**综合实验报告**

**一、设计内容**

本人设计了一个打地鼠小游戏，在FPGAOL上实现，在该游戏中，七个LED灯管的闪与灭代表地鼠的探头与缩头，玩家快速选定开关并按下按钮代表锤击，只有在一定时间内选定正确位置并按下按钮，才能得分。LED灯管随机闪烁，每按一次按钮就会刷新一次，代表着地鼠的随机探头，一轮游戏时长为30秒，总得分显示在数码管上。本游戏支持查看最近四次游戏得分和历史最高分。

**二、设计思路**

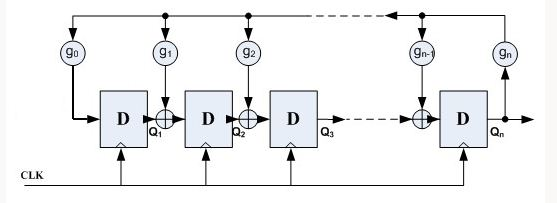
1.确定状态机状态数

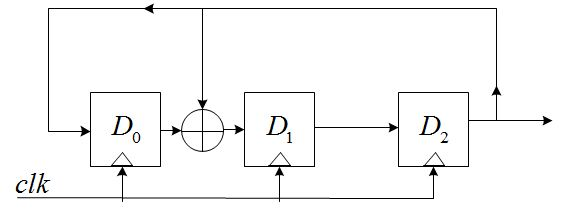
本人设计了一个包含4个状态的状态机，00：菜单（初始界面）；01：游戏；10：近四次分数；11：历史最高分。显然，这四个状态足以满足该游戏设计的需求。于是，我们将sw开关的前两位：sw[0]、sw[1]作为转换状态的入口，按下button时，根据sw[0~1]的值改变状态。综上，这是一个Mealy型状态机。

2.对button信号的处理

由于需要用到button来进行各种操作，如选取模式和传达敲击指令等，所以，我们需要对button信号做处理，在本设计中做了两种处理：去毛刺和取边缘信号，这两种处理都是在前面的实验中学习应用过的，故不在此赘述。

3.取随机数

打地鼠游戏中，地鼠从哪个洞口探头是随机的，因此，本游戏中的LED灯闪烁也应当是随机的，那么，怎么实现这种随机功能呢？   
 本设计中，使用的是由LFSR(线性反馈移位寄存器)引出伪随机数的方法。LFSR是由n个D触发器和若干个异或门交替线性串接组成的，如下图：  
 其中，gn为反馈系数，取值只能为0或1，取为0时表明不存在该反馈之路，取为1时表明存在该反馈之路；n个D触发器最多可以提供2^n-1个状态(不包括全0的状态)，为了保证这些状态没有重复，gn的选择必须满足一定的条件。

下面以n=3，g0=1，g1=1,g2=0,g3=1为例，说明LFSR的特性，具有该参数的LFSR结构如下图：

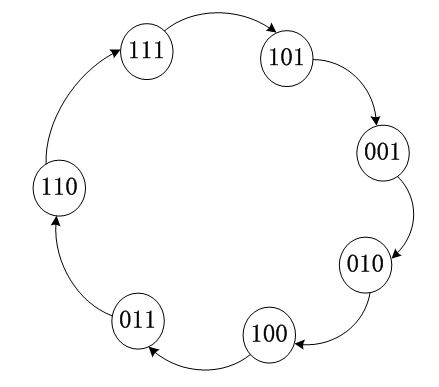
假设在开始时，D2D1D0=111(seed)，那么，当时钟到来时，有：D2=D1\_OUT=1；

D1=D0\_OUT^D2\_OUT=0；

D0=D2\_OUT=1；

即 D2D1D0=101；同理，又一个时钟到来时，可得D2D1D0=001 ……

画出状态转移图如下：



从图可以看出，正好有2^3-1=7个状态，不包括全0；

理解了上图，至少可以得到三条结论：

1)初始状态是由SEED提供的；

2)当反馈系数不同时，得到的状态转移图也不同；必须保证  
 gn===1,否则没有反馈

3)D触发器的个数越多，产生的状态就越多，也就越“随机”

基于以上原理，我们就可以在FPGA上实现伪随机数的生成了，具体的代码在稍后展示。

4.在数码管上同时显示多个数字

使用前面实验中已经学习应用过的时分复用的原理。不做赘述。

**三、代码实现**

分七个模块编写该游戏的Verilog代码，再在顶层代码中引用串联，最终实现了该游戏。模块分布如下：



各模块功能为：

·Whac：顶层模块

·buttonclean：去毛刺模块

·buttonedg：取边沿信号模块

·FSM：状态机模块

·creatpuzzle：生成“地鼠”模块

·answermodule：“锤子”信号接受模块

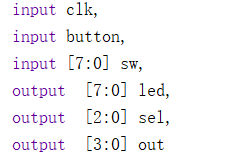
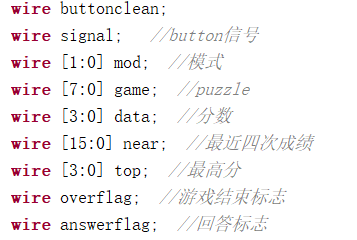
·recordscore：得分记录模块

