



非常天空

Computer Organization & Design



Hardware/Software interface

楼学庆

浙江大学计算机学院

<http://10.214.47.99/>

[Email:hzlou@163.com](mailto:hzlou@163.com)



玉泉校区曹光彪东楼507室



17:58:11

浙江大学计算机学院

联系方式

- 网站:

- <http://10.214.47.99>

- 邮箱:

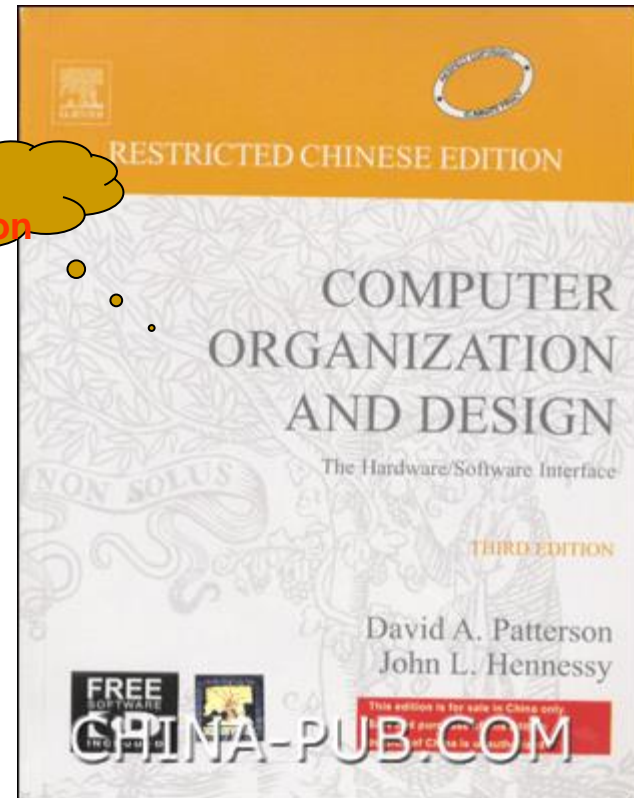
- hzlou@163.com (不收作业)



Textbook



2nd
Edition



3rd
Edition

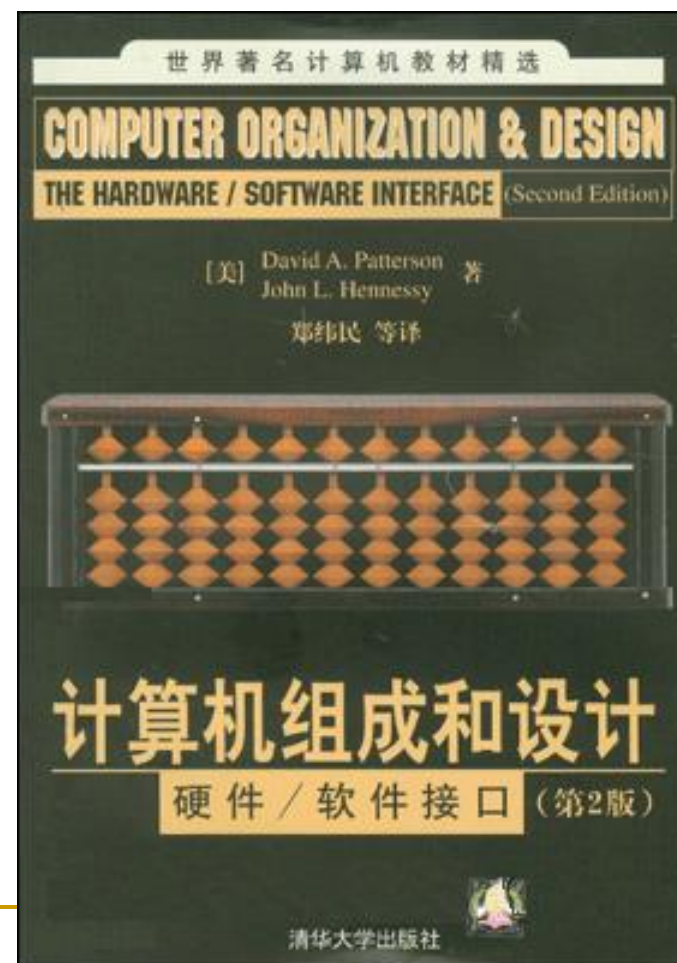
Reference

- 《计算机组成与设计》
 - 潘雪增、平玲娣
 - 浙江大学出版社
 - ISBN 7-308-03523-9/TP.25



Reference

- 《计算机组成和设计》 硬件 / 软件接口（第2版）
 - 作者：郑纬民译
 - ISBN: 9787302069010
 - 定价：76元





Course outline

- Name:

Computer Organization & Design

- Students:

Undergraduate students in Computer or some others department.

- Score : 4.5

- Hours/week: 3.5-2

- Total: 88 hours

教学内容研究性

- 没有必然，只有更好
 - 尽力寻求、实现各种可能的方法
 - 比较、评价各种不同的方法 (现行方法的优点、特点)
 - 相对完整实现所选择的方法
- 理论与实际相结合
 - 理论学习
 - 软件模拟 (C语言编程)
 - 硬件实验 (FPGA设计)

成绩 (精研班)

- 考试: 30% (不低于30分)
 - 英文
 - 闭卷
- 平时, 五部分14知识点:
 - 每个知识点: 5%, 总共70% (雷同0分)
- 额外:
 - 课堂练习 (+5%): 每题1分
- QQ: 25534460

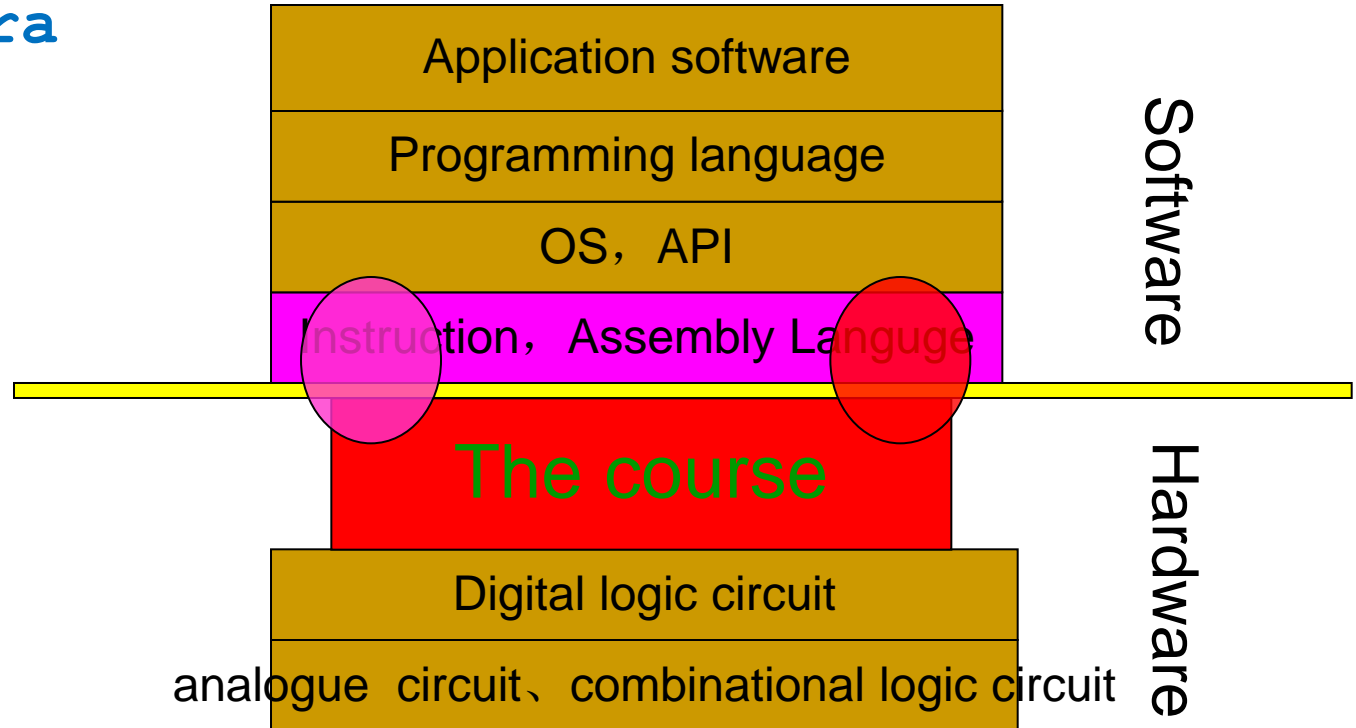
小班化教学

- 分组学习制
 - 四人一组组长负责
 - 定期汇报
 - 随时交流
- 按知识点教学，每个知识点：
 - 课堂讲解(多媒体)
 - 讨论、布置练习
 - 演示、讲解
 - 相互评价

The Course

■ Prerequisites

- C Program Language
- Digital Logic
- Algebra



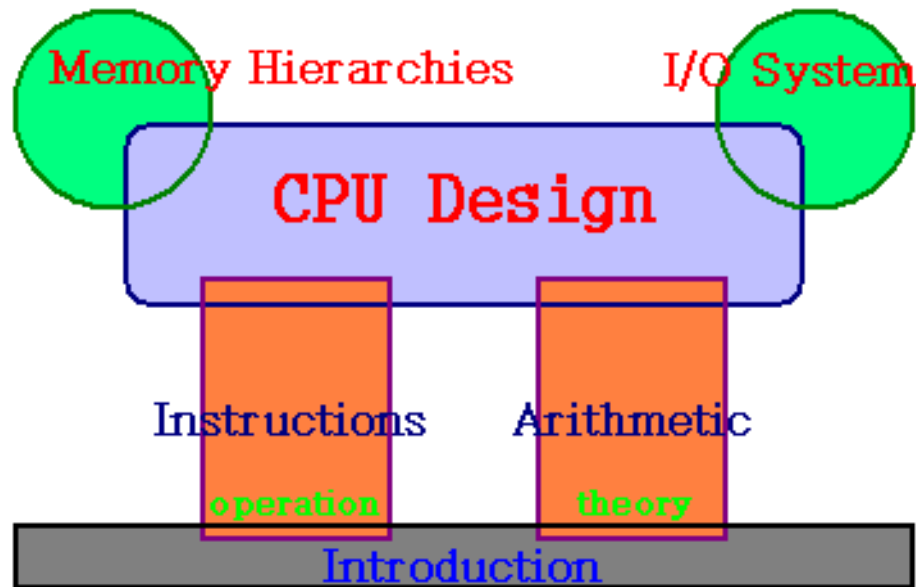


《计算机系统原理》 课程目标

- 利用一个学期，覆盖《逻辑与计算机设计》、《计算机组成》、《体系结构》与《汇编与接口》等计算机硬件系统系列主要课程。作为非计算机技术方向，学习了解掌握计算机硬件、计算机系统方面知识的主要课程。
 - ☆前导课程：《C语言程序设计》
- 强烈建议不是只想玩软件的同学，改选 《计算机组成》！



