

## 实验四 模型机时序部件的实现

班级 软件 1801 姓名 肖云杰 学号 201826010113

### 一、实验目的

1. 熟悉计数器、寄存器和 RAM 的工作原理。
2. 了解模型机中 SM 的作用。
3. 学会使用 VHDL 语言设计时序电路

### 二、实验内容

- 1、用 VHDL 语言设计 SM;
- 2、用 VHDL 语言设计一个 8 位的指令计数器 PC;
- 3、用 VHDL 语言设计 3 个 8 位寄存器组成的寄存器组, 实现读写操作。
- 4、用 LPM\_RAM\_IO 定制一个 256\*8 的 RAM, 实现对 RAM 的读写操作;

### 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

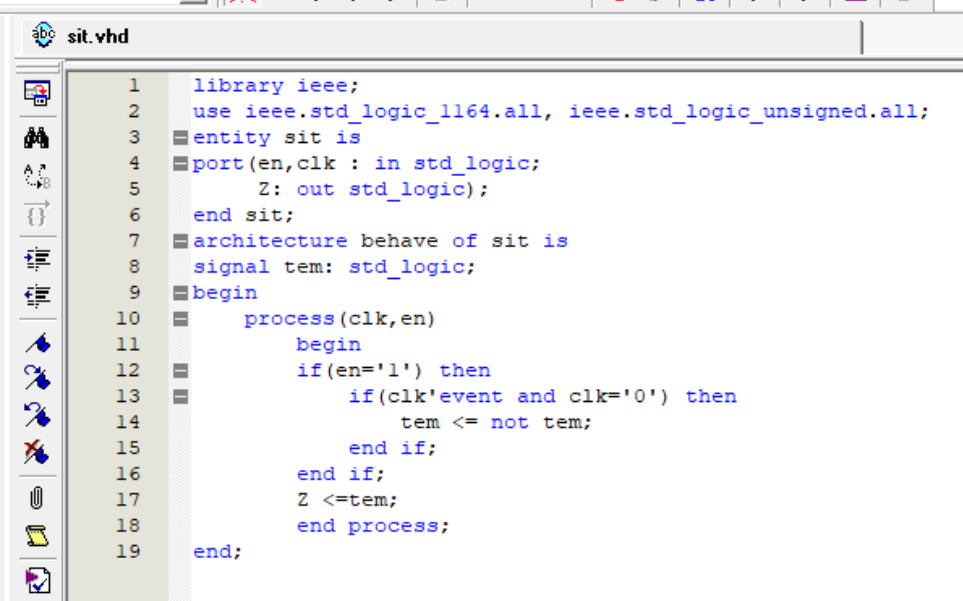
- 1、新建, 编写源代码。
  - (1).选择保存项和芯片类型: 【File】 - 【new project wizard】 - 【next】 (设置文件路径+设置 project name 为 xor2) - 【next】 - 【properties】 (type=VHDL) - 【next】 - 【next】 - 【finish】
  - (2).新建: 【file】 - 【new】 (第二个 VHDL File) - 【OK】
- 2、写好源代码, 保存文件。
- 3、编译与调试。确定源代码文件为当前工程文件, 点击 【processing】 - 【start compilation】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。按照程序所述插入输入节点。然后 【start simulation】, 出输出节点的输出图。
- 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

#### SM

#### 1、编译过程

a) 源代码如图 (VHDL 设计)



