基于关键路径的三模冗余表决器插入算法

谭宜涛[①②] 杨海钢[*①] 黄 娟[①②] 郝亚男[①②] 崔秀海[①]
[①](中国科学院电子学研究所可编程芯片与系统实验室 北京 100190)
[②](中国科学院研究生院 北京 100190)

摘 要：在 FPGA 的三模冗余设计中，寄存器的反馈环路会导致错误持续出现，严重影响三模冗余的容错性能，因此需要在寄存器的反馈环路上插入表决器。该文首次提出了一种针对映射后网表进行三模冗余设计的方法，同时提出了基于关键路径的表决器插入算法，该算法在表决器的插入时避开关键路径，缓解了三模冗余设计中插入表决器时增加延时的影响。与国外同类算法相比，该文算法在不降低电路可靠性的前提下，以不到 1%的面积开销，使得关键路径延时减少 3%~10%，同时算法运算速度平均提高 35.4%。

关键词：FPGA；三模冗余；表决器插入；映射后网表；关键路径
中图分类号：TN406 **文献标识码**：A **文章编号**：1009-5896(2012)02-0487-06
DOI: 10.3724/SP.J.1146.2011.00571

Voter Insertion Algorithm Based on Critical Path for Triple Module Redundancy

Tan Yi-tao[①②] Yang Hai-gang[①] Huang Juan[①②] Hao Ya-nan[①②] Cui Xiu-hai[①]
[①](*System on Programmable Chip Research Department*, *Institute of Electronics*, *Chinese Academy of Sciences*, *Beijing* 100190, *China*)
[②](*Graduate University of the Chinese Academy of Sciences*, *Beijing* 100190, *China*)

Abstract: In the Triple Module Redundancy (TMR) design for the FPGA, the feedback loop of the register will lead to the persistent errors which would have a negative impact on the fault-tolerant capability of the triple module redundancy design, hence the voter insertion in the feedback loop is necessary. This paper presents a triple module redundancy design method to the mapped netlist for the first time, and proposes a voter insertion algorithm based on the critical path. This algorithm proposed can avoid inserting the voter in the critical path and alleviate the negative impact on timing performance during voter insertion. Compared with the similar algorithms, the proposed algorithm can reduce the critical-path delay by 3% to 10% and improve the run time averagely by 35.4% while keeping the design reliability non-decreasing with less than 1% area penalty.

Key words: FPGA; Triple Module Redundancy (TMR); Voter insertion; Mapped netlist; Critical path

1 引言

随着 SRAM 型 FPGA 器件在航天航空领域的广泛应用，研究 SRAM 型 FPGA 抗单粒子翻转的方法越来越迫切。目前，提高 SRAM 型 FPGA 抗单粒子性能的方法主要包括两种：三模冗余(Triple Module Redundancy, TMR)设计和 SRAM 刷新。通过 TMR 设计和 SRAM 刷新两种方法的配合使用，SRAM 型 FPGA 的抗单粒子翻转问题基本得以解决[1−3]。

在 TMR 设计中必须在反馈环路逻辑中插入多数表决器来消除持续性错误[4]，而表决器的插入位置和数量对 TMR 设计电路的时序、面积和可靠性影响很大[5]。一些研究者提出了选择性的部分 TMR 设计从而减少 TMR 设计所需的面积[6−10]，但是这些方法只适用于特定的电路类型且易导致可靠性降低，同时这些方法没有对表决器的插入方法进行研究。文献[11]对有限数字滤波器的 4 种不同的表决器插入方案进行实验和可靠性分析，但是没有从理论上对表决器插入的优化方法进行研究。为了优化表决器的插入位置，文献[12]首次提出了几种 TMR 表决器插入算法来确定表决器的插入位置，但是算法的时序性能不是很理想，且算法时间复杂度较高。同时，现有 TMR 设计都是针对综合后网表进行展开[13,14]，而综合后网表还需要经过逻辑映射转换成 FPGA 的器件库才能进一步进行布局布线操作，因

2011-06-13 收到，2011-09-01 改回
国家重大科学研究计划(2011CB933202)资助课题
*通信作者：杨海钢 ic_design_group@mail.ie.ac.cn