课程结构

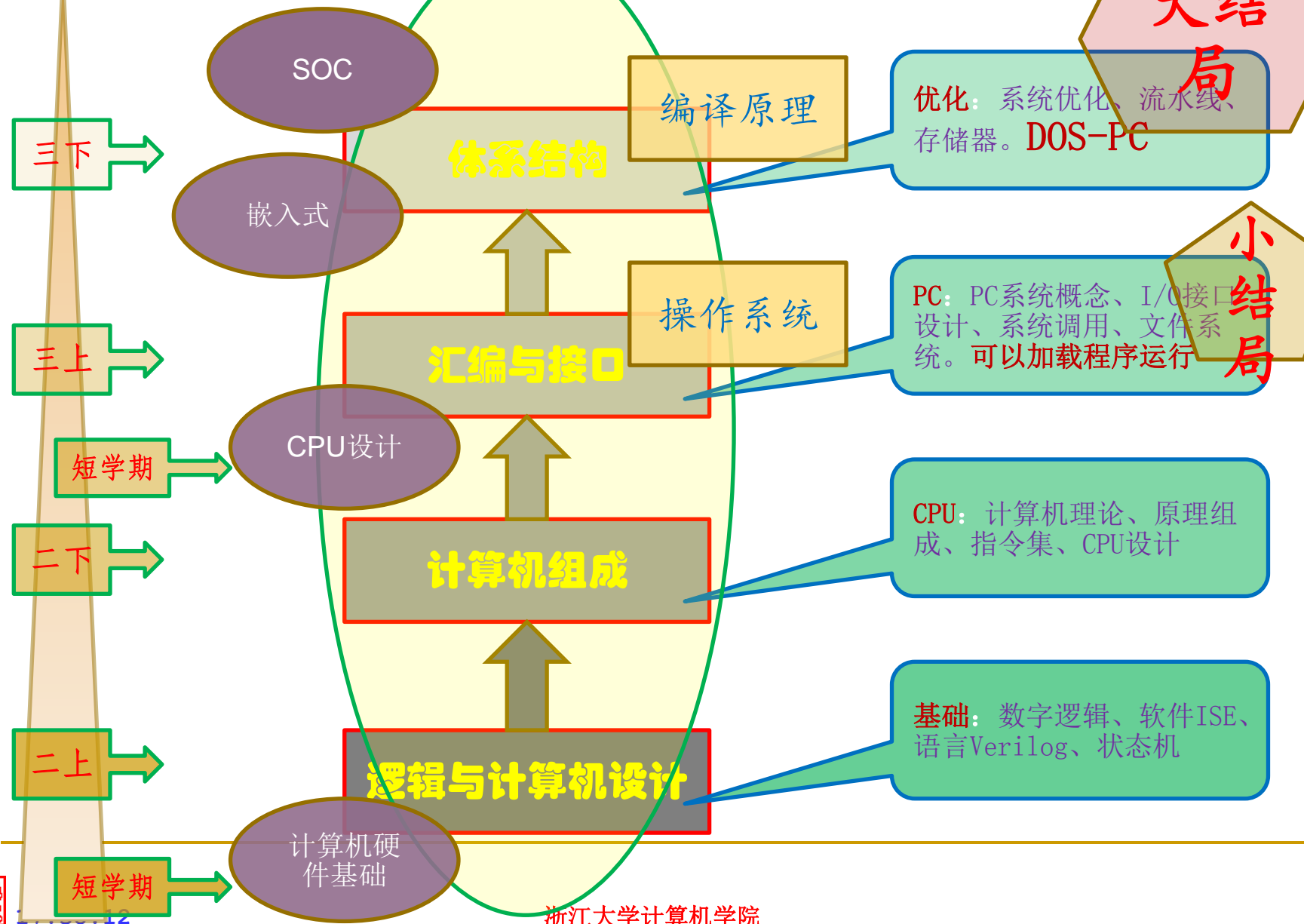




计算机系统能力培养系列课程体系

时间轴

主干线





我有一壺酒
足以慰風塵
計組一堂課
匯編有緣人

你我相約混一年，**約**嗎？

课程计划

- 每学期提供课程计划，
- 确定计算机系统设计的总目标。
- 给出实现的路线图。

ZPC 设计路线图之春夏时节

*: 春夏《计算机组成》以设计运算器 ALU、CPU 为主。

| 周 | 章节 | 内容 | 理论 | 实验 | 说明 | 周 |
|----|--------|-------|---------------|------------|--|----|
| 01 | 概论 | 吹牛 | 作 01 汉字键盘设计 | 实 01 运算器设计 | 1. 课堂实验当堂布置当堂完成。每次实验需画出逻辑图 2. 大设计时间、分工各组自行安排 3. 验收周提交设计大纲各自负责部分 4. 各大程汇总第 16 周由组里每人提交一个: 1) 运算器、 2) 单时钟、 3) 多时钟、 4) 模拟器(软件) | 01 |
| 02 | 数据表达 | 进制、整数 | 作 02 移/原/补码算法 | 实 02 | | 02 |
| 03 | | 算术运算 | 作 03 乘除法算法推导 | 实 03 | | 03 |
| 04 | | 汉字、浮点 | 作 04 浮点处理 | 实 04 | | 04 |
| 05 | 指令系统 | 指令 | 作 05 汇编器 | 实 05 | | 05 |
| 06 | | 子程序 | 作 06 反汇编 | 实 06 | | 06 |
| 07 | | 汇编编程 | 作 07 模拟执行 | 运算器验收 | | 07 |
| 08 | I/O 系统 | 显示原理 | 作 08 显示模拟 | 展示 | | 08 |
| 中 | | | | | | 中 |
| 09 | CPU 设计 | 单时钟 | 作 09 模拟器设计 | 实 09 单时钟设计 | | 09 |
| 10 | | 控制器 | 作 10 | 实 10 | | 10 |
| 11 | | 多时钟设计 | 作 11 | 实 11 | | 11 |
| 12 | 存储器 | 存储器 | 作 12 | 单时钟验收 | | 12 |
| 13 | | 虚拟存储 | 作 13 虚拟存储 | 实 13 多时钟设计 | | 13 |
| 14 | | Cache | 作 14 | 实 14 | | 14 |
| 15 | I/O 系统 | 数据传送 | 模拟器验收 | 多时钟验收 | | 15 |
| 16 | 复习 | | 综合展示 | 综合展示 | | 16 |
| 假 | | | | | | 假 |



系统模块表



《计算机组成》2015 课程计划

ZPC 软硬件模块一览表 (增减中)

总分 **3** 大块: PC 端软件、MIPS 软件、硬件 (基本组件、CPU、外设)。

| 序 | PC 端 | MIPS | 硬件 | | | 备注 |
|----|----------|--------|--------|--------|--------|----------|
| | | | 基本组件 | CPU | 外设 | |
| 01 | ☆补码表示 | 指令执行 | 简单 I/O | 寄存器组 | 数码管显示 | ☆: 需理论证明 |
| 02 | ☆整数加减 | VGA 模拟 | 多路选择器 | 单时钟 | 计数器 | |
| 03 | ☆整数乘法 | 图形模式 | 移位寄存器 | 多时钟 | 编码器 | |
| 04 | ☆整数除法 | 系统调用 | 译码器 | 微程序 | LCD | |
| 05 | ☆浮点表示 | BIOS | 逻辑运算 | 流水线 | 总线 | |
| 06 | ☆浮点运算 | 中断 | 加法器 | 总线 CPU | 地址译码 | |
| 07 | 汉字显示 | 磁盘系统 | 先行进位 | | VGA | |
| 08 | 汇编器 | 汇编 | 大小比较 | | 中断 | |
| 09 | 反汇编 | 反汇编 | 乘法器 | | 中断控制器 | |
| 10 | 程序汇编 | 编辑器 | 阵列乘法 | | 定时器 | |
| 11 | MIPS 模拟器 | | 除法器 | | 串口 | |
| 12 | 虚拟机 | | 存储器 | | 可编程串口 | |
| 13 | 虚拟存储 | | 不对齐读写 | | 并行输入输出 | |
| 14 | 串口通讯 | | 地址译码 | | 键盘 | |
| 15 | 虚拟盘 | | | | DDK | |
| 16 | | | | | 浮点运算 | |
| 17 | | | | | | |
| 18 | | | | | | |

1. 以上组件并不全在春夏实现。
2. 硬件组件可分单独与联调两种。应该先单独能够连通, 然后考虑接入你的系统。
3. 当你把这些组件一个个实现, 一个个加入你的系统, 你的 PC 就在眼前。

小班化教学

- 分组学习制：个人做研究，小组做产品
 - 四人一组，加强交流、分工与协作相结合。
 - 个人可以对专题作更广泛深入的研究，
 - 把相对成熟的、认为最佳的提交小组集成。
 - 避免简单重复工作。
 - 强调产品化的设计思路
- 教学方法：
 - 理论与实践相结合
 - 原理推导与程序模拟相结合
 - 逻辑设计与硬件实验相结合
 - 强调系统整体概念
 - 强调课程衔接

课程任务

■ 理论:

- 完成全书知识点学习。（详见大纲）

■ 软件:

- 完成各种算法证明，程序模拟；
- 编写MIPS汇编、反汇编程序；
- 完成一个MIPS模拟器，模拟调试MIPS指令执行。
- 编写虚拟磁盘程序；

■ 硬件:

- 完成运算器设计，包括整数算术运算、逻辑运算，浮点算术运算；
- 完成一个基于FPGA的有限指令CPU；
 - 单时钟、多时钟、微程序
- 中断处理、总线结构设计研究。
- VGA显示器、键盘的专用设计研究；

课程安排

| 周 | 章节 | 知识点 | 分 | 实验方式 | |
|----|-------|--------|---|------|----------------|
| 1 | 概论 | | | | 汉字键盘设计，熟悉C语言编程 |
| 2 | 数据表达 | 数据表示加减 | 5 | 程序模拟 | 用移码表示整数，实现加减运算 |
| 3 | | 乘除 | 5 | | 补码乘法设计模拟 |
| 4 | | 浮点数 | 5 | | 浮点处理或汉字 |
| 5 | 汇编语言 | 汇编 | 5 | | MIPS汇编到机器码 |
| 6 | | 反汇编 | 5 | | 机器码到MIPS汇编 |
| 7 | | 模拟CPU | 5 | | MIPS CPU模拟 |
| 8 | CPU设计 | 单个组件 | 5 | 硬件实验 | 交硬件实验报告 |
| 9 | | 单时钟 | 5 | | 交硬件实验报告 |
| 10 | | 多时钟 | 5 | | 交硬件实验报告 |
| 11 | 存储系统 | 存储器 | 5 | 平时 | (期中考试) |
| 12 | | Cache | 5 | | (平时机动) |
| 13 | | 虚拟存储 | 5 | 程序模拟 | 程序模拟虚拟存储 |
| 14 | I/O系统 | I/O | 5 | | 显示模拟 |
| 15 | | 综合 | 5 | | 综合演示 |
| 16 | 复习 | | | | |

实验设计

- 软件实验：
 - MIPS汇编
 - MIPS模拟器
- 硬件实验：
 - 运算器
 - 单时钟
 - 多时钟（组合逻辑与微程序）
- 小实验：每周作业，课堂实验
 - 软件：算法模拟
 - 硬件：部件实验

实验考核

- 大设计与课堂实验相结合。
 - 每次实验灵活布置与设计有关的小实验，
 - 大设计由小组自行安排时间。
- 个人分工与小组合作相结合
- 验收清单

Spartan-3:



非常天空

当激情演化成一种淡泊和豁达。
年轮里，

浙江大學
:25534460

今天:43 / 总共:1086612 2015-12-24 星期四 农历:乙未年冬月十三

计算机组成精研2015春夏

| ID | 队名 | 序 | 文档 | 具体 | 纸质 | 电子 | 备注 | |
|-----|------------------|---|----|---|--------------------------|-------------------------------------|-------------------------------------|------|
| 180 | 人生苦短 | 选 | 1 | 产品说明 <input type="checkbox"/> 名称, <input type="checkbox"/> 标志, <input type="checkbox"/> 封面设计 | <input type="checkbox"/> | <input type="checkbox"/> | 图片上传 | |
| 175 | 数据挖掘机 | 选 | 2 | 宣传彩页 | <input type="checkbox"/> | <input type="checkbox"/> | 双面A4, 上传 | |
| 169 | 欲穷千里目 | 选 | 3 | 实验报告 | | <input type="checkbox"/> | | |
| 168 | paranoid | 选 | 4 | 演示PPT | | <input type="checkbox"/> | | |
| 163 | 我歌月徘徊 | 选 | 序 | 模块名称 | 原理 | 模拟/纸质 | 实验代码 | 备注 |
| 160 | DuangDuangDuang~ | 选 | 1 | 与 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 158 | 一开始叫我组队其实我是拒绝的 | 选 | 2 | 或 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 146 | 更上一层楼 | 选 | 3 | 异或 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 143 | 4618 | 选 | 4 | 非 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 140 | 红蓝四侠 | 选 | 5 | 左右 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 133 | 救世组 | 选 | 6 | 移位算术 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 132 | 倔不宕壁 | 选 | 7 | 循环 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 128 | 天下皆白,唯我独黑 | 选 | 8 | 加减 <input type="checkbox"/> 进位, <input type="checkbox"/> 溢出, <input type="checkbox"/> 比较, <input type="checkbox"/> 零标志 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | 串行 |
| 123 | 折戟沉沙 | 选 | 9 | 整数 乘 <input type="checkbox"/> 符号, <input type="checkbox"/> 无符号, | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | 移位加 |
| 122 | 楼sir大法好 | 选 | 10 | 除 <input type="checkbox"/> 符号, <input type="checkbox"/> 无符号, | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | 恢复除数 |
| 120 | Domori | 选 | 11 | 加减 <input checked="" type="checkbox"/> 零值, <input checked="" type="checkbox"/> 无穷, <input checked="" type="checkbox"/> 非数, <input checked="" type="checkbox"/> 四舍五入 | | <input type="checkbox"/> | <input checked="" type="checkbox"/> | |
| 112 | 延毕小分队1队 | 选 | 12 | 浮点 乘 <input type="checkbox"/> 零值, <input type="checkbox"/> 无穷, <input type="checkbox"/> 溢出 | | <input type="checkbox"/> | <input type="checkbox"/> | |
| | | | 13 | 除 <input type="checkbox"/> 零值, <input type="checkbox"/> 无穷, <input type="checkbox"/> 溢出 | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | |

检查得分 记录

路上,不是组合逻辑就是时序电路。由

用[]电路实现时,先不要急用if,case

节,读出显示。系统按字节编址。

IFF.

学期提交

■ 刻写光盘可便于后续课程

春夏学期终极提交

*: 平时注意积累资料，期末按 **组** 统一刻盘。

| 号 | 材质 | 名称 | 时间 | 说明 | 备注 | 号 |
|----|-----------------------|----------|--------|------------------------------------|---------------|----|
| 01 | 纸质 | 运算器使用手册 | 第 06 周 | 设计彩色封面 (宣传册页可选) | | 01 |
| 02 | | CPU 使用手册 | 第 14 周 | 设计彩色封面 | | 02 |
| 03 | | CPU 宣传册页 | 第 14 周 | 单页彩色 A4 双面 | | 03 |
| 04 | 光盘 组 目录 | 运算器 | 第 06 周 | A. 实验报告 | 全组汇总, 每人负责一个。 | 04 |
| 05 | | 单时钟 | 第 11 周 | B. 所有代码 | | 05 |
| 06 | | 多时钟 | 第 14 周 | C. 使用手册 | | 06 |
| 07 | | 虚拟机 | 第 14 周 | D. 演示 PPT E. 其它 (如视频) | | 07 |
| 08 | 光盘 个人 目录 | 基本资料 | 第 14 周 | A. 个人介绍 B. 周记 C. 照片 D. 作业 | | 08 |
| 09 | | 课堂实验 | 第 14 周 | 个人课堂实验代码 | | 09 |
| 10 | | 运算器 | 第 06 周 | A. 个人部分所有代码 | | 10 |
| 11 | | 单时钟 | 第 11 周 | B. 个人部分实验报告 | | 11 |
| 12 | | 多时钟 | 第 14 周 | | | 12 |
| 13 | | 虚拟机 | 第 14 周 | | | 13 |
| 14 | | | | | | 14 |



