Fault Tolerant System for Embedded System Architecture

Namita Arya, Amit Prakash Singh
University School of Information & Communication Technology,
Guru Gobind Singh Indraprastha University, Dwarka, New Delhi
Email: aryanamita12@gmail.com, amit@ipu.ac.in

Abstract: This paper is a study based paper of an accession to the conception fault-tolerant embedded systems architecture in which we mingle the hardware and software fault resistance methods. There is an efficient interpose between crystallization in hardware and system reimplementation in software to have a maximum fault tolerance system with lowest possible cost possible. This paper is an approach towards the building of such a fault tolerant system which has maximum crystallization in hardware and possible attempts of execution at software level at a minimum of possible cost. This system should have maximum possible reliability characteristics which can satisfy all the conditions to have a fault tolerant error free system. In the reference, we operate re-execution concept in operations to avoid or tolerate temporary fault like transient faults in software and this is called time redundancy concept.

Keywords: Fault tolerance, Embedded System, Types of faults

I. Introduction

A system may fail if it cannot meet its promises well. This failure is a result of an error in system elements. Systems are called fault-tolerant system when they perform with its particularization even in the existence of errors. There is no failure in the system as the system is fault tolerant system [3].

A fault tolerant system is an important aim to have a much fault-free application like medical, aviation, banking, and engineering. The main concept to have a fault tolerant system is a redundancy. A redundant parts or component in a system can perform longer even in the presence of faults in the system. A system can easily perform of remained same if it has some extra or spare part to perform if any part of the system is damaged.

A. Definition

Your system level requirement defined the level of fault tolerance operations as what should be the actually admissible performance in the presence of the fault.

“Dependability” is the main approach in respect to the fault tolerant system. This can be characterized by the following number of headings. This can be characterized by a number of headings:

B. Difference between Fault-Intolerance and Fault- Tolerance System

We can also call fault intolerance as fault avoidance method. A system can be more reliable by this approach of fault avoidance or fault intolerance. The reliability concept removes all the possible sources of errors which are a presence in the system before its performance at hardware or software level [9].

We must introduce some techniques and methods in reference to the failures before introducing the Embedded System Architecture [1].

An element is known as faulty once its performance is not performing with its actual behavior. In respect of theses, we introduce two respective classes of faulty performance.

In the year 1982 author exhibited that due to faulty circuitry, the performance can be erratic. Byzantine Failures: the circuitry can show erratic and faulty performance, perhaps including complicity with other faulty circuitry in the system. [19].

Earlier in 1984, Schneider talked about Fail-stop Failures: In respect to the failure to the circuitry the element or the part of the system move to the state that allows another part of the system to identify the error present in the system that prevents the failure of the whole system by stopping the operation [21].

Byzantine failures are not a very good solution for the failure in the circuit. It is a risky solution for the circuit run and sometimes all the assumption may not satisfy by this approach. Moreover many applications use another concept as Byzantine fail-stop failures.

In year 1982 according to Siewiork and Swarz, A system which subsist many components in the circuit is a more reliable system than any other circuit. Run the system can perform best in Maximum interval of circuit because of its distinct components.

That system is called fault tolerant system. Fault tolerance generally has been stated as mean time between failures (MTBF).That stand for the probability of failure over distinct intervals, and many other analytical operations. [22].

I. Fault Model

Types of Fault

Transient Fault: Transient fault is the fault which arises once and at the couple of seconds it disperses. This means that the fault dissipate due to some environmental conditions.

Intermittent Fault: Intermittent Fault is the fault which arises and become invisible again appears but does not follow any peculiar path. This category of fault is worst kind of fault because it is difficult to find. This fault may appear due to present of poor connections in the circuitry.

Permanent Fault: Permanent fault is the fault if it appears then there is the only one solution as removal or reinstatement of the components so it can perform longer. This type of Fault if occurs then only the replacement or repair of a faulty component will allow the defected system to function normally. Means the fault persist until it is removed, repair or replaced by another component.

A fault can be

1. Hardware fault: It includes glitches at hardware level. Hardware level includes components like processor communication line, switch, etc.

2. Software fault: it includes faults at software level. There is a combined method named as design diversity which combines hardware and software fault-tolerance. This approach implements such a computer which includes many different redundant components at different channels.

Each channel is designed for some specifications and the approach detects the faulty channel if it cannot perform according to its specifications. The main aim is to provide a system with both software and hardware level fault tolerant system.

