The Impact of Soft Errors in Logic and its Commercialisation in ARM IP

Derek Graham

A thesis submitted to
the Universities of

Edinburgh
Glasgow
Heriot-Watt
Strathclyde

For the Degree of
Doctor of Engineering in System Level Integration

Abstract

The significance of soft errors in logic has grown because of reduced memory vulnerability and the shrinking dimensions of semiconductor technology coupled with the increasing amount of logic integrated into a chip. Consequently, some of ARM's customers are concerned about how soft errors on the bus interconnect will affect the dependability of their systems, since the interconnect is a critical hub of communication in a SoC and represents a substantial and growing amount of logic. With the rising complexity of their systems, the interconnect will become larger and more complex in the future, adding to their concern. In this work the impact of soft errors on the bus interconnect logic was investigated and a product was developed to ameliorate the effects of such errors on ARM's customers' products.

Methods to measure the SER of ARM IP were investigated by focusing on logical masking, which is a component in the calculation of the SER. The effect that the topology of a combinatorial logic circuit has on its logical masking rate was considered by performing gate-level statistical fault injection on different implementations of adder circuits. Significant variation in logical masking was found ranging from a factor of 3.1 at a synthesis frequency of 100 MHz to a factor of 2.1 at 900 MHz. This difference is explained in an original way by correlating logical masking with the circuit's path length and fan-out. These properties could be used to create a static method of measuring the logical masking rather than the current time-consuming method of dynamic simulation. Additionally, nearly 30% of faults injected cause more than one error, which means that the combinational SER will be underestimated if research does not take gate fan-out into consideration. Using this methodology a circuit designer can now base his choice or development of a circuit on its reliability as well as its performance, power, and area. Studying the variation in the factors that affect the SER is important to ensure accuracy in addressing customer requirements.

Although it is important to consider the rate of soft error occurrence, in this work the impact of errors is demonstrated to be critical. Using protocol-level fault injection it is shown that faults on the ARM AXI bus interconnect can have a serious effect on the reliability of the entire SoC such as deadlock, memory corruption, or undefined behaviour. Using a fault-path traversal algorithm, it is demonstrated that traditional error detection codes are not sufficient at preventing these failures when faults occur on certain AXI bus signals. This led to the development of novel fault tolerant methods that provide protection for these identified signals. Based on these developments, a product was proposed for an add-on to the AXI bus interconnect that can detect, correct, and report logic soft errors without changing the AMBA standard or the customer's connecting IP. With agreement from internal stakeholders and interest from automotive and storage networking customers, this proposal was successfully transferred out of ARM R&D and could be available as a commercial offering in the next two years.

Acknowledgements

I would like to thank my industrial supervisors Per Strid and Daryl Bradley and academic supervisors Scott Roy and Fernando Rodriguez for their support and feedback; my friends in ARM R&D in Cambridge for advice, reading my thesis, and for helping to expand my knowledge; colleagues around ARM for enabling the advancement of the product proposal; Siân Williams at the ISLI for guiding me through the EngD process; my friends in Cambridge for providing distractions; my family for their encouragement; and finally to Mariken Schipper for her care and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It's ridiculous. I've got a \$300 000 server that doesn't work. The thing should be bulletproof.

— Company president whose Sun Microsystems server repeatably crashed due to soft errors in 2000 [3]

Since the inception of computers, building them to operate reliably has been paramount. Changes in the underlying technology, from vacuum tubes through to Complementary Metal-Oxide-Semiconductor (CMOS), has brought ever decreasing manufacturing defect rates [4, 5, 6] and ever increasing complexity that has meant that the design of reliable systems has surpassed local computational hardware to encompass networks, software, and the user interface [7]. Faults induced by natural interference originating from the varied and sometimes hostile environment in which technology has permeated have grown in importance to the semiconductor industry [8, 9, 10, 11]. There are several reasons for this: manufacturing defect rates have fallen so that the relative importance of these less controllable fault sources has grown[1]; it is in the commercial interest of business to minimise the blame, and thus the cost, attributable to themselves due to their customers' problems; and the occurrence of these natural faults is not as amenable to control as, for instance, design and manufacturing faults are. Transient faults induced by radiation, commonly called soft errors [12], were traditionally a problem for the aerospace industry [13, 14, 9] but are of growing concern at ground level [15, 16]. At a terrestrial altitude the primary causes are alpha particles from computer chip packaging and high-energy atmospheric neutrons from cosmic rays [9]. These particles can strike sensitive areas of the chip and cause the value held by memory elements to flip or the output value of logic elements to experience a momentary glitch [2]. Although these events are rare[2], their random nature skews their perceived risk. Rare events still matter because the enormous volumes of computer chips sold around the world amplifies the total failure rate. For some companies, such as those who build data centres that provide business critical services, the impact that soft errors have on the

[1]Along with other sources, such as software.

[2]For instance from laboratory experiments, a megabit of SRAM manufactured in 65 nm technology (the cutting-edge a few years ago) could experience about a thousand bit-flips during an operating period of approximately 100 000 years.

reliable operation of their systems is important. Moreover, with shrinking device geometries and the increasing integration that technology scaling brings, the rate of occurrence of soft errors is predicted to rise [17, 2, 18], which will mean that more users of technology will be affected.

For ARM, automotive and networking customers are particularly concerned with the adverse impact that soft errors can have on reliability because it affects the safety and availability, respectively, of their computer systems. These systems will typically already feature memory error protection, which increases the importance of the reliability of the other components embedded in their electronic chips. In the embedded Intellectual Property (IP) that ARM sells to those customers who build the Systems-on-a-Chip (SoCs) that are used in automotive and networking hardware equipment, it is important to consider the effect of soft errors on the on-chip communications bus that connects the IP blocks together. ARM's Advanced Microcontroller Bus Architecture (AMBA) standard features in many embedded SoCs, it can occupy a large area comparable to the microprocessor[3], and faults that happen in this communication bus can have severe consequences, such as deadlocking the whole system.

Based on the analysis above, the objectives and contributions of this work are:

1. *Add knowledge and capabilities to ARM in the area of reliability and soft errors.* ARM's current offering is found wanting by some of its customer base. This work seeks to understand whether soft errors are an issue for ARM's partners such that some form of action is required. And in course, to find out what action should be taken. The case that soft errors are a current concern for some customers is made, namely those in the automotive and storage networking markets. We identify trends from the existing literature which indicate that the Soft Error Rate (SER) may increase in the future with shrinking transistor process dimensions, which means it will become more important for those already highlighted and may become a concern for other markets. Chapters two and three, the technical and commercial background, capture some of this knowledge.