而对综合后网表进行 TMR 设计可能导致映射后的电路和 TMR 设计意图不符。

因此，本文首次提出了针对映射后网表进行 TMR 设计的方法，用来消除 TMR 设计中的持续性错误，同时提出了一种基于关键路径的 TMR 表决器插入算法，避免了在电路的关键路径上插入表决器。通过 ITC99 测试电路实验的结果表明，相比文献[12]中的优化算法，本文算法在不降低电路可靠性的前提下以少量的面积开销有效减少了关键路径延时和算法运行时间。

2 映射后网表的 TMR 设计方法

文献[13,14]都是基于综合后网表直接进行 TMR 设计，得到的冗余网表再经过映射和布局布线进行处理，其网表级 TMR 设计的输出文件都是电路设计接口格式(Electronic Design Interchange Format, EDIF)网表。EDIF 网表是符合工业标准的描述电路结构的网表形式，其电路构成基本单元包括 LUT、寄存器、多路选择器、异或门以及信号连线等库单元。

和前人基于综合后网表 TMR 设计不同的是，本文首次提出对 FPGA 映射后的网表进行 TMR 设计，映射后网表中节点的基本形式为 FPGA 的基本逻辑单元[15](Logic Element, LE)，一个 LE 包括 LUT、触发器和进位链等资源。映射后的电路网表更接近于 FPGA 电路布局布线后的结果，映射后再进行网表 TMR 设计可以避免冗余网表在电路映射过程中被优化，从而很好地保持了 TMR 冗余设计的意图。由于 FPGA 中的块存储器和锁相环等 IP 的 TMR 设计较复杂且设计方法不具有通用性，本文的研究重点针对 LE 的 TMR 设计。

每一个 LE 根据所实现的逻辑功能其输出可以分为 3 类：组合输出、寄存输出和进位链输出。同一个 LE 可以只有一个输出，也可以同时具有多个输出，在有向图的建立过程中，根据 LE 的输出个数将该 LE 分别建立成组合模式、寄存模式和进位链模式的多个逻辑节点。另外，不同厂家和系列的 FPGA 的 LE 结构不同，因此在进行有向图建模时，必须对该 FPGA 中 LE 输入输出端口之间的连接关系建立规则。

本文提出的映射后网表级 TMR 设计流程如图 1 所示。首先对 FPGA 映射后网表节点进行分析，根据 LE 输入输出端口连接规则进行有向图建模；然后对 IO 和内部逻辑资源进行三备份的逻辑复制；最后采用基于关键路径的表决器插入算法找出所有需要插入多数表决器逻辑的位置，在需要插入多数表决器的位置插入多数表决器，形成最终的 TMR 处理后网表。由于表决器的逻辑功能是固定的，需要插入的表决器逻辑可以使用固定的 LE 模型进行实现。

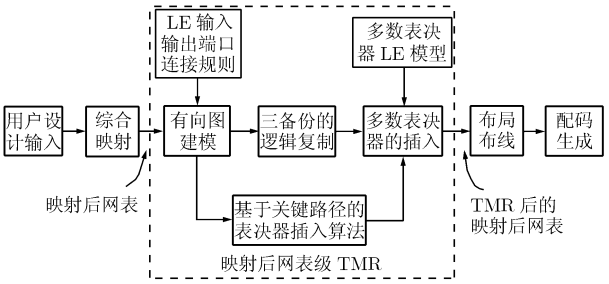

图 1 映射后网表级 TMR 的设计流程

3 基于关键路径的表决器插入算法

3.1 问题的提出

在本文中，将映射后同步时序电路建模成有向图 $G=(V, E)$，其中节点 V 代表电路中的逻辑节点，有向边 E 代表逻辑节点之间的连接关系。给定一个节点 $v\in V$, $\mathrm{FI}(v)$代表节点 v 的直接扇入节点集，$\mathrm{FO}(v)$代表节点 v 的直接扇出节点集，二者的定义如下所示：

$$\mathrm{FI}(v)=\{u:(u\to v)\in E\},\ \mathrm{FO}(v)=\{t:(v\to t)\in E\}$$

定义节点 v 的最长组合路径延迟 $d(v)$为有向图中逻辑节点 v 到其他任一个逻辑节点之间经过组合模式逻辑节点个数的最大值，其计算公式如式(1)。

$$\left.\begin{aligned} &d(v)=0,\ \mathrm{FO}(v)=\varnothing \text{ or } \mathrm{FO}(v)\text{的节点全部为寄存模式}\\ &d(v)=1+\max(d(i_1),d(i_2),\cdots,d(i_{|\mathrm{FO}(v)|}),\\ &\qquad i_1,i_2,\cdots,i_{|\mathrm{FO}(v)|}\in\mathrm{FO}(v)\end{aligned}\right\} \tag{1}$$

