第 1 篇　综合知识

第1章 计算机系统知识

1.1 备考指南

计算机组成与结构主要考查的是计算机硬件组成以及系统结构，包括计算机基本硬件组成、中央处理单元组成、计算机指令系统、存储系统、总线系统等相关知识，在软件设计师的考试中只会在选择题里考查，约占 6 分。

1.2 考点梳理及精讲

1.2.1 计算机系统基础知识

1. 计算机硬件组成

计算机的基本硬件系统由运算器、控制器、存储器、输入设备和输出设备 5 大部件组成。

（1）运算器、控制器等部件被集成在一起统称为中央处理单元（Central Processing Unit，CPU）。CPU 是硬件系统的核心，用于数据的加工处理，能完成各种算术运算、逻辑运算及控制功能。

（2）存储器是计算机系统中的记忆设备，分为内部存储器和外部存储器。前者速度快、容量小，一般用于临时存放程序、数据及中间结果；而后者容量大、速度慢，可以长期保存程序和数据。

（3）输入设备和输出设备合称为外部设备（简称外设），输入设备用于输入原始数据及各种命令，而输出设备则用于输出计算机运行的结果。

2. 中央处理单元（CPU）

（1）CPU 的功能。

1）程序控制。CPU 通过执行指令来控制程序的执行顺序，这是 CPU 的重要功能。

2）操作控制。一条指令功能的实现需要若干操作信号配合来完成，CPU 产生每条指令的操作信号并将操作信号送往对应的部件，控制相应的部件按指令的功能要求进行操作。

3）时间控制。CPU 对各种操作进行时间上的控制，即指令执行过程中操作信号的出现时间、

持续时间及出现的时间顺序都需要进行严格控制。

4）数据处理。CPU 通过对数据进行算术运算及逻辑运算等方式进行加工处理，数据加工处理的结果被人们所利用。所以，对数据的加工处理也是 CPU 最根本的任务。

此外，CPU 还需要对系统内部和外部的中断（异常）做出响应，并进行相应的处理。

（2）CPU 的组成：CPU 主要由运算器、控制器、寄存器组和内部总线等部件组成。

（3）运算器：由算术逻辑单元（Arithmetic and Logic Unit，ALU）（实现对数据的算术和逻辑运算）、累加寄存器（Accumulator，AC）（运算结果或源操作数的存放区）、数据缓冲寄存器（Data Register，DR）（暂时存放内存的指令或数据）、状态条件寄存器（Program Status Word，PSW）（保存指令运行结果的条件码内容，如溢出标志等）组成。执行所有的算术运算，如加减乘除等；执行所有的逻辑运算并进行逻辑测试，如与、或、非、比较等。

（4）控制器：由指令寄存器（Instruction Register，IR）（暂存 CPU 执行指令）、程序计数器（Program Counter，PC）（存放指令执行地址）、地址寄存器（Address Register，AR）（保存当前 CPU 所访问的内存地址）、指令译码器（Instruction Decoder，ID）（分析指令操作码）等组成。控制整个 CPU 的工作，十分重要。

CPU 依据指令周期的不同阶段来区分二进制的指令和数据，因为在指令周期的不同阶段，指令会命令 CPU 分别去取指令或者数据。

1.2.2 数据的表示

1. 进制的转化

（1）R 进制转十进制：位权展开法，用 R 进制数的每一位乘以 R 的 n 次方，n 是变量，从 R 进制数的最低位开始，依次为 0,1,2,3,…累加。

例如有六进制数 5043，此时 R=6，用六进制数的每一位乘以 6 的 n 次方，n 是变量，从六进制数的最低位开始（5043 从低位到高位排列：3,4,0,5），n 依次为 0,1,2,3，那么最终 $5043=3\times6^0+4\times6^1+0\times6^2+5\times6^3=1107$。

（2）十进制转 R 进制。

1）十进制整数（除以 R 倒取余数）。用十进制整数除以 R，记录每次所得的余数，若商不为 0，则继续除以 R，直至商为 0，而后将所有余数从下至上记录，排列成从左至右的顺序，即为转换后的 R 进制数。

2）十进制小数（乘 R 正取整数）。用十进制小数乘以 R，记录每次所得的整数，若结果小数部分不为 0，则将小数部分继续乘以 R，直至没有小数，而后将所有整数从第一个开始排列为从左至右的顺序，即为转换后的 R 进制数。

（3）m 进制转 n 进制：先将 m 进制转化为十进制数，再将十进制数转化为 n 进制数，中间需要通过十进制中转，但下面两种进制间可以直接转化：

1）二进制转八进制：每三位二进制数转换为一位八进制数，若二进制数位个数不是三的倍数，则在前面补 0，如二进制数 01101 有五位，前面补一个 0 就有六位，为 001 101，每三位转换为一

位八进制数，001=1,101=1+4=5，也即 01101=15。

2）二进制转十六进制：每四位二进制数转换为一位十六进制数，若二进制数位个数不是四的倍数，则在前面补 0，如二进制数 101101 有六位，前面补两个 0 就有八位，为 0010 1101，每四位转换为一位十六进制数，0010=2，1101=13=D，也即 101101=2D。

2. 数的表示

各种数值在计算机中表示的形式称为机器数，其特点是使用二进制计数制，数的符号用 0 和 1 表示，小数点则隐含，不占位置。

机器数有无符号数和带符号数之分。无符号数表示正数，没有符号位。带符号数最高位为符号位，正数符号位为 0，负数符号位为 1。定点表示法分为纯小数和纯整数两种，其中小数点不占存储位，而是按照以下约定。

纯小数：约定小数点的位置在机器数的最高数值位之前。

纯整数：约定小数点的位置在机器数的最低数值位之后。

真值：机器数对应的实际数值。

带符号数有下列编码方式。

原码：一个数的正常二进制表示，最高位表示符号，数值 0 的原码有两种形式：+0（0 0000000）和–0（1 0000000）。

反码：正数的反码即原码；负数的反码是在原码的基础上，除符号位外，其他各位按位取反。数值 0 的反码也有两种形式：+0（0 0000000）和–0（1 1111111）。

