

# 存储器的编码增益<sup>1</sup>

隋厚堂

(中国科学院空间科学与应用研究中心 北京 100080)

**摘要** 本文首先定义了存储器的编码自增益、互增益和它们的推广形式, 导出了计算公式。然后举例给出了计算结果和曲线, 并进行了分析讨论, 为系统设计者提供了一种定量的选码标准。

**关键词** 存储器, 编码增益, 自增益, 互增益

**中图分类号** TN911.22

## 1 引言

由于编码存储器 (M-ECC) 能有效地屏蔽存储器中出现的错误, 而得到广泛研究和实际应用<sup>[1-3]</sup>。对于星载 VLSI 存储器芯片, 受辐射影响, 单粒子翻转 (SEU) 严重<sup>[4,5]</sup>, 如何采用 ECC 技术来消除 SEU 的影响, 已成为人们近年来十分关注的课题<sup>[6-12]</sup>, 但上述研究都未能给出一种实用的选码标准。

文献 [13] 依据编码的两种基本用法: 查错和纠错, 分析比较了各种码本身的重要特性; 漏查和误译概率, 并给出了错误分布新概念。但对于 M-ECC 系统的设计者来说, 只知道码的本身特性, 漏查和误译概率还是不够的。还需要知道该码的系统性能, 以便正确选码, 使 M-ECC 的系统性能价格比最佳。本文提出了用编码增益作为码的选择标准, 给出了自增益和互增益的定义、公式和一些常用码的计算结果和曲线。

## 2 存储器的编码增益

在通信系统中, 用相同比特错误率 (BER) 条件下, 传送每比特信息得到的功耗好处作为编码增益。但在 M-ECC 系统中, 不存在这种对应关系, 本文定义的存储器编码增益 (记为  $G$ ) 是单位时间内, 相同功耗条件下, 每比特信息由于编码得到的平均无故障时间增加的倍数, 下面给出  $G$  的计算公式。

要计算增益  $G$ , 先要计算平均无故障时间 (MTTF)。设有一个  $(N, A, K)$  分组码 (计算机常用这种码), 如果我们能给出一个码字到  $t$  时刻未出错误的概率函数 (或可靠性函数)  $R(t)$ , 并考虑到码字的串联独立一致性, 则容量为  $M$  的编码存储器, 到  $t$  时刻未出错误的概率为  $R^M(t)$ 。由可靠性理论可知<sup>[14]</sup>, 其平均无故障时间为

$$\text{MTTF}_c = \int_0^{\infty} R^M(t) dt. \quad (1)$$

<sup>1</sup> 1994-10-10 收到, 1995-04-05 定稿  
国家自然科学基金资助项目

同样, 对于未编码的存储器, 一个  $K$ -bit 字节信息, 到  $t$  时刻未出错误的概率为  $R_u(t)$ , 则  $M$  个字节均未出错误的概率为  $R_u^M(t)$ , 它的 MTTF 为

$$\text{MTTF}_u = \int_0^{\infty} R_u^M(t) dt. \quad (2)$$

在翻转率和信息量相同的条件下, 由 (1) 和 (2) 式可得到  $(N, A\#, K)$  码的编码增益为

$$G_s = \eta_s \text{MTTF}_c / \text{MTTF}_u, \quad (3)$$

其中  $\eta_s = K/N$ , 为编码效率 (也称信息率)。为了区别, 称 (3) 式的  $G_s$  为自增益系数。类似可定义互增益系数, 设  $\text{MTTF}_1$  是用  $(N_1, A\#, K_1)$  码对  $I$  个信息单位编码得到的 MTTF,  $\text{MTTF}_2$  是用  $(N_2, A\#, K_2)$  码编码得到的, 前者对后者的互增益系数为

