**体系结构课程第 一 次实报告**

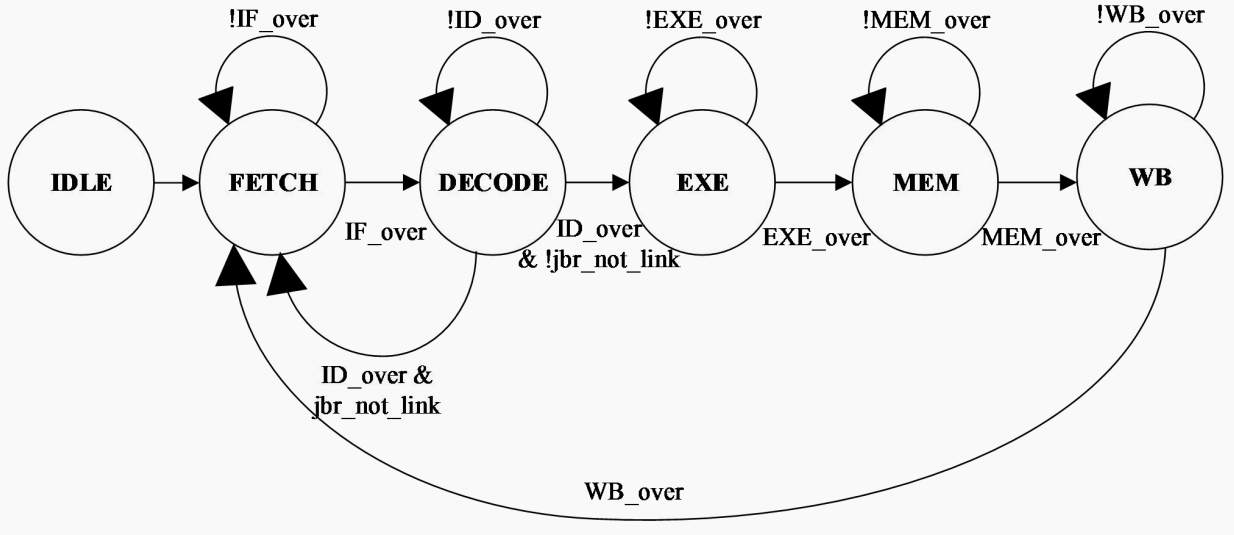
**实验名称：多周期 CPU 实现**

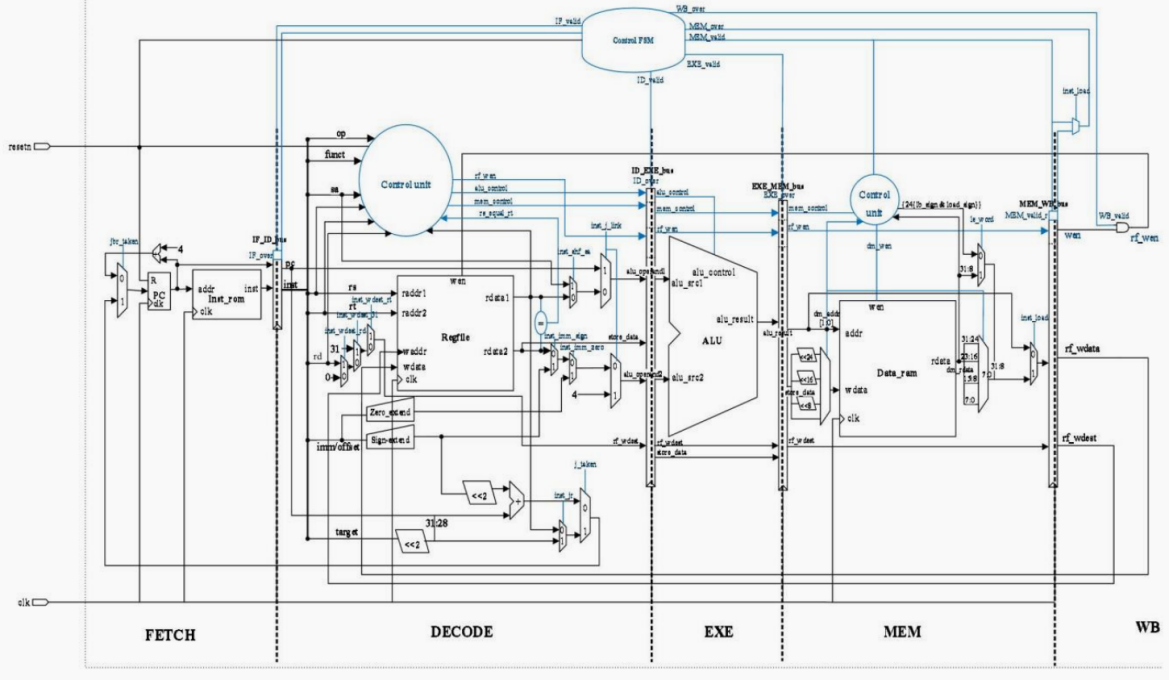
学号： 2113203 姓名： 付政烨 班次： 0855

1. **实验目的**
2. 在单周期 CPU 实验完成的提前下，理解多周期的概念。
3. 熟悉并掌握多周期 CPU 的原理和设计。
4. 进一步提升运用 verilog 语言进行电路设计的能力。
5. 为后续实现流水线 cpu 的课程设计打下基础。
6. **实验内容说明**

