

# 计算机组成原理

# 课程概要

李曦

llxx@ustc.edu.cn



# 李曦： Who's Who😊



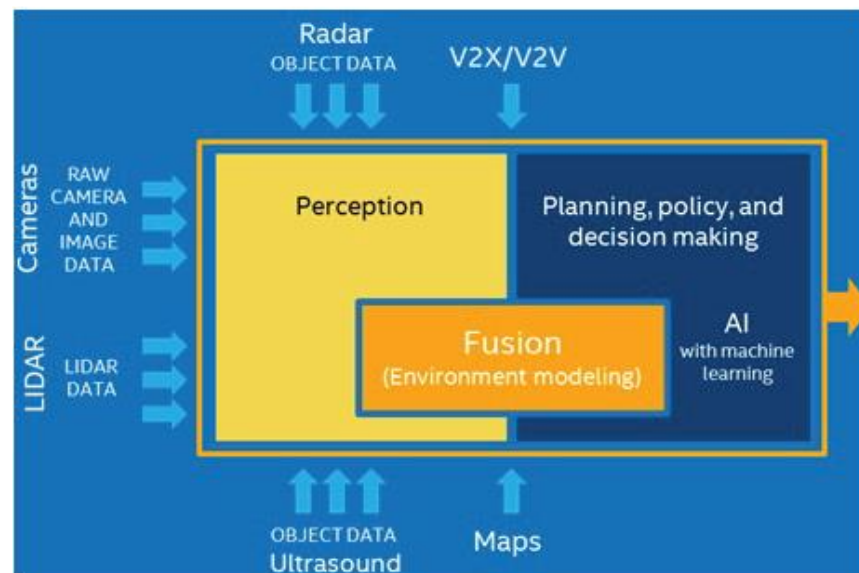
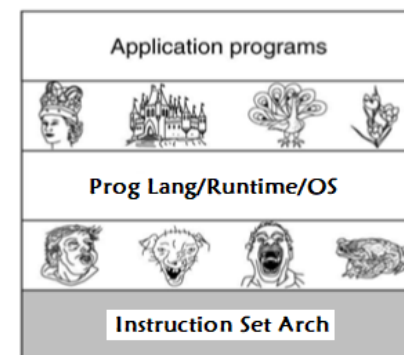
- 教师@软件学院/计算机学院

- 博士，教授，博导

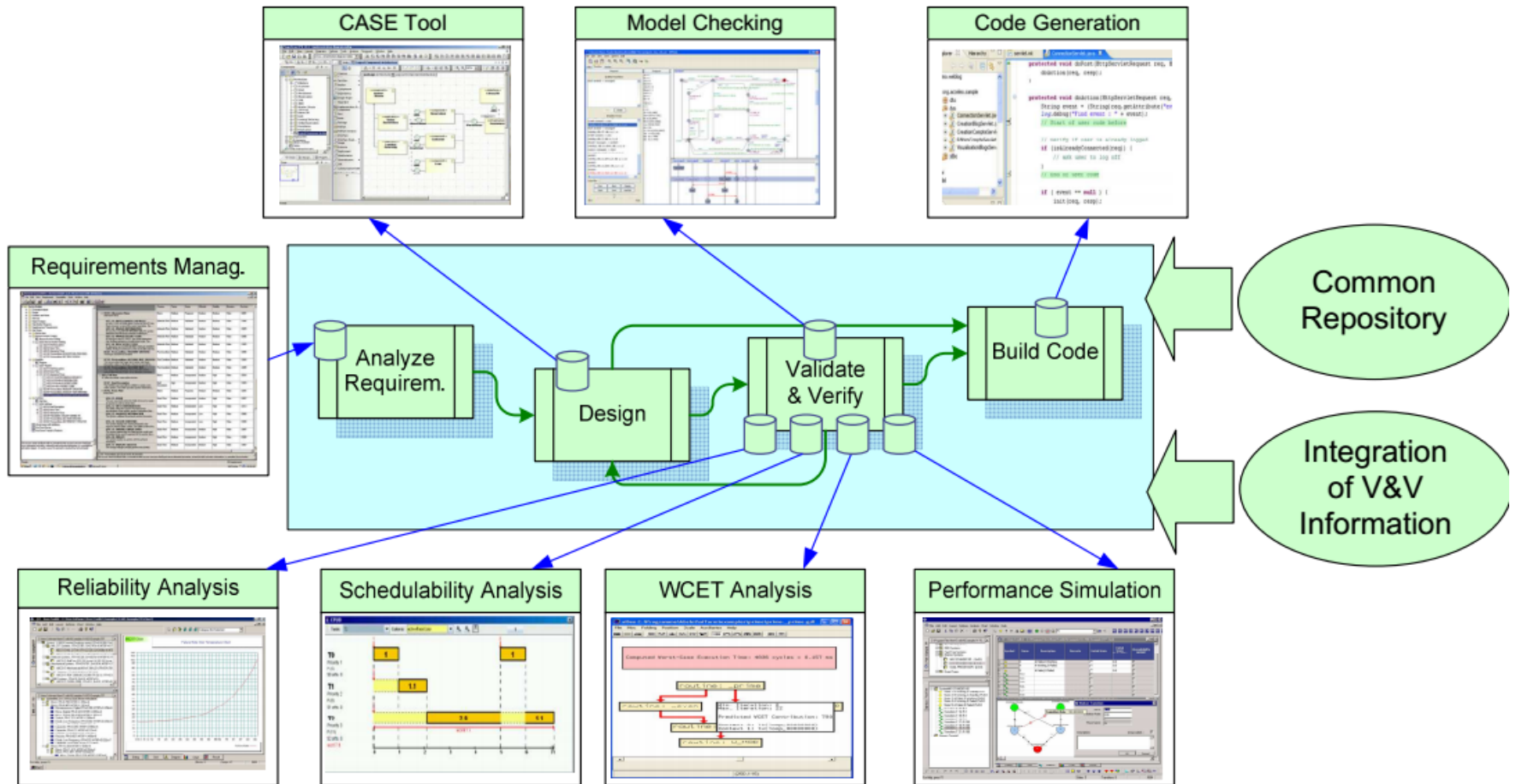
- 研究方向：计算机系统结构

- 实时系统，可重构系统，嵌入式智能系统

- 科学问题：系统的时间行为可预测性与可组合性



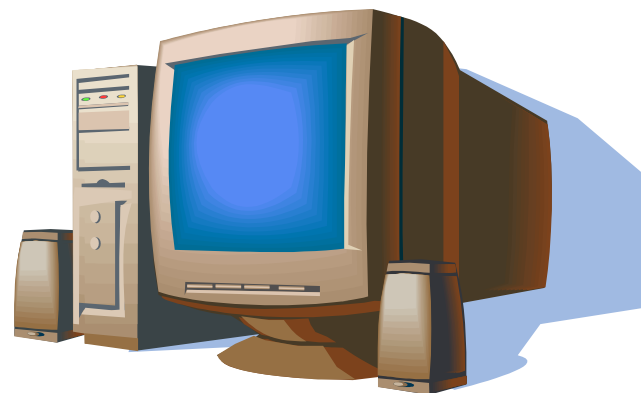
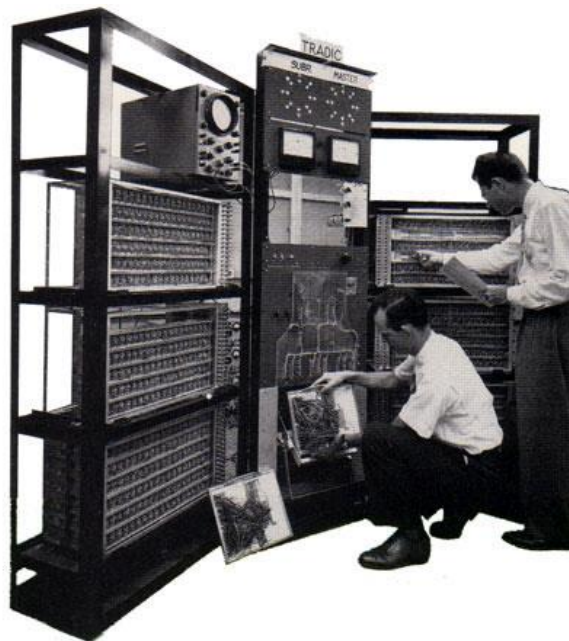
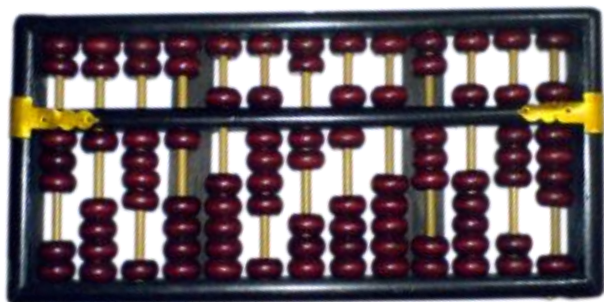
# Making Things Easy and Predictable ☹



A Framework of Collaborative Environments, from SPEC to Implementation.



# 计算机是什么？



算盘 = 计算器  $\neq$  计算机?  
Tiny?