·dcdtest：数码管显示模块

下面对各模块的关键点做一些阐述：

·Whac

该模块是顶层设计模块，无需阐述太多，下面给出定义的各种信号，注释已在图中：

·buttonclean

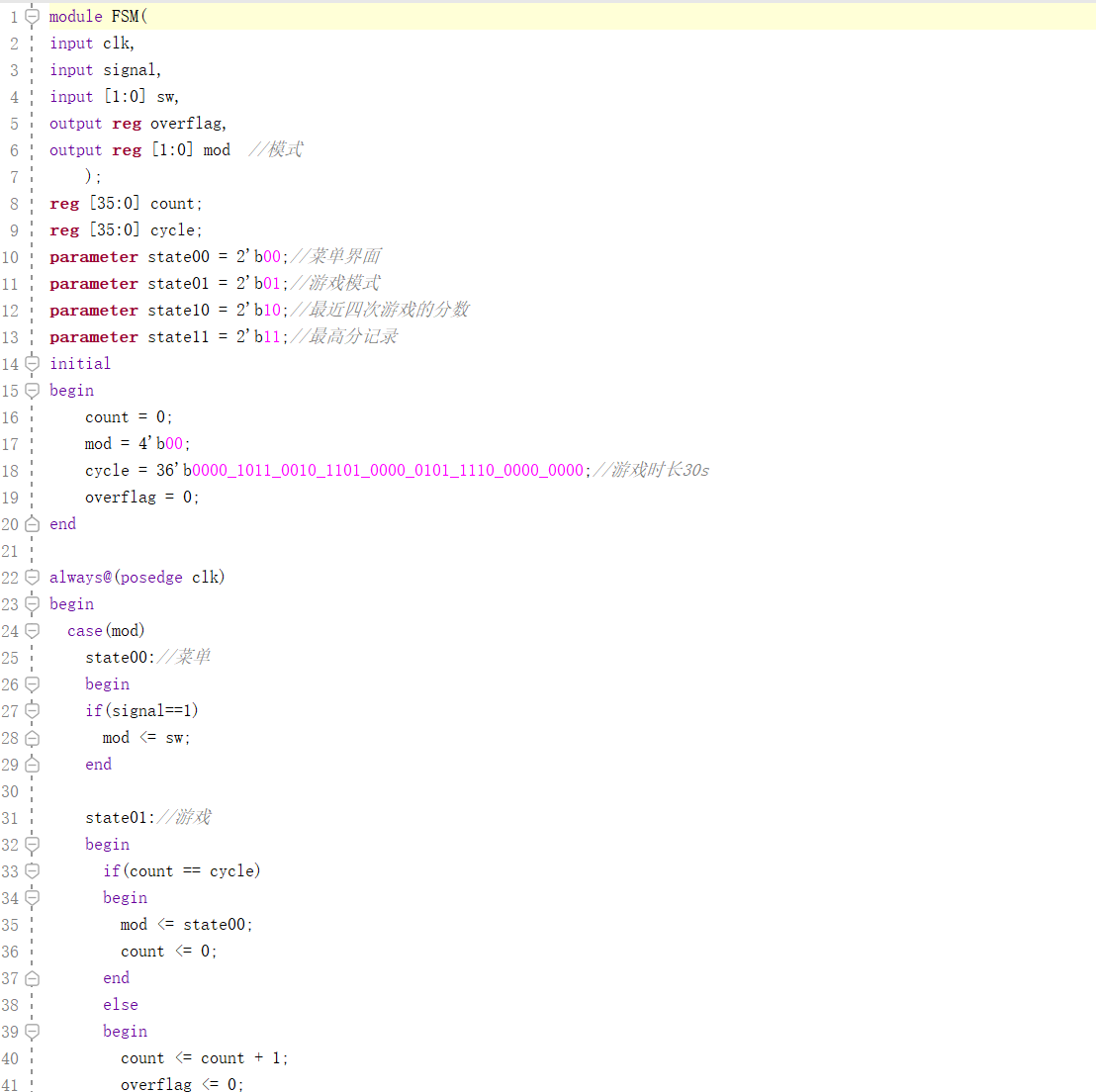
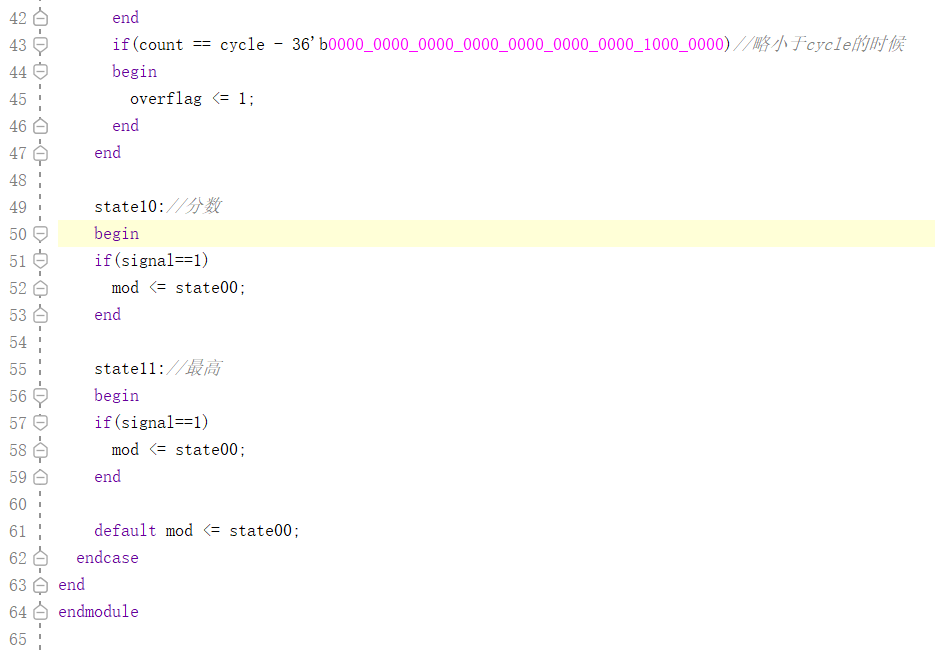
该模块是去毛刺模块，在前几次实验中多次用到，故不多作阐述。

