

Reconfigurable Triple Modular Redundant and N-Modular Redundant Systems with Variable Reliability in Multi-Processor Environment

Sachin Aithal*, S. Krishna Kumar†

*Student, Department of ECE, National Institute of Technology Surathkal, India
Email: sachinaithal@gmail.com

†Senior Research Scientist, SJB Reaseach Foundation, Bangalore, India
Email: skkumar@accsindia.org

Abstract—Voting Logic (VL) is an important component of Triple Modular Redundant (TMR) and N-Modular Redundant (NMR) systems. A number of voting logic designs are presented in this paper. The Design Profile, Diagnosability and Reliability calculation of a word voter for TMR system is presented.

The notion of reconfiguring a 4 processor design for an “unconventional” TMR and 4MR mode is introduced. Reconfiguration is implemented on the fly using a “reconfiguration instruction” with the “ON” operand in the normal program code. Processors and VL for TMR are chosen using metrics such as millions of instructions executed for processors and reliability of the VLs. The TMR or 4MR configuration can be “dissolved” by the same reconfiguration instruction with the “OFF” operand.

The major contribution is the Dynamic Reconfiguration of the processors at run time for enhanced Reliability (unlike all other systems where Reliability is always a decreasing function of time). The second contribution is that TMRs are built and dissolved during normal operation, unlike original TMRs which are hard wired for lifetime operation.

Index Terms—Multi-processors systems, Reliability, TMR/NMR systems, Reconfiguration, Design Profiles, Diagnosable Voters, Totally self-checking circuits.

I. INTRODUCTION

Multi-Processor Systems (MPS): MPS can be broadly classified as Multiple Instruction Multiple Data stream (MIMD) or Single Instruction Multiple Data stream (SIMD) systems [1]. Multi-Processors System-on-chip (MPSoC) [2], with Network-on-Chip (NoC) infrastructure is a variant of MPS. It is now an important, well known and useful class of VLSI circuits, but does not belong in the SIMD or MIMD classification necessarily. The abundance of processors in SIMD, MIMD and MPSoC can be used to increase the reliability of computation at run-time. In this paper, a four processor system similar to QuadroCore [Figure 5.8 of [8]] is used for simplicity. The techniques derived in this paper can be used in ICs as well as PCBs.

Fault classes: A Fault is a hardware defect or a programming mistake [5]. Only hardware faults are considered here. An error is the manifestation of a fault. Various types of malfunctions have been identified in the literature. These

include intermittent failures, transient failures, soft errors, rapid fluctuations in power supply voltages, burst noise and common mode failures. Only single and multiple stuck-at permanent faults are usually considered.

TMR and NMR: Triple Modular Redundancy (TMR) and N-Modular Redundancy (NMR) are widely used redundancy techniques in the design of highly dependable systems [9,5]. A wide variety of applications deploying such techniques can be found in [4]. In traditional systems, the TMR/NMR modules and voting logic modules are “hard-wired” throughout the operational lifetime of entire system. Such hard-wiring results in poor reliability in the later stages of system operation. This paper provides the technique to configure and deconfigure TMR or NMR systems on-the-fly, so as to increase/lower the reliability as much as is desired and possible.

Reliability $R(t)$: Reliability is usually a decreasing function time. In this paper, we show that in a multi-processor environment, the value of reliability can be changed by changing the constituent processors.

The reliability of a single component (Simplex System) $R(t)$ can be shown to be $R(t) = e^{-\lambda t}$ [5], assuming a constant failure rate λ . Generally, the Mean Time to Failure (MTTF) is

$$MTTF = \int_0^{\infty} R(t) dt \quad (1)$$

Hence the value of

$$MTTF_{simplex} = \int_0^{\infty} R(t) dt = \frac{1}{\lambda} \quad (2)$$

$$MTTF_{TMR} = \int_0^{\infty} (3R^2(t) - 2R^3(t)) dt = \frac{5}{6\lambda} \quad (3)$$

Hence,

$$MTTF_{simplex} > MTTF_{TMR}. \quad (4)$$

Thus the MTTF of a hard-wired TMR system is less than that of a corresponding single module system. Also, for mission times greater than 0.7 times MTTF of simplex system, the TMR/NMR reliability is lower than the reliability

of the corresponding simplex system.

