《数字电路与数字系统实验》实验报告

**实验6: 寄存器**

**姓名:** 尹浚宇

**学号:** 161130118

**班级:** 2018-2019第一学期数字电路与数字系统实验2班

**邮箱:** [908664035@qq.com](mailto:908664035@qq.com)

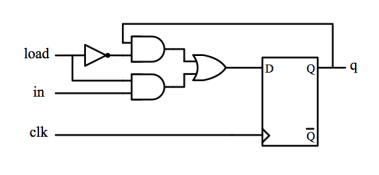
**实验时间:** 2018-10-23

1. **实验目的**
2. 复习寄存器的原理.
3. 掌握几种常用寄存器的设计方法.
4. 自行设计一个多功能的移位寄存器.
5. 自行利用移位寄存器实现一个随机数发生器.
6. **实验原理**

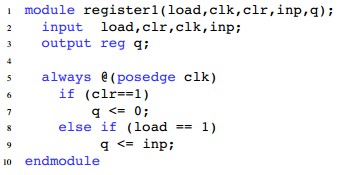
寄存器也是最常⽤的时序逻辑电路, 是⼀种存储电路. 寄存器的存储电路是由锁存器或触发器构成的, 因为⼀个锁存器或触发器能存储1位二进制数, 所以由N个锁存器或触发器可以构成N位寄存器.

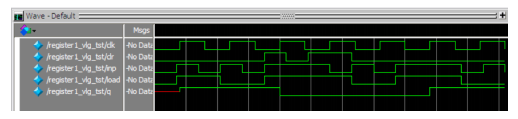
1. 寄存器

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, 如下图所示. 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.

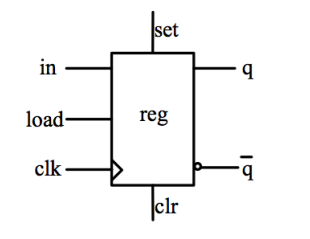


下图是一个带有清零端和输入端的1位寄存器的verilog代码和仿真波形图.



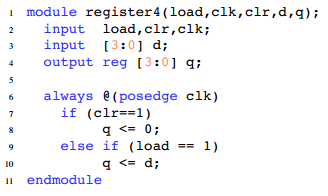


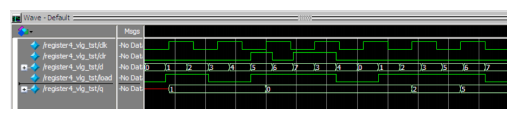
有一些寄存器还带有置位端, 下图就是一个同时带有清0端、输入端和置位端的寄存器的逻辑示意图.

，

将2个或者2个以上的1位寄存器组合在⼀起, 这些寄存器共⽤⼀个时钟信号, 这就构成了多位寄存器, 寄存器常被⽤在计算机中存储数据, 如指令存器、数据寄存器等.

下图是一个带有清零端和输入端的4位寄存器的verilog代码和仿真波形图.

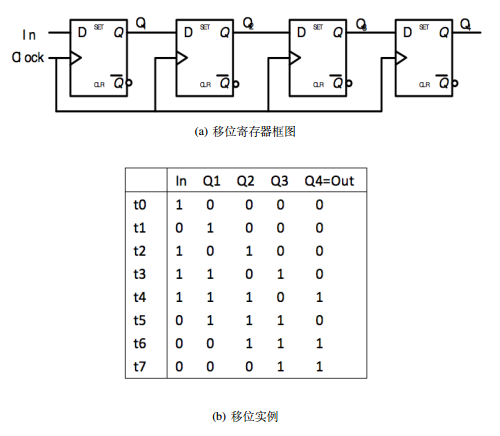




1. 移位寄存器

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(a)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. 下图(b)是⼀个此移位寄存器的序列传递实例.

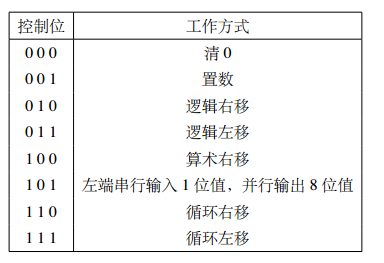


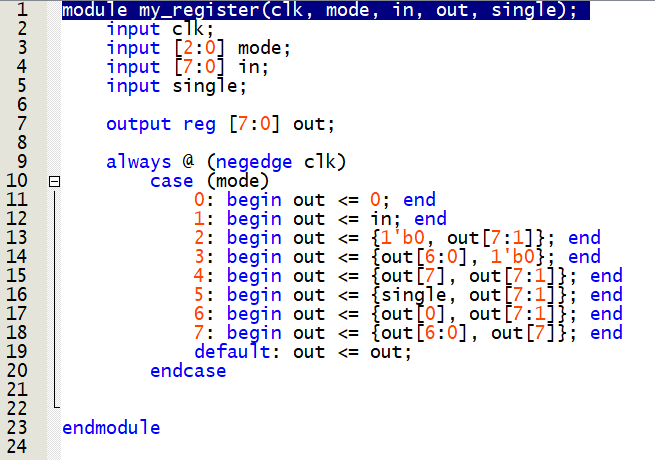
1. **实验环境/器材**

系统环境是window10, 硬件环境是DE10-Standard开发板, 软件环境是Quartus.

1. **程序代码**
2. 多功能移位寄存器

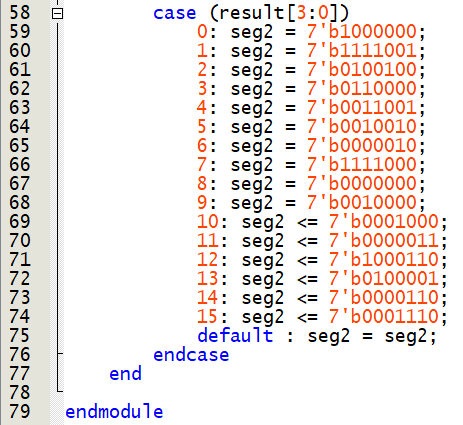
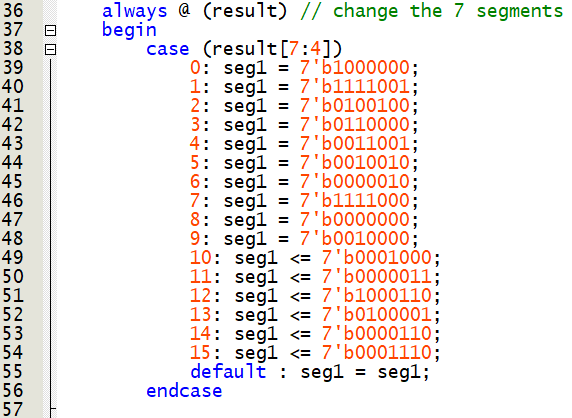
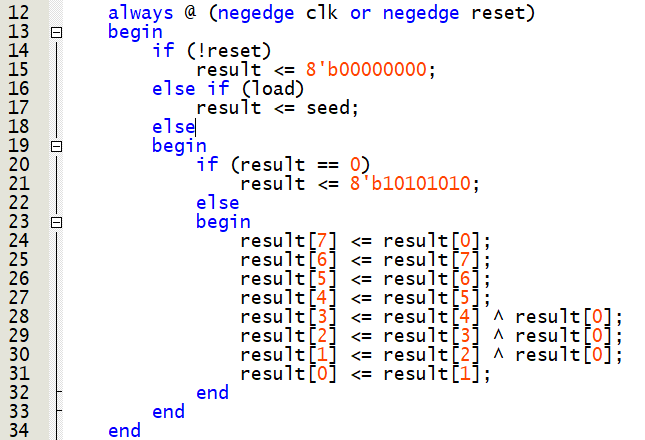
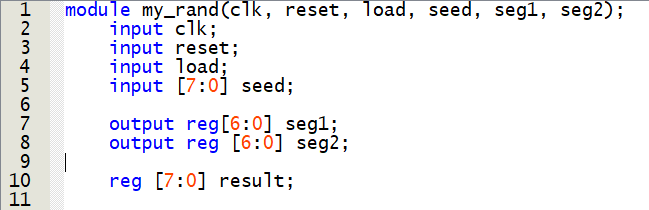
clk是时钟端, mode是工作模式选择端, in和out分别是输入输出, single是左端串行输入模式下的输入值. 时钟下降沿有效.工作方式如下图:





1. 随机数发生器

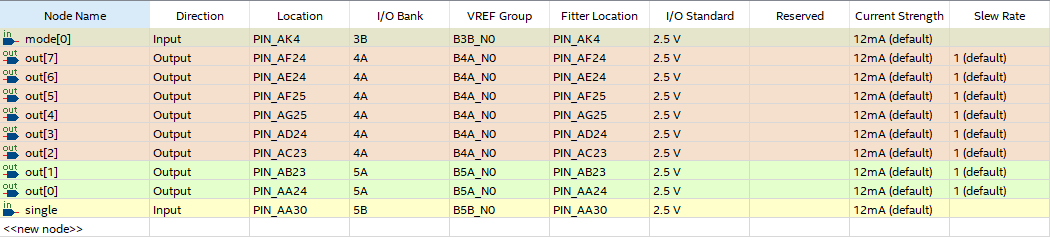
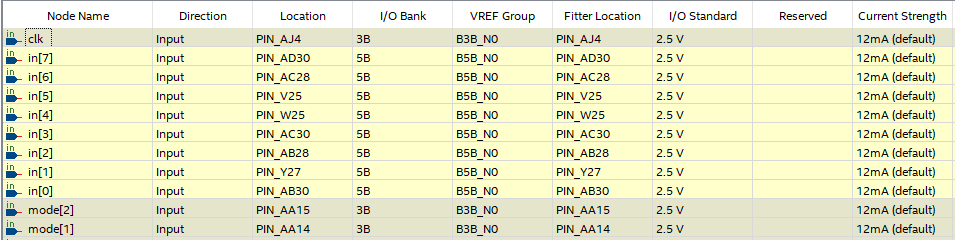
clk是时钟端, reset是清零端, seed是随机种子, load为1时可以读入给定的随机种子, seg1和seg2分别控制两个数码管. 时钟端和清零端都为下降沿有效.



1. **实验步骤**

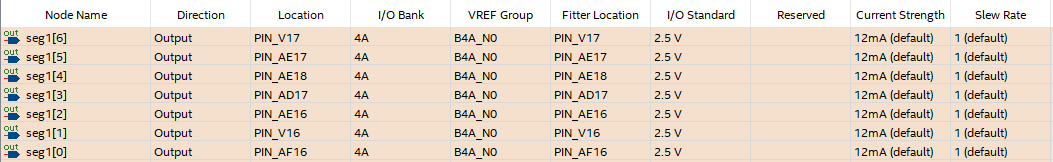
首先分别设计了一个多功能的移位寄存器和一个随机器发生器 (具体代码见四), 通过编译后又分别设计了仿真模拟的代码(见六), 仿真模拟通过后分配了引脚(如下图), 最后讲生成的二进制烧写文件导入开发板进行硬件验证, 并通过了助教的查验.

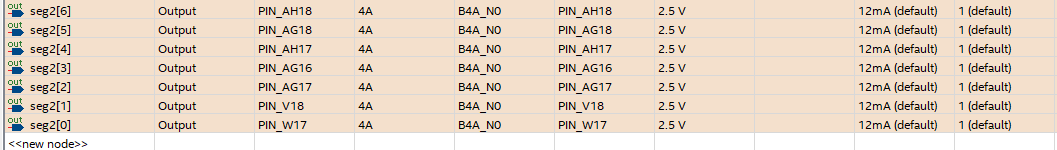
1. 多功能移位寄存器



1. 随机数发生器



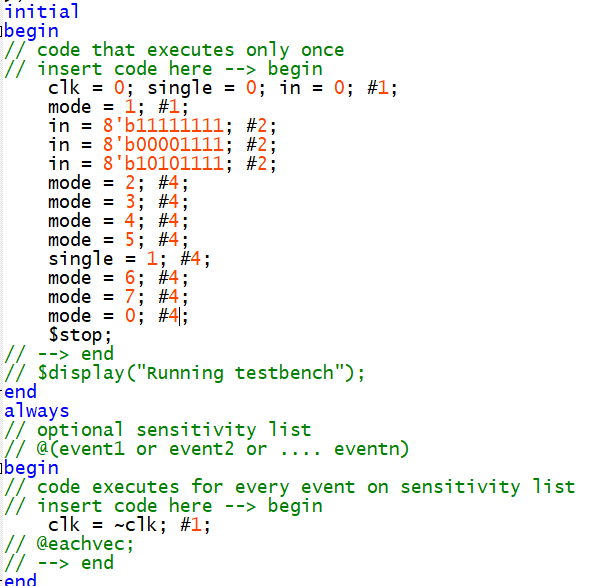




1. **测试方法**

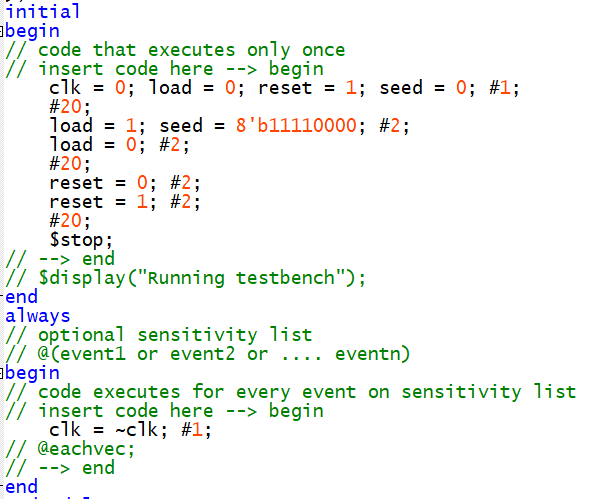
采用仿真验证和硬件验证的测试方法, 首先针对两个不同的器件分别编写了如下的测试代码(见下图)进行仿真验证(仿真波形图见七), 而后进行了硬件验证(见七的图片和压缩包的视频).

1. 多功能移位寄存器



从功能一开始分别测试了每种功能, 最后测试了清零功能.

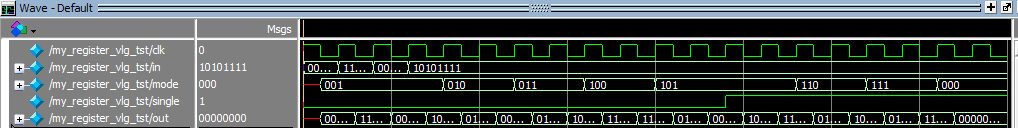
1. 随机数发生器



指定初值后检验能否自启动, 然后检测置数功能是否有效, 然后检验清零功能是否有效, 然后检测清零后能否继续运行, 最后停止.

