



存储系统的可靠性

llxx@ustc.edu.cn

内容提要



- 数据存储与传输的可靠性与容错技术
- 校验码技术, RV\$5.5, 掌握
 - 编码纠错理论
 - 海明码: 存储器, 唐\$4.2.6
 - 分组校验
 - CRC码: 磁盘, 网络, 唐\$4.4.6
- RAID技术: 磁盘阵列, RV\$5.11, 了解
 - 海明, 奇偶



Google2009：数据存储设备可靠性

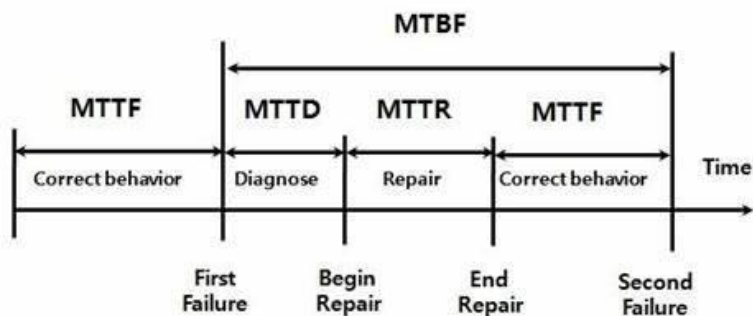
- DRAM错误率超出人们预想
 - 错误类型：软错误、硬错误
 - 由电磁干扰或者硬件故障所导致
 - 软错误：很少损坏字位，是可修正的；
 - 硬错误：会损坏字位而成为物理缺陷，从而造成数据错误的反复发生。
 - “可能成为系统宕机和服务中断的罪魁祸首”
 - DIMM中有约8.2%受到了可修正错误的影响
 - 平均一个DIMM每年发生3700次可修正错误
- 硬盘：数据失效率高达6%（厂商：2%）
 - 错误类型：位跳变（可由ECC纠错），物理损坏



FAULT-TOLERANT COMPUTING



- **Failure(失效/故障)**: When a component is not living up to its specifications, a failure occurs
- **Error(错误)**: The part of a component's **state** that can lead to a failure
- **Fault(缺陷/故障)**: The cause of an error. Types:
 - **Transient(偶发)**: occur once, then disappear
 - **Intermittent(间歇)**: occur, then vanish, then reappear
 - **Permanent(持久)**: continues to exist
- **指标**
 - 可靠性: $MTBF = MTTF + MTTR$, 年失效率AFR
 - 可用性: $MTTF / (MTTF + MTTR) = \text{工作时间} / (\text{工作时间} + \text{维修时间})$



9' s	Availability	Downtime/Year	Examples
1	90.0%	36 days 12 hours	Personal clients
2	99.0%	87 hours 36 minutes	Entry-level businesses
3	99.9%	8 hours 46 minutes	ISPs, mainstream businesses
4	99.99%	52 minutes 33 seconds	Data centers
5	99.999%	5 minutes 15 seconds	Carrier-grade Telco, medical, banking
6	99.9999%	31.5 seconds	Military defense system

Fault Tolerance: Redundancy

- 提高MTTF：避免，容忍，预测
 - 故障“难以消除（eliminate），只能掩盖（mask）”
- 容错计算系统：出现一定限度的失效时，依然能够提供所需要的服务。
 - 服务降级：使之不影响系统的正常使用
 - 容错能力：检错（发现，定位），纠错
- 冗余技术类型
 - Information redundancy
 - Eg, a Hamming code can be added to transmitted data to recover from noise on the transmission line.
 - Time redundancy
 - is especially helpful for transient or intermittent faults.
 - Eg, using transactions(回滚, rollback)
 - Physical redundancy
 - Eg, 747s have four engines but can fly on three
 - RAID

Error Checking and Correcting

- **Two** major types of data errors can occur in data transmission:
 - **hard** errors, which are permanent, arise from broken interconnects, internal shorts, or open leads
 - **soft** errors, which are transient, are caused by system noise, power surges, and alpha particles.
- The **processor** (MIPS R4000) verifies data correctness by using either the **parity** or the **SECDED** code as it
 - passes data from the System interface to the secondary cache, or
 - it moves data from the secondary cache to the primary caches or to the System interface.

奇偶编码校验 (Parity Check Code)



- 编码规则

- 在被传送的 n 位代码($b_{n-1}b_{n-2}\dots b_1b_0$)上 (最后) 增加一位校验位 P (Parity), 将原数据与奇 (偶) 校验位 (生成算法) 一起进行存取或传送(即传送 $Pb_{n-1}b_{n-2}\dots b_1b_0$)。

- 奇校验: 使 “1”的个数为奇数

- 0000 0000 - >0000 0000 **1**

- 0000 0001 - >0000 0001 **0**

- 偶校验: 使 “1”的个数为偶数

- 0000 0000 - >0000 0000 **0**

- 0000 0001 - >0000 0001 **1**

- 为什么能容错? 具有什么样的容错能力?



0000

0000

0001

0001

0010

0010

合法编码

0011

0011

0100

0100

0101

0101

非法编码

4位表示

0110

4位表示

0110

16个状态

0111

8个状态

0111

不能检出错误!

1000

可能检出错误!

1000

1001

1001

1010

1010

1011

1011

1100

1100

1101

1101

1110

1110

1111

1111

《通信原理》

码距：海明（Hamming）距离

两个等长码字之间对应位不同的个数

编码系统的码距

任意两个码字的最小码距

奇偶，格雷，海明，Reed-Solomon?



编码纠错理论

- 任何一种编码是否具有检测能力或纠错能力，都与编码的**最小距离**有关。
- 根据纠错理论，有： $L-1 = D + C$ ，且 $D \geq C$
 - 即编码最小距离 L 越大，则检错的位数 D 越大，纠错的位数 C 也越大，且纠错能力恒小于等于检测能力。
 - 例如， $L = 3$ ，则 $D = 2$ ， $C = 0$ ；或 $D = 1$ ， $C = 1$ 。
 - 增大 L ，提高检错和纠错能力。——位数
- 应用
 - 内存：奇偶，ECC（错误检查和纠正），SECDED
 - 硬盘：CRC
 - 通信：奇偶（串行，物理层），海明，CRC（网络层）



奇偶编码校验

- 在被传送的 n 位数据($b_{n-1}b_{n-2}\dots b_1b_0$)上增加一位校验位 P
 - 将原数据和得到的奇（偶）校验位一起进行存取或传送，即传送 $Pb_{n-1}b_{n-2}\dots b_1b_0$

$$\text{偶校验: } r_i = I_{1i} + I_{2i} + \dots + I_{pi} \quad (i=1, 2, \dots, q),$$

$$\text{奇校验: } r_i = I_{1i} + I_{2i} + \dots + I_{pi} + 1 \quad (i=1, 2, \dots, q)$$

- 码距?
- 只能检错，不能纠错
 - 可以发现“奇数”个错：一位出错的概率高
- “数据位增加，Parity检验位**成倍**增加”
 - 当数据位为16位时需要增加2位用于检查，当数据位为32位时则需增加4位？

