**中国科学技术大学计算机学院**

**《数字电路实验》报告**



实验题目：信号处理及有限状态机

学生姓名：\_\_\_\_\_\_\_\_徐奥\_\_\_\_\_\_\_\_

学生学号：\_\_\_\_\_PB20061343\_\_\_\_\_

完成日期：\_\_2021年12月8日\_\_

计算机实验教学中心制

2020年09月

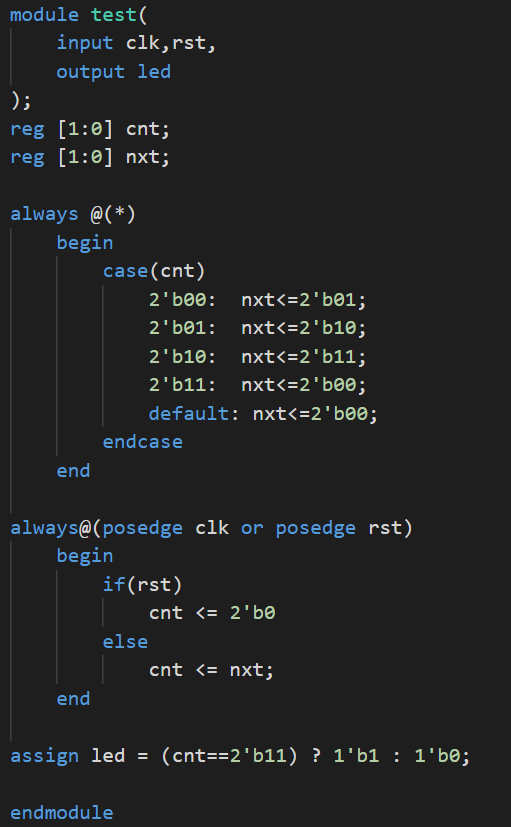
【实验题目】

本次实验中，我们将介绍几种常用的数字信号处理技巧并学习一种数字电路开发中非常重要的设计方法：有限状态机（FSM：Finite State Machine）

【实验练习】

**题目1.** 在不改变电路功能和行为的前提下，将前面Step5中的代码 改写成三段式有限状态机的形式，写出完整的Verilog代码

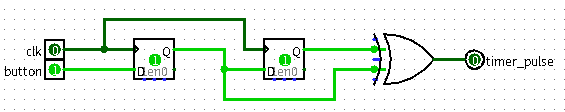
代码如下：



**图1**

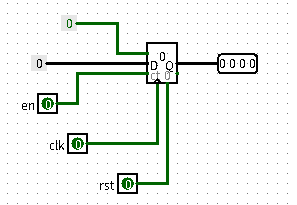
**题目2.** 请在Logisim中设计一个4bit位宽的计数器电路，clk信号为计数器时钟，复位时（rst==1）计数值为0，在输入信号sw电平发生变化时，计数值cnt加1，即在sw信号上升沿时刻和下降沿时刻各触发一次计数操作，其余时刻计数器保持不变

