文章编号:1671-4598(2021)05-0169-05 DOI:10.16526/j.cnki.11—4762/tp.2021.05.034 中图分类号:V302.8 文献标识码:A

FPGA 块存储器的多位翻转容错设计

钟 敏[1,2], 苏海冰[2], 潘广涛[2]
(1. 中国科学院大学 计算机科学与技术学院, 北京 100049
2. 中国科学院光电技术研究所, 成都 610209)

摘要: 在高可靠性航空航天、航空电子设备和军用应用中, 辐射引发的多比特翻转 (MBU) 成为 FPGA 存储器的一个主要的可靠性问题; 传统的单比特错误纠正 (SEC) 和双比特错误检测 (DED) 无法对 FPGA 存储器发生的 MBU 故障提供防护, 引发存储器的存储故障; 为了减少 MBU 造成的影响, 设计了 RM (2, 5) 编码防护系统对 FPGA 块存储器进行容错防护, 实现了单个码字小于 4 位的翻转错误的纠正; 对 RM 编码系统进行了三模冗余设计, 解决了 RM 码不具备抗辐射的缺陷; 设计的 RM (2, 5) 编译码模块在 Xilinx Virtex—5 FPGA 中实现, 编码模块频率以 225.284 MHz 运行, 占用 LUT 资源 1.33%; 通过理论分析和硬件实验表明, 该错误检测与纠正 (EDAC) 系统能够纠正 4 位以下的翻转, 提高 FPGA 存储器的可靠性。

关键词: 多比特翻转; 三模冗余; 块存储器; 错误纠正码

Multiple Bit Upset Fault Tolerance Design of FPGA Block Memory

Zhong Min[1,2], Su Haibing[2], Pan Guangtao[2]
(1. School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China;
2. Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu 610209, China)

Abstract: In high—reliability aerospace, avionics and military applications, radiation—induced multiple bit upset (MBU) has become a major reliability concern in FPGA memory cells. Traditional single—bit error correction (SEC) and double—bit error detection (DED) cannot provide protection against MBU fault in FPGA memory, causing memory storage failures. In order to reduce the impact of MBU, the RM (2, 5) coding system is used for fault—tolerant protection of FPGA block memory.. Realize the correction of the flip error of a single code word less than 4 bits. The RM coding system is designed with triple module redundancy (TMR), which solves the defect that the RM code does not have radiation resistance. RM (2, 5) codec module has been implemented in Xilinx Virtex—5 FPGA and the codec module runs at 225.284 MHz, which only 1.33% LUT resource occupation. Theoretical analysis and hardware experiments show that the Error Detection And Correction (EDAC) system can correct errors of less than 4 bits and improve the reliability of the FPGA memory.

Keywords: multiple bit upset; TMR; BRAM; error correct code

0 引言

为了满足不断增长的性能和功率需求, FPGA 制造技术流程需不断演变, 采用先进的工艺节点制造, 这使得 FPGA 组件尺寸不断缩小形成高逻辑密度器件。在这么小的几何结构器件中, 太空环境单个辐射粒子撞击器件所引发的单粒子翻转 (single error upset, SEU) 效应已不限于单比特翻转 (single bit upset, SEU) 事件, 单粒子撞击甚至会导致存储器阵列的多个相邻位的翻转, 即多比特翻转 (multiple bit upset, MBU) 事件[1]。器件物理尺寸的减小还会导致电源和阈值电压的比例下降[2], 导致 FPGA 器件受太空辐射影响的概率不断增大, 进一步导致 FPGA 存储单元发生 SEU 的概率增加。

FPGA 的存储资源主要有配置存储器和用户设计中的存储单元组如: BRAM、DRAM、触发器等构成。BRAM 作为 FPGA 的第二大内存, 提供了大量的高容量存储单元用于存储用户设计状态, 实现 FPGA 内部数据的随机存储。一旦 BRAM 发生翻转故障, 会导致 FPGA 逻辑电路发生异常, 严重时甚至会导致电路失效。因此本文主要针对 BRAM 进行抗 SEU 防护。

