《可编程逻辑器件应用》教学大纲

适用专业：电子信息工程技术专业

教学周数：18

学　分：3

学时：54 学时

说 明

一、课程设计的性质、目的和任务：

课程性质：《可编程逻辑器件应用》是电子信息工程技术专业的核心课程，兼具理论性与极强的实践性、应用性。本课程以 FPGA（现场可编程门阵列）为核心载体，融合数字电子技术、硬件描述语言（Verilog HDL）、FPGA 开发工具应用等核心内容，旨在通过“理论讲授+演示操作+实训实操”相结合的方式，培养学生 FPGA 软硬件设计、仿真验证、板级调试的综合能力，为学生从事嵌入式、数字系统开发等相关岗位奠定坚实基础。

课程目的：通过本课程的学习，学生将能够系统掌握 FPGA 的基本原理、Quartus II、ModelSim-Altera 等开发工具的使用方法，熟练运用 Verilog HDL 进行数字逻辑电路设计，完成从电路设计、代码编写、仿真验证到板级下载调试的全流程操作。同时，课程注重培养学生的逻辑思维、创新能力、问题解决能力和团队协作精神，引导学生树立严谨的工程实践理念。

课程任务：使学生掌握 FPGA 概述、逻辑设计基础、Verilog HDL 语法规范、FPGA 开发工具配置等核心知识；具备时钟分频、蜂鸣器 PWM 驱动、流水灯、3-8 译码器等典型数字系统的 FPGA 设计、仿真与调试能力；熟悉实验报告的撰写规范，能够独立完成实训任务和综合考查任务。此外，课程还将融入思政元素，培养学生的社会责任感、职业道德和对国家数字电子技术发展的认同感与自豪感。

课程思政融入

思政目标：在传授 FPGA 相关专业知识和实践技能的同时，培养学生的爱国情怀、工匠精神和创新意识，激发学生为国家数字电子技术进步、集成电路产业发展贡献力量的使命感，引导学生树立“理论联系实际、严谨务实、精益求精”的学习和工作态度。

思政内容：通过案例分析、项目实践等方式，展示我国在 FPGA、集成电路领域的发

展成就和技术突破，引导学生认识到核心技术自主可控的重要性；结合实训过程中的难点突破、代码调试中的反复打磨，培养学生坚韧不拔、精益求精的工匠精神；通过小组协作完成实训任务，培养学生的团队协作意识和责任担当；在实验报告撰写、代码规范编写中，培养学生严谨细致、诚信务实的职业素养。

思政实践：在课程实训和综合考查环节中，设置贴近实际工程应用的设计任务，让学生在实践中体会数字电子技术在智能家居、工业控制、智能终端等领域的应用价值，感受科技服务社会的意义；鼓励学生主动探索、大胆创新，对典型设计项目进行优化改进，培养创新思维；在实训过程中，引导学生互相帮助、分工协作，共同解决实践中遇到的问题，提升团队协作能力。

二、课程内容和学时分配

根据教学计划规定的学时数，本课程共计 54 学时，分 18 周完成，每周 3 学时。

课程内容和学时分配表

章数	内　容	课时
1	课程介绍与 FPGA 概述	3
2	逻辑设计基础	6
3	软件安装与配置(Quartus II 和 ModelSim-Altera 等)	3
4	Verilog 语法	18
5	实验一：时钟二分频 FPGA 设计和仿真验证	3
6	实验二：FPGA 蜂鸣器 PWM 发声设计和板级验证	6
7	实验三：FPGA 流水灯应用设计、仿真及板级验证	6
8	实验四：FPGA3-8 译码器应用设计、仿真及板级验证	6
9	综合考查	3
合 计		54

三、教学建议

原则上教师应遵照本教学大纲的要求，结合教学日历的安排，完成全部教学内容和实训任务。在教学过程中，可根据学生的实际学习情况、实训设备条件，对教学内容的讲解顺序、学时分配、实训任务的难度进行灵活调整，重点突出实践技能的培养，确保理论与实践紧密结合。

教学内容

第 1 章 课程介绍与 FPGA 概述

[教学目的和要求]

1. 使学生了解本课程的教学安排、考核方式、实训要求及学习重点。

2. 掌握 FPGA 的基本定义、特点及应用场景，理解 FPGA 与 ASIC、ARM、DSP 的区别。

3. 了解 FPGA 的主流厂商、产品系列及 FPGA 开发的基本流程。

4. 激发学生对 FPGA 技术的学习兴趣，明确课程学习的意义和目标。

[教学内容]

