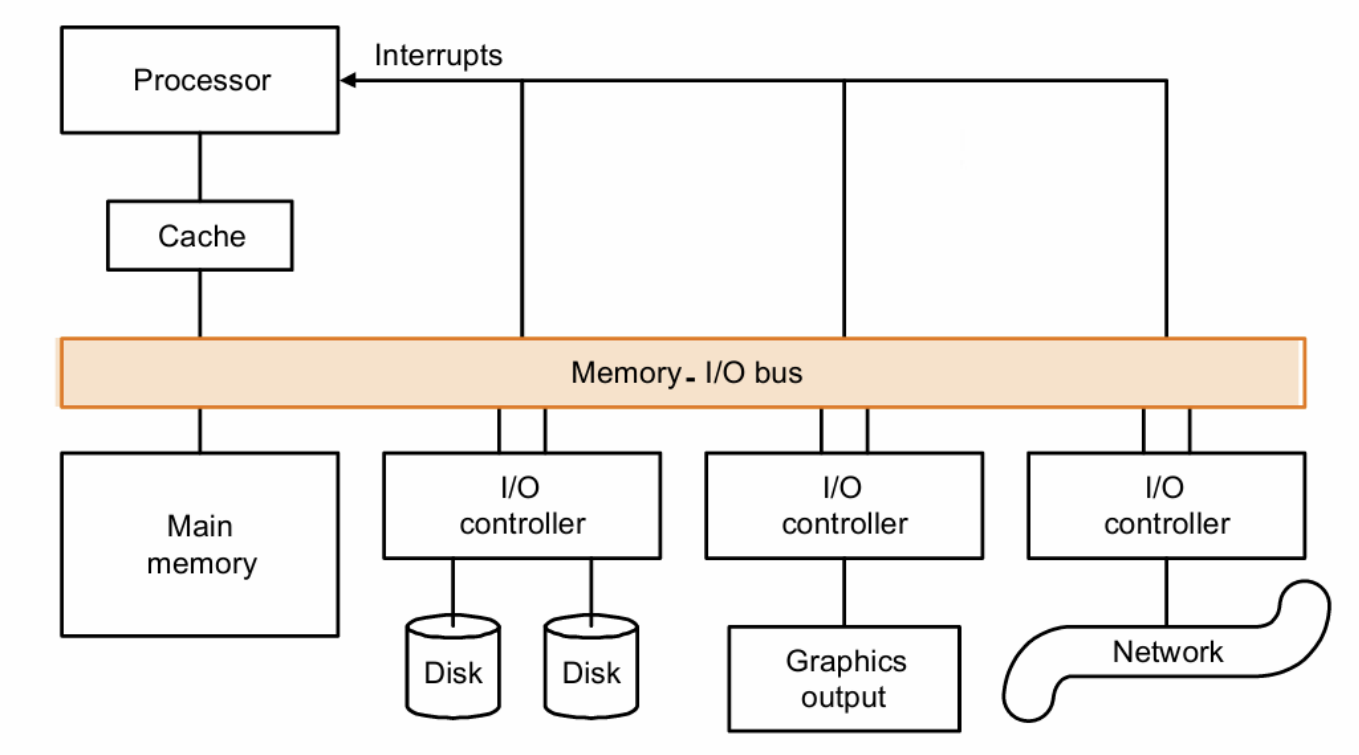
I/O设备

这一章主要学习有关存储,网络和其他输入输出设备的一些知识

I/O设备的架构图



三个IO的特征:

- 行为:输入,输出还是存储
- 交互对象:人或者是机器
- 数据传输率:数据在IO设备和存储或者CPU之间的最高传输效率

IO设备的表现:

- `Throughput`:可以理解为带宽,但是这个带宽可以有两种不同的解释方式,第一种是在一个固定的时间内能传输多少数据,也就是我们下意识认为的带宽;另一种是在单位时间内可以进行几次IO操作
- `Response time`:响应时间

I/O设备的多样性

Device	Behavior	Partner	Data rate (KB/sec)
Keyboard	input	human	0.01
Mouse	input	human	0.02
Voice input	input	human	0.02
Scanner	input	human	400.00
Voice output	output	human	0.60
Line printer	output	human	1.00
Laser printer	output	human	200.00
Graphics display	output	human	60,000.00
Modem	input or output	machine	2.00-8.00
Network/LAN	input or output	machine	500.00-6000.00
Floppy disk	storage	machine	100.00
Optical disk	storage	machine	1000.00
Magnetic tape	storage	machine	2000.00
Magnetic disk	storage	machine	2000.00-10,000.00

Amdahl's law

需要注意的是,即使CPU的速度能够大幅提升,但I/O设备的速度不能跟上的话,也会导致速度提高的倍率下降,也就是说一昧的只提升CPU的速度最终会达到瓶颈,这个瓶颈就是I/O设备的速度

Storage

磁盘主要有两种:

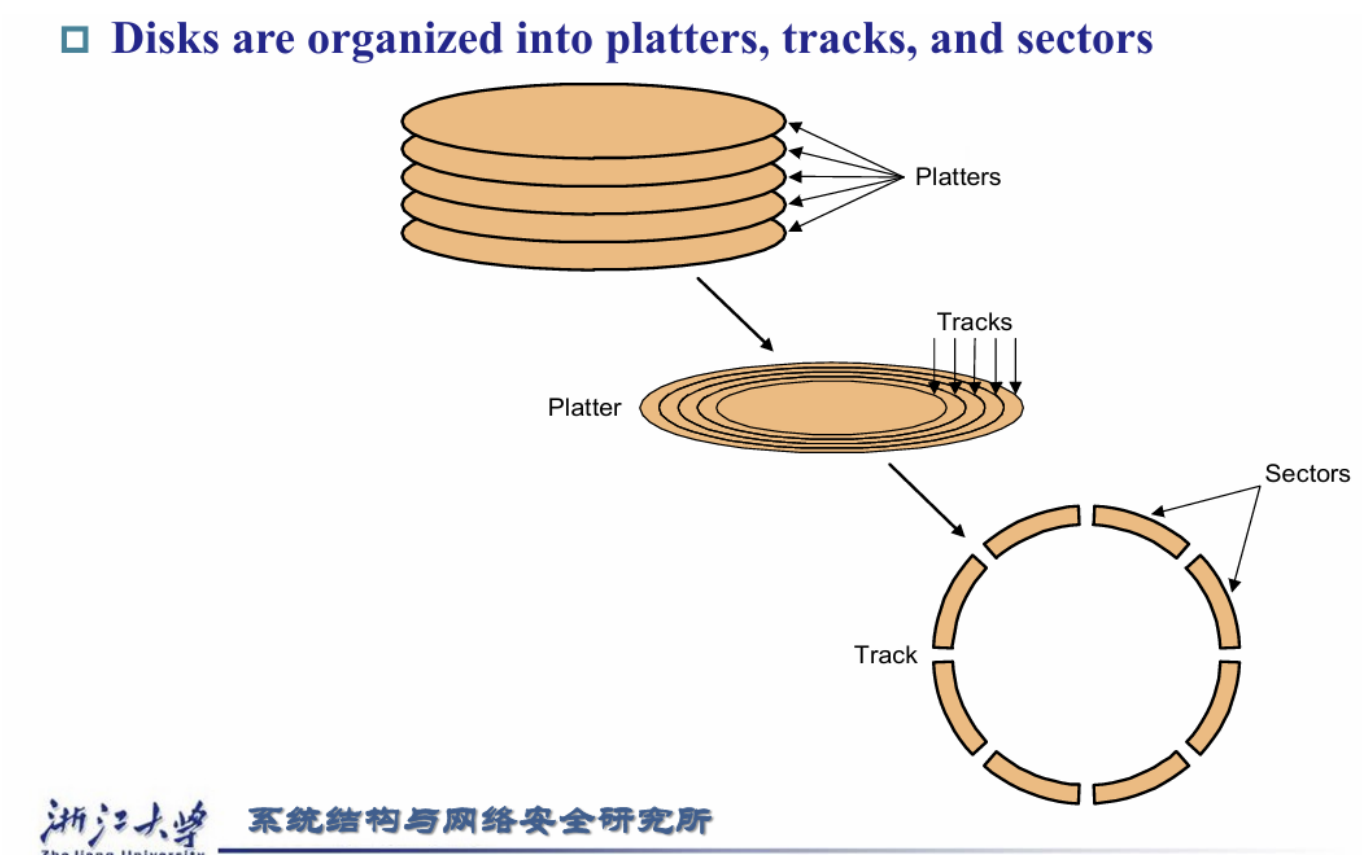
- `floppy disk`:软盘
- `hard disk`:硬盘

机械硬盘

被称作HDD(hard-disk drive)

机械硬盘的主要结构有以下几个部分:

- `platter`:盘片,一个硬盘由许多个盘片构成,每个盘片的上下两面都是可以记录的,同时每个盘片配备了一个读写磁头
- `track`:轨道,每个盘片的表面划了许多道同心的轨道
- `sector`:扇区,每条轨道又被划分成多个扇区,是最小的读写单元,一个扇区通常大小为512Byte



访问硬盘中的数据需要以下几步:

1. `seek`:使读写头找到对应的轨道
2. `rotational latency`:盘片进行旋转,使读写头到对应的扇区
3. `transfer time`:数据传输时间
4. `disk controller`:各种控制信号的处理时间

Disk Read Time 512B/sector 50MB/S

Access Time = Seek time + Rotational Latency + Transfer time + Controller Time

$$= 6\text{ms} + \frac{0.5}{10{,}000\text{PRM}} + \frac{0.5\text{KB}}{50\text{MB/sec}} + 0.2\text{ms}$$

$$= 6\text{ms} + 3.0 + 0.01 + 0.2 = 9.2\text{ms}$$

Assuming the measured seek time is 25% of the calculated average

$$\text{Access Time} = 25\% \times 6\text{ms} + 3.0\text{ ms} + 0.01\text{ms} + 0.2\text{ms} = 4.7\text{ms}$$

