**《计算机组成原理与结构》重要知识点**

1. **概论**
2. 基本概念
3. 冯.诺依曼体制？存储程序方式？

**a.采用二进制形式表示数据和指令b.采用存储程序方式工作(核心)**

**c.硬件组成:运算器、控制器、存储器、输入设备、输出设备;并规定了这五大部件的基本功能**

1. 控制流？数据流？

**控制流是指程序执行的顺序或流程，是程序中代码执行的控制结构。控制流可以通过条件分支语句（如if-else语句）、循环语句（如for循环、while循环）、函数调用等方式进行控制。例如，在一个if-else语句中，程序根据条件的真假决定执行哪个分支，这就是控制流的变化。**

**数据流是指程序中数据在各个变量之间流动的过程，是程序中数据处理的流程。数据流可以通过变量的赋值、函数的返回值等方式进行流动。例如，在一个简单的加法程序中，数据流就是先将两个数相加，然后将结果存储在一个变量中，这个过程就是数据流的流动。**

1. 模拟信号？数字信号？数字信号有哪两种？

**a.模拟信号:随时间连续变化的电信号**

**b.数字信号:在时间或空间上断续变化的电信号**

**电平型**

**脉冲型**

1. 总线及其组成？

**总线：能为多个部件分时共享的一组信息传送线路及相应的控制逻辑。**

**数据总线、地址总线和控制总线**

1. 接口的概念？

**系统总线与I/O设备之间的交接部分称为I/O接口。**

1. 基本字长？

**基本字长指参加一次运算的操作数的位数。如：8、16、32、64位**

**字长影响计算精度、指令功能。**

1. CPU主频？时钟频率？

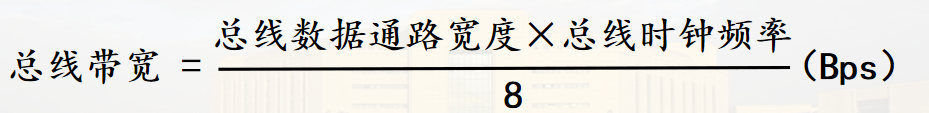
**CPU主频和时钟频率：决定CPU内核工作速度的重要因素**

1. 数据通路宽度？数据传输率？

**指数据总线一次能并行传送的数据位数。**

**指数据总线每秒传送的数据量（字节），也称为带宽。**

**计算公式**：



1. **计算机中的信息表示**
2. 基本概念
3. 一个数值型数据的完整表示需三个方面？

**一个数值型数据的完整表示需要考虑数值本身、数据的符号以及数据的精度**

1. 权、基数？

**基数：每个数位中所允许使用的数码个数。**

**(位)权：与所在数位相关的常数，表示该进位计数制中，数符“1”在该数位所代表的值。**

1. 真值？机器数的定义？分类？

**真值：用正、负符号加绝对值表示数值**

**机器数:在计算机内部使用，将符号数值化；**

**机器数由符号位+有效位构成，最高位表示符号，**

**正数最高位为0，负数最高位为1；**

**根据编码方法，机器数有原码、反码、补码表示。**

1. 数的定点表示与浮点表示？

**定点表示是一种表示数值的方式，其中数值被表示为定点数（Fixed-point number）。在定点表示中，数值通常分为整数部分和小数部分，小数点的位置是固定的，因此被称为定点。定点表示的数值通常使用整数类型来存储。**

**浮点表示是另一种表示数值的方式，其中数值被表示为浮点数（Floating-point number）。在浮点表示中，数值通常表示为一个带有指数部分的科学计数法形式。浮点表示的数值通常使用浮点类型来存储，浮点类型可以表示更大的数值范围和更高的精度。**

(下面是计组第三章内容)