1 FPGA 的 MBU 事件

FPGA 缓解 SEU 的防护可从工艺、版图设计和系统设计等不同层面进行, 但工艺和版图设计代价昂贵且设计周期较长, 目前的抗辐射加固主要是在系统设计层面上进行。缓解 SEU 最常用的系统容错方案是采用具有单比特纠错和双比特检错 (SEC—DED) 的汉明码技术。但是, 随着制造工艺节点的减小使得太空辐射环境中 FPGA 的 MBU 成为高发事件。在采用的 45 纳米技术中, MBU 事件占 FPGA 软错误的概率高达 48.28%[1], 而且对于更小的技术节点,

收稿日期:2020-09-11; **修回日期**:2020-11-02。
作者简介:钟 敏(1996-),女,广西博白人,硕士,主要从事航天 FPGA 容错技术方向的研究。
通讯作者:苏海冰(1969-),男,博士,研究员,主要从事电子学系统设计与仿真测试方向的研究。
引用格式:钟 敏,苏海冰,潘广涛.FPGA 块存储器的多位翻转容错设计[J].计算机测量与控制,2021,29(5):169-173,178.

该比率也相对变高[3]。传统的 SEC－DED 技术已无法满足 FPGA 存储器纠正多位错误的需求。MBU 的发生主要有单粒子撞击造成的多位翻转（single event multiple upset，SEMU）和一位翻转的积累这两种形式，后者的消除通过缩短系统的整个刷新时间即可实现，因此，目前讨论的多比特翻转多指 SEMU。它是由单个高能粒子以某种入射角击中存储器的敏感区造成相邻存储单元的多个瞬时错误[4-5]，也称之为多比特突发错误（burst error，BE）。

为了能够纠正 MBU 软错误，国内外学者提出了多种设计方案，文献［6］采用 2－D 海明码校正配置存储器的多位翻转，可用性超过 99.9999999%。文献［2］采用 InD 奇偶校验编码来检测配置存储器的 MBU，利用纠删码技术重建码字实现纠错，能检测 100%MBU，且资源占用率仅为 3.3%。文献［7］将多个存储器交错排列成阵列，把单个码字分散到多个存储器，从而使单个存储器上产生的逻辑相邻的多位翻转错误分散到不同码字，使用传统的 SEC－DED 方法即可进行纠错，但是这样需要大量的硬件资源。文献［8］基于信息和硬件冗余的组合，提出一种奇偶校验字节和复制（Parity per Byte and Duplication，PBU）的 EDAC 技术，将每个存储字分成字节段，奇偶校验位与每个段相关联实现多位纠错，使用的存储块仅占 3.85%。在国内，文献［9］提出一种改进型的（14，8）循环码实现了存储器 MBU 的纠错检错，代码易于实现，且实时性高。文献［10］提出一种位交错技术，将它与汉明码结合，可实现存储器 MBU 的 99.82%纠错。文献［11］通过将两个能纠正一位随机错误和二位突发错误的（13，8）系统码结合，交织度为 2 的（26，16）交织码，实现了存储器的多位翻转纠错。诸如其它纠错能力强的错误检查和纠正（error correcting code，ECC）码，比如 BCH（Bose Chaudhuri Hocquenghem）码、RS（Reed－Solomon）码、矩阵码等，具有速度和时滞的缺点。

本文使用 RM（Reed－Muller）码来进行多位翻转纠错，它是最老的错误纠正线性码之一[12]，所实现的 RM（2，5）码构造简单，结构特性丰富，占用资源少，可实现少于 4 位的翻转纠错，而且具有低时滞和高性能的特点。通过对编码系统进行了可靠性分析，利用故障注入进行了仿真实现和实物实验，从理论分析和硬件实验验证证明该编码可提高 FPGA BRAM 的可靠性。

