# 实验二 多路复用器与加法器的实现

# 班级 计科1903 姓名 陈旭 学号 201914020128

一、实验目的

1. 熟悉多路复用器、加法器的工作原理。
2. 学会使用 VHDL 语言设计多路复用器、加法器。
3. 掌握 generic 的使用，设计 n-1 多路复用器。
4. 兼顾速度与成本，设计行波加法器和先行进位加法器。

二、实验内容

1. 用 VHDL 语言设计 8 重 3-1 多路复用器。
2. 用 VHDL 语言设计 n-1 多路复用器，调用该 n-1 多路复用器定制为 8-1多路复用器。
3. 用 VHDL 语言设计 4 位行波进位加法器。
4. 用 VHDL 语言设计 4 位先行进位加法器。

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。
   * 采用基本逻辑门电路和组合逻辑电路实现8 重 3-1 多路复用器、n-1 多路复用器，并调用该 n-1 多

路复用器定制为 8-1多路复用器、4 位行波进位加法器、4 位先行进位加法器。

* + 采用的软件工具是Quartus II。

1. 实验软件操作步骤

* 8重3-1多路复用器

1. 新建，编写源代码。

(1)选择保存项和芯片类型：【File】-【new project wizard】-【next】（设置文件路径为C:\Users\86150\Desktop\mylearn\vscode-c++\logic and computer design fundamentals\experiment2\mux\_8\_3\_1设置project name为mux\_8\_3\_1）-【next】（设置文件名mux\_8\_3\_1）-【next】（设置芯片类型为【cyclone-EP1CT144C8】）-【finish】

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

1. 写好源代码，保存文件（mux\_8\_3\_1.vhd）。
2. 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，编译结果有警告，编译成功。
3. 波形仿真及验证。新建一个vector waveform file。按照程序所述插入inputs0, inputs1, inputs2, selinputs, outputs四个节点（inputs0, inputs1, inputs2为八位输入节点向量，selinputs 为两位输出节点向量， outputs为八位输出节点向量）。(操作为：右击 -【insert】-【insert node or bus】-【node finder】（pins=all；【list】）-【>>】-【ok】-【ok】)。任意设置inputs0, inputs1, inputs2, selinputs的输入波形…点击保存按钮保存。（操作为：点击name（如：enable））-右击-【value】-【count】（如设置binary；start value=0; count value=5.0ns），同理设置name b（如0,1,5），保存）。然后【start simulation】，出name C的输出图。
4. 时序仿真和功能仿真。
5. 查看RTL Viewer:【Tools】-【netlist viewer】-【RTL viewer】。

* **n-1多路复用器定制的8-1多路复用器**

1. 新建，编写源代码。

(1)选择保存项和芯片类型：【File】-【new project wizard】-【next】（设置文件路径为C:\Users\86150\Desktop\mylearn\vscodec++\logicandcomputerdesignfundamentals\experiment2\generic\_mux设置两个文件的project name为generic1(类属实体声明文件)以及generic\_mux\_8\_1(调用类属实体的顶层实体文件)）-【next】（设置文件名generic1、generic\_mux\_8\_1）-【next】（设置芯片类型为【cyclone-EP1CT144C8】）-【finish】

(2).新建：【file】-【new】（VHDL File）-【OK】

1. 写好源代码，保存文件（generic1.vhd、generic\_mux\_8\_1.vhd）。
2. 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，编译结果有警告，编译成功。
3. 波形仿真及验证。新建generic\_mux\_8\_1的vector waveform file。按照程序所述插入节点inputs, output, selects。（inputs, selects为输入节点向量，output为输出节点）。(操作为：右击 -【insert】-【insert node or bus】-【node finder】（pins=all；【list】）-【>>】-【ok】-【ok】)。按顺序设置inputs和selects的输入波形…点击保存按钮保存。（操作为：点击name（如：inputs））-右击-【value】-【count】（如设置binary；start value=00000000; end value=11111111;count value=2.0ns），同理设置name b（如000），保存）。然后【start simulation】，出name C的输出图。
4. 功能仿真判断代码的实现无误后时序仿真。
5. 查看RTL Viewer:【Tools】-【netlist viewer】-【RTL viewer】

