**北京科技大学数字逻辑实验报告**

学院： 计通 专业： 班级：

姓名： 学号： 实验日期： 2024年 9 月 27 日

**实验名称：**实验一、基础练习

**实验目的：**熟悉实验环境，学习如何安装Vivado、如何使用 Vivado 2018创建工程、代码编辑、RTL分析、仿真等设计流程。

**实验内容：**

（1）学习所有视频以及“lab0.pdf”，了解 Vivado 设计流程和功能

（2）按照“lab1.pdf”完成数码管和加法器实验；

（3）Vivado 代码编辑和RTL分析；

**实验结果与分析：**

（1）观看提供的所有视频资料；学习：Vivado设计流程中的基本概念.pdf、约束文件.ppt。回答以下问题：

* 描述 Vivado 的设计流程

编写、综合、实现verilog源码，编写管脚约束，创建比特流，链接电路板

* 什么是网表

是描述设计的一种方式，包括单元、引脚、端口和网络

* Vivado 设计流程中，Synthesis 的作用是什么？

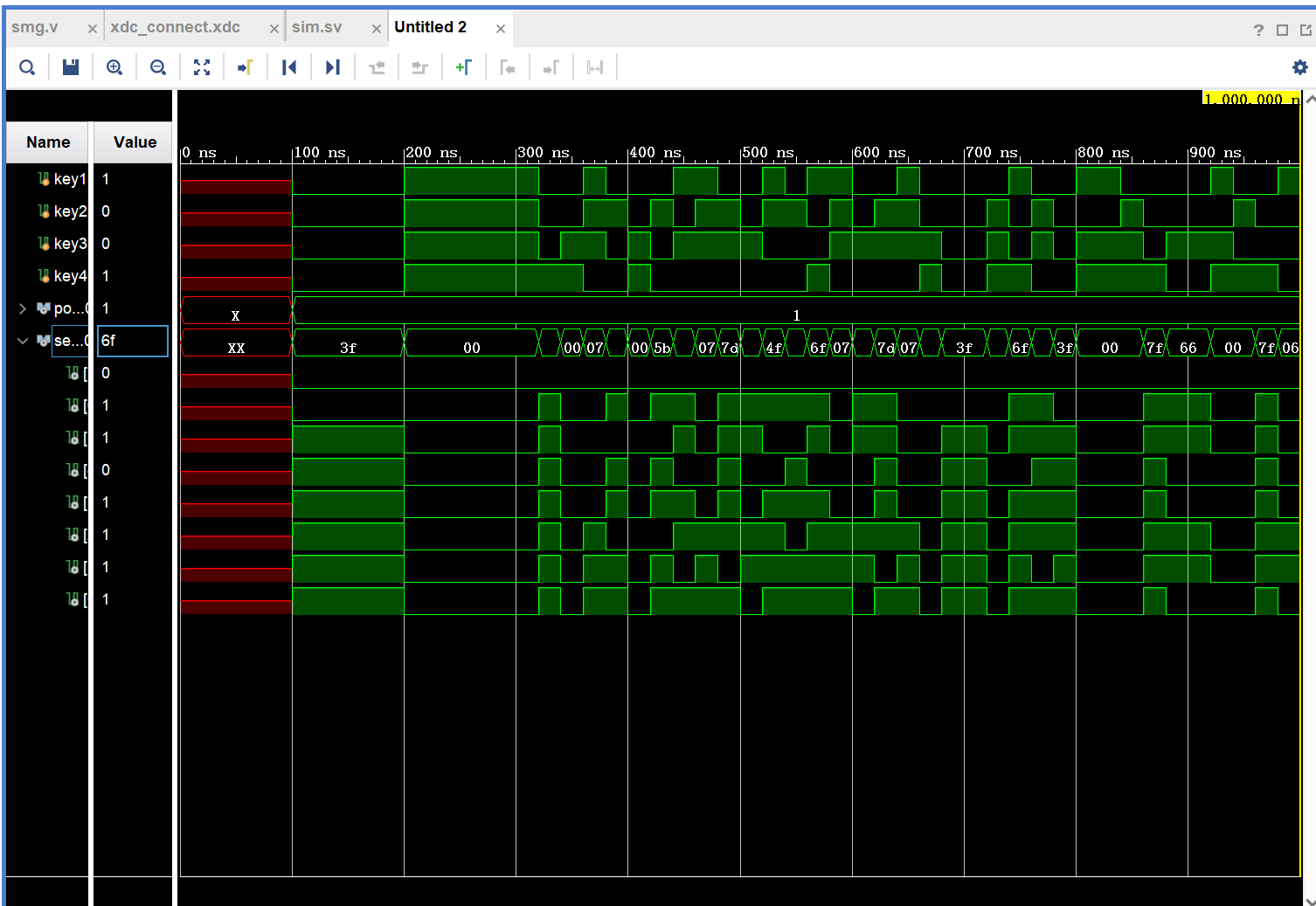
将Verilog编写的设计代码转化，以电路图或网表文件的形式呈现

* Vivado 设计流程中，Implementation 的作用是什么？

将综合阶段生成的门级网表转化为具体的硬件实现。通过实现阶段，设计被具体化为可以在FPGA上物理实现的形式

（2）按照“lab1.pdf”完成数码管实验；

* 给出数码管仿真结果截图，对波形进行简要解释：



根据key1234的输入信号按照规则转化为seg0到7的高低电平

* 给出板子运行结果照片，以及你的操作过程：

图片包含 游戏机, 电子, 电路

描述已自动生成

输出9

图片包含 电子, 电路, 游戏机

描述已自动生成

输出6

右侧的4个开关表示二进制从高到低的4位，高电平表示1，低电平表示0，数码管显示这些开关表示的二进制数

* segMsg的输入信号的作用都是什么？

Key：表示开关输入的是0还是1

Seg：控制数码管亮灭达到显示数字的目的

Pos：控制显示数码管的位置

（3）按照“lab1.pdf”完成加法器实验；

* 给出两位半加器源代码，并进行简要解释：

module TwoBitAdder(

input wire [1:0] A, // 两位输入A

input wire [1:0] B, // 两位输入B

output wire [1:0] Sum, // 输出的和

output wire Cout // 输出进位

);

wire carry1; // 第一位的进位

// 第一位的半加器

assign Sum[0] = A[0] ^ B[0]; // 计算和

assign carry1 = A[0] & B[0]; // 计算进位

// 第二位的半加器

assign Sum[1] = A[1] ^ B[1] ^ carry1; // 第二位加上前一位的进位

assign Cout = (A[1] & B[1]) | (carry1 & (A[1] ^ B[1])); // 计算总进位

