

# FT51:一种容软错误高可靠微控制器

龚 锐 陈 微 刘 芳 戴 葵 王志英

(国防科技大学计算机学院 长沙 410073)

**摘 要** 文中给出一种容软错误高可靠微控制器 FT51. 首先它具有基于异步电路的时空三模冗余结构, 采用此结构可以对时序逻辑单事件翻转(SEU)和组合逻辑单事件瞬态(SET)进行防护. 所有的片内存储器采用 Hamming 编码进行防护. 针对现有控制流检测的不足, 该设计采用了软硬件结合的控制流检测与恢复机制. FT51 在 HJTC 0.25 $\mu\text{m}$  工艺下进行了实现, 与未经加固的版本相比, 其额外的面积开销为 80.6%, 额外的性能开销为 19%~133%. 文中还提出了一种微处理器可靠性评估框架, 在此框架下通过模拟和理论推导证明: 典型情况下 FT51 的故障检出和屏蔽率为 99.73%.

**关键词** 微控制器; 软错误; 单事件翻转; 单事件瞬态; 时空三模冗余; 控制流检测

**中图法分类号** TP332

## FT51: A Soft Error Tolerant High Reliable Micro Controller

GONG Rui CHEN Wei LIU Fang DAI Kui WANG Zhi-Ying

(School of Computer, National University of Defense Technology, Changsha 410073)

**Abstract** This paper proposes a soft error tolerant high reliable micro controller, FT51. Temporal spatial triple modular redundancy based on asynchronous circuit technique is designed to tolerate SEU (Single Event Upset) in sequential logic and SET (Single Event Transient) in combinational logic. On-chip memory is protected by Hamming code. Control flow checking and recovering by compiler signature and hardware checking is designed to reduce performance overhead and binary code size overhead in traditional software implemented control flow checking. A reliability evaluation method is also proposed in this paper. FT51 is implemented in HJTC 0.25 $\mu\text{m}$  process, with 80.6% area overhead and 19%~133% performance overhead. The results of simulation and theoretical verification indicate that FT51 can mask or detect 99.73% faults under typical condition.

**Keywords** micro controller; soft error; single event upset; single event transient; temporal spatial triple modular redundancy; control flow checking

## 1 引 言

随着空间技术和集成电路技术的高速发展, 微

处理器已广泛应用于航空航天领域. 在宇宙辐射环境中存在着大量的包括电子、质子、光子、 $\alpha$  粒子和重离子在内的高能粒子<sup>[1]</sup>. 过去曾经认为由于高能粒子在穿越大气层时能量迅速减弱, 将不会引起地

收稿日期: 2007-05-07; 修改稿收到日期: 2007-08-01. 本课题得到国家自然科学基金(90407022)资助. 龚 锐, 男, 1980 年生, 博士研究生, 研究方向为高可靠微处理器设计、异步集成电路设计. E-mail: iamlaogong@gmail.com. 陈 微, 女, 1982 年生, 博士研究生, 主要研究方向为计算机体系结构、高可靠微处理器设计. 刘 芳, 女, 1976 年生, 博士, 研究方向为高可靠微处理器设计、信息安全. 戴 葵, 男, 1968 年生, 博士, 副教授, 主要研究方向为微处理器设计、高性能计算机体系结构、异步集成电路设计. 王志英, 男, 1956 年生, 教授, 博士生导师, 研究领域为微处理器设计、高性能计算机体系结构和异步集成电路设计.

面上的集成电路发生功能错误. 但是随着集成电路特征尺寸的急剧减小, 单芯片中的节点迅速增加, 并且由于供电电压的降低, 节点的电量也随之降低. 所以较低能量的粒子也有可能影响集成电路的正常运行<sup>[2]</sup>. 因此, 不仅在空间环境中, 甚至在地面上的集成电路也面临着高能粒子的威胁. 应用于航空航天领域的微处理器必须考虑高能粒子效应, 以提高系统的可靠性.

高能粒子效应可分为永久效应和瞬时效应<sup>[3]</sup>. 永久效应是由于高能粒子陷入芯片内部硅氧界面引起的, 一旦发生永久效应芯片就宣告报废, 不能使用. 永久效应只有长期暴露于高辐射环境才有可能发生. 高能粒子引起的瞬时效应主要包括单事件门锁 (Single Event Latch, SEL)、单事件翻转 (Single Event Upset, SEU) 和单事件瞬态 (Single Event Transient, SET). SEL 的产生是由于高能粒子引起 CMOS 工艺中的寄生三极管发生偏置, 通过断电重启可消除 SEL. 当高能粒子轰击存储器或触发器等时序逻辑电路时, 将引发 SEU, 时序逻辑电路的值将发生翻转, 错误的值将保持到下一个值写入. SEU 发生的频率很高, 根据文献<sup>[4]</sup>的数据, 商用 MIPS R3000 微处理器在 500 公里地球近地轨道上大约每分钟发生一次 SEU. 高能粒子轰击组合逻辑电路, 将发生 SET, 产生一个宽 0.35 ns~1.3 ns 的毛刺<sup>[5]</sup>, 并且有可能沿组合逻辑通路传递. 如果毛刺恰好被时序逻辑采样到, 将会导致集成电路功能错误. 随着时钟频率的提高和两级触发器之间组合逻辑深度的减少, SET 毛刺被采样到的概率大大增加<sup>[6]</sup>. 由高能粒子瞬时效应引起的错误都可以被消除, 因此也被称为软错误 (soft error). 由于 SEU 和 SET 发生频率远高于 SEL, 应用于专门领域的微处理器进行容错设计主要考虑容 SEU 和容 SET.

本文基于 R80515 体系结构, 设计实现了容软错误的高可靠微控制器 FT51. 它是对寄存器和存储器采用不同的方法进行防护. 提出的基于异步电路技术的时空三模冗余, 在传统三模冗余结构可比拟的面积延迟开销下, 可有效地屏蔽 SEU 和 SET, 加固寄存器. 本文对片内存储器采用了纠一检二的 Hamming 编码进行防护. 由于 8051 系列微控制器多用于控制领域, 运行的代码具有大量分支跳转指令, 软错误容易引起控制流失效. 针对传统编译实现的控制流检测性能、代码量开销大的情况, 本文提出了软硬件结合的控制流检测与恢复技术, 有效地控制了开销, 并且在检测到控制流失效后能快速

恢复, 提高了系统的可用性. FT51 在 HJTC 0.25  $\mu\text{m}$  工艺下实现, 采用商用的设计流程和 EDA 工具, 未使用任何全定制单元. 本文还给出对 FT51 进行错误注入和代码扰动的模拟结果. 在模拟结果的基础上, 根据 FT51 体系结构和防护策略特点, 提出了一套评估模型对其容错特性进行了理论评估.

本文第 2 节、第 3 节分别介绍相关工作和本文的研究方法; 第 4 节具体介绍 FT51 中采用的防护策略和技术; 第 5 节给出 VLSI 实现; 第 6 节给出模拟结果和理论评估模型; 第 7 节与已有工作进行比较; 第 8 节总结本文的工作并指出将来的工作.

## 2 相关工作

容软错误的防护技术一般被分为软件技术和硬件技术两种. 从设计方法学的角度看, 微处理器的设计自顶向下包括如下几个层次: 编译级、体系结构级、门级、电路级和工艺级, 每一级都为上一级提供实现基础. 我们将相关的防护技术按照实现层次进行了进一步的细分.

工艺级防护采用抗辐照技术, 即在集成电路生产过程中采用特殊制造工艺, 以防止高能粒子效应. IBM 提出在工艺级采用绝缘体上硅 (SOI) 技术以实现抗辐照. SOI 工艺非常昂贵且目前国内没有商用生产线.

