# 数字电路与数字系统实验

# 实验六

# 寄存器

计算机科学与技术系

191220008 陈南曈

924690736@qq.com

2020.9.30

# 一、实验目的

寄存器也是最常用的时序逻辑电路，是一种存储电路。寄存器的存储电路是由锁存器或触发器构成的，因为一个锁存器或触发器能存储 1 位二进制数，所以由 N 个锁存器或触发器可以构成 N 位寄存器。

本实验通过介绍几中常用寄存器的设计方法，复习寄存器的原理，学习寄存器和常用的移位寄存器的设计。

## （一）算术移位和逻辑移位寄存器

这里的算术移位是指考虑到符号位的移位，算术移位要保证符号位不改变，算术左移同逻辑左移一样，算术右移最左面的空位补符号位。逻辑移位不管是向左移位还是向右移位都是空缺处补 0。循环是将移出去的那一位补充到空出的最高/低位的移位方式。置数是将一个 8 位的数据输入到寄存器中，即给寄存器赋一个初始值。

用 Verilog HDL 语言很容易描述出移位寄存器，如：

```
Q <= {Q[0], Q[7:1]};   //循环右移
Q <= {Q[7], Q[7:1]};   //算术右移
```

请根据下表，用 Verilog HDL 语言设计一个移位寄存器，并进行仿真查看移位寄存器的功能。

| 控制位 | 工作方式 |
|---|---|
| 000 | 清 0 |
| 001 | 置数 |
| 010 | 逻辑右移 |
| 011 | 逻辑左移 |
| 100 | 算术右移 |
| 101 | 左端串行输入 1 位值，并行输出 8 位值 |
| 110 | 循环右移 |
| 111 | 循环左移 |

## （二）利用移位寄存器实现随机数发生器

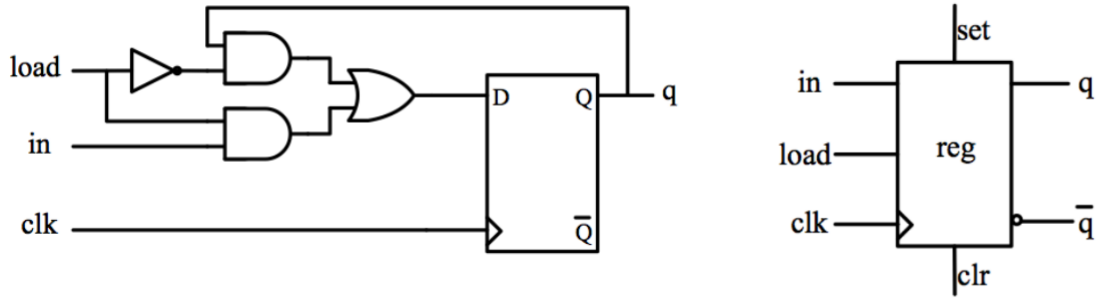
我们可以利用 8 位移位寄存器来实现一个简单的随机数发生器。参考教科书第 534 页 LFSR 反馈方程设计一个 n=8，共有 255 种状态的随机数发生器。

请将 8 位二进制数以十六进制显示在数码管上，在 DE10-Standard 开发板上观察生成的随机数序列。系统需要能够自启动。

# 二、实验原理（知识背景）

## 1、寄存器

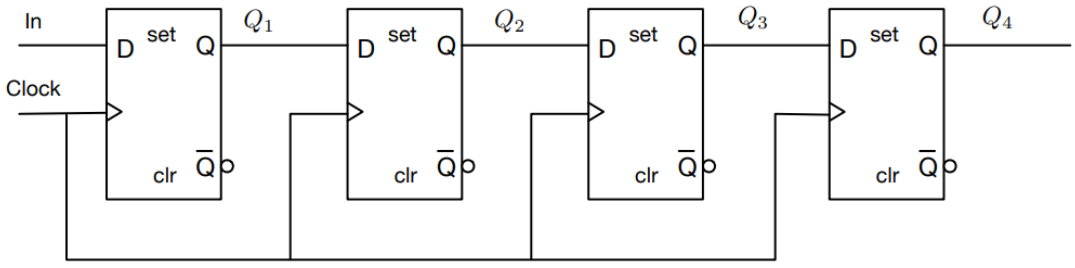
D 触发器可以用于存储比特信号，给 D 触发器加上置数功能就变成了一位寄存器，如图所示。由图中可以看出，如果 load 信号为 1，则输入信号 in 被送入或门中，或门的另一个输入端为 0，此时 D=in，所以在下一个时钟里 q=in。当 load 值为 0 时，q 值被反馈到或门中，或门的另一个输入值为 0，此时 D=q，因此在下一个时钟周期里 q 值保持先前的值不变。