2 RM 码的原理

RM 编码与其他纠错检错码一样，均需要经过编码和译码两个过程。

2.1 RM（2，5）编码

对于任意整数 m 和 r，$0 \leqslant r \leqslant m$ 存在一个二进制码长为 2^m 的 r 阶 RM 码，记为 RM（r，m），其主要参数由三部分[13]：

$$\text{码长}:n = 2^m \tag{1}$$

$$\text{维数}:k(r,m) = \sum_{i=0}^{r} C_m^i \tag{2}$$

$$\text{最小码距}:d_{\min} = 2^{m-r} \tag{3}$$

式中，$C_m^i = \dfrac{m!}{i!\ (m-i)!}$是二项式系数，即 RM（2，5）码是两阶的二进制线性组合代码，其信息位为 16 bits，生成的编码字为 32 bits，最小码距 $d_{\min}=8$，由 $d_{\min}$ 可知其能纠正的最大错误位数为：

$$t = (d_{\min}-1)/2 = 3^{[13]}$$

设信息码为 $\boldsymbol{m}=（m_15，m_14，\cdots，m_0）$，生成矩阵为 $\boldsymbol{G}$，则 R（2，5）码的编码生成的编码字 C 为：

$$C = m \times G \tag{4}$$

RM（2，5）码共有 5 个变量：$X1$、$X2$、$X3$、$X4$、$X5$，它可以纠正大多数的三位随机翻转错误，其生成矩阵定义为 $\boldsymbol{G}=（1，G_1，G_2）$[14]。其中：

$$\boldsymbol{G}_1 = (X_1, X_2, X_3, X_4, X_5,)^T$$

$$\boldsymbol{G}_2 = (X_1\ X_2, X_1\ X_3, X_1\ X_4, X_1\ X_5, X_2\ X_3, X_2\ X_4, X_2\ X_5, X_3\ X_4, X_3\ X_5, X_4\ X_5)$$

2.2 RM（2，5）译码

RM（r，m）为正交结构，因此译码通常采用大数逻辑译码（majority logic decoder，MLD），整个译码过程由 3 个阶段构成[15]。

第一阶段先译码出 10 个信息位（m_9，…，m_0），而且每个信息位都由 8 位校验和来确定，以 m_9 为例：

$$\begin{aligned}
S_{9,7} &= C'_0 + C'_1 + C'_2 + C'_3 \\
S_{9,6} &= C'_4 + C'_5 + C'_6 + C'_7 \\
S_{9,5} &= C'_8 + C'_9 + C'_10 + C'_11 \\
S_{9,4} &= C'_12 + C'_13 + C'_14 + C'_15 \\
S_{9,3} &= C'_16 + C'_17 + C'_18 + C'_19 \\
S_{9,2} &= C'_20 + C'_21 + C'_22 + C'_23 \\
S_{9,1} &= C'_24 + C'_25 + C'_26 + C'_27 \\
S_{9,0} &= C'_28 + C'_28 + C'_30 + C'_31
\end{aligned} \tag{5}$$

其中：C'为译码器接所收到的编码字，通过对 8 位校验和进行多数表决决定该译码信息位是 0 还是 1，即：分别比较 0 和 1 的个数，若 0 的个数多，则该信息位为 0，否则为 1；若 0 和 1 的个数相等，则表示为发生 4 位错误，为未知信息位，无法进行正确译码。

第二阶段的译码需要生成 16 位中间码字 C''，由公式（6）计算得到，该中间字的值跟第一阶段译码的信息位相关。该阶段译码出 5 个信息位（m_14，…，m_10），每个信息由 16 个校验和进行多数表决确实，式（7）给出 m_14 的 16 位校验和计算公式。

$$\boldsymbol{C''} = C' - [m_9 \cdots m_0]\, G_2 \tag{6}$$

$$\begin{aligned}
S_{14,15} &= C''_0 + C''_1 \\
S_{14,14} &= C''_16 + C''_17 \\
S_{14,13} &= C''_8 + C''_9 \\
S_{14,12} &= C''_24 + C''_25 \\
S_{14,11} &= C''_4 + C''_5 \\
S_{14,10} &= C''_20 + C''_21 \\
S_{14,9} &= C''_12 + C''_13
\end{aligned}$$

