计算机组成与系统结构

第四章 指令系统

吕昕晨

lvxinchen@bupt.edu.cn

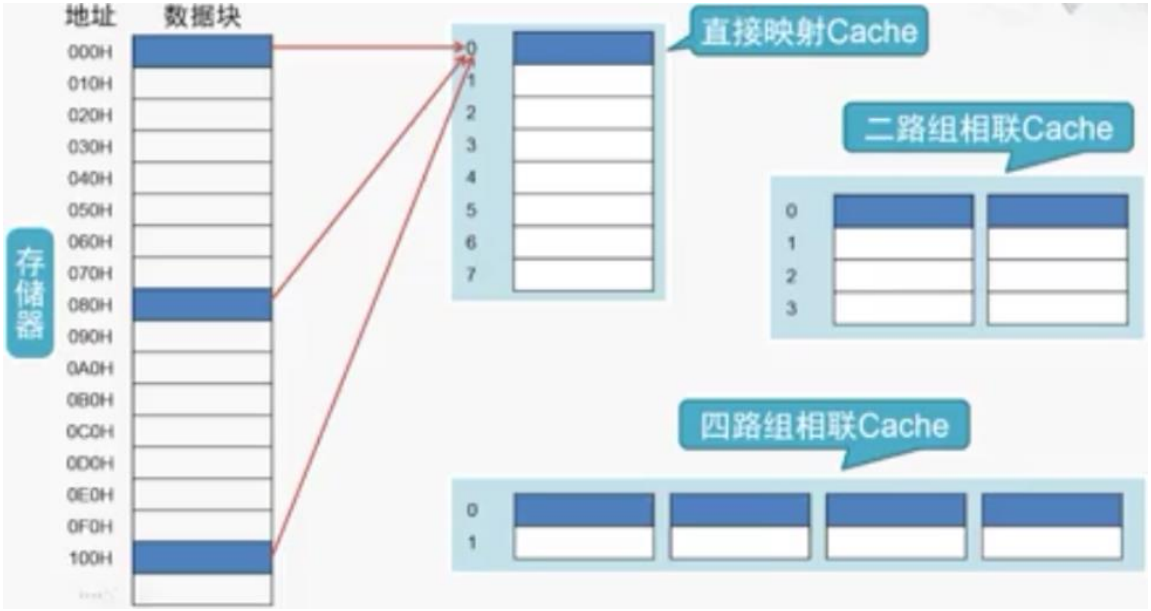
网络空间安全学院

复习

- Cache地址映射
- 交叉存储器

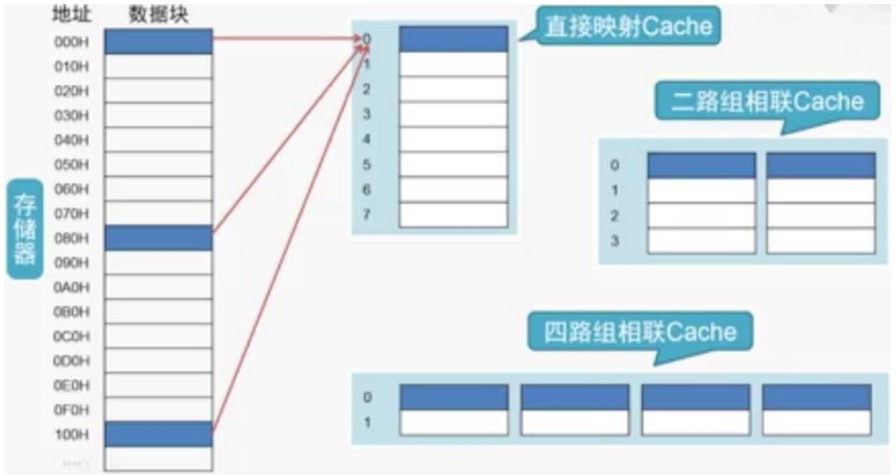
Cache地址映射

- Cache大小：N个数据块
  - 数据块 → X字
  - 直接映射：N*1
  - 全相联：1*N
  - 组相联：U（组数）*V（组内块数）



Cache地址映射划分

- 求解方法
  - 确定字号w
    - 块内偏移量?
    - 数据块→X字，$X=2^w$)
  - 确定行号r
    - 总共有多少行?
    - 直接映射N/组相联U，$N/U=2^r$)
  - 确定标记
    - 多少个数据块被映射到同一行?
    - 计算方法：内存总数据块地址位s-行号r
    - 内存大小→Y数据块，$2^s=Y$



地址映射方式例题

- 主存容量1MB，字长8位，块大小16B，Cache容量64KB
- 1）采用直接映射，给出[F0010H]对应标记为 [填空1] 、行号为[填空2]、字号为 [填空3] 。
  - 确定字号w
    - 数据块→16字，X=2^w　　∴ 字号w＝4
  - 确定行号r
    - 行数N=64KB/16B=2^{12}　∴ 行号r＝12
  - 确定标记
    - 内存大小1MB/16B=216 ∴ s＝16
    - 标记位数：s-r=4
  - F0010H= 1111 0000 0000 0001 0000

标记　　行号　　字号

地址映射方式例题

- 主存容量1MB，字长8位，块大小16B，Cache容量64KB
- 2）采用二路组相联映射，给出[F0010H]对应标记为 [填空1] 、行号为[填空2]、字号为 [填空3] 。
  - 确定字号w
    - 数据块→16字，X=2^w　　∴ 字号w＝4
  - 确定行号r
    - 行数N=64KB/16B/2=2^{11}　∴ 行号r＝11
  - 确定标记
    - 内存大小1MB/16B=216 ∴ s＝16
    - 标记位数：s-r=5
  - F0010H= 1111 0000 0000 0001 0000

标记　　行号　　字号

地址映射方式例题

- 主存容量1MB，字长8位，块大小16B，Cache容量64KB
- 3）采用全相联映射，给出[F0010H]对应标记为 [填空1] 、字号为 [填空2] 。
  - 确定字号w
    - 数据块→16字，$X=2^w$　　∴ 字号w＝4
  - 确定标记
    - 内存大小1MB/16B=216 ∴ s＝16
  - F0010H= 1111 0000 0000 0001 0000
    标记　　字号

