

文章编号: 2096-1618(2018)04-0353-06

# 纠错模式可配置的 NAND Flash BCH 译码器设计

谢蓉芳<sup>1</sup>, 李子夫<sup>2</sup>, 叶松<sup>1</sup>

(1. 成都信息工程大学通信工程学院, 四川 成都 610225; 2. 中国科学院微电子研究所, 北京 100029)

**摘要:**针对 NAND Flash 的可靠性和使用寿命, 完成一种模式可配置的 BCH 码的译码电路结构设计。结构实现了 (8640, 8192, 32)、(8416, 8192, 16)、(8304, 8192, 8) 3 种模式的 BCH 码译码电路, 可根据存储器误码率配置译码模式, 通过合理配置译码电路内部资源, 减小功耗。译码器采用求余式的校正子求解法、SiBM 迭代算法、有限域固定因子乘法器的并行钱氏搜索算法。与单纠错模式的 BCH 码 (8640, 8192, 32) 相比, 在只增加极少硬件资源开销的情况下, 使低误码率时译码器的功耗大幅减少。优化后的纠错能力  $t=8$  的 BCH 译码器, 校正子结构、钱氏搜索结构分别节约了 49.1%、64.9% 的功耗, 纠错能力  $t=16$  的 BCH 译码器, 校正子结构、钱氏搜索结构分别节约了 34.0%、42.4% 的功耗。译码器基于 Xilinx 公司 Zynq 系列芯片, 在 Xilinx Vivado 上完成了电路仿真与验证。

**关键词:**微电子学与固体电子学; 集成电路; NAND Flash; 模式可配置; BCH 译码器; 低功耗

**中图分类号:** TN432

**文献标志码:** A

**doi:** 10.16836/j.cnki.jcuit.2018.04.001

## 0 引言

近年来, 大规模集成电路技术飞速发展, 以 NAND Flash 为代表的半导体固态存储器, 已成为各类电子产品中外部存储器的主要选择。其具有大容量、高存储密度、高访问速度、低成本、低功耗等优势, 在嵌入式设备和移动设备中广泛应用。当前市场 NAND Flash 的需求持续增加, 但随着存储容量的增加、工艺尺寸的减小, 电荷泄漏、编程干扰、保持时间 (retention)、编程擦除次数 (P/E cycle) 增加等导致存储器的可靠性和使用寿命面临严峻的挑战<sup>[1]</sup>。单比特纠错的汉明码已无法满足 ECC 纠错系统的应用要求, 取而代之的是纠错能力更强的 RS 码、BCH 码、LDPC 码等。RS 码是非二进制 BCH 码, 在纠正相同错误位数时, 所需的校验码长度大于 BCH 码, 硬件资源消耗量较大; LDPC 码是一类可逼近香农限的编码<sup>[2]</sup>, 但编译码算法复杂、硬件实现成本高等特点使其在主流存储器中还未广泛应用。

相比之下, BCH 码在中短码长下, 具有能纠正多位随机错误、速度快、电路简单等优点<sup>[3]</sup>。由于 NAND Flash 数据出错具有随机性, 且随着 P/E Cycle、Retention 的增加, 其误码率会以指数的倍率增加。如图 1 所示<sup>[4]</sup>, 在 P/E Cycle 小于 1.5 K 时误码率较低, 使用 BCH 码 (8304, 8192, 8) 的译码器即可纠正错误。若使用 BCH 码 (8640, 8192, 32) 纠错, 会浪费较多的硬件资

源和功耗。针对该问题, 设计了模式可配置、纠错能力强的 BCH 译码器纠错系统。采用了求余式、并行的校正子计算法, 简化的无求逆 SiBM 算法, 有限域固定因子乘法器 (constant finite field multiplier, CFFM) 的 8 位并行钱氏搜索算法, 减少了译码器的功耗, 以适应 NAND Flash 的应用。

