课程大纲

COURSE SYLLABUS

1.	**课程代码/名称** **Course Code/Title**	CSE5030 高级计算机体系结构 Advanced Computer Architecture
2.	**课程性质** **Compulsory/Elective**	Compulsory
3.	**开课单位** **Offering Dept.**	Department of Computer Science and Engineering 计算机科学与工程系
4.	**课程学分/学时** **Course Credit/Hours**	3/64
5.	**授课语言** **Teaching Language**	English 英文
6.	**授课教师** **Instructor(s)**	Fengwei Zhang, Associate Professor, Department of Computer Science and Engineering 张锋巍，副教授，计算机科学与工程系
7.	**开课学期** **Semester**	Spring 春季
8.	**是否面向本科生开放** **Open to undergraduates or not**	No
9.	**先修要求** **Pre-requisites**	（如面向本科生开放，请注明区分内容。 If the course is open to undergraduates, please indicate the difference.） 请写本科课程课号、课名 Principles of Computer Organization CS202 计算机组成原理 Operating Systems CS302 计算机操作系统

10. 教学目标
Course Objectives

（如面向本科生开放，请注明区分内容。 If the course is open to undergraduates, please indicate the difference.）

1. Understand the fundamental principles, performance evaluation methods, and design trade-offs in modern computer architecture;
2. Analyze the RISC-V ISA and the XiangShan microarchitecture;
3. Develop the ability to read and understand top-tier architecture conference papers (e.g., ISCA, MICRO, HPCA, ASPLOS);
4. Design, simulate, or optimize a high-performance processor or accelerator module through hands-on projects;
5. Understand emerging trends in architecture, including system security and AI acceleration.

1. 掌握现代计算机体系结构设计的基本原理、性能评估方法与权衡思维；
2. 理解并分析 RISC-V 指令集架构与香山微体系结构设计；
3. 具备阅读并理解 ISCA、MICRO、HPCA、ASPLOS 等顶会论文的能力；
4. 能够通过实践项目设计、模拟或优化一个高性能处理器或加速器模块；
5. 理解体系结构在系统安全、AI 计算等的最新发展趋势。

11. 教学方法
Teaching Methods

（如面向本科生开放，请注明区分内容。 If the course is open to undergraduates, please indicate the difference.）

Lectures + Paper Reading & Presentation (top-tier papers) + Lab;
XiangShan open-source architecture lab: simulate microarchitectural components (e.g., cache, predictor, fetch unit) using RISC-V simulators and tools like GEM5;

Paper Reading & Presentation: Students work in groups to complete a microarchitecture design or optimization project, submit a technical report, and give a presentation.
课堂讲授 + 论文阅读与演示 + 实验；
香山开源架构实验：使用开源 RISC-V 模拟器与香山代码仓库，修改或扩展香山微体系结构（如 cache、预测器或取指单元），利用 GEM5 等工具进行性能分析；
论文阅读与演示：学生阅读并评析计算机体系结构顶级会议论文（如 ISCA、MICRO、HPCA、ASPLOS），并在课堂上进行展示，概述研究问题、核心思路、评估方法及后续可能的研究方向。

12. 教学内容
Course Contents
（如面向本科生开放，请注明区分内容。 If the course is open to undergraduates, please indicate the difference.）

Section 1	第一周：课程导论与体系结构基础 • 计算机体系结构的设计与分析基础 • 定量分析原理（Amdahl 定律、CPU 性能公式） • 各种性能评估基准程序（Benchmarks）介绍 • 体系结构发展趋势（从单核到多核及异构） [Lab 1] RISC-V 环境搭建与基准测试。安装完整的 RISC-V 软件栈，运行基准 RV 程序以验证环境有效性。 Week 1. **Introduction & Architecture Fundamentals** o Design and analysis principles of computer architecture o Quantitative principles (Amdahl's Law, CPU performance equation) o Benchmarks and performance evaluation o Emerging trends. [Lab 1] Install RISC-V software stack and execute a baseline RV program.
Section 2	第二周：指令集体系结构（ISA）设计 • 指令集分类（RISC vs CISC） • 寻址模式与指令格式 • RISC-V 指令集架构详解 • 操作数与过程调用约定 [Lab 2] ISA 模拟与执行。分别用汇编和 C 语言编写 RV64 应用程序，使用 RISC-V 工具链编译并在 Verilator 上执行，观察取指-译码-执行的语义流程。 Week 2. **Instruction Set Architecture (ISA)** o ISA Classification (RISC vs. CISC) o Addressing Modes and Instruction Formats o RISC-V ISA Details o Operands and Procedure Call Conventions [Lab 2] Write RV64 applications in Assembly/C, compile, and execute on Verilator to observe instruction semantics.
Section 3	第三周：存储层次结构：缓存设计 • 局部性原理（时间与空间） • 缓存映射策略（直接映射、全相联、组相联） • 缓存替换算法与写策略