四、实验过程

1. 编译过程
2. **源代码（VHDL设计）如下**

* **8重3-1多路复用器**

library ieee;

use ieee.std\_logic\_1164.all;

entity mux\_8\_3\_1 is

    port(

        inputs0: in std\_logic\_vector(7 downto 0);

        inputs1: in std\_logic\_vector(7 downto 0);

        inputs2: in std\_logic\_vector(7 downto 0);

        selinputs: in std\_logic\_vector(1 downto 0);

        outputs: out std\_logic\_vector(7 downto 0)

    );

end mux\_8\_3\_1;

architecture mux\_8\_3\_1 of mux\_8\_3\_1 is

    begin

        outputs<=inputs0 when selinputs="00" else

        inputs1 when selinputs="01" else

        inputs2 when selinputs="10"else

        "00000000";

end mux\_8\_3\_1;

* **n-1多路复用器定制的8-1多路复用器**
* **generic1.vhd(类属实体声明文件)**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity generic1 is

    --generic (n:integer:=3);

    generic (n:integer);

    port(

        genein: in std\_logic\_vector(2\*\*n-1 downto 0);

        genesel: in std\_logic\_vector(n-1 downto 0);

        geneout: out std\_logic

    );

    end generic1;

architecture generic1 of generic1 is

    begin

        geneout<=genein(conv\_integer(genesel));

    end generic1;

* **generic\_mux\_8\_1(调用类属实体的顶层实体文件)**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity generic\_mux\_8\_1 is

    port(

        inputs: in std\_logic\_vector(7 downto 0);

        output: out std\_logic;

        selects: in std\_logic\_vector(2 downto 0)

    );

end generic\_mux\_8\_1;

architecture generic\_mux\_8\_1 of generic\_mux\_8\_1 is

    component generic1 is

        generic (n:integer:=3);

        port(

            genein: in std\_logic\_vector(2\*\*n-1 downto 0);

            genesel: in std\_logic\_vector(n-1 downto 0);

            geneout: out std\_logic

        );

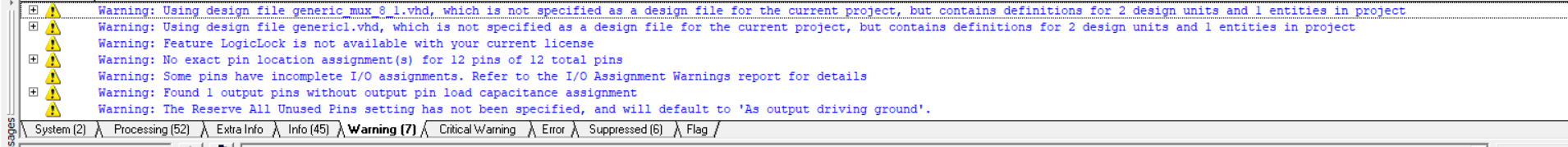
    end component;

