**实验二报告**

学号 2018K8009922027

姓名 李国峰

箱子号 1

一、实验任务（10%）

Lab2有三个子任务，第一个任务是对一组有32个32位寄存器、两读一写端口的寄存器堆进行仿真，结合代码观察波形并理解其工作原理。

第二个任务是调用Xilinx的库IP分别实例化同步RAM和异步RAM，进行仿真，观察波形对比二者的异同，然后进行综合和实现，并且利用VIVADO自带的工具查看时序结果和资源利用率，借此预测设计上板的成功率。

第三个任务简而言之就是debug任务，通过阅读代码、观察波形、观察综合情况和上板调试来定位与预期功能不符的地方，并进行相应的修改。

二、实验设计（0%）

略。

三、实验过程（90%）

（一）实验流水账

2020年9月16日20:00-23:00 阅读讲义，基本掌握提到的design和debug技巧；

2020年9月17日15:00-17:00/ 18:00-21:00阅读任务要求，完成子任务1、2的仿真和实现，阅读debug任务要求和源代码，没有读懂；

2020年9月18日10:30-11:30，基本读懂任务要求和代码的对应关系

2020年9月18日12:00-15:00，完成debug，并完成仿真和最后的上板测试，测试通过。

（二）子任务一

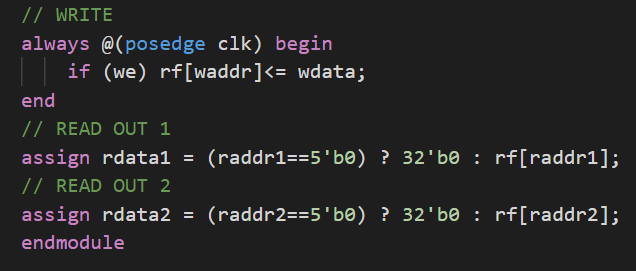
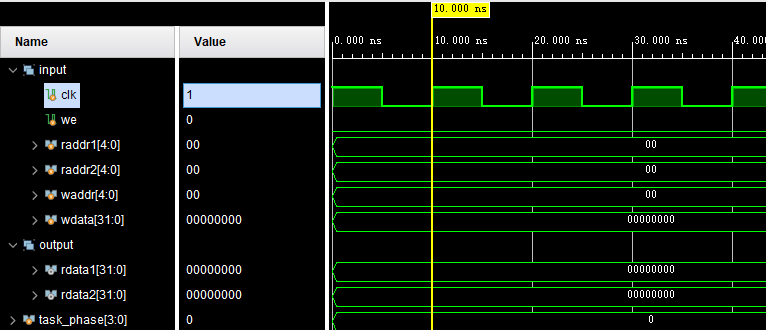
新建工程后导入代码源文件和仿真激励文件，进行仿真，结合代码（图1）观察并理解波形行为。

图2

图1

根据代码，我们可以得到以下信息：当且仅当时钟上升沿（posedge clk）且写使能信号（we）位于高电平时才能将数据写入指定地址的寄存器中。在图2所示时间段内，写使能信号都位于低电平，因此无论地址和数据如何变化，寄存器都不会被写入数据。

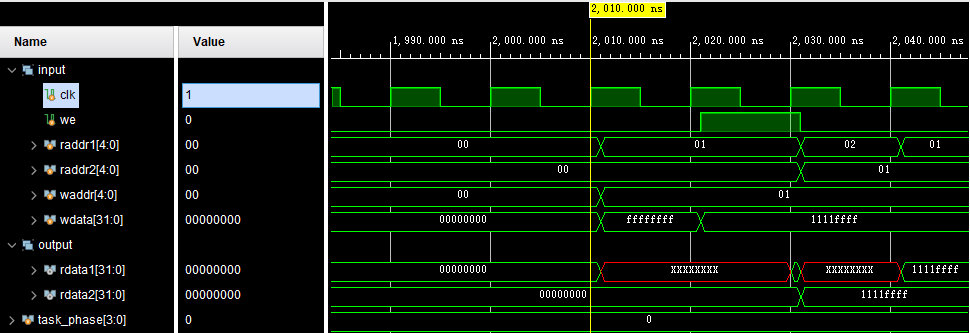
观察图3，当时钟上升沿到来且写使能信号有效时（2030ns），写地址（waddr）和写数据（wdata）同时生效，数据1111ffff被写入01寄存器中，此后在新的数据写入之前，01寄存器读出的数据会一直保持1111ffff。换句话说，数据的写入只有在时钟上升沿到来且写使能信号有效时才能进行，而数据的读取是一直在进行的，读出的数据只与读取地址有关，与时钟信号无关。

