2025年

计算机组成原理考研复习指导

王道论坛 组 编

王道官方版本，仅限内部使用!

支持正版，严禁传播!

|  |  |  |
| --- | --- | --- |
| 扫 码 加 入 读 者Q 群 | **電** **子** **工** **業** **出** **钣** **社.** | 扫 码 加 入 读 者Q 群 |

Publishing House of Electronics Industry

北京·BEIJING

**内** **容** **简** **介**

本书是计算机专业硕士研究生入学考试“计算机组成原理”课程的复习用书，内容包括计算机系统概 述、数据的表示和运算、存储系统、指令系统、中央处理器、总线、输入/输出系统等。全书严格按照最 新计算机考研大纲计算机组成原理部分的要求，对大纲所涉及的知识点进行集中梳理，力求内容精练、重 点突出、深入浅出。本书精选各名校的历年考研真题，给出详细的解题思路，力求实现讲练结合、灵活掌 握、举一反三的功效。

本书可作为考生参加计算机专业硕士研究生入学考试的复习用书，也可作为计算机专业学生学习计算机 组成原理课程的辅导用书。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

**图书在版编目(CIP)** **数据**

2025年计算机组成原理考研复习指导/王道论坛组编.一北京：电子工业出版社，2024.1

ISBN 978-7-121-46735-6

I.①2… Ⅱ.① 王 … Ⅲ.①计算机组成原理一研究生一入学考试一自学参考资料 IV.①TP301

中国国家版本馆 CIP 数据核字(2023)第223519号

责任编辑：谭海平

印 刷：山东华立印务有限公司

装 订： 山东华立印务有限公司

出版发行：电子工业出版社

北京市海淀区万寿路173信箱

邮编：100036

开 本： 787×1092 1/16 印张：21.75 字数：612.5千字

版 次 ：2024年1月第1版

印 次： 2024年1 月 第 1 次印刷

定 价：71.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联 系及邮购电话：(010)88254888,88258888。

质量投诉请发邮件至zlts@phei com cn,盗版侵权举报请发邮件至dbqq@phei com.cn。 本书咨询联系方式：(010)88254552,tan02@phei com.cn。



本 书 附 赠 资

源 兑 换 方 法



扫码关注

“王道在线”

点击公众号左下角的菜单进入

资源兑换中心。



2

兑换中心

×

点击公众号菜单

兑换码 邀请码

“兑换中心”

—

**输入兑换码**

例： H7xWBrYt

或扫码加客服微信获取

立即兑换

兑换记录>

 **配套资源内容**

**25数据结构配套源**

**…◎**

a

工通计算机教育

睡

tasagn

主质

熙

收题

m 

期哪 相

放生解

U4.H

24H

**【关于兑换配套课件的说明】**

2 4 . 选 样

1.凭兑换码兑换相应课程部分选 择 题 的 视 频 以 及B站 免 费 课 程 的 课件 。

88

2.B站免费课程不是2025版付费 课程，但差别不大，详情请咨 询客服 。

3. 兑换码贴于封面右下角，刮开涂

1.盗版书无兑换码请勿购买。 2.免费视频非王道最新网课。 3.免费视频不包含答疑服务。

层可见。

4. 兑换截止时间为2024年12月31日。

**前** **言**

由王道论坛 (cskaoyan.com) 组织名校状元级选手编写的“王道考研系列”辅导书，不仅参 考了国内外的优秀教辅资料，而且结合了高分选手的独特复习经验，包括对考点的讲解以及对习

题的选择和解析。“王道考研系列”单科辅导书共有如下四本：

● 《2025年数据结构考研复习指导》

● 《2025年计算机组成原理考研复习指导》

· 《2025年操作系统考研复习指导》

● 《2025年计算机网络考研复习指导》

我们还围绕这套书开发了一系列赢得众多读者好评的计算机考研课程，包括考点精讲、习题 详解、暑期强化训练营、冲刺串讲、伴学督学和全程答疑服务等，读者可扫描封底的二维码加客 服微信咨询。对于基础较为薄弱的读者，相信这些课程和服务能助你一臂之力。此外，我们在B 站免费公开了本书配套的基础课程，读者可凭兑换码获取课程的课件及部分选择题的讲解视频。

基础课程升华了单科辅导书中的考点讲解，强烈建议读者结合使用。

在冲刺阶段，我们还将出版如下两本冲刺用书：

● 《2025年计算机专业基础综合考试冲刺模拟题》

● 《2025年计算机专业基础综合考试历年真题解析》

深入掌握专业课的内容没有捷径，考生也不应抱有任何侥幸心理。只有扎实打好基础，踏实 做题巩固，最后灵活致用，才能在考研时取得高分。我们希望本套辅导书能够指导读者复习，但 是学习仍然要靠自己，高分无法建立在空中楼阁之上。对想要继续在计算机领域深造的读者来说， 认真学习和扎实掌握计算机专业的四门重要专业课是最基本的前提。

“王道考研系列”是计算机考研学子口碑相传的辅导书，自2011版首次推出以来，始终占据 同类书销量的榜首，这就是口碑的力量。有这么多学长的成功经验，相信只要读者合理地利用辅 导书，并且采用科学的复习方法，就一定能够收获属于自己的那份回报。

“不包就业、不包推荐，培养有态度的程序员。”王道训练营是王道团队打造的线下魔鬼式编 程训练营。打下编程功底、增强项目经验，彻底转行入行，不再迷茫，期待有梦想的你!

参与本书编写工作的人员主要有赵霖、罗乐、徐秀瑛、张鸿林、赵淑芬、赵淑芳、罗庆学、 赵晓宇、喻云珍、余勇、刘政学等。予人玫瑰，手有余香，王道论坛伴你一路同行!

对本书的任何建议，或发现有错误，欢迎扫码联系我们，以便及时改正优化。



风华漫舞

**致** **读** **者**

——关于王道单科辅导书使用方法的道友建议

我是“二战考生”,2012年第 一 次考研成绩333分(专业代码408,成绩81 分),痛定思痛 后决心再战。潜心复习了半年后终于以392分(专业代码408,成绩124分)考入上海交通大学 计算机系，这半年里我的专业课成绩提高了43分，成了提分主力。从未达到录取线到考出比较 满意的成绩，从蒙头乱撞到有了自己明确的复习思路，我想这也是为什么风华哥从诸多高分选手 中选择我给大家介绍经验的一个原因吧。

整个专业课的复习是围绕王道辅导书展开的，从一遍、两遍、三遍看单科辅导书的积累提升， 到做8套模拟题时的强化巩固，再到看思路分析时的醍醐灌顶。王道辅导书能两次押中算法原题 固然有运气成分，但这也从侧面说明编者的编写思路和选题方向与真题很接近。

下面说一说我的具体复习过程。

每天划给专业课的时间是3～4小时。第一遍仔细看课本，看完一 章做一 章单科辅导书上的 习题(红笔标注错题),这 一 遍共持续2个月。第二遍主攻单科辅导书(红笔标注重难点),辅看 课本。第二遍看单科辅导书和课本的速度快了很多，但感觉收获更多，常有温故知新的感觉，理 解更深刻。(风华注：建议这里再速看第三遍，特别针对错题和重难点。模拟题做完后再跳看第 四遍。)

以上是打基础阶段，注意：单科辅导书和课本我仔细精读了两遍，以便尽量弄懂每个知识点 和习题。大概11 月上旬开始做模拟题和思路分析，期间遇到不熟悉的地方不断回头查阅单科辅 导书和课本。8 套模拟题的考点覆盖得很全面，所以大家做题时如果忘记了某个知识点，千万 不要慌张，赶紧回去看这个知识点，最后的模拟就是查漏补缺。模拟题一 定要严格按考试时间 (3 小时)去做，注意应试技巧，做完试题后再回头研究错题。算法题的最优解法不太好想，如 果实在没思路，建议直接“暴力”解决，结果正确也能有10分，总比苦拼出15分来而将后面比 较好拿分的题耽误了好(这是我第一次考研的切身教训)。最后剩了几天看标注的错题，第三遍 跳看单科辅导书，考前一夜浏览完网络，踏实地睡着了 … …

考完专业课，走出考场终于长舒一口气，考试情况也心中有数。回想这半年的复习，耐住了 寂寞和诱惑，雨雪风霜从未间断地跑去自习，考研这人生一站终归没有辜负我的良苦用心。佛教 徒说世间万物生来平等，都要落入春华秋实的代谢中去；辩证唯物主义认为事物作为过程存在，

凡是存在的终归要结束。你不去为活得多姿多彩而拼搏，真到了和青春说再见时，你是否会可惜 虚枉了青春?风华哥说过，我们都是有梦想的青年，我们正在逆袭，你呢?

感谢风华哥的信任，给我这个机会为大家分享专业课复习经验，作为一个铁杆道友在王道受 益匪浅，也借此机会回报王道论坛。祝大家金榜题名!

ccg1990@SJTU

**王道训练营**

王道是道友们考研路上值得信赖的好伙伴，十多年来陪伴了数百万计算机考研人，不离不 弃。王道尊重的不是考研这个行当，而是考研学生的精神和梦想。考研可能是同学们实现梦想 的起点，但专业功底和学习能力更是受用终生的资本，它决定了未来在技术道路上能走多远。 从考研图书，到辅导课程，再到编程培训，王道只专注于计算机考研及编程领域。

计算机专业是一个靠实力吃饭的专业。王道团队中很多人的经历或许和现在的你们相似，也 经历过本科时的迷茫，无非是自知能力太弱，以致底气不足。学历只是敲门砖，同样是名校硕 士，有人如鱼得水，最终成为“Offer 帝”,有人却始终难入“编程与算法之门”,再次体会迷茫的 痛苦。我们坚信一个写不出合格代码的计算机专业学生，即便考上了研究生，也只是给未来的失 业判了个“缓期执行”。我们也希望所做的事情能帮助同学们少走弯路。

考研结束后的日子，或许是一段难得的提升编程能力的连续时光，趁着还有时间，应该去 弥补本科期间应掌握的能力，缩小与“科班大佬们”的差距。

把参加王道训练营视为一次对自己的投资，投资自身和未来才是最好的投资。

**王** **道** **训** **练** **营** **简** **介**

**1.** **面向就业**

希望转行就业，但编程能力偏弱的学生。

考研并不是人生的唯一 出路，努力拼搏奋斗的经历总是难忘的，但不论结果如何，都不应 有太大的遗憾。不少考研路上的“失败者”在王道都到达了自己在技术发展上的新里程碑，我 们相信一个肯持续努力、积极上进的学生一定会找到自己正确的人生方向。

再不抓住当下，未来或将持续迷茫，逝去了的青春不复返。在充分竞争的技术领域，当前 的能力决定了你能找一份怎样的工作，踏实的态度和学习的能力决定了你未来能走多远。

王道训练营致力于给有梦想、肯拼搏、敢奋斗的道友提供最好的平台!

**2.** **面向硕士**

希望提升能力，刚考上计算机相关专业的准硕士。

考研逐年火爆，能考上名校确实是重要的转折，但硕士文凭早已不再稀缺。考研高分并不等 于高薪 Offer, 学历也不能保证你拿到好Offer, 名校的光环能让你获得更多面试机会，但真正要拿 到好 Offer, 比拼的是实力。同为名校硕士， Offer 的成色可能千差万别，有人轻松拿到腾讯、阿 里、抖音、百度等优秀公司的 Offer, 有人面试却屡屡碰壁，最后只能“将就”签约。

人生中关键性的转折点不多，但往往能对自己的未来产生深远的影响，甚至决定了你未来

的走向，高考、选专业、考研、找工作都是如此，把握住关键转折点需要眼光和努力。

**3.** **报名要求**

●具有本科学历，愿意通过奋斗去把握自己的人生，愿意重回高三冲刺式的学习状态。

●完成开课前的作业，用作业考察态度，合格者才能获得最终的参加资格，宁缺毋滥!对于 意志不够坚定的同学而言，这些作业也算是设置的一道门槛，决定了是否有参加的资格。

作业完成情况是最重要的考核标准，我们不会歧视跨度大的同学，坚定转行的同学往往会 更努力。跨度大、学校弱这些是无法改变的标签，唯一可以改变的就是通过持续努力来提升自 身的技能，而通过高强度的短期训练是完全有可能逆袭的，太多的往期学员已有过证明。

**4.** **学习成效**

迅速提升编程能力，结合项目实战，逐步打下坚实的编程基础，培养积极、主动的学习能 力。以动手编程为驱动的教学模式，解决你在编程、思维上的不足，也为未来的深入学习提供 方向指导，掌握学习编程的方法，引导进入“编程与算法之门”。

道友们在训练营里从“菜鸟”逐步成长，训练营中不少往期准硕士学员后来陆续拿到了阿 里、腾讯、抖音、百度、美团、小米等一线互联网大厂的 Offer 。 这就是竞争力!

**王道训练营优势**

这里都是道友，他们信任王道，乐于分享与交流，氛围好而纯粹。

一起经历过考研训练的生活、学习，大家很快会成为互帮互助的好战友，相互学习、共同 进步，在转行的道路上，这就是最好的圈子。正如某期学员所言：“来了你就发现，这里无关程 序员以外的任何东西，这是一个过程， 一个对自己认真、对自己负责的过程。”

考研绝非人生的唯一 出路，给自己换一条路走，去职场上好好发展或许会更好。即便考上 研究生也不意味着高枕无忧，人生的道路还很漫长。

王道团队成员皆具有扎实的编程功底，他们用自己的技术和态度去影响训练营的学员，尽 可能指导学员走上正确的发展道路是对道友信任的回报，也是一种责任!

王道训练营是一个平台，网罗王道论坛上有梦想、有态度的青年，并为他们的梦想提供土 壤和圈子。王道始终相信“物竞天择，适者生存”,这里的生存不是指简简单单地活着，而是指 活得有价值、活得有态度!

**王道训练营课程**

王道训练营开设4种班型：

·Linux C 和 C++ 短期班(40～45天，初试后开课，复试冲刺)

·Java EE方向(4个月，武汉校区)

·Linux C/C++方向(4个月，武汉校区)

·Python 大数据方向(3个半月，直播授课或深圳校区)

短期班的作用是在初试后及春节期间，快速提升学员的编程水平和项目经验，给复试、面试加 成。其他3科班型既面向有就业需求的学员，又适合想提升能力或打算继续考研的学员。

要想了解王道训练营，可以关注王道论坛“王道训练营”版面，或者扫码加老师微信。



**目** **录**

**第** **1** **章** **计算机系统概述** [1](#bookmark2)

\*1.1 计算机发展历程 [1](#bookmark3)

\*1.1.1 计算机硬件的发展 [1](#bookmark4)

\*1.1.2 计算机软件的发展 [2](#bookmark5)

1.2 计算机系统层次结构 [2](#bookmark6)

1.2.1 计算机系统的组成 [2](#bookmark7)

1.2.2 计算机硬件 [3](#bookmark8)

1.2.3 计算机软件 [5](#bookmark9)

1.2.4 计算机系统的层次结构 [6](#bookmark10)

1.2.5 计算机系统的工作原理 [7](#bookmark11)

1.2.6 本节习题精选 [8](#bookmark1)

1.2.7 答案与解析 [10](#bookmark12)

1.3 计算机的性能指标 [12](#bookmark13)

1.3.1 计算机的主要性能指标 [12](#bookmark14)

1.3.2 几个专业术语 [15](#bookmark15)

1.3.3 本节习题精选 [15](#bookmark16)

1.3.4 答案与解析 [19](#bookmark17)

1.4 本章小结 [23](#bookmark18)

1.5 常见问题和易混淆知识点 [24](#bookmark19)

**第** **2** **章** **数据的表示和运算** [26](#bookmark20)

2.1 数制与编码 [26](#bookmark21)

2.1.1 进位计数制及其相互转换 [26](#bookmark22)

2.1.2 定点数的编码表示 [29](#bookmark23)

2.1.3 整数的表示 [31](#bookmark24)

2.1.4 C语言中的整数类型及类型转换 [32](#bookmark25)

2.1.5 本节习题精选 [34](#bookmark26)

2.1.6 答案与解析 [36](#bookmark27)

2.2 运算方法和运算电路 [39](#bookmark28)

2.2.1 基本运算部件 [39](#bookmark29)

2.2.2 定点数的移位运算 [41](#bookmark30)

2.2.3 定点数的加减运算 [41](#bookmark31)

2.2.4 定点数的乘除运算 [44](#bookmark32)

2.2.5 本节习题精选 [46](#bookmark33)

2.2.6 答案与解析 [50](#bookmark34)

目 录 IX

2.3 浮点数的表示与运算 [55](#bookmark35)

2.3.1 浮点数的表示 [55](#bookmark36)

2.3.2 浮点数的加减运算 [59](#bookmark37)

2.3.3 C语言中的浮点数类型 [60](#bookmark38)

2.3.4 数据的大小端和对齐存储 [61](#bookmark39)

2.3.5 本节习题精选 [62](#bookmark40)

2.3.6 答案与解析 [69](#bookmark41)

2.4 本章小结 [77](#bookmark42)

2.5 常见问题和易混淆知识点 [78](#bookmark43)

**第** **3** **章** **存** **储** **系** **统** [80](#bookmark44)

3.1 存储器概述 [80](#bookmark45)

3.1.1 存储器的分类 [80](#bookmark46)

3.1.2 存储器的性能指标 [81](#bookmark47)

3.1.3 多级层次的存储系统 [82](#bookmark48)

3.1.4 本节习题精选 [83](#bookmark49)

3.1.5 答案与解析 [84](#bookmark50)

3.2 主存储器 [85](#bookmark51)

3.2.1 SRAM 芯片和 DRAM 芯片 [85](#bookmark52)

3.2.2 只读存储器 [88](#bookmark53)

3.2.3 主存储器的基本组成 [89](#bookmark54)

3.2.4 多模块存储器 [90](#bookmark55)

3.2.5 本节习题精选 [92](#bookmark56)

3.2.6 答案与解析 [96](#bookmark57)

3.3 主存储器与CPU 的连接 [101](#bookmark58)

3.3.1 连接原理 [101](#bookmark59)

3.3.2 主存容量的扩展 [101](#bookmark60)

3.3.3 存储芯片的地址分配和片选 [103](#bookmark61)

3.3.4 存储器与 CPU 的连接 [104](#bookmark62)

3.3.5 本节习题精选 [104](#bookmark63)

3.3.6 答案与解析 [106](#bookmark64)

3.4 外部存储器 [109](#bookmark65)

3.4.1 磁盘存储器 [109](#bookmark66)

3.4.2 固态硬盘 [111](#bookmark67)

3.4.3 本节习题精选 [112](#bookmark68)

3.4.4 答案与解析 [113](#bookmark69)

3.5 高速缓冲存储器 [115](#bookmark70)

3.5.1 程序访问的局部性原理 [115](#bookmark71)

3.5.2 Cache 的基本工作原理 [117](#bookmark72)

3.5.3 Cache 和主存的映射方式 [118](#bookmark73)

3.5.4 Cache 中主存块的替换算法 [122](#bookmark74)

3.5.5 Cache 的一致性问题 [123](#bookmark75)

X -2025 年计算机组成原理考研复习指导

3.5.6 本节习题精选 [124](#bookmark76)

3.5.7 答案与解析 [130](#bookmark77)

3.6 虚拟存储器 [137](#bookmark78)

3.6.1 虚拟存储器的基本概念 [137](#bookmark79)

3.6.2 页式虚拟存储器 [138](#bookmark80)

3.6.3 段式虚拟存储器 [141](#bookmark81)

3.6.4 段页式虚拟存储器 [142](#bookmark82)

3.6.5 虚拟存储器与Cache 的比较 [142](#bookmark83)

3.6.6 本节习题精选 [142](#bookmark84)

3.6.7 答案与解析 [148](#bookmark85)

3.7 本章小结 [153](#bookmark86)

3.8 常见问题和易混淆知识点 [154](#bookmark87)

**第** **4** **章** **指令系统** [155](#bookmark88)

4.1 指令系统 [155](#bookmark89)

4.1.1 指令集体系结构 [155](#bookmark90)

4.1.2 指令的基本格式 [156](#bookmark91)

4.1.3 定长操作码指令格式 [157](#bookmark92)

4.1.4 扩展操作码指令格式 [158](#bookmark93)

4.1.5 指令的操作类型 [158](#bookmark94)

4.1.6 本节习题精选 [159](#bookmark95)

4.1.7 答案与解析 [161](#bookmark96)

4.2 指令的寻址方式 [164](#bookmark97)

4.2.1 指令寻址和数据寻址 [164](#bookmark98)

4.2.2 常见的数据寻址方式 [165](#bookmark99)

4.2.3 本节习题精选 [169](#bookmark100)

4.2.4 答案与解析 [177](#bookmark101)

4.3 程序的机器级代码表示 [183](#bookmark102)

4.3.1 常用汇编指令介绍 [184](#bookmark103)

4.3.2 选择语句的机器级表示 [189](#bookmark104)

4.3.3 循环语句的机器级表示 [190](#bookmark105)

4.3.4 过程调用的机器级表示 [191](#bookmark106)

4.3.5 本节习题精选 [194](#bookmark107)

4.3.6 答案与解析 [198](#bookmark108)

4.4 CISC 和 RISC 的基本概念 [200](#bookmark109)

4.4.1 复杂指令系统计算机(CISC) [201](#bookmark110)

4.4.2 精简指令系统计算机(RISC) [201](#bookmark111)

4.4.3 CISC 和 RISC 的比较 [201](#bookmark112)

4.4.4 本节习题精选 [202](#bookmark113)

4.4.5 答案与解析 [203](#bookmark114)

4.5 本章小结 [203](#bookmark115)

4.6 常见问题和易混淆知识点 [204](#bookmark116)

**第5章** **中央处理器** [205](#bookmark117)

5.1 CPU 的功能和基本结构 [205](#bookmark118)

5.1.1 CPU 的功能 [205](#bookmark119)

5.1.2 CPU 的基本结构 [206](#bookmark120)

5.1.3 CPU 的寄存器 [206](#bookmark121)

5.1.4 本节习题精选 [207](#bookmark122)

5.1.5 答案与解析 [209](#bookmark123)

5.2 指令执行过程 [212](#bookmark124)

5.2.1 指令周期 [212](#bookmark125)

5.2.2 指令周期的数据流 [213](#bookmark126)

5.2.3 指令执行方案 [214](#bookmark127)

5.2.4 本节习题精选 [215](#bookmark128)

5.2.5 答案与解析 [217](#bookmark129)

5.3 数据通路的功能和基本结构 [218](#bookmark130)

5.3.1 数据通路的功能 [218](#bookmark131)

5.3.2 数据通路的组成 [218](#bookmark132)

5.3.3 数据通路的基本结构 [219](#bookmark133)

5.3.4 数据通路的操作举例 [220](#bookmark134)

5.3.5 本节习题精选 [221](#bookmark135)

5.3.6 答案与解析 [228](#bookmark136)

5.4 控制器的功能和工作原理 [234](#bookmark137)

5.4.1 控制器的结构和功能 [234](#bookmark138)

5.4.2 硬布线控制器 [235](#bookmark139)

5.4.3 微程序控制器 [235](#bookmark140)

5.4.4 本节习题精选 [239](#bookmark141)

5.4.5 答案与解析 [243](#bookmark142)

5.5 异常和中断机制 [247](#bookmark143)

5.5.1 异常和中断的基本概念 [247](#bookmark144)

5.5.2 异常和中断的分类 [247](#bookmark145)

5.5.3 异常和中断响应过程 [249](#bookmark146)

5.5.4 本节习题精选 [249](#bookmark147)

5.5.5 答案与解析 [251](#bookmark148)

5.6 指令流水线 [252](#bookmark149)

5.6.1 指令流水线的基本概念 [252](#bookmark150)

5.6.2 流水线的基本实现 [253](#bookmark151)

5.6.3 流水线的冒险与处理 [254](#bookmark152)

5.6.4 流水线的性能指标 [258](#bookmark153)

5.6.5 高级流水线技术 [258](#bookmark154)

5.6.6 本节习题精选 [259](#bookmark155)

5.6.7 答案与解析 [265](#bookmark156)

5.7 多处理器的基本概念 [273](#bookmark157)

XIl -2025 年计算机组成原理考研复习指导

5.7.1 SISD、SIMD 、MIMD 的基本概念 [273](#bookmark158)

5.7.2 硬件多线程的基本概念 [274](#bookmark159)

5.7.3 多核处理器的基本概念 [274](#bookmark160)

5.7.4 共享内存多处理器的基本概念 [275](#bookmark161)

5.7.5 本节习题精选 [276](#bookmark162)

5.7.6 答案与解析 [277](#bookmark163)

5.8 本章小结 [278](#bookmark164)

5.9 常见问题和易混淆知识点 [279](#bookmark165)

**第** **6** **章** **总** **线** [280](#bookmark166)

6.1 总线概述 [280](#bookmark167)

6.1.1 总线基本概念 [280](#bookmark168)

6.1.2 总线的分类 [281](#bookmark169)

6.1.3 系统总线的结构 [282](#bookmark170)

\*6.1.4 常见的总线标准 [282](#bookmark171)

6.1.5 总线的性能指标 [283](#bookmark172)

6.1.6 本节习题精选 [284](#bookmark173)

6.1.7 答案与解析 [287](#bookmark174)

6.2 总线事务和定时 [290](#bookmark175)

6.2.1 总线事务 [290](#bookmark176)

6.2.2 总线定时 [290](#bookmark177)

6.2.3 本节习题精选 [292](#bookmark178)

6.2.4 答案与解析 [294](#bookmark179)

6.3 本章小结 [296](#bookmark180)

6.4 常见问题和易混淆知识点 [296](#bookmark181)

**第7章** **输入/输出系统** [297](#bookmark182)

\*7.1 I/O系统基本概念 [297](#bookmark183)

\*7.1.1 输入/输出系统 [297](#bookmark184)

\*7.1.2 I/O 控制方式 [298](#bookmark185)

\*7.1.3 外部设备 [298](#bookmark186)

\*7.1.4 本节习题精选 [299](#bookmark187)

\*7.1.5 答案与解析 [300](#bookmark188)

7.2 I/O 接口 [300](#bookmark189)

7.2.1 I/O 接口的功能 [300](#bookmark190)

7.2.2 I/O 接口的基本结构 [300](#bookmark191)

7.2.3 I/O接口的类型 [301](#bookmark192)

7.2.4 I/O端口及其编址 [301](#bookmark193)

7.2.5 本节习题精选 [302](#bookmark194)

7.2.6 答案与解析 [304](#bookmark195)

7.3 I/O方式 [305](#bookmark196)

7.3.1 程序查询方式 [305](#bookmark197)

目 录 -XII

7.3.2 程序中断方式 [307](#bookmark198)

222004A 元 式

[311](#bookmark199)

7.3.4 本节习题精选 [315](#bookmark200)

72.5 答 安 与 解 析

[323](#bookmark201)

74 大 高 小

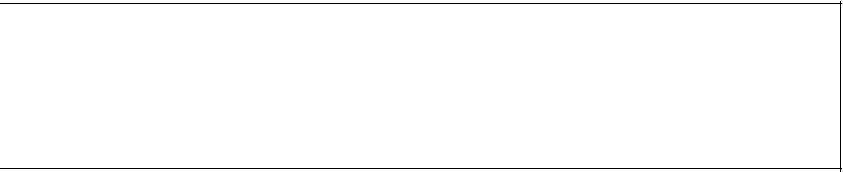
[332](#bookmark202)

7.5 常见问题和易混淆知识点 [332](#bookmark203)

**参考文献** [334](#bookmark204)







第 1 章

计算机系统概述

**【考纲内容】**

(一)计算机系统层次结构

计算机系统的基本组成

计算机硬件的基本组成

计算机软件和硬件的关系

扫一扫



视频讲解

计算机系统的工作原理：“存储程序”方式；高级语言程序与机器语言程序的转换；程 序和指令的执行过程

(二)计算机性能指标

吞吐量；响应时间；CPU 时钟周期；主频； CPI;CPU 执行时间；

MIPS;MFLOPS;GFLOPS;TFLOPS;PFLOPS;EFLOPS;ZFLOPS

**【复习提示】**

本章是组成原理的概述，考查时易针对有关概念或性能指标出选择题，也可能综合后续章节 的内容出有关性能分析的综合题。掌握本章的基本概念，是学好后续章节的基础。部分知识点在 初学时理解不深刻也无须担忧，相信随着后续章节的学习， 一定会有更为深入的理解。

学习本章时，请读者思考以下问题：

1)计算机由哪几部分组成?以哪部分为中心?

2)主频高的 CPU 一定比主频低的 CPU 快吗?为什么?

3)翻译程序、汇编程序、编译程序、解释程序有什么差别?各自的特性是什么?

4)不同级别的语言编写的程序有什么区别?哪种语言编写的程序能被硬件直接执行? 请读者在学习本章的过程中寻找答案，本章末尾会给出参考答案。



**\*1.1** **计算机发展历程**

**\*1.1.1** **计算机硬件的发展**

**1.** **计算机的四代变化**

从1946 年世界上第一台电子数字计算机 (Electronic Numerical Integrator And Computer, ENIAC) 问世以来，计算机的发展已经经历了四代。

1)第一代计算机(1946— 1957 年 ) —— 电子管时代。特点：逻辑元件采用电子管；使用机

① 加“\*”的章节表示已从最新统考大纲中删除，仅供学习参考。

2 -2025年计算机组成原理考研复习指导

器语言进行编程；主存储器用延迟线或磁鼓存储信息，容量极小；体积庞大，成本高；

运算速度较低， 一般只有几千次到几万次每秒。

2)第二代计算机(1958 — 1964年)— —晶体管时代。特点：逻辑元件采用晶体管；运算速 度提高到几万次到几十万次每秒；主存储器使用磁芯存储器；计算机软件也得到了发展， 开始出现了高级语言及其编译程序，有了操作系统的雏形。

3)第三代计算机(1965— 197 1 年 ) ——中小规模集成电路时代。特点：逻辑元件采用中小 规模集成电路；半导体存储器开始取代磁芯存储器；高级语言发展迅速，操作系统也进 一步发展，开始有了分时操作系统。

4)第四代计算机(1972年至今) ——超大规模集成电路时代。特点：逻辑元件采用大规模 集成电路和超大规模集成电路，产生了微处理器；诸如并行、流水线、高速缓存和虚拟 存储器等概念用在了这代计算机中。

**2.** **计算机元件的更新换代**

1)摩尔定律。当价格不变时，集成电路上可容纳的晶体管数目，约每隔18个月便会增加一 倍，性能也将提升一倍。也就是说，我们现在和18个月后花同样的钱买到的CPU, 后者

的性能是前者的两倍。这一定律揭示了信息技术进步的速度。

2)半导体存储器的发展。1970年，美国仙童半导体公司生产出第一个较大容量的半导体存

储器，至今，半导体存储器经历了11代：单芯片1KB 、4KB 、16KB 、64KB 、256KB、 1MB 、4MB 、16MB 、64MB 、256MB 和现在的1GB。

3)微处理器的发展。自1971年Intel 公司开发出第一个微处理器Intel4004 至今，微处理器 经历了 Intel 8008(8 位)、Intel 8086(16位 ) 、Intel 80386(32 位)、Pentium(32 位)、 Pentium I(64位)、Pentium 4(64 位)、Core i7(64 位)等。这里的32位、64位指的 是机器字长，是指计算机进行一次整数运算所能处理的二进制数据的位数。

**\*1.1.2** **计算机软件的发展**

计算机软件技术的蓬勃发展，也为计算机系统的发展做出了很大的贡献。

计算机语言的发展经历了面向机器的机器语言和汇编语言、面向问题的高级语言。其中高级 语言的发展真正促进了软件的发展，它经历了从科学计算和工程计算的 FORTRAN 、结构化程序 设计的PASCAL 到面向对象的C++ 和适应网络环境的 Java。

与此同时，直接影响计算机系统性能提升的各种系统软件也有了长足的发展，特别是操作系 统，如 Windows 、UNIX 、Linux 等。



**1.2** **计算机系统层次结构**

**1.2.1** **计算机系统的组成**

硬件系统和软件系统共同构成了一个完整的计算机系统。硬件是指有形的物理设备，是计算 机系统中实际物理装置的总称。软件是指在硬件上运行的程序和相关的数据及文档。

计算机系统性能的好坏，很大程度上是由软件的效率和作用来表征的，而软件性能的发挥又 离不开硬件的支持。对某一功能来说，若其既可以用软件实现，又可以用硬件实现，则称为软/ 硬件在逻辑功能上是等价的。在设计计算机系统时，要进行软/硬件的功能分配。通常来说， 一个

數据寄存器

存 储 体

地址寄存器

**第1章** **计算机系统概述** 3

功能若使用较为频繁且用硬件实现的成本较为理想，则使用硬件解决可以提高效率。

**1.2.2** **计算机硬件**

**1.** **冯** **·** **诺依曼机基本思想**

▶ **冯·诺依曼计算机的特点(2019)**

冯 · 诺依曼在研究EDVAC 机时提出了“存储程序”的概念，“存储程序”的思想奠定了现代

计算机的基本结构，以此概念为基础的各类计算机统称冯 · 诺依曼机，其特点如下：

1)采用“存储程序”的工作方式。

2)计算机硬件系统由运算器、存储器、控制器、输入设备和输出设备5大部件组成。

3)指令和数据以同等地位存储在存储器中，形式上没有区别，但计算机应能区分它们。 4)指令和数据均用二进制代码表示。

5)指令由操作码和地址码组成，操作码指出操作的类型，地址码指出操作数的地址。

“存储程序”的基本思想是：将事先编制好的程序和原始数据送入主存储器后才能执行， 一 旦程序被启动执行，就无须操作人员的干预，计算机会自动逐条执行指令，直至程序执行结束。

**2.** **计算机的功能部件**

命 题 追 踪▶ **MAR** **和** **MDR** **位数的概念和计算(2010、2011)**

( 1)输入设备

输入设备的主要功能是将程序和数据以机器所能识别和接受的信息形式输入计算机。最常用 也最基本的输入设备是键盘，此外还有鼠标、扫描仪、摄像机等。

(2)输出设备

输出设备的任务是将计算机处理的结果以人们所能接受的形式或其他系统所要求的信息形 式输出。最常用、最基本的输出设备是显示器、打印机。输入/输出设备(简称I/O 设备)是计算 机与外界联系的桥梁，是计算机中不可缺少的重要组成部分。

(3 ) 存储器

存储器分为主存储器(也称内存储器或主存) 和 辅 助 存 储 器 ( 也 称外存储器或外存)。CPU 能够直接访问的存储器是主存储器。辅助存储器用于帮助主存储器记忆更多的信息，辅助存储器 中的信息必须调入主存储器后，才能为CPU 所访问。主存储器的工作方式是按存储单元的地址进 行存取，这种存取方式称为按地址存取方式。

主存储器的最基本组成如图1. 1所示。存储体存放二进制信息， 存储器地址寄存器 (MAR)

存放访存地址，经过地址译码后找到所选的存储单元。存储器数据寄存器 (MDR) 用于暂存要从 存储器中读或写的信息，时序控制逻辑用于产生存储器操作所需的各种时序信号。

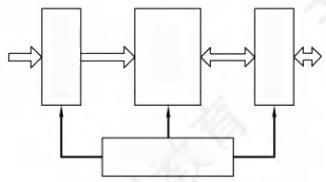
时序控制逻辑

图1.1 主存储器逻辑图

ALU

4 -2025年计算机组成原理考研复习指导

存 储 体 由 许 多 存 储 单 元 组 成 ， 每 个 存 储 单 元 包 含 若 干 存 储 元 件 ， 每 个存 储 元 件存 储 一 位 二 进 制代码“0”或“1”。因此存储单元可存储一串二进制代码，称这串代码为存储字，称这串代码 的位数为存储字长，存储字长可以是1B(8bit) 或是字节的偶数倍。

MAR 用于寻址，其位数反映最多可寻址的存储单元的个数，如 MAR 为 1 0 位，则最多有 2l⁰=1024 个存储单元，记为1K 。MAR 的长度与 PC 的长度相等。

MDR 的 位 数 通 常 等 于 存 储 字 长 ， 一 般 为 字 节 的 2 次 幂 的 整 数 倍 。



**注** **意**

MAR 与 MDR 虽然是存储器的一部分，但在现代计算机中却是存在于CPU 中的；另外， 后文提到的高速缓存 (Cache) 也存在于CPU 中。

( 4 ) 运 算 器

运 算 器 是 计 算 机 的 执 行 部 件 ， 用 于 进 行 算 术 运 算 和 逻 辑 运 算。算 术 运 算是 按 算 术 运 算 规 则 进 行的运算，如加、减、乘、除；逻辑运算包括与、或、非、异或、比较、移位等运算。

运 算 器 的 核 心 是算 术 逻 辑 单 元 (Arithmetic and Logic Unit,ALU)。运 算 器 包 含 若 干 通 用 寄 存

器 ， 用 于 暂 存 操 作 数 和 中 间 结 果 ， 如 累 加 器 (ACC)、 乘 商 寄 存 器 (MQ)、 操 作 数 寄 存 器 (X)、

变址寄存器 (IX) 、 基址寄存器 (BR) 等，其中前三个寄存器是必须具备的。

运 算 器 内 还 有程 序 状 态 寄 存 器 (PSW), 也 称 标 志 寄 存 器 ， 用 于 存 放ALU 运 算 得 到 的 一 些 标 志信息或处理机的状态信息 ， 如结果是否溢出 、 有无产生进位或借位 、 结果是否为负等 。

(5)控制器

控制器是计算机的指挥中心 ， 由其 “ 指挥 ” 各部件自动协调地进行工作 。 控制器由程序计数 器 (PC) 、 指令寄存器 (IR) 和控制单元 (CU) 组成。

PC 用 来 存 放 当 前 欲 执 行 指 令 的 地 址 ， 具 有 自 动 加 1 的 功 能 ( 这 里 的 “ 1 ” 指 一 条 指 令 的 长 度 ) ,

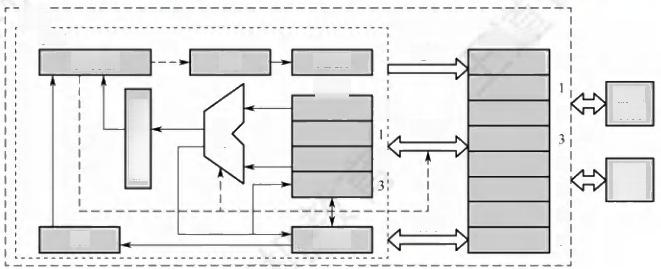
即 可 自 动 形 成 下 一 条 指 令 的 地 址 ， 它 与 主 存 储 器 的 MAR 之 间 有 一 条 直 接 通 路。

IR 用来存放当前的指令，其内容来自主存储器的 MDR 。指令中的操作码 OP(IR) 送至 CU, 用 以 分 析 指 令 并 发 出 各 种 微 操 作 命 令 序 列 ； 而 地 址 码 Ad(IR) 送 往 MAR, 用 以 取 操 作 数 。

一 般 将 运 算 器 和 控 制 器 集 成 到 同 一 个 芯 片 上 ， 称 为中 央 处 理 器 (C PU) 。CPU 和 主 存 储 器 共 同 构 成 主 机 ， 而 除 主 机 外 的 其 他 硬 件 装 置 ( 外 存 、I/O 设 备 等 ) 统 称 外 部 设 备 ， 简 称 外 设 。

图 1 . 2 所 示 为 冯 · 诺 依 曼 结 构 的 模 型 机 。CPU 包 含ALU 、 通 用 寄 存 器 组GPRs 、 标 志 寄 存 器 、 控 制 器 、 指 令 寄 存 器IR、 程 序 计 数 器 PC 、 存 储 器 地 址 寄 存 器 MAR 和 存 储 器 数 据 寄 存 器 MDR。

图 中 从 控 制 器 送 出 的 虚 线 就 是 控 制 信 号 ， 可 以 控 制 如 何 修 改 PC 以 得 到 下 一 条 指 令 的 地 址 ， 可 以 控 制ALU 执 行 什 么 运 算 ， 可 以 控 制 主 存 储 器 是 进 行 读 操 作 还 是 写 操 作 ( 读 / 写 控 制 信 号 ) 。



王机

CPU

控制器

输入 设备

控制

输出 设备

6i

71

MAR

GPRs

标志寄存器

主存储器

4| 5|

地址

数据

MDR

0|

PC

2;`

IR

o:

2

图1 . 2 冯 · 诺依曼结构的模型机

第1章 计算机系统概述 5

CPU 和主存储器之间通过一组总线相连，总线中有地址、控制和数据3组信号线。MAR 中 的地址信息会直接送到地址线上，用于指向读/写操作的主存储器存储单元；控制线中有读/写信 号线，指出数据是从 CPU 写入主存储器还是从主存储器读出到 CPU, 根据是读操作还是写操作 来控制将 MDR 中的数据是直接送到数据线上还是将数据线上的数据接收到 MDR 中 。

**1.2.3** **计算机软件**

**1.** **系统软件和应用软件**

软件按其功能分类，可分为系统软件和应用软件。

系统软件是一组保证计算机系统高效、正确运行的基础软件，通常作为系统资源提供给用户 使用。系统软件主要有操作系统 (OS)、 数据库管理系统 (DBMS )、语言处理程序、分布式软件 系统、网络软件系统、标准库程序、服务性程序等。

应用软件是指用户为解决某个应用领域中的各类问题而编制的程序，如各种科学计算类程 序、工程设计类程序、数据统计与处理程序等。

在本学科范畴内，编写诸如操作系统、编译程序等各种系统软件的人员称为系统程序员；利 用计算机及所支持的系统软件来编写解决具体应用问题的人员称为应用程序员。

**2.** **三个级别的语言**

**命** **题** **追** **踪** ▶ **三种机器语言的特点(2015)**

1)机器语言。也称二进制代码语言，需要编程人员记忆每条指令的二进制编码。机器语言

是计算机唯 一可 以直接识别和执行的语言。

2 ) 汇 编 语 言。汇编语言用英文单词或其缩写代替二进制的指令代码，更容易为人们记忆和 理 解。使用汇编语言编辑的程序，必须经过一个称为汇编程序的系统软件的翻译，将其 转换为机器语言程序后，才能在计算机的硬件系统上执行。

3)高级语言。高 级 语 言 ( 如C 、C++ 、Java 等)是为方便程序设计人员写出解决问题的处理 方案和解题过程的程序。通常高级语言需要经过编译程序编译成汇编语言程序，然后经 过汇编操作得到机器语言程序，或直接由高级语言程序翻译成机器语言程序。

▶ **各种翻译程序的概念(2016)**

因此计算机无法直接理解和执行高级语言程序，所以需要将高级语言程序转换为机器语言程 序，通常把进行这种转换的软件系统称翻译程序。翻译程序有以下三类：

1 ) 汇 编 程 序 (汇编器)。将汇编语言程序翻译成机器语言程序。

2)解释程序(解释器)。将源程序中的语句按执行顺序逐条翻译成机器指令并立即执行。 3)编译程序 (编译器)。将高级语言程序翻译成汇编语言或机器语言程序。

**3.** **软件和硬件的逻辑功能等价性**

硬件实现的往往是最基本的算术和逻辑运算功能，而其他功能大多通过软件的扩充得以实 现。对某一功能来说，既可以由硬件实现，又可以由软件实现，从用户的角度来看，它们在功能 上是等价的。这 一 等价性被称为软/硬件逻辑功能的等价性。例如，浮点数运算既可以用专门的 浮点运算器硬件实现，又可以通过一段子程序实现，这两种方法在功能上完全等效，不同的只是 执行时间的长短而已，显然硬件实现的性能要优于软件实现的性能。

软件和硬件逻辑功能的等价性是计算机系统设计的重要依据，软件和硬件的功能分配及其界 面的确定是计算机系统结构研究的重要内容。当研制一 台计算机时，设计者必须明确分配每一级

6 - 2025年计算机组成原理考研复习指导

的任务，确定哪些功能使用硬件实现，哪些功能使用软件实现。软件和硬件功能界面的划分是由 设计目标、性能价格比、技术水平等综合因素决定的。

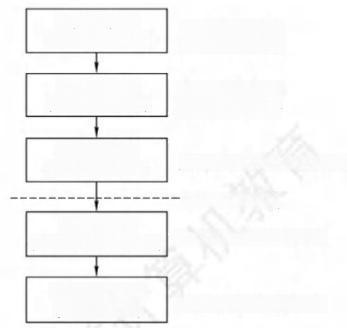
**1.2.4** **计算机系统的层次结构**

计算机是一个硬软件组成的综合体。因为面对的应用范围越来越广，所以必须有复杂的系 统软件和硬件的支持。由于软/硬件的设计者和使用者从不同的角度、用不同的语言来对待同一 个计算机系统，因此他们看到的计算机系统的属性对计算机系统提出的要求也就各不相同。

计算机系统的多级层次结构的作用，就是针对上述情况，根据从各种角度所看到的机器之间的 有机联系，来分清彼此之间的界面，明确各自的功能，以便构成合理、高效的计算机系统。

关于计算机系统层次结构的分层方式，目前尚无统一 的标准，这里采用如图1.3所示的层次

结构。



|  |  |
| --- | --- |
| 虚拟机器M4  (高级语言机器) | 用编译程序翻译  成汇编语言程序 |
| 虚拟机器M3  (汇编语言机器) | 用汇编程序翻译  成机器语言程序 |
| 虚拟机器M2  (操作系统机器) | 用机器语言解释操作系统  软/硬件交界面 |
| 传统机器M1  (用机器语言的机器) | 用微程序解释机器指令 |
| 微程序机器M0 (微指令系统) | 由硬件直接执行微指令 |

图1.3 计算机系统的多级层次结构

第 1 级 是微程序机器层，这是 一个实在的硬件层，它由机器硬件直接执行微指令。

第 2 级 是传统机器语言层，它也是 一个实际的机器层，由微程序解释机器指令系统。

第3级是操作系统层，它由操作系统程序实现。操作系统程序是由机器指令和广义指令组成 的，这些广义指令是为了扩展机器功能而设置的，是由操作系统定义和解释的软件指令，所以这 一 层也称混合层。

第4级是汇编语言层，这一层由汇编程序支持和执行，借此可编写汇编语言源程序。

第5级是高级语言层，它是面向用户的，是为方便用户编写应用程序而设置的。该层由各种 高级语言编译程序支持和执行。在高级语言层之上，还可以有应用程序层，它由解决实际问题的 处理程序组成，如文字处理软件、多媒体处理软件和办公自动化软件等。

没有配备软件的纯硬件系统称裸机。第3层～第5层称为虚拟机，简单来说就是软件实现的 机 器。虚拟机器只对该层的观察者存在，这里的分层和计算机网络的分层类似，对于某层的观察 者来说，只能通过该层的语言来了解和使用计算机，而不必关心下层是如何工作的。

层次之间的关系紧密，下层是上层的基础，上层是下层的扩展。

软件和硬件之间的界面就是指令集体系结构(ISA),ISA 定义了一 台计算机可以执行的所有 指令的集合，每条指令规定了计算机执行什么操作，以及所处理的操作数存放的地址空间和操作 数类型。可以看出， ISA 是指软件能感知到的部分，也称软件可见部分。

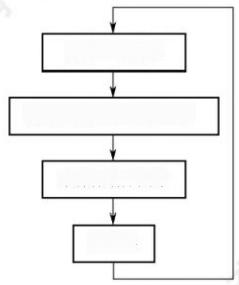
本门课程主要讨论传统机器 M1 和微程序机器 M0 的组成原理及设计思想。

第1章 计算机系统概述 7

**1.2.5** **计算机系统的工作原理**

**1.** **“存储程序”工作方式**

“存储程序”工作方式规定，程序执行前，需要将程序所含的指令和数据送入主存储器， 一 旦程序被启动执行，就无须操作人员的干预，自动逐条完成指令的取出和执行任务。如图1.4所 示， 一个程序的执行就是周而复始地执行一条一条指令的过程。每条指令的执行过程包括：从主 存储器中取指令、对指令进行译码、计算下条指令地址、取操作数并执行、将结果送回存储器。



根据PC取指令

指令译码， PC-PC+“]”

取操作数并执行

送结果

图1.4 程序执行过程

程序执行前，先将程序第一条指令的地址存放到PC 中，取指令时，将 PC 的内容作为地址 访问主存储器。在每条指令执行过程中，都需要计算下条将执行指令的地址，并送至PC 。若 当 前指令为顺序型指令，则下条指令地址为 PC 的内容加上当前指令的长度；若当前指令为转跳 型指令，则下条指令地址为指令中指定的目标地址。当前指令执行完后，根据 PC 的内容到主 存储器中取出的是下一条将要执行的指令，因而计算机能周而复始地自动取出并执行一条一条 的指令。

**2.** **从源程序到可执行文件**

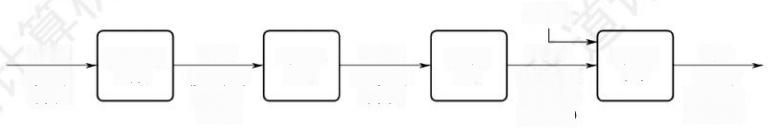
命 题 追 踪▶》 **翻译过程的四个阶段(2022)**

在计算机中编写的C 语言程序，都必须被转换为一系列的低级机器指令，这些指令按照一种

称为可执行目标文件的格式打好包，并以二进制磁盘文件的形式存放起来。

以 UNIX 系统中的 GCC 编译器程序为例，读取源程序文件 hello.c, 并把它翻译成一个可执

行目标文件 hello, 整个翻译过程可分为四个阶段完成，如图1.5所示。



printf.o

hello.0

可重定位 目标程序 (二进制)

hello

可执行 目标程序 (二进制)

hello.i

修改了的 源程序 (文本)

hello.s

汇编程序

(文木)

hello.c

源程序 (文本)

预处理器 (cpp)

编译器 (ccl)

链接器 (ld)

汇编器 (as)

图1.5 源程序转换为可执行文件的过程

1)预处理阶段：预处理器 (cpp) 对源程序中以字符#开头的命令进行处理，例如将#include

命令后面的.h 文件内容插入程序文件。输出结果是一个以i 为扩展名的源文件hello.i。

2)编译阶段：编译器 (ccl) 对预处理后的源程序进行编译，生成一个汇编语言源程序hello.s。

汇编语言源程序中的每条语句都以一种文本格式描述了一条低级机器语言指令。

8 -2025年计算机组成原理考研复习指导

3)汇编阶段：汇编器 (as) 将 hello.s 翻译成机器语言指令，把这些指令打包成一个称为可

重定位目标文件hello.o, 它是一种二进制文件，因此用文本编辑器打开会显示乱码。

4)链接阶段：链接器 (ld) 将多个可重定位目标文件和标准库函数合并为一个可执行目标文

件，简称可执行文件。本例中，链接器将hello.o和标准库函数 printf 所在的可重定位目标 模块printfo 合并，生成可执行文件 hello。最终生成的可执行文件被保存在磁盘上。

**3.** **指令执行过程的描述**

可执行文件代码段是由一条一条机器指令构成的，指令是用0和1表示的一串0/1序列，用 来指示 CPU 完成一个特定的原子操作。例如，取数指令从存储单元中取出一个数据送到CPU 的 寄存器中，存数指令将 CPU 寄存器的内容写入一个存储单元，ALU 指令将两个寄存器的内容进 行某种算术或逻辑运算后送到一个CPU 寄存器中，等等。指令的执行过程在第5章中详细描述。 下面以取数指令(送至运算器的ACC 中)为例来说明，其信息流程如下：

1)取指令： PC→MAR→M→MDR→IR

根据PC取指令到IR。将 PC 的内容送MAR,MAR 中的内容直接送地址线，同时控制器将 读信号送读/写信号线，主存储器根据地址线上的地址和读信号，从指定存储单元读出指令，送到 数据线上， MDR 从数据线接收指令信息，并传送到IR 中。

2)分析指令： OP(IR)→CU

指令译码并送出控制信号。控制器根据 IR 中指令的操作码，生成相应的控制信号，送到不 同的执行部件。在本例中，IR 中是取数指令，因此读控制信号被送到总线的控制线上。

3)执行指令： Ad(IR)→MAR→M→MDR→ACC

取数操作。将 IR 中指令的地址码送 MAR,MAR 中的内容送地址线，同时控制器将读信 号送读/写信号线，从主存储器中读出操作数，并通过数据线送至MDR, 再传送到ACC 中 。

每取完一条指令，还须为取下条指令做准备，计算下条指令的地址，即(PC)+1→PC。



**注** **意**

(PC)指程序计数器 PC 中存放的内容。PC→MAR 应理解为(PC)→MAR, 即程序计数器中 的值经过数据通路送到 MAR, 也即表示数据通路时括号可省略(因为只是表示数据流经的途 径，而不强调数据本身的流动)。但运算时括号不能省略，即(PC)+1→PC 不能写为PC+1→PC 。 当题目中(PC)→MAR 的括号未省略时，考生最好也不要省略。

**1.2.6** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 完整的计算机系统应包括( )。

A. 运算器、存储器、控制器 B. 外部设备和主机

C. 主机和应用程序 D. 配套的硬件设备和软件系统

**02.** 冯·诺依曼机的基本工作方式是( )。

A. 控制流驱动方式 B. 多指令多数据流方式

C. 微程序控制方式 D. 数据流驱动方式

**03.** 下列( )是冯·诺依曼机工作方式的基本特点。

A. 多指令流单数据流 B. 按地址访问并顺序执行指令

C. 堆栈操作 D. 存储器按内容选择地址

**04.** 以下说法错误的是( )。

第1章 计算机系统概述 9

A. 硬盘是外部设备

B. 软件的功能与硬件的功能在逻辑上是等价的

C. 硬件实现的功能一般比软件实现具有更高的执行速度

D. 软件的功能不能用硬件取代

**05.** 存放当前执行指令的寄存器是()。

A.MAR B.PC C.MDR D.IR

**06.** 在 CPU 中，跟踪下一条要执行的指令的地址的寄存器是( )。

A.PC B.MAR C.MDR D.IR

07.CPU 不包括()。

A. 地址寄存器 B. 指令寄存器 (IR)

C. 地址译码器 D. 通用寄存器

08.MAR 和 MDR 的位数分别为( )。

A. 地址码长度、存储字长 B. 存储字长、存储字长

C. 地址码长度、地址码长度 D. 存储字长、地址码长度

**09.** 在运算器中，不包含()。

A. 状态寄存器 B. 数据总线 C.ALU D. 地址寄存器

**10.** 下列关于 CPU 存取速度的比较中，正确的是()。

A.Cache> 内 存 > 寄 存 器 B.Cache> 寄 存 器 > 内 存

C. 寄 存 器 >Cache> 内存 D. 寄 存 器 > 内 存 >Cache

**11.** 若一个8位的计算机系统以16位来表示地址，则该计算机系统有()个地址空间。 A.256 B.65535 C.65536 D.131072

**12.** ()是程序运行时的存储位置，包括所需的数据。

A. 数据通路 B. 主存 C. 硬盘 D. 操作系统

**13.** 关于编译程序和解释程序，下列说法中错误的是()。

A. 编译程序和解释程序的作用都是将高级语言程序转换成机器语言程序 B. 编译程序编译时间较长，运行速度较快

C. 解释程序方法较简单，运行速度也较快

D. 解释程序将源程序翻译成机器语言，并且翻译一条以后，立即执行这条语句 **14.** 可以在计算机中直接执行的语言和用助记符编写的语言分别是()。

**I.** 机器语言 II. 汇编语言Ⅲ . 高级语言 IV. 操作系统原语 V. 正则语言 A.Ⅱ 、Ⅲ B.Ⅱ 、IV C.I 、Ⅱ D.I 、V

**15.** 只有当程序执行时才将源程序翻译成机器语言，并且一次只能翻译一行语句，边翻译边 执行的是( )程序，把汇编语言源程序转变为机器语言程序的过程是( )。

I. 编译 IⅡ. 目标 Ⅲ . 汇 编 IV. 解释

A.I 、Ⅱ B.IV、H C.IV 、I D.IV、Ⅲ

**16.** 下列关于各种级别语言的描述中，错误的是( )。

A. 可用高级语言和低级语言编写出功能等价的程序

B. 低级语言的执行效率一般情况下高于高级语言

C. 机器语言源程序可在机器上直接执行，而高级语言和汇编语言源程序不可以 D. 汇编语言与机器结构无关

**17.** 下列关于机器指令和汇编指令的叙述中，错误的是()。

10-2025年计算机组成原理考研复习指导

A. 可以直接用机器语言(机器指令)编写程序

B. 汇编指令和机器指令都能被计算机直接执行

C. 汇编语言和机器语言都与计算机系统结构相关

D. 汇编指令和机器指令一一对应，功能相同

**18.** 下列叙述中，正确的是( )。

I. 实际应用程序的测试结果能够全面代表计算机的性能

II. 系列机的基本特性是指令系统向后兼容

Ⅲ.软件和硬件在逻辑功能上是等价的

A.Ⅱ B.Ⅲ C. Ⅱ 和 Ⅲ D.I 、Ⅱ 和 Ⅲ

**19.** 在 CPU 的组成中，不包括()。

A. 运算器 B. 存储器 C. 控制器 D. 寄存器

**20.** 关于相联存储器，下列说法中正确的是()。

A. 只可以按地址寻址 B. 只可以按内容寻址

**C**. 既可按地址寻址又可按内容寻址 D. 以上说法均不完善

**21.【2** 015统考真题】计算机硬件能够直接执行的是()。

**I.** 机器语言程序 II. 汇编语言程序 ⅢI. 硬件描述语言程序

A. 仅 I B. 仅 I 、Ⅱ C. 仅 I 、II D.I 、I 、Ⅲ

**22.** 【2016统考真题】将高级语言源程序转换为机器级目标代码文件的程序是( )。

A. 汇编程序 B. 链接程序 C. 编译程序 D. 解释程序

**23.** 【2019统考真题】下列关于冯·诺依曼计算机基本思想的叙述中，错误的是()。

A. 程序的功能都通过中央处理器执行指令实现

B. 指令和数据都用二进制数表示，形式上无差别

C. 指令按地址访问，数据都在指令中直接给出

D. 程序执行前，指令和数据需预先存放在存储器中

**24.【**2022统考真题】将高级语言源程序转换为可执行目标文件的主要过程是( )。

A. 预处理 →编译 → 汇编 →链接 B. 预处理 → 汇编 → 编译 →链接

C. 预处理 → 编译 →链接 → 汇编 D. 预处理 → 汇编 →链接 →编译

**1.2.7** **答** **案** **与** **解** **析**

**单项选择题**

**01.D**

A 是计算机主机的组成部分，而 B 、C 只涉及计算机系统的部分内容，都不完整。

**02.A**

冯·诺依曼机的基本工作方式是控制流驱动方式，也就是按照指令的执行序列，依次读取指 令，然后根据指令所含的控制信息，调用数据信息进行处理。因此，在执行程序的过程中，始终 以控制信息流为驱动工作的因素，而数据信息流则是被动地被调用处理。

**03.B**

A 是不存在的机器， B 是对“存储程序”的阐述，因此正确。C 是与题干无关的选项。D 是 相联存储器的特点。

**04.D**

软件和硬件具有逻辑功能上的等价性，硬件实现具有更高的执行速度，软件实现具有更好的

第1章 计算机系统概述 11

灵活性。执行频繁、硬件实现代价不是很高的功能通常由硬件实现。

05.D

IR 存放当前执行的指令代码，PC 存放下一条指令的地址，不要将它们混淆。此外， MAR 用 来存放待访问的存储单元地址， MDR 则用来存放取处存储单元的数据。

**06.** A

在 CPU 中 ，PC 用来跟踪下一条要执行的指令在主存储器中的地址。

07.C

地址译码器是主存储器的构成部分，不属于CPU 。 地址寄存器虽然一般属于主存储器，但现 代计算机中绝大多数 CPU 内集成了地址寄存器。

**08.** A

地址寄存器 (MAR ) 存放访存地址，因此位数与地址码长度相同。数据寄存器 (MDR) 用 于暂存要从存储器中读或写的信息，因此位数与存储字长相同。

09.D

运算器的核心是ALU 。地址寄存器位于CPU 内，但并未集成到运算器与控制器中。地址寄 存器用来保存当前CPU 所访问的内存单元的地址。因为内存和CPU 之间存在着操作速度上的差 别，所以必须使用地址寄存器来保持地址信息，直到内存的读/写操作完成为止。

10.C

寄存器在 CPU 内部，速度最快。Cache 采用高速的SRAM 制作，而内存常用DRAM 制作， 其速度较 Cache 慢。本题也可根据存储器层次结构的速度关系得出答案。

11.C

8位计算机表明计算机字长为8位，即一次可以处理8位的数据；而16位表示地址码的长度， 因此该机器有2¹⁶=65536个地址空间。

12.B

计算机只能从主存储器中取指令与操作数，不能直接与外存交换数据。

13.C

编译程序是先完整编译后运行的程序，如 C 、C++ 等；解释程序是逐句翻译且边翻译边执行 的程序，如 JavaScript 、Python 等。由于解释程序要边翻译成机器语言边执行，因此一般速度较 编译程序慢。为增加对该过程的理解，附C 语言编译链接的过程：

源程序(.c)—C编译器→汇编源程序—汇编程序→ 目标程序一链接程序→可执行程序

14.C

机器语言是计算机唯一可以直接执行的语言，汇编语言用助记符编写，以便记忆。而正则语 言是编译原理中符合正则文法的语言。

15.D

解释程序的特点是翻译一句执行一句，边翻译边执行；由高级语言转化为汇编语言的过程称 为编译，把汇编语言源程序翻译成机器语言程序的过程称为汇编。

16.D

在不同的设备中，汇编语言对应着不同的机器语言指令集，通过汇编程序转换成机器指令。

特定的汇编语言与特定的机器语言指令集是一一对应的，不同平台之间不可直接移植。

**17.B**

计算机只能直接执行机器指令，而汇编指令需要通过汇编程序转换成机器指令才能被计算机

12-2025 年计算机组成原理考研复习指导

直接执行。

**18.C**

全面代表计算机性能的是实际软件的运行情况。向后兼容是指时间上向后兼容，即新机器兼 容使用以前机器的指令系统。软件和硬件在逻辑功能上是等价的，如浮点运算即可以用专门的浮 点运算器实现，也可以通过编写一段子程序实现。

**19.B**

CPU 由运算器和控制器两个部件组成，而运算器和控制器中都含有寄存器。存储器是一个独 立的部件。

**20.C**

相联存储器既可以按地址寻址又可以按内容( 通常是某些字段)寻址，为与传统存储器区别， 也称按内容寻址的存储器。

**21.A**

硬件能直接执行的只能是机器语言(二进制编码),汇编语言是增强机器语言的可读性和记 忆性的语言，经过汇编后才能被执行。

**22.** C

翻译程序是指把高级语言源程序转换成机器语言程序的软件。翻译程序有两种： 一种是编译 程序，它将源程序一次全部翻译成目标程序，并且会生成目标代码文件。另一种是解释程序，它 将源程序的一条语句翻译成对应的机器目标代码，并立即执行，翻译一句执行一句，并且不会生 成目标代码文件。汇编程序也是一种翻译程序，它把汇编语言源程序翻译为机器语言程序。

**23.** C

冯·诺依曼结构计算机的功能部件包括输入设备、输出设备、存储器、运算器和控制器，程序 的功能都通过中央处理器(运算器和控制器)执行指令，A 正确。指令和数据以同等地位存放于存 储器内，形式上无差别，只在程序执行时具有不同的含义， B 正确。指令按地址访问，数据由指令 的地址码指出，除立即寻址外，数据均存放在存储器内， C 错 误。在程序执行前，指令和数据需预 先存放在存储器中，中央处理器可以从存储器存取代码，D 正确。

**24.A**

将源程序转换为可执行目标文件的过程分为预处理、编译、汇编、链接四个阶段。



**1.3计算机的性能指标**

**1.3.1** **计算机的主要性能指标**

**1.** **机器字长**

命 题 追 踪▶ **与机器字长位数相同的部件(2020、2021)**

通常所说的“某16位或32位机器”,其中的16、32指的是机器字长，简称字长。字长是指 计算机进行一次整数运算(即定点整数运算)所能处理的二进制数据的位数，通常与CPU 的寄存 器位数、ALU 有关。因此，字长一般等于通用寄存器的位数或ALU 的宽度，字长越长，数的表 示范围越大，计算精度越高。计算机字长通常选定为字节(8位)的整数倍。



**注** **意**

机器字长、指令字长和存储字长的关系(见章末的常见问题3)。



第1章 计算机系统概述 13

**2.** **数据通路带宽**

数据通路带宽是指数据总线一次所能并行传送信息的位数。这里所说的数据通路宽度是指外 部数据总线的宽度，它与CPU 内部的数据总线宽度(内部寄存器的大小)有可能不同。



**注** **意**

各个子系统通过数据总线连接形成的数据传送路径称为数据通路。

**3.** **主存容量**

主存容量是指主存储器所能存储信息的最大容量，通常以字节来衡量，也可用字数×字长(如 512K×16位)来表示存储容量。其中， MAR 的位数反映了存储单元的个数， MDR. 的位数反映了 存储单元的字长。例如， MAR 为16位，表示2l⁶=65536, 即此存储体内有65536个存储单元( 可 称为64K 内存，1K=1024 ), 若 MDR 为32位，则表示存储容量为64K×32 位。

**4.** **运算速度**

命 题 追 踪▶》 **提高系统性能的综合措施(2010)**

(1 ) 吞吐量和响应时间。

· 吞吐量。指系统在单位时间内处理请求的数量。它取决于信息能多快地输入内存， CPU 能 多快地取指令，数据能多快地从内存取出或存入，以及所得结果能多快地从内存送给一台 外部设备。几乎每步都关系到主存储器，因此系统吞吐量主要取决于主存储器的存取周期。 ●响应时间。指从用户向计算机发送一个请求，到系统对该请求做出响应并获得所需结果的 等待时间。通常包括 CPU 时间(运行一个程序所花费的时间)与等待时间(用于磁盘访

问、存储器访问、I/O 操作、操作系统开销等的时间)。

(2)主频和CPU 时钟周期。

▶ **时钟脉冲信号和时钟周期的相关概念(2019)**

·CPU 时钟周期。机器内部主时钟脉冲信号的宽度，它是 CPU 工作的最小时间单位。

时钟脉冲信号由机器脉冲源发出的脉冲信号经整形和分频后形成。

时钟周期以相邻状态单元间组合逻辑电路的最大延迟为基准确定。

时钟周期也以指令流水线的每个流水段的最大延迟时间确定。

**主频和时钟周期的转换计算(2013)**

●主频 (CPU 时钟频率)。机器内部主时钟的频率，即时钟周期的倒数，它是衡量机器速度 的重要参数。对于同一个型号的计算机，其主频越高，完成指令的一个执行步骤所用的时 间越短，执行指令的速度越快。主频最直观的理解就是每秒有多少个时钟周期。



**注** **意**

CPU 时钟周期=1/主频，主频通常以 Hz (赫兹)为单位，10Hz 表示每秒10次。

(3 ) CPI(Cycle Per Instruction), 即执行一条指令所需的时钟周期数。

**命题追踪** ▶ **IPS** **的相关计算(2023)**

不同指令的时钟周期数可能不同，因此对于一个程序或一台机器来说，其CPI 指该程序或该 机器指令集中的所有指令执行所需的平均时钟周期数，此时 CPI是一个平均值。

14-2025年计算机组成原理考研复习指导

·IPS(Instructions Per Second), 即每秒执行多少条指令，IPS= 主频/平均 CPI。 (4)CPU 执行时间。指运行一个程序所花费的时间。

命题追踪 ▶ **CP** **U** **执行时间的相关计算(2012、2013、2014、2017、2022、2023)**

CPU 执 行 时 间 =CPU 时钟周期数/主频=(指令条数×CPI)÷ 主频

上式表明， CPU 的性能 (CPU 执行时间)取决于三个要素：主频、CPI和指令条数。主频、 CPI 和指令条数是相互制约的。例如，更改指令集可以减少程序所含的指令条数，但同时可能引 起 CPU 结构的调整，从而可能会增加时钟周期的宽度(降低主频)。

**【例1.1】**假定计算机M1 和 M2 具有相同的指令集体系结构，M1 的主频为2GHz, 程 序P 在 M1 上的运行时间为10s 。M2 采用新技术可使主频大幅提升，但平均CPI 也增加到M1 的1.5倍。 则 M2 的主频至少提升到多少才能使程序P 在 M2 上的运行时间缩短为6s?

解 ：

程 序P 在M1 上的时钟周期数=指令条数×CPI=CPU 执行时间×主 频 = 1 0s×2GHz=2×10¹ 。 M2 的平均CPI 为 M1 的1.5倍，因此程序P 在 M2 上的时钟周期数=1.5×2×10¹⁰=3×10¹°。 要使程序P 在 M2 上的运行时间缩短到6s, 则 M2 的主频至少应为

程序P 所含时钟周期数=CPU 执行时间=3×10¹⁰÷6s=5GHz 由此可见， M2 的主频是 M1 的2.5倍，但M2 的速度却只是 M1 的1.67倍。 (5)MIPS(Million Instructions Per Second), 即每秒执行多少百万条指令。

▶ **MIPS 相关的计算(2012、2013)**

MIPS= 指令条数÷(执行时间×10⁶)=主频÷(CPI×10⁶)。

MIPS 对不同机器进行性能比较是有缺陷的，因为不同机器的指令集不同，指令的功能也就 不同，比如在机器M1 上某条指令的功能也许在机器M2 上要用多条指令来完成；不同机器的CPI 和时钟周期也不同，因而同一条指令在不同机器上所用的时间也不同。

(6)FLOPS(Floating-point Operations Per Second), 即每秒执行多少次浮点运算。

命 题 逢 踪 ▶ **浮点数运算指标的概念(2011、2021)**

·MFLOPS(Million ·GFLOPS(Giga

·TFLOPS(Tera ·PFLOPS(Peta

FLOPS), 即每秒执行多少百万(10⁶)次浮点运算。

FLOPS), 即每秒执行多少十亿 ( 10°)次浮点运算。

FLOPS), 即每秒执行多少万亿(10'²)次浮点运算。

FLOPS), 即每秒执行多少千万亿(10 ¹⁵)次浮点运算。

·EFLOPS(ExaFLOPS), 即每秒执行多少百京(10¹⁸)次浮点运算(1京=1亿亿=10¹⁶)。

·ZFLOPS(Zetta FLOPS), 即每秒执行多少十万京(10²¹)次浮点运算。



**注** **意**

在描述存储容量、文件大小等时，K 、M 、G 、T 通常用2的幂次表示，如1Kb=2Ib; 在描 述速率、频率等时， k 、M 、G 、T 通常用10的幂次表示，如1kb/s=10b/s 。 通常前者用大写的 K, 后者用小写的 k, 但其他前缀均为大写，表示的含义取决于所用的场景。

**5.** **基准程序**

基准程序 (Benchmarks) 是专门用来进行性能评价的一组程序，能够很好地反映机器在运行 实际负载时的性能，可以通过在不同机器上运行相同的基准程序来比较在不同机器上的运行时

第1章 计算机系统概述 15

间，从而评测其性能。对于不同的应用场合，应该选择不同的基准程序。

使用基准程序进行计算机性能评测也存在一些缺陷，因为基准程序的性能可能与某一小段的 短代码密切相关，而硬件系统设计人员或编译器开发者可能会针对这些代码片段进行特殊的优 化，使得执行这段代码的速度非常快，以至于得不到准确的性能评测结果。

**1.3.2** **几** **个** **专** **业** **术** **语**

1)系列机。具有基本相同的体系结构，使用相同基本指令系统的多个不同型号的计算机组 成的一个产品系列。

2 ) 兼 容。指软件或硬件的通用性，即运行在某个型号的计算机系统中的硬/软件也能应用 于另一个型号的计算机系统时，称这两台计算机在硬件或软件上存在兼容性。

3)固件。将程序固化在 ROM 中组成的部件称为固件。固件是一种具有软件特性的硬件，吸 收了软件/硬件各自的优点，其执行速度快于软件，灵活性优于硬件，是软/硬件结合的产 物。例如，目前操作系统已实现了部分固化(把软件永恒地存储于ROM 中)。

**1.3.3** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 关于 CPU 主频、CPI 、MIPS 、MFLOPS, 说法正确的是( )。

A.CPU 主频是指 CPU 系统执行指令的频率，CPI 是执行一条指令平均使用的频率

B.CPI 是执行一条指令平均使用CPU 时钟的个数，MIPS 描述一条CPU 指令平均使用

的CPU 时钟数

C.MIPS 是描述CPU 执行指令的频率， MFLOPS 是计算机系统的浮点数指令

D.CPU 主频指CPU 使用的时钟脉冲频率， CPI 是执行一条指令平均使用的 CPU 时钟数 **02.** 存储字长是指( )。

A. 存放在一个存储单元中的二进制代码组合

B. 存放在一个存储单元中的二进制代码位数

C. 存储单元的个数

D. 机器指令的位数

**03.** 以下说法中，错误的是()。

A. 计算机的机器字长是指数据运算的基本单位长度

B. 寄存器是由触发器构成的

C. 计算机中一个字的长度都是32位

D. 磁盘可以永久性存放数据和程序

**04.** 下列关于机器字长、指令字长和存储字长的说法中，正确的是()。

I. 三者在数值上总是相等的 Ⅱ.三者在数值上可能不等

Ⅲ.存储字长是存放在一个存储单元中的二进制代码位数

IV. 机器字长就是MDR 的位数

A.I 、II B.I 、IV C.I 、I D.I 、IV

**05.** 下列关于机器字长的叙述中，错误的是()。

A. 机器字长是指 CPU 中定点运算的数据通路宽度

B. 机器字长一般与 CPU 中通用寄存器的位数有关

C. 机器字长决定了数据的表示范围和表示精度

16-2025年计算机组成原理考研复习指导

D. 机器字长对计算机硬件的造价没有影响

06.32位微机是指该计算机所用CPU()。

A. 具有32位寄存器 B. 能同时处理32位的二进制数

C. 具有32个寄存器 D. 能处理32个字符

07. 在用于科学计算的计算机中，标志系统性能的最有用的参数是()。

A. 主时钟频率 B. 主存容量 C.MFLOPS D.MIPS

**08.** 在计算机 M1 和计算机 M2 上分别运行功能完全相同的高级语言程序，程序在 M1 和 M2 上的平均 CPI 相等，则对于该类程序而言()。

A.M1 和 M2 执行速度相等

B.M1 和 M2 中主频高的计算机执行速度快

C.M1 和 M2 中主频低的计算机执行速度快

D. 无法确定哪台机器的执行速度快

**09.** 若一台计算机的机器字长为4B, 则表明该机器()。

A. 能处理的数值最大为4位十进制数

B. 能处理的数值最多为4位二进制数

C. 在 CPU 中能够作为一个整体处理32位的二进制代码

D. 在 CPU 中运算的结果最大为2³2

**10.** 在 CPU 的寄存器中，()对用户是完全透明的。

A. 程序计数器 B. 指令寄存器 C. 状态寄存器 D. 通用寄存器

**11.** 计算机操作的最小单位时间是()。

A. 时钟周期 B. 指令周期 C.CPU 周期 D. 中断周期

**12.** 计算机中，CPU 的 CPI 与下列()因素无关。

A. 时钟频率 B. 系统结构 C. 指令集 D. 计算机组织

**13.** 从用户观点看，评价计算机系统性能的综合参数是()。

A. 指令系统 B. 吞吐率 C. 主存容量 D. 主频率

**14.** 当前设计高性能计算机的重要技术途径是( )。

A. 提 高CPU 主频 B. 扩大主存容量

C. 采用非冯·诺依曼体系结构 D. 采用并行处理技术

**15.** 下列关于“兼容”的叙述，正确的是()。

A. 指计算机软件与硬件之间的通用性，通常在同一系列不同型号的计算机间存在 B. 指计算机软件或硬件的通用性，即它们在任何计算机间可以通用

C. 指计算机软件或硬件的通用性，通常在同一系列不同型号的计算机间通用 D. 指软件在不同系列计算机中可以通用，而硬件不能通用

**16.** 若某基准测试程序在机器A 上运行需要的时间是20s, 而在机器B 上的运行时间是16s, 那么,相对来说，下列给出的结论中，()是正确的。

A. 所有程序在机器A 上都比在机器 B 上运行速度慢

B. 机器B 的速度是机器A 的1.25倍

C. 机器A 的速度是机器 B的1.25倍

D. 机器 A 比机器B 慢1.25倍

**17.** 机器A 的主频为800MHz, 某程序在A 上运行需要12s。现在硬件设计人员想设计机器 **B,** 希望该程序在B 上的运行时间能缩短为8s, 使用新技术后可使B 的主频大幅度提高，

第1章 计算机系统概述 17

但在B上运行该程序所需的时钟周期数为在A 上的1.5倍。则机器B的主频至少应为( )。 A.800MHz B.1.2GHz C.1.5GHz D.1.8GHz

**18.** 下列可用于评价计算机系统性能的指标是()。

I.MIPS I.IPC I.CPI IV. 字长

A.I 、 Ⅲ B.I 、Ⅲ 和 IV C.I 、Ⅱ 和 Ⅲ D. 全部

**19.** 计算机的机器字长与下列( )指标最为密切相关。

A. 运算速度 B. 存取速度 C. 内存容量 D. 运算精确度

**20.** 假定编译器对高级语言的某条语句可以编译生成两种不同的指令序列， A 、B 和 C 三类 指令的 CPI 和两种不同序列所含的三类指令条数如下表所示，两个指令序列都在时钟周 期为2ns 的机器上运行，则下列结论中正确的是()。

|  |  |  |  |
| --- | --- | --- | --- |
| **指令类型** | **CPI** | **序列一的指令条数** | **序列二的指令条数** |
| A |  | 1 | 2 |
| B | 2 | 1 | 1 |
| C | 3 | 4 | 2 |

A. 序列一的 MIPS 数比序列二多50,序列一的执行速度比序列二快10ns B. 序列一的 MIPS 数比序列二多50,序列二的执行速度比序列一快10ns C. 序列二的 MIPS 数比序列一多50,序列一的执行速度比序列二快10ns D. 序列二的 MIPS 数比序列一多50,序列二的执行速度比序列一快10ns

**21.** 下列给出了改善计算机性能的4种措施：

I. 用更快的处理器来替换原来的慢速处理器

II. 增加同类处理器个数，使得不同的处理器同时执行程序

Ⅲ.优化编译生成的代码，使得程序执行的总时钟周期数减少

IV. 减少指令执行过程中访问内存的时间

对于某个特定的程序，在以上措施中，能缩短其执行时间的措施是()。

A.I 、Ⅱ 和 Ⅲ B.I 、Ⅱ 和IV C.I 、Ⅲ 和IV D. 全 部

**22.** 【2010统考真题】下列选项中，能缩短程序执行时间的措施是( )。

I. 提 高CPU 时钟频率 II. 优化数据通路结构 ⅢI. 对程序进行编译优化

A. 仅I 和I B. 仅 I 和 ⅢI C. 仅 Ⅱ 和 Ⅲ D.I、Ⅱ、Ⅲ

**23.** 【2011 统考真题】下列选项中，描述浮点数操作速度指标的是()。

A.MIPS B.CPI C.IPC D.MFLOPS

**24.**【2012统考真题】假定基准程序 A 在某计算机上的运行时间为100s, 其中90s 为 CPU 时间，其余为I/O 时间。若CPU 速度提高50%, I/O 速度不变，则运行基准程序A 所耗 费 的 时 间 是 ( )

A.55s B.60s C.65s D.70s

**25.**【2013统考真题】某计算机的主频为1.2GHz, 其指令分为4类，它们在基准程序中所占 比例及 CPI 如下表所示。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **指令类型** | **所占比例** | **CPI** | **指令类型** | **所占比例** | **CPI** |
| A | 50% | 2 | C | 10% | 4 |
| B | 20% | 3 | D | 20% | 5 |

该机的 MIPS 数 是 ( ) 。

18 -2025年计算机组成原理考研复习指导

A.100 B.200 C.400 D.600

**26.【**2014统考真题】程序P 在机器M 上的执行时间是20s, 编译优化后，P 执行的指令数 减少到原来的70%,而CPI 增加到原来的1.2倍，则P 在 M 上的执行时间是( )。

A.8.4s B.11.7s C.14s D.16.8s

**27.**【2017统考真题】假定计算机M1 和 M2 具有相同的指令集体系结构 (ISA), 主频分别 为1.5GHz 和1.2GHz。 在 M1 和M2 上运行某基准程序P, 平均CPI 分别为2和1,则程

序P 在 M1 和 M2 上运行时间的比值是( )。

A.0.4 B.0.625 C.1.6 D.2.5

**28.【**2020统考真题】下列给出的部件中，其位数(宽度)一定与机器字长相同的是()。

I. ALU II. 指令寄存器 ⅢI. 通用寄存器 IV. 浮点寄存器

A. 仅 I 、I B. 仅 I、ⅢⅢ C. 仅Ⅱ、Ⅲ D. 仅Ⅱ、Ⅲ、 IV

**29.【**2021统考真题】2017年公布的全球超级计算机 TOP500 排名中，我国“神威·太湖之 光”超级计算机蝉联第一，其浮点运算速度为93.0146 PFLOPS, 说明该计算机每秒完成 的浮点操作次数约为( )。

A.9.3×10¹³ 次 B.9.3×10¹⁵ 次 C.9.3 千万亿次 D.9.3 亿亿次

**30.**【2022统考真题】某计算机主频为1GHz, 程序P 运行过程中，共执行了10000条指令， 其中，80%的指令执行平均需1个时钟周期，20%的指令执行平均需10个时钟周期。程 序P 的平均CPI 和CPU 执行时间分别是()。

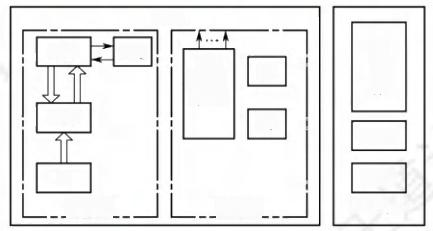
A.2.8,28μs B.28,28μs C.2.8,28ms D.28,28ms

**31.【**2023统考真题】若机器M 的主频为1.5GHz, 在 M 上执行程序P 的指令条数为5×10⁵, P 的平均CPI 为1.2,则P 在 M 上的指令执行速度和用户 CPU 时间分别为( )。

A.0.8GIPS,0.4ms B.0.8GIPS,0.4μs C.1.25GIPS,0 .4ms D.1.25GIPS,0.4μs

**二、综合应用题**

**01.** 设主存储器容量为64K×32位，且指令字长、存储字长、机器字长三者相等。写出如下 图所示各寄存器的位数，并指出哪些寄存器之间有信息通路【本题涉及第5章的内容】。



CPU

MQ

CU

(控制

单元)

PC

MDR

MAR

主存储器

存 储 体

运算器

控制器

ACC

ALU

IR

X

**02.** 用一 台40MHz 的处理器执行标准测试程序，它所包含的混合指令数和响应所需的时钟 周期见下表。求有效的 CPI 、MIPS 速率和程序的执行时间(I 为程序的指令条数)。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 指令类型 | CPI | 指令混合比 | 指令类型 | CPI | 指令混合比 |
| 算术和逻辑 | 1 | 60% | 转移 | 4 | 12% |
| 高速缓存命中的访存 | 2 | 18% | 高速缓存失效的访存 | 8 | 10% |

**03.** 微机 A 和 B 是采用不同主频的 CPU 芯片，片内逻辑电路完全相同。

1)若A 机的CPU 主频为8MHz,B 机为12MHz , 则 A 机的 CPU 时钟周期为多少?

第1章 计算机系统概述 19

2 ) 若A 机的平均指令执行速度为0.4MIPS, 则 A 机的平均指令周期为多少?

3)B 机的平均指令执行速度为多少?

**04.** 某台计算机只有LOAD/STORE 指令能对存储器进行读/写操作，其他指令只对寄存器进

行操作。根据程序跟踪试验结果，已知每条指令所占的比例及CPI 数如下表所示。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **指令类型** | **指令所占比例** | **CPI** | **指令类型** | **指令所占比例** | **CPI** |
| 算术逻辑指令 | 43% | 1 | STORE指令 | 12% | 2 |
| LOAD指令 | 21% | 2 | 转移指令 | 24% | 2 |

求上述情况下的平均 CPI。

假设程序由 M 条指令组成。算术逻辑运算中25%的指令的两个操作数中的一个已在寄 存器中，另一个必须在算术逻辑指令执行前用LOAD 指令从存储器中取到寄存器中。因 此有人建议增加另一种算术逻辑指令，其特点是一个操作数取自寄存器，另一个操作数 取自存储器，即寄存器-存储器类型，假设这种指令的 CPI 等于2。同时，转移指令的 CPI 变为3。求新指令系统的平均CPI。

**1.3.4** **答案与解析**

**一** **、单项选择题**

**01.D**

CPU 主频指 CPU 的时钟脉冲频率，CPI 是执行一条指令平均使用的 CPU 时钟数。

**02.B**

存储体由许多存储单元组成，每个存储单元又包含若干存储元件，每个存储元件能寄存一位 二进制代码“0”或“1”。可见， 一个存储单元可存储一 串二进制代码，称这串二进制代码为一 个存储字，称这串二进制代码的位数为存储字长。

**03.** C

计算机中一个字的长度可以是16、32、64位等， 一般是8的整数倍，不一 定都是32位。

**04.** C

机器字长、存储字长和指令字长，三者在数值上可以相等也可以不等，视不同机器而定。机 器字长等于CPU 内部的运算器位数和通用寄存器宽度。 一个存储单元中的位数称为存储字长，它 等于MDR 的位数。指令字长取决于指令的功能和格式，可以是单字长、半字长或双字长。

05.D

机器字长会影响硬件的造价。它将直接影响加法器(或ALU), 内部总线宽度及寄存器的位 数。所以机器字长不能只从数据的精度和表示范围来考虑，还要考虑硬件成本和效率。

**06.B**

计算机的位数是指机器字长，也就是计算机一次能处理的二进制数的长度，通常等于 ALU 的宽度或通用寄存器的位数。操作系统的位数是指操作系统可寻址的位数，它与机器字长不同。

07.C

MFLOPS 是指每秒执行多少百万次浮点运算，该参数用来描述计算机的浮点运算性能，而用 于科学计算的计算机主要评估浮点运算的性能。

08.D

CPU 执行时间=指令条数×CPI× 时钟周期的长度，程序在 M1 和 M2 上的平均 CPI 相等，但 影 响CPU 执行时间的因素还有指令条数和时钟周期的长度，此外相同的高级语言程序在不同计算

20-2025年计算机组成原理考研复习指导

机上编译生成的机器指令条数可能不同，因此无法确定哪台机器执行该类程序的速度快。

**09.C**

机器字长是CPU 一 次可以处理的二进制代码的位数，因此该机一 次可处理4×8=32位的二

进制代码。计算机的数据表示格式有多种，不同的表示方式所能表示的数据范围不同。

**10.B**

汇编程序员可以通过JMP 指令来设置 PC 的值。状态寄存器、通用寄存器只有为汇编程序员 可见，才能实现编程，而IR、MAR、MDR 是 CPU 的内部工作寄存器，对程序员均不可见。

11.A

时钟周期即 CPU 频率的倒数，是最基本的时间单位，其余选项均大于时钟周期。另外， CPU 周期也称机器周期，它由多个时钟周期组成。

**12.** A

CPI 是执行一条指令所需的时钟周期数，系统结构、指令集、计算机组织都会影响 CPI, 而 时钟频率并不会影响CPI, 但可加快指令的执行速度。例如，执行一条指令需要10个时钟周期， 则 一 台主频为1GHz 的 CPU, 执行这条指令要比一 台主频为100MHz 的 CPU 快 。

**13.B**

主频、主存储器容量和指令系统(间接影响 CPI) 并不是综合性能的体现。吞吐率指系统在 单位时间内处理请求的数量，是评价计算机系统性能的综合参数。

**14.D**

提高 CPU 主频、扩大主存储器容量对性能的提升是有限度的。采用并行技术是实现高性能

计算的重要途径，现今超级计算机均采用多处理器来增强并行处理能力。

**15.C**

兼容是指计算机软件或硬件的通用性，因此A、D 错误。对于B, 它们在任何计算机间可以

通用，错误。对于C, 兼容通常在同一系列的不同型号计算机间，正确。

**16.B**

机器的速度与基准程序在该机器上的运行时间呈相反关系，因此可知：机器B 的速度/机器A 的 速 度 = 基 准 程 序 在 机 器A 上的运行时间/基准程序在机器B 上 的 运 行 时 间 = 2 0s÷16s =1.25。 因此，可以说，机器B 的速度是机器A 的1 .25倍，或者机器 A 的速度是机器B 的0 . 8倍。

17.D

该程序在机器A 上需要的时钟周期数为12×800M=9600M, 因为在机器B 上运行该程序所

需的时钟周期数为在A 上的1 . 5倍，所以在 B 上需要的时钟周期数为9600M×1.5=14400M=

14.4G, 要求运行时间为8s, 故 B 的时钟频率为14.4G÷8=1.8GHz。

**18.D**

显 然 ，MIPS 、CPI 、 字长都是评价计算机系统的性能指标。IPC 表示每个时钟周期运行多少 条指令，它是CPI 的倒数。

19.D

机器字长越长，数据的位数越多，定点数或浮点数所表示及运算的精度就越高， D 正确。机

器字长与运算速度的关系不大，机器字长与存取速度和内存容量基本没有关系。

20.D

MIPS= 主频+(CPI×10), 主频=1/时钟周期=1/2ns=500M, 序列一的CPI=(1×1+1×2+4×3)-6=

15÷6=2.5,序列二的CPI=(2×1+1×2+2×3)÷5=10=5=2, 故序列一的MIPS=500M÷(2.5×10⁶)=200,

第1章 计算机系统概述- 21

序列二的MIPS=500M÷(2×10⁶)=250。CPU 执行时间=指令条数×CPI×时钟周期=程序的时钟周期 数×时钟周期，序列一所需的时钟周期数是15,序列二所需的时钟周期数是10,故序列一的执行时间 为15×2ns=30ns, 序列二的执行时间为10×2ns=20ns。

**21.D**

采用更快的处理器，可以减少单条指令的执行时间；增加处理器的个数，可以增加程序执行 的并行性，缩短程序的执行时间；优化编译代码，可以减少指令之间的各种冲突；访存时间占指 令执行的大部分时间，减少访存时间同样可以大大加快指令的执行时间。

**22.D**

CPU 时钟频率(主频)越高，完成指令的一个执行步骤所用的时间就越短，执行指令的速度 就越快， I 正 确。数据通路的功能是实现CPU 内部的运算器和寄存器及寄存器之间的数据交换， 优化数据通路结构，可以有效提高计算机系统的吞吐量，从而加快程序的执行，Ⅱ正确。计算机 程序需要先转化成机器指令序列才能最终得到执行，通过对程序进行编译优化可以得到更优的指 令序列，从而使得程序的执行时间也越短，Ⅲ正确。

**23.** D

MIPS 是每秒执行多少百万条指令，适用于衡量标量机的性能。CPI 是平均每条指令的时钟 周期数。IPC 是 CPI 的倒数，即每个时钟周期执行的指令数。MFLOPS 是每秒执行多少百万条浮 点数运算，用来描述浮点数运算速度，适用于衡量向量机的性能。

**24.D**

程序 A 的运行时间为100s, 除 去CPU 时间90s, 剩余10s 为 I/O 时 间 。CPU 提速50%后运 行基准程序 A 所耗费的时间是T=90÷1.5+1 0=70s。

**误区：**CP U速度提高50%,而误认为CPU 时间减少一半，从而误选A。

**25.** C

基准程序的CPI=2×0.5+3×0.2+4×0.1+5×0.2=3 。 计算机的主频为1.2GHz, 即1200MHz, 因

此该机器的 MIPS=1200÷3=400。

**26.D**

假设原来的指令条数为x, 则 原CPI 为20f/x(f 为 CPU 的时钟频率),经过编译优化后，指 令条数减少到原来的70%,即指令条数为0.7x, 而 CPI 增加到原来的1.2倍，即24f/x, 则现在P 在 M 上的执行时间就为：(指令条数×CPI)/f=(0.7x×24×f/x)/f=24×0.7=16.8s。

27.C

运行时间=指令数×CPI/主频。M1 的时间=指令数×2÷1.5,M2 的时间=指令数×1/1.2, 两者之比为(2/1.5):(1/1.2)=1.6。

**28.B**

机器字长是指 CPU 内部用于整数运算的数据通路的宽度。数据通路是指数据在指令执行过 程中所经过的路径及路径上的部件，主要是CPU 内部进行数据运算、存储和传送的部件，这些部 件的宽度基本上要一致才能相互匹配。因此，机器字长等于ALU 位数和通用寄存器宽度。

29.D

PFLOPS= 每秒1000万亿(10¹⁵)次浮点运算。故93.0146 PFLOPS≈ 每秒9.3×10¹⁶次浮点运 算，即每秒9.3亿亿次浮点运算。

**30.** A

CPI 指平均每条指令的执行需要多少个时钟周期。由于80%的指令执行平均需要1个时钟周

22-2025年计算机组成原理考研复习指导

期，20%的指令执行平均需要10个时钟周期，因此CPI=80%×1+20%×10=2.8 。 计算机主频为 1GHz, 程 序P 共执行10000条指令，平均每条指令需要2.8个时钟周期，因此，CPU 执 行 时 间 = (10000×2.8)÷10⁹=2.8×10~⁵s=28μs。

**31.C**

程 序P 的指令条数为5×10³,平均CPI 为1 . 2,程序P 的总时钟周期数为5×10⁵ ×1.2=6×10°, 主频1 . 5GHz 说 明 1s 有 1 . 5G=1.5×10° 个时钟周期。因此，指令执行速度=主频/平均 CPI = 1.5G+1.2=1.25GIPS, 用 户CPU 时间=6×10³+(1.5×10°)s=4×10\*s=0.4ms。

**二、** **综合应用题**

**01.** 【解答】

因为主存储器容量为64K×32 位，且2¹6=64K, 所以地址总线宽度为16位，32位表示数据 总线宽度，于是MAR 为 1 6 位 ，PC 为 1 6 位 ，MDR 为 3 2 位 。

因为指令字长=存储字长=机器字长，所以IR、ACC 、MQ 、X 均 为 3 2 位 。

寄存器之间的信息通路有：

PC→MAR

Ad(IR)→MAR

MDR→IR

取 数 ：MDR→ACC, 存 数 ：ACC→MDR

MDR→X

**02.** 【解答】

CPI 即执行一条指令所需的时钟周期数。本标准测试程序共包含4种指令，则CPI 就是这4 种指令的数学期望，即

CPI=1×60%+2×18%+4×12%+8×10%=2.24

MIPS 即每秒执行的百万条指令数。已知处理器时钟频率为40MHz, 即每秒包含40M 个时钟 周期，因此

MIPS=40/CPI=40÷2.24=17.9

程序的执行时间T=CPI×T\_IC×I, 其 中T\_IC 是 一 个CPU 时钟的时间长度，是 CPU 时钟频

率f 的倒数，因此有

T=CPI×T\_IC×I=CPI×(1f)×I=5.6×10⁸×1 秒

本题中的1对于解题应无作用，程序的执行时间应是指令的期望即CPI 乘以时钟的时间长度， 即 T=CPI×T\_IC。

03. 【解答】

1)A 机 的CPU 主 频 为 8MHz, 所 以A 机 的CPU 时钟周期=1÷8MHz=0.125μs。

2)A 机的平均指令周期=1÷0.4MIPS =2.5μs。

3)A 机平均每条指令的时钟周期数=2.5μs+0.125μs=20。

因微机 A 和 B 的片内逻辑电路完全相同，所以B 机平均每条指令的时钟周期数也为20。

因 为B 机 的CPU 主频为12MHz, 所 以B 机 的CPU 时钟周期=1÷12MHz=1/12μs。 B 机的平均指令周期=20×(1/12)=5/3μs。

B 机的平均指令执行速度=1+(5/3)us=0.6MIPS。

【 另 解 】B 机的平均指令执行速度=A 机的平均指令执行速度×(12/8)=0.4MIPS×(12/8)=

0.6MIPS。

**1.4**

第1章 计算机系统概述 23

**04.【** 解答】

① 本处理机共包含4种指令，则CPI 就是这4种指令的数学期望，即

CPI=1×43%+2×21%+2×12%+2×24%=1.57

② 设原指令总数为M, 由于新增的算术操作有取操作数的功能，替代了LOAD 的功能，所

以新指令总数为

M+(0.25×0.43M)-(0.25×0.43M)-(0.25×0.43M)=0.8925M

增加另一种算术逻辑指令后，每种指令所占的比例及CPI 数如下表所示：

|  |  |  |
| --- | --- | --- |
| **指令类型** | **指令所占比例** | **CPI** |
| 算术逻辑指令 | (0.43M-0.43M×0.25V0.8925M=0.3613 | 1 |
| 算术逻辑指令(新) | (0.43M×0.25)/0.8925M=0.1204 | 2 |
| LOAD指令 | (0.21M-0.43M×0.25)0.8925M=0.1149 | 2 |
| STORE指令 | 0.12M/0.8925M=0.1345 | 2 |
| 转移指令 | 0.24M/0.8925M=0.2689 | 3 |

所 以CPI'=1×0.3613+2×0.1204+2×0.1149+2×0.1345+3×0.2689=1.9076。



**本章小结**

本章开头提出的问题的参考答案如下。

1)计算机由哪几部分组成?以哪部分为中心?

计算机由运算器、控制器、存储器、输入设备及输出设备五大部分构成，现代计算机通常把 运算器和控制器集成在一个芯片上，合称中央处理器。

在微处理器面世之前，运算器和控制器分离，而且存储器的容量很小，因此设计成以运算器 为中心的结构，其他部件都通过运算器完成信息的传递。随着微电子技术的发展，同时计算机需 要处理、加工的信息量也与日俱增，大量I/O 设备的速度和CPU 的速度差距悬殊，因此以运算器 为中心的结构不能满足计算机发展的要求。现代计算机已发展为以存储器为中心，使 I/O 操作尽 可能地绕过CPU, 直 接 在I/O 设备和存储器之间完成，以提高系统的整体运行效率。

2)主频高的 CPU 一 定比主频低的 CPU 快 吗 ? 为 什 么 ?

衡量 CPU 运算速度的指标有很多，不能以单独的某个指标来判断 CPU 的好坏。CPU 的 主 频 表示CPU 内数字脉冲信号振荡的速度，主频和实际的运算速度存在一定的关系，但目前还没有一 个确定的公式能够定量两者的数值关系，因为CPU 的运算速度还要看 CPU 的流水线的各方面的 性能指标(架构、缓存、指令集、CPU 的位数、Cache 大 小 等 )。由于主频并不直接代表运算速度， 因此在一定情况下很可能会出现主频较高的CPU 实际运算速度较低的现象。

3)翻译程序、汇编程序、编译程序、解释程序有什么差别?各自的特性是什么?

见常见问题和易混淆知识点1。

4)不同级别的语言编写的程序有什么区别?哪种语言编写的程序能被硬件直接执行?

机器语言和汇编语言与机器指令对应，而高级语言不与指令直接对应，具有较好的可移植性。 其中机器语言编写的程序可以被硬件直接执行。

24 -2025年计算机组成原理考研复习指导



**1.5** **常见问题和易混淆知识点**

1. 翻译程序、解释程序、汇编程序、编译程序的区别和联系是什么?

翻译程序有两种： 一种是编译程序，它将高级语言源程序一次全部翻译成目标程序，只要源 程序不变，就无须重新翻译。另一种是解释程序，它将源程序的一条语句翻译成对应的机器目标 代码，并立即执行，然后翻译下一条源程序语句并执行，直至所有源程序语句全部被翻译并执行 完。所以解释程序的执行过程是翻译一句执行一句，并且不会生成目标程序。

汇编程序也是一种语言翻译程序，它把汇编语言源程序翻译为机器语言程序。

编译程序与汇编程序的区别：若源语言是诸如 C 、C++ 、Java 等“高级语言”,而目标语言是 诸如汇编语言或机器语言之类的“低级语言”,则这样的一个翻译程序称为编 译程序。若源语言 是汇编语言，而目标语言是机器语言，则这样的一个翻译程序称为汇编程序。

2. 什么是透明性?透明是指什么都能看见吗?

在计算机领域中，站在某类用户的角度，若感觉不到某个事物或属性的存在，即“看”不到 某个事物或属性，则称为“对该用户而言，某个事物或属性是透明的”。这与日常生活中的“透 明”概念(公开、看得见)正好相反。例如，对于高级语言程序员来说，浮点数格式、乘法指令 等这些指令的格式、数据如何在运算器中运算等都是透明的；而对于机器语言或汇编语言程序员 来说，指令的格式、机器结构、数据格式等则不是透明的。

在 CPU 中 ，IR、MAR 和 MDR 对各类程序员都是透明的。

3. 字、字长、机器字长、指令字长、存储字长的区别和联系是什么?

字长是指 CPU 内部用于整数运算的数据通路的宽度，因此字长等于CPU 内部用于整数运算 的运算器位数和通用寄存器宽度，它反映了计算机处理信息的能力。 字和字长的概念不同。字用 来表示被处理信息的单位，用来度量数据类型的宽度，如x86 机器中将 一个字定义为16位。

指令字长： 一个指令字中包含的二进制代码的位数。

存储字长： 一个存储单元存储的二进制代码的位数。

它们都必须是字节的整数倍。

指令字长一般取存储字长的整数倍，若指令字长等于存储字长的2倍，则需要2个访存周期 来取出一条指令；若指令字长等于存储字长，则取指令周期等于机器周期。

早期的存储字长一般与指令字长、字长相等，因此访问一次主存储器便可取出一条指令或一 个数据。随着计算机的发展，指令字长、字长都可变，但必须都是字节的整数倍。

4. 计算机体系结构和计算机组成的区别和联系是什么?

计算机体系结构是指机器语言或汇编语言程序员所看得到的传统机器的属性，包括指令集、 数据类型、存储器寻址技术等，大都属于抽象的属性。

计算机组成是指如何实现计算机体系结构所体现的属性，它包含许多对程序员来说透明的硬 件细节。例如，指令系统属于结构的问题，但指令的实现即如何取指令、分析指令、取操作数、 如何运算等都属于组成的问题。因此，当两台机器的指令系统相同时，只能认为它们具有相同的 结构，至于这两台机器如何实现其指令，则完全可以不同，即可以认为它们的组成方式是不同的。 例如， 一台机器是否具备乘法指令是一个结构的问题，但实现乘法指令采用什么方式则是一个组

第1章 计算机系统概述 一 25

成的问题。许多计算机厂商提供一系列体系结构相同的计算机，而它们的组成却有相当大的差别， 即使是同一系列的不同型号机器，其性能和价格差异也很大。

5. 基准程序执行得越快说明机器的性能越好吗?

一般情况下，基准测试程序能够反映机器性能的好坏。但是，由于基准程序中的语句存在频 度的差异，因此运行结果并不能完全说明问题。

|  |
| --- |
| 第 2 章  数据的表示和运算 |

**【考纲内容】**

(一)数制与编码

进位计数制及其相互转换；定点数的编码表示

(二)运算方法和运算电路

基本运算部件：加法器；算术逻辑单元 (ALU)

加/减运算：补码加/减运算器；标志位的生成

乘/除运算：乘/除法运算的基本原理；乘法电路和除法电路的基本结构 (三)整数的表示和运算

无符号整数的表示和运算；有符号整数的表示和运算

(四)浮点数的表示和运算

浮点数的表示： IEEE754 标准；浮点数的加/减运算

**【复习提示】**

扫一扫



视频讲解

本章内容较为繁杂，由于计算机中数的表示和运算方法与人们日常生活中的表示和运算方法不 同，因此理解也较为困难。纵观历年统考真题，不难发现 unsigned 、short 、int 、long 、float 、double 等在C 语言中的表示、运算、溢出判断、隐式类型转换、强制类型转换、IEEE754 浮点数的表示， 以及浮点数的运算，都是考研考查的重点，需要牢固掌握。

在学习本章时，请读者思考以下问题：

1)在计算机中，为什么要采用二进制来表示数据?

2)计算机在字长足够的情况下能够精确地表示每个数吗?若不能，请举例说明。

3)字长相同的情况下，浮点数和定点数的表示范围与精度有什么区别?

4)用移码表示浮点数的阶码有什么好处?

请读者在本章的学习过程中寻找答案，本章末尾会给出参考答案。

**2.1** **数制与编码**

**2.1.1** **进位计数制及其相互转换**

命 题 追 踪 ▶ **采用二进制编码的原因(2018)**

在计算机系统内部，所有信息都是用二进制进行编码的，这样做的原因有以下几点。

1)二进制只有两种状态，使用有两个稳定状态的物理器件就可以表示二进制数的每一位，

第2章 数据的表示和运算 27

制造成本比较低，例如用高低电平或电荷的正负极性都可以很方便地表示0和1。

2)二进制位1和0正好与逻辑值“真”和“假”相对应，为计算机实现逻辑运算和程序中

的逻辑判断提供了便利条件。

3)二进制的编码和运算规则都很简单，通过逻辑门电路能方便地实现算术运算。 1. 进位计数法

常用的进位计数法有十进制、二进制、八进制、十六进制等。十进制数是日常生活中最常 使用的，而计算机中通常使用二进制数、八进制数和十六进制数。

在进位计数法中，每个数位所用到的不同数码的个数称为基数。十进制的基数为10(0～9), 每个数位计满10就向高位进位，即“逢十进一”。十进制数101,其个位的1显然与百位的1所 表示的数值是不同的。每个数码所表示的数值等于该数码本身乘以一个与它所在数位有关的常 数，这个常数称为位权。一个进位数的数值大小就是它的各位数码按权相加。

一个r进制数(K ,K, …K K\_…Km) 的数值可表示为



式 中 ，r 是基数； √是第i位的位权；K,的取值可以是0,1,…,r-1 共r个数码中的任意一个。

1)二进制。计算机中用得最多的是基数为2的计数制，即二进制。二进制只有0和1两种

数码，计数“逢二进一”。它的任意数位的权为2',i 为所在位数。

2)八进制。基数为8,有0～7共8个不同的数码。计数逢八进一。因为 r=8=2³, 所以把

二进制中的3位数码编为一组就是1位八进制数码，两者之间的转换极为方便。

3)十六进制。基数为16,有0～9、A～F 共16个不同的数码，其中A～F 分别表示10～15。

计数逢十六进一。因为r=16=2⁴, 所以4位二进制数码与1位十六进制数码相对应。

可以用后缀字母标识一个数的进位计数制，用B 表示二进制，用O 表示八进制，用D 表示十

进制(通常直接省略),用H 表示十六进制，有时也用前缀Ox表示十六进制数。

2. 不同进制数之间的相互转换

(1)二进制数转换为八进制数和十六进制数

对于一个二进制混合数(既包含整数部分，又包含小数部分),在转换时应以小数点为界。 其整数部分，从小数点开始往左数，将一串二进制数分为3位(八进制)一组或4位(十六进制) 一组，在数的最左边可根据需要加“0”补齐；对于小数部分，从小数点开始往右数，也将一串 二进制数分为3位一组或4位一组，在数的最右边也可根据需要加“0”补齐。最终使总的位数 为3或4的整数倍，然后分别用对应的八进制数或十六进制数取代。

【例2.1】将二进制数1111000010.01101 分别转换为八进制数和十六进制数。 解：



所以，对应的八进制数为(1702.32)g=(1111000010.01101)。

高位补0,凑足4位

分界点 低位补0,凑足4位

↓ \*



所以，对应的十六进制数为(3C2.68)₆=(1111000010.01101)₂。



28 -2025年计算机组成原理考研复习指导

同样，由八进制数或十六进制数转换为二进制数，只需将每位改为3位或4位二进制数即可 (必要时去掉整数最高位或小数最低位的0)。八进制数和十六进制数之间的转换也能方便地实现， 十六进制数转换为八进制数(或八进制数转换为十六进制数)时，先将十六进制(八进制)数转 换为二进制数，然后由二进制数转换为八进制(十六进制)数较方便。

(2)任意进制数转换为十进制数

将任意进制数的各位数码与它们的权值相乘，再把乘积相加，就得到了 一 个十进制数。这种 方 法 称 为按权展开相加法。例如，(11011.1)₂=1×2⁴+1×2³+0×2²+1×2¹+1×2⁰+1×2-¹=27.5。

(3)十进制数转换为任意进制数

▶ **十进制小数转换为二进制小数(2021、2022)**

一 个十进制数转换为任意进制数，通常采用基 数 乘 除 法。这种转换方法对十进制数的整数部 分和小数部分将分别进行处理，对整数部分采用除 基 取 余 法，对小数部分采用乘基取整法，最后 将整数部分与小数部分的转换结果拼接起来。

**【例2** **.** **2】**将十进制数123 . 6875转换成二进制数。

解 ：

除 基 取 余 法 (整数部分):整数部分除基取余，最先取得的余数为数的最低位，最后取得的 余 数 为 数 的 最 高 位 ( 即除基取余，先余为低，后余为高) , 商 为 0 时 结 束 。

整数部分：



取余

1 最低位

0

最高位

除基

23

61

3

15

3

0

2

2

2

因此整数部分123=(1111011)2。

乘 基 取 整 法 (小数部分):小数部分乘基取整，最先取得的整数为数的最高位，最后取得的 整数为数的最低位(即乘基取整，先整为高，后整为低),乘积为1 . 0(或满足精度要求)时结束。

小数部分：

乘基 取整

0.6875

× .. 最高位

X

1.5000

0.5000

× 2

10000 最低位

因此小数部分0.6875=(0.1011)₂,所以123.6875=(1111011.1011)2。



**注** **意**

关于十进制数转换为任意进制数为何采用除基取余法和乘基取整法，以及所取之数放置位 置的原理，请结合r 进制数的数值表示公式思考，而不应死记硬背。

第2章 数据的表示和运算- 29



注 意

在计算机中，小数和整数不一样，整数可以连续表示，但小数是离散的，所以并不是每个 十进制小数都可以准确地用二进制表示。例如0.3,无论经过多少次乘二取整转换都无法得到 精确的结果。但任意一个二进制小数都可以用十进制小数表示，希望读者引起重视。

2.1.2 定点数的编码表示

1. 真值和机器数

在日常生活中，通常用正号、负号来分别表示正数(正号可省略)和负数，如+15、 -8等。 这种带“+”或“-”符号的数称为真值。 真值是机器数所代表的实际值。

在计算机中，通常将数的符号和数值部分一起编码，将数据的符号数字化，通常用“0”表 示“正”,用“1”表示“负”。这种把符号“数字化”的数称为机器数。常用的有原码、补码和 反码表示法。如0,101(这里的逗号“,”仅为区分符号位与数值位)表示+5。

2. 机器数的定点表示

根据小数点的位置是否固定，在计算机中有两种数据格式：定点表示和浮点表示。在现代计 算机中，通常用补码整数表示整数，用原码小数表示浮点数的尾数部分，用移码表示浮点数的阶 码部分， 历年统考真题的命题信息也主要落在这个范畴之内。

定点表示法用来表示定点小数和定点整数。

1)定点小数。定点小数是纯小数，约定小数点位置在符号位之后、有效数值部分最高位之 前。若数据X 的形式为X=x₀x₁x₂…x, (其中x₀ 为符号位， x₁~x, 是数值的有效部分，也 称尾数，x₁ 为最高有效位),则在计算机中的表示形式如图2.1所示。

2)定点整数。定点整数是纯整数，约定小数点位置在有效数值部分最低位之后。若数据 X 的形式为X=xox₁x₂…x₂ (其中x₀ 为符号位， x₁~x₁,是尾数，x, 为最低有效位),则在计 算机中的表示形式如图2.2所示。

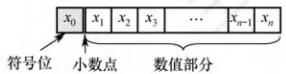


图2.1 定点小数表示



图2.2 定点整数表示

事实上，在机器内部并没有小数点，只是人为约定了小数点的位置。因此，在定点数的编码 和运算中不用考虑对应的定点数是小数还是整数，而只需关心它们的符号位和数值位即可。

定点数的编码表示法主要有以下4种：原码、补码、反码和移码。

3. 原码、补码、反码、移码

(1)原码表示法

用机器数的最高位表示数的符号，其余各位表示数的绝对值。原码的定义如下。



例如，若x₁=+1110,x₂=-1110, 字长为8位，则其原码表示为[x₁]=0,0001110,[x₂]=2⁷+ 1110 =1,0001110,其中最高位是符号位。

若字长为n+1, 则原码整数的表示范围为-(2”-1)≤x≤2"-1 (关于原点对称)。

30 -2025年计算机组成原理考研复习指导



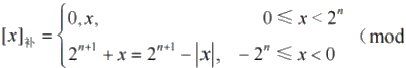
注 意

零的原码表示有正零和负零两种形式，即[+0]=00000000和[-0]x=10000000。

原码表示的优点： ①与真值的对应关系简单、直观，与真值的转换简单；②用原码实现乘除 运算比较简便。缺点：①0的表示不唯一，有±0两个编码；②原码加减运算比较复杂。在原码加 减运算中，对于两个不同符号数的加法(或同符号数的减法),先要比较两个数的绝对值大小， 然后用绝对值大的数减去绝对值小的数，最后还要为结果选择合适的符号。

(2)补码表示法

补码表示法中的加减运算统一采用加法操作实现。正数的补码和原码相同，负数的补码等于 模 (n+1 位补码的模为2\*\*I) 与该负数绝对值之差。补码的定义如下：

2\*)

综合上述定义可知，无论是正数还是负数，[x]\*=2"\*¹+x(-2°≤x<2”,mod 2"I)。

例如，若x₁=+1010,x₂=-1101, 字长为8位，则其补码表示为[x₁]\*=0,0001010,[x₂]n=2⁸- 0,0001101=1,1110011。

命题追踪▶补码的表示范围(2010、2013、2014、2022)

若字长为n+1, 则补码整数的表示范围为-2°≤x≤2”-1 (比原码多表示“-2””)。

●几个特殊数据的补码表示

1)[+0]=[-0]\*=0,00.…0(含符号位共n+1 个0),说明0的补码表示是唯一的。 2)[-1]\*=2"\*I-1=1,11…1 (含符号位共n+1 个1)。

3)[2”-1]\*=0,11…1(n 个1),即n+1 位补码能表示的最大整数。

4)[-2"]\*=1,00…0(n 个0),即n+1 位补码能表示的最小整数。

●模运算

在模运算中， 一个数与它除以“模”后得到的余数是等价的，如A、B、M满足A=B+K×M

(K 为整数),则记为A=B(modM), 即 A、B 各除以M 后的余数相同。在补码运算中，[A]\*-

[B]\*=[A]n+M-[B]N, 而 M-[B]\*=[-B]\*, 因此补码可以借助加法运算来实现减法运算。

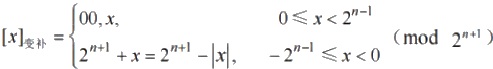
●补码与真值之间的转换

补码和真值的相互转换(2020、2023)

真值转换为补码 ：对于正数，与原码的方式一样。对于负数，符号位取1,其余各位由真 值“各位取反，末位加1”得到。 补码转换为真值：若符号位为0,与原码的方式一样。若符 号位为1,真值的符号为负，数值部分各位由补码“各位取反，末位加1”得到。

●变形补码

变形补码是一种采用双符号位的补码表示，也称模4补码。假定变形补码的位数为n+1 ( 其 中符号位占2位，数值位占n-1 位),则整数变形补码的表示为



模4补码双符号位00 表示正，11 表示负，用在执行算术运算的ALU 中。

第2章 数据的表示和运算 31

(3)反码表示法(了解)

负数的补码可采用“各位取反，末位加1”的方法得到，若仅各位求反而末尾不加1,则是 负数的反码表示，因此负数反码的定义就是在相应的补码表示中再末位减1。

正数反码的定义和相应的补码(或原码)表示相同。

反码表示存在以下几个方面的不足：①0的表示不唯一(即存在±0);②表示范围比补码少 一个最小负数。反码在计算机中很少使用，通常用作数码变换的中间表示形式。

(4)移码表示法

移码常用来表示浮点数的阶码，它只能表示整数。

移码就是在真值X 上加上一个常数(偏置值),通常这个常数取2”,相当于X 在数轴上向正 方向偏移了若干单位，这就是“移码”一词的由来。移码的定义如下。

[x]\*=2"+x(-2"≤x<2”, 其中机器字长为n+1)

例如，若正数x₁=+10101,x₂=-10101, 字长为8位，则其移码表示为[x;]=2⁷+10101=1, 0010101;[x₂]w=2⁷+(-10101)=0,1101011。

移码具有以下特点：

①移码中零的表示唯一，[+0]=2”+0=[-0]=2"-0=100.0 (n 个“0”)。

②一个真值的移码和补码仅差一个符号位， [x] 的符号位取反即得[x](“1”表示正，“0”

表示负，这与其他机器数的符号位取值正好相反),反之亦然。

③移码全0时，对应真值的最小值-2”;移码全1时，对应真值的最大值2"-1。 ④移码保持了数据原有的大小顺序，移码大真值就大，移码小真值就小。

原码、补码、反码和移码这4种编码表示的总结如下：

命题追踪▶补码大小的判断(2015)

①原码、补码、反码的符号位相同，正数的机器码相同。

②原码、反码的表示在数轴上对称，二者都存在+0和-0两个0。

③补码、移码的表示在数轴上不对称，零的表示唯一，它们比原码、反码多表示一个数。 ④整数的补码、移码的符号位相反，数值位相同。

⑤负数的补码、反码末位相差1。

⑥原码很容易判断大小。而负数的补码、反码很难直接判断大小，可采用如下规则快速判 断：对于负数，数值位部分越小，其绝对值越大，即负得越多。

2.1.3 整数的表示

1. 无符号整数的表示

命题追踪▶机器码与补码、无符号数之间的转换(2021)

当一个编码的全部二进制位均为数值位而没有符号位时，该编码表示就是无符号整数，简称 无符号数。此时，默认数的符号为正。因为无符号整数省略了一位符号位，所以在字长相同的情 况下，它能表示的最大数比有符号整数能表示的大。 一般在全部是正数运算且不出现负值结果的 场合下，使用无符号整数表示。例如，可用无符号整数进行地址运算，或用它来表示指针。

例如，对于8位无符号整数，最小数为00000000(值为0),最大数为11111111(值为28-1= 255),即表示范围为0～255;而对于8位有符号整数，最小数为10000000(值为-2⁷=-128),最 大数为01111111(值为2?-1=127),即表示范围为-128～127。

32-2025年计算机组成原理考研复习指导

**2.** **有符号整数的表示**

将符号数值化，并将符号位放在有效数字的前面，就组成了有符号整数。虽然前面介绍的原 码、补码、反码和移码都可以用来表示有符号整数，但补码表示有其明显的优势：

① 与原码和反码相比，0的补码表示唯一。

② 与原码和移码相比，补码运算规则比较简单，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，补码比原码和反码多表示一个最小负数。

计算机中的有符号整数都用补码表示，故n 位有符号整数的表示范围是-2"-1～2"-1-1。

**2.1.4** **C 语** **言** **中** **的** **整** **数** **类** **型** **及** **类** **型** **转** **换**

统考大纲要求考生具有对高级程序设计语言( 如 C 语言)中相关问题进行分析的能力，而C 语言变量之间的类型转换是统考中经常出现的题目，需要读者深入掌握这一内容。

**1.C** **语言中的整型数据类型**

▶ **int** **型数据的表示范围(2017、2019)**

C 语言中的整型数据就是定点整数，根据位数的不同，可分为字符型( char,8 位)、短整型 (short 或 short int,16位)、整型 (int,32 位)、长整型 (long 或 long int, 在32位机器中为32 位，在64位机器中为64位)。char 是整型数据中比较特殊的一种，其他如 shor/intlong 等不指定 signed/unsigned 时都默认是有符号整数，但 char 默认是无符号整数。无符号整数 (unsigned short/intlong) 的全部二进制位均为数值位，没有符号位，相当于数的绝对值。

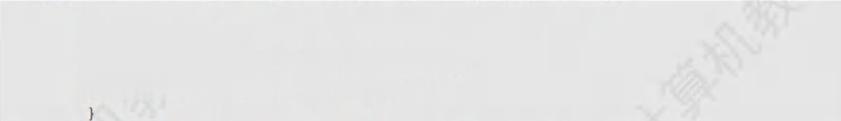
signed/unsigned 整型数据都是按补码形式存储的，只是 signed 型的最高位代表符号位，而在 unsigned 型中表示数值位，因此这两者所表示的数据范围也有所不同。

**2.** **有符号数和无符号数的转换**

**有符号数与无符号数的相互转换(2011、2016、2019)**

C 语言允许在不同的数据类型之间做类型转换。强制类型转换格式为“TYPEb=(TYPE)a”, 强制类型转换后，返回一个具有TYPE 类型的数值，这种操作并不会改变操作数本身。

先看由short 型转换到 unsigned short 型的情况。考虑如下代码片段：



int main(){

short x=-4321;

"x=y=8yu=\(; short)x;

有符号数x 是一个负数，而无符号数y 的表示范围显然不包括x 的 值。

在采用补码的机器上，上述程序会输出如下结果：

x=-4321,y =61215

输出的结果中，得到的y 值似乎与原来的x 没有一点关系。不过将这两个数转换为二进制表 示时，我们就会发现其中的规律，如表2.1所示。

**表2.1** **y** **与x** **的对比**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 变量 | 值 | 位 | | | | | | | | | | | | | | | |
|  |  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 |  | 0 |
| X | -4321 | 1 | 1 | 1 | 0 | 1 | 1 |  | 1 | 0 | 0 | 0 |  | 1 | 1 |  | 1 |
| y | 61215 | 1 |  | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |  | 1 | 1 |  |  |

**第2章** **数据的表示和运算** 33

观察可知，将 short 型强制转换为unsigned short 型只改变数值，而两个变量对应的每位都是

一样的。通过本例可知， 强制类型转换的结果是保持位值不变，仅改变了解释这些位的方式。



再看由unsigned short 型转换到 short 型的情况。考虑如下代码片段：

int main(){

unsigned short x=65535;

short y=(short)x;

printf("x=%u,y=8d\n",x,y);

}

同样在采用补码的机器上，上述程序会输出如下结果：

x=65535,y=-1

把这两个数转换为二进制表示，同样可以证实之前的结论。因此，有符号数转换为等长的无 符号数时，符号位解释为数值的一部分，负数转换为无符号数时数值将发生变化。同理，无符号 数转换为有符号数时最高位解释为符号位，也可能发生数值的变化。



**注** **意**

若同时有无符号数和有符号数参与运算，则C 语言标准规定按无符号数进行运算。

**3.** **不同字长整数之间的转换**

▶ **无符号数的零扩展(2012)**

另一种常见的运算是在不同字长的整数之间进行类型转换。

先看大字长变量向小字长变量转换的情况。考虑如下代码片段：



int main(){

rt xhort)u;

printf("x=8d,y=8d\n",x,y);

printf("u=8d,v=8d\n",u,v);

}

//int 型占用4B

//short 型占用2B

运行结果如下：

x=165537,y=-31071

=30545

u=-34991,v

其中x,y,u,v 的十六进制表示分别为0x000286al,Ox86al,OxfMr7751,0x7751, 观察上述数字很容 易得出结论，当大字长变量向小字长变量强制类型转换时，系统把多余的高位部分直接截断，低 位部分直接赋值，因此也是一种保持位值的处理方法。

再看小字长变量向大字长变量转换的情况。考虑如下代码片段：



int main(){

short x=-4321;

int y=x;

unsigned short u=(unsigned short)x;

unsigned int v=u;

printf("x=8d,y=8d\n",x,y);

printf("u=8u,v=gu\n",u,v);

运行结果如下：



x=-4321,y =-4321

u =61215,v =61215

34 -2025年计算机组成原理考研复习指导

命题追踪 ▶ **无符号数的零扩展(2012),补码的符号扩展(2021)**

x,y,u,v 的十六进制表示分别为Oxeflf,Oxffffeflf,Oxeflf,0x0000eflf。 由本例可知，小字长 到大字长的转换时，不仅要使相应的位值相等，还要对高位部分进行扩展。若原数字是无符号 整数，则进行零扩展，扩展后的高位部分用0填充。否则进行符号扩展，扩展后的高位部分用 原数字符号位填充。其实两种方式扩展的高位部分都可理解为原数字的符号位。这与之前的三 个例子都不一样，从位值与数值的角度看，前三个例子的转换规则都是保证相应的位值相等， 而小字长向大字长的转换，在位值相等的条件下还要补充高位的符号位，可以理解为数值的相 等。注意， char 型为8位无符号整数，其在转换为int 型时高位补0即可。

**2.1.5** **本节习题精选**

**单项选择题**

**01.** 若十进制数为137.5,则其八进制数为( )。

A.89.8 B.211.4 C.211.5 D.1011111.101 **02.** 一个16位无符号二进制数的表示范围是()。

A.0～65536 B.0～65535

C.-32768～32767 D.-32768～32768

**03.** 下列说法有误的是()。

A. 任何二进制整数都可以用十进制表示

B. 任何二进制小数都可以用十进制表示

C. 任何十进制整数都可以用二进制表示

D. 任何十进制小数都可以用二进制表示

**04.** 对真值0表示形式唯一的机器数是()。

A. 原码 B. 补码和移码 C. 反码 D. 以上都不对

**05.** 若[X]\*=1.1101010, 则[X]x=()。

A.1.0010101 B.1.0010110 C.0.0010110 D.0.1101010

**06.** 若 X 为负数，则由[X]求 [ -X]是 将 ( ) 。

A.[X]. 各值保持不变

B.[X]\* 符号位变反，其他各位不变

C.[X]. 除符号位外，各位变反，末位加1

D.[X]4 连同符号位一起变反，末位加1

**07.** 8位原码能表示的不同数据有()个。

A.15 B.16 C.255 D.256

**08** .一 个 n+1 位整数x 原码的数值范围是()。

A.-2"+1<x<2"-1 B.-2"+1≤x<2"-1

C.-2"+1<x≤2"-1 D.-2"+1≤x≤2"-1

**09.** n 位定点整数(有符号)表示的最大值是()。

A.2" B.2"-1 C.2m-1 D.2"-¹-1

**10.** 对于相同位数(设为N 位，不考虑符号位)的二进制补码小数和十进制小数，二进制小 数能表示的数的个数/十进制小数所能表示数的个数为()。

A.(0.2)^ B.(0.2)^~| C.(0.02)\* D.(0.02)\*~1

**11.** 若定点整数为64位，含1位符号位，则采用补码表示的绝对值最大的负数为( )。

**第2章** **数据的表示和运算** - 35

A.-264 B.-(2⁶⁴-1) C.-263 D.-(2⁶³-1)

**12.** 下列关于补码和移码关系的叙述中，()是不正确的。

A. 相同位数的补码和移码表示具有相同的数据表示范围

B.0 的补码和移码表示相同

C. 同一个数的补码和移码表示，其数值部分相同，而符号相反

D. 一般用移码表示浮点数的阶码，而补码表示定点整数

**13.** 若 [x]\*=1,x₁ X₂X₂x₄X₂N₆, 其 中x; 取 0 或 1 , 若 要x>-32, 应 当 满 足 ( ) 。

A.x₁ 为0,其他各位任意 B.x₁ 为1,其他各位任意

C.x₁ 为 1 ,x₂…x₆ 中至少有一位为1 D.x₁ 为 0 , x₂…x₆ 中至少有一位为1

**14.** 设 x 为整数，[x]\*=1,x₁x₂xyx₄Xs, 若 要x<-16,x₁~xs 应满足的条件是()。

A.x₁~xs 至少有一个为1 B.x₁ 必 须 为 0 ,x₂~xs 至少有一个为1

C.x₁ 必须为0, x₂~xs 任意 D.x₁ 必 须 为 1 ,x₂~xs 任意

**15.** 设 x 为真值， x\* 为其绝对值，满足[-x\*]=[-x], 当且仅当()。

A.x 任意 B.x 为正数 C.x 为负数 D. 以上说法都不对

**16.** 假定一个十进制数为-66,按补码形式存放在一个8位寄存器中，该寄存器的内容用十 六进制表示为( )。

A.C2H B.BEH C.BDH D.42H

**17** . 设机器数采用补码表示(含1位符号位),若寄存器内容为9BH, 则对应的十进制数为 ()。

A.-27 B.-97 C.-101 D. 155

**18.** 若寄存器内容为10000000,若它等于-0,则为()。

A. 原 码 B. 补码 C. 反码 D. 移 码

**19.** 若寄存器内容为11111111,若它等于+127,则为()。

A. 反 码 B. 补码 C. 原 码 D. 移 码

**20.** 若寄存器内容为11111111,若它等于-1,则为()。

A. 原 码 B. 补 码 C. 反 码 D. 移码

**21.** 若寄存器内容为00000000,若它等于- 128,则为()。

A. 原 码 B. 补码 C. 反码 D. 移码

**22.** 若二进制定点小数真值是-0.1101,机器表示为1.0010,则为( )。

A. 原 码 B. 补码 C. 反码 D. 移 码

**23.** 下列为8位移码机器数[x], 求[-x]n 时，()将会发生溢出。

A.11111111 B.00000000 C.10000000 D.01111111

**24.** 一个8位的二进制整数由2个“0”和6个“1”组成，采用补码或者移码表示，则下列 说法中正确的是( )。

A. 若采用移码表示，偏置值为127,则此整数最小为-64

B. 若采用移码表示，偏置值为128,则此整数最大为123

C. 若采用补码表示，则此整数最小为-96

**D.** 若采用补码表示，则此整数最大为252

**25.** 计算机内部的定点数大多用补码表示，以下是一些关于补码特点的叙述： I. 零的表示是唯一的 II. 符号位可以和数值部分一起参加运算

HI. 和其真值的对应关系简单、直观 IV. 减法可用加法来实现

36 -2025年计算机组成原理考研复习指导

在以上叙述中，()是补码表示的特点。

A.I 和 I] B.I 和 Ⅲ C.I 和 Ⅱ 和 Ⅲ D.I 和Ⅱ和IV

**26.** 在计算机中，通常用来表示主存地址的是( ).

A. 移码 B. 补码 C. 原 码 D. 无符号数

**27.** 16位补码Ox8FA0 扩展为32位应该是( )。

A.0x00008FA0 B. OxFFFF 8FA0 C. OxFFFF FFA0 D.0x80008FA0

**28.** 【2012统考真题】假定编译器规定 int 型和 short 型长度分别为32位和16位，执行下列

hort x=65530;

unsigned int y=x;

得到y 的机器数为()。

A.00007FFAH B.0000 FFFAH C.FFFF 7FFAH D.FFFF FFFAH

**29.** 【2015统考真题】由3个“1”和5个“0”组成的8位二进制补码，能表示的最小整数

是 ( ) 。

A.-126 B.-125 C.-32 D.-3

**30.** t6统 题-37下C 语言程序段：

unsigned short usi =si;

执行上述两条语句后， usi的值为( )。

A.-32767 B.32767 C.32768 D.32769

**31.【2**018统考真题】冯·诺依曼结构计算机中的数据采用二进制编码表示，其主要原因是( )。

I. 二进制的运算规则简单 IⅡ. 制造两个稳态的物理器件较容易

Ⅲ.便于用逻辑门电路实现算术运算

A. 仅 I 、Ⅱ B. 仅 I 、Ⅲ C. 仅Ⅱ、Ⅲ D.I 、Ⅱ 和 Ⅲ

**32.** 代码：

short si =usi;

