**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称：** | **数字逻辑** |
| **学生姓名：** | **张旭超 葛子午** |
| **学生学号：** | **202330552991** |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2022-2023学年第二学期** |

**华南理工大学**

**2023年5月**

**目 录**

[实验一 Using IP Integrator 1](#_Toc169380487)

[实验二 Buildling Basic Elements for IPl 5](#_Toc169380493)

[实验三 Create a Simple Digital Circuit. 9](#_Toc169380499)

[实验四 Finite State Machines. 14](#_Toc169380532)

[实验4- Moore FSM（1-1） 15](#_Toc169380533)

[实验4- Mealy FSM（2-1） 20](#_Toc169380555)

# Using IP Integrator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7 楼 | 138 房； | **实验台号：** | 143 |
| **实验日期与时间：** | 2024/6/1 | | **评 分：** |  |
| **预习检查纪录：** | 无 | | **实验教师：** | 吴秋霞 |
| **电子文档存放位置：** | 无 | | | |
| **电子文档文件名：** |  | | | |
| **批改意见：** |  | | | |

**实验1-** **Using IP Integrator**

1. **实验目的**
2. 针对 Nexys4 或 Basys3 板上的特定 FPGA 器件创建 Vivado 项目
3. 使用提供的部分完成的 Xilinx 设计约束 （XDC） 文件来约束某些引脚位置



1. 使用 Vivado 的 Tcl 脚本功能添加其他约束
2. 使用 XSim 仿真器仿真设计
3. 综合并实施设计
4. 生成比特流
5. 使用生成的比特流配置FPGA并验证其功能

**二、实验原理**

使用 Xilinx Vivado 软件，通过 Vivado IP Integrator （IPI）创建简单的数字电路。整个设计流程涉及从创建项目到生成比特流，并最终在硬件上验证功能。实验的核心目标是掌握针对特定 FPGA 器件（如基于 Artix-100t 的 Nexys4 或基于 Artix-35t 的 Basys3 板）的设计流程。

完成本实验，需要正确安装软件并激活。按照实验步骤所示完成操作的认识与熟练，最后得到仿真效果并实现设计。

**三、实验环境**

（1）硬件环境需求：xc7a35tcpg236-1(Basys3)

（2）软件环境需求：Vivado 2014.4

**四、实验内容**

（1） 创建 Vivado 项目，设置和配置用户定义的 IP 库，以便在设计中使用特定的 IP 核。

（2）通过 Vivado IP Integrator 添加和配置所需的 IP 核，构建设计模块。

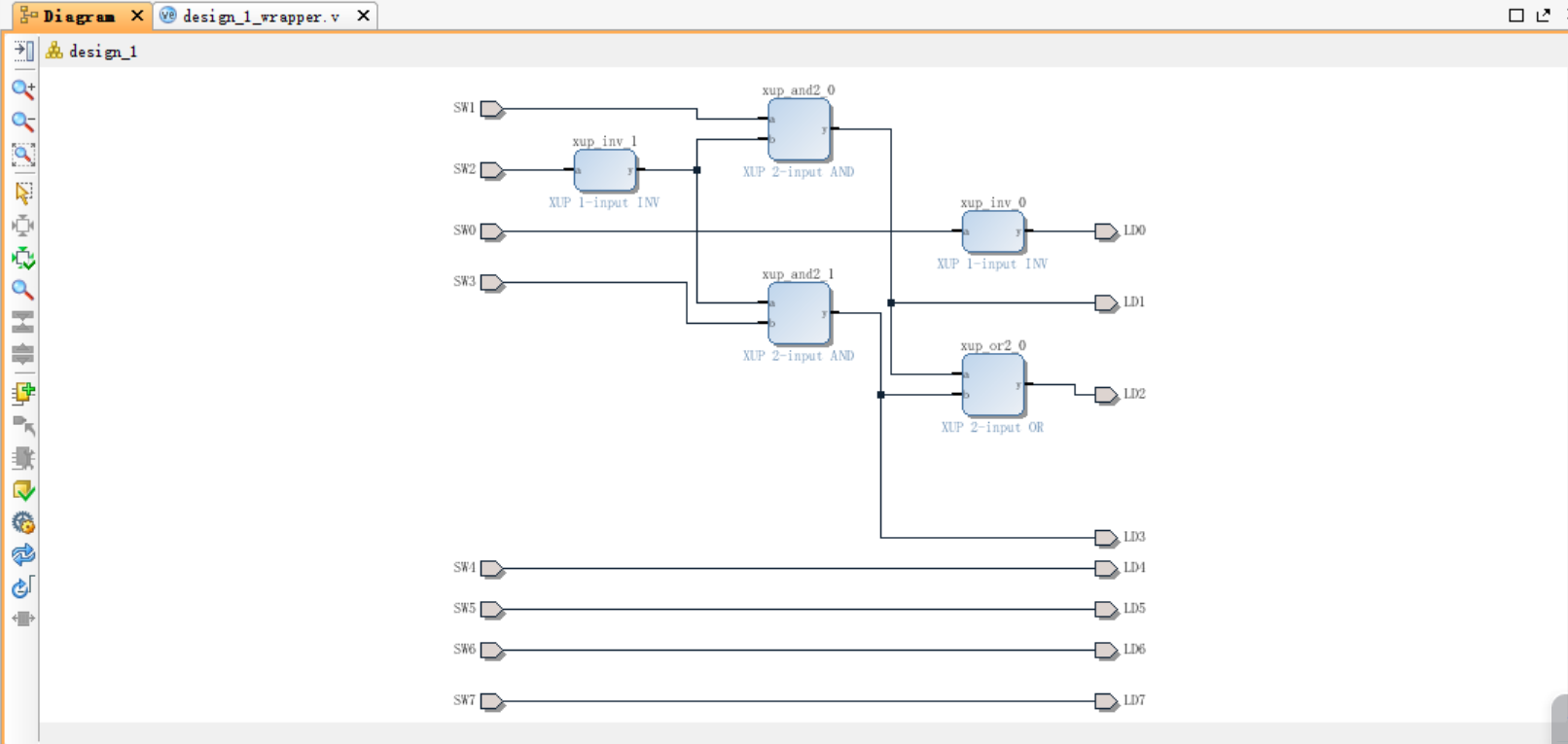
（3）为设计生成 HDL 包装器，将 IP 核和其他模块连接在一起形成顶层设计文件。

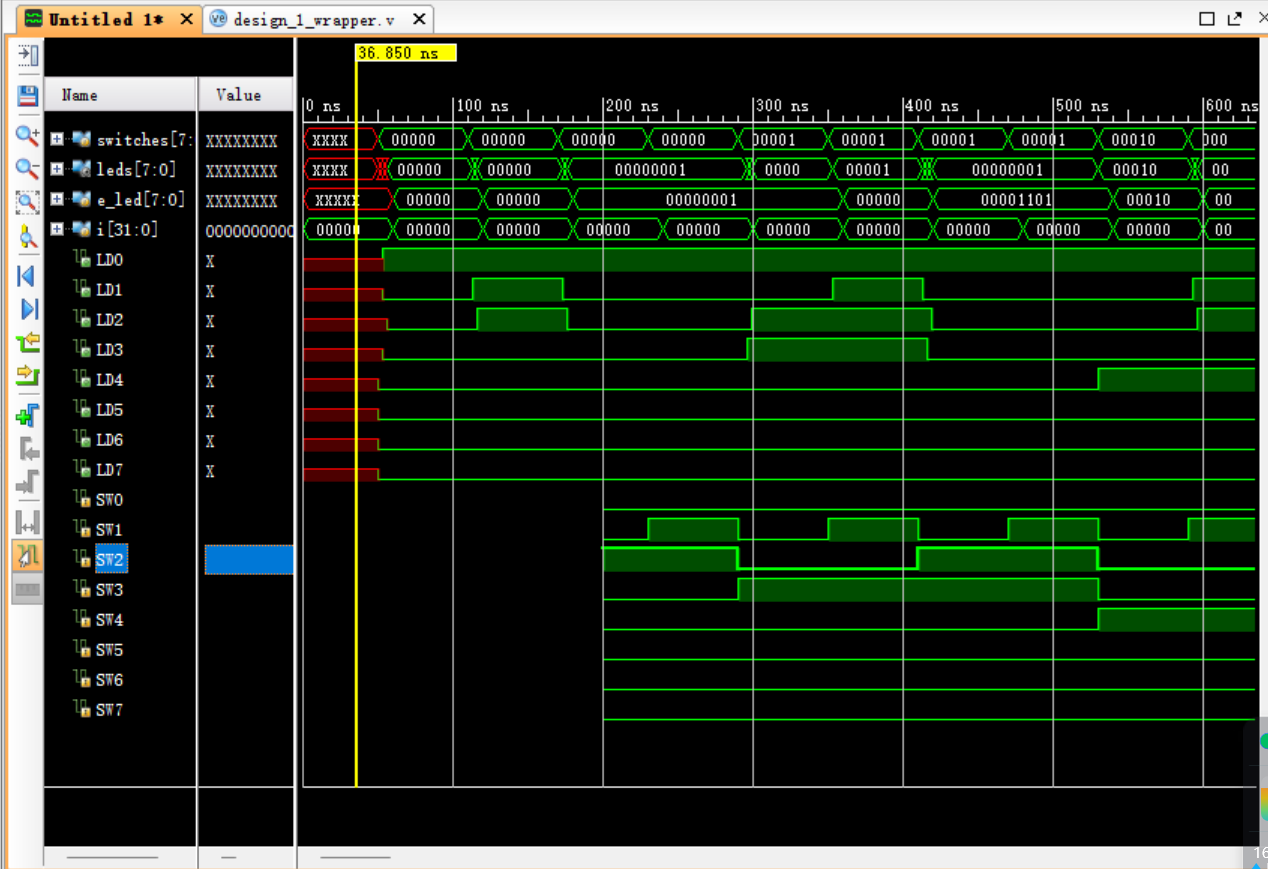
（4）创建和/或添加用户约束文件，使用提供的 Xilinx 设计约束 （XDC） 文件对某些引脚位置进行约束。

