A Rapid and Convenient Reliability Evaluation Method for Spacecraft System

YOU Hong-jun, and CHEN Wei-nan, *Member, IAENG*

Abstract—**Evaluating the robustness of digital circuits with respect to soft errors is an important part of the design flow for spacecraft system. System-level soft error reliability analysis is carried out early in design process, which is more efficient and can provide earlier guidance for soft error tolerance design to avoid reworking. Tradition reliability evaluation methods are not suitable for spacecraft systems using FPGA with dynamic partial reconfigurable capacity (DPR-FPGA) in the early stages of system design process. In this context, a two-step component replacement method is proposed to evaluate the reliability of spacecraft system. It allows a spacecraft system designer to evaluate the robustness early in the concept selection phase and to identify whether the dependability of system design does meet the performance requirement of spacecraft in a short time and with low cost.**

Index Terms—**spacecraft; system-level; reliability; evaluation strategy**

I. Introduction

THE dependability analysis of modern embedded systems is a major concern for spacecraft designers. Among the many possible solutions to overcome failures due to soft errors, the architectural approach might be the most cost-efficient. Spacecraft designers often increase system reliability by applying temporal and spatial redundancy methods, such as triple modular redundancy (TMR). One problem with this is that redundancy carries additional costs. To reach the design goals of reliability, performance, power consumption and area simultaneously, the design can only be protected selectively. For selective protection, reliability evaluation is critical. System-level soft error reliability analysis is carried out early in design process, which is more efficient and can provide earlier guidance for soft error tolerance design to avoid reworking [1]. System-level soft error reliability analysis has been the common research focus of the industrial circle and the academy circle.

Because of reconfigurable system provides both flexibility of software and performance of hardware, more and more DPR-FPGA are used in spacecraft system to meet the requirement of performance, power consumption and area simultaneously. As circuit system with DPR-FGA has different characteristics as that of the ordinary electronic system, the reliability estimate method is not suitable for such systems in the early stages of system design.

To avoid costly feedback loops, spacecraft designers need to assess whether the system's reliability meet the demand in the concept selection stage of system design process. In this paper, a two-step component replacement method is proposed to evaluate the reliability of spacecraft system.

This paper is structured as follows. Section II offers an overview of related work. Section III presents the basic principles for evaluating the reliability of the system. A practical case study is elaborated in section IV to estimate the reliability of the orbital internal replacement system by this method. At last, section V presents the conclusions of the work.

II. Related Work

Most of the existing approaches for system-level reliability evaluation of circuits can be classified into two categories: simulation based approaches and formal techniques based approaches.

A. *Simulation based approaches*

Simulation based approaches are to build a test platform, to simulate the behavior of the system, and to assess the reliability of the system by injecting faults. Concerning on soft errors in combinational and sequential elements of digital circuits, a methodology is proposed in [3] to compute the failures in time rate of a sequential circuit where the failures are at the system-level. The goal of [4] is to characterize program reliability phase behavior at the micro-architecture level. The approach of [5] focuses on component based engineering practices and works by an integrated framework for component selection. [6] presents an automatic placement methodology for fault injection evaluations using saboteur techniques which enables the designer to evaluate a high number of different dependability and fault attack scenarios during early design phases using FPGA-based functional emulation.

Early prediction of component reliability is a challenging problem because of many uncertainties associated with components under development. A software component reliability prediction framework is developed in [7] by exploiting architectural models and associated analysis techniques, stochastic modeling approaches, and information sources available early in the development lifecycle.

B. *Formal techniques based approaches*

Formal techniques based approaches apply formal verification methods in the context of dependability

Manuscript received Dec 16, 2012; revised Jan 26, 2013.

YOU Hong-jun is with School of Computer Science, Fudan University, Shanghai, China and with Shanghai Institute of Aerospace Electronic Technology , China(IAENG membership number is: 126318， phone: 086-13564550209; e-mail: 061021044@ fudan.edu.cn).