It can be shown that by picking processors with the highest current reliability and highest reliability VL and then configuring them as TMRs and NMRs in the MPS environment addresses problems related to hard-wired systems.

Voting Logic (VL): Almost all systems using hardware redundancy use majority voter circuits to determine the correct output. The basic assumption is the existence of a single permanent fault causing a functional failure in one and only module (in case of a TMR system) and also that there are no faults in the VL itself. Such assumptions about VL are quite suspect in nanoscale technology, given transistor density and variability.

Voters can be classified as Word Voters or Bit Voters. Majority bit voting [10] is the most basic voting scheme with the output being the majority among the ‘n’ input bits to the voter (also known as m-of-n bit voter). A Sum Of Products (SOP) implementation of an m-of-n bit voter for large values of n appears to be infeasible because of the fan-in constraints. For a TMR system with n outputs, a bit-wise voter requires 5n 2-input gates [7]. Bit by bit comparison of words can yield incorrect but legal outputs. Word Voters [7] take all the bits into account in parallel to determine the final output.

Voters can also be classified into Distributed voters [6] or Centralised voters. Centralised voters suffer from a “single point of failure”. This voter [6] is also called Distributed bit voter and can overcome multiple failures up to $\lceil \frac{n}{m} \rceil$ failures.

Design Profile: A profile of a design is the characteristics of an instance of its implementation. This includes the value of certain metrics like area, speed, latency, power dissipation, energy, etc. It is recommended that the reliability of a module and the reliability of VL schemes also be part of the design profile. There may be many design profiles for a given design. Xilinx FPGAs and Xilinx ISE 12.4 are used in this paper for profiling.

II. DESIGN AND ANALYSIS OF VOTERS

In this section, new and improved voters are presented. Diagnosis, not discussed in the literature in this context is also given for every voter. Important enhancements and profiles to the Word Voter [7] are included for completeness.

A. Enhanced Exact Word Voter

Since, Bit by bit voting on word outputs can lead to incorrect but legal word outputs, the Exact Word Voter [Figure 2.2 of [7]] had been developed earlier. It has other advantages also like handling common-mode failures. One disadvantage is that it is a Centralised voter. The design can be improved by deploying Totally Self-checking Circuit (TSC) in this voter to prevent faults from remaining undetected in the VL.

- 1) As far as we know, the Design Profile of this voter is presented in this paper for the first time.

TABLE I
RESOURCE UTILISATION AND DELAY FOR DIFFERENT WORD LENGTHS OF EXACT WORD VOTER

	8-bit voter	16-bit voter	32-bit voter	64-bit voter
Slices	13	25	54	99
LUTs	24	48	100	192
IOBs	33	65	129	257
Delay(ns)	10.5	11.14	13.41	19.47

- 2) An additional finding in the analysis of the VL reveals that many single faults in the VL are safely masked, but some faults will produce incorrect output. A TSC is shown below to avoid this problem.

All single faults inside the word voter [7] are masked except for the outputs called ERROR, Z_1, Z_2, \dots, Z_n . The self-checking circuit for ERROR signal is shown below (Figure 1). All other outputs Z_1, Z_2, \dots, Z_n are also derived similarly. The principle is based on Dual Rail logic. The output combinations that are legal are (0, 1) and (1, 0) whereas (0, 0) and (1, 1) are illegal.

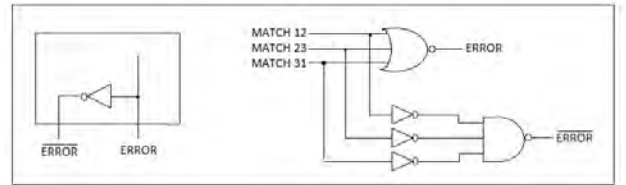


Fig. 1. Self-checking circuit for ERROR signal

- 3) The reliability of this VL is calculated below. In general, it is advised that the reliability become part of the design profile.