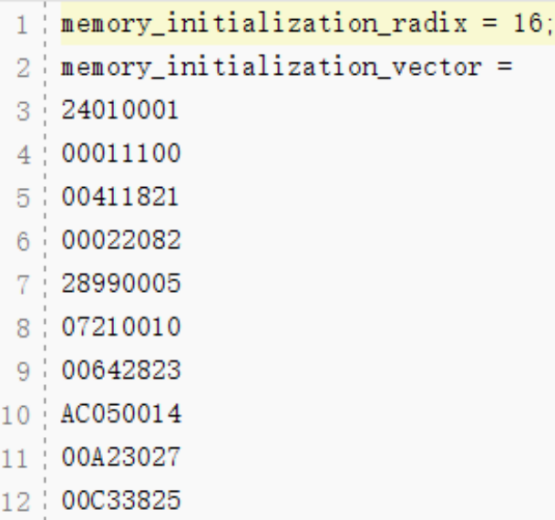
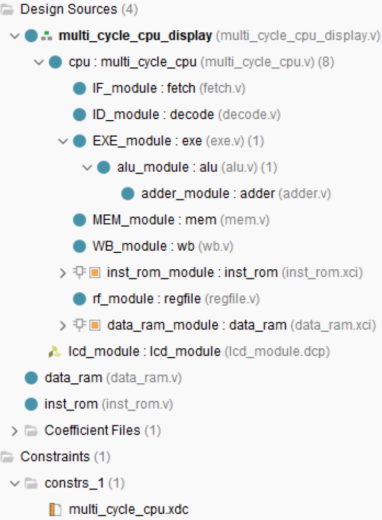
结合实验指导手册的多周期实验内容，实现多周期 CPU 功能并完成如下内容：

1. 在给出代码的基础上，完成多周期 CPU 的功能仿真，分析在指令 rom 中的指令的
2. 执行过程。
3. 找到程序执行过程中存在的 bug 并尝试修复（修复方式不唯一，建议多做尝试）。
4. 完成实验报告，并在报告中介绍自己的分析、bug 追踪核调试过程，并做好自己的
5. 总结，为后续实验做准备。（本次实验着重看仿真结果，可以不上箱验证）
6. **实验原理图**