CHEN Wei-nan is with School of Computer Science, Fudan University, Shanghai, China (e-mail: wnchen@fudan.edu.cn).

evaluation. They consider various problems, and most of them make use of model checking or symbolic simulation techniques. To assess the threat of semiconductor transient faults to reliable software execution, a unified framework, Sim-SODA (SOftware Dependability Analysis), is proposed in [8] to estimate microprocessor reliability in the presence of soft errors at the architectural level. The goal of [9] is to overcome the limitation of the program image and proposes an approach for transient fault injection based on symbolic simulation and model checking that circumvents the problems experienced due to application dependent fault injection and RTL modification. To achieve efficient early identification of unacceptable effects of multiple faults, an approach is proposed in [10] by combining formal property checking and the generation of specific circuit mutants.

In summary, both simulation based approaches and formal techniques based approaches need to provide too much technical details for spacecraft designers to be suitable for the concept design stage.

III. THE PROPOSED APPROACH

Starting with a concept exploration specification, first a high level model of the system is implemented. In space, the spacecraft is affected random by the environment and has nothing to do with the history state. We use finite state machine (FSM) as system-level language enriched by Markov chain. In this combination we descript all possible state of the spacecraft system by FSM and make a model to calculate the state probability. In particular, the legal states can be easily addressed. The high-level FSM model forms an executable specification and serves as the golden model for which also a set of functional properties is specified. These can be validated at the high level of abstraction.

The integration of dependability into a system-level design in concept selection phase in general is a challenging task, which requires explicit knowledge of the application and its environment. According to the faults or observable misbehavior to which the design should react, the designer selects the object of protection and defines the adaptation mechanism.

To evaluate the reliability of each design, we have to follow steps:

Step1. Draw tasks state transition diagram according to the adaptation mechanism.

Step2. Divide the system into modules. Each module consists of a task and its adaptation mechanism, ranging from simple redundancy to runtime reconfiguration or adaptive paths. Calculate the reliability of each module.

Step3. Calculate the reliability of the entire system.

Due to the characteristics of DPR-FPGA, there are no suitable models to evaluate the reliability of tasks running in a chip of DPR-FPGA.

For tasks running in a chip of DPR-FPGA, we call them hardware tasks.

A. Conventions and assumptions

There may be multiple hardware tasks running in one chip of DPR-FPGA. The data flow of the relationship between hardware tasks determines hardware tasks are connected in series or in parallel. The chip welding relationship reflects the spacecraft system circuit versatility and does not reflect relationship between hardware tasks.

Because DPR-FPGA has the ability of dynamical partial reconfiguration, the spacecraft system with DPR-FPGA becomes a repairable system. In order to improve the reliability of this system, it is assumed that follows:

1. Communication data is exchanged through data buffer between hardware tasks, hardware tasks and software tasks. For example, hardware task A's output is hardware task B's input. When hardware task A is abnormal, it does not spread to hardware task B that would affect hardware task B, even makes hardware task B work abnormal.

2. For each chip of DPR-FPGA, only on-chip bus control logic and I/O control logic are in static area, and the chip area being occupied is small. For the convenience of calculation, it is assumed that the ratio of its occupied chip area is negligible.

3. The failure probability of FPGA with DPR ability isλ, and its life cycle X complies with exponential distribution:

$$P\{X \le t\} = 1 - e^{-\lambda t}, t \ge 0, \lambda > 0 \quad (1)$$

Repair time Y after the failure complies with the exponential distribution:

$$P\{Y \le t\} = 1 - e^{-\mu t}, t \ge 0, \mu > 0 \quad (2)$$

Assume that X and Y are independent of each other, and that life distribution of FPGA which encountered SEU and are repaired is same as that of FPGA which never encounter SEU.

Since hardware tasks run in FPGA, their life cycle and their repair time after failure comply with exponential distribution.

For hardware tasks running in the same environment, the following theorem is clear:

Lemma 1 In the condition of different failure probability λ_a and λ_b of different hardware tasks, the corresponding reliability of modules are R_a and R_b .If $\lambda_a < \lambda_b$, then $R_a > R_b$.

