# 数字电路与数字系统实验

## EX05:计数器和时钟

191220029 傅小龙

周一5-6节班

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

2020年10月11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)……………………………………………………………………………………3

2.1 基础实验

2.1.1 [模型概述](#_2.1模型概述)………………………………………………………………………3

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

2.1.3 [建立模型](#_2.3建立模型)………………………………………………………………………4

2.1.4 [分析/综合](#_2.4分析/综合)…………………………………………………………………… 5

2.1.5 [仿真测试](#_2.5仿真测试)………………………………………………………………………7

2.1.6 [分配引脚](#_2.6分配引脚)………………………………………………………………………8

2.1.7 [全编译](#_2.7全编译)…………………………………………………………………………8

2.2 拓展试验

2.2.1 [模型概述](#_2.2.1模型概述)………………………………………………………………………9

2.2.2 [数字抽象](#_2.2.2数字抽象)………………………………………………………………………9

2.2.3 [建立模型](#_2.2.3建立模型)………………………………………………………………………10

2.2.4 [分析/综合](#_2.2.4分析/综合)…………………………………………………………………… 14

2.2.5 [仿真测试](#_2.2.5仿真测试)………………………………………………………………………16

2.2.6 [分配引脚](#_2.2.6分配引脚)………………………………………………………………………16

2.2.7 [全编译](#_2.2.7全编译)…………………………………………………………………………18

三、[实验总结](#_实验总结)……………………………………………………………………………………18

#### 实验内容

##### 1.1实验要求

***I)基础实验***

在 DE10-Standard 开发板上实现一个计时器，在七段数码管上直接以十进制显示。

利用开发板上的频率为 50MHz 的时钟，先设计一个分频器，输入为50MHz 的时钟，输出为一个频率为 1Hz，周期为 1 秒的时钟信号。再用这个新的频率为 1Hz 的时钟信号作为你设计的时钟信号，进行计数。

此计时器有开始、暂停和清零功能，从 00 计数到 99，计数值到99 后重新从零开始计数。在数码管上用两位数字显示。

可以在计时结束的时候让某一个发光二极管闪烁，提示计时结束。

***II)拓展试验***

在 DE-10 Standard 开发板上实现一个电子时钟，时钟要求能够显示时、分、秒；还可以有以下功能：调整时间；闹铃（在特定时间 LED 闪烁）；秒表；等。

##### 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 基础实验

###### 2.1.1模型概述

使用 Verilog HDL 实现一个模100计数器. 计数器要求的输入时钟信号频率为1Hz，故需要先将开发板上50MHZ的时钟信号转换为1Hz的时钟信号，该分频器本质上也是个计数器. 计数结果以十进制的形式在数码管上显示，当完成一个计数周期时使某一LED灯亮起提示，在下一时钟周期到来后熄灭. 该计数器具有清零功能. 这里设计为异步清零.

###### 2.1.2数字抽象

1. 输入:

时钟信号clk： 与分频器的输出时钟信号连接.

使能信号en： 当en为1时计数器正常工作，否则暂停计数.

清零信号clr\_n： 当清零信号为0时（低有效）计数异步清零.

1. 输出:

num1[3:0]：当前计数的十位数字.

num0[3:0]：当前计数的个位数字.

rco： 完成一个计数周期的提示信号.

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 | DE-10平台信号 |
| 输入 | clk | LEDR[0]① |
| en | SW[0] |
| clr\_n | SW[1] |
| 输出 | [3:0]num1 | [3:0]num1② |
| [3:0]num0 | [3:0]num0② |
| rco | LEDR[9] |

①:LEDR[0]为分频器产生的1Hz时钟信号输出

②:num1,num0在顶层文件中为wire型变量，作为参数传递给七段数码管编码器

表 2-1-1:模100计数器输入输出信号与DE10平台信号对应关系

###### 2.1.3建立模型

**实现思路：**模块内使用integer变量cnt计数. 从0开始计数，达到99时rco置1，cnt置0，在下个计数周期开始时rco置0.

模100计数器的Verilog HDL实现如下：

module counter\_mod100(clk, en, clr\_n, num1, num0, rco);

input clk, en, clr\_n;

output reg [3:0]num1;

output reg [3:0]num0;

output reg rco;

integer cnt;

initial begin //初始化

cnt = 0;

end

always @(posedge clk or negedge clr\_n) begin

if(!clr\_n) begin //异步清零

cnt <= 0; rco <= 0;

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15  end  else begin  if(en) //计数  if(cnt == 99) begin  cnt <= 0; rco <= 1;  end  else begin  cnt <= cnt + 1;  rco <= 0;  end  else cnt <= cnt;//暂停  end  //输出  num1 <= cnt / 10;  num0 <= cnt % 10;  end  endmodule |  |
| 16 |  |
| 17 |  |
| 18 |  |
| 19 |  |
| 20 |  |
| 21 |  |
| 22 |  |
| 23 |  |
| 24 |  |
| 25 |  |
| 26 |  |
| 27 |  |
| 28 |  |
| 29 |  |
| 30 |  |
| 31 |  |

参照exp05.pdf表5-2秒时钟生成代码，本实验所用到的分频器的实现如下：

|  |  |
| --- | --- |
| 1  module clock\_1(clk\_50M, clk\_o);  input clk\_50M;  output reg clk\_o;  integer cnt;    always @(posedge clk\_50M) begin  if(cnt == 25000000) begin  cnt <= 0;  clk\_o <= ~clk\_o;  end  else cnt <= cnt + 1;  end  endmodule |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |

七段数码管编码器相关的代码实现详见2.2.3.