补码：正数的补码即原码；负数的补码是在原码的基础上，除符号位外，其他各位按位取反，而后末位+1，若有进位则产生进位。因此数值 0 的补码只有一种形式+0 = –0 = 0 0000000。

移码：用作浮点运算的阶码，无论正数还是负数，都是将该原码的补码的首位（符号位）取反得到移码。

符号表示：要注意的是，原码最高位是代表正负号，且不参与计数；而其他编码最高位虽然也是代表正负号，但参与计数。各编码取值范围见表 1-1。

表 1-1　机器字长为 n 时各种码制表示的带符号数的取值范围

码制	定点整数	定点小数
原码	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$	$-(1-2^{-(n-1)})\sim+(1-2^{-(n-1)})$
反码	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$	$-(1-2^{-(n-1)})\sim+(1-2^{-(n-1)})$
补码	$-2^{n-1}\sim+(2^{n-1}-1)$	$-1\sim+(1-2^{-(n-1)})$
移码	$-2^{n-1}\sim+(2^{n-1}-1)$	$-1\sim+(1-2^{-(n-1)})$

差别在于 0 的表示，原码和反码分+0 和–0，补码只有一个 0。

3. 浮点数的运算

浮点数：表示方法为 $N = F\times2^{E}$，其中 E 称为阶码，F 称为尾数；类似于十进制的科学计数法，

如 $85.125 = 0.85125\times10^2$，二进制如 $101.011 = 0.101011\times2^3$。

在浮点数的表示中，阶码为带符号的纯整数，尾数为带符号的纯小数，要注意符号占最高位（正数为 0，负数为 1），其表示格式如下：

阶符　阶码　数符　尾数

很明显，与科学计数法类似，一个浮点数的表示方法不是唯一的，浮点数所能表示的数值范围由阶码确定，所表示的数值精度由尾数确定。

尾数的表示采用规格化方法，也即带符号尾数的补码必须为 1.0xxxx（负数）或者 0.1xxxx（正数），其中 x 可为 0 或 1。

浮点数的运算：对阶（使两个数的阶码相同，小阶向大阶看齐，较小阶码增加几位，尾数就右移几位）——尾数计算（相加，若是减运算，则加负数）——结果规格化（即尾数表示规格化，带符号尾数转换为 1.0xxxx 或 0.1xxxx）。

4. 算术运算和逻辑运算

数与数之间的算术运算包括加、减、乘、除等基本算术运算，对于二进制数，还应该掌握基本逻辑运算，包括：

逻辑与（&&）：0 和 1 相与，只要有一个为 0 结果就为 0，两个都为 1 才为 1。

逻辑或（||）：0 和 1 相或，只要有一个为 1 结果就为 1，两个都为 0 才为 0。

异或（⊕）：同 0 非 1，即参加运算的二进制数同为 0 或者同为 1 结果为 0，一个为 0 另一个为 1 结果为 1。

逻辑非（!）：0 的非是 1，1 的非是 0。

逻辑左移（<<）：二进制数整体左移 n 位，高位若溢出则舍去，低位补 0。

逻辑右移（>>）：二进制数整体右移 n 位，低位溢出则舍去，高位补 0。

算术左移、算术右移：乘以 2 或者除以 2 的算术运算，涉及加、减、乘、除的都是算术运算，与逻辑运算区分。

短路计算方式：指通过逻辑运算符（&&、‖等）左边表达式的值就能推算出整个表达式的值，不再继续执行逻辑运算符右边的表达式。

若计算机存储数据采用的是双符号位（00 表示正号、11 表示负号），两个符号相同的数相加时，如果运算结果的两个符号位经异或运算得 1，则可断定这两个数相加的结果产生了溢出。

从计算的角度理解，正数和负数相加其结果肯定不会溢出，如果有溢出，必然同为正数或者同为负数，结果才会更大，才有可能溢出。因此，正常两个同符号数相加时，不考虑溢出，其符号位必然还是 00 或者 11，如果有溢出，那么数据位必然最高位进位 1，符号位就需要加 1，变为 01 或者 10，因此当符号位为 01 或者 10 时数据溢出。观察这两种溢出情况的两个符号位都是一个为 0，一个为 1，其异或运算必然为 1，没有其他可能，而逻辑或运算有可能两个都为 1，也能得出 1。

1.2.3 校验码

1. 码距

就单个编码 A：00 而言，其码距为 1，因为其只需要改变一位就变成另一个编码。在两个编码中，从 A 码到 B 码转换所需要改变的位数称为码距，如 A：00 要转换为 B：11，码距为 2。一般来说，码距越大，越利于检错和纠错。

2. 奇偶校验码

在编码中增加 1 位校验位来使编码中 1 的个数为奇数（奇校验）或者偶数（偶校验），从而使码距变为 2。奇校验可以检测编码中奇数个数据位出错，即当合法编码中的奇数位发生了错误时，即编码中的 1 变成 0，或者 0 变成 1，则该编码中 1 的个数的奇偶性就发生了变化，从而检查出错误，但无法纠错。

3. 循环冗余校验码（CRC）

CRC 只能检错，不能纠错，其原理是找出一个能整除多项式的编码，因此首先要将原始报文除以多项式，将所得的余数作为校验位加在原始报文之后，作为发送数据发给接收方。

使用 CRC 编码，需要先约定一个生成多项式 G(x)。生成多项式的最高位和最低位必须是 1。假设原始信息有 m 位，则对应多项式 M(x)。生成校验码的思想就是在原始信息位后追加若干校验位，使得追加的信息能被 G(x)整除。接收方接收到带校验位的信息，然后用 G(x)整除，余数为 0，则没有错误；反之则发生错误。

例：假设原始信息串为 10110，CRC 的生成多项式为 $G(x)=x^4+x+1$，求 CRC 校验码。

（1）在原始信息位后面添 0，假设生成多项式的阶为 r，则在原始信息位后添加 r 个 0，本题中，G(x)阶为 4，则在原始信息串后加 4 个 0，得到的新串为 101100000，作为被除数。

（2）由多项式得到除数，多项式中 x 的幂指数存在的位置 1，不存在的位置 0。本题中，x 的幂指数为 0、1、4 的变量都存在，而幂指数为 2、3 的不存在，因此得到串 10011。