首先搭建一个电路，实现通过按键信号生成一个时钟周期宽度的脉冲信号，即sw发生变化时，会产生一个时钟周期宽度的脉冲信号，电路如下图：



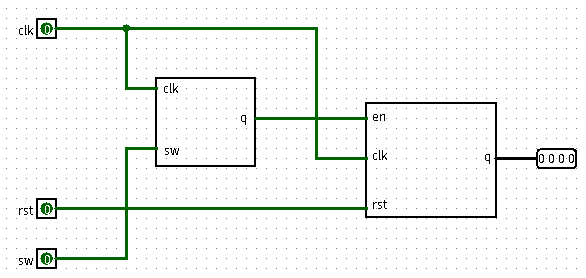
**图2**

然后搭建4bit位宽的计数器，使用Logisim自带的计数器，调整其为加一计数器，并且rst==1时计数值为0，如下图：



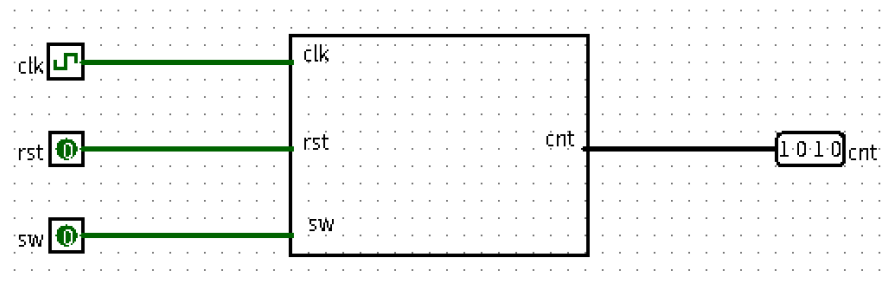
**图3**

将上述两个电路封装，搭建题目要求的计数器，如下图：



**图4**

最后将其封装，在1KHz的时钟频率下实现题目要求，如下图：

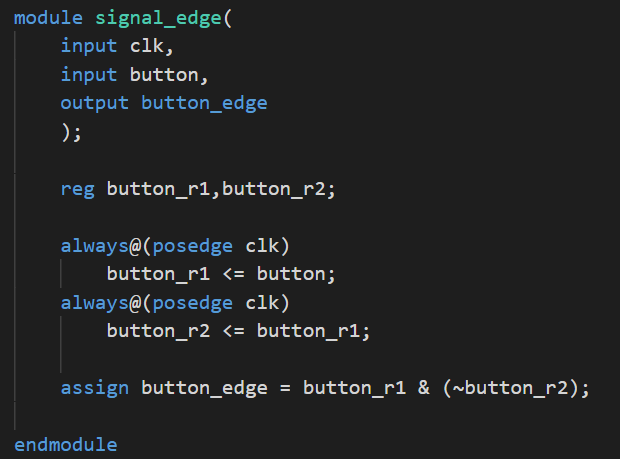


**图5**

**题目3.** 设计一个8位的十六进制计数器，时钟采用板载的100MHz时钟，通过sw[0]控制计数模式，开关为1时为累加模式，为0时为 递减模式，按键控制计数，按下的瞬间根据开关的状态进行累加或递 减计数。计数值用数码管显示，其复位值为“1F”。

