Lab2:时序逻辑电路实验

1. 实验目的：

掌握锁存器、触发器和寄存器的设计方法和应用。

通过实验一中的寄存器堆设计，我们学习了如何设计和应用寄存器堆，包括锁存器和寄存器的使用，以及如何进行读写操作。

1. 掌握寄存器堆、计数器、移位寄存器的设计方法和应用。

实验一中的寄存器堆设计让我们熟悉了寄存器堆的设计和应用，同时也涉及到计数器和移位寄存器的原理，尤其在寄存器堆的时序逻辑操作中。

1. 掌握数字时钟的设计方法。

实验三中的数字时钟设计让我们掌握了数字时钟系统的设计方法，包括时钟信号的分频、状态机的设计、七段数码管和LED的控制等。这一实验提供了数字时钟设计的实际经验*。*

1. 实验环境：
2. Vivado 开发环境
3. Xilinx A7-100T 实验板
4. 实验原理：

1. 双稳态元件

锁存器和触发器是双稳态元件，由逻辑门和反馈电路构成。

锁存器使用电平控制，保持其状态在控制信号有效电平期间。

触发器使用时钟边沿触发，状态变化与时钟信号的上升沿或下降沿相关。

锁存器和触发器可用于构建时序逻辑电路，如寄存器、计数器和移位寄存器。

2. 触发器

触发器是时序逻辑电路的基本构建块，常用于存储和同步信号。

使用 Verilog 进行时序电路设计，通常采用 always 语句进行行为建模。

阻塞赋值（=）和非阻塞赋值（<=）语句用于对寄存器进行赋值。

同步复位和异步复位触发器的设计，根据需求选择合适的复位方式。

3. 寄存器

寄存器用于暂存信息，CPU中有通用寄存器组用于存储中间数据。

寄存器可以由多个触发器组成，可通过 Verilog 实现。

阻塞赋值和非阻塞赋值语句用于寄存器的行为建模。

同步和异步复位方式的设计，根据应用场景选择合适的方式。

4. 计数器

计数器用于对外部激励信号进行总数统计，通常从 0 开始计数。

计数器可实现各种功能，如测量、计数和控制。

通过组合逻辑和触发器构建，Verilog 代码用于描述其行为。

同步和异步清零、载入功能，根据需求进行设计和测试。

5. 移位寄存器

移位寄存器实现信息的左移和右移功能，用于数字系统中的特定操作。

可通过串联触发器实现移位寄存器，Verilog 代码描述其行为。

支持清零、保持、右移、左移和载入等功能，根据应用需求配置。

1. 实验内容：
2. 寄存器堆设计：

1. 背景

CPU中的寄存器堆是一个关键的组件，用于暂存指令执行过程中的中间数据。本实验的目标是设计一个包含32个32位寄存器的寄存器堆，支持读取和写入操作，并使用Vivado进行仿真、综合、实现，并在实验开发板上验证。

2. 目标

设计一个32个32位寄存器的寄存器堆。

实现读取操作，支持两个读数据端口。

实现写入操作，需要时钟信号控制。

验证寄存器堆的功能，包括读取和写入操作。

3. 设计

3.1 寄存器堆模块

我们将创建一个名为regfile32的Verilog模块，用于实现32个32位寄存器的寄存器堆。

3.2 顶层模块

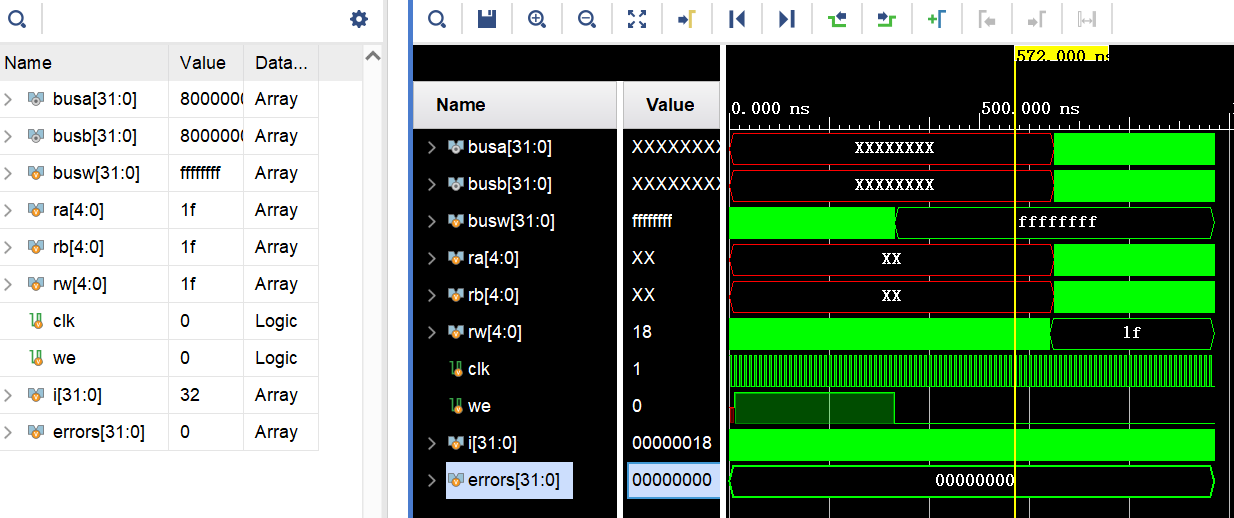
我们还将创建一个顶层模块regfile\_top，用于连接开关、LED指示灯以及regfile32模块。：