（3）生成 CRC 校验码，将前两步得出的被除数和除数进行模 2 除法运算（即不进位也不借位的除法运算）。除法过程如图 1-1 所示。

```
          10110
      ┌─────────
10011 ) 101100000
        10011
        -----
          10100
          10011
          -----
            11100
            10011
            -----
             1111
```

图 1-1 模 2 除法运算

得到余数 1111。

注意：余数不足 r，则余数左边用若干个 0 补齐。如求得余数为 11，r=4，则补两个 0 得到 0011。

（4）生成最终发送信息串，将余数添加到原始信息后。上例中，原始信息为 10110，添加余数 1111 后，结果为 10110 1111。发送方将此数据发送给接收方。

（5）接收方进行校验。接收方的 CRC 校验过程与生成过程类似，接收方接收了带校验和的帧后，用多项式 G(x)来除。余数为 0，则表示信息无错；否则要求发送方进行重传。

注意：收发信息双方需使用相同的生成多项式。

4. 海明校验码

海明校验码本质也是使用奇偶校验方式检验，通过下面例题详解海明校验码。

例：求信息 1011 的海明码。

（1）校验位的位数和具体的数据位的位数之间的关系。所有位都编号，最低位编号从 1 开始递增，校验位处于 2 的 n（n=0,1,2,…）次方中，即处于第 1,2,4,8,16,32,…位上，其余位才能填充真正的数据位，若信息数据为 1011，则可知，第 1,2,4 位为校验位，第 3,5,6,7 位为数据位，用来从低位开始存放 1011，得出信息位和校验位分布见表 1-2。

表 1-2 信息位和校验位分布表

7	6	5	4	3	2	1	位数
I_4	I_3	I_2		I_1			信息位
			r_2	r_1	r_0		校验位

实际考试时可以依据公式 $n+k \leqslant 2^k-1$ 快速得出答案（n 是已知的数据位个数，k 是未知的校验位个数，依次取 k=1 代入计算，得出满足上式的最小的 k 的值）。

（2）计算校验码。将所有信息位的编号都拆分成二进制表示，如下所示：

$7=2^2+2^1+2^0$，$6=2^2+2^1$，$5=2^2+2^0$，$3=2^1+2^0$；

$r_2=I_4 \oplus I_3 \oplus I_2$；

$r_1=I_4 \oplus I_3 \oplus I_1$；

$r_0=I_4 \oplus I_2 \oplus I_1$；

7=4+2+1，表示 7 由第 4 位校验位（r_2）和第 2 位校验位（r_1）和第 1 位校验位（r_0）共同校验，同理，第 6 位数据位 6=4+2，第 5 位数据位 5=4+1，第 3 位数据位 3=2+1，前面知道，这些 2 的 n 次方都是校验位，可知，第 4 位校验位校验第 7、6、5 三位数据位，因此，第 4 位校验位 r_2 等于这三位数据位的值异或，第 2 位和第 1 位校验位计算原理同上，最终结果见表 1-3。

表 1-3 最终生成的校验码

7	6	5	4	3	2	1	位数
1	0	1		1			信息位
			0		0	1	校验位

计算出三个校验位后，可知最终要发送的海明校验码为 1010101。

（3）检错和纠错原理。接收方收到海明码之后，会将每一位校验位与其校验的位数分别异或，即做如下三组运算：

$r_2 \oplus I_4 \oplus I_3 \oplus I_2$

$r_1 \oplus I_4 \oplus I_3 \oplus I_1$

$r_0 \oplus I_4 \oplus I_2 \oplus I_1$

如果是偶校验，那么运算得到的结果应该全为 0，如果是奇校验，应该全为 1，才正确（若采用奇校验，则前面计算校验码时需要将各校验位的偶校验值取反），假设是偶校验，且接收到的数据为 1011101（第四位出错），此时，运算的结果为：

$r_2 \oplus I_4 \oplus I_3 \oplus I_2 = 1 \oplus 1 \oplus 0 \oplus 1 = 1$

$r_1 \oplus I_4 \oplus I_3 \oplus I_1 = 0 \oplus 1 \oplus 0 \oplus 1 = 0$

$r_0 \oplus I_4 \oplus I_2 \oplus I_1 = 1 \oplus 1 \oplus 1 \oplus 1 = 0$

这里不全为 0，表明传输过程有误，并且按照 $r_2r_1r_0$ 排列为二进制 100，这里指出的就是错误的位数，表示第 100 位，即第 4 位出错，找到了出错位，纠错方法就是将该位逆转。

1.2.4 计算机体系结构

计算机体系结构按处理机的数量进行分类：单处理系统（一个处理单元和其他设备集成）、并行处理系统（两个以上的处理机互联）、分布式处理系统（物理上远距离且松耦合的多计算机系统）。

除此之外，一个典型的分类方法是 Flynn 分类法，见表 1-4。

表 1-4 Flynn 分类法

体系结构类型	结果	关键特性	代表
单指令流单数据流 SISD	控制部分：一个 处理器：一个 主存模块：一个	串行结构，一条指令执行完才能执行下一条指令	单处理器系统
单指令流多数据流 SIMD	控制部分：一个 处理器：多个 主存模块：多个	各处理器以异步的形式执行同一条指令	并行处理机　阵列处理机　超级向量处理机
多指令流单数据流 MISD	控制部分：多个 处理器：一个 主存模块：多个	被证明不可能，至少是不实际	目前没有，有文献称流水线计算机为此类
多指令流多数据流 MIMD	控制部分：多个 处理器：多个 主存模块：多个	能够实现作业、任务、指令等各级全面并行	多处理机系统 多计算机

由表 1-4 可知，分类有两个因素，即指令流和数据流。指令流由控制部分处理，每一个控制部分处理一条指令流，多指令流就有多个控制部分；数据流由处理器来处理，每一个处理器处理一条

数据流，多数据流就有多个处理器。至于主存模块，是用来存储的，存储指令流或者数据流，因此，无论是多指令流还是多数据流，都需要多个主存模块来存储，对于主存模块，指令和数据都一样。

依据计算机特性，是由指令来控制数据的传输，因此，一条指令可以控制一条或多条数据流，但一条数据流不能被多条指令控制，否则会出错，就如同上级命令太多还互相冲突，不知道该执行哪一个，因此多指令单数据（MISD）不可能。

