



计算机组成原理

第四章 存储器

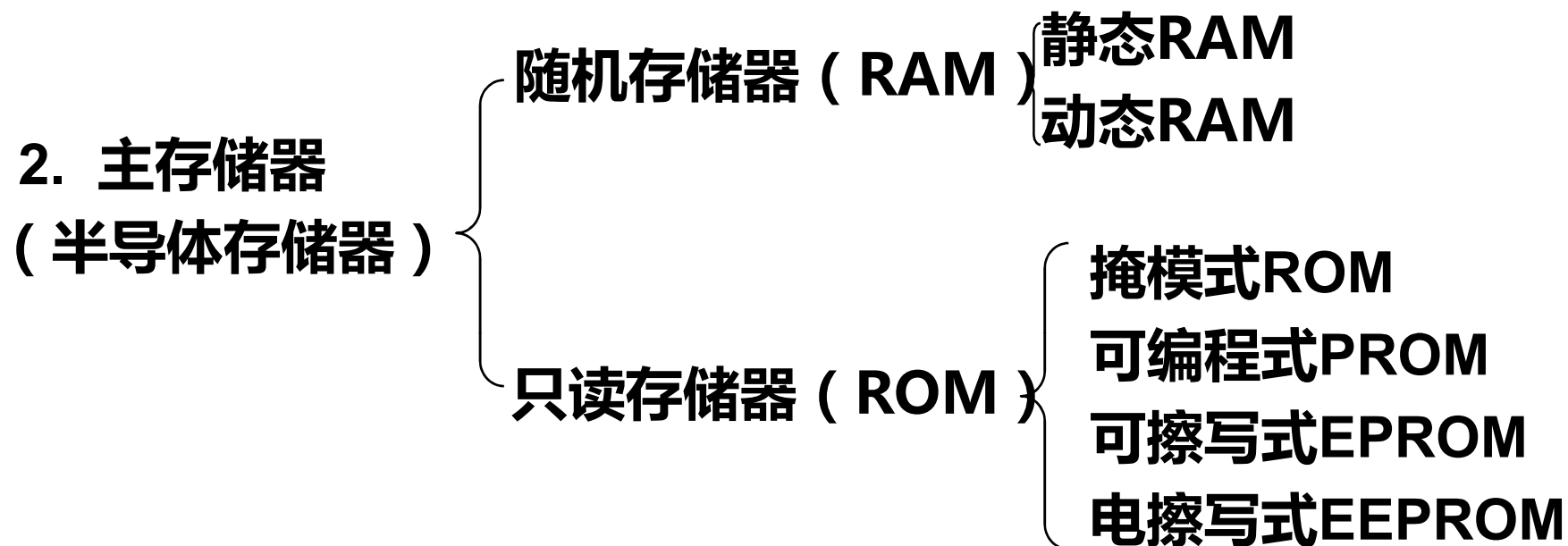
llxx@ustc.edu.cn

wjluo@ustc.edu.cn

内容



1. 主存储器的基本组成和技术指标



3. 存储器与CPU的连接

4. 存储器的校验—海明码

5. 提高访存速度的措施

Google2009：数据存储设备可靠性

- DRAM错误率超出人们预想
 - “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
 - 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。
- 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)**: 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
- **容错**计算系统: 出现一定限度的失效时, 依然提供所需要的服务.
 - “难于消除, 只能掩盖(使之不影响系统的正常使用)”
 - 服务降级
 - 容错能力: 检错 (发现, 定位), 纠错

Fault Tolerance: Redundancy

Main approach: mask failures using redundancy

- 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(回滾, rollback)
- Physical redundancy
 - Eg, 747s have **four** engines but can fly on three
 - RAID

奇偶编码校验 (Parity Check Code)



- 在被传送的 n 位代码($b_{n-1}b_{n-2}\dots b_1b_0$)上增加一位校验位 P (Parity) , 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 $Pb_{n-1}b_{n-2}\dots b_1b_0$)。
 - 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？具有什么容错能力？