非常天空

ZPC 2015
ZHEJIANG UNIVERSITY
浙江大学计算机学院

ZPC 2016
ZHEJIANG UNIVERSITY
浙江大学计算机学院

ZPC 2014
design
浙江大学计算机学院

ZPC 2017
ZHEJIANG UNIVERSITY
浙江大学计算机学院



ZPC 2018
ZHEJIANG UNIVERSITY
浙江大学计算机学院

CPU 2013
design
浙江大学计算机学院

CPU 2012
design
浙江大学计算机学院

CPU 2011
design
浙江大学计算机学院




三、ZPC之MIPS指令集

- 目录
- 一、前言
- 二、寄存器
 - § 2.1 通用寄存器
 - § 2.2 协处理器0
 - § 2.3 其它寄存器
- 三、指令系统
 - § 3.1 指令类型
 - § 3.1.1 R类型指令
 - § 3.1.2 I类型指令
 - § 3.1.3 J类型指令
 - § 3.1.4 C类型指令
 - § 3.2 指令格式
 - § 3.2.1 数据传送指令
 - § 3.2.2 算术运算指令
 - § 3.2.3 逻辑运算指令
 - § 3.2.4 移位指令
 - § 3.2.5 转移指令
 - § 3.2.6 协处理器指令
 - § 3.2.7 系统功能调用

| ZPC之MIPS汇编语言规范 | |
|---------------------|----|
| ZPC之MIPS指令集 | |
| 版本 0.8 | |
| 浙江大學计算机学院 | |
| 2015.07.15.Q. | |
| 目录 | |
| 一、前言 | 3 |
| 二、寄存器 | 4 |
| § 2.1 通用寄存器 | 4 |
| § 2.2 协处理器0 | 5 |
| § 2.3 其它寄存器 | 5 |
| 三、指令系统 | 6 |
| § 3.1 指令类型 | 6 |
| § 3.1.1 R类型指令 | 6 |
| § 3.1.2 I类型指令 | 6 |
| § 3.1.3 J类型指令 | 6 |
| § 3.1.4 C类型指令 | 7 |
| § 3.2 指令格式 | 7 |
| § 3.2.1 数据传送指令 | 7 |
| § 3.2.2 算术运算指令 | 9 |
| § 3.2.3 逻辑运算指令 | 13 |
| § 3.2.4 移位指令 | 15 |
| § 3.2.5 转移指令 | 17 |
| § 3.2.6 协处理器指令 | 20 |
| § 3.2.7 系统功能调用 | 20 |
| 四、伪指令 | 22 |
| 五、格式指令 | 26 |
| 六、表达式 | 29 |
| 七、例程 | 30 |
| 八、总结 | 32 |
| 九、参考 | 32 |
| 十、附录 | 33 |
| 附录一：GB2014 统一字符编码标准 | 33 |



| | | |
|---|--------------------|----|
| ■ | 四、伪指令..... | 22 |
| ■ | 五、格式指令..... | 26 |
| ■ | 六、表达式..... | 29 |
| ■ | 七、例程..... | 30 |
| ■ | 八、总结..... | 32 |
| ■ | 九、参考..... | 32 |
| ■ | 十、附录..... | 33 |
| ■ | 附录一：ZB2014统一字符编码标准 | |

| | |
|---|----|
|  | |
| ZPC之MIPS汇编语言规范 | |
| ZPC之MIPS指令集 | |
| 版本 0.8 | |
| 浙江大學计算机学院 | |
| 2015.07.15.Q. | |
| 目录 | |
| 一、前言..... | 3 |
| 二、寄存器..... | 4 |
| §2.1 通用寄存器..... | 4 |
| §2.2 协处理器0..... | 5 |
| §2.3 其它寄存器..... | 5 |
| 三、指令系统..... | 6 |
| §3.1 指令类型..... | 6 |
| §3.1.1 R类型指令..... | 6 |
| §3.1.2 I类型指令..... | 6 |
| §3.1.3 J类型指令..... | 6 |
| §3.1.4 C类型指令..... | 7 |
| §3.2 指令格式..... | 7 |
| §3.2.1 数据传送指令..... | 7 |
| §3.2.2 算术运算指令..... | 9 |
| §3.2.3 逻辑运算指令..... | 13 |
| §3.2.4 移位指令..... | 15 |
| §3.2.5 转移指令..... | 17 |
| §3.2.6 协处理器指令..... | 20 |
| §3.2.7 系统功能调用..... | 20 |
| 四、伪指令..... | 22 |
| 五、格式指令..... | 26 |
| 六、表达式..... | 29 |
| 七、例程..... | 30 |
| 八、总结..... | 32 |
| 九、参考..... | 32 |
| 十、附录..... | 33 |
| 附录一：ZB2014统一字符编码标准..... | 33 |
| 2/34 | |



网站建设

- 课程资料
- 参考资料下载
- 作业提交、批改

作业实例1：大汉键盘

■ 创意来源：

- 围棋、田字格、笔画

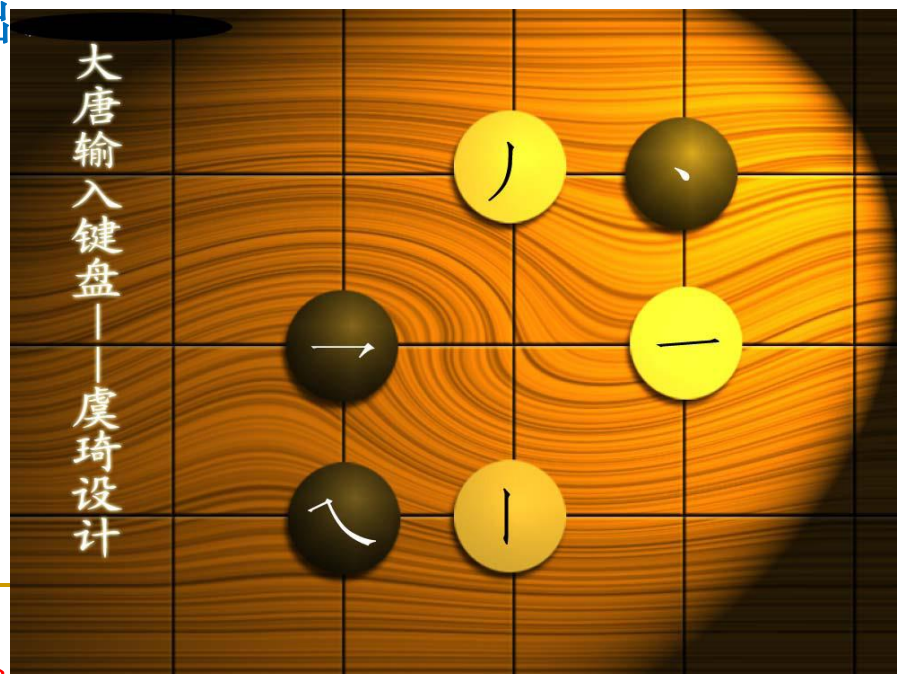
■ 输入方法：

- 1. 根据需输入要输入的汉字结构在田字格中摆放棋子。
- 2. 每摆放一次则屏幕响应位置出现该笔画。
- 3. 笔画组合到一定程度后，出现符合要求的待选汉字。
- 4. 选择你需要的汉字。

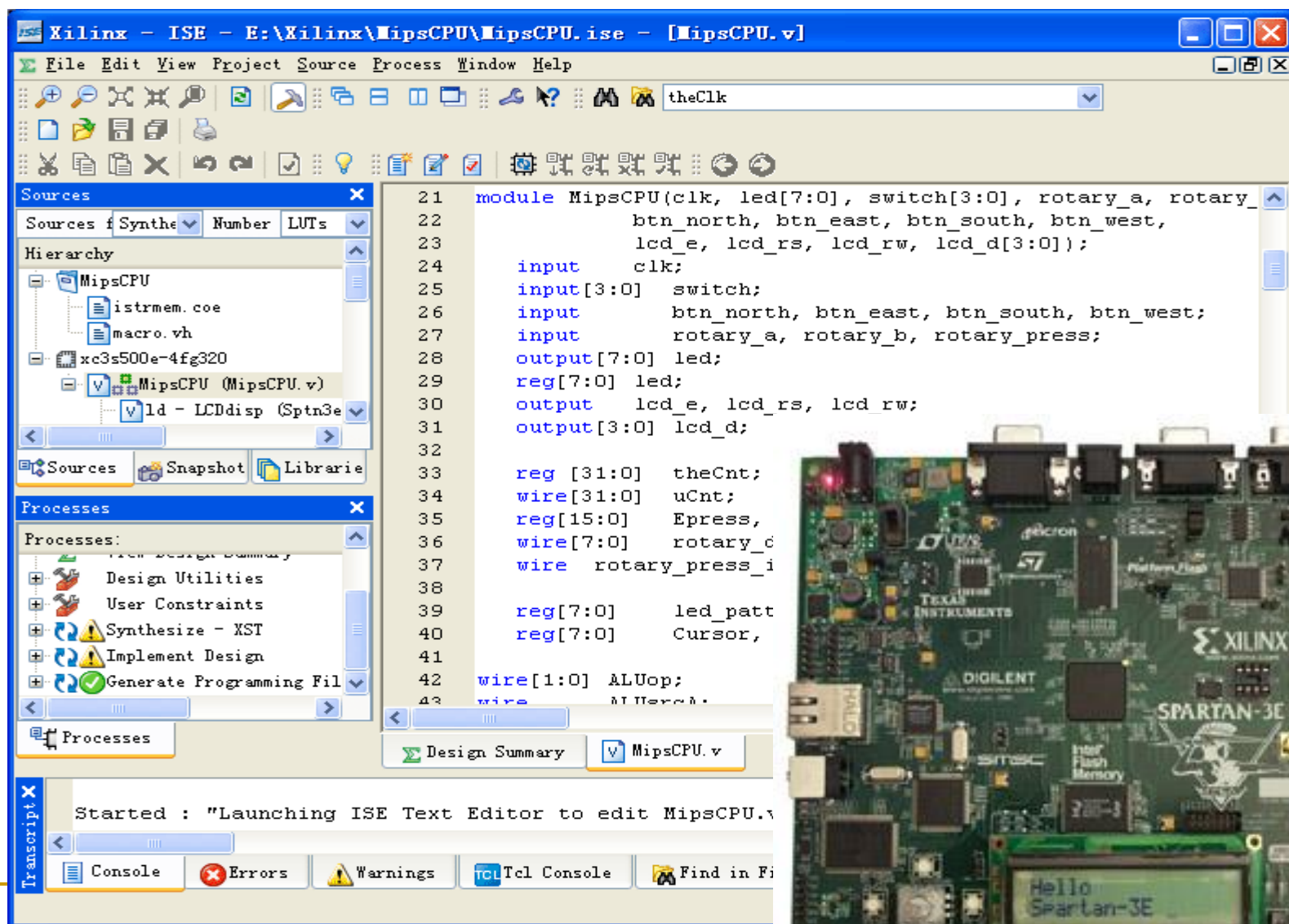
■ 实现条件：

- 传感棋盘、
- 识别软件、
- 汉字字库

**强调
设计！**



作业实例2: ISE



作业实例3：模拟器

 MIPS模拟器

MIPS

```

addi $t0, $t0, 3
addi $t6, $zero, 1
addi $s0, $s0, 4
add $t1, $t0, $t6
sub $t2, $zero, $t1
or $t3, $t0, $s0
ori $t4, $t0, 12
nor $t5, $t0, $t1
and $s1, $t0, $t1
slt $s2, $t0, $t1
addi $s4, $s4, 65
sw $s4, 5116($s3)
lw $s5, 5116($s3)

```

机器码

```

00100001000010000000000000000011
00100000000011100000000000000001
00100010000100000000000000000100
00000001000011100100100000100000
00000000000010010101000000100010
00000001000100000101100000100101
0011010100001100000000000001100
00000001000010010110100000100111
00000001000010011000100000100100
00000001000010011001000000101010
00100010100101000000000001000001
10100110011101000001001111111100
10001110011101010001001111111100

```

寄存器

| | |
|-----------|----------|
| \$zero: 0 | \$s0: 4 |
| \$at: 0 | \$s1: 0 |
| \$v0: 0 | \$s2: 1 |
| \$v1: 0 | \$s3: 0 |
| \$a0: 0 | \$s4: 65 |
| \$a1: 0 | \$s5: 65 |
| \$a2: 0 | \$s6: 0 |
| \$a3: 0 | \$s7: 0 |
| \$t0: 3 | \$t8: 0 |
| \$t1: 4 | \$t9: 0 |
| \$t2: -4 | \$k0: 0 |
| \$t3: 7 | \$k1: 0 |
| \$t4: 0 | \$gp: 0 |
| \$t5: -8 | \$sp: 0 |
| \$t6: 1 | \$fp: 0 |
| \$t7: 0 | \$ra: 0 |

