# 数字电路与数字系统实验

**EX02:译码器和编码器的设计**

191220029 傅小龙

周一5-6节班

[1830970417@qq.com](mailto:1830970417@qq.com)

2020年9月14日

# **目录**

1. [实验内容](#_实验内容)……………………………………………………………………………………3
   1. [实验要求](#_1.1实验要求)………………………………………………………………………………3
   2. [实验工具](#_1.2实验工具)………………………………………………………………………………3
2. [实验过程](#_实验过程)……………………………………………………………………………………4

2.1 [模型概述](#_2.1模型概述)………………………………………………………………………………4

2.2 [数字抽象](#_2.2数字抽象)………………………………………………………………………………4

2.3 [建立模型](#_2.3建立模型)………………………………………………………………………………5

2.4 [分析/综合](#_2.4分析/综合)…………………………………………………………………………… 9

2.5 [仿真测试](#_2.5仿真测试)………………………………………………………………………………10

2.6 [分配引脚](#_2.6分配引脚)………………………………………………………………………………11

2.7 [全编译](#_2.7全编译)…………………………………………………………………………………13

三、[实验总结](#_实验总结)……………………………………………………………………………………14

四、[附](#_附)……………………………………………………………………………………………14

4.1[实验中遇到的问题及处理办法](#_4.1实验中遇到的问题及处理办法)………………………………………………………14

4.2[关于思考题](#_4.2 关于思考题)……………………………………………………………………………14

#### 实验内容

##### 1.1实验要求

实现一个 8-3 优先编码器，完成 8-3编码器的设计、功能仿真和硬件实现。使用拨动开关，SW7—SW0 随机输入一个 8 位二进制值，对此 8 位二进制数进行高位优先编码成一个 3 位二进制值，并根据是否有输入增加一位输入指示位，即 8 个开关全 0 时指示位为 0，有任何一个开关为 1 时指示位为 1。将此编码结果以二进制形式显示在四个发光二极管上。再将此结果跟据七段数码管的显示进行译码，将二进制的优先编码结果以十进制的形式显示在数码管上。编码器的使能端可选实现。

##### 1.2实验工具

软件环境：

设计、编译、仿真：Quartus Prime Version 17.1.0 Build 590 10/25/2017 SJ Lite Edition

DE10\_Standard\_SystemBuilder

硬件环境： DE-10 Standard开发平台

FPGA芯片： Cyclone V 5CSXFC6D6F31C6

#### 实验过程

##### 2.1模型概述

使用 Verilog HDL 实现一个带使能端的8-3优先编码器和七段数码管编码器。当译码器的输入有效时（非全0且使能信号有效），使一LED灯亮起。七段数码管编码器将优先译码器的二进制译码结果进行编码，编码结果作为七段数码管的输入以显示优先译码器的译码结果。

##### 2.2数字抽象

1. 8-3优先编码器
2. 输入:

使能端：en

数据输入[7:0]x：x[0], x[1] ,x[2],...,x[7]依次对应0~7的输出码字的十进制形式。

1. 输出:

输出端[2:0]y：输出x[7:0]对应优先译码结果的二进制形式。高位优先译码。

指示位h: 当输入有效（非全0且使能信号有效）时为1，否则为0.

下表\图给出了以上输入输出信号在DE10平台对应的信号：

|  |  |  |
| --- | --- | --- |
|  | 信号名称 | DE2-70平台信号 |
| 输入 | en | SW[0] |
| [7:0]x | SW[8:1] |
| 输出 | [2:0]y | LEDR[3:1] |
| h | LEDR[0] |

表 2-1-a:8-3优先编码器的输入输出信号与DE10平台信号对应关系

1. 七段数码管编码器
2. 输入:

数据输入[3:0]x：四位二进制数输入，数码管的显示数据范围：0~f.