复习

- Cache地址映射
- 交叉存储器

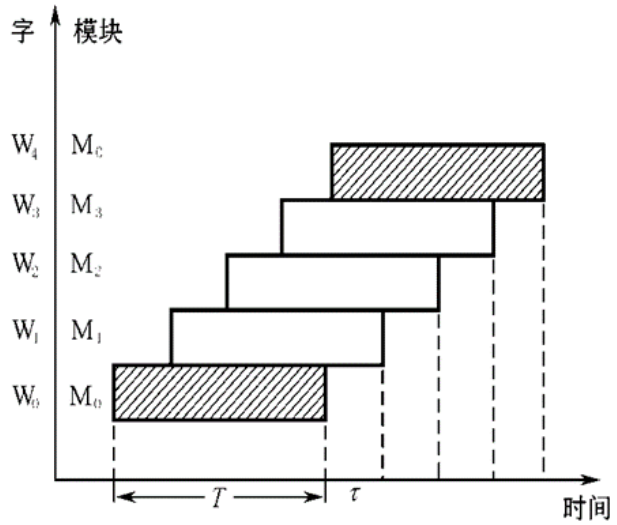
多模块交叉存储器效率

- 通常在一个存储器周期内
  - n个存储体必须分时启动
  - 各个存储体的启动间隔为 $t = T / n$
  - n为交叉存取度（顺序方式n=1，交叉方式为扩展芯片数）

- 整个存储器的存取速度有望提高n倍

$$t_{顺序} = xT$$

$$t_{交叉} = T + (x-1)t = T(\frac{x+n-1}{n})$$

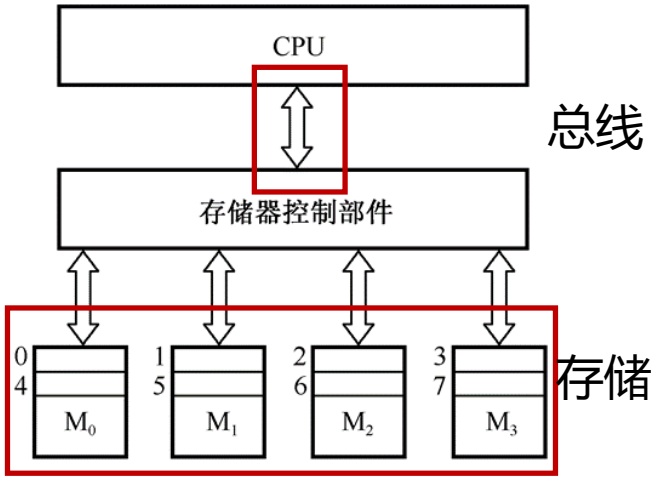


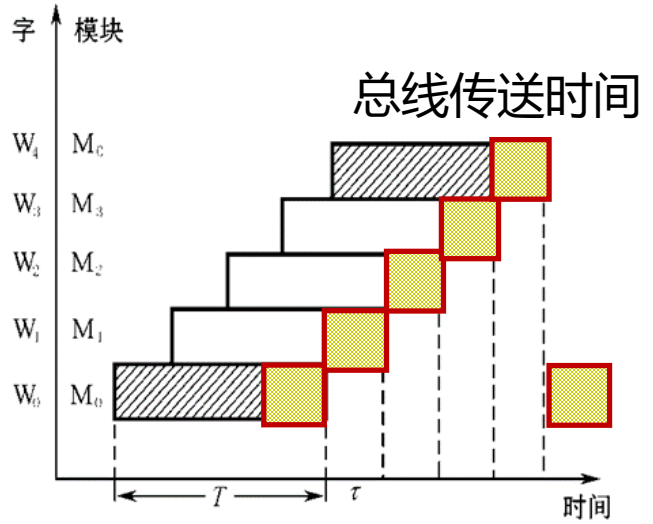
存储周期与总线周期

- 存储周期
  - 存储器读出一个字所需时间
  - 记作：T
- 总线传送周期
  - 总线传送一个字所需时间
  - 记作：τ（最高带宽=字长/总线周期）
- 关系
  - 各存储体共用总线
  - 存储体的启动间隔不低于总线周期
  - 一般设计上使得

$$\tau = T / n$$

  - 调整模块数n，流水平衡状态





例5 设存储器容量为32字，字长64位，模块数m=4，分别用顺序方式和交叉方式进行组织。存储周期T=200ns，数据总线宽度为64位，总线传送周期=50ns。
若连续读出4个字，问顺序存储和交叉存储的带宽各是多少?

解:

顺序存储器和交叉存储器连续读出m=4个字的信息总量都是:

$$q=64\text{（字长）}\times 4=256\ \text{bit}$$

顺序存储器和交叉存储器连续读出4个字所需的时间分别是:

$$t_{顺序}=mT=4\times 200ns=800ns=8\times 10^{-7}s$$

$$t_{交叉}=T+(m-1)T/n=200+3*200/4=350ns=35\times 10^{-7}s$$

顺序存储器和交叉存储器的带宽分别是:

$$W_{顺序}=q/t_{顺序}=256b\div(8\times 10^{-7})s=320Mb/s$$

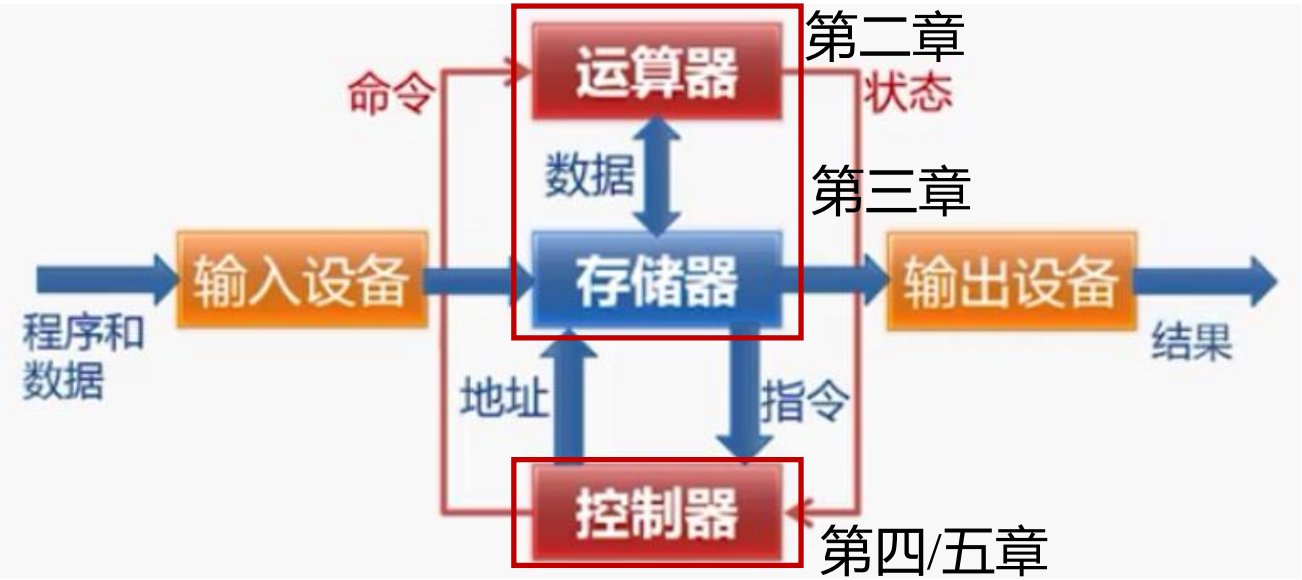
$$W_{交叉}=q/t_{交叉}=256b\div(35\times 10^{-7})s=730Mb/s$$

问题：若连续读出8个字，总线周期为25ns、100ns该如何计算?

