

Transaction Level Error Susceptibility Model for Bus Based SoC Architectures

I.-C. Lin, S. Srinivasan, N. Vijaykrishnan
Dept. of Computer Science and Engineering
The Pennsylvania State University
University Park, PA, 16802
{ilin,ssriniva, vijay}@cse.psu.edu

N. Dhanwada
IBM Electronic Design Automation
Systems and Technology Group
Hopewell Junction, NY
nagu@us.ibm.com

Abstract

System on Chip architectures have traditionally relied upon bus based interconnect for their communication needs. However, increasing bus frequencies and the load on the bus calls for focus on reliability issues in such bus based systems. In this paper, we provide a detailed analysis of different kinds of errors and the susceptibility of such systems to such errors on various components that the bus comprises of. With elaborate experiments we determine the effect of a single bit error on the bus system during the course of different transactions. The work demonstrates the fact that only a few signals in a bus system are really critical and need to be guarded. Such transaction based analysis helps us to develop an effective prediction methodology to predict the effect of a single bit error on any application running on a bus based architecture. We demonstrate that our transaction based prediction scheme works with an average accuracy of 92% over all the benchmarks when compared with the actual simulation results.

1 Introduction and Motivation

Bus based interconnect have been traditionally used as communication channel in System on Chip architectures. However, growing demands for on-chip integration have lead to significant increase in the number of components connected to the bus, raising concerns over architectures of such bus systems. Consequently, bus architectures have evolved over time from single shared bus models to multi layer complex topologies for supporting the growing demands of bandwidth, concurrency and the constrained power budgets. The frequency of operation has also been on a rise, from traditionally under 100Mhz to as high as close to 200Mhz. Commercial buses standards like ARM AMBA [1], PCI Express [3], and IBM CoreConnect [2] has experienced frequency increase by nearly three times over their original designs. Further complexity is introduced in such systems due to different power and performance enhancing features, like operating different buses connected by bridges at different frequencies, voltages etc [16]. Consequently, the probability of errors occurring in such sys-

tem buses increases significantly, which is further aggravated due to technology scaling. The errors may occur due to wide range of causes, including capacitive coupling, soft errors, cross talk, and process variations, as discussed in [11, 15]. Most of such errors lead to transient single-bit errors which may effect the system in many different ways, ranging from creating undetected data corruption to system crashes. The reasons for such single-bit errors and various error protection schemes have been widely studied for different bus architectures.

On-chip communication architecture could be primarily classified into either packet-based communication or bus-based communication as shown in Figure 1. Packet-based communication is employed in different bus protocols like PCI Express and Xpipes [9]. Bus-based communication is adopted in commercial bus standards like AMBA, AMBA AHB [1], CoreConnect and Wishbone [4]. Most packet-based communication system have the reliability models adopted from the traditional packet-based error detection and fault tolerance schemes. PCI Express has all the packets containing CRC bits appended to them at the link layer to ensure a highly reliable data transfer. Similar approaches may not be completely applicable or suitable to the bus-based bus protocols. Various schemes for error detection and correction in bus-based systems are described in [5, 6, 7, 12].

However, with the increasingly stringent power and area budget all such schemes are under reconsideration. [10] provides a detailed comparison of the power and area implications of various error protection schemes. In this paper we particularly focus on the reliability issues in the bus-based protocols. We demonstrate the significance of different control signals of the bus based on the effect of an error in any of those signals. It is however quite important to note that not all the signals in the system are actually very critical for the functionality of the system. [8] presents a vulnerability analysis of various components in contemporary micro architectures. We exploit such an observation for bus architecture under consideration to provide an effective criticality factor for each of the signals of the bus system. We provide a transaction based error characterization model for bus systems. Such transaction based models have been analyzed with respect to power consumption previously in [14].

