软件错误检测与纠正技术可靠性研究

刘小汇 伍 微 欧 钢
(国防科技大学电子科学与工程学院卫星导航研发中心，湖南 长沙 410073)

摘 要：基于信息冗余的错误检测与纠正(Error Detection and Correction，EDAC)技术是常见的系统级抗单粒子翻转(Single Event Upsets，SEU)的容错方法，软件实现的 EDAC 技术是硬件 EDAC 技术的替代方案，通过软件编程，在现有存储段上增加具有纠错功能的编码(Error-correcting Codes，ECC)来实现存储区错误的检测和纠正。分析了软件 EDAC 方案中，纠错编码的纠错能力及编码效率、刷新间隔、需保护代码量等因素对可靠性的影响，分析和仿真实验结果表明，对于单个粒子引起的存储器随机错误，提高单个码字的纠错能力及编码效率、增大刷新间隔对可靠性的影响不大，而通过缩短任务执行的代码量来提高刷新间隔，以及压缩需保护代码的总量，对可靠性有较大改进。分析结论能够指导工程实践中，在实现资源、实时性、可靠性之间进行优化选择。

关键词：单粒子翻转；可靠性；容错；错误检测与纠正
中图分类号：TN432 **文献标识码**：A **文章编号**：1003-0530(2011)08-1140-07

Study of the Reliability for Software-Implemented EDAC Technology

LIU Xiao-hui WU Wei OU Gang
(Satellite Navigation R&D Center, School of Electronic Science and Engineering, National University of Defense Technology, Changsha, Hunan 410073, China)

Abstract: EDAC(Error Detection and Correction) based information redundancy is a well-known system level fault-tolerance technique for SEU(Single Event Upset) in space applications. Software-implemented EDAC technique is a substitute for hardware-implemented EDAC. The encoding and checking program is added to detect and correct memory errors through accommodate extra ECC(Error-correcting Codes). The reliability of software-implemented EDAC would be analyzed for capability and the code rate of error-correcting code, the scrubbing interval and the number of program words protected. Simulation and analysis are presented to the random error of a single-bit upset, that the capability and code rate of the codeword, the scrubbing interval can be increased without appreciably affecting reliability, however reducing the number of program size protected would be obviously improve the reliability. It can provide important reference for application among the memory resource, real-time performance and reliability of choice.

Key words: Single Event Upset; Reliability; Fault-tolerance; Error Detection and Correction

1 引言

空间单粒子翻转(Single Event Upset，SEU)效应导致的存储器瞬时故障对卫星系统而言是不可忽视的，即使单个粒子造成的某个逻辑错误，对整个卫星系统都有可能是灾难性的，最终造成卫星运行异常或者失控[1][2]。通常情况下可以通过避错设计和容错设计提高存储模块的可靠性。避错设计一般是从工艺上提高器件的可靠性，如选用宇航级器件，或者对存储模块进行屏蔽、电路优化设计以减少外部辐射干扰等[3]；容错设计是通过增加冗余资源，实现故障的检测和纠正，从而屏蔽故障的影响。错误检测与纠正(Error Detection and Correction，EDAC)技术是目前卫星上通用的系统级容错设计技术，可以通过增加具有差错纠正和检测能力的硬件器件来实现[4][5]，也可以通过软件编程在现有存储段上增加具有纠错功能的代码来实现[6]。相对于硬件 EDAC 技术，软件 EDAC 技术是实现成本更低的一种容错技术，利用软件代码来实现存储器中随机错误的纠正和检测。目前对存储器的抗 SEU 研究，多集中在处理器片外 SRAM 区中[7][8]，软件 EDAC 技术，不仅能对片外 SRAM，同时也能够对片内 RAM 运行的程序代码进行检纠错，这是它有别于硬件 EDAC 的一个亮点。在美国的高级研究和全球观测卫星(Advance Research and Global Observations Satellite，ARGOS)项目的空间实验中，就使用了软件

收稿日期：2011 年 5 月 2 日；修回日期：2011 年 7 月 22 日

EDAC 技术进行代码的检纠错[6],其可靠性虽然比硬件方案低,但其实现灵活,对资源的占用量小,不需要额外增加硬件器件,在一定条件下,其不失为硬件 EDAC 的替代方案。

对于软件 EDAC 的实现,设计者要在资源占用率、实时性、可靠性之间进行优化选择,如何在满足资源占用率及实时性的基础上,选择可靠性最高的 EDAC 容错方法,是设计者需要解决的问题。本文首先描述了软件 EDAC 的实现原理,然后从编码方案选择、刷新间隔设定以及需保护代码量的大小等方面,分析了软件 EDAC 设计方案中各个因素对系统可靠性的影响,同时通过故障注入模拟实验证明了分析的正确性,最后给出了软件 EDAC 实现时对可靠性有益的结论,为软件 EDAC 技术的空间应用提供参考。