# 计算机在干啥？



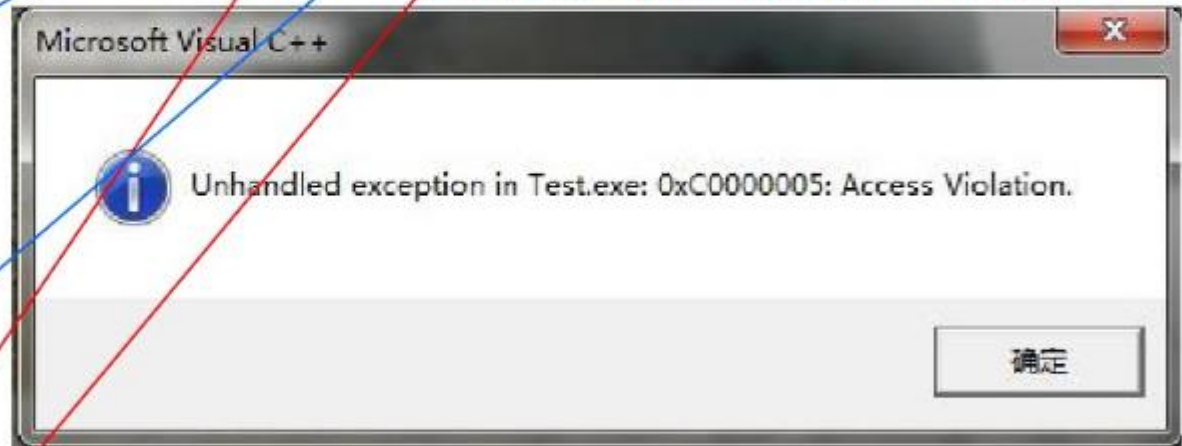


# 程序如何执行的？用到哪些部件？

```
/*---sum.c---*/  
int sum(int a[ ], unsigned len)  
{  
    int i, sum = 0;  
    for (i = 0; i <= len-1; i++)  
        sum += a[i];  
    return sum;  
}
```

```
/*---main.c---*/  
int main()  
{  
    int a[1]={100};  
    int s;  
    s=sum(a,0);  
    printf(“%d”,s);  
}
```

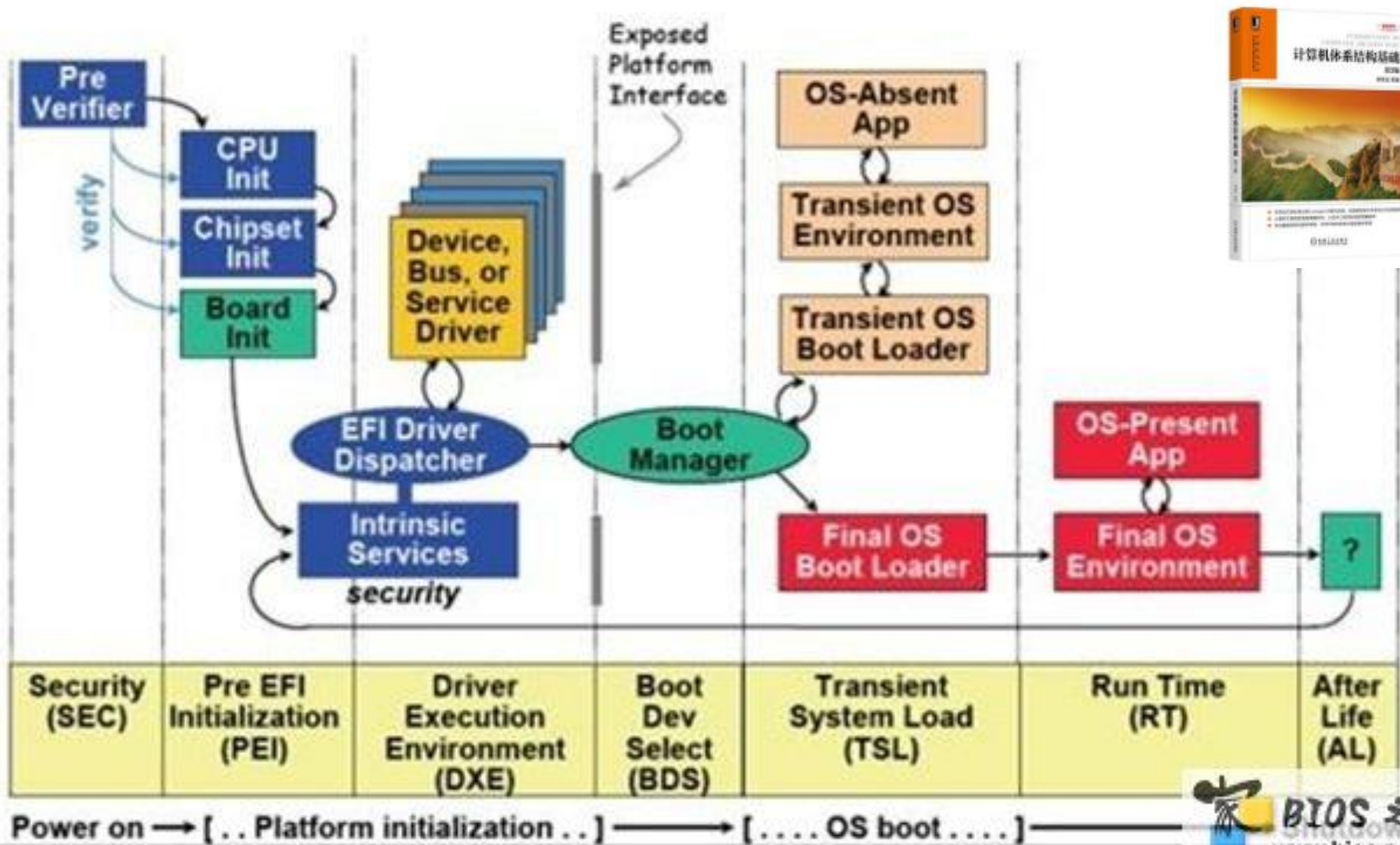
- 数据的表示
- 数据的运算
- 各类语句的转换与表示(指令)
- 各类复杂数据类型的转换表示
- 过程(函数)调用的转换表示



- 链接(linker)和加载
- 程序执行(存储器访问)
- 异常和中断处理
- 输入输出(I/O)

