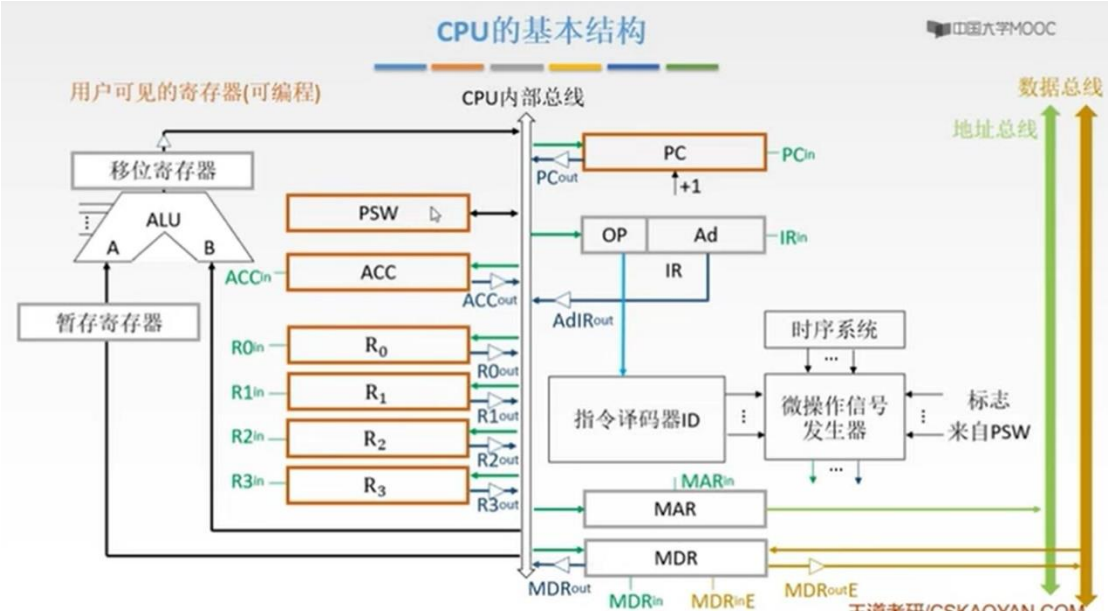
IPS = f/CPI

每个字中最小的字节地址为字地址

(PC) + A = 2002H + A = 1F00H

A = 1F00H – 2002H = 1EFFH – 2002H +1H = FEFDH+1H = FEFEH



指令周期：CPU从主存中每取出并执行一条指令所需的全部时间。

指令周期常常用若干**机器周期**来表示，机器周期又叫**CPU周期**。

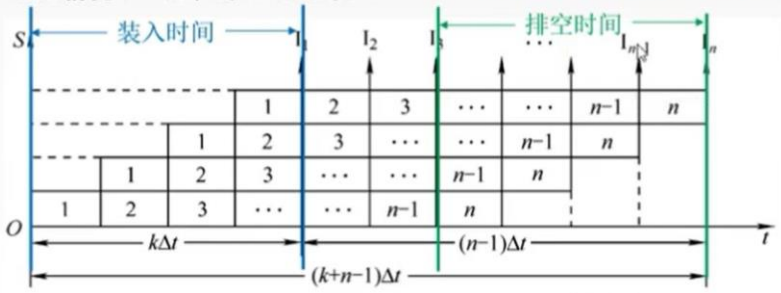
一个**机器周期**又包含若干**时钟周期**（也称为**节拍、T周期**或**CPU时钟周期**，它是CPU操作的**最基本单位**）。

1. 吞吐率　吞吐率是指在单位时间内流水线所完成的任务数量，或是输出结果的数量。

设任务数为n；处理完成n个任务所用的时间为T_k

则计算流水线吞吐率（TP）的最基本的公式为 $\mathrm{TP}=\dfrac{n}{T_k}$

理想情况下，流水线的时空图如下：　当连续输入的任务$n\to\infty$时，得最大吞吐率为$\mathrm{TP}_{max}=1/\Delta t$。



$T_k=(k+n\text{-}1)\,\Delta t$

流水线的实际吞吐率为

$$\mathrm{TP}=\frac{n}{(k+n-1)\Delta t}$$

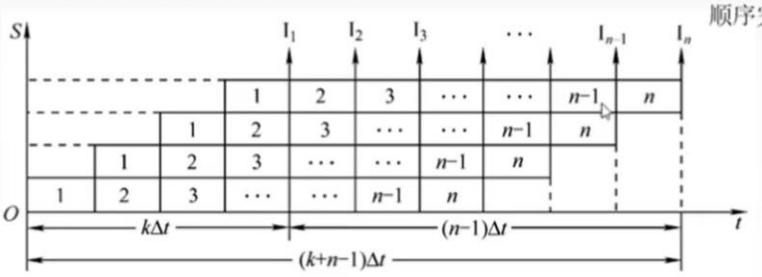
一条指令的执行分为k个阶段，每个阶段耗时Δt，一般取Δt=一个时钟周期

2. 加速比　完成同样一批任务，**不使用流水线所用的时间**与**使用流水线所用的时间**之比。

设T_0表示不使用流水线时的执行时间，即顺序执行所用的时间；T_k表示使用流水线时的执行时间

则计算流水线加速比（S）的基本公式为 $S=\dfrac{T_0}{T_k}$　当连续输入的任务$n\to\infty$时，最大加速比为$S_{max}=k$。

理想情况下，流水线的时空图如下：



单独完成一个任务耗时为$k\,\Delta t$，则顺序完成n个任务耗时$T_0=nk\,\Delta t$

$T_k=(k+n\text{-}1)\,\Delta t$

实际加速比为

$$S=\frac{kn\Delta t}{(k+n-1)\Delta t}=\frac{kn}{k+n-1}$$

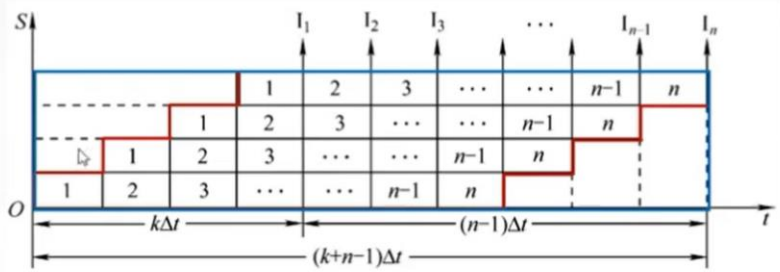
一条指令的执行分为k个阶段，每个阶段耗时Δt，一般取Δt=一个时钟周期

3. 效率　流水线的设备利用率称为流水线的效率。

在时空图上，流水线的效率定义为**完成n个任务占用的时空区有效面积**与**n个任务所用的时间与k个流水段所围成的时空区总面积**之比。

则流水线效率（E）的一般公式为 $E=\dfrac{n\text{个任务占用}k\text{时空区有效面积}}{n\text{个任务所用的时间与}k\text{个流水段所围成的时空区总面积}}=\dfrac{T_0}{kT_k}$