电路级防护技术是指采用新颖电路结构, 实现容 SEU 或容 SET 能力. 在电路级实现的容 SEU 技术包括 Heavy Ion Tolerant (HIT) 单元<sup>[7]</sup> 和 Dual Interlock Storage Cell (DICE) 单元<sup>[8]</sup> 等. 这些技术对传统的 D 触发器进行电路级改进, 使其具有容 SEU 的能力. 只须将寄存器用上述单元进行替换, 就能有效地防护 SEU. 在电路级也开发了容 SET 的技术. 文献<sup>[9]</sup>中提出了 SET 过滤器, 将其插入时序逻辑的输入端, 可以有效地过滤组合逻辑的输出毛刺. 上述电路级容软错误单元都需要进行全定制设计, 这使得芯片的设计流程有别于商用的自顶向下的半定制设计流程, 不利于开发和实现.

门级防护技术是指在门级引入空间或时间冗余, 达到容软错误的目的. 三模冗余 (Triple Modular Redundancy, TMR) 是传统的门级容 SEU 结构. 该结构对时序逻辑单元进行空间三模冗余, 并通过表决器表决进行三选二输出. Gaisler 采用 TMR 实现了基于 SPARK V8 体系结构的航天级微处理器 LEON-FT<sup>[10]</sup>. 文献<sup>[11]</sup>提出了复用扫描链寄存器

进行 SEU 屏蔽,在非扫描模式时将扫描链寄存器作为正常寄存器的镜像,并采用 C 单元<sup>[12]</sup>屏蔽可能发生的 SEU.文献[13]在空间三模冗余的基础上加上时间冗余,对冗余的寄存器采用不同的时钟进行控制,只要三路时钟的间隔大于 SET 毛刺宽度,最多只有一个冗余寄存器采样到毛刺,通过表决输出以后就能有效屏蔽错误.表决输出的结果被第四个时钟控制的寄存器锁存并作为下一级组合逻辑的输入.如果第四个时钟控制的寄存器发生 SEU,将引发不可屏蔽的错误.门级防护引入大量的空间冗余,对微处理器的芯片面积有较大的影响.

体系结构级防护是指在模块级引入冗余或错误检测等机制,以屏蔽或检测软错误.Compaq 的高可靠服务器 Himalaya 和 IBM 的 Z900<sup>[14]</sup>采用了双处理器同时运行代码并比较的结构.文献[15]提出了一种通用可配置的双核结构,在安全模式下,双核被配置成 master/checker 模式运行相同的程序,并对运行结果进行检测.体系结构级防护还包括存储器防护,一般采用检错纠错码(EDAC)检测和纠正存储器中发生的 SEU.常用的 EDAC 包括奇偶校验码和 Hamming 码<sup>[16]</sup>.

编译级防护是指在编译时自动加入冗余信息或检测指令.近年来,在同时多线程(SMT)体系结构下开发出了很多基于时间冗余的容错机制,包括 AR-SMT<sup>[17]</sup>、SRT<sup>[18]</sup>、SRTR<sup>[19]</sup>和 SlipStream<sup>[20]</sup>,这些容错机制都需要编译的支持.编译器还能自动插入冗余指令和数据,硬件执行后再由检测指令进行比较,若检测到错误则重新执行该指令<sup>[21-23]</sup>.

需要注意的是,有些防护技术,如控制流检测,可以在不同的设计层次上实现.软错误可能引起程序发生错误的跳转,导致运行轨迹发生混乱,即控制流失效.控制流检测实时监测程序的运行轨迹并与预期轨迹进行比较,可以有效地防止由于控制流失效导致的系统崩溃.控制流检测可以在体系结构级实现,即采用片内的看门狗(watchdog)模块<sup>[24]</sup>或单独的看门狗处理器<sup>[25]</sup>对程序存储器的地址总线进行监测.控制流检测也可以在编译级实现.文献[26]提出了纯编译实现的控制流检测 CFCSS.文献[27]利用 PowerPC 处理器提供的分支踪迹异常(BTE)机制在 PowerPC 体系结构上实现了软件控制流检测.体系结构级实现的控制流检测不能应用于有 cache 的微处理器中,而编译级实现的控制流检测增大了二进制代码量,并带来大量性能开销.

### 3 研究方法

根据法国 TIMA 实验室利用 THESIC 系统<sup>[28]</sup>对未经加固的 8051 微控制器进行辐照测试的结果<sup>[1]</sup>可以看出,内部存储器和特殊功能寄存器(SFR)是 8051 微控制器中的关键单元,如果对这些单元进行容软错误加固,将屏蔽 97.94%的错误.而根据文献[29]的结论,超过一半的软错误将引起控制流失效,特别是对于 8051 这种以控制为主要应用的微控制器,其程序中有大量的分支跳转指令,将有高达 70%的软错误导致控制流失效.基于上述文献的结论,我们对 R80515 中所有的寄存器和内部存储器进行了防护,并且实现了控制流检测以获得更全面有效的防护效果.

在设计具体防护技术时,我们针对已有方法的不足,设计了新的寄存器防护技术和控制流检测恢复技术,并且注意了其可实现性.超大规模集成电路设计需要良好的设计方法学保证.我们采用的防护技术都不需要进行全定制,使用成熟的商用设计流程和 EDA 软件就可实现.

容软错误策略的有效性可以通过模拟、理论评估或辐射照射实验的方法进行验证.这三种方法是等效且可以互相验证的.本文首先采用模拟的方法,进行错误注入和代码扰动,以检验防护的有效性.然后提出理论评估模型,在模拟结果的基础上进行理论推导.已有的评估模型都是对未经加固的微处理器进行评估以分析出其易于发生软错误的关键单元.微处理器发生软错误的概率正比于其受到辐射的面积.而任何加固技术都可能增加芯片的面积.本文提出的评估模型将芯片面积作为重要的参数,可以更精确地评估各项加固技术对微处理器可靠性的影响.在 FT51 流片生产以后,还将进行辐射照射实验,与模拟和理论评估的结果进行对比,进一步验证其可靠性.

### 4 防护策略

#### 4.1 基于异步电路技术的时空三模冗余<sup>[30]</sup>

对寄存器的 SEU 防护一般采用传统的 TMR 技术.由于 TMR 中冗余寄存器采用相同的时钟进行控制,有可能三个冗余的寄存器都采样到 SET 毛刺,从而引发不可屏蔽的错误.一种简单的解决办法是采用时空三模冗余(Temporal Spatial Triple

Modular Redundancy, TSTMR)<sup>[10]</sup>, 其结构如图 1 所示. 通过分离三路冗余单元的时钟, 并在时钟上加延迟单元, 达到错开冗余单元采样时机的目的. 只要保证时钟的间隔  $d$  大于毛刺宽度, 即使毛刺被其中的一个冗余单元采样到, 其他的两个冗余单元仍将采样到正确的结果.

TSTMR 结构只能用于线性流水线. 如果用于带反馈回路的流水线, 将引发功能错误. 图 2(a) 所示反馈回路, 每个周期将其输出取反, 并送回输入端. 从图 2(b) 所示的波形图可知, 复位后所有的寄存器都为 0,  $b$  路寄存器在第一个时钟沿锁存到新的数据, 表决器的输出变高, 导致  $c$  路寄存器采样到 0, 不同于  $a, b$  两路寄存器. 而在第二个时钟周期, 表决器输出的值先为 0 后为 1, 发生功能错误.