If $R(t)$ is the reliability of an individual module (the probability that the module is still operational at time t), for TMR systems the system reliability is given by

$$R_{TMR}(t) = 3R^2(t) - 2R^3(t) \quad (5)$$

If the failure rate of the voter is taken into account then the above equation is modified as

$$R_{TMR}(t) = R_{voter} \{3R^2(t) - 2R^3(t)\} \quad (6)$$

To implement an n bit Exact Word Voter the number of gates required are 3n 2-input XNOR gates, 3 n-input AND gates (equivalent to 3n 2-input AND gates), 2n 2-input AND gates, n 2-input OR gates to generate n bit output plus one 3-input NOR gate to produce an error signal [7]. The number of interconnects in the design is 6n (3n interconnects above the three matching circuits and 3n interconnects above the circuit that generates the output [Figure 2.2 of [7]]). The reliability of this voter with 9n+2 2-input gates (AND/OR/XNOR/NOR) and 6n

interconnects is $R_{g_avg}^{9n+2} R_n^{6n}$ where R_{g_avg} is the average reliability of a 2-input gate and R_n is the reliability of an interconnect. Hence the reliability of TMR system with an n bit Exact Word Voter is

$$R_{TMR}(t) = R_{g_avg}^{9n+2} R_n^{6n} \{3R^2(t) - 2R^3(t)\} \quad (7)$$

B. Centralized Diagnosable Word Voter

The Centralized Diagnosable word voter (Figure 2) is an extension to the previous voter. This voter is capable of module level error detection, i.e. detecting the particular module in error in case a single module of TMR system fails. It is also capable of detecting error in match logic.

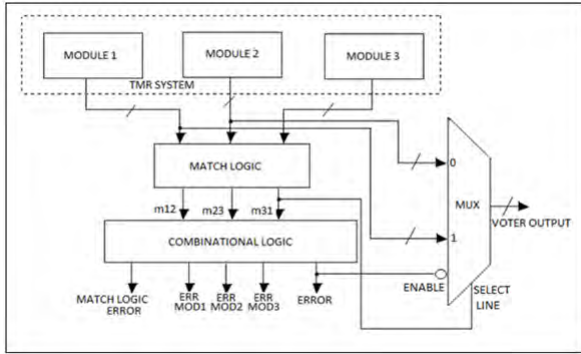


Fig. 2. Centralized Diagnosable Word Voter (using Xilinx 12.4 ISE with virtex4)

TABLE II

RESOURCE UTILISATION AND DELAY FOR DIFFERENT WORD LENGTHS OF CENTRALIZED DIAGNOSABLE WORD VOTER

	8-bit voter	16-bit voter	32-bit voter
Slices	16	24	45
LUTs	28	45	85
IOBs	37	69	133
Delay(ns)	6.553	6.830	7.539

C. Centralized Diagnosable Sub-Word Voter

This voter (Figure 3) is a derivative of centralized word voter with the matching carried out by ignoring ‘ n ’ bits, $0 \leq n \leq N$, out of the N bit input to the voter. These ‘ n ’ bits, not necessarily contiguous, are left out depending on the requirement. This voter is useful in applications which require a byte/half-word output and to check different control bits. The corresponding ‘ n ’ bits of all the modules are then masked by a mask register and the result is pair wise matched. Based on the output of the matching, the output of the voter is selected. Since the 3 N -bit inputs to the voter need not be exactly the same in order to get a match, output of the voter is the $(N-n)$ bits of the matched input, with ‘ n ’ don’t care bits in the bit range that is ignored. There is no need to worry about these ‘ n ’ bits at the voter output. Hence, at the voter output these ‘ n ’ bits can be set to ‘0’ or ‘Z’.

