存储系统

存储系统的基本要求

- 基本要求是大容量、高速度、低价格

存储器容量

存储器容量$S_M = W * l * m$

- W 为存储体的字长， 单位是位、字节
- l 为存储体的字数
- m 为并行工作的存储体数

存储速度

存储速度可以用访问时间T_A、存储周期 T_M 和 频宽（带宽）B_m描述

访问时间T_A

是存储器从接收访存读请求申请 到 信息倍读取到数据总线上的时间 是处理机启动访存后必须等待的时间 是确定处理机与存储器时间关系的重要参数

存储周期T_M

是连续启动一个存储体所需要的时间间隔一般比Ta大

存储器频宽B_m

- 是存储器可提供的数据传送速率
- 用每秒传送的信息位数或字节数衡量
- 有最大频宽和实际频宽之分,最大频宽是是存储器连续访问时的频宽
- 存储器频度 Bm=W/Tm(存储体字长/访问时间)
- m个存储体并行的最大频度 $Bm = W * m/Tm$

存储器容量、速度、价格关系

- 容量越大，因其延迟增大会使速度越低
- 容量越大，总价格越高
- 存储器速度越快，价格越高

使用并行系统的原因

使用单一工艺无法同时满足容量，速度与价格的要求。因此系统中必须同时采用多种不同工艺存储器组成的存储器系统（memory system）例如：主存+辅存

为了弥补CPU与存储器在速度上的差距，引入了并行与重叠技术构成并行主存系统，在保存每位价格基本不变的情况下，是主存的频宽得到较大的提高。但是这种方法提供频宽有限

并行主存系统

单体单字存储器

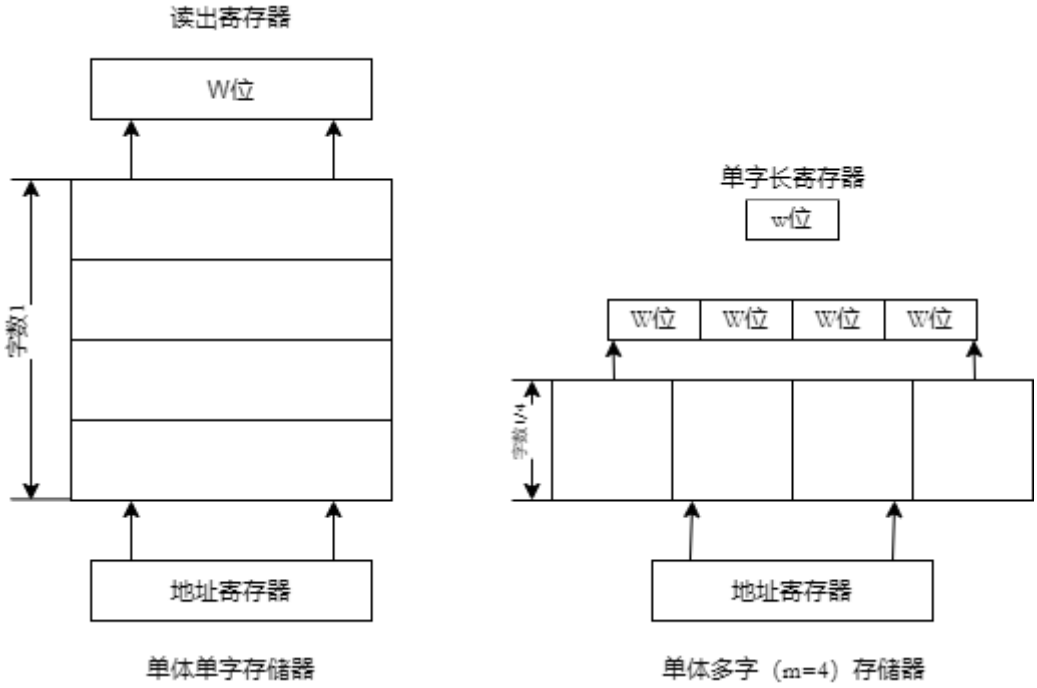
一个字长位W位的单体主存，一次可以访问一个存储器字 存储器的字长W与CPU要访问的字的字长W相同，则CPU从主存获得数据的速度就是$B_m = W/T_M$