1. 课程介绍：教学日历解读、学时分配、教学方式、考核要求、实训任务安排、学习方法指导。

2. FPGA 概述：FPGA 的定义、核心特点（可编程性、并行性、高集成度等）、应用领域（工业控制、智能终端、通信等）。

3. FPGA 与其他芯片的区别：对比 FPGA 与 ASIC、ARM、DSP 在设计流程、灵活性、成本、功耗等方面的差异。

4. FPGA 主流厂商及产品：Altera（Intel）、Xilinx（AMD）等厂商介绍，典型 FPGA 芯片系列讲解。

5. FPGA 开发基本流程：需求分析、电路设计、代码编写、仿真验证、板级调试、下载固化的完整流程概述。

[教学建议]

结合 FPGA 应用案例演示，提问互动调动学生积极性，简要介绍后续实训项目，帮助学生建立课程整体认知。

[作业]

1. 查阅资料，介绍 FPGA 最新发展趋势及典型应用案例。

2. 熟悉本课程教学日历和大纲，明确各阶段学习重点与实训任务。

第 2 章 逻辑设计基础（一）

[教学目的和要求]

1. 掌握逻辑代数的基本运算、基本定律和常用公式。

2. 学会运用逻辑代数进行逻辑函数的分式化简方法。

3. 掌握卡诺图的绘制方法，能够运用卡诺图化简逻辑函数（2-4 变量）。

4. 理解逻辑函数化简的意义，为后续 FPGA 逻辑设计奠定理论基础。

[教学内容]

1. 逻辑代数基础：二进制、八进制、十六进制的转换，逻辑变量的定义（0 和 1 的物理意义）。

2. 逻辑代数基本运算：与运算、或运算、非运算，复合逻辑运算（与非、或非、与或非、异或、同或）。

3. 逻辑代数基本定律和常用公式：交换律、结合律、分配律、摩根定律，常用吸收公式、冗余公式及其应用。

4. 逻辑函数的分式化简：化简的目的和标准，公式法化简的步骤和技巧，典型例题讲解。

5. 卡诺图化简：卡诺图的结构（2-4 变量），逻辑函数与卡诺图的对应关系，卡诺图化简的规则（合并相邻项、消去冗余变量），典型例题讲解。

[教学建议]

结合数字电子技术基础循序渐进讲解，多举例演示化简方法，引导学生总结技巧，强化知识点掌握。

[作业]

1. 完成若干道逻辑函数分式化简题（公式法）和卡诺图化简题（2-4 变量）。

2. 整理卡诺图化简的核心规则，简要说明公式法与卡诺图化简的适用场景。

第 3 章 逻辑设计基础（二）

[教学目的和要求]

1. 理解组合逻辑电路和时序逻辑电路的定义、特点及区别。

2. 掌握常见组合逻辑电路（编码器、译码器、数据选择器、加法器）的工作原理和逻辑功能。

3. 掌握 D 触发器的工作原理、特性方程、时序图，理解触发器的触发方式。

4. 了解寄存器、移位寄存器、计数器的基本工作原理，为后续 Verilog 时序逻辑设计奠定基础。

[教学内容]

1. 组合逻辑电路：定义、特点（无记忆性、输出仅由输入决定），组合逻辑电路的分析和设计步骤。

2. 常见组合逻辑电路：编码器（8-3 线编码器）、译码器（3-8 线译码器）、数据选择器（4 选 1、8 选 1）、加法器（半加器、全加器）的工作原理、逻辑表达式、真值表及逻辑图。

3. 时序逻辑电路：定义、特点（有记忆性、输出由输入和现态决定），时序逻辑电路的基本组成（组合电路+存储电路）。

4. D 触发器：基本结构、工作原理，特性方程，触发方式（边沿触发），时序图绘制方法，典型应用场景。

5. 时序逻辑模块简介：寄存器（并行输入并行输出）、移位寄存器（串入串出、串入并出）、计数器（异步计数器、同步计数器）的基本工作原理和逻辑功能。

[教学建议]

结合真值表、时序图讲解电路原理，重点演示 D 触发器时序图绘制，结合后续实训铺垫译码器相关知识。

[作业]

