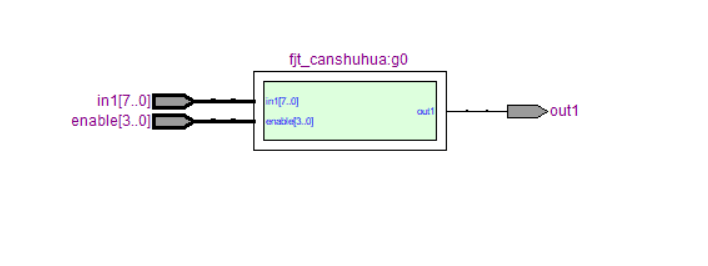
a）源代码如图（VHDL设计）



b)编译、调试过程



c) RTL视图



d)结果分析及结论

学习了如何使用参数化方法设计多路复用器

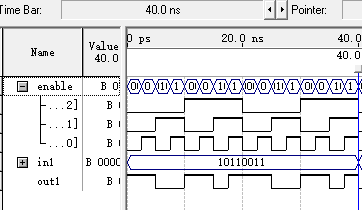
学会了generic函数设定参数

学会了使用conv\_integer函数实现单一变量的多路复用器输出

1. 波形仿真

a)波形仿真过程（详见实验步骤）

b)波形仿真波形图



c)结果分析及结论

当输入为 000 时，输出右边第一位

当输入为 001 时，输出右边第二位

当输入为 010 时，输出右边第三位

当输入为 011 时，输出右边第四位

当输入为 100 时，输出右边第五位

当输入为 101 时，输出右边第六位

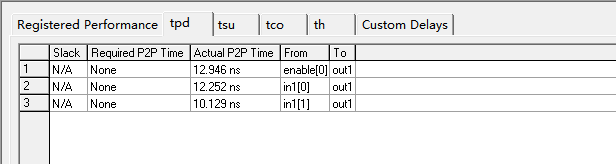
当输入为 110 时，输出右边第七位

当输入为 111 时，输出右边第八位

1. 时序仿真
2. 时序仿真过程

做好上述步骤后，编译【classic timing analysis】-在compilation report中选择【timing analysis】-【tpd】（引脚到引脚的延时）