###### 2.1.4分析/综合

分析/综合实验成功，如下图所示：

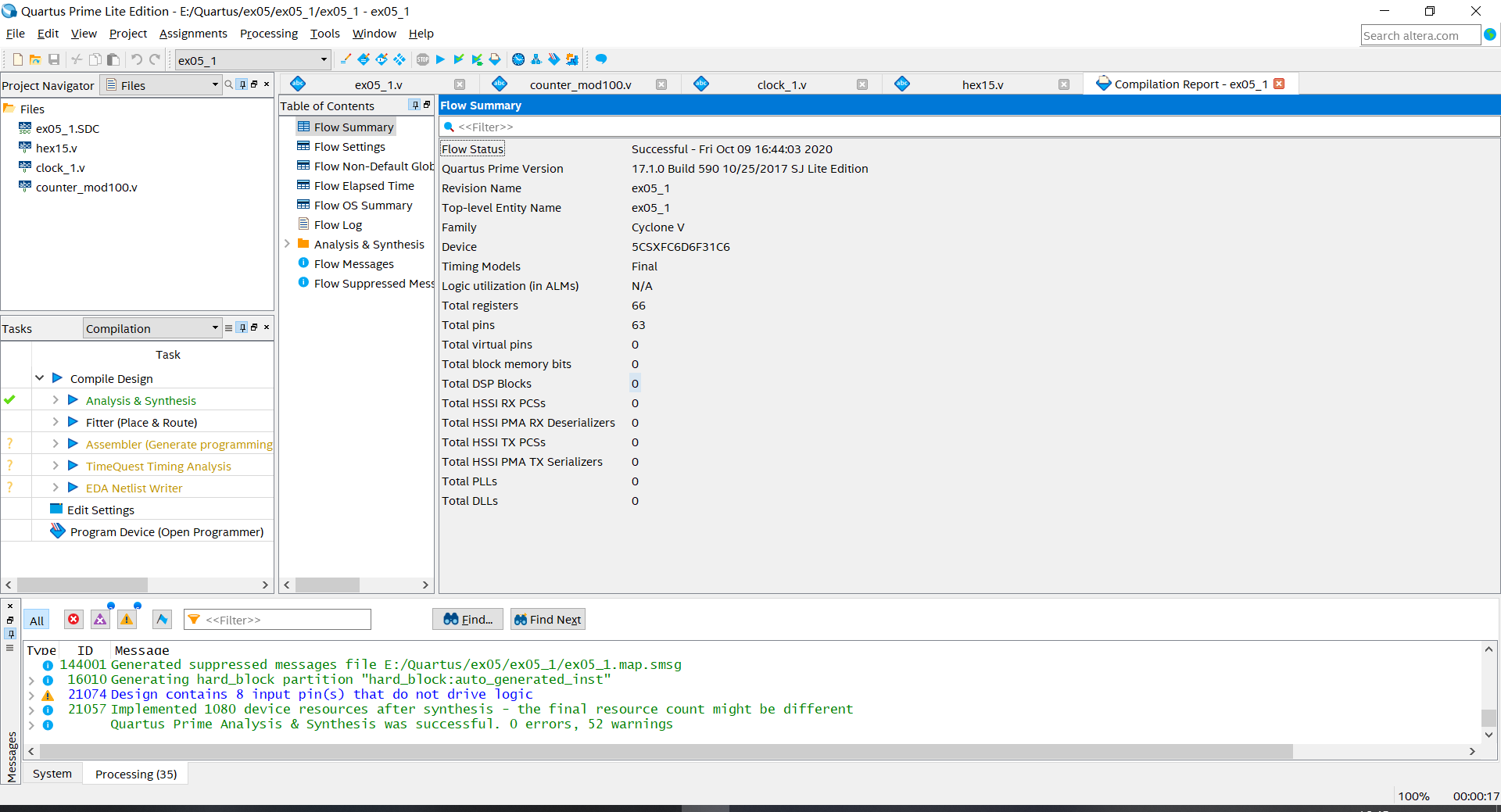
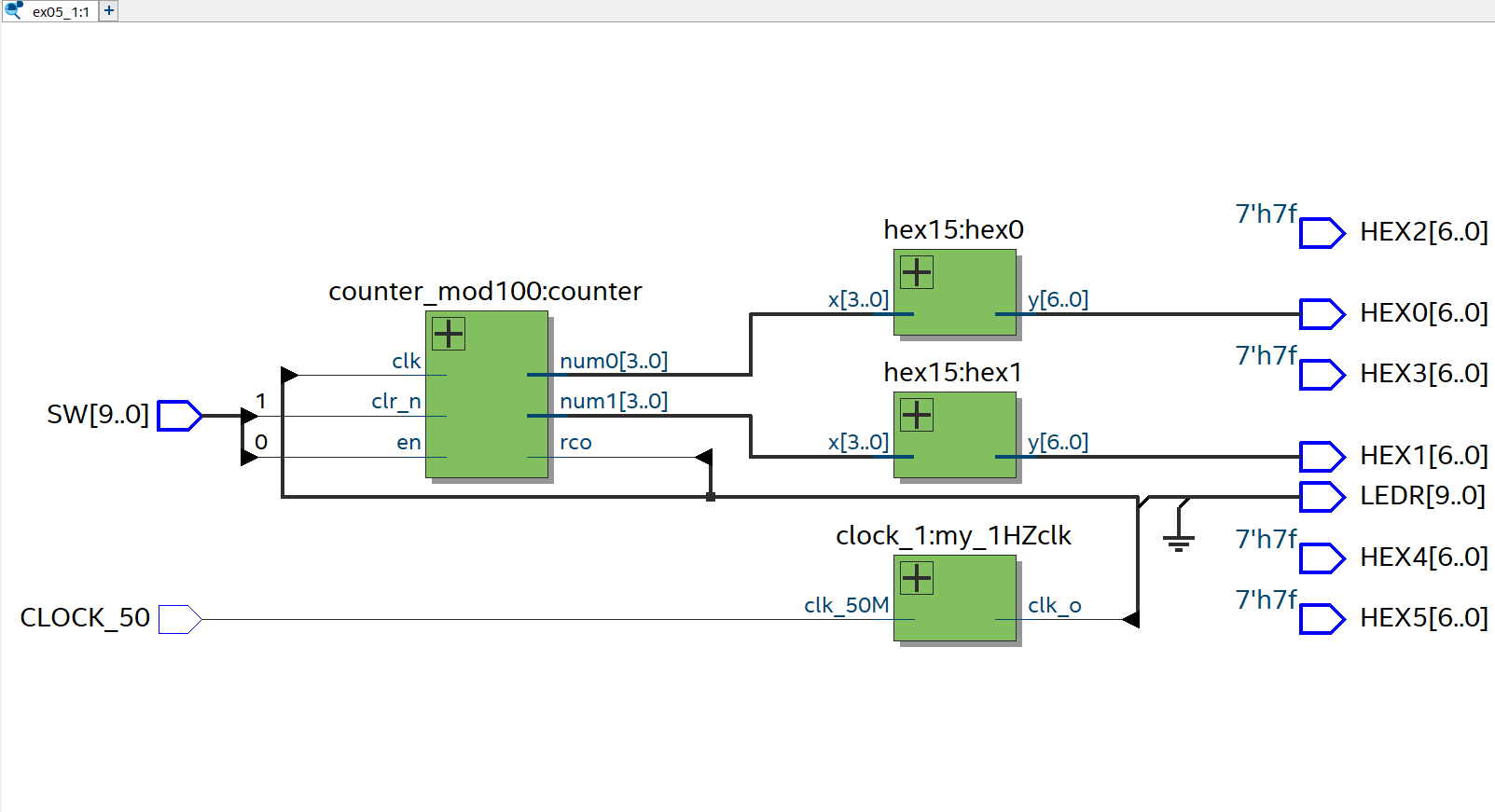
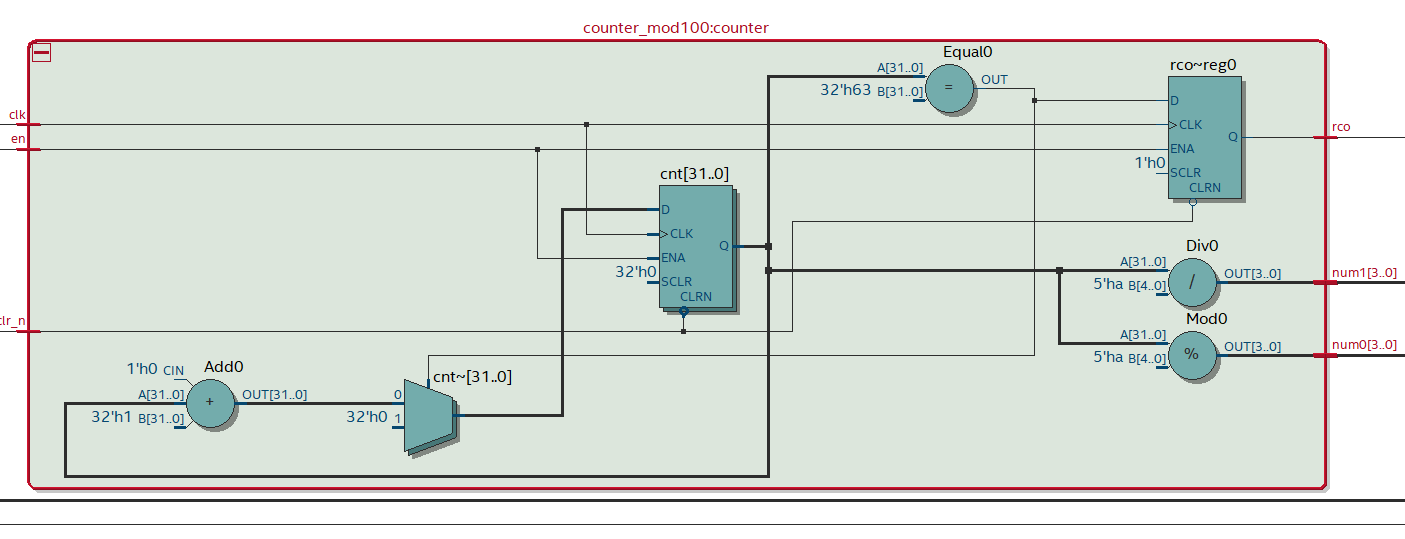


图2-1-1：分析/综合成功





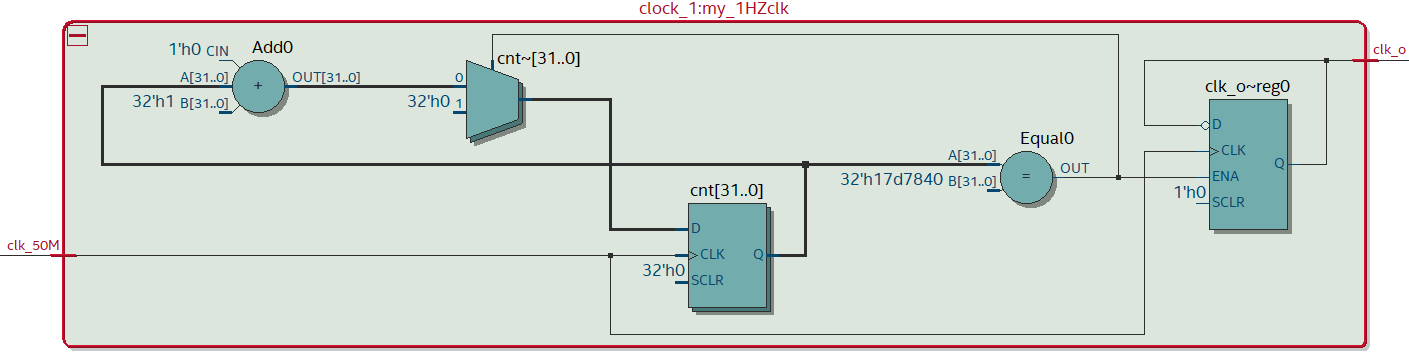


图 2-1-2:RTL视图

###### 2.1.5仿真测试

仿真测试中计数器模块的时钟信号输入与DE-10开发板的CLOCK\_50信号连接.