2　实现原理

如图 1 所示是随机可访问存储器(Random access memory RAM)的结构图。

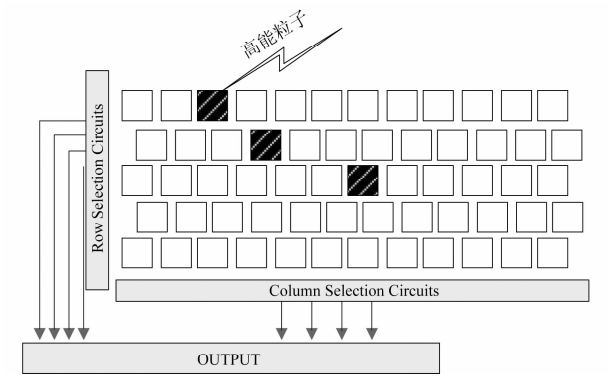

图 1　存储器的结构图

Fig. 1　Random access memory structure

外部处理器通过行选和列选电路对存储器中的一个或者多个单元进行访问,宇宙中的高能粒子横向入射存储器的节点,使得电荷在节点产生沉积,当沉积的电荷比临界电荷大时,存储器的状态就会发生逆转,当这种逆转是可恢复的,就称之为软错误。图中深色部分就是发生状态翻转的存储单元。在空间卫星系统中,处理器上电初始化完毕后,程序代码从片外非易失存储器(如 PROM)中加载到程序存储区(RAM)运行。相对于数据,程序代码在运行过程中是不变的,因此可以利用这个特点,将这些不变的代码作为信息位,首先离线进行纠错编码,增加校验位,然后将校验位作为程序区的一部分也加载至程序存储区。在程序正常运行时,EDAC 任务作为其中的一个任务周期性调用,对程序区的代码进行检纠错,当检测到可以纠正的错误时,将正确的内容覆盖掉原来地址的错误内容。考虑到 EDAC 代码也在程序存储区中作为一个特殊的任务执行,也有单粒子翻转的可能性,为了保证它的正常运行,一种方案是在程序存储区同时存放它的另一个副本,两段 EDAC 代码交错运行,相互将对方进行刷新[6];或者将 EDAC 代码移至片外,由专门的硬件对它进行检纠错,处理器在片外运行 EDAC 代码。

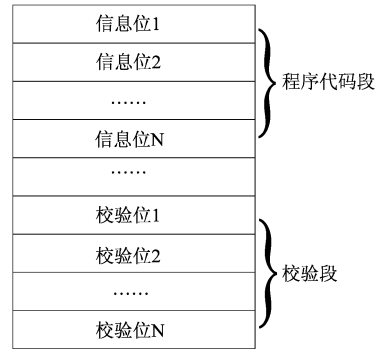

图 2　软件 EDAC 的存储区组织形式

Fig. 2　Memory organization of the software-implemented EDAC

3　可靠性分析

以下进行软件实现 EDAC 的可靠性分析,假设需保护的软件运行于多任务的环境中,且系统内各个任务按照某种调度准则进行调度,EDAC 刷新程序作为其中的一种任务被操作系统调度执行,整个软件存在三种状态:运行(run)、等待(idle)、刷新(scrub)。运行状态是程序代码被读取和执行,软件进行正常的任务处理;等待状态是没有任务运行,系统处于空闲等待期;刷新状态是 EDAC 程序在执行需保护程序区的字节错误检测(检错)和重写(纠错),可以看做运行状态下的一种特殊任务。刷新间隔是执行两次代码刷新任务的时间间隔。为了分析系统的可靠性,定义以下变量如表 1 所示。

假设单粒子翻转率的典型值是 $1.38 \cdot 10/\text{b-s}^{-11}$,同时,软件 EDAC 的纠错编码采用(72,64)的 Hamming 码,信息位为 64 比特,连续分布在程序区中,校验位 8 比特由计算机事前计算出来,存储在程序区的专门一块校验区域中,这块区域存放受保护程序段的所有校验位。假设程序存储区总的大小为 0.5 Mbyte,受保护代码占用了 0.3 Mbyte(39321 个 64 bit),"运行"状态下每次执行的代码量为受保护代码量的 10%。对于 EDAC 代码,即"刷新"任务下执行的代码,根据前文所述,可以存放相同的拷贝于程序区中交错进行相互刷新,也可以放至片外由专门的硬件对其进行检纠错,本文讨论前一种情况,即其在程序存储区中占用 *M1*(本文取值 228)个码字,对于后种情况,在假设专用硬件检纠错可靠性为 100% 的情况下,可以将参数 *M1* 设置为