图3

值得一提的是，在图3中出现的X波形是因为寄存器中事先没有存入数据产生的，并非设计缺陷。

（三）子任务二

1. 仿真行为对比分析

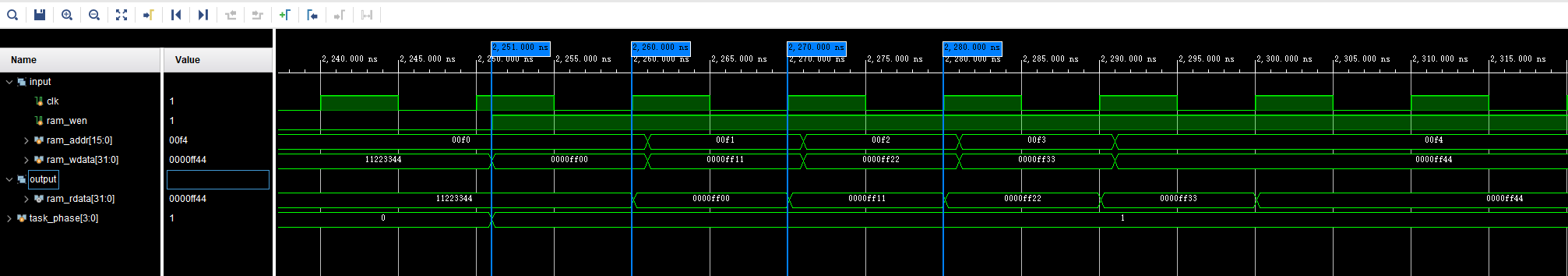
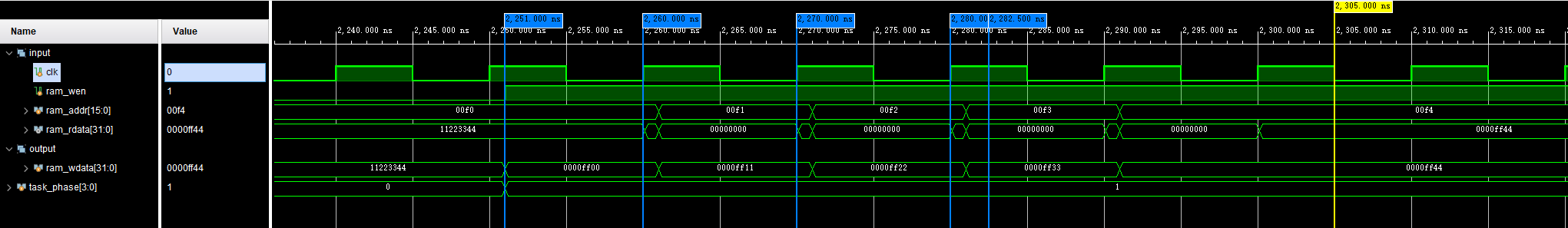
如图4和图5分别为同步RAM和异步RAM在同一时间段内的仿真波形。

图5

图4

对比图4和图5可知，在ram\_wen信号有效时，同步RAM在某一时钟上升沿将数据写入，只有在下一个时钟上升沿到来时才可以读出数据；而异步RAM可以实时读取写入的数据。

1. 时序、资源占用对比分析

由下图6和图7可知，同步RAM时序满足极好，异步RAM时序违约且十分糟糕，意味着异步RAM设计上板失败的可能性很大。

考虑资源利用率，同步RAM耗费主要资源远远少于异步RAM，BRAM占用率也远远小于LUTRAM，同步RAM和异步RAM在IO和BUFG上占用资源相同，这意味着异步RAM电路的综合实现比同步RAM更加复杂，花费时间更多，可以大致判断异步RAM的电路结构比同步RAM复杂得多。