相关测试样例/代码如下：

initial

begin

// code that executes only once

CLOCK\_50 = 1;

SW[0] = 1; SW[1] = 1; #12;

SW[0] = 0; #10; //暂停计数

SW[1] = 0; #10; //清零

SW[0] = 1; SW[1] = 1; #1100;

$display("Running testbench");

$stop;

end

always

begin

// code executes for every event on sensitivity list

#5 CLOCK\_50 = ~CLOCK\_50;

end

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |
| 16 |  |

通过ModelSim执行上述测试代码得到的仿真结果如下：

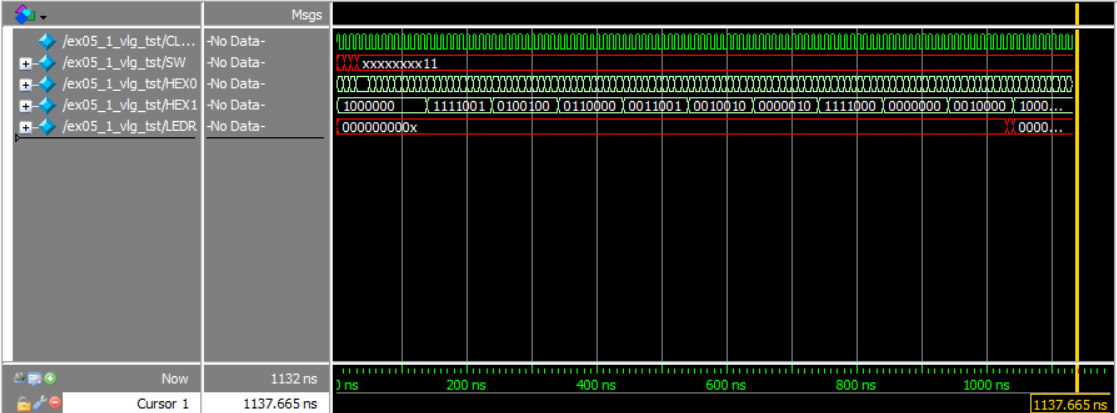
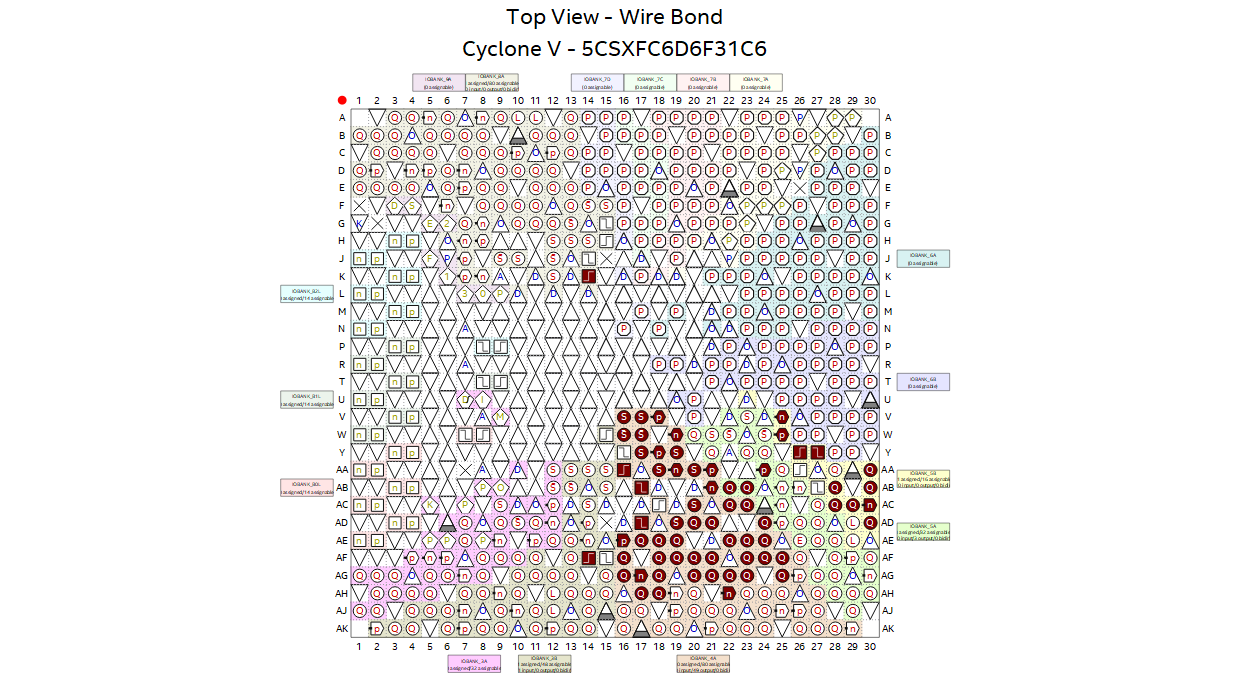


图 2-1-3仿真波形图

观察仿真波形图发现暂停计数、清零的用例均符合设计要求，在1035ns附近完成计数周期时的rco输出也符合设计要求.

###### 2.1.6分配引脚

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



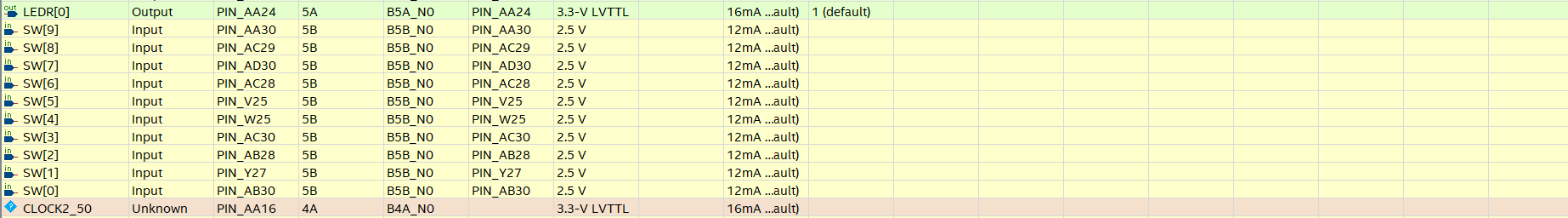
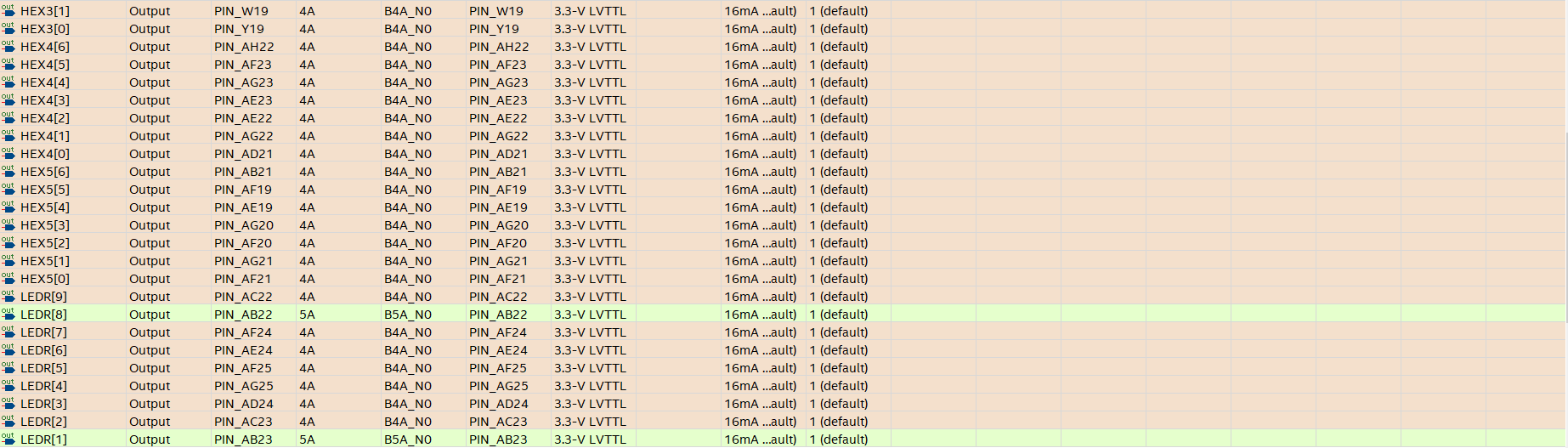
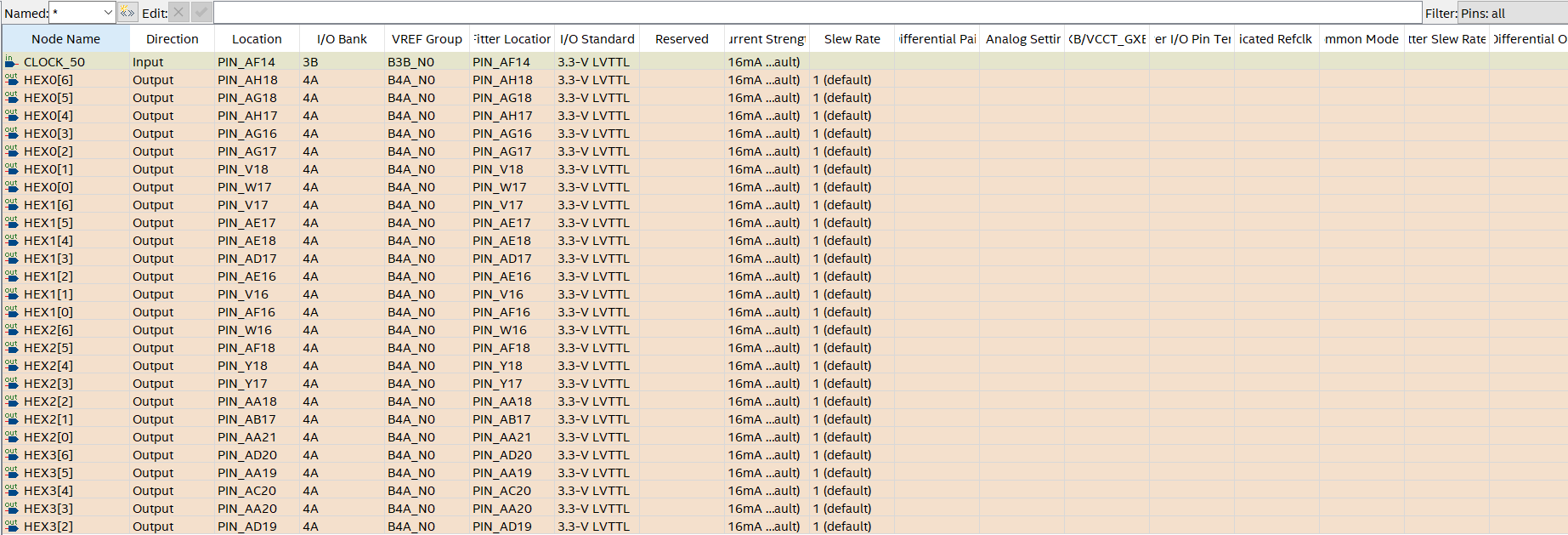