复杂（25ns*3+400ns）；max(200/4,100)=100ns

本周教学安排

- 直播内容
  - 指令系统基本概念
  - 指令结构
    - 基本原理
    - 地址映射



第四章 指令系统

- 指令系统概述
- 简单的计算机指令系统
- 指令格式
- 典型指令格式

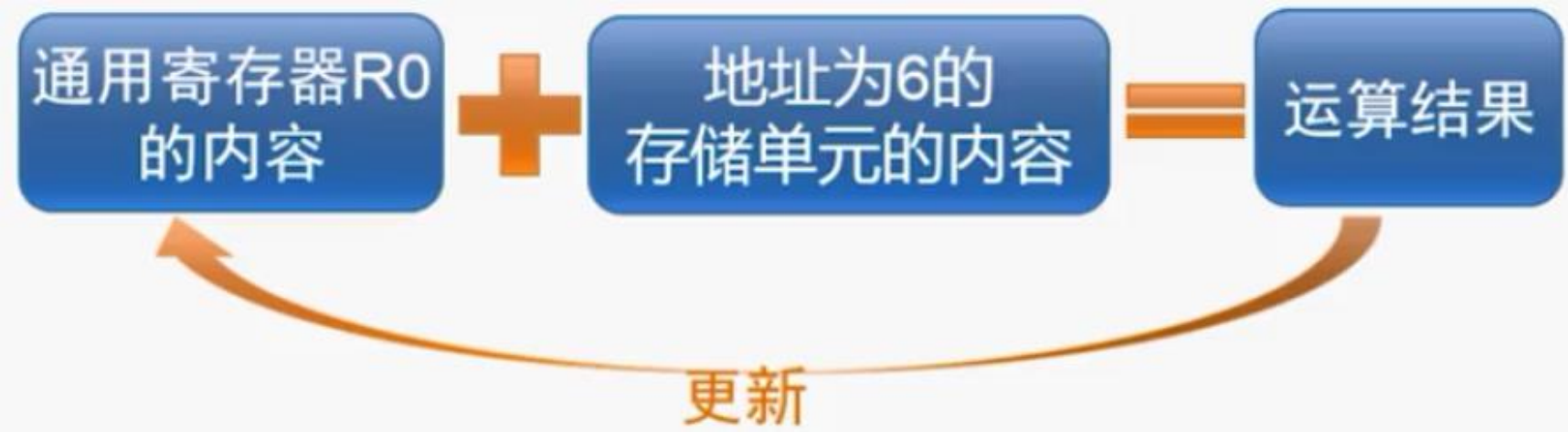
机器指令/微指令

- 指令：计算机执行某种操作的命令
- 从计算机组成的层次结构来说，计算机的指令有微指令、机器指令和宏指令之分
  - 微指令：微程序级的命令，属于硬件（第五章）
  - 宏指令：由若干条机器指令组成的软件指令，属于软件
  - 机器指令：介于微指令与宏指令之间，通常简称为指令，每一条指令可完成一个独立的算术运算或逻辑运算操作（第四章）
- 指令系统：一台计算机中所有机器指令的集合
  - 指令系统是表征一台计算机性能的重要因素
  - 它的格式与功能不仅直接影响到机器的硬件结构，而且也直接影响到系统软件，影响到机器的适用范围

指令示例

- 指令示例
  - 加法指令：ADD R0，[6]