3.3 连接寄存器堆

在顶层模块regfile\_top中，我们需要实例化regfile32模块，并连接其输入和输出信号。这将确保寄存器堆与其他部分正确通信。

4. 仿真测试

完成设计后，我们将使用Vivado进行仿真测试，以验证寄存器堆的功能。在仿真测试期间，我们将提供输入数据，观察输出数据，并分析时序波形和控制台信息。

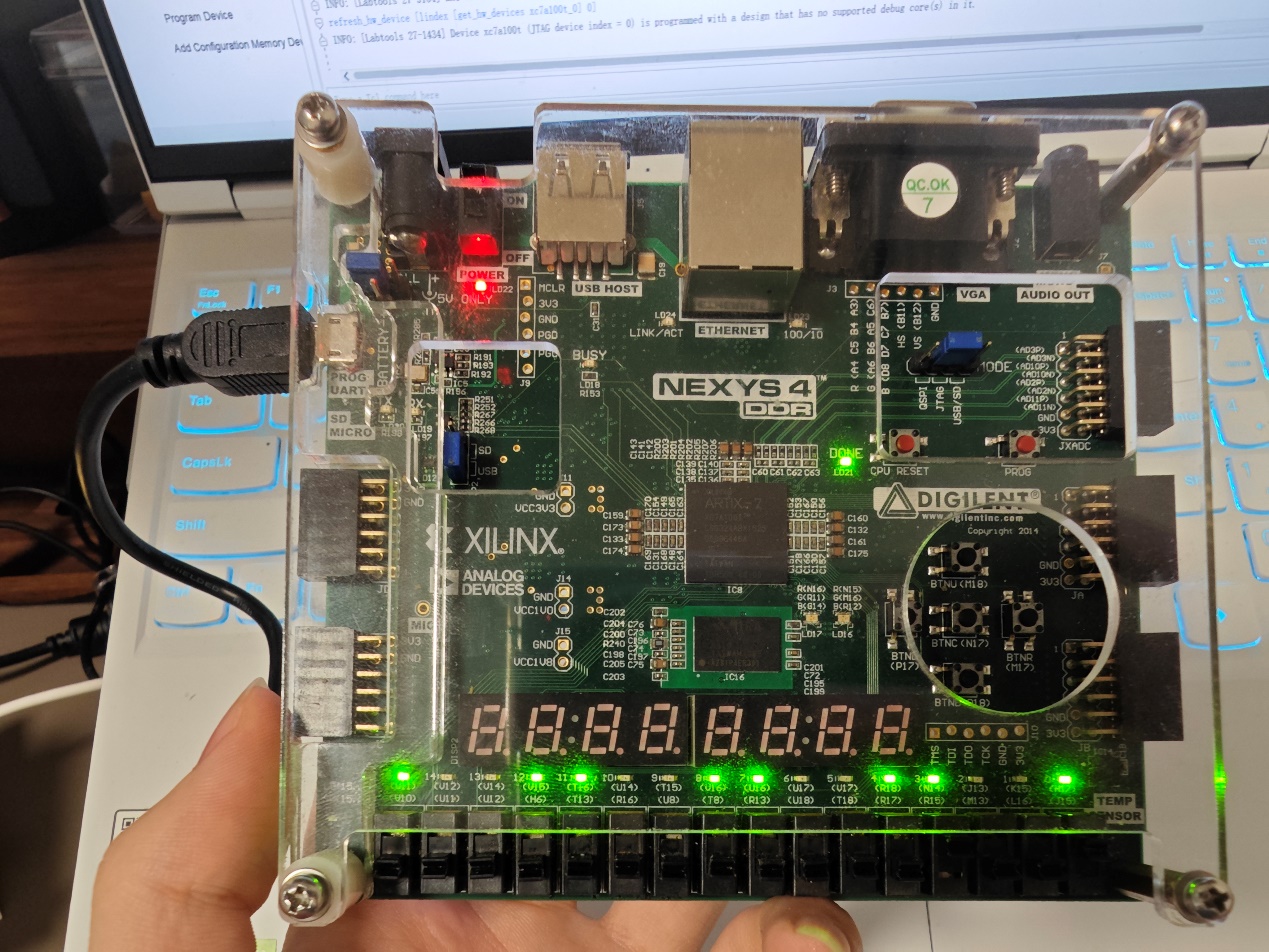


5. 综合与实现

仿真通过后，我们将使用Vivado工具进行综合和实现，以生成比特流文件。这个文件将用于加载到实验开发板进行验证。

6. 验证

最后，我们将加载生成的比特流文件到实验开发板上，使用拨档开关、LED指示灯和按钮等元件进行验证。我们将记录验证过程并确保寄存器堆在实验开发板上正常运行。



比特流加密实验

1.实验目的：

通过使用线性移位寄存器（LFSR）生成随机比特流，并利用该流对字符串进行加密和解密，以理解加密和解密的基本概念。

2.实验原理：

本实验使用了一个64位的LFSR，其反馈方程为x64 = x4 ⊕ x3 ⊕ x1 ⊕ x0，并且初始化seed为64'ha845fd7183ad75c4。LFSR的工作原理是在每个时钟上升沿时进行移位操作，并将生成比特计数加1，当计数达到6的倍数时，输出ready信号为高电平，说明输出有效。在ready信号为高电平时，将输入数据的低6位与LFSR输出的高6位异或，并输出。如果不是6的倍数周期，测试系统会用"x"替代实际输出。