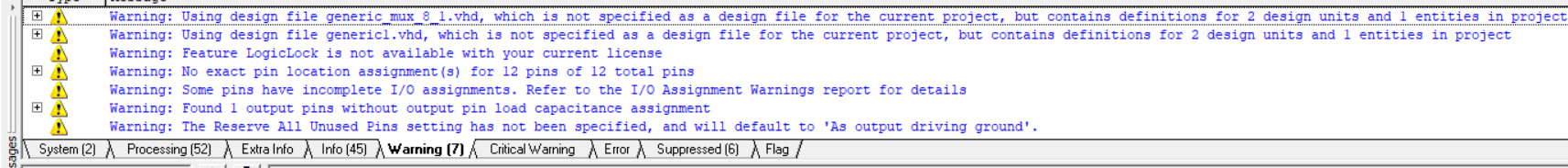
    begin

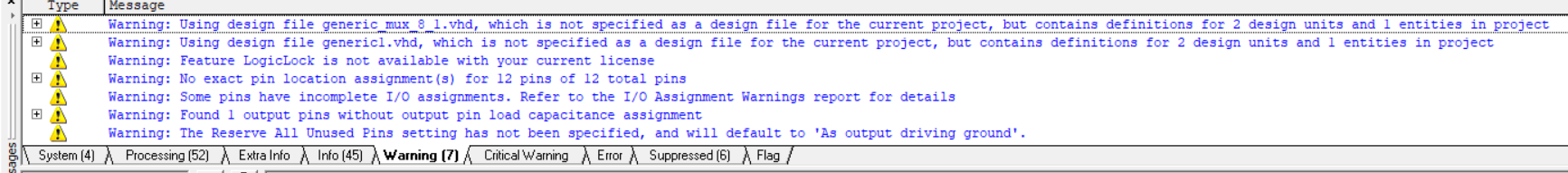
         g1:generic1 generic map(n=>3) port map(genein=>inputs, geneout=>output, genesel=>selects);

    end generic\_mux\_8\_1;

b)编译、调试过程







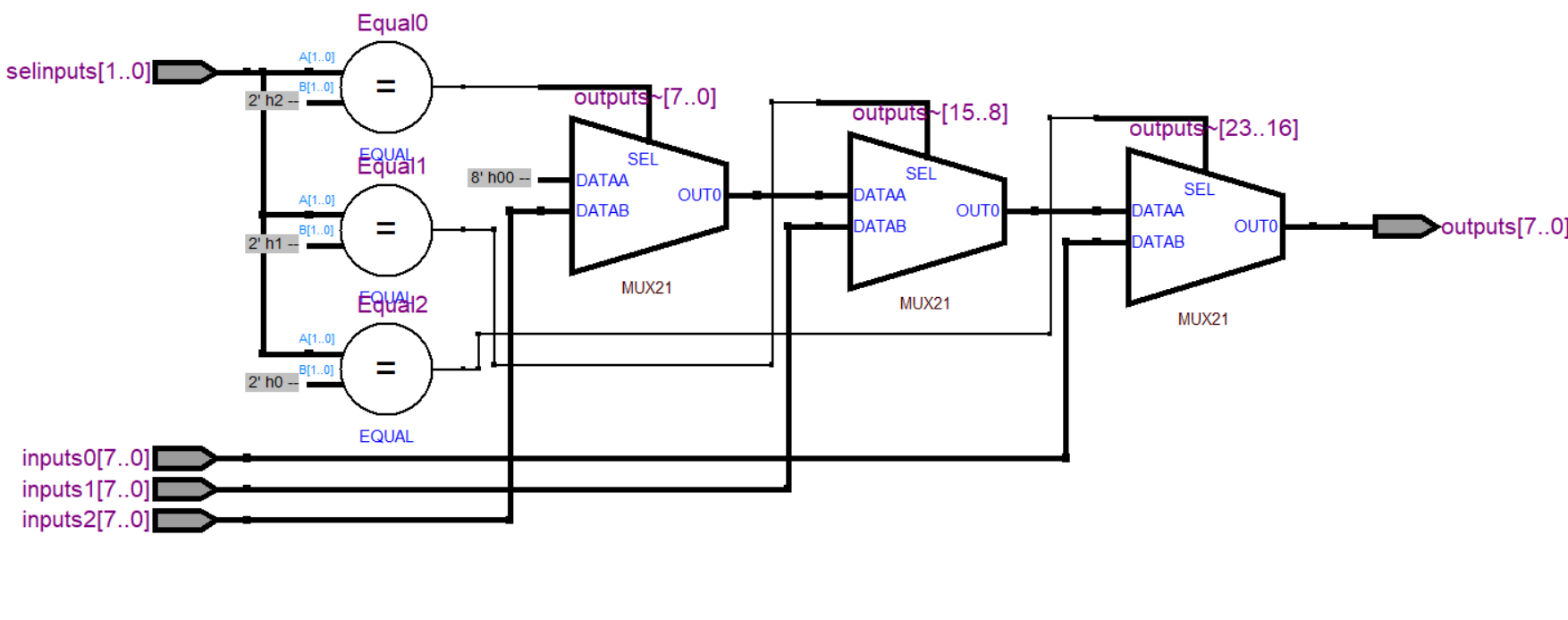
自上至下依次为mux\_8\_3\_1.vhd、generic1.vhd、generic\_mux\_8\_1.vhd的编译器提示信息。