时序电路的电路性能取决于其关键路径的延时，寄存模式的逻辑节点的最长组合路径延迟越大，其后面的路径成为关键路径的可能性越大，因此寄存模式逻辑节点的最长组合路径延迟能够近似地反映映射后网表电路的时序性能。TMR 设计中确定表决器插入位置问题的实质是切断所有反馈环路并插入表决器从而消除持续性错误。由于表决器的插入数量和位置直接影响到经过 TMR 设计后加固电路的时序和面积，采用不同的算法选择不同的插入位置直接影响到 TMR 设计后时序和面积。为了使插入的表决器对 TMR 设计后电路的关键路径影响最小，必须减少由于表决器逻辑增加的关键路径节点上的最长组合路径延迟，即满足式(2)条件：

$$\mathrm{Min}D=\sum_{i=0}(d'(v_i)-d(v_i))\ \text{ and }\ \mathrm{Illegal(voter)}=\varnothing \tag{2}$$

在式(2)中，v_i 是为寄存模式且中最长组合路径延迟最大的那些逻辑节点，$d'(v_i)$ 和 $d(v_i)$ 分别表示插入表决器之后和之前的逻辑节点的最长组合路径延迟，Illegal(voter)代表非法位置插入的表决器的集合。

针对以上分析，本文提出了基于关键路径的表决器插入算法。本文算法通过对映射后网表进行时序分析，找出各个寄存器节点的最长组合路径延迟，并根据最长组合路径延迟来确定表决器的插入位置，使得添加的表决器逻辑尽量不出现在电路的关键路径上，从而优化了 TMR 设计后电路的时序性能。

3.2 基于关键路径的表决器插入算法

本文算法根据最长组合路径延迟来确定表决器的插入位置，当电路的同一个逻辑环路中存在多个寄存模式的 LE 时，选择在最长组合路径延迟最大的 LE 后面添加表决器，使得添加的表决器逻辑对关键路径影响降到最低。同时本文算法只在寄存模式的 LE 后面插入表决器，避免了表决器插入位置非法的情况，并可以保证两个寄存模式的 LE 之间最多插入一组表决器，而不会同时插入多组表决器导致 TMR 设计性能的进一步降低。

为了找到有向图中所有的反馈环路，本文采用强连通分量分解的方法[16]。有向图中反馈环路的节点都属于同一个强连通分量，因此通过强连通分量分解就可以发现有向图中的反馈环路。本算法的核心思想是，首先对有向图进行强连通分量分解找到网表逻辑的所有反馈环路，然后根据反馈环路中寄存模式节点的最长组合路径延迟来选择反馈环路中的边进行删除，重复对有向图进行强连通分量分解并不断删除其中的边直到强连通分量分解完成，此时所有的反馈环路都被消除，分解过程中被删除的边就是需要插入表决器的位置。最长组合路径延迟可以根据式(1)计算得到，本文算法中采用深度优先搜索算法进行实现。由于插入表决器的位置是在反馈环路中最长组合路径延迟最小的寄存器后面，避免了表决器逻辑对关键路径时序造成影响，因此本文算法从本质上是一种以时序优化为目标的算法。本文算法具体实现的伪代码如表 1 所示。