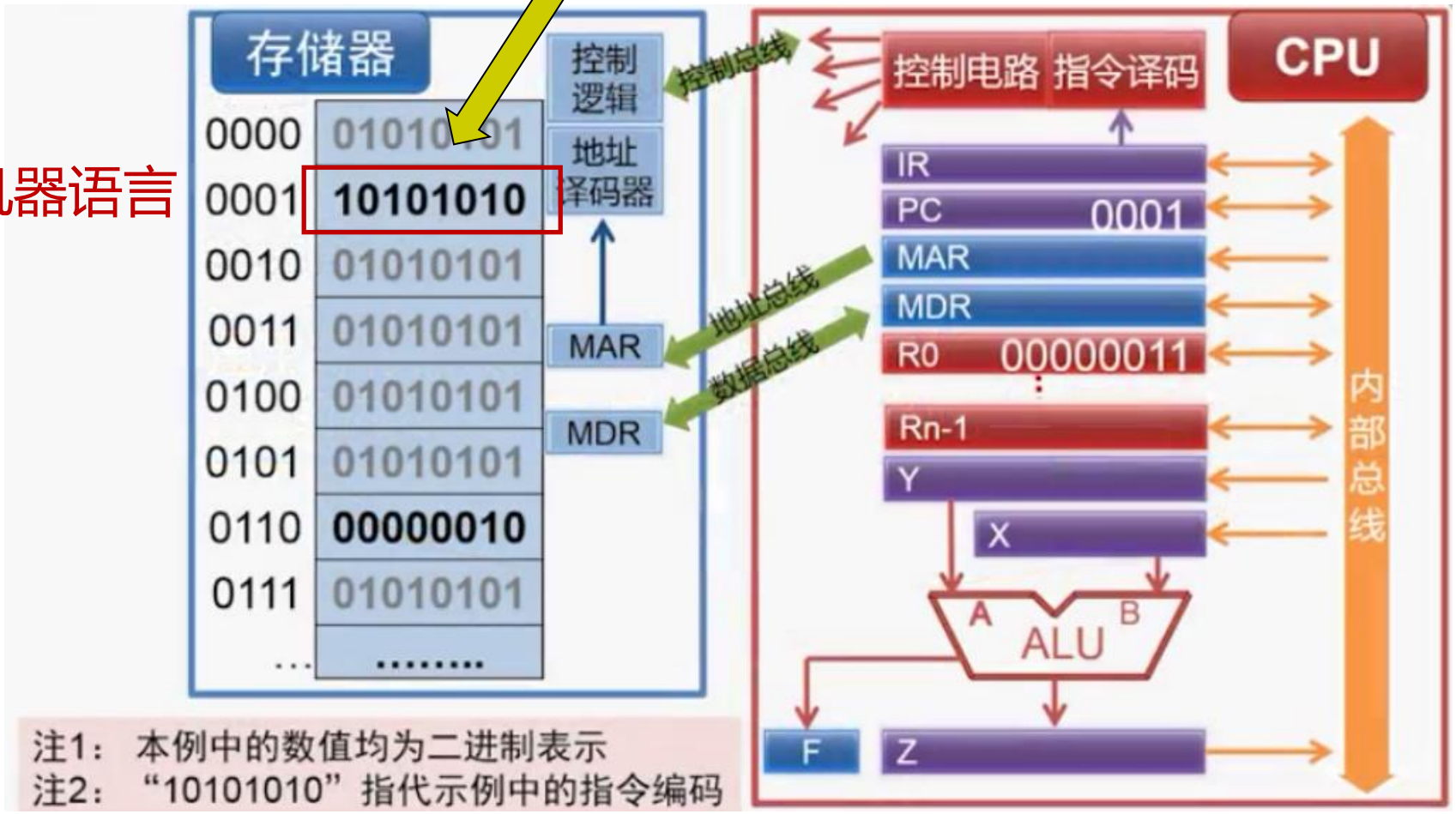
- 指令功能



指令示例

汇编语言：ADD R0，[6]

机器语言



指令系统性能的要求

- **完备性**
  - 汇编语言编写各种程序时，指令系统直接提供的指令足够使用，而不必用软件来实现
  - 完备性要求指令系统丰富、功能齐全、使用方便
  - 一台计算机中最基本、必不可少的指令是不多的
  - 例如，乘除运算指令、浮点运算指令可直接用硬件来实现，也可用基本指令编写的程序来实现。采用硬件指令的目的是提高程序执行速度，便于用户编写程序。
- **有效性**
  - 有效性是指利用该指令系统所编写的程序能够高效率地运行
  - 高效率主要表现在程序占据存储空间小、执行速度快
  - 一般来说，一个功能更强、更完善的指令系统，必定有更好的有效性

指令系统性能的要求

- **规整性：**对称性、匀齐性、指令格式和数据格式的一致性
  - 对称性：在指令系统中所有的寄存器和存储器单元都可同等对待，所有的指令都可使用各种寻址方式
  - 匀齐性：一种操作性质的指令可以支持各种数据类型，如算术运算指令可支持字节、字、双字整数的运算，十进制数运算和单、双精度浮点数运算等
  - 指令格式和数据格式的一致性：指令长度和数据长度有一定的关系，以方便处理和存取。例如指令长度和数据长度通常是字节长度的整数倍
- **兼容性**
  - 各机种之间具有相同的基本结构和共同的基本指令集，因而指令系统是兼容的，即各机种上基本软件可以通用
  - 但由于不同机种推出的时间不同，即低档机上运行的软件可以在高档机上运行

指令系统分类

- **复杂指令系统计算机**：简称CISC
  - 庞大的指令系统，指令变长
  - 译码较复杂，指令复杂度差别大
  - 计算机的研制周期变长，难以保证正确性，不易调试维护，而且由于采用了大量使用频率很低的复杂指令而造成硬件资源浪费
- **精简指令系统计算机**：简称RISC
  - 2-8定律：20%的指令占程序80%的比例
  - 便于VLSI技术实现的精简指令系统计算机
  - 指令定长，复杂度相近，便于流水线技术实现

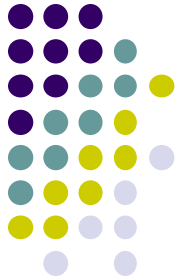
指令体系结构

- **指令系统体系结构ISA**（Instruction Set Architecture）
- 机器语言程序员看到的计算机的属性，是与程序设计有关的计算机架构
  - 寄存器组织
  - 存储器的组织和寻址方式
  - I/O系统结构
  - 数据类型及其表示
  - 指令系统
  - 中断机制
  - 机器工作状态的定义及切换
  - 保护机制

低级语言与高级语言对比关系

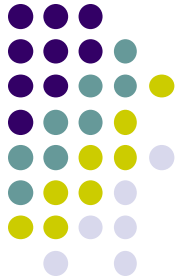
	比较内容	高级语言	低级语言
1	对程序员的训练要求 (1)通用算法 (2)语言规则 (3)硬件知识	有 较少 不要	有 较多 要
2	对机器独立的程度	独立	不独立
3	编制程序的难易程度	易	难
4	编制程序所需时间	短	较长
5	程序执行时间	较长	短
6	编译过程中对计算机资源的要求	多	少

第四章 指令系统



- 指令系统概述
- 简单的计算机指令系统
- 指令格式
- 典型指令格式

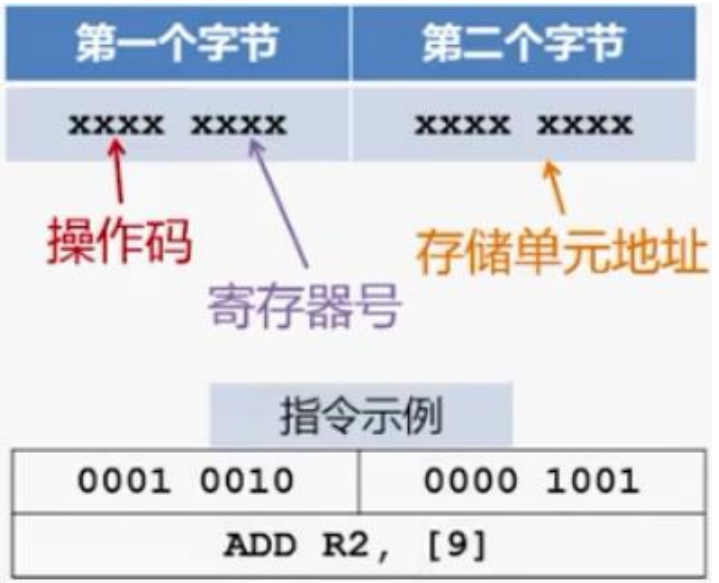
简单的计算机指令系统



- 运算类指令
  - ADD R, M
    - 功能：将寄存器R内容与存储器M中内容相加后存入R
- 访存类指令
  - LOAD R, M
    - 功能：将存储器M中内容装入寄存器R
  - STORE M, R
    - 功能：将寄存器R的内容存入存储器M
- 转移类指令
  - JMP L
    - 功能：无条件挑战至标号L处

