一种基于RCRF+BCH算法的NAND FLASH纠错方案的FPGA设计与实现

刘 洋[1]，李 杰[1]，李金强[2]，李炳臻[1]，赵计贺[1]
(1.中北大学仪器科学与动态测试教育部重点实验室，太原，030051；2.山东航天电子技术研究所，山东烟台，264000)

摘要 针对目前NAND FLASH随着使用时间的增长误码率随之增高的特性，提出了一种在使用更少校验位的情况下纠错能力更强的高速并行RCRF+BCH纠错方案，初步用RCRF的思想对部分初始擦除错误进行纠正，然后级联BCH码纠正剩余的位错，很好地保证数据的准确性，显著提高存储系统的可靠性。首先详细阐述了该高速并行算法的编译码原理和执行步骤，在BCH部分中使用消耗更少硬件资源的无求逆的iBM关键方程求解算法，然后推导出错误位置多项式不同路径的几种确定形式，方便应用组合逻辑对其进行描述，避免了复杂的迭代判断过程，进一步提高了译码速度，并采用模块化的处理方式和流水线的操作模式优化了BCH的编译码结构。最终在FPGA平台硬件实现并仿真验证了此方案的有效性。

关键词 纠错系统；Reset-Check-Reverse-Flag算法；Bose-Chaudhuri-Hocquenghem；NAND FLASH
DOI 10.3969/j.issn.1009-3516.2020.06.008
中图分类号 TN919 **文献标志码** A **文章编号** 1009-3516(2020)06-0046-06

FPGA Design and Implementation of a NAND FLASH Error Correction Scheme Based on RCRF + BCH Algorithm

LIU Yang[1], LI Jie[1], LI Jinqiang[2], LI Bingzhen[1], ZHAO Jihe[1]
(1. Key Laboratory of Instrumental Science and Dynamic Testing of Ministry of Education, North University of China, Taiyuan 030051, China; 2. Shandong Aerospace Electronic Technology Institute, Yantai 264000, Shandong, China)

Abstract Aimed at the characteristic that the error rate of NAND FLASH increases with the increase of the time of use, a high-speed parallel RCRF+BCH error correction scheme with stronger error correction capability while using fewer parity bits is proposed. The idea of RCRF corrects some of the initial erasure errors, and then cascades BCH codes to correct the remaining bit errors, which can greatly ensure the accuracy of data and significantly improve the reliability of the storage system. The article explains in detail the encoding and decoding principles and execution steps of the high-speed parallel algorithm. In the BCH part, the iBM key equation solving algorithm without inversion that consumes less hardware resources is used, and then the different paths of the error position polynomial are listed through derivation Several deterministic forms facilitate the application of combinatorial logic to describe them, thus avoiding the

收稿日期：2020-06-03
基金项目：国家自然科学基金(61973280)；中国博士后科学基金(2019M661069).
作者简介：刘 洋(1995—)，男，吉林四平人，硕士生，主要从事微系统集成与组合导航等研究。E-mail:lylyly357@163.com

引用格式：*刘洋，李杰，李金强，等. 一种基于RCRF+BCH算法的NAND FLASH纠错方案的FPGA设计与实现[J]. 空军工程大学学报(自然科学版)，2020，21(6)：46-52. LIU Yang，LI Jie，LI Jinqiang，et al. FPGA Design and Implementation of a NAND FLASH Error Correction Scheme Based on RCRF + BCH Algorithm[J]. Journal of Air Force Engineering University (Natural Science Edition)，2020，21(6)：46-52.*

complicated iterative judgment process and further improving the decoding speed. And adopt the modular processing method and the pipeline operation mode to optimize the BCH codec structure. Finally, it was implemented on FPGA platform hardware and simulated to verify the effectiveness of this scheme.

Key words　error correction system; Reset-Check-Reverse-Flag; Bose-Chaudhuri-Hocquenghem; NAND FLASH