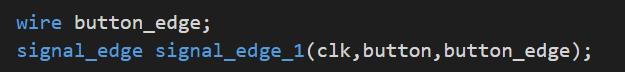
首先，编写Verilog代码。

通过按键button产生一个时钟周期宽度的脉冲信号，使得该信号在button信号的上升沿附近为高电平，其余时间为低电平，代码如下：



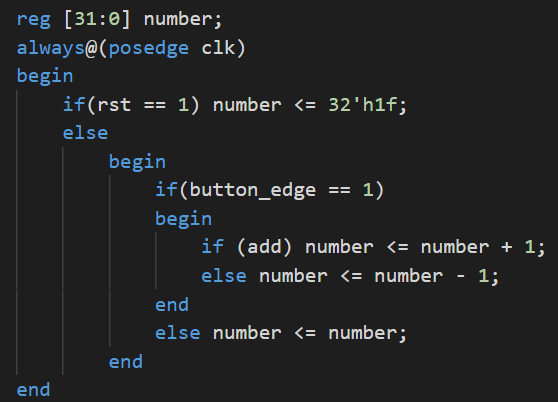
**图6**

该模块将作为子模块，引用代码如下：



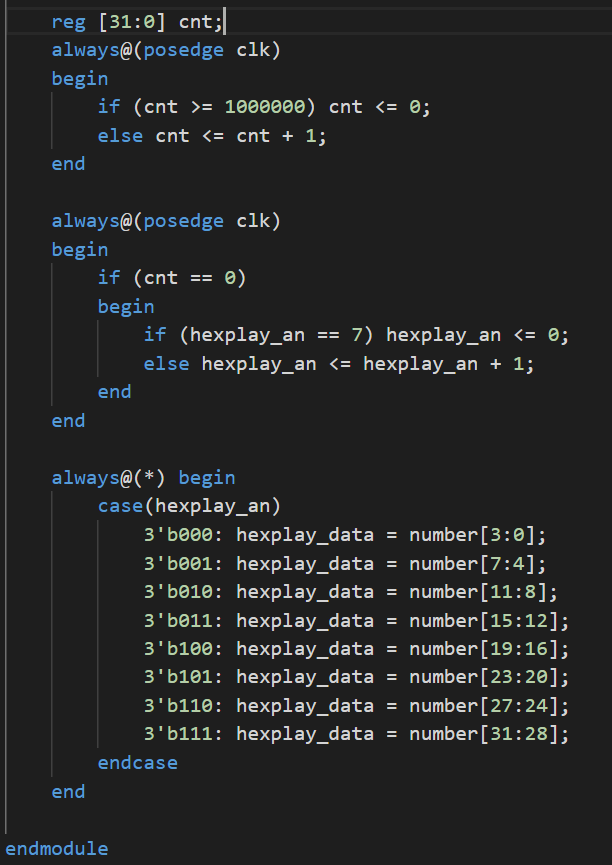
**图7**

对8位十六进制数进行计数，如果标志位（add）为1，那么为加一计数，否则为减一。并且只有当button==1时，才进行计数操作，代码如下：