$$S_{14,8}=C''_{28}+C''_{29} \quad (7)$$
$$S_{14,7}=C''_{2}+C''_{3}$$
$$S_{14,6}=C''_{18}+C''_{19}$$
$$S_{14,5}=C''_{10}+C''_{11}$$
$$S_{14,4}=C''_{26}+C''_{27}$$
$$S_{14,3}=C''_{6}+C''_{7}$$
$$S_{14,2}=C''_{23}+C''_{22}$$
$$S_{14,1}=C''_{14}+C''_{15}$$
$$S_{14,0}=C''_{30}+C''_{31}$$

第三阶段译码出最后一位信息位 m_{15}，该信息位的译码需要生成 32 位中间码字 C'''，由公式（8）计算得到。该阶段无需生成校验和，可直接对 C''' 进行多数表决得到 m_{15}。同时 C''' 也指示了 BRAM 数据的具体翻转位置，若无翻转故障发生，C''' 的所有位全为‘0’或者‘1’，发生翻转时，若 1 的个数大于 0 的个数，则不为‘1’的那几位所在的位置即为数据翻转位；相反，若 0 的个数多，则不为‘0’的那几位所在位置即为数据翻转位。

$$\boldsymbol{C'''}=C''[m_1 4\cdots m_1 0]\,G_1 \quad (8)$$

至此，经过 3 个阶段的译码，译码器将接收到的 32 位编码字译码成 16 位信息位，并完成纠错检错。RM（2，5）码译码思想总结为：

（1）在 RM（2，5）编码矩阵的每行找到 $2^{m-r}=8$ 个特征矢量，将其与接收到的码字 C 进行点乘得到 8 位校验和。

（2）将每个译码位所得的校验和 0 和 1 的个数进行判断，若 0 的个数多，则该译码位为 0，否则为 1。

（3）将（2）中得到的译码位与生成矩阵 G_2 相乘，用接收到的码字 C' 减去相乘结果得到新的码字 C''，在 C'' 找到新的特征矢量求出 16 位校验和，重复（2）。

（4）将（2）中得到的译码位与 G_1 相乘，用 C'' 减去相乘结果得到新的码字 C'''，重复（2）。

（5）将（2）（3）（4）中得到的所有译码信息位进行整合输出。

3　BRAM 防护系统实现

Xilinx Virtex－5 FPGA 的 BRAM 其内嵌有可实现 SEC－DED 的汉明码，用于缓解 SEU 的 SBU 效应，一旦发生 MBU，内嵌的汉明码防护无法起到防护作用。常用的解决方案有：对 BRAM 进行 TMR，屏蔽单个存储器的 SEU 事件；或者为 BRAM 提供具有纠错能力更强的 ECC 防护，在存储数据前进行编码防护，读取数据时进行译码修复。

3.1　故障注入

FPGA 常用的可靠性评估方法主要为辐射实验和故障注入。辐射实验使用粒子加速器提供的各种粒子来源以模拟太空辐射环境，通过对目标器件进行照射实现。该方法可以得到较为真实的实验数据，但是目前国内能提供实验条件的设备不多，而且实现操作复杂、器件价格昂贵并且实验周期过长。

故障注入技术通过对待测电路模块进行定点或者随机的故障注入，通过观察分析模块电路的输出结果，可得到被测模块电路的可靠性评估结果。该技术可以弥补辐射实验的不足，且成本低、操作简便，已成为 FPGA 评估单粒子效应的一种重要方法。本文通过对 BRAM 存储的数据进行随机故障注入，进行 0/1 翻转，验证 RM 码防护时 BRAM 的可靠性。