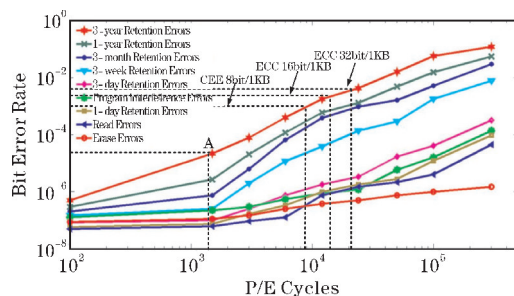


图1 各种误码率与编程擦除次数关系图

## 1 BCH 码简介

BCH 码是由 Bose、Chaudhuri、Hocquenghem 3 人分别独立发现和提出的一种可以纠正多个随机错误的线性循环码。BCH 码具有构造方便、编码简单以及译码易于实现等优点, 且 BCH 码有完备的代数理论支持<sup>[2]</sup>。

对于伽罗华域  $GF(2)$  及扩域  $GF(2^m)$ , 假设  $\alpha$  为  $GF(2^m)$  的本原元, 若  $GF(2)$  上最低次数多项式  $g(x)$  含有  $\alpha, \alpha^2, \alpha^3, \dots, \alpha^{d-1}$  等  $d-1$  个连续根, 则由  $g(x)$  生成的循环码称为 BCH 码。二进制 BCH 码  $(n, k, t)$  中,  $n$  为码长,  $k$  为信息位长度,  $t$  为纠错能力,  $d$  称为 BCH

码距离,二进制 BCH 码( $n, k, t$ )的参数满足下列关系:

$$\begin{cases} n = 2^m - 1 & (m \geq 3) \\ n - k \leq mt \\ t < 2^{m-1} \\ d_{\min} \geq 2t + 1 \end{cases}$$

其生成多项式为

$$g(x) = (x - a)(x - a^2) \cdots (x - a^d) = g_0 + g_1x + \cdots + g_rx^r$$

BCH 码编码过程比较简单,可用线性反馈移位寄存器(LFSR)实现。而译码过程相对比较复杂,且对整个 ECC 模块起着至关重要的作用,文中主要研究 BCH 译码器。

## 2 BCH 译码算法

BCH 译码算法由 3 部分组成:校正子  $S_i$  计算;错误位置多项式  $\sigma(x)$  计算;错误模式  $E(x)$  计算。设计的模式可配置的 BCH 译码器总体框架图如图 2 所示,除了上述的 3 部分,BCH 译码器还加入了 ms 信号纠错控制模块<sup>[4]</sup>,纠错控制模块的功能是调控各个模块的工作状态,接收到输入的 ms 信号后,设定译码器的工作模式。

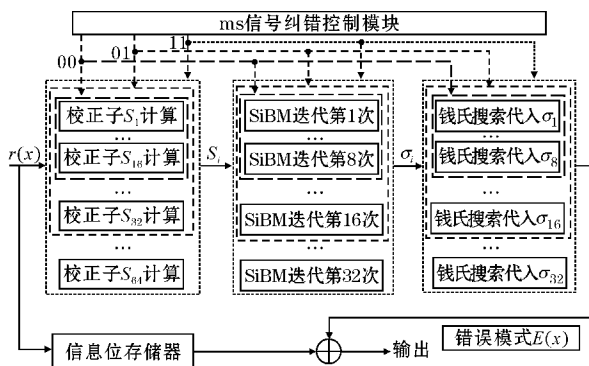


图2 BCH 译码器总体框架图

设计实现了 BCH 码(8640, 8192, 32)、(8416, 8192, 16)、(8304, 8192, 8)这 3 种纠错能力的译码电路,配置方式如表 1 所示。以图 1 中曲线 3-year Retention Errors 为例,P/E Cycles 小于 9.5 K 时,配置信号 ms 设置为“00”,调用  $t=8$  的工作模式,计算校正子  $S_1 \sim S_{16}$ ,SiBM 运算从第 1 次迭代到第 8 次,钱氏搜索代入  $\sigma_1 \sim \sigma_8$  参与运算。当 P/E Cycles 大于 9.5 K 小于 12.5 K 时,配置信号 ms 设置为“01”,调用  $t=16$  的工作模式,计算校正子  $S_1 \sim S_{32}$ ,SiBM 运算从第 1 次迭代到第 16 次,钱氏搜索代入  $\sigma_1 \sim \sigma_{16}$  参与运算;当 P/E Cycles 大于 12.5 K 小于 22.9 K 时,配置信号 ms 设置为“11”,调用  $t=32$  的工作模式,计算校正子  $S_1 \sim S_{64}$ ,