NAND FLASH 在每次重新写入前都需要区段擦除，随着擦除次数的增多，误码率随之升高。因此为了保证数据的准确性，选择一种性能良好的纠错算法至关重要。针对具有高误码率的存储器，采用能纠正一位错误的汉明码已经不能满足需求[1]。BCH 和 LDPC 是 2 种被广泛使用的具有出色纠错能力的算法[2-3]。但 LDPC 具有译码复杂度高的缺点，而且每次译码需要多次读取[4]。BCH 算法结构相对简单，不仅可以完成写入数据时的纠错，还能够对读取干扰和数据保持过程中电荷泄露造成的随机位错进行纠正，且每次译码只需读取一次，但 LDPC 和 BCH 都有较长的校验位开销。对于任意的正整数 $m \geqslant 3$ 和 $t < 2^m - 1$，一个码长为 $2^m - 1$ 的二元 BCH 码，在整个码字跨度上纠正 t 位错误，那么需要有 mt 个校验位，即纠错能力每增加一位就需要多使用 m 个校验位。针对已经因为擦写次数过多导致误码率很高的情况，如果单独使用 BCH 纠错算法，其校验位显然会占用非常多的空间。Reset-Check-Reverse-Flag，RCRF）是一种只需在每段信息位前添加一个标志位就可以纠正一个或多个复位错误的算法[5]，但是并不能纠正随机错误。现有文献在提出此种算法后应用其纠正新兴存储 NRAM 的“复位”错误（复位后存储单元的信息为 0），并提议与 BCH 结合，在使用相同校验位数（标志位＋BCH 校验位）的情况下和单独应用 BCH 进行对比分析，得出组合方案具有更好纠错效果的结论，但文献中并未给出 BCH 部分的实现方法和总体方案具体的硬件验证过程。

本文在其基础上提出将 RCRF 算法与 BCH 算法结合，对应用更广泛的 NAND FLASH 中的位错进行纠正。与新兴存储不同的是，NAND FLASH 不存在复位错误，而是存在一种擦除错误（擦除后的存储单元信息为 1，如果出错则为 0），所以初步采用 RCRF 的思想对部分初始擦除错误进行纠正，然后级联 BCH 纠正剩余的位错，同时从实际硬件实现角度出发，重点提出了 BCH 的改进方法。对于设计的 2 048 位信息位中识别并纠正 3 位错误的 BCH 码，只需添加一位标志位就可以达到在整个码字区间纠正 3 位及以上位错的效果。图 1 展示了传统单独使用 BCH 方法与新方案 RCRF＋BCH 的码字分布示意图。

本文详细阐述了该高速并行算法的编译码原理和执行步骤，使用消耗更少硬件资源的无求逆的 iBM 关键方程求解算法，然后提出通过推导列出错误位置多项式不同路径的几种确定形式，进一步提高了译码速度。最终在 FPGA 平台硬件实现并仿真验证了此方案的有效性。

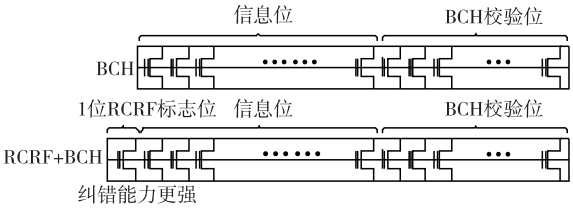

图 1　BCH 与 BCH＋RCRF 码字分布示意图

1　RCRF 编解码

1.1　RCRF 编码

NAND FLASH 在每次重新写入前需要区段擦除，擦除过的 BLOCK 所有的存储单元存储的值都为 1，如果此位发生错误则为 0。定义一帧数据由 1 位标志位和 2 048 位数据位构成。如图 2 所示，RCRF 编码过程可以用简单的 4 部分来描述[5-6]。首先读取擦除过后的 NAND FLASH 中的一帧内容；然后判断其中是否有位错发生，如果从右数的位置 a 处发生位错，则识别准备写入的 2 048 位信息位对应位置 a 处的值是否为 1；如果是 1，且因为擦除失败造成不能重新写入 1，可以将标志位和准备写入的信息位全部翻转并组合，使 a 处的值转变为 0，与帧数据位错的值一致，保证翻转后的 2 049 位数据写入 NAND FLASH 时是正确的，同时标志位从 1 变为 0 指示信息位已经被翻转，方便解码时确认；如果信息位 a 处的值是 0，则相当于不构成位错，可以直接送入 BCH 编码器。

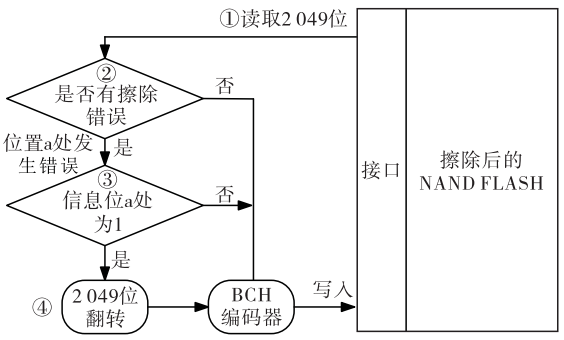

图 2　RCRF 编码过程

1.2 RCRF 解码

图 3 提供了 RCRF 解码的流程。在 BCH 解码器纠正了因其他噪声影响造成的位翻转后,此时标志位如果为 0,则翻转 2 048 位数据位即可获得正确的信息位,否则直接输出。值得注意的是即使标志位翻转,同样可以依靠 BCH 完成纠正。图 4 展示了一种 RCRF 可以纠正一位擦除错误的情况以及 RCRF+BCH 纠错方案的整体流程。