1. 分析 3-8 线译码器逻辑功能，写出真值表和逻辑表达式。

2. 绘制 D 触发器时序图，简要说明组合逻辑与时序逻辑电路的核心区别。

第 4 章 软件安装与配置

[教学目的和要求]

1. 掌握 Quartus II 13.1 软件的安装步骤、界面布局及基本操作。

2. 掌握 ModelSim-Altera 软件的安装方法及与 Quartus II 的关联配置。

3. 掌握 Notepad++的安装及配置，能够用于 Verilog 代码的编写和编辑。

4. 掌握 USB-Blaster 下载线、串口芯片的驱动安装方法，确保下载调试正常。

[教学内容]

1. Quartus II 13.1 安装与配置：软件下载、安装步骤（适配 XP/WIN7/WIN8 系统），许可证配置，界面布局（项目导航栏、编辑区、编译区等），基本操作介绍。

2. ModelSim-Altera 安装与关联：软件安装步骤，与 Quartus II 的关联配置方法，确保

能够调用 ModelSim 进行仿真验证。

3. Notepad++安装与配置：软件安装，Verilog 语法高亮设置，与 Quartus II 的关联，方便代码编写和调用。

4. 驱动安装：USB-Blaster 下载线驱动安装步骤，串口芯片驱动安装方法，驱动安装常见问题及解决方案。

5. 软件测试：创建简单项目，测试软件安装和配置是否正常，确保后续开发工作顺利进行。

[教学建议]

现场演示各软件安装配置步骤，强调注意事项，实训环节巡回指导，及时解决学生安装过程中的问题。

[作业]

1. 完成所有软件及驱动安装，截图留存安装成功界面。

2. 记录安装配置过程中遇到的问题及解决方案，整理成简短文档。

第 5 章 Verilog 语法（一）

[教学目的和要求]

1. 了解 Verilog HDL 的基本特点、应用场景及与 VHDL 的区别。

2. 掌握 Verilog 的时标定义方法，理解时间单位和时间精度的含义。

3. 掌握 Verilog 模块的基本结构（模块声明、端口声明、内部信号声明、功能描述）。

4. 掌握 Verilog 的注释方法、常量和变量类型，能够正确声明端口方向和端口类型。

[教学内容]

1. Verilog HDL 概述：基本特点（简洁、灵活、可综合），应用场景，与 VHDL 的区别（语法风格、适用场景）。

2. 时标定义：`timescale 命令的用法，时间单位（如 ns）和时间精度（如 ps）的定义，时标在仿真中的作用。

3. 模块基本结构：模块声明（module 关键字），端口列表，模块结束（endmodule 关键字），内部信号声明区域，功能描述区域。

4. 注释方法：单行注释（//）和多行注释（/*...*/）的用法，注释的规范和注意事项。

5. 常量类型：整数型、实数型、字符串型，常量的表示方法（十进制、二进制、八进制、十六进制）。

6. 变量类型：reg 型、wire 型的定义和区别，变量的赋值规则，其他常用变量类型（integer、parameter）简介。

7. 端口方向和声明：输入端口（input）、输出端口（output）、双向端口（inout）的声明方法，端口类型的指定（reg 型或 wire 型）。

[教学建议]

结合简单代码实例讲解模块结构和语法，重点区分 reg 型与 wire 型用法，引导学生动手练习模块框架编写。

[作业]

1. 编写简单 Verilog 模块框架，包含输入、输出端口及内部信号声明，添加规范注释。

2. 用不同进制表示同一常量，简要说明 reg 型与 wire 型的核心区别及应用场景。

第 6 章 Verilog 语法（二）

[教学目的和要求]

1. 掌握 Verilog 的赋值语句（assign 连续赋值）的用法，理解连续赋值的特点。

2. 掌握 always 过程块的定义和用法，理解阻塞赋值（=）和非阻塞赋值（<=）的区别及应用场景。

3. 能够正确运用赋值语句和 always 过程块描述简单的逻辑功能。

4. 避免赋值语句使用过程中的常见错误，规范代码编写格式。

[教学内容]