理想情况下，流水线的时空图如下：



当连续输入的任务$n\to\infty$时，最高效率为$E_{max}=1$。

一条指令的执行分为k个阶段，每个阶段耗时Δt，一般取Δt=一个时钟周期

数据的基本操作：读（R）、写（W）

冲突的基本类型：RAW、WAR、WAW

RAW　　注：“按序发射，按序完成”时，只可能出现RAW相关。

I1：ADD R5，R2，R4；(R2)+(R4) -> R5

I2：ADD R4，R5，R3；(R5)+(R3) -> R4

WAR

I1: STA M，R2;　　(R2) -> M,M为主存单元　　乱序发射，编写程序的时候希望I1在I2前完成，

I2: ADD R2，R4，R5;　　(R4)+(R5) ->R2　　但优化手段导致I2在I1前发射。

WRW

I1: MUL R3，R2，R1;　　(R2)*(R1)->R3　　存在多个功能部件时，后一条指

I2: SUB R3，R4，R5;　　(R4)-(R5)->R3　　令可能比前一条指令先完成。

1. 部件功能级、处理机级和处理机间级流水线

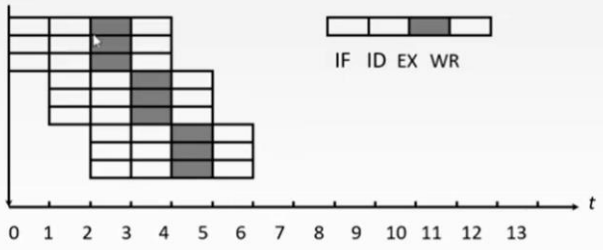
根据**流水线使用的级别**的不同，流水线可分为部件功能级流水线、处理机级流水线和处理机间流水线。

部件功能级流水就是将复杂的算术逻辑运算组成流水线工作方式。例如，可将浮点加法操作分成求阶差、对阶、尾数相加以及结果规格化等4个子过程。

处理机级流水是把一条指令解释过程分成多个子过程，如前面提到的取指、译码、执行、访存及写回5个子过程。

处理机间流水是一种宏流水，其中每一个处理机完成某一专门任务，各个处理机所得到的结果需存放在与下一个处理机所共享的存储器中。

1. 超标量技术



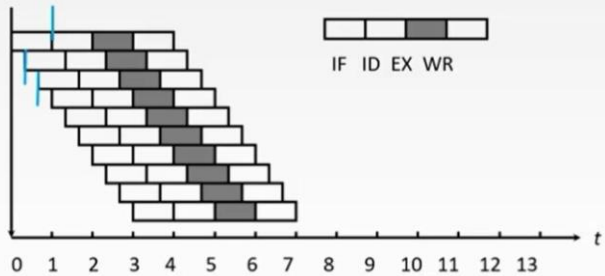
每个时钟周期内可 并发多条独立指令

要配置多个功能部件

不能调整 指令的 执行顺序

通过编译优化技术，把可并行执行的指令搭配起来

2. 超流水技术



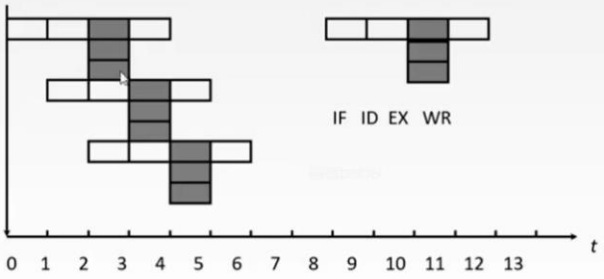
在 一个时钟周期 内 再分段 （3 段）

在一个时钟周期内 一个功能部件使用多次（3 次）

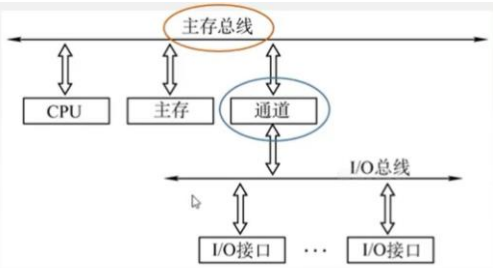
不能调整 指令的 执行顺序

靠编译程序解决优化问题

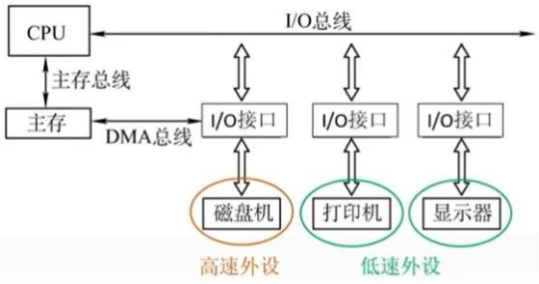
3. 超长指令字



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）

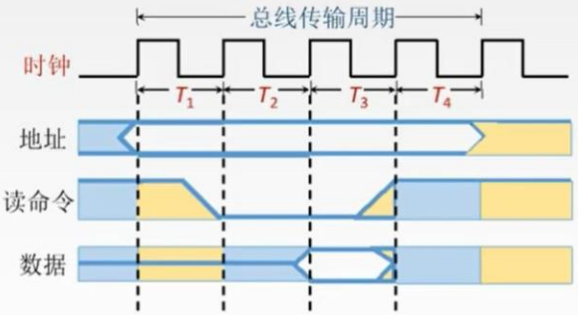


- **结构：**双总线结构有两条总线，一条是**主存总线**，用于CPU、主存和通道之间进行数据传送；另一条是**I/O总线**，用于多个外部设备与通道之间进行数据传送。
- **优点：**将较低速的I/O设备从单总线上分离出来，实现存储器总线和I/O总线分离。
- **缺点：**需要增加通道等硬件设备。