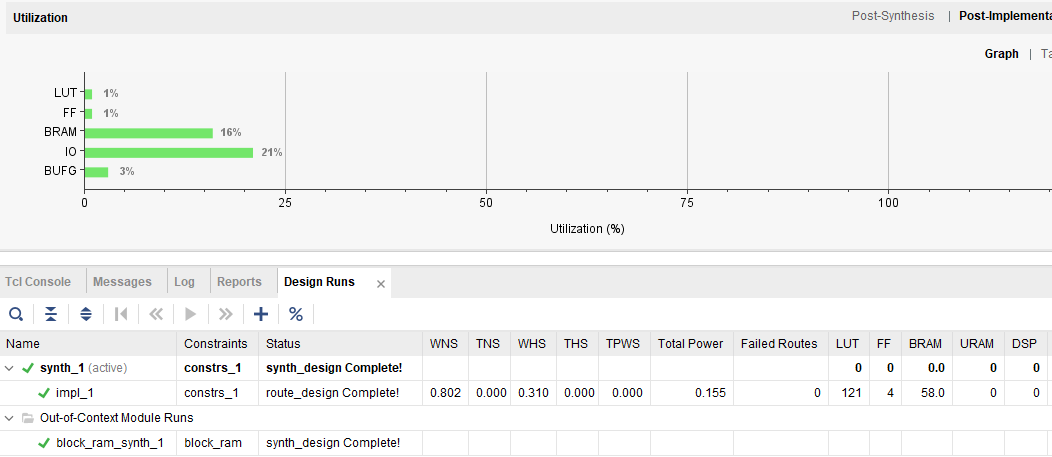
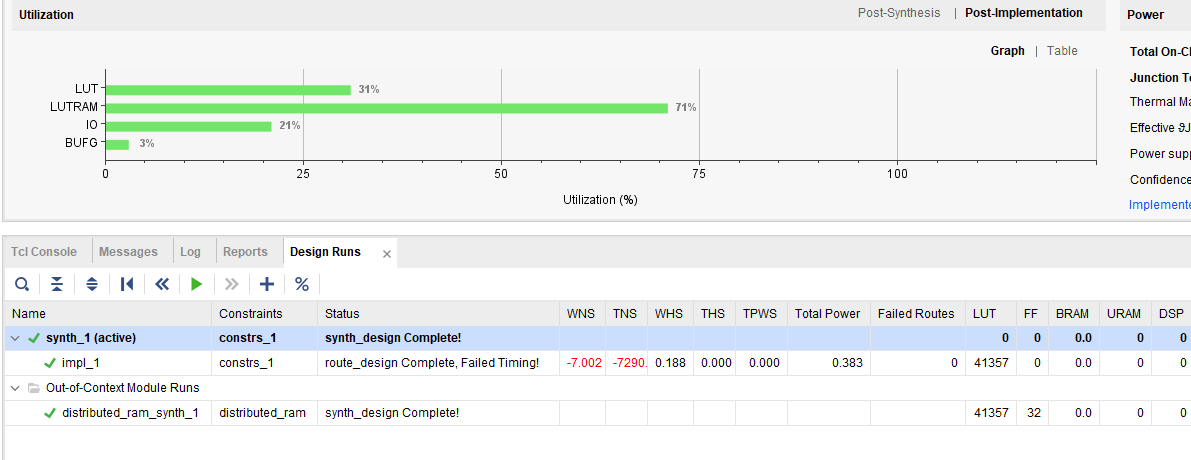


图7

图6

1. 总结

（四）子任务三

1、错误1：波形为Z

（1）错误现象

在波形中存在蓝色的波线，读出来的值为Z，如图8。

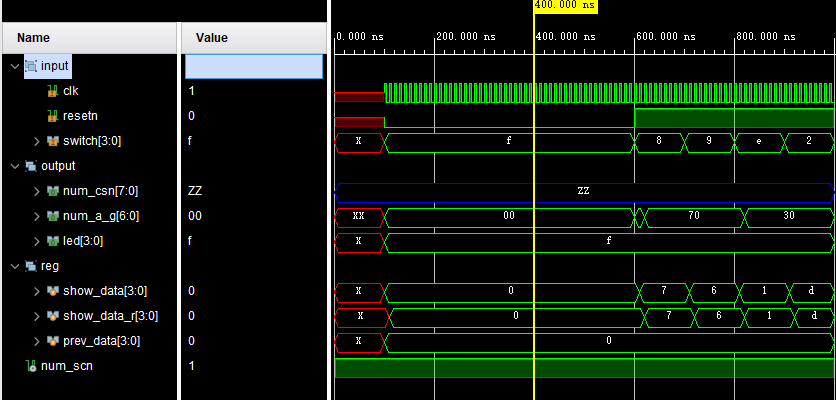


图8

（2）分析定位过程

观察波形图，找到与出现的Z波形有关系的所有信号即num\_csn，回到代码中寻找这些波形之间的关系，再结合讲义中讲述的Z波形出现的原因可能是存在端口没有连接，有针对性地进行寻找。同时在波形图中发现突兀的num\_scn信号。