借鉴异步电路解同步流水线<sup>[31]</sup>中显式分离主从锁存器的电路结构, 本文提出了一种双时钟触发寄存器 (Dual Clock Triggered Register, DCTREG), 其电路结构如图 3(a) 所示. 通用的  $D$  触发器也是由主从锁存器构成, 其主从锁存器采用统一的时钟进行同步控制. DCTREG 中主从锁存器采用一种异步的方式进行控制. 时钟信号 CLK1 的相位比 CLK2 超前, 在 CLK1 的上升沿将输入  $D$  采样并保持在中间节点  $Q_m$ , 在 CLK2 的上升沿将  $Q_m$  输出到  $Q$  端. 需要注意的是, 从锁存器的控制信号是 CLK1 和 CLK2 的与. 如果只采用 CLK2 信号控制从锁存器, 在 CLK1 的下降沿到 CLK2 的下降沿之间, 两个锁存器都将处于透明状态,  $D, Q$  端直接连通, 这是不允许的.

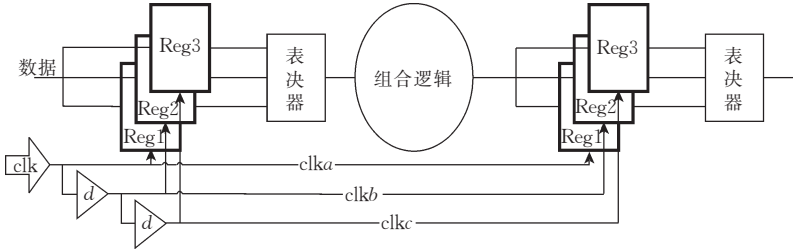


图 1 TSTMR 结构

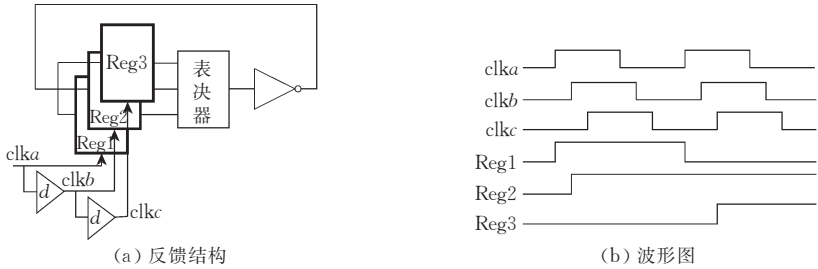


图 2 带反馈回路的结构

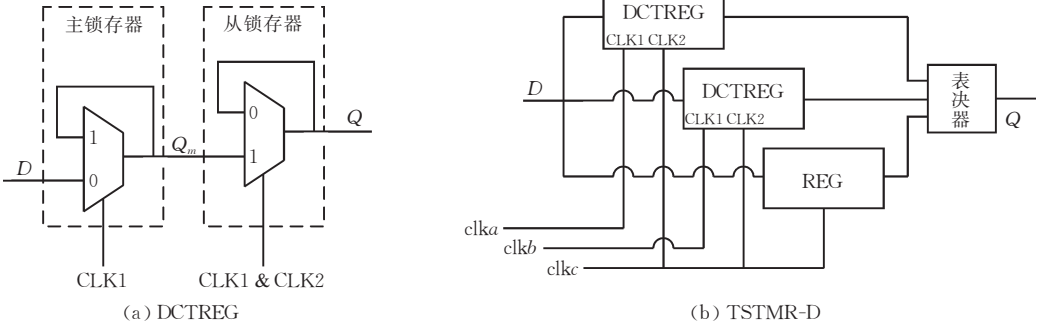


图 3 DCTREG 与 TSTMR-D 结构

采用 DCTREG 的 TSTMR 结构 (TSTMR-D) 如图 3(b) 所示. 其中  $a$  路和  $b$  路寄存器采用 DCTREG 实现,  $c$  路寄存器仍采用普通的  $D$  触发器. 两个 DCTREG 分别在  $a$  路时钟和  $b$  路时钟的上升沿

采样组合逻辑输出, 直到  $c$  路时钟的上升沿之后才将其送到表决器的输入端. 三个冗余单元都在  $c$  路时钟沿上输出, 解决了文献[10]提出的传统 TSTMR 无法运用于带反馈回路流水线的问题. 因此,

TSTM-RD 可以广泛运用于各种类型的流水线结构,在合理的面积和延迟开销下,全面提高芯片的容 SET 和容 SEU 能力.与文献[13]的结构相比,TSTM-RD 取消位于关键路径上的额外寄存器,防止了由于该寄存器发生 SEU 引发的不可屏蔽错误.FT51 中的所有寄存器都采用 TSTM-RD 进行防护.

#### 4.2 存储单元防护

FT51 中所有片内存储单元,包括 2KB 的指令 ROM,256B 的内部数据 RAM 等都采用了可以纠一检二(SEC-DED)的 Hamming 编码进行防护.该编码方式可以纠正一位错误,检出两位错误.在数据写入时,先通过编码模块进行编码,将 8 位数据编码为 12 位,再写入存储单元.读出时进行解码,如果检测到错误,就进入中断处理例程.若为可以纠正的一位错误,则由中断处理例程负责将正确的值重新写入存储单元.若为不可纠正的多位错误,则直接复位芯片.

#### 4.3 软硬件结合的控制流检测与恢复

现有的硬件实现的控制流检测,采用看门狗实时监控程序存储器的地址总线,以获得程序跳转等控制流信息并进行检测.对于有 cache 的结构,若取指时指令 cache 命中,就不会读程序存储器,所以采用看门狗进行控制流检测不适用于有 cache 的结构.而编译实现的控制流检测由编译器在跳转目的地址处插入额外的检测指令,引入大量的性能开销.更重要的是,由编译实现的控制流检测将增加程序的二进制代码量.这就意味着在嵌入式系统中将需要更大的片内程序存储器以容纳程序代码,这将增加芯片受辐射的面积,也就增大芯片发生软错误的概率.传统的控制流检测发现失效后一般直接对芯片进行复位,这对于一些不能停机的高可用系统来说是不可接受的.针对以上的不足,本文结合 8051 体系结构特点,提出软硬件结合的控制流检测(Control Flow Checking by Compiler Signature and Hardware Checking, CFCCH),由编译在程序中插入签名数据而非检测指令,由硬件自动进行检测,并且提供控制流恢复机制(CFCCH-R),发生失效以后可以恢复到正确的控制流继续执行.

首先定义程序流图为有向图  $CFG=(V,E)$ ,其中  $V=\{v|v \text{ 为基本块}\}$ ,  $E=\{\langle v_i, v_j \rangle | \text{存在从 } v_i \text{ 到 } v_j \text{ 的分支或跳转}\}$ .对于某个特定的基本块  $v_j$ ,赋予其唯一的签名值  $S_j$ ,并存储于基本块  $v_j$  的块头.如果  $\exists \langle v_i, v_j \rangle \in E$ ,则  $v_i$  到  $v_j$  的签名距离  $d_j = S_i \oplus S_j$ ,同样将  $d_j$  也存储在基本块  $v_j$  的块头.当程序执行从

$v_i$  到  $v_j$  的控制流转移时,计算运行时签名值  $s_j = S_i \oplus d_j$ .如果分支或转移正确,则  $s_j = S_i \oplus d_j = S_i \oplus (S_i \oplus S_j) = S_j$ .如果  $s_j \neq S_j$ ,则表明发生了控制流错误.