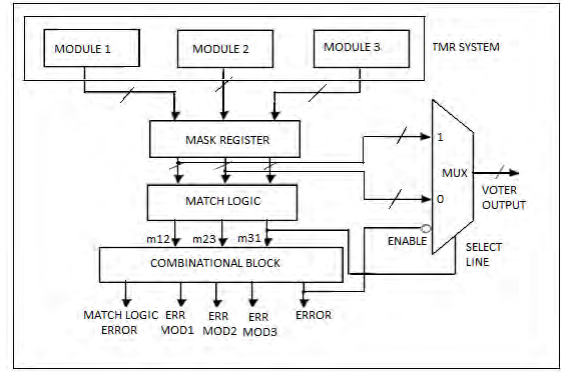


Fig. 3. Centralized Diagnosable Sub-Word Voter (using Xilinx 12.4 ISE with virtex4)

TABLE III

RESOURCE UTILISATION AND DELAY FOR DIFFERENT WORD LENGTHS OF CENTRALIZED DIAGNOSABLE SUB-WORD VOTER

	1 bit masked	8 bit masked	16 bit masked	31 bit masked
Slices	45	35	24	2
LUTs	84	65	45	4
IOBs	130	109	85	40
Delay(ns)	7.502	7.380	7.090	4.946

D. Median Diagnosable Word Voter

As this name indicates the output of the voter (Figure 4) is the middle value of the 3 inputs. Good discussions of Median Voters can be found in [11,12]. A threshold is used to identify and isolate a faulty module in this case. A threshold is the maximum amount of deviation from the middle value that is tolerated before a module is declared as faulty. If any input (to the voter) is out of range then a signal goes high indicating the voter input corresponding to that module is in error. If more than one such signal goes high then an “error” signal becomes ‘1’. The threshold set must be such that most of the failures are detected. Determination of appropriate levels of threshold requires extensive testing.

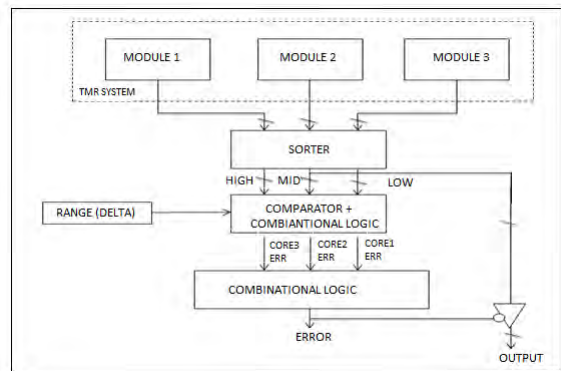


Fig. 4. Median Diagnosable Word voter (Xilinx 12.4 ISE with virtex4)

TABLE IV

RESOURCE UTILISATION AND DELAY FOR DIFFERENT WORD LENGTHS OF MEDIAN DIAGNOSABLE VOTER

	8-bit voter	16-bit voter	32-bit voter
Slices	77	131	239
LUTs	139	263	477
IOBs	36	68	132
Delay(ns)	12.652	13.326	13.436

E. 3-of-4 Centralized Diagnosable Word Voter

This voter (Figure 5) is an extension to the centralized diagnosable word voter described in section 2.B. The voter outputs the majority of the 4 modules involved. To ensure that the system is working correctly, at any time, at least 3 of the 4 modules must be fault free. However, the reliability of this voter is always less than centralized diagnosable word voter.

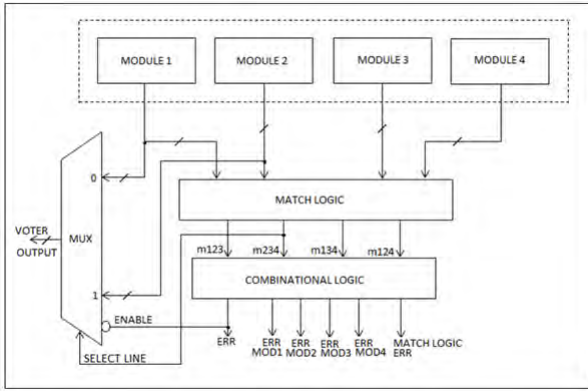


Fig. 5. 3-of-4 Centralized Diagnosable Word Voter (Xilinx 12.4 ISE with virtex4)