3.2　BRAM 的 TMR 防护

TMR 是一种易于实现的静态冗余防护方法，将电路模块进行三备份，最终通过表决器进行 3 取 2 的表决输出，达到屏蔽单个电路模块失效的故障[16]。由于在同一时刻，3 个模块均发生单粒子翻转事件的概率较低，且电路实现过程相对简单，被广泛应用于缓解 SEU。因此，对 BRAM 进行三模冗余可提高存储器的可靠性。

尽管 TMR 易于实现，有效缓解 SEU，且提高 BRAM 的可靠性，但其也存在以下缺陷[17]：

1）只能对错误进行屏蔽不能进行纠错。单个模块发生故障后，只能将其屏蔽，由于不具备纠错功能，且翻转错误的累积，最终会导致 TMR 防护失效，甚至导致系统崩溃。

2）尽管多个模块同时发生 SEU 的概率较低，但仍有可能发生，对系统可靠性来说是一个较大的威胁。

3）BRAM 的 TMR 防护导致硬件资源开销增大，其资源利用率在大型设计中较低，且布线资源会增加系统的延迟。

4）表决器本身不具备抗辐射的能力，也可能发生 SEU 事件，导致表决出错。

因此，单纯的对 BRAM 进行 TMR 防护，在一定程度生可有效缓解 SEU，提高系统的可靠性，但随着时间的推移，未及时修复的故障会积累，TMR 的可靠性也会下降。

3.3　BRAM 的 RM（2，5）码防护

使用纠错能力更强的 ECC 码进行防护，可节省硬件资源开销，提高资源的利用率。RM（2，5）码对 BRAM 的防护原理如图 1 所示，数据处理单元控制 BRAM 的读写以及故障注入，RM（2，5）模块实现数据的编译码故障注入模块实现故障模拟。

数据处理单元发出写操作指令时，16 位原始数据通过 encin 端口送入编码器，编码器进行编码后将 32 位编码字由 encout 端口写入 BRAM；当数据处理单元发出读操作指令时，BRAM 中的 32 位数据先从 decin 端口输入到译码器进行解码，如果块存储器中的数据发生翻转故障，解码器能够对翻转的数据位进行纠正。故障注入模块用来进行故障注入测试，根据处理器单元提供的注入地址从 BRAM 中读取相应的字对其进行 0/1 翻转，并将翻转后的数据重写回 BRAM，模拟 BRAM 发生翻转故障的状态。error 指示故障状态，error 输出“00”表示无故障发生，“01”表示检测到一位、两位或者 3 位翻转错误，并进行了修复，“10”表示检测到四位翻转错误，无法进行修复，“11”表示检测到无效位错误。

RM（2，5）编译码原理图如图 2 所示。存入数据时，

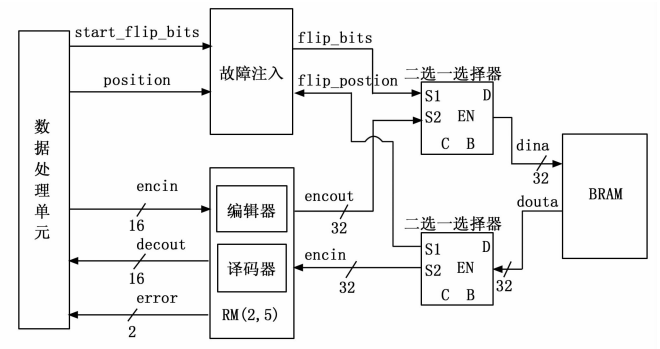

图 1 BRAM 的防护原理

16 位原始信息码 M（15：0）从 encin 端口进入编码器与 32 位生成矩阵 ***G*** 进行异或运算，得到的 32 位编码字存入 BRAM 中。读取数据时，从 BRAM 读取的 32 位译码数据从 decin 端口进入译码器，第一阶段译码出 M（9：0），第二阶段译码出 M（14：10），第三阶段译码出 M_{15}，最后从 decout 端口输出 16 位译码字，译码过程中同时修复存储器中的最多 3 位翻转错误，同时指示了错误发生的位置，以及错误发生的位数，译码结果输出到处理器。

