



2023秋季

计算机系统概论

Introduction to Computer Systems

⊗ Zhang, Youhui (张悠慧)

✉ zyh02@tsinghua.edu.cn



课程引言

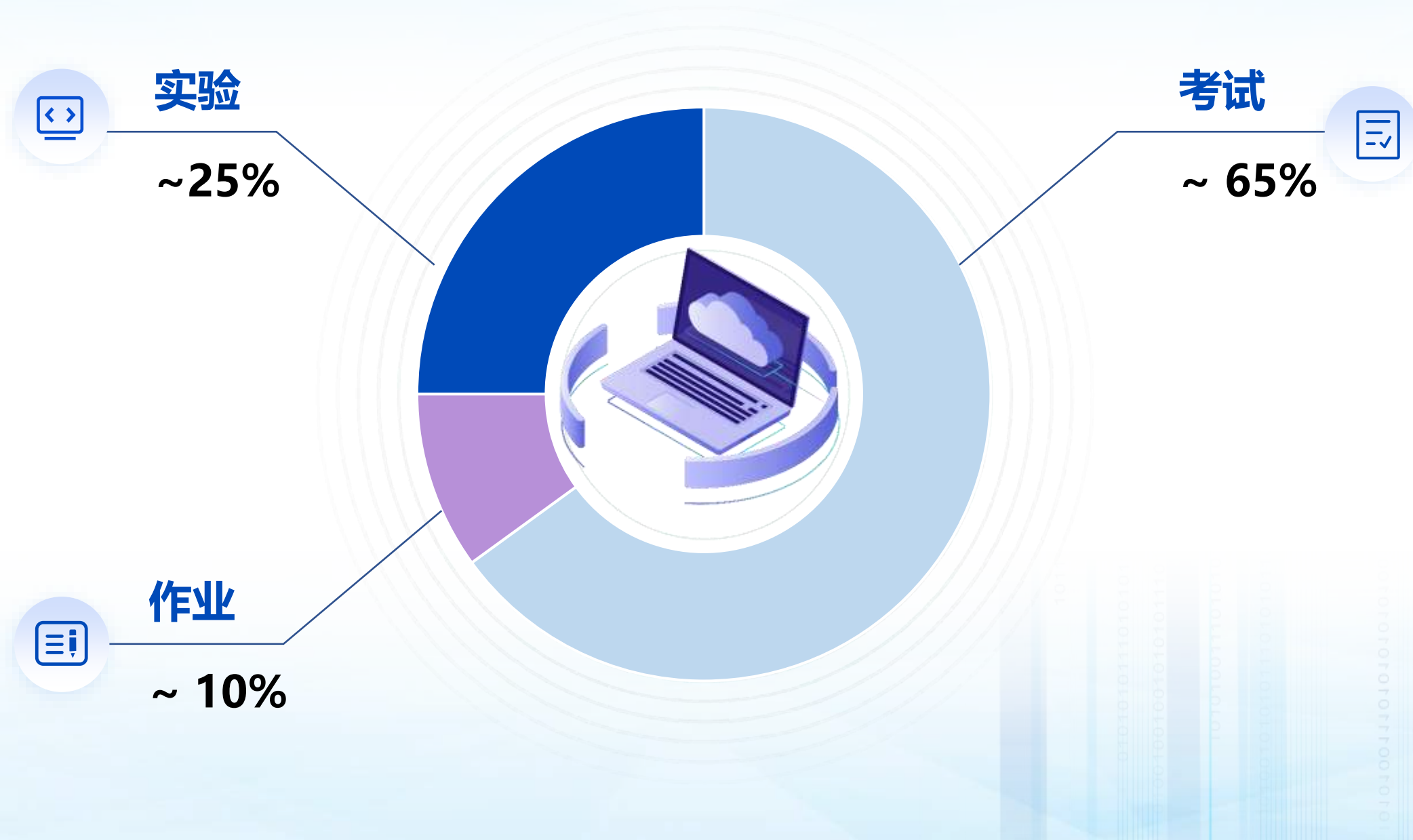
参考书目：《深入理解计算机系统（第三版）》
(Computer Systems: A Programmer's Perspective)

课本内容对照要求（随着进展，可能略有变化）：

章节	要求的内容	说明
第二章	2.1; 2.2; 2.3（乘/除不要求，但2的整数次幂除外）； 2.4（2.4.5的运算不要求）	相关C语言概念要掌握，不专门讲
第三章	3.2; 3.3; 3.4; 3.5; 3.6; 3.7; 3.8; 3.9; 3.10	
第七章	7.1; 7.2; 7.3; 7.4; 7.5; 7.6; 7.7; 7.8; 7.9; 7.10; 7.12.1	
第八章	8.1; 8.2; 8.3; 8.4; 8.5; 8.6	
第九章	9.1; 9.2; ; 9.4; 9.5; 9.6（多级页表不要求）； 9.7.2; 9.8; 9.9（9.9.12之前，包括实验）	
第十章	10.1; 10.2; 10.3; 10.4; 10.6; 10.7; 10.8; 10.9; 10.10; 10.11	
第十二章	12.3; 12.4; 12.5（12.5.5不要求）； 12.7	

联系方式： 8-209@东主楼; 62783505-8003; zyh02@tsinghua.edu.cn


计分



关于本堂课

01 关于计算机系统自身的课（相对于算法、应用等）

课程定位

 系统类的引导课
与其它系统课程的关系

 国外大学相关课程

 课程知识点

02 课程的一些重要概念

 计算机体系结构

 程序；编译与链接

03 课程章节安排

 课程章节安排

计算机系统概论

本科专业主修课

大二秋季学期 / 课内学时48 (3学分)

定位

系统课组入门课

C/C++语言程序设计

计算机系统概论

计算机组成原理

操作系统

编译原理

体系结构

课程学习目标

C 语言程序设计

计算机编程者 使用者的视角

计算机系统概论

构建程序、进程以及初步的计算机系统概念，掌握系统硬件层面的程序表示与运行过程，并具备一定的系统编程能力，为学习后续的计算机系统课程打下基础

计算机组成原理

冯·诺依曼结构 / 计算机指令系统 / 汇编指令与编程

体系结构

线程与并发概念

编译原理

数据与程序的机器表示 / 程序链接 / 编译优化（示例）

操作系统

异常处理 / 虚拟存储管理 / 系统调用

从编程者、使用者的角度入手

- ◆ 《计算机组成》 / 《计算机系统结构》 等课程：从计算机构造者的角度入手
- ◆ 作为**理解整个计算机系统的起点**
 - ◆ 从“计算机编程者、使用者”视角进行内容组织与调整，为具有一定C语言编程能力的本科生构建程序、进程以及初步的计算机系统概念
 - ◆ **从C语言到汇编语言再到程序生成这一过程中所解决的基本问题，进程产生/运行/结束这一过程中的基本概念与处理流程**
- ◆ 从而理解与掌握硬件层面的数据表示/程序表示与运行，并具备一定的系统编程能力

数据存储位置对齐

对齐的一般原则

- 已知某种基本数据类型的大小为 K 字节
- 那么，其存储地址必须是K的整数倍
- X86-64的对齐要求基本上就是这样

为何需要对齐

- 计算机访问内存一般是以内存块为单位的，块的大小是地址对齐的，如4、8、16字节对齐等
- 如果数据访问地址跨越“块”边界会引起额外的内存访问

条件移动

- 语义: $\text{if (Test) Dest} \leftarrow \text{Src}$
- 1995年以后的x86处理器都支持此类指令
- GCC编译器在确保“安全”的前提下会使用它们

Why?