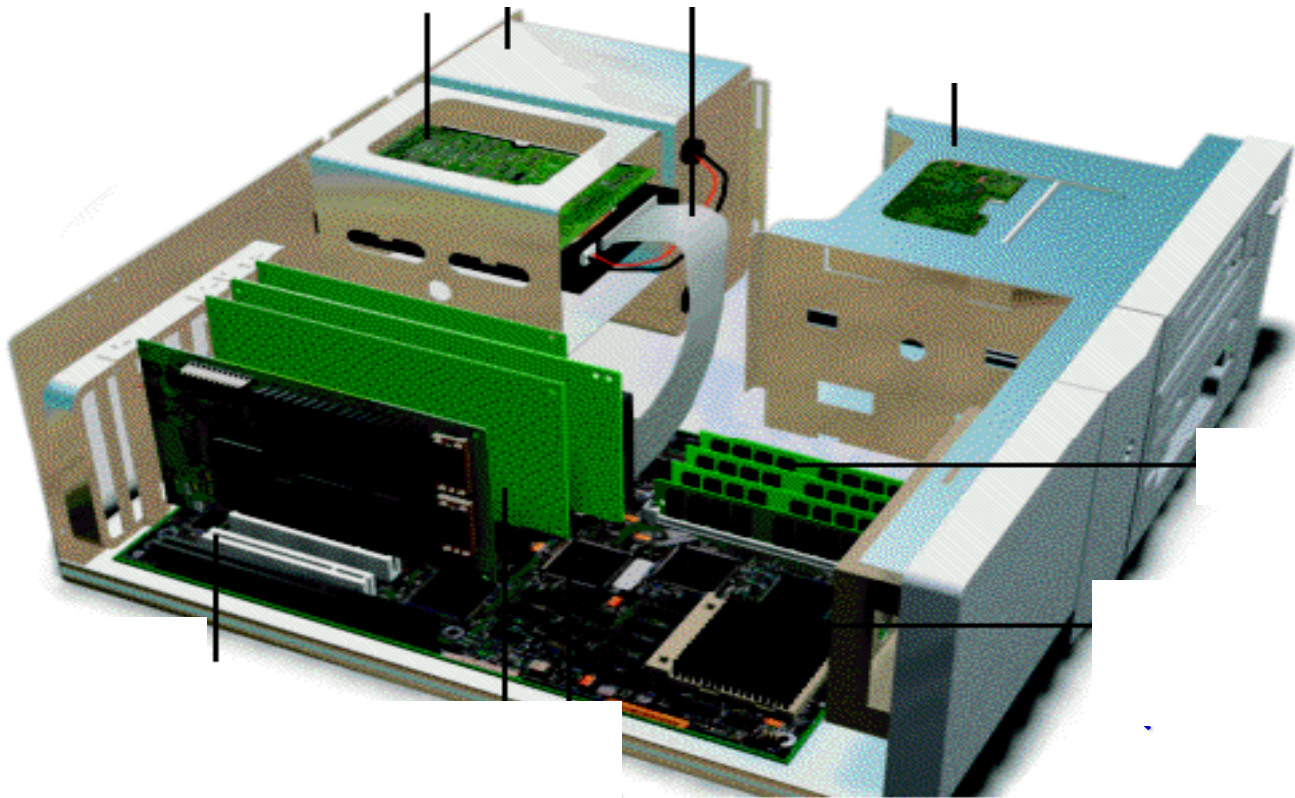
• C语言计算机？

# 何时可以执行你的C程序？ 胡伟武Ch7

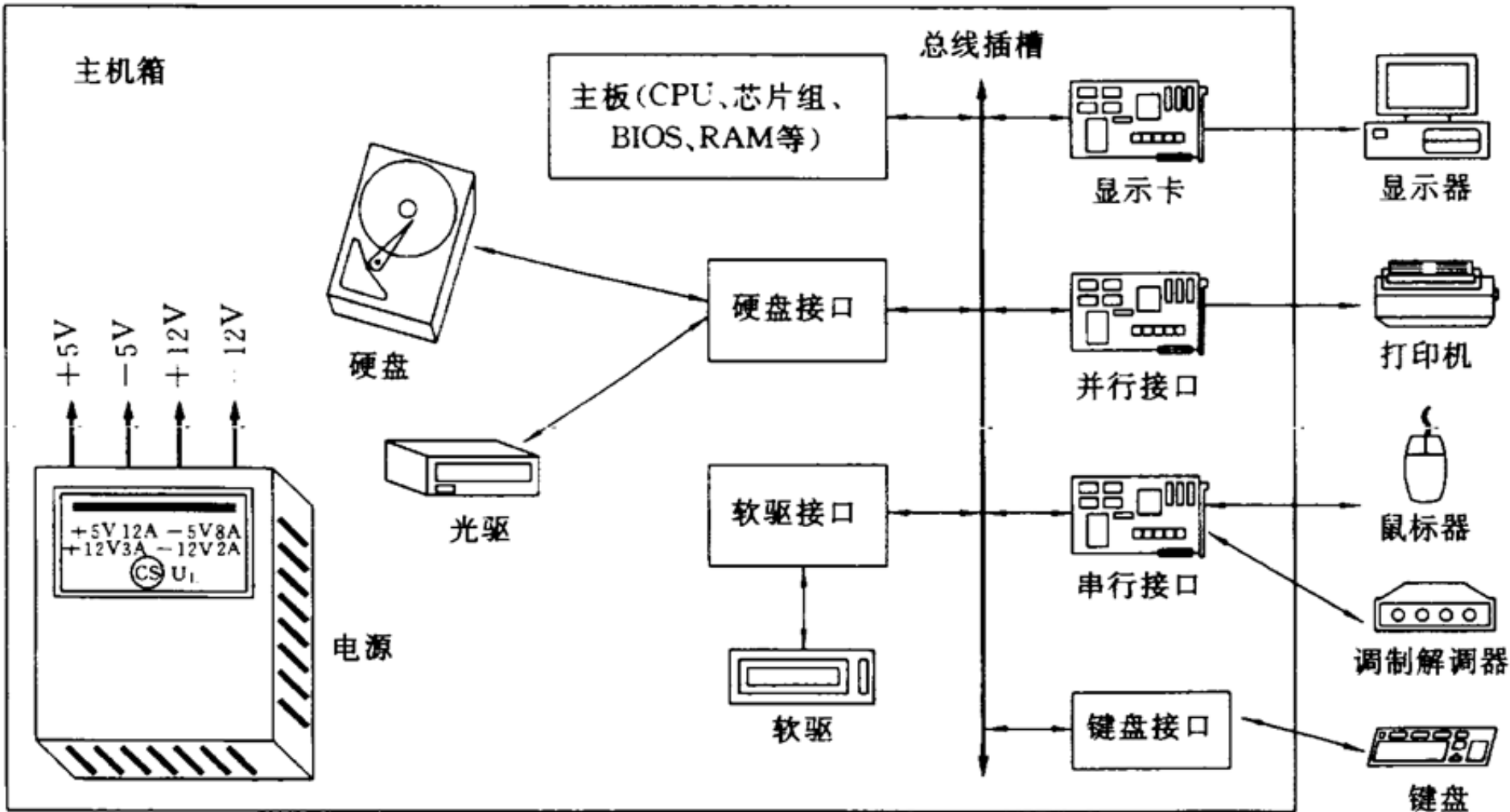




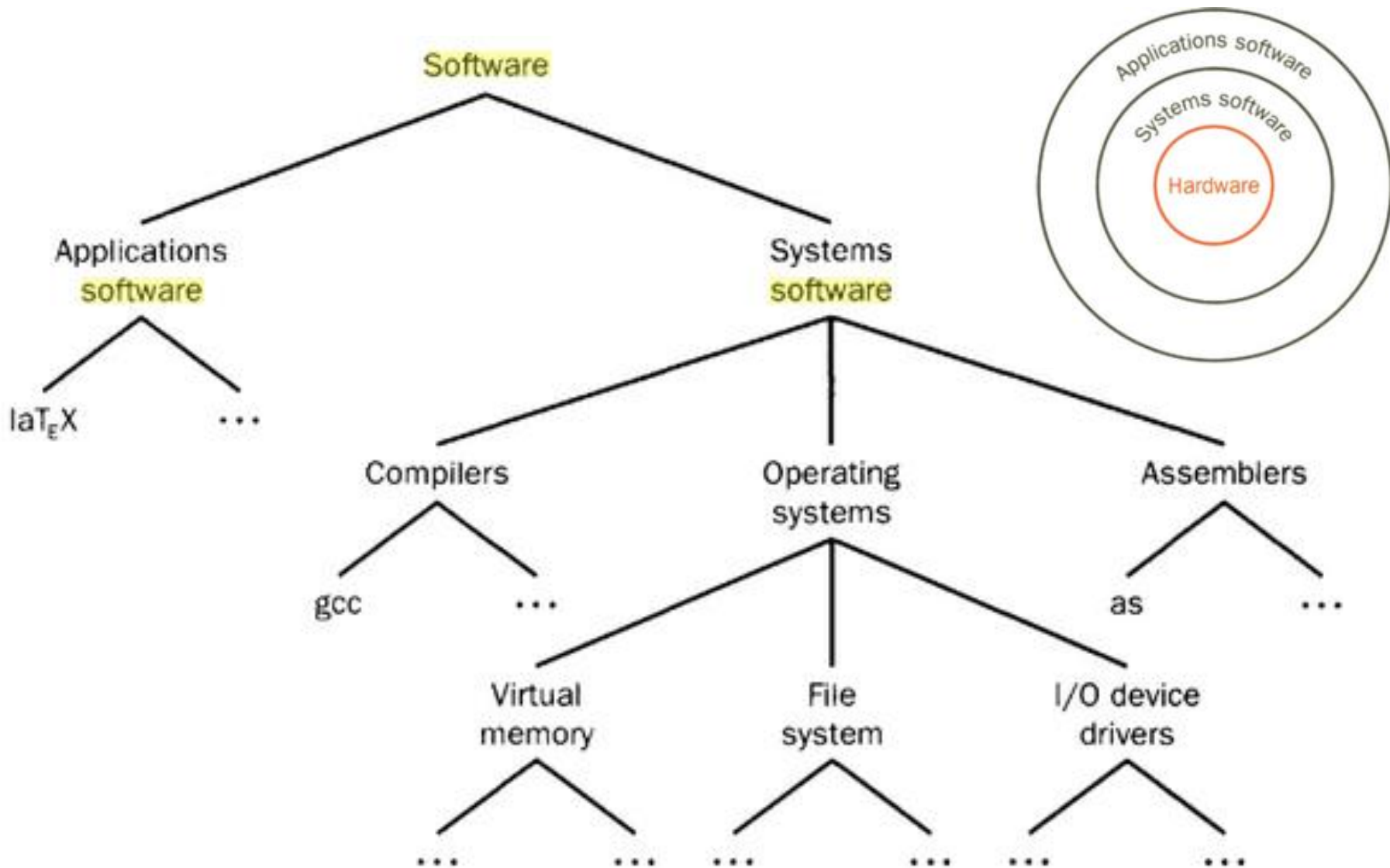
# 计算机组成 (CBD)



# 计算机组成：硬件与接口



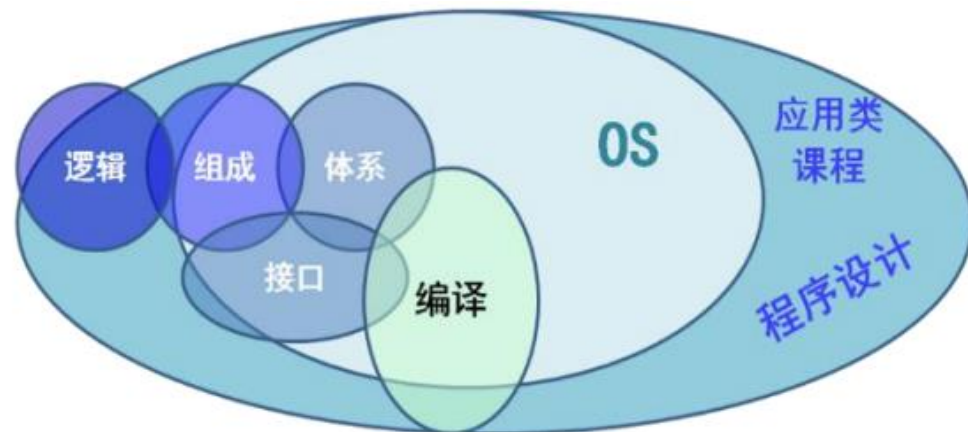
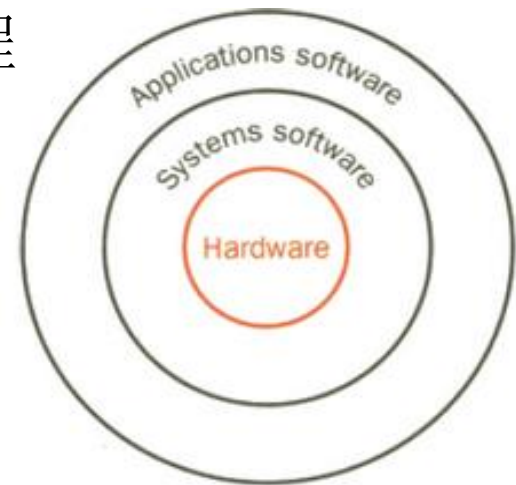
# 计算机组成：软件与软硬件接口



# COD课程的作用： what, how, why



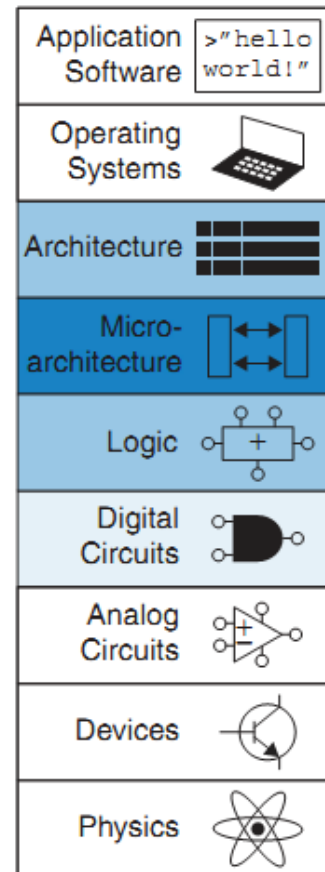
- 专业核心课：所有专业课的基础
  - 所有大学计算机系都开设COD相关课程
- 密切相关课程
  - 数字逻辑电路，程序设计
  - 组成原理
    - 重点“单处理器单核”系统
      - 介绍多核、多处理器等并行系统
    - 强调软硬件接口
  - 体系结构
  - 操作系统
  - 编译原理
  - 微机原理/嵌入式系统



# 课程标准： CS2013/CE2016



- *IEEE/ACM Computer Science Curriculum 2013*
  - ARKA含8个KU(5个必修, 3个选修)
    - 数字逻辑与系统
    - 数据表示
    - 汇编级机器组成
    - 存储系统组成与结构
    - 接口与通信
    - 功能组成
    - 多处理及结构
    - 性能优化
  - CE2016增加
    - “标准与工具”
    - “分布式系统结构, 并行与分布式算法”
- 两个层次
  - system architecture
    - the organization of the computer **at the cache and bus level**
  - organization/microarchitecture
    - the internal implementation of a computer **at the register and functional unit level**