1. assign 连续赋值语句：语法格式，适用场景（描述组合逻辑，赋值对象为 wire 型变量），赋值表达式的写法，典型例子（如与门、或门逻辑描述）。

2. always 过程块：语法格式（always @(敏感信号列表)），敏感信号列表的写法（组合逻辑、时序逻辑的敏感信号区别），always 块的执行机制。

3. 阻塞赋值（=）：赋值规则（立即执行，顺序执行），适用场景（组合逻辑描述），典型例子讲解。

4. 非阻塞赋值（<=）：赋值规则（延迟执行，并行执行），适用场景（时序逻辑描述），与阻塞赋值的对比演示。

5. 常见错误及注意事项：赋值对象与变量类型不匹配，阻塞与非阻塞赋值混用，敏感信号列表遗漏等问题及解决方案。

[教学建议]

多举对比案例，演示两种赋值方式的区别，引导学生动手练习，强调代码编写规范。

[作业]

1. 用 assign 语句描述 3 个基本逻辑门，用 always 块描述与非门和 D 触发器。

2. 简要说明阻塞与非阻塞赋值的执行差异，避免混用的注意事项。

第 7 章 Verilog 语法（三）

[教学目的和要求]

1. 掌握 if-else 条件语句的用法，理解 if-else 与锁存器陷阱的关系，学会避免锁存器生成。

2. 掌握 case 语句（case、casez、casex）的用法，理解 case 语句与 if-else 语句的区别及适用场景。

3. 掌握常用循环语句（for、while、repeat、forever）的用法，明确各循环语句的适用场景。

4. 能够运用条件语句和循环语句描述复杂的逻辑功能。

[教学内容]

1. if-else 条件语句：语法格式（单分支、双分支、多分支），执行流程，典型应用例子（如二选一选择器）。

2. if-else 与锁存器陷阱：锁存器的生成条件，if-else 语句中遗漏 else 分支导致锁存器生成的问题，避免锁存器的方法（确保所有条件下都有赋值）。

3. case 语句：case 语句的语法格式，casez、casex 语句的特点（忽略高阻态、不定态），执行流程，典型应用例子（如四选一选择器）。

4. case 语句与 if-else 语句的区别：逻辑描述效率、可读性、适用场景对比，引导学生根据逻辑需求选择合适的语句。

5. 循环语句：for 循环（适用组合逻辑和仿真代码，可综合）、while 循环（多用于仿真代码）、repeat 循环（多用于仿真代码，固定循环次数）、forever 循环（多用于仿真代码，无限循环）的语法格式和应用例子。

[教学建议]

重点讲解锁存器陷阱的避免方法，结合实例演示条件语句和循环语句用法，明确循环语句的可综合性。

[作业]

1. 分别用 if-else 和 case 语句描述四选一选择器（避免锁存器）。

2. 用 for 循环实现 8 位数据移位操作，简要说明各循环语句的适用场景。

第 8 章 Verilog 语法（四）

[教学目的和要求]

1. 掌握 Verilog 中各类运算符（算术、逻辑、位、关系、条件等）的用法和优先级。

2. 掌握参数化设计的方法（parameter 关键字），理解参数化设计的优势。

3. 了解 task 与 function 的定义和用法，能够运用 task 和 function 实现模块化代码设计。

4. 掌握常用编译指令的用法，规范代码编写和编译过程。

[教学内容]

1. 运算符：算术运算符（+、-、*、/、%）、逻辑运算符（&&、||、!）、位运算符（&、|、~、^、^~）、关系运算符（>、<、>=、<=、==、!=）、条件运算符（?:）的用法、优先级和结合性，典型例子演示。

2. 参数化设计：parameter 关键字的用法，参数的定义和赋值（模块内部赋值、模块实例化时赋值），参数化设计的优势（代码复用、灵活修改），典型例子（参数化计数器）。

3. task 与 function：task（任务）和 function（函数）的定义格式、区别（返回值、输入输出、调用方式），task 用于实现复杂操作（多输入多输出），function 用于实现简单运算（单返回值），典型例子讲解。

4. 编译指令：`define（宏定义）、`include（文件包含）、`ifdef/`else/`endif（条件编译）等常用编译指令的用法，编译指令在代码复用和调试中的应用。

[教学建议]

整理运算符优先级表格，重点讲解参数化设计方法，结合实例演示 task 与 function 的用法，强化模块化设计思路。

[作业]

1. 编写参数化二选一选择器，实例化时设置不同数据宽度。

2. 定义 function 实现 8 位数据加法，定义 task 实现 8 位数据移位清零，编写代码并调用。

第 9 章 Verilog 语法（五）

[教学目的和要求]