b)时序仿真图

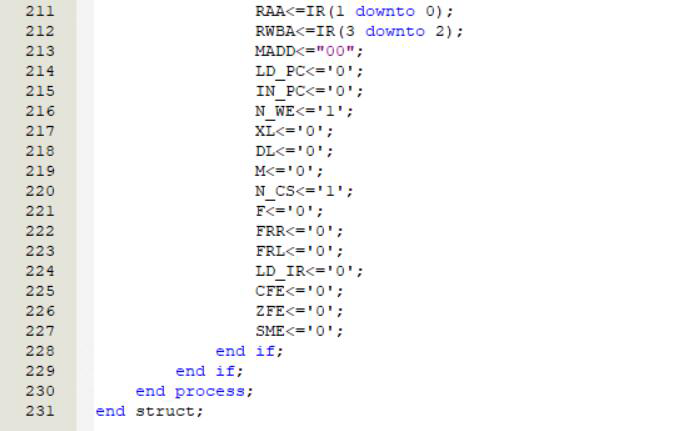
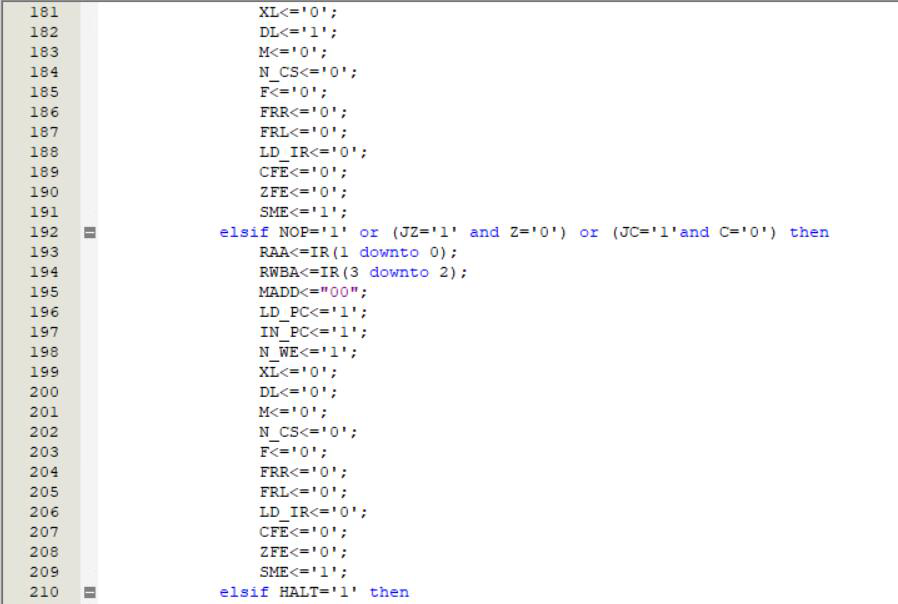
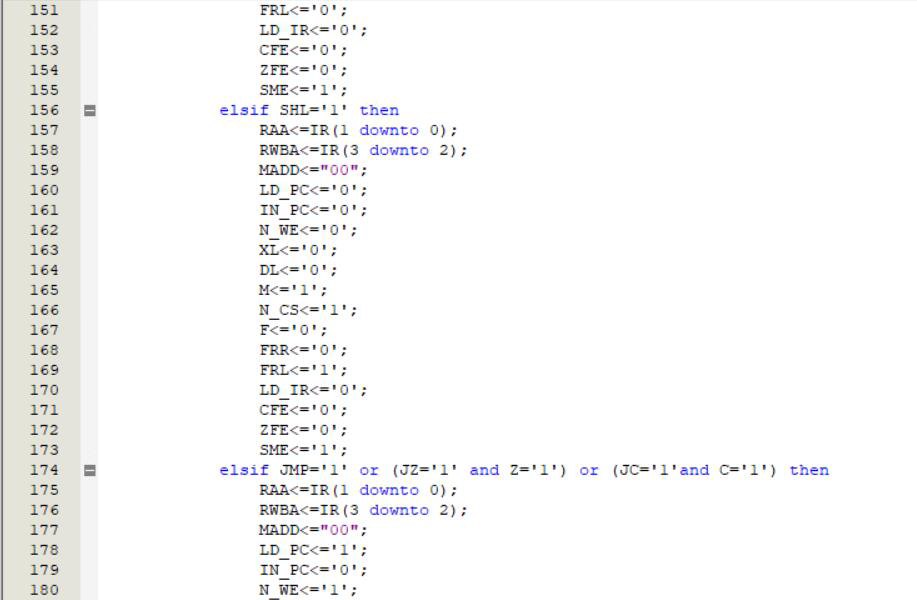
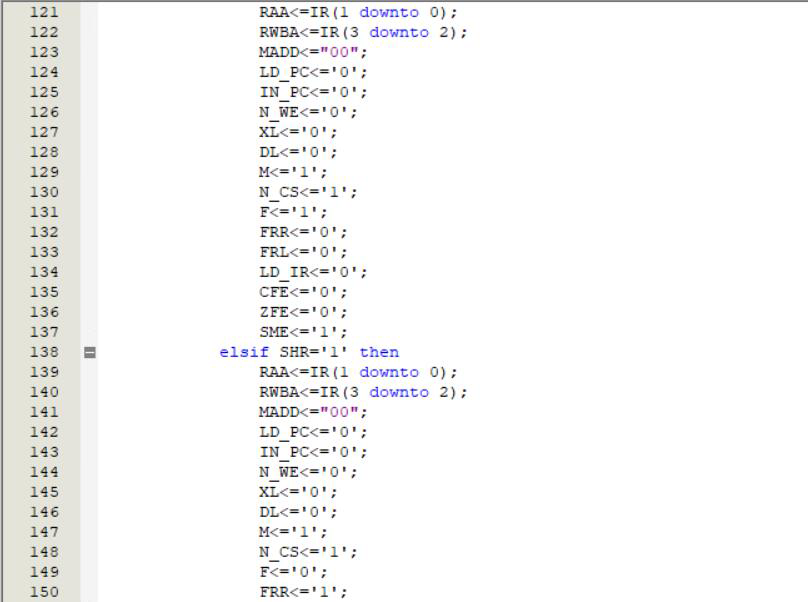