指令的格式

- 指令长度
  - 等长指令，均为2字节
- 第一个字节高4位为操作码
  - LOAD：0000；ADD：0001
  - STORE：0010；JMP：0011
  - 可扩展至16条指令
- 第一个字节低4位为寄存器号
  - R0~R15，支持16个寄存器
- 第二个字节是存储单元地址
  - 存储器地址范围最大为256个字节



指令示例

- 操作码表
  - LOAD：0000；ADD：0001
  - STORE：0010；JMP：0011

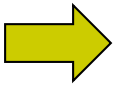
- 机器指令翻译
  - 机器语言：0000 0000 0000 0000
  - 汇编语言：LOAD R0, [0]

  - 机器语言： 0010 0011 0000 1111
  - 汇编语言： STORE R3, [15]

  - 机器语言： 0101 0000 1111 0110
  - 汇编语言：指令错误，操作码未定义

程序举例

- 任务说明
  - 将M1的内容与M2的内容相加后存入M3
  - 完成运算后，程序跳转到L处继续执行
  - M1-M3为存储单元地址
- 程序实现
  - LOAD R, M1
  - ADD R, M2
  - STORE R, M3

  ADD M3, M1, M2
  - JMP L
- 思考
  - 如果有汇编语言支持ADD M3, M1, M2?
  - 目前指令是否完备，还差哪些指令?

程序举例

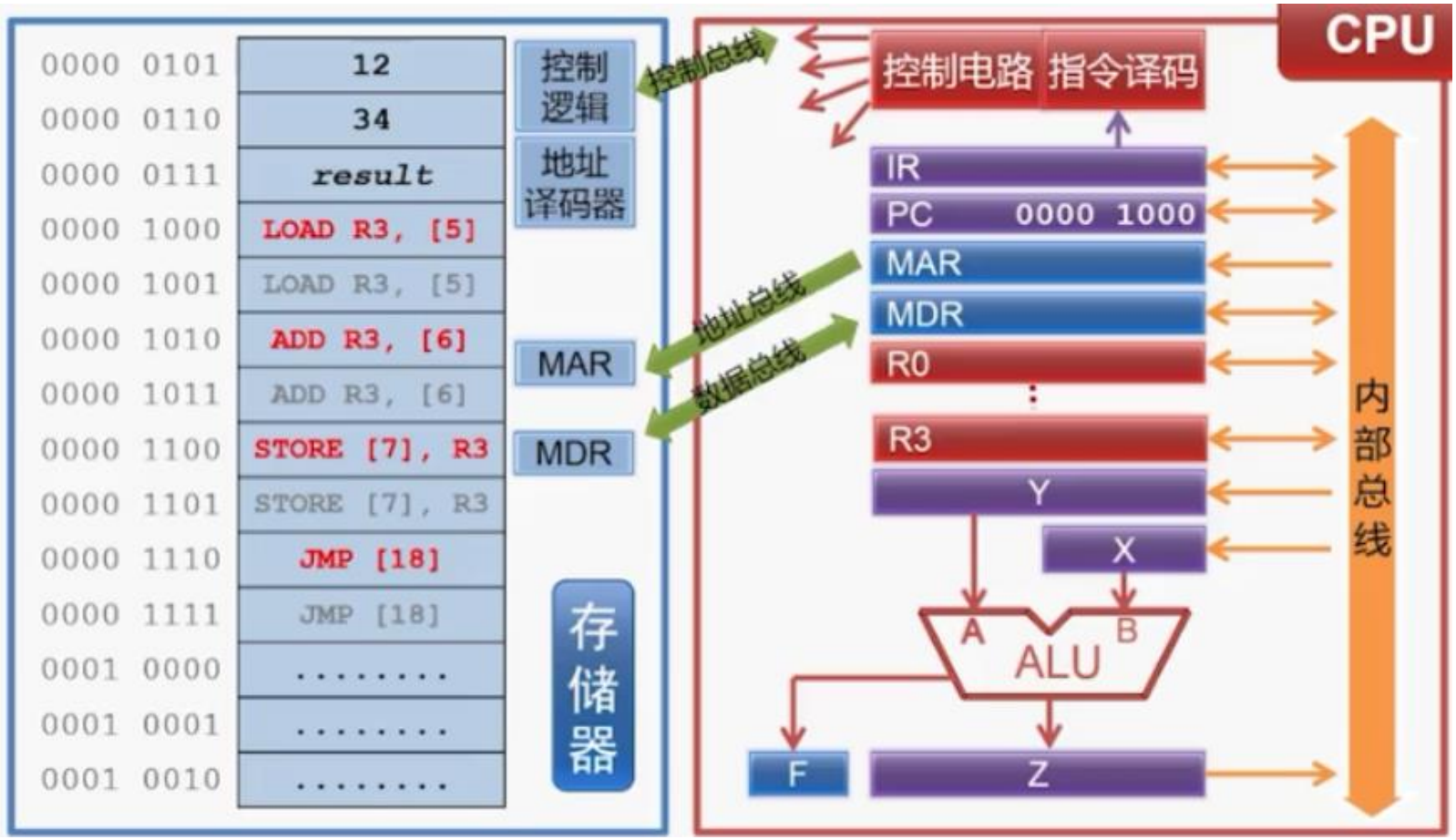
- 任务说明
  - 将M1的内容与M2的内容相加后存入M3，无条件跳转到L处
  - M1=5，M2=6，M3=8，L=18
- 汇编—机器语言对应

汇编语言程序	机器语言程序	程序的功能
LOAD R3, [5]	00000011 00000101	将存储单元[5]的内容送入寄存器R3
ADD R3, [6]	00010011 00000110	将寄存器R3的内容加上存储单元[6]的内容，再送回R3
STORE [7], R3	00100011 00000111	将寄存器R3的内容送入存储单元[7]中
JMP [18]	00110000 00010010	转向存储单元[18]，取出指令继续执行