1279 A

1311

1343

1375

1407

1439

1471

1503

1535

1567

1599

1631

1663

1695

1727

1759

1791

1823

1855

1887

1919

1951

1983

2015

显示器

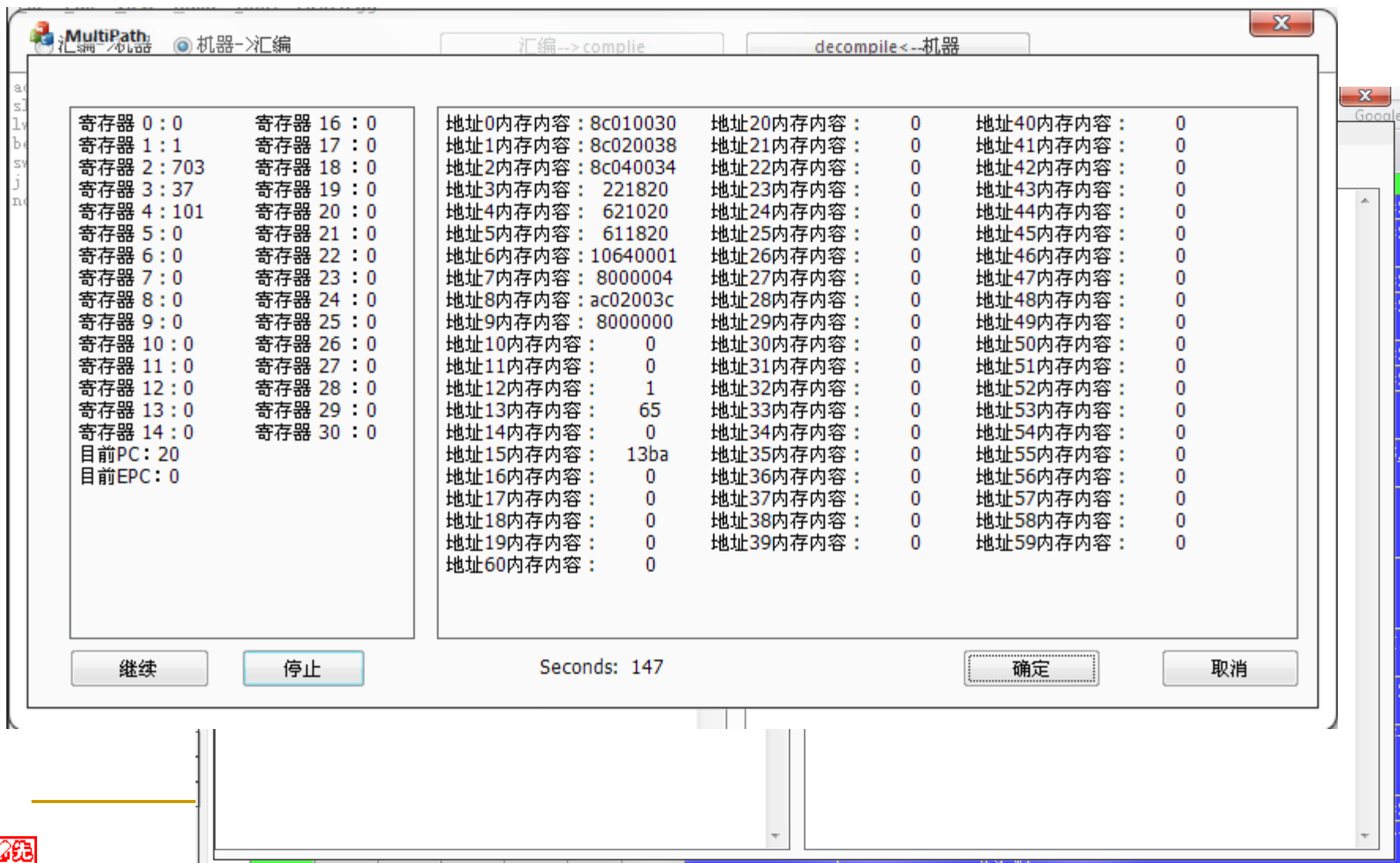
执行

退出

隐藏<<

清空

作业实例4：模拟器



The image shows a screenshot of the MultiPath simulator interface. The window has a title bar with 'MultiPath' and a menu bar with '汇编->汇编' and 'decompile<--机器'. Below the menu bar are two tabs: '汇编-->compile' and 'decompile<--机器'. The main area is divided into three sections: registers, memory, and a status bar.

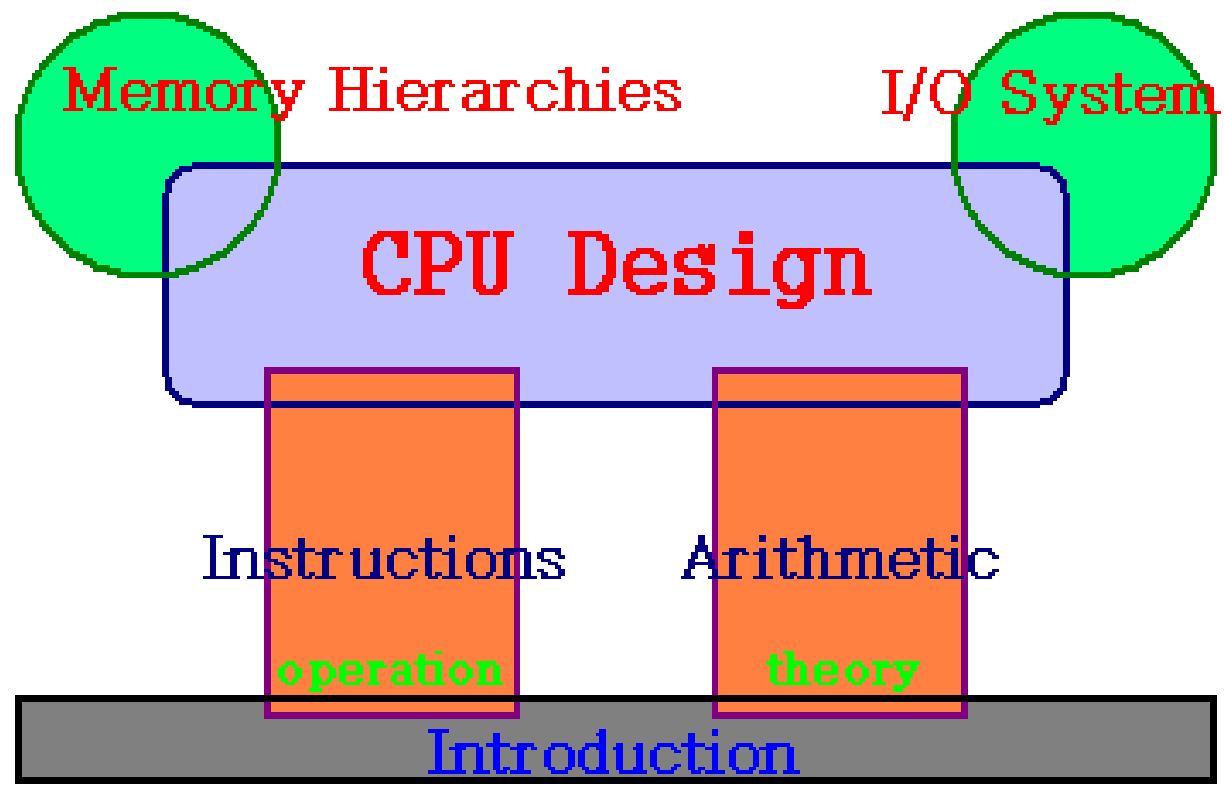
| 寄存器 | 值 | 地址 | 内存内容 |
|--------|-----|------|----------|
| 寄存器 0 | 0 | 地址0 | 8c010030 |
| 寄存器 1 | 1 | 地址1 | 8c020038 |
| 寄存器 2 | 703 | 地址2 | 8c040034 |
| 寄存器 3 | 37 | 地址3 | 221820 |
| 寄存器 4 | 101 | 地址4 | 621020 |
| 寄存器 5 | 0 | 地址5 | 611820 |
| 寄存器 6 | 0 | 地址6 | 10640001 |
| 寄存器 7 | 0 | 地址7 | 8000004 |
| 寄存器 8 | 0 | 地址8 | ac02003c |
| 寄存器 9 | 0 | 地址9 | 8000000 |
| 寄存器 10 | 0 | 地址10 | 0 |
| 寄存器 11 | 0 | 地址11 | 0 |
| 寄存器 12 | 0 | 地址12 | 1 |
| 寄存器 13 | 0 | 地址13 | 65 |
| 寄存器 14 | 0 | 地址14 | 0 |
| 目前PC | 20 | 地址15 | 13ba |
| 目前EPC | 0 | 地址16 | 0 |
| | | 地址17 | 0 |
| | | 地址18 | 0 |
| | | 地址19 | 0 |
| | | 地址20 | 0 |
| | | 地址21 | 0 |
| | | 地址22 | 0 |
| | | 地址23 | 0 |
| | | 地址24 | 0 |
| | | 地址25 | 0 |
| | | 地址26 | 0 |
| | | 地址27 | 0 |
| | | 地址28 | 0 |
| | | 地址29 | 0 |
| | | 地址30 | 0 |
| | | 地址31 | 0 |
| | | 地址32 | 0 |
| | | 地址33 | 0 |
| | | 地址34 | 0 |
| | | 地址35 | 0 |
| | | 地址36 | 0 |
| | | 地址37 | 0 |
| | | 地址38 | 0 |
| | | 地址39 | 0 |
| | | 地址40 | 0 |
| | | 地址41 | 0 |
| | | 地址42 | 0 |
| | | 地址43 | 0 |
| | | 地址44 | 0 |
| | | 地址45 | 0 |
| | | 地址46 | 0 |
| | | 地址47 | 0 |
| | | 地址48 | 0 |
| | | 地址49 | 0 |
| | | 地址50 | 0 |
| | | 地址51 | 0 |
| | | 地址52 | 0 |
| | | 地址53 | 0 |
| | | 地址54 | 0 |
| | | 地址55 | 0 |
| | | 地址56 | 0 |
| | | 地址57 | 0 |
| | | 地址58 | 0 |
| | | 地址59 | 0 |

At the bottom of the window, there is a status bar showing 'Seconds: 147' and buttons for '继续', '停止', '确定', and '取消'.

Course

- Introduction → Chapter 1 & 4
- Arithmetic → Chapter 3
- MIPS assembly language → Chapter 2
- **CPU design** → **Chapter 5**
- Memory Hierarchy → Chapter 7
- I/OInterface & Peripherals → Chapter 8

Course Structure





课程安排 (精研班)

| 周 | 章节 | 知识点 | 分 | 实验方式 | |
|----|-------|--------|---|-------|-----------------|
| 1 | 概论 | | | | 熟悉C语言编程 |
| 2 | 数据表达 | 数据表示加减 | 5 | 程序模拟 | 选择一种表示方式，实行加减运算 |
| 3 | | 乘除 | 5 | | 乘除法设计模拟 |
| 4 | | 浮点数 | 5 | | 浮点处理或汉字 |
| 5 | 汇编语言 | 汇编 | 5 | | MIPS汇编到机器码 |
| 6 | | 反汇编 | 5 | | 机器码到MIPS汇编 |
| 7 | | 模拟CPU | 5 | | MIPS CPU模拟 |
| 8 | CPU设计 | 单个组件 | 5 | 硬件实验 | 交硬件实验报告 |
| 9 | | 单时钟 | 5 | | 交硬件实验报告 |
| 10 | | 多时钟 | 5 | | 交硬件实验报告 |
| 11 | 存储系统 | 存储器 | 5 | | (期中考试) |
| 12 | | Cache | 5 | | (平时机动) |
| 13 | | 虚拟存储 | 5 | 程序模拟 | 程序模拟虚拟存储 |
| 14 | I/O系统 | I/O | 5 | 硬件实验 | 显示模拟 |
| 15 | | 综合 | 5 | 软件/硬件 | 综合演示 |
| 16 | 复习 | | | | |





Computer Organization & Design

第01章：Introduction

楼学庆

<http://10.214.47.99/>

[Email:hzlou@163.com](mailto:hzlou@163.com)

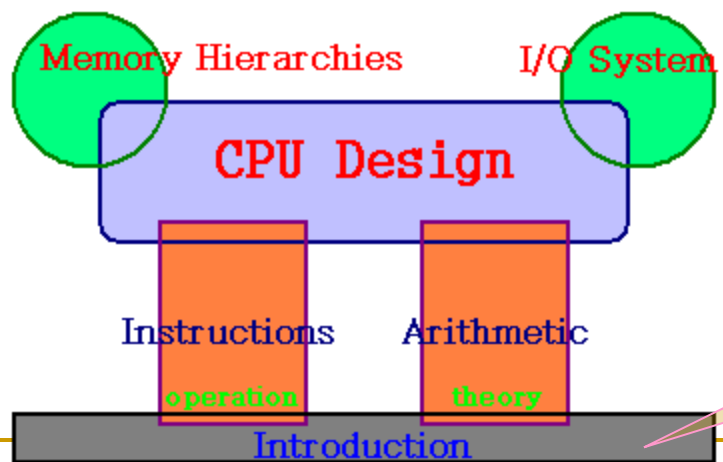


玉泉校区曹光彪东楼507室



17:58:13

浙江大学计算机学院



von Neumann' Computer

- Today's computers are built on 2 key principles:
 - ①Instruction are represented as numbers.
 - ②Programs can be stored in memory to be read or written just like numbers.

Computer Generations

- First generation
 - 1950-1959, vacuum tubes, commercial electronic computer
- Second generation
 - 1960-1968, transistors, cheaper computers
- Third generation
 - 1969-1977, integrated circuit, minicomputer
- Fourth generation
 - 1978-1997, LSI and VLSI, PCs and workstations
- Fifth generation
 - 1998-?, micromation and hugeness



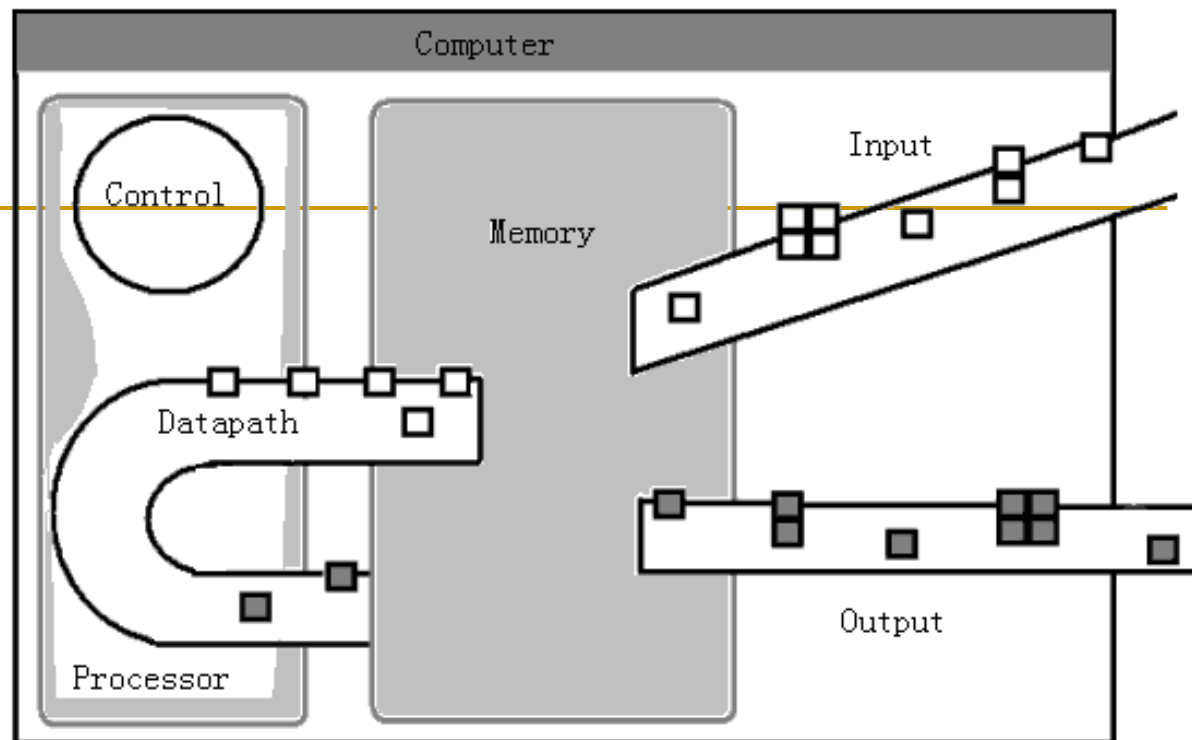
Four Design Principles

- 1. Simplicity favors regularity
- 2. Smaller is faster
- 3. Good design demands good compromises
- 4. Make the common case fast