1.2.5 指令系统

1. 计算机指令的组成

一条指令由操作码和操作数两部分组成，操作码决定要完成的操作，操作数指参加运算的数据及其所在的单元地址。

在计算机中，操作要求和操作数地址都由二进制数码表示，分别称作操作码和地址码，整条指令以二进制编码的形式存放在存储器中。

2. 计算机指令执行过程

计算机指令执行过程可分为取指令、分析指令、执行指令三个步骤，首先将程序计数器（PC）中的指令地址取出，送入地址总线，CPU 依据指令地址去内存中取出指令内容存入指令寄存器（IR）；而后由指令译码器进行分析，分析指令操作码；最后执行指令，取出指令执行所需的源操作数。

3. 指令寻址方式

顺序寻址方式：由于指令地址在主存中顺序排列，当执行一段程序时，通常是一条指令接着一条指令地顺序执行。从存储器取出第一条指令，然后执行这条指令；接着从存储器取出第二条指令，再执行第二条指令，以此类推。这种程序顺序执行的过程称为指令的顺序寻址方式。

跳跃寻址方式：所谓指令的跳跃寻址，是指下一条指令的地址码不是由程序计数器给出，而是由本条指令直接给出。程序跳跃后，按新的指令地址开始顺序执行。因此，指令计数器的内容也必须相应改变，以便及时跟踪新的指令地址。

4. 指令操作数的寻址方式

立即寻址方式：指令的地址码字段指出的不是地址，而是操作数本身。

直接寻址方式：在指令的地址字段中直接指出操作数在主存中的地址。

间接寻址方式：与直接寻址方式相比，间接寻址中指令地址码字段所指向的存储单元中存储的不是操作数本身，而是操作数的地址。

寄存器寻址方式：指令中的地址码是寄存器的编号，而不是操作数地址或操作数本身。寄存器的寻址方式也可以分为直接寻址和间接寻址，两者的区别在于前者的指令地址码给出寄存器编号，寄存器的内容就是操作数本身；而后者的指令地址码给出寄存器编号，寄存器的内容是操作数的地址，根据该地址访问主存后才能得到真正的操作数。

基址寻址方式：将基址寄存器的内容加上指令中的形式地址而形成操作数的有效地址，其优点是可以扩大寻址能力。

变址寻址方式：变址寻址方式计算有效地址的方法与基址寻址方式很相似，它是将变址寄存器

的内容加上指令中的形式地址而形成操作数的有效地址。

相对寻址方式：相对于当前的指令地址而言的寻址方式。相对寻址是把程序计数器（PC）的内容加上指令中的形式地址而形成操作数的有效地址，而程序计数器的内容就是当前指令的地址，所以相对寻址是相对于当前的指令地址而言的。

5. CISC 和 RISC

CISC 是复杂指令系统，兼容性强，指令繁多、长度可变，由微程序实现。

RISC 是精简指令系统，指令少，使用频率接近，主要依靠硬件实现（通用寄存器、硬布线逻辑控制）。

二者各方面的区别见表 1-5。

表 1-5　CISC 和 RISC 的区别

指令系统类型	指令	寻址方式	实现方式	其他
CISC（复杂）	数量多，使用频率差别大，可变长格式	支持多种	微程序控制技术（微码）	研制周期长
RISC（精简）	数量少，使用频率接近，定长格式，大部分为单周期指令，操作寄存器，只有 Load/Store 操作内存	支持方式少	增加了通用寄存器；硬布线逻辑控制为主；适合采用流水线	优化编译，有效支持高级语言

6. 流水线原理

将指令分成不同段，每段由不同的部分去处理，因此可以产生叠加的效果，所有的部件去处理指令的不同段，如下图所示：

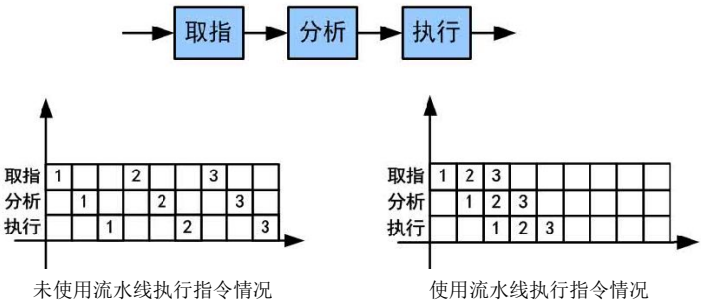

图 1-2　未使用流水线和使用流水线执行对比

RISC 中的流水线技术：

超流水线（Super Pipe Line）技术。它通过细化流水、增加级数和提高主频，使得在每个机器周期内能完成一个甚至两个浮点操作。其实质是以时间换取空间。

超标量（Super Scalar）技术。它通过内装多条流水线来同时执行多个处理，其时钟频率虽然

与一般流水接近，却有更小的 CPI。其实质是以空间换取时间。

超长指令字（Very Long Instruction Word，VLIW）技术。VLIW 和超标量都是 20 世纪 80 年代出现的概念，其共同点是要同时执行多条指令，其不同在于超标量依靠硬件来实现并行处理的调度，VLIW 则充分发挥软件的作用，而使硬件简化，性能提高。

7. 流水线时间计算

流水线相关计算公式如下：

流水线周期：指令分成不同执行段，其中执行时间最长的段为流水线周期。

流水线执行时间：1 条指令总执行时间+（总指令条数–1）×流水线周期。

流水线吞吐率计算：吞吐率即单位时间内执行的指令条数。公式：指令条数/流水线执行时间。

流水线的加速比计算：加速比即使用流水线后的效率提升度，即比不使用流水线快了多少倍，越高表明流水线效率越高，公式：不使用流水线执行时间/使用流水线执行时间。

单缓冲区和双缓冲区：此类题型不给出具体流水线执行阶段，需要考生自己区分出流水线阶段，一般来说，能够同时执行的阶段就是流水线的独立执行阶段；只能独立执行的阶段应该合并为流水线中的一个独立执行阶段。

