International Journal on Recent Researches In Science, Engineering & Technology
(Division of Computer Science and Engineering)
A Journal Established in early 2000 as National journal and upgraded to International journal in 2013 and is in existence for the last 10 years. It is run by Retired Professors from NIT, Trichy. It is an absolutely free (No processing charges, No publishing charges etc) Journal Indexed in JIR, DIIF and SJIF.
Research Paper
Available online at: www.jrrset.com

ISSN (Print) : 2347-6729
ISSN (Online) : 2348-3105

Volume 4, Issue 1, February 2016.

JIR IF : 2.54
DIIF IF : 1.46
SJIF IF : 1.329

An Interacting Automata Based Redundant Modular Code Fault Tolerant Systems in Special Processors for DSP Applications

Dr.K.Srinivasan
Department of Electrical and Electronics Engineering,Shadan College of Engineering and Technology HYD,T.S,INDIA

Abstract - The objective of the work is to propose an Interacting Extended Finite State Machine (IEFSM) model of a fault tolerant system with enhanced fault detection and recovery capabilities applicable for special processors used in Digital Signal Processing (DSP) Applications. The fault or error detection latency is to be minimal and thereby any recovery action can be triggered at the earliest time to minimize the system failure. The state and transition level analysis of the system will try to identify the bad transitions or the bad set of states so that the system can safely recover into safe states immediately when the error has been identified. The faster error detection techniques are analyzed so as to develop a fault tolerant system model using interacting extended linear finite state machines. The focus is not only to identify the faulty states and transitions but also identify the best recoverable states and minimize the recovery time. The proposed model being a parallel interaction of individual FSMs is verified as a Continuous Time Markov chain (CTMC) using PRISM model checker.

KEYWORDS Fault Tolerant, Automata, FSM

1. INTRODUCTION

Hardware fault tolerance mainly focuses the challenge of designing reliable systems from unreliable components. Systems must be protected from a variety of potential faults like permanent stuck at faults or intermittent faults. The classical approaches to fault tolerance are having many shortcomings especially in the phases of fault detection and removal. A framework on the notion of finite-state a machine for describing discrete state and discrete-event systems, which continuously receive inputs from and react to their environment, is considered [1]. In principle, any hardware system can be represented by a set of interconnected finite state machines (FSM). Finite state machines define the acceptable states and transitions between states under normal and faulty conditions. The possible state transitions of the model with the calculus for binary relations can be used [2]. In this present work, an FSM model of fault tolerant system has been proposed that will help in determining the faulty transitions between states rather than the faulty states themselves. An EFSM comprises states, variables and transition among states to
represent complex systems with many guard expressions [3]. The finite state machines that are modeled in this paper can have transitions only if they satisfy a set of guard expressions. Time, hardware and information redundancy provides fault tolerance in a system through self healing, self stabilization and self reconfiguration [4, 5,6]. This model aims at identifying the transitions for safe

landings in any one of the normal states with the help of different categories of redundancy, which would provide a type of forward error recovery. The reliable operation of the system can be divided into three stages like the detection of an error or output deviating from the normal design value, assessment of the resulting effects of the faults and activation of a suitable recovery procedure. Recovery can take the form of one of two methods, either backward error recovery that can return the system to a previously stored valid state or forward error recovery can make selective corrections to the current state until an acceptable state is reached.

2. FAULT MODEL OF A HYBRID FAULT TOLERANT SYSTEM

It is proposed to model the hybrid fault tolerant system as a combination of continuous time model and discrete event model. The system is modeled in synchronous data flow paradigm with a number of finite state machines connected in series and parallel to enhance the reliability using the redundant components at runtime. The architecture of the hybrid fault tolerant system using a number of interacting ELFSMs is designed and the performance of such system is determined in terms of the number of errors detected, corrected and the damage caused by the various injected faults. The hybrid behavior is examined for the absolute timing of interleaved processes in the system, based on the external concurrent inputs. The reliable computing system has fault tolerant mechanism whereby the system timeliness and efficiency are not allowed to degrade considerably by activating the respective redundant components. The hardware and software redundant components and the time and information redundant components are being considered to enhance the system reliability. The forward and backward recovery mechanisms are used to recover from the faulty state.