由于控制流图中存在多扇入多扇出节点的情况,仅有签名值和签名距离是不够的.如图 4(a)所示的 CFG 子图中,假设存储于  $v_3$  节点的签名距离由  $v_1$  节点确定,即  $d_3 = S_1 \oplus S_3$ ,则从  $v_2$  到  $v_3$  的正确控制流转移将被判断为非法.解决这种情况的办法是对每一个基本块  $v_i$  引入运行时调整签名  $D_i$ ,当发生控制流转移时,计算运行时签名值  $s_j = S_i \oplus D_i \oplus d_j$ .如图 4(b)所示,  $v_1$  节点运行时调整签名  $D_1=0$ ,并由  $v_1$  节点确定  $v_3$  节点的签名距离  $d_3 = S_1 \oplus S_3$ .赋予  $v_2$  节点唯一的签名值  $S_2 \neq S_1$ ,由于  $S_3 = S_2 \oplus D_2 \oplus d_3 = S_2 \oplus D_2 \oplus (S_1 \oplus S_3)$ ,所以  $D_2 = S_1 \oplus S_2$ .引入运行时调整签名后,解决了正常控制流转移被判断为非法的问题,同时也引入了混淆.  $v_4$  节点的签名距离由  $v_2$  节点唯一确定,即  $d_4 = S_2 \oplus D_2 \oplus S_4 = S_1 \oplus S_4$ ,所以由  $v_1$  到  $v_4$  节点的非正常控制流转移将被判断为合法.纯编译实现的控制流检测解决混淆的办法是从  $v_i$  进入单扇入节点  $v_s$  时只计算  $s_s = S_i \oplus d_s$ ,而进入多扇入节点  $v_m$  时计算  $s_m = S_i \oplus D_i \oplus d_m$ .这样图 4(b)中的  $d_4 = S_2 \oplus S_4$ ,如果发生从  $v_1$  到  $v_4$  的非法转移也能被检测到.本文采用的是硬件检测,处理器从  $v_i$  节点进入  $v_j$  节点时无法判断  $v_j$  是否为多扇入节点.所以硬件进行检测时,统一计算  $s_j = S_i \oplus D_i \oplus d_j$ .这样可能发生混淆的情况,但是这只发生于  $v_1$  恰好跳转到  $v_4$  基本块的块头,其概率是很小的.

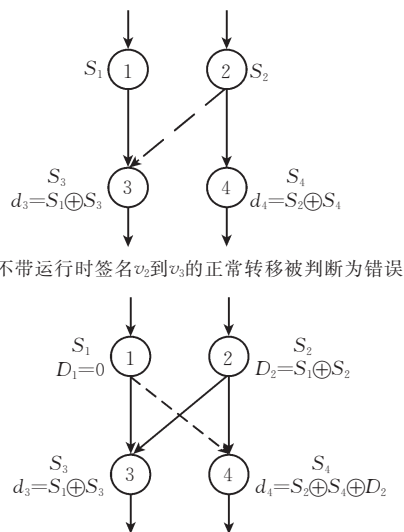


图 4 程序流图



编译进行签名的具体算法如下:

```
assign a unique signature value  $S_i$  to every node  $v_i$ 
insert  $S_i$  in the head of  $v_i$ 
for every node  $v_i$  in CFG do
    if  $v_i$  is NOT a multi-fan-in node then
        select  $\{v_j\} = pred(v_i)$ 
        calculate signature distance of  $v_i$ ,  $d_i = S_j \oplus S_i$ 
        insert  $d_i$  before  $S_i$  in the head of  $v_i$ 
    else
        select  $v_j \in pred(v_i)$ , calculate  $d_i = S_j \oplus S_i$ 
        insert  $d_i$  before  $S_i$  in the head of  $v_i$ 
        for every node  $v_k \in pred(v_i)$  do
            if  $v_k \neq v_j$  then
                calculate  $D_k = S_k \oplus S_j$ 
                inset  $D_k$  after  $S_k$  in the head of  $v_k$ 
            else
                Insert  $D_j = 0$  after  $S_j$  in the head of  $v_j$ 
        endif
    endif
endfor
```

CFCCH 由硬件进行控制流检测,所以编译只需在源代码中插入三个字节的签名数据,而没有检测语句,这样可以大大减小目标代码的体积.为了实现硬件检测,增加了两个特殊寄存器  $Sreg$  和  $Dreg$ ,分别记录当前基本块的签名值和运行时调整签名.执行完分支/跳转指令后,立即触发检测,从程序存储器中取出签名数据,进行检测.检测共需 3 个时钟周期,每个时钟周期的工作如表 1 所示.

表 1 硬件检测操作

周期	操作
1	读程序存储器,得 $d_i$ ,更新 $Sreg = Sreg \oplus Dreg \oplus d_i$ .
2	读程序存储器,得 $S_i$ ,与 $Sreg$ 比较,若不同,生成 error 信号.
3	读程序存储器,得 $D_i$ ,更新 $Dreg = D_i$ .

特别需要注意的是中断的处理.处理器发生中断后,转移到中断处理程序进行中断处理.由主程序到中断处理程序的控制流转移不在编译的预计之内.解决的办法是发生中断后由软件或硬件将  $Sreg$  和  $Dreg$  压栈,然后将其置为 0,这样可以进行中断处理程序子控制流图的检测.而在退出中断处理程序时进行弹栈,重新恢复主程序的控制流图上下文.

控制流失效是由软错误引起的,只需恢复正常的控制流再继续执行就可以消除.恢复正常的控制流需要进行现场保存和恢复.本文在 CFCCH 的基础上加上了现场保存和恢复的机制(CFCCH-R).处理器运行的现场是指处理器内部所有和运行状态相关的存储单元,包括特殊寄存器,通用寄存器文件和

片内数据 RAM.本文针对 8051 体系结构中特殊功能寄存器 SFR 和内部数据 RAM 的不同特点,分别采用冗余和写缓冲机制进行现场保存和恢复.将所有的 SFR 进行冗余备份,分为运行组和备份组.复位时两组都复位为相同的值.处理器首先在运行组上运行,在确定没有发生控制流错误的现场保存点,将运行组的数据写入备份组 SFR.检测到控制流错误以后,根据备份组数据恢复运行组现场. SFR 的保存和恢复都只需要一个时钟周期.

8051 体系结构中还有 256B 的内部数据 RAM,本文引入写缓冲对其进行现场保存.在写内部 RAM 时,只写入写缓冲.因此写缓冲包含了被修改的现场.运行至现场保存点时才将写缓冲的数据写入内部 RAM,保存现场的时间开销依赖于写缓冲中数据的项数.而在发生错误时,只需花费一个时钟周期作废写缓冲中的数据,就能恢复现场.需要注意的是,写缓冲也采用 4.2 小节所示的 Hamming 编码进行保护.

要能进行正确的恢复,必须能够保存正确的现场.在进入每个基本块,进行硬件控制流检测无误后进行现场保存.此时程序位于正确的控制流轨迹上.当一个基本块中发生多次对内部 RAM 的写操作导致写缓冲写满后,如果再继续写将导致写缓冲溢出.此时需要触发一次现场保存,将写缓冲中的数据写入内部 RAM.检测到控制流轨迹混乱以后,可以自动回退到上一基本块块头或上一基本块内的某一条写内部 RAM 指令之后.

由以上的分析可知,写缓冲的大小对于 FT51 的性能有直接的影响.本文对几种典型应用程序在不同写缓冲大小下的性能开销做了比较.具体数据如图 5 所示.图 5 采用了归一化的性能比较,纵轴数据为不同程序在 CFCCH-R 结构下的运行时间与在无控制流检测机制的 R80515 结构下的运行时间之比.从图 5 中可以看出,写缓冲入口太少,将频繁发生由写缓冲满触发的现场保存,极大地影响程序的运行性能.而写缓冲大到一定的程度以后,将能容纳

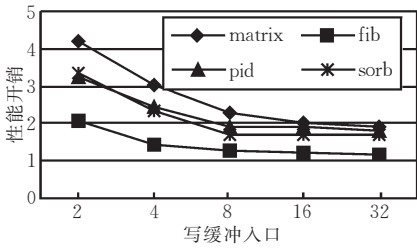


图 5 写缓冲大小对程序性能的影响

绝大多数基本块内的写数据,所以对性能的加速逐渐降低,并且过大的写缓冲将增加芯片被辐射面积,增大发生软错误的概率.综合考虑,FT51 中采用了 8 个入口的写缓冲.