$$G_m = \eta_m \text{MTTF}_1 / \text{MTTF}_2, \quad (4)$$

$\eta_m = K_1 N_2 / K_2 N_1$ 。(3) 和 (4) 式是通用的, 对任何码, 任何错误类型都是适用的。

前面给出了自增益和互增益的一般定义, 下面导出具体的计算公式。目前给出的 MTTF 计算公式主要基于两种错误分布: 一种是二项式分布 [7], 另一种是 Poisson 分布 [11], 后者是一种保守的估算, 但比较简单。设出现的比特错误是独立随机过程,  $P$  为每比特单位时间翻转率 (与通信系统的 BER 不同, 它和时间有关),  $N$  为码字长度,  $M$  为总码个数,  $I$  为信息量。考虑 Poisson 分布, 设  $\mu$  为一个码字的平均翻转率,  $\mu = PN$ , 到  $t$  时刻未出错误的概率为

$$r_0(t) = \exp(-\mu t). \quad (5)$$

出现一个错误的概率为

$$r_1(t) = \mu t \exp(-\mu t). \quad (6)$$

因此, 对纠一位错误码, 一个码字的无故障概率函数为

$$R_1(t) = (1 + \mu t) \exp(-\mu t). \quad (7)$$

由  $M$  个串联独立一致的码字组成的存储器, 其无故障函数为

$$R_1^M(t) = (1 + \mu t)^M \exp(-\mu t M). \quad (8)$$

对于纠  $X$  位错误码, 则有

$$R_r^M(t) = \exp(-\mu t M) \left( \sum_{i=0}^X (\mu t)^i / i! \right)^M \quad (9)$$

如果考虑二项式分布, 同样可得

$$R_r^M(t) = \left( \sum_{i=0}^X \binom{N}{i} (1 - Pt)^{N-i} (Pt)^i \right)^M \quad (10)$$

对未编码系统，不妨设  $I$  以  $K$ -bit 为单位存储，有

$$r_u(t) = \exp(-\mu't), \quad (11)$$

$\mu' = KP$  (通常  $\mu' = 8P$ )。因此得

$$R_u^M(t) = \exp(-\mu'tM). \quad (12)$$

(8) 式代入 (1) 式积分得

$$T^1 = \exp(M)M! / (\mu M^{M+1}) \stackrel{M \gg 1}{\approx} \sqrt{2\pi M} / (M\mu). \quad (13)$$

(12) 式代入 (2) 式得

$$T^0 = \int_0^\infty \exp(-\mu'tM) dt = 1/(\mu'M). \quad (14)$$

把 (13) 和 (14) 式代入 (3) 式得自增益为

$$G_s = K\mu'\sqrt{2\pi M} / (N\mu) = (K^2/N^2)\sqrt{2\pi M}. \quad (15)$$

不失一般性，考虑一种软错误 (SEU) 刷新纠错工作方式。刷新纠错就是定期地查出存储器中出现的 SEU，并加以纠正。刷新的时间间隔称为刷新周期。对于短期任务，硬错误通常是可忽略的 (比 SEU 率小若干数量级)。如果硬错误 (包括支持电路的故障) 的影响不可忽略，则采用查错能力来实现备份更新，本文称这种功能为硬错误软化。设刷新周期为  $S$ ，在  $S$  时间段，未出一位错误的概率为  $\exp(-\mu S)$ ，出一位错误的概率为  $\mu S \exp(-\mu S)$ 。因此，不难得到纠一位错误码，到  $t$  时刻未出故障的概率函数

$$R_1^M(t, S) = \exp(-\mu Mt)(1 + \mu S)^{Mt/S}, \quad (16)$$

将上式代入 (1) 式可得平均无故障时间

$$T_1(S) = \int_0^\infty \exp(-\mu Mt)(1 + \mu S)^{Mt/S} dt = 1/[\mu M - (M/S) \ln(1 + \mu S)]. \quad (17)$$

当  $S$  取单位时间，得归一化表示

$$T_1 = 1/[M(\mu - \ln(1 + \mu))]. \quad (18)$$

将 (14) 和 (18) 式代入 (3) 式，可得纠一位错误码的 M-ECC 系统自增益公式为

$$G_s = K\mu' / [N(\mu - \ln(1 + \mu))]. \quad (19)$$

从 (19) 式可以看出， $G_s$  与  $M$  无关，这是因为  $N$  的有限长度引入的误差造成的。

对于刷新纠错系统可用一种更简单的非积分方法来计算 MTTF。由文献 [7] 可知， $X$  个随机错误纠错码的 M-ECC 系统的平均无故障时间为

$$T_x(S) = S/P(E > X). \quad (20)$$

$P(E > X)$  表示在  $S$  时间段内,  $M$ -ECC 中至少出现一个不可纠错误的概率, 显然  $P(E > X)$  的倒数是  $S$  的倍数, 因此  $T_u(S)$  的意义是显然的. 用二项式表示的  $P(E > X)$  为

$$P(E > X) = 1 - \left( \sum_{i=0}^X \binom{N}{i} (1-SP)^{(N-i)} (SP)^i \right)^M. \quad (21)$$