（3）错误原因

调用模块时端口名输入错误（笔误）导致连接故障，正确的端口名num\_csn被写成num\_scn。如图9。

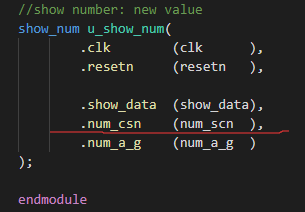


图9

（4）修正效果

更正端口名即可，之后Z波形消失。

（5）归纳总结（可选）

输入笔误，可能是由于输入过快导致的字母顺序错误，更仔细一些即可，无需刻意关注。

2、错误2：波形为X

（1）错误现象

在波形中存在红色的波线，读出来的值为X。如图10所示，show\_data和show\_data\_r的波形始终为X。

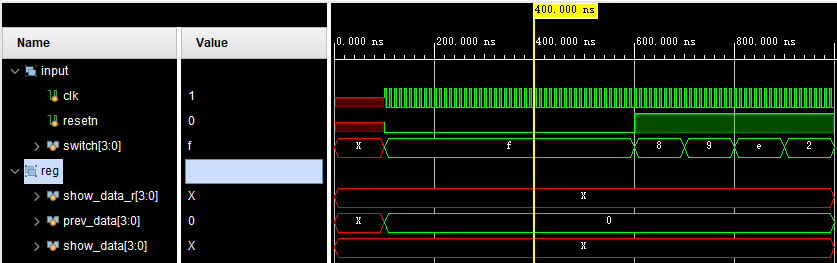


图10

（2）分析定位过程

观察波形图，找到与出现的X波形有关系的所有信号，回到代码中寻找这些波形之间的关系，再结合讲义中讲述的X波形出现的原因可能是存在reg型变量没有被赋值或形成了多驱动，然后有针对性地进行寻找。如图11源代码片段所示，show\_data\_r在时钟上升沿被赋给show\_data的值，而show\_data被赋给输入信号switch取反的值，但这一语句被注释掉了，所以reg型信号show\_data没有被赋值。

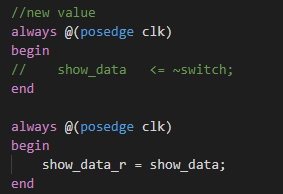


图11

（3）错误原因

Reg型变量show\_data没有被赋值。

（4）修正效果

去掉//，让赋值生效，X波形消失。

（5）归纳总结（可选）

Reg型变量必须被赋值，在非组合逻辑中（always语句中），能被赋值的变量一定是reg型。

3、错误3：波形停止

（1）错误现象

进行仿真时出现了波形停止。

（2）分析定位过程

出现波形停止，认为是出现了组合环路。由于变量嵌套较少，很快找到了互相调用的变量。

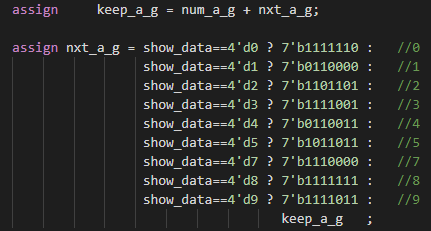
（3）错误原因

图12

如图12所示，存在变量的互相调用，nxt\_a\_g被赋予keep\_a\_g的值，keep\_a\_g同时也被nxt\_a\_g的值影响，形成了组合环路。

（4）修正效果

根据功能需要删除多余的信号调用，只需将keep\_a\_g的赋值语句改为keep\_a\_g=num\_a\_g;即可。组合环消失，仿真正常进行。

（5）归纳总结（可选）

深度较低的组合环可以根据观察得到，深度较深时可以先进行综合实现，根据log来判断出问题的信号。

4、错误4：越延采样

（1）错误现象

波形中信号并没有随信号正确变化。如图13所示，pre\_data每一位的波形一直为0，并没有如预期一样显示前一状态的波形。根据代码，只要在时钟上升沿来临，resetn信号为高电平且show\_data与show\_data\_r不相同时，prev\_data信号必然要获得show\_data\_r的值，但事实并非如此。