2. *Investigate and develop methods to measure the SER of ARM IP.* It is important to quantify the SER of a product to determine whether it meets customer requirements and, if improvement is needed, where to target effort and to allow the effectiveness of such modifications to be assessed. Externally, such information is valuable for customers to compare with other offerings, however the SER is regarded as commercially sensitive meaning that this comparison is rarely possible. ARM has not comprehensively measured the SER of its IP before and lacks the relevant expertise, therefore, there is a need to evaluate whether there is sufficient value in doing so and this requires investigating the benefits and costs of different approaches. This work focuses on part of the process of calculating the SER in combinational logic because this type of logic forms a notable part of ARM IP such as processors and bus interconnects and combinational logic is predicted to become a major contributor to the overall SER [17]. We look at the variation of one of the factors that affect the SER of a

[3]From an internal test-chip the bus interconnect logic was about 70% of the processor's logic area (excluding the floating-point unit and cache memory)

combinational logic circuit, logical masking, in a novel way by exploiting the control that an IP designer has over circuit synthesis. The aim is to allow the designer to explore the trade-off between SER and performance, power, and area. We find that there can be significant difference across circuit implementations, which is important as it effects the calculation of the total error rate. Additionally, we correlate characteristics of the circuit topology with the logical masking rate to help explain the observed variation. Focusing on one part keeps the investment small while exploring the whole SER measurement process.

3. *Investigate the impact of soft errors on the bus interconnect logic and develop novel methods to improve its reliability.* The AMBA IP bus interconnect is an important ARM product that serves as the hub of communication in an embedded SoC. It is a significant piece of logic that is comparable in area to some embedded ARM processors and which will increase in size with the rising complexity of the surrounding system. It is demonstrated that soft errors on the logic of the bus interconnect can have serious consequences and presents a high enough risk to be of a concern to those customers who care about dependability. While it is shown that for certain circumstances known parity and Error Correcting Code (ECC) techniques are sufficient, there are some cases where they do not provide adequate protection. In the former case, these techniques add extra latency onto the transfer of data across the bus, which is contrary to the requirements of automotive customers, therefore novel ways of applying these techniques were developed to avoid extra latency overhead. In the latter case, new methods were developed to detect faults on the bus logic that techniques such as ECC cannot detect.

4. *Develop a product to enable ARM's partners to cost effectively mitigate the consequences of soft errors on their ARM based bus interconnect.* Automotive and storage networking customers are concerned about the safety and data integrity, respectively, of their Advanced eXtensible Interface (AXI) based bus interconnects to soft errors. Trends indicate that their interconnects will grow in size, complexity, speed, and data transferred, thus their concern will continue in the future. Existing customers of AXI look to ARM to provide error protection as another feature because this is seen as cheaper than implementing it on their own or switching to a competitor (although it is believed that no other competitors offer such protection). Some potential customers in the storage networking market have their own interconnect that features ECC on the data payload, but want to use AXI because it is no longer cost effective to develop their own interconnect. A lack of error protection on AXI presents a barrier for them to adopt this protocol. Therefore, the proposed product is an add-on to the AXI bus to improve its fault tolerance to transient faults that occur on the bus logic during communication. Two revisions of the product are advocated to lower the barrier for adoption. The first revision will include ECC protection of most of the AXI signals, while the second revision will include mechanisms to detect faults that occur on the signals that cannot be fully covered by ECC. It will integrate with an existing product to allow information on those transactions where correctable and uncorrectable errors have occurred to be logged and the response action to be programmable. This

proposed solution minimises the barrier for adoption as the techniques are encapsulated in the bus implementation and its functionality exposed through a programmable interface for software. Thus, it does not require changes to the widely used AMBA standard nor does it require customers to change their existing IP. This is important for customers as it minimises the extra usage costs involved. This has resulted in the successful transfer of a product proposal out of Research and Development (R&D), which could be available as a commercial product in the next two years.

The following papers were published in support of the above contributions:

1. D. Graham, D. Bradley, S. Roy, P. Strid, and F. Rodriguez (2008). Design For Reliability: An Analysis of Logical Masking on Transient Faults. In *Workshop on System Effects of Logic Soft Errors (SELSE).*

2. D. Graham, S. Roy, P. Strid, and F. Rodriguez (2009). A Low-tech Solution to Avoid the Severe Impact of Transient Errors on the IP Interconnect. In *International Conference on Dependable Systems and Networks (DSN).*

The first paper (see Appendix B on page 82) presents the work on logical masking variation and the correlation with circuit topology as outlined in the second objective above. The second paper (see Appendix C on page 87) covers the third objective by presenting some of the work on the impact of faults on the AXI bus and the fault tolerant techniques that avoid adding extra communications latency. The methods to detect faults on the bus matrix logic that Error Correction or Detection Codes (EDACs) cannot detect are presently considered important enough to ARM that they are under publication embargo. A patent covering these methods is currently before the ARM patent review committee. Within ARM, a product proposal requires two key documents: the Product Requirement Specification (PRS) and the Engineering Requirement Specification (ERS), which represent why the product should be developed and what should be implemented. An additional report was written which describes the reasons why certain choices were made, which is an important record of decision making for the implementation team. These three documents support the first and fourth objectives. They are the culmination of a substantial effort to construct a credible business case and the technical knowledge of what to build and how it integrates with other products. Crucial to this was attaining the commitment of key personnel: a product manager who represents the interface between engineering and customers, the lead technical engineer of the bus interconnect product that this project is to be integrated into, and the authority on the AMBA standard. These strategic relationships were formed and consolidated over many presentations and discussions.

The outline of the rest of the EngD thesis portfolio is as follows:

- Chapter 2 provides direct and indirect evidence of the importance of soft errors in the design of dependable computer systems; some important definitions; how soft errors are manifested; trends in the SER of different elements with technology scaling; approaches in which dependability can be achieved; and the main methods to measure the rate of soft errors.

- Chapter 3 presents the commercial motivation for a product that improves the fault tolerance of the AMBA AXI bus interconnect by examining the trends that affect the semiconductor industry and target markets. Additionally, it explains more about the customers in the target markets and why they care about reliability.