对未编码系统,  $X=0$ ,  $K$  代替  $N$ . 由 (21) 式得

$$P_u(E > 0) = 1 - (1-SP)^{KM}. \quad (22)$$

由 (20) 和 (22) 式可得未编码系统的 MTTF<sub>u</sub> 为

$$T_0(S) = S/P_u(E > 0) = S/[1 - (1-SP)^{KM}]. \quad (23)$$

把 (20) 和 (23) 式代入 (3) 式, 得自增益为

$$G_s = \eta_s P_u(E > 0)/P(E > X) = \eta_s [1 - (1-SP)^{KM}] / \left[ 1 - \left( \sum_{i=0}^X \binom{N}{i} (1-SP)^{(N-i)} (SP)^i \right)^M \right]. \quad (24)$$

$\eta_s = K/N$ , 表示一个  $(N, K)$  码的信息率. 如果实际纠错个数为  $L$ ,  $L=0, \dots, X$ , 则 (24) 式可改写成

$$G_s(L) = \eta_s P_u(E > 0)/P(E > L) = \eta_s [1 - (1-SP)^{KM}] / \left[ 1 - \left( \sum_{i=0}^L \binom{N}{i} (1-SP)^{(N-i)} (SP)^i \right)^M \right]. \quad (25)$$

为了区别, 将 (24) 式定义的自增益称为狭义自增益函数, (25) 式为广义自增益函数. (25) 式的意义在于: 可用于计算同一码字不同纠错个数时的编码增益. 这样, 在  $M$ -ECC 设计中, 不仅可选择合适的码字, 而且可进一步选择不同的纠错个数, 以便减小漏查和误译概率. 当  $X > 1$ , (25) 式可进一步推广, 设  $L'$  和  $L'' \leq X$ , 对应的 MTTF 分别为  $T'$  和  $T''$ ,  $G_s = \eta_s T'/T''$ . 该式可用于纠多个错误码的相对自比较. 对于 SEC 或 SEC-DED 码,  $X=1$ , 由 (24) 式, 可得到一种最常用的纠一位错误码的增益公式

$$G_s = \eta_s [1 - (1-SP)^{KM}] / [1 - (P_0 + P_1)^M], \quad (26)$$

式中  $P_0$  和  $P_1$  分别表示未出错和出一位错误时的概率,  $P_0 = (1-SP)^N$ ,  $P_1 = N(1-SP)^{(N-1)}SP$ . 令  $S=1$ , 得归一化表示

$$G_s = \eta_s [1 - (1-P)^{KM}] / [1 - ((1-P)^N + N(1-P)^{N-1}P)^M]. \quad (27)$$

设对  $I$  个字节编码, 用  $X_1$  纠错码  $(N_1, A\#, K_1)$ , 记为  $C_1$ , 需  $M_1 = \lceil 8I/K_1 \rceil$  个码字;  $X_2$  纠错码  $(N_2, K_2)$ , 记为  $C_2$ , 需  $M_2 = \lceil 8I/K_2 \rceil$  码字,  $\lceil \cdot \rceil$  表示向上取整. 设  $P_i^j$ ,  $0 \leq i \leq X_1$ ;