例如有三个阶段，即读入缓冲区+送入用户区+数据处理，在单缓冲区中，缓冲区和用户区都只有一个，一个盘块必须执行完前两个阶段，下一个盘块才能开始，因此前两个阶段应该合并，整个流水线为送入用户区+数据处理；而在双缓冲区中，盘块可以交替读入缓冲区，但用户区只有一个，因为缓冲区阶段可以同时进行，流水线前两个阶段不能合并，就是读入缓冲区+送入用户区+数据处理三段。

划分出真正的流水线阶段后，套用流水线时间计算公式可以轻易得出答案。

1.2.6 存储系统

1. 计算机存储结构层次结构

计算机存储结构层次图如图 1-3 所示。

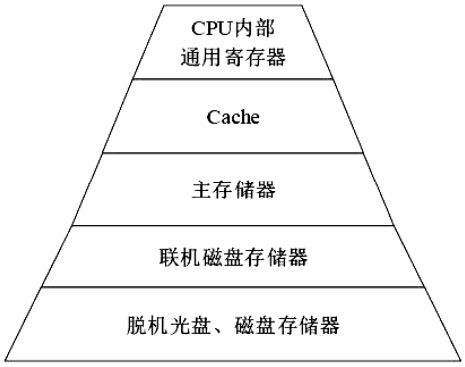

图 1-3 计算机存储结构层次图

计算机采用分级存储体系的主要目的是解决存储容量、成本和速度之间的矛盾问题。

两级存储映像为 Cache-主存、主存-辅存（虚拟存储体系）。

2. 存储器的分类

按存储器所处的位置可分为：内存、外存。

按存储器构成材料：磁存储器（磁带）、半导体存储器、光存储器（光盘）。

按存储器的工作方式：可读可写存储器（RAM）、只读存储器（ROM 只能读，PROM 可写入一次，EPROM 和 EEPOM 既可以读也可以写，只是修改方式不同，闪存 Flash Memory）。

按存储器访问方式：按地址访问、按内容访问（相联存储器）。

按寻址方式：随机存储器（访问任意存储单元所用时间相同）、顺序存储器（只能按顺序访问，如磁带）、直接存储器（二者结合，如磁盘，对于磁道的寻址是随机的，在一个磁道内则是顺序的）。

3. 局部性原理

总的来说，在 CPU 运行时，所访问的数据会趋向于一个较小的局部空间地址内（例如循环操作，循环体被反复执行）。

时间局部性原理：如果一个数据项正在被访问，那么在近期它很可能会被再次访问，即在相邻的时间里会访问同一个数据项。

空间局部性原理：在最近的将来会用到的数据的地址和现在正在访问的数据地址很可能是相近的，即相邻的空间地址会被连续访问。

4. 高速缓存 Cache

（1）Cache 的组成。高速缓存 Cache 用来存储当前最活跃的程序和数据，直接与 CPU 交互，位于 CPU 和主存之间，容量小，速度为内存的 5～10 倍，由半导体材料构成。其内容是主存内存的副本拷贝，对于程序员来说是透明的。

Cache 由控制部分和存储器组成，存储器存储数据，控制部分判断 CPU 要访问的数据是否在 Cache 中，在则命中，不在则依据一定的算法从主存中替换，如图 1-4 所示。

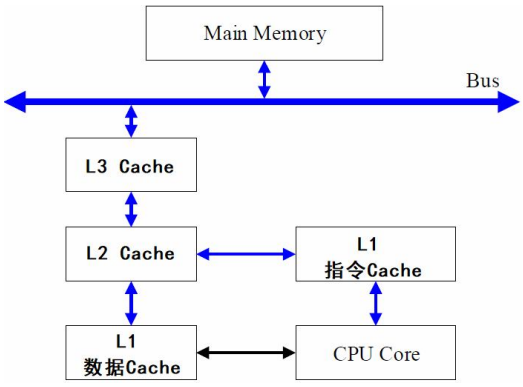

图 1-4　Cache 存储结构

（2）Cache 地址映像方法。在 CPU 工作时，送出的是主存单元的地址，应从 Cache 存储器中读/写信息。这就需要将主存地址转换为 Cache 存储器地址，这种地址的转换称为地址映像，由**硬件自动完成映像**，分为下列三种方法。

直接映像：将 Cache 存储器等分成块，主存也等分成块并编号。主存中的块与 Cache 中的块的对应关系是固定的，也即二者块号相同才能命中。地址变换简单但不灵活，容易造成资源浪费。

全相联映像：同样都等分成块并编号。主存中任意一块都与 Cache 中任意一块对应。因此可以随意调入 Cache 任意位置，但地址变换复杂，速度较慢。因为主存可以随意调入 Cache 任意块，只有当 Cache 满了才会发生块冲突，是最不容易发生块冲突的映像方式。

组相联映像：前面两种方式的结合，将 Cache 存储器先分块再分组，主存也同样先分块再分组，组间采用直接映像，即主存中组号与 Cache 中组号相同的组才能命中，但是组内全相联映像，也即组号相同的两个组内的所有块可以任意调换。

（3）Cache 替换算法。目标就是使 Cache 获得尽可能高的命中率。常用算法有如下几种。

1）随机替换算法。就是用随机数发生器产生一个要替换的块号，将该块替换出去。

2）先进先出算法。就是将最先进入 Cache 的信息块替换出去。

3）近期最少使用算法。这种方法是将近期最少使用的 Cache 中的信息块替换出去。

4）优化替换算法。这种方法必须先执行一次程序，统计 Cache 的替换情况。有了这样的先验信息，在第二次执行该程序时便可以用最有效的方式来替换。

（4）Cache 命中率及平均时间。Cache 存储器的大小一般为 KB 或者 MB 单位，很小，但是最快，仅次于 CPU 中的寄存器，而寄存器一般不算作存储器，CPU 与内存之间的数据交互，内存会先将数据拷贝到 Cache 里，这样，根据局部性原理，若 Cache 中的数据被循环执行，则不用每次都去内存中读取数据，会加快 CPU 工作效率。

因此，Cache 有一个命中率的概念，即当 CPU 所访问的数据在 Cache 中时，命中，直接从 Cache 中读取数据，设读取一次 Cache 时间为 1ns，若 CPU 访问的数据不在 Cache 中，则需要从内存中读取，设读取一次内存的时间为 1000ns，若在 CPU 多次读取数据过程中，有 90%命中 Cache，则 CPU 读取一次的平均时间为(90%×1 + 10%×1000)ns。