奇偶校验的实现 (VHDL)



entity IPAR is

generic(PROP_DEL: time);

port(R: in Std_logic_vector(7 down to 0); P: out Std_logic);

end IPAR;

architecture LOOP4 of IPAR is

signal CLOCK: Std_logic := '0';

begin

OPAR: process(R)

variable X: Std_logic;

begin

X := '0';

for I in 7 downto 0 loop

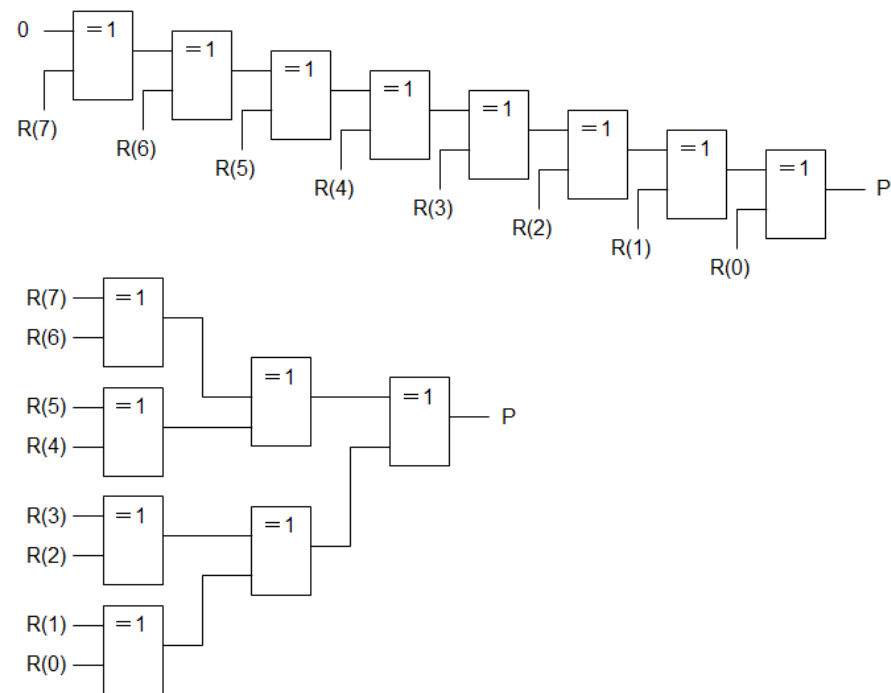
X := X xor R(I);

end loop;

P <= X after PROP_DEL;

end process;

end LOOP4;



偶校验生成电路

验证电路？

交叉奇偶校验 (ECC)



数据块的横向和纵向都进行奇偶校验位。例如：

	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	横向校验位
第1字节	1	1	0	0	1	0	1	1	→ 1
第2字节	0	1	1	1	1	1	0	0	→ 1
第3字节	1	0	0	1	1	0	1	0	→ 0
第4字节	1	0	0	1	0	1	0	1	→ 0
	↓	↓	↓	↓	↓	↓	↓	↓	
纵向校验位	1	0	1	1	1	0	0	0	

Error Correcting Code: “检两位，纠一位”？

ECC memory

1位纠错 Hamming码校验码(校验位数)



- 设有 k 位数据, r 位校验位。
- r 位校验位有 2^r 个组合。
 - 若用0表示无差错, 则剩余 2^r-1 个值表示有差错, 并指出错在第几位。
- 由于差错可能发生在 k 个数据位中或 r 个校验位中, 因此有: $2^r-1 \geq r+k$

– 海明码需要几位校验码? 唐表4.2

数据位 k	校验位 r	总位数 n
1	2	3
2~4	3	5~7
5~11	4	9~15
12~26	5	17~31
27~57	6	33~63
58~120	7	65~127

Richard Hamming@1950
1968年图灵奖



海明校验码(校验位置)

– 校验位和数据位是如何排列的

校验位排列在 2^{i-1} ($i = 0, 1, 2, \dots$) 的位置上

例1：有一个4位数为 $D_4D_3D_2D_1$ ，需要3位校验码 $P_3P_2P_1$ ，由此生成一个海明码

7	6	5	4	3	2	1
D_4	D_3	D_2	P_3	D_1	P_2	P_1
			2^2		2^1	2^0

例2：有一字节的信息需生成海明码，需要4位校验码

12	11	10	9	8	7	6	5	4	3	2	1
D_8	D_7	D_6	D_5	P_4	D_4	D_3	D_2	P_3	D_1	P_2	P_1
				8				4		2	1



校验位取值公式及计算举例

- 海明码的校验位 P_i 和数值位 D_i 的关系——**分组校验**
 - 设 k 位数据， r 位校验码，把 $k+r=m$ 个数据记为 $H_m H_{m-1} \dots H_2 H_1$ (海明码)，每个校验位 P_i 在海明码中被分配在 2^{i-1} 位置上。
 - H_i 由多个校验位校验：每个海明码的位号要等于参与校验它的几个校验位的位号之和。**
- 分解 $i = 2^{j_1} + 2^{j_2} + \dots + 2^{j_x} \ (j_1 \neq j_2 \neq \dots \neq j_x)$
- 得：H3由P1和P2校验，H5由P1和P4校验，H6由P2和P4校验，H7由P1、P2和P4校验，。。。
- 即： P_i 参与第 j_1 、 j_2 、...、 j_x 个校验位的计算(P1参与H3、H5、H7、...)

类RV图5-23

海明码位号	H_{12}	H_{11}	H_{10}	H_9	H_8	H_7	H_6	H_5	H_4	H_3	H_2	H_1
数据位/校验位	D_8	D_7	D_6	D_5	P_4	D_4	D_3	D_2	P_3	D_1	P_2	P_1
参与校验的校验位位号	4, 8	1, 2, 8	2, 8	1, 8	8	1, 2, 4	2, 4	1, 4	4	1, 2	2	1

例：



每个数据位至少出现在两个Pi值的形成关系中。当任一数据位发生变化时，必将引起二或三个Pi值跟着变化。

数据位为**1011**（偶校验）

$$P_3 = D_4 \oplus D_3 \oplus D_2$$

$$0 = 1 \oplus 0 \oplus 1$$

$$P_2 = D_4 \oplus D_3 \oplus D_1$$

$$0 = 1 \oplus 0 \oplus 1$$

$$P_1 = D_4 \oplus D_2 \oplus D_1$$

$$1 = 1 \oplus 1 \oplus 1$$

H	7	6	5	4	3	2	1
	D_4	D_3	D_2	P_3	D_1	P_2	P_1
2^2	D_4	D_3	D_2	P_3			
2^1	D_4	D_3			D_1	P_2	
2^0	D_4		D_2		D_1		P_1

最后，海明码为**1010101**



海明码的纠错原理

- 海明码的接收端的公式:

- $S_3 = P_3 \oplus D_4 \oplus D_3 \oplus D_2$

- $S_2 = P_2 \oplus D_4 \oplus D_3 \oplus D_1$

- $S_1 = P_1 \oplus D_4 \oplus D_2 \oplus D_1$

- 假定 海明码10**1**0101在传送中变成了10**0**0101

- $S_3 = P_3 \oplus D_4 \oplus D_3 \oplus D_2 = 0 \oplus 1 \oplus 0 \oplus \textcolor{red}{0} = \textcolor{red}{1}$