1. 指令？指令系统？

**指令：一系列按照某种规律有序排列的，能被CPU识别、执行的二进制代码。**

**指令系统（指令集）：一台计算机所能执行的全部指令的集合。**

1. 地址码？寻址方式？地址结构？

**地址码是指计算机中用于指示存储器中某个特定单元的代码。计算机中每个存储单元都有一个唯一的地址码，用于访问该存储单元中的数据。地址码通常是一个数字，用于标识存储单元在存储器中的位置。**

**地址结构是指计算机中存储器地址的组织结构。地址结构通常由若干位组成，每一位表示存储器中的一个存储单元，因此地址结构决定了计算机可以寻址的存储单元数量。**

1. 寻址方式：

**大致可将众多的寻址方式归纳为以下四大类，其它的寻址方式则是它们的变型或组合。**

**① 立即寻址。在读取指令时也就从指令之中获得了操作数，即操作数包含在指令中。**

**② 直接寻址类。直接给出主存地址或寄存器编号，从CPU内或主存单元内读取操作数。**

**③ 间接寻址类。先从某寄存器中或主存中读取地址，再按这个地址访问主存以读取操作数。**

**④ 变址类。指令给出的是形式地址（不是最终地址），经过某种变换（例如相加、相减、高低位地址拼接等），才获得有效地址，据此访问主存储器以读取操作数。**

8.何谓隐式I/O指令？其主要特点是什么？

**所谓隐式I/O指令，就是用传送指令实现I/O操作，即针对统一编址，用传送指令访问I/O端口**

**不影响标志寄存器的值**

1. 主机调用外围设备，外设编制可采用那几种方式？

**无条件传送，适用于总是处于准备好状态的外设；2、查询方式传送；3、中断方式传送；4、直接存储器存取（DMA）。**

**{I/O端口：外设可以通过I/O端口与主机进行通信，主机可以向指定的I/O端口发送命令或数据，并从指定的I/O端口接收数据。**

**中断：外设可以通过中断方式向主机发出信号，请求主机的处理器暂停当前任务，转而处理外设的请求。这种方式可以避免主机不断轮询外设的情况，提高系统效率。**

**DMA（直接内存访问）：DMA是一种数据传输方式，允许外设直接访问主机内存中的数据，而不需要经过处理器。这种方式可以提高数据传输速度，减少CPU占用率。**

**存储器映射I/O：存储器映射I/O允许外设通过访问特定的内存地址来与主机进行通信。这种方式使得外设编制可以像访问内存一样来访问I/O设备，简化了编程流程。}**

二、计算题

1.扩展操作码：根据条件计算指令条数？（ppt中练习题）



**2.根据寻址方式寻找操作数？（重点，作业题）**

**3.IEEE754短浮点数格式表示？（重点，作业题）**

1. **中央处理器**
2. 基本概念
3. CPU内部各寄存器的功能？

**程序计数器（PC）：用于存放当前正在执行的指令地址，每次执行完一条指令，PC会自动加上该指令的长度，以便继续执行下一条指令。**

**指令寄存器（IR）：用于存放当前正在执行的指令，IR会从内存中读取指令，并将其存入寄存器中，以便CPU进行解码和执行。**

**累加器（ACC）：用于存放算术或逻辑运算的结果，例如加、减、乘、除等。在大多数CPU中，ACC是最常用的寄存器之一**

**状态寄存器（PSW）：用于存放CPU的状态标志，例如零标志、进位标志、溢出标志等。PSW的状态会受到各种指令的影响，以便CPU进行判断和跳转。**

**堆栈指针（SP）：用于指示当前堆栈的位置，每次调用子程序或发生中断时，SP会自动向下移动，以便将返回地址等数据存入堆栈中。**

**基址指针（BP）：用于指示当前堆栈帧的基址，即堆栈中存放当前子程序局部变量和参数的位置。**

**段寄存器（Segment Registers）：用于存放内存段的起始地址，通常包括代码段、数据段、堆栈段等。不同的段寄存器对应不同的内存段，以便CPU进行内存访问。**

**通用寄存器（General Registers）：用于存放临时数据和地址，通常包括AX、BX、CX、DX等。在x86架构中，通用寄存器可以被用于存放任意数据或地址，具有很高的灵活性。**

1. 数据通路结构？

**数据通路（Data Path）是指CPU中用于执行指令的数据路径，通常由一组互相连接的数据通路元件（Data Path Element）组成，例如加法器、ALU、寄存器等。**

**数据通路的结构通常分为以下几个部分：**

**输入寄存器：用于暂时存放指令或数据，通常包括指令寄存器、数据寄存器等。输入寄存器接收来自存储器或其他外部设备的数据，以便CPU进行处理。**

**运算器：用于执行算术或逻辑运算，通常包括加法器、乘法器、逻辑单元等。运算器的输入来自输入寄存器或其他数据通路元件，输出则传送到输出寄存器或其他数据通路元件。**

**输出寄存器：用于暂存运算器的输出结果，通常包括累加器、状态寄存器等。输出寄存器的数据可以被传送到存储器或其他外部设备，也可以作为下一条指令的操作数。**

**存储器：用于存储指令和数据，通常包括指令存储器和数据存储器两种。指令存储器用于存放程序代码，数据存储器用于存放程序数据和堆栈。**

**控制器：用于控制整个CPU的操作流程，通常包括时钟、计数器、解码器等。控制器根据指令的类型和操作数，生成对应的控制信号，以便数据通路元件进行对应的操作。**

1. 何谓同步控制方式，何谓异步控制？有何主要特征？应用场合？

**同步控制方式是指控制信号的传输是基于系统的时钟信号的，控制信号的产生和传输都遵循时序规律，所有的操作都在时钟的节拍下完成，因此同步控制方式具有时序准确性和可靠性高的优点。应用场合通常是在需要高精度、高可靠性、高效率的场景中，例如CPU内部的控制器、数字信号处理器等。**

**异步控制方式则是指控制信号的传输不依赖于时钟信号，而是通过状态的变化来产生和传输，控制信号的产生和传输是根据硬件电路的实际变化来完成的，因此异步控制方式具有响应速度快、适应性强等特点。应用场合通常是在需要高速、低延迟、可扩展性好的场景中，例如异步串行通信接口、异步电路等。**

1. 何谓主从设备，试举例说明。

**主设备是指可以主动控制数据传输的设备，而从设备是指被动等待主设备控制的设备。主设备通过控制总线向从设备发送命令，从设备根据命令执行相应的操作，最终将数据传输回主设备。**

**例如，计算机中的硬盘和光驱就是从设备，它们等待主设备（通常是CPU）发出的命令来读取或写入数据。而主板上的芯片组和PCIe控制器等设备则可以作为主设备，向硬盘或光驱等从设备发出读写命令，并控制数据传输的过程。**

**在传统的计算机系统中，CPU通常是主设备，其他的设备都是从设备。但随着计算机系统的发展，一些新型的设备也可以作为主设备，例如一些具备处理器功能的外围设备（例如GPU）可以作为主设备，向其他设备发出命令并控制数据传输的过程。**

1. 同步控制中如何引入异步应答的，试举例说明.