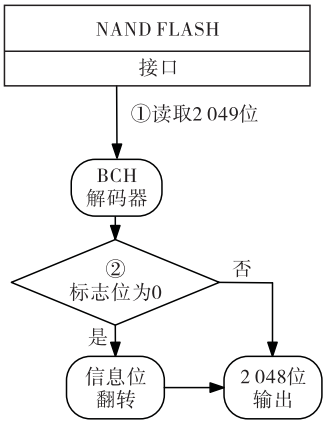

图 3 RCRF 解码过程

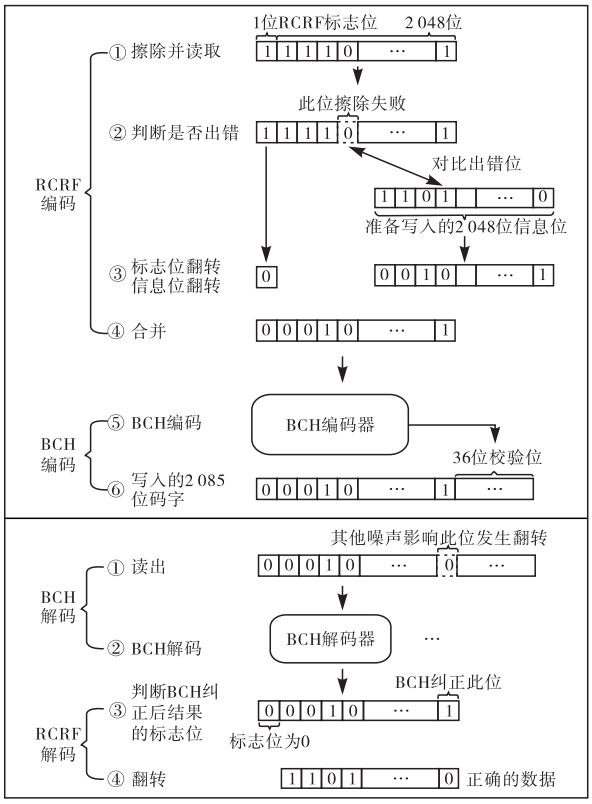

图 4 RCRF 编解码纠正一位错误(RCRF+BCH 纠错方案的整体流程)

理论上 RCRF 不仅具有纠正一位擦除错误的能力,还可以应对发生多位错误的情况[5]。以存在 2 位擦除错误为例进行分析,即该段数据包含任意 2 个位置的值都为 0,用 00 来表示。此时亟待写入的信息位中对应的值存在 4 种可能性,分别用 00,11,01 和 10 这 4 个图样来表示。信息位图样为 00 时等于没有位错发生,图样为 11 时与上述原理相同,方便执行 RCRF 编码操作并在解码时翻转,即可完成全部位错的纠正过程,标志位可以根据图 5 的电路产生。针对信息位图样为 01 和 10 的情况,RCRF 编解码对数据恢复没有效果,因为翻转后也不能完全写入正确的数据。但是即使判断存在擦除错误,标志位与信息位翻转,也可以应用 BCH 对剩余位错进行检测并纠正。所以实际在相应的情况下 RCRF 可以纠正多位错误。

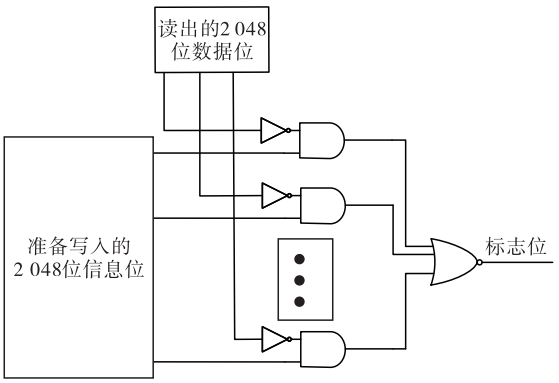

图 5 标志位生成电路

2 BCH 编解码

2.1 BCH 编码

在一个(n,k,t)BCH 码 $C(x)$中,n 是整个码字的长度,k 为待编码的信息位长度,t 是在整个码字跨度上纠正位错的个数,$n-k$ 和 mt 都代表码的校验位长度[7-8]。此方案设计级联 RCRF 后采用 BCH 算法更正 3 位错误,实现的 BCH 码为 $GF(2^{12})$上的缩短码(2 085,2 049,3)。