执行上述程序段后，si 的值是( )。

A.-1 B.-32767 C.-32768 D.-65535

**33.【2** 021 统考真题】已知有符号整数用补码表示，变量x,y,= 的机器数分别为 FFFDH,

FFDFH,7FFCH, 下列结论中，正确的是()。

A. 若x,y 和=为无符号整数，则=<x<y

B. 若 x,y 和=为无符号整数，则x<y<z

C. 若 x,y 和=为有符号整数，则x<y<z

D. 若 x,y 和=为有符号整数，则y<x<z

**34.** 【2022统考真题】32位补码所能表示的整数范围是()。

*A.-2³2~23¹-1* B.-2³¹~2³¹-1 C.-2²~2¹²-1 D.-2³¹~2²-1

**2.1.6** **答案与解析**

**单项选择题**

**01.B**

十进制数转换成八进制数，整数部分采用除基取余法：将整数除以8,所得余数即为转换后

第2章 数据的表示和运算- 37

的八进制数的个位数码，再将商除以8,余数为八进制数十位上的数码，如此反复进行，直到商 是0为止。小数部分采用乘基取整法：将小数乘以8,所得积的整数部分即为八进制数十分位上 的数码，再将此积的小数部分乘以8,得到百分位上的数码，如此反复直到积是1.0为止。经转 换得到的八进制数为211.40。

02.B

一个16位无符号二进制数的表示范围是0～2l⁶-1, 即0～65535。

03.D

A 、B 、C 明显正确，二进制整数和十进制整数可以相互转换，仅仅是每位的位权不同而已。 而二进制的小数位只能表示1/2,1/4,1/8,…,1/2”,因此无法表示所有的十进制小数， D 错误。

04.B

假设位数为5位(含1位符号位),[+0]=00000,[-0]=10000,[+0]x=00000,[-0]=11111, [+0]\*=[-0]\*=00000,[+0]=[-0]s=10000 。 可知，0的补码和移码的表示是唯一的。

05.B

若X 为负数，则其补码转换成原码的规则是“符号位不变，数值位取反，末位加1”,即[X]w= 0010101+1=0010110。

06.D

不论X 是正数还是负数，由[X] 求[-X]\*的方法是连同符号位一起，每位取反，末位加1。 07.C

8个二进制位有2⁸=256种不同表示。原码中0有两种表示，因此原码能表示的不同数据为 2⁸-1=255个。由于0在反码中也有两种表示，因此若题目改为反码，答案也为 C 。0 在补码与 移码中只有一种表示，因此题目若改为补码或移码，答案为D。

08.D

n+1 位整数原码的表示范围为-2"+1≤x≤2"-1。

09.D

n 位二进制有符号定点整数，数值位只有n-1 位最高位为符号位，所以最大值为2-1-1。

10.A

N位二进制小数共有2\*种状态，每种状态都能表示一个不同的小数，而十进制小数能表示的 数的个数为10\*,二者的商为(0.2)^。这也是为何在计算机的运算中会出现误差情况的原因，它表 明仅有(0.2)\*概率的十进制数可以精确地用二进制表示。

11.C

对于长度为n+1 (含1位符号位)定点整数x, 用补码表示时， x 他最大数=-2”,这里n=63。

12.B

以机器字长5位为例，[0]=00000,[0]=2⁴+0=10000,[0]≠[0],表示不相同，但在补 码或移码中的表示形式是唯一的。

13.C

对于此类题型，先写出特定值的机器码表示，然后根据机器数判断大小的规则来推导数值位 的特点(若条件允许，也可以取特殊值来推断)。 -32的补码为1,100000,根据负数补码判断大小 的规则：数值位部分越小，其绝对值越大，即负得越多。因此，若要x>-32, 数值位x₁FX₃X₄F₅X₆ 需大于100000,即x₁ 必须为1,而x₂…x₆ 中至少有一位为1。

【特殊值法】对于 A, 取1,000000,真值为-64,错误。对于B, 取1,100000,真值为-32, 错误。对于C, 取1,100001,真值为-31,符合。对于D, 取1,000001,真值为-63,错误。

38 -2025 年计算机组成原理考研复习指导

14.C

解题思路与上题类似(也可以采用特殊值解法，请读者自行思考),-16的补码为1,10000, 根据负数补码判断大小的规则：数值位部分越小，其绝对值越大，即负得越多。因此，若要x<-16, 数值位x₁x₂x₃x₄xs需小于10000,即x₁ 必为0,而x₂~xs任意。

15.D

当x 为0或为正数时，满足[-x\*]=[-x],B 为充分条件，因此B 错误。而x 为负数时， -x

为正数，而-x\*为负数，补码的表示是唯一的，显然二者不等，因此C 错误。

16.B

x=-66 用二进制表示， [x]x=11000010, 则有[x]=10111110=BEH。

17.C

9BH=(10011011)₂, 最高位的1表示负数，故其真值为(11100101)₂=-(64+32+4+1)=-101。

18.A

值等于-0说明只可能是原码或反码(因为补码和移码表示0时是唯一的，没有+0和-0之分), [-0]\*=10000000,[-0]=11111111。

19.D

这里寄存器长度为8,[+127]x=[+127]=[+127]\*= 01111111,又知同一数值的移码和补码除 最高位相反外，其他各位相同，则[+127]=11111111或[+127]=2⁷+01111111=11111111。

20.B

这里寄存器长度为8,[-1]=[10000001]=11111111。

21.D

这里寄存器长度为8,[-128]=2⁷+(-10000000)=00000000。

22.C

真值-0.1101,对应的原码表示为1.1101,补码表示为1.0011,反码表示为1.0010,移码通常 用于表示阶码，不用来表示定点小数。

23.B

选项B 对应8位最小的值-128,而-x=128 发生溢出，因此无法表示其移码。

24.A

若采用补码表示，要使得数值最大，就要让符号位为0,且把“1”放在高位，得到的补码 为01111110B=126; 要使得数值最小，就要让符号位为1,且把“1”放在低位，得到的补码 为10011111B=-97 。 若采用移码表示，偏置值为128时，要使得数值最大，就要把“1”放在高 位，得到的移码为11111100B-10000000B=252-128=124; 偏置值为127时，要使得数值最 小，则应把“1”放在低位，得到的移码为00111111B-01111111B=11000000B=-64,A 正确。

25.D

[+0]n和[-0]\*是相同的，所以1正确。在进行补码定点数的加减运算时，符号作为数的一部分 参加运算，Ⅱ正确，[A]-[B]=[A]\*+[-B], 即将减法采用加法实现， IV 正确。实际上，补码和 其真值的对应关系远不如原码和其真值的对应关系简单直观，Ⅲ错误。

26.D

主存地址都是正数，因此不需要符号位，因此直接采用无符号数表示。

27.B

16位扩展为32位，符号位不变，附加位是符号位的扩展。该数是一个负数，需用1来填 补。A 是一个正数， C 的数值位发生变化， D 用0来填充附加位，均不正确。

**2.2**

第2章 数据的表示和运算 39

**28.B**

将 一个16位 unsigned short型数转换成32位unsigned int 型数时，因为都是无符号数，新表 示形式的高位用0填充。16位无符号整数所能表示的最大值为65535,其十六进制表示为FFFFH,

因此x 的十六进制表示为FFFFH-5H=FFFAH, 所 以y 的十六进制表示为0000 FFFAH。

排除法：先直接排除 C 、D, 然后分析余下选项的特征。由 于A 、B 的值相差几乎近1倍， 因此可以算出00010000H ( 接 近B 且好算的数)的值后，再推断出答案。

**29.B**

原码很容易判断大小。而负数的补码很难直接判断大小，可采用如下规则快速判断：对于 负数，数值位部分越小，其绝对值越大，即负得越多。采用补码整数表示时，负数的符号位为 1,因此剩下的两个“1”放在末位时其值最小，补码形式为10000011,转换为真值为-125。此 外，考虑负数的补码转换为原码的方法，从右向左找到第一个数值为1的位，之后的每位进行 取反操作，符号位不变，不难发现，当符号位为1,剩下的两个“1”放在末位时，补码的绝对 值最大。

**30.D**

因 C 语言中的数据在内存中为补码表示形式， si 对应的补码二进制表示为100000000000 0001B, 最前面的一位“1”为符号位，表示负数，即-32767。由signed 型转换为等长的 unsigned 型数据时，符号位成为数据的一部分，即负数转换为无符号数(即正数)时，其数值将发生变化。 usi 对应的补码二进制表示与 si 的表示相同，但表示正数，为32769。

**31.D**

对于I, 二进制只有0和1两种数值，运算规则较简单，都通过ALU 转换成加法运算。对于 Ⅱ,二进制只需要高电平和低电平两个状态就可表示，这样的物理器件很容易制造。对于Ⅲ,二 进制与逻辑量相吻合。二进制的0和1正好与逻辑量的“真”和“假”相对应，因此用二进制数 表示二值逻辑显得十分自然，采用逻辑门电路很容易实现运算。

**32.** A

unsigned short 型为无符号短整型，长度为2字节，因此 unsigned short usi 型转换为二进制代 码即1111111111111111。short型为短整型，长度为2字节，在采用补码的机器上，short si 的二 进制代码为1111111111111111,因此 si 的值为-1。

**33.D**

若x,y 和=均为无符号整数，则x>y>=,A 和 B 错误。若x,y 和=均为有符号整数，补码

的最高位是符号位，0表示正数，1表示负数，因此=为正数，而x 和 y 为负数。对 于x 和 y 的比

较，数值位取反加1,可知x=-3,y=-33, 故 x>y 。D 正确。

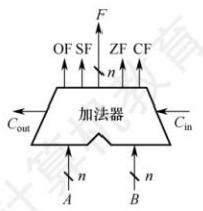
**34.B**

n 位补码整数的最小值是1,00…0(即-2”-1);最大值是0,11…1(即2”-¹-1)。n 位补码整数 所能表示的范围是-2”-1～2”-l-1,32 位补码整数所能表示的范围是-23¹~2³¹-1。

**运算方法和运算电路**

**2.2.1** **基本运算部件**

在计算机中，运算器由算术逻辑单元 (Arithmetic Logic Unit,ALU)、移位器、状态寄存器 (PSW) 和通用寄存器组等组成。运算器的基本功能包括加、减、乘、除四则运算，与、或、非、

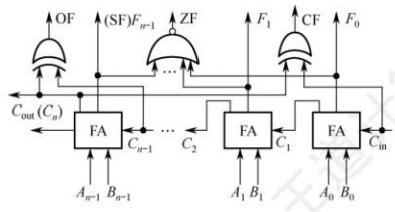


40-2025年计算机组成原理考研复习指导

异或等逻辑运算，以及移位、求补等操作。ALU 的核心部件是加法器。

1. 带标志加法器

无符号数加法器只能用于两个无符号数相加，不能进行有符号整数的加/减运算。为了能进行 有符号整数的加/减运算，还需要在无符号数加法器的基础上增加相应的逻辑门电路，使得加法器 不仅能计算和/差，还要能生成相应的标志信息。图2.3是带标志加法器的实现电路。



(b) 带标志加法器的逻辑电路

(a) 带标志加法器符号

图2.3 用全加器实现n 位带标志加法器的电路

在图2.3中，溢出标志的逻辑表达式为OF =C,@C-1;符号标志就是和的符号，即SF=F-1; 零标志ZF=1 当且仅当F=0; 进位/借位标志CF=Cou④Cm。

2. 算术逻辑单元 (ALU)

ALU 是一种功能较强的组合逻辑电路，它能进行多种算术运算和逻辑运算。由于加、减、乘、 除运算最终都能归结为加法运算，因此ALU 的核心是带标志加法器，同时也能执行“与”“或” “非”等逻辑运算。ALU 的基本结构如图2.4所示，其中A 和 B 是两个n 位操作数输入端，Cm 是进位输入端，ALUop是操作控制端(发出控制信号),用来决定ALU 所执行的处理功能。例如， ALUop 选择 Add运算，ALU就执行加法运算，输出的结果就是A 加 B 之和。ALUop的位数决定 了操作的种类。例如，当位数为3时， ALU 最多只有8种操作。

图2.5给出了能够完成3种运算“与”“或”和“加法”的一位ALU结构图。其中， 一位加 法用一个全加器实现，在ALUop 的控制下，由一个多路选择器(MUX) 选择输出3种操作结果 之一。这里有3种操作，所以ALUop 至少要有两位。

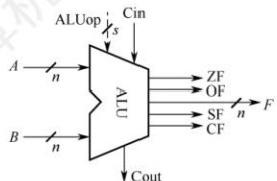


图2.4 ALU 的基本结构

同时，ALU 也可以实现左移或右移的移位操作。

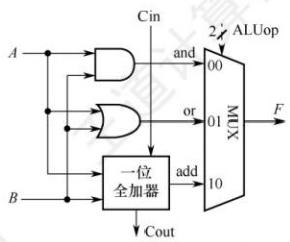


图2.5 一位 ALU 的结构

第2章 数据的表示和运算 41

2.2.2 定点数的移位运算

当计算机中没有乘/除法运算电路时，可以通过加法和移位相结合的方法来实现乘/除法运算。 对于任意二进制整数，左移一位，若不产生溢出，相当于乘以2(与十进制数的左移一位相当于 乘以10类似);右移一位，若不考虑因移出而舍去的末位尾数，相当于除以2。

根据操作数的类型不同，移位运算可以分为逻辑移位和算术移位。

1. 逻辑移位

命题追踪▶逻辑移位运算(2018)

逻辑移位将操作数视为无符号整数。 逻辑移位的规则：左移时，高位移出，低位补0;右移 时，低位移出，高位补0。对于无符号整数的逻辑左移，若高位的1移出，则发生溢出。

2. 算术移位

命题追踪▶算术移位运算(2012、2017、2018)

算术移位需要考虑符号位的问题，即将操作数视为有符号整数。

计算机中的有符号整数都是用补码表示的，因此对于有符号整数的移位操作应采用补码算术 移位方式。 算术移位的规则：左移时，高位移出，低位补0,若移出的高位不同于移位后的符号 位，即左移前后的符号位不同，则发生溢出；右移时，低位移出，高位补符号位，若低位的1移 出，则影响精度。例如，补码1001和0101 左移时会发生溢出，右移时会丢失精度。

2.2.3 定点数的加减运算

1. 补码的加减法运算

命题追踪▶不同字长补码的加法运算(2009)、补码和无符号数的减法运算(2011、2017) 补码加减运算规则简单，易于实现。补码加减运算的公式如下(设机器字长为n+1)。

[A+B]=[A]+[B](mod 2"+1)

[A-B]\*=[A]+[-B]\*(mod 2"+1)

补码运算的特点如下。

1)按二进制运算规则运算，逢二进一。

2)若做加法，两个数的补码直接相加；若做减法，则将被减数与减数的负数补码相加。 3)符号位与数值位一起参与运算，加、减运算结果的符号位也在运算中直接得出。

4)最终运算结果的高位丢弃，保留n+1 位，运算结果亦为补码。

【例2.3】设字长为8位(含1位符号位),A=15,B=24, 求[A+B]\* 和[A-B]。

解：

A=+15=+0001111,B=+24=+0011000; 得[A]=00001111,[B]=00011000。

求得[-B]=11101000 。 所以

[A+B]=00001111+00011000=00100111, 符号位为0,对应真值为+39。

[A-B]\*=[A]\*+[-B]=00001111+11101000=11110111, 符号位为1,对应真值为-9。

2. 溢出判别方法

命题追踪▶补码加减运算后的溢出判断(2010、2011、2014、2018、2021)

仅当两个符号相同的数相加或两个符号相异的数相减才可能产生溢出，如两个正数相加，而结

42 -2025年计算机组成原理考研复习指导

果的符号位却为1(结果为负);一个负数减去一个正数，结果的符号位却为0(结果为正)。 补码定点数加减运算溢出判断的方法有3种。

(1)采用一位符号位

由于减法运算在机器中是用加法器实现的，因此无论是加法还是减法，只要参加操作的两个 数的符号相同，结果又与原操作数的符号不同，则表示结果溢出。

设A 的符号为A,B 的符号为B, 运算结果的符号为S, 则溢出逻辑表达式为

V=ABS+ABS

若 V=0, 表示无溢出；若V=1, 表示有溢出。

(2)采用双符号位

双符号位法也称模 4 补码。运算结果的两个符号位 S₄S₂ 相同，表示未溢出；运算结果的两 个符号位Ss₁S₂ 不同，表示溢出，此时最高位符号位代表真正的符号。

符号位S₁S₂ 的各种情况如下：

①S₅₁S₂=00: 表示结果为正数，无溢出。

②S₄S₂=01: 表示结果正溢出。

③S₃S₂=10: 表示结果负溢出。

④S₅₁S₂=11: 表示结果为负数，无溢出。

溢出逻辑判断表达式为V=Ss₁田 S₂, 若 V=0, 表示无溢出；若V=1, 表示有溢出。 (3)采用一位符号位根据数值位的进位情况判断溢出

若符号位(最高位)的进位 C 与最高数位(次高位)的进位C,-1相同，说明无溢出，否 则说明有溢出。溢出逻辑判断表达式为V=C, 田 C-1, 若 V=0, 表示无溢出； V=1, 表示有 溢出：

3. 加减运算电路

命题追踪▶无符号数与有符号数加/减运算能用同一个加法器实现的理由(2011)

已知一个数的补码表示为 Y, 则这个数的负数的补码为Y+1, 因此，只要在原加法器的Y 输入端加n 个反向器以实现各位取反的功能，然后加一个2选1多路选择器，用一个控制端Sub 来控制，以选择是将Y输入加法器还是将γ输入加法器，并将 Sub 同时作为低位进位送到加法器 (做减法时实现末位加1),如图2.6所示。该电路可实现模2”补码加减运算。当Sub 为1时，做 减法，实现X+Y+1=[x]+[-y]; 当Sub 为0时，做加法，实现X+Y=[x]+[y]。

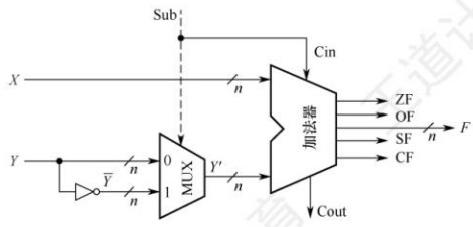


图2.6 加减运算部件

无符号整数相当于正整数的补码表示，因此图2.6的电路同时也能实现无符号数的加/减运算， 对于有符号数x 和y, 图中X 和 Y 分别是x 和y 的补码表示；对于无符号数x 和y, 图中X 和 Y

第2章 数据的表示和运算 43

分别是x 和y 的二进制表示。不论是补码减法还是无符号数减法，都是用被减数加上减数的负数 的补码(即γ+1)来实现的。



注 意

运算器本身无法识别所处理的二进制串是有符号数还是无符号数。例如，0-1=00…0+ 11…1=11…1,若解释为有符号数，对应值为-1,结果正确；若解释为无符号数，对应值为2"-1 (n 位无符号数的最大值),结果出错。此类易混点是统考极易考查的内容。

可通过标志信息来区分有符号整数运算结果和无符号整数运算结果。

命题追踪▶无符号数、有符号数加减运算后CF 和OF 的值(2011、2018、2023);SF 和 OF 的逻辑表达式(2022)

零标志 ZF:Z F=1 表示结果 F 为0。对于无符号数和有符号数的运算，ZF 都有意义。

溢出标志 OF: 判断有符号数运算是否溢出，它是符号位进位与最高数位进位的异或结果，

即OF=C,④C₁ 。 对于无符号数运算， OF 没有意义，通俗地说，就是根据OF 无法判断无符

号数运算是否溢出。例如，无符号数加法010+011=101,此时OF=1, 但结果未溢出。 符号标志 SF: 表示结果的符号，即F 的最高位。对于无符号数运算， SF 没有意义。

进/借位标志CF: 表示无符号数运算时的进位/借位，判断是否发生溢出。加法时，CF=1 表 示结果溢出，因此CF 等于进位输出Cou。减法时， CF=1 表示有借位，即不够减，故CF 等于进 位输出Cou取反。综合可得CF=Sub 田Cut。例如，无符号数加法110+011最高位产生进位，无 符号数减法000-111最高位产生借位，结果均发生溢出(即 CF =1)。对于有符号数运算，CF 没有意义，也就是说，根据CF 无法判断有符号数运算是否溢出。

(1)无符号数大小的比较

对于无符号数的运算，零标志ZF、进/借位标志CF 才有意义。假设有两个无符号数A 和B, 下面以执行 A-B 为例来说明ZF 、CF标志的几种可能情况。

若A=B, 如A-B=011-011=000, 此时结果为零ZF=1, 无借位CF=0。

若A>B, 如A-B=010-001=001, 此时结果非零ZF=0, 无借位CF=0。

若A<B, 如A-B=000-001=(1)000-001=111, 此时ZF=0, 有借位CF=1。

当ZF=1 时，说明A=B。 当ZF=0 且 CF=0 时，说明A>B。 当CF=1 时，说明A<B。 (2)有符号数大小的比较

对于有符号数的运算，零标志 ZF、 溢出标志OF、 符号标志SF 才有意义。假设两个有符号 数A 和 B, 用补码表示，以执行[A]-[B]n 为例来说明ZF 、OF 、SF 标志的几种可能情况。

若A=B, 如[A]-[B]=011-011=[A]+[-B]=011+101=(1)000, 此时结果为零ZF=1,

最高位进位与次高位进位的异或结果OF=C; 田 C₂=0, 结果的最高位 SF=0。

若A>B, 如[A]-[B]=010-001=010+111=(1)001, 此时ZF=0,OF=0,SF=0; 又

如[A]\*-[B]=011-101=011+011=110, 此时ZF=0,OF=1,SF=1。

若A<B, 如[A]\*-[B]n=000-001=000+111=111, 此时ZF=0,OF=0,SF=1 。 又

如[A]-[B] =101-011=101+101=(1)010, 此时ZF=0,OF=1,SF=0。

当ZF=1 时，说明A=B 。 当ZF=0 且未发生溢出时，即 OF=0 时，若 SF=0, 则表示结果 非负，说明A>B; 当发生溢出时，即OF=1 时，若SF=1, 则必然是正数减去负数发生溢出导 致结果为负，因此， 当OF=SF ( 或OF④SF=0) 且ZF=0 时，说明A>B 。当ZF=0 且未发生



44—2025年计算机组成原理考研复习指导

溢出时，即OF=0 时，若SF=1, 则表示结果为负，说明A<B; 当发生溢出时，即OF=1 时， 若 SF=0, 则必然是负数减去正数发生溢出导致结果为正，因此， 当OF≠SF ( 或OF@SF=1) 且ZF=0 时，说明A≤B。

4. 原码的加减法运算(了解)

在原码加减运算中，将符号位和数值位分开处理，具体的规则如下。

加法规则：遵循“同号求和，异号求差”的原则，先判断两个操作数的符号位。具体来说， 符号位相同，则数值位相加，结果符号位不变，若最高数值位相加产生进位，则发生溢出；符号 位不同，则做减法，绝对值大的数减去绝对值小的数，结果符号位与绝对值大的数相同。

减法规则：先将减数的符号取反，然后将被减数与符号取反后的减数按原码加法进行运算。



注 意

原码的加减运算规则比较复杂，因此计算机采用的大多是补码加减运算。

2.2.4 定点数的乘除运算 1. 定点乘法运算

(1)乘法运算的基本原理

命题追踪  如何用加、减、移位指令实现乘法指令(2020);用软/硬件实现乘法指令的速度

对比(2020)

原码乘法的特点是符号位与数值位是分开求的，原码乘法运算分为两步：①乘积的符号位由 两个乘数的符号位“异或”得到；②乘积的数值位是两个乘数的绝对值之积。两个定点数的数值 部分之积可视为两个无符号数的乘积。下面是两个无符号数相乘的手算过程。

0.1101 被乘数X=0.x₁X₂x₁X₁=0.1101

×0.1.011 乘数 Y=0.y₁Y₂Y₃Y₄=0.1011

1101~--------X×y₄×2\* X×1 右移4位

1101----------X×x×2- ¹ X×1 右移3位

0000------------X×y₂×2² X×0 右移2位

1101-------------X×y₁×2¹ X×1 右移1位

0.10001111

上述过程可写成数学推导过程：

X×Y=X×y×2⁴+X×y₃×2-³+X×yz×2-²+X×y₁×2-'

=2-¹{2-'[2-(2-^(0+X×y₄)+X×y₃)+X×y₂]+X×y}

更普遍地，n 位无符号数乘法X×Y可递归地定义如下。

P₀=0 (初始)

*P₁=2-(P₀+X×y,)*

P₂=2-¹(P₁+X×y-

*P=2-'(P-1+X×y₁)*

其递推公式为

P=2(P₁+X×y-)(i=0,1,2,…,n-1)

最终乘积为

**第2章** **数据的表示和运算** 45

*P₁=X×Y*

由上述分析可知，乘法运算可用加法和移位运算来实现(乘以2-'相当于做一次右移),两个 n位无符号数相乘共需进行n 次加法和n 次移位运算。原码乘法运算的过程可归纳如下：

① 被乘数和乘数均取绝对值参加运算，视为无符号数，符号位为x,④y,。

② 部分积P, 是乘法运算的中间结果，初值P₀=0 。 从乘数的最低位y,, 开始，将前面所得的 部分积 P, 加 上X×y,, 然后右移一位，此步骤重复n 次。



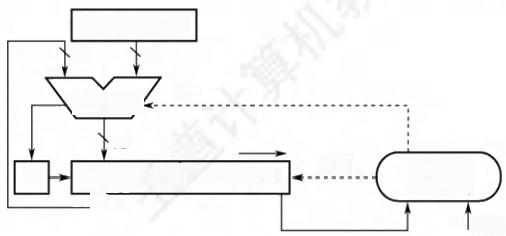
**注** **意**

由于参与运算的是两个数的数值位，因此运算过程中的右移操作均为逻辑右移。

(2)乘法运算电路

**命题追踪** ▶ **乘法电路中控制逻辑的作用(2020)**

图2.7是32位无符号数乘法运算的逻辑结构图。部分积和被乘数 X 做无符号数加法时，可 能产生进位，因此设置一个专门的进位位C 。乘积寄存器P 初始置0。计数器C, 初值为32,每循 环 一 次减1。ALU 是乘法器的核心部件，对乘积寄存器P 和被乘数寄存器X 的内容做“无符号加 法”运算，结果送回寄存器 P, 进位存放在C 中 。每次循环都对进位位 C 、寄存器 P 和寄存器Y 实现同步“逻辑右移”,此时，进位位C 移入寄存器P 的最高位，寄存器Y的最低位移出。每次 从寄存器Y移出的最低位都被送到控制逻辑，以决定被乘数是否“加”到部分积上。



被乘数寄存器 X

、32 32

加

32位ALU/

32

64位

乘积寄存器P

×32

时钟

控制逻辑 计数器C,

乘积寄存器 Y

右移

C

图2.7 32位无符号数乘法运算的逻辑结构图

命题追踪 ▶ **无符号数和有符号数乘法指令的溢出判断(2019、2020、2021)**

在字长为32位的计算机中，对于两个int型变量x 和 y 的乘积，若乘积高32位的每一位都 相同，且都等于乘积低32位的符号，则表示不溢出，否则表示溢出。当x 和 y 都为 unsigned int 型变量时，若乘积的高32位全为0,则表示不溢出，否则表示溢出。

**2.** **除法运算**

(1)除法运算的基本原理

原码的除法运算与乘法运算很相似，都是一种移位和加减运算迭代的过程，但比乘法运算更 复杂。n 位定点数的除法运算，需统一为： 一个2n 位的数除以一个n 位的数，得到一个n 位的商， 因此需要对被除数进行扩展。对于定点正小数(即原码小数),只需在被除数低位添n 个0即可。 对于定点正整数(即无符号数),只需在被除数高位添n 个0即可。做整数除法时，若除数为0, 则发生“除数为0”异常，此时需调出操作系统相应的异常处理程序进行处理。

下面以两个无符号数为例说明手算除法步骤。



46 -2025年计算机组成原理考研复习指导

00111 0010)000011

0010

00111

0010

0011

0010

0001

商=0111=7

被除数X=15=1111

=00001111

除数Y=2=0010

余数=0001=1

上述除法运算的过程可归纳如下：

① 被除数与除数相减，够减则上商为1,不够减则上商为0。

② 每次得到的差为中间余数，将除数右移后与上次的中间余数比较。用中间余数减除数， 够减则上商为1,不够减则上商为0。如此重复，直到商的位数满足要求为止。

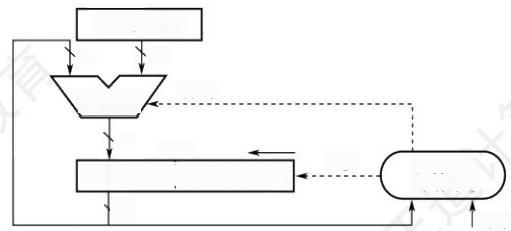
若 是 2n 位 除 以n 位的无符号数，商的位数为 n+1 位，当第一 次试商为1时，则表示结果溢 出 ( 即 无 法 用n 位表示商),如11111111/1111=10001。若是两个n 位的无符号数相除，则第一 位商为0,且结果肯定不会溢出，如两个4位数相除的最大商为00001111/0001=1111。对于浮点 数尾数的原码小数相除，第一次试商为1,则说明尾数部分有溢出，可通过右规消除。

计算机内部的除法运算与手算除法一样，通过被除数(中间余数)减除数来得到每一位商，

够减上商1,不够减上商0。原码除法运算也要将符号位和数值位分开处理，商的符号位是两个 数的符号位的“异或”结果，商的数值位是两个数的绝对值之商。

(2)除法运算电路

图2 . 8是 一 个32 位除法逻辑结构图。寄存器 Y 存放除数；寄存器R 初始时存放扩展被除数 的高32位，运算过程中存放中间余数的高位部分，结束时存放的是余数；寄存器 Q 初始时存放 扩展被除数的低32 位，运算过程中部分存放中间余数的低位部分、部分存放商，结束时存放的 是32位商。ALU 是除法器的核心部件，对寄存器R 和 Y的内容做加/减运算，运算结果被送回寄 存器 R 。计 数 器C, 初值为32,每循环 一 次减1。每次循环，寄存器 R 和 Q 实现同步左移，左移 时 ，Q 的最高位移入R 的 最 低 位 ，Q 中空出的最低位被上商。从低位开始，逐次把商的各个数位 左移到Q 中。每次由控制逻辑根据 ALU 运算结果的符号来决定上商是0还是1。



除数寄存器 Y

|  |  |
| --- | --- |
| 32  32位ALU,  32 | 32  加/减  64位 |

余数寄存器R: 余数/商寄存器Q

32

控制逻辑 计数器C,

|时钟

左移

图2.8 32位除法运算的逻辑结构图

若是两个32位int型整数相除，则除了-2¹/- 1会溢出，其余情况都不会溢出。

**2.2.5** **本** **节** **习** **题** **精** **选**

**一** **、** **单** **项** **选** **择** **题**

**01.** ALU 作为运算器的核心部件，其属于()。

A. 时序逻辑电路 B. 组合逻辑电路 C. 控 制 器 D. 寄存器

第2章 数据的表示和运算 47

**02.** 组成一个运算器需要多个部件，但下面的()不是组成运算器的部件。

A. 状态寄存器 B. 数据总线 C.ALU D. 地址寄存器

**03.** 算术逻辑单元 (ALU) 的功能一般包括()。

A. 算术运算 B. 逻辑运算

C. 算术运算和逻辑运算 D. 加法运算

**04.** 补码定点整数01010101算术左移两位后的值为()。

A.01000111 B.01010100 C.01000110 D.01010101 **05.** 下列四个补码整数存放于8位寄存器中，算术左移不会发生溢出的是( )。

A.80H B.90H C.BOH D.COH **06.** 补码定点整数10010101右移一位后的值为( )。

A.01001010 B.010010101 C.10001010 D.11001010 **07.** 两个机器数7E5H 和 4D3H 相加，得()。

A.BD8H B.CD8H C.CB8H D.CC8H

**08.** 设机器数字长8位(含1位符号位),若机器数BAH 为补码，算术左移1位和算术右移 1位分别得()。

A.F4H,EDH B.B4H,6DH C.74H,DDH D.B5H,EDH **09.** 在定点运算器中，无论是采用双符号位还是采用单符号位，必须有()。

A. 译码电路，它一般用“与非”门来实现

B. 编码电路，它一般用“或非”门来实现

C. 溢出判断电路，它一般用“异或”门来实现

D. 移位电路，它一般用“与或非”门来实现

**10.** 机器运算发生溢出的根本原因是()。

A. 寄存器的位数有限 B. 运算中将符号位的进位丢弃

C. 运算中将符号位的借位丢弃 D. 数据运算中发生错误

**11.** 假定有两个整数用8位补码分别表示为r=F5H,r₂=EEH。 若将运算结果存放在一个8

位寄存器中，则下列运算会发生溢出的是( )。

A.r₁+r₂ B.r₁-r: C.r₁×r: D.r₁/r₂ **12.** 关于模4补码，下列说法正确的是()。

A. 模4补码和模2补码不同，它更容易检查乘除运算中的溢出问题

B. 每个模4补码存储时只需一个符号位

C. 存储每个模4补码需要两个符号位

**D.** **模4补码，在算术与逻辑单元中为一个符号位**

**13 .** 若采用双符号位，则两个正数相加产生溢出的特征时，双符号位为()。 A.00 B.01 C.10 D.11

**14.** 判断加减法溢出时，可采用判断进位的方式，若符号位的进位为 Co, 最高位的进位为 C, 则产生溢出的条件是()。

1.C₀ 产生进位 II.C₁ 产生进位

ⅢI.C₀ 、 C₁ 都产生进位 IV.C₀ 、C₁ 都不产生进位

V.C₀ 产生进位， C₁ 不产生进位 VI.C₀ 不产生进位， C 产生进位

A.I 和 Ⅱ B.III C.IV D.V 和VI

**15.** 在补码的加减法中，用两位符号位判断溢出，两位符号位 S₅S=10 时，表示( )。

48 -2025年计算机组成原理考研复习指导

A. 结果为正数，无溢出 B. 结果正溢出

C. 结果负溢出 D. 结果为负数，无溢出

**16.** 若 [X]\*=X₀X₁X₂…X, 其 中X₀ 为符号位，X₁ 为最高数位。若(),则当补码左移时，

将会发生溢出。

A.X₀=X₁ B.Xo≠X₁ C.X₁=0 D.X₁=1

**17.** 原码乘法是( )。

A. 先取操作数绝对值相乘，符号位单独处理

B. 用原码表示操作数，然后直接相乘

C. 被乘数用原码表示，乘数取绝对值，然后相乘

D. 乘数用原码表示，被乘数取绝对值，然后相乘

**18.** 在原码乘法中，()。

A. 符号位参加运算

**B.** 符号位不参加运算

C. 符号位参加运算，并根据运算结果改变结果中的符号位

D. 符号位不参加运算，并根据运算结果确定结果中的符号

**19.** 原码乘法时，符号位单独处理乘积的方式是()。

A. 两个操作数的符号相“与” B. 两个操作数的符号相“或”

C. 两个操作数的符号相“异或” D. 两个操作数中绝对值较大数的符号

**20.** 下列关于移位运算的说法中，正确的是()。

I. 补码算术左移时，高位移出，低位补0,若左移前后的符号位不同，则发生溢出 **Ⅱ.** 无符号数逻辑左移时，若最高位移出的是1,则发生溢出

Ⅲ.逻辑左移和算术左移的结果都一样，都是移出最高位，并在低位补0

A.I 、H B. 仅 Ⅱ C. 只 有 Ⅲ D.I 、Ⅱ 、Ⅲ

**21.** 某计算机字长为8位， CPU 中有一个8位加法器。已知无符号数x=69,y=38, 若在该加

法器中计算x-y, 则加法器的两个输入端信息和输入的低位进位信息分别为()。 A.01000101、00100110、0 B.01000101、11011001、1

C.01000101、11011010、0 D.01000101、11011010、1

**22.** 某计算机中有一个8位加法器，有符号整数x 和 y 的机器数用补码表示，[x]=F5H,

[y]=7EH, 若在该加法器中计算x-y, 则加法器的低位进位输入信息和运算后的溢出 标志 OF 分别是( )。

A.1 、1 B.1 、0 C.0 、1 D.0 、0

**23** . 某8位计算机中， x 和 y 是两个有符号整数，用补码表示，[x]=44H,[v]=DCH, 则

x/2+2y 的机器数及相应的溢出标志OF 分别是( )。

A.CAH 、0 B.CAH 、1 C.DAH 、0 D.DAH 、1

**24.** 某8位计算机中，x 和 y 是两个有符号整数，用补码表示，[x]=44H,[y]=DCH, 则

x-2y 的机器数及相应的溢出标志OF 分 别 是 ( ) 。

A.8CH 、1 B.8CH 、0 C.68H、1 D.68H、0

**25.【**2009统考真题】一个C 语言程序在一台32位机器上运行。程序中定义了三个变量x,y, z, 其 中x 和 z 为 int 型 ，y 为 short 型。当x=127,y=-9 时，执行赋值语句z=x+y 后， X,y,z 的值分别是( )

A.x=0000007FH,y=FFF9H,z=00000076H

第2章 数据的表示和运算- 49

B.x=0000007FH,y=FFF9H,z=FFFF0076H

C.x=0000007FH,y=FFF7H,z=FFFF0076H

D.x=0000007FH,y=FFF7H,z=00000076H

**26.**【2010统考真题】假定有四个整数用8位补码分别表示：r=FEH,r₂=F2H,r₃=90H,

r₄=F8H, 若将运算结果存放在一个8位寄存器中，则下列运算会发生溢出的是()。 *A.r₁×r₂* B.r₂×r: C.r₁×r₂ D.r₂×r

**27.**【2013统考真题】某字长为8位的计算机中，已知整型变量x 、y 的机器数分别为[x]= 11110100,[y]\*=10110000。 若整型变量==2x+y/2, 则=的机器数为()。

A.11000000 B.00100100 C.10101010 D. 溢出

**28.**【2014统考真题】若x = 103,y=-25, 则下列表达式采用8位定点补码运算实现时，会

发生溢出的是()。

A.x+y B.-x+y C.x-y D.-x-y

**29.**【2018统考真题】假定有符号整数采用补码表示，若 int 型变量x 和 y 的机器数分别是 FFFF FFDFH 和0000004IH, 则 x 、y 的值及x-y 的机器数分别是()。

A.x=-65,y=41,x-y 的机器数溢出

B.x=-33,y=65,x-y 的机器数为FFFF FF9DH

C.x=-33,y=65,x-y 的机器数为 FFFF FF9EH

D.x=-65,y=41,x-y 的机器数为 FFFF FF96H

**30.【**2018统考真题】整数x 的机器数为11011000,分别对x 进行逻辑右移1位和算术右移 1位操作，得到的机器数各是( )。

A.11101100、11101100 B.01101100、11101100

C.11101100、01101100 D.01101100、01101100

**31.**【2018统考真题】减法指令 “sub R1,R2,R3” 的功能为“(R1)-(R2)→R3”, 该指令执

行后将生成进位/借位标志 CF 和溢出标志 OF 。若(R1) =FFFF FFFFH,(R2)=FFFF

FFFOH, 则该减法指令执行后， CF 与 OF 分别为( )。

A.CF=0,OF=0 B.CF=1,OF=0 C.CF=0 ,OF= 1 D.CF=1,OF=1

**32.【**2023统考真题】已知x,y 为 int 类型，当x=100,y=200 时，执行“x 减 y” 指令得到 的溢出标志OF 和借位标志CF 分别为0,1,那么当x=10,y=-20 时，执行该指令得到 的OF 和 CF 分别为( )。

A.OF=0,CF=0 B.OF=0,CF=1 C.OF=1,CF=0 D.OF=1,CF=1

**二、** **综合应用题**

**01.** 已 知A=-1001、B=-0101, 求[A+B] 和[A-B]。

**02.** 已知32位寄存器R1 中存放的变量x 的机器码为80000004H,unsigned int型的乘除法

采用逻辑移位操作， int 型的乘除法采用算术移位操作，请问：

1 ) 当x 是 unsigned int 型时， x 的真值是多少? x/2 存放在 R1 中的机器码是什么? x/2 的真值是多少?2x 存放在R1 中的机器码是什么?2x 的真值是多少?

2 ) 当x 是 int 型时，x 的真值是多少? x/2 存放在R1 中的机器码是什么? x/2 的真值是多 少 ? 2x 存放在R1 中的机器码是什么?2x 的真值是多少?

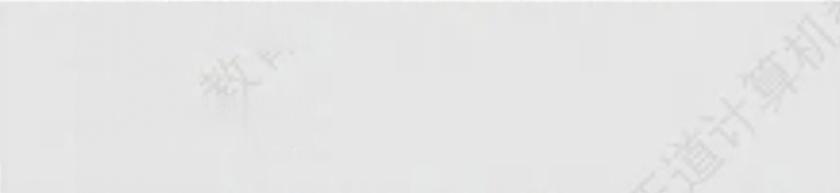
**03.** 假设有两个整数x=-68,y=-80, 采用补码形式(含1位符号位)表示，x 和y 分别存 放在寄存器A 和 B 中。另外，还有两个寄存器C 和 D 。A 、B 、C 、D 都是8位的寄存 器。请回答下列问题(要求最终用十六进制表示二进制序列):

50 -2025年计算机组成原理考研复习指导

1)寄存器 A和 B 中的内容分别是什么?

2)x 和 y 相加后的结果存放在寄存器C 中，则寄存器C 中的内容是什么?此时，溢出 标志OF、符号标志 SF 各是什么?

3)x 和y 相减后的结果存放在寄存器 D 中，寄存器 D 中的内容是什么?此时，溢出标 志 OF 、符号标志 SF 各是什么?



**04.【2**011 统考真题】假定在一个8位字长的计算机中运行如下C 程序段： unsigned int x=134;

ix; int y=246;

int n=y;

unsigned int z1=x-y;

unsigned int z2=x+Y;

intk1=m-n;

int k2=m+n;

若编译器编译时将8个8位寄存器R1～R8 分别分配给变量x,y,m,n,zl,z2,k1 和 k2。

请回答下列问题(提示：有符号整数用补码表示)。

1)执行上述程序段后，寄存器R1、R5 和 R6 的内容分别是什么(用十六进制表示)? 2)执行上述程序段后，变量m 和 k1 的值分别是多少(用十进制表示)?

3)上述程序段涉及有符号整数加减、无符号整数加减运算，这四种运算能否利用同一 个加法器辅助电路实现?简述理由。

4)计算机内部如何判断有符号整数加减运算的结果是否发生溢出?上述程序段中，哪

些有符号整数运算语句的执行结果会发生溢出?

**05.【**2020统考真题】有实现x ×y 的两个 C 语言函数如下：

unsigned umul(unsigned x,unsigned y) {return x\*y; }

int imul(int x,int y){returnx\*y;}

假定某计算机M 中的 ALU 只能进行加减运算和逻辑运算。请回答下列问题。

1 ) 若 M 的指令系统中没有乘法指令，但有加法、减法和位移等指令，则在 M 上也能

实现上述两个函数中的乘法运算，为什么?

2)若M 的指令系统中有乘法指令，则基于ALU 、位移器、寄存器及相应控制逻辑实现 乘法指令时，控制逻辑的作用是什么?

3)针对以下三种情况： a) 没有乘法指令； b) 有 使 用ALU 和位移器实现的乘法指令； c) 有使用阵列乘法器实现的乘法指令，函数 umul()在哪种情况下执行的时间最长? 在哪种情况下执行的时间最短?说明理由。

4)n 位整数乘法指令可保存2n 位乘积，当只取低n 位作为乘积时，其结果可能会发生 溢出。当n=32,x=23³¹-1,y=2 时，有符号整数乘法指令和无符号整数乘法指令得 到的x×y 的 2n 位乘积分别是什么(用十六进制表示)?此时函数 umul()和 imulO 的返回结果是否溢出?对于无符号整数乘法运算，当仅取乘积的低n 位作为乘法结 果时，如何用2n 位乘积进行溢出判断?

**2.2.6** **答** **案** **与** **解** **析**

**一、单项选择题**

**01.B**

ALU 是由组合逻辑电路构成的，最基本的部件是并行加法器。由于单纯的ALU 不能够存储

第2章 数据的表示和运算- 51

运算结果和中间变量，因此往往将 ALU 和寄存器或暂存器相连。

**02.D**

ALU 为运算器核心， C 正确；数据总线供 ALU 与外界交互数据使用，B 正确；溢出标志 即为 一个状态寄存器， A 正确。地址寄存器不属于运算器，而属于存储器， D 错 误 。

**03.** C

ALU 既能进行算术运算又能进行逻辑运算。

**04.B**

该数是 一个正数(最高位为0),按照补码算术移位规则，算术左移两位后，移出了最高位 01,低位补0,因此算术左移两位后的结果是01010100。虽然移位后该数的符号位仍为0,但是 移出了有效位1,所以本次算术移位发生了溢出。

**05.D**

80H=(10000000)<<1=00000000, 左移前的符号位为1,左移后的符号位为0,溢出。90H=(1001 0000)<<1=00100000,左移前的符号位为1,左移后的符号位为0,溢出。B0H=(10110000)<<1=0110

0000,左移前的符号位为1,左移后的符号位为0,溢出。COH=(11000000)<<1=10000000, 左 移前的符号位为1,左移后的符号位为1,未溢出， D 正确。

**06.D**

该数是 一 个负数(最高位为1),按照算术补码移位规则，负数右移添1,负数左移添0,所 以10010101 右移一位后的值为11001010。

**07.** C

在十六进制数的加减法中，逢十六进一 ，因此有7E5H+4D3H=CB8H。

**08.** C

算术左移时，低位补0;算术右移时，高位补符号位。BAH=(10111010)₂, 算 术 左 移 1 位 得 (01110100)₂=74H, 左移前后的符号位不同，溢出；算术右移1位得(11011101)₂= DDH。

**09.** C

三种溢出判别方法，均须有溢出判别电路，可用“异或”门来实现。

**10.** A

机器运算发生溢出的根本原因是计算机的字长有限，所以不能表示超过一定范围的数据。

**11.C**

首先将r₁ 和 r₂ 转换为真值， F5H=11110101, 转换为原码是10001011,真值为- 11;EEH=

11101110,转换为原码是10010010,真值为- 18,8位补码的表示范围为(- 128,127),r₁×r₂ 的 结 果为198,超出了8位补码的表示范围，发生溢出。

**12.** B

模4补码具有模2补码的全部优点且更易检查加减运算中的溢出问题， A 错 误。需要注意的 是，存储模4补码仅需一个符号位，因为任何一个正确的数值，模4补码的两个符号位总是相同 的 ，B 正确。只在把两个模4补码的数送往 ALU 完成加减运算时，才把每个数的符号位的值同 时送到ALU 的双符号位中，即只在ALU 中采用双符号位， C 、D 错误。

**13.B**

采用双符号位时，第一符号位表示最终结果的符号，第二符号位表示运算结果是否溢出。若 第二位和第一位符号相同，则未溢出；若不同，则溢出。若发生正溢出，则双符号位为01,若发 生负溢出，则双符号位为10。

**14.D**

采用进位位来判断溢出时，当最高有效位进位和符号位进位的值不相同时才产生溢出。两 正 数相加，当最高有效位产生进位 (C₁=1) 而符号位不产生进位 (C₀=0) 时，发生正溢出；两负 数相加，当最高有效位不产生进位 (C₁=0) 而符号位产生进位 (C₀=1) 时产生负溢出。因此溢 出条件为CG+C₀C=C₀④C。

**15.C**

用两位符号位判断溢出时，两个符号位不同时表示溢出，即01时表示正溢出；10时表示负 溢出；两个符号位相同时(11或00)表示未溢出。

**16.B**

补码左移时，若移出的高位不同于移位后的符号位，即左移前后的符号位不同，则发生溢出。 补 码 左 移 时 ，X₀ 移 出 ，X₁ 取 代X₀ 成为新的符号位，因此若X₀≠Xj, 则表示发生了溢出。

17.A

原码一位乘法中，符号位与数值位是分开进行运算的。运算结果的数值部分是乘数与被乘数 数值位的乘积，符号是乘数与被乘数的符号位的异或。

**18.B**

在原码一位乘法中，符号位不参加运算，符号位单独处理，同号为正，异号为负。

**19.C**

原码的符号位为1表示负数，为0表示正数。原码乘法时， 符号位单独处理，乘积的符号是 两个操作数的符号相“异或”的结果，同号为正，异号为负。

**20.D**

对于左移操作，逻辑左移和算术左移的结果都一样，高位移出，低位补0。逻辑移位不考虑 符号位的问题，逻辑左移时，若最高位移出的是1,表示发生溢出。算术左移时，若移出的高位 不同于移位后的符号位，即左移前后的符号位不同，表示发生溢出。因此I、Ⅱ、Ⅲ 均正确。

**21.B**

不管是补码减法，还是无符号数减法，都是用被减数加上减数的负数的补码来实现的。根据求 补公式，减数y 的负数的补码[-y]\*=Y+1, 因此，在加法器的Y’输入端用一个反向器实现，并用 控制端Sub 控制多路选择器是否将y 的各位取反后，输入Y’端，同时将Sub 作为低位进位送到加法 器。当 Sub 为 1 时 ， 做 减 法 ，Sub=1 控制将『输入到加法器Y’端，即实现“各位取反”功能；同 时 将 Sub=1 作为低位进位送到加法器，实现“末位加1”功能。69的二进制数为01000101;38 的二进制数为00100110,各位取反得11011001。做减法时，低位进位为Sub, 即 为 1 。



**注** **意**

若仅记忆补码加减运算的过程，而未掌握加法电路的原理，则本题易误选 D。

**22.A**

对于补码减法运算，控制端Sub 为1,故低位进位输入位= Sub=1。[x]a=11110101,[y]n=0111

1110,[-y]=10000001+1,[x]-[y]=[x]\*+[-y]=11110101+10000010=01110111, 进位丢

掉，参与运算的两个数的符号均为1,结果的符号位为0,故溢出标志OF 为 1 。

**23.** C

[x2+2y]=[x]>1+[y]x<<1=01000100>>1+11011100<<1=00100010+10111000=

11011010=DAH。x 右移移出了0,没有溢出或损失精度； y 为负数，左移后，符号位仍为1,没

有溢出；且从最后一步加法操作来看， 一个正数和一个负数相加，必然不会溢出。

第2章 数据的表示和运算- 53

24.A

[x]=44H=01000100,[v]=DCH=11011100 。 执行x-2y 时，先将y 算术左移一位，得到

10111000,未溢出，然后各位取反，再与x 相加，做减法时sub=1, 即01000100+01000111+1= 10001100(8CH),两个加数的符号都为0,而结果的符号为1,因此发生了溢出，即OF=1。

25.D

C 语言中的整型数据为补码形式， int 型为32位，short 型为16位，因此 x 、y的机器数写为 0000007F 、FFF7H。执行z=x+y 时，由于x 为 int 型，y 为short 型，因此需将y 的类型强制转 换为int 型，在机器中通过符号位扩展实现，由于y 的符号位为1,因此在y 的前面添加16个1, 即可将 y 强制转换为 int 型，其十六进制形式为 FFFFFFF7H。然后执行加法，即0000007FH+ FFFFFFF7H=00000076H, 其中最高位的进位1自然丢弃。

26.B

本题的真正意图是考查补码的表示范围，采用补码乘法规则计算出四个选项是费力不讨好的 做法，且极易出错。8位补码所能表示的整数范围为-128～+127。将四个数全部转换为十进制数：

r₁=-2,r₂=-14,r₃=-112,r₄=-8, 得 r₂×r₃=1568, 远超出了表示范围，发生溢出。

27.A

x\*2, 将x 算术左移一位为11101000;y/2, 将y 算术右移一位为11011000,均无溢出或丢 失精度。补码相加为11101000+11011000=11000000,亦无溢出。

28.C

8位定点补码表示的数据范围为-128～127,若运算结果超出这个范围，则会溢出。对于A,x+

y=103-25=78, 符合范围。对于B,-x+y=-103-25=-128, 符合范围。对于D,-x-y=-103+

25=-78,符合范围。对于C,x-y=103+25=128, 超过127。

29.C

利用补码转换成原码的规则：负数的符号位不变数值位取反加1;正数补码等于原码。两个

机器数对应的原码是[x]\*=80000021H, 对应的数值是-33,[y]=[y]=00000041H=65 。 排除 A、

D。x-y 直接利用补码减法准则，[x]n-[y]\*=[x]\*+[-y],-y 的补码是连同符号位取反加1,最

终减法变成加法，得出结果为 FFFFFF9EH。

30.B

逻辑移位：左移和右移空位都补0,且所有数字参与移动；补码算术移位：仍然是所有数字 参与移动，右移空位补符号位，左移空位补0。根据该规则，轻松选出B。

31.A

[x]\*-[y]\*=[x]n+[-y]\*,[-R2]\*=00000010H, 很明显[R1]\*+[-R2]\* 的最高位进位和符号位进 位都是1(当最高位进位和符号位进位的值不相同时才产生溢出),可以判断溢出标志OF 为0。同 时，减法操作只需判断借位标志， R1 大于 R2, 所以借位标志为0。

32.B

ALU生成标志位时只负责计算，而不管运算对象是有符号数还是无符号数，CF =1表示当 作无符号数运算时溢出，OF=1 表示当作有符号数运算时溢出。当作有符号数时，x=10,y=-20,

x-y=30, 未超过32位有符号数范围，不溢出， OF=0 。 当作无符号数时，x'=10,y'=2-20 (符号位读作数值位),x'-y'=30-2², 为负，超过32位无符号数范围，溢出， CF=1。

二、综合应用题

01. 【解答】

为判断溢出，采用双符号位。

54—2025年计算机组成原理考研复习指导

因为A=-1001,B=-0101, 所 以[A]=11,0111,[B]=11,1011,[-B]=00,0101,

则[A]+[B]=11,0111+11,1011=11,0010。

则[A]-[B]=[A]+[-B]=11,0111+00,0101=11,1100。

加/减运算结果的两个符号位都相同，无溢出，结果正确。

**02.【**解答】

1)对于无符号数，所有二进制位均为数值位。乘以2和除以2运算，相当于无符号数的逻 辑左移和逻辑右移。x 的真值为2³¹+2²。R1 中的机器码逻辑右移一位(高位补0)为4000 0002H, 相当于除以2,故x/2 的真值为230+2。R1 中的机器码逻辑左移一位(低位补0) 为00000008H, 相当于是乘以2,高位丢1,结果溢出，2x 的真值为2³(溢出)。

2)对于有符号数(补码),最高位为符号位。乘以2和除以2运算，相当于补码的算术左移和 算术右移。80000004H 对应二进制数的最高位为1,即为负数，其真值为-(2³¹-22)。R1 中 的机器码算术右移一位(高位补1)为C0000002H, 相当于除以2,x2 的真值为-(2²0-2)。 R1中的机器码算术左移一位(低位补0)为00000008H, 相当于乘以2,移位前后的符号位 不同，表示溢出，2x 的真值为-(2³¹-2³)(溢出)。

**03.【**解答】

1 ) 因 为x=-68=-(1000100)₂, 则[-68]=10111100=BCH; 因 y=-80=-(1010000)₂, 则

[-80]\*=10110000=BOH, 所以寄存器A 和 B 中的内容分别是 BCH、BOH。

2)[x+y]\*=[x]+[y]=10111100+10110000=(1)01101100=6CH, 所以寄存器C 中的内 容 是 6CH, 其真值为108。此时，溢出标志OF 为1,表示溢出，说明寄存器C 中的内容 不是正确结果；符号标志SF 为0,表示结果为正数 (OF 为 1 , 说 明SF 也是错的)。

3)[x-y]=[x]+[-y]=10111100+01010000=(1)00001100=0CH, 最高位前面的一位 被丢弃(取模运算),结果为12,所以寄存器D 中的内容是0CH, 其真值为12。此时， 溢出标志OF 为0,表示不溢出，也就是说，寄存器D 中的内容是正确的结果；符号标志 SF 为0,表示结果为正数。

**04.【**解答】

1)因为134=128+6=10000110B, 所 以x 的机器数为10000110B, 因此R1 的内容为86H。

246=255-9=11110110B, 所 以y 的机器数为11110110B,x-y=10000110 +0000

1010=(0)10010000,括号中为加法器的进位，因此R5 的内容为90H。x+y=10000110+ 11110110=(1)01111100,括号中为加法器的进位，因此R6 的 内 容 为 7CH。

2)m 的机器数与x 的机器数相同，皆为86H=10000110B, 解释为有符号整数m (用补码表 示) 时，其值为-1111010B=-122 。m-n 的机器数与x-y 的机器数相同，皆为90H=1001 0000B, 解释为有符号整数k1 (用补码表示)时，其值为-1110000B=-112 。

3 ) 能 。n 位加法器实现的是模2”无符号整数加法运算。对于无符号整数a 和 b,a+b 可 以

直接用加法器实现，而a-b 可 用a 加 -b 的补数实现，即a-b=a+[-b]#(mod 2"),所

以 n 位无符号整数加减运算都可在n 位加法器中实现。

因为有符号整数用补码表示，补码加减运算公式为[a+b] =[a]+[b](mod 2"),[a-b]=

[a]\*+[-b]\*(mod 2"), 所 以n 位有符号整数加减运算都可在n 位加法器中实现。

4)有符号整数加减运算的溢出判断规则为：若加法器的两个输入端(加法)的符号相同，

且不同于输出端(和)的符号，则结果溢出，或加法器完成加法操作时，若次高位(最 高数位)的进位和最高位(符号位)的进位不同，则结果溢出。

最后一条语句执行时会发生溢出。因为10000110+11110110=(1)01111100,括号中为 加法器的进位，根据上述溢出判断规则可知结果溢出。或者，因为两个有符号整数均为

**第2章** **数据的表示和运算** 55

负数，它们相加之后，结果小于8位二进制所能表示的最小负数。

**05.【**解答】

1)乘法运算可以通过加法和移位来实现。编译器可以将乘法运算转换为一个循环代码段，

在循环代码段中通过比较、加法和移位等指令实现乘法运算。

2)控制逻辑的作用是控制循环次数，控制加法和移位操作。

3)a 最 长 ，c 最短。对于a, 需要用循环程序段实现乘法操作，因而需要反复执行很多条指 令，而每条指令都需要取指令、译码、取数、执行并保存结果，所以执行时间很长；对 于b 和 c, 都只需用一条乘法指令实现乘法操作，不过b 中的乘法指令需要多个时钟周期 才能完成，而c 中的乘法指令可在一个时钟周期内完成，所以c 的执行时间最短。

4 ) 当n=32,x=2³l-1,y=2 时，有符号整数和无符号整数乘法指令得到的64位乘积都是 00000000 FFFF FFFEH 。int型的表示范围为[-2”,2³'-1],故函数 imul() 的结果溢出； unsigned int型的表示范围为[0,2³2-1],故函数umul() 的结果不溢出。对于无符号整数 乘法，若乘积高n 位全为0,即使低n 位全为1也正好是23² - 1,不溢出，否则溢出。

注意，无论是无符号数还是有符号数，用2n 位来表示两个n 位整数的相乘结果都不会溢出， 因 为 2n 位可以完整地存储两个n 位整数的乘积。但是，若只用低n 位来表示结果，则可能溢出。 因此，要保证低n 位转换成的真值与2n 位转换成的真值相等才算是不溢出。对于无符号数，只 要高 n 位全为0,就不会溢出，因为高 n 位在转换成真值后不会影响低n 位的值。对于有符号 数，要考虑符号位的影响。当结果是正数时，符号位为0,要求高n 位也全为0,且低n 位的最 高位也为0(否则正数变负数)。当结果是负数时，符号位为1,要求高n 位 也 全 为 1 , 且 低n 位的最高位也为1(否则负数变正数)。因此，在有符号数的情况下，高n+1 位相同表示不溢出。



2.**3** **浮点数的表示与运算**

**2.3.1** **浮** **点** **数** **的** **表** **示**

浮点数表示法是指以适当的形式将比例因子表示在数据中，让小数点的位置根据需要而浮 动。这样，在位数有限的情况下，既扩大了数的表示范围，又保持了数的有效精度。例如，用定 点数表示电子的质量(9×10-²g) 或太阳的质量(2×10³³g) 是非常不方便的。

**1.** **浮点数的表示格式**

通常，浮点数表示为

N=(-1)⁵×M×R

式中， S 取值0或1,用来决定浮点数的符号； M 是一个二进制定点小数，称为尾数， 一般用定 点原码小数表示； E 是一个二进制定点整数，称为阶码或指数，用移码表示。R 是基数(隐含), 可以约定为2、4、16等。可见浮点数由符号、尾数和阶码三部分组成。

图2.9是一个32位短浮点数格式的例子。

0 78 31

|  |  |  |
| --- | --- | --- |
| 符号 | 阶码 | 尾数 |

图2.9 浮点数格式的例子

其中，第0位为符号S; 第1～7位为移码表示的阶码 E (偏置值为64);第8～31 位为24 位二进制原码小数表示的尾数M; 基 数R 为2。阶码的值反映浮点数的小数点的实际位置；阶码

56-2025年计算机组成原理考研复习指导

的位数反映浮点数的表示范围；尾数的位数反映浮点数的精度。

2. 浮点数的表示范围

原码是关于原点对称的，故浮点数的范围也是关于原点对称的，如图2.10所示。



可表示的正数

正上溢

(1-2-24)×263 数轴

零

负下溢 正下溢

-2-24×2-64 0 2-24×2-64

可表示的负数

负上溢

-(1-2-24)×263

图2.10 浮点数的表示范围

运算结果大于最大正数时称为正上溢，小于绝对值最大负数时称为负上溢，正上溢和负上溢 统称上溢。数据一旦产生上溢，计算机必须中断运算操作，进行溢出处理。当运算结果在0至最 小正数之间时称为正下溢，在0至绝对值最小负数之间时称为负下溢，正下溢和负下溢统称下溢。 数据下溢时，浮点数值趋于零，计算机将其当作机器零处理。

3. 浮点数的规格化

为了在浮点数运算过程中尽可能多地保留有效数字的位数，使有效数字尽量占满尾数数位， 必须在运算过程中对浮点数进行规格化操作。所谓规格化操作，是指通过调整一个非规格化浮点 数的尾数和阶码的大小，使非零浮点数在尾数的最高数位上保证是一个有效值。

左规：当运算结果的尾数的最高数位不是有效位，即出现±0.0…0×…×的形式时，需要进 行左规。左规时，尾数每左移一位、阶码减1(基数为2时)。左规可能要进行多次。

右规：当运算结果的尾数的有效位进到小数点前面时，需要进行右规，右规只需进行一次。 将尾数右移一位、阶码加1(基数为2时)。右规时，阶码增加可能导致溢出。

基数为2的原码规格化尾数 M 应满足1/2≤|M<1, 形式如下：①正数为0.1×…×的形式，最 大值为0.11…1,最小值为0.100…0,表示范围为1/2≤M≤(1-2~");② 负数为1.1×…×的形式，最 大值为1.10…0,最小值为1.11…1,表示范围为-(1-2-")≤M≤-1/2。

基数不同，浮点数的规格化形式也不同。当浮点数尾数的基数为2时，原码规格化数的尾数 最高位一定是1。当基数为4时，原码规格化数的尾数最高两位不全为0。

4.IEEE754 标准

命 题 遍 踪▶ IEEE 754单精度数大小的比较(2014)

按照 IEEE754 标准，常用的浮点数的格式如图2.11 所示。

1位 8位 23位

|  |  |  |
| --- | --- | --- |
| 符号 | 阶码 | 尾数 |

(a)32 位单精度格式

1位 11位

52位

|  |  |  |
| --- | --- | --- |
| 符号 | 阶码 | 尾数 |

(b)64 位双精度格式

图2.11 IEEE754标准浮点数的格式

IEEE754 标准规定常用的浮点数格式有32位单精度浮点数(短浮点数、float 型)和64位双 精度浮点数(长浮点数、double 型),其基数隐含为2,见表2.2。

第2章 数据的表示和运算 57

**表2.2** **IEEE 754 浮点数的格式**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 类型 | 符号s | 阶码e | 尾数f | 总位数 | **偏置值** | |
| 十六进制 | 十进制 |
| 单精度 | 1 | 8 | 23 | 32 | 7FH | 127 |
| 双精度 | 1 | 11 | 52 | 64 | 3FFH | 1023 |

单精度格式中包含1位符号s 、8 位阶码 e 和23位尾数f; 双精度格式包含1 位符号 s、11 位阶码e 和52位尾数f。 基数隐含为2;尾数用原码表示。对于规格化的二进制浮点数，尾数的 最高位总是1,为了能使尾数多表示一位有效位，将这个1隐藏，称为隐藏位， 因此23位尾数实 际表示了24位有效数字。IEEE754 规 定隐藏位1的位置在小数点之前，例如，(12)o=(1100)₂,

将它规格化后结果为1.1×2³,其中整数部分的“1”将不存储在23位尾数内。



**注** **意**

单精度与双精度浮点数都采用隐藏尾数最高位的方法，因而使浮点数的精度更高。

在 IEEE754 标准中，指数用移码表示，但偏置值并不是通常n 位移码所用的2"-I, 而是2\*-I-1, 因此，单精度和双精度浮点数的偏置值分别为127和1023。在存储浮点数阶码之前，偏置值要先加 到阶码真值上。上例中，阶码值为3,因此在单精度浮点数中，移码表示的阶码为127+3=130(82H); 在双精度浮点数中，阶码为1023+3=1026(402H)。

IEEE 754 标准中，规格化单精度浮点数的真值为

(-1)⁶ ×1,f×2~127

规格化双精度浮点数的真值为

(-1) ×1,f×2- 1023

式中，规格化单精度浮点数的阶码e 的取值范围为1～254(8位);规格化双精度浮点数的阶码e 的取值范围为1～2046(11位)。IEEE 754 规格化浮点数的表示范围见表2.3。

命题追踪 **IEEE** **754** **单精度数的表示范围和有效位(2017、2018)**

**表2.3** **IEEE** **754规格化浮点数的表示范围**

|  |  |  |
| --- | --- | --- |
| **格式** | **最小值** | **最大值** |
| 单精度 | e=1,f=0  1.0×2l-127=2-126 | e=254,f=.111…,1.111…1×2254-127=2¹?⁷ ×(2-2-23 |
| 双精度 | e=1,f=0  1.0×2¹-1023=2-102 | e=2046,f=.111…,1.111…1×22046-102³=2¹02³×(2-2) |

对于IEEE754 格式的浮点数，阶码全为0或全为1时，有其特别的解释，如表2.4所示。

命题追踪 ▶ **IEEE 754标准中阶码全为0或全为1时的特殊意义(2017、2023)**

**表2.4阶码全为0或全为1时IEEE754浮点数的解释**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **值的类型** | **单精度(32位)** | | | | **双精度(64位)** | | | |
| **符号** | **阶码** | **尾数** | **值** | **符号** | **阶码** | **尾数** | **值** |
| 正零 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 负零 | 1 | 0 | 0 | -0 | 1 | 0 | 0 | -0 |
| 正无穷大 | 0 | 255(全1) | 0 | 0 | 0 | 2047(全1) | 0 | 0 |
| 负无穷大 | 1 | 255(全1) | 0 | -0 | 1 | 2047(全1) | 0 | -00 |

58 -2025 年计算机组成原理考研复习指导

**(续表)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **值的类型** | 单精度(32位) | | | | **双精度(64位)** | | | |
| **符号** | **阶码** | **尾数** | 值 | 符号 | **阶码** | 尾数 | 值 |
| 无定义数(非数) | 0或1 | 255(全1) | f≠0 | NaN | 0 或 1 | 2047(全1) | f≠0 | NaN |
| 非规格化正数 | 0 | 0 | f≠0 | 2(0. | 0 | 0 | f≠0 | 2-\*(0.) |
| 非规格化负数 | 1 | 0 | f≠0 | -2-^(0.f | 1 | 0 | f≠0 | -2+”(0.) |

1)全0阶码全0尾数：+0/-0。零的符号取决于符号s, 一般情况下+0和-0是等效的。

2)全1阶码全0尾数：+a/-o 。+o 在数值上大于所有有限数， -0则小于所有有限数。引入 无穷大数的目的是，在计算过程出现异常的情况下使得程序能继续进行下去。

3)全1阶码非0尾数：NaN(Nota Number)。表示一个没有定义的数，称为非数。

4)全0阶码非0尾数：非规格化数。非规格化数的特点是阶码为全0,尾数高位有一个或几 个连续的0,但不全为0。因此，非规格化数的隐藏位为0,且单精度和双精度浮点数的 指数分别为-126或-1022。非规格化数可以用于处理阶码下溢。

命题遍踪▶实数与 IEEE 754 单精度数的相互转换(2011、2013、2020、2022、2023)

【例2.5】将十进制数-8.25转换为IEEE754 单精度浮点数格式表示。

解：

IEEE 754 规定：单精度浮点数的偏置值是127;尾数最高位的“1”是被隐藏的。

先将-8.25转换成二进制数，即-1000.01=-1.00001×2³,再计算阶码E,E-127=3, 因此E= 130,转换成二进制数为10000010。

IEEE 754单精度浮点数格式：符号(1位)+阶码(8位)+尾数(23位)。因此，单精度 格式表示为

1;10000010;00001000 O00000000000000

即

11000001000001000000000000000000=C1040000H

【例2.6】求 IEEE 754 单精度浮点数 C6400000H 的值是多少。

解：

先将C6400000H按二进制展开为

11000110010000000000000000000000

其对应IEEE754 单精度浮点数的格式如下：

|  |  |  |
| --- | --- | --- |
| **符号** | **阶码** | **尾数** |
| 1 | **10001100** | **10000000000000000000000** |

因此，符号=1表示负数；阶码值为10001100-01111111=00001101 =13;尾数值为1.5 (注意其有隐藏位，要加1)。因此，浮点数的值为-1.5×2³。

5. 定点、浮点表示的区别

(1)数值的表示范围

若定点数和浮点数的字长相同，则浮点表示法所能表示的数值范围远大于定点表示法。 (2)精度

对于字长相同的定点数和浮点数来说，浮点数虽然扩大了数的表示范围，但精度降低了。 (3)数的运算

浮点数包括阶码和尾数两部分，运算时不仅要做尾数的运算，还要做阶码的运算，而且运算



**第2章** **数据的表示和运算** 59

结果要求规格化，所以浮点运算比定点运算复杂。

(4)溢出问题

在定点运算中，当运算结果超出数的表示范围时，发生溢出；在浮点运算中，运算结果超出 尾数表示范围却不一定溢出，只有规格化后阶码超出所能表示的范围时，才发生溢出。

**2.3.2** **浮点数的加减运算**

浮点数运算的特点是阶码运算和尾数运算分开进行，浮点数加减运算分为以下几步。

命 题 追 踪 ▶ **float型能否通过左移实现乘以2运算(2017);浮点数的加减运算(2009)**

**1.** **对阶**

对阶的目的是使两个操作数的小数点位置对齐，即使得两个数的阶码相等。为此，先求阶码 差，然后以小阶码向大阶码看齐的原则，将阶码小的尾数右移 一位(基数为2),阶码加1,直到 两个数的阶码相等为止。尾数右移时，若舍弃有效位会产生误差，影响精度。为了保证运算的精 度，尾数右移时，低位移出的位不要丢掉，应保留并参加尾数部分的运算。



**注** **意**

若采用大阶码向小阶码看齐的原则，则尾数需要左移，最高有效位被移出，导致结果出错。

**2.** **尾数加减**

将对阶后的尾数按定点原码小数的加(减)运算规则进行运算。因为 IEEE 754 浮点数尾数 中有一个隐藏位，因此在进行尾数加减时，必须将隐藏位还原到尾数部分。运算后的尾数不一定 是规格化的，因此，浮点数的加减运算需要进一步进行规格化处理。

**3.** **尾数规格化**

IEEE 754 规格化尾数的形式为±1.×…×。尾数相加减后会得到各种可能结果，例如：

1.×.…×+1.×…x=±1× . × . … ×

1.×…×-1.×…×=±0.0…01×…×

1 ) 右 规：当结果为±1×,×…×时，需要进行右规。尾数右移一位，阶码加1。尾数右移时，最 高位1被移到小数点前一位作为隐藏位，最后一位移出时，要考虑舍入。

2 ) 左 规：当结果为±0.0…01×… ×时，需要进行左规。尾数每左移一位，阶码减1。可能需要 左规多次，直到将第一位1移到小数点左边。



**注** **意**

① 左规 一 次相当于乘以2,右规 一 次相当于除以2;②需要右规时，只需进行 一 次。

**4.** **舍入**

在对阶和尾数右规时，可能会对尾数进行右移，为保证运算精度， 一般将移出的部分低位保 留下来，参加中间过程的运算，最后再将运算结果进行舍入，还原表示成IEEE 754格式。

IEEE 754 提供了以下4种可选的舍入模式。

1)就近舍入：舍入为最近的可表示数。当运算结果是两个可表示数的非中间值时，实际上 是“0舍1入”方式(类似于十进制的“四舍五入”法);当运算结果正好在两个可表示 数的中间时，则选择结果为偶数。例如，计算1.24×10⁵+5.04×10²(假定科学记数法的精 度保留两位小数),若只采用2位保留位，则结果是1.2400×10⁵+0.0050×10⁵=1.2450×10,

60-2025年计算机组成原理考研复习指导

这个结果在两个可表示数1.24×10°和1.25×10⁵ 的中间，采用就近舍入方式到偶数，则结 果应该是1 . 24×10°;若采用3 位保留位，则结果是1 .24000×10⁵+0 .00504×10⁵= 1.24504×10⁵,这个结果就不在1.24×10⁵和1.25×10的中间，而更接近于1.25×10⁵,采用 就近舍入方式，则结果应该是1.25×10⁵ 。显然，后者更为精确。

2)正向舍入：朝数轴+方向舍入，即取右边最近的可表示数。

3)负向舍入：朝数轴- ～方向舍入，即取左边最近的可表示数。

4)截断法：直接截取所需位数，丢弃后面的所有位，这种舍入处理最简单。对正数或负数 来说，都是取更接近原点的那个可表示数，是一种趋向原点的舍入。

**5.** **溢出判断**

▶ **浮点数运算时的溢出判断(2015)**

在尾数规格化和尾数舍入时，可能会对结果的阶码执行加/减运算。因此，必须考虑指数溢出 问题。若 一个正指数超过了最大允许值(127或1023),则发生指数上溢，产生异常。若 一个负 指数超过了最小允许值(- 149或- 1074)①,则发生指数下溢，通常把结果按机器零处理。

1)右规和尾数舍入。数值很大的尾数舍入时，可能因为末位加1 而发生尾数溢出，此时需 要通过右规来调整尾数和阶码。右规时阶码加1,导致阶码增大，因此需要判断是否发生 了指数上溢。当调整前的阶码为11111110时，加1后，会变成11111111而发生指数上溢。

2 ) 左 规。左规时阶码减1,导致阶码减小，因此需要判断是否发生了指数下溢。其判断规则 与指数上溢类似，左规一 次，阶码减1,然后判断阶码是否为全0来确定指数是否下溢。

由此可见，浮点数的溢出并不是以尾数溢出来判断的，尾数溢出可以通过右规操作得到纠正。 运算结果是否溢出主要看结果的指数是否发生了上溢，因此是由指数上溢来判断的。



**注** **意**

某些题目中可能会指定尾数或阶码采用补码表示。通常可以采用双符号位，当尾数求和结 果溢出(如尾数为10.××.…×或01.x×.…×) 时，需右规一次；当结果出现00.0××.…×或11.1××.…× 时，需要左规，直到尾数变为00.1××.…×或11.0××.…×。

**2.3.3** **C** **语** **言** **中** **的** **浮** **点** **数** **类** **型**

命 题 道 踪▶ **不同类型数据转换后数值的变化(2010)**

C 语言中的 float 型 和 double 型分别对应于 IEEE 754 单精度浮点数和双精度浮点数。long double 型对应于扩展双精度浮点数，但long double 型的长度和格式随编译器和处理器类型的不同 而有所不同。在 C 程序中，等式的赋值和判断会导致强制类型转换，以 char→int→long→double

和 float→double 最为常见，从前到后范围和精度都从小到大，转换过程没有损失。

不同类型数的混合运算时，遵循的原则是“类型提升”,即较低类型转换为较高类型。如 long 型 与int 型一起运算时，需先将int 型转换为long 型，然后进行运算，结果为long 型。若 float 型 和 double 型一起运算，虽然两者同为浮点型，但精度不同，则仍需先将 float 型转换为 double 型后再 进行运算，结果亦为 double 型。所有这些转换都是系统自动进行的，这种转换称为隐式类型转换。

▶ i **nt** **型和** **float型的精度和范围的分析(2017)**

1)int 型转换为 float 型时，虽然不会发生溢出，但 float 型尾数连隐藏位共24位，当 int 型

① 对于非规格化数的情况，当尾数f 为0.0…01时，指数的最小允许值为-126-23=-149或-1022-52=-1074。

大端方式

小端方式

**第2章** **数据的表示和运算** 61

数的第24～31 位非0时，无法精确转换成24位浮点数的尾数，需舍入处理，影响精度。 2)int 型 或 float 型 转 换 为double 型 时 ， 因 double 型的有效位数更多，因此能保留精确值。 3)double 型转换为 float 型时，因 float 型的表示范围更小，因此大数转换时可能会发生溢出。

此外，由于尾数有效位数变少，因此高精度数转换时会发生舍入。

4)float 型 或 double 型 转 换 为int 型 时 ， 因int 型没有小数部分，因此数据会向0方向截断 ( 仅 保留整数部分),发生舍入。另外，因int 型的表示范围更小，因此大数转换时可能会溢出。

在不同数据类型之间转换时，往往隐藏着 一 些不容易察觉的错误，编程时要非常小心。

**2.3.4** **数据的大小端和对齐存储**

**1.** **数据的“大端方式”和“小端方式”存储**

在存储数据时，数据从低位到高位可以按从左到右排列，也可以按从右到左排列。因此，无 法用最左或最右来表征数据的最高位或最低位，通常用最 低 有 效 字 节 (LSB) 和 最 高 有 效 字 节 (MSB) 来分别表示数据的低位和高位。例如，在32位计算机中， 一 个 int 型 变 量 i 的 机 器 数 为 01234567H, 其最高有效字节 MSB=01H, 最 低 有 效 字 节LSB=67H 。

命 题 追 踪▶ **数据的大小端存储(2016、2018、2019)**

现代计算机基本都采用字节编址，即每个地址编号中存放1字节。不同类型的数据占用的字 节数不同，如 int 型 和float 型 占 4 字 节 ，double 型占8字节等，而程序中对每个数据只给定 一 个

地址。假设变量 i 的 地 址 为 0 8 0 0H, 字 节 0 1H、23H、45H、67H 应各有 一 个内存地址，那么4

字节在内存中应如何排列呢?根据数据中各字节在连续字节序列中的排列顺序不同，可以采用两种 排列方式： 大 端 方 式 (big endian) 和小端方式 (little endian),如图2.12所示。

0800H 0801H 0802H 0803H

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 01H | 23H | 45H | 67H |  |

0800H 0801H 0802H 0803H

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 67H | 45H | 23H | 01H |  |

图2.12 采用大端方式和小端方式存储数据

▶ **根** **据存放顺序判断大小端方式(2019、2023)**

1 ) 大 端 方 式：先存储高位字节，后存储低位字节。字中的字节顺序和原序列的相同。 2)小端方式：先存储低位字节，后存储高位字节。字中的字节顺序和原序列的相反。

在检查底层机器级代码时，需要分清各类型数据字节序列的顺序，例如，以下是由反汇编器

(汇编的逆过程，即将机器代码转换为汇编代码)生成的 一 行机器级代码的文本表示：

4004d3: 0105 64940408 add eax,0x8049464

其中，“4004d3” 是十六进制数表示的地址，“010564940408”是指令的机器代码， “add eax, 0x8049464” 是指令的汇编形式，该指令的第二个操作数是 一 个立即数Ox8049464。 执行指令时， 从指令代码的后4字节中取出该立即数，立即数存放的字节序列为64H、94H、04H、08H, 正 好 与操作数的字节顺序相反，即采用的是小端方式存储，得到08049464H, 去 掉 开 头 的 0 , 得 到 值 0x8049464, 在阅读以小端方式存储的机器代码时，要注意字节是按相反顺序显示的。

**2.** **数据按“边界对齐”方式存储**

现代计算机都是按字节编址的，假设字长为32位，数据按边界对齐方式存放要求其存储地

|  |  |  |  |
| --- | --- | --- | --- |
| 字节1 | 字节2 | 字节3 | 半字1-1 |
| 半字1-2 | 半字2 | | 半字3-1 |
| 半字3-2 | 字1-1 | | |
| 字1-2 |  | | |

62 —2025年计算机组成原理考研复习指导

址是自身大小的整数倍，半字地址一定是2的整数倍，字地址一定是4的整数倍，这样无论所取 的数据是字节、半字还是字，均可一次访存取出。当所存数据不满足上述要求时，可通过填充空 白字节使其符合要求。这样做虽然会浪费一些存储空间，但可以提高存取数据的速度。当数据不 按边界对齐方式存储时，半字长或字长的数据可能在两个存储字中，此时需要两次访存，并对高 低字节的位置进行调整后才能得到所需数据，从而影响了系统的效率。

例如，“字节1、字节2、字节3、半字1、半字2、半字3、字1”的数据按序存放在存储器 中，按边界对齐方式和按边界不对齐方式存储时，格式分别如图2.13和图2.14所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 字节1 | 字节2 | 字节3 | 填充 |
| 半字1 | | 半字2 | |
| 半字3 | | 填充 | |
| 字1 | | | |

图2.13 按边界对齐方式存储 图2.14 按边界不对齐方式存储

**命题追踪** ▶ **结构体的小端、边界对齐存储(2012、2020)**

在C 语言的struct 类型中，“边界对齐”有两个重要要求：①每个成员按其类型的大小对齐， char 型的对齐值为1,short 型的对齐值为2,int 型的对齐值为4,单位为字节；②struct 的长度 必须是成员中最大对齐值的整数倍(不够就补空字节)。这样就能保证 struct 数组的每项都满足边 界对齐的条件。

先来看两个例子(32位，x86 环境， GCC 编译器):

struct A{ struct B{

int a; char b;

char b; int a;

short c; short c;

} }

结果却是： sizeof(A)=8,sizeof(B)=12。

之所以出现上面的结果，是因为编译器要使结构体成员在空间上对齐。为此必须满足： ①每 个成员存储的“起始地址%该成员的长度=0”,而结构体中的成员都是按定义的先后顺序排放的。 ②结构体的长度也必须是最大成员长度的整数倍，即结构体也要对齐排放。

设 B 从地址0x0000 开始，第一个成员b 的对齐值是1,其存放地址符合0x0000%1=0; 第 二个成员a 的对齐值是4,只能存放在从0x0004 到 0x0007 这四个字节中，满足0x0004%4=0 且 紧邻第一个成员；第三个成员c 的对齐值是2,可以存放在从0x0008 到 0x0009 这两个字节中。 此外，结构体长度必须是最大对齐值的整数倍，故0x000A 到 0x000B 也为B 所占用，共12字节。

设A 也从地址0x0000 开始，第一个成员a 的对齐值是4,存放在从0x0000 到 0x0003 这四个 字节中；第二个成员b 的对齐值是1,存放在0x0004 中；第三个成员c 的对齐值是2,为满足“起 始地址%N=0” 的条件，只能存放在从0x0006 到 0x0007 这两个字节中，结构体共占用8字节。

边界对齐方式相对边界不对齐方式是一种空间换时间的思想。精简指令系统计算机 RISC 通常采用边界对齐方式，因为边界对齐方式取指令时间相同，因此能适应指令流水。

**2.3.5** **本节习题精选**

**一** **、单项选择题**

**01.** 在 C 语言的不同类型的数据混合运算中，要先转换成同一类型后进行运算。设一表达式

第2章 数据的表示和运算- 63

中包含有int 型 、long 型 、char 型和 double 型的变量与数据，则表达式最后的运算结果 是(),这4种类型数据的转换规律是()。

A.long,int→char→double→long B.long,char→int→long→double

C.double,char→int→long→double D.double,char→int→double→long

**02.** 长度相同但格式不同的两种浮点数，假设前者阶码长、尾数短，后者阶码短、尾数长，

其他规定均相同，则它们可表示的数的范围和精度为( )。

A. 两者可表示的数的范围和精度相同 B. 前者可表示的数的范围大但精度低 C. 后者可表示的数的范围大且精度高 D. 前者可表示的数的范围大且精度高

**03.** 长度相同、格式相同的两种浮点数，假设前者基数大，后者基数小，其他规定均相同， 则它们可表示的数的范围和精度为( )。

A. 两者可表示的数的范围和精度相同 B. 前者可表示的数的范围大但精度低

C. 后者可表示的数的范围大且精度高 D. 前者可表示的数的范围大且精度高 **04.** 下列说法中正确的是( )。

A. 采用变形补码进行加减法运算可以避免溢出

B. 只有定点数运算才可能溢出，浮点数运算不会产生溢出

C. 定点数和浮点数运算都可能产生溢出

D. 两个正数相加时一定产生溢出

**05.** 在规格化浮点运算中，若某浮点数为2⁵ ×1.10101,其中尾数为补码表示，则该数()。

A. 不需规格化 B. 需右移规格化

C. 需将尾数左移一位规格化 D. 需将尾数左移两位规格化

**06.** 某浮点机，采用规格化浮点数表示，阶码用移码表示(最高位代表符号位),尾数用原 码表示。下列( )的表示不是规格化浮点数。

A.11111111,1.1000.00 B.0011111,1.0111…01

C.1000001,0.1111…01 D.0111111,0.1000.…10

**07.** 下列关于对阶操作说法正确的是( )。

A. 在浮点加减运算的对阶操作中，若阶码减小，则尾数左移

B. 在浮点加减运算的对阶操作中，若阶码增大，则尾数右移；若阶码减小 ，则尾数左移 C. 在浮点加减运算的对阶操作中，若阶码增大，则尾数右移

D. 以上都不对

**08.** 浮点数的IEEE754 标准对尾数编码采用的是( )。

A. 原 码 B. 反码 C. 补码 D. 移码

**09** . 在 IEEE 754 标准规定的64位浮点数格式中，符号位为1位，阶码为11位，尾数为52

位，则它所能表示的最小规格化负数为()。

A.-(2-2⁵²)×2-1023 B.-(2-2-52)×2+1023

C.-1×2~1024 D.-(1-2~52)×2+2047

**10.** 按 照IEEE754 标准规定的32位单精度浮点数41A4C000H 对应的十进制数是( )。

A.4.59375 B.-20.59375 C.-4.59375 D.20.59375

**11.** 在浮点数编码表示中，( )在机器数中不出现，是隐含的。

A. 阶 码 B. 符号 C. 尾数 D. 基 数

**12.** 若某单精度浮点数、某原码、某补码、某移码的32位机器数均为OxF0000000, 则这些 数从大到小的顺序是( )。

A. 浮原补移 B. 浮移补原 C. 移原补浮 D. 移补原浮

**13.** 采用规格化的浮点数最主要是为了()。

A. 增加数据的表示范围 B. 方便浮点运算

C. 防止运算时数据溢出 D. 增加数据的表示精度

**14.** 下列说法中，正确的是( )。

I. 在计算机中，表示的数有时会发生溢出，根本原因是计算机的字长有限 I. 一个正数的补码和这个数的原码表示一样，而正数的反码是原码各位取反

**Ⅲ.** 设有两个正的规格化浮点数N₁=2"×M₁ 和N₂=2"×M₂, 若 m>n, 则有N₁>N₂

A.I B.Ⅲ C.I、Ⅱ、Ⅲ D.I 、 Ⅲ

**15.** 在浮点运算中，下溢指的是().

A. 运算结果的绝对值小于机器所能表示的最小绝对值

B. 运算的结果小于机器所能表示的最小负数

C. 运算的结果小于机器所能表示的最小正数

**D.** 运算结果的最低有效位产生的错误

**16.** 判断浮点数运算是否溢出，取决于()。

A. 尾数是否上溢 B. 尾数是否下溢

C. 阶码是否上溢 D. 阶码是否下溢

**17.** 假定采用 IEEE 754 标准中的单精度浮点数格式表示一个数为45100000H, 则该数的值

是 ( ).

A.(+1.125)o×2l⁰B.(+1.125)w×2" C.(+0.125)1×2!"D.(+0.125)1o×2¹⁰

**18.**已知float 型采用IEEE754 单精度浮点格式，若x、y为 float 型变量，且x=-126,y=15.75, 则执行语句z=x+y 时，在浮点运算单元中进行对阶操作后的结果是( )。

A.x 不变， y 为010000101,0.001111110…0

B.x 不 变 ，y 为010000110,0.001111110.0

C.y 不 变 ，x 为110000101,0.001111110.0

D.y 不变， x 为110000110,0.001111110…0

**19.** 在 IEEE754 标准浮点格式中，非规格化浮点数表示为()。

A. 阶码为0,尾数为任意非0的二进制数

B. 阶码为255,尾数全为0

C. 阶码为255,尾数为任意非0的二进制数

D. 阶码为0,尾数全为0

**20** . 在规格化的浮点数中，将阶码部分的n 位移码表示改为n 位补码表示，其他不变，则浮 点数的表示范围( )。

A. 增大 B. 减小 C. 不变 D. 都不对

**21.** 设浮点数共12位。其中阶码含1位阶符共4位，以2为底，补码表示；尾数含1位符

号共8位，补码表示，规格化。则该浮点数所能表示的最大正数是( )。 A.27 B.28 C.28\_1 D.2⁷-1

**22.** 计算机在进行浮点数的加减运算之前先进行对阶操作，若x 的阶码大于y 的阶码，则应 将 ( )。

A.x 的阶码缩小至与y 的阶码相同，且使x 的尾数部分进行算术左移

**第2章** **数据的表示和运算** 65

B.x 的阶码缩小至与y 的阶码相同，且使x的尾数部分进行算术右移

C.y 的阶码扩大至与x 的阶码相同，且使y 的尾数部分进行算术左移 D.y 的阶码扩大至与x 的阶码相同，且使y 的尾数部分进行算术右移 **23.** 若浮点数的尾数用补码表示，则下列()中的尾数是规格化数形式。

A.1.11000 B.0.01110 C.0.01010 D.1.00010 **24.** 设浮点数的基数为4,尾数用原码表示，则以下()是规格化的数。

A.1.001101 B.0.001101 C.1.011011 D.0.000010 **25.** 下列关于舍入的说法，正确的是()。

1. 不仅仅只有浮点数需要舍入，定点数在运算时也可能要舍入

II. 在浮点数舍入中，只有左规格化时可能要舍入

Ⅲ.在浮点数舍入中，只有右规格化时可能要舍入

IV. 在浮点数舍入中，左、右规格化均可能要舍入

V. 舍入不一定产生误差

A.I 、HI 、V B.I 、Ⅱ 、V C.V D.I 、IV

**26.** 计算机中的信息按边界对齐方式存储的含义是()。

A. 信息的字节长度必须是整数 B. 信息单元的字节长度必须是整数

C. 信息单元的存储地址必须是整数 D. 信息单元的存储地址是其字节长度的整数倍 **27** .假设已定义三个int型变量x、y和z,sizeof(int)=4,double型采用IEEE 754双精度浮

点数格式，变量dx、dy和dz的声明和初始化如下：

double dx =(double)x;

double dy=(double)y;

double dz =(double)z;

则下列关系表达式中永远为真的是()。

I.dx+dy=(double)(x+y)

II.dx×dx>=0

III.dx/dx=dy/dy

IV.(dx+dy)+dz=dx+(dy+dz)

A.I 和Ⅱ B.Ⅱ 和 Ⅲ C.Ⅲ 和IV D.Ⅱ 和IV

28.在按字节编址的计算机中，采用小端方式存储数据，某静态二维数组b的声明如下： static short b[2][4]={{2,9,-1,5},{3,1,-6,2}};

若b 的首地址为0x8049820,采用按行优先存储，地址0x804982c中的内容是()。

A.FAH B.FFH C.00H D.05H

**29.**在按字节编址的计算机中，数据在存储器中以小端方式存放。假定 int 型变量 i 的地址 为08000000H,i 的机器数为01234567H, 地址08000000H单元的内容是()。

A.01H B.23H C.45H D.67H

**30.**在按字节编址的32位计算机中，按边界对齐方式为以下结构型变量x 分配存储空间：

struct cont\_info{

char id;

unsigned post;

char phone;

}x;

若x 的首地址为Ox8049820,则成员变量phone 的起始地址为()。

A.0x8049828 B.0x8049826 C.0x8049825 D.0x8049822

66 -2025年计算机组成原理考研复习指导

**31.** 在按字节编址、采用大端方式的16位计算机中，执行完下列C 语言程序片段后， m 的

低字节地址的内容为( )。

int n =0xA1B6;

unsigned int m =n;

m =m>>1;

A.50H B.A1H C.B6H D.DBH

**32.** 假定变量i 、f 的数据类型分别是int 、float 。已知i=12345,f=1.2345×23, 则在一个32 位机器中执行下列表达式时，结果为“假”的是()。

A.i=(int)(double)i B.f==(float)(double)f

C,i==(int)(float)i D.f==(float)(int)f

**33.** 言代码段：

float a=12.6,x;

x=m/2+a/2;

printf("%f\n",x);

执行上述代码后，输出的x 值为( )。

A.12.000000 B.12.300000 C.12.800000 D.12

**34.【2**009统考真题】浮点数加、减运算过程一般包括对阶、尾数运算、规格化、舍入和判 断溢出等步骤。设浮点数的阶码和尾数均采用补码表示，且位数分别为5和7(均含2 位符号位)。若有两个数X=27×29/32 和 Y=2⁵×5/8, 则用浮点加法计算X+Y 的最终结 果是( )。

A.001111100010 B.001110100010 C.010000010001 D. 发生溢出

**35.【2** 010统考真题】假定变量i 、f 和 d 的数据类型分别为int 、float 和 double(int 型用补 码表示，float型和 double型分别用IEEE754 单精度和双精度浮点数格式表示),已知i= 785、f=1.5678E3、d=1.5E100, 若在32位机器中执行下列关系表达式，则结果为“真” 的是( )。

I.i=(int)(float)i Ⅱ.f=(float)(int)f I.f=(float)(double)f IV.(d+f)-d=f

A. 仅 I 和 Ⅱ B. 仅 I 和 Ⅲ C. 仅Ⅱ和ⅢⅢ D. 仅 Ⅲ 和IV

**36.【**2011统考真题】float 型数据通常用IEEE 754单精度格式表示。若编译器将 float 型变 量x 分配在一个32位浮点寄存器 FR1中，且x=-8.25, 则 FR1 的内容是()。

A.C1040000H B. C2420000H C.C1840000H D.C1C20000H

**37.**【2012统考真题】float 型 ( 即IEEE754 单精度浮点数格式)能表示的最大正整数是( )。

A.2¹26-2103 B.2¹27\_2104 C.2¹27-2103 D.2l²8\_2104

**38.**【2012统考真题】某计算机存储器按字节编址，采用小端方式存放数据。假定编译器规

定 int 型和short 型长度分别为32位和16位，并且数据按边界对齐存储。某C 语言程序 段如下：

struct{

int a;

char b;

short c;

}record;

record.a =273;

若record变量的首地址为OxC008, 地址OxC008中的内容及 recordc 的地址分别为()。 A.0x00,0xC00D B.0x00,OxCOOE C.0x11,0xC00D D.0x11,0xC0OE

第2章 数据的表示和运算- 67

**39.**【2013统考真题】某数采用IEEE 754单精度浮点数格式表示为C6400000H, 则该数的

值是( )。

A.-1.5×213 B.-1.5×211 C.-0.5×213 D.-0.5×212

**40.**【2014统考真题】float 型数据常用IEEE 754单精度浮点格式表示。假设两个 float 型变

量x 和 y 分别存放在32位寄存器 fl 和 f2 中，若(fl)=CC900000H,(f2)=BOC00000H, 则x 和 y 之间的关系为()。

A.x<y 且符号相同 B.x<y 且符号不同

C.x>y 且符号相同 D.x>y 且符号不同

**41.**【2015统考真题】下列有关浮点数加减运算的叙述中，正确的是()。

I. 对阶操作不会引起阶码上溢或下溢

II. 右规和尾数舍入都可能引起阶码上溢

H. 左规时可能引起阶码下溢

IV. 尾数溢出时结果不一定溢出

A. 仅Ⅱ、Ⅲ B. 仅 I 、I 、IV C. 仅 I 、Ⅲ 、IV D.I 、Ⅱ 、Ⅲ 、IV

**42.**【2016统考真题】某计算机字长为32 位，按字节编址，采用小端方式存放数据。假定有 一个 double 型变量，其机器数表示为1122334455667788H, 存放在以00008040H 开 始的连续存储单元中，则存储单元00008046H中存放的是( )。

A.22H B.33H C.77H D.66H

**43.**【2018统考真题】IEEE 754单精度浮点格式表示的数中，最小的规格化正数是()。 A.1.0×2-126 B.1.0×2-127 C.1.0×2~128 D.1.0×2-149

**44.**【2018统考真题】某32位计算机按字节编址，采用小端方式。若语句 “inti=0;” 对应 指令的机器代码为“C745 FC00000000”, 则语句“inti=-64;”对应指令的机器代码 是 ( ) 。

A.C745 FC C0 FF FF FF B.C745 FC 0C FF FF FF

C.C745 FC FF FF FF CO D.C745 FC FF FF FF OC

**45.**【2020统考真题】在按字节编址、采用小端方式的32位计算机中，按边界对齐方式为以 下C 语言结构型变量a 分配存储空间：

struct record{

short

x1; x2;

int

}a;

若a 的首地址为2020 FE00H,a 的成员变量x2 的机器数为12340000H, 则其中34H 所 在存储单元的地址是( )。

A.2020 FE03H B.2020 FE04H C.2020 FE05H D.2020 FE06H

**46** .【2020统考真题】已知有符号整数用补码表示， float型数据用IEEE 754标准表示，假定

变量x 的类型只可能是int 或 float, 当 x 的机器数为C8000000H 时 ，x 的值可能是( )。

*A.-7×227* B.-2lt C.2l⁷ D.25×227

**47.**【2021统考真题】下列数值中，不能用IEEE754 浮点格式精确表示的是( )。

A.1.2 B.1.25 C.2.0 D.2.5

**48.**【2022统考真题】-0.4375的IEEE754 单精度浮点数表示为( )。

A.BEE00000H B.BF600000H C.BF700000H D.COE00000H **49.**【2023统考真题】若 short 型变量x=-8190, 则 x 的机器数是()。



68 -2025年计算机组成原理考研复习指导

A.E002H B.E001H C.9FFFH D.9FFEH

50.【2023统考真题】已知float 型变量用IEEE754 单精度浮点数格式表示。若float 型变量 x 的机器数为80200000H, 则x 的值是()。

A.-2-128 B.-1.01×2-127 C.-1.01×2-126 D. 非数 (NaN)

二 、综合应用题

01. 现有一计算机字长32位(D₃₁~D₀), 符号位是第31位。

对于二进制10001111111011111100000000000000,

1)表示一个补码整数，其十进制值是多少?

2)表示一个无符号整数，其十进制值是多少?

3)表示一个IEEE 754标准的单精度浮点数，其值是多少?

02.假定变量i 是一个32位的int 型整数，f和d 分别为float型(32位)和 double 型(64 位)实数。分析下列各布尔表达式，说明结果是否在任何情况下都是“true”。

1)i=(int)((double)i)

2)f==(float)((int)f)

3)f=(float)((double)f)

4)d=(double)((float)d)

03. 已知两个实数x=-68,y=-8.25, 它们在C 语言中定义为 float 型变量，分别存放在寄

存器A 和 B 中。另外，还有两个寄存器C 和D。A 、B 、C 、D 都是32位的寄存器。请 问(要求用十六进制表示二进制序列):

1)寄存器 A 和 B 中的内容分别是什么?

2)x 和y 相加后的结果存放在C 寄存器中，寄存器C 中的内容是什么? 3)x 和y 相减后的结果存放在D 寄存器中，寄存器D 中的内容是什么?

04. 对下列每个 IEEE 754单精度数值，解释它们所表示的是哪种数字类型(规格化数、非 规格化数、无穷大、0)。当它们表示某个具体数值时，请给出该数值。

1)0000 0000 0000 0000 0000 0000 0000

0000

0000

0000

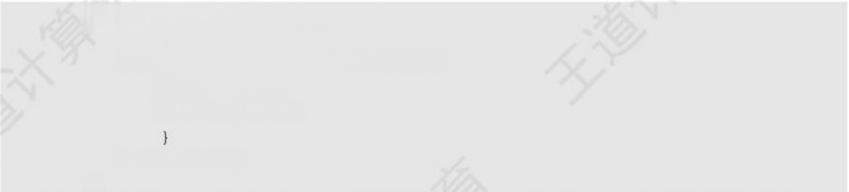
2)0100 0010 0100 0000 0000 0000 0000

3)1000 0000 0100 0000 0000 0000 0000

4)11111111 100000000000000000000000

,计算(n) 的C 语言函数fl 如下：

*05.[2017统考真题】已知*



int fl(unsigned n){

int sum=1,power=1;

for(unsigned i=0;i<=n-1;i++){

power \*=2;

sum +=power;

return sum;

}

将 fl 中的 int 都改为 float, 可得到计算f(n)的另一个函数 f2。假设 unsigned 型和 int 型 数据都占32位， float 型数据采用IEEE754 单精度标准。请回答下列问题：

1 ) 当n=0 时 ，fl 会出现死循环，为什么?若将fl 中的变量i 和n 都定义为int 型，则 fl 是否还会出现死循环?为什么?

第2章 数据的表示和运算- 69

2)fl(23) 和 f2(23)的返回值是否相等?机器数各是什么(用十六进制表示)? 3)fl(24) 和f2(24)的返回值分别为33554431和33554432.0,为什么不相等?

4)f(31)=2³²-1, 而 fl(31) 的返回值却为- 1,为什么?若使fl(n) 的返回值与 f(n)相等，

则最大的n 是多少?

5)f2(127) 的机器数为7F800000H, 对应的值是什么?若使f2(n) 的结果不溢出，则最大 的 n 是多少?若使f2(n)的结果精确(无舍入),则最大的n 是多少?

**2.3.6** **答案与解析**

**一、单项选择题**

**01.C**

不同类型的数据混合运算时，遵循的原则是“类型提升”,即较低类型转换为较高类型，最 终结果为 double 型。4种类型数据的转换规律为 char→int→long→double。

例 如 ，long 型数据与 int 型数据一起运算时，需先将 int 型转换为 long 型，然后两者再进行 运算，结果为long 型 。float 型数据和 double 型数据一起运算时，虽然它们同为实型，但两者精 度不同，仍要先将 float 型转换成 double 型再进行运算，结果亦为 double 型。所有这些转换都是 由系统自动进行的，这种转换通常称为隐式转换。

注意在强制类型转换中，从 int 型转换为 float 型时，虽然不会发生溢出，但因尾数位数的关 系，可能有数据舍入，而转换为 double 型则能保留精度。double 型转换为 float 型时亦是如此。 从 float 型或 double 型转换为int 型时，小数部分被截断，且由于int 型的表示范围更小，还可能 发生溢出。

**02.B**

在浮点数总位数不变的情况下，阶码位数越多，尾数位数越少；即表示的数的范围越大，精 度越差(数变稀疏)。

**03.B**

基数是浮点数的进制，决定了阶码变化的权重。基数越大，阶码每变化一位，尾数小数点需 要移动的位数越多，表示的数的绝对值就越大，范围就越大。但是，在浮点数的总位数不变的情 况下，能表示的不同状态个数是一定的。若范围增大，则意味着浮点数的离散程度增大，相邻两 个浮点数之间的间隔就越大，精度就越低。例如，假设符号为 S 、尾数为 M 、阶码为 E, 则基数 为2时的浮点数表示形式为(-1)×M×2\*, 基数为4时的浮点数表示形式为(-1)⁶ ×M×4^, 显然基数 为4时的表示范围大，但数据的离散程度也增大，精度降低。

**04.** C

变形补码，即用两个二进制位来表示数字的符号位，其余与补码相同，所以并不能避免溢出， A 错误。定点数和浮点数运算都可能产生溢出，但是溢出判断有区别，因此B 错误、C 正确。在 定点运算中，当运算结果超出数的表示范围时，就发生溢出；在浮点运算中，运算结果超出尾数 表示范围却不一定溢出，只有规格化后阶码超出所能表示的范围时，才发生溢出， D 错误。

**05.C**

考查浮点数的规格化。当尾数为补码表示，且为1.0×xx×形式时为规格化数，因此该尾数需 左移一位，阶码同时应减1,才为规格化数。

**06.B**

原码表示时，正数的规格化形式为0.1×…×,负数的规格化形式为1.1×…×,因此B 错误。 **07.** C

70 -2025年计算机组成原理考研复习指导

对阶操作，是将较小的阶码调整到与较大的阶码一致，因此不存在阶码减小、尾数左移的情 况，因而 A、B 错误。

**08.A**

IEEE 754 标准中尾数采用原码表示，阶码部分用移码表示。

**09.B**

长浮点数，其阶码11位，尾数52位，采取隐藏位策略，因此其最小规格化负数为阶码取最 大值2+1023(1023=2¹¹-I-1), 尾数取最大值2-2-52(注意其有隐藏位要加1),符号位为负。

**10.D**

在IEEE 754 单精度浮点数中，最高位为符号位；其后是8位阶码，以2为底，用移码表示， 阶码的偏置值为127;其后23位是尾数数值位。对于规格化的二进制浮点数，数值的最高位总是 “1”,为了能使尾数多表示一位有效值，将这个“1”隐藏，因此尾数数值实际上是24位。隐藏 的“1”是一位整数。在浮点格式中表示出来的23位尾数是纯小数，用原码表示。41A4C000H 写 成二进制为01000001101001001100000000000000,第一位为符号位0,表示是正数。之后的 8位10000011表示阶码，真值为(100),即4。剩下的是隐藏了最高1的尾数，故而为1.0100100 1100000000000000,数值左移四位后整数部分10100 表示为20。

**11.D**

浮点数表示中基数的值是约定好的，因此将其隐含。

**12.D**

这个机器数的最高位为1,对于原码、补码、单精度浮点数而言为负数，对于移码而言为正数，所 以移码最大，而补码为-228,原码为-(2⁰+2²⁹+228),单精度浮点数为-1.0×2°,大小依次递减。

**13.D**

与非规格化浮点数相比，采用规格化浮点数的目的主要是为了增加数据的表示精度。

**14.D**

I 正确；正数的原码、反码和补码都相同，Ⅱ错误；因为是规格化正浮点数，所以 M₁ 、M₂ 均为0.1xx形式，有N₁ 阶码至少比 N₂ 大 1 , 所 以N₁>N₂,Ⅲ 正确。

**15.A**

运算结果在0至规格化最小正数之间时称为正下溢，运算结果在0至规格化最大负数之间时 称为负下溢，正下溢和负下溢统称下溢。

**16.C**

判断浮点数运算是否溢出，取决于阶码是否上溢。阶码下溢可以通过非规格化数来表示。尾 数上溢或下溢，可以通过左移或右移进行调整。

17.B

写成二进制表示为01000101000100000000000000000000,第一位为符号位，0表示正 数，随后8位 (float 型)10001010是用移码表示的阶码，因此减去01111111后得十进制数11, 而 IEEE754 标准中单精度浮点数在阶码不为0时隐藏1,因此尾数为(1.0010)g=(1.125)p, 因 此 该数值为(+1.125)o×2"。

**18.A**

规格化IEEE 754浮点数尾数部分的数值范围为(1,2),x=-1111110B=-1.111110B×2⁶,y =

1111.11B=1.11111B×2²³,故浮点数x 、y 的阶数分别为6和3。对阶操作是小阶向大阶看齐，即y 的阶数变为6,移码表示为6+127=133,即10000101B,y 的尾数右移3位，变为0.00111111B。

**19.** A

在 IEEE 754标准格式中，阶码全为0,尾数不全为0表示非规格化数，非规格化数可用于处 理阶码下溢，使得出现比最小规格化数还小的数时程序也能继续进行下去。

20.C

在位数相同的条件下，移码和补码的表示范围是相同的，故阶码的表示范围不变，因此浮点 数的表示范围不变，只是改变了浮点数的表示形式。

21.D

为使浮点数取正数最大，可使尾数取正数最大，阶码取正数最大。尾数为8位补码(含符号 位),正值最大为0.1111111,即1-2-7,阶码为4位补码(含符号位),正值最大为0111,即7, 则最大正数为(1-2-7)×2⁷=2⁷-1。

22.D

浮点数加减运算时，首先要进行对阶，根据对阶的规则，阶码和尾数将进行相应的操作。

对阶的规则是小阶码向大阶码看齐，即阶码小的数的尾数算术右移，每右移一位，阶码加1, 直到两数的阶码相等为止。

23.D

补码的规格化表示是小数点后一位与符号位不同。

24.C

原码表示的规格化小数是小数点后2位(基数为4,用2位表示)不全为0的小数。

25.C

舍入是浮点数的概念，定点数没有舍入的概念。浮点数舍入的情况有两种：对阶、右规格化。 舍入不一定产生误差，如向下舍入11.00到11.0时是没有误差的。

26.D

信息在存储器中按边界对齐方式存储的含义是信息单元的存储地址是其字节长度的整数倍。 这样可以保证对一个字长数据的读/写只需要一次存储器访问，提高了访存效率，但有时会导致 存储空间的浪费。因此，这是一种以空间换时间的办法。

27.D

1非永真，因为x+y 可能会溢出，而dx+dy 不会溢出。Ⅱ永真，double 型采用 IEEE754 表 示，尾数采用原码小数，符号和数值部分分开计算，不管结果是否溢出，都不会影响乘积的符号。 Ⅲ非永真，dx 和dy 中只要有一个为0、另一个不为0就不相等。IV 永真，因为dx、dy 和dz 是 由32位int 型转换得到的， double 型可以精确地表示 int型，且对阶时尾数移动位数不会超过52 位，因此尾数不会舍入，因而不会发生大数吃小数的情况。

28.A

二维数组b的元素是short 型，占2字节，采用按行优先存储，b[0][0]的地址为0x8049820, b[0][1]的地址为0x8049822, 以此类推，b[1][2]的地址为0x804982c。b[1][2]的值为-6,补码表示 为1111111111111010,采用小端方式存储，因此地址0x804982c 存放的是低位字节FAH。

29.D

小端方式是将最低有效字节存储在最小位置。在数01234567H 中，最低有效字节为67H。

30.A

结构体按边界对齐存放的要求：数据成员的起始地址是其数据类型大小的整数倍， char 型占 1字节， char型的起始地址必须是1字节的整数倍； unsigned 型占4字节，故 unsigned 型的起始 地址必须是4字节的整数倍。据此分析，id的起始地址为0x8049820,post的起始地址为0x8049824,

72-2025年计算机组成原理考研复习指导

故 phone 的起始地址为Ox8049828。 结构体 x 的存放方式如下所示。

地址

|  |  |  |  |
| --- | --- | --- | --- |
| 8049820H | 8049821H | 8049822H | 8049823H |
| char |  |  |  |
| 8049824H | 8049825H | 8049826H | 8049827H |
| post | | | |
| 8049828H | 8049829H | 8049830H | 8049831H |
| phone |  |  |  |

地址

地址

**31.A**

int 型强制转换为unsigned int 型后，两个变量对应的每一位都是一样的，强制类型转换的结 果保持位值不变，仅改变了解释这些位的方式。因 此 ，m 的值仍为 AlB6H, 逻辑移位后的值为 50DBH, 由于采用大端方式，内存的低地址存放高位字节，因此低字节地址的内容为50H。

**32.** D

对于A 和 B,int 型的有效位数不会超过31位， float 型的有效位数比 double 型的小得多，因 此都能精确转换为具有53位有效位的 double 型。对于C,12345<1024×16=2, 因此12345对 应的二进制的位数一定小于14,因此可精确转换为具有24位有效位的float 型。对于D,f=1234.5, 转换为int 型后，小数点后面的数字丢失，因此与原来的f 不相等。

**33.B**

整数与整数运算，结果为整数，所以m/2 的结果为6。实数与整数运算，结果为实数，所以 a/2 的结果为6.3,相加为12.3。C 语言的输出格式可使输出值保留小数点后6位，输出为12.300000。

**34.D**

X 的浮点数格式为00,111;00,11101 ( 分号前为阶码，分号后为尾数),Y的浮点数格式为00, 101;00,10100。然后根据浮点数的加法步骤进行运算。

① 对 阶 。X 、Y阶码相减，即00,111-00,101=00,111+11,011=00,010,可知X 的阶码比 Y的价码大2(这一步可直接目测)。根据小阶码向大阶码看齐的原则，将Y 的阶码加2, 尾数右移2位，将Y变为00,111;00,00101。

② 尾数相加。即00,11101+00,00101=01,00010,尾数相加结果符号位为01,因此需要 右规。

③ 规格化。将尾数右移1位，阶码加1,得X+Y 为01,000;00,10001。

④ 判断溢出。阶码符号位为01,说明发生溢出。

本题容易误选 B 、C, 因为 B 、C 本身并无计算错误，只是它们不是最终结果， B 少 了 第 3 步 和 第 4 步 ，C 少了第4步。

**35.** B

题中三种数据类型强制类型转换的顺序为 int→float→double 。若将 float 型转换为int 型，小 数位部分会被舍去， int 型是精确到32位的整数，而 float 型只保存到1+23位，因此一个32位的 int 型整数在转换为float 型时可能会有损失，具体判断方法如下：先将int 型整数转换成二进制真 值，然后将真值写为±1x…x×2” 的形式，若小数点后的位数超过23位，则转换为 float 型会发生 精度损失。本 题 中i=785, 转换成二进制真值为1.100010001×2°,小数点后只有9位，不会发生 精度损失， I 正 确。对 于I, 将 float 型的f 转换为int 型，小数点后的数位丢失，结果非真。double 型的精度和范围都比 float 型的大，float 型转换为double 型不会有损失，Ⅲ正确。对于IV, 初 看 似乎没有问题，但浮点运算d+f 时需要对阶，对阶后f 的尾数有效位被舍去而变为0,因此d+f 仍然为d, 再减去d 后结果为0,结果非真。注意，从int 型转换为float 型时，虽然不会发生溢出， 但由于尾数位数的关系，可能有数据舍入，影响精度，而转换为double 型则能保留精度。

第2章 数据的表示和运算- 73

**36.A**

本题的目的在于考查IEEE754 单精度浮点数的表示。首先先将x 转换成二进制为-1000.01= -1.00001×2³,然后计算阶码E, 根 据IEEE 754单精度浮点数格式，有E-127=3, 因 此 E=130,

转换成二进制为10000010。最后，根据IEEE754 标准，最高位的“1”是被隐藏的。

IEEE 754单 精 度 浮 点 数 格 式 ： 符 号 ( 1 位 ) + 阶 码 ( 8 位 ) + 尾 数 ( 2 3 位 ) 。

因此 FR1 的内容为1;10000010;00001000000000000000000。

即11000001000001000000000000000000=C1040000H。

本题易误选 D, 未考虑IEEE 754 标准隐藏最高位1 的情况，把偏置值认为是128。 37.D

IEEE 754 单精度浮点数是尾数用采取隐藏位策略的原码表示，且阶码用移码(偏置值为127) 表示的浮点数。规格化短浮点数的真值为(-1)⁵ ×1.m×2E-127,其 中S 为符号位，阶码E 的取值为1~ 254(8位表示),尾数m 为23位，共32位；因此，foat 类型能表示的最大整数是1.111…1×225-127= 2127×(2-2-2³)=2128-210\*。

**【另解】**IEEE 754 单精度浮点数的格式如下图所示。

|  |  |  |
| --- | --- | --- |
| 符号(1) | 阶码(8) | 尾数(23) |

表示最大正整数时：符号取0;阶码取最大值127;尾数部分隐藏了整数部分的“1”,23位 尾数全取1时尾数最大，为2-2-2,此时浮点数的大小为(2-2-2)×2127=2128-2l04。

**38.D**

尽管 record 大小为7B (成员a 有 4B, 成 员b 有 1B, 成 员c 有 2B), 由于数据按边界对齐 方式存储，因此 record 共占用8B 。record.a 的十六进制表示为0x00000111, 由于采用小端方式存 放数据，因此地址OxC008 中的内容应为低字节Ox11;record.b 只 占 1B, 后面的1B 留空；record.c 占 2B, 因此其地址为 OxC00E 。各字节的存储分配如下表所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地址** | **0xC008** | **OxC009** | **0xC00A** | **0xC00B** |
| 内容 | record.a (0x11) | record.a (0x01) | **record.a** **(0x00)** | **record.a** **(0x00)** |
| 地址 | 0xC00C | **0xC00D** | **0xC00E** | **0xC00F** |
| 内容 | record.b | 一 | **record.c** | **record.c** |

**39.A**

IEEE 754单精度浮点数格式为C6400000H, 二进制格式为110001100100000000000000

00000000,转换为标准的格式为

|  |  |  |
| --- | --- | --- |
| 符号 | 阶码 | 尾数 |
| 1 | 10001100 | 10000000000000000000000 |

符号为1表示负数；阶码值为10001100-01111111=00001101=13;尾数值为1.5(注意其有隐 藏位，要加1)。因此，浮点数的值为-1.5×2¹³。

**40.A**

(fl) 和(f2)对应的二进制分别是(110011001001…)₂和(101100001100…)₂,根据 IEEE 754 浮点 数标准，可知(fl) 的符号为1,阶码为10011001,尾数为1.001,而(f2) 的符号为1,阶码为01100001, 尾数为1.1,可知两数均为负数，符号相同，B 、D 排除； (fl) 的绝对值为1.001×226,(f2) 的绝对值 为1.1×2-30,(fl)的绝对值比(f2)的绝对值大，而符号为负，真值大小相反，即(fl)的真值比(f2)的 真值小，即x<y。

**41.** D

对阶是较小的阶码向较大的阶码对齐，所以对阶后的阶码是当前那个较大的阶码而不会导致 阶码溢出，I 正确。右规和尾数舍入过程，阶码加1而可能上溢，Ⅱ正确，同理Ⅲ也正确。尾数 溢出时可能仅产生误差，结果不一定溢出，IV 正确。

**42.A**

大端方式： 一个字中的高位字节存放在内存中这个字区域的低地址处。小端方式： 一个字中 的低位字节存放在内存中这个字区域的低地址处。各字节的存储分配如下表所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 地址 | 00008040H | 00008041H | 00008042H | 00008043H |
| 内容 | 88H | 77H | 66H | 55H |
| 地址 | 00008044H | 00008045H | 00008046H | 00008047H |
| 内容 | 44H | 33H | 22H | 11H |

从而存储单元00008046H 中存放的是22H。

**43.** A

IEEE754 单精度浮点数的符号位、阶码位、尾数位(省去正数位1)所占的位数分别是1、8、 23。最小正数，符号位取0,移码的取值范围是1～254,取1,得阶码值1- 127=- 126(127为 我们规定的偏置值),尾数取全0,最终推出最小规格化正数为A。

**44.A**

按字节编址，采用小端方式，低位的数据存储在低地址位、高位的数据存储在高地址位，并 且按照一字节相对不变的顺序存储。由题意，存储0的位数是后32位，则我们只需要把-64的补 码按字节存储在其中即可，而-64表示成32位的十六进制是 FFFFFF C0, 根据小端方式的特点， 低位字节存储在低地址，就是C0 FF FFFF。

**45.D**

在32位计算机中，按字节编址，根据小端方式和按边界对齐的定义，变量a 的存放方式：

地址 2020 FE00H 2020 FE01H 2020 FE02H 2020 FE03H

|  |  |  |  |
| --- | --- | --- | --- |
| 未知 | 未知 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 说明 | x1(LSB) | x1(MSB) |  |  |
| 地址 | 2020 FE04H | 2020 FE05H | 2020 FE06H | 2020 FE07H |

|  |  |  |  |
| --- | --- | --- | --- |
| 00H | 00H | 34H | 12H |

说明 x2(LSB) x2(MSB)

于是，34H 所在存储单元的地址为2020 FE06H。

**46.** A

C8000000H=1100100000000000 O000000000000000。

将其转换为对应的 float 型或 int 型。

1)若为float 型，则尾数隐藏最高位1,符号为1表示负数，阶码10010000=2⁷+2⁴=128+16,

再减去偏置值127得到17,算出x 值为-2”。

2)若为int 型，则有符号补码，为负数，数值部分取反加1,得01110000000 O00000000000 00000000,算出x 值为-7×2”。

**47.** A

使用排除法。选项B:1.25=1.01B×2°; 选 项C:2.0=1.0B×2'; 选 项 D:2.5=1.01B×2' 。 因 此，B 、C 和 D 均可以用IEEE754 浮点格式精确表示。选项A 的十进制小数1.2转换成二进制的 结果是无限循环小数1.001100110011…,无法用精度有限的IEEE 754 格式精确表示。

**48.** A

第2章 数据的表示和运算- 75

IEEE754 单精度浮点数格式中依次为符号1位、阶码8位(偏置值127)、尾数23位(隐藏 1位)。 -0.4375=-1.75×2-²,保证小数点前是1。根据单精度浮点数格式，符号为1;阶码为移 码表示， -2+127=125,写成8位二进制数为01111101;尾数隐藏小数点前的1,剩下的0.75 写成二进制数为0.11,所以尾数部分是1100…0。该浮点数的二进制格式为1011111011100000 000000000000000,对应的十六进制格式为BEE00000H。

49.A

short 型变量是补码表示的16位有符号整数。x 是负数，可先求出8190的机器数，8190=8192- 2=2¹³-2',8190的机器数为0010000000000000B-0000000000000010B =000111111111 1110B, 因此-8190的机器数为1110000000000010B=E002H (按位取反，末位加1)。

50.A

把x 的机器数按二进制展开，80200000H=10000000001000000000000000000000B, 符 号为负，阶码全为0,尾数不全为0,由下表可知，这是非规格化数，对于32位非规格化负数， 若尾数的二进制为f, 则真值为-2-126×0.f=-2-126×0.01=-2-128。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **值的类型** | **单精度(32位)** | | | |
| 符号 | **阶码** | **尾数** | **值** |
| 非规格化正数 | 0 | 0 | f≠0 | 2-1²\*(0.f) |
| 非规格化负数 | 1 | 0 | f≠0 | -2-26(0.f) |

二、综合应用题

01. 【解答】

1)最高位为符号位，符号位为1,表示是一个负数，对应真值的二进制为

-1110000000100000100000000000000(数值位取反，末位加1)

对应的十进制值为-(230+2²⁹+228+220+2⁴)。

2)全部32位均为数值位，按权相加可知其十进制值为

2³¹+22⁷+22⁶+2²⁵+2²⁴+2²³+2²²+2²¹+2l⁹+2l⁸+2¹⁷+2¹⁶+2¹⁵+2l⁴

3)表示一个 IEEE 754 标准的单精度浮点数：

|  |  |  |
| --- | --- | --- |
| 符号 | 阶码  00011111 | 尾数  11011111100000000000000 |

因为阶码为00011111,所以对应的十进制数为31。IEEE 754标准中的阶码用移码表示，其 偏置值为127,所以阶码的十进制真值为31-127=-96。

因为尾数为1.11011111100000000000000。IEEE 754 标准中的尾数用原码表示，且采用隐藏 尾数最高数位“1”的方法，隐藏的“1”是一位整数。所以尾数真值为

1+2-¹+2-²+2~+2-⁵+2-⁶+2-⁷+2-8+2-⁹

因为符号为1,表示这个浮点数是个负数。所以单精度浮点数的真值为

-(1+2-¹+2-²+2⁴+2-⁵+2⁶+2-⁷+2-8+2-)×2-46

02. 【解答】

强制类型转换，转换过程有两个， 一是unsigned int→int→long→double, 二是float→double, 从后向前转换会使得数据丢失，进而使等号不成立。

1)是。因为double型比 int型精度高，所以int型变量转换为double型变量时不会有精度损失。 2)不是。因为 float 型有小数部分，而 int 型没有小数部分，所以把 float 型变量转换为 int

76-2025 年计算机组成原理考研复习指导

型变量时，可能会丢失小数部分。

3)是。因为 double型比 float 型精度高，所以 float 型变量转换为double 型变量时不会有精 度损失。

4)不是。因为 float 型比 double 型的有效位数少，所以double 型变量转换为float 型变量时 会有精度损失。

03. 【解答】

1)float 型变量在计算机中都被表示成 IEEE 754 单精度格式。X=-68=-(1000100)₂=

-1.0001×2⁶,符号位为1,阶码为127+6=128+5=(10000101)₂,尾数为1.0001,所以 小数部分为00010000000000000000000,合起来整个浮点数表示为110000101000 10000000000000000000,写成十六进制为C2880000H。

γ=-8.25=-(1000.01)₂=-1.00001×2³,符号位为1,阶码为127+3=128+2=(1000 0010)₂,尾数为1.00001,所以小数部分为00001000000000000000000,合起来整个浮 点数表示为11000001000001000000000000000000,写成十六进制为C1040000H。

因此，寄存器 A 和B 的内容分别为C2880000H、C1040000H。

2)两个浮点数相加的步骤如下。

①对阶： E₄=10000101,E₃=10000010, 则

[E₄-E₁]\*=[E;]\*+[-E,]=10000101+01111110=00000011

E₂ 大于E,, 所以对y 进行对阶。对阶后，y=-0.00100001×2⁶。

②尾数相加：x 的尾数为-1.00010000000000000000000,y 的尾数为-0.001000010000000 00000000,用原码加法运算实现，两数的符号相同，做加法，结果为-1.001100010000000

00000000。

即x 加y 的结果为-1.00110001×2⁶,所以符号位为1,尾数为0011000100000000000 0000,阶码为127+6=128+5,即10000101。合起来为110000101001100010000000 00000000,转换为十六进制形式为C2988000H。

所以C 寄存器中的内容是C2988000H。

3)两个浮点数相减的步骤同加法，对阶的结果也一样，只是尾数相减。

尾数相减：x 的尾数为-1.00010000000000000000000,y 的尾数为-0.001000010000000 00000000。

用原码减法运算实现，两数的符号相同，做减法；符号位：取大数的符号，负数，为1;数 值部分：大数加小数负数的补码：

1.000 10000000000000000000

+ 1. 110 1111 1000000000000000

0. 111 0111 1000 0000 0000 0000

x 减y 的结果为-0.11101111×2⁶=-1.1101111×2⁵,所以符号位为1,尾数为110111100000000 00000000,阶码为127+5=128+4,即10000100。

合起来为11000010011011110000000000000000,转换为十六进制形式为C26F0000H。所 以寄存器 D 中的内容是 C26F0000H。



提 示

若为选择题，则第二问可先将十进制的x+y,x-y 计算后的结果再转成IEEE 754。

第2章 数据的表示和运算 一 77

**04.【**解答】

1)因为该数的阶码字段内容为0,符号位为0,尾数字段内容也为0,所以它表示 IEEE 浮 点格式的+0。

2)该数的阶码字段内容为132,尾数字段内容为10000000000000000000000,因为阶码字 段的内容既不全部为0,也不全部为1,所以它表示 一 个规格化数，其实际值为 (1.1)₂×2⁵=48。

3)因为该数的阶码字段内容全部为0,且尾数字段内容不全部为0,所以它表示一个非规格 化数，其实际值为(-0.1)₂ ×2-126=-2-127=-5.877×10-39(表示成4位有效数字形式)。

4)因为该数的阶码字段内容全部为1,且尾数字段内容为0,符号位为1,所以它表示负无 穷大。

**05.【**解答】

1)由于i 和 n 是 unsigned 型，因此“i<=n-l” 是无符号数比较，n=0 时 ，n-1 的机器数为全1, 值是2² - 1,为unsigned 型可表示的最大数，条件 “i<=n-1” 永真，因此出现死循环。

若i 和 n 改 为int 型，则不会出现死循环。

因 为 “i<=n-1” 是有符号整数比较，当n=0 时 ，n-1 的值是 - 1,当i=0 时，条件 “i

<=n-1” 不成立，此时退出 for 循环。

2)fl(23) 与 f2(23)的返回值相等。f(23)=22³+1-1=2²⁴-1, 其二进制形式是24个1。int 型

数占32位，没有溢出。float 型数有1个符号位，8个指数位，23个底数位，23个底数位

可以表示24位的底数，所以两者返回值相等。

fl(23) 的机器数是00FF FFFFH,f2(23)的机器数是4B7FFFFFH。

显而易见，前者是24个1,即00000000111111111111111111111111(zy, 后者的符号位 是0,指数位为23+127uo)=10010110zy, 底数位是1111111111111111111111l(zy。

3 ) 当n=24 时 ，f(24)=1111111111111111111111111 B,而 float 型数只有24位有效位，

舍入后数值增大，所以 f2(24)比 fl(24) 大1。

4)显然f(31) 已超出了int 型数的表示范围，用fl(31) 实现时得到的机器数为32个1,作为 int 型解释时其值为- 1,即fl(31) 的返回值为-1。

因 为int 型最大可表示的数是0后面加31个1,因此使fl(n) 的返回值与 f(n)相等的最大n 值是30。

5)IEEE 754标准使用“阶码全1、尾数全0”表示无穷大。f2 的返回值为float 型，机器 数 7F800000H 对 应 的 值 是 + 。 当n=126 时 ，f(126)=2127-1=1.1…1×2126, 对应的 阶码为127+126=253,尾数部分舍入后阶码加1,最终阶码为254,是IEEE 754 单

精度格式表示的最大阶码。因此使 f2 结果不溢出的最大n 值为126。

当 n=23 时 ，f(23) 为 2 4 位 1 , float 型数有24位有效位，所以不需要舍入，结果精确。 因此使 f2 获得精确结果的最大n 值为23。

**2** **.** **4本章小结**

本章开头提出的问题的参考答案如下：

1)在计算机中，为什么要采用二进制来表示数据?

答案已在本章开头说明。

78-2025年计算机组成原理考研复习指导

2)计算机在字长足够的情况下能够精确地表示每个数吗?若不能，请举例说明。

计算机采用二进制来表示数据，在字长足够时，可以表示任何一个整数。而二进制表示小数 时只能够用1/(2")的和的任意组合表示，即使字长很长，也不可能精确表示出所有小数，只能无 限接近。例如0. 1 就无法用二进制精确地表示。

3)字长相同的情况下，浮点数和定点数的表示范围与精度有什么区别?

字长相同时，浮点数取字长的一部分作为阶码，所以表示范围比定点数要大，而取一部分作 为阶码也就代表着尾数部位的有效位数减少，而定点数字长的全部位都用来表示数值本身，精度 要比同字长的浮点数更大。

4)用移码表示浮点数的阶码有什么好处?

移码的两个好处：

① 浮点数进行加减运算时，要比较阶码的大小，移码比较大小更方便。

② 检验移码的特殊值(0和 max) 时比较容易。阶码以移码编码时的特殊值如下。0:表示 指数为负无穷大，相当于分数分母无穷大，整个数无穷接近0,在尾数也为0时可用来表 示0;尾数不为零表示未规格化的数。 max: 表示指数正无穷大，若尾数为0,则表示浮 点数超出表示范围(正负无穷大);尾数不为0,则表示浮点数运算错误。



**2.5** **常见问题和易混淆知识点**

1. 如何表示一个数值数据?计算机中的数值数据都是二进制数吗?

