International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 05, May-2015

Implementation of Error Recovery in TMR Systems using Scan Chain-based Technique

Sarfaraz Nawaz Qureshi
M.tech, Dept. of ECE
B.N.M Institute of Technology
Bengaluru, India

Vrunda Kusanur
Assistant Professor, Dept. of ECE
B.N.M Institute of technology
Bengaluru, India

Abstract*—In this project a scan chain based technique is employed to recover multiple errors in Triple modular redundancy (TMR) systems. This technique makes use of the scan chain flip-flops that are employed for testability purposes to recover the correct state of a Triple Modular Redundancy system in the presence of temporary or permanent errors.This technique can detect both permanent faults and transient faults (temporary faults) and can be used in safety critical applications such as patient care systems,avionics, etc. Real-time computing systems are extensively used in our daily lives. Cars, telephone networks, and patient care systems, all contain real-time computing systems. For instance, a delayed braking signal in a cruise control system of a car may cause a car accident, and a delayed output in an anesthesia control system may give incorrect dosage of medicine to a patient. Real-time systems are not necessarily fast systems, but their outputs have to meet strict time constraints. Hence, predictable performance is needed and is very important in these systems. In order to meet the reliability requirement, these systems must be facilitated with suitableerror detection and correction mechanisms. However, it must be realized that achieving a considerably high level of reliability and assuring that the timing requirements are met are conflicting objectives, that is, the reliability enhancement may have anunwanted impact on timing constraint. The generalized idea to achieve error detection and correction is to incorporate redundancy (i.e., some extra data) to the actual message, to check the consistency of the message delivered and to restorethe data determined to be corrupted. On detection of any mismatch, the faulty or erroneous modules are identified and the state of anerror free module is copied into the erroneous modules. In case a permanent fault is detected, the system is updated to a master/checker mode by disregarding the erroneous module. Exhaustive fault injection experiments reveal that the this architecture has the maximum error detection and their recovery and imposes a negligible performance and area overhead as compared to traditional techniques based on TMR.

Keywords—Triple Modular Redundancy, transient faults, Exhaustive faults, Real time computing.

I. INTRODUCTION

Nowadays, the usage of embedded systems in applications such as process control and life support monitoring of patient has become prominent. Such a system often has timing constraints and error tolerance requirements. Many real-time computing systems are used under hazardous or remote circumstances, such as in nuclear power plants, aircraft, and spacecraft. In these environments, computing systems are highly susceptible to errors due to radiation. Maintenance and repair is usually very expensive and time consuming for these applications. Hence, besides performance, fault tolerance, which is the ability of a system to operate correctly in spite of the occurrence of faults become a very important issue.

The maindrawback of the traditional TMR is its incapability to cope with TMR failures. In case of fault arrivals in two different modules and if none of the faults is restored, it may result in a failure for long term applications.The absence of appropriate fault restoring mechanisms considerably increases the chances of TMR failure occurrence. To address this issue, TMR must be facilitated with an error recovery mechanism. Most of the traditional TMR based error recovery techniques proposed exploit retry mechanisms. These mechanisms are not tolerable for applications where tight deadline is to be achieved, as the re-computation may result in a task completion after its deadline has already elapsed.

A TMR based technique which is applicable to general purpose circuits is Scan chain based TMR(ScTMR). ScTMR considerably reduces the chances of TMR failures; it suffers from two major limitations. Firstly, in the presence of latent faults ScTMR is unable to recover a single faulty module in the TMR system. Secondly, ScTMR is not able to restore the system if more than one faultis simultaneously propagated to the outputs of two modules.

The Scan chain based Multiple Error Recovery TMR has the capability to identify and restore latent faults in TMR modules and also restore the system from multiple faults affecting two TMR modules. It compares the internal states of TMR modules to identify and restore the error-free state using the state of non-faulty modules. In case of permanent faultsthe system is updated to a master/checker (M/C) configuration. In roll-forward recovery mechanisms there is no re-computation as compared to retry based recovery mechanism hence it can be used in safety critical applications. A roll forward mechanism for TMR systems has been proposed here.