SiBM 运算从第 1 次迭代到第 32 次,钱氏搜索代入  $\sigma_1 \sim \sigma_{32}$  参与运算。3 种纠错模式都在同一有限域  $GF(2^{14})$  内,且信息位的位数相同,不同的是校验位的位数。将校验位寄存器的个数设定为最大值 448,通过配置信号来控制译码器对应资源参与工作,从而减小整个 ECC 模块的功耗。

表 1 纠错模式配置

配置参数	纠错能力	BCH 码参数/bit
00	8	(8304,8192,8)
01	16	(8416,8192,16)
11	32	(8640,8192,32)

### 2.1 校正子 $S_i$

BCH 码译码的第一步是计算校正子  $S_i$ ,假设接收的码字为  $r(x)$ ,校验矩阵为  $H$ ,则生成的  $2t$  个校正子定义如下:

$$S = (S_1, S_2, S_3, \dots, S_{2t}) = r \cdot H^T \quad (1)$$

$$r(x) = r_0 + r_1x + r_2x^2 + r_3x^3 + \cdots + r_{n-1}x^{n-1} \quad (2)$$

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \cdots & \alpha^{2(n-1)} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \cdots & \alpha^{3(n-1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & \alpha^{2t} & \alpha^{4t} & \alpha^{6t} & \cdots & \alpha^{2t(n-1)} \end{bmatrix} \quad (3)$$

对于  $1 \leq i \leq 2t$ ,校正子的第  $i$  个分量为

$$S_i = r(\alpha^i) = \sum_{j=0}^{n-1} r_j (\alpha^i)^j = r_0 + r_1 \alpha^i + r_2 \alpha^{2i} + r_3 \alpha^{3i} + \cdots + r_{n-1} \alpha^{(n-1)i} \quad (4)$$

直接按式(4)计算一个  $S_i$  需要  $n-1$  个有限域乘法器(FFM)、 $n-1$  个有限域加法器(FFA)。通过将接收多项式化简,即用接收多项式  $r(x)$  除以  $\alpha^i$  的最小多项式  $\Phi_i(x)$  得到  $r(x)$  的余式<sup>[5]</sup>。化简后的校正子计算公式为:

$$r(x) = q(x) \Phi_i(x) + b_i(x) \quad (5)$$

$$S_i = r(\alpha^i) = b_i(\alpha^i) \quad (6)$$

因此将校正子的计算电路分为两步:根据最小多项式求余式;将  $\alpha^i$  代入余式,求出  $S_i$ 。余式求解电路可用 LFSR 实现<sup>[6]</sup>,如图 3 所示, $\Phi_e$  为最小多项式的系数,其取值为 1 或 0, $e$  的最大取值为 13, $n$  位码字全部输入后,寄存器中的值就是余式的系数。此时在第一步求余式电路里面至多需要 13 个 FFM、13 个 FFA,第二步需要 13 个 FFM、13 个 FFA。计算一个校正子  $S_i$ ,至多需要  $13+13=26$  个 FFM、 $13+13=26$  个 FFA。资源消耗减为原来的  $26/(n-1)$ ,大大减少了校正子

计算模块的硬件消耗<sup>[7]</sup>。

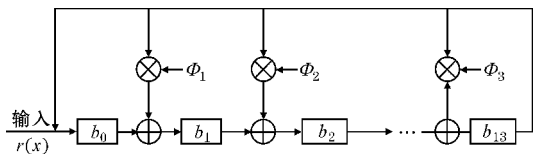


图3 余式求解电路

为了加快译码速度,采用8位并行执行的 LFSR 电路计算校正子<sup>[7]</sup>。每个时钟周期输入8位数据,通过最小多项式的系数  $\Phi_e$  进行反馈,在  $n/8$  个时钟周期内计算出校正子。8位并行算法公式推导如下:

$$S_i = r(\alpha^i) = \sum_{j=0}^{n-1} r_j (\alpha^i)^j \bmod \Phi_i(x) =$$

$$[r_0 + r_1 \alpha^i + \dots + r_{n-1} \alpha^{(n-1)i}] \bmod \Phi_i(x) =$$

$$\{ [r_0 + r_1 \alpha^i + \dots + r_7 \alpha^{7i}] \alpha^{(i) \times 0} + [r_8 + r_{8+1} \alpha^i + \dots +$$

$$r_{2 \times 8-1} \alpha^{7i}] \alpha^{(i) \times 8} + \dots + [r_{n-8} + r_{n-8+1} \alpha^i + \dots +$$

$$r_{n-1} \alpha^{7i}] \alpha^{(i) \times (n-8)} \} \bmod \Phi_i(x) =$$

$$\{ \sum_{j=0}^{n/8-1} [ \sum_{k=0}^7 r_{8 \times j + k} (\alpha^i)^k ] (\alpha^i)^{8j} \} \bmod \Phi_i(x)$$