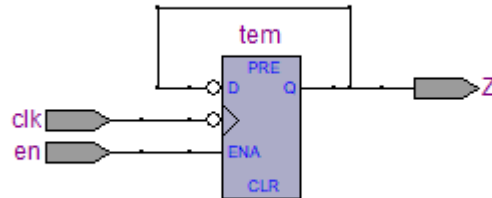
```
1 library ieee;
2 use ieee.std_logic_1164.all, ieee.std_logic_unsigned.all;
3 entity sit is
4 port(en,clk : in std_logic;
5       Z: out std_logic);
6 end sit;
7 architecture behave of sit is
8 signal tem: std_logic;
9 begin
10    process(clk,en)
11    begin
12        if(en='1') then
13            if(clk'event and clk='0') then
14                tem <= not tem;
15            end if;
16        end if;
17        Z <=tem;
18    end process;
19 end;
```

b)编译、调试过程

共有五个警告，没有报错

5个警告都为管脚未分配导致的警告，程序没有问题。

c) RTL 视图



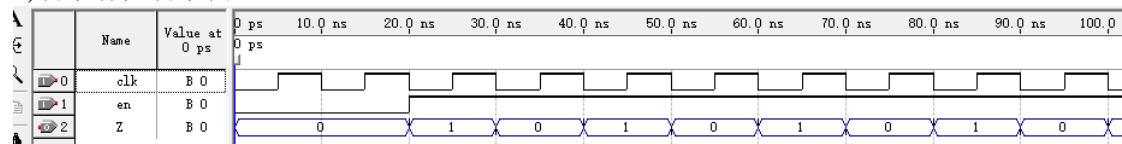
d)结果分析及结论

程序设计正常，没有错误

## 2、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

0-20ns: 使能端 en 为低电平，无效，因此，Z 值不变，模块没有问题

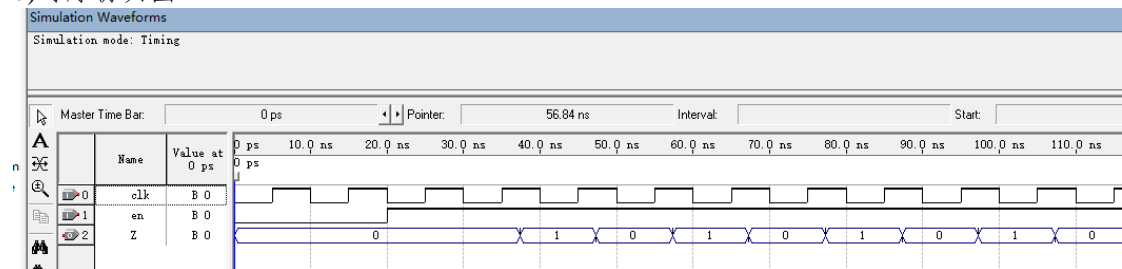
20-100ns: 使能端 en 为高电平，输出有效，因此，在每个时钟下降沿，数据进行取反操作