·buttonedg

该模块是取边沿信号模块，同上，故不多作阐述。

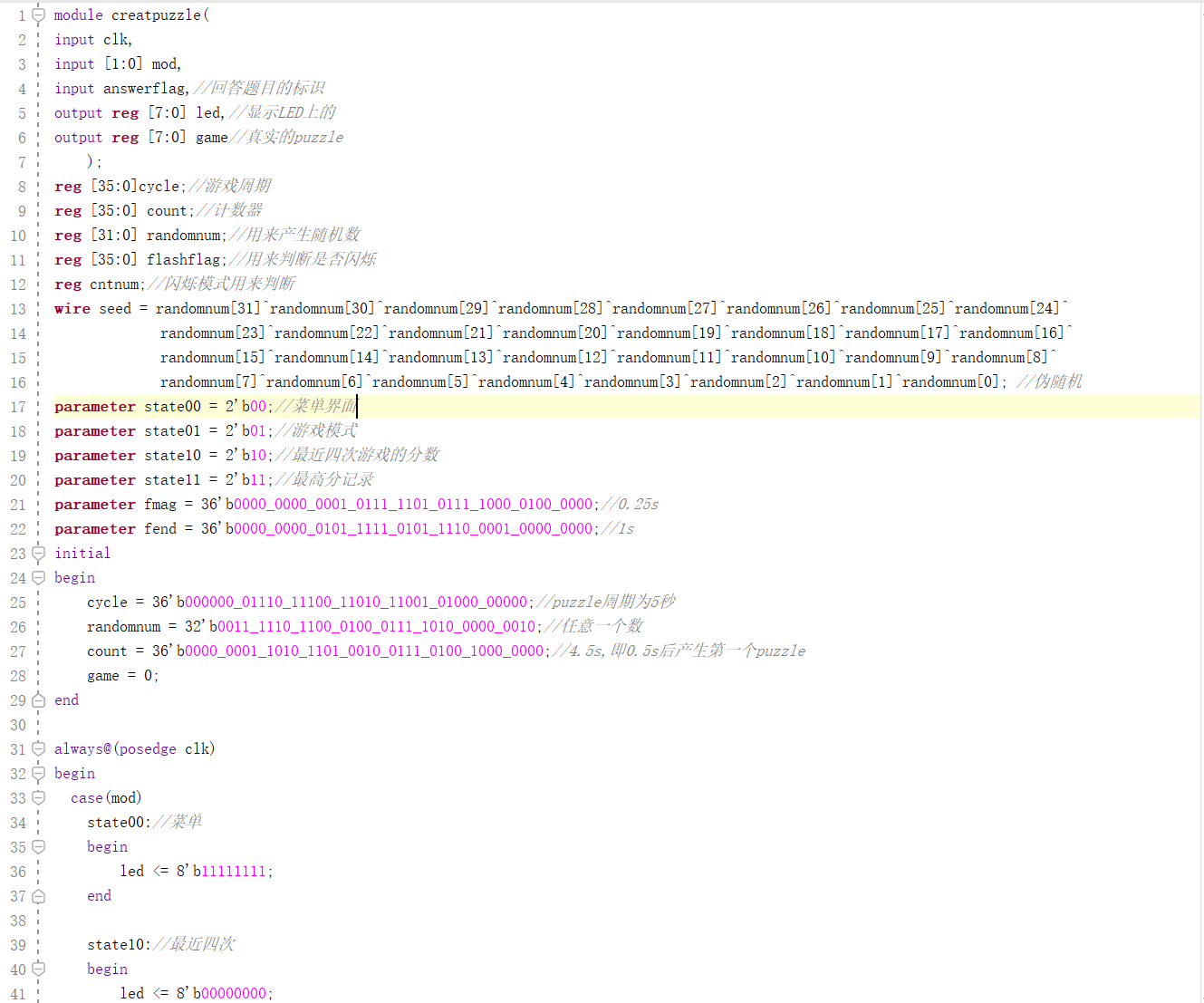
·FSM

对状态机的分析在上文中已经给出，下面给出具体代码。必要注释已在图中：

·creatpuzzle

该模块是生成“地鼠”模块。在该模块中，上文阐述的产生(伪)随机数的方法就要发挥作用了。原理已在上文详细阐述了，下面给出代码细节：

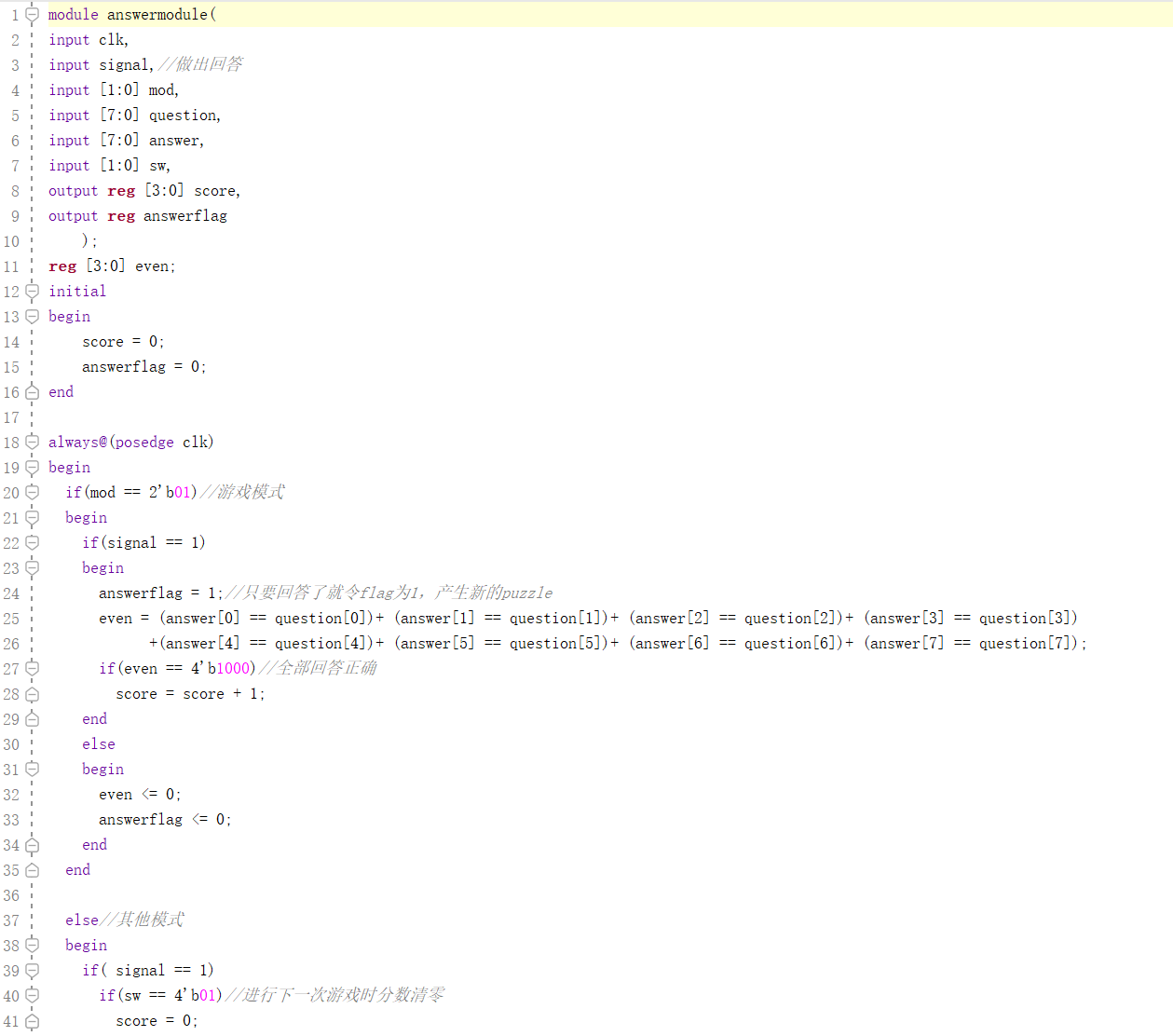