**在同步控制中，通常要求主设备在发送命令后等待从设备的应答信号，然后才能继续进行下一步操作。这种方式虽然保证了数据传输的正确性，但会降低系统的效率和响应速度。为了克服这个问题，可以引入异步应答机制。**

**具体而言，异步应答机制是指从设备在完成操作后不等待主设备的控制信号，而是主动向主设备发送应答信号。主设备接收到应答信号后，就可以继续进行下一步操作，从而提高系统的效率和响应速度。**

**一个典型的应用场景是在内存控制器中，由于内存访问的速度较慢，因此需要引入异步应答机制。当CPU向内存控制器发送读写请求时，内存控制器会立即向CPU发送应答信号，表示已经开始处理请求，然后内存控制器会在完成请求后再次向CPU发送应答信号，表示操作已经完成。由于CPU不需要等待内存控制器的控制信号，可以继续执行其他指令，从而提高了系统的效率和响应速度。**

1. 微程序控制思想的基本要点是什么？

**将指令集的执行过程分解为一组基本操作，每个操作对应一组微指令。**

**将微指令存储在专门的微程序存储器中，每个指令包含一组控制信号，用于控制计算机硬件的操作。**

**当CPU执行指令时，根据指令操作码从微程序存储器中读取对应的微指令，然后根据微指令中的控制信号来控制计算机硬件的操作。**

**在微程序存储器中，可以使用分支和跳转等指令来实现复杂的控制流程，从而可以支持各种不同的指令集架构和复杂的操作。**

**微程序控制方式具有灵活性和可扩展性，因为只需更新微程序存储器中的微指令，就可以支持新的指令集或者修改指令的执行方式。**

**微程序控制方式的缺点是速度较慢，因为需要额外的读取微指令的过程。同时，微程序控制方式也需要较大的存储空间来存储微程序存储器，占用了较多的芯片面积。**

1. 判断下面叙述是否正确，说明理由：串行加法器中的进位链是串行进位链，并行加法器中的进位链只有并行进位链。

**错，并行加法器中的串行进位链结构：Cn = Gn + PnCn-1**

1. 常用的加法器进位链结构有哪几种？

**串行进位链结构：每个加法器模块的进位输出作为下一个加法器模块的进位输入，进位信号串行传递，因此速度较慢，但硬件资源消耗较少。**

**并行进位链结构：每个加法器模块的进位输入都连接到同一级的进位逻辑电路中，多个进位信号并行传递，因此速度较快，但硬件资源消耗较多。**

**分组进位链结构：将加法器模块分成若干个组，每个组内部采用串行进位链结构，不同组之间采用并行进位链结构，可以在保证速度的同时节省硬件资源。**

**多级进位链结构：将进位逻辑电路分成多个级别，每个级别中的进位逻辑电路只处理该级别的进位信号，可以减小进位传递延迟，提高加法器的运行速度。**

1. 组合逻辑控制器、微程序控制器有何区别？

**组合逻辑控制器（Combinational Logic Controller，CLC）是一种基于组合逻辑电路实现的控制器。它的主要特点是没有状态信息存储，只能根据输入条件直接输出相应的控制信号。因此，它的设计简单、速度快、可靠性高，适用于逻辑比较简单的控制场合。**

**微程序控制器（Microprogrammed Controller，MPC）是一种基于微程序实现的控制器。它的主要特点是在控制器内部存储了一组微指令，每个微指令对应一组控制信号，通过按照一定的顺序执行微指令序列来控制外部设备的工作。微程序控制器可以根据需要修改微指令序列，从而实现不同的控制功能，具有较高的灵活性和可扩展性。**

**总的来说，组合逻辑控制器和微程序控制器的主要区别在于控制方式的不同。组合逻辑控制器采用硬连线的方式实现控制功能，缺乏灵活性和可扩展性，适用于逻辑比较简单的控制场合；而微程序控制器采用存储器存储微指令序列，可以灵活地修改和扩展控制功能，适用于逻辑较为复杂的控制场合。**

1. 组合逻辑控制器、微程序控制器的时序系统是如何划分的？

**指令存储器访问时序：这个部分控制指令的读取，包括指令存储器的选通、地址计数器的自增等。**

**指令译码时序：这个部分对从指令存储器读取的指令进行译码，确定控制器需要执行的操作，生成相应的控制信号。在组合逻辑控制器中，这个部分由硬连线的逻辑电路实现；而在微程序控制器中，这个部分由微指令存储器和微指令译码电路实现。**

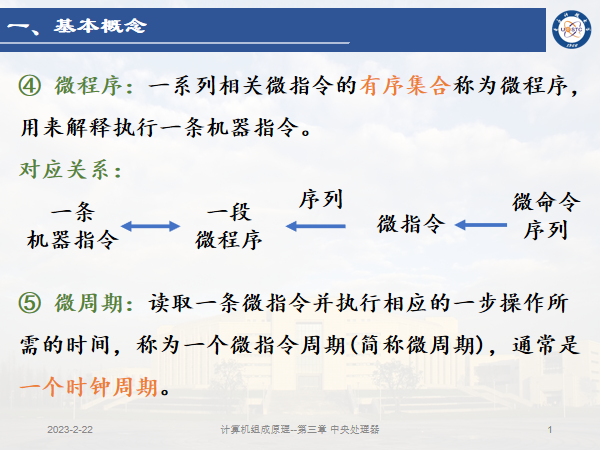
**控制信号发生器时序：这个部分根据指令译码结果生成相应的控制信号，并将控制信号输出到外部设备中。**

**外部设备响应时序：这个部分控制外部设备的响应时间，包括输入设备的数据获取、输出设备的数据输出等。**

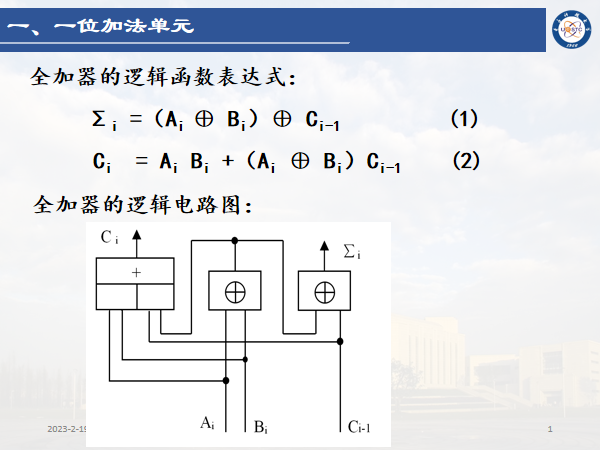
**在微程序控制器中，时序系统还包括微指令执行时序，即控制器内部执行微指令的时序控制，包括微指令读取、微指令译码、控制信号生成等。由于微程序控制器的控制逻辑是由一组微指令序列实现的，因此它的时序系统要比组合逻辑控制器更为复杂。**