（5）运行行为仿真并将 HDL 代码和 IP 核综合成门级网表，检查综合报告，确保设计在资源和时序上满足要求。

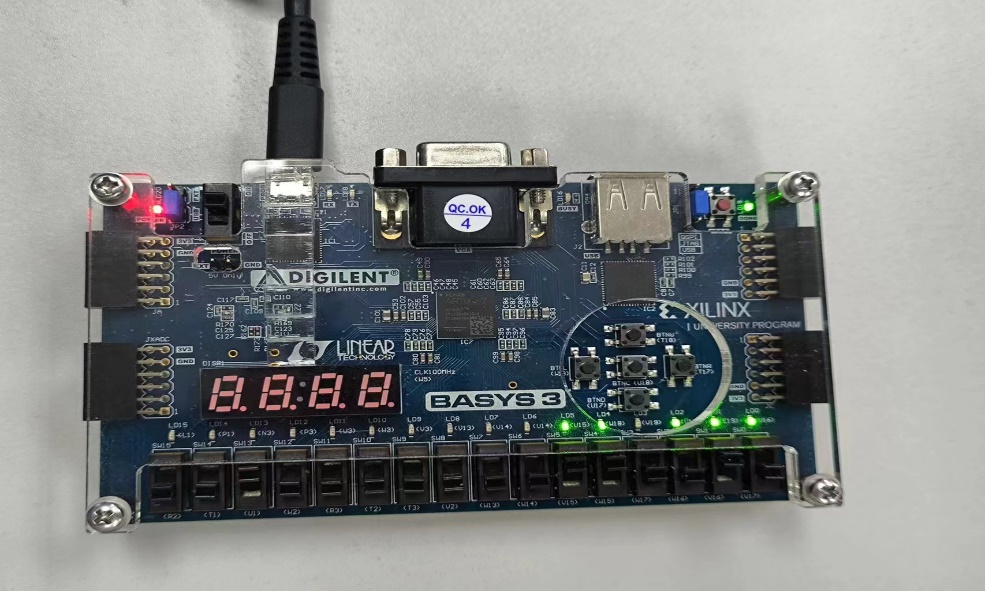
（6）实现设计，执行布局布线，将综合后的网表映射到实际 FPGA 资源上。实现阶段包括映射、放置和路由，生成比特流，配置 FPGA 并验证功能。

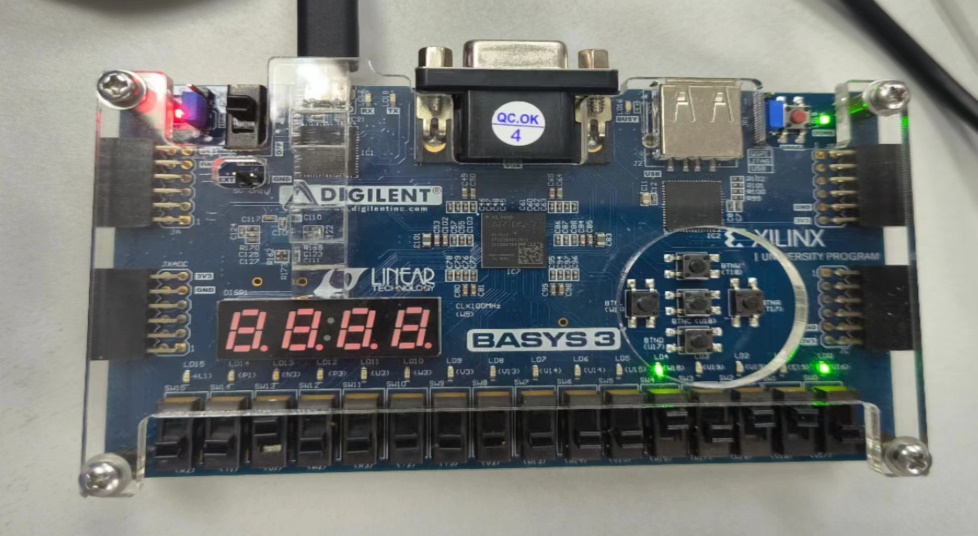
**五、实验结果及总结**

（1）电路图如下：

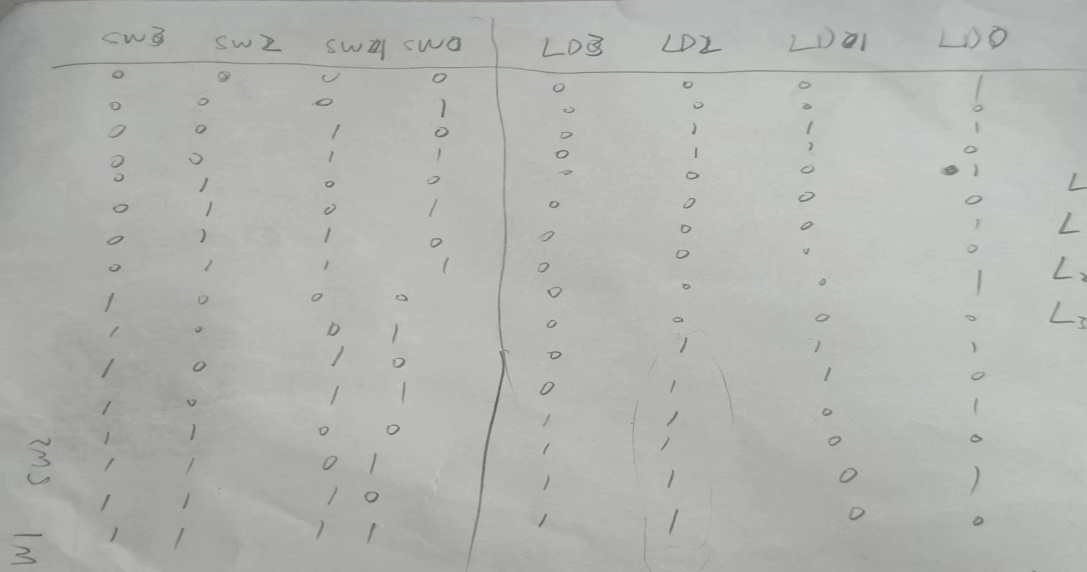
 （2）仿真波形图如下

（3）**实验板测试结果照片：**

sw0,sw2,sw3关闭，sw1,sw4,sw5,sw6打开。由电路图可知，LD0=sw0，LD1=sw1&(~sw2),LD2=sw1&(~sw2)|(~sw2)&sw3,LD3=sw3&(~sw2)

灯LD0,LD1,LD2,LD4,LD5亮起。

Sw0,sw3关闭，sw1,sw2,sw4,打开。由电路图和真值表可知，灯LD0,LD4亮起

（4）真值表

# Buildling Basic Elements for IPl

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7 楼 | 138 房； | **实验台号：** | 143 |
| **实验日期与时间：** | 2024/6/1 | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** | 吴秋霞 |
| **电子文档存放位置：** |  | | | |
| **电子文档文件名：** |  | | | |
| **批改意见：** |  | | | |

**实验2-Buildling Basic Elements for IPl**

1. **实验目的**

（1）使用 Vivado 的 Create and Package IP 功能创建 IP

（2）仿真和验证 IP 功能

（3）生成比特流并验证硬件中的功能

**二、实验原理**

本实验需要学习创建基本数字设计课程中常用的基本元素，帮助掌握使用 Xilinx Vivado 软件创建和验证自定义 IP 核的过程。

预习内容:

1. **Vivado 软件基础**：

熟悉 Vivado 软件的基本界面和操作方法，包括如何创建新项目和管理现有项目。了解 Vivado 中的 Create and Package IP 功能，该功能用于创建自定义 IP 核。

1. **IP 核创建**：

学习如何使用 Vivado 创建自定义 IP 核。了解 IP 核的基本组成部分，包括接口定义、功能实现和验证。

1. **仿真工具使用**：

了解如何使用 Vivado 的仿真工具对设计进行仿真，验证自定义 IP 的功能。学习如何设置仿真环境、运行仿真和分析仿真结果。

1. **约束文件的使用**：

熟悉 Xilinx 设计约束（XDC）文件的作用，学习如何编写和应用约束文件来定义引脚映射和时序约束。

**三、实验环境**

（1）硬件环境需求：xc7a35tcpg236-1(Basys3)

（2）软件环境需求：vivado 2014.4

**四、实验内容**

（1）创建 Vivado 项目和自定义 IP

1.打开 Vivado，创建新项目，选择 "RTL Project"

2.添加源文件和约束文件，设置目标硬件设备

（2）创建第一个自定义 IP

1.在 Vivado 中选择 "Tools" -> "Create and Package New IP"