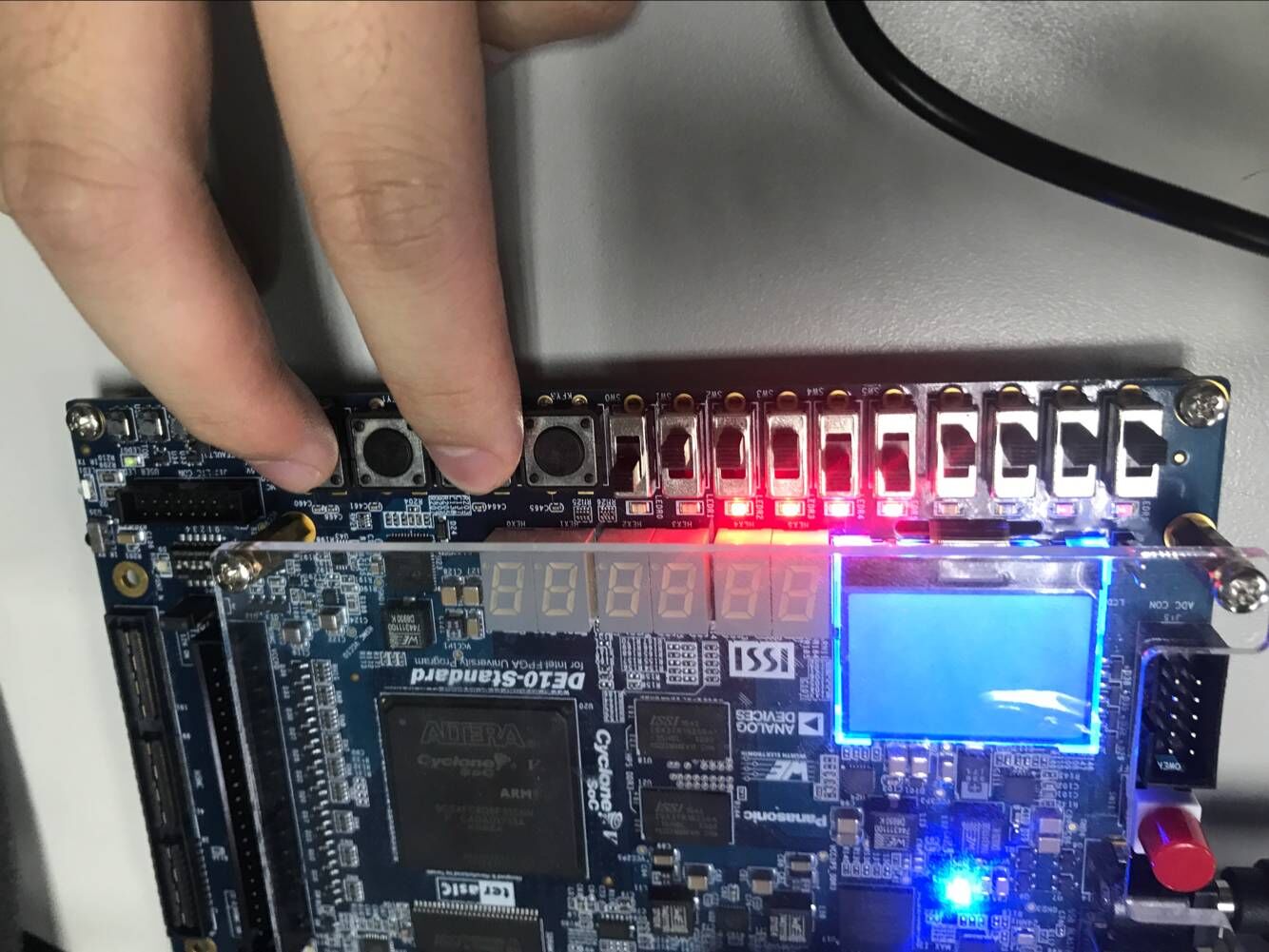
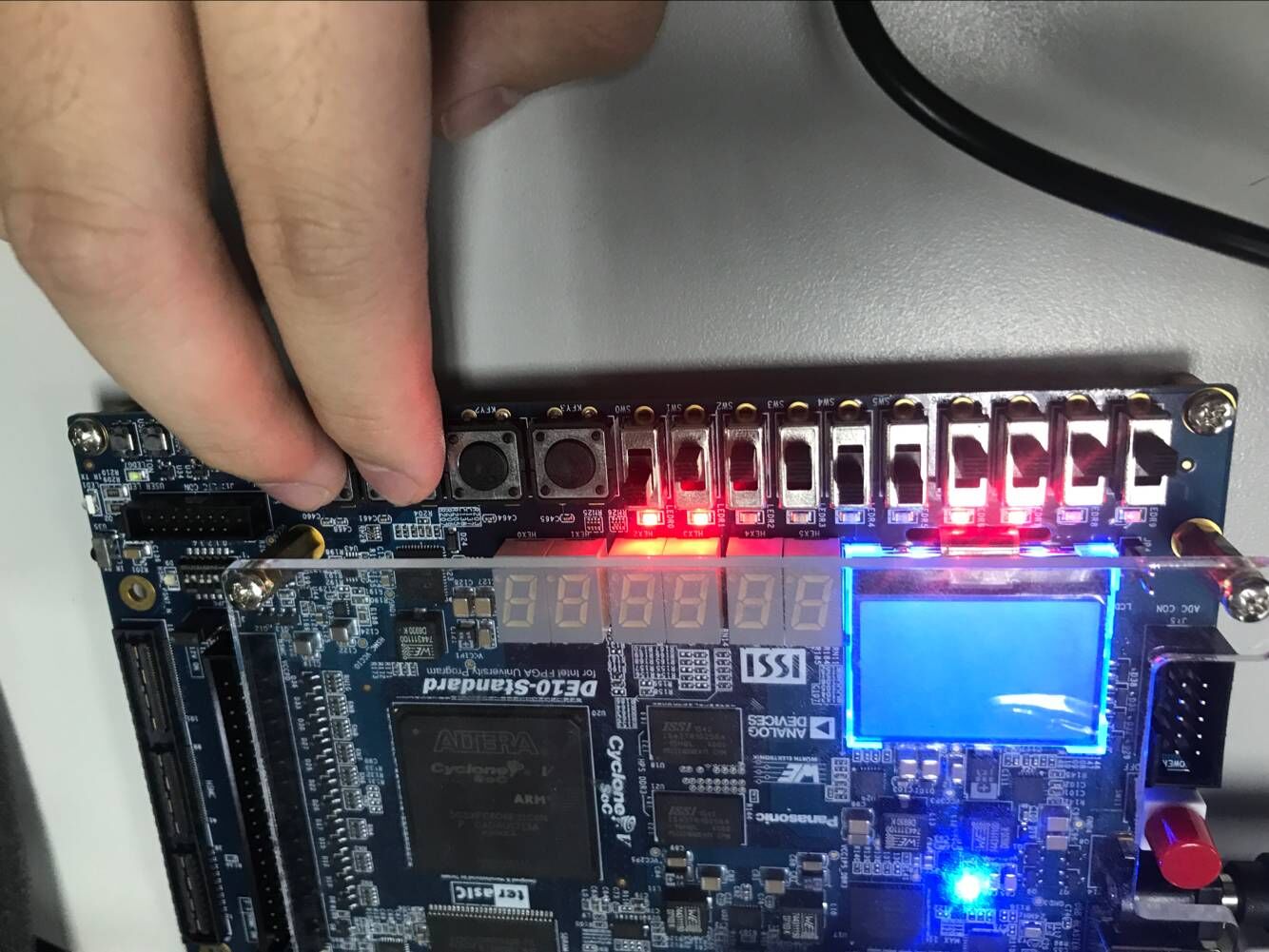
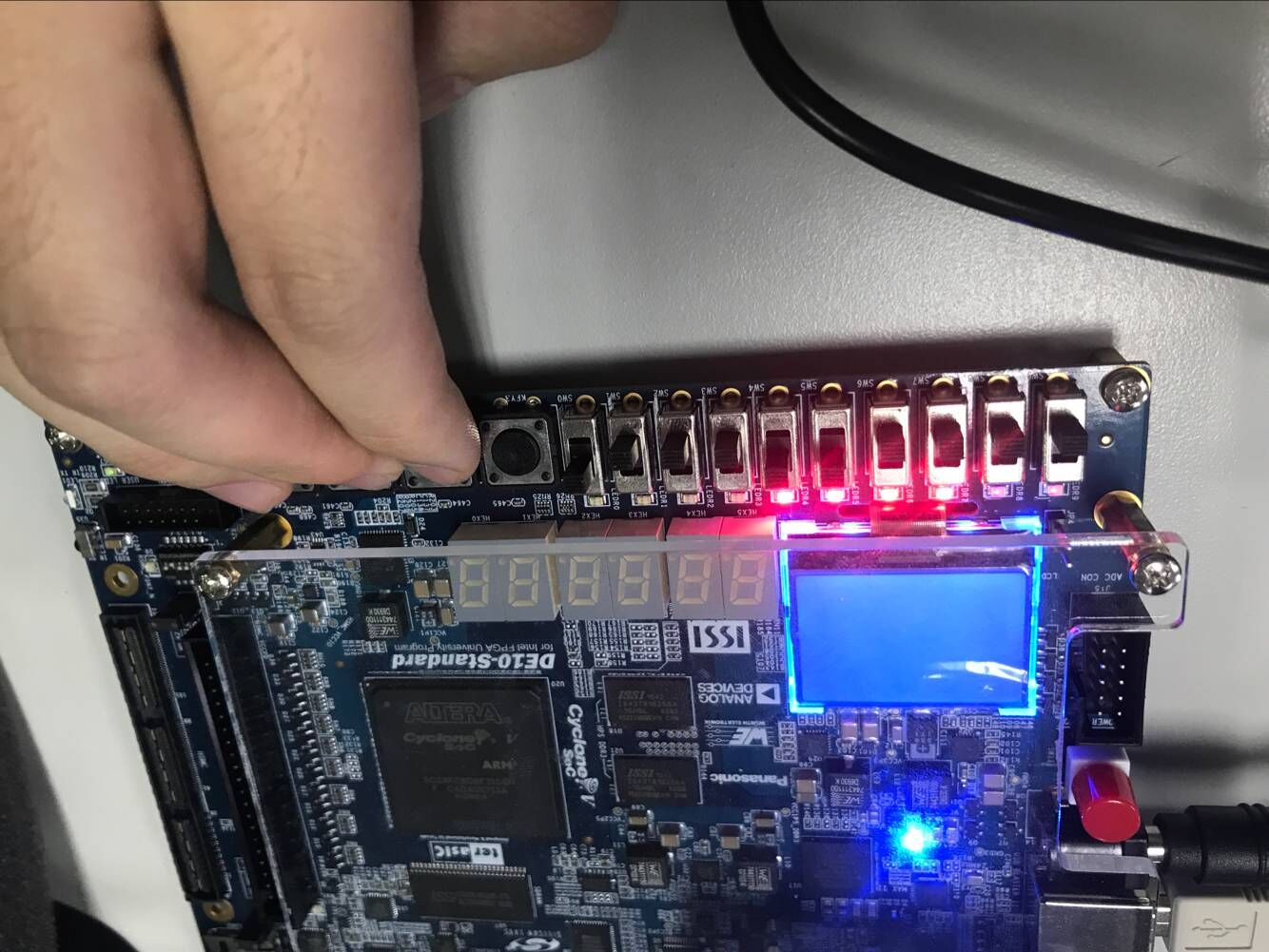