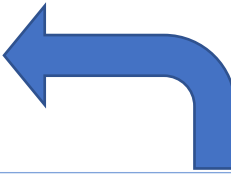
- 条件跳转指令对于现代流水线处理器的执行效率有很大的负面影响
- 条件移动指令可以避免这一现象

C代码

```
val = Test  
    ? Then_Expr  
    : Else_Expr;
```

Goto风格的代码表示

```
result = Then_Expr;  
eval = Else_Expr;  
nt = !Test;  
if (nt) result = eval;  
return result;
```



从程序员/编译器实现程序功
求出发，从高级语言（C语言）
涉及的计算机体系结构的硬

从处理器的变化/演进角度来



» CMU

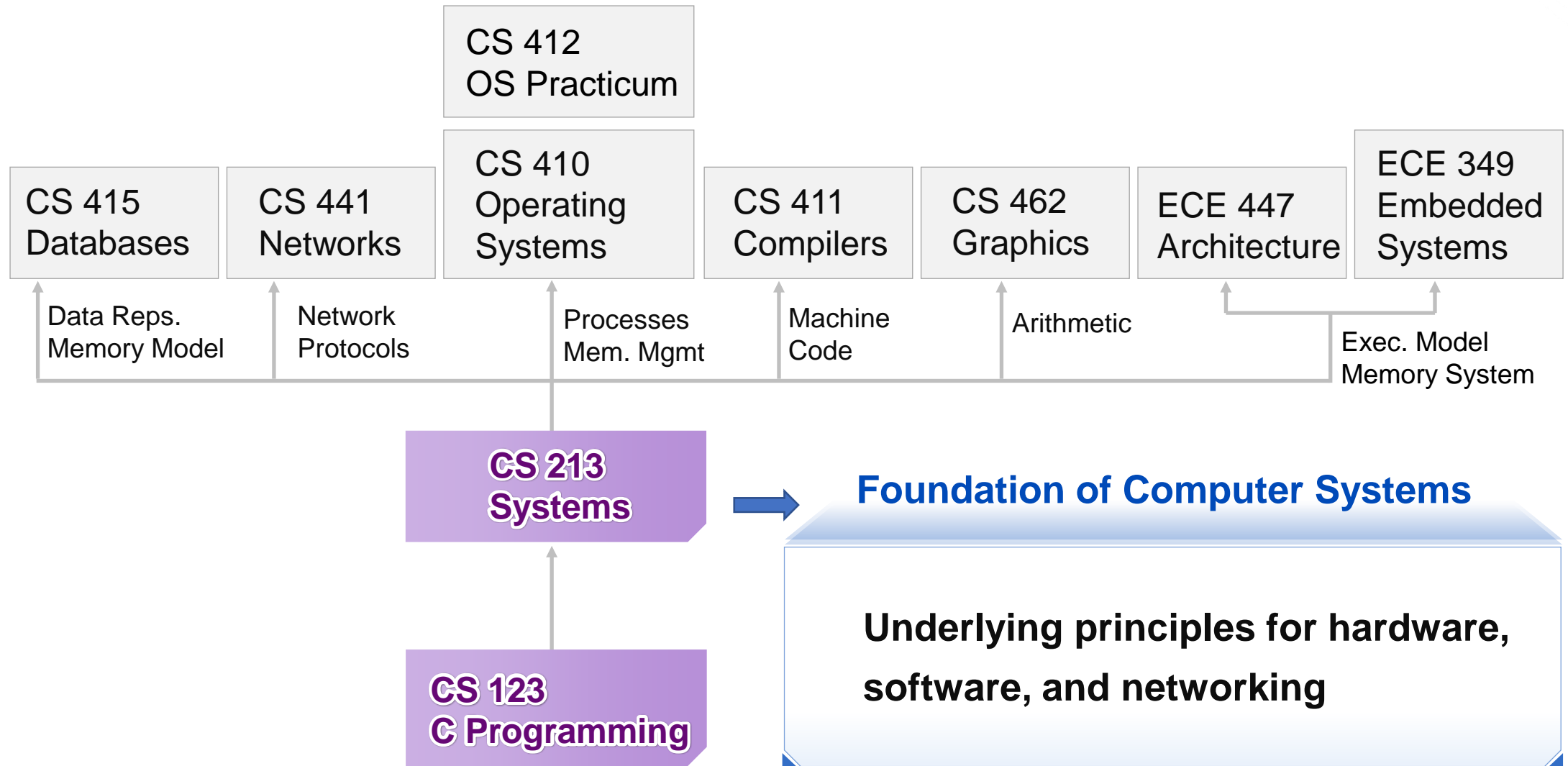
- 一个CS专业，提供5个可选方向
- 专业核心课如下：

Computer Science Core :