- Chapter 4 explains the method for measuring the logical masking of a combinational circuit using Statistical Fault Injection (SFI) and the results are linked to measurable graph properties of the the circuit.
- Chapter 5 continues the investigation of soft errors in logic by examining the impact they have on the AXI bus interconnect. The workings of the AXI protocol and the innards of the bus interconnect which implements the protocol are explained. This is followed by the requirements which narrow the solution space based upon the needs of target customers. After which the argument is put forward as to why ECC is not sufficient to protect against all the consequences of faults on the AXI bus and potential solutions based upon the fault effect are explained. The chapter concludes with a description of the add-on product to the AXI bus that improves its fault tolerance.
- The final chapter summarises the work and contributions for the achievement of the Engineer Doctorate award and outlines future directions of research.

Chapter 2

Technical Background

Soft errors were initially a key problem for the aerospace industry due to elevated levels of radiation at high altitudes and stringent requirements on the reliability of avionics systems. After Intel discovered in the late 70s that contamination of the water supply used in one of their fabrication facilities had caused soft errors in their memory products [12], concern mounted that soft errors could be a problem in the terrestrial environment. Intel's discovery was the first report that alpha particles from the packaging of a computer chip could cause these types of errors. Since then standards were created to limit alpha particle emissions from packaging and most fabrication plants cleaned up their processes [2]. Not long after the publication of Intel's results, Ziegler and Landford from IBM [15] were the first to publish data that showed cosmic rays could also cause soft errors in electronics at sea-level. Later, a field study in the mid-90s of computer servers and corroboration with accelerated testing determined that high-energy atmospheric neutrons, a product of cosmic radiation, were the main cause of soft errors in Dynamic Random Access Memory (DRAM) and Static Random Access Memory (SRAM) at ground level [16]. A third cause was discovered by Baumann et al. [19] in 1995, due to the interaction of low-energy atmospheric neutrons with boron, which is used as a dopant in silicon and is found as a diffused impurity in gate dielectrics. As a result, this was removed from the manufacturing process for virtually all advanced technologies [2]. In 2001, an internationally agreed standard on the measurement and reporting of soft errors, induced by alpha particles and atmospheric neutrons, was published [20].

Soft errors are one of many sources of failure in modern computer systems. This chapter seeks to assess their importance now, by examining direct and indirect evidence, and in the future as it is predicted that their occurrence will increase with semiconductor technology scaling. Before doing so, the standard definitions needed to aid understanding are described and an explanation of a soft error is provided.

2.1 Definitions

The following definitions are taken from Avizienis et al. [1], a culmination of work that dates back to 1980, which "aims to give precise definitions characterising the various concepts that come into play when addressing the dependability and security of computing and communication systems". The dependability of a system is defined as "the ability to avoid service failures that are more frequent and more severe than is acceptable" and integrates the following attributes:

- availability: readiness for correct service
- reliability: continuity of correct service
- safety: absence of catastrophic consequences on the user(s) and the environment
- integrity: absence of improper system alterations
- maintainability: ability to undergo modifications and repairs

Correct service is delivered when the service implements what the system is intended to do and a failure is an event that occurs when the delivered service deviates from correct service. An error is the part of the system's total state that deviates from the correct service state which may lead to its subsequent failure. The cause of the error is called a fault. It is importance to note that an error does not necessary lead to a failure nor a fault to an error. For example, an error in a memory location that holds a value that is never used during the operation of a software program will not result in failure.

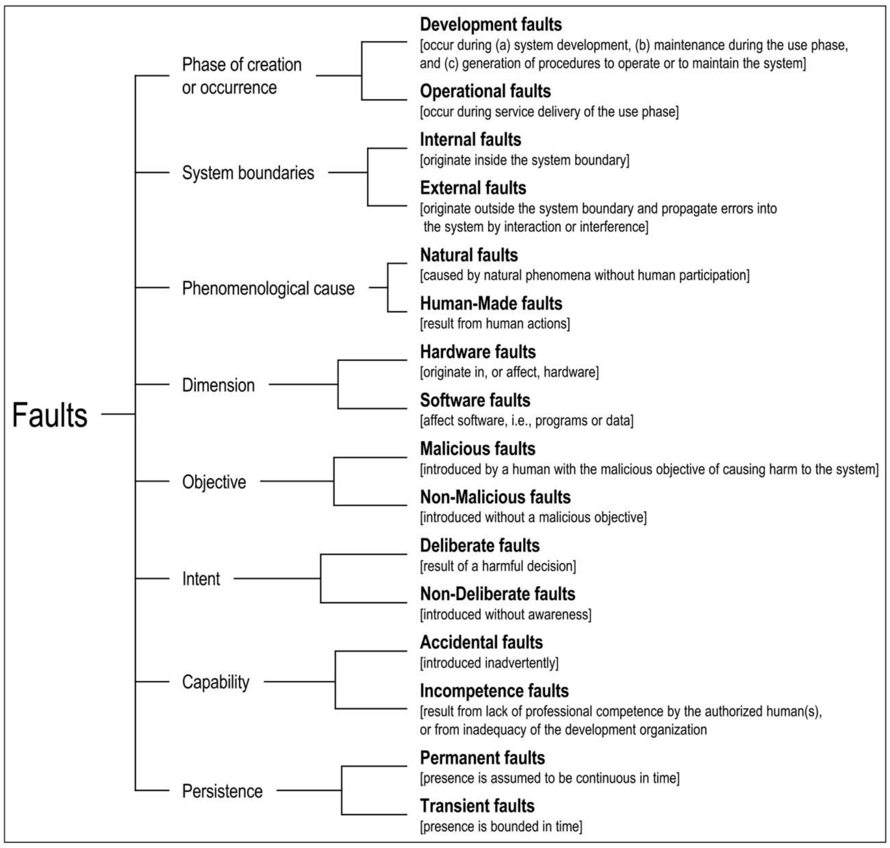
The eight basic perspectives that are used to classify all faults that may affect a system during its life are given in Figure 2.1a and the combinations that, according to the authors, are likely are given in Figure 2.1b. This work considers transient physical faults that affect the hardware during operation, caused by natural phenomena from inside the Integrated Chip (IC) or from the environment. Transient means that the fault's presence is bounded in time.