由参考文献[2]中式子(6-31)可知,关于校正子的运算有:  $S_{2i} = S_i^2$ 。因此校正子计算时,只需要计算  $t$  个奇数次项校正子,再分别用  $t$  个 CFFM 计算对应的  $t$  个偶数项校正子<sup>[8]</sup>,从而将校正子计算模块的运算量减少了50%左右。简化后的8位并行校正子计算电路如图4所示。

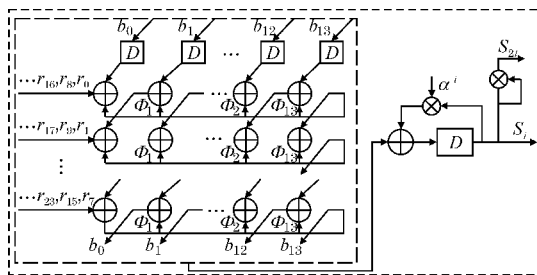


图4 简化的8位并行校正子计算电路

## 2.2 错误位置多项式 $\sigma(x)$

错误位置多项式  $\sigma(x)$  如式(7),  $\sigma(x)$  的求解电路是 BCH 译码电路中最复杂的模块,由于需要大量的 FFM,其占用资源也最多。经典的  $\sigma(x)$  求解算法为伯利坎普-梅西算法(Berlekamp Massey,简称BM算法),该算法实现中涉及有限域的求逆运算,耗费大量硬件资源,且可纠错数为  $t$  的 BM 算法需要进行  $2t$  次迭代运算。

$$\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2 + \dots + \sigma_t x^t \quad (7)$$

采用简化的无求逆 BM 算法<sup>[9]</sup>(SiBM),极大地缩短了关键路径,可以快速地实现  $\sigma(x)$  的求解。该算法

无需求逆,且利用二元 BCH 码的奇数次迭代时迭代差值为0,进一步节省了迭代周期,只需要  $t$  个周期即可完成迭代。ms 信号纠错控制模块实现模式的配置,ms 信号设置为3种模式,对应3种纠错能力,相应的 SiBM 运算分别迭代到第8次、第16次和第32次。SiBM 算法流程图如图5所示,其算法的迭代步骤如下:

步骤1 初始化:  $i = -1, d_{-1} = 1, d_0 = S_1, dq^{(-1)} = 1, \sigma^{(-1)}(x) = 1, \sigma^1(x) = 1, D^{(-1)}(x) = 1$ ;

步骤2 计算  $i = i + 2$ ,判断  $d_i$  是否等于0,若等于0则跳转到步骤3,否则跳转到步骤4;

步骤3 计算:

$$D^{(i)}(x) = x^2 D^{(i-2)}(x)$$

$$dq^{(i)} = dq^{(i-2)}$$

$$\sigma^{(i+2)}(x) = \sigma^{(i)}(x)$$

步骤4 计算:

$$D^{(i)}(x) = \sigma^{(i)}(x); dq^{(i)} = d_i$$

$$\sigma^{(i+2)}(x) = \begin{cases} dq^{(i-2)} \sigma^{(i)}(x) + d_i x D^{(i-2)}(x) & (i=1) \\ dq^{(i-2)} \sigma^{(i)}(x) + d_i x^2 D^{(i-2)}(x) & (i>1) \end{cases}$$