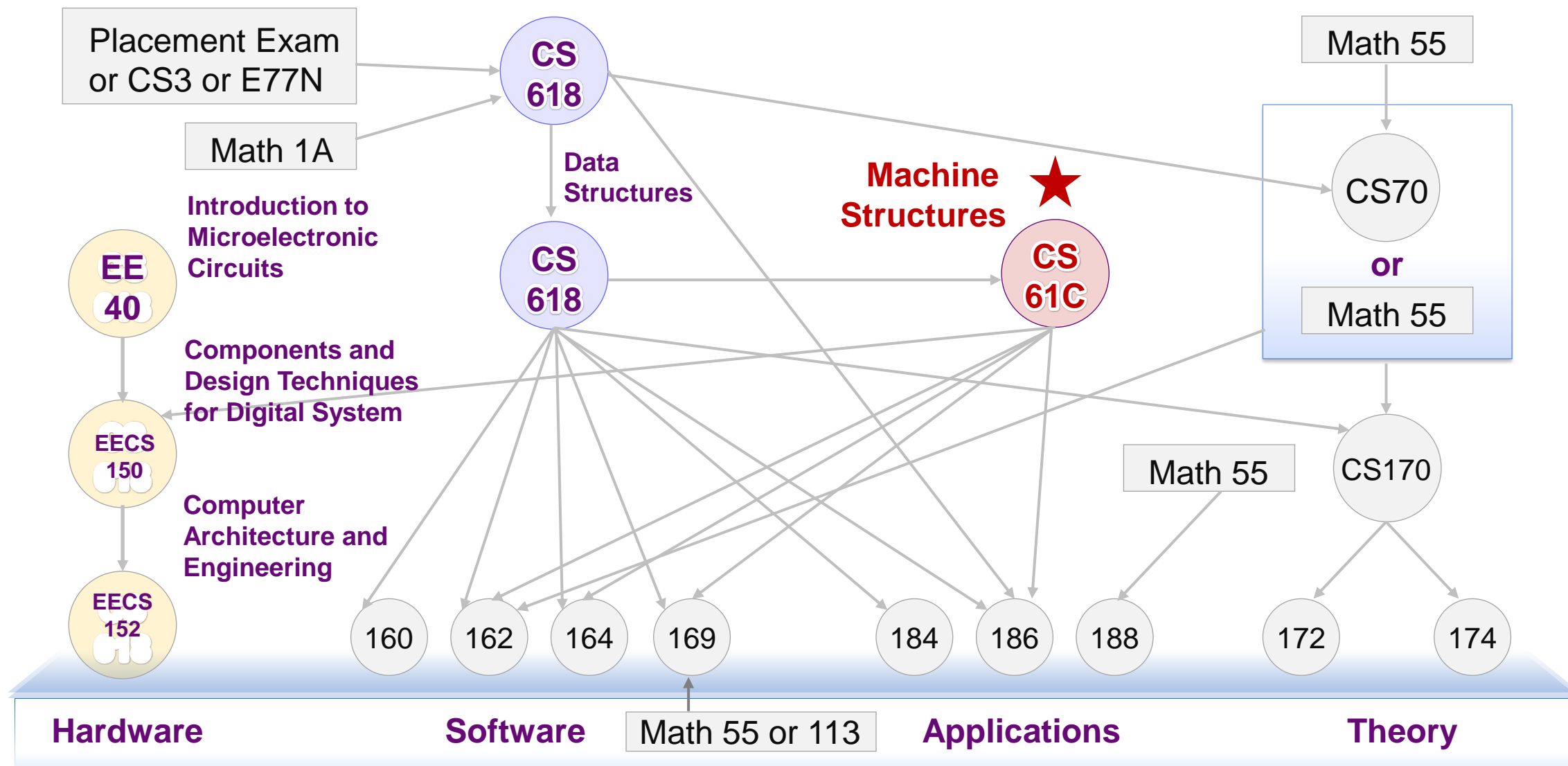
Units

15-128	Freshman Immigration Course	1
15-122	Principles of Imperative Computation (requires 21-127 as a co-req; students with no prior programming experience take 15-112 before 15-122)	10
15-150	Principles of Functional Programming	10
15-210	Parallel and Sequential Data Structures and Algorithms	12
15-213	Introduction to Computer Systems	12
15-251	Great Theoretical Ideas in Computer Science	12
15-451	Algorithm Design and Analysis	12

国外相关课程



UC Berkeley相关课程设置



美国一流大学相关必修课情况

	MIT	UC Berkeley	Stanford	CMU
课程名称	Computation Structures	Great Ideas in Computer Architecture (Machine Structures)	Computer Organization and Systems (COS)	Introduction to Computer System (ICS)
开设专业	EE、CS	CS	CS	CS
课程描述	门电路→功能部件→单周期和流水线CPU; C语言→汇编→指令→过程→进程; 并行、性能评价	C语言→汇编→指令; 应用级并行→数据级并行→线程级并行→指令级并行→寄存器传送级硬件描述	C语言→汇编→指令→微体系结构; 编译→链接→装入执行; 程序性能优化、存储器结构与管理、并发和多线程、网络编程	C语言→汇编→指令→微体系结构; 编译→链接→装入→执行; 程序性能优化、存储器结构与管理、并发和多线程、网络编程-
教程**	未指定 (讲义)	C"K&R"+ COD"P&H"+ wsc"B&H"	PP"B&O"+ C"K&R"	PP"B&O"+ C"K&R"
模型机	自定义 (RISC)	MIPS(RISC)	IA32/IA64	IA32/IA64
助教人数	12	7	15	未列出
先行课程或要求	了解编程 (函数式) 和数据结构基础, 具备电子技术基础知识	了解和掌握编程 (函数式) 和数据结构基础, 具备电子技术基础知识	了解和掌握编程和数据结构基础, 具备电子技术基础知识	了解和掌握编程 (函数式)和数据结构基础, 具备电子技术基础知识
实验内容	共8个实验,涉及门电路特性、ALU、图灵机、汇编、处理器设计、鼠标中断方式I/O等	共12个实验, 4个大作业, 涉及到 Debug、EC2、MapReduse、MIPS汇编、数据级并行、线程级并行、cache、虚拟存储管理等, 大作业和实验成绩占60%	共8个实验和7个大作业,涉及数据的表示、堆区分配、过程调用和栈的构成及使用、溢出、编译工具和编译优化等, 大作业和实验成绩占60%	共7个实验,涉及数据的表示、cache、缓冲区溢出、过程调用及栈的构成与使用、堆的分配、代理设置等, 实验成绩占50%
实验手段	各类模拟器	编程、云计算平台、模拟器	编程	编程
相关后继课程或教学内容	“数学系统设计”和“计算机体系结构及系统”等,涉及CPU设计实验和体系结构方面的模拟实验	“数字系统设计”和“计算机体系结构及工程”等,涉及CPU设计实验和体系结构方面的模拟实验, 前者是同时为CS和EE的学生开设的课程。	后继课程为 “ Digital Systems II”, 教材为COD"P&H",主要是流水线CPU和存储系统设计, 实验为用DHL设计CPU(CS学生不需要)	CS学生可选ECE开设的课程 Introduction to Computer Architecture”, 教材为COD"P&H", 主要是流水线CPU和存储系统设计,实验为用Verilog语言设计CPU

》 基础部分 (程序/数据在机器层面的表示与运行)

基础概念：计算机体系结构；程序；编译与链接作用

编译相关：数据的机器表示；汇编基本指令；程序的机器表示

链接相关：程序的链接过程；内存布局；栈溢出攻击

》 计算机系统基础概念

异常与异常控制流、虚存与虚存管理、用户层动态内存分配、线程与并发

》 计算机系统编程

系统调用：进程管理、信号处理、文件IO、线程编程

》 实验初步安排 (2-3个)

炸弹实验 or 栈溢出攻击 (程序的机器表示、debug) | **动态内存分配** (空间和时间效率的取舍) | **协程** (程序状态的直观展示)