1. 总线的**传输周期**(**总线周期**)
   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），通常由若干个总线时钟周期构成。

2. 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。



3. 总线的**工作频率**
   总线上各种操作的频率，为**总线周期的倒数**。
   若总线周期=N个时钟周期，则总线的工作频率=时钟频率/N。
   实际上指**一秒内传送几次数据**。

4. 总线的**时钟频率**
   即机器的时钟频率，为**时钟周期的倒数**。
   若时钟周期为T，则时钟频率为1/T。
   实际上指**一秒内有多少个时钟周期**。

总线带宽 = 总线工作频率 × 总线宽度 （bit/s）= 总线工作频率 × (总线宽度/8) （B/s）

$$= \frac{\text{总线宽度}}{\text{总线周期}}\ \text{（bit/s）} = \frac{\text{总线宽度/8}}{\text{总线周期}}\ \text{（B/s）}$$

仲裁方式 / 对比项目	链式查询	计数器定时查询	独立请求
控制线数	3 总线请求：1 总线允许：1 总线忙：1	$\lceil \log_2 n \rceil + 2$ 总线请求：1 总线允许：$\lceil \log_2 n \rceil$ 总线忙：1	2n+1 总线请求：n 总线允许：n 总线忙：1
优点	优先级固定 结构简单，扩充容易	优先级较灵活	响应速度快 优先级灵活
缺点	对电路故障敏感 优先级不灵活	控制线较多 控制相对复杂	控制线多 控制复杂

总线周期的四个阶段

1）申请分配阶段：由需要使用总线的主模块（或主设备）提出申请，经总线仲裁机构决定将下一传输周期的总线使用权授予某一申请者。也可将此阶段细分为**传输请求**和**总线仲裁**两个阶段。

2）寻址阶段：获得使用权的主模块通过总线**发出**本次要访问的从模块的**地址**及有关**命令**，启动参与本次传输的从模块。

3）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。

4）结束阶段：主模块的**有关信息**均从系统总线上**撤除**，让出总线使用权。

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，它的实质是一种协议或规则

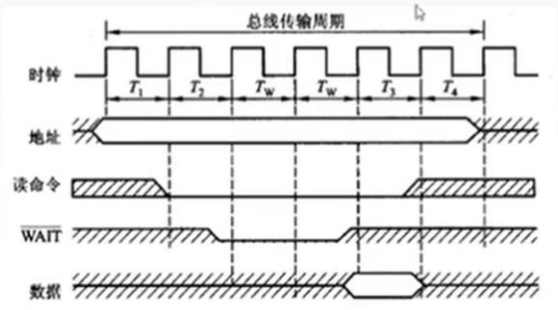
- 同步通信(同步定时方式)：由 统一时钟 控制数据传送
- 异步通信(异步定时方式)：采用 应答方式，没有公共时钟标准
- 半同步通信：同步、异步结合
- 分离式通信：充分 挖掘 系统总线每瞬间 的 潜力

同步　发送方 用系统 时钟前沿 发信号

接收方 用系统 时钟后沿 判断、识别

异步　允许不同速度的模块和谐工作

半同步通信：统一时钟的基础上，增加一个“等待”响应信号 $\overline{\text{WAIT}}$



上述三种通信的共同点

一个总线传输周期（以输入数据为例）

- 主模块发地址、命令　　使用总线
- 从模块准备数据　　不使用总线　　总线空闲
- 从模块向主模块发数据　　使用总线

分离式通信的一个总线传输周期

子周期1　主模块申请占用总线，使用完后放弃总线的使用权

子周期2　从模块申请占用总线，将各种信息送至总线上

特点：
1. 各模块均有权申请占用总线
2. 采用同步方式通信，不等对方回答
3. 各模块准备数据时，不占用总线
4. 总线利用率提高

VGA：Video Graphics Array，也称为D-sub端口　　传输模拟信号
CRT显示器，模拟信号：数字信号—>模拟信号—>VGA—>CRT
LCD液晶显示器，数字信号：模拟信号—>VGA—>数字信号—>LCD
模拟信号在超过1280×1024分辨率—>转换损耗明显

DVI：Digital Visual Interface
传输数字信号
但在分辨率1024×768以下时与VGA差别不大

HDMI：High Definition Multimedia Interface
理论最大传输速度可达Gb/s
影像数据+8声道的音讯信号
源于DVI技术
三种类型
A型：高清电视，投影仪等
C型：平板电脑，MP4等
D型：智能手机，平板电脑等

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 × 灰度级位数

2.磁盘的性能指标

① 磁盘的容量：一个磁盘所能存储的字节总数称为磁盘容量。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以利用的磁化单元总数。
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。

② 记录密度：记录密度是指盘片单位面积上记录的二进制的信息量，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。

注意：磁盘所有磁道记录的信息量一定是相等的，并不是圆越大信息越多，故每个磁道的位密度都不同。

③ 平均存取时间：

平均存取时间 = 寻道时间（磁头移动到目的磁道）+
旋转延迟时间（磁头定位到所在扇区）+
传输时间（传输数据所花费的时间）

④ 数据传输率：磁盘存储器在单位时间内向主机传送数据的字节数，称为数据传输率。

假设磁盘转数为r（转/秒），每条磁道容量为N个字节，则数据传输率为$D_r=rN$

RAID的分级如下所示。在RAID1～RAID5的几种方案中，无论何时有磁盘损坏，都可以随时拔出受损的磁盘再插入好的磁盘，而数据不会损坏。

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，不仅扩大了存储容量，而且提高了磁盘数据存取速度，但RAID0没有容错能力。

RAID1是为了提高可靠性，使两个磁盘同时进行读写，互为备份，如果一个磁盘出现故障，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。

