《计算机体系结构》双语课程大纲

课程号：21191061

课程名称：计算机体系结构

周学时：2.5-2.0　　学分：3.5

教学目的与基本要求：（限100－150字）

通过课堂教学和实验掌握计算机体系结构的基本概念和原理，掌握计算机设计的基本方法、定量设计原理并能应用于实践，了解现代处理器流水线的原理（包括支持多周期操作的流水线、动态调度流水线，以及高级流水线技术）；了解存储器层次结构的原理，以及提高Cache、memory性能的方法；了解指令级并行性（ILP）、数据并行性（DLP）、线程级并行性的开发技术。

主要内容及学时分配：

Ch0: Course Introduction （1学时）

Ch1：量化设计与分析原理（5学时）

1.1 引言
1.2 _计算机分类
1.3 计算机体系结构定义
1.4 器件技术趋势
1.5 集成电路的电压和功耗
1.6 成本趋势
1.7 可靠性
1.8 性能测量、评价、总结
1.9 计算机设计的量化设计原理
1.10 _ 综述：性能、价格、功耗
1.11 谬误与陷阱
1.12 总结

Ch2：存储器层次结构设计（5学时）

2.1 引言
2.2 存储技术与优化
2.3 10种Cache性能优化方法 (+ App B.3)
2.4 虚拟存储器与虚拟机
2.5 交叉问题：存储器层次结构设计
2.6 综合：ARM Cortex-A53的存储器层次结构
2.7 谬误与陷阱
2.8 总结：技术前望

Ch3： 指令级并行性及其开发方法（16 学时）

3.0 流水线异常、扩展流水线以支持多周期操作 (App C.4-C.6)
3.1 指令级并行性: 概念和挑战
3.2 开发ILP的基础编译技术
3.3 用高级转移预测减少分支代价（+ App C.2）
3.4 用动态调度克服数据竞争延迟 (+ App C.7)
3.5 动态调度: 案例与算法
3.6 基于硬件的推测执行
3.7 用多发射与静态调度开发ILP
3.8 用动态调度、多发射、推测执行开发ILP
3.9 指令发送与推测执行的高级技术
3.10 交叉问题
3.11 多线程: 开发线程级并行性提高单处理器的吞吐率
3.12 综合：Intel Core i7 6700 和ARM Cortex-A53以及鲲鹏流水线技术
3.13 谬误与陷阱
3.14 总结：技术展望

Ch4： 向量机,SIMD和GPU中的数据级并行性（8 学时）

4.1 引言
4.2 向量机结构
4.3 用于多媒体处理的SIMD指令集扩展
4.4 图形处理部件GPU
4.5 检测和强化循环级并行性
4.6 交叉问题
4.7 综合: 嵌入式系统/服务器GPUs，特斯拉和Core i7
4.8 谬误与陷阱
4.9 结束语

Ch5： 线程级并行性（9 学时）

5.1 引言
5.2 集中共享存储的多处理器结构
5.3 对称共享存储的多处理器结构的性能
5.4 分布共享存储器和基于目录的一致性协议
5.5 同步机制基础
5.6 存储连贯性模型简介
5.7 交叉问题
5.8 综合: 多处理器结构及其性能
5.9 谬误与陷阱
5.10 多核技术的未来
5.11 结束语

实验内容：

1. 通过实验深入理解CPU的高级流水线的原理。
2. 用Verilog语言，在FPGA板上完成下列实验,并进行功能模拟：
   1） 实现带forwarding和Predict-Not-taken的pipeline CPU，支持完整的RISC V 32I全部指令；
   2） 在实验1）的基础上实现简单中断、异常处理；
   3） 设计实现2-路组关联的Cache。
   4） 将Cache整合到实验2)的CPU流水线中。
   5） 扩展pipeline CPU支持多周期操作：整数乘法，整数除法/取余运算；按序发射，乱序完成，能够检测流水线竞争（WAW，WAR）
   6） 实现Scoreboard或Tomasulo算法，实现动态调度算法流水线。

评分规则：

最终成绩 = 平时成绩* 60% + 期末考试成绩 * 40%
平时成绩 = 作业成绩、测验成绩（28%）+ 实验成绩（32%）+ 奖励（<=10%）
实验成绩 = Lab1(3weeks, 6%) + Lab2(2weeks, 4%) + Lab3 (1weeks, 3%) + Lab4(2weeks, 4%) + Lab5 (3weeks, 7%) + Lab6 (5weeks, 8%)