图2-1-4引脚分配图

###### 2.1.7全编译

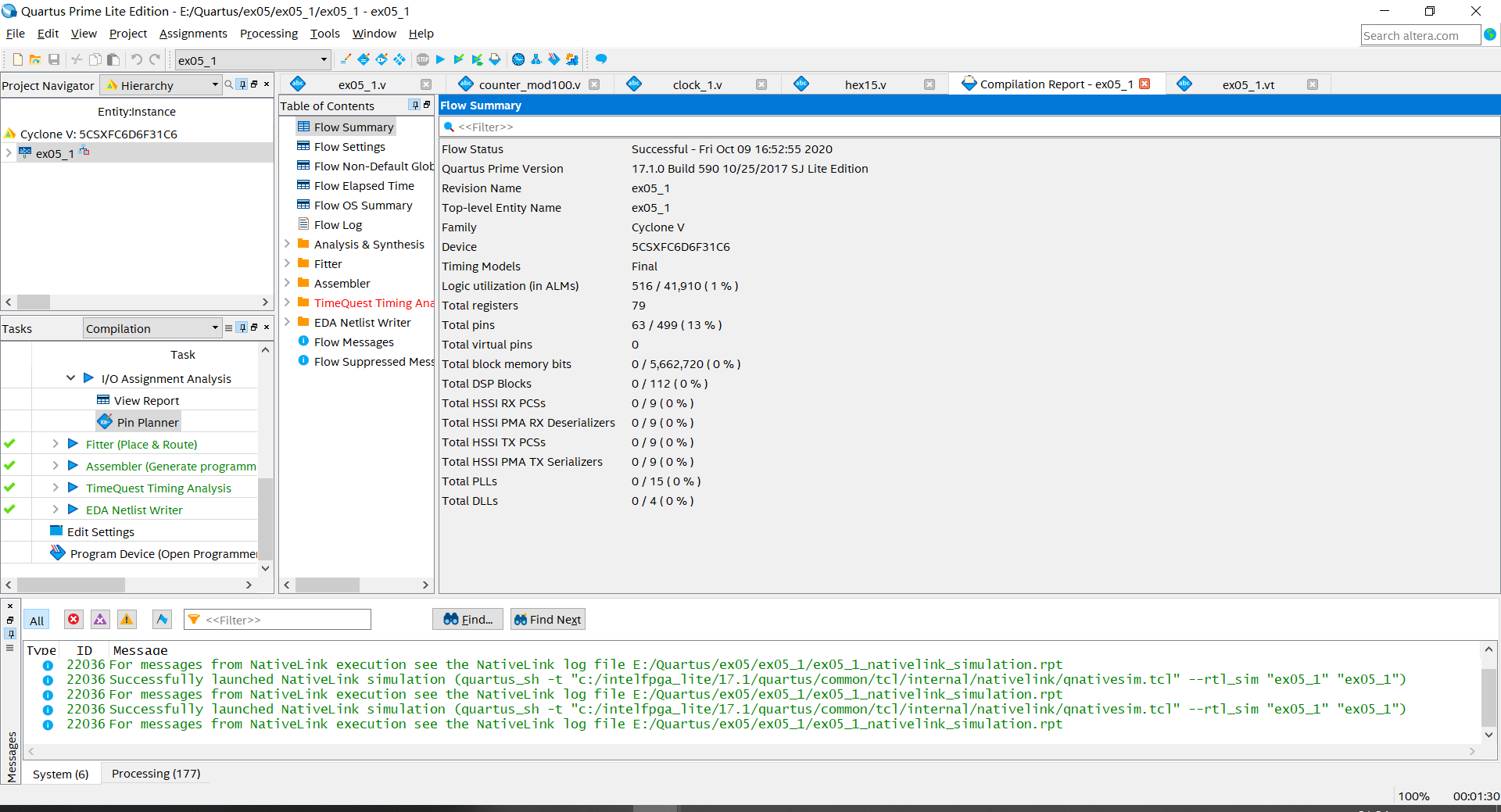


图2-1-6全编译成功

##### 2.2 拓展试验

###### 2.2.1模型概述

使用 Verilog HDL 设计一个电子时钟，显示时、分、秒，本质是一个特殊的计数器，和2.1基础实验同样需要一个1Hz时钟信号分频器. 调整时间通过开关输入数字，由按钮选择修改的是时、分或秒. 具有闹钟功能， 能够设定闹钟时间，和调整时间功能相类似. 当计数到设定时间时启动闹钟，再由特定输入信号关闭闹钟. 秒表功能和时钟功能类似，具有清零、暂停功能.

###### 2.2.2数字抽象

1. 输入:

时钟信号clk： 与分频器的输出时钟信号连接.

模式选择信号[1:0]mod: 根据该信号选择电子时钟的功能（时钟、设置时间、设置闹钟、秒表）

选择信号[2:0]select\_set： 在设置时间、闹钟时起作用，用于选择修改的对象（时、分、秒）

时间设置输入[2:0]set1： 调整时间的数字输入.这是十位.

时间设置输入[3:0]set0： 调整时间的数字输入.这是个位.

秒表清零信号clr\_l\_cnt： 秒表清零信号（低有效）.

秒表使能信号en\_cnt： 当该项输入为0时秒表停止计时.

闹钟关闭信号clr\_l\_alarm： 用于关闭闹钟（低有效）.

1. 输出:

数据输出[3:0]hr1： 小时数.这是十位.

数据输出[3:0]hr0： 小时数.这是个位.

数据输出[3:0]min1： 分钟数.这是十位.

数据输出[3:0]min0： 分钟数.这是个位.

数据输出[3:0]sec1： 秒数.这是十位.

数据输出[3:0]sec0： 秒数.这是个位.

信号输出alarm： 闹钟信号.

下表给出了以上输入输出信号在DE10平台对应的信号：

|  |  |  |
| --- | --- | --- |
|  | 信号名称 | DE-10平台信号 |
| 输入 | clk | LEDR[9]① |
| [1:0]mod | SW[9:8] |
| [2:0]select\_set | KEY[2:0] |
| [2:0]set1 | SW[7:4] |
| [3:0]set0 | SW[3:0] |
| clr\_l\_cnt | KEY[3] |
| en\_cnt | SW[0] |
| clr\_l\_alarm | KEY[3] |
| 输出 | [3:0]hr1 | [3:0]hour1② |
| [3:0]hr0 | [3:0]hour0② |
| [3:0]min1 | [3:0]min1② |
| [3:0]min0 | [3:0]min0② |
| [3:0]sec1 | [3:0]sec1② |
| [3:0]sec0 | [3:0]sec0② |
| alarm | LEDR[0] |

①:LEDR[9]为分频器产生的1Hz时钟信号输出

②:hour1,hour0,min1,min0,sec1,sec0在顶层文件中为wire型变量，作为参数传递给七段数码管编码器.

表 2-2-1:电子时钟输入输出信号与DE10平台信号对应关系

###### 2.2.3建立模型

下表给出了电子时钟的行为表：

|  |  |  |  |
| --- | --- | --- | --- |
| 模式选择信号[1:0]mod | 选择信号[2:0]select\_set | 对应功能 | 备注 |
| 00 | xxx | 时钟 | 当时间和闹钟设置时间相同时启动闹钟 |
| 01 | 110 | 修改时间 | 修改秒数 |
| 101 | 修改分数 |
| 011 | 修改小时数 |
| 10 | 110 | 设置闹钟 | 修改秒数 |
| 101 | 修改分数 |
| 011 | 修改小时数 |
| 11 | xxx | 秒表 | - |

表 2-2-3:同步清零D触发器行为表

**实现思路：**由于时钟、闹钟、秒表的计时是彼此独立的，故将这3组时间需要在模块内各以integer变量存储. 使用case语句来实现模式选择，且时钟的计时应该与其他功能相独立，即时钟计时相关的语句在case语句外，以免执行其他功能时影响到时钟计时.