如图 6 所示,$C(x)$的串行编码可以通过带反馈回路的 $n-k$ 级移位寄存器的有限域除法电路来实现,反馈回路的连接关系由生成多项式 $g(x)$的系数 g_i 是否为 1 来确定,其中⊕代表异或关系,然后取寄存器中存储的余式作为校验位 $B(x)$。即:

$$B(x)=(x^{n-k}m(x))\bmod g(x)$$

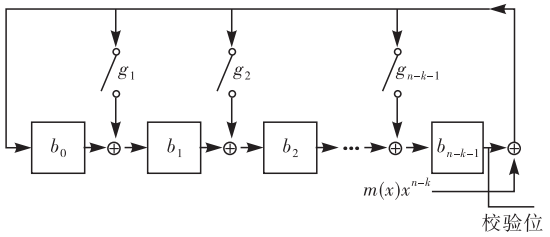

图 6 有限域除法电路

为提高编码效率,通过推导合并串行运算的方法,可以得到 p 位并行迭代编码的计算公式(1)。t_e 表示迭代的次数,$b(t_e)$表示第 t_e 次迭代后 $n-k$ 个

寄存器的值，$R_p(t_e)$表示第 t_e 次迭代输入的 p 位信息位[9]。在硬件电路中，式(1)描述的寄存器状态更新关系，可以通过使用 matlab 编写生成对应的 verilog 组合逻辑函数来实现。

$$b(t_e+1)=\boldsymbol{F}^p b(t_e)+\boldsymbol{FG}_p R_p(t_e) \tag{1}$$

式中矩阵 $\boldsymbol{FG}_p$ 为：

$$\boldsymbol{FG}_p=(\boldsymbol{F}^{p-1}\boldsymbol{G},\cdots,\boldsymbol{FG},\boldsymbol{G})$$

其中向量 $\boldsymbol{G}$ 为：

$$\boldsymbol{G}=(g_{n-k-1},\cdots,g_1,g_0)^{\mathrm{T}}$$

矩阵 $\boldsymbol{F}$ 为：

$$\boldsymbol{F}=\begin{pmatrix} g_{n-k-1} & 1 & 0 & \cdots & 0 \\ g_{n-k-2} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ g_1 & 0 & 0 & 0 & 1 \\ g_0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

2.2　BCH 解码

BCH 解码总体上包含计算伴随式，确定错误位置多项式和钱搜索这 3 个环节[10-11]，本研究分别对其进行模块化的逻辑电路功能描述，采用流水线式的操作结构，每个硬件模块执行完成即切换处理下一帧数据。设 $v(x)$发送多项式，$e(x)$为差错多项式，则接收多项式 $r(x)=v(x)+e(x)$。

2.2.1　伴随式求解

伴随式 $s(x)$是 $r(x)$除以 $g(x)$所得的余数。即：

$$s(x)=r(x)\bmod g(x)$$

若 $s(x)=0$，则认为接收多项式与发送多项式完全一致，即在整个信道传输过程中数据没有任何错误。若 $s(x)\neq 0$，则表示检测到了错误并需根据式 $s_z=s(\alpha^z)$计算参数 s_z 以用于确定错误位置多项式($1\leqslant z\leqslant 2t$，α 是本原元)[12]。伴随式的计算过程分为 2 部分。首先将接收数据的前 2 049 位送入编码电路，然后将得到的余式与接收数据的后 36 位进行异或操作，完成后即可得出 $s(x)$。

2.2.2　确定错误位置多项式

BM 算法是用于确定错误位置多项式 $\sigma(x)$的一种经典迭代译码算法。其迭代方程为：

$$\sigma^{(\mu+1)}(x)=\sigma^{(\mu)}(x)+d_\mu d_\rho^{-1}\sigma^{\rho}(x)x^{(\mu-\rho)} \tag{2}$$

式中：d_μ 代表第 μ 次迭代的修正项($0\leqslant\mu\leqslant 2t-1$)，有：

$$d_\mu=s_{\mu+1}+\sigma_1^{(\mu)}s_\mu+\sigma_2^{(\mu)}s_{\mu-1}+\cdots+\sigma_{l_\mu}^{(\mu)}s_{\mu+1-l_\mu} \tag{3}$$

ρ 的选择原则是满足 $\rho<\mu$ 且 $d_\rho\neq 0$ 的所有 ρ 中使得 $\rho-l_\rho$ 值最大的那个。l_ρ 代表 $\sigma^{(\rho)}(x)$的阶。

由于该迭代方程中存在有限域求逆运算，对应的多项式除法算法复杂度高且不易于硬件实现，故采用一种无求逆的 iBM 算法[13-14]。可以将有限域多项式求逆运算转换为多项式乘法运算[15]：

$$A(x)B(x)=(a_{m-1}x^{m-1}+\cdots+a_0)\cdot(b_{m-1}x^{m-1}+\cdots+b_0)\bmod Q(x)=$$

$$\begin{pmatrix} f_{0,0} & \cdots & f_{0,m-1} \\ \vdots & & \vdots \\ f_{m-1,0} & \cdots & f_{m-1,m-1} \end{pmatrix}\begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{pmatrix} \tag{4}$$

