

浙江大学



数字电路分析与设计 实验报告

实验名称 四位二进制加法器设计

姓名 小黄

学号 3100100000

实验地点 浙江大学紫金港校区

实验日期 2020 年 5 月 25 日

目录

1 实验目的	1
2 实验内容	1
3 实验原理	1
3.1 奇偶位判断电路	1
3.2 全加器	1
3.3 4 位串行进位二进制全加器	2
4 简单的二位与非门实现	2
4.1 用原理图实现	3
4.2 用 VHDL 实现	3
5 奇偶位判断实验	4
5.1 原理图实现	4
5.2 底层 xor4d 的代码	4
5.3 仿真结果	6
6 4 位全加器实验	6
6.1 创建 4 位串行二进制全加器原理图	6
6.2 创建 1 位二进制全加器的 VHDL 源文件	7
6.3 4 位二进制全加器仿真测试	9

插图

1 实验原理图	2
2 原理图实现 nand2	3
3 nand2 仿真结果 1	3
4 nand2 仿真结果 2	4
5 xor8d 原理图实现	4

6	xor8d 仿真结果	6
7	adder4 原理图实现	6
8	adder4 仿真结果	9

1 实验目的

- 熟悉 Quartus II 软件的使用
- 掌握逻辑功能的 VHDL 语言描述和原理图描述的方法
- 进一步掌握四位串行二进制加法器的设计方法
- 掌握用仿真波形验证电路功能的方法

2 实验内容

- 利用原理图实现简易与非门电路
- 利用 VHDL 实现与非门电路
- 利用原理图和 VHDL 实现奇偶位判断电路
- 用原理图方式描述 4 位全加器的功能
- 通过波形仿真验证 4 位全加器的功能

3 实验原理

3.1 奇偶位判断电路

判断 1 的个数，奇数个 1 输出 1，偶数个 1 输出 0。对于 4 位二进制奇偶位判断电路，利用卡诺图可以得到，若输入为 $ABCD$ ，则输出应为 $A \oplus B \oplus C \oplus D$ ，可以用两个这样的电路拓展到八位奇偶位判断。利用原理图的方式用两个 4 位异或电路实现 8 位异或，然后底层的 4 位异或利用 VHDL 实现。

3.2 全加器

通过列出卡诺图我们可以得到全加器的输入和输出：

$$S = A \oplus B \oplus C_{i-1}$$

$$C_i = AB + BC_{i-1} + AC_{i-1}$$

3.3 4 位串行进位二进制全加器

4 位串行进位二进制全加器以 1 位全加器的设计为基础, 将四个 1 位二进制全加器串接即可构成四位二进制全加器;

在使用 Quartus 构建时, 顶层采用原理图描述, 底层采用 VHDL 语言描述,

串行连接所依据的式子如下 (由于实验中标号从 1 开始的, 这里的原理也按下标从 1 开始)

$$\begin{array}{r}
 A_4 A_3 A_2 A_1 \\
 + \quad B_4 B_3 B_2 B_1 \\
 \text{进位} \quad C_3 \ C_2 \ C_1 \\
 \hline
 C_4 S_4 S_3 S_2 S_1
 \end{array}$$

实验原理图如下:

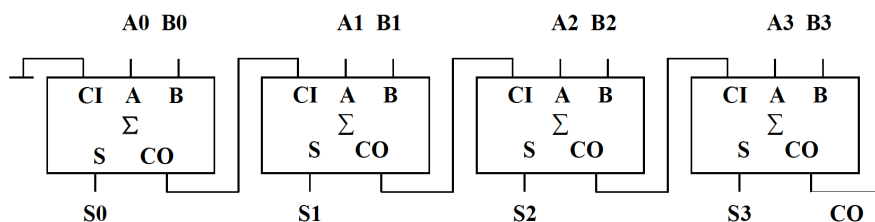


图 1: 实验原理图

4 简单的二位与非门实现

本实验的目的是熟悉 Quartus 软件。

4.1 用原理图实现

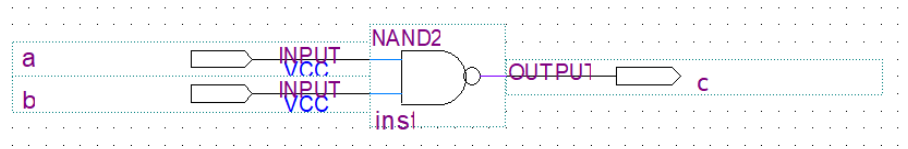


图 2: 原理图实现 nand2

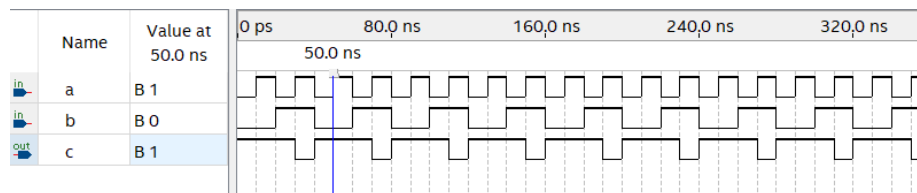


图 3: nand2 仿真结果 1

4.2 用 VHDL 实现

```
1  -- nand2
2  library ieee;
3  use ieee.std_logic_1164.all;
4  entity nand2_3 is
5  port(a,b:in std_logic; z:out std_logic);
6  end nand2_3;
7  architecture a1 of nand2_3 is
8  begin
9      z<=not(a and b);
10     end a1;
```