Cache 命中率和容量之间并不是线性关系，而是如图 1-5 所示的曲线增长关系。

5. 虚拟存储器

虚拟存储器技术是将很大的数据分成许多较小的块，全部存储在外存中。运行时，将用到的数据调入主存中，马上要用到的数据置于缓存中，这样，一边运行一边进行所需数据块的调入/调出。对于应用程序来说，就好像有一个比实际主存空间大得多的虚拟主存空间，基本层级为：主存——缓存——外存，与 CPU——高速缓存 Cache——主存的原理类似。但虚拟存储器中程序员无需考虑地址映像关系，由系统自动完成，因此对于程序来说是透明的。

其管理方式分为页式、段式、段页式，在 2.2.3 存储管理中详细介绍。

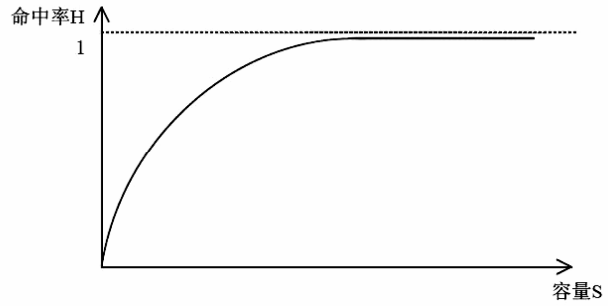

图 1-5　Cache 命中率和容量之间的关系

6. 磁盘

（1）磁盘结构和参数。磁盘有正反两个盘面，每个盘面有多个同心圆，每个同心圆是一个磁道，每个同心圆又被划分为多个扇区，数据就被存放在一个个扇区中。

磁头首先要寻找到对应的磁道，然后等待磁盘进行周期旋转，旋转到指定的扇区，才能读取到对应的数据，因此，会产生寻道时间和等待时间。公式为：

存取时间=寻道时间+等待时间（平均定位时间+转动延迟）

注意：寻道时间是指磁头移动到磁道所需的时间；等待时间为等待读写的扇区转到磁头下方所用的时间。

（2）磁盘调度算法。之前已经说过，磁盘数据的读取时间分为寻道时间+旋转时间，也即先找到对应的磁道，而后再旋转到对应的扇区才能读取数据，其中寻道时间耗时最长，需要重点调度，有如下调度算法。

先来先服务（First Come First Service，FCFS）：根据进程请求访问磁盘的先后顺序进行调度。

最短寻道时间优先（Shortest Seek Time First，SSTF）：请求访问的磁道与当前磁道最近的进程优先调度，使得每次的寻道时间最短。会产生“饥饿”现象，即远处进程可能永远无法访问。

扫描算法（SCAN）：又称“电梯算法”，磁头在磁盘上双向移动，其会选择离磁头当前所在磁道最近的请求访问的磁道，并且与磁头移动方向一致，磁头永远都是从里向外或者从外向里一直移动完才掉头，与电梯类似。

单向扫描调度算法（CSCAN）：与 SCAN 不同的是，其只做单向移动，即只能从里向外或者从外向里。

（3）磁盘冗余阵列技术。RAID 即磁盘冗余阵列技术，RAID0 没有提供冗余和错误修复技术。

RAID1 在成对的独立磁盘上产生互为备份的数据，可提高读取性能。

RAID2 将数据条块化地分布于不同硬盘上，并使用海明码校验。

RAID3 使用奇偶校验，并用单块磁盘存储奇偶校验信息。

RAID5 在所有磁盘上交叉地存储数据及奇偶校验信息（所有校验信息存储总量为一个磁盘容量），读/写指针可同时操作。

RAID0+1（是两个 RAID0，若一个磁盘损坏，则当前 RAID0 无法工作，即有一半的磁盘无法工作），RAID1+0（是两个 RAID1，不允许同一组中的两个磁盘同时损坏）与 RAID1 原理类似，磁盘利用率都只有 50%。

1.2.7 输入输出技术

1. 内存和接口编址

计算机系统中存在多种内存与接口地址的编址方法，常见的是下面两种：内存与接口地址独立编址和内存与接口地址统一编址。

（1）内存与接口地址独立编址。在内存与接口地址独立编址方法下，内存地址和接口地址是完全独立且相互隔离的两个地址空间。访问数据时所使用的指令也完全不同，用于接口的指令只用于接口的读/写，其余的指令全都是用于内存的。因此，在编程序或读程序时很容易使用和辨认。这种编址方法的缺点是用于接口的指令太少、功能太弱。

（2）内存与接口地址统一编址。在这种编址方法中，内存地址和接口地址统一在一个公共的地址空间里，即内存单元和接口共用地址空间。在这些地址空间里划分出一部分地址分配给接口使用，其余地址归内存单元使用。这种编址方法的优点是原则上用于内存的指令全都可以用于接口，这就大大地增强了对接口的操作功能，而且在指令上也不再区分内存或接口指令。该编址方法的缺点是整个地址空间被分成两部分，其中一部分分配给接口使用，剩余的为内存所用，这经常会导致内存地址不连续。

2. 外设数据传输方式

程序控制（查询）方式：CPU 主动查询外设是否完成数据传输，效率极低。

程序中断方式：外设完成数据传输后，向 CPU 发送中断请求，等待 CPU 处理数据，效率相对较高。

DMA 方式（直接内存存取）：CPU 只需完成必要的初始化等操作，数据传输的整个过程都由 DMA 控制器来完成，在内存和外设之间建立直接的数据通路，效率很高。在一个总线周期结束后，CPU 会响应 DMA 请求开始读取数据；CPU 响应程序中断方式请求是在一条指令执行结束时；区分指令执行结束和总线周期结束。

通道：也是一种处理机，内部具有独立的处理系统，使数据的传输独立于 CPU。分为字节多路通道的传送方式（每一次传送一个通道的一个字节，多路通道循环）和选择通道的传送方式（选择一个通道，先传送完这个通道的所有字节，再开始下一个通道传送）。