B. Selection of hardware task's failure probability λ_{task}

When hardware task's configuration bitstream is generated, a synthesis summary would tell us the number of different resources such as flip-flops, LUTs and BRAMS used. The characterization of these resources is tested and observed in space condition [11-13]. According to the series-parallel relationship between these resources, the failure rate of the hardware task can be calculated. Because of the lack of CAD tools and the deep hardware knowledge requirement, it is difficult to calculate the failure rate λ_{task}.

For an active device, the failure rate is noted in [14] as

$$\lambda = (\lambda_{die} + \lambda_{package} + \lambda_{overstress}) * 10^{-9} / h \quad (3)$$

In which

$$\lambda_{die} = (\lambda_a \times N \times e^{-0.35\alpha} + \lambda_b) \times [\frac{\sum_{i=1}^{y} (\pi_t)_i \times \tau_i}{\tau_{on} + \tau_{off}}] \quad (4)$$

In which, λ_a is per transistor base failure rate of the integrated circuit family. *N* is number of transistors of the integrated circuit. α is equal to[(year of manufacturing) –

1998]. λ_b is failure rate related to the technology mastering of the integrated circuit. $(\pi_t)_i$ is i th temperature factor related to the i th junction temperature of the integrated circuit mission profile. τ_i is i th working time ratio of the integrated circuit for the i th junction temperature of the mission profile. π_{on} is total working time ratio of the integrated circuit, with $\tau_{on} = \sum_{i=\alpha}^{y} \tau_i$. τ_{off} is time ratio for the integrated circuit being in storage (or dormant), with $\tau_{on} + \tau_{off} = 1$.

As shown in [11-13], the failure probability of different resources in FPGA chip in different orbit environment are different. These values may be used to calculate the failure probability coefficient λ_{die}.

For hardware tasks running in the FPGA chip, $\lambda_{package}$ and $\lambda_{overstress}$ are the same as the corresponding parameters of the FPGA chip. It is only different that the number of transistors of the integrated circuit is bigger than that of hardware tasks. From equation (3) and (4), we can draw a conclusion:

$$\lambda_{task} < \lambda_{FPGAchip}.$$

It could be inferred from lemma 1 that the reliability of the hardware task is higher than the value calculated by $\lambda_{FPGAchip}$ instead of λ_{task}. So $\lambda_{FPGAchip}$ is used to evaluate the reliability of the hardware task subsystem in this paper.

IV. A CASE STUDY

A. *Hardware task dynamic maintenance architecture*

In order to exchange some hardware tasks and to keep them health, a co-maintenance framework is designed as shown in Fig.1. Replacement and maintenance of hardware tasks are cooperated by HRRM, hardware task agents and HSM. Monitoring hardware tasks is cooperated by hardware task agents and HSM.

The following is a summary of the various components in the framework introduced in accordance with the order from top to bottom:

HRRM: responsible for the hardware task's status updates and maintenance, and distribution of the required reconfigurable computing. During hardware tasks running, it monitors whether is there a hardware task working not as expected. According to the health information provided by HSM and by hardware task agents, HRRM makes decisions for hardware tasks maintenance. If a hardware task is out of the way, HRRM informs HSM to reconfigure the hardware task.

Hardware task agent: for each hardware task, there is a software agent serving for it. The communication and the corresponding synchronization operations are completed by the agent. It needs to receive the heartbeat information from the corresponding hardware task and report the hardware task's status to HRRM. If it continuously receives abnormal heartbeat information twice, the corresponding hardware task configuration bitstream (HTCB) maybe have been destroyed