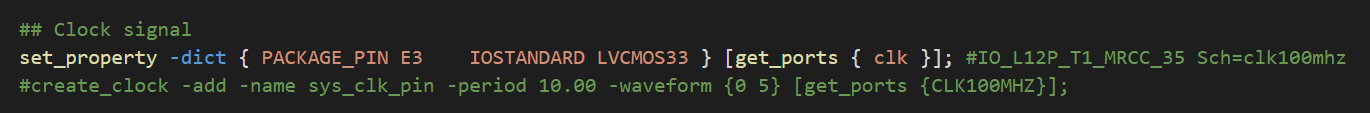
**图8**

利用视觉暂留效应实现同时输出8位十六进制数，代码如下：

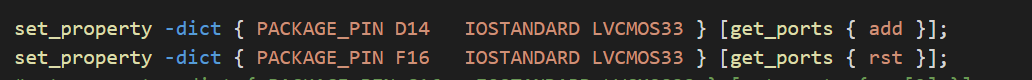


**图9**

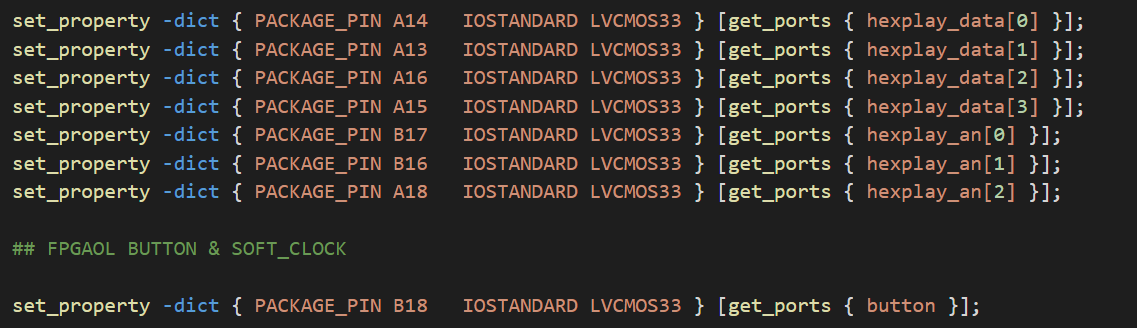
然后，修改引脚约束文件，使用到如下几个引脚：



**图10**

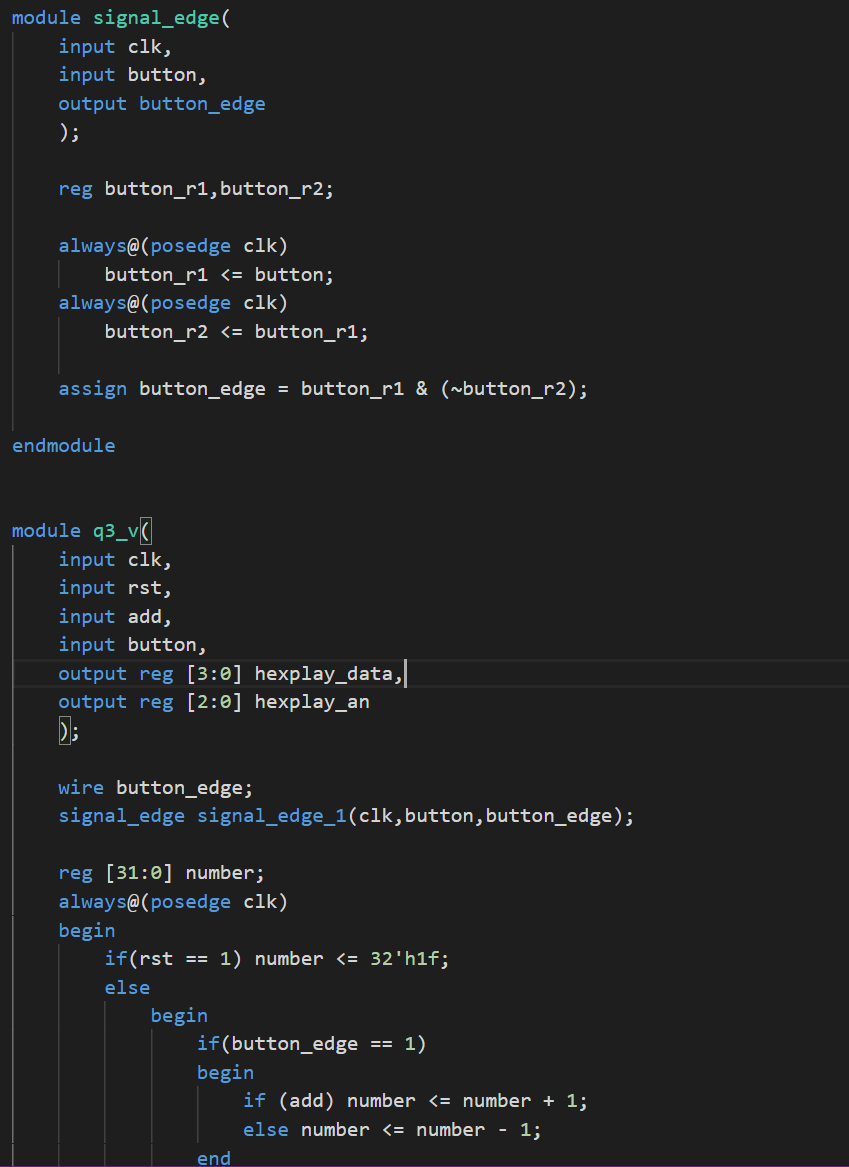


**图11**

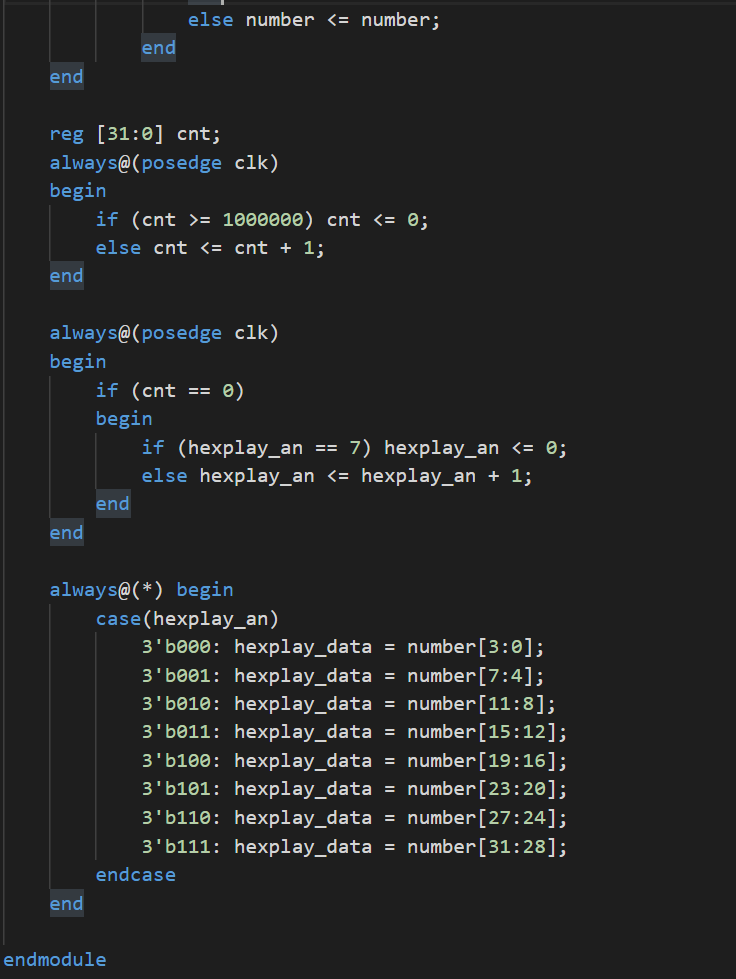