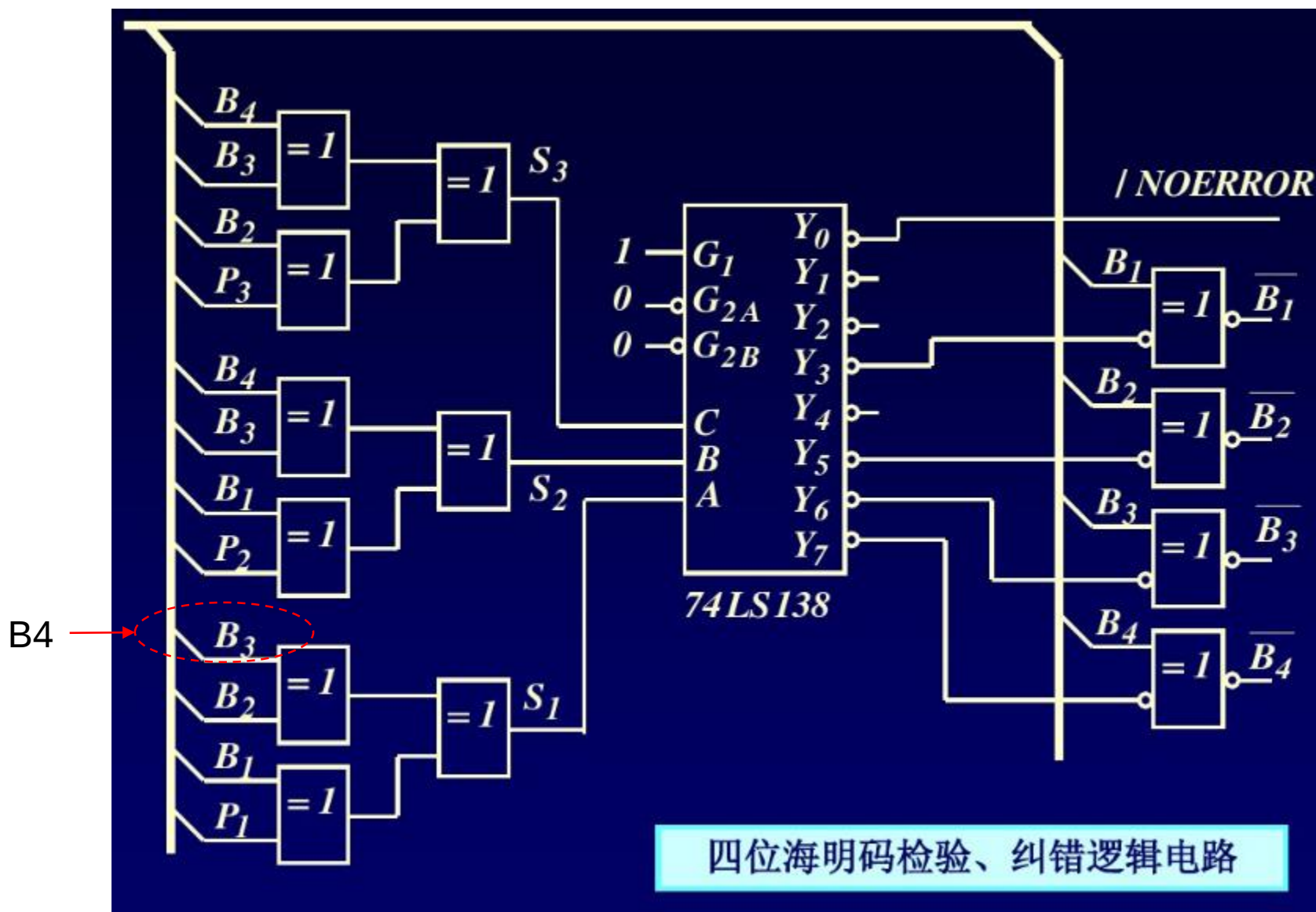
- $S_2 = P_2 \oplus D_4 \oplus D_3 \oplus D_1 = 0 \oplus 1 \oplus 0 \oplus 1 = 0$

- $S_1 = P_1 \oplus D_4 \oplus D_2 \oplus D_1 = 1 \oplus 1 \oplus \textcolor{red}{0} \oplus 1 = \textcolor{red}{1}$

因此, 由 $S_3S_2S_1 = 101$, 指出第5位错, 应由0变1

如何实现纠错? 译码器

实现：有错？



1位纠错海明码的应用

- 上述海明码称为单纠错码 (SEC)
 - 是否可以发现多位错? 奇数位错? 偶数位错?
- 通常半导体存储器采用SEC-DED
 - 单纠错-双检错码
 - SEC-DED与SEC相比需要增加1个附加位。
 - 在IBM3000系列中, 主存64位数据采用8位SEC-DED码进行校验(下右);
 - VAX计算机中32位字长机器, 采用7位SEC-DED码。





CRC校验码

CRC (Cyclic Redundancy Check)



- CRC: 各类介质存储器、数据通信
- 基于模2运算: 不考虑进位和借位
 - 模2加减运算: 异或 (相同为 “0”, 不同为 “1”)
 - 模2乘: 按模2加求部分积之和
 - 模2除: 按模2减求部分余数
 - 部分余数首位为1, 商1
 - 部分余数首位为0, 商0
 - 每上商一次, 部分余数减少一位。
 - 部分余数位数少于除数位数时, 结束

模2运算举例



$$0 \pm 1 = 1 \quad 0 \pm 0 = 0 \quad 1 \pm 0 = 1 \quad 1 \pm 1 = 0$$

$$\begin{array}{r} 1010 \\ \times 101 \\ \hline 1010 \\ 0000 \\ 1010 \\ \hline 100010 \end{array}$$

$$\begin{array}{r} 101 \\ \hline 101 \overline{) 10000} \\ \underline{101} \\ 0010 \\ \underline{000} \\ 0100 \\ \underline{101} \\ 001 \end{array}$$

余数=01

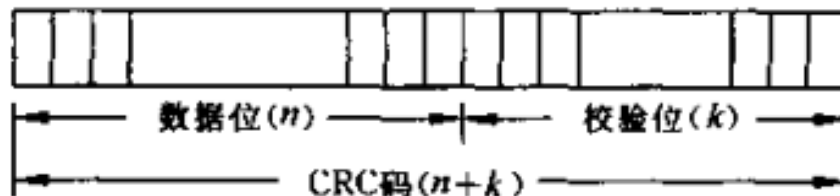


CRC校验步骤

• CRC生成

- 将 n 位数据 $D_{n-1}, \dots, D_0$ 用 $n-1$ 次多项式 $M(x)$ 表示, 即
$$M(x) = D_{n-1}x^{n-1} + D_{n-2}x^{n-2} + \dots + D_1x^1 + D_0x^0$$
- 将 $M(x)$ 左移 k 位 (补0), 即得: $M(x)*x^k$
- 将 $M(x)*x^k$ 除以 $k+1$ 位的生成多项式 $G(x)$, 余数即为 k 位的CRC校验位
- 将CRC校验位拼装在 $D_{n-1}, \dots, D_0$ 之后, 成为 $n+k$ 位数据, 也称 $(n+k, n)$ 码。

CRC码的组成



CRC码—编码举例



例:有效信息为1100,生成多项式 $G(x)=1011$,将其编成CRC码.