三者编译器均给出警告，但均能够编译通过。

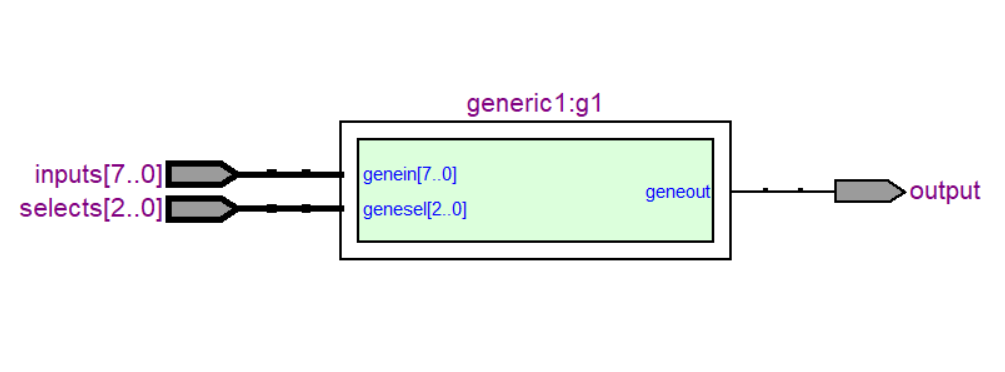
三者资源消耗基本都趋近于0.

c) RTL视图

* 8重3-1多路复用器



* 调用类属实体实现的8-1多路复用器



d)结果分析及结论

* 8重3-1多路复用器

当输入selinputs为 0 0输出inputs0对应的向量值

当输入selinputs为 0 1输出inputs1对应的向量值

当输入selinputs为 1 0输出inputs2对应的向量值

当输入selinputs为 1 1输出00000000(无效选择输入)