II. Redundancy:

A. Time redundancy: Time redundancy is the concept in which the programming is set to rerun the programmed until the faulty circuit is not recover.
B. Hardware redundancy: Hardware redundancy provides more hardware circuitry in the system to make the system more redundant.
C. Software redundancy: The basic concept of the software redundancy is to provide distinct software with distinct versions; if one version fails another can handle the rest of the system without the failure of whole system.
D. Information redundancy: Information redundancy provides certain data codes for the system and if the system found error in bits it can be detected or corrected by the same data codes. These are of many types like parity coding, checksum codes, cyclic codes.

III. Hardware Recovery

The main aspect of fault tolerant design is provided with default programming computers for recovery. If random faults appear in the system it will automatically recover the system.

A. Passive (static): Passive recovery provides fault masking for fault recovery. In this no extra effort is necessary by the system.
B. Active (dynamic): The main concept of active recovery is to compare of results for detection of faults. It directly removes the faulty components from the system.
C. Hybrid: It combines both approaches active and passive until detection is not complete. It is much expensive but more reliable technique for making system fault tolerant.

IV. Hardware redundancy uses:

Basically hardware redundancy is to provide extra parts or hardware components to the system for making system more redundant.

A. Fault detection, correction, and masking: There is the number of hardware components provided for the same specific task in the parallel session and by the end of the session results can compare to remove fault.
B. Detection: A system shows disagreement in the result if a part of the system is faulty but rest part of the system is performing well .This is the best way to detect fault in the system by its actual result variation.
C. Correction and masking: If a small portion of the system is faulty and the large one is error free then the majority result can be used to make system fault free.

Replacement of malfunctioning units:

Correction and masking are the limited features for the system to make it fault tolerant. In these the level of fault tolerance is defined by restore the actual system or replacement of the faulty components.

1. Mistakes in specification or design: these mistakes can at both level in hardware and software.

2. Defects in components: Hardware faults are occurring due to defects at manufacturing process.

3. Operating environment: hardware faults can be the cause inimical surroundings like: temperature, radiation, vibration, etc.

V. EMBEDDED SYSTEM ARCHITECTURE

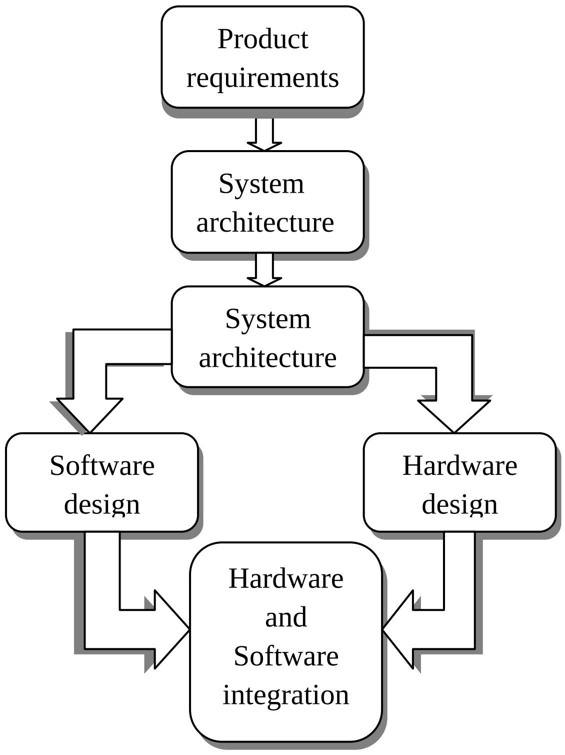

Figure1: Embedded system design and Development

Embedded system can be design for fault tolerance system in VLSI by blending the design process.

VI. Fault Tolerance Techniques in Embedded Systems Architecture

A. The fault diagnosis and fault tolerance are part of the software architecture of the system..
B. The software architecture, mainly include the process of Re-execution. If a system found faulty firstly the fault is discovered then the process will re-executed.
C. Re-execution process restores all the initial inputs to remove fault.
D. We use following two techniques for tolerating faults at software level.
E. Rollback Recovery with Check pointing

VII. Active replication.

A. Rollback Recovery with Check pointing

Rollback recovery technique reduces re-execution time to detect fault in the system. The approach is based on to find the best and last error Free State in the execution and then restores that sate to prevent whole system failure.

The last fault free state or the check point has to be saved in static memory for backup so that if the system occur any failure it can be resorted by the static memory thus the system can be prevent by the failure.

