计算机系统原理
（13015 适用全国）
速 记 宝 典

命题来源： 围绕学科的基本概念、原理、特点、内容。

答题攻略：

（1）不能像名词解释那样简单，也不能像论述题那样长篇大论，但需要加以简要扩展。

（2）答案内容要简明、概括、准确，即得分的关键内容一定要写清楚。

（3）答案表述要有层次性，列出要点，分点分条作答，不要写成一段；

（4）如果对于考题内容完全不知道，利用选择题找灵感，找到相近的内容，联系起来进行作答。如果没有，随意发挥，不放弃。

考点 1：简述冯·诺依曼结构计算机的基本思想

1）采用“存储程序”工作方式。

2）计算机由运算器、控制器、存储器、输入设备和输出设备五大基本部件组成。

3）存储器能存放数据，也能存放指令，在形式上没有区别，但计算机应能区分它们

4）计算机内部以二进制形式表示指令和数据；

考点 2：简述程序和指令的执行过程

（1）取数指令（load）从主存单元中取出数据存放到通用寄存器中；

（2）存数指令（store）将通用寄存器的内容写入主存单元；

（3）加法指令（add）将两个通用寄存器内容相加后送入结果寄存器，

（4）传送指令（mov）将一个通用寄存器的内容送到另一个通用寄存器，如此等等。

考点 3：简述翻译程序的分类

1）汇编程序（Assembler）：也称汇编器。用于将汇编语言源程序翻译成机器语言目标程序。

2）解释程序（Interpreter）：也称解释器。用于将源程序中的语句按其执行顺序逐条翻译成机器指令并立即执行，如 PYTHON，BASIC 等；

3）编译程序（Compiler）：也称编译器。用于将高级语言源程序翻译成汇编语言或机器语言目标程序。

考点 4：简述从源程序到可执行文件的转换过程。

1）预处理阶段

2）编译阶段

3）汇编阶段

4）链接阶段

考点 5：简述计算机的用户分类

（1）最终用户

（2）系统管理员

（3）应用程序员

（4）系统程序员

考点 6：简述定点数编码表示方法

（1）原码

（2）补码

（3）反码

（4）移码

考点 7：简述补码的突出优点

（1）与原码和反码相比，数 0 的补码表示形式唯一。

（2）与原码和移码相比，补码运算系统是一种模运算系统，因而可用加法实现减法运算，且符号位可以和数值位一起参加运算。

（3）与原码和反码相比，它比原码和反码多表示一个最小负数。

（4）与反码相比，不需要通过循环进位来调整结果。

考点 8：简述无符号数除法运算的判断操作

（1）若被除数为 0、除数不为 0，或者定点整数除法时|被除数|<|除数|，则说明商为 0，余数为被除数，不再继续执行。

（2）若被除数不为 0、除数为 0，对于整数，则发生“除数为 0”异常；对于浮点数，则结果为无穷大。

（3）若被除数和除数都为 0，对于整数，则发生除法错异常；对于浮点数，则有些机器产生一个不发信号的 NaN，即“quiet NaN”。

考点 9：简述浮点数乘法的运算步骤

（1）尾数相乘、阶相加

（2）尾数规格化

（3）尾数舍入处理

（4）阶码溢出判断

考点 10：简述指令的类型

（1） RR 型（两个操作数都来自寄存器）

（2） RS 型（两个操作数分别来自寄存器和存储单元）

（3） SI 型（两个操作数分别来自存储单元和立即数）

（4） SS 型（两个操作数都来自存储单元）等。

考点 11：简述与 CISC 相比、RISC 指令系统的主要特点

①指令数目少；

②指令格式规整，采用定长指令字方式，操作码和操作数地址等字段的长度固定；

③只有 Load/Store 指令中的数据需要访存、这种称为 Load/Store 型指令风格；

④采用大量通用寄存器。

考点 12：简述生成机器代码的过程

①预处理

②编译

③汇编

④链接

考点 13：简述 C 语言程序中的基本数据类型

1）指针或地址

2）序数、位串

3）带符号整数

4）浮点数

考点 14：简述 IA-32 指令的操作数类型

（1）立即数

（2）寄存器操作数

（3）存储器操作数

考点 15：简述 IA-32 中的通用寄存器

（1）8 个 8/16/32 位定点通用寄存器

（2） 8 个 MMX 指令/x87 FPU 使用的 64 位/80 位寄存器 MMO/ST（0）～MM7/ST（7）

（3）8 个 SSE 指令使用的 128 位寄存器 XMMO～XMM7