* 调用类属实体实现的8-1多路复用器

当输入select为000时，输出向量inputs[0]对应的元素

当输入select为001时，输出向量inputs[1]对应的元素

当输入select为010时，输出向量inputs[2]对应的元素

当输入select为011时，输出向量inputs[3]对应的元素

当输入select为100时，输出向量inputs[4]对应的元素

当输入select为101时，输出向量inputs[5]对应的元素

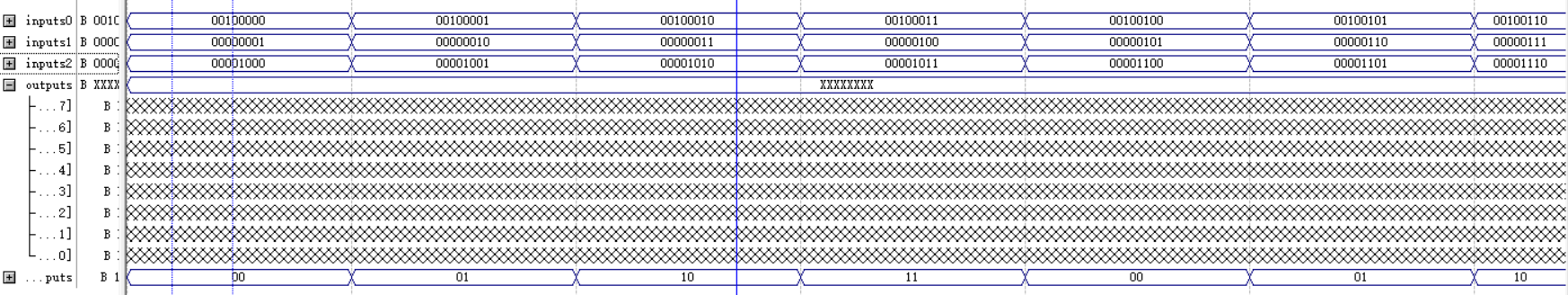
当输入select为110时，输出向量inputs[6]对应的元素

当输入select为111时，输出向量inputs[7]对应的元素

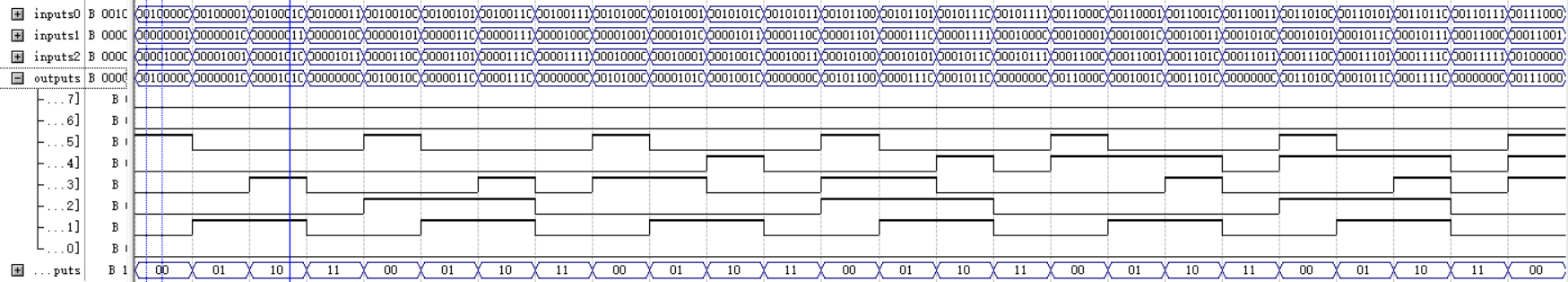
1. 波形仿真

* 8重3-1多路复用器

a)波形仿真过程（过程详见实验步骤）



b)波形仿真波形图



c)结果分析及结论

以第一个周期为例

输入向量selinputs为00时，输出inputs0对应值，为00100000;

输入向量selinputs为01时，输出inputs1对应值，为00000010;

输入向量selinputs为10时，输出inputs2对应值，为00001010;