3.实验步骤：

创建一个新的Vivado工程。

添加设计源码文件，encryption6b.v。

添加仿真测试文件，encryption6b\_tb.v。

添加约束文件，包括encryption6b.xdc。

根据实验要求完成源码文件的设计。

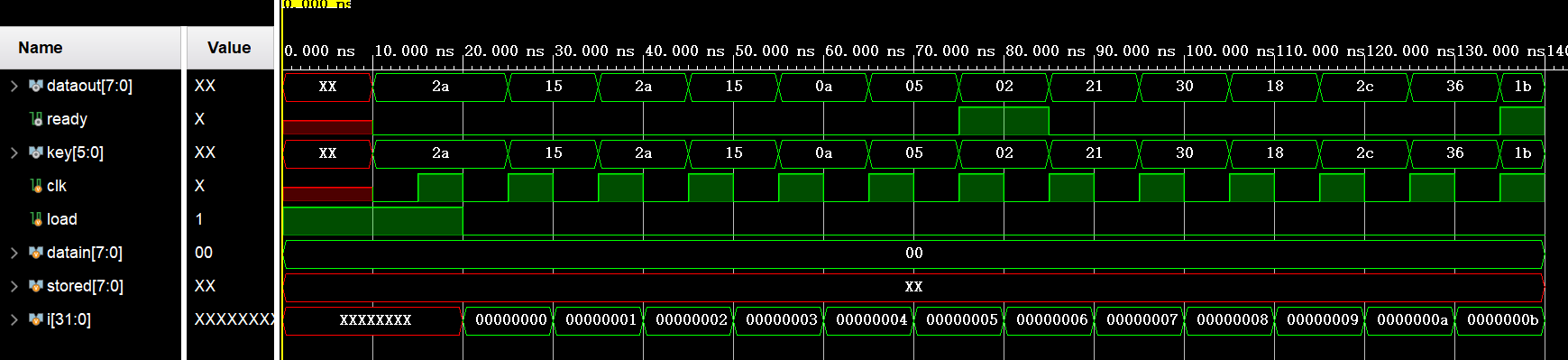
对工程进行仿真测试，分析输入输出时序波形和控制台信息。

仿真通过后，进行综合、实现并生成比特流文件。

生成比特流文件后，加载到实验开发板，进行调试验证，并记录验证过程。

4.仿真结果：

在Vivado中进行了仿真测试，分析了输入输出时序波形和控制台信息。在仿真中，我们看到了数据输入、加密、解密和ready信号的波形，并确保加密和解密过程按照预期进行。

