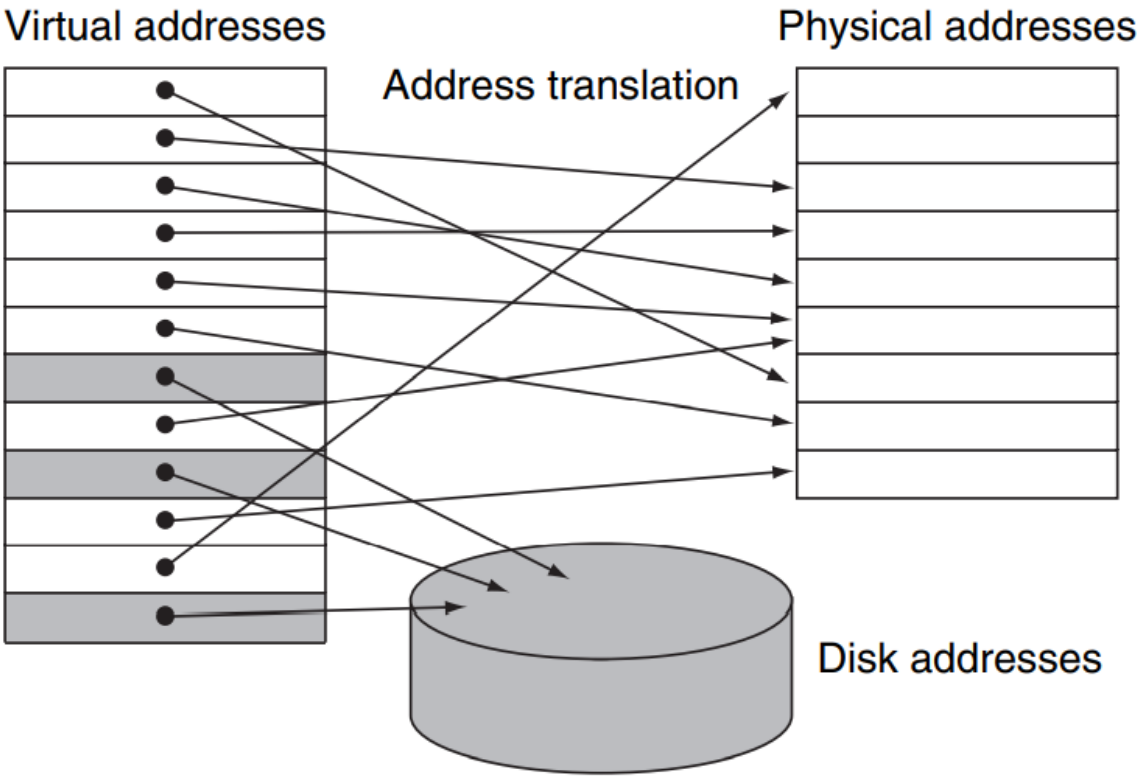
计算题

算 CPI，但是会给出指令正常 CPI，取址指令 L1-cache miss rate，L2-cache miss rate，分支预测错误率，因为指令内存和数据内存会造成结构冒险，算最后 CPI

虚拟地址

虚拟地址

实际上的 main memory（我们称之为 **物理内存, physical memory**）中的地址称为 **物理地址, physical addresses**；而我们给每一个程序内部使用到的内存另外编一套地址，称为 **虚拟地址, virtual addresses**；虚拟内存技术负责了这两个地址之间的转换 (**address translation**，我们稍后再讨论转换的方式)：