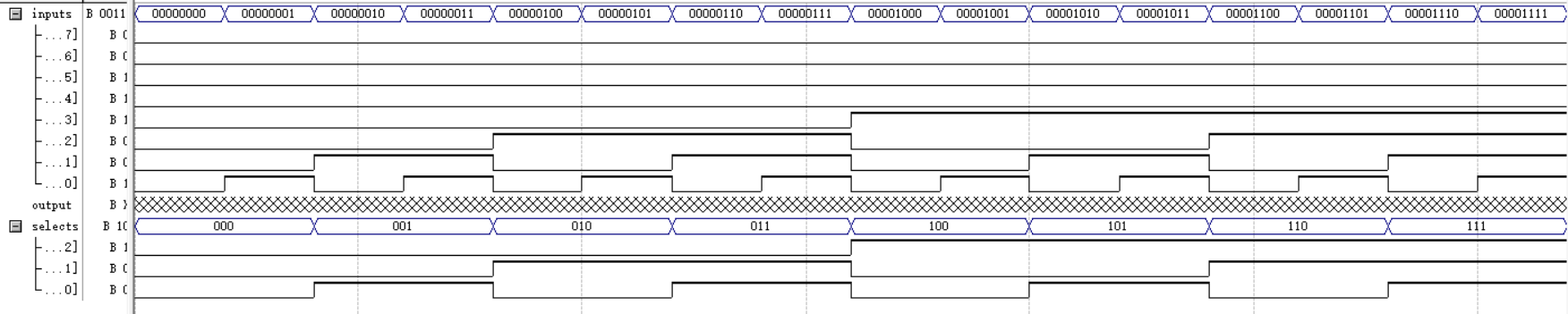
输入向量selinputs为11时，为无效的输入选择值，输出00000000。

其他任意周期均符合上述规律。

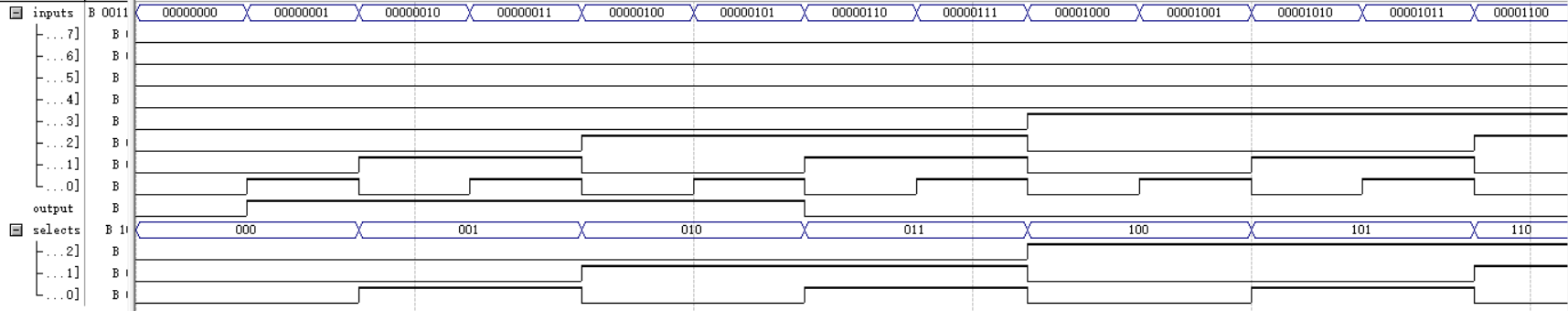
结果正确

* 调用类属实体实现的8-1多路复用器

a)波形仿真过程（过程详见实验步骤）



b)波形仿真波形图



c)结果分析及结论

以select输入从000变化到110的波形段为例。

select输入为000时,输出inputs[0]的值，在图中output的值先为0，然后变化为1

select输入为001时,输出inputs[1]的值，在图中output的值保持为1

select输入为010时,输出inputs[2]的值，在图中output的值保持为1

select输入为011时,输出inputs[3]的值，在图中output的值保持为0

select输入为100时,输出inputs[4]的值，在图中output的值保持为0

select输入为101时,输出inputs[5]的值，在图中output的值保持为0

select输入为110时,输出inputs[6]的值，在图中output的值保持为0

结果正确

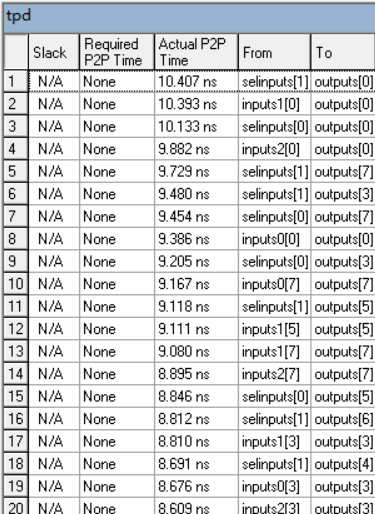
1. 时序仿真

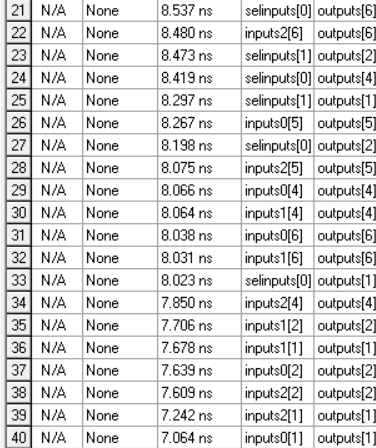
* 8重3-1多路复用器

1. 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）

1. 时序仿真图





1. 结果分析及结论

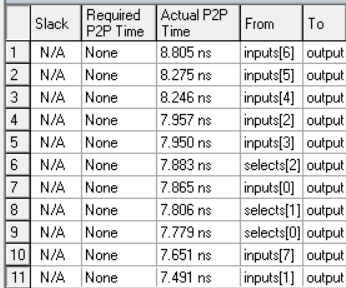
每个引脚只间相互传递产生的延时各不相同，挑选其中p2p时间的最大值，为selinputs[1]传递给outputs[0]。为10.407ns，故整体延时为10.407ns。