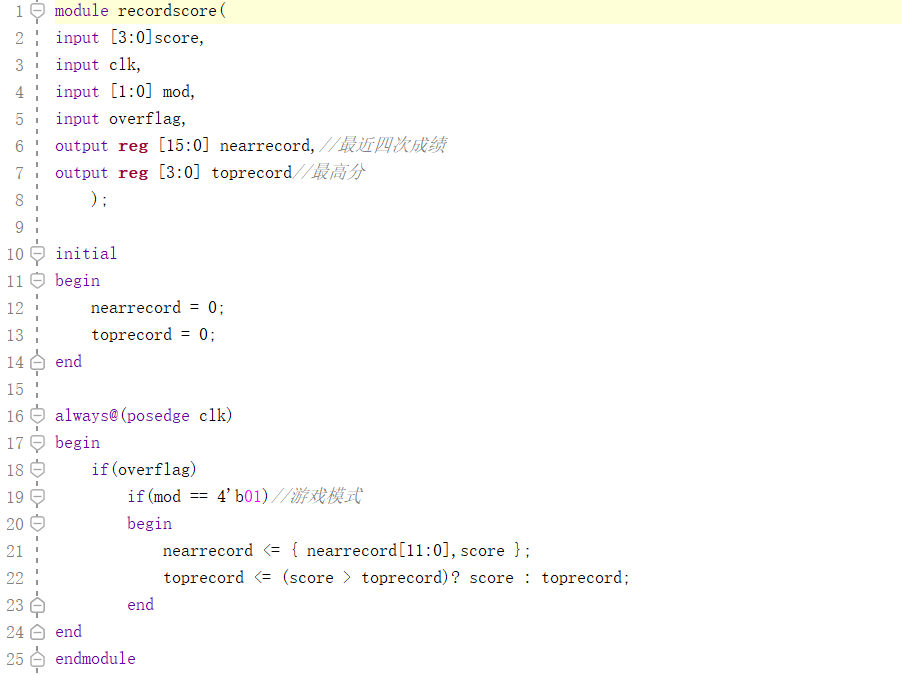


可以看到，上述代码第13行开始，给出了seed的取法，即对每一位进行按位异或，然后在第51行可以看到，seed与原数的[30:0]进行拼接，这就是上文中提到的LFSR引出伪随机数的代码实现。

·answermodule

该模块是“锤子”信号接受模块。上文中已经提到了其实现方式，即用sw开关对应需要敲击的位置，按下button代表着敲击，根据这个思路，就可以编写出代码了。