式中：$Q(x)=x^n+q_{n-1}x^{n-1}+\cdots+q_1x+1=x^{12}+x^6+x^4+x+1$，代表有限域 $GF(2^{12})$的本原多项式。矩阵 $\boldsymbol{f}_{ij}$ 为：

$$\boldsymbol{f}_{ij}=\begin{cases} a_i, & j=0;i=0,1,\cdots,m-1 \\ u(i-j)a_{i-j}+\sum\limits_{i=0}^{j-1}q_{j-1-t,1}a_{m-1-i}, & \\ & j=1,2,\cdots,m-1;i=0,1,\cdots,m-1 \end{cases} \tag{5}$$

$\boldsymbol{q}_{i,j}$ 的第 1 行为 $Q(x)$的多项式系数，即 $q_{0,j}=q_j$，其余元素可以通过式(6)使用递归方法得到。进而方便求出 $GF(2^{12})$中易于用组合逻辑描述的多项式乘法的确定形式。

$$\boldsymbol{q}_{i,j}=\begin{cases} q_{i-1,m-1}, & i=1,2,\cdots,m-2;j=0 \\ q_{i-1,j-1}+q_{i-1,m-1}q_{0,j}, & \\ & i=1,2,\cdots,m-2;j=1,2,\cdots,m-1 \end{cases} \tag{6}$$

同时通过推导与总结，可得奇数次迭代的修正项 d_μ 的值始终为 0，所以对于纠正 3 位错的 BCH 码，求解错误位置多项式的迭代过程可以从 6 步缩减为 3 步。凭借迭代方程描述的特定关系，进一步简化迭代过程，通过判断 d_μ 的值是否为 0，图 7 列出了最终错误位置多项式的 4 种确定情况。e_x 表示当 $d_2=0$，$d_4\neq 0$ 时接收数据中的位错数量超出了 BCH 码的纠错能力范围。应用这种直接的错误位置多项式求解方法，避免了繁琐的迭代过程，可显著减小算法复杂度，加快译码进程并降低电路延迟。

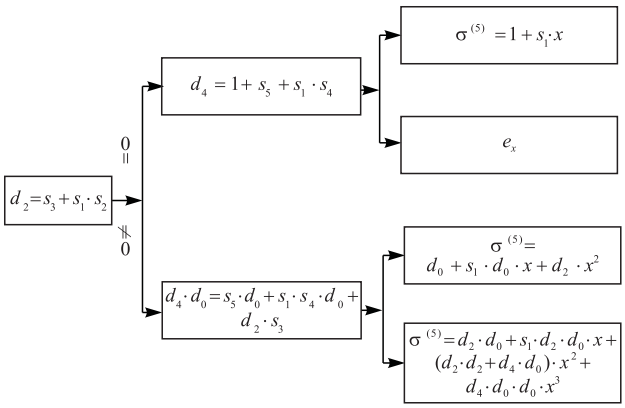

图 7　错误位置多项式的 4 种情况

2.2.3　钱搜索

钱搜索算法是将 $GF(2^{12})$域中元素分别代入 $\sigma(x)$，以验证 $\sigma(\alpha^{-i})$是否等于 0 的过程[16]，如果满足条件则说明接收数据的第 i 位为错误位置，对错误位置取反即可完成纠错。为加快译码速度，例化

了 32 个钱搜索模块并行处理该搜索过程。并行电路如图 8 所示,图中⊗符号代表有限域多项式乘法操作。在实际电路实现时,由于采用的是缩短码(2 085,2 049,3),所以不用遍历原码域中全部的元素,可以将根的范围锁定在 $\alpha^{2\,011}$ 到 $\alpha^{4\,095}$ 之间。同时通过判断 $\sigma(x)$ 的阶可以得出发生随机错误的个数,因此在搜索到对应个数的根后可提前终止搜索过程。通过使用提出的这 2 个方法可显著提高译码速度并减小功耗(具体速度的提升空间取决于码字中错误的位置分布)。