## 3、时序仿真

a) 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



b) 结果分析及结论

观察到信号有 8ns 左右的延时，为引脚之间传输的延迟。

## 8 位的指令计数器 PC;

### 1、编译过程

a) 源代码如图（VHDL 设计）

```

1  library ieee;
2  use ieee.std_logic_1164.all, ieee.std_logic_unsigned.all;
3
4  entity pcjq is
5  port(ld, inc, clk : in std_logic;
6        a : in std_logic_vector(7 downto 0);
7        c : out std_logic_vector(7 downto 0));
8  end pcjq;
9
10 architecture behave of pcjq is
11   signal tem : std_logic_vector(7 downto 0);
12 begin
13   process(ld, inc, clk)
14   begin
15     if (clk'event and clk = '0') then
16       if inc = '1' then
17         if tem = "11111111" then
18           tem <= "00000000";
19         else
20           tem <= tem + 1;
21         end if;
22       elsif ld = '1' then
23         tem <= a;
24       end if;
25     end if;
26   end process;
27   c <= tem;
28 end;

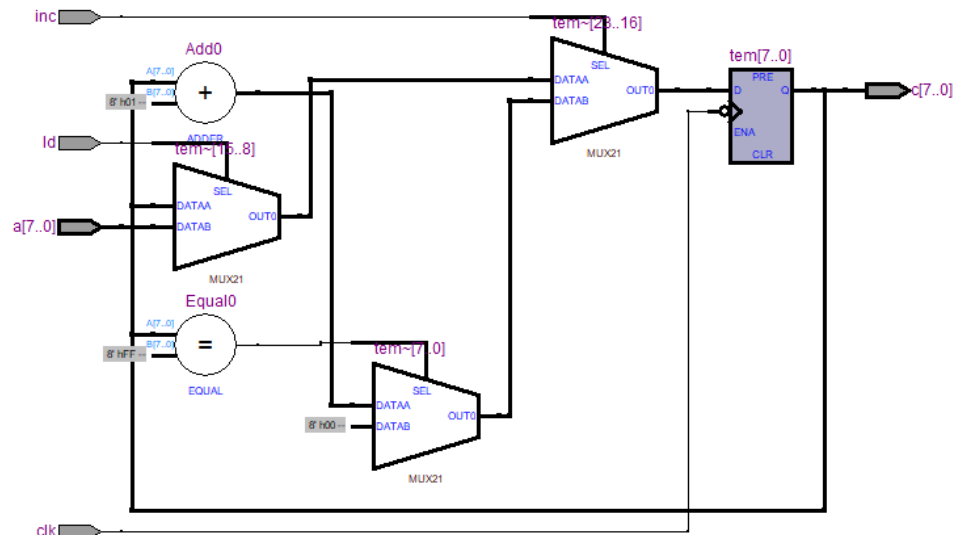
```