11.微命令、微操作、微指令、微指令周期、微程序？





12.1位全加器的结构及关系表达式。



13.并行加法器中的串行进位链结构：Cn = Gn + PnCn-1

并行进位链结构：Cn = Gn + PnGn-1+ … + Pn…P1C0

1. 时序信号？时序系统？

**时序信号指在数字系统中用来同步或控制操作序列的信号，例如时钟信号、复位信号、启动信号等。时序信号一般是脉冲信号，其状态发生变化表示一个特定的时间点。**

**时序系统指一个数字系统中所有时序信号的集合，包括时钟信号、复位信号、启动信号等。时序系统控制着数字系统中所有的操作和数据流，确保它们按照正确的顺序进行和传输。时序系统的设计需要考虑各个时序信号的相互作用和时序要求，以满足数字系统的功能要求和性能要求。**

二、设计题：（重点）

CPU的逻辑组成及工作机制

1.CPU的逻辑组成（模型机框图）

（1）CPU的逻辑组成→模型机框图；

（2）CPU内每个寄存器的作用；

（3）总线的分类及定义；

（4）控制器的分类及区别；

2.CPU的指令流程

（1）指令类型：MOV指令、双操作数算数逻辑运算指令、单操作数算是逻辑运算指令、转移/返回指令、转子指令；

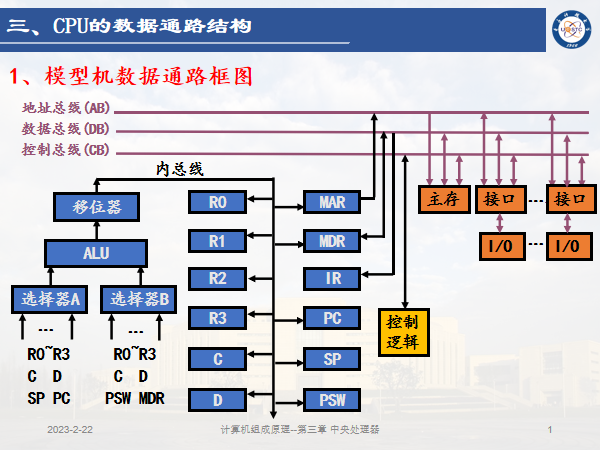
（2）核心是寻址方式：立即寻址、R、(R)、—(R)、(R)+、@(R)+、X(R)；

3.操作时间表的安排（微命令的安排）：

（1）CPU数据通路操作：按照数据的流向分成四段

ALU输入选择→AUL功能选择→移位器功能选择→分配脉冲（打入到寄存器中的脉冲）；

1. 与访问主存有关的微命令。



1. **存储子系统**
2. 基本概念
3. 主存、缓存、外存？

**能由CPU直接编程访问，存放当前CPU需要执行的程序与需要处理的数据**

**存放即将要使用的程序与数据，作为主存中当前活跃信息的副本。**

**存放需要联机保存、但暂不使用的大量程序与数据。**

1. 高速缓存Cache用来存放什么内容？设置它的主要目的是什么？

**存放即将要使用的程序与数据，作为主存中当前活跃信息的副本。**

**有效缓解访存的瓶颈问题**

1. 存储介质？

**半导体存储器、磁表面存储器、光盘存储器**

1. 何谓随机存取？何谓顺序存取？何谓直接存取？请各试举一例。

**① 可按地址随机地访问任一存储单元；**

**② 访问各存储单元所需时间相同，与地址无关；作主存、高速缓存**

**典型的随机存取存储器是RAM（Random Access Memory），如计算机中的内存。**

**信息按顺序存放，访问时间与信息存放位置有关。**

**磁带是采取顺序存取方式的存储器。**

**典型的顺序存取存储器是磁带存储器，如早期的计算机中使用的磁带。**

**（访问时读/写部件先直接指向一个小区域，再在该区域内顺序查找。访问时间与数据位置有关）直接存取是指可以通过存储单元的物理地址直接访问存储单元的存储器访问方式。在直接存取方式下，可以直接访问任意存储单元，而不需要先访问其他存储单元。典型的直接存取存储器是ROM（Read Only Memory），如计算机中的固化程序存储器。**

1. 静态存储器SRAM、动态存储器DRAM存储原理？

**静态存储器SRAM（静态MOS型）,不需刷新，依靠双稳态电路内部交叉反馈的机制存储信息。**

**动态存储器DRAM（动态MOS型），需定期刷新，依靠电容存储电荷的原理存储信息。**

1. 动态刷新分为哪几种情况，各有什么特点？

**集中刷新**

**2ms内集中安排所有刷新周期，用在实时要求不高的场合**

**分散刷新**

**各刷新周期分散安排在存取周期中，用在低速系统中**

**异步刷新**

**各刷新周期分散安排在2ms内。每隔一段时间刷新一行用在大多数计算机中。**

1. 全地址译码方式？部分地址译码方式？

**1）全地址译码方式**

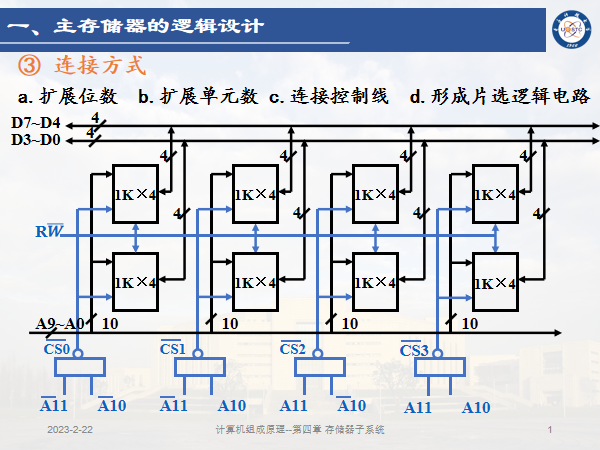
**全部地址信号都使用，高位地址信号用作译码器的输入，低位地址信号接存储芯片的地址输入线；**