**图12**

完整代码如下：



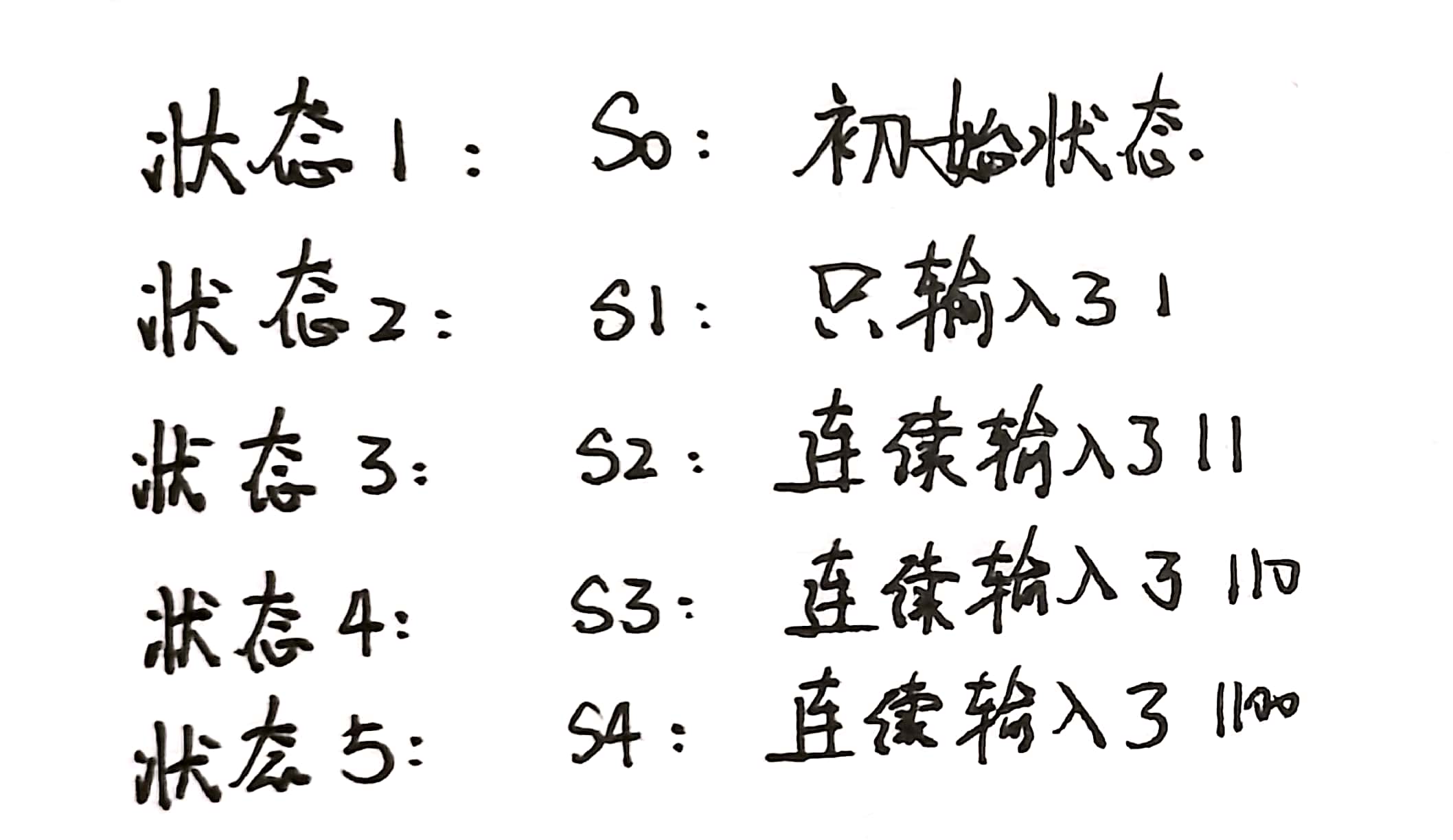
**图13**



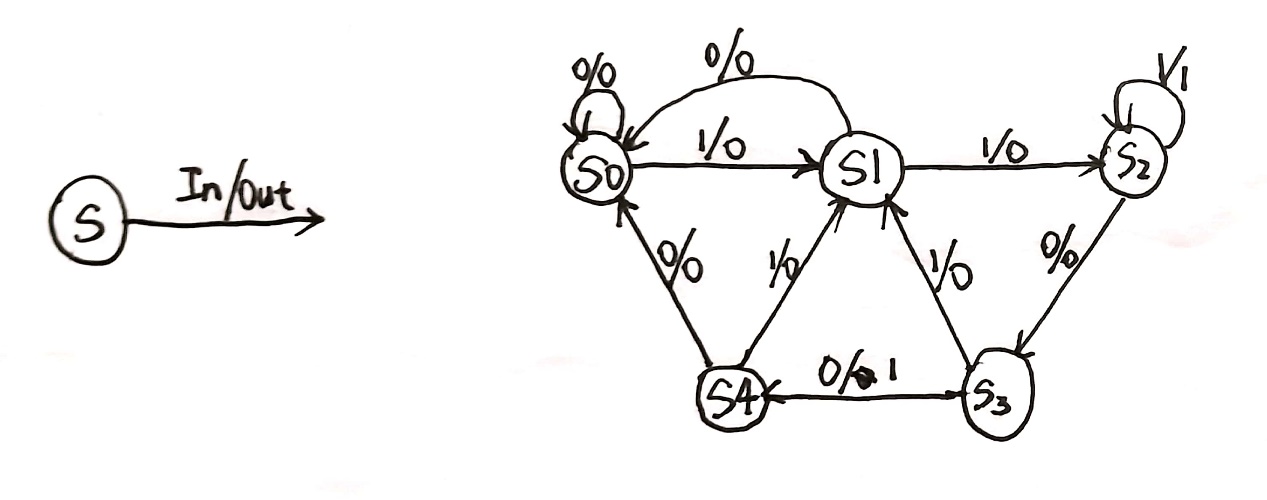
**图14**

**题目4.** 使用有限状态机设计一个序列检测电路，并进行计数，当检 测到输入序列为“1100”时，计数器加一，用一个数码管显示当前状态编码，一个数码管显示检测到目标序列的个数，用4个数码管显示最近输入的4个数值，用sw[0]进行数据的串行输入，按键每按下一次将输入一次开关状态，时钟采用板载的100MHz时钟。要求画出状态跳转图，并在FPGA开发板上实现电路