TABLE V

RESOURCE UTILISATION AND DELAY FOR DIFFERENT WORD LENGTHS OF 3-OF-4 CENTRALIZED DIAGNOSABLE VOTER

	8-bit voter	16-bit voter	32-bit voter
Slices	35	45	54
LUTs	61	86	102
IOBs	46	86	166
Delay(ns)	7.455	7.172	7.584

F. Observations on Resource utilisation and delay

Generally a broad discussion would include an almost exhaustive Design Space Exploration and many profiles for a specific design. This discussion is however, limited to one profile per design.

It can be observed from the available results that:

- 1) The resource growth of the Exact voter(Table I) is linear relative to the number of bits in the implementation. However the delay characteristics are relatively close for

8/16 bits, but grow non-linearly. Results for 32/64 bits do not scale similarly. This need further investigation.

- 2) More resources are used in the Centralized Diagnosable voters(Table II, Table III, Table V) but only a modest increase relative to the benefits obtained. The delays are less compared to the Exact voter and this requires more investigation as well.
- 3) The Diagnosable median voter is the most expensive both in terms of Utilized resources as well as the delay characteristics(Table IV). Since the real world consists of multiple inputs with a number of minor disagreements, the Median is still an useful scheme, but to be deployed after due consideration.

III. RECONFIGURATION FOR TMR/NMR

Reconfiguration is defined here, as the run-time ability of a system to change itself. Generally the change may be in functionality, in performance by addition/removal of compute elements or by parametric changes such as voltage/frequency. Architectural changes such as switching between SIMD and MIMD via compile time analysis are discussed using the QuadroCore [Figure5.8 of 8].

QuadroCore is a research processor of the MIMD classification with no prebuilt reconfigurability and built with 4 M.Cores. M.Core [13] is a standard single 32 bit RISC processor offered by Motorola. Hence it is considered as a standard processor. In this paper, we propose similar mechanisms to reconfigure QuadroCore like architecture for higher reliability via 2 out of 3 (TMR) or 3 out of 4 (NMR) voters configuration, for a specified duration T of the system run-time. Processors are selected at a given time based on a suitable metric like the total number of instructions executed so far by each processor (aging). Current reliability can also be used as a metric. The processor outputs (end of execute stage of pipeline) are fed to the VL (also chosen). The next time, TMR or NMR is required a fresh selection of the candidate processors is made given the current value of the metric.

A “RECONFIG” instruction, specifically for reconfiguration is added to the instruction set. Reconfiguration is set active by turning the active flag “ON”. The Zones in figure 6 can be thought of as operands related to architectural changes, reliability changes and other changes. In this paper, only reliability changes are discussed. A possible assembly code would be “RECONFIG = ON, RELIABILITY = TMR”. Such a code would be inserted by the programmer in the program code. The deactivation of the TMR would require “RECONFIG = OFF”.

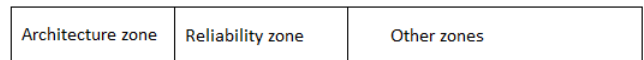


Fig. 6. Reconfiguration Zones or opcodes

Reconfiguration is introduced as an extra pipeline stage (Figure 7). This extra stage in the pipeline contains one or more of the VL schemes that have been discussed earlier.

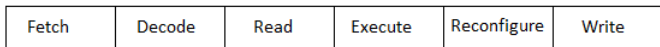


Fig. 7. Reconfiguration as a pipeline stage

The reconfiguration instruction is used to handle interconnects between the decode stage and the execute stage and also between the execute stage and the write stage (Figure 9). During decode stage of a RECONFIG instruction processors are selected for TMR or NMR use based on metrics such as current reliability. The selection of the processors and the metrics is the subject of our next paper.

