A Roll-Forward Recovery Scheme in TMR Systems for Real-Time Applications

Shu-Yi Yu and Edward J. McCluskey
CENTER FOR RELIABLE COMPUTING
Stanford University, Stanford, California 94305-4055
{syu, ejm}@crc.stanford.edu

Abstract

Triple Modular Redundancy (TMR) is known to improve reliability in real-time computing systems for short mission times. However, TMR-based systems are not effective for longer missions. We present a new recovery scheme for TMR systems to recover from failures caused by transient faults, so that mission time of TMR systems can be lengthened. This scheme can effectively recover computing systems from single transient faults without introducing any re-computation delay. Hence, it is suitable for real-time applications. We implemented the recovery scheme in a robot controller as a case study. Analytical results show that the recovery scheme can significantly improve system reliability, and synthesized area results show that the scheme introduces very little hardware overhead.

1. Introduction

TMR systems are extensively used in real-time applications because of their fault masking ability [Siewiorek 00]. However, they are only effective for short mission times. Different methods have been presented to lengthen lifetime of TMR, such as self-purging [Losq 76], which is suitable for recovering from both transient and permanent faults, and recovery schemes for transient faults [Chande 89]. However, re-computation or retry is used in these schemes to recover systems from transient faults. In real-time applications, re-computation and retry delay is not allowed because of strict deadlines. To meet real-time application requirements, in this paper we present a roll-forward recovery scheme that exploits redundancy in TMR so that no re-computation or retry is needed.

2. Roll-Forward Recovery Scheme for TMR

Our scheme is suitable for recovering corrupt data in storage elements such as latches and registers in computing modules when failures are caused by transient faults. Figure 1 illustrates the recovery scheme for a TMR system. System outputs are obtained through voting among three modules. Single module failures are masked by voters. Error detection is added to TMR by comparing voted outputs with module outputs [Losq 76]. When the module outputs are different, an error indication signal is generated to identify the faulty module. A separated recovery controller checks error detection results periodically at pre-scheduled computation points to determine if there is a need for recovery. If recovery is needed, the corrupt data is recovered by copying correct data from fault-free modules to the faulty one. After recovery the normal operation is continued again. As shown in Fig. 1, for faults in a single module, the fault-free modules provide sufficient information for recovery. Therefore, no re-computation is needed.

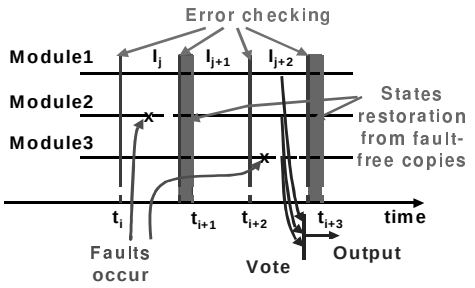

Figure 1. The recovery scheme for TMR.

State restoration is accomplished by copying data from correct copies to the faulty one by separated controller logic. Two schemes, the voted scheme (VT) and the direct-load scheme (DL) are presented here. The VT scheme obtains correct states by voting among the three copies. The voted results are loaded into all three copies for state restoration. The DL scheme obtains correct states by loading states directly from one of the correct copies into the faulty one. Both schemes are effective for recovering from single faults. The voted scheme can also recover from some multiple faults.

3. Reliability Analysis

Reliability of a TMR system with our recovery scheme is calculated and compared with a simplex, a TMR, and a TMR-Simplex system. We choose a robot controller as our case study for reliability analysis. This is part of our research in the ROAR project [ROAR 01]. Details of our previous work on the fault tolerant robot controller can be found in [Yu 00]. The recovery scheme is applied in two types of TMR, *hardware redundancy* (HR) and *multi-threading* (MT) [Saxena 00]. The synthesized area of the robot controller is used to calculate reliability. Details of the synthesis procedure are described in Sec. 4. Synthesized data for reliability analysis were extracted using Xilinx Virtex FPGA library, since the controller will be used in an adaptive computing system in the ROAR project.

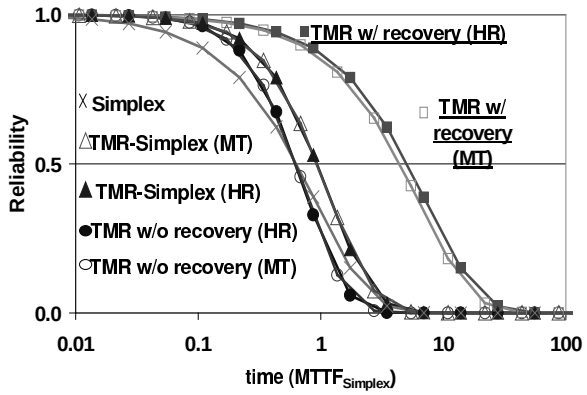

Figure 2. Reliability of the robot controller with different fault tolerance techniques.