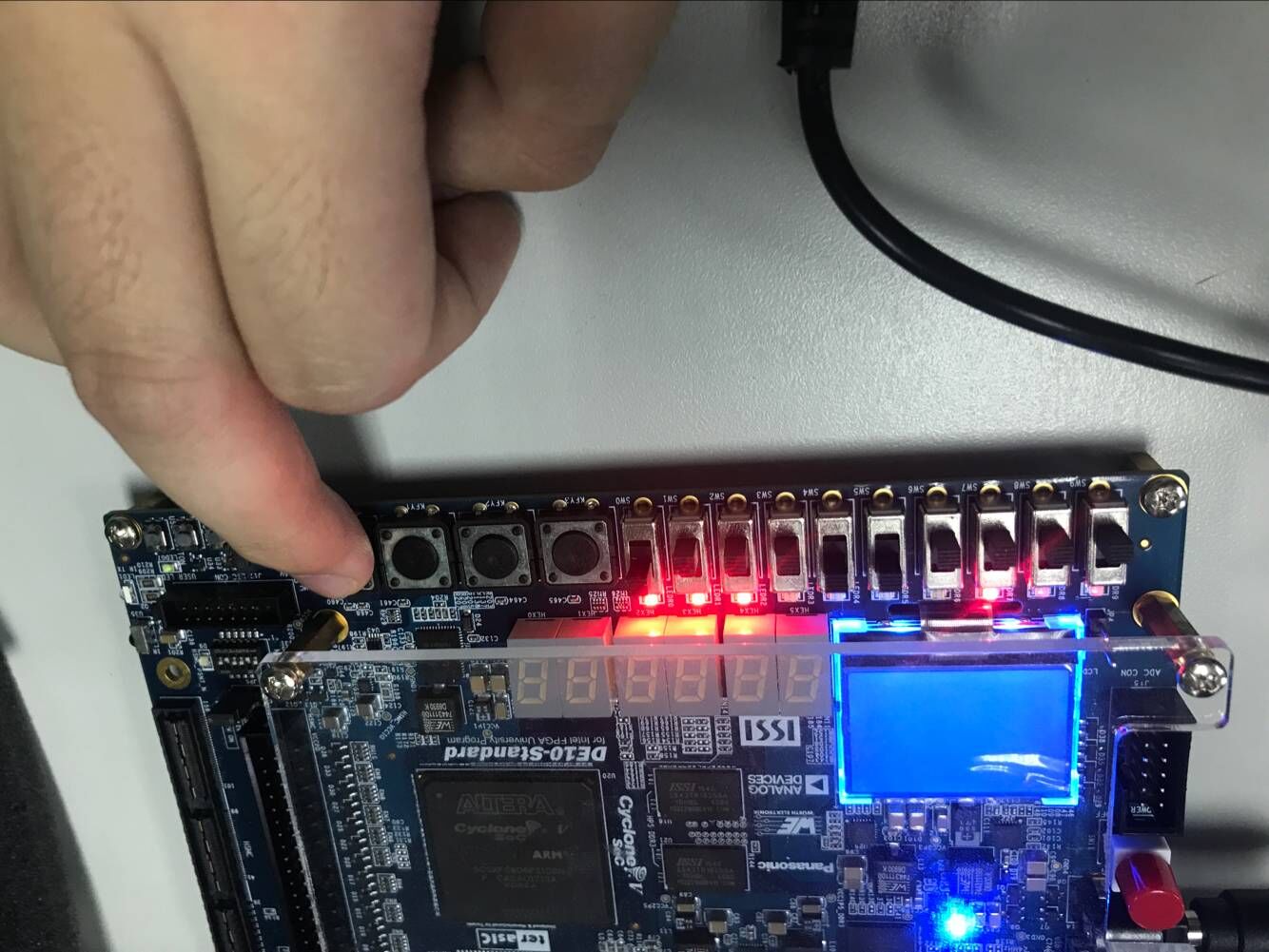
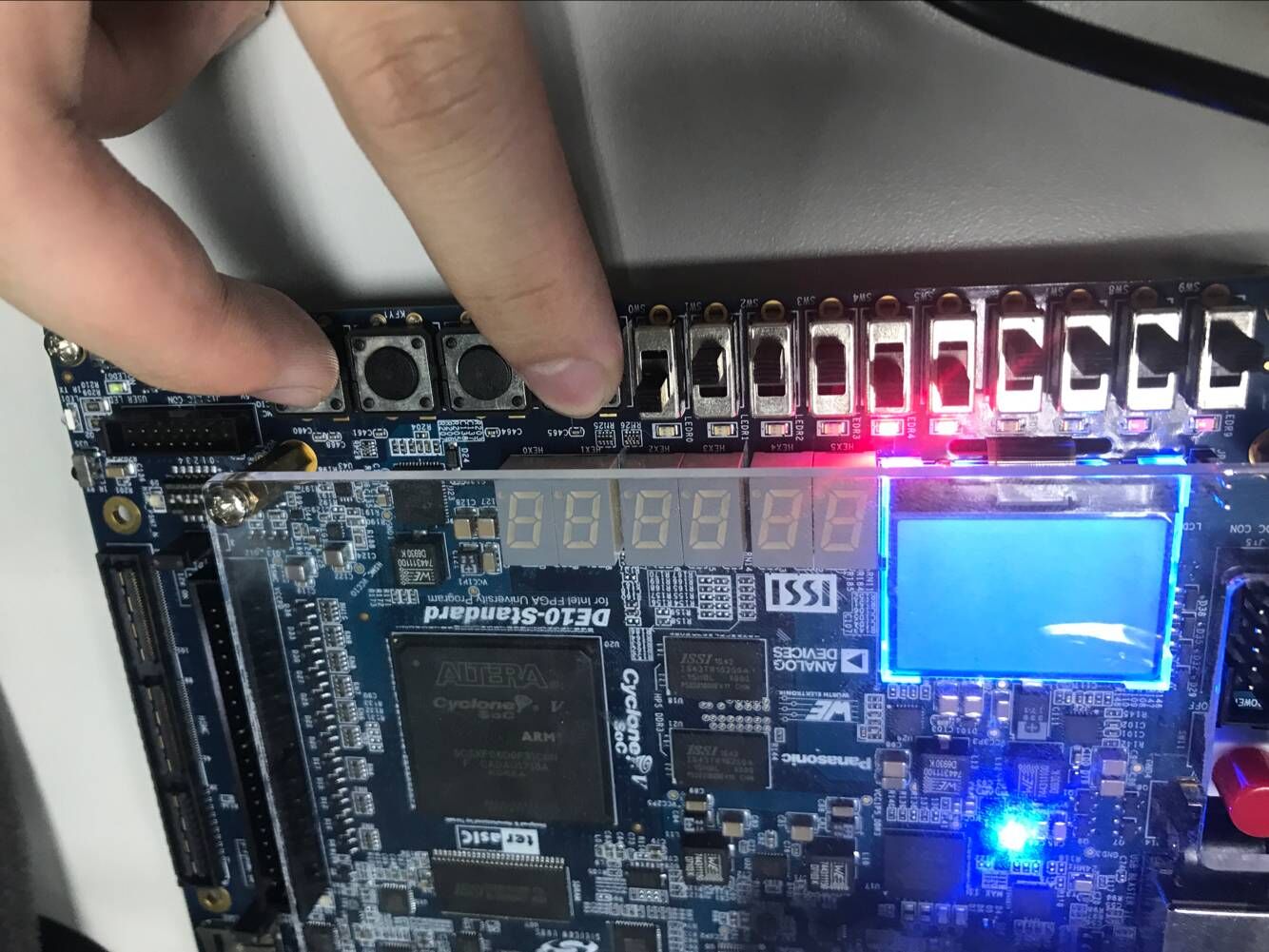
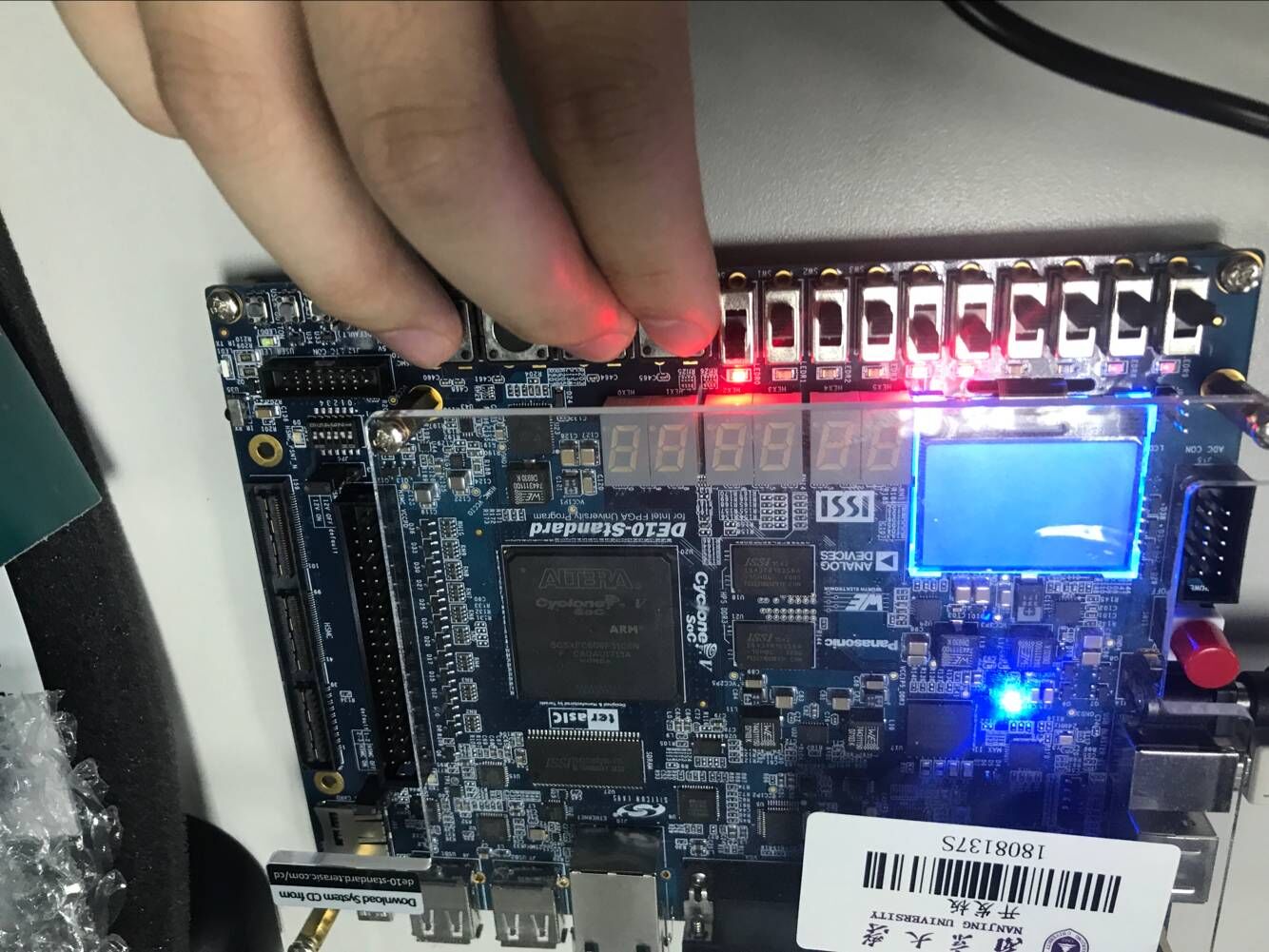
1. **实验结果**
2. 多功能移位寄存器