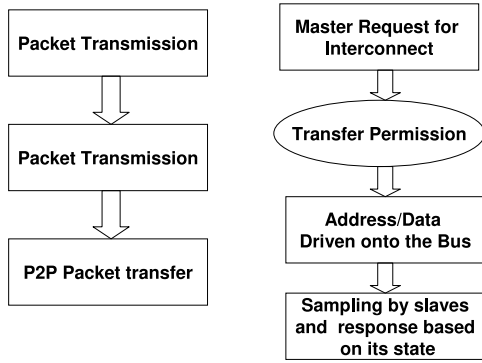


Figure 1. Packet-based vs bus based protocols

In this paper we introduce a novel transaction based reliability model. We provide a detailed analysis of the types and causes of different errors that may occur due to single-bit flips on a bus based system during different transactions. We characterize the susceptibility of AMBA bus on errors in various signals over different transactions. Such susceptibility analysis provides an effective way to characterize the error susceptibility of a bus architecture based on the transactions in the application. The proposed prediction scheme provides results with 92% accuracy as compared to the simulation results on an average for all the applications.

2 Bus Architectures

Figure 2 shows a typical single shared bus architecture. This is the architecture of most of the bus-based systems. The main components in such systems are essentially:

- **Master and Slaves:** The core components that are connected to the bus. Typically the masters are processing cores, DMA controllers etc, and the slaves are memories, bridges and peripheral devices.
- **Interconnect Structure:** The logic that deals with the transfer of the data. Most of the bus control follows state machine based logic. This primarily comprises of the arbiter, decoding logic and the various control signals to control the bus protocol.

In our analysis we deal with errors on the interconnect structure which in turn includes the errors occurring in the masters and the slave ports connected to interconnect. These are the typical components of any bus-based architectures. We have provided an comprehensive error analysis for the AMBA bus architecture. Moreover, our model and characterization can be extended to any bus architecture.

3 Error Characterization Model

In order to develop a generic characterization model we need to focus on the following issues:

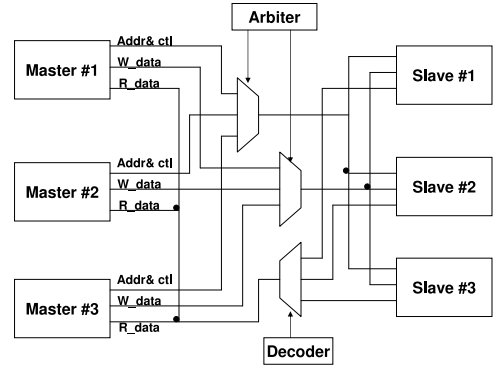


Figure 2. AMBA Bus Architecture

- Determine an effective error injection model which simulates a single-bit error scenario in the best manner.
- Find the most common effects of single-bit errors in the bus system
- Find the cause for such effects and quantify their dependency on various components of the bus system.
- Provide an effective scheme to predict the effect of any single-bit error on the whole system while executing any application.

We demonstrate achieving the aforementioned goals in the context of AMBA bus. A cycle-accurate virtual platform simulator written in SystemC, for the AMBA bus-based System on Chip architecture, is used for simulations of our model. Details about the platform may be obtained from [13]. Table 1 shows the various signals in AMBA bus which we consider for analysis in our model. We have considered almost all the signals in the AMBA bus system in this study. Other signals are either not implemented in the virtual platform used by us for simulations or are used to support different kinds of operations which are beyond the scope of this study.

3.1 Error Injection

We inject a single-bit error by corrupting the various signals of interconnect during the operating phase of the bus's finite state machine. The single bit error is injected just once by modifying the state of one of the interconnect signals. Such an error is injected randomly at any time instant. However, to obtain the effect of the error on a particular transaction we ensure that the error is injected during the required transaction in the bus. This simulates the single-bit error in our simulation precisely. In this paper we just consider two types of transaction in the bus, namely the BUS-READ and the BUS-WRITE operations. The characterization model could be further extended to other transactions like, BURST-READ, SPLIT-READ and similarly for the write transactions.

