实验名称：实验六 寄存器

姓名：张涵之

学号：191220154

班级：周一5-6

邮箱：[191220154@smail.nju.edu.cn](mailto:191220154@smail.nju.edu.cn)

实验时间：2020/10/12

6.3.1 算术移位和逻辑移位寄存器

请根据表6-3，用Verilog HDL语言设计移位寄存器，进行仿真查看移位寄存器的功能。对于移位寄存器的实现细节，请自行复习数电教科书8.5节内容，参考8.5.9节内容实现。

实验目的：设计一个带功能选择端的移位寄存器。

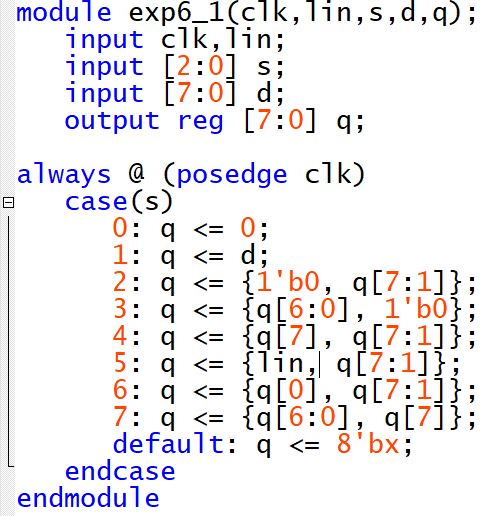
实验原理：利用case语句实现功能选择，用Verilog HDL语言很描述移位寄存器，如：Q <= {Q[0],Q[7：1]}; 表示循环右移，Q <= {Q[7],Q[7：1]}; 表示算术右移。