## 2、移位寄存器

移位寄存器是一类寄存器，它在时钟的触发沿，根据其控制信号，将存储在其中的数据向某个方向移动一位。移位寄存器也是数字系统的常用器件。

下图是一个由 4 个 D 触发器构成的简单向右移位寄存器，数据从移位寄存器的左端输入，每个触发器的内容在时钟的正跳变沿（上升沿）将数据传到下一个触发器。下表是一个此移位寄存器的序列传递实例。



| | In | Q1 | Q2 | Q3 | Q4=Out |
|---|---|---|---|---|---|
| t0 | 1 | 0 | 0 | 0 | 0 |
| t1 | 0 | 1 | 0 | 0 | 0 |
| t2 | 1 | 0 | 1 | 0 | 0 |
| t3 | 1 | 1 | 0 | 1 | 0 |
| t4 | 1 | 1 | 1 | 0 | 1 |
| t5 | 0 | 1 | 1 | 1 | 0 |
| t6 | 0 | 0 | 1 | 1 | 1 |
| t7 | 0 | 0 | 0 | 1 | 1 |

# 三、实验环境/器材等

1) 软件环境:

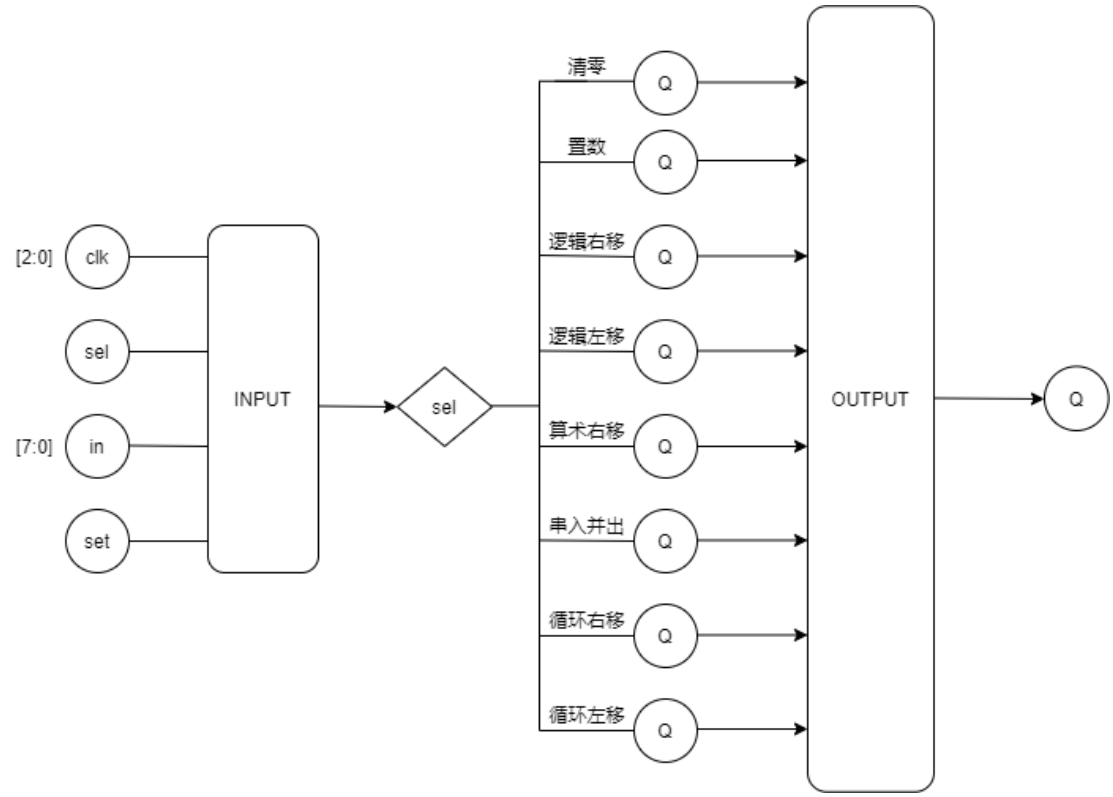
Quartus (Quartus Prime 17.1) Lite Edition

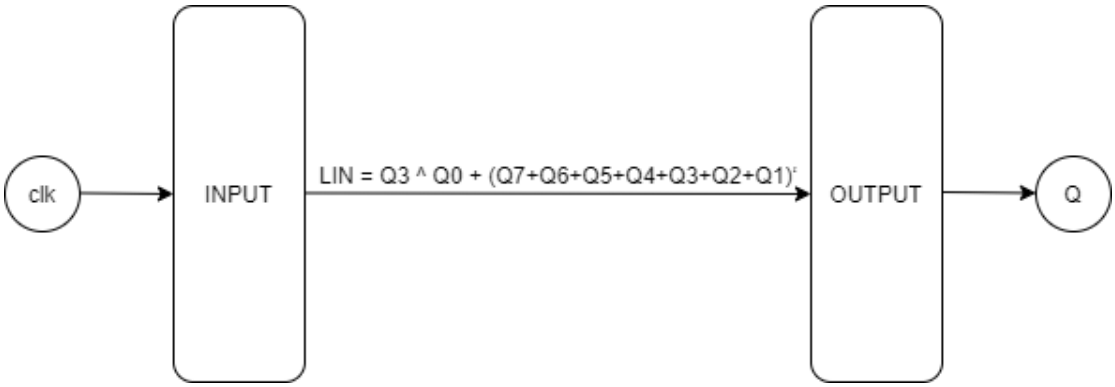
2) 硬件环境:

DE10-Standard 开发板
FPGA 部分:
Intel Cyclone V SE 5CSXFC6D6 F31C6N
- 110K 逻辑单元
- 5,761Kbit RAM

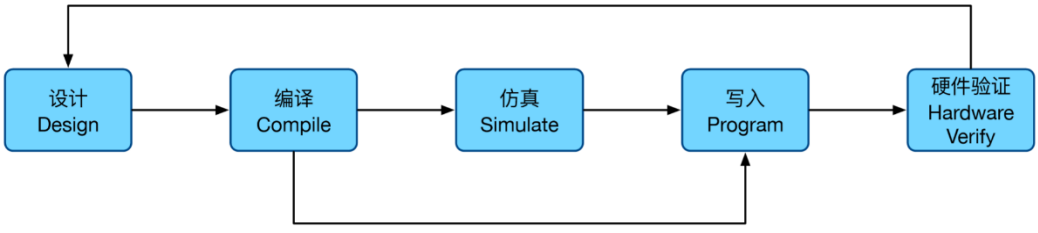
HPS 部分:
Dual-core ARM Cortex A9
- 925MHz
- 1GB DDR

# 四、程序代码或流程图


[2:0]
clk
sel
[7:0]
in
set
INPUT
sel
清零
置数
逻辑右移
逻辑左移
算术右移
串入并出
循环右移
循环左移
Q
OUTPUT
Q