零,本文分析的公式将同样适用。

表 1 变量定义及典型取值

Tab. 1 Parameter definitions and typical values

变量名	描述	本文取值
λ	单粒子翻转率(单位: upset/bit-second),由于空间环境的影响,单位时间内单个存储位发生单粒子翻转的概率	$1.38\cdot10^{-11}$
m	单个码字的长度(单位: bit),包括信息位和校验位	72
n	信息位长度(单位: bit),即一个码字中信息位的长度	64
Tr	"运行"任务的最大执行时间(单位: s)	0.42
Ti	"等待"任务的最大执行时间(单位: s)	0.58
Ts	"刷新"任务的最大执行时间(单位: s)	0.01
M	系统中原需要保护代码的总码字数(无量纲),即 n 比特码字的个数	39321
$M1$	增加的"刷新"任务所占代码段的总码字数(无量纲),即 n 比特码字的个数	228
S	每次运行的代码量占总受保护代码量的百分比(无量纲)	10%

同时,为了方便分析,对模型进行简化处理,列出以下的基本假设:

a) 各个存储位的翻转事件是相互独立的,即一个存储位的翻转不会对其他位造成影响[9]-[11],由单个粒子造成的多位翻转(Multiple Bit Upset, MBU)不在本文讨论范围;

b) 单粒子产生的概率服从泊松分布[9]-[11],即在时间 t 内单个存储位的可靠性(不发生翻转)为 $p=e^{-\lambda t}$;

c) 系统在"运行"状态下,"运行"段代码任何一位存储器的翻转均会造成系统的失效;

d) 系统在"刷新"状态下,"刷新"段代码任何一位存储器的翻转均会造成系统的失效;

e) 系统在"等待"状态下,一个编码的码字中一位或几位错误能够被纠正,不会造成系统失效;

f) 本文分析仅针对程序代码区的可靠性,假设 cache、程序指针寄存器、数据区的可靠性是 100%。

首先分析没有 EDAC 保护方案时程序系统的可靠性,当不做 EDAC 加固处理时,假设程序区中任意一位存储器的翻转均能够导致整个系统的失效。定义在一个时间段 T 内,单个存储位发生单粒子翻转的概率为 p_T:

$$p_T=(1-e^{-\lambda T})\cong\lambda T \quad (\lambda T<<1) \tag{1}$$

则无 EDAC 保护方案时系统的抗单粒子翻转可靠性为:

$$P_{r\_noEDAC}=(1-p_T)^{nM}\cong1-\lambda TnM \quad (\lambda TnM<<1) \tag{2}$$

当采用软件 EDAC 代码保护方案后,假设 T 时间段内的一个码字中(m 比特),发生 i 个比特单粒子翻转事件的概率为:

$$P_{(i,m)}=C_m^i {p_T}^i(1-p_T)^{m-i}$$

同时由(1)式可知在"运行""等待""刷新"任务执行时间的单个存储位发生单粒子翻转的概率分别为 $p_{Tr}\cong\lambda Tr$, $p_{Ti}\cong\lambda Ti$, $p_{Ts}\cong\lambda Ts$,假设"运行"任务每次执行的代码量是被保护代码量的 $S(S=10\%)$ 倍,则当单个码字能够纠 r 比特错误时,系统在"运行"、"等待"、"刷新"任务执行一次时的抗单粒子翻转可靠性为:

$$P_{r\_EDAC}=(1-p_{Ts})^{nM1}(1-p_{Tr})^{nMS}\Big[\sum_{j=0}^{r}C_m^j {p_{Ti}}^j(1-p_{Ti})^{m-j}\Big]^{(M1+M)} \tag{3}$$

图 3 表示了在各参数取典型值的情况下,有无 EDAC 方案时,程序系统的可靠性。其中有 EDAC 保护方案时,采用了纠一检二(Single-Error-Correcting, Double-Error-Detecting, SEC-DED)的纠错编码。由图可见,当没有 EDAC 保护方案时,经过 48 小时运行后,系统的可靠性降为 0 左右,系统已明显不可用,而增加了 EDAC 代码保护方案后,系统的可靠性有了明显的提升,经过 48 小时运行后,其可靠性仍然在 80% 左右。