首先，画出状态跳转图：

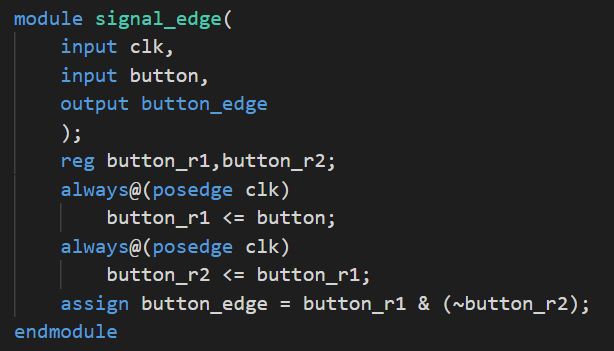


**图15**



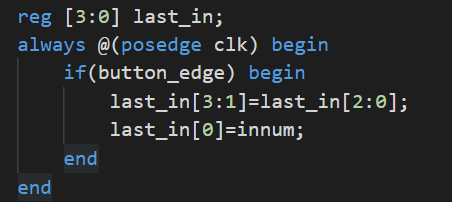
**图16**

通过按键button产生一个时钟周期宽度的脉冲信号，使得该信号在button信号的上升沿附近为高电平，其余时间为低电平，代码如下：



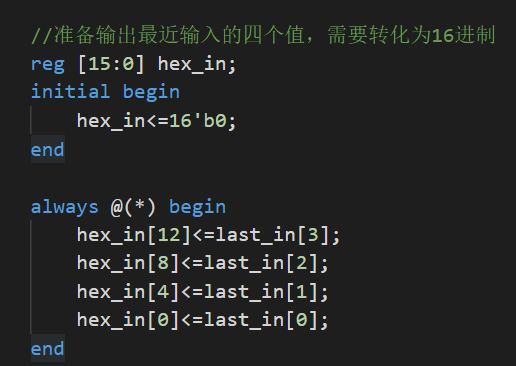
**图17**

保存最近读入的四个数值：



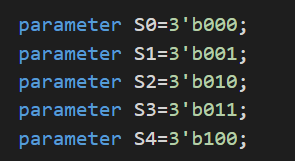
**图18**

并准备将其以十六进制格式输出



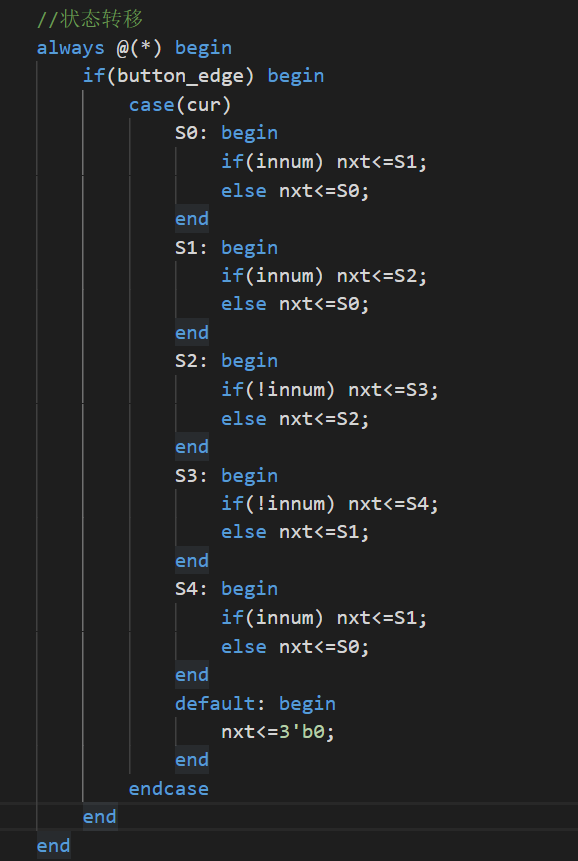
**图19**

将五个状态编码：