此外，在这模块中，我们还需要设置一个信号，该信号传递的信息是button是否被按下。当button被按下，令该信号有效，传输到creatpuzzle模块中，立刻产生下一个新的随机数。用这样的方法，就可以模拟地鼠不断随机出现的情形了。具体代码实现如下:

·recordscore

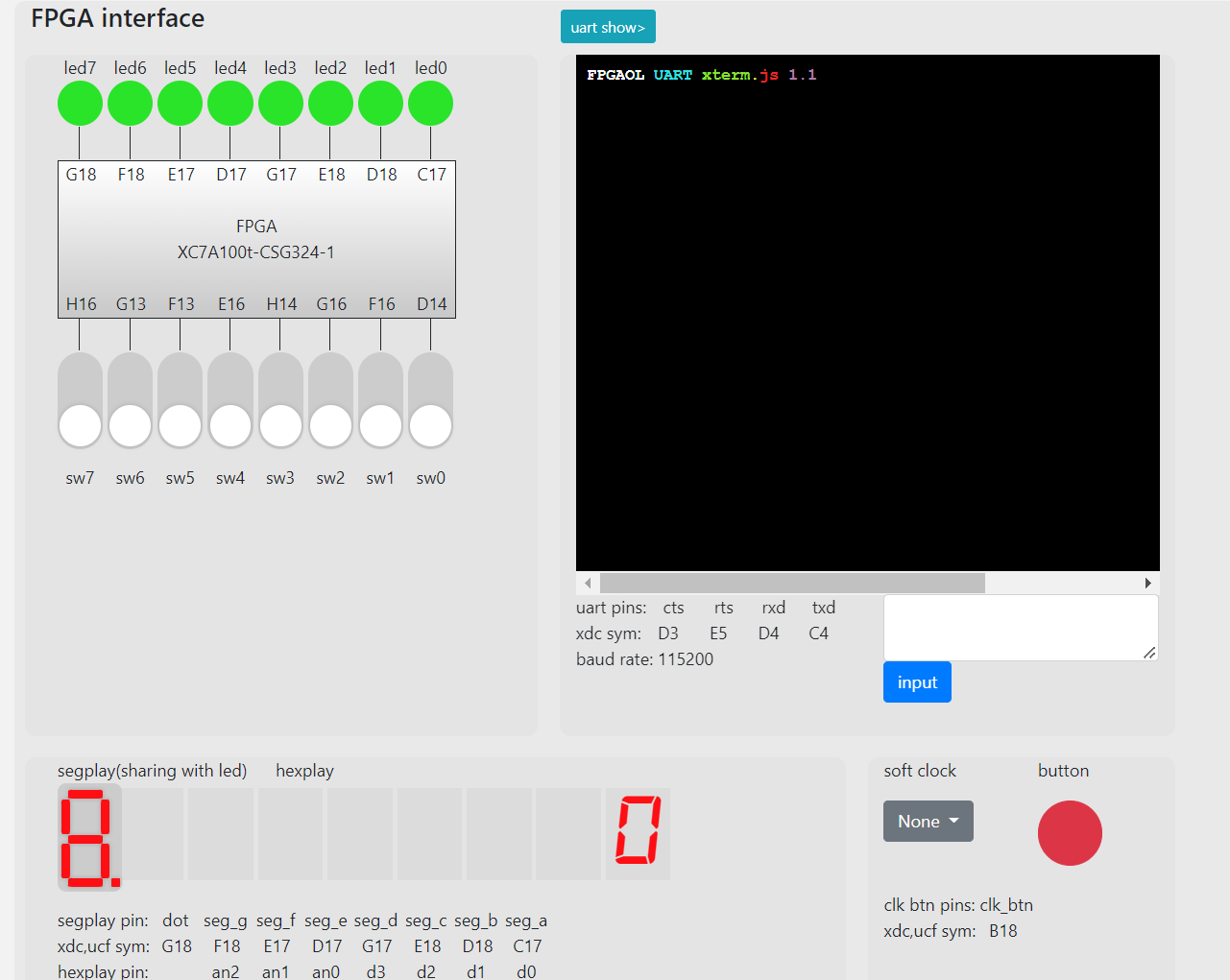
该模块是得分记录模块，得分的规定为：成功在一个有效时间内击打一次所有出现的地鼠，就得一分，否则不得分。根据这个规则，就能很容易地写出该模块的代码了：