其中左端串行输入1位数值，并行输出8位数值是指每个时钟到来时右移一位，移入的最左位由外部开关决定，输出同其他情况一样为同时输出8位。



实验环境/器材：实验箱一个，笔记本电脑一台。

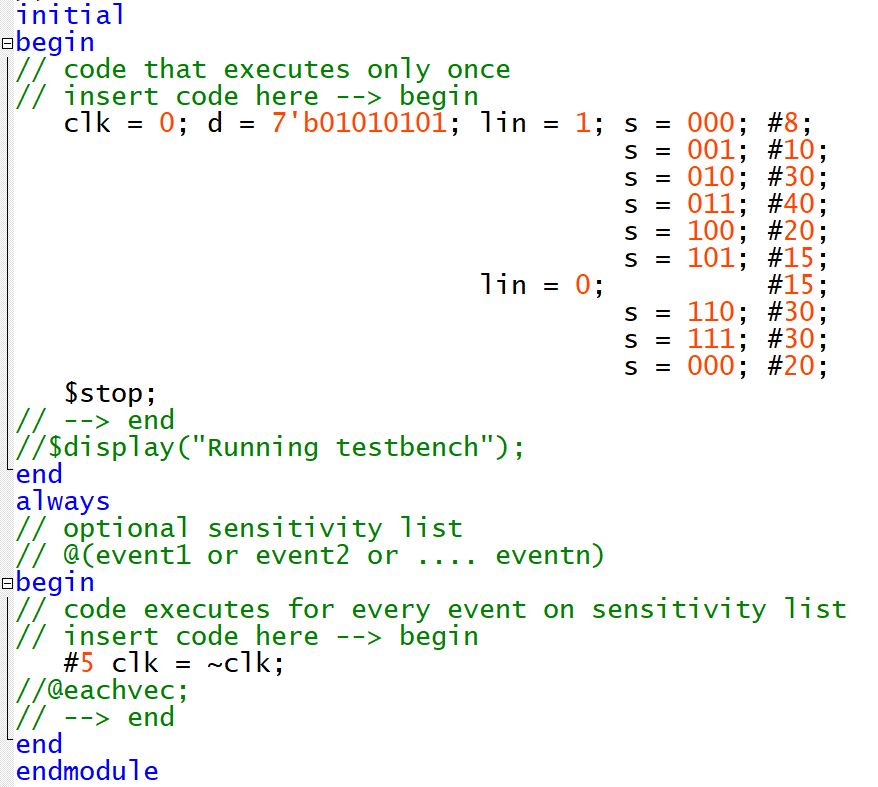
程序代码或流程图：



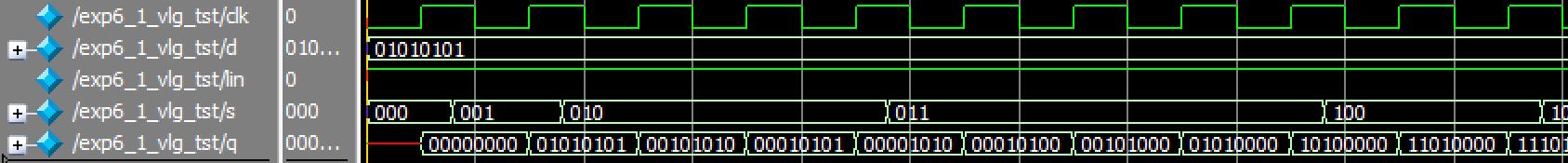
实验步骤/过程：用case语句进行功能选择，采用赋值符号<=进行非阻塞赋值。

用SW0~SW7作数据端，SW9为左端串行输入位，LEDR0~LEDR7为输出显示，button KEY0模拟时钟信号，每当时钟信号为上升沿时执行赋值语句，写测试代码进行仿真模拟。

测试方法：



实验结果：





通过观察对比，移位寄存器的输出符合预期。经接入实验箱检验，显示也符合预期。

实验中遇到的问题及解决办法：实验非常顺利，没有碰到任何问题。

实验得到的启示：无。

意见和建议：无。

6.3.2 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。

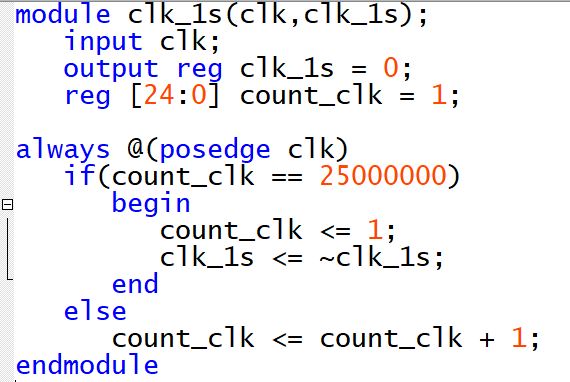
请将8位二进制数以十六进制显示在数码管上，在DE10-Standard开发板上观察生成的随机数序列。系统需要能够自启动。