Figure 2 shows the reliability vs. time with time axis normalized to mean time to failure (MTTF) of the simplex design. For systems with recovery, we use the data of the VT scheme for calculation since it has larger area and we want to calculate a lower bound for reliability. The result shows that TMR with the recovery scheme has much better reliability than other systems. Both hardware redundancy and multi-threading with the recovery scheme have shown this improvement. HR-TMR has reliability slightly better than MT-TMR, since the MT design has larger common logic, such as the thread scheduler, that is shared by the three threads and is not protected by TMR.

4. Implementation

We implemented the recovery scheme in the robot controller to estimate the overhead of the roll-forward recovery scheme. The designs are synthesized with target libraries including Xilinx Virtex XCV1000 FPGA, and LSI lcbg10p library, using Synplify® and Synopsys® tools, respectively. The results are listed in Table 1 and Table 2. The numbers are normalized, assuming that the area of the simplex design is 1. Sequential logic (registers) and combinational logic (lookup tables -- LUTs) areas are listed. The algorithm takes 7 cycles to finish in hardware redundancy, and 11 cycles for multi-threading. Both VT and DL schemes take 1 clock cycle for recovery.

Table 1. Synthesized area with FPGA library.

	TMR	Recover	Restore	Reg	LUT
Simplex	N/A	No	N/A	1.00	1.00
TMR	HR	No	N/A	3.00	3.03
		Yes	VT	3.09	3.55
			DL	3.09	3.45
	MT	No	N/A	2.30	2.31
		Yes	VT	2.38	2.78
			DL	2.38	2.76

From the results, we found that the recovery logic introduces very small overhead compared with triplication. Among state restoration schemes, the direct-load scheme has less overhead than the voting scheme. A multi-threaded design costs less hardware than a hardware-redundant one, but has longer computation latency. The overhead of combinational logic is less in the LSI library than in the FPGA library. It is because FPGAs have fewer types of logic cells and they are less flexible for logic mapping. Nevertheless, FPGAs provide the flexibility of reconfiguration that can be exploited for repairing permanent faults [Huang 01].

Table 2. Synthesized area with LSI library.

	TMR	Recover	Restore	Seq	Comb
Simplex	N/A	No	N/A	1.00	1.00
TMR	HR	No	N/A	3.00	3.01
		Yes	VT	3.06	3.14
			DL	3.06	3.13
	MT	No	N/A	2.35	1.29
		Yes	VT	2.41	1.42
			DL	2.41	1.40

5. Conclusion

In this paper, we have presented a new roll-forward recovery scheme to enhance the reliability of TMR systems. Analysis based on a robot controller has shown that reliability of a TMR design is significantly improved by adding the recovery scheme. We have applied the scheme to two forms of TMR: hardware redundancy and multi-threading. Hardware redundancy is more suitable when deadline requirements are strict, while multi-threading is suitable for limited hardware. The recovery mechanism introduces small hardware overhead, and does not need re-computation for single faults. Therefore, it is very suitable for real-time applications.

Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. DABT63-97-C-0024.

References

[Chande 89] Chande, P. K., A. K. Romani, and P. C. Sharma, "Modular TMR Multiprocessor System," *IEEE Trans. On Industrial Electronics*, Vol. 36, No. 1, pp. 34-41, 1989.

[Huang 01] Huang, W.-J. and E. J. McCluskey, "Column-Based Precompiled Configuration Techniques for FPGA Fault Tolerance," to appear in *FCCM'01*, 2001.

[Losq 76] Losq, J. "A Highly efficient Redundancy Scheme: Self-Purging Redundancy," *IEEE Trans. Comp.* C-25, pp. 569-578, 1976.

[ROAR 01] crc.stanford.edu/projects/roar/roarSummary.html, 2001.

[Saxena 00] Saxena, N.R., S. Fernandez-Gomez, W.J. Huang, S. Mitra, S.-Y. Yu and E.J. McCluskey, "Dependable Computing and Online Testing in Adaptive and Configurable Systems", *IEEE Design and Test of Computers*, Vol. 17, No. 1, pp. 29-41, 2000.

[Siewiorek 00] Siewiorek, D. P. and R. S. Swarz, *Reliable Computer Systems – Design and Evaluation*, 3rd Ed, Digital Press, 2000

[Yu 00] Yu, S.-Y., N. Saxena, and E. J. McCluskey, "An ACS Robotic Control Algorithm with Fault Tolerant Capabilities," *FCCM'00*, pp. 175-84, 2000.