·dcdtest

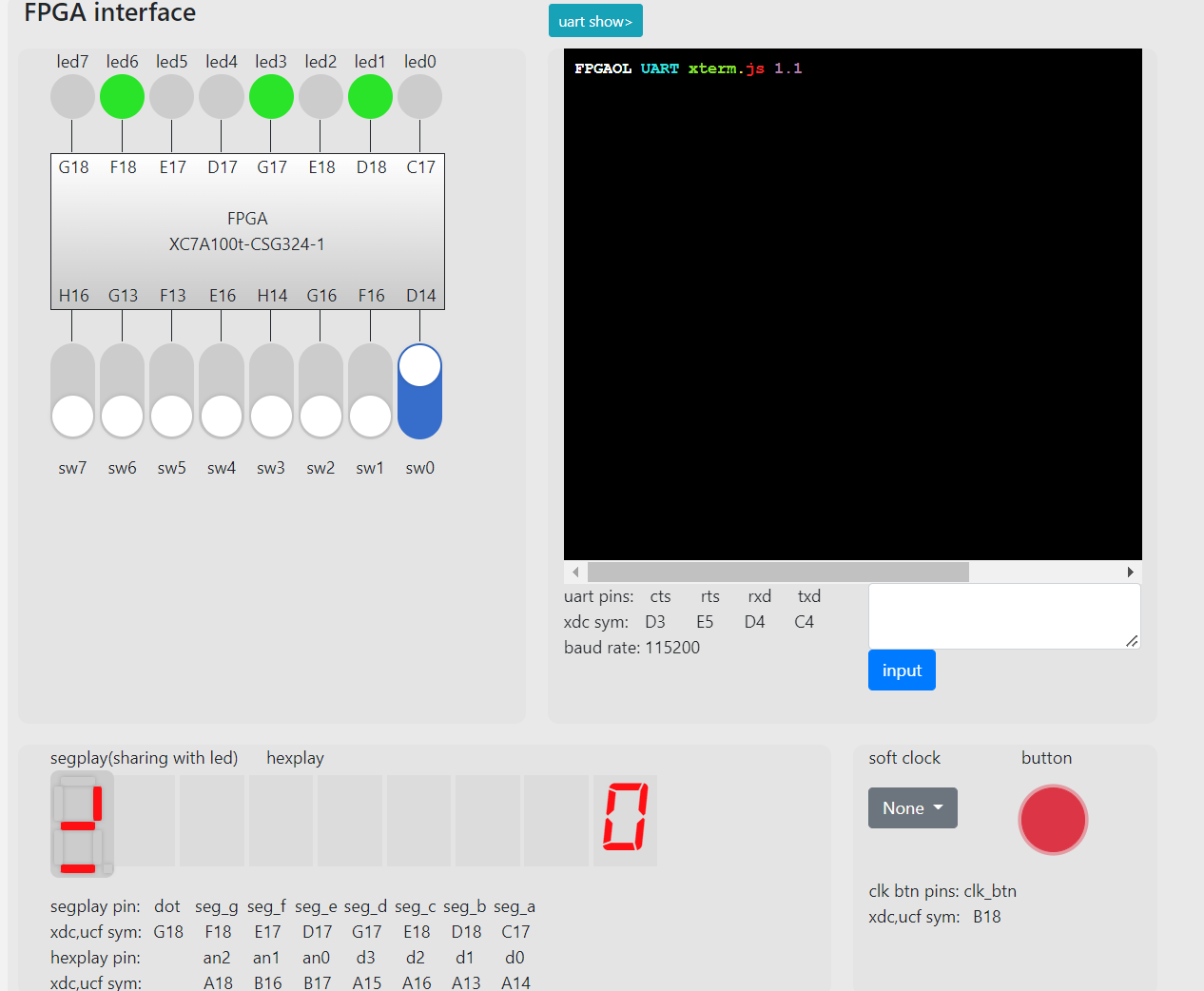
该模块是数码管显示模块，前面几次实验多次使用用时分复用的方法，就很容易得到该模块代码。由于代码很长，而原理其实很简单，故不做展示了。