- 解: 数据位数 $n=4$, 校验位数 $k=G(x)$ 位数-1=3

$$M(x) = x^3 + x^2 = 1100$$

$$M(x) \cdot x^3 = x^6 + x^5 = 1100\textcolor{red}{000}$$

$$G(x) = x^3 + x + 1 = 1011$$

$$\frac{M(x) \cdot x^3}{G(x)} = \frac{1100000}{1011} = 1110 + \frac{010}{1011}$$

$$\begin{aligned} M(x) \cdot x^3 + R(x) &= 1100000 + 010 \\ &= 1100010 \end{aligned}$$

编好的循环校验码称为(7,4)码,即 $n+k=7$, $n=4$



CRC的译码与纠错

- 将收到的 $n+k$ 位CRC码用**约定**的生成多项式 $G(x)$ 去除
 - 正确，则余数 = 0。
 - 如果某一位出错，则余数不为0。不同位出错，余数不同。
- 余数和出错位之间的对应关系不变。
 - 与待测码字无关，与码制和生成多项式有关

对应 $G(x)=1011$ 的(7,4)码的**出错模式**

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	余数
正确	1	1	0	0	0	1	0	0 0 0
A_7 错	1	1	0	0	0	1	1	0 0 1
A_6 错	1	1	0	0	0	0	0	0 1 0
A_5 错	1	1	0	0	1	1	0	1 0 0
A_4 错	1	1	0	1	0	1	0	0 1 1
A_3 错	1	1	1	0	0	1	0	1 1 0
A_2 错	1	0	0	0	0	1	0	1 1 1
A_1 错	0	1	0	0	0	1	0	1 0 1



CRC的译码与纠错 (con't)

- **余数循环**

- 如果对余数补0, 除以 $G(x)$, 得下一余数。
- 继续除下去, 各次余数将按右表顺序循环。
- 特定生成多项式的余数模式固定

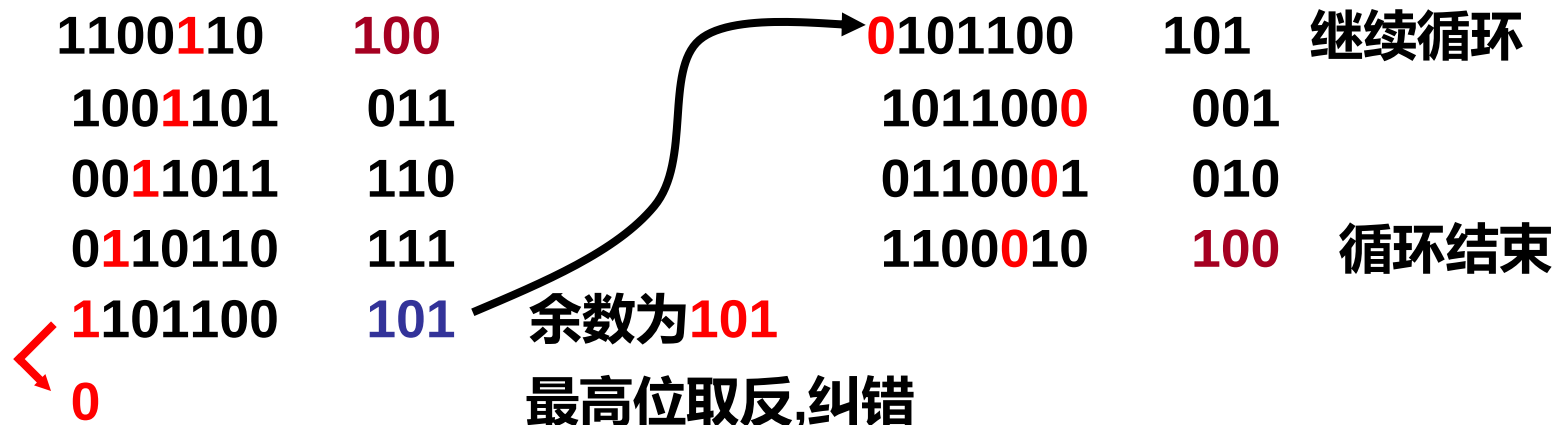
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	余数
正确	1	1	0	0	0	1	0	0 0 0
A_7 错	1	1	0	0	0	1	1	0 0 1
A_6 错	1	1	0	0	0	0	0	0 1 0
A_5 错	1	1	0	0	1	1	0	1 0 0
A_4 错	1	1	0	1	0	1	0	0 1 1
A_3 错	1	1	1	0	0	1	0	1 1 0
A_2 错	1	0	0	0	0	1	0	1 1 1
A_1 错	0	1	0	0	0	1	0	1 0 1

$$1011 \overline{) 0010}$$

CRC码的纠错方法 - 循环移位法



- 将CRC码进行左循环移位，至**出错位**被移至最高位
 - 余数添0继续除法，当余数为**101**时，出错位被移到最高位
- 对最高位取反，纠错
- 继续循环移位，直至**循环一周**
 - 继续余数除法，直至余数变成第一次的余数。
- 例A₅出错





生成多项式 $G(x)$

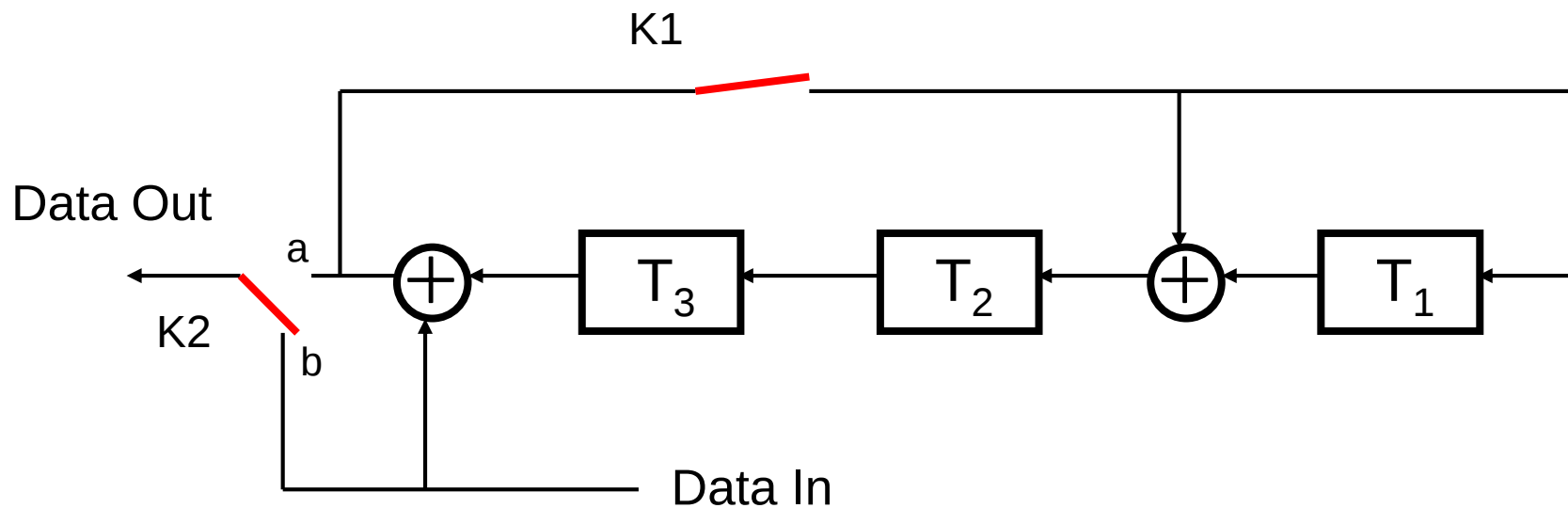
- 生成多项式 $G(x)$ 应能满足下列要求:
 - 任何一位发生错误都应使余数不为0.
 - 不同位发生错误应当使余数不同.
 - 对余数继续作模2除,应使余数循环.
 - 例:
 - $G(x)=x+1=11$ (7,6)码,判一位错
 - $G(x)=x^3+x+1=1011$ (7,4)码,判二位错或纠一位错
 - $G(x)=x^3+x^2+1=1101$ (7,4)码,判二位错或纠一位错
 - $G(x)=(x+1)(x^3+x+1)=11101$ (7,3)码,判二位错并纠一位错