There are a number of terms different than those set out by Avizienis et al., which are used primarily within the confines of this research area. It will not be attempted to normalise these terms into his taxonomy, however, they are given because they have arisen from historical usage in this field or they represent a refinement of some of the concepts.

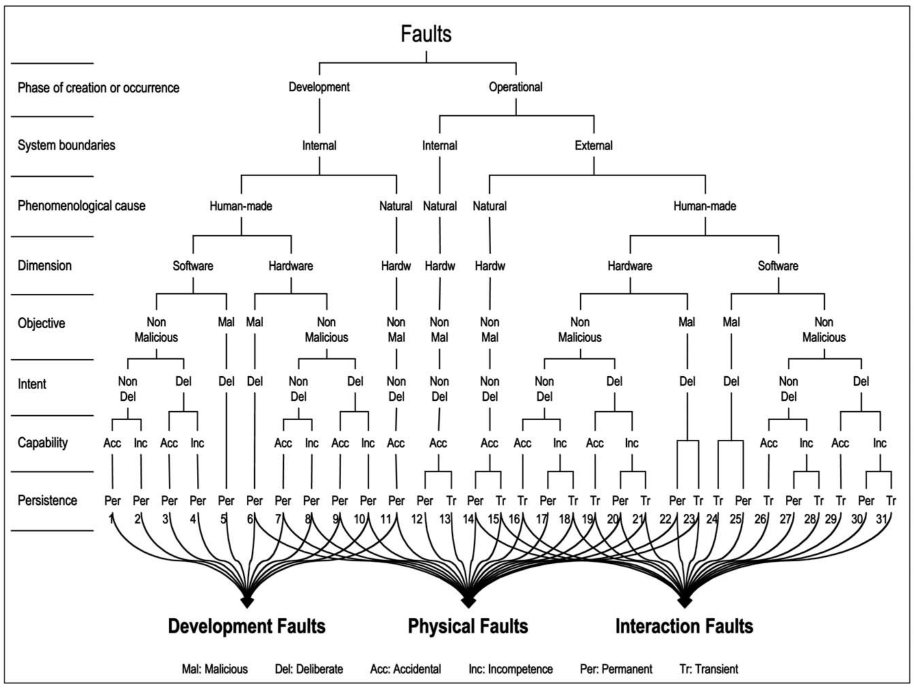
The natural phenomena considered are due to radiation, which has two principal effects on microelectronic systems: total radiation dose effects and Single Event Effects (SEEs).

Total radiation dose is concerned with the accumulated effect of radiation impinging on a device, resulting in permanent changes in its electrical behaviour. Radiation impact can damage the silicon substrate and gate oxide of transistors, and can also cause an accumulation of charged particles within the device. Total radiation dose primarily affects a device's electrical behaviour and long-term reliability. Total radiation dose effects are primarily a concern in adverse operating environments, such as space and military applications [2].

SEEs are the result of a single radioactive particle strike in a sensitive area of a transistor that can cause the logical value of the circuit to change. An SEE is also often referred to as a soft error because no permanent damage to the device is caused, unlike a hard error. Although a soft error does not permanently damage the system, its consequences may do. The conditions in which a soft error is created are explained in more detail in Section 2.2. SEEs are classified into a number of categories depending on where or how they occur [2]:


Faults
Phase of creation or occurrence
Development faults
[occur during (a) system development, (b) maintenance during the use phase, and (c) generation of procedures to operate or to maintain the system]
Operational faults
[occur during service delivery of the use phase]
System boundaries
Internal faults
[originate inside the system boundary]
External faults
[originate outside the system boundary and propagate errors into the system by interaction or interference]
Phenomenological cause
Natural faults
[caused by natural phenomena without human participation]
Human-Made faults
[result from human actions]
Dimension
Hardware faults
[originate in, or affect, hardware]
Software faults
[affect software, i.e., programs or data]
Objective
Malicious faults
[introduced by a human with the malicious objective of causing harm to the system]
Non-Malicious faults
[introduced without a malicious objective]
Intent
Deliberate faults
[result of a harmful decision]
Non-Deliberate faults
[introduced without awareness]
Capability
Accidental faults
[introduced inadvertently]
Incompetence faults
[result from lack of professional competence by the authorized human(s), or from inadequacy of the development organization
Persistence
Permanent faults
[presence is assumed to be continuous in time]
Transient faults
[presence is bounded in time]

(a) Elementary



(b) Combined

Figure 2.1: Fault classes (from [1])

- Single Event Transient (SET): an SEE occurs in combinational logic which creates a transient current/voltage pulse and may propagate through the circuit
- Single Event Upset (SEU): either an SEE directly occurs in a storage element, such as an SRAM cell or latch, which flips the value held or an SET propagates and is captured by a storage element
- Single Event Latch-up (SEL): an SEE activates a parasitic junction between the transistor's well and substrate, leading to a potentially destructive path between the power rails, which causes the transistor to be 'stuck' at a particular value. Powering down the chip will remove this condition, as long as no physical damage has occurred to the circuit

Nguyen et al. [21] define the outcome of a soft error into four types according to the system's response, or lack thereof:

- Masked error: the corrupted bit disappears before being noticed because it is not used in a computation, is overwritten, or is deleted; therefore it does not cause failure
- Correctable error: detected and corrected by hardware or software mechanisms
- Detected Uncorrectable Error (DUE): detected but not correctable by the mechanisms and can cause failure, but since it is known to have occurred remedial action can be taken, for instance, if it occurred in a clean cache memory line, then the line could be marked as dirty since the datum exists in main memory.
- Silent Data Corruption (SDC): undetected and are therefore regarded as the most serious

There are also a number of error classifications that are used in the context of soft errors in memory:

- Single-Bit Upset (SBU): one bit of a memory word is corrupt
- Multi-Bit Upset (MBU): when more than one bit of the same memory word is corrupted
- Multi-Cell Upset (MCU): when more than one bit of different memory words are corrupted. This distinction between MBU and MCU is made since ECCs have a bound on the number of errors that can be detected and corrected, so that a MBU may exceed the capabilities of the ECC that protects it while a MCU with the same magnitude may invoke multiple ECCs but not exceed any of their capability of detection or correction.