	• 缓存性能评估与优化技术 [Lab 3] 缓存性能分析。在 Gem5 中构建 CPU 模型，扫描 L1/L2 Cache 大小与相联度，使用 perf stat 量化加速比与缺失率的关系。 Week 3. **Cache and Memory Hierarchy** o Principle of locality o Cache mapping strategies o Replacement algorithms, and write policies o Average Memory Access Time analysis [Lab 3] Build CPU model in Gem5, sweep cache parameters, and quantify speed-up vs. miss rate.
Section 4	第四周：流水线技术基础 • 经典五级流水线设计 • 流水线冒险（结构冒险、数据冒险、控制冒险）及其处理机制 • 流水线性能分析 [Lab 4] 流水线冒险可视化。构造包含强制数据或控制冒险的 RISC-V 代码片段，在 Verilator 仿真环境中运行，以直观展示停顿现象。 Week 4. **Pipelining Basics** o Classical 5-stage pipeline o Pipeline hazards (Structural, Data, Control) o Pipeline performance [Lab 4] Create hazard-triggering code, run on Verilator, and dump pipeline traces to visualize stalls.
Section 5	第五周：高级流水线 • 动态指令调度 • 乱序执行 • 硬件预取技术原理 [Lab 5] 预取器设计与评估。为 Gem5 集成预取器套件，使用 perf 测量 IPC 提升，并找出不同负载下的最优预取深度。 Week 5. **Advanced Pipelining** o Dynamic instruction scheduling o Out-of-Order execution principles o Hardware prefetching techniques [Lab 5] Integrate prefetcher suite into Gem5, measure IPC improvement, and optimize prefetch depth.
Section 6	第六周：分支预测技术 • 静态与动态分支预测 • 局部分支预测与全局分支预测 • 锦标赛预测器与 TAGE 预测器 • 分支目标缓冲（BTB）与返回地址栈（RAS）

	[Lab 6] 使用 perf record / gem5 统计将误判率与前端停顿周期进行关联分析 Week 6. **Branch Prediction** o Static vs. Dynamic prediction o Local, Global prediction o Tournament predictors o BTB and RAS. [Lab 6] Correlate misprediction ratios with front-end stall cycles using simulation stats.
Section 7	第七周：多核缓存一致性 • 共享存储多处理器架构 • 缓存一致性协议 • MSI、MESI、MOESI 协议详解 • 伪共享（False Sharing）问题 [Lab 7] 缓存一致性操作实践。使用 NANHU 架构的 CBO 指令集进行缓存行的清洗/作废操作，确保读取核心总能获取最新数据，验证一致性机制。 Week 7. **Cache Coherence** o Shared-memory multiprocessors o Cache coherence protocol o MSI、MESI、MOESI o False sharing [Lab 7] Use CBO instructions to clean/invalidate lines and verify data coherence between cores.
Section 8	第八周：内存一致性模型 • 顺序一致性 • 全存储定序（TSO） • 弱一致性模型（Release Consistency, RISC-V WMO） • 存储屏障（FENCE）与同步机制 [Lab 8] 内存模型验证。运行 litmus-tests-riscv 测试套件，验证 RVWMO（RISC-V Weak Memory Ordering）模型行为，并分析和绘制测试结果。 Week 8. **Memory Consistency** o Sequential Consistency o TSO o RISC-V Weak Memory Ordering o Synchronization and Fences [Lab 8] Run litmus-tests-riscv suite to verify the RVWMO model and analyze results.
Section 9	第九周：高性能开源处理器架构实例（香山） • 香山微架构深度解析 • 敏捷硬件开发流程