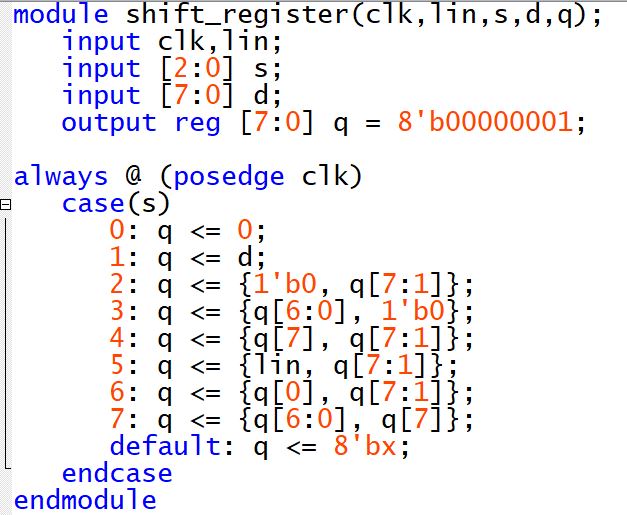
实验目的：实现一个简单的随机数发生器，以十六进制显示在数码管上。

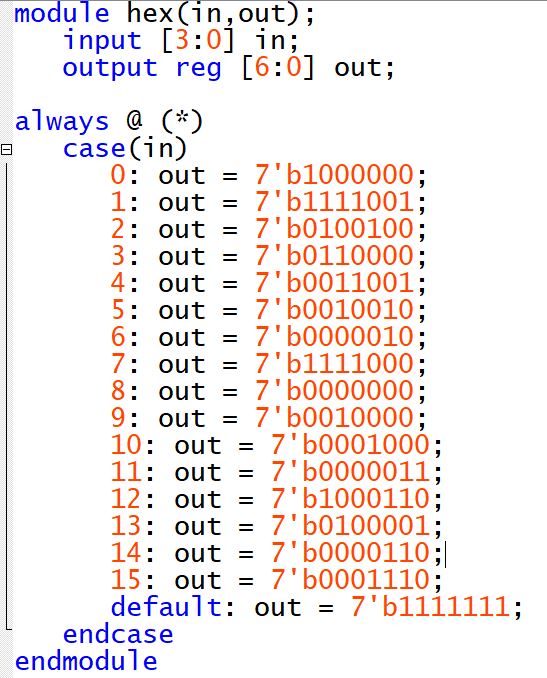
实验原理：参考教科书第534页LFSR反馈方程，LSFR计数器有2^n-1种有效状态（非零状态），查表可知n=8时的反馈方程为X8 = X4⊕X3⊕X2⊕X0，通过6.3.1中移位寄存器的左端串行输入功能生成8位二进制数，每四位转化为十六进制显示在数码管上。