A fault in a system may be due to an external event driving the system to a recoverable faulty state or a non recoverable faulty state. The control may be faulty due to bad transition without satisfying the guard conditions or untimed transitions. Even the fault may be due to improper unanticipated interaction between various FSMs. Hence the faulty condition may be represented as a tuple, <*state_fault, transition_fault, interaction_fault*> and denoted as < $\mathbf{f_e^i, f_t^j, f_i^k}$ > where i,j and k denote theith occurrence of a state fault due to an event, the jth occurrence of a transition fault, the kth occurrence of an interaction fault. In a continuous time Markov chain, the states are allowed to interact between the two processes out of which one is a series combination of Error Detection(ED), Damage Identifier (DI) and Fault Recovery (FR) FSMs and the other process is the Fault Treatment (FT) FSM for the injected fault. Whenever a fault is detected indicating the successful interaction between any two states of the processes may be treated as a reward in the computational model. The number of such faults in a parallel mode of control in the two processes P_1 and P_2 that can be given by the equations 1,2,3 where m,n,p are the number of states, transitions and interactions respectively.

$$P_1 = \{ ED, DI, FR\}$$
$$P_2 = \{ FT\}$$

$$\sum_{i=1}^{m} S_i \ (ED \wedge FR) = f_e \qquad (1)$$

$$\sum_{i=1}^{n} T_j \ (ED \wedge DI) = f_t \qquad (2)$$

$$\sum_{i=1}^{p} I_k \ (ED \wedge FT) = f_i \qquad (3)$$

3. ARCHITECTURAL MODEL OF INTERACTING AUTOMATA

The proposed system includes hardware component redundancy and a set of reusable software components that will provide the fault tolerance for the system. The blocks in series represent the injected faults, error detection, fault identification, fault recovery FSMs. The finite state machine representation of the above blocks expresses the behavioral model of each and every phase of a fault tolerant system. The parallel block represents the fault treatment FSM which continuously interacts with the series blocks. The fault tolerance feature is incorporated as highly reactive one when the inputs appear intermittently rather than periodic samples. The above architecture was modeled as stochastic and simulated in Prism tool where the transition and interaction behavior is studied as given in Figure 2 and the model results are tabulated in Table 1.

Table 1. Output from the Prism Tool

Step	Time	Tester	S1	S2	Pro 1	Pro 2
0	0.0	0	0	0	0.0	0.0
1	0.335267	1	0	0	0.0	5.0
2	0.347809	1	0	1	5.0	0.0
3	1.1438	1	1	1	5.0	0.0
4	1.41618	1	2	1	0.0	5.0
5	1.53706	1	2	2	0.0	5.0
6	2.58119	1	2	3	5.0	0.0
7	3.71366	1	3	3	5.0	0.0
8	4.16423	1	4	3	5.0	0.0
9	4.35851	1	5	3	0.0	5.0
10	4.4145	1	5	4	5.0	0.0
11	5.14037	1	6	4	0.0	5.0
12	5.22115	1	6	5	0.0	5.0
13	6.65372	1	6	6	0.0	5.0
14	6.74337	1	6	7	0.0	5.0
15	7.747732	1	6	0	0.0	5.0
16	8.68286	1	6	1	?	?

3.1. ERROR DETECTION AND DAMAGE IDENTIFICATION FSM

Each and every phase is modeled and designed in FSM so that the functionality of that machine will be achieved through proper interfaces like the type of inputs and outputs. Each FSM has two ports; one is the incoming port and the other one is outgoing port. The incoming port links to the transitions that are coming into the FSM and the outgoing port connects the transitions that are going out of the FSM. The port can handle input parameters of type: string, integer, float and double. A transition is enabled if the conditions, which are called as “guard expressions” using the input variables or FSM parameters, evaluate to true. The input variables and their values indicate the status of a particular FSM. In the proposed fault tolerant system, multiple inputs and outputs of different widths are considered in the constituent FSMs. The number of errors detected or reported determines the status of the individual machine.

One of the foremost behaviors that are implemented in the proposed fault tolerance system is the error detection unit, which is expressed as a finite state machine as shown in Figure 1. The machine is initiated to check the occurrence of the unexpected sequence that appears at the input port. The error is injected from Gaussian error source with seed 1, mean 1.0 and the standard deviation 0.15. The time at which the faulty sequence occurs is immediately reported. The necessary time conversion and buffering are also considered so that the user can identify the current status of the system. Considering only two types of error injection sources, the error detection FSM will make transition either to component fault or to design fault and responded to the other subsystems connected. The design of the fault tolerant system will be an iterative process of identifying the possible faults that could affect the system. The types of errors that are being considered in the system are the design errors and the application component errors. The error detection unit is modeled as a simple finite state machine as shown in the Figure 2 which will be triggered between two states after receiving the input as an exception thrown from the underneath hardware.