CRC电路实现：移位寄存器

- (7, 4) 的CRC实现

1. K1闭合, K2打到b, 4步之后, T3T2T1为余数。
2. K1断开, K2打到a, 移位送出余数, 形成CRC码



CRC-CCITT实现



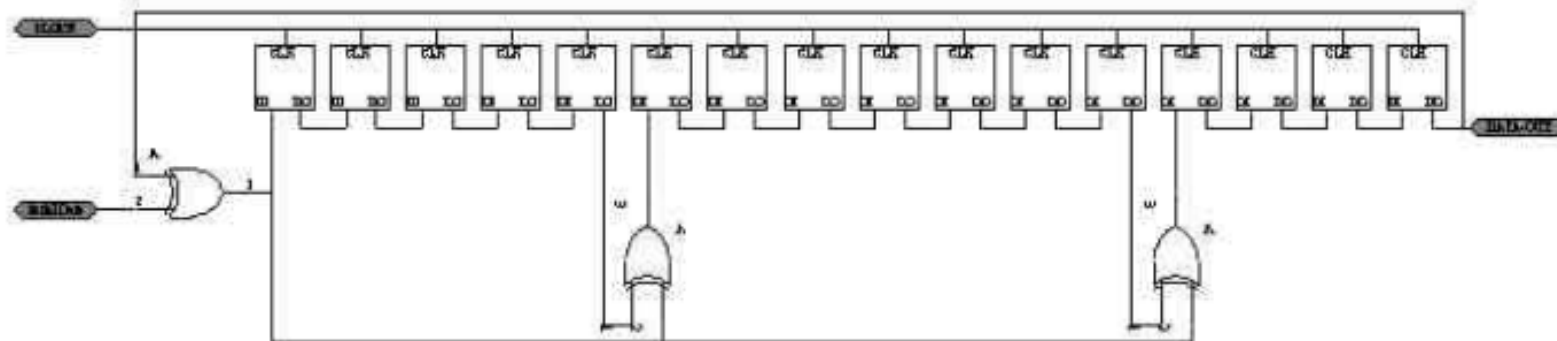
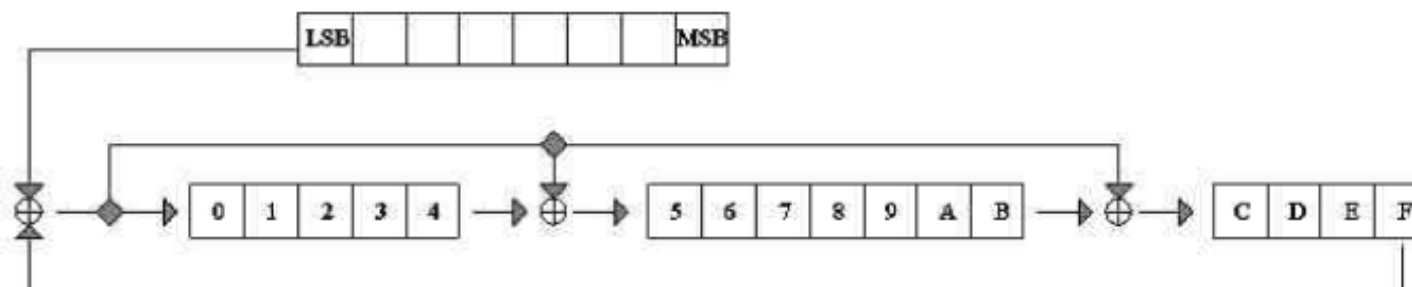
$$\text{CRC8} = X^8 + X^5 + X^4 + 1$$

$$\text{CRC-CCITT} = X^{16} + X^{12} + X^5 + 1$$

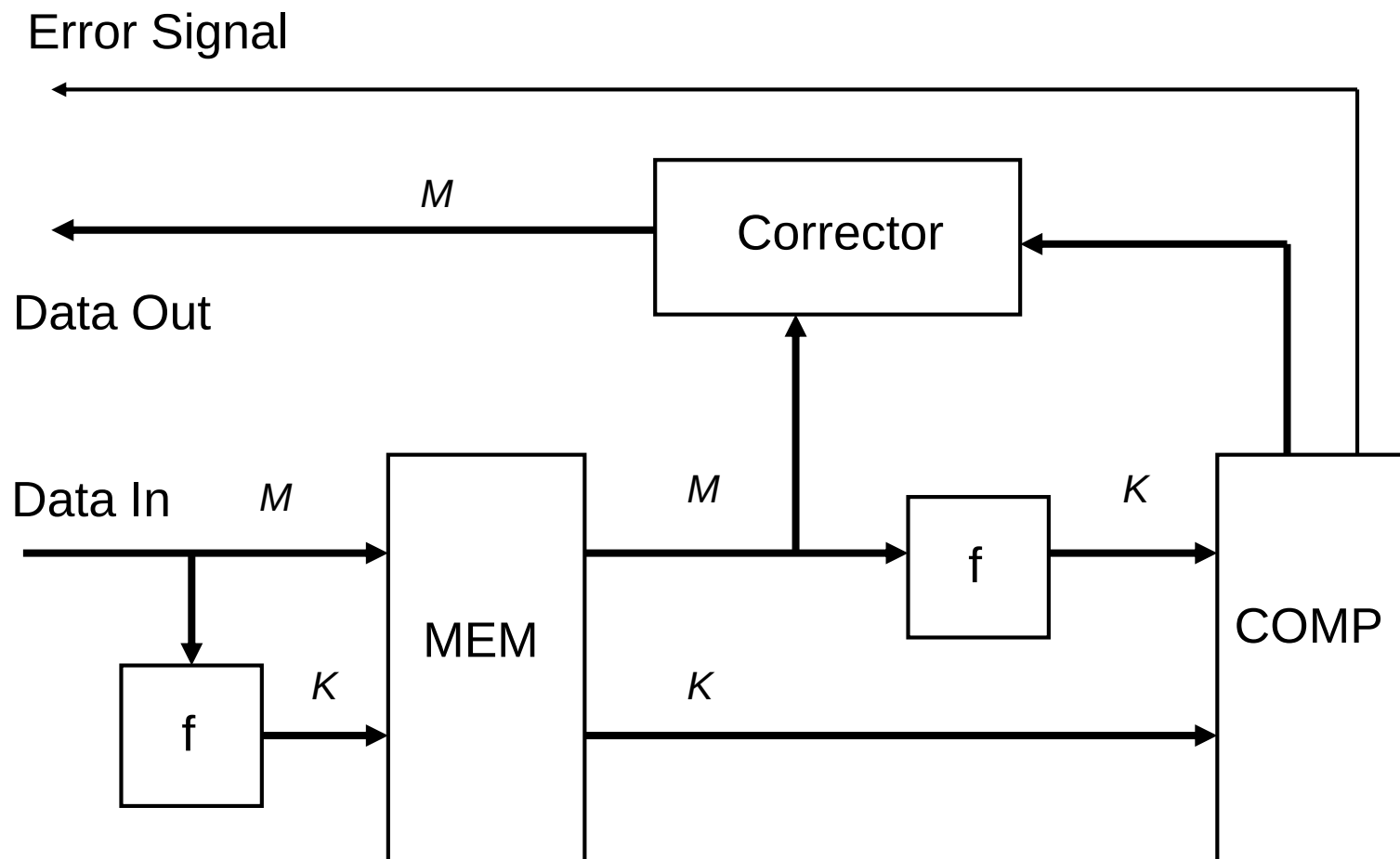
$$\text{CRC16} = X^{16} + X^{15} + X^5 + 1$$

$$\text{CRC12} = X^{12} + X^{11} + X^3 + X^2 + 1$$

$$\text{CRC32} = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$



存储校验系统



RAID技术：性能，可靠性，RV\$5.11



- 1987年，Patterson等@UCB
 - 将多只容量较小的、相对**廉价**的硬盘组合，使其**性能**超过一只昂贵的大硬盘
 - 性能（存储速度）
 - 可靠性
- Redundant Array of Inexpensive/Independent Disk
 - 支持自动检测故障硬盘；
 - 支持重建硬盘坏轨信息；
 - 支持不须停机的硬盘备援(Hot Spare)
 - 支持不须停机的硬盘替换(Hot Swap)
 - 支持动态扩充硬盘容量