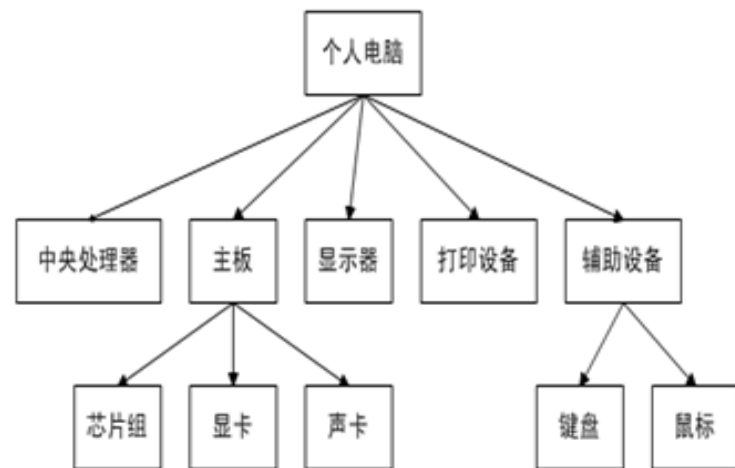
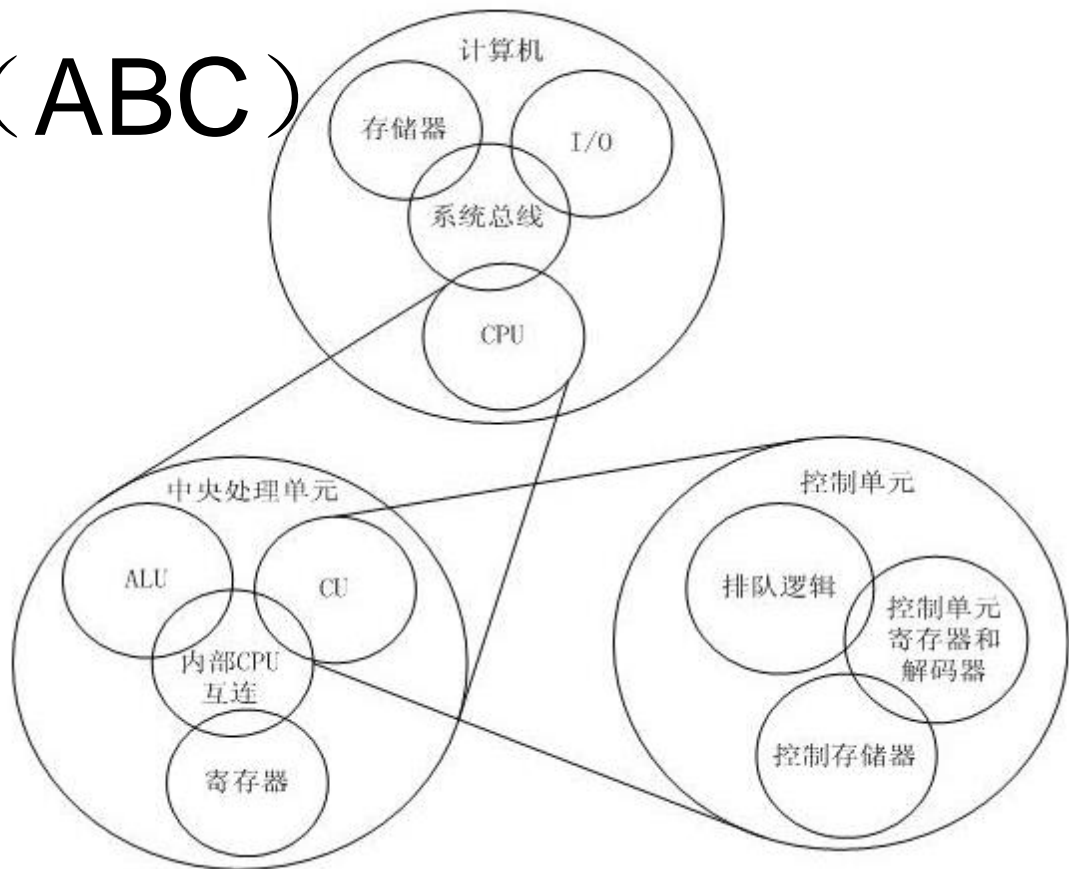
# 课堂教学内容 (ABC)

- 中央处理器 (CPU)

- 数据表示
- 算术逻辑运算
- 指令系统
- 数据通路
- 控制单元

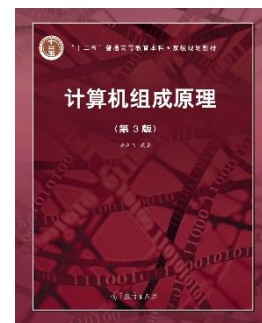
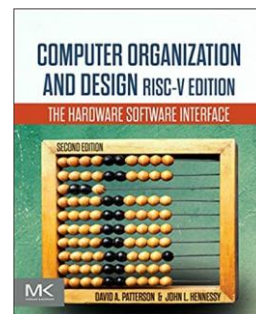
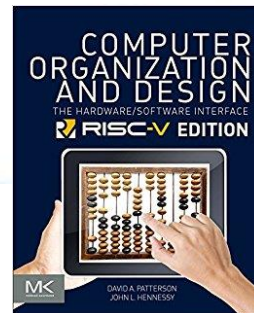
- 外围部件

- 系统总线
  - 总线性能、总线结构、总线控制
- 存储器
  - 主存、高速缓存、辅存、虚存
- 输入输出系统
  - I/O接口、I/O控制方式、外设



# 教材与参考书

- 《Computer Organization and Design: The HW-SW I/F》
  - David Patterson and John Hennessy, 5th, RV版, 2018/2020
    - CPU和Cache强, MEM、总线、I/O弱
    - **COD**涉及并行和性能分析: 一些学校用作Computer architecture教材
- 《计算机组成原理》, 2nd/3rd, 2008/2020, 唐朔飞
  - 以William Stallings Ed. 4th为框架, 参考7th和COD 3rd
    - CISC, A模型CPU, 但无OS、VM、并行等内容
    - CPU和Cache设计弱, MEM、总线、I/O强
- “Computer organization and architecture: Designing for performance”, **COA**
  - Dr. William Stallings(澳大利亚新南威尔士大学)
    - 1996第4版, 2000第5版, 2003第6版(2005张昆藏中译本);
    - 2005第7版, 2010第8版(彭蔓蔓译, 电子版)
    - 2012第9版
    - 2015第10版(目录同9版, 增加一些新资料。电子版)
      - The computer system, Arithmetic and logic, The central processing unit
      - Parallel organization, including multicore
      - The control unit (online!)
  - Ed. 8th前偏organization, 之后偏architecture

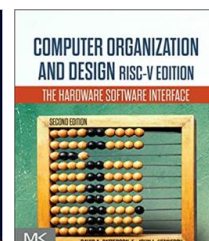
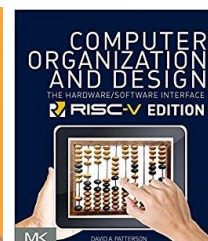
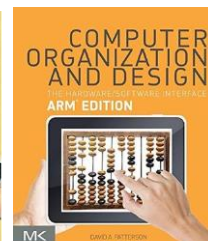
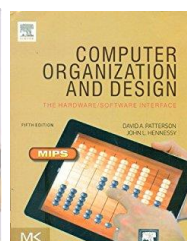
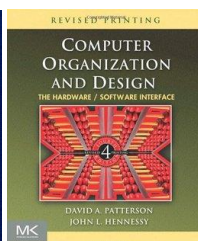
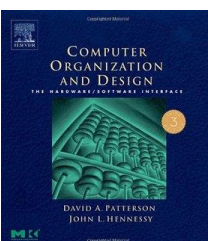
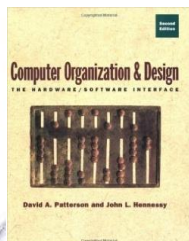


# Patterson and Hennessy 2018图灵奖



- “COD: The Hardware/ Software Interface”

- 1<sup>st</sup> Edition, 1994
- 2<sup>nd</sup> Edition, 1997
- 3<sup>rd</sup> Edition, 2004, 郑纬民译
  - Covers the revolutionary change from sequential to parallel computing
- 4<sup>th</sup> Edition, 2009, 2011康继昌译
  - the **switch** from uniprocessor to multicore microprocessors
  - make parallelism a first class citizen(spread parallelism throughout all the chapters)
- 5<sup>th</sup> Edition, 2013 (2016MIPS/ARM, 2018**RV64**, 2020**RV32**)
  - moves forward into the **post-PC** era with new examples, exercises, and material highlighting the emergence of mobile computing and the cloud.
  - This **generational change** is emphasized and explored with updated content featuring tablet computers, cloud infrastructure, and the ARM (mobile computing devices) and x86 (cloud computing) architectures.
  - features the **RISC-V** open source instruction set architecture, the **first** open source architecture designed for use in modern computing environments such as cloud computing, mobile devices, and other embedded systems.







# 教材主要内容比较

## COD5: RISC-V

1. 计算机概念与工艺
  2. 指令
  3. 算逻运算
  4. RISC处理器
    - 单周期、多周期、流水线
    - 异常: CP0
    - 微程序: COD5 (C.9)
  5. 存储器层次结构
    - SRAM、DRAM、FLASH、DISK
    - Cache、Cache控制器
    - 可靠性问题 (海明码)
    - 虚存
  6. 并行处理器
    - 硬件多线程
    - 多核
- COD5: “I/O不单独成章” !
- 总线?