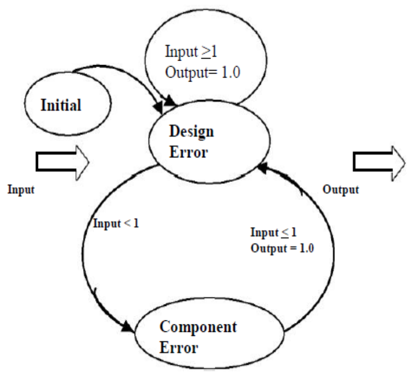

Figure 1. Error Detection FSM

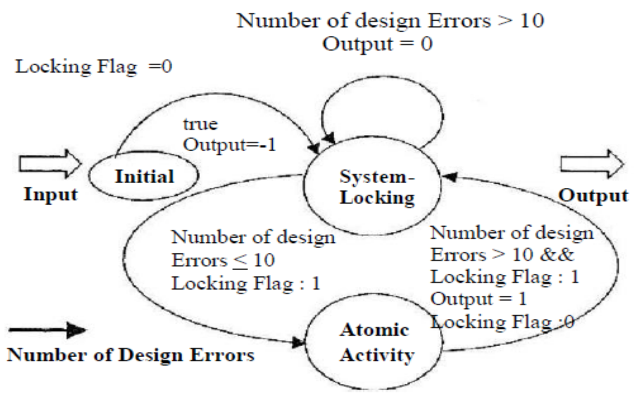

Figure 2. Damage Identification FSM

In the Error detection FSM, the initial state having an outgoing transition with the guard expression equals to true is introduced due to the initialization of parameters. If the input value coming through the Input port of 1 bit width is 1 and the guard expression is Input 1, then the output action is represented as Output = 1 and the value is sent to the Output port. The transitions can be declared as preemptive or non-preemptive, that is the current state of the FSM is to be changed for successive iterations or not. The transitions are also having a reset parameter to initialize the destination state. In the discrete event systems, each and every event has both a value and a time stamp and this model executes the eldest input event first.

If the Gaussian Error Source with the mean =1.0 and standard deviation = 0.15 injects its output, then the error detection FSM reads the injected value and checks whether it is greater or less than 1. The output action of the FSM leads to Design Error state or Component Fault state depends on the value read at the Input port. For example, if the value injected is 1.013436411761 on its second iteration, the FSM detects the value and its time of occurrence through its Input port and remains in the Design Error state. If the value is 0.7342017685387, then the Error Detection FSM makes a transition to Component Fault state and reports at its 35th iteration of checking the input. The time duration between the last failure state and the current one which expresses the non-availability of the system has been incorporated in this model. Once the injected error has been detected, the proposed model counts the number of design errors and it is assumed that if the number of design errors were more than the accepted one for the system maximum performance, then the system will be locked to avoid further failures. A Locking Flag bit that indicates the overflow of the number of design errors is set to 0 and an error counter is implemented using an add-subtract unit with a sample delay and Feedback connection.

The error identification unit represented as a damage identifier FSM shown in Figure 4 checks the input and the expected output of the system to determine the nature of errors occurred. The damage identifier FSM has two states: system-locking state and an atomic activity state. The Damage Identifier FSM allows the atomic activity of any previous process to get completed with a sufficient delay if the numbers of design errors are less than a specific tolerable value; otherwise the FSM sets the Locking Flag to 1are considered and the appropriate switching action is modeled. The constraints on the transitions in the finite state machine model of the fault recovery unit are specified in the FSM and the transitions are achieved only when those constraints are satisfied.

3.2. FAULT RECOVERY AND FAULT TREATMENT FSM

To recover from the above faults, a Fault Recovery FSM unit is initiated with the input from the Fault identification unit. The two values, one from the Serial fault input and the other one from the output of the Fault identification unit will determine the behavior of the Fault Recovery Unit (FRU) and its behavioral model is shown in Figure 3. The recovery unit FSM incorporates all the necessary redundant components so that the system is able to recover as quickly as possible. Forward recovery, backward recovery, check point and audit trial techniques are considered and the appropriate switching action is modeled. The constraints on the transitions in the finite state machine model of the fault recovery unit are specified in the FSM and the transitions are achieved only when those constraints are satisfied.