2.2 The Physical Mechanics of a Soft Error

Soft errors in the terrestrial environment, i.e. around sea-level, are predominately caused by high-energy[1] cosmic neutrons or alpha particles, which interact with

[1]Usually considered to be above 1 MeV.

the silicon material of the victim transistor to generate energetic ions[2]. Ionisation of silicon atoms or the further ionisation of dopant atoms in a transistor will create one or more electron-hole pairs in the physical location of the radiation strike. If the electric field near the p-n junction is large enough, then it can prevent any electron-hole pairs created in the wake of the incident particle near to the p-n junction from recombining and possibly lead them to the transistor's contacts where they can change the logic value of the output signal [22]. Alpha particles originate from radioactive impurities in the packaging of a chip and neutrons from cosmic rays. Baumann in the mid-90s [19] discovered that the interaction of thermal neutrons[3] with Boron-10 that was used for doping the silicon could cause soft errors, which has helped its removal as the default doping material.

The neutrons are the result of the interaction of cosmic rays, which is a term used to encompass unknown energetic particles from outer space either of galactic origin or from the sun [9], with the atmosphere that produce complex cascades of particles. Those particles that hit the Earth's surface number only 1% of the original amount of cosmic ray particles and neutrons make up the vast majority (97%). The spectrum of neutron energy conventionally considered at sea-level spans 10 MeV to 1 GeV and has a flux of about 14 neutrons per cm^2-hour [23]. However, the neutron flux varies with the solar cycle[4], altitude[5], and location on Earth[6]. Ziegler et al. [9] provide a quick estimate that only 1 in 40 000 neutrons will hit a silicon nucleus within 10 μm of the surface of a Large Scale Integration (LSI) circuit[7], which at sea-level equates to 2.5 silicon-neutron hits per year for every cubic centimetre of active volume.

Neutrons and alpha particles interact with the lattice of the silicon in different ways: neutrons have no charge so their interaction is purely kinetic while alpha particles interact with both kinetic and Coulomb forces. This means alpha particles interact directly with electrons, while it has been shown that the majority of soft errors due to neutrons are caused by the inelastic collisions that create secondary particles [25]. Although most neutrons do not interact with silicon, they tend to generate more charge than alpha particles because the secondary particles created from a neutron hit have higher energies [2].

The mass and energy of the primary or secondary particle and the material in which it is travelling determines the magnitude of the disturbance the ion creates, called the Linear Energy Transfer (LET). The LET influences the amount of charge collected (Q_{coll}) typically within a few microns of the junction and can range between one to several hundred femtocoulombs [2]. Figure 2.2 shows the charge generation and collection processes for a reversed biased junction and the waveform of the resultant current pulse. Q_{coll} depends upon many factors including size of the device, biasing of the various circuit nodes, substrate structure, device doping, the type of ion, its energy, its trajectory, the initial position of the event within the device, and the state of the device.

The minimum charge required to alter the logic state of an element, such as

[2]An atom where the total number of electrons is not equal to the total number of protons.

[3]Neutrons that are very slow with an energy of a fraction of an eV

[4]Accounted for a 30% reduction in terrestrial cosmic ray flux during two years of the most intense solar activity recorded in the period between 1954 and 1990 [24].

[5]For example, from New York at sea-level to Denver, Colorado at 5 280 ft neutron flux increases by 3.4× [9].

[6]About a 2×variation.

[7]Considered to contain tens of thousands of transistors per chip.

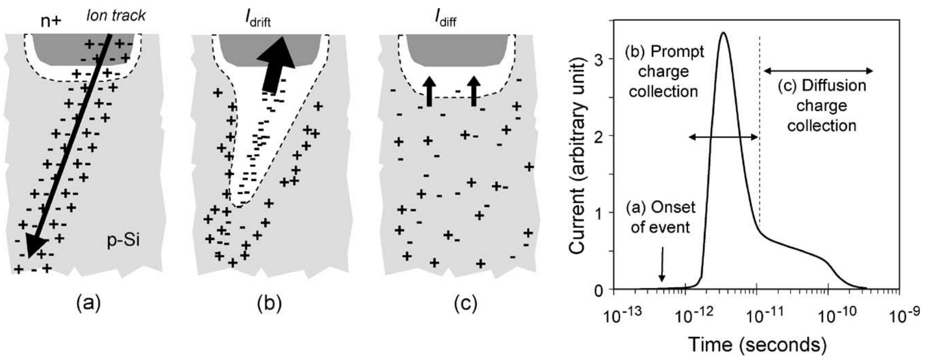


Figure 2.2: Charge generation and collection phases in a reverse-biased junction and the resultant current pulse caused by the passage of a high-energy ion (From [2])

a memory cell or combinatorial gate, is called the critical charge, Q_{crit}. This value is mainly dependent on the node capacitance, operating voltage, and the strength of feedback transistors. This value is normally found from injecting current pulses into a circuit model and is further explained in section 2.6 on page 22.

2.3 Importance of Soft Errors

To assess the importance of soft errors we look at: real life reports that highlight their industrial relevance; direct evidence in terms of failure data and trends in that data; and indirectly through public industry assertions, and more importantly, industry action directly related to soft errors.

Firstly, we highlight three examples of the impact of soft errors that have been publicly disclosed. A study in 1998 of nearly 600 people with implanted cardiac defibrillators recorded the occurrence of 22 soft errors, some of which were multiple bit errors and could not be corrected [26]. This caused alarm since a cardiac defibrillator can deliver a 30 J shock direct to the heart. In another example, in 2000, Sun Microsystems found that soft errors in the level two caches, which had defective error protection, had caused it's flagship servers to crash [3]. In a third example, Cyprus Semiconductor reported in 2004 that a single soft error brought a billion-dollar automotive factory to a halt once a month [27].

Unfortunately, detailed data on soft error failure rates are sparse, which makes it difficult to compare the magnitude of the problem to other fault sources. The first reason for the paucity of this data is that companies, who have the hardware and resources to perform the required testing, view the data as highly confidential, so normally only report normalised figures. Secondly, the soft error rate is dependent on many factors, ranging from the manufacturing process to the running software application, which makes a fair comparison difficult. A third aspect arises due to the unspecified type of errors or failures reported because the severity may differ, e.g. application crash or system hang.

