**第四章、指令系统**

一、指令系统的发展和性能要求（P114-116）

1、指令系统的发展

\*指令：控制计算机执行某种操作的命令。一条指令可完成一个独立的算术/逻辑运算；是硬件和低层软件的接口。

\*指令的分类：

微指令（微程序级的命令）——硬件

宏指令（若干机器指令组成的软件指令）——软件

机器指令（介于微指令和宏指令之间，简称为指令）

\*程序：一系列机器指令组成的集合。

\*指令系统：所有机器指令的集合。——衡量计算机性能的重要因素，与软硬件都有关。

\*系列计算机：基本指令系统、基本结构相同的一类计算机。

\*复杂指令系统计算机（CISC）：研制周期长，硬件资源浪费。

使用频率高的指令，总的指令条数少；

\*精简指令系统计算机（RISC）：便于VLSI技术实现

\*指令集的二八规律： CISC中大约有20%使用频率高的指令占据了80%的处理时间，而有80%的不常用指令只占用处理机的20%时间。

\*RISC特点（采用流水线技术）：

简单而统一格式的指令译码

简单的寻址方式

较多的寄存器

对称的指令格式

大部分指令可以单周期执行

只有取/存数指令（LOAD/STORE）可以访问存储器，其余均在寄存器间进行

\*RISC和CISC的比较：

RISC更能充分利用 VLSI 芯片的面积；

RISC更利于提高计算机运算速度；

RISC有利于编译程序代码优化；

RISC 便于设计，可降低成本，提高可靠性

RISC不易实现指令系统兼容；

2、指令系统的性能要求

完整性：指令系统丰富、功能齐全、使用方便

有效性：高效运行、存储空间小、速度快

规整性：指令系统对称性、匀齐性、指令/数据格式一致性

兼容性：各机种上基本软件通用、向上兼容

3、低级结构与硬件结构的关系

\*程序设计：编写程序的过程——使用的工具：计算机语言

高级语言（C、JAVA、FORTRAN）与计算机硬件和指令系统无关

计算机语言

机器语言（二进制语言）： 指令代码

低级语言（计算机唯一能识别的语言）

（面向机器）

汇编语言（符号语言） ：指令助记符

均与硬件系统和指令系统有关

汇编器、编译器

高级语言、汇编语言 机器语言

**低级语言与高级语言比较**

|  |  |  |
| --- | --- | --- |
| **比较内容** | **高级语言** | **低级语言** |
| 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 |
| 对机器独立的程度 | 独立 | 不独立 |
| 编制程序的难易程度 | 易 | 难 |
| 编制程序所需时间 | 短 | 较长 |
| 程序执行时间 | 较长 | 较短 |
| 编译过程中对计算机资源的要求 | 多 | 少 |

二、指令格式（P116-121）

1、指令的结构

|  |  |
| --- | --- |
| 操作码字段OP | 地址码字段A |

操作功能特性 操作数地址

\*机器指令：用机器字表示

\*指令字（简称指令）：一条指令的机器字

\*指令格式：指令字用二进制代码表示的结构形式，由操作码字段和地址码字段组成

2、操作码OP（表示指令性质）——n位操作码最多2­­n指令

一般指令字中的操作码字段和地址码字段长度是固定等长（指令规整，译码简单）32位，而在单片机中操作码字段和地址码字段不固定

\*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。

3、地址码

\*按操作数地址个数分

零地址指令：只有操作吗，无地址码（停机、空操作、清除）

|  |  |
| --- | --- |
| OP |  |

一地址指令：一个地址码，有一个隐含操作数地址（单操作数运算指令，+1、-1、求反）

|  |  |
| --- | --- |
| OP | A |

AC←（AC）OP（A）

二地址指令：两个地址码段，其中A1既指明操作数地址，也指明存放数地址

|  |  |  |
| --- | --- | --- |
| OP | A1 | A2 |

A1←（A1）OP（A2）

三地址指令：A1源/被操作数地址 A2（终点）操作数地址 A3存放结果地址

|  |  |  |  |
| --- | --- | --- | --- |
| OP | A1 | A2 | A3 |

A3←（A1）OP（A2）

\* 按操作数物理地址分

存储器-存储器（SS）：访问内存 慢

寄存器-存储器（RS）：既访问内存也访问寄存器

寄存器-寄存器（RR）：访问寄存器 快

\*指令字长度：一个指令字中二进制位数

\*机器字长：计算机能直接处理的二进制位数

单字长指令—— 指令字长度=机器字长

半字长指令—— 指令字长度=0.5\*机器字长

双字长指令—— 指令字长度=2\*机器字长

\*多字长指令的优缺点

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

\*采用等长指令的优点：指令字结构简单，且指令字长度是不变的 ；

\*采用非等长指令的的优点：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

\*目前多采用的是32位固定长度的指令字长

\*指令助记符（如ADD、SUB）

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

必须用汇编语言翻译成二进制代码

4、指令格式举例

\*8位微型计算机的指令格式

如8088，字长8位，指令结构可变

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

\*MIPS R4000指令格式

RISC计算机系统，32位字长，32个通用寄存器。  
R型（寄存器）指令：

6位 5位 5位 5位 5位 6位

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| op | rs | rt | rd | shamt | funct |