在计算机内部，数值数据的表示方法有以下两大类。

① 直接用二进制数表示。分为有符号数和无符号数，有符号数又分为定点数表示和浮点数 表示。无符号数用来表示无符号整数(如地址等信息)。

② 二进制编码的十进制数， 一 般采用 BCD 码表示，用来表示整数。

所以，计算机中的数值数据虽然都用二进制表示，但不全是二进制，也有用十进制表示的。 例如在指令类型中，就分别有二进制加法指令和十进制加法指令。

2. 什么称为无符号整数的“溢出”?

对于无符号定点整数来说，若寄存器位数不够，则计算机运算过程中一般保留低n 位，舍弃 高位。这样，会产生以下两种结果。

① 保留的低 n 位数不能正确表示运算结果。在这种情况下，意味着运算的结果超出了计算

机所能表达的范围，有效数值进到了第n+1 位，称此时发生了“溢出”现象。 ② 保留的低 n 位数能正确表达计算结果，即高位的舍去并不影响其运算结果。

3. 如何判断一个浮点数是否是规格化数?

为了使浮点数能尽量多地表示有效位数， 一般要求运算结果用规格化数形式表示。规格化浮 点数的尾数小数点后的第一位一定是个非零数。因此，对于原码编码的尾数来说，只要看尾数的 第一位是否为1就行；对于补码表示的尾数，只要看符号位和尾数最高位是否相反。需要注意的 是， IEEE 754 标准的浮点数尾数是用原码编码的。

4. 对于位数相同的定点数和浮点数，可表示的浮点数个数比定点数个数多吗?

不是，可表示的数据个数取决于编码所采用的位数。编码位数一定，编码出来的数据个数就

**第2章** **数据的表示和运算** -79

是 一 定 的 。n 位编码只能表示2”个数，所以对于相同位数的定点数和浮点数来说，可表示的数据 个数应该一样多(有时可能因为一个值有两个或多个编码对应，编码个数会有少量差异)。

5. 现代计算机中是否要考虑原码加减运算?如何实现?

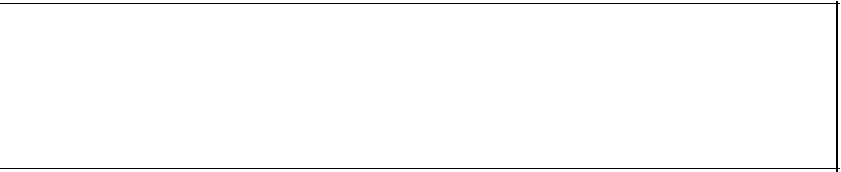
现代计算机中的浮点数采用 IEEE 754 标准，所以在进行两个浮点数的加减运算时，必须考 虑原码的加减运算，因为IEEE 754 标准的浮点数尾数都采用原码表示。

原码的加减运算可以有以下两种实现方式：

1)转换为补码后，用补码加减法实现，结果再转换为原码。

2)直接用原码进行加减运算，符号位和数值位分开处理(见原码加减运算部分)。





第 3 章

存储系统

**【考纲内容】**

( 一 )存储器的分类

(二)层次化存储器的基本结构

(三)半导体随机存储器 (RAM)

SRAM 、DRAM 、Flash 存储器

( 四 ) 主 存 储 器

DRAM 芯片和内存条、多模块存储器、主存储器和CPU 之间的连接 (五)外部存储器

磁盘存储器、固态硬盘 (SSD)

(六)高速缓冲存储器 (Cache)

Cache 的基本原理； Cache 和主存之间的映射方式

Cache 中主存块的替换算法； Cache 写策略

(七)虚拟存储器

虚拟存储器的基本概念

页式虚拟存储器：基本原理、页表、地址转换、TLB ( 快 表 ) 段式虚拟存储器的基本原理；段页式虚拟存储器的基本原理

**【复习提示】**

扫一扫



视频讲解

本章是历年命题重点，特别是有关 Cache 和虚拟存储器的考点容易出综合题。此外，存 储器的特点，存储器的扩展(芯片选择、连接方式、地址范围等),交叉存储器， Cache 的 相 关计算与替换算法，虚拟存储器与TLB 也容易出选择题。读者应在掌握基本原理的基础上，

多结合习题进行反复训练，以加深巩固。另外，读者需掌握存在Cache 和 TLB 的计算机中的 地址翻译与 Cache 映射问题，也建议结合《操作系统考研复习指导》复习。

在学习本章时，请读者思考以下问题：

1)存储器系统为何要分这些层次?计算机如何管理这些层次?

2 ) 影 响 Cache 性能的因素有哪些?

3)虚拟存储系统的页面是设置得大一些好还是设置得小一些好?

请读者在学习本章的过程中寻找答案，本章末尾会给出参考答案。



**3.1** **存储器概述**

**3.1.1** **存储器的分类**

存储器种类繁多，可从不同角度对存储器进行分类。

第3章 存储系统 81

**1.** **按在计算机中的作用(层次)分类**

1)主存储器。简称主存，也称内存储器(内存),用来存放计算机运行期间所需的程序和数 据， CPU 可以直接随机地对其进行访问，也可以和高速缓冲存储器 (Cache) 及辅助存储 器交换数据。其特点是容量较小、存取速度较快、每位的价格较高。

2 ) 辅 助 存储器。简称辅存，也称外存储器或外存，用来存放当前暂时不用的程序和数据， 以及一些需要永久性保存的信息。辅存的内容需要调入主存后才能被 CPU 访问。其特点 是容量大、存取速度较慢、单位成本低。

3)高速缓冲存储器。简称 Cache, 位于主存和CPU 之间，用来存放当前CPU 经常使用的指 令和数据，以便CPU 能高速地访问它们。Cache 的存取速度可与CPU 的速度相匹配，但 存储容量小、价格高。现代计算机通常将它们制作在CPU 中。

**2.** **按存储介质分类**

按存储介质，存储器可分为磁表面存储器( 磁盘、磁带)、磁芯存储器、半导体存储器 (MOS 型存储器、双极型存储器)和光存储器(光盘)。

**3.** **按存取方式分类**

命 题 道 踪▶ **采用随机存取的存储器(2011)**

1)随机存储器 (R AM)。 存储器的任何一个存储单元都可以随机存取，而且存取时间与存储 单元的物理位置无关。其优点是读/写方便、使用灵活，主要用作主存或高速缓冲存储器。

RAM 又分为静态 RAM 和动态 RAM ( 第2节会详细介绍)。

2)只读存储器 (ROM) 。 存储器的内容只能随机读出而不能写入。信息一旦写入存储器就固 定不变，即使断电，内容也不会丢失。因此，通常用它存放固定不变的程序、常数和汉 字字库等。它与随机存储器可共同作为主存的一部分，统一构成主存的地址域。

由 ROM 派生出的存储器也包含可反复重写的类型， ROM 和 RAM 的存取方式均为随机 存取。广义上的只读存储器已可通过电擦除等方式进行写入，其“只读”的概念没有保 留，但仍保留了断电内容保留、随机读取特性，但其写入速度比读取速度慢得多。

3)串行访问存储器。对存储单元进行读/写操作时，需按其物理位置的先后顺序寻址，包括 顺序存取存储器(如磁带)和直接存取存储器(如磁盘、光盘)。

顺序存取存储器的内容只能按某种顺序存取，存取时间的长短与信息在存储体上的物理位置 有关，其特点是存取速度慢。直接存取存储器既不像 RAM 那样随机地访问任何一个存储单元， 又不像顺序存取存储器那样完全按顺序存取，而是介于两者之间。存取信息时通常先寻找整个存 储器中的某个小区域(如磁盘上的磁道),再在小区域内顺序查找。

**4.** **按信息的可保存性分类**

断电后，存储信息即消失的存储器，称为易失性存储器，如 RAM。 断电后信息仍然保持的 存储器，称为非易失性存储器，如 ROM 、 磁表面存储器和光存储器。

若某个存储单元所存储的信息被读出时，原存储信息被破坏，则称为破坏性读出； 若读出时， 被读单元原存储信息不被破坏，则称为非破坏性读出。具有破坏性读出性能的存储器，每次读出 操作后，必须紧接一个再生的操作，以便恢复被破坏的信息。

**3.1.2** **存储器的性能指标**

存储器有三个主要性能指标，即存储容量、单位成本和存储速度。这三个指标相互制约，设

82-2025年计算机组成原理考研复习指导

计存储器系统所追求的目标就是大容量、低成本和高速度。

1 ) 存 储 容 量 = 存 储 字 数 × 字 长 ( 如 1M×8 位 )。单位换算： IB(Byte, 字 节 ) = 8b(bit, 位 ) 。

存储字数表示存储器的地址空间大小，字长表示 一 次存取操作的数据量。 2 ) 单 位 成 本 ： 每 位 价 格 = 总 成 本 / 总 容 量 。

3)存储速度：数据传输速率(每秒传送信息的位数) =数据的宽度/存取周期。

① 存 取 时 间(T₂): 存取时间是指从启动 一 次存储器操作到完成该操作所经历的时间，

分为读出时间和写入时间。

② 存 取 周 期 (Tm): 存取周期是指存储器进行 一 次完整的读/写操作所需的全部时间，即 连续两次独立访问存储器操作(读或写操作)之间所需的最小时间间隔。

③ 主 存 带 宽 ( Bm): 也 称数 据 传 输 速 率，表示每秒从主存进出信息的最大数量，单位为 字/秒、字节/秒 (B/s) 或位/秒 (b/s)。

存取时间不等于存取周期，通常存取周期大于存取时间。这是因为对任何 一种存储器，在读/ 写操作之后，总要有 一段恢复内部状态的复原时间。对于破坏性读出的存储器，存取周期往往比存 取时间大得多，甚至可达Tm=2T₂, 因为存储器中的信息读出后需要马上进行再生。

存取时间与存取周期的关系如图3 . 1所示。



下次存取

*t₃*

恢复时间

y

存取周期



存取时间

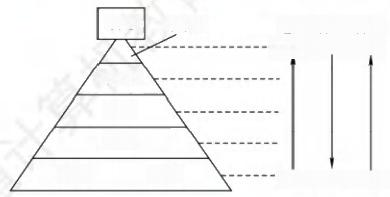
启动存取

存取完

图3.1 存取时间与存取周期的关系

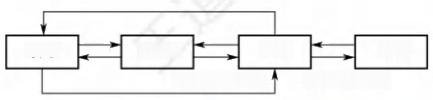
**3.1.3** **多级层次的存储系统**

为了解决存储系统大容量、高速度和低成本这三个相互制约的矛盾，在计算机系统中，通常 采用多级存储器结构，如图3 . 2所示。在图中由上至下，位价越来越低，速度越来越慢，容量越 来越大， CPU 访问的频度也越来越低。实际上，存储系统层次结构主要体现在 Cache- 主 存 层 和 主存 - 辅存层。在存储体系中， Cache 、 主 存 能 与 CPU 直接交换信息，辅存则要通过主存与 CPU 交换信息；主存与 CPU 、Cache 、 辅存都能交换信息，如图3 . 3 所 示 。



|  |  |
| --- | --- |
| CPU 寄存器  Cache  主存  磁盘  磁带 光盘 | 速度容量价格 -最快最小最高  最慢最大最低 |

图3.2 多级存储器结构



CPU Cache 主存 辅存

Cache-主存层 主存-辅存层

图3.3 三级存储系统的层次结构及其构成

存储器层次结构的主要思想是上 一 层的存储器作为低 一 层存储器的高速缓存。 当 **CPU** 要 从 存储器中存取数据时，先访问Cache, 若 不 在 Cache 中，则访问主存，若不在主存中，则访问磁 盘，此时，操作数从磁盘读出送到主存，然后从主存送到Cache。 从 CPU 的 角 度 看 ，Cache- 主 存 层的速度接近于Cache, 容量和位价却接近于主存。从主存 - 辅存层分析，其速度接近于主存，容

量和位价却接近于辅存。这就解决了速度、容量、成本这三者之间的矛盾。

Cache- 主存层主要解决 CPU 和主存速度不匹配的问题，主存和 Cache 之间的数据调动是由 硬件自动完成的， 对所有程序员均是透明的。主存-辅存层主要解决存储系统的容量问题，主存 和辅存之间的数据调动是由硬件和操作系统共同完成的， 对应用程序员是透明的。

在主存-辅存层的不断发展中，逐渐形成了虚拟存储系统，在这个系统中程序员编程的地址 范围与虚拟存储器的地址空间相对应，编程时可用的地址空间远大于主存空间。



**注** **意**

在 Cache-主存层和主存-辅存层中，上一层中的内容都只是下一层中的内容的副本，也即 Cache (或主存)中的内容只是主存(或辅存)中的内容的一部分。

**3.1.4** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 磁盘属于( )类型的存储器。

A. 随机存储器(RAM) B. 只读存储器 (ROM)

C. 顺序存取存储器 (SAM) D. 直接存取存储器 (DAM)

**02**. 存储器的存取周期是指( )。

A. 存储器的读出时间

B. 存储器的写入时间

C. 存储器进行连续读或写操作所允许的最短时间间隔

D. 存储器进行一次读或写操作所需的平均时间

**03.** 设机器字长为32位， 一个容量为16MB 的存储器，CPU 按半字寻址，其可寻址的单元

数是( )。

*A.224* B.223 C.222 D.22l

**04.** 相联存储器是按()进行寻址的存储器。

A. 地址指定方式

B. 堆栈存储方式

C. 内容指定方式和堆栈存储方式相结合

D. 内容指定方式和地址指定方式相结合

**05.** 在下列几种存储器中， CPU 不能直接访问的是()。

**A.** 硬盘 B. 内存 C.Cache D. 寄存器

**06.** 若某存储器存储周期为250ns, 每次读出16位，该存储器的数据传输速率是( )。

A.4×10B/s B. 16MB/s C.8×10°B/s D.8×22°B/s

**07.** 设机器字长为64位，存储容量为128MB, 若按字编址，它可寻址的单元个数是()。 A.16MB B.16M C.32M D.32MB

**08.** 计算机的存储器采用分级方式是为了()。

A. 方便编程 B. 解决容量、速度、价格三者之间的矛盾

C. 保存大量数据方便 D. 操作方便

**09.** 计算机的存储系统是指( )。

A.RAM B.ROM

C. 主存储器 D. 寄存器、Cache 、主存储器和外存储器

84 -2025年计算机组成原理考研复习指导

**10.** 在多级存储体系中， “Cache- 主存”结构的作用是解决()的问题。

A. 主存容量不足 B. 主存与辅存速度不匹配

C. 辅存与CPU 速度不匹配 D. 主存与CPU 速度不匹配

**11.** 存储器分层体系结构中，存储器从速度最快到最慢的排列顺序是( )。

A. 寄存器-主存-Cache- 辅存 B. 寄存器-主存-辅存-Cache

C. 寄存器-Cache- 辅存-主存 D. 寄存器-Cache- 主存-辅存

**12.** 在 Cache 和主存构成的两级存储体系中，主存与 Cache 同时访问，Cache 的存取时间是 100ns, 主存的存取时间是1000ns, 设 Cache 和主存同时访问，若希望有效(平均)存 取时间不超过Cache 存取时间的115%,则 Cache 的命中率至少应为()。

A.90% B.98% C.95% D.99% **13.** 下列关于多级存储系统的说法中，正确的有( )。

I. 多级存储系统是为了降低存储成本

II. 虚拟存储器中主存和辅存之间的数据调动对任何程序员是透明的

ⅢI.CPU 只 能 与Cache 直接交换信息， CPU 与主存交换信息也需要经过 Cache

A. 仅 I B. 仅 I 和 Ⅱ C.I 、Ⅱ 和 Ⅲ D. 仅 Ⅱ

**14.【**2011 统考真题】下列各类存储器中，不采用随机存取方式的是( )。

A.EPROM B.CD-ROM C.DRAM D.SRAM

**3.1.5** **答案与解析**

**单项选择题**

**01.D**

磁盘属于直接存取存储器，其速度介于随机存储器和顺序存取存储器之间。

**02.C**

存取时间 T₂是指从存储器读出或写入一次信息所需要的平均时间；存取周期 Tm是指连续两 次访问存储器之间所必需的最短时间间隔。对Tm一 般有Tm=T₄+T, 其 中T; 为复原时间；对SRAM 指存取信息的稳定时间，对DRAM 指刷新的又一次存取时间。D 指的是存取时间。

03.B

16MB=2⁴B, 字长为32位，现按半字(2B) 寻址，可寻址单元数为22\*B/2B=2²3。

04.D

相联存储器的基本原理是把存储单元所存内容的某一部分作为检索项(即关键字项)去检索 该存储器，并将存储器中与该检索项符合的存储单元内容进行读出或写入。所以它是按内容或地 址进行寻址的，价格较为昂贵。 一般用来制作 TLB、 相联 Cache 等。

**05.** A

CPU 不能直接访问硬盘，需先将硬盘中的数据调入内存才能被CPU 访问。

06.C

每个存储周期读出16bit=2B, 因此数据传输速率为2B÷(250×10-°)s, 即8×10°B/s。

07.B

机器字长位64位，即8B, 按字编址，因此可寻址的单元个数是128MB/8B=16M。

**08.B**

存储器有三个主要特性：速度、容量和价格/位(简称位价)。存储器采用分级方式是为了解 决这三者之间的矛盾。

**09.D**

计算机的存储系统包括 CPU 内部寄存器、Cache、 主存和外存。

**10.D**

Cache 中的内容只是主存内容的部分副本(拷贝),因而 “Cache- 主存”结构并未增加主存容

量，目的是解决主存与CPU 速度不匹配的问题。

**11.** D

在存储器分层结构中，寄存器在CPU 中，因此速度最快， Cache 次之，主存再次之，最慢的 是辅存(如磁盘、光盘等)。

12.D

假设命中率为x, 可得100x+1000(1-x)≤100×(1+15%), 简单计算后得结果为x≥98.33%, 因

此命中率至少为99%。



**注** **意**

本题采用同时访问Cache 和主存的方式，此时不命中的访问时间为1000ns, 但若题中没有 说明(通常有说明),则默认Cache 不命中的时间为访问Cache 和主存的时间之和。

**13.** A

主存和辅存之间的数据调动是由硬件和操作系统共同完成的，仅对应用级程序员透明。CPU

与主存可直接交换信息。

**14.B**

随机存取是指 CPU 可对存储器的任意一个存储单元中的内容随机存取，而且存取时间与存 储单元的物理位置无关。A 、C 和 D 均采用随机存取方式， CD-ROM 即光盘，采用串行存取方式 (直接存取)。注意， CD-ROM 是只读型光盘存储器，不属于只读存储器ROM。

**3.2** **主存储器**

**3.2.1** **SRAM 芯片和** **DRAM 芯片**

半导体存储器分为随机存储器 (RAM) 和只读存储器 (ROM) 。RAM 又分为静态随机存储器 (SRAM) 和动态随机存储器(DRAM), 主存储器主要由DRAM 实现，靠近处理器的那一层(Cache) 则 由SRAM 实现，它们都是易失性存储器。ROM 是非易失性存储器。

**1.SRAM 的工作原理**

通常把存放一个二进制位的物理器件称为存储元，它是存储器的最基本的构件。地址码相同 的多个存储元构成一个存储单元。若干存储单元的集合构成存储体。

静态随机存储器 (SR AM) 的存储元是用双稳态触发器(六晶体管 MOS) 来记忆信息的，静 态是指即使信息被读出后，它仍保持其原状态而不需要再生(非破坏性读出)。

SRAM 的存取速度快，但集成度低，功耗较大，价格昂贵， 一般用于高速缓冲存储器。

**2.DRAM 的工作原理**

与 SRAM 的存储原理不同，动态随机存储器(DRA M) 是利用存储元电路中栅极电容上的电 荷来存储信息的，DRAM 的基本存储元通常只使用一个晶体管，所以它比 SRAM 的密度要高很



86 —2025年计算机组成原理考研复习指导

多。相对于 SRAM 来 说 ，DRAM 具有集成度高、位价低和功耗低等优点，但DRAM 的存取速度 比 SRAM 慢，且必须定时刷新和读后再生， 一般用于大容量的主存系统。

▶ **需要刷新的存储芯片：SDRAM(2015** **)**

DRAM 电容上的电荷一般只能维持1～2ms, 因此即使电源不断电，信息也会自动消失。此 外，读操作会使其状态发生改变 ( 破坏性读出),需读后再生，这也是称其为动态存储器的原因。 刷新可以采用读出的方法进行，根据读出内容对相应单元进行重写，即读后再生。对同一行进行 相邻两次刷新的时间间隔称为刷新周期，通常取2ms 。常用的刷新方式有以下3种：

1)集中刷新：在一个刷新周期内，利用一段固定的时间，依次对存储器的所有行进行逐一 再生，在此期间停止对存储器的读/写操作，称为死时间，也称访存死区。优点是读/写操 作时不受刷新工作的影响；缺点是在集中刷新期间(死区)不能访问存储器。

2)分散刷新：将一个存储器系统的工作周期分为两部分：前半部分用于正常的读/写操作； 后半部分用于刷新。这种刷新方式增加了系统的存取周期，如存储芯片的存取周期为 0.5μs,则系统的存取周期为lμs 。 优点是没有死区；缺点是加长了系统的存取周期。

3)异步刷新：结合了前两种方法，使得在一个刷新周期内每一行仅刷新一次。具体做法是 将刷新周期除以行数，得到相邻两行之间刷新的时间间隔 t, 每隔时间t 产生一次刷新请 求。这样就使“死时间”的分布更加分散，避免让CPU 连续等待过长的时间。

DRAM 的刷新需要注意以下问题：①刷新对 CPU 是透明的，即刷新不依赖于外部的访问； ②DRAM 的刷新单位是行，由芯片内部自行生成行地址；③刷新操作类似于读操作，但又有所不 同。另外，刷新时不需要选片，即整个存储器中的所有芯片同时被刷新。



**注** **意**

虽然 DRAM 的刷新和再生都是恢复数据，但刷新与再生的过程并不完全相同。刷新是以 行为单位，逐行恢复数据的，而再生仅需恢复被读出的那些单元的数据。

▶ **DRAM 芯片行缓冲器容量的计算(2022)**

目前更常用的是 SDRAM(同步DRAM) 芯片，其工作方式与传统 DRAM 的不同，传统DRAM 与 CPU 采用异步方式交换数据， CPU 发出地址和控制信号后，经过一段延迟时间，数据才读出 或写入，在读/写完成之前， CPU 不能做其他工作。而 SDRAM 与 CPU 采用同步方式交换数据， 它将 CPU 发出的地址和控制信号锁存起来， CPU 在其读/写完成之前可进行其他操作。SDRAM 的每一步操作都在系统时钟的控制下进行，支持突发传输方式”。第一次存取时给出首地址，同 一行的所有数据都被送到行缓冲器，因此，以后每个时钟都可以连续地从SDRAM 输出一个数据。 行缓冲器用来缓存指定行中整行的数据，其大小为“列数×位平面数”,通常用SRAM 实现。

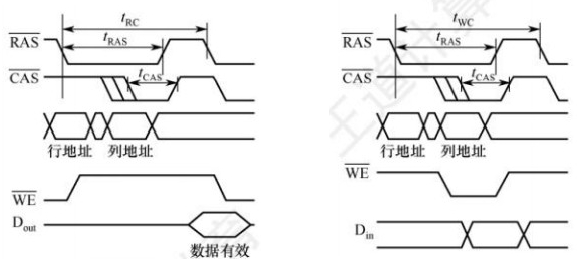
**3.DRAM 芯片的读/写周期**

DRAM 芯片读/写周期的时序图如图3.4所示。为了使芯片能正确接收行、列地址并实现读/ 写操作，各信号的时间关系应符合一定要求。读(写)周期时间tgc(twc) 表 示 DRAM 芯片进行

两次连续读(写)操作时所必须间隔的时间。

在读周期中，在RAS 有效前将行地址送到芯片的地址引脚，CAS 滞后RAS 一段时间，在CAS 有效前再将列地址送到芯片的地址引脚， RAS 、CAS 应分别至少保持 tgAs和 tcas 的时间。在读 周期中WE 为高电平，并在CAS 有效前建立。

① 突发传输方式是指在寻址阶段发送数据单元的首地址，在传输阶段传送多个连续单元的数据。



RAS 行选通信号

CAS 列选通信号

WE 读/写控制信号

(a) 读周期 (b) 写周期

图3 . 4 DRAM 芯 片 读 / 写 周 期 时 序 图



在写周期中，行列选通信号的时序关系和读周期相同。在写周期中 WE 为低电平，同样在CAS 有效前建立。为了保证数据可靠地写入，写数据必须在CAS 有效前在数据总线上保持稳定。

**4.SRAM 和** **DRAM 的比较**

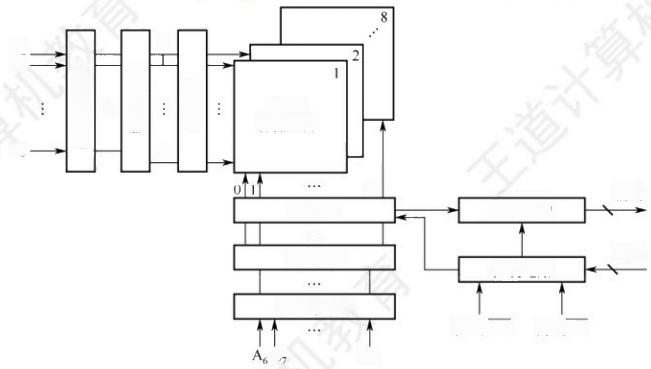
表3 . 1详细列出了SRAM 和 DRAM 各自的特点。

**表** **3** **.** **1** **SRAM** **和** **DRAM** **各** **自** **的** **特** **点**

|  |  |  |
| --- | --- | --- |
| **类型**  **特点** | **SRAM** | **DRAM** |
| 存储信息 | 触发器 | 电容 |
| 破坏性读出 | 非 | 是 |
| 需要刷新 | 不要 | 需要 |
| 送行列地址 | 同时送 | 分两次送(复用) |
| 运行速度 | 快 | 慢 |
| 集成度 | 低 | 高 |
| 存储成本 | 高 | 低 |
| 主要用途 | 高速缓存 | 主机内存 |

**5.** **存储器芯片的内部结构**

如 图 3 . 5 所 示 ， 存 储 器 芯 片 由 存 储 体 、I/O 读 / 写 电 路 、 地 址 译 码 器 和 控 制 电 路 等 部 分 组 成 。



A₀.

A₁

64×64

存储矩阵

As ·

输 出

输出驱动



输入

控制电路

地址寄存器

读/写WE 片 选CS

A, An

译码器

地址寄存器

驱动器

Y译码器

VO电 路

63

63

0

:

图 3 . 5 存 储 器 芯 片 结 构 图

88 2025年计算机组成原理考研复习指导

1)存储体(存储矩阵)。存储体是存储单元的集合，它由行选择线 (X) 和列选择线 (Y)

来选择所访问单元，存储体的相同行、列上的多位(位平面数)同时被读出或写入。

2)地址译码器。用来将地址转换为译码输出线上的高电平，以便驱动相应的读/写电路。地

址译码有单译码法( 一 维译码)和双译码法(二维译码)两种方式。

● 单译码法。只有一个行译码器，同一行中所有存储单元的字线连在一起，同一行中的 各单元构成一个字，被同时读出或写入。缺点是地址译码器的输出线数过多。

● 双译码法。如图3 .5所示，地址译码器分为X 和 Y 方向两个译码器，在选中的行和列 交叉点上能确定一个存储单元，这是DRAM 芯片目前普遍采用的译码结构。

3)I/O 控制电路。用以控制被选中的单元的读出或写入，具有放大信息的作用。

4)片选控制信号。单个芯片容量太小，往往满足不了计算机对存储器容量的要求，因此需 用一定数量的芯片进行存储器的扩展。在访问某个字时，必须“选中”该存储字所在的

芯片，而其他芯片不被“选中”,因此需要有片选控制信号。

5)读/写控制信号。根 据 CPU 给出的读命令或写命令，控制被选中单元进行读或写。

**3.2.2** **只** **读** **存** **储** **器**

**1.** **只读存储器** **(ROM) 的特点**

命 题 追 踪▶ **RAM 和** **ROM 的区别(2010)**

ROM 和 RAM 都是支持随机访问的存储器，其中SRAM 和 DRAM 均为易失性半导体存储器。 而 ROM 中一旦有了信息，就不能轻易改变，即使掉电也不会丢失。ROM 具有两个显著的优点： ①结构简单，所以位密度比可读/写存储器的高。②具有非易失性，所以可靠性高。

**2.ROM** **的类型**

根据制造工艺的不同， ROM 可分为掩模式只读存储器 (MROM) 、 一 次可编程只读存储器 (PROM) 、 可擦除可编程只读存储器 (EPROM) 、Flash 存储器和固态硬盘 (SSD)。

(1)掩模式只读存储器

MROM 的内容由半导体制造厂按用户提出的要求在芯片的生产过程中直接写入，写入以后 任何人都无法改变其内容。优点是可靠性高，集成度高，价格便宜；缺点是灵活性差。

(2) 一 次可编程只读存储器

PROM 是可以实现一次性编程的只读存储器。允许用户利用专门的设备 ( 编 程 器) 写入自己 的程序， 一旦写入，内容就无法改变。

(3)可擦除可编程只读存储器

EPROM 不仅可以由用户利用编程器写入信息，而且可以对其内容进行多次改写。EPROM 虽 然既可读又可写，但它不能取代RAM, 因 为EPROM 的编程次数有限，且写入时间过长。

(4)Flash 存储器

命 题 追 踪 ▶ **Flash** **存储器的特点(2012)**

Flash 存储器是在 EPROM 的基础上发展起来的，它兼有 ROM 和 RAM 的优点，可在不加电 的情况下长期保存信息，又能在线进行快速擦除与重写。Flash 存储器既有EPROM 价格便宜、集 成度高的优点，又有 E²PROM 电可擦除重写的特点，且擦除重写的速度快。

(5)固态硬盘 (Solid State Drive,SSD)

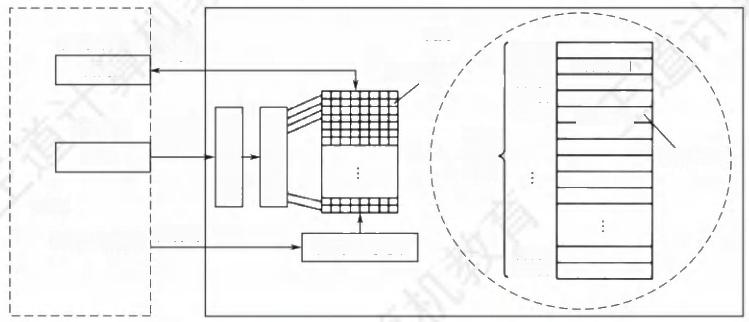
基于闪存的固态硬盘是用固态电子存储芯片阵列制成的硬盘，由控制单元和存储单元 (Flash

第3章 存储系统 89

芯片)组成。保留了 Flash 存储器长期保存信息、快速擦除与重写的特性。对比传统硬盘也具有 读/写速度快、低功耗的特性，缺点是价格较高。

**3.2.3** **主存储器的基本组成**

图3 . 6是主存储器 (Main Memory,MM) 的基本框图，其中由 一 个个存储0或1的记忆单元 (也称存储元件)构成的存储矩阵(也称存储体)是存储器的核心部件。 存储元件是具有两种稳 态的能表示二进制0和1的物理器件。为了存取存储体中的信息，必须对存储单元编号(也称编 址)。编址单位是指具有相同地址的那些存储元件构成的一个单位，可以按字节编址，也可以按 字编址。现代计算机通常采用字节编址方式，此时存储体内的一个地址中有1字节。



读/写的数据

MDR

10101010

主存地址

MAR

CPU

读/写控制信号

地址译码器

地址寄存器

00000 [ 00001 00010 00011 00100|

MM

记忆单元

存储 单元< 地址

数据线 (64位)

地址线 (36位)

读/写控制电路

11110[ 11111

存储内容

控制线

01101001

图3.6 主存储器的基本组成框图

▶ **MAR 和** **MDR 位数与地址线/数据线数的关系、寻址范围的计算(2021)**

指令执行过程中需要访问主存时， CPU 首先把被访问单元的地址送到MAR 中，然后通过地 址线将主存地址送到主存中的地址寄存器，以便地址译码器进行译码，选中相应单元，同时 CPU 将读/写信号通过控制线送到主存的读/写控制电路。若是写操作，则CPU 同时将要写的信息送到 MDR 中，在读/写控制电路的控制下，经数据线将信号写入选中的单元；若是读操作，则主存读 出选中单元的内容送至数据线，然后被送到 MDR 中 。MDR的位数与数据线的位数相同，MAR 的位数与地址线的位数相同。图3 .6采用64位数据线，所以在按字节编址方式下，每次最多可以 存取8个单元的内容。地址线的位数决定了主存地址空间的最大可寻址范围。例如，36位地址的 最大寻址范围为0～236- 1,即地址从0开始编号。



**注** **意**

数据线的位数通常等于存储字长，因此 MDR 的位数通常等于存储字长；若数据线的位数 不等于存储字长，则 MDR 的位数由数据线的位数决定。

**▶** **DRAM 芯片的地址引脚复用技术(2014)**

DRAM 芯片容量较大，地址位数较多，为了减少芯片的地址引脚数，通常采用地址引脚复用 技术，行地址和列地址通过相同的引脚分先后两次输入，这样地址引脚数可减少一半。

▶ **RAM 芯片行、列数的优化原则(2018)**

假定有一个2°×b 位 DRAM 芯片的存储阵列，其行数为r, 列 数 为c, 则 2 " =rxc 。 存储阵列



90- 2025年计算机组成原理考研复习指导

的地址位数为n, 其中行地址位数为logzr, 列地址位数为log₂c, 则 n=logr+log,c 。 由 于 DRAM

芯片采用地址引脚复用技术，为减少地址引脚数，应尽量使行、列位数相同，即满足 |r-c| 最小。 又由于 DRAM 按行刷新，为减少刷新开销，应使行数较少，因此还需满足r≤c 。

**3.2.4** **多模块存储器**

多模块存储器是一种空间并行技术，利用多个结构完全相同的存储模块的并行工作来提高存储 器的吞吐率。常用的有单体多字存储器和多体低位交叉存储器。



**注** **意**

CPU 的速度比存储器快得多，若同时从存储器中取出n 条指令，就可以充分利用 CPU 资 源，提高运行速度。多体交叉存储器就是基于这种思想提出的。

**1.** **单体多字存储器**

在单体多字系统中，每个存储单元存储m 个字，总线宽度也为m 个字， 一 次并行读出m 个 字。在一个存取周期内，从同一地址取出m 条指令，然后将指令逐条送至CPU 执行，即每隔1/m 存取周期， CPU 向主存取一条指令。这显然提高了单体存储器的工作速度。

缺点：只有指令和数据在主存中连续存放时，这种方法才能有效提升存取速度。 一旦遇到转 移指令，或操作数不能连续存放时，这种方法的提升效果就不明显。

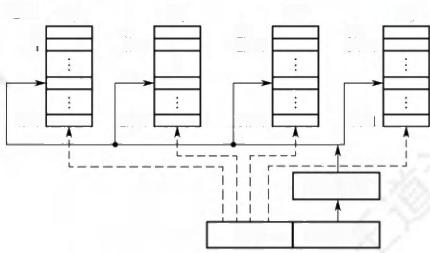
**2.** **多体并行存储器**

多体并行存储器由多体模块组成。每个模块都有相同的容量和存取速度，各模块都有独立的 读/写控制电路、地址寄存器和数据寄存器。它们既能并行工作，又能交叉工作。

多体并行存储器分为高位交叉编址和低位交叉编址两种。

(1)高位交叉编址(顺序方式)

高位地址表示模块号(或体号),低位地址为模块内地址(或体内地址)。如图3 . 7所示，存 储器共有4个模块 M₀~M₃, 每个模块有n 个单元，各模块的地址范围如图中所示。



地址 M₀

9|

:

n-1 2n-1 3m-1 *4n-1*

地址译码

体内地址

M₃ 3n

3n+1

M₂ 2n

2n+1

n

n+1

体号

M₁

图3.7 高位交叉编址的多体存储器

在高位交叉方式下，总把低位的体内地址送到由高位体号确定的模块内进行译码。访问一个 连续主存块时，总是先在一个模块内访问，等到该模块访问完才转到下一个模块访问， CPU 总 是 按顺序访问存储模块，各模块不能被并行访问，因而不能提高存储器的吞吐率。



**注** **意**

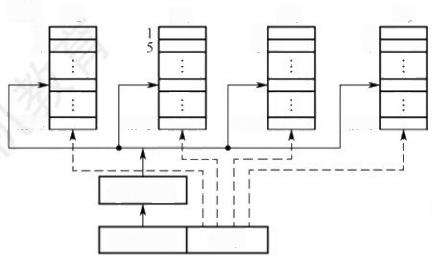
模 块 内 的 地 址 是 连 续 的 ， 存 取 方 式 仍 是 串 行 存 取 ， 因 此 这 种 存 储 器 仍 是 顺 序 存 储 器 。

37

(2)低位交叉编址(交叉方式)

命 题 追 踪 交叉存储器中数据的存放方式(2017)

低位地址为模块号，高位地址为模块内地址。如图3 . 8所示，每个模块按“模m” 交 叉 编 址 ， 模 块 号 = 单 元 地 址 %m, 假 定 有m 个模块，每个模块有k 个单元，则单元0,m,…,(k-1)m 位 于 M₀; 单元1,m+1,…,(k-1)m+1 位于M₁; 以此类推。



M₀ M₁ M₂ M₃



4n-4 4n-3| *4n-2* 4n-1

地址译码

体内地址 体号

地址 0 4

2 9

图3.8 低位交叉编址的多体存储器

低位交叉方式下，总是把高位的体内地址送到由低位体号所确定的模块内进行译码。程序 连续存放在相邻模块中，因此称采用此编址方式的存储器为交叉存储器。

交叉存储器可以采用轮流启动或同时启动两种方式。

● 轮流启动方式

若每个模块 一 次读/写的位数正好等于数据总线位数，模块的存取周期为 T, 总 线 周 期 为r, 为实现轮流启动方式，存储器交叉模块数应大于或等于

*m =T/r*

命 题 追 踪 ▶ **交叉存储器存取时间和带宽的计算(2012、2013)**

按每隔1/m 个存取周期轮流启动各模块，则每隔1/m 个存取周期就可读出或写入 一 个数据， 存 取 速 度 提 高m 倍，图3 . 9展示了4体交叉轮流启动的时间关系。交叉存储器要求其模块数大 于 或 等 于m, 以保证启动某模块后经过m×r 的时间后再次启动该模块时，其上次的存取操作已经 完成(以保证流水线不间断)。这样，连续存取m 个字所需的时间为

t₁=T+(m-1)r

而顺序方式连续读取m 个 字 所 需 的 时 间 为t₂=mT 。 可见交叉存储器的带宽大大提高。

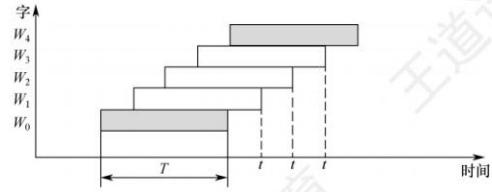


图3.9 低位交叉轮流启动的存取时间示意图

▶ **交叉存储器中访存冲突的分析(2015)**

在理想情况下， m 体交叉存储器每隔1/m 存取周期可读/写 一 个数据，若相邻的m 次 访 问 的

92 -2025年计算机组成原理考研复习指导

访存地址出现在同一个模块内，则会发生访存冲突，此时需延迟发生冲突的访问请求。

●同时启动方式

若所有模块一次并行读/写的总位数正好等于数据总线位数，则可以同时启动所有模块进行 读/写。设每个模块一次读/写的位数为16位，模块数m=4, 数据总线位数为64位，4个模块 一共提供64位，正好构成一个存储字，因此应该同时启动4个模块进行并行读/写。

**3.2.5** **本** **节** **习** **题** **精** **选**

**一、单项选择题**

**01.** 某一 SRAM 芯片，容量为1024×8位，该芯片的地址引脚和数据引脚总数至少是()。 A.8 B.10 C.18 D.13

02. 某存储器容量为32K×16位，则( )。

A. 地址线为16根，数据线为32根 B. 地址线为32根，数据线为16根

C. 地址线为15根，数据线为16根 D. 地址线为15根，数据线为32根

03.DRAM 的刷新是以( )为单位的。

A. 存储单元 B. 行 C. 列 D. 存储字

04.DRAM 采用下列哪种刷新方式时，不存在死时间()。

A. 集中刷新 B. 分散刷新 C. 异步刷新 D. 都不对

05. 下面是有关 DRAM 和 SRAM 存储器芯片的叙述：

I.DRAM 芯片的集成度比 SRAM 芯片的高

II.DRAM 芯片的成本比 SRAM 芯片的高

Ⅲ.DRAM 芯片的速度比 SRAM 芯片的快

IV.DRAM 芯片工作时需要刷新， SRAM 芯片工作时不需要刷新

通常情况下，错误的是()。

A.I 和Ⅱ B.Ⅱ 和ⅢⅢ C.Ⅲ 和IV D.I 和IV

**06.** 下列关于随机存储器的说法中，正确的是( )。

A. 半导体 RAM信息可读可写，且断电后仍能保持记忆

B.DRAM 是易失性RAM, 而 SRAM 中的存储信息是不易失的

C. 半导体 RAM是易失性 RAM, 但只要电源不断电，所存信息是不丢失的 D. 半导体 RAM 是非易失性 RAM

**07.** 下列关于存储器的说法中，不正确的是()。

A. 随机存储器和只读存储器不可以统一编址

B. 在访问随机存储器时，访问时间与存储单元的物理位置无关

C. 随机存储器 RAM 芯片可随机存取信息，掉电后信息会丢失

D. 只读存储器 ROM 芯片可随机存取信息，掉电后信息不会丢失

**08.** 关于半导体存储器的组织，下列选项中()是不正确的。

A. 在同一个存储器中，每个存储单元的宽度可以不同

B. 所谓“编址”,是指给每个存储单元一个编号

C. 存储器的核心部分是存储阵列，由若干存储单元构成

D. 每个存储单元由若干存储元件构成，每个存储元件存储一个比特位 **09.** 关于SRAM 和 DRAM, 下列叙述中正确的是( )。

A. 通常 SRAM 依靠电容暂存电荷来存储信息，电容上有电荷为1,无电荷为0

B.DRAM 依靠双稳态电路的两个稳定状态来分别存储0和1

C.SRAM 速度较慢，但集成度稍高； DRAM 速度稍快，但集成度低 D.SRAM 速度较快，但集成度稍低； DRAM 速度稍慢，但集成度高

10. 某一 DRAM 芯片，采用地址复用技术，容量为1024×8位，该芯片的地址引脚和数据引

脚总数至少是( )。

A.18 B.13 C.8 D.17

**11.** 下列几种存储器中，()是易失性存储器。

A.Cache B.EPROM C.Flash存储器 D.CD-ROM

**12.** U 盘属于( )类型的存储器。

A. 高速缓存 B. 主存 C. 只读存储器 D. 随机存储器

**13.** 某计算机系统，其操作系统保存于硬盘上，其内存储器应该采用()。

A.RAM B.ROM C.RAM 和 ROM D. 均不完善

**14.** 下列说法正确的是( )。

A.EPROM 是可改写的，因此可以作为随机存储器

B.EPROM 是可改写的，但不能作为随机存储器

C.EPROM 是不可改写的，因此不能作为随机存储器

D.EPROM 只能改写一次，因此不能作为随机存储器

**15.** 下列( )是动态半导体存储器的特点。

I. 在工作中存储器内容会产生变化

II. 每隔一定时间，需要根据原存内容重新写入一遍

Ⅲ.一次完整的刷新过程需要占用两个存储周期

IV. 一次完整的刷新过程只需要占用一个存储周期

A.I 、Ⅲ B.Ⅱ 、Ⅲ C.I 、IV D. 只 有 Ⅲ

**16.** 采用64K×1 位的 DRAM 芯片构成256K×8位的存储器，若采用异步刷新方式，每单元 刷新间隔不超过2ms, 则生成的刷新信号的间隔时间是();若采用集中刷新方式，则 存储器刷新一遍最少用( )个读/写周期。

A.7.8μs,256 B.1.9μs,256 C.7.8μs,128 D.1.9μs,256 17.DRAM 具有破坏性读出的特性，需要定时刷新，下列说法中不正确的是()。

A. 刷新是以行为单位的

B. 刷新是为了给DRAM 存储单元中的存储电容重新充电

C. 刷新是通过对存储单元进行“读但不输出数据”,即“假读”的操作来实现的

D.DRAM 内部设有专门的刷新电路，不会影响到 CPU 的正常访存 **18.** 下列关于 DRAM 和 SDRAM 的说法中，不正确的是()。

A. 传统 DRAM 芯片与 CPU 采用异步方式交换数据

B.SDRAM 芯片与 CPU 采用同步方式交换数据

C.DRAM 需要定期刷新，而 SDRAM 不需要定期刷新

D.SDRAM 的行缓冲器通常用 SRAM 实现

**19.** 每推出新一代DRAM 芯片，地址线至少增1根，则容量至少提高到原来的( )倍。

A.2 B.4 C.8 D.16

**20.** 若一个内存条中有16个 DRAM 芯片，每个芯片中有4个位平面，每个位平面的存储阵 列为4096行×4096列，则内存条的总容量为( ) MB。

94 —2025 年计算机组成原理考研复习指导

A.64 B.128 C.256 D.512

**21.** 已知单个存储体的存储周期为110ns, 总线传输周期为10ns, 采用低位交叉编址的多模 块存储器时，存储体数应( )。

A. 小于11 B. 等于11 C. 大于11 D. 大于或等于11

**22.** 一个四体并行低位交叉存储器，每个模块的容量是64K×32 位，存取周期为200ns, 总 线周期为50ns, 在下述说法中，( )是正确的。

A. 在200ns 内，存储器能向 CPU 提供256位二进制信息

B. 在200ns 内，存储器能向CPU 提供128位二进制信息

C. 在 5 0ns 内，每个模块能向 CPU 提供32位二进制信息

D. 以上都不对

**23.** 某机器采用四体低位交叉存储器，现分别执行下述操作：①读取6个连续地址单元中存 放的存储字，重复80次；②读取8个连续地址单元中存放的存储字，重复60次。则①、 ② 所花费的时间之比为()。

A.1:1 B.2:1 C.4:3 D.3:4

**24.** 假定用若干16K×8 位的存储芯片组成一个64K×8 位的存储器，芯片各单元采用交叉编 址方式，则地址 BFFFH 所在的芯片的最小地址为( )。

A.0000H B.0001H C.0002H D.0003H

**25.** 下列关于单体多字存储器的说法中，不正确的是(一)。

A. 单体多字存储器主要解决主存容量太小的问题

B. 单体多字存储器中，每个存储单元存储多个字

C. 指令与数据的连续存放有利于单体多字存储器提高主存的读/写速度

D. 过多的跳转指令会严重影响单体多字存储器的工作效率

**26.** 多模块存储器之所以能提高存储器的访问速度，是因为( )。

A. 采用了高速元器件 B. 各模块有独立的读/写电路

C. 采用了信息预读技术 D. 模块内各单元地址连续

**27.【2** 010统考真题】下列有关 RAM 和 ROM 的叙述中，正确的是( )。 I.RAM 是易失性存储器， ROM 是非易失性存储器

I.RAM 和 ROM 都采用随机存取方式进行信息访问

Ⅲ.RAM 和 ROM 都可用作 Cache

IV.RAM 和 ROM 都需要进行刷新

A. 仅 I 和 Ⅱ B. 仅 Ⅱ 和 Ⅲ C. 仅 I 、Ⅱ 和 I D. 仅Ⅱ、Ⅲ和 IV

**28.【** 2012统考真题】下列关于闪存的叙述中，错误的是( )。

A. 信息可读可写，并且读、写速度一样快

B. 存储元由MOS 管组成，是一种半导体存储器

C. 掉电后信息不丢失，是一种非易失性存储器

D. 采用随机访问方式，可替代计算机外部存储器

**29.【**2014统考真题】某容量为256MB 的存储器由若干4M×8 位的 DRAM 芯片构成，该 DRAM 芯片的地址引脚和数据引脚总数是()。

A.19 B.22 C.30 D.36

**30.【2** 015统考真题】下列存储器中，在工作期间需要周期性刷新的是()。

A.SRAM B.SDRAM C.ROM D.Flash

**31.【** 2015 统考真题】某计算机使用四体交叉编址存储器，假定在存储器总线上出现的主存 地址(十进制)序列为8005,8006,8007,8008,8001,8002,8003,8004,8000,则可能发生 访存冲突的地址对是()。

A.8004 和8008 B.8002 和8007 C.8001 和8008 D.8000 和8004

**32.** 【2017统考真题】某计算机主存按字节编址，由4个64M×8 位的 DRAM 芯片采用交叉

编址方式构成，并与宽度为32位的存储器总线相连，主存每次最多读/写32位数据。若 double 型变量x 的主存地址为804001AH, 则读取x 需要的存储周期数是( )。

A.1 B.2 C.3 D.4

**33.**【2018统考真题】假定 DRAM 芯片中存储阵列的行数为 r 、列数为 c, 对于 一 个2K×1 位的 DRAM 芯片，为保证其地址引脚数最少，并尽量减少刷新开销，则r 、c 的取值分 别 是 ( ) 。

A.2048,1 B.64,32 C.32,64 D.1,2048

**34.** 【2022统考真题】某内存条包含8个8192×8192×8位的 DRAM 芯片，按字节编址，支 持突发 (burst) 传送方式，对应存储器总线宽度为64位，每个 DRAM 芯片内有一个行 缓冲区 (row buffer)。下列关于该内存条的叙述中，不正确的是()。

A. 内存条的容量为512 MB B. 采用多模块交叉编址方式

C. 芯片的地址引脚为26位 D. 芯片内行缓冲有8192×8位

**二** **、综合应用题**

**01.** 在显示适配器中，用于存放显示信息的存储器称为刷新存储器，它的重要性能指标是带 宽。具体工作中，显示适配器的多个功能部分要争用刷新存储器的带宽。设总带宽50% 用于刷新屏幕，保留50%的带宽用于其他非刷新功能，且采用分辨率为1024×768像素、

颜色深度为3B 、刷新频率为72Hz 的工作方式。

1)试计算刷新存储器的总带宽。

2)为达到这样高的刷新存储器带宽，应采取何种技术措施?

**02.** 一个四体并行交叉存储器，每个模块的容量是64K×32位，存取周期为200ns, 问 ： 1)在一个存取周期中，存储器能向CPU 提供多少位二进制信息?

2)若存取周期为400ns, 则在0.1μs 内存储器可向CPU 提供32位二进制信息，该说法 正确否?为什么?

**03.** 设存储器容量为32个字，字长为64位，模块数m=4, 分别采用顺序方式和交叉方式 进行组织。存取周期T=200ns, 数据总线宽度为64位，总线传输周期r=50ns 。 在连 续读出4个字的情况下，求顺序存储器和交叉存储器各自的带宽。

**04** . 某计算机字长32 位，存储体的存储周期为200ns。

1)采用四体交叉工作，用低2位的地址作为体地址，存储数据按地址顺序存放。主机

最快多长时间可以读出一个数据字?存储器的带宽是多少?

2)若4个体分别保存主存中前1/4、次1/4、再下个1/4、最后1/4这四段的数据，即选

用高2位的地址作为体地址，可以提高存储器顺序读出数据的速度吗?为什么? 3)若把存储器改成单体4字宽度，会带来什么好处和问题?

4)比较采用四体低位地址交叉的存储器和四端口读出的存储器这两种方案的优缺点。

**05.** 假定一个存储器系统支持四体交叉存取，某程序执行过程中访问地址序列为3,9,17,2,51, 37,13,4,8,41 ,67,10,哪些地址访问可能会发生体冲突?

**3.2.6** **答** **案** **与** **解** **析**

**一** **、单项选择题**

**01.C**

芯片容量为1024×8位，8位说明数据线要8根，地址线数要10根(1024=2¹⁰)。故该芯片 的地址引脚和数据引脚总数至少需要18根。

**02.** C

该芯片16位，所以数据线为16根，寻址空间32K=215, 所以地址线为15根。

03.B

DRAM 的刷新按行进行。

04.B

集中刷新必然存在死时间。采用分散刷新时，机器的存取周期中的一段用来读/写，另一段用 来刷新，因此不存在死时间，但存取周期变长。异步刷新虽然缩短了死时间，但死时间依然存在。

05.B

DRAM 芯片的集成度高于 SRAM,I 正确； SRAM 芯片的速度高于 DRAM,Ⅲ 错误；可以 推出 DRAM 芯片的成本低于 SRAM,Ⅱ 错误； SRAM 芯片工作时不需要刷新，DRAM 芯片工作 时需要刷新， IV 正确。本题要求选择描述错误的表述，故选Ⅱ和Ⅲ。

06.C

RAM 属于易失性半导体， SRAM 和 DRAM 的区别在于是否需要动态刷新。

07.A

主 存 由RAM 和 ROM 构成，两者统一编址，A 错误。B 描述的是随机访问特性，正确。RAM 芯片具有随机访问特性和易失性， C 正确。ROM 芯片具有随机访问特性和非易失性，D 正确。

08.A

同一个存储器中，每个存储单元的宽度必须相同，即每个存储单元存储的比特位数必须相同。

09.D

SRAM 依靠双稳态电路的两个稳定状态来分别存储0和1;SRAM 速度较快，不需要动态刷 新，但集成度稍低，功耗大，单位价格高。DRAM 依靠电容暂存电荷来存储信息，电容上有电荷 为1,无电荷为0;DRAM 集成度高，功耗小，单位价格较低，需定时刷新，速度慢。

**10.B**

1024×8位，寻址范围是1024=2¹°。采用地址复用技术时，分两次传送行、列地址，地址引

脚减半为5根，数据引脚仍为8根，因此地址引脚和数据引脚总数至少为13根。

注意 SRAM 和 DRAM 的区别， DRAM 采用地址复用技术，而SRAM 不采用。

11.A

Cache 由 SRAM 组成，掉电后信息即消失，属于易失性存储器。

**12.** C

U 盘采用 Flash 存储器技术，它是在 E²PROM 的基础上发展起来的，属于ROM 的一种。由 于擦写速度和性价比均很可观，因此其常用作辅存。



**注意**

随机存取与随机存储器 (RAM) 不同，只读存储器 (ROM) 也是随机存取的。因此，支

持随机存取的存储器并不一定是 RAM。

**13.C**

因计算机的操作系统保存于硬盘上，所以需要 BIOS 的引导程序将操作系统引导到主存 (RAM) 中，而引导程序则固化于 ROM 中。

14.B

EPROM 可多次改写，但改写较为烦琐，写入时间过长，且改写的次数有限，速度较慢，因 此不能作为需要频繁读/写的 RAM 使 用。

15.C

动态半导体存储器利用电容存储电荷的特性记录信息，由于电容会放电，因此必须在电荷流 失前对电容充电，即刷新。方法是每隔一定的时间，根据原存内容重新写入一遍，因此1错误。 这里的读并不是把信息读入CPU, 也不是从 CPU 向主存存入信息，它只是把信息读出，通过一 个刷新放大器后又重新存回存储单元，而刷新放大器是集成在 RAM 上的。因此，这里只进行了 一次访存，也就是占用一个存取周期，Ⅱ、IV 正确，Ⅲ错误。

**16.** A

64K×1 位的 DRAM 芯片由一个256×256的位平面组成。 构成存储器的所有芯片同时按行刷 新，每个芯片有256行，故存储器所有单元刷新一遍至少需要256次刷新操作。若采用异步刷新 方式，则相邻两次刷新信息的时间间隔为2ms/256≈7.8μs。若采用集中刷新方式，则整个存储器 刷新一遍最少需256个读/写周期，在刷 新过程中，存储器不能进行读/写操作。

17.D

刷新也是一个读取的过程，根据读出内容对相应单元进行重写，因此会和CPU 的访存冲突， 会有访存“死时间”。刷新是指每隔一定的时间必须向栅极电容补充一次电荷，并以行为单位。

18.C

SDRAM (同步 DRAM) 与 SRAM 不同，其与 CPU 采用同步方式交换数据。SDRAM 也是 DRAM 的一种，需要定期刷新。行缓冲器用来缓存指定行中整行的数据，通常用SRAM 实现。

19.B

DRAM 芯片采用地址线复用技术，行地址和列地址分时复用，每增加1根地址线，则行地址 和列地址各增加1位，所以行数和列数各增加1倍，因此容量至少提高到原来的4倍。

20.B

DRAM 芯片的容量=位平面数×行数×列数，即由位平面数、存储阵列的行数和列数决定。 故一个 DRAM 芯片的容量为4096×4096×4b=8MB, 故内存条的总容量为8MB×16=128MB 。

21.D

低位交叉编址多模块存储器，采用轮流启动的方式时，类似于流水线的工作方式，为保证某 个模块再次启动时，其上次的存取操作已完成(流水线不间断),要求两次启动间隔的时间必须 大于或等于一个存储周期，即“模块数×总线周期≥存储周期”,得出存储体数应大于或等于11。

**22.B**

低位交叉存储器采用流水线技术，可以在一个存取周期内连续访问4个模块，32位×4=128 位。本题答案为B。

注：本题若作为计算题来考虑，从第一个字的读/写请求发出，到第4个字读/写结束，共需 要350ns, 但这里考查的是整体工作性能，可从以下角度理解：

1)连续取m 个字耗时t=T+(m-1)r, 平均每个字的存取时间是t₁/m, 实际工作时m 非常大，

因此t/m 也就非常接近r, 可认为存储器在每个总线周期r 都能给 CPU 提供一个字。

2)流水线充分流动起来后，每个总线周期后都能完成一个字的读/写，所以本题中每4个总

98 -2025年计算机组成原理考研复习指导

线周期(200ns) 都能完成4个字的读/写。

**23.** C

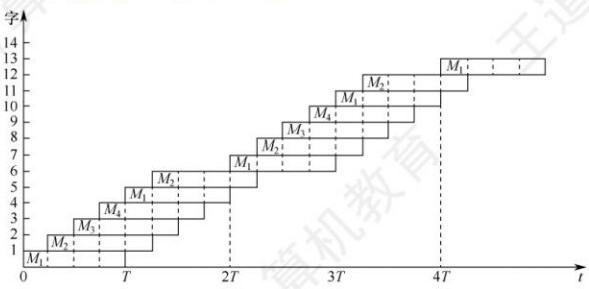
1)在每轮读取存储器的前6个 T/4 时 间 ( 共 3T/2) 内，依次进入各体。下一轮欲读取存储 器时，最近访问的M₁ 还在占用中(才过T/2 的时间),因此必须再等待 T/2 的时间才能开 始新的读取 (M₁ 连续完成两次读取，也即总共2T 的时间才可进入下一轮)。



**注** **意**

进入下 一轮不需要第6个字读取结束，第5个字读取结束时 M₁ 就已空出，即可马上进入 下一轮。

最后一轮读取结束的时间是本轮第6个字读取结束，共(6- 1)×(T/4)+T=2.25T。 情况1)的总时间为(80- 1)×2T+2.25T=160.25T。



2)每轮读取8个存储字刚好经过2T 的时间，每轮结束后，最近访问的 M₁ 刚好经过了时间 T, 此时可以立即开始下一轮的读取。

最后一轮读取结束的时间是本轮第8个字读取结束，共(8- 1)×(T/4)+T=2.75T。

情况2)的总时间为(60- 1)×2T+2 .75T=120.75T。

因此情况1)和2)所花费的总时间比为4:3。

**24.** D

64K×8 位/16K×8 位=4,可知芯片数为4。芯片各单元采用交叉编址，所以每个芯片的片选

信号由最低两位地址确定，高14 位为片内地址。4个芯片内各存储单元的最低两位地址分别为

00、01、10、11,即最小地址分别为0000H、0001H、0002H、0003H。 地 址 BFFFH 最低两位为

11,因此该存储单元所在芯片的最小地址为0003H。

25.A

单体多字存储器主要解决访存速度的问题，并没有解决主存容量太小的问题。在单体多字存 储器中，每个存储单元存储多个字，当指令和数据连续存放，且没有过多的跳转指令时，单体多 字存储器能有效地提高主存的读/写速度。

**26.B**

多模块存储器各模块有独立的读/写电路，可以实现并行操作，故多模块存储器能进行高速 的读/写操作。采用低位交叉编址的多模块存储器各单元地址不连续。

27.A

RAM ( 分DRAM 和 SRAM) 断电后会失去信息，而 ROM 断电后不会丢失信息，它们都采 用随机存取方式。Cache 一般采用高速的 SRAM 制成，而ROM 只可读，不能用作Cache,Ⅲ 错

误。DRAM 需要定期刷新，而ROM 不需要刷新，故IV 错误。

**28.** A

闪存是 E²PROM 的进一步发展，可读可写，用MOS 管的浮栅上有无电荷来存储信息。闪存 依然是 ROM 的一种，写入时必须先擦除原有数据，所以写速度要比读速度慢。闪存是一种非易 失性存储器，它采用随机访问方式。现在常见的 SSD 固态硬盘，即由Flash 芯片组成。

**29.** A

4M×8 位的芯片数据线应为8根，地址线应为log₂4M=22 根，而DRAM 采用地址复用技术， 地址线是原来的1/2,且地址信号分行、列两次传送。地址线数为22/2=11 根，所以地址引脚与 数据引脚的总数为11+8=19根，选A 。此题需要注意 DRAM 采用的是传两次地址的策略，所 以地址线为正常的一半，这是很多考生容易忽略的地方。

30.B

DRAM 使用电容存储，所以必须隔一段时间刷新一次，若存储单元未被刷新，则存储的信息 就会丢失。同步动态随机存储器 SDRAM 是现在最常用的一种 DRAM。

**31.** D

每个访存地址对应的存储模块序号(0,1,2,3)如下所示：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 访存地址 | 8005 | 8006 | 8007 | 8008 | 8001 | 8002 | 8003 | 8004 | 8000 |
| 模块序号 | 1 | 2 | 3 | 0 | 1 | 2 | 3 | 0 | 0 |

其中，模块序号=访存地址%存储器交叉模块数。

判断可能发生访存冲突的规则如下：给定的访存地址在相邻的四次访问中出现在同一个存储 模块内。据此，根据上表可知8004和8000对应的模块号都为0,即表明这两次的访问出现在同 一模块内且在相邻的访问请求中，满足发生冲突的条件。

**32.** C

交叉编址多模块存储器有轮流启动和同时启动两种方式，本题中所有存储模块一次并行读/ 写的总位数正好等于系统总线中的数据线数，故可以判定采用的是同时启动方式。在同时启动方 式下， 一个存储周期可以对所有芯片的同一行都读取一个字节。double 型变量占64位(8B)。

其主存地址804001AH 的最低两位是10,说明它从编号为2的芯片开始存储(编号从0开始) , 共占3行，因此需要同时启动3轮才能完成对 double 型变量的读取。从本题也可发现，采用同时 启动方式时， 一次读行也许会有没用的数据读入。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 第i轮 |  |  |  |  |
| 第i+1轮 |  |  |  |  |
| 第i+2轮 |  |  |  |  |
| 第i+3轮 |  |  |  |  |
| 第i+4轮 |  |  |  |  |
| 体号 | 00 | 01 | 10 | 11 |

33.C

由题意，首先根据 DRAM 采用的是行列地址线复用技术，我们尽量选用行列差值不要太大的， 选项 B 、C的地址线只需6根(取行或列所需地址线的最大值),轻松排除A 和 D。其次，为了减小 刷新开销，而DRAM 一般是按行刷新的，所以应选行数值较少的。

**34.** C

8×8192×8192×8bit=512MB, 内存条的容量为512MB,A 正确。存储器总线宽度64=8×8bit,

100-2025年计算机组成原理考研复习指导

而每个芯片一次只能传输8bit, 需要8体多模块交叉编址采用同时启动方式才能实现， B 正确。 芯片容量为8192×8192×8bit, 按字节编址，地址线数应为log(8192×8192)=26,DRAM 采用地址 复用技术，地址信号分行、列两次传送，因此地址引脚数为26/2=13根， C 错误。芯片内行数是 8192,一行的大小是8192×8bit, 行缓冲长度就是一行的大小， D 正确。

二 、综合应用题

01.【解答】

1)因为刷新带宽 W₁= 分辨率×像素点颜色深度×刷新速率

=1024×768×3B×72/s

=169869KB/s

所以刷新总带宽 W₀=W₁ (WJ/W₁)

=169869KB/s×100/50=339738KB/s

=339.738MB/s (其中1K=1000)

2)要提高刷新存储器带宽，可采用以下技术：①采用高速DRAM芯片；②采用多体交叉存 储结构；③刷新存储器到显示控制器的内部总线宽度加倍；④采用双端口存储器将刷新 端口和更新端口分开。

02.【解答】

1)一个存取周期，四体并行交叉存储器可取32位×4=128位，其中32位为总线宽度，4为 交叉存储器内的存储体个数。

2)该说法不正确。因为在0.1μs 内整个存储器可向CPU 提供32位二进制信息，但每个存储 体必须经过400ns 才能向CPU 提供32位二进制信息。

03.【解答】

顺序存储器和交叉存储器连续读出m=4 个字的信息总量均是

q=64 位×4=256位

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

t₁=mT=4×200ns=800ns=8×10~'s

tz=T+(m-1)r=200ns+3×50ns=350ns=35×10~\*s

顺序存储器和交叉存储器的带宽分别是

W₁=q/t₁=256÷(8×10-⁷)=32×10⁷b/s

W₂=ql₁₂=256-(35×10)=73×10'b/s

04.【解答】

交叉存储器在统考真题中曾多次考查，希望能引起读者重视，本题是这一类题中较难的。

1)因为每个体的存取周期是200ns。四体交叉工作，每两个体间读出操作的延时为1/4个存 储周期，理想情况是每个存取周期平均可读出4个数据字，读出一个数据字的时间平均 为200ns/4=50ns 。数据字长为32位，数据传输速率为32位/50ns=640Mb/s=80MB/s。

2)若对多体结构的存储器选用高位地址交叉，通常起不到提高存储器读/写速度的作用，因 为它不符合程序运行的局部性原理， 一次连续读出彼此地址相差一个存储体容量的4 个 字的机会太少。因此，通常只有一个存储模块在不停地忙碌，其他存储模块是空闲的。

3)若把存储器的字长扩大为原来的4倍，实现的则是一个单体4字结构的存储器，每次读 可以同时读出4个字的内容，有利于提高存储器每个字的平均读/写速度，但其灵活性不 如多体单字结构的存储器，还会多用到几个缓冲寄存器。

4)多端口存储器是对同一个存储体使用多套读/写电路实现的，扩大存储容量的难度显然比 多体结构的存储器要大，而且不能对多端口存储器的同一个存储单元同时执行多个写入 操作，而多体结构的存储器则允许在同一个存储周期对几个存储体执行写入操作。

**05.【** 解答】

对于四体交叉访问的存储系统，每个存储模块的地址分布如下：

Bank0:0,4,8,12,16,…

Bank1:1,5,9,13,17,…,37,…,41, …

Bank2:2,6,10,14,18,…

Bank3:3,7,11,15,19,…,51,…,67

若给定的访存地址在相邻的4次访问中出现在同一个 Bank 内，则可能发生访存冲突。所以 17和9、37和17、13和37、8和4可能发生冲突。 易 错 点： 虽然41和13号单元也在同 一 个模 块内，并且访问间隔小于4,但是由于访问8号单元发生冲突而使其访问延迟3个间隔，从而使 41号单元的访问也延迟3个间隔，因此其访问不会和13号单元的访问发生冲突。

**3.3** **主存储器与** **CPU** **的连接**

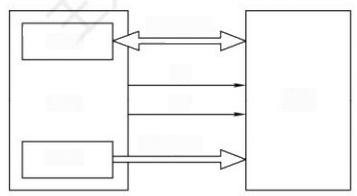
**3.3.1** **连接原理**

1)主存储器通过数据总线、地址总线和控制总线与 CPU 连接。

2)数据总线的位数与工作频率的乘积正比于数据传输速率。

3)地址总线的位数决定了可寻址的最大内存空间。

4)控制总线(读/写)指出总线周期的类型和本次输入/输出操作完成的时刻。 主存储器与 CPU 的连接如图3 . 10所示。

数据总线

MDR

读

写

主存

CPU

地址总线

MAR

图3.10 主存储器与CPU 的连接

单个芯片的容量是有限的，因此通过存储器芯片扩展技术，将多个芯片集成在一个内存条上， 然后由多个内存条及主板上的ROM 芯片组成计算机所需的主存空间，再通过总线与CPU 相连。

**3.3.2** **主存容量的扩展**

由于单个存储芯片的容量是有限的，它在字数或字长方面与实际存储器的要求都有差距，因 此需要在字和位两方面进行扩充才能满足实际存储器的容量要求。

**1.** **位扩展法**

位扩展是指对字长进行扩展(增加存储字长)。当 CPU 的系统数据线数多于存储芯片的数 据位数时，必须对存储芯片扩位，使其数据位数与CPU 的数据线数相等。

102-2025年计算机组成原理考研复习指导

位扩展的连 接 方 式： 各芯片的地址线、片选线和读/写控制线与系统总线相应并联；各芯片的 数据线单独引出，分别连接系统数据线。各芯片同时工作。

如 图 3 . 1 1 所 示 ， 用 8 片 8K×1 位 的RAM 芯 片 组 成 8K×8 位 的 存 储 器 。 8 片RAM 芯 片 的 地 址

线 A₁₂~A₀ 、CS 、WE 都分别连在 一 起，每片的数据线依次作为 CPU 数据线的 一 位。



8

Vo

I/O VO

VO vo

l/O

vo

中央

处理器

D₀

数据总线

D₇

地址总线 Ag

M

A₂

8K×1位 I/O

图3.11 位扩展连接示意图

**2.** **字** **扩** **展** **法**

字 扩 展是指对存储字的数量进行扩展，而存储字的位数满足系统要求。系统数据线位数等于 芯片数据线位数，系统地址线位数多于芯片地址线位数。

字扩展的连接方式：各芯片的地址线与系统地址线的低位对应相连；芯片的数据线和读/写控

制线与系统总线相应并联；由系统地址线的高位译码得到各芯片的片选信号。各芯片分时工作。

命 题 追 踪 ▶ **字扩展(或字位扩展)后存储芯片的地址范围(2010、2016)**

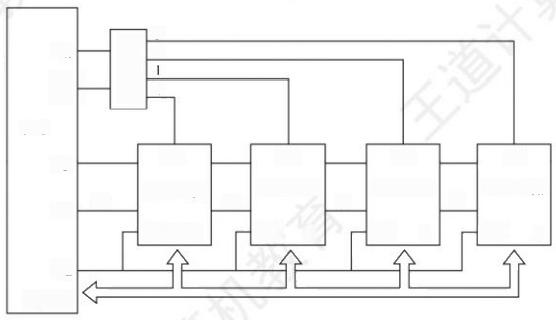
如图3 . 12所示，用4片16K×8 位 的RAM 芯 片 组 成 6 4K×8 位 的 存 储 器 。 4 片RAM 芯 片 的 数 据 线 D₀~D₇ 和 WE 都分别连在 一 起。将A₁sAj₄ 用 作 片 选 信 号 ，AjsA₁₄=00 时 ， 译 码 器 输 出 端 0 有效，选中最左边1号芯片； A₁sA₁4=01 时，译码器输出端1有效，选中2号芯片，以此类推(同 一 时刻只能有 一 个芯片被选中)。各芯片的地址分配如下：

第一片，最低地址：0000000000000000;最高地址：0011111111111111(16位)

第二片，最低地址：0100000000000000;最高地址：0111111111111111

第三片，最低地址：1000000000000000;最高地址：1011111111111111

第四片，最低地址：1100000000000000;最高地址：1111111111111111



A₁s

A-

CPU

Ao

16K×8位

A₃

WE 2

wE

D₀~D₇

Cs

16K×8位

WE 4

Cs

16K×8位

WE 1

Cs

16K×8位

2/4 译 码 器

WE 3

M

M

3

2

Cs

M

M

0

图3.12 字扩展连接示意图

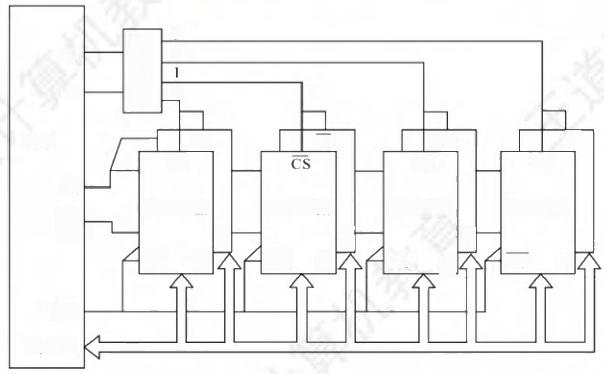
第 3 章 存 储 系 统 -103

**3.** **字位同时扩展法**

字位同时扩展是前两种扩展的组合，这种方式既增加存储字的数量，又增加存储字长。

字位同时扩展的连接方式： 将进行位扩展的芯片作为一组，各组的连接方式与位扩展的相 同；由系统地址线高位译码产生若干片选信号，分别接到各组芯片的片选信号。

如图3.13所示，用8片16K×4 位的RAM 芯片组成64K×8 位的存储器。每两片构成一组16K×8 位的存储器(位扩展),4组便构成64K×8 位的存储器(字扩展)。地址线A₁sA₁₄ 经译码器得到4 个片选信号， A₁ sA₄=00 时，输出端0有效，选中第一组的芯片(①和②);AjsA₁₄=01 时，输 出端1有效，选中第二组的芯片(③和④),以此类推。



2/4 3

译 2

码

器 0

CS CS cs Cs

cs cs cs

Ao|

16K×4位 16K×4位 16K×4位 16K×4位

② ④ 回 圆

wE ① WE ③ WE ⑤[ WE ⑦

WE

D₀~D₇

Ais|

A14|

An-

CPU

图3. 13 字位同时扩展及CPU 的连接图

**3.3.3** **存储芯片的地址分配和片选**

CPU 要实现对存储单元的访问，首先要选择存储芯片，即进行片选；然后在选定的芯片中选 择具体的存储单元，以进行数据的读/写，即进行字选。芯片内的字选通常是由CPU 送出的N 条 低位地址线完成 (N 由片内存储容量2\*决定)。片选信号的产生方法分为线选法和译码片选法。

**1.** **线选法**

线选法用除片内寻址外的高位地址线直接连接至各个存储芯片的片选端，当某位地址线信息 为“0”时，就选中与之对应的存储芯片。这些片选地址线每次寻址时只能有一位有效，不允许 同时有多位有效，这样才能保证每次只选中一个芯片(或芯片组)。

**表3.2** **线选法的地址分配**

假设4片2K×8 位存储芯片采用线选法构成8K×8 位存储器，各芯片

|  |  |
| --- | --- |
| 芯片 | Au～An |
| 0# | 1110 |
| 1# | 1101 |
| 2# | 1011 |
| 3# | 0111 |

的片选信号见表3.2,其中低位地址线 Aio～A₀ 作为字选线，用于片

内寻址。

优点：不需要地址译码器，线路简单。缺点：地址空间不连续，

选片的地址线必须分时为低电平(否则不能工作),不能充分利用系

统的存储器空间，造成地址资源的浪费。

**2.** **译码片选法**

译码片选法用除片内寻址外的高位地址线通过地址译码器产生片选信号。如用8片8K×8 位 的 存 储 芯 片 组 成 6 4K×8 位 存 储 器 ( 地 址 线 为 1 6 位 ， 数 据 线 为 8 位 ) , 需 要 8 个 片 选 信 号 ； 若 采 用

104-2025 年计算机组成原理考研复习指导

线选法，除去片内寻址的13位地址线，仅余高3位，不足以产生8个片选信号。因此，采用译 码片选法，即用一片74LS138 作为地址译码器，高3位用于片选，则AjsA₁₄A₁=000 时选中第 一 片 ，A₁sA₁₄A₁₃=001 时选中第二片，以此类推。

**3.3.4** **存** **储** **器** **与** **CPU 的** **连** **接**

**1.** **合理选择存储芯片**

▶ **根据要求合理选择存储芯片(2009、2021)**

要组成一个主存系统，选择存储芯片是第一步，主要指存储芯片的类型 (RAM 或 ROM) 和 数量的选择。通常选用ROM 存放系统程序、标准子程序和各类常数，RAM 则是为用户编程而设 置的。此外，在考虑芯片数量时，要尽量使连线简单、方便。

**2.** **地址线的连接**

命 题 追 踪▶ **地址范围与存储容量的对应关系(2016、2023)**

存储芯片的容量不同，其地址线数也不同，而 CPU 的地址线数往往比存储芯片的地址线数 要多。通常将CPU 地址线的低位与存储芯片的地址线相连，以选择芯片中的某一单元(字选), 这部分的译码是由芯片的片内逻辑完成的。而CPU 地址线的高位则在扩充存储芯片时使用，用来 选择存储芯片(片选),这部分译码由外接译码器逻辑完成。

例如，设CPU 地址线为16位，即Ajs～A₀,1K×4 位的存储芯片仅有10根地址线，此时可 将 CPU 的低位地址Ag～A₀ 与存储芯片的地址线Ag～A₀ 相连。

**3.** **数据线的连接**

CPU 的数据线数与存储芯片的数据线数不一定相等，在相等时可直接相连；在不等时必须对 存储芯片扩位，使其数据位数与 CPU 的数据线数相等。

**4.** **读/写命令线的连接**

CPU 读/写命令线一般可直接与存储芯片的读/写控制端相连，通常高电平为读，低电平为写。 有 些CPU 的读/写命令线是分开的(读为RD, 写 为WE, 均为低电平有效),此时 CPU 的读命令 线应与芯片的允许读控制端相连，而CPU 的写命令线则应与芯片的允许写控制端相连。

**5.** **片选线的连接**

片选线的连接是 CPU 与存储芯片连接的关键。存储器由许多存储芯片叠加而成，哪一片被 选中完全取决于该存储芯片的片选控制端CS 是否能接收到来自 CPU 的片选有效信号。

片选有效信号与 CPU 的访存控制信号MREQ (低电平有效)有关，因为只有当 CPU 要求访

存时，才要求选中存储芯片。若 CPU 访 问I/O, 则 MREQ 为高，表示不要求存储器工作。

**3.3.5** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 用存储容量为16K×1 位的存储器芯片来组成一个64K×8 位的存储器，则在字方向和位 方向分别扩展了( ) 倍 。

A.4,2 B.8,4 C.2,4 D.4,8

**02.** 80386DX 是32位系统，以4B 为编址单位，当在该系统中用8KB(8K×8 位 ) 的 存 储 芯

片构造32KB 的存储体时，应完成存储器的( ) 设 计 。

A. 位扩展 *B. 字扩展* C. 字位扩展 D. 字位均不扩展

**03.** 某计算机字长为16位，存储器容量为256KB,CPU 按字寻址，其寻址范围是()。 A.0~2¹⁹-1 B.0~220-1 C.0~2¹⁸-1 D.0～217-1

**04.** 4个16K×8 位的存储芯片，可设计为()容量的存储器。

A.32K×16 位 B.16K×16 位 C.32K×8 位 D.8K×16 位

05 . 16片2K×4 位的存储器可以设计为()存储容量的16位存储器。

A.16K B.32K C.8K D.2K

**06.** 设 CPU 地址总线有24根，数据总线有32根，用512K×8 位的RAM 芯片构成该机的主 存储器，则该机主存最多需要( )片这样的存储芯片。

A.256 B.512 C.64 D.128

07. 地址总线A₀ ( 高 位 ) ~Ajs (低位),用4K×4位的存储芯片组成16KB 存储器，则产生

片选信号的译码器的输入地址线应该是()。

A.A₂A₂ B.A₀A C.Ai₂A₁₃ D.A₁₄A₁:

**08** .若内存地址区间为4000H～43FFH, 每个存储单元可存储16位二进制数，该内存区域用 4片存储器芯片构成，构成该内存所用的存储器芯片的容量是()。

A.512×16bit B.256×8bit C.256×16bit D.1024×8bit

**09.** 内存按字节编址，地址从90000H到 CFFFFH, 若用存储容量为16K×8 位芯片构成该内

存，至少需要的芯片数是( )。

A.2 B.4 C.8 D.16

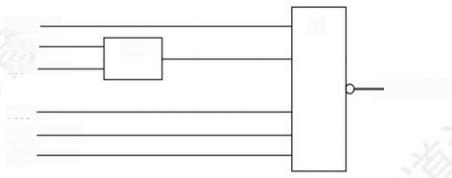
**10.** 若片选地址为111时，选定某一32K×16位的存储芯片工作，则该芯片在存储器中的首 地址和末地址分别为()。

A.00000H,01000H B.38000H,3FFFFH

C.3800H,3FFFH D.0000H,0100H

**11.** 如下图所示，若低位地址 (Ao~A) 接在内存芯片地址引脚上，高位地址 (Al₂~Aig)

进行片选译码(其中A₁₄ 和 Al6未参加译码),且片选信号低电平有效，则对图中所示的 译码电路，不属于此译码空间的地址是( )。



&

≥1

A17——

一译码输出

A15——

A₁3 ——

A12———

A19 ——

A18 ——

A.AB000H～ABFFFH B.BB000H~BBFFFH

C.EF000H~EFFFFH D.FE000H~FEFFFH

**12.**【2009统考真题】某计算机主存容量为64KB, 其 中ROM 区为4KB, 其余为RAM 区， 按字节编址。现要用2K×8 位的ROM 芯片和4K×4 位的RAM 芯片来设计该存储器，需 要上述规格的 ROM 芯片数和RAM 芯片数分别是( )。

A.1,15 B.2,15 C.1,30 D.2,30

**13.**【2010统考真题】假定用若干2K×4位的芯片组成一个8K×8 位的存储器，则地址 OBIFH 所在芯片的最小地址是()。

A.0000H B.0600H C.0700H D.0800H

106-2025年计算机组成原理考研复习指导

**14.【**2011统考真题】某计算机存储器按字节编址，主存地址空间大小为64MB, 现 用 4M×8

位的RAM 芯片组成32MB 的主存储器，则存储器地址寄存器 MAR 的位数至少是( )。

A.22 位 B.23 位 C.25 位 D.26 位

**15.【**2016统考真题】某存储器容量为64KB, 按字节编址，地址4000H～5FFFH 为 ROM 区 ， 其余为 RAM 区。若采用8K×4 位的 SRAM 芯片进行设计，则需要该芯片的数量是( )。 A.7 B.8 C.14 D.16

**16.【2**021 统考真题】某计算机的存储器总线中有24位地址线和32位数据线，按字编址， 字长为32位。若000000H～3 F FFFFH 为 RAM 区，则需要512K×8 位 的RAM 芯片数 为 ( ) 。

A.8 B.16 C.32 D.64

**17.【**2023统考真题】某计算机的CPU 有30根地址线，按字节编址，CPU 和主存连接时， 要求主存芯片占满所有可能的存储地址空间，并且 RAM 区和 ROM 区所分配的空间大 小比是3:1。若 RAM 在低地址区， ROM 在高地址区，则 ROM 的地址范围是()。

A.00000000H～OFFFFFFFH B.10000000H~2FFF FFFFH

C.30000000H～3FFF FFFFH D.40000000H～4FFF FFFFH

**二、综合应用题**

**01.** 主存储器的地址寄存器和数据寄存器各自的作用是什么?设一个1MB 容量的存储器， 机器字长和存储字长均为32位，问：

1)按字节编址，地址寄存器和数据寄存器各几位?编址范围为多大? 2)按字编址，地址寄存器和数据寄存器各几位?编址范围为多大?

**02.** 用一个512K×8 位的 Flash 存储芯片组成一个4M×32 位的半导体只读存储器，存储器按 字编址，试回答以下问题：

1)该存储器的数据线数和地址线数分别为多少?

2)共需要几片这样的存储芯片?

3)说明每根地址线的作用。

**03.** 有一组16K×16位的存储器，由1K×4 位 的DRAM 芯片构成(芯片是32×32结构)。问： 1)共需要多少RAM 芯片?

2)采用异步刷新方式，如单元刷新间隔不超过2ms, 则刷新信号周期是多少?

**3.3.6** **答** **案** **与** **解** **析**

**一、单项选择题**

**01.D**

字方向扩展了64K/16K=4 倍，位方向扩展了8bit/1bit=8 倍。

02.A

因为以4B 为编址单位，要扩展到32KB, 即扩展到8K×32bit, 所以只用进行位扩展。

03.D

256KB=2l⁸B, 按字寻址，且字长为16bit=2B, 可寻址的单元数=2¹⁸ B/2B=2'7, 其寻址范 围是0～2l⁷-1。

04.A

4个16K×8 位的存储芯片构成的存储器容量=4×16K×8 位 = 5 1 2K 位或64KB, 只有选项A 的容量为64KB 。注意，若有某项为128K×4 位，则此选项不能选，因为芯片为8位，不可能将字

第3章 存储系统- 107

长“扩展”成4位。

05.C

设存储容量为M, 则有(M×16)-(2K×4)=16, 因此M=8K。

06.D

地址线为24根，寻址范围是22\*;数据线为32根，字长为32位。主存的总容量=224×32位， 因此所需存储芯片数=(2²⁴ ×32)=(512K×8)=128。

07.A

由于Ajs 为地址线的低位，接入各芯片地址端的是地址线的低12位，即A₄~Als, 共有8个 芯片(16KB/4K=4B, 并且位扩展时每组两片分为4组)组成16KB的存储器，因此由高两位地 址线A₂A₃ 作为译码器的输入。

08.C

43FF-4000+1=400H, 即内存区域为1K 个单元，总容量为1K×16位。现由4片存储芯片 构成，则构成该内存的芯片容量为1K×16 位/4=256×16位。

09.D

CFFFF-90000+1=40000H, 即内存区域有256K 个单元。若用存储容量为16K×8位的芯片， 则需要的芯片数=(256K×8)÷(16K×8)=16片。

10.B

32K×16的存储芯片有地址线15根(片内地址),片选地址为3位，因此地址总位数为18位， 现高3位为111,则首地址为111000000000000000 =38000H, 末地址为111111111111111111 = 3FFFFH。

11.D

这是一个部分译码的片选信号，高8位地址中有2位 (A₁4 和A₁6) 未参与译码，根据译码器 电路，译码输出的逻辑表达式应为

CS=A,(Aμ+A₉)A₃A₁An

因此不属于此译码空间的是这几位不合该逻辑表达式的， A 选项为 AB, 即10101011,去掉14 位和16位为101111; B 选项为101111;C 选项为111111; D 选项为111110。由逻辑表达式可 知 A17与A18 至少有一个为1,AgA₁sA₁₃A₁₂ 应全为1,仅D 无法满足。

12.D

首先确定ROM的个数，ROM区为4KB, 选用2K×8 位的ROM芯片，需要(4K×8)÷(2K×8)=2 片，采用字扩展方式； RAM 区为60KB, 选用4K×4位的 RAM 芯片，需要(60K×8)=(4K×4)=30 片，采用字和位同时扩展的方式。

13.D

用2K×4 位的芯片组成一个8K×8 位存储器，共需8片2K×4位的芯片，分为4组，每组由2 片2K×4位的芯片并联组成2K×8位的芯片，各组芯片的地址分配如下：

第一组(两个芯片并联):0000H～07FFH。

第二组(两个芯片并联):0800H～0FFFH。

第三组(两个芯片并联):1000H～17FFH。

第四组(两个芯片并联):1800H～1FFFH。

地址OBIFH 所在的芯片属于第二组，故其所在芯片的最小地址为0800H。

14.D

主存按字节编址，地址空间大小为64MB,MAR 的寻址范围为64M=226, 因此是26位。实

108-2025年计算机组成原理考研复习指导

际的主存容量32MB 不能代表MAR 的位数，考虑到存储器扩展的需要， MAR 应保证能访问到整 个主存地址空间，反过来， MAR的位数决定了主存地址空间的大小。

15.C

5FFF-4000+1=2000H, 即 ROM区容量为2¹³B=8KB(2000H=2×16³=2¹³),RAM 区容量为

56KB(64KB-8KB=56KB) 。 需要8K×4位的SRAM芯片的数量为14(56KB/8K×4位=14)。

16.C

000000～3FFFFF, 共有3FFFFFH-000000H+1H=400000H=2² 个地址，按字编址，字长为32 位(4B), 因此RAM 区大小为2²²×4B=2²×32bit 。每个RAM芯片的容量为512K×8bit=2¹⁹×8bit, 所 以需要RAM芯片的数量为(2²×32bit)=(2¹⁹×8bit)=32。

17.C

地址空间为20,地址范围00000000H～3FFF FFFFH 。RAM:ROM=3:1, 则 ROM可分配的 地址空间为228,从3FFF FFFFH往前数228个地址，即ROM 的地址范围是30000000H～3FFF

FFFFH。

二、综合应用题

01. 【解答】

在主存储器中，地址寄存器 MAR用来存放当前 CPU 访问的内存单元地址，或存放 CPU写 入内存的内存单元地址。数据寄存器MDR 用来存放由内存中读出的信息或写入内存的信息。

1)按字节编址，1MB=220×8位，地址寄存器为20位，数据寄存器位数由存储字长决定， 为32位，编址范围为00000H～FFFFFH(FFFFFH-00000H+1=100000H=220)。

2)按字编址，1MB=2l⁸×32 位，地址寄存器为18位，数据寄存器为32位，编址范围为 00000H～3FFFFH(3FFFFH-00000H+1=40000H=2l⁸)。

02. 【解答】

1)因为所需的组成存储器的最终容量为4M×32 位，所以需要32根数据线。而存储器又是 按字编址的，所以此时不需要将存储器的容量先转换成16M×8 位，直接是4M×32 位中 的4M, 所以只需要22根地址线(2²²=4M)。

2)采用512K×8 位的 Flash 存储芯片组成4M×32 位的存储器时，需要同时进行位扩展和字 扩展。位扩展：4片512K×8位的 Flash 存储芯片位扩展可组成512K×32位的Flash 存储 芯片。字扩展：8片512K×32 位的Flash 存储芯片字扩展可组成4M×32 位的存储器。综 上可知， 一共需要4×8=32片512K×8位的存储芯片。

3)在 CPU 的22根地址线中 (Ao～A₂₁), 地址线的作用分配如下：首先，此时不需要指定 A₀ 、A₁来标识每组中的4片存储器，因为此时是按字寻址的，所以4片每次都是一起取

的，而不是按字节编址时需要取4片中的某一片。

A₀~A₁g: 每片都是512K, 所以需要19位(21⁹=512K) 来表示。

Alg 、Az₀ 、A₂₁: 因为在扩展中4片一组， 一共有8组(=2³),所以需要用3位地址线来

决定取哪一组(通过3/8译码器形成片选信号)。

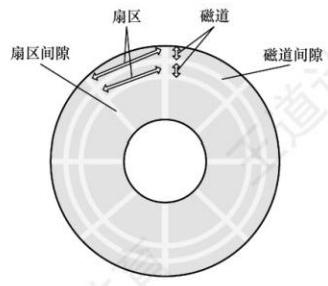
03. 【解答】

1)存储器总容量为16K×16位， RAM 芯片为1K×4位，因此所需芯片总数为(16K×16位/

(1K×4 位)=64片。

2)采用异步刷新方式，在2ms 时间内分散地把芯片64行刷新一遍，因此刷新信号的时间间

隔 为 2ms/64=31.25μs,即可取刷新信号周期为31μs。





**注** **意**

刷新周期也可取30μs, 只要小于31.25μs 即可，但通常取刷新间隔的整数部分。

**3.4** **外部存储器**

**3.4.1** **磁盘存储器**

磁盘存储器是以磁盘为存储介质的存储器，其主要优点：①存储容量大，位价格低；②记录 介质可重复使用；③记录信息可长期保存而不丢失，甚至可脱机存档；④非破坏性读出，读出时 不需要再生。缺点：存取速度慢，机械结构复杂，对工作环境要求较高。

**1.** **磁盘存储器**

命 题 追 踪 ▶》**磁盘存储器的相关概念(2019)**

(1)磁盘设备的组成

① 磁盘存储器的组成。磁盘存储器由磁盘驱动器、磁盘控制器和盘片组成。

● 磁盘驱动器。驱动磁盘转动并在盘面上通过磁头进行读/写操作的装置，如图3.14所示。

● 磁盘控制器。磁盘驱动器与主机的接口，负责接收并解释CPU 发来的命令，向磁盘驱

动器发出各种控制信号，并负责检测磁盘驱动器的状态。



(a)磁盘盘片 (b)磁盘的组成

图3.14 磁盘驱动器示意图

② 存储区域。一个磁盘含有若干记录面，每个记录面划分为若干圆形的磁道，而每条磁道

又划分为若干扇区，扇区(也称块)是磁盘读/写的最小单位，即磁盘按块存取。

● 磁头数 (Heads): 即记录面数，表示磁盘共有多少个磁头，磁头用于读取/写入盘片上

记录面的信息， 一个记录面对应一个磁头。

● 柱面数 (Cylinders): 表示磁盘每面盘片上有多少条磁道。在一个盘组中，不同记录面 的相同编号(位置)的诸磁道构成 一 个圆柱面。

● 扇区数 ( Sectors): 表示每条磁道上有多少个扇区。

相邻磁道及相邻扇区间通过一定的间隙分隔开，以避免精度错误。由于扇区按固定圆心 角度划分，因此位密度从最外道向里道增加，磁盘的存储能力受限于最内道的最大记录 密度。

110-2025 年计算机组成原理考研复习指导

③ 磁盘高速缓存 (Disk Cache)。在内存中开辟一部分区域，用于缓冲将被送到磁盘上的数 据。优点：写磁盘时是按“簇”进行的，可以避免频繁地用小块数据写盘；有些中间结

果数据在写回磁盘之前可被快速地再次使用。

(2)磁记录原理

原理：磁头和磁性记录介质相对运动时，通过电磁转换完成读/写操作。

编码方法：按某种方案(规律),把一连串的二进制信息变换成存储介质磁层中一个磁化翻 转状态的序列，并使读/写控制电路容易、可靠地实现转换。

磁记录方式：通常采用调频制 (FM) 和改进型调频制 (MFM) 的记录方式。

(3)磁盘的性能指标

① 记录密度。记录密度是指盘片单位面积上记录的二进制信息量，通常以道密度、位密度 和面密度表示。道密度是沿磁盘半径方向单位长度上的磁道数，位密度是磁道单位长度 上能记录的二进制代码位数，面密度是位密度和道密度的乘积。

② 磁盘的容量。磁盘容量有非格式化容量和格式化容量之分。非格式化容量是指磁记录表面 可利用的磁化单元总数，非格式化容量=记录面数×柱面数×每条磁道的磁化单元数。格 式化容量是指按照某种特定的记录格式所能存储信息的总量。格式化容量=记录面数×柱 面数×每道扇区数x每个扇区的容量。格式化后的容量比非格式化容量要小。

**命题追踪** ▶ **磁盘存取时间的计算(2013、2015、2022)**

③ 存取时间。存取时间由寻道时间(磁头移动到目的磁道的时间)、旋转延迟时间(磁头定 位到要读/写扇区的时间)和传输时间(传输数据所花费的时间)三部分构成。因为寻道 和找扇区的距离远近不一，所以寻道时间和旋转延迟时间通常取平均值(平均寻道时间 取从最外道移动到最内道时间的一半，平均旋转延迟时间取旋转半周的时间)。

④ 数据传输速率。磁盘存储器在单位时间内向主机传送数据的字节数，称为数据传输速率。 假设磁盘转数为r 转/秒，每条磁道容量为N 字节，则数据传输速率为

*D₁ =rN*

(4)磁盘地址

**命题追踪** ▶ **磁盘地址结构的计算(2022)**

主机向磁盘控制器发送寻址信息，磁盘的地址一般如下图所示。

|  |  |  |
| --- | --- | --- |
| 柱面(磁道)号 | 盘面(磁头)号 | 扇区号 |

若磁盘有16个盘面，每个盘面有256个磁道，每个磁道划分为16个扇区，则每个扇区地址 要16位二进制代码，其格式如下图所示。

|  |  |  |
| --- | --- | --- |
| 柱面(磁道)号(8位) | 盘面(磁头)号(4位) | 扇区号(4位) |

(5)磁盘的工作过程

磁盘的主要操作是寻址、读盘、写盘。每个操作都对应一个控制字，磁盘工作时，第一步是 取控制字，第二步是执行控制字。磁盘属于机械式部件，其读/写操作是串行的，不可能在同一时 刻既读又写，也不可能在同一时刻读两组数据或写两组数据。

**2.** **磁盘阵列**

RAID (独立冗余磁盘阵列)是指将多个独立的物理磁盘组成一个独立的逻辑盘，数据在多 个物理盘上分割交叉存储、并行访问，具有更好的存储性能、可靠性和安全性。

命 题 追 踪 ▶ **提高** **RAID** **可靠性的措施(2013)**

RAID 的分级如下所示。在 RAID1～RAID5 几种方案中，无论何时有磁盘损坏，都可随时拔 出受损的磁盘再插入好的磁盘，而数据不会损坏，提升了系统的可靠性。

●RAID0: 无冗余和无校验的磁盘阵列。

·RAID1: 镜像磁盘阵列。

● RAID2: 采用纠错的海明码的磁盘阵列。

● RAID3: 位交叉奇偶校验的磁盘阵列。

●RAID4: 块交叉奇偶校验的磁盘阵列。

●RAID5: 无独立校验的奇偶校验磁盘阵列。

RAID0 把连续多个数据块交替地存放在不同物理磁盘的扇区中，几个磁盘交叉并行读/写， 即条带化技术，这样不仅扩大了存储容量，还提高了磁盘存取速度，但RAIDO 没有容错能力。

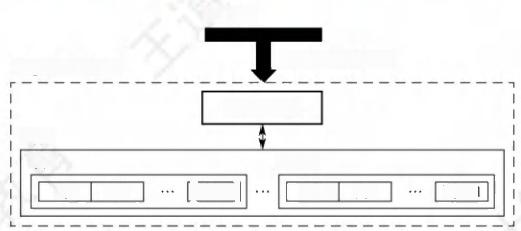
为了提高可靠性， RAID1 使两个磁盘同时进行读/写，互为备份，若一个磁盘出现故障，可 从另一磁盘中读出数据。两个磁盘当一个磁盘使用，意味着容量减少一半。

总 之 ，RAID 通过同时使用多个磁盘，提高了传输速率；通过在多个磁盘上并行存取来大幅 提高存储系统的数据吞吐量；通过镜像功能，提高安全可靠性；通过数据校验，提供容错能力。

**3.4.2** **固** **态** **硬** **盘**

**1.** **固态硬盘的特性**

固态硬盘 (SSD) 是一种基于闪存技术的存储器。它与U 盘并无本质差别，只是容量更大， 存取性能更好。 一个 SSD 由一个或多个闪存芯片和闪存翻译层组成，如图3.15所示。闪存芯片 替代传统旋转磁盘中的机械驱动器，而闪存翻译层将来自 CPU 的逻辑块读/写请求翻译成对底层 物理设备的读/写控制信号，因此，这个闪存翻译层相当于代替了磁盘控制器的角色。



I/O 总线

读/写逻辑磁盘块

闪存翻译层

块B-1

页 0 页1

固态硬盘 (SSD)

闪存 块0

页 P-1

页P-1

页I

页0

图3.15 固态硬盘 (SSD)

在图3.15中， 一个闪存由B 块组成，每块由P 页组成。通常，页的大小是512B～4KB, 每 块由32～128页组成，块的大小为16KB～512KB 。数据是以页为单位读/写的。只有在一页所属 的块整个被擦除后，才能写这一页。不过， 一旦一个块被擦除，块中的每个页就都可以直接再写 一次。某个块进行了若干次重复写之后，就会磨损坏，不能再使用。

随机写很慢，有两个原因。首先，擦除块较慢，通常比访问页高一个数量级。其次，若写操 作试图修改一个包含已有数据的页P, 则这个块中所有含有用数据的页都必须被复制到一个新(擦 除过的)块中，然后才能进行对页P, 的写操作。

比起传统磁盘， SSD 有很多优 点，它由半导体存储器构成，没有移动的部件，因而随机访问 时间比机械磁盘要快很多，也没有任何机械噪声和振动，能耗更低，抗震性好，安全性高等。

112-2025年计算机组成原理考研复习指导

**2.** **磨损均衡** (Wear Leveling)

固态硬盘也有缺点，闪存的擦写寿命是有限的， 一般是几百次到几千次。若直接用普通闪存 组装 SSD, 则实际的寿命表现可能非常令人失望——读/写数据时会集中在 SSD 的一部分闪存， 这部分闪存的寿命会损耗得特别快。一旦这部分闪存损坏，整块SSD 也就损坏了。这种磨损不均 衡的情况，可能会导致一块256GB的 SSD, 只因数兆字节空间的闪存损坏而整块损坏。

为了弥补 SSD 的寿命缺陷，引入了磨损均衡。SSD 磨损均衡技术大致分为两种：

1)动态磨损均衡。写入数据时，自动选择较新的闪存块。老的闪存块先歇一歇。

2)静态磨损均衡。这种技术更为先进，就算没有数据写入， SSD 也会监测并自动进行数据 分配，让老的闪存块承担无须写数据的存储任务，同时让较新的闪存块腾出空间，平常

的读/写操作在较新的闪存块中进行。如此一来，各个闪存块的寿命损耗就都差不多。

有了这种算法加持， SSD 的寿命就比较可观了。例如，对于一个256GB 的 SSD, 若闪存的 擦写寿命是500次，则需要写入125TB 数据，才寿终正寝。就算每天写入10GB数据，也要三十 多年才能将闪存磨损坏，更何况很少有人每天往 SSD 中写入10GB数据。

**3.4.3** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 下列关于磁盘的说法中，错误的是( )。

A. 本质上，U 盘(闪存)是一种只读存储器

B.RAID 技术可以提高磁盘的磁记录密度和磁盘利用率

C. 未格式化的硬盘容量要大于格式化后的实际容量

D. 计算磁盘的存取时间时，“寻道时间”和“旋转等待时间”常取其平均值 **02.** 下列关于磁盘驱动器的叙述中，错误的是()。

A. 送到磁盘驱动器的地址由磁头号、盘面号和扇区号组成

B. 能控制磁头移动到指定磁道，并发回“寻道结束”信号

C. 能控制磁盘片转过指定的扇区，并发回“扇区符合”信号

D. 能控制对指定盘面的指定扇区进行数据的读或写操作

**03.** 下列有关磁盘存储器读/写操作的叙述中，错误的是( )。

A. 最小读/写单位可以是一个扇区

B. 采用直接存储器存取 DMA方式进行输入/输出

C. 按批处理方式进行一个数据块的读/写

D. 磁盘存储器可与 CPU 交换盘面上的存储信息

**04.** 若磁盘的转速提高一倍，则()。

A. 平均寻道时间减少一半 B. 存取速度也提高一倍

C. 平均旋转延迟时间减少一半 D. 不影响磁盘传输速率

**05.** 下列关于固态硬盘 (SSD) 的叙述中，不正确的是()。

A. 固态硬盘的读/写是以页为单位的

B. 固态硬盘的擦除是以页为单位的

C. 固态硬盘的写入速度比读取速度慢很多

D. 固态硬盘的写入次数有限，引入磨损均衡可以延长使用寿命

**06.** 下列关于固态硬盘(SSD) 的说法中，错误的是( )。

第3章 存储系统一 113

A. 基于闪存的存储技术 B. 随机读/写性能明显高于磁盘

C. 随机写比较慢 D. 读/写速度快，常用作主存

**07**. 一个磁盘的转速为7200转/分，每个磁道有160个扇区，每个扇区有512字节，则在理

想情况下，磁盘每秒传输的数据量是()。

A.7200×160KB/s B. 7200KB/s C.9600KB/s D.19200KB/s

**08.** 某磁盘盘面共有200个磁道，盘面总存储容量为60MB, 磁盘旋转一周的时间为25ms, 每个磁道有8 个扇区，各扇区之间有一间隙，磁头通过每个间隙需1.25ms。则磁盘接口 所需的最大传输速率是()。

A.10MB/s B.60MB/s C.83.3MB/s D.20MB/s

**09.** 【2013 统考真题】某磁盘的转速为10000转/分，平均寻道时间是6ms, 磁盘传输速率是

20MB/s, 磁盘控制器延迟为0.2ms, 读取一个4KB 的扇区所需的平均时间约为( )。

A.9ms B.9.4ms C.12ms D.12.4ms **10.**【2013统考真题】下列选项中，用于提高RAID 可靠性的措施有()。

1. 磁盘镜像 IⅡ. 条带化 III. 奇偶校验 IV. 增加 Cache 机制

A. 仅 I 、Ⅱ B. 仅 I 、IⅢ C. 仅 I 、Ⅲ 和 IV D. 仅Ⅱ、Ⅲ和IV

**11** .【2015统考真题】若磁盘转速为7200转/分，平均寻道时间为8ms, 每个磁道包含1000 个扇区，则访问一个扇区的平均存取时间大约是()。

A.8.1ms B. 12.2ms C.16.3ms D.20.5ms

**12.**【2019统考真题】下列关于磁盘存储器的叙述中，错误的是()。

A. 磁盘的格式化容量比非格式化容量小

B. 扇区中包含数据、地址和校验等信息

C. 磁盘存储器的最小读/写单位为1字节

D. 磁盘存储器由磁盘控制器、磁盘驱动器和盘片组成

**二、** **综合应用题**

**01.** 某个硬磁盘共有4个记录面，存储区域内半径为10cm, 外半径为15.5cm, 道密度为60 道/cm, 外层位密度为600bit/cm, 转速为6000转/分。

1)硬磁盘的磁道总数是多少?

2)硬磁盘的容量是多少?

3)将长度超过一个磁道容量的文件记录在同一个柱面上是否合理?

4)采用定长数据块记录格式，直接寻址的最小单位是什么?寻址命令中磁盘地址如何

表示?

5)假定每个扇区的容量为512B, 每个磁道有12 个扇区，寻道的平均等待时间为 10.5ms, 试计算磁盘平均存取一个扇区的时间。

**3.4.4** **答案与解析**

**一、单项选择题**

**01.B**

闪存是在 E²PROM 的基础上发展起来的，本质上是只读存储器。RAID 将多个物理盘组成像 单个逻辑盘，不会影响磁记录密度，也不可能提高磁盘利用率。在磁盘的格式化过程中，要对磁

114-2025 年计算机组成原理考研复习指导

盘划分扇区，每个扇区要写入一些控制信息，扇区尾部还要留有一定的空隙，这些均需占用一些 存储空间，因此导致格式化后的实际容量比非格式化的容量要小。

**02.A**

因为每个盘面对应一个磁头，所以盘面号和磁头号是同一个概念，显然A 的说法是错误的， 磁盘地址应该由磁道号(柱面号)、磁头号(盘面号)和扇区号组成。

**03.D**

磁盘存储器以成批(组)方式进行数据读/写，CPU 中没有那么多通用寄存器用于存放交换 的数据，且磁盘与通用寄存器的速度相差过大，因此磁盘存储器通常直接和主存交换信息。

**04.C**

磁盘存取的步骤为：启动磁头、寻找磁道(寻道时间)、查找扇区(旋转延迟时间)、传输数 据，转速提高对寻道时间无影响；存取速度取决于所有步骤的时间，虽然会提高，但不会提高一 倍；平均旋转延迟时间为旋转半圈的时间，因此会减少一半；转速提高则传输速率也提高。

**05.B**

固态硬盘的擦除以块为单位，读/写以页为单位， B 错误。固态硬盘的写入速度比读取速度要 慢很多，因为在写入时需要擦除，且写入次数有限，否则该块就会因为磨损而无法再次写入。

**06.D**

固态硬盘基于闪存技术，没有机械部件，随机读/写不需要机械操作，因此速度明显高于磁盘， A 和 B 正确。选项C 已在考点讲解中解释过。SSD 常用作外存而非主存， D 错误。

07.C

磁盘的转速为7200转/分=120转/秒，转一圈经过160个扇区，每个扇区为512B, 所以磁 盘每秒传输的数据量为120×160×512/1024=9600KB。

**08.D**

每个磁道的容量=60MB/200=0.3MB, 读一个磁道数据的时间等于磁盘旋转一周的时间减 去通过扇区间隙的总时间(每个磁道有8个间隙),即25ms-1 .25ms×8=15ms, 数据传输速率= 0.3MB/l5ms =20MB/s。

**09.B**

磁盘转速是10000转/分，转一圈的时间为6ms, 因此平均查询扇区的时间为3ms, 平均寻道 时间为6ms, 读 取 4KB 扇区信息的时间为4KB÷20MB/s=0.2ms, 磁盘控制器延迟为0.2ms, 总时 间为3+6+0 . 2+0 . 2=9 .4ms。

**10.B**

RAID0 方案是无冗余和无校验的磁盘阵列，而RAID1～RAID5 方案均是加入了冗余(镜像) 或校验的磁盘阵列。因此，提高RAID 可靠性的措施主要是对磁盘进行镜像和奇偶校验，其余选 项不符合条件。条带化是一种将数据分片，分别存储至不同的磁盘，提高读/写速度的技术。条带 化的优点是读/写速度快，缺点是没有冗余，若其中一块磁盘损坏，则数据就会丢失。因此，条带 化通常和其他技术如磁盘镜像或奇偶校验结合使用，形成不同的RAID 级别。

**11.B**

存取时间=寻道时间+延迟时间+传输时间。存取一个扇区的平均延迟时间为旋转半周 的时间，即(60/7200/2=4.17ms, 传输时间为(60/7200)/1000=0.01ms, 因此访问一个扇区的平均 存取时间为4.17+0.01+8=12.18ms, 保留一位小数则为12.2ms。

**12.C**

磁盘存储器的最小读/写单位为一个扇区，即磁盘按块存取。磁盘存储数据之前需要进行格式

第3章 存储系统-115

化，将磁盘分成扇区，并写入信息，因此磁盘的格式化容量比非格式化容量小。磁盘扇区中包含 数据、地址和校验等信息。磁盘存储器由磁盘控制器、磁盘驱动器和盘片组成。

二、综合应用题

01. 【解答】

1)有效存储区域=15.5-10=5.5cm, 道密度=60道/cm, 因此每个面为60×5.5=330道，

即有330个柱面，因此磁道总数=4×330=1320个磁道。

2)外层磁道的长度为2πR=2×3.14×15.5=97.34cm。

每道信息量=600bit/cm×97.34cm=58404bit=7300B。

利用1)的结果，可得磁盘总容量=7300B×1320=9636000B (非格式化容量)。

3)若长度超过一个磁道容量的文件，将它记录在同一个柱面上是比较合理的，因为不需要

重新寻找磁道，这样数据读/写速度快。

4)采用定长数据块格式，直接寻址的最小单位是一个扇区，每个扇区记录固定字节数目的

信息，在定长记录的数据块中，活动头磁盘组的编址方式可用如下格式：

|  |  |  |
| --- | --- | --- |
| 12～6(7位) | 5～4(2位) | 3～0(4位) |
| 柱面号 | 盘面号 | 扇区号 |

此地址格式表示共有4个记录面，每面有128个磁道，每道最多可有16个扇区。

5)读一个扇区中数据所用的时间=找磁道的时间+找扇区的时间+磁头扫过一个扇区的时间。

找磁道的时间是指磁头从当前所处磁道运动到目标磁道的时间， 一般选用磁头在磁盘径 向方向上移动1/2个半径长度所用的时间为平均值来估算，题中给出的是10.5ms。

找扇区的时间是指磁头从当前所处扇区运动到目标扇区的时间， 一般选用磁盘旋转半周 所用的时间作为平均值来估算，题中给出磁盘转速为6000转/分，即100转/秒，所以磁 盘转一周用时10ms, 转半周用时5ms。

题中给出每个磁道有12个扇区，磁头扫过一个扇区用时为10/12 =0.83ms, 因此磁盘平 均存取时间为10.5+5+0.83=16.33ms。

3.5 高速缓冲存储器

由于程序的转移概率不会很低，数据分布的离散性较大，因此单纯依靠并行主存系统提高主 存系统的效率是有限的。高速缓存 Cache 拥有比主存更快的速度，因此在 CPU 和主存之间设置 Cache 可以显著提高存储系统的效率。Cache 由 SRAM组成，通常直接集成在 CPU中。

3.5.1 程序访问的局部性原理

程序访问的局部性原理包括时间局部性和空间局部性。

命题追踪▶分析给定代码的时空局部性(2017、2023)

时间局部性是指最近的未来要用到的信息，很可能是现在正在使用的信息，因为程序中存在 循环和需要多次重复执行的子程序段，以及对数组的存储和访问操作。空间局部性是指最近的未 来要用到的信息，很可能与现在正在使用的信息在存储空间上是邻近的，因为指令通常是顺序存

|  |
| --- |
| 1 int sumarraycols(int a[M][N]) 2  3 int i,j,sum =0;  4 for(j=0;j<N;j++)  5 for(i=0;i<M;i++)  6 sum +=a[i][j];  7 return sum;  8 ) |

放、顺序执行的，数据一般也是以向量、数组等形式簇聚地存储的。

高速缓冲技术就是利用局部性原理，把程序中正在使用的部分数据存放在一个高速的、容量 较小的Cache 中，使CPU 的访存操作大多数针对Cache 进行，从而提高程序的执行速度。

**【例3.2】**假设数组元素按行优先方式存储，对于下面的两个程序：

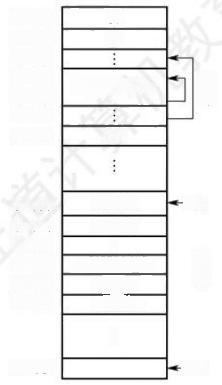
程序 A: 程序 B:

|  |
| --- |
| 1 int sumarrayrows(int a[M][N])  2  3 int i,j,sum =0;  4 for(i=0;i<M;i++)  5 for(j=0;j<N;j++)  6 sum +=a[i][j];  7 return sum;  8 ) |

1)对于数组a 的访问，哪个空间局部性更好?哪个时间局部性更好?

2)对于指令访问来说，for 循环体的空间局部性和时间局部性如何?

解：假定M 、N 都为2048,按字节编址，每个数组元素占4字节，则指令和数据在主存中 的存放情况如图3.16所示。



10

I

0×104

0×108

0×10C

0×110

0×114

a[0][0] a[0]

:

a[0][2047] al1]I0]

a[1]J]

:

0×7A4

0×400

0×404

0×408

0×40C 0×410

0×414

O×0FC

0×100

for内/外循环

数 据

指令

—V

—A

136

图3 . 16 指令和数据在主存的存放

**命题追踪** ▶ **数组按行或列访问命中率的分析(2010);数组循环访问的命中率分析(2016、2020)**

1)对于数组a, 程序A 和程序B 的空间局部性相差较大。

程 序A 对数组a 的访问顺序为a[0][O],a[0][1],…,a[0][2047];a[1][0],a[1][1],…,a[1][2047]; …。 由此可见，访问顺序与存放顺序是一致的，因此空间局部性好。

程序B 对数组a 的访问顺序为a[0][0],a[1][0],…,a[2047][0];a[0][1],a[1][1],…,a[2047][1]; …。

由此可见，访问顺序与存放顺序不一致，每次访问都要跳过2048个数组元素，即8192字节，若 主存与 Cache 的交换单位小于8KB, 则每访问一个数组元素都需要将一个主存块装入Cache, 因 而没有空间局部性。

两个程序中，数组 a 的时间局部性都差，因为每个数组元素都只被访问一次。

命 题 追 隙 ▶》**程序中指令** **Cache** **的命中率分析(2014)**

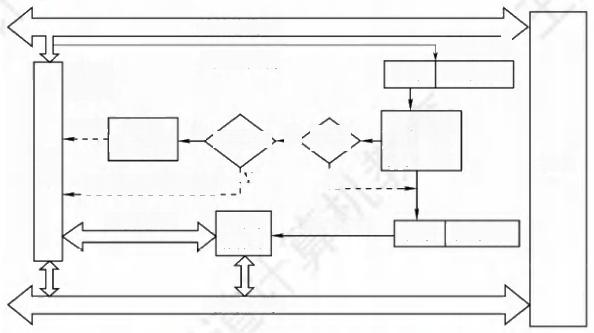
2 ) 对 于 for 循环体，程序A 和 程 序B 中的访问局部性是 一样的。因为循环体内指令按序连续存

放，所以空间局部性好；内循环体被连续重复执行2048×2048次，因此时间局部性也好。

由上述分析可知，虽然程序 A 和程序 B 的功能相同，但因内、外两重循环的顺序不同而导 致两者对数组 a 访问的空间局部性相差较大，从而带来执行时间的巨大差异。

**3.5.2** **Cache** **的基本工作原理**

为 便 于 Cache 与主存交换信息， Cache 和主存都被划分为大小相等的块， Cache块也 称 Cache 行，每块由若干字节组成，块的长度称为块 长 (也称行长)。因为 Cache 的容量远小于主存的容 量，所以 Cache 中的块数要远少于主存中的块数， Cache 中仅保存主存中最活跃的若干块的副本。 因此，可按照某种策略预测 CPU 在未来 一 段时间内欲访存的数据，将其装入Cache 。 图3 . 17所示 为 Cache 的基本结构。



地址总线

由CPU完成

N 命中②

CPU

访问主存装入Cache 1YYY

直接通路 Cache

存储体

数据总线

主存地址 √ 块号 块内地址

lY\_

Cache地址

主存Cache 地址映射 变换机构

Cache Cache 替换机构

块号 块内地址

访问主 存替换

N可装进?

主

存

图3.17 高速缓冲存储器的工作原理

**命题追踪** ▶ **Cache 命** **中** **对CPU 执行时间影响的分析(2013、2015)**

当 C P U 发 出 读 请 求 时， 若访存地址在 Cache 中命中，就将此地址转换成 Cache 地址，直接 对 Cache 进行读操作，与主存无关；若 Cache 不命中，则仍需访问主存，并把此字所在的块 一 次 性 地 从 主 存 调 入Cache。 若 此 时 Cache 已满，则需根据某种替换算法，用这个块替换 Cache 中 原 来的某块信息。整个过程全部由硬件实现。值得注意的是， CPU 与 Cache 之间的数据交换以字为 单位，而 Cache 与主存之间的数据交换则以 Cache 块为单位。

当 C P U 发 出 写 请 求 时， 若 Cache 命中，有可能会遇到 Cache 与主存中的内容不 一 致的问题。 例如，由于CPU 写 Cache, 把 Cache 某单元中的内容从X 修 改 成X, 而主存对应单元中的内容仍 然 是 X, 没有改变，因此若 Cache 命中，需要按照 一 定的写策略处理，常见的处理方法有全写法 和回写法，详见本节的Cache 写策略部分。



**注** **意**

某些计算机中也采用同时访问Cache 和主存的方式，若Cache 命中，则终止访存。

**命** **题** **追** **踪** ▶ **C** **ache** **命中率的计算(2009)**

CPU 欲访问的信息已在 Cache 中的比率称为 Cache 的 命 中 率。设 一 个程序执行期间， Cache

118-2025年计算机组成原理考研复习指导

的总命中次数为N, 访问主存的总次数为Nm, 则命中率H 为

H=N₄/(Ne+Nm)

可见为提高访问效率，命中率H 越接近1越好。设t 为命中时的 Cache 访 问 时 间 ，tm为未命 中时的访问时间，1 -H 表示未命中率，则 Cache- 主存系统的平均访问时间 T₂ 为

T₄=H+(1-H)tm

▶ **Cache 缺失率对主存带宽的影响(2012)**

**【例3.3】**假 设Cache 的速度是主存的5倍，且Cache 的命中率为95%,则采用Cache 后，存 储器性能提高多少(假设采用先访问Cache,Cache 不命中时，才采用访问主存的方式)?

解 ： 设Cache 的存取周期为t, 主存的存取周期为5t, 得出系统的平均访问时间T 为 T=Cache 命中时的访问时间×命中率+Cache 缺失时的访问时间×缺失率

=0.95×1+0.05×(t+5t)=1.25t

或

T=Cache 命中时的访问时间+Cache 缺失时的访存开销×缺失率=1+0.05×5t=1.25t 可知，采用 Cache 后的存储器性能为原来的5t/1.25t≈4 倍 。

根据 Cache 的读、写流程，可知实现 Cache 时需解决以下关键问题：

1 ) 数 据 查 找。如何快速判断数据是否在 Cache 中。

2)地址映射。主存块如何存放在 Cache 中，如何将主存地址转换为Cache 地址。 3 ) 替 换 策 略 。Cache 满后，使用何种策略对 Cache 块进行替换或淘汰。

4)写入策略。如何既保证主存块和 Cache 块的数据一致性，又尽量提升效率。

**3.5.3** **Cache** **和** **主** **存** **的** **映** **射** **方** **式**

由 于Cache 行数比主存块数少得多，因此主存中只有一部分块的信息可放在Cache 中，因此 在 Cache 中要为每块加一个标记位，指明它是主存中哪一块的副本。该标记的内容相当于主存中 块的编号。为了说明 Cache 行中的信息是否有效，每个Cache 行需要 一个有效位。

Cache 行中的信息是主存中某个块的副本，地址映射是指把主存地址空间映射到 Cache 地 址 空间，即把存放在主存中的信息按照某种规则装入Cache 。 地址映射的方法有以下3种。

**1.** **直接映射**

主存中的每一块只能装入Cache 中的唯一位置。若这个位置已有内容，则产生块冲突，原来 的块将无条件地被替换出去(无须使用替换算法)。直接映射实现简单，但不够灵活，即使 Cache 的其他许多地址空着也不能占用，这使得直接映射的块冲突概率最高，空间利用率最低。

命 题 追 踪▶ **直接映射的地址结构及映射关系的分析(2010、2011、2015)**

直接映射的关系可定义为

Cache 行 号 = 主 存 块 号 mod Cache 总行数

假 设 Cache 共有2行，主存有2”块，在直接映射方式中，主存的第0块、第26块、第2e+1 块 … … 只能映射到 Cache 的第0行；而主存的第1块、第2°+1块、第20+1+1块 … … 只能映射 到 Cache 的第1行，以此类推。由映射函数可看出，主存块号的低c 位正好是它要装入的 Cache

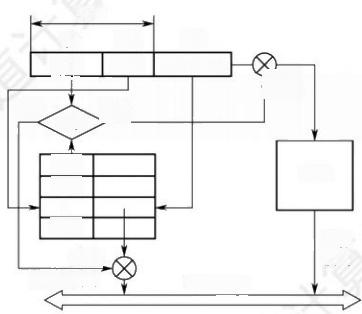
行号。给每个 Cache 行设置 一个长为t=m-c 的 标 记 (tag), 当主存某块调入Cache 后，就将其

块号的高t 位设置在对应 Cache 行的标记中，如图3. 18(a) 所示。

…



(a)Cache 和主存间的映射关系



b位

块内地址

个缺失

相等 比较 不等

标记

标记

:

标记

命中，

Cache读出

数据总线

m位 /位

c位 行号

主存 读出

主存

标记

(b)CPU 访存过程

图3 . 18 Cache 和主存之间的直接映射方式

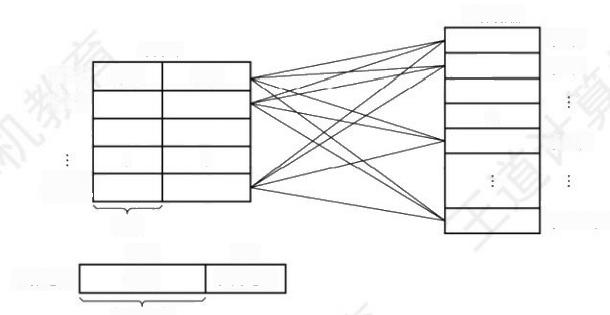
直接映射的地址结构为

|  |  |  |
| --- | --- | --- |
| 标记 | Cache行号 | 块内地址 |

CPU 访存过程如图3.18(b) 所示。首先根据访存地址中间的c 位，找到对应的Cache 行，将对 应Cache 行中的标记和主存地址的高t 位标记进行比较，若相等且有效位为1,则访问Cache“命 中”,此时根据主存地址中低位的块内地址，在对应的 Cache 行中存取信息；若不相等或有效位 为0,则“不命中”,此时 CPU 从主存中读出该地址所在的一块信息送到对应的 Cache 行中，将 有效位置1,并将标记设置为地址中的高t 位，同时将该地址中的内容送CPU。

**2.** **全相联映射**

主存中的每一块可以装入Cache 中的任何位置，如图3.19所示，每行的标记用于指出该行来 自主存的哪一块，因此CPU 访存时需要与所有 Cache 行的标记进行比较。优点：①Cache 块的冲 突概率低，只要有空闲Cache 行，就不会发生冲突；②空间利用率高；③命中率高；缺点：①标 记的比较速度较慢；②实现成本较高，通常需采用按内容寻址的相联存储器。



**主存储器**

第0块

Cache

第0行

第1行

;

第15块

:

第15行[

11位

11位

主存地址

主存块号

9位

块内地址

第2047块

第1块

数据

标记

标记

图3 . 19 Cache 和主存之间的全相联映射方式

全相联映射的地址结构为

|  |  |
| --- | --- |
| 标记 | 块内地址 |

120-2025年计算机组成原理考研复习指导

CPU 访存过程如下：首先将主存地址的高位标记(位数=log₂ 主存块数)与 Cache 各行的标 记进行比较，若有一个相等且对应有效位为1,则命中，此时根据块内地址从该 Cache 行中取出 信息；若都不相等，则不命中，此时CPU 从主存中读出该地址所在的一块信息送到Cache 的任意 一个空闲行中，将有效位置1,并设置标记，同时将该地址中的内容送CPU。

命 题 追 踪 ▶ **根据地址结构和比较器数量判断映射方式(2018)**

通常为每个Cache行都设置 一个比较器，比较器位数等于标记字段的位数。访存时根据标记 字段的内容来访问 Cache 行中的主存块，因而其查找过程是一种“按内容访问”的存取方式，所 以是一种“相联存储器”。这种方式的时间开销和硬件开销都较大，不适合大容量 Cache。

**3.** **组相联映射**

命 题 追 踪▶ **组相联映射方式的原理(2009、2016、2018～2020、2023)**

将 Cache 分成 Q 个大小相等的组，每个主存块可以装入固定组中的任意一行，即组间采用直 接映射、而组内采用全相联映射的方式，如图3.20所示。它是对直接映射和全相联映射的一种折 中，当Q=1 时变为全相联映射，当Q=Cache 行数时变为直接映射。路数越大，即每组 Cache 行的数量越大，发生块冲突的概率越低，但相联比较电路也越复杂。选定适当的数量，可使组相 联映射的成本接近直接映射，而性能上仍接近全相联映射。假设每组有r 个 Cache 行，则称为 r 路组相联，图3.20中每组有两个 Cache 行，因此称为二路组相联。

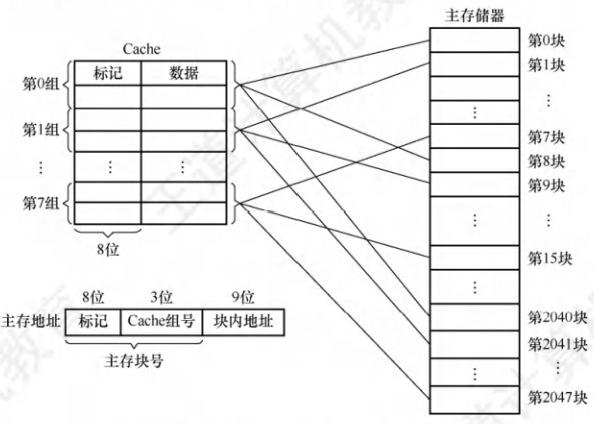


图3.20 Cache 和主存之间的二路组相联映射方式

组相联映射的关系可以定义为

Cache 组 号 = 主 存 块 号 mod Cache 组 数 (Q)

组相联映射的地址结构为

|  |  |  |
| --- | --- | --- |
| 标记 | 组号 | 块内地址 |

命 题 追 踪 ▶ **组相联映射的访存过程及** **Cache** **缺失处理过程(2020)**

CPU 访存过程如下：首先根据访存地址中间的组号找到对应的Cache 组；将对应 Cache 组中 每个行的标记与主存地址的高位标记进行比较；若有一个相等且有效位为1,则访问Cache 命中，

其中每一行代表 Cache的每一组

第3章 存储系统-121

此时根据主存地址中的块内地址，在对应Cache 行中存取信息；若都不相等或虽相等但有效位为 0,则不命中，此时CPU 从主存中读出该地址所在的一块信息送到对应Cache 组的任意一个空闲 行中，将有效位置1,并设置标记，同时将该地址中的内容送CPU。

▶ **组相联映射中比较器的个数和位数(2022)**

直接映射因为每块只能映射到唯一的Cache 行，因此只需设置1个比较器。而r 路组相联映 射需要在对应分组中与r 个 Cache 行进行比较，因此需设置r 个比较器。

**命** **题** **追** **踪** ▶ **直接映射、组相联映射相关标记位及总容量的分析(2010)**

**【例3.4】**假设某个计算机的主存地址空间大小为256MB, 按字节编址，其数据 Cache 有 8 个Cache 行，行长为64B。

1)若不考虑用于 Cache 的一致维护性位(脏位)和替换算法控制位，并且采用直接映射方

式，则该数据 Cache 的总容量为多少?

2 ) 若 该Cache 采用直接映射方式，则主存地址为3200(十进制)的主存块对应的Cache 行

号是多少?采用二路组相联映射时又是多少?