一个进程（程序）从诞生到消亡



其它涉及内容还有：信号处理；文件IO；线程与并发



**计算机体系结构
基本概念**



**程序;
编译与链接**



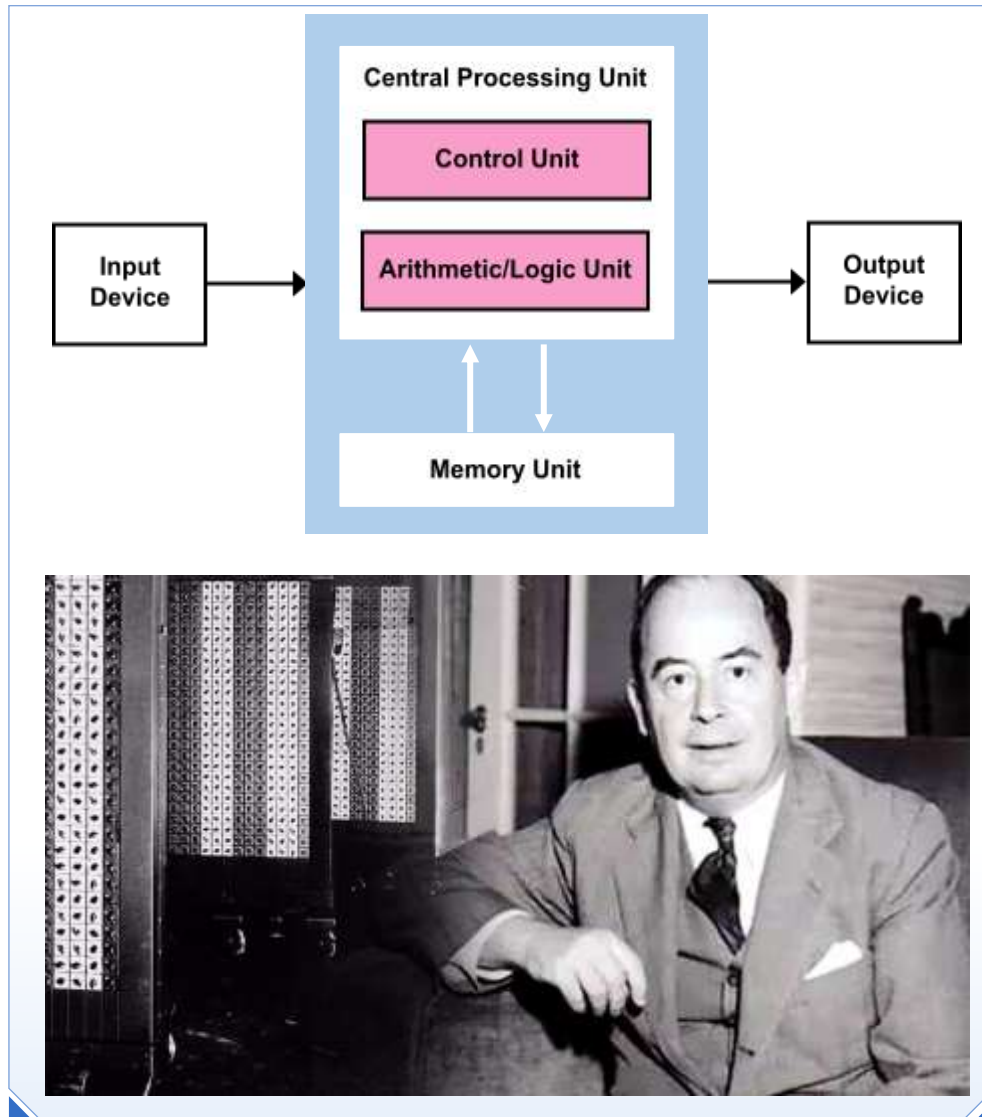
IBM360

当G. M. Amdahl于1964年推出世界上第一个系列计算机IBM360时，把**Computer Architecture**引入计算机领域

程序员所看到的计算机系统的属性，包括**概念性结构和功能特性**

- ▶ **程序员**：系统程序员（汇编语言、机器语言、编译程序、操作系统等）
- 看到的**：编写出能在机器上正确、高效运行的程序所必须了解的

概念性结构——“冯·诺依曼架构”



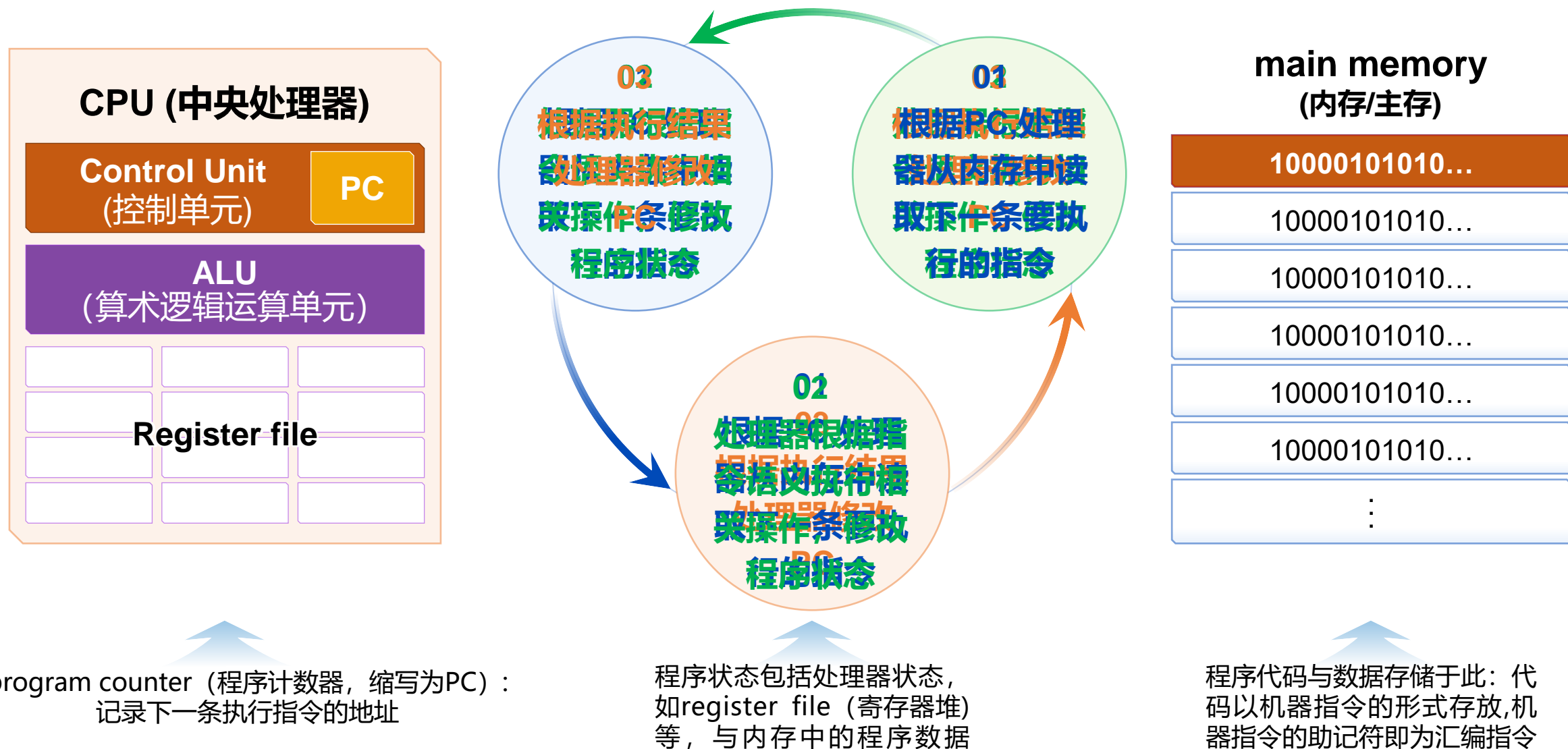
A design architecture for an electronic digital computer with parts consisting of a **processing unit containing an arithmetic logic unit and processor registers, a control unit containing an instruction register and program counter**, a **memory** to store both data and instructions, external mass storage, and input and output mechanisms.