CPU响应中断必须满足以下3个条件：

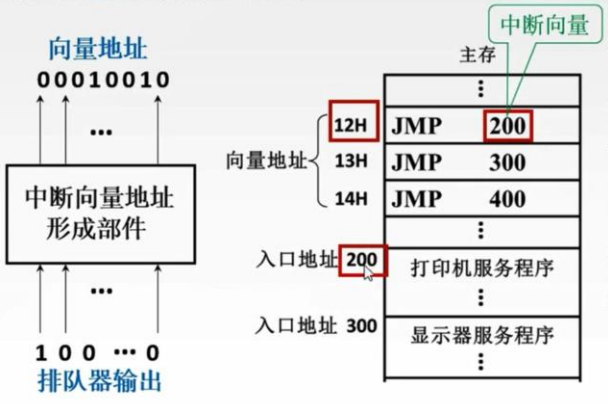
① 中断源有中断请求。

② CPU允许中断即开中断。

③ 一条指令执行完毕，且没有更紧迫的任务。

由 硬件 产生 向量地址

再由 向量地址 找到 入口地址



中断隐指令的主要任务：

① **关中断**。在中断服务程序中，为了保护中断现场（即CPU主要寄存器中的内容）期间不被新的中断所打断，必须关中断，从而保证被中断的程序在中断服务程序执行完毕之后能接着正确地执行下去。

② **保存断点**。为了保证在中断服务程序执行完毕后能正确地返回到原来的程序，必须将原来程序的断点（即程序计数器（PC）的内容）保存起来。可以存入堆栈，也可以存入指定单元。

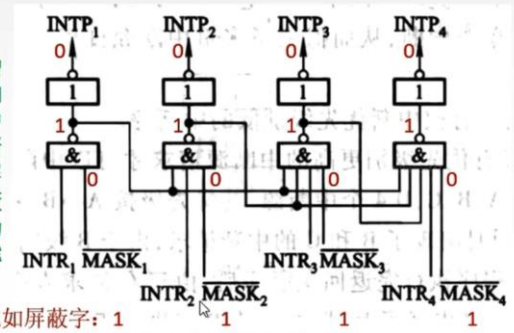
③ **引出中断服务程序**。引出中断服务程序的实质就是取出中断服务程序的入口地址并传送给程序计数器（PC）。

- 软件查询法
- 硬件向量法

	单重中断	多重中断
中断隐指令	关中断	关中断
	保存断点（PC）	保存断点（PC）
	送中断向量	送中断向量
中断服务程序	保护现场	保护现场和屏蔽字
	-	开中断
	执行中断服务程序	执行中断服务程序
	-	关中断
	恢复现场	恢复现场和屏蔽字
	开中断	开中断
	中断返回	中断返回

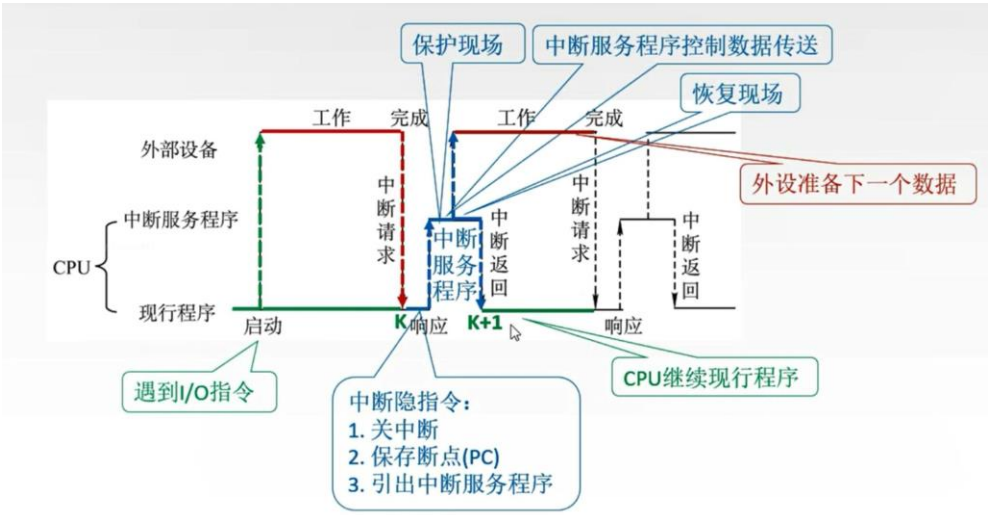
屏蔽字设置的规律：
1. 一般用‘1’表示屏蔽，‘0’表示正常申请。
2. 每个中断源对应一个屏蔽字(在处理该中断源的中断服务程序时，屏蔽寄存器中的内容为该中断源对应的屏蔽字)。
3. 屏蔽字中‘1’越多，优先级越高。每个屏蔽字中至少有一个‘1’(至少要能屏蔽自身的中断)。

增加中断屏蔽功能



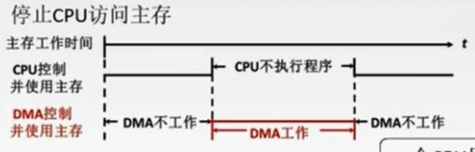
比如屏蔽字：1 1 1 1

调整多重中断的优先级



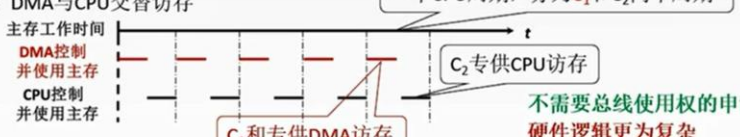
主存和DMA控制器之间有一条数据通路，因此主存和I/O设备之间交换信息时，不通过CPU。但当I/O设备和CPU同时访问主存时，可能发生冲突，为了有效地使用主存，DMA控制器与CPU通常采用以下3种方法使用主存。