B. Active and Passive Replication

The main disadvantage of recovery method is that they are restricted to analyze redundant scope of applicable computation nodes. On the other side roll back recovery, re-execution, and active and passive replication methods can handle redundant scope basically replication provide distinct parallel session of the system to recover the fault present in the system.

C. Transparency

Transient faults can be adjusted only by the dynamically approach to prevent from the system failure. The system operations are proportionally equal to the number of tolerated transient faults. In reference to correct, test, or adjust the circuit all its processing specifications have to be taken into narration. Therefore, debugging, verification and testing become very difficult. An effective solution of this difficulty is transparency.

A. Product requirements for transparency
B. System architecture
C. System architecture Software design
D. Hardware design
E. Hardware and Software integration

VIII. Dependability Concept Classification

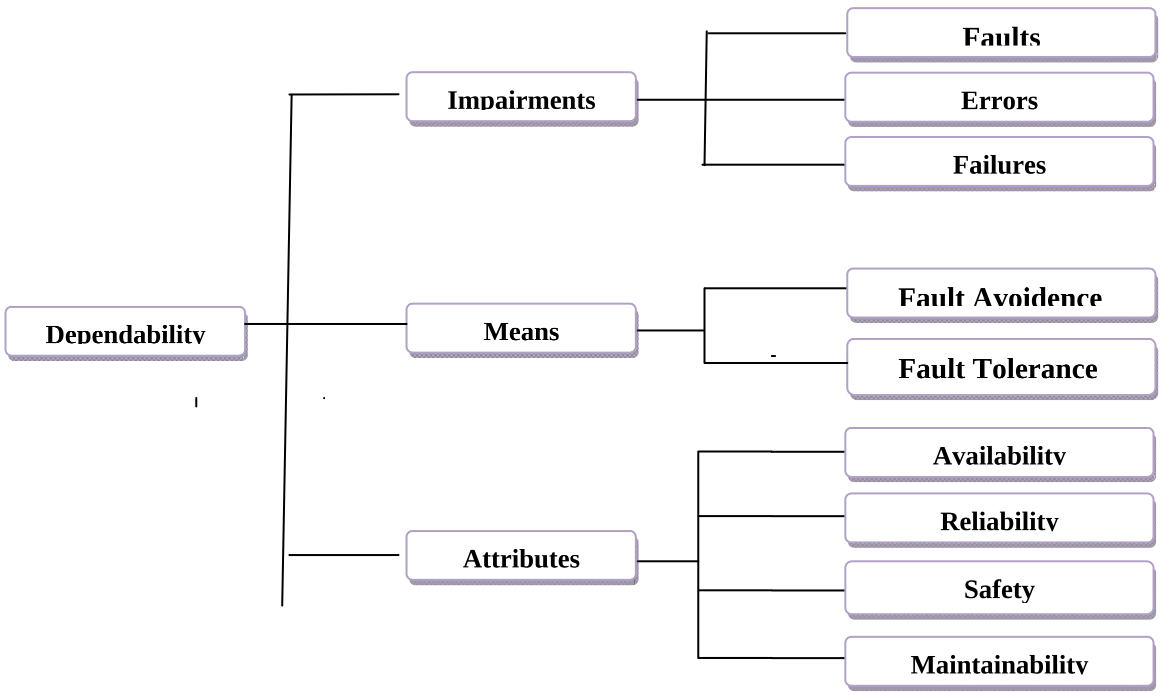

Figure2: Dependability flow chart

IX. CONCLUSION AND FUTURE SCOPE

The best advantage of tools and experience can be observed in the single stuck-at fault model. Stuck-at fault tests cover many other faults like multiple stuck, stuck- open and bridging. .Special tests are required by technology dependent faults, delay faults and stuck-short faults. Different types of test models are present in the testing world to analyze fault tolerant model especially in analog and memory circuits.

A system can be a fault tolerant system if it is designed for the specific test sets. As new advancements of the circuits, the system requires new approaches and technologies for fault testing and diagnosis. There are many varieties of the circuit testing due to the combination of hardware and software of the system. Thus this is the advance research topic to make a fault tolerant circuit with the combination of hardware and software circuit which is called embedded system architecture.

There is a novel approach to handle fault tolerant system with maximum fault toleration degree to stick with stuck at fault models. Fault-tolerant research already present in every field as in control development, transportation, electronic commerce, space, communications and many other areas that truly bang our lives.