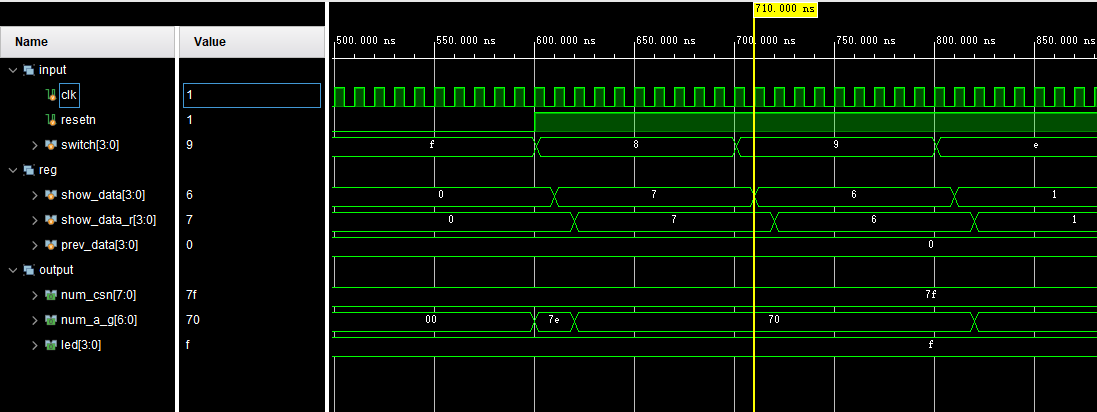


图13

（2）分析定位过程

阅读代码的过程中发现在两个always模块中分别出现了阻塞赋值和非阻塞赋值，相关的信号恰好是出现波形异常的信号。

（3）错误原因

不合理使用阻塞赋值导致越延采样。

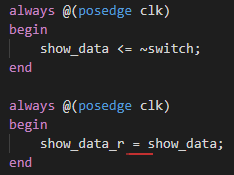


图14

（4）修正效果

使用非阻塞赋值，信号变化符合期望。

（5）归纳总结（可选）

需要深入理解阻塞和非阻塞赋值。阻塞赋值的过程是立即执行的，即阻塞赋值运算符右侧表达式求值完后会立刻更新至运算符左侧，且该执行过程不受其他语句执行的影响。非阻塞赋值在执行中，在当前仿真时间槽（time-slot）开始分析计算获得右侧表达式的值，在当前时间槽执行结束时更新左侧表达式的值，在右侧表达式分析计算和左侧表达式被更新之间，任何其他事件都可以执行，同时也有可能修改以及计算完成的右侧表达式的值，即左侧表达式获得的值不一定是最新的，非阻塞赋值的过程不影响其他语句的执行。

5、错误5：功能bug

（1）错误现象

上板时最左侧数码管无法显示数字6。

（2）分析定位过程

阅读代码，发现缺失实现数字6的内容，如图6所示。

（3）错误原因

代码不完整导致功能缺失。

（4）修正效果

加上数字6的驱动代码，然后可以正常显示数字6。值得一提的是，数码管的7段信号与num\_a\_g每一位的对应关系并不是显而易见的，需要根据其他的数字的num\_a\_g信号来推出该对应关系。

（5）归纳总结（可选）

略。

四、实验总结（可选）

实验手册的内容讲解十分详尽，特别适合我这样数字系统设计基础不牢固、经验不足的学生阅读和参考，我在阅读和上手敲代码的过程中学到了很多实用的设计、调试技巧。

做完试验后，回顾做实验的整个过程，觉得不是很难，很多时间花在对debug实验题目的理解上，不是很懂代码的各个端口和开发板的对应关系，因此一开始觉得无从下手。这种感觉十分玄学，可能和思维的状态好坏有关。

异步RAM的综合实现花了很长时间，一开始还以为vivado又卡住了。同步RAM和异步RAM的波形比较费了一番功夫，因为仿真的时间尺度较小，为了能够充分观察局部的细节进而方便比较，牺牲了同时对整体变化情况的把握。此外，将信息完整且清晰可见的波形图放到实验报告中也造成了一些麻烦。

希望以后的每一个项目都能按时高质量完成。