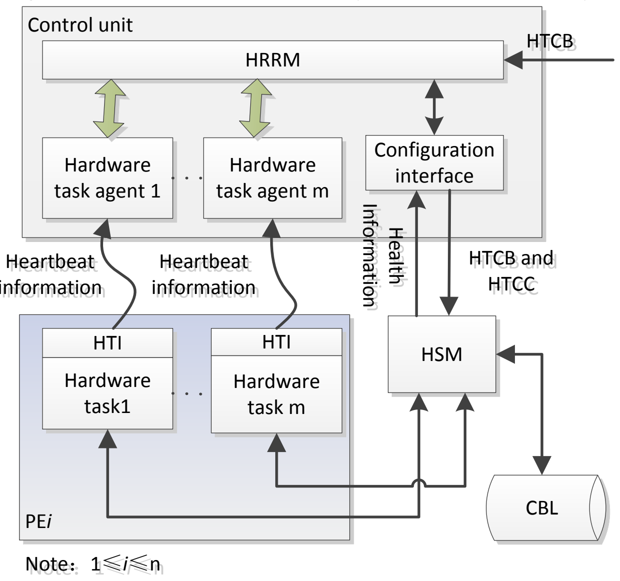

Fig. 1. Hardware task dynamic maintenance architecture

and need to be repaired by reconfiguring HTCB.

Hardware task interface (HTI)[15]: the communication interface of the hardware task which deals with the details of the underlying communication, synchronizes the hardware task with the hardware task agent by blocking, regularly forwards hardware tasks heartbeat information to the hardware task agent.

HSM: is responsible for real-time monitoring hardware tasks and reporting to HRRM. It has configuration control functions. After receiving hardware task configuration command (HTCC), it selects the corresponding configuration data from CBL to complete the configuration. It is high-grade anti-radiation devices that are the best implement for HSM. A hardware task needs to be reconfigured if HSM found it abnormal twice.

CBL: stores all configuration bitstream including all hardware task's configuration bitstream in non-volatile memory (such as anti-radiation FLASH EEPROM).

B. *Calculation of the hardware task subsystem's reliability*

As shown as Fig1, if there is any fault in HSM, it is difficult to record the health status of HTCB correctly. So HSM's function is implemented by high-grade aerospace grade anti-fuse FPGA chip. Thus, the model shown in Fig1 can be simplified to the model shown in Figure 2.

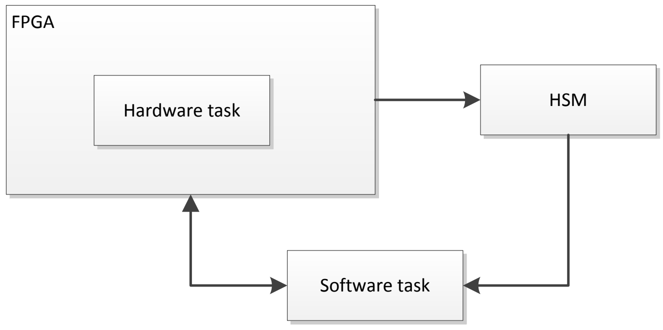

Fig. 2. Hardware task model

According to the model, there are, for hardware tasks, four states: normal state (heartbeat information and HTCB information show hardware task working healthy), repairing state I (heartbeat information appears exception once), repairing state II (HTCB information appears exception once), and the failure state (two exceptions are showing by heartbeat information or HTCB information). These states are expressed as S_0, S_1, S_2, and S_3.

Now there are three parameters needed to be introduced. Parameter and in turn indicate the abnormal probability of communication between hardware tasks and hardware task agents, which is caused by that the components of the FPGA such as flip-flop and BRAM encounter SEU, and the abnormal probability of HTCB information. The repair rate of hardware task is indicated by the parameter μ. The hardware task state transition relationship is shown in Fig 3.

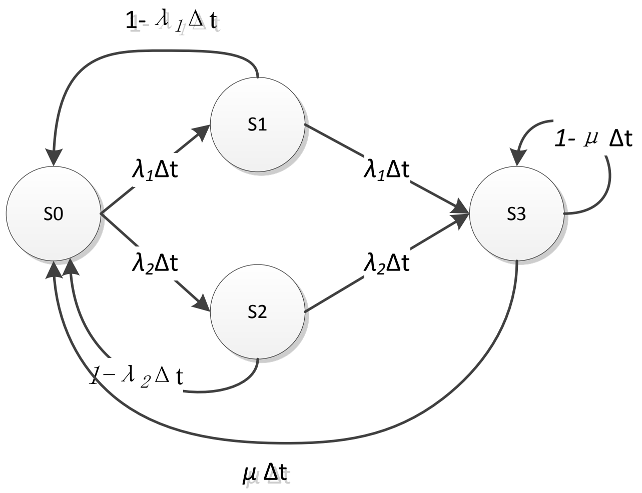

Fig. 3. Hardware task state transition

S_0 is the normal state. S_1 is the repairing state I. S_2 is the repairing state II. S_3 is the failure state.