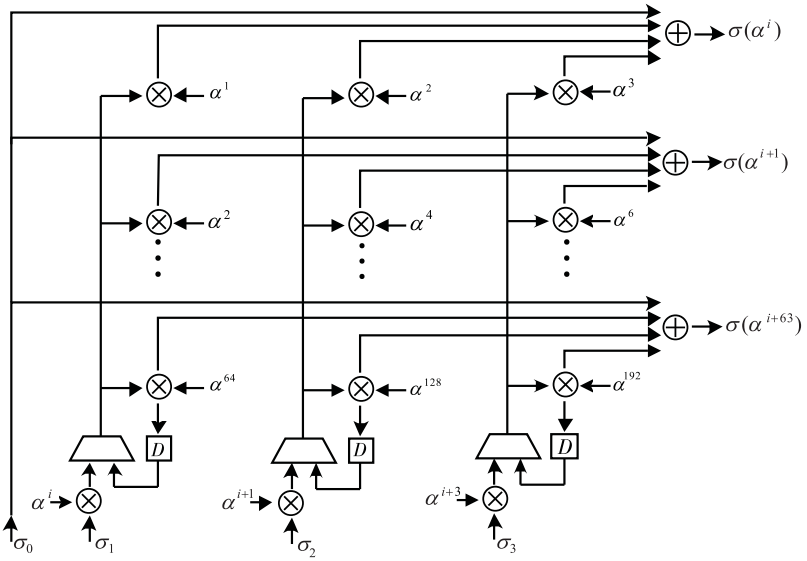

图 8 并行钱搜索电路

3 仿真与验证

已使用 Verilog 对电路进行描述,并搭建了 matlab 软件平台,方便对其进行对比验证,Vivado 综合后的仿真结果证明了该方案具有出色高效的纠错性能。

取擦除后从 NAND FLASH 读出的 2 049 位数据,假设从高位数的第 2 位、第 3 位发生擦除错误,即最高 9 位从 1FF 改变为 13F。且设此时亟待写入的 2 048 位信息位最高 8 位为 C0,即在对应错误位置的 2 位数据都为 1。此情况满足 RCRF 编码的判断条件,所以首先采用 RCRF 编码,翻转 1 位标志位和 2 048 位信息位(此时最高 9 位为 03F),并随之送入 128 位并行 BCH 编码器。图 9 展示了编码电路仿真结果的主要信号。RCRF_read 表示从 NAND FLASH 读出的数据;message 表示准备写入的 2 048 位数据;信号 flag_bit 显示标志位翻转后始终为 0;encode_128 展示了 128 并行编码器每次迭代后 36 位寄存器的状态更新过程;最终图中显示经过 0~16 个 cnt 加 1 的过程后(共计迭代 17 次),输出编码结果 parity,其值为 023FB3197,与 matlab 的编码输出结果一致。

设读出时 2 085 位数据中(翻转后的 1 位标志位+翻转后的 2 048 位信息位+36 位校验位),从最高位数的第 6、第 7 和第 8 位发生随机错误,即前 9 位从 03F 转变为 031。接下来对该 2 085 位数据 din 进行译码操作。为了方便说明,现将解码过程的部分主要信号分成图 10、图 11 进行阐述。

首先输出 BCH 伴随式求解结果 S(x)=8B2E8A389,因为此时 S(x)不为 0,代表检测到了随机错误,所以随之 S(x)_complish 信号置 1 表明伴随式计算完成且有效。将值送入 S1~S6 计算模块,图 10 中信号 S1~S6 给出了对应参数的求解结果,并拉高有效信号 s1_6_vld,同时使能错误位置多项式计算模块。使用图 7 所示的优化结构,仅在 4 个时钟周期后便可快捷得到错误位置多项式系数 sigma0~sigma3 的计算结果。three_err 变为高电平表明根据错误位置多项式的最终形式判断得出接收数据中有 3 位数据发生错误。

然后执行 32 位并行钱搜索操作。如图 11 所示,chien_vld_6~chien_vld_8 表明在 6、7、8 这 3 个钱搜索模块中搜寻到错误位置多项式的 3 个根,分别为 $\alpha^{2\,016}$、$\alpha^{2\,017}$ 和 $\alpha^{2\,018}$。根据此信息将对应位置数据取反,信号 BCH_de 显示 3 个随机位错已经被纠正,即前 9 位从 031 更正为 03F。此时 BCH 解码过程结束,执行 RCRF 译码操作,经过判断,标志位当前值为 0,已知如果标志位为 0 说明数据经历过翻转操作,因此将 2 048 位数据再次翻转得到最终的正确数据 decode_result。图中该信号的最高 8 位为 C0,表明从数据发送到接受产生的 5 个位错已经被全部纠正,译码完毕。