clk
INPUT
LIN = Q3 ^ Q0 + (Q7+Q6+Q5+Q4+Q3+Q2+Q1)'
OUTPUT
Q

## 五、实验步骤/过程


设计
Design
编译
Compile
仿真
Simulate
写入
Program
硬件验证
Hardware
Verify

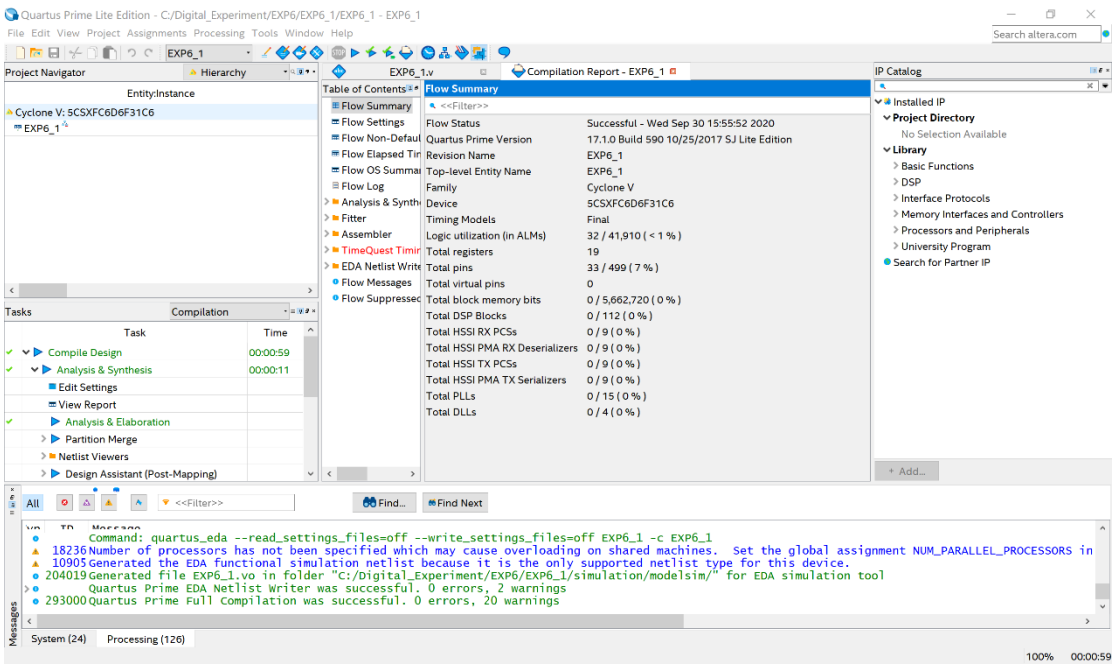
# 1、移位寄存器

设计：

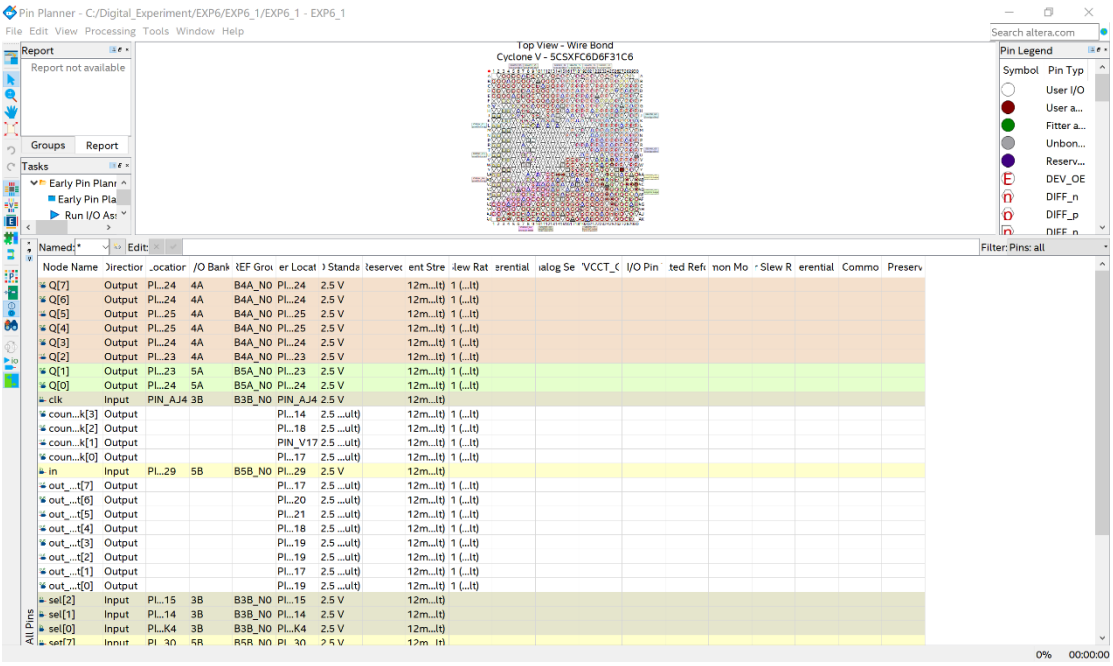
```
module EXP6_1(clk, sel, in, set, Q, count_clk, out_eight);
    input clk;
    input [2:0] sel;
    input in;
    input [7:0] set;

    output reg [7:0] Q = 0;
    output reg [3:0] count_clk = 0;
    output reg [7:0] out_eight = 0;

    always @(posedge clk)
    begin
      case (sel)
        0: begin Q <= 0; count_clk = 0; end
        1: begin Q <= set; count_clk = 0; end
        2: begin Q <= {1'b0, Q[7:1]}; count_clk = 0; end
        3: begin Q <= {Q[6:0], 1'b0}; count_clk = 0; end
        4: begin Q <= {Q[7], Q[7:1]}; count_clk = 0; end
        5: ;
        6: begin Q <= {Q[0], Q[7:1]}; count_clk = 0; end
        7: begin Q <= {Q[6:0], Q[7]}; count_clk = 0; end
        default: Q <= Q;
      endcase
      if (sel == 5)
      begin
        out_eight[count_clk] = in;
        if(count_clk == 7)
          Q <= out_eight;
        count_clk = (count_clk + 1) % 8;
      end
    end

endmodule
```