考点 16：简述二进制数运算指令的类型

1.加/减运算指令

2.增/减运算指令

3.取负指令

4.比较指令

5.乘/除运算指令

考点 17：简述过程调用的执行步骤

假定过程 P 调用过程 Q，则 P 称为调用者（Caller），Q 称为被调用者（Callee）。

过程调用的执行步骤如下。

1）P 将入口参数（实参）放到 Q 能访问到的地方。

2）P 将返回地址存到特定的地方，然后将控制转移到 Q。

3）Q 保存 P 的现场，并为自己的非静态局部变量分配空间。

4）执行 Q 的过程体（函数体）。

5）Q 恢复 P 的现场，并释放局部变量所占空间。

6）Q 取出返回地址，将控制转移到 P。

考点 18：简述 C 语言中循环结构

（1）for 语句

（2）while 语句

（3）do～while 语句

考点 19：简述 i386 System V ABI 对 struct 结构体数据的对齐方式规则

①整个结构体变量的对齐方式与其中对齐方式最严格的成员相同；

②每个成员在满足其对齐方式的前提下，取地址最小的可用位置作为成员在结构体

中的偏移量，这可能导致内部插空；

③结构体大小应为对齐边界长度的整数倍。这可能会导致尾部插空。

考点 20：简述与 IA-32 代码相比，x86-64 代码主要特点

1）比 IA-32 具有更多的通用寄存器个数。

2）比 IA-32 具有更长的通用寄存器位数，从 32 位扩展到 64 位。

3）字长从 32 位变为 64 位，因而逻辑地址从 32 位变为 64 位。

4）对于 long double 型数据，虽然还是采用与 IA-32 相同的 80 位扩展精度格式，但是，所分配的存储空间从 IA-32 的 12 字节大小扩展为 16 字节大小。

5）过程调用时，对于整型入口参数只有 6 个以内的情况，用通用寄存器而不是用栈来传递。

6）128 位的 XMM 寄存器从原来的 8 个增加到 16 个，浮点操作采用基于 SSE 的面向 XMM 寄存器的指令集，浮点数存放在 128 位的 XMM 寄存器中。

考点 21：简述 x86-64 的寄存器的使用约定

①可以不用帧指针寄存器 RBP 作为栈帧底部，此时，使用 RSP 作为基址寄存器来访问栈帧中的信息，而 RBP 可作为普通寄存器使用；

②传送入口参数的寄存器依次为 RDI、RSI、RDX、RCX、R8 和 R9，返回参数存放在 RAX 中；

③调用者保存的寄存器为 R10 和 R11，被调用者保存的寄存器为 RBX、RBP、R12、R13、R14 和 R15；

④RSP 用于指向栈顶元素；

⑤RIP 用于指向正在执行或即将执行的指令。

考点 22：简述四趟扫描处理

第一趟扫描进行词法分析；

第二趟扫描进行语法分析；

第三趟扫描进行代码优化和存储分配；

第四趟扫描生成代码。

考点 23：简述与 ELF 可重定位文件格式相比，ELF 可执行文件的不同点

1）ELF 头中字段 e_entry 给出程序执行入口地址，可重定位文件中此字段为 0。

2）通常会有.init 节和.fini 节，其中，init 节定义一个 init 函数，用于可执行文件开始执行时的初始化工作。

3）少了.rel，text 和.rel，data 等重定位信息节。

4）多了一个程序头表，也称段头表（Segment Header Table），它是一个结构数组。

考点 24：简述符号表中的符号类型

1）在 m 中定义并被其他模块引用的全局符号（Global Symbol）。这类符号包括非静态的函数名和全局变量名。

2）由其他模块定义并被 m 引用的全局符号，称为 m 的外部符号（External Symbol），包括在 m 中引用的在其他模块定义的外部函数名和外部变量名。

3）在 m 中定义并在 m 中引用的本地符号（Local Symbol）。

考点 25：简述通过 shell 命令行输入可执行文件名 a.out 进行程序加载的过程

1）shell 命令行解释器输出一个命令行提示符（如：unix>），并开始接受用户输入的命令行。

2）当用户在命令行提示符后输入命令行“./a，out[enter]”后，shell 命令行程序开始对命令行进行解析，获得各个命令行参数并构造传递给函数 execve 的参数列表 argv 和参数个数 argc。