其中的RPM表示每分钟旋转的次数,由于平均旋转为半圈,所以分子为0.5

提高disk的表现性能有几种方法:

- 利用局部性和操作系统的调度让实际的平均seek时间降低
- 利用更好的disk controller来分配物理扇区,将逻辑扇区的接口暴露给主机即可,常见的协议有SCSI,ATA,SATA
- 给disk准备缓存(cache),预取出扇区中的数据,从而避免seek和rotational latency

Flash

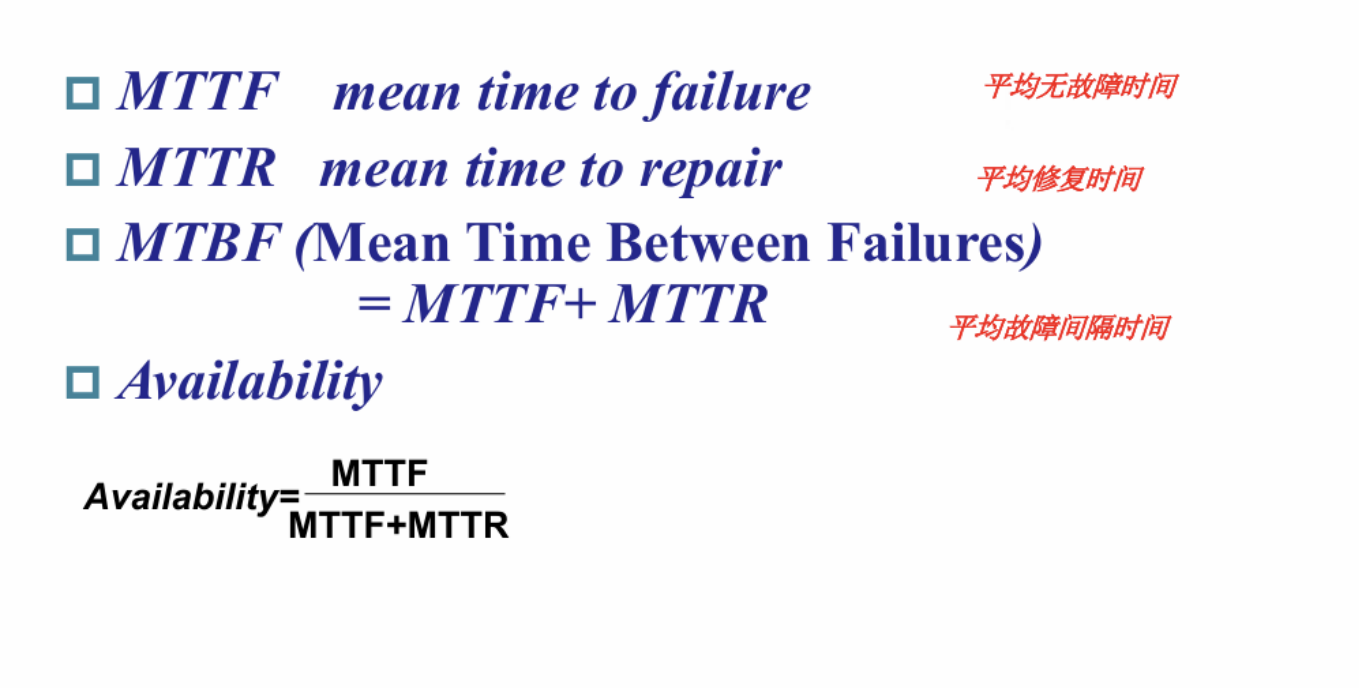
Flash是一种非易失性存储 比机械硬盘存储快上100~1000倍,更小,功耗更低,也更耐糙,但价格更高

Flash有以下几种:

- NOR:更多用来作为嵌入式系统的存储
- NAND:更多用来作为U盘或者SSD(solid-state drive/固态硬盘)

flash被多次访问后会坏掉,因此不适合直接作为RAM和disk的替代品,万一某个单元坏掉了,那么可以提前把数据重新映射到用的较少的block中

可用性(Availability)



三种方法来提升MTTF:

- Fault avoidance:避免错误的发生
- Fault tolerance:通过冗余的设计来承担一定程度的错误发生
- Fault forecasting:提前预测错误的发生

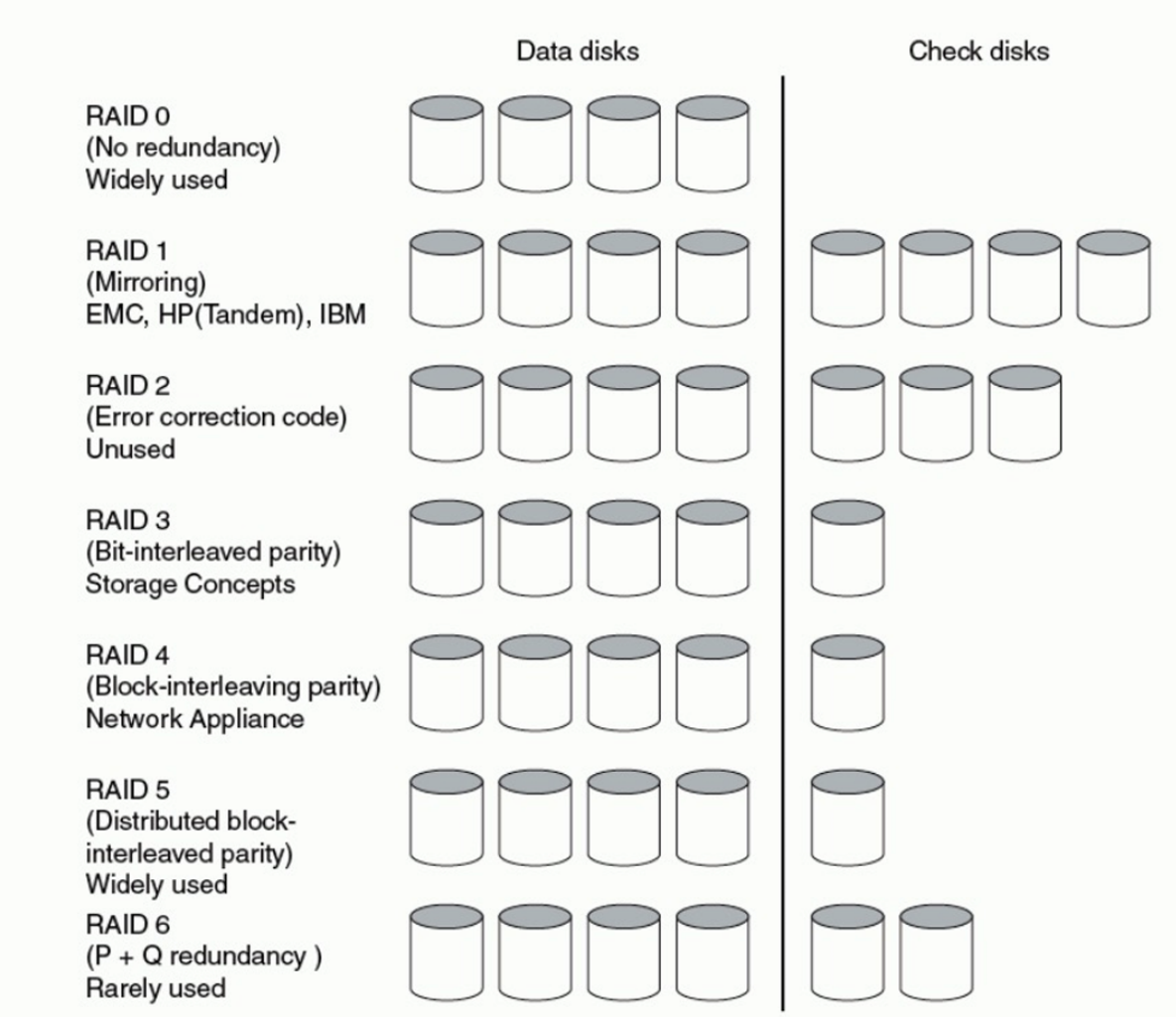
RAID

所谓的RAID就是一种提升错误承担能力的方法,通过冗余设计让其能够自主修复

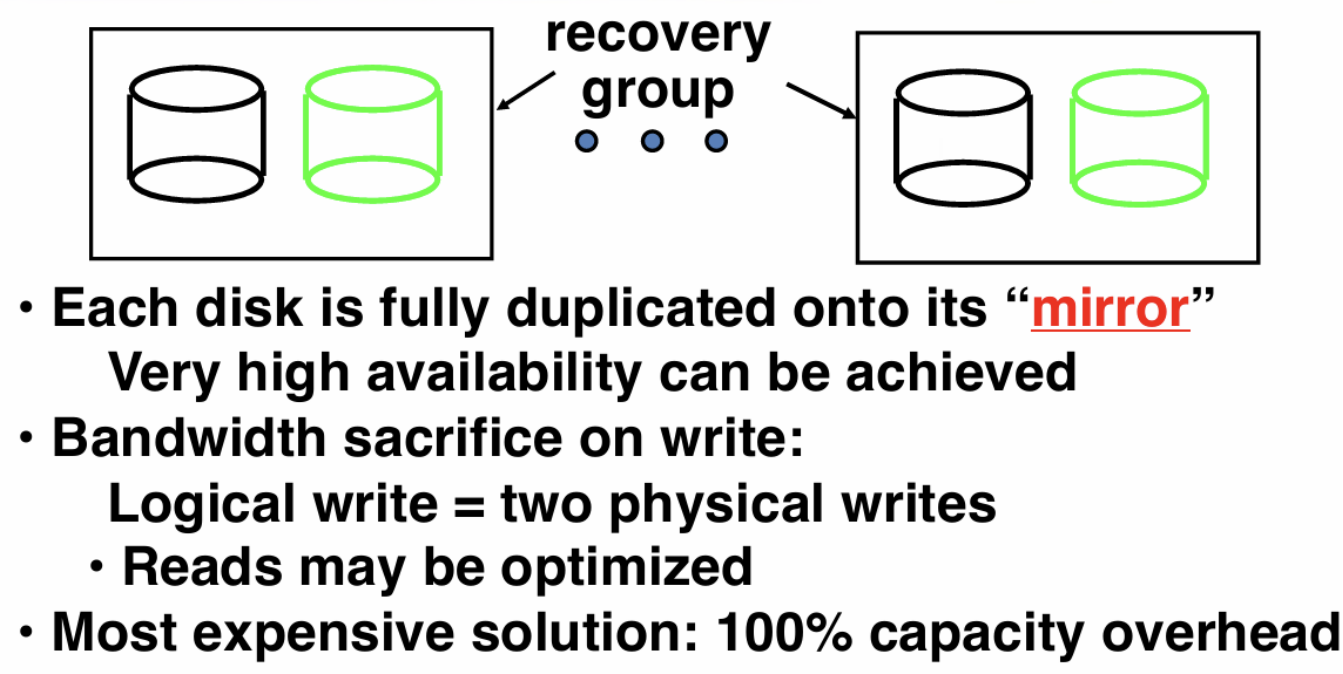
RAID的全称是Redundant Arrays of Inexpensive Disks