Signal name	Function
HBURST[2:0]	This signal indicates if the transfer is a form of a burst.
HWRITE	When HIGH, this signal indicates a write transfer and when LOW, a read transfer is executed.
HTRANS[1:0]	This signal indicates the type of current transfer, which can be non-sequential, sequential, idle or busy.
HSIZE[2:0]	Indicates the size of transfer; typically byte(8-bit), halfword(16-bit) or word(32-bit)
HREADY	When HIGH, the signal indicates a transfer completed on the bus. The signals can be set on low if we need to extend a transfer.
HRESP[1:0]	Response from the slave. Four different responses are provided, OKAY, ERROR, RETRY and SPLIT.
HBUSREQ	This signal is from a bus master to the bus arbiter which indicates that the bus master requires the bus.
HMASTER[3:0]	Arbiter indicating the bus master that is currently performing a transfer.
HGRANT	Ownership of the address and control signals changes at the end of a transfer when HREADY is HIGH, so a master gets access to the bus when both HREADY and HGRANT signals are HIGH.
HSEL	This signal indicates that the current transfer is intended for the selected slave.
HADDR[31:0]	This is the 32-bit system address bus that indicates the address to read or to write.

Table 1. Definition of signals and its function

3.2 Consequences of Errors

Based on the various errors observed due to single-bit fault we have categorized the effects of the error into four main classes. The errors are classified based on their effect on the system. The four main kinds of implications of the errors in the system are:

- **Fatal Error (FE):** Errors that may lead to system crashes. The fatal error means the system detects a fatal situation that causes the system to crash. Fatal error includes:
 1. The address can not be mapped to any existing slave. The reason for this is the address signals are changed to incorrect status.
 2. The core is trying to write an instruction that is read-only. One reason for this error is an incorrect HWRITE signal
 3. The core accepted a signal that should not have happened. For example, consider a scenario when the core expects to stop, but receives a spurious grant signal from the arbiter due to an erroneous bit flip.
 4. Incorrect or bad address that should not have occurred during the transfer operations. For example, during a burst transfer, the next requested address is not correct offset address of previous address. It is however important to note that this error could easily be overcome if we somehow retain the correct address information, in which case we may prevent system crashes.

- **Deadlock(DL):** Errors that lead the system to get into no progress states. Several reasons for the error are infinite loops due to modifications in loop invariants, or the errors cause incorrect arbitration so no master can do its work and so the system deadlocks. We found out two types of deadlock situations in our simulations.

1. Program counter corruption: Cases when the program counter jumps to an unknown location and there is an invalid instruction and the processor keeps retrying. We detect it by having a *stop_simulation()* function in our benchmarks, which contains a software interrupt, is used to end the simulations. If the Program counter is affected and the program behavior is changed, and the *stop_simulation()* is either not executed or executed incorrectly the system enters infinite loop.
2. Undefined state: Cases when the state machine of the bus which defines the bus protocol, enters an undefined state, each of the processors wait for the response from the bus to obtain access, however there is no progress from anyone.

- **Silent Data Corruption (SDC):** Errors that remain unnoticed until the end of the simulation but provide incorrect results. SDC primarily occurs due to accumulation of error over many cycles without the system getting in deadlocks or crashing. One of the prime reasons for SDC is the incorrect HADDR signals that generate read or write transaction on the incorrect address. This causes incorrect data processing that lead to incorrect results.

- **No Effect:** Errors that do not affect the system in any way.

3.3 Transaction based Error Characterization

The error simulations are executed for multiple benchmarks and the criticality of each signal is assessed based on the effects of a single-bit error on any signal over different types of transactions, namely, read and writes in our case. We define values $P_{f|T=r}$, $P_{d|T=r}$, $P_{f|T=w}$ and $P_{d|T=w}$ for each signal, where:

- $P_{f|T=r}$ stands for percentage of times the error in a signal leads to fatal error during a read transaction.
- $P_{f|T=w}$ stands for percentage of times the error in a signal leads to fatal error during a write transaction.
- $P_{d|T=r}$ stands for percentage of times the error in a signal leads to a deadlock during a read transaction.
- $P_{d|T=w}$ stands for percentage of times the error in a signal leads to a deadlock during a write transaction.