电子时钟的Verilog HDL实现如下：

module myClock(clk, mod, select\_set, set1, set0, clr\_l\_cnt, en\_cnt, clr\_l\_alarm, hr1, hr0, min1, min0, sec1, sec0, alarm);

input clk; //外部时钟输入

input [1:0]mod; //模式选择信号

input [2:0]select\_set; //设置时间 选择信号 \*\*\*考虑到按钮KEY按下为0，选择110, 101, 011分别作为s, m, h的设置信号

input [2:0]set1; //设置时间的对应数字输入，十位+个位

input [3:0]set0;

input clr\_l\_cnt; //秒表清零

input en\_cnt; //秒表计数开始

input clr\_l\_alarm; //关闭闹钟

output reg [3:0]hr1; //小时数输出 十位+个位

output reg [3:0]hr0;

output reg [3:0]min1; //分钟数输出

output reg [3:0]min0;

output reg [3:0]sec1; //秒数输出

output reg [3:0]sec0;

output reg alarm; //闹钟

integer h, m, s; //内部存储：小时、分钟、秒

integer h\_alm, m\_alm, s\_alm; //内部存储：闹钟对应的时间

integer h\_swch, m\_swch, s\_swch; //内部存储：秒表对应的时间

initial begin

h = 0; m = 0; s = 0;

h\_alm = 24; m\_alm = 0; s\_alm = 0;

h\_swch = 0; m\_swch = 0; s\_swch = 0;

end

always @(posedge clk) begin

//时钟计时模块

if(h == 23 && m == 59 && s == 59) begin

h <= 0; m <= 0; s <= 0;

end

else if(m == 59 && s == 59) begin

h <= h + 1; m <= 0; s <= 0;

end

else if(s == 59) begin

m <= m + 1; s <= 0;

end

else s <= s + 1;

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |
| 16 |  |
| 17 |  |
| 18 |  |
| 19 |  |
| 20 |  |
| 21 |  |
| 22 |  |
| 23 |  |
| 24 |  |
| 25 |  |
| 26 |  |
| 27 |  |
| 28 |  |
| 29 |  |
| 30 |  |
| 31 |  |
| 32 |  |
| 33 |  |
| 34 |  |
| 35 |  |
| 36 |  |
| 37 |  |
| 38 |  |
| 39 |  |
| 40 |  |
| 41 |  |
| 42  case(mod)  0: begin //正常计时  hr1 <= h / 10; hr0 <= h % 10;  min1 <= m / 10; min0 <= m % 10;  sec1 <= s / 10; sec0 <= s % 10;    //检查当前时间与闹钟设置时间是否一致，一致则alarm为1  if(h == h\_alm && m == m\_alm && s == s\_alm)  alarm = 1;  else alarm = alarm;  //闹钟的关闭  if(!clr\_l\_alarm) alarm = 0;  else alarm = alarm;  end    1: begin //设置时间  case(select\_set) //根据select\_set信号选择要设置的是h, m 或者 s  3'b110: begin//s  if(set1 \* 10 + set0 <= 59)  s <= set1 \* 10 + set0;  else s <= 59; //确保时间是合法的  end  3'b101: begin//m  if(set1 \* 10 + set0 <= 59)  m <= set1 \* 10 + set0;  else m <= 59;  end  3'b011: begin//h  if(set1 \* 10 + set0 <= 23)  h <= set1 \* 10 + set0;  else h <= 23;  end  default:;  endcase    hr1 <= h / 10; hr0 <= h % 10;  min1 <= m / 10; min0 <= m % 10;  sec1 <= s / 10; sec0 <= s % 10;  end    2: begin //设置闹铃  case(select\_set) //根据select\_set信号选择要设置的是h\_alm, m\_alm 或者 s\_alm  3'b110: begin//s  if(set1 \* 10 + set0 <= 59)  s\_alm <= set1 \* 10 + set0;  else s\_alm <= 59;  end  3'b101: begin//m  if(set1 \* 10 + set0 <= 59)  m\_alm <= set1 \* 10 + set0;  else m\_alm <= 59;  end  3'b011: begin//h  if(set1 \* 10 + set0 <= 24)  h\_alm <= set1 \* 10 + set0;  else h\_alm <= 24;  end  default:;  endcase    //显示闹铃时间  hr1 <= h\_alm / 10; hr0 <= h\_alm % 10;  min1 <= m\_alm / 10; min0 <= m\_alm % 10;  sec1 <= s\_alm / 10; sec0 <= s\_alm % 10;  end    3: begin //秒表  if(!clr\_l\_cnt) begin //清零  h\_swch <= 0; m\_swch <= 0; s\_swch <= 0;  end  else if(en\_cnt) begin//计时  if(h\_swch == 23 && m\_swch == 59 && s\_swch == 59) begin  h\_swch <= 0; m\_swch <= 0; s\_swch <= 0;  end  else if(m\_swch == 59 && s\_swch == 59) begin  h\_swch <= h\_swch + 1; m <= 0; s <= 0;  end  else if(s == 59) begin  m\_swch <= m\_swch + 1; s\_swch <= 0;  end  else s\_swch <= s\_swch + 1;  end  else begin//暂停  h\_swch <= h\_swch; m\_swch <= m\_swch; s\_swch <= s\_swch;  end  hr1 <= h\_swch / 10; hr0 <= h\_swch % 10;  min1 <= m\_swch / 10; min0 <= m\_swch % 10;  sec1 <= s\_swch / 10; sec0 <= s\_swch % 10;  end    default: begin  hr1 <= hr1; hr0 <= hr0;  min1 <= min1; min0 <= min0;  sec1 <= sec1; sec0 <= sec0;  end  endcase    end  endmodule |  |
| 43 |  |
| 44 |  |
| 45 |  |
| 46 |  |
| 47 |  |
| 48 |  |
| 49 |  |
| 50 |  |
| 51 |  |
| 52 |  |
| 53 |  |
| 54 |  |
| 55 |  |
| 56 |  |
| 57 |  |
| 58 |  |
| 59 |  |
| 60 |  |
| 61 |  |
| 62 |  |
| 63 |  |
| 64 |  |
| 65 |  |
| 66 |  |
| 67 |  |
| 68 |  |
| 69 |  |
| 70 |  |
| 71 |  |
| 72 |  |
| 73 |  |
| 74 |  |
| 75 |  |
| 76 |  |
| 77 |  |
| 78 |  |
| 79 |  |
| 80 |  |
| 81 |  |
| 82 |  |
| 83 |  |
| 84 |  |
| 85 |  |
| 86 | if(set1 \* 10 + set0 <= 59)  s\_alm <= set1 \* 10 + set0;  else s\_alm <= 59;  end  3'b101: begin//m  if(set1 \* 10 + set0 <= 59)  m\_alm <= set1 \* 10 + set0;  else m\_alm <= 59;  end  3'b011: begin//h  if(set1 \* 10 + set0 <= 24)  h\_alm <= set1 \* 10 + set0;  else h\_alm <= 24;  end  default:;  endcase    //显示闹铃时间  hr1 <= h\_alm / 10; hr0 <= h\_alm % 10;  min1 <= m\_alm / 10; min0 <= m\_alm % 10;  sec1 <= s\_alm / 10; sec0 <= s\_alm % 10;  end    3: begin //秒表  if(!clr\_l\_cnt) begin //清零  h\_swch <= 0; m\_swch <= 0; s\_swch <= 0;  end  else if(en\_cnt) begin//计时  if(h\_swch == 23 && m\_swch == 59 && s\_swch == 59) begin  h\_swch <= 0; m\_swch <= 0; s\_swch <= 0;  end  else if(m\_swch == 59 && s\_swch == 59) begin  h\_swch <= h\_swch + 1; m <= 0; s <= 0;  end  else if(s == 59) begin  m\_swch <= m\_swch + 1; s\_swch <= 0;  end  else s\_swch <= s\_swch + 1;  end  else begin//暂停  h\_swch <= h\_swch; m\_swch <= m\_swch; s\_swch <= s\_swch; |
| 87 |  |
| 88 |  |
| 89 |  |
| 90 |  |
| 91 |  |
| 92 |  |
| 93 |  |
| 94 |  |
| 95 |  |
| 96 |  |
| 97 |  |
| 98 |  |
| 99 |  |
| 100 |  |
| 101 |  |
| 102 |  |
| 103 |  |
| 104 |  |
| 105 |  |
| 106 |  |
| 107 |  |
| 108 |  |
| 109 |  |
| 110 |  |
| 111 |  |
| 112 |  |
| 113 |  |
| 114 |  |
| 115 |  |
| 116 |  |
| 117 |  |
| 118 |  |
| 119 |  |
| 120 |  |
| 121 |  |
| 122 |  |
| 123 |  |
| 124 |  |
| 125 |  |
| 126 |  |
| 127 |  |
| 128 |  |
| 129 |  |
| 130 | end  hr1 <= h\_swch / 10; hr0 <= h\_swch % 10;  min1 <= m\_swch / 10; min0 <= m\_swch % 10;  sec1 <= s\_swch / 10; sec0 <= s\_swch % 10;  end    default: begin  hr1 <= hr1; hr0 <= hr0;  min1 <= min1; min0 <= min0;  sec1 <= sec1; sec0 <= sec0;  end  endcase  end  endmodule |
| 131 |  |
| 132 |  |
| 133 |  |
| 134 |  |
| 135 |  |
| 136 |  |
| 137 |  |
| 138 |  |
| 139 |  |
| 140 |  |
| 150 |  |
| 151 |  |
| 152 |  |