3)以直接映射方式为例，简述访存过程(设访存的地址为0123456H)。

解 ：

1 ) 因 为 Cache 包括了可以对 Cache 中所包含的存储器地址进行跟踪的硬件，即 Cache 的 总 容量=存储容量+标记阵列容量(有效位、标记位),本题不考虑脏位和替换算法位。

▶ **直接映射相关标记位的分析(2015、2021)**



**注** **意**

每个 Cache 行对应一个标记项(包括有效位、脏位、替换算法位、标记位),在组相联中， 将每组各行的标记项排成一行，将各组从上到下排列，构成一个二维的标记阵列。查找 Cache 时就是查找标记阵列的标记项是否符合要求。二路组相联的标记阵列如图3.21 所示。

|  |  |
| --- | --- |
| 标记项 | 标记项 |
| 标记项 | 标记项 |
| 标记项 | 标记项 |
| 标记项 | 标记项 |

|  |  |  |  |
| --- | --- | --- | --- |
| 有效位 | 脏位 | 替换控制位 | 标记位 |

图3.21 二路组相联的标记阵列示意图

因此本题中每行相关的存储器容量如图3.22所示。



1bit 19bit 64B=512bit

,有效位 标记位

该行对应的标记项 每行存储的数据

图3.22 Cache 行的存储容量示意图

标记字段长度的计算：主存地址有28位(256MB=2²°B), 其中6位为块内地址(2 °B=

64B),3 位为行号(2³=8),剩余28 - 6 - 3=19位为标记字段，故数据 Cache 的总容

量为8×(1+19+512)=4256位。



122-2025年计算机组成原理考研复习指导

2)直接映射方式中，主存按照块的大小划分，主存地址3200对应的字块号为3200B/64B= 50。而Cache 只 有 8 行 ， 则 5 0mod 8=2, 因此对应的 Cache 行号为2。

二路组相联映射方式，实质上就是将两个 Cache 行合并，内部采用全相联方式，外部采 用直接映射方式，50mod4=2, 对应的组号为2,即对应的Cache 行 号 为 4 或 5。

3)直接映射方式中，28位主存地址可分为19位的主存标记位，3位的块号，6位的块内地 址，即0000000100100011010为主存标记位，001 为块号，010110为块内地址。

首先根据块号，查Cache ( 即 0 0 1 号Cache 行)中对应的主存标记位，看是否相同。若 相 同，再看 Cache 行中的有效位是否为1,若是，称此访问命中，按块内地址010110读出 Cache 行所对应的单元并送入CPU 中，完成访存。

若出现标记位不相等或有效位为0的情况，则不命中，访问主存将数据取出并送往 CPU 和 Cache 对应块中，把主存地址的高19位写入001 行的标记位，并将有效位置1。

**思** **考：**①若第 一 问中采用二路组相联，则 Cache 总容量是多少?②仔细分析主存划分和 Cache 划分的关系，自行推导二路组相联映射方式的主存地址划分和访存过程。

三种映射方式中，直接映射的每个主存块只能映射到Cache 中的某一 固定行；全相联映射可以

映射到所有 Cache 行 ；N 路组相联映射可以映射到N 行 。 当 Cache 大小、主存块大小一定时，

1)直接映射的命中率最低，全相联映射的命中率最高。

2)直接映射的判断开销最小、所需时间最短，全相联映射的判断开销最大、所需时间最长。 3)直接映射标记所占的额外空间开销最少，全相联映射标记所占的额外空间开销最大。

**3.5.4 Cache 中主存块的替换算法**

在采用全相联映射或组相联映射方式时，从主存向 Cache 传送一个新块，当 Cache 或 Cache 组中的空间已被占满时，就需要使用替换算法置换 Cache 行。而采用直接映射时，一个给定的主 存块只能放到唯一 的固定 Cache 行中，所以在对应Cache 行已有一个主存块的情况下，新的主存 块毫无选择地把原先已有的那个主存块替换掉，因而无须考虑替换算法。

常用的替换算法有随机 (RAND) 算法、先进先出 (FIFO) 算法、近期最少使用 (LRU) 算 法和最不经常使用 (LFU) 算 法。其中最常考查的是LRU 算法。

1)随机算法：随机地确定替换的 Cache 行。它的实现比较简单，但未依据程序访问的局部 性原理，因此可能命中率较低。

2)先进先出算法：选择最早调入的 Cache 行进行替换。它比较容易实现，但也未依据程序 访问的局部性原理，因为最早进入的主存块也可能是目前经常要用的。

命 题 追 隙 ▶》**组相联映射** **中LRU 算法的命中分析(2012、2021)**

3)近期最少使用算法(L RU): 依据程序访问的局部性原理，选择近期内长久未访问过的

Cache 行进行替换，其平均命中率要比 FIFO 的高，是堆栈类算法。

命 题 追 踪▶ **LRU 替换位及其位数的计算(2018、2020)**

LRU 算法对每个 Cache 行设置一个计数器( 也 称 L RU替换位 ),用计数值来记录主存块的使 用情况，并根据计数值选择淘汰某个块，计数值的位数与Cache 组大小有关，二路时有1位LRU 位，四路时有2位LRU 位。假定采用四路组相联，有5个主存块{1,2,3,4,5}映射到Cache 的 同

① 本考点建议结合《操作系统考研复习指导》复习。



**第3章** **存储系统** -123

一组，对于主存访问序列{1,2,3,4,1,2,5,1,2,3,4,5},采用LRU 算法的替换过程如图3.23所示。 图中左边阴影的数字是对应 Cache 行的计数值，右边的数字是存放在该行中的主存块号。

2 3 4 1 2 5 2 3 4 5

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0| |  |  | 2|1 | | 3|1 | 0|1 | 1! | 1 | 2 | 1 | 01 | | 11 | 2!1 |  | 0|5 |
|  | 0 | 2 | 1 | 2 | 2|2 | 3|2 | 0 | 2 | 1|2 | | 2i | 2 | 0|2 | 112 |  | 3|2 |
|  |  | | 0 | 3 | 1|3 | 2!3 | 3 | 3 | 0|5 | |  | | 2 5 | 3|5 |  | 1|4 |
|  |  |  |  |  | 0|4 | 114 | 2 | 4 | 3|4 | | 4 | | 3|4 | 0|3 | 113 | 2|3 |

图3.23 LRU 算法的替换过程示意图

计数器的变化规则：①命中时，所命中的行的计数器清零，比其低的计数器加1,其余不变； ②未命中且还有空闲行时，新装入的行的计数器置0,其余全加1;③未命中且无空闲行时，计 数值为3的行的信息块被替换，新装入的行的计数器置0,其余全加1。

当集中访问的存储区超过 Cache 组的大小时，命中率可能变得很低，如上例的访问序列变为 1,2,3,4,5,1,2,3,4,5,…,而Cache 每组只有4行，则命中率为0,这种现象称为抖动。

4)最不经常使用算法：将一段时间内被访问次数最少的 Cache 行换出。每行也设置一个计 数器，新行装入后从0开始计数，每访问一次，被访问的行计数器加1,需要替换时比较 各特定行的计数值，将计数值最小的行换出。这种算法与LRU 类似，但不完全相同。

**3.5.5** **Cache 的** **一** **致** **性** **问** **题**

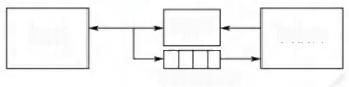
因为 Cache 中的内容是主存块副本，当对 Cache 中的内容进行更新时，就需选用写操作策略 使 Cache 内容和主存内容保持一致。此时分两种情况。

对于 Cache写操作命中 (write hit), 有两种处理方法。

命题追踪 ▶ **直写法的特点(2015)、直写法是否需设修改位(2020)**

1)全写法(直写法、write-through)。当 CPU 对 Cache 写命中时，必须把数据同时写入 Cache

和主存。当某一块需要替换时，就不必把这一块写回主存了，用新调入的块直接覆盖即 可。这种方法实现简单，能随时保持主存数据的正确性。缺点是增加了访存次数，降低 了 Cache 的效率。写缓冲：为减少全写法直接写入主存的时间损耗，在 Cache 和主存之 间加一个写缓冲(Write Buffer),如下图所示。CPU 同时写数据到 Cache 和写缓冲中，写 缓冲再将内容写入主存。写缓冲是一个 FIFO 队列，写缓冲可以解决速度不匹配的问题。 但若出现频繁写时，会使写缓冲饱和溢出。



Cache

DRAM

Write Buffer

CPU

命题追踪 **回写法的修改位(2018、2020)**

2)回写法 (write-back) ①。当 CPU 对 Cache 写命中时，只把数据写入 Cache, 而不立即写 入主存，只有当此块被替换出时才写回主存。这种方法减少了访存次数，但存在数据不 一致的隐患。为了减少写回主存的次数，给每个Cache 行设置一个修改位 (脏位)。若修 改位为1,则说明对应Cache 行中的块被修改过，替换时须写回主存；若修改位为0,则 说明对应 Cache 行中的块未被修改过，替换时无须写回主存。

① 大多数教材将其翻译为写回法，但2015年和2021年统考真题都采用回写法，故本书采用该名称。

124-2025年计算机组成原理考研复习指导

全写法和回写法都对应于Cache 写命中(要被修改的块在 Cache 中)时的情况。

对于 Cache写操作不命中，也有两种处理方法。

1)写分配法 (write-allocate) 。 更新主存单元，然后把这个主存块调入Cache 。它试图利用程

序的空间局部性，缺点是每次写不命中都要从主存读一个块到Cache 中。

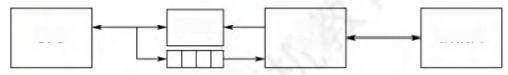
2)非写分配法 (not-write-allocate)。只更新主存单元，而不把主存块调入Cache。 非写分配法通常与全写法合用，写分配法通常和回写法合用。

**命** **题** **追** **踪**▶ **采用分离的指令** **Cache 和数据** **Cache 的主要目的(2014)**

随着指令流水技术的发展，需要将指令Cache 和数据 Cache 分开设计，这就有了分离的Cache 结 构。 统 一Cache 的优点是设计和实现相对简单，但由于执行部件存取数据时，指令预取部件要 从同 一Cache 读指令，因此会引发冲突。采用分离的Cache 结构可以解决这个问题，而且分离的 指令和数据 Cache 还可以充分利用指令和数据的不同局部性来优化性能。

现代计算机的 Cache 通常设立多级 Cache, 假 定 设 2 级Cache, 按 离 CPU 的远近可各自命名 为 L₁Cache 、L2 Cache, 离 CPU 越远，访问速度越慢，容量越大。指令Cache 与 数 据Cache 分 离 一 般在 L1 级，此时通常为写分配法与回写法合用。下图是一个含有两级Cache 的系统， L₁Cache

对 L2 Cache 使 用 全 写 法 ，L2 Cache 对主存使用回写法，由于L2 Cache 的存在，其访问速度大于 主存，因此避免了因频繁写时造成的写缓冲饱和溢出。

CPU Cache L2 DRAM

Cache

Write Buffer

**3.5.6** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 在高速缓存系统中，主存容量为12MB,Cache 容量为400KB, 则该存储系统的容量为()。

A.12MB+400KB B.12MB

C.12MB-12MB+400KB D.12MB-400KB

02. 访 问Cache 系统失效时，通常不仅主存向CPU 传送信息，同时还需要将信息写入 Cache, 在此过程中传送和写入信息的数据宽度各为( )。

A. 块、页 B. 字、字

C. 字 、 块 D. 块、块

**03.** 假定用作Cache 的 SRAM 的存取时间为2ns, 用作主存的SDRAM 的存取时间为40ns。 为使存储系统的平均存取时间达到3ns, 则 Cache 命 中 率 应 达 到 ( ) 左 右 。

A.92.5% B.85% C.97.5% D.99.9%

**04.** 关 于 Cache 的更新策略，下列说法中正确的是( )。

A. 读操作时，全写法和回写法在命中时应用

B. 写操作时，回写法和写分配法在命中时应用

C. 读操作时，全写法和写分配法在失效时应用

D. 写操作时，写分配法、非写分配法在失效时应用

**05.** 在不同的情况下，需要采用适合的Cache 写策略。对于下面两种情况：①主要运行访问 密集型应用，其中包含写操作；②安全性要求很高，不允许有任何数据不一致的情况发 生。适合它们的写策略分别是()。

第3章 存储系统 125

A. 回写法，全写法 B. 全写法，回写法

C. 回写法，回写法 D. 全写法，全写法

**06.** 局部性通常有两种不同的形式：时间局部性和空间局部性。程序员是否编写出高速缓存

友好的代码，就取决于这两方面的问题。对于下面这个函数，说法正确的是()。



int sumvec(int v[N]){

int i,sum=0;

for(i=0;i<N;i++)

sum+=v[i];

return sum;

}

A. 对于变量i 和 sum, 循环体具有良好的空间局部性

B. 对于变量i 、sum 和v[N], 循环体具有良好的空间局部性

C. 对于变量 i 和 sum, 循环体具有良好的时间局部性

D. 对于变量i 、sum 和 v[N], 循环体具有良好的时间局部性

**07.** 对于下列代码，以下哪种变化将使其具有更好的空间局部性()。

① int i,j,k,sum=0;

② for(i=0;i<n;i++)

③ for(j=0;j<n;j++)

④ for(k=0;k<n;k++)

⑤ sum+=a[k][j][i];

A. 将第2行与第3行互换 B. 将第2行与第4行互换

C. 将第5行改为sum+=a[i][k][]; D. 将第5行改为sum+=a[j][i][k];

**08.** 下列关于高速缓存Cache 的描述中，正确的是()。

A.Cache 的功能全部由硬件实现

B.Cache 替换时的单位为字

C.Cache 与主存统一编址，即主存地址空间的某一部分属于Cache

D. 无论何时，Cache 中的信息一定与主存中的信息一致

**09.** 下列关于Cache 的描述中，比较合理的是( )。

I. 指令 Cache 通常比数据 Cache 具有更好的空间局部性

II. 由于空间局部性，适当增加 Cache块大小通常会提高命中率

ⅢI.回写法的写主存操作次数少于写直达法

A.Ⅲ B.I 和 Ⅱ C.Ⅱ 和 Ⅲ D.I 和Ⅱ和Ⅲ

**10.** 某虚拟存储器系统采用页式内存管理，使用 LRU 页面替换算法，考虑下面的页面访问 地址流(每次访问在一个时间单位中完成):

18 178272 1 83 82 1 317 1 3 7

假定内存容量为4个页面，开始时是空的，则页面失效率是( )。

A.30% B.5% C.1.5% D.15%

**11.** 某个具有两级 Cache 的存储系统中，访存时依次通过两级 Cache, 某程序在执行过程中 访存1000次，其中访问第一级 Cache 时有40次不命中，接着访问第二级 Cache, 仍有 10次不命中，则总命中率是()。

A.99% B.96% C.95% D.97%

**12.** 假设一个Cache 中共有M 块，每 K 块组成一个组，则下列描述中正确的是( )。

126-2025年计算机组成原理考研复习指导

A. 若 K=1, 则该 Cache是直接映射 Cache

B. 若 K=1, 则该 Cache 是全相联映射 Cache

C. 若 K=M, 则该 Cache 是直接映射 Cache

D. 若 K>1 且 K<M, 则该Cache 是M/K-路组相联映射 Cache

**13.** 在Cache 中，常用的替换策略有随机法 (RAND) 、 先进先出法 (FIFO) 、 近期最少使用 法 (LRU), 其中与局部性原理有关的是()。

A. 随机法 (RAND) B. 先进先出法 (FIFO)

C. 近期最少使用法(LRU) D. 都不是

**14.** 某存储系统中，主存容量是Cache容量的4096倍，Cache被分为64个块，当主存地址 和 Cache 地址采用直接映像方式时，地址映射表的大小应为( )。(假设不考虑一致维 护和替换算法位。)

A.6×4097bit B.64×12bit C.6×4096bit D.64×13bit

**15.** 有效容量为128KB的Cache, 每块16B, 采用8路组相联。字节地址为1234567H 的单 元调入该 Cache, 则其Tag 应为( )。

A.1234H B.2468H C.048DH D.12345H

**16.** 有一主存-Cache 层次的存储器，其主存容量为1MB,Cache 容量为16KB, 每块有8个 字，每字32位，采用直接地址映像方式，Cache起始字块为第0块，若主存地址为35301H, 且CPU访 问Cache命中，则在Cache 的第()(十进制表示)字块中。

A.152 B.153 C.154 D.151

**17**. 对于由高速缓存、主存、硬盘构成的三级存储体系， CPU 直接根据( )进行访问。

A. 高速缓存地址 B. 虚拟地址 C. 主存物理地址 D. 磁盘地址

**18.** 设有8页的逻辑空间，每页有1024B, 它们被映射到32块的物理存储区中，则按字节 编址逻辑地址的有效位是(),物理地址至少是()位。

A.10,12 B.10,15 C.13,15 D.13,12

**19.** 对于n-路组相联映射 Cache, 在保持n 及主存和Cache 总容量不变的前提下，将主存块

大小和 Cache 块大小都增加一倍，则下列描述中正确的是()。

A. 字块内地址的位数增加1位，主存tag 字段的位数增加1位

B. 字块内地址的位数增加1位，主存tag字段的位数不变

C. 字块内地址的位数减少1位，主存tag 字段的位数增加1位

D. 字块内地址的位数增加1倍，主存tag 字段的位数减少一半

**20.** 某计算机的 Cache有16行，块大小为16B, 其映射方式可配置为直接映射或2-路组相 联映射，主存按字节编址，主存单元从0开始编号。若依次访问下列主存单元，则不论 采取上述哪种映射方式都可能引起 Cache 冲突的是()。

A.52 号和102号单元 B.48 号和308号单元

C.60 号和160号单元 D.46 号和236号单元

**21.** 假设主存地址位数为32位，按字节编址，主存和 Cache 之间采用全相联映射方式，主 存块大小为1个字，每字32位，采用回写 (write back)方式和随机替换策略，则能存 放32K 字数据的 Cache 的总容量至少应有()位。

A.1536K B.1568K C.2016K D.2048K

**22.** 假设主存按字节编址，Cache 共有64行，采用四路组相联映射方式，主存块大小为32 字节，所有编号都从0开始。则第2593号存储单元所在主存块的 Cache组号是()。

A.1 B.15 C.14 D.4

**23.** 假定 CPU 通过存储器总线读取数据的过程为：发送地址和读命令需1个时钟周期，存 储器准备一个数据需8个时钟周期，总线上每传送1个数据需1个时钟周期。若主存和 Cache之间交换的主存块大小为64B, 存取宽度和总线宽度都为8B, 则 Cache的一次缺 失损失至少为( )个时钟周期。

A.64 B.72 C.80 D.160

**24.** 假定8个存储器模块采用交叉方式组织，存储器芯片和总线支持突发传送， CPU通过存 储器总线读取数据的过程为：发送首地址和读命令需1个时钟周期，存储器准备第一个 数据需8个时钟周期，随后每个时钟周期总线上传送1个数据，可连续传送8个数据(即 突发长度为8)。若主存和Cache 之间交换的主存块大小为64B, 存取宽度和总线宽度都 为8B, 则 Cache 的一次缺失损失至少为()个时钟周期。

*A.17* B.20 C.33 D.80

**25.** 下列关于Cache替换算法的叙述中，错误的是()。

A. 组相联映射和全相联映射都必须考虑如何进行替换

B. 先进先出算法无须对每个 Cache 行记录替换信息

C. 直接映射是多对一的映射，无须考虑替换问题

D.LRU 算法需要对每个 Cache行记录替换信息

**26.** 下列关于Cache大小、主存块大小和Cache 缺失率之间关系的叙述中，错误的是( )。

A. 主存块大小和 Cache 容量无直接关系

B.Cache 容量越大， Cache 缺失率越低

C. 主存块大小通常为几十到上百字节

D. 主存块越大， Cache 缺失率越低

**27.**【2009统考真题】假设某计算机的存储系统由Cache 和主存组成，某程序执行过程中访 存1000次，其中访问Cache 缺失(未命中)50次，则 Cache的命中率是( )。

A.5% B.9.5% C.50% D.95%

**28.【**2009统考真题】某计算机的 Cache 共有16块，采用二路组相联映射方式(即每组2 块)。每个主存块大小为32B, 按字节编址，主存129号单元所在主存块应装入的 Cache 组号是()。

A.0 B.2 C.4 D.6

**29.**【2012统考真题】假设某计算机按字编址， Cache 有4行， Cache 和主存之间交换的块大 小为1个字。若 Cache 的内容初始为空，采用二路组相联映射方式和 LRU 替换策略， 则访问的主存地址依次为0,4,8,2,0,6,8,6,4,8时，命中Cache 的次数是( )。【提示，

本题的映射方式与本书所讲的映射方式不同，具体见解析部分的“注意”】 A.1 B.2 C.3 D.4

**30.**【2016统考真题】有如下C 语言程序段：

for(k=0;k<1000; k++)

a[k]=a[k ]+32;

若数组a 和变量k 均为int 型 ，int 型数据占4B, 数据 Cache采用直接映射方式，数据区 大小为1KB、块大小为16B, 该程序段执行前Cache 为空，则该程序段执行过程中访问 数组 a 的 Cache 缺失率约为( )。

A.1.25% B.2.5% C.12.5% D.25%

128-2025年计算机组成原理考研复习指导

**31.【2**017统考真题】某C 语言程序段如下：



for(i=0;i<=9;i++){

temp=1;

for(j=0;j<=i;j++)temp\*=a[j];

sum +=temp;

}

下列关于数组 a 的访问局部性的描述中，正确的是( )。

A. 时间局部性和空间局部性皆有 B. 无时间局部性，有空间局部性

C. 有时间局部性，无空间局部性 D. 时间局部性和空间局部性皆无

**32.【**2021统考真题】若计算机主存地址为32位，按字节编址， Cache 数据区大小为32KB, 主存块大小为32B, 采用直接映射方式和回写 (Write Back)策略，则 Cache 行的位数 至 少 是 ( ) 。

A.275 B.274 C.258 D.257

**33.** 【2022统考真题】若计算机主存地址为32位，按字节编址，某 Cache 的数据区容量为 32KB, 主存块大小为64B, 采用8路组相联映射方式，该 Cache 中比较器的个数和位数 分别为()。

A.8,20 B.8,23 C.64,20 D.64,23

**二、综合应用题**

**01.** 假定某处理器可通过软件对高速缓存设置不同的写策略，则在下列两种情况下，应分别 设置成什么写策略?为什么?

1)处理器主要运行包含大量存储器写操作的数据访问密集型应用。

2)处理器运行程序的性质与1)相同，但安全性要求高得多，不允许有任何数据不一致

的情况发生。

**02.** 某计算机的主存地址位数为32位，按字节编址。假定数据 Cache 中最多存放128个主 存块，采用四路组相联方式，块大小为64B, 每块设置了1位有效位。采用随机替换算 法，写磁盘采用回写策略，为此每块设置了1位“脏”位。要求：

1)分别指出主存地址中标记 (Tag) 、 组 号 (Index) 和块内地址 (Offset) 三部分的位置

与位数。

2)计算该数据 Cache 的总位数。

**03.** 某个 Cache 的容量大小为64KB, 行长为128B, 且是四路组相联 Cache, 主存使用32 位地址，按字节编址。

1 ) 该 Cache 共有多少行?多少组?

2 ) 该 Cache 的标记阵列中需要有多少标记项?每个标记项中标记位长度是多少?

3 ) 该 Cache 采用 LRU 替换算法，若当该 Cache为写直达式 Cache 时，标记阵列总共需 要多大的存储容量?回写式又该如何?(提示：四路组相联 Cache 使用 LRU 算法的

替换控制位为2位。)

**04.** 某计算机有容量为256B 的数据 Cache, 主存块大小为32B。 现有如下C 语言程序段：



int i,j,c,s,a[128];

for(i=0;i<10000;i++)

for(j=0;j<128;j=j+s)

c=a[j];

int 型数据用32位补码表示，编译器将变量i,j,c,s 都分配在通用寄存器中，因此，只需

考虑数组元素的访存情况，假定数组起始地址正好在一个主存块的开始。请回答： 1 ) 若 Cache 采用直接映射，则当s=64 和 s=63 时，缺失率分别为多少?

2 ) 若 Cache 采用2-路组相联映射，则当s=64 和 s=63 时，缺失率分别为多少?

**05.**【2010统考真题】某计算机的主存地址空间大小为256MB, 按字节编址。指 令Cache 和

数据 Cache 分离，均有8个Cache 行，每个 Cache 行大小为64B, 数 据 Cache 采用直接 映射方式。现有两个功能相同的程序A 和 B, 其伪代码如下所示：



程序 B:

int a[256][256];

…

int sum\_array2()

{

int i,j,sum=0; for(j=0;j<256;j++)

for(i=0;i<256;i++) sum +=a[i][j];

return sum;

}

程序A:

int a[256][256];

…

int sum\_array1()

int i,j,sum=0; for(i=0;i<256;i++)

for(j= 0;j<256;j++) sum +=a[i][j];

return sum;

}

假定int型数据用32位补码表示，程序编译时， i、j和 sum 均分配在寄存器中，数组 a 按行优先方式存放，其首地址为320(十进制数)。请回答下列问题，要求说明理由或给 出计算过程。

1)不考虑用于Cache 一致性维护和替换算法的控制位，数据 Cache 的总容量为多少? 2)数组元素 a[0][31] 和 a[1][1] 各自所在的主存块对应的 Cache 行号是多少 (Cache 行号

从 0 开 始 ) ?

3 ) 程 序A 和 B 的数据访问命中率各是多少?哪个程序的执行时间更短?

**06.【**2013统考真题】某32位计算机， CPU 主频为800MHz,Cache 命中时的CPI 为 4 ,Cache 块大小为32B; 主存采用8体交叉存储方式，每个体的存储字长为32位、存储周期为 40ns; 存储器总线宽度为32位，总线时钟频率为200MHz, 支持突发传送总线事务。每 次读突发传送总线事务的过程包括：传送首地址和命令、存储器准备数据、传送数据。 每次突发传送32B, 传送地址或32位数据均需要一个总线时钟周期。请回答下列问题， 要求给出理由或计算过程。

1)CPU 和总线的时钟周期各为多少?总线的带宽(即最大数据传输速率)为多少? 2)Cache 缺失时，需要用几个读突发传送总线事务来完成一个主存块的读取?

3)存储器总线完成一次读突发传送总线事务所需的时间是多少?

4)若程序 BP 执行过程中共执行了100条指令，平均每条指令需进行1.2次访存， Cache 缺失率为5%,不考虑替换等开销，则 BP 的 CPU执行时间是多少?

**07** . 【2020统考真题】假定主存地址为32位，按字节编址，指令 Cache 和数据 Cache 与主存 之间均采用8路组相联映射方式，直写 (Write Through) 写策略和 LRU 替换算法，主 存块大小为64B, 数据区容量各为32KB 。开始时 Cache 均为空。请回答下列问题。

1)Cache 每一行中标记 (Tag) 、LRU 位各占几位?是否有修改位?

2)有如下C 语言程序段：

s[k]=2 \*s[k];

k++)

for (k=0;k<1024;

若数组s 及其变量k 均 为int 型 ，int 型数据占4B, 变 量k 分配在寄存器中，数组 s

在主存中的起始地址为008000COH, 则在该程序段执行过程中，访问数组s 的数据

130-2025年计算机组成原理考研复习指导

Cache 缺失次数为多少?

3 ) 若 CPU 最先开始的访问操作是读取主存单元00010003H 中的指令，简要说明从 Cache 中访问该指令的过程，包括 Cache 缺失处理过程。

**3.5.7** **答** **案** **与** **解** **析**

**一** **、单项选择题**

**01.B**

选 项A 为干扰项。各层次的存储系统不是孤立工作的，三级结构的存储系统是围绕主存储器 来组织、管理和调度的存储器系统，它们既是一个整体，又要遵循系统运行的原理，其中包括包 含性原则。因为Cache 中存放的是主存中某一部分信息的副本，所以不能认为总容量为两个层次 容量的简单相加。

**02.** C

一 个块通常由若干字组成，CPU 与 Cache (或主存)间信息交互的单位是字，而Cache 与 主 存间信息交互的单位是块。当 CPU 访问的某个字不在 Cache 中时，将该字所在的主存块调入 Cache, 这 样 CPU 下次要访问的字才有可能在 Cache 中 。

**03.** C

Cache 命中时的存取时间为2ns;Cache 不命中时先访问Cache, 再访问主存，总存取时间为

42ns。 设 Cache 命中率为x, 则平均存取时间为2×x+42×(1-x)=3, 解 得x=97.5%。

**04.D**

在写不命中时，加载相应的低一层中的块到高速缓存 (Cache) 中，然后更新这个高速缓存 块，称为写分配法；而避开 Cache, 直接把这个字写到主存中，则称为非写分配法。这两种方法 都是在不命中Cache 的情况下使用的，而回写法和全写法是在命中Cache 的情况下使用的。在写 Cache 时，写分配法和回写法搭配使用，非写分配法和全写法搭配使用。

05.A

写操作比较密集，采用回写法速度快，更适合访问密集型的应用。全写法每次均写入主存和 Cache, 能够随时保持主存数据的一致性，适合安全性要求很高的应用。

**06.** C

时间局部性是指一个内存位置被重复引用，循环体中的变量i 和 sum 具有良好的时间局部性。 空间局部性是指若一个内存位置被引用，则它附近的位置很快也会被引用，因为指令通常是顺序 存放、顺序执行的，数据一般也是以向量、数组等形式存储的，v[N] 具有良好的空间局部性。

07.B

空间局部性是指程序在一段时间内所访问的存储空间的集中度。为了提高空间局部性，应尽 量按照数组在内存中的存储顺序依次访问数组元素。根据C 语言的规定，数组a 在内存中是按最 右下标变化最快的方式存储的，即a[0][0][0],a[0][0][1],…,a[0][0][n-1],a[0][1][0],…,a[0][n-1][n-1],

a[1][0][0]…,a[r-1][n-1][n-1] 。 因此，若将代码的第2行与第4行互换，则可使得对数组 a 的 访 问变成顺序访问，从而提高空间局部性。

**08.A**

Cache 的功能完全由硬件实现，A 正 确 。Cache 替换时的单位是块，而不是字或字节，因为 Cache 和主存是以块为单位进行数据交换的。Cache 地址空间和主存地址空间相互独立，通过地 址映射把主存地址空间映射到Cache 地址空间。Cache 中的信息不一定与主存中的信息一致，因 为 Cache 可能采用回写策略，只有当被修改的块被换出时才写回主存。

第3章 存储系统 -131

09.D

指令Cache 通常比数据 Cache具有更好的空间局部性，这是因为指令流通常是顺序执行的， 而数据流跳转或随机访问的概率较高， I 正确。由于空间局部性，同一主存块中的数据的访问概 率较高，因此增加Cache块大小会提高命中率，Ⅱ正确。写回法只有在被修改的块被换出时才写 回主存，而写直达法每次写操作都会同时写回主存，Ⅲ正确。

10.A

LRU 表如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 单元1 |  |  |  |  |  | 2 | 7 | 2 | 1 | 8 | 3 | 8 | 2 |  | 3 |  | 7 | 1 | 3 | 7 |
| 单元2 |  |  |  | 7 | 8 | 8 | 2 | 7 | 2 | 1 | 8 | 3 | 8 | 2 | 1 | 3 | 1 | 7 | 1 | 3 |
| 单元3 |  | 8 | 1 | 1 | 7 | 7 | 8 | 8 | 7 | 2 | 1 | 1 | 3 | 8 | 2 | 2 | 3 | 3 | 7 | 1 |
| 单元4 | 1 | 1 | 8 | 8 | 1 | 1 | 1 | 1 | 8 | 7 | 2 | 2 | 1 | 3 | 8 | 8 | 2 | 2 | 2 | 2 |
| 命中否 | 否 | 否 |  | 否 |  | 否 |  |  |  |  | 否 |  |  |  |  |  | 否 |  |  |  |

可见页面失效率是6/20=30%。

11.A

访存1000次，访问第2级Cache 后，仍有10条不命中，故总命中率为1-10/1000=99%。

12.A

K=1 即一块为一组，每个主存块映射到唯一的Cache块，为直接映射， A 正确，B 错误。K=M 即每个主存块可映射到所有 Cache 块中，为全相联映射，C 错误。D 应为 K-路组相联映射。

13.C

LRU 算法根据程序访问局部性原理选择近期使用得最少的存储块作为替换的块。

14.D

地址映射表即标记阵列，由于Cache 被分为64个块，因此Cache 有64行，采用直接映射， 一行相当于一组。因此标记阵列每行存储一个标记项，其中主存标记项为12位(2l²=4096, 是 Cache 容量的4096倍，即地址长度比Cache 长12位),加上1位有效位，因此为64×13位。

15.C

块大小为16B, 所以块内地址字段为4位； Cache 容量为128KB, 采用8 路组相联，共有 128KB÷(16B×8)=1024 组，组号字段为10位；剩下的为标记字段。1234567H 转换为二进制数0001 001000110100010101100111,标记字段对应高14位，即048DH。

16.A

先写出主存地址的二进制形式，然后分析 Cache 块内地址、Cache 字块地址和主存字块标记。 主存地址的二进制数00110101001100000001,根据直接映射的地址结构，字块内地址为低5位 (每个字块32B,2⁵=32, 因此为5位),主存字块标记为高6位(1MB/16KB=64,2⁶=64, 因 此为6位),其余010011000即为 Cache 字块地址，转换为十进制数152。

17.C

当 CPU 访存时，先要到 Cache 中查看该主存地址是否在 Cache 中，所以发送的是主存物理 地址。只有在虚拟存储器中，CPU 发出的才是虚拟地址，这里并未指出是虚拟存储系统。磁盘地 址是外存地址，外存中的程序由操作系统调入主存中，然后在主存中执行，因此CPU 不可能直接 访问磁盘。

18.C

对于逻辑地址，因为8=2³页，所以表示页号的地址有3位，又因为每页有1024=2l°B, 所

132-2025年计算机组成原理考研复习指导

以页内地址有10位，因此逻辑地址共13位。

对于物理地址，块内地址和页内地址一样有10位，内存至少有32=25个物理块，所以表示

块号的地址至少有5位，因此物理地址至少有15位。

**19.B**

组相联映射的主存地址结构为： tag 标 记 +Cache 组 号 + 字 块 内 地 址 。Cache 块大小增加一 倍，则字块内地址的位数增加1位。Cache 组 数 =(Cache 总容量/Cache 块 大 小 )n, 故 Cache 组数 减 少 一 半 ；Cache 组 号 = 主 存 块 号MOD Cache组数，故 Cache 组号也减少1位。主存总容量不 变，则主存地址总长度不变，字块内地址和 Cache 组 号 一 个 增 1 一 个 减 1 , 因 此 tag 字段的位数 不变。

**20.B**

块大小为16B=2\*B, 所以块内地址占4位。若采用直接映射， Cache 共16行，主存地址中 块内地址的前4位为Cache 行号， Cache 行 号 = 主 存 块 号 %Cache 总行数=(主存地址/16)%16, 选项B 的 地 址 4 8 和 3 0 8 的Cache 行号均为3,产生冲突。若采用2-路组相联映射，共有16/2=8 组，主存地址中块内地址的前3位为Cache 组 号 ，Cache 组 号 = 主 存 块 号 %Cache 组 数 = ( 主 存 地址/16)%8,选项B 的 地 址 4 8 和 3 0 8 的Cache 组号均为3,可能产生冲突。

21.D

主存块大小为1个字，即32位，按字节编址，所以块内地址占2位。在全相联映射方式下， 主存地址只有两个字段，所以标志占32 - 2=30位。 因采用回写法，故需1位修改位；因为采用 随机替换策略，故无须替换控制位。每个 Cache 行的总位数为32bit ( 数 据 位 ) + 3 0bit(tag 位 ) + 1bit ( 修 改 位 ) + 1bit ( 有 效 位 ) = 6 4bit 。综 上 ，Cache 总容量至少应有32K×64bit=2048K bit。

**22.A**

主存块大小为32 字节，按字节编址，所以块内地址占5位。采用四路组相联映射方式，共 64行，分64/4=16组，故组号占4位。因为2593=0 … 0101000100001,根据主存地址划分的结 果，可以看出第2593号存储单元所在主存块的 Cache 组号为0001。

**23.** C

一 次缺失损失需要从主存读出一个主存块(64B), 每个总线事务读取8B, 因 此 需 要 8 个 总

线事务。每个总线事务所用的时间为1+8+1=10个时钟周期，共需要80个时钟周期。

**24.** A

一次缺失损失需要从主存读出一个主存块( 64B ), 每个突发传送总线事务可读取8B×8 =64B, 因此只需要一个突发传送总线事务。首先，发送首地址和读命令需要一个时钟周期，然后轮流启 动每个存储器模块，每隔一个时钟周期启动一个存储器模块，采用流水线工作方式，所以每个突 发传送总线事务所用的时间为1+8+8=17个时钟周期，因此共需17个时钟周期。

**25.B**

对于直接映射，主存中的每一块只能装入Cache 中的唯一位置，若产生块冲突，原来的块将 被无条件换出，因此无须考虑替换问题，而组相联映射和全相联映射都需要考虑替换问题。先 进 先出算法需要对每个 Cache 行打一个时间戳，记录何时装入了一个新主存块。

**26.D**

主存块太小，不能很好地利用空间局部性，从而导致缺失率变高；但主存块太大也会使得 Cache 行数变少，即 Cache 中可以存放主存块的位置变少，从而也会降低命中率。因此，主存块 大小应该适中，既不能太大，又不能太小，通常为几十字节到上百字节。

27.D

命 中 率 =Cache 命中次数/总访问次数。注意看清题目，题中说明的是缺失50次，而不是命 中50次，仔细审题是做对题的第一步。

**28.** C

由于 Cache 共有16块，采用二路组相联，共分为8组，组号为0,1,2, … ,7,组号占3位。 主存块大小为32B, 按字节编址，块内地址占5位。主存单元地址129=0 … 010000001,后5位 是块内地址，块内地址的前3位是组号，因此将映射到组号4的任意 一个 Cache 块 中。

29.C

地址映射采用二路组相联，主存字地址为0～1、4～5、8～9可映射到第0组 Cache 中，主 存地址为2～3、6～7可映射到第1组 Cache 中。Cache 置换过程如下表所示。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 走向 | | 0 | 4 | 8 | 2 | 0 | 6 | 8 | 6 | 4 | 8 |
| 第0组 | 块0 |  | 0 | 4 | 4 | 8 | 8 | 0 | 0 | 8 | 4 |
| 块1 | 0 | 4 | & | 8 | Q | 0 | 8\* | 8 | 4 | 8\* |
| 第 1 组 | 块2 |  |  |  |  |  | 2 | 2 | 2 | 2 | 2 |
| 块3 |  |  |  | 2 | 2 | 6 | 6 | 6\* | 6 | 6 |

注：“\_”表示当前访问块，“\*”表示本次访问命中。

|  |  |
| --- | --- |
| **注** **意** |  |
| **在不同的计算机组成原理教材中，关于组相联映射的介绍并不相同。通常是采用上题中的** **方式，也是本书及唐朔飞所编教材中的方式，但本题中采用的是蒋本珊所编教材中的方式。可** **以推断两次命题的老师应该不是同一老师，这也给考生答题带来了困扰。** | |

**30.C**

分析语句 “a[k]=a[k]+32”: 首先读取a[k] 需要访问 一 次a[k], 之后将结果赋值给 a[k] 需要

访问一次，共访问两次。第一次访问a[k]未命中，并将该字所在的主存块调入Cache 对应的块中， 对该主存块中的4个整数的两次访问中，只在访问第一 次的第一个元素时发生缺失，其他的7次 访问中全部命中，因此该程序段执行过程中访问数组 a 的 Cache 缺失率约为1/8。

**31.** A

时间局部性是指最近的未来要用到的信息，很可能是现在正在使用的信息，本题的外层循环 每次都会访问一次数组 a, 体现了时间局部性。空间局部性是指最近的未来要用到的信息，很可 能与现在正在使用的信息在存储空间上是邻近的，本题在访问数组a 的过程中是顺序访问的，体 现了空间局部性。

***32.*** *A*

Cache 数据区大小为32KB, 主存块的大小为32B, 于 是 Cache 中 共 有 1K 个 Cache 行，物理 地址中偏移量部分的长度为5bit。因为采用直接映射方式，所以1K 个 Cache 行映射到1K 个分组， 物理地址中组号部分的长度为10bit 。32bit 的主存地址除去5bit 的偏移量和10bit 的组号后，还剩 17bit 的 tag 部分。又因为 Cache 采用回写法，所以Cache 行的总位数应为32B ( 数 据 位 ) + 1 7bit (tag 位 ) + 1bit ( 脏 位 ) + 1bit ( 有 效 位 ) = 2 7 5bit。

**33.** A

Cache 采用组相联映射，主存地址结构应分为 Tag 标记、组号、块内地址三部分。主存块大 小 =Cache 块 大 小 = 6 4B=2°B, 因此块内地址占6位。Cache 数据区容量为32KB, 每 个 Cache

块大小为64B, 则 Cache 总 块 数 = 3 2KB/64B=2°, 由于采用8路组相联映射，即每8个Cache

134-2025年计算机组成原理考研复习指导

块为一个分组，因此总共被分为2°/8=2⁶组，因此，组号占6位。除了块内地址和组号，剩余的 位为 Tag 标记，占32-6-6=20位。地址结构如下所示。

|  |  |  |
| --- | --- | --- |
| **Tag标记** | 组号 | 块内地址 |
| **20位** | 6位 | 6位 |

Cache 采用8路组相联映射，因此在访问一个物理地址时，要先根据组号定位到某一分组， 然后用物理地址的高20位 (Tag 标记)与分组中8个 Cache 行的 Tag 标记做并行比较(用8个 20位“比较器”实现),若某个 Cache 行的 Tag 标记与物理地址的高20 位完全一致，则选中该 Cache 行。综上所述，在组相联映射的 Cache 中 ， “比较器”用于并行地比较分组中所有 Cache 行的 Tag 标记位与要访问物理地址的Tag 标记位，因此比较器的个数就是分组中的Cache 行数8, 比较器的位数就是Tag 标记位数20。

**二、** **综合应用题**

**01.**【解答】

回写法 (Write Back) 减少了访存次数，但存在不一致的隐患。因此若题目中出现了“较高 的安全要求”,则尽量要使用写直通法 (Write Through)。

1)采用 Write Back 策略较好，可减少访存次数。

2 ) 采 用 Write Through 策略较好，能保证数据的一致性。

**02.【**解答】

块大小为64B, 因此块内地址字段占6位；Cache 中有128个主存块，采用四路组相联，故 Cache 分为32组(128/4=32),因此组号字段占5位；标记字段为剩余的32- 5-6=21位。

数据Cache 的总位数应包括标记项的总位数和数据块的位数。每 个Cache 块对应一个标记项， 标记项中应包括标记字段、有效位和“脏”位(仅适用于回写法)。

1)主存地址中Tag 为21位，位于主存地址前部；组号 Index 为5位，位于主存地址中部； 块内地址 Offset 为6位，位于主存地址后部。

2)标记项的总位数=128×(21+1+1)=128×23=2944位，数据块位数=128×64×8=65536 位，所以数据 Cache 的总位数=2944+65536=68480 位。

**03.【**解答】

1)由于64KB/128B=512, 因此有512行。而该Cache 是四路组相联，所以512/4=128 组。

2)每行有一个标记项，因此有512 个标记项。主存字块标记长度就是标记位的长度，因为 该 Cache 有128组( =27),所以7位为组地址。而行长128B(=27),7 位为字块内地址， 因此该标记项中的标记位长度为32- 7- 7=18位。

3)LRU 替换策略要记录每个 Cache 行的生存时间，因此每个标记项有两位替换控制位。而 全写法没有“脏”位(一致性控制位),再加一个有效位即可。因此每个标记项位数是 18+2+1=21位，因此总大小为512×21=10752位。

回写式则是每个标记项加一个一致性控制位，因此为512×22=11264位。 **04.【**解答】

块大小为32B, 数组起始地址正好是一个主存块的开始，因此每8个数组元素占一个主存块； Cache 共有256B/32B=8 行，采用2-路组相联映射时， Cache 有4组。下面分析两种情况。

1)直接映射。当s=64 时：访存顺序为 a[0],a[64];a[0],a[64],…; 循环10000次。因为 a[0] 所在主存块和a[64] 所在主存块正好相差8个主存块，在直接映射方式下，除以8同余，

这两个主存块会映射到同一个 Cache 行，每次都会发生冲突，缺失率为100%。当s=63 时：访存顺序为 a[0],a[63],a[126];a[0],a[63],a[126],…; 循环10000次。因为a[63]所在 主存块和 a[126]所在主存块正好相差8个主存块，在直接映射方式下，这两个主存块会 映射到同一个 Cache 行，每次都会发生冲突，而a[0] 不会发生冲突，缺失率约为67%。

2)2-路组相联映射。当s=64 时：访存顺序为a[0],a[64];a[0],a[64],…; 循环10000次。因 为 a[0]所在主存块和 a[64]所在主存块正好相差8个主存块，在2-路组相联映射方式下，

除 以 4 同余，这两个主存块会映射到同一组，可放在同一组的不同 Cache 行中，不会发 生冲突，总缺失次数仅为2次，缺失率近似为0。当s=63 时：访存顺序为a[0],a[63],a[126]; a[0],a[63],a[126],…; 循环10000次。因为a[63] 所在主存块和a[126]所在主存块正好相 差8个主存块，这两个主存块会映射到同一组，可放在同一组的不同Cache 行中，而 a[0] 不会发生冲突，总缺失次数仅为3次，缺失率近似为0。

05. 【解答】

1)每个 Cache 行对应一个标记项，如下图所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 有效位 | 脏位 | 替换控制位 | 标记位 |

不考虑用于 Cache一致性维护和替换算法的控制位。地址总长度为28位(228=256M), 块 内地址6位(2⁶=64), Cache 块号3位(2³=8),因此Tag 的位数为28 - 6 - 3=19位，还 需使用一个有效位，因此题中数据Cache 行的结构如下图所示。

1bit

有效位

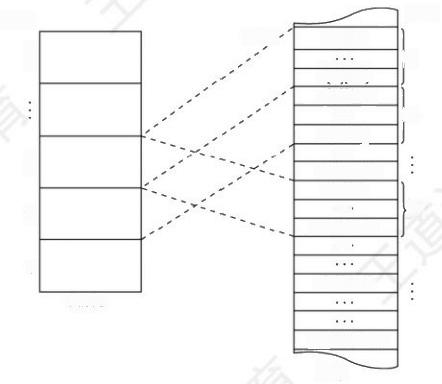
19bit

标记位

64B=512bit

该行对应的标记项 每行存储的数据

数据 Cache 共有8行，因此数据 Cache 的总容量为8×(64+20/8)B=532B。 2 ) 数 组 a 在主存的存放位置及其与Cache 之间的映射关系如下图所示。



行号

a[0][0]

0

{6

a[0][31]



a[0][255]

a[1][0]

·

6

a(1)[15]



7|

a[1][255]

Cache

a[255][255]

主存

a[0157 a[0][16]

块号

{5

320

5

21



数组按行优先方式存放，首地址为320,数组元素占4B。 Cache 行号为[(320+(0×256+31)×4)div2]mod2³=6;a[1][1] 行号为[(320+(1×256+1)×4)div2⁶]mod2³=5。

**【另解】**由1)可知主存和 Cache 的地址格式如下图所示。

a[0][31] 所在的主存块对应的 所在的主存块对应的 Cache

主存地址

136-2025年计算机组成原理考研复习指导

27 865 0

|  |  |  |
| --- | --- | --- |
| 标记 | 块号 | 块内地址 |

8 6.5 0



Cache地址 块号 块内地址

数组按行优先方式存放，首地址为320,数组元素占4B 。a[0][31] 的地址为320+31×4= 110111100g, 因此其对应的 Cache 行号为110g=6;a[1][1] 的地址为320+256×4+1×4= 1348=10101000100,因此其对应的 Cache 行号为101g=5。

3 ) 编 译 时 i,j,sum 均分配在寄存器中，所以数据访问命中率仅考虑数组a 的情况。

数 组a 的大小为256×256×4B=2¹B, 占 用 2l⁸/64=2¹²2 个主存块，按行优先存放，程序A

逐行访问数组 a, 共需访问的次数为2¹⁶次，未命中次数为2¹²次(即每个字块的第一个数 未命中),因此程序A 的命中率为(2¹⁶-2¹²)/2¹⁶ ×100%=93.75%。

**【另解】**数 组 a 按行存放，程序A 按行存取。每个字块中存放16个int 型数据，除访问的 第一个不命中外，随后的15个全都命中，访问全部字块都符合这一规律，且数组大小为 字块大小的整数倍，因此程序A 的命中率为15/16=93 .75%。

程序B 逐列访问数组a,Cache 总数据容量为64B×8=512B, 数 组a 一行的大小为1KB, 正 好是Cache 容量的2倍，可知不同行的同一列数组元素使用的是同一个Cache 单元，因此逐 列访问每个数据时，都会将之前的字块置换出，即每次访问都不会命中，命中率为0。

因为从 Cache 读数据比从主存读数据快很多，所以程序A 的执行比程序B 快得多。 **06.【** 解答】

1)CPU 的时钟周期是主频的倒数，即1/800MHz=1.25ns 。

总线的时钟周期是总线频率的倒数，即1/200MHz=5ns。

总线宽度为32位，因此总线带宽为4B×200MHz=800MB/s 或 4B/5ns =800MB/s。

2)Cache 块大小是32B, 因 此Cache 缺失时需要一个读突发传送总线事务读取一个主存块。

3)一次读突发传送总线事务包括一次地址传送、32B 数据准备和传送：用一个总线时钟周期

传输地址；之后每隔40ns/8=5ns 启动 一个存储体(各进行 一 次读操作),第 一个体准备 数 据 花 费 4 0ns, 之后这个字的传送操作与下一个字的准备操作重叠；用8个总线时钟周 期传送数据。读突发传送总线事务时间为5ns+40ns+8×5ns=85ns。

另解：首先5ns 的传送地址和命令，然后把存储体准备数据的时间视为流水线，因为总线 周期是5ns, 存储体的存储周期是40ns, 所以相当于准备数据是一个8段流水线，因此准 备8个数据的时间是40+5×7,最后再花5ns 传输最后一个数据，因为之前的7个存储体 的数据的传输时间和其下一个存储体准备数据的时间是并行的，所以共需要5+40+5×7+ 5=85ns。 也可以这样理解，将从数据准备到传输结束视为一个完整的流水线，也就是共 视为9个流水段，每个流水段的时间是5ns, 这样总共花费的时间就是5+45+7×5=85ns 。 只要是有关于流水线思想的，最关键的就是分清楚流水段，剩下的就是简单计算，不同 的算法不是关键，本质上都是一样的。

4)CPU 执 行 时 间 = Cache 命中时的指令执行时间+ Cache 未命中时的额外访存开销×缺失 率。 一条指令在 Cache 命中时的执行时间= Cache 命中时的 CPI× 时钟周期=4×1.25ns = 5ns 。一 条指令因Cache 缺失而导致的平均访存开销=平均访存次数x一次突发传送总线 事务时间=1.2×85ns =102ns。因 此 BP 的 CPU 执 行 时 间 = ( 5ns +102ns×5%)×100 = 1010ns 。100 条指令中，平均有95%的指令Cache 命中，只需要5ns; 平 均 有 5 % 的 指 令

Cache 缺失，需要5ns+102ns=107ns 。 本题说明了408真题采用先访问Cache 再访问主

存的方式。

**3.6**

**07.【** 解答】

1)主存块大小为64B=2⁶ 字节，故主存地址低6位为块内地址，Cache 组数为32KB÷(64B×8)= 64=2⁶,故主存地址中间6位为 Cache 组号，主存地址中高32- 6- 6=20位为标记，采用 8路组相联映射，故每行中的LRU 位占3位，采用直写方式，故没有修改位。

2)008000C0H=00000000100000000000000011000000B, 主存地址的低6位为块内地 址 ， 为 全 0 , 故s 位于一个主存块的开始处，占1024×4B/64B=64 个主存块；在执行程 序段的过程中，每个主存块中的64B/4B=16 个数组元素依次读、写1次，因而对每个主 存块，总是第一次访问缺失，此时会将整个主存块调入Cache, 之后每次都命中。综上， 数组s 的数据 Cache 访问缺失次数为64次。

3)00010003H=00000000000000010000000000000011B, 根据主存地址划分可知，组 索引为0,故该地址所在主存块被映射到指令Cache 的第0组；因为 Cache 初始为空，所 有 Cache 行的有效位均为0,所以 Cache 访问缺失。此时，将该主存块取出后存入指令 Cache 的第0组的任意 一 行，并将主存地址高20位(00010H) 填入该行标记字段，设置 有效位，修改 LRU 位，最后根据块内地址000011B 从该行中取出相应的内容。



**虚拟存储器**

主存和辅存共同构成了虚拟存储器，二者在硬件和系统软件的共同管理下工作。对于应用程 序员而言，虚拟存储器是透明的。虚拟存储器具有主存的速度和辅存的容量。

**3.6.1** **虚拟存储器的基本概念**

虚拟存储器将主存或辅存的地址空间统一编址，形成一个庞大的地址空间，在这个空间内， 用户可以自由编程，而不必在乎实际的主存容量和程序在主存中实际的存放位置。

用户编程允许涉及的地址称为虚地址或逻辑地址，虚地址对应的存储空间称为虚拟空间或程 序空间。实际的主存单元地址称为实地址或物理地址，实地址对应的是主存地址空间，也称实地 址 空 间。虚地址比实地址要大很多。虚拟存储器的地址空间如图3.24所示。

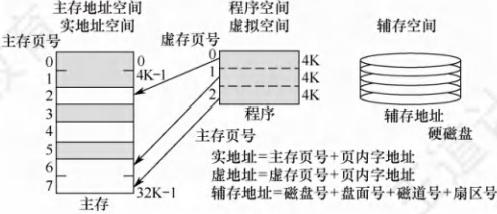


图3.24 虚拟存储器的三个地址空间

CPU 使用虚地址时，先判断这个虚地址对应的内容是否已装入主存。若已在主存中，则通过 地址变换，CPU 可直接访问主存指示的实际单元；若不在主存中，则把包含这个字的一页或一段 调入主存后再由CPU 访问。若主存已满，则采用替换算法置换主存中的交换块(页面)。

命 题 追 踪▶ **虚拟存储器只能采用回写法的原因(2016)**

虚拟存储器也采用和 Cache 类似的技术，将辅存中经常访问的数据副本存放到主存中。但是



138-2025年计算机组成原理考研复习指导

缺页(或段)而访问辅存的代价很大，提高命中率是关键，因此虚拟存储机制采用全相联映射， 每个虚页面可以存放到对应主存区域的任何一个空闲页位置。此外，当进行写操作时，不能每次 写操作都同时写回磁盘，因而，在处理一致性问题时，采用回写法。

**3.6.2** **页式虚拟存储器**

页式虚拟存储器以页为基本单位。主存空间和虚拟地址空间都被划分成相同大小的页，主存 空间中的页称为物理页、实页、页框，虚拟地址空间中的页称为虚拟页、虚页。页表记录了程序 的虚页调入主存时被安排在主存中的位置。页表一般长久地保存在内存中。

**1.** **页** **表**

图3.25是一个页表示例。有效位也称装入位，用来表示对应页面是否在主存，若为1,则表 示该虚拟页已从外存调入主存，此时页表项存放该页的物理页号；若为0,则表示没有调入主存， 此时页表项可以存放该页的磁盘地址。脏位也称修改位，用来表示页面是否被修改过，虚存机制 中采用回写策略，利用脏位可判断替换时是否需要写回磁盘。引用位也称使用位，用来配合替换 策略进行设置，例如是否实现最先调入 (FIFO 位)或最近最少用 (LRU 位)策略等。

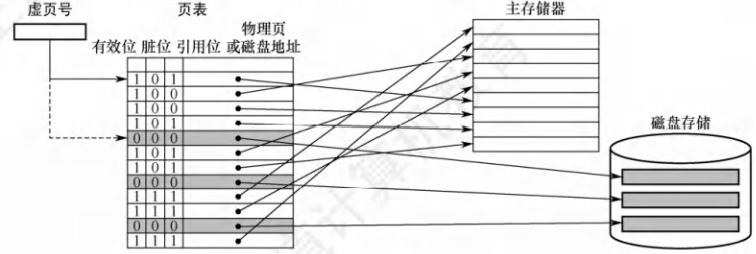


图3.25 主存中的页表示例

命题通录 ▶ **数组的分页存放、缺页异常分析及缺页处理过程(2014、2019、2023)**

以图3.25的页表为例，假设 CPU 欲访问的数据在第1页，对应的有效位为1,说明该页已 存放在主存中，再通过地址转换部件将虚拟地址转换为物理地址，然后到相应的主存实页中存取 数据。若该数据在第5页，有效位为0,则发生“缺页”异常，需调用操作系统的缺页异常处理 程序。缺页处理程序根据对应表项中的存放位置字段，将所缺页面从磁盘调入一个空闲的物理页 框。若主存中没有空闲页框，还需要选择一个页面替换。由于采用回写策略，因此换出页面时根 据脏位确定是否要写回磁盘。缺页处理过程中需要对页表进行相应的更新。

页式虚拟存储器的优点是，页面的长度固定，页表简单，调入方便。缺点是，因为程序不可 能正好是页面的整数倍，最后一页的零头将无法利用而造成浪费，并且页不是逻辑上独立的实体， 所以处理、保护和共享都不及段式虚拟存储器方便。

**2.** **地址转换**

命 题 追 踪▶ **虚拟地址结构的分析(2011、2019、2021)**

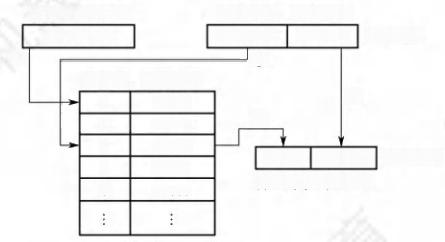
在虚拟存储系统中，指令给出的地址是虚拟地址，因此当 CPU 执行指令时，要先将虚拟地 址转换为主存物理地址，才能到主存中存取指令和数据。虚拟地址分为两个字段：高位为虚页号，

① 本节内容建议结合《操作系统考研复习指导》进行学习。

低位为页内偏移地址。物理地址也分为两个字段：高位为物理页号，低位为页内偏移地址。由于 两者的页面大小相同，因此页内偏移地址是相等的。虚拟地址到物理地址的转换是由页表实现的， 页表是一张存放在主存中的虚页号和实页号的对照表。

**虚拟地址主存物理地址(2011、2013、2018、2022)**

每个进程都有一个页表基址寄存器，存放该进程的页表首地址，据此找到对应的页表首地址 (对应①),然后根据虚拟地址高位的虚拟页号找到对应的页表项( 对应②),若装入位为1,则取 出物理页号(对应③),和虚拟地址低位的页内地址拼接，形成实际物理地址(对应④)。若装入 位为0,说明缺页，需要操作系统进行缺页处理。地址变换过程如图3 .26所示。



页表基址寄存器

页表基地址

②

装入位

④

③

0

1

11010 1010110 物理地址 物理页号页内地址

\* \* 中 中 \*

页表

物理页号 10101

\*\*\*\*\*

11010

00101

虚拟页号

00000010

页内地址

1010110

虚拟地址

1

0

1

图3.26 页式虚拟存储器的地址变换过程

**3.** **快** **表** **(TLB)**

由地址转换过程可知，访存时先访问一次主存去查页表，再访问主存才能取得数据。若缺页， 则还要进行页面替换、页面修改等，因此采用虚拟存储机制后，访问主存的次数更多了。

**命** **题** **追** **踪** ▶ TLB 的硬件实现(2018) ,TLB 和 Cache 的比较(2020)

依据程序访问的局部性原理，在一段时间内总是经常访问某些页时，若把这些页对应的页表 项存放在高速缓冲器组成的快表 (TLB) 中，则可以明显提高效率。相应地把放在主存中的页表 称为慢表 (Page) 。 在地址转换时，首先查找快表，若命中，则无须访问主存中的页表。

命 题 追 踪▶ **TLB** **映射方式、地址划分与标记字段：与** **Cache** **相同(2016、2021)**

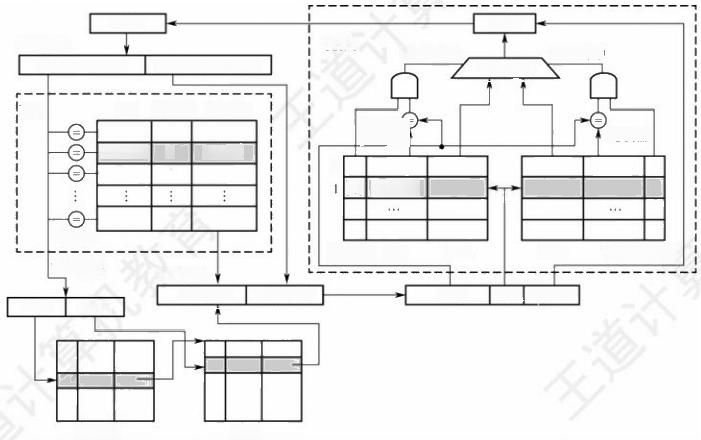
快表用 SRAM 实现，其工作原理类似于 Cache, 通常采用全相联或组相联映射方式。TLB 表 项由页表表项内容和 TLB 标记组成。全相联映射下， TLB 标记就是对应页表项的虚拟页号；组相 联方式下， TLB 标记则是对应虚拟页号的高位部分，而虚拟页号的低位部分作为TLB 组的组号。

**4.** **具** **有TLB 和** **Cache 的多级存储系统**

图 3.27是 一个具有 TLB 和 Cache 的多级存储系统，其中 Cache 采用二路组相联方式。CPU 给出 一 个32 位的虚拟地址， TLB 采用全相联方式，每一项都有一个比较器，查找时将虚页号与 每个 TLB 标记字段同时进行比较，若有某一项相等且对应有效位为1,则 TLB 命中，此时可直 接通过TLB 进行地址转换；若未命中，则TLB 缺失，需要访问主存去查页表。图中所示的是两 级页表方式，虚页号被分成页目录索引和页表索引两部分，由这两部分得到对应的页表项，从而 进行地址转换，并将相应表项调入 TLB, 若 TLB 已满，则还需要采用替换策略。完成由虚拟地

址到物理地址的转换后， Cache 机构根据映射方式将物理地址划分成多个字段，然后根据映射规 则找到对应的Cache 行或组，将对应Cache 行中的标记与物理地址中的高位部分进行比较，若相 等且对应有效位为1,则Cache 命中，此时根据块内地址取出对应的字送给CPU。

1 4 0 - 2 0 2 5 年 计 算 机 组 成 原 理 考 研 复 习 指 导



CPU

虚拟地址

虚页号(20位) 页内地址(12位)

有效位

比较器

1

0

0 00400H

比较器”

TLB缺失

10位 10位

页目录

数 据

多路选择

标记 组号块内地址

Tag 有 效 位

页 号

Cache命中

Cache命中

页内地址

物理地址

实页号

有效位

比 较 器

0008CH

页表

Cache

0040H

Tag

Tag

Data

Data

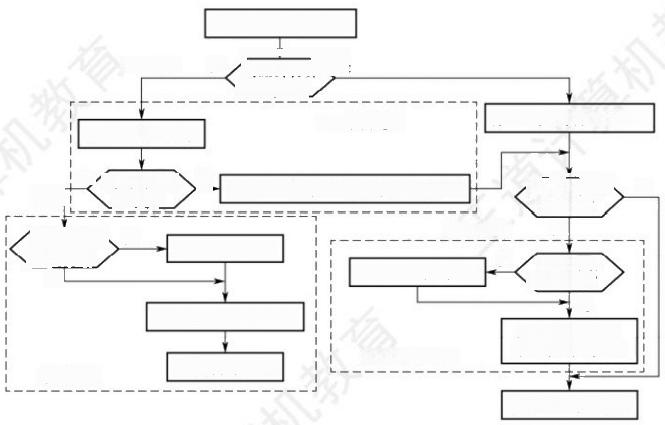
TLB

图 3 . 2 7 TLB 和 Cache 的 访 问 过 程

查找时，快表和慢表也可以同步进行，若快表中有此虚页号，则能很快地找到对应的实页号， 并且使慢表的查找作废，从而就能做到虽采用虚拟存储器但访问主存速度几乎没有下降。

命 题 追 踪 ▶ **TLB、Cache 和** **Page 缺失组合的分析(2010)**

在一个具有TLB 和 Cache 的多级存储系统中，CPU 一次访存操作可能涉及TLB、页表、Cache、 主存和磁盘的访问，访问过程如图3.28所示。可见， CPU 访存过程中存在三种缺失情况：①TLB 缺失：要访问的页面的页表项不在 TLB 中：②Cache 缺失：要访问的主存块不在 Cache 中：③Page 缺失：要访问的页面不在主存中。由于TLB 只是页表的一部分副本，因此 Page缺失时， TLB也 必然缺失。 同 理 ，Cache 也只是主存的一部分副本，页表未命中意味着信息不在主存，因此 Page 缺失时， Cache也必然缺失。这三种缺失的可能组合情况如表3.3所示。



CPU 给出虚拟地址 VA

对 应 页 表 国 是

\ 在TLB 中 ?

TLB 缺失处理

访问主存中的页表

对 应 主 存 块 是

在Cache?

否

主存中存在

Cache 中 存 \在空闲行?

是

从磁盘读出一页到主存

主存块送 Cache,

并置标记和有效位

更新页表和 TLB

访 问 Cache存取数据

否 / 访 问 页 面 在 缺页[ 主存中?

更新 TLB 并 将 VA转换为物理地址 PA

空 闲 页 框 ? 是

将 VA 转换为物理地址 PA

从 Cache 替换出 一块

TLB 缺 失 否

从主存换出一 页

Cache 缺失处理

缺页处理

Cache 缺

失 否

是

否

图 3 . 2 8 带 TLB 虚 拟 存 储 器 的CPU 访 存 过 程

**表3.3** **TLB、Page、Cache三种缺失的可能组合情况**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **TLB** | **Page** | **Cache** | **说** **明** | | |
|  | 命中 | 命中 | 命中 | TLB命中则Page一定命中， | 信息在主存， | 就可能在Cache中 |
| 2 | 命中 | 命中 | 缺失 | TLB命中则Page一定命中， | 信息在主存， | 也可能不在Cache中 |
| 3 | 缺失 | 命中 | 命中 | TLB缺失但Page可能命中， | 信息在主存， | 就可能在Cache中 |
| 4 | 缺失 | 命中 | 缺失 | TLB缺失但Page可能命中， | 信息在主存， | 也可能不在Cache中 |
| 5 | 缺失 | 缺失 | 缺失 | TLB缺失则Page也可能缺失，信息不在主存，也一定不在Cache | | |

最好的情况是第1种组合，此时无须访问主存；第2种和第3种组合都需要访问 一 次主存；

第4种组合需要访问两次主存；第5种组合发生“缺页异常”,需要访问磁盘，并且至少访问两 次主存。Cache 缺失处理由硬件完成；缺页处理由软件完成，操作系统通过“缺页异常处理程序” 来实现；而TLB 缺失既可以用硬件，又可以用软件来处理。



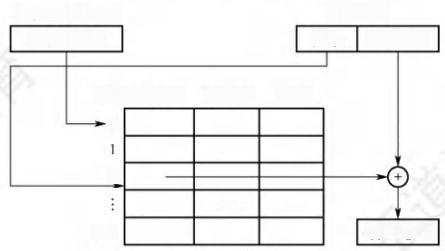
**注** **意**

在《操作系统考研复习指导》的第3章中，介绍了在同时具有TLB 和 Cache 的存储系统中 虚实地址转换的实例，读者可以结合这些内容进行学习。

**3.6.3** **段式虚拟存储器**

段式虚拟存储器中的段是按程序的逻辑结构划分的，各个段的长度因程序而异。把虚拟地址 分为两部分：段号和段内地址。虚拟地址到实地址之间的变换是由段表来实现的。段表是程序的 逻辑段和在主存中存放位置的对照表。段表的每行记录与某个段对应的段号、装入位、段起点和 段 长等信息。因为段的长度可变，所以段表中要给出各段的起始地址与段的长度。

CPU 根据虚拟地址访存时，首先根据段表基地址与段号拼接成对应的段表项，然后根据该段 表项的装入位判断该段是否已调入主存(装入位为“1”,表示该段已调入主存；装入位为“0”, 表示该段不在主存中)。已调入主存时，从段表读出该段在主存中的起始地址，与段内地址(偏 移量)相加，得到对应的主存实地址。段式虚拟存储器的地址变换过程如图3 . 29所示。



|  |  |
| --- | --- |
| 段表基址寄存器 | 虚拟地址 |
| 段表基地址 | 段号 段内地址 |

段号段首址 装入位 段长

-0

3

\*\*\*

n-1

段表(在主存中)

物理地址

图3.29 段式虚拟存储器的地址变换过程

因为段本身是程序的逻辑结构所决定的 一 些独立部分，因而分段对程序员来说是不透明的； 而分页对程序员来说是透明的，程序员编写程序时不需知道程序将如何分页。

段式虚拟存储器的优 点是 ，段的分界与程序的自然分界相对应，因而具有逻辑独立性，使得 它易于编译、管理、修改和保护，也便于多道程序的共享； 缺 点是因为段长度可变，分配空间不 便，容易在段间留下碎片，不好利用，造成浪费。

142-2025年计算机组成原理考研复习指导

**3.6.4** **段** **页** **式** **虚** **拟** **存** **储** **器**

在段页式虚拟存储器中，把程序按逻辑结构分段，每段再划分为固定大小的页，主存空间也 划分为大小相等的页，程序对主存的调入、调出仍以页为基本交换单位。每个程序对应一个段表， 每段对应一个页表，段的长度必须是页长的整数倍，段的起点必须是某一页的起点。

虚地址分为段号、段内页号、页内地址三部分。CPU 根据虚地址访存时，首先根据段号得到 段表地址；然后从段表中取出该段的页表起始地址，与虚地址段内页号合成，得到页表地址；最 后从页表中取出实页号，与页内地址拼接形成主存实地址。

段页式虚拟存储器的优 点是，兼具页式和段式虚拟存储器的优点，可以按段实现共享和保护。

缺点是在地址变换过程中需要两次查表，系统开销较大。

**3.6.5** **虚** **拟** **存** **储** **器** **与** **Cache 的** **比** **较**

虚拟存储器与 Cache 既有很多相同之处，又有很多不同之处。

**1.** **相同之处**

1)最终目标都是为了提高系统性能，两者都有容量、速度、价格的梯度。

2)都把数据划分为小信息块，并作为基本的交换单位，虚存系统的信息块更大。 3)都有地址映射、替换算法、更新策略等问题。

4)都依据局部性原理应用“快速缓存”的思想，将活跃的数据放在相对高速的部件中。

**2.** **不同之处**

1)Cache 主要解决系统速度，而虚拟存储器却是为了解决主存容量。

2)Cache 全由硬件实现，是硬件存储器，对所有程序员透明；而虚拟存储器由OS 和硬件共 同实现，是逻辑上的存储器，对系统程序员不透明，但对应用程序员透明。

▶ **Cache 缺失和缺页的处理开销对比(2016)**

3)对于不命中性能影响，因为 CPU 的速度约为 Cache 的10倍，主存的速度为硬盘的100 倍以上，因此虚拟存储器系统不命中时对系统性能影响更大。

4)CPU 与 Cache 和主存都建立了直接访问的通路，而辅存与 CPU 没有直接通路。也就是说 在 Cache 不命中时主存能和 CPU 直接通信，同时将数据调入Cache; 而虚拟存储器系统 不命中时，只能先由硬盘调入主存，而不能直接和CPU 通信。

**3.6.6** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 为使虚拟存储系统有效地发挥其预期的作用，所运行程序应具有的特性是( )。

A. 不应含有过多的I/O 操作 B. 大小不应小于实际的内存容量

C. 应具有较好的局部性 D. 顺序执行的指令不应过多

**02.** 虚拟存储管理系统的基础是程序访问的局部性原理，此理论的基本含义是( )。 A. 在程序的执行过程中，程序对主存的访问是不均匀的

B. 空间局部性

C. 时间局部性

D. 代码的顺序执行

**03.** 虚拟存储器的常用管理方式有段式、页式、段页式，对于它们在与主存交换信息时的单 位，以下表述正确的是( )。

A. 段式采用“页” B. 页式采用“块”

C. 段页式采用“段”和“页” D. 页式和段页式均仅采用“页”

**04.** 下列关于虚存的叙述中，正确的是()。

A. 对应用程序员透明，对系统程序员不透明

B. 对应用程序员不透明，对系统程序员透明

C. 对应用程序员、系统程序员都不透明

D. 对应用程序员、系统程序员都透明

**05.** 在虚拟存储器中，当程序正在执行时，由()完成地址映射。

A. 程序员 B. 编译器 C. 装入程序 D. 操作系统

**06.** 采用虚拟存储器的主要目的是()。

A. 提高主存储器的存取速度 B. 扩大主存储器的存储空间

C. 提高外存储器的存取速度 D. 扩大外存储器的存储空间

**07.** 下列关于Cache 与虚拟存储器的说法中，错误的有()。

I. 一次访存时，页表不命中，则Cache 一定也不命中

II.Cache 不命中的损失要大于页表不命中的损失

ⅢI. Cache 和 TLB 缺失后的处理都由硬件完成

IV. 虚拟存储器的实际容量可以大于主存和辅存的容量之和

A.I 和 Ⅱ B.Ⅱ 和 Ⅲ C.I 和Ⅲ和IV D.Ⅱ 和 Ⅲ 和IV

**08.** 下列有关页式存储管理的叙述中，错误的是( )。

A. 进程地址空间被划分成等长的页，内存被划分成同样大小的页框 B. 采用全相联映射，每页可以映射到任何一个空闲的页框中

C. 当从磁盘装入的信息不足一页时会产生页内碎片

D. 相对于段式存储管理，分页式更利于存储保护

**09.** 下列有关虚拟存储管理机制中地址转换的叙述，错误的是( )。

A. 地址转换是指把逻辑地址转换为物理地址

B. 通常逻辑地址的位数比物理地址的位数少

C. 地址转换过程中会发现是否“缺页’

D. 内存管理单元 (MMU) 在地址转换过程中要访问页表项

**10.** 下列有关虚拟存储管理机制的页表的叙述中，错误的是()。

A. 系统中每个进程有一个页表

B. 页表中每个表项与一个虚页对应

C. 每个页表项中都包含装入位(有效位)

D. 所有进程都可以访问页表

**11.** 下列有关缺页处理的叙述中，错误的是( )。

A. 若对应页表项中的有效位为0,则发生缺页

B. 缺页是一种外部中断，需要调用操作系统提供的中断服务程序来处理 C. 缺页处理过程中需根据页表中给出的磁盘地址去读磁盘数据

D. 缺页处理完后要重新执行发生缺页的指令

**12.** 下列关于段式虚拟存储管理的叙述中，错误的是( )。

144-2025年计算机组成原理考研复习指导

A. 段是逻辑结构上相对独立的程序块，因此段是可变长的

B. 按程序中实际的段来分配主存，所以分配后的存储块是可变长的

C. 每个段表项必须记录对应段在主存的起始位置和段的长度

D. 分段方式对低级语言程序员和编译器来说是透明的

**13.** 虚拟存储器中的页表有快表和慢表之分，下面关于页表的叙述中正确的是( )。 A. 快表与慢表都存储在主存中，但快表比慢表容量小

B. 快表采用了优化的搜索算法，因此查找速度快

C. 快表比慢表的命中率高，因此快表可以得到更多的搜索结果

D. 快表采用相联存储器件组成，按照查找内容访问，因此比慢表查找速度快 **14.【**2010统考真题】下列命令组合的一次访存过程中，不可能发生的是( )。

A.TLB 未命中， Cache 未命中，Page 未命中

B.TLB 未命中， Cache 命中， Page 命中

C.TLB 命 中 ，Cache 未命中， Page 命中

D.TLB 命中， Cache 命中， Page 未命中

**15.**【2013统考真题】某计算机主存地址空间大小为256MB, 按字节编址。虚拟地址空间大 小为4GB, 采用页式存储管理，页面大小为4KB,TLB (快表)采用全相联映射，有4 个页表项，内容如下表所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 有效位 | 标记 | 页框号 |  |
| 0 | FF180H | 0002H |  |
| 1 | 3FFF1H | 0035H |  |
| 0 | 02FF3H | 0351H |  |
| 1 | 03FFFH | 0153H |  |

则对虚拟地址03FFF180H 进行虚实地址变换的结果是()。

A.0153180H B.0035180H C.TLB 缺失 D. 缺页

**16.【**2015统考真题】假定编译器将赋值语句 “x=x+3;” 转换为指令 “add xaddr,3”, 其中 xaddr 是x 对应的存储单元地址。若执行该指令的计算机采用页式虚拟存储管理方式， 并配有相应的 TLB, 且 Cache 使用直写方式，则完成该指令功能需要访问主存的次数 至少是()。

A.0 B. 1 C.2 D.3

**17.**【2015统考真题】假定主存地址为32位，按字节编址，主存和Cache 之间采用直接映射 方式，主存块大小为4个字，每字32位，采用回写方式，则能存放4K 字数据的 Cache 的总容量的位数至少是( )。

A.146K B.147K C.148K D.158K **18.【**2019统考真题】下列关于缺页处理的叙述中，错误的是()。

A. 缺页是在地址转换时 CPU 检测到的一种异常

B. 缺页处理由操作系统提供的缺页处理程序来完成

C. 缺页处理程序根据页故障地址从外存读入所缺失的页

D. 缺页处理完成后回到发生缺页的指令的下一条指令执行

**19.**【2020统考真题】下列关于 TLB 和 Cache 的叙述中，错误的是()。

A. 命中率都与程序局部性有关 B. 缺失后都需要去访问主存

C. 缺失处理都可以由硬件实现 D. 都由 DRAM 存储器组成

82

129

130

|  |  |  |
| --- | --- | --- |
| **虚拟地址** | **页号** | **页内地址** |
|  | 15 | 0324 |
| 2 | 7 | 0128 |
| 3 | 48 | 0516 |

第 3 章 存 储 系 统 -145

20 . 【2022统考真题】某计算机主存地址为24位，采用分页虚拟存储管理方式，虚拟地址空间

大 小 为 4GB, 页 大 小 为 4KB, 按字节编址。某个进程的页表部分内容如下表所示。

虚页号 实页号(页框号) 存在位

|  |  |
| --- | --- |
| 024H | 0 |
| … |  |
| 180H | 1 |
| 018H | 1 |

当 CPU 访问虚拟地址00082840H 时，虚 - 实地址转换的结果是( )。

A. 得到主存地址024840H B. 得到主存地址180840H

C. 得到主存地址018840H D. 检测到缺页异常

**二** **、综合应用题**

**01.** 某计算机系统采用虚拟页式存储管理，某个进程的页表见下表，每项的起始编号是0,

所有的地址均按字节编址，每页大小为1024B 。分别将逻辑地址0793,1197,2099,3320, 4188,5332,转换为物理地址，写出计算过程，对不能计算的说明为什么。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **逻辑页号** | **存在位** | **引用位** | **修改位** | **页框号** |
| 0 |  |  | 0 | 4 |
| 1 | 1 | 1 | 1 | 3 |
| 2 | 0 | 0 | 0 | — |
| 3 | 1 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | — |
| 5 | 1 | 0 | 1 | 5 |

**02.** 下图表示使用快表(页表)的虚实地址转换条件，快表存放在相联存储器中，其容量为 8个存储单元。

|  |  |
| --- | --- |
| **页号** | **该页在主存中的起始位置** |
| 32 | 42000 |
| 25 | 38000 |
| 7 | 96000 |
| 6 | 60000 |
| 4 | 40000 |
| 15 | 80000 |
| 5 | 50000 |
| 34 | 70000 |

1 ) 当 CPU 按虚拟地址1去访问主存时，主存的实地址码是多少?

2 ) 当 CPU 按虚拟地址2去访问主存时，主存的实地址码是多少?

3 ) 当 CPU 按虚拟地址3去访问主存时，主存的实地址码是多少?

**03.** 一个两级存储器系统有8个磁盘上的虚拟页面需要映像到主存中的4个页中。某程序生成 以下访存页面序列：1,0,2,2,1,7,6,7,0,1,2,0,3,0,4,5,1,5,2,4,5,6,7,6,7,2,4,2,7,3。

采用 LRU 替换策略，设初始时主存为空。

1)画出每个页号访问请求之后存放在主存中的位置。

0

1

2

3

4

5

6

7

0

1

2)计算主存的命中率。

**04.【2** 011 统考真题】某计算机存储器按字节编址，虚拟(逻辑)地址空间大小为16MB,

主存(物理)地址空间大小为1MB, 页面大小为4KB;Cache 采用直接映射方式，共8 行；主存与Cache 之间交换的块大小为32B 。 系统运行到某一 时刻时，页表的部分内容 和 Cache 的部分内容分别如下的左图和右图所示，图中页框号及标记字段的内容为十六 进制形式。

回答下列问题；

1)虚拟地址共有几位，哪几位表示虚页号?物理地址共有几位，哪几位表示页框号(物

理 页 号 ) ?

2)使用物理地址访问 Cache 时，物理地址应划分成哪几个字段?要求说明每个字段的

位数及在物理地址中的位置。

虚页号 有效位 页框号 行 号 有效位 标记

0

1

2

3

4

5

6

7

|  |  |  |
| --- | --- | --- |
| 1 | 06 |  |
| 1 | 04 |  |
|  | 15 |  |
| 1 | 02 |  |
| 0 | — |  |
| 1 | 2B |  |
| 0 |  |  |
| 1 | 32 |  |

|  |  |  |
| --- | --- | --- |
| 1 | 020 |  |
| 0 | —— |  |
|  | 01D |  |
|  | 105 |  |
|  | 064 |  |
| 1 | 14D |  |
| 0 |  |  |
|  | 27A |  |

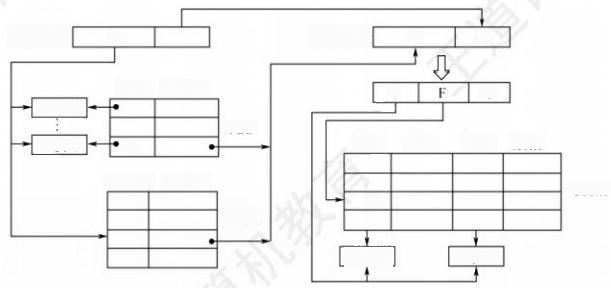
3)虚拟地址001C60H 所在的页面是否在主存中?若在主存中，则该虚拟地址对应的物 理地址是什么?访问该地址时是否 Cache 命中?要求说明理由。

4)假定为该机配置 一个四路组相联的 TLB, 共可存放8个页表项，若其当前内容(十 六进制)如下图所示，则此时虚拟地址024BACH 所在的页面是否存在主存中?要求

说明理由。

组号 有效位标记页框号 有效位标记页框号 有效位标记页框号 有效位标记页框号

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | — | — |  | 1 | 001 | 15 |  | 0 | — | — |  | 1 | 012 | 1F |
| 1 | 013 | 2D | 0 | — | —— | 1 | 008 | 7E | 0 |  |  |



**05.** 【2016 统考真题】某计算机采用页式虚拟存储管理方式，按字节编址，虚拟地址为32 位，物理地址为24位，页大小为8KB;TLB 采用全相联映射； Cache 数据区大小为 64KB, 按二路组相联方式组织，主存块大小为64B 。存储访问过程的示意图如下。

物理地址[

E G

比较器

TLB

标 记 数 据 标 记 数 据

页框号

H

页表

比较器

标 记 页 框 号

虚拟地址[

存在位

比较器

比较器

Cache

A

D

C

B

第 3 章 存 储 系 统 147

回 答 下 列 问 题 ：

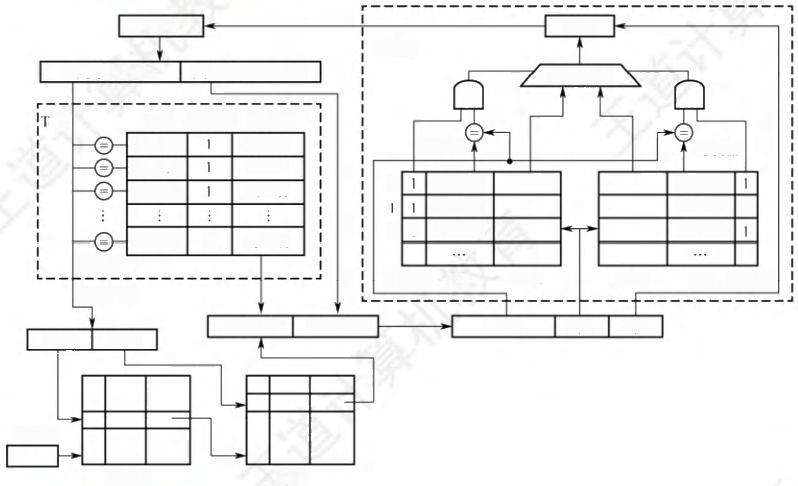
1)图中字段 A~G 的位数各是多少? TLB 标记字段B 中存放的是什么信息?

2)将块号为4099的主存块装入Cache 时，所映射的 Cache 组号是多少?对应的H 字段 内容是什么?

3 ) 是 Cache 缺失处理的时间开销大还是缺页处理的时间开销大?为什么?

4)为什么 Cache 可以采用直写策略，而修改页面内容时总是采用回写策略?

**06.【** 2018统考真题】某计算机采用页式虚拟存储管理方式，按字节编址。CPU 进行存储访 问的过程如下图所示。根据该图回答下列问题。



CPU

Cache

虚页号(20位) 页内地址(12位) 多路选择

实页号

F800H

0040H

0 F8000H F2840H

0160H

008C0H 23008H 0

2 0 00400H 01600H

04320H

比较器

有效位 Tag Data Data Tag 有 效 位

TLB 缺 失

实 页 号 页内地址

10位

PDBR

页目录 页 表

LB Tag

00324H

0008CH

8C040H

比 较 器 比 较 器

3 位 5 位

有效位

2 0 位

数 据

10位

0E80H

0

1)主存物理地址占多少位?

2)TLB 采用什么映射方式? TLB 是 用 SRAM 还是用 DRAM 实现?

3)Cache 采用什么映射方式?若 Cache 采 用 LRU 替换算法和回写策略，则 Cache 每 行中除数据 (Data) 、Tag 和有效位外，还应有哪些附加位? Cache 总容量是多少? Cache 中有效位的作用是什么?

4 ) 若 CPU 给出的虚拟地址为0008 C040H, 则对应的物理地址是多少?是否在 Cache 中命中?说明理由。若CPU 给出的虚拟地址为0007 C260H, 则该地址所在主存块映 射到的 Cache 组号是多少?

**07.** 【2021统考真题】假设计算机M 的主存地址为24位，按字节编址；采用分页存储管理 方式，虚拟地址为30位，页大小为4KB;TLB 采用二路组相联方式和LRU 替换策略， 共8组。请回答下列问题。

1)虚拟地址中哪几位表示虚页号?哪几位表示页内地址?

2 ) 已 知 访 问TLB 时虚页号高位部分用作 TLB 标记，低位部分用作 TLB 组 号 ，M 的 虚

拟地址中哪几位是TLB 标 记 ? 哪 几 位 是TLB 组 号 ?

3 ) 假 设TLB 初始时为空，访问的虚页号依次为10,12,16,7,26,4,12和20,在此过程

中 ， 哪 一 个 虚 页 号 对 应 的TLB 表 项 被 替 换 ? 说 明 理 由。

int a[24][64];

148—2025年计算机组成原理考研复习指导

4 ) 若 将 M 中的虚拟地址位数增加到32位，则TLB 表项的位数增加几位?

**08.【** 2023统考真题】已知计算机 M 的字长为32位，按字节编址，采用请求调页策略的虚 拟存储管理方式，虚拟地址为32位，页大小为4KB; 数 据 Cache 采用4路组相联映射 方式，数据区大小为8KB, 主存块大小为32B 。现 有C 语言程序段如下：

for(i=0;i<24;i++ )

for( j =0;j<64;j++)a[i][j]=10;

已知二维数组a 按行优先存放，在虚拟地址空间中分配的起始地址为00422000H,sizeoftint)=4, 假定在 M 上执行上述程序段之前数组 a 不在主存，且在该程序段执行过程中不会发生页 面置换。请回答下列问题：

1)数组 a 分布在几个页面中?对于数组a 的访问，会发生几次缺页异常?页故障地址 各是什么?

2)不考虑变量i 和 j, 该程序段的数据访问是否具有时间局部性?为什么?

3 ) 计 算 机 M 的虚拟地址 (A31~A0) 中哪几位用作块内地址?哪几位用作 Cache 组

号 ? a[1][0] 的虚拟地址是多少?其所在主存块对应的 Cache 组号是多少?

4)数组 a 占用多少主存块?假设上述程序段执行过程中数组a 的访问不会和其他数据 发生 Cache 访问冲突，则数组 a 的 Cache 命中率是多少?若将循环中i 和 j 的次序按 如下方式调换：



for(++)a[i][j]=10;

则数组a 的 Cache 命中率又是多少?

**3.6.7** **答案与解析**

**一** **、单项选择题**

**01.C**

虚拟存储系统利用的是局部性原理，程序应当具有较好的局部性，C 正确。而含有输入、输 出操作产生中断，与虚存无关， A 错误。大小较小但可以多个程序并发执行，也可以发挥虚存的 作 用 ，B 错误。。顺序执行的指令应当占较大比重为宜，这样可增强程序的局部性，D 错误。

**02.A**

局部性原理的含义是在一个程序的执行过程中，其大部分情况下是顺序执行的，某条指令或数 据使用后，在最近一段时间内有较大的可能再次被访问(时间局部性);某条指令或数据使用后， 其邻近的指令或数据可能在近期被使用(空间局部性)。在虚拟存储管理系统中，程序只能访问主 存获得指令和数据， A 正确。选项B 、C 、D 均是局部性原理的一个方面而已。

**03.D**

页式虚拟存储方式对程序分页，采用页进行交互；段页式则先按照逻辑分段，然后分页，以 页为单位和主存交互， D 正确。

**04.A**

虚存需要通过对操作系统实现地址映射，因此对操作系统的设计者即系统程序员是不透明 的。而应用程序员写的程序所使用的是逻辑地址(虚地址),因此对其是透明的。

**05.D**

虚拟存储器中，地址映射由操作系统来完成，但需要一部分硬件基础的支持，如快表、地址

映射系统等。

**06.B**

引入虚拟存储器的目的是为了解决内存容量不够大的问题。

07.D

页表不命中，表示该页面没有调入主存，而Cache 是页面的副本，因此 Cache 一 定也不命中， I 正 确 。Cache 不命中时只需从主存读取数据，页表不命中时需要从辅存读取数据，而辅存的速度 比主存慢很多，Ⅱ错误。Cache 缺失处理由硬件完成， TLB 缺失处理既可以由硬件完成，又可以 由软件完成，Ⅲ错误。虚拟存储器的实际容量小于或等于主存和辅存的容量之和， IV 错 误 。

08.D

段的分界与程序的逻辑分界相对应，使得它易于编译、修改、保护和共享。

**09.B**

虚拟存储管理的目的是让程序员可以在一个比主存地址空间大得多的虚拟地址空间中编程， 显然逻辑地址空间比主存空间大，因此逻辑地址的位数比物理地址的位数多，B 错误。在执行程 序时，由CPU 中 的 MMU 进行逻辑地址到物理地址的转换。在转换过程中， MMU 需要查找对应 的页表项，根据页表项中的装入(有效)位是否为1来确定是否发生缺页。

10.D

选项A 、B 和 C 都正确。页表中的每个表项反映的是对应虚拟页面的位置和使用等信息，通 常只能由操作系统和硬件进行访问，虚拟存储管理机制对用户进程来说是透明的， D 错误。

11.B

缺页是CPU 在执行指令过程中进行取指令或读/写数据时发生的一种故障，属于内部异常。

12.D

选项A 、B 和 C 都正确。分段方式对低级语言程序员和编译器来说是不透明的，因为低级语

言程序员需要使用段号来编程，编译器需要使用段号来链接， D 错误。

13.D

快表采用高速相联存储器，它的速度快来源于硬件本身，而不是依赖搜索算法来查找的；慢 表存储在内存中，通常是依赖于查找算法，故A 和 B 错误。快表与慢表的命中率没有必然联系， 快表仅是慢表的一个部分拷贝，不能够得到比慢表更多的结果， C 错误。

14.D

Cache 的内容是主存的一部分副本， TLB 的内容是Page (页表)的 一 部分副本。在同时具有 TLB 和 Cache 的虚拟存储系统中，CPU 发出访存命令，先查找对应的 Cache 块 。

1 ) 若 Cache 命中，则说明所需内容在 Cache 内，其所在页面必然已调入主存，因此 Page 必 然命中，但 TLB 不一定命中。

2 ) 若 Cache 未命中，则并不能说明所需内容未调入主存，和 TLB 、Page 命中与否没有联系。 但 若TLB 命中， Page 也必然命中；而当Page 命 中 ，TLB 则未必命中，因此D 不可能发生。

15.A

按字节编址，页面大小为4KB, 页内地址共12位。地址空间大小为4GB, 虚拟地址共32位， 前20位为页号。虚拟地址为03FFF180H, 因此页号为03FFFH, 页内地址为180H 。查找页标记 03FFFH 所对应的页表项，页框号为0153H, 页框号与页内地址拼接即为物理地址0153180H。

16.B

上述指令的执行过程可划分为取数、运算和写回过程，取数时读取 xaddr 可能不需要访问主 存而直接访问Cache, 而直写方式需要把数据同时写入Cache 和主存，因此至少访问1次。

**17.C**

直接映射的地址结构为

|  |  |  |
| --- | --- | --- |
| 主存字块标记 | Cache字块标记 | 字块内地址 |

按字节编址，块大小为4×3 2 位 = 1 6B=2B, 则“字块内地址”占4位；“能存放4K 字数据 的 Cache” 即 Cache 的存储容量为4K 字 ( 注 意 单 位 ) , 则Cache 共 有 1K=21° 个 Cache 行 ，Cache 字块标记占10位；主存字块标记占32 - 10 - 4=18位。

Cache 的总容量包括：存储容量和标记阵列容量(有效位、标记位、 一致性维护位和替换算 法控制位)。标记阵列中的有效位和标记位 一 定存在，而 一致性维护位(脏位)和替换算法控制 位的取舍标准是看词眼，题目中明确说明了采用回写法，则一定包含一致性维护位，而关于替换 算法的词眼题目中未提及，所以不予考虑。因此，每个Cache 行 标 记 项 包 含 1 8 + 1 + 1 = 2 0 位 ， 标记阵列容量为2l⁰×20 位 = 2 0K 位，存储容量为4K×32位 = 1 2 8K 位，总容量为128K+20K = 148K 位 。

**18.D**

在请求分页系统中，每当要访问的页面不在内存中时， CPU 检测到异常，便会产生缺页中断， 请求操作系统将所缺的页调入内存。缺页处理由缺页中断处理程序完成，根据发生缺页故障的地 址从外存读入所缺失的页，缺页处理完成后回到发生缺页的指令继续执行。选项 D 中描述回到发 生缺页的指令的下一条指令执行，明显错误。

19.D

Cache 由 SRAM 组成； TLB 也 由SRAM 组成。DRAM 需要不断刷新，性能偏低，不适合组 成 TLB 和 Cache 。 选 项A 、B 和 C 都是 TLB 和 Cache 的特点。

20.C

页 大 小 为 4KB=2°B, 按字节编址，故页内地址为12位。虚拟地址空间大小为4GB=2²B , 故虚拟地址共32位，其中低12位为页内地址，高20位为虚页号。题中给出的虚拟地址为0008 2840H, 虚页号为高20位即00082H (页内地址为低12位即840H),82H 对应的十进制数为130 (注意题中页表的虚页号部分末尾未写H, 所以是十进制数，故查找时要先将虚页号转换为十进 制数),查页表命中，并且存在位为1,对应页框号为018H。将查找到的页框号018H 和页内地址 840H 拼接，得到主存地址为018840H。

**二、** **综合应用题**

**01.【** 解答】

所有地址均可转换为页号和页内偏移量。地址转换时，先取出逻辑页号，然后查找页表，得 到页框号，再将页框号与页内偏移量拼接，即可获得物理地址。根据题意，计算逻辑地址的页号 和页内偏移量，拼接的物理地址如下表所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **逻辑地址** | **逻辑页号** | **页内偏移量** | **页框号** | **物理地址** |
| 0793 | 0 | 793 | 4 | 4889 |
| 1197 | 1 | 173 | 3 | 3245 |
| 2099 | 2 | 51 | — | 缺页中断 |
| 3320 | 3 | 248 | 1 | 1272 |
| 4188 | 4 | 92 |  | 缺页中断 |
| 5332 | 5 | 212 | 5 | 5332 |

注：在本题中，物理地址=页框号×1024B+ 页内偏移量，页内偏移量=逻辑地址-逻辑页号×1024B, 逻 辑页号=逻辑地址/1024B (结果向下取整)。

第3章 存储系统 151

02. 【解答】

1)虚拟地址1 的页号为15,页内地址为0324,在左表中页号15对应的主存起始位置为

80000,则主存的实地址码为0324+80000=80324。

2)按1)中的方法易知，主存的实地址码为0128+96000=96128。

3)虚拟地址3的页号为48,在左表中无对应项，因此该页面在快表(页表)中无记录。 03. 【解答】

1)LRU 替换策略是换出最近最久未使用的页面，因此每个页号访问请求之后存放在主存中 的位置如下图所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 页框号 | 虚拟页号 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 |  |  |  |  |  | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 3 | 3 | 3 | 3 |  |  |  |  | 1 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 3 |
| 3 |  |  | 2 | 2 | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 2 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 2 |  | 0 | 0 | 0 | 0 | 0 | 6 | 6 | 6 | 6 | 2 | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4 | 4 | 4 |
| 1 | 1 | 1 | 1 | 1 | 1 |  |  | 1 |  | 1 |  | 1 |  |  | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 2 | 2 | 2 | 2 | 2 |
| 命中 |  |  |  | \* | \* |  |  | 申 |  | 申 |  | \* |  | \* |  |  |  | \* |  | 申 | 申 |  |  | \* | \* |  |  | 申 | \* |  |

2)共30次访存，有13次命中，因此主存的命中率为13/30=43%。

04. 【解答】

1)存储器按字节编址，虚拟地址空间大小为16MB=2²\*B, 因此虚拟地址为24位；页面大 小为4KB=2²B, 因此高12位为虚页号。主存地址空间大小为1MB=22°B, 因此物理地

址为20位；由于页内地址为12位，因此高8位为物理页号。

2)因为 Cache 采用直接映射方式，所以物理地址各字段的划分如下：

|  |  |  |
| --- | --- | --- |
| 主存字块标记 | Cache字块标记 | 字块内地址 |

由于块大小为32B, 因此字块内地址占5位；Cache 共8行，因此Cache 字块标记占3位； 主存字块标记占20-5-3=12位。

3)虚拟地址001C60H 的前12位为虚页号，即001H, 查看001H 处的页表项，其对应的有 效位为1,因此虚拟地址001C60H 所在的页面在主存中。页表001H 处的页框号为04H, 与页内偏移(虚拟地址后12位)拼接成物理地址04C60H。物理地址04C60H=00000100

110001100000B, 主存块只能映射到Cache 的第3行(即第011B 行),由于该行的 有效位=1,标记(值为105H)≠04CH (物理地址高12位),因此未命中。

4)由于TLB 采用四路组相联，因此TLB 被分为8/4=2个组，因此虚页号中高11位为TLB 标记、最低1位为TLB 组号。虚拟地址024BACH=000000100100101110101100B, 虚 页号为000000100100B,TLB 标记为00000010010B ( 即 0 1 2H),TLB 组号为0B, 因 此该虚拟地址所对应的物理页面只能映射到TLB 的第0组。组0中存在有效位=1、标 记=012H 的项，因此访问TLB 命中，即虚拟地址024BACH 所在的页面在主存中。

05. 【解答】

1)页大小为8KB, 页内偏移地址为13位，因此A=B=32-13=19;D=13;C=24-13=

11;主存块大小为64B, 因此G=6 。 二路组相联，每组数据区容量有64B×2=128B, 共

有64KB/128B=512 组，因此F=9;E=24-G-F=24-6-9=9。

因而A=19,B=19,C=11,D=13,E=9,F=9,G=6。

TLB 中标记字段B 的内容是虚页号，表示该TLB 项对应哪个虚页的页表项。

2)块号4099=000001000000000011B, 因此所映射的Cache 组号为000000011B=3, 对

152—2025年计算机组成原理考研复习指导

应的H 字段内容为000001000B。

3)Cache 缺失带来的开销小，而处理缺页的开销大。因为缺页处理需要访问磁盘，而Cache

缺失只要访问主存。

4)因为采用直写策略时需要同时写快速存储器和慢速存储器，而写磁盘比写主存慢很多，

所以在 Cache-主存层次， Cache 可以采用直写策略，而在主存-外存 ( 磁盘)层次，修改 页面内容时总是采用回写策略。

**06.【**解答】

1)物理地址由实页号和页内地址拼接，因此其位数为16+12=28;或直接得20+3+5=28。

2)TLB 采用全相联映射，可把页表内容调入任意一块空TLB 项中， TLB 中的每项都有一个 比较器，没有映射规则，只要空闲就行。TLB 采用静态存储器 (SRAM), 读/写速度快， 但成本高，多用于容量较小的高速缓冲存储器。

3)图中可看到， Cache 中每组有两行，因此采用二路组相联映射方式。因为是二路组相联并采 用LRU 替换算法，所以每行需要1位LRU 位；因为采用回写策略，所以每行有1位修改 位(脏位),根据脏位判断数据是否被更新，若脏位为1,则需要写回内存。28位物理地址 中 Tag 字段占20位，组索引字段占3位，块内偏移地址占5位，因此 Cache 共有2³=8组， 每组2行，每行有2⁵=32B;Cache 的总容量为8×2×(20+1+1+1+32×8)=4464b=558B。

Cache 中有效位用来指出所在 Cache 行中的信息是否有效。

4)虚拟地址分为两部分：虚页号、页内地址；物理地址分为两部分：实页号、页内地址。 利用虚拟地址的虚页号部分去查找 TLB 表( 缺失时从页表调入),将实页号取出后和虚 拟地址的页内地址拼接，形成物理地址。虚页号0008CH 恰好在 TLB 表中对应实页号 0040H (有效位为1,说明存在),虚拟地址的后3位为页内地址040H, 对应的物理地址 是0040040H。

物理地址为0040040H, 其中高20位00400H 为标志字段，低5位00000B 为块内偏移量， 中间3位010B 为组号2,因此将00400H与 Cache 中的第2组两行中的标志字段同时比 较，可以看出，虽然有一个Cache 行中的标志字段与00400H 相等，但对应的有效位为0, 而另一 Cache 行的标志字段与00400H 不相等，因此访问Cache 不命中。

因为物理地址的低12位与虚拟地址的低12位相同，即为001001100000B。根据物理地 址的结构，物理地址的后八位01100000B的前三位011B 是组号，因此该地址所在的主存 映射到Cache 组号为3。

**07.【**解答】

注意：对于本题的TLB, 需要采用处理Cache 的方式求解。

1)按字节编址，页面大小为4KB=2²B, 页内地址为12位。虚拟地址中高30-12=18位

表示虚页号，虚拟地址中低12位表示页内地址。

2)TLB 采用二路组相联方式，共8=23组，用3位来标记组号。虚拟地址(或虚页号)中

高18-3=15位为TLB 标记，虚拟地址中随后3位(或虚页号中低3位) 为TLB 组号。

3)虚页号4对应的TLB 表项被替换。因为虚页号与TLB 组号的映射关系为TLB 组 号 = 虚 页

号 mod TLB组数=虚页号 mod 8, 因此，虚页号10,12,16,7,26,4,12,20 映射到的TLB 组号依次为2,4,0,7,2,4,4,4。TLB 采用二路组相联方式，从上述映射到的 TLB组号序列可 以看出，只有映射到4号组的虚页号数量大于2,相应虚页号依次是12,4,12和20。根据LRU 替换策略，当访问第20页时，虚页号4对应的TLB 表项被替换出来。

4)虚拟地址位数增加到32位时，虚页号增加了32-30=2位，使得每个TLB 表项中的标

记字段增加2位，因此，每个 TLB 表项的位数增加2位。

**08.【** 解答】

1 ) 数 组a 的起始地址为00422000H, 页 大 小 为 4KB, 所以页内偏移量占12位，数组a 共 有 24×64=1536个元素，每个int 型数据占4字节，因此数组a 共占1536×4B=6KB, 故分 布在2个相邻的页面中。页号分别为00422H 和00423H, 当访问这两个页面的第一个数 组元素的地址时，因为页面尚未调入内存，所以会发生2 次缺页异常，两个页故障地址

分别是00422000H 和00423000H。

2)若不考虑变量i 和 j, 该程序段的数据访问只涉及对数组元素的访问，由于每个数组元素

只访问一次，因此该程序段的数据访问没有时间局部性。

3)在组相联映射方式下，物理地址结构为 Tag 标 记 +Cache 组号+块内地址，主存块大小 为32B, 故块内地址占5位； Cache 数据区共有8KB÷32B=256 行，采用4路组相联，共 有 6 4 组 ， 故 Cache 组号占6位，因此虚拟地址中低5位 (A4～A0) 用作块内地址；低 11位虚拟地址中高6位(A10～A5) 用 作 Cache 组 号 。a[1][0] 的虚拟地址为00422000H+

1×64×4+0×4=00422100H。 虚拟地址为32位，页框大小为4KB, 虚拟地址的低12位

表示页内偏移量，因此物理地址的低12位和虚拟地址的低12位相同，因此 a[1][0]所在 主存块对应的 Cache 组号为001000B=8。

4 ) 数 组a 占24×64×4B÷32B=192 个主存块。每个主存块存放32B÷4B=8 个数组元素，访问 数组a 的 Cache 命中率为(8-1)/8=87.5%。8行数组元素占8×64×4B÷32B=64 个主存块， 分别映射到64个 Cache 组的某 Cache 行，数组 a 共有24行，因此每个 Cache 组中只有 24/8=3个Cache 行存放数组a 中的数据，而每个 Cache 组有4行，因而不会发生替换， 访问数组 a 的 Cache 命中率为7/8=87.5%。



**3.7** **本章小结**

本章开头提出的问题的参考答案如下。

1)存储器系统为何要分这些层次?计算机如何管理这些层次?

Cache- 主存层在存储系统中主要对 CPU 访存起加速作用，即从整体运行的效果看，CPU 访 存速度加快，接近于 Cache 的速度，而寻址空间和位价却接近于主存。主存-辅存层在存储系统 中主要起扩容作用，即从程序员的角度看，他所使用的存储器的容量和位价接近于辅存，而速度 接近于主存。因此从整个存储系统来看，就达到了速度快、容量大、位价低的效果。

主存与Cache 之间的信息调度全部由硬件自动完成。而主存与辅存的信息调度则采用虚拟存 储技术实现，即将主存与辅存的一部分通过软/硬结合的技术组成虚拟存储器，程序员可用这个比 主存实际空间(物理地址空间)大得多的虚拟地址空间(逻辑地址空间)编程，当程序运行时， 再由软/硬件自动配合完成虚拟地址空间与主存实际物理空间的转换。

2. 影 响 Cache 性能的因素有哪些?

决定 Cache 系统访存效率重要因素是命中率，它与很多因素有关。

1)命中率与映射方式有关，全相联映射方式的命中率最高，直接映射方式的命中率最低。 2)命中率与 Cache 容量有关，显然 Cache 容量越大，命中率就越高。

3)命中率还与主存块(或 Cache 行 ) 的大小有关，主存块的大小要适中。

154-2025年计算机组成原理考研复习指导

除上述因素外，系统是采用单级还是采用多级 Cache, 数 据 Cache 和 指 令 Cache 是 分 离 还 是

合在 一起，主存- 总线-Cache-CPU 之间采用什么架构等，都会影响 Cache 的总体性能。

3)虚拟存储系统的页面是设置得大 一 些好还是设置得小 一 些好?

页面大小要适中。页面太小时，平均页内剩余空间较小，可节省存储空间，但会使得页表增 大，页面太小时也不能充分利用空间局部性来提高命中率；页面太大时，可减少页表空间，但平 均页内剩余空间较大，会浪费较多存储空间，页面太大还会使页面调入/调出的时间较长。



**3.8** **常见问题和易混淆知识点**

1.Cache 行的大小和命中率之间有什么关系?

Cache 行的长度较大时，能充分利用程序访问的空间局部性，使 一 个较大的局部空间被 一 起 调 到 Cache 中，因而可以增加命中机会。但是，行长也不能太大，主要原因有两个：

1)行长大使失效损失变大。也就是说，若未命中，则需花更多时间从主存读块。 2 ) 行 长 太 大 ，Cache 项数变少，因而命中的可能性变小。

Cache 行的长度较小时，命中率会很低，但好处是存取块的代价较小。

2. 发 生 取 指 令Cache 缺失的处理过程是什么?

1)程序计数器恢复当前指令的值。

2)对主存进行读的操作。

3 ) 将 读 入 的 指 令 写 入Cache 中，更改有效位和标记位。

4)重新执行当前指令。

3.Cache 总容量与映射方式有何种关系?

Cache 总 容 量 = [ 每 个 Cache 行标记项的容量(有效位、脏位、LRU 替换位、标记位)+Cache 行长]xCache 总行数。其中，有效位和标记位是所有 Cache 所必需的；脏位只在 Cache 采 用 回 写 法时才需要设置； LRU 替换位只在 Cache 采 用LRU 替换算法时才需要设置。

有 效 位： 占 1 位 ， 用 于 说 明Cache 行中的数据是否有效。

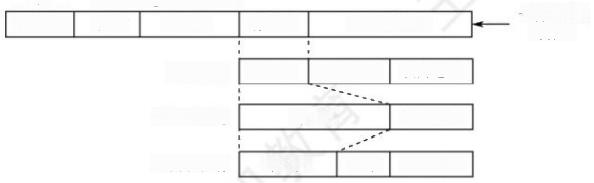
脏 位 ( 修 改 位 ) :占 1 位 ， 回 写 法 才 需 要 设 置 ， 用 以 说 明Cache 行中的数据是否被修改过。

LRU 替 换 位： 位 数 为log₂ ( 组 内 块 数 ) , 用 于LRU 替换算法中的访问计数。

标 记 位 Tag: 主存地址结构中的标记字段，其位数取决于所用的映射方式，用于匹配 Cache

行对应主存中的哪个块。

Cache 容量与映射方式的具体关系如图3 . 30所示。



标记位 Cache块数据

标记位 Cache行号 块内地址

标记位 块内地址

标记位 组号 块内地址

1位 log₂ (组内块数) 脏位 LRU位

直接映射

全相联映射

组相联映射

1位 有效位

一行Cache 的容量

图3.30 Cache 容量与映射方式的关系



|  |
| --- |
| 第 4 章  指令系统 |

**【考纲内容】**

扫一扫

(一)指令格式的基本概念

(二)指令格式

(三)寻址方式

(四)数据的对齐和大/小端存放方式”

视频讲解

( 五 )CISC 和 RISC 的基本概念

(六)高级语言程序与机器级代码之间的对应

编译器、汇编器与链接器的基本概念2;选择结构语句的机器级表示 循环结构语句的机器级表示；过程(函数)调用对应的机器级表示

**【复习提示】**

指令系统是表征一台计算机性能的重要因素。应掌握各种寻址方式的特点及有效地址的计 算，三种偏移寻址(相对寻址、基址寻址和变址寻址)的计算，CISC 与 RISC 的特点与区别。2022 年大纲新增机器级表示，机器级代码相关的题型曾在历年统考中多次考查，要能读懂代码。本章 知识点既可能出选择题，又可能结合其他章节出有关指令的综合题。指令格式、机器指令和指令 寻址方式与 CPU 指令执行过程部分紧密相关，需引起重视。

在学习本章时，请读者思考以下问题：

1)什么是指令?什么是指令系统?为什么要引入指令系统?

2)一般来说，指令分为哪些部分?每部分有什么用处?

3)对于一个指令系统来说，寻址方式多和少有什么影响?

请读者在本章的学习过程中寻找答案，本章末尾会给出参考答案。



**4.1** **指令系统**

**4.1.1** **指** **令** **集** **体** **系** **结** **构**

命 题 追 踪▶ **指令集体系结构** **(ISA)** **规定的内容(2022)**

机器指令(简称指令)是指示计算机执行某种操作的命令。 一台计算机的所有指令的集合构 成该机的指令系统，也称指令集。指令系统是指令集体系结构 (ISA) 中最核心的部分， ISA 完

① 本考点在第2章的2.3节中介绍。

② 本考点在第1章的1.2节中介绍。

零地址：

156-2025年计算机组成原理考研复习指导

整定义了软件和硬件之间的接口，是机器语言或汇编语言程序员所应熟悉的。 ISA 规定的内容主要包括；

1)指令格式，指令寻址方式，操作类型，以及每种操作对应的操作数的相应规定。 2)操作数的类型，操作数寻址方式，以及是按大端方式还是按小端方式存放。

3)程序可访问的寄存器编号、个数和位数，存储空间的大小和编址方式。 4)指令执行过程的控制方式等，包括程序计数器、条件码定义等。

ISA 规定了机器级程序的格式，机器语言或汇编语言程序员必须对机器的 ISA 非常熟悉。不 过，大多数程序员不会用汇编语言或机器语言编写程序，通常用高级语言(如C/C++/Java) 编 写 程序，这样开发效率更高，也不易出错。但是，高级语言抽象层太高，隐藏了许多机器级程序的 细节，使得高级语言程序员不能很好地利用与机器结构相关的一些优化方法来提升程序的性能。 若程序员对ISA 和底层硬件实现细节有充分的了解，则可以更好地编制高性能程序。

**4.1.2** **指** **令** **的** **基** **本** **格** **式**

一条指令就是机器语言的一个语句，它是一组有意义的二进制代码。 一条指令通常包括操作 码字段和地址码字段两部分：

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

其中，操作码指出该指令应执行什么操作以及具有何种功能。操作码是识别指令、了解指令功能 及区分操作数地址内容等的关键信息。例如，指出是算术加运算还是算术减运算，是程序转移还 是返回操作。地址码给出被操作的信息(指令或数据)的地址，包括参加运算的一个或多个操作 数的地址、运算结果的保存地址、程序的转移地址、被调用子程序的入口地址等。

指令字长是指一条指令所包含的二进制代码的位数，其取决于操作码的长度、地址码的长度 和地址码的个数。指令字长与机器字长没有固定的关系，它既可以等于机器字长，又可以大于或 小于机器字长。通常，把指令长度等于机器字长的指令称为单字长指令，指令长度等于半个机器 字长的指令称为半字长指令，指令长度等于两个机器字长的指令称为双字长指令。



**注** **意**

指令长度的不同会导致取指令时间开销的不同，单字长指令只需访存1次就能将指令完整 取出；而双字长指令则需访存2次才能完整取出，耗费2个存取周期。

在一个指令系统中，若所有指令的长度都是相等的，则称为定长指令字结构。定字长指令的 执行速度快，控制简单。若各种指令的长度随指令功能而异，则称为变长指令字结构。然而，因 为主存一般是按字节编址的，所以指令字长通常为字节的整数倍。

根据指令中操作数地址码的数目的不同，可将指令分成以下几种格式。

命 题 追 踪 ▶ **根据指令格式及相关编码条件组合成机器代码(2015)**

**1.** **零地址指令**



OP

只给出操作码 OP, 没有显式地址。这种指令有两种可能：

1)不需要操作数的指令，如空操作指令、停机指令、关中断指令等。

2)零地址的运算类指令仅用在堆栈计算机中。通常参与运算的两个操作数隐含地从栈顶和 次栈顶弹出，送到运算器进行运算，运算结果再隐含地压入堆栈。

|  |
| --- |
| OP |

一地址：

|  |
| --- |
| A₂ |

二地址：

|  |
| --- |
| A₃ (结果) |

三地址：

|  |
| --- |
| A₄ (下址) |

四地址：

**2.** **一地址指令**

|  |
| --- |
| A₁ |

这种指令也有两种常见的形态，要根据操作码的含义确定究竟是哪一种。

1)只有目的操作数的单操作数指令，按A₁ 地址读取操作数，进行 OP 操作后，结果存回原

地址。

指令含义： OP(A₁)→A

如操作码含义是加1、减1、求反、求补、移位等。

2)隐含约定目的地址的双操作数指令，按指令地址A₁ 可读取源操作数，指令可隐含约定另

一 个操作数由ACC (累加器)提供，运算结果也将存放在ACC 中 。

指令含义：(ACC)OP(A₁)→ACC

命 题 追 踪 ▶ **地址位数与寻址范围的关系(2010、2021)**

若指令字长为32位，操作码占8位，1个地址码字段占24位，则指令操作数的直接寻址范 围 为 2 2⁴= 1 6 M 。若地址码字段均为主存地址，则完成 一 条 一地址指令需要3次访存(取指令1 次，取操作数1次，存结果1次)。

**3.** **二地址指令**

|  |  |
| --- | --- |
| OP | A₁ |

指令含义： (A₁)OP(A₂)→A₁

对于常用的算术和逻辑运算指令，往往要求使用两个操作数，需分别给出目的操作数和源操 作数的地址，其中目的操作数地址还用于保存本次的运算结果。

若指令字长为32位，操作码占8位，两个地址码字段各占12位，则每个操作数的直接寻址 范 围 为 2 ¹ ² = 4K 。若地址码字段均为主存地址，则完成 一 条二地址指令需要4次访存(取指令1 次，取两个操作数2次，存结果1次)。

**4.** **三地址指令**

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

指 令 含 义 ：(A₁)OP(A₂)→A₃。

若指令字长为32位，操作码占8位，3个地址码字段各占8位，则每个操作数的直接寻址范 围为2⁸=256。若地址码字段均为主存地址，则完成 一 条三地址需要4次访问存储器(取指令1

次，取两个操作数2次，存结果1次)。

**5.** **四地址指令**

|  |  |  |  |
| --- | --- | --- | --- |
| OP | A₁ | A₂ | A₁ (结果) |

指令含义： (A₁)OP(A₂)→A₃,A₄= 下一条将要执行指令的地址。

若指令字长为32位，操作码占8位，4个地址码字段各占6位，则每个操作数的直接寻址范 围为2⁶=64。若地址码字段均为主存地址，则完成 一 条四地址指令需要4次访存(取指令1次， 取两个操作数2次，存结果1次)。

**4.1.3** **定长操作码指令格式**

命 题 追 踪 ▶ **定长操作码的指令条数(2015)**

定长操作码指令在指令字的最高位部分分配固定的若干位(定长)表示操作码。 一般n 位 操



作码字段的指令系统最大能够表示2”条指令。定长操作码对于简化计算机硬件设计，提高指令译

码和识别速度很有利。当计算机字长为32位或更长时，这是常规用法。

**4.1.4** **扩展操作码指令格式**

▶ **扩展操作码的设计与分析(2017、2021、2022)**

为了在指令字长有限的前提下仍保持比较丰富的指令种类，可采取可变长度操作码，即 全 部 指令的操作码字段的位数不固定，且分散地放在指令字的不同位置上。显然，这将增加指令译码 和分析的难度，使控制器的设计复杂化。最常见的变长操作码方法是扩展操作码，它使操作码 的长度随地址码的减少而增加，不同地址数的指令可具有不同长度的操作码，从而在满足需 要的前提下，有效地缩短指令字长。图4 . 1所示即为一种扩展操作码的安排方式。

操 作 码 的 位 数随 地 址 数 的 减 少 而 增 加

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

4 位 操 作 码

8 位 操 作 码

1 2 位 操 作 码

1 6 位 操 作 码

0000

A₂ A₂

A₁

A₁

0001

A₃ 1 5 条 三 地 址 指 令

1110 A₁ A₂ A₃

A₂

A₃ A₃

A₃

1111

1111

1111

0000

0001

1 5 条 二 地 址 指 令

A₂

A₂

1110

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1111  1111  1111 | 1111  1111  111 | 0000  0001  1110 | A₃  A₃  Aa | 1 5 条 一 地 址 指 令 |

1111 1111 1111 0000

1111 11111111 0001

1 6 条 零 地 址 指 令

1111 1111 1111 1111

图 4 . 1 扩 展 操 作 码 技 术

在图4 . 1中，指令字长为16位，其中4位为基本操作码字段 OP, 另 有 3 个 4 位 长 的 地 址 字 段 A₁ 、A₂ 和 A₃ 。4 位基本操作码若全部用于三地址指令，则有16条。图4. 1 中所示的三地址指 令为15条，1111 留作扩展操作码之用；二地址指令为15条，11111111留作扩展操作码之用； 一 地址指令为15条，111111111111留作扩展操作码之用；零地址指令为16条。

除这种安排外，还有其他多种扩展方法，如形成15条三地址指令、12条二地址指令、63条 一地址指令和16条零地址指令，共106条指令，请读者自行分析。

在设计扩展操作码指令格式时，必须注意以下两点：

1)不允许短码是长码的前缀，即短操作码不能与长操作码的前面部分的代码相同。 2)各指令的操作码 一 定不能重复。

通常情况下，对使用频率较高的指令分配较短的操作码，对使用频率较低的指令分配较长的 操作码，从而尽可能减少指令译码和分析的时间。

**4.1.5** **指令的操作类型**

设计指令系统时必须考虑应提供哪些操作类型，指令操作类型按功能可分为以下几种。

**1.** **数据传送**

传送指令通常有寄存器之间的传送 (MOV) 、 从内存单元读取数据到CPU 寄存器 (LOAD)、

从CPU 寄存器写数据到内存单元 (STORE) 、 进栈操作 (PUSH) 、 出栈操作 (POP) 等。

**2.** **算术和逻辑运算**

这类指令主要有加 (ADD) 、 减 (SUB) 、 乘 (MUL) 、 除 (DIV) 、 加 1 (INC) 、 减 1 (DEC)、

与 (AND) 、 或 (OR) 、 取 反 (NOT) 、 异 或 (XOR) 等 。

**3.** **移位操作**

移位指令主要有算术移位、逻辑移位、循环移位等。

**4.** **转移操作**

命 题 追 踪▶ **转跳指令、调用和返回指令、条件转移指令的区分(2019)**

转移指令主要有无条件转移 (JMP) 、 条件转移 (BRANCH) 、 调 用 (CALL) 、 返 回 (RET) 、 陷阱 (TRAP) 等。无条件转移指令在任何情况下都执行转移操作，而条件转移指令仅在特定条 件满足时才执行转移操作，转移条件一般是某个标志位的值，或几个标志位的组合。

调用指令和转移指令的区别：执行调用指令时必须保存下一条指令的地址(返回地址),当 子程序执行结束时，根据返回地址返回到主程序继续执行；而转移指令则不返回执行。

**5.** **输入输出操作**

这类指令用于完成CPU 与外部设备交换数据或传送控制命令及状态信息。

**4.1.6** **本节习题精选**

**一、单项选择题**

**01.** 下列关于指令集体系结构和指令系统的说法中，错误的是( )。

A. 指令集体系结构位于计算机软/硬件的交界面上

B. 指令集体系结构是指低级语言程序员所看到的概念结构和功能特性 C. 任何程序运行前都要先转换为机器语言程序

D. 指令系统和机器语言是无关的

**02.** 下列有关指令集体系结构(ISA) 的叙述中，错误的是()。

A.ISA 规定了执行每条指令时所包含的控制信号

B.ISA 规定了指令获取操作数的方式，即寻址方式

C.ISA 规定了所有指令的集合，包括指令格式和操作类型

D.ISA 规定了程序可访问的寄存器个数、存储空间大小、编址方式和大端/小端方式 **03** . 在 CPU 执行指令的过程中，指令的地址由( )给出。

A. 程序计数器 (PC) B. 指令的地址码字段

C. 操作系统 D. 程序员

**04.** 运算型指令的寻址与转移型指令的寻址的不同点在于()。

A. 前者取操作数，后者决定程序转移地址

B. 后者取操作数，前者决定程序转移地址

C. 前者是短指令，后者是长指令

D. 前者是长指令，后者是短指令

**05.** 程序控制类指令的功能是( )。

A. 进行算术运算和逻辑运算

B. 进行主存与 CPU 之间的数据传送

160-2025年计算机组成原理考研复习指导

C. 进行 CPU 和 I/O 设备之间的数据传送

D. 改变程序执行的顺序

**06.** 下列指令中不属于程序控制指令的是()。

A. 无条件转移指令 B. 条件转移指令

C. 中断隐指令 D. 循环指令

**07.** 下列指令中应用程序不准使用的指令是()。

A. 循环指令 B. 转换指令 C. 特权指令 D. 条件转移指令

**08.** 堆栈计算机中，有些堆栈零地址的运算类指令在指令格式中不给出操作数的地址，参加

的两个操作数来自()。

A. 累加器和寄存器 B. 累加器和暂存器

C. 堆栈的栈顶和次栈顶单元 D. 堆栈的栈顶单元和暂存器

**09.** 以下叙述错误的是()。

A. 为了便于取指令，指令的长度通常为存储字长的整数倍

B. 单地址指令是固定长度的指令

C. 单字长指令可加快取指令的速度

D. 单地址指令可能有一个操作数，也可能有两个操作数

**10.** 能够完成两个数的算术运算的单地址指令，地址码指明一个操作数，另一个操作数来自 ( )方式。

A. 立即寻址 B. 隐含寻址 C. 间接寻址 D. 基址寻址

**11.** 设机器字长为32位， 一个容量为16MB 的存储器， CPU 按半字寻址，其寻址单元数是

()。

A.224 B.223 C.222 D.221

**12.** 某指令系统有200条指令，对操作码采用固定长度二进制编码，最少需要用()位。 A.4 B.8 C.16 D.32

**13.** 在指令格式中，采用扩展操作码设计方案的目的是( )。

**A.** 减少指令字长度

**B.** 增加指令字长度

C. 保持指令字长度不变而增加指令的数量

D. 保持指令字长度不变而增加寻址空间

**14.** 一个计算机系统采用32位单字长指令，地址码为12位，若定义了250条二地址指令，

则还可以有()条单地址指令。

A.212 B.2l3 C.214 D.3×2l3

**15.**【2017统考真题】某计算机按字节编址，指令字长固定且只有两种指令格式，其中三 地址指令29条、二地址指令107条，每个地址字段为6位，则指令字长至少应该是

( )。

A.24 位 B.26 位 C.28 位 D.32 位

**16.**【2022统考真题】下列选项中，属于指令集体系结构 (ISA) 规定的内容是( )。

1. 指令字格式和指令类型 I.CPU 的时钟周期

**ⅢI.** 通用寄存器个数和位数 IV. 加法器的进位方式

A. 仅 I 、I B. 仅 I 、I C. 仅Ⅱ、IV D. 仅 I 、II 、IV

**17.【2** 022 统考真题】设计某指令系统时，假设采用16位定长指令字格式，操作码使用扩

展编码方式，地址码为6位，包含零地址、 一地址和二地址3种格式的指令。若二地址 指令有12条， 一地址指令有254条，则零地址指令的条数最多为( )。

A.0 B.2 C.64 D.128

**二** **、综合应用题**

**01.** 一个处理器中共有32个寄存器，使用16位立即数，其指令系统结构中共有142条指令。 在某个给定的程序中，20%的指令带有一个输入寄存器和一个输出寄存器；30%的指令带 有两个输入寄存器和一个输出寄存器；25%的指令带有一个输入寄存器、 一个输出寄存器、 一个立即数寄存器；其余25%的指令带有一个立即数输入寄存器和一个输出寄存器。

1)对于以上4种指令类型中的任意一种指令类型来说，共需要多少位?假定指令系统

结构要求所有指令长度必须是8的整数倍。

2)与使用定长指令集编码相比，当采用变长指令集编码时，该程序能够少占用多少存

储器空间?

**02.** 假设指令字长为16位，操作数的地址码为6位，指令有零地址、 一地址、二地址3种

格式

1)设操作码固定，若零地址指令有M 种， 一地址指令有N 种，则二地址指令最多有几种? 2)采用扩展操作码技术，二地址指令最多有几种?

3)采用扩展操作码技术，若二地址指令有P 条，零地址指令有Q 条，则一地址指令最 多有几种?