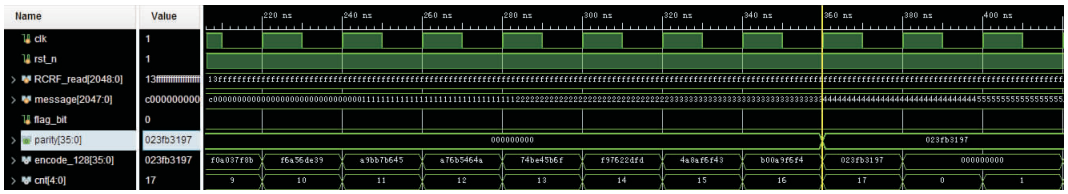

图 9　128 位并行 RCRF＋BCH 编码仿真结果

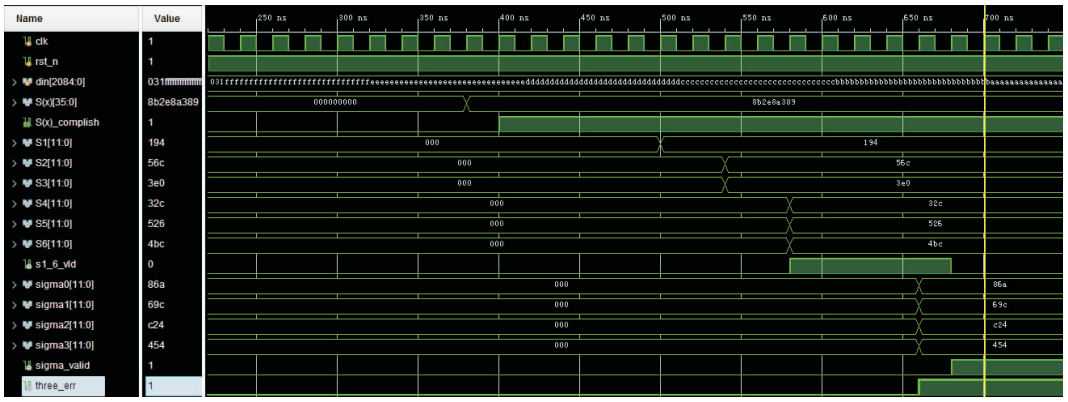

图 10　128 位并行 RCRF＋BCH 伴随式及错误位置多项式仿真结果

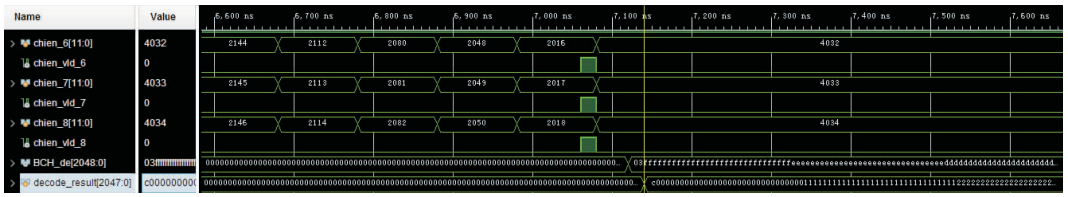

图 11　并行钱搜索仿真结果

4　结语

针对在具有高误码率情况下的 NAND FLASH 或其他存储器，本文提出并实现了一种实用的高速并行 RCRF＋BCH 纠错方案。对比单独使用 BCH 算法，此方案在仅添加一位标志位的情况下具备纠正更多位错的能力，减小了校验位开销，极大保证了数据的准确性，显著提高了存储系统的可靠性。同时由于 FPGA 具有灵活度高，可移植性强的特点，后续可以对这种方案配置不同的设计参数，以满足不同的纠错需求。

参考文献

[1] 罗金飞，赵帅兵，覃落雨，等. 存储系统中两级纠错编码方案设计与验证[J]. 航空学报，2019，40(12)：188-197.

[2] WEI L，JUNRYE R，WONYONG S. Low-Power High-Throughput BCH Error Correction VLSI Design for Multi-level Cell NAND Flash Memories[C]//IEEE Workshop on Signal Processing Systems Design and Implementation. Canada：IEEE，2006：303-308.

[3] CAI Y，YALCIN G，MUTLU O，et al. Flash Correct-and-Refresh：Retention-Aware Error Management for Increased Flash Memory Lifetime[C]//IEEE International Conference on Computer Design (ICCD). Canada：IEEE，2012：94-101.

[4] TANAKAMARU S，YANAGIHARA Y，TAKEUCHI K. Error-Prediction LDPC and Error-Recovery Schemes for Highly Reliable Solid-State Drives (SSDs)[J]. IEEE Journal of Solid-State Circuits，2013，48(11)：2920-2933.

[5] NING S，TOMOKO O I，SHUHEI T，et al. Reset-Check-Reverse-Flag Scheme on NRAM With 50% Bit Error Rate or 35% Parity Overhead and 16% Decoding Latency Reductions for Read-Intensive Storage Class Memory[J]. IEEE Journal of Solid-State Circuits，2016，51(8)：1938-1951.

[6] NING S Y. Advanced Bit Flip Concatenates BCH Code Demonstrates 0.93% Correctable BER and Faster Decoding on (36864，32768) Emerging Memories[J]. IEEE Transactions on Circuits and Systems I-Regular Papers，2018，65(12)：4404-4412.