	0000		0000	
	0001		0001	
	0010		0010	合法编码
	0011		0011	
	0100		0100	
	0101		0101	
4位表示	0110	4位表示	0110	
16个状态	0111	8个状态	0111	非法编码
不能检出错误!	1000	可能检出错误!	1000	
	1001		1001	
	1010		1010	
	1011		1011	码距: 海明 (Hamming) 距离
	1100		1100	两个等长码字之间对应位不同的个数
	1101		1101	
	1110		1110	相邻两个合法代码之间的不相同位数
	1111		1111	



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。
- 根据纠错律论： $L-1 = D+C$ 且 $D \geq C$
 - 即编码最小距离 L 越大，则其检测错误的位数 D 也越大，纠正错误位数 C 也越大，且纠错能力恒小于或等于检测能力。
 - 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。
- 应用
 - 数据通信：奇偶校验（串行），CRC（网络）
 - 硬盘：CRC
 - 内存：ECC（错误检查和纠正）校验



奇偶编码校验

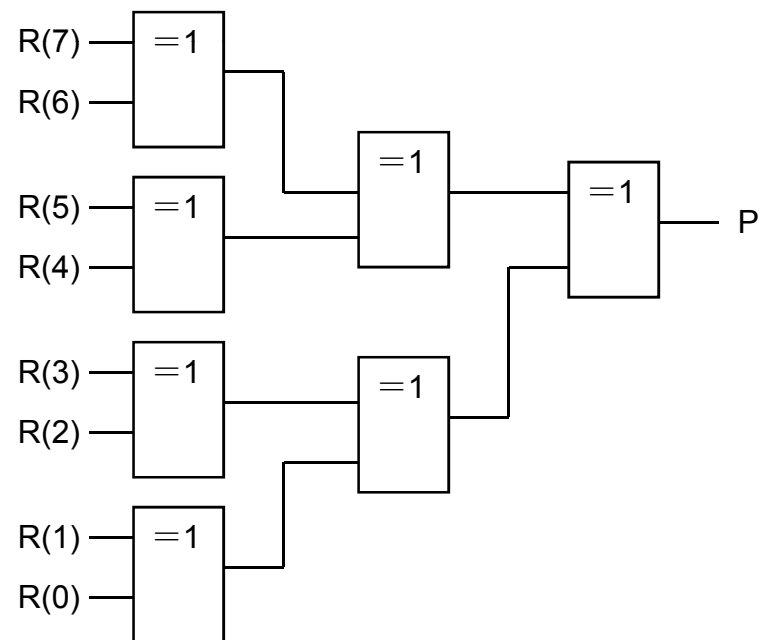
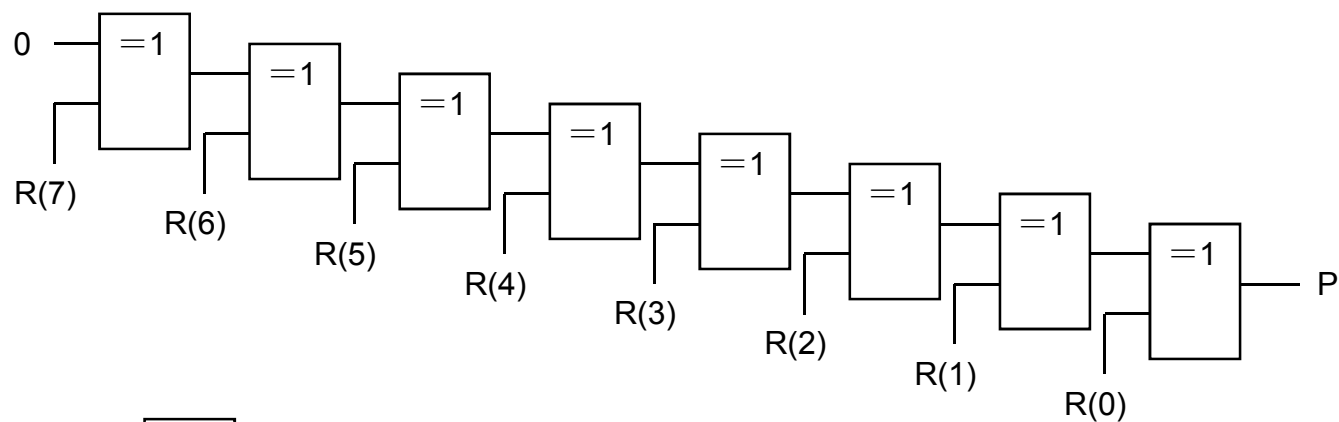
- 在被传送的 n 位代码($b_{n-1}b_{n-2}\dots b_1b_0$)上增加一位校验位 P ，将原数据和得到的奇（偶）校验位一起进行存取或传送(即传送 $Pb_{n-1}b_{n-2}\dots b_1b_0$)。