**03.** 在一个36位长的指令系统中，设计一个扩展操作码，使之能表示下列指令：

1)7条具有两个15位地址和一个3位地址的指令。

2)500条具有一个15位地址和一个3位地址的指令。

3)50 条无地址指令。

**4.1.7** **答** **案** **与** **解** **析**

**一、单项选择题**

**01.D**

指令集体系结构 (ISA) 完整定义了软件和硬件之间的接口，是机器语言或汇编语言程序员 所应熟悉的。指令系统是计算机硬件的语言系统，这显然和机器语言有关。

**02.A**

指令集体系结构 (ISA) 是软件和硬件之间接口的一个完整定义，包含了基本数据类型、指 令集、寄存器、寻址模式、存储体系、中断和异常处理及外部I/O 。ISA 规定了执行每条指令时所 需要的操作码、操作数、寻址方式等信息，以及指令的功能和效果。控制信号是由控制单元根据 ISA 生成的，它属于微架构层面的实现细节，而不是ISA 层面的抽象定义。

**03.** A

PC 存放当前欲执行指令的地址，而指令的地址码字段则保存操作数地址。

**04.** A

运算型指令寻址的是操作数，而转移型指令寻址的是下次欲执行的指令的地址。

**05.D**

程序控制类指令用于改变程序执行的顺序，并使程序具有测试、分析、判断和循环执行的能力。

**06.** C

162 - 2025年计算机组成原理考研复习指导

程序控制类指令主要包括无条件转移、有条件转移、子程序调用和返回指令、循环指令等。中 断隐指令是由硬件实现的，并不是指令系统中存在的指令，更不可能属于程序控制类指令。

07.C

特权指令是指仅用于操作系统或其他系统软件的指令。为确保系统与数据安全起见，这类指 令不提供给用户使用。

08.C

零地址的运算类指令也称堆栈运算指令，参与的两个操作数来自栈顶和次栈顶单元。



注 意

堆栈指令的访存次数，取决于采用的是软堆栈还是硬堆栈。若是软堆栈(堆栈区由内存实 现),则对于双目运算需要访问4次内存：取指令、取源数1、取源数2、存结果。若是硬堆栈 (堆栈区由寄存器实现),则只需在取指令时访问一次内存。



09.B

指令的地址个数与指令的长度是否固定没有必然联系，即使是单地址指令，也可能由于单地 址的寻址方式不同而导致指令长度不同。

10.B

单地址指令中只有一个地址码，在完成两个操作数的算术运算时， 一个操作数由地址码指出， 另一个操作数通常存放在累加寄存器(ACC) 中，属于隐含寻址。

11.B

16M=2²\*, 字长为32位，现在按半字(16位)寻址，相当于有8M(=22³) 个存储单元，

每个存储单元中存放16位。

12.B

因128=27⁷<200<2⁸=256,因此采用定长操作码时，至少需要8位。

13.C

扩展操作码并未改变指令的长度，而是使操作码长度随地址码的减少而增加。

14.D

地址码为12位，二地址指令的操作码长度为32-12-12=8位，已定义了250条二地址指 令，28-250=6,即可以设计出单地址指令6×2¹²=3×2¹3条。

15.A

三地址指令有29条，所以其操作码至少为5位。以5位进行计算，它剩余32-29=3种操作 码给二地址。而二地址额外多了6位给操作码，因此其数量最大达3×64=192。所以指令字长最少 为23位，因为计算机按字节编址，需要是8的倍数，所以指令字长至少应该是24位。

16.B

指令集体系结构处于软/硬件的交界面上。指令字和指令格式、通用寄存器个数和位数都与机 器指令有关，由ISA 规定。两个 CPU 可以有不同的时钟周期，但指令集可以相同；加法器的进 位方式涉及电路设计，这两项都属于计算机的硬件部分，不由ISA 规定。

17.D

地址码为6位， 一条二地址指令会占用2⁶条一地址指令的空间， 一条一地址指令会占用2⁶ 条零地址指令的空间。若全都是零地址指令，则最多有2¹⁶条，减去一地址指令和二地址指令所 占用的零地址指令空间，即2¹⁶-254×2⁶-12×2⁶ ×2⁶=(2l⁰-254-12×2⁶)×2⁶=2×2⁶=128。

另解：二地址指令有12条，则剩余16-12 =4种操作码给一地址指令， 一地址指令有254

条，剩余4×64-254=2种操作码给0地址指令，所以0地址一共有2×2⁶=128条。

**二、综合应用题**

**01.【** 解答】

1)因为有142条指令，所以至少需要8位才能确定各条指令的操作码(2⁸=256)。由于该处理器 有32个寄存器，也就是说要用5位对寄存器ID 编码，而每个立即数需要16位。因此，有：

20%的一个输入寄存器和一个输出寄存器指令需要8+5+5=18位，长度对齐到8的倍 数，便是24位。

30%的两个输入寄存器和一个输出寄存器指令需要8+5+5+5=23位，对齐到24位。

25%的一个输入寄存器、 一个输出寄存器、 一个立即数寄存器指令需要8+5+5+16=34 位，对齐到40位。

25%的一个立即数输入寄存器和一个输出寄存器指令需要8+16+5=29位，对齐到32位。 2)因为变长指令最长的长度为40位，所以定长指令编码每条指令的长度均为40位。而采

用变长编码，将各个指令长度和其概率相乘，得出平均长度为30位。所以该程序中，变 长编码比定长编码少占用25%的存储空间。

**02.** 【解答】

1)根据操作数地址码为6位，得到二地址指令中操作码的位数为16- 6- 6=4,这4位操 作码可有16种操作。由于操作码固定，因此除了零地址指令有 M 种， 一地址指令有N 种，剩下的二地址指令最多有16-M-N 种。

2)采用扩展操作码技术，操作码位数可随地址数的减少而增加。对于二地址指令，指令字 长16位，减去两个地址码共12位，剩下4位操作码，共16种编码，去掉一种编码(如 1111)用于一地址指令扩展，二地址指令最多可有15种操作。

3)采用扩展操作码技术，操作码位数可变，二地址、 一地址和零地址的操作码长度分别为4 位、10位和16位。这样，二地址指令操作码每减少一个，就可以多构成2⁶条一地址指令 操作码； 一地址指令操作码每减少一个，就可以多构成2⁶条零地址指令操作码。设一地址 指令有R 条，则一地址指令最多有(2⁴-P)×2⁶ 条，零地址指令最多有[(2⁴-P)×2⁶-R]×2⁶ 条。 根据题中给出零地址指令为Q 条，即Q=[(2⁴-P)×2⁶-RJ×2⁶, 得 R=(2⁴-P)×2⁶-

[e×2\*]。

**03.【**解答】

1)

3 15 15 3

|  |  |  |  |
| --- | --- | --- | --- |
| OP | addr1 | addr2 | addr3 |

000

l

110

2)

3 15 15 3

|  |  |  |  |
| --- | --- | --- | --- |
| OP1 | OP2 | addr1 | addr2 |

000000000000000

111

000000111110011

164-2025年计算机组成原理考研复习指导

3)

3 15 18

|  |  |  |
| --- | --- | --- |
| OP1 | OP2 |  |

000000111110100

111

000001000100101

0000….00000 (18个0)

0000…00000 (18个0)



**4.2** **指令的寻址方式**

寻址方式是指寻找指令或操作数有效地址的方式，即确定本条指令的数据地址及下一条待执 行指令的地址的方法。寻址方式分为指令寻址和数据寻址两大类。

**4.2.1** **指** **令** **寻** **址** **和** **数** **据** **寻** **址**

寻找下一条将要执行的指令地址称为指令寻址；寻找本条指令的数据地址称为数据寻址。

**1.** **指令寻址**

指令寻址方式有两种： 一种是顺序寻址方式，另一种是跳跃寻址方式。

(1)顺序寻址

通过程序计数器 PC 加1(1条指令的长度),自动形成下 一 条指令的地址。

命 题 追 踪 ▶ **P** **C** **自增大小与编址方式、指令字长的关系(2013、2014、2019、2023)**



**注** **意**

PC 自增的大小与编址方式、指令字长有关。现代计算机通常是按字节编址的，若指令字

长为16位，则PC 自增为(PC)+2; 若指令字长为32位，则PC 自增为(PC)+4。

(2)跳跃寻址

通过转移类指令实现。跳跃是指由本条指令给出下条指令地址的计算方式。而是否跳跃可能 受到状态寄存器的控制，跳跃的方式分为绝对转移(地址码直接指出转移目标地址)和相对转移 (地址码指出转移目的地址相对于当前 PC 值的偏移量),由于CPU总是根据PC的内容去主存取 指 令 的，因此转移指令执行的结果是修改 PC 值，下一条指令仍然通过PC 给出。

**2.** **数据寻址**

命 题 追 踪▶ **指令格式中各字段的位数分析(2020)**

数据寻址是指如何在指令中表示一个操作数的地址，或怎样计算出操作数的地址。数据寻址 的方式较多，为区别各种方式，通常在指令字中设置一个寻址特征字段，用来指明属于哪种寻址 方式(其位数决定了寻址方式的种类),由此可得指令的格式如下所示：

命 题 道 隙 ▶》**指令格式中寻址特征字段的作用(2023)**

|  |  |  |
| --- | --- | --- |
| 操作码 | 寻址特征 | 形式地址A |

指令中的地址码字段并不代表操作数的真实地址，这种地址称为形式地址 (A) 。 形式地址结 合寻址方式，可以计算出操作数在存储器中的真实地址，这种地址称为有效地址 (EA)。

ALU

年

-A 操作数 ACC

●若为立即寻址，则形式地址的位数决定了操作数的范围。

●若为直接寻址，则形式地址的位数决定了可寻址的范围。

· 若为寄存器寻址，则形式地址的位数决定了通用寄存器的最大数量。

●若为寄存器间接寻址，则寄存器的位数决定了可寻址的范围。



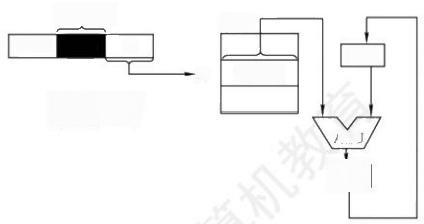
**注** **意**

(A) 表 示 地 址 为A 的 数 值 ，A 既可以是寄存器编号，又可以是内存地址。

**4.2.2** **常** **见** **的** **数** **据** **寻** **址** **方** **式**

**1.** **隐含寻址**

这种类型的指令不明显地给出操作数的地址，而是隐含操作数的地址。例如，单地址的指令 格式就隐含约定第二个操作数由累加器 (ACC) 提供，指令中只明显指出第一个操作数的地址。 因此，累加器 (ACC) 对单地址指令格式来说是隐含寻址，如图4.2所示。



寻址特征

ADD A

ACC

A 操作数

另一个操作数

隐含在ACC中

主存

图4.2 隐含寻址

优点是有利于缩短指令字长；缺点是需增加存储操作数或隐含地址的硬件。

**2.** **立** **即** **(** **数** **)** **寻** **址**

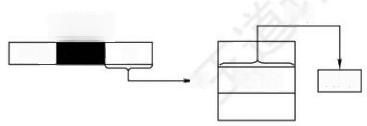
命 题 追 踪 ▶▶ **立即寻址的概念(2023)**

指令字中的地址字段指出的不是操作数的地址，而是操作数本身，也称立即数，采用补码表 示。图4.3所示为立即寻址示意图，图中#表示立即寻址特征， A 就是操作数。

优点是指令在执行阶段不访存，指令执行速度最快；缺点是A 的位数限制了立即数的范围。

**3.** **直接寻址**

指令字中的形式地址A 就是操作数的真实地址 EA, 即 EA=A, 如图4 .4所示。

主存

寻址特征

立即寻址特征

A

LDA

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

立即数

图4.3 立即寻址

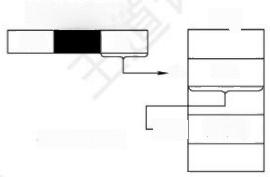
图4.4 直接寻址

优点是简单，不需要专门计算操作数的地址，指令在执行阶段仅需访存一次；缺点是A 的 位 数限制了该指令操作数的寻址范围，操作数的地址不易修改。

**4.** **间接寻址**

间 接 寻 址 是 相 对 于 直 接 寻 址 而 言 的 ， 指 令 的 地 址 字 段 给 出 的 不 是 操 作 数 的 真 正 地 址 ， 而 是 操

作数有效地址所在主存单元的地址，也就是操作数地址的地址，即EA=(A), 如图4 . 5所示。



间接寻址特征位 OP A

一次间接寻址

主存

A EA

→EA 操作数

图4.5 间接寻址

优点是可扩大寻址范围(有效地址 EA 的位数大于形式地址 A 的位数),便于编制程序(用 间接寻址可方便地完成子程序返回);缺点是指令在执行阶段要多次访存( 一 次间接寻址需2 次 访存)。由于执行速度较慢， 一般为了扩大寻址范围时，通常采用寄存器间接寻址。

**5.** **寄存器寻址**

与直接寻址的原理一样，只是把访问主存改为访问寄存器，指令的地址字段给出的是操作数 所在寄存器的编号，即 EA=R, 其操作数在由 R,所指的寄存器内，如图4.6所示。

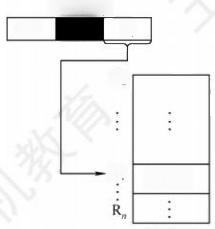
命 题 追 踪 ▶ **寄存器编号位数与寄存器数量的关系(2022)**

优点是指令在执行阶段不用访存，只访问寄存器，执行速度快；寄存器数量远小于内存单元 数，所以地址码位数较少，指令字长较短；缺点是寄存器价格昂贵， CPU 的寄存器数量有限。

**6.** **寄存器间接寻址**

命题追踪 ▶ **寄存器间接寻址的取数操作(2010)**

这种方式综合了间接寻址和寄存器寻址各自的特点，指令字中的R,所指寄存器给出的不是一 个操作数，而是操作数所在主存单元的地址，即EA=(R), 如图4 . 7所示。



寻址特征

OP R,

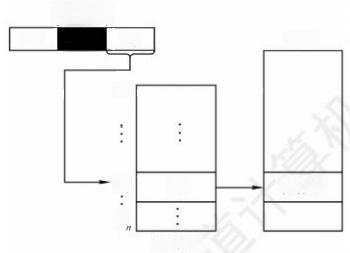
R₀|

操作数

R,

寄存器

图4.6 寄存器寻址



寻址特征

R

R₀|

EA

R,

寄存器

操作数

主存

OP

R

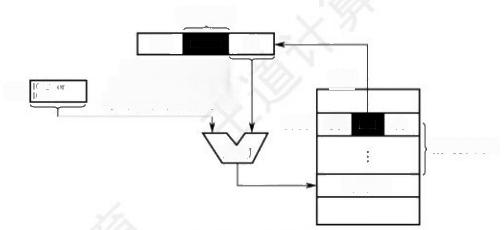
图4.7 寄存器间接寻址

相比间接寻址，这种方式既扩大了寻址范围，又减少了访存次数，在执行阶段仅需访存1次。 相比寄存器寻址，这种方式在执行阶段需要访存(因操作数在主存中)获得操作数。

**7.** **相对寻址**

命 题 追 踪▶ **相对寻址的偏移量或目标地址的计算(2009、2013、2014、2019、2023)**

相对寻址是把PC 的内容加上指令格式中的形式地址 A 而形成操作数的有效地址，即 EA = (PC)+A, 其 中A 是相对于当前PC 值的偏移量，可正可负，补码表示，如图4.8所示。



寻址特征

OP A

(PC取值后进行了

自增运算，自增长度

与当前指令长度有关)

1000 OP A

相对距离A

操作数

主存

ALU

PC

02

004

图4.8 相对寻址

在图4 . 8中，A 的位数决定操作数的寻址范围。

优点是操作数的地址不是固定的，它随 PC 值的变化而变化，且与指令地址之间总是相差一

个固定的偏移量，因此便于程序浮动。 相对寻址广泛应用于转移指令。

命 题 追 踪▶ **相对寻址转跳范围的计算(2010、2013、2014)**



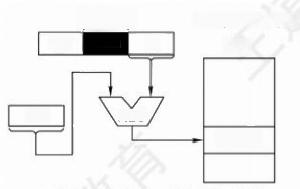
**注** **意**

对于转移指令JMPA, 若指令的地址为X, 且 占 2B, 则在取出该指令后， PC 的值会增2, 即(PC)=X+2, 这样在执行完该指令后，会自动跳转到 X+2+A 的地址继续执行。

**8.** **基址寻址**

命 题 追 踪▶ **基址寻址的** **EA** **的计算(2019)**

基址寻址是指将基址寄存器 (BR) 的内容加上指令字中的形式地址 A 而形成操作数的有效 地址，即EA=(BR)+A 。 其中基址寄存器既可采用专用寄存器，又可指定某个通用寄存器作为基 址寄存器，如图4.9所示。



BR 为基址寄存器

主存

BR ALU

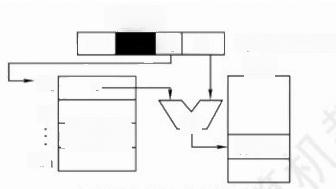
操作数

(a) 采用专用寄存器BR作为基址寄存器

寻址特征 OP

A

图4.9



寻址特征

OP R₀ A

主存

R₀[

R₁

A

通用奇存器

R

(b) 采用通用寄存器Ro作为基址寄存器

R₀ 为基址寄存器

操作数



基址寻址

基址寄存器是面向操作系统的，其内容由操作系统或管理程序确定，主要用于解决程序逻辑 空间与存储器物理空间的无关性。在程序执行过程中，基址寄存器的内容不变(作为基地址), 形式地址可变(作为偏移量)。采用通用寄存器作为基址寄存器时，可由用户决定哪个寄存器作 为基址寄存器，但其内容仍由操作系统确定。

基址寻址的优点是可以扩大寻址范围(基址寄存器的位数大于形式地址 A 的位数);用户不 必考虑自己的程序存于主存的具体位置，因此有利于多道程序设计，并可用于编制浮动程序，但 偏移量(形式地址 A) 的位数较短。

**9.** **变址寻址**

命 题 追 踪▶ **变址寻址的** **EA** **的计算(2013),先变址后间址方式的** **EA** **的计算(2016)**

变址寻址是指将变址寄存器 (IX) 的内容加上指令字中的形式地址 A 而形成操作数的有效地

168-2025 年计算机组成原理考研复习指导

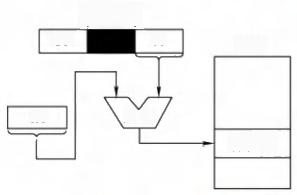
址，即EA=(IX)+A, 其 中IX 为变址寄存器(专用),也可用通用寄存器作为变址寄存器。图4 . 10

所示为采用专用寄存器IX 的变址寻址示意图。

变址寄存器是面向用户的，在程序执行过程中，变址寄存器的内容可由用户改变 ( 作 为 偏 移 量 ) , 形 式 地 址 A 不 变 ( 作 为 基 地 址 ) 。

▶ **变址寻址的适用场景(2017)**

变址寻址的优点是可扩大寻址范围(变址寄存器的位数大于形式地址A 的位数);在数组处理 过程中，可设定A 为数组的首地址，不断改变变址寄存器IX 的内容，便可很容易形成数组中任意 一个数据的地址，特别适合编制循环程序。偏移量(变址寄存器IX) 的位数足以表示整个存储空间。



寻址特征

OP A

IX ALU,

操作数

主存

命题追踪

图4.10 变址寻址

变址寻址访问数组的过程(2018)

▶

显然，变址寻址与基址寻址的有效地址形成过程极为相似。但从本质上讲，两者有较大区别。 基址寻址面向系统，主要用于为多道程序或数据分配存储空间，因此基址寄存器的内容通常由操 作系统或管理程序确定，在程序的执行过程中其值不可变，而指令字中的A 是可变的。变址寻址 立足于用户，主要用于处理数组问题，在变址寻址中，变址寄存器的内容由用户设定，在程序执 行过程中其值可变，而指令字中的A 是 不 可 变 的 。

命 题 追 踪▶ **偏移寻址的范畴(2011)**

相对寻址、基址寻址和变址寻址三种寻址方式非常类似，都将某个寄存器的内容与 一 个形式 地址相加而生成操作数的有效地址，通常把这三种寻址方式称为偏移寻址。

**10.** **堆栈寻址**

堆栈是存储器(或寄存器组)中 一 块特定的、按后进先出(LIFO) 原则管理的存储区，该存 储区中读/写单元的地址是用 一 个特定寄存器给出的，该寄存器称为堆栈指针 (SP) 。堆 栈 可 分 为 硬堆栈和软堆栈两种。寄存器堆栈也称硬堆栈，硬堆栈的成本较高，不适合做大容量的堆栈。而 从主存中划出 一 段区域来做堆栈是最合算且最常用的方法，这种堆栈称为软堆栈。

在采用堆栈结构的计算机中，大部分指令表面上都表现为无操作数指令的形式，因为操作数 地址都隐含使用了 SP 。 因此在读/写堆栈的前后都伴有自动完成对SP 的加减操作。

下面简单总结寻址方式、有效地址及访存次数(不含取本条指令的访存),见表4 . 1。

**表4.1寻址方式、有效地址及访存次数**

|  |  |  |
| --- | --- | --- |
| **寻址方式** | **有效地址** | **访存次数** |
| 立即寻址 | A即是操作数 | 0 |
| 直接寻址 | EA=A | 1 |
| 一次间接寻址 | EA=(A) | 2 |

**(续表)**

|  |  |  |
| --- | --- | --- |
| **寻址方式** | **有效地址** | **访存次数** |
| 寄存器寻址 | EA=R; | 0 |
| 寄存器间接一次寻址 | EA=(R) | 1 |
| 相对寻址 | EA=(PC)+A | 1 |
| 基址寻址 | EA=(BR)+A | 1 |
| 变址寻址 | EA=(IX)+A |  |

**4.2.3** **本** **节** **习** **题** **精** **选**

**一、单项选择题**

**01.** 指令系统中采用不同寻址方式的目的是()。

A. 提供扩展操作码的可能并降低指令译码难度

B. 可缩短指令字长，扩大寻址空间，提高编程的灵活性

C. 实现程序控制

D. 三者都正确

**02** . 采用直接转移的无条件转移指令的功能是将指令中的地址码送入()。

A. 程序计数器 (PC) B. 累加器(ACC)

C. 指令寄存器 (IR) D. 地址寄存器 (MAR)

**03.** 为了缩短指令中某个地址段的位数，有效的方法是采取()。

A. 立即寻址 B. 变址寻址 C. 间接寻址 D. 寄存器寻址

**04.** 简化地址结构的基本方法是尽量采用()。

A. 寄存器寻址 B. 隐含寻址 C. 直接寻址 D. 间接寻址

**05.** 在指令寻址的各种方式中，获取操作数最快的方式是( )。

A. 直接寻址 B. 立即寻址 C. 寄存器寻址 D. 间接寻址

**06.** 假定指令中地址码所给出的是操作数的有效地址，则该指令采用( )。

A. 直接寻址 B. 立即寻址 C. 寄存器寻址 D. 间接寻址

**07.** 设指令中的地址码为 A, 变址寄存器为 X, 程序计数器为 PC, 则变址间址寻址方式的

操作数的有效地址 EA是 ( )。

A.((PC)+A) B.((X)+A) C.(X)+(A) D.(X)+A

**08.** ( )便于处理数组问题。

A. 间接寻址 B. 变址寻址 C. 相对寻址 D. 基址寻址

**09** .堆栈寻址方式中，设A 为累加器，SP 为堆栈指示器，M 为 SP 指示的栈顶单元。若进 栈操作的动作是(A)→M,(SP)-1→SP, 则出栈操作的动作应为()。

A.(Mg)→A,(SP)+1→SP B.(SP)+1→SP,(Msp)→A

C.(SP)-1→SP,(Msp)→A D.(Msp)→A,(SP)-1→SP

**10.** 相对寻址方式中，指令所提供的相对地址实质上是一种()。

A. 立即数

B. 内存地址

C. 以本条指令在内存中首地址为基准位置的偏移量

D. 以下条指令在内存中首地址为基准位置的偏移量

**11.** 下列关于堆栈寻址的描述中，错误的是()。

A. 可以用内存来实现堆栈

B. 堆栈寻址要求计算机中设有堆栈

170-2025年计算机组成原理考研复习指导

C. 可以用硬盘来实现堆栈，称为硬堆栈

D. 可以用寄存器组来实现堆栈

**12.** 指令寻址方式有顺序和跳跃两种，采用跳跃寻址方式可以实现()。

A. 程序浮动 B. 程序的无条件浮动和条件浮动

C. 程序的无条件转移和条件转移 D. 程序的调用

**13.** 寄存器 R1 、R2 均为16位，指令 MOV R1,[R2]的功能是把内存数据传送至寄存器R1, 寻址方式为寄存器间接寻址。R2 的值为1234H, 内存单元1234H 存放数据56H, 内 存 单元1235H 存放数据78H, 采用小端方式存储。则执行指令后R1 的 值 为 ( )。

A.5678H B.7856H C.8765H D.6587H

**14.** 某计算机机的字长为16位，主存按字编址。转移指令由两个字节组成，采用相对寻址， 第一个字节为操作码字段，第二个字节为相对偏移量字段。若某转移指令所在的主存地 址为4000H, 相对偏移量字段的内容为06H, 则该转移指令执行后的 PC 值为( )。

A.4002H B.4004H C.4007H D.4008H

**15.** 某计算机的指令字长为16位，由低到高第0~7位是形式地址 D, 第8~9位为寻址特 征位X, 第10～15位为操作码。当X=00 时为直接寻址；当X=01 时使用X1 进行变 址寻址；当X=10 时使用X2 进行变址寻址；当X=11 时为相对寻址。设(PC)=1234H, (X1)=0005H,(X2)=1188H, 则指令2222H 的有效地址是()。

A.1256H B.0027H C.2222H D.11AAH

**16.** 某机器指令字长为16位，主存按字节编址，取指令时，每取一字节，PC 自动加1。当 前指令地址为2000H, 指令内容为相对寻址的无条件转移指令，指令中的形式地址为 40H。 则取指令后及指令执行后PC 的内容为( )。

A.2000H,2042H B.2002H,2040H

C.2002H,2042H D.2000H,2040H

**17**.某计算机的主存容量为4M×16位，且存储字长等于指令字长，若该机能完成97种操作， 操作码位数固定，且有直接、间接、基址、变址、相对、立即六种寻址方式，则相对寻 址的偏移量范围为( )。

A.(-32,+31) B.(-64,+63) C.(-128,+127) D.(-256 ,+255)

**18.** 对按字寻址的机器，程序计数器和指令寄存器的位数各取决于( )。

A. 机器字长，存储器的字数 B. 存储器的字数，指令字长

C. 指令字长，机器字长 D. 地址总线宽度，存储器的字数

**19.** 假设寄存器R 中的数值为200,主存地址为200和300的地址单元中存放的内容分别是

300和400,则()方式下访问到的操作数为200。

A. 直接寻址200 B. 寄存器间接寻址 (R)

C. 存储器间接寻址(200) D. 寄存器寻址 R

**20.** 假设某条指令的第一个操作数采用寄存器间接寻址方式，指令中给出的寄存器编号为8, 8号寄存器的内容为1200H, 地址为1200H 的单元中的内容为12FCH, 地址为12FCH 的单元中的内容为38D8H, 而地址为38D8H 的单元中的内容为88F9H, 则该操作数的 有效地址为( )。

A.1200H B.12FCH C.38D8H D.88F9H

**21.** 设相对寻址的转移指令占3B, 第一字节为操作码，第二、三字节为相对位移量(补码 表示),而且数据在存储器中采用以低字节为字地址的存放方式。每当 CPU 从存储器取

出一字节时，即自动完成(PC)+1→PC 。 若 PC 的当前值为240(十进制),要求转移到 290(十进制),则转移指令的第二、三字节的机器代码是( ) ; 若 PC 的当前值为240 (十进制),要求转移到200(十进制),则转移指令的第二、三字节的机器代码是( )。

A.2FH 、FFH B.D5H 、00H C.D5H 、FFH D.2FH 、00H

**22.** 某计算机按字节编址，采用大端方式，某指令的一个操作数的机器数为ABCD00FFH,

该操作数采用基址寻址方式，指令中形式地址(用补码表示)为FF0OH, 当前基址寄存 器的内容为C0000000H, 则该操作数的LSB ( 即 FFH) 存放的地址是( )。

A.C000 FF00H B. C000 FFO3H C.BFFF FFOOH D.BFFF FFO3H

**23.** 关于指令的功能及分类，下列叙述中正确的是( )。

A. 算术与逻辑运算指令，通常完成算术运算或逻辑运算，都需要两个数据 B. 移位操作指令，通常用于把指定的两个操作数左移或右移一位

C. 转移指令、子程序调用与返回指令，用于解决数据调用次序的需求 D. 特权指令，通常仅用于实现系统软件，这类指令一般不提供给用户

**24.** 【2009统考真题】某机器字长为16位，主存按字节编址，转移指令采用相对寻址，由2 字节组成，第一字节为操作码字段，第二字节为相对位移量字段。假定取指令时，每取 一字节 PC 自动加1。若某转移指令所在主存地址为2000H, 相对位移量字段的内容为 06H, 则该转移指令成功转移后的目标地址是( )。

A.2006H B.2007H C.2008H D.2009H

**25.**【2011 统考真题】偏移寻址通过将某个寄存器的内容与一个形式地址相加来生成有效地

址。下列寻址方式中，不属于偏移寻址方式的是( )。

A. 间接寻址 B. 基址寻址 C. 相对寻址 D. 变址寻址

**26.【**2011统考真题】某机器有一个标志寄存器，其中有进位/借位标志CF 、零标志ZF 、符 号标志 SF 和溢出标志 OF, 条件转移指令 bgt (无符号整数比较大于时转移)的转移条 件 是 ( ),

A.CF+OF=1 B.SF+ZF=1 C.CF+ZF=1 D.CF+SF=1

**27.** 【2013统考真题】假设变址寄存器R 的内容为1000H, 指令中的形式地址为2000H; 地

址1000H 中的内容为2000H, 地址2000H 中的内容为3000H, 地址3000H 中的内容为 4000H, 则变址寻址方式下访问到的操作数是( )。

A.1000H B.2000H C.3000H D.4000H

**28.【2**014统考真题】某计算机有16个通用寄存器，采用32位定长指令字，操作码字段(含 寻 址 方 式 位 ) 为 8 位 ，STORE 指令的源操作数和目的操作数分别采用寄存器直接寻址 和基址寻址方式。若基址寄存器可使用任意一个通用寄存器，且偏移量用补码表示，则 STORE 指令中偏移量的取值范围是( )。

A.-32768～+32767 B.-32767～+32768

C.-65536～+65535 D.-65535～+65536

**29.【**2016统考真题】某指令格式如下所示。

|  |  |  |  |
| --- | --- | --- | --- |
| OP | M | 1 | D |

其 中M 为寻址方式， I 为变址寄存器编号， D 为形式地址。若采用先变址后间址的寻址 方式，则操作数的有效地址是()。

A.I+D B.(I)+D C.((I)+D ) D.((I))+D

172—2025年计算机组成原理考研复习指导

**30.【2**017统考真题】下列寻址方式中，最适合按下标顺序访问一维数组元素的是( )。

A. 相对寻址 B. 寄存器寻址 C. 直接寻址 D. 变址寻址

**31.【**2018统考真题】按字节编址的计算机中，某 double型数组 A 的首地址为2000H, 使用变 址寻址和循环结构访问数组A, 保存数组下标的变址寄存器的初值为0,每次循环取一个数 组元素，其偏移地址为变址值乘以 sizeof(double), 取完后变址寄存器的内容自动加1。若 某次循环所取元素的地址为2100H, 则进入该次循环时变址寄存器的内容是()。

A.25 B.32 C.64 D.100

**32.【**2019统考真题】某计算机采用大端方式，按字节编址。某指令中操作数的机器数为1234 FFOOH, 该操作数采用基址寻址方式，形式地址(用补码表示)为 FF12H, 基址寄存器的 内容为 F0000000H, 则该操作数的 LSB (最低有效字节)所在的地址是()。

A.F000 FF12H B.F000 FF15H C.EFFF FF12H D.EFFF FF15H

**33.【**2020统考真题】某计算机采用16位定长指令字格式，操作码位数和寻址方式位数固 定，指令系统有48 条指令，支持直接、间接、立即、相对4种寻址方式。在单地址指

令中，直接寻址方式的可寻址范围是( )。

A.0~255 B.0~1023 C.-128～127 D.-512～511

**34.**【2023 统考真题】某运算类指令中有一个地址码为通用寄存器编号，对应通用寄存器

中存放的是操作数或操作数的地址， CPU 区分两者的依据是( )。

A. 操作数的寻址方式 B. 操作数的编码方式

C. 通用寄存器的编号 D. 通用寄存器的内容

**二、** **综合应用题**

**01.** 某机字长为16位，存储器按字编址，访问内存指令格式如下：

15 1110 87 0

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

其中， OP 为操作码，M 为寻址特征，A 为形式地址。设 PC 和 Rx 分别为程序计数器和 变址寄存器，字长为16位，问：

1)该指令能定义多少种指令?

2)下表中各种寻址方式的寻址范围为多少?

3)写出下表中各种寻址方式的有效地址 EA 的计算公式。

|  |  |  |
| --- | --- | --- |
| **寻** **址** **方** **式** | **有效地址EA的计算公式** | **寻** **址** **范** **围** |
| 直接寻址 |  |  |
| 间接寻址 |  |  |
| 变址寻址 |  |  |
| 相对寻址 |  |  |

**02.** 一条双字长的LOAD 指令存储在地址为200和201的存储位置，该指令将指定的内容装 入累加器(ACC) 中。指令的第一个字指定操作码和寻址方式，第二个字是地址部分。 主存内容示意图如下图所示。PC 值为200,RI 值为400,XR (变址寄存器)值为100。

指令的寻址方式字段可指定任何一种寻址方式。请在下列寻

址方式中，分析装入ACC 的值为多少。

1)直接寻址。

2)立即寻址。

200

201

202

300

400

500

600

702

800

3)间接寻址。

4)相对寻址。

5)变址寻址。

6)寄存器 R1 寻址。

7)寄存器R1 间接寻址。

地址 主存

|  |  |
| --- | --- |
| LOAD | MOD |
| 500 | |
|  | |
|  | |
| 450 | |
|  | |
| 700 | |
|  | |
| 800 | |
|  | |
| 900 | |
|  | |
| 325 | |
|  | |
| 300 | |

**03.** 某机的机器字长为16位，主存按字编址，指令格式如下：

15 10g 87 [0](#bookmark210)

|  |  |  |
| --- | --- | --- |
| 操作码 | X | D |

其中， D 为位移量； X 为寻址特征位。

X=00: 直接寻址。

X=01: 用变址寄存器 X1 进行变址。

X=10: 用变址寄存器 X2 进行变址。

X=11: 相对寻址。

设(PC)=1234H,(X1)=0037H,(X2)=1122H(H 代表十六位进制数),请确定下列指

令的有效地址：

①4420H ② 2244H ③ 1322H ④ 3521H ⑤ 6723H

**04.** 某计算机字长16位，标志寄存器FLAGS 中 的ZF 、SF 和 OF 分别是零标志、符号标志 和溢出标志，采用双字节字长指令字。假定 bgt (大于零转移)指令的第一个字节指明

操作码和寻址方式，第二个字节为偏移地址 Imm8, 用补码表示。指令功能是： 若 (ZF+(SF④OF)=0), 则 PC=PC+2+Imm8×2; 否 则 ，PC=PC+2。

请回答下列问题：

1)该计算机的编址单位是多少?

2)bgt 指令执行的是有符号整数比较，还是无符号整数比较?

3)偏移地址 Imm8 的含义是什么?转移目标地址的范围是什么?

**05.** 一条双字长的取数指令 (LDA) 存于存储器的200和201单元，其中第一个字为操作码 OP 和寻址特征 M, 第二个字为形式地址 A 。假设 PC 的当前值为200,变址寄存器 IX

174-2025年计算机组成原理考研复习指导

的内容为100,基址寄存器的内容为200,存储器相关单元的内容如下表所示：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 201 | 300 | 400 | 401 | 500 | 501 | 502 | 700 |
| 内容 | 300 | 400 | 700 | 501 | 600 | 700 | 900 | 401 |

下表的各列分别为寻址方式、该寻址方式下的有效地址及取数指令执行结束后累加器 (AC) 的内容，试补全下表：

|  |  |  |
| --- | --- | --- |
| 寻 址 方 式 | 有效地址(EA) | **累加器(AC)的内容** |
| 立即寻址 |  |  |
| 直接寻址 |  |  |
| 间接寻址 |  |  |
| 相对寻址 |  |  |
| 变址寻址 |  |  |
| 基址寻址 |  |  |
| 先变址后间址 |  |  |
| 先间址后变址 |  |  |

**06.** 【2010统考真题】某计算机字长为16位，主存地址空间大小为128KB, 按字编址，采 用单字长指令格式，指令各字段定义如下：

15 1211 65 0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OP | Ms | Rs | Md | Rd |

源操作数 目的操作数

转移指令采用相对寻址方式，相对偏移量用补码表示，寻址方式定义见下表。

Ms/Md 寻 址 方 式 助 记 符 含 义

000B 寄存器直接

001B 寄存器间接

010B 寄存器间接、自增

011B 相对

Rn

(Rn)

(Rn)+

D(Rn)

操作数=(Rn)

操作数=((Rn))

操作数=((Rn)),(Rn)+1→Rn

转移目标地址=(PC)+(Rn)

注：(X)表示存储器地址X或寄存器X的内容。

回答下列问题：

1)该指令系统最多可有多少条指令?该计算机最多有多少个通用寄存器?存储器地址 寄存器 (MAR) 和存储器数据寄存器 (MDR) 至少各需要多少位?

2)转移指令的目标地址范围是多少?

3)若操作码0010B 表示加法操作(助记符为add), 寄 存 器R4 和 R5 的编号分别为100B 和101B,R4 的内客为1234H,R5 的内容为5678H, 地址1234H 中的内容为5678H, 5678H 中的内容为1234H, 则 汇 编 语 句“add(R4),(R5)+” (逗号前为源操作数，逗号 后为目的操作数)对应的机器码是什么(用十六进制表示)?该指令执行后，哪些 寄存器和存储单元的内容会改变?改变后的内容是什么?

**07.** 【2013统考真题】某计算机采用16位定长指令字格式，其 CPU 中有一个标志寄存器， 其中包含进位/借位标志CF、 零 标 志ZF 和符号标志NF。 假定为该机设计了条件转移指 令，其格式如下：

15 1110987 0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 00000 | C | Z | N | OFFSET |

第4章 指令系统-175

其中，00000为操作码OP;C 、Z 和 N 分别为CF 、ZF 和 NF 的对应检测位，某检测位 为1时表示需检测对应标志，需检测的标志位中只要有一个为1就转移，否则不转移。

例如，若C=1,Z=0,N= 1, 则需检测CF 和 NF 的值，当CF=1 或 NF=1 时发生转

移； OFFSET 是相对偏移量，用补码表示。转移执行时，转移目标地址为(PC)+2 +

2×OFFSET; 顺序执行时，下一条指令地址为(PC)+2。 请回答下列问题：

1)该计算机存储器是按字节编址还是按字编址?该条件转移指令向后(反向)最多可 跳转多少条指令?

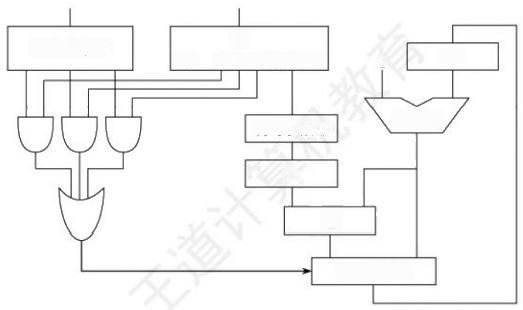
2)某条件转移指令的地址为200CH, 指令内容如下图所示，若该指令执行时CF=0,

ZF=0,NF=1, 则该指令执行后PC 的值是多少?若该指令执行时CF=1,ZF=0,

NF=0, 则该指令执行后 PC 的值又是多少?请给出计算过程。

|  |  |  |  |
| --- | --- | --- | --- |
| 0ooo0 | D |  | oo |

3)实现“无符号数比较小于或等于时转移”功能的指令中，C 、Z 和 N 应各是什么? 4)以下是该指令对应的数据通路示意图，要求给出图中部件①~③的名称或功能说明。



标志寄存器

OP

**符号扩展器**

②

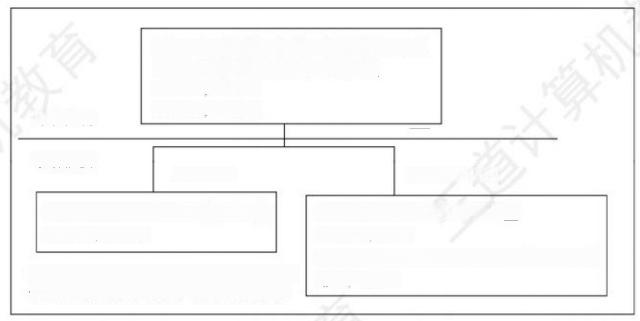
③

**多路选择器**

C Z N OFFSET 2 PC

加法器

①



**08.** 【2015统考真题】题中描述的计算机，某部分指令执行过程的控制信号如下所示。

PCout=1,MARin=1,Tin=1,MEMop=read

MUXop=①,ALUop=add,SRop=②

SRout=1,PCin=1

MDRout,IRin=1

执行阶段

sub R0.R2.(RI)

Rlout=1,Tin=1,ALUop=③SRop=④

SRout=1,R2in=1

**注：值为0的寄存器输入/输出控制信号以及**

值为任意的其他控制信号均未在图中标出。

该机指令格式如下图所示，支持寄存器直接和寄存器间接两种寻址方式，寻址方式位分 别为0和1,通用寄存器 R0～R3 的编号分别为0,1,2和3。

Rlout=1,MARin=1,MEMop=⑤

R2out=1,Tin=1

MDRout=1,MUXop=1,ALUop=⑥,SRop=⑦ ⑧=1,R0in=1

**取指阶段**

sh1 R2.R1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000000 | rs | rt | rd | opl |
| op2 | rs | rt | imm | |
| op3 | target | | | |

176-2025年计算机组成原理考研复习指导

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 指令操作码 目的操作数 源操作数1 源操作数2   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | OP | Md | Rd | Msl | Rsl | Ms2 | Rs2 |   **其中：Md、Ms1、Ms2为寻址方式位，Rd、Rs1、Rs2为奇存器编号。**  三地址指令： **源操作数1** **OP源操作数2→** **目的操作数地址**  二地址指令(末3位均为0): **OP源操作数1** **→** **目的操作数地址**  单地址指令(末6位均为0): **OP目的操作数→** **目的操作数地址** |

回答下列问题：

1)该机的指令系统最多可定义多少条指令?

2 ) 假 定 inc 、shl 和 sub 指令的操作码分别为01H 、02H 和 0 3H, 则以下指令对应的机器 代码各是什么?

① inc R1 (R1)+1→R

② shl R2,R1 ; **(**R1)<<1→R2

③ sub R3,(RI),R2 ((R1))-(R2)→R3

3)假设寄存器X 的输入和输出控制信号分别为Xin 和 Xout, 其值为1表示有效，为0 表 示 无 效 ( 如PCout=1 表 示PC 内容送总线);存储器控制信号为MEMop, 用于控 制存储器的读 (read) 和 写 (write) 操作。写出本题第一 幅图中标号①~⑧处的控制 信号或控制信号的取值。

4 ) 指 令 “sub R1,R3,(R2)” 和 “inc R1” 的执行阶段至少各需要多少个时钟周期?

**09.【** 2021 统考真题】假定计算机M 字长为16位，按字节编址，连接 CPU 和主存的系统总线

中地址线为20位、数据线为8位，采用16位定长指令字，指令格式及说明如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 格式 | 6位 | 2位 2 位 2 位 | 4位 | 指令功能或指令类型说明 |
| R型  1型 |  |  |  | R[rd]—R[rs]oplR[n]  含ALU运算、条件转移和访存 操作3类指令 |
| J型 |  |  |  | PC的低10位—target |

其中， opl~op3 为操作码， rs,rt 和 rd 为通用寄存器编号，R[r] 表示寄存器r 的 内 容 ，imm

为 立 即 数 ，target为转移目标的形式地址。请回答下列问题。

1)ALU 的宽度是多少位?可寻址主存空间大小为多少字节?指令寄存器、主存地址寄

存 器 (MAR) 和主存数据寄存器 (MDR) 分别应有多少位?

2)R 型格式最多可定义多少种操作? I 型 和J 型格式总共最多可定义多少种操作?通用 寄存器最多有多少个?

3 ) 假 定opl 为0010和0011时，分别表示有符号整数减法和有符号整数乘法指令，则指 令 0 1B2H 的功能是什么(参考上述指令功能说明的格式进行描述)?若1,2,3号通

用寄存器当前内容分别为B052H,0008H,0020H, 则分别执行指令01B2H 和 0 1B3H

后，3号通用寄存器内容各是什么?各自结果是否溢出?

4 ) 若 采 用I 型格式的访存指令中 imm (偏移量)为有符号整数，则地址计算时应对 imm

进行零扩展还是符号扩展?

5)无条件转移指令可以采用上述哪种指令格式?

**4.2.4** **答案与解析**

**一** **、单项选择题**

**01.B**

采用不同寻址方式的目的是为了缩短指令字长，扩大寻址空间，提高编程的灵活性，但这也 提高了指令译码的复杂度。程序控制是靠转移指令而非寻址方式实现的。

**02.A**

转移指令有条件/无条件、直接间接、相对/绝对三种属性。条件转移是指需要先判断条件是 否成立，才决定是否转移；无条件转移是指不用判断条件就可以转移，典型的是函数调用和返回。 直接转移是指转移目标地址直接放在指令中，执行时直接将地址码送入 PC; 间接转移是指转移 目标地址存放在寄存器或内存单元中。相对转移是指转移目标地址为当前 PC 值加上偏移量，偏 移量一般在指令中；绝对转移是指转移目标地址直接由指令或寄存器给出。

**03.D**

CPU 中寄存器的数量都不会太多，用很短的编码就可以指定寄存器，寄存器寻址需要的地址 段位数为[log, (通用寄存器个数)],因此能有效地缩短地址段的位数。立即寻址，操作数直接保存 在指令中，若地址段位数太小，则操作数表示的范围会很小；变址寻址， EA= 变址寄存器IX 的 内 容 + 形 式 地 址 A,A 与主存寻址空间有关；间接寻址中存放的仍然是主存地址。

**04.B**

隐含寻址不明显给出操作数地址，而在指令中隐含操作数的地址，因此可以简化地址结构。

**05.B**

立即寻址最快，指令直接给出操作数；寄存器寻址次之，只需访问一次寄存器；直接寻址再 次之，访问一次内存；间接寻址最慢，要访问内存两次或以上。

**06.A**

指令字中的形式地址为操作数的有效地址，这种方式为直接寻址。

**07.B**

变址寻址的有效地址是(X)+A, 再进行间址，即把(X)+A 中取出的内容作为真实地址 EA,

即EA=((X)+A).

寄存器中的内容和指令地址码相加得到的是操作数的地址码。

**08.B**

变址寻址便于处理数组问题。基址寻址与变址寻址的区别见下表。

|  |  |  |
| --- | --- | --- |
|  | 基 址 寻 址 | 变 址 寻 址 |
| 有效地址 | EA=(BR)+A | EA=(IX)+A |
| 访存次数 | 1 | 1 |
| 寄存器内容 | 由操作系统或管理程序确定 | 由用户设定 |
| 程序执行过程中值可变否 | 不可变 | 可变 |
| 特点 | 有利于多道程序设计和编制浮动程序 | 有利于处理数组问题和编制循环程序 |

**09.B**

进、出堆栈时对栈顶指针的操作顺序是不同的，进栈时是先压入数据(A)→Msp, 后修改指针 (SP)-1→SP, 说明栈指针是指向栈顶的空单元的，所以出栈时要先修改指针(SP)+1→SP, 然后 才能弹出数据(Msp)→A。

178-2025年计算机组成原理考研复习指导

**10.D**

相对寻址中，有效地址 EA=(PC)+A(A 为形式地址),执行本条指令时， PC 已完成加1操

作 ，PC 中保存的是下一条指令的地址，因此以下一条指令的地址为基准位置的偏移量。

11.C

堆栈是主存(或寄存器)中一块特定的、按后进先出原则管理的存储区，寄存器堆栈称为硬 堆栈，主存中划分出来的区域作为堆栈是最合算且最常用的方法，这种堆栈称为软堆栈。

**12.** C

跳跃寻址通过转移类指令(如相对寻址)来实现，可用来实现程序的条件或无条件转移。

**13.B**

寄存器R2 中的值是1234H, 内存单元1234H 中的值是56H,1235H 中的值是78H, 由于采 用小端方式，因此实际存储的数据为7856H, 取出后存放到R1, 因此 R1 的值为7856H。

**14.** C

主存按字编址，指令字长为1个字(2字节),因此取出该指令后，PC 自动加1,相对偏移 量 为 0 6H, 所以该转移指令执行后的PC 值为4000H+06H+1H=4007H。

15.D

将指令2222H 展开成二进制为0010001000100010B, 因此寻址特征位X=10, 即使用X2 进行变址寻址，其有效地址为1188H+22H=11AAH 。

16.C

指令字长为16位，2字节，因此取指令后PC 的内容为(PC)+2=2002H; 无条件转移指令将 下一条指令的地址送至PC, 形式地址为40H, 指令执行后PC=2002H+0040H=2042H。

17.A

操作码位数固定，且能完成97种操作，则操作码位数是[log₂97 ]=7 位；具有六种寻址方式， 则寻址特征位数是[1og₂6]=3 位；指令字长为16位，因此地址码位数是16- 3- 7=6位，6位补 码的表示范围为-32～+31,即为相对寻址的偏移量范围。

**18.B**

机器按字寻址，程序计数器 (PC) 给出下一条指令字的访存地址(指令在内存中的地址), 因此取决于存储器的字数；指令寄存器 (IR) 用于接收取得的指令，因此取决于指令字长。

**19.D**

直接寻址200访问的操作数是300, A 错 误。寄存器间接寻址 (R) 的访问结果与I一 样 ，B 错误。存储器间接寻址(200)表示主存地址200中的内容为有效地址，有效地址为300,访问的 操作数是400, C 错误。寄存器寻址R 表示寄存器R 的内容为操作数，只有D 正确。

**20.** A

寄存器间接寻址中操作数的有效地址 EA=(R,),8 号寄存器内容为1200H, 因 此 EA=1200H 。

**21.** D 、C

首先需要讲解一下补码扩充的问题。补码的扩充只需使用符号位补足即可，也就是说正数补 码的扩充只要补0,负数补码的扩充只需补1(这是由补码的性质决定的)。理解了该性质，这道

题就变成了十进制转换为十六进制的简单问题。

1)PC 的当前值为240,该指令取出后 PC 的值为243,要求转移到290,即相对位移量为 290 - 243=47,转换成补码为2FH 。由于数据在存储器中采用以低字节地址为字地址的 存放方式，因此该转移指令的第二字节为2FH, 由于47 是正数，因此只需在高位补0, 所以第三字节为00H。

2)PC 的 当 前 值 为 2 4 0 , 该 指 令 取 出 后 PC 的 值 为 2 4 3 , 要 求 转 移 到 2 0 0 , 即 相 对 位 移 量 为 200 - 243= - 43,转换成补码为 D5H 。 由于数据在存储器中采用以低字节地址为字地址的 存放方式，因此该转移指令的第二字节为D5H, 由于 - 43是负数，因此只需在高位补1, 所以第三字节为 FFH。

22.D

基址寻址的操作数的有效地址为基址寄存器内容加上形式地址，即C0000000H+FF00H =C000 0000H+FFFF FF0OH=BFFF FFOOH。由于是大端方式，因此LSB 的存放地址为BFFF FFO3H。

23.D

算术与逻辑运算指令用于完成对 一 个(如自增、取反等)或两个数据的算术运算或逻辑运算， A 错误。移位操作用于把 一 个操作数左移或右移 一 位或多位， B 错 误。转移指令、子程序调用与 返回指令用于解决变动程序中指令执行次序的需求，而不是数据调用次序的需求， C 错 误。

24.C

相 对 寻 址 EA=(PC)+A, 首先计算取指令后的PC 值。转移指令由2字节组成，每取 一 字节

PC 加 1 , 取 指 令 后 的PC 值为2002H, 因 此EA=(PC)+A=2002H+06H=2008H。 本题易误选 A

或 B, 选 项A 未 考 虑PC 值的自动更新，选项B 虽 然 考 虑 了 PC 值的自动更新，但未注意到该转

移 指 令 是 一 条 2 字 节 指 令 ，PC 值应是“+2”而不是“+1”。

25.A

间接寻址不需要寄存器， EA=(A) 。 基 址 寻 址 EA=A+ 基 址 寄 存 器 BR 的 内 容 ； 相 对 寻 址

EA=A+ 程序计数器 PC 的内容；变址寻址EA=A+ 变 址 寄 存 器IX 的内容。后三者都是将某个 寄存器的内容与 一 个形式地址相加而形成有效地址，所以统称偏 移 寻 址。

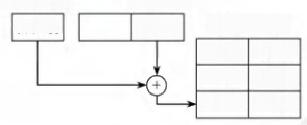
26.C

假设两个无符号整数A 和 B,bgt 指 令 会 将A 和 B 进行比较，也就是将A 和 B 相减。若A>B,

则A-B 肯定无进位/借位，也不为0(为0时表示两数相等),因此CF 和 ZF 均 为 0 , 选C。 其 余 选项中用到了符号标志SF 和 溢 出 标 志OF,SF 表 示 结 果 的 符 号 ，OF 是有符号整数的溢出标志位， 对于无符号数运算， SF 和 OF 没有意义，显然应当排除。

27.D

根据变址寻址的方法，变址寄存器的内容(1000H) 与形式地址的内容(2000H) 相 加 ， 得 到 操作数的实际地址(3000H), 根据实际地址访问内存，获取操作数4000H, 如 下 图 所 示 。



形式地址

2000H

2000H

3000H

3000 H 4000H

地址 内容 1000H 2000H

**变址寄存器**

1000 H

**28.A**

采 用 3 2 位 定 长 指 令 字 ， 其 中 操 作 码 为 8 位 ， 两 个 地 址 码 共 占 用 3 2 - 8 = 2 4 位 ， 而STORE 指令的源操作数和目的操作数分别采用寄存器直接寻址和基址寻址，机器中共有16个通用寄存器， 因此寻址 一个寄存器需要log,16=4 位，源操作数中的寄存器直接寻址用掉4位，而目的操作数采用 基址寻址也要指定 一 个寄存器，同样用掉4位，则留给偏移量的位数为24 - 4 - 4=16位，而偏移量 用补码表示，因此16位补码的表示范围为-32768～+32767。

**29.C**

变址寻址中，有效地址 (EA) 等于指令字中的形式地址 D 与 变 址 寄 存 器 I 的内容之和，即

EA=(I)+D。 间接寻址是相对于直接寻址而言的，指令的地址字段给出的形式地址不是操作数的 真正地址，而是操作数地址的地址，即 EA=(D)。 从而该操作数的有效地址是((I)+D)。

30.D

在变址操作时，将计算机指令中的地址与变址寄存器中的地址相加，得到有效地址，指令提 供数组首地址，由变址寄存器来定位数据中的各元素。所以它最适合按下标顺序访问一维数组元 素，选 D。 相对寻址以 PC 为基地址，以指令中的地址为偏移量确定有效地址。寄存器寻址则在 指令中指出需要使用的寄存器。直接寻址在指令的地址字段直接指出操作数的有效地址。

31.B

根据变址寻址的公式EA=(IX)+A, 有(IX)=2100H-2000H=100H=256,sizeof(double)=8 (双精度浮点数用8位字节表示),因此数组的下标为256/8=32。

32.D

注意，内存地址是无符号数。

操作数采用基址寻址方式，EA=(BR)+A, 基址寄存器 BR 的内容为F0000000H, 形式地址 用补码表示为FF12H 即1111111100010010B, 因此有效地址为 F0000000H+(-00EEH)=EFFF FF12H。计算机采用大端方式编址，所以低位字节存放在字的高地址处，机器数一共占4字节， 该操作数的LSB 所在的地址是 EFFF FF12H+3=EFFF FF15H。

33.A

48条指令需要6位操作码字段(2⁵<48<26),4种寻址方式需要2位寻址特征位(4=2²), 还剩16-6-2=8位作为地址码，所以直接寻址范围为0～255。注意，主存地址不能为负。

34.A

指令字由操作码、寻址特征和地址码三个字段组成，寻址特征字段用来指明指令属于哪种寻 址方式。若寻址方式是寄存器直接寻址，则地址码所指的通用寄存器中存放的是操作数，若寻址 方式是寄存器间接寻址，则对应通用寄存器中存放的是操作数的地址。

二、综合应用题

01.【解答】

1)因为OP 字段长为5位，所以指令能定义2⁵=32种指令。

2)、3)各种寻址方式的有效地址 EA 的计算公式、寻址范围见下表。

|  |  |  |
| --- | --- | --- |
| **寻** **址** **方** **式** | **有效地址EA的计算公式** | **寻** **址** **范** **围** |
| 直接寻址 | EA=A | 2⁸=256 |
| 间接寻址 | EA=(A) | 2¹⁶ |
| 变址寻址 | EA=(Rx)+A | 2¹⁶ |
| 相对寻址 | EA=(PC)+A | 2⁸=256(PC附近256) |

02.【解答】

1)直接寻址时，有效地址是指令中的地址码部分500,装入ACC 的是800。

2)立即寻址时，指令的地址码部分是操作数而不是地址，所以将500装入ACC。

3)间接寻址时，操作数的有效地址存储在地址为500的单元中，由此得到有效地址为800, 操作数是300。

4)相对寻址时，有效地址EA=(PC)+A=202+5 00=702, 所以装入ACC 的操作数是325。 这是因为指令是双字长，在该指令的执行阶段， PC 的内容已经加2,更新为下一条指令

5)变址寻址时，有效地址EA=(XR)+A=100+500=600, 所以装入ACC 的操作数是900。

6)寄存器寻址时，R1 的内容400装入ACC。

7)寄存器间接寻址时，有效地址是R1 的内容400,装入ACC 的操作数是700。

|  |  |
| --- | --- |
| 03. 【解答】 |  |
| 取指令后，PC=1235H | (注意，不是1236H, 因主存按字编址)。 |
| ① X=00,D=20H, | 有效地址 EA=20H。 |
| ② X=10,D=44H, | 有效地址 EA=1122H+44H=1166H。 |
| ③ X=11,D=22H, | 有效地址EA=1235H+22H=1257H。 |
| ④ X=01,D=21H, | 有效地址EA=0037H+21H=0058H。 |
| ⑤ X=11,D=23H, | 有效地址 EA=1235H+23H=1258H。 |
| 04. 【解答】 |  |

1 ) 因 为PC 的增量是2,且每条指令占2字节，所以编址单位是字节。

2)根据“大于”条件判断表达式，可以看出该bgt 指令实现的是有符号整数比较。因为无符 号数比较时，其判断表达式中没有溢出标志 OF 。继续分析该逻辑表达式，bgt 指令的含 义是当两数相减的结果大于0时，执行转移操作。因此，要满足bgt 指令的条件，必须保 证如下两个条件： 一是结果不为0,即零标志位 ZF 为0;二是结果的符号位与溢出标志 位 OF 相同，即 SF④OF 为0(两数相减结果大于0,有两种情况：第一种情况是结果没 有溢出，此时OF 位和 SF 位都为0;第二种情况是结果发生了溢出，此时 OF 和 SF 位都 为1)。综上所述，逻辑表达式可表示为ZF+(SF④OF)=0。

3)偏移地址 Imm8 为补码表示，说明转移目标地址可能在bgt 指令之后。计算转移目标地址 时，偏移量为Imm8×2, 说明Imm8 不是相对地址，而是相对指令数。Imm8 的范围为-128~ 127,所以转移目地址的范围是PC+2+(-128×2)～PC+2+127×2, 也即转移目标地址的

范围是相对于 bgt 指令的前127条指令到后128条指令之间。

**05.【** 解答】

直接寻址：寄存器的内容是有效地址EA, 所以直接寻址的有效地址为300,根据题给出的表 格可知，地址300 对应的内容为400。

间接寻址：根据寄存器的内容寻找到的内容才是真正的有效地址，所以根据寄存器内容300 找到的400才是间接寻址的有效地址，因此有效地址为400,地址400对应的内容为700。

相对寻址：寄存器的内容加上PC 的内容为有效地址，PC 的当前值为200,所以当取出一条 指令后，变为202,因此有效地址为202+300=502,地址502对应的内容为900。

变址寻址：变址寻址的有效地址为变址寄存器的内容加上累加器的内容，所以有效地址为100+ 300=400,地址400 对应的内容为700。

基址寻址：基址寻址的有效地址为基址寄存器的内容加上累加器的内容，所以有效地址为 200+300=500,地址500 对应的内容为600。

先变址后间址：先变址，即先让变址寄存器的内容加上累加器的内容，即400;再间址，意 思就是根据地址400找到的内容才是有效地址，所以先变址后间址的有效地址为700。地址700 对应的内容为401。

先间址后变址：先间址，即先根据累加器的内容300找到间址的有效地址400;再变址，即 400再加上变址寄存器的内容，也就是400+100=500,地址500对应的内容为600。

综上，得到下表：

182-2025年计算机组成原理考研复习指导

|  |  |  |
| --- | --- | --- |
| **寻** **址** **方** **式** | **有效地址(EA)** | **累加器(AC)的内容** |
| 立即寻址 |  | 300 |
| 直接寻址 | 300 | 400 |
| 间接寻址 | 400 | 700 |
| 相对寻址 | 502 | 900 |
| 变址寻址 | 400 | 700 |
| 基址寻址 | 500 | 600 |
| 先变址后间址 | 700 | 401 |
| 先间址后变址 | 500 | 600 |

06.【解答】

1)操作码占4位，则该指令系统最多可有2⁴=16条指令。操作数占6位，其中寻址方式占 3位、寄存器编号占3位，因此该机最多有2³=8个通用寄存器。主存地址空间大小为 128KB, 按字编址，字长为16位，共有128KB/2B=2¹6 个存储单元，因此 MAR 至少为

16位；本题已说明了存储字长为16位，因此 MDR至少为16位。

2)寄存器字长为16位， PC 可表示的地址范围为0～216-1,Rn 可表示的相对偏移量为-2I⁵~2¹5

-1,而主存地址空间为2¹⁶,因此转移指令的目标地址范围为0000H~FFFFH(O~2l⁶-1)。 3)汇编语句 “add(R4),(R5)+” 对应的机器码为

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | OP | Ms | Rs | Md | Rd |
| 内容 | 0010 | 001 | 100 | 010 | 101 |
| 说明 | add | 寄存器间接 | R4 | 寄存器间接、自增 | R5 |

将对应的机器码写成十六进制形式为0010001100010101B=2315H。

该指令的功能是将 R4 的内容所指的存储单元的数据与 R5 的内容所指的存储单元的数据相 加，并将结果送入R5 的内容所指的存储单元中。(R4)=1234H,(1234H)=5678H;(R5)=5678H,

(5678H)=1234H; 执行加法操作5678H+1234H=68ACH。 之后R5 自增。

该指令执行后， R5 和存储单元5678H的内容会改变，R5 的内容从5678H 变为5679H, 存储 单元5678H 中的内容变为该指令的计算结果68ACH。

07.【解答】

1)因为指令长度为16位，且下一条指令地址为(PC)+2, 因此编址单位是字节。

相对偏移量OFFSET为8位补码，表示范围为-128～127,根据转移目标地址为(PC)+2+ 2×OFFSET, 若要向后跳转，则要求OFFSET 必须为负数，OFFSET的最小值为-128,但 在执行转移指令之前， PC 进行了自增+2的操作，所以向后最多可跳转127条指令。

2)指令中C=0,Z=1,N=1, 因此应根据ZF 和NF 的值来判断是否转移。CF=0,ZF=0, NF=1 时，需转移。已知指令中的偏移量为11100011B=E3H, 符号扩展后为FFE3H, 左移一位(乘以2)后为FFC6H, 因此PC的值(即转移目标地址)为200CH+2+FFC6H= 1FD4H 。CF=1,ZF=0,NF=0 时不转移。PC 的值为200CH+2=200EH。

3)指令中的C 、Z 和 N 应分别设置为C=Z=1,N=0 。 两个数之间的大小比较通常是对两 个数做减法运算，即两个数相减当结果为0或为负时转移，若为0,则ZF 标志应当是1, 若为负，则借位标志应该是1,而无符号数并不涉及符号标志NF。

4)部件①用于存放当前指令，不难得出为指令寄存器；多路选择器根据符号标志 C/Z/N 来决定下一条指令的地址是PC+2 还是PC+2+2×OFFSET, 因此多路选择器左边线

上的结果应是 PC+2+2×OFFSET 。 根据运算的先后顺序及与 PC+2 的连接，部件②

用于左移一位实现乘以2,为移位寄存器。部件③用于PC+2 和2×OFFSET 相加，为

加法器。

部件②:移位寄存器(用于左移一位);部件③:加法器(地址相加)。 **08.【**解答】

1)指令操作码有7位，因此最多可定义2⁷=128条指令。

2)各条指令的机器代码如下：

① “inc R1” 的机器码为0000001001000000,即0240H。

② “shl R2,R1” 的机器码为0000010010001000,即0488H。

③ “sub R3,(R1),R2” 的机器码为0000011011101010,即06EAH。 3)各标号处的控制信号或控制信号取值如下：

①0;②mov;③mova;④left;⑤read;⑥sub;⑦mov;⑧SRout 。

4 ) 指 令 “subR1,R3,(R2)” 的执行阶段至少包含4个时钟周期；指令 “inc R1” 的执行阶段 至少包含2个时钟周期。

**09.【**解答】

1)ALU 的宽度为16位，ALU 的宽度即 ALU 运算对象的宽度，通常与字长相同。地址线为 20位，按字节编址，可寻址主存空间大小为22°字节(或1MB) 。指令寄存器有16位，和单 条指令长度相同。MAR 有20位，和地址线位数相同。MDR 有8位，和数据线宽度相同。

2)R 型格式的操作码有4位，最多有2⁴ (或16)种操作。 I 型 和J 型格式的操作码有6位， 因为它们的操作码部分重叠，所以共享这6位的操作码空间，且前6位全为0的编码已 被 R 型格式占用，因此I 和 J 型格式最多有2⁶- 1=63种操作。从 R 型和1型格式的寄存 器编号部分可知，只用2位对寄存器编码，因此通用寄存器最多有4个。

3)指令01B2H=0000000110110010B 为一条R 型指令，操作码0010表示有符号整数减法指

令，其功能为R[3]-R[1]-R[2] 。 执行指令01B2H 后 ，R[3]=B052H-0008H=B04AH, 结

果未溢出。指令01B3H=0000000110110011B, 操作码0011 表示有符号整数乘法指令，执 行指令01B3H 后，R[3]=R[1]×R[2]=B052H×0008H=8290H,B052H 乘以8相当于将 B052H 算术左移3位，由于B052H 是一个负数，符号位为1,在算术左移的过程中移出了101,不 全为1,由此可以判断结果溢出。

4)在进行指令的跳转时，既可能向前跳转，又可能向后跳转，偏移量是一个有符号整数， 因此在地址计算时，应对 imm 进行符号扩展。

5)无条件转移指令可以采用J 型格式，将target 部分写入 PC 的低10位，完成跳转。

**4.3** **程序的机器级代码表示**

▶ **涉及过汇编代码的真题的年份(2012、2014、2015、2017、2019、2023)**

本节是2022 年才新增的考点，但历年统考真题曾多次以综合题的形式考查过，难度较大， 不少跨考生对此无从下手，相信通过本节的学习后，应能从容应对。统考大纲没有指定具体指令 集，但历年统考真题主要考查的是 x86 汇编指令，因此本节主要介绍 x86 汇编指令。

184-2025年计算机组成原理考研复习指导

**4.3.1** **常** **用** **汇** **编** **指** **令** **介** **绍**

**1.** **相关寄存器**

x86 处理器中有8个32位的通用寄存器，各寄存器及说明如图4.11 所示。为了向后兼容， EAX 、EBX 、ECX 和 EDX 的高两位字节和低两位字节可以独立使用，E 表示 Extended, 表示32 位的寄存器。例如， EAX 的低两位字节称为AX, 而 AX 的高低字节又可分别作为两个8位寄存 器，分别称为AH 和 AL。

通用寄存器

1615

31

87

0

|  |  |  |
| --- | --- | --- |
|  | AH | AL |
|  | BH | BL |
|  | CH | CL |
|  | DH | DL |
| ESI | | |
| EDI | | |
| EBP | | |
| ESP | | |

16bit 32bit

AX EAX

BX EBX

CX ECX

DX EDX ESI EDI EBP

ESP

说明

累加器 (Accumulator)

基地址寄存器 (Base Register)

计数寄存器 (Count Register)

数据寄存器 (Data Register)

变址寄存器 (Index Register)

堆栈基指针 (Base Pointer)

堆栈顶指针 (Stack Pointer)

图4.11 x86 处理器中的主要寄存器及说明

除 EBP 和 ESP 外，其他几个寄存器的用法是比较灵活的。

**2.** **汇编指令格式**

使用不同的编程工具开发程序时，用到的汇编程序也不同， 一般有两种不同的汇编格式： AT&T 格式和Intel 格式(统考要求掌握的是 Intel 格式 )。它们的区别主要体现如下：

① AT&T 格式的指令只能用小写字母，而Intel 格式的指令对大小写不敏感。

② 在 AT&T 格式中，第一个为源操作数，第二个为目的操作数，方向从左到右，合乎自然； 在 Intel 格式中，第一个为目的操作数，第二个为源操作数，方向从右向左。

③ 在 AT&T 格式中，寄存器需要加前缀“%”,立即数需要加前缀“$”;在 Intel 格式中，寄 存器和立即数都不需要加前缀。

④ 在内存寻址方面， AT&T 格式使用“(”和“)”,而Intel 格式使用“[”和“]”。

⑤ 在处理复杂寻址方式时，例如AT&T 格式的内存操作数“disp(base,index,scale)” 分别表

示偏移量、基址寄存器、变址寄存器和比例因子，如“8(%edx,%eax,2)” 表示操作数为 M[R[edx]+R[eax]\*2+8], 其对应的Intel 格式的操作数为“[edx+eax\*2+8]”。

⑥ 在指定数据长度方面，AT&T 格式指令操作码的后面紧跟一个字符，表明操作数大小，“b” 表示 byte ( 字节)、“w” 表 示 word ( 字)或“1”表示long ( 双 字 ) 。Intel 格式也有类似 的语法，它在操作码后面显式地注明 byte ptr 、word ptr 或 dword ptr。



**注** **意**

由于32或64位体系结构都是由16位扩展而来的，因此用 word (字)表示16位。

表4.2展示了两种格式的几条不同指令。其中， mov 指令用于在内存和寄存器之间或者 寄存器之间移动数据； lea 指令用于将 一个内存地址(而不是其所指的内容)加载到目的寄 存 器。

第4章 指令系统 185

**表4.2** **AT&T** **格式指令和** **Intel** **格式指令的对比**

|  |  |  |  |
| --- | --- | --- | --- |
| AT&T格式 | Intel格式 | | 含义 |
| mov $100,&eax | mov | eax,100 | 100→R[eax] |
| mov &eax,&ebx | mov | ebx,eax | R[eax]→R[ebx] |
| mov &eax,(&ebx) | mov | [ebx],eax | R[eax]→M[R[ebx]] |
| mov seax,-8(8ebp) | mov | [ebp-8],eax | R[eax]→M[R[ebp]-8] |
| lea 8(&edx,&eax,2),&eax | lea | eax,[edx+eax\*2+8] | R[edx]+R[eax]\*2+8→R[eax] |
| mov1 &eax,&ebx | mov | dword ptr ebx,eax | 长度为4字节的R[eax]→R[ebx |

注：R[r]表示寄存器r 的内容， M|addr]表示主存单元 addr 的内容， →或一表示信息传送方向。

两种汇编格式的相互转换并不复杂，但历年统考真题采用的均是 Intel 格式。

**3.** **常用指令**

汇编指令通常可分为数据传送指令、算术和逻辑运算指令和控制流指令，下面以 Intel 格式为 例，介绍一些常用的指令。以下用于操作数的标记分别表示寄存器、内存和常数。

·<reg>: 表示任意寄存器，若其后带有数字，则指定其位数，如<reg32> 表示32位寄存器(eax, ebx,ecx,edx,esi,edi,esp 或 ebp);<reg16> 表示16位寄存器 (ax,bx,cx 或 dx);<reg8> 表 示8位寄存器 (ah,al,bh,bl,ch,cl,dh,dl)。

●<mem>: 表示内存地址(如[eax] 、[var+4] 或 dword ptr [eax+ebx])。

·<con>: 表示8位、16位或32位常数。<con8> 表示8位常数； <con16> 表示16位常数； <con32> 表示32位常数。

**命题追踪** ▶ **分析汇编指令对应的二进制代码(2010)**

x86 中的指令机器码长度为1 字节，对同一指令的不同用途有多种编码方式，比如 mov 指令 就有28种机内编码，用于不同操作数类型或用于特定寄存器，例如，

|  |  |
| --- | --- |
| mov ax,<con16> | #机器码为B8H |
| mov al,<con8> | #机器码为BOH |
| mov <reg16>,<reg16>/<mem16> | #机器码为89H |
| mov <reg8>/<mem8>,<reg8> | #机器码为8AH |
| mov <reg16>/<mem16>,<reg16> | #机器码为8BH |

**命题追踪** ▶ **模仿写出简单语句的机器级指令(2012)**

(1)数据传送指令

1) **mov 指令。**将第二个操作数(寄存器的内容、内存中的内容或常数值)复制到第一个操

作数(寄存器或内存)。

其语法如下：

|  |  |
| --- | --- |
| mov | <reg>,<reg> |
| mov | <reg>,<mem> |
| mov | <mem>,<reg> |
| mov | <reg>,<con> |
| mov | <mem>,<con> |

#将ebx 值复制到eax

#将5保存到 var 值指示的内存地址的一字节中

举例：

mov

mov

eax,ebx

byte ptr

[var],5

双操作数指令的两个操作数不能都是内存，即 mov 指令不能用于直接从内存复制到内存，若



186—2025年计算机组成原理考研复习指导

需在内存之间复制，可先从内存复制到一个寄存器，再从这个寄存器复制到内存。

2) **push 指令。**将操作数压入内存的栈，常用于函数调用。ESP 是栈顶，入栈前先将 ESP 值 减4(栈增长方向与内存地址增长方向相反),然后将操作数压入ESP 指示的地址。

其语法如下：

push <reg32>

push <mem>

push <con32>

举例(注意，栈中元素固定为32位 ) :



push eax #将eax 值入栈

push [var] #将var 值指示的内存地址的4字节值入栈

3) **pop** **指** **令** **。**与 push 指令相反，pop 指令执行的是出栈工作，出栈前先将 ESP 指示的地址 中的内容出栈，然后将 ESP 值加4。



其语法如下：

pop eax #弹出栈顶元素送到eax

pop [ebx] #弹出栈顶元素送到 ebx 值指示的内存地址的4字节中

(2)算术和逻辑运算指令

1) **add/sub 指令。**add 指令将两个操作数相加，相加的结果保存到第一个操作数中。sub 指

令用于两个操作数相减，相减的结果保存到第一个操作数中。

它们的语法如下：

add <reg>,<reg>/sub <reg>,<reg>

add <reg>,<mem>/sub <reg>,<mem>

add <mem>,<reg>/sub <mem>,<reg>

add <reg>,<con>/sub <reg>,<con>

add <mem>,<con>/sub <mem>,<con>

举例：

sub eax,10 #eax ←eax-10

add byte ptr [var],10 #10与 var 值指示的内存地址的一字节值相加，并将结果

#保存在 var 值指示的内存地址的字节中 2) **inc/dee** **指** **令** **。**inc、dec 指令分别表示将操作数自加1、自减1。

它们的语法如下：

inc <reg>/dec <reg>

inc <mem>/dec <mem>

举例：

dec eax #eax 值自减1

inc dword ptr [var] #var 值指示的内存地址的4字节值自加1

3) **imul** **指令。**有符号整数乘法指令，有两种格式：①两个操作数，将两个操作数相乘，将 结果保存在第一个操作数中，第一个操作数必须为寄存器；②三个操作数，将第二个和

第三个操作数相乘，将结果保存在第一个操作数中，第一个操作数必须为寄存器。 其语法如下：

imul <reg32>,<reg32>

imul <reg32>,<mem>

imul <reg32>,<reg32>,<con>

imul <reg32>,<mem>,<con>

举 例 ：



imul eax,[var] #eax -eax \*[var]

imul esi,edi,25 #esi -edi \*25

乘法操作结果可能溢出，则编译器置溢出标志OF=1, 以 使CPU 调出溢出异常处理程序。

第4章 指令系统-187

4) **idiv** **指令。**有符号整数除法指令，它只有一个操作数，即除数，而被除数则为edx:eax 中 的内容(共64位),操作结果有两部分：商和余数，商送到eax, 余数则送到edx。

其语法如下：

idiv <reg32>

idiy <mem>

举例：

idiv ebx

idiv dword ptr [var]

5) **and/or/xor 指令。**and 、or 、xor 指令分别是逻辑与、逻辑或、逻辑异或操作指令，用于操

作数的位操作，操作结果放在第一个操作数中。

它们的语法如下：

|  |  |  |  |
| --- | --- | --- | --- |
| and | <reg>,<reg>/or | <reg>,<reg>/xor | <reg>,<reg> |
| and | <reg>,<mem>/or | <reg>,<mem>/xor | <reg>,<mem> |
| and | <mem>,<reg>/or | <mem>,<reg>/xor | <mem>,<reg> |
| and | <reg>,<con>/or | <reg>,<con>/xor | <reg>,<con> |
| and | <mem>,<con>/or | <mem>,<con>/xor | <mem>,<con> |



举例：

and eax,0fH #将eax 中的前28位全部置为0,最后4位保持不变

xor edx,edx #置edx 中的内容为0

6) **not 指令。**位翻转指令，将操作数中的每一位翻转，即0 → 1、1 →0。 其语法如下：

not <reg>

not <mem>

|  |
| --- |
| 举例： not byte ptr [var] #将var值指示的内存地址的 一 字节的所有位翻转 |
| 7) **neg 指令。**取负指令。  其语法如下： |
| neg <reg>  neg <mem> |
| 举 例 ： |

neg eax #eax --eax

**8)shl/shr 指令。**逻辑移位指令，shl 为逻辑左移， shr 为逻辑右移，第一个操作数表示被操 作数，第二个操作数指示移位的位数。

它们的语法如下：



shl <reg>,<con8>/shr <reg>,<con8>

shl<mem>,<con8>/shr <mem>,<con8>

shl <reg>,<cl>/shr <reg>,<cl>

shl <mem>,<cl>/shr <mem>,<cl>

举例：

shl eax,1 #将 eax 值左移1位

shr ebx,cl #将ebx 值右移n 位 (n 为 cl 中的值)

(3)控制流指令

x86 处理器维持着一个指示当前执行指令的指令指针 (IP), 当一条指令执行后，此指针自动 指向下一条指令。IP 寄存器不能直接操作，但可以用控制流指令更新。通常用标签 (label) 指示 程序中的指令地址，在x86 汇编代码中，可在任何指令前加入标签。例如，

mov esi,[ebp+8]

begin: xor ecx,ecx

mov eax,[esi]

188-2025年计算机组成原理考研复习指导

这样就用 begin 指示了第二条指令，控制流指令通过标签就可以实现程序指令的跳转。

▶ **条件转移指令的指令格式(2021)**

1) **jmp 指令。**jmp 指令控制IP 转移到label 所指示的地址(从label 中取出指令执行)。 其语法如下：

jmp <label>

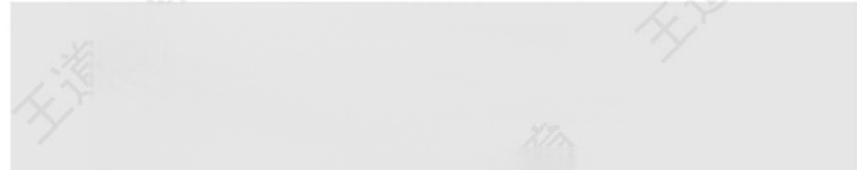
举例：

jmp begin #转跳到begin 标记的指令执行

**命题追踪** ▶ **条件转移指令与标志位的结合(2013)**

2) **jcondition** **指令。**条件转移指令，依据 CPU 状态字中的一系列条件状态转移。CPU 状态

字中包括指示最后一个算术运算结果是否为0,运算结果是否为负数等。 其语法如下：



je <label>(jump when equal)

jz <label>(jump when last result was zero)

jne <label>(jump when not equal)

jg <label>(jump when greater than)

jge <label>(jump when greater than or equal to)

jl <label>(jump when less than)

jle <label>(jump when less than or equal to)

举例：

#若eax 值<=ebx 值，则跳转到done 执行；否则执行下一条指令

cmp jle

eax,ebx done

3) **cmp/test 指令**。cmp 指令的功能相当于sub 指令，用于比较两个操作数的值。test 指令的 功能相当于and 指令，对两个操作数进行逐位与运算。与 sub 和 and 指令不同的是，这两

类指令都不保存操作结果，仅根据运算结果设置CPU 状态字中的条件码。 其语法如下：

cmp <reg>,<reg>/test <reg>,<reg>

cmp <reg>,<mem>/test <reg>,<mem>

cmp <mem>,<reg>/test <mem>,<reg>

cmp <reg>,<con>/test <reg>,<con>

cmp 和 test 指令通常和 jcondition指令搭配使用，举例：

cmp dword ptr [var],10 #将var 指示的主存地址的4字节内容，与10比较

jne loop #若相等则继续顺序执行；否则跳转到loop 处执行

test eax,eax #测试 eax 是否为零

jz xxxx #为零则置标志zF 为1,转跳到xxxx 处执行

**命题追踪** ▶ **call指令的功能(2019)**

4) **call/ret 指令。**分别用于实现子程序(过程、函数等)的调用及返回。

其语法如下：

call <label>

ret

call 指令首先将当前执行指令地址入栈，然后无条件转移到由标签指示的指令。与其他简单 的跳转指令不同，call 指令保存调用之前的地址信息(当 call 指令结束后，返回调用之前的地址)。 ret 指令实现子程序的返回机制， ret 指令弹出栈中保存的指令地址，然后无条件转移到保存的指 令地址执行。call 和ret 是程序(函数)调用中最关键的两条指令。

第4章 指令系统-189

理解上述指令的语法和用途，可以更好地帮助读者解答相关题型。读者在上机调试C 程序代

码时，也可以尝试用编译器调试，以便更好地帮助理解机器指令的执行。

**4.3.2** **选择语句的机器级表示**

常见的选择结构语句有if-then 、if-then-else 等。编译器通过条件码(标志位)设置指令和各 类转移指令来实现程序中的选择结构语句。条件码描述了最近的算术或逻辑运算操作的属性，可 以检测这些寄存器来执行条件分支指令，最常用的条件码有CF 、ZF 、SF 和 OF。

常见的算术逻辑运算指令 (add,sub,imul,or,and,shl,inc,dec,not,sal 等)会设置条件码，还 有 cmp 和 test 指令只设置条件码而不改变任何其他寄存器。

之前介绍的**jcondition** 条件转跳指令，就是根据条件码 ZF 和 SF 来实现转跳的。



if-else 语句的通用形式如下：

if(test\_expr)

**then\_statement**

else

**else\_statement**

这里的 test\_expr 是 一 个整数表达式，它的取值为0(假),或为非0(真)。两个分支语句 (then\_statement 或 else\_statement) 中只会执行一个。

这种通用形式可以被翻译成如下所示的 goto 语句形式：

e!\_expr;

goto false;

**then\_statement**

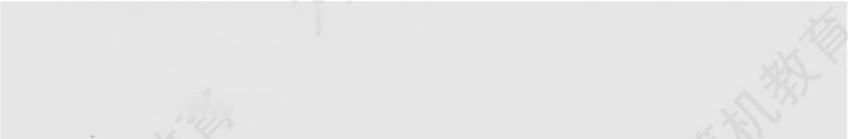
goto done;

false:

**else\_statement**

done:

对于下面的C 语言函数：



int get\_cont(int \*pl,int \*p2){

if(p1>p2)

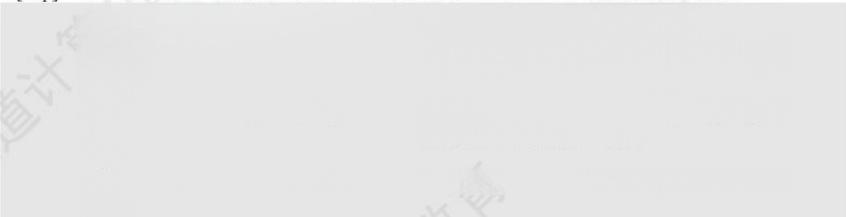
return \*p2;

else

return \*pl;

}

已 知pl 和 p2 对应的实参已被压入调用函数的栈帧，它们对应的存储地址分别为 R[ebp]+8、



R[ebp]+12(EBP 指向当前栈帧底部),返回结果存放在EAX 中。对应的汇编代码为

mov eax,dword ptr [ebp+8] #R[eax]-M[R[ebp]+8],即R[eax]=p1

mov edx,dword ptr [ebp+12] #R[edx]-M[R[ebp]+12],即R[edx]=p2

cmp eax,edx #比较p1 和 p2, 即根据p1-p2 的结果置标志

jbe .Ll #若 pl<=p2, 则转标记L1 处执行

mov eax,dword ptr [edx] #R[eax]-M[R[edx]],即R[eax]=M[p2]

jmp .L2 #无条件跳转到标记 L2 执行

.L1:

mov eax,dword ptr [eax] #R[eax]-M[R[eax]], 即 R[eax]=M[p1]

.L2:

pl 和 p2 是指针型参数，所以在32位机中的长度是dword ptr, 比较指令 cmp 的两个操作数都 应来自寄存器，因此应先将pl 和 p2 对应的实参从栈中取到通用寄存器，比较指令执行后得到各个 条件码，然后根据各条件码值的组合选择执行不同的指令，因此需要用到条件转移指令。

190-2025年计算机组成原理考研复习指导

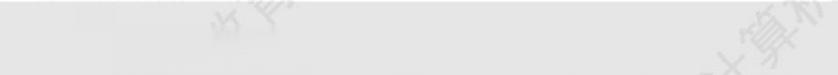
**4.3.3** **循环语句的机器级表示**

命题追踪 **循环语句的机器级代码分析(2014、2017、2019、2023)**

常见的循环结构语句有while 、for 和 do-while。汇编中没有相应的指令存在，可以用条件测 试和转跳组合起来实现循环的效果，大多数编译器将这三种循环结构都转换为 do-while 形式来产 生机器代码。在循环结构中，通常使用条件转移指令来判断循环条件的结束。

(1)do-while 循环

do-while 语句的通用形式如下：

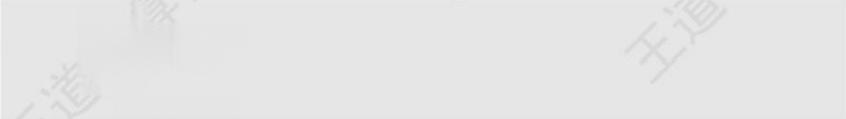


do

**body\_statement**

while(test\_expr);

这种通用形式可以被翻译成如下所示的条件和 goto 语句：



loop:

**body\_statement**

t=test expr;

if(t)

goto loop;

也就是说，每次循环，程序会执行循环体内的语句， body\_statement 至少会执行一次，然后 执行测试表达式。若测试为真，则继续执行循环。

(2)while 循环

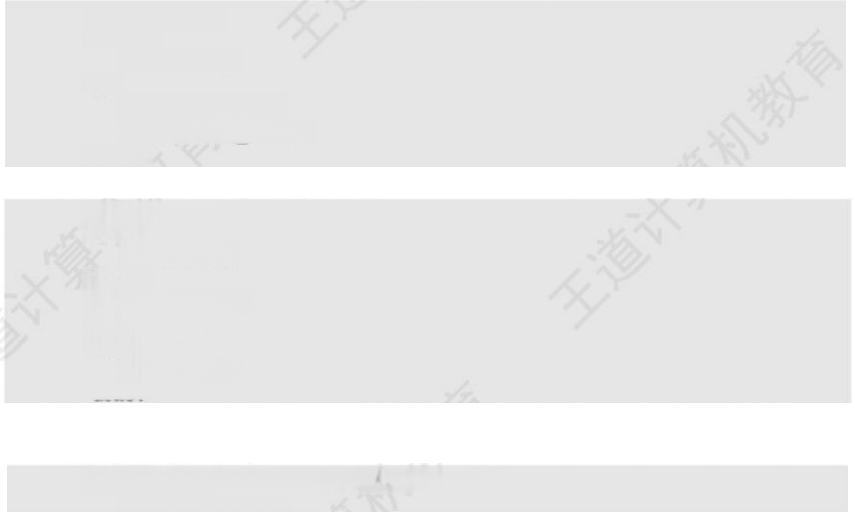
while 语句的通用形式如下：



while(test\_expr)

**body\_statement**

与 do-while 的不同之处在于，第一次执行 body\_statement 之前，就会测试test\_expr 的值，循 环有可能中止。GCC 通常会将其翻译成条件分支加 do-while 循环的方式。



用如种方法，将通用的 while 循环格式翻译成 do-while 循环：

goto done;

do

**body\_statement**

while(test\_expr);

done;

相应=，tt一e翻译成 goto 语句：

if(!t)

goto done;

loop:

**o****d**s**ent**

goto loop;

done:

(3)for 循环

for 循环的通用形式如下：

for(init\_expr;test\_expr;update\_expr)

**body\_statement**

第4章 指令系统 191

这个 for 循环的行为与下面这段 while 循环代码的行为一样：

init\_expr;

while(test\_expr){

**body\_statement**

update\_expr;

}

进一步把它翻译成 goto 语句：

init\_expr;

loop:

**body\_statement**

t\_\_er;

goto loop;

done:

下面是一个用 for 循环写的自然数求和的函数：

(t!t\_expr;

goto done;



int nsum\_for(int n){

(ii0+;+)

result +=i;

return result;

}



这段代码中的 for 循环的不同组成部分如下：

init expr i=1

test\_expr

update\_expr

body\_statement

通过替换前面给出的模板中的相应位置，很容易将 for 循环转换为 while 或 do-while 循环。 将这个函数翻译为goto 语句代码后，不难得出其过程体的汇编代码：

i<=n

i++

result +=i



ptr [ebp+]-即[Rt8=]0,即R[ecx]=n

#R[edx]-1, 即 i=1

#Compare R[edx]:R[ecx], 即比较i:n

#If greater, 转跳到L2 执行

#loop:

#R[eax]-R[eax]+R[edx],即result +=i #R[edx]-R[edx]+1, 即 i++

#比较R[edx] 和R[ecx], 即比较i:n #If less or equal,转跳到 L1执行

mov ecx,dword mov eax,0

mov edx,1

cmp edx,ecx jg .L2

eax,edx edx,1

edx,ecx .Ll

.L1: add add

cmp

jle .L2:

已知n 对应的实参已被压入调用函数的栈帧，其对应的存储地址为R[ebp]+8, 过程nsum\_for 中 的局部变量i 和 result 被分别分配到寄存器 EDX和 EAX 中，返回参数在 EAX中。

**4.3.4** **过程调用的机器级表示**

前面提到的 call/ret 指令主要用于过程调用，它们都属于一种无条件转移指令。

192-2025年计算机组成原理考研复习指导

假定过程 P (调用者)调用过程Q (被调用者),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(通用寄存器的内容),并为自己的非静态局部变量分配空间。

4)执行过程Q。

5)Q 恢 复P 的现场，将返回结果放到P 能访问到的地方，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。

步 骤 2 ) 是 由call 指令实现的，步骤6)通过ret 指令返回到过程 P。在上述步骤中，需要为 入口参数、返回地址、过程P 的现场、过程Q 的局部变量、返回结果找到存放空间。

用户可见寄存器数量有限，调用者和被调用者需共享寄存器，若直接覆盖对方的寄存器，则 会导致程序出错。因此有如下规范：寄存器 EAX、ECX 和 EDX 是调用者保存寄存器，当P 调 用 Q 时，若Q 需用到这些寄存器，则由P 将这些寄存器的内容保存到栈中，并在返回后由P 恢复它 们的值。寄存器 EBX 、ESI 、ED I 是被调用者保存寄存器，当 P 调用Q 时 ，Q 必须先将这些寄存 器的内容保存在栈中才能使用它们，并在返回P 之前先恢复它们的值。

每个过程都有自己的栈区，称为栈帧，因此， 一个栈由若干栈帧组成，寄存器 EBP 指示栈帧 的起始位置，寄存器 ESP 指示栈顶，栈从高地址向低地址增长。过程执行时，ESP 会随着数据的 入栈而动态变化，而 EBP 固定不变。当前栈帧的范围在 EBP 和 ESP 指向的区域之间。



下面用一个简单的C 语言程序来说明过程调用的机器级实现。

int add(int x,int y){

return x+y;

}

int caller(){

int templ=125;

int temp2=80;

int sum=add(templ,temp2);

return sum;

}

经 GCC 编译后， caller 过程对应的代码如下(#后面的文字是注释):



caller:

push ebp

mov ebp,esp

sub esp,24

mov [ebp-12],125 #M[R[ebp]-12]-125,即temp1=125

mov [ebp-8],80 #M[R[ebp]-8]-80, 即temp2=80

mov eax,dword ptr [ebp-8] #R[eax]-M[R[ebp]-8], 即 R[eax]=temp2

mov [esp+4],eax #M[R[esp]+4]-R[eax], 即 temp2 入栈

mov eax,dword ptr [ebp-12]#R[eax]~M[R [ebp]-12], 即 R[eax]=temp1

mov [esp],eax #M[R[esp] ]-R[eax], 即 temp1 入栈

call add #调用 add, 将返回值保存在 eax 中

mov [ebp-4],eax #M[R[ebp]-4]-R[eax], 即 add 返回值送 sum

mov eax,dword ptr [ebp-4] #R[eax]-M [R[ebp]-4], 即 sum作为返回值

leave

ret

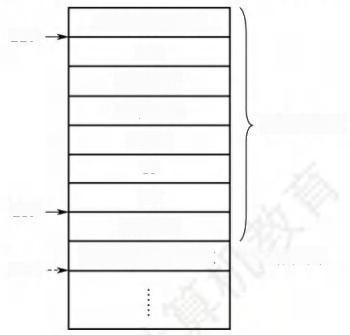
图4 . 12给出了caller 栈帧的状态，假定 caller 被过程 P 调用。执行第4行的指令后， ESP 所 指的位置如图中所示，可以看出GCC 为 caller 的参数分配了24字节的空间。从汇编代码中可以

看 出 ，caller 中只使用了调用者保存寄存器 EAX, 没有使用任何被调用者保存寄存器，因此在 caller 栈帧中无须保存除 EBP 外的任何寄存器的值；caller 有三个局部变量 templ 、temp2 和 sum, 皆 被 分配在栈帧中；在用 call 指令调用 add 函数之前， caller 先将入口参数从右向左依次将 temp2 和 templ 的值(即80和125)保存到栈中。在执行 call 指令时再把返回地址压入栈中。此外，在最 初进入caller 时，还将 EBP 的值压入了栈，因此 caller 的 栈 帧 中 用 到 的 空 间 占 4 + 1 2 + 8 + 4 = 2 8 字节。但是， caller 的栈帧共有4+24+4=32字节，其中浪费了4字节的空间(未使用)。这是 因为 GCC 为保证数据的严格对齐而规定每个函数的栈帧大小必须是16字节的倍数。

call 指令执行后， add 函数的返回参数存放在 EAX 中，因此 call 指令后面的两条指令中，指

令 “mov [ebp-4],eax” 将 add 的结果存入sum 变量的存储空间，该变量的地址为 R[ebp]-4; 指 令

“mov eax,dword ptr [ebp-4]”将 sum 变量的值作为返回值送到寄存器 EAX 中 。



EBP在P中的旧值

EBP

sum

temp2=80

templ=125

caller的栈帧

未使用

80

125

ESP

返回地址

EBP在 caller 中的值

EBP

add栈帧底部

图4.12 caller 和 add 的栈帧

在执行ret 指令之前，应将当前栈帧释放，并恢复旧EBP 的值，上述第14行leave 指令实现

了这个功能， leave 指令功能相当于以下两条指令的功能：

mov esp,ebp

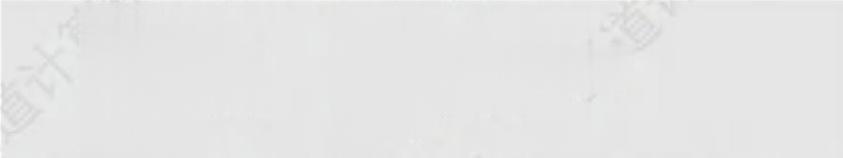
pop ebp

其中，第一条指令使 ESP 指向当前 EBP 的位置，第二条指令执行后， EBP 恢 复 为P 中的旧值， 并使 ESP 指向返回地址。

执行完leave 指 令 后 ，ret 指令就可从ESP 所指处取返回地址，以返回P 执 行。当然，编译器 也可通过pop 指令和对 ESP 的内容做加法来进行退栈操作，而不一定要使用leave 指令。

编译并进行链接后，对应的代码如下所示：

add 过程经 GCC

8048469:55

push ebp

mov ebp,esp

mov eax,dword ptr [ebp+12]

mov edx,dword ptr [ebp+8]

lea eax,[edx+eax]

pop ebp

ret

e5

450c

55 08

04 02

804846a:89

804846c:8b

804846f:8b

8048472:8d

8048475:5d

8048476:c3

通常， 一个过程对应的机器级代码都有三个部分：准备阶段、过程体和结束阶段。

上述第1、2行的指令构成准备阶段的代码段，这是最简单的准备阶段代码段，它通过将当 前栈指针ESP 传 送 到 EBP 来完成将 EBP 指向当前栈帧底部的任务，如图4.12所示，EBP 指 向add 栈帧底部，从而可以方便地通过EBP 获取入口参数。这 里 add 的入口参数x 和 y 对应的值(125 和80)分别在地址为R[ebp]+8 、R[ebp]+12 的存储单元中。

194-2025年计算机组成原理考研复习指导

上述第3、4、5行的指令序列是过程体的代码段，过程体结束时将返回值放在 EAX 中。这 里好像没有加法指令，实际上第5行lea 指令执行的是加法运算 R[edx]+RJeax]=x+y。

上述第6、7行的指令序列是结束阶段的代码段，通过将 EBP 弹出栈帧来恢复 EBP 在 caller 过程中的值，并在栈中退出 add 过程的栈帧，使得执行到 ret 指令时栈顶中已经是返回地址。这 里的返回地址应该是caller 代码中第12行的指令 “mov [ebp-4],eax” 的地址。

add 过程中没有用到任何被调用者保存寄存器，没有局部变量，此外， add 是一个被调用过 程，并且不再调用其他过程，因此也没有入口参数和返回地址要保存，因此，在add 的栈帧中除 了需要保存 EBP, 无须保留其他任何信息。

**4.3.5** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 假设R[ax]=FFE8H,R[bx]=7FE6H, 执行指令 “add ax,bx” 后，寄存器的内容和各标

志的变化为( )。

A.R[ax]=7FCEH,OF=1,SF=0,CF=0,ZF=0

B.R[bx]=7FCEH,OF=1,SF=0,CF=0,ZF=0

C.R[ax]=7FCEH,OF=0,SF=0,CF=1,ZF=0

D.R[bx]=7FCEH,OF=0,SF=0,CF=1,ZF=0

**02.** 假设 R[ax]=7FE6H,R[bx] =FFE8H, 执行指令 “sub bx,ax” 后，寄存器的内存和各标

志的变化为( )。

A.R[ax]=8002H,OF=0,SF=1,CF=1,ZF=0

B.R[bx]=8002H,OF=0,SF=1,CF=0,ZF=0

C.R[ax]=8002H,OF=1,SF=1,CF=0,ZF=0

D.R[bx]=8002H,OF=1,SF=1,CF=0,ZF=0

**03.** 某计算机的数据采用小端方式存储，减法指令“sub ax,imm”的功能为(ax)-imm→ax, imm 表示立即数，该指令对应的十六进机器码为2dxxxx (从左到右以字节为单位由低 地址到高地址),其中xxxx 对应imm 的机器码，若imm=-3,(ax)=7, 则该指令对应 的机器码和执行后OF 标志位的值分别为( )。

A.2DFFFDH,0 B.2DFFFDH,1 C.2DFDFFH,0 D . 2DFDFFH,1

**04.** 某C 语言程序中对数组变量b 的声明为 “int b[10][5];”, 有一条 for 语句如下：

for(i=0;i<10;i++)

for(j=0;j<5;j++)

sum+=b[i][j];

假设执行到“Sum+=b[i][i ];”时 ，sum 的值在 eax 中，b[i[0] 所在的地址在 edx 中 ，j 在

esi中，则“Sum+=b[i][j];”所对应的指令(Intel 格式)可以是( )。

A.add dword ptr eax,[edx+esi\*4] B.add dword ptr eax,[edx+esi\*4]

C.add dword ptr eax,[edx+esi\*2] D.add dword ptr eax,[esi+edx\*2]

**05.** 假设 R[eax]=080480B4H,R[ebx]=00000011H,M[080480F8H]=000000BOH, 执行指

令 “imul eax,[eax+ebx\*4],-16” 后，寄存器或存储单元的内容变为()。 A.R[eax]=00000B00H B.M[080480F8H]=00000BOOH

C.R[eax]=FFFFF500H D.M[080480F8H]=FFFFF500H

**06** . 程序P 中有两个变量i 和 j, 被分别分配在寄存器eax 和 edx 中，P 中 语 句“if(i<j)(…}”

对应的指令序列如下(左边为指令地址，中间为机器代码，右边为汇编指令),其中 jle

指令的偏移量为0d:

804846a 39 c2 cmp dword ptr edx,eax

804846c 7e 0d jle xxxxxxxx

若执行到804846aH 处的cmp 指令时，i=105,j=100, 则 jle 指令执行后将会转到()

处的指令执行。

A.8048461H B.804846eH C.8048479H D.804847bH

**07.** 假定全局数组a 的声明为 double\*a[8],a的首地址为80498cOH, 变 量i被分配在寄存器 ecx 中，现要将a[i]取到 eax 相应宽度的寄存器中，则所用的汇编指令是()。

A.mov eax,[ecx\*4+80498c0H] B.mov eax,ecx\*4+80498cOH

C.mov eax,[ecx\*8+80498c0H] D.mov eax,ecx\*8+80498c0H

**08.** 子程序调用指令的完整功能是( )。

A. 改变堆栈指针 SP 的值

B. 改变程序计数器 PC的值

C. 改变程序计数器 PC的值和堆栈指针 SP 的值

D. 改变地址寄存器的值

**09.** 下列关于选择结构语句“if(comp\_A)then statement\_B;else statement\_C” 对应的机器级代 码表示的叙述中，错误的是( )。

A. 一定包含一条无条件转移指令

B. 一定包含一条条件转移指令(分支指令)

C. 计算comp\_A 的代码段一定在条件转移指令之前

D. 对应 statement\_B 的代码一定在对应 statement\_C 的代码之前

**10.** 下列关于循环结构语句的机器级代码表示的叙述中，错误的是()。

A. 一定至少包含一条条件转移指令

B. 不一定包含无条件转移指令

C. 循环结束条件可以用一条比较指令CMP 来实现

D. 循环体内执行的指令不包含条件转移指令

**11.** 下列有关调用指令(转子指令)的叙述中，错误的是( )。

A. 与高级语言源程序中的过程调用相对应， 一次过程调用对应一条调用指令

B. 指令执行时必须保留返回地址，调用指令随后一条指令的地址是返回地址

C. 嵌套调用时返回地址通常保存在栈中，非嵌套调用时可保存在特定寄存器中

D. 指令执行时将无条件转移到目标地址处，转移目标地址无须在指令中明显给出

**12.** 假设 P 为调用过程，Q 为被调用过程，程序在32位x86 处理器上执行，以下是 C 语言

程序中过程调用所涉及的操作：

① 过程Q 保存P 的现场，并为非静态局部变量分配空间

② 过程 P 将实参存放到 Q 能访问到的地方

③ 过程 P 将返回地址存放到特定处，并转跳到Q 执行

④ 过程Q 取出返回地址，并转跳回到过程P 执行

⑤ 过程Q 恢复P 的现场，并释放局部变量所占空间

⑥ 执行过程Q 的函数体

过程调用的正确执行步骤是()。

1 9 6 - 2 0 2 5 年 计 算 机 组 成 原 理 考 研 复 习 指 导

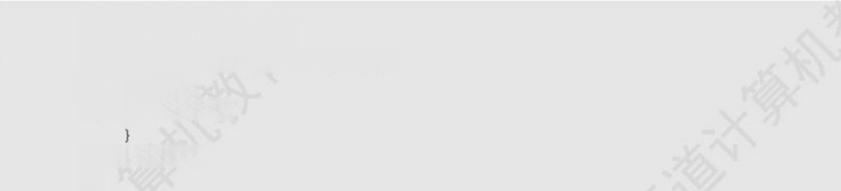
A.②→③→④→①→⑤→⑥ B.②→③→①→④→⑥→⑤

C.②→③→①→( ⑥→⑤→④ D.②→③→①→⑤→⑥→④

**二、** **综合应用题**

**01.**【2017统考真题】在按字节编址的计算机M 上 ，fl 的部分源程序(阴影部分)如下。将

fl 中的 int 都改成 float, 可得到计算f(n) 的另一个函数 f2。

int f1(unsigned n){

r(uns,ipn- 1;i++){

power \*=2;

sum +=power;

return sum;

}

对应的机器级代码(包括指令的虚拟地址)如下：

int fl(unsigned n)

1 00401020 55 push ebp

\*…

for(unsigned i=0;i<=n-1;i++)

20

0040105E

…

394D F4

…

cmp dword ptr[ebp-0Ch],eca

power \*=2;

23 00401066 D1 E2 shl edx,1

return sum;

 …

35 0040107F C3 ret

其中，机器级代码行包括行号、虚拟地址、机器指令和汇编指令。

1)计算机 M 是 RISC 还 是CISC? 为什么?

2)fl 的机器指令代码共占多少字节?要求给出计算过程。

3)第20条指令cmp 通 过i 减 n-1 实现对i 和 n-1 的比较。执行fl(0) 的过程中，当i=0

时 ，cmp 指令执行后，进位/借位标志CF 的内容是什么?要求给出计算过程。

4)第23条指令shl 通过左移操作实现了 power \*2运算，在 f2 中能否用 shl 指令实现

power \*2? 为什么?

**02.**【2019统考真题】已知f(n)=n!=n×(n-1)×(n-2)×…×2×1, 计 算(n) 的 C 语言函数 fl

的源程序(阴影部分)及其在32位计算机 M 上的部分机器级代码如下：



int fl(int n){

1 00401000 55

if (n>1)

11 00401018 837D 0801

12 0040101C 7E 17

return n\*f1(n-1);

13 0040101E 8B 4508

14 00401021 83 E801

15 00401024 50

eax,dword ptr [ebp+8]

eax,1

dword ptr [ebp+8],1

f1+35h(00401035)

mov

sub

push

cmp jle

push

ebp

eax

第4章 **指令系统** **197**



16 00401025 E8 D6 FF FF FF call f1(00401000)

19 00401030 0F AF C1

20 00401033 EB 05

else return 1;

21 00401035 B801000000

}

26 00401040 3B EC

30 0040104A C3

imul eax,ecx

jmp f1+3Ah(0040103a)

mov eax,1

cmp

ret

ebp,esp

其中，机器级代码行包括行号、虚拟地址、机器指令和汇编指令，计算机 M 按 字 节 编 址 ， int型数据占32位。请回答下列问题：

1)计算10)需要调用函数fl 多少次?执行哪条指令会递归调用 fl?

2)上述代码中，哪条指令是条件转移指令?哪几条指令一定会使程序跳转执行?

3)根据第16行的 call 指令，第17行指令的虚拟地址应是多少?已知第16行的 call 指 令 采 用 相 对 寻 址 方 式 ， 该 指 令 中 的 偏 移 量 应 是 多 少 ( 给 出 计 算 过 程 ) ? 已 知 第 1 6 行

的 call 指 令 的 后 4 字 节 为 偏 移 量 ，M 是 采 用 大 端 方 式 还 是 采 用 小 端 方 式 ?

4)(13)=6227020800,但fl(13) 的返回值为1932053504,为什么两者不相等?要使fl(13)

能 返 回 正 确 的 结 果 ， 应 如 何 修 改fl 的 源 程 序 ?

5)第19行的imul 指令(有符号整数乘)的功能是R[eax]—R[eax]×R[ecx],当乘法器输 出的高、低32位乘积之间满足什么条件时，溢出标志OF=1? 要 使CPU 在发生溢出

时 转 异 常 处 理 ， 编 译 器 应 在 imul 指 令 后 加 一 条 什 么 指 令 ?

**03.**【2019统考真题】对于题2,若计算机M 的主存地址为32位，采用分页存储管理方式， 页 大 小 为 4KB, 则 第 1 行 的push 指 令 和 第 3 0 行 的 ret 指 令 是 否 在 同 一 页 中 ( 说 明 理 由 ) ? 若指令 Cache 有64行，采用4路组相联映射方式，主存块大小为64B, 则32位主存地 址中，哪几位表示块内地址?哪几位表示 Cache 组号?哪几位表示标记 (tag) 信 息 ? 读 取 第 1 6 行 的 call指 令 时 ， 只 可 能 在 指 令 Cache 的 哪 一 组 中 命 中 ( 说 明 理 由 ) ?

**04.**【2023统考真题】某C 语言程序段在计算机M 上的部分机器级代码如下，数组a 的定义 为 “int a[24][64];”,每 个 机 器 级 代 码 行 中 依 次 包 含 指 令 序 号 、 虚 拟 地 址 、 机 器 指 令 和 汇 编指令。

for(i=0;i<24;i++)

1 00401072 C745 F800000000

**200401079**

**EB** **09**

8B 55 F8

30040107B

mov [ebp-8].0

**jmp** **00401084h**

mov eax,[ebp-8]

**700401088** **7D** **32** **jge** **004010bch**

for(j=0;j<64;j++)

8 0040108A C745 FC 00000000 mov [ebp-4].0

a[i][j]=10;

**19004010AE** **C78482002042000A** **000000** **mov** **[ecx+edx\*4+00422000h],0Ah**

198-2025年计算机组成原理考研复习指导

请回答下列问题。

1)第20条指令的虚拟地址是多少?

2 ) 已 知 第 2 条jmp 和 第 7 条jge 都是跳转指令，其操作码分别是 EBH 和 7DH, 跳转目 标地址分别为00401084H、004010BCH, 这两条指令都采用什么寻址方式?给出第 2条指令jmp 的跳转目标地址计算过程。

3)已知第19条mov 指令的功能为 “a[i][i]-10”, 其 中ecx 和edx 为寄存器名，00422000H 是数组 a 的首地址，指令中源操作数采用什么寻址方式?已知edx 中存放的是变量j, ecx 中存放的是什么?根据该指令的机器码判断 M 采用的是大端还是小端方式。

4)第一次执行第19条指令时，取指令过程中是否会发生缺页异常?为什么?

**4.3.6** **答** **案** **与** **解** **析**

**一** **、单项选择题**

**01.C**

该指令是Intel 格式， add 指令的目的寄存器为ax 。add 指令的补码加法过程为111111111110 1000+0111111111100110=(1)0111111111001110(7FCEH), 两个操作数的符号不同，必然不会溢 出 ，OF=0; 结果的符号位为0,SF=0; 有进位， CF=C④Sub=1 田 0 = 1 ; 非 0 ,ZF=0。



**注** **意**

无论是无符号数还是有符号数，都以二进制代码形式无差别地存放在计算机内。即便两个 有符号数相加，也会导致 CF 的变动，只是 CF 值对有符号数运算是没有意义的。同理，两个 无符号数相加，也会导致 OF 和 SF 的变动，只是OF 值和 SF 值仅对有符号数运算有意义。

**02.B**

该指令是Intel 格 式 ，sub 指令的目的寄存器为bx 。sub 减法运算用补码加法实现，被减数+ 减数逐位取反+1=1111111111101000+1000000000011001+1=(1)1000000000000010 (8002H), 两个操作数的符号位都是1,结果的符号位也是1,无溢出，OF=0; 结果为负数，SF=1;

进位输出Cu=1, 低位进位Sub=1,CF=Cou 田 Sub=1 田 1 = 0 ; 非 0 , ZF=0。

**03.C**

imm 的值为-3,转换成二进制为1111111111111101B, 即 FFFDH, 因为该计算机采用小端存 储，先存储低位字节，所以该指令对应的机器码为2DFDFFH,OF 是有符号数运算的溢出标志位， 7-(-3)显然没有溢出，因此 OF 标志位为0。

**04.** A

b[i][0] 所在的地址在edx 中 ，j 在 esi 中， 一个数组元素占4字节，所以 b[iJ] 的地址为 R[edx]+ R[esi]\*4, 指令格式为Intel 格式，第一个为目的操作数，第二个为源操作数，于是A 正确。

05.C

指令的一个源操作数在内存单元中，地址为R[eax]+R[ebx]\*4=080480B4H+00000011H\*4 =

080480F8H。 指令的功能是 R[eax]-M[080480F8H]\*(-16)=(-000000B0H)<<4=FFFFFF50H<<4= FFFFF500H 。 目的操作数保存在 eax 中，所以主存单元080480F8H 中的内容不会改变。

**06.D**

i=105,j=100, 即 edx 的内容为100,eax 的内容为105 ,cmp 指令就是对这两个数做减法， 显然100<105,满足jle 指令小于或等于的条件，jle 指令长度为2字节，所以 jle 指令执行后将 转移到当前PC 值 + 偏 移 量 = 8 4 8 4 6cH+2+0dH=804847bH 处执行。

07.C

每个 double 型的数组元素占8字节，数组a 的首地址为80498cOH,i 存储在ecx 中，所以a[i ] 在主存中的地址可表示为[ecx\*8+80498cOH], 因此汇编指令可以是 mov eax,[ecx\*8+80498cOH]。

**08.C**

子程序调用指令需要将程序断点保存至堆栈，这个过程会改变 SP 的值，同时也会改变 PC 的值以跳转到子程序执行。

09.D

在 if 语句的机器级代码中，comp\_A 后面紧接着有一个条件跳移指令，条件成立则转跳到 statement\_B,statement\_B 中有一个无条件跳移指令，会转跳到 if-else 的下一条语句， A 、B 和 C 正确。 statement\_B 不一定在 statement\_C 之前，这取决于条件转移指令的类型和方向， D 错误。

10.D

循环结构循环体内最后会有一条条件转移指令，判断是否跳出循环，可以用比较指令(CMP ) 来实现，A 和 C 正确， D 错 误。循环结构不一定包含无条件转移指令， B 正确。

11.D

为了能保证从被调用过程返回到调用过程继续执行，必须确定并保存返回地址，这个地址是 调用指令随后的指令的地址，返回地址只能由调用指令来计算并保存，因为执行调用指令后就跳 转到了被调用过程，因此无法获取返回地址。为了保证嵌套调用时能够返回到调用过程，必须将 返回地址压栈，若不压栈而保存在特定寄存器中，则后面执行的调用指令会将前面调用指令保存 的返回地址覆盖掉。调用指令执行时将无条件转移到目标地址处，这个目标地址就是被调用过程 第一条指令的地址，它一定在调用指令中明显给出，因此D 错误。

**12.** C

过程调用的具体过程已在4.3.4节中介绍。

**二、** **综合应用题**

**01.** 【解答】

1)M 为 CISC 。M 的指令长短不一，不符合RISC 指令系统的特点。

2)fl 的机器代码占96B。因 为fl 的第一条指令 “pushebp” 所在的虚拟地址为00401020H, 最 后一条指令“ret”所在的虚拟地址为0040107FH, 所以 fl 的机器指令代码长度为0040

107FH-00401020H+1=60H=96B。

3)CF=1 。cmp 指令实现i 与 n-1 的比较功能，进行的是减法运算。在执行fl(O) 的过程中， n=0, 当 i=0 时，i=00000000H, 并 且n-1=FFFFFFFFH 。因此，执行第20条指令时， 在补码加/减运算器中执行“0减FFFFFFFFH” 操作，即00000000H+00000000H+1=0000

0001H, 此时进位输出Cu=0, 低位进位Sub=1,CF=Cou 田 Sub=0④1=1。

4)f2 中不能用shl指令实现 power\*2 。因 为shl指令把一个整数的所有有效数位整体左移， 而 f2 中的变量 power 是 float 型，其机器数中不包含最高有效数位，但包含了阶码部分， 将其作为一个整体左移时并不能实现“乘以2”的功能，因此 f2中不能用 shl 指令实现 power \*2。浮点数运算比整型运算要复杂，耗时也较长。

**02.【**解答】

1)计算/10)需要调用函数fl 共10次，执行第16行的 call 指令会递归调用fl。

2 ) 第 1 2 行 的 jle 指令是条件转移指令，其含义为小于或等于时转移，本行代码的意义为： 当 n≤1 时，跳转至地址00401035H。第16行的 call 指令为函数调用指令，第20行的jmp 指令为无条件转移指令，第30行的 ret 指令为子程序的返回指令，这三条指令一定会使

200-2025年计算机组成原理考研复习指导

程序跳转执行。

3)在计算机M 上按字节编址，第16行的 call 指令的虚拟地址为00401025H, 长度为5字 节，因此第17行的指令的虚拟地址为00401025H+5=0040102AH。 第16行的 call 指 令采用相对寻址方式，即目标地址=(PC)+ 偏移量，call 指令的目标地址为00401000H, 所以偏移量=目标地址- (PC)=00401000H-0040102AH =FFFF FFD6H。根据第16 行 的call 指令的偏移量字段为D6 FF FF FF, 可以确定M 采用小端方式。

4)因为(13)=6227020800,其结果超出了32位int 型数据可表示的最大范围，因此13) 的返回值是一个发生了溢出的错误结果。为使 fl(13) 能返回正确结果，可将函数 fl 的返

回值类型改为 double ( 或 long long, 或 long double, 或 float) 类型。

5)若乘积的高33位不全为0或不全为1,则OF=1 。 编译器应在imul 指令后加一条“溢出

自陷指令”,使得CPU 自动查询溢出标志OF, 当 OF=1 时调出“溢出异常处理程序”。 **03.【**解答】

因为页大小为4KB, 所以虚拟地址的高20位为虚拟页号。第1行的 push 指令和第30行的 ret 指令的虚拟地址的高20位都是00401H, 因此两条指令在同一页中。

指令 Cache 有64块，采用4路组相联映射方式，因此指令 Cache 共有64/4 =16组， Cache 组号共4位。主存块大小为64B, 因此块内地址为低6位。综上所述，在32位主存地址中，低6 位为块内地址，中间4位为组号，高22位为标记。

因为页大小为4KB, 所以虚拟地址和物理地址的最低12位完全相同，因此 call 指令虚拟地 址00401025H 中的025H=000000100101B 为物理地址的低12位，对应的7～10位为组号，因 此对应的 Cache 组号为0。

**04.【**解答】

1)第19条指令的虚拟地址为004010AEH, 且第19条指令占11字节，因此第20条指令的 虚拟地址为004010AEH+11 (十进制)=004010B9H。

2)第2条指令的虚拟地址为00401079H,占2字节，取该指令后， PC+2, 变为0040107AH,

转移指令的目标地址为00401084H, 因此偏移量为00401084H-0040107AH=09H, 根据

第2条指令的机器码可知，09H 恰好是第2条指令给出的偏移量。第7条指令的分析同 理。因此，第2条jmp 和 第 7 条jge 指令都采用相对寻址方式。第2条指令jmp 的跳转目 标地址=00401079H+2 (十进制)+09H=00401084H。

3)第19条指令的源操作数为OAH, 直接在机器指令中 (OA000000) 给出，因此采用立即

(数)寻址方式。数组a 的一行有64个元素，每个元素占4字节，因此a[i][j] 的地址应为 00422000h+i×64×4+j×4=00422000h+i×256+j×4, 根据汇编指令中给出的计算公式ecx+ edx\*4+00422000h 可知， ecx 中存放的是i×256。M 采用小端方式。

4)第一次执行第19条指令时，取指令过程中不会发生缺页异常。因为第19 条指令所在的 该程序段都在页号为00401H 的同一个页面中，执行第19条指令时，该页已在主存，因 此取指令过程中不会发生缺页异常。



**4.4** **CISC 和** **RISC 的基本概念**

指令系统朝两个截然不同的方向的发展； 一是增强原有指令的功能，设置更为复杂的新指令 实现软件功能的硬化，这类机器称为复杂指令系统计算机(CIS C), 典型的有采用x86 架构的计

算机；二是减少指令种类和简化指令功能，提高指令的执行速度，这类机器称为精简指令系统计 算机 (RISC), 典型的有ARM 、MIPS 架构的计算机。

**4.4.1** **复** **杂** **指** **令** **系** **统** **计** **算** **机** **(CISC)**

随着集成电路技术的发展，软件成本不断上升，促使人们在指令系统中增加更多、更复 杂的指令，以适应不同的应用领域，这样就构成了复杂指令系统计算机 (CISC) 。

▶ CISC 的特点(2017)

CISC 的主要特点如下：

1)指令系统复杂庞大，指令数目一般为200条以上。

2)指令的长度不固定，指令格式多，寻址方式多。

3)可以访存的指令不受限制。

4)各种指令使用频度相差很大。

5)各种指令执行时间相差很大，大多数指令需多个时钟周期才能完成。

6)控制器大多数采用微程序控制。有些指令非常复杂，以至于无法采用硬连线控制。 7)难以用优化编译生成高效的目标代码程序。

如此庞大的指令系统，对指令的设计提出了极高的要求，研制周期变得很长。后来人们发现， 一味地追求指令系统的复杂和完备程度不是提高性能的唯一途径。对传统 CISC 指令系统的测试 表明，各种指令的使用频率相差悬殊，大概只有20%的简单指令被反复使用，约占整个程序的80%; 而80%左右的复杂指令则很少使用，约占整个程序的20%。从这一事实出发，人们开始用最常用 的20%的简单指令，重组实现不常用的80%的指令功能， RISC 随之诞生。

**4.4.2** **精** **简** **指** **令** **系** **统** **计** **算** **机** **(RISC)**

精简指令系统计算机(RISC) 的中心思想是要求指令系统简化，尽量使用寄存器-寄存器操 作指令，指令格式力求一致。RISC 的主要特点如下：

1)选取使用频率最高的一些简单指令，复杂指令的功能由简单指令的组合来实现。 2)指令长度固定，指令格式种类少，寻址方式种类少。

3)只有 LOAD/STORE (取数/存数)指令访存，其余指令的操作都在寄存器之间进行。 4)CPU 中通用寄存器的数量相当多。

5)一定采用指令流水线技术，大部分指令在一个时钟周期内完成。

6)以硬布线控制为主，不用或少用微程序控制。

7)特别重视编译优化工作，以减少程序执行时间。

值得注意的是，从指令系统兼容性看， CISC 大多能实现软件兼容，即高档机包含了低档机 的全部指令，并可加以扩充。但 RISC 简化了指令系统，指令条数少，格式也不同于老机器，因 此大多数RISC 机不能与老机器兼容。由于RISC 具有更强的实用性，因此应该是未来处理器的发 展方向。但事实上，当今时代Intel 几乎一统江湖，且早期很多软件都是根据 CISC 设计的，单纯 的 RISC 将无法兼容。此外，现代 CISC 结构的 CPU 已经融合了很多 RISC 的成分，其性能差距 已经越来越小。CISC 可以提供更多的功能，这是程序设计所需要的。

**4.4.3** **CISC** **和** **RISC** **的** **比** **较**

和 CISC 相比， RISC 的优点主要体现在以下几点：

1)RISC 更能充分利用 VLSI (超大规模集成电路)芯片的面积。CISC 采用微程序控制，其

202-2025年计算机组成原理考研复习指导

控制存储器占CPU 芯片面积的50%以上，而RISC 采用组合逻辑控制，其硬布线逻辑只

占CPU 芯片面积的10%左右。

2)RISC 更能提高运算速度。RISC 的指令数、寻址方式和指令格式种类少，又设有多个通 用寄存器，采用流水线技术，所以运算速度更快，大多数指令在一个时钟周期内完成。

3)RISC 便于设计，可降低成本，提高可靠性。RISC 指令系统简单，因此机器设计周期短； 其逻辑简单，出错概率低，有错也易发现，因此可靠性高。

4)RISC 有利于编译程序代码优化。RISC 指令类型少，寻址方式少，使编译程序容易选择 更有效的指令和寻址方式，并适当地调整指令顺序，使得代码执行更高效化。

CISC 和 RISC 的对比见表4.3。

**表4.3** **CISC** **与RISC** **的对比**

|  |  |  |
| --- | --- | --- |
| **类别**  **对比项目** | **CISC** | **RISC** |
| 指令系统 | 复杂，庞大 | 简单，精简 |
| 指令数目 | 一般大于200条 | 一般小于100条 |
| 指令字长 | 不固定 | 定长 |
| 可访存指令 | 不加限制 | 只有LOAD/STORE指令 |
| 各种指令执行时间 | 相差较大 | 绝大多数在一个周期内完成 |
| 各种指令使用频度 | 相差很大 | 都比较常用 |
| 通用寄存器数量 | 较少 | 多 |
| 目标代码 | 难以用优化编译生成高效的目标代码程序 | 采用优化的编译程序，生成代码较为高效 |
| 控制方式 | 绝大多数为微程序控制 | 绝大多数为组合逻辑控制 |
| 指令流水线 | 可以通过一定方式实现 | 必须实现 |

**4.4.4** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 下列关于RISC 的叙述中，正确的是()。

A.RISC 机一定采用流水技术 B. 采用流水技术的机器一定是 RISC 机

C.RISC 机的兼容性优于CISC 机 D.CPU 配备很少的通用寄存器

**02.** 下列描述中，不符合 RISC 指令系统特点的是()。

A. 指令长度固定，指令种类少

B. 寻址方式种类尽量减少，指令功能尽可能强

C. 增加寄存器的数目，以尽量减少访存次数

D. 选取使用频率最高的一些简单指令，以及很有用但不复杂的指令

**03.** 以下有关RISC 的描述中，正确的是()。

A. 为了实现兼容，新设计的 RISC是从原来CISC 系统的指令系统中挑选一部分实现的 B. 采用 RISC技术后，计算机的体系结构又恢复到了早期的情况

C.RISC 的主要目标是减少指令数，因此允许以增加每条指令的功能的方法来减少指令 系统所包含的指令数

D. 以上说法都不对

**04.** 下列关于 RISC 和 CISC 的说法中，不正确的是( )。

A.RISC 指令格式种类少，寻址方式少，指令长度固定，更容易用硬布线电路实现 B.CISC 指令功能强大，寻址方式多，便于汇编程序员编程

C.CISC 指令格式种类多，所以更有利于编译优化

D.RISC 多数指令能够在一个时钟周期内完成，特别适合流水线工作 05. 【2009统考真题】下列关于 RISC 的说法中，错误的是()。

A.RISC 普遍采用微程序控制器

B.RISC 大多数指令在一个时钟周期内完成

C.RISC 的内部通用寄存器数量相对 CISC 多

D.RISC 的指令数、寻址方式和指令格式种类相对 CISC 少

**4.4.5** **答** **案** **与** **解** **析**

**单项选择题**

**01.** A

RISC 必然采用流水线技术，这也是由其指令的特点决定的。而CISC 则无此强制要求，但为 了提高指令执行速度，CISC 也往往采用流水线技术，因此流水线技术并非 RISC 的专利。CISC 机可以兼容很多不同的高级语言和软件，而 RISC 机的指令系统简单精简，只包含一些基本的指 令，这些指令需要通过组合来实现复杂的功能，从而增加了编译器的设计难度和程序员的编程难 度，因此CISC 机的兼容性更好。CPU 配备很多通用寄存器是 RISC 机的主要特点。

**02.B**

A 、C 和 D 都 是 RISC 的特点。对于 B, 寻址方式种类尽量减少是RISC 的特点，而增强指令 的功能则是CISC 的 特 点。RISC 指令功能简单，复杂指令的功能由简单指令的组合来实现。

03.D

RISC 选择一些常用的寄存器型指令，并不是为了兼容CISC,RISC 也不可能兼容 CISC,A

错误。RISC 只是 CPU 的结构发生变化，基本不影响整个计算机的结构，并且即使是采用 RISC 技术的 CPU, 其架构也不可能像早期一样简单， B 错 误 。RISC 的指令功能简单，通过简单指令 的组合来实现复杂指令的功能，C 错误，但 RISC 的主要目标是减少指令数是正确的。

04.C

CISC 指令格式种类多，增大了编译优化的复杂性，因此不利于编译优化。

**05.** A

相对于CISC,RISC 的特点是：指令条数少；指令长度固定，指令格式和寻址种类少；只有 取数/存数指令访问存储器，其余指令的操作均在寄存器之间进行； CPU 中通用寄存器多；大部 分指令在一个时钟周期内完成；以硬布线逻辑为主，不用或少用微程序控制。B 、C 和 D 都 是 RISC 的特点。由于RISC 的速度快，因此普遍采用硬布线控制器， A 错误。



**4.5** **本章小结**

本章开头提出的问题的参考答案如下。

1)什么是指令?什么是指令系统?为什么要引入指令系统?

指令就是要计算机执行某种操作的命令。 一台计算机中所有机器指令的集合，称为这台计算 机的指令系统。引入指令系统后，避免了用户与二进制代码直接接触，使得用户编写程序更为方 便。另外，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。

204-2025年计算机组成原理考研复习指导

2) 一般来说，指令分为哪些部分?每部分有什么用处?

一条指令通常包括操作码字段和地址码字段两部分。其中，操作码指出指令中该指令应该执行 什么性质的操作和具有何种功能，它是识别指令、了解指令功能与区分操作数地址内容的组成和使 用方法等的关键信息。地址码用于给出被操作的信息(指令或数据)的地址，包括参加运算的一个 或多个操作数所在的地址、运算结果的保存地址、程序的转移地址、被调用子程序的入口地址等。

3)对于一个指令系统来说，寻址方式多和少有什么影响?

寻址方式的多样化能让用户编程更为方便，但多重寻址方式会造成 CPU 结构的复杂化(详 见下章),也不利于指令流水线的运行。而寻址方式太少虽然能够提高CPU 的效率，但对于用户 而言，少数几种寻址方式会使编程变得复杂，很难满足用户的需求。



**4.6** **常见问题和易混淆知识点**

1. 简述各常见指令寻址方式的特点和适用情况。

立即寻址操作数获取便捷，通常用于给寄存器赋初值。

直接寻址相对于立即寻址，缩短了指令长度。

间接寻址扩大了寻址范围，便于编制程序，易于完成子程序返回。

寄存器寻址的指令字较短，指令执行速度较快。

寄存器间接寻址扩大了寻址范围。

基址寻址扩大了操作数寻址范围，适用于多道程序设计，常用于为程序或数据分配存储空间。

变址寻址主要用于处理数组问题，适合编制循环程序。

相对寻址用于控制程序的执行顺序、转移等。

基址寻址和变址寻址的区别：两种方式有效地址的形成都是寄存器内容+偏移地址，但是 在基址寻址中，程序员操作的是偏移地址，基址寄存器的内容由操作系统控制，在执行过程中是 动态调整的；而在变址寻址中，程序员操作的是变址寄存器，偏移地址是固定不变的。

2. 一个操作数在内存可能占多个单元，怎样在指令中给出操作数的地址?

现代计算机都采用字节编址方式，即一个内存单元只能存放一字节的信息。 一个操作数(如 char 、int 、float 、double) 可能是8位、16位、32位或64位等，因此可能占用1个、2个、4个 或8个内存单元。也就是说， 一个操作数可能有多个内存地址对应。

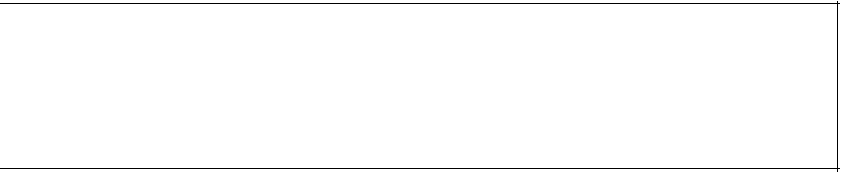
有两种不同的地址指定方式：大端方式和小端方式。

大端方式：指令中给出的地址是操作数最高有效字节 (MSB) 所在的地址。 小端方式：指令中给出的地址是操作数最低有效字节 (LSB) 所在的地址。

3. 装入/存储 (LOAD/STORE) 型指令有什么特点?

装入/存储型指令是用在规整型指令系统中的一种通用寄存器型指令风格。这种指令风格在 RISC 指令系统中较为常见。为了规整指令格式，使指令具有相同的长度，规定只有LOAD/STORE 指令才能访问内存。而运算指令不能直接访问内存，只能从寄存器取数进行运算，运算的结果也 只能送到寄存器。因为寄存器编号较短，而主存地址位数较长，通过某种方式可使运算指令和访 存指令的长度一致。

这种装入/存储型风格的指令系统的最大特点是，指令格式规整，指令长度一致， 一般为32 位。由于只有LOAD/STORE 指令才能访问内存，因此程序中可能包含许多装入指令和存储指令， 与一般通用寄存器型指令风格相比，其程序长度会更长。



第 5 章

中央处理器

**05**

**【考纲内容】**

( 一 )CPU 的功能和基本结构

(二)指令执行过程

(三)数据通路的功能和基本结构

(四)控制器的功能和工作原理

(五)异常和中断机制

异常和中断的基本概念；异常和中断的分类；异常和中断的检测与响应 (六)指令流水线

指令流水线的基本概念；指令流水线的基本实现；

结构冒险、数据冒险和控制冒险的处理；超标量和动态流水线的基本概念 (七)多处理器基本概念

SISD 、SIMD 、MIMD 、 向量处理器的基本概念；硬件多线程的基本概念；

扫一扫



视频讲解

多核 (multi-core)处理器的基本概念；共享内存多处理器 (SMP) 的基本概念

**【复习提示】**

中央处理器是计算机的中心，也是本书的难点。其中，数据通路的分析、指令执行阶段的节 拍与控制信号的安排、流水线技术与性能分析易出综合题。而关于各种寄存器的特点、各种指令 执行的周期与特点、控制器的相关概念、流水线的相关概念易出选择题。

在学习本章时，请读者思考以下问题：

1)指令和数据均存放在内存中，计算机如何从时间和空间上区分它们是指令还是数据?

2)什么是指令周期、机器周期和时钟周期?它们之间有何关系?

3)什么是微指令?它和第4章谈到的指令有什么关系?

4)什么是指令流水线?指令流水线相对于传统体系结构的优势是什么?