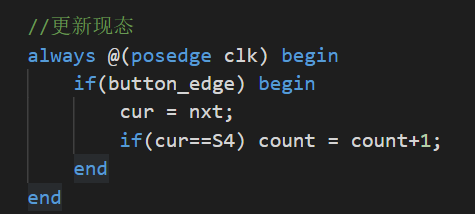
**图20**

求出次态：



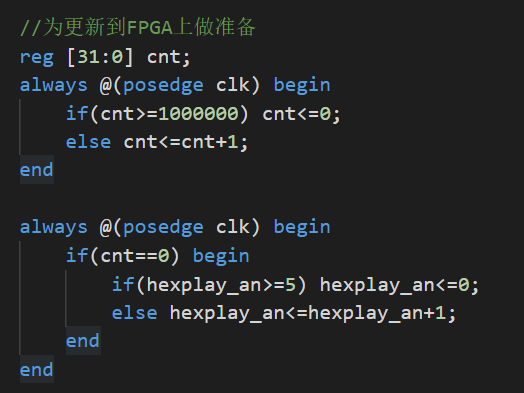
**图21**

在时钟有效边沿状态转移：

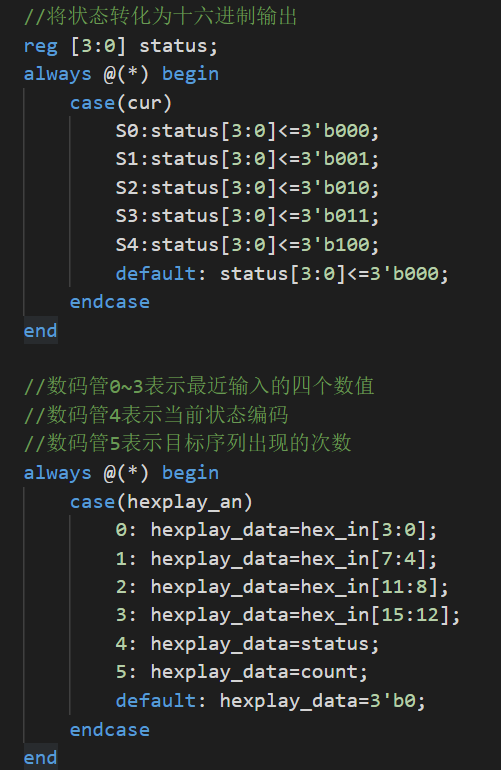


**图22**

为输出到FPGA上，设计刷新与输出：

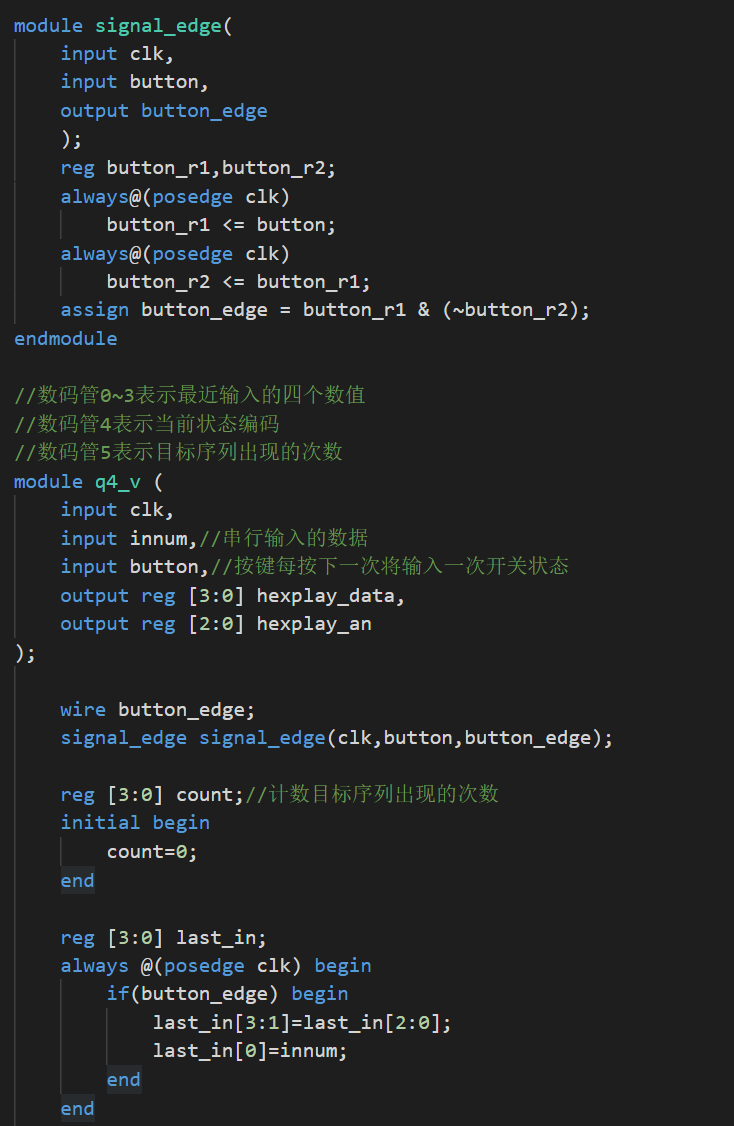


**图23**

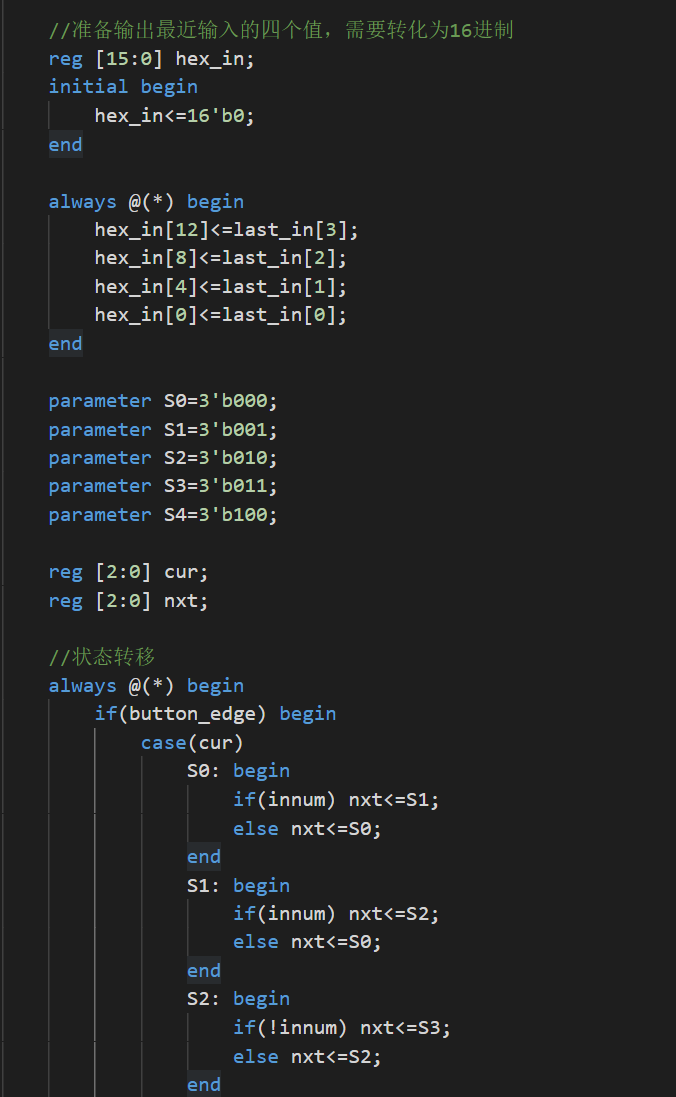


**图24**

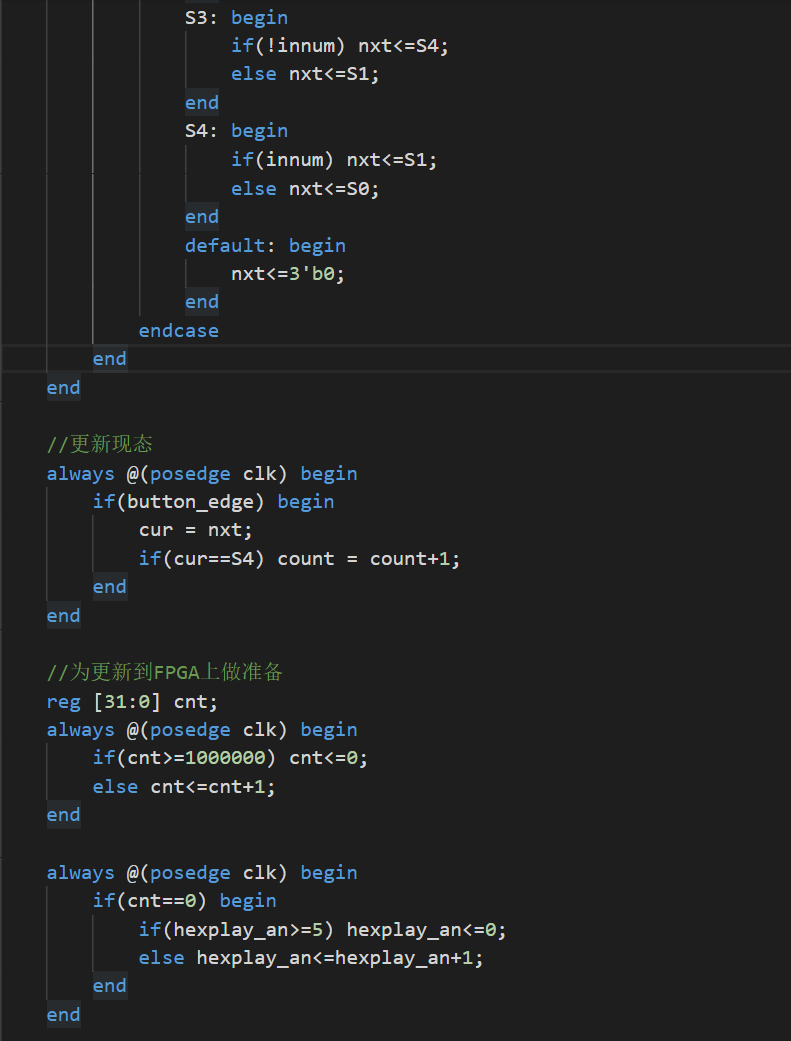
完整代码为：