2.选择 "Create a new AXI4 peripheral" 或其他类型

3.填写信息，完成 IP 核创建，添加 HDL 代码和文件，配置接口

（3）打包 IP并创建第二个自定义 IP

1.在 "IP Packager" 中打包第一个 IP

2再次创建第二个自定义 IP，重复类似步骤，确保功能和接口配置正确

（4）仿真和验证 IP 功能

1.创建新项目，添加自定义 IP

2.配置仿真环境，添加 Testbench 文件

3.使用 Vivado 仿真工具进行功能仿真，检查结果

（5）测试并运行仿真

1.编写测试用例覆盖主要功能

2.运行测试用例，分析仿真结果，调整设计

（6）生成比特流

1.确保设计文件和约束文件正确配置

2.运行综合（Synthesis）和实现（Implementation）

3.生成比特流文件，确保无错误和关键警告

（7）硬件验证

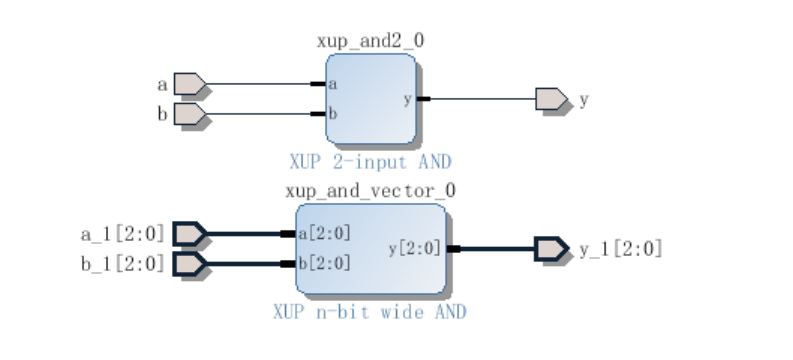
1.将比特流文件下载到目标 FPGA

2.连接 FPGA 开发板，使用 Vivado 配置

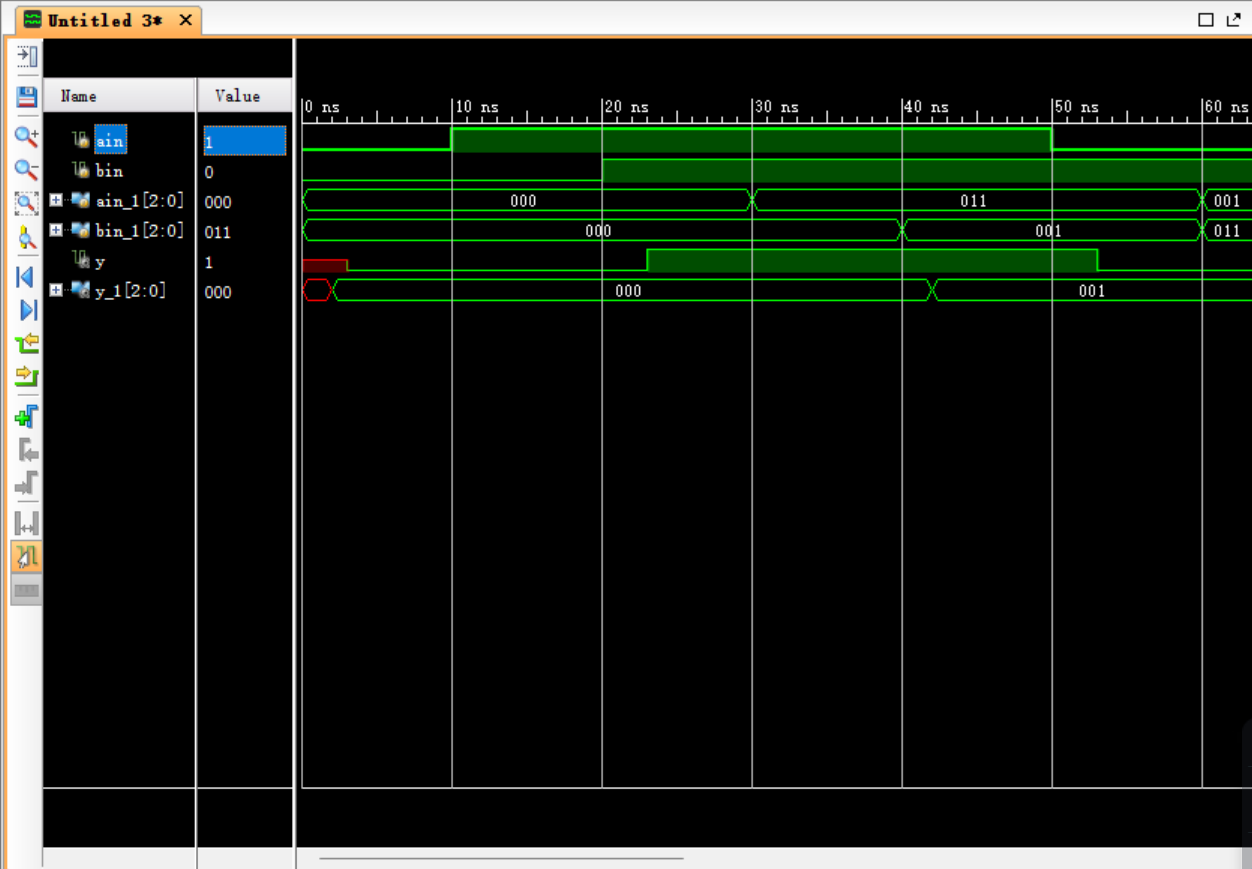
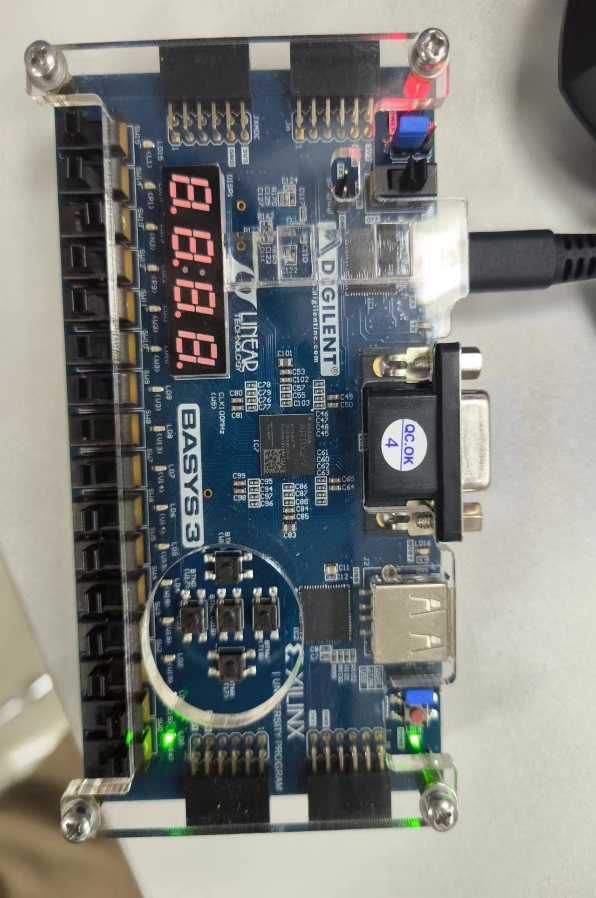
3.在硬件上运行测试用例，验证功能，调整设计并重新验证

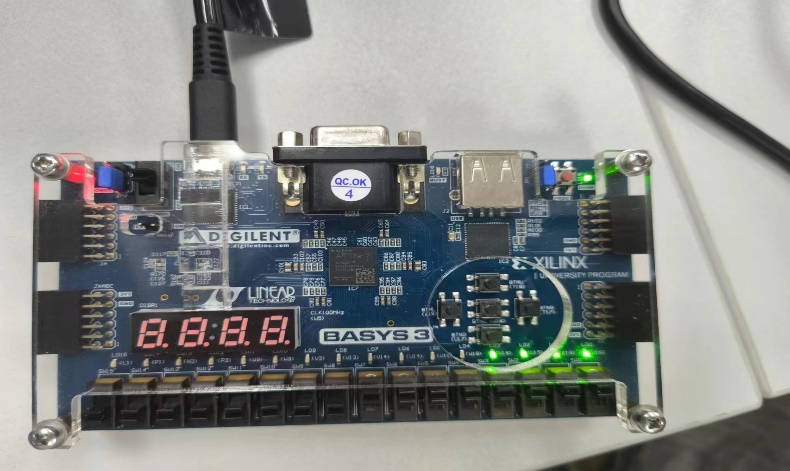
**五、实验结果及总结**

（1）电路图如下：



（2）仿真波形图如下：

（3）实验板测试结果如下；

由电路图可知，Ld1=sw0&sw1,因此sw0,sw1打开时Ld0亮起

由电路图可知，对两个三位输入，由有一个三位输出Ld1,Ld2,Ld3对应，sw2~sw8打开时，Ld1~Ld3亮起

# Create a Simple Digital Circuit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7 楼 138 | 房； | **实验台号：** | 143 |
| **实验日期与时间：** | 2024/6/1 | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** | 吴秋霞 |
| **,** |  | | | |
| **电子文档文件名：** |  | | | |
| **批改意见：** |  | | | |

**实验3-** **Create a Simple Digital Circuit**

1. **实验目的**

1.创建一个 Vivado 项目，并针对位于 Basys3 和 Nexys4 DDR 板上的特定 FPGA 器件导入 HDL 模型。

2.使用提供的部分完成的 Xilinx 设计约束（XDC）文件来约束某些引脚位置。

3.使用 Vivado 的 Tcl 脚本功能添加其他约束。

4.使用 XSim 仿真器仿真设计。

5.综合并实施设计。

6.生成比特流。

7.使用生成的比特流配置 FPGA 并验证其功能。

**二、实验原理或预习内容**

本实验通过使用 Xilinx Vivado 软件，完成从 Verilog HDL 创建简单的数字电路设计到在硬件上验证功能的全过程。实验涵盖了创建模型、编写约束文件、项目管理、仿真、综合、实现和硬件配置等步骤。理解这些步骤及其相互关系，是掌握 FPGA 设计和验证流程的基础。

**预习内容：**

1. HDL 设计基础：

熟悉 Verilog HDL 的基本语法和结构，掌握如何编写简单的 Verilog 模块。

1. 综合与实现：

学习综合和实现的基本概念和流程，了解如何分析综合和实现报告，确保设计满足资源和时序要求。

3. 约束文件：

熟悉 Xilinx 设计约束（XDC）文件的基本语法和作用，学习如何编写和应用 XDC 文件来约束引脚位置和时序。

**三、实验环境**

（1）硬件环境需求：xc7a35tcpg236-1(Basys3)

