**实验报告 Lab 1**

**任务1** 设计一个2-1数据选择器，将输入端口a，b，s连接到三个开关，将输出端口y连接到一个LED上。在FGPA板上实现该电路，并测试验证其能否正常工作。

（1）使用门级描述

（2）使用数据流描述

（3）使用行为描述（if-else）

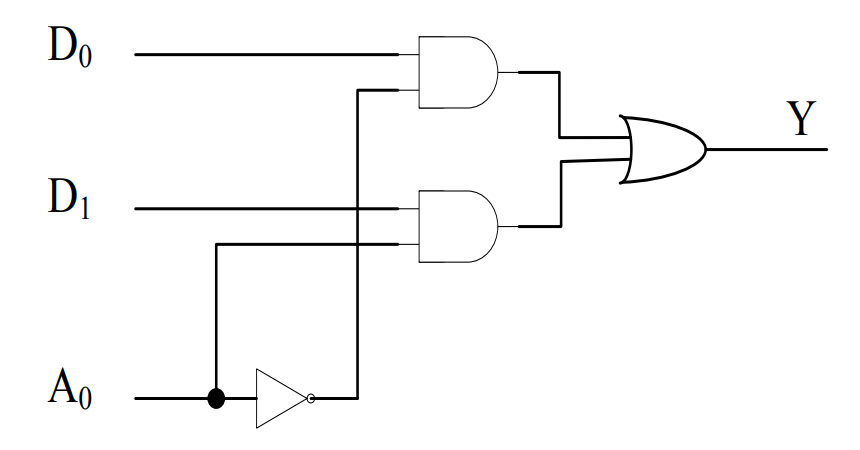
（4）使用行为描述（case）

**任务2** 使用刚才设计的2-1数据选择器设计一个4-1数据选择器。

列出2-1选择器的真值表，根据真值表写出逻辑表达式并化简：

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | S | Y |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |

很容易得到如下的门电路：



据此便可写出门级描述。要注意：

（1）要声明中间变量，即除了输入输出之外在各组件间的连线端口；

（2）调用逻辑门时先写输出，再写输入

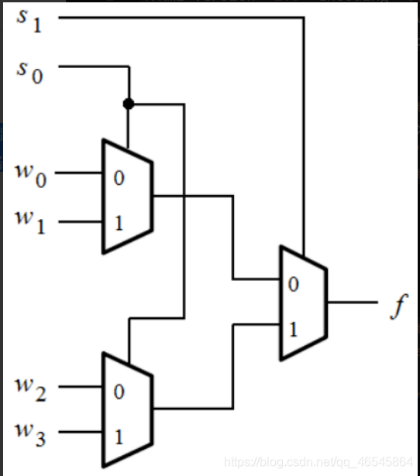
接下来可以写数据流描述了。数据流描述多用于描述组合逻辑电路，使用assign，简单粗暴，直接使用布尔表达式赋值。这里要注意逻辑运算的优先级，某些地方需要加括号。

最后写行为描述。只需描述清楚输入和输出信号的行为,而不去关心输入输出中间具体发生了什么。注意y在always语句里被赋值，要定义成reg y。

将写好的程序经综合、实现、比特流生成并写入板子，板子现象如下：

|  |  |
| --- | --- |
|  |  |
| P5=0,P4=0,N4=0;F6=0 | P5=1,P4=0,N4=0;F6=1 |
|  |  |
| P5=1,P4=0,N4=1;F6=0 | P5=0,P4=1,N4=1;F6=1 |

至此任务1完成。

要想设计4-1数据选择器，可以通过三个2-1数据选择器组合而成。4路输入，先选择13还是24，再在1和3、2和4中做出选择。同时后者的选择线可以合并，因为我们最终只需要一个输入。总有一组输入不会被输出，其值是多少无所谓。接下来只需要确认好接线即可。据此我们可以写出4-1选择器的代码了，直接调用先前写的2-1选择器即可。不过要特别注意符号，稍不注意就会各种报错各种跑不通。

将写好的程序经综合、实现、比特流生成并写入板子，板子现象如下：

|  |  |
| --- | --- |
|  |  |
| P5=1,P4=0,P3=0,P2=0,M4=0,N4=0 | P5=0,P4=1,P3=0,P2=0,M4=0,N4=1 |
|  |  |
| P5=0,P4=0,P3=1,P2=0,M4=1,N4=0 | P5=0,P4=0,P3=0,P2=1,M4=1,N4=1 |

至此任务2完成。

**附：代码**

二选一数据选择器代码：

module mux2(A,B,sel,Y);

input A,B,sel;

output Y;

wire Y1,Y2,notsel; //门级描述

and a1(Y1,A,notsel);

and a2(Y2,B,sel);

not(notsel,sel);

or(Y,Y1,Y2);

endmodule

module mux2(A,B,sel,Y);

input A,B,sel;

output Y;

assign Y= (sel & B)| ((~sel) & A); //数据流描述

endmodule

module mux2(A,B,sel,Y);

input A,B,sel;

output Y;

reg Y;

always @(\*)//行为描述（if-else）

begin

if(sel==0) Y=A;

else Y=B;

end

endmodule

module mux2(A,B,sel,Y);

input A,B,sel;

output Y;

reg Y;

always @(\*)//行为描述（case）

case (sel)

0: begin

Y=A;

end

1: begin

Y=B;

end

endcase

endmodule

四选一数据选择器代码：

mux2代码同上，mux4代码如下：

module mux4(

input A1,

input A2,

input A3,

input A4,

input S1,

input S2,

output Y0

);

wire Y1,Y2;

mux2 mux21(.A(A1),.B(A2),.sel(S1),.Y(Y1));

mux2 mux22(.A(A3),.B(A4),.sel(S1),.Y(Y2));

mux2 mux23(.A(Y1),.B(Y2),.sel(S2),.Y(Y0));

endmodule