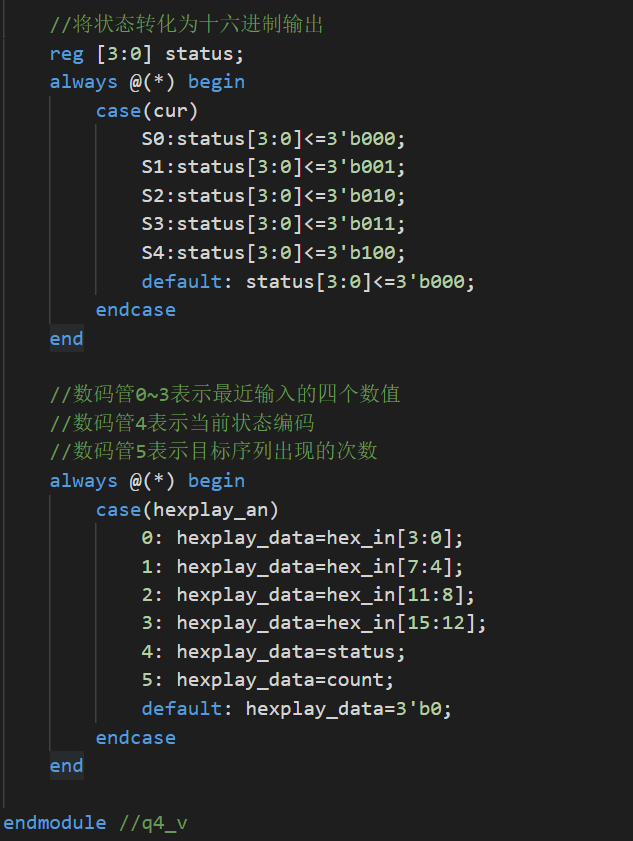
**图25**



**图26**

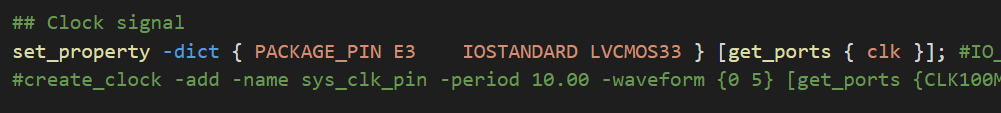


**图27**



**图28**

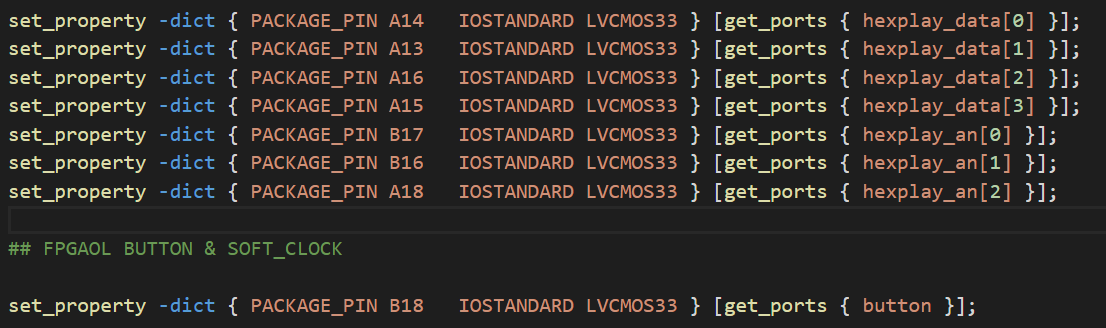
引脚约束文件为：



**图29**



**图30**



**图31**

【总结与思考】

1. 本次实验前三题相对比较简单，第四题在思维量和代码量上都有较大的难度提升
2. 我在第一次写第四题的Verilog代码时，没有在写之前就想清楚各模块作用，导致最后写得比较混乱，无法生成有效的比特流文件，最终是重新写了一遍，并且在第二遍时加上了很多注释，写起来更有条理一些
3. 本次实验量中等偏多
4. 本次实验对于Verilog代码能力有充分的练习