单体多字存储器

一个字长为W的单体主存，其字长是CPU字的倍数（相当于一次读取多个字）。

一次可以读取多个CPU字。

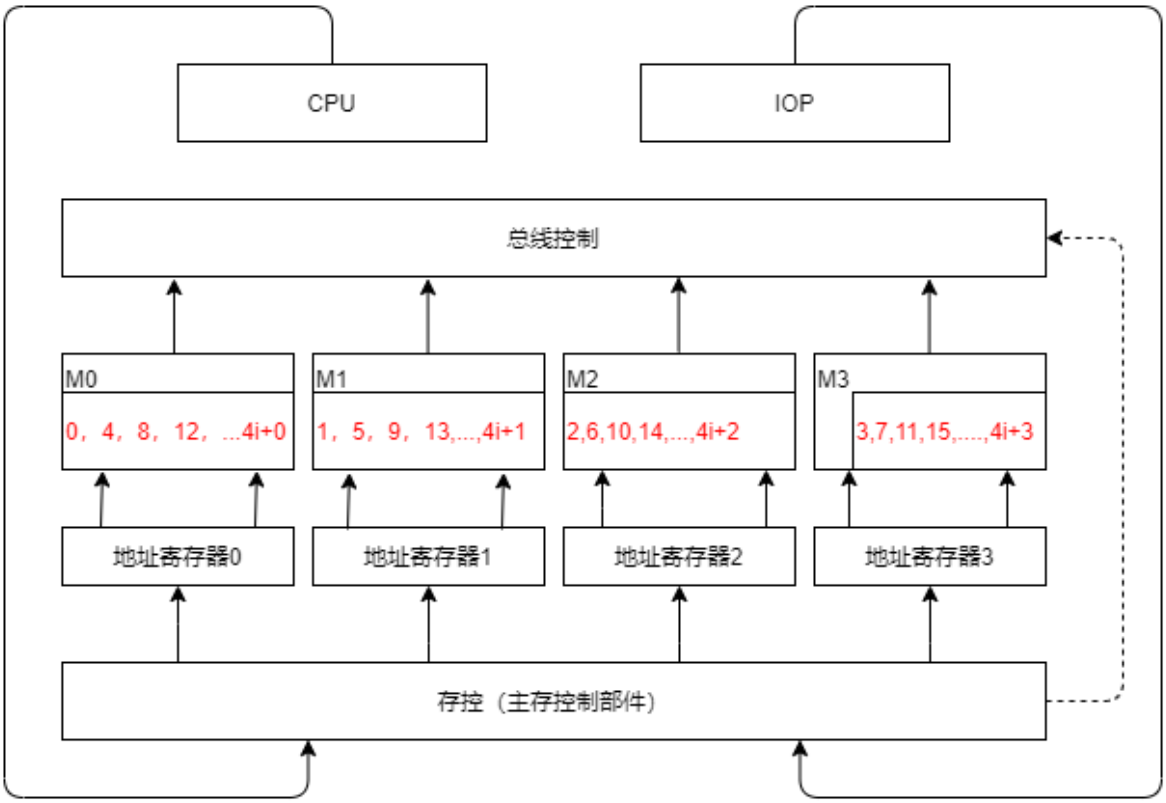
> 将单体单字扩展为单体多字



多体单字交叉存储器

一个大容量的半导体主存往往是由多个容量较小、字长较短的存储器芯片（内存颗粒)组搭而成的，每个存储芯片都由其自己的地址译码、读/写驱动等外围电路。因此可以采用多体单字存储。