II. PROPOSED METHOD

The proposed technique has the capability of recovering multiple errors up to two errors and is called as scan chain based multiple error recovery technique for TMR systems (SMERTMR). This technique operates in two modes:Comparison mode and recovery mode.The comparison mode is activated in the following two cases:

First when the voter detects an error and second when the checkpoint signal is asserted. In the second case, the checkpoint signal is employed to trigger the comparison mode so that latent faults can be eliminated. This checkpoint signal can be asserted during slack times, if there is nosufficient slack time, the comparison mode is activated only in the first case, i.e., the comparison mode can be activated once an error in a module propagates to the outputs of the module and voter detects it. So when the comparison mode is not regularly activated by the checkpoint signal, latent faults are detected and identified once a next fault is propagated to the module outputs and detected by the voter. This can increment the chances of having multiple faulty or erroneous modules since there is more opportunity for the next fault to occur. In order to reduce the delay in locating latent faults, the comparison mode can frequently be triggered by activating the checkpoint signal in predefined time intervals. It can be noted that having slack time is common in real-time embedded systems. In this technique, we have utilized the available slack time in such systems to enhance the reliability of the system.

As mentioned before, in the SMERTMR technique the internal states of modules are compared together, whereas in the ScTMR technique merely the module outputs are compared by the voter circuit. As a result of this following advantages are achieved for SMERTMR technique: 1) In case, if there is enough slack time to regularly execute the comparison process, the chances of having multiple faulty modules is significantly decremented.

2) 2 faulty modules can be efficiently detected and identified. So, the faulty modules can be recovered using the states of the fault-free or error-free module.

A. *Comparison mode*

In the SMERTMR technique, whenever the voter circuit detects a fault, it asserts an error signal to activate the SMERTMR controller circuit. On assertion of the error signal, the SMERTMR controller changes from the normal operation mode to the comparison process mode to locate the erroneous modules. Once the faulty module is located, SMERTMR changes to the recovery mode to recover the erroneous modules using the state of one of the fault-free or error free modules. Fig1 shows a block diagram of the SMERTMR controller operating in the comparison mode. In this mode, for comparison of internal states of the TMR modules are shifted out and all module pairs (I/II, I/III, and II/III) are checked for mismatches using scan chain. As shown below, three counters are used namely, $counter_{12}$, $counter_{13}$, and $counter_{23}$, so that the number of mismatches between each module pairs can be stored. For example, $counter_{12}$ is used to store the number of mismatches between modules 1 and 2. The SMERTMR controller activates the scan chains of the SMERTMR modules in such a way that the SCI signal in each module is connected to the SCO signal of the same module.During the shift operationbyusing XOR gates the internal states of the modules are compared.

Every time a mismatch is sensed, the corresponding counter is increased by 1 unit. By making use of this configuration, $counter_{ij}$ will be containing the number of mismatches between modules j and i after Lsc cycles of the clock.

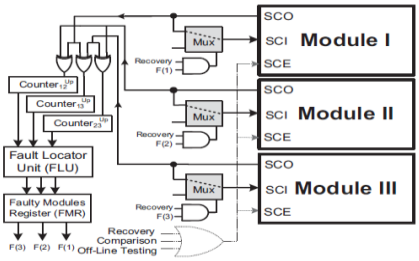

Fig 1: Comparison mode of SMERTM

B. *Recovery Mode*

After the location of fault-free and faulty module by the Fault location unit at the end of the comparison mode, the system will enter the recovery mode if there is 1 or 2 faulty modules in the system. Otherwise, it returns to the normal operation. In recovery mode, the state of the faulty module is restored by the state of error-free modules using the implemented scan chains. Fig 2 shows a simplified block diagram of the SMERTMR controller in the recovery mode. In this mode, the SMERTMR controller will enable the scan chains of the SMERTMR modules and configure the multiplexers which is given as follows: The SCI signal of error-free modules is connected to the SCO signal of the same module. In addition, the SCI signal of the erroneous module is connected to the SCO of one of the error-free modules. As shown in Fig. below, the value of the faulty module register (FMR) is utilized to select the incoming driver of the correct signal driver for the SCI signals in recovery mode. By using the configuration shown in Fig. 2, the state of 1 of the error-free modules is copied into the faulty modules after Lsc cycles of clock. During shifting out the states of the modules in the recovery process, similar to the comparison mode, they are also analyzed to find any mismatch due to faults occurring in the recovery process. During the recovery mode, whenever a mismatch is sensed, the corresponding counter value containing the number of mismatches is decreased by 1 unit. At the end of the recovery mode, all counters should be 0.