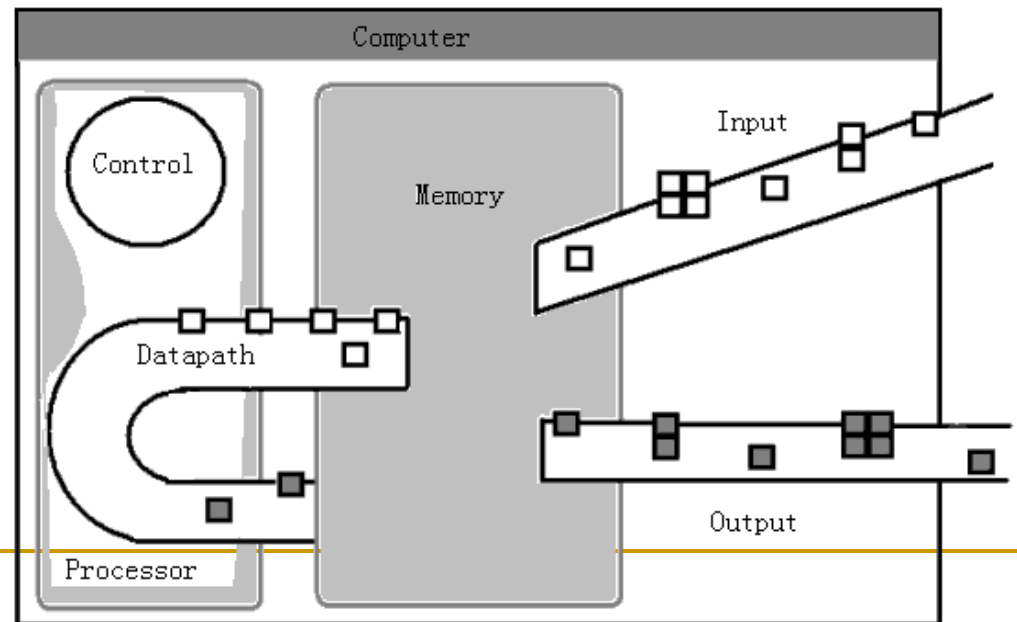
Part 1:

Hardware



Five Parts

- Control (CU)
 - Datapath (EU)
 - Memory
 - Input
 - Output
- } CPU
- } RAM
- } I/O



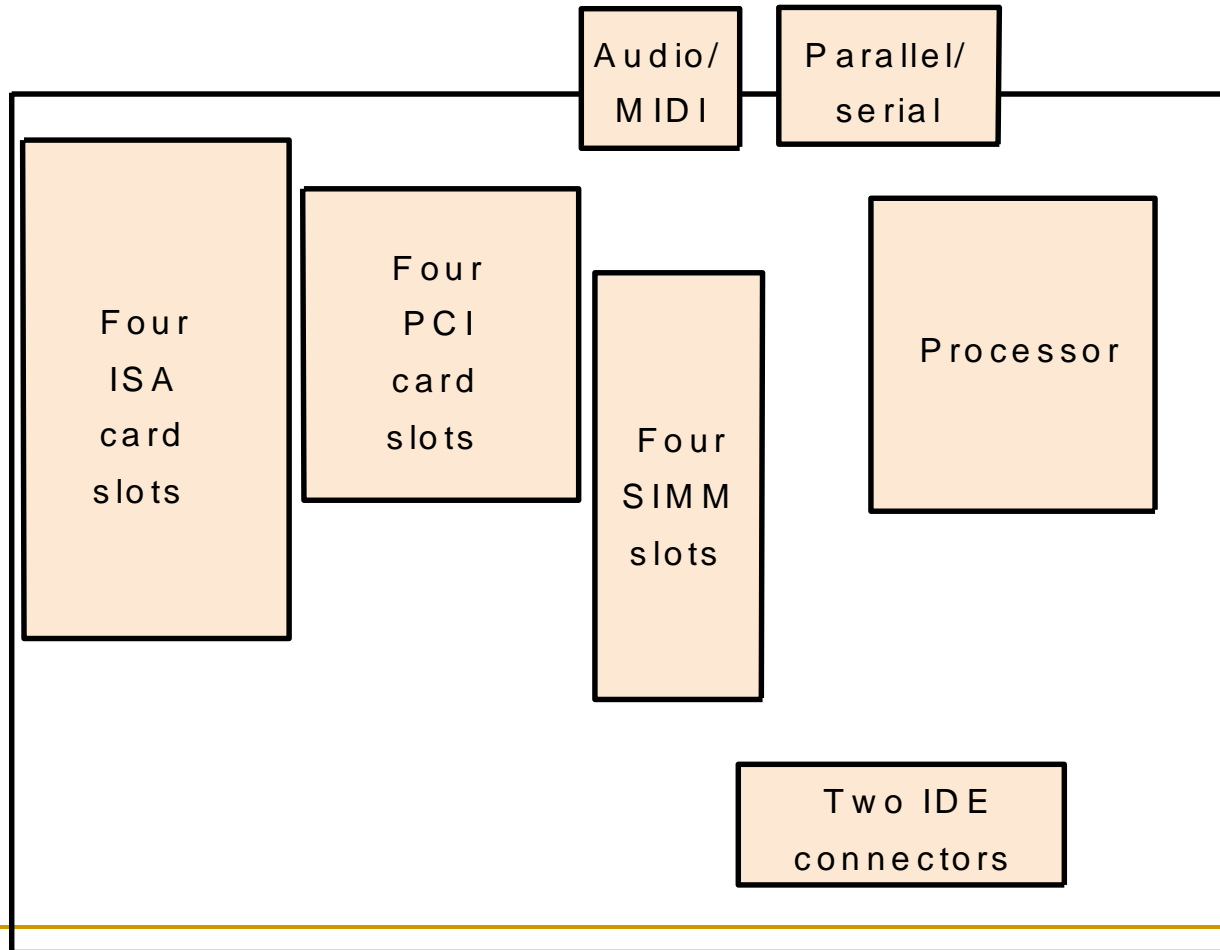


Lecture 1: Introduction

- § 1.1 Below Your Program
- § 1.2 Under the Covers
- § 1.3 CPU Performance and Its Factors
- § 1.4 Evaluating Performance



Close-up of PC motherboard

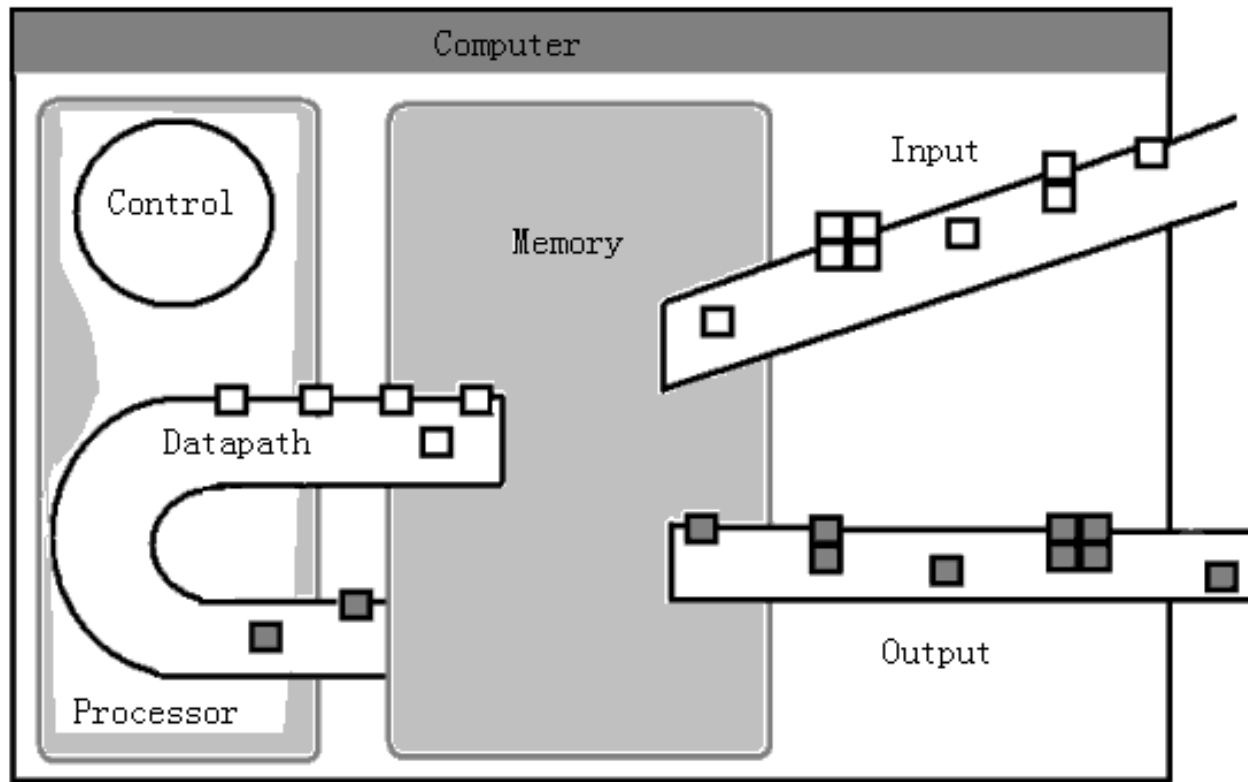


Must a Programmer Care About Hardware?



- Memory management: if we understand how/where data is placed, we can help ensure that relevant data is nearby
 - Thread management: if we understand how threads interact, we can write smarter multi-threaded programs
- Why do we care about multi-threaded programs?

Computer Organization





Control Process Unit →

CPU

- CPU
 - Control (CU)
 - Datapath (EU)
- Specialties
 - Main Frequency
 - IBM-PC: 4.77MHz
 - Now: 3.8 GHz
 - Machine Word
 - IBM-PC: 8086/8088: 16bits
 - Now: 32bits

Control Process Unit →



CPU

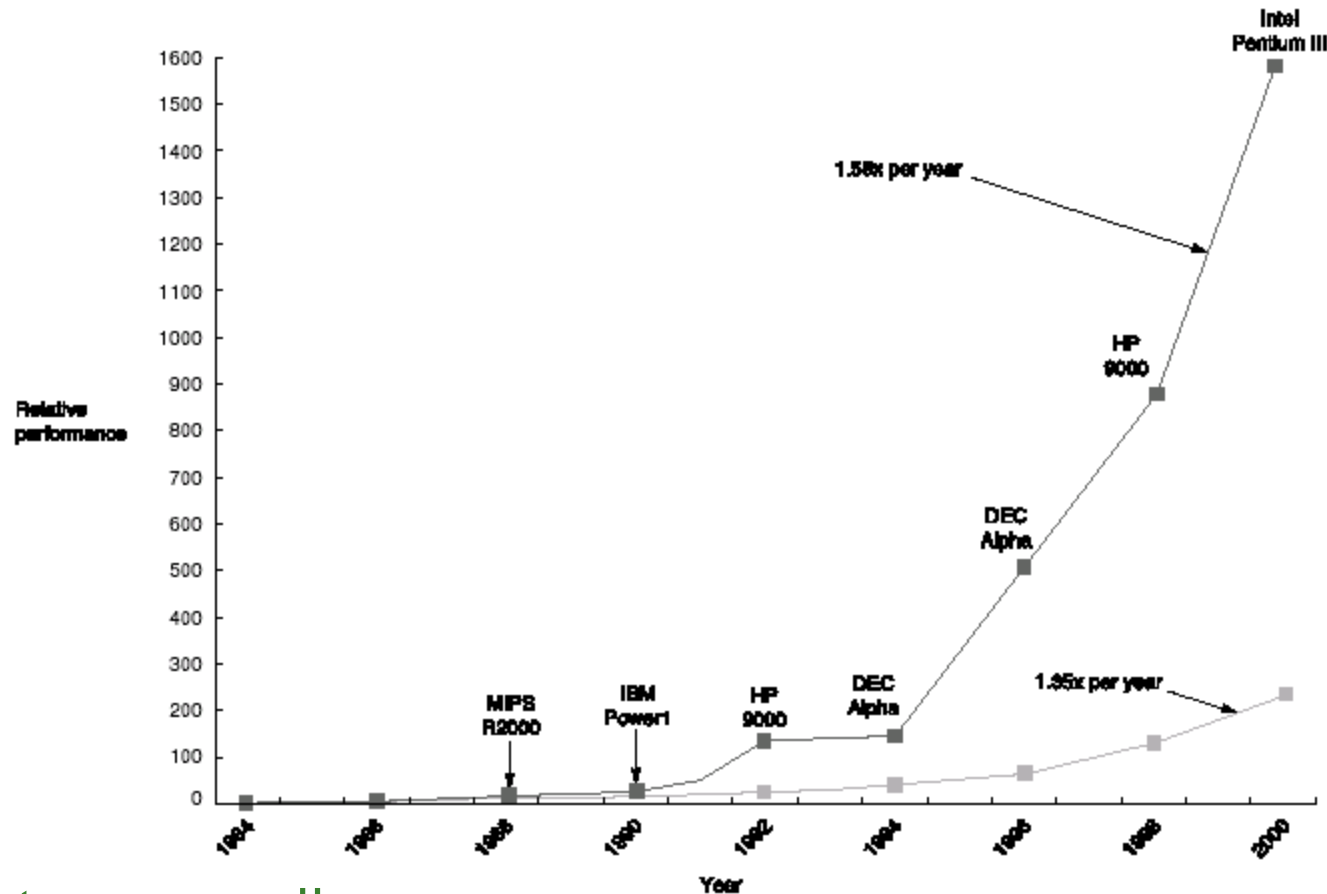
■ Instruction Set

- CISC: Complex Instruction Set Computer
- RISC: Reduced Instruction Set Computer

■ Performance

- CPI: Clock Cycles per Instruction
- MIPS: Millions Instructions per Second
- MFLOPS: millions of floating-point operations per second

Microprocessor Performance



50% improvement every year!!