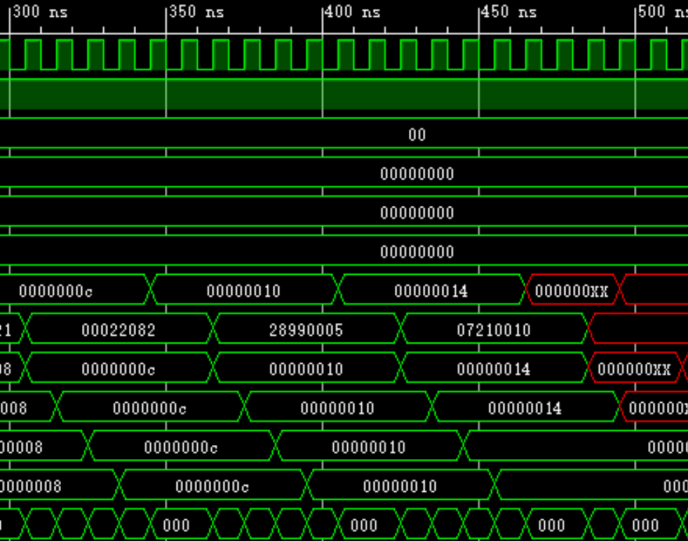


1. **实验步骤**
2. **生成 IP 核**：根据实验教程中的指导，使用 Xilinx 提供的库函数进行 RAM 和 ROM 的 IP 核实例化，生成对应的 IP 核。具体来说，在生成 RAM IP 核时，利用 Xilinx 的 IP 核生成器（IP Catalog）来选择合适的内存模块，根据设计需求进行配置；对于 ROM IP 核的生成，需要在配置过程中加载初始化文件（.coe 文件），以便为 ROM 提供预设的内容数据。值得注意的是，在编写 .coe 文件时，我们需确保文件格式符合 Xilinx 工具链的要求，包括数据的存储格式（如二进制、十六进制等）及对应的地址映射关系。
3. **导入源码**：在完成 IP 核的生成后，接下来需将设计源码导入到项目中。导入过程包括将编写的 Verilog 代码，以及生成的 IP 核文件正确集成到项目目录中。此步骤的关键在于确保所有源码文件与 IP 核文件的路径正确，且编译设置与设计要求相匹配，从而确保在后续的仿真和实现阶段，所有模块能够顺利连接与协同工作。

**生成 IP 核**  **导入源码**

1. **实验结果分析**
2. **问题分析（改正前）：**在程序执行过程中，当程序计数器（PC）值为 00000014 时，程序发生异常。根据查询实验指导手册表 8.5，发现该位置对应的指令为 bgez $25, #16，即“当寄存器 $25 中的值大于等于零时，跳转至地址 PC + 16”。进一步查阅表 8.4，得知该指令为比较后跳转指令（Branch on Greater Than or Equal to Zero）。结合指令功能，可以推测异常发生的原因是在执行此跳转指令时，寄存器 $25 中的值尚未被上一条指令正确写入，从而导致跳转判断错误，出现程序异常。这种现象可能是由于数据冒险问题导致的。



1. **问题修正（代码改正）：**在源代码中，信号的锁存通常是通过 always 块来实现的，且锁存信号的触发通常依赖于时钟沿。由于信号变化与时钟边沿的同步问题，可能会导致数据传输出现错误，特别是在某些时钟周期内，信号变化未能及时锁存，从而引发时序问题。为解决此问题，可以将所有总线信号的锁存改为 always @(\*) 触发，而不是依赖时钟沿。这种做法能够确保所有相关信号的变化在组合逻辑中得到及时反映，从而避免因时序不匹配而导致的数据冒险和其他异常情况。

// IF到ID的锁存信号

// always @(posedge clk)

always @(\*)

begin

if (IF\_over)