步骤5 计算下式,其中  $L_{i+2}$  表示  $\sigma^{i+2}(x)$  最高次幂

$$d_{i+2} = \sum_{j=0}^{L(i+2)} S_{i+1-j} \sigma_j^{(i+2)}$$

步骤6 检验  $i$  是否等于  $2t-2$ ,若相等,则停止迭代,否则跳转到步骤2。

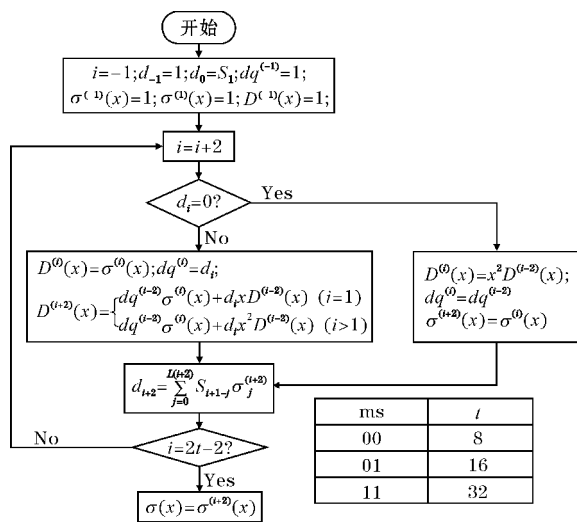


图5 SiBM 算法流程图

SiBM 算法<sup>[10]</sup>的电路图如图6所示,校正子计算值存储在寄存器  $S_i$  中,错误位置多项式系数存放在寄存器  $\sigma_i$  中,寄存器  $D_i$  起中间缓存多项式系数作用,第一个时钟周期到来时,对每个寄存器进行初始化;接下来每个时钟周期到来时,将两个校正子的值同时输入,且将校正子寄存器  $S_i$  的值向右移两位进行更新。寄

寄存器  $S_i$  与寄存器  $\sigma_i$  中的值相乘,之后进行累加得到结果记为  $d_i$ 。当  $d_i$  等于 0 时,  $d_q, \sigma_i, D_i$  值不变,将寄存器  $D_i$  中的值向相邻高位移动两位;当  $d_i$  不等于 0 时,将  $d_i$  存储在寄存器  $d_q$  中,将  $\sigma_i$  存储在寄存器  $D_i$  中,将寄存器  $d_q$  中的值与寄存器  $\sigma_i$  中的值相乘,然后把向右移动两位后的寄存器中  $D_i$  的值与  $d_i$  相乘,接下来将两个值相加,获得的结果用来更新  $\sigma_i$  的值。迭代  $t$  个时钟周期后,寄存器  $\sigma_i$  中的值即为错误位置多项式系数。

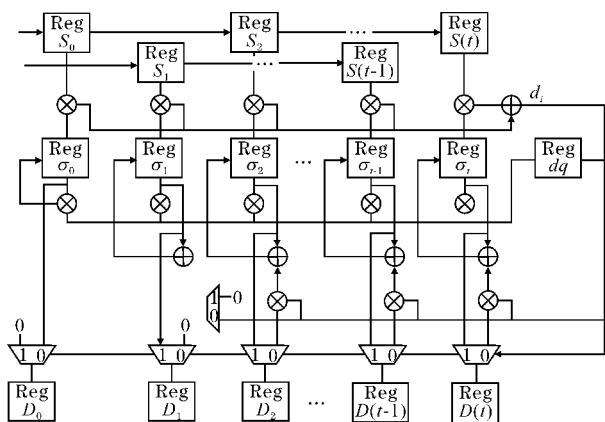


图6 SiBM 算法的电路图

### 2.3 错误模式 $E(x)$

错误模式计算采用钱氏搜索电路实现,其基本原理是利用错误位置多项式的根,来确定错误的位置。钱氏搜索算法的思路是从高位开始逐位校验输出,它的特点是不用等待所有的差错数验证完、纠错好后才解码输出,而是可以逐位解码,从高到低解码输出,大大减少译码的延时。串行钱氏搜索电路如图7所示。

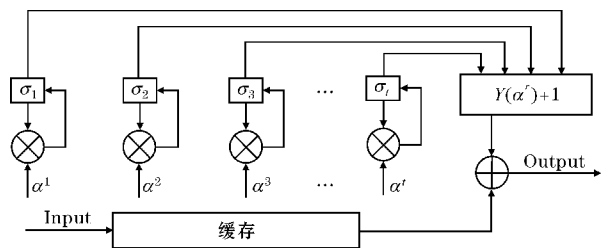


图7 串行钱氏搜索电路

串行钱氏搜索电路需要  $t$  个 FFM 和  $t$  个 FFA,其译码速度较慢,文献[10]提出一种8位并行 BCH 译码方案,相比于串行结构速度大幅度提升,但没有对 FFM 进行优化,复杂度较大。有限域  $GF(2^{14})$  中,每个 FFM 约需要 100 个查找表(LUT)。若其中一个因数改为常数  $\alpha^{100}$  后,只需要 12 个 LUT,资源消耗减少到原来的 12%,不同常数的乘法器,资源消耗减少不同<sup>[11-12]</sup>。