仿真波形图:



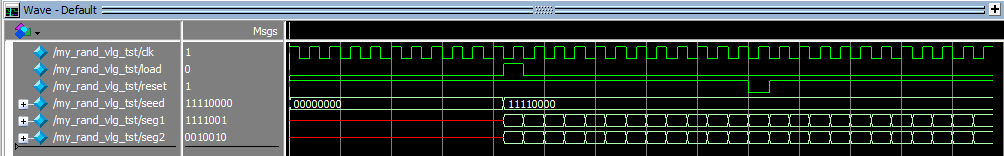
硬件验证:

由于功能太多, 只给出部分图片, 具体可参照上述的仿真波形图.



1. 随机数发生器

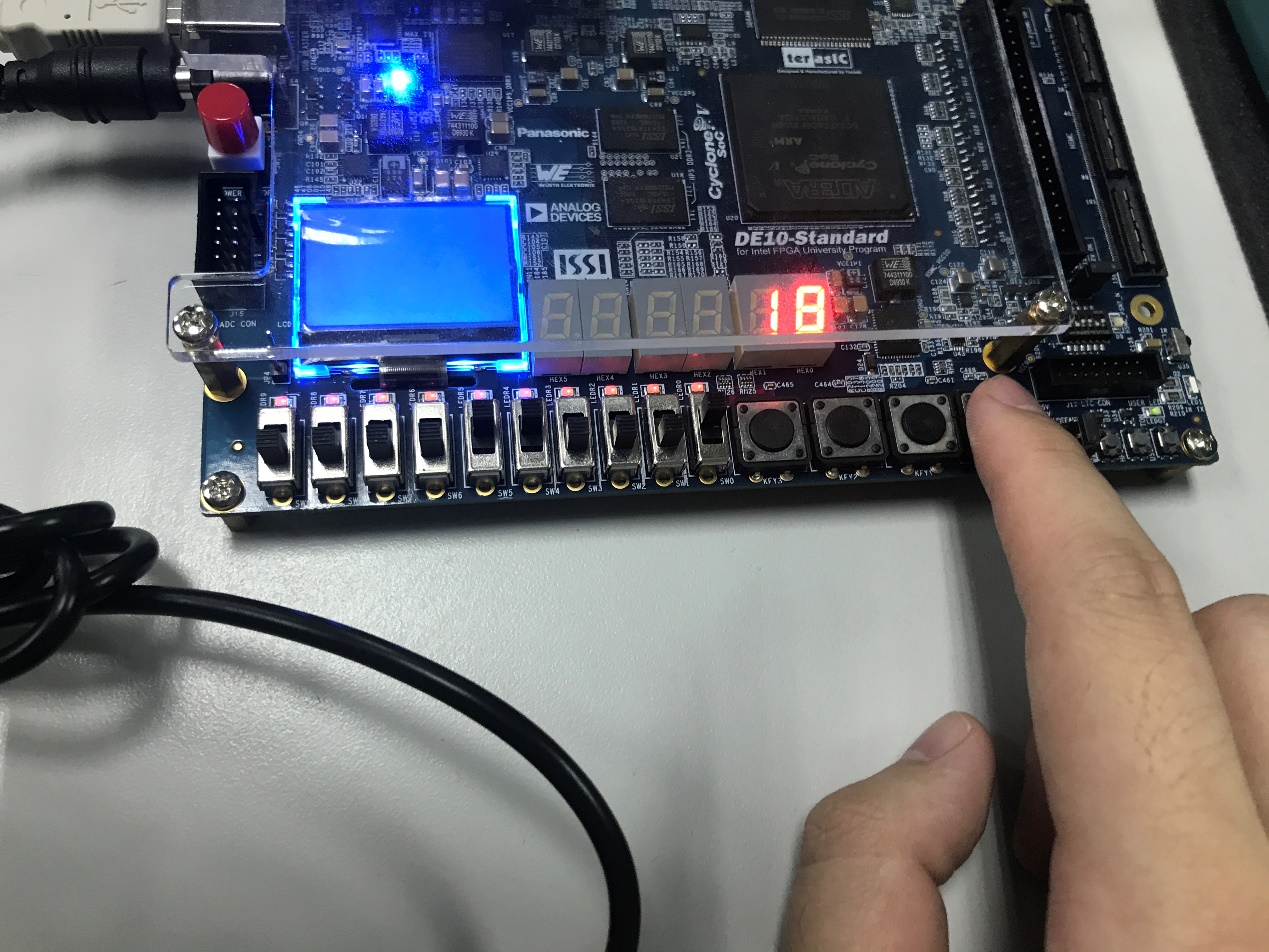