表 1 实现本文算法的伪代码

```
有向图 G 是从映射后网表提取得到的有向图；
链表 L 用于记录在强连通分量分解过程中删除的边，其中包含插入表决器的位置信息；
堆栈 S 用来保存强连通分量分解得到的强连通分量；
变量 leastCombDepth 表示当前最小的最长组合路径延迟；
变量 leastCombDepthFFNode 表示具有最长组合路径延迟最小的寄存模式节点。
(1)   清空链表 L
(2)   清空堆栈 S
(3)   for (G 中每个寄存模式的节点 v) {
(4)         最长组合路径延迟 CombDepth =DFS(v)
(5)   }
(6)   强连通分量 SCCs =SccDecomposition(G)
(7)   for (SCCs 中的每一组 SCC) {
(8)         S.push(SCC)
(9)   }
(10)  while (S 非空) {
(11)        SCC = S.pop()
(12)        leastCombDepth = +∞
(13)        leastCombDepthFFNode = null
(14)        for(SCC 中每个寄存模式的节点 v) {
(15)          if (节点 v 的 CombDepth < leastCombDepth) {
(16)                leastCombDepth = CombDepth
(17)                leastCombDepthFFNode = node
(18)          }
(19)        }
(20)        从有向图 G 中删除 leastCombDepthFFNode 节点的扇出边得到有向图 G'
(21)        将删除的边加入链表 L
(22)        newSCCs = SccDecomposition (有向图 G' )
(23)        for (newSCCs 中的每一组 newSCC) {
(24)              S.push(newSCC)
(25)        }
(26)  }
(27)  在链表 L 中的边对应的位置插入表决器
```

3.3 算法复杂度分析

本算法中有向图寄存模式节点的最长组合路径延迟只需计算一次。其中，while 循环的次数为 $O(|V|)$，强连通分量分解的操作复杂度为 $O(|V|+|E|)$，因而 while 循环体内部的操作复杂度为 $O(|V|+|E|)$，故整个算法的时间复杂度为 $O(|V|^2+|V||E|)$。

4 实验结果

实验中，FPGA 选用 Altera 公司 Cyclone 系列的 EP1C20F400I7，其资源包括 20060 个 LE, 294912 bit 片内存储器资源，301 个 IO。所用测试电路来自 ITC99 测试电路集，并分别记录其不同算法下的关键路径延时、使用的表决器组以及 LE 的数量。

分别采用文献[12]中的所有寄存器后面都添加表决器的算法 1、基于寄存器扇出数量的表决器插入算法 2 以及本文算法进行 TMR 设计，并对布局布线后的 TMR 设计结果进行面积和时序性能方面的比较。由于本算法能够避免在关键路径插入表决器，因而生成的 TMR 网表和其他两种算法相比，时序性能和面积延时积均得到了改善；同时本算法的复杂度相比算法 2 较低，减少了算法的运行时间。

4.1 关键路径延时和实现面积分析

由于 TMR 设计中需要大量的逻辑资源进行逻辑复制和表决器插入，TMR 设计在时序性能和面积上对原始设计影响很大。由表 2 可以得出，算法 1、算法 2 和本文算法进行 TMR 设计后电路在关键路径延时方面，相对于原始设计在具体各个电路上分别平均增加 20.7%、18.6%和 12.2%；在 LE 数量方面，算法 1、算法 2 和本文算法相对于原始设计在具体各个电路上分别平均增加 286.3%、259.1%和 261.5%。

3 种算法的 TMR 设计后电路在实现面积上的不同是由于在表决器的插入数量上有区别。算法 1 在所有寄存模式的节点处都插入表决器，导致的结果是插入了太多的表决器逻辑从而导致实现面积太大；而本文算法虽然是以时序优化为目标来确定表决器的插入位置，但是和算法 2 相比，本文在使用表决器和 LE 的数量上增加非常有限。而在 LE 数量方面，本文算法相对于算法 1 在具体各个电路上平均减少 6.1%，而相对算法 2 在具体各个电路上平均增加不到 1%。

表 2 表决器组和 LE 数量的比较

电路名称	原始设计		算法 1			算法 2			本文算法		
	关键路径长度 (ns)	LE 数量	表决器组的数量	关键路径长度 (ns)	LE 数量	表决器组的数量	关键路径长度 (ns)	LE 数量	表决器组的数量	关键路径长度 (ns)	LE 数量
b04	10.7	180	66	11.9	753	18	11.8	612	19	11.2	615
b05	8.9	203	34	10.7	783	23	10.6	756	25	9.8	762
b07	5.1	96	43	6.8	423	23	6.8	363	22	6.1	360
b10	3.9	68	17	5.1	243	10	4.6	222	12	4.3	228
b11	6.2	200	31	8	711	23	7.5	687	21	7.2	681
b13	3.7	87	49	5.3	387	24	5.2	354	28	5	366
b14	17.2	937	216	20.2	3522	130	20.3	3360	133	19.2	3369
b20	19.1	1936	431	20.8	6975	325	20.9	6843	330	20.1	6858
b21	18.4	1960	431	19.6	7233	387	19.8	7101	393	19.2	7119
b22	19.1	2923	614	20.4	10398	456	20.7	10203	460	20.1	10215
平均	11.2	859	193	12.9	3143	142	12.8	3050	144	12.2	3057