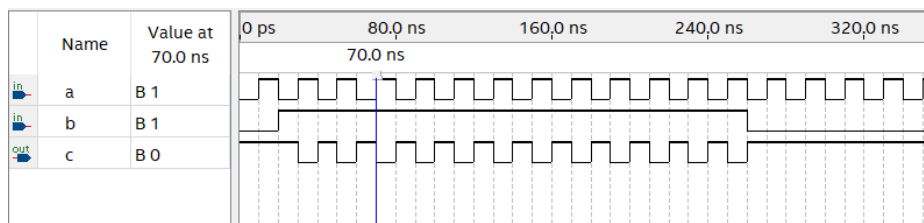


图 4: nand2 仿真结果 2

可见，我们实现了预期与非的功能（只有 11 输出 0）。

5 奇偶位判断实验

5.1 原理图实现

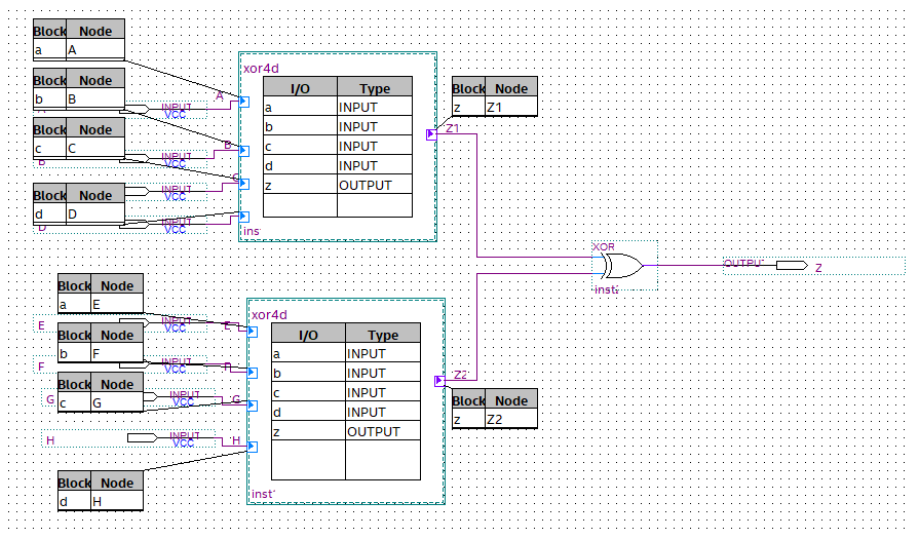


图 5: xor8d 原理图实现

5.2 底层 xor4d 的代码

```

1  -- xor4d
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4
5
6  -- Entity Declaration
7
8  ENTITY xor4d IS
9  -- {{ALTERA_IO_BEGIN}} DO NOT REMOVE THIS LINE!
10 PORT
11 (
12  a : IN STD_LOGIC;
13  b : IN STD_LOGIC;
14  c : IN STD_LOGIC;
15  d : IN STD_LOGIC;
16  z : OUT STD_LOGIC
17 );
18 -- {{ALTERA_IO_END}} DO NOT REMOVE THIS LINE!
19
20 END xor4d;
21
22
23 -- Architecture Body
24
25 ARCHITECTURE xor4d_architecture OF xor4d IS
26
27
28 BEGIN
29  z<= a xor b xor c xor d;
30
31 END xor4d_architecture;