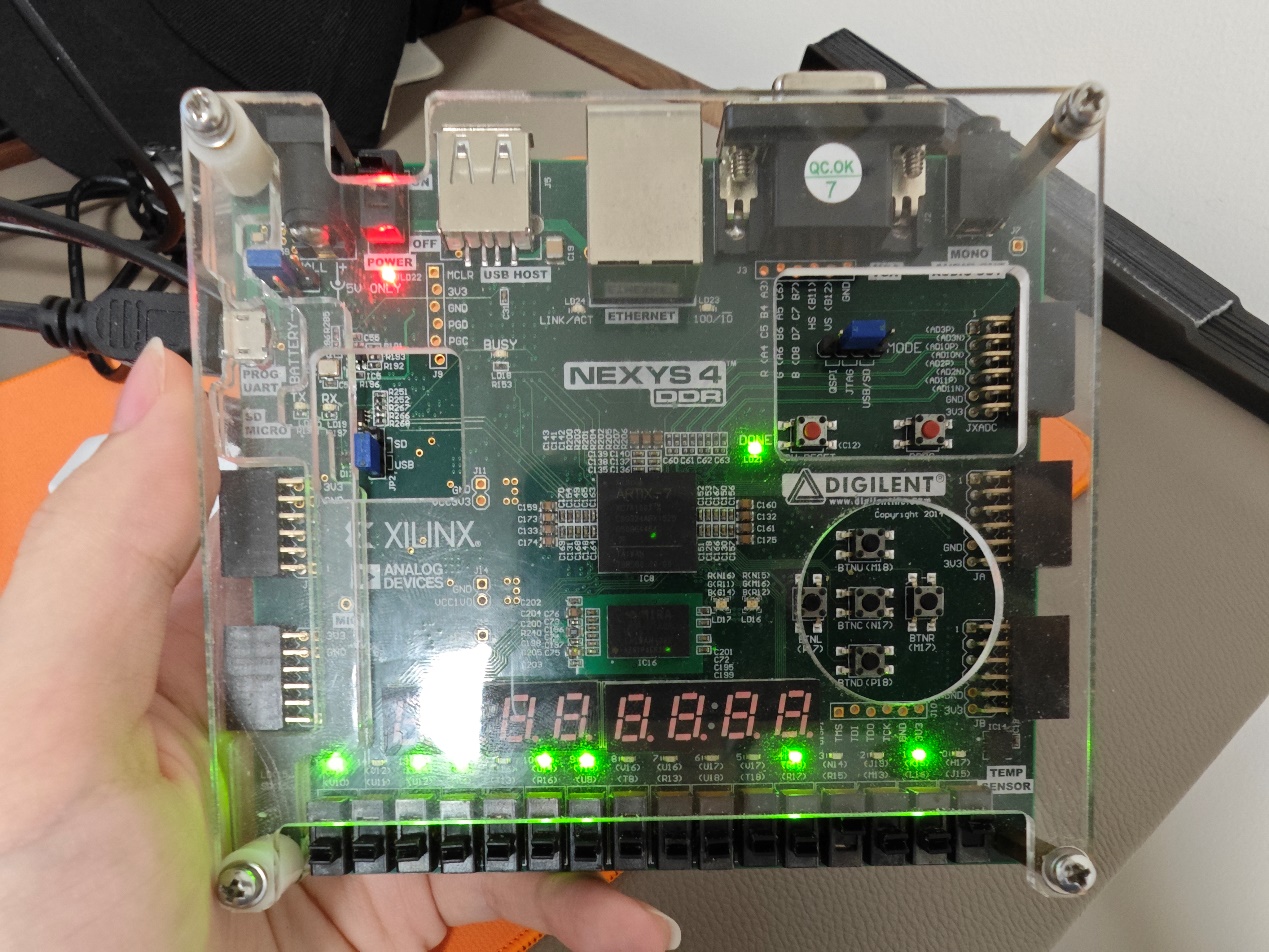
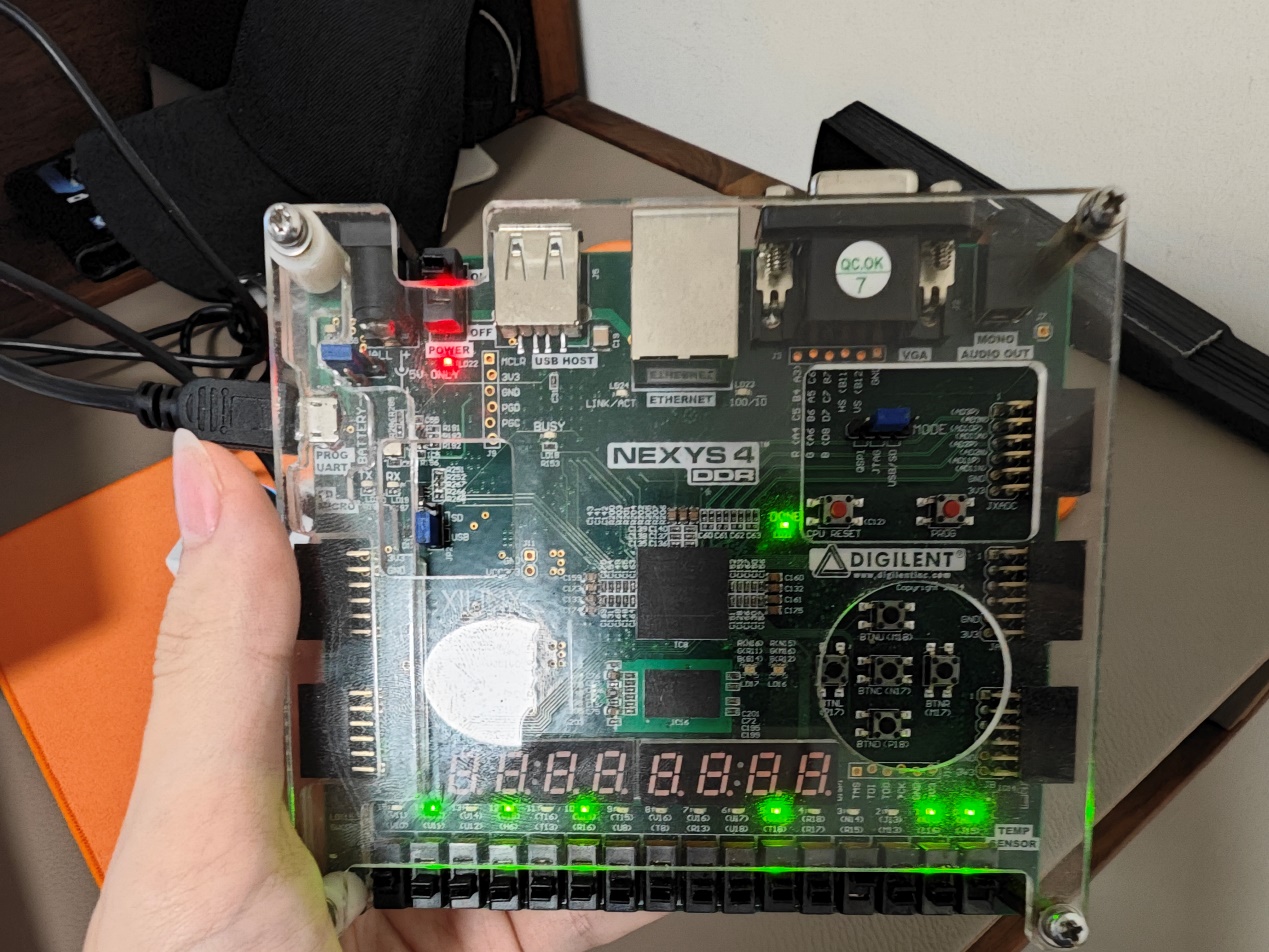


5.综合和实现：

综合和实现步骤根据Vivado工具的指导进行，最终生成了比特流文件。

6.验证和测试：

将生成的比特流文件加载到实验开发板中，通过拨动开关输入7位输入数据，使用LED指示灯观察密码和加密后的数据。通过按下按钮模拟时钟信号，确保加密和解密过程按照预期进行。验证了加密和解密的正确性。



数字时钟实验

1.实验目的：

本实验旨在设计并实现一个数字时钟系统，能够显示当前的小时、分钟和秒数，具备复位功能，能够设置初始时间，以及在整点报时时闪烁点亮三色LED灯。

2.实验原理：

数字时钟系统基于100MHz的时钟信号进行分频，以实现1Hz的计时。通过一位按钮（BTNC）进行复位操作，可以将时钟清零。通过两位输入开关（s）设置时钟的工作状态，包括正常计时状态、设置小时初值状态、设置分钟初值状态和设置秒初值状态。在设置状态下，高位和低位的BCD码用于表示初始值。时钟的显示通过七段数码管实现，同时使用七段数码管控制位（an）来控制时、分、秒的显示。此外，在整点时，将轮流点亮三色LED灯（ld16）持续5秒，并在计时到23时59分59秒后进行清零操作，然后轮流点亮LED灯持续10秒。