	[Lab 9] 完整体验敏捷开发流程。 Week 9. **XiangShan Microarchitecture Case Study** o Deep dive into XiangShan o Agile hardware development flow [Lab 9] Experience the agile flow.
Section 10	第十周：向量体系结构 • 向量指令集设计理念 • RISC-V "V" Extension • SIMD 与多媒体指令扩展 [Lab 10] 向量编程实践。手工编写 RVV 向量加法内核。 Week 10. **Vector Architecture** o Vector ISA design o RISC-V "V" Extension o SIMD [Lab 10] Hand-code an RVV vector, and analyze program results.
Section 11	第十一周：并行与异构计算 • 线程级并行 • GPU 体系结构原理 • 异构计算编程模型 [Lab 11] GPGPU 仿真实验，并在 Verilator 仿真的 GPU 模型上成功运行高斯消元。 Week 11. **Parallel and Heterogeneous Computing** o Thread-Level Parallelism o GPU Architecture o Heterogeneous Computing Programming Model. [Lab 11] Compile OpenCL Gaussian elimination kernel and run on a simulated GPU.
Section 12	第十二周：安全体系结构 • 硬件安全漏洞 • 可信执行环境 • 领域专用加速器安全 [Lab 12] 可信执行环境部署。在模拟器上启动 RISC-V 可信执行环境 “Coffer”，理解安全隔离机制。 Week 12. **Secure Architectures** o Hardware security vulnerabilities o Trusted Execution Environment o DSA Security [Lab 12] Boot RISC-V TEE “Coffer” on the emulator.
Section 13 - 16	第十三到十六周：前沿研究论文阅读与项目展示

		• 顶级会议论文选读（ISCA, MICRO, HPCA, ASPLOS） • 最终答辩 Week13-16: **Research Paper Reading & Presentation** o Reading top-tier conference papers o Final presentations and Q&A

13. 课程考核
Course Assessment

（①考核形式 Form of examination；②.分数构成 grading policy；③如面向本科生开放，请注明区分内容。 If the course is open to undergraduates, please indicate the difference.）

Class Participation & Discussion: 10%
Paper Reading & Presentation: 20%
12 intensive Lab Assignments: 60%
Final Examination: 10%
课堂参与与讨论：10%
论文阅读与演示：20%
12 个高强度的实验作业：60%
期末考试：10%

Week 1 Course Introduction & Architecture Evaluation
• Install the complete RISC-V software stack (toolchain, drivers, simulation libraries) and execute a baseline RV program to validate the environment.

Week 2 Instruction-Set Architecture (ISA)
• Write a simple RV64 application in both assembly and C, compile it with the RISC-V toolchain, and execute it on Verilator to observe the fetch-decode-execute semantics.

Week 3 Caches & Memory Management
• Build an out-of-order CPU model in gem5, sweep L1/L2 cache size and associativity, and use perf stat to quantify the relationship between speed-up and miss rate.

Week 4 Pipelining
• Create a RISC-V code snippet that forces a data or control hazard, run it on the XiangShan Verilator, and dump the pipeline trace to illustrate the stall.

Week 5 Advanced Pipelines
• Integrate a prefetcher suite (stride, GHB, bingo) into gem5, use perf to measure IPC improvement, and identify the optimal prefetch depth.

Week 6 Branch Prediction
• Repeat the prefetch sweep from Week 5 to decouple its effect from branch misprediction cost; correlate misprediction ratios with front-end stall cycles using perf record / gem5 stats.

Week 7 Cache Coherence
• Use NANHU's CBO instruction set to clean/invalidate lines, ensuring a reader core always fetches the latest data.

Week 8 Memory Consistency
• Run the litmus-tests-riscv suite to verify the RVWMO model and plot the results.

Week 9 Xiang-Shan Micro-Architecture
• Experience the agile development flow: clone XiangShan, generate Verilog, launch NEMU as a reference, and ensure successful Linux boot through difftest.

Week 10 Vector ISA
• Hand-code an RVV vector addition kernel, compile it with LLVM, and analyze the program results.