John Von Neumann, 1903 - 1957 (From wiki)

**Long live the von Neumann
architecture!**

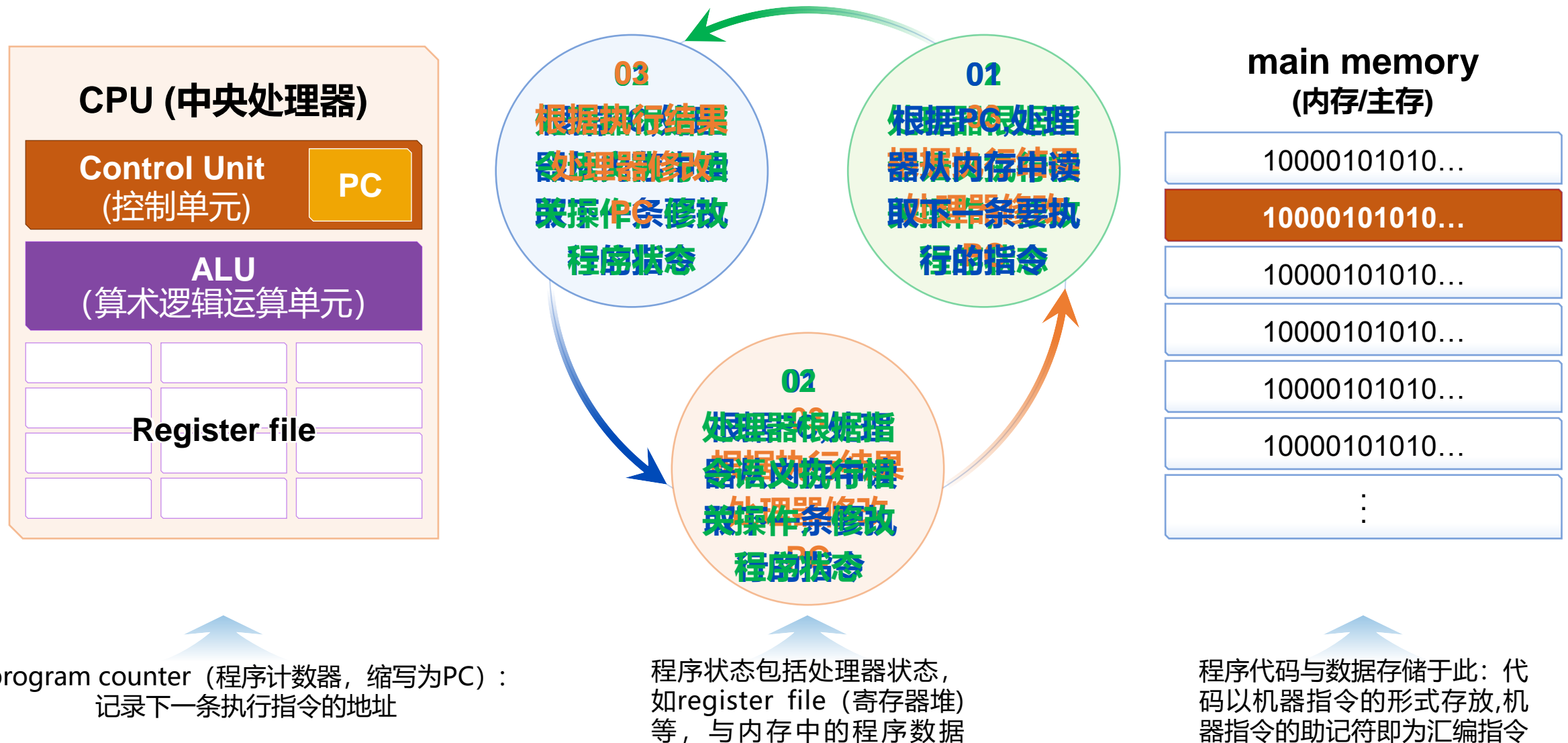
指令执行示意

冯诺依曼架构的程序执行特征：以指令为中心 / 存算分离



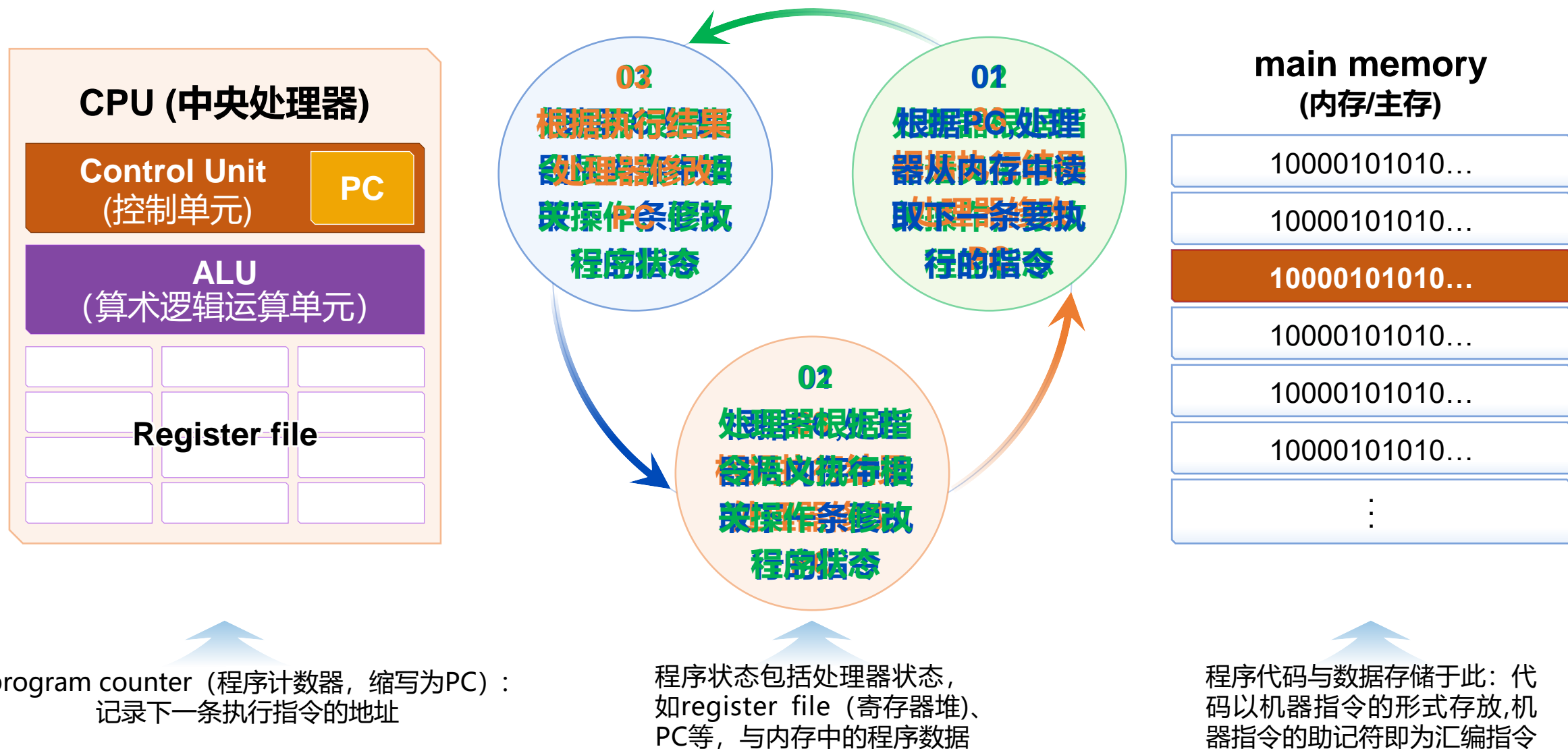
指令执行示意

冯诺依曼架构的程序执行特征：以指令为中心 / 存算分离



指令执行示意

冯诺依曼架构的程序执行特征：以指令为中心 / 存算分离



功能特性——指令系统及其执行模式

数据表示

硬件直接识别和处理的数据类型

寻址技术

编址方式、寻址方式和定位方式

寄存器定义

寄存器定义、数量和使用规则

指令系统

指令的操作类型、格式、操作数类型等

存储系统

要求速度高、容量大、价格便宜

异常处理系统

异常类型、级别和响应方式

输入输出系统

控制数据交换方式与交换过程

机器工作状态

定义、切换工作方式，如内核态、用户态等



IBM Compatibility Problem in early 1960s

By early 1960's, IBM had
4 incompatible lines of computers!

701	7094
650	7074
702	7080
1401	7010



Each system had its own

- Instruction set
- I/O system and Secondary Storage: magnetic tapes, drums and disks
- Assemblers, compilers, libraries,...
- Market niche: business, scientific, real time, ...

IBM System/360 – one ISA to rule them all



IBM 360: A Computer Family

	Model 30	...	Model 70
Storage	8K - 64 KB		256K - 512 KB
Datapath	8-bit		64-bit
Circuit Delay	30 nsec/level		5 nsec/level
Registers	Main Store		Transistor Registers

The IBM 360 is why bytes are 8-bits long today!

IBM 360 instruction set architecture (ISA) completely hid the underlying technological differences between various models.

