**北京科技大学实验报告**

学院： 自动化学院 专业： 测控技术与仪器 班级： 测控192

姓名： 张正阳 学号： 41827166 实验日期： 2021年 12月 07 日

**实验名称： 实验一：组合逻辑电路设计**

**实验目的：**

1．熟悉 Vivado 2017.2 的编译环境；

2．了解在 Vivado 2017.2 环境下运用 Verilog HDL 语言的编程开发流程，包括源程序的输入、编译、模 拟仿真及程序下载

**实验仪器：**

FPGA开发板、Vivado2018开发工具

**实验内容与步骤：**

1. 简单门电路的实现；

编写代码实现一个简单的非门电路：

电路逻辑表达式

具体操作步骤如下：

(1) 利用向导，建立一个新项目，工程名为 expe1，顶层文件名也指定为 not\_lab。

(2) 新建一个 Verilog HDL 文件。

(3) 编写testbench文件，进行时序仿真。

(4) 给出时序图

在 Verilog HDL 文件中输入程序：

`timescale 1ns / 1ps

1. module not\_lab(
2. input in,
3. output out
4. );
5. assign out = ~in;
6. endmodule

编写testbench程序，如下：

`timescale 1ns / 1ps

1. module not\_lab\_tb();
2. // testbench 时钟信号
3. reg clk = 0;
4. always #10 clk <= ~clk;
5. // 输出信号
6. wire out;
7. // 调用test模块
8. Not\_lab mytest(clk, out);
9. endmodule

1. 四选一数据选择器；

四选一数据选择器的实现：数据选择器又被称为多路选择器或多路开关，它是常用组合逻辑部件之一。数据选择器的功能是在选择控制端的控制下，从多个数据输入中选择一个并将其送到一个公共的输出端。四选一数据选择器输出的逻辑函数如下：

数据选择器的电路结构一般由与或门阵列组成，也有用传输门开关和门电路混合而成的。四选一数据选择器电路模块含有四个数据输入端，一个数据输出端和两个地址选择输入端构成的。

具体操作步骤如下：

(1) 利用向导，建立一个新项目，工程名为 expe2，顶层文件名也指定为 mux4\_1。

(2) 新建一个 Verilog HDL 文件。

(3) 编写testbench文件，进行时序仿真。

(4) 给出时序图

在 Verilog HDL 文件中输入四选一数据选择器程序：

`timescale 1ns / 1ps

module mux4\_1(out,in0,in1,in2,in3,sel);

input in0,in1,in2,in3;

input [1:0] sel;

output out;

reg out;

always @(sel or in0 or in1 or in2 or in3)

begin

case(sel)

2'b00: out=in0;// 若地址选择输入为00， 则out输出为in0

2'b01: out=in1;

2'b10: out=in2;

2'b11: out=in3;

default: out=1'b0;//确保不存在未赋值的分支

endcase

end

endmodule

编写testbench文件:

`timescale 1ns / 1ps

module mux4\_1\_tb();

reg[3:0] in;

reg[1:0] sel\_in;

wire f;

mux4\_1 mux4\_1(f,in[0],in[1],in[2],in[3],sel\_in);//实例化待测模块

always #150 sel\_in=sel\_in+1;

always #50 in=in+1;

initial begin

in=0; sel\_in=0;

#1200 $finish;

end

endmodule

1. 全加器的实现

全加器其实就是考虑到进位的加法器。一位全加器的电路符号如图1.1 所示，真值表如表1.1 所示。

|  |  |  |
| --- | --- | --- |
| A | 全 加 器 | **Si** |
| B |
| Cin | Cout |
|  |
|  |

图1.1 全加器电路符号

### 表1.1 一位全加器真值表

全加器输入

全加器输出

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| **A** | **B** | **Cin** | **Si** | **Cout** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

源程序如下：（四位全加器）

`timescale 1ns / 1ps

module full\_adder(cout, si,ai,bi,cin);

output si;

output cout;

input ai,bi;

input cin;

assign{cout,si}=ai+bi+cin;

endmodule

testbench程序如下：

`timescale 1ns / 1ps

module full\_adder\_tb( );

reg [2:0] in;

wire sum;

wire cout;

full\_adder FA(.cout(cout),.si(sum),.ai(in[2]),.bi(in[1]),.cin(in[0]));//实例化待测模块

always #10 in=in+1;

initial begin

in=0;

#160 $finish;

end

endmodule

1. 编码器的实现

编码器通常分为两大类：普通编码器和优先编码器。其中普通编码其就是对某一个给定时刻只能对 一个输入信号进行编码的编码器，它的输入端口不允许同一时刻出现两个以上的有效输入信号；优先编 码器就是对某一个给定时刻只对优先级最高的输入信号进行编码的编码器，它的输入端口允许多个输入 信号同时有效。

现以八至三线编码器为例，介绍普通编码器的 Verilog HDL 语言程序设计。通常，八至三线编码器 的逻辑电路符号如图 1.2 所示，真值表如表1.2 所示。不难看出该编码器的工作原理为：编码器将对八个输入信号进行编码操作，然后以三位二进制码的形式输出，这里输入信号为低电平有效

D0

|  |  |  |
| --- | --- | --- |
|  | 8-3 线  编码器 | Q0 |
| D1 |
| D2 |
| Q1 |
| D3 |
| D4 |
| Q2 |
| D5 |
| D6 |
|  |
| D7 |
|  |

图 1.2 八至三线编码器的电路符号

**表 1.2 八至三线编码器的真值表**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Q2 | Q1 | Q0 |
| 0  1  1  1  1  1  1  1 | 1  0  1  1  1  1  1  1 | 1  1  0  1  1  1  1  1 | 1  1  1  0  1  1  1  1 | 1  1  1  1  0  1  1  1 | 1  1  1  1  1  0  1  1 | 1  1  1  1  1  1  0  1 | 1  1  1  1  1  1  1  0 | 1  1  1  1  0  0  0  0 | 1  1  0  0  1  1  0  0 | 1  0  1  0  1  0  1  0 |

具体操作过程如下：

（1） 利用项目向导，建立一个新项目，顶层文件名指定为 encoder8\_3。

（2） 新建一个 Verilog HDL 文件，并输入源程序：

module encoder8\_3(q,d);

input[7:0] d;

output[2:0] q;

reg[2:0] q;

always@(d) begin

case(d)

8'b01111111: q<=3'b111;

8'b10111111: q<=3'b110;

8'b11011111: q<=3'b101;

8'b11101111: q<=3'b100;

8'b11110111: q<=3'b011;

8'b11111011: q<=3'b010;

8'b11111101: q<=3'b001;

8'b11111110: q<=3'b000;

default: q<=3'bzzz;

endcase end

endmodule

a． 对源程序进行语法检查并编译。

b． 对项目进行时序逻辑仿真。

c． 有兴趣的同学还可以尝试分配管脚，生成\*.bit 文件。（管脚分配可参照实如下表）

d． 下载。

**实验数据：**

1. 简单门电路的实现
2. 四选一数据选择器；
3. 全加器的实现
4. 编码器的实现

**实验数据处理：**

**实验结果与分析：**

**实验名称：**

**实验目的：**

**实验仪器：**

**实验原理：**

**实验内容与步骤：**

**实验数据：**

**实验数据处理：**

**实验结果与分析：**