Week 11 Parallel & Heterogeneous Computing
• Compile an OpenCL Gaussian elimination kernel with LLVM and successfully run it on a Verilator-simulated GPU.

Week 12　Secure Architectures
• Boot the RISC-V Trusted Execution Environment “Coffer” on NEMU.
Weeks 13–16　Research Paper Reading & Presentation
• Form groups, select one paper each from ISCA/MICRO/HPCA/ASPLOS, prepare a 15-min conference-style slide deck, and lead a 10-min Q&A on novelty, overhead, and reproducibility.

Week 1　课程导论与体系结构评估
• 安装完整的 RISC-V 软件栈（工具链、驱动、仿真库），运行基准 RV 程序以验证环境有效性。
Week 2　指令集体系结构（ISA）
• 用汇编和 C 各写一个 RV64 应用，使用 RISC-V 工具链编译并在 Verilator 上执行，观察取指-译码-执行语义。
Week 3　缓存与内存管理
• 在 gem5 中构建乱序 CPU 模型，扫描 L1/L2 大小与相联度，用 perf stat 量化加速比与缺失率的关系。
Week 4　流水线
• 构造一条 RISC-V 片段强制数据或控制冒险，在香山 Verilator 上运行并转储流水线轨迹以展示停顿。
Week 5　高级流水线
• 为 gem5 集成预取器套件（stride、GHB、bingo），用 perf 测量 IPC 提升，找出最优预取深度。
Week 6　分支预测
• 重复 Week 5 的预取扫描以分离其影响与分支误判代价；用 perf record / gem5 统计将误判率与前级停顿周期关联。
Week 7　缓存一致性
• 使用 NANHU 的 CBO 指令集清写/作废缓存行，确保读核心总能获取最新数据。
Week 8　内存一致性
• 运行 litmus-tests-riscv 套件检验 RVWMO 模型，绘制结果图。
Week 9　香山微架构
• 完整体验敏捷流程：克隆香山、生成 Verilog、启动 NEMU 作为参考，并通过 difftest 确保 Linux 成功启动。
Week 10　向量指令集
• 手工编写 RVV 向量加内核，使用 LLVM 编译，分析程序结果。
Week 11　并行与异构计算
• 用 LLVM 编译 OpenCL 高斯消元内核，在 Verilator 仿真的 GPU 成功运行。
Week 12　安全体系结构
• 在 NEMU 上启动 RISC-V 可信执行环境 “Coffer”
Weeks 13–16　研究论文阅读与展示
• 学生分组，从 ISCA/MICRO/HPCA/ASPLOS 中各选一篇论文，制作 15 分钟会议风格幻灯片，并就创新性、开销与可复现性主持 10 分钟问答。

14.　教材及其它参考资料
Textbook and Supplementary Readings

Main Textbooks:
1. John L. Hennessy & David A. Patterson, Computer Architecture: A Quantitative Approach, 6th Edition, 2019.
- NOTE: 6th edition (and 6.5900) uses RISC-V as the main ISA; prior editions (and 6.5900/6.823 before Fall 2023) use MIPS

- Used at MIT 6.5900 / 6.004, Stanford, Berkeley and CMU.

Supplementary:
- XiangShan Project Docs & Code: https://github.com/OpenXiangShan
- Modern Processor Design by Shen & Lipasti
- Parallel Computer Architecture by David Culler et al.
- Top-tier architecture conference papers (ISCA, MICRO, HPCA, ASPLOS)

主要教材：
John L. Hennessy & David A. Patterson, Computer Architecture: A Quantitative Approach, 6th Edition, 2019.
注：第 6 版（以及 MIT 6.5900）以 RISC-V 为主要指令集架构；此前版本（2023 年秋季前的 6.5900/6.823）使用 MIPS。
用于 MIT 6.5900/6.004、斯坦福、伯克利和卡内基梅隆等课程。

补充资料：
香山项目文档与源码：https://xiangshan.github.io/
Modern Processor Design by Shen & Lipasti
Parallel Computer Architecture by David Culler et al.
计算机体系结构顶级会议论文（ISCA、MICRO、HPCA、ASPLOS）