## • COA9, 10: 以CISC为主

- 第一部分 概论
    - 性能
  - 第二部分 计算机系统
    - 总线, 内存, Cache, 外存, I/O
    - OS (虚存) (唐本无)
  - 第三部分 CPU
    - 算逻运算、ISA, CPU结构
    - 介绍RISC、流水线概念
  - 第四部分 CU
    - A模型处理器
    - 多周期: 硬连线实现, 微程序实现
  - 第五部分 并行 (唐本无)
- 强调系统完整性: 总线/内存/I/O
- 考研: 唐朔飞or白中英

COD5 (CPU、Cache、存储器、异常、MMU) +唐 (总线、I/O、中断、DMA)

# Course Schedule



Lec	Topic	Read BEFORE class		Keypoint
		RV1	唐	
1	概论	CHP1		
	逻辑设计基础	4.2、附录 A		
2	指令系统, 汇编程序设计	CHP2, 5.14	CHP7	冒泡排序
	RISC 概述	附录 D		
3	RISC 处理器 (单周期、多周期)	4.4、RV32 4.5		
	控制器 (组合, 顺序, 微程序)	附录 C	微程序 10.2	
	RISC 处理器 (流水线)	4.5、4.13		
	流水线冒险	4.6~4.8、4.13		
	多发射 (静态、动态), HMT	4.10、6.4		
	异常与中断	4.9	8.4	
6	SRAM, DRAM, ROM, SDRAM	5.2、A.9	第四章	
	Cache 原理	5.3、5.4	4.3	
	Cache Controllers	5.9		
	外存	5.2, 5.11	第四章	
	虚存	5.7		
	存储器的可靠性	5.5、5.11、A.9	第四章	
7	总线	6.9	第三章	
8	I/O	1.4、6.9	第五章	TinyComputer
9	A 模型 CPU 设计(多周期, 微程序)		第八/九/十	
10	算术运算	CHP3	第六章	
11	专题: 并行, 非冯机, 低功耗	CHP1、CHP6		

# 课程成绩考核：发现抄袭取消成绩



- 课堂教学：70%
  - 期末考试：50%（闭卷）
  - 作业：30%，不能补交！
  - 课堂练习：10%（出勤5%，正确5%）
  - 课内实验：5%
    - 冒泡汇编，Tiny系统设计
    - 实验报告：参照小论文格式
  - 调研：2~3篇，5%
    - 调研报告：参照小论文格式
    - 资料来源：参考书、Internet
- 实验课：30%



# 课内实验

```
-d
0B06:0100 36 B9 98 BF 76 97 B8 00-29 CD 21 80 3C 00 74 06 6...v...>.!<.t.
0B06:0110 BA 92 8A E9 53 FF A1 BC-98 80 FC 3A 34 00 F5 0A .....S.....:4...
0B06:0120 8A 0E D8 99 0C 20 2C 60-A2 76 97 A0 BB 98 8A 26 .....`..v....&
0B06:0130 1E 99 25 02 02 0A C0 74-13 3A C4 75 0F 80 3E D9 ..%.....t.:.u..>.
0B06:0140 99 00 74 08 FE 06 17 99-32 C0 EB 06 34 02 22 C4 ..t.....2...4..."
0B06:0150 D0 E8 0A 06 D9 99 A2 C7-96 D0 E0 D0 E0 A2 D2 99 .....
0B06:0160 80 3E D4 99 00 75 24 A2-D8 99 0A C9 75 1D 0A C0 ->...u$......u...
0B06:0170 74 19 8B 0E D5 96 E3 13-B0 1A 06 33 FF 8E 06 B4 t.....3....
```

- **【必选】** 汇编语言程序设计：任意10个数冒泡排序
  - 基于RISC-V仿真器Ripes
  - 成果要求：
    - 报告设计过程、结果（汇编代码、内存数据段映像）、出现的典型问题及解决过程；
    - 要求结果体现**个人ID**：10个数中含**姓名拼音**字母
- **【任选】** 基于Xilinx ISE，设计一个TinyComputer
  - 系统硬件组成：包含CPU、on\_chip\_ram和JTAG UART三个模块；
  - 编写并运行一个简单C程序“hello\_world\_small”；
  - 成果要求：
    - 报告设计过程、结果（原理图、仿真结果、源代码）、出现的典型问题及解决过程；
    - 要求结果体现**个人ID**：“hello\_world\_small—**llxx**”

# Ripes

File Edit Help

100 1010 0+ Editor

Processor

Memory

Source code

```

1 # This example shows an implementation of the mathematical
2 # factorial function (! function) to find the factorial value of !7 = 5040.
3
4 .data
5 argument: word 7
6 str1: .string "Factorial value of ".
7 str2: .string " is "
8
9 .text
10 main:
11     lw a0, argument # Load argument from static data
12     jal ra, fact    # Jump-and-link to the 'fact' label
13
14     # Print the result to console
15     mv a1, a0
16     lw a0, argument
17     jal ra, printResult
18
19     # Exit program
20     li a7, 10
21     ecall
22
23 fact:
24     addi sp, sp, -16
25     sw ra, 8(sp)
26     sw a0, 0(sp)
27     addi t0, a0, -1
28     bge t0, zero, nfact
29
30     addi a0, zero, 1
31     addi sp, sp, 16
32     jr x1
33
34 nfact:
35     addi a0, a0, -1
36     jal ra, fact
37     addi t1, a0, 0
38     lw a0, 0(sp)
39     lw ra, 8(sp)
40     addi sp, sp, 16
41     mul a0, a0, t1
42     ret
43

```

Input type: Assembly Executable code

View mode: Binary Disassembled

```

10: 10000517 auipc x10 0x10000
14: ff925203 lw x10 -16(x10)
18: 04c0000f jal x10 0x64 <printResult>
1c: 00c00893 addi x17 x0 10
20: 00000073 ecall

00000024 <fact>:
24: ff010113 addi x2 x2 -16
28: 00112423 sw x1 8(x2)
2c: 00a12023 sw x10 0(x2)
30: fff5e293 addi x5 x10 -1
34: 00020863 bge x5 x0 16 <nfact>
38: 00100513 addi x10 x0 1
3c: 01010113 addi x2 x2 16
40: 00000067 jalr x0 x1 0

00000044 <nfact>:
44: fff50513 addi x10 x10 -1
48: fd9ff0ef jal x1 0x24 <fact>
4c: 00050313 addi x5 x10 0
50: 00125093 lw x10 0(x2)
54: 00a12083 lw x1 8(x2)
58: 01010113 addi x2 x2 16
5c: 02650533 mul x10 x10 x6
60: 00000067 jalr x0 x1 0

00000064 <printResult>:
64: 00050293 addi x5 x10 0
68: 00050313 addi x6 x11 0
6c: 10000517 auipc x10 0x10000
70: f9950513 addi x10 x10 -104
74: 00400893 addi x17 x0 4
78: 00000073 ecall
7c: 00020513 addi x10 x5 0
80: 00100893 addi x17 x0 1
84: 00000073 ecall
88: 10000517 auipc x10 0x10000
8c: f9950513 addi x10 x10 -112
90: 00400893 addi x17 x0 4
94: 00000073 ecall
98: 00030513 addi x10 x5 0
9c: 00100893 addi x17 x0 1
a0: 00000073 ecall

```

File Edit Help

100 1010 0+ Editor

Processor

Memory

5-Stage RISC-V Processor

addi x10 x0 1    bge x5 x0 16 <nfact>    addi x5 x10 -1    sw x10 0(x2)    sw x1 8(x2)

Registers

Name	Alias	Value
x0	zero	0x00000000
x1	ra	0x0000000c
x2	sp	0x7ffff7f8
x3	gp	0x10000000
x4	tp	0x00000000
x5	l0	0x00000000
x6	l1	0x00000000
x7	l2	0x00000000
x8	s0	0x00000000
x9	s1	0x00000000

Display type: Hex

Instruction memory

BP	Addr	Stage	Instruction
<input type="checkbox"/>	0x28	WB	sw x1 8(x2)
<input type="checkbox"/>	0x2c	MEM	sw x10 0(x2)
<input type="checkbox"/>	0x30	EX	addi x5 x10 -1
<input type="checkbox"/>	0x34	ID	bge x5 x0 16 ...
<input type="checkbox"/>	0x38	IF	addi x10 x0 1
<input type="checkbox"/>	0x3c		addi x2 x2 16
<input type="checkbox"/>	0x40		jalr x0 x1 0
<input type="checkbox"/>	0x44		addi x10 x10 -1
<input type="checkbox"/>	0x48		jal x1 0x24 <nfact>

Console

Execution info

Cycles: 10  
 Instrs. retired: 4  
 CPI: 2.5  
 IPC: 0.4  
 Clock rate: 2.13 Hz



# 调研/课内实验：小论文格式

- 题目（宋体，小二，加粗）
- 姓名、学号（宋体，小四）
- 摘要（宋体，小四）
- 引言（宋体，小四）
- 正文（宋体，小四）
  - 问题，过程，结果
- 结论（宋体，小四）
- 参考文献（宋体，小五）  
（不少于**3**篇，中英文皆可）



# 实验课：1学分

- Xilinx ISE：逻辑仿真

- Digilent Nexys3平台

- Spartan-6 FPGA

- 48Mb片外存储器：含2个非易失相变存储器

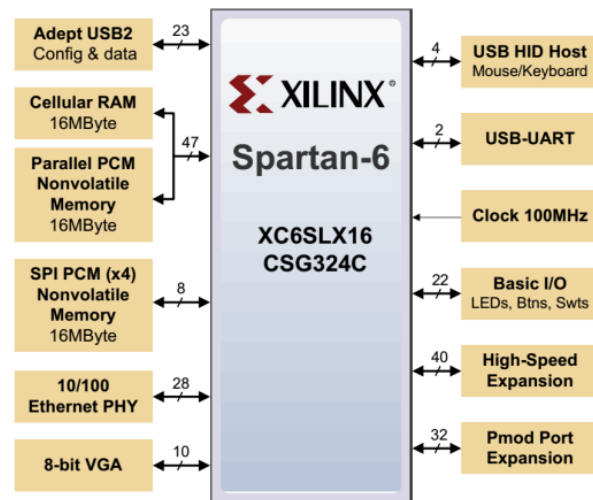
- 100 ~ 200MHz 振荡器（差分）



- RV32 ISA子集？

- 约40条指令，VerilogHDL

- 单周期/多周期实现（必做）
- 流水线实现（选作）





# 实验内容与计划（第5-15周）

- 运算与排序（1周）
- 寄存器堆与队列（1周）
- 单周期CPU（1周）
- 多周期CPU（2周）
- 中断与异常（1周）
- 总线与I/O接口（1周）
- 综合设计（3周）