###### 2.2.4分析/综合

分析/综合实验成功，如下图所示：

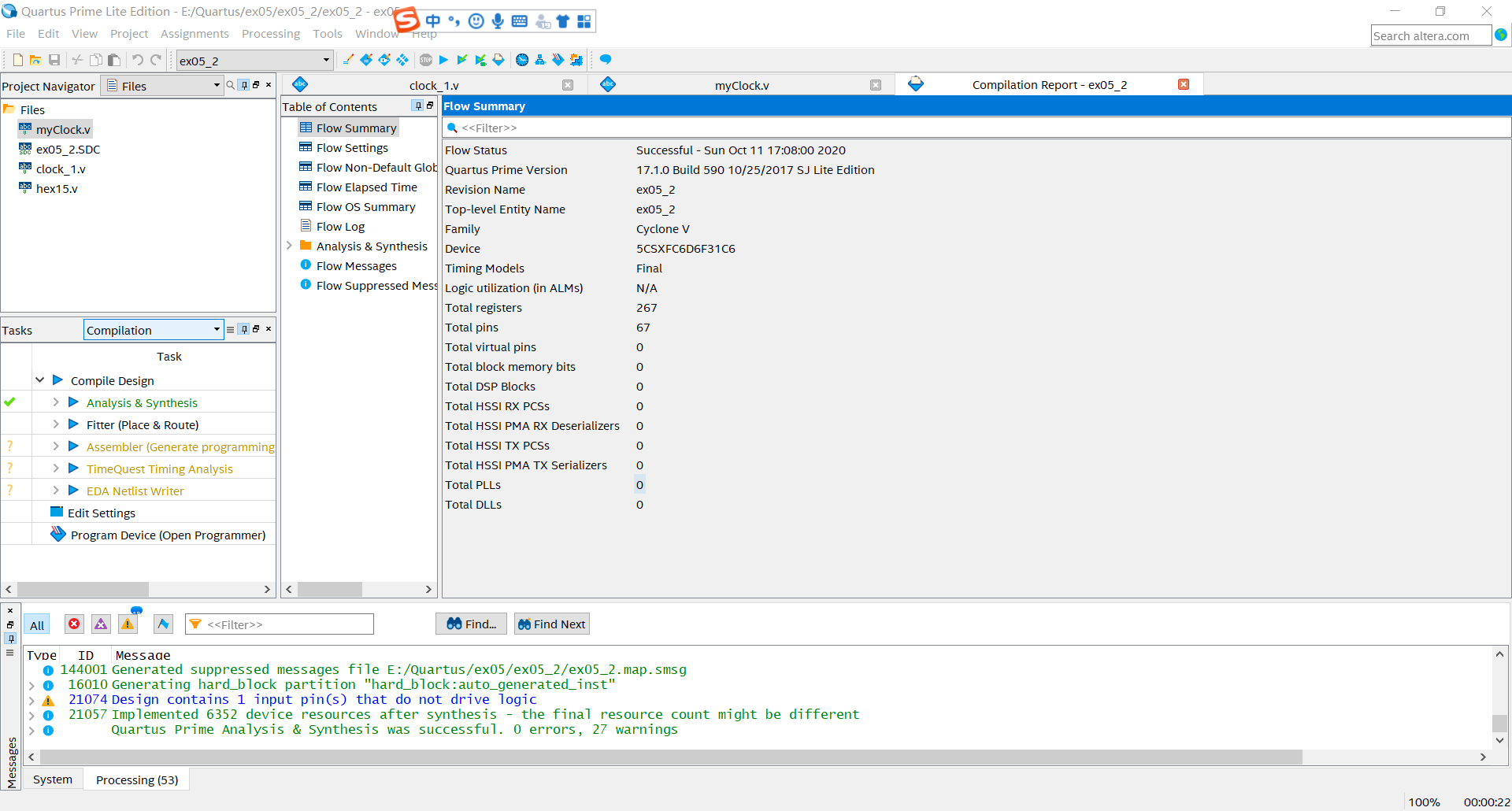
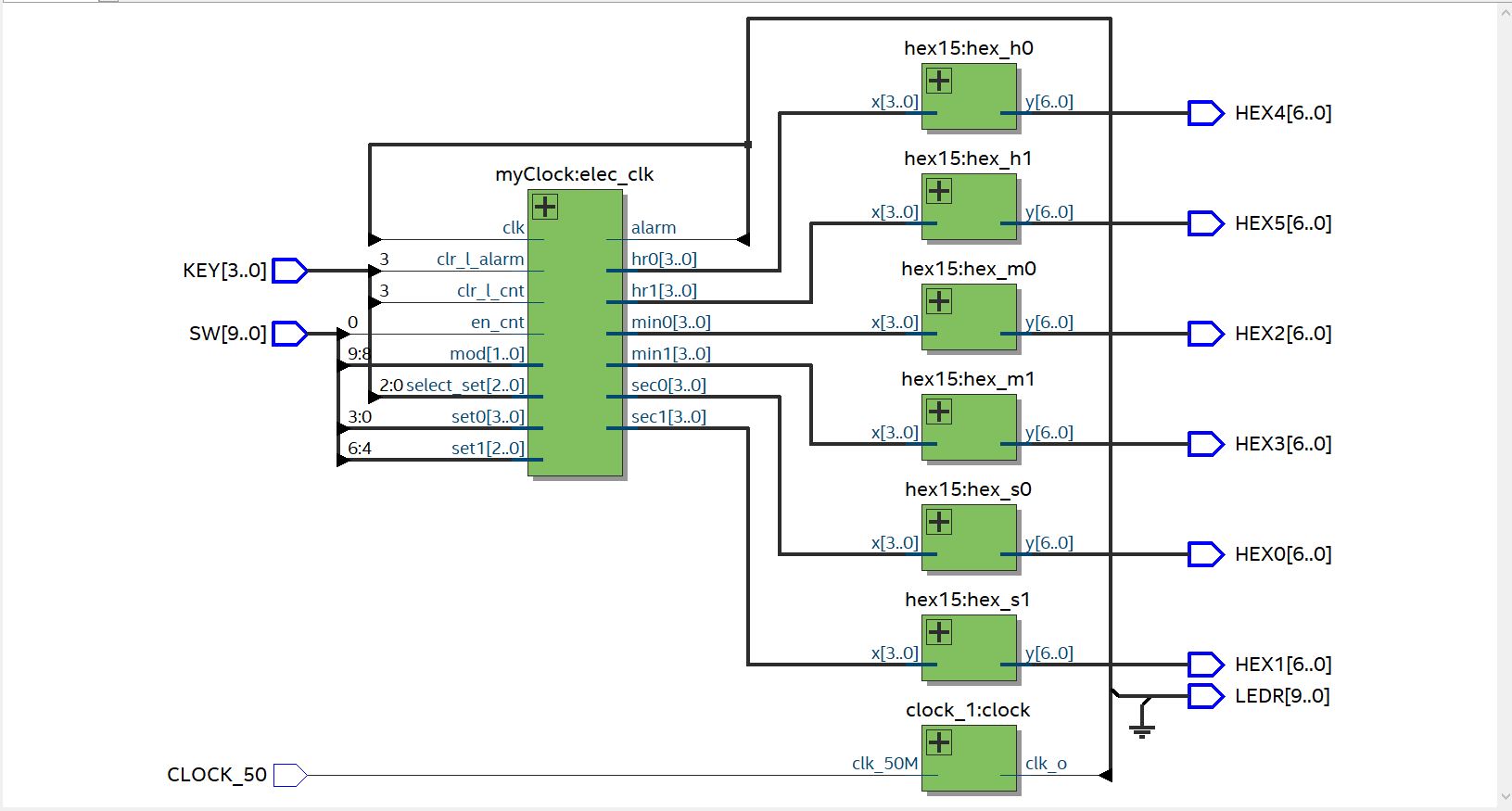