3. 中断原理

中断：指 CPU 在正常运行程序时，由于程序的预先安排或内外部事件，引起 CPU 中断正在运行的程序，转到发生中断事件程序中。

中断源：引起程序中断的事件称为中断源。

中断向量：中断源的识别标志，中断服务程序的入口地址。

中断向量表：按照中断类型号从小到大的顺序存储对应的中断向量，总共存储 256 个中断向量。

中断流程图如图 1-6 所示。

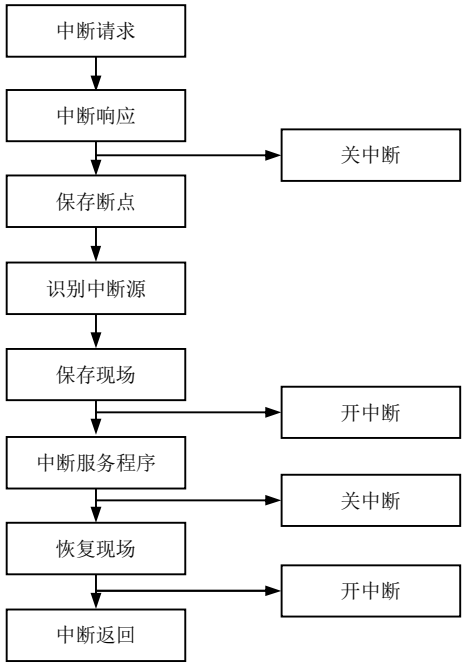

图 1-6　中断流程图

中断响应：CPU 在执行当前指令的最后一个时钟周期去查询有无中断请求信号，有则响应。

关中断：在保护现场和恢复现场过程中都要先关闭中断，避免堆栈错误。

保护断点：是保存程序当前执行的位置。

保护现场：是保存程序当前断点执行所需的寄存器及相关数据。

中断服务程序：识别中断源，获取到中断向量，就能进入中断服务程序，开始处理中断。

中断返回：返回中断前的断点，继续执行原来的程序。

中断响应时间指的是从发出中断请求到开始进入中断处理程序；中断处理时间指的是从中断处理开始到中断处理结束。中断向量提供中断服务程序的入口地址。多级中断嵌套，使用堆栈来保护断点和现场。

1.2.8　总线结构

总线（Bus）是指计算机设备和设备之间传输信息的公共数据通道。总线是连接计算机硬件系

统内多种设备的通信线路，它的一个重要特征是由总线上的所有设备共享，因此可以将计算机系统内的多种设备连接到总线上。

从广义上讲，任何连接两个以上电子元器件的导线都可以称为总线，通常分为以下三类。

（1）**内部总线**：内部芯片级别的总线，芯片与处理器之间通信的总线。

（2）**系统总线**：板级总线，用于计算机内各部分之间的连接，具体分为数据总线（并行数据传输位数）、地址总线（系统可管理的内存空间的大小）、控制总线（传送控制命令）。代表的有 ISA 总线、EISA 总线、PCI 总线。

（3）**外部总线**：设备一级的总线，微机和外部设备的总线。代表的有 RS232（串行总线）、SCSI（并行总线）、USB（通用串行总线，即插即用，支持热插拔）。

并行总线适合近距离高速数据传输，串行总线适合长距离数据传输，专用总线在设计上可以与连接设备实现最佳匹配。

总线计算：总线的时钟周期=时钟频率的倒数；总线的宽度（传输速率）=单位时间内传输的数据总量/单位时间大小。

1.2.9 系统可靠性分析

1. 可靠性指标

平均无故障时间（MTTF）=1/失效率。

平均故障修复时间（MTTR）=1/修复率。

平均故障间隔时间（MTBF）=MTTF+MTTR。

系统可用性=MTTF/(MTTF+MTTR)×100%。

2. 串并联系统可靠性

无论什么系统，都是由多个设备组成的，协同工作，而这多个设备的组合方式可以是串联、并联，也可以是混合模式，假设每个设备的可靠性为 $R_1,R_2,\cdots,R_n$，则不同的系统的可靠性公式如下。

串联系统，一个设备不可靠，整个系统崩溃，整个系统可靠性 $R=R_1 \times R_2 \times\cdots\times R_n$。原理如图 1-7 所示。

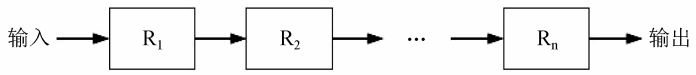

图 1-7 串联系统可靠性

并联系统，所有设备都不可靠，整个系统才崩溃，整个系统可靠性 $R=1-(1-R_1)\times(1-R_2)\times\cdots\times(1-R_n)$。原理如图 1-8 所示。

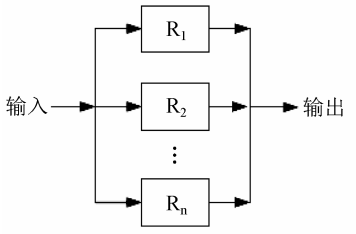

图 1-8　并联系统可靠性

N 模冗余系统：N 模冗余系统由 N 个（N=2n+1）相同的子系统和一个表决器组成，表决器把 N 个子系统中占多数相同结果的输出作为输出系统的输出，如图 1-9 所示。在 N 个子系统中，只要有 n+1 个或 n+1 个以上子系统能正常工作，系统就能正常工作，输出正确的结果。

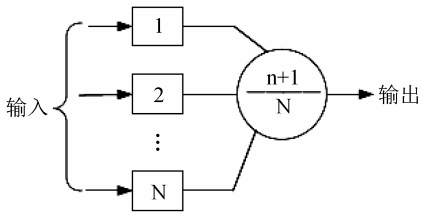

图 1-9　表决器可靠性

1.2.10　计算机系统的性能评测

1. 性能评测的常用方法

（1）时钟频率。一般来讲，主频越高，速度越快。

（2）指令执行速度。计量单位 KIPS、MIPS。

（3）等效指令速度法。统计各类指令在程序中所占的比例，并进行折算，是一种固定比例法。

（4）数据处理速率（Processing Data Rate，PDR）法。采用计算 PDR 值的方法来衡量机器性能，PDR 值越大，机器性能越好。PDR 与每条指令和每个操作数的平均位数以及每条指令的平均运算速度有关。