$P_j''$ ,  $0 \leq j \leq X_2$  分别表示  $C_1$  和  $C_2$  码出  $i$  和  $j$  个错误的概率, 则用  $C_1$  或  $C_2$  对  $I$  个字节编码的存储器, 在一个  $S$  时间内, 至少有一个码字出错的概率分别为

$$P(E > X_1) = 1 - (P_0' + P_1' + \cdots + P_{X_1}')^{M_1}, \quad (28)$$

$$P(E > X_2) = 1 - (P_0'' + P_1'' + \cdots + P_{X_2}'')^{M_2}. \quad (29)$$

将 (28), (29) 式代入 (4) 式, 以  $C_1$  为基准码, 得到的  $G_m$  公式为

$$G_m = \eta_m P_2(E > X_2) / P_1(E > X_1) = \eta_m \left[ 1 - \left( \sum_{j=0}^{X_2} P_j'' \right)^{M_2} \right] / \left[ 1 - \left( \sum_{i=0}^{X_1} P_i' \right)^{M_1} \right], \quad (30)$$

其中  $\eta_m = K_1 N_2 / K_2 N_1$ 。事实上, (30) 式等于  $C_1$  和  $C_2$  的自增益之比。设  $C_1$  和  $C_2$  码的自增益分别为  $G_{s1}$  和  $G_{s2}$ , 则有

$$G_m = G_{s1} / G_{s2}. \quad (31)$$

上式允许不同码之间进行比较, 为系统设计者提供一种实用的选码标准。类似 (25) 式, 可定义广义互增益函数为

$$G_m(L) = \eta_m P(E > L_2) / P(E > L_1) = \eta_m \left[ 1 - \left( \sum_{j=0}^{L_2} P_j'' \right)^{M_2} \right] / \left[ 1 - \left( \sum_{i=0}^{L_1} P_i' \right)^{M_1} \right], \quad (32)$$

其中  $L_1 = 0, \dots, X_1$ ;  $L_2 = 0, \dots, X_2$ 。

用 (17), (20) 和 (23) 式计算的平均无故障时间  $T$  的单位与  $S$  相同。因为  $T$  是  $S$  的函数, 所以  $G$  也是  $S$  的函数。为了方便, 令  $S = 1$ , 得  $G$  的归一化表示, 这时刷新时间是通过  $P$  对  $G$  的影响来实现的。如果增益用分贝表示, 以上各式计算的  $G$  要进行如下变换:

$$G_{dB} = 10 \log(G) \quad (\text{dB}). \quad (33)$$

对于非积分法, 也可用 Poisson 分布表示  $P(E > X)$ , 可以验证两种分布计算的 MTTF 和  $G$  基本相同。当  $P$  足够小,  $\mu S < 1$ , 展开  $\ln(1 + \mu S)$ , 取前两项, 且  $X = 1$ , 由 (17) 式得近似式为

$$T_1(S) = 2 / [SM(NP)^2]. \quad (34)$$

如果刷新时间  $S = 1$ , 则上式可改写成

$$T_1 = 2 / [M(NP)^2]. \quad (35)$$

由 (35), (14) 式可得

$$G_s = \eta_s 2K / (N^2 P) = 2K^2 / (N^3 P), \quad (36)$$

或由 (34) 式得

$$G_s(S) = \eta_s 2K / (SN^2 P) = 2K^2 / (SN^3 P). \quad (37)$$

由 (35) 式, 对等量信息, 用  $C_1$  码编码得到的近似 MTTF 为  $2/[M_1(N_1P)^2]$ , 用  $C_2$  得到  $2/[M_2(N_2P)^2]$ 。

代入 (30) 式可得  $G_m$  的近似表达式为

$$G_m = \eta_m M_2(N_2)^2 / [M_1(N_1)^2] = (K_1/K_2)^2 (N_2/N_1)^3. \quad (38)$$

当  $I \ll 1/P$ , 用上式估算  $G_m$  的精度是足够的。

### 3 计算结果和讨论

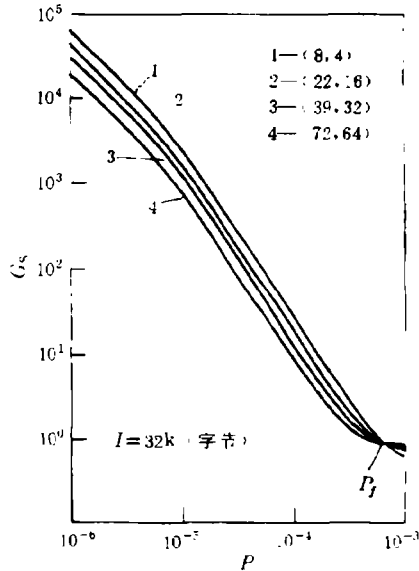


图 1 给定信息量  $I, P$  和  $G_s$  之间的函数曲线

图 1、2 和 3 是用 (27) 式计算得到的,  $I$  表示编码信息量, 以字节为单位。图 1 以  $P$  为自变量, 给出 4 种 SEC-DED 码的自增益曲线。它们有一个公共的交点  $P_f$ , 当  $P < P_f$ , 短码比长码的自增益大; 当  $P > P_f$ , 情况相反。