1. 输出:

输出端[6:0]y：为输入端的编码结果，控制七段数码管LED的亮与否。

下表\图给出了以上输入输出信号在DE10平台对应的信号：

|  |  |  |
| --- | --- | --- |
|  | 信号名称 | DE2-70平台信号 |
| 输入 | [3:0]x | LEDR[3:1] |
| 输出 | [6:0]y | HEX0[6:0] |

表 2-1-b:七段数码管编码器的输入输出信号与DE10平台信号对应关系

##### 2.3建立模型

1. 8-3优先编码器

下表给出了8-3优先编码器的输出与输入的关系：

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | | 输出 | |
| 使能信号en | 输入信号[7:0]x | 输出信号[2:0]y | 输出指示信号h |
| 0 | xxxxxxx | 000 | 0 |
| 1 | 00000000 | 000 | 0 |
| 1 | 00000001 | 000 | 1 |
| 1 | 0000001x | 001 | 1 |
| 1 | 000001xx | 010 | 1 |
| 1 | 00001xxx | 011 | 1 |
| 1 | 0001xxxx | 100 | 1 |
| 1 | 001xxxxx | 101 | 1 |
| 1 | 01xxxxxx | 110 | 1 |
| 1 | 1xxxxxxx | 111 | 1 |

表 2-2-a:8-3优先编码器行为表

**实现思路：**使用for循环从低位向高位依次扫描输入[7:0]x, 若为1，则对y进行赋值（循环次数即为该位输入”1”对应的译码结果），指示信号置为1.若更高位还有1，则y会在循环中被修改为对应的值。

8-3优先编码器的Verilog HDL实现如下：

module pri\_encoder83(en, x, y, h);

input en;

input [7:0]x;

output reg [2:0]y;

output reg h;

integer i;

always @(x) begin

if(en) begin

y = 0; h = 0;

for(i = 0; i <= 7; i = i + 1)

if(x[i] == 1) begin

y = i; h = 1;

end

end

else begin

y = 0; h = 0;

end

end

endmodule

1. 七段数码管编码器

DE-10 Standard开发平台上的七段数码管为共阳极的，即一端已接高电平，给指定的LED灯输入低电平即可使该LED灯工作.

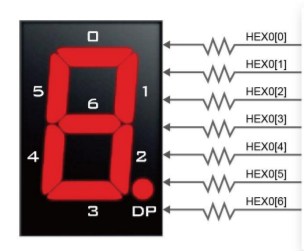


图2：HEX七段数码管

下表给出了七段数码管编码器的输出与输入的关系：

|  |  |
| --- | --- |
| 输入信号[3:0]x | 输出信号[6:0]y |
| 0000 | 1000000 |
| 0001 | 1111001 |
| 0010 | 0100100 |
| 0011 | 0110000 |
| 0100 | 0011001 |
| 0101 | 0010010 |
| 0110 | 0000010 |
| 0111 | 1111000 |
| 1000 | 0000000 |
| 1001 | 0010000 |
| 1010 | 0001000 |
| 1011 | 0000011 |
| 1100 | 1000110 |
| 1101 | 0100001 |
| 1110 | 0000110 |
| 1111 | 0001110 |

表 2-2-b:8-3七段数码管编码器行为表

**实现思路：**使用case语句实现.

七段数码管编码器的Verilog HDL实现如下：

module hex15(x, y);

input [3:0]x;

output reg [6:0]y;

always @(\*) begin

case(x)

4'b0000:y = 7'b1000000;

4'b0001:y = 7'b1111001;

4'b0010:y = 7'b0100100;

4'b0011:y = 7'b0110000;

4'b0100:y = 7'b0011001;

4'b0101:y = 7'b0010010;

4'b0110:y = 7'b0000010;

4'b0111:y = 7'b1111000;

4'b1000:y = 7'b0000000;

4'b1001:y = 7'b0010000;

4'b1010:y = 7'b0001000;

4'b1011:y = 7'b0000011;

4'b1100:y = 7'b1000110;

4'b1101:y = 7'b0100001;

4'b1110:y = 7'b0000110;

4'b1111:y = 7'b0001110;

default: y = 7'b1111111;