**2）部分地址译码方式**

**仅用地址总线的一部分信号线与存储器连接：通常是用高位地址信号的一部分作为片选译码信号；低位地址信号接存储芯片的地址输入线。**

二、设计题：（重点）

半导体存储器逻辑设计：(地址分配、片选逻辑、逻辑框图)，片选逻辑采用全/部分译码方式。



1. **I/O系统**
2. 基本概念
3. 系统总线上一般包含哪三组信息？

**一组能为多个部件分时共享的公共的信息传送线。地址信息、数据信息、控制信息。**

1. 中断方式定义、实质、特点？

**CPU暂时中止现行程序的执行，转去执行为某个随机事态服务的中断处理程序，处理完毕后自动恢复原程序的执行。**

**程序切换**

**可处理随机事件**

1. 中断向量、中断向量表、向量地址？

**中断向量：中断服务程序的入口地址(还可包含状态字)；**

**中断向量表：按中断类型码的顺序存放各中断向量，在一段连续的存储区；**

**向量地址：访问中断向量表内的中断向量所需的存储单元地址**

1. 何谓向量中断？何谓非向量中断？各有何优点和缺点？

**向量中断:由硬件提供中断服务程序入口地址**

**非向量中断:由软件提供中断服务程序入口地址**

**向量中断机制更灵活和高效，但是实现起来也更为复杂，需要预定义向量表，并且在添加新的中断类型时需要进行修改。非向量中断机制比较简单，但是只能处理一个中断类型**

1. 单级中断、多重中断？

CPU**响应后只处理一个中断源的请求，处理完毕后才能响应新的请求。**

**在中断服务过程中，允许响应处理更高级别的中断请求。**

1. 程序直传方式定义、实质、特点？

**程序直接传送方式是指完全通过程序来控制主机和外围设备之间的数据传送**

**利用计算机系统中的直接内存访问（Direct Memory Access，DMA）控制器，将外部设备中的程序数据直接传输到内存中，绕过了CPU的参与，从而实现了高速的数据传输**

**高速传输：DPL可以绕过CPU，直接将数据传输到内存中，因此速度较快。**

**硬件支持：DPL需要硬件支持，计算机系统必须有DMA控制器才能实现。**

**直接加载：DPL可以直接从外部设备中加载程序，不需要先将程序复制到内存中，从而节省了空间和时间。**

**非常规加载：DPL可以通过不同的方式加载程序，如从网络中加载程序等，这使得它具有更多的灵活性。**

**安全性：DPL在传输过程中可以进行数据校验和错误纠正，从而确保了数据的完整性和安全性。**

1. 程序直传方式，中断方式，DMA方式之间的区别？

**程序直传方式（Direct Program Load，DPL），中断方式（Interrupt Driven I/O）和DMA方式（Direct Memory Access）都是计算机系统中常见的实现数据传输的方式，它们之间的主要区别如下：**

**实现方式不同：DPL和DMA都是硬件级别的机制，而中断方式是一种软件级别的机制**

**中断处理方式不同：在中断方式中，CPU会在I/O设备完成操作之后接收到一个中断请求，并通过中断服务程序来处理数据传输；而在DPL和DMA中，CPU可以绕过I/O设备，直接访问内存，不需要中断处理。**

**处理效率不同：DMA和DPL的效率较高，因为它们可以绕过CPU直接进行数据传输，而中断方式需要通过CPU进行中断处理，效率较低。**

**实现难度不同：中断方式的实现相对简单，但需要一定的编程技巧；而DPL和DMA需要硬件支持，并且需要进行更复杂的配置和调试。**

**应用场景不同：中断方式适合小量数据传输，因为数据传输的过程中需要频繁中断CPU，从而降低了效率；而DPL和DMA适合大量数据传输，可以提高数据传输的效率和速度。**

**《微型计算机原理与接口技术》重要知识点**

**第2章 微处理器与总线**

一、基本概念

1.微处理器主要由哪几部分构成？

**微处理器主要由运算器、控制器、内部寄存器等三个部分构成。**

1. 说明8088CPU中EU和BIU的主要功能。在执行指令时，EU能直接访问存储器吗？

**EU主要负责指令的执行，包括指令的解码、操作数的读取和结果的存储等。EU包含了寄存器组、算术逻辑单元（ALU）和标志寄存器等部分，可以执行各种算术和逻辑运算，以及数据的移动和比较等操作。**

**BIU主要负责和外部设备进行数据交换，包括内存的读写、I/O端口的读写等。BIU包含了指令队列、指令指针寄存器、地址计算单元等部分，可以实现指令的预取和地址的计算等功能。**

**在执行指令时，EU不能直接访问存储器，需要通过BIU进行数据交换。具体地，BIU会根据EU中的指令指针寄存器的值，从存储器中读取指令，并将指令存储到EU中的指令队列中。EU从指令队列中读取指令进行解码和执行，如果需要读取或写入内存或I/O端口，则通过BIU进行数据交换。**