* 调用类属实体实现的8-1多路复用器

1. 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）

1. 时序仿真图



1. 结果分析及结论

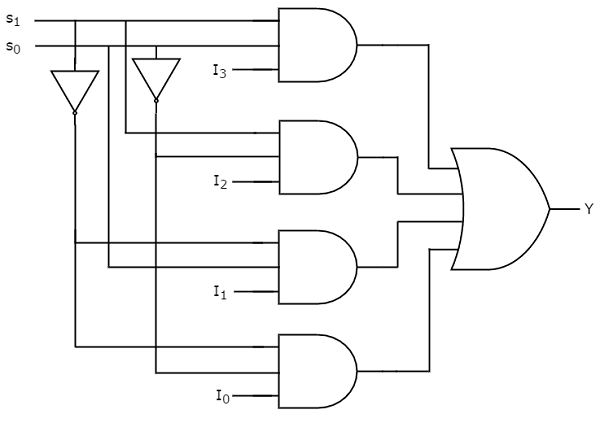
每个引脚只间相互传递产生的延时各不相同，挑选其中p2p时间的最大值，为inputs[6]传递给output。为8.805ns，故整体延时为8.805ns。

tpd (引脚到引脚的延时)

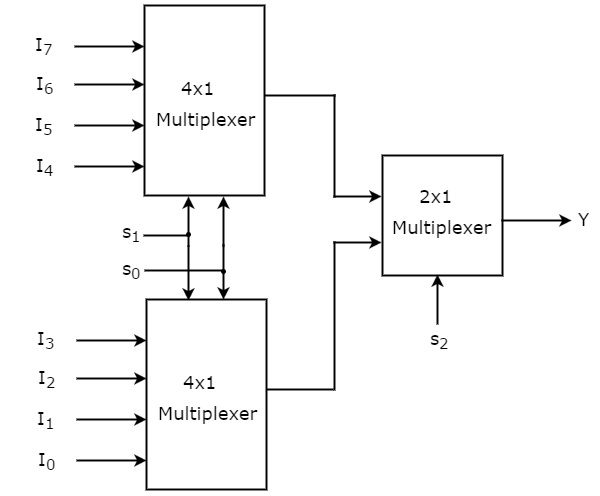
五、实验结论

1、思考题

1. 经查阅相关资料，得到多路复用的以下实现方式
2. 使用译码器和与或门实现多路复用器，如图4-2多路复用器



1. 使用多层多路复用器串接成一个更大的多路复用器，如图8-1多路复用器



1. 用VHDL语言的描述方法有：
2. 顺序语句描述

该种方法经常用到的语句有：

* when-else语句

1. if选择语句描述

该种方法经常用到的语句有：

* if-then-else语句

1. 并行语句描述

该种方法经常用到的语句有：

* 基本逻辑语句(and、not等)

以上三种方法均需使用的语句有

库引入、实体声明、端口方向、结构体、库，程序包的调用，进程语句等

1. 实验总结与实验心得

本次实验学习了用 VHDL 语言设计 8 重 3-1 多路复用器、设计 n-1 多路复用器，并调用该 n-1 多路复用器定制为 8-1多路复用器、设计 4 位行波进位加法器、设计 4 位先行进位加法器。需要的能力有VHDL的简单编程能力，VHDL的简单仿真能力，类属参数的设计能力，组合电路的设计与分析能力，实验报告的撰写能力，实验过程分析总结能力。通过使用软件实现硬件，既培养了编码能力，又增加了硬件设计能力和社会实践能力。该次实验使我对组合电路的设计与CPU中命令的传递与选择有了更深刻的认识，巩固了组合电路的设计相关的知识点。我因此受益良多。