3.实验步骤：

工程创建： 使用Vivado创建一个新的工程。

添加文件： 添加设计源码文件（DigitalTimer.v）和约束文件（DigitalTimer.xdc）。

设计实验代码： 根据实验要求，完成Verilog源码文件的设计，包括数字时钟模块。

仿真测试： 对工程进行仿真测试，分析输入输出时序波形和控制台信息，确保数字时钟功能按照预期工作。

综合和实现： 使用Vivado工具进行综合和实现，生成比特流文件。

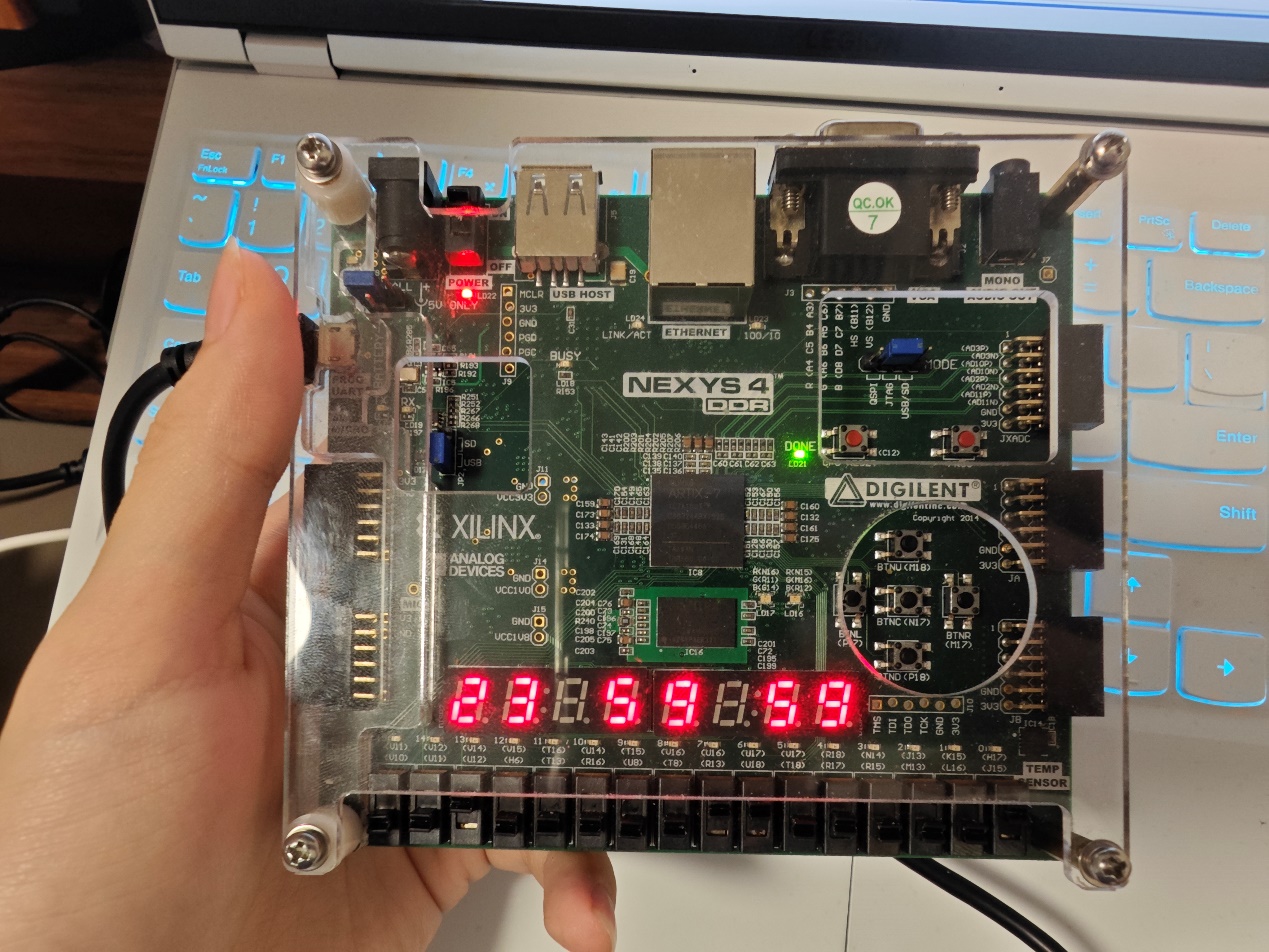
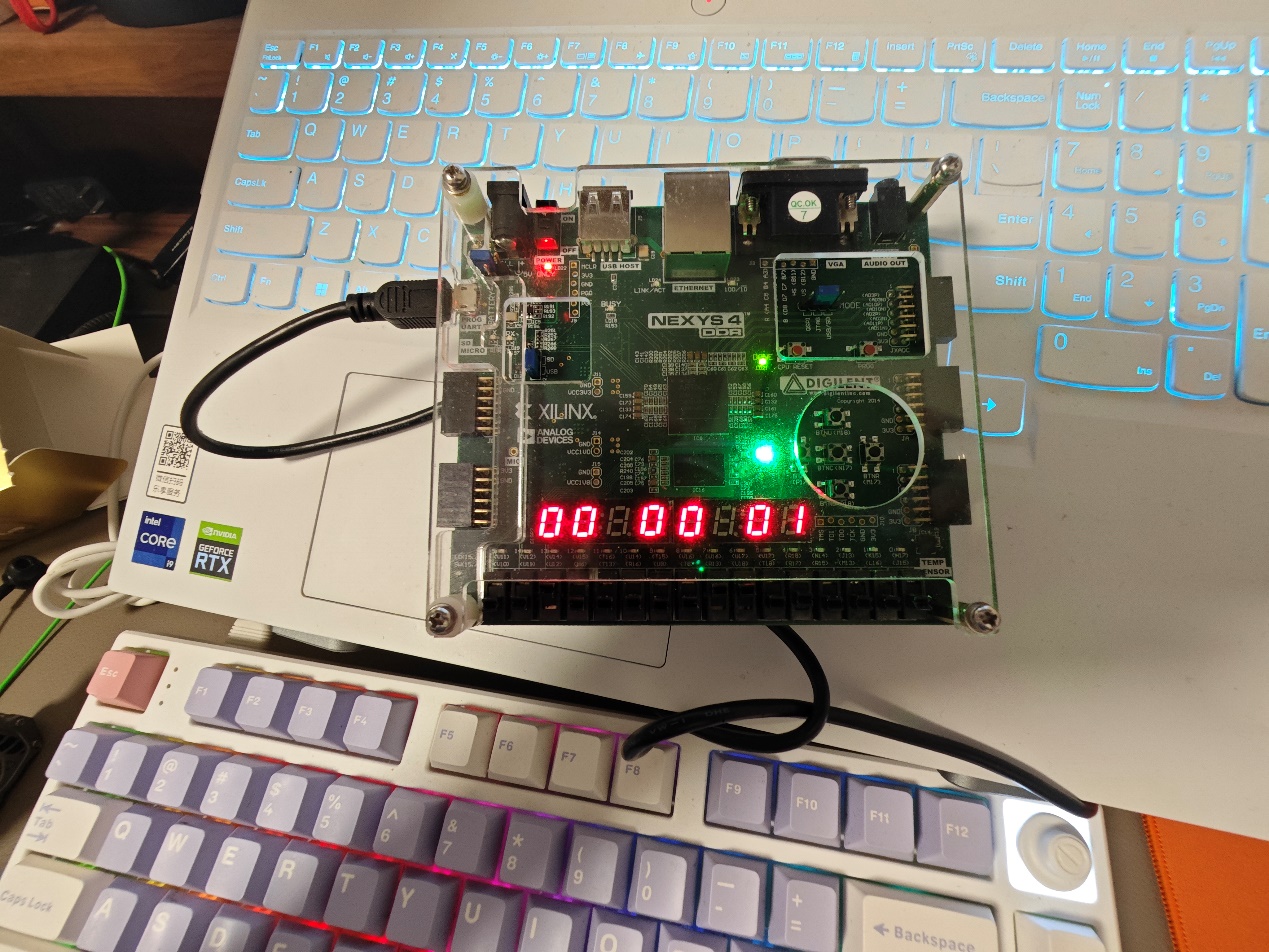