提出一种8位并行、模式可配置、优化 FFM 的钱氏搜索算法<sup>[13-14]</sup>。利用 ms 信号纠错控制模块实现模式的配置,ms 信号设置为“00”、“01”和“11”,分别调用  $t=8, t=16$  和  $t=32$  的工作模式,钱氏搜索分别代入  $\sigma_1 \sim \sigma_8, \sigma_1 \sim \sigma_{16}$  和  $\sigma_1 \sim \sigma_{32}$  参与运算。其电路图如图8所示,此电路结构最多需要  $9t=288$  个 FFM,乘法器优化后,消耗的资源、功耗会大量减少。通过 ms 信号控制纠错模式,  $m_i$  信号控制参与运算的  $\sigma_i$ ,实现模式可配置。二进制 BCH 码(8640,8192,32)是二进制本原 BCH 码(16383,15935,32)的缩短码,故不需要遍历所有元素,搜索位可以从 16383 位降低到 8640 位,在钱氏搜索时直接从第 7743 位开始,经过  $t$  个 CFFM 得到寄存器的初始化值分别为:  $\alpha^{7743} \sigma_1, \alpha^{7743 \times 2} \sigma_2, \alpha^{7743 \times 3} \sigma_3, \dots, \alpha^{7743 \times t} \sigma_t$ ,初始化完成后,以8位并行方式搜索错误位置<sup>[15]</sup>,当第  $i$  位出错时,钱氏搜索的结果输出为“0”,将该位上的值取反,即可纠正错误。此电路结构最长在  $8640/8=1080$  个时钟周期后,即可完成整个错误位置的搜索。

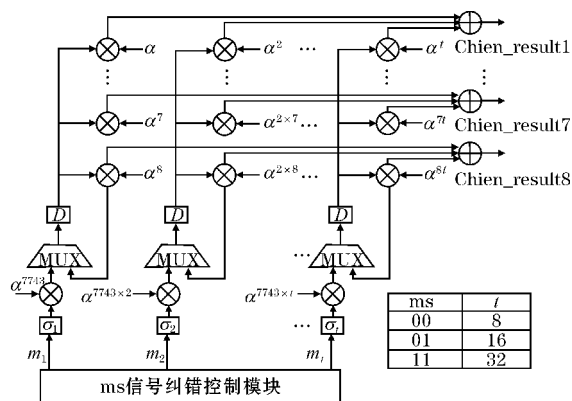


图8 并行钱氏搜索电路

## 3 设计仿真与验证

设计的纠错电路使用 Verilog 语言完成 RTL 级设计,用 Xilinx Vivado 软件完成仿真,仿真波形结果如图9~11所示,其中 clk 为系统时钟, rst 为系统复位信号, I\_data 为输入数据, out\_location 为错误位置。图9~11分别为纠错能力  $t=32, t=16, t=8$  的仿真波形图。当 out\_location 为“1”时,表明存入的数据和读出的数据一致,无错误;当 out\_location 为“0”时,表明读出的数据产生了错误,从图9可知,存入数据的第1~32位发生了错误。同理,图10的第1、2、3、4、9、10、11、12、16、19、20、24、25、26、27、28位发生了错误。图11的第1、3、4、5、6、7、8、10位发生了错误。经过大量的随机错误模式的仿真表明该 BCH 译码器可以正