在这种策略下,文件被条带化,也就是分成多份后存储在不同的磁盘当中

下面是不同级别的RAID的示意图:

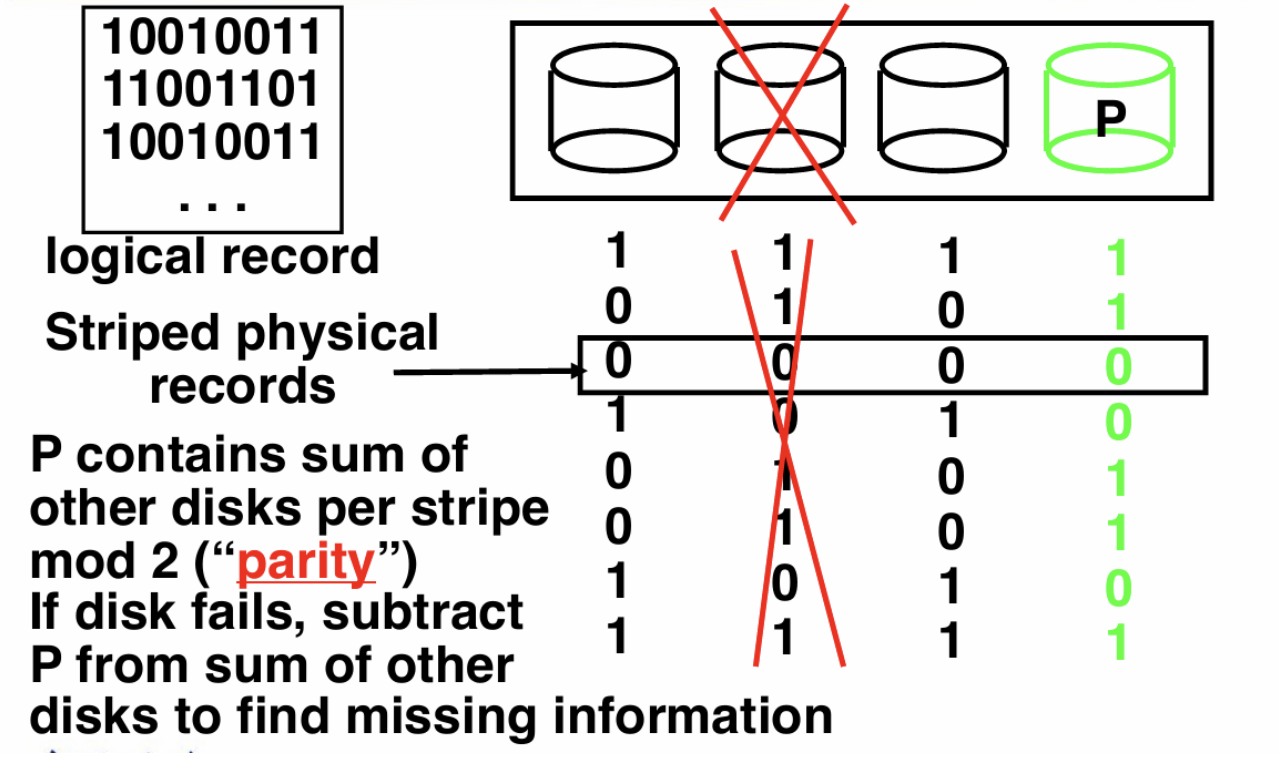


- `RAID 0`:也就是没有冗余
- `RAID 1`:Disk Mirroring/Shadowing,每个disk都有一个完全的复制在另一个磁盘中

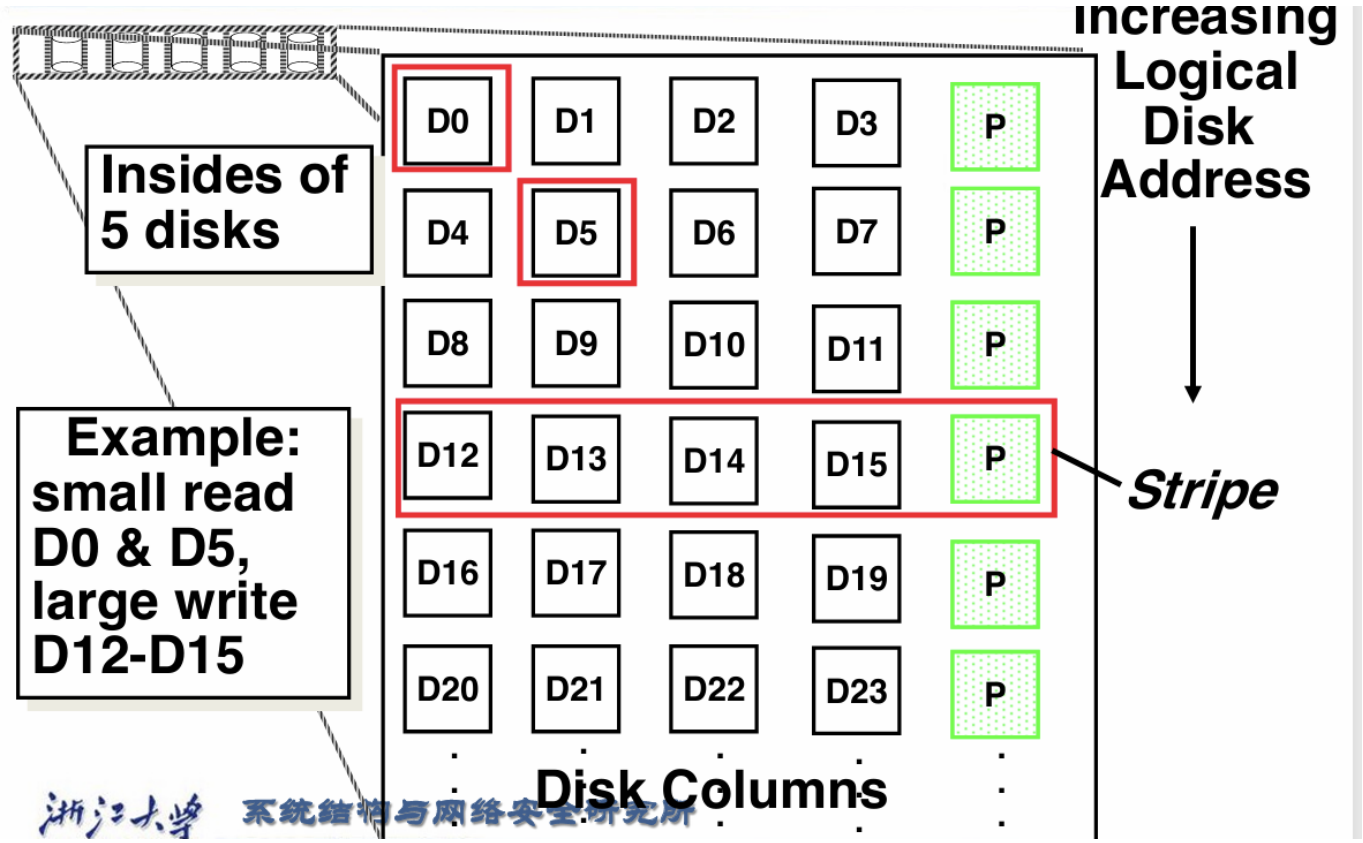


- `RAID 2`:没有使用,所以这里不研究

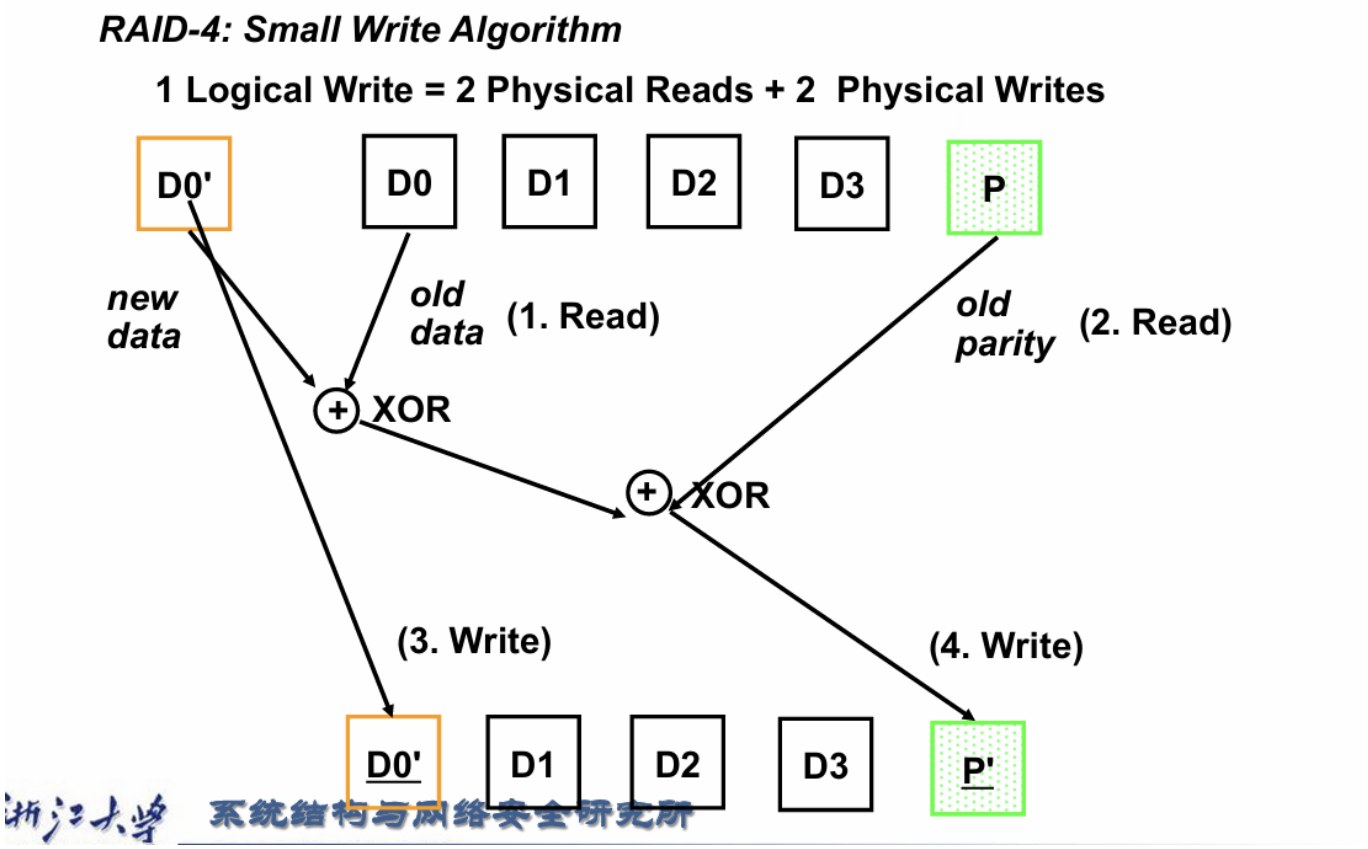
- RAID 3:Bit-Interleaved Parity disk,通过一个额外的磁盘来存放校验码,通过校验码就可以恢复对应磁盘中的数据