X. REFERENCE:

[1] F. B. Schneider, “Implementing Fault-Tolerant services using the state machine approach: A tutorial”, ACM Computing Surveys, vol 22, No. 4, December 1990
[2] F. B. Schneider, “The State Machine Approach: A tutorial”, pp 18-41, Springer, 1990
[3] David A. Rennels, “Fault -tolerant computing :encyclopedia of computer science” pp 698-702, John Wiley and Sons Ltd. Chichester, UK
[4] Krishnendu Chakrabarty ,” VLSI System Testing: A Book,” Section 4.4 (pp. 60-70), Spring 2011-2015
[5] A. D. Singh “Interstitial Redundancy: An Area-Efficient Fault-Tolerance Scheme for Larger Area VLSI Processor Arrays", IEEE Trans. Computers, vol. 37, no. 11, pp.1,398 -1,410 1988 .
[6] Anderson, T., and R. Kerr, “Recovery Blocks in Action: A System Supporting High Reliability”, Proceedings of the Second International Conference on Software Engineering, 1976, 447-457.
[7] Avizienis, A. and J.P.J. Kelly, “Fault Tolerance by Design Diversity: Concepts and Experiments”, IEEE Computer, vol. 17 no. 8, August 1984, 67-80.
[8] Avizienis, A., et al., (Ed.). Dependable Computing and Fault-Tolerant Systems Vol. 1: The Evolution of Fault-Tolerant Computing, Vienna: Springer-Verlag. (Though some what dated, the best historical reference available.)
[9] Wenbing Zhao, “Application-Aware Byzantine Fault Tolerance Dependable”, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference, Year: 2014,pp. 45-50.
[10] A. Aviˇzienis and J.-C. Laprie. Dependable computing: From concepts to design diversity. Proceedings of the IEEE, 74(5), 1986.
[11] A. Azagury, D. Dolev, G. Goft, J. Marberg, and J. Satran. Highly available cluster: A case study. In1994 IEEE 24th International Symposium On Fault-Tolerant Computing, pages 404–413, 1994.
[12] W. G. Cuan. Fault tolerance: Tutorial and implementations. Computing Futures (supplement to IEEE Computer November 1989), 22(Inaugural issue), 1989.
[13] Y. Huang and C. Kintala. Software implemented fault tolerance: Technologies and experience. In 1993 IEEE 23th International Symposium On Fault-Tolerant Computing, pages 2–9, 1993.

[14] Y. Huang and C. Kintala. Software implemented fault tolerance: Technologies and experience. In 1993 IEEE 23th International Symposium On Fault-Tolerant Computing, pages 2–9, 1993.
[15] N. Kandasamy, J. P. Hayes, B. T. Murray, "Transparent Recoveryf rom Intermittent Faults in Time-Triggered Distributed Systems", IEEE Trans. on Computers, 52(2), 113-125, 2003.
[16] S. Punnekkat, A. Burns, R. Davis, "Analysis of Check pointing for Real-Time Systems", Real-Time Systems Journal, 20(1), 83-102, 2001.
[17] Ying Zhang and K. Chakrabarty, "A Unified Approach for Fault Tolerance and Dynamic Power Management in Fixed-Priority Real-Time Embedded Systems", IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, 25(1), 111-125, 2006
[18] A. M. Álvarez, Felipe Restrepo-Calle, Luis Alberto Vivas Tejuelo, Sergio Cuenca-Asensi, Fault tolerant embedded systems design by multi-objective optimization, Expert Systems with Applications, Volume 40, Issue 17, 1 December 2013, Pages 6813-6822
[19] Leslie Lamport and S. Merz,"Specifying and Verifying Fault-tolerant System", Proceedings of the Third International Symposium on Formal Techniques in Real Time and Fault Tolerant Systems.
[20] Viacheslav Izosimo, V., "Analysis and Optimization of Fault-Tolerant Embedded Systems with Hardened Processors", Design, Automation & Test in Europe Conference & Exhibition, 2009.
[21] Schneider , "Implementing Fault-Tolerant Service using the state machine", ACM Computing Surveys, Vol. 22, No. 4, December 1990.
[22] Siewiorek and R.swarz ,"Reliable computer system : Design and Evaluation" MA : Digital Press, c1982.
[23] A. Sung and B. Choi, "An Interaction Testing Technique between Hardware and Software in Embedded Systems", Proceedings of Ninth Asia-Pacific Software Engineering Conference, 2002. 4-6 Dec. 2002 Page(s):457 – 464