与文献[26]提出的 CFCSS 相比,本文提出的 CFCCH-R 只需插入 3 个字节的签名数据而非检测指令,可以减小代码体积.进行一次硬件检测只需额外增加 3 个时钟周期,可以提高代码的运行速度. CFCSS 中,如果发生控制流失效,跳转到基本块内部,仍然可以正常执行至下一检测指令.而在发生控制流失效到失效被检测到这段时间内,可能已经对片外发出了错误的控制信号或写入了错误的计算结果,这对于高可靠实时控制系统来说是不可接受的.而 CFCCH-R 在译码阶段检测到分支或跳转指令后置上标志位,在其执行之后立即触发硬件检测,可以保证检测的实时性.一般的控制流检测并未考虑检测到失效后的进一步措施,很多只是简单的复位系统. CFCCH-R 在检测到控制流失效后不用复位系统就可以快速恢复正确的控制流,可以有效地保证芯片的可用性.

## 5 VLSI 实现

本文在 HJTC 0.25 $\mu$ m CMOS 工艺下实现了 FT51,该微控制器的时钟频率可达 100MHz,其版图如图 6 所示,带 pad 的芯片总面积为 2.1mm $\times$ 2.1mm. FT51 兼容所有的 Intel8051 指令集,并且提供了软硬件中断接口、串行通信端口和独立的乘法单元.指令的执行周期为 2~6 个时钟周期,采用流水的方式实现了取指和执行的并行.实现时首先采用 Verilog 在寄存器传输级(RTL)对 FT51 进行描述,其中 DCTREG 描述为两个锁存器(latch). TSTMTR-D 中时钟树所需要的延迟单元用标准单元

库中的 buffer 实现,并且调整三路时钟延迟  $d$  为 2ns. 由于没有采用特殊的定制单元,FT51 采用了通用的自顶向下的设计流程,所使用的 EDA 工具都是成熟的商用工具,有效地加速了设计过程.

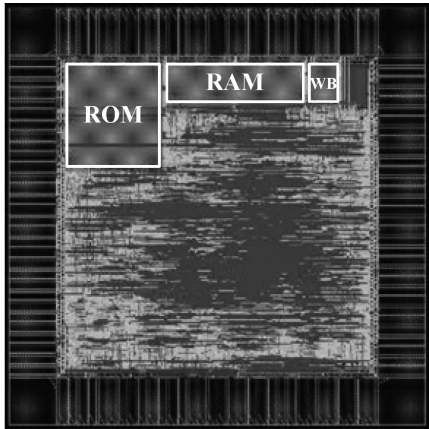


图 6 FT51 版图

表 2 比较了不同的防护策略对芯片面积的影响.表中只比较了综合后得到的标准单元面积和由 Memory Compiler 得到的存储器面积,而非整个芯片的面积.因为整个芯片面积还包含布线损耗,此损耗与后端设计人员的设计经验有关,故而标准单元面积和存储器面积能更好地体现各种防护策略带来的面积开销.从表 2 中可以看出,由于对寄存器进行了时空三模冗余,TSTMTR-D 结构带来了 36.3% 的标准单元面积增加.在此基础上增加控制流检测与恢复,对已经三模的 SFR 再进行冗余备份,并且增加了控制流硬件检测、写缓冲控制和现场自动备份等逻辑功能,使得标准单元面积增加了 91.2%,而增加的写缓冲使得存储器面积增加了 14.4%.对片内存储单元进行 Hamming 编码防护,进一步增加了存储单元面积,并且由于增加了编解码电路,使得标准单元的面积也有少量增加,因此 FT51 的总面积比未经加固的版本增加 80.6%.

表 2 面积开销

	Cell 面积/mm <sup>2</sup>	Cell 面积增加/%	Mem 面积/mm <sup>2</sup>	Mem 面积增加/%	总面积/mm <sup>2</sup>	总面积增加/%
未加固	0.977		0.397		1.374	
TSTMTR-D	1.332	+36.3	0.397	+0	1.729	+25.8
TSTMTR-D+CFCCH-R	1.868	+91.2	0.454	+14.4	2.322	+69.0
TSTMTR-D+CFCCH-R+Hamming	1.869	+91.3	0.612	+54.2	2.481	+80.6

FT51 与未经加固的版本相比,其二进制代码量和性能的开销主要由控制流检测与恢复引起.图 7 表示了针对不同测试程序,未经加固的版本(original)、纯编译实现的控制流检测(CFCSS<sup>[26]</sup>)、软硬件结合的控制流检测(CFCCH)和带恢复功能的

CFCCH (CFCCH-R) 4 个版本的二进制代码量和性能的比较.图 7 中的数据进行了归一化处理,以未做任何软硬件加固的 original 版本为基准进行比较. CFCSS 插入的签名代码,编译为 8051 指令后占 11 个字节. CFCCH 和 CFCCH-R 二进制代码量相

同,由于只插入了 3 个字节签名数据,其额外的代码开销约在 11%~27%之间,远低于纯编译实现的 CFCSS. CFCSS 的签名检测代码执行一次需要 13 个周期,性能开销较大. CFCCH 在每个检测点增加了额

外的 3 个时钟周期,带来了 9%~76%的性能开销,低于 CFCSS. 由于有额外的现场保存点的存在, CFCCH-R 的性能开销略大于 CFCCH,为 19%~133%,仍低于 CFCSS.

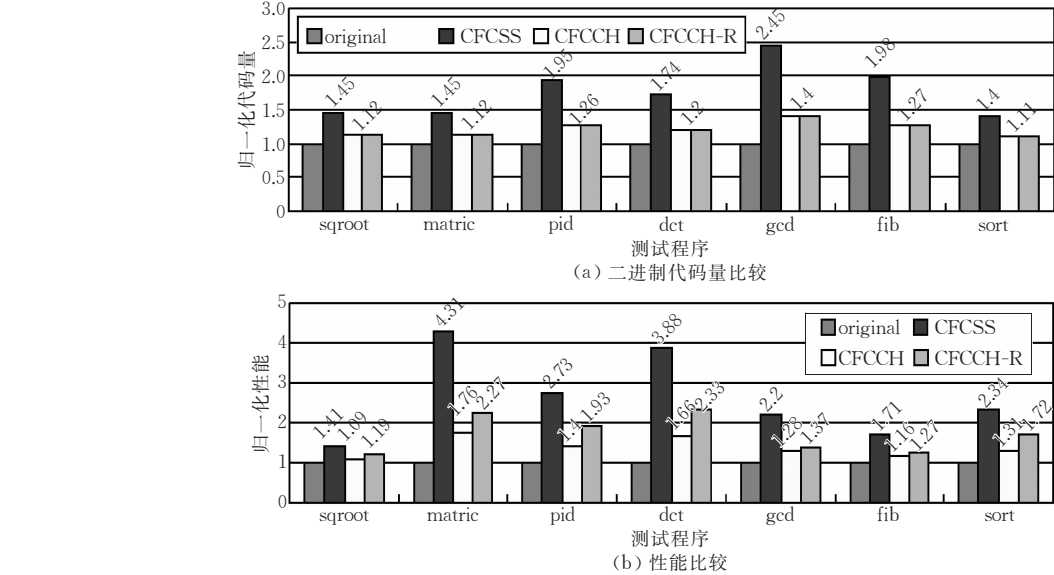


图 7 二进制代码量与性能比较

6 模拟与建模

6.1 模拟实验与结果

本文对 FT51 进行了故障模拟. 对 TSTMR-D 的容 SEU 和容 SET 能力,采用了故障注入的方式进行模拟. 为便于错误检测,只对 PC 寄存器进行了故障注入. 检测时只需将程序的运行轨迹与预期轨迹比较,就能检测到故障. 采用了图 8 所示结构对寄存器进行了故障注入. 通过异步复位端(AR)随机复位寄存器的值来模拟 SEU. 测试时对 PC 的第一位注入 10000 次 SEU. 而对组合逻辑 SET 通过在图 8 中所示的 sel 信号端随机加 10000 次 1ns 宽的毛刺来模拟. 模拟的两次粒子轰击的时间间隔服从均值

$\mu$  为  $10\mu\text{s}$ ,方差  $\sigma^2$  为 1 的高斯分布. 模拟的 SEE 发生概率远大于实际的大约每分钟一次的概率. 这是为了在短时间内测试出尽可能多的不可屏蔽故障.

