# 第2章 数据表示和运算

## 定点数

1. 机器数是补码。进行二进制运算前，务必先将真值写成原码，再转换成补码，代入运算。
2. 算术位移，正数左右移都添0，负数的补码右移添1。

逻辑位移，当作无符号数。

循环位移，带进位标志位CF的、不带CF的。

1. 强制类型转换： int 🡪 unsigned int 机器数不变，解释不同

short 🡪 int 高位扩展，机器数是补码，负数添1

1. 溢出判断： CF 异或 C，CF为符号位进位，C为最高数字位进位

直接对比结果和中间数的符号，溢出逻辑

(实列参考PTA的A1065)

双符号位(模4补码)看两个符号位是否一致

1. 边界对齐：找题干中条件：按X编址，字长X位。存一个数据至少用一个字或半个字。大端方式，数据的高位(最左)对齐。比如按字节编址，采用小端对齐，从地址0xC004开始连续存储一个64位double a= 0xFF2301A200EC58FF，8位的char b = 0x19和16位的short c = 0x3323，则存储示意图如下：

地址 C004 C005 C006 C007

数据 FF 58 EC 00

地址 C008 C009 C00A C00B

数据 A2 01 23 FF

地址 C00C C00D C00E C00F

数据 19 - 23 33

## 浮点数

1. IEEE754：1+8+23，1+11+52。1位数符，阶码=真值-1+128，尾数用原码，隐藏最高位“1”，运算时需要写出“1”。几个特殊值+∞，机器0

例：尾数右移3位 .011… 🡪 0.001011… 然后用0.001011… 进行原码运算

1. 十进制数与IEEE754浮点数转换
2. IEEE754浮点数加减运算，重点：右规舍入，机器数变化
3. 浮点数的表示范围，最大正数，最小负数，最大负数，最小正数之类
4. 规范化，原则：最高位必须是一个有效值

补码尾数：

原码尾数：

1. 强制类型转换：int 🡪 float：int 32位，float尾数23位，精度降低。而且0舍1入，机器数发生改变

# 第3章 存储系统

## 存储芯片

1. ROM：EEPROM、Flash、SSD

RAM：SRAM不用刷新。DRAM三种刷新，**地址复用**

1. 字扩展，位扩展

解题思路：需要的地址数 × 数据单元位数 / 芯片规格(比如16K×8)

1. 低位交叉编址：存储周期是总线周期的m倍，m个存储模块流水作业

## Cache

1. 通常主存按字节编址，即一个数据单元存1B。几个单元存一个字，所以才有边界对齐。CPU对主存的访问和交换数据以字为单位。Cache和主存交换数据以块为单位。
2. 经典题型：P124.13 代码具体执行中读写次数。P125.3
3. X路组相联：组间直接映射，组内全相联映射，**一组X块**
4. Cache中每块都有标记阵列，有效位、脏位、替换控制位、标记位
5. 替换算法：RAND、FIFO、LRU、LFU

全相联、组相联的组内才有替换

1. Cache写策略：写命中时，全写法(两个都写)、写回法(回过头再写)

写不命中时，写分配法(调块写cache)、非写分配法(只写主存)

1. 物理地址中不同段的作用的正确描述：32位的物理地址中，低6位是块内地址，中间4位是Cache组号，高22位是Tag标记。

# 第4&5章 指令与中央处理器

1. 常见的寻址方式：隐含寻址、立即数寻址、直接寻址 🡪 寄存器寻址、间接寻址 🡪 寄存器间接寻址、相对寻址(PC)、基址寻址、变址寻址、堆栈寻址
2. 相对寻址广泛用于转移指令JMP X，若转移指令占2个字节，按字节编址，则取出指令JMP X后PC+2，然后再加上X。
3. 在手推寻址过程时，使用有效地址EA作为中间值。如果遇到嵌套型的，比如形式地址先变址后间址，变址的EA = A + (IX)，间址的EA = (A)，则嵌套后EA = (A + (IX))，这样不容易出错。
4. 精简指令系统RICS中，指令长度固定；只有取数/存数指令可以访存；通用寄存器数量很多；硬布线控制器为主。
5. 在求地址偏移量时，偏移量 = 目标地址 – 现在的地址。
6. 机器字长 = 数据总线位数 = 通用寄存器位数，机器字长是CPU一次能处理的数据位数，可能大于等于MDR位数。存储字长 = MDR位数