λ_1 represents the failure probability of FPGA's components such as flip-flop and BRAM when encountered the SEU events. λ_2 represents the failure probability of the configuration memory. μ represents the probability of hardware tasks recovering from failure state by reconfiguring. According to Fig.3, the state transition matrix of the sub-system is

$$T = \begin{bmatrix} 1-\lambda_1-\lambda_2 & 1-\lambda_1 & 1-\lambda_2 & \mu \\ \lambda_1 & 0 & 0 & 0 \\ \lambda_2 & 0 & 0 & 0 \\ 0 & \lambda_1 & \lambda_2 & 1-\mu \end{bmatrix} \quad (5)$$

The coefficient matrix of the state equation is

$$A = T - I = \begin{bmatrix} -\lambda_1-\lambda_2 & 1-\lambda_1 & 1-\lambda_2 & \mu \\ \lambda_1 & -1 & 0 & 0 \\ \lambda_2 & 0 & -1 & 0 \\ 0 & \lambda_1 & \lambda_2 & -\mu \end{bmatrix} \quad (6)$$

The state equation is

$$\begin{bmatrix} \dot{p}_0(t) \\ \dot{p}_1(t) \\ \dot{p}_2(t) \\ \dot{p}_3(t) \end{bmatrix} = \begin{bmatrix} -\lambda_1-\lambda_2 & 1-\lambda_1 & 1-\lambda_2 & \mu \\ \lambda_1 & -1 & 0 & 0 \\ \lambda_2 & 0 & -1 & 0 \\ 0 & \lambda_1 & \lambda_2 & -\mu \end{bmatrix} \begin{bmatrix} p_0(t) \\ p_1(t) \\ p_2(t) \\ p_3(t) \end{bmatrix} \quad (7)$$

The initial condition is that $p_0(t)=1$, $p_1(t)=0$, $p_2(t)=0$, $p_3(t)=0$. After being calculated, the reliability of the sub-system is

$$\begin{aligned} R(t) &= p_0(t)+p_1(t)+p_2(t) \\ &= \frac{1}{A}\Big\{4\mu + W[(A-4\mu)\cosh(\frac{1}{2}tM) \\ &\quad + \frac{(-3a+b+3\mu-1)A-4\mu^2+2\lambda_1^2+2\lambda_2^2}{M}\sinh(\frac{1}{2}tM)]\Big\} \end{aligned} \quad (8)$$

Among them, A= $\lambda_1\mu+\lambda_2\mu+\mu+\lambda_1^2+\lambda_2^2$,

W= $e^{-\frac{1}{2}(\lambda_1+\lambda_2+\mu+1)t}$,

$$M = \sqrt{-3\lambda_1^2+2\lambda_1\lambda_2-2\lambda_1\mu+2\lambda_1-3\lambda_2^2-2\lambda_2\mu+2\lambda_2+\mu^2-2\mu+1} \quad (9)$$

In the LEO orbit, if the hardware task can be repaired successful by reconfiguration only once, the reliability of the sub-system could be higher than 0.99 after three years, according to the formula (8). In general, it could meet the requirement of spacecraft.