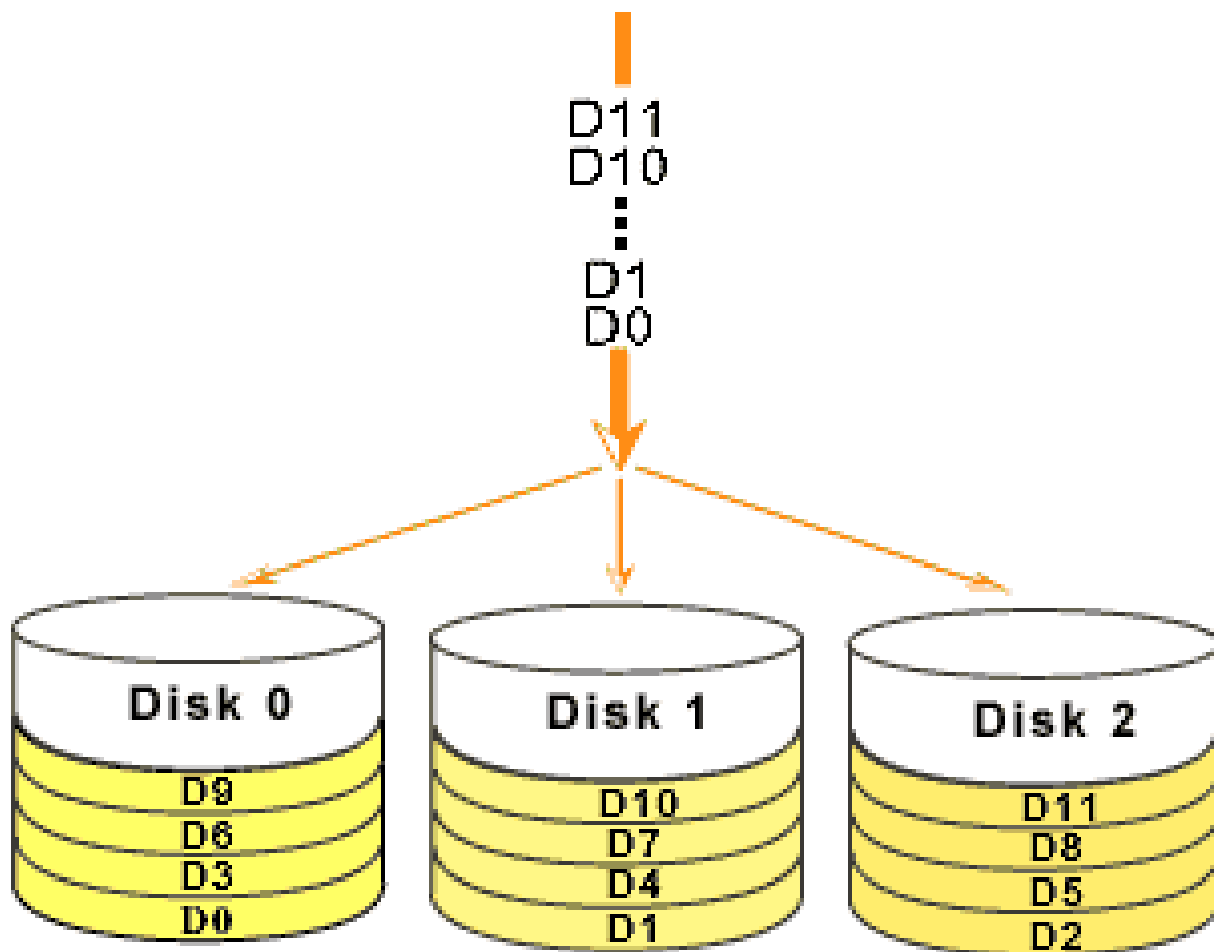
RAID级别

- RAID0：无差错控制的带区组
- RAID1：镜象结构
- RAID2：带海明码校验
- RAID3：带奇偶校验码的并行传送
- RAID4：带奇偶校验码的独立磁盘结构
- RAID5：分布式奇偶校验的独立磁盘结构
- RAID6：带有两种分布存储的奇偶校验码的独立磁盘结构
 - 对RAID5的扩展，主要是用于要求数据绝对不能出错的场合。
- RAID7：优化的高速数据传送磁盘结构
 - 采用并行和Cache技术

RAID 0 (无差错控制的带区组)