存储器中的机器语言程序

存储器地址	存储器内容	说明	
0000 0101	00001100	地址 [5]，存放了数据12	数据段
0000 0110	00100010	地址 [6]，存放了数据34	
0000 0111	00000000	地址 [7]，准备存放运算结果	
0000 1000	00001011	“LOAD R3, [5]”的第一个字节	程序段
0000 1001	00000101	“LOAD R3, [5]”的第二个字节	
0000 1010	00011011	“ADD R3, [6]”的第一个字节	
0000 1011	00000110	“ADD R3, [6]”的第二个字节	
0000 1100	00101011	“STORE [7], R3”的第一个字节	
0000 1101	00000111	“STORE [7], R3”的第二个字节	
0000 1110	00110000	“JMP [18]”的第一个字节	
0000 1111	00010001	“JMP [18]”的第二个字节	
0001 0000	.....	第五条指令的第一个字节	
0001 0001	.....	第五条指令的第二个字节	
0001 0010	.....	第六条指令的第一个字节（地址 [18]）	

模型机中程序运行


0000 0101
12
0000 0110
34
0000 0111
result
0000 1000
LOAD R3, [5]
0000 1001
LOAD R3, [5]
0000 1010
ADD R3, [6]
0000 1011
ADD R3, [6]
0000 1100
STORE [7], R3
0000 1101
STORE [7], R3
0000 1110
JMP [18]
0000 1111
JMP [18]
0001 0000
0001 0001
0001 0010
控制逻辑
地址译码器
MAR
MDR
存储器
控制总线
地址总线
数据总线
CPU
控制电路
指令译码
IR
PC
0000 1000
MAR
MDR
R0
R3
Y
X
A
B
ALU
F
Z
内部总线

指令系统设计

- 指令系统要求
  - 完备性
  - 有效性
  - 规整性
  - 兼容性
- 问题1：规定指令格式，支持更多指令功能
  - 操作码、操作数、寄存器号、存储单元地址……
- 问题2：寻址方式
  - 地址范围受地址码限制（8位地址码　256个字节）
  - 增加地址码，影响指令系统效率
  - 设计寻址方式，支持更大范围寻址

第四章 指令系统

- 指令系统概述
- 简单的计算机指令系统
- 指令格式
  - 操作码
  - 地址码
  - 指令长度/助记符
- 典型指令格式

指令格式

- 指令格式
  - 指令用二进制代码表示的结构形式
- 组成
  - 操作码字段
    - 表征指令操作特性与功能
    - 运算类、访存类、转移类
  - 地址码字段
    - 参与操作的操作数地址
    - 寄存器、存储器、操作数……

操作码字段	地址码字段

操作码（1）

- 操作码：OP（Operation code）
  - 表示该指令应进行什么性质的操作
  - 如进行加法、减法、乘法、除法、取数、存数等
  - 不同的指令用操作码字段的不同编码来表示，每一种编码代表一种指令
- 组成操作码字段的位数一般取决于计算机指令系统的规模
- 较大的指令系统就需要更多的位数
- n位（操作码）→2^n
  - 32条指令：5位操作码

操作码（2）

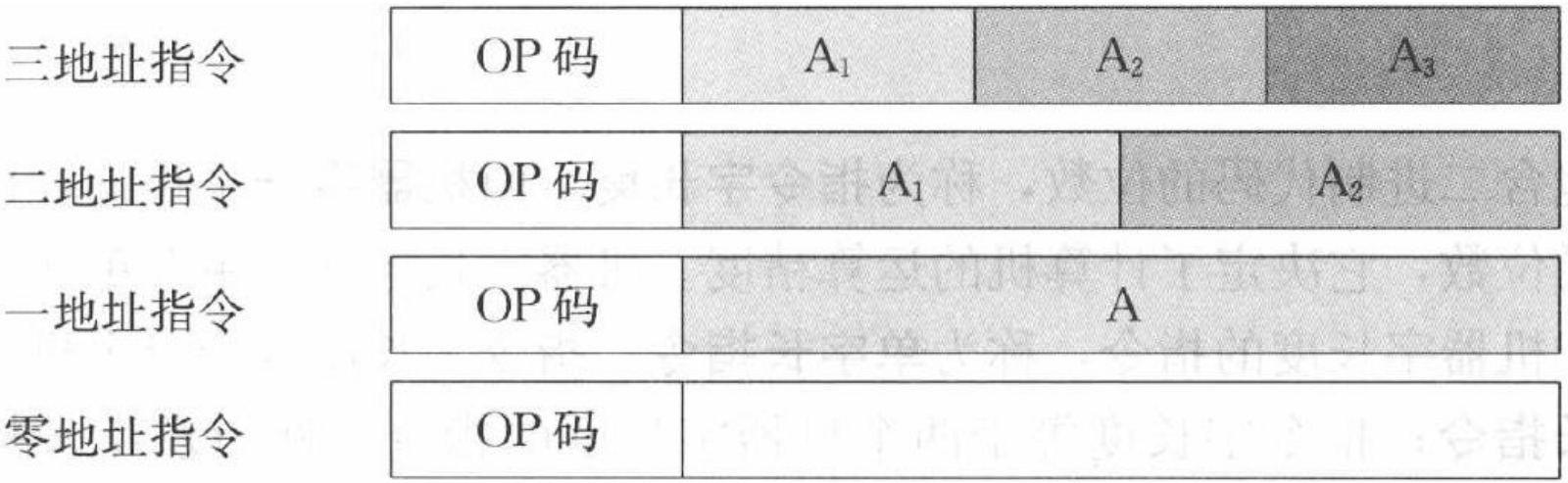
- 等长操作码（通常）
  - 指令规整，译码简单
- 例如IBM 370机
  - 该机字长32位，共有183条指令
  - 指令的长度可以分为16位、32位和48位等几种，所有指令的操作码都是8位固定长度
  - 固定长度编码的主要缺点是：信息的冗余极大，使程序的总长度增加
- 不等长操作码/地址码
  - 充分利用指令长度
  - 适合于单片机等，指令字较短系统

第四章 指令系统

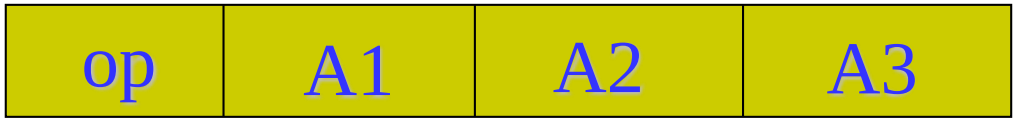
- 指令系统概述
- 简单的计算机指令系统
- 指令格式
  - 操作码
  - 地址码
  - 指令长度/助记符
- 典型指令格式