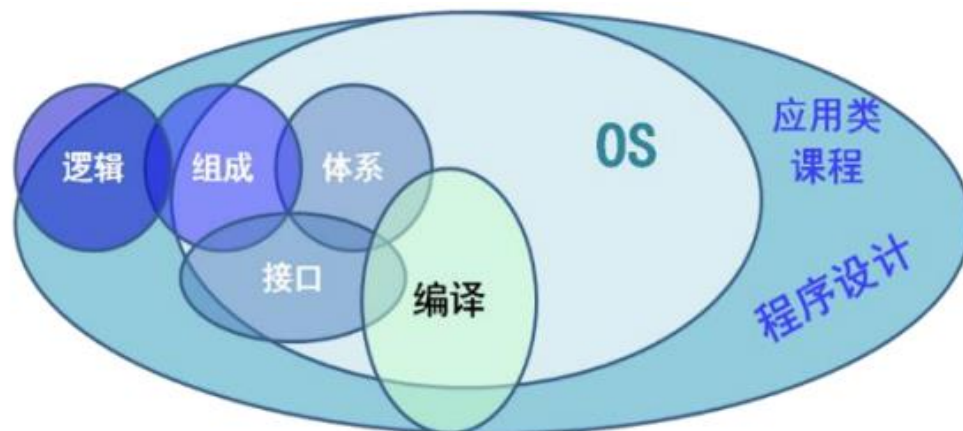




# 大纲



- 课程简介
- 教材选择
- 教学计划
- 成绩考核
- 学习方法
  - *Learn to do*
  - *Learn to be*
  - *Learn to be with others*
  - *Learn to learn*



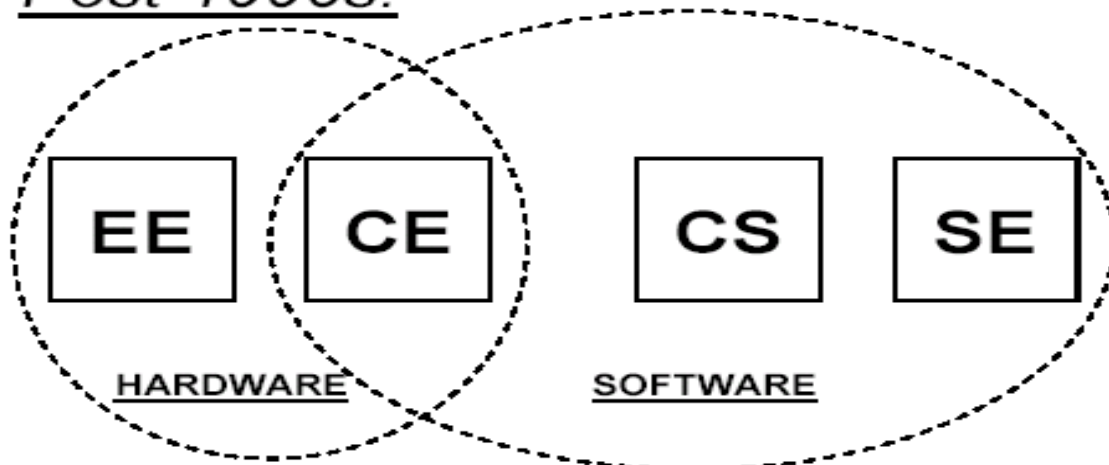
# 课程知识范畴：科学、技术、工程？



Pre-1990s:



Post-1990s:



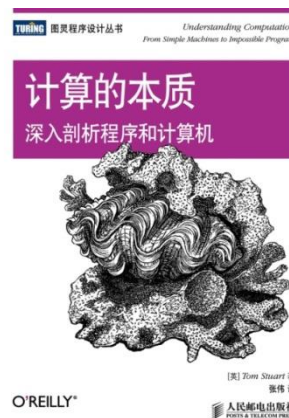
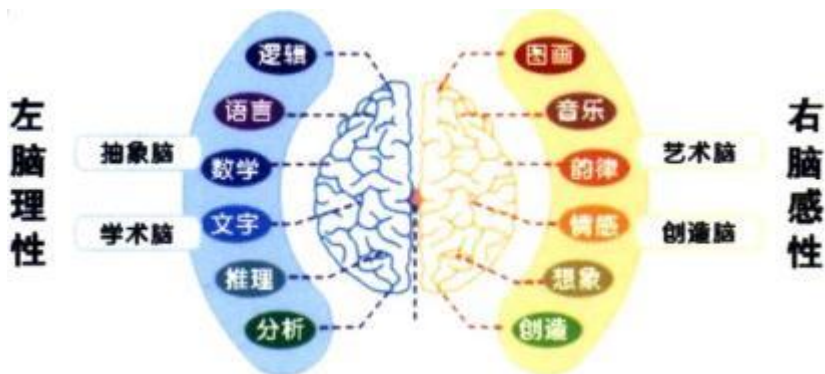
# 计算机系统，软件系统，硬件系统？



- 钱学森“系统论”
  - “系统”定义：由若干相互关联的要素以一定结构形式连接，具有明确的边界，共同完成某种功能的有机整体。
  - 系统学：研究系统结构与功能一般规律的科学。
- 分类：1990年提出
  - 简单系统：满足叠加原理（经典牛顿力学—线性）的单一层次的系统
  - 巨系统：规模大小，结构层次，是否出现自组织
    - 简单巨系统：智能机器人，至少可以分成系统层和子系统层两个层次
      - 熵：衡量子系统对系统总体性质的贡献，以系统状态之间的差值表示
    - 复杂巨系统：人体系统，社会系统
      - 巨 ≠ 大：出现新的性质，“整体大于局部”，“ $1 + 1 > 2$ ”
- 复杂性：不能用还原论方法（从结果回溯）处理的都是复杂性问题
  - Most real systems – as the term itself suggests – are complex
  - 异构性、多层次、过程多阶段和动态、相互关系和作用的非线性
- 开放性：与外部环境有物质、能量和信息的交换

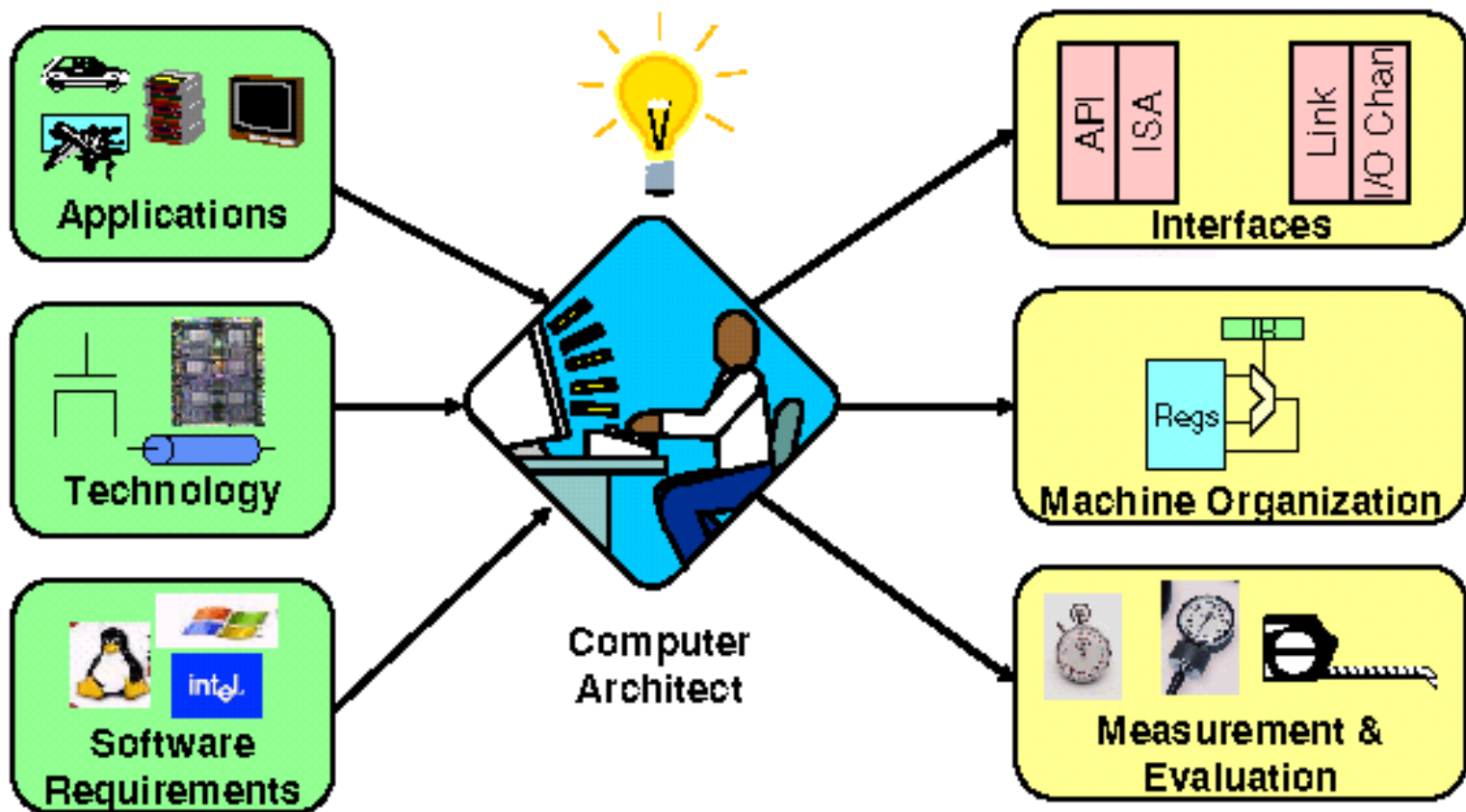
# 计算思维(Computational Thinking)概念

- 周以真 (Jeannette M. Wing) @CS.CMU, 2006
  - “计算思维是运用计算机科学的基础概念去求解问题、设计系统和理解人类行为的涵盖了计算机科学之广度的一系列思维活动”
  - “计算思维与数学思维和艺术思维不同”
- 计算：机器组织？MoC？程序的语义？编程模型？
  - Tom Stuart, 《Understanding Computation》
- 语言影响思维：Jon Bentley, 算法十大师之一
  - “我使用的Algol语言塑造了我的计算思维”
  - 《编程珠玑（续）》