请读者在本章的学习过程中寻找答案，本章末尾会给出参考答案。

5.**1** **CPU 的功能和基本结构**

**5.1.1** **CPU** **的功能**

中央处理器 (CPU) 由运算器和控制器组成。其中，控制器的功能是负责协调并控制计算机 各部件执行程序的指令序列；运算器的功能是对数据进行加工。CPU 的具体功能包括：

1)指令控制。完成取指令(也称取指)、分析指令和执行指令的操作，即程序的顺序控制。

206-2025年计算机组成原理考研复习指导

2)操作控制。产生完成一条指令所需的操作信号，把各种操作信号送到相应的部件，从而

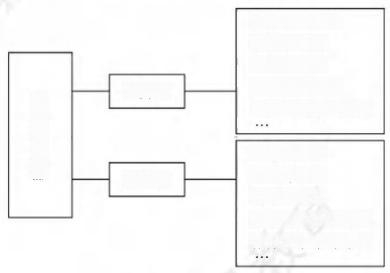
控制这些部件按指令的要求正确执行。

3)时间控制。严格控制各种操作信号的出现时间、持续时间及出现的时间顺序。 4)数据加工。对数据进行算术和逻辑运算。

5)中断处理。对运行过程中出现的异常情况和中断请求进行处理。

**5.1.2** **CPU 的基本结构**

在计算机系统中，CPU 主要由运算器和控制器两大部分组成[也可将CPU 分为数据通路(见 5 . 3节)和控制部件两大组成部分],如图5 . 1所示。



算术逻辑单元

暂存寄存器

累加寄存器

通用寄存器组

程序状态字寄存器

程序计数器

指令寄存器

指令译码器

存储器地址寄存器 存储器数据寄存器

中央处理器

运算器

控制器

图5.1 中央处理器的组成

**1.** **运** **算** **器**

运算器主要由算术逻辑单元(ALU) 、 暂存寄存器、累加寄存器(ACC) 、 通用寄存器组(GPRs)、

程序状态字寄存器 (PSW) 、 移位寄存器、计数器 (CT) 等组成。其主要功能是根据控制器送来 的命令，对数据执行算术运算(加、减、乘、除)、逻辑运算(与、或、非、异或、移位、求补 等)或条件测试(用于设置ZF 、SF 、OF 和 CF 等标志位，作为条件转移的判断条件)。

**2.** **控制器**

控制器主要由程序计数器 (PC )、指令寄存器 (IR) 、 指令译码器 (ID) 、 存储器地址寄存器 (MAR) 、 存储器数据寄存器 (MDR) 、 时序电路和微操作信号发生器等组成。其主要功能是执行 指令，每条指令的执行是由控制器发出的一组微操作实现的。

控制器的工作原理是，根据指令操作码、指令的执行步骤(微命令序列)和条件信号来形成 当前计算机各部件要用到的控制信号。计算机整机各硬件系统在这些控制信号的控制下协同运 行，产生预期的执行结果。控制器是整个系统的指挥中枢，在控制器的控制下，运算器、存储器 和输入/输出设备等功能部件构成一个有机的整体，根据指令的要求指挥全机协调工作。

**5.1.3** **CPU 的寄存器**

命 题 追 踪▶ **汇编程序员可见的寄存器(2010、2015、2021)**

CPU 中的寄存器按汇编语言(或机器语言) 程序是否可访问，可分为两类： 一类是用户可见 寄存器，可对这类寄存器编程，以通过使用这类寄存器减少对主存储器的访问次数，如通用寄存 器组(含基址/变址寄存器)、程序状态字寄存器、程序计数器、累加寄存器、移位寄存器；另一 类是用户不可见寄存器，对用户是透明的，不可对这类寄存器编程，它们被控制部件使用，以控

第5章 中央处理器 207

制CPU 的操作，如存储器地址寄存器、存储器数据寄存器、指令寄存器、暂存寄存器。

命题追踪 ▶ **CP** **U** **中各种寄存器的作用(2013)**

**1.** **运算器中的寄存器**

1)通用寄存器组 (GPRs) 。 用于存放操作数(包括源操作数、目的操作数及中间结果)和各 种地址信息等，如 AX、BX、CX、DX、SP 等。在指令中要指定寄存器的编号，才能明 确是对哪个寄存器进行访问。SP 是堆栈指针，用于指示栈顶的地址。

2)累加寄存器 (AC C) 。 它是一个通用寄存器，用于暂时存放ALU 运算的结果。

3)移位寄存器(S R) 。不但可用来存放操作数，而且在控制信号的作用下，寄存器中的数据 可根据需要向左或向右移位。

4)暂存寄存器。用于暂存从数据总线或通用寄存器送来的操作数，以便在取出下一个操作 数时将其同时送入 ALU。暂存寄存器对应用程序员是透明的(不可见)。

5)程序状态字寄存器(PSW) 。保留由算术/逻辑运算指令或测试指令的运行结果而建立的各 种状态信息，如溢出标志 (OF) 、 符号标志 (SF) 、 零标志 (ZF) 、 进位标志 (CF) 等。 每个标志位通常由一位触发器来保存，这些标志位组合在一起称为程序状态字。

**2.** **控制器中的寄存器**

命题追踪▶> **PC** **和IR** **的位数与主存储器空间和指令字长的关系(2016、2021)**

1)程序计数器 (P C) 。用于指出欲执行指令在主存储器中的存放地址。若 PC 和主存储器均

按字节编址，则PC 的位数等于主存储器地址位数。CPU 根据PC 的内容从主存储器中取 指令，然后送入指令寄存器。指令通常是顺序执行的，因此 PC 具有自动加1的功能(这 里的“1”是指一条指令的字节数);当遇到转移类指令时，PC 的新值由指令计算得到。

2)指令寄存器 (IR) 。 用于保存当前正在执行的指令， IR 的位数等于指令字长。

3)存储器地址寄存器 (MAR) 。 用于存放要访问的主存储器单元的地址，MAR 的位数等于 主存储器地址线数，它反映了最多可寻址的存储单元的个数。

4)存储器数据寄存器 ( MDR) 。用于存放向主存储器写入的信息或从主存储器读出的信息， MDR 的位数等于存储字长。当 CPU 和主存储器交换信息时，都要用到MAR 和 MDR。

**5.1.4** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 下列部件不属于控制器的是( )。

A. 指令寄存器 B. 程序计数器 C. 程序状态字寄存器 D. 时序电路

**02.** 通用寄存器是( )。

A. 可存放指令的寄存器

B. 可存放程序状态字的寄存器

C. 本身具有计数逻辑与移位逻辑的寄存器

D. 可编程指定多种功能的寄存器

**03.** CPU 中保存当前正在执行指令的寄存器是()。

A. 指令寄存器 B. 指令译码器 C. 数据寄存器 D. 地址寄存器

**04.** 在 CPU 中，跟踪后继指令地址的寄存器是( )。

A. 指令寄存器 B. 程序计数器 C. 地址寄存器 D. 状态寄存器

208-2025年计算机组成原理考研复习指导

**05.** 条件转移指令执行时所依据的条件来自()。

A. 指令寄存器 B. 标志寄存器 C. 程序计数器 D. 地址寄存器

**06.** 在所谓的 n 位 CPU 中，n 是 指 ( ) 。

A. 地址总线线数 B. 数据总线线数 C. 控制总线线数 D.I/O 线数

**07.** 在 CPU 的寄存器中，( )对用户是透明的。

A. 程序计数器 B. 状态寄存器 C. 指令寄存器 D. 通用寄存器

**08.** 指令( )从主存储器中读出。

A. 总是根据程序计数器

B. 有时根据程序计数器，有时根据转移指令

C. 根据地址寄存器

D. 有时根据程序计数器，有时根据地址寄存器

**09.** 程序计数器 (PC) 属 于 ( ) 。

A. 运算器 B. 控制器 C. 存储器 D.ALU

**10.** 下面有关程序计数器 (PC) 的叙述中，错误的是()。

A.PC 中总是存放指令地址

B.PC 的 值 由CPU 在执行指令过程中进行修改

C. 执行转移指令时，PC 的值总是修改为转移指令的目标地址

D.PC 的位数一般和存储器地址寄存器 (MAR) 的位数一样

**11.** 程序计数器 (PC) 可以使用字节地址或字地址，其位数取决于( )。

I. 存储器的容量 IⅡ. 机器字长 ⅢI. 指令字长

A.I B.I 和 Ⅲ C.Ⅱ 和 Ⅲ D.I 、Ⅱ 和 Ⅲ

**12.** 下列关于程序计数器 (PC) 的叙述中，错误的是( )。

A. 机器指令中不能显式地使用 PC

B. 指令顺序执行时，PC 值总是自动加1

C. 调用指令执行后， PC 值一定是被调用过程的入口地址

D. 无条件转移指令执行后， PC 值一定是转移目标地址

**13.** 指令寄存器 (IR) 的位数取决于()。

A. 存储器的容量 B. 机器字长 C. 指令字长 D. 存储字长

**14.** CPU 中通用寄存器的位数取决于( )。

A. 存储器的容量 B. 指令的长度 C. 机器字长 D. 都不对

**15.** CPU 中的通用寄存器，()。

A. 只能存放数据，不能存放地址

B. 可以存放数据和地址

C. 既不能存放数据，又不能存放地址

D. 可以存放数据和地址，还可以替代指令寄存器

**16.** 在计算机系统中表示程序和机器运行状态的部件是( )。

A. 程序计数器 B. 累加寄存器 C. 中断寄存器 D. 程序状态字寄存器

**17.** 状态寄存器用来存放( )。

A. 算术运算结果 B. 逻辑运算结果

C. 运算类型 D. 算术、逻辑运算及测试指令的结果状态

**18.** 下列关于标志寄存器(EFLAGS 寄存器或PSW 寄存器)的叙述中，错误的是( )。

第5章 中央处理器 209

A. 不需要像通用寄存器那样，对标志寄存器进行编号

B. 条件转移指令根据其中的一些的标志位来确定PC 的值

C. 可以通过指令直接访问标志寄存器并修改它的值

D. 可以用它来存放执行指令得到的各种标志信息

**19.** 控制器的全部功能是()。

A. 产生时序信号

**B.** 从主存储器中取出指令并完成指令操作码译码

C. 从主存储器中取出指令、分析指令并产生有关的操作控制信号

D. 都不对

**20.** 指令译码是指对()进行译码。

A. 整条指令 B. 指令的操作码字段

C. 指令的地址码字段 D. 指令的地址

**21.** CPU 中不包括()。

A. 存储器地址寄存器 B. 指令寄存器

C. 地址译码器 D. 程序计数器

**22** . 以下关于计算机系统的概念中，正确的是()。

I.CPU 不包括地址译码器

II.CPU 的程序计数器中存放的是操作数地址

.CPU 中决定指令执行顺序的是程序计数器

IV.CPU 的状态寄存器对用户是完全透明的

A.I 、II B.II 、IV C.Ⅱ 、I 、IV D.I 、IⅢ 、IV

**23.** 间址周期结束后， CPU 内寄存器MDR 中的内容为()。

A. 指令 B. 操作数地址 C. 操作数 D. 无法确定

**24.** 一台32位计算机的主存储器容量为4GB, 按字节编址，存储字长和指令字长都是32位。 若指令按字边界对齐存放，则程序计数器 (PC) 的宽度至少是()。

A.32 位 B.30 位 C.8 位 D.34 位

25. 【2010统考真题】下列寄存器中，汇编语言程序员可见的是( )。

A. 存储器地址寄存器 (MAR) B. 程序计数器 (PC)

C. 存储器数据寄存器 (MDR) D. 指令寄存器 (IR)

26.【2016统考真题】某计算机的主存储器空间为4GB, 字长为32位，按字节编址，采用 32位字长指令字格式。若指令按字边界对齐存放，则程序计数器 (PC) 和指令寄存器 (IR) 的位数至少分别是()。

A.30,30 B.30,32 C.32.30 D.32,32

**二、综合应用题**

01.CPU 中有哪些专用寄存器?

**5.1.5** **答案与解析**

**一、单项选择题**

**01.C**

控制器由程序计数器PC、指令寄存器 IR、存储器地址寄存器 MAR、存储器数据寄存器 MDR、

210-2025年计算机组成原理考研复习指导

指令译码器、时序电路和微操作信号发生器等组成。程序状态字寄存器 (PSW) 属于运算器的组 成部分， PSW 包括两个部分： 一是状态标志，如进位标志 (CF) 、 结果为零标志 (ZF ) 等，大多 数指令的执行将会影响到这些标志位；二是控制标志，如中断标志、陷阱标志等。

**02.D**

存放指令的寄存器是指令寄存器，A 错误。存放程序状态字的寄存器是程序状态字寄存器， B 错误。通用寄存器本身并不一定具有计数逻辑和移位逻辑功能，C 错 误。

**03.A**

指令寄存器用于存放当前正在执行的指令。

**04.B**

程序计数器用于存放下一条指令在主存储器中的地址，具有自增功能。

**05.B**

指令寄存器用于存放当前正在执行的指令；程序计数器用于存放下一条指令的地址；地址寄 存器用于暂存指令或数据的地址；程序状态字寄存器用于保存系统的运行状态。条件转移指令执 行时，需要对标志寄存器的内容进行测试，判断是否满足转移条件。

**06.B**

数据总线的位数与处理器的位数相同，它表示CPU 一次能处理的数据的位数，即CPU 的位数。 07.C

指令寄存器中存放当前正在执行的指令，不需要用户的任何干预，所以对用户是透明的。其 他三种寄存器的内容可由程序员指定。

**08.A**

CPU 根据程序计数器 PC 中的内容从主存储器中取指令。读者可能会想到无条件转移指令或 中断返回指令，认为不一定总是根据PC 读出。实际上，当前指令正在执行时，PC 已经是下一条 指令的地址。若遇到无条件转移指令，则只需简单地用跳转地址覆盖原 PC 的内容即可，最终的 结果还是根据PC 从主存储器中读出。地址寄存器用来指出所取数据在主存储器中的地址。

**09.B**

控制器是计算机中处理指令的部件，包含程序计数器。

**10.C**

PC 中存放下一条要执行的指令的地址，A 正确。PC 的值会根据 CPU 在执行指令的过程中修 改(确切地说是在取指周期),或自增，或转移到程序的某处， B 正确。转移指令时，需要判别转 移是否成功，若成功则 PC 修改为转移指令的目标地址，否则下一条指令的地址仍然为 PC 自增后 的 地 址 ，C 错 误 。PC 的位数通常和 MAR 的 位 数 一 样 ，D 正 确 。

11.B

程序计数器(PC) 用于指出下一条指令在主存储器中的地址。可以用字节地址表示指令地址， 此时 PC 的位数与存储器地址的位数相等，而存储器地址的位数取决于存储器的容量；也可以用 字地址表示指令地址，这种情况下指令必须采用按边界对齐的方式存放，此时 PC 的 位 数 = 存 储 器地址的位数 -log₂(指令字长的字节数)。可 知 ，PC 的位数取决于存储器的容量和指令字长。

**12.B**

机器指令中不能显式地使用 PC,PC 的值是自增的，或者是由转移类指令设置的。指令顺序 执行时， PC 自动加1,这里的“1”是指 一 条指令的长度， PC 的值不 一 定总是自动加1,而是根 据指令长度来确定的，具体取决于指令长度占几个编址单位。其余说法均正确。

**13.** C

第5章 中央处理器-211

指令寄存器中保存当前正在执行的指令，所以其位数取决于指令字长。

**14.** C

通用寄存器用于存放操作数和各种地址信息等，其位数与机器字长相等，因此便于操作控制。

**15.B**

通用寄存器供用户自由编程，可以存放数据和地址。而指令寄存器是专门用于存放指令的专 用寄存器，不能由通用寄存器代替。

**16.D**

程序状态字寄存器用于存放程序状态字，而程序状态字的各位表征程序和机器的运行状态， 如含有进位标志 (CF) 、 结果为零标志 (ZF) 等 。

17.D

程序状态字寄存器用于保留算术、逻辑运算及测试指令的结果状态。

**18.** C

标志寄存器是专用寄存器，不需要编号，也不能在指令中直接指定编号来访问；标志寄存器 中的内容是执行指令的过程中， CPU 根据指令执行的结果生成的各种标志信息，用户不能直接修 改它的值。标志寄存器中的标志位主要用于条件转移或条件设置类指令的条件判断。

19.C

控制器的功能是取指令、分析指令和执行指令，执行指令就是发出有关操作控制信号。

**20.B**

指令包括操作码字段和地址码字段，但指令译码器仅对操作码字段进行译码，借以确定指令 的操作功能。

**21.C**

地址译码器是主存等存储器的组成部分，其作用是根据输入的地址码唯一选定一个存储单 元，它不是CPU 的组成部分。而 MAR、IR、PC 都是 CPU 的组成部分。

**22.** A

地址译码器位于存储器， I 正确；程序计数器中存放的是欲执行指令的地址，Ⅱ错误；程序

计数器决定程序的执行顺序，Ⅲ正确；程序状态字寄存器对用户不透明，IV 错误。

**23.B**

间址周期的作用是取操作数的有效地址，因此，间址周期结束后，MDR 中的内容为操作 数地址。

**24.B**

计算机按字节编址，指令字长为32位，占4字节，指令按字边界对齐方式存放，则指令存 放的起始地址必须是4字节的整数倍，4GB/4B=230, 故 PC 的宽度至少是30位。

**25.B**

汇编语言程序员可见的是程序计数器 (PC), 即汇编语言程序员通过汇编程序可以对某个寄 存器进行访问。汇编程序员可以通过指定待执行指令的地址来设置 PC 的值，如转移指令、子程 序调用指令等。而IR、MAR、MDR 是 CPU 的内部工作寄存器，对程序员不可见。

**26.B**

程序计数器 (PC) 用于指出下一条指令在内存中的地址，虽然可以用32位的地址来表示指 令地址，但实际上内存中最多只能存放4GB/32 位=2°条指令，故可以用30位的字地址来表示 指令地址，这种情况下指令必须采用按边界对齐的方式存放，故PC 的位数至少是30位，即 PC 给出的地址是字地址。题干已说明指令按字边界对齐的方式存放，也就是说，指令地址都是4字

212-2025年计算机组成原理考研复习指导

节的整数倍，因此为了让PC 的位数最少，可以采用字地址，取指令时将PC 值 左 移 2 位 到 主 存 中 取指令。指令寄存器 (IR) 用于存放从内存中取出的指令，它取决于指令字长，故 IR 的 位 数 至 少 是 3 2 位 。

**二** **、综合应用题**

01 . 【解答】

CPU 中的专用寄存器有程序计数器 (PC) 、 指 令 寄 存 器 (IR) 、 存储器数据寄存器 (MDR)、

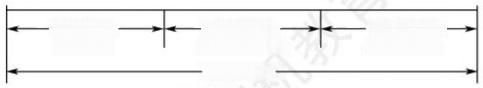
存储器地址寄存器 (MAR) 和程序状态字寄存器 (PSW)。



**5.2** **指令执行过程**

**5.2.1** **指令周期**

CPU 每取出并执行 一 条指令所需的全部时间称为指令周期，不同指令的指令周期可能不同。 指令周期通常可用若干机器周期来表示，每个指令周期内的机器周期数可以不等。图5 . 2反映了 上述关系。图5 . 2(a) 所示为定长的机器周期，图5 . 2(b) 所示为不定长的机器周期。



-间址周期-

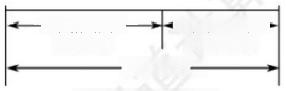
(取有效地址)

-指令周期一

(a)定长的机器周期

执行周期一 (执行指令)

取指周期- (取指令)



←执行周期→ (执行指令)

-指令周期—

(b)不定长的机器周期

一取指周期一 (取指令)

图5.2 指令周期和机器周期的关系

对于无条件转移指令JMPX, 在执行时不需要访问主存，只包含取指阶段(包括取指和分析) 和执行阶段，所以其指令周期仅包含取指周期和执行周期。

对于间接寻址的指令，为了取操作数，需要先访问 一 次主存，取出有效地址，然后访问主存，

取出操作数，所以还需包括间址周期。间址周期介于取指周期和执行周期之间。

当 CPU 采用中断方式实现主机和 I/O 设 备 的 信 息 交 换 时 ，CPU 在每条指令执行结束前，都 要发中断查询信号，若有中断请求，则CPU 进入中断响应阶段，也称中 断 周 期。这样， 一 个完整

的指令周期可包括取指、间址、执行和中断4个周期，如图5 . 3所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 取指周期 | 间址周期 | 执行周期 | 中断周期 |
|  |  |  |  |
| - 指 令 周 期 — | | | |

图5.3 带有间址周期、中断周期的指令周期

命 题 追 踪 ▶ **指令执行的过程(2011)**

当 CPU 执行指令时，首先进入取指周期，从PC 指出的主存单元中取出指令，送至指令寄

存器，同时 PC 加“1”以作为下 一 条指令的地址。当遇到转移指令等改变执行顺序的指令时，

地址总线

数据总线

控制总线

**第5章** **中央处理器** -213

在 PC 加“1”后会重新计算并更新 PC 值。然后判断是否有间接寻址，如果有，那么进入间址 周期以获取操作数的有效地址。之后进入执行周期，完成取操作数、执行运算和存操作数的任 务。执行周期结束后，如果CPU 检测到中断请求，则进入中断周期，此时需要关中断、保存断 点、修改 PC 值为中断服务程序的入口地址，并转向中断服务程序。关于中断的具体内容，见 本章的5.5节。



**注** **意**

中断周期中的进栈操作是将 SP 减“1”,这和传统意义上的进栈操作相反，原因是计算机 中的堆栈都是向低地址方向增长，所以进栈操作是减“1”而不是加“1”。

**5.2.2** **指令周期的数据流**

数据流是根据指令要求依次访问的数据序列。在指令执行的不同阶段，要求依次访问的数据 序列是不同的。而且对于不同的指令，它们的数据流往往也是不同的。

**1.** **取指周期**

取指周期的任务是根据 PC 中的内容从主存中取出指令代码并存放在IR 中 。

取指周期的数据流如图5.4所示。PC 中存放的是指令的地址，根据此地址从内存单元中取出 的是指令，并放在指令寄存器IR 中，取指令的同时， PC 加 1 。

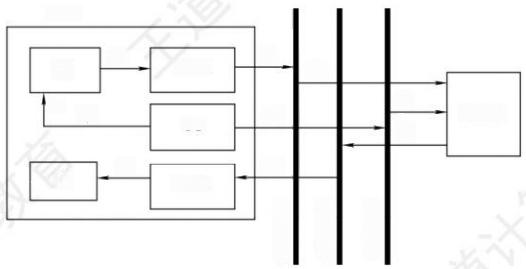
取指周期的数据流向如下：

1)PC①MAR② 地址总线③存储器。

2)CU 发出读命令④控制总线⑤存储器。

3 ) 主 存 @ 数 据 总 线 ②MDR③IR (存放指令)。

4)CU 发出控制信号⑨PC 内容加1。



CPU

PC

+1 ⑨

⑥

⑧

IR

④

⑦

存储器

MDR

MAR

②

③

⑤

①

CU

图5.4 取指周期的数据流

**2.** **间址周期**

间址周期的任务是取操作数有效地址。以一次间址为例(见图5.5),将指令中的地址码送到 MAR 并送至地址总线，此后CU 向存储器发出读命令，以获取有效地址并存至MDR。

间址周期的数据流向如下：

1)Ad(IR) ( 或 MDR)①MAR② 地址总线③存储器。

2)CU 发出读命令④控制总线⑤存储器。

3 ) 主 存 @ 数 据 总 线 ②MDR (存放有效地址)。

其 中 ，Ad(IR) 表 示 取 出IR 中存放的指令字的地址字段。

地址总线

数据总线

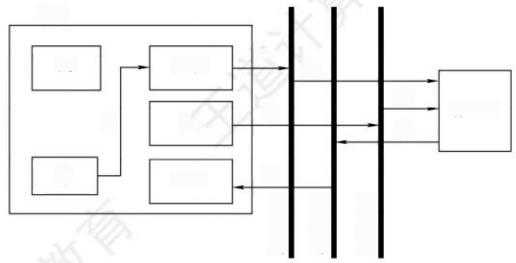
控制总线

地址总线

数据总线

控制总线

214-2025年计算机组成原理考研复习指导



CPU

PC

⑤

CU

IR

④

⑦

存储器

MDR

MAR

②

③

⑥

①

图5.5 一次间址周期的数据流

**3.** **执** **行** **周** **期**

执行周期的任务是取操作数，并根据IR 中的指令字的操作码通过ALU 操作产生执行结果。

不同指令的执行周期操作不同，因此没有统 一 的数据流向。

**4.** **中** **断** **周** **期**

中断周期的任务是处理中断请求。假设程序断点存入堆栈中，并用 SP 指示栈顶地址，而且 进栈操作是先修改栈顶指针，后存入数据，数据流如图5 . 6所示。

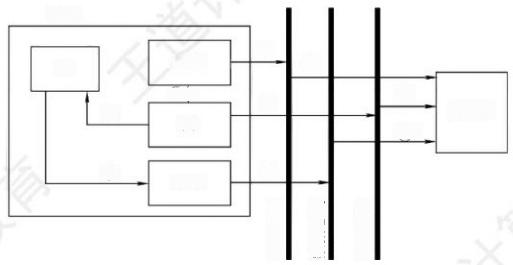
中断周期的数据流向如下：

1)CU 控 制 将SP 减 1 , SP①MAR② 地址总线③存储器。

2)CU 发出写命令④控制总线⑤存储器。

3)PC⑥MDR② 数据总线②主存(程序断点存入存储器)。

4)CU ( 中 断 服 务 程 序 的 入 口 地 址 ) ⑨PC。



CPU

PC

①

⑨

CU

⑥

⑤

⑧

存储器

MAR

MDR

②

③

④

⑦

图5.6 中断周期的数据流

**5.2.3** **指令执行方案**

一 个指令周期通常要包括几个执行步骤，每个步骤完成指令的 一 部分功能，几个依次执行的

步骤完成这条指令的全部功能。不同的处理器采用不同的方案来安排指令的执行步骤。

命 题 追 踪 ▶ **单周期和多周期** **CP** **U** **的** **CPI(2020** **)**

**1.** **单周期处理器**

命 题 追 踪▶ **单周期CPU** **的特点(2016)**

单周期处理器对所有指令都选用相同的执行时间来完成。此时每条指令都在 一 个时钟周期内

**第5章** **中央处理器** - 215

完成(即 CPI=1), 指令之间串行执行，即下一条指令只能在前一条指令执行结束后才能启动。 因此，指令周期取决于执行时间最长的指令的执行时间。对于那些本来可以在更短时间内完成的 指令，仍要使用这个较长的周期来完成，会降低整个系统的运行速度。

**2.** **多周期处理器**

多周期处理器对不同类型的指令选用不同的执行步骤。指令需要几个周期就为其分配几个周 期，因此可选用不同个数的时钟周期来完成不同指令的执行过程(即CPI>1), 不再要求所有指 令占用相同的执行时间。多指令周期方案中指令之间仍是串行执行。

**3.** **流水线处理器**

流水线处理器采用指令之间并行执行的方案，其追求的目标是力争在每个时钟周期完成一条 指令的执行过程(只在理想情况下才能达到该效果，此时CPI=1) 。 这种方案通过在每个时钟周 期启动一条指令，尽量让多条指令同时运行，但各自处在不同的执行步骤中。

**5.2.4** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 计算机工作的最小时间周期是()。

A. 时钟周期 B. 指令周期 C.CPU 周期 D. 总线周期

02. 采用 DMA 方式传递数据时，每传送一个数据就要占用( )。

A. 指令周期 B. 时钟周期 C. 机器周期 D. 存取周期

**03.** 指令周期是指( )。

A.CPU 从主存取出一条指令的时间

B.CPU 执行一条指令的时间

C.CPU 从主存取出一条指令加上执行这条指令的时间

D. 时钟周期时间

**04.** 在一条无条件跳转指令的指令周期内，程序计数器 (PC) 的值被修改了( )次。

A.1 B.2 C.3 D. 不能确定

**05.** 取指操作后，程序计数器中存放的是( )。

A. 当前指令的地址 B. 程序中指令的数量

C. 已执行的指令数量 D. 下一条指令的地址

**06.** 下列关于指令执行的叙述中，错误的是()。

A. 指令周期的第一个操作是取指令

B. 为了进行取指操作，控制器需要得到相应的指令

C. 取指操作是控制器自动进行的

D. 指令执行时有些操作是相同或相似的

**07.** 下列关于指令执行过程的叙述中，错误的是()。

**A**. 取指操作是控制器固有的功能，不需要在操作码控制下完成

**B. 所有指令的取指操作是相同的**

C. 在指令长度相同的情况下，所有指令的取指操作是相同的

D. 中断周期是在指令执行完成后出现的

**08.** 指令周期由一个到几个机器周期组成，第一个机器周期是()。

A. 从主存中取出指令字 B. 从主存中取出指令操作码

C. 从主存中取出指令地址码 D. 从主存中取出指令的地址

216-2025年计算机组成原理考研复习指导

**09.** 由于CPU 内部操作的速度较快，而CPU 访问一次存储器的时间较长，因此机器周期通 常由( )来确定。

A. 指令周期 B. 存取周期 C. 间址周期 D. 中断周期

**10.** 下列有关机器周期的叙述中，错误的是( )。

A. 通常把通过一次总线事务访问一次主存或I/O 的时间定为一个机器周期 B. 一个指令周期通常包含多个机器周期

C. 不同的指令周期所包含的机器周期数可能不同

D. 每个指令周期都包含一个中断响应机器周期

**11.** 下列关于多周期 CPU 的说法中，合理的是( )。

A. 执行各条指令的机器周期数相同，各机器周期的长度均匀

B. 执行各条指令的机器周期数相同，各机器周期的长度可变

C. 执行各条指令的机器周期数可变，各机器周期的长度均匀

D. 执行各条指令的机器周期数可变，各机器周期的长度可变

**12.** 以下关于间址周期的描述中，正确的是( )。

A. 所有指令的间址操作都是相同的

B. 凡是存储器间接寻址的指令，它们的操作都是相同的

C. 对于存储器间接寻址和寄存器间接寻址，它们的操作是不同的

D. 都不对

**13.** ()可区分存储单元中存放的是指令还是数据。

A. 控制器 B. 运算器 C. 存储器 D. 数据通路

**14.** 下列关于各种字长的说法中，正确的是()。

I. 指令字长等于机器字长的前提下，取指周期等于机器周期

I. 指令字长等于存储字长的前提下，取指周期等于机器周期

Ⅲ.指令字长和机器字长的长度没有任何关系

IV. 为了硬件设计方便，指令字长都和存储字长一样大

A.Ⅱ 、I B.Ⅱ 、Ⅲ 、IV C.I 、Ⅲ 、IV D.I 、IV

**15.** 下列关于单周期CPU 和多周期CPU 的描述中，错误的是( )。

A. 执行任何指令，单周期CPU 的时间都要小于多周期CPU

B. 单周期 CPU 部件冗余大，时间利用率低，多周期 CPU 则刚好相反 C. 单周期CPU 在1个时钟周期内执行一条指令， CPI=1

D. 多周期CPU 至少需要2个时钟周期才能执行一条指令，CPI>1

**16.【**2009统考真题】冯·诺依曼计算机中指令和数据均以二进制形式存放在存储器中，CPU 区分它们的依据是( )。

A. 指令操作码的译码结果 B. 指令和数据的寻址方式

C. 指令周期的不同阶段 D. 指令和数据所在的存储单元

**17.【2** 011统考真题】假定不采用 Cache 和指令预取技术，且机器处于“开中断”状态，则 在下列有关指令执行的叙述中，错误的是()。

A. 每个指令周期中 CPU 都至少访存一次

B. 每个指令周期一定大于或等于一个 CPU 时钟周期

C. 空操作指令的指令周期中任何寄存器的内容都不会被改变

D. 当前程序在每条指令执行结束时都可能被外部中断打断

第5章 中央处理器 217

**5.2.5** **答案与解析**

**单项选择题**

**01.A**

时钟周期是计算机操作的最小时间单位，为主频的倒数。指令周期则可由多个CPU 周 期 ( 即

机器周期)组成。总线周期是指一次总线操作所需的时间，通常为一个或多个时钟周期。

**02.D**

CPU 从主存中每取出并执行一条指令所需的全部时间称为指令周期；时钟周期通常称为节 拍 或 T 周期，它是 CPU 操作的最基本单位；指令周期可分为若干机器周期；存取周期是指存储器进 行两次独立的存储操作(连续两次读或写操作)所需的最小间隔时间。

**03.C**

指令周期是指CPU 从主存取出一条指令加上执行这条指令的时间，间址周期不是必需的。

**04.B**

首先在取指周期结束后，PC 值自动加1;在执行周期中， PC 值修改为要跳转到的地址。综 上，在一条无条件跳转指令的指令周期内，程序计数器 (PC) 的值被修改了2次。

05.D

在取指操作后，程序计数器中的内容将被修改为下一条指令的地址，而不是当前指令的地址。

**06.B**

取指操作是自动进行的，控制器不需要得到相应的指令。

**07.B**

不同长度的指令，其取指操作可能是不同的。例如，双字指令、三字指令与单字指令的取指 操作是不同的。

**08.A**

指令周期的第一个机器周期是取指周期，即从主存中取出指令字的时间。

**09.B**

存储器进行一次读或写操作所需的时间称为存储器的读/写时间，而连续启动两次独立的读 或写操作(如连续两次读操作)所需的最短时间称为存取周期。机器周期通常由存取周期确定。

**10.D**

在指令的执行周期完成后，处理器会判断是否出现中断请求，只有在出现中断请求时才会进

入中断周期。

**11.** D

机器周期是指令执行中每步操作(如取指令、存储器读、存储器写等)所需要的时间，每个 机器周期的长度可变。各种指令的功能不同，因此各指令执行所需的机器周期数是可变的。

**12.C**

指令的间址分为 一 次间址、两次间址和多次间址，因此它们的操作是不同的，A 、B 错 误 。

存储器间址通过形式地址访存，寄存器间址通过寄存器内容访存， C 正确。

**13.A**

存储器本身无法区分存储单元中存放的是指令还是数据。而在控制器的控制下，计算机在不同 的阶段对存储器进行读/写操作时，取出的代码也就有不同的用处。而在取指阶段读出的二进制代码 为指令，在执行阶段读出的二进制代码则可能为数据；运算器和数据通路显然不能区分。

218-2025年计算机组成原理考研复习指导

**14.A**

指令字长一般都取存储字长的整数倍，若指令字长等于存储字长的2倍，则需要两次访存， 取指周期等于机器周期的2倍；若指令字长等于存储字长，则取指周期等于机器周期，因此I 错 。 根据 I 的分析可知，Ⅱ正确。指令字长取决于操作码的长度、操作数地址的长度和操作数地址的 个数，与机器字长没有必然的联系。但为了硬件设计方便，指令字长一般取字节或存储字长的整 数倍，因此Ⅲ正确。根据Ⅲ的分析可知，指令字长一般取字节或存储字长的整数倍，而不一 定 都和存储字长一样大，因此IV 错误。综上所述，Ⅱ、Ⅲ正确。

**15.A**

单周期CPU 执行任何指令的时间不一定都会小于多周期CPU, 这取决于单周期 CPU 和多周 期 CPU 的时钟周期的长短，以及该指令在多周期CPU 下所需的时钟周期数，故 A 错误。

**16.C**

虽然指令和数据都以二进制形式存放在存储器中，但 CPU 可以根据指令周期的不同阶段来 区分是指令还是数据，通常在取指阶段取出的是指令，在执行阶段取出的是数据。本题容易误选 选项A, 需要清楚的是，CPU 只有在确定取出的是指令后，才会将其操作码送去译码，因此不可 能依据译码的结果来区分指令和数据。

17.C

由于不采用指令预取技术，每个指令周期都需要取指令，而不采用 Cache 技术，因此每次取 指令都至少要访存一次(当指令字长与存储字长相等且按边界对齐时),A 正 确。时钟周期是 CPU 的最小时间单位，每个指令周期一定大于或等于一个 CPU 时钟周期， B 正确。即使是空操作指令， 在取指操作结束后， PC 也会自动加1,C 错误。由于机器处于“开中断”状态，在每条指令执行 结束时都可能被外部中断打断。



**5.3数据通路的功能和基本结构**

**5.3.1** **数据通路的功能**

随着技术的发展，更多的功能逻辑被集成到CPU 芯片中，但不论 CPU 的内部结构多么复杂， 它都可视为由数据通路 (Data Path) 和控制部件 ( Control Unit) 两大部分组成。

▶ **数据通路的组成部件(2017、2021)**

数据在指令执行过程中所经过的路径，包括路径上的部件，称为数据通路，ALU 、 通用寄存 器、状态寄存器、异常和中断处理逻辑等都是指令执行时数据流经的部件，都属于数据通路的一 部分。数据通路描述了信息从哪里开始，中间经过哪些部件，最后被传送到哪里。数据通路由控 制部件控制，控制部件根据每条指令功能的不同，生成对数据通路的控制信号。

**5.3.2** **数据通路的组成**

组成数据通路的元件主要分为组合逻辑元件和时序逻辑元件两类。

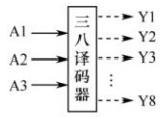
命 题 追 踪▶》 **数据通路中的组合逻辑元件和时序逻辑元件(2021、2023**)

**1.** **组合逻辑元件(操作元件)**

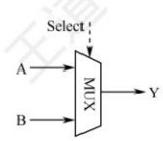
任何时刻产生的输出仅取决于当前的输入。组合电路不含存储信号的记忆单元，也不受时钟

第5章 中央处理器-219

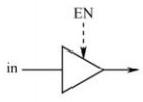
信号的控制，输出与输入之间无反馈通路，信号是单向传输的。数据通路中常用的组合逻辑元件 有加法器、算术逻辑单元 (ALU) 、 译码器、多路选择器、三态门等，如图5.7所示。



(a) 译码器



(b)多路选择器



(c)三态门

图5.7 数据通路中的几种常用组合逻辑元件

图中虚线表示控制信号，译码器可用于操作码或地址码译码， n 位输入对应2"种不同组 合，因此有2”个不同输出。多路选择器 (MUX) 需要控制信号 Select 来确定选择哪个输入被 输出。三态门可视为一种控制开关，由控制信号EN 决定信号线的通断，当EN =1时，三态 门被打开，输出信号等于输入信号；当EN =0时，输出端呈高阻态(隔断态),所连寄存器 与总线断开。

2. 时序逻辑元件(状态元件)

任何时刻的输出不仅与该时刻的输入有关，还与该时刻以前的输入有关，因而时序电路必然 包含存储信号的记忆单元。此外，时序电路必须在时钟节拍下工作。各类寄存器和存储器，如通 用寄存器组、程序计数器、状态/移位/暂存/锁存寄存器等，都属于时序逻辑元件。

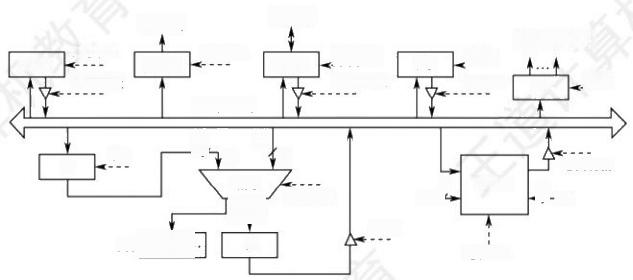
5.3.3 数 据 通 路 的 基 本 结 构

数据通路的基本结构主要有以下几种。

1.CPU 内部单总线方式

命题追踪▶数据通路中的部件及连接方式(2013、2015、2022)

将 ALU 及所有寄存器都连接到一条内部公共总线上，称为单总线结构的数据通路。这种结 构比较简单，但数据传输存在较多的冲突现象，性能较低。此总线在 CPU 内部，注意不要把它与 连接 CPU 、存储器和外设的系统总线相混淆。图5.8所示为单总线的数据通路和控制信号。



数据线

送CU等部件

MDR

IR IRin

B

GPRout

ALUop

Zout

GPRin

地址线

ACCin MARin

ACC MAR

ACCoul

内部总线

*Y* 16, 16

Y

ALU

I6/F

FR Sror: … Z

MDRin

MDRout

*in* A

PCin

PCout

4 rd≠

GPRs

4

PC

rs

图5.8 单总线的数据通路和控制信号

命题追踪▶数据通路中的三态门及其作用(2015)

在图5.8中，GPRs 为通用寄存器组， rs 、rd 分别为所读、写的通用寄存器的编号； Y 和Z 为



220-2025年计算机组成原理考研复习指导

暂存器； FR 为标志寄存器，用于存放ALU 产生的标志信息。带箭头的虚线表示控制信号，字母 加“in”表示该部件允许写入，字母加“out”表示该部件允许输出。MDRin 表示内部总线上信息 写入MDR,MDRout 表示 MDR 的内容送入内部总线。能输出到总线的部件均通过一个三态门与 内部总线相连，用于控制该部件与内部总线之间数据通路的连接与断开。



**注** **意**

单周期处理器 (CPI=1) 不能采用单总线方式，因为单总线将所有寄存器都连接到 一

条公共总线上， 一个时钟内只允许 一 次操作，无法完成 一 条指令的所有操作。

**2.CPU 内部多总线方式**

CPU 内部有两条或更多的总线时，构成双总线结构或多总线结构。将所有寄存器的输入端和 输出端都连接到多条公共通路上，相比之下单总线中一个时钟内只允许传送一个数据，因而指令 执行效率很低，因此采用多总线方式，同时在多个总线上传送不同的数据，提高效率。

命 题 追 踪▶ **单周期CPU** **的特点(2016)**

**3.** **专用数据通路方式**

根据指令执行过程中的数据和地址的流动方向安排连接电路，避免使用共享的总线，性能较 高，但硬件量大。



**注** **意**

内部总线是指同 一 部件，如CPU 内部连接各寄存器及运算部件之间的总线；系统总线 是指同 一 台计算机系统的各部件，如CPU 、 内存和各类I/O 接口间互相连接的总线。

**5.3.4** **数** **据** **通** **路** **的** **操** **作** **举** **例**

总线是一组共享的传输信号线，它不能存储信息，任一时刻也只能有一个部件把信息送到总 线上。下面以图5.8所示的单总线数据通路为例，介绍一些常见操作的流程及控制信号。

命 题 追 踪 》 **指令执行的节拍及有效控制信号(2009、2015);指令在取数和执行阶段所用到的**

部件(2019)

**1.** **通用寄存器之间传送数据**

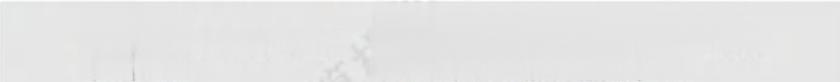
在寄存器和总线之间有两个控制信号： Rin 和 Rout 。当 Rin 有效时，控制将总线上的信息存 到寄存器 R 中；当Rout 有效时，控制将寄存器R 的内容送至总线。下面以程序计数器PC 为例， 将 PC 的内容送至MAR 。 实现该操作的流程及控制信号为

(PC)→MAR PCout 和MARin 有效，PC 内容→MAR

**2,从主存读取数据**

命 题 追 踪 ▶ **取指令阶段所需时钟周期分析(2022)**

从主存中读取的信息可能是数据或指令，现以 CPU 从主存中取指令为例，说明数据在单总 线数据通路中的传送过程。实现该操作的流程及控制信号为



PCout 和MARin 有效，现行指令地址→MAR

MDRin有效， cu 发出读命令，取出指令后 PC+1 MDRout 和 IRin 有效，现行指令→IR

(PC)→MAR

MEM(MAR ) →MDR,(PC)+1→PC (MDR)→IR

第5章 中央处理器 221

第 一 步，将 PC 的内容通过内部总线送至 MAR, 需 要 1 个 时 钟 周 期 。 第 二 步 ，CU 向 主 存发出读命令，从 MAR 所指主存单元读取 一 个字，并送至 MDR; 同 时 PC 加 1 为 取 下 一 条 指令做准备，需要1个主存周期。第三步，将 MDR 的内容通过内部总线送至IR, 需 要 1 个 时钟周期。

**3.** **将数据写入主存**

将寄存器 R1的内容写入寄存器 R2 所指的主存单元，完成该操作的流程及控制信号为

(R1)→MDR Rlout 和 MDRin 有效

(R2)→MAR R2out 和 MARin 有效

MDR→MEM(MAR) MDRout 有效， cu 发出写命令

**4.** **执行算术或逻辑运算**

▶ **ALU 中设置暂存器的原因(2015、2022)**

在单总线数据通路中，每 一 时刻总线上只有 一个数据有效。由于 ALU 是 一个没有存储功 能的组合逻辑元件，在其执行运算时必须保持两个输入端同时有效，因此先将一个操作数经内 部总线送入暂存器Y 保 存 ，Y 的 内 容 在ALU 的左输入端始终有效，再将另 一个操作数经内部 总线直接送到ALU 的右输入端。此外， ALU 的输出端也不能直接与总线相连，否则其输出会 通过总线反馈到输入端，影响运算结果，因此将运算结果暂存在暂存器 Z 中。假设加法指令

ADD ACC,R1, 实 现 将ACC 的 内 容 和R1 的内容相加并写回ACC, 完成该操作的流程及控制

信号为

R1out 和 Yin 有效，操作数→Y

(R1)→Y

(ACC)+(Y )→Z

(Z)→ACC

ACCout 和 ALUin 有效， cu 向 ALU 发出加命令，结果→z Zout 和 ACCin 有效，结果→ACC

**分析减法和自增指令执行所需的时钟周期数(2015)**

以上3步不能同时执行，否则会引起总线冲突，因此该操作需要3个时钟周期。

**5.** **修改程序计数器的值**

转移指令通过修改程序计数器 PC 的值来达到转跳的目的。假设转移指令JMPaddr,addr 为

目标转移地址，实现将 IR 中的地址字段写入PC, 完成该操作的流程及控制信号为

Ad(IR)→PC IRout 和PCin 有效

数据通路结构直接影响 CPU 内各种信息的传送路径，数据通路不同，指令执行过程的微操 作序列的安排也不同，它关系着微操作信号形成部件的设计。

**5.3.5** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 下列不属于 CPU 数据通路结构的是().

A. 单总线结构 B. 多总线结构

C. 部件内总线结构 D. 专用数据通路结构

**02.** 下列有关数据通路的叙述中，错误的是( )。

A. 数据通路由若干组合逻辑元件和时序逻辑元件连接而成

B. 数据通路的功能由控制部件送出的控制信号决定

C.ALU 属于操作元件，用于执行各类算术和逻辑运算

D. 通用寄存器属于状态元件，但不包含在数据通路中

222-2025年计算机组成原理考研复习指导

**03.** 在单总线的 CPU 中，( )。

A.ALU 的两个输入端及输出端都可与总线相连

B.ALU 的两个输入端可以与总线相连，但输出端需通过暂存器与总线相连

C.ALU 的一个输入端可以与总线相连，其输出端也可与总线相连

D.ALU 只能有一个输入端可以与总线相连，另一输入端需通过暂存器与总线相连

**04.** CPU 内部如果多个部件共享一条总线，则每个部件与总线之间需设置一个常用的器件，

CPU 控制该器件的状态，实现某个部件与总线的连接或断开。该器件是( )。

A. 触发器 B. 多路选择器 C. 三态门 D. 与非门

**05.** CPU 内部电路通常采用总线连接方式，总线上信号流动的原则是( )。

|  |  |
| --- | --- |
| 1. 每个时刻只有一个器件发出信息 | II. 每个时刻有一个或多个器件发出信息 |
| ⅢI. 每个时刻只有一个器件接收信息 | IV. 每个时刻有一个或多个器件接收信息 |
| A.I 、II B.I、IV | C.Ⅱ 、II D.I 、IV |

**06.** 下列关于单周期数据通路和多周期数据通路的说法中，正确的是()。 A. 单周期CPU 的 CPI 总比多周期CPU 的 CPI 大

B. 单周期 CPU 的时钟周期通常比多周期CPU 的时钟周期短

C. 在一条指令执行过程中，单周期 CPU 中的每个控制信号取值一直不变，而多周期

CPU 中的控制信号可能会发生改变

D. 在一条指令执行过程中，单周期数据通路和多周期数据通路中的每个部件都可使用

多次

07. 采用 CPU 内部总线的数据通路与不采用 CPU 内部总线的数据通路相比，( )。

A. 前者性能较高 B. 后者的数据冲突问题较严重

C. 前者的硬件量大，实现难度高 D. 以上说法都不对

**08.** CPU 的读/写控制信号的作用是()。

A. 决定数据总线上的数据流方向 B. 控制存储器操作的读/写类型

C. 控制流入、流出存储器信息的方向 D. 以上都是

**09.** 【2016统考真题】单周期处理器中所有指令的指令周期为一个时钟周期。下列关于单周 期处理器的叙述中，错误的是( )。

A. 可以采用单总线结构数据通路 B. 处理器时钟频率较低

C. 在指令执行过程中控制信号不变 D. 每条指令的 CPI 为 [1](#bookmark211)

**10.** 【2021 统考真题】下列关于数据通路的叙述中，错误的是()。

A. 数据通路包含ALU 等组合逻辑(操作)元件

B. 数据通路包含寄存器等时序逻辑(状态)元件

C. 数据通路不包含用于异常事件检测及响应的电路

D. 数据通路中的数据流动路径由控制信号进行控制

**11.【**2023 统考真题】数据通路由组合逻辑元件(操作元件)和时序逻辑元件(状态元件) 组成。下列给出的元件中，属于操作元件的是( )。

1. 算术逻辑单元 (ALU) II. 程序计数器 (PC)

Ⅲ . 通用寄存器组(GPRs) IV. 多路选择器 (MUX)

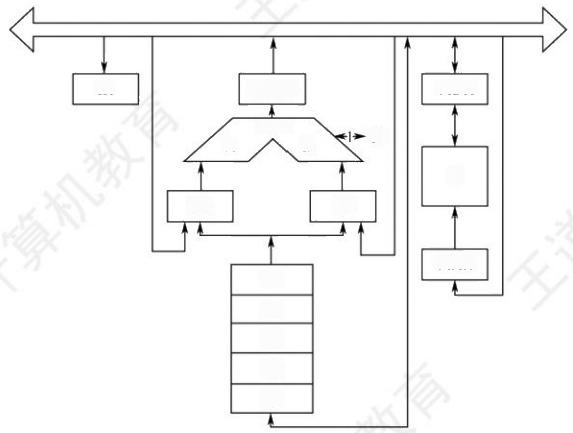
A. 仅 I 、Ⅱ B. 仅 I 、IV C. 仅Ⅱ、Ⅲ D. 仅 I 、Ⅱ 、IV

总线

第5章 中央处理器 223

**二** **、** **综** **合** **应** **用** **题**

**01.** 某计算机的数据通路结构如下图所示，写出实现 ADDR1,(R2) 的微操作序列(取指令及 指令执行的过程，包括PC 自增的过程)。

IR 移位器

MDR

ALU

CO

A

B

M

LA LB

MAR

R0

R1

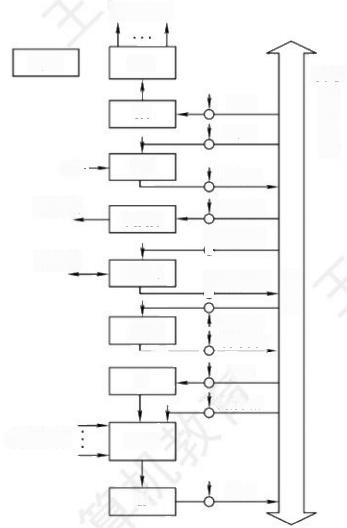
R2

R3

PC

**02.** 设 CPU 内部结构如下图所示，此外还设有 B、C、D、E、H、L 六个寄存器(图中未画出), 它们各自的输入端和输出端都与内部总线相通，并分别受控制信号控制(如 Bin 受寄存器B 的输入控制；Bout 受寄存器B 的输出控制),假设ALU 的结果直接送入寄存器Z。要求从 取指令开始，写出完成下列指令的微操作序列及所需的控制信号。

|  |  |
| --- | --- |
| ADD B,C | (B)+(C)→ B |
| SUB ACC,H | (ACC)-(H)→ACC |



控制信号

CU

CPU

IRin

IR

PCin

PC

PCout

MARin

6MDRin

数据线

MDR

SMDRout

ACCin

ACCout

Yin

ALUin

控制信号 ALU

Zout

Z

内部总线

地址线

时钟

MAR

ACC

+1 一

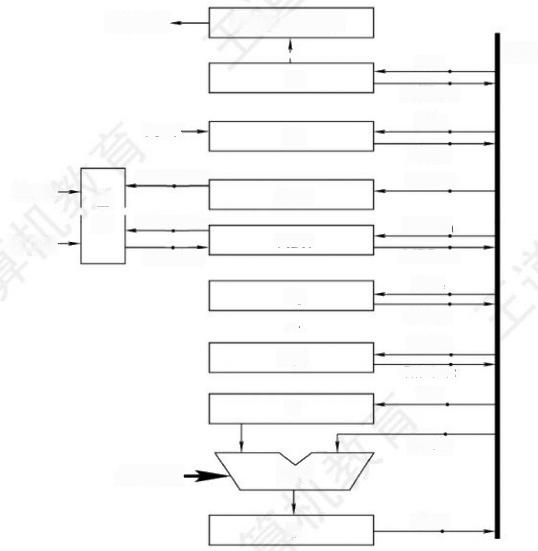
Y

主 存

224-2025年计算机组成原理考研复习指导

**03** . 设 有 如 下 图 所 示 的 单 总 线 结 构 ，

指 令 流 程 和 控 制 信 号 。

控制信号-

PC+1-

.MARoutr

MemR-

.MDRoutE

MemW-

MDRimE

控制信号-

分 析 指 令 ADD(RO),R1

指令译码/控制器

IR

PC

MAR

MDR

R₀

…

R1

Y

ALU

[ 即 实 现 ((RO))+(R1)→ (RO)] 的

总线

IRin

AdIRout

PCin

PCout

MARin

MDRin

MDRout

ROin

R0oui

Rn-lin

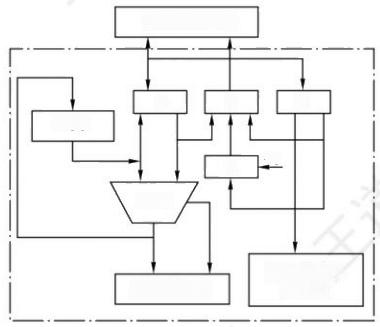
Rn-lout

Yin

ALUin

Z Zout

**04.** 下 图 是 一 个 简 化 的 CPU 与 主 存 连 接 结 构 示 意 图 ( 图 中 省 略 了 所 有 的 多 路 选 择 器 ) 。 其 中 有一个累加寄存器 (ACC)、 一个状态数据寄存器和其他4个寄存器：主存地址寄存器 (MAR) 、主存数据寄存器(MDR) 、程序寄存器(PC) 和指令寄存器 (IR), 各部件及 其之间的连线表示数据通路，箭头表示信息传递方向。



主存储器(M)

b

ACC

+1

ALU

微操作信号 发生器

状态寄存器

c

a

d

要求；

1)请写出图中a 、b 、c 、d四个寄存器的名称。

2 ) 简 述 图 中 取 指 令 的 数 据 通 路 。

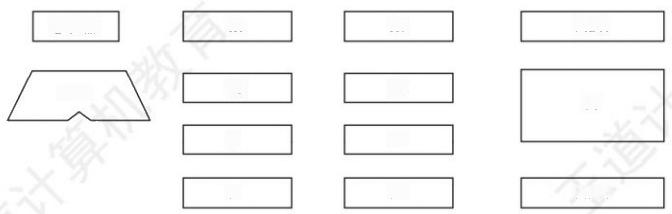
3 ) 简 述 数 据 在 运 算 器 和 主 存 之 间 进 行 存 / 取 访 问 的 数 据 通 路 ( 假 设 地 址 已 在MAR 中 ) 。 4 ) 简 述 完 成 指 令LDAX 的 数 据 通 路 (X 为 主 存 地 址 ，LDA 的 功 能 为(X)→ACC)。

第5章 中央处理器 225

5)简述完成指令ADD Y的数据通路 (Y 为主存地址， ADD 的功能为(ACC)+(Y)→ ACC)。

6)简述完成指令STAZ 的数据通路 (Z 为主存地址， STA 的功能为(ACC)→Z)。

**05.** 某机主要功能部件如下图所示，其中 M 为 主 存 ，MDR 为主存数据寄存器， MAR 为主 存地址寄存器， IR 为指令寄存器，PC 为程序计数器(并假设当前指令地址在PC 中), R0～R3 为通用寄存器， C 、D 为暂存器。



MDR

M

MAR

移位器

ALU

IR

PC

C

D

R0

R1

R2

R3

1)请补充各部件之间的主要连接线(总线自己画),并注明数据流动方向。

2 ) 画 出 “ADD(R1),(R2)+” 指令周期流程图，该指令的含义是进行求和运算，源操作 数地址在R1 中，目标操作数寻址方式为自增型寄存器间接寻址方式(先取地址后加 1),并将相加结果写回R2 寄存器。

**06.** 已知单总线计算机结构如下图所示，其中M 为主存， XR 为变址寄存器，EAR 为有效地 址寄存器， LATCH 为暂存器。假设指令地址已存在于PC 中，请给出 ADD X,D指令周

期信息流程和相应的控制信号。说明：

1)ADD X,D指令字中， X 为变址寄存器 XR,D 为形式地址，指令的功能是将变址寻

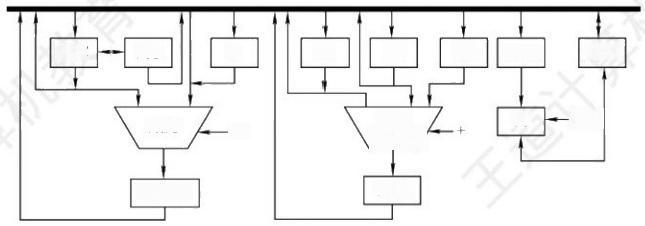
址得到的操作数和ACC 中的操作数相加，结果送回 ACC。

2)寄存器的输入/输出均采用控制信号控制，如 PC, 表示 PC 的输入控制信号，MDR。

表示 MDR 的输出控制信号。

3)凡需要经过总线的传送，都需要注明，如(PC)→MAR, 相应的控制信号为PC。和

MAR。

ACC MO X IR PC XR MAR MDR

地址

W及

M

ALU

-Ki

加法器/

EAR

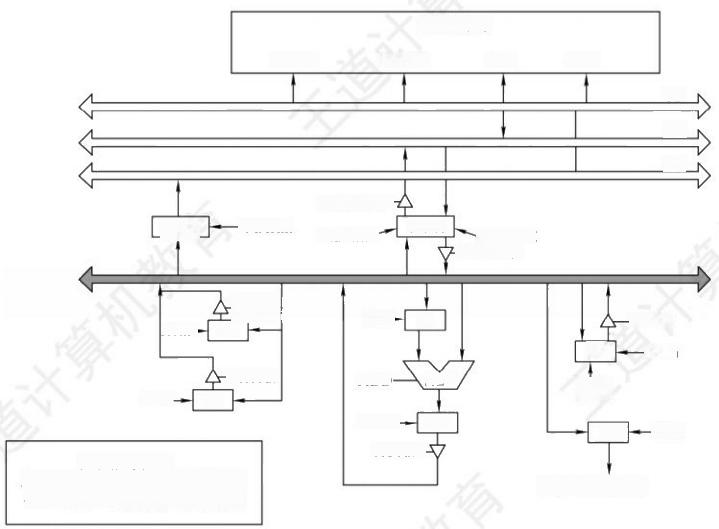
LATCH

**07.**【2009统考真题】某计算机字长16位，采用16位定长指令字结构，部分数据通路结构 如下图所示。图中所有控制信号为1 时表示有效，为0 时表示无效。例如，控制信号 MDRinE 为1表示允许数据从DB 打入MDR,MDRin 为1表示允许数据从总线打入MDR。 假设MAR 的输出一直处于使能状态。加法指令 “ADD(R1),R0” 的功能为(RO)+((R1))→

(R1), 即将R0 中的数据与 R1 的内容所指主存单元的数据相加，并将结果送入R1 的内

容所指主存单元中保存 。

226 - 2025年计算机组成原理考研复习指导

存储器(M)

MemR MemW Data Addr

CB

DB

AB

MDRoutE—

-MARin 一 MDR ~MDRinE

MAR MDRin

—MDRout

R0in→

Rlin-

=80ou

R0

—Rlout

R1

Ain→

ADD-

ACin—

A

AL U

AC

PC

pC+i

IR

-PCout

-PCin

-IRin

ACout—

控制信号图例

△—Xout 三态门及其控制信号

——Xin 寄存器输入控制信号

至指令译码部件

下表给出了上述指令取指和译码阶段每个节拍(时钟周期)的功能和有效控制信号，请 按表中描述方式用表格列出指令执行阶段每个节拍的功能和有效控制信号。

|  |  |  |
| --- | --- | --- |
| 时 钟 | 功 能 | 有效控制信号 |
| C1 | MAR-(PC) | PCout,MARin |
| C2 | MDR-M(MAR)  PC-(PC)+1 | MemR,MDRinE  PC+1 |
| C3 | IR--(MDR) | MDRout,IRin |
| C4 | 指令译码 | 无 |

**08.** 【2015统考真题】某16位计算机的主存按字节编码，存取单位为16位；采用16位定长指 令字格式； CPU 采用单总线结构，主要部分如下图所示。图中R0～R3 为通用寄存器； T 为暂存器； SR 为移位寄存器，可实现直送 (mov) 、 左移一位(left) 和右移一位 (right) 三种操作，控制信号为SRop,SR 的输出由信号SRout 控制； ALU 可实现直送A(mova)、

A加B(add)、A 减B(sub)、A 与B(and)、A 或B(or)、 非 A(not)、A 加 1(inc) 七 种

操作，控制信号为ALUop.

回答下列问题：

1)图中哪些寄存器是程序员可见的?为何要设置暂存器T?

2)控制信号ALUop 和 SRop 的位数至少各是多少?

3)控制信号 SRout 所控制部件的名称或作用是什么?

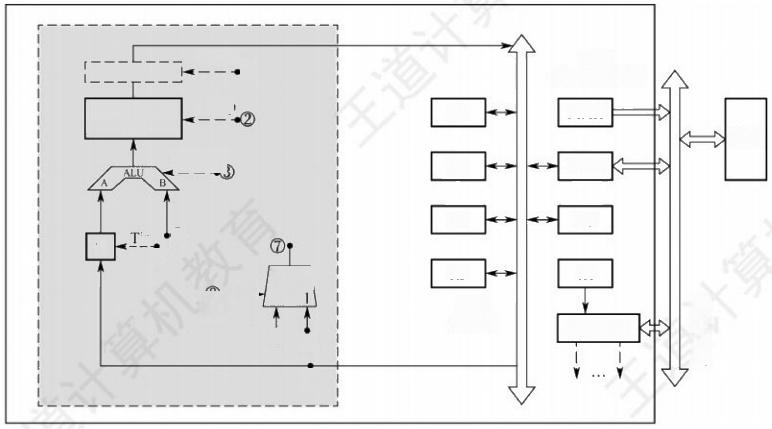
4)端点①~⑨中，哪些端点须连接到控制部件的输出端?

5)为完善单总线数据通路，需要在端点①~⑨中相应的端点之间添加必要的连线。写

出连线的起点和终点，以正确表示数据的流动方向。

6)为什么二路选择器 MUX 的一个输入端是2?

第 5 章 中 央 处 理 器 227



CPU

(中央处理器)

MAR

主 存

R1

R2

in ④

⑤

R3

MUX

0

MUXop

内总线

⑨

⑥

移位寄存器 SR

主存总线

控制部件

ALUop

SRop

SRout

MDR

CPU

①

PC

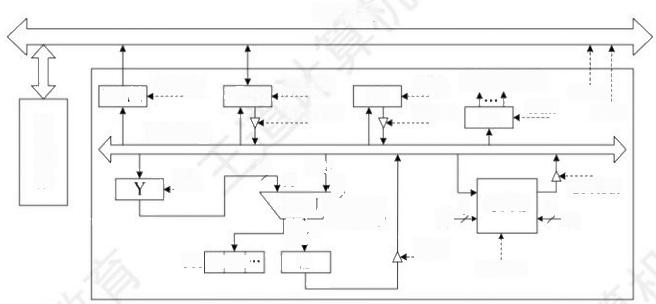
R0

IR

T

2

**09.**【2022统考真题】某CPU中部分数据通路如下图所示，其中， GPRs 为通用寄存器组； FR 为标志寄存器，用于存放ALU 产生的标志信息；带箭头虚线表示控制信号，如控制信号 Read 、Write分别表示主存读、主存写，MDRin 表示内部总线上的数据写入MDR,MDRout 表示MDR 的内容送给内部总线。



系统总线

MAlRMARin

IRin IR

内部总线

Yin

4. rd-

F

16j

Zout

Z

GPRin

MDRin

MDRout

PCin

PCout

主存储器

送CU 等部件

A 16L

FR SF|OF …

/ALUop

GPRout

Write

GPRs

MDR

Read

ALU

B

PC

16

rs

4

请回答下列问题。

1 ) 设 ALU 的输入端 A 、B 及输出端 F 的最高位分别为Als 、Bis 及 Fs,FR 中的符号标志和 溢出标志分别为 SF 和 OF, 则 SF 的逻辑表达式是什么? A 加 B 、A 减 B 时 OF 的逻辑表

达式分别是什么?要求逻辑表达式的输入变量为Als、B₁ s 及 Fjs。

2)为什么要设置暂存器 Y 和 Z?

3 ) 若GPRs 的输入端 rs、rd分别为所读、写的通用寄存器的编号，则GPRs 中最多有多少个 通用寄存器? rs 和 rd 来自图中的哪个寄存器?已知GPRs 内部有一个地址译码器和一个

多 路 选 择 器 ，rd 应 连 接 地 址 译 码 器 还 是 多 路 选 择 器 ?

4)取指令阶段(不考虑PC 增量操作)的控制信号序列是什么?若从发出主存读指令到主存 读出数据并传送到MDR 共需5个时钟周期，则取指令阶段至少需要几个时钟周期?

5)图中控制信号由什么部件产生?图中哪些寄存器的输出信号会连到该部件的输入端?

228-2025年计算机组成原理考研复习指导

**5.3.6** **答案与解析**

**一** **、单项选择题**

**01.C**

对 CPU 而言，数据通路的基本结构分为总线结构和专用数据通路结构，其中总线结构又分 为单总线结构、双总线结构、多总线结构。

**02.D**

数据通路中的部件包括组合逻辑元件和时序逻辑元件。数据通路的功能由控制部件送出的控 制信号决定。数据通路中一个重要的组合逻辑元件为 ALU, 用于执行各类算术和逻辑运算；另一 个重要的元件为通用寄存器，属于时序逻辑元件。

**03.D**

由于 ALU 是一个组合逻辑电路，因此其运算过程中必须保持两个输入端的内容不变。又由 于 CPU 内部采用单总线结构，因此为了得到两个不同的操作数，ALU 的一个输入端与总线相连， 另一个输入端需通过一个寄存器与总线相连。此外， ALU 的输出端也不能直接与内部总线相连， 否则其输出又会通过总线反馈到输入端，影响运算结果，因此输出端需通过一个暂存器(用来暂 存结果的寄存器)与总线相连。

**04.C**

三态门可视为一种控制开关，由控制端决定信号线的通断，能输出到内部总线的部件均通过 一个三态门与内部总线相连，用于控制该部件与内部总线之间数据通路的连接与断开。

**05.B**

当 CPU 内部电路采用总线连接方式时，总线上信号流动的原则如下：每个时刻只有一个器 件发出信息(否则会导致总线冲突),每个时刻可以有一个或多个器件接收信息。

**06.** C

多周期CPU 中的指令通常需要多个时钟周期才能完成，CPI>1; 单周期CPU 的每条指令在 一 个时钟周期内完成，CPI=1 。 单 周 期 CPU 的时钟周期取决于最复杂指令的执行时间，通常 比多周期 CPU 的时钟周期长。在 一 条指令的执行过程中，单周期 CPU 的每个控制信号保持 不变，每个部件只能使用 一 次；多周期CPU 的控制信号可能会发生改变，同 一 个部件可使用 多次。

07.D

采用 CPU 内部总线方式的数据通路的特点：结构简单，实现容易，性能较低，存在较多的 冲突现象；不采用CPU 内部总线方式的数据通路的特点：结构复杂，硬件量大，不易实现，性能 高，基本不存在数据冲突现象。

**08.D**

读/写控制信号线决定了是从存储器读还是向存储器写，显然A 、B 、C 都正确。

**09.A**

单周期处理器中所有指令的指令周期为 一个时钟周期，D 正 确。因为每条指令的CPI 为 1 , 要考虑比较慢的指令，所以处理器的时钟频率较低，B 正确。单总线数据通路将所有寄存器的 输入输出端都连接在一条公共通路上， 一个时钟内只允许一次操作，无法完成指令的所有操作， A 错 误。控制信号是 CU 根据指令操作码发出的信号，对于单周期处理器来说，每条指令的执 行只有一个时钟周期，而在一个时钟周期内控制信号并不会变化；若是多周期处理器，则指令 的执行需要多个时钟周期，在每个时钟周期控制器会发出不同信号， C 正 确 。

**第5章** **中央处理器** 229

10.C

指令执行过程中数据所经过的路径，包括路径上的部件，称为数 据 通 路 。ALU 、通用寄存器、 状 态 寄 存 器 、Cache 、MMU 、 浮点运算逻辑、异常和中断处理逻辑等，都是指令执行过程中数据 流经的部件，都属于数据通路的 一 部分。数据通路中的数据流动路径由控制部件控制，控制部件 根据每条指令功能的不同，生成对数据通路的控制信号。C 错 误 。

**11.B**

组合逻辑元件(操作元件)不含存储信号的记忆单元，任何时刻产生的输出仅取决于当前的 输入，加法器、算术逻辑单元 (ALU) 、 译码器、多路选择器、三态门等都属于操作元件。时序 逻辑元件(状态元件)包含存储信号的记忆单元，各类寄存器和存储器，如通用寄存器组、程序 计数器、状态/移位/暂存/锁存寄存器等，都属于状态元件。

**二** **、综合应用题**

**01.【**解答】



实 现 ADD R1,(R2)的微操作序列如下：

微操作

(PC)→MAR

M→MDR

(PC)+1→PC

(MDR)→IR

(R1)→LA

(R2)→MAR

M→MDR

(MDR)→LB

(LA)+(LB)→R1

**02.【**解答】

两条指令的微操作序列和控制信号如下。

(1)ADDB,C 指令。

微操作 控制信号

(PC)→MAR PCout,MARin

(PC)+1→PC +1

M(MAR)→MDR→IR MDRout, IRin

(B)→Y Bout,Yin

(Y)+(C)→Z Cout,ALUin,“+”

(Z)→B Zout,Bin

(2)SUBACC,H 指令。

微操作 控制信号

(PC)→MAR PCout,MARin

(PC)+1→PC +1

M(MAR)→MDR→IR MDRout,IRin

(ACC)→Y ACCout,Yin

(Y)-(H)→Z Hout,ALUin,“-”

(Z)→ACC Zout,ACCin

注 ：Y 是 与 ALU 的 一 个输入端相连接的暂存器。

**03.**【解答】

指 令ADD(RO),R1 的 功 能 是 把RO 的内容作为地址送到主存中取得 一 个操作数，再与R1 中 的内容相加，最后将结果送回主存，即实现((RO))+(R1)→(RO) 。 其流程和控制信号如下。

230-2025年计算机组成原理考研复习指导

1)取指周期：公共操作。

|  |  |  |  |
| --- | --- | --- | --- |
| 时 序 | 微 操 作 | 有效控制信号 | **具** **体** **功** **能** |
| 1 | (PC)→MAR | PCout,MARin | 将PC经内部总线送至MAR |
| 2 | M(MAR)→MDR,Read | MemR,MARout,MDRinE | 主存通过数据总线将MAR所指单元的内容送至MDR |
| 3 | (MDR)→IR | MDRout,IRin | 将MDR的内容送至IR |
| 4 | 指令译码 | 一 | 操作字开始控制CU |
| 5 | (PC)+1→PC |  | 当PC+1有效时，使PC内容加1 |

2)取数周期：完成取数操作，被加数在主存中，加数已经放在寄存器R1 中 。

|  |  |  |  |
| --- | --- | --- | --- |
| 时 序 | 微 操 作 | 有效控制信号 | **具** **体** **功** **能** |
| 1 | (R0)→MAR | R0out,MARin | 将R0中的地址(形式地址)送至存储器地址寄存器 |
| 2 | M(MAR)→MDR | MemR,MARout,MDRinE | 主存通过数据总线将MAR所指单元的内容(有效地址)送 至MDR中 |
| 3 | (MDR)→Y | MDRout,Yin | 将MDR中数据通过数据总线送至Y |

3)执行周期：完成加法运算，并将结果返回主存。

|  |  |  |  |
| --- | --- | --- | --- |
| **时** **序** | **微** **操** **作** | **有效控制信号** | **具** **体** **功** **能** |
| 1 | (R1)+(Y)→Z | Rlout,ALUin,CU向ALU 发ADD控制信号 | R1的内容和Y的内容相加，结果送至寄存器Z |
| 2 | (Z)→MDR | Zout,MDRin | 将运算结果送至MDR |
| 3 | (MDR)→M(MAR) | MemW,MDRoutE  MARout | 向主存写入数据 |

**04.【**解答】

1)b 单向连接微控制器，由微控制器的作用可以推出b 是 IR;a 和 c 直接连接主存，只可能 是MDR 和 MAR,c 到主存是单向连接， a 和主存双向连接，根据指令执行的特点，MAR 只单向给主存传送地址，而MDR 既存放从主存中取出的数据，又存放将要写入主存的 数据，因此c 为 MAR,a 为 MDR 。d 具有自动加1的功能，且单向连接MAR, 为 PC 。 因此， a 为 MDR,b 为 IR,c 为 MAR,d 为 PC。

2)将指令地址从 PC 送入 MAR, 在相关的控制下从主存中取出指令送至 MDR, 然后将 MDR 中的指令送至IR, 最后流向微控制器。取指令的数据通路为

PC→MAR→ 主存 →MDR→IR

3 ) 根 据MAR 中的地址从主存取数据，将取出的数据送至 MDR, 然后将 MDR 中的数据送

至 ALU 进行运算，运算的结果送至 ACC 。存储器读的数据通路为

MAR (先置数据地址),主存 M→MDR→ALU→ACC

将 ACC 中的结果送至MDR, 再 将MDR 中的数据写入主存。存储器写的数据通路为 MAR (先置数据地址),ACC→MDR→ 主存M

4 ) 指 令LDAX 的数据通路为

X→MAR→ 主存 →MDR→ALU→ACC

5 ) 指 令ADDY 的数据通路为

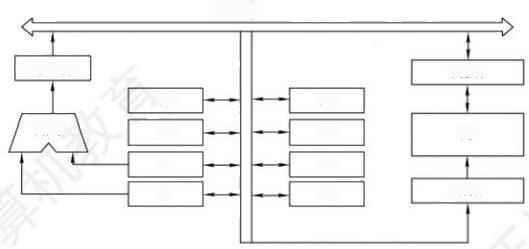


6 ) 指 令STAZ 的数据通路为 (ACC 中的数据需放在主存中)

Z→MAR,ACC→MDR→ 主存

05. 【解答】

1)各功能部件的连接关系及数据通路如下图所示。



内总线

MDR

IR

M

C

MAR

R0

R1

R2

R3

移位器

ALU

PC

D

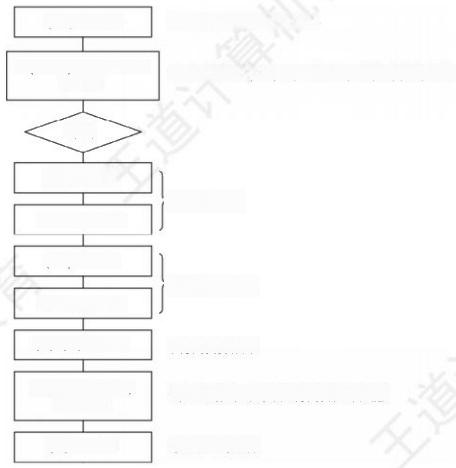
2)分析过程如下：

●取指令地址送到IR 并译码。

●取源操作数和目的操作数。

●将源操作数和目的操作数相加送到 MAR, 随之送到以前目的操作数所在内存的地址。

● 将寄存器 R2 的内容加1。 取指周期流程如下图所示。



|  |  |
| --- | --- |
| (PC)→MAR | 送当前指令地址到MAR |
| M(MAR)→MDR→IR (PC)+1  译码 | 取出当前指令，(PC)+1 为取下条指令做好准备 |
| (R1)→MAR |  |
|  | 取源操作数 |
| M→MDR→C |  |
| (R2)→MAR |  |
|  | 取目的操作数 |
| M→MDR→D |  |
| (C)+(D)→MDR | 两操作数相加 |
| MDR→M(MAR) (R2)→D | 写回主存中原来目的操作数的位置 |
| (D)+1→R2 | 修改R2 的内容 |

**06.【**解答】

ADDX,D 指令周期信息流程和相应的控制信号见下表。

|  |  |  |
| --- | --- | --- |
| 周 期 | 微 操 作 | 有效控制信号 |
| 取指周期 | (PC)→MAR | PC。。MAR, |
| M(MAR)→MDR  (PC)+1→PC | MAR,R/W,MDR  +1 |
| (MDR)→IR | MDR。IR |

232-2025年计算机组成原理考研复习指导

**(续表)**

|  |  |  |
| --- | --- | --- |
| 周 期 | 微 操 作 | 有效控制信号 |
| 执行周期 | (XR)+Ad(IR)→EAR | XR。IR。,+,EAR |
| (EAR)→MAR | EAR。,MAR |
| M(MAR)→MDR | MAR。R/W,MDR |
| (MDR)→X | MDR。,X₁ |
| (ACC)+(X)→LATCH | ACC。,X₀,K₁=+,LATCH₁ |
| (LATCH)→ACC | LATCH,ACC; |

注：题目中的D即为Ad(IR)。

07. 【解答】

题干已给出取值和译码阶段每个节拍的功能和有效控制信号，我们应以了解取指阶段中数据 通路的信息流动为突破口，读懂每个节拍的功能和有效控制信号，然后应用到解题思路中，包括 划分执行步骤、确定完成的功能、需要的控制信号。

先分析题干中提供的示例(本部分解题时不做要求):

取指令的功能是根据 PC 的内容所指的主存地址，取出指令代码，经过 MDR, 最终送至IR。 这部分和后面的指令执行阶段的取操作数、存运算结果的方法是相通的。

C1:(PC)→MAR

在读/写存储器前，必须先将地址(这里为(PC)) 送 至MAR。

C2:M(MAR)→MDR,(PC)+1→PC

读/写的数据必须经过MDR, 指令取出后 PC 自 增 1 。

C3:(MDR)→IR

然后将读到的 MDR 中的指令代码送至IR 进行后续操作。

指 令 “ADD(R1),RO” 的操作数一个在主存中， 一个在寄存器中，运算结果在主存中。根据 指令功能，要读出R1 的内容所指的主存单元，必须先将R1 的内容送至MAR, 即(R1)→MAR。

而读出的数据必须经过MDR, 即 M(MAR)→MDR。

因此，将R1 的内容所指的主存单元的数据读出到MDR 的节拍安排如下：

C5:(R1)→MAR

C6:M(MAR)→MDR

ALU 一端是寄存器 A,MDR 或 R0 中必须有一个先写入A 中，如 MDR。

C7:(MDR)→A

然后执行加法操作，并将结果送入寄存器AC。

C8:(A)+(RO)→AC

之后将加法结果写回到R1 的内容所指的主存单元，注意MAR 中的内容没有改变。

C9:(AC)→MDR

C10:(MDR)→M(MAR)

有效控制信号的安排并不难，只需看数据是流入还是流出，如流入寄存器X 就 是Xin, 流 出 寄存器X 就 是Xout 。 还需注意其他特殊控制信号，如PC+1 、Add 等 。

于是得到参考答案如下表所示。

|  |  |  |
| --- | --- | --- |
| **时** **钟** | 功 能 | 有效控制信号 |
| C5 | MAR-(R1) | Rlout,MARin |
| C6 | MDR-M(MAR) | MemR,MDRinE |

第5章 中央处理器 233

(续表)

|  |  |  |
| --- | --- | --- |
| 时 钟 | 功 能 | 有效控制信号 |
| C7 | A-(MDR) | MDRout,Ain |
| C8 | AC-(A)+(RO) | R0out,ADD,ACin |
| C9 | MDR-(AC) | ACout,MDRin |
| C10 | M(MAR)-(MDR) | MDRoutE,MemW |

本题答案不唯一，若在 C6 执行 M(MAR)→MDR 的同时，完成(RO)→A (即选择将(R0)写入 A), 并不会发生总线冲突，这种方案可节省1个节拍，见下表。

|  |  |  |
| --- | --- | --- |
| 时 钟 | 功 能 | 有效控制信号 |
| C5 | MAR-(R1) | Rlout,MARin |
| C6 | MDR-M(MAR),A(RO) | MemR,MDRinE,ROout,Ain |
| C7 | AC-(MDR)+(A) | MDRout,ADD,ACin |
| C8 | MDR--(AC) | ACout,MDRin |
| C9 | M(MAR)—(MDR) | MDRoutE,MemW |

08. 【解答】

1)程序员可见寄存器为通用寄存器 (R0～R3) 和PC。因为采用了单总线结构，因此若无暂 存器T, 则 ALU的A 、B 端口会同时获得两个相同的数据，使数据通路不能正常工作。

2)ALU 共有7种操作，其操作控制信号 ALUop 至少需要3位；移位寄存器有3种操作， 其操作控制信号SRop 至少需要2位。

3)信号 SRout 所控制的部件是一个三态门，用于控制移位器与总线之间数据通路的连接与 断开。

4)端口①、②、③、⑤、⑧须连接到控制部件输出端。

5)连线1,⑥ → ⑨;连线2,⑦ → ④。

6)因为每条指令的长度为16位，按字节编址，所以每条指令占用2个内存单元，顺序执行 时，下条指令地址为(PC)+2。MUX 的一个输入端为2,可便于执行(PC)+2 操作。

09.【解析】

1)符号标志 SF 表示运算结果的正负性，因此 SF=Fjs。

对于加法运算A+B→F, 若A 、B为负，且F 为正，则说明发生溢出；或者，若A 、B为正， 且F 为负，也说明发生溢出。因此，加运算时，溢出标志OF=Ajs·B₁₅·Fis+AjsBisFs。

对于减法运算A-B→F, 若A 为负、B 为正，且F 为正，则说明发生溢出；或者，若A 为正、B 为负，且F 为负，也说明发生溢出。因此，减运算时，溢出标志OF=A₁s·B₁sFis+

Ais·B₁s·Fs。

2)因为在单总线结构中，每一时刻总线上只有一个数据有效，而ALU 有两个输入端和一个 输出端。因此，当ALU 运算时，需要先用暂存器Y 缓存其中一个输入端的数据，再通过 总线传送另一个输入端的数据。与此同时， ALU的输出端产生运算结果，但由于总线正 被占用，因此需要暂存器Z, 以缓存ALU 的输出端数据。

3)由图可知， rs 和rd都是4bit, 因此GPRs中最多有2⁴=16个通用寄存器； rs 和rd 来自指令 寄存器IR;rd 表示寄存器编号，应连接地址译码器。

4)取指阶段需要根据程序计数器PC 取出主存中的指令，并将指令写入指令寄存器IR 中。控制 信号序列如下：



234-

**2025年计算机组成原理考研复习指导**

①PCout,MARin

② Read

③MDRout,IRin

//将指令的地址写入MAR

//读主存，并将读出的数据写入MDR

//将MDR的内容写入指令寄存器 IR

步骤①需要1个时钟周期，步骤②需要5个时钟周期，步骤③需要1个时钟周期，因此取指

令阶段至少需要7个时钟周期。

5)图中控制信号由控制部件 (CU) 产生。指令寄存器IR 和标志寄存器 FR 的输出信号会连

到控制部件的输入端。



**5.4控制器的功能和工作原理**

**5.4.1** **控制器的结构和功能**

从图5 . 9可以看到计算机硬件系统的五大功能部件及其连接关系。它们通过数据总线、地址

总线和控制总线连接在 一 起，其中点画线框内的是控制器部件。

现对其主要连接关系简单说明如下：

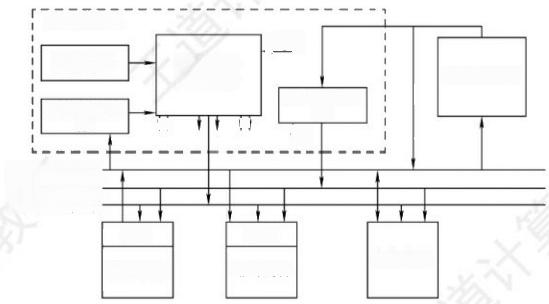
1)运算器部件通过数据总线与内存储器、输入设备和输出设备传送数据。 2)输入设备和输出设备通过接口电路与总线相连接。

3)内存储器、输入设备和输出设备从地址总线接收地址信息，从控制总线得到控制信号，

通过数据总线与其他部件传送数据。

4)控制器部件从数据总线接收指令信息，从运算器部件接收指令转移地址，送出指令地址

到地址总线，还要向系统中的部件提供它们运行所需要的控制信号。



控制器部件

\_条件

步骤标记

运算器部件

指令奇存器

IR

4+1

数据总线(DB)-

地址总线(AB)一

控制总线(CB)一

接口

输入设备

控制信号 产生电路

开 f

接口

输出设备

程序计数器 PC

内存储器

图5.9 计算机硬件系统和控制器部件的组成

控制器是计算机系统的指挥中心，控制器的主要功能有：

1)从主存中取出一条指令，并指出下一条指令在主存中的位置。

2)对指令进行译码或测试，产生相应的操作控制信号，以便启动规定的动作。

3 ) 指 挥 并 控 制CPU 、 主存、输入设备和输出设备之间的数据流动方向。

根据控制器产生微操作控制信号的方式的不同，控制器可分为硬布线控制器和微程序控制 器，两类控制器中的 PC 和 IR 是相同的，但确定和表示指令执行步骤的办法及给出控制各部件运

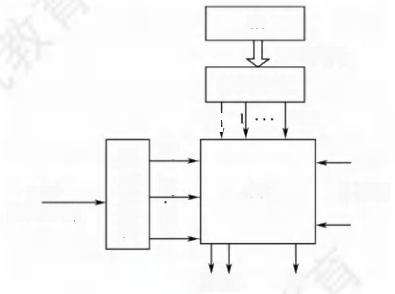
行所需要的控制信号的方案是不同的。

第5章 中央处理器 235

**5.4.2** **硬布线控制器**

硬布线控制器由复杂的组合逻辑门电路和触发器构成，也称组合逻辑控制器，其原理是根据 指令的要求、当前的时序及内外部的状态，按时间的顺序发送一系列微操作控制信号。

指令的操作码是决定控制单元 (CU) 发出不同控制信号的关键。为了简化CU 的逻辑，将存 放在IR 的 n 位操作码经过译码电路产生2”个输出，每种操作码对应一个输出送至CU 。 如果将指 令译码器和节拍发生器从CU 中分离出来，便可得到简化的控制单元框图，如图5. 10所示。



IR

n位操作码 操作码译码

0 2°-1

To

CU : 标志

C₀C₁ C,

CLK

(机器主频)

节拍发生器

T

…

1

图5.10 带指令译码器和节拍输入的控制单元框图

控制单元 (CU) 的输入信号来源如下：

1)经指令译码器译码产生的指令信息。现行指令的操作码决定了不同指令在执行周期所需 完成的不同操作，它与时钟配合产生不同的控制信号。

2)时序系统产生的机器周期信号和节拍信号。为了使控制单元按一 定的先后顺序、 一 定的 节奏发出各个控制信号，控制单元必须受时钟控制。

3)来自执行单元的反馈信息即标志。控制单元有时需依赖CPU 当前所处的状态产生控制信

号，如 BAN 指令，控制单元要根据上条指令的结果是否为负来产生不同的控制信号。

图5. 10中，节拍发生器产生各机器周期中的节拍信号，使不同的微操作命令C, (控制信号) 按时间的先后发出。个别指令的操作不仅受操作码控制，而且受到状态标志控制，因此CU 的 输 入来自操作码译码电路ID 、 节拍发生器及状态标志，其输出到CPU 内部或外部控制总线上。

硬布线控制的功能由逻辑门组合实现，其速度主要取决于电路延迟，因此高速计算机中的关 键核心部件 CPU 往往采用硬布线逻辑实现。因此， RISC 一般都选用硬布线控制器。硬布线控制 器的控制信号先用逻辑式列出，经化简后用电路来实现，因此显得零乱复杂，当需要修改或增加 指令时就必须重新设计电路，非常麻烦。而且指令系统功能越全，微操作命令就越多，电路也就 越庞杂，调试就更困难。为了克服这些缺点，便产生了微程序设计方法。

**5.4.3** **微程序控制器**

微程序控制器采用存储逻辑实现，也就是将微操作信号代码化，使每条机器指令转化成为一 段微程序并存入一个专门的存储器(控制存储器)中，微操作控制信号由微指令产生。

**1.** **微程序控制的基本概念**

微程序的设计思想就是将每条机器指令编写成一个微程序，每个微程序包含若干微指令，每条

微指令对应一个或几个微操作命令。因此，执行一条指令的过程就是执行一个微程序的过程，这些

236-2025年计算机组成原理考研复习指导

微程序存储在一个控制存储器中。目前，大多数计算机都采用微程序设计技术。 微程序设计技术涉及的基本术语如下。

(1)微命令与微操作

在微程序控制的计算机中，控制部件向执行部件发出的各种控制命令称为微命令，它是构成 控制序列的最小单位。例如，打开或关闭某个控制门的电位信号、某个寄存器的打入脉冲等。执 行部件收到微命令后所进行的操作称为微操作，微命令和微操作是一一对应的。

微命令有相容性和互斥性之分。相容性微命令是指那些可以同时出现、共同完成某一些微操 作的微命令；而互斥性微命令是指在机器中不允许同时出现的微命令。



**注** **意**

硬布线控制器中也有微命令与微操作的概念，并非微程序控制器的专有概念。

(2)微指令与微周期

微指令是若干微命令的集合， 一条微指令通常至少包含两大部分信息：

① 操作控制字段，也称微操作码字段，用于产生某一步操作所需的各种操作控制信号。 ② 顺序控制字段，也称微地址码字段，用于控制产生下 一条要执行的微指令地址。

微周期是指从控制存储器中取出并执行一条微指令所需的全部时间，通常为一个时钟周期。 (3)主存储器与控制存储器

**命** **题** **追** **踪** ▶ **主存储器和控制存储器的区别(2017)**

主存储器用于存放程序和数据，在CPU 外部，用RAM 实 现。控 制存 储 器 ( CM) 用于存放 微程序，在 CPU 内部，用ROM 实 现。存放微指令的控制存储器的单元地址称为微地址。

(4)程序与微程序

微程序和程序是两个不同的概念。程序是指令的有序集合，用于完成特定的功能。微程序是 微指令的有序集合，用于描述机器指令， 一条指令的功能由一段微程序来实现。微程序实际上是 机器指令的实时解释器，是由计算机设计者事先编制好并存放在控制存储器中的， 一般不提供给 用户。对于程序员来说，系统中微程序的结构和功能是透明的，无须知道。程序最终由机器指令 组成，并且由软件设计人员事先编制好并存放在主存储器或者辅助存储器中。

读者应注意区分以下寄存器：

① 地址寄存器 ( MAR )。用 于存放主存的读/写地址。

② 微指令地址寄存器( μPC 或 CMAR)。 用于存放待执行的微指令在控制存储器中的微地址。 ③ 指令寄存器 (IR )。用 于存放从主存中读出的指令。

④ 微指令寄存器(μlR 或 CMDR)。 用于存放从控制存储器中读出的微指令。

**2** **.微程序控制器的组成和工作过程**

(1)微程序控制器的基本组成

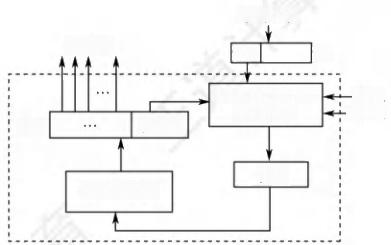
图5.11 所示为一个微程序控制器的基本结构，其主要部件包括：

① 起始和转移地址形成部件( 或 简 称微地址形成部件)。用于产生初始和后继微地址，以保 证微指令的连续执行。

② 微指令地址寄存器。接收微地址形成部件送来的微地址，为读取微指令做准备。

③ 控制存储器。它是微程序控制器的核心部件，用于存放各指令对应的微程序。 ④ 微指令寄存器。其位数等于微指令字长。





来自主存

OP| IR

起始和转移地址 形式部件

μlR

μPC

控制存储器

-标志 —CLK

微命令

下地址

图5.11 微程序控制器的基本结构

第5章 中央处理器- 237

(2)微程序控制器的工作过程

实际上就是在微程序控制器的控制下计算机执行机器指令的过程，这个过程可描述为：

① 执行取指令公共操作。在机器开始运行时，自动地将取指微程序的入口地址送入μPC,

并从 CM 中读出相应的微指令并送入μlR 。取指微程序的入口地址一般为 CM 的 0 号 单 元，取指微程序执行完成后，从主存中取出的机器指令就已存入指令寄存器中。

② 由机器指令的操作码字段通过微地址形成部件产生该机器指令所对应的微程序的入口地 址，并送入μPC。

③ 从 CM 中逐条取出对应的微指令并执行。

④ 执行完对应于一条机器指令的一个微程序后，又回到取指微程序的入口地址，继续第① 步，以完成取下一条机器指令的公共操作。

以上是一条机器指令的执行过程，如此周而复始，直到整个程序执行完毕。 (3)微程序和机器指令

通常， 一条机器指令对应一个微程序。由于任何机器指令的取指令操作都是相同的，因此可 将取指令操作的微命令统一编成一个微程序，这个微程序只负责将指令从主存单元中取出并送至 指令寄存器。此外，也可编写出对应间址周期的微程序和中断周期的微程序。这样，控制存储器 中的微程序个数应为机器指令数再加上对应取指、间址和中断周期等公共的微程序数。

**3.** **微指令的编码方式**

微指令的编码方式也称微指令的控制方式，是指如何对微指令的控制字段进行编码，以形成 控制信号。编码的目标是在保证速度的情况下，尽量缩短微指令字长。

(1)直接编码(直接控制)方式

微指令的直接编码方式如图5.12所示。直接编码法无须进行译码，微指令的操作控制字段中 每一位都代表一个微命令。设计微指令时，选用或不选用某个微命令，只要将表示该微命令的对 应位设置成1或0即可。每个微命令对应并控制数据通路中的一个微操作。

这种编码的优点是简单、直观，执行速度快，操作并行性好；缺点是微指令字长过长，n 个 微命令就要求微指令的操作字段有n 位，造成控制存储器容量极大。

(2)字段直接编码方式

▶ **字段直接控制的编码方法(2012)**

将微指令的操作控制字段分成若干小字段，把互斥性微命令放在同一字段中，把相容性微命 令放在不同字段中，每个字段独立编码，每种编码代表一个微命令且各字段编码含义单独定义， 与其他字段无关，这就是字段直接编码方式，如图5.13所示。这种方式可以缩短微指令字长，但 因为要通过译码电路后再发出微命令，因此比直接编码方式慢。

238—2025年计算机组成原理考研复习指导

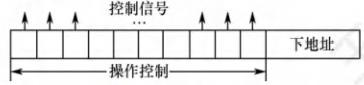
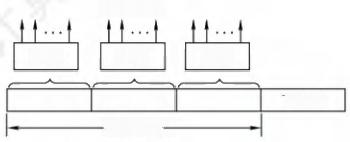


图5.12 直接编码方式



控制信号

译码 译码 译码

下地址

-操作控制—

图5.13 字段直接编码方式

微命令字段分段的原则：

① 互斥性微命令分在同一段内，相容性微命令分在不同段内。

② 每个小段中包含的信息位不能太多，否则将增加译码电路的复杂性和译码时间。

③ 一般每个小段还要留出一个状态，表示本字段不发出任何微命令。因此，当某字段的长 度为3位时，最多只能表示7个互斥的微命令，通常用000表示不操作。

(3)字段间接编码方式

一个字段的某些微命令需由另一个字段中的某些微命令来解释，由于不是靠字段直接译码发 出的微命令，因此称为字段间接编码，也称隐式编码。这种方式可进一步缩短微指令字长，但因 削弱了微指令的并行控制能力，因此通常作为字段直接编码方式的一种辅助手段。

**4.** **微指令的地址形成方式**

后继微地址的形成主要有以下几个基本类型：

1)由微指令的后继地址字段(也称下地址字段) 指 出。在微指令格式中设置一个后继地址 字段，由微指令的后继地址字段直接指出后继微指令的地址，这种方式也称断定方式。

2)根据机器指令的操作码形成。当机器指令取自指令寄存器后，微指令的地址由操作码经 微地址形成部件形成，该部件输出的是对应机器指令微程序的首地址。

3)增量计数器法，即(μPC)+1→μPC, 适用于后继微指令地址是连续的情况。 4)根据各种标志决定下一条微指令分支转移的地址。

5)由硬件直接产生微程序入口地址。电源加电后，第一条微指令的地址可由专门的硬件电 路产生，并送至μPC, 这个地址即为取指周期微程序的入口地址。

**5.** **微指令的格式**

微指令格式与微指令的编码方式有关，通常分为水平型微指令和垂直型微指令两种。

命 题 追 踪▶ **微指令后继地址字段位数与微指令条数的关系(2014)**

(1)水平型微指令

从编码方式看，直接编码、字段直接编码和字段间接编码都属于水平型微指令。水平型微指令 的基本指令格式如图5.14所示，指令字中的一位对应一个控制信号，有输出时为1,否则为0。 一条 水平型微指令定义并执行多个并行操作的微命令。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A₁ | A₂ |  | A-i | An | 判断测试字段 | 后继地址字段 |
| 操作控制 | | | | | 顺序控制 | |

图5.14 水平型微指令格式

水平型微指令的优点是微程序短，并行能力强，执行速度快；缺点是微指令长，编写微程序 较麻烦。

(2)垂直型微指令

采用类似机器指令操作码的方式，在微指令字中设置微操作码字段，垂直型微指令的基本格

**第5章** **中央处理器** 239

式如图5.15所示。 一条垂直型微指令通常只能定义并执行一种微命令。

|  |  |  |
| --- | --- | --- |
| μOP | Rd | Rs |
| 微操作码 | 目的地址 | 源地址 |

图5.15 垂直型微指令格式

垂直型微指令的优点是微指令短、简单、规整，便于编写微程序；缺点是微程序长，执行速 度慢，效率低。

水平型微指令和垂直型微指令的比较如下：

① 水平型微指令并行操作能力强、效率高、灵活性强；垂直型微指令则较差。 ② 水平型微指令执行一条指令的时间短；垂直型微指令执行的时间长。

③ 用水平型微指令编写的微程序，微指令字较长但微程序短；垂直型微指令正好相反。 ④ 水平型微指令难以掌握；而垂直型微指令与机器指令比较相似，相对容易掌握。

**6.** **硬布线和微程序控制器的特点**

命 题 追 踪▶ **硬布线控制器和微程序控制器的特点(2009)**

(1)硬布线控制器的特点

硬布线控制器的优点是由于控制器的速度取决于电路延迟，所以速度快；缺点是由于将控制 部件视为专门产生固定时序控制信号的逻辑电路，所以把用最少元件和取得最高速度作为设计目 标， 一旦设计完成，就不可能通过其他额外修改添加新功能。

(2)微程序控制器的特点

相比组合逻辑控制器，微程序控制器的优点是具有规整性、灵活性和可维护性；缺点是由于 微程序控制器采用了存储程序原理，所以每条指令都要从控制存储器中取一次，影响速度。

为便于比较，下面以表格的形式对比二者的不同，见表5.1。

**表5.1** **微程序控制器与硬布线控制器的对比**

|  |  |  |
| --- | --- | --- |
| **类别**  **对比项** | **微程序控制器** | **硬布线控制器** |
| 工作原理 | 微操作控制信号以微程序的形式存放在控制 存储器中，执行指令时读出即可 | 微操作控制信号由组合逻辑电路根据当前的 指令码、状态和时序即时产生 |
| 执行速度 | 慢 | 快 |
| 规整性 | 较规整 | 烦琐、不规整 |
| 应用场合 | CISC CPU | RISC CPU |
| 易扩充性 | 易扩充修改 | 扩充修改困难 |

**5.4.4** **本节习题精选**

**一** **、单项选择题**

**01.** 取指令操作()。

A. 受到上一条指令的操作码控制

B. 受到当前指令的操作码控制

C. 受到下一条指令的操作码控制

D. 是控制器固有的功能，不需要在操作码控制下进行

**02** . 在组合逻辑控制器中，微操作控制信号的形成主要与()信号有关。

A. 指令操作码和地址码 B. 指令译码信号和时钟

240-2025年计算机组成原理考研复习指导

C. 操作码和条件码 D. 状态信息和条件

**03.** 在微程序控制器中，形成微程序入口地址的是()。

A. 机器指令的地址码字段 B. 微指令的微地址码字段

C. 机器指令的操作码字段 D. 微指令的微操作码字段

**04.** 下列不属于微指令结构设计所追求目标的是( )。

A. 提高微程序的执行速度 B. 提供微程序设计的灵活性

C. 缩短微指令的长度 D. 增大控制存储器的容量

**05.** 微程序控制器的速度比硬布线控制器慢，主要是因为( )。

A. 增加了从磁盘存储器读取微指令的时间

**B.** 增加了从主存读取微指令的时间

C. 增加了从指令寄存器读取微指令的时间

D. 增加了从控制存储器读取微指令的时间

**06.** 下列关于微指令的说法中，错误的是()。

I. 字段直接编码方式可用较少的二进制位数表示较多的微操作命令。如果有两组互斥

的微命令，每组微命令的个数分别为4和9,则分别只需要2位和4位即可 II. 直接编码方式不用进行译码操作，微命令字段中的每一位都代表一个微命令

Ⅲ.垂直型微指令用较长的微程序结构换取较短的微指令结构，所以在执行效率和灵活 性两方面都高于水平型微指令

IV. 在字段间接编码方式中，某个字段的译码输出需要依靠另外某个字段的输入

A.Ⅱ B.I 、Ⅱ C.I 、Ⅲ D.I 、Ⅲ 、IV

**07.** 微程序控制存储器属于()的一部分。

A. 主存 B. 外存 C.CPU D. 缓存

**08.** 以下说法中，正确的是()。

A. 采用微程序控制器是为了提高速度

B. 控制存储器由高速RAM 电路组成

C. 微指令计数器决定指令执行顺序

D. 一条微指令存放在控制器的一个控制存储器单元中

**09.** 硬布线控制器与微程序控制器相比，( )。

A. 硬布线控制器的时序系统比较简单

B. 微程序控制器的时序系统比较简单

C. 两者的时序系统复杂程度相同

D. 可能是硬布线控制器的时序系统比较简单，也可能是微程序控制器的时序系统比较 简单

**10.** 在微程序控制器中，控制部件向执行部件发出的某个控制信号称为( )。

A. 微程序 **B.** 微指令 C. 微操作 D. 微命令

**11.** 在微程序控制器中，机器指令与微指令的关系是( )。

A. 每条机器指令由一条微指令来执行

**B.** 每条机器指令由若干微指令组成的微程序来解释执行

C. 若干机器指令组成的程序可由一个微程序来执行

D. 每条机器指令由若干微程序执行

**12.** 水平型微指令与垂直型微指令相比，()。

第5章 中央处理器 241

A. 前者一次只能完成一个基本操作

B. 后者一次只能完成一个基本操作

C. 两者都是一次只能完成一个基本操作

D. 两者都能一次完成多个基本操作

**13.** 垂直型微指令的特点是( )。

A. 控制信号经过编码产生 B. 强调并行控制功能

C. 采用微操作码 D. 微指令格式垂直表示

**14.** 下列关于微命令的描述中，正确的是( )。

A. 同 一CPU 周期中，可以同时出现的微命令叫相容性微命令

B. 同 一CPU 周期中，可以同时出现的微命令叫互斥性微命令

C. 在执行过程中可能会引起总线冲突的微命令叫互斥性微命令

D. 同 一CPU 周期中，不允许同时出现的微命令叫相容性微命令

**15.** 在微程序控制方式中，以下说法正确的是( )。

1. 采用微程序控制器的处理器称为微处理器

II. 每条机器指令由一段微程序来解释执行

Ⅲ.在微指令的编码中，效率最低的是直接编码方式

IV. 水平型微指令能充分利用数据通路的并行结构

A.I 、Ⅱ B.I 、IV C.I 、IIl D.Ⅲ 、IV **16.** 下列说法中，正确的是()。

1. 微程序控制方式和硬布线方式相比较，前者可以使指令的执行速度更快

II. 若采用微程序控制方式，则可用μPC 取 代PC

Ⅲ.控制存储器可以用 ROM 实现

IV. 指令周期也称 CPU 周期

A.I 、Ⅲ B.Ⅱ 、I C. 只 有 Ⅲ D.I 、II 、IV

**17.** 通常一条指令对应一个微程序， 一个微程序的周期对应一个()。

A. 指令周期 B. 主频周期 C. 机器周期 D. 工作周期

**18 .** 下列部件中属于控制部件的是( )。

I. 指令寄存器Ⅱ . 操作控制器 ⅢI. 程序计数器 IV. 状态条件寄存器

A.I 、II 、IV B.I 、Ⅱ 、Ⅲ C.I 、Ⅱ 、IV D.I 、I 、Ⅲ 、IV **19.** 为了确定下一条微指令的地址，通常采用断定方式，其基本思想是()。

A. 用程序计数器 (PC) 来产生后继微指令地址

B. 用微程序计数器(μPC) 来产生后继微指令地址

C. 通过微指令后继地址字段由设计者指定或转移控制字段控制产生后继微指令地址

D. 通过指令中指定一个专门字段来控制产生后继微指令地址

**20.【2** 009统考真题】相对于微程序控制器，硬布线控制器的特点是( )。

A. 指令执行速度慢，指令功能的修改和扩展容易

B. 指令执行速度慢，指令功能的修改和扩展难

C. 指令执行速度快，指令功能的修改和扩展容易

D. 指令执行速度快，指令功能的修改和扩展难

**21.【2** 012统考真题】某计算机的控制器采用微程序控制方式，微指令中的操作控制字段采 用字段直接编码法，共有33个微命令，构成5个互斥类，分别包含7、3、12、5和6

242-2025年计算机组成原理考研复习指导

个微命令，则操作控制字段至少有()。

A.5 位 B.6 位 C.15 位 D.33 位

**22.【**2014统考真题】某计算机采用微程序控制器，共有32条指令，公共的取指令微程序包 含2条微指令，各指令对应的微程序平均由4条微指令组成，采用断定法(后继地址字 段法)确定下条微指令地址，则微指令中后继地址字段的位数至少是( )。

A.5 B.6 C.8 D.9

**23.【**2017统考真题】下列关于主存储器 (MM) 和控制存储器 (CS) 的叙述，错误的是()。 A.MM在CPU外，CS在CPU内

B . MM 按地址访问，CS 按内容访问

C.MM 存储指令和数据， CS 存储微指令

D.MM 用 RAM 和 ROM 实现， CS 用 ROM 实现

**24.【**2019统考真题】下列有关处理器时钟脉冲信号的叙述中，错误的是()。

A. 时钟脉冲信号由机器脉冲源发出的脉冲信号经整形和分频后形成

B. 时钟脉冲信号的宽度称为时钟周期，时钟周期的倒数为机器主频

C. 时钟周期以相邻状态单元间组合逻辑电路的最大延迟为基准确定

D. 处理器总是在每来一个时钟脉冲信号时就开始执行一条新的指令

**25.【**2019统考真题】某指令的功能为R[r2]--R[rl]+M[R[r O]], 其两个源操作数分别采用寄 存器、寄存器间接寻址方式。对于下列给定部件，该指令在取数及执行过程中需要用到 的是( )。

I. 通用寄存器组 (GPRs) II. 算术逻辑单元(ALU)

Ⅲ . 存 储 器(M emory ) IV. 指令译码器 (ID)

A. 仅 I 、Ⅱ B. 仅 I 、Ⅱ 、Ⅲ C. 仅Ⅱ、HⅢ、IV D. 仅 I 、Ⅲ 、IV

**26.【**2021 统考真题】下列寄存器中，汇编语言程序员可见的是()。

I. 指令寄存器 II. 微指令寄存器

Ⅲ .基址寄存器 IV. 标志/状态寄存器

A. 仅 I 、Ⅱ B. 仅 I 、IV C. 仅Ⅱ、IV D. 仅 Ⅲ 、IV

**二、综合应用题**

**01.** 若某机主频为200MHz, 每个指令周期平均为2.5个 CPU 周期，每个 CPU 周期平均包 括2个主频周期，问：

1)该机平均指令执行速度为多少 MIPS?

2)若主频不变，但每条指令平均包括5个 CPU 周期，每个 CPU 周期又包含4个主频

周期，平均指令执行速度又为多少 MIPS?

3)由此可得出什么结论?

**02.** 某机有80条指令，平均每条指令由4条微指令组成(包含取指微指令),其中有一条取

指微指令是所有指令公用的。已知微指令长度为32位，请估算控制存储器CM 容量。

**03.** 某微程序控制器中，采用水平型直接控制(编码)方式的微指令格式，后续微指令地址

由微指令的后继地址字段给出。已知机器共有28个微命令，6个互斥的可判定的外部条 件，控制存储器的容量为512×40位。试设计其微指令的格式，并说明理由。

**04.** 某机共有52个微操作控制信号，构成5个相斥类的微命令组，各组分别包含5、8、2、 15、22个微命令。已知可判定的外部条件有两个，微指令字长28位。

CPU

W

微操作命令形成部件

R

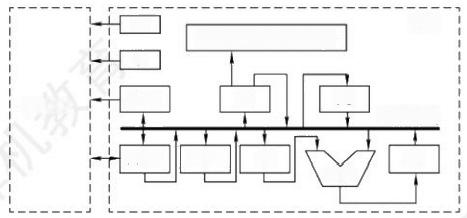
第5章 中央处理器 243

1)按水平型微指令格式设计微指令，要求微指令的后继地址字段直接给出后继微指

令地址。

2)指出控制存储器的容量。

**05.** 设 CPU 中各部件及其相互连接关系如下图所示，其中W 是写控制标志； R 是读控制标 志； R1 、R2 是暂存器。



存储器

内部总线

R2

ALU

MDR

MAR

ACC

PC

R1

IR

1)写出指令ADD#a(# 为立即寻址特征，隐含的操作数在ACC 寄存器中)在执行阶段

所完成的微操作命令及节拍安排。

2)假设要求在取指周期实现(PC)+1→PC, 且 由ALU 完成此操作(ALU 能对它的一个源 操作数完成加1运算)。以最少的节拍写出取指周期全部微操作命令及节拍安排。

**5.4.5** **答案与解析**

**一** **、单项选择题**

**01.D**

取指令阶段完成的任务是将现行指令从主存中取出并送至指令寄存器，这个操作是公共的操 作，是每条指令都要进行的，与具体的指令无关，所以不需要操作码的控制。

**02.B**

CU 的输入信号来源如下：①经指令译码器译码产生的指令信息；②时序系统产生的机器周 期信号和节拍信号；③来自执行单元的反馈信息即标志。前两者是主要因素。

**03.C**

执行公用的取指微程序从主存中取出机器指令后，由机器指令的操作码字段指出各个微程序 的入口地址(初始微地址)。

**04.D**

微指令的设计目标和指令结构的设计目标类似，都是基于执行速度、灵活性和指令长度这三

个主要方面考虑的。而控制存储器容量的大小与微指令的设计目标无关。

**05.** D

在微程序控制中，控制存储器中存放有微指令，在执行时需要从中读出相应的微指令，从而 增加了时间消耗。

**06.C**

字段直接编码方式为了缩短微指令字长而牺牲了速度，当微命令个数为4时需要3位，2位 会导致每个编码都输出一个微命令，而不能表示不输出，I 错误。Ⅱ正确。垂直型微指令的缺点 是微程序长、执行速度慢、工作效率低，Ⅲ错误。在字段间接编码方式中， 一个字段的某些微命 令要由另一个字段的某些微命令来解释，即依赖另一个字段的译码输出， IV 正确。

**07.C**

微程序控制存储器用来存放微程序，是微程序控制器的核心部件，属于 CPU 的一部分，而 不属于主存。

**08.D**

硬布线控制器采用硬件电路，速度较快，但设计难度大、成本高。微程序控制器的速度较慢， 但灵活性高。通常控制存储器采用ROM 组成。微指令计数器决定了微指令执行的顺序。

**09.B**

硬布线控制器需要结合各个微操作的节拍安排，综合分析，写出逻辑表达式，再设计成逻辑电 路图，因此时序系统比较复杂；而微程序只需按照节拍的安排，顺序执行微指令，因此比较简单。

10.D

在微程序控制器中，控制部件向执行部件发出的控制信号称为微命令，微命令执行的操作称 为微操作。微指令则是若干微命令的集合，若干微指令的有序集合称为微程序。

**11.B**

在一个 CPU 周期中， 一组实现一定功能的微命令的组合构成一条微指令，有序的微指令序 列构成一段微程序，微程序的作用是实现一条对应的机器指令。

12.B

一条水平型微指令能定义并执行几种并行的基本操作； 一条垂直型微指令只能定义并执行一 种基本操作。

**13.C**

垂直型微指令是一种微指令格式，相比于水平型微指令而言的，并不是指令格式垂直表示， 在微指令中设置了微操作码字段，结构类似于机器指令格式。控制信号经过编码产生是一种控制 字段的编码方法，属于水平型微指令，强调并行控制功能是一种控制字段的设计目标，适合水平 型微指令而不适合垂直型微指令。

**14.A**

在同一个 CPU 周期中，可以同时出现的微命令叫相容性微命令，不允许同时出现的微命令 叫互斥性微命令。不允许同时出现的原因有可能是会引起总线冲突，也有可能是其他原因。

**15.B**

微处理器是相对于一些大型处理器而言的，与微程序控制器没有必然联系。不管是采用微程 序控制器，还是采用硬布线控制器，微机的CPU 都是微处理器，I 错误。微程序的设计思想就是 将每条机器指令编写成一个微程序，每个微程序包含若干微指令，每条微指令对应一个或几个微 操作命令，Ⅱ正确。直接编码方式中每位代表一个微命令，不需要译码，因此执行效率最高，只 是这种方式会使得微指令的位数大大增加，Ⅲ错误。 一条水平型微指令能定义并执行几种并行的 基本操作，因此能够更充分利用数据通路的并行结构， IV 正确。

**16.C**

微程序控制方式采用编程方式来执行指令，而硬布线方式则采用硬件方式来执行指令，因此 硬布线方式速度较快，I 错 误。μPC 无法取代PC, 因为它只在微程序中指向下一条微指令地址的 寄存器。因此它也必然不可能知道这段微程序执行完毕后下一条是什么指令，Ⅱ错误。由于每条 微指令执行时所发出的控制信号是事先设计好的，不需要改变，因此存放所有控制信号的存储器 应为 ROM,Ⅲ 正确。指令周期是从一条指令启动到下一条指令启动的间隔时间，而CPU 周期是 机器周期，是指令执行中每步操作所需的时间， IV 错误。

17.A

一条指令对应一个微程序，所以一个微程序的周期对应一个指令周期。

第5章 中央处理器 245

**18.B**

CPU 控制器主要由三个部件组成：指令寄存器、程序计数器和操作控制器。状态条件寄存器 通常属于运算器的部件，保存由算术指令和逻辑指令运行或测试的结果建立的各种条件码内容， 如运算结果进位标志 (CF) 、 运算结果溢出标志( VF) 等。

**19.C**

断定法是指在微指令(后继地址字段)中直接明确指出下一条微指令的地址，这样相当于每 条都是转移微指令，此外，还有一些其他如条件测试和转移控制字段，也用于控制微指令的寻址。 因此，后继微指令地址可由微程序设计者指定，或者根据微指令所规定的转移控制字段控制产生。

**20.D**

微程序控制器采用了“存储程序”的原理，每条机器指令对应一个微程序，因此修改和扩 充容易，灵活性好，但每条指令的执行都要访问控制存储器，所以速度慢。硬布线控制器采用 专门的逻辑电路实现，其速度主要取决于逻辑电路的延迟，因此速度快，但修改和扩展困难， 灵活性差。

**21.** C

字段直接编码法将微命令字段分成若干小字段，互斥性微命令组合在同一字段中，相容性微 命令分在不同字段中，每个字段还要留出一个状态，表示本字段不发出任何微命令。5个互斥类， 分别包含7、3、12、5和6个微命令，需要3、2、4、3和3位，共15位。

22.C

计算机共有32条指令，各个指令对应的微程序平均为4条，则指令对应的微指令为32×4=128 条，而公共微指令还有2条，整个系统中微指令的条数共为128+2=130条，所以需要[log₂130]=

8位才能寻址到130条微指令。

23.B

主存储器 (MM) 在 CPU 外，用于存储指令和数据，由RAM 和 ROM 实 现 ( 主 要 是RAM) 。 控制存储器 (CS) 用来存放构成指令系统的所有微指令，是一种只读型存储器，机器运行时只读 不写，在CPU 的控制器内。控制存储器按照微指令的地址访问。

24.D

时钟脉冲信号的宽度称为时钟周期，时钟周期的倒数为机器主频。时钟脉冲信号由机器脉冲 源发出的脉冲信号经整形和分频后形成，时钟周期以相邻状态单元间组合逻辑电路的最大延迟为 基准确定。对于单周期CPU, 一个指令周期就是一个时钟周期，每个时钟周期执行一条新指令； 对于多周期CPU, 每个指令周期(包含若干时钟周期)执行一条新指令；对于流水线 CPU, 只 有 在理想情况下才能实现每个时钟周期执行一条新指令，选项D 的描述有误。

25.B

该指令的两个源操作数分别采用寄存器、寄存器间接寻址方式，因此在取数阶段需要用到通 用寄存器组 (GPRs) 和存储器 (Memory); 在执行阶段，两个源操作数相加需要用到算术逻辑单 元 (ALU) 。 而指令译码器 (ID) 用于对操作码字段进行译码，向控制器提供特定的操作信号， 在取数及执行阶段用不到。

**26.D**

汇编程序员可见的寄存器有基址寄存器(用于实现多道程序设计或者编制浮动程序)和状态 /标志寄存器、程序计数器PC 及通用寄存器组；而MAR、MDR、IR 是 CPU 的内部工作寄存器， 对汇编程序员不可见。微指令寄存器属于微程序控制器的组成部分，它是硬件设计者的任务，对 汇编程序员是透明的(不可见的)。

246-2025年计算机组成原理考研复习指导

**二、综合应用题**

**01.【**解答】

1)主频为200MHz, 所以主频周期=1/200MHz=0.005μs。

每个指令周期平均为2.5个CPU 周期，每个 CPU 周期平均包括2个主频周期，所以一条 指令的执行时间=2×2.5×0.005μs=0.025μs。

该机平均指令执行速度=1/0.025=40MIPS。

2)每条指令平均包括5个CPU 周期，每个 CPU 周期又包含4个主频周期，所以一条指令 的执行时间=4×5×0.005μs=0.1μs。

该机平均指令执行速度=1/0.1=10MIPS

3)由此可见，指令的复杂程度会影响平均指令执行速度。

**02.【**解答】

总的微指令条数=(4-1)×80+1=241条，每条微指令占一个控制存储器单元，控制存储器 CM 的容量为2的n 次幂，而241刚好小于256,所以CM 的容量=256×32位=1KB。

**03.**【解答】

水平型微指令由操作控制字段、判别测试字段和后继地址字段三部分构成。因为微指令采用 直接控制(编码)方式，所以其操作控制字段的位数等于微命令数，为28 位。又由于后继微指 令地址由后继地址字段给出，因此其后继地址字段的位数可根据控制存储器的容量(512×40位) 确定为9位(512=29)。当微程序出现分支时，后续微指令地址的形成取决于状态条件——6个 互斥的可判定外部条件，因此状态位应编码成3位。非分支时的后续微指令地址由微指令的后继 地址字段直接给出。微指令的格式如下图所示。

|  |  |  |
| --- | --- | --- |
| 操作控制字段 | 判别测试字段 | 后继地址字段 |

28位 3位 9位

**04.【**解答】

1)根据5个互斥类的微命令组，各组分别包含5、8、2、15、22个微命令，考虑到每组必 须增加一种不发送命令的情况，条件测试字段应包含一种不转移的情况，则5个控制字 段分别需给出6、9、3、16、23种状态，对应3、4、2、4、5位(共18位),条件测试 字段取2位。根据微指令字长为28位，后继地址字段取28- 18-2=8位，则其微指令 格式如下图所示。

5个

微命令

8个

微命令

2个

微命令

15个

微命令

22个

微命令

2个

判断条件

后继地址

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | 条件测试 | 后继地址 |

3位 4位 2位 4位 5位 2位 8位

2)根据后继地址字段为8位，微指令字长为28位，得出控制存储器的容量为28×28位。 **05.【**解答】

1 ) 含 有ACC 的立即寻址， 一个操作数隐藏在 ACC 中，立即寻址的加法指令执行周期的微 操作命令及节拍安排如下：



T₀ Ad(IR)→R1

T ₁ (R1) +(ACC)→R2

T₂(R2)→ACC

立即数→R1

Acc 通过总线送 ALU 结 果 →ACC

2)由于(PC)+1→PC 需由ALU 完成，因此 PC 的值可作为 ALU 的一个源操作数，在ALU

第5章 中央处理器- 247

做加1运算得到(PC)+1 后，结果送至与ALU 输出端相连的 R2, 然后送至 PC。

此题的关键是要考虑总线冲突的问题，因此，取指周期的微操作命令及节拍安排如下：

Pc 通过总线送MAR

PC 通过总线送 ALU 完成加1 MDR 通过总线送 IR

R2 通过总线送 Pc

T₀ (PC)→MAR,1→R

T₁M(MAR)→MDR, ( PC)+1→R2

T₂ (MDR)→IR,OP(IR)→ 微操作命令形成部件

T₃ (R2 )→PC



**5.5** **异常和中断机制**

现代计算机中都配有完善的异常和中断处理系统，CPU 的数据通路中有相应的异常检测和响 应逻辑，外设接口中有相应的中断请求和控制逻辑，操作系统中有相应的中断服务程序。这些中 断硬件电路和中断服务程序有机结合，共同完成异常和中断的处理过程。

**5.5.1** **异** **常** **和** **中** **断** **的** **基** **本** **概** **念**

**命题追踪▶异常事件的性质(2015)**

由 CPU 内部产生的意外事件被称为异常，有些教材中也称内中断。由来自CPU 外部的设备 向 CPU 发出的中断请求被称为中断，通常用于信息的输入和输出，有些教材中也称外中断。异常 是 CPU 执行一条指令时，由CPU 在其内部检测到的、与正在执行的指令相关的同步事件；中断 是一种典型的由外部设备触发的、与当前正在执行的指令无关的异步事件。

命 题 追 踪▶ **异常响应的时机(2023)**

异常和中断处理过程的描述如下：若 CPU 在执行用户程序的第i 条指令时检测到一个异 常事件，或者执行第i 条指令后发现一个中断请求信号，则 CPU 打断当前程序，然后转去执 行相应的异常或中断处理程序。若异常或中断处理程序能够解决相应的问题，则在异常或中 断处理程序的最后， CPU 通过执行异常或中断返回指令，回到被打断的用户程序的第 i 条 指 令或第i+1 条指令继续执行；若异常或中断处理程序发现是不可恢复的致命错误，则终止用 户程序。通常情况下，对异常和中断的具体处理过程由操作系统(和驱动程序)完成。

异常和中断的处理过程基本是相同的，这也是有些教材将两者统称为中断的原因。

**5.5.2** **异** **常** **和** **中** **断** **的** **分** **类**

**1.** **异常的分类**

异常是由 CPU 内部产生的意外事件，分为硬故障中断和程序性异常。硬故障中断是由硬连 线出现异常引起的，如存储器校验错、总线错误等。程序性异常也称软件中断，是指在CPU 内部 因执行指令而引起的异常事件。如整除0、溢出、断点、单步跟踪、非法指令、栈溢出、地址越 界、缺页等。按异常发生原因和返回方式的不同，可分为故障、自陷和终止。

(1)故障 (Fault)

命 题 追 踪 ▶ **异常或中断处理后指令重新执行的断点(2021)**

指在引起故障的指令启动后、执行结束前被检测到的异常事件。例如，指令译码时，出现“非 法操作码”;取数据时，发生“缺段”或“缺页”;执行整数除法指令时，发现“除数为0”等。

248-2025年计算机组成原理考研复习指导

对于“缺段”“缺页”等异常事件，经处理后，可将所需的段或页面从磁盘调入主存，回到发生 故障的指令继续执行， 断点为当前发生故障的指令；对于“非法操作码”“除数为0”等，因为无 法通过异常处理程序恢复故障，因此不能回到原断点执行，必须终止进程的执行。

(2)自陷 (Trap)

命 题 追 踪 ▶ **自陷的原理和性质(2020)**

自陷也称陷阱或陷入，它是预先安排的一种“异常”事件，就像预先设定的“陷阱”一样。

通常的做法是：事先在程序中用一条特殊指令或通过某种方式设定特殊控制标志来人为设置一个 “陷阱”,当执行到被设置了“陷阱”的指令时，CPU 在执行完自陷指令后，自动根据不同“陷阱” 类型进行相应的处理，然后返回到自陷指令的下一条指令执行。注意，当自陷指令是转移指令时，

并不是返回到下一条指令执行，而是返回到转移目标指令执行。

在x86 机器中，用于程序调试“断点设置”和单步跟踪的功能就是通过陷阱机制实现的。此 外，系统调用指令、条件自陷指令(如 MIPS 中的 teq、teqi、tne、tnei 等)等都属于陷阱指令， 执行到这些指令时，无条件或有条件地自动调出操作系统内核程序进行执行。

故障异常和自陷异常属于程序性异常(软件中断)。

(3)终止 (Abort)

如果在执行指令的过程中发生了使计算机无法继续执行的硬件故障，如控制器出错、存储器 校验错、总线错误等，那么程序将无法继续执行，只能终止，此时，调出异常服务程序来重启系 统。这种异常与故障和自陷不同，不是由特定指令产生的，而是随机发生的。

终止异常和外中断属于硬件中断。

**2.** **中断的分类**

命 题 追 踪▶ **对中断和异常事件的判断(2009、2016、2020)**

中断是指来自CPU 外部、与CPU 执行指令无关的事件引起的中断，包括I/O 设备发出的I/O 中断(如键盘输入、打印机缺纸等),或发生某种特殊事件(如用户按 Esc 键、定时器计数时间到) 等。外部I/O 设备通过特定的中断请求信号线向 CPU 提出中断请求， CPU 每执行完一条指令就 检查中断请求信号线，如果检测到中断请求，则进入中断响应周期。

中断可分为可屏蔽中断和不可屏蔽中断。

(1)可屏蔽中断

指通过可屏蔽中断请求线 INTR 向 CPU 发出的中断请求。CPU 可以通过在中断控制器中设 置相应的屏蔽字来屏蔽它或不屏蔽它，被屏蔽的中断请求将不被送到CPU。

(2)不可屏蔽中断

指通过专门的不可屏蔽中断请求线NMI 向 CPU 发出的中断请求，通常是非常紧急的硬件故 障，如电源掉电等。这类中断请求信号不可被屏蔽，以让CPU 快速处理这类紧急事件。

中断和异常在本质上是一样的，但它们之间有以下两个重要的不同点：

1)“缺页”或“溢出”等异常事件是由特定指令在执行过程中产生的，而中断不和任何指令 相关联，也不阻止任何指令的完成。

2)异常的检测由CPU 自身完成，不必通过外部的某个信号通知 CPU 。对于中断， CPU 必 须通过中断请求线获取中断源的信息，才能知道哪个设备发生了何种中断。



**注** **意**

所有的异常和中断事件都是由硬件检测发现的。

第5章 中央处理器一 249

此外，根据识别中断服务程序地址的方式，可分为向量中断和非向量中断；根据中断处理过 程是否允许被打断，还可分为单重中断和多重中断。

**5.5.3异** **常** **和** **中** **断** **响** **应** **过** **程**

CPU 执行指令时，如果发生了异常或中断请求，必须进行相应的处理。从 CPU 检测到异常 或中断事件，到调出相应的处理程序，整个过程称为异常和中断响应。CPU 对异常和中断响应的 过程可分为关中断、保存断点和程序状态、识别异常和中断并转到相应的处理程序。

(1)关中断

在保存断点和程序状态期间，不能被新的中断打断，因此要禁止响应新的中断，即关中断。 通常通过设置“中断允许”(IF) 触发器来实现，若IF 置为1,则为开中断，表示允许响应中断； 若 IF 置为0,则表示关中断，表示不允许响应中断。

(2)保存断点和程序状态

为了能在异常和中断处理后正确返回到被中断的程序继续执行，必须将程序的断点(返回地 址)送到栈或特定寄存器中。通常保存在栈中，这是为了支持异常或中断的嵌套。

异常和中断处理后可能还要回到被中断的程序继续执行，被中断时的程序状态字寄存器 PSW 的内容也需要保存在栈或特定寄存器中，在异常和中断返回时恢复到PSW 中。

(3)识别异常和中断并转到相应的处理程序

异常和中断源的识别有软件识别和硬件识别两种方式。异常和中断源的识别方式不同，异常 大多采用软件识别方式，而中断可以采用软件识别方式或硬件识别方式。

软件识别方式是指 CPU 设置一个异常状态寄存器，用于记录异常原因。操作系统使用一个 统一 的异常或中断查询程序，按优先级顺序查询异常状态寄存器，以检测异常和中断类型，先查 询到的先被处理，然后转到内核中相应的处理程序。

硬件识别方式也称向量中断，异常或中断处理程序的首地址称为中断向量，所有中断向量都 存放在中断向量表中。每个异常或中断都被指定一个中断类型号。在中断向量表中，类型号和中 断向量一一对应，因而可以根据类型号快速找到对应的处理程序。

整个响应过程是不可被打断的。中断响应过程结束后，CPU 就 从PC 中取出对应中断服务程 序的第一条指令开始执行，直至中断返回，这部分任务是由CPU 通过执行中断服务程序完成的， 整个中断处理过程是由软/硬件协同实现的。