指令分类

- 根据一条指令中有几个操作数地址，可将该指令称为几操作数指令或几地址指令
  - 三地址指令
  - 二地址指令
  - 单地址指令
  - 零地址指令



三地址指令



- 指令组成
  - 操作码op
  - 第一操作数A1/第二操作数A2 、结果A3
- 指令功能：
  - (A1) op (A2)→A3
  - (PC) +1→PC
- 特点
  - 指令长度仍比较长，通常不用于微型机
  - A1/A2/A3通常为寄存器，加快指令执行速度

ADD R0,R1,R2
R0=R1+R2

二地址指令（1）

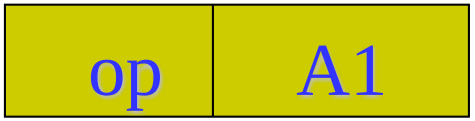
- 指令组成
  - 操作码op
  - 第一操作数A1、第二操作数A2
- 指令功能
  - (A1) op (A2)→A1
  - (PC)+1→PC
- 特点
  - 二地址指令在计算机中得到了广泛的应用
  - 注意：指令执行之后，A1中原存的内容已经被新的运算结果替换了

ADD R0, [6]

二地址指令（2）

- 根据操作数的物理位置分为：
  - 存储器-存储器（SS）
    - 参与操作的数均存放在内存里，需要多次访问内存
    - 速度慢
  - 寄存器-存储器类型（RS）
    - 参与操作的数存放在内存与寄存器内，一般需要一次访存
    - 速度较慢
  - 寄存器-寄存器类型（RR）
    - 参与操作的数均存放在寄存器中，不需要访问内存
    - 速度快

一地址指令



- 指令组成
  - 操作码op
  - 操作数A
- 指令功能
  - (AC) op (A) →AC / C op (A) →A ……
  - (PC)+1→PC
  - AC表示累加寄存器AC中的数、C代表常数
- 特点
  - 单操作数运算指令，+1（INC）、-1、取反
  - 指令中给出一个源操作数的地址，另一操作数隐含

零地址指令

- 指令组成
  - 操作码op
- 特点
  - “停机”、“空操作”、“清除”等控制类指令

第四章 指令系统

- 指令系统概述
- 简单的计算机指令系统
- 指令格式
  - 操作码
  - 地址码
  - 指令长度/助记符
- 典型指令格式

指令字长度

- 基本概念
  - 指令字长度（一个指令字包含二进制代码的位数）
  - 机器字长：计算机能直接处理的二进制数据的位数
- 指令字长度
  - 单字长指令
  - 半字长指令
  - 双字长指令
- 指令字长可变
- 例如，IBM370系列（32位），指令格式
  - 16位（半字长）
  - 32位（单字长）
  - 48位（一个半字长）

指令字长度

- 多字长指令
  - 优点：提供足够的地址位来解决访问内存任何单元的寻址问题
  - 缺点：必须两次或多次访问内存以取出一整条指令，降低了CPU的运算速度，又占用了更多的存储空间
- 等长指令系统
  - RISC：各种指令字长度是相等的，指令字结构简单，且指令字长度是不变的
- 变长指令系统
  - CISC：各种指令字长度随指令功能而异，结构灵活，能充分利用指令长度，但指令的控制较复杂

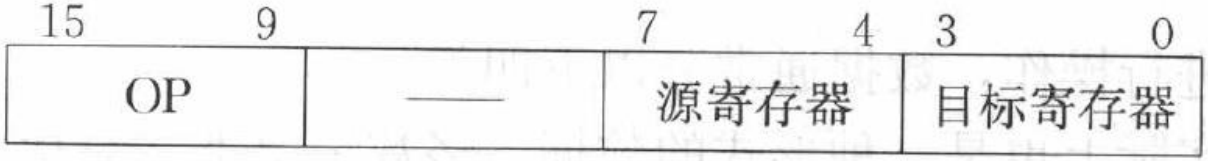
指令助记符

- 硬件只能识别1和0，机器指令二进制
- 人工书写二进制程序十分麻烦
- 指令助记符：指令通常用3个或4个英文缩写字母来表示
  - 用3～4个英文字母来表示操作码，一般为英文缩写
  - 不同的计算机系统，规定不一样
  - 必须用汇编器翻译成二进制代码
  - 7条指令指令系统，操作码3位

典型指令	指令助记符	二进制操作码
加法	ADD	001
减法	SUB	010
传送	MOV	011
跳转	JMP	100
转子	JSR	101
存数	STO	110
取数	LAD	111