MAR位数 = 程序计数器PC位数 = log2(主存容量)

指令寄存器IR位数 = 指令字长，一般是存储字长的整数倍

1. CPU的运算器包括ALU、ACC、暂存寄存器、通用寄存器、PSW、移位器、计数器

控制器包括PC、IR、MAR、MDR、时序系统、微操作信号发生器

1. 一个指令周期可以分为取址周期、间址周期(如果有的话)、执行周期、中断周期。这四部分都称为一个机器周期。一个机器周期又是若干个时钟周期。时钟周期就是主频的倒数，是计算机的最小时间单位。
2. 取址周期结束时，MDR中放的是指令。间址周期结束时，MDR中放的是操作数的有效地址。
3. 数据通路的题目以P192.1和P193.4为范本就行，不复杂的。遇到结构图想不通的地方比如P196.9，要想到PC+1也是用ALU来计算完成的这一点。
4. 指令的执行实质上是一系列微操作的执行。微命令即是微操作的控制信号。在硬布线控制器中微命令由逻辑电路来实现，而在微程序控制器中微命令由若干条微指令组成。一系列的微指令构成了一个微程序，而每一条机器指令对应一个微程序。
5. 微指令的字段直接编码方式：有多少种互斥类的微命令(一条指令中容不下2个同类)，一条指令中就有多少字段。若字段位数是n，该字段最多可以对应2n-1个微命令，空一个表示无操作。
6. 指令流水线冲突：结构相关即两指令同时争用同一资源；数据相关即是后一条指令要用到前一条指令的执行结果数据；控制相关即是流水线遇到跳转指令改变PC值。
7. 一个k段流水线完成n个任务所用的时钟周期为k+n-1
8. 详细的解题攻略P234.6分析部分有。遇到数据相关，后一条指令的ID必须在前一条的WB之后。如果流水线是按序发射的，后一条的IF不得早于前一条的ID。
9. 指令的机器代码就是指令在计算机中存储的形式。所以问到指令的某一部分是啥，直接将该指令由十六进制翻译为二进制，找出来就行。

# 第6章 总线

1. 突发(brust)模式：传送一次地址和n次数据，地址或数据的宽度=总线宽度。

传一次地址或数据通常用一个时钟周期。此时，一个总线周期为n+1个时钟周期。

# 第7章 I/O

1. I/O系统由外部设备和I/O接口两部分组成。I/O接口是在数据传输时在主机和外设间进行协调的逻辑部件。
2. I/O接口的功能：(1)实现主机和外设间的通信联络。(2)进行地址译码和设备选择。(3)实现数据缓冲。(4)信号格式的转换，比如电平、串并转换。(5)传送控制命令和状态信息。
3. I/O接口和I/O端口的区别。端口是接口电路中的读/写寄存器，用来实现数据缓冲。I/O端口加上相应的控制电路就是I/O接口。
4. I/O端口编址：统一编址方式，又称存储器映射方式，把端口当作主存单元分配地址。通过不同的地址码来区分端口还是主存，所以I/O地址要固定在地址的某部分(方便区分)。CPU不用专门的I/O指令，用访存指令即可访问I/O端口。独立编址方式，又称I/O映射方式，端口的地址与主存地址相互独立(两者可重复)，需要另设专门的I/O指令来访问I/O端口。
5. I/O控制方式：程序查询，程序中断，DMA，通道
6. 程序查询：CPU启动I/O后，停止当前的进程，插入执行查询程序，即不停查询I/O状态端口直到I/O工作完成。CPU与I/O是串行工作的，效率低下。