测试：无

编译：

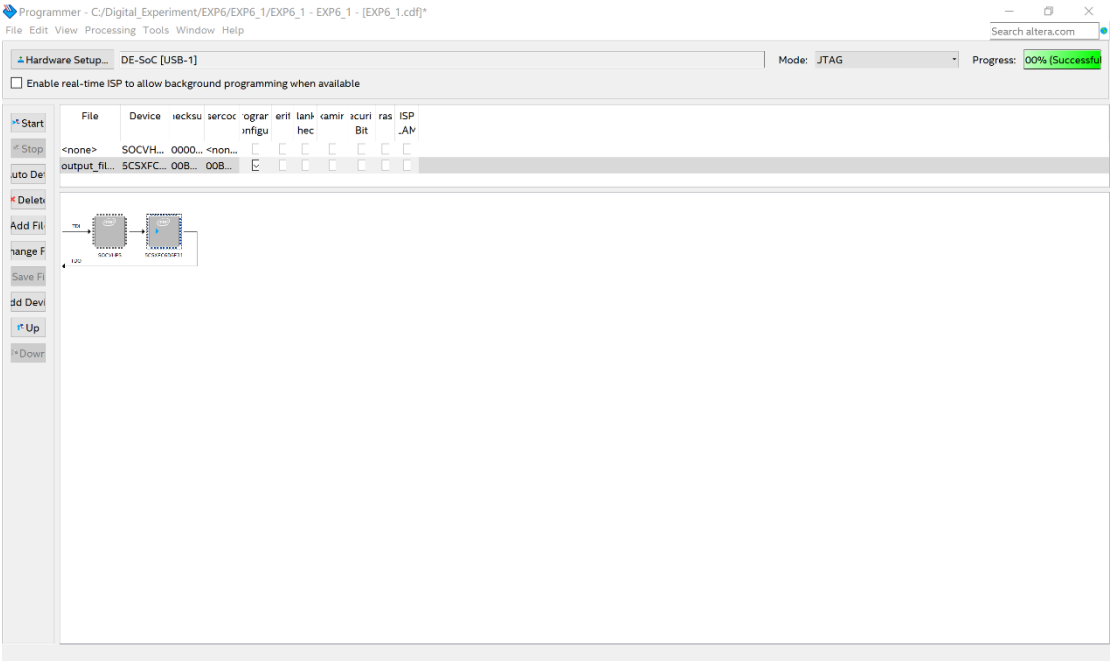


引脚分配：

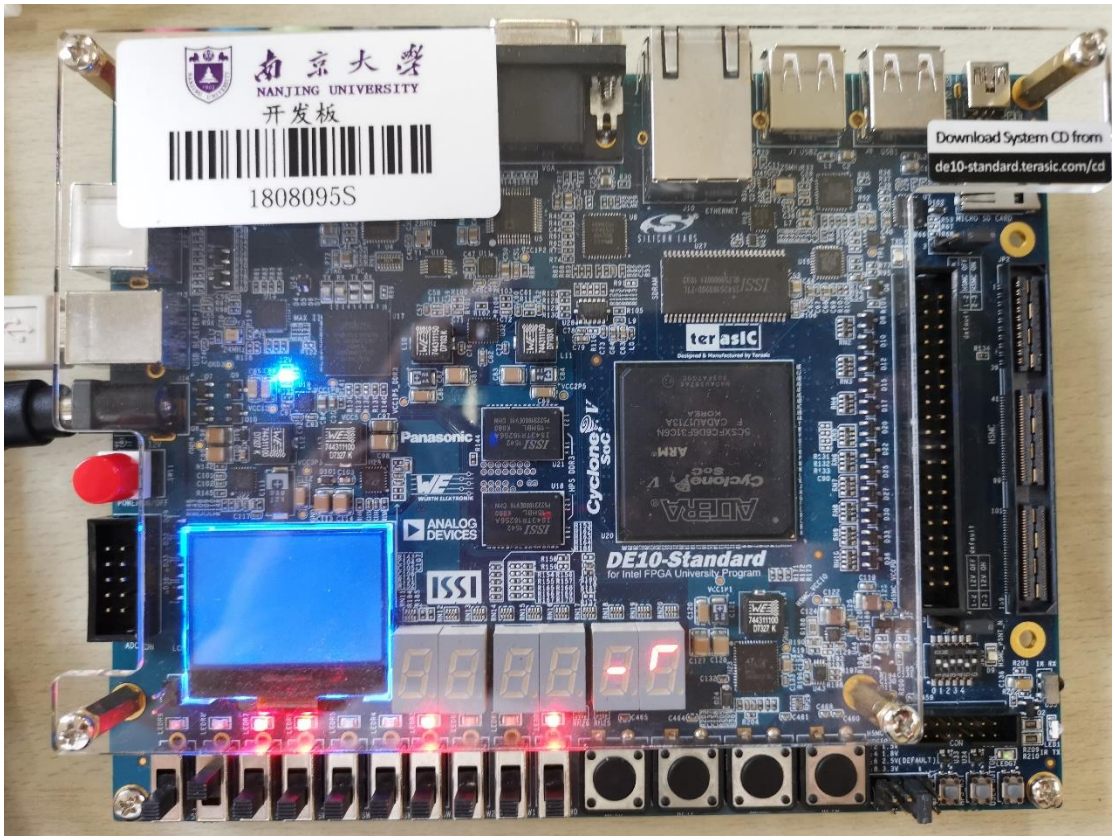

Pin Planner - C:/Digital_Experiment/EXP6/EXP6_1/EXP6_1 - EXP6_1
File Edit View Processing Tools Window Help
Search altera.com
Report
Report not available
Groups
Report
Tasks
Early Pin Planr
Early Pin Pla
Run I/O As
Top View - Wire Bond
Cyclone V - 5CSXFC6D6F31C6
Pin Legend
Symbol
Pin Typ
User I/O
User a...
Fitter a...
Unbon...
Reserv...
DEV_OE
DIFF_n
DIFF_p
Named:
Edit:
Filter: Pins: all
Node Name
Q[7]
Q[6]
Q[5]
Q[4]
Q[3]
Q[2]
Q[1]
Q[0]
clk
coun...k[3]
coun...k[2]
coun...k[1]
coun...k[0]
in
out_...t[7]
out_...t[6]
out_...t[5]
out_...t[4]
out_...t[3]
out_...t[2]
out_...t[1]
out_...t[0]
sel[2]
sel[1]
sel[0]
All Pins
0%
00:00:00

仿真：无

写入：


Programmer - C:/Digital_Experiment/EXP6/EXP6_1/EXP6_1 - EXP6_1 - [EXP6_1.cdf]*
File Edit View Processing Tools Window Help
Search altera.com
Hardware Setup...
DE-SoC [USB-1]
Mode: JTAG
Progress:
00% (Successful
Enable real-time ISP to allow background programming when available
Start
Stop
Add Fil
Up
File
Device
<none>
SOCVH...
output_fil...
5CSXFC...
Bit

硬件验证:



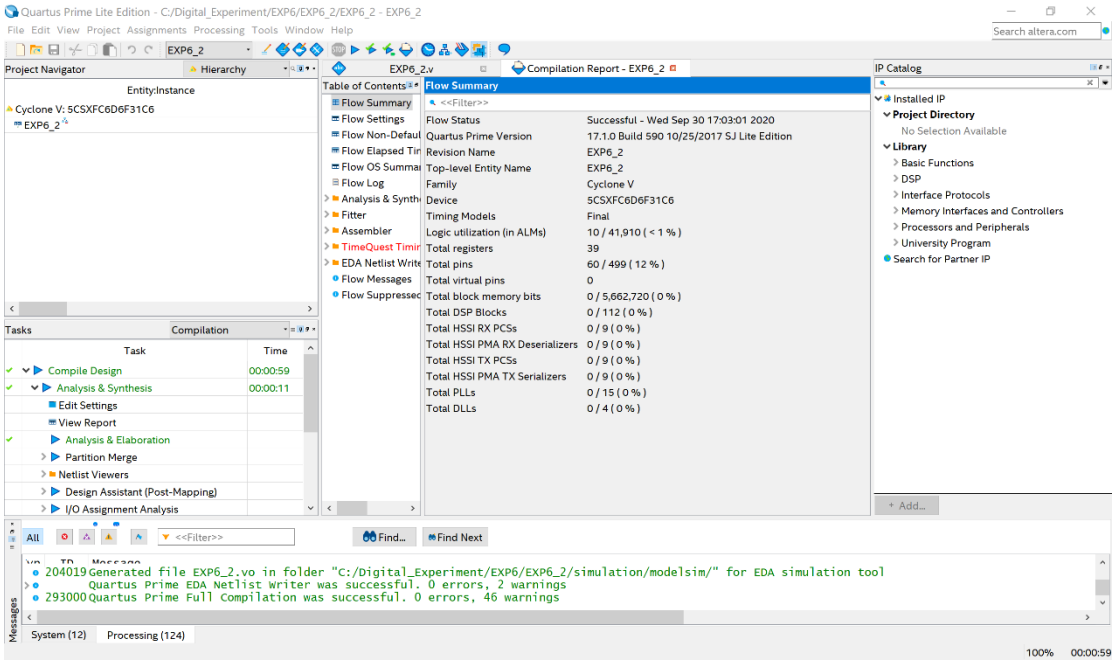

## 2、随机数发生器

设计:

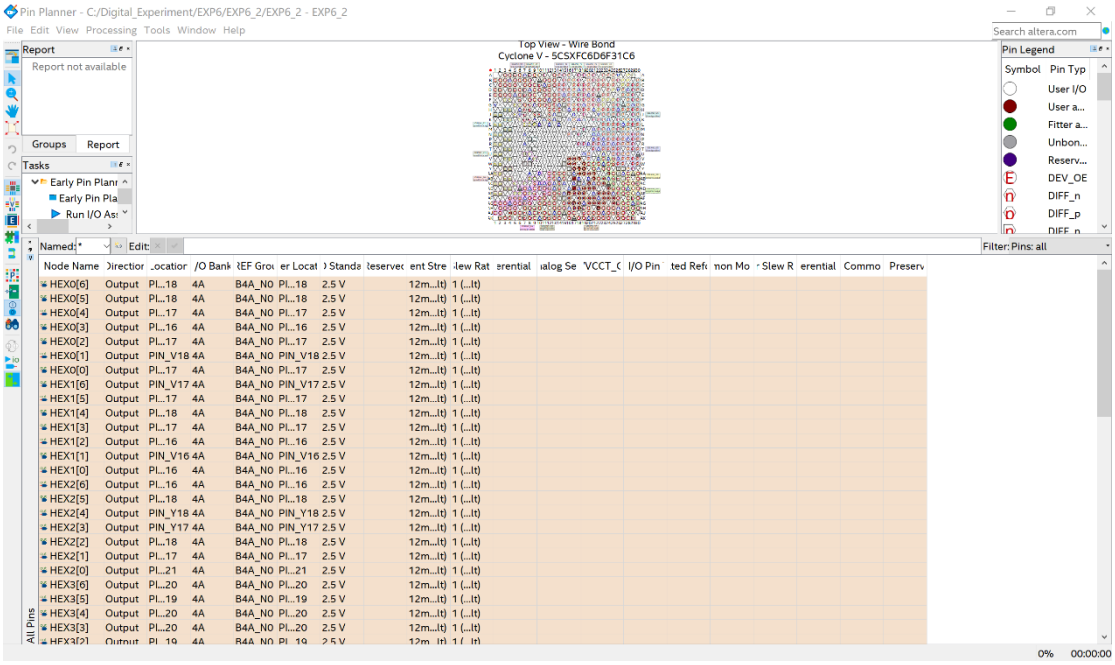
```
module EXP6_2(clk, Q, LIN, digit0, digit1, HEX0, HEX1, HEX2, HEX3, HEX4, HEX5);
    input clk;

    output reg [7:0] Q = 0;
    output reg LIN;
    output reg [3:0] digit0;
    output reg [3:0] digit1;
    output reg [6:0] HEX0 = 64;
    output reg [6:0] HEX1 = 64;
    output reg [6:0] HEX2 = 127;
    output reg [6:0] HEX3 = 127;
    output reg [6:0] HEX4 = 127;
    output reg [6:0] HEX5 = 127;

    always @(posedge clk)
    begin
      LIN = (Q[3] ^ Q[0]) | ~(Q[0] || Q[1] || Q[2] || Q[3] || Q[4] || Q[5] || Q[6] ||
      Q = {LIN, Q[7:1]};
      digit0 = Q[3:0];
      digit1 = Q[7:4];
      case (digit0)
        0: HEX0 = 64;
        1: HEX0 = 121;
        2: HEX0 = 36;
        3: HEX0 = 48;
        4: HEX0 = 25;
        5: HEX0 = 18;
        6: HEX0 = 2;
        7: HEX0 = 120;
        8: HEX0 = 0;
        9: HEX0 = 16;
        10: HEX0 = 8;
        11: HEX0 = 3;
        12: HEX0 = 70;
        13: HEX0 = 33;
```