3.8086/8088CPU中，有哪些通用寄存器和专用寄存器？说明他们的作用。

**通用寄存器包括：**

**AX寄存器：累加器，可以用于算术运算、逻辑运算、移位操作等。**

**BX寄存器：基址寄存器，可以用于存储指针、数组的基地址等。**

**CX寄存器：计数器，可以用于循环控制、移位操作等。**

**DX寄存器：数据寄存器，可以用于存储数据、I/O端口地址等。**

**SP寄存器：堆栈指针，用于指向当前堆栈顶部。**

**BP寄存器：基址指针，用于存储堆栈帧的基地址。**

**SI寄存器：源变址寄存器，可以用于存储源数据地址。**

**DI寄存器：目的变址寄存器，可以用于存储目的数据地址。**

**专用寄存器包括：**

**IP寄存器：指令指针，用于存储下一条要执行的指令地址。**

**CS寄存器：代码段寄存器，用于存储代码段的段地址。**

**DS寄存器：数据段寄存器，用于存储数据段的段地址。**

**ES寄存器：附加段寄存器，用于存储附加数据段的段地址。**

**SS寄存器：堆栈段寄存器，用于存储堆栈段的段地址。**

**FLAGS寄存器：标志寄存器，用于存储标志位，如进位标志、零标志、符号标志等。**

4.8086/8088系统中，存储器为什么要分段？一个段最大为多少字节？最小为多少字节？

**在8086/8088系统中，存储器需要分段主要是为了灵活地管理大容量存储器。每个段有独立的段基址和段长度，以及访问权限。通过分段，CPU可以方便地管理内存，充分利用内存空间，避免存储器碎片化，提高系统运行效率。**

**在8086/8088系统中，一个段的最大字节数为64KB（2的16次方），最小字节数为1个字节。这是因为8086/8088采用了基于16位的存储结构，每个段的地址空间为2的16次方，即64KB。由于一个字节占8位，因此最小字节数为1个字节。同时，每个段的长度也要是16位对齐，即为偶数个字节，这是为了保证数据的正确对齐和访问效率。**

5.8086/8088系统中，物理地址和逻辑地址是指什么？

**逻辑地址是CPU在访问内存时使用的地址，也称为虚拟地址。它由段地址和偏移地址组成，段地址和偏移地址都是16位的。段地址指向段的起始地址，偏移地址指向段内的具体地址。CPU通过将段地址左移4位（乘以16），再加上偏移地址，得到物理地址。**