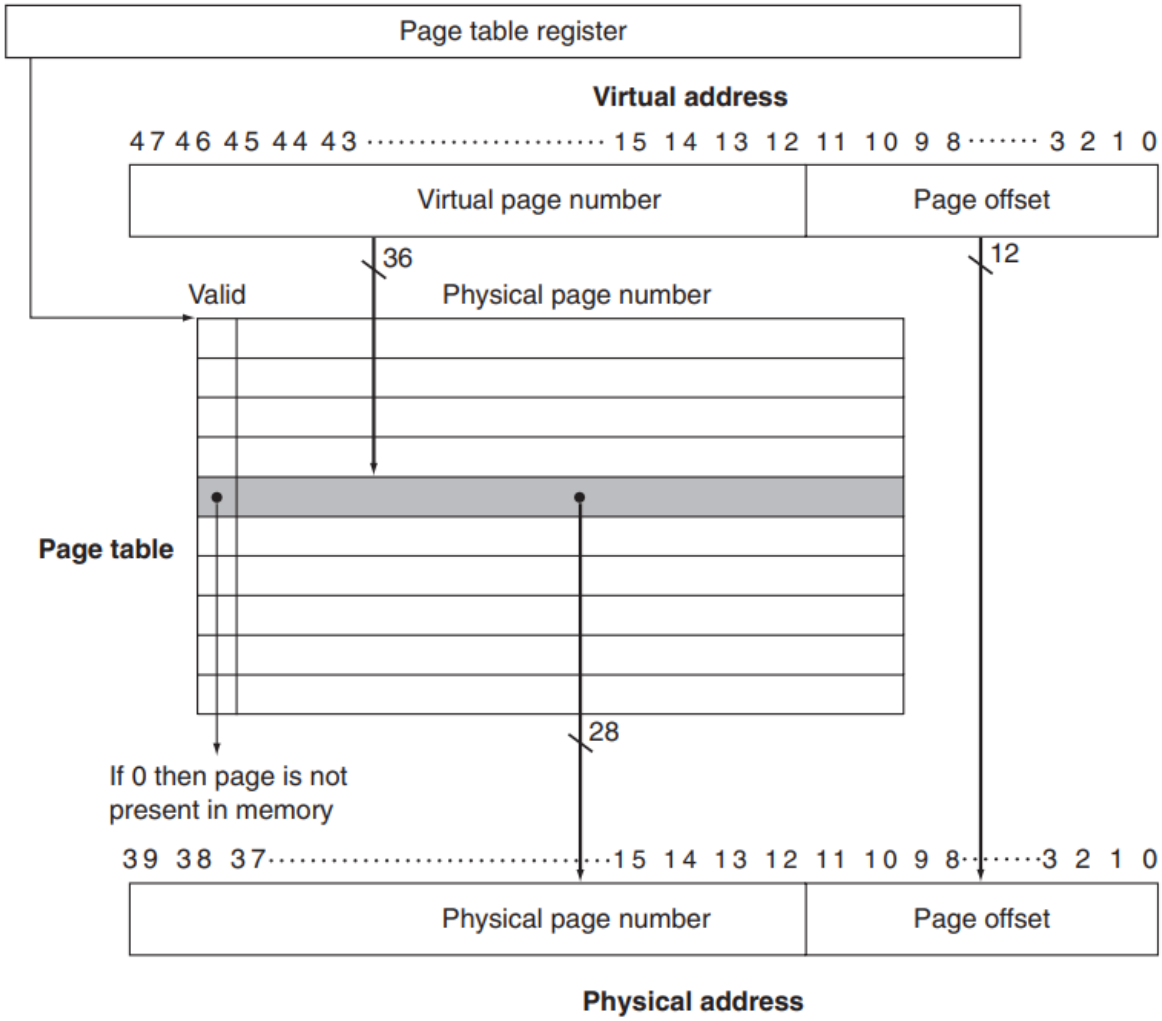
虚拟存储的技术和 cache 的原理是一样的，但是一些术语的名字并不相同。对应于 cache 中的 block / line，虚拟存储的内存单元称为 **page**，当我们要访问的 page 不在主存中而是在磁盘里，也就是 miss，我们称之为一次 **page fault**。

physical memory 的存放并没有分组的概念，即用 cache 的术语来说，main memory 是 fully-associative 的。page fault 的开销是非常大的，因此比较低的 page fault 的概率相对于额外的查询来说是非常划算的。同样，由于读写磁盘是非常慢的，write through 的策略并不合适，因此在 virtual memory 的技术中，我们采取 write back 的方式。