图 2(a) 和 2(b) 分别给出了  $P = 10^{-6}, 10^{-7}$ , 以  $I$  为自变量, 三种 SEC-DED 码的自增益曲线。不难发现, 把图 2(a) 的纵轴坐标乘 10, 然后把曲线沿横轴向右移, 就可近似得到图 2(b)。还可看出, 当  $I \ll 1/P$  时, 曲线变平,  $G_s$  变的和  $I$  无关。

图 4 给出了由 (30) 式计算的 (8, 4) 码相对其它几种码的互增益系数, 这里的曲线也存在一个  $P_f$  点, 对应  $G_m=1$ 。当  $P < P_f$ , 短码比长码优越;  $P > P_f$  相反。另外, 当  $P < 10^{-4}$  时,  $G_m$  渐趋最大饱和。

同为 SEC-DED 码, 对等量信息编码, (8, 4) 码相对其它 3 种码的  $G_m$  为 1.4, 2.0 和 3.2。

可见, 在选码时, 不能依据  $K/N$ , 要比较  $G_m$ 。

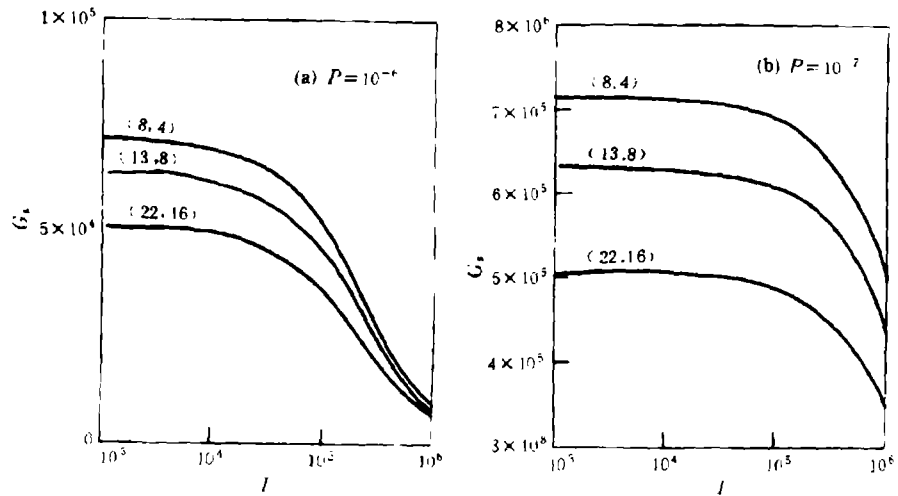


图 2 (a) 和 (b) 是以信息量  $I$  为自变量的  $G_s$  曲线

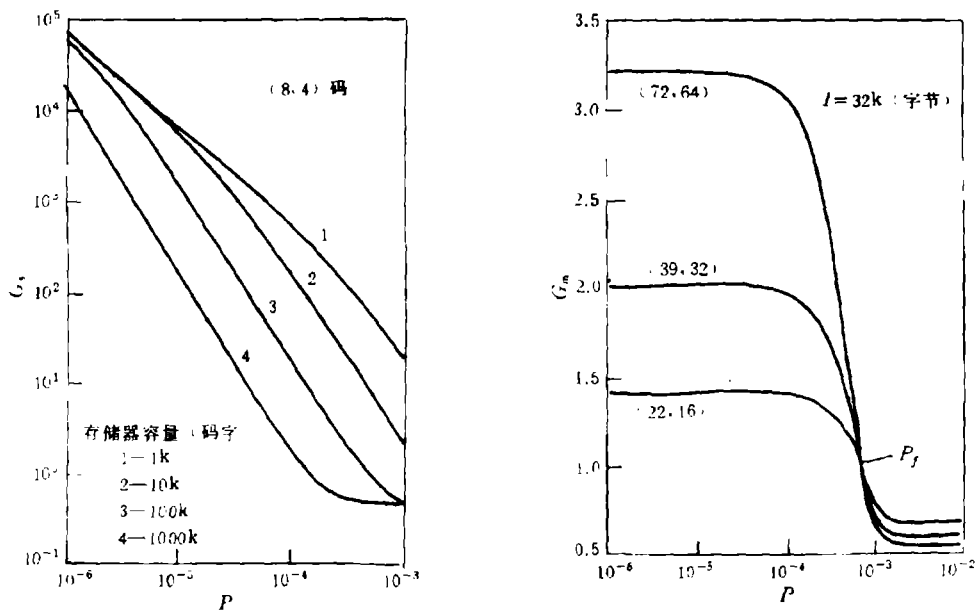


图 3 以存储器容量为参量的  $G_s$  曲线

图 4 (8,4) 码相对其它码的  $G_m$  曲线

### 4 结 论

本文给出了 M-ECC 系统的选码标准——编码增益。给出了自增益和互增益的定义，计算公式，以及它们的多种推广形式。利用导出的公式，针对刷新纠错工作方式，举例计算了一些常用分组码的增益系数。可以看出，当  $P$  较小（对于刷新系统可人为确定，通常小于  $10^{-6}$ ），短码比长码更有效，从编译码器的实现难度来看，前者更容易。因此，为了清除软错误应采用短码。

## 参 考 文 献

- [1] Fujiwara E, Pradhan D K. *Computers*, 1990, 23(7): 63-72.
- [2] Chen C L, Hsiao M Y. *IBM J Res. Develop.*, 1984, 28(2): 120-134.
- [3] Stapper C H, *et al.* *IEEE Trans. on Reliability*, 1993, R-42(4): 596-603.
- [4] Wilkinson D C, *Journal of Spacecraft and Rockets*, 1994, 31(2): 160-165.
- [5] Underwood C I, Ward J W, *et al.* *IEEE Trans. on Nuclear Science*, 1992, NS-39(6): 1817-1827.
- [6] Newman D B, Hutton S J. *Journal of the British Interplanetary Society*, 1992, 45(3): 121-126.