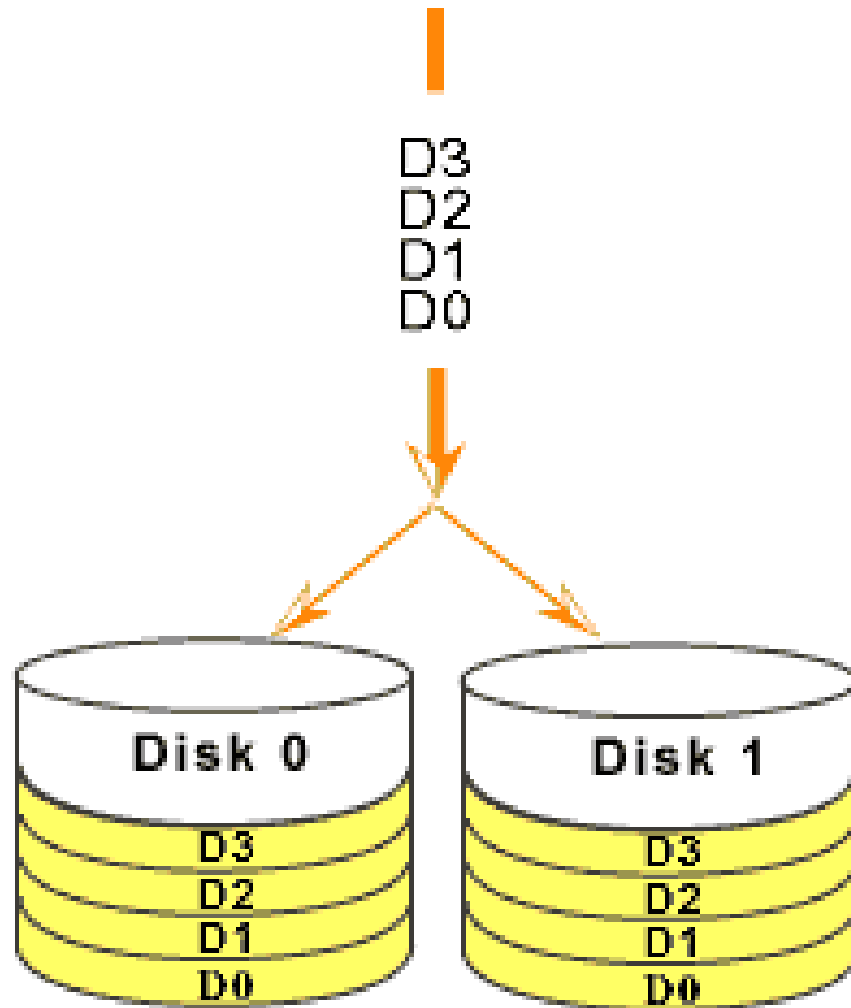


- 目的：利用多体并行提高存储性能



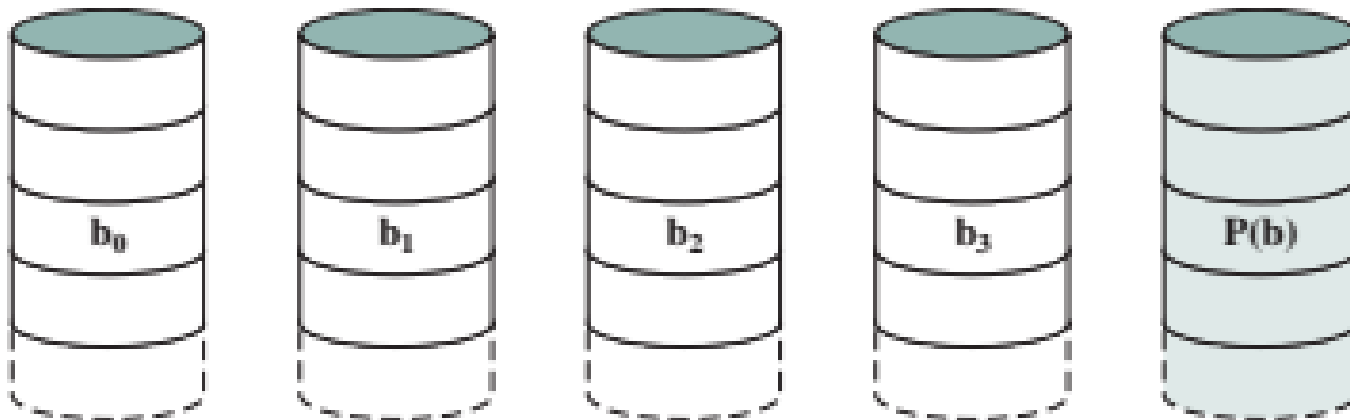
RAID 1 (别名: 镜像)

- 目标: 保证数据的可用性和可修复性



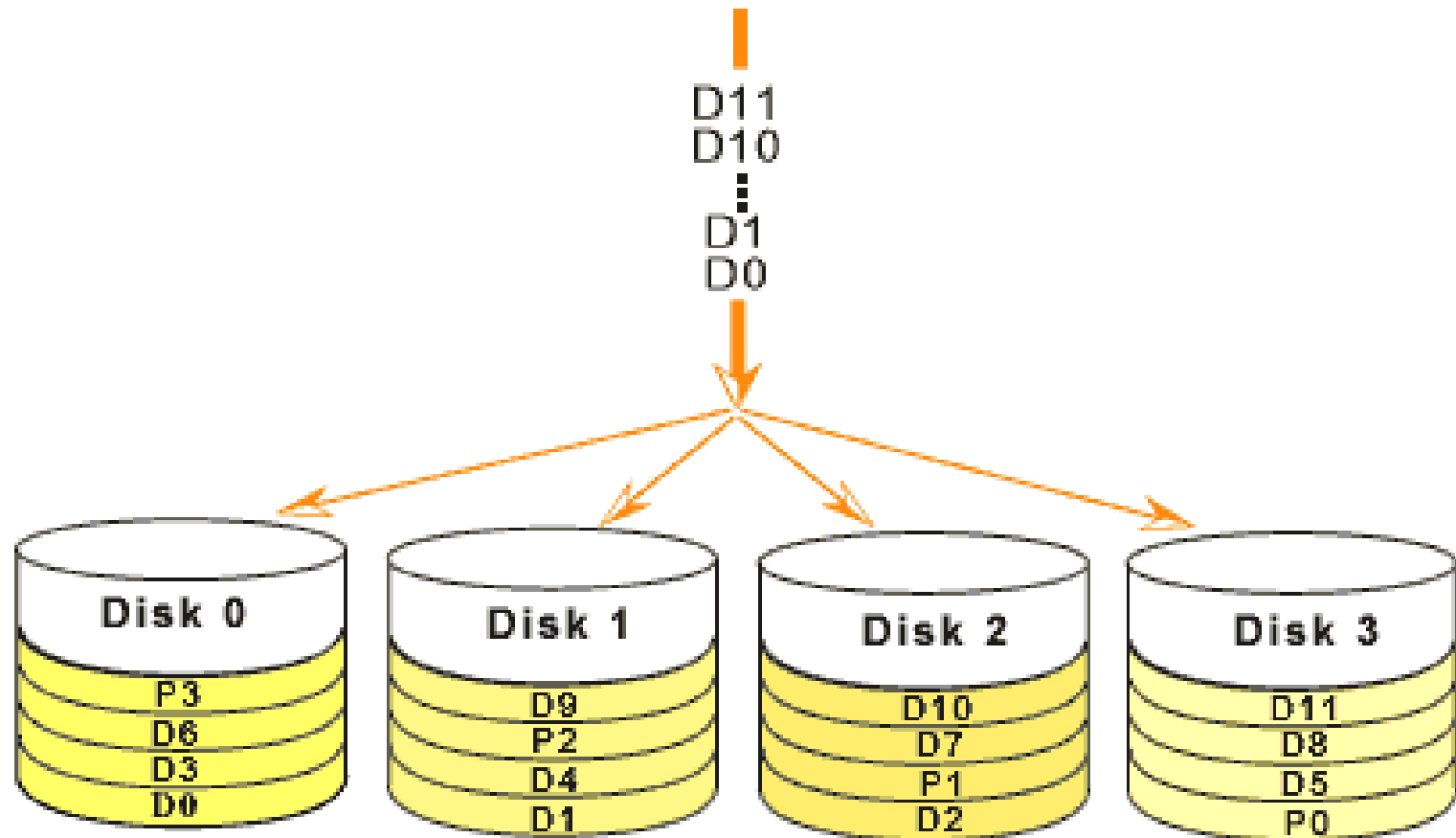
RAID3

- *RAID2 (少用) : 带海明码校验*
- RAID3: 带奇偶校验码的并行传送
 - 将数据条块化分布于不同的硬盘上
 - 条块单位为位或字节。
 - 必须要有三个以上的驱动器
 - 校验码在写入数据时产生, 保存在另一个磁盘上。
- *RAID4 (少用) : 按数据块访问数据*