本文算法在强连通分量分解过程中根据最长组合路径延迟确定表决器的插入位置，因而 TMR 后电路中增加的表决器逻辑对时序性能的影响降到最低。由图 2 可以看出，本文算法相比算法 1 和算法 2 在时序性能方面提高明显，其关键路径延时在具体各个电路上分别平均提高了 6.8%和 5.2%。其中，和算法 2 进行比较，本文算法在时序性能方面根据不同的电路能够得到 3%～10%的改善。通过图 2 还可以看出，本文算法在规模较小电路上进行实现，相比较大电路在关键路径延时方面有更好的性能改善，这是因为大电路中面积资源利用率较高，布线资源相对紧张，造成布线延迟在关键路径延时中所占比重较大，而本文算法是通过映射后网表的最长组合路径延迟来预估关键路径，没有考虑布线延迟对关键路径的影响。

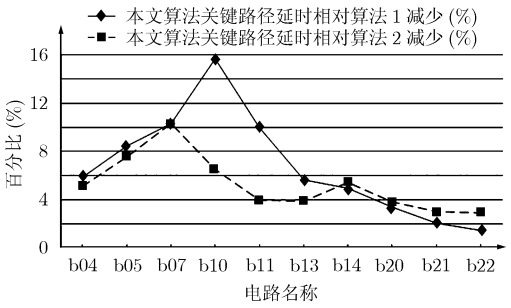

图 2 3 种算法时序性能的比较

本文算法相对算法 1，面积和关键路径延时都得到改善；但是相对于算法 2，在提高时序性能的同时牺牲了少量面积作为代价。为了评估 3 种算法在面积和时序性能的综合表现，本文通过面积延时积来进行比较。由于 FPGA 中的一个 LE 可以等效为固定门数的逻辑门，本文使用 LE 数量替代逻辑门数进行面积延时积的计算。图 3 是本文算法相对于算法 1 和算法 2 的面积延时积改善情况，其中面

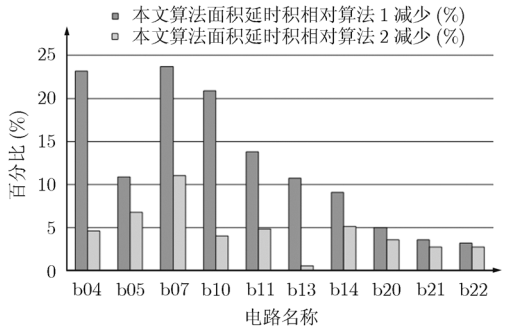

图 3　3 种算法面积延时积的比较

积延时积=LE 数量×关键路径延时。通过计算图中数据可以得出，本文算法相对于算法 1 在具体各个电路上平均减少 12.4%，而相对算法 2 在具体各个电路上平均减少 4.6%，因而本文算法在面积和时序性能的总体表现相对于算法 1 和算法 2 都要好。

4.2 算法运算时间比较

算法 1 在所有的寄存器后面都加入表决器，因而确定表决器位置的算法复杂度为 $O(|V|)$。算法 2 在算法执行过程中由于需要对寄存器的扇出数量进行重新计算，其时间复杂度为 $O(|V|^2|E|)$。而本文算法由于最长组合路径延迟只需在算法执行时计算一次，其时间复杂度为 $O(|V|^2+|V||E|)$，因而本文算法的运算时间比算法 2 要少。

虽然算法 1 算法复杂度很低，但是算法 1 在面积和关键路径延时方面相比算法 2 和本文算法都要高，因此本文只比较算法 2 和本文算法的算法运算时间。如表 3 所示，本文算法与算法 2 相比，在具体各个电路上平均运算时间降低了 35.4%，此实验结果与上述理论分析相符合。故本文算法在运算速度方面优于算法 2。