I型（立即数）指令：  
6位 5位 5位 16位

|  |  |  |  |
| --- | --- | --- | --- |
| op | rs | rt | 常数或地址 |

\*ARM的指令格式

4位 2位 1位 4位 1位 4位 4位 12位

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Cond | F | I | Opcode | S | Rn | Rd | Operand2 |

三、操作数类型（P122-123）

地址数据:地址实际上也是一种形式的数据。地址看做无符号整数

数值数据:定点整数/小数、浮点数、压缩十进制数。

字符数据:文本数据或字符串，目前广泛使用ASCII码。

逻辑数据:一个单元中有几位二进制bit项组成，每个bit的值可以是1或0。

\*Power PC数据类型：

|  |  |
| --- | --- |
| 类型 | 说明 |
| 无符号字节 | 用于逻辑和整数算术运算 |
| 无符号半字 | 用于逻辑和整数算术运算，通用寄存器左端以0填充 |
| 有符号半字 | 用于16位的算术运算，通用寄存器左端以符号位填充 |
| 无符号字 | 用于32位逻辑运算、或作为地址指针 |
| 有符号字 | 用于32位算术运算 |
| 无符号双字 | 用于64位地址指针 |
| 字节串 | 0-128字节长 |
| 浮点数 | IEEE754标准中单、双精度浮点数 |

\*Pentium数据类型：

|  |  |
| --- | --- |
| 数据类型 | 说明 |
| 常规 | 字节、字（16位）、双字、四字 |
| 整数 | 字节、字、双字、四字中有符号二进制，补码表示 |
| 序数 | 字节、字、双字、四字中的无符号整数 |
| 未压缩的BCD | 0-9，每个字节表示一个BCD数字 |
| 压缩的BCD | 0-99，每个字节表示两个BCD数字 |
| 近指针 | 表示段内偏移的32位有效地址 |
| 位串 | 一个连续的位序列，最长为232-1 |
| 字符串 | 一个连续的字节、字或双字序列，最长为232-1 |
| 浮点数 | 单精度（32位）、双精度（64位）、扩展双精度（128） |

四、指令和数据的寻址方式（P123-129）

几乎所有计算机在内存中都采用地址指定方式，其形成操作数和指令地址的方式即寻址方式。

\*冯.诺依曼型计算机：指令的寻址和数据的寻址交替进行

\*哈佛型计算机：指令的寻址和数据的寻址独立进行

顺序寻址方式：必须用PC计数指令在内存的地址，由PC给出地址

\*指令的寻

址方式

跳跃寻址方式：下条指令地址不是由PC给出，而是由上条指令给出

|  |  |  |  |
| --- | --- | --- | --- |
| 操作码OP | 变址X | 间址I | 形式地址A |

寻址方式特征位 偏移量

\*操作数的寻址方式

隐含寻址：指令中隐含着操作数的地址

立即寻址：地址即操作数本身，数据就包含在指令中（立即数）

直接寻址：地址码字段的地址A就是操作数的有效地址EA（EA＝A）

间接寻址：地址码中的地址中A是操作数地址的指示器EA＝(A)

寄存器寻址：地址码给出CPU内某一通用寄存器的编号，指令的操作数存放在相应的寄存器中EA=Ri（如RR型指令）

寄存器间接寻址 ：操作数放在主存储器中，操作数的地址放在某一通用寄存器中，然后在地址码部分给出该通用寄存器的编号EA=(Ri)

偏移寻址 相对寻址：由PC提供基准地址，地址码给出相对的位移量D，两者相加后作为操作数的有效地址EA＝(PC)＋D。

基址寻址：基址寄存器的位数可以设置得很长，从而可以在较大的存储空间中寻址。

变址寻址：将地址码基准地址A与CPU内某特定的变址寄存器Rx中的内容相加，以形成操作数的有效地址

段寻址方式：存储空间划分为多段

堆栈寻址： 分寄存器堆栈、存储器堆栈，以先进后出原理存储数据，都与栈顶地址有关，进栈+1，退栈-1

五、典型指令（P130-134）

\*数据传送类指令（用于主存和寄存器间，或者寄存器和寄存器间）

一般传送指令：   MOV AX，BX

数据交换指令：   XCHG

堆栈操作指令：   PUSH，POP

\*运算类指令

算术运算指令（定点/浮点）： 加、减、乘、除以及加1、减1、比较

逻辑运算指令：移位 算术移位（不带符号位的移动）

逻辑移位（带符号位的移动）

\*程序控制类指令（转移指令）

程序控制类指令用于控制程序的执行方向，并使程序具有测试、分析与判断的能力。直接寻址——绝对转移 相对寻址——相对转移

\*输入和输出指令

\*字符串处理指令

\*特权指令

\*其他指令

六、ARM汇编语言（P134-136）

汇编语言是计算机机器语言（二进制指令代码）进行符号化的一种表示方法，每一个基本汇编语句对应一条机器指令。