测试：无

编译：

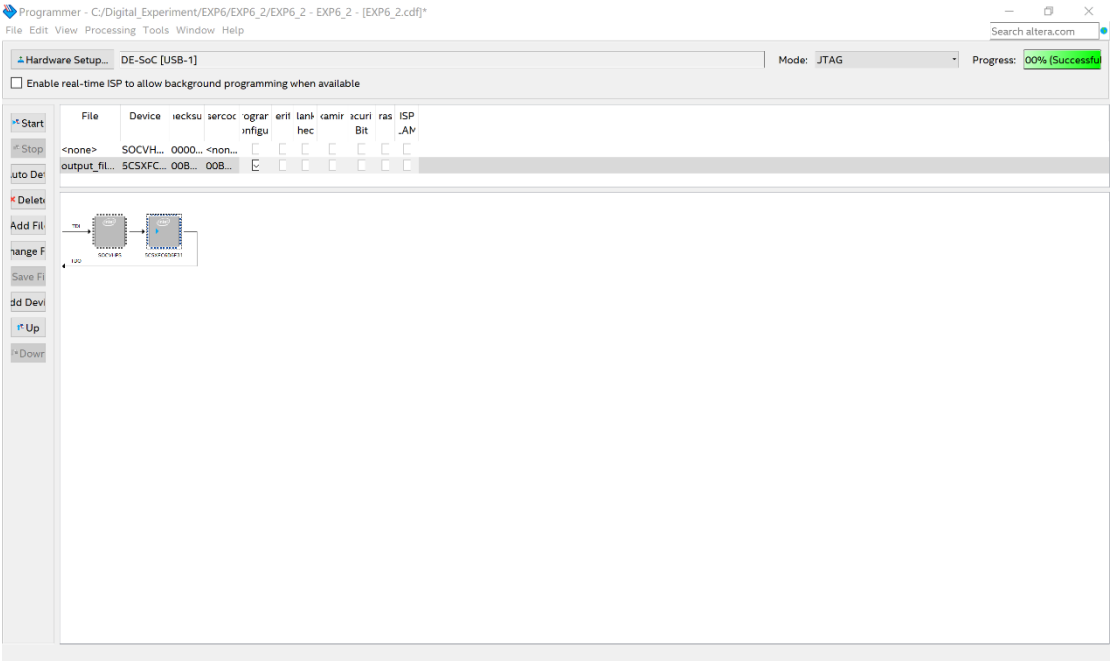

Quartus Prime Lite Edition - C:/Digital_Experiment/EXP6/EXP6_2/EXP6_2 - EXP6_2
Flow Summary
Flow Status Successful - Wed Sep 30 17:03:01 2020
Quartus Prime Version 17.1.0 Build 590 10/25/2017 SJ Lite Edition
Revision Name EXP6_2
Top-level Entity Name EXP6_2
Family Cyclone V
Device 5CSXFC6D6F31C6
Timing Models Final
Logic utilization (in ALMs) 10 / 41,910 ( < 1 % )
Total registers 39
Total pins 60 / 499 ( 12 % )
Total virtual pins 0
Total block memory bits 0 / 5,662,720 ( 0 % )
Total DSP Blocks 0 / 112 ( 0 % )
Total HSSI RX PCSs 0 / 9 ( 0 % )
Total HSSI PMA RX Deserializers 0 / 9 ( 0 % )
Total HSSI TX PCSs 0 / 9 ( 0 % )
Total HSSI PMA TX Serializers 0 / 9 ( 0 % )
Total PLLs 0 / 15 ( 0 % )
Total DLLs 0 / 4 ( 0 % )
Compile Design 00:00:59
Analysis & Synthesis 00:00:11
204019 Generated file EXP6_2.vo in folder "C:/Digital_Experiment/EXP6/EXP6_2/simulation/modelsim/" for EDA simulation tool
293000 Quartus Prime Full Compilation was successful. 0 errors, 46 warnings