## 中断

1. 程序中断：计算机在运行中遇到某些异常或特殊请求，CPU暂停执行当前程序，转而处理遇到的情况。处理完毕后CPU自动返回原先程序的断点处继续运行。比如，CPU对外设发出一个读命令，当前进程A进入等待状态，CPU转去运行其它进程。外设将数据读入数据端口后给CPU发送中断请求。CPU收到中断请求后，保存现场，执行中断服务，将数据从I/O端口读入CPU中的寄存器，再存入主存。进程A转入就绪状态。
2. 内中断源自CPU执行指令内部的事件，可分为指令中断、硬件故障、软件中断。比如非法操作码、地址越界、缺页、算术溢出、除以0、自陷指令、访管指令。内中断完全无视屏蔽。
3. 外中断源自指令以外的事件发生，比如外设发出I/O结束的中断，时钟中断等。可进一步分为非屏蔽中断和可屏蔽中断。
4. 中断判优：多个中断源同时提出请求时，判优逻辑电路决定先响应哪个。硬件故障 > 软件中断，非屏蔽优于可屏蔽，DMA中断优于普通I/O中断，高速的实时的优于低速的普通的。
5. CPU响应中断的条件：(1)中断源有中断请求。(2)开中断即中断标志位IF=1。(3)当前指令执行阶段结束。
6. 中断处理过程主要是两步。第一步执行中断隐指令，第二步执行中断服务程序。

中断隐指令细分为三步。关中断，为了保护现场时不被新的中断所打断。保存断点即程序计数器PC。引出中断服务程序。

中断服务程序：(1)保存现场，包括程序状态字PSW，中断屏蔽字寄存器，其它寄存器。(2)开中断，此时允许更高级的中断得到响应。(3)进行中断服务，中断系统的核心，数据经历 外设🡪CPU🡪主存。(4)关中断。(5)恢复现场和屏蔽字。(6)开中断+中断返回，中断服务的最后一条指令通常是中断返回指令(和跳转不同)，返回原先程序的断点处。

1. 多重中断：又称中断嵌套，CPU执行中断服务的过程中，出现了优先级更高的中断，则暂停当前的中断程序转而处理新的中断请求。
2. 中断屏蔽和判优的关系：两者不是一件事。中断优先级是由硬件排队电路决定的(相当于天生的属性)。优先级影响中断响应次序。然而优先级较低的中断可以通过设置屏蔽字，使自己运行时不被新到的优先级更高的中断请求打断(人为的改命)。中断屏蔽能改变中断处理次序。

## DMA

1. DMA方式：在外设和内存间开辟一条直接数据通道。数据传送不再经过CPU。传输数据时，CPU可以省去现场保护再恢复的繁琐过程。效率高于程序中断。
2. DMA的传送方式：(1)停止CPU访存。完全DMA优先，抢占式。(2)DMA与CPU交替访存，一个机器周期前一半CPU访存后一半DMA访存，相当于时分复用。(3)周期挪用。DMA优先但非抢占式。DMA请求批准后，CPU给其分配1个总线周期(单字传送)或多个总线周期(成组连续传送)。
3. DMA工作流程：预处理、数据传送、后处理。预处理是由CPU完成的必要准备工作。CPU执行I/O指令调动外设，设置DMA控制器的参数。外设完成工作后向DMA控制器发送DMA请求，DMA控制器向CPU发送总线请求。开始传输数据后，数据传送的过程完全由DMA控制器实现，CPU不参与。结束后DMA控制器向CPU发送中断请求，CPU执行中断服务，包括数据校验和决定是否继续使用DMA传其它数据。
4. DMA与中断的区别：(1)DMA除了预处理和后处理，中间传输数据部分不占有CPU的资源，而中断方式中的数据传输过程是CPU的中断服务提供的。(2)CPU对中断请求的响应只能在每条指令执行阶段结束时。对DMA请求的响应可以发生在任一机器周期结束时，即没有总线事务时即可。DMA请求优先级高于中断请求。DMA方式下数据传送由硬件完成，中断下由软件完成。DMA仅限于数据传送，中断可用于异常事件处理。

## 通道

1. I/O通道时专门负责输入输出的处理机，是DMA方式的进一步发展，进一步减少CPU的参与。例如，当CPU要完成一组读写操作时，只需向I/O通道发送一条I/O指令，给出要执行的通道程序的首地址和要访问的I/O设备。通道收到指令后执行通道程序便可以完成CPU指定的任务。任务结束后向CPU发出中断请求。
2. 通道与CPU的区别：通道指令单一，通道没有自己的内存。
3. 通道与DMA的区别：DMA需要CPU控制传输的数据块大小、传输的内存位置。通道自行决定这些。每个DMA控制器只对应一台外设与内存传送数据，一个通道可以控制多台设备与内存交换。通道指令的编程灵活性适用性高于DMA的硬件方式。