（5）核心程序法。把应用程序中用得最频繁的那部分核心程序作为评价计算机性能的标准程序，在不同的机器上运行，测得其执行时间，作为各类机器性能评价的依据。

2. 基准程序法（Benchmark）

基准程序法是目前被用户一致承认的测试性能的较好方法，有多种多样的基准程序，主要包括以下内容：

（1）整数测试程序。同一厂家的机器，采用相同的体系结构，用相同的基准程序测试，得到的 MIPS 值越大，一般说明机器速度越快。

（2）浮点测试程序。指标 MFLOPS（理论峰值浮点速度）。

（3）SPEC 基准程序（SPEC Benchmark）。重点面向处理器性能的基准程序集，将被测计算机的执行时间标准化，即将被测计算机的执行时间除以一个参考处理器的执行时间。

（4）TPC 基准程序。用于评测计算机在事务处理、数据库处理、企业管理与决策支持系统等方面的性能。其中，TPC-C 是在线事务处理（On-Line Transaction Processing，OLTP）的基准程序，TPC-D 是决策支持的基准程序。TPC-E 是大型企业信息服务的基准程序。

1.3 课后演练

- 计算机中 CPU 对其访问速度最快的是__（1）__。

  （1）A．内存　　B．Cache　　C．通用寄存器　　D．硬盘

- 机器字长为 n 位的二进制数可以用补码来表示__（2）__个不同的有符号定点小数。

  （2）A．2^n　　B．2^{n-1}　　C．2^n-1　　D．$2^{n-1}+1$

- Cache 的地址映像方式中，发生块冲突次数最少的是__（3）__。

  （3）A．全相联映像　　B．组相联映像　　C．直接映像　　D．无法确定的

- 计算机中 CPU 的中断响应时间指的是__（4）__的时间。

  （4）A．从发出中断请求到中断处理结束

  B．从中断处理开始到中断处理结束

  C．CPU 分析判断中断请求

  D．从发出中断请求到开始进入中断处理程序

- 总线宽度为 32bit，时钟频率为 200MHz，若总线上每 5 个时钟周期传送一个 32bit 的字，则该总线的宽度为__（5）__MB/s。

  （5）A．40　　B．80　　C．160　　D．200

- 以下关于指令流水线性能度量的叙述中，错误的是__（6）__。

  （6）A．最大吞吐率取决于流水线中最慢一段所需的时间

  B．如果流水线出现断流，加速比会明显下降

  C．要使加速比和效率最大化应该对流水线各级采用相同的运行时间

  D．流水线采用异步控制会明显提高其性能

- CPU 是在__（7）__结束时响应 DMA 请求的。

  （7）A．一条指令执行　　B．一段程序

  C．一个时钟周期　　D．一个总线周期

- 虚拟存储体系由__（8）__两级存储器构成。

  （8）A．主存-辅存　　B．寄存器-Cache

  C．寄存器-主存　　D．Cache-主存

- 浮点数能够表示的数的范围是由其__（9）__的位数决定的。

  （9）A．尾数　　B．阶码　　C．数符　　D．阶符

- 在机器指令的地址字段中，直接指出操作数本身的寻址方式称为__（10）__。

（10）A．隐含寻址　　B．寄存器寻址　　C．立即寻址　　D．直接寻址

- 内存按字节编址从 B3000H 到 DABFFH 的区域其存储容量为__（11）__。

（11）A．123KB　　B．159KB　　C．163KB　　D．194KB

- CISC 是__（12）__的简称。

（12）A．复杂指令系统计算机　　B．超大规模集成电路

C．精简指令系统计算机　　D．超长指令字

1.4　课后演练答案解析

（1）**答案**：C

解析：访问速度：通用寄存器>Cache>内存>硬盘。

（2）**答案**：A

解析：n 位二进制数可以表示 2^n 个数值，例如取 n=2 验证，二位二进制数可表示四个数值（00,01,10,11），同理，n 位二进制数可表示 2^n 个数值。将这个通用结论和编码结合起来，因为在补码里，0 只有一种表示方式，因此可以表示全 2^n 个数值。但是要注意的是，原码、反码的可以表示的是 2^n–1 个数值，因为原码、反码里 0 分正 0 和负 0。

（3）**答案**：A

解析：当每一个 Cache 块和主存块的映射范围越大时，越不容易发生冲突，因为可以任意放置在非空块上，因此是全相联映像，只有当 Cache 全满了才会发生冲突。

（4）**答案**：D

解析：注意是响应时间，不包括处理时间，所谓响应就是 CPU 做出反应的时间，即从发出中断请求到开始进入中断处理程序，接下来是中断处理时间。

（5）**答案**：C

解析：时钟周期=时钟频率的倒数，因此一个时钟周期为 1/200M 秒，5 个时钟周期就是 5/200M 秒，传送 32bit，那么总线宽度就是用传输数据除以时间，即 32bit/(5/200M)=160MB/s。

（6）**答案**：D

解析：吞吐率就是单位时间内执行的指令条数，受短板原理影响，在多段流水线执行过程中，由执行最慢的一段决定整个流水线执行时间；流水线出现断流，意味着无法重叠，消耗时间增大，无法加速；流水线最理想的就是各级运行时间相同，这样可以保证无延迟；流水线是一种同步控制机制。

（7）**答案**：D

解析：在一个总线周期结束后，CPU 会响应 DMA 请求开始读取数据；CPU 响应程序中断方式请求是在一条指令执行结束时；区分指令执行结束和总线周期结束。

（8）**答案**：A

解析：注意考查的是虚拟存储体系，是主存-辅存中存在虚拟存储映像，Cache-主存是真实存储。

（9）**答案**：B

解析：浮点数的范围由阶码决定，精度由尾数决定。

（10）**答案**：C

解析：指令地址字段，操作数地址存储的就是操作数本身，是立即寻址。

（11）**答案**：B

解析：这段区域共有 DABFFH–B3000H+1=27C00H 个存储空间，内存按字节编址，即一个存储空间为 1 个字节，也即存储容量共 27C00H 个字节，转化为十进制为 159KB。

（12）**答案**：A

解析：CISC 的首字母 C 是 complex，肯定是复杂指令系统。