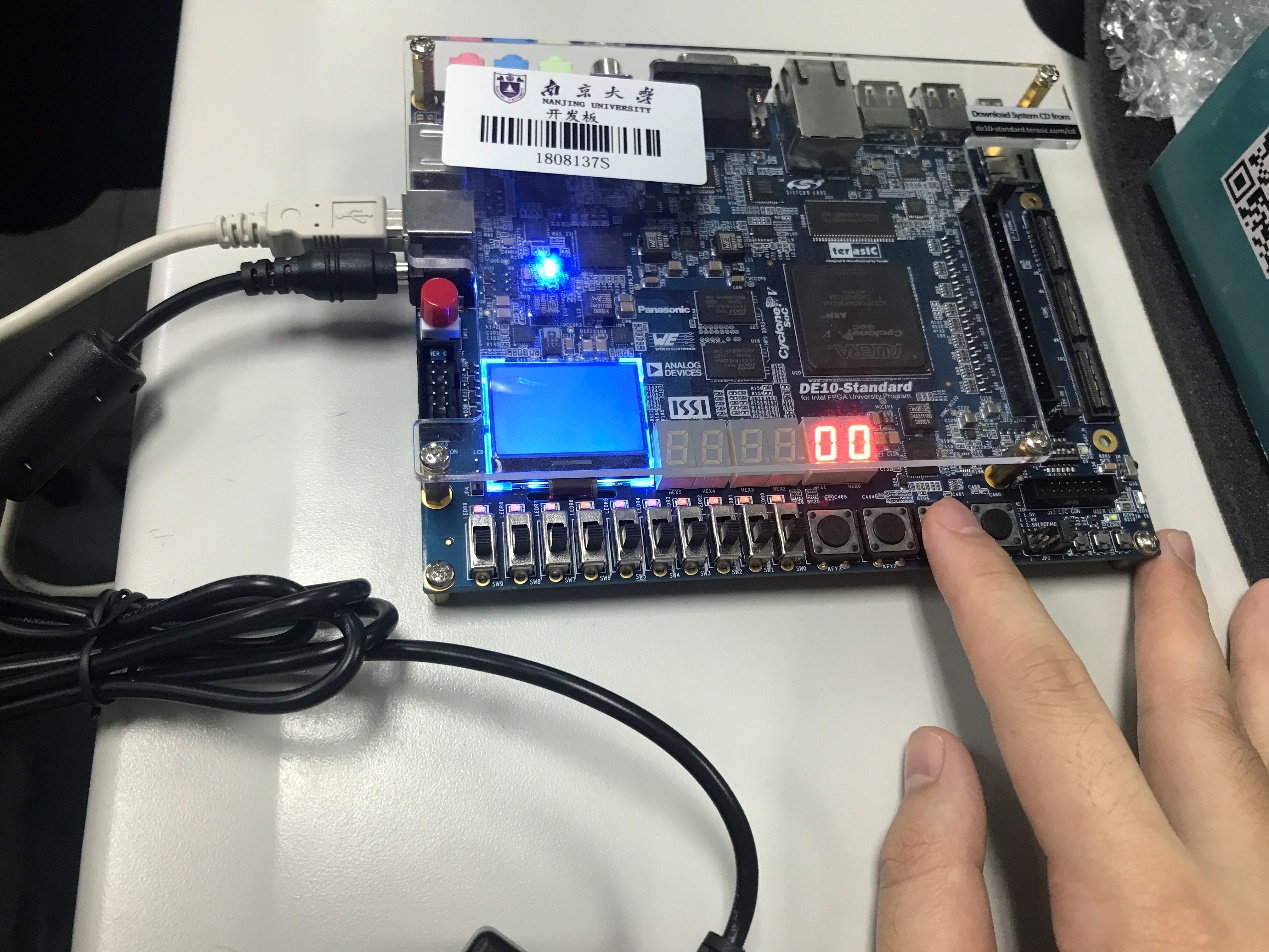
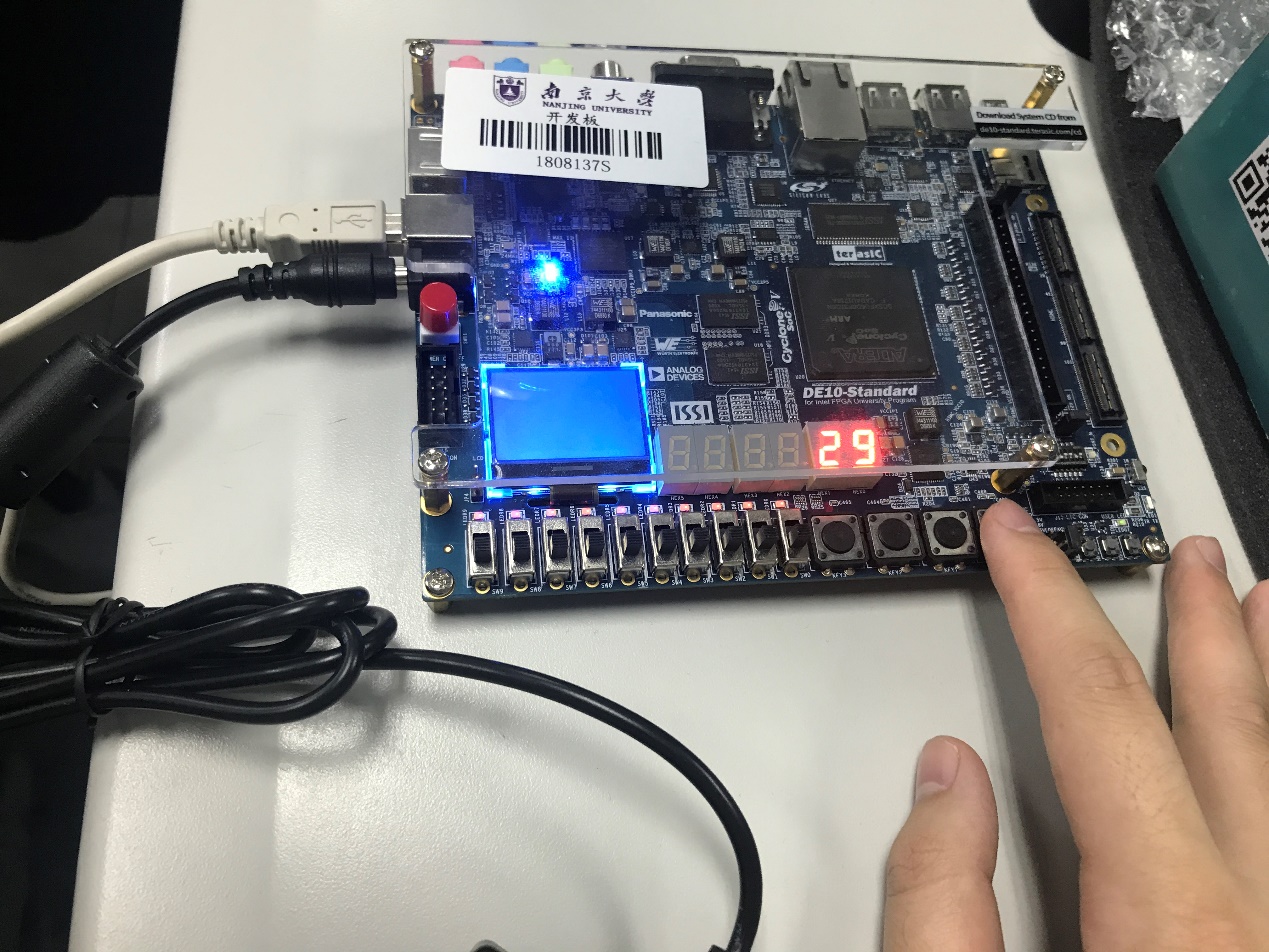
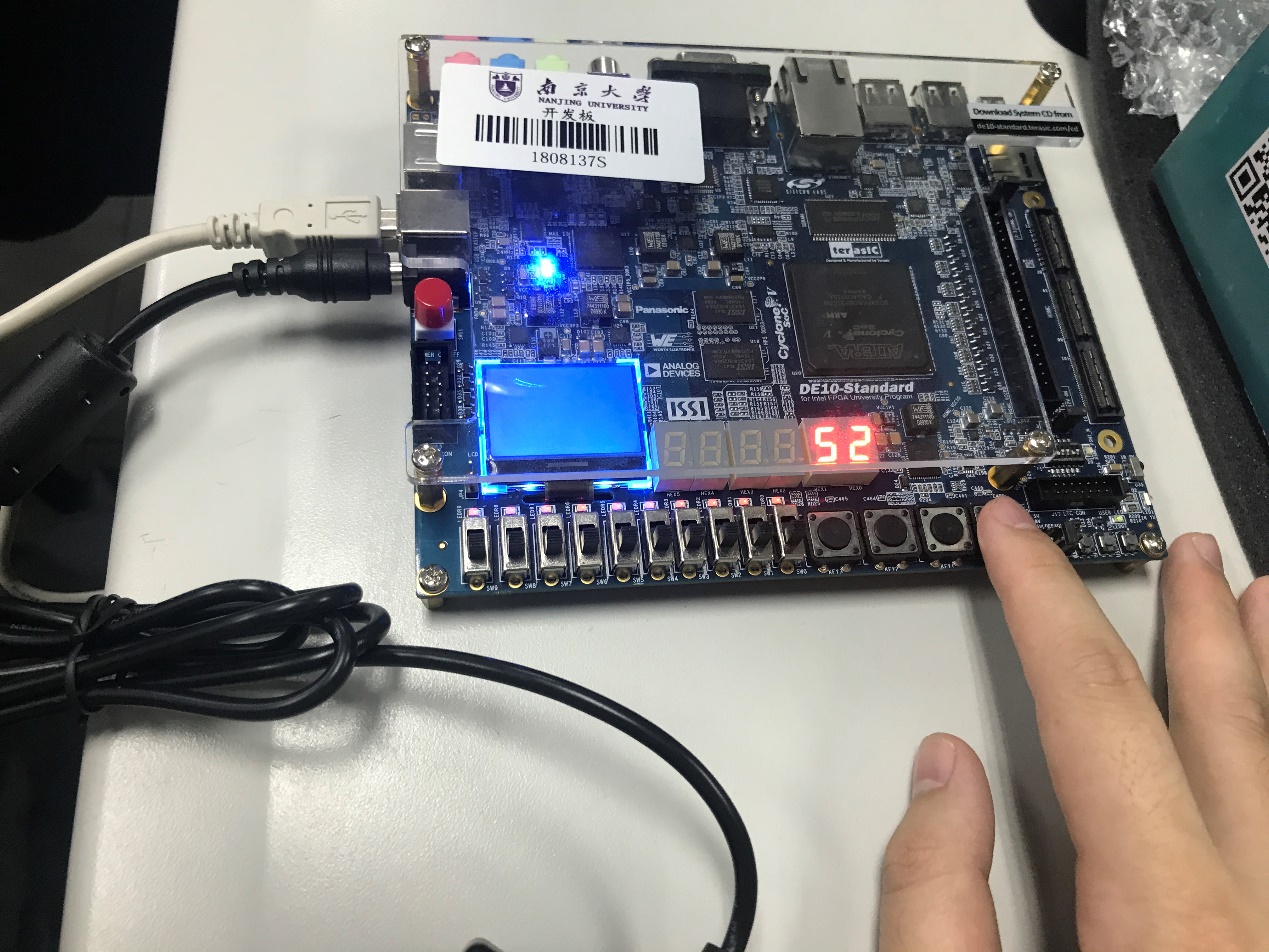