endcase

end

endmodule

#### **2.4分析/综合**

分析/综合实验成功，如下图所示：

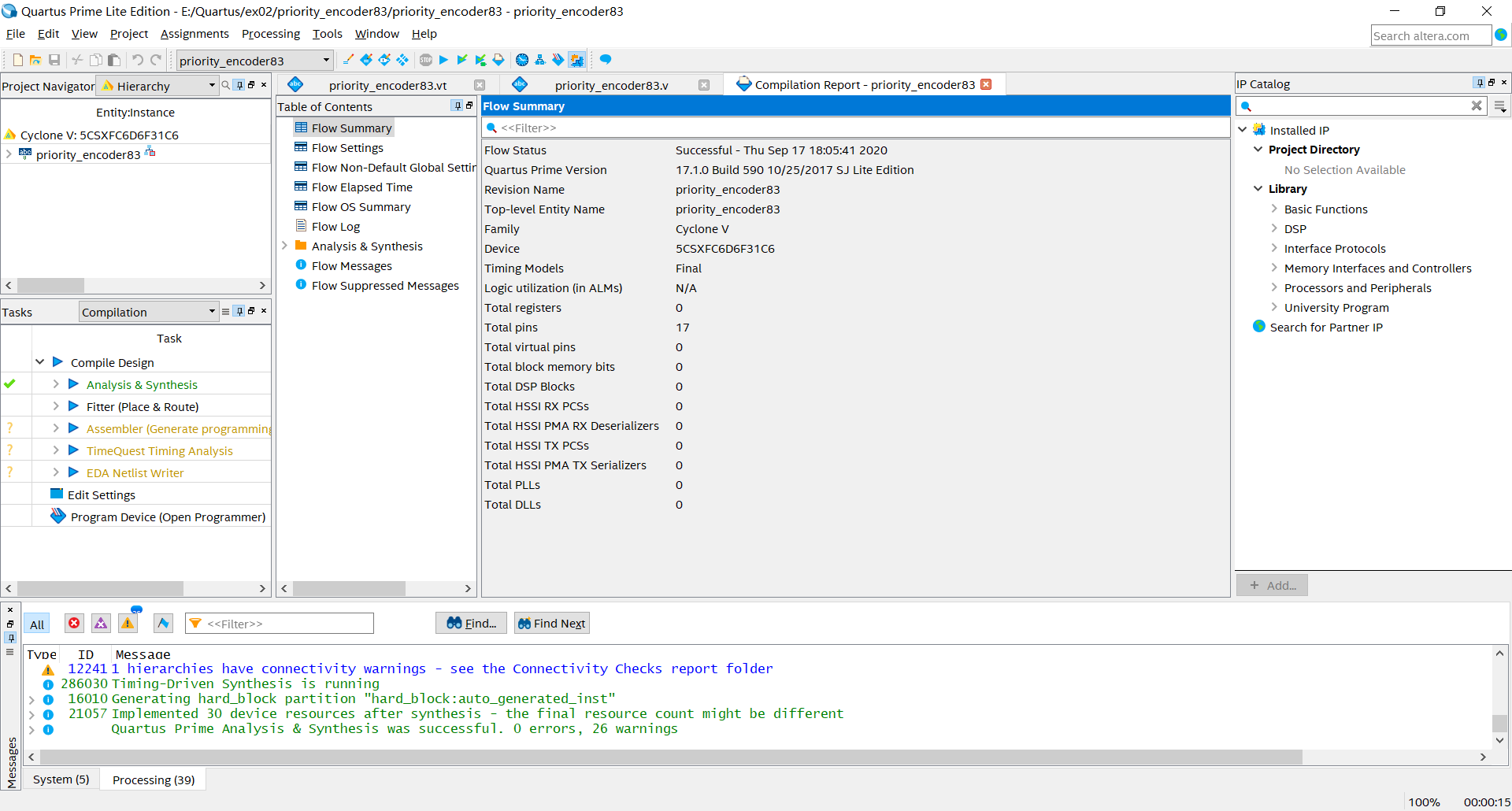


图2-1：分析/综合成功

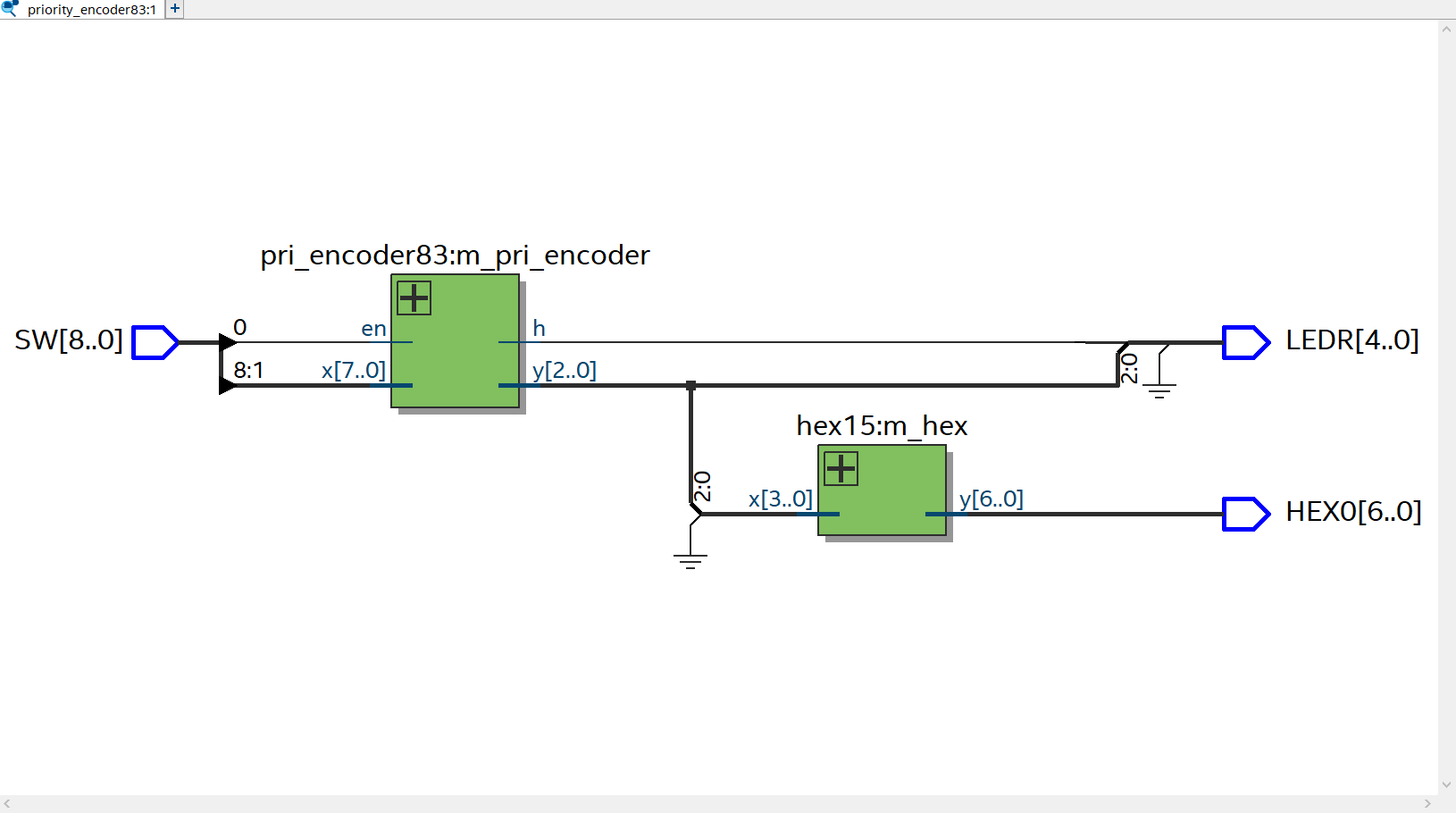


图 2-2:RTL视图

##### 2.5仿真测试

在Test Bench Template Writer提供的仿真文件模板的基础上给出以下测试样例：

|  |  |  |
| --- | --- | --- |
| 输入[8:0]SW | 期望输出[3:0]LEDR | 期望输出[6:0]HEX0 |
| 11111111 0 | 000 0 | 1000000 |
| 000000001 | 000 0 | 1000000 |
| 000000011 | 000 1 | 1000000 |
| 000000111 | 001 1 | 1111001 |
| 000001111 | 010 1 | 0100100 |
| 000011111 | 011 1 | 0110000 |
| 000111111 | 100 1 | 0011001 |
| 001111111 | 101 1 | 0010010 |
| 011111111 | 110 1 | 0000010 |
| 111111111 | 111 1 | 1111000 |