确纠正任意错误数量不超过 32、16、8 比特的数据。

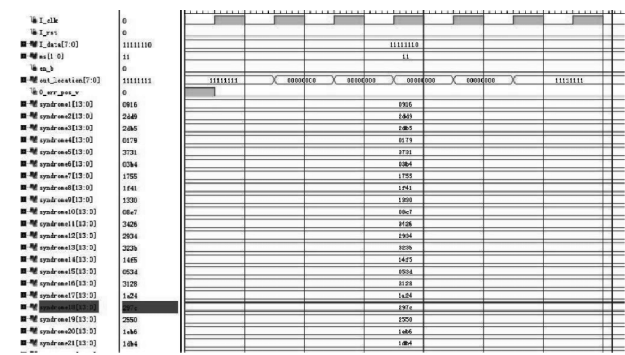


图9  $t=32$  仿真波形图

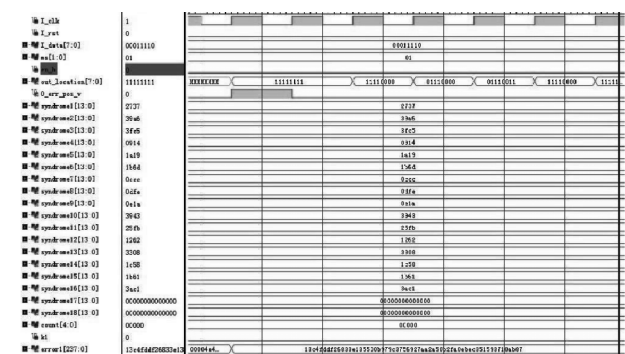


图10  $t=16$  仿真波形图

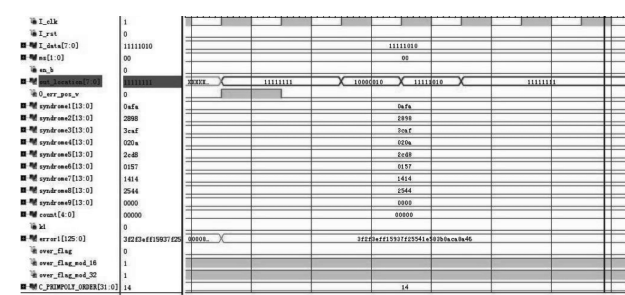


图11  $t=8$  仿真波形图

进一步对优化的校正子结构,钱氏搜索结构进行了功耗仿真,在 Vivado 下,从综合后的设计产生的 DCP 文件用于功耗的估计,在向量模式下,选择仿真生成的 SAIF 文件,Report Power 会自动生成 XPE 功耗估计所需文件。结果如表 2 所示,3 种模式下的出错位数均为 8 位,当采用 ms = “00” 模式,校正子功耗为 0.178 W,钱氏搜索结构功耗为 0.464 W,分别为 ms = “11” 模式下功耗的 50.9%、35.1%。此外,在 ms = “01” 模式下,百分比分别为 66.0%、57.6%。

表2 校正子、钱氏搜索结构功耗

配置参数	纠错/出错	校正子		钱氏搜索	
		功耗/W	比例/%	功耗/W	比例/%
00	8/8	0.178	50.9	0.464	35.1
01	16/8	0.231	66.0	0.762	57.6
11	32/8	0.35	100	1.322	100

## 4 结论

针对 NAND Flash 的可靠性和使用寿命,设计了模式可配置的 BCH 译码电路结构。在传统设计的基础上,对 BCH 译码器各个模块进行了有效优化:(1)硬件开销略大于单纠错模式  $t=32$  时硬件消耗,在低误码率情况下,只调用部分电路,随着误码率增大,复用误码率小的电路,实现硬件复用;(2)校正子计算模块求余式,8 位并行的设计,将求解一个校正子的 FFM 和 FFA 的资源消耗减少为原来的  $26/(n-1)$ ,整个校正子求解模块的资源消耗减少了 50% 左右。提高了计算速度,节约了电路功耗;(3)错误模式的计算,采用 8 位并行的钱氏搜索电路实现,且将 FFM 改为 CFFM 实现,极大地减少硬件消耗。

优化后的纠错能力  $t=8$  的 BCH 译码器,校正子结构、钱氏搜索结构分别节约了 49.1%、64.9% 的功耗,纠错能力  $t=16$  的 BCH 译码器,校正子结构、钱氏搜索结构分别节约了 34.0%、42.4% 的功耗。优化后的模式可配置的译码器结构显著提高了纠错性能。