Reconfiguration mechanism: Let’s say all the processors are working in MIMD mode. A processor fetches a RECONFIG ON instruction from the memory. The instruction is decoded and the processors are selected based on the metrics mentioned above. After the interconnects are changed the processors start working in an “unconventional” TMR mode - a single processor fetches the instruction, decodes it, then reads the registers and sends the values of the registers to the execution units of the processors that are selected. This mode of operation is different from the conventional TMR mode and we call this “triple sub modular redundant” mode. Processors switch back to MIMD mode when a RECONFIG OFF instruction is fetched.

The RECONFIG ON instruction format is shown in figure 8. The RECONFIG ON instruction contains the following details among others:

- The VL which is to be made use of, for voting (2 bits).
- The group of processors that are going to constitute the TMR system (4 bits: P0, P1, P2 and P3).
- The processor which is going to fetch the instructions following the RECONFIG ON instruction (2 bits: S1 and S0).

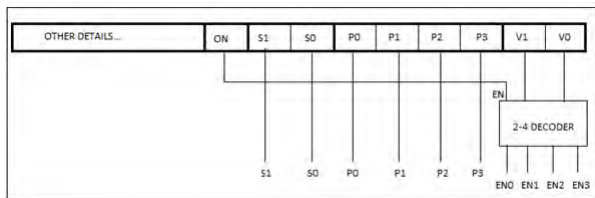


Fig. 8. Instruction format of RECONFIG ON instruction

For e.g. If the group of processors that are selected to operate in TMR mode are 0,1 and 3 with processor 3 doing the instruction fetch then (S1, S0) = “11”; (P0, P1, P2, P3) = “1101”. A suitable VL is selected based on the bits V0 and V1. ‘En’ signals are output from the decoder based on the values of V0 and V1.

NOTE: RECONFIG OFF instruction will have the same format with all the fields set to ‘0’.

Figure 9 shows the complete design of the Reconfigurable Quadrocore system. This Quadrocore can operate in MIMD mode or TMR mode using any of the Voting schemes (VL0: Centralized Diagnosable word voter; VL1: Centralized Diagnosable sub-word voter; VL2: Median Diagnosable word voter; VL3: 3-of-4 Centralized Diagnosable word voter) described above.

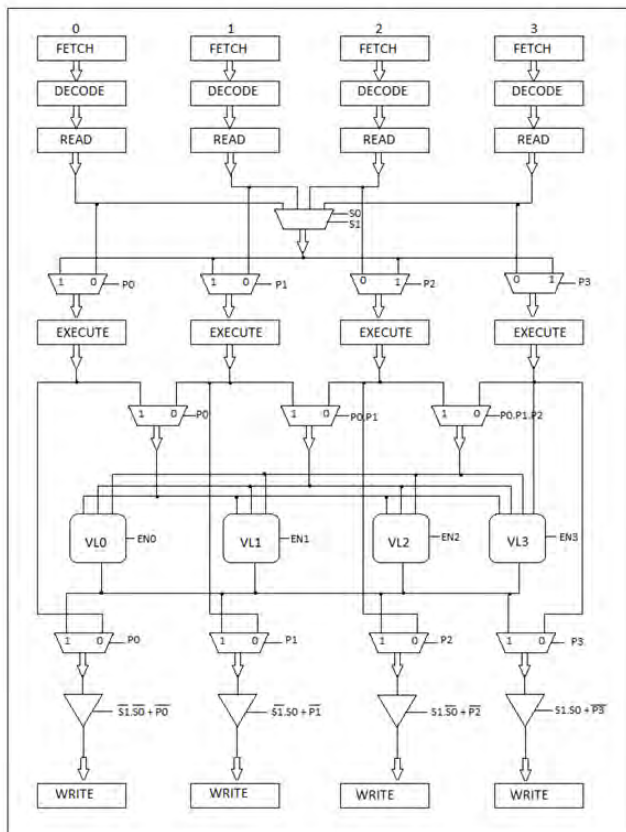


Fig. 9. Reconfigurable Quadrocore showing the usage of different Voting Logic schemes

Features of the proposed reconfiguration mechanism:

- When 3 processors are selected to operate in TMR mode, the fourth processor may continue to work separately (or asynchronously) on its own.
- Among the processors selected to operate in TMR mode, the write back is done only to the register set of the processor that fetches the instruction, since only the execute stage of the other involving processors are made use of in TMR mode.
- Reconfiguration can be used to increase the reliability

of output of certain instructions. (Say in a code, if the outputs of certain instructions are required to be highly reliable then RECONFIG ON instruction can be inserted just before these instructions and hence the instructions that follow are executed in the “unconventional” TMR mode, thus increasing the reliability of the output.)