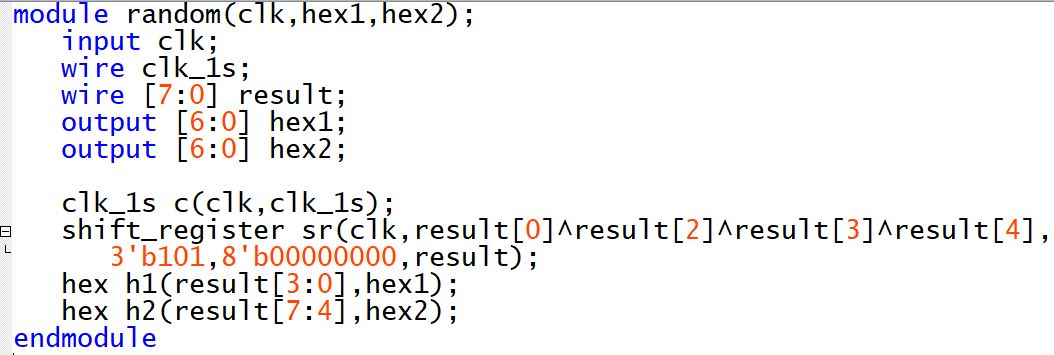
实验环境/器材：实验箱一个，笔记本电脑一台。

程序代码或流程图：









\*此处sr参数直接代入clk为RTL测试用，便于在短时间内（拖动较短的进度条内）观察到更多组输出，实际测试（使用实验箱接入时钟信号）时改为clk\_1s。

clk\_1s生成时钟信号 -> 时钟信号接入shift\_register生成序列

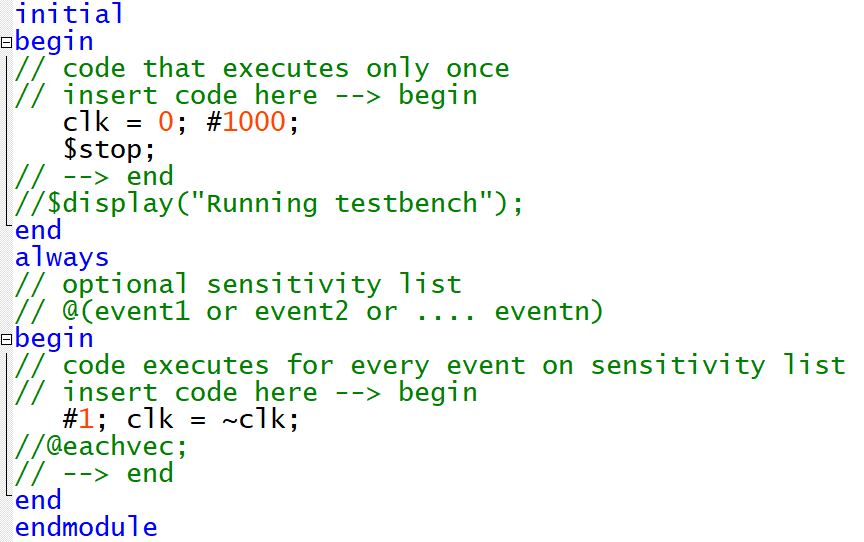
-> hex将生成序列转化为七段数码管的输入 -> 接入七段数码管显示。

实验步骤/过程：

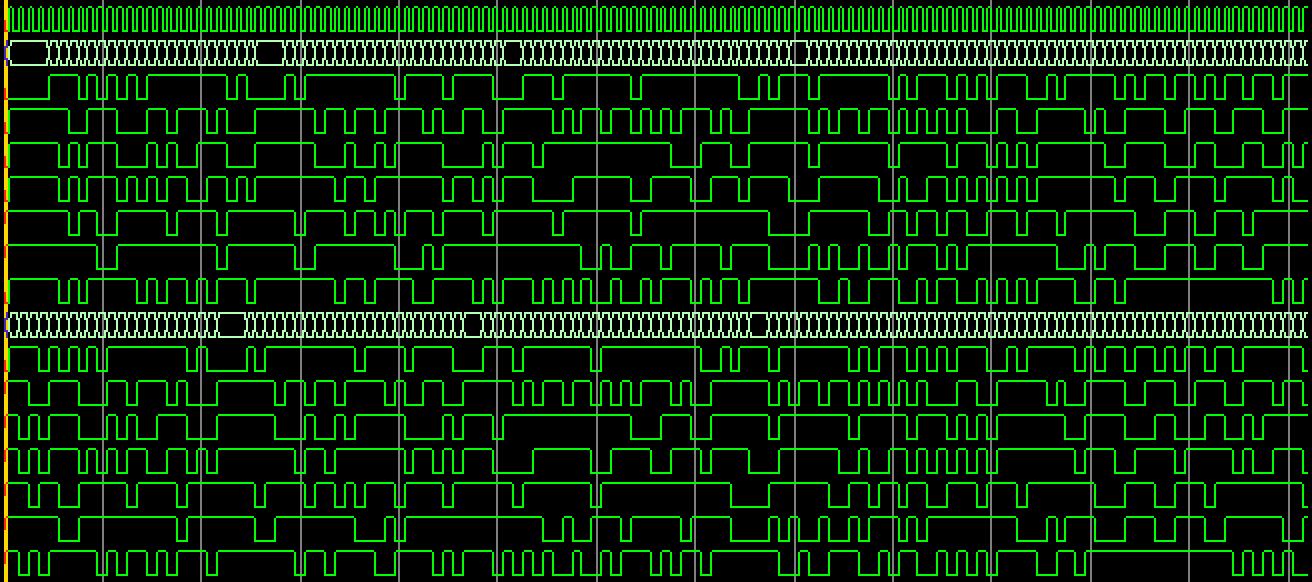
单独写出四个模块，把四个模块合并，写测试文件对顶层实体进行仿真，发现没有出现预期的效果，根据对自己的认识，推测是1秒钟生成器的问题。

对模块单独进行仿真，发现问题并修改，重新测试，在实验箱上进行操作。

测试方法：



实验结果：



通过观察，生成随机数行为符合预期。经接入实验箱检验，显示也符合预期。

实验中遇到的问题及解决办法：

1. 没有分别测试每个模块，直接测试了顶层模块，发现输出数据从头至尾保持不变。

解决方法：将最不熟悉（不确定）的1秒时钟生成器（做实验6在实验5之前）设为顶层实体，循环终止条件25000000改为2（便于测试），写测试代码进行仿真实验，观察到clk\_1s的输出始终为不确定值（红色）。原来是没有初始化clk\_1s和count\_clk导致程序完全没有在计数，clk\_1s = ~clk\_1s代码始终无效。

2. 直接将1秒钟生成器的输出用作时钟接到顶层模块，会导致仿真时需要很长时间数据才会发生一次变化，需要大量拖动进度条，给仿真结果的观察造成困难。

解决方法：分开测试1秒钟生成器和顶层模块，便于写测试代码和观察仿真结果，实际想要实现的1秒生成1次随机数操作效果直接用实验箱进行测试。

实验得到的启示：

1. 不要把所有模块全部写完、链接好再开始测试，出现问题时会造成排查调试非常困难，最好每写完一个模块就对应地测试一次。

2. 可以适当进行分模块测试，最后直接在实验箱上进行综合。

意见和建议：无。

思考题：

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？

为了生成更加复杂的伪随机数序列，可以考虑对单纯左端移入产生的序列进行一些打乱重排操作，如生成的序列为8位，取序列高四位设为a，低四位设为b，则a和b各自表示一个大小在0和15之间的数，取序列的第a%8和第b%8位交换，得到更复杂的伪随机数。