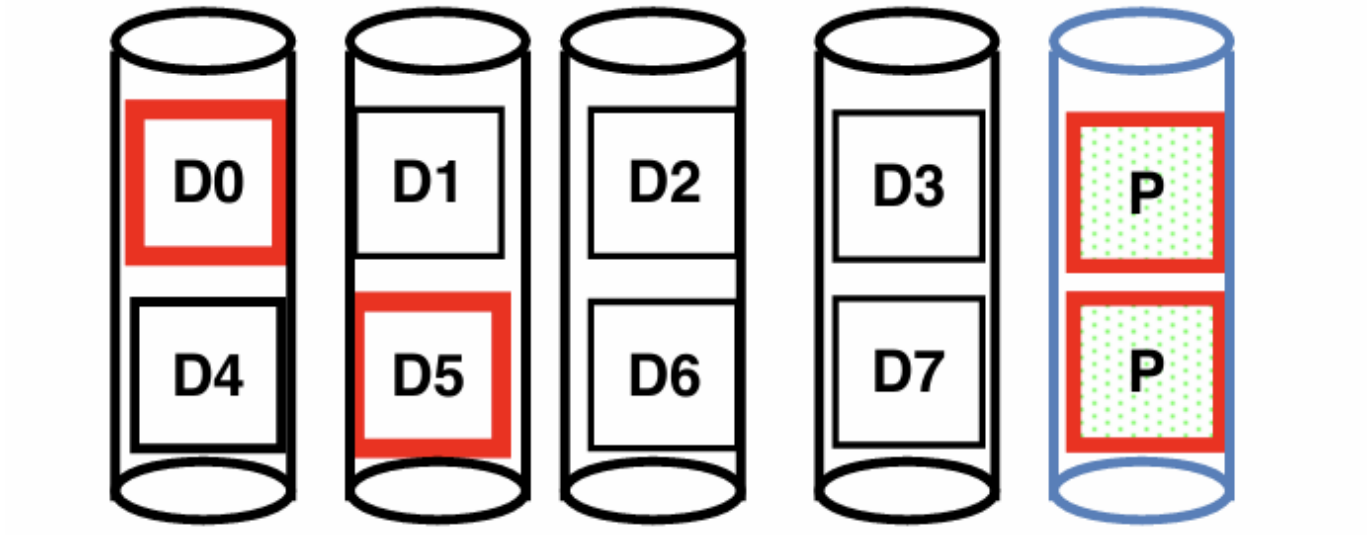
- RAID 4:延续RAID3的思想,只不过是以块为单位进行分配



small write和large write分别表示写一个磁盘块和写一整行的磁盘块(在不同磁盘中)

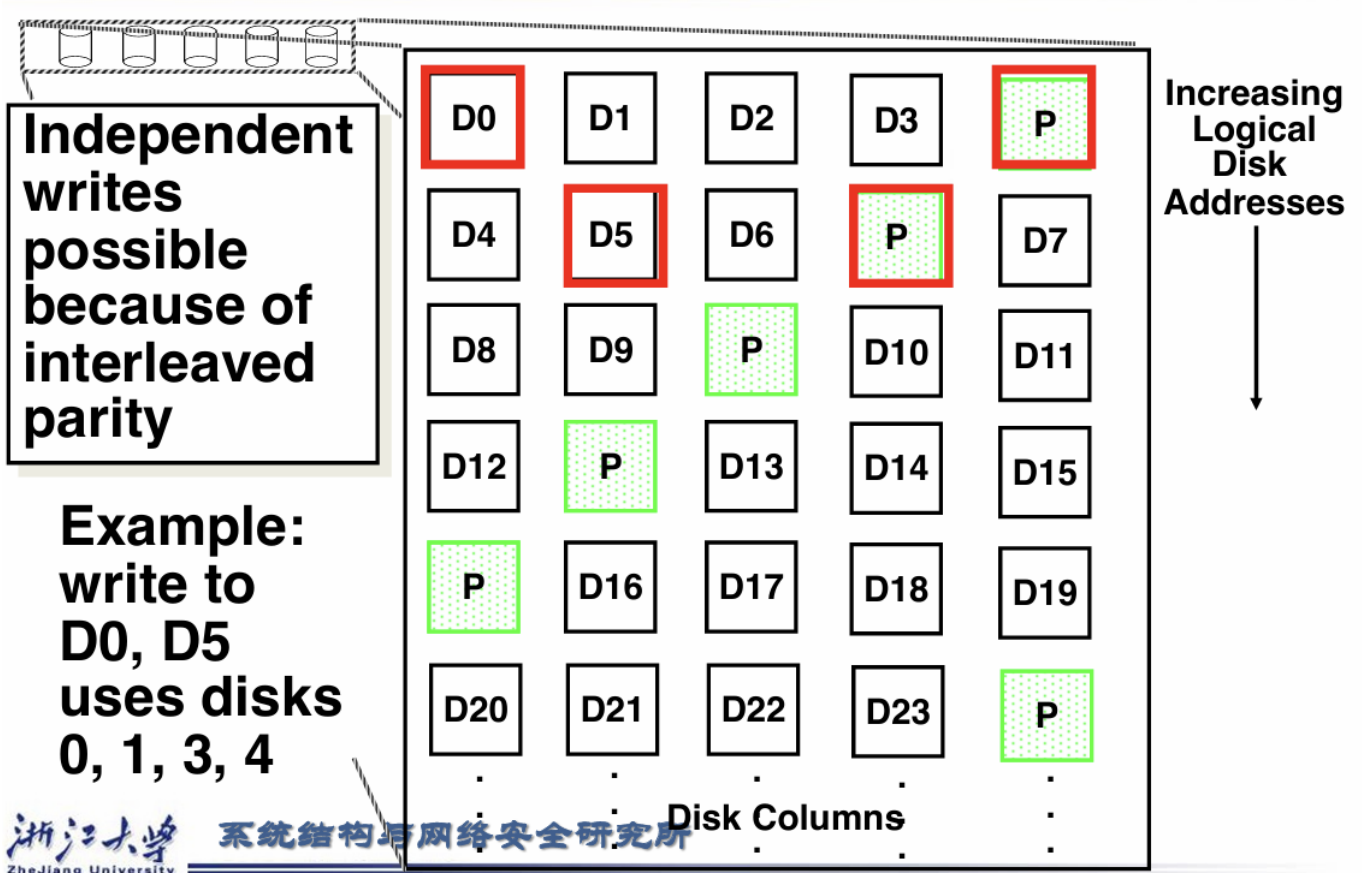


但是RAID4会遇到一个问题:



当同时写D0和D5的时候,由于对应的P都要被修改,而P又在同一个磁盘中,所以不能同时写,由此引出了RAID5

- `RAID 5`:把校验的块分配到不同的磁盘中



- `RAID 6`:有双重校验码,能够容忍两个磁盘同时损坏

总线(BUS)