endmodule

* 给出两位半加器仿真结果截图，对波形进行简要解释：

图形用户界面

描述已自动生成

尝试了7种不同的 A B 组合，其中A B 表示两个二位二进制数，Sum表示他们的和，Cout 表示进位

* 给出板子运行结果照片，以及你的操作过程：

这四个是半加器板子上的照片

后面四个开关表示两个二进制位

后两个灯表示sum，倒数第三个表示进位





* 如果要实现两位全加器，需要在两位半加器的基础上做什么改动？

使用两个二位半加器就可以实现全加器了

module two\_bit\_full\_adder(

input wire [1:0] A, // 两位输入A

input wire [1:0] B, // 两位输入B

input wire Cin, // 输入进位Cin

output wire [1:0] Sum, // 输出的和

output wire Cout // 输出进位Cout

);

wire carry1, carry2; // 用于存储中间进位

// 第一位的全加器

assign Sum[0] = A[0] ^ B[0] ^ Cin; // 计算第一位的和

assign carry1 = (A[0] & B[0]) | (Cin & (A[0] ^ B[0])); // 计算第一位的进位

// 第二位的全加器

assign Sum[1] = A[1] ^ B[1] ^ carry1; // 计算第二位的和

assign carry2 = (A[1] & B[1]) | (carry1 & (A[1] ^ B[1])); // 第二位进位

assign Cout = carry2; // 总的进位输出

endmodule

（4）创建一个工程，自己指定工程位置和工程名称，新建空白源程序文件，依次完成下面代码编辑和RTL分析：

* 分别编写教材图2.37、2.38例子，观察vivado工具的RTL分析结果，截图如下；

图示

描述已自动生成

图示, 示意图

描述已自动生成

* 分别编写教材图2.40、2.41例子，观察vivado工具的RTL分析结果，截图如下；

图示, 示意图

描述已自动生成

图示, 示意图

描述已自动生成

* 分别编写教材图3.18、3.20例子，对比vivado工具的RTL分析结果，截图并给出你对结果的理解；

图示, 示意图

描述已自动生成

图示, 示意图

描述已自动生成

这两个是全加器的两个不同的写法，vivado工具生成的电路图是等价相同的。

* 编写教材图3.22例子，观察vivado工具的RTL分析结果，截图并给出你对结果的理解；

图示, 示意图

描述已自动生成

它以一个全加器为基础，实现了一个四位的加法器

（5）实验中遇到哪些问题，是如何解决的。（如果没遇到问题可以不写）

Verilog 语法不熟悉，不知道板子接口代号，不会仿真。

通过查阅网上代码，自己摸索，查阅视频资料解决。

（6）本次实验的感受及建议（如没有体会和想法可以不写）。

豪难。。。