The behavior of the error was uniform in the cases of the fatal errors and deadlocks for all the benchmarks; however the silent data corruption was completely dependent on the nature of the benchmarks and hence characterizing it was not possible. The reason for such an abrupt behavior with respect to silent data corruption could be attributed to the fact that SDC are highly dependent on the nature of the application. This error characterization step helps us finding the criticality of different signals and our observations are discussed in the results section 4.

3.4 System Level Prediction

The signal level susceptibility analysis is extended to system level by using the error-effect percentages of the signals described in the previous section. The errors are injected in the signals based on a probability dependent on the width of the signal. The effect of the errors are recorded and compared with the estimated values. The effect of any single-bit error on any signal is estimated using the following equation:

$$P_{sf} = Write\% \times \sum_{all\ signals} p_i \times P_f|T = w \quad (1)$$

$$+ Read\% \times \sum_{all\ signals} p_i \times P_f|T = r$$

, where P_{sf} is the probability of any error leading to a fatal error in the system, p_i is the probability of an error hitting the signal and $P_f|T = r$ and $P_f|T = w$ are as defined earlier. The value p_i is computed as the ratio of the signal width over the total number of bits in all the signals considered in our analysis. This reflects that the fact that the probability of an error occurring in any signal is directly proportional to the number of bits of a signal.

Such analysis provides a near accurate estimate of the effect of any single bit error on the execution of any application on the bus system. Similar equation for the different effects of error may be written and their probabilities may be estimated for different applications. Such probability numbers characterize the effect of errors completely in any bus-based systems. Note that the accuracy of the prediction may further be improved by increasing the granularity of our analysis. We may consider more system parameters while characterizing each of the signals, like number of memory accesses etc, and use those to increase the prediction accuracy of the model. In such cases all the parameters should be incorporated into the prediction equation.

4 Experimental Setup

We modify a cycle-accurate virtual platform, which models System On Chip architectures having ARM cores and has a capability of simulating different interconnect architectures. We analyze AMBA-AHB interconnect in our work. A boot time error file is enabled in the system during which various types of errors may be injected at different intervals on different components of the bus system. Another

set of scripts is written to perform the profiling operation and determine the effects of various errors and finally observing the criticality of each of the errors. We perform the susceptibility analysis for different signals of the AMBA bus system.

We model the error susceptibility of the System-on-chip by first identifying the error susceptibility of each signal. The error susceptibility of each signal is found by executing a benchmark numerous numbers of times while injecting only one error on a signal in each execution. The results of the benchmarks are analyzed and we determine the number of executions resulting in fatal errors and deadlock situation. A deadlock is flagged when an application does not complete by a particular time. After getting the results from these executions of the benchmark, we can calculate the error susceptibility of each signal by dividing the error states over the number of executions. We do the same for each of the benchmarks and then get the average error susceptibility for each signal of all the benchmarks.

We instantiated four ARM cores in the virtual simulation platform and four private memories attached to the bus. Private memories are slaves that could be used as caches by the processor cores. The shared and semaphore memory space was instantiated as well. Table 2 shows information about the different applications used by us for error characterization when executed on the SoC platform with the aforementioned specifications. Except for the first three benchmarks which were implemented by us, the rest of the benchmarks were picked from the SPLASH suite.

App	C	BB(%)	BT(%)	R	W
Qsort	16576	47.4	23.53	56	743
Matrix	133503	36.59	18.29	240	23444
PIL Filter	220288	41.59	19.10	2640	7151
FFT	418920	61.21	29.97	1669	28053
DES	205307	54.01	20.61	33417	9370
LU	2051411	68.05	27.71	260717	302519

C: The number of the executing cycles. BB: The percent of time that the bus is busy. BT: The percent of time that the bus is transferring data. R: The number of read transaction W: The number of write transaction