**物理地址是实际存在于存储器中的地址，也称为实地址。在8086/8088系统中，物理地址空间为1MB，由20根地址线构成。CPU通过地址总线将逻辑地址转换为物理地址，然后将物理地址传输给存储器或外设进行读写操作。**

6.8088/8086 CPU的特点？

**8088/8086采用复杂指令集计算机（CISC）体系结构，支持大量的指令集，包括算术操作、逻辑操作、移位操作、控制转移指令等。**

**8088/8086拥有通用寄存器和专用寄存器，其中通用寄存器包括AX、BX、CX和DX等寄存器，专用寄存器包括IP、SP、BP和FLAGS等寄存器。**

**8088/8086采用分段存储器结构，将存储器分为若干段，每个段的大小为64KB，最多可以有64KB × 4 = 256KB的存储空间。**

**8088/8086采用基于总线的架构，数据总线和地址总线宽度均为16位，可以访问64KB的存储器空间。**

**8088/8086拥有一个复杂的指令执行流水线，可以实现高效的指令执行，但由于流水线深度较浅，其性能相对较低。**

**8088/8086的主频为4.77MHz，可以通过外部时钟信号提高时钟频率，最高可达10MHz。**

1. 计算题：

8086/8088系统中，物理地址和逻辑地址的转换？

**物理地址 = 段基址 × 16 + 偏移地址**

**其中，偏移地址是指逻辑地址相对于段起始位置的偏移量。由于8086/8088是16位系统，所以需要将段基址左移4位（乘以16）才能得到实际的物理地址。**

1. **8086/8088指令系统**
2. 基本概念

1.8086/8088指令系统的8种寻址方式

**直接寻址（Immediate Addressing）：直接将数据或操作数作为指令的一部分，例如：MOV AX, 1234H。**

**寄存器寻址（Register Addressing）：将操作数或数据存储在寄存器中，例如：MOV AX, BX。**

**寄存器间接寻址（Register Indirect Addressing）：通过寄存器存储内存地址，访问内存中的数据或操作数，例如：MOV AX, [BX]。**

**寄存器相对寻址（Register Relative Addressing）：通过一个基址寄存器加上一个偏移量来访问内存，例如：MOV AX, [BX+SI]。**

**基址加变址寻址（Base Index Addressing）：通过一个基址寄存器和一个变址寄存器加上一个可选的偏移量来访问内存，例如：MOV AX, [BX+SI+10]。**

**相对基址加变址寻址（Based Index Addressing with Displacement）：通过一个基址寄存器和一个变址寄存器加上一个偏移量来访问内存，偏移量可以是正数或负数，例如：MOV AX, [BX+SI+10]。**

**相对寻址（Relative Addressing）：通过一个相对偏移量来访问内存，例如：JMP LABEL。**

**基址加相对寻址（Based Addressing）：通过一个基址寄存器和一个相对偏移量来访问内存，例如：JMP FAR PTR [BX+10]。**

二、六大类指令

1.数据传送：

1）通用数据传送：MOV、堆栈操作指令push/pop、交换指令XCHG、查表转换指令XLAT、（字位扩展指令不要求）

2）输入输出IN、OUT

3）地址传送LEA、LDS、LES

4）标志位操作PUSHF、POPF、LAHF、SAHF

2.算术运算类指令：

1）加法运算指令ADD、ADC、INC

2）减法运算指令SUB、SBB、DEC、CMP（求补指令NEG不要求）

3）乘法指令（不要求）

4）除法指令（不要求）

**MOV指令用于将一个数据或者寄存器中的数据传送到另一个寄存器或内存位置。**

**PUSH和POP指令用于实现堆栈操作，将数据压入或弹出堆栈。PUSH将数据推入堆栈，POP将数据从堆栈中弹出。**

**XCHG指令用于交换两个寄存器或内存位置中的数据。**

**XLAT指令用于查找一个由AL指定的表中的数据，并将表中的值传送到AL中。**

**2）指令解释：**

**IN指令用于从指定的端口读取一个字节或一个字。**

**OUT指令用于将一个字节或一个字发送到指定的端口。**

**3）指令解释：**

**LEA指令用于将一个有效地址（偏移地址）传送到一个寄存器中。**

**LDS和LES指令用于将一个有效地址传送到一个寄存器对应的段寄存器中。**

**4）指令解释：**

**PUSHF和POPF指令用于将标志寄存器中的标志推入堆栈或者从堆栈中弹出。**

**LAHF指令用于将标志寄存器的低八位传送到AH寄存器中。**

**SAHF指令用于将AH寄存器中的低八位传送到标志寄存器中。**

**5）指令解释：**

**ADD指令用于将两个操作数相加，结果存储在第二个操作数中。**

**ADC指令用于将两个操作数以及进位标志相加，结果存储在第二个操作数中。**

**INC指令用于将一个操作数加1，结果存储在该操作数中。**

**SUB指令用于将第二个操作数从第一个操作数中减去，结果存储在第一个操作数中。**

**SBB指令用于将第二个操作数和借位标志从第一个操作数中减去，结果存储在第一个操作数中。**

**DEC指令用于将一个操作数减1，结果存储在该操作数中。**

**CMP指令用于比较两个操作数的值，结果并不存储在任何地方。**

3.逻辑运算和移位指令：AND、OR、NOT、XOR、TEST、SHL、SHR、SAL、SAR、ROL、ROR、RCR、RCL

4.串操作指令：MOVS、CMPS、SCAS、LODS、STOS

5.程序控制指令：

1）转移指令：JMP、条件转移指令

2）循环控制指令：LOOP、LOOPZ/LOOPE、LOOPNZ/LOOPNE

3）过程调用指令：CALL

4）中断控制指令：INT

6.处理器控制指令：常用的CLD 、STD、CLI、STI、HLT、CLC、STC

**AND：按位与操作，将两个操作数对应位进行与运算，结果存放在目的操作数中。**

**OR：按位或操作，将两个操作数对应位进行或运算，结果存放在目的操作数中。**

**NOT：按位非操作，将操作数的每一位取反，结果存放在目的操作数中。**

**XOR：按位异或操作，将两个操作数对应位进行异或运算，结果存放在目的操作数中。**

**TEST：测试指令，与AND指令相似，但不改变目标操作数的值。**

**SHL/SHR/SAL/SAR：移位指令，SHL/SAL左移，SHR/SAR右移，将操作数的二进制位进行移位操作，移动的位数由另一个操作数给出。**

**串操作指令：**

**MOVS：将一个字符串中的字节或字移动到另一个字符串中。**

**CMPS：比较两个字符串中对应的字节或字。**

**SCAS：在一个字符串中查找一个字节或字。**

**LODS：从一个字符串中取出一个字节或字，存放到寄存器中。**

**STOS：将一个字节或字存放到一个字符串中。**

**程序控制指令：**

**JMP：无条件转移指令，直接跳转到指定的目标地址。**

**条件转移指令：根据条件码寄存器的值，决定是否跳转到指定的目标地址。**

**LOOP：根据CX寄存器的值，决定是否跳转到指定的目标地址。**

**LOOPZ/LOOPE指令在CX计数器不为0且ZF标志位为1时，会将CX计数器减1并跳转到指定的地址处执行，否则继续执行下一条指令。这条指令通常用于在字符串处理中查找某个字符或字节，直到找到或者到达字符串结尾。**

**LOOPNZ/LOOPNE指令在CX计数器不为0且ZF标志位为0时，会将CX计数器减1并跳转到指定的地址处执行，否则继续执行下一条指令。这条指令通常用于在字符串处理中查找某个字符或字节，直到未找到或者到达字符串结尾。**

**CALL：调用子程序，保存返回地址并跳转到指定的目标地址。**

**INT：中断指令，触发特定的中断服务程序。**

**处理器控制指令：**

**CLD：清除方向标志位。**

**STD：设置方向标志位。**

**CLI：清除中断标志位。**

**STI：设置中断标志位。**

**HLT：停机指令，停止处理器执行。**

**CLC：清除进位标志位。**

**STC：设置进位标志位。**

1. **汇编语言程序设计**
2. 基本概念

1.指令与伪指令区别？

**指令是CPU所能识别的机器指令，它们在执行时能够直接操作硬件资源，包括执行算术逻辑操作、数据传输、分支跳转等。指令的编写方式是直接使用CPU指令集中的指令助记符，例如ADD、MOV等。指令的处理需要由汇编器将其翻译成相应的二进制机器码，然后由CPU执行。**

**伪指令则是汇编语言所特有的语句，不是CPU指令集中的机器指令，而是用来协助程序员组织和调整程序结构、定义符号常量、存储空间等的一类指令。伪指令的编写方式和指令类似，但是它们不能直接被CPU所执行，而是在汇编过程中被汇编器处理，并转化成实际的指令或数据。**