表 2-3:仿真测试样例与期望输出

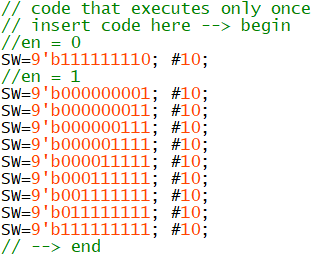


图2-3 仿真样例

通过ModelSim得到的仿真波形图与期望输出一致：

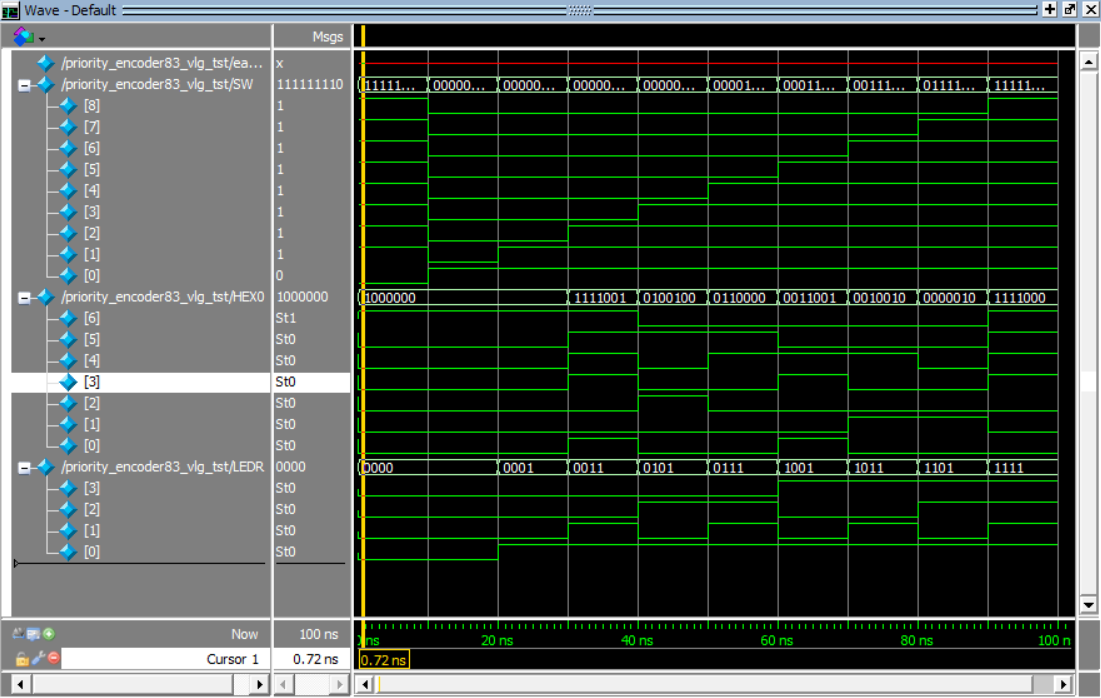
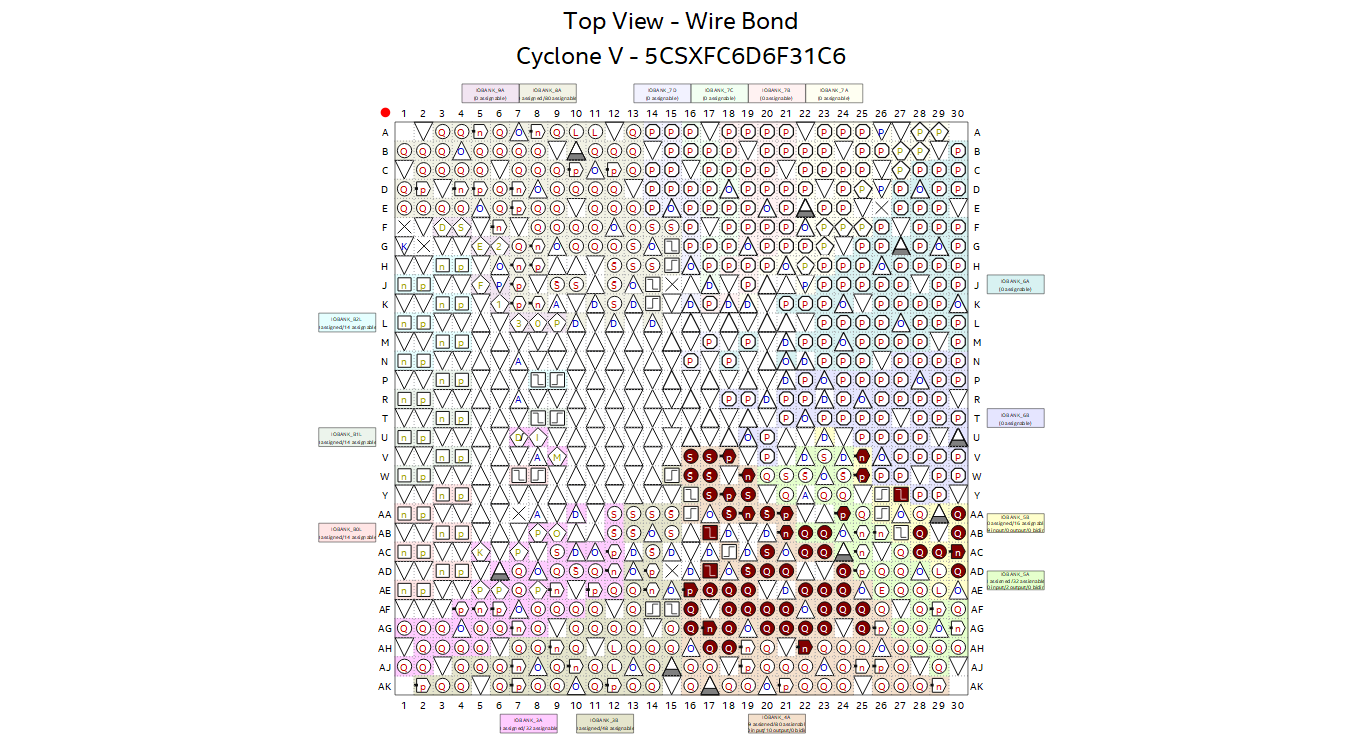