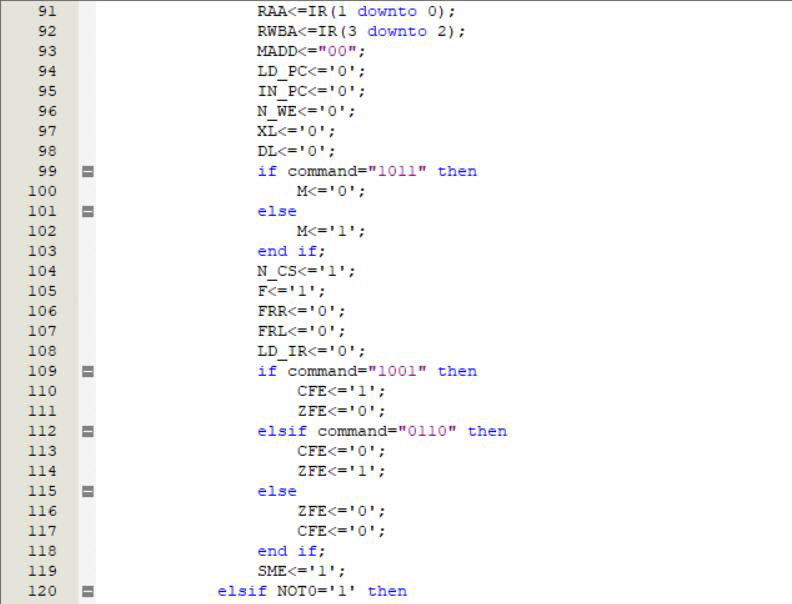
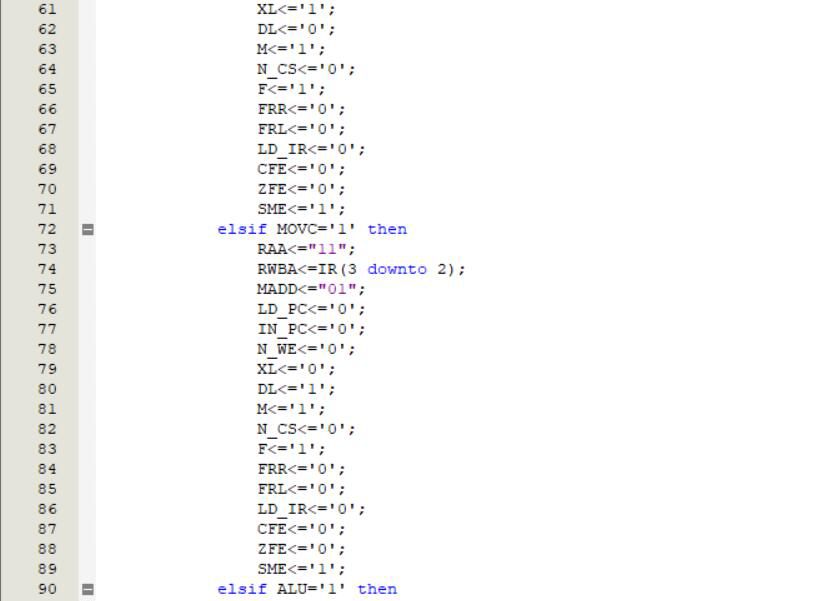
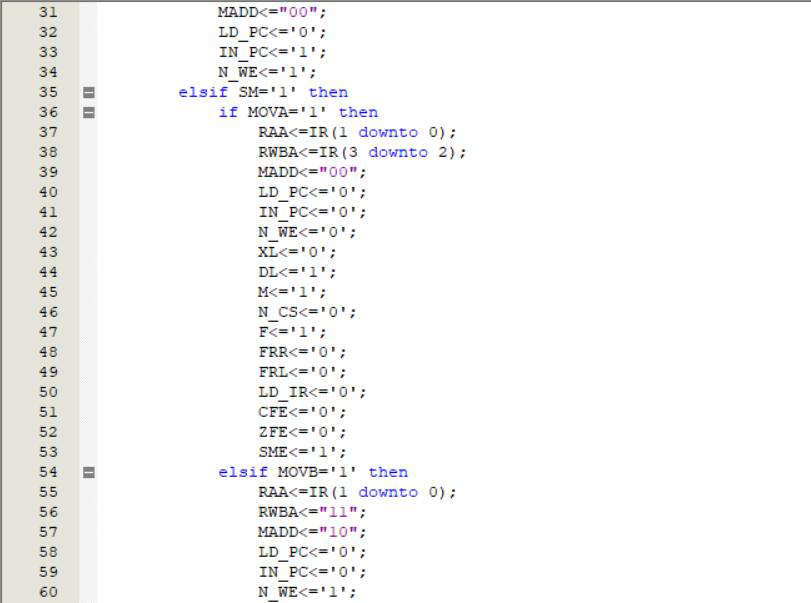
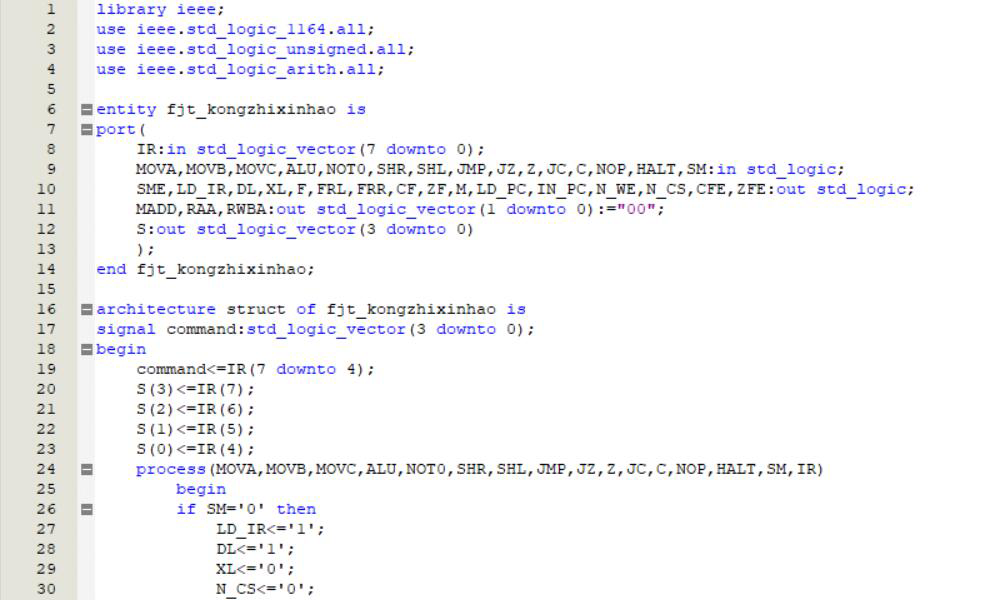