相关教学环节安排：

1. 采用多媒体投影教学。
2. 在整个教学过程中有两次以上课堂小测验，计入课程总成绩。
3. 每章会布置相应作业，总作业量大约需20小时。
4. 课程实验，总学时32学时。
5. 在学在浙大平台上提供教学网站，提供课程资料， 发布作业、实验，提交作业报告；
6. 学习鼓励政策，对超额完成实验者酌情加分。

推荐教材或参考书：（含教材名、主编、出版社、出版年）

1. 《Computer Architecture--A Quantitative Approach》, 6th Edition,
   John L. Hennessy，David A. Patterson，
   机械工业出版社. ISBN: 978-7-111-63110-1,
   出版日期：2019年7月。

Teaching Program

Course Code: 21191061

Course Name: Computer Architecture

Weekly Hours: 2.5-2.0 **Credits:** 3.5

Teaching Goal and Basic Requirements:

Teaching Objectives：

To learn the basic concepts and principles of computer architecture. To know how to design a computer using the quantitative approaches. To learn the methods evaluating the performance of a computer system. We learn this course to understand the principles of modern pipelined processors and to implement a dynamic scheduling pipelined CPU, including pipelined CPU supporting multi-cycle operations, dynamic scheduling pipeline. Students are expected to learn the memory hierarchy and methods to improvement cache performance, memory performance. Understanding the hardware and software techniques to explore the ILP、DLP and TLP.

Content of Courses & Hours Allocation:

Ch0: Course Introduction （1学时）

Ch1：Fundamentals of Quantitative Design and Analysis（5学时）

1.1 Introduction
1.2 _Classes of Computers
1.3 Defining Computer Architecture
1.4 _Trends in Technology
1.5 Trends in Power and Energy in Integrated Circuits
1.6 Trends in Cost
1.7 Dependability
1.8 Measuring, Reporting, and Summarizing Performance
1.9 Quantitative Principles of Computer Design
1.10 _ Putting it All Together: Performance, Price and Power
1.11 Fallacies and Pitfalls
1.12 Concluding Remarks

Ch2：Memory Hierarchy Design（5学时）

2.1 Introduction
2.2 Memory Technology and Optimizations
2.3 Ten Advanced Optimizations of Cache Performance + App B.3
2.4 Virtual Memory and Virtual Machines
2.5 Cross-Cutting Issues: The Design of Memory Hierarchies
2.6 Putting it together: Memory Hierarchies in the ARM Cortex-A53
2.7 Fallacies and Pitfalls
2.8 Concluding Remarks: Looking Ahead

Ch3: Instruction-Level Parallelism and Its Exploitation （16 学时）

3.0 exception, extend pipeline to support multicycle operations (App C.4-C.6)
3.1 Instruction-Level Parallelism: Concepts and Challenges
3.2 Basic Compiler Techniques for Exposing ILP
3.3 Reducing Branch Costs with Advanced Branch Prediction (+ App C.2)

3.4 Overcoming Data Hazards with Dynamic Scheduling (+ App C.7)
3.5 Dynamic Scheduling: Examples and the Algorithm
3.6 Hardware-based Speculation
3.7 Exploring ILP using Multiple Issue and Static Scheduling
3.8 Exploring ILP using Dynamic Scheduling, Multiple Issue, and Speculation
3.9 Advanced Techniques for Instruction Delivery and Speculation
3.10 Cross-Cutting Issues
3.11 Multithreading: Exploiting Thread-Level Parallelism to Improve Uniprocessor Throughput
3.12 Putting It All Together：The Intel Core i7 6700 and ARM Cortex-A53 and Kunpeng pipeline technology
3.13 Fallacies and Pitfalls
3.14 Concluding Remarks： What’s Ahead

Ch4： Data-Level Parallelism in Vector, SIMD, and GPU Architectures （8 学时）
4.1 Introduction
4.2 Vector Architecture
4.3 SIMD Instruction Set Extensions for Multimedia
4.4 Graphic Processing Units
4.5 Detecting and Enhancing Loop-level Parallelism
4.6 Cross-Cutting Issues
4.7 Putting it All Together: Embedded Versus Server GPUs and Tesla Versus Core i7
4.8 Fallacies and Pitfalls
4.9 Concluding Remarks

Ch5： Thread-Level Parallelism（9 学时）
5.1 Introduction
5.2 Centralized Shared-Memory Architecture
3.15 Performance of Symmetric Shared-Memory Multiprocessor
3.16 Distributed Shared-Memory and Directory-Based Coherence
3.17 Synchronization: The Basics
3.18 Models of Memory Consistency: An Introduction
3.19 Cross-Cutting Issues
3.20 Putting It All Together: Multicore Processors and Their Performance
3.21 Fallacies and Pitfalls
3.22 The Future of Multicore Scaling
3.23 Concluding Remarks

Lab assignments:
1. Enhance comprehension of principles of High level Pipelined CPU.