Accelerated neutron testing[8] by Normand [16] in 1996 found in the region

[8]Irradiating an IC with much more particles than in the normal environment — explained

of 1 000 to 2 000 FIT/Mbit[9] for a group of nine SRAM and DRAM chips of sizes 256 Kbit and 4 Mbit. However, this can vary significantly depending on the vendor and underlying memory technology as Dodd et al. [28] reported that the SER of SRAM ranged from 10 to 1 400 FIT/Mbit. This was based on the accelerated neutron testing of SRAM chips from 250 nm to 140 nm, 6T CMOS to 4T poly resistive memory cells, and test voltages from 4.5 V to 1.4 V. For example, for 250 nm 6T CMOS SRAM at 1.5V the author recorded approximately 1 400 FIT/Mbit while the same vendor's 140 nm memory recorded approximately 1 000 FIT/Mbit .

For sequential logic, it was agreed in 2006 at the Workshop on Silicon Errors in Logic - System Effect (SELSE), where academia and industry that are actively concerned about soft errors meet, that 1 milli-FIT per flip-flop is representative in bulk CMOS technology, which is comparable to the above memory per bit SERs. There has been less research in measuring or calculating predictions of the contribution of combinational logic because memory has been at the forefront of shrinking and tends to occupy the majority of the chip area in modern SoCs, so it has been the focus of soft error analysis. However, Shivakumar et al. [17] in 2002 made the prediction that the SER attributable to all combinational logic found on their model of a microprocessor would be comparable to that of unprotected memory by the 50 nm technology generation because the scaling of combinational logic would catch-up with memory. More recent data from Intel [29], based upon accelerated testing, disputes Shivakumar's claim by reporting that the contribution is only 30% of the total SER of a modern microprocessor. How the SER varies with technology scaling is discussed in detail in Section 2.4.

Thirdly, if businesses have no desire to publicise data on soft errors, we can instead acquire indirect evidence of the importance of soft errors by examining what is openly stated and, more importantly, what experiments are performed and what level of protection their products feature. The International Technology Roadmap for Semiconductors (ITRS), which is a set of documents produced by the international industry, to promote pre-competitive research, has identified an urgent need to address the issue of soft errors since the 2005 edition [30]. Cisco have said that soft errors are their number one issue from customers regarding reliability. The down-time caused by these type of failures can cost customers up to $1 million per minute and quoted 0.2 soft errors/year/unit (40 000 soft errors per year with an average shipment of 200 000 units per year)[10]. Baumann [2], from Texas Instruments, who is well known within the soft error community states that the total permanent failure rate is typically in the range of 50 to 200 FIT, while "advanced processors with large multimegabit-embedded SRAM can easily have soft failure rates in excess of 50 000 FIT per chip". Based on his "typical" rate of 1 600 FIT/Mbit (which is in the range of the figures quoted above), this would imply that the processor chip has an equivalent error rate of approximately 30 Mbit of SRAM. Around 2005 when that paper was published the 65 nm Intel Pentium 4 had 16 Mbit of level two cache memory[31]; which from a budget of 50 000 FIT gives about an even split in SER between the level two cache and the processor (including its own internal memory structures).

further in Section 2.6

[9]1 FIT means one failure every billion hours that the device is in operation; thus 1 000 FIT/Mbit is one thousand failures every billion operating hours for every million bits of memory. The SER is normally given in FITs.

[10]Data from a panel talk given by a Cisco engineer at the SELSE 3 workshop in 2007

In terms of what is done, the UK's Department for Business, Innovation and Skills[11] has co-funded two projects since 2004 specifically looking at SEEs with a total funding of about £5 million over six years. A study of a fault-tolerant microprocessor examined the impact on availability of transient faults over an error rate of 2 000 000 FIT to 10 000 000 FIT [32]. The most significant indicator of the concern over soft errors is the ubiquity of error detection and correction codes (ECCs) for memory. This has become a standard feature not only in high-end severs, but can be found in desktop and mobile processors, for example, ARM's Cortex-R4 supports either parity or ECC on its level one internal memories [33] . Intel's latest Itanium microprocessor released early 2010, codenamed Tukwila, apparently has similar per chip SER as its predecessor, but with almost three times the number of logic circuits [11]. It features Double Error Correction (DEC) ECC covering the 30 MB of on-die cache, SER-hardened circuits, background memory scrubbing, and a new point-to-point system interconnect called QuickPath with Cyclic Redundancy Checks (CRCs) on the data, transaction retry, self-healing links, and clock fail-over. The change to DEC ECC indicates the number of errors that they need to protect against has increased. These features are targeted particularly to reduce the SER rather than other sources of faults, such as power rail or supply noise.

2.4 Trends in the Soft Error Rate with Technology Scaling

A modern computer system normally consists of external DRAM, possibly on-chip SRAM (in the form of cache memory), sequential logic (latches and flip-flops), and combinatorial logic (e.g. NAND gates, multiplexers, etc). Differences in circuit structure require separate calculation of the SER for each of these elements. This section presents how the per element (e.g. bit, latch, gate) SER and the per system SER due to all those elements varies with technology shrinking. The SER of each element is calculated using [21]:

$$Element\ SER = Nominal\ SER \times Timing\ Derating \times Logical\ Derating$$

Unless stated otherwise, the SER data only represents the nominal value. For DRAM and SRAM, timing derating is close to one [34]. Methods for estimating the timing derating of other elements can be found in Nguyen et al. [21]. Logical derating is usually not calculated because it depends on how the elements are used by the software under the instruction of the user.

2.4.1 DRAM

Baumann [35, 2] reports that the Single-Bit Upset (SBU) SER per bit of DRAM is exponentially decreasing, but the number of DRAM bits per system is exponentially increasing at about the same rate to give a roughly constant SER contribution from DRAM per system.

[11]Known as the Department of Trade and Industry (DTI) at the time

2.4.2 SRAM

Seifert et al. [18], from Intel, reported that in 2006 there seemed to be agreement between academia and industry that the SBU SER per bit of SRAM is decreasing. This was based on their own accelerated neutron testing and alpha particle simulations and on data from five other sources, which are based on a mix of accelerated testing or simulations of neutrons and/or alpha particles. These other sources include work from Baumann [2] who finds that with recent SRAM technology no longer using Borophosphosilicate Glass (BPSG) the SBU SER per bit has saturated and is possibly decreasing. Additionally, from accelerated testing data of SRAM test chips at geometries from 350 nm to 130 nm, Hareland et al. [36] find that from 250 nm the inverse dependence between SER and voltage begins to weaken and beyond 130 nm, where stored node charge is less than 10 fC, the SER per bit will decrease because the number of particles that can cause upsets will saturate as the bit area decreases. However, Hazucha et al. [37] find that this per bit rate increases: from accelerated neutron testing of a 90 nm SRAM and using data from other references the SER per bit of SRAM from 250 nm to 90 nm increased by 8% per generation at nominal voltage (V_{DD} decreased by 14% per node).