What contributes to this improvement?

© 2003 Elsevier Science (USA). All rights reserved.

Modern Trends

- Historical contributions to performance:
 - Better processes (faster devices) ~20%
 - Better circuits/pipelines ~15%
 - Better organization/architecture ~15%
- In the future, bullet-2 will help little and bullet-3 will not help much for a single core!

| | Pentium | P-Pro | P-II | P-III | P-4 | Itanium | Montecito |
|-------------|---------|-------|------|-------|------|---------|-----------|
| Year | 1993 | 95 | 97 | 99 | 2000 | 2002 | 2005 |
| Transistors | 3.1M | 5.5M | 7.5M | 9.5M | 42M | 300M | 1720M |
| Clock | 60M | 200M | 300M | 500M | 750M | 800M | 1800M |

Moore's Law in action

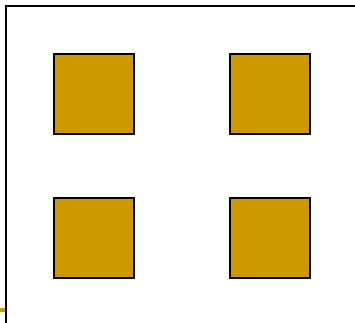


At this point, adding transistors to a core yields little benefit

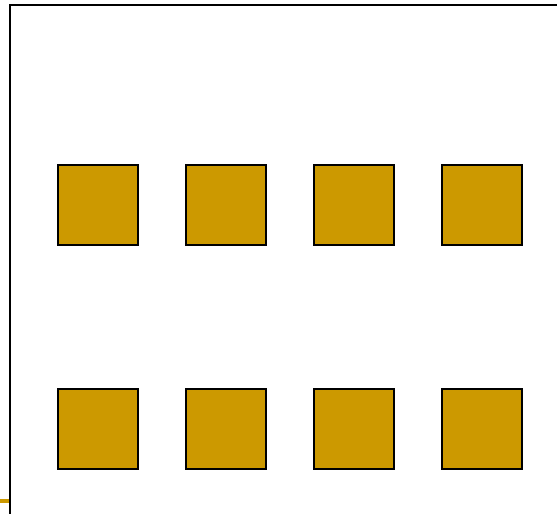
What Does This Mean to a Programmer?

- In the past, a new chip directly meant 50% higher performance for a program
- Today, one can expect only a 20% improvement, unless... the program can be broken up into multiple threads
- Expect #threads to emerge as a major metric for software quality

4-way multi-core



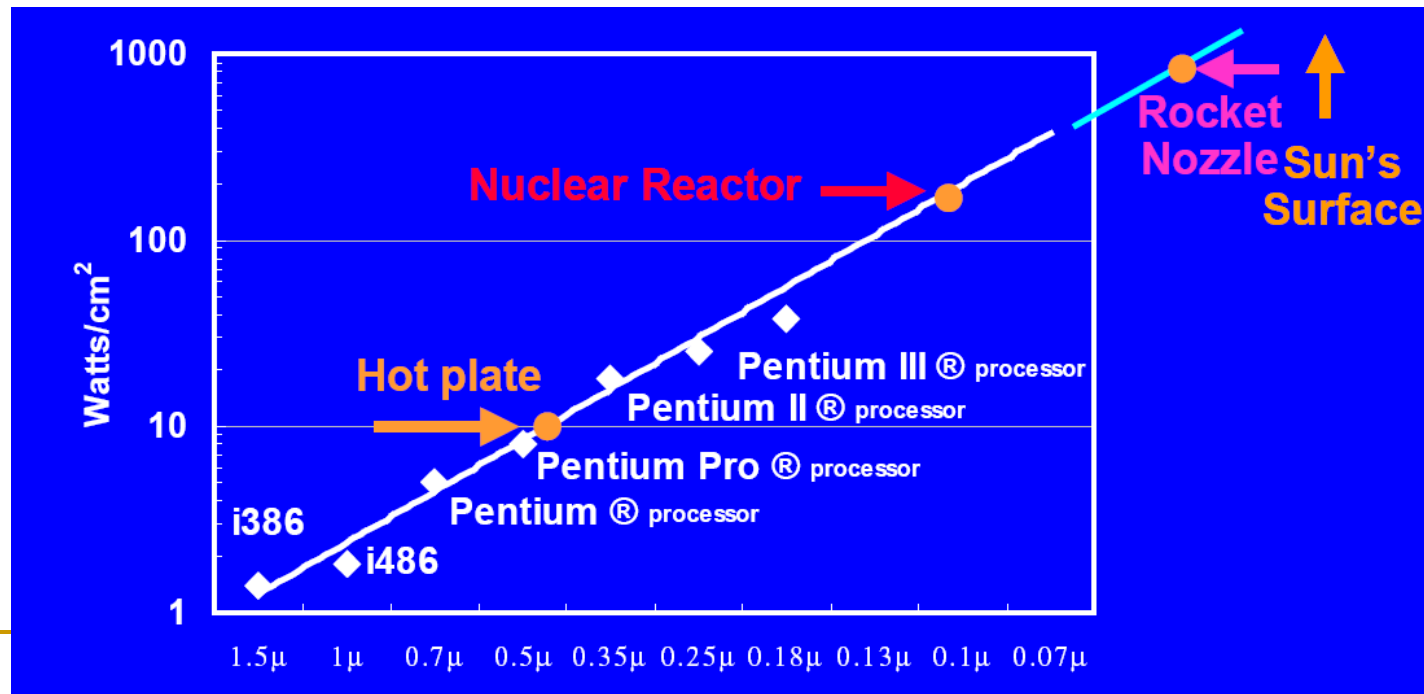
8-way multi-core



Challenges for the Hardware Designers

Major concerns:

- The performance problem (especially scientific workloads)
- The power dissipation problem (especially embedded processors)
- The temperature problem
- The reliability problem



The HW/SW Interface

Application software

Systems software
(OS, compiler)

Hardware

$a[i] = b[i] + c;$

↓
Compiler

```
lw    $15, 0($2)
add   $16, $15, $14
add   $17, $15, $13
lw    $18, 0($12)
lw    $19, 0($17)
add   $20, $18, $19
sw    $20, 0($16)
```

↓
Assembler

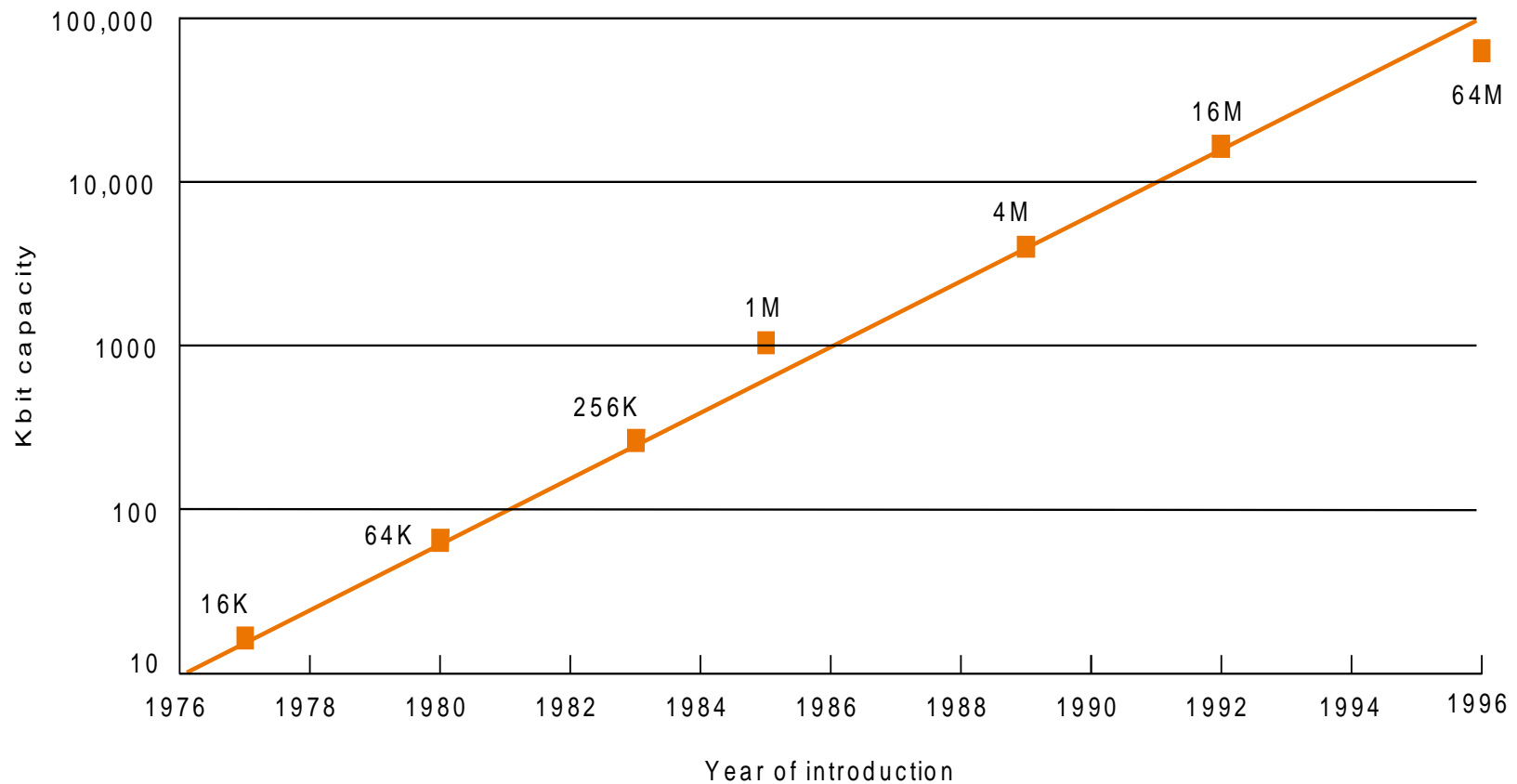
```
000000101100000
110100000100010
```

Computer Components

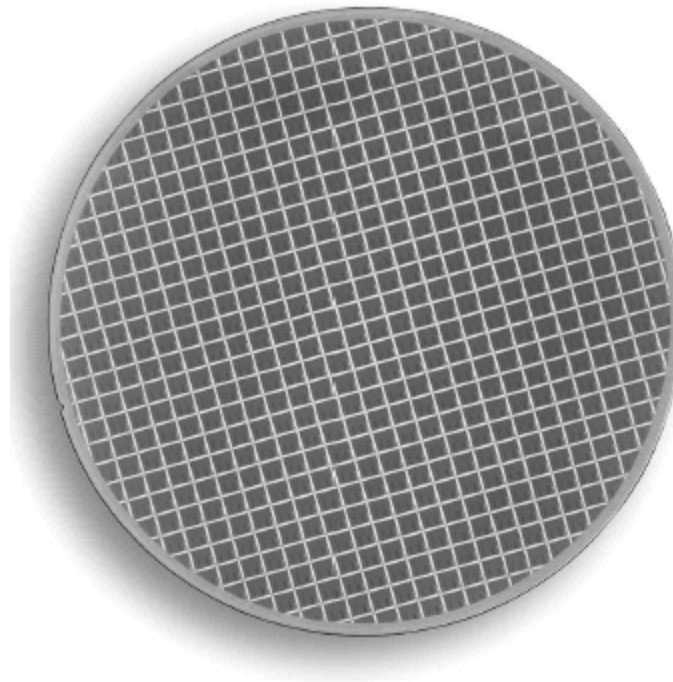
- Input/output devices
- Secondary storage: non-volatile, slower, cheaper
- Primary storage: volatile, faster, costlier
- CPU/processor