（2）软件环境需求：vivade哦2014.4

**四、实验内容**

1.创建 Vivado 项目

2.创建 HDL 模型

(1) 创建 Verilog 文件：编写简单的 Verilog 模块，实现所需的数字功能。

(2)验证 HDL 代码：确保代码无语法错误，并符合设计规范。

3.创建用户约束文件

(1)编写 XDC 文件：根据设计要求，编写约束文件，定义引脚位置和时序约束。

(2)导入 XDC 文件：将编写好的 XDC 文件导入 Vivado 项目中。

4.仿真设计

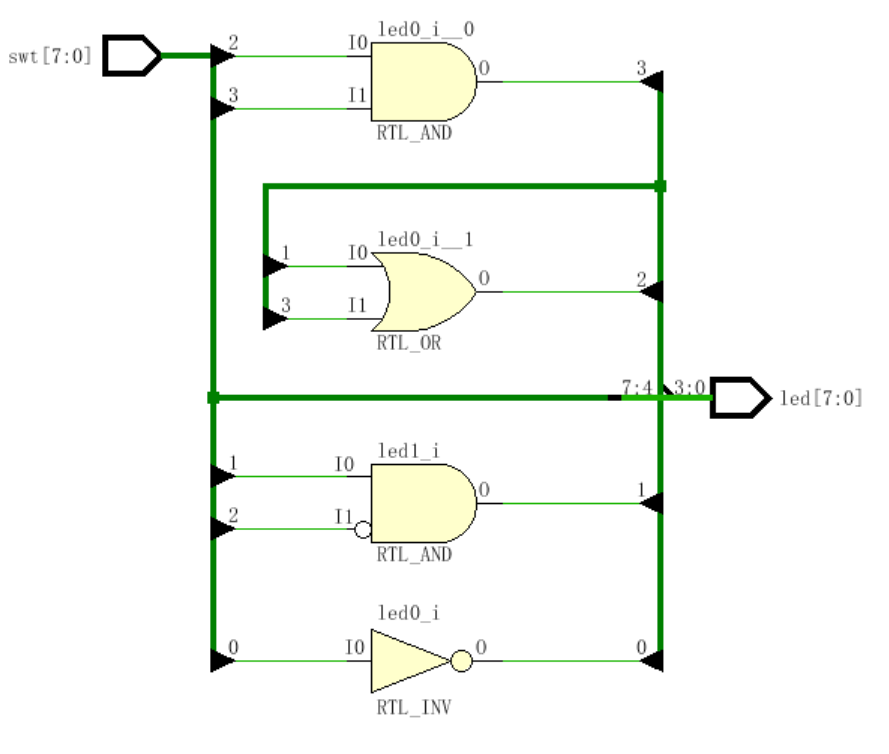
5.综合设计

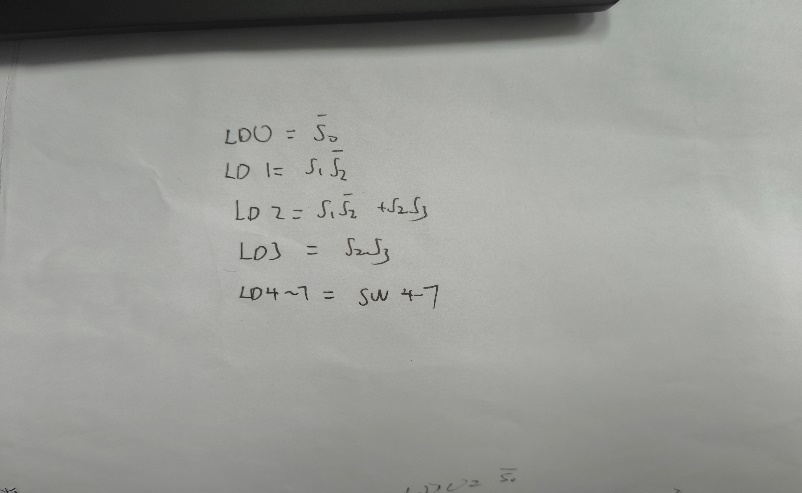
(1)综合 HDL 代码：在 Vivado 中综合设计，生成门级网表。

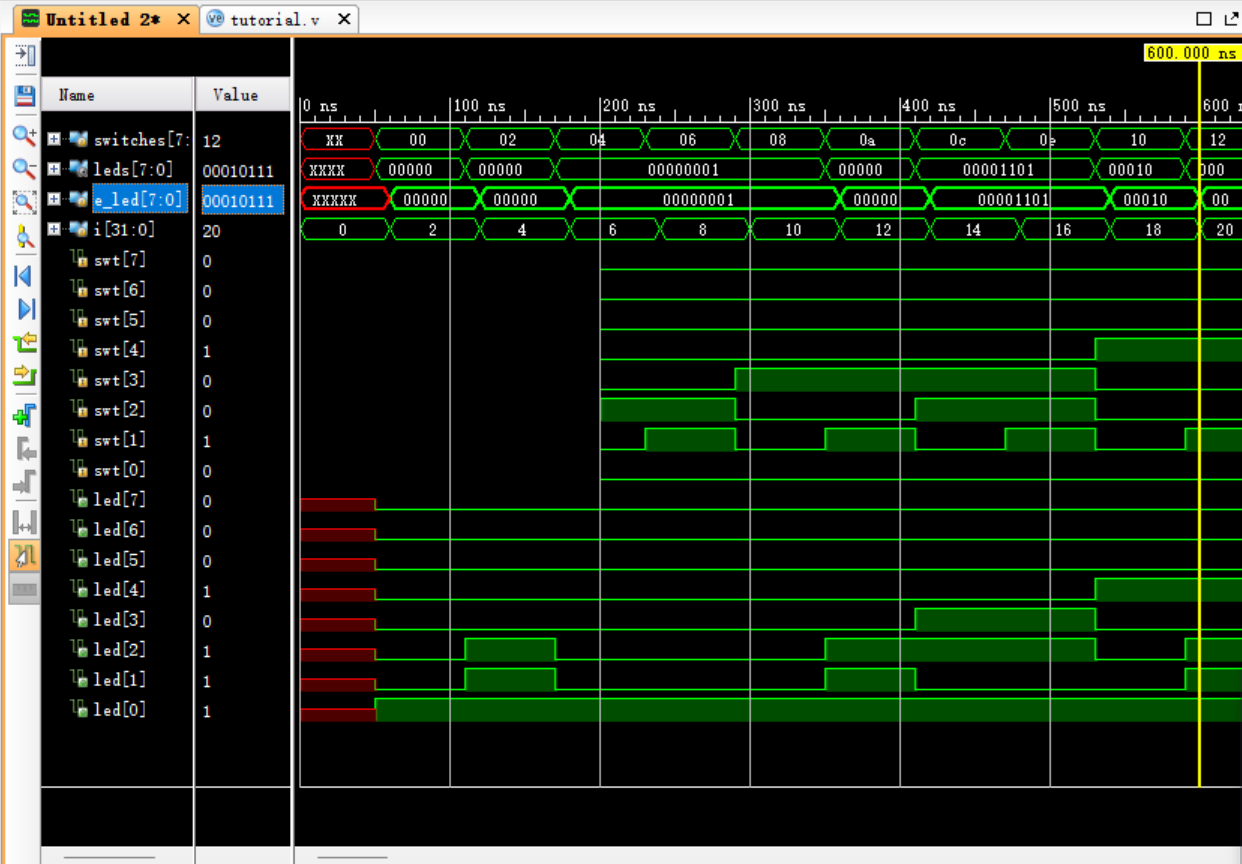
(2)检查综合报告：分析综合报告，确保设计资源和时序满足要求。

6.生成比特流并验证硬件功能

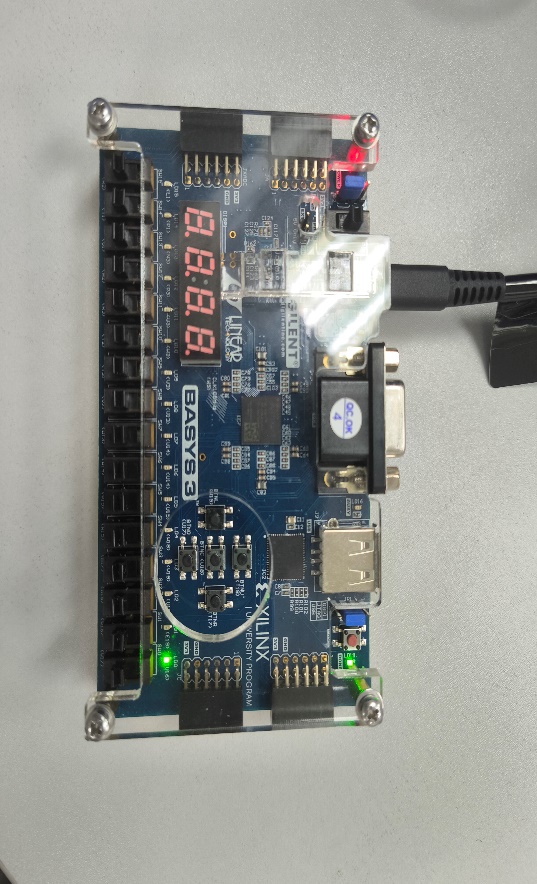
**五、实验结果及总结**

（1）电路图：

（2）表达式：

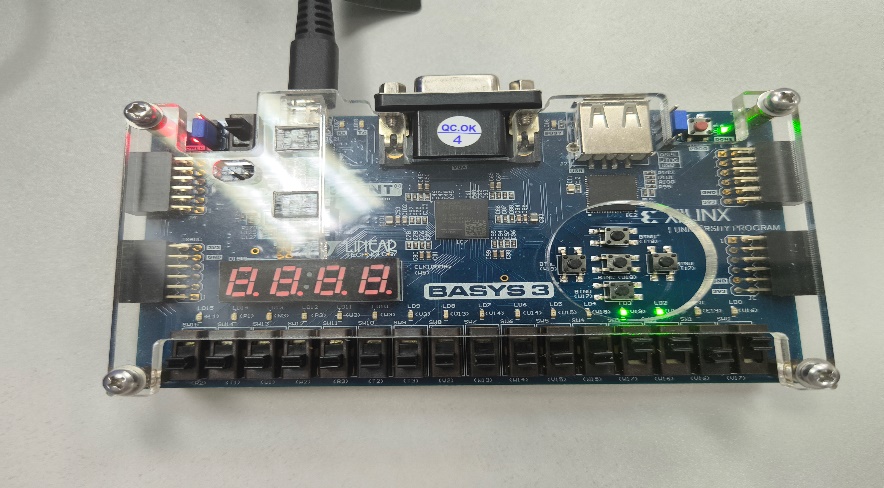
 （3）仿真波形图：

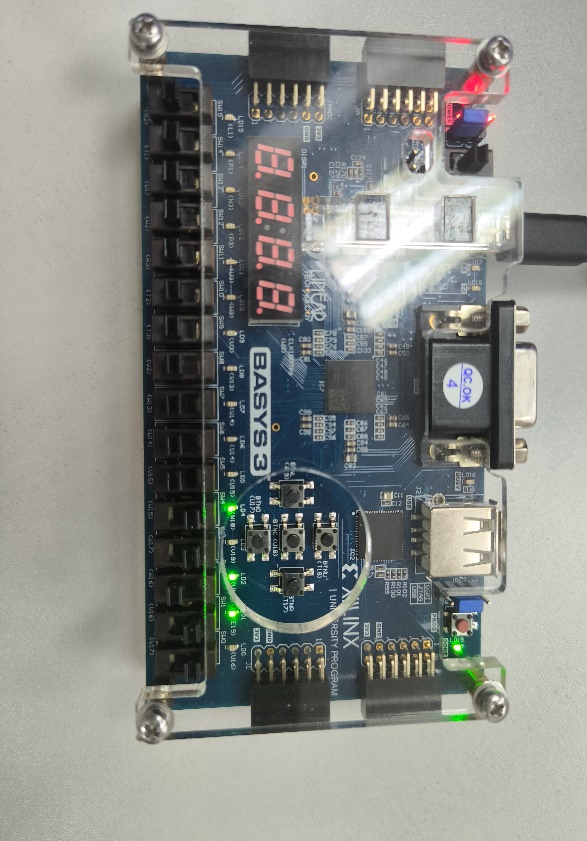
（4）实验板测试结果：

根据所示的原理图及布尔表达式，可对下图进行解释

下图为最小位输入为0时取反得到1

下图输入为00001101,由布尔表达式可得，输出应为00000001

下图输入为00011011，则输出应为00001100

下图输入为00011011，得到输出为00010110

# Finite State Machines.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7 楼 | 138 房； | **实验台号：** | 143 |
| **实验日期与时间：** | 2024/6/1 | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** | 吴秋霞 |
| **电子文档存放位置：** |  | | | |
| **电子文档文件名：** |  | | | |
| **批改意见：** |  | | | |

**实验4-** **Moore FSM（1-1）**

1. **实验目的**

本实验的目的是设计一个序列检测器，使用 Mealy 状态机，通过三块 always 块实现。该状态机有一个输入 (ain) 和一个输出 (yout)。当且仅当接收到的 1 的总数能被 3 整除时，输出 yout 为 1（包括 0）。通过开发测试平台并进行行为仿真来验证模型。在 Basys3 开发板上生成比特流文件并验证设计功能。使用 SW15 作为时钟输入，SW0 作为 ain 输入，BTNU 按钮作为复位输入电路，LED7:LED4 显示 1 的计数，LED0 作为 yout 输出。

**二．实验原理**

Mealy 状态机是一种有限状态机，其输出不仅取决于当前状态，还取决于当前输入。在本实验中，我们将设计一个 Mealy 状态机来检测输入序列中的 1 的个数，并在该个数能被 3 整除时输出 1。状态机由三块 always 块实现：状态寄存器块、下一个状态逻辑块和输出逻辑块。

设计的主要部分包括：

1. 状态寄存器：存储当前状态，并在每个时钟周期更新。

2. 下一个状态逻辑：根据当前状态和输入计算下一个状态。

3. 输出逻辑：根据当前状态和输入计算输出。

预习内容：

1. Mealy 状态机的基础知识：

2. Verilog 编程基础

**三、实验环境**

（1）硬件环境需求：xc7a35tcpg236-1(Basys3)