## 参考文献:

- [1] Micheloni R, Crippa L, Marelli A. Inside NAND Flash Memories [M]. Springer Netherlands, 2010: 1–422.
- [2] Lin S, Costello D J. Error Control Coding, Second Edition [M]. Prentice-Hall, Inc, 2004: 140–165.
- [3] Lin Y M, Yang C H, Hsu C H, et al. A MPCN-Based Parallel Architecture in BCH Decoders for NAND Flash Memory Devices [J]. IEEE Transactions on Circuits & Systems II Express Briefs, 2011, 58(10): 682–686.
- [4] Cai Y, Haratsch E F, Mutlu O, et al. Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis [C]. Design, Automation & Test in Europe Conference & Exhibition. IEEE, 2012: 521–526.
- [5] M Wang, N Deng, H Wu et al. Theory study and implementation of configurable ECC on RRAM memory [J]. 15th Non-Volatile Memory Technology Symposium (NVMTS), Beijing, 2015: 1–3.
- [6] C Fougstedt, K Szczerba P Larsson-Edefors. Low-Power Low-Latency BCH Decoders for Energy-Efficient Optical Interconnects [J]. Journal of Light-

- wave Technology, 2017, 35 (23): 5201–5207.
- [7] N Lin, S Cai, X Ma. Block Markov superposition transmission of BCH codes with iterative hard-decision decoding [C]. IEEE International Symposium on Information Theory (ISIT), Aachen, 2017: 1598–1602.
- [8] Y Wu. Generalized integrated interleaving BCH codes [C]. 2016 IEEE International Symposium on Information Theory (ISIT), Barcelona, 2016: 1098–1102.
- [9] P Chen, C Zhang, H Jiang, et al. High performance low complexity BCH error correction circuit for SSD controllers [C]. 2015 IEEE International Conference on Electron Devices and Solidv-State Circuits (EDSSC), Singapore, 2015: 217–220.
- [10] CHEN T H, Hsiao Y Y, HSING Y T, et al. An Adaptive-Rate Error Correction Scheme for NAND Flash Memory [C]. VLSI Test Symposium, 2009. VTS'09. IEEE, 2009: 53–58.
- [11] J Jung, I C Park, Y Lee. A 2.4pJ/bit, 6.37Gb/s SPC-enhanced BC-BCH decoder in 65nm CMOS for NAND flash storage systems [C]. 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), Jeju, 2018: 329–330.
- [12] D Kim, I Yoo, I C Park. Fast Low-Complexity Triple-Error-Correcting BCH Decoding Architecture [C]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2017.
- [13] H Yoo, Y Lee, I C Park. Low-Power Parallel Chien Search Architecture Using a Two-Step Approach [J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2016, 63 (3): 269–273.
- [14] 蔡二龙. 并行 BCH 编解码的快速实现方法 [D]. 西安: 西安电子科技大学, 2015.
- [15] B Park, J Park, Y Lee. Area-optimized Fully-flexible BCH Decoder for Multiple GF Dimensions [C]. IEEE Access, 2018.

## Design of Mode Configurable NAND Flash BCH Decoder

XIE Rong-fang<sup>1</sup>, LI Zi-fu<sup>2</sup>, YE Song<sup>1</sup>

(1. College of Communacation Engineering, Chengdu University of Information Technology, Chengdu 610225, China; 2. Institute of Microelectronics of the Chinses Academy of Sciences, Beijing 100049, China)

**Abstract:** Aiming at the reliability and lifetime of the NAND Flash memory, a mode configurable BCH decoding circuit structure is implemented. The structure achieves three modes of (8640, 8192, 32), (8416, 8192, 16), (8304, 8192, 8) BCH decoding circuit which can be configured according to the memory bit error rate (BER), the power consumption is reduced by rationally configuring the internal resources of the decoding circuits. The decoder solves syndromes by remainder, and adopts simplified inversionless Berlekamp Massey (SiBM) iterative algorithm and Constant Finite Field Multiplier (CFFM) in the parallel Chien search algorithm. Compared with the single mode of BCH code (8640, 8192, 32), the proposed design which significantly reduces the power consumption in the case of low BER decoder sacrificing only a few hardware resources consumption. For the optimized BCH decoder of error correcting capability  $t=8$ , the Syndrome structure, Chien search structures are of 49.1%, 64.9% power saving. And for the BCH decoder of  $t=16$ , the power saving are 34.0% and 42.4% respectively. The decoder is based on the Xilinx Zynq series chip and the circuit simulation and verification is done by Xilinx Vivado.

**Keywords:** microelectronics and solid state electronics; integrated circuit; NAND Flash; mode configurable; BCH decoder; low power