More Programming Pearls  
**编程珠玑**  
续 英文版  
[美] Jon Bentley 著

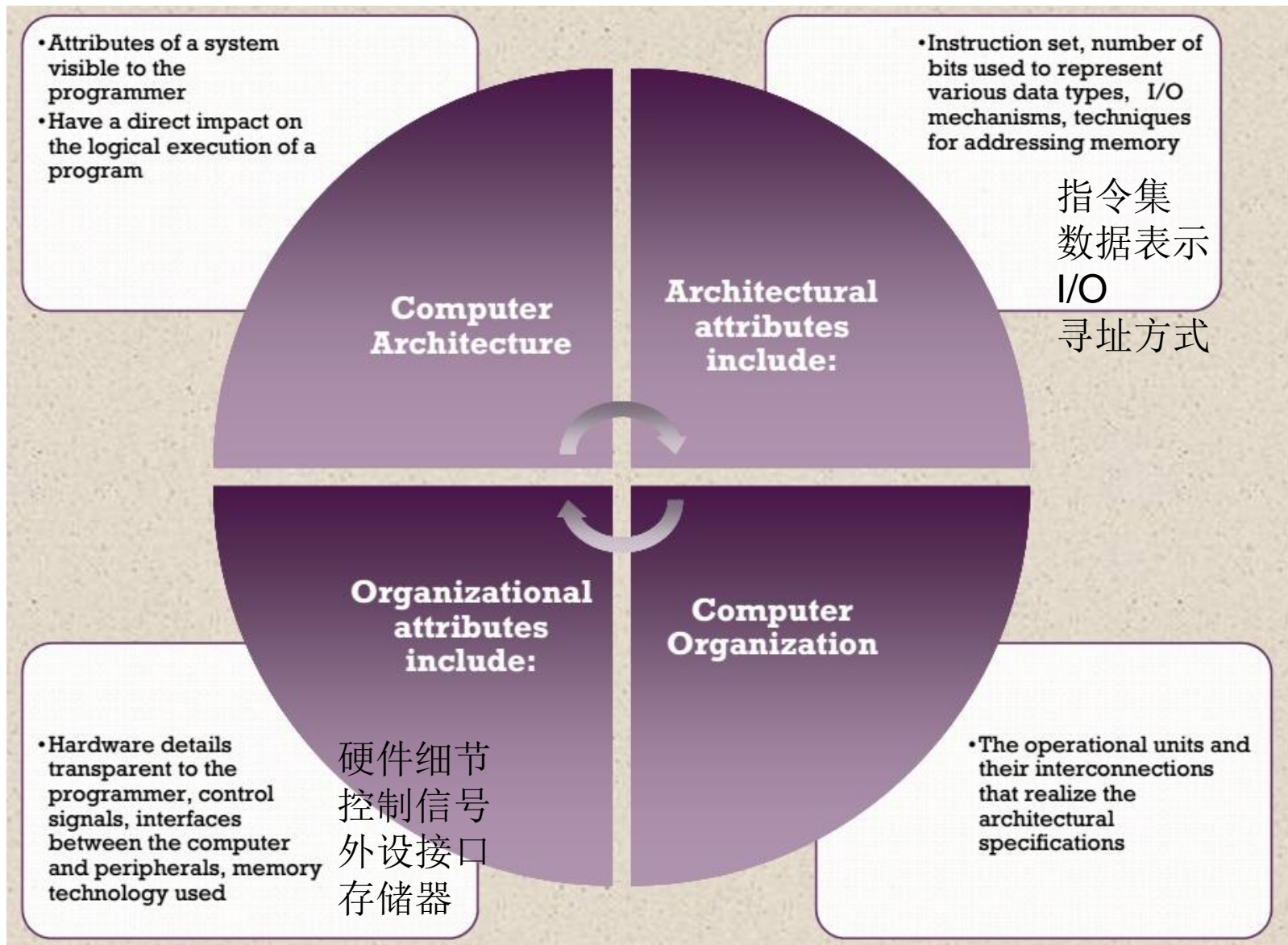
# 计算机系统设计：3-1-3模型



计算机系统：简单巨系统

计算机科学永恒的问题：如何设计出更好的计算机，如何利用好现有的计算机

# Computer Architecture & Organization

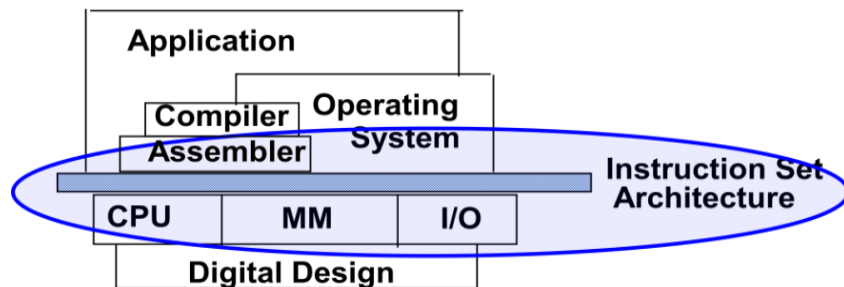


# 计算机组成 vs. 计算机体系结构



- 计算机体系结构(Computer architecture)

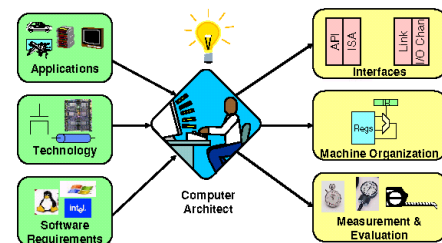
- “程序员可见的机器属性”
  - **G. M. Amdahl, 1964**
  - 程序员：特指机器语言程序员
  - 属性：包含结构特征和功能特征



- 对物理机器的抽象

- 计算机组成(org & impl, 也称 $\mu$ -architecture)

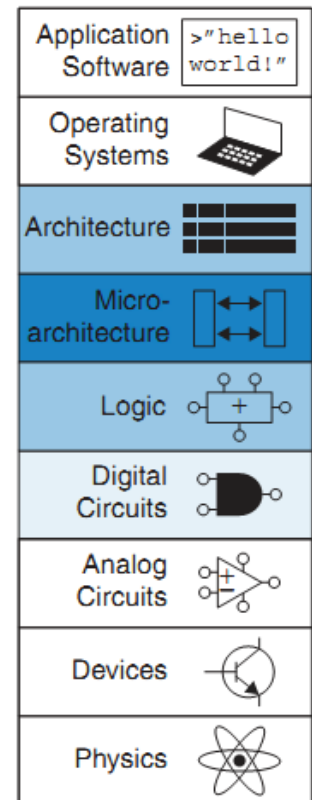
- 计算机体系结构的逻辑实现，包括
  - 实际机器的数据流和控制流的组成和逻辑设计等。
  - 物理机器内部各事件的排序方式与控制方式，各部件的功能，以及各部件的联系。



# COD思维（“系统思维”？）



- COD思维三要素——Ixx
  - 模块化：Component Based Design
    - The **block** diagram way of thinking
    - 复杂性：“separation of concerns”
  - 层次化：Abstraction/Refinement
    - **ABC范式**：结构，行为，约束
- COD语言：语法、语义、语用？
  - LTS、Automata、FSM，时态逻辑
  - 框图、原理图，时序图，CFG，DFG
- “计算机在干什么”？

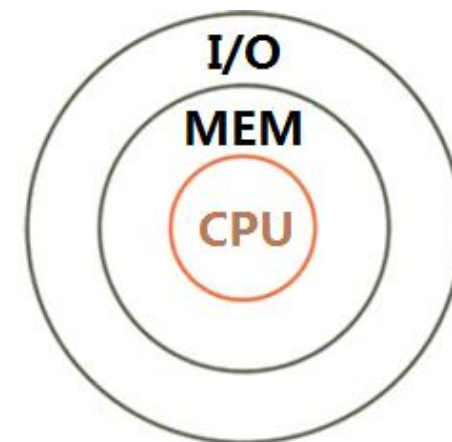
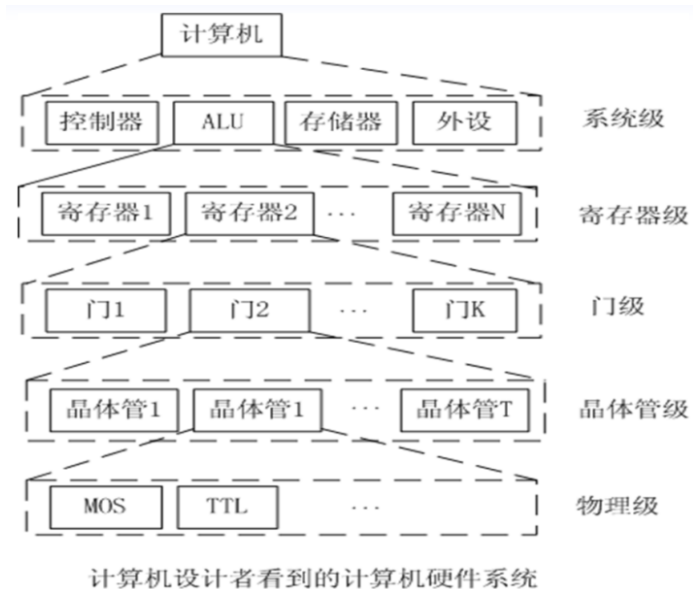