表 3 算法运算时间的比较

电路名称	有向图顶点个数	运算时间(s)		本文算法相对算法 2 减少(%)
		算法 2	本文算法	
b14	1254	1.8	1.6	11.1
b14_1	1243	2.1	1.8	14.3
b15	2688	6.7	4.7	29.9
b15_1	2688	6.8	4.7	30.9
b20	2621	3.9	3.6	7.7
b20_1	2613	5.3	2.7	49.1
b21	2675	5.1	2.3	54.9
b21_1	2664	5.4	2.3	57.4
b22	3943	11.2	6.8	39.3
b22_1	3933	18.3	7.4	59.6
平均	2632.2	6.7	3.8	35.4

4.3 电路可靠性分析

由于采用辐照实验和仿真验证的方法对 FPGA 的 TMR 设计进行可靠性分析的成本和时间代价太大，本文采用文献[17]中 SRAM 刷新和 TMR 方法共同作用下的可靠性模型进行系统错误概率分析。

将 TMR 电路中的电路逻辑以表决器为界划分成不同组 E_i，每组后面都接表决器逻辑进行表决，假设每组之间相互独立，同一组电路的三备份逻辑 $E_{i,1}, E_{i,2}, E_{i,3}$ 也相互独立，且三备份逻辑的受单粒子翻转概率相同即 $P(E_{i,1})=P(E_{i,2})=P(E_{i,3})$，并且单粒子造成的错误概率模型为泊松分布：$P$=1−$\exp(-N_i r T_C)$，则当 r 较小时，系统错误概率 E 的计算公式可近似为

$$E = 3T_c r^2 \sum_{i=1}^{M} N_i^2 \tag{3}$$

其中 M 为 TMR 电路的分组数量，N_i 为三备份逻辑其中一个备份逻辑所包含的 SRAM 数量，r 为 FPGA 中特定辐射环境下 SRAM 单元的翻转率，T_C 为 SRAM 刷新的刷新周期。

根据式(3)不难得出，在 SRAM 刷新和 TMR 方法的共同作用下，当对 TMR 电路进行分组的数量固定时，对 TMR 电路的分组大小越均匀，其 E 越小，即系统的可靠性概率也越大；在保证 TMR 电路的分组均匀的前提下，对 TMR 电路进行分组的数量越多，系统可靠性概率越大；在原有 TMR 电路基础之上，对 TMR 电路增加表决器即增加分组数量后，系统的可靠性概率提高。

分析本文算法和其他两组算法对电路可靠性的影响，算法 1 和本文算法都是在寄存模式的节点后面插入表决器，而算法 1 在每个寄存模式的节点后面都插入表决器，因此算法 1 可以认为是在本文算法进行 TMR 设计后电路的基础之上进行的，其通过增加表决器数量形成更多的分组，根据以上分析得到的结论，可知其可靠性相对本文算法要高，但是在面积和时序性能方面相对本文算法要差很多。而比较本文算法和算法 2 实现 TMR 电路的可靠性，虽然两种算法导致 TMR 电路中多数表决器的插入位置不同，但是由于对同一个电路而言反馈环路逻辑是固定的，表决器的插入位置都是在反馈环路逻辑中寄存模式的节点，两种算法插入表决器的相对位置以及对 TMR 电路的分组大小基本相同，因此可以认为本文算法是在算法 2 进行 TMR 设计的基础之上再增加表决器形成的，根据以上分析结论，可以认为采用本文算法进行的 TMR 设计在可靠性方面和算法 2 相比在总体上是相当的。

5 结论

本文首次提出了对映射后网表进行三模冗余设计的方法，同时提出了基于关键路径的网表级 TMR 表决器插入算法对 TMR 设计的时序性能进行优化，并对该算法进行 TMR 设计的可靠性进行了分析。本算法在强连通分量分解的过程中以最长组合路径延迟最小的寄存模式节点作为表决器插入的位置，降低了表决器逻辑对关键路径延时的影响和算法的执行时间。实验结果表明，本文算法与文献[12]中的优化算法相比，在保持电路可靠性的前提下，以少量的面积开销，提高了 TMR 电路的时序性能。本文提出的映射后网表级 TMR 方法和优化算法可为自主研发 FPGA 的可靠性设计提供参考。

参 考 文 献

[1] Carmichael C, Fuller E, Blain P, *et al*.. SEU mitigation techniques for Virtex FPGAs in space applications[C]. Proceedings of the Military and Aerospace Programmable Logic Devices International Conference, Laurel, USA, Sept. 28–30, 1999: 28–30.