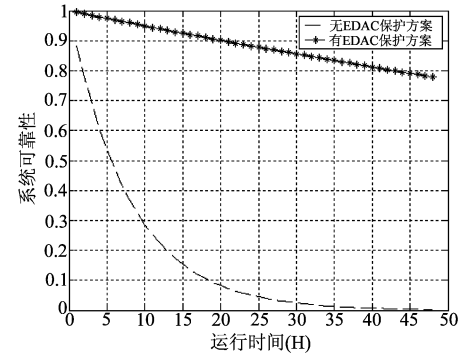

图 3 有无 EDAC 保护方案时的系统可靠性比较

Fig. 3 Reliability comparison of EDAC and no EDAC

3.1 纠错编码对可靠性的影响分析

软件 EDAC 实现的基础是采用纠错编码进行纠检错,因此纠错编码的选择及其性能对系统的可靠性也起了一定的影响。通常使用的纠错编码有 RS 码、Hamming 码、BCH 码、交织码、循环汉明码等[12],它们各自具有不同的检纠错能力,纠错编码的选择,就是在译码效率、码率和最小汉明距离(检纠错性能)之间进行折中。RS 码是一种编码效率和纠错性能都很高的特殊多进制 BCH 码[13],它以符号为单位,具有纠单字检双字的功能(Single-byte-error-correcting, double-byte-error-detecting, SbEC-DbED),但是由于纠错能力

强,译码过程较复杂,若考虑到系统的实时性,此类码未必是最好的选择。Hamming 码、交织码等的优点是译码较为简单,占用的资源较小,译码实时性较强,在实际工程中使用得较普遍,但是其编码效率和纠检错性能不如 RS 码高。以下分析不同性能的纠错编码对系统可靠性的影响。

3.1.1　检纠错性能对可靠性的影响分析

假设采用能纠一位错的纠错编码时,即一个码字中可以出现小于等于一个随机错误,由(3)式得系统执行一次“运行”、“等待”、“刷新”后的可靠性为:

$$P_{r1}=(1-p_{Ts})^{nM1}(1-p_{Tr})^{nMS}[(1-p_{Ti})^{m}+mp_{Ti}(1-p_{Ti})^{(m-1)}]^{(M+M1)} \tag{4}$$

当 $p_{Ts}nM1<<1$、$p_{Tr}nMS<<1$、$p_{Ti}m<<1$ 时,上式可化简为:

$$P_{r1}=(1-p_{Ts}nM1)(1-p_{Tr}nMS)[1-m(m-1)(M1+M)p_{Ti}^{2}] \tag{5}$$

当采用纠二位错的纠错编码时,即一个码字中允许出现小于等于二个随机错误,则系统执行一次“运行”、“等待”、“刷新”后的可靠性为:

$$P_{r2}=(1-p_{Ts})^{nM1}(1-p_{Tr})^{nMS}[(1-p_{Ti})^{m}+mp_{Ti}(1-p_{Ti})^{(m-1)}+m(m-1)p_{Ti}^{2}(1-p_{Ti})^{(m-2)}/2]^{(M+M1)} \tag{6}$$

同理上式可化简为:

$$P_{r2}=(1-p_{Ts}nM1)(1-p_{Tr}nMS)\{1-\frac{(M1+M)m(m-1)p_{Ti}^{2}}{2}[1+(m-2)p_{Ti}]\} \tag{7}$$

为了比较方便,构造能够纠一位错误的(78,64)Hamming 码,同时从(127,113)循环码中选择所有前 49 位为零的码字构成能纠二位错误的(78,64)缩短循环码。下图为在其余参数均相同的情况下,采用(78,64)的 Hamming 码与采用(78,64)的缩短循环码的系统可靠性的比较图。由图可见,在辐射剂量较低的空间环境中(单粒子翻转率较低),且存储器容量较小(Mbyte 量级以下)的情况下,单个码字纠错性能的提升对可靠性的增长效果不明显。

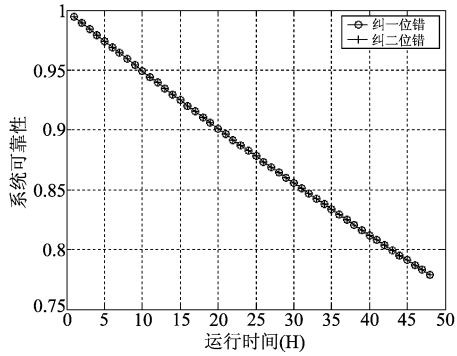

图 4　两种编码方法下的系统可靠性比较

Fig. 4　Reliability comparison of two coding methods

3.1.2　编码效率对可靠性的影响分析

编码效率定义为 $\eta=n/m$,即信息元位数 n 在码字长度 m 中所占的比重,编码的效率决定了 EDAC 方案中,加入纠错编码后实际保护代码的膨胀程度。对于线性码,其纠错能力 t 与校验位长度 r 具有如下关系:

$$2^{r}-1\geq\sum_{i=1}^{t}C_{m}^{i}$$

即能够纠 t 位错误的线性码,其校验位长度 r 和码字长度可以有多种取值的组合,也就对应了有多个编码效率。以下讨论在纠错能力 t 固定的情况下,不同编码效率对系统可靠性的影响。假设系统采用能纠一位错($t=1$)的 Hamming 码,由于

$$\ln(1-x)=-(x+\frac{x^{2}}{2}+\frac{x^{3}}{3}+\cdots+\frac{x^{n}}{n}+\cdots)\quad(-1\leq x<1) \tag{8}$$