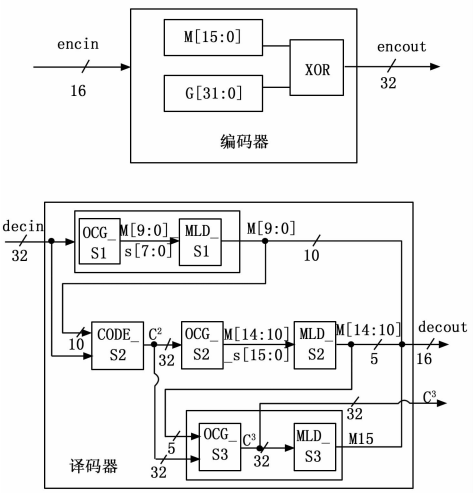

图 2 RM（2，5）码编译原理图

3.4 编码系统的三模冗余设计

由 RM（2，5）码组成的 EDAC 电路，并不具备抗辐射的能力，在太空辐照环境中，编译码电路也可能发生 SEU 事件，导致在编码和译码过程中出现翻转故障，无论是编码数据还是译码数据均会出现错误。因此通过将 TMR 与 EDAC 相结合，屏蔽单个编译码电路模块的故障，可有效避免 EDAC 电路的 SEU 事件，提高系统的可靠性。其原理框图如图 3 所示，将编码器和译码器分别进行三备份，数据同时进入 3 个模块，通过表决器进行 3 取 2 表决输出，提高了 EDAC 的可靠性。

4 可靠性与实验验证

通过理论分析对比不同条件下 BRAM 的可靠性以及

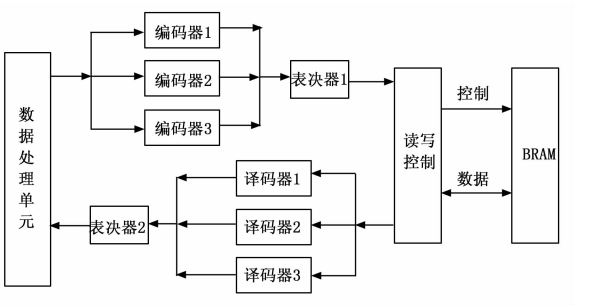

图 3 EDAC 的 TMR 框图

RM（2，5）防护系统的资源利用率，最后通过仿真和实验验证，证明 RM（2，5）码可实现小于 4 位的翻转修复。

4.1 RM（2，5）码可靠性分析

FPGA 块存储器单元未进行容错设计时其可靠性基本服从泊松分布[18]，即假设每个 BRAM 单元发生 SEU 的概率为，在时间 t 内未发生 SEU 的概率为：

$$P(t) = e^{-\lambda t} \tag{9}$$

假设一个存储大小为 m 的 BRAM 其输入输出位宽为 n，则它在时间 t 内发生 i 位翻转的概率为：

$$P(t) = \left[C_n^i (1-e^{-\lambda t})^i e^{-(n-i)\lambda t}\right]^m \tag{10}$$

其中：i=0，1，2，3。RM（2，5）码能纠正单个码字中的 3 位翻转错误，因此，在时间 t 内 BRAM 的可靠性为：

$$R_i(t) = \left[\sum_{i=0}^{3} C_n^i (1-e^{-\lambda t})^i e^{-\lambda(n-i)t}\right]^m \tag{11}$$

其中：i=0，1，2，3。在 NASA 提供的试验用粒子束条件下测得每个 BRAM 单元发生单粒子翻转的概率 $\lambda = 1.1 \times 10^{-6}$ b/d[19]。图 4 是存储器大小分别为 m=512 byte 和 m =1 024 byte 下，在无防护、（12，8）汉明码防护和 RM（2，5）码防护 3 种条件下由 MATLAB 计算得到的可靠性对比。其中 R0（t）是无防护时的可靠性曲线，R1（t）是（12，8）汉明码防护下的可靠性曲线，R2（t）是 RM（2，5）码防护下的可靠性曲线。由图可见，在相同存储器大小下，RM（2，5）码的可靠性明显更优；在相同的防护条件下，容量小的存储器具有更高的可靠性。