对只采用了 TMR 和采用了 TSTMR-D 的 FT51 在不同的时钟频率下注入故障,得到的不可屏蔽的错误数目如图 9 所示. 由图 9(a)可知,时钟频率越低,由 SEU 引发的不可屏蔽错误就越多. 这是因为时钟周期越长,在一个时钟周期内同一位的多个冗余单元同时发生 SEU 的概率就越大. 在相同频率下, TMR 和 TSTMR-D 由 SEU 引起的不同频率错

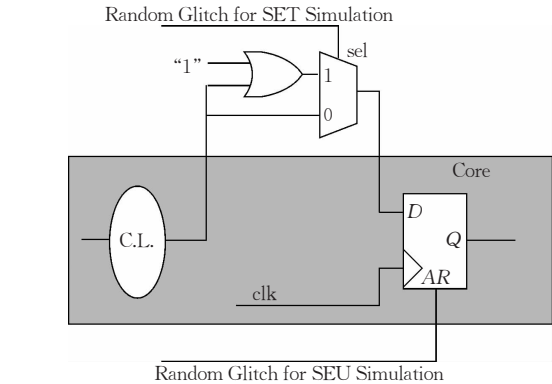
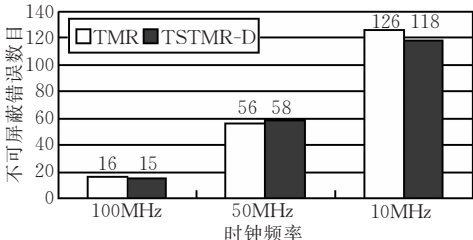
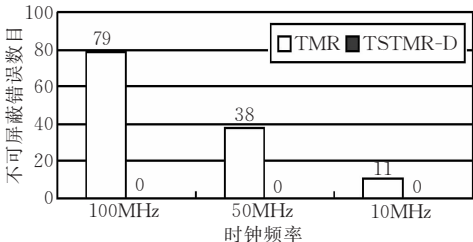


图 8 错误注入结构



(a) SEU引起不可屏蔽错误



(b) SET引起不可屏蔽错误

图 9 错误注入结果



误数目大致相当. 由图 9(b)可知, 在组合逻辑注入 SET 错误的情况下, TSTMR-D 没有发生不可屏蔽错误. 这是由其分时采样的结构所决定的. 时钟频率越高, TMR 对组合逻辑输出的采样越频繁, 采样到 SET 毛刺并发生错误的概率也就越高. TMR 是通用的有效的容 SEU 结构, 并已在实际的航天应用中得到验证. 该故障注入结果表明, TSTMR-D 与 TMR 相比有相同的容 SEU 能力, 并且由于采用分离时钟的方式, 该结构对 SET 也具有很好的防护能力.

对控制流检测和 Hamming 编码的有效性, 本文采用通用的代码扰动的方法进行测试. 对控制程序 pid 进行了扰乱, 将其二进制代码随机翻转. 表 3 比较了不同的故障注入数目下发生不可检测的错误数. 从表 3 中可以看出, 在故障注入数目较多的情况下, 单独使用 Hamming 编码和单独使用 CFCCH-R 方法, 都不能很好地检测. 而将这两种方法结合起来, 就能达到较好的防护效果.

表 3 代码扰动结果

翻转位数	Hamming	CFCCH-R	Hamming+CFCCH-R
100	2	1	0
500	14	12	1
1000	32	26	3

## 6.2 理论模型与评估

故障屏蔽或检出率受辐射剂量、体系结构、加固策略和程序形态的影响, 没有统一通用的理论分析框架. 本文结合 FT51 体系结构和加固策略, 并以 6.1 小节的模拟结果为基础, 对该微控制器的故障屏蔽检出率做了理论建模和分析.

首先定义故障(fault)、错误(error)和失效(failure). 故障是产生错误和失效的似然条件和推理上的原因, 本文定义由于受到高能粒子轰击发生 SEU 和 SET 为故障. 如果由于故障没有被有效屏蔽而导致系统中某部分发生了非正常的行为或状态, 称为错误, 由 SEE 引起的错误也称为软错误. 如果错误没有被有效检测或屏蔽, 导致系统偏离其预期设计的要求或规定的功能, 称为失效.

定义一块芯片在一段时间内发生的故障数目为  $No_{\text{fault}}$ , 其中发生在片内存储单元的故障数目为  $No_m$ , 而发生在内核中组合逻辑和寄存器的故障数目为  $No_c = No_{\text{fault}} - No_m$ . 由于芯片发生 SEE, 即故障的概率正比于其面积, 所以有

$$No_m = No_{\text{fault}} \cdot A_m / (A_m + A_c) \quad (1)$$

$$No_c = No_{\text{fault}} \cdot A_c / (A_m + A_c) \quad (2)$$

其中  $A_m, A_c$  分别为片内存储器和内核所占面积. 分别定义片内存储器和内核对故障的不可屏蔽率为  $UM_m$  和  $UM_c$ , 则由片内存储器和内核故障引起的软错误数目  $No_{\text{se}}$  为

$$No_{\text{se}} = No_m \cdot UM_m + No_c \cdot UM_c \quad (3)$$

这些软错误可能引起控制流失效、运算失效, 也有可能不引起失效. 其中控制流失效是指程序运行轨迹发生混乱, 而运算失效不影响程序的正常运行, 只是对运算结果造成影响. 在有推断执行(speculation)的体系结构中, 如果在推断执行阶段发生软错误, 其后该段推断执行的指令被作废, 就不会引起失效. 定义由软错误引起控制流失效的概率为  $P_{\text{cfc}}$ , 引起运算失效的概率为  $P_{\text{com}}$ , 不引起失效的概率为  $P_{\text{crt}}$ . 有

$$P_{\text{cfc}} + P_{\text{com}} + P_{\text{crt}} = 1 \quad (4)$$

定义控制流检测机制对控制流失效的不可检出率为  $UM_{\text{cfc}}$ , 运算错误检测机制(如指令重执行与检测)对运算失效的不可检出率为  $UM_{\text{com}}$ . 因此, 总的失效数目  $No_{\text{failure}}$  为

$$No_{\text{failure}} = No_{\text{se}} \cdot (P_{\text{cfc}} \cdot UM_{\text{cfc}} + P_{\text{com}} \cdot UM_{\text{com}}) \quad (5)$$

因此整个微处理器对故障的不可屏蔽率  $UM$  为

$$UM = \frac{No_{\text{failure}}}{No_{\text{fault}}} \quad (6)$$

带入式(1)~式(3), 式(5), 有

$$UM = \left( \frac{A_m}{A_m + A_c} UM_m + \frac{A_c}{A_m + A_c} UM_c \right) \cdot (P_{\text{cfc}} UM_{\text{cfc}} + P_{\text{com}} UM_{\text{com}}) \quad (7)$$