图2-4仿真测试波形图

##### 2.6分配引脚

引脚分配使用DE10\_Standard\_SystemBuilder生成。



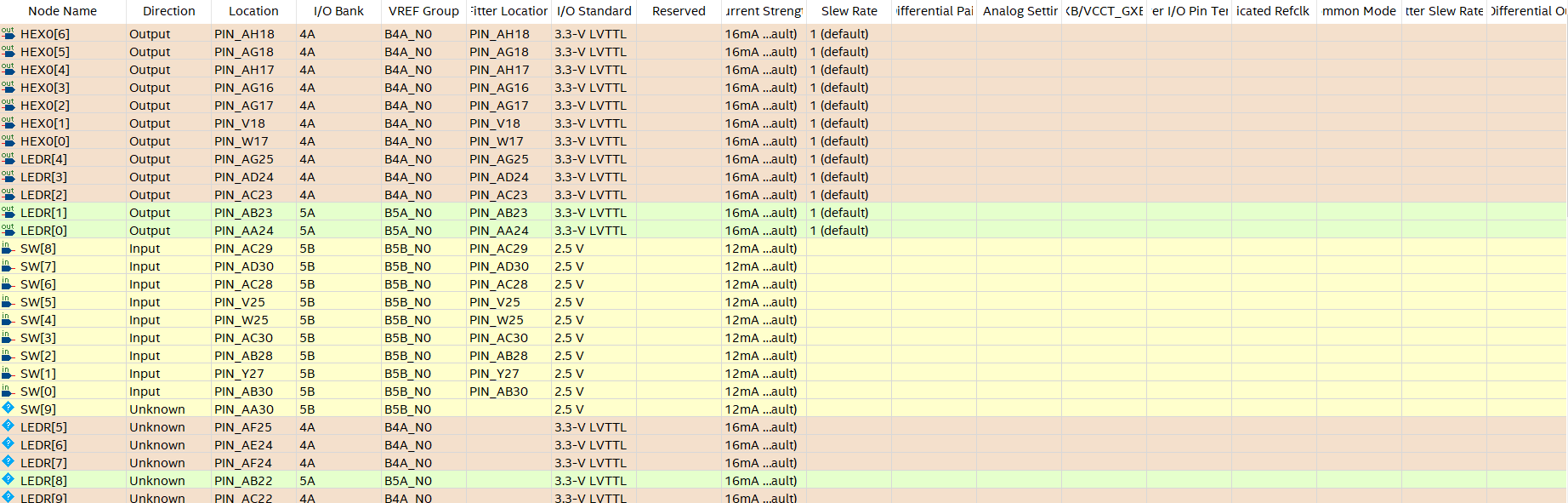


图2-5引脚分配图

##### 2.7全编译

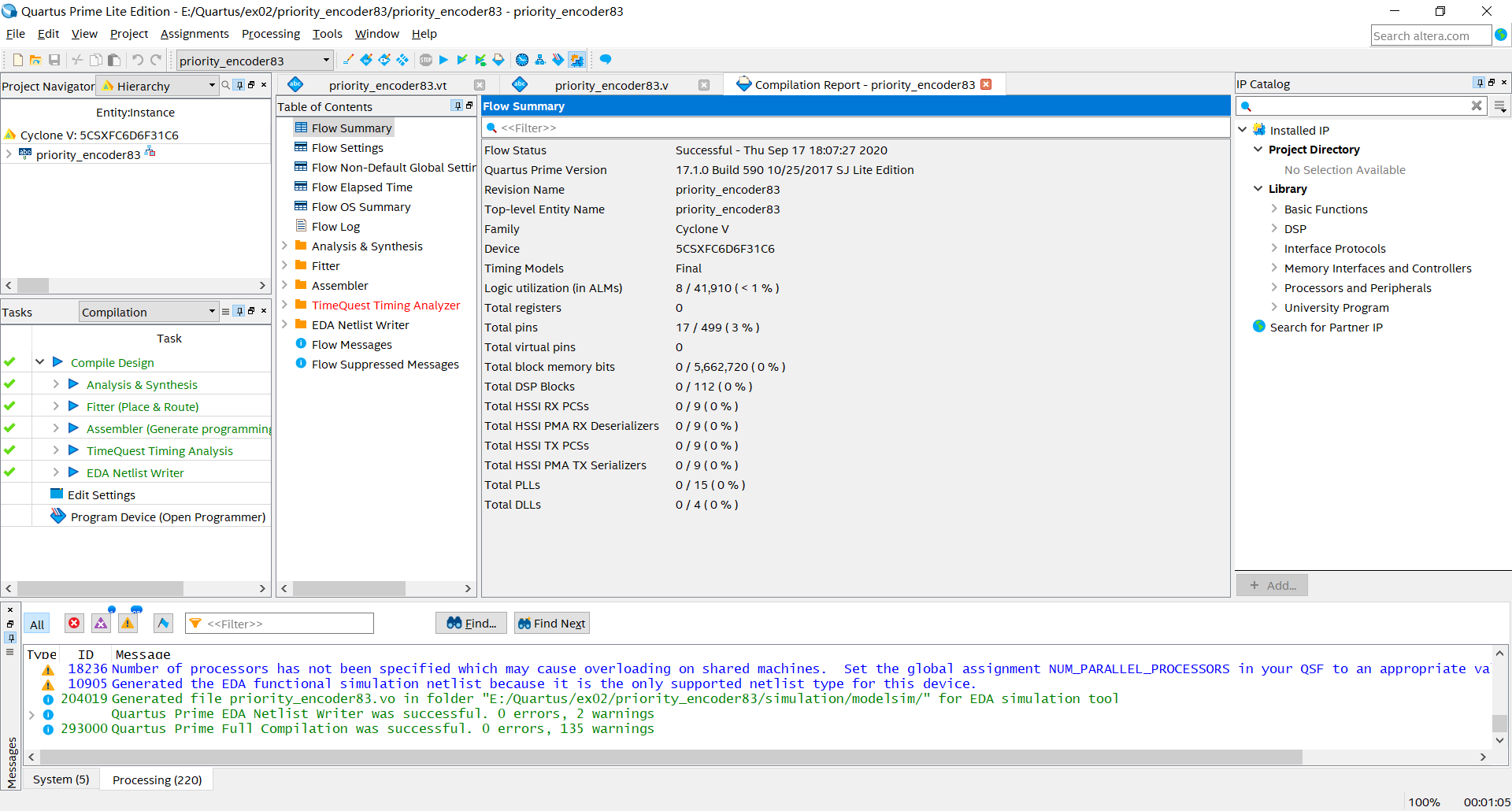


图2-6全编译成功

#### 实验总结

本次实验实现了8-3 优先编码器，通过仿真、实操验证了设计正确性。思考题中给出的casex，casez语句提示了不同的实现方式，详见附4.2。

#### 附

##### 4.1实验中遇到的问题及处理办法

***问题I：***

七段数码管的仿真波形图显示为不确定状态（Htx状态），实际操作时却能正常运行.

***原因分析及处理办法：***

七段数码管编码的模块提供的输入为4位，而顶层文件中给出的输入为3位的8-

3编码器的编码结果.Modelsim在仿真时因七段数码管的第四位输入状态未知导致七段数码管状态为Htx.