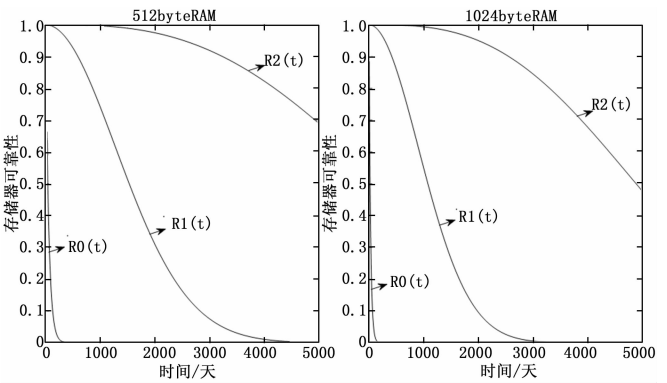

图 4 可靠性对比

4.2 资源分析

本文设计的 RM（2，5）码防护系统利用 Xilinx ISE14.7 设计工具在 Virtex5 FPGA（xc5vfx70t）上用 VHDL 实现，

工作时钟频率为 50 MHz。译码器和编码器主要是采用组合逻辑方式实现，整个编码系统的延迟由所占用的异或逻辑数量决定，编译综合后得到编码器模块 LUT 利用率约为 0.107%，译码器模块 LUT 利用率约为 1.12%，实现后整个编码模块的延迟为 4.456 ns。RM（2，5）编译码模块三模冗余后，其硬件资源约为原始的三倍左右，如表 1 所示。

表 1　RM 编译码模块资源　　%

逻辑利用率	RM(2,5)	TMR RM(2,5)
寄存器数量	0.748%	2.29%
LUT 数量	1.33%	3.70%

4.3　硬件实验平台验证

实验前对该防护系统进行了连续注入故障的仿真测试，分别对存储的数据进行 1 位、2 位以及 3 位的翻转。仿真结果表明可纠正不少于 3 位的翻转故障。仿真结果如图 5 所示。故障注入模块 inf _ error（2：0）端口对 BRAM 进行 3 位随机翻转，可以看到经过译码器修复后，error 端口输出 1，3 位翻转错误被修复。

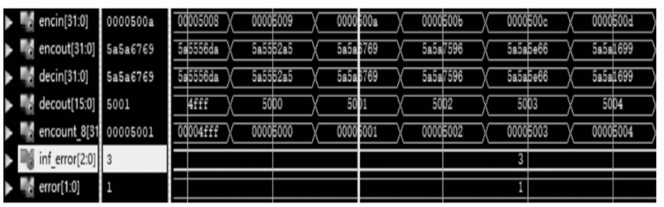

图 5　仿真结果

经过仿真验证后，初步验证了该防护可修复不少于 3 位的翻转，进一步利用 FPGA 内部 ICAP 接口将配置 RAM 的配置帧内容回读写入 BRAM 作为原始存储数据。Virtex－5 每个配置帧大小为 41×32 位，所设计的 RM（2，5）编码系统其编码输入宽度为 16 位，通过将其级联实现 32 位的输入。本实验采用单端口的 BRAM，存储宽度为 64 位，深度为 41。使用 Xilinx ISE14.7 开发工具套件提供的 ChipScope™ Pro 工具抓取读取 BRAM 数据时的信号波形图，观察其存入 BRAM 的数据、故障注入后的数据以及纠错后的数据，实验结果与仿真结果相同。