具体对 FT51 进行分析, 由 6.1 小节的数据可知

$$A_m / (A_m + A_c) = 24.7\%,$$

$$A_c / (A_m + A_c) = 75.3\%.$$

根据 6.1 小节的模拟结果, 在正常工作频率 100MHz 下对 TSTMR-D 注入 10000 次 SEU 和 10000 次 SET, 共发生不可屏蔽的错误 15 次, 故

$$UM_c = 15/20000 = 0.75\%.$$

对二进制代码进行 1000 次扰动, Hamming 编码未检出错 32 个, 故

$$UM_m = 32/1000 = 3.2\%.$$

只采用 CFCCH-R 时, 未检出失效为 26 个, 故

$$UM_{\text{cfc}} = 26/1000 = 2.6\%.$$

由于 FT51 中没有采用运算失效检测机制, 故

$$UM_{\text{com}} = 1.$$

为了求得  $UM$  的最高上限, 取  $P_{\text{crt}}$  为 0, 即软错误或者导致控制流失效, 或者导致运算失效. 有

$$P_{\text{com}} = 1 - P_{\text{cfc}}.$$

将以上数据带入式(7),得不可屏蔽率  $UM$  随  $P_{\text{cfc}}$  的变化如图 10 所示. 由于 FT51 没有对运算错误进行防护,当  $P_{\text{cfc}}$  较低时,不可屏蔽的运算失效较多, $UM$  最大,为 0.85%. 实际的基于 8051 体系结构的应用多为控制型,具有大量的分支跳转指令,软错误容易引起控制流失效. 当  $P_{\text{cfc}}$  为 100% 时, $UM$  最小,为 0.02%. 根据文献[29]的实际辐射实验结论,8051 体系结构下典型的  $P_{\text{cfc}}$  为 70%,此时  $UM$  为 0.27%.

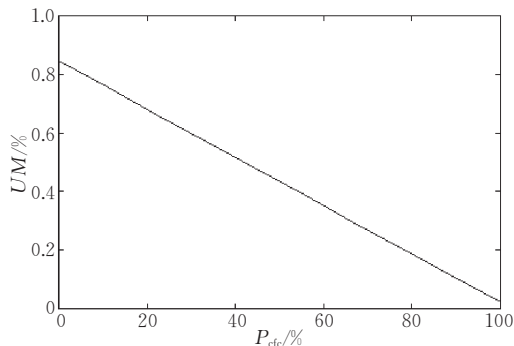


图 10 发生控制流失效的概率与不可屏蔽率的关系

需要注意的是,各种防护方法的屏蔽率受辐射注入量、程序形态等的影响. 本文采用的数值为模拟结果,在模拟时采用的故障注入频率远高于实际频率. 理论分析采用的数值都是取其上限,且假设不可屏蔽 SEE 总是引起错误( $P_{\text{crt}}=0$ ),所以得到的  $UM$  为不可屏蔽率的上限. 也即在典型情况下( $P_{\text{cfc}}$  为 70% 时),FT51 至少能屏蔽或检测 99.73% 以上的 SEE.

## 7 与已有工作的比较

文献[1]对 MSC8051 中的内部数据存储器采用 Hamming 编码进行防护,将其综合到 FPGA 上进行了辐射测试. 该款微控制器没有考虑对寄存器进行 SET 和 SEU 防护. 文献[32]对文献[1]的结构进行了改进,对内部数据存储器 and 寄存器都采用了 Hamming 编码进行防护. 该款微控制器同样没有考虑 SET 防护,并且由于 Hamming 编解码模块位于流水线的每一级,将会影响主频的提高,带来很大的性能开销. 文献[33]对 8 位微控制器 M68HC11 进行了容错加固,采用 TMR 屏蔽 SEU,将组合逻辑输出延迟,采用 Code Word State Persevering(CWSP)单元同步延迟信号与原有信号,屏蔽 SET. 该款微控制器没有考虑对存储单元的防护. 文献[10]设计实现了基于 SPARK V8 体系结构的 LEON-FT,采

用的防护策略包括 TMR 和 EDAC 编码. 该款微处理器也没有考虑对 SET 的防护. 文献[34]在 130nm SOI 工艺下实现了第五代 SPARK64 微处理器,该款微处理器对存储单元采用 ECC 进行保护,所有位于数据通路的寄存器都采用奇偶校验. 该款微处理器的防护策略比较简单,主要依靠在工艺级采用 SOI 工艺进行防护. 文献[35]对流处理器 Merrimac 进行了容错加固,采用了 Hamming 编码、程序重执行、指令重复等防护策略. 由于流处理器主要用于计算密集型应用、跳转或分支指令较少,故软错误主要引起运算失效而非控制流失效,所以该款微处理器没有控制流检测机制.

## 8 结论与将来的工作

本文给出的 HJTC 0.25 $\mu\text{m}$  下设计实现的容软错误高可靠微控制器 FT51,采用了基于异步电路技术的时空三模冗余 TSTMR-D,对寄存器进行 SEU 和 SET 防护. 对片内存储器采用了纠一检二的 Hamming 编码. 本文还给出了针对现有控制流检测的不足,设计实现的软硬件实现的控制流检测与恢复方法 CFCCH-R. FT51 采用成熟的商用 EDA 工具和流程在标准单元库上的实现,其面积比未经加固的版本增加 80.6%,针对不同的测试程序,增加性能开销为 19%~133%. 本文给出了采用故障注入和二进制代码扰动对采用的三种防护技术的防护能力的模拟结果. 并在模拟的基础上进行了理论建模和评估,得出典型情况下 FT51 对故障的屏蔽检测率下限为 99.73%.

本文只对 FT51 的容错能力进行了模拟和理论推导,流片后得到实际的芯片,需要进一步做辐射照射实验. 在实际的辐射照射实验中,常常使用大剂量的粒子注入以加速实验过程. 采用大剂量注入后,发生多位翻转(Multiple Bit Upset, MBU)的概率大大增加. 如果在冗余的 TSTMR-D 寄存器或是存储器中一个字发生 MBU,将导致大量的不可屏蔽错误. 在得到实际的辐射照射实验结果以后,可以和模拟以及理论推导结果相比较以互相验证.

## 参 考 文 献

- [1] de Lima F G et al. Designing a radiation hardened 8051-like micro-controller//Proceedings of the 13th Symposium on Integrated Circuits and Systems Design. Manaus, Brazil, 2000: 255-260