（1）停止CPU访问主存



控制简单
CPU 处于不工作状态或保持状态
未充分发挥 CPU 对主存的利用率

（2）DMA与CPU交替访存



一个CPU周期，分为C_1和C_2两个周期

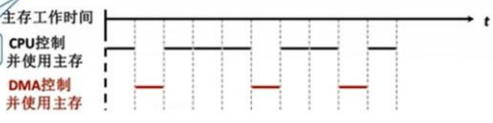
C_2专供CPU访存

不需要总线使用权的申请、建立和归还过程
硬件逻辑更为复杂

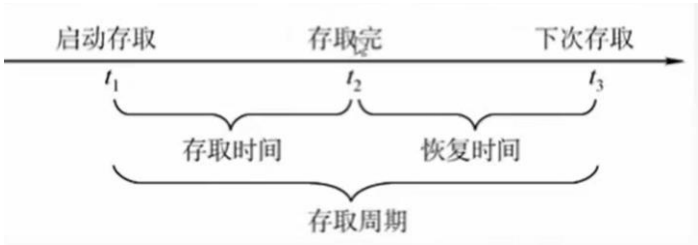
C_1和专供DMA访存

（3）周期挪用(周期窃取)

存取周期



DMA 访问主存有三种可能:
CPU 此时不访存（不冲突）
CPU 正在访存（存取周期结束让出总线）
CPU 与 DMA 同时请求访存（I/O访存优先）



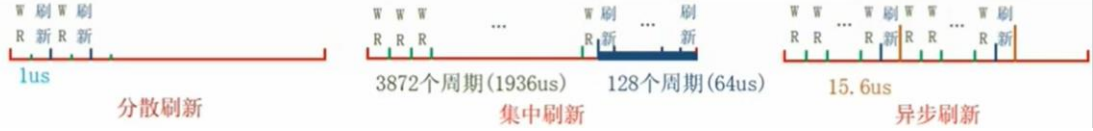
1. 多久需要刷新一次？ 刷新周期：一般为2ms
2. 每次刷新多少存储单元？以行为单位，每次刷新一行存储单元
   ——为什么要用行列地址？减少选通线的数量
3. 如何刷新？ 有硬件支持，读出一行的信息后重新写入，占用1个读/写周期
4. 在什么时刻刷新？

存取周期
假设DRAM内部结构排列成128×128的形式，读/写周期0.5us
2ms共 2ms/0.5us = 4000 个周期

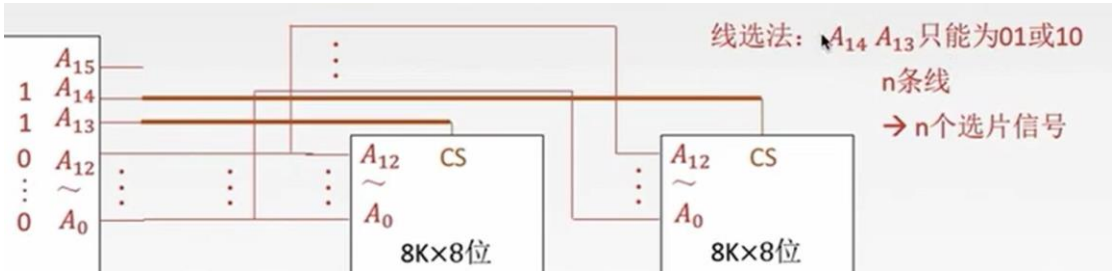
思路一：每次读写完都刷新一行
→系统的存取周期变为1us
前0.5us时间用于正常读写
后0.5us时间用于刷新某行

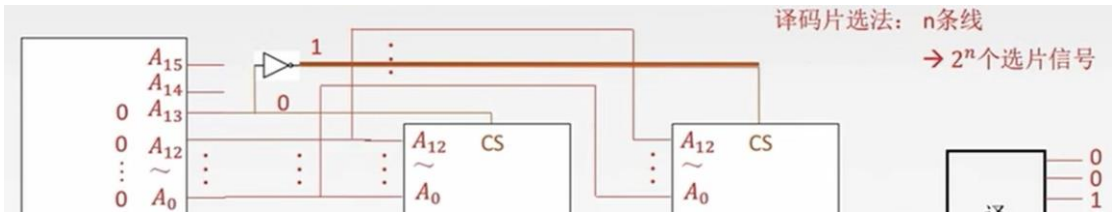
思路二：2ms内集中安排时间全部刷新
→系统的存取周期还是0.5us
有一段时间专门用于刷新，
无法访问存储器，称为访存“死区”

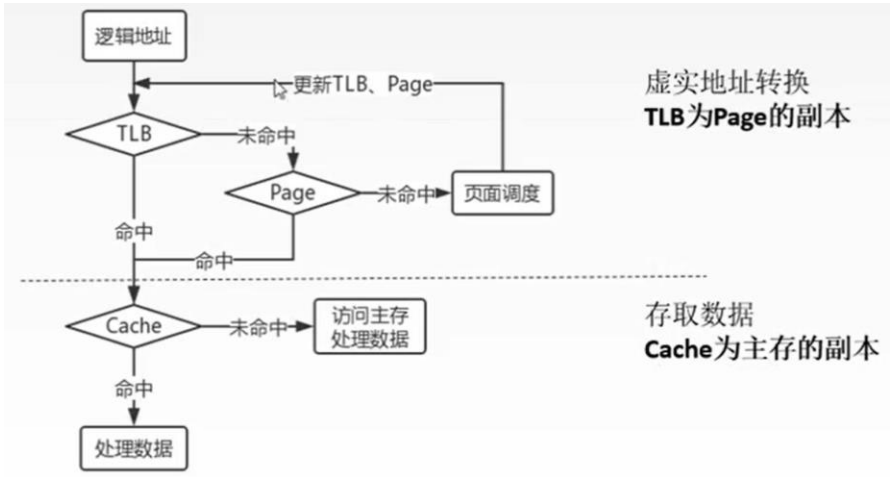
思路三：2ms内每行刷新1次即可
→2ms内需要产生128次刷新请求
每隔2ms/128 = 15.6us 一次
每15.6us内有0.5us的“死时间”