Milestone: The first true ISA designed as portable hardware-software interface!

With minor modifications it still survives today!

From “50 years of Computer Architecture“, by David Patterson, May, 2017



广义的概念

广义概念

计算机体系结构(狭义)
Computer Architecture

程序员所看的计算机
系统的属性

计算机组成
Computer Organization

计算机系统的逻辑实现

计算机实现
Computer Implementation

计算机系统的物理实现

课程的一些重要概念



计算机程序



编译和链接

计算机程序是一组告诉计算机要做什么的指令 (from wiki)



编译

```
int array[4] = {1, 2, 3, 4};
static int brray[4] = {1, 2, 3, 4};

static int intra_sum (int x[4], int y)
{
    return x[y-1];
}

int main()
{
    int val = intra_sum(array, 3) + inter_sum(brray, 3);
    return val;
}
```

C语言程序 (.c)

编译:把用一种编程语言编写的计算机程序翻译成另一种语言
(from wiki)

```
intra_sum:
    movslq    %rsi, %rsi
    movl      -4(%rdi,%rsi,4), %eax
    ret

main:
    pushq     %rbx
    movl      $3, %esi
    movl      $array, %edi
    call      intra_sum
    movl      %eax, %ebx // val -> ebx
    movl      $3, %esi
    movl      $brray, %edi
    movl      $0, %eax
    call      inter_sum
    addl      %ebx, %eax // val -> eax
    popq      %rbx
    ret

brray:
    .long     1
    .long     2
    .long     3
    .long     4

array:
    .long     1
    .long     2
    .long     3
    .long     4
```

汇编程序 (.s)