（2）软件环境需求：vivado 2014.4

**四．实验内容**

1.设计 Mealy 状态机

（1）编写 Verilog 代码：

实现 Mealy 状态机，包括状态寄存器、下一个状态逻辑和输出逻辑。

2.编写测试平台代码并进行仿真

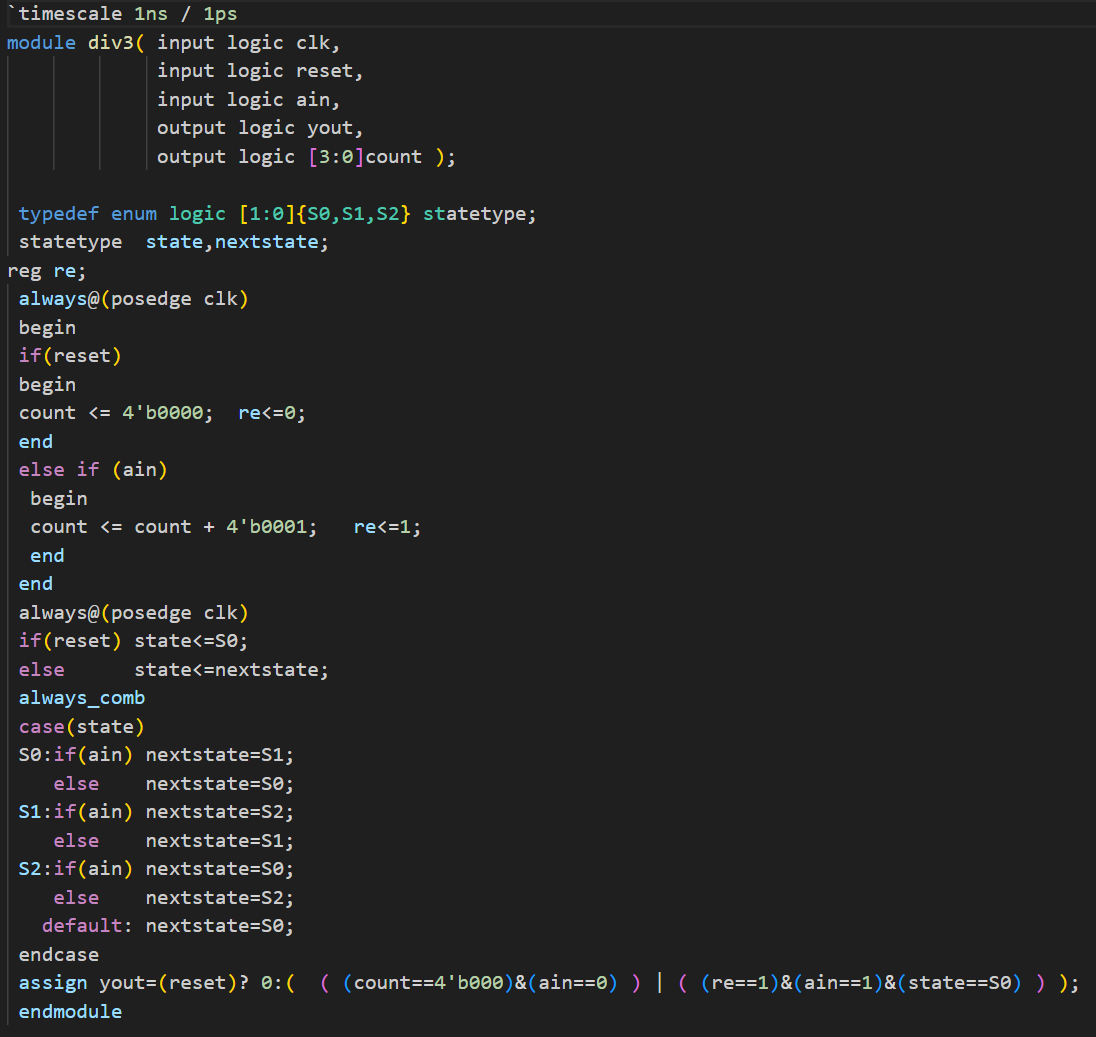
3.综合设计

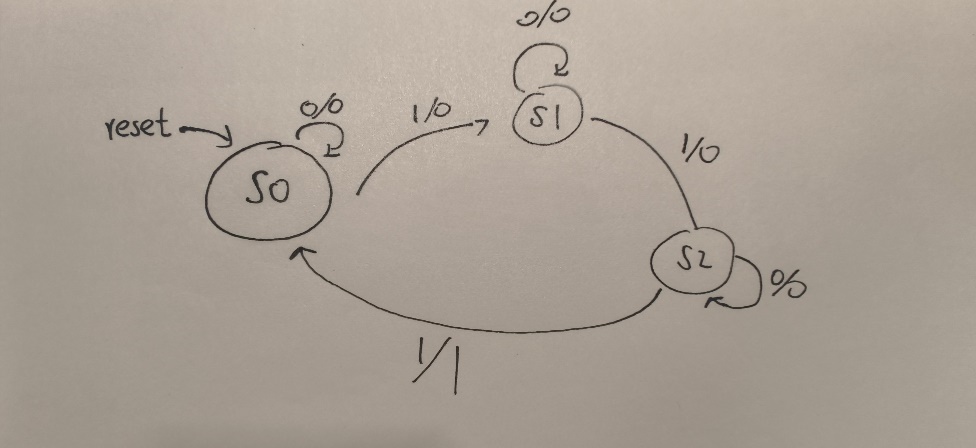
4.实现设计

5.生成比特流并验证硬件功能

**五、实验结果及总结**

（1）实验设计

状态机代码：

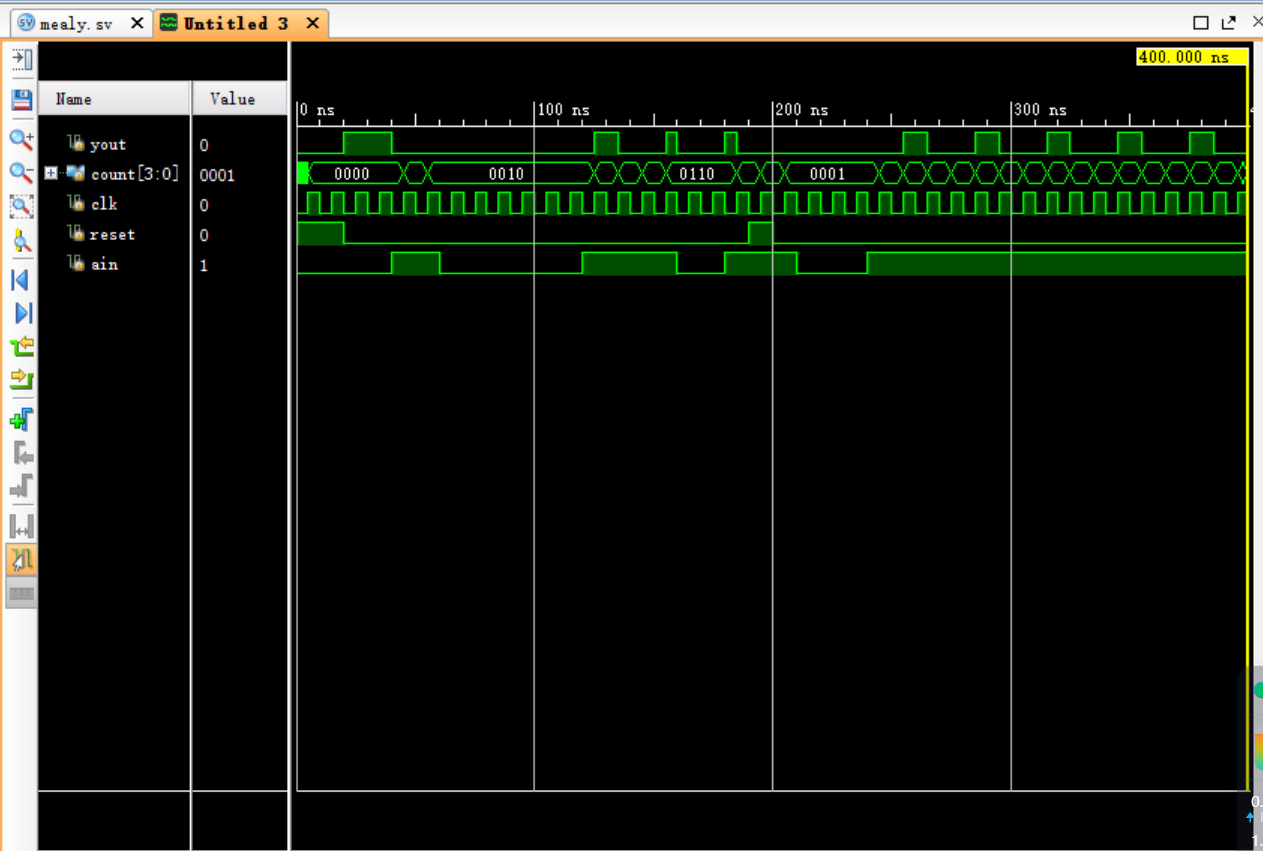
 （2）状态转移图

由图，状态机接受一位输入，输出一位输出。