Table 2. The six benchmarks used in our simulation

Signal	Qsort	MATRIX	PILFILTER	FFT	DES	LU
HBURST	0.1	0.1	0.1	0.3	0.4	0.0
HWRITE	1.4	0	0.5	1.1	0.9	1.1
HTRANS	1.2	0	0.3	0.7	0.55	0.7
HSIZE	18.6	18.4	18.2	23.1	23.4	21.5
HREADY	0.5	0.1	0.9	2	2.2	1.8
HRESP	0	0	0	0.1	0.1	0
HBUSREQ	0	0	0	0	0	0
HMASTER	0.3	0	0	0	0.2	0.1
HGRANT	1.8	0.1	0.1	0.7	1.6	1.6
HSEL	22.3	24.9	19.1	29	26.9	28.3
HADDR	7.4	5.2	10.3	8	8.1	9.4

Table 3. Fatal Error rate for each signal in different benchmarks during a bus read transaction ($P_f|T = r$)

5 Results

To analyze the susceptibility of errors on different signals we first conducted experiments to find out the effect of error on each signal for different benchmarks for different transaction types. Table 3 shows the value $P_f|T = r$ for different applications obtained over 10000 runs. Clearly we can observe that in general the effect of the error on different signals remains similar across different benchmarks, mainly due to the fact that the error is characteristic of the protocol more than the nature of the benchmark. Similar characteristics are observed for the values $P_d|T = r$, $P_f|T = w$ and $P_d|T = w$, values for all the applications.

To obtain the criticality measure for each of the signals we obtain the average probability values for all the signals over different benchmarks for read and write transactions. Table 4 shows the average error rate of $P_f|T = r$, $P_d|T = r$, $P_f|T = w$ and $P_d|T = w$ for all the signals. We may observe that signals like HBUSREQ are very sensitive with respect to the system getting into deadlocks, on the other hand the probability of an error in HBURST leading to malfunctioning of the system is really low. The reason for higher deadlocks due to erroneous HBUSREQ, HRESP and HREADY signals could be attributed to the fact that all of them are quite important in establishing the right communication protocol, and consequently an error in them makes all the resources in the system in a busy wait state. HADDR is another signal which leads to system crashes with a high probability mainly due to the system address going out of known or eligible ranges. The advantage of such an analysis is the fact that it provides a good opportunity for the reliability engineer to prioritize his focus on guarding the signals.

Signal	$P_d T = r$	$P_f T = r$	$P_d T = w$	$P_f T = w$
HBURST	2.57	0.17	0.51	0.33
HWRITE	11.80	0.83	0.77	0.37
HTRANS	8.29	0.58	0.70	0.29
HSIZE	0.00	20.53	0.14	23.70
HREADY	13.20	1.25	12.14	2.07
HRESP	30.01	0.03	24.49	0.00
HBUSREQ	32.62	0.00	14.25	0.00
HMASTER	0.00	0.01	0.99	0.63
HGRANT	5.53	1.07	1.56	0.50
HSEL	5.59	25.08	8.21	35.63
HADDR	4.36	8.06	0.53	9.81

Table 4. Average error rate for each signal

The criticality, however, is also dependent on the bit width of each of the signals, which reflects a higher vulnerability of the signal with respect to others. For example, HADDR is more prone to errors as compared any of the other signals due to higher bit-width. To analyze the criticality of the signals while taking into account the bit widths, we executed the benchmark circuits 20000 times, with a single-bit error injected during each run. The error injection was done probabilistically on all the signals, with the probability of error on any signal proportional to its width. The results were recorded and compared Figure 3 and 4 show the percentage of fatal errors and deadlocks caused due to errors in different signals on different benchmarks.

As expected the probability of deadlock and fatal errors due to the signal HADDR are the maximum due to its higher vulnerability towards getting struck by an error.