1. 掌握生成块 generate 的定义和用法，能够运用生成块实现重复模块的实例化。

2. 掌握 Verilog 模块化实例化的方法（位置关联、名称关联），理解模块化设计的优势。

3. 掌握 Testbench 的编写方法，能够编写简单的 Testbench 对设计模块进行仿真验证。

4. 熟悉仿真验证的基本流程，能够查看仿真波形，分析仿真结果。

[教学内容]

1. 生成块 generate：generate 块的定义格式（generate...endgenerate），生成块的作用（实现重复模块的实例化、条件实例化），典型例子（生成块实现多个 D 触发器实例化）。

2. 模块化实例化：模块化设计的思路和优势，模块实例化的两种方法（位置关联、名称关联），实例化时的参数传递，典型例子（实例化选择器模块）。

3. Testbench 编写基础：Testbench 的作用（仿真验证设计模块），Testbench 的基本结构（模块声明、信号声明、设计模块实例化、激励信号产生）。

4. 激励信号产生：时钟信号、复位信号、输入信号的产生方法（利用 always 块、initial 块、循环语句），激励信号的编写规范。

5. 仿真验证流程：编写 Testbench 代码，调用 ModelSim 进行仿真，查看仿真波形，分析仿真结果是否符合设计要求。

[教学建议]

现场演示模块实例化和 Testbench 编写、仿真流程，引导学生动手练习，熟悉仿真波形分析方法。

[作业]

1. 用 generate 块实现 4 个 D 触发器实例化，编写设计模块和 Testbench。

2. 编写 Testbench，对四选一选择器进行仿真验证，分析仿真结果。

第 10 章 实验一：时钟二分频 FPGA 设计和仿真验证

[实验目的和要求]

1. 理解时钟二分频的工作原理，掌握时钟二分频电路的设计思路。

2. 能够运用 Verilog HDL 编写时钟二分频（25MHz→12.5MHz）的设计代码，包含异步复位同步释放功能。

3. 掌握 Quartus II 中 FPGA 项目的创建、代码编写、编译流程，能够排查简单的编译错误。

4. 掌握时钟二分频模块 Testbench 的编写方法，能够产生合理的激励信号（时钟、复位）。

5. 掌握 Quartus II 与 ModelSim 的联合仿真方法，能够查看和分析仿真波形，验证时钟二分频功能是否符合设计要求，排查简单的仿真错误。

[实验内容]

1. 时钟二分频原理学习：明确时钟二分频的定义，理解 25MHz→12.5MHz 时钟的频率关系，掌握利用 D 触发器或计数器实现二分频的工作原理。

2. 异步复位同步释放学习：理解异步复位、同步释放的概念和优势，掌握其避免复位信号带来亚稳态问题的原理及 Verilog 实现方法。

3. FPGA 项目创建与代码编写：在 Quartus II 中创建新项目，正确设置项目名称、保存路径、目标芯片；编写时钟二分频的 Verilog 设计代码，包含模块声明、端口声明、异步复位同步释放逻辑、二分频逻辑，添加规范注释。

4. 代码编译与错误排查：执行 Quartus II 中代码编译流程，查看编译报告，排查语法错误、端口错误等常见问题，确保代码可综合、无编译报错。

5. Testbench 编写：针对时钟二分频模块，编写 Testbench 代码，产生 25MHz 时钟信号、异步复位信号，设置仿真时间，添加合适的激励信号。

6. 联合仿真配置与运行：在 Quartus II 中设置仿真工具为 ModelSim-Altera，配置仿真参数，关联 Testbench 代码；调用 ModelSim 进行仿真，启动仿真流程，设置仿真时长。

7. 仿真波形分析与验证：查看时钟输入（25MHz）、复位信号、二分频输出（12.5MHz）的波形，验证二分频功能是否正常，异步复位同步释放是否有效；若存在问题，排查仿真错误并修正。

[实验建议]

结合时序图讲解核心原理，现场演示项目创建、编译及仿真全流程，巡回指导，及时解决学生实操问题。

[实验作业]

1. 提交设计代码、Testbench 代码及编译、仿真相关截图。

2. 简要分析仿真结果，记录实验中遇到的错误及解决方案，完成简易实验报告。

第 11 章 实验二：FPGA 蜂鸣器 PWM 发声设计和板级验证

[实验目的和要求]