图2-2-1：分析/综合成功



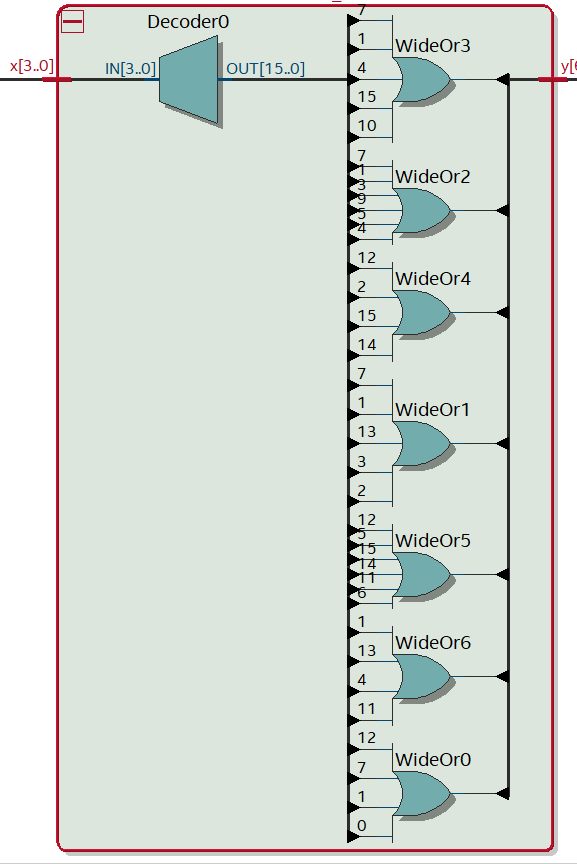
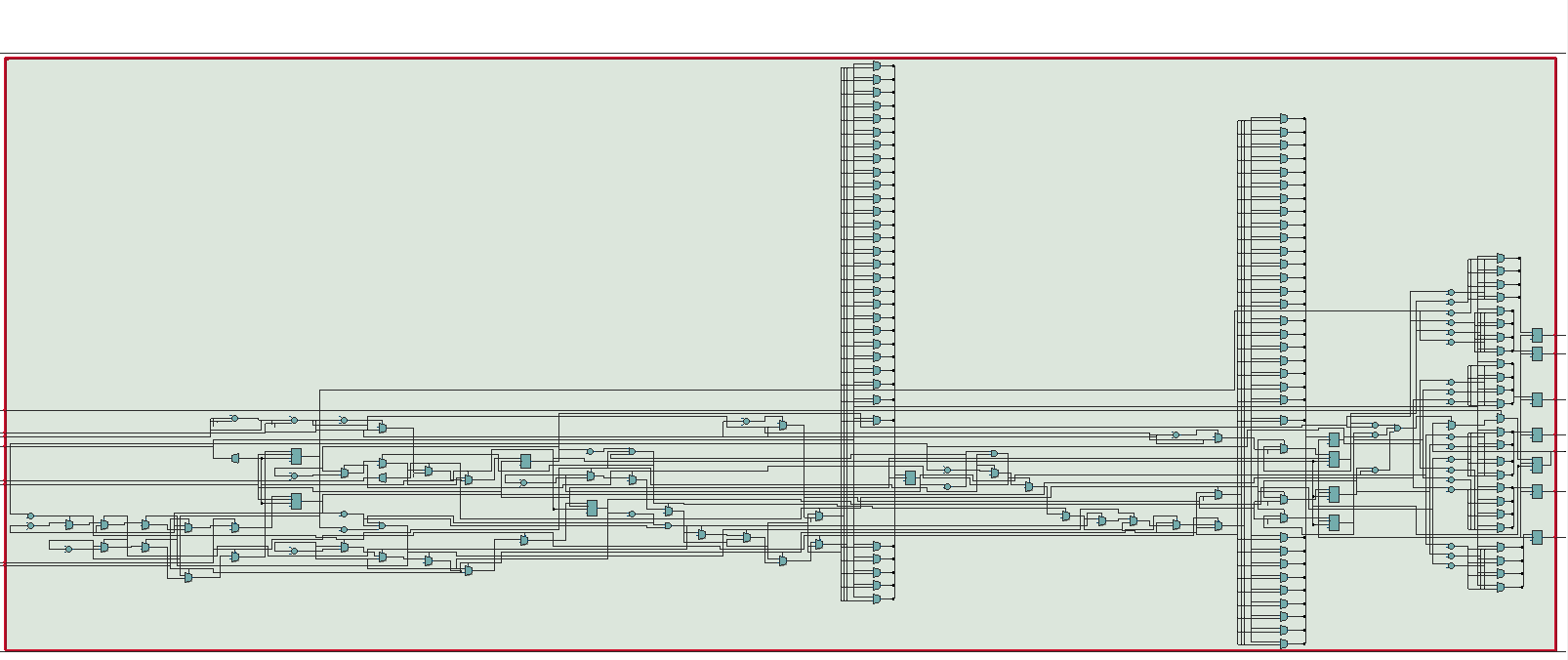


图 2-2-2:RTL视图

###### 2.2.5仿真测试

由于该模型功能较多，测试方法繁琐，不对该模型进行仿真测试.