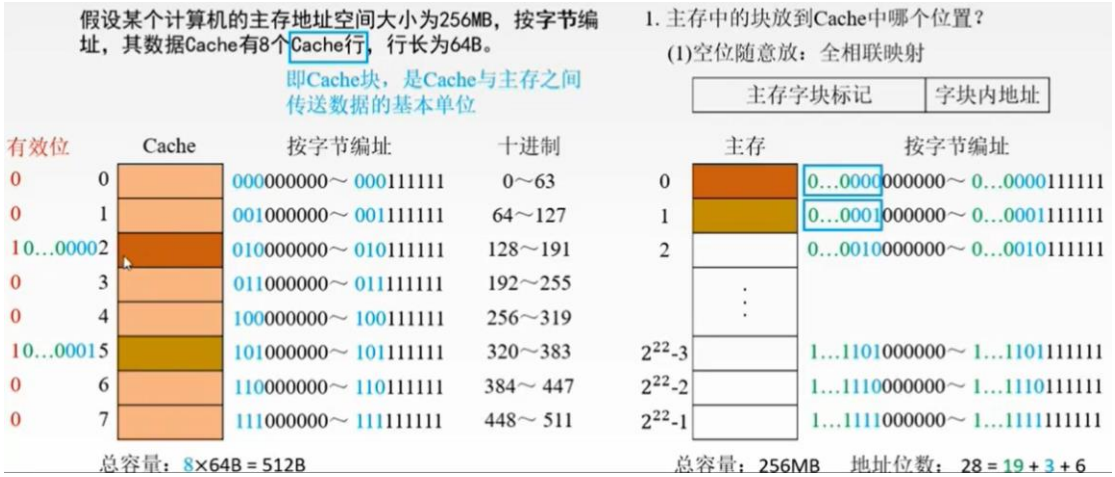


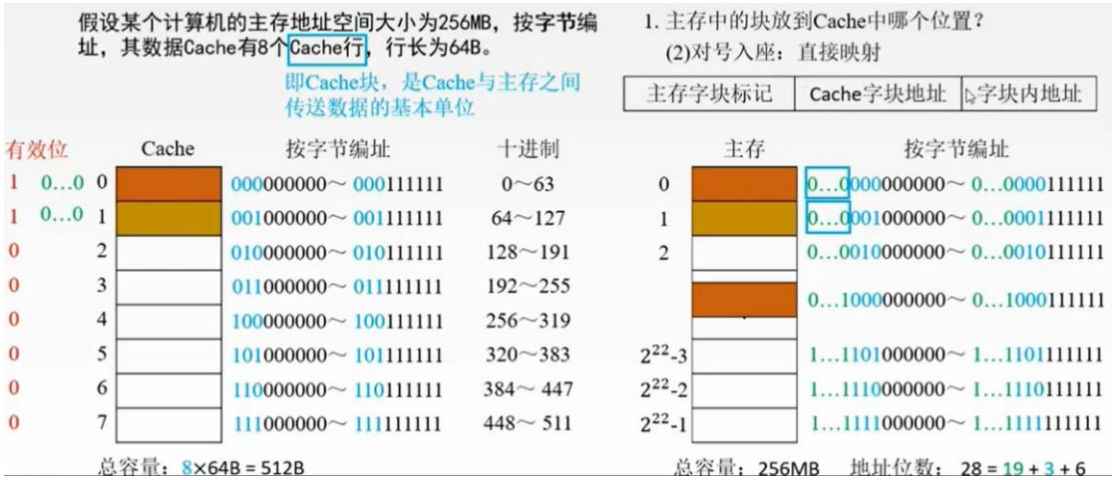
分散刷新　集中刷新　异步刷新

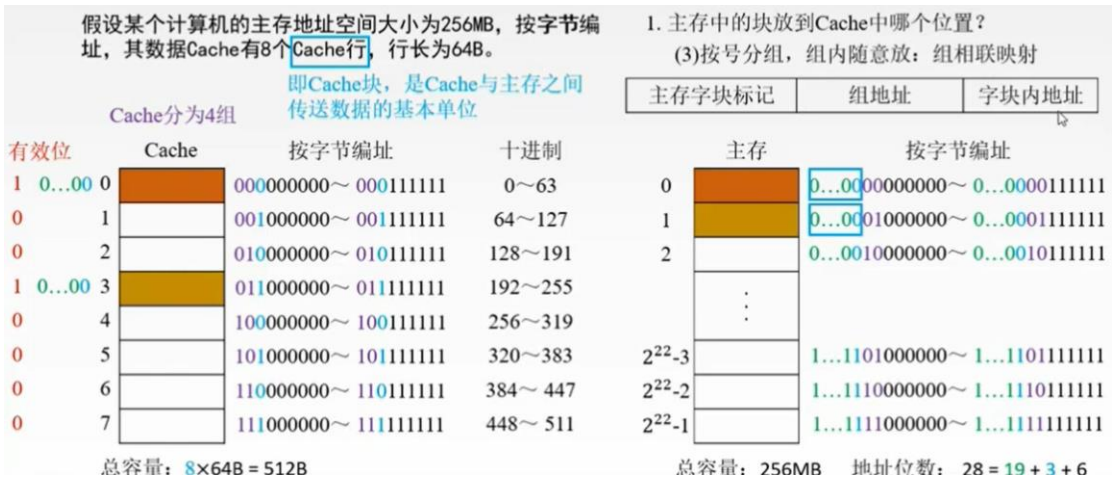







假设某个计算机的主存地址空间大小为256MB，按字节编址，其数据Cache有8个Cache行，行长为64B。
即Cache块，是Cache与主存之间传送数据的基本单位
1. 主存中的块放到Cache中哪个位置？
(1)空位随意放：全相联映射
主存字块标记
字块内地址
有效位
Cache
按字节编址
十进制
主存
按字节编址
0 0 000000000～000111111 0～63 0 0…0000000000～0…0000111111
0 1 001000000～001111111 64～127 1 0…0001000000～0…0001111111
1 0…0000 2 010000000～010111111 128～191 2 0…0010000000～0…0010111111
0 3 011000000～011111111 192～255
0 4 100000000～100111111 256～319
1 0…0001 5 101000000～101111111 320～383 2^22-3 1…1101000000～1…1101111111
0 6 110000000～110111111 384～447 2^22-2 1…1110000000～1…1110111111
0 7 111000000～111111111 448～511 2^22-1 1…1111000000～1…1111111111
总容量：8×64B = 512B
总容量：256MB 地址位数：28 = 19 + 3 + 6