1. 理解 PWM（脉冲宽度调制）的基本原理，掌握蜂鸣器 PWM 发声的设计思路。

2. 能够运用 Verilog HDL 编写 FPGA 蜂鸣器 PWM 发声的设计代码，实现不同频率的发声控制。

3. 掌握 Quartus II 中设计代码的编写、编译和语法检查方法，能够排查简单的逻辑错误和语法错误。

4. 熟悉蜂鸣器的工作原理，理解 PWM 占空比、频率对蜂鸣器音量、音调的影响。

5. 掌握 FPGA 板级验证的基本流程，能够将编译通过的代码下载到 FPGA 开发板，完

成蜂鸣器 PWM 发声的板级验证，排查简单的板级调试错误。

[实验内容]

1. 蜂鸣器与 PWM 原理学习：了解蜂鸣器的分类（有源、无源），掌握无源蜂鸣器的发声原理及需要 PWM 信号驱动的原因；理解脉冲宽度调制的定义，明确占空比、频率的概念，掌握 PWM 信号的产生方法，了解 PWM 占空比、频率对蜂鸣器音量、音调的影响。

2. 实验设计思路梳理：明确 FPGA 蜂鸣器 PWM 发声的整体设计框架，划分时钟分频模块、PWM 生成模块的分工，梳理两个模块的衔接逻辑。

3. 设计代码编写：编写 Verilog 设计代码，包含时钟分频（生成 PWM 所需时钟）、PWM 生成（控制占空比和频率）、蜂鸣器驱动逻辑，添加规范注释，确保代码逻辑正确、可综合。

4. 代码编译与错误排查：在 Quartus II 中对代码进行编译，查看编译报告，重点排查模块例化、信号声明、时序逻辑编写中的常见问题，修正语法错误和逻辑错误，确保代码编译通过。

5. 板级验证准备：检查 FPGA 开发板、USB-Blaster 下载线连接是否正常，确认下载线驱动安装无误；在 Quartus II 中设置下载参数，关联编译生成的下载文件。

6. 代码下载与板级调试：将编译通过的代码下载到 FPGA 开发板，给开发板上电，测试蜂鸣器是否正常发声；尝试调整 PWM 占空比和频率，观察蜂鸣器音量、音调的变化，验证设计功能是否符合要求；若蜂鸣器不发声或发声异常，排查板级连接、代码逻辑等问题并修正。

[实验建议]

结合实物演示蜂鸣器工作状态，讲解 PWM 生成逻辑，强调板级连接和下载规范，引导学生自主调试。

[实验作业]

1. 提交设计代码、编译截图及板级验证相关资料（截图、照片）。

2. 简要分析 PWM 参数对蜂鸣器的影响，记录调试问题及解决方案，完成简易实验报告。

第 12 章 实验三：FPGA 流水灯应用设计、仿真及板级验证

[实验目的和要求]

1. 掌握 FPGA 流水灯的设计思路和实现方法，理解时序逻辑在流水灯设计中的应用。

2. 能够运用 Verilog HDL 编写流水灯设计代码，实现不同速度、不同模式的流水效果（如左移、右移、循环闪烁）。

3. 掌握流水灯模块的仿真验证方法，能够编写 Testbench、查看仿真波形，验证设计逻辑正确性。

4. 熟练完成 FPGA 板级验证，能够将代码下载到开发板，调试流水灯效果，排查板级调试常见错误。

[实验内容]

1. 流水灯原理学习：明确流水灯的工作机制，理解时钟分频、移位寄存器在流水灯设计中的作用，梳理左移、右移等流水模式的实现逻辑。

2. 设计思路梳理：划分时钟分频模块（生成流水灯所需时钟）、移位逻辑模块（实现 LED 灯移位控制），明确模块间信号衔接关系，确定流水速度调节方式。

3. 设计代码编写：在 Quartus II 中编写 Verilog 代码，包含时钟分频、移位逻辑、LED 驱动功能，添加规范注释，支持至少两种流水模式，可调节流水速度。

4. 代码编译与错误排查：执行编译流程，查看编译报告，修正语法、逻辑及模块例化中的常见错误，确保代码可综合、无报错。

5. 仿真验证：编写 Testbench 代码，产生时钟、复位等激励信号，调用 ModelSim 进行联合仿真，查看 LED 输出波形，验证流水模式和速度是否符合设计要求，排查仿真错误。