Dodd et al. [28] found that the SER of SRAM from several vendors at the same technology generation varied substantially. They performed neutron accelerated testing of SRAM from different vendors to find the SER ranged widely from 10 FIT/Mbit to 1 400 FIT/Mbit . They tested memory manufactured at 250 nm to 140 nm, using 6T Full CMOS to 4T Poly R, and test voltages from 4.5V to 1.4V. This substantial variation highlights the impact that the design and process methodology has on the failure rate. Additionally, Hazucha et al. [37] measured a 22% difference in SRAM SER when they rotated the chip 180° with it still normal to the beam.

Despite whether the per bit SER has saturated or is decreasing, the total SRAM SER will increase given that the amount of SRAM bits in a system continues to rise and occupy a greater die area [2, 6]. Although Shivakumar et al. [17] suggest that the SBU SER per bit of SRAM is falling in deep submicron technologies, they predict that the SER of the total SRAM in a system has already saturated at the 180 nm node. They base their findings on Hazucha and Svensson's semi-empirical equation [38] and scale an Alpha processor by quadratically increasing the number of transistors, so their prediction does not take into account further SRAM memory (such as higher levels of cache) that may be found on the whole chip.

Seifert et al. [18] additionally found that the rate of MBUs (due to soft errors) per bit of SRAM is increasing, due to decreasing cell pitch and Q_{crit}. Two years later, based on accelerated testing of a 45 nm High-κ-Metal Gate (Hκ-MG) SRAM process, Seifert et al. [39] reported that SRAM neutron SBU and MBU Soft Error Rates (SERs) per memory bit were unchanged. Neutron SRAM SER per bit still decreases approximately 2× with technology generation. MCU upset rates remain unchanged despite a 2× increase in SRAM integration density, which keeps MCU rates less than or equal to 10% of SBU rates according to the authors.

2.4.3 Sequential and Combinational Logic

Although similar in design to SRAM, flip-flops and latches (sequential logic) tend to be more robust because they are constructed with larger transistors so that the charge needed to upset, Q_{crit}, is larger. However, it is expected that they will become more sensitive to soft errors as technology shrinks. Mitra et al. [34] believe that although there is evidence that the SER per bit of SRAM is saturating or decreasing, there is less consistency in the literature for latches and flip-flops.

From simulation and experimental data of flip-flops between 180 nm and 65 nm, Baumann [35, 40, 2] found a linear increase in the per bit nominal sequential SER with scaling, which would reach the SER per bit of unprotected SRAM by 65 nm. Thus, he believes that sequential logic will be the dominating factor in measuring the system failure rate once SRAM is sufficiently protected with ECC. To assess the validity of this belief, the vulnerability factors of the sequential logic need to be calculated and the system SER would need to be calculated accounting for the fact that there is usually much less logic than SRAM on a SoC. From accelerated neutron testing of a 90 nm test chip of latches, Hazucha et al. [37] find, on average, an 18% increase in SER per latch for every 10% reduction in supply voltage. Cohen et al. [41] performed accelerated alpha testing of latches in a 250 nm process to show that the SER per latch increases exponentially with decreasing voltage — about 2.1 decades per volt. Scaling an Alpha microprocessor from 350 nm to 250 nm, the corresponding drop in nominal voltage accounted for approximately 75% of the decrease in Q_{crit}, thus voltage scaling is seen as the primary concern for the increase in soft errors. Using the projections from the Semiconductor Industry Association (SIA) when the paper was published in 1999, the authors forecast that the latch SER could increase 100× from 1997 (250 nm) to 2012 (50 nm). Although the revised projections (from the 2009 ITRS [10]) put the 2012 DRAM half-pitch smaller than thought (36 nm instead of 50 nm), the increase in latch SER will not be as much as Cohen et al. thought because most of the dependence is on V_{DD}, which has not scaled as much (0.9 V instead of 0.6 V).

Contrary to the above data, Seifert et al. [18] found that using Intel's process technology the average SER per latch is fairly constant from the 130 nm to 65 nm nodes. This is in agreement with the simulations performed by Shivakumar et al. [17]. Additionally, Karnik et al. [42] carried out accelerated neutron experiments of a 180 nm flip-flop test chip to find that the SER per flip-flop would likely be constant if V_{DD} scales by 0.8×. According to the ITRS roadmaps [43, 10], the supply voltage ("V_{DD} high-performance") has scaled from 1.1 V in 2001 to 1 V in 2009 — a reduction of only 0.9×, suggesting that the SER per latch has slightly decreased.

The study of combinational logic has in the past received less attention because it has been seen as the least important contributor to the system SER because: there is less of it on a chip than other element types, it tends not to use as aggressive feature dimensions, and there are several masking processes that lessen the error rate. Mitra et al. [34], also from Intel at the time, state that the system SER trend is dominated by SRAM and sequential elements, and the contribution of combinational logic is considerably smaller. The contributions to the system SER are reported to be 40% from unprotected SRAM, 49% from sequential logic, and 11% from static combinational logic. However, there is no further discussion or reference to how these figures were calculated.