指令格式例题（1）

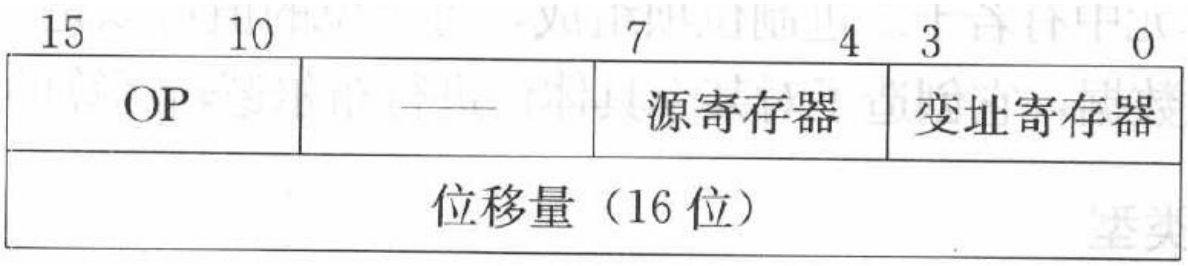
- 例1 16位字长，指令格式如下，分析指令格式特点



- 单字长（16位）二地址指令
- 操作码字段（7位）：可支持128条指令
- RR型指令
- 寄存器数目：16个

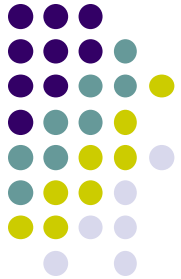
指令格式例题（2）

- 例2 16位字长，指令格式如下，分析指令格式特点



- 双字长二地址指令
- 操作码字段（6位）：可支持64条指令
- RS型指令
- 寄存器数目：16个/存储器（变址寄存器+位移量）

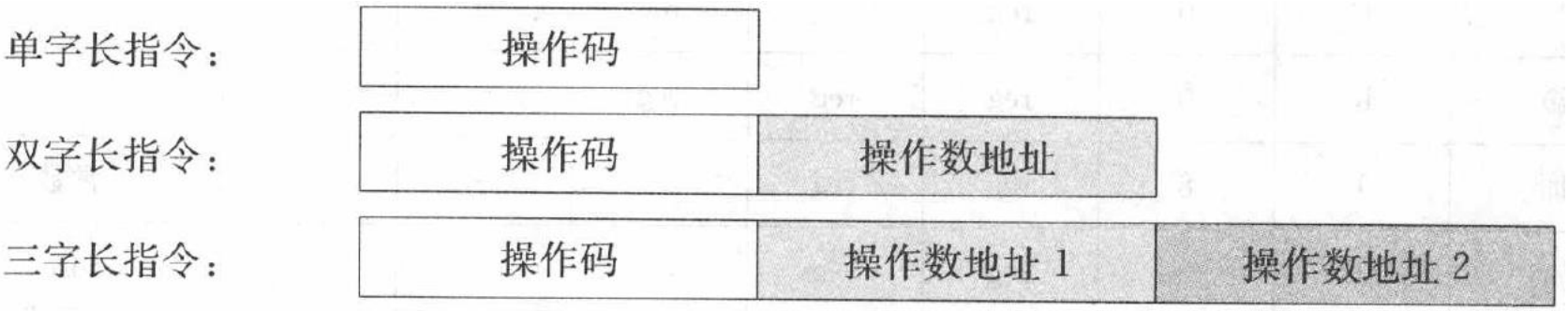
第四章 指令系统



- 指令系统概述
- 简单的计算机指令系统
- 指令格式
- 典型指令格式

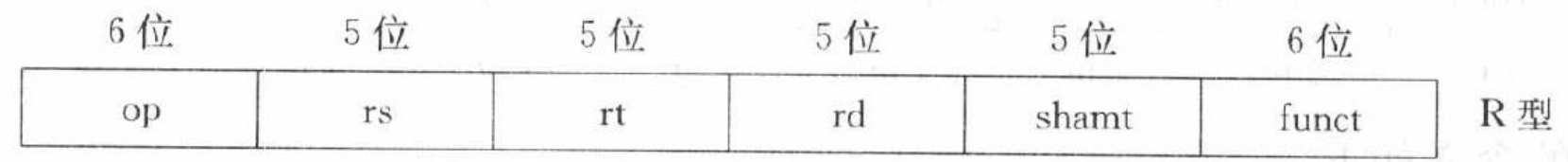
八位微型计算机指令格式

- 特点
  - 早期微机系统：字长8位，指令结构可变长
  - 包括单字长指令、双字长指令和三字长指令
  - 操作码长度固定
  - 如8088，字长8位，指令结构可变

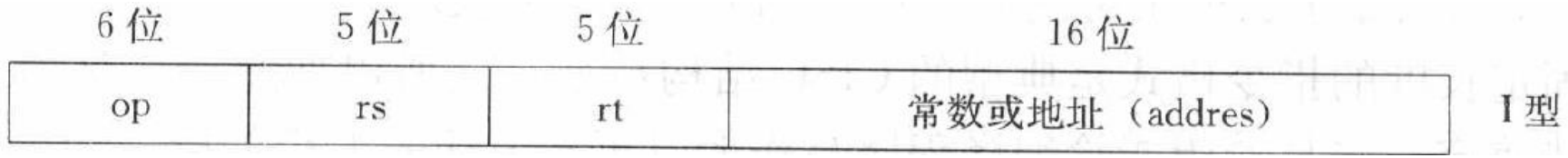


MIPS R400指令格式

- MIPS R4000：20世纪80年代，RISC系统
- 32位字长，32个通用寄存器，字节寻址
- R型（寄存器）指令：



- I型（立即数）指令：



MIPS R400指令格式

- 保持一致的指令格式
- R/I型指令：前3个字段（16位）相同
- 以op值区分指令

表 4.3　MIPS 指令的字段值

指令	格式	op	rs	rt	rd	shamt	funct	常数或地址
add（加）	R	0	reg	reg	reg	0	32	—
sub（减）	R	0	reg	reg	reg	0	34	—
立即数加	I	8	reg	reg	—	—	—	常数
lw（取字）	I	35	reg	reg	—	—	—	address
sw（存字）	I	43	reg	reg	—	—	—	address

ARM指令格式

- 嵌入式处理器
- 以32位ARM指令集的一种格式为例

cond	F	I	opcode	S	Rn	Rd	operand 2
4 位	2 位	1 位	4 位	1 位	4 位	4 位	12 位

各字段的含义如下：

opcode——指明指令的基本操作，称为操作码。

Rd——指明目标寄存器地址（4 位），共 16 个寄存器。

Rn——指明源寄存器地址（4 位），共 16 个寄存器。

operand 2——指明第 2 个源操作数。

I——指明立即数，如果 I=0，第 2 个源操作数在寄存器中；如果 I=1，第 2 个源操作数是 12 位的立即数。

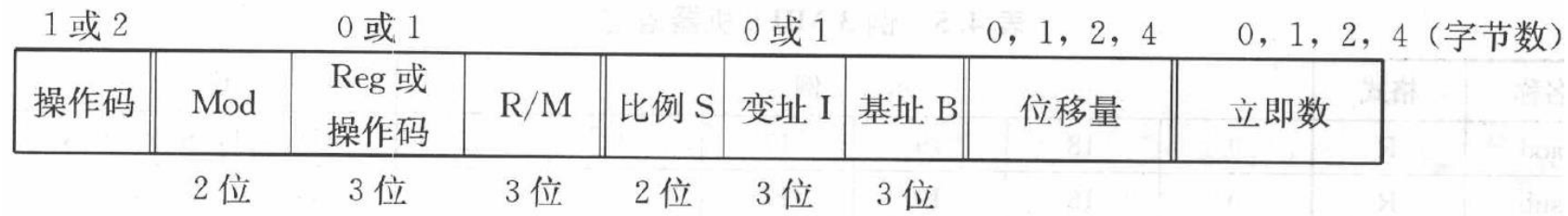
S——设置状态，该字段涉及条件转移指令。

cond——指明条件，该字段涉及条件转移指令。

F——说明指令类型，当需要时该字段允许设置不同的指令。

Pentium指令格式

- 指令长度可变
- 最短1个字节，最长12个字节
- 典型的CISC指令系统



- 组成
  - 必选：操作码
  - 可选：Mod-R/M、SIB、位移量、立即数
  - Mod-R/M字段规定存储器操作数寻址方式与寄存器号
  - Pentium采用RS型指令：只允许一个存储器操作数

总结

- 指令系统概述
- 简单指令系统（运算、访存、跳转）
  - 问题1：指令格式
  - 问题2：地址寻址
- 指令结构
- 典型指令系统（8位、ARM、Pentium）