3）调用 fork 函数，创建一个子进程。

4）以第 2）步命令行解析得到的参数个数 argc、参数列表 argv 以及全局变量 environ 作为参数，调用函数 execve，从而实现在当前进程（用 fork 新创建的子进程）的上下文中加载并运行 a.out 程序。

考点 26：简述 CPU 执行指令的过程。

（1）取指令

（2）指令译码

（3）计算源操作数地址并取操作数

（4）执行数据操作

（5）计算目的操作数地址并存结果、

（6）计算下条指令地址

考点 27：简述 CPU 的基本功能和组成。

1）程序计数器（PC）。

2）指令寄存器（IR）。IR 用以存放现行指令。

3）指令译码器（ID）。

4）启停控制逻辑。

5）时序信号产生部件。

6）操作控制信号形成部件。

7）总线控制逻辑

8）中断机构。实现对异常情况和外部中断请求的处理。

考点 28：简述 CPU 对异常和中断的响应过程步骤

1. 保护断点和程序状态

2. 关中断

3. 识别异常和中断事件并转相应处理程序

考点 29：简述指令的处理过程阶段

1）取指令并 PC 加 1（IF）：根据 PC 的值从存储器取出指令，并 PC←PC+1。

2）译码并读寄存器（ID）：对指令操作码进行译码并生成控制信号，同时读取寄存器 rs 和 rt 的内容。

3）运算或读存储器（EX）：在 ALU 中对寄存器操作数进行运算，或者根据 addr 读存储器

4）结果写回（WB）：将结果写入目的寄存器 rt，或写人主存单元 addr 中。

考点 30：简述磁盘读/写的操作步骤

1）寻道操作

2）旋转等待操作

3）读/写操作

考点 31：简述磁盘存储器的性能指标

（1）记录密度

（2）存储容量

（3）数据传输率

（4）平均存取时间等

考点 32：简述闪存的基本操作

（1）编程（充电）

（2）擦除（放电）

（3）读取

考点 33：简述 SSD 的三个限制

①写某一页信息之前，必须先擦除该页所在的整个区块；

②擦除后区块内的页必须按顺序写入信息；

③擦除/编程次数有限。

考点 34：简述主存块和 cache 行之间的映射方式。

（1）直接映射：每个主存块映射到 cache 的固定行中。

（2）全相联映射：每个主存块映射到 cache 的任意行中。

（3）组相联映射：每个主存块映射到 cache 的固定组的任意行中。

考点 35：简述用户空间映射到用户进程的区域

1）用户栈（User Stack）。

2）共享库（Shared Libraries）。

3）堆（Heap）。

4）可读/写数据区。

5）只读数据和代码区。

考点 36：简述虚拟存储器的类型

1. 段式虚拟存储器
2. 页式虚拟存储器
3. 段页式虚拟存储器

考点 37：简述指令集架构和硬件的基本功能。

1）使部分 CPU 状态只能由操作系统内核程序访问而用户进程只能读不能写，或者根本不能访问。

2）支持至少两种特权模式。

3）提供在不同特权模式之间相互切换的机制。

考点 38：简述使用标准 I/O 库函数的不足

①I/0 为同步操作，即程序必须等待 I/O 操作真正完成后才能继续执行；

②在一些情况下不适合甚至无法使用标准 I/O 库函数实现 I/O 功能，如 C 标准 VO 库中不提供读取文件元数据的函数；

③标准 I/O 库函数还存在一些问题，用它进行网络编程容易造成缓冲区溢出等风险，同时它也不提供对文件进行加锁和解锁等功能。

考点 39：简述 I/O 的主要控制方式

1. 程序直接控制
2. 中断控制
3. DMA 控制

考点 40：简述 I/O 接口的主要职能

1）数据缓冲。

2）错误和就绪检测。

3）控制和定时。

4）数据格式的转换。

考点 41：简述中断系统的基本功能

①及时记录各种中断请求，通常用一个中断请求寄存器来记录。

②自动响应中断请求。CPU 在“开中断”状态下，执行一条指令后会自动检测中断请求引脚，发现有中断请求后会自动响应中断。

③同时有多个中断请求时，能自动选择并响应优先级最高的中断请求。

④保护被打断程序的断点和现场。断点指被打断程序中将要执行的下一条指令的地址，由 CPU 保存，现场指被打断程序在断点处各通用寄存器的内容，由中断服务程序保存。

⑤通过中断屏蔽实现多重中断的嵌套执行。