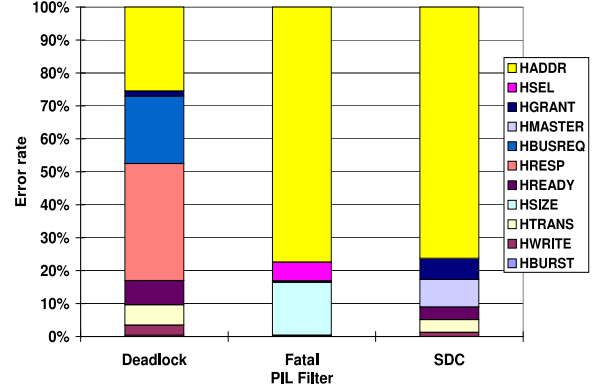


Figure 3. Effects of error on different signals for PIL-Filter benchmark

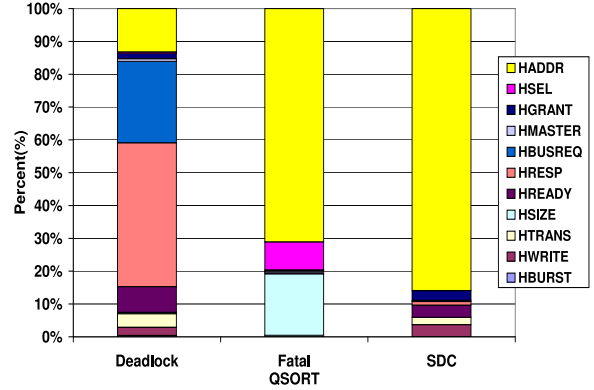


Figure 4. Effects of error on different signals for Qsort benchmark

Using the probability values and the number of transactions in different benchmarks, we predict the probability of the effect of any single bit error in the system. Such prediction is then validated with actual simulations. Figure 5 and 6 demonstrate the comparison of the predicted values and the simulated values. Our prediction scheme predicts the probability of the deadlocks with an accuracy of 94% and fatal errors with an accuracy of 90% respectively. It is important to note that the prediction accuracy is quite important considering the run-time of the simulations. The prediction is just using the signal probability values from Table 4 and the number of transactions from Table 2. Table 5 shows the average run-time of the 20000 simulations which we perform to obtain the system level effect of any single-bit error. Clearly the prediction scheme provides a much better methodology to obtain an estimate on the effect of any error in the whole system.

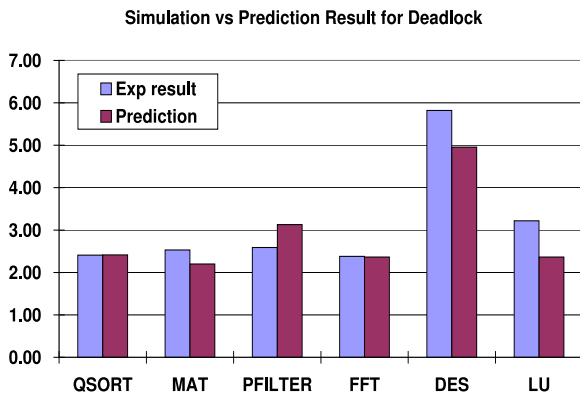


Figure 5. Comparing experimental value with predicted value for deadlocks

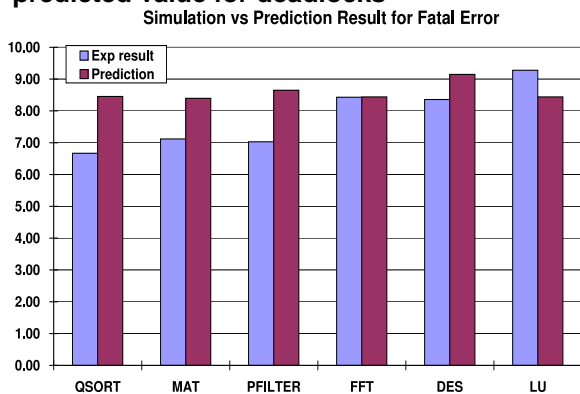


Figure 6. Comparing experimental value with predicted value for fatal errors

6 Conclusion and Future Work

Our work has demonstrated a novel, generic and effective transaction based error characterization scheme for bus architectures in SoCs. Such a model is quite generic and can be extended to other systems. The criticality measure of each of the signals provides an opportunity to ideally prioritize employing any error correction schemes on the system, which is quite critical with the shrinking power and area budgets. The error effect prediction scheme provides the probability of the effect of any single-bit error on the system with an accuracy of 92% on an average. Future work could be focused on extending this model at a finer granularity by classifying the impact of errors at different states