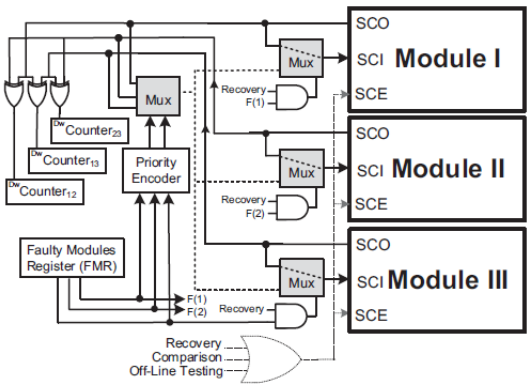

Fig 2: Recovery mode of SMERTMR

Because for each mismatch, the corresponding counter is increased by 1 unit during the comparison process and is decreased by 1 unit during the recovery process. If either of the counters holds nonzero value at the end of the recovery

mode, it indicates another fault may have occurred during the recovery mode. In such case, the SMERTMR system enters the unrecoverable condition since faults in such situations cannot be located.

Permanent error detection in SMERTMR is same as that of ScTMR technique. SMERTMR uses the history of erroneous modules to detect permanent errors. The only difference between ScTMR and SMERTMR in permanent error detection is that ScTMR saves the status of the module (faulty or fault-free) based on the results of the voter circuit in the MRFM register but SMERTMR saves the results of comparison process (which are saved in FMR) in the MRFM register. The proposed history-based detection mechanism for permanent fault, however, can misinterpret a transient fault as a permanent fault or vice versa. If the consecutive transient error occurs in a less period in 1 module, the SMERTMR misinterprets transient faults as a permanent fault and the system is updated to the master checker (M/C) configuration. However, the probability of occurrence of consecutive transient faults in 1 module while no error occurs in the other modules is very low. If a predefined threshold (TR) is exceeded by NCF in a module, the module is assumed as permanently faulty. In such case, the probability that consecutive transient faults can be detected as a permanent fault is equal to a value $1/3^{TR-1}$.

C. Algorithm

1) *If $counter_{ij}$ =$counter_{ik}$ =$counter_{jk}$ =0 then*
2) *Next_state=normal state*
3) *Else if ($counter_{ij}$ =$counter_{ik}$) & ($counter_{jk}$ =0) then*
4) *Next_state=recovery state*
5) *Faulty module register=i is the faulty module*
6) *Else if ($counter_{jk}$ =x) & ($counter_{ik}$ =y) & ($counter_{ij}$ =x+y) then*
7) *Next_state=recovery state*
8) *Faulty module register=i and j are faulty module*
9) *Else*
10) *Next_state=Unrecoverable condition*
11) *End*

D. State diagram

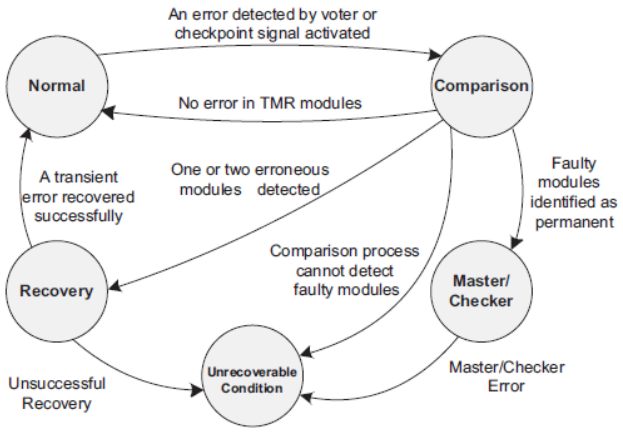

Fig 3: State diagram for SMERTMR

By default the system is in normal state. Once the error is detected the system is changed to comparison mode to check the internal states of the module. If either one or two errors are detected, the system switches to recovery mode. After the recovery process is done the system returns to its default normal mode. Else if permanent error is detected the system is changed to master/checker mode. If the comparison mode cannot detect the errors the unrecoverable condition is entered.

III. Results And Discussion

The below simulation results depict the output of SMERTMR technique. It can be seen that errors are injected in two modules i.e. module I and module II. Due to this outputs of these modules vary and are erroneous .The system is able to detect both of these errors as it supports multiple error recovery. Once the error is detected comparison mode is activated to detect the errors and consecutively the recovery mode is activated. Hence outputs of the erroneous modules are restored to the fault free state. The power consumed by the circuit is around 0.034W as shown in table 1. Design summary in Table 2 illustrates the number of slice registers the implementation takes 26 slice registers which comprises of 24 flip flops and 2 latches.Hence the area overhead to implement this technique is quite less. The analysis of results is done in Xilinx 14.2 software.