V. CONCLUSIONS

In the early stages of the spacecraft design, the designers are concerned about whether the lower limit of the system reliability meets the requirement, rather than the accurate value of the system reliability. If the lower limit of the system does meet the demand, the design of the spacecraft system will not be re-working. Because of the characteristics of DPR-FPGA, there is no suitable method to evaluate the reliability of the system with DPR-FPGA. In this paper we have presented a two-step component replacement method to assess the reliability of the spacecraft. First, the reliability of hardware tasks sub-system is evaluated by $\lambda_{FPGAchip}$ instead of λ_{task}. Then the reliability of the spacecraft system could be assessed according to the relationship of hardware tasks. When the circuit is still on paper, designers can assess whether the design to meet the demand by this method. When all technical indicators can meet the needs of the system, sub-system design tasks could be subcontracted to the cooperators. Therefore, it is possible to avoid reworking.

REFERENCES

[1] T.P .Monaghan, "Fault tolerant system design in the concept exploration stage of a mission critical computing system", *1996 IEEE Aerospace Applications Conference*, pp. 321 - 333.

[2] R. Leveugle, L. Pierre, P. Maistri, et,al, "Soft Error Effect and Register Criticality Evaluations: Past, Present and Future", *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE'09)*, Stanford (CA)

[3] D. Holcomb, Li Wenchao, S.A. Seshia, "Design as you see FIT: System-level soft error analysis of sequential circuits," *Design, Automation & Test in Europe Conference & Exhibition*, 2009, pp.785 – 790.

[4] Fu Xin, J. Poe, Li Tao, et al, "Characterizing Microarchitecture Soft Error Vulnerability Phase Behavior," *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '06)*, pp. 147 – 155.

[5] C. Calvert, G.L Hamza-Lup, A. Agarwal, et al, "An Integrated Component Selection Framework for ystem-Level Design," *2011 IEEE International Systems Conference (SysCon)*, pp. 261 – 266.

[6] J. Grinschgl, A. Krieg, C. Steger, et al, "Automatic Saboteur Placement for Emulation-Based Multi-Bit Fault Injection," *2011 6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, pp. 1 – 8

[7] L. Cheung, R. Roshandel, N. Medvidovic, et al, "Early Prediction of Software Component Reliability," *ACM/IEEE 30th International Conference on Software Engineering*, 2008, pp. 111 – 120

[8] Fu Xin, Li Tao, F. Jose, "Sim-SODA: A Unified Framework for Architectural Level Software Reliability Analysis," *Workshop on Modeling, Benchmarking and Simulation* (Held in conjunction with International Symposium on Computer Architecture), June 2006

[9] A. Darbari, B. Al Hashimi, P. Harrod, et al, "A New Approach for Transient Fault Injection using Symbolic Simulation," *14th IEEE International On-Line Testing Symposium 2008*, pp. 93 – 98

[10] R. Leveugle, "A New Approach for Early Dependability Evaluation Based on Formal Property checking and Controlled Mutations," *Proceedings of the 11th IEEE International On-Line Testing Symposium (IOLTS'05)*, 2005,pp. 260 – 265

[11] A. Gregory, S. Gary, C. Carl, "VIRTEX-4QV STATIC SEU CHARACTERIZATION SUMMARY," http://parts.jpl.nasa.gov/wp-content/uploads/NEPP07FPGAv4Static.pdf

[12] Radiation-Hardened, "Space-GradeVirtex-5QV Family Overview," DS192 (v1.3) March 8, 2012.[EB/OL]. [2012.04.28] http://www.xilinx.com/support/documentation/data_sheets/ds192_V5QV_Device_Overview.pdf

[13] G. Swift., "Compendium of XRTC Radiation Results on All Single-Event Effects Observed in the Virtex-5QV," [C] //*ReSpace/MAPLD 2011 Conference*.NewMexico,USA. 2011:1-33

[14] IEC-TR-62380: 2004, *Reliability data handbook-Universal model for reliability prediction of electronics components, PCBs and equipment first edition* [S]

[15] Wang Ying, Zhou Xue-Gong, You Hong-Jun, Peng Cheng-Lian, "A Unified SW/HW Multi-thread Programming Model Supporting Dynamic Reconfiguration[J]," *JOURNAL OF COMPUTER-AIDED DESIGN & COMPUTER GRAPHICS*, 2009,21(6):736-745. (in Chinese)