2. Implement the following labs using Verilog on FPGA board.
    1) Implement pipelined CPU with forwarding paths and prediction-not-taken supporting RISC V 32i instructions.
    2) Implement Interruption and Exception on Pipelined CPU of Lab 1).
    3) Implement a 2-way associative cache
    4) Adding the cache into the pipelined CPU in Lab 2).
    5) Expend the pipelined CPU to support multi-cycle operations: integer multiplier, integer divider / remainder operation, issue in order and complete out of order, detecting pipeline hazards (WAW, RAW)
    6) Implement the Dynamic Scheduling (Scoreboard or Tomasulo), implement a dynamic scheduling pipelined CPU.

Grading Policy：

Final Grade =Performance * 60% + Final exam * 40%
Performance = Homeworks、PoP Quiz(28%) + Labs(32%) +Bonus(<=10%)
Max(Performance) = 60
Labs = Lab1(3weeks, 6%) + Lab2(2weeks, 4%) + Lab3 (1weeks, 3%) + Lab4(2weeks, 4%) + Lab5 (3weeks, 7%) + Lab6 (5weeks, 8%)

Teaching Plan:

1. Lecturing in Multimedia classroom using PPT.
2. At least two pop quizzes.
3. Homework for each chapter, total 20 hours.
4. Lab assignments, about 32 credit hours.
5. Course website at course.zju.edu.cn, providing learning materials.
6. Encouraging self-study, giving bonus to additional lab assignments.

Recommended Textbooks and Other References: (books, editors, publishing company, publishing time)

《Computer Architecture-A Quantitative Approach》 6th Edition，
John L. Hennessy，David A. Patterson,
China Machine Press.
ISBN: 978-7-111-63110-1,
July 2019.

课程简介：

《计算机体系结构》是计算机专业的一门重要专业课，从计算机系统角度介绍计算机设计原理和设计方法。课程内容包括：1）计算机体系结构基本概念、计算机设计生命周期、量化设计原理和计算机性能评价方法；2）存储体系改进性能的方法，包括 cache 的提高性能的方法、存储器的性能提升、虚拟存储器和虚拟机；3）开发指令级并行性提高性能：转移预测、

动态调度、猜测执行、多发射处理器等；4）利用数据级并行性提高性能：向量处理器 、SIMD 以及 GPU；5）利用线程级并行性提高性能：多处理器结构、Cache 一致性以及同步。在学习理论同时，要求掌握硬件设计工具和环境，在 Vivado 环境下，用 Verilog 语言进行硬件设计实现，并在 FPGA 板上验证正确性。实验包括：实现支持 RISC V 32i 指令的带 forwarding 和 Predict-not-taken 的流水线 CPU、支持精确中断、实现 2 路组关联 Cache 并与应用于指令流水线；进一步扩展流水线以支持多周期的复杂操作；最终实现动态调度的流水线 CPU。

This course is one of the most important professional courses in computer science that systemically introduce the fundamental concepts and design approaches of computer architecture from the view of the whole computer system. The topics cover the following contents. 1) Introduce fundamental concepts, lifecycle of computer design, quantitative principles, and performance evaluation. 2) Introduce the improvement approaches of memory hierarchy, including optimizations of Cache performance, memory technology and optimizations, virtual memory and virtual machine. 3) Explore the ILP to improve CPU performance via branch prediction, dynamic scheduling, speculation and multiple issue. 4) Explore the DLP to improve CPU performance via vector, SIMD and GPU architecture. 5) Explore the TLP to improve CPU performance introducing multiple processor architecture, cache coherence, and synchronization. At the same time, the students are required to master the hardware design approaches and skillfully use hardware design toolkits, using Verilog in Vivado environment and testing on FPGA board. The lab assignments include implement a pipelined CPU supporting all RISC V 32i instructions with forwarding paths and predict-not-take, adding precise interruption, implementing a 2-way associative cache and applied in CPU pipeline, extending integer CPU pipeline to support multi-cycle operations such as multiplier and divider, and finally a dynamic scheduling CPU pipeline.