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

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

- 可以发现奇数个错
 - 只能检错，不能纠错（？）
 - 一位出错的概率高

奇偶校验的实现



验证电路？

奇偶校验的实现（续）



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;

交叉奇偶校验 (ECC)



数据块的横向和纵向都进行有奇偶校验位。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？

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？

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



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

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
 - 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、...)

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1010101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



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；
 - 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



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

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

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

余数=01

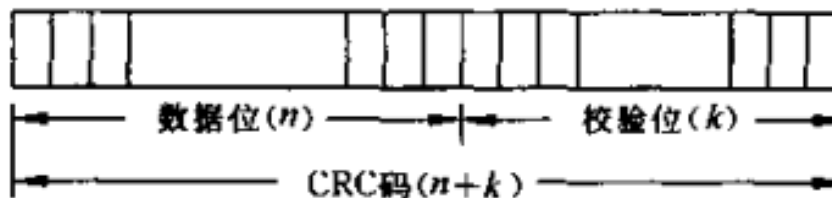


CRC校验步骤

• 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： $M(x) \cdot x^k$
- 将 $M(x) \cdot x^k$ 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

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

$$M(x) \cdot x^3 = x^6 + x^5 = 1100000$$

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

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

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

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

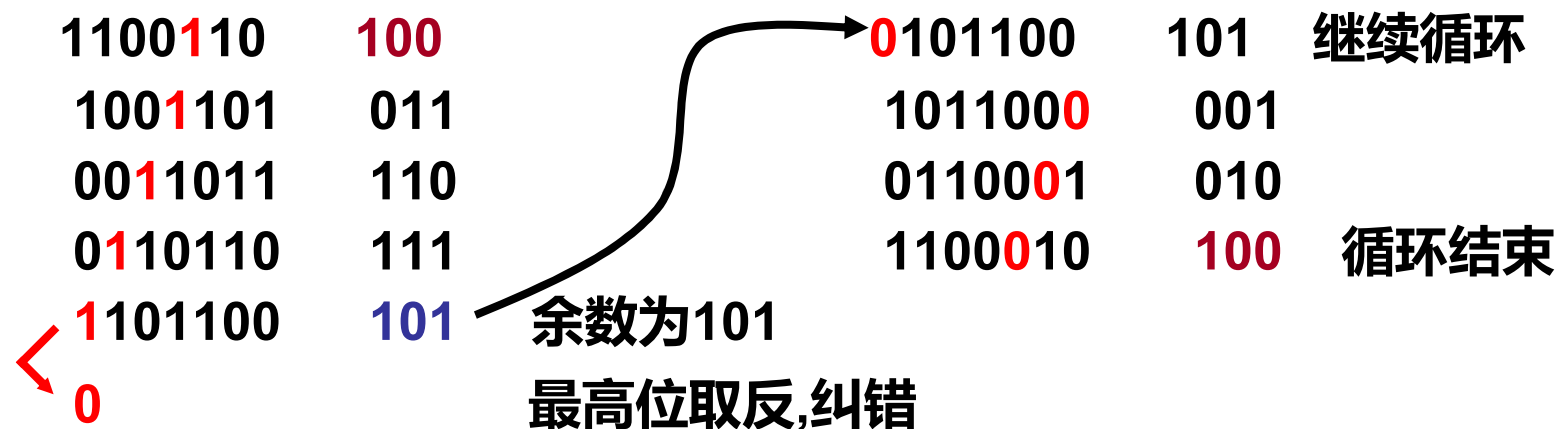
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	余数
正确	1	1	0	0	0	1	0	0 0 0
A_7 错	1	1	0	0	0	1	1	0 0 1
A_6 错	1	1	0	0	0	0	0	0 1 0
A_5 错	1	1	0	0	1	1	0	1 0 0
A_4 错	1	1	0	1	0	1	0	0 1 1
A_3 错	1	1	1	0	0	1	0	1 1 0
A_2 错	1	0	0	0	0	1	0	1 1 1
A_1 错	0	1	0	0	0	1	0	1 0 1