- [7] 隋厚堂. 用单个纠错码来提高半导体存储器的可靠性. 空间探测专业委员会第六次会议论文, 湖南张家界: 1992-11. 10-15.
- [8] Schwarty J W, Wolf J K. *IEEE Trans. on Computers*, 1990, C-39(11): 1403-1404.
- [9] Walker W K S, Sundberg C E, Black C J. *IEEE Trans. on Computers*, 1979, C-28(7): 493-500.
- [10] Karp S, Gilbert B K. *IEEE Trans. on AES*, 1993, AES-29(2): 310-316.
- [11] White J B, *IEEE Trans. on AES*, 1982, AES-18(1): 39-47.
- [12] O'Neill P M, Badhwar G D. *IEEE Trans. on NS*, 1994, 41(5): 1755-1764.
- [13] 隋厚堂. 系统 (16, 8) 码的错误分布和性能比较. 空间探测专业第七次会议论文集, 浙江舟山: 1994-10, 265-268.
- [14] Dhillon B S. *Reliability in computer system design*. Norwood, NJ: Ablex Publishing Corporation, 1987.

## THE CODING GAIN OF MEMORIES

Sui Houtang

(Center for Space Science and Applied Research, Academia Sinica, Beijing 100080)

**Abstract** The coding gains, that is, self-gain and mutual-gain with their variations for memories are defined and the formulas for computing the gains are derived, so that the criteria, which will enable the system designers for choosing best code, are given. Finally, the numerical results of some examples are analysed and discussed.

**Key words** Memory, Coding-Gain, Self-gain, Mutual-gain

隋厚堂: 男, 1943年生, 副研究员, 原从事装置研制, 后从事数据分组同步传输、数据压缩、信道编码和计算机可靠性等方面的研究。