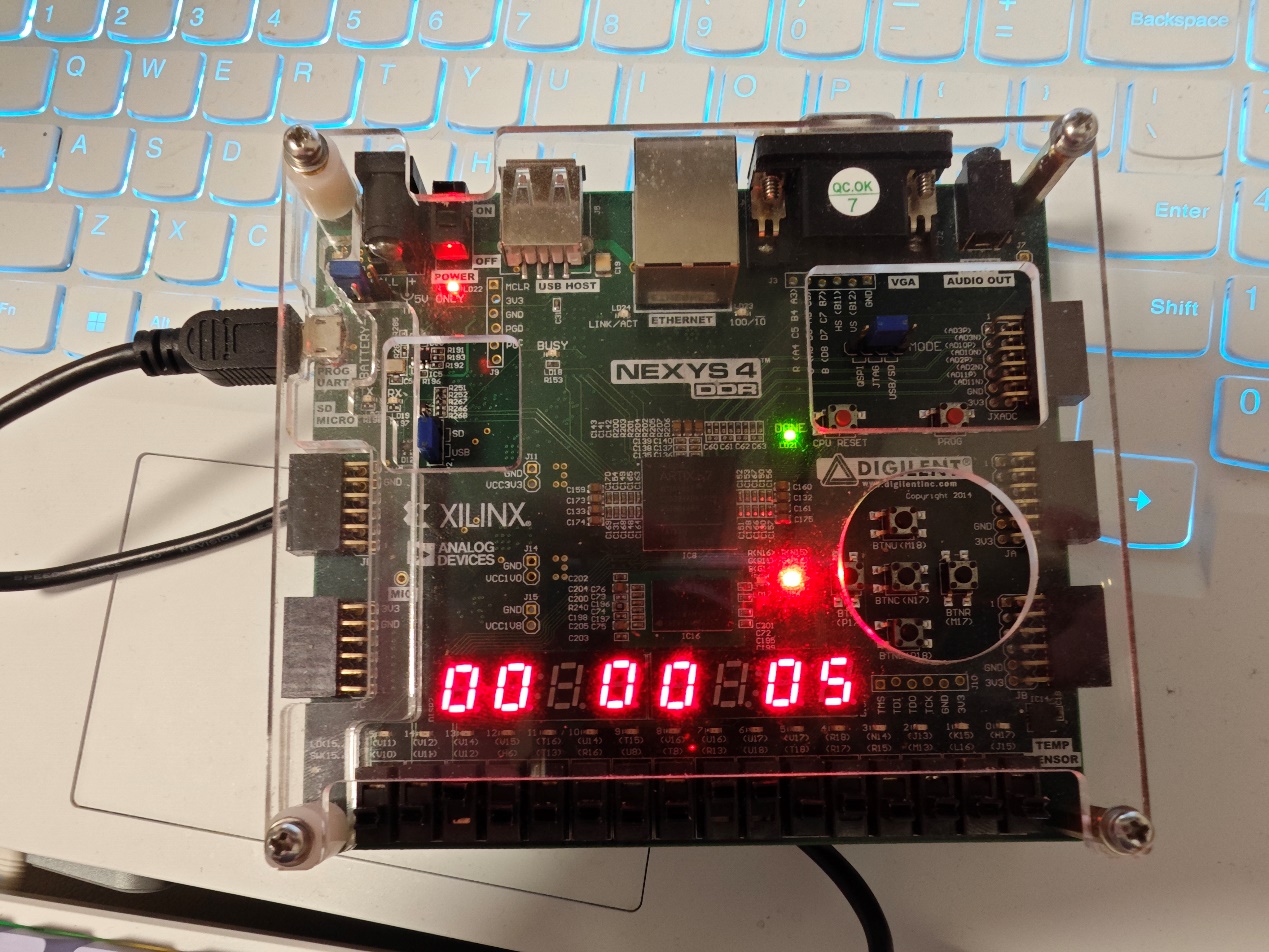
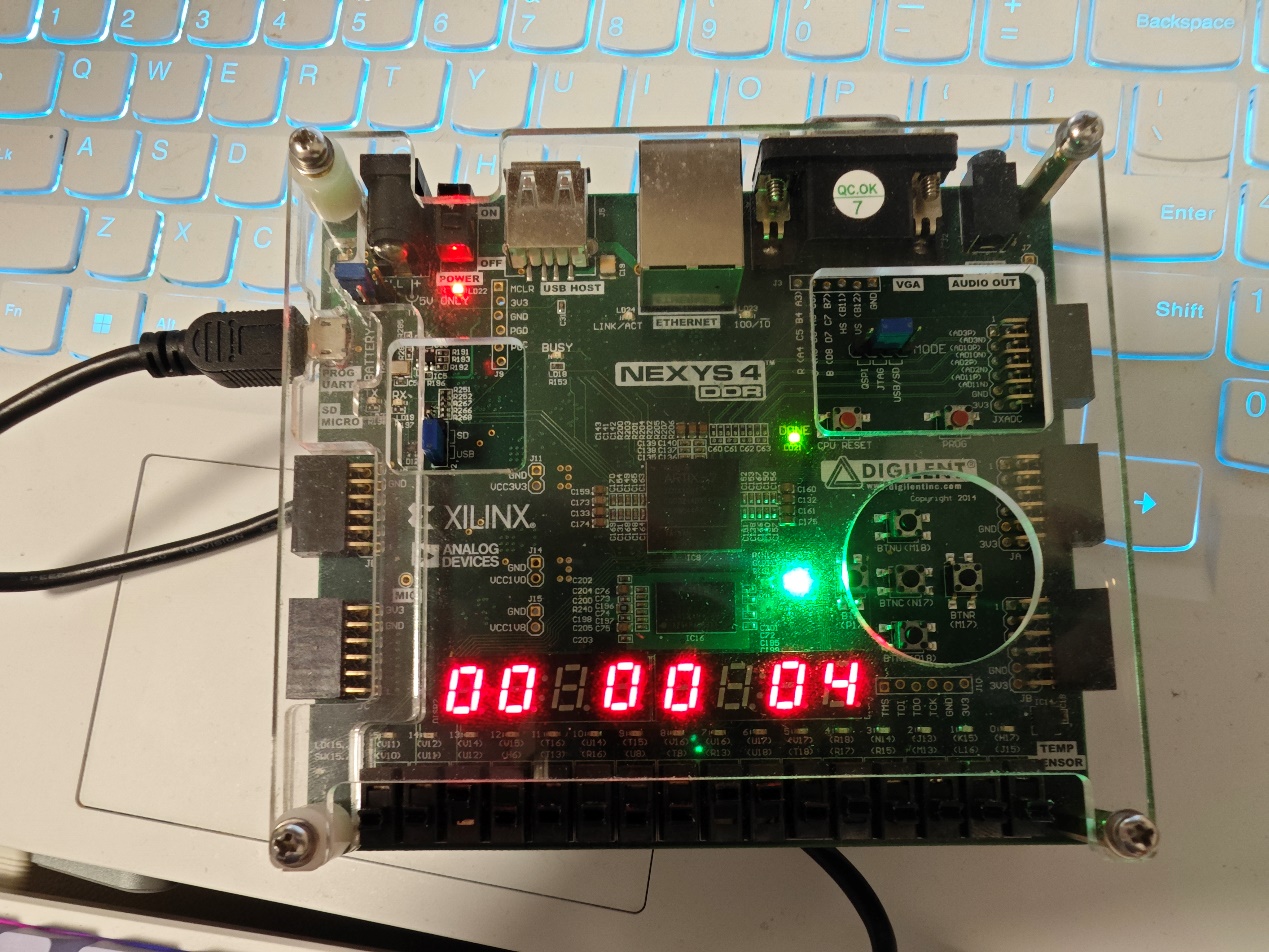
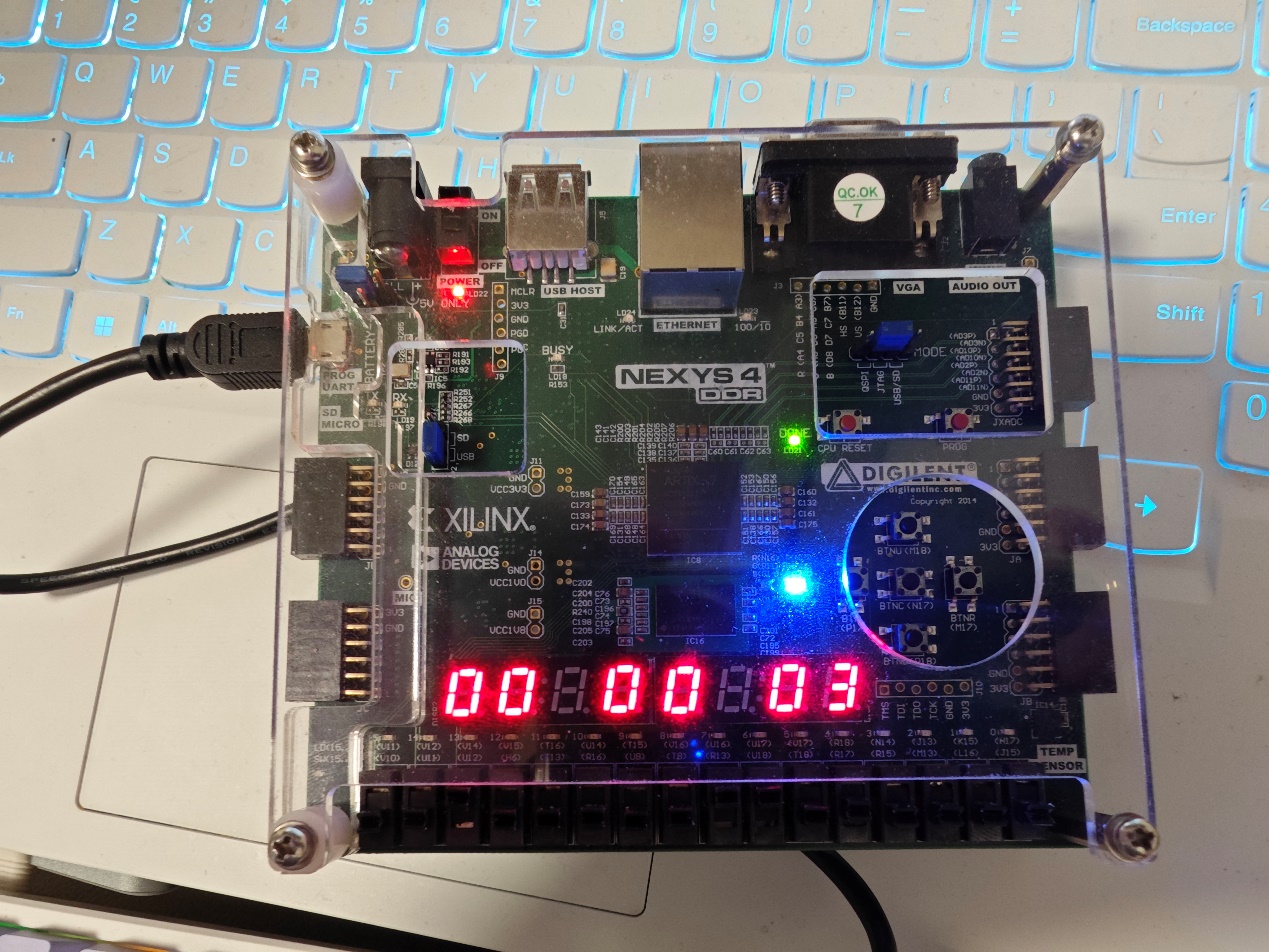
4.验证测试：