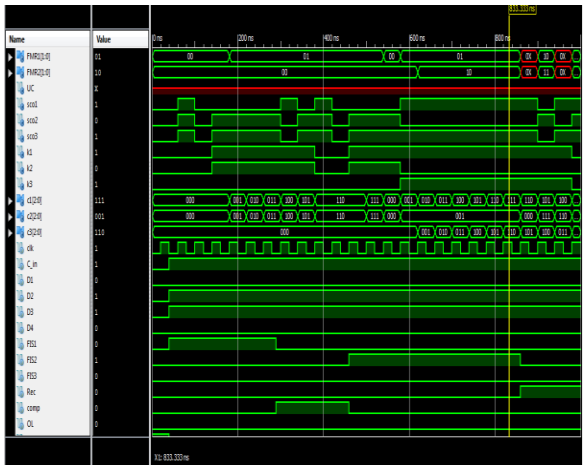

Fig 4a: Simulation results

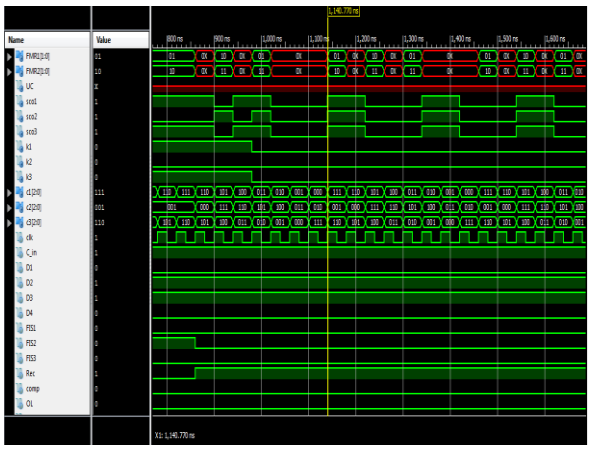

Fig 4b: Simulation results

A	B
On-Chip	Power (W)
Clocks	0.002
Logic	0.000
Signals	0.000
IOs	0.000
Leakage	0.032
Total	0.034

Table 1: Power consumption

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Total Number Slice Registers	26	11,776	1%	
Number used as Flip Flops	24			
Number used as Latches	2			
Number of 4 input LUTs	93	11,776	1%	
Number of occupied Slices	49	5,888	1%	
Number of Slices containing only related logic	49	49	100%	
Number of Slices containing unrelated logic	0	49	0%	
Total Number of 4 input LUTs	93	11,776	1%	
Number of bonded IOBs	33	372	8%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	3.73			

Table 2: Design utilization summary

IV. CONCLUSION AND FUTURE SCOPE

A. *Conclusion*

In this project a technique called roll-forward technique is presented to recover errors using SMERTMR method in triple modular redundancy systems. The SMERTMR technique have the capability to recover multiple errors i.e. up to two faulty TMR modules. In is to be known that fault injection here is done manually. Inference can be drawn that SMERTMR technique is quite reliable one as it is capable of countering more than one error. The area and the performance overhead of SMERTMR technique is quite less compared to the traditional TMR system. Another important thing is that SMERTMR technique consumes less power compare to the traditional techniques.

B. *Future scope*

The adverse effects due to the transient and permanent errors may cause system failure in many real-time applications where a system failure may result in life challenging problems. This can be more prominent in deep sub-micron technologies. Hence in order to elevate the reliability of the system this scan chain technique can be well extended to more number of modules such as Five modular redundancy (FMR) system. With this, the fault tolerance capability of the system increases as there will be more number of modules holding correct output. It should be noted here that the redundancy in system increases the reliability but at the increased device cost.

REFERENCES

(1) HosseinAsadi, SeyedGhassemMiremadi and MojtabaEbrahimi, "ScTMR: A Scan Chain-Based Error Recovery Technique for TMR Systems in Safety-Critical Applications," pp. 1-4, IEEE Paper, march 2011.

(2) K. Shin and H. Kim, "Design and analysis of an optimal instructionretry policy for TMR controller computers," IEEE Transactions on Computers, vol. 45, no. 11, pp. 1217–1225, nov. 1996.

(3) E. McCluskey and S. Yu, "On-line testing and recovery in TMR systems for real-time applications," in Proceedings of International Test Conference, 2001, pp. 240–249.

(4) H. Kim and K. Shin, "A time redundancy approach to TMR failures using fault-state likelihoods," IEEE Transactions on Computers, vol. 43, no. 10, pp. 1151–1162, oct. 1994.

(5) F. Saigne, M. Gedion,R. Schrimpf, J. Gasiot, and F. Wrobel, "Radioactive nuclei induced soft errors at ground level," IEEE Transactions on Nuclear Science, vol. 56, no. 6, pp. 3437–3441, dec. 2009.