**五、效果展示**

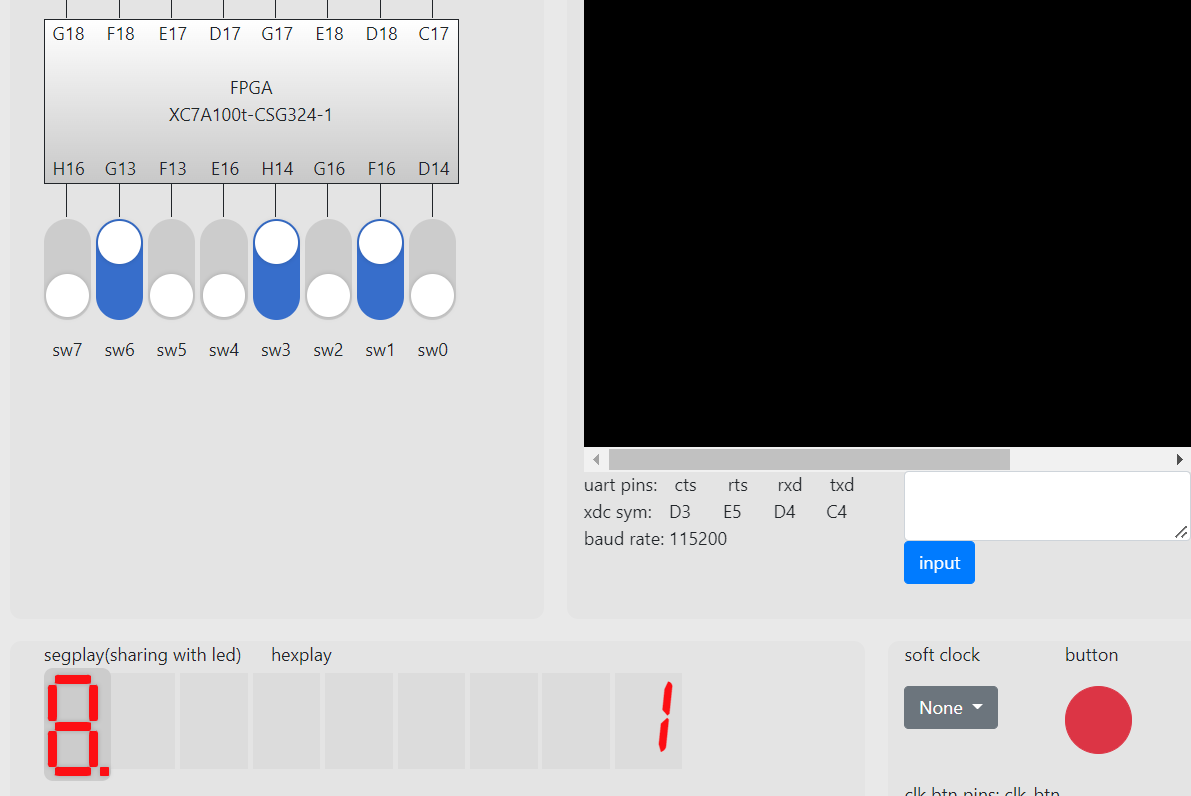
将bit流文件烧写到FPGAOL平台上，就可以开始游戏了。开始时，界面显示如下，所有LED灯亮起，这就是初始菜单界面：