图 6（a）为 BRAM 写入数据的波形图，当 ICAP 回读计数器 rb _ cmd _ cnt 计数到 71 时，BRAM 地址 addra 从 0 加到 1 开始写入数据，原始数据首先从 encin 端口进入编码器进行编码，编码字落后 1 个时钟由 dina 端口存入 BRAM 中。

图 6（b）显示了故障注入的过程。翻转信号 start _ flip _ bit 在时钟上升沿处拉高，表示开始从 BRAM 的 douta 端口读取地址 07，对位地址为 25～27 的 3 位数据进行翻转，图中可以看到相应的位 25～27 由“0”变为“E”，翻转结束后通过 dina 端口将数据写回 BRAM 中，flip _ start _ done 拉高表示一轮故障注入结束 。

图 6（c）是数据处理单元从 BRAM 读取数据并实现三位纠错的过程，当 ICAP 写计数器 wf _ cmd _ cnt 计数到 21 时，BRAM 的 douta 端口开始输出数据，先进入译码器进行译码，当 wf _ cmd _ cnt 计数为 37 时，可以看到，译码器 decout 端口输出的数据为正常数据，翻转的三位被修复。

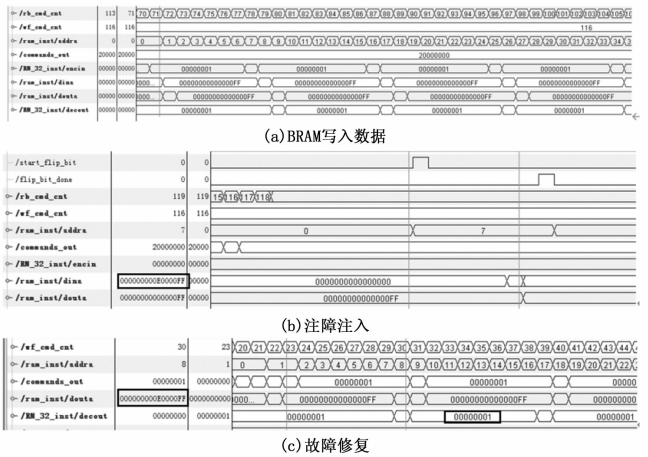

图 6　实验分析

经过仿真验证以及实物验证，均证明 RM（2，5）码可修复 BRAM 中发生的不少于三位的翻转错误，有效提高 FPGA 块存储器的可靠性。

5　结束语

辐射诱发的 MBU 是 FPGA 的一个严重的可靠性问题，器件晶体管的减少、集成度的提高，使得 FPGA 的 MBU 概率变高。BRAM 作为 FPGA 的第二大内存，一旦发生 MBU，其设置的 SEC－DED 防护将会失效，造成所存储的用户数据出现错误，从而引发电路逻辑故障。

本文利用 RM（2，5）码对 FPGA 的 BRAM 进行了 EDAC 防护，理论分析了防护系统的可靠性，并且通过仿真和实物测试实验，结果表明该方法可实现对存储器不少于 3 位翻转故障的修复，验证了 RM（2，5）码组成的防护系统的有效性和可靠性。将 TMR 与 EDAC 模块相结合，提高了 EDAC 模块的可靠性，相比较传统的汉明码技术，具有更好的防护效果，该防护系统可灵活应用于寄存器和存储器的防护中。下一步，将继续研究 FPGA 配置存储器配置帧的 MBU 修复，将用 RM（2，5）防护的 BRAM 的来存取配置帧的配置数据，保证在整个配置帧的重配置中配置数据的可靠性。

参考文献：

[1] Ebrahimi M，Rao P M B，Seyyedi R，et al. Low－Cost Multiple Bit Upset Correction in SRAM－Based FPGA Configuration Frames [J]. IEEE Transactions on Very Large Scale Integration Systems，2016，24 (3)：932－943.

[2] Frigerio L，Radaelli M A，Salice F A. Generalized Approach for the Use of Convolutional Coding in SEU Mitigation [A]. IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems [C]. 2008，427－435.

（下转第 178 页）