将(5)式两边取对数,可化简得

$$\ln P_{r1}=-p_{Ts}nM1-p_{Tr}nMS-m(m-1)p_{Ti}^{2}(M1+M) \tag{9}$$

将 $m=n/\eta$ 代入上式得

$$\ln P_{r1}=-p_{Ts}nM1-p_{Tr}nMS-\frac{n(n-\eta)}{\eta^{2}}p_{Ti}^{2}(M1+M) \tag{10}$$

表 2 给出了(78,64)、(22,16)和(12,8)三种 Hamming 码的 EDAC 系统在运行 24、48、72 小时后所对应的系统可靠性,三种码的编码效率分别是 0.82、0.73、0.67,其对应的系统可靠性几乎没有变化。可见,当纠错能力一定时,编码效率对系统可靠性的影响很小。

表 2　不同编码效率的系统可靠性比较

Tab. 2　Reliability comparison of different code rates

	η	24H	48H	72H
(78,64)	0.82	0.88253	0.77887	0.68738
(22,16)	0.73	0.88253	0.77886	0.68737
(12,8)	0.67	0.88253	0.77886	0.68737

3.2　刷新间隔对可靠性的影响分析

刷新间隔指的是执行两次“刷新”任务之间的时间间隔。在正常情况下,系统按照“刷新”、“运行”、“等待”的任务次序运行,则刷新间隔为 $T=Ts+Ti+Tr$。将“运行”任务进行分解成 $K(K\geq1)$ 个子任务,假设每个子任务执行的时间为 $Tr_{j}(j=1,2\cdots K)$,则 $Tr=\sum_{j=1}^{K}Tr_{j}$。为了使“运行”状态的子任务能够满足系统调度规则,“等待”任务也要相应的拆分成 K 个子任务,且每个子任务的执行时间为 $Ti_{j}(j=1,2\cdots K)$,同理 $Ti=\sum_{j=1}^{K}Ti_{j}$。若在系统能调度的前提下,在每次子任务执行前进行“刷新”

操作,且“刷新”操作的执行时间不变,即相当于刷新间隔为 $T_j=Ts+Ti_j+Tr_j$。为了分析方便,假设每个子任务的执行时间相等,即 $Tr_j=Tr/K, Ti_j=Ti/K(j=1,2\cdots K)$,且执行的代码量也相等,则进行任务拆分后,系统在刷新间隔 T_j 下执行一次“运行”、“等待”、“刷新”任务后的抗单粒子翻转可靠性为:

$$\ln P_{r1}=-p_{Ts}nM1-\frac{p_{Tr}nM}{SK^2}-\frac{m(m-1)(M+M1)p_{Ti}^2}{K^2} \quad (11)$$

图 5 为在不同刷新间隔下的系统运行 24、48、72 小时后的可靠性:随着拆分任务的增加,刷新的间隔和每次运行的代码量都会减少,可靠性将会增长,并且时间越长,其变化越明显:当任务量由 1 拆分成 2 时,即刷新间隔由 1s 减少到 0.5s 左右时,运行 72 小时后,可靠性增加了约 20%。可见,在系统可调度的前提下,拆分的子任务越多,每次执行的代码量越少,则可靠性的增长将越大。

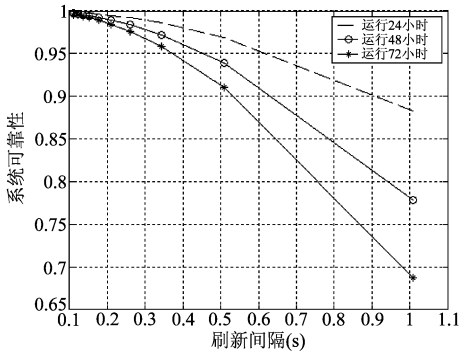

图 5 系统可调度时,不同刷新间隔下系统可靠性的比较

Fig. 5 Reliability comparison of different scrubbing interval with schedulability