$$1011 \overline{) 0010}$$

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





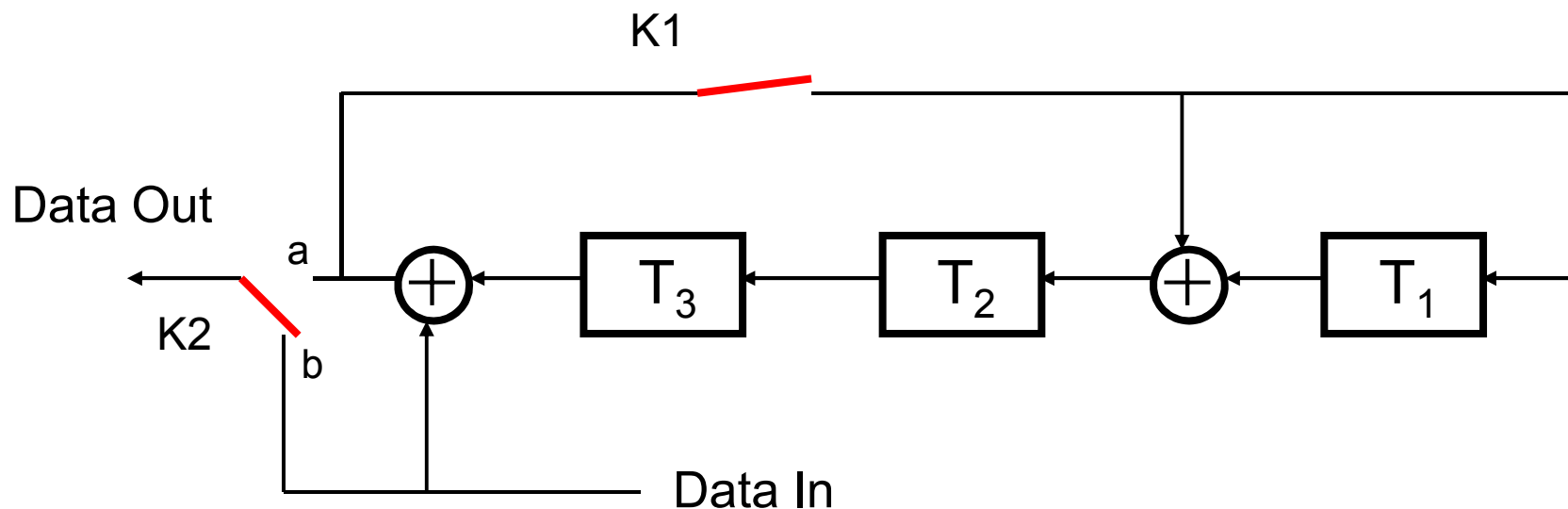
生成多项式 $G(x)$

- 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

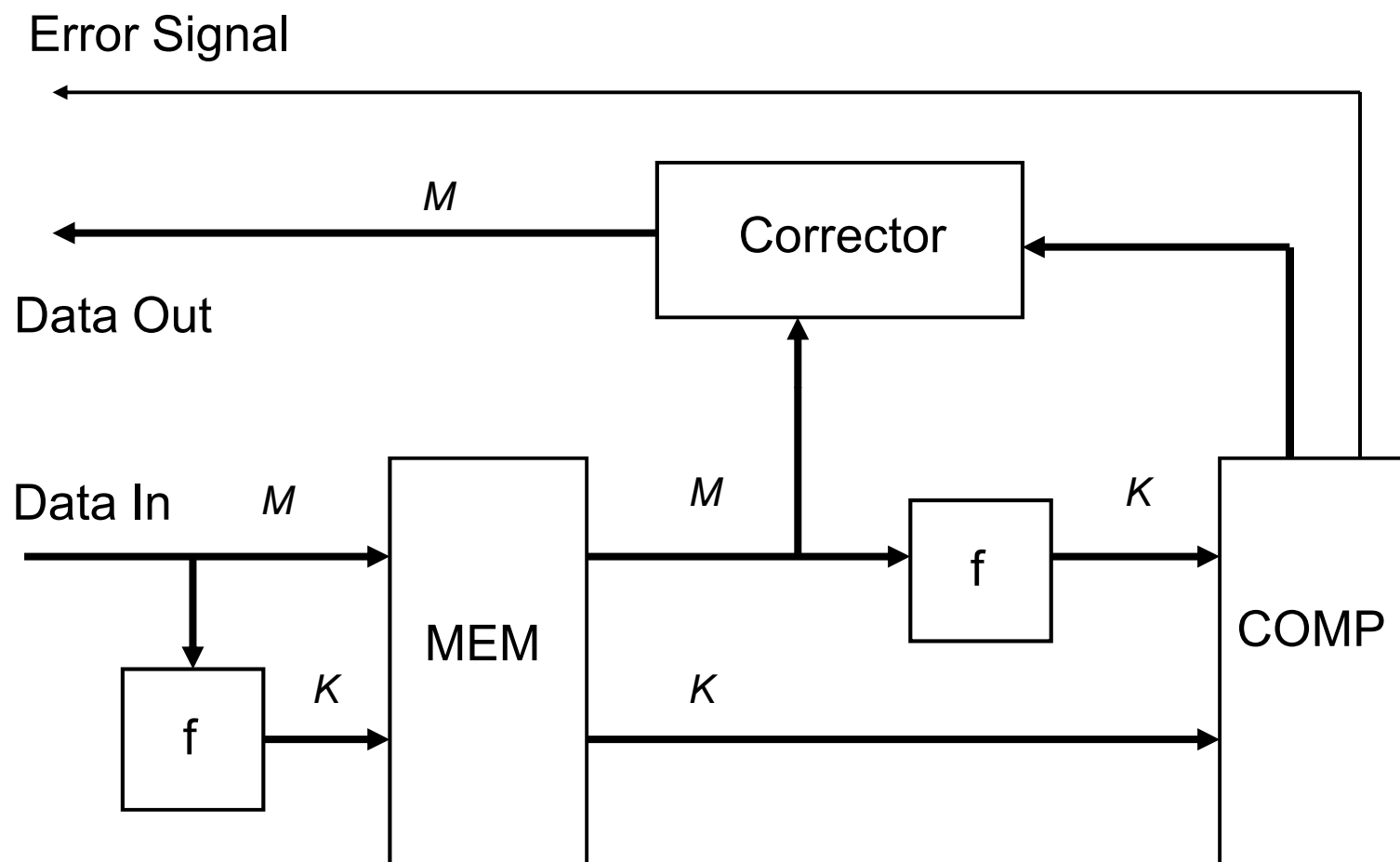
CRC电路实现



- **(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



存储校验系统



小结



- **理解“码距”与“编码体系的码距”**
 - 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？
- **比较海明码与CRC码的容错能力**
- **作业：4.18、20、4.42**



Thank you