引脚分配：

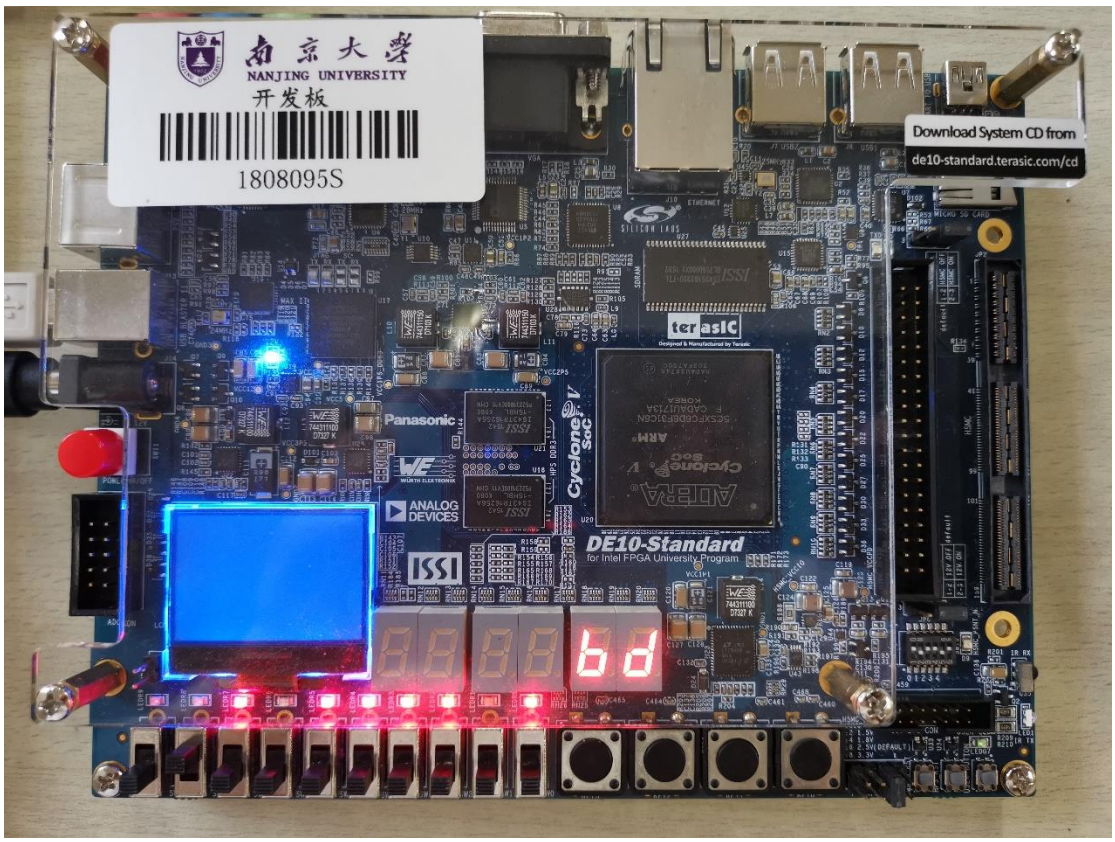

Pin Planner - C:/Digital_Experiment/EXP6/EXP6_2/EXP6_2 - EXP6_2
Top View - Wire Bond
Cyclone V - 5CSXFC6D6F31C6
Pin Legend

仿真：无

写入：


Programmer - C:/Digital_Experiment/EXP6/EXP6_2/EXP6_2 - EXP6_2 - [EXP6_2.cdf]*
File Edit View Processing Tools Window Help
Search altera.com
Hardware Setup...
DE-SoC [USB-1]
Mode: JTAG
Progress: 00% (Successful
Enable real-time ISP to allow background programming when available
Start
Stop
Add Fil
Save Fi
Up
File
Device
<none>
SOCVH...
output_fil...
5CSXFC...

硬件验证：


南京大学
NANJING UNIVERSITY
开发板
1808095S
Download System CD from
de10-standard.terasic.com/cd
terasIC
Panasonic
ANALOG DEVICES
ISSI
DE10-Standard
for Intel FPGA University Program
bd

# 六、测试方法

该试验不易通过 Test Bench 进行，测试效率过低，故选择直接在实验板上验证

# 七、实验结果

实验板上的实际结果与预期结果一致

# 八、思考题

**1、生成的伪随机数序列仍然有一定的规律，如何能够生成更加复杂的伪随机数序列？**

可以增加触发器的个数，产生更多的随机状态，反馈方程也更复杂，使得其更“随机”

# 八、实验中遇到的问题及解决办法

1、发现 case 语句里不能使用 ifelse 语句

解决办法：将需要使用 ifelse 语句的选项，单独放在 case 语句后讨论

2、阻塞和非阻塞的运用容易混淆，两者经常误用

解决办法：画一个简单的时序图，弄清变量的变化顺序

# 九、实验得到的启示

对于过程中变量的变化顺序要很清晰，不然容易混用“=”和“<=”

# 十、意见和建议

实验五难度较大，但实验六难度较小，可以平均一下