页表

page table 这种结构，它被存放在 main memory 中，每个程序（实际上是进程，但是写课本的人好像现在并不想引入这个概念）都有一个自己的 page table；同时硬件上有一个 **page table register** 保存当前进程这个页表的地址。

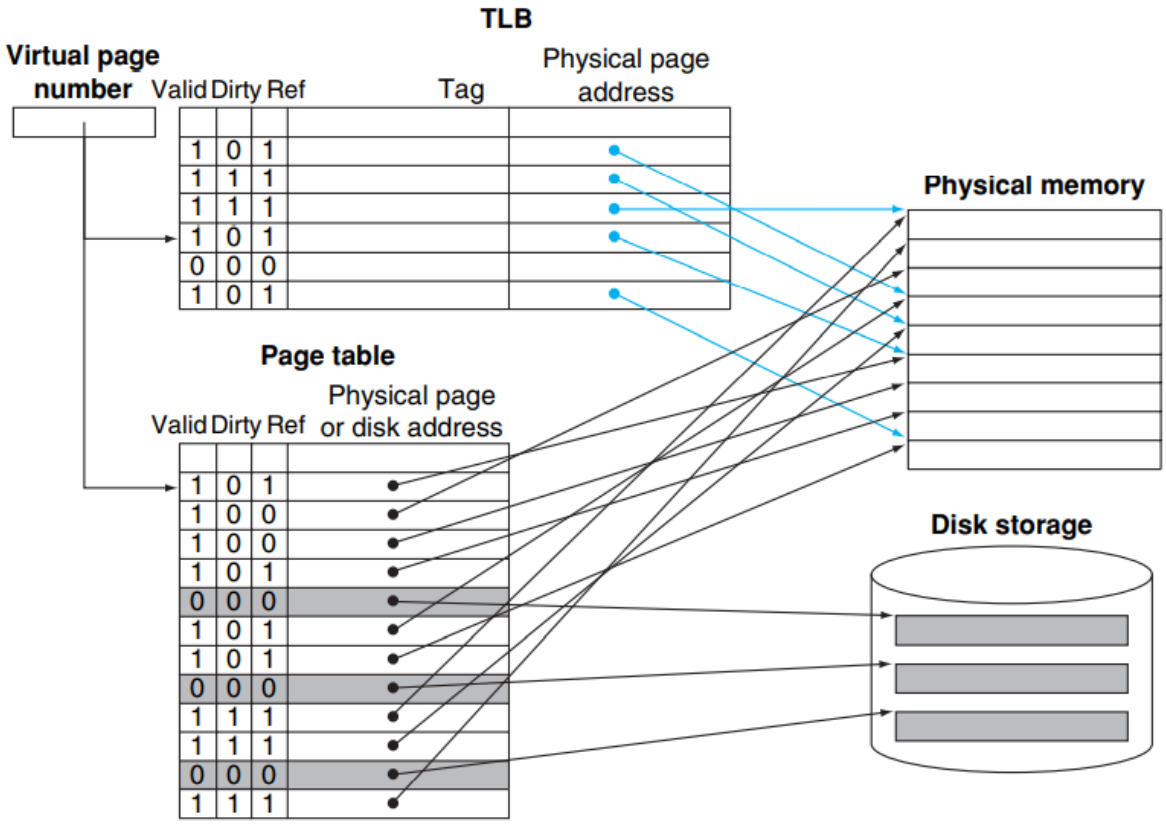
每个 entry 中包含了一个 valid bit 和 physical page number。如果 valid bit = 1，那么转换完成；否则触发了 page fault，handle 之后再进行转换。



TLB

结构

一个专用的高速查找硬件 cache，这里称它为 **translation look-aside buffer (TLB)**。它实际上就是 page table 的专用 cache（它真的是 cache；page table 并不是 cache，只是像 cache），其 associativity 的设计可以根据实际情况决定。