b) 编译、调试过程

共有五个警告，没有报错

5 个警告都为管脚未分配导致的警告，程序没有问题。

c) RTL 视图



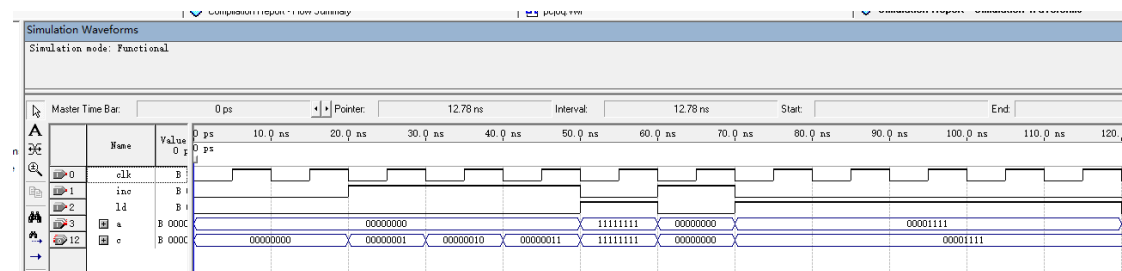
d) 结果分析及结论

程序设计正常，没有错误

## 2、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

0-20ns: 两个功能选择都为低电平，无功能，因此，输出不变，模块没有问题

20-50ns: inc 为高电平，功能为自增一，因此，在每个时钟下降沿，数据进行自增 1

50-60ns: 设置为极限数据

60-70ns: 判断溢出归零操作

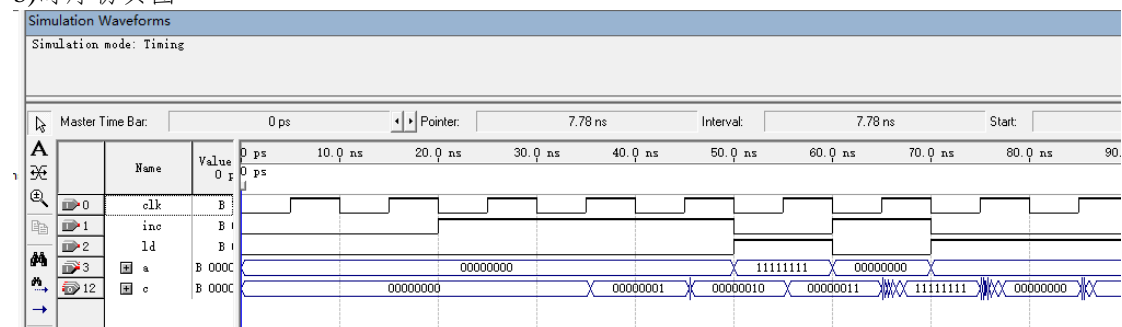
70-100ns: ld 为 1，为置值功能，测试正确。

### 3、 时序仿真

c) 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



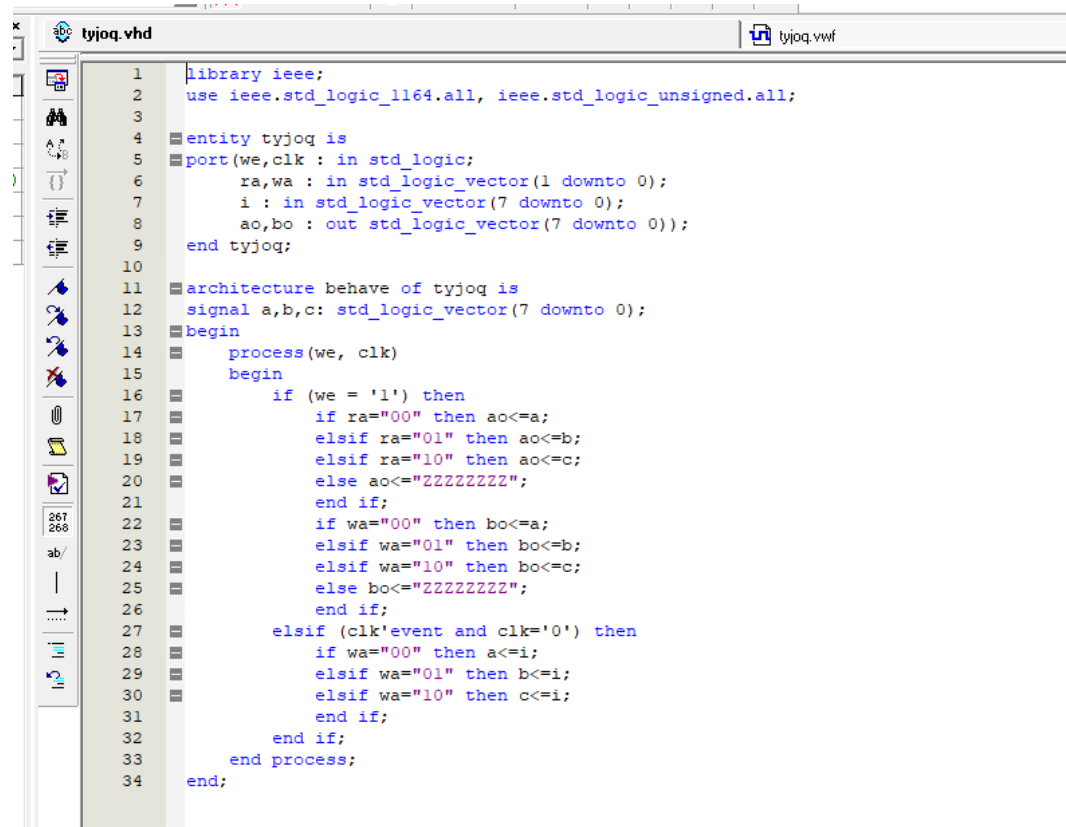
d) 结果分析及结论

观察到信号有 6ns 左右的延时，为引脚之间传输的延迟。

## 通用寄存器组

### 4、 编译过程

a) 源代码如图（VHDL 设计）



```
1 library ieee;
2 use ieee.std_logic_1164.all, ieee.std_logic_unsigned.all;
3
4 entity tyjq is
5 port (we, clk : in std_logic;
6       ra, wa : in std_logic_vector(1 downto 0);
7       i : in std_logic_vector(7 downto 0);
8       ao, bo : out std_logic_vector(7 downto 0));
9 end tyjq;
10
11 architecture behave of tyjq is
12 signal a, b, c: std_logic_vector(7 downto 0);
13 begin
14 process (we, clk)
15 begin
16     if (we = '1') then
17         if ra="00" then ao<=a;
18         elsif ra="01" then ao<=b;
19         elsif ra="10" then ao<=c;
20         else ao<="ZZZZZZZZ";
21         end if;
22         if wa="00" then bo<=a;
23         elsif wa="01" then bo<=b;
24         elsif wa="10" then bo<=c;
25         else bo<="ZZZZZZZZ";
26         end if;
27     elsif (clk'event and clk='0') then
28         if wa="00" then a<=i;
29         elsif wa="01" then b<=i;
30         elsif wa="10" then c<=i;
31         end if;
32     end if;
33 end process;
34 end;
```