```


5.3 仿真结果

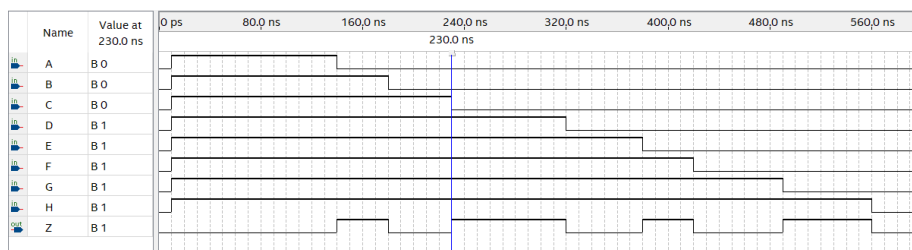


图 6: xor8d 仿真结果

可见，我们实现了预期奇偶位判断的功能：奇数个 1 输出 1，否则输出 0。

6 4 位全加器实验

6.1 创建 4 位串行二进制全加器原理图

在 Quartus 中的连接图如下：

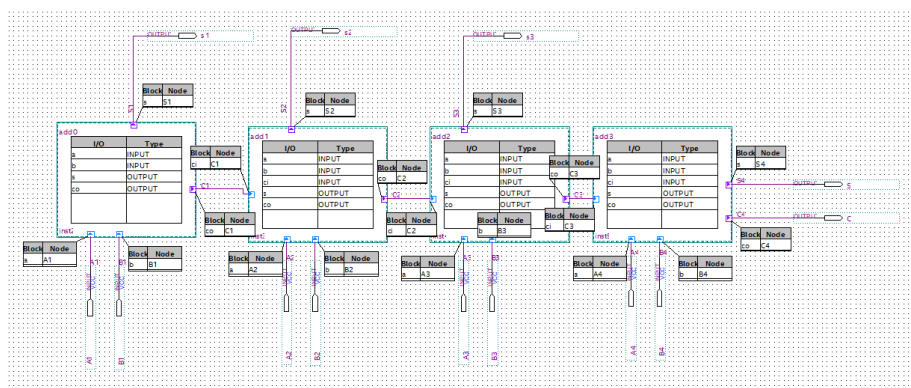


图 7: adder4 原理图实现

6.2 创建 1 位二进制全加器的 VHDL 源文件

实验代码如下。

```
1  -- add0
2
3  LIBRARY ieee;
4  USE ieee.std_logic_1164.all;
5
6  -- Entity Declaration
7
8  ENTITY add0 IS
9  -- {{ALTERA_IO_BEGIN}} DO NOT REMOVE THIS LINE!
10 PORT
11 (
12  a : IN STD_LOGIC;
13  b : IN STD_LOGIC;
14  s : OUT STD_LOGIC;
15  co : OUT STD_LOGIC
16 );
17 -- {{ALTERA_IO_END}} DO NOT REMOVE THIS LINE!
18
19 END add0;
20
21
22 -- Architecture Body
23
24 ARCHITECTURE add0_architecture OF add0 IS
25
26
27 BEGIN
28  s<=a xor b;
29  co <= a and b;
30 END add0_architecture;
31
32
```

```

33 -- add1 add2 add3
34 LIBRARY ieee;
35 USE ieee.std_logic_1164.all;
36
37
38 -- Entity Declaration
39
40 ENTITY add1 IS
41 -- {{ALTERA_IO_BEGIN}} DO NOT REMOVE THIS LINE!
42 PORT
43 (
44 a : IN STD_LOGIC;
45 b : IN STD_LOGIC;
46 ci : IN STD_LOGIC;
47 s : OUT STD_LOGIC;
48 co : OUT STD_LOGIC
49 );
50 -- {{ALTERA_IO_END}} DO NOT REMOVE THIS LINE!
51
52 END add1;
53
54
55 -- Architecture Body
56
57 ARCHITECTURE add1_architecture OF add1 IS
58
59
60 BEGIN
61 s <= a xor b xor ci;
62 co <= (a and b) or (b and ci) or (a and ci);
63 END add1_architecture;

```

6.3 4 位二进制全加器仿真测试

1. 新建一个仿真波形文件
2. 添加需要仿真的输入、输出信号
3. 给输入信号设置仿真激励
4. 点击工具栏中的图标启动波形仿真过程
5. 观察仿真结果

仿真结果如下:

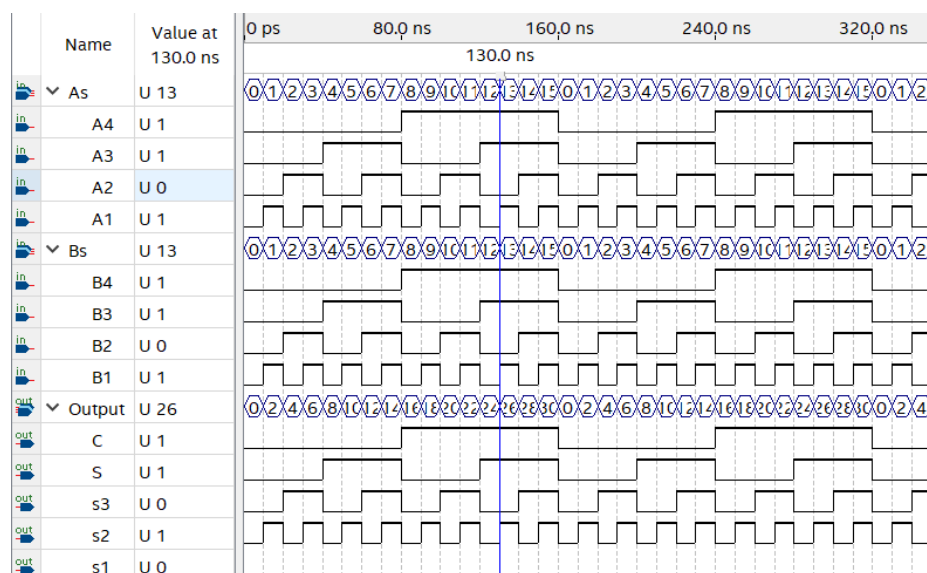


图 8: adder4 仿真结果

可见，我们实现了 4 位二进制加法！