**5.5.4** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 以下关于“自陷”(Trap) 异常的叙述中，错误的是()。

A. “自陷”是人为预先设定的一种特定处理事件

B. 可由访管指令或自陷指令的执行进入“自陷”

C. 一定是出现某种异常情况才会发生“自陷”

D. “自陷“发生后CPU 将进入操作系统内核程序并执行

**02.** 指令执行结果出现异常而引起的中断是()。

A.I/O 中断 B . 机器校验中断 C. 故 障 D. 外部中断

**03.** 访问主存时发生的校验错误属于()。

A. 故障 B. 自陷 C. 终 止 D. 外中断

**04.** 下列关于异常和中断响应的叙述中，错误的是()。

250-2025年计算机组成原理考研复习指导

A. 异常事件检测由CPU 在执行每一条指令的过程中进行

B. 中断请求检测由 CPU 在每条指令执行结束、取下条指令之前进行

C.CPU 检测到异常事件后所做的处理和检测到中断请求后所做的处理完全相同

D.CPU 在中断响应时会关中断、保存断点和程序状态并转到相应的中断服务程序 **05.** 以下给出的事件中，无须异常处理程序进行处理的是( )。

A. 缺页故障 B.Cache 缺失 C. 地址越界 D. 除数为0

**06.** CPU 响应中断的时间是( )。

A. 一条指令执行结束 B.I/O 设备提出中断

C. 取指周期结束 D. 指令周期结束

**07** . 下列选项中，不属于外部中断事件的是()。

A. 采样定时到 B. 无效操作码 C. 打印机缺纸 D. 键盘缓冲满

**08.** 下列关于异常/中断机制与进程上下文切换机制的叙述中，错误的是()。 A. 进程上下文切换和异常/中断响应两者都会产生异常控制流

B. 进程上下文切换后， CPU 执行的是另一个进程的代码

C. 响应异常/中断请求后，CPU 执行的是内核程序的代码

D. 进程上下文切换和异常/中断响应处理都通过执行内核程序实现

**09** .异常或中断处理结束后，返回到被中断原程序继续执行的指令地址称为“断点”,以下 关于“断点”的说法中，错误的是( )。

A. “陷阱”类异常的断点为陷阱指令下一条指令的地址

B. “故障”类异常的断点为当前发生异常的指令的地址

C. 外部中断的断点总是当前刚执行完的指令的地址

D. “终止”类异常的断点可以是当前指令或下一条指令的地址

**10.**【2015统考真题】内部异常(内中断)可分为故障 (fault) 、 陷阱 (trap) 和终止 (abort)

三类。下列有关内部异常的叙述中，错误的是( )。

A. 内部异常的产生与当前执行指令相关

B. 内部异常的检测由 CPU 内部逻辑实现

C. 内部异常的响应发生在指令执行过程中

D. 内部异常处理后返回到发生异常的指令继续执行

**11.【**2016统考真题】异常是指令执行过程中在处理器内部发生的特殊事件，中断是来自处 理器外部的请求事件。下列关于中断或异常情况的叙述中，错误的是()。

A. “访存时缺页”属于中断 B. “整数除以0”属于异常

C.“DMA 传送结束”属于中断 D. “存储保护错”属于异常

**12.【**2020统考真题】下列关于“自陷”(Trap, 也称陷阱)的叙述中，错误的是()。

A. 自陷是通过陷阱指令预先设定的一类外部中断事件

B. 自陷可用于实现程序调试时的断点设置和单步跟踪

C. 自陷发生后 CPU 将转去执行操作系统内核相应程序

D. 自陷处理完成后返回到陷阱指令的下一条指令执行

**13.**【2021 统考真题】异常事件在当前指令执行过程中进行检测，中断请求则在当前指令执 行后进行检测。下列事件中，相应处理程序执行后，必须回到当前指令重新执行的是( )。

A. 系统调用 B. 页缺失 C.DMA 传送结束 D. 打印机缺纸

第5章 中央处理器 251

**5.5.5** **答案与解析**

**单项选择题**

**01.C**

自陷是人为设定的特殊中断机制，不是出现某些异常情况而产生的， C 错误。 **02.** C

异常是 CPU 执行指令过程中发生的与当前指令执行有关的意外事件，而中断请求则是 CPU 外部的I/O 部件或时钟等向CPU 发出的与当前指令执行无关的意外事件。指令执行结果出现异常 与当前指令执行有关，如运算溢出等，属于内中断中的故障。

**03.C**

如果在执行指令的过程中发生严重错误，如控制器出错、存储器校验错等，则程序将无法继 续执行，只能终止。严重情况下，甚至要调出中断服务程序来重启系统。

**04.** C

CPU 对于异常和中断的响应处理大体是一致的，都需要保存断点和程序状态字并转到相应的 处理程序去执行，但有些细节并不一样。例如，检测到中断请求后， CPU 必须通过“中断回答” 信号启动中断控制器进行中断查询，以确定当前发出的优先级最高的中断请求，并通过数据线获 取相应的中断类型号；而对于异常， CPU 无须进行中断回答。

**05.B**

缺页、地址越界和除数为0都是执行某条指令时可能发生的故障，需要调出操作系统内核中 相应的异常处理程序来处理，而Cache 缺失则由CPU 硬件实现，无须调出异常处理程序进行处理。

**06.** A

中断周期用于响应中断，若有中断，则在指令的执行周期后进入中断周期。

**07.B**

无效操作码是由 CPU 在对某条指令译码时发现的，因而是内部异常。采样定时时间到、打 印机缺纸、键盘缓冲满都与当前指令的执行无关，是由CPU 外部的中断源发出的中断请求。

**08.D**

在硬件层，CPU 中有检测异常和中断事件并将控制转移到操作系统内核执行的机制；在操作 系统层，内核能通过进程的上下文切换将一个进程的执行转移到另一个进程的执行，它们都会产 生异常控制流。响应异常/中断请求后， CPU 执行的是异常/中断服务程序，是操作系统的内核程 序。进程上下文切换由操作系统的内核程序实现，而异常/中断的响应则由硬件实现。

**补充知识：**CPU 所执行指令的地址序列称为 CPU 的控制流。在程序正常执行时，通过顺序 执行指令或转移指令得到的控制流称为正常控制流。在正常执行过程中，因遇到异常或中断事件 而引起用户程序的正常执行被打断所形成的意外控制流，称为异常控制流。

**09.C**

外部中断请求信号的检测总是在一条指令执行完之后，取下一条指令之前。因此，如果检测 到有外部中断请求，那么响应中断请求并转到中断服务程序执行后，应返回到原来被中断的程序 中已经执行完成的指令的下一条指令执行，而不返回到刚执行完的指令执行。

**10.D**

内部异常是指来自 CPU 内部产生的中断，如非法指令、地址非法、校验错、页面失效、运



252-2025年计算机组成原理考研复习指导

算溢出和除数为零等，以上都是在指令的执行过程中产生的，A 正确。内部异常的检测是由CPU 自身完成的，不必通过外部的某个信号通知CPU,B 正 确。内部异常不能被屏蔽， 一旦出现应立 即处理， C 正确。对于非法指令、除数为零等异常，无法通过异常处理程序恢复故障，因此不能 回到原断点执行，必须终止进程的执行，D 错误。

**11.A**

中断是指来自CPU 执行指令以外的事件，如设备发出的I/O 结束中断，表示设备输入/输出 已完成，希望处理机能向设备发出下一个输入/输出请求，同时让完成输入/输出后的程序继续运 行。异常也称内中断，指源自CPU 执行指令内部的事件。

**12.A**

自陷是一种内部异常，A 错误。在 x86 中，用于程序调试的“断点设置”功能是通过自陷机 制实现的，B 正确。执行到自陷指令时，无条件或有条件地自动调出操作系统内核程序进行执行， C 正确。CPU 执行陷阱指令后，会自动地根据不同陷阱类型进行相应的处理，然后返回到陷阱指 令的下一条指令执行， D 正确。

**13.B**

系统调用属于自陷，“断点”为自陷指令的下一条指令地址。DMA 传送结束后， DMA 控制 器需要向CPU 发送中断请求，属于外中断，外中断的“断点”为下一条指令地址。打印机缺纸同 样属于外部中断。页缺失属于内部异常中的故障，“断点”为发生故障的指令地址，执行完缺页 异常处理程序之后必须返回发生故障的指令重新执行。



**5.6** **指令流水线**

前面介绍的指令都是在单周期处理机中采用串行方法执行的，同一时刻 CPU 中只有一条指令 在执行，因此各功能部件的使用率不高。现代计算机普遍采用指令流水线技术，同一时刻有多条指 令在CPU 的不同功能部件中并发执行，大大提高了功能部件的并行性和程序的执行效率。

**5.6.1** **指令流水线的基本概念**

可从两方面提高处理机的并行性：①时间上的并行技术，将一个任务分解为几个不同的子阶 段，每个子阶段在不同的功能部件上并行执行，以便在同一时刻能够同时执行多个任务，进而提 升系统性能，这种方法被称为流水线技术。②空间上的并行技术，在一个处理机内设置多个执行 相同任务的功能部件，并让这些功能部件并行工作，这样的处理机被称为超标量处理机。

一条指令的执行过程可分解为若干阶段，每个阶段由相应的功能部件完成。如果将各阶段视 为相应的流水段，则指令的执行过程就构成了一条指令流水线。

假设一条指令的执行过程分为如下5个阶段(也称功能段或流水段)°:

● 取指 (IF): 从指令存储器或 Cache 中取指令。

●译码/读寄存器 (ID): 操作控制器对指令进行译码，同时从寄存器堆中取操作数。 ·执行/计算地址 (EX): 执行运算操作或计算地址。

●访存 (MEM): 对存储器进行读/写操作。

● 写回 (WB): 将指令执行结果写回寄存器堆。

① 不同的教材有不同的划分举例，本书参考了历年统考真题中的划分。

第5章 中央处理器 253

把 k+1 条指令的取指阶段提前到第k 条指令的译码阶段，从而将第k+1 条指令的译码阶段

与第k 条指令的执行阶段同时进行，如图5. 16所示。

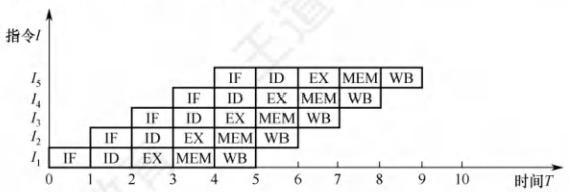


图5.16 一个5段指令流水线

从图5. 16看出，理想情况下，每个时钟周期都有一条指令进入流水线，每个时钟周期都有一

条指令完成，每条指令的时钟周期数 ( 即 CPI) 都 为 1 。

命题追踪 流水线对指令集的要求(2011)

为了利于实现指令流水线，指令集应具有如下特征：

1)指令长度应尽量一致，有利于简化取指令和指令译码操作。否则，取指令所花的时间长

短不一 ，使得取指部件极其复杂，并且也不利于指令译码。

2)指令格式应尽量规整，尽量保证源寄存器的位置相同，有利于在指令未知时就可取寄存

器操作数，否则须译码后才能确定指令中各寄存器编号的位置。

3 ) 采 用LOAD/STORE 型指令，其他指令都不能访问存储器，这样可把LOAD/STORE 指 令

的地址计算和运算指令的执行步骤规整在同一个周期中，有利于减少操作步骤。

4)数据和指令在存储器中“按边界对齐”存放。这样，有利于减少访存次数，使所需数据

在一个流水段内就能从存储器中得到。

**5.6.2** **流水线的基本实现**

**1.** **流水线设计的原则**

在单周期实现中，虽然不是所有指令都必须经历完整的5个阶段，但只能以执行速度最慢的 指令作为设计其时钟周期的依据，单周期CPU 的时钟频率取决于数据通路中的最长路径。

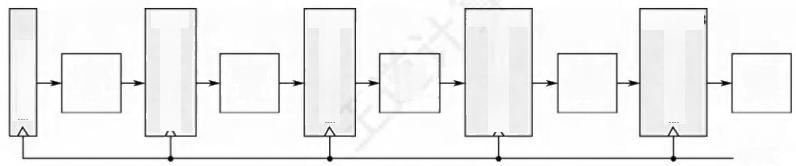
命 题 追 踪 ▶ **流水线时钟周期的设计(2009)**

流水线设计的原则： ①指令流水段个数以最复杂指令所用的功能段个数为准；②流水段的 长度以最复杂的操作所花的时间为准。假设某条指令的5个阶段所花的时间分别如下。①取指： 200ps;② 译码：100ps;③ 执行：150ps;④ 访存：200ps;⑤ 写回：100ps, 该指令的总执行时 间为750ps。按照流水线设计原则，每个流水段的长度为200ps, 所以每条指令的执行时间为 Ins, 反而比串行执行时增加了250ps 。 假设某程序有N 条指令，单周期处理机所用的时间为N×750ps, 而流水线处理机所用的时间为(N+4)×200ps。 由此可见，流水线方式并不能缩短单条指令的执行 时间，但对于整个程序来说，执行效率得到了大幅提高。

**2.** **流水线的逻辑结构**

每个流水段后面都要增加一个流水段寄存器，用于锁存本段处理完的所有数据，以保证本段 的执行结果能在下个时钟周期给下一流水段使用，如图5. 17所示。各种寄存器和数据存储器均采 用统一 时钟 CLK 进行同步，每来一个时钟，各段处理完的数据都将锁存到段尾的流水段寄存器 中，作为后段的输入。同时，当前段也会收到前段通过流水段寄存器传递过来的数据。

254-2025年计算机组成原理考研复习指导



MEMWE

访存

MEM

CLK

流 水 段 寄 存 器

流水段寄存器

流水段寄存器

流水段寄存器

程序计数器

译码 ID

写回 WB

取指

IF

执行 EX

EX/MEM

IF/ID

ID/EX

图5.17 流水线的逻辑结构图

一条指令会依次进入IF、ID、EX、MEM、WB 五个功能段进行处理，第一条指令进入 WB 段后，各流水段都包含一条不同的指令，流水线中将同时存在5条不同的指令并行执行。

命 题 追 踪 ▶ **存在流水段寄存器时延的时钟周期的设计(2018)**



注 意

在考试中，若没有明确说明，则可以不用考虑流水寄存器的时延。

**3.** **流水线的时空图表示**

通常用时空图来直观地描述流水线的执行情况，如图5.18所示。

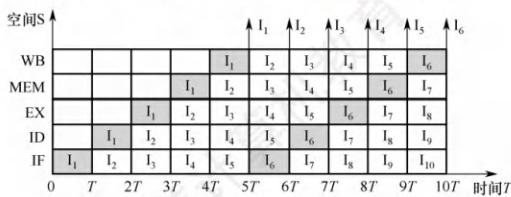


图5.18 一 个5段指令流水线时空图

在时空图中，横坐标表示时间，它被分割成长度相等的时间段T; 纵坐标为空间，表示当前 指令所处的功能部件。在图5.18中，第一条指令I₁ 在时刻0进入流水线，在时刻5T 流出流水线。 第二条指令I₂ 在时刻T 进入流水线，在时刻6T 流出流水线。以此类推，每隔一个时间T 就有一 条指令进入流水线，从时刻5T 开始每隔一个时间T 就有一条指令流出流水线。

命题追踪 **▶流水线执行4条指令所需的时钟周期数(2012)**

从图5.18中可看出，在时刻10T 时，流水线上便有6条指令流出。若采用串行方式执行，在 时刻10T时，只能执行2条指令，可见使用流水线方式成倍地提高了计算机的速度。

只有大量连续任务不断输入流水线，才能充分发挥流水线的性能，而指令的执行正好是连续 不断的，非常适合采用流水线技术。对于其他部件级流水线，如浮点运算流水线，同样也仅适合 于提升浮点运算密集型应用的性能，对于单个运算是无法提升性能的。

**5.6.3** **流水线的冒险与处理**

命 题 追 踪▶ **导致流水线阻塞的各种原因(2010)**

在指令流水线中，可能会遇到一些情况使得后续指令无法正确执行而引起流水线阻塞，这种 现象称为流水线冒险。根据导致冒险的原因不同分为结构冒险、数据冒险和控制冒险3种。

不同类型指令在各流水段的操作是不同的，表5.2中列出了几类指令在各流水段中的操作。

第5章 中央处理器 255

**表5.2不同类型指令在各流水段中的操作**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **流水段**  **指令** | **IF** | **ID** | **EX** | **MEM** | **WB** |
| ALU | 取指 | 译码  读寄存器堆 | 执行 | — | 结果写回寄存器堆 |
| 取/存 | 取指 | 译码  读寄存器堆 | 计算访存有效地址 | 访存(读写) | 将读出的数据写入寄 存器堆/一 |
| 转移 | 取指 | 译码  读寄存器堆 | 计算转移目的地址， 设置条件码 | 若条件成立，将转移 目的地址送PC |  |

这几类指令将会在下面介绍流水线冲突时涉及。

**1.** **结构冒险**

命 题 追 踪 ▶

**解决结构冒险的办法(2016)**

由不同指令在同一时刻争用同一功能部件而形成的冲突，也称资 源 冲突，即由硬件资源竞争 造成的冲突。例如，指令和数据通常都存放在同一存储器中，在第4个时钟周期，第i 条 LOAD 指令进入MEM 段时，第i+3 条指令的IF 段也要访存取指令，此时会发生访存冲突，为此可在 前一条指令访存时，暂停(一个时钟周期)取后一条指令的操作，如表5 .3 所示。当然，如果第 i 条指令不是 LOAD 指令，在 MEM 段不访存，也就不会发生访存冲突。

**表5.3** **用暂停后续指令的方法解决访存冲突**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **指令** | **时钟周期** | | | | | | | | |
| 1 | 1 | 2 | 3 | 4 | 5 | **6** | **8** | 9 |
| LOAD指令 | IF | ID | EX | MEM | WB |  |  |  |  |
| 指令i+1 |  | IF | ID | EX | MEM | WB |  |  |  |
| 指令i+2 |  |  | IF | ID | EX | MEM | WB |  |  |
| 指令i+3 |  |  |  | 停顿 | IF | ID | EX | MEM | WB |
| 指令i+4 |  |  |  |  |  | IF | ID | EX | MEM |

解决结构冲突有以下两种办法：

1)前 一 指令访存时，使后 一 条相关指令(及其后续指令)暂停 一个时钟周期。

2)设置多个独立的部件。例如，对于寄存器访问冲突，可将寄存器的读口和写口独立开来； 对于访存冲突，单独设置数据存储器和指令存储器。在现代 Cache 机 制 中 ，L1 级 Cache 通常采用数据Cache 和指令Cache 分离的方式，从而也就避免了资源冲突的发生。

**2.** **数据冒险**

命 题 追 踪▶ **分析指令之间的数据冒险(2012、2014、2016、2019、2023)**

数据冒险也称数据相关。引起数据冒险的原因是，后面指令用到前面指令的结果时，前面指 令的结果还没有产生。在以非乱序执行的流水线中，所有数据冒险都是由于前面指令写结果之前，

后面指令就需要读取而造成的，这种数据冒险称为写后读 (ReadAfter Write,RAW) 冲 突 。



**注** **意**

在按序执行“的流水线中(统考中通常采用这种方式),只可能出现 RAW 冲 突。

① 统考只考查过“**按序发射，按序完成”**的方式，即指令按顺序进入流水线，先流入的指令先流出流水线。

256-2025年计算机组成原理考研复习指导

例如，考虑下列两条指令：



Il add R1,R2,R3 #(R2)+(R3)→R1

I2 sub R4,R1,R5 #(R1)-(R5)→R4

在写后读 (RAW) 冲突中，指令12的源操作数是指令I1 的目的操作数。正常的读/写顺序是 由指令I1 先写入R1, 再由指令I2 来 读R1 。在非流水线中，这种先写后读的顺序是自然维持的。 但在流水线中，由于重叠操作，读/写的先后顺序关系发生了变化，如表5.4所示。

**表5.4** **add和sub 指令发生先写后读** **(RAW) 冲突**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 指令 | **时钟周期** | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 |
| add | IF | ID | EX | MEM | WB |  |
| sub |  | IF | ID | EX | MEM | WB |

读 R1 写R1

可以采用以下几种办法解决 RAW 数据冲突。

(1)延迟执行相关指令

把遇到数据相关的指令及其后续指令都暂停一至几个时钟周期，直到数据相关问题消失后再 继续执行，可分为软件插入空操作 “nop” 指令和硬件阻塞 (stall) 两种方法。

由表5.4可见，在第5个时钟周期， add 指令才将运算结果写入R1, 但后继 sub 指令在第3 个时钟周期就要从 R1 中读数，使先写后读的顺序改变为先读后写，发生了先写后读 (RAW) 的 数据冲突。如果不采取措施，按表5.4的读/写顺序，就会导致结果出错。为此，可以暂停 sub 指 令3个时钟周期，直至前面add 指令的结果生成，如表5.5所示。

**表5.5** **用延迟相关指令的办法来解决** **RAW** **冲突**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | **时钟周期** | | | | | | | | |
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 8 | 9 |
| add | IF | ID | EX | MEM | WB |  |  |  |  |
| sub |  | 阻塞 | 阻塞 | 阻塞 | IF | ID | EX | MEM | WB |

对于I1 和 I2 的数据相关问题，还可以通过将寄存器的写口和读口分别控制在前、后半个时 钟周期内操作，使前半周期写入R1 的值在后半周期马上被读出，在一个周期内读/写寄存器不会 产生冲突，这样II 的 WB 段 和I2 的 ID 段就可重叠执行，从而只需延迟2个时钟周期。

(2)采用转发(旁路)技术

设置相关转发通路，不等前一条指令把计算结果写回寄存器，下一条指令也不再从寄存器读， 而将数据通路中生成的中间数据直接转发到ALU 的输入端。如表5.6所示，指令I1 在 EX 段结 束时已得到R1 的新值，被存放到 EX/MEM 流水段寄存器中，因此可以直接从该流水段寄存器中 取出数据返送到ALU 的输入端，这样，在指令I2 执行时ALU 中用的就是 R1 的新值。

**表5.6用转发技术来解决RAW冲突**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 指令 | 时钟周期 | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 |
| add | IF | ID | EX | MEM | WB |  |
| sub |  | IF | ID | EX | MEM | WB |

增加转发通路后，相邻两条运算类指令之间、相隔一条的两个运算类指令之间的数据相关带

来的数据冒险问题就都能解决了。

第5章 中央处理器- 257

(3)load-use 数据冒险的处理

如 果load 指令与其后紧邻的运算类指令存在数据相关问题，则无法通过转发技术来解决，通 常把这种情况称为load-use 数据冒险。对于下列两条指令，由表5.2可知， load 指令只有在 MEM 段结束时才能得到主存中的结果，然后送 MEM/WB 流水段寄存器，在 WB 段的前半周期才能存 入 R2 的新值，但随后的add 指令在 EX 阶段就要取 R2 的值，因此，得到的是旧值。

I2 load r2,12(r1) #M[(r1 )+12]→(r2)

I3 add r4,r3,r2 #(r3)+(r2 )→(r4)

对 于 load-use 数据冒险，最简单的做法是由编译器在 add 指令之前插入一条 nop 指令，这样 在 add 指令的 EX 段就可以从MEM/WB 流水段寄存器中取出load 指令的最新结果，如表5.7所 示。当然，最好的办法是在程序编译时进行优化，通过调整指令顺序以避免出现 load-use 现 象。

**表5.7** **用延迟加转发技术来解决load-use冲突**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | 时钟周期 | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| add | IF | ID | EX | MEM | WB |  |  |
| sub |  | 阻塞 | IF | ID | EX | MEM | WB |

**3.** **控制冒险**

命 题 追 踪 分析指令之间的控制冒险(2014、2023)

指令通常是顺序执行的，但当遇到改变指令执行顺序的情况，例如执行转移或返回指令、发 生中断或异常时，会改变PC 值，从而造成断流，也称控制冲突。

对于由转移指令引起的冲突，最简单的处理方法就是推迟后续指令的执行。通常把因流水线 阻塞带来的延迟时钟周期数称为延迟损失时间片C 。在下列指令中，假设 R2 存放常数N,R1 的 初值为1, bne 指 令 在EX 段通过计算设置条件码，并在MEM 段确定是否将PC 值更新为转移目 的地址，因此仅当bne 指令执行到第5个时钟结束时才能将转移目标地址送PC 。为此，在数据通

路检测到分支指令后，可以在分支指令后插入C ( 此 处C=3) 条 nop 指令，如表5.8所示。

Il loop:add R1,R1,1 #(R1)+1→R1

I2 bne R1,R2,loop #if(R1)!=(R2) goto loop

**表5.8** **用插入空操作的办法解决控制冲突**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | **时钟周期** | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| add | IF | ID | EX | MEM | WB |  |  |  |  |  |
| bne |  | IF | ID | EX | MEM | WB |  |  |  |  |
| add |  |  |  |  |  | IF | ID | EX | MEM | WB |

有以下几种办法解决控制冲突：

1)对于由转移指令引起的冲突，可采用和解决数据冲突相同的软件插入 “nop” 指令和硬件 阻 塞 (stall) 的方法。比如，延迟损失多少时间片，就插入多少条 nop 指令。

2)对转移指令进行分支预测，尽早生成转移目标地址。分支预测分为简单(静态)预测和 动态预测。若静态预测的条件总是不满足，则按序继续执行分支指令的后续指令。动态 预测根据程序转移的历史情况，进行动态预测调整，有较高的预测准确率。



**注** **意**

Cache 缺失的处理过程也会引起流水线阻塞。

258-2025年计算机组成原理考研复习指导

5.6.4 流水线的性能指标

1. 流水线的吞吐率

命题追踪 流水线吞吐率的计算(2013)

流水线的吞吐率是指在单位时间内流水线所完成的任务数量，或输出结果的数量。 流水线吞吐率 (TP) 的最基本公式为



式中，n 是任务数， T 是处理完n 个任务所用的总时间。设k 为流水段的段数，△t 为时钟周期。 在输入流水线中的任务连续的理想情况下， 一条k 段流水线能在k+n-1 个时钟周期内完成n 个 任务。得出流水线的吞吐率为



连续输入的任务数n→~时，得到最大吞吐率为TPm=1/At。

2. 流水线的加速比

完成同样一批任务，不使用流水线与使用流水线所用的时间之比。

流水线加速比 (S) 的基本公式为



式中，T₀ 表示不使用流水线的总时间； T,表示使用流水线的总时间。 一条k 段流水线完成n 个任 务所需的时间为T=(k+n-1)At。 顺序执行n 个任务时，所需的总时间为T₀=kn△t。 将 T₀ 和 T 值代入上式，得出流水线的加速比为



连续输入的任务数n→~时，得最大加速比为 Smx=k。

5.6.5 高级流水线技术

有两种增加指令级并行的策略： 一种是多发射技术，它通过采用多个内部功能部件，使流水 线功能段能同时处理多条指令，处理机一次可以发射多条指令进入流水线执行；另一种是超流水 线技术，它通过增加流水线级数来使更多的指令同时在流水线中重叠执行。

1. 超标量流水线技术

命题追踪▶超标量流水线的特性(2017)

也称动态多发射技术，每个时钟周期内可并发多条独立指令，以并行操作方式将两条或多条 指令编译并执行，为此需配置多个功能部件，如图5.19所示。在简单的超标量 CPU 中，指令是 按顺序发射执行的。为了更好地提高并行性能，多数超标量CPU 都结合动态流水线调度技术，通 过动态分支预测等手段，指令不按顺序执行，这种方式称为乱序执行。

2. 超长指令字技术

也称静态多发射技术，由编译程序挖掘出指令间潜在的并行性，将多条能并行操作的指令组 合成一条具有多个操作码字段的超长指令字(可达几百位),为此需要采用多个处理部件。

**第5章** **中央处理器** 259

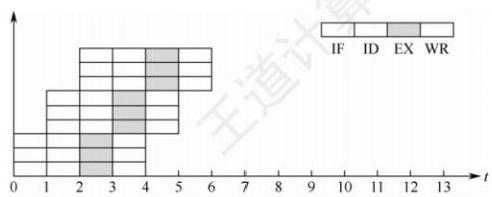


图5.19 超标量流水线技术

**3.** **超流水线技术**

如图5.20所示，流水线功能段划分得越多，时钟周期就越短，指令吞吐率也就越高，因此超 流水线技术是通过提高流水线主频的方式来提升流水线性能的。但是，流水线级数越多，用于流 水寄存器的开销就越大，因而流水线级数是有限制的，并不是越多越好。

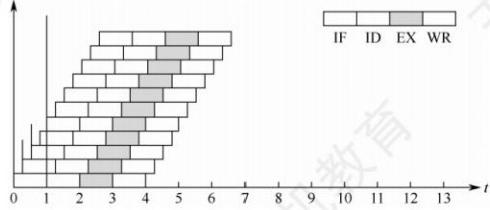


图5.20 超流水线技术

命题追踪 ▶ **基本流水线** **CPU** **和超标量流水线** **CPU** **的** **CPI(2020)**

超流水线 CPU 在流水线充满后，每个时钟周期还是执行一条指令， CPI=1, 但其主频更高； 多发射流水线CPU 每个时钟周期可以处理多条指令， CPI<1, 但其成本更高、控制更复杂。

**5.6.6** **本** **节** **习** **题** **精** **选**

**一、单项选择题**

**01.** 下列关于流水 CPU 基本概念的描述中，正确的是( )。

A. 流水 CPU 是以空间并行性为原理构造的处理器

B. 流水CPU一定是RISC 机器

C. 流水CPU一定是多媒体 CPU

D. 流水 CPU 是一种非常经济而实用的时间并行技术

02. 流水CPU 是由一系列称为“段”的处理电路组成的。 一个m 段流水线稳定时的 CPU 的 吞吐能力，与m 个并行部件的CPU 的吞吐能力相比，()。

A. 具有同等水平的吞吐能力 B. 不具备同等水平的吞吐能力

C. 吞吐能力大于前者的吞吐能力 D. 吞吐能力小于前者的吞吐能力

**03.** 设指令由取指、分析、执行3个子部件完成，并且每个子部件的时间均为△t, 若采用常 规标量单流水线处理机(即处理机的度为1),连续执行12条指令，共需()。

A.12△t B.14△t C.16△t D.18△t

**04.** 设指令由取指、分析、执行3个子部件完成，并且每个子部件的时间均为At, 若采用度 为4的超标量流水线处理机，连续执行20条指令，只需( )。

260-2025年计算机组成原理考研复习指导

A.3△t B.5△t C.7△t D.9△t

**05.** 设指令流水线把一条指令分为取指、分析、执行3部分，3部分执行时间不等长，且3部

分的时间分别是tm=2ns, t=2ns, =1ns, 则100条指令全部执行完毕需( ) 。

A.163ns B.183ns C.193ns D.203ns **06.** 下列关于指令流水线设计的叙述中，错误的是( )。

A. 指令执行过程中的各个子功能都需要包含在某个流水段中

B. 所有子功能都必须按一定的顺序经过流水段

C. 虽然各子功能所用实际时间可能不同，但经过每个流水段的时间都一样

D. 任何时候各个流水段的功能部件都不可能执行空操作

**07.** 下列关于流水段寄存器的叙述中，正确的是( )。

A. 指令译码得到的控制信号需通过流水段寄存器传递到下一个流水段 B. 每个流水段之间的流水段寄存器位数一定相同

C. 每个流水段之间的流水段寄存器存放的信息一定相同

D. 用户程序可以通过指令指定访问哪个流水段寄存器

**08.** 下列关于流水线数据通路的描述中，错误的是( )。

A. 每个流水段由执行指令子功能的功能部件和流水段寄存器组成

B. 控制信号仅作用在功能部件上，时钟信号仅作用在流水段寄存器上 C. 在没有阻塞的情况下，PC 的值在每个时钟周期都会改变

D. 取指令阶段和指令译码阶段不需要控制信号的控制

**09.** 下列关于结构冒险的叙述中，正确的是()。

I. 结构冒险是指同时有多条指令使用同一个资源

II. 避免结构冒险的基本做法是使每个指令在相同流水段中使用相同的部件 Ⅲ. 重复设置功能部件可以避免结构冒险

IV. 数据 Cache和指令 Cache 分离可解决两条指令同时分别取数据和取指令的冒险

A.I 、Ⅱ 、IV B.I 、Ⅱ 、Ⅲ C.I 、Ⅲ 、IV D.I 、I 、Ⅲ 和IV

**10.** 指令流水线中出现数据相关时流水线将受阻，()可解决数据相关问题。

A. 增加硬件资源 B. 采用旁路技术

C. 采用分支预测技术 D. 以上都可以

**11.** 下列关于数据冒险和转发技术的叙述中，正确的是()。

I.并非所有数据冒险都能通过转发技术解决

II. 五段流水线中load-use数据冒险会引起至少一个时钟周期的阻塞

Ⅲ.前面的分支指令和后面的ALU 运算指令之间肯定不会发生数据冒险

A.I 、Il B.I 、 I C.I 、II D.I 、I 、 Ⅲ

**12.** 下列关于数据冒险的叙述中，正确的是()。

I. 数据冒险是指后面指令用到的数据还未来得及由前面的指令产生

II. 在发生数据冒险的指令之间插入空操作指令能避免数据冒险

Ⅲ.采用转发(旁路)技术可以解决一部分数据冒险现象

IV. 通过编译器调整指令顺序可解决部分数据冒险

A.I 、Ⅱ 、IV B.I 、Ⅱ 、Ⅲ C.I 、Ⅲ 、IV D.I 、Ⅱ 、Ⅲ 和 IV

**13.** 下列指令序列中，指令 I1 和 I3 、I2 和 I3 之间发生数据相关。假定采用“取指、译码/ 取数、执行、访存、写回”五段流水线方式，那么在采用转发技术时，需要在指令 I3

第5章 中央处理器 261

之前加入( )条空操作指令才能使这段程序不发生数据冒险。

Il:add rl,r0,1 #(rl)←(rO)+1

I2:load r3,12(r2) #(r3)-M[(r2)+12]

I3:add r5,r3,rl #(r5)←(r3)+(rl)

A.3 B.2 C.0 D.1

**14.** 下面有关控制冒险的描述中，错误的是()。

I. 无条件转移指令不会发生控制冒险

II. 在分支指令加入若干空操作可以避免控制冒险

ⅢI. 采用转发(旁路)技术，可以解决部分控制冒险

IV. 流水段的数量与控制冒险引发的开销无关

A.I 、IV B.II C.I 、II D.I、Ⅲ、IV

**15.** 下列关于分支预测的叙述中，正确的是()。

I. 分支预测技术可用于处理控制冒险和数据冒险

II. 使用静态预测技术时，每次的预测结果是一样的

.动态预测技术通常比静态预测技术的预测成功率高

IV. 若预测错误，已被错误放入流水线执行的指令必须被舍弃

A.I 、Ⅱ 、Ⅲ B.I 、I 、IV C.Ⅱ 、Ⅲ 、IV D.I、Ⅱ、Ⅲ、IV **16.** 下列关于指令流水线和指令执行效率的叙述中，错误的是( )。

A. 加倍增加流水段个数不能成倍提高指令执行效率

B. 为了提高指令吞吐率，流水段个数应无限制地增加

C. 增加流水段个数，可以提高处理器的时钟频率

D. 随着流水段个数的增加，流水段之间缓存开销的比例增大

**17.** 设指令由取指、分析、执行三个子部件完成，并且每个子部件的时间均为 t, 若采用常 规标量单流水线处理机，连续执行8条指令，则该流水线的加速比为( )。

A.3 B.2 C.3.4 D.2.4

**18.** 下列关于超标量流水线的描述中，不正确的是( )。

A. 在一个时钟周期内一条流水线可执行一条以上的指令

B. 一条指令分为多段指令由不同电路单元完成

C. 超标量通过内置多条流水线来同时执行多个处理器，其实质是以空间换取时间

D. 超标量流水线仅仅是指运算操作并行

**19.** 关于流水线技术的说法中，错误的是( )。

**A.** 超标量技术需要配置多个功能部件和指令译码电路等

**B** . 与超标量技术和超流水线技术相比，超长指令字技术对优化编译器要求更高，而无 其他硬件要求

C. 在按序流动的流水线中，只可能出现 RAW 相关

**D.** **超流水线技术相当于将流水线再分段，从而提高每个周期内功能部件的使用次数**

**20.【2**009统考真题】某计算机的指令流水线由4个功能段组成，指令流经各功能段的时间 (忽略各功能段之间的缓存时间)分别为90ns、80ns、70ns和60ns, 则该计算机的CPU 周期至少是( )

A.90ns B.80ns C.70ns D.60ns

**21.【2**010统考真题】下列不会引起指令流水线阻塞的是( )。

262-2025年计算机组成原理考研复习指导

A. 数据旁路 B. 数据相关 C. 条件转移 D. 资源冲突

**22.【2** 011 统考真题】下列指令系统的特点中，有利于实现指令流水线的是()。

I. 指令格式规整且长度一致 II. 指令和数据按边界对齐存放

Ⅲ . 只 有LOAD/STORE 指令才能对操作数进行存储访问

A. 仅 I 、Ⅱ B. 仅Ⅱ、Ⅲ C. 仅1、Ⅱ D.I、H、Ⅲ

**23.【**2013统考真题】某CPU 主频为1.03GHz, 采用4级指令流水线，每个流水段的执行需 要1个时钟周期。假定 CPU 执行了100 条指令，在其执行过程中，没有发生任何流水 线阻塞，此时流水线的吞吐率为()。

A.0.25×10°条指令/秒 B.0.97×10°条指令/秒

C.1.0×10° 条指令/秒 D.1.03×10°条指令/秒

**24.【**2014统考真题】采用指令 Cache 与数据 Cache 分离的主要目的是()。

A. 降 低 Cache 的缺失损失 B. 提 高Cache 的命中率

C. 降 低CPU 平均访存时间 D. 减少指令流水线资源冲突

**25.**【2016统考真题】在无转发机制的五段基本流水线(取指、译码/读寄存器、运算、访存、 写回寄存器)中，下列指令序列存在数据冒险的指令对是( )。

|  |  |
| --- | --- |
| I1:add | R1,R2,R3;(R2)+(R3)→R1 |
| I2:add | R5,R2,R4;(R2)+(R4)→R5 |
| I3:add | R4,R5,R3;(R5)+(R3)→R4 |

I4:add R5,R2,R6;(R2)+(R6)→R5

A.I1 和 I2 B.I2 和 I3 C.I2 和 I4 D.I3 和 I4

**26.【**2017统考真题】下列关于超标量流水线特性的叙述中，正确的是()。 I. 能缩短流水线功能段的处理时间

II. 能在一个时钟周期内同时发射多条指令

ⅢI. 能结合动态调度技术提高指令执行并行性

A. 仅 Ⅱ B. 仅1、Ⅲ C. 仅Ⅱ、HI D.I 、Ⅱ 和 Ⅲ

**27.【2** 017统考真题】下列关于指令流水线数据通路的叙述中，错误的是( )。

A. 包含生成控制信号的控制部件

B. 包含算术逻辑运算部件(ALU)

C. 包含通用寄存器组和取指部件

D. 由组合逻辑电路和时序逻辑电路组合而成

**28.** 【2018统考真题】若某计算机最复杂指令的执行需要完成5个子功能，分别由功能部件 A~E 实现，各功能部件所需时间分别为80ps 、50ps 、50ps 、70ps 和50ps, 采用流水线 方式执行指令，流水段寄存器延时为20ps, 则 CPU 时钟周期至少为()。

A.60ps B.70ps C.80ps D.100ps

**29.** 【2019统考真题】在采用“取指、译码/取数、执行、访存、写回”5段流水线的处理器 中，执行如下指令序列，其中 s0 、s1 、s2 、s3 和 t2 表示寄存器编号。

11:add s2,sl,s0 //R[s2]←R[sl]+R[s0]

I2:load s3,0(t2) //R[s3]←M[R[t2]+0]

I3:add s2,s2,s3 //R[s2]←R[s2]+R[s3]

I4:store s2,0(t2) //M[R[t2]+0]←R[s2]

下列指令对中，不存在数据冒险的是()。

第5章 中央处理器 263

A.I1 和 I3 B.I2 和 I3 C.I2 和14 D.I3 和14

**30.【2** 020统考真题】下列给出的处理器类型中，理想情况下， CPI 为 1 的 是 ( ) 。

I. 单周期CPU II. 多周期CPU ⅢI. 基本流水线 CPU IV. 超标量流水线 CPU

A. 仅 I 、Ⅱ B. 仅 I 、Ⅲ C. 仅Ⅱ、IV D. 仅 Ⅲ 、IV

**31.** 【2023统考真题】在采用“取指、译码/取数、执行、访存、写回”5段流水线的 RISC 处理器中，执行如下指令序列(第一列为指令序号),其中 s0 、s1 、s2 、s3 和 t2 表示寄 存器编号。

Il add s2,sl,s0 //R[s2]-R[sl]+R[s0]

I2 load s3,0(s2) /IR[s3]—M[R[s2]+0]

I3 beq t2,s3,L1 //ifR[t2]=R[s3]jump to L1

I4 addi t2,t2,20 //R[[2]-R[t2]+20

15 L1:.

若采用转发(旁路)技术处理数据冒险，采用硬件阻塞方式处理控制冒险，则在指令I1~ 14的执行过程中，发生流水线阻塞的指令有( )。

A. 仅 I3 B. 仅 I2 、I4 C. 仅 I3 、I4 D. 仅 I2 、I3 、I4

**二、综合应用题**

**01.** 现有四级流水线，分别完成取指令、指令译码并取数、运算、回写四步操作，假设完成 各部操作的时间依次为100ns、100ns、80ns 和50ns。 试问：

1)流水线的操作周期应设计为多少?

2)若相邻两条指令如下，发生数据相关(假设在硬件上不采取措施),试分析第二条指 令要推迟多少时间进行才不会出错。

ADD

SUB

R1,R2,R3

R4,R1,R5

#R2+R3

#R1-R5

->R1

->R4

3)若在硬件设计上加以改进，至少需要推迟多少时间?

**02.** 假设指令流水线分为取指 (IF) 、 译 码 (ID) 、 执 行 (EX) 、 回 写(WB)4 个过程，共有

10条指令连续输入此流水线。

1)画出指令周期流程图。

2)画出非流水线时空图。

3)画出流水线时空图。

4)假设时钟周期为100ns, 求流水线的实际吞吐量(单位时间执行完毕的指令数)。

**03.**【2012统考真题】某16位计算机中，有符号整数用补码表示，数据 Cache 和指令 Cache

分离。下表给出了指令系统中的部分指令格式，其中 Rs 和 Rd 表示寄存器， mem 表示 存储单元地址， (x) 表示寄存器x 或存储单元x 的内容。

**表指令系统中部分指令格式**

|  |  |  |  |
| --- | --- | --- | --- |
| **名** **称** | **指令的汇编格式** | | **指令功能** |
| 加法指令 | ADD | Rs,Rd | (Rs)+(Rd)→Rd |
| 算术/逻辑左移 | SHL | Rd | 2\*(Rd)→Rd |
| 算术右移 | SHR | Rd | (Rd)/2→Rd |
| 取数指令 | 10AD | Rd.mem | (mem)→Rd |
| 存数指令 | STORE | Rs,mem | (Rs)→mem |

264-2025年计算机组成原理考研复习指导

该计算机采用5段流水方式执行指令，各流水段分别是取指(IF) 、 译码/读寄存器(ID) 、 执行/计算有效地址(EX)、 访问存储器(M) 和结果写回寄存器(WB), 流水线采用“按 序发射，按序完成”方式，未采用转发技术处理数据相关，且同一寄存器的读和写操作 不能在同一个时钟周期内进行。请回答下列问题：

1 ) 若int 型 变 量x 的值为-513,存放在寄存器R1 中，则执行 “SHR R1” 后 ，R1 中 的

内容是多少(用十六进制表示)?

2)若在某个时间段中，有连续的4 条指令进入流水线，在其执行过程中未发生任何阻 塞，则执行这4条指令所需的时钟周期数为多少?

3)若高级语言程序中某赋值语句为x=a+b,x、a 和 b 均 为int型变量，它们的存储单 元地址分别表示为[x] 、[a] 和[b] 。 该语句对应的指令序列及其在指令流中的执行过程 如下所示。



|  |  |  |
| --- | --- | --- |
| I1 | LOAD | R1,[a] |
| I2 | LOAD | R2, [b] |
| I3 | ADD | R1,R2 |
| I4 | STORE | R2,[x] |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **时钟**  指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | **12** | **13** | 14 |
| I₁ | IF | ID | EX | M | WB |  |  |  |  |  |  |  |  |  |
| I₂ |  | IF | ID | EX | M | WB |  |  |  |  |  |  |  |  |
| l₃ |  |  | IF |  |  |  | ID | EX | M | WB |  |  |  |  |
| L₄ |  |  |  |  |  |  | IF |  |  |  | ID | EX | M | WB |

则这4条指令执行过程中I3 的 ID 段 和 1 4 的IF 段被阻塞的原因各是什么?

4)若高级语言程序中某赋值语句为x=x\*2+a,x 和 a 均 为unsigned int类型的变量，它

们的存储单元地址分别表示为[x] 、[a], 则执行这条语句至少需要多少个时钟周期?要 求模仿上图画出这条语句对应的指令序列及其在流水线中的执行过程示意图。

**04.【** 2014统考真题】某程序中有循环代码段P:“for(int i=0;i<N;i++)sum+=A[i];”。假

设编译时变量sum 和 i 分别分配在寄存器R1 和 R2 中。常量N 在寄存器R6 中，数组 A 的首地址在寄存器 R3 中。程序段P 的起始地址为08048100H, 对应的汇编代码和机器 代码如下表所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编** **号** | **地** **址** | **机器代码** | **汇编代码** | **注** **释** |
| 1 | 08048100H | 00022080H | loop:sll R4,R2,2 | (R2)<<2→R4 |
| 2 | 08048104H | 00083020H | add R4,R4,R3 | (R4)+(R3)→R4 |
| 3 | 08048108H | 8C850000H | load R5,0(R4) | ((R4)+0)→R5 |
| 4 | 0804810CH | 00250820H | add R1,R1,R5 | (R1)+(R5)→R1 |
| 5 | 08048110H | 20420001H | add R2,R2,1 | (R2)+1→R2 |
| 6 | 08048114H | 1446FFFAH | bne R2,R6,loop | f(R2)!=(R6)goto loop |

执行上述代码的计算机M 采用32位定长指令字，其中分支指令 bne 采用如下格式：

31 26 25 21 20 1615 0

|  |  |  |  |
| --- | --- | --- | --- |
| OP | Rs | Rd | OFFSET |

OP 为 操 作 码 ；Rs 和 Rd 为寄存器编号； OFFSET 为偏移量，用补码表示。

第5章 中央处理器 265

请回答下列问题，并说明理由。

1)M 的存储器编址单位是什么?

2 ) 已 知sll 指令实现左移功能，数组 A 中每个元素占多少位?

3 ) 表 中 bne 指令的 OFFSET 字段的值是多少?已知 bne 指令采用相对寻址方式，当前 PC 内 容 为 bne 指令地址，通过分析表中指令地址和 bne 指令内容，推断 bne 指令的

转移目标地址计算公式。

4 ) 若M 采用如下“按序发射、按序完成”的5级指令流水线： IF ( 取 值 ) 、ID (译码及 取数)、EXE ( 执 行 ) 、MEM ( 访 存 ) 、WB (写回寄存器),且硬件不采取任何转发措 施，分支指令的执行均引起3个时钟周期的阻塞，则 P 中哪些指令的执行会由于数 据相关而发生流水线阻塞?哪条指令的执行会发生控制冒险?为什么指令1 的 执 行 不会因为与指令5的数据相关而发生阻塞?

**05.** 【2014统考真题】假设对于上题中的计算机 M 和程序 P 的机器代码， M 采用页式虚拟

存储管理； P 开始执行时， (R1)=(R2)=0,(R6)=1000, 其机器代码已调入主存但不在

Cache 中；数组 A 未调入主存，且所有数组元素在同一页，并存储在磁盘的同一个扇区。 请回答下列问题并说明理由。

1)P 执行结束时， R2 的内容是多少?

2)M 的 指 令Cache 和数据 Cache 分离。若指令 Cache 共 有 1 6 行 ，Cache 和主存交换的 块大小为32B, 则其数据区的容量是多少?若仅考虑程序段P 的执行，则指令Cache 的命中率为多少?

3)P 在执行过程中，哪条指令的执行可能发生溢出异常?哪条指令的执行可能产生缺页 异常?对于数组 A 的访问，需要读磁盘和TLB 至少各多少次?

**5.6.7** **答案与解析**

**一** **、单项选择题**

**01.D**

空间并行即资源重复，主要指多个功能部件共同执行同一任务的不同部分，典型的如多处理 机系统。时间并行即时间重叠，让多个功能部件在时间上相互错开，轮流重叠执行不同任务的相 同部分，因此流水CPU 利用的是时间并行性， A 错 误 。RISC 都采用流水线技术，以提高资源利 用率。但反过来并不成立，因为大部分CISC 同样采用了流水线技术， B 错误。流水 CPU 和多媒 体 CPU 无必然联系，多媒体CPU 是指能够处理多种媒体数据(如图像、音频、视频等)的CPU, 它通常具有特殊的指令集和功能部件，与流水CPU 的概念不同， C 错误。

**02.** A

吞吐能力是指单位时间内完成的指令数。 m 段流水线在第m 个时钟周期后，每个时钟周期都 可以完成一条指令；而m 个并行部件在m 个时钟周期后能完成全部的m 条指令，等价于平均每 个时钟周期完成一条指令。因此两者的吞吐能力等同。

**03.B**

单流水线处理机执行12条指令的时间为(3+(12-D)A=14At。

**04.** C

这个超标量流水线处理机可以发送4条指令，所以执行指令的时间为(3+(20-4)/4)N=7At。 05.D

每个功能段的时间设定为取指、分析和执行部分的最长时间2ns, 第 一 条指令在第5ns 时 执

266-2025年计算机组成原理考研复习指导

行完毕，其余的99条指令每隔2ns 执行完一条，所以100条指令全部执行完毕所需的时间为 (5+99×2)ns=203ns。

**06.D**

指令执行过程中的各个子功能都需要包含在某个流水段中，每条指令都会依次进入所有流水 段进行处理。不同指令的复杂度不同，所需的功能段不同，但为了保证指令流水线正常运行，流 水段个数以最复杂指令所用的功能段个数为准，流水段长度以最复杂的操作所花的时间为准。因 此，其他指令可以通过加入“空操作”功能段向最复杂的指令靠齐，

**07.A**

在某个时钟周期内，不同的流水段受不同指令的控制信号控制，执行不同指令的不同功能段， 在指令译码阶段由控制器产生指令各流水段的所有控制信号，分别在随后的各个时钟周期内被使 用，因此随后各流水段寄存器都要保存相应的控制信号，并通过流水段寄存器传递到下一个流水 段， A 正确。不同流水段寄存器存放的信息不同，因此流水段寄存器位数不一定相同，流水段寄 存器对用户程序是透明的，用户程序不能通过指令指定访问哪个流水段寄存器。

**08.B**

在流水线数据通路中，时钟信号不仅作用在流水段寄存器上，还要作用在 PC 、各类寄存器、 存储器等状态元件上。每条指令的取指令阶段和指令译码阶段的功能都相同，是公共流水段，且 控制信号是指令译码之后才产生的，因此这两个阶段不需要控制信号。

**09.D**

解决结构冒险的策略有两个方面(袁春风老师所撰教材中的结论):①一个部件每条指令只 能使用一次，且只能在特定阶段使用，可以避免一部分结构冒险；②通过设置多个独立的部件来 避免硬件资源冲突，例如可将寄存器的读口和写口分开，以及将指令Cache 和数据 Cache 分离等。 因此，四个说法都正确。

**10.B**

处理数据相关问题有两种方法： 一种是暂停相关指令的执行，即暂停流水线，直到能够正确 读出寄存器操作数为止；另一种是采用专门的数据通路，直接把结果送到 ALU 的输入端，这种 方法称为旁路技术。

**11.A**

部分数据冒险可以通过转发技术解决，但有些数据冒险不行，例如load-use 类型的数据冒险 (当下一条指令需要用到本条指令的访存结果时)。load-use 类型的数据冒险会引起一个或多个时 钟周期的阻塞，需要添加空操作指令解决。若 ALU 运算指令的某个操作数是分支指令转移后的执行 结果，就会发生数据冒险。例如，分支指令 “sltrl,r2,r3”, 其含义是如果(r2)<(r3), 则 r1=1; 否则 rl=0 。如果紧挨着一条ALU 运算指令要用到rl 的值，就会发生数据冒险。

**12.D**

插入空操作指令，使相关指令延迟执行，可以避免数据冒险。采用转发技术，将数据通路中 生成的中间数据直接转发到ALU 的输入端，可以解决部分数据冒险，但不能解决 load-use 类型的 数据冒险。通过编译器调整相关指令的顺序，也可以解决部分数据冒险。

13.D

转发技术可以解决部分数据冒险，但不能解决load-use 类型的数据冒险。分析上述指令序列， 指令 Il 在 EX 段结束时已得到rl 的新值，采用转发技术后，指令I3 在 ALU 中用到的rl 值可以 直接从EX/MEM 流水段寄存器中取，可以解决指令I1 和 I3 的数据冒险。指令12和13是load-use 类型的数据冒险， load 指令只在 MEM 段结束时才能取到主存中的数据，然后送 MEM/WB 流水

第5章 中央处理器一267

段寄存器，在WB 段的前半个周期才能将新值写入r3, 但随后的add 指 令 在EX 阶段就要取r3 的 值，因此会发生数据冒险。需要在 add 指令之前插入一条空操作指令，这样在 add 指令的 EX 段 就可从 MEM /WB 流水段寄存器中取出 load 指令的最新结果。

14.D

直接转移指令的转移目标地址在执行阶段才确定，会发生控制冒险， I 错误。插入空操作可 使条件转移指令的结果在取下一条有效指令之前确定，从而避免控制冒险，Ⅱ正确。采用转发技 术，可以解决的是数据相关，Ⅲ错误。流水段的数量越多，意味着在转移结果确定之前，可能取 出更多的错误指令，从而需要更多的时间和资源来处理这些错误指令， V 错 误 。

**15.C**

分支预测技术用于处理控制冒险。静态预测技术假定分支总是不发生或者总是发生，每次预 测结果是一样的。动态预测技术根据之前条件跳转的比较结果来预测，根据局部性原理，其预测 成功率通常比静态预测技术高。预测错误时，已被错误放入流水线执行的指令必须被舍弃。

**16.B**

适当增加流水段的个数，会使得每个流水段内的操作更简单，流水段的延迟更小，缩短了时 钟周期，从而可以提高时钟频率。但是，流水段之间的流水段寄存器也随之增多，增加了流水段 之间的额外缓存开销，因此加倍增加流水段个数不能成倍提高指令执行效率，且流水段个数也不 能无限制地增加。此外，随着流水段个数的增加，也将导致流水段的控制逻辑更复杂。

17.D

采用流水线时，第一条指令完成的时间是3t, 以后每经过t 都有一条指令完成，因此共需要 的时间为31+(8 - 1)t=10r; 而不采用流水线时，完成8条指令总共需要的时间为8×31=241,所 以流水线的加速比=24r÷10t=2.4。

18.D

超标量流水线不仅指运算操作并行，还包括取指、译码、访存、写回等其他操作，超标量技 术使 CPU 在同一 时间内执行多条指令，从而发挥更大的效率， D 错误。

**19.B**

要实现超标量技术，要求 CPU 中配置多个功能部件和指令译码电路，以及多个寄存器和总 线，以便实现同时执行多个操作， A 正确。超长指令字技术不仅对优化编译器要求更高，还需要 更多的硬件资源，如寄存器、功能部件、指令译码电路等，B 错误。流水线按序流动，肯定不会 出现先读后写 (WAR) 相关和写后写 (WAW) 相关；只可能出现没等到上一条指令写入而当前 指令就读寄存器的错误， C 正确。由超流水线技术的定义可知D 正 确 。

**20.** A

时钟周期应以各功能段的最长执行时间为准，否则用时长的流水段将不能正确完成。

21.A

采用流水线方式，相邻或相近的两条指令可能会因为存在某种关联，后一条指令不能按照原 指定的时钟周期运行，从而使流水线断流。有三种相关可能引起指令流水线阻塞：①结构相关， 也称资源相关； ②数据相关；③控制相关，主要由转移指令引起。

数据旁路技术的主要思想是，不必等某条指令的执行结果送回到寄存器，再从寄存器中取出 该结果，而是直接将执行结果送到其他指令所需的地方，这样可以使流水线不发生停顿。

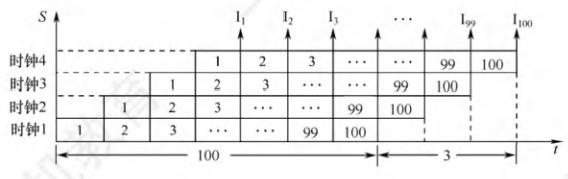
**22.D**

指令长度一致、按边界对齐存放、仅 LOAD/STORE 指令访存，这些都是 RISC 的特征，它 们使取指令、取操作数的操作简化且时间长度固定，能够有效地简化流水线的复杂度。

268-2025年计算机组成原理考研复习指导

**23.C**

采用4级流水执行100条指令，在执行过程中共使用4+(100- 1)=103个时钟周期，如下图 所示。CPU 的主频是1.03GHz, 即每秒有1.03G 个时钟周期。流水线的吞吐率为1.03G×100/103 = 1.0×10°条指令/秒



**24.D**

把指令Cache 与数据 Cache 分离后，取指和取数分别到不同的Cache 中寻找，则指令流水线 中取指部分和取数部分就可以很好地避免冲突，即减少了指令流水线的冲突。

**25.B**

数据冒险即数据相关，指在一个程序中存在必须等前一条指令执行完才能执行后一条指令的 情况，此时这两条指令即为数据相关。当多条指令重叠处理时就会发生冲突。首先这两条指令发 生写后读相关，且两条指令在流水线中的执行情况(发生数据冒险)如下表所示。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时钟  指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 12 | 取指 | 译码/读寄存器 | 运算 | 访存 | 写回 |  |  |
| I3 |  | 取指 | 译码/读寄存器 | 运算 | 访存 | 写回 |  |

指令12在时钟5时将结果写入寄存器R5, 但 指 令I3 在 时 钟 3 时 读R5 。 本来指令12应先写 入 R5, 指 令I3 后 读R5, 结果变成指令I3 先 读R5, 指令12后写入R5, 因而发生数据冲突。

26.C

超标量是指在CPU 中有一条以上的流水线，并且每个时钟周期内可以完成一条以上的指令， 其实质是以空间换时间。 I 错误，它不影响流水线功能段的处理时间；Ⅱ、Ⅲ正确。

27.A

数据在功能部件之间传送的路径称为数据通路，包括数据通路上流经的部件，如程序计数器、 ALU 、 通用寄存器、状态寄存器、异常和中断处理逻辑等。数据通路由控制部件控制，控制部件 根据每条指令功能的不同生成对数据通路的控制信号。因此，不包括控制部件。

**28.** D

指令流水线中每个流水段的时间单位为一个时钟周期，题中指令流水线的指令需要用到 A~ E 五个部件，所以每个流水段时间应取最大部件时间80ps, 此外还有流水段寄存器延时20ps, 因 此 CPU 时钟周期至少是100ps:

**29.C**

画出这四条指令在流水线中执行的过程如下图所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| add s2,sl,s0 | 取指 | 译码/取数 | 执行 | 访存 | 写回 |  |  |  |  |  |  |  |  |  |
| load s3,0(t2) |  | 取指 | 译码/取数 | 执行 | 访存 | 写回 |  |  |  |  |  |  |  |  |
| add s2,s2,s3 |  |  | 取指 |  |  |  | 译码/取数 | 执行 | 访存 | 写回 |  |  |  |  |
| store s2,0(t2) |  |  |  |  |  |  | 取指 |  |  |  | 译码/取数 | 执行 | 访存 | 写回 |

第5章 中央处理器 269

数据冒险即数据相关，指在程序中存在必须等前一条指令执行完才能执行后一条指令的情 况，此时这两条指令即为数据相关。其中I1 和 I3、12 和 I3 、I3 和 I4 均发生了写后读相关，因 此必须等相关的前一条指令执行完才能执行后一条指令。只有12和I4 不存在数据冒险。

**30.B**

CPI 表示执行指令所需的时钟周期数。对于一个程序或一台机器来说，其CPI 是指执行该程 序或机器指令集中的所有指令所需的平均时钟周期数。对于单周期 CPU, 令指令周期=时钟周 期 ，CPI=1,I 正确。对于多周期CPU,CPU 的执行过程分成几个阶段，每个阶段用一个时钟完 成，每种指令所用的时钟数可以不同，CPI>1,Ⅱ 错 误。对于基本流水线CPU, 让每个时钟周期

流出一条指令， CPI=1,Ⅲ 正确。超标量流水线CPU 在每个时钟周期内并发执行多条独立的指

令，每个时钟周期流出多条指令， CPI<1,IV 错 误。

31.C

I2 和 Il 之间存在数据冒险， I1 在 WB 段才将新值写回寄存器 R[s2], 但12的 Ex 段就要读 R[s2]以计算访存的有效地址，I1 在 EX 段结束时就已生成 R[s2]的新值，被存放在 Ex/Mem 流 水段寄存器中，采用转发技术后，可直接从该寄存器中取出数据送到ALU 的输入端，这样 I2 执行时ALU 用的是R[s2] 的新值，解决了I2 和 I1 之间的数据冒险。I3 和 I2 之间存在数据冒险， 属于 load-use 数据冒险，用转发电路无法解决I3 和12的数据相关问题，原因在于12的功能是 从内存中取数，只有在 Mem 段结束时才能从主存中得到R[s3] 的新值，但I3 的 EX 段就要用到 R[s3], 因此无法用转发技术解决， I3 仍需阻塞一个时钟周期，等到12的 Mem 段结束后，从 12的Mem/Wb 流水段寄存器中取到R[s3] 的新值。1 4 和I3 之间存在控制冒险，beq 指令在 Ex 段设置条件码，在 Mem 段控制是否将转移地址送到PC, 这之后才能开始根据 PC 内容取指 令，因此14需要进行硬件阻塞。综上所述，I3 和14的执行需要被阻塞，指令的执行过程如 下表所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Il | IF | ID | EX | Mem | WB |  |  |  |  |  |  |  |
| 12 |  | IF | ID | EX | Mem | WB |  |  |  |  |  |  |
| I3 |  |  | IF | ID | 阻塞 | EX | Mem | WB |  |  |  |  |
| 14/15 |  |  |  | 阻塞 | 阻塞 | 阻塞 | 阻塞 | IF | ID | EX | Mem | WB |

**二、综合应用题**

**01.【**解答】

1)流水线操作的时钟周期T 应按四步操作中的最长时间来考虑，所以T=100ns。 2)分析如下：

首先该两条指令发生写后读相关，且两条指令在流水线中的执行情况如下表所示。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时钟  指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ADD | 取指 | 指令译码并取数 | 运算 | 写回 |  |  |  |
| SUB |  | 取指 | 指令译码并取数 | 运算 | 写回 |  |  |

ADD 指令在时钟4时将结果写入寄存器堆(R1), 但 SUB 指令在时钟3时读寄存器堆(R1) 。 本来 ADD 指令应先写入R1,SUB 指令后读 R1, 结果变成 SUB 指令先读 R1,ADD 指 令后写入R1, 因而发生数据冲突。若硬件上不采取措施，第二条指令SUB 至少应推迟两 个时钟周期(2×100ns), 即 SUB 指令中的指令译码并取数周期应在 ADD 指令的写回周 期之后才能保证不会出错，如下表所示。

270-2025年计算机组成原理考研复习指导

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时钟  指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ADD | 取指 | 指令译码并取数 | 运算 | 写回 |  |  |  |
| SUB |  | 取指 |  |  | 指令译码并取数 | 运算 | 写回 |

3)若在硬件上加以改进，可以不用任何延迟。因为在ADD 指令中，运算周期已得到结果。 可以通过数据旁路技术得到运算结果后，就将结果送入ALU 的一端，而不需要等到写回 周期完成。流水线中的执行情况如下表所示。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时钟  指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ADD | 取指 | 指令译码并 取数 | 运算(并采用数据旁路技  术写入寄存器R1) | 写回 |  | 取指 |  |
| SUB |  | 取指 | 指令译码并取数 | 运算 | 写回 |  |  |

02.【解答】

1)因为指令周期包括IF、ID、EX、WB 四个子过程，因此其指令周期流程图如下图所示。

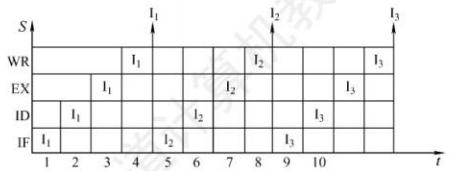
WB

IF

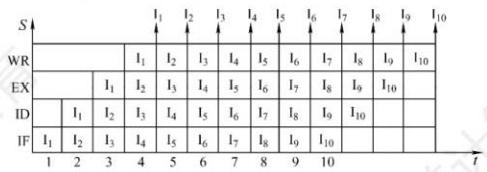
EX

ID

2)假设一个时间单位为一个时钟周期，则每隔4个周期才有一个输出结果。非流水线的时 空图如下图所示。



3)第一条指令出结果需要4条指令周期。流水线满载时，以后每个时钟周期都可输出一个 结果，即执行完一条指令，如下图所示。



4)由上图可知，在13个时钟周期结束时， CPU执行完10条指令，因此实际吞吐率 (T) 为



03. 【解答】

1)x 的机器码为[x]=1111110111111111B, 即指令执行前(R1)=FDFFH, 右移1位后为

1111111011111111B, 即指令执行后(R1)=FEFFH。

2)每个时钟周期只能有一条指令进入流水线，从第5个时钟周期开始，每个时钟周期都会

有一条指令执行完毕，因此至少需要4+(5-1)=8个时钟周期。

3)I₃ 的 ID 段被阻塞的原因：因为I₃ 与I₁ 和 I₂ 都存在数据相关，需等到I₁ 和I₂ 将结果写回寄

存器后， I₃ 才能读寄存器内容，所以I₃ 的 ID 段被阻塞。I₄ 的 IF 段被阻塞的原因：因为I₄ 的前一条指令I₃ 在 ID 段被阻塞，所以I₄ 的 IF 段被阻塞。若I₄ 的 IF 段不被阻塞，则会覆 盖指令寄存器的内容，导致I₃ 段的译码结果出错。

注意：要求“按序发射，按序完成”,因此，第二问中下一条指令的 IF 可以和上一条指 令的ID 并行，以免因上一条指令发生冲突而导致下一条指令先执行完。

4 ) 因 2 \*x 操作有左移和加法两种实现方法，因此x=x\*2+a 对应的指令序列为

R1,[x]

LOAD

LOAD

SHL

ADD

STORE

I1

I2

I3

I4

I5

R1,R1

//或者

ADD

R2,[a]

R1

R1,R2

R2,[x]

这5条指令在流水线中执行过程如下图所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **时间单元** | | | | | | | | | | | | | | | | |
| 指令 | 1 | 2 | 3 | 4 | 5 | 6 | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** |
| l1 | IF | ID | EX | M | WB |  |  |  |  |  |  |  |  |  |  |  |  |
| 12 |  | IF | ID | EX | M | WB |  |  |  |  |  |  |  |  |  |  |  |
| 13 |  |  | IF |  |  | ID | EX | M | WB |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  | IF |  |  |  | ID | EX | M | WB |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  | IF |  |  |  | ID | EX | M | WB |

因此执行x=x\*2+a 语句最少需要17个时钟周期。

**04.【**解答】

该题为计算机组成原理的综合题型，涉及指令系统、存储管理和 CPU 三部分内容，特别是 五段式流水线应引起考生的高度重视。整个指令执行过程中各流水段的时间是相同的，由统一的 时钟控制。各流水段在5.6.1节中介绍过，这里讨论流水段发生阻塞的情况：

① 若本条指令的源寄存器是上一条指令的目的寄存器，如果不采取任何措施，本条指令取 到的将是寄存器的旧值，这就是 RAW 数据冒险。这时有3个时钟周期的阻塞，使本指令 ID 应在上一条指令WB 后。本题中， I1 在 WB 段结束才将结果写回 R4, 而 1 2 在ID 段 就需要取 R4 的值，如下图所示。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| I₁ | IF | ID | EX | M | WB |  |  |  |  |
| l₂ |  | IF |  |  |  | ID | EX | M | WB |

第三条指令阻塞了3个时钟周期。

② 跳转指令 (JMP)(bra): 由于流水线默认直接取下一条指令，若指令为JMP 或 JC (条件

转移),在没有分支预测的情况下，默认有3个时钟周期的阻塞使下一条指令 IF 在分支 指令M 后(分支指令在M 段才会确定是否更新 PC) 。本 题 中 ，I6 是分支指令，其后一条 指令被阻塞3个时钟周期，若跳移条件成立，则转到I1 执行，反之顺序执行17。本题最 后一问是分析为什么I1 的执行不会因为与15的数据相关而阻塞，可以从两个角度来分析： 第一，16与I5 存在数据冒险，从而I6 的 ID 段被阻塞3个时钟周期，不论16是不是分支 指 令 ，I1 的 IF 段都会被阻塞，从而ID 段也会相应地推迟。第二，因为16是分支指令， 后续指令会阻塞3个时钟周期，从而也能消除Il 和15的数据冒险。

272-2025年计算机组成原理考研复习指导

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| I₅ | IF | ID | EX | M | WB |  |  |  |  |  |  |  |  |
| l₆ |  | IF |  |  |  | ID | EX | M | WB |  |  |  |  |
| I/l₇ |  |  |  |  |  | 阻塞 | 阻塞 | 阻塞 | IF | ID | EX | M | WB |

在了解上面的基础知识后，我们再看这道大题。

1)已知计算机 M 采用32位定长指令字，即 一 条指令占4B, 观察表中各指令的地址可知， 每条指令的地址差为4个地址单位，即4个地址单位代表4B, 一个地址单位就代表了1B, 所以该计算机是按字节编址的。

2)在二进制中某数左移两位相当于乘以4,由该条件可知，数组间的数据间隔为4个地址单

位，而计算机按字节编址，所以数组A 中的每个元素占4B 。

3 ) 由 表 可 知 ，bne 指令的机器代码为1446FFFAH, 根据题目给出的指令格式，后2B 的内容 为OFFSET 字段，所以该指令的OFFSET 字 段 为FFFAH, 用补码表示，值为-6。系统执 行 到bne 指 令 时 ，PC 自 动 加 4 ,PC 的内容为08048118H, 而跳转的目标是08048100H,

两者相差了18H, 即 2 4 个单位的地址间隔，所以偏移地址的一位即是真实跳转地址的 -24/-6=4位。可知bne 指令的转移目标地址计算公式为(PC)+4+OFFSET×4。

4)由于数据相关而发生阻塞的指令为第2、3、4、6条，因为第2、3、4、6条指令都与各

自前一条指令发生数据相关。第6条指令会发生控制冒险 。

当前循环的第5条指令与下次循环的第1 条指令虽然有数据相关，但由于第6条指令后 有3个时钟周期的阻塞，因而消除了该数据相关(或者解释如下：第6条指令因为与第5 条指令存在数据冒险，导致后续I1 的执行也相应地推迟，因此消除了该数据冒险)。

**05.【** 解答】

该题继承了上题中的相关信息，统考中首次引入此种设置，具体考查程序的运行结果、Cache 的大小和命中率的计算，以及磁盘和 TLB 的相关计算，是一道比较综合的题型。2015年同样出 现了23分大题的设定，希望读者对其足够重视。

1)R2 中装的是i 的值，循环条件是i<N(1000), 即 当i 自增到不满足这个条件时跳出循环， 程序结束，所以此时 R2 的值为1000。

2)Cache 共有16行，每块32字节，所以 Cache 数据区的容量为16×32B=512B。

P 共有6条指令，占24B, 小于主存块大小32B, 其起始地址为08048100H, 对应 一块 的开始位置，由此可知所有指令都在一个主存块内。读取第一条指令时会发生 Cache 缺 失，因此将P 所在的主存块调入Cache 的某一行，以后每次读取指令时，都能在指令Cache 中命中。因此在1000次循环中，只会发生1次指令访问缺失，所以指令Cache 的命中率 为(1000×6-1)-(1000×6)=99.98%。

3)指令4为加法指令，即对应sum+=A[i], 当 数 组 A 中元素的值过大时，会导致这条加 法指令发生溢出异常；而指令2、5虽然都是加法指令，但它们分别为数组地址的计算 指令和存储变量i 的寄存器进行自增的指令，而i 最大到达1000,所以它们都不会产生 溢出异常。

只有访存指令可能产生缺页异常，即指令3可能产生缺页异常。因为数组 A 在磁盘的 一 页上，而一开始数组并不在主存中，第一次访问数组时会导致访盘，把A 调入内存，而 以后数组 A 的元素都在内存中，不会导致访盘，所以该程序共访盘一次。

每访问一次内存数据，就查一次TLB, 共访问数组1000次，所以此时又访问1000次 TLB, 还要考虑到第一次访问数组A, 即 访 问A[0] 时，会多访问一次 TLB ( 第 一 次 访 问A[0] 时

第5章 中央处理器- 273

会先查一次TLB, 然后产生缺页，处理完缺页中断后，会重新访问A[0], 此时又查TLB), 所以访问TLB 的次数一共是1001次。



**5.7** **多处理器的基本概念**

▶ **多处理器的基本概念(2022)**

**5.7.1** **SISD 、SIMD 、MIMD 的基本概念**

基于指令流的数量和数据流的数量，将计算机体系结构分为 SISD、SIMD、MISD 和 MIMD

四类。常规的单处理器属于 SISD, 而常规的多处理器属于MIMD。

**1.** **单指令流单数据流** **(SISD) 结构**

SISD 是传统的串行计算机结构，这种计算机通常仅包含一个处理器和一个存储器，处理器 在一段时间内仅执行一条指令，按指令流规定的顺序串行执行指令流中的若干指令。为了提高速 度，有些 SISD 计算机采用流水线的方式，因此，SISD 处理器有时会设置多个功能部件，并且采 用多模块交叉方式组织存储器。本书前面介绍的内容多属于SISD 结构。

**2.** **单指令流多数据流** **(SIMD) 结构**

SIMD 是指一个指令流同时对多个数据流进行处理， 一般称为数据级并行技术。这种结构的 计算机通常由一个指令控制部件、多个处理单元组成。每个处理单元虽然执行的都是同一条指令， 但是每个单元都有自己的地址寄存器，这样每个单元就都有不同的数据地址，因此，不同处理单 元执行的同一条指令所处理的数据是不同的。 一个顺序应用程序被编译后，既可能按 SISD 组织 并运行于串行硬件上，又可能按SIMD 组织并运行于并行硬件上。

SIMD 在使用 for 循环处理数组时最有效，比如， 一条分别对16对数据进行运算的 SIMD 指 令如果在16个ALU 中同时运算，则只需要一次运算时间就能完成运算。SIMD 在使用case 或 switch 语句时效率最低，此时每个执行单元必须根据不同的数据执行不同的操作。

**3.** **多指令流单数据流** **(MISD) 结构**

MISD 是指同时执行多条指令，处理同一个数据，实际上不存在这样的计算机。

**4.** **多指令流多数据流** **(MIMD) 结构**

MIMD 是指同时执行多条指令分别处理多个不同的数据，MIMD 分为多计算机系统和多处理 器系统。多计算机系统中的每个计算机节点都具有各自的私有存储器，并且具有独立的主存地址 空间，不能通过存取指令来访问不同节点的私有存储器，而要通过消息传递进行数据传送，也称 消息传递 MIM D 。多处理器系统是共享存储多处理器 (SMP ) 系统的简称，它具有共享的单一地 址空间，通过存取指令来访问系统中的所有存储器，也称共享存储MIMD。

向量处理器是 SIMD 的变体，是一种实现了直接操作一维数组(向量) 指令集的CPU, 而 串 行处理器只能处理单一数据集。其基本理念是将从存储器中收集的一组数据按顺序放到一组向量 寄存器中，然后以流水化的方式对它们依次操作，最后将结果写回寄存器。向量处理器在特定工 作环境中极大地提升了性能，尤其是在数值模拟或者相似的领域中。

SIMD 和 MIMD 是两种并行计算模式，其中 SIMD 是一种数据级并行模式，而 MIMD 是一 种并行程度更高的线程级并行或线程级以上并行计算模式。

|  |  |
| --- | --- |
| 时钟 | CPU |
| i | 发射线程A的指令j、j+1 |
| i+1 | 发射线程B的指令k、k+1 |
| i+2 | 发射线程A的指令j+2、j+3 |
| i+3 | 发射线程B的指令k+2、k+3 |

274-2025年计算机组成原理考研复习指导

**5.7.2** **硬件多线程的基本概念**

在传统 CPU 中，线程的切换包含一系列开销，频繁地切换会极大影响系统的性能，为了减 少线程切换过程中的开销，便诞生了硬件多线程。在支持硬件多线程的CPU 中，必须为每个线程 提供单独的通用寄存器组、单独的程序计数器等，线程的切换只需激活选中的寄存器，从而省略 了与存储器数据交换的环节，大大减少了线程切换的开销。

硬件多线程有3种实现方式：细粒度多线程、粗粒度多线程和同时多线程 (SMT)。

**1.** **细粒度多线程**

多个线程之间轮流交叉执行指令，多个线程之间的指令是不相关的，可以乱序并行执行。在 这种方式下，处理器能在每个时钟周期切换线程。例如，在时钟周期i, 将线程A 中的多条指令 发射执行；在时钟周期i+1, 将线程B 中的多条指令发射执行。

**2.** **粗粒度多线程**

连续几个时钟周期都执行同一线程的指令序列，仅在当前线程出现了较大开销的阻塞时，才 切换线程，如 Cache 缺失。在这种方式下，当发生流水线阻塞时，必须清除被阻塞的流水线，新 线程的指令开始执行前需要重载流水线，因此，线程切换的开销比细粒度多线程更大。

上述两种多线程技术都实现了指令级并行，但线程级不并行。

**3.** **同时多线程**

同时多线程 (SMT) 是上述两种多线程技术的变体。它在实现指令级并行的同时，实现线程 级并行，也就是说，它在同一个时钟周期中，发射多个不同线程中的多条指令执行。

图5.21 分别是三种硬件多线程实现方式的调度示例。

|  |  |
| --- | --- |
| 时钟 | CPU |
| i | 发射线程A的指令j、j+1 |
| i+1 | 发射线程A的指令j+2、j+3,发现Cache miss |
| i+2 | 线程调度，从A切换到B |
| i+3 | 发射线程B的指令k、k+1 |
| i+4 | 发射线程B的指令k+2、k+3 |

(a)细粒度多线程示例 (b)粗粒度多线程示例

|  |  |
| --- | --- |
| 时钟 | CPU |
|  | 发射线程A的指令j、j+1,线程B的指令k、k+1 |
| i+1 | 发射线程A的指令j+2,线程B的指令k+2,线程C的指令m |
| i+2 | 发射线程A的指令j+3,线程C的指令m+1、m+2 |

(c)同时多线程示例

图5.21 三种硬件多线程方式的调度示例

Intel 处理器中的超线程 (Hyper-threading) 就是同时多线程SMT, 即在一个单处理器或单个

核中设置了两套线程状态部件，共享高速缓存和功能部件。

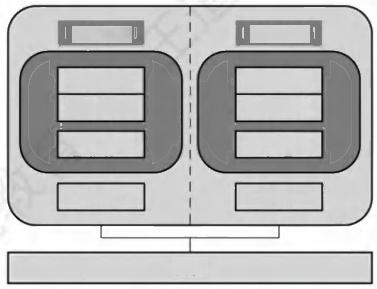
**5.7.3** **多核处理器的基本概念**

多核处理器是指将多个处理单元集成到单个 CPU 中，每个处理单元称为一个核 (core), 通

第5章 中央处理器- 275

常也称片上多处理器。每个核既可以有自己的 Cache, 又可以共享同 一个 Cache, 所有核通常共

享主存储器。图5.22是一个不共享Cache 的双核 CPU 结构。

CPU核 1

CPU 状态

中断逻辑

执行单元

Cache

CPU核 2

CPU 状态

中断逻辑

执行单元

Cache

内存

图5.22 不共享Cache 的双核 CPU 结构

在多核计算机系统中，如要充分发挥硬件的性能，必须采用多线程(或多进程)执行，使得 每个核在同一时刻都有线程在执行。与单核上的多线程不同，多核上的多个线程是在物理上并行 执行的，是真正意义上的并行执行，在同一时刻有多个线程在并行执行。而单核上的多线程是一 种多线程交错执行，实际上在同一时刻只有一个线程在执行。

下面通过一个例子来理解相关的概念。假设要将四颗圆石头滚到马路对面，滚动每颗石头平 均需花费1分钟。串行处理器会逐 一滚动每颗石头，花费4分钟。拥有两个核的多核处理器让两 个人滚石头，即每人滚两颗，花费2分钟。向量处理器找到一根长木板，放在四颗石头后面，推 动木板即可同时滚动四块石头，理论上只要力量够大，就只需要1分钟。多核处理器相当于拥有 多名工人，而向量处理器拥有一种方法，可以同时对多件事进行相同的操作。

**5.7.4** **共享内存多处理器的基本概念**

具有共享的单一物理地址空间的多处理器称为共享内存多处理器 (SMP )。 处理器通过存储 器中的共享变量互相通信，所有处理器都能通过存取指令访问存储器的任何位置。注意，即使这 些系统共享同一个物理地址空间，它们仍然可在自己的虚拟地址空间中单独地运行程序。

单一地址空间的多处理器有两种类型：

● 统 一 存储访问(UMA) 多处理器。每个处理器对所有存储单元的访问时间是大致相同的，

即访问时间与哪个处理器提出访存请求及访问哪个字无关。

● 非统 一 存储访问( NUMA) 多处理器。某些存储器的访存速度要比其他的快，具体取决于

哪个处理器提出访问请求及访问哪个字，这是由于主存被分割分配给了不同处理器。

早期的计算机，内存控制器没有整合进 CPU, 访存操作需要经过北桥芯片(集成了内存控 制器，并与内存相连),CPU 通过前端总线和北桥芯片相连，这就是统一存储访问(UMA) 构架。 随着 CPU 性能提升由提高主频转到增加CPU 数量(多核、多CPU), 越来越多的CPU 对前端总 线的争用使得前端总线成为瓶颈。为了消除 UMA 架构的瓶颈，非统一存储访问(NUMA) 构 架 诞生，内存控制器被集成到CPU 内部，每个CPU 都有独立的内存控制器。每 个CPU 都独立连接 到 一 部分内存，CPU 直连的这部分内存被称为本地内存。CPU 之间通过 QPI 总线相连。CPU 可 以通过QPI 总线访问其他CPU 的远程内存。与UMA 架构不同的是，在NUMA 架构下，内存的 访问出现了本地和远程的区别，访问本地内存明显要快于访问远程内存。

由于可能会出现多个处理器同时访问同一共享变量的情况，在操作共享变量时需要进行同

276-2025年计算机组成原理考研复习指导

步，否则， 一个处理器可能会在其他处理器尚未完成对共享变量的修改时，就开始使用该变量。 常用方法是通过对共享变量加锁的方式来控制对共享变量互斥访问。在一个时刻只能有一个处理 器获得锁，其他需要操作该共享变量的处理器必须等待，直到该处理器解锁该变量为止。

第3章讨论的一致性是指Cache 与主存之间的数据一致性。在 UMA构架的多处理器中，所 有CPU 共享同一内存空间，每个 CPU的 Cache 都是共享内存中的一部分副本，因此多核系统的 Cache一致性既包括 Cache 和内存之间的一致性，还包括各 CPU的 Cache 之间的一致性，也就是 说，对内存同一位置的数据，不同CPU 的 Cache 不应该有不一致的内容。

**5.7.5** **本节习题精选**

**单项选择题**

**01.** 按照 Flynn提出的计算机系统分类方法，多处理机属于( )。

A.SISD B.SIMD C.MISD D.MIMD

**02.** 从体系结构的角度来看，阵列处理机属于( )结构。

A.SISD B.SIMD C.MIMD D.MISD

**03.** 以下机器中，不属于SIMD 结构的是( )。

A. 并行处理机 B. 阵列处理机 C. 向量处理机 D. 标量流水线处理机

**04.** 具有一个控制部件和多个处理单元的计算机系统属于()结构。

A.SISD B.SIMD C.MISD D.MIMD

**05.** 下列关于超线程 (HT) 技术的描述中，正确的是( )。

A. 超线程技术可以让四核的 Intel Core i7处理器变成八核

B. 超线程是一项硬件技术，能使系统性能大幅提升，与操作系统和应用软件无关 C. 含有超线程技术的CPU 需要芯片组的支持才能发挥技术优势

D. 超线程模拟出的每个CPU 核都具有独立的资源，各自工作互不干扰 **06.** 双核 CPU 和超线程 CPU 的共同点是( )。

A. 都有两个内核 B. 都能同时执行两个运算

C. 都包含两个 CPU D. 都不会出现争抢资源的现象

07. 下列关于双核技术的叙述中，正确的是( )。

A. 双核是指主板上有两个 CPU

B. 双核是利用超线程技术实现的

C. 双核是指在 CPU上集成两个运算核心

D. 双核 CPU 是时间并行的并行计算

**08.** 下列有关多核 CPU和单核 CPU的描述中，错误的是()。

A. 双核的频率为2.4GHZ, 那么其中每个核心的频率也是2.4GHZ

B. 采用双核 CPU可以降低计算机系统的功耗和体积

C. 多核 CPU 共用一组内存，数据共享

D. 所有程序在多核 CPU上运行速度都快

**09.** 下列关于多核CPU 的描述中，正确的是()。

A. 各核心完全对称，拥有各自的 Cache

B. 任何程序都可以同时在多个核心上运行

C. 一颗 CPU 中集成了多个完整的执行内核，可同时进行多个运算

D. 只有使用了多核CPU 的计算机，才支持多任务操作系统

第5章 中央处理器 277

**10.** 下 列 关 于 多 处 理 器 的 说 法 中 ， 正 确 的 是 ( )。

I. 通常采用偶数路CPU, 如2路、4路、6路等

II.NUMA 构 架 比 UMA 构架的运算扩展性要强

.UMA 构架需要解决的重要问题是 Cache 一 致 性

A.I B.I 和 Ⅱ C.I 和 Ⅲ D.I 、Ⅱ 和 ⅢI

**11.** 下列关于多核处理器的说法中，不正确的是( )。

A. 多核处理器并不能使单线程程序的执行速度加快

B. 多核处理器在Flynn 分类法中属于 MIMD系统

C. 多核处理器实际上就是在 一 个 CPU 上集成了多个控制核心

D. 多核处理器通常比单核处理器的能耗更高

**12.**【2022统考真题】下列关于并行处理技术的叙述中，不正确的是( )。

A. 多核处理器属于MIMD 结构 B. 向量处理器属于 SIMD 结构

C. 硬件多线程技术只可用于多核处理器

D.SMP 中所有处理器共享单一物理地址空间

**5.7.6** **答** **案** **与** **解** **析**

**单项选择题**

**01.D**

Flynn 分类法将计算机体系结构分为 SISD 、SIMD 、MISD 和 MIMD 四类。常规的单处理器

属于 SISD, 常规的多处理机属于MIMD。

02.B

阵列处理机包含 一 个计算阵列，此阵列由多个处理单元组成。它使用单 一 的控制部件控制多

个处理单元，使每个处理单元对各自的数据进行同样的操作，属于 SIMD 结 构 。

03.D

A 、B 和 C 通常可理解为同一种概念，是 SIMD 结构。标量流水线处理机是 SISD 结构。

**04.B**

单指令流多数据流 (SIMD) 结构的计算机通常由 一 个指令控制部件、多个处理单元组成，

不同处理单元执行的同 一 条指令所处理的数据可以不同。

**05.** C

超线程技术是在 一 个 CPU 中，提供两套线程处理单元，让单个处理器实现线程级并行。虽 然采用超线程技术能够同时执行两个线程，但是当两个线程同时需要某个资源时，其中 一 个线程 必须暂时挂起，直到这些资源空闲后才能继续运行。因此，超线程的性能并不等于两个 CPU 的 性 能。而且，超线程技术的CPU 需 要 芯 片 组 、 操 作 系 统 ( 如Windows 98 不 支 持 超 线 程 技 术 ) 和 应 用软件的支持，才能发挥该项技术的优势。双核技术是指将两个一样的CPU 集成到 一 个封装内(或 者直接将两个 CPU 做成 一 个芯片),而超线程技术在CPU 内部仅复制必要的线程资源来让两个线 程同时运行，能并行执行两个线程，模拟实体双核。仅 C 正 确 。

**06.B**

超线程技术在CPU 内部仅复制必要的线程资源，共享CPU 的高速缓存和功能部件，让两个线 程可以并行执行，模拟双核CPU 。 当两个线程同时需要某个共享资源时，其中 一个线程必须暂时挂 起，直到这些资源空闲后才能继续运行。仅 B 正 确 。

278-2025年计算机组成原理考研复习指导

**07.C**

双核是指将两个CPU 核心集成到一个封装中，核心也称内核，是 CPU 最重要的组成部分， C 正确。主板上有两个 CPU 属于多处理器。超线程技术是模拟实体双核，不能算作真正意义上 的双核。时间并行是指流水线技术，空间并行则是指硬件资源的重复，空间并行导致了两类并 行机的产生，按 Flynn 分类法分为 SIMD 和 MIMD 。

**08.D**

多 核 CPU 的核心通常都是对称的，因此2.4GHz 双 核 CPU 中两个核的主频也是2.4GHz 。 早 期 CPU 性能提升主要靠提高主频，导致功耗增大，发热量大，而且当主频提高到一定程度后， CPU 性能的提升不再明显，后来转到增加 CPU 核心的方向，将2个核心集成到 一个芯片内，提 供等同双 CPU 的性能，这显然也降低了CPU 的体积。C 显然正确。在多核 CPU 上运行一个不支 持多线程的程序，显然不能发挥多核CPU 的 优 势 ，D 错 误。

09.C

多 核 CPU 的各核心既可以有独自的 Cache, 又可以共享同 一 个 Cache 。 只有支持多线程 的并行处理程序才能同时在多个核心上运行，发挥多核的优势。C 正 确。多任务系统也称多 道 程 序 系 统，可以运行在单核 CPU 上，宏观上并行，微观上串行。

10.D

SMP 也 称对称多处理器，一般采用偶数路 CPU,I 正 确。UMA 构架由于所有 CPU 共享相同 的内存，增加CPU 路数会加大访存冲突，通常2或4路的性能最好，而NUMA 理 论上支持无限 扩展，Ⅱ正确。UMA 构架中所有CPU 共享同一 内存空间，每个CPU 的 Cache 中都是共享内存中 的一部分副本，因此各 CPU 的 Cache 一致性是需要解决的重要问题，Ⅲ正确。

11.C

单线程程序只有一个执行流，因此多核处理器并不能使其执行速度加快。多核处理器属于 Flynn 分类法的 MIMD 系统。多核处理器是在一颗CPU 上集成了多个执行内核而非控制核心的处 理 器 ，C 错误。多核处理器可在一个时钟周期内处理多个并行任务，因此能耗通常更高。

**12.** C

MIMD 结构分为多计算机系统和多处理器系统。向量处理器是 SIMD 的变体，属于 SIMD 结构。硬件多线程技术在一个核中处理多个线程，可用于单核处理器， C 错误。共享内存多处理 器 (SMP) 具有共享的单一物理地址空间，所有核都可通过存取指令来访问同一片主存地址空间。



**5.8** **本章小结**

本章开头提出的问题的参考答案如下。

1)指令和数据均存放在内存中，计算机如何从时间和空间上区分它们是指令还是数据?

从时间上讲，取指令事件发生在“取指周期”,取数据事件发生在“执行周期”。从空间上讲， 从内存读出的指令流流向控制器(指令寄存器),从内存读出的数据流流向运算器(通用寄存器)。

2)什么是指令周期、机器周期和时钟周期?它们之间有何关系?

CPU 每取出并执行一条指令所需的全部时间称为指令周期；机器周期是在同步控制的机器 中，执行指令周期中 一 步相对完整的操作(指令步)所需的时间，通常安排机器周期长度=主 存周期；时钟周期是指计算机主时钟的周期时间，它是计算机运行时最基本的时序单位，对应完 成 一个微操作所需的时间，通常时钟周期=计算机主频的倒数。

第5章 中央处理器- 279

3)什么是微指令?它和第4章谈到的指令有什么关系?

控制部件通过控制线向执行部件发出各种控制命令，通常把这种控制命令称为微命令，而一 组实现一定操作功能的微命令的组合，构成一条微指令。许多条微指令组成的序列构成微程序， 微程序完成对指令的解释执行。指令，即指机器指令。每条指令可以完成一个独立的算术运算或 逻辑运算操作。在采用微程序控制器的CPU 中， 一条指令对应一个微程序， 一个微程序由许多微 指令构成， 一条微指令会发出很多不同的微命令。

4)什么是指令流水线?指令流水线相对于传统体系结构的优势是什么?

指令流水线是把指令分解为若干子过程，通过将每个子过程与其他子过程并行执行，来提高 计算机的吞吐率的技术。采用流水线技术只需增加少量硬件就能把计算机的运算速度提高几倍， 因此成为计算机中普遍使用的一种并行处理技术，通过在同一个时间段使用各功能部件，使得利 用率明显提高。



**5.9** **常见问题和易混淆知识点**

1. 流水线越多，并行度就越高。是否流水段越多，指令执行越快?

错误，原因如下：

1)流水段缓冲之间的额外开销增大。每个流水段有一些额外开销用于缓冲间传送数据、进 行各种准备和发送等功能，这些开销加长了一条指令的整个执行时间，当指令间逻辑上 相互依赖时，开销更大。

2)流水段间控制逻辑变多、变复杂。用于流水线优化和存储器(或寄存器)冲突处理的控 制逻辑将随流水段的增加而大增，这可能导致用于流水段之间控制的逻辑比段本身的控 制逻辑更复杂。

2. 读 后 写 (WAR) 相关和写后写 (WAW) 相关的概念

1 ) 读 后 写 (Write After Read,WAR)冲突。表示当前指令读出数据后，下一条指令才能写

该寄存器。否则，先写后读，读到的就是错误(新)数据。在下列指令中，寄存器R1 可 能存在这样的冲突，当指令12试图在指令II 读 R1 之前就写入该寄存器时，指令I1 就 错

误地读出该寄存器新的内容。

Il add R3,R1,R2 #(R1)+(R2)→R3

I2 sub R1,R4,R5 #(R4)-(R5)→R1

在读后写 (WAR) 冲突中，指令12的目的操作数是指令I1 的源操作数。

3 ) 写 后 写 (Write After Write,WAW) 相关。表示当前指令写入寄存器后，下一条指令才能 写该寄存器。否则，下一条指令在当前指令之前写，将使寄存器的值不是最新值。在下 列指令中，寄存器R1 可能存在这样的冲突，当指令12试图在指令I1 之前就写入R1 时 ， 就会错误地使由指令11写入的值成为该寄存器的内容。

Il add R1,R2,R3 #(R2)+(R3)→R1

I2 sub R1,R4,R5 #(R4)-(R5)→R1

在写后写 (WAW) 冲突中，指令I2 和 指 令II 的目的操作数是相同的。

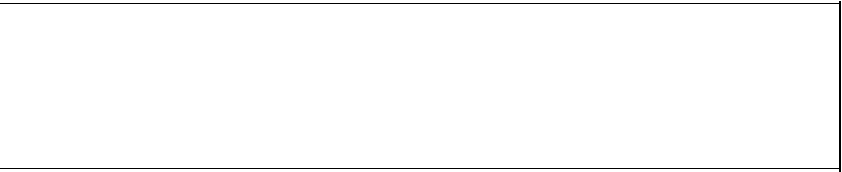


**注** **意**

在非按序执行的流水线中，因为允许后进入流水线的指令超过先进入流水线的指令而先流

出流水线，所以既可能发生RAW 相关，又可能发生 WAR 和 WAW 相关。





第 6 章

总 线

**【考纲内容】**

总线的基本概念

总线的组成及性能指标

总线事务和定时

**【复习提示】**

扫一扫



视频讲解

本章的知识点较少，通常以选择题的形式出现，特别是总线的特点、猝发传输方式、性能指 标、定时方式及常见的总线标准等。总线带宽的计算也可能结合其他章节出综合题。

在学习本章时，请读者思考以下问题：

1)引入总线结构有什么好处?

2)引入总线结构会导致什么问题?如何解决?

请读者在学习本章的过程中寻找答案，本章末尾会给出参考答案。



**6.1** **总线概述**

早期计算机的各部件之间是通过单独的连线互连的，这种方式称为分散连接。但是，随着I/O 设备的种类和数量越来越多，为了更好地解决 I/O 设备和主机之间连接的灵活性，计算机的结构 从分散连接发展为总线连接。为了进一步简化设计，又提出了各类总线标准。

**6.1.1** **总线基本概念**

**1.** **总线的定义**

总线是一组能为多个部件分时和共享的公共信息传送线路。 分时和共享是总线的两个特点。 分时是指同一时刻只允许有一个部件向总线发送信息，若系统中有多个部件，则它们只能分时地 向总线发送信息。共享是指总线上可以挂接多个部件，各个部件之间互相交换的信息都可通过这 组线路分时共享，多个部件可同时从总线上接收相同的信息。

**2.** **总线设备**

总线上所连接的设备，按其对总线有无控制功能可分为主设备和从设备两种。 主设备：指发出总线请求且获得总线控制权的设备。

从设备：指被主设备访问的设备，它只能响应从主设备发来的各种总线命令。

**3.** **总线特性**

总线特性是指机械特性(尺寸、形状)、电气特性(传输方向和有效的电平范围)、功能特性

(每根传输线的功能)和时间特性(信号和时序的关系)。

**6.1.2** **总线的分类**

**1.** **按功能层次分类**

**命** **题** **追** **踪** ▶ **总线相关的概念与特点(2016、2017)**

1) **片内总线。**芯片内部的总线，用于CPU 芯片内部各寄存器之间及寄存器与ALU 的连接。 2) **系统总线。**计算机系统内各功能部件 (CPU 、 主存、I/O 接口)之间相互连接的总线。按

系统总线传输信息内容的不同，又可分为3类：数据总线、地址总线和控制总线。

**命** **题** **追** **踪** ▶ **数据总线上传输的内容(2011)**

① 数据总线用来在各部件之间传输数据、指令和中断类型号等，它是双向传输总线，数据总 线的位数反映一次能传送的数据的位数。

② 地址总线用来指出数据总线上源数据或目的数据所在的主存单元或I/O 端口的地址，它是 单向传输总线，地址总线的位数反映最大的寻址空间。

③ 控制总线用来传输各种命令、反馈和定时信号，典型的控制信号包括时钟、复位、总线 请求/允许、中断请求/回答、存储器读/写、I/O 读、I/O 写、传输确认等。

注意区分数据通路和数据总线：各个功能部件通过数据总线连接形成的数据传输路径称为数 据通路。数据通路表示的是数据流经的路径，而数据总线是数据传输的媒介。

3)I/O **总线。**主要用于连接中低速的I/O 设备，通过I/O 接口与系统总线相连接，目的是将 低速设备与高速总线分离，以提升总线的系统性能，常见的有 USB 、PCI 总线。

4) **通信总线。**计算机系统之间或计算机系统与其他系统(如远程通信设备、测试设备)之 间传送信息的总线，通信总线也称外部总线。

**2.** **按时序控制方式分类**

1) **同步总线。**总线上连接的部件或设备通过统一的时钟进行同步，在规定的时钟节拍内进

行规定的总线操作，来完成部件或设备之间的信息传输。

2) **异步总线**。总线上连接的部件或设备没有统一的时钟，而以信号握手的方式来协调各部 件或设备之间的信息传输，总线操作时序不是固定的。

**3.** **按数据传输方式分类**

1) **串行总线。**只有一条双向传输或两条单向传输的数据线，数据按比特位串行顺序传输，

其效率低于并行总线。串行传输对数据线的要求不太高，因此适合长距离通信。

2) **并行总线**。有多条双向传输的数据线，可以实现多比特位的同时传输，其效率比串行总 线更高。缺点是各条数据线的传输特点可能存在一些差异，比如有的信息位可能会延迟， 并且数据线之间相互干扰还会造成传输错误，因此并行总线适合近距离通信。



**注** **意**

并行总线并不一定总比串行总线快，它们适合不同的场景。并行总线由于是多个数据 位同时传输，需要考虑数据的协同性，以及线路之间的相互干扰，导致工作频率无法持续 提高。而串行总线可通过不断提高工作频率来提高传输速度，使其速度最终超越并行总线 的速度。



系统总线

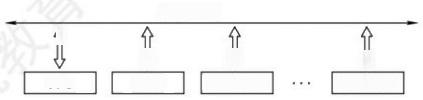
282-2025年计算机组成原理考研复习指导

**6.1.3** **系** **统** **总** **线** **的** **结** **构**

**1.** **单** **总** **线** **结** **构**

单总线结构将 CPU、主 存 、I/O 设备(通过I/O 接口)都挂在一组总线上，允许 I/O 设备之 间 、I/O 设 备 与 主 存 之 间 直 接 交 换 信 息 ， 如 图 6 . 1 所 示 。CPU 与 主 存 、CPU 与 外 设 之 间 都 可 以 通 过总线直接进行信息交换 ， 而无须经过中间设备的干预 。 需要注意的是 ， 单总线并不是指只有一

根信号线 ， 系统总线按传送信息的不同可细分为地址总线 、 数据总线和控制总线 。



介

人

CPU

少

设备接口

√ 主存

设备接口

图6. 1 单总线结构

优点：结构简单，成本低，易于接入新的设备。

缺点：带宽低、负载重，多个部件只能争用唯一的总线，且不支持并发传送操作。

**2.** **双** **总** **线** **结** **构**

双 总 线 结 构有 两 条 总 线 ： 一 条 是 主 存 总 线 ， 用 于 在 CPU 、 主 存 和 通 道 之 间 传 送 数 据 ； 另 一 条

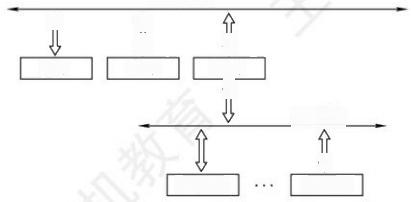
是 I/O 总 线 ， 用 于 在 多 个 外 部 设 备 与 通 道 之 间 传 送 数 据 ， 如 图 6 . 2 所 示 。

优 点 ： 将 低 速I/O 设 备 从 原 单 总 线 上 分 离 出 来 ， 实 现 了 存 储 器 总 线 和I/O 总 线 分 离 。

缺点 ： 需要增加通道等硬件设备 。

**3.** **三** **总** **线** **结** **构**

三 总 线 结 构是 在 计 算 机 系 统 各 部 件 之 间 采 用 3 条 各 自 独 立 的 总 线 来 构 成 信 息 通 路 ， 这 三 条 总 线 分 别 为 主 存 总 线 、I/O 总 线 和 直 接 内 存 访 问 (DMA) 总 线 ， 如 图 6 . 3 所 示 。



主存总线

个

上 上

主存 通道

个

VO总线

飞

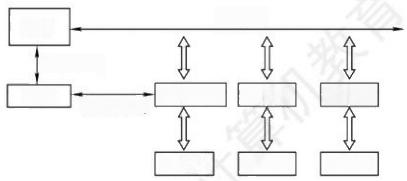
I/O接口

介

CPU

I/O接口

图6.2 双总线结构



CPU

主存总线

DMA 接口

DMA总线

磁盘机

I/O接口

I/O接口

VO总线

打印机

显示器

主存

图6 . 3 三 总 线 结 构

主 存 总 线 用 于 在 CPU 和 内 存 之 间 传 送 地 址 、 数 据 和 控 制 信 息 。I/O 总 线 用 于 在 CPU 和 各 类

外 设 之 间 通 信 。DMA 总 线 用 于 在 内 存 和 高 速 外 设 之 间 直 接 传 送 数 据 。

优 点 ： 提 高 了I/O 设 备 的 性 能 ， 使 其 更 快 地 响 应 命 令 ， 提 高 系 统 吞 吐 量 。

缺点 ： 任意时刻只能使用一种总线 ， 系统工作效率较低 。

**\*6.1.4** **常** **见** **的** **总** **线** **标** **准**

总线标准是国际上公布的互连各个模块的标准 ， 是把各种不同的模块组成计算机系统时必须

① 本节的内容2021年已从统考大纲中删除，仅供学习参考。

遵守的规范。典型的总线标准有 ISA 、EISA 、VESA 、PCI 、AGP 、PCI-Express 、USB 等。它们

的主要区别是总线宽度、带宽、时钟频率、寻址能力、是否支持突发传送等。

**命** **题** **追** **踪** ▶ **总线标准的英文缩写(2010)**

1)ISA,Industry Standard Architecture,工业标准体系结构。最早出现的系统总线，缺点是

传输速率低、CPU 占用率高、占用硬件中断资源。它属于系统总线、并行总线。

2)EISA,Extended Industry Standard Architecture, 扩 展 的ISA 。EISA 支持多个总线主控器 和突发传送， EISA 对 ISA 完全兼容。它属于系统总线、并行总线。

命匙道踪 区分设备总线和局部总线(2013)

3)VESA,Video Electronics Standards Association, 视频电子标准协会。是一个32位的局部 总线，是针对多媒体 PC 要求高速传送活动图像的大量数据而推出的。它属于局部总线、 并行总线。局部总线是在 ISA 总线和 CPU 总线之间增加的一级总线或管理层，这样就可 将一些高速外设(如图形卡、硬盘控制器、网卡等)从 ISA 总线上卸下，而通过局部总 线直接挂接到 CPU 总线上，使之能与高速 CPU 总线相匹配。

4)PCI,Peripheral Component Interconnect, 外部设备互连。是高性能的32位或64位总线， 目前常用的 PCI 适配器有显卡、声卡、网卡等。PCI 总线是一个与处理器时钟频率无关的

高速外围总线，支持即插即用。它属于局部总线、并行总线。

5)AGP,Accelerated Graphics Port, 加速图形接口。是一种视频接口标准，专用于连接主存

和图形卡，用于传输视频和三维图形数据。它属于局部总线、并行总线。

命题追踪 PCl-E 总线的特点(2017)

6)PCI-E,PCI-Express 。 高速串行双通道传输，其传输速率远高于 PCI 和 AGP, 且支持双

向传输模式，还可以支持全双工模式。它属于局部总线、串行总线。

7)RS-232C。 应用于串行二进制交换的数据终端设备 (DTE) 和数据通信设备 (DCE) 之间

的标准接口。它属于设备总线、串行总线。

▶ **USB 总线的特点(2012)**

8)USB,Universal Serial Bus, 通用串行总线。 一种连接外部设备的I/O 总线，具有即插即

用、热插拔等优点，有很强的连接能力。它属于设备总线、串行总线。

9)PCMCIA,Personal Computer Memory Card International Association。应用于笔记本电脑的

接口标准，通常用于扩展功能，具有即插即用功能。它属于设备总线、并行总线。

10)IDE,Integrated Drive Electronics,集成设备电路。更准确地称为ATA, 硬盘和光驱通过

IDE 接口与主板连接。它属于设备总线、并行总线。

11)SCSI,Small Computer System Interface, 小型计算机系统接口。 一种用于计算机和周边

设备之间(硬盘等)系统级接口的独立处理器标准。它属于设备总线、并行总线。

12)SATA,Serial Advanced Technology Attachment, 串行高级技术附件。 一种基于行业标准

的串行硬件驱动器接口。它属于设备总线、串行总线。

**6.1.5** **总线的性能指标**

1)总线时钟周期。即机器的时钟周期。计算机有一个统一的时钟，以控制整个计算机的各

个部件，总线也要受此时钟的控制。

2)总线时钟频率。即机器的时钟频率，它为时钟周期的倒数。

284-2025年计算机组成原理考研复习指导

3)总线传输周期。指一次总线操作所需的时间，包括申请阶段、寻址阶段、传输阶段和结 束阶段。总线传输周期通常由若干总线时钟周期构成。

4)总线工作频率。总线上各种操作的频率，为总线周期的倒数。实际上指1 秒内传送几次 数据。若总线周期=N 个时钟周期，则总线的工作频率=时钟频率/N; 此外，若一个时 钟周期可以传送K 次数据，则总线工作频率是总线时钟频率的K 倍。

5)总线宽度。总线宽度也称总线位宽，是总线上能够同时传输的数据位数，通常指数据总 线的根数，如32根称为 32位总线。

**命题追踪** ▶ **总线带宽的相关计算(2009、2013、2014、2019、2020)**

6)总线带宽。单位时间内总线上最多可传输数据的位数，通常用每秒传送信息的字节数来 衡量，单位可用字节/秒 (B/s) 表示。总线带宽=总线工作频率×(总线宽度/8)。

7)总线复用。总线复用是指一种信号线在不同的时间传输不同的信息。例如，有些总线没 有单独的地址线，地址信息通过数据线来传送，这种情况称为地址/数据线复用。因此可

以使用较少的线传输更多的信息，从而节省空间和成本。

8)信号线数。地址总线、数据总线和控制总线3种总线数的总和称为信号线数。

▶ **提高总线带宽的办法(2018)**

其中，总线最主要的性能指标为总线宽度、总线工作频率、总线带宽。总线带宽是指总线的 最大数据传输速率，它是衡量总线性能的重要指标。 三者的关系：总线带宽=总线宽度×总线工 作频率。例如，总线工作频率为22MHz, 总线宽度为16位，则总线带宽=22M×(16/8)=44MB/s。

**6.1.6 本节习题精选**

**一、单项选择题**

**01.** 挂接在总线上的多个部件()。

A. 只能分时向总线发送数据，并只能分时从总线接收数据

B. 只能分时向总线发送数据，但可同时从总线接收数据

C. 可同时向总线发送数据，并同时从总线接收数据

D. 可同时向总线发送数据，但只能分时从总线接收数据

**02.** 在总线上，同一时刻()。

A. 只能有一个主设备控制总线传输操作

B. 只能有一个从设备控制总线传输操作

C. 只能有一个主设备和一个从设备控制总线传输操作

D. 可以有多个主设备控制总线传输操作

**03.** 在计算机系统中，多个系统部件之间信息传送的公共通路称为总线，就其所传送的信息 的性质而言，下列()不是在公共通路上传送的信息。

A. 数据信息 B. 地址信息 C. 系统信息 D. 控制信息

**04.** 系统总线用来连接()。

A. 寄存器和运算器部件 B. 运算器和控制器部件

C.CPU、 主存和外设部件 D. 接口和外部设备

**05.** 计算机使用总线结构便于增减外设，同时()。

A. 减少信息传输量 B. 提高信息的传输速度

C. 减少信息传输线的条数 D. 提高信息传输的并行性

**06.** 间址寻址第一次访问内存所得到的信息经系统总线的( )传送到 CPU。

A. 数据总线 B. 地址总线 C. 控制总线 D. 总线控制器

**07.** 系统总线中地址线的功能是( )。

A. 选择主存单元地址 B. 选择进行信息传输的设备

C. 选择外存地址 D. 指定主存和I/O 设备接口电路的地址

**08.** 系统总线中控制线的主要功能是()。

A. 提供时序信号

B. 提供主存和I/O 模块的回答信号

C. 提供定时信号、操作命令和各种请求/回答信号等

D. 提供数据信息

**09.** 在单机系统中，三总线结构计算机的总线系统组成是()。

A. 片内总线、系统总线和通信总线 B. 数据总线、地址总线和控制总线

C.DMA 总线、主存总线和 I/O 总 线 D.ISA 总线、VESA 总线和 PCI 总线 **10.** 不同信号在同一条信号线上分时传输的方式称为( )。

A. 总线复用方式 B. 并串行传输方式

C. 并行传输方式 D. 串行传输方式

**11.** 主存通过( )来识别信息是地址还是数据。

A. 总线的类型 B. 存储器数据寄存器 (MDR)

C. 存储器地址寄存器 (MAR) D. 控制单元(CU)

**12.** 在32位总线系统中，若时钟频率为500MHz, 传送一个32位字需要5个时钟周期，则 该总线的数据传输速率是()。

A.200MB/s B.400MB/s C.600MB/s D.800MB/s

**13.** 传输一幅分辨率为640×480像素、颜色数量为65536的照片(采用无压缩方式),假设 有效数据的传输速率为56kb/s, 则大约需要的时间是()。

A.34.82s B.43.86s C.85.71s D.87.77s

**14.** 某总线有104根信号线，其中数据线(DB) 为32根，若总线工作频率为33MHz, 则其

理论最大传输速率为( )。

A.33MB/s B.64MB/s C.132MB/s D.164MB/s

**15.** 在一个16位的总线系统中，若时钟频率为100MHz, 总线周期为5个时钟周期传输一个 字，则总线带宽是()。

A.4MB/s B.40MB/s C.16MB/s D.64MB/s

**16.** 微机中控制总线上完整传输的信号有()。

I. 存储器和I/O 设备的地址码

II. 所有存储器和I/O 设备的时序信号与控制信号

III. 来 自I/O 设备和存储器的响应信号

A. 仅 I B.Ⅱ 和 Ⅲ C. 仅 IⅡ D.I 、Ⅱ 、 Ⅲ

**17.** 下列信号中，可在系统总线中的控制总线上传输的有( )。

I. 存储器和I/O 设备的地址信息 I. 存储器和I/O 设备的时序信号、控制信号

HI. 存储器和 I/O 设备的响应信号 IV. 存储器中存放的数据

A.I 和 IV B.Ⅱ 和 Ⅲ C.I 、Ⅱ 和 Ⅲ D.Ⅱ 、Ⅲ 和 IV

286-2025年计算机组成原理考研复习指导

**18.** 总线中，有些信息是单向传输的，有些信息是双向传输的，下列说法中正确的是()。 A. 数据信息是单向传输的，由内存或外设传送至CPU

B. 地址信息是单向传输的，由CPU 发送至内存或外设

C. 控制信息是双向传输的，由CPU 发送至内存或外设，也可反向

D. 状态信息是双向传输的，由CPU 发送至内存或外设，也可反向

**19.** 按连接部件不同，总线通常可以分为以下哪几种? ( )。

I. 系统总线 II. 片内总线 Ⅲ . 地址线 IV. 通信总线

A.I 、Ⅱ 和 Ⅲ B.I 、Ⅲ 和 IV C.I 、Ⅱ 和 IV D.I 、Ⅲ 和IV

**20.**【2009统考真题】假设某系统总线在一个总线周期中并行传输4字节信息， 一个总线周 期占用2个时钟周期，总线时钟频率为10MHz, 则总线带宽是( )。

A.10MB/s B.20MB/s C.40MB/s D.80MB/s **21.【**2010统考真题】\*下列选项中的英文缩写均为总线标准的是()。

A.PCI 、CRT 、USB 、EISA B.ISA 、CPI 、VESA 、EISA

C.ISA 、SCSI 、RAM 、MIPS D.ISA 、EISA 、PCI 、PCI-Express **22.** 【2011 统考真题】在系统总线的数据线上，不可能传输的是( )。

A. 指令 B. 操作数 C. 握手(应答)信号 D. 中断类型号

**23.【**2012统考真题】\*下列关于 USB 总线特性的描述中，错误的是()。

A. 可实现外设的即插即用和热拔插 B. 可通过级联方式连接多台外设

C. 是一种通信总线，连接不同外设 D. 同时可传输2位数据，数据传输速率高

**24.【2**013统考真题】\*下列选项中，用于设备和设备控制器之间互连的接口标准是( )。 A.PCI B.USB C.AGP D.PCI-Express

**25.【**2014统考真题】某同步总线采用数据线和地址线复用方式，其中地址/数据线有32根， 总线时钟频率为66MHz, 每个时钟周期传送两次数据(上升沿和下降沿各传送一次数 据),该总线的最大数据传输速率(总线带宽)是( )。

A.132MB/s B.264MB/s C.528MB/s D.1056MB/s

**26.【**2019统考真题】某计算机采用3通道存储器总线，配套的内存条型号为 DDR3-1333,

即内存条所接插的存储器总线的工作频率为1333MHz, 总线宽度为64位，则存储器总 线的总带宽大约是( )。

A.10.66GB/s B.32GB/s C.64GB/s D.96GB/s

**27.【**2020统考真题】QPI 总线是一种点对点全双工同步串行总线，总线上的设备可同时接 收和发送信息，每个方向可同时传输20位信息(16位数据+4位校验位),每个 QPI 数据包有80位信息，分2个时钟周期传送，每个时钟周期传递2次。因 此 ，QPI 总线 带宽为：每秒传送次数×2B×2。若 QPI 时钟频率为2.4GHz, 则总线带宽为( )。

A.4.8GB/s B.9.6GB/s C.19.2GB/s D.38.4GB/s

**二、** **综合应用题**

**01.** 某总线的时钟频率为66MHz, 在一个64位总线中，总线数据传输的周期是7个时钟周 期传输6个字的数据块。

1)总线的数据传输速率是多少?

2)若不改变数据块的大小，而将时钟频率减半，这时总线的数据传输速率是多少?

**02.** 某总线支持二级Cache块传输方式，若每块6个字，每个字长4字节，时钟频率为100MHz。

1)读操作时，第一个时钟周期接收地址，第二、三个为延时周期，另用4个周期传送 一个块。读操作的总线传输速率为多少?

2)写操作时，第一个时钟周期接收地址，第二个为延时周期，另用4个周期传送一个 块，写操作的总线传输速率是多少?

3)设在全部的传输中，70%的时间用于读，30%的时间用于写，该总线在本次传输中 的平均传输速率是多少?

**6.1.7** **答案与解析**

**一** **、单项选择是**

**01.B**

为了使总线上的数据不发生“冲突”,挂在总线上的多个设备只能分时地向总线发送数据， 即某个时刻只能有一个设备向总线传送数据，而从总线接收数据的设备可以有多个，因为接收数 据的设备不会对总线产生“干扰”。

**02.** A

只有主设备才能获得总线控制权，总线上的信息传输由主设备启动， 一条总线上可以有多个 设备作为主设备，但在同一时刻只能有一个主设备控制总线的传输操作。

**03.C**

总线包括数据线、地址线和控制线，传送的信息分别为数据信息、地址信息和控制信息。

**04.C**

系统总线用于连接计算机中的各个功能部件(如CPU 、 主存和 I/O 设备)。

**05.C**

计算机使用总线结构便于增减外设，同时减少信息传输线的条数。但相对于专线结构，其实 际上也降低了信息传输的并行性及信息的传输速度。

**06.A**

间址寻址首次访问内存所得到的信息是操作数的有效地址，该地址作为数据通过数据总线传送 至CPU, 地址总线是用于CPU 选择主存单元地址和I/O 端口地址的单向总线，不能回传。

地址总线由单向的多根信号线组成，可用于 CPU 向主存、外设传送地址信息；数据总线由 双向的多根信号线组成，CPU 既可以沿着这些线从主存或外设读入数据，又可以发送数据；控制 总线上传输控制信息，包括控制命令和反馈信号等。

07.D

地址总线上的代码用来指明 CPU 要访问的存储单元或 I/O 端口的地址。

**08.** C

系统总线中控制线的主要功能是提供定时信号、操作命令和各种请求/回答信号等。

**09.C**

选项 A 是总线按功能层次的划分，单机系统可不需要通信总线。选项 B 都属于系统总线。 选项 D 则是三种不同的总线标准。只有选项C 组成了三总线结构系统。

**10.** A

串行传输是指数据的传输在一条线路上按位进行，并行传输是指每个数据位有一条单独的传 输线，所有数据位同时传输。不同信号在同一条信号线上分时传输的方式，称为总线复用。

11.A

288-2025年计算机组成原理考研复习指导

地址和数据在不同的总线上传输，根据总线传输信息的内容进行区分，地址在地址总线上传 输，数据在数据总线上传输。

**12.B**

总线带宽=总线宽度×总线频率，本题中的总线宽度为32位，即4B, 总线频率为500MHz/5= 100MHz, 因此总线的数据传输速率为4B×(500MHz/5)=400MB/s。

**13.D**

65536=2¹⁶色，因此颜色深度为16位，占据的存储空间为640×480×16=4915200位。有效 传输时间=4915200÷(56×10³)s≈87.77s。

**14.C**

数据总线32根，因此每次传输32位，即4B 数据，总线工作频率为33MHz, 因此理论最大 传输速率为33×4=132MB/s。

**15.B**

时钟频率为100MHz, 因此时钟周期=1/100MHz =0.01μs, 总 线 周 期 = 5 个时钟周期= 5×0.01μs =0.05μs, 总线工作频率=1/0.05=20MHz, 因总线是16位的，即2B, 因此总线带宽= 20×(16/8)=40MB/s。

**16.B**

CPU 的控制总线提供的控制信号包括时序信号、I/O 设备和存储器的响应信号等。

**17.B**

控制总线主要用来传输计算机内各种控制信号，控制信号包括存储器和 I/O 设备的时序信号 和响应信号，Ⅱ、Ⅲ正确。存储器和I/O 设备的地址信息通过地址总线传输， I 错误。存储器中存 放的数据通过数据总线传输，IV 错误。

**18.B**

总线中，数据总线是双向传输的，数据信息既可由 CPU 传送至内存或外设，又可由内存、 外设传送至CPU,A 错误。地址总线是单向传输的，地址信息只能由CPU 发送至内存或外设， B 正确。控制信息和状态信息也是单向传输的，它们的传输方向正好相反，控制信息通过控制总线 由 CPU 发送至内存或外设，而状态信息则通过状态总线由内存或外设发送至CPU。

**19.C**

按连接部件不同，总线通常可分为片内总线、系统总线和通信总线，计算机系统的三总线结

构通常是指I/O 总线、主存总线和 DMA 总线。

20.B

总线带宽是指单位时间内总线上传输数据的位数，通常用每秒传送信息的字节数来衡量， 单位为B/s 。由题意可知，在1个总线周期(=2个时钟周期)内传输了4字节信息，时钟周期= 1/10MHz=0.1μs, 因此总线带宽为4B=(2×0.1μs)=4B÷(0.2×10~\*s)=20MB/s。

**21.D**

典型的总线标准有ISA 、EISA 、VESA 、PCI 、PCI-Express 、AGP 、USB 、RS-232C 等。选项 A 中 的CRT 是纯平显示器；选项B 中 的CPI 是每条指令的时钟周期数；选项C 中的 RAM 是半 导体随机存储器、MIPS 是每秒执行多少百万条指令数。

22.C

取指令时，指令便是在数据线上传输的。操作数显然在数据线上传输。中断类型号用于指出 中断向量(中断服务程序的入口地址)的地址，CPU 响应某一外部中断后，就会从数据总线上获

有 C 立 凸砂

289

取该中断源的中断类型号，然后据此计算对应中断向量在中断向量表(存放在内存中)的位置。 而握手(应答)信号属于总线定时的控制信号，应在控制总线上传输。

23.D

USB (通用串行总线)的特点有：①即插即用；②热插拔；③有很强的连接能力，采用菊花 链形式将众多外设连接起来；④有很好的可扩充性， 一个USB 控制器可扩充高达127个外部 USB 设备；⑤高速传输，速率可达480Mb/s 。对于 D,USB 是串行总线，不能同时传输2位数据。

**24.B**

USB 是一种连接外部设备的/O 总线标准，属于设备总线，是设备和设备控制器之间的接口。 而PCI、AGP、PCI-E 作为计算机系统的局部总线标准，通常用来连接主存、网卡、视频卡等。

25.C

数据线有32根，也就是一次可以传送32b/8=4B 的数据，66MHz 意味着有66M 个时钟周期， 而每个时钟周期传送两次数据，可知总线每秒传送的最大数据量为66M×2×4B=528MB, 所以总 线的最大数据传输速率为528MB/s。

**26.B**

由题目可知，计算机采用3通道存储器总线，存储器总线的工作频率为1333MHz, 即 1 秒 内 传送1333M 次数据，总线宽度为64位即单条总线工作一次可传输8字节 (Byte), 因此存储器总 线的总带宽为3×8×1333MB/s, 约为32GB/s。

27.C

每个时钟周期传送2次，所以每秒传送的次数=时钟频率×2=2.4G×2/s。 总线带宽=每秒传送次数×2B×2=2.4G×2×2B×2/s=19.2GB/s。

题中已给出总线带宽公式，降低了难度。公式中的“×2B” 是因为每次传输16位数据。

**二、** **综合应用题**

**01.【** 解答】

1)总线周期为7个时钟周期，总线频率为66/7MHz。

总线在一个完整的操作周期中传输了一个数据块，总线在一个周期内传输的数据量为

64bit/8×6=48B, 所以总线的宽度为48B, 传输速率为48B×66/7MHz=452.6MB/s。

2)时钟频率减半时的总线频率为(66/7)/2MHz, 因数据块大小不变，因此总线宽度仍为48B,

传输速率为48B×33/7MHz=226.3MB/s。

注意总线周期和时钟周期的联系与区别，总线周期通常由多个时钟周期组成。 02. 【解答】

1)读操作的时钟周期数： 1+2+4=7

对应的频率： 100MHz/7

总线宽度： 6×4B=24B

所以数据传输速率=总线宽度/读取时间=24×(100MHz/7)=343MB /s。 2)写操作的时钟周期数： 1+1+4=6

对应的频率： 100MHz/6

总线宽度： 6×4B=24B

所以数据传输速率=总线宽度/写操作时间=24×(100MHz/6)=400MB/s 。 3)平均传输速率=343×70%+400×30%=360.1MB/s。

290-2025年计算机组成原理考研复习指导



**6.2** **总线事务和定时**

**6.2.1** **总线事务**

从请求总线到完成总线使用的操作序列称为总 线 事务，它是在一个总线周期中发生的一系列 活动。典型的总线事务包括请求操作、仲裁操作、地址传输、数据传输和总线释放。

1)请求阶段。主 设 备 ( CPU 或 DMA) 发出总线传输请求，并且获得总线控制权。 2)仲裁阶段。总线仲裁机构决定将下一个传输周期的总线使用权授予某个申请者。 3)寻址阶段。主设备通过总线给出要访问的从设备地址及有关命令，启动从模块。

4)传输阶段。主模块和从模块进行数据交换，可单向或双向进行数据传送。 5 ) 释 放 阶 段 。主模块的有关信息均从系统总线上撤除，让出总线使用权。

命 题 造 原 ▶ **非突发传送的时间分析(2023)**

总线上的数据传送方式分为非突发方式和突发方式两种。非突发传送方式在每个传送周期内 都先传送地址，再传送数据，主、从设备之间通常每次只能传输一个字长的数据。

▶ **突发传送的概念(2014),突发传送的时间分析(2012、2013)**

突发(猝发)传送方式能够进行连续成组数据的传送，其寻址阶段发送的是连续数据单元的 首地址，在传输阶段传送多个连续单元的数据，每个时钟周期可以传送一个字长的信息，但是不 释放总线，直到一组数据全部传送完毕后，再释放总线。

**6.2.2** **总线定时**

总线定时是指总线在双方交换数据的过程中需要时间上配合关系的控制，这种控制称为总 线 定 时，其实质是一种协议或规则，主要有同步、异步、半同步和分离式四种定时方式。

命 题 追 踪▶ **各种总线定时方式的特点(2015、2021)**

**1.** **同步定时方式**

所 谓同步定时方式，是指系统采用一个统一的时钟信号来协调发送和接收双方的传送定时关 系。时钟产生相等的时间间隔，每个间隔构成一个总线周期。在一个总线周期中，发送方和接收 方可以进行一次数据传送。因为采用统一的时钟，每个部件或设备发送或接收信息都在固定的总 线传送周期中， 一个总线的传送周期结束，下一个总线的传送周期开始。

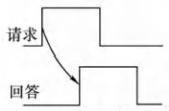
优点：传送速度快，具有较高的传输速率；总线控制逻辑简单。

缺点：主从设备属于强制性同步；不能及时进行数据通信的有效性检验，可靠性较差。 同步通信适用于总线长度较短及总线所接部件的存取时间比较接近的系统。

同步串行通信方式是发送方时钟直接控制接收方时钟，使双方完全同步的一种逐位传输的通 信方式。使用同步串行通信时，由于收发双方的时钟严格一致，因此仅在数据块的头尾处添加了 开始和结束标记，传输效率较高，但实现的硬件设备也更复杂，所以较少采用。

**2.** **异步定时方式**

在异步定时方式中，没有统一的时钟，也没有固定的时间间隔，完全依靠传送双方相互制约 的“握手”信号来实现定时控制。通常，主设备提出交换信息的“请求”信号，经接口传送到从



设备；从设备接到主设备的请求后，通过接口向主设备发出“回答”信号。

优点：总线周期长度可变，能保证两个工作速度相差很大的部件或设备之间可靠地进行信息 交换，自动适应时间的配合。

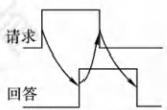
缺点：比同步控制方式稍复杂一些，速度比同步定时方式慢。

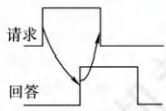
根据“请求”和“回答”信号的撤销是否互锁，异步定时方式又分为以下3种类型。

1)不互锁方式。主设备发出“请求”信号后，不必等到接到从设备的“回答”信号，而是经 过一段时间便撤销“请求”信号。而从设备在接到“请求”信号后，发出“回答”信号， 并经过一段时间后自动撤销“回答”信号。双方不存在互锁关系，如图6.4(a) 所示。

2)半互锁方式。主设备发出“请求”信号后，必须在接到从设备的“回答”信号后，才撤 销“请求”信号，有互锁的关系。而从设备在接到“请求”信号后，发出“回答”信号， 但不必等待获知主设备的“请求”信号已经撤销，而是隔一段时间后自动撤销“回答” 信号，不存在互锁关系。半互锁方式如图6.4(b) 所示。

3)全互锁方式。主设备发出“请求”信号后，必须在从设备“回答”后才撤销“请求”信 号；从设备发出“回答”信号后，必须在获知主设备“请求”信号已撤销后，再撤销其 “回答”信号。双方存在互锁关系，如图6.4(c)所示。



(b)半互锁

(a)不互锁

(c) 全互锁

图6.4 请求和回答信号的互锁

**命题追踪** ▶ **异步串行通信方式的特点(2016)**

现在越来越多的总线采用异步串行通信方式，使用异步串行通信时，由于收发双方时钟不严 格一致，因此每个字符都要用开始位和停止位作为字符开始和结束的标志，从而保证数据传输的 准确性。异步串行通信的第一位是开始位，表示字符传送的开始。当通信线上没有数据传送时处 于逻辑“1”状态，当发送方要发送一个字符时，首先发出一个逻辑“0”信号，即开始位。接收 方检测到这个逻辑低电平后，就开始准备接收数据位。在字符传送过程中，数据位从最低位开始， 一位 一位地传输。当字符发送完后，就可以发送奇偶校验位(可选),以用于有限的差错检测。 在奇偶位或数据位之后发送的是停止位，表示一个字符数据的结束。

**3.** **半同步定时方式**

半同步定时方式保留了同步定时的特点，如所有地址、命令、数据信号的发出时间都严格参 照系统时钟的某个前沿开始，而接收方都采用系统时钟后沿时刻来进行判断识别；同时，又像异 步定时那样，允许不同速度的设备和谐地工作。为此增设一条 Wait 响应信号线。例如，某个半同 步总线总是从某个时钟开始，在每个时钟到来时，采样 Wait 信号，若无效，则说明数据未准备好， 下个时钟到来时，再采样 Wait 信号，直到检测到有效，再去数据线上取数据。半同步定时适用于 系统工作速度不高，但又包含了由许多速度差异较大的各类设备组成的简单系统。

优点：控制方式比异步定时简单，各模块在系统时钟的控制下同步工作，可靠性较高。 缺点：系统时钟频率不能要求太高，所以从整体上来看，系统工作的速度不是很高。

以上三种定时方式都从主设备发出地址和读/写命令开始，直到数据传输结束，在整个传输周 期中，总线的使用权完全由主设备及由其选中的从设备占据。其实，从设备在准备数据的阶段，

292-2025年计算机组成原理考研复习指导

总线纯属空闲等待，为进一步挖掘总线的潜力，又提出了分离式定时方式。

**4.** **分离式定时方式**

分离式定时方式将总线事务分解为请求和应答两个子过程。在第一个子过程中，主设备A 获 得总线使用权后，将命令、地址等信息发到总线上，经总线传输后由从设备B 接收。此过程占用 总线的时间很短，主设备一旦发送完，立即释放总线，以便其他设备使用。在第二个子过程中， 设备B 收到设备A 发来的有关命令后，将设备A 所需的数据准备好后，便由设备B 申请总线使 用权， 一旦获准，设备B 便将相应的数据送到总线上，由设备A 接收。上述两个子过程都只有单 方向的信息流，每个设备都变为主设备。

优点：在不传送数据时释放总线，使总线可接受其他设备的请求，不存在空闲等待时间。 缺点：控制复杂，开销也大。

**6.2.3** **本** **节** **习** **题** **精** **选**

**一、单项选择题**

**01.** 在不同速度的设备之间传送数据，()。

A. 必须采用同步控制方式

B. 必须采用异步控制方式

C. 可以选用同步控制方式，也可选用异步控制方式

D. 必须采用应答方式

**02.** 某机器 I/O 设备采用异步串行传送方式传送字符信息，字符信息格式为1位起始位、7 位数据位、1位校验位和1位停止位。若要求每秒传送480个字符，则该设备的数据传 输速率为( )。

A.380b/s B.4800B/s C.480B/s D.4800b/s

**03.**假设某存储器总线采用同步通信方式，时钟频率为50MHz, 总线以突发方式传输8个字， 以支持块长为8字(每字4B) 的 Cache 行的读/写。若全部访问都为读操作，访问顺序 是1个时钟周期接收地址，3个时钟周期等待存储器读数，8个时钟周期用于传输8个 字。则该存储器的数据传输速率为( )。

A.114.3MB/s B.126MB/s C.133.3MB/s D.144.3MB/s

**04.** 同步控制方式是()。

A. 只适用于CPU 控制的方式 B. 只适用于外部设备控制的方式

C. 由统一的时序信号控制的方式 D. 所有指令执行时间都相同的方式

**05.** 同步通信之所以比异步通信具有较高的传输速率，是因为()。

A. 同步通信不需要应答信号且总线长度较短

B. 同步通信用一个公共的时钟信号进行同步

C. 同步通信中，各部件的存取时间较接近

D. 以上各项因素的综合结果

**06.** 以下各项中，()是同步传输的特点。

A. 需要应答信号 B. 各部件的存取时间比较接近

C. 总线长度较长 D. 总线周期长度可变

**07.** 在异步总线中，传送操作( )。

A. 由设备控制器控制 B. 由 CPU 控制

C. 由统一时序信号控制 D. 按需分配时间

**08.** 总线的异步通信方式是( )。

A. 既不采用时钟信号，又不采用“握手”信号

B. 只采用时钟信号，不采用“握手”信号

C. 不采用时钟信号，只采用“握手”信号

D. 既采用时钟信号，又采用“握手”信号

**09.** 在各种异步通信方式中，()的速度最快。

A. 全互锁 B. 半互锁 C. 不互锁 D. 速度均相等

**10.** 在下列各种情况下，最应采用异步传输方式的是( )。

A.I/O 接口与打印机交换信息 B.CPU 与主存交换信息

C.CPU 和 PCI 总线交换信息 D. 由统一时序信号控制方式下的设备

**11.** 在手术过程中，医生将手伸出，等护士将手术刀递上，待医生握紧后，护士才松手。若 把医生和护士视为两个通信模块，上述动作相当于( )。

A. 同步通信 B. 异步通信的全互锁方式

C. 异步通信的半互锁方式 D. 异步通信的不互锁方式

**12.** 一个总线传输周期依次包括以下几个阶段()。

① .握手阶段 ②. 寻址阶段 ③. 申请分配阶段 ④ . 传输阶段

A.①②③④ B.②③④ C.①③④ D.①②③

**13.**【2012统考真题】某同步总线的时钟频率为100MHz, 宽度为32位，地址/数据线复用， 每传输一个地址或数据占用一个时钟周期。若该总线支持突发(猝发)传输方式，则一 次“主存写”总线事务传输128位数据所需要的时间至少是( )。

A.20ns B.40ns C.50ns D.80ns

**14.** 【2014统考真题】一次总线事务中，主设备只需给出一个首地址，从设备就能从首地址

开始的若干连续单元读出或写入多个数据。这种总线事务方式称为()。

A. 并行传输 B. 串行传输 C. 突发传输 D. 同步传输

**15.**【2015统考真题】下列有关总线定时的叙述中，错误的是( )。

A. 异步通信方式中，全互锁协议最慢

B. 异步通信方式中，非互锁协议的可靠性最差

C. 同步通信方式中，同步时钟信号可由各设备提供

D. 半同步通信方式中，握手信号的采样由同步时钟控制

**16.** 【2016统考真题】下列关于总线设计的叙述中，错误的是( )。

A. 并行总线传输比串行总线传输速度快

B. 采用信号线复用技术可减少信号线数量

C. 采用突发传输方式可提高总线数据传输速率

D. 采用分离事务通信方式可提高总线利用率

**17.** 【2017统考真题】下列关于多总线结构的叙述中，错误的是( )。

A. 靠近 CPU 的总线速度较快 B. 存储器总线可支持突发传送方式

C. 总线之间须通过桥接器相连 D.PCl-Express×16 采用并行传输方式

**18.** 【2018统考真题】下列选项中，可提高同步总线数据传输速率的是( )。

1. 增加总线宽度 II. 提高总线工作频率

H. 支持突发传输 IV. 采用地址/数据线复用

A. 仅 I 、Ⅱ B. 仅 I 、Ⅱ 、IⅢ C. 仅 Ⅲ 、IV D.I 、Ⅱ 、Ⅲ 和 IV

**19.** 【2021 统考真题】下列关于总线的叙述中，错误的是( )。

294-2025年计算机组成原理考研复习指导

A. 总线是在两个或多个部件之间进行数据交换的传输介质

B. 同步总线由时钟信号定时，时钟频率不一定等于工作频率

C. 异步总线由握手信号定时， 一次握手过程完成一位数据交换

D. 突 发 (Burst) 传送总线事务可以在总线上连续传送多个数据

**20.【** 2023统考真题】某存储器总线宽度为64位，总线时钟频率为1GHz, 在总线上传输 一 个数据或地址需要一个时钟周期，不支持突发传送方式。若通过该总线连接 CPU 和 主 存，主存每次准备 一个64位数据需要6ns, 主存块大小为32B, 则读取一个主存块所需 的 时 间 是 ( ) 。

A.8ns B.11ns C.26ns D.32ns

**二、** **综合应用题**

**01.** 在异步串行传输方式下，起始位为1位，数据位为7位，偶校验位为1位，停止位为1 位，假设每秒传输1200比特，试问有效数据传输速率为多少?

**6.2.4** **答** **案** **与** **解** **析**

**一** **、单项选择题**

**01.C**

在不同速度的设备之间传送数据时，既可采用同步方式，又可采用异步方式。异步方式主要 用于在不同的设备间进行通信，两种速度不同的设备使用同一时钟进行控制时，采用同步控制方 式同样可以进行数据的传送，但不能发挥快速设备的高速性能。

02.D

一 个字符占用1+7+1+1=10位，因此数据传输速率为10×480=4800位/秒。

**03.C**

一次总线事务传输的数据量为8×4B=32B, 所需时钟周期数为1+3+8=12,每个时钟周期为 1/50MHz, 总时间为12×(/50MHz)=0.24s 。 数据传输速率为32B=[(12×(I/50MHz)s]=133.3MB/s。

**04.C**

同步控制是指由统一时序控制的通信方式，同步通信采用公共时钟，有统一 的时钟周期。同 步控制既可用于CPU 控制，又可用于高速的外部设备控制。

**05.D**

同步通信采用统一 的时钟，每个部件发送或接收信息都在固定的总线传送周期中， 一个总线 传送周期结束，开始下一个总线传送周期。它适用于总线长度较短且各部件的存取时间较接近的 情况，因此具有较高的传输速率。选项A、B、C 都是正确原因。

**06.B**

各部件的存取时间比较接近时，最适合采用同步传输，以发挥其优势。

07.D

异步总线即采用异步通信方式的总线。在异步方式下，没有公用的时钟，完全依靠传送双方 相互制约的“握手”信号来实现定时控制。传送操作是由双方按需求分配时间的。

**08.C**

异步通信方式也称应答方式，没有公用的时钟信号，也没有固定的时间间隔，完全依靠传送 双方相互制约的“握手”信号来实现定时控制。

**09.C**

在全互锁、半互锁和不互锁三种“握手”方式中，只有不互锁方式的请求信号和回答信号没



第6章

总线

有相互的制约关系，主设备发出请求信号后，不必等待回答信号的到来，便自己撤销了请求信号， 所以速度最快。

10.A

异步通信方式依靠“握手”信号来实现定时控制，能保证两个工作速度相差很大的部件或设 备之间可靠地进行信息交换。I/O 接口与打印机的速度差异较大，应采用异步传输方式。

**11.B**

由题意可知，医生是主模块，护士是从模块。医生伸出手后(即主模块发出请求信号),等 待护士将手术刀递上(主模块等待来自从模块的回答信号),护士也必须等待医生握紧后才松开 手(从模块等待主模块的回答信号),以上整个流程就是异步通信的全互锁方式。

**12.B**

总线的传输周期分为寻址阶段、申请分配阶段、传输阶段。

13.C

由于总线频率为100MHz, 因此时钟周期为10ns 。总线位宽与存储字长都是32位，因此每 个时钟周期可传送一个32 位存储字。猝发式发送可以连续传送地址连续的数据，因此总传送时 间为：传送地址10ns, 传送128位数据40ns, 共需50ns。

14.C

猝发(突发)传输是在一个总线周期中，可以传输多个存储地址连续的数据，即一次传输一 个地址和一批地址连续的数据，并行传输是在传输中有多个数据位同时在设备之间进行的传输， 串行传输是指数据的二进制代码在一条物理信道上以位为单位按时间顺序逐位传输的方式，同步 传输是指传输过程由统一的时钟控制。

15.C

在同步通信方式中，系统采用一个统一的时钟信号，而不由各设备提供，否则无法实现统一 的时钟。

**16.A**

初看起来会觉得A 正确，并行总线通常比串行总线传输速率快，但这不是绝对的。在时钟频 率较低的情况下，并行总线因为可以同时传输若干比特，速率确实比串行总线快。但是，随着技 术的发展，时钟频率越来越高，并行总线之间的相互干扰越来越严重，当时钟频率提高到一定程 度时，传输的数据已无法恢复。而串行总线因为导线少，线间干扰容易控制，反而可通过不断提 高时钟频率来提高传输速率，A 错误。总线复用是指一种信号线在不同的时间传输不同的信息， 它可使用较少的线路传输更多的信息，从而节省空间和成本，B 正确。突发传输是指在一个总线 周期中，可以传输多个存储地址连续的数据，即一次传输一个地址和一批地址连续的数据，C 正 确。分离事务通信是总线复用的一种，相比单一的传输线路可以提高总线的利用率， D 正确。

17.D

多总线结构用速率高的总线连接高速设备，用速率低的总线连接低速设备。 一般来说， CPU 是计算机的核心，是计算机中速度最快的设备之一，A 正确。突发传送方式把多个数据单元作为 一个独立传输处理，从而最大化设备的吞吐量。现实中一般用支持突发传送方式的总线来提高存 储器的读/写效率， B 正确。各总线通过桥接器相连，后者起流量交换作用。PCI-Express 总线都 采用串行数据包传输数据。

**18.B**

总线数据传输速率=总线工作频率×(总线宽度/8),所以I 和Ⅱ会影响总线数据传输速率。 采用突发(猝发)传输方式，可在一个总线周期内传输存储地址连续的多个数据字，因此能提高 传输效率。采用地址/数据线复用只是减少了线的数量，节省了成本，并不能提高传输速率。

296-2025年计算机组成原理考研复习指导

**19.C**

总线是在两个或多个设备之间进行通信的传输介质，A 正确。同步总线是指总线通信的双方 采用同一个时钟信号，但是一次总线事务不一定在一个时钟周期内完成，即时钟频率不一定等于 工作频率， B 正确。异步总线采用握手的方式进行通信，每次握手的过程完成一次通信，但是一 次通信往往会交换多位而非一位数据， C 错误。突发传送总线事务是指发送方在传输完地址后， 连续进行若干次数据的发送，D 正确。

**20.D**

每次传输需经过传输地址、准备数据和传输数据三个过程，分别需要1ns (时钟频率为1GHz, 因此时钟周期为1ns) 、6ns 和 1ns, 共 8ns 。总线宽度为64位，所以每次传输的数据为64位，主 存块大小为32B, 所以读取一个主存块需要传输4次，即8ns×4=32ns。

**二** **、综合应用题**

**01.【**解答】

在这样的一个数据帧中，有效数据位是7位，传输过程中发送的代码位共有1+7+1+1=10 位，所以有效数据传输速率为1200×7÷(1+7+1+1)=840b/s。



**6** **.3本章小结**

本章开头提出的问题的参考答案如下。

1)引入总线结构有什么好处?