当 TLB miss 的时候，处理器去 page table 查找对应的项；如果发现对应项是 valid 的，那么就把他拿到 TLB 里（此时被替换掉的 TLB entry 的 dirty bit 如果是 1，也要写会 page table）；否则就会触发一个 page fault，然后在做上述的事。

TLB	Page table	Cache	Possible? If so, under what circumstance?
Hit	Hit	Miss	Possible, although the page table is never really checked if TLB hits.
Miss	Hit	Hit	TLB misses, but entry found in page table; after retry, data is found in cache.
Miss	Hit	Miss	TLB misses, but entry found in page table; after retry, data misses in cache.
Miss	Miss	Miss	TLB misses and is followed by a page fault; after retry, data must miss in cache.
Hit	Miss	Miss	Impossible: cannot have a translation in TLB if page is not present in memory.
Hit	Miss	Hit	Impossible: cannot have a translation in TLB if page is not present in memory.
Miss	Miss	Hit	Impossible: data cannot be allowed in cache if the page is not in memory.

FIGURE 5.33 The possible combinations of events in the TLB, virtual memory system, and cache. Three of these combinations are impossible, and one is possible (TLB hit, page table hit, cache miss) but never detected.

大小计算

TLB的大小以及TLB对应内存的大小

$$表项数量 = \frac{2^{虚拟地址位数}}{页表大小}$$

$$page\ table_{大小} = 表项数量 \times 表项大小$$

多核情况

即上课讲的 FSM

外设

纠错码

海明码估计不考

稳定性的衡量

- MTTF mean time to failure 平均无故障时间
- MTTR mean time to repair 平均修复时间
- MTBF (Mean Time Between Failures) 平均故障间隔时间
  = MTTF+ MTTR

$$Availability = \frac{MTTF}{MTTF + MTTR}$$

RAID

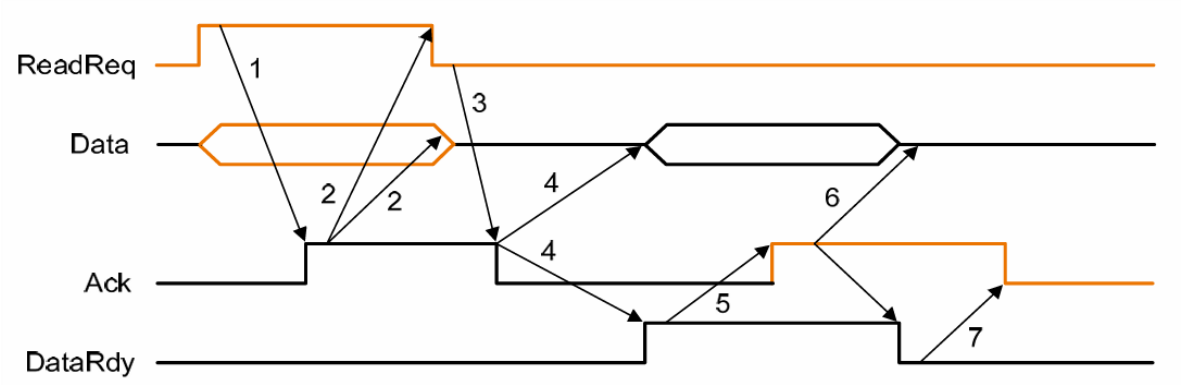
> 我们需要在硬盘里储存必要的信息并保持一定的稳定

RAID level	可容纳最多故障盘	例子盘数	check disks
0（无冗余）	0	8	0
1（完全复制）	1	8	8
2	1	8	4
3（奇偶校验）	1	8	1
4（按扇区校验）	1	8	1
5（检验块分盘）	1	8	1
6（有两位）	2	8	2