汇编指令

```
0000000000000000 <intra_sum>:
0:  48 63 f6
3:  8b 44 b7 fc
7:  c3

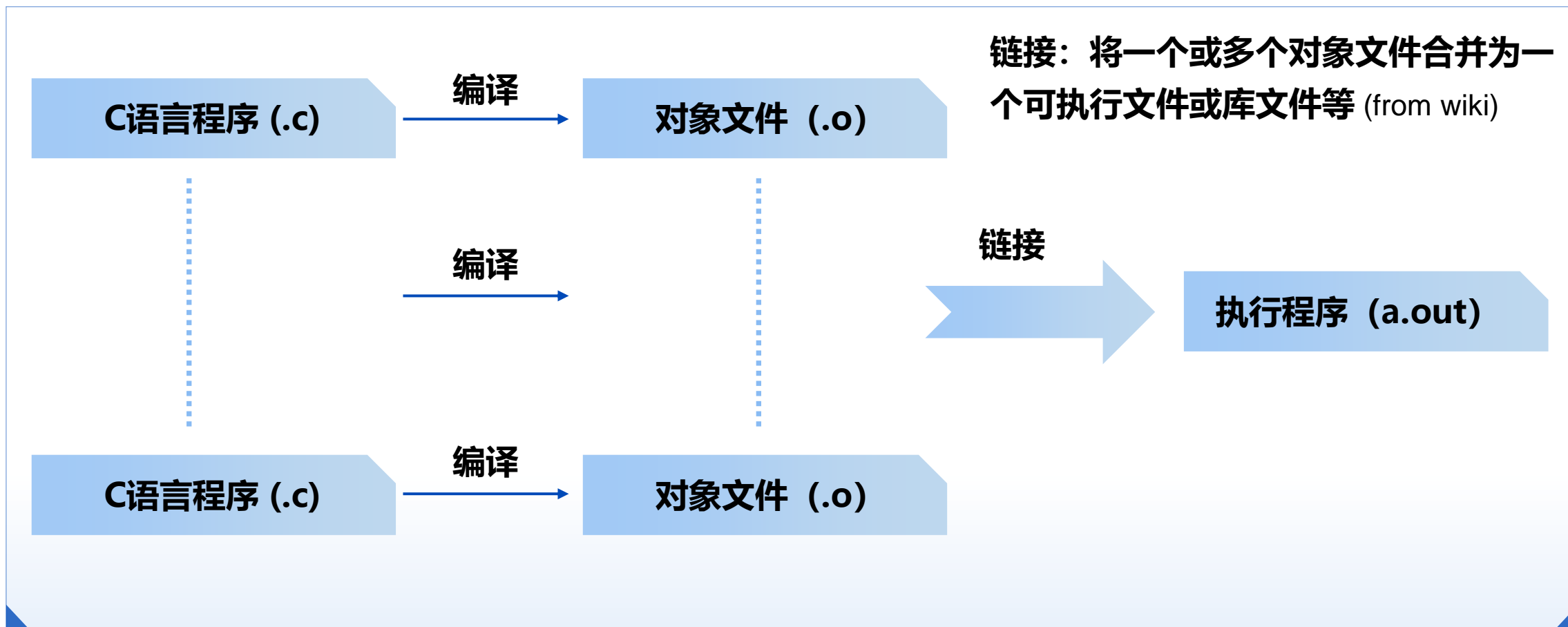
0000000000000008 <main>:
8:  53
9:  be 03 00 00 00
e:  bf 00 00 00 00
13: e8 00 00 00 00
18: 89 c3
1a: be 03 00 00 00
1f: bf 00 00 00 00
24: b8 00 00 00 00
29: e8 00 00 00 00
2e: 01 d8
30: 5b
31: c3
```

机器指令

数据部分没有显示

对象程序 (.o)

链接



程序的编译、链接、运行

```
int array[4] = {1, 2, 3, 4};
static int brray[4] = {1, 2, 3, 4};

static int intra_sum (int x[4], int y)
{
    return x[y-1];
}

int main()
{
    int val = intra_sum(array, 3) + inter_sum(brray, 3);
    return val;
}
```

main.c

编译

```
intra_sum:
    movlq %esi, %rsi
    movl -4(%rdi,%rsi,4), %eax
    ret

main:
    pushq %rbx
    movl $3, %esi
    movl $array, %edi
    call intra_sum
    movl %eax, %ebx// val -> ebx
    movl $3, %esi
    movl $brray, %edi
    movl $0, %eax
    call inter_sum
    addl %ebx, %eax// val -> eax
    popq %rbx
    ret

brray:
    .long 1
    .long 2
    .long 3
    .long 4

array:
    .long 1
    .long 2
    .long 3
    .long 4
```

main.o

链接

```
0000000004004de <main>:
4004de: 53                push %rbx
4004df: be 03 00 00 00    mov $0x3,%esi
4004e4: bf 40 10 60 00    mov $0x601040,%edi
4004e9: e8 e8 ff ff ff    callq 4004d6 <intra_sum>
4004ee: 89 c3             mov %eax,%ebx
4004f0: be 03 00 00 00    mov $0x3,%esi
4004f5: bf 30 10 60 00    mov $0x601030,%edi
4004fa: b8 00 00 00 00    mov $0x0,%eax
4004ff: e8 04 00 00 00    callq 400508 <inter_sum>
400504: 01 d8             add %ebx,%eax
400506: 5b               pop %rbx
400507: c3               retq
```