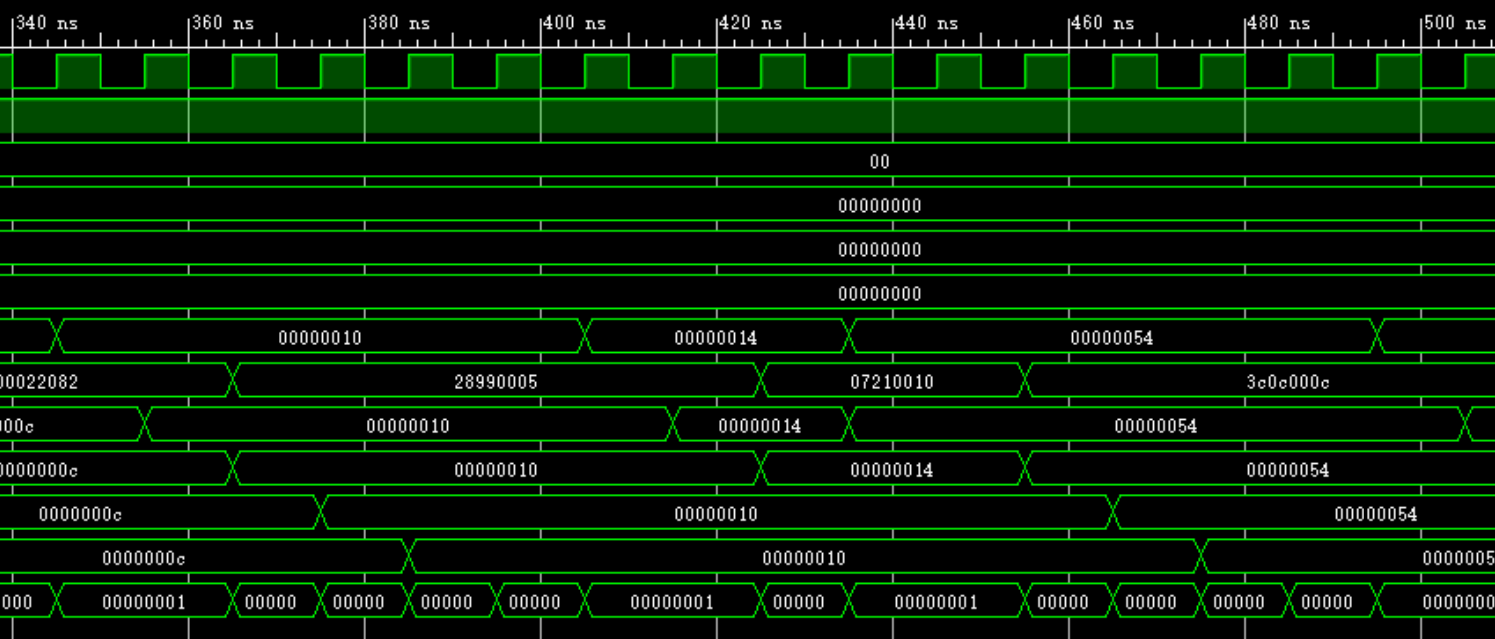
begin

IF\_ID\_bus\_r <= IF\_ID\_bus;

end

end

1. **修正结果（改正后）：**经过上述修改后，观察波形仿真结果，发现当程序计数器（PC）值为 14 时，程序能够正确跳转至 54H，且后续程序能够正常运行，未再出现异常情况。这表明通过对信号同步机制的调整，解决了数据冒险问题，确保了程序的正确执行。因此，本次修改成功消除了由于时序不同步导致的异常。



1. **总结感想**

通过本次实验，我深入理解了多周期 CPU 的设计原理与实现过程，并掌握了如何使用 Verilog 语言实现多周期 CPU 的功能仿真。实验中，结合实验指导手册的多周期内容，我逐步实现了多周期 CPU 的功能，并通过仿真分析指令执行过程，成功发现并修复了程序运行中的数据冒险问题。问题定位时，通过查阅实验手册指令表，分析了指令的功能及执行逻辑，从而发现问题原因是信号的时序不同步。随后，我通过将锁存逻辑从基于时钟沿的触发机制调整为组合逻辑触发，确保了信号变化能够及时反映在总线数据中，从而解决了数据冒险问题。本次实验让我更加深刻认识到时序问题在 CPU 设计中的重要性，同时也体会到仿真分析和问题调试的复杂性与挑战性。在 bug 修复过程中，我尝试了多种方法，最终通过信号触发机制的优化解决了问题。这一过程不仅提升了我对 Verilog 语言的应用能力，还让我对 CPU 内部结构与信号交互的细节有了更深入的理解。