The Fault Treatment FSM after initialization changes its state when the guard expression evaluates to true. For example, when the guard expression is “Fault Treatment” Input equals to 1 AND the input value from the Input port is equals to the value 1”, then the output of the FSM is 0, it means that the fault has been treated. If the Fault Treatment Input value is more than 1, then the FSM reports that the fault is being treated. After a sample delay, if the process of correction or the activation of appropriate redundant unit has been completed, then the FSM changes its state back to the default state, i.e., Fault Treated state as shown in Figure 4.

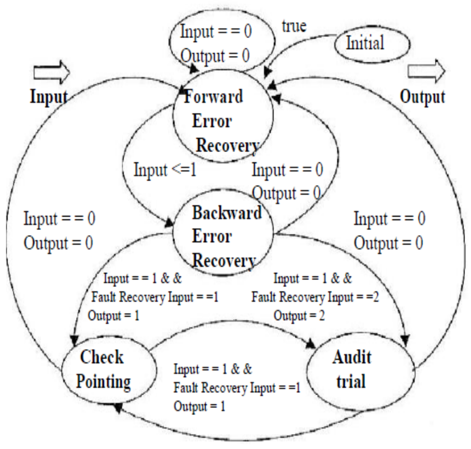

Figure 3. Fault Recovery FSM

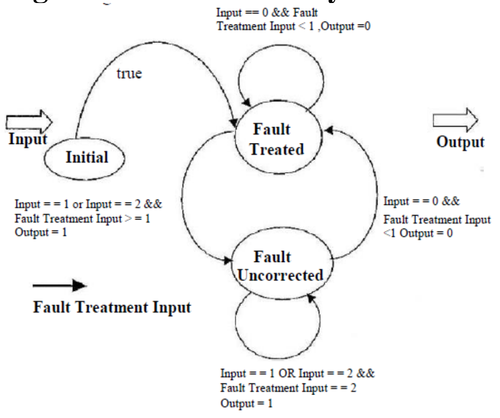
Input == 0 && Fault Treatment Input < 1 ,Output =0
true
Input
Initial
Fault Treated
Output
Input = = 1 or Input = = 2 &&
Fault Treatment Input > = 1
Output = 1
Input = = 0 &&
Fault Treatment Input
<1 Output = 0
Fault Uncorrected
Fault Treatment Input
Input = = 1 OR Input = = 2 &&
Fault Treatment Input = = 2
Output = 1

Figure 4. Fault Treatment FSM

4. HARDWARE AND SOFTWARE REDUNDANCY

The selection of the proper recovery mechanism and the verification of the entire recovery action with time stamps using hardware and software redundancies are considered in the hybrid fault tolerant system. The activation of the suitable redundant components either in the form of hardware or software is modeled as two different FSMI as shown in Figures 7 and 8 respectively. From the higher level of modeling, the implementation of the hybrid system incorporates both active and passive hardware redundancy states, i.e., the redundant components are activated or the maintenance actions are carried out. In case of input equals to 1, the FSM intimates for an operator assistance or produce a warning signal. If the value at the Input port is 0, then the system activates a redundant or a spare hardware unit.

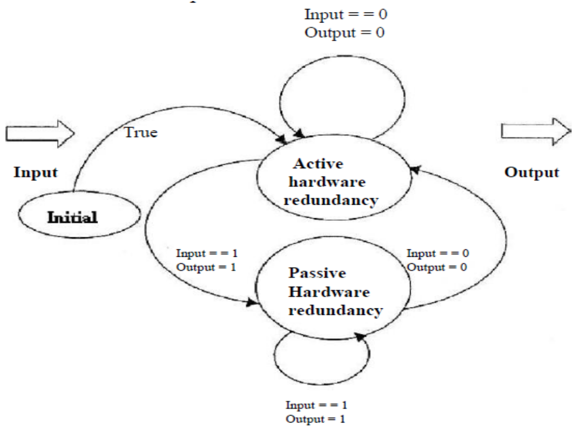

Figure 4. Hardware Redundancy FSM

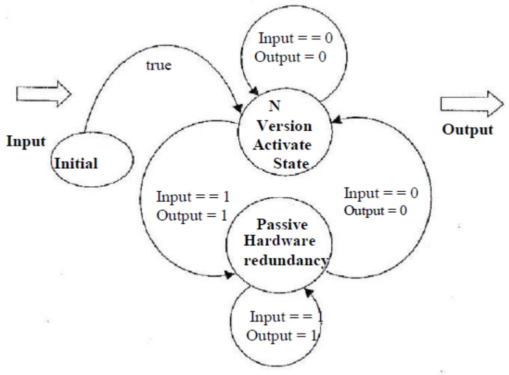

Figure 5. Software Redundancy FSM

The activation module in the case of software faults is modeled as an FSM with two states. The system activates the Recovery blocks when it reads a value of 1 at the Input port of the FSM, which is shown in Figure 8. The system is brought to the normal operable condition if it reads a value of 0 at the FSM Input port while running by recovery blocks.