在状态S0时，若输入为1，则进入下一个状态，表示数字1，并输出0；反之仍在S0，输出为0；

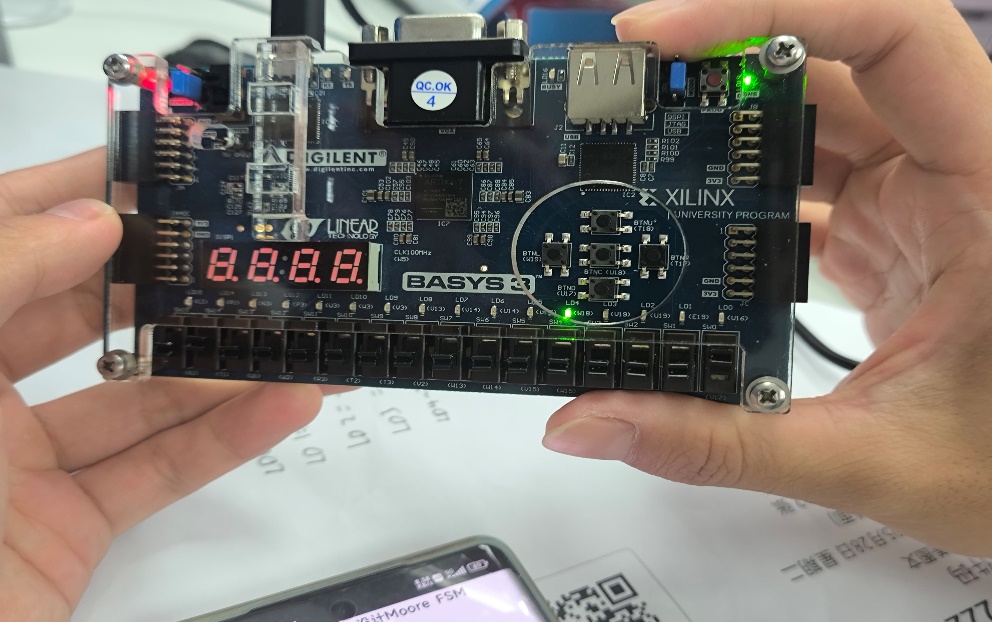
在状态S1时，若输入为1，则进入下一个状态，表示数字2，并输出0；反之停留在本状态，等待输入1.输出0；

在状态S2时，若输入为1，则回到初始状态S0，表示数字3，此时可以输出1，表示找到了可以被三整除的数字；反之停留在本状态，等待输入1，此时输出0；

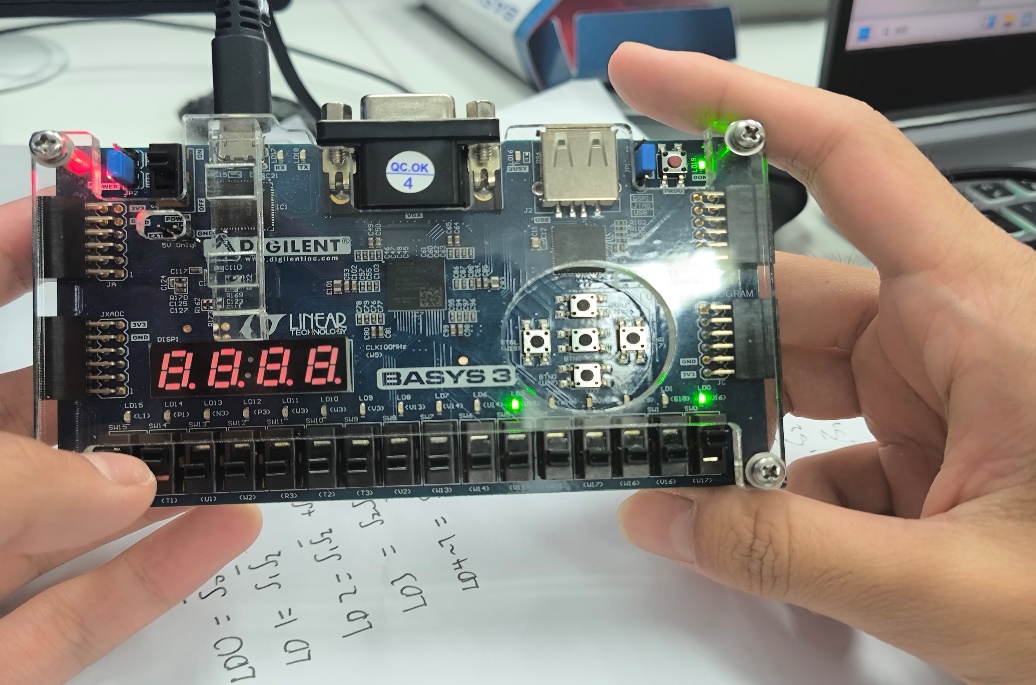
（3）仿真波形图：

（4）实验板测试结果：

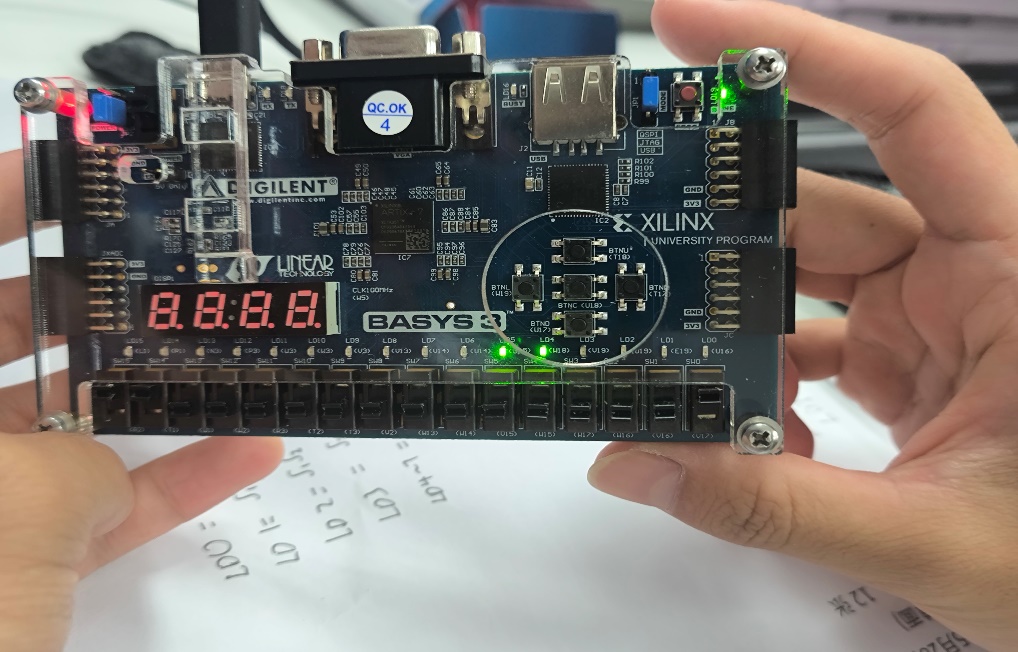
该图：L7-L4对应的二进制编码是0001，对应了计数器的第二种状态，这时候接受了两个1的输入，不能被3整除，故LD0输出为0；