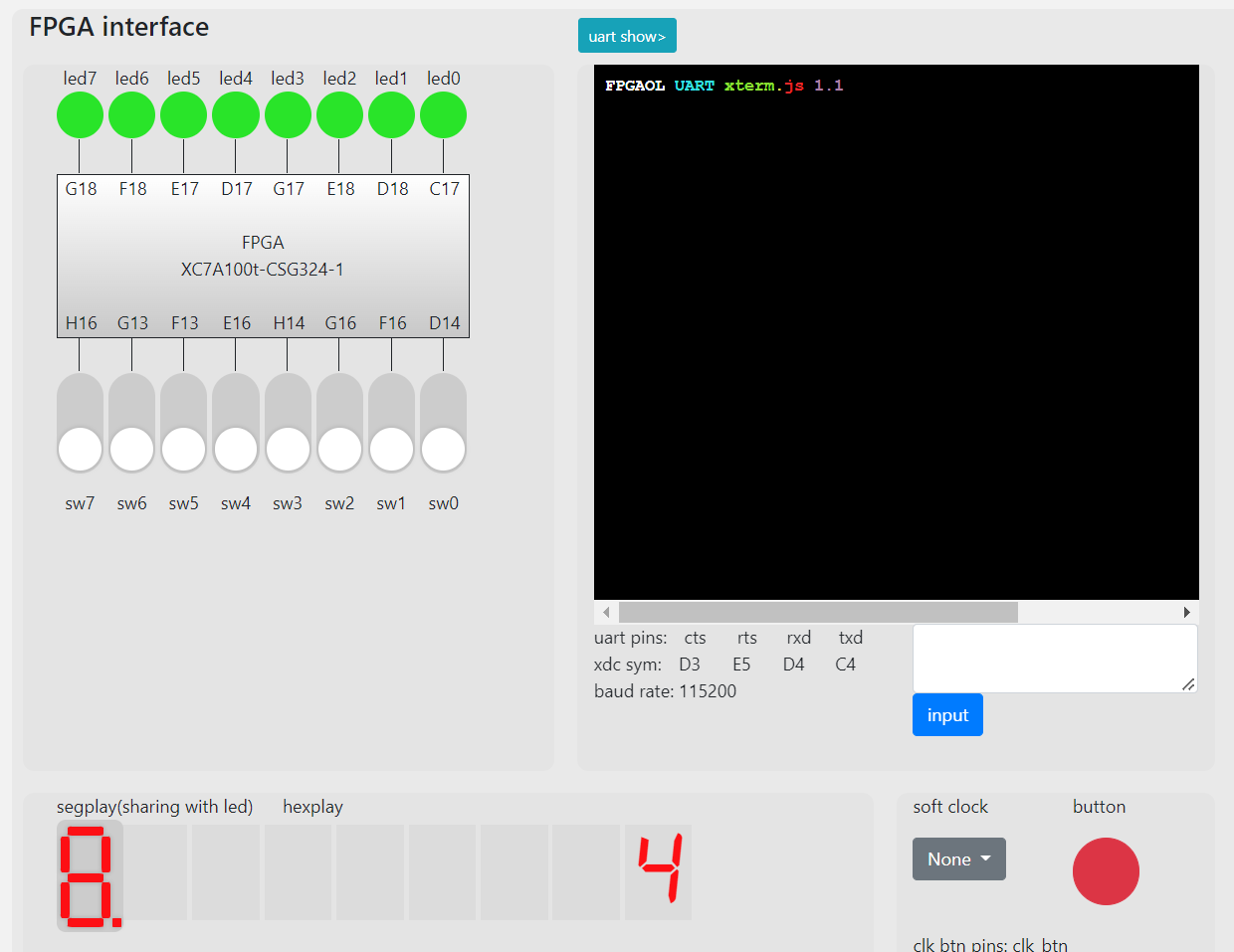
下面，选择01状态，按下button，就可以进入游戏了：



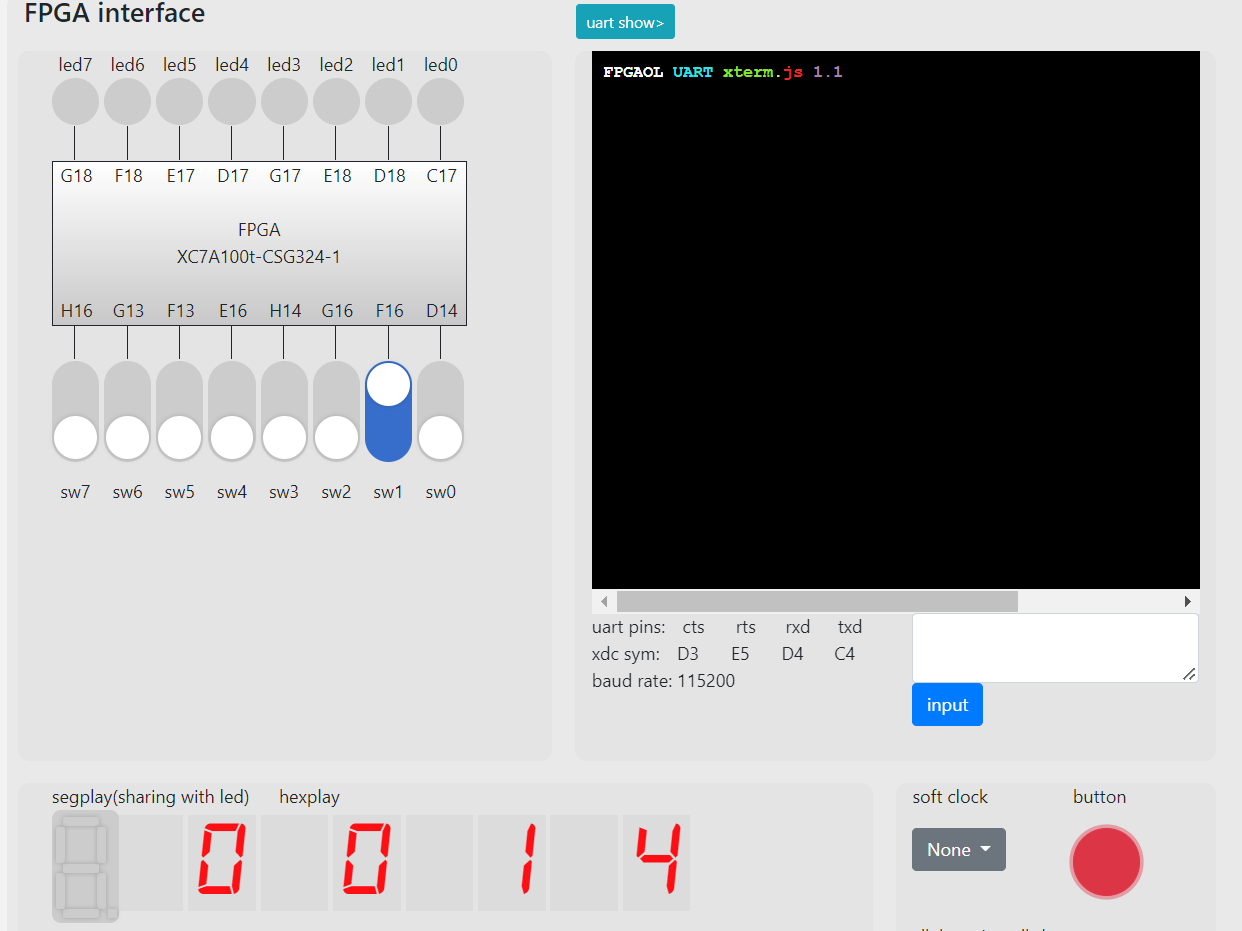
可以看到，led6/3/1这三个灯在闪烁，这代表这三处有地鼠出现，于是，我们将对应的开关拨上去，按下button，就获得了一分，分数在下方右边第一个数码管上显示。



30秒后，游戏结束，本轮我们总共获得了4分：



下面选择状态10，可以查看最近四次游戏的分数，测试时进行了两轮游戏，分别得分为1、4：



接下来选择状态11，按下button，就可以查看历史最高分了：