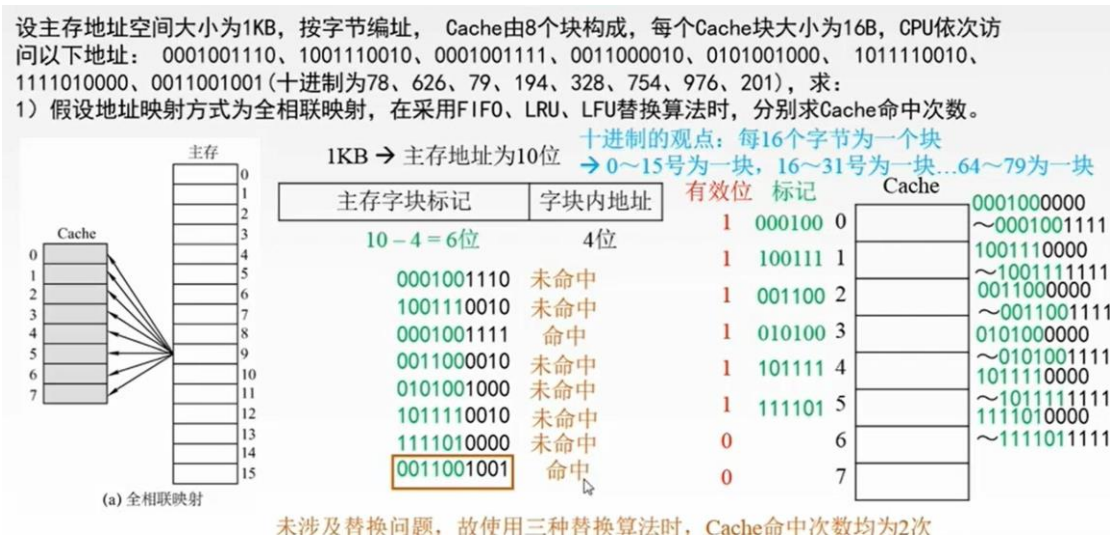

假设某个计算机的主存地址空间大小为256MB，按字节编址，其数据Cache有8个Cache行，行长为64B。
即Cache块，是Cache与主存之间传送数据的基本单位
1. 主存中的块放到Cache中哪个位置？
(2)对号入座：直接映射
主存字块标记
Cache字块地址
字块内地址
有效位
Cache
按字节编址
十进制
主存
按字节编址
1 0…0 0 000000000～000111111 0～63 0 0…0000000000～0…0000111111
1 0…0 1 001000000～001111111 64～127 1 0…0001000000～0…0001111111
0 2 010000000～010111111 128～191 2 0…0010000000～0…0010111111
0 3 011000000～011111111 192～255 0…1000000000～0…1000111111
0 4 100000000～100111111 256～319
0 5 101000000～101111111 320～383 2^22-3 1…1101000000～1…1101111111
0 6 110000000～110111111 384～447 2^22-2 1…1110000000～1…1110111111
0 7 111000000～111111111 448～511 2^22-1 1…1111000000～1…1111111111
总容量：8×64B = 512B
总容量：256MB 地址位数：28 = 19 + 3 + 6


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。
Cache分为4组
即Cache块，是Cache与主存之间传送数据的基本单位
1. 主存中的块放到Cache中哪个位置？
(3)按号分组，组内随意放：组相联映射
主存字块标记
组地址
字块内地址
有效位
Cache
按字节编址
十进制
主存
按字节编址
1 0…00 0 000000000～000111111 0～63 0 0…0000000000～0…0000111111
0 1 001000000～001111111 64～127 1 0…0001000000～0…0001111111
0 2 010000000～010111111 128～191 2 0…0010000000～0…0010111111
1 0…00 3 011000000～011111111 192～255
0 4 100000000～100111111 256～319
0 5 101000000～101111111 320～383 2^22-3 1…1101000000～1…1101111111
0 6 110000000～110111111 384～447 2^22-2 1…1110000000～1…1110111111
0 7 111000000～111111111 448～511 2^22-1 1…1111000000～1…1111111111
总容量：8×64B = 512B
总容量：256MB 地址位数：28 = 19 + 3 + 6

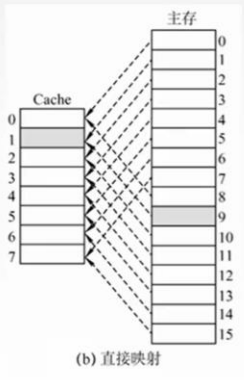
命中---写回法：Cache 被换出时才写回主存；全写法（写直通法）：同时写入 Cache 和主存

未命中---写分配法：主存中的块调入 Cache，在 Cache 中修改，搭配写回法

非写分配法：只写入主存，不调入 Cache，搭配全写法



设主存地址空间大小为1KB，按字节编址， Cache由8个块构成，每个Cache块大小为16B，CPU依次访问以下地址： 0001001110、1001110010、0001001111、0011000010、0101001000、 1011110010、1111010000、0011001001(十进制为78、626、79、194、328、754、976、201)，求：
2）假设地址映射方式为直接映射，求Cache命中次数。



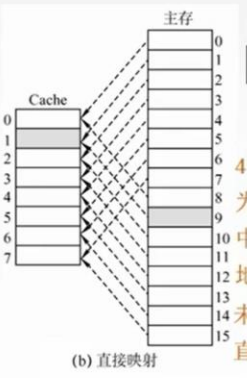
1KB → 主存地址为10位

十进制的观点：每16个字节为一个块
→ 0～15号为0号块，16～31号为1号块…64～79为4号块

主存字块标记	Cache字块地址	字块内地址
10－4－3＝3位	3位	4位

		有效位	标记	Cache
0001001110	未命中	0		0
1001110010	未命中	0		1
0001001111	命中	0		2
0011000010		0		3
0101001000		1	000	4
1011110010		0		5
1111010000		0		6
0011001001		1	100	7

设主存地址空间大小为1KB，按字节编址， Cache由8个块构成，每个Cache块大小为16B，CPU依次访问以下地址： 0001001110、1001110010、0001001111、0011000010、0101001000、 1011110010、1111010000、0011001001(十进制为78、626、79、194、328、754、976、201)，求：
2）假设地址映射方式为直接映射，求Cache命中次数。