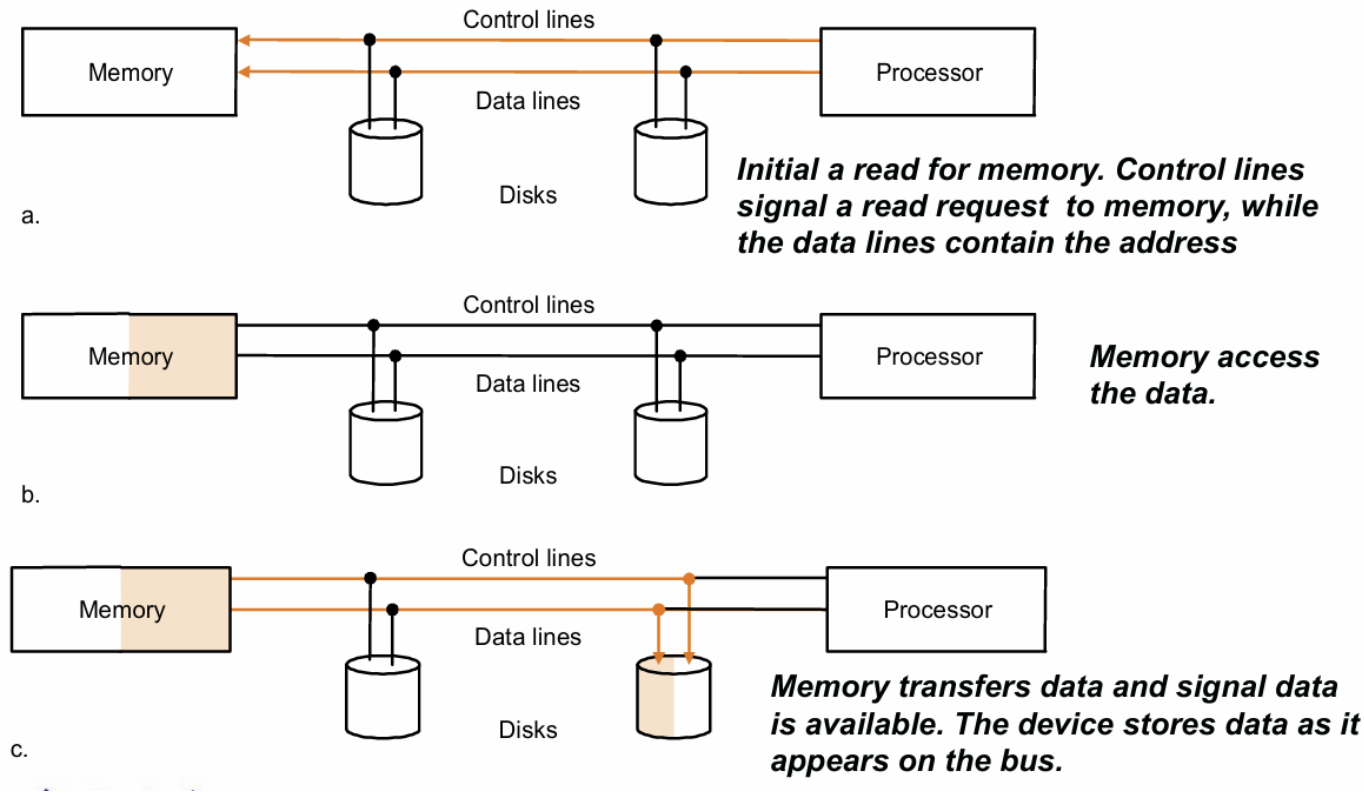
总线就是一条共享的数据传输线路

总线有两种不同的类型:

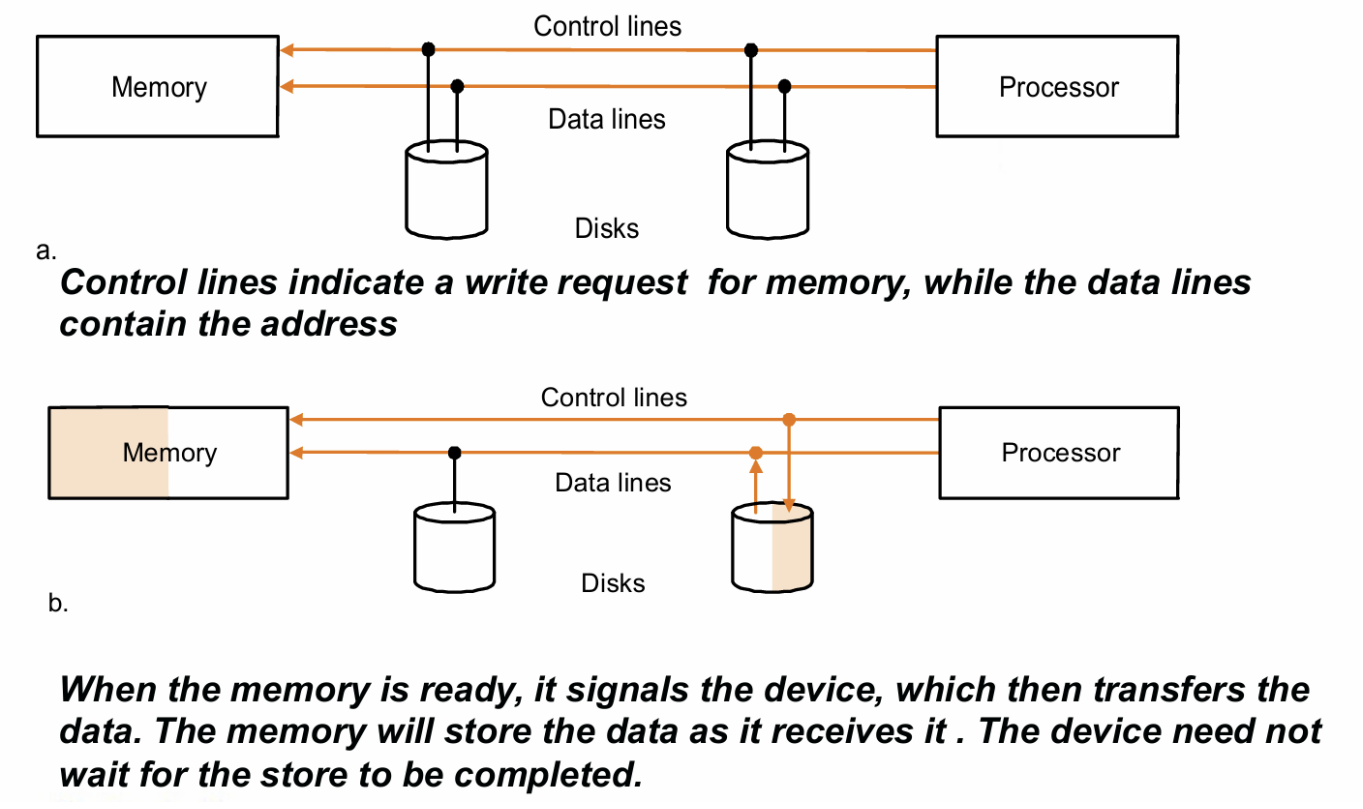
- 控制线:用来传输请求信号和接收信号,并且表明此时数据线路上的数据是什么类型
- 数据线:用来传输数据,包括data和地址和复杂的指令等

总线的传输分为两个部分:发送地址和接受或发出数据 两种操作:

- `input`:从设备传输数据到内存中



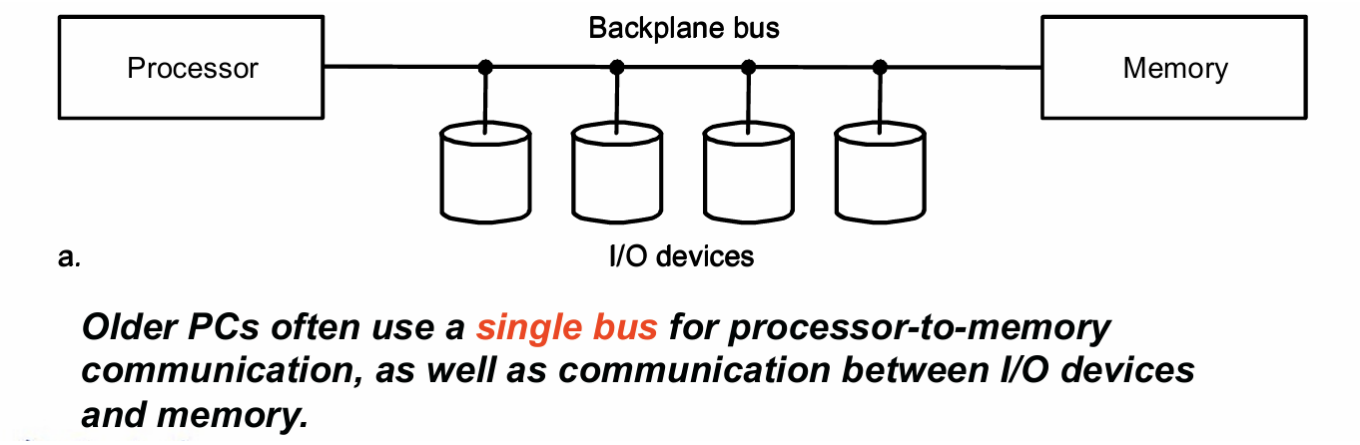
- `output`:从内存中传输数据到设备中



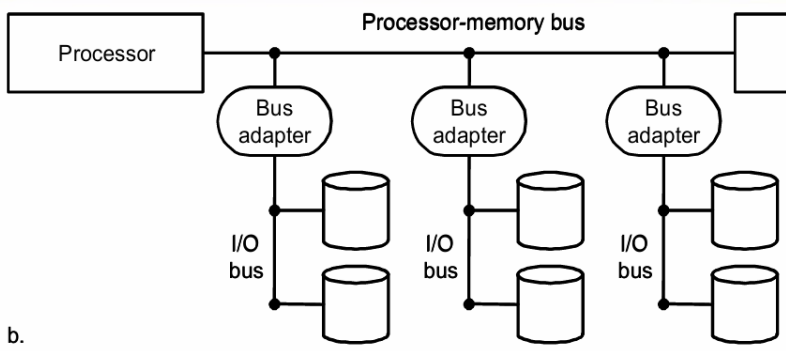
种类

总线也分为许多种类:

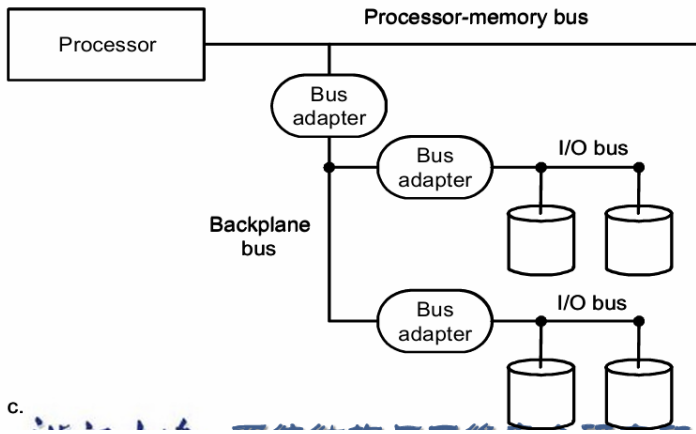
- `processor-memory`:短但快,是定制的
- `back-plane`:高速,通常是标准化的
- `I/O`:长,连接不同的设备,标准化的



Older PCs often use a single bus for processor-to-memory communication, as well as communication between I/O devices and memory.



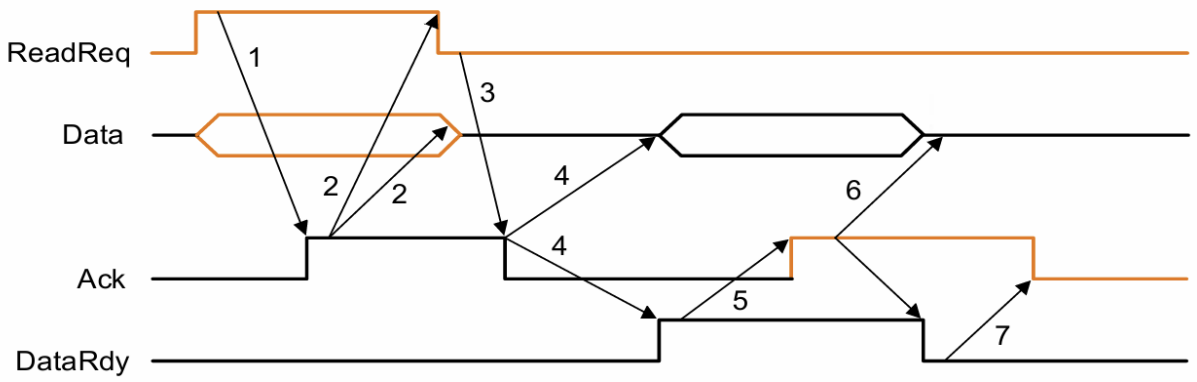
A separate bus is used for processor-memory traffic. The I/O bus use a bus adapter to interface to the processor-memory bus.



A separate bus is used for processor-memory traffic. A small number of backplane buses tap into the processor-memory bus.

同步与异步

- 同步总线:用时钟和固定的协议来传输数据,速度快并且规模小,但是所有的设备必须在同样的时钟下工作,并且由于时钟倾斜的存在(线路过长会导致传输时间大于一个时钟),该总线不能太长
- 异步总线:不使用时钟,用握手协议(Handshaking),所谓的握手,就是通过几个步骤来协调异步总线的传输,通过下面的例子来理解



1. 内存看到ReadReq信号,读取总线上的地址数据,并且开始内存的读取操作,然后拉高Ack信号告诉设备其发起的信号被接受了
2. I/O设备看到Ack高信号,就会释放掉ReadReq信号
3. 内存看到ReadReq信号降低了就知道I/O设备接受到了,这样一来就可以把Ack信号释放掉
4. 等到内存的数据准备好了后,内存将数据放到数据总线上,然后拉高DataRdy信号
5. I/O设备看到了DataRdy信号后,从总线中读取数据,然后升高Ack信号来表明自己接受到了数据
6. 内存看到Ack信号就知道,设备已经接受到了自己的数据,那么就可以释放掉DataRdy和data信号
7. 最后I/O设备看到DataRdy信号降低,就知道自己接受到的信号被内存所接受,这样一来就可以拉低Ack信号,表明一次数据传输的完成

通过两个例题计算同步异步总线的带宽来加深理解:

一.

Assume: **The synchronous bus has a clock cycle time of 50 ns, and each bus transmission takes 1 clock cycle .**
The asynchronous bus requires 40 ns per handshake.
The data portion of both buses is 32 bits wide.
Question: **Find the bandwidth for each bus when reading one word from a 200-ns memory.**

Answer: *synchronous bus:*

the bus cycles is 50 ns. The steps and times required for the synchronous bus are follows:

1. *Send the address to memory : 50ns*
2. *Read the memory : 200ns*
3. *Send the data to the device : 50ns*

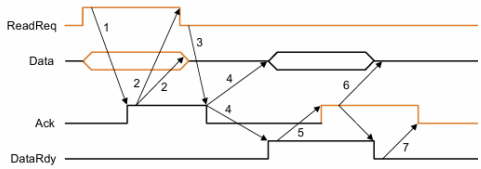
Thus, the total time is 300 ns. So,

$$\textit{the bandwidth} = 4bytes/300ns = 4MB/0.3seconds = 13.3MB/second$$

Answer: *asynchronous bus:*

we realize that several of the steps can be overlapped with the memory access time.

In particular, the memory receives the address at the end of step 1 and does not need to put the data on the bus until the beginning of step 5; steps 2, 3, and 4 can overlap with the memory access time.



This leads to the following timing:

step1: 40ns

step2,3,4: maximum(2×40ns+40ns,200ns)=200ns

step5,6,7: 3×40ns=120ns

Thus, the total time is 360ns, so

$$\textit{the maximum bandwidth} = 4bytes/360ns = 4MB/0.36seconds$$
$$= 11.1MB/second$$

Accordingly, the synchronous bus is only about 20% faster.

二.

Suppose we have a system with the following characteristic:

1. A memory and bus system supporting block access of 4 to 16 32-bit words
2. A 64-bit synchronous bus clocked at 200 MHz, with each 64-bit transfer taking 1 clock cycle, and 1 clock cycle required to send an address to memory.
3. Two clock cycles needed between each bus operation.
4. A memory access time for the first four words of 200ns; each additional set of four words can be read in 20 ns. Assume that a bus transfer of the most recently read data and a read of the next four words can be overlapped.

Find the sustained bandwidth and the latency for a read of 256 words for transfers that use 4-word blocks and for transfers that use 16-word blocks. Also compute effective number of bus transactions per second for each case.

一次读取4-word

Answer: the 4-word block transfers:

64-bit synchronous bus

each block takes

1. *1 clock cycle to send the address to memory*
2. *200ns/(5ns/cycle) = 40 clock cycles to read memory*
3. *2 clock cycles to send the data from the memory*
4. *Two clock cycles needed between each bus operation.*

This is a total of 45cycles.

There are *256/4 = 64 blocks*.

So the transfer of 256 words takes

$$45 \times 64 = 2880 \text{ clock cycles}$$

The latency for the transfer of 256 words is:

$$2880 \text{ cycles} \times (5\text{ns/cycle}) = 14{,}400\text{ns}.$$

so the number of bus transactions per second is:

$$64 \text{ transactions} \times \frac{1\text{second}}{14{,}400 \text{ ns}} = 4.44\text{M transactions/second}$$

The bus bandwidth is:

$$(256 \times 4)\text{bytes} \times \frac{1\text{second}}{14{,}400 \text{ ns}} = 71.11 \text{ MB/sec}$$

一次读取16-word

the first block requires

1. *1 clock cycle to send an address to memory*
2. *200ns or 40 cycles to read the first four words in memory.*
3. *2 cycles to transfer the data of the set, during which time the read of the next 4-word set is started.*
4. *It only takes 20ns or 4 cycles to read the next set. After the read is completed, the set will be transferred. Each of the three remaining sets requires repeating only the last two steps.*
5. *Two clock cycles needed between each bus operation.*

Thus, the total number of cycles for each 16- word block is:

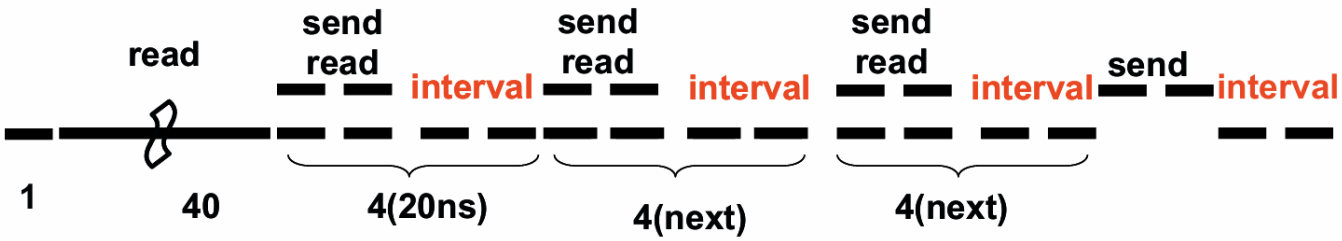
1+40+4×3+2+2=57 cycles.

There are 256/16=16 blocks.

so the transfer of 256 words takes *57×16=912 cycles*.

Thus the latency is:

912cycles × 5ns/cycles = 4560ns.