以下分析系统在实际调度情况下不允许将任务进行拆分时,即每次运行的代码量不能减少时,刷新间隔对系统可靠性的影响。在一个刷新间隔 $T(T=Tr+Ti+Ts)$ 内,将系统可靠性写成时间 $t(0\le t\le T)$ 的函数如下:

$$P_r(t)=\begin{cases}(1-\lambda t)^{nMS} & 0\le t<Tr\\ (1-\lambda Tr)^{nMS}\{[1-\lambda(t-Tr)]^m+ & \\ m\lambda(t-Tr)[1-\lambda(t-Tr)]^{m-1}\}^{(M1+M)} & \\ & Tr\le t<Tr+Ti\\ (1-\lambda Tr)^{nMS}[1-\lambda(t-Tr-Ti)]^{nM1} & \\ [(1-\lambda Ti)^m+m\lambda Ti(1-\lambda Ti)^{m-1}]^{(M1+M)} & \\ & Tr+Ti\le t\le Tr+Ti+Ts\end{cases} \quad (12)$$

则在刷新间隔为 T 的条件下,系统在运行任意时间 $t(t>0)$ 后的可靠性为:

$$P_{r\_T}(t)=P_r(T)^{\lfloor t/T\rfloor}P_r(t-\lfloor t/T\rfloor T) \quad (13)$$

其中,$\lfloor\rfloor$表示向下取整。当刷新间隔增大,即在 N 个“运行”“等待”任务后执行刷新任务,此时刷新间隔 $T_N=Ts+NTr+NTi(N\ge1)$,则系统运行任意时间 t 后的可靠性为:

$$P_{r\_TN}(t)=P_r(T_N)^{\lfloor t/(T_N)\rfloor}P_r(t-\lfloor t/T_N\rfloor T_N) \quad (14)$$

表 3 及图 6 为在系统不可调度情况下,通过增大刷新间隔,在运行不同时间后的系统可靠性,由表及图可见,当系统调度规则不允许任务拆分时,刷新间隔的选择对系统可靠性的影响已经不大。

表 3 不同刷新间隔下的系统可靠性

Tab. 3 Reliability of different scrubbing intervals

刷新间隔	24H	48H	72H
1 second(N=1)	0.8825	0.7789	0.6874
10 second(N=10)	0.8817	0.7774	0.6854
100 second(N=100)	0.8816	0.7772	0.6852

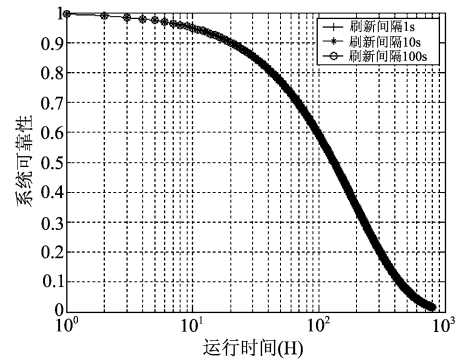

图 6 系统不可调度时,不同刷新间隔下的系统可靠性

Fig. 6 Reliability comparison of different scrubbing interval with unschedulability

3.3 需保护的代码量对可靠性的影响分析

假设系统仍然采用 SEC-DED 的纠错编码来纠检错,由(4)式可知需保护的代码长度 nM 以及译码段的代码长度 $nM1$ 对系统抗单粒子翻转的可靠性是有影响的,为了比较方便,仅假设 M 可变化的情况。假设将需保护的程序代码进行精简,精简后的代码为 $LM(0<L\le1)$ 个码字长度,则精简代码后的系统可靠性为 P_{r_L},由(9)式得:

$$\ln P_{r\_L}=-p_{Ts}nM1-p_{Tr}nLMS-m(m-1)p_{Ti}^2(M1+LM) \quad (15)$$

设精简代码前后系统可靠性的比值为 τ,即 $\tau=P_{r\_L}/P_r$,则

$$\ln\tau=p_{Tr}nM(1-L)S+m(m-1)p_{Ti}^2M(1-L) \quad (16)$$

图 7 为在典型参数条件下,系统运行 24、48、72 小时后,代码压缩比 L 与系统可靠性比值 τ 的关系图,

由图可见由于单粒子翻转率 p_{Tr}、p_{Ti}以及存储区 M 均相对较小,因此 L 与 τ 呈近似线性关系,代码的压缩对于可靠性的增长具有明显的作用,在系统运行 24 小时的情况下,当压缩比提高 0.1 时,可以带来可靠性约 1 个百分点的增长,并且随着运行时间的延长,增长的效果更明显。相反,当压缩后的代码量与之前比反而增大了,即压缩比 L 大于 1 时,会造成系统可靠性的降低,并且随着运行时间的延长,降低效果越明显。此性质同样适用于刷新代码段的代码。