# 学习思路



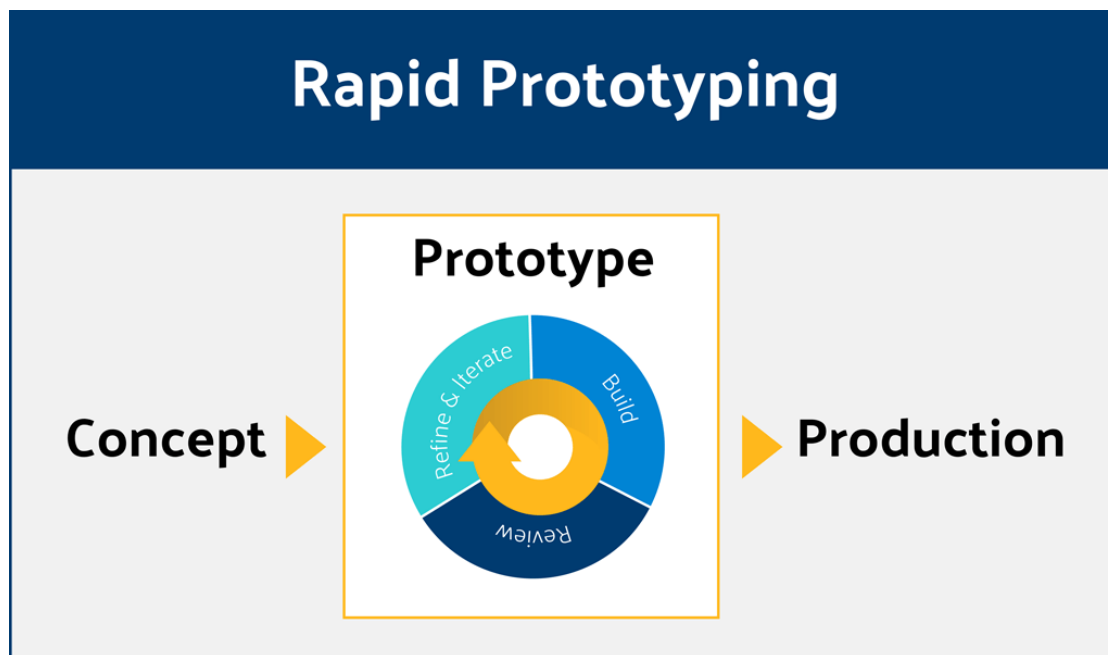
- Yale Patt@ece.utexas
  - Top-down: 适于“设计”——精化
    - 从高级语言开始入手
    - 前提是对底层有很好的理解
  - Bottom-up: 适于“学习”
    - 《计算机系统概论》
      - 从最底层的MOS晶体管开关器件开始，逻辑门、锁存器、逻辑结构；
      - 之后，有限状态机控制、顺序电路的实现、冯·诺伊曼结构、简单的计算机LC-3；
      - 最后，LC-3的机器和汇编语言、C高级语言、递归等，最后和基本数据结构。
- 唐（William Stallings）：自外而内
  - 总线、存储、I/O、ALU、ISA、CPU
- COD: 自内而外
  - ALU、ISA、CPU、存储、I/O



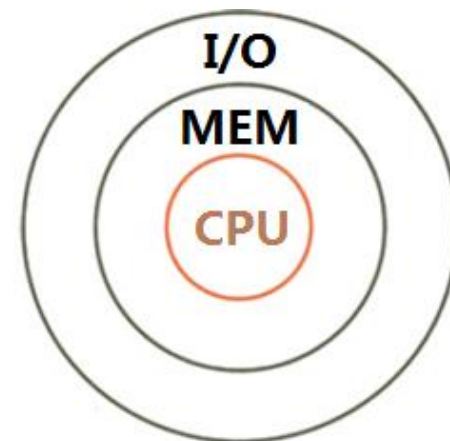
# 工程化方法

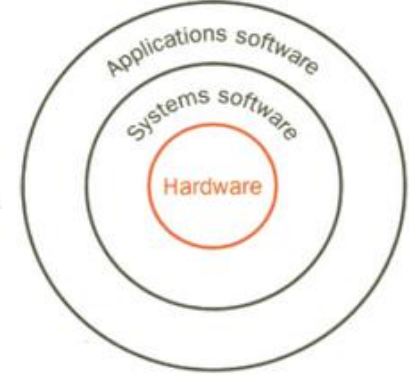


- 快速原型法
  - Build
  - Review
  - Refine & Iterate



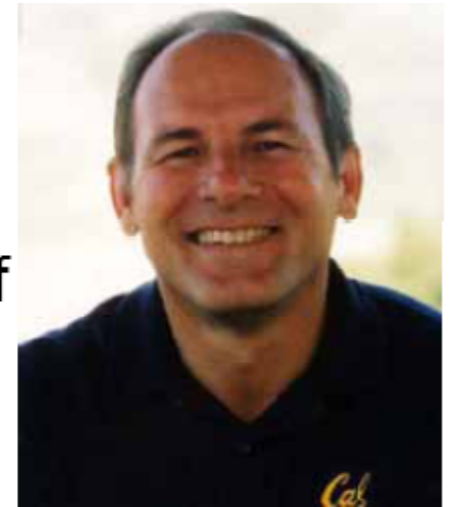
- 原型=最小的计算机？
  - ISA, CPU, INTR, SRAM;
  - DRAM, Cache, Disk, VM;
  - Bus, I/O;
  - ALU





# “害怕/不喜欢硬件” ☹️

- David A. Patterson@CS.UCB
  - “软件系统性能取决于软件设计者对系统硬件的理解程度。操作系统设计者、更多的软件工程师需要有较强的计算机组织与设计的背景知识”
  - 程序执行结果不仅取决于算法和程序设计，而且取决于其运行时环境和计算机体系结构！
- University of California at Berkeley in 1977
- Design and implementation of RISC I, first VLSI RISC Computer, foundation of the SPARC architecture, used by Sun
- leader of Redundant Arrays of Inexpensive Disks project (or RAID)
- Chair of the CS division at Berkeley



# COD: HW or SW?



Chapter or Appendix	Sections	Software focus	Hardware focus
1. Computer Abstractions and Technology	1.1 to 1.11		
	1.12 (History)		
	2.1 to 2.14		
2. Instructions: Language of the Computer	2.15 (Compilers & Java)		
	2.16 to 2.20		
	2.21 (History)		
E. RISC Instruction-Set Architectures	E.1 to E.17		
	3.1 to 3.5		
3. Arithmetic for Computers	3.6 to 3.8 (Subword Parallelism)		
	3.9 to 3.10 (Fallacies)		
	3.11 (History)		
B. The Basics of Logic Design	B.1 to B.13		
	4.1 (Overview)		
4. The Processor	4.2 (Logic Conventions)		
	4.3 to 4.4 (Simple Implementation)		
	4.5 (Pipelining Overview)		
	4.6 (Pipelined Datapath)		
	4.7 to 4.9 (Hazards, Exceptions)		
	4.10 to 4.12 (Parallel, Real Stuff)		
	4.13 (Verilog Pipeline Control)		
	4.14 to 4.15 (Fallacies)		
	4.16 (History)		
	D.1 to D.6		
D. Mapping Control to Hardware	5.1 to 5.10		
	5.11 (Redundant Arrays of Inexpensive Disks)		
5. Large and Fast: Exploiting Memory Hierarchy	5.12 (Verilog Cache Controller)		
	5.13 to 5.16		
	5.17 (History)		
6. Parallel Process from Client to Cloud	6.1 to 6.8		
	6.9 (Networks)		
	6.10 to 6.14		
A. Assemblers, Linkers, and the SPIM Simulator	6.15 (History)		
	A.1 to A.11		
C. Graphics Processor Units	C.1 to C.13		

Read carefully Review or read

Read if have time Read for culture

Reference



# 刻苦刻苦刻苦：MIT

- 我在这里的第一学期上了一门叫《计算机系统设计》得本科课程，其**辛苦**程度真实一言难尽。
- 十五个星期内交了十次作业，作了六次课程设计。有的设计还分几个部分，分开交设计报告。所以设计报告大概也交了有十次左右。
- 最恐怖的是有一次，十天内要交六份作业或设计报告，而且当时正值其他几门课正在期中考试。**抱怨是没有用的**，老师说：“我很抱歉。但这门课很重要，请大家不停的工作”。
- 学生从一般的逻辑时序电路开始设计（数电都已忘得差不多了），核心是自行设计“麻雀虽小五脏俱全”得ALU，单指令周期CPU（**single cycle CPU**），多指令CPU（**Multi-cycle CPU**），以直到最后实现流水线（**pipe line**）32位MIPS CPU和Cache。
- 一门课下来，所有与计算机CPU有关的知识全部**融会贯通**。硬件设计水平也有了很大提高（就是太累）。



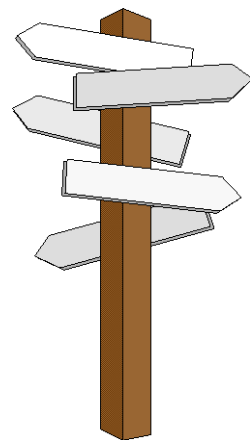
# 相关课程

- 美国UC Berkeley大学 “Machine Structure”2012年课程网站：  
<http://inst.eecs.berkeley.edu/~cs61c/sp12/>
- 美国UC Berkeley大学 “Components and Design Techniques for Digital System”2012年课程网站：  
<http://inst.eecs.berkeley.edu/~cs150/sp12/>
- 美国UC Berkeley大学 “Computer Architecture and Engineering”2012课程网站：  
<http://inst.eecs.berkeley.edu/~cs152/sp12/>
- 美国Stanford大学 “Computer Organization and Systems” 2012年课程网站：  
<https://courseware.stanford.edu/pg/courses/281000/cs107-spring-2012>
- 美国Stanford大学 “Digital Systems II”课程网站：  
<http://www.stanford.edu/class/ee108b/>
- 美国Stanford大学 “Digital Systems II”课程网站：  
<http://www.stanford.edu/class/cs110/>
- 美国Carnegie Mellon 大学 “Introduction to Computer Architecture”课程网站：  
<http://www.cs.cmu.edu/~213/>
- 美国Carnegie Mellon 大学 “Introduction to Computer Architecture”课程网站：  
<http://www.ece.cmu.edu/~ece447/>
- 美国Univ. Illinois at Urbana-Champaign “Computer Architecture II”课程网站：  
<http://www.cs.uiuc.edu/class/sp11/cs232/>
- 美国麻省理工学院(MIT)“Computation Structures”课程网站：  
<http://6004.csail.mit.edu>

# 调研报告一



- 选1，下周5交
  - 综述计算机技术的发展历程及热点问题
  - 最新Intel/ARM处理器的性能指标？
  - 计算机的开机过程？
  - C语言背后的计算机组成？
  - getchar()函数的实现过程？
  - PC系统活动与性能分析
    - 跑一个Benchmark，给出结果？
      - 每次执行同一个程序，结果都一样？
    - 主频与计算性能的关系？
- 参考小论文格式





Thank You