- [2] Normand E. Single event upsets at ground level. *IEEE Transactions on Nuclear Science*, 1996, 43(6): 2742-2750
- [3] Ma T, Dressendorfer P. *Ionizing Radiation Effects in MOS Devices and Circuits*. New York: Wiley-Interscience, 1989.
- [4] Kaschmitter J L et al. Operation of commercial R3000 processors in the low earth orbit (LEO) space environment. *IEEE Transactions on Nuclear Science*, 1991, 38(6): 1415-1428
- [5] Eaton P et al. Single event transient pulsewidth measurements using a variable temporal latch technique. *IEEE Transactions on Nuclear Science*, 2004, 51(6): 3365-3368
- [6] Shivakumar P et al. Modeling the effect of technology trends on the soft error rate of combinational logic//*Proceedings of the International Conference on Dependable Systems Networks*. Bethesda, US, 2002: 389-398
- [7] Bessot D, Velazco R. Design of SEU-hardened CMOS memory cells: The HIT cell//*Proceedings of the 2nd European Conference on Radiation and its Effects on Components and Systems*. Saint-Malo, France, 1993: 563-570
- [8] Calin T, Nicolaidis M, Velazco R. Upset hardened memory design for submicron CMOS technology. *IEEE Transactions on Nuclear Science*, 1996, 43(6): 2874-2878
- [9] Mongkolkachit P, Bhuva B. Design technique for mitigation of alpha-particle-induced single-event transients in combinational logic. *IEEE Transactions on Device and Materials Reliability*, 2003, 3(3): 89-92
- [10] Gaisler J. A portable and fault-tolerant microprocessor based on the SPARC V8 architecture//*Proceedings of the International Conference on Dependable Systems and Networks*. Bethesda, US, 2002: 409-415
- [11] Mitra S, Seifert N, Zhang M, Shi Q, Kim K S. Robust system design with built-in soft-error resilience. *Computer*, 2005, 38(2): 43-52
- [12] Muller D, Bartky W. A theory of asynchronous circuits//*Proceedings of the International Symposium on the Theory of Switching*. Cambridge MA, 1959: 204-243
- [13] Mavis D G, Eaton P H. Soft error rate mitigation techniques for modern microcircuits//*Proceedings of the International Reliability Physics Symposium*. Dallas TX, 2002: 216-225
- [14] Slegel T J et al. IBM's S/390 G5 microprocessor design. *IEEE Micro*, 1999, 19(2): 12-23
- [15] Kottke T, Steininger A. A reconfigurable generic dual-core architecture//*Proceedings of the International Conference on Dependable Systems and Networks*. Philadelphia PA, 2006: 45-54
- [16] Kohavi Z. *Switching and Finite Automata Theory*. New York: McGraw-Hill, 1970
- [17] Rotenberg E. AR-SMT: A microarchitectural approach to fault tolerance in microprocessors//*Proceedings of the International Symposium on Fault-Tolerant Computing*. Madison, WI, 1999: 84-91
- [18] Mukherjee S S, Kontz M, Reinhardt S K. Detailed design and evaluation of redundant multithreading alternatives//*Proceedings of the 29th Annual International Symposium on Computer Architecture*. Anchorage AK, 2002: 99-110
- [19] Vijaykumar T, Pomeranz I, Cheng K. Transient-fault recovery via simultaneous multithreading//*Proceedings of the 29th Annual International Symposium on Computer Architecture*. Anchorage AK, 2002: 87-98
- [20] Sundaramoorthy K, Purser Z, Rotenburg E. Slipstream processors: Improving both performance and fault tolerance//*Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*. Cambridge MA, 2000: 257-268
- [21] Nicolescu B, Velazco R, Sonza-Reorda M, Rebaudengo M, Violante M. A software fault tolerance method for safety-critical systems: Effectiveness and drawbacks//*Proceedings of the 15th Symposium on Integrated Circuits and Systems Design*. Porto Allegre, Brazil, 2002: 101
- [22] Oh N, Mitra S, McCluskey E J. Ed4i: Error detection by diverse data and duplicated instructions. *IEEE Transactions on Computer*, 2002, 51(2): 180-199
- [23] Reis G A, Chang J, Vachharajani N, Rangan R, August D I. SWIFT: Software implemented fault tolerance//*Proceedings of the International Symposium on Code Generation and Optimization*. San Jose, CA, 2005: 243-254
- [24] Arora S et al. Secure embedded processing through hardware-assisted run-time monitoring//*Proceedings of the Design Automation and Test in Europe Conference and Exposition*. Munich, Germany, 2005: 178-183
- [25] Mahmood A, McCluskey E J. Concurrent error detection using watchdog processors- a survey. *IEEE Transactions on Computer*, 1988, 37(2): 160-174
- [26] Oh N, Shirvani P, McCluskey E J. Control flow checking by software signatures. *IEEE Transactions on Reliability*, 2002, 51(2): 111-122
- [27] Fazeli M, Farivar R, Miremadi S G. A software-based concurrent error detection technique for PowerPC processor-based embedded systems//*Proceedings of the 20th International Symposium on Defect and Fault Tolerance in VLSI Systems*. Monterey, CA, 2005: 266-274
- [28] Velazco R, Rezgui S, Cheynet P, Bofill A, Ecoffet R. THE-SIC: A testbed suitable for the qualification of integrated circuits devoted to operate in harsh environment//*Proceedings of the IEEE European Test Workshop*. Barcelona, Spain, 1998: 89-90
- [29] Czech E W, Siewiorek D. Effects of transient gate-level faults on program behavior//*Proceedings of the 20th International Symposium on Fault-Tolerant Computing*. Newcastle, UK, 1990: 236-243
- [30] Gong Rui, Chen Wei, Liu Fang, Dai Kui, Wang Zhi-Ying. Modified triple modular redundancy structure based on asynchronous circuit technique//*Proceedings of the 21st International Symposium on Defect and Fault Tolerance in VLSI Systems*. Arlington, Washington DC, 2006: 184-196
- [31] Blunno I et al. Handshake protocols for de-synchronization//*Proceedings of the International Symposium on Advanced Research in Asynchronous Circuits and Systems*. Crete, Greece, 2004: 149-158
- [32] Cota Érika et al. Synthesis of an 8051-like micro-controller tolerant to transient faults. *Journal of Electronic Testing: Theory and Applications*, 2001, 17(2): 149-161

- [33] Bastos R P, Kastensmidt F L, Reis R. Design at high level of a robust 8-bit microprocessor to soft errors by using only standard gates//Proceedings of the 19th Annual Symposium on Integrated Circuits and Systems Design. Ouro Preto, Brazil, 2006: 196-201
- [34] Ando Hisashige et al. A 1.3GHz fifth generation SPARC64

microprocessor//Proceedings of the 40th Conference on Design Automation. Anaheim, CA, 2003: 702-705

- [35] Erez M et al. Fault tolerance techniques for the merrimac streaming supercomputer//Proceedings of the ACM/IEEE Conference on Supercomputing. Seattle WA, 2005: 29



**GONG Rui**, born in 1980, Ph. D. candidate. His research interests include high reliable microprocessor design and asynchronous integrated circuit design.

**CHEN Wei**, born in 1982, Ph. D. candidate. Her research interests include computer architecture and high reliable microprocessor design.

**LIU Fang**, born in 1976, Ph. D. . Her research interests

include high reliable microprocessor design and information security.

**DAI Kui**, born in 1968, Ph. D. , associate professor. His research interests include microprocessor design, high performance computer architecture and asynchronous integrated circuit design.

**WANG Zhi-Ying**, born in 1956, Ph. D. , professor. His research interests include microprocessor design, high performance computer architecture and asynchronous integrated circuit design.

## Background

Microprocessors now are widely used in space environment, which consists of various high-energy particles. The particle-induced soft errors threaten reliability of integrated circuits and systems in space. Moreover, the number of nodes in a chip keeps increasing with shrinking feature size, and the charge stored in a node keeps decreasing with lower supply voltage. As a result, low-energy particles that once were considered negligible now are able to affect the operation of microprocessors.

An ideal soft error tolerant microprocessor for space applications should fulfill the following requirements; (I) tolerating frequent faults, including Single Event Upset (SEU) and Single Event Transient (SET); (II) implementing with reasonable area and performance overheads; (III) supporting fast recovery from errors; (IV) implementing in commercial design flow and manufacture process. Since Intel 8051 series micro controllers are widely used in many space applications, the authors designed and implemented a soft error tolerant 8051, FT51. The temporal spatial triple modular redundancy based on asynchronous circuit technique is proposed to protect registers against both SEU and SET, with reasonable chip area overhead comparing to conventional methods. The

control flow checking and recovering by compiler signatures and hardware checking is also designed to implement fast recovery from control flow errors. All hardening techniques in the authors' design can be implemented in commercial design flow. A theoretical evaluation approach is presented, in which the chip area is used as an important parameter to evaluate the microprocessor's reliability after hardening.

This work is supported by the National Natural Science Foundation of China under grant No. 90407022. The NSF project aims to solve the design methodology of asynchronous circuit and explore the fault tolerant ability of asynchronous circuit. The research work of this paper belongs to the fault tolerant ability of asynchronous circuit. As is presented, the register is protected by temporal spatial triple modular redundancy based on asynchronous circuit. In this field, the authors have published two papers. One is "Modified triple modular redundancy structure based on asynchronous circuit technique" in DFT'06, the other is "A new approach to single event effect tolerance based on asynchronous circuit technique" in Journal of Electronic Testing: Theory and Applications by Springer (accepted).