1. 结果分析及结论

三、模型机的控制信号产生逻辑

1. 编译过程

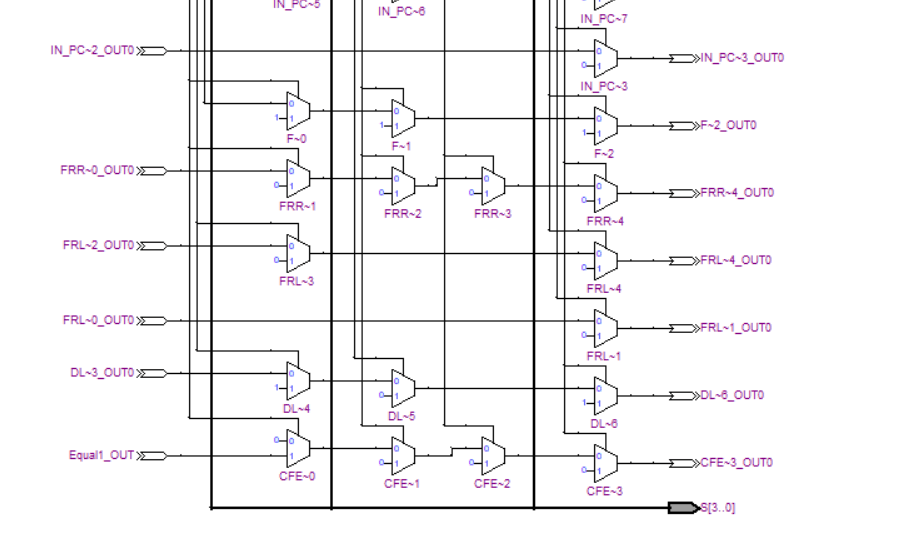
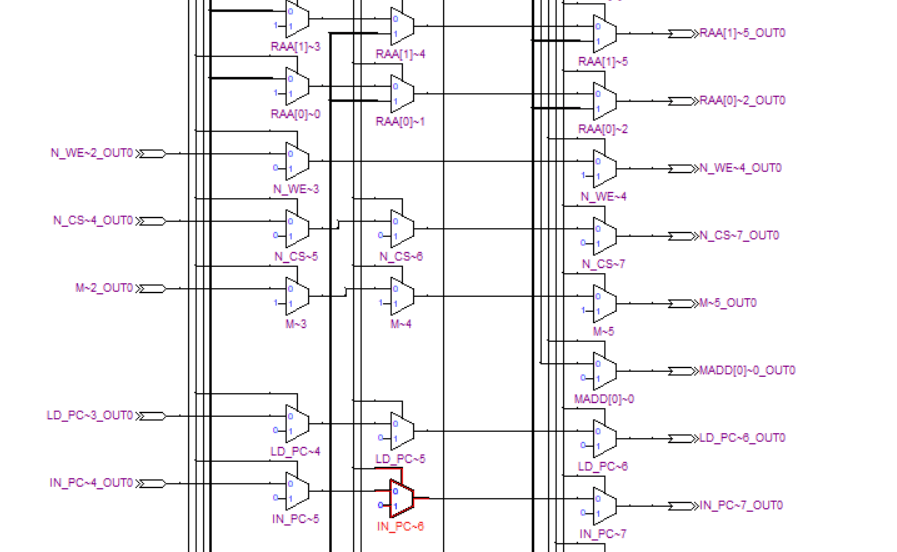
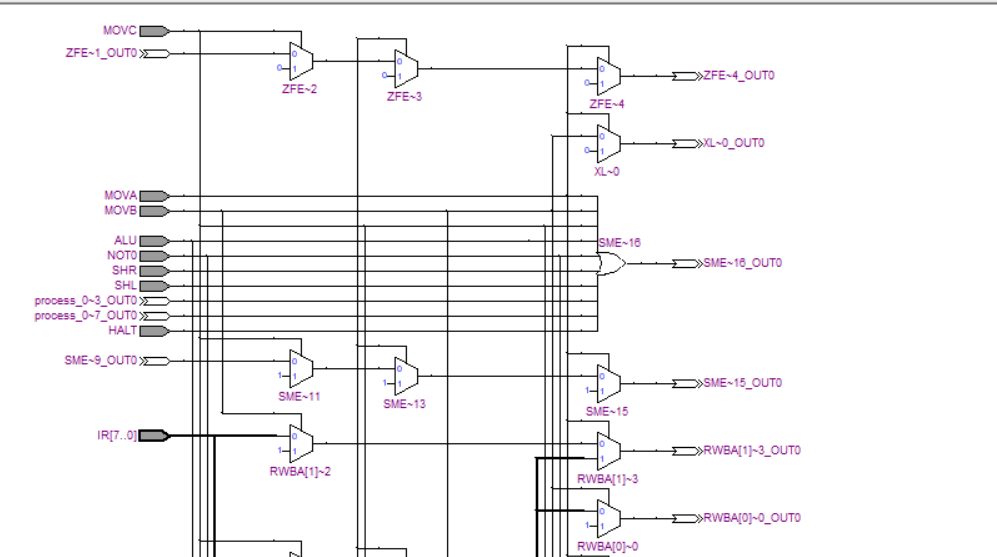
a）源代码如图（VHDL设计）