**因此，指令和伪指令的最大区别在于它们的处理方式和作用。指令是直接被CPU执行的机器指令，而伪指令则是汇编器处理的语句，用来辅助程序员组织和调整程序结构**

2.伪指令：段定义伪指令、数据定义伪指令：常数(字符串的使用DB、DW、DD)、变量和标号,当前位置计数器$（$!作变量使用，是16位偏移量）与定位伪指令ORG等伪指令。

**段定义伪指令用于定义程序中不同的数据和代码段，包括段的名称、段的属性等信息。**

**数据定义伪指令用于定义程序中的数据，包括常数和变量。常数可以使用DB（定义字节）、DW（定义字）、DD（定义双字）等伪指令定义。变量和标号可以使用DW（定义字）和DD（定义双字）等伪指令定义，变量定义的同时还可以给变量赋初值。**

**当前位置计数器$可以用作变量使用，它指向当前汇编指令的地址，是16位偏移量。**

**定位伪指令ORG用于指定汇编程序的起始地址，可以在程序中修改程序计数器的值，以便实现程序的定位。**

二、汇编语言程序设计

1.汇编语言源程序结构（重点：程序框架，必须记得）



2.汇编语言程序设计四种类型：顺序程序设计、分支程序设计、循环程序设计、子程序设计

**第6章 输入输出和中断技术**

一、基本概念

1.输人/输出系统主要由哪几个部分组成？主要有哪些特点？

**I/O 控制器：负责实际控制 I/O 设备的工作，包括数据传输、状态监测和控制命令的执行等。**

**I/O 接口电路：连接计算机系统和 I/O 控制器之间的数据通路，包括地址线、数据线、控制线等。**

**设备驱动程序：是在操作系统中运行的软件模块，负责控制 I/O 设备的工作，包括打开设备、关闭设备、读取数据、写入数据等操作。**

**I/O 系统的主要特点如下：**

**可编程性：I/O 控制器和接口电路都是由软件来控制的，因此可以根据具体的应用需要编写相应的软件程序。**

**数据传输方式多样：I/O 系统支持多种数据传输方式，如程序查询方式、中断方式、DMA 方式等，以适应不同的应用场合。**

**设备独立性：I/O 系统可以支持不同的设备类型和品牌，而不需要对软件程序进行修改。**

**异步性：I/O 设备和 CPU 的工作时钟不同步，因此需要采用一定的同步措施，以保证数据传输的正确性和可靠性。**

**可靠性：I/O 系统对数据的传输过程进行了严格的控制和管理，以确保数据传输的正确性和完整性，同时还可以进行错误检测和纠正。**

**高效性：I/O 系统的设计旨在提高数据传输的效率和速度，以满足高性能计算机系统的需求。**

2.I/0接口的主要功能有哪些？有哪两种编址方式？在8088/8086系统中采用哪一种编址方式？

**（1）数据的缓冲与暂存；信号电平和类型的转换；增加信号的驱动能力；对外设进行检测、控制与管理、中断处理**

**I/O地址译码：I/O接口需要将CPU发出的I/O指令地址进行解码，确定具体的外部设备。**

**数据缓冲：I/O接口需要提供数据缓冲区，将外部设备传来的数据进行暂存，以便CPU进行数据读取或写入。**

**控制信号生成：I/O接口需要产生各种控制信号，控制外部设备的工作方式，如启动、停止、读写、中断等。**

**数据格式转换：I/O接口需要进行数据格式的转换，将外部设备传来的数据转换为计算机内部所需的格式。**

**中断处理：I/O接口需要监控外部设备发出的中断信号，并在发生中断时将中断请求通知CPU，以便CPU及时响应。**

**DMA控制：I/O接口需要控制DMA传输，使得外设和主存之间的数据传输可以不经过CPU的中间处理，提高数据传输的速度。**

**（2）与内存统一编址，单独编址**

**（3）8088/8086采用I/O端口独立编址**

3.试比较 4种基本输入/输出方法的特点。

**（1）无条件传送：要求外设总是处于准备好状态。适用于简单外设**

**（2）查询式传送：每满足一次条件只能进行一次数据传送。适用于外设并不总是准备好，对传送速率和效率要求不高**

**（3）中断方式传送：外设在需要时向CPU提出请求，CPU再去为它服务。服务结束后或在外设不需要时，CPU可执行自己的程序。CPU效率高，实时性好，速度快。**

**以上三种：信息的传送均需通过CPU**

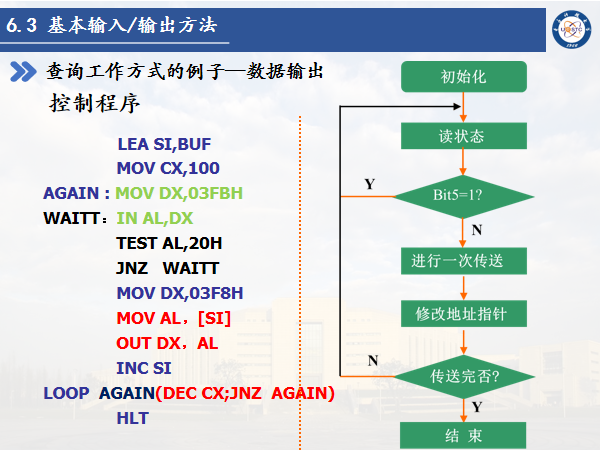
**（4）直接存储器存取：总线由DMA控制器（DMAC）进行控制（CPU要放弃总线控制权），内存/外设的地址和读写控制信号。CPU不再担当数据传输的中介者**

4.主机与外部设备进行数据传送时,采用哪一种传送方式CPU的效率最高？

**DMA**

二、逻辑电路及程序设计（重点）

1.根据状态端口地址查询相关状态位的值，符合条件通过数据端口传送一次数据。



2.中断向量表中中断向量设置。