总线

分为 Synchronous bus （同步总线use a clock）和asynchronous bus（异步总线使用如下握手协议）

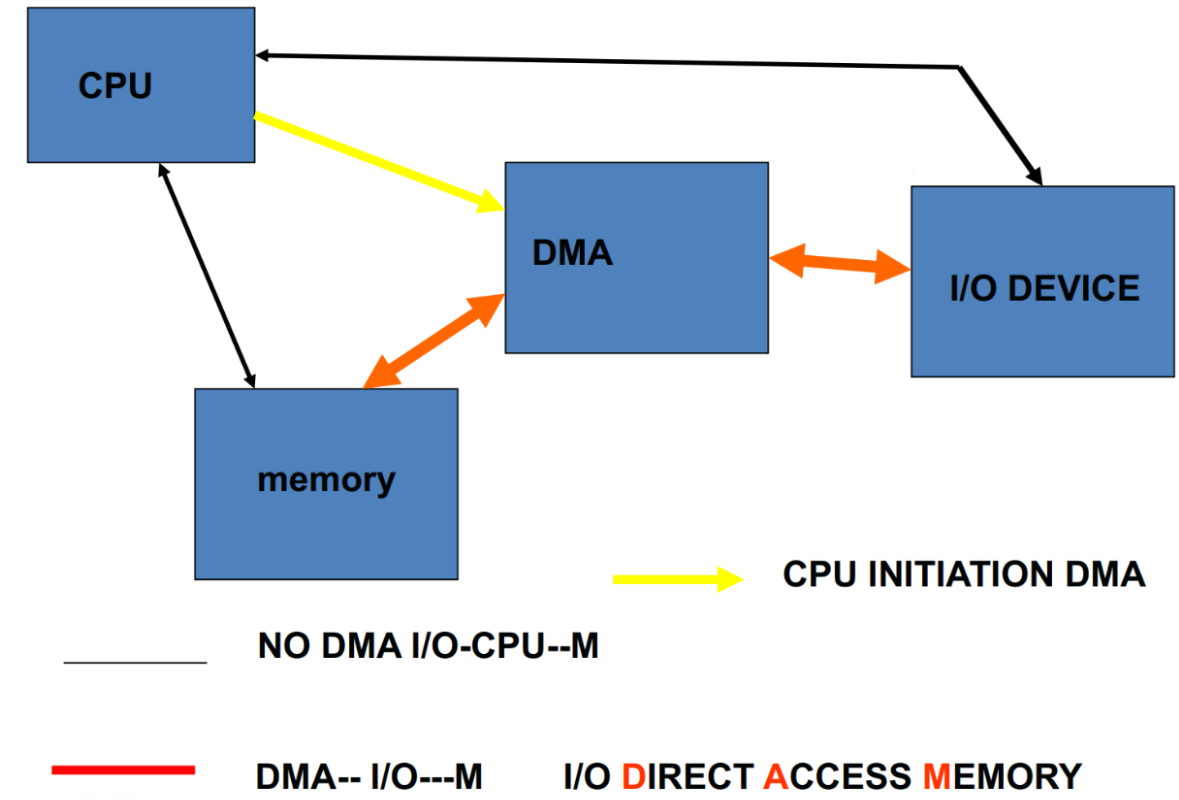
asynchronous bus



读数据时 , CPU 把 read request 拉起来。内存看到后，会把 Data 总线上的读走（即地址），随后进行内存读取，同时把 Ack 信号拉起来，告诉 IO 设备我们已经接收到 read request 了 . IO 设备看到 Ack 后把自己的 read request 放下，内存看到 read request 放下后，把 Ack 也放下。
内存读出数据后，会把 data ready 拉起来，把数据放在 data line 上. IO 设备看到 data ready 后会把数据取走，并把 Ack 信号拉起。内存看到 Ack 信号后会放下 data ready 信号，随后 IO 设备放下 ack 信号。