###### 2.2.6分配引脚

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

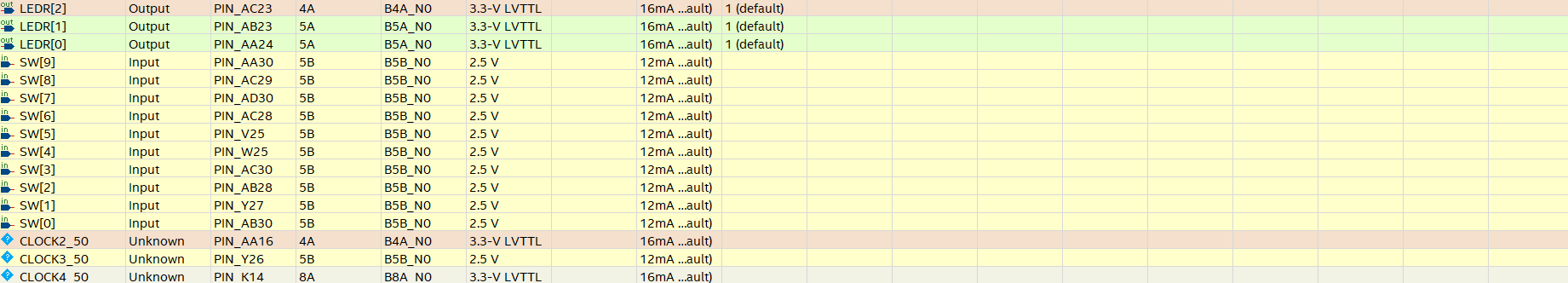
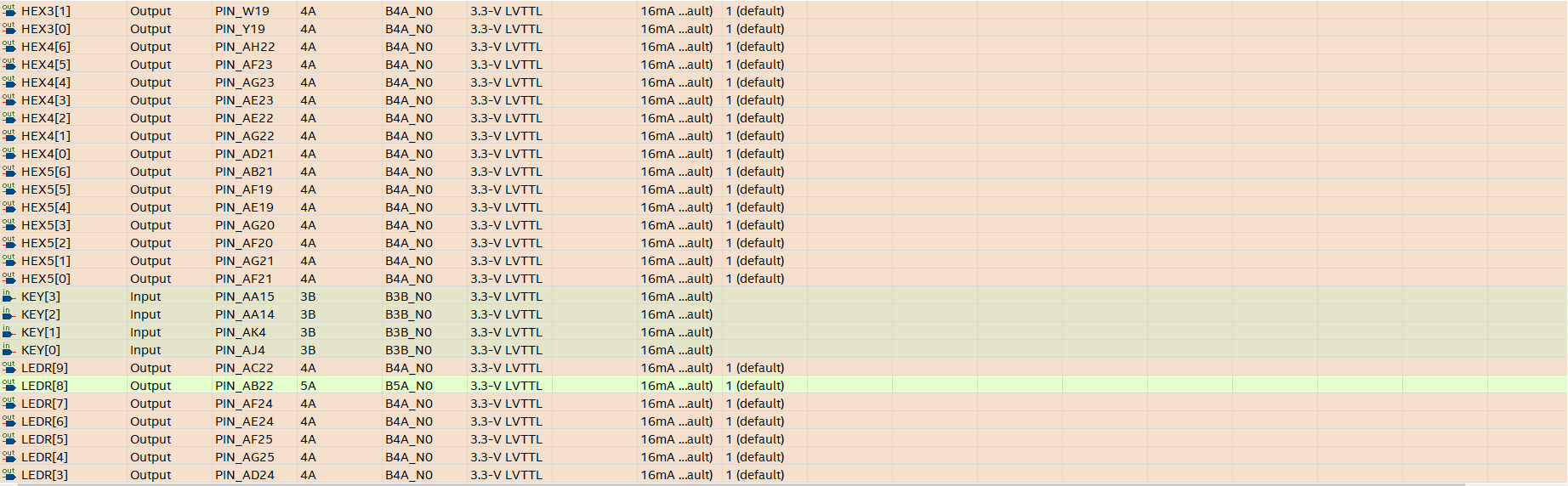
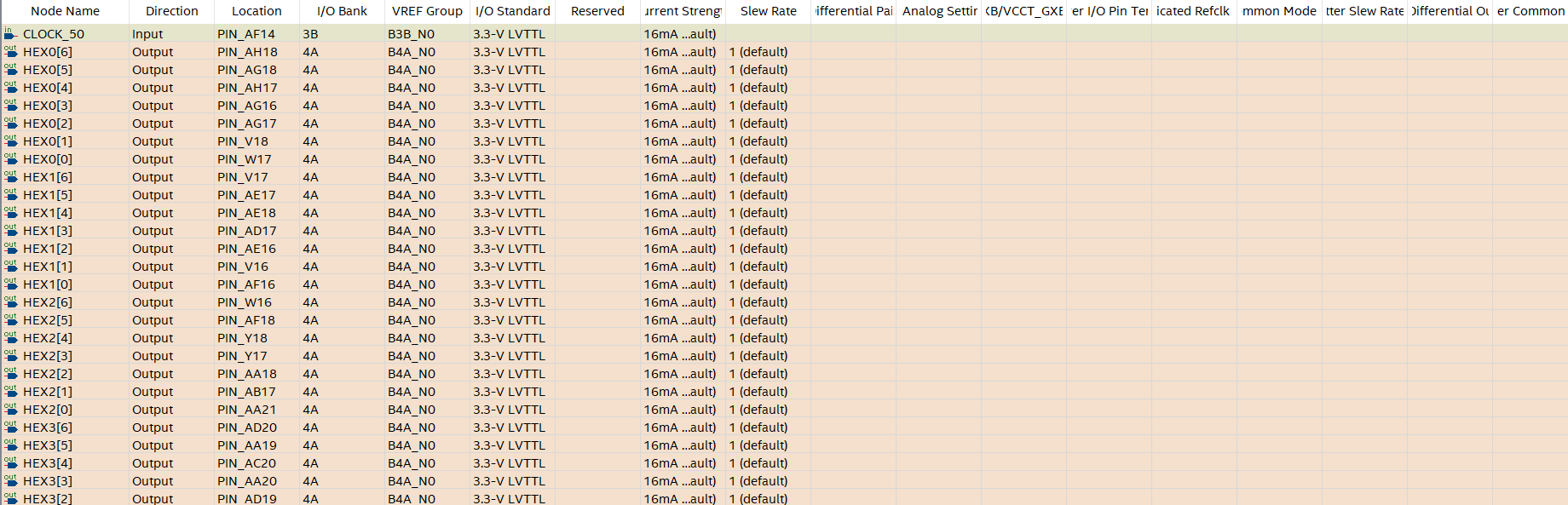
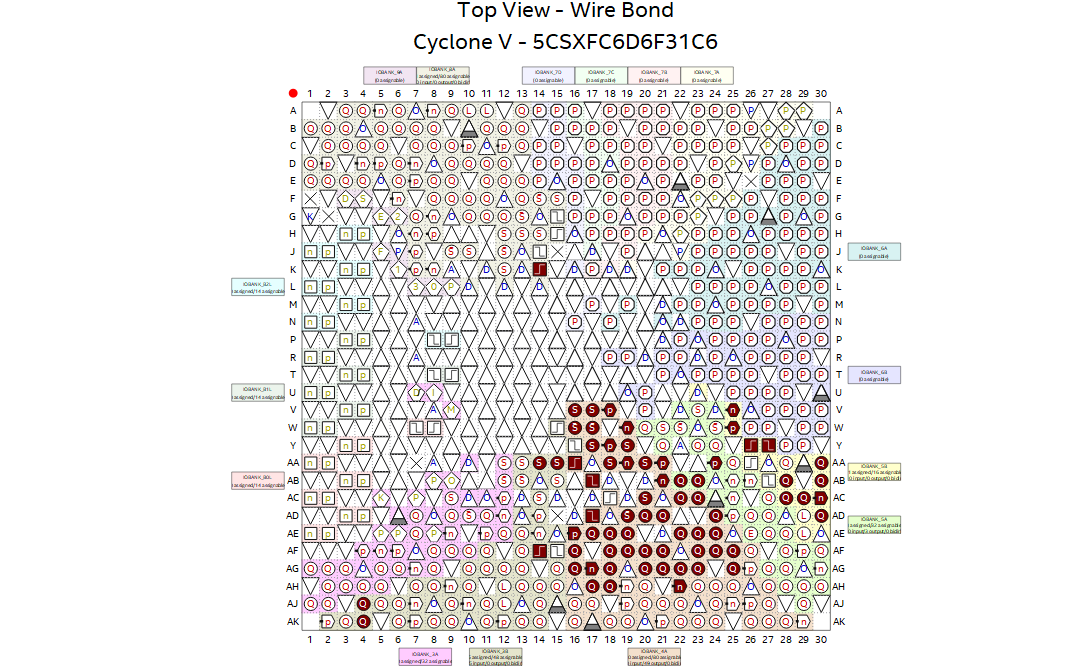


图2-2-3引脚分配图

###### 2.2.7全编译

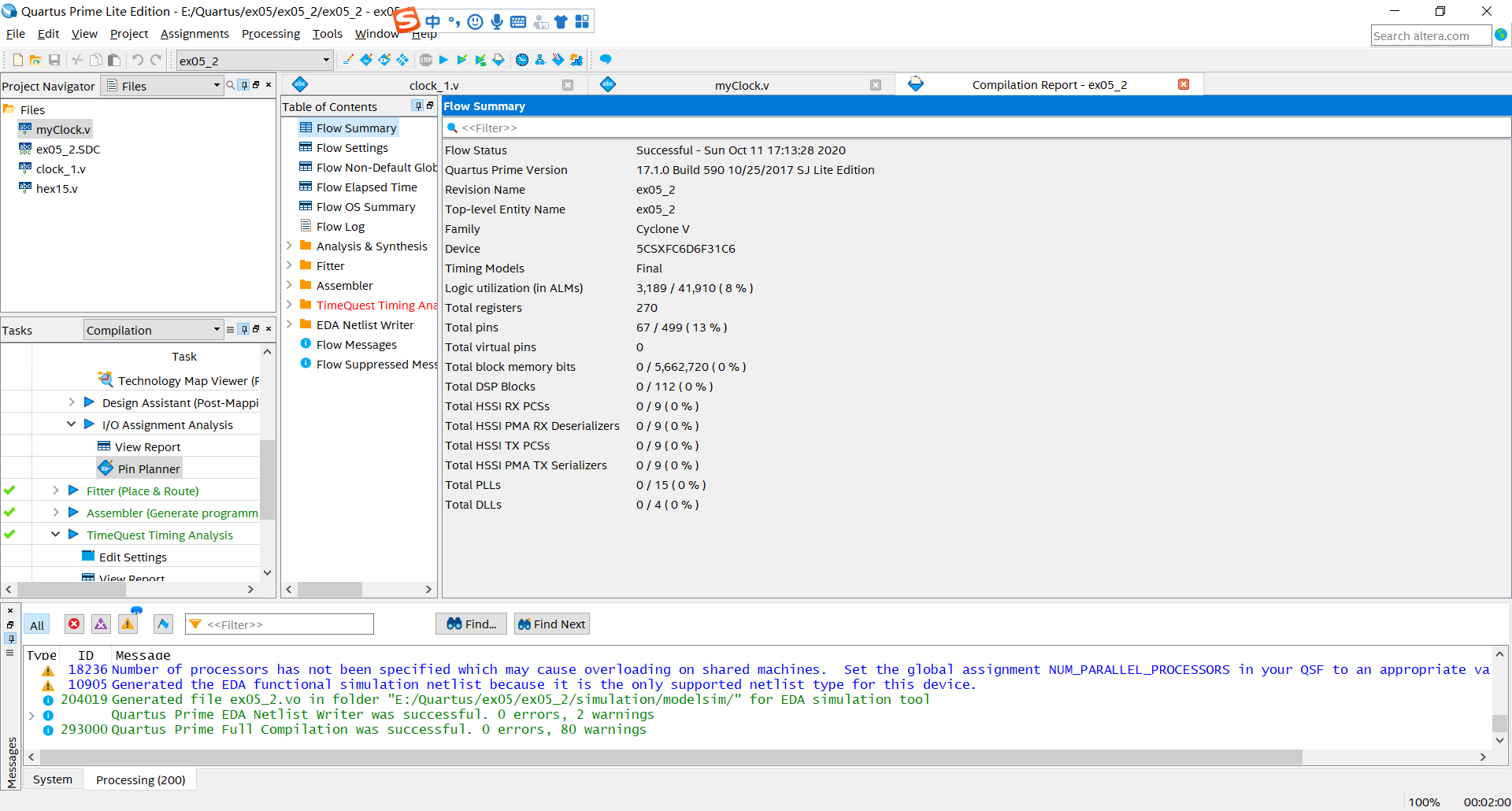


图2-2-4全编译成功

#### 实验总结

本次实验主要学习实践了计数器的设计和应用. 计数器可以用作分频器将开发板的50MHz时钟信号转变成本次实验中需要的1Hz时钟信号. 设计时需注意输入信号间的时序关系及可能会产生的影响.