汇编指令

内存地址

仅给出部分内容

运行

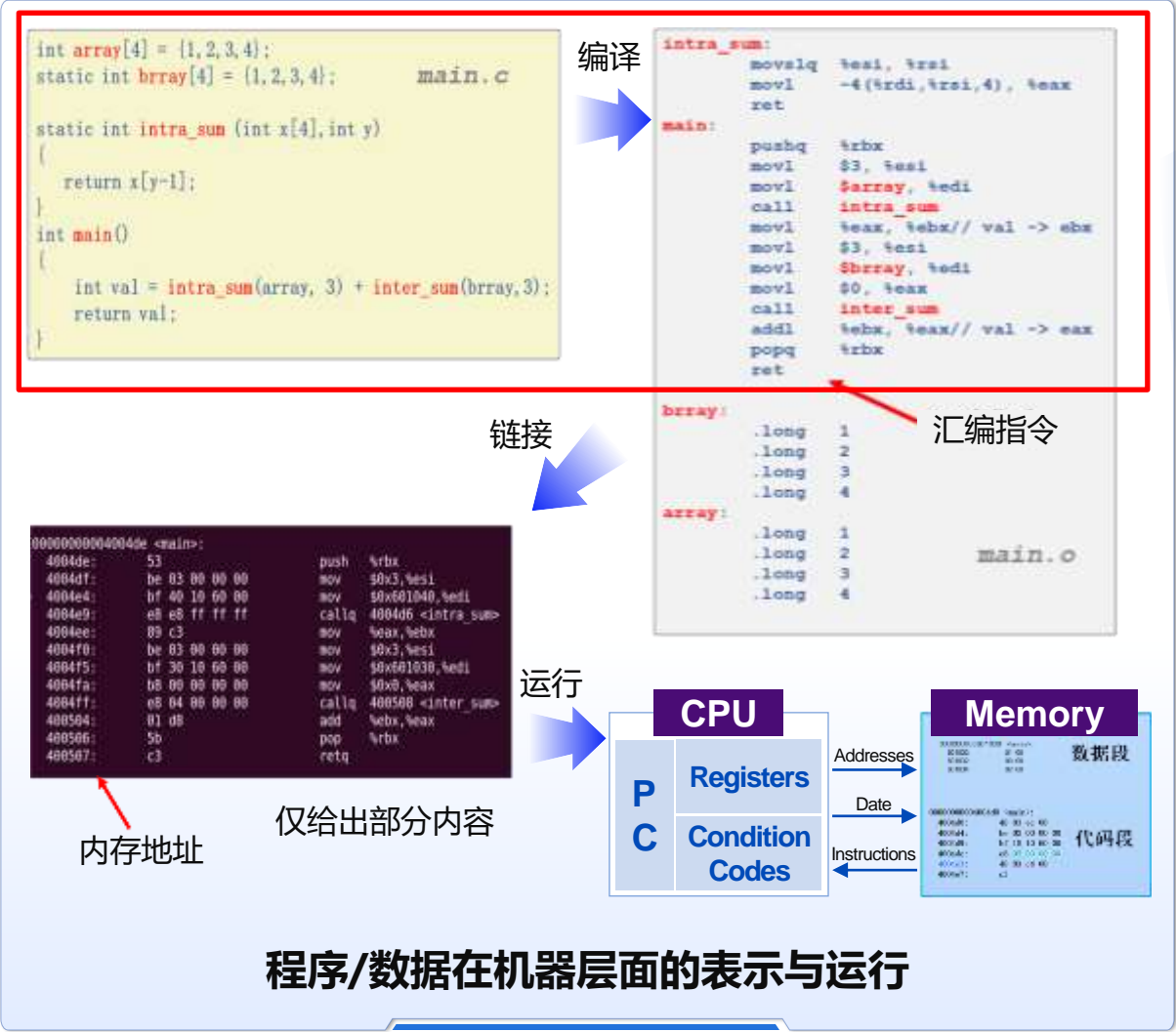


程序/数据在硬件层面的表示与运行



课程章节安排





C程序在硬件层面的表示

数据

- ▶ 整数 / 浮点数 (第二讲)
- ▶ 数组 / 结构 (第五讲)

代码

- ▶ 基本概念/基本指令/寻址方式/程序控制流与相关指令 (第三讲)
- ▶ 函数调用与相关指令 (第四讲)

课程章节安排

C程序在硬件层面的表示

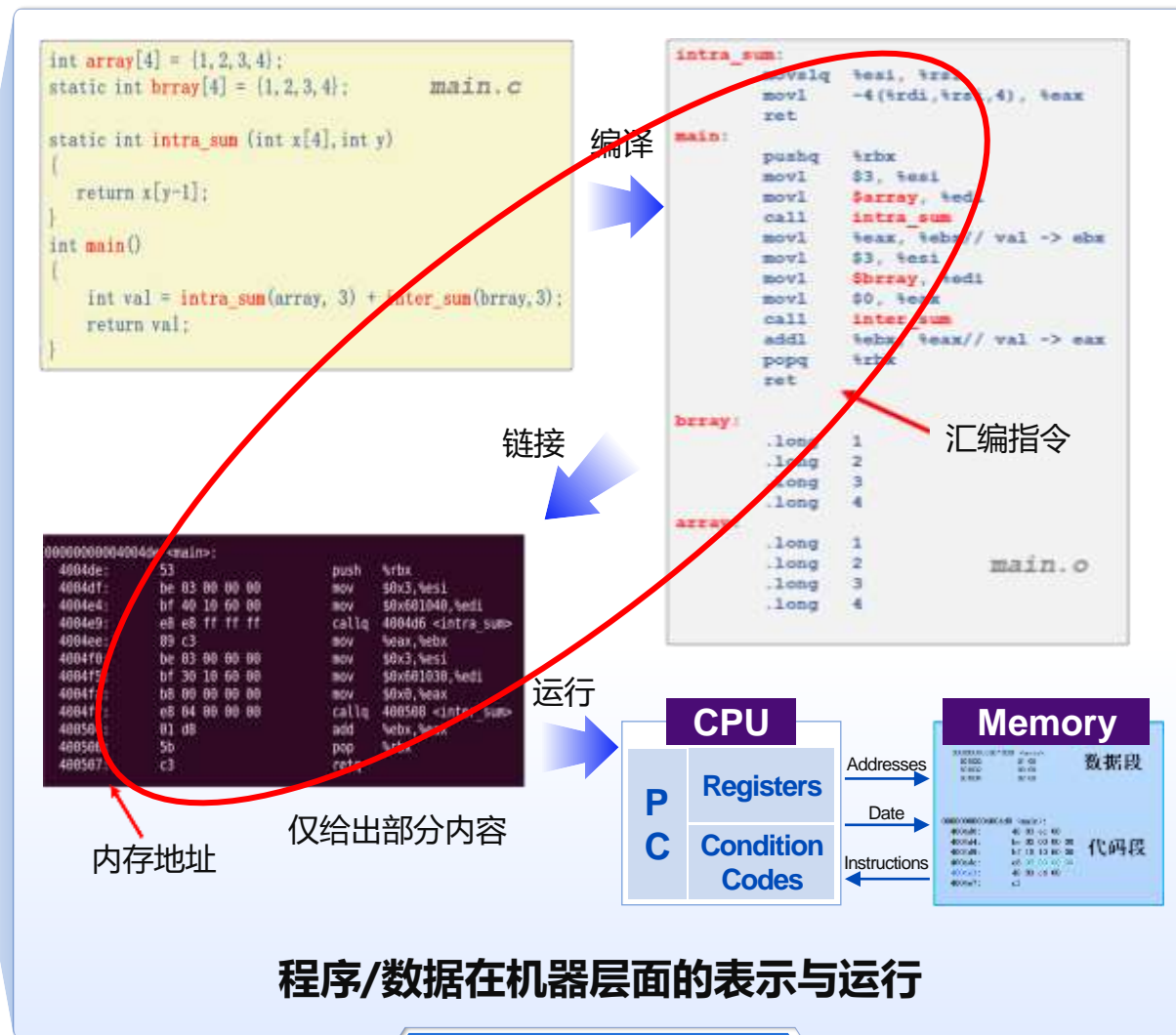
数据/函数的内存地址定位

- ▶ 链接 (第六讲)

数据/代码的内存布局

- ▶ 栈、堆等各类数据段以及代码段的layout (第六讲)
- ▶ 缓冲区溢出等 (第六讲)

讲解基本调试工具GDB等的使用



课程章节安排

(看上去)



一个程序占据了整个处理器及一块完整的内存空间，但是实际情况（多任务）要远为复杂！

怎么做到的？

关系到计算机系统的两个重要概念

- ▶ **虚存** - 在有限物理内存前提下设计出连续的、相互独立的虚拟内存
- ▶ **异常** - 各个任务切换的重要机制
(当然异常还有很多其他作用)



虚 存

在有限物理内存前提下设计出
连续的、相互独立的虚拟内存

异常

任务切换的重要机制
(当然异常还有很多其他作用)

虚存与虚存管理 (第七、八讲)

- 包括用户层动态内存分配

异常与异常控制流 (第九讲)

- 进程上下文切换
- 进程层次结构
- 信号处理

《——— 其他内容 (第十、十一讲) ———》

系统调用：系统级IO、线程与并发、线程简单编程

感谢各位同学 积极参与

⑧ Zhang, Youhui (张悠慧)

✉ zyh02@tsinghua.edu.cn