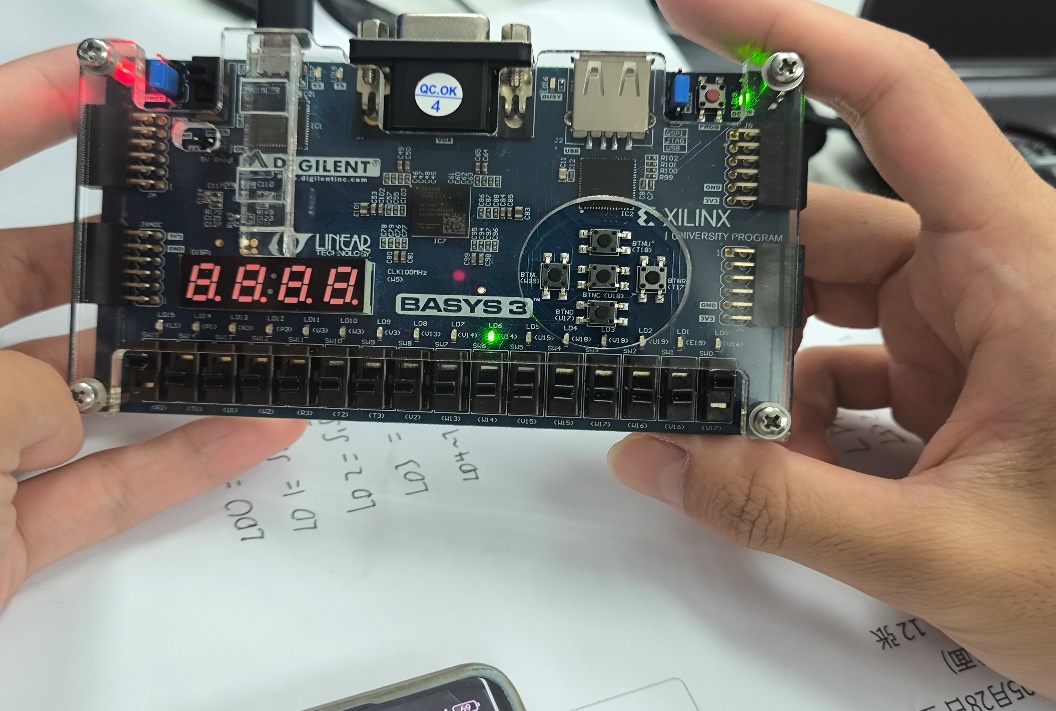
该图：L7-L4对应的二进制编码是0010，对应了计数器的第三种状态，这时候接受了三个1的输入，可以被3整除，故LD0输出为1；



该图：L7-L4对应的二进制编码是0011，对应了计数器的第四种状态，这时候接受了四个1的输入，不可以被3整除，故LD0输出为0；



该图：L7-L4对应的二进制编码是0100，对应了计数器的第五种状态，这时候接受了五个1的输入，不可以被3整除，故LD0输出为0；



**实验4-** **Mealy FSM（2-1）**

1. **实验目的**

设计一个序列检测器，使用 Moore 状态机，通过三个 always 块实现。该状态机有两个输入 (ain[1:0]) 和一个输出 (yout)。输出 yout 初始为 0，只有在以下输入序列发生时才会改变：

1. 输入序列 ain[1:0] = 01, 00 使输出变为 0

2. 输入序列 ain[1:0] = 11, 00 使输出变为 1

3. 输入序列 ain[1:0] = 10, 00 使输出切换状态

通过开发测试平台并进行行为仿真来验证模型。在 Basys3 开发板上生成比特流文件并验证设计功能。使用 SW15 作为时钟输入，SW1-SW0 作为 ain[1:0] 输入，BTNU 按钮作为复位输入电路，LED0 作为 yout 输出。

**二．实验原理**

Moore 状态机的输出仅取决于当前状态。我们将使用三个 always 块分别实现状态寄存器、下一个状态逻辑和输出逻辑。

**三、实验环境**

（1）硬件环境需求：xc7a35tcpg236-1(Basys3)

（2）软件环境需求：vivado 2014.4

**四．实验内容**

1.创建 Vivado 项目

2. 编写 Verilog 代码

3. 创建测试平台并进行仿真

(1)创建 Testbench 文件，用于仿真验证 Verilog 代码。

(2)配置仿真环境，使用 Vivado 的仿真工具（如 XSIM）运行仿真。

(3)检查仿真结果，确保设计功能正确。

4. 综合设计

(1)在 Vivado 中运行综合（Synthesis）过程。

(2)检查综合报告，确保设计无错误和关键警告。

5. 实现设计

(1)在 Vivado 中运行实现（Implementation）过程。

(2)完成布线和时序分析，确保设计满足性能要求。

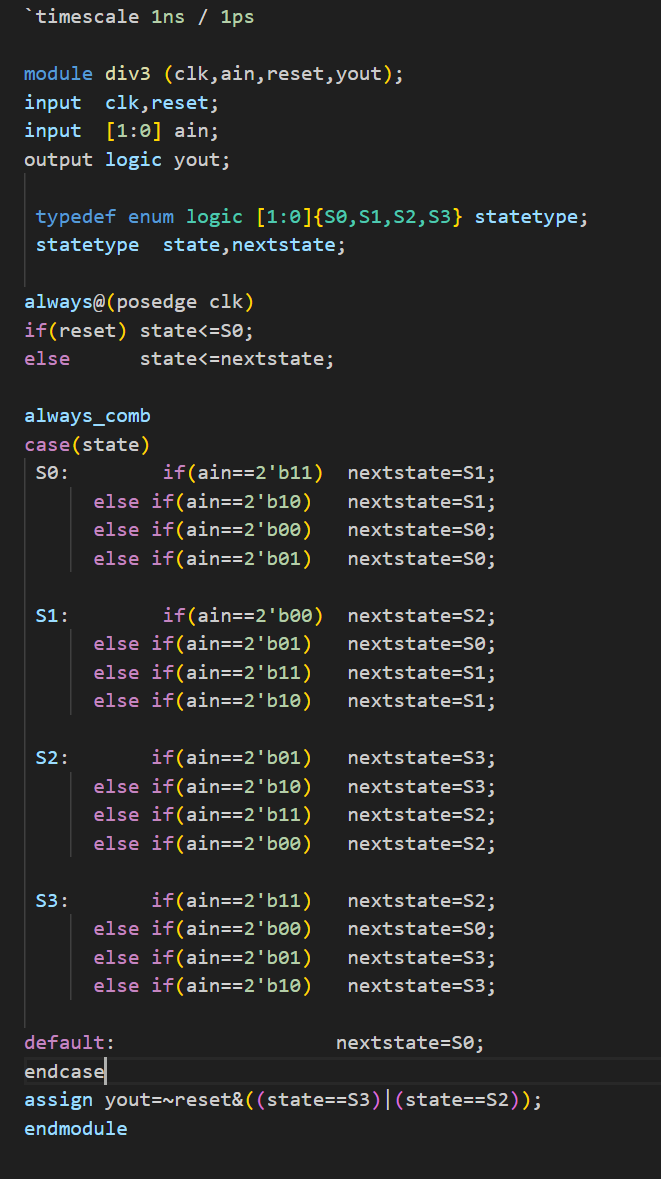
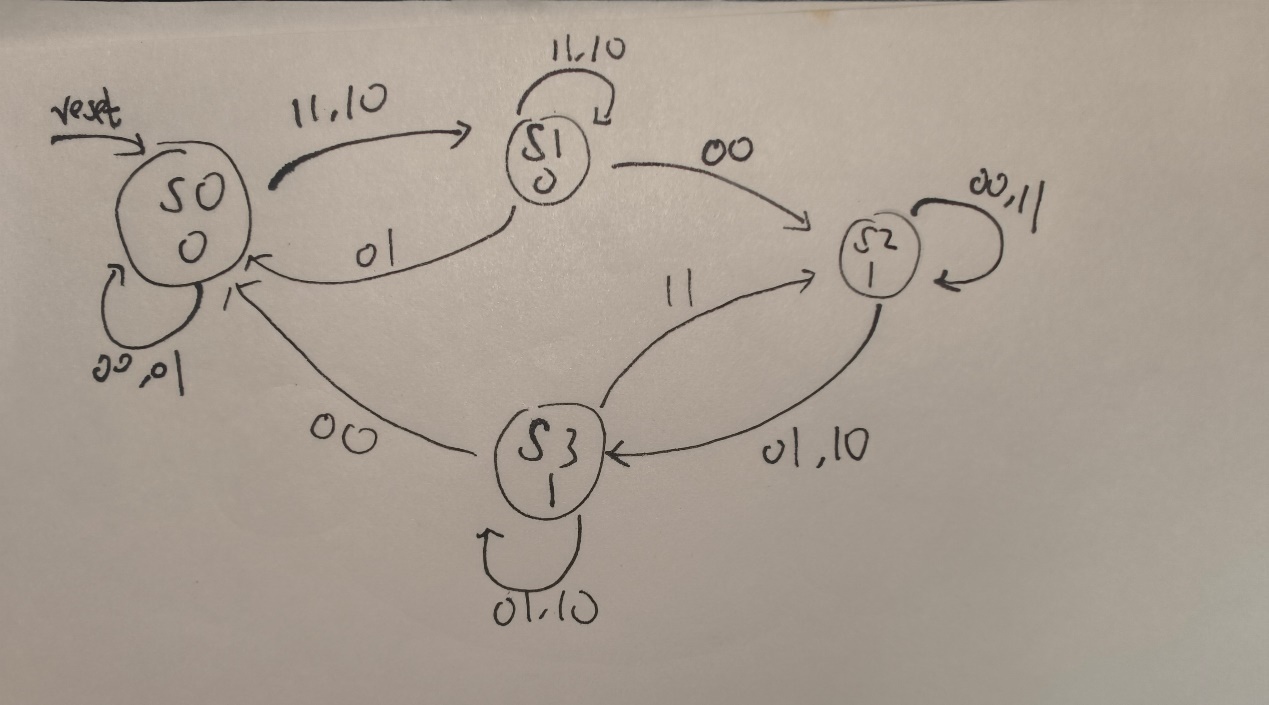
6. 生成比特流