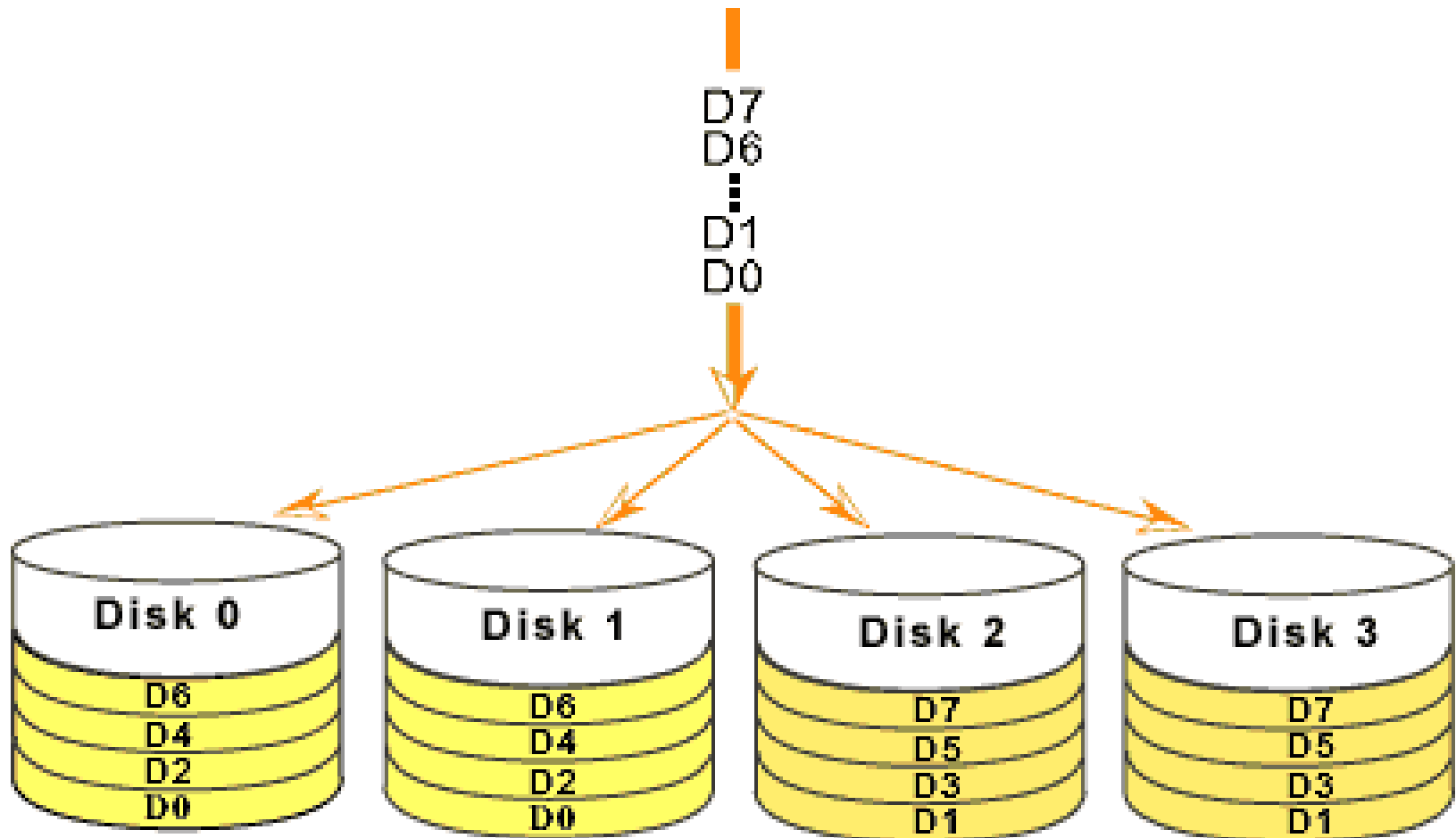
RAID 5

- 分布式奇偶校验的独立磁盘结构
 - 奇偶校验码存在于所有磁盘上



RAID 10 = RAID 0 + RAID 1

- 镜像阵列条带



RAID选择



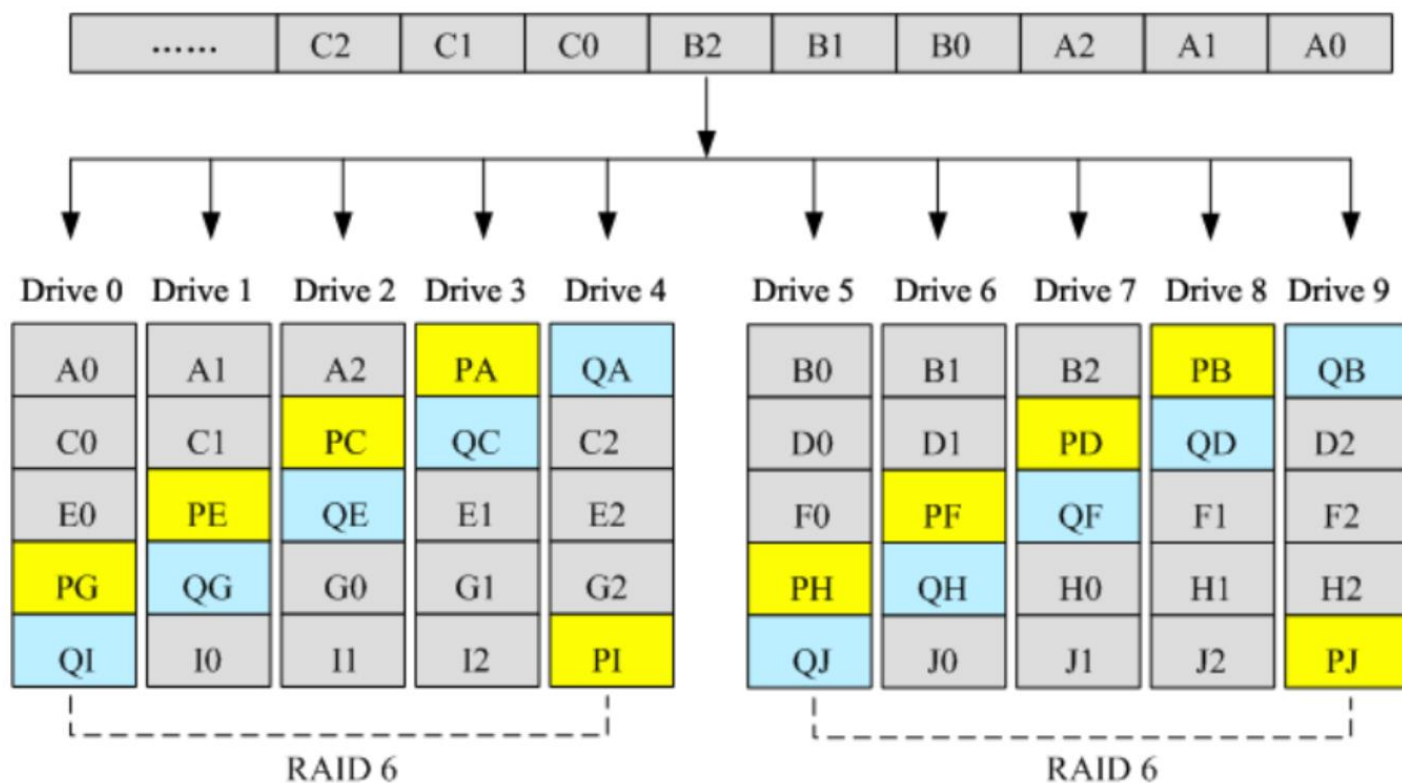
RAID 等级	RAID0	RAID1	RAID3	RAID5	RAID6	RAID10
别名	条带	镜像	专用奇偶校验条带	分布奇偶校验条带	双重奇偶校验条带	镜像加条带
容错性	无	有	有	有	有	有
冗余类型	无	有	有	有	有	有
热备份选择	无	有	有	有	有	有
读性能	高	低	高	高	高	高
随机写性能	高	低	低	一般	低	一般
连续写性能	高	低	低	低	低	一般
需要磁盘数	$n \geq 1$	$2n (n \geq 1)$	$n \geq 3$	$n \geq 3$	$n \geq 4$	$2n(n \geq 2) \geq 4$
可用容量	全部	50%	$(n-1)/n$	$(n-1)/n$	$(n-2)/n$	50%

- RAID50、RAID53、RAID60



RAID60 工作原理

- RAID60 由几个 RAID6 子组组成。



数据信息

校验信息1

校验信息2

小结



- 理解 “码距” 与 “编码体系的码距”
 - 奇偶的校验能力？码距？
 - 具有1位纠错能力的编码系统最小码距是多少？
 - SEC海明码码距是多少？
 - $(n+k, n)$ CRC码码距是多少？
- 如何确定所需校验码位数？
- 比较海明码与CRC码的容错能力
- 作业：唐：4.18、4.42



Thank you