引入总线结构主要有以下优点：

① 简化了系统结构，便于系统设计制造。

② 大大减少了连线数目，便于布线，减小体积，提高系统的可靠性。

③ 便于接口设计，所有与总线连接的设备均采用类似的接口。

④ 便于系统的扩充、更新与灵活配置，易于实现系统的模块化。

⑤ 便于设备的软件设计，所有接口的软件对不同的接口地址进行操作。

⑥ 便于故障诊断和维修，同时也能降低成本。

2)引入总线会导致什么问题?如何解决?

引入总线后，总线上的各个设备分时共享同一总线，当总线上多个设备同时请求使用总线 时就会导致冲突。为解决多个设备同时竞争总线控制权的问题，应当采用总线仲裁部件，以某 种方式选择一个主设备获得总线控制权，只有获得了总线控制权的设备才能开始数据传送。

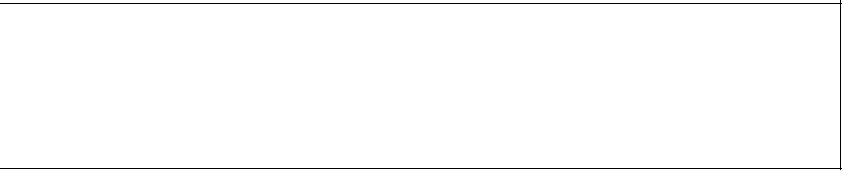


**6.4** **常见问题和易混淆知识点**

1. 一个总线在某一时刻可以有多对主从设备进行通信吗?

不可以。在某个总线周期内，总线上只有一个主设备控制总线，选择一个从设备与之进行通 信(即一对一的关系),或对所有设备进行广播通信(即一对多的关系)。所以一个总线在某一时 刻不能有多对主从设备进行通信，否则会发生数据冲突。





第 7 章

输入/输出系统

**【考纲内容】**

( 一 )I/O 接 口(I/O 控制器)

I/O 接口的功能和基本结构； I/O 端口及其编址

( 二 )I/O 方式

程序查询方式

程序中断方式：中断的基本概念；中断响应过程；中断处理过程； 多重中断和中断屏蔽的概念

扫一扫



视频讲解

DMA 方式： DMA 控制器的组成； DMA 传送过程

**【复习提示】**

I/O 方式是本章的重点和难点，每年不仅会以选择题的形式考查基本概念和原理，而且可能 会以综合题的形式考查，特别是各种I/O 方式效率的相关计算，中断方式的各种原理、特点、处 理过程、中断屏蔽， DMA 方式的特点、传输过程、与中断方式的区别等。

在学习本章时，请读者思考以下问题：

1)I/O 设备有哪些编址方式?各有何特点?

2)CPU 响应中断应具备哪些条件?

请读者在学习本章的过程中寻找答案，本章末尾将给出参考答案。



**\*7.1 I/O 系统基本概念①**

**\*7.1.1** **输** **入** **/** **输** **出** **系** **统**

输入/输出是以主机为中心而言的，将信息从外部设备传送到主机称为输入，反之称为输出。 输入/输出系统解决的主要问题是对各种形式的信息进行输入和输出的控制。

I/O 系统中的几个基本概念如下：

1 ) 外 部 设 备。包括输入/输出设备及通过输入/输出接口才能访问的外存储设备。

2 ) 接口。在各个外设与主机之间传输数据时进行各种协调工作的逻辑部件。协调包括传输 过程中速度的匹配、电平和格式转换等。

3)输入设备。用于向计算机系统输入命令和文本、数据等信息的部件。键盘和鼠标是最基 本的输入设备。

① 本节的内容2021年已从统考大纲中删除，仅供学习参考。

298-2025年计算机组成原理考研复习指导

4)输出设备。用于将计算机系统中的信息输出到计算机外部进行显示、交换等的部件。显 示器和打印机是最基本的输出设备。

5)外存设备。指除计算机内存及CPU 缓存等外的存储器，如硬磁盘、光盘等。 一 般来说， I/O 系统由I/O 软件和 I/O 硬件两部分构成：

1)I/O 软件。包括驱动程序、用户程序、管理程序、升级补丁等。通常采用I/O 指令和通道

指令实现 CPU 与 I/O 设备的信息交换。

2)/O 硬件。包括外部设备、设备控制器和接口、I/O 总线等。通过设备控制器来控制 I/O 设备的具体动作；通过I/O 接口与主机(总线)相连。

**\*7.1.2** **I/O 控制方式**

在输入/输出系统中，经常需要进行大量的数据传输，而传输过程中有各种不同的I/O 控制方 式，基本的控制方式主要有以下四种：

1)程序查询方式。由 CPU 通过程序不断查询I/O 设备是否已做好准备，从而控制I/O 设备 与主机交换信息。

2)程序中断方式。只 在I/O 设备准备就绪并向CPU 发出中断请求时才予以响应。

3)DMA 方式。主存和I/O 设备之间有一条直接数据通路，当主存和I/O 设备交换信息时， 无须调用中断服务程序。

4)通道方式。在系统中设有通道控制部件，每个通道都挂接若干外设，主机在执行 I/O 命令 时，只需启动有关通道，通道将执行通道程序，从而完成I/O 操作。

其中，方式1和方式2主要用于数据传输速率较低的外部设备，方式3和方式4主要用于数 据传输速率较高的设备。

**\*7.1.3** **外部设备**

最基本的外部设备主要有键盘、鼠标、显示器、打印机、磁盘存储器和光盘存储器等。

**1.** **输入设备**

(1)键盘

键盘是最常用的输入设备，通过它可发出命令或输入数据。

(2)鼠标

鼠标是常用的定位输入设备，它把用户的操作与计算机屏幕上的位置信息相联系。

**2.** **输出设备**

(1)显示器

按所用的显示器件分类，有阴极射线管 (CRT) 显示器、液晶显示器 (LCD) 、 发光二极管

(LED) 显示器等。显示器属于用点阵方式运行的设备，有以下主要参数。

1)屏幕大小：以对角线长度表示，单位为英寸，常用的有12～32英寸等。

2)分辨率：能表示的像素个数，屏幕上的每个光点就是一个像素，以宽和高的像素数的乘

积表示，如800×600、1024×768和1280×1024等。

3)灰度级：指黑白显示器中所显示的像素点的亮暗差别，在彩色显示器中则表现为颜色的

不同，灰度级越多，图像层次越清楚、逼真，典型的有8位(256级)、16位等。

4 ) 刷 新：光点只能保持极短的时间便会消失，为此必须在光点消失之前再重新扫描显示一 遍，这个过程称为刷新。

5)刷新频率：指单位时间内扫描整个屏幕内容的次数。按照人的视觉生理，刷新频率大于 30Hz 时才不会感到闪烁，通常显示器的刷新频率为60～120Hz。

**第7章** **输入/输出系统** 299

命 题 追 踪 ▶ **显存刷新带宽的计算(2010)**

6)显示存储器 (VRAM): 也称刷新存储器，为了不断提高刷新图像的信号，必须把一帧图 像信息存储在刷新存储器中。其存储容量由图像分辨率和灰度级决定，分辨率越高，灰 度级越多，刷新存储器容量越大。

VRAM 容 量 = 分 辨 率x灰度级位数

VRAM 带宽=分辨率×灰度级位数×帧频

(2 ) 打印机

用于将计算机的处理结果打印在相关介质上。按工作方式，打印机可分为点阵打印机、针式 打印机、喷墨式打印机、激光打印机等。

1)针式打印机。针式打印机擅长“多层复写打印”,实现各种票据或蜡纸等的打印。其工作

原理简单，造价低廉，耗材(色带)便宜，但打印分辨率和打印速度不够高。

2)喷墨式打印机。彩色喷墨打印机基于三基色原理，即分别喷射三种颜色的墨滴，按一定

的比例混合出所要求的颜色。喷墨式打印机可实现高质量彩色打印。

3)激光打印机。计算机输出的二进制信息，经过调制后转换为激光束，在感光鼓上形成潜

像，再经过显影、转印和定影，在纸上得到所需的字符或图像。激光打印机打印质量高、 速度快、处理能力强，它是将激光技术和电子显像技术相结合的产物。

**3.** **外部存储器(辅存)**

(1)磁表面存储器

所谓“磁表面存储”,是指把某些磁性材料薄薄地涂在金属铝或塑料表面上作为载磁体来存 储信息。磁盘存储器、磁带存储器和磁鼓存储器均属于磁表面存储器。

(2)固态硬盘 (SSD)

微小型高档笔记本电脑采用高性能 Flash 存储器作为硬盘来记录数据，这种“硬盘”称为固 态硬盘 (SSD) 。 固态硬盘除需要Flash 存储器外，还需要其他硬件和软件的支持。

(3 ) 光盘存储器

光盘存储器是利用光学原理读/写信息的存储装置，它采用聚焦激光束对盘式介质以非接触方 式记录信息。完整的光盘存储系统由光盘片、光盘驱动器、光盘控制器等组成。

**\*7.1.4** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 在微型机系统中， I/O 设备通过( )与主板的系统总线相连接。

A.DMA 控制器 B. 设备控制器 C. 中断控制器 D.I/O 端口

**02.** 显示汉字采用点阵字库，若每个汉字用16×16的点阵表示，7500个汉字的字库容量是()。

A.16KB B.240KB C.320KB D.1MB

**03.** CRT 的分辨率为1024×1024像素，像素的颜色数为256,则刷新存储器的每单元字长为 ( ),总容量为( )。

A.8B,256MB B.8bit,1MB C.8bit,256KB D.8B,32MB

**04.【**2010统考真题】假定一台计算机的显示存储器用DRAM 芯片实现，若要求显示分辨率 为1600×1200,颜色深度为24位，帧频为85Hz, 显存总带宽的50%用来刷新屏幕，则 需要的显存总带宽至少约为( )。

A.245Mb/s B.979Mb/s C.1958Mb/s D.7834Mb/s

300-2025年计算机组成原理考研复习指导

**\*7.1.5** **答案与解析**

**单项选择题**

**01.B**

I/O 设备不可能直接与主板总线相连，它总是通过设备控制器来相连的。

**02.B**

每个汉字占用16×16/8=32B, 则汉字库容量=7500×32B=240000B≈240KB。

**03.B**

刷新存储器中存储单元的字长取决于显示的颜色数，颜色数为m, 字 长 为n, 二者的关系为 2”=m 。 本题中的颜色数为256=28,因此刷新存储器单元字长为8位。刷新存储器的容量是每个 像素点的位数和像素点个数的乘积，因此刷新存储器的容量为1024×1024×8bit=1MB。

**04.D**

刷新所需带宽=分辨率×色深×帧频=1600×1200×24bit×85Hz=3916.8Mb/s, 显存总带宽的 50%用来刷新屏幕，于是需要的显存总带宽至少为3916.8/0.5=7833.6 Mb/s≈7834Mb/s 。



**7.2** **I/O** **接** **口**

I/O 接 口 ( 也 称 I/O 控制器)是主机和外设之间的交接界面，通过接口可以实现主机和外设 之间的信息交换。外设种类繁多，且具有不同的工作特性，它们在工作方式、数据格式和工作速 度等方面有着很大的差异，接口正是为了解决这些差异而设置的。

**7.2.1** **I/O** **接口的功能**

▶ VO 接口的定义与特性(2021)

I/O 接口的主要功能如下：

1)进行地址译码和设备选择。CPU 送来选择外设的地址码后，接口必须对地址进行译码以

产生设备选择信息，使主机能和指定外设交换信息。

2)实现主机和外设的通信联络控制。解决主机与外设时序配合问题，协调不同工作速度的

外设和主机之间交换信息，以保证整个计算机系统能统一 、协调地工作。

3)实现数据缓冲。CPU 与外设之间的速度往往不匹配，为消除速度差异，接口必须设置数

据缓冲寄存器，用于数据的暂存，以避免因速度不一致而丢失数据。

4)信号格式的转换。外设与主机两者的电平、数据格式都可能存在差异，接口应提供主机与

外设的信号格式的转换功能，如电平转换、并/串或串/并转换、模/数或数/模转换等。

5)传送控制命令和状态信息。CPU 要启动某外设时，通过接口中的命令寄存器向外设发出

启动命令；外设准备就绪时，则将“准备好”状态信息送回接口中的状态寄存器，并反 馈给 CPU 。 外 设 向CPU 提出中断请求时， CPU 也应有相应的响应信号反馈给外设。

**7.2.2** **I/O** **接口的基本结构**

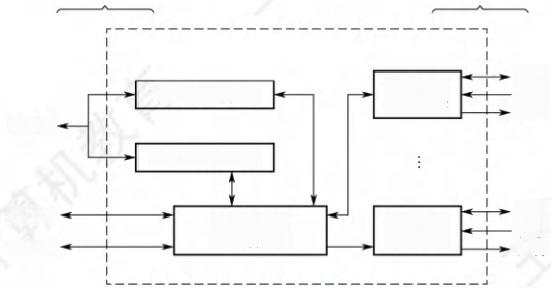
▶ **VO 端口与** **CPU 交换的内容(2015)**

图7 . 1所示是 一个I/O 接口的通用结构， I/O 接口在主机侧通过I/O 总线与内存、CPU 相连。

**第7章** **输入/输出系统** -301

数据缓冲寄存器用来暂存与CPU 或内存之间传送的数据信息，状态寄存器用来记录接口和设备的 状态信息，控制寄存器用来保存 CPU 对外设的控制信息。状态寄存器和控制寄存器在传送方向上 是相反的，在访问时间上也是错开的，因此可将它们合二为一。

主机侧(系统总线) 设备侧(接口电缆)



外设界面 控制逻辑

数据线 ·

状态/控制寄存器

地址译码和

/O 控制逻辑

地址线

控制线

外设界面 控制逻辑

数 据 状 态 控 制

数 据 状 态 控 制

数据缓冲寄存器

图 7.1 I/O 接口的通用结构

命题追踪 ▶I **/O 接口的数据线上传输的内容(2012)**

I/O 接口中的数据线传送的是读/写数据、状态信息、控制信息和中断类型号。 地 址 线传 送 的 是 要 访 问I/O 接口中的寄存器的地址。 控 制 线传送的是读/写控制信号，以确认是读寄存器还是 写寄存器，此外控制线还会传送中断请求和响应信号、仲裁信号和握手信号。

I/O 接口中的 I/O 控制逻辑还要能对控制寄存器中的命令字进行译码，并将译码得到的控制 信号通过外设界面控制逻辑送到外设，同时将数据缓冲寄存器的数据发送到外设或从外设接收数 据到数据缓冲寄存器。另外，它还要具有收集外设状态到状态寄存器的功能。

对数据缓冲寄存器、状态/控制寄存器的访问操作是通过相应的指令来完成的，通常称这类指

令为 I/O 指 令 ，I/O 指令只能在操作系统内核的底层I/O 软件中使用，它们是一种特权指令。

**7.2.3** **I/O 接口的类型**

从不同的角度看，I/O 接口可以分为不同的类型。

1)按数据传送方式(外设和接口 一侧),可分为并行接口( 一 字节或 一个字的所有位同时传

送 ) 和串行接口(一位一位地有序传送),接口要完成数据格式的转换。

2 ) 按 主 机 访 问 /O 设备的控制方式，可分为程序查询接口、中断接口和 DMA 接口等。

3)按功能选择的灵活性，可分为可编程接口 (通过编程改变接口功能)和不可编程接口。

**7.2.4** **I/O 端口及其编址**

命 题 追 踪 ▶> **VO 端口的定义及相关特性(2014)**

I/O 端 口是 指I/O 接口电路中可被 CPU 直接访问的寄存器，主要有数据端口、状态端口和控 制端口。通常， CPU 能够对数据端口中的数据进行读/写操作；但对状态端口中的外设状态只能 进行读操作，对控制端口中的各种控制命令只能进行写操作。



**注** **意**

端口和接口是两个不同的概念，端口是指接口电路中可以进行读/写的寄存器。

302-2025年计算机组成原理考研复习指导

I/O 端口要想能够被CPU 访问，就必须要对各个端口进行编址，每个端口对应一个端口地址。 而对I/O 端口的编址方式有与存储器独立编址和统一编址两种。

(1 ) 独立编址

**▶** **VO 指令的作用(2017)**

独立编址也称 I/O 映射方式，是指对所有的I/O 端口单独进行编址。I/O 端口的地址空间与主 存地址空间是两个独立的地址空间，它们的范围可以重叠，相同地址可能属于不同的地址空间。 因此需设置专门的VO 指令来表明访问的是/O 地址空间，I/O 指令的地址码给出I/O 端口号。

优点：I/O 端口数比主存单元数少得多，只需少量地址线，使得I/O 端口译码简单，寻址速度 更快。使用专用I/O 指令，可使得程序更加清晰，便于理解和检查。

缺点：/O 指令少，只提供简单的传输操作，所以程序设计的灵活性较差。此外，CPU 需要 提供存储器读/写 、I/O 设备读/写两组控制信号，增大了控制的复杂性。

(2 ) 统一编址

统一编址也称存储器映射方式，是指把主存地址空间分出一部分给 I/O 端口进行编址，I/O 端口和主存单元在同一地址空间的不同分段中，根据地址范围就能区分访问的是 I/O 端口还是主 存单元，因此无须设置专门的I/O 指令，用统一的访存指令就可访问I/O 端口。

优点：不需要专门的/O指令，使得CPU 访 问I/O 的操作更加灵活和方便，还使端口有较大 的编址空间。I/O 访问的保护机制可由虚拟存储管理系统来实现，无须专门设置。

缺点：端口地址占用了部分主存地址空间，使主存可用容量变小。此外，由于在识别I/O 端 口时全部地址线都需要参加译码，使得译码电路变得更复杂，降低了译码速度。

**7.2.5** **本** **节** **习** **题** **精** **选**

**单项选择题**

**01.** 在统一编址的方式下，区分存储单元和I/O 设备是靠()。

A. 不同的地址码 B. 不同的地址线

C. 不同的控制线 D. 不同的数据线

**02.** 下列功能中，属于I/O 接口的功能的是( )。

I. 数据格式的转换 II.I/O 过程中错误与状态检测

Ⅲ.I/O 操作的控制与定时 IV. 与主机和外设通信

A.I 和IV B.I 、Ⅲ和IV C.I 、Ⅱ 和IV D.I 、I 、Ⅲ 和IV **03.** 下列关于 I/O 端口和接口的说法中，正确的是( )。

A. 按照不同的数据传送格式，可将接口分为同步传送接口和异步传送接口

B. 在统一编址方式下，存储单元和I/O 设备是靠不同的地址线来区分的 C. 在独立编址方式下，存储单元和I/O 设备是靠不同的地址线来区分的 D. 在独立编址方式下，CPU 需要设置专门的输入/输出指令访问端口

**04.** 下列属于I/O 接口中寄存器的有( )。

I. 指令寄存器 II. 控制寄存器 ⅢI. 状态寄存器

V. 数据缓冲寄存器 IV. 地址寄存器

A.I 、Ⅱ 、Ⅲ 和 V B.Ⅱ 、Ⅲ 和 IV C.Ⅱ 、Ⅲ 和 V D.Ⅱ 、Ⅲ 、IV 和 V

**05.** I/O 的编址方式采用统一编址方式时，进行输入/输出的操作的指令是( )。

A. 控制指令 B. 访存指令 C. 输入/输出指令 D. 都不对

第7章 输入/输出系统-303

**06.** 下列关于I/O指令的说法中，错误的是()。

A.I/O 指令是 CPU 系统指令的一部分

B.I/O 指令是机器指令的一类

C.I/O 指令反映 CPU 和 I/O 设备交换信息的特点

D.I/O 指令的格式和通用指令的格式相同

**07.** 下列叙述中，正确的是()。

A. 只有I/O 指令可以访问I/O 设备

B. 在统一编址下，不能直接访问I/O 设备

C. 访问存储器的指令一定不能访问I/O 设备

D. 只有在具有专门I/O 指令的计算机中， I/O 设备才可以单独编址

**08.** 在内存地址空间与接口地址空间统一编址的计算机中，不需要的指令是()。 A. 数据传送类(如MOV 指令)

B. 算术、逻辑运算类(如ADD 、SUB 、AND 和 OR 指令)

C. 输入/输出类(如IN 和 OUT 指令)

D. 程序控制类(如条件转移指令和子程序调用指令)

**09.** 在统一编址的情况下，就I/O 设备而言，其对应的I/O 地址不可取的是()。

A. 要求固定在地址高端 B. 要求固定在地址低端

C. 要求相对固定在地址的某部分 D. 可以随意在地址的任何地方

**10.** 磁盘驱动器向盘片磁道记录数据时采用()方式写入。

A. 并行 B. 串行 C. 并行-串行 D. 串行-并行

**11.** 程序员进行系统调用访问设备使用的是()。

A. 逻辑地址 B. 物理地址 C. 主设备地址 D. 从设备地址

**12.** 采用中断方式进行打印控制时，在打印控制接口和打印机之间交换的信息不包括()。

A. 打印字符点阵信息 B. 打印控制信息

C. 打印机状态信息 D. 中断请求信号

**13.** 主机和外设之间的正确连接通路是( )。

A.CPU 和主存—I/O总线—通信总线(电缆)—/O 接口—外设

B.CPU 和主存—I/O 总线—I/O 接口—通信总线(电缆)—外设

C.CPU 和主存—I/O 接口—I/O 总线—通信总线(电缆)—外设

D.CPU 和主存— I/O 接口—通信总线(电缆)—I/O 总线—外设

**14.** 下列有关I/O接口功能和结构的叙述中，错误的是()。

A.I/O 接口中主机侧数据宽度与设备侧数据宽度总是一样的

B.I/O 接口是像显卡或网卡之类的一种外设控制逻辑

C.CPU 可以从 I/O 接口读取状态信息，以了解接口和外设的状态

D.CPU 可以向I/O 接口传送用来对设备进行控制的命令

**15.**【2012统考真题】下列选项中，在I/O 总线的数据线上传输的信息包括( )。

I./O 接口中的命令字 IⅡ.I/O 接口中的状态字 Ⅲ . 中断类型号

A. 仅 I 、Ⅱ B. 仅 I 、Ⅲ C. 仅Ⅱ、Ⅲ D.I 、Ⅱ、Ⅲ

**16.**【2014统考真题】下列有关 I/O 接口的叙述中，错误的是( )。

A. 状态端口和控制端口可以合用同一个寄存器

B.I/O 接口中CPU 可访问的寄存器称为I/O 端口

304-2025年计算机组成原理考研复习指导

C. 采用独立编址方式时， I/O 端口地址和主存地址可能相同

D. 采用统 一 编址方式时， CPU 不能用访存指令访问I/O 端口

17. 【2017统考真题】I/O 指令实现的数据传送通常发生在()。

A.I/O 设 备 和I/O 端口之间 B. 通用寄存器和 I/O 设备之间

C.I/O 端 口 和I/O 端口之间 D. 通用寄存器和 I/O 端口之间

18. 【2021统考真题】下列选项中，不属于I/O 接 口 的 是 ( ) 。

A. 磁盘驱动器 B. 打印机适配器 C. 网络控制器 D. 可编程中断控制器

**7.2.6** **答** **案** **与** **解** **析**

**单项选择题**

**01.A**

在统一编址的情况下，没有专门的 I/O 指令，因此用访存指令来实现 I/O 操作，区分存储单 元和 I/O 设备是靠它们各自不同的地址码。

**02.** D

I/O 接口的功能有：①选址功能、②传送命令功能、③传送数据功能、④反映 I/O 设备工作 状态的功能。选项I 可参考唐朔飞老师的《计算机组成原理》教材，为设置接口的原因之一，也 是接口应具有的功能；选项Ⅱ属于④;选项Ⅲ属于②;选项IV 属于③。

**03.D**

D 显然正确。按照不同的数据传送格式，可将接口分为并行接口和串行接口，A 错误；在统一编 址方式下，存储单元和/O 设备是靠不同的地址码而非地址线来区分的， B 错误；在独立编址方式下， 存储单元和I/O 设备是靠不同的指令来区分的，C 错误。

**04.C**

I/O 接口中的寄存器主要有数据缓冲寄存器、控制寄存器和状态寄存器。

**05.B**

统一编址时，直接使用指令系统中的访存指令来完成输入/输出操作；独立编址时，则需要使 用专门的输入/输出指令来完成输入/输出操作。

**06.D**

I/O 指令是指令系统的一部分，是机器指令的一类，但其为了反映与 I/O 设备交互的特点， 格式和其他通用指令相比有所不同。

**07.D**

在统一编址的情况下，访存指令也可访问I/O 设 备 ，A 、B 、C 错误。在独立编址的方式下， 访 问I/O 地址空间必须通过专门的I/O 指 令 ，D 正确。

**08.** C

统一编址方式把I/O 端口当作存储器的单元进行地址分配， CPU 不需要设置专门的I/O 指令 (即输入/输出类指令),用统一 的访存指令就可以访问I/O 端 口 。

**09.D**

在统一编址方式下，指令靠地址码区分内存和I/O 设备，若随意在地址的任何地方编址，将 给编程造成极大的混乱， D 错误。选项A 、B 、C 的做法都是可取的。

**10.B**

磁盘驱动器向盘片磁道记录数据时采用串行方式写入。

**11.A**

物理地址是外部连接使用的，且是唯一的，它与“地址总线相对应”;而逻辑地址是内部和

第7章 输入/输出系统-305

编程使用的，并不唯一。在内存中的实际地址就是所谓的“物理地址”,而逻辑地址就是用于逻 辑段管理内存的，因此程序员使用逻辑地址访问设备。

12.D

对打印机的中断控制过程通常是： CPU 先将需要打印的字符编码送到打印控制接口(也称打 印适配器)中，打印控制接口再将字符编码转换为点阵信息，然后通过电缆传送到打印机，以控 制打印针头在何处进行打印。同时，打印控制接口需要将“初始化”“选通”“自动走纸”等控 制信息通过电缆传送到打印机，并通过电缆把打印机的“联机”“忙”“缺纸”等状态信号取到 打印控制接口，以供CPU 读取。中断请求信号是打印控制接口通过中断控制器发送给CPU 的 ， 因此不在打印控制接口和打印机之间进行交换，D 错 误。

13.B

CPU 和主存通过I/O 总 线 和I/O 接口连接，/O 接口通过通信总线和外设相连。

**14.** A

I/O 接口中主机侧通过 I/O 总线与主机相连，设备侧通过通信总线(电缆)与外设相连。显 然， I/O 总线中的数据线宽度和连接设备的电缆中的数据线宽度不一定相同。

**15.** D

I/O 总线分为三类：数据线、控制线和地址线。数据缓冲寄存器和命令/状态寄存器的内容都 是通过数据线来传送的；地址线用以传送与CPU 交换数据的端口地址；而控制线用于给I/O 端口 发送读/写信号，仅用于对端口进行读/写控制。中断类型号用于指出中断向量的地址， CPU 响应 某一外部中断后，就从数据总线上获取该中断源的中断类型号，然后据此计算对应中断向量在中 断向量表(存放在内存中)的位置。因此1、Ⅱ和Ⅲ均正确。

**16.D**

采用统一编址时， CPU 访存和访问I/O 端口用的是一样的指令，所以访存指令可访问/O 端 口 ，D 错误。其他三个选项均为正确陈述。

17.D

I/O 端口是指I/O 接口中用于缓冲信息的寄存器，由于主机和 VO 设备的工作方式和工作速度 有很大差异，I/O 端口应运而生。在执行一条指令时， CPU 使用地址总线选择所请求的/O 端口， 使用数据总线在CPU 寄存器和端口之间传输数据。

**18.** A

I/O 接口即I/O 控制器，其功能是接收主机发送的 I/O 控制信号，并实现主机和外部设备之间 的信息交换。磁盘驱动器是由磁头、磁盘和读/写电路等组成的，也就是我们平常所说的磁盘本身， A 错误。选项 B 、C 和 D 均 为I/O 控制器。



**7.3 I/O 方** **式**

输入/输出系统实现主机与I/O 设备之间的数据传送，可以采用不同的控制方式，各种方式在 代价、性能、解决问题的着重点等方面各不相同，常用的 I/O 方式有程序查询、程序中断和 DMA

等，其中前两种方式更依赖于CPU 中程序指令的执行。

**7.3.1** **程序查询方式**

信息交换的控制直接由 CPU 执行程序实现。程序查询方式接口中设置一个数据缓冲寄存器

306-2025年计算机组成原理考研复习指导

(数据端口)和 一个设备状态寄存器(状态端口)。主机进行 I/O 操作时，先读取设备的状态并根 据设备状态决定下一步操作究竟是进行数据传送还是等待。

▶ **程序查询方式的特点(2023)**

程序查询方式的工作流程如下(见图7.2):

① CPU 执行初始化程序，并预置传送参数。

② 向 I/O 接口发出命令字，启动I/O 设备。

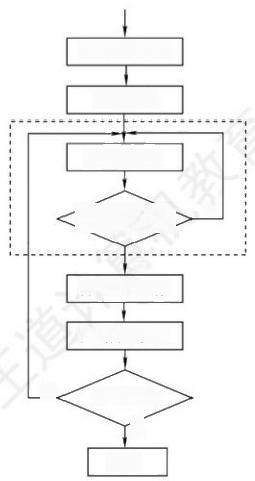
③ 从外设接口读取其状态信息。

④ CPU 周期或持续的查询设备状态，直到外设准备就绪。

⑤ 传送一次数据。

⑥ 修改地址和计数器参数。

⑦ 判断传送是否结束，若未结束转第③步，直到计数器为0。



预置传送参数

启动外设

取外设状态

外设准备就结少

Y

传送一次数据

修改传送参数

N 传送完否?

√

结束

N

图7.2 程序查询方式流程图

根据上述流程④中查询方式的不同，程序查询方式可分为如下两类。

1) **独** **占** **查** **询。**一 旦设备被启动， CPU 就一直持续查询接口状态， CPU 花费100%的时间用 于I/O 操作，此时外设和CPU 完全串行工作。

2) **定时查询**。CPU 周期性地查询接口状态，每次总是等到条件满足才进行一个数据的传送， 传送完成后返回到用户程序。定时查询的时间间隔与设备的数据传输速率有关。

命 题 追 踪▶ **定时查询的特点、效率分析及计算(2011、2018)**

**【例7.1】**假设计算机的主频为500MHz,CPI 为4,某外设的数据率为2MB/s,I/O 接口中有 一个32位数据缓冲寄存器，采用定时查询方式，每次I/O 都执行10条指令。外设最多间隔多长 时间查询一次才能不丢失数据? CPU 用于外设I/O 的 时 间 占CPU 总时间的百分比至少是多少?

解 ：

由于端口大小有限，必须在外设传输完端口大小的数据时访问端口，以防止数据未被及时读

第7章 输入/输出系统-307

取而丢失。外设准备32位数据所用的时间为4B+2MB/s=2μs, 所以最多每隔2 μs 就必须查询 一 次，即每秒的查询次数至少是1s÷2μs=5×10°, 每 秒CPU 用于外设I/O 的时间至少为5×10⁵ ×10×4 =2 ×107个时钟周期，占整个 CPU 时间的百分比至少是2×107/500M=4%。

程序查询方式的优 点 是设计简单、硬件量少。 缺点是CPU 要花费很多时间来查询和等待， 且在一段时间内只能和一 台外设交换信息，CPU 与设备串行工作，效率很低。

7.3.2 程序中断方式

1. 程序中断的基本概念

程序中断是指在计算机执行程序的过程中，出现某些急需处理的异常情况或特殊请求， CPU 暂时中止现行程序，而转去对这些异常情况或特殊请求进行处理，处理完毕后再返回到原程序的 断点处，继续执行原程序。早期的中断技术是为了处理数据传送。

**命** **题** **追** **踪** ▶ **程序中断方式的特点(2022、2023)**

随着计算机的发展，中断技术不断被赋予新的功能，主要功能有：

① 实现 CPU 与 I/O 设备的并行工作。

② 处理硬件故障和软件错误。

③ 实现人机交互，用户干预机器需要用到中断系统。

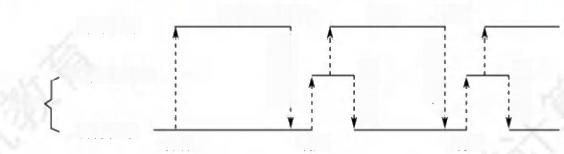
④ 实现多道程序、分时操作，多道程序的切换需借助于中断系统。

⑤ 实时处理需要借助中断系统来实现快速响应。

⑥ 实现应用程序和操作系统(管态程序)的切换，称为软中断。

⑦ 多处理器系统中各处理器之间的信息交流和任务切换。

程序中断方式的思想：CPU 在程序中安排好在某个时机启动某台外设，然后 CPU 继续执行 当前的程序，不需要像查询方式那样一直等待外设准备就绪。 一旦外设完成数据传送的准备工作， 就主动向 CPU 发出中断请求。在可以响应中断的条件下， CPU 暂时中止正在执行的程序，转去 执行中断服务程序为外设服务，在中断服务程序中完成一次主机与外设之间的数据传送，传送完 成后， CPU 返回原来的程序。此时，外设和CPU 又开始并行工作，如图7.3所示。



工作 完成

中

中断服务程序 ::

求

启动 响应

外部设备

现行程序

中断返回

中断请求

中断返回

完成

应 响

工作

CPU

图7.3 程序中断方式示意图

命 题 追 踪 ▶ **程序中断的效率分析及相关计算(2009、2014、2016、2018、2019)**

**【例7.2】**假定计算机的主频为500MHz,CPI 为4,某外设的数据率为40MB/s,I/O 接口中 有 一个32位数据缓冲寄存器。在 中 断I/O 方式下，若每次中断响应和中断处理的总时钟周期数至 少为400,则该外设能否采用中断 VO 方式?为什么?

解 ：

中断响应和中断处理的时间为400×1/500M=0.8μs, 只需计算外设准备32位数据的时间，若 准备数据的时间小于中断响应和中断处理的时间，则数据被刷新，造成丢失。外设准备32 位数

据所用的时间为4B÷40MB/s=0.1μs, 因此该外设不适合采用中断I/O 方式。



308-2025 年计算机组成原理考研复习指导

**2.** **程序中断的工作流程**

▶ **中断工作流程中的相关细节(2017、2018、2021)**

(1)中断请求

中断源是请求 CPU 中断的设备或事件， 一台计算机允许有多个中断源。每个中断源向 CPU 发出中断请求的时间是随机的。为记录中断事件并区分不同的中断源，中断系统需对每个中断源 设 置中断请求标记触发器，当其状态为“1”时，表示该中断源有请求。这些触发器可组成中断 请求标记寄存器，该寄存器可集中在 CPU 中，也可分散在各个中断源中。

命 题 追 踪▶ **可屏蔽中断和不可屏蔽中断的特点(2020)**

通过INTR 线发出的是可屏蔽中断，通过NMI 线发出的是不可屏蔽中断。可屏蔽中断的优先 级最低，在关中断模式下不被响应。不可屏蔽中断用于处理紧急和重要的事件，如时钟中断、电 源掉电等，其优先级最高，其次是内部异常，即使在关中断模式下也被响应。

(2)中断响应判优

中断响应优先级是指 CPU 响应中断请求的先后顺序。由于许多中断源提出中断请求的时间 都是随机的，因此当多个中断源同时提出请求时，需通过中断判优逻辑来确定响应哪个中断源的 请求，中断响应的判优通常是通过硬件排队器(或中断查询程序)实现的。

一 般来说，①不可屏蔽中断>内部异常>可屏蔽中断；②在内部异常中，硬件故障>软件 中断；③DMA 中 断 请 求 > I/O 设备的中断请求；④在I/O 传送类中断请求中，高速设备>低速 设备；输入设备>输出设备；实时设备>普通设备。



**注** **意**

中断优先级包括响应优先级和处理优先级，响应优先级由硬件线路或查询程序的查询顺序 决定，不可动态改变。处理优先级可利用中断屏蔽技术动态调整，以实现多重中断。

(3)CPU 响应中断的条件

命 题 追 踪 ▶ **CP** **U** **响应中断的条件(2023)**

CPU 在满足一定的条件下响应中断源发出的中断请求，并经过一些特定的操作，转去执行中 断服务程序。CPU 响应中断必须满足以下3个条件：

① 中断源有中断请求。

② CPU 允许中断及开中断(异常和不可屏蔽中断不受此限制)。

③ 一条指令执行完毕 ( 异常不受此限制),且没有更紧迫的任务。



**注** **意**

I/O 设备的就绪时间是随机的，而 CPU 在统一 的时刻即每条指令执行结束时，采样中断请 求信号(开中断的情况下),以获取 I/O 的中断请求，也就是说， CPU 响应中断的时间是在每 条指令执行阶段的结束时刻。这里说的中断仅指 I/O 中断，异常不属于此类情况。

(4)中断响应过程

CPU 响应中断后，经过某些操作，转去执行中断服务程序。这些操作是由硬件直接实现的， 我们将它称为中断隐指令。中断隐指令并不是指令系统中的一条真正的指令，只是一种虚拟的说 法，本质上是硬件的一系列自动操作。它所完成的操作如下：

1 ) 关 中 断。CPU 响应中断后，首先要保护程序的断点和现场信息，在保护断点和现场的过



第7章 输入/输出系统-309

程 中 ，CPU 不能响应更高级中断源的中断请求。否则，若断点或现场保存不完整，在中

断服务程序结束后，就不能正确地恢复并继续执行现行程序。

2)保存断点。为保证在中断服务程序执行完后能正确地返回到原来的程序，必须将原程序 的断点(指令无法直接读取的 PC 和 PSW 的内容)保存在栈或特定寄存器中°。

注意异常和中断的差异：异常指令通常并没有执行成功，异常处理后要重新执行，所以 其断点通常是当前指令的地址。中断的断点则是下一条指令的地址。

3)引出中断服务程序。识别中断源，将对应的服务程序入口地址送入程序计数器PC 。 有 两

种方法识别中断源：硬件向量法和软件查询法。本节主要讨论比较常用的向量中断。 (5)中断向量

中断识别分为向量中断和非向量中断两种。非向量中断即软件查询法，第5章已介绍。

每个中断源都有一个唯一的类型号，每个中断类型号都对应一个中断服务程序，每个中断服 务程序都有一个入口地址，即中断向量，CPU 必须找到入口地址。把系统中的全部中断向量集中 存放到存储器的某个区域内，这个存放中断向量的存储区就称为中断向量表。

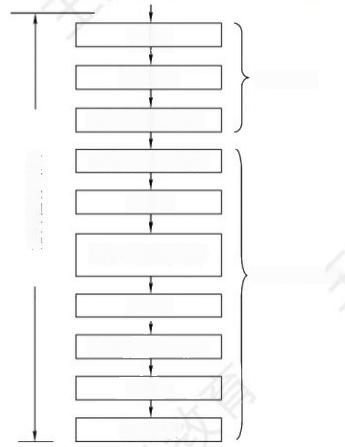
**命** **题** **追** **踪** ▶ **中断向量表的数据结构(2023)**

CPU 响应中断后，通过识别中断源获得中断类型号，然后据此计算出对应中断向量的地址； 再根据该地址从中断向量表中取出中断服务程序的入口地址，并送入程序计数器 PC, 以转去执 行中断服务程序，这种方法被称为中断向量法，采用中断向量法的中断被称为向量中断。

|  |  |
| --- | --- |
| **注** **意** |  |
| **中断请求和响应信号是在I/O总线的控制线上传送的。CPU响应某一中断后，就从数据线** **上获取该中断源的中断类型号，并据此计算对应中断向量在中断向量表中的位置。** | |

(6)中断处理过程

不同计算机的中断处理过程各具特色，图7.4所示为一个可嵌套中断的典型处理流程。



关中断

保存断点

中断服务程序寻址

保存现场和屏蔽字

开中断

执行中断服务程序

中断程序完成

关中断

恢复现场和屏蔽字

开中断

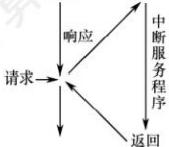
中断返回

中 断 处 理 过 程

硬件完成

图7.4 可嵌套中断的处理流程

① x86 机器保存 PC 和 PSW 到内存栈中； MIPS 机器没有PSW, 只保存PC 到特定寄存器中。



栈

主程序

|  |
| --- |
|  |
|  |
| K3+1 |
|  |
| K2+1 |
|  |
| K1+1 |

(a)单重中断

310-2025年计算机组成原理考研复习指导

中断处理流程如下：

① 关中断。

② 保 存 断 点

③ 中断服务程序寻址。

④ 保存现场和屏蔽字。进入中断服务程序后首先要保存现场和中断屏蔽字，现场信息是指 用户可见的工作寄存器的内容，它存放程序执行到断点处的现行值。



**注** **意**

现场和断点，这两类信息都不能被中断服务程序破坏。由于现场信息用指令可直接访问， 因此通常在中断服务程序中通过指令把它们保存到栈中，即由软件实现。而断点信息由 CPU 在中断响应时自动保存到栈或指定的寄存器中，即由硬件实现。

⑤ 开中断。允许更高级中断请求得到响应，以实现中断嵌套。

⑥ 执行中断服务程序。这是中断请求的目的。

⑦ 关中断。保证在恢复现场和屏蔽字时不被中断。

⑧ 恢复现场和屏蔽字。将现场和屏蔽字恢复到原来的状态。

⑨ 开中断、中断返回。中断服务程序的最后 一条指令通常是一条中断返回指令，使其返回 到原程序的断点处，以便继续执行原程序。

命题追踪 ▶ **中断隐指令的功能(2012)**

其中，①~③由中断隐指令(硬件自动)完成；④~⑨由中断服务程序完成。

命题追踪 ▶ **单级中断的处理流程(2010)**



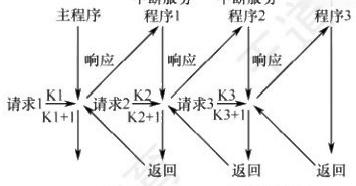
**注** **意**

若是单重中断(或称单级中断),则在上述流程中去掉⑤和⑦即可。

**3.** **多重中断和中断屏蔽技术**

命题追踪 **多重中断的中断屏蔽字相关的性质(2017、2020、2021)**

在 CPU 执行中断服务程序的过程中，若又出现了新的优先级更高的中断请求，而CPU 对新的 中断请求不予响应，则称这种中断为单 重 中断，如图7.5(a) 所示。若CPU 暂停现行的中断服务程序， 转去处理新的中断请求，则称这种中断为多重中断，也称中断嵌套，如图7.5(b) 所示。



中断服务 中断服务 中断服务

(b)多重中断

图7.5 单重中断和多重中断示意图

在图7.5(b) 中 ，CPU 在执行主程序时发生中断请求1,主程序未屏蔽任何中断，因此响应中

第7章 输入/输出系统-311

断请求1,将主程序的断点压栈保存，然后转去执行中断服务程序1。在执行中断服务程序1时， 又发生中断请求2,而其处理优先级比中断请求1的更高，此时须中止中断请求1的处理，响应 中断请求2,将中断服务程序1 的断点压栈保存，然后转去执行中断服务程序2。同样，拥有更 高处理优先级的中断请求3可以打断中断请求2的处理。当中断请求3处理完后， CPU 从栈顶取 出断点信息，回到中断服务程序2的断点 (K3+1) 处继续执行。以此类推，直到所有中断服务 程序执行完毕，最终回到原程序的断点 (K1+1) 处继续执行主程序。

CPU 要具备多重中断的功能，必须满足下列条件：

① 在中断服务程序中提前设置开中断指令。

② 优先级别高的中断源有权中断优先级别低的中断源。

中断处理优先级是指多重中断的实际优先级处理次序，可以利用中断屏蔽技术动态调整，从 而可以灵活地调整中断服务程序的优先级，使中断处理更加灵活。若不使用中断屏蔽技术，则处 理优先级和响应优先级相同。现代计算机一般使用中断屏蔽技术，每个中断源都有一个屏蔽触发 器 (MASK),1 表示屏蔽该中断源的请求，0表示可以正常请求，所有屏蔽触发器组合在一起便 构成一个屏蔽字寄存器，屏蔽字寄存器的内容称为屏蔽字。

关于中断屏蔽字的设置及多重中断程序执行的轨迹，下面通过实例说明。

**【例7.3】**设某机有4个中断源A 、B 、C 、D, 其硬件排队优先次序为A>B>C>D, 现要求

将中断处理次序改为 D>A>C>B, 写出每个中断源对应的屏蔽字。

解 ：

在中断处理次序改为D>A>C>B 后 ，D 具有最高优先级，可以屏蔽其他所有中断，且不能 中断自身，因此 D 对应的屏蔽字为1111;A 具有次高优先级，只能被D 中断，因此 A 对应的屏 蔽字为1110,以此类推，得到4个中断源的屏蔽字，见表7 . 1。

**表7.1** **中断源对应的中断屏蔽字**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中断源 | **屏蔽字** | | | |
| A | B | C | D |
| A | 1 | 1 | 1 | 0 |
| B | 0 | 1 | 0 | 0 |
| C | 0 | 1 | 1 | 0 |
| D | 1 | 1 | 1 | 1 |

从宏观上看，虽然程序中断方式克服了程序查询方式中CPU 的等待现象，提高了CPU 的 利 用率。但从微观操作分析， CPU 在处理中断时，仍需暂停原程序的运行，尤其是当高速设备频繁 成批地与主存交换信息时，需要不断打断 CPU 执行现行程序而去执行中断服务程序。

**7.3.3** **DMA** **方** **式**

DMA 方 式是一种完全由硬件进行成组信息传送的控制方式，它具有程序中断方式的优点， 即在数据准备阶段，CPU 与外设并行工作。DMA 方式在外设与内存之间开辟了一条“直接数据 通 路”,信息传送不再经过CPU, 降低了 CPU 在传送数据时的开销，因此称为直接存储器存取方 式。由于数据传送不经过CPU, 因此不需要保护、恢复CPU 现场等烦琐操作。

这种方式适用于磁盘、显卡、声卡、网卡等高速设备大批量数据的传送，它的硬件开销比较 大。在 DMA 方式中，中断的作用仅限于故障和正常传送结束时的处理。

**1.DMA 方式的特点**

主存和 DMA 接口之间有一条直接数据通路。由于DMA 方式传送数据不需要经过CPU, 因

312-2025年计算机组成原理考研复习指导

此 不 必 中 断 现 行 程 序 ，I/O 与主机并行工作，程序和传送并行工作。

DMA 方式具有下 列 特 点：

1 ) 它 使 主 存 与CPU 的固定联系脱钩，主存既可被CPU 访问，又可被外设访问。

2)在数据块传送时，主存地址的确定、传送数据的计数等都由硬件电路直接实现。

3)主存中要开辟专用缓冲区，以及时提供和接收外设的数据。

4)DMA 传 送 速 度 快 ，CPU 和外设并行工作，提高了系统效率。

5)DMA 在传送开始前要通过程序进行预处理，结束后要通过中断方式进行后处理。

**2.DMA 控制器的组成**

在 DMA 方式中，对数据传送过程进行控制的硬件称为 DMA 控 制 器 (DMA 接 口 ) 。 当 I/O 设备需要进行数据传送时，通过DMA 控 制 器 向CPU 提 出DMA 传 送 请 求 ，CPU 响 应 之 后 将 让 出

系 统 总 线 ， 由DMA 控制器接管总线进行数据传送。其主要功能如下：

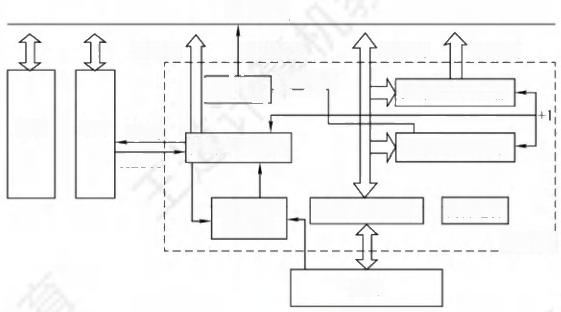
1 ) 接 受 外 设 发 出 的 DMA 请求，并向CPU 发出总线请求。

2)CPU 响应并发出总线响应信号， DMA 接管总线控制权，进入DMA 操作周期。

3)确定传送数据的主存起始地址及长度，并自动修改主存地址计数和传送长度计数。

4)规定数据在主存和外设间的传送方向，发出读/写等控制信号，执行数据传送操作。 5 ) 向CPU 报 告 DMA 操作结束。

图7 . 6给出了 一 个简单的DMA 控制器。



系统总线

数据线 地址线

主存地址计数器

CPU HRQi

传送长度计数器

HLDA

设备选择

DMAC」

设备

DMA 请求

触发器

数据缓冲寄存器

控制/状态逻辑

中断机构

中断请求

溢出信号

控制线

主存

图7.6 简单的 DMA 控制器

● 主存地址计数器：存放要交换数据的主存地址。在传送前，其保存的是传送数据的主存起

始地址；每传送 一 个字，地址寄存器的内容就加1,直至该批数据传送完毕。

● 传送长度计数器：记录传送数据的总长度。在传送前，其记录的是该批数据的总字数；每

传送 一 个字，计数器就减1,直至计数器为0,表示该批数据传送完毕。

● 数据缓冲寄存器： 暂 存 每 次 传 送 的 数 据 。 通 常 ，DMA 接口与主存之间的传送单位为字，

而 DMA 与设备之间的传送单位可能为字节或位。

● DMA请求触发器：每当I/O 设备准备好数据后，发出 一个控制信号，使 DMA 请 求 触 发 器

置位 。

· “控制/状态”逻辑：用于指定传送方向，修改传送参数，并对 DMA 请 求 信 号 、CPU 响 应

信号进行协调和同步。

● 中断机构：当 一 批数据传送完毕后触发中断机构，向 CPU 提出中断请求。

第7章 输入/输出系统-313

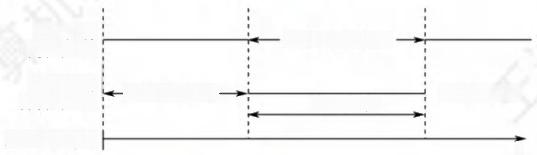
在 DMA 传送过程中， DMA 控制器接管系统总线。而当 DMA 传送结束后，将恢复CPU 的 一切权利并开始执行其操作。由此可见， DMA 控制器必须具有控制系统总线的能力。

**3.DMA 的传送方式**

主 存 和 /O 设备之间交换信息时，不通过CPU 。但 当I/O 设备和 CPU 同时访问主存时，可能 发生冲突，为了有效地使用主存， DMA 与 CPU 通常采用以下3种方式使用主存。

( 1 ) 停 止 CPU 访 存

当 I/O 设备有 DMA 请求时，由DMA 接口向 CPU 发送一个停止信号，使 CPU 放弃总线控 制权，停止访问主存，直到DMA 传送一块数据结束，如图7.7所示。数据传送结束后， DMA 接 口通知CPU 可以使用主存，并把总线控制权交回给 CPU。



CPU控制 并使用主存

DMA控制 并使用主存

主存工作时间

CPU不执行程序

—DMA不工作一

DMA不工作

DMA 工作

7

图7.7 停止CPU 访存

优点：控制简单，适用于数据传输速率很高的I/O 设备实现成组数据的传送。 缺点： DMA 在访问主存时，CPU 基本上处于不工作状态。

(2)周期挪用

▶ **周期挪用的特点及挪用次数分析(2012、2020、2022)**

由 于I/O 访存的优先级高于CPU 访 存(I/O 不立即访存就可能丢失数据),因此由I/O 设备挪 用一个存取周期，传送完一个数据字后立即释放总线，如图7.8所示。它是一种单字传送方式。

当 I/O 设备有 DMA 请求时，会遇到3种情况：①此时 CPU 不在访存，因此 I/O 的访存请求与 CPU 未发生冲突；②CPU 正在访存，此时必须待存取周期结束后，CPU 再将总线占有权让出； ③ I/O 和 CPU 同时请求访存，出现访存冲突，此时 CPU 要暂时放弃总线占有权。

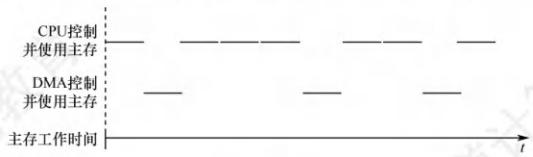


图7.8 周期挪用

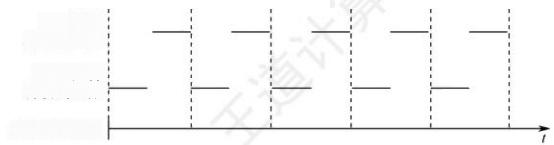
优 点 ： 既 实 现 了 /O传送，又较好地发挥了主存与CPU 的效率。

缺点：每挪用 一个主存周期，DMA 接口都要申请、建立和归还总线控制权。

(3 ) DMA 与 CPU 交替访存

将 CPU 的工作周期分成两个时间片， 一个给 CPU 访存，另一个给 DMA 访存，这样在每个 CPU 周 期 内 ，CPU 和 DMA 就都可以轮流访存，如图7.9所示。这种方式适用于CPU 的工作周期 比主存存取周期长的情况。例如，若 CPU 的工作周期是1.2μs, 主存的存取周期小于0.6μs, 则可 将 一个 CPU 周期分为C₁ 和 C₂ 两个周期，其中 C₁ 专 供DMA 访 存 ，C₂ 专 供CPU 访 存。这种方式 不需要申请、建立和归还总线使用权，总线使用权是通过C₁ 和 C₂ 分时控制的。

314 — 2025年计算机组成原理考研复习指导



CPU 控 制

并使用主存

DMA 控 制

并使用主存

主存工作时间

图 7 . 9 DMA 与 CPU 交 替 访 存

优点：不需要总线控制权的申请、建立和归还过程，具有很高的传送速率。 缺点：相应的硬件逻辑变得更复杂。

**命题追踪** **DMA 方** **式** **的** **效** **率** **分** **析** **及** **相** **关** **计** **算** **(** **2** **0** **1** **1** **、** **2** **0** **1** **8** **)**

**【例7.4】**假定计算机的主频为500MHz,CPI 为4,某外设的数据率为40MB/s,/O 接口中 的数据端口为32位，采用DMA 方式，每次DMA 传送块大小为1000B, 且 DMA 预处理和后处 理的总时钟周期数为500,则CPU 用于该外设I/O 的时间占CPU 总时间的百分比是多少?

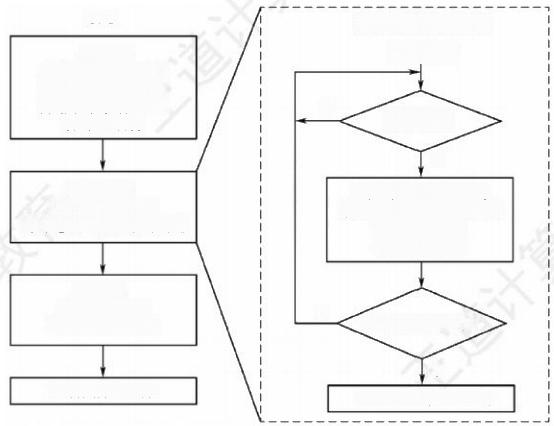
解 ：

外设每秒的 DMA 次数为40MB/s÷1000B=40000, 在 DMA 方式中，只有预处理和后处理 需要CPU 处理，数据传送全程由DMA 控制。CPU 用于外设I/O 的总时间为40000×500=2×10? 个时钟周期，占CPU 总时间的百分比为2×10⁷ ÷500M=4%。

**4.DMA 的** **传** **送** **过** **程**

▶ **DM** **A** **方式的传送过程(2019)**

图7.10所示为 DMA 的数据传送流程，分为预处理、数据传送和后处理3个阶段。



数据传送阶段的细化

DMA 请求

允许传送?

Y

主存起始地址送总线

数 据 送 /O设备(或主存)

修改主存地址

修改字计数器

数据块传送结束?

Y

向CPU申请程序中断

CPU

预处理：

主存起始地址 →AR I/O 设备地址 →DAR 传送数据个数 →WC 启动I/O设备

数据传送：

继续执行主程序

同时完成一批数据的传送

后处理：

中断服务程序

做DMA结束处理

继续执行主程序

N

N

图7 . 10 DMA 的 传 送 流 程

(1)预处理

由 CPU 完成一些必要的准备工作。首先，初始化 DMA 控制器中的有关寄存器、设置传送方 向、测试并启动设备等。然后， CPU 继续执行原程序，直到I/O 设备准备好发送的数据(输入情 况)或接收的数据(输出情况)时， I/O 设备向 DMA 控制器发送 DMA 请求，再由 DMA 控制器

第7章 输入/输出系统- 315

向CPU 发出总线请求(这两个过程也可统称 DMA 请求),用以传输数据。

(2)数据传送

DMA 以数据块为基本传送单位。DMA 占用总线后的数据输入/输出操作都是通过循环来实 现的，这一循环也是由DMA 控制器实现的，即数据传送阶段完全由DMA (硬件)控制。

(3)后处理

DMA 控制器向CPU 发送中断请求， CPU 执行中断服务程序做 DMA 结束处理，包括校验 数据(出错则转诊断程序)等后处理工作。

在 DMA 方式下，整个数据块的传送过程都不需要 CPU 参 与 ，CPU 只在最初的 DMA 控 制器初始化和最后的 DMA 结束处理时才介入，因此CPU 用于 I/O 的开销非常小。

**5.DMA 方式和中断方式的区别**

命 题 追 踪▶ **中断方式与** **DMA** **方式的区别(2013、2023)**

DMA 方式和中断方式的重要区别如下：

① 中断方式是程序的切换，需要保护和恢复现场；而DMA 方式不中断现行程序，无需保护 现场，除了预处理和后处理，其他时候不占用任何CPU 资源。

② 对中断请求的响应只能发生在每条指令执行结束时(执行周期后);而对DMA 请求的响 应可以发生在任意一个机器周期结束时(取指、间址、执行周期后均可)。

③ 中断传送过程需要CPU 的干预；而 DMA 传送过程不需要CPU 的干预，因此数据传输速 率非常高，适合于高速外设的成组数据传送。

命题追踪 ▶ **DM** **A** **与CP** **U** **请求总线的优先级对比(2012、2022)**

④ DMA 请求的优先级高于中断请求。

⑤ 中断方式具有处理异常事件的能力，而 DMA 方式仅局限于大批数据的传送。 ⑥ 从数据传送来看，中断方式靠程序传送，DMA 方式靠硬件传送。

**7.3.4** **本** **节** **习** **题** **精** **选**

**一** **、单项选择题**

**01.** 设置中断排队判优逻辑的目的是( )。

A. 产生中断源编码

B. 使同时提出的请求中的优先级别最高者得到及时响应

C. 使 CPU 能方便地转入中断服务子程序

D. 提高中断响应速度

**02.** 以下说法中，错误的是()。

A. 中断服务程序一般是操作系统模块

B. 中断向量方法可提高中断源的识别速度

C. 中断向量地址是中断服务程序的入口地址

D. 重叠处理中断的现象称为中断嵌套

**03.** 当有中断源发出请求时，CPU 可执行相应的中断服务程序，可以提出中断(包括内中断 和外中断)的有( )。

I. 外部事件 I.Cache I. 虚拟存储器失效

IV. 浮点数运算下溢 V. 浮点数运算上溢

316-2025年计算机组成原理考研复习指导

A.I 、Ⅲ和IV B.I和V C.I 、Ⅱ 和 V D.I 、Ⅲ 和 V

**04.** 关于程序中断方式和DMA 方式的叙述，错误的是()。

I.DMA 的优先级比程序中断的优先级要高

II. 程序中断方式需要保护现场， DMA 方式在传输过程中不需要保护现场

ⅢI. 程序中断方式的中断请求是为了报告CPU 数据的传输结束，而 DMA 方式的中断请 求完全是为了传送数据

A. 仅 Ⅱ B.Ⅱ 、Ⅲ C. 仅 Ⅲ D.I 、 Ⅲ

**05.** 下列说法中，错误的是()。

I. 程序中断过程是由硬件和中断服务程序共同完成的

I. 在每条指令的执行过程中，每个总线周期要检查一次有无中断请求 Ⅲ .检测有无 DMA 请求， 一般安排在一条指令执行过程的末尾

IV. 中断服务程序的最后指令是无条件转移指令

A.II 、IV B.Ⅱ 、I 、IV C.I 、IV D.I、Ⅱ、Ⅲ、IV

**06.** 能产生 DMA 请求的总线部件是()。

I. 高速外设 IⅡ. 需要与主机批量交换数据的外设

Ⅲ . 具有 DMA 接口的设备

A. 仅I B. 仅 Ⅲ C.1 、II D.I 、Ⅲ

07. 在具有中断向量表的计算机中，中断向量地址是()。

A. 子程序入口地址 B. 中断服务程序的入口地址

C. 中断服务程序入口地址的地址 D. 中断程序断点

**08.** 中断响应是在( )。

A. 一条指令执行开始 B. 一条指令执行中间

C. 一条指令执行之末 D. 一条指令执行的任何时刻

**09.** 在下列情况下，可能不发生中断请求的是( )。

A.DMA 操作结束 B. 一条指令执行完毕

C. 机器出现故障 D. 执行“软中断”指令

**10.** 在配有通道的计算机系统中，用户程序需要输入/输出时，引起的中断是()。

A. 访管中断 B.I/O 中断 C. 故障 D. 外中断

**11.** 某计算机有4级中断，优先级从高到低为1→2→3→4。若将优先级顺序修改，改后1级 中断的屏蔽字为1101,2级中断的屏蔽字为0100,3级中断的屏蔽字为1111,4级中断 的屏蔽字为0101,则修改后的优先顺序从高到低为( )。

A.1→2→3→4 B.3→ 1→4→2 C.1→3→4→2 D.2→ 1→3→4

**12.** 下列不属于程序控制指令的是( )。

A. 无条件转移指令 B. 有条件转移指令

C. 中断隐指令 D. 循环指令

**13.** 在中断响应周期中， CPU 主要完成的工作是( )。

A. 关中断，保护断点，发中断响应信号并形成向量地址

B. 开中断，保护断点，发中断响应信号并形成向量地址

C. 关中断，执行中断服务程序

D. 开中断，执行中断服务程序

**14.** 下列关于中断 I/O 方式的叙述中，错误的是()。

第7章 输入/输出系统-317

A.CPU 对外部中断的响应不可能发生在一条指令的执行过程中

B. 在中断I/O 方式下，外设接口中的寄存器和CPU 中的寄存器直接交换数据 C. 中断请求的是CPU 时间，要求 CPU 执行程序来处理发生的相关事件

D. 只要有中断请求发生， 一条指令执行结束后CPU 就进入中断响应周期

**15.** 当 CPU 响应中断时，进入“中断响应周期”,采用硬件方法保护并更新程序计数器(PC) 内容，而不是由软件完成的，主要是为了( )。

A. 能进入中断处理程序，并能正确返回源程序

B. 节省主存空间

C. 提高处理机速度

D. 易于编制中断处理程序

**16.** 在 I/O 接口中设置中断触发器保存外设发出的中断请求，是因为( )。 A. 中断不需要立即处理

B. 中断设备的处理速度比 CPU 快

C.CPU 无法对发生的中断请求立即进行处理

D. 可能有多个中断同时发生

**17** . 在中断响应周期中，由()将允许中断触发器置0。

A. 关中断指令 B. 中断隐指令 C. 开中断指令 D. 中断服务程序

**18.** CPU 响应中断时最先完成的步骤是( )。

A. 开中断 B. 保存断点 C. 关中断 D. 转入中断服务程序

**19.** 设置中断屏蔽标志可以改变( )。

A. 多个中断源的中断请求优先级 B.CPU 对多个中断请求响应的优先次序

C. 多个中断服务程序开始执行的顺序 D. 多个中断服务程序执行完的次序 **20.** 在 CPU 响应中断时，保护两个关键的硬件状态是( )。

A.PC 和 IR B.PC 和 PSW C.AR 和 IR D.AR 和 PSW

**21.** 在各种I/O 方式中，中断方式的特点是( ), DMA 方式的特点是( )。 A.CPU 与外设串行工作，传送与主程序串行工作

B.CPU 与外设并行工作，传送与主程序串行工作

C.CPU 与外设串行工作，传送与主程序并行工作

D.CPU 与外设并行工作，传送与主程序并行工作

**22.** 下列关于程序查询方式及其工作过程的叙述中，正确的是( )。

A. 按启动查询方式的不同，可分为软件查询方式和硬件查询方式

B.CPU 主要负责启动外设和查询其状态，不参与数据传送

C. 每完成一次数据传送后，会修改主存地址和计数值

D.CPU 需一直查询外设的状态，直到外设准备就绪时才可去执行其他程序

**23.** 在 DMA 传送方式中，由( )发出 DMA 请求，在传送期间总线控制权由( )掌握。

A. 外部设备、CPU B.DMA 控制器、DMA 控制器

C. 外部设备、DMA 控制器 D.DMA 控制器、内存

**24.** 下列叙述中，( )是正确的。

A. 程序中断方式和 DMA 方式中实现数据传送都需要中断请求

B. 程序中断方式中有中断请求，DMA 方式中没有中断请求

C. 程序中断方式和 DMA方式中都有中断请求，但目的不同

318-2025年计算机组成原理考研复习指导

D.DMA 要等指令周期结束时才可以进行周期窃取

**25.** 以下关于 DMA 方式进行I/O 的描述中，正确的是()。

A. 一个完整的 DMA 过程，部分由DMA 控制器控制，部分由CPU 控制 B. 一个完整的DMA 过程，完全由CPU 控 制

C. 一个完整的DMA 过程，完全由DMA 控制器控制， CPU 不介入任何控制

D. 一个完整的 DMA 过程，完全由CPU 采用周期挪用法控制

**26.** CPU 响应 DMA 请求的条件是当前( )执行完。

A. 机器周期 B. 总线周期

C. 机器周期和总线周期 D. 指令周期

**27.** 当某五级流水线CPU 正在执行某条指令的第二级流水段时，外部设备产生了一个DMA

请求，则CPU 对该 DMA 请求响应的时机是( )。

A. 立即响应

B. 在该指令的第二级流水段执行完毕后响应

C. 在该指令的第三级流水段执行完毕后响应

D. 在该指令执行结束后响应

**28.** 关于外中断(故障除外)和 DMA, 下列说法中正确的是( )。

A.DMA 请求和中断请求同时发生时，响应 DMA 请求

B.DMA 请求、非屏蔽中断、可屏蔽中断都要在当前指令结束之后才能被响应 C. 非屏蔽中断请求优先级最高，可屏蔽中断请求优先级最低

D. 若不开中断，所有中断请求就不能响应

**29.** 以下有关 DMA 方式的叙述中，错误的是()。

A. 在 DMA 方式下， DMA 控制器向 CPU 请求的是总线使用权

B.DMA 方式可用于键盘和鼠标的数据输入

C. 在数据传输阶段，不需要CPU 介入，完全由DMA 控制器控制

D.DMA 方式要用到中断处理

**30.** 在主机和外设的信息传送中，( )不是一种程序控制方式。

A. 直接程序传送 B. 程序中断

C. 直接存储器存取 (DMA) D. 通道控制

**31.** 中断发生时，程序计数器内容的保护和更新是由( )完成的。

A. 硬件自动 B. 进栈指令和转移指令

C. 访存指令 D. 中断服务程序

**32.** 在 DMA 方式传送数据的过程中，因为没有破坏( )的内容，所以 CPU 可以正常工作 (访存除外)。

A. 程序计数器 B. 程序计数器和寄存器

C. 指令寄存器 D. 堆栈寄存器

**33.** 在 DMA 方式下，数据从内存传送到外设经过的路径是()。

A. 内存→数据总线→数据通路→外设 B. 内存→数据总线→DMAC→外设

C. 内存→数据通路→数据总线→外设 D. 内存 →CPU→ 外设

**34.** 采用周期挪用进行 DMA 数据传送时，每传送一个数据要占用一个( )的时间。

A. 指令周期 B. 机器周期 C. 时钟周期 D. 存取周期

**35.** 启动一次 DMA 传送，外设和主机之间将完成一个()的数据传送。

第7章 输入/输出系统-319

A. 字节 B . 字 C. 总线宽度 D. 数据块

**36.** 在磁盘存储器进行读/写操作之前， CPU 需要对磁盘控制器或 DMA 控制器进行初始化。 在下列选项中，不包含在初始化信息中的是( )。

A. 传送信息所在的主存起始地址 B. 传送方向(是读磁盘还是写磁盘)

C. 传送信息所在的通用寄存器编号 D. 传送数据的字数或字节数

**37.** 【2009统考真题】下列选项中，能引起外部中断的事件是( )。

A. 键盘输入 B. 除数为0 C. 浮点运算下溢 D. 访存缺页

**38.【** 2010统考真题】单级中断系统中，中断服务程序内的执行顺序是()。

1. 保护现场 II. 开中断 Ⅲ . 关 中 断 IV. 保存断点 V. 中断事件处理 VI. 恢复现场 VII. 中断返回

A.I→V→VI→I→VII B.Ⅲ→I→V→VI

C.Ⅲ→IV→V→VI→VH D.IV→I→V→VI→VI

**39.【2**011统考真题】某计算机有五级中断L₄~Lo, 中断屏蔽字为M₁M₃M₂M₁M₀,M,=1(O≤

i≤4) 表示对 L,级中断进行屏蔽。若中断响应优先级从高到低的顺序是 Lo→L₁→L₂→ L₃→L₄, 且要求中断处理优先级从高到低的顺序为L₄→L₀→L₂→L₁→L₃, 则 L₁ 的中断处 理程序中设置的中断屏蔽字是( )。

A.11110 B.01101 C.00011 D.01010

**40.【2**011统考真题】某计算机处理器主频为50MHz, 采用定时查询方式控制设备 A 的 I/O, 查询程序运行一次所用的时钟周期数至少为500。在设备A 工作期间，为保证数据不丢 失，每秒需对其查询至少200次，则CPU 用于设备A 的 I/O 的时间占整个CPU 时间的 百分比至少是( )。

A.0.02% B.0.05% C.0.20% D.0.50%

**41.**【2012统考真题】响应外部中断的过程中，中断隐指令完成的操作，除保护断点外，还 包括( )。

1. 关中断 II. 保存通用寄存器的内容

ⅢI. 形成中断服务程序入口地址并送PC

A. 仅 I、Ⅱ B. 仅 I 、Ⅲ C. 仅Ⅱ、Ⅲ D.I 、Ⅱ 、Ⅲ

**42.** 【2013统考真题】下列关于中断I/O 方式和DMA 方式比较的叙述中，错误的是()。 A. 中断 I/O 方式请求的是 CPU 处理时间，DMA 方式请求的是总线使用权

B. 中断响应发生在一条指令执行结束后， DMA 响应发生在一个总线事务完成后 C. 中 断 I/O 方式下数据传送通过软件完成，DMA 方式下数据传送由硬件完成

D. 中 断I/O 方式适用于所有外部设备， DMA 方式仅适用于快速外部设备

**43.【**2014统考真题】若某设备中断请求的响应和处理时间为100ns, 每400ns 发出一次中断 请求，中断响应所允许的最长延迟时间为50ns, 则在该设备持续工作过程中， CPU 用于 该设备的I/O 时间占整个 CPU 时间的百分比至少是( )。

A.12.5% B.25% C.37.5% D.50%

**44.** 【2015统考真题】在采用中断I/O 方式控制打印输出的情况下， CPU 和打印控制接口中

的I/O 端口之间交换的信息不可能是()。

A. 打印字符 B. 主存地址 C. 设备状态 D. 控制命令

**45.** 【2017统考真题】下列关于多重中断系统的叙述中，错误的是( )。

A. 在一条指令执行结束时响应中断

320-2025年计算机组成原理考研复习指导

B. 中断处理期间 CPU 处于关中断状态

C. 中断请求的产生与当前指令的执行无关

D.CPU 通过采样中断请求信号检测中断请求

**46.** 【2018统考真题】下列关于外部I/O 中断的叙述中，正确的是( )。

A. 中断控制器按所接收中断请求的先后次序进行中断优先级排队

B.CPU 响应中断时，通过执行中断隐指令完成通用寄存器的保护

C.CPU 只有在处于中断允许状态时，才能响应外部设备的中断请求

D. 有中断请求时，CPU 立即暂停当前指令执行，转去执行中断服务程序

**47.**【2019统考真题】某设备以中断方式与CPU 进行数据交换，CPU 主 频 为 1GHz, 设备接 口中的数据缓冲寄存器为32位，设备的数据传输速率为50kB/s 。 若每次中断开销(包 括中断响应和中断处理)为1000个时钟周期，则CPU 用于该设备输入/输出的时间占整 个 CPU 时间的百分比最多是( )。

A.1.25% B.2.5% C.5% D.12.5%

**48.【2** 019统考真题】下列关于 DMA 方式的叙述中，正确的是( )。

I.DMA 传送前由设备驱动程序设置传送参数

II. 数据传送前由 DMA 控制器请求总线使用权

II. 数据传送由DMA 控制器直接控制总线完成

IV.DMA 传送结束后的处理由中断服务程序完成

A. 仅 I 、Ⅱ B. 仅 I 、Ⅲ 、IV C. 仅Ⅱ、Ⅲ、IV D.I 、I 、Ⅲ 、IV

**49.【2** 020统考真题】下列事件中，属于外部中断事件的是()。

I. 访存时缺页 II. 定时器到时 Ⅲ . 网络数据包到达

A. 仅 I 、Ⅱ B. 仅1、Ⅲ C. 仅Ⅱ、Ⅲ D.I 、Ⅱ 和 Ⅲ

**50.【**2020统考真题】外部中断包括不可屏蔽中断 (NMI) 和可屏蔽中断，下列关于外部中

断的叙述中，错误的是()。

A.CPU 处于关中断状态时，也能响应NMI 请求

B. 一旦可屏蔽中断请求信号有效，CPU 就立即响应

C. 不可屏蔽中断的优先级比可屏蔽中断的优先级高

D. 可通过中断屏蔽字改变可屏蔽中断的处理优先级

**51.【**2020 统考真题】若设备采用周期挪用 DMA 方式进行输入和输出，每次 DMA 传送的 数据块大小为512字节，相应的I/O 接口中有一个32位数据缓冲寄存器。对于数据输入 过程，下列叙述中，错误的是( )。

A. 每准备好32位数据， DMA 控制器就发出一次总线请求

B. 相对于CPU,DMA 控制器的总线使用权的优先级更高

C. 在整个数据块的传送过程中，CPU 不可以访问主存储器

D. 数据块传送结束时，会产生 “DMA 传送结束”中断请求

**52.【**2021统考真题】下列是关于多重中断系统中CPU 响应中断的叙述，错误的是( )。

A. 仅在用户态(执行用户程序)下， CPU 才能检测和响应中断

B.CPU 只有在检测到中断请求信号后，才会进入中断响应周期

C. 进入中断响应周期时， CPU 一 定处于中断允许(开中断)状态

D. 若 CPU 检测到中断请求信号，则一定存在未被屏蔽的中断源请求信号 **53.【**2022统考真题】下列关于中断I/O 方式的叙述中，不正确的是()。

第7章 输入/输出系统 321

A. 适用于键盘、针式打印机等字符型设备

B. 外设和主机之间的数据传送通过软件完成

C. 外设准备数据的时间应小于中断处理时间

D. 外设为某进程准备数据时 CPU 可运行其他进程

**54.**【2023统考真题】下列关于硬件和异常/中断关系的叙述中，错误的是( )。 A.CPU 在执行一条指令的过程中检测异常事件

B.CPU 在执行完一条指令时检测中断请求信号

C. 开中断时 CPU 检测到中断请求后就进行中断响应

D. 外部设备通过中断控制器向 CPU 发中断结束信号

**55.**【2023统考真题】下列关于 I/O 控制方式的叙述中，错误的是( )。

A. 查询方式下，通过CPU 执行查询程序进行I/O 操作

B. 中断方式下，通过CPU 执行中断服务程序进行I/O 操作

C.DMA 方式下，通过CPU 执 行DMA 传送程序进行I/O 操作

D. 对于 SSD 、网络适配器等高速设备，采用 DMA 方式输入/输出

**二、综合应用题**

**01.** 在 DMA 方式下，主存和I/O 设备之间有一条物理通路相连吗?

**02.** 假定某I/O 设备向CPU 传送信息的最高频率为4万次/秒，而相应中断处理程序的执行

时间为40μs, 则该I/O 设备是否可采用中断方式工作?为什么?

**03.** 在程序查询方式的输入/输出系统中，假设不考虑处理时间，每个查询操作需要100个时钟 周期，CPU 的时钟频率为50MHz。现有鼠标和硬盘两个设备，而且 CPU 必须每秒对鼠标 进行30次查询，硬盘以32位字长为单位传输数据，即每32位被 CPU 查询一次，传输速 率为2×2°B/s。求 CPU 对这两个设备查询所花费的时间比率，由此可得出什么结论?

**04.** 设某计算机有4个中断源1、2、3、4,其硬件排队优先次序按1→2→3→4降序排列， 各中断源的服务程序中所对应的屏蔽字如下表所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中断源 | **屏蔽字** | | | |
| 1 | 2 | 3 | 4 |
| 1 | 1 | 1 | 0 | 1 |
| 2 | 0 |  | 0 | 0 |
| 3 | 1 |  | 1 | 1 |
| 4 | 0 | 1 | 0 | 1 |

1)给出上述4个中断源的中断处理次序。

2)若4个中断源同时有中断请求，画出 CPU 执行程序的轨迹。

**05.** 一个 DMA 接口可采用周期窃取方式把字符(字节)传送到存储器，它支持的最大批量为 400B。若存取周期为0.2μs,每处理一次中断需5μs,现有的字符设备的传输速率为9600b/s。 假设字符之间的传输是无间隙的，试问DMA 方式每秒因数据传输占用处理器多少时间? 若完全采用中断方式，又需占处理器多少时间(忽略预处理所需时间)?

**06.** 假设磁盘传输数据以32位的字为单位，传输速率为1MB/s,CPU 的时钟频率为50MHz。

回答以下问题：

1)采取程序查询方式，假设查询操作需要100个时钟周期，求 CPU 为I/O 查询所花费 的时间比率(假设进行足够的查询以避免数据丢失)。

322-2025年计算机组成原理考研复习指导

2)采用中断方式进行控制，每次传输的开销(包括中断处理)为80 个时钟周期。求 CPU 为传输硬盘所花费的时间比率。

3 ) 采 用 DMA 的方式，假定 DMA 的启动需要1000个时钟周期， DMA 完成时后处理需 要500个时钟周期。若平均传输的数据长度为4KB ( 此 处K=1000), 试问硬盘工作

时 CPU 将用多少时间比率进行输入/输出操作?忽略 DMA 申请总线的影响。

**07.** 【2009统考真题】某计算机的CPU 主频为500MHz,CPI 为5(即执行每条指令平均需5个 时钟周期)。假定某外设的数据传输速率为0.5MB/s, 采用中断方式与主机进行数据传送， 以32位为传输单位，对应的中断服务程序包含18条指令，中断服务的其他开销相当于 2条指令的执行时间。回答下列问题，要求给出计算过程。

1)在中断方式下， CPU 用于该外设I/O 的时间占整个CPU 时间的百分比是多少?

2)当该外设的数据传输速率达到5MB/s 时，改用 DMA 方式传送数据。假定每次 DMA 传送块大小为5000B, 且 DMA 预处理和后处理的总开销为500个时钟周期，则 CPU 用于该外设I/O 的时间占整个CPU 时间的百分比是多少(假设 DMA 与 CPU 之间没 有访存冲突)?

**08.**【2012统考真题】假定某计算机的CPU 主频为80MHz,CPI 为4,平均每条指令访存1 .5 次，主存与 Cache 之间交换的块大小为16B,Cache 的命中率为99%,存储器总线宽带 为32位。回答下列问题。

1)该计算机的 MIPS 数是多少?平均每秒 Cache 缺失的次数是多少?在不考虑 DMA 传 送的情况下，主存带宽至少达到多少才能满足CPU 的访存要求?

2 ) 假 定 在 Cache 缺失的情况下访问主存时，存在0.0005%的缺页率，则 CPU 平均每秒 产生多少次缺页异常?若页面大小为4KB, 每次缺页都需要访问磁盘，访问磁盘时 DMA 传送采用周期挪用方式，磁盘I/O 接口的数据缓冲寄存器为32位，则磁盘I/O 接口平均每秒发出的 DMA 请求次数至少是多少?

3)CPU 和 DMA 控制器同时要求使用存储器总线时，哪个优先级更高?为什么?

4)为了提高性能，主存采用4体低位交叉存储模式，工作时每1/4个存储周期启动一个

体。若每个体的存储周期为50ns, 则该主存能提供的最大带宽是多少?

**09.【**2016统考真题】假定CPU 主 频 为 5 0MHz,CPI 为4。设备 D 采用异步串行通信方式向 主 机 传 送 7 位 ASCⅡ 码字符，通信规程中有1位奇校验位和1位停止位，从 D 接 收 启 动命令到字符送入I/O 端口需要0.5ms。 回答下列问题，要求说明理由。

1)每传送一个字符，在异步串行通信线上共需传输多少位?在设备D 持续工作过程中， 每秒最多可向I/O 端口送入多少个字符?

2 ) 设 备 D 采用中断方式进行输入/输出，示意图如下所示：



I/O 端口每收到一个字符申请一次中断，中断响应需10个时钟周期，中断服务程序共有 20条指令，其中第15条指令启动D 工 作。若 CPU 需 从 D 读取1000个字符，则完成这 一任务所需时间大约是多少个时钟周期? CPU 用于完成这一任务的时间大约是多少个 时钟周期?在中断响应阶段 CPU 进行了哪些操作?

第7章 输入/输出系统- 323

**10.【**2018统考真题】假定计算机的主频为500MHz,CPI 为4。现有设备A 和 B, 其数据传 输速率分别为2MB/s 和 4 0MB/s, 对应I/O 接口中各有一个32位数据缓冲寄存器。回答 下列问题，要求给出计算过程。

1)若设备 A 采用定时查询/O 方式，每次输入/输出都至少执行10条指令。设备 A 最 多间隔多长时间查询一次才能不丢失数据? CPU 用于设备 A 输入/输出的时间占 CPU 总时间的百分比至少是多少?

2)在中断 VO 方式下，若每次中断响应和中断处理的总时钟周期数至少为400,则设备 B 能否采用中断 I/O 方式?为什么?

3)若设备B 采用 DMA 方式，每次 DMA 传送的数据块大小为1000B,CPU 用于 DMA 预处理和后处理的总时钟周期数为500,则CPU 用于设备B 输入/输出的时间占CPU 总时间的百分比最多是多少?

**11.**【2022统考真题】假设某磁盘驱动器中有4个双面盘片，每个盘面有20000个磁道，每 个磁道有500个扇区，每个扇区可记录512字节的数据，盘片转速为7200rpm (转/分), 平均寻道时间为5ms 。请回答下列问题。

1)每个扇区包含数据及其地址信息，地址信息分为3个字段。这3个字段的名称各是

什么?对于该磁盘，各字段至少占多少位?

2)一个扇区的平均访问时间约为多少?

3)若采用周期挪用 DMA 方式进行磁盘与主机之间的数据传送，磁盘控制器中的数据缓 冲区大小为64位，则在一个扇区的读/写过程中， DMA 控制器向 CPU 发送了多少 次总线请求?若 CPU 检测到 DMA 控制器的总线请求信号时也需要访问主存，则 DMA 控制器是否可以获得总线使用权?为什么?

**7.3.5** **答案与解析**

**一** **、单项选择题**

**01.B**

当有多个中断请求同时出现时，中断服务系统必须能从中选出当前最需要给予响应的且最重 要的中断请求，这就需要预先对所有的中断进行优先级排队，这个工作可由中断判优逻辑来完成， 排队的规则可由软件通过对中断屏蔽寄存器进行设置来确定。

**02.C**

中断服务程序是处理器处理的紧急事件，可理解为一种服务，是事先编好的某些特定的程序， 一般属于操作系统的模块，以供调用执行，A 正确。中断向量由向量地址形成部件，即由硬件产 生，并且不同的中断源对应不同的中断服务程序，因此通过该方法，可以较快速地识别中断源， B 正确。中断向量是中断服务程序的入口地址，中断向量地址是内存中存放中断向量的地址，即 中断服务程序入口地址的地址，C 错误。重叠处理中断的现象称为中断嵌套， D 正确。

**03.D**

外部事件如按 Esc 键以退出运行的程序等，属于外中断， I 正确。Cache 完全是由硬件实现的， 不会涉及中断层面，Ⅱ错误。虚拟存储器失效如缺页等，会发出缺页中断，属于内中断，Ⅲ正确。 浮点数运算下溢，直接当作机器零处理，而不会引发中断，IV 错误。浮点数上溢，表示超过了浮 点数的表示范围，属于内中断， V 正确。

04.C

DMA 方式不需要CPU 干预传送操作，仅在开始和结尾借用CPU 一点时间，其余不占用CPU 任何资源；中断方式是程序切换，每次操作需要保护和恢复现场，所以 DMA 优先级高于中断请

324-2025年计算机组成原理考研复习指导

求，从而可以加快处理效率， I 正确。从I 的分析可知，程序中断需要中断现行程序，因此需保护 现场，以便中断执行完后还能回到原来的点去继续没有完成的工作；DMA 方式不需要中断现行 程 序 ，CPU 仅仅做一些辅助性工作，因为主存和 DMA 接口之间有一条数据通路，所以无须使用 CPU 内部寄存器，也就无须保护现场，Ⅱ正确。Ⅲ的说法正好相反。



**注** **意**

程序中断的保护现场是由中断服务子程序完成的，不同中断源对应的中断子程序是不同 的，可以理解为因 DMA 方式无须使用 CPU 内部寄存器，所以其对应的中断服务子程序也无须 保存CPU 现场。此外，“DMA 方式无须保护现场”是唐朔飞老师所撰教材中的原话。

**05.B**

程序中断过程是由硬件(称中断隐指令)和中断服务程序共同完成的， I 正确。在每条指令 执行结束时(而不是执行过程中),CPU 统一扫描各个中断源，检查有无中断请求，Ⅱ错误。CPU 会在每个存储周期 ( 总线周期)结束后检查是否有 DMA 请求，而不是在一条指令执行过程的末 尾，Ⅲ错误。中断服务程序的最后指令通常是中断返回指令，与无条件转移指令不同的是，它不 仅要修改 PC 值，而且要将 CPU 中的所有寄存器都恢复到中断前的状态， IV 错 误。

**06.B**

只有具有 DMA 接口的设备才能产生 DMA 请求，即使当前设备是高速设备或需要与主机批 量交换数据，若没有 DMA 接口的话，也不能产生 DMA 请求。

07.C

中断向量地址是中断向量表的地址，因为中断向量表保存着中断服务程序的入口地址，所以 中断向量地址是中断服务程序入口地址的地址。

**08.C**

CPU 响应中断必须满足下列3个条件：①CPU 接收到中断请求信号。首先中断源要发出中 断请求，同时 CPU 还要收到这个中断请求信号。②CPU 允许中断，即开中断。③一条指令执行 完毕。因此中断响应是在指令执行末尾， C 正确。

**09.B**

DMA 操作结束、机器出现故障、执行“软中断”指令时都会产生中断请求。一条指令执行 完毕可能响应中断请求，但它本身不会引起中断请求。

**10.A**

用户程序需要输入/输出时，需要调用操作系统提供的接口(请求操作系统服务),此时会使 得访管中断，系统由用户态转为核心态。

11.B

屏蔽字“1”表示不可被中断，“0”表示可被中断。由3级中断的屏蔽字可知，它屏蔽所有 中断，优先级最高；再由1级中断的屏蔽字可知，它屏蔽除3外的所有中断，优先级次之；以此 类推，可知选B。

【另解】“1”越多表示优先级越高，因此屏蔽其他中断源就越多。

12.C

中断隐指令并不是一条由程序员安排的真正的指令，因此不可能把它预先编入程序中，只能在 响应中断时由硬件直接控制执行。中断隐指令不在指令系统中，因此不属于程序控制指令。

13.A

在中断响应周期，CPU 主要完成关中断、保护断点、发中断响应信号并形成中断向量地址的

第7章 输入/输出系统 325

工作，即执行中断隐指令。

**14.D**

CPU 总是在一条指令结束时检查外中断请求，因此对外中断的响应只可能发生在一条指令结 束时。中断 I/O 方 式 下 ，CPU 执行中断服务程序时会执行相应的I/O 指令，实现CPU 的通用寄存 器和外设接口中的寄存器之间的直接数据交换。中断请求就是要求CPU 执行程序来处理发生的相 关事件。选项 D 在下列两种情况下错误：①关中断时，CPU 检测不到中断请求，因此不会进入中 断响应周期；②当有中断请求的请求源被中断屏蔽字屏蔽时，也不会进入中断响应周期。

15.A

在中断响应周期中，采用硬件方法保护并更新 PC 内容，而不由软件完成，这样可以避免因 为软件保存和恢复PC 内容而造成的时间开销和错误风险，提高中断处理的效率和正确性。

16.C

因 为 CPU 无法对发生的中断请求立即进行处理，因此需要在I/O 接口中设置中断触发器，以 保存是哪些外设发出了中断请求，等CPU 当前的指令周期结束后，响应中断并进行处理。

17.B

允许中断触发器置0表示关中断，在中断响应周期由硬件自动完成，即中断隐指令完成。虽 然关中断指令也能实现关中断的功能，但在中断响应周期，关中断是由中断隐指令完成的。在恢 复现场和屏蔽字的时候，也需要关中断的操作，此时是由关中断指令来完成的。

18.C

只有先关中断，才可以保护断点。若先不保护断点，则可能会丢失当前程序的断点。同理， 在恢复现场前也要关中断。这个过程和操作系统中的信号量 PV 操作类似，都是将内部过程变为 不可打断的原子操作。

**19.D**

中断优先级包括响应优先级和处理优先级，中断屏蔽标志改变的是处理优先级。中断响应优 先级是由中断查询程序或中断判优电路决定的，它反映的是多个中断同时请求时哪个先被响应， 即中断服务程序开始执行的顺序。在多重中断系统中，中断处理优先级决定了本中断是否能打断 正在执行的中断服务程序，决定了多个中断服务程序执行完的次序。

**20.B**

PC 的内容是被中断程序尚未执行的第一条指令地址， PSW 寄存器保存各种状态信息。CPU 响应中断后，需要保护中断的CPU 现场，将PC 和 PSW 压入堆栈，这样等到中断结束后，就可 以将压入堆栈的原PC 和 PSW 的内容恢复到相应的寄存器，原程序从断点开始继续执行。

21.B、D

在程序查询方式中， CPU 与外设串行工作，传送与主程序串行工作。在中断方式中， CPU 与外设并行工作，当数据准备好时仍需中断主程序以执行数据传送，因此传送与主程序仍是串行 的。在 DMA 方式中， CPU 与外设、传送与主程序都是并行的。

22.C

按启动查询方式的不同，程序查询方式可分为定时查询方式和独占查询方式。在程序查询方 式 中 ， 由CPU 负责数据的传送。每完成 一 次数据传送后，将主存地址加1,计数值减1。

23.C

在 DMA 方式中，由外部设备向 DMA 控制器发出 DMA 请求信号，然后由 DMA 控制器向 CPU 发出总线请求信号。DMA 控制器在传送期间有总线控制权，此时 CPU 不能响应I/O 中 断。

326-2025年计算机组成原理考研复习指导

**24.C**

程序中断方式在数据传输时，首先要发出中断请求，此时 CPU 中断正在进行的操作，转而 进行数据传输，直到数据传送结束， CPU 才返回中断前执行的操作。 DMA 方式只是在后处理阶 段需要用中断方式请求CPU 做结束处理，但在整个数据传送过程，并不需要中断请求， A 错误。 DMA 方式和程序中断方式都有中断请求，但目的不同，程序中断方式的中断请求是为了进行数 据传送，而 DMA 方式的中断请求是在DMA 传送结束后请求 CPU 做 DMA 结束处理， B 错误、 C 正 确 。CPU 对 DMA 的响应可在指令执行过程中的任何两个存取周期之间， D 错误。

25.A

一个完整的 DMA 过程主要由 DMA 控制器控制，但也需要CPU 参与控制，只是 CPU 干 预 比较少，只需在数据传输开始和结束时干预，从而提高了CPU 的效率。

**26.A**

每个机器周期结束后，CPU 就可以响应 DMA 请求。注意区别： DMA 在与主存交互数据时 通过周期窃取方式，窃取的是存取周期。

27.B

DMA 请求的是总线的使用权，因此 CPU 对 DMA 请求的响应时机是一个总线周期结束时。 在流水线CPU 中，流水段的长度以最复杂的操作所花的时间为准，总线周期(访存时间)通常是 耗时最长的，因此通常可认为总线周期、存取周期、机器周期和流水段长度是等价的。

28.A

DMA 连接的是高速设备，其优先级高于中断请求，以防止高速设备数据丢失，A 正 确 。DMA 请求的响应时间可以发生在每个机器周期结束时，只要CPU 不占用总线；中断请求的响应时间 只能发生在每条指令执行完毕， B 错 误 。DMA 的优先级要比外中断(非屏蔽中断、可屏蔽中断) 高， C 错误。若不开中断，则内中断和非屏蔽中断仍可响应， D 错误。

29.B

DMA 方式只能用于数据传输，它不具有对异常事件的处理能力，不能中断现行程序，而键 盘和鼠标均要求 CPU 立即响应，因此无法采用 DMA 方式。

30.C

只有 DMA 方式是靠硬件电路实现的，三种基本的程序控制方式即直接程序传送、程序中断、 通道控制都需要程序的干预。

**31.** A

中断发生时，程序计数器内容的保护和更新是由硬件自动完成的，即由中断隐指令完成。

32.B

DMA 传送数据时，挪用周期不会改变CPU 现场，因此无须占用CPU 的 PC 和寄存器。

33.B

DMA 方式的数据传送不经过CPU, 但需要经过DMA 控制器中的数据缓冲寄存器。输入时， 数据由外设(如磁盘)先送往 DMA 的数据缓冲寄存器，再通过数据总线送到主存。反之，输出 时，数据由主存通过数据总线送到 DMA 的数据缓冲寄存器，然后送到外设。

**34.D**

当采用周期挪用进行 DMA 数据传送时，每当CPU 收 到DMAC 的总线申请，就将下一个总 线周期的总线控制权交给 DMAC 。DMAC 利用这个总线周期完成一个数据字的传送后，立即将 总线控制权交回给 CPU, 因此这里的总线周期也等于存取周期的长度。

35.D

DMA 方式主要用于磁盘等高速设备的成批数据传送，这类高速设备的记录方式多采用数据

第7章 输入/输出系统 327

块组织方式，因此每启动一次 DMA 传送，外设和主机之间就完成一个数据块的数据传送。

**36.** C

传送信息所在的通用寄存器编号不包含在初始化信息中，因为数据不是通过 CPU 中的通用 寄存器来传输的，而是直接通过DMA 控制器进行数据传输的。

**37.** A

外部中断是指CPU 执行指令以外的事件产生的中断，通常指来自CPU 与内存以外的中断。 选项 A 中键盘输入属于外部事件，每次键盘输入CPU 都需要执行中断以读入输入数据，所以能 引起外部中断。选项B 中除数为0属于异常，也就是内中断，发生在 CPU 内部。选项C 中浮点 运算下溢将按机器零处理，不会产生中断。而选项D 中访存缺页属于CPU 执行指令时产生的中 断，也不属于外部中断。所以能产生外部中断的只能是输入设备键盘。

**38.** A

在单级(或单重)中断系统中，不允许中断嵌套。中断处理过程为：①关中断；②保存断点； ③识别中断源；④保存现场；⑤中断事件处理；⑥恢复现场；⑦开中断；⑧中断返回。其中①~ ③由硬件完成，④~⑧由中断服务程序完成。

**39.** D

中断响应优先级是由硬件线路( 或查询程序)决定的，不便改动，而中断处理优先级可以利用 屏蔽字技术来动态调整。1 表示屏蔽该中断源的请求，0表示可以被该中断源中断。从中断处理优 先级来看，L₁ 只能屏蔽L₃ 和其自身，因此中断屏蔽字 M₁M₃M₂M₁M₀=01010。

**40.C**

每秒至少查询200次，每次查询至少500个时钟周期，则每秒最少占用200×500=100000个 时钟周期，因此占 CPU 时间的百分比为100000/50M=0.20%。

**41.B**

在响应外部中断的过程中，中断隐指令完成的操作包括：①关中断；②保存断点；③引出中 断服务程序(形成中断服务程序入口地址并送 PC), 所以只有 I 、Ⅲ 正 确。Ⅱ中保存通用寄存器 的内容是在进入中断服务程序后首先进行的操作。

42.D

中断 I/O 方式：在 I/O 设备输入每个数据的过程中，由于无须CPU 干预，因此可使 CPU 与 I/O 设备并行工作。仅当输完一个数据时，才需CPU 花费极短的时间去做一些中断处理。因此中 断申请使用的是CPU 处理时间，发生的时间是在一条指令执行结束之后，数据在软件的控制下完 成传送。而 DMA 方式与之不同。DMA 方式：数据传输的基本单位是数据块，即在 CPU 与 /O 设备之间，每次传送至少一个数据块； DMA 方式每次申请的是总线的使用权，所传送的数据是 从设备直接送入内存的，或者相反；仅在传送一个或多个数据块的开始和结束时，才需要 CPU 干 预，整块数据的传送是在控制器的控制下完成的。中断 I/O 方式不适合高速外设；多路型 DMA 控制器也适合同时为多个慢速外设服务，D 错误。

**43.B**

每400ns 发出一次中断请求，而响应和处理时间为100ns, 其中允许的延迟为干扰信息，因 为在50ns 内，无论怎么延迟，每400ns 仍要花费100ns 处理中断，所以该设备的I/O 时间占整个 CPU 时间的百分比为100ns/400ns=25%。

**44.B**

在程序中断I/O 方式中， CPU 和打印机直接交换，打印字符直接传输到打印机的I/O 端口， 不会涉及主存地址。而CPU 和打印机通过I/O 端口中的状态口和控制口来实现交互。

328-2025年计算机组成原理考研复习指导

**45.B**

多重中断在保护被中断进程现场时关中断，执行中断处理程序时开中断，B 错 误。CPU 一 般 在一条指令执行结束的阶段采样中断请求信号，查看是否存在中断请求，然后决定是否响应中断， A、D 正确。中断是指来自CPU 执行指令以外的事件， C 正确。

**46.C**

中断优先级分为响应优先级和处理优先级，响应优先级由硬件排队器( 或中断查询程序)决 定，处理优先级由屏蔽字决定，而非请求的先后次序决定。中断隐指令完成的工作有：①关中断； ②保存断点；③引出中断服务程序，通用寄存器的保护由中断服务程序完成。中断允许状态(即 开中断后),才能响应外部设备的中断请求，外部设备通常不能发出不可屏蔽中断，外部设备的 中断请求通常是为了输入/输出，这些事件并不是系统级的紧急事件，可以被屏蔽或延迟处理，若 允许外部设备发出不可屏蔽中断，则可能影响系统的稳定性和安全性。有中断请求时，若是关中 断的状态，或新中断请求的优先级较低，则不能响应新的中断请求。

47.A

设备接口中的数据缓冲寄存器为32位，即一次中断可以传输4B 数据，设备数据传输速率为 50kB/s, 共需要12 .5k 次中断，每次中断开销为1000个时钟周期， CPU 主 频 为 1GHz, 则 CPU 用 于该设备输入/输出的时间占整个CPU 时间的百分比最多是(12.5k×1000)÷1G=1.25%。

**48.D**

每类设备都配置一个设备驱动程序，设备驱动程序向上层用户程序提供一组标准接口，负责 实现对设备发出各种具体操作指令，用户程序不能直接和 DMA 打交道。DMA 的数据传送过程分 为预处理、数据传送和后处理3个阶段。预处理阶段由CPU 完成必要的准备工作，数据传送前由 DMA 控制器请求总线使用权；数据传送由 DMA 控制器直接控制总线完成；传送结束后， DMA 控制器向CPU 发送中断请求， CPU 执行中断服务程序做 DMA 结束处理。

**49.C**

访存时缺页属于内部异常， I 错误；定时器到时描述的是时钟中断，属于外部中断，Ⅱ正确； 网络数据包到达描述的是 CPU 执行指令以外的事件，属于外部中断，Ⅲ正确。

**50.B**

由 CPU 内部产生的异常称为内中断，内中断是不可屏蔽中断。通过中断请求线INTR 和 NMI,

从 CPU 外部发出的中断请求称为外中断，通过INTR 信号线发出的外中断是可屏蔽中断，而通过NMI 信号线发出的是不可屏蔽中断。不可屏蔽中断即使在关中断 (IF=0) 情况下也被响应， A 正确。不 可屏蔽中断的优先级最高，任何时候只要发生不可屏蔽中断，都要中止现行程序的执行，转到不可屏 蔽中断处理程序执行， C 正 确。CPU 响应中断需要满足3个条件：①中断源有中断请求；②CPU 允 许中断及开中断；③一条指令执行完毕，且没有更紧迫的任务。所以B 错误。

**51.** C

周期挪用法由DMA 控制器挪用一个或几个主存周期来访问主存，传送完一个数据字后立即 释放总线，是一种单字传送方式，每个字传送完后CPU 可 以 访 问 主 存 ，C 错 误。停 止CPU 访存 法则是指在整个数据块的传送过程中，使CPU 脱离总线，停止访问主存。

**52.** A

中断服务程序在内核态下执行，若只能在用户态下检测和响应中断，则显然无法实现多重中 断(中断嵌套), A 错 误。在多重中断中，CPU 只有在检测到中断请求信号后(中断处理优先级 更低的中断请求信号是检测不到的),才会进入中断响应周期。进入中断响应周期时，说明此时

第7章 输入/输出系统一329

CPU 一定处于中断允许状态，否则无法响应该中断。若所有中断源都被屏蔽(说明该中断的处理 优先级最高),则CPU 不会检测到任何中断请求信号。

**53.C**

中断 I/O 方式适用于字符型设备，此类设备的特点是数据传输速率慢，以字符或字为单位进 行传输， A 正确。若采用中断I/O 方式，当外设准备好数据后，向CPU 发出中断请求， CPU 暂时 中止现行程序，转去运行中断服务程序，由中断服务程序完成数据传送，B 正确。若外设准备数 据的时间小于中断处理时间，则可能导致数据丢失，以输入设备为例，设备为进程准备的数据会 先写入设备控制器的缓冲区(缓冲区大小有限),缓冲区每写满一次，就向CPU 发出一次中断请 求，CPU 响应并处理中断的过程，就是将缓冲区中的数据“取走”的过程，因此若外设准备数据 的时间小于中断处理时间，则可能导致外设往缓冲区写入数据的速度快于CPU 从缓冲区取走数据 的速度，从而导致缓冲区的数据被覆盖，进而导致数据丢失，C 错误。若采用中断 I/O 方式，则 外设为某进程准备数据时，可令该进程阻塞， CPU 运行其他进程，D 正确。

**54.D**

A 和 B 显然正确。开中断时， CPU 在执行完一条指令时检测中断请求信号，若检测到中断请 求信号，就立即响应；即便是多重中断，CPU 正在处理某个中断的过程中，由于中断屏蔽字的存 在，因此CPU 检测不到处理优先级更低的中断请求信号，若检测到中断请求信号，则说明其处理 优先级更高，同样也立即响应， C 正确。外设通过中断控制器向 CPU 发中断请求信号， CPU 响 应中断请求后开始执行中断服务程序，中断服务程序执行结束后CPU 自行返回(中断服务程序的 最后一条指令是返回指令),无须外设发中断结束信号，D 错误。

55.C

DMA 在预处理和后处理阶段需要CPU 来处理，而数据传输阶段由DMA 控制器完成。

**二、** **综合应用题**

01. 【解答】

没有。通常所说的 DMA 方式在主存和I/O 设备之间建立一条“直接的数据通路”,使得数据 在主存和I/O 设备之间直接进行传送，其含义并不是在主存和I/O 之间建立一条物理直接通路， 而是主存和 I/O 设备通过 I/O 设备接口、系统总线及总线桥接部件等相连，建立一个信息可以相 互通达的通路，这在逻辑上可视为直接相连的。其“直接”是相对于要通过CPU 才能和主存相连 这种方式而言的。

02. 【解答】

I/O 设备传送一个数据的时间为1÷(4×10⁴)s=25μs, 所以请求中断的周期为25μs, 而相应中 断处理程序的执行时间为40μs, 大于请求中断的周期，会丢失数据(单位时间内I/O 请求数量比 中断处理的多，自然会丢失数据),所以不能采用中断方式。

03. 【解答】

1)CPU 每秒对鼠标进行30次查询，所需的时钟周期数为100×30=3000。CPU 的时钟频 率为50MHz, 即每秒50×10⁶个时钟周期，因此对鼠标的查询占用 CPU 的时间比率为

[3000-(50×10⁶)]×100%=0.006%

可见，对鼠标的查询基本不影响 CPU 的性能。

2)对于硬盘，每32位(4B) 被 CPU 查询一次，因此每秒查询次数为2×2²°B÷4B=512K;

则每秒查询的时钟周期数为

100×512×1024=52.4×10⁶

330-2025年计算机组成原理考研复习指导

因此对硬盘的查询占用 CPU 的时间比率为

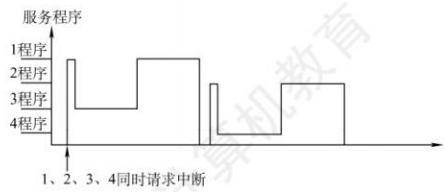
[52.4×10⁶ ÷(50×10⁶)]×100%=105%

可见，即使 CPU 将全部时间都用于对硬盘的查询，也不能满足磁盘传输的要求，因此 CPU 一般不采用程序查询方式与磁盘交换信息。

04. 【解答】

1)中断屏蔽字“1”表示不可被中断，“0”表示可被中断。根据表中“1”的个数的降序排 列可知，4个中断源的处理次序是3 → 1 → 4 → 2。

2)当4个中断源同时有中断请求时，由于硬件排队的优先次序是1 → 2 → 3 → 4,因此CPU 先 响应1的请求，执行1的服务程序。该程序中设置了屏蔽字1101,因此开中断指令后转 去执行3服务程序，且3服务程序执行结束后又回到了1服务程序。1服务程序结束后， CPU 还有2、4两个中断源请求未响应。由于2的响应优先级高于4,因此CPU 先响应2 的请求，执行2服务程序。在2服务程序中由于设置了屏蔽字0100,意味着1、3、4可 中断2服务程序。而1、3的请求已经结束，因此在开中断指令后转去执行4服务程序， 4服务程序执行结束后又回到2服务程序的断点处，继续执行2服务程序，直至该程序执 行 结 束 。CPU 执行程序的轨迹如下图所示。



**05.** 【解答】

根据字符设备的传输速率为9600b/s, 得每秒能传输

9600/8=1200B, 即1200个字符(本题中字符、字节不加以区分)

1)若采用 DMA 方式，传输1200个字符共需1200个存取周期，考虑到每传400个字符需中 断处理一 次，因此 DMA 方式每秒因数据传输占用处理器的时间是

5μs×(1200/400)=15μs

2)若采用中断方式，每秒因数据传输占用处理器的时间是

5μs×1200=6000μs

**06.【** 解答】

1)采用程序查询方式，硬盘传输速率为1MB/s, 一 个 字 为 3 2bit =4B, 每秒查询的次数为 1MB/4B=2.5×10⁵, 每秒查询所需的总时钟周期数为2.5×10⁵ ×100=2.5×10⁷。

CPU 的时钟频率为50MHz。

因 此 ，I/O 查询所花费的时间比率为2.5×10⁷ ÷50M=2.5×10⁷÷(5×107)=50%。 2)采用中断方式时，每传输一个字便进行一 次中断处理。

每秒产生的中断次数为1MB/4B=2.5×10 次 。

每秒用于传输的开销为2.5×10⁵ ×80=2×107个时钟周期。

因此花费的时间比率为(2×10?)÷(5×107)=40%。

3 ) 采 用 DMA 方式时， CPU 所花时间仅为启动时间和后处理时间。

每传输一次数据CPU 所花的时间为1000+500=1500个时钟周期。

第7章 输入/输出系统- 331

DMA 平均传送长度为4000B, 每秒产生的DMA 次数为1MB/s=(4×10³B)=250 。 因 此 ，CPU 为 DMA 所花费时间的比率为(1500×250)÷50M=0.75%。

07. 【解答】

1)按题意，外设每秒传送0.5MB, 中断时每次传送32bit=4B 。 由于CPI 为5,在中断方式 下 ，CPU 每次用于数据传送的时钟周期为5×18+5×2=100(中断服务程序+其他开销)。 为达到外设0.5MB/s 的数据传输速率，外设每秒申请的中断次数为0.5MB/4B=125000。

1秒内用于中断的开销为100×125000=12500000=12.5M 个时钟周期。

CPU 用于外设I/O 的时间占整个CPU 时间的百分比为12.5M/500M=2.5%。

2)当外设数据传输速率提高到5MB/s 时改用DMA 方式传送，每次 DMA 传送一个数据块， 大小为5000B, 则1秒内需产生的DMA 次数为5MB/5000B=1000。

CPU 用于DMA 处理的总开销为1000×500=500000=0.5M 个时钟周期。 CPU 用于外设I/O 的时间占整个CPU 时间的百分比为0.5M/500M=0.1%。

**08.【**解答】

本题涉及多个考点：计算机的性能指标、存储器的性能指标、DMA 的性能分析、DMA 方式 的特点及多体交叉存储器的性能分析。

1)平均每秒 CPU 执行的指令数为80M/4=20M, 因 此 MIPS 数为20。平均每条指令访存1.5

次，因此平均每秒Cache 缺失的次数=20M×1.5×(1-99%)=300K。 当 Cache 缺失时，

CPU 访问主存，主存与Cache 之间以块为传送单位，此时主存带宽为16B×300k/s=4.8MB/s。 在不考虑 DMA 传送的情况下，主存带宽至少达到4.8MB/s 才能满足CPU 的访存要求。

2)题中假定在 Cache 缺失的情况下访问主存，平均每秒产生缺页中断300000×0.0005%=1.5 次。因为存储器总线宽度为32位，所以每传送32位数据，磁盘控制器发出一次 DMA 请求，因此平均每秒磁盘 DMA 请求的次数至少为1.5×4KB/4B=1.5K=1536。

3)CPU 和 DMA 控制器同时要求使用存储器总线时， DMA 请求的优先级更高。因为，若 DMA 请求得不到及时响应， I/O 传输数据可能会丢失。

4)4体交叉存储模式能提供的最大带宽为4×4B/50ns=320MB/s 。

09. 【解答】

1)每传送一个 ASCⅡ 码字符，需要传输的位数有1位起始位、7位数据位 (ASCⅡ 码字符占 7位)、1位奇校验位和1位停止位，因此总位数为1+7+1+1=10。

I/O 端口每秒最多可接收1000/0.5=2000个字符。

2)一个字符传送时间包括：设备 D 将字符送 I/O 端口的时间、中断响应时间和中断服务程 序前15条指令的执行时间。时钟周期为1+50MHz=20ns, 设 备D 将字符送/O端口的时 间为0.5ms/20ns=2.5×10⁴个时钟周期。 一个字符的传送时间约为2.5×10⁴+10+15×4=25070 个时钟周期。完成1000个字符传送所需的时间约为1000×25070=25070000个时钟周期 。 CPU 用于该任务的时间约为1000×(10+20×4)=9×10⁴个时钟周期。

在中断响应阶段，CPU 主要进行以下操作：关中断、保护断点和程序状态、识别中断源。 **10.【**解答】

1)程序定时向缓存端口查询数据，由于缓存端口大小有限，必须在传输完端口大小的数据

时访问端口，以防止部分数据未被及时读取而丢失。设备A 准备32位数据所用的时间为 4B/2MB=2μs, 所以最多每隔2μs 必须查询一次，每秒的查询次数至少是1s/2μs=5×10⁵,

每秒 CPU 用于设备A 输入输出的时间至少为5×10⁹ ×10×4 =2×107个时钟周期，占整个 CPU 时间的百分比至少是2×10⁷ ÷500M=4%。

332-2025年计算机组成原理考研复习指导

2)中断响应和中断处理的时间为400×(1/500M)=0.8μs, 这时只需判断设备B 准备32位数 据要多久，若准备数据的时间小于中断响应和中断处理的时间，则数据被刷新，造成丢 失。经过计算，设备B 准备32位数据所用的时间为4B/40MB = 0.1μs,因此设备B 不适 合采用中断 I/O 方式。

3 ) 在 DMA 方式中，只有预处理和后处理需要 CPU 处理。设备B 每秒的 DMA 次数最多 为 4 0MB/s÷1000B=40000,CPU 用于设备B 输入/输出的时间最多为40000×500=2×10⁷ 个时钟周期，占CPU 总时间的百分比最多为(2×10/500M=4%。

**11.【**解析】

1)3个字段的名称为柱面号(或磁道号)、磁头号(或盘面号)、扇区号。由于每个盘面有 20000个磁道，因此该磁盘共有20000个柱面，柱面号字段至少占「log₂20000]=15 位 ； 由于该磁盘共有4个盘片，每个盘片有2个盘面，因此磁头号字段至少占log₂(4×2)=3 位； 由于每个磁道有500个扇区，因此扇区号字段至少占「log₂500]=9 位。

2)一个扇区的访问时间由寻道时间、延迟时间、传输时间三部分组成。平均寻道时间为5ms, 平均延迟时间等于磁盘转半圈所需要的时间，平均传输时间等于一个扇区划过磁头下方 所需要的时间。而该磁盘转一圈的时间为(60×10³)Y7200≈8.33ms, 因此一个扇区的平均访 问时间约为5+8.33/2+8.33/500≈9.18ms。

3)磁盘控制器中的数据缓冲区每充满一次，DMA 控制器就需要发出一次总线请求，将这 64bit 数据通过总线传送到主存，因此，在一个扇区的读/写过程中， DMA 控制器向 CPU 发送了512B/64bit=64 次总线请求。由于采用周期挪用 DMA 方式，因此当 CPU 和 DMA 控制器都需要访问主存时， DMA 控制器可以优先获得总线使用权。因为一旦磁盘开始读 /写，就必须按时完成数据传送，否则数据缓冲区中的数据会发生丢失。



**7.4** **本章小结**

本章开头提出的问题的参考答案如下。

1)I/O 设备有哪些编址方式?各有何特点?

统一编址和独立编址。统一编址是在主存地址中划出一定的范围作为I/O 地址，以便通过访 存指令即可实现对/O 的访问，但主存的容量相应减少。独立编址是指I/O 地址和主存是分开的， I/O 地址不占主存空间，但访存需专门的I/O 指令。

2)CPU 响应中断应具备哪些条件?

① 在 CPU 内部设置的中断屏蔽触发器必须是开放的。

② 外设有中断请求时，中断请求触发器必须处于“1”状态，保持中断请求信号。 ③ 外设(接口)中断允许触发器必须为“1”,这样才能把外设中断请求送至CPU。 具备上述三个条件时， CPU 在现行指令结束的最后一个状态周期响应中断。



**7.5** **常见问题和易混淆知识点**

1. 在开中断情况下，CPU 检测到中断就一定会立即响应吗?

在开中断情况下，CPU 总在每条指令执行结束时采样中断信号，此时若检测到中断请求信号，

第7章 输入/输出系统-333

则会立即响应。即便是多重中断，CPU 正在处理某个中断的过程中，由于中断屏蔽字的存在，CPU 检测不到处理优先级更低的中断请求信号，因此若检测到中断请求信号，则说明新中断源的处理 优先级更高，同样也会立即响应。

2. 向量中断、中断向量、向量地址三个概念是什么关系?

中断向量：每个中断源都有对应的处理程序，这个处理程序称为中断服务程序，其入口地址 称为中断向量。所有中断的中断服务程序入口地址构成一个表，称为中断向量表；也有的机器把 中断服务程序入口的跳转指令构成一张表，称为中断向量跳转表。

向量地址：中断向量表或中断向量跳转表中每个表项所在的内存地址或表项的索引值，称为 向量地址或中断类型号。

向量中断：指一种识别中断源的技术或方式。识别中断源的目的是找到中断源对应的中断服 务程序的入口地址的地址，即获得向量地址。

3. 程序中断和调用子程序有何区别?

两者的根本区别主要表现在服务时间和服务对象上不一样。

1)调用子程序过程发生的时间是已知的和固定的，即在主程序中的调用指令 (CALL) 执行 时发生主程序调用子程序过程，调用指令所在位置是已知的和固定的。而中断过程发生 的时间一般是随机的， CPU 在执行某个主程序时收到中断源提出的中断申请，就发生中 断过程，而中断申请一般由硬件电路产生，申请提出时间是随机的。也可以说，调用子 程序是程序设计者事先安排的，而执行中断服务程序是由系统工作环境随机决定的。

2)子程序完全为主程序服务，两者属于主从关系。主程序需要子程序时就去调用子程序， 并把调用结果带回主程序继续执行。而中断服务程序与主程序二者一般是无关的，不存 在谁为谁服务的问题，两者是平行关系。

3)主程序调用子程序的过程完全属于软件处理过程，不需要专门的硬件电路；而中断处理 系统是一个软/硬件结合的系统，需要专门的硬件电路才能完成中断处理的过程。

4)子程序嵌套可实现若干级，嵌套的最多级数受计算机内存开辟的堆栈大小限制；而中断 嵌套级数主要由中断优先级来决定， 一般优先级数不会很大。

**参** **考** **文** **献**

[1] 袁春风.计算机系统基础[M]. 北京：机械工业出版社，2018.

[2] 袁春风.计算机组成与系统结构[M]. 北京：清华大学出版社，2015.

[3] 袁春风.计算机系统基础习题解答与教学指导.北京：机械工业出版社，2019.

[4] 唐朔飞.计算机组成原理[M]. 北京：高等教育出版社，2008.

[5] 唐朔飞.计算机组成原理——学习指导与习题解答[M]. 北京：高等教育出版社.2012. [6] Randal E.Bryant. 深入理解计算机系统[M]. 北京：机械工业出版社，2016.

[7] 李春葆，肖忠付，杭小庆.计算机组成原理联考辅导教程[M]. 北京：清华大学出版社，2010.

[8] 本书编写组.全国硕士研究生入学统一考试计算机专业基础综合考试大纲解析[M]. 北京：高等教育出 版社，2014.

[9]徐爱萍 .计算机组成原理考研指导[M]. 北京：清华大学出版社，2003.

[10]谭志虎 .计算机组成原理(微课版) [M]. 北京：人民邮电出版社，2022.