Growth of capacity per DRAM chip over time

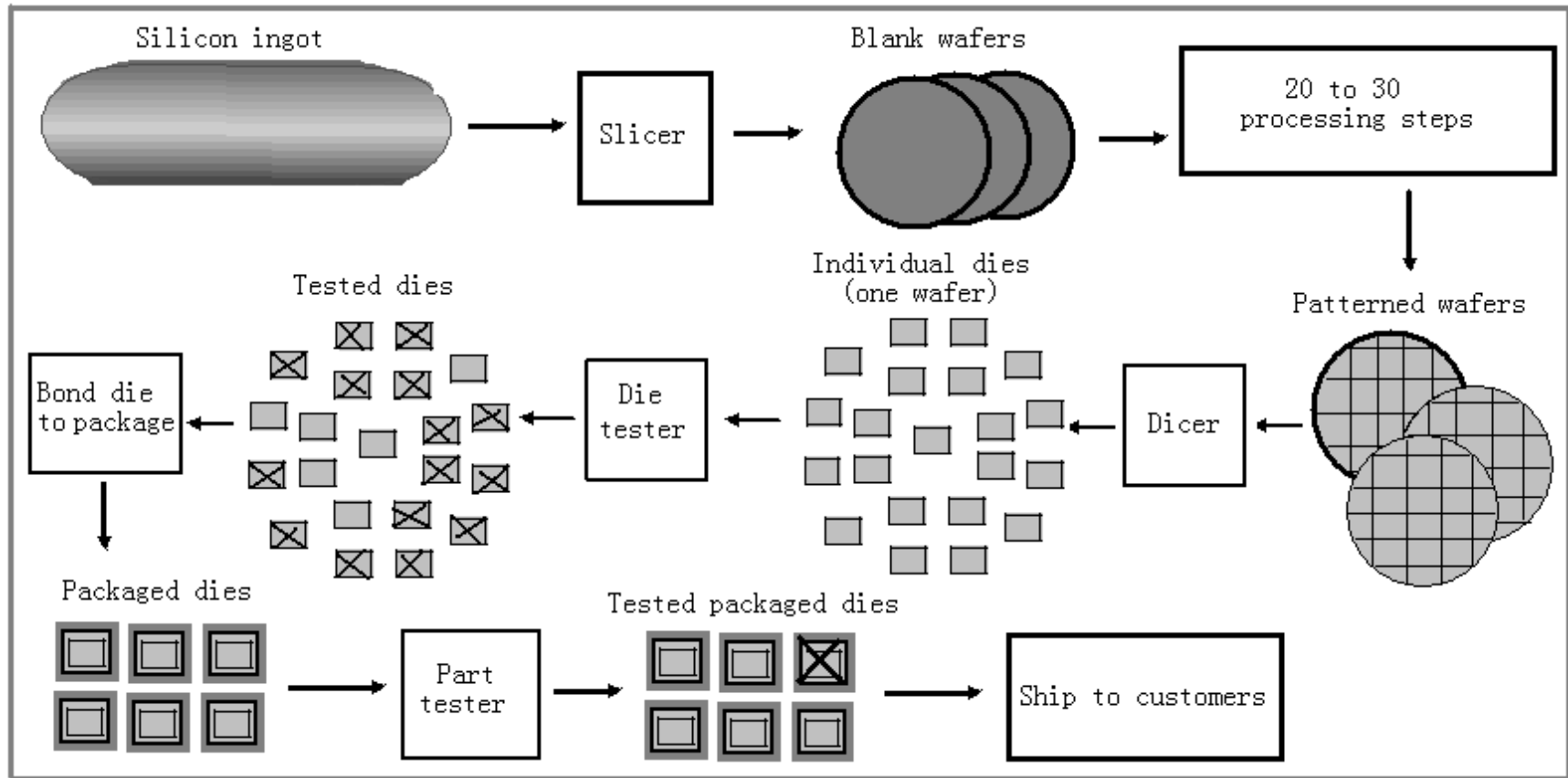


Wafers and Dies

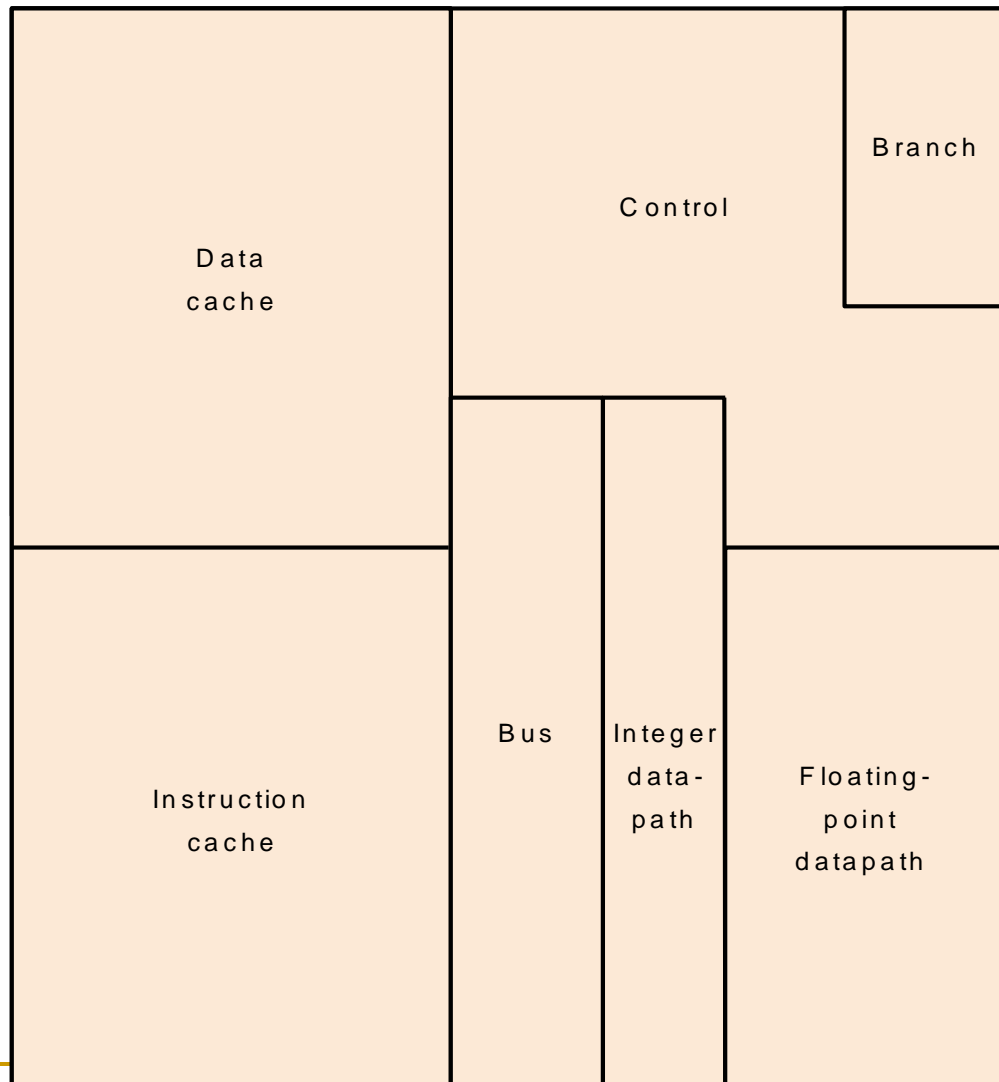


© 2003 Elsevier Science (USA). All rights reserved.

The semiconductor silicon and the chip manufacturing process



Inside the processor chip



Manufacturing Process

- Silicon wafers undergo many processing steps so that different parts of the wafer behave as insulators, conductors, and transistors (switches)
- Multiple metal layers on the silicon enable connections between transistors
- The wafer is chopped into many dies – the size of the die determines yield and cost

Processor Technology Trends

- Shrinking of transistor sizes: 250nm (1997) → 130nm (2002) → 70nm (2008) → 35nm (2014)
- Transistor density increases by 35% per year and die size increases by 10-20% per year... functionality improvements!
- Transistor speed improves linearly with size (complex equation involving voltages, resistances, capacitances)
- Wire delays do not scale down at the same rate as transistor delays

Memory

- RAM: volatile, faster, costlier
 - IBM-PC: 640KB
 - Now: 16GB or ...
- Disk: non-volatile, slower, cheaper
 - IBM-PC:
 - Hard-disk: 10MB, Tape → 1TB+128GB
 - Floppy-disk: 5¼" (360KB~1.2MB), 3½" (1.4MB)
 - Now:
 - Hard-disk: ...
 - Flash-disk: 1 GB
 - Flash

Memory and I/O Technology Trends

- DRAM density increases by 40-60% per year, latency has reduced by 33% in 10 years (the memory wall!), bandwidth improves twice as fast as latency decreases
- Disk density improves by 100% every year, latency improvement similar to DRAM
- Networks: primary focus on bandwidth; 10Mb → 100Mb in 10 years; 100Mb → 1Gb in 5 years

Input & Output

- Input

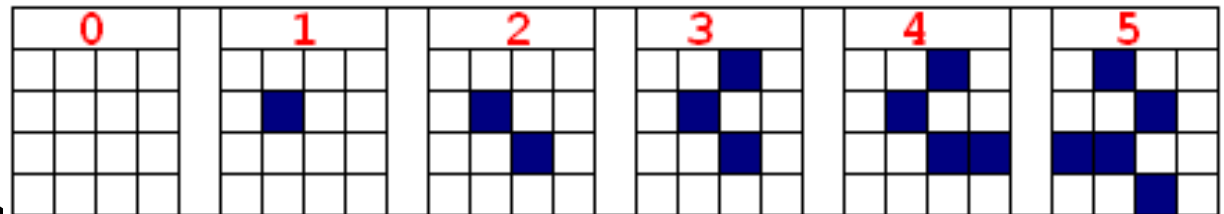
- ❑ Keyboard
- ❑ Mouse
- ❑ Scanner, ...

■ Output

- ❑ Display
- ❑ Printer, ...

■ Input & Output

- ## Disk, ...



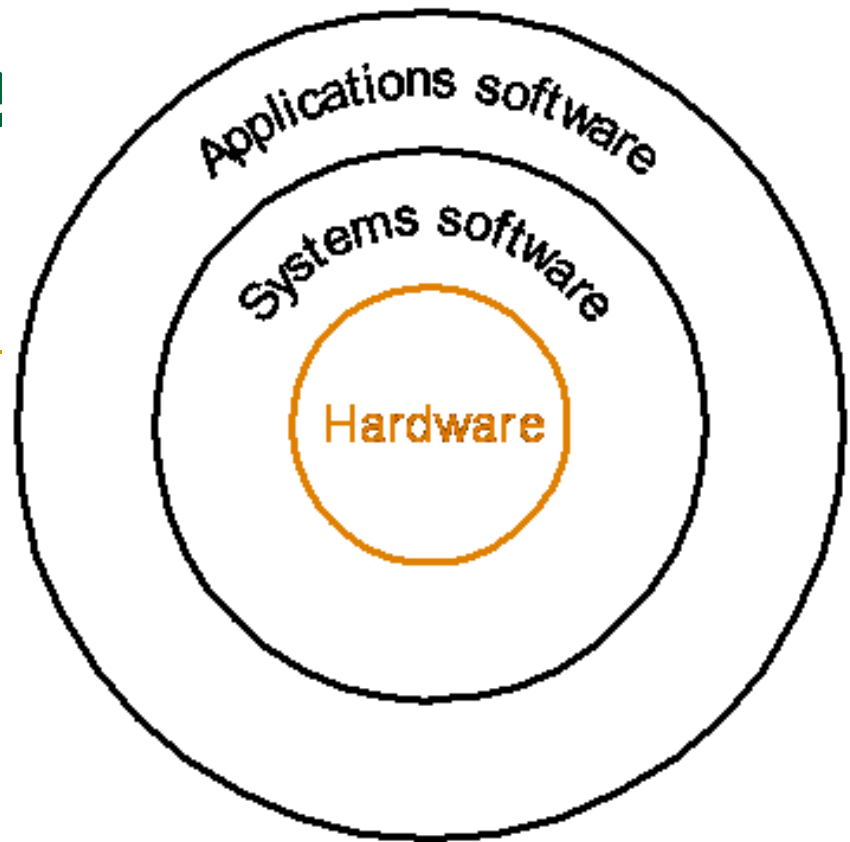
The diversity of I/O devices

| Device | Behavior | Partner | Data rate (KB/sec) |
|------------------|-----------------|---------|--------------------|
| Keyboard | input | human | 0.01 |
| Mouse | input | human | 0.02 |
| Voice input | input | human | 0.02 |
| Scanner | input | human | 400.00 |
| Voice output | output | human | 0.60 |
| Line printer | output | human | 1.00 |
| Laser printer | output | human | 200.00 |
| Graphics display | output | human | 60,000.00 |
| Modem | input or output | machine | 2.00-8.00 |
| Network/LAN | input or output | machine | 500.00-6000.00 |
| Floppy disk | storage | machine | 100.00 |
| Optical disk | storage | machine | 1000.00 |
| Magnetic tape | storage | machine | 2000.00 |
| Magnetic disk | storage | machine | 2000.00-10,000.00 |

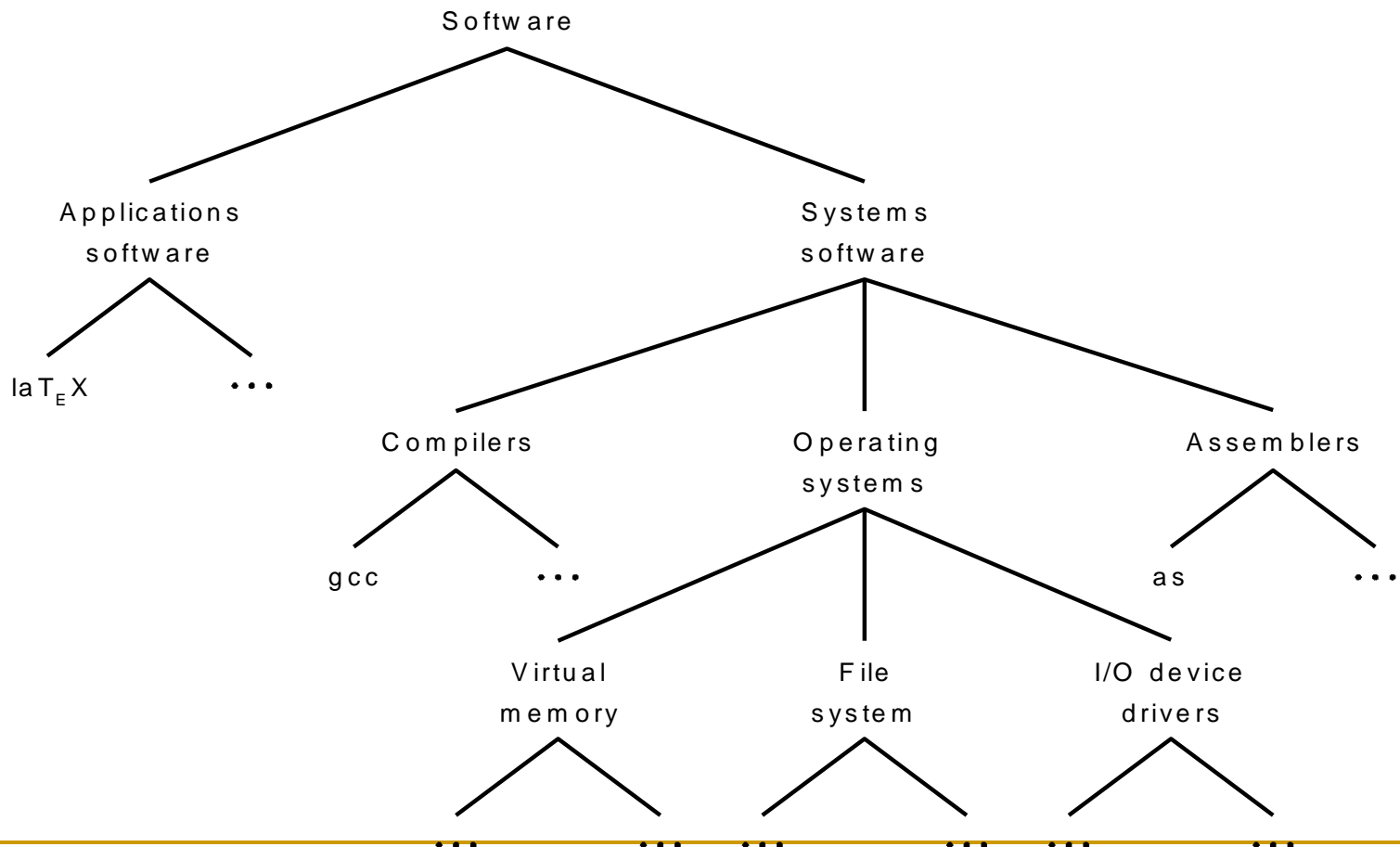


Part 2:

Software



An example of the decomposability of computer systems



The HW/SW Interface

Application software

Systems software
(OS, compiler)

Hardware

$a[i] = b[i] + c;$

↓
Compiler

```
lw    $15, 0($2)
add   $16, $15, $14
add   $17, $15, $13
lw    $18, 0($12)
lw    $19, 0($17)
add   $20, $18, $19
sw    $20, 0($16)
```

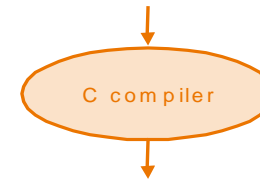
↓
Assembler

```
000000101100000
110100000100010
```

High-level
language
program
(in C)

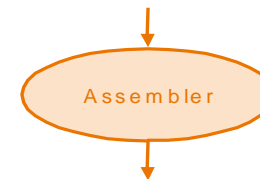
```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```

■ The process of compiling and assembling



Assembly
language
program
(for MIPS)

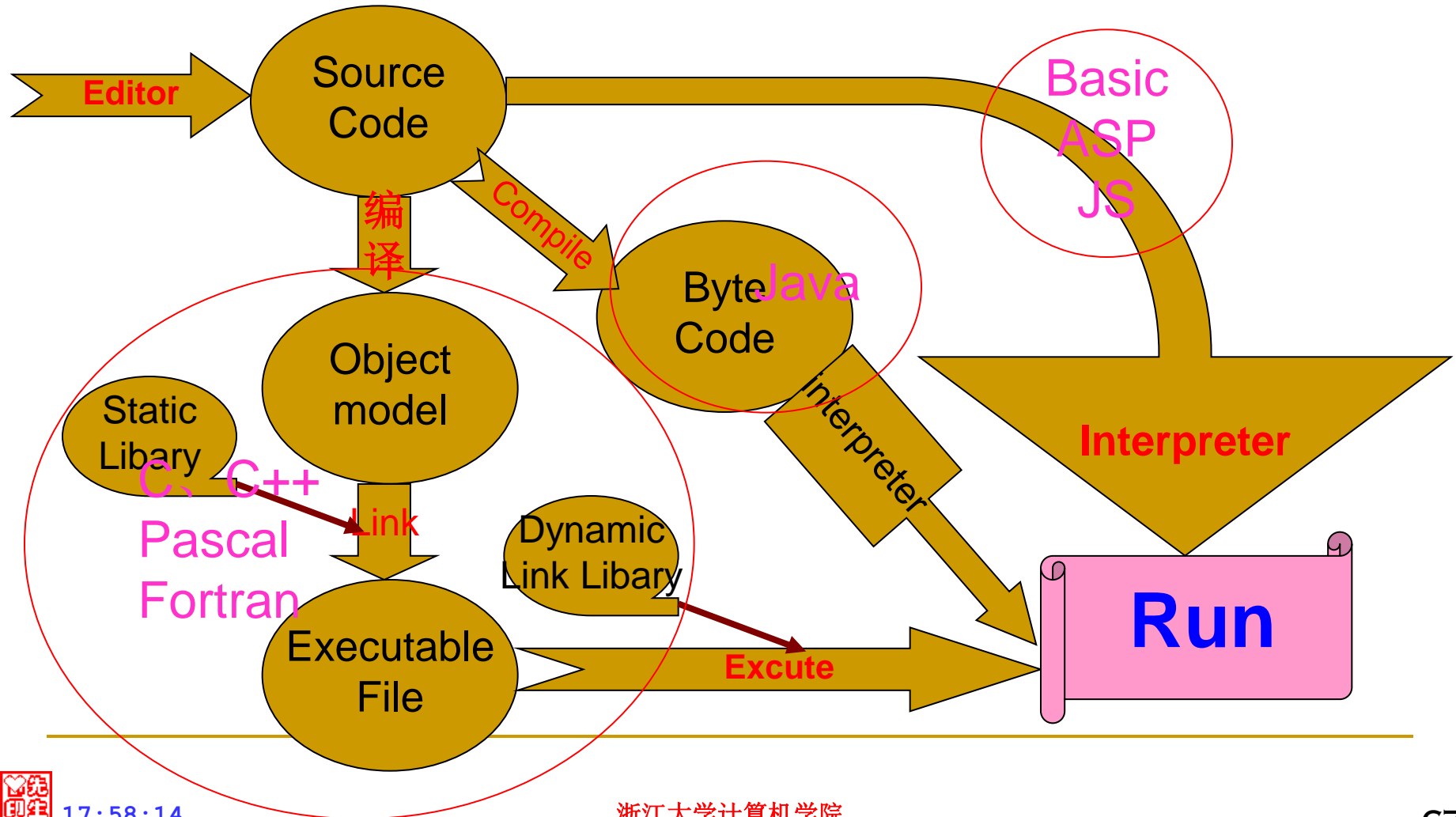
```
swap:
  muli $2, $5, 4
  add $2, $4, $2
  lw $15, 0($2)
  lw $16, 4($2)
  sw $16, 0($2)
  sw $15, 4($2)
  jr $31
```



Binary machine
language
program
(for MIPS)

```
000000001010000100000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
100011001111001000000000000000100
10101100111100100000000000000000
101011000110001000000000000000100
00000011111000000000000000001000
```

程序执行



■ Java

```
public class Fruit
{
    public static void main(String args[])
    {
        // Declare and initialize three variables
        int numOranges = 5;                // Count of oranges
        int numApples = 10;                // Count of apples
        int numFruit = 0;                  // Count of fruit

        numFruit = numOranges + numApples; // Calculate the total fruit

        // Display the result
        System.out.println("A totally fruity program");
        System.out.println("Total fruit is " + (numFruit +22));
    }
}
```



Start a C program in a file on disk to run 1

- **Compiling**
 - C program → assembly language program
- **Assembling**
 - Assembly language program → machine language module
- **Linking**
 - Object modules (including library routine) → executable program
 - 6 pieces of the object file for Unix systems:
 - object file header
 - text segment
 - data segment
 - relocation information
 - symbol table
 - debugging information
 - Place code and data modules symbolically in memory
 - Determine the addresses of data and instruction labels
 - Patch both the internal and external references

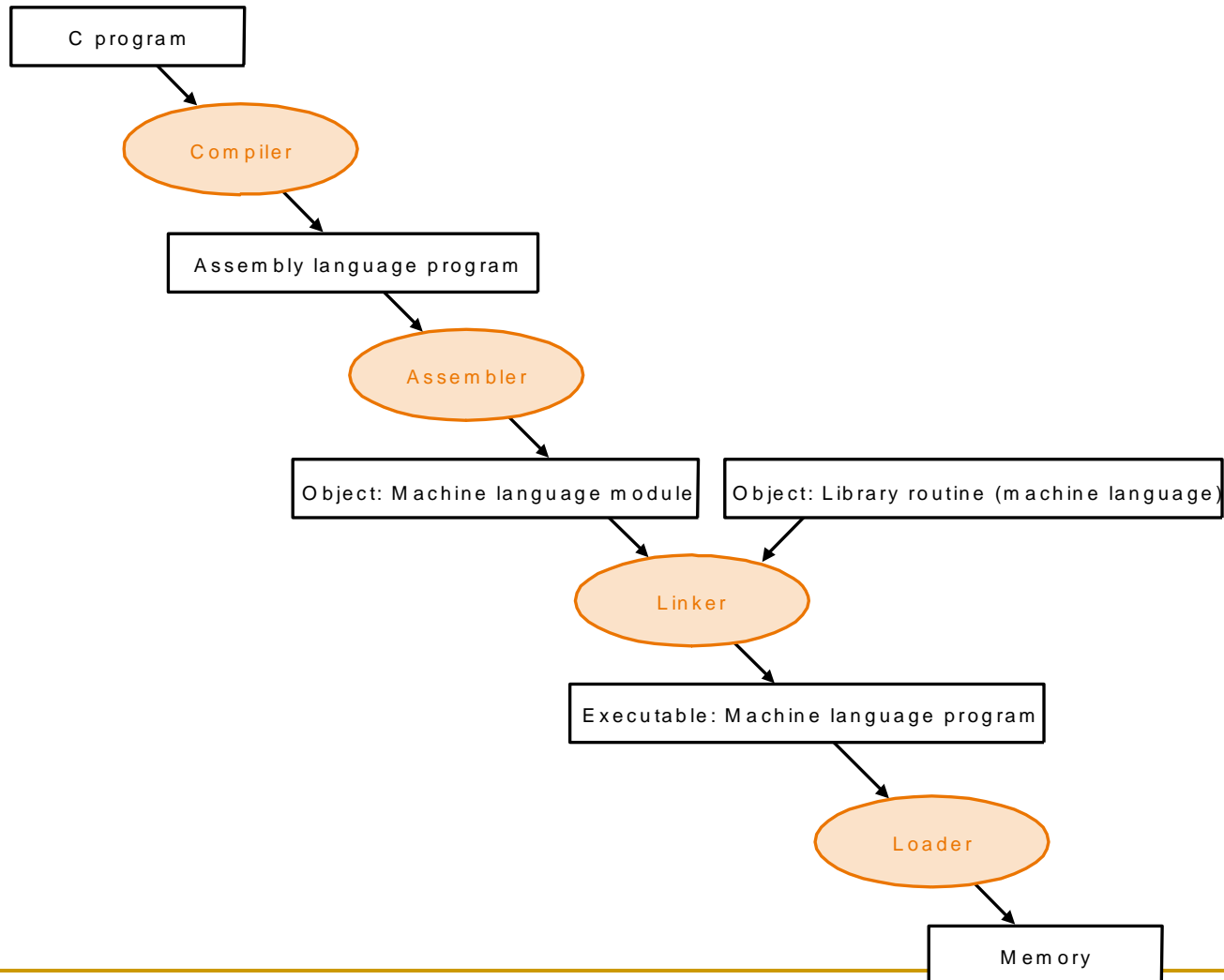


Start a C program in a file on disk to run 2

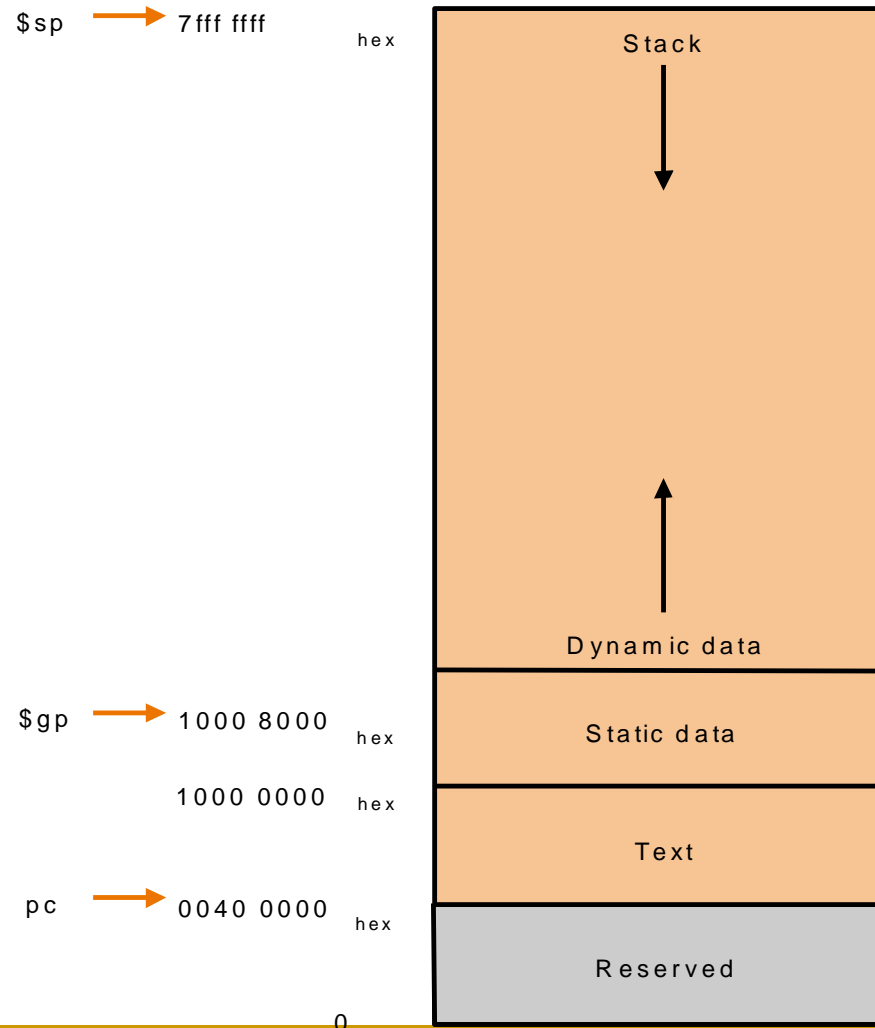
■ Loading

- ❑ Determine size of text and data segments
- ❑ Create an address space large enough
- ❑ Copy instructions and data from executable file to memory
- ❑ Copy parameters (if any) to the main program onto the stack
- ❑ Initialize registers and set \$sp to the first free location
- ❑ Jump to a start-up routine

A translation hierarchy



MIPS memory allocation for program and data





Thank You!

非常天空

浙江大学计算机学院



成绩

- 考试：70% (不低于40分)
 - 英文
 - 闭卷
- 平时：30%
 - 作业
 - 期中
 - 点名
- 额外：
 - 课堂练习 (+5%)：每题1分