The number of bus transactions per second with 16-word blocks is:

$$16\ \text{transactions} \times \frac{1\text{second}}{4560\ \text{ns}} = 3.51\text{M transactions/second}$$

The bus bandwidth with 16-word blocks is:

$$(256 \times 4)\text{bytes} \times \frac{1\text{second}}{4560\ \text{ns}} = 224.56\ \text{MB/sec}$$

Now,let's put two bus bandwidth together:

4-word blocks: 71.11 MB/sec

16-word blocks:224.56 MB/sec

The bandwidth for the 16-word blocks is 3.16 times higher than for the 4-word blocks.

增加总线带宽的方式:

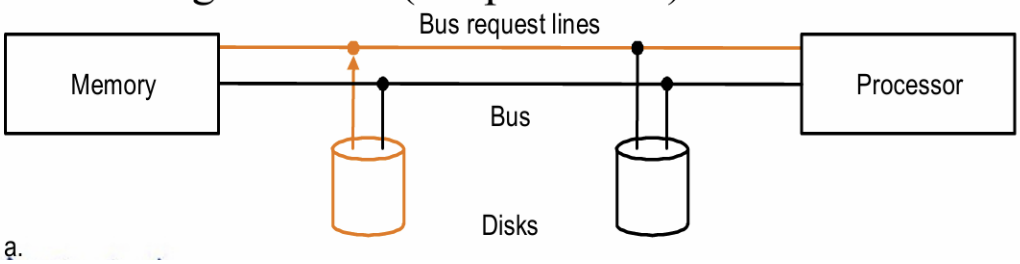
- 增加数据总线的宽度
- 将地址线和数据线分开
- 一次性传输多个字

总线仲裁

同一时刻可能有多个设备想要使用总线,这样一来就需要一个总线控制器(bus master)来控制这些请求,CPU可以视为一个控制器

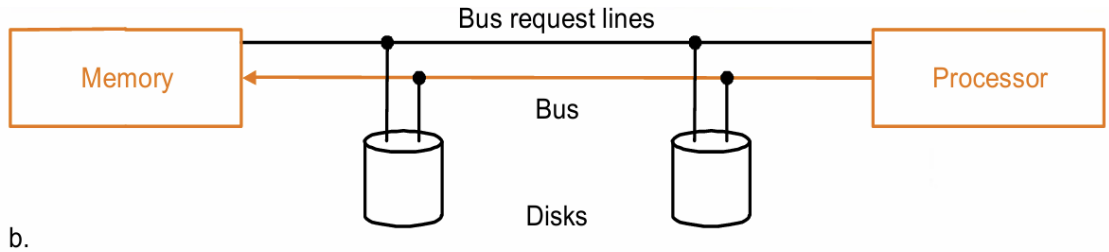
举一个简单的例子:

- Example: the initial steps in a bus transaction with a single master (the processor).



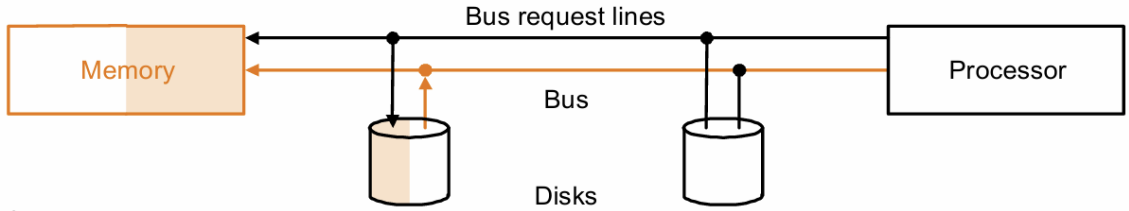
a.

First, the device generates a bus request to indicate to the processor that it wants to use the bus.



b.

The processor responds and generates appropriate bus control signals. For example, if the devices wants to perform output from memory, the processor asserts the read request lines to memory.



c.

The processor also notifies the device that its bus request is being processed; as a result, the device knows it can use the bus and places the address for the request on the bus.

四种仲裁的方法:

- 串联仲裁(Daisy Chain Arbitration):仲裁信号从一个设备传递到下一个设备直到总线的主设备(如CPU)接收到请求,存在不公平的现象,处于设备串联靠后的设备必须等待前面的设备使用完总线才能得到访问
- 集中式并行仲裁(Centralized,Parallel Arbitration):由一个集中式的仲裁器负责管理总线的访问请求,仲裁器根据一定的规则来选择一个设备使用总线
- 自我选择(Self-selection):每个设备都能根据某种方式自行决定是否能获得总线访问权
- 冲突检测(collision detection):设备在发送数据的时候会监听总线上的信号,如果发生冲突,会停止发送并重新尝试

对接I/O设备

三个I/O系统的特征:

- 被多个程序共享使用
- 通常使用中断来传输关于I/O操作的信息
- I/O设备的底层控制是十分复杂的

三种交流是需要的:

- 操作系统必须能够给I/O设备提供指令
- 设备必须能够通知操作系统,当其完成了一个操作或者是遇到了一个错误
- 数据必须能够在内存和I/O设备中传输

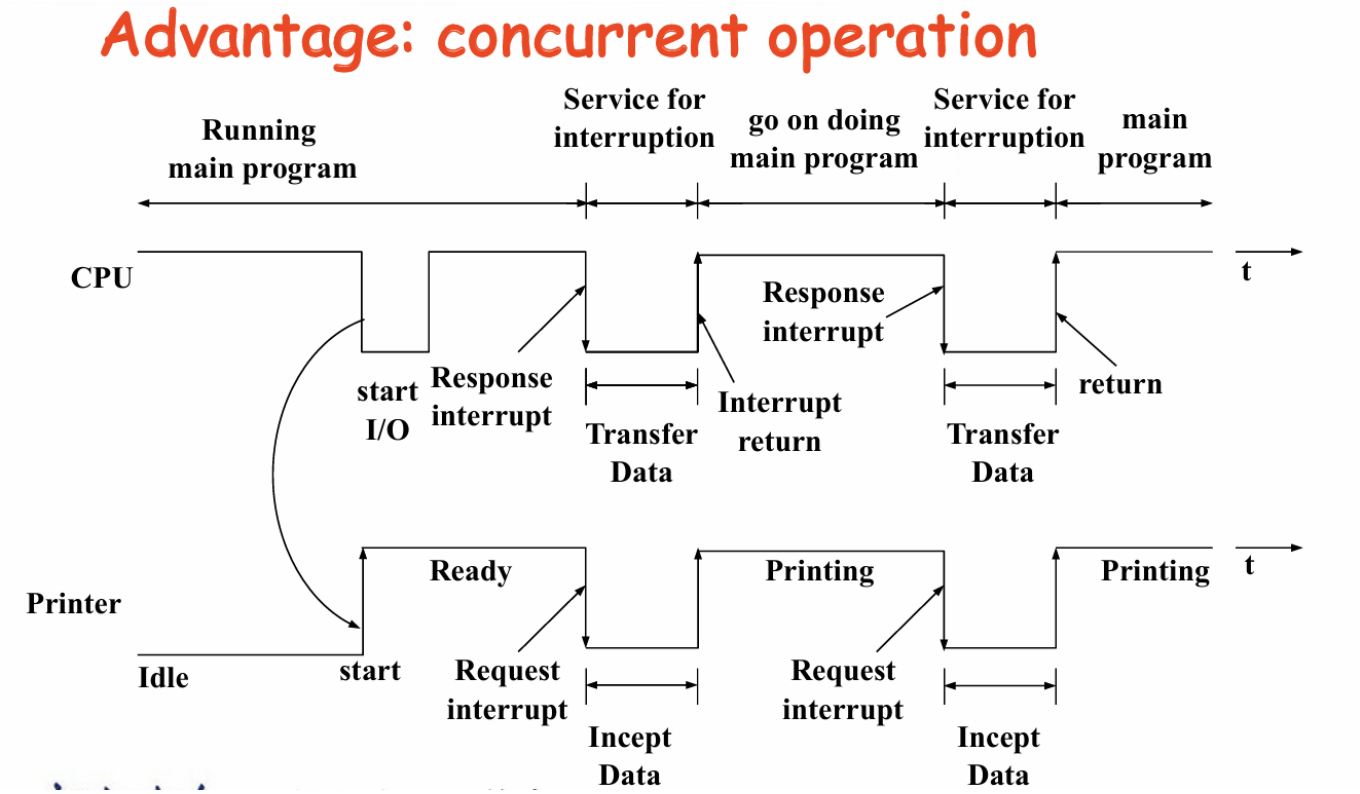
为了将控制信号传给I/O设备,必须要能够访问对应设备的地址,那么就需要考虑一个问题,I/O设备的地址是怎么规定的:

- `memory-mapped I/O`:内存的地址值一部分被划分给I/O设备,利用如lw sw的指令直接访问地址来访问设备

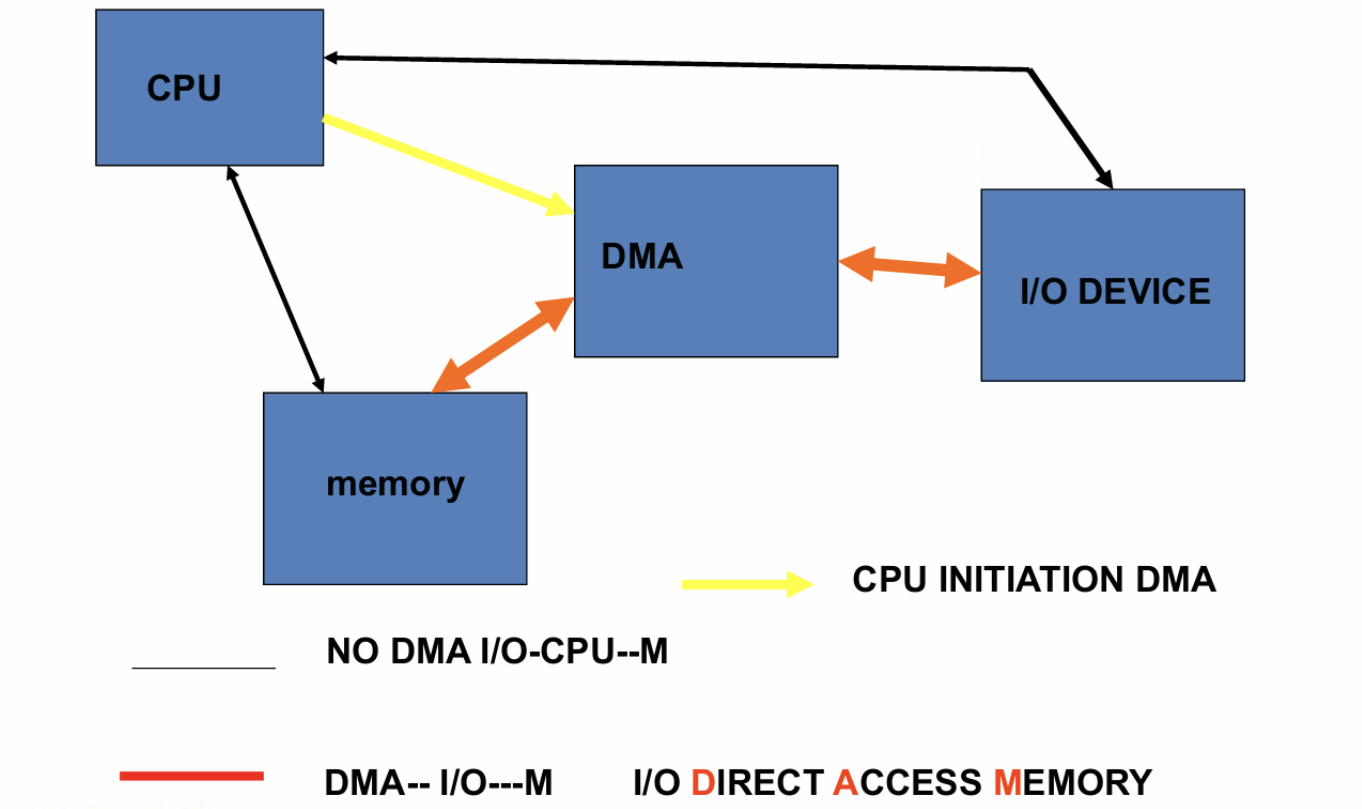
- `Port-Mapped I/O`:设备不适用内存空间进行寻址,而是通过专门的I/O端口来进行访问,即通过一些特定的指令专门用来访问设备

与CPU的交互方式:

- `Polling`(轮询):处理器定期检查设备的状态寄存器,来判断是否要进行下一次I/O操作
- `Interrupt`(中断):当一个设备想要进行某些操作的时候,会发起中断,让CPU在中断中处理



- `DMA`(direct memory access):无需通过处理器来转移数据,相当于一个助手,通过控制器来完成内存和设备之间数据交互



DMA传输需要三个步骤:

- 处理器设置DMA,包括对设备的识别,操作,数据要传输的内存地址,传输的bytes数等
- DMA开始进行对设备进行操作,同时还要对总线进行仲裁,如果在总线上的传输不止一次,那么自动计算出下一次要存放的内存地址
- 一旦DMA传输完成,控制器会对CPU发起中断,然后检测是否有错误发生

通过以下几个例题来对交互方式的不同产生更深刻的理解:

采用轮询方法:

Assume: that the number of clock cycles for a polling operation is 400 and that processor executes with a 500-Mhz clock.

Determine the fraction of CPU time consumed for the mouse, floppy disk, and hard disk.

We assuming that you poll often enough so that no data is ever lost and that those devices are potentially always busy.

We assume again that:

1. *The mouse must be polled 30 times per second to ensure that we do not miss any movement made by the user.*
2. *The floppy disk transfers data to the processor in 16-bit units and has a data rate of 50 KB/sec. No data transfer can be missed.*
3. *The hard disk transfers data in four-word chunks and can transfer at 4 MB/sec. Again, no transfer can be missed.*

the mouse:

clock cycles per second for polling :
30×400=12,000 cycles

Fraction of the processor clock cycles consumed is

$$\frac{12\times 10^3}{500\times 10^6} = 0.002\%$$

the floppy disk:

the number of polling access per second:

50KB/2B = 25K

clock cycles per second for polling: 25K×400cycles

Fraction of the processor clock cycles consumed:

$$\frac{10\times 10^6}{500\times 10^6} = 2\%$$

the hard disk:

The number of polling access per second :

4MB/16B = 250K

Clock cycles per second for polling = 250K×400

Fraction of the processor clock cycles consumed:

$$\frac{100\times 10^6}{500\times 10^6} = 20\%$$

Now,let's put three fractions together:

Mouse: 0.002%

Floppy disk: 2%

Hard disk: 20%

Clearly, polling can be used for the mouse without much performance impact on the processor, but it is unacceptable for a hard disk on this machine.

采用中断方法:

Suppose we have the same hard disk and processor we used in the former example, but we used interrupt-driven I/O. The overhead for each transfer, including the interrupt, is 500 clock cycles. Find the fraction of the processor consumed if the hard disk is only transferring data 5% of the time.

Answer: First, we assume the disk is transferring data 100% of the time. So, the interrupt rate is the same as the polling rate.

Cycles per second for disk is:

$$250K\times 500=125\times 10^6 \text{ cycles per second}$$

Fraction of the processor consumed during a transfer is:

$$\frac{125\times 10^6}{500\times 10^6} = 25\%$$

Now, we assume that the disk is only transferring data 5% of the time. The fraction of the processor time consumed on average is:

$$25\% \times 5\% = 1.25\%$$

As we can see, no CPU time is needed when an interrupt-driven I/O device is not actually transferring. This is the major advantage of an interrupt-driven interface versus polling.

采用DMA方法:

Suppose we have the same hard disk and processor we used in the former example.

Assume that the initial setup of a DMA transfer takes 1000 clock cycles for the processor, and assume the handling of the interrupt at DMA completion requires 500 clock cycles for the processor.

The hard disk has a transfer rate of 4MB/sec and uses DMA. The average transfer from disk is 8 KB. Assume the disk is actively transferring 100% of the time.

Please find what fraction of the processor time is consumed.

Time for each 8KB transfer is:

$8KB/(4MB/second)=2\times10^{-3}seconds.$

It requires the following cycles per second:

$$\frac{(1000+500)\ \frac{cycles}{transfer}}{2\times 10^{-3}\ \frac{seconds}{transfer}} = 750\times 10^{3}\ \frac{clock\ cycles}{second}$$

Fraction of processor time: $\frac{750\times 10^{3}}{500\times 10^{6}} = 0.2\%$

Unlike either polling or interrupt-driven I/O, DMA can be used to interface a hard disk without consuming all the processor cycles for a single I/O.

设计I/O系统

找到整个I/O系统中最薄弱的地方,也就是会限制I/O道路的组件,工作负载(workload)和配置(configuration)都可能表明了这个薄弱的地方在哪

通过一个例子来理解:

Examples:

Consider the following computer system:

1. **A CPU sustains 3 billion instructions per second and it takes average *100,000* instructions in the operating system per I/O operation.**
2. **A memory backplane bus is capable of sustaining a transfer rate of 1000 MB/sec.**
3. **SCSI-Ultra320 controllers with a transfer rate of 320 MB/sec and accommodating up to 7 disks.**
4. **Disk drives with a read/write bandwidth of 75 MB/sec and an average seek plus rotational latency of 6 ms.**

If the workload consists of 64-KB reads (assuming the the data block is sequential on a track), and the user program need *200,000* instructions per I/O operation, please find the maximum sustainable I/O rate and the number of disks and SCSI controllers required.

Answer:

The two fixed component of the system are the memory bus and the CPU. Let's first find the I/O rate that these two components can sustain and determine which of these is the bottleneck.

$$\text{Maximum I/O rate of CPU} = \frac{\text{Instruction execution rate}}{\text{Instruction per I/O}} = \frac{3 \times 10^9}{(200 + 100) \times 10^3} = 10000 \frac{\text{I/Os}}{\text{seconds}}$$

$$\text{Maximum I/O rate of bus} = \frac{\text{Bus bandwidth}}{\text{Bytes per I/O}} = \frac{1000 \times 10^6}{64 \times 10^3} = 15625 \frac{\text{I/Os}}{\text{seconds}}$$

The CPU is the bottleneck, so we can now configure the rest of the system to perform at the level dictated by the bus, 10000 I/Os per second.

Now, let's determine how many disks we need to be able to Accommodate 10000 I/Os per second. To find the number of disks, we first find the time per I/O operation at the disk:

$$\textit{Time per I/O at disk} = \text{Seek/rotational time} + \text{Transfer time}$$

$$= 6\ ms + \frac{64KB}{75MB/sec} = 6.9\ ms$$

This means each disk can complete 1000ms/6.9ms, or 146 I/Os per second. To saturate the bus,the system need 10000/146≈69 disks.

To compute the number of SCSI buses, we need to know the average transfer rate per disk, which is given by:

$$\textit{Transfer rate} = \frac{\textit{Transfer size}}{\textit{Transfer time}} = \frac{64KB}{6.9ms} \approx 9.56MB/sec$$

$$69 \times 9.56MB/sec < 320MB/s$$

$$69 \div 7 \approx 10$$

Assuming the disk accesses are not clustered so that we can use all the bus bandwidth, we can place 7 disks per bus and controller. This means we will need 69/7, or 10 SCSI buses and controllers.

最后那边应该是7 * 9.56 < 320 表示此时带宽不是限制,而是数量