7. 硬件验证

**五、实验结果及总结**

（1）实验设计

状态机代码：

（2）状态转移图

 **S0 (初始状态)**:

如果输入是 00 或 01，继续保持 S0，因为这两种输入不会改变输出。

如果输入是 11 或 10，进入中间状态 S1，因为这些输入可能是我们感兴趣的序列的第一部分。

 **S1 (中间状态)**:

如果输入是 00，意味着完成了两个感兴趣的序列之一 (11, 00 或 10, 00)，所以进入 S2 或 S3 以更新输出。

如果输入是 11 或 10，保持在 S1 继续等待第二个输入 00。

 **S2 (输出 1 的状态)**:

如果输入是 00 或 11，保持在 S2，因为 00 完成了序列，而 11 可能是新序列的开始。

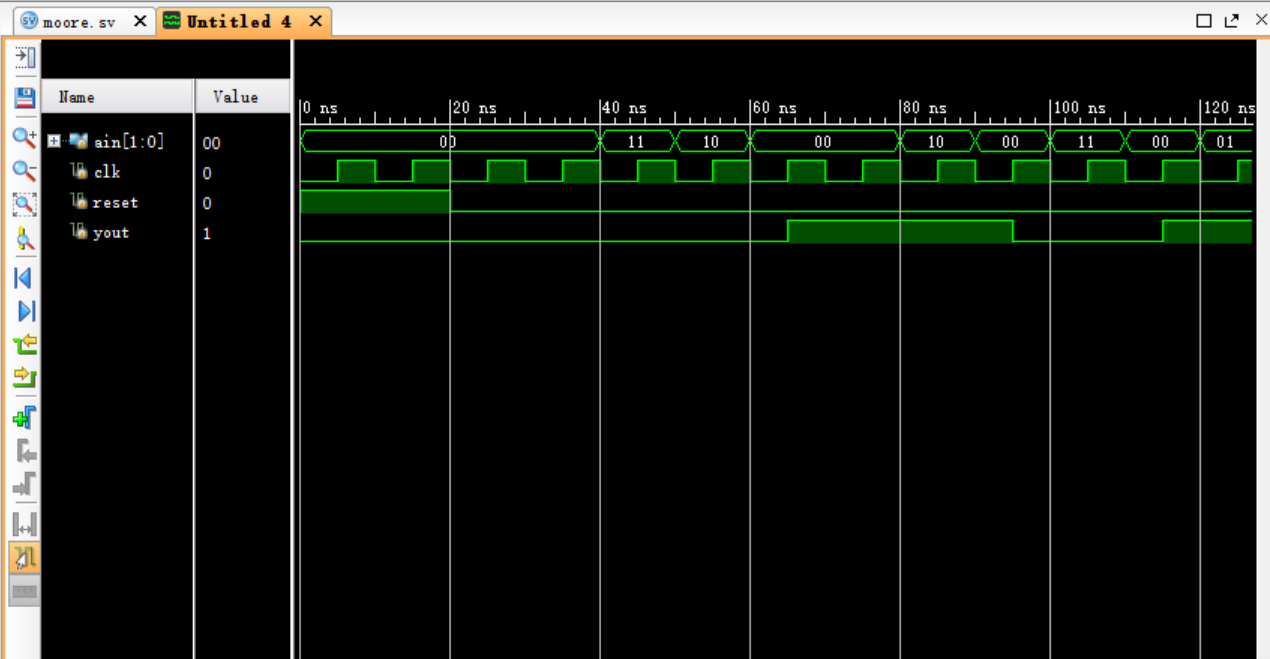
如果输入是 01 或 10，进入 S3，因为这些输入符合序列 10, 00，需要切换输出。

 **S3 (输出切换状态)**:

如果输入是 01 或 10，保持在 S3。

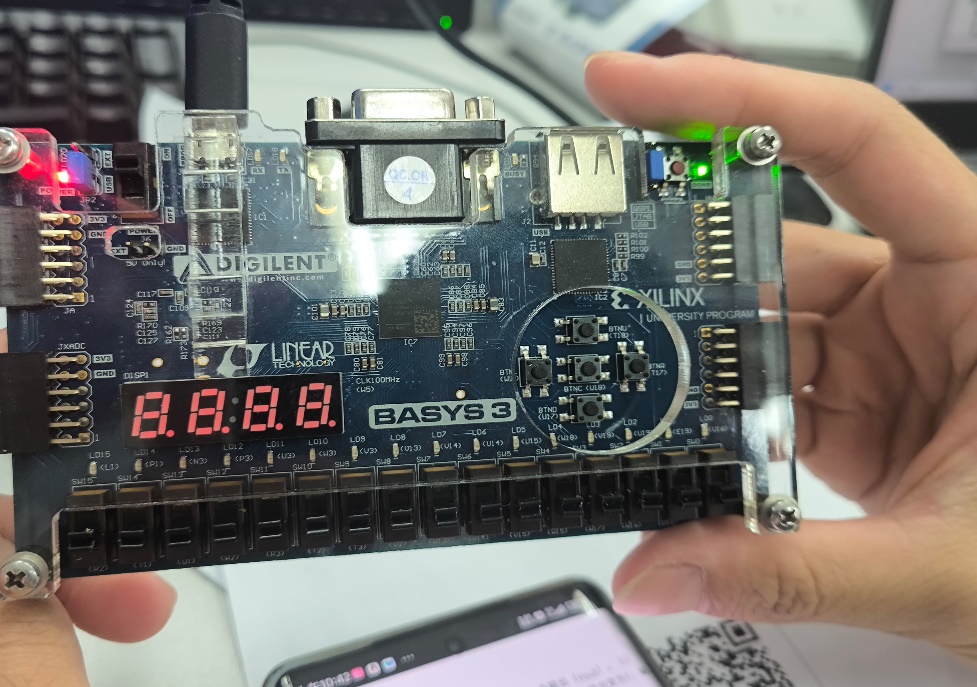
如果输入是 11，开始新的序列，回到 S2。

如果输入是 00，完成了序列，回到初始状态 S0。

 （3）仿真波形图：

（4）实验板测试结果照片

1、程序开始执行，置位为S0，输出为0，LD0无变化



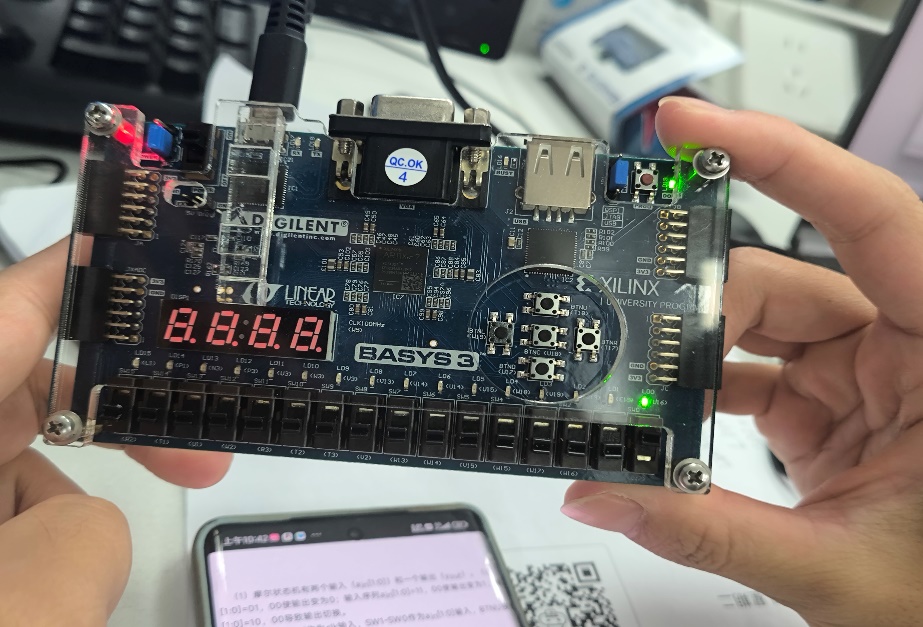
2. 输入变为11，当时钟SW15上升时候，接收输入，状态变化到S1，此时输出为0，LD0灯不亮



3. 将输入调整为00，当时钟SW15上升的时候，接收输入，状态变化到S2，对应的输出为1，LD0灯变亮



4.将输入调整为01，当时钟SW15上升的时候，接收输入，状态变化到S3，对应的输出为1，LD0灯保持亮的状态



5将输入调整为00，当时钟SW15上升的时候，接收输入，状态变化到S0，对应的输出为0，LD0灯熄灭