1KB → 主存地址为10位

十进制的观点：每16个字节为一个块
→ 0～15号为0号块，16～31号为1号块…64～79为4号块

主存字块标记	Cache字块地址	字块内地址
10－4－3＝3位	3位	4位

			有效位	标记	Cache
0001001110	未命中		0		0
1001110010	未命中		0		1
0001001111	命中		0		2
0011000010	未命中		0		3
0101001000	未命中		1	001	4
1011110010	未命中		1	111	5
1111010000	未命中		0		6
0011001001	未命中		1	101	7

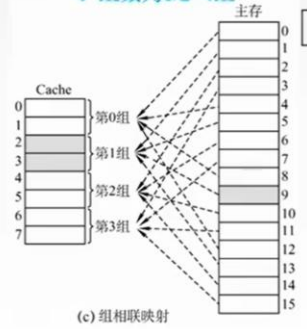
4号块有效位为1，但Cache中标记与当前地址标记不同，未命中，直接替换

命中：有效位为1，且标记相同

对号入座→有冲突直接替换，不涉及替换策略，命中次数为1次

设主存地址空间大小为1KB，按字节编址， Cache由8个块构成，每个Cache块大小为16B，CPU依次访问以下地址： 0001001110、1001110010、0001001111、0011000010、0101001000、 1011110010、1111010000、0011001001(十进制为78、626、79、194、328、754、976、201)，求：
3）假设地址映射方式为二路组相联映射，在采用FIFO、LRU、LFU替换算法时，分别求Cache命中次数。

即2块为一组
→ 组数为8/2=4组



1KB → 主存地址为10位

十进制的观点：每16个字节为一个块
→ 0～15号对应第0组…64～79对应第4 mod 4 = 0组

主存字块标记	组地址	字块内地址
10－4－2＝4位	2位	4位

			有效位	标记	Cache	
0001001110	未命中	第0组已满，但当	1	0001	0	0
1001110010	未命中	前标记与Cache两	1	0011	1	
0001001111	命中	个标记都不同	0		2	1
0011000010	未命中	FIFO：最早调入	0		3	
0101001000		的块标记是0001，	0		4	2
1011110010		故替换之	0		5	
1111010000		LRU：刚用过0011，	1	1001	6	3
0011001001		故替换0001	0		7	

LFU：0011和0001均用过1次，需要更多依据，详见操作系统

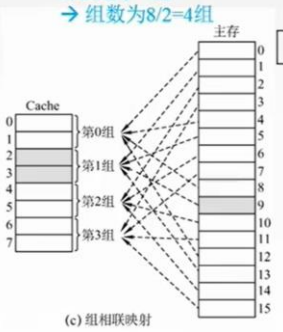
设主存地址空间大小为1KB，按字节编址， Cache由8个块构成，每个Cache块大小为16B，CPU依次访问以下地址： 0001001110、1001110010、0001001111、0011000010、0101001000、 1011110010、1111010000、0011001001(十进制为78、626、79、194、328、754、976、201)，求：

3）假设地址映射方式为二路组相联映射，在采用FIFO、LRU、LFU替换算法时，分别求Cache命中次数。

即2块为一组 → 组数为8/2=4组

1KB → 主存地址为10位

十进制的观点：每16个字节为一个块 → 0～15号对应第0组…64～79对应第4 mod 4 = 0组



主存字块标记	组地址	字块内地址
10－4－2＝4位	2位	4位

		有效位	标记		Cache
0001001110	未命中	1	0101	0	0
1001110010	未命中	1	0011	1	
0001001111	命中	1	1111	2	1
0011000010	未命中	0		3	
0101001000	未命中	0		4	
1011110010	未命中	0		5	2
1111010000	未命中	1	1001	6	
0011001001	命中	1	1011	7	3

对号入组→不同的替换策略，本例中替换结果相同，命中次数均为2次

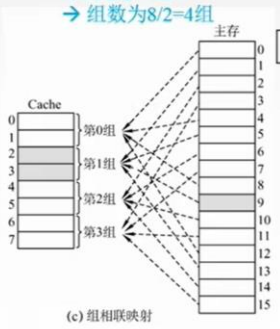
设主存地址空间大小为1KB，按字节编址， Cache由8个块构成，每个Cache块大小为16B，CPU依次访问以下地址： 0001001110、1001110010、0001001111、0011000010、0101001000、 1011110010、1111010000、0011001001(十进制为78、626、79、194、328、754、976、201)，求：

3）假设地址映射方式为二路组相联映射，在采用FIFO、LRU、LFU替换算法时，分别求Cache命中次数。

即2块为一组 → 组数为8/2=4组

1KB → 主存地址为10位

十进制的观点：每16个字节为一个块 → 0～15号对应第0组…64～79对应第4 mod 4 = 0组



主存字块标记	组地址	字块内地址
10－4－2＝4位	2位	4位

			有效位	标记		Cache
0001001110	未命中	FIFO：最早调入的块标记是0011，故替换之	1	0101	0	0
1001110010	未命中		1	0011	1	
0001001111	命中		1	1111	2	1
0011000010	未命中	LRU：刚用过0011，故替换0101	0		3	
0101001000	未命中		0		4	
1011110010	未命中	LFU：0011用过2次，0101用过1次，故替换0101	0		5	2
1111010000	未命中		1	1001	6	
0011001001	命中		1	1011	7	3
1011001001	未命中					

有效位	标记位	一致性维护位	替换算法位
1位	地址映射 4位	写策略	替换策略

地址映射：

主存字块标记	组地址	字块内地址
10-4-2=4	2	4

写策略：写回法1位，直写法0位

替换策略：详见操作系统，此处暂不考虑

写回法：(1+4+1)×8 = 48bit，共4096+48 = 4144bit

直写法：(1+4)×8 = 40bit，共4096+40 = 4136bit

标记项	数据
标记项	数据
标记项	数据
标记项	数据
标记项	数据
标记项	数据
标记项	数据
标记项	数据

8 × 16B = 512B = 4096bit