在Vivado中进行了仿真测试，分析了输入输出时序波形和控制台信息。在仿真中，我们观察了不同状态下时钟的工作情况、七段数码管的显示和LED灯的闪烁。

文本

描述已自动生成（根据需测试功能修改仿真测试代码）

将生成的比特流文件加载到实验开发板中，通过按钮BTNC进行复位操作，使用输入开关s进行设置和切换工作状态，观察七段数码管上的时钟显示，同时在整点时观察LED灯的闪烁，验证数字时钟的功能是否正确。



1. 思考题：
2. 分析32个32位的寄存器堆占用的逻辑片资源。

图形用户界面, 应用程序, Teams

描述已自动生成

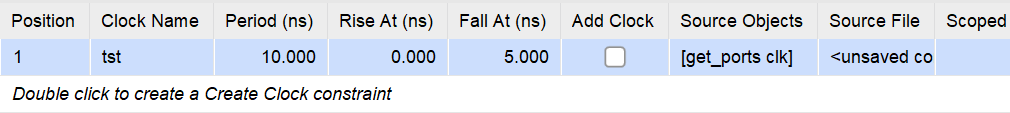
LUTs占用情况如图所示

图表

描述已自动生成

1. 分析64位移位寄存器的时序性能和资源占用情况；并通过资料查找到其他的生成LFSR的反馈公式

设置时钟：



时序报告：

图形用户界面, 文本, 应用程序

描述已自动生成

无红色警告，时序正常

资源占用情况：

表格

描述已自动生成

表格

描述已自动生成

图表

描述已自动生成

其他生成LFSR的反馈公式：

7 位 LFSR：

反馈公式：x7 = x6 ⊕ x0

15 位 LFSR：

反馈公式：x15 = x13 ⊕ x14

23 位 LFSR：

反馈公式：x23 = x18 ⊕ x17

31 位 LFSR：

反馈公式：x31 = x28 ⊕ x27

XORSHIFT 算法：

XORSHIFT 算法是一种伪随机数生成算法，不同于传统的 LFSR，但同样可以生成伪随机序列。

反馈公式：x = x ^ (x << a)

1. 数字时钟中如何实现倒计时和毫秒计时器功能
2. 控制信号增加一位
3. 为实现倒计时功能设置新的逻辑判断语句，与计时相同从低位开始判断
4. 为实现毫秒计时器功能设置新频率时钟作为信号控制每毫秒计数，其余部分与计时功能相同即可