- Stalls are introduced to other processors when a processor fetches the RECONFIG ON instruction. That is, if processor 0 fetches the RECONFIG ON instruction and if the processors 0, 1 and 2 are selected (based on the metrics mentioned in the paper) to work in TMR mode then stalls are introduced to processor 1 and 2 until processor 0 fetches RECONFIG OFF instruction and everything turns back to MIMD mode. Stalls are also introduced to the 4th processor (in this case processor 3) if it fetches a RECONFIG ON instruction when the other 3 processors are operating in TMR mode.

IV. SUMMARY AND CONCLUSIONS

Concepts from different domains have been used to define a new class of Voting Logic Circuits (VL) called as “Diagnosable voters”. An extension to the classical Word Voter is implemented. The design profile of all the voters is included for different word widths. A small analysis on the resource utilisation and delay of the voters discussed is also presented.

Reconfiguration of MIMD processors into TMR/NMR systems is proposed. This reconfiguration is done on-the-fly and can be nullified with a single instruction. Such a design helps with increasing or decreasing the reliability of the overall system.

Many interesting problems related to implementation of the proposed reconfiguration mechanism, which include selection of processors, introduction of stalls and also realising the complete design on a multi-processor netlist can be studied in future.

V. ACKNOWLEDGEMENTS

We would like to thank Dr. Sumam David for her valuable comments and suggestions.

REFERENCES

- [1] Flynn, M. J. “Very High-Speed Computing Systems,” *Proceedings of the IEEE*, vol.54, December 1966, pp. 1901-09.
- [2] Wolf, W. Jerraya, A.A and Martin, G. “MultiProcessor System-on-Chip (MPSoC) Technology,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Oct. 2008.
- [3] Trivedi, K. S. “Probability and Statistics with Reliability, Queuing and Computer Science Applications,” Prentice Hall, New Jersey, USA 1982.
- [4] Siewiorek, D. P. and Swarz, R. S. “Reliable Computer Systems: Design and Evaluation,” Digital Press, USA, 1992.
- [5] Koren, I. and Krishna, C. M. “Fault-Tolerant Systems,” Elsevier Inc., 2007
- [6] Ali Namazi, A. and Nourani, M. “Distributed Voting for Fault-Tolerant Nanoscale Systems,” *Proceedings of the IEEE*, pp 568-73, 2007.
- [7] Mitra, S. and McCluskey, E. J. “Word-Voter: A New Voter Design for Triple Modular Redundant Systems,” *Proceedings of the IEEE VLSI Test Symposium*, pp 465-70, May 2000.
- [8] Madhura Purnaprajna, “Run-time Reconfigurable Multiprocessors,” Ph. D Thesis, University of Paderborn, January 2010.
- [9] von Neumann, J. “Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components,” *Automata Studies, Annals of Math.Studies*, No.34, pp 43-98, 1956.
- [10] Parhami, B. “Voting Networks,” *IEEE Transactions on Reliability*, Vol. 40, No. 3, pp 380-93, 1991.
- [11] Bass J. M, Latif-Shabgahi, G. and Bennett, “Experimental Comparison of Voting Algorithms in Cases of Disagreement,” *Proceedings of the IEEE*, pp 516-23, 1997.
- [12] Yu, Shu-Yi. “Fault Tolerance in Adaptive Real-Time Computing Systems,” Ph.D Thesis, Stanford University, 2001.
- [13] “M.CORE Reference manual with M210/M210S Specification,” Motorola, Inc