Benchmark	Time (in hours)
Qsort	5.44
MATRIX	15.93
PILFILTER	20.76
FFT	54.73
DES	56.16
LU	481.1

Table 5. Run time for obtaining system level effect of any single-bit error

of the system.

7 Acknowledgement

This work is supported by NSF 0093085, MARCO/DARPA, SRC 00541. The authors would like to thank University of Bologna for their support with the tools for implementing the proposed idea.

References

- [1] ARM AMBA specification (rev2.0) and multi-layer AHB specification. This document is available at http://www.arm.com/products/solutions/AMBA_Spec.html.
- [2] IBM CoreConnect Bus Architecture White Paper. This document is available at <http://www.chips.ibm.com/products/coreconnect>.
- [3] PCI Express Specification 1.0a. This document is available at <http://www.pcisig.com/specifications/pciexpress/>.
- [4] Wishbone specification. This document is available at <http://www.silicore.net/wishbone.htm>.
- [5] C. Metra and B. Ricco. Optimization of Error Detecting Codes for the Detection of Crosstalk Originated Errors. In *Proceedings of Design Automation and Test in Europe*, 2001.
- [6] C. Metra and M. Favalli. Bus Crosstalk Fault-Detection Capabilities of Error-Detecting Codes for On-Line Testing. In *IEEE Trans. on VLSI Systems*, pages 392-396, Sept. 1999.
- [7] C. Svensson. Optimum Voltage Swing on On-Chip and Off-Chip Interconnect. In *IEEE Journal of Solid-State Circuits*, pages 1108-1112, July 2001.
- [8] C. Weaver, J. Emer, S. S. Mukherjee and S. K. Reinhardt. Techniques to Reduce the Soft Error Rate of a High-Performance Microprocessor. In *Proceedings of International Symposium on Computer Architecture*, 2004.
- [9] D. Bertozzi and L. Benini. Xpipes: A Network-on-Chip Architecture for Gigascale Network-On-Chip. In *IEEE Circuits and Systems Magazine*, Vol 4, No.2, pages 18-32, 2004.
- [10] D. Bertozzi, L. Benini and G. De Micheli. Low Power Error Resilient Encoding for On-Chip Data Buses. In *Proceedings of Design Automation and Test in Europe*, 2002.
- [11] K. L. Shepard and V. Narayanan. Noise in Deep Submicron Digital Design. In *Proceeding of International Conference on Computer Aided Design*, pages 524-531, 1996.
- [12] L. Anghel and M. Nicolaidis. Cost Reduction and Evaluation of Temporary Faults Detecting Technique. In *Proceedings of Design Automation and Test in Europe*, 2000.
- [13] M. Loghi, F. Angiolini, D. Bertozzi, L. Benini and R. Zafalon. Analyzing On-Chip Communication in a MPSoC. In *Proceedings of Design Automation and Test in Europe*, 2004.
- [14] N. Dhanwada, I.-C. Lin and V. Narayanan. A Power Estimation Methodology for SystemC Transaction Level Models. In *Proceedings of CODES-ISSS*, 2005.
- [15] R. Ramanarayanan, V. Degalahal, N. Vijaykrishnan, M. J. Irwin and D. Duarte. Analysis of Soft Error Rate in Flip-Flops and Scannable Latches. In *Proceedings of ASIC03*, September 2003.
- [16] S. Srinivasan, L. Li and V. Narayanan. Simultaneous Partitioning and Frequency Assignment for On-Chip Bus Architectures. In *Proceedings of Design Automation and Test in Europe*, 2004.