CPU字在主存中按模m交叉编址，根据应用特点分为 `低位交叉` 和 `高位交叉`



以低位交叉为例，其m在单体多字方式中为一个主存字所包含的CPU字数，在多体单字方式中则为分体体数。

以多体单字交叉为例，单体容量为 l 的多个 m 个分体，其 M_j 体的编制模式为 $m*i+j$ 。其中 $i=,1,2,\ldots,l-1, j=0,1,2,\ldots,m-1$。如上图，如果多体存储器的体数为4， 存储体$M_0$中的编址序列为 $4*i+0$,存储体M_1中的编址序列为 $4*i+1$,存储体M_2中的编址序列为 $4*i+2$,存储体M_3中的编址序列为 $4*i+3$。

各分体地址对应二进制码最末尾两位的状态

M_0 的地址最后两位是 00, $4*i+0$, 2的倍数

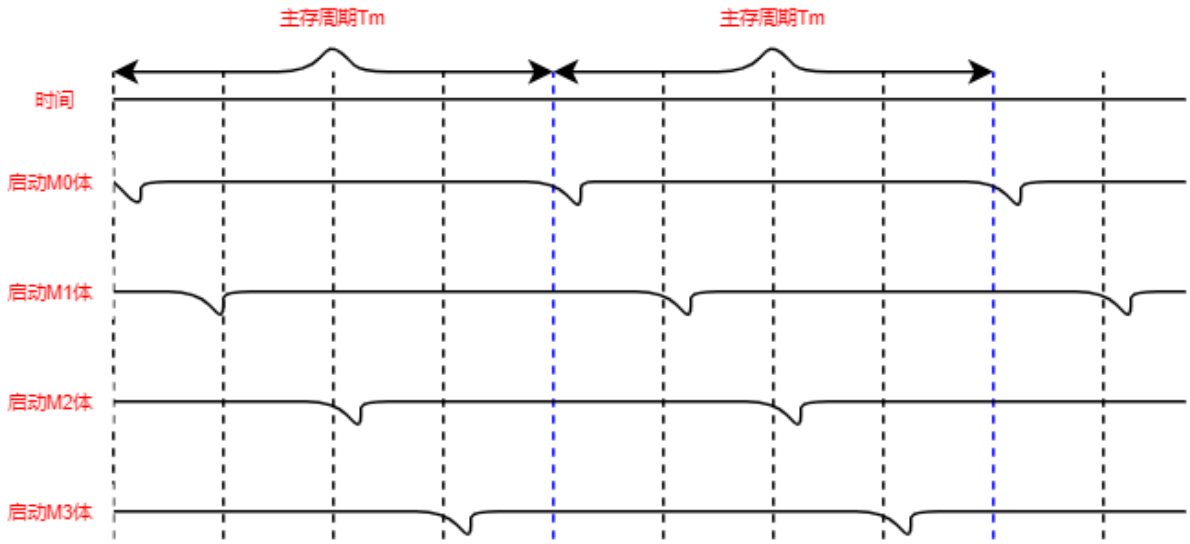
M_1 的地址最后两位是 01, $4*i+1$, 1=01

M_2 的地址最后两位是 10, $4*i+2$, 2=10

M_3 的地址最后两位是 11, $4*i+3$, 3=11

各分体的工作方式

各分体的工作方式可以采用 `同时启动` 或 `分时启动` 方式工作。相对而言，分时启动方式所用的硬件较节省。



并行主存系统

能并行读出多个CPU字的单体多字和多体单字、多体多字交的交叉访问主存系统称作并行主存系统。

提高模 m 的值（体数）是能提高主存系统的最大频宽的，但是实际频宽并不是随 m 值增大而线性提高。也就是说实际效率并不是希望的那么高。原因是：

1. 系统效率问题
    - 对模m交叉，若都是顺序取指，效率是可以提高到m倍的。
    - 但实际程序中指令不总是顺序执行，一旦出现转移效率就会下降。
    - 转移的频度越高，并行主存系统效率的下降越大
    - 数据的顺序性比指令的要差，实际的频宽可能还要低些
2. 工程实现上
    - 在工程实现上由于模 m 的越高，存储器数据总线越长，总线上并联的负载越重。
    - 有时还不得不增加门的级数，这些都会使传输延迟增加