[2] Ostler P, Caffrey M, Gibelyou D, *et al*.. SRAM FPGA reliability analysis for harsh radiation environments[J]. *IEEE Transactions on Nuclear Science*, 2009, 56(6): 3519–3526.

[3] Yoshihiro Ichinomiya, Shiro Tanoue, Motoki Amagasaki, *et al*.. Improving the robustness of a softcore processor against SEUs by using TMR and partial reconfiguration[C]. 2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, Kumamoto, Japan, May 2–4, 2010: 47–54.

[4] Morgan K, Caffrey M, Graham P, *et al*.. SEU-induced persistent error propagation in FPGAs[J]. *IEEE Transactions on Nuclear Science*, 2005, 52(6): 2438–2445.

[5] Manuzzato A, Gerardin S, Paccagnella A, *et al*.. Effectiveness of TMR-based techniques to mitigate alpha-induced SEU accumulation in commercial SRAM-based FPGAs[J]. *IEEE Transactions on Nuclear Science*, 2008, 55(4): 1968–1973.

[6] Samudrala P K, Ramos J, and Katkoori S. Selective triple modular redundancy (STMR) based single-event upset (SEU) tolerant synthesis for FPGAs[J]. *IEEE Transactions on Nuclear Science*, 2004, 51(5): 2957–2969.

[7] Siozios K and Soudris D. A methodology for alleviating the performance degradation of TMR solutions[J]. *IEEE Embedded Systems Letters*, 2010, 2(4): 111–114.

[8] Pratt B, Caffrey M, Carroll J F, *et al*.. Fine-grain SEU mitigation for using partial TMR[J]. *IEEE Transactions on Nuclear Science*, 2008, 55(4): 2274–2280.

[9] Sullivan M A, Loomis H H, and Ross A A. Employment of reduced precision redundancy for fault tolerant FPGA applications[C]. 17th IEEE Symposium on Field Programmable Custom Computing Machines, California, USA, April 5–7, 2009: 283–286.

[10] Gavros A, Loomis H, and Ross A. Reduced precision redundancy in a Radix-4 FFT implementation on a field programmable gate array[C]. 2011 IEEE Aerospace Conference, Monterey, USA, Mar. 5–12, 2011: 1–15.

[11] Kastensmidt F L, Sterpone L, Carro L, *et al*.. On the optimal design of triple modular redundancy logic for SRAM-based FPGAs[C]. Proceedings of Design, Automation and Test in Europe, Guaiba, Brazil, 2005, 2: 1290–1295.

[12] Johnson J and Wirthlin M. Voter insertion algorithms for FPGA designs using triple modular redundancy[C]. FPGA'10 Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, USA, 2010: 249–258.

[13] Xilinx TMRTool User Guide[M]. Xilinx Corporation, 2007: 11–23.

[14] BYU-LANL Triple Modular Redundancy Usage Guide[M]. Brigham Young University Configurable Computing Lab, 2009: 8–10.

[15] Cyclone Device Handbook, Volume1[M]. San Jose: Altera Corporation, 2010: 25–32.

[16] 邓俊辉. 数据结构与算法(Java语言描述)[M]. 北京：机械工业出版社，2006: 261–262.
Deng Jun-hui. Data Structure and Algorithms (in Java)[M]. Beijing: Mechanical Industry Press, 2006: 261–262.

[17] Edmonds L D. Analysis of single-event upset rates in triple modular redundancy devices. http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/41123/1/09-6.pdf. 2011.2.

谭宜涛： 男，1984 年生，博士生，研究方向为 FPGA 可靠性 IP 核设计技术.

杨海钢： 男，1960 年生，研究员，博士生导师，研究方向为高速可编程逻辑芯片设计技术和数模混合信号 SOC 设计技术.

黄 娟： 女，1983 年生，博士生，研究方向为集成电路设计自动化技术.