In 2002, based upon neutron simulation Shivakumar et al. [17] predicted that “the SER per chip of [combinatorial] logic circuits will increase nine orders of magnitude from 1992 [600 nm] to 2011 [50 nm] and at that point will be comparable to the SER per chip of unprotected memory elements [SRAM bits and latches]”. The author’s model takes into account electrical masking and latch-window masking, but not logical masking. Since ECC is employed to reduce the memory SER, this would mean that soft errors from logic dominate. Therefore, this has provided evidence for some to focus on further analysis and mitigation techniques. Such a rapid increase is attributed to a quadratic decrease in the Q_{crit} of combinational logic, which has scaled more rapidly than that of memory and is estimated to converge on memory around 50 nm. In their model, this capacitance reduction results in a 10^6increase in SER per combinational logic chain and coupled with a 100 times increase in logic chains per chip gives a nine orders of magnitude rise. Their focus is on combinational logic chains between processor pipeline stages, so the SER is based on those SETs that are subsequently stored by the latch at the end of the chain. Shivakumar et al. predict that the SER of a combinational logic chain (ranging from six to sixteen fan-out of four gates long) would be comparable to that of an unprotected SRAM bit and latch by the 50 nm technology generation. They base their chip-level results on quadratically increasing the number of transistors that make up an Alpha processor for each subsequent technology node. The transistors are allocated to 20% combinational logic and to 80% memory (SRAM cache and latches) based on an examination of the die photos of a 350 nm Alpha processor. It is unclear if these proportions hold for each new processor design rather than a straight scale of the same design. Based upon simulation too, the results from Seifert et al. [18] show that for combinational logic the SER per pipeline stage is flat for some simple test circuits and decreasing for an adder test circuit. The most recent empirical data from Intel refutes Shivakumar’s prediction. Gill et al. [29] find that the SER of one single logic chain is less than 10% of a nominal (without taking masking into account) unhardened latch SER at 1 GHz in Intel’s 32 nm process. In this calculation they assume a latch’s SER is half of a flip-flop’s SER and that the P/N ratios of the gates are balanced to give similar low and high transition times (a skewed P/N ratio increases the SER, e.g. a $4\times$ ratio gives $10\times$ higher SER). Using a figure-of-merit based on “typical clock speeds and fan-in statistics for modern microprocessors”, Gill et al. calculate an upper bound on the contribution of total combinational logic SER to be about 30% of the total nominal latch SER on a chip.

Although there is some consistency in the literature as to the nominal per bit SER of DRAM and SRAM, there is fewer coherent experimental evidence on which to form an agreement on logic SER trends. However, the continued integration of more transistors in advancing technology will result in a higher chip-level SER. The SER contribution from DRAM and SRAM can be substantially lowered through the use of ECC, which has meant that logic soft errors have increased, relatively, in importance. Therefore it is considerably valuable to investigate the effect of soft errors in logic and suitable methods for mitigating their impact.

2.5 Attaining Dependability

Avizienis et al. [1] group the means to attain dependability into four major categories:

- Fault prevention means to prevent the occurrence or introduction of faults
- Fault tolerance means to avoid service failures in the presence of faults
- Fault removal means to reduce the number and severity of faults
- Fault forecasting means to estimate the present number, the future incidence, and the likely consequences of faults

The authors explain that the difference is that fault prevention and tolerance aim to provide a dependable service, while fault removal and forecasting aim to provide confidence in that ability. An example method of fault removal is verification during development and fault injection on a circuit model is an example of fault forecasting.

Fault prevention methods specific to soft errors include reducing alpha particle emissions by purification of the materials and equipment used in the semiconductor manufacturing process. Baumann [2] states that through contamination reduction, alpha particles emissions have reduced from levels as high as 100 particles per hour per cm^2 to below 0.001. However, this is not a guarantee because certain byproducts of the original contamination may regrow, so emissions from materials must be monitored over several months. While it is possible to shield against alpha particles[12] by adding extra material between sensitive areas and likely locations such as solder joints, it is not cost effective to shield against cosmic neutrons (it has been found that a foot of concrete lowers the intensity of cosmic neutrons by 1.4× [44]). Radiation hardening is normally used for electronics that are used in aerospace applications and is done by modifications to the process, such as using different materials or substrate structures (e.g. Silicon on Insulator (SOI)), or by altering the design, such as using larger drive transistors in SRAM cells or different transistor layouts which can reduce the charge collection efficiency. Baumann [2] says "the majority of process solutions seldom reduce SER by more than five times so their use does not justify the expense of additional process complexity, yield loss, and substrate cost".

The most commonly employed technique in memory is the use of ECC, which is a fault tolerant method since it does not attempt to prevent the occurrence of the fault in the first place, but when a fault does happen there is a guarantee that a certain number of incorrect bits can be masked or at least detected. ECC that allows two errors to be detected and a single error to be corrected typically results in a greater than 10 000×reduction in error rates [2].

Since ARM provides hardware IP, it has little control over the implementation and manufacture, therefore the focus has been on fault tolerant methods.

2.6 Measuring the Soft Error Rate

The SER of electronic devices can be evaluated in four complementary ways [45]: modelling, accelerated testing, life testing, and from field reports of reliability.

[12]In silicon, a 10 MeV alpha particle can travel up to 100μm — typical impurities can emit alphas with energies between 4 and 9 MeV

Life testing — where hundreds of chips are evaluated over a long time period under normal radiation conditions — and reports from customers are not favourable methods. The former is prohibitively expensive due to the large amount of chips required and the long time needed to gather statistically significant results. For instance, a batch of 500 SRAM chips with a typical failure rate of modern SRAM chip (5 000 FIT) would cost over $200 000 for the hardware alone and would take about nine months to produce decisive results [45]. Although failure data from customers is valuable in providing tangible evidence and is an impetus for research in this area, it is difficult to find root causes from customer data, or infer from this data the SER of other chips.

Accelerated testing involves the irradiation of ICs with hadrons (particles with a strong nuclear force) that have a substantially larger flux than present at sea-level, to artificially induce errors over a short period of time. This is the preferred route to generate a “real” failure rate figure because it is based upon actual hardware and is cheaper than life testing, for example a popular laboratory costs in the region of $10 000 per day. Although protons comprise less than 5% of the total flux at sea-level — nearly all the rest are neutrons — they are more readily available than neutron sources and have near equivalent behaviour at energies above 30 Mega electron Volt (MeV). Thus below this energy neutron beams are used and above this proton beams are commonly used. The JEDEC Solid State Technology Association JESD89 standard [20] was introduced to ensure best practise and comparable reporting between the results of accelerated testing. This method is valuable to both industry and academia because of the credibility of the results, but requires fabricated technology which is difficult for academia and companies like ARM to access and is not helpful when input is needed to lower the SER at an early stage in a design. However, it is helpful in calibrating and validating models.

Computing the SER of a chip requires analysis in two areas [22, p48]: first, the calculation of the intrinsic failure rate of the elements that comprise the chip and, second, the calculation of the vulnerability factors which reduce the intrinsic failure rate, often called the ‘derating’ or masking effects. Modelling the intrinsic SER is comprised of modelling the radiation sources