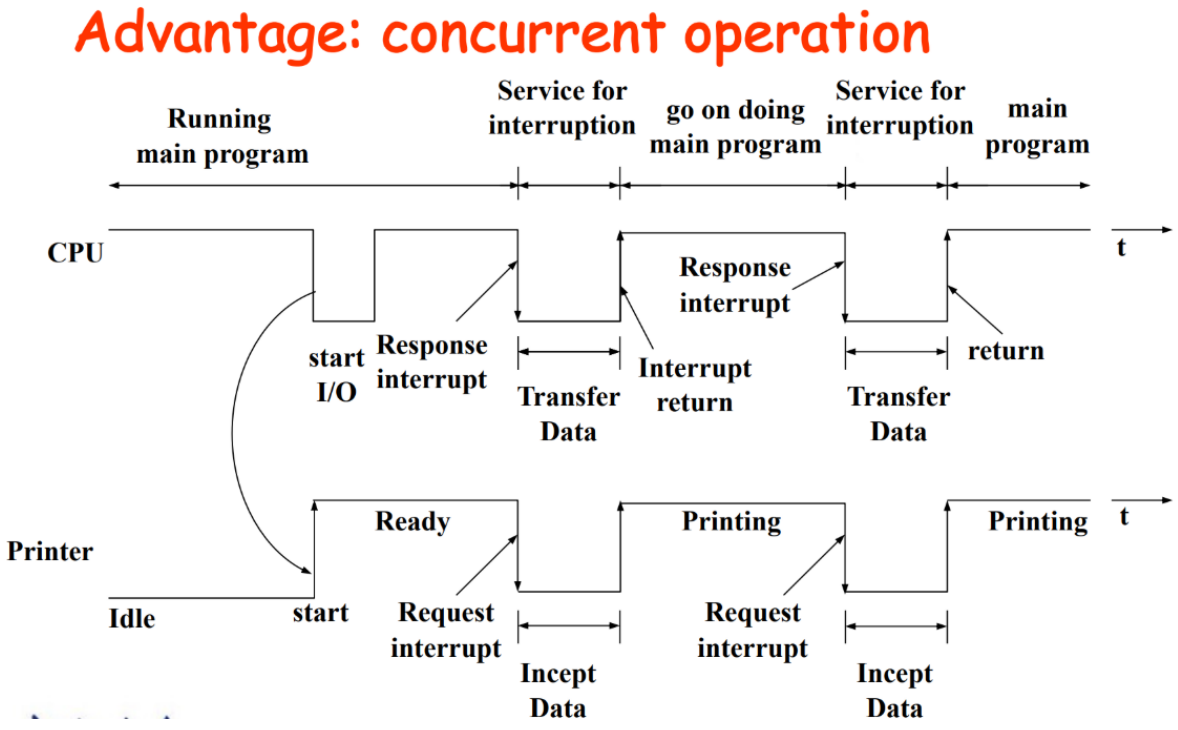
CPU与外设的交互

- Polling: The processor periodically checks status bit to see if it is time for the next I/O operation.
  定期检查设备，但是会占用 CPU.
- Interrupt: When an I/O device wants to notify processor that it has completed some operation or needs attentions, it causes processor to be interrupted.
  当 IO 设备完成操作给 CPU 一个中断，等待其响应。好处是 CPU 可以一直做自己的事情。
- DMA (direct memory access): the device controller transfer data directly to or from memory without involving processor.
  IO 设备直接和内存交互，不需要 CPU 参与。



CPU 需要配置 DMA. DMA 会和 IO 设备交互，把数据搬到内存，不需要 CPU 参与。

- CPU 配置 DMA，包括哪个设备、做什么操作、内存地址、数据大小等。
- DMA 开始操作设备，占用总线。如果需要多次传输，DMA 会生成下一个内存地址，开始下一次传输。
  DMA 也是挂在总线上的 master, 优先级没有 CPU 高。因此他会趁 CPU 空闲的时候搬运数据，可以充分利用总线。
- DMA 完成后，给 CPU 发中断，CPU 检查是否有错误。



假设 IO 是个打印机。每次打印一个字符，就会给 CPU 发一个中断。CPU 会去读取打印机的状态，看是否完成。完成后 CPU 继续做自己的事情。