5. RESULTS

The interacting behavior of the FSMs is examined for absolute timing of interleaved processes in the system, based on the external inputs. The various responses are found to be highly reliable and fail safe states are analyzed. The performance of the FSMs used in the hybrid fault tolerant system model is tested with respect to the error detection and activation of the dynamic redundant components to enhance the overall system reliability in the presence of faults. The performance results are given in Table 2 and the error detection performance is shown in Figure 6. The error detection and identification along with their containment are measured in terms of the system clock periods.

Sample Output of the System

Parameters:	
Vectorization factor	1
	15
Iterations	0

Table 2. System Performance

Nature of Fault	**Time of Detection(ms)**	**Fail Safe State**	**Recovery Action**
Design error	1.0001113613499	System locking	Forward recovery
Design error	1.01343641761	System locking	Forward recovery
Design error	1.0353888501563	System locking	Check pointing
Design error	1.1111515907139	System locking	Check pointing
Design error	1.3889271372763	System locking	Forward recovery
Component fault	0.565940655521	System locking	Forward recovery
Component fault	0.9013635199457	System locking	Check pointing
Component	0.7342017685387	System locking	Forward recovery
Component fault	0.927339233 0601	System locking	Forward recovery recovery
Component fault	0.9181239491367	System locking	Forward recovery

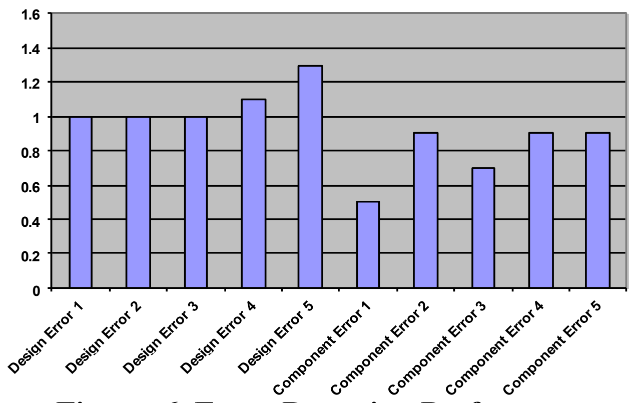

Figure 6. Error Detection Performances

The interacting FSMs in the enhanced architecture are able to provide the essential features of a hybrid fault tolerant system, like error detection, identification, redundancy activation and recovery with minimum number of states and transitions.The model also incorporates specific number of conditions, different types of fault parameters and utilizing various forms of redundancy. Thus the model gives a quicker and correct response to injected faulty data thereby enhances the error detection and location capabilities of the hybrid system.

6. CONCLUSION

The system with fault detection and location can improve the overall reliability of the system. Since there are number of hardware and software components that will interact in the system, the interaction between them is the main focus and it can be measured through the reliability importance of each and every component. The individual FSMs are checked with random inputs and the time of detection and the nature of redundant component activated are noted. The model is verified as a stochastic model and the results are tabulated. With the minimum complexity and cost of additional overhead in the system, the fault tolerant system can be controlled for achieving fail safe condition for any type of errors.

7. REFERENCES

[1] Dal Cin.M, "Verifying fault-tolerant behavior state machine", IMMD3,International Report,97/1.
[2] Brink.C., Kahl.W and Schmidt.G,"Relational Methods in Computer Science",Springer Wien Publications, 1997.
[3] Abdulsalam Kalaji,Robert M Hierons and Stephen Swift ," A Search-Based approach for Automatic Test Generation from Extended Finite State machine",IEEE,2009.
[4] Levin I.,Ostrovsky V., et al., " Self checking sequential circuits with self healing ability " Proceedings of the 12-th ACM Symposium on VLSI,Newyork, 2002., 71-76.
[5] A. Dhama, O. Theel, T. Warns, "Reliabilty and Availability Analysis of self – stabilizing systems, Proceedings of SSS 2006, pp.244-261.
[6] D.Buel, „High Performance reconfigurable Computing", IEEE Computer, March 2007.