定量分析主存频宽与分体数m、转移概率λ的关系

对有m个独立分体的主存系统，CPU发出的访存申请队为：$A_1, A_2, \ldots, A_q$。

- 在每一个主存周期到来之前扫描申请队，并从对头截取序列$A_1, A_2, \ldots, A_k$作为申请序列。
- 申请序列是在要求访存的k个地址中没有两个或以上的地址在同一个分体中的最长序列。
- 即：A_1到A_k的地址不一定是顺序编址，只要它们之间不发生分体冲突。显然k是随机变量，$1 \leq k \leq m$。
- k 最大可以到m，但实际通常小于m。且k表示了可以并行访问k个分体。因此该系统的效率取决于k的平均值

即每一个地址都在不同的分体中，就可以并行访问这些分体中的数据。

设$P(k)$表示申请序列长度为k的概率，其中$k = 1, 2, \ldots, m$。k的平均值用B表示， 则：

$$B = \sum_{k=1}^{m} k * P(k)$$

B实际上就是每个主存周期所能访问到的`平均字数`，正比与主存`实际频宽`，只差一个常数比值 T_M/W

$P(k)$ 与程序密切相关。如果访存申请队都是指令的话，那么影响最大的就是`转移概率`

- 转移概率

- 是给定指令的下条指令地址为非顺序地址的概率。

指令在程序中一般是顺序执行的，但是如遇转移成功，则申请序列中在转移指令之后的，与它在同一存储周期内读取出的其他顺序单元内容就没用了。

如果第一条指令就是转移指令且转移成功， 与第一条指令并行读取出的其他 $m-1$ 条指令就是没用的，相当于 $k=1$所以 $P(1)=\lambda=(1-\lambda)^0*\lambda$

第二条指令是转移指令且转移成功的概率就是第一条没有转移的概率， 即$1-\lambda$。

第一条指令是转移指令且转移成功，即 $k=1, P(1)=\lambda=(1-\lambda)^0*\lambda$

第二条指令是转移指令且转移成功，即 $k=2, P(2)=(1-P(1))*\lambda=$ （$1-\lambda$）$^1*\lambda$

第三条指令是转移指令且转移成功，即 $k=3, P(3)=(1-P(1)-P(2))*\lambda=(1-\lambda)^2*\lambda$

第k条指令是转移指令且转移成功，即 $k=m, P(k)=(1-\lambda)^{k-1}*\lambda$ ，其中$1\leq k\leq m$

所以：$P(m)=(1-\lambda)^{m-1}*\lambda$

$$B=\sum_{i=0}^{m}(1-\lambda)^i$$

这是一个等比级数，因此：

$$B=\frac{1-(1-\lambda)^m}{\lambda}$$

由此得出：

若每条指令都是转移指令且成功，即$\lambda=1$时，$B=1$ 即k的平均长度为1。意味着使用并行多体交叉存储的实际频宽低到与使用单体单字的一样。

若所有指令都不转移,即$\lambda=0$, $B=m$ 即k的平均长度为m。意味着使用并行多体交叉存储的效率最高。

【例3-1】设，访存申请队的转移概率$\lambda=25\%$，比较在模32和模16的多体单字交叉存储器中，每个周期能访问到的平均字数

根据每个存储周期能访问到的平均字数 $B=\frac{1-(1-\lambda)^m}{\lambda}$

将$\lambda=25\%, m=32$带入，可求得：

$B=\frac{1-0.75^{32}}{0.25}=4$

将$\lambda=25\%, m=16$带入，可求得：

$B=\frac{1-0.75^{16}}{0.25}=3.96$