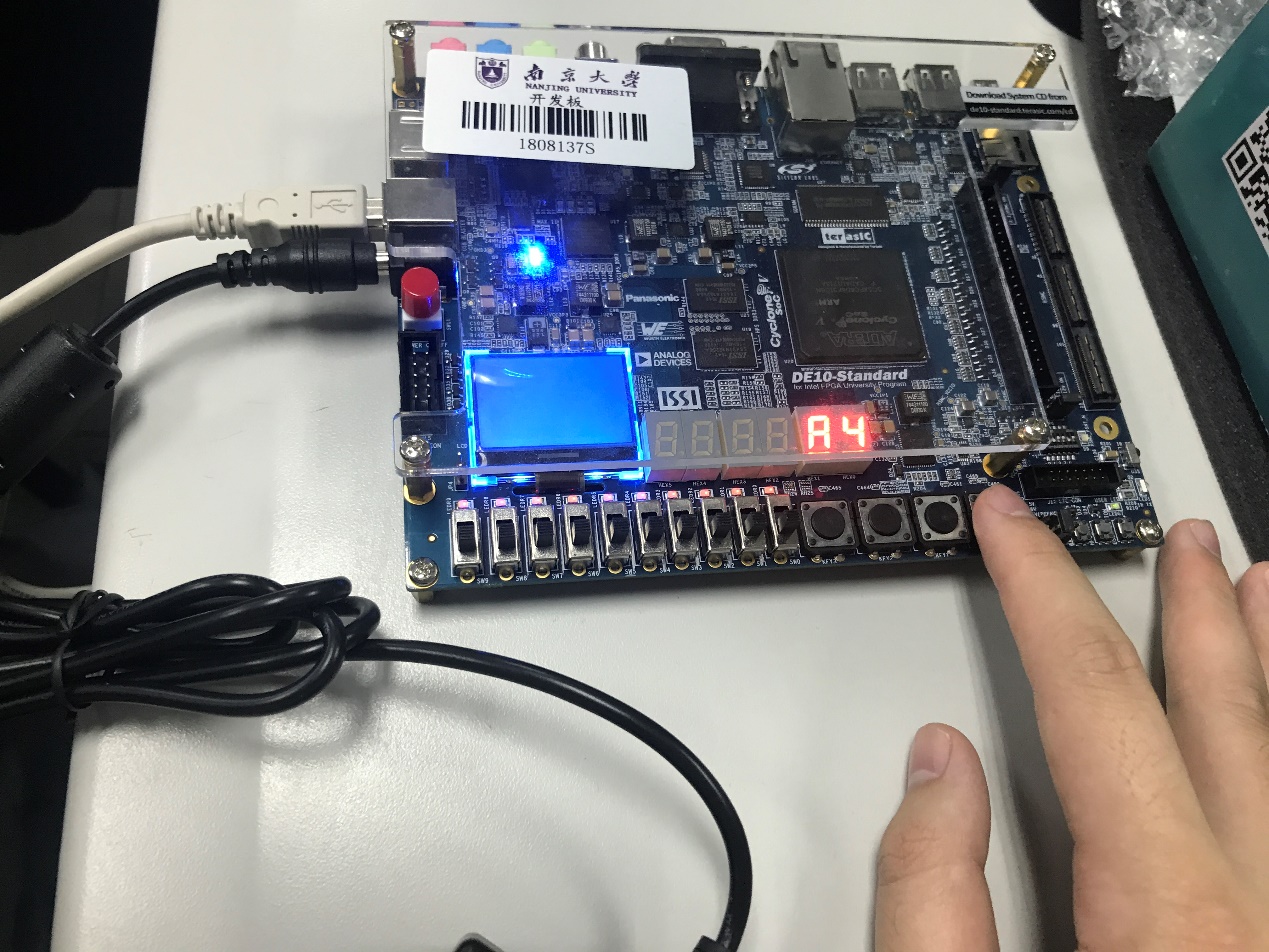
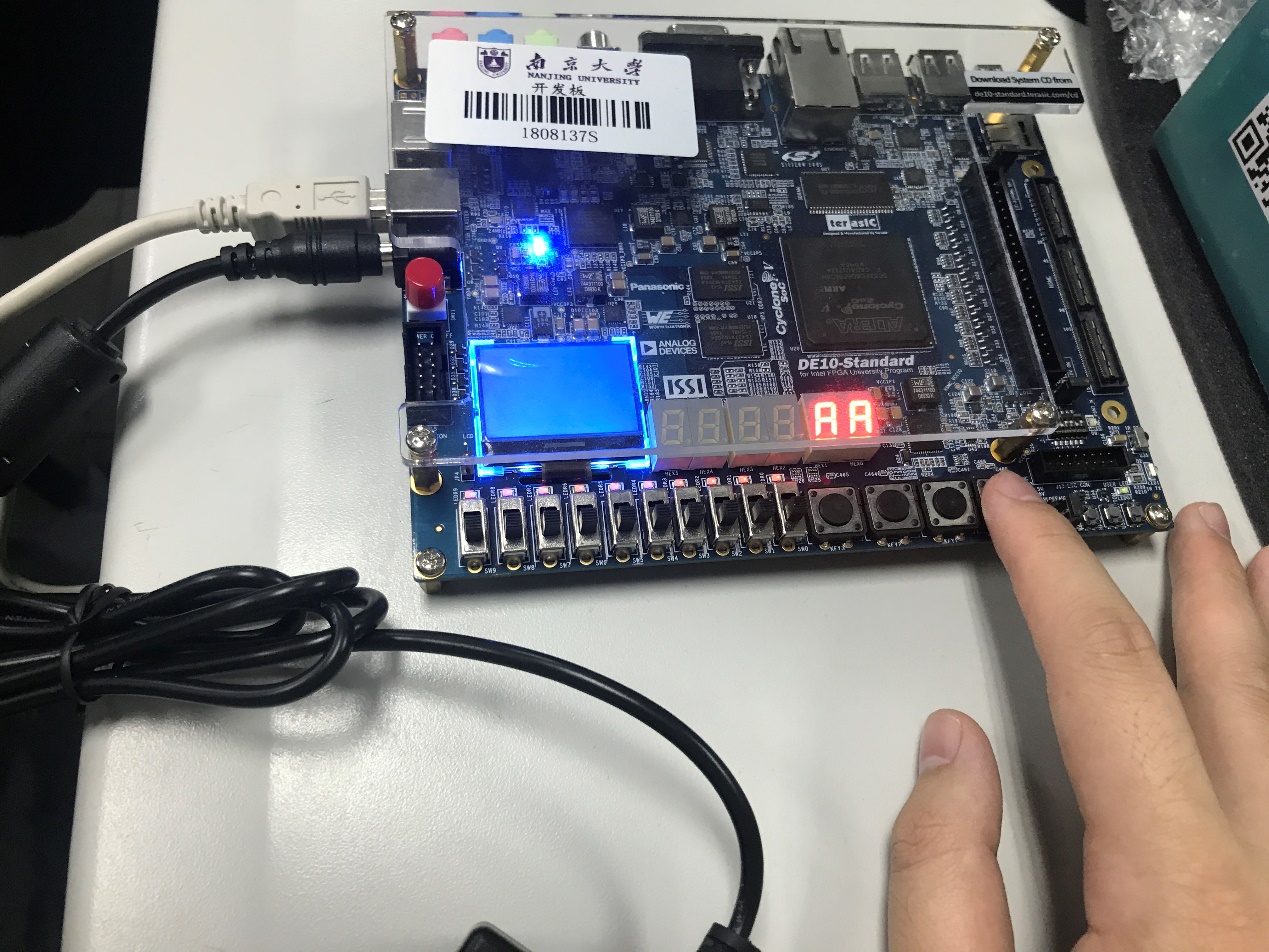
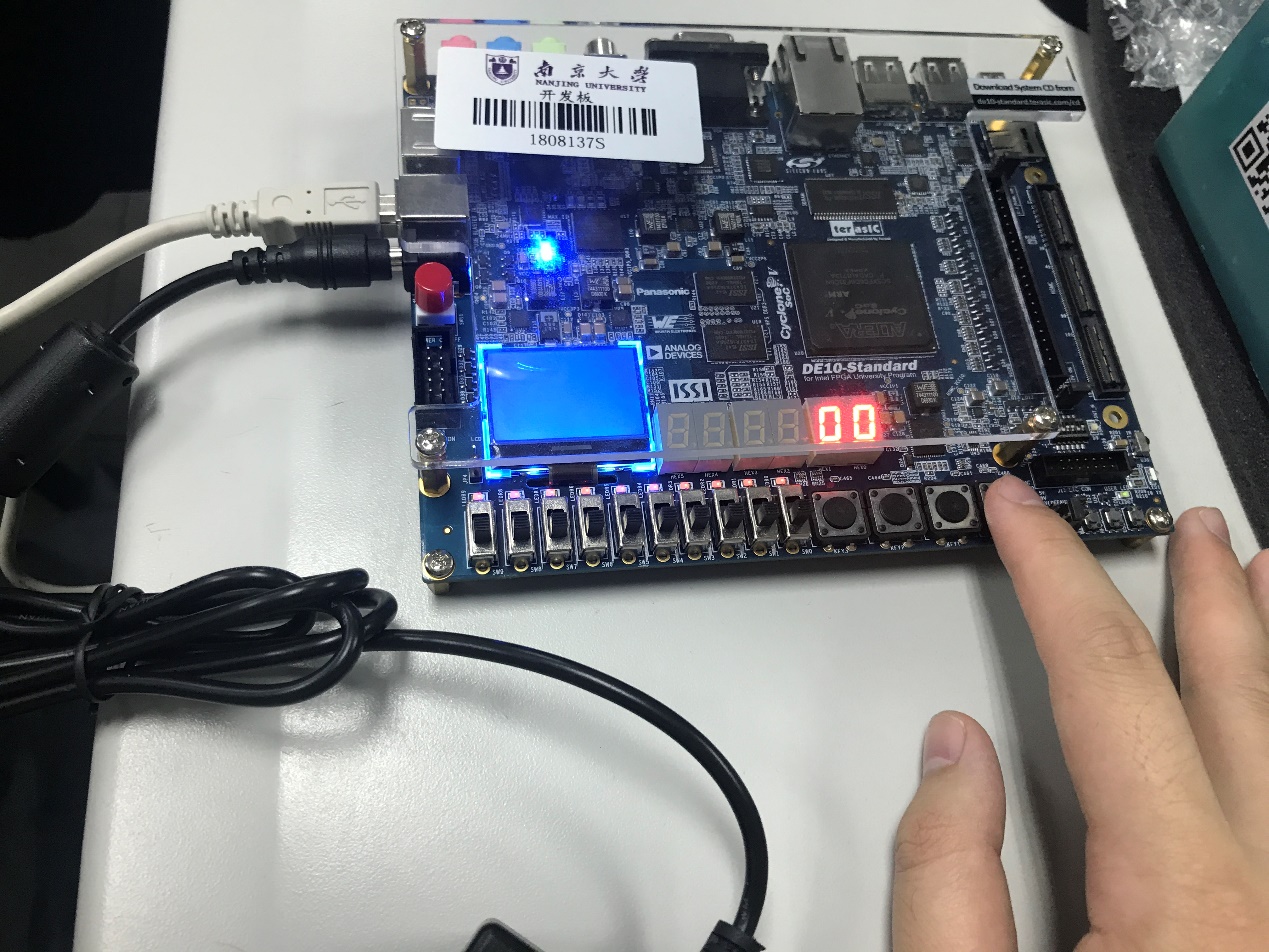
仿真波形图:



由于仿真验证软件初值为x和硬件不同, 自启动功能在波形图中没有得到体现, 自启动功能是通过硬件验证的方法验证的.

硬件验证:

具体见压缩包内视频.



1. **实验中遇到的问题及解决方案**

1.

问题: 一开始设计的随机数发生器的初值为0, 若使用根据书上的反馈方程设计出来的随机数生成算法, 则结果永远是0, 系统无法自启动.

解决方案: 在原有的基础上给随机数发生器加上了一个置数功能, 可以通过该功能给随机数发生器指定一个初值, 使其能够正常使用. 同时为了能使系统自启动,

在该随机数发生器触发时, 检测之前结果是否为0, 若为0, 则给结果赋一个不为0的值, 使其能够正常运行.

1. **实验得到的启示**
2. 当设计一个具有自启动功能的器件时, 可以在检测到无效状态后, 人为的使其跳转到一个有效状态.
3. 只通过程序和算法并不能产生真正的随机数, 但借助硬件可以产生统计意义上的真随机数, 如维护了一个熵池, 不断收集非确定性的设备事件(如时钟, IO请求的响应时间, 特定硬件中断的时间间隔, 键盘敲击速度, 鼠标位置变化, 甚至周围的电磁波等), 来作为随机数生成算法的种子.
4. **意见和建议**
5. DE10-Standard开发板上的按钮已经进行了消抖, 是开关没有进行消抖, 实验讲义上有误.
6. 可以多给出一些关于随机数发生器的设计指导.