[7] MOON T K. Error Correction Coding：Mathematical Methods and Algorithms[M]. United States：John Wiley and Sons，1995.

[8] BERLEKAMP E. Algebraic Coding Theory，Revised edition[M]. United States：World Scientific Publishing Co，1995.

[9] HU G H，SHA J，WANG J F. High-Speed Parallel

LFSR Architectures Based on Improved State-Space Transformations[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2017, 25(3): 1159-1163.

[10] 王杰. NAND Flash 主控中 BCH 编译码器的研究和 ASIC 实现[D]. 杭州:浙江大学, 2010.

[11] 毛旭富. BCH 编解码器在 NAND Flash 主控中的研究与优化实现[D]. 上海:上海交通大学, 2013.

[12] CHOI H, LIU W, SUNG W. VLSI Implementation of BCH Error Correction for Multilevel Cell NAND Flash Memory[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2010, 18(5): 843-847.

[13] BELLORADOJ, KAVČI Ć A. Low-Complexity Soft-Decoding Algorithms for Reed-Solomon Codes-part I: An Algebraic Soft-in Hard-Out Chase Decoder[J]. IEEE Transactions on Information Theory, 2010, 56(3): 945-959.

[14] 徐富新,刘应,刘雁群,等. 模式可配置的 NAND Flash 纠错系统设计与实现[J]. 中南大学学报(自然科学版). 2013, 44(5): 1918-1925.

[15] PAAR C. A New Architecture for a Parallel Finite Field Multiplier with Low Complexity Based on Composite Fields[J]. IEEE Transactions on Computers, 1996, 45(7): 856-861.

[16] ZHANG M, WU F, XIE C, et al. A Novel Optimization Algorithm for Chien Search of BCH Codes in NAND Flash Memory Devices[C]//IEEE International Conference on Networking, Architecture and Storage (Nas 2015). Boston: IEEE, 2015:106-111.

(编辑:徐敏)

(上接第 38 页)

[7] 杨力,孔志翔,石怀峰. 软件定义空间信息网络多控制器动态部署策略[J]. 计算机工程, 2018, 44(10): 64-69.

[8] WU S, LIU X F, CHEN Q, et al. Update Method for Controller Placement Problem in Software-Defined Satellite Networking[C]//28th International Conference on Computer Communication and Networks (ICCCN). Valencia, Spanish:[s. n.] 2019: 1-7.

[9] XU S, WANG X W, GAO B Y, et al. Controller Placement in Software-Defined Satellite Networks [C]//2018 International Conference on Mobile Ad-hoc and Sensor Networks. Shenyang, China: [s. n.], 2018: 146-151.

[10] HU H F, ZHANG S S, TANG B H. LEO Software Defined Networking Based on Onboard Controller [C]//2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP). California, USA: IEEE, 2018: 1068-1071.

[11] WU S, CHEN X Q, YANG L, et al. Dynamic and Static Controller Placement in Software-Defined Satellite Networking[J]. Acta Astronautica, 2018, 152: 1-10.

[12] Papa A, Cola T D, Vizarreta P, et al. Dynamic SDN Controller Placement in a LEO Constellation Satellite Network [C]//2018 IEEE Global Communications Conference (GLOBECOM). Abu Dhabi, UAE:IEEE, 2018: 9-13.

[13] 刘治国, 卢美玲, 李慧, 等. 基于 SDN 的卫星网络多控制器部署方法研究 [J]. 计算机仿真, 2020, 37(4): 62-66,97.

[14] XIA D, ZHENG X, DUAN P, et al. Ground-Station Based Software-Defined LEO Satellite Networks[C]// 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS). Tianjin, China: IEEE, 2019: 687-694.

[15] LING T, LIU L, ZHENG C, et al. An Optimized Forward Scheduling Algorithm in a Software-Defined Satellite Network[C]//2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA). Lagos, Nigeria: IEEE, 2019: 27-32.

[16] XU S, WANG X W, HUANG M, et al. Software Defined Next Generation Satellite Networks: Architecture, Challenges, and Solutions[J]. IEEE Access, 2018, 6: 4027-4041.

[17] BI Y G, HAN G J, XU S, et al. Software Defined Space-Terrestrial Integrated Networks: Architecture, Challenges, and Solutions[J]. IEEE Network, 2019, 33(1): 22-28.

[18] LEOPLOD R J, MILLER A. The Iridium Communications System[J]. IEEE Potentials, 1993, 12(2): 6-9.

[19] LONG F. Satellite Network Robust QoS-Aware Routing [M]. Berlin: Springer, 2014:41-74.

(编辑:韩茜,徐楠楠)