b)编译、调试过程



c) RTL视图



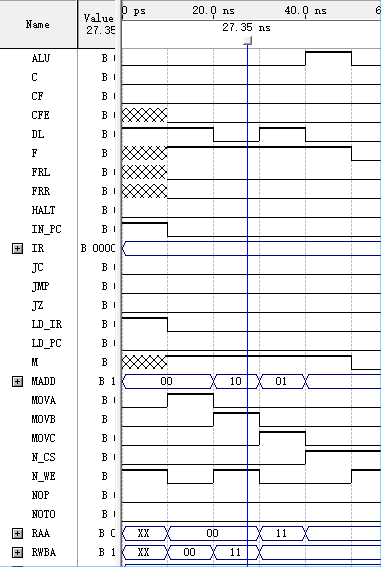
d)结果分析及结论

首先完成取指操作，并根据指令完成后续操作

1. 波形仿真

a)波形仿真过程（详见实验步骤）

b)波形仿真波形图

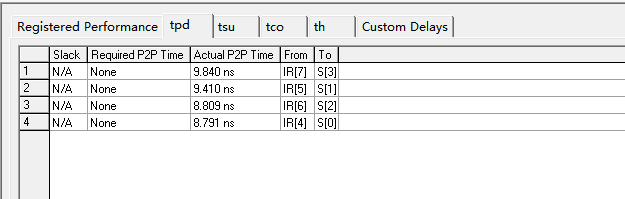


c)结果分析及结论

1. 时序仿真
2. 时序仿真过程

做好上述步骤后，编译【classic timing analysis】-在compilation report中选择【timing analysis】-【tpd】（引脚到引脚的延时）

b)时序仿真图



1. 结果分析及结论

tpd (引脚到引脚的延时)

五、实验结论

多路复用器的原理简单来说就是首先选择作用，输入00，选择第0号输入位的数据，01选择1号输入位的数据一次类推，至于几重多路复用器则代表可选数据的长度，例如一个8位二进制数则需要一个八重多路复用器才能实现选择。常用的函数 conv\_integer 可实现二进制转十进制从而输出vector中内容实现多路复用器。

要掌握了3-1多路复用器的原理，八重3-1多路复用器采用数据流描述相比结构描述能极大的简化代码长度。

学习了如何使用参数化方法设计多路复用器。

学会了generic函数设定参数。

学会了使用conv\_integer函数实现单一变量的多路复用器输出。