6. 板级验证：检查开发板连接、下载线驱动，设置下载参数，将编译通过的代码下载到 FPGA 开发板；上电测试，调试流水速度和模式，确保 LED 灯流水效果正常，排查板级连接、引脚配置等问题。

[实验建议]

简化流水灯设计难度，重点讲解移位逻辑实现方法，现场演示仿真和板级调试流程，引导学生自主调整流水参数。

[实验作业]

1. 提交设计代码、Testbench 代码及编译、仿真、板级验证相关截图。

2. 简要说明流水灯设计思路，记录实验中遇到的问题及解决方案，完成简易实验报告。

第 13 章 实验四：FPGA3-8 译码器应用设计、仿真及板级验证

[实验目的和要求]

1. 巩固 3-8 译码器的工作原理，掌握基于 FPGA 的 3-8 译码器设计与实现方法。

2. 能够运用 Verilog HDL 编写 3-8 译码器设计代码，实现输入信号到输出信号的正确译码。

3. 掌握 3-8 译码器的仿真验证流程，能够编写 Testbench、分析仿真波形，验证译码功能正确性。

4. 完成板级验证，能够将代码下载到 FPGA 开发板，通过按键或拨码开关控制译码输出，排查板级调试错误。

[实验内容]

1. 3-8 译码器原理回顾：复习 3-8 译码器的真值表、逻辑表达式，明确输入（3 位）与输出（8 位）的对应关系，理解译码器的使能端作用（可选）。

2. 设计思路梳理：确定 3-8 译码器的 Verilog 实现方式（assign 连续赋值或 always 块），明确输入端口（3 位地址输入、使能端）和输出端口（8 位译码输出）的定义。

3. 设计代码编写：在 Quartus II 中创建项目，编写 3-8 译码器 Verilog 代码，添加规范注释，确保代码逻辑符合真值表要求，可综合。

4. 代码编译与错误排查：执行编译流程，查看编译报告，修正语法、信号声明等常见错误，确保代码无编译报错。

5. 仿真验证：编写 Testbench 代码，产生 3 位地址输入、使能信号等激励，覆盖所有输入组合；调用 ModelSim 进行仿真，查看译码输出波形，验证是否与真值表一致，排查仿真错误。

6. 板级验证：将 FPGA 开发板的按键/拨码开关连接到译码器输入端口，LED 灯连接到输出端口；检查开发板连接、下载参数设置，下载代码并上电测试；通过操作按键/拨码开关，观察 LED 灯显示，验证译码功能正常，排查引脚配置、板级连接等问题。

[实验建议]

结合前期逻辑设计基础知识点，回顾译码器原理，多举例演示代码编写逻辑，重点指导板级引脚配置和调试方法。

[实验作业]

1. 提交设计代码、Testbench 代码及编译、仿真、板级验证相关截图。

2. 附上 3-8 译码器真值表，简要分析实验结果，记录问题及解决方案，完成简易实验报告。

综合考查

[考查内容]

1. 核心知识点考查：FPGA 的基本原理、Verilog 语法（模块结构、赋值语句、条件语句、循环语句等）、逻辑设计基础（逻辑代数、组合逻辑、时序逻辑）的相关知识点，以理论问答或代码分析形式考查。

2. 实操能力考查：给出综合设计任务（如“基于 FPGA 的流水灯+蜂鸣器控制系统”“基于 FPGA 的 3-8 译码器显示系统”），要求学生独立完成代码编写、仿真验证、引脚分配、程序下载和板级验证，实现指定功能。

3. 问题解决能力考查：在实操过程中，设置常见的错误（如代码逻辑错误、引脚分配错误、仿真参数设置错误），考查学生的排查和解决能力；要求学生规范记录设计过程、遇到的问题及解决方案。

4. 思政素养考查：考查学生的严谨务实、精益求精的工匠精神，以及规范编写代码、认真记录实训过程的职业素养。

[建议]

考查任务的难度要适中，要结合整个教学过程中学生的具体表现情况来进行，尽可能覆盖本课程的核心知识点和实操技能，既要考查基础能力，也要兼顾对学生创新能力的考查，如允许学生在指定功能基础上进行优化。

另外，由于课程性质注重实操，所以建议最终的考核成绩构成为 50%平时+50%期末，鼓励学生在平时多动手实践。