将顶层文件中的七段数码管编码器的输入改为4位（最高位为0）后问题得到解决.



图4-1问题I的处理方式图

##### 4.2 关于思考题

思考：查阅相关资料，了解 casex 和 casez 语句的使用，思考如何用用casex 语句来完成优先编码器的设计？

**casex语句：**如果比较双方的对应位的值是x或者z（高阻态），那么这些位的比较不予考虑.

**casez语句：**如果比较双方的对应位的值是z（高阻态），那么这些位的比较不予考虑.

**用casex语句实现优先编码器的设计：**

修改2.3-1.中给出的pri\_encoder83模块代码如下：

module pri\_encoder83(en, x, y, h);

input en;

input [7:0]x;

output reg [2:0]y;

output reg h;

integer i;

always @(x) begin

if(en) begin

y = 0; h = 0;

casex(x)

8'b00000001:begin y = 0; h = 1; end

8'b0000001x:begin y = 1; h = 1; end

8'b000001xx:begin y = 2; h = 1; end

8'b00001xxx:begin y = 3; h = 1; end

8'b0001xxxx:begin y = 4; h = 1; end

8'b001xxxxx:begin y = 5; h = 1; end

8'b01xxxxxx:begin y = 6; h = 1; end

8'b1xxxxxxx:begin y = 7; h = 1; end

default:;

endcase

end

else begin

y = 0; h = 0;

end

end

endmodule

采用和2.5中相同的仿真样例进行仿真测试得到的波形图：

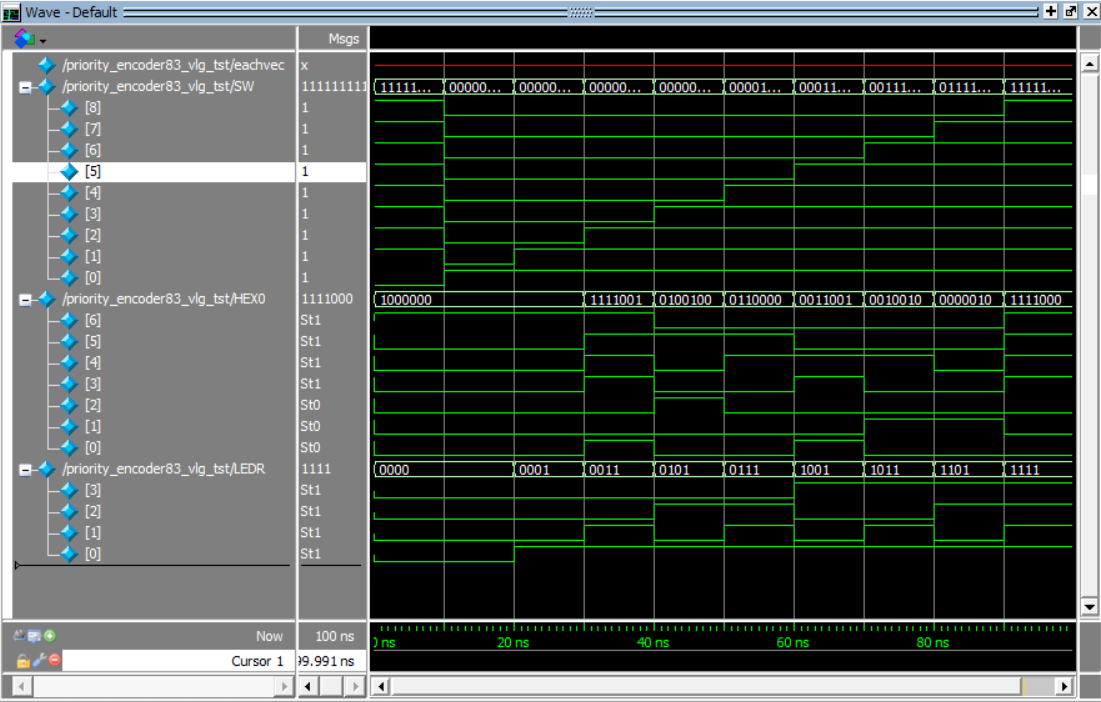


图4-2仿真测试波形图

发现得到的仿真结果和图2-4相同，在开发平台上的实际操作得到的结果也是正确的，说明该实现方式也是正确的.