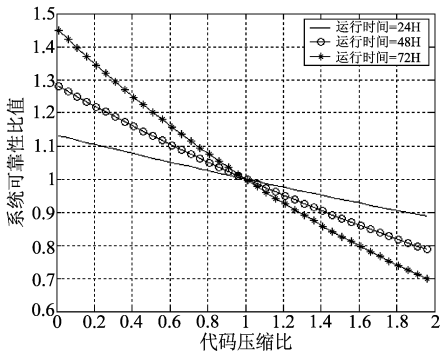

图 7　不同代码量系统可靠性的比较

Fig. 7　Reliability of different program words protected

4　仿真实验及结果

为了验证以上分析的正确性,采用软件故障注入的方法来进行地面模拟验证,对于验证目标平台选择,本文采用了当前在卫星实时信号处理中逐渐取代单片机而占主流的数字信号处理器(Digital Signal Processor, DSP),对空间环境运行中运行的 DSP 而言,通常的 SEU 容错办法是通过扩展存储器接口(External Memory Interface, EMIF)外接 SRAM 做为程序区[14],将程序在片外执行,片外静态存储器(Static RAM, SRAM)中增加了硬件 EDAC 电路,能够纠正和检测一定数量的存储位翻转错误,但由于片外时钟的限制,代码在 SRAM 中执行的速度远远小于片内 RAM 执行的速度,DSP 高速处理的性能难以发挥。采用软件 EDAC 方案后,代码能够在片内运行,同时在时序调度中增加了 EDAC 刷新的任务,对实时性的影响远小于片外。

其实验平台如图 8 所示,DSP 采用 TI 公司的 TMS320C6416,具有共 1Mbyte 的程序空间和数据空间,FPGA 采用 XILINX 公司的 XC3S5000,测试 PC 机通过 FPGA 向 DSP 发送中断信号,使用 DSP 的中断服务程序进行故障注入。同时,DSP 将运行部分状态参数通过 FPGA 发送给 PC 机,在 PC 机上进行测试统计,得出测试结果。

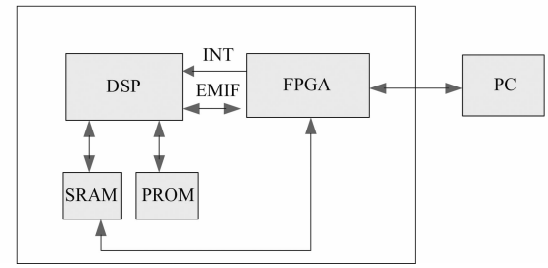

图 8　软件故障注入平台

Fig. 8　Software fault-inject flatform

DSP 中有 4 个任务在运行: Task1 表示正常运行的任务; Task2 表示软件 EDAC 的任务,它定时对受保护区域进行刷新,对随机错误进行检测和纠正; Task3 为故障注入任务,它响应中断后,在测试区域特定位置将代码的某一位或多位进行取反操作;Task4 为空闲任务。当 DSP 上电复位后,首先运行位于地址 0 处的 Bootload 将程序代码从片外非易失存储器 PROM 中加载至片内的程序区,加载完成后,读取 FPGA 的程序运行计数器,累加 1 后写入 FPGA,由 FPGA 将保存的计数器发送给 PC 作为实验次数的统计值,同时也是新的实验开始标志。PC 在一个随机的时间点上发送中断给 DSP,作为故障注入信号,DSP 在中断服务程序中将特定地址的代码更改;在正常运行任务的代码部分,有对故障位置内容的判断,若故障被 EDAC 程序刷新改正了,则 DSP 继续正常运行,等待 PC 的再次故障注入,若否跳转到地址 0 处进行程序的重新加载,同时 FPGA 的计数器加 1,表示系统失效 1 次,新的一次实验开始。表 4 是对纠错编码性能、编码效率、刷新间隔、代码量等 4 个参数修改后的对比实验结果,每种情况均做 10^3 次试验,统计 DSP 程序正常运行的次数,即可得到一个存储位的可靠性。由于实验系统中,故障注入中断对运行的任务会有影响,实际任务的开销不能精确计算等因素,实验结果与理论计算存在略微偏差,但是实验结果的整体规律是符合理论分析的。

表 4　不同情况下的实验结果

Tab. 4　Result of several experiments

	编码性能		编码效率		刷新间隔		刷新间隔		受保护代码量	
	纠一位错	纠二位错	η=0.5	η=0.8	0.5T	0.25T	10T	100T	L=0.3	L=0.8
失效次数	280	265	285	291	215	194	251	259	202	250
单个存储位可靠性	72%	73.5%	71.5%	70.9%	78.5%	80.6%	74.9%	74.1%	79.8%	75.0%

5 结论

与硬件 EDAC 技术相比,软件 EDAC 技术是一种低成本的实现方案,具有实现方法灵活、资源占用率低等特点,但同时可靠性也不如硬件实现方案高,它只适合于在低辐射环境下,在总体硬件体系结构不变的前提下,对系统可靠性进行改进与提升的场合。通过本文对软件 EDAC 实现中纠错编码、刷新间隔以及需保护代码量等因素对可靠性的影响分析,可以得出以下结论:

1、对于 SEU 造成的存储器随机瞬时错误,单个码字的纠错性能对系统可靠性的贡献不大,考虑到纠错能力越强的码,其译码越复杂,结合实现效率考虑,应当优先选择 SEC-DED 的纠错编码;

2、对于 SEC-DED 码(如 Hamming 码),纠错能力为 1,考虑校验位(长度 r)最充分利用情况下的完备码,编码效率 $\eta=1-\frac{r}{2^r-1}$,可知当校验位 r 越短,编码效率越低,但是其译码实现的资源占用率和时延会越少,由于编码效率对系统可靠性的影响不大,在实际应用中,只需要在译码实时性与新增校验位对存储器的占用率之间进行折中;

3、在系统可调度的情况下,将运行任务拆分成若干小任务,从而缩短每次执行的代码量,通过这种方法减小刷新间隔,可以明显提高系统的可靠性;但若系统调度不允许将任务拆分成更小的任务,增加刷新间隔,可靠性也不会明显下降;

4、需保护的代码量是影响可靠性的关键因素,代码量越少,系统可靠性会越高。

参考文献

[1] Wilkinson D C, Stone J L, et al. TDRS-1 single event upsets and the effect of the space environment[J]. IEEE Trans. on Nuclear Science, 1991, 38(6): 1708-1712.

[2] Underwood C I, Oldfield M K. Observations on the reliability of COTS-device-based solid state data recorders operating in low-earth orbit[J]. IEEE Trans. on Nuclear Science, 2000, 47(3): 647-653.

[3] Purohit S, Harrington D, Margala M. An area efficient design methodology for SEU tolerant digital circuits[J]. Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 2010: 981-984.

[4] Li Hao, Lixin Yu. A study on the hardware implementation of EDAC[J]. Third 2008 International Conference on Convergence and Hybrid Information Technology, 2008: 222-225.

[5] Gherman V, Evain S, Cartron M, et al. System-level hardware-base protection of memories against soft-errors[J]. Design, Automation & Test in Europe Conference & Exhibition, 2009: 1222-1225.

[6] Shirvani P P, Saxena N R, McCluskey E J. Software-implemented EDAC protection against SEUs[J]. IEEE Trans. on Reliability, 2000, 49(3):273-284.

[7] Pontarelli S, Ottavi M, Salsano A. Error detection and correction in content addressable memories[J]. 2010 IEEE 25th International Symposium on Defect and Fault Tolerance in VLSI Systems. 2010: 420-428.

[8] Shiyanovskii Y, Rajendran A, Wolff F, et al. Effect of voltage scaling on soft error protection methods for SRAMs[J]. Proceeding of the IEEE 2010 National Aerospace and Electronics Conference, 2010: 320-328.

[9] Goodman R M, Sayano M. The reliability of semiconductor RAM memories with on-chip error-correction coding[J]. IEEE Transactions of Information Theory, 1991, 37(3): 884-896.

[10] Blaum M, Goodman R, McEliece R. The reliability of single-error protected computer memories[J]. IEEE Trans. on Computers, 1988, 37(1): 114-119.

[11] Yang G C. Reliability of semiconductor RAMs with soft-error scrubbing techniques[J]. IEEE Proceedings Computers and Digital Techniques, 1995, 142(5):337-344.

[12] 唐朝京, 雷菁. 信息论与编码基础[M]. 长沙: 国防科技大学出版社,2003, 200-229.

[13] Erl-Huei Lu, Chiou-Yng Lee, Shao-wei Wu. A decoding algorithm for DEC RS codes[J]. IEEE Tencon, 1999, 294-296.

[14] 贺兴华, 肖山竹, 等. 空间 DSP 信息处理系统存储器 SEU 加固技术研究[J]. 宇航学报, 2010, 31(2): 472-477.
HE Xing-hua, XIAO Shan-zhu, et al. Research on Memory's SEU Mitigation Technology in DSP-based Space Information Processing System[J]. Journal of Astronautics, 2010,31(2): 472-477. (in Chinese)

作者简介

刘小汇(1976-),女,广西柳州人,博士生,副研究员。主要研究方向为卫星导航接收机信号处理,空间信号处理系统可靠性设计。E-mail:lululiu_nudt@ sina. com

伍 微(1981-),男,湖北秭归人,博士,讲师,主要研究方向为卫星导航接收机 DSP 实现。

欧 钢(1969-),男,湖南株洲人,博士,教授,博士生导师。主要研究方向为卫星导航接收机信号处理。