## b) 编译、调试过程

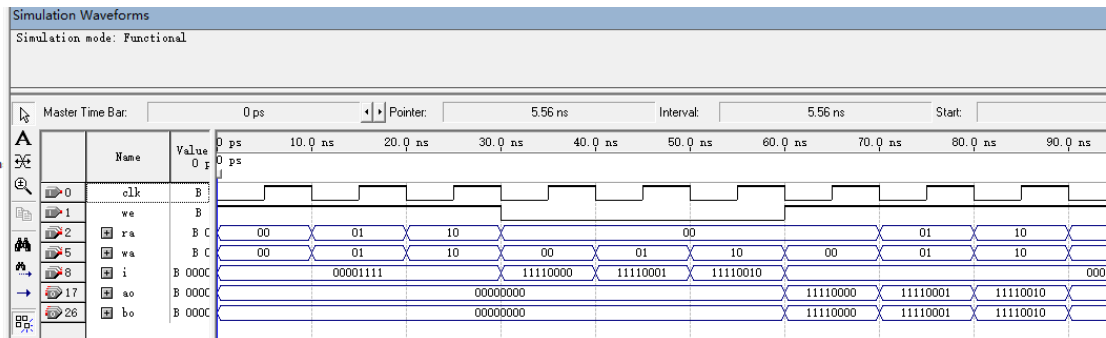
共有 33 个警告，没有报错

33 个警告都为管脚未分配导致的警告，程序没有问题。

## c) RTL 视图

程序设计正常，没有错误

### b) 波形仿真波形图



## c) 结果分析及结论

0-10ns: we 为 1, 选择输出, 此时选择的为 A 寄存器的值, 都为 00000000

10-20ns: we 为 1, 选择输出, 此时选择的为 B 寄存器的值, 都为 00000000

20-30ns: we 为 1, 选择输出, 此时选择的为 C 寄存器的值, 都为 00000000

40-60ns: we 为 0, 按照 wa, 将 11110000, 11110001, 11110010 分别赋值给三个寄存器。

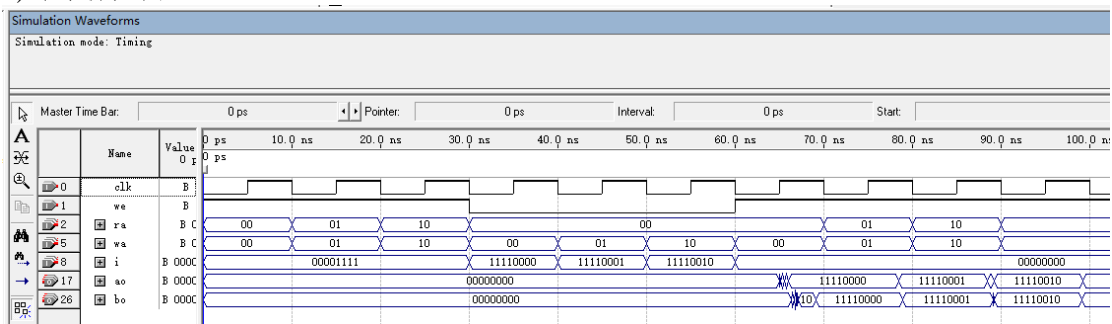
60-90ns: we 为 1, 再次从三个寄存器取值, 观察, 正确。

## 6、时序仿真

## e) 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



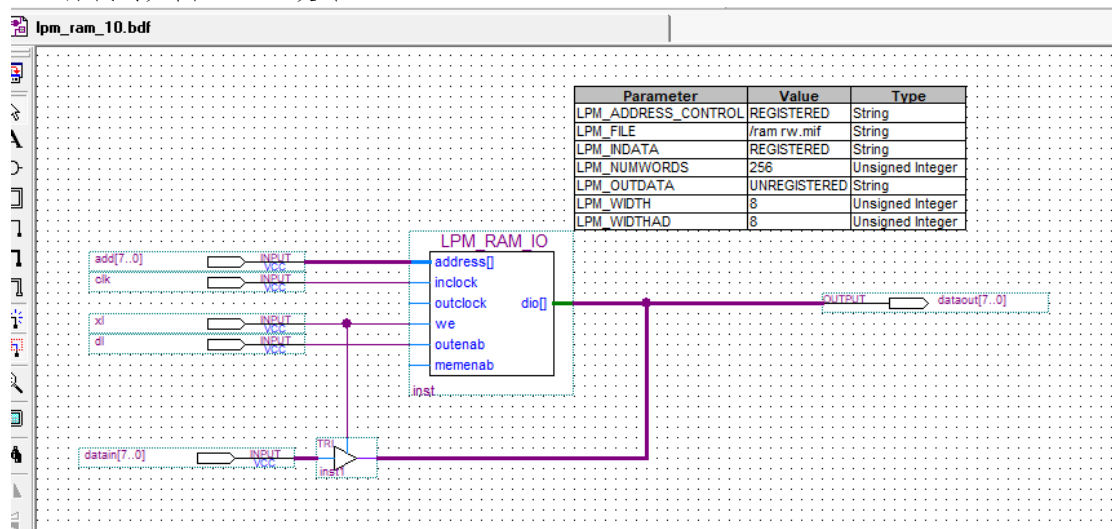
## f) 结果分析及结论

观察到信号有 6ns 左右的延时, 为引脚之间传输的延迟。

## RAM

## 7、编译过程

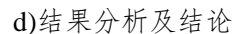
## a) 源代码如图 (BDF 设计)



## b) 编译、调试过程

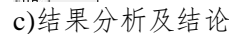
共有 15 个警告, 没有报错

c) RTL 视图



## 8、波形仿真

### b)波形仿真波形图



10-40ns: we (xl) 为 1, 写入, 将 00000001 写入 00000001 地址, 将 00000010 写入 00000010 地址, 将 00000100 写入 00000100 地址。

## 9、 时序仿真

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)

### h) 结果分析及结论

第 8 页 共 9 页



## 五、思考题

1、时钟周期的上升沿实现对 RAM 的读写操作，为何 PC、SM 以及寄存器的操作是下降沿完成？

一个操作在一个周期完成，因此上升沿取出，下降沿进行操作

2、总结 VHDL 语言描述时序部件的方法和常用语句。

Process

If elsif then end if;

敏感信号相关语句

Case when

## 六、实验结论

在本次实验中，开始时很多不懂，缺少了很多细节。比如，初始值的设置等等

在lpm定制化时，遇到了困难，因为不了解，第一次接触

在不断错误的尝试中，我成功解决了问题，百度csdn和问同学，在各种方法中，找到最好的方法，成功做出了实验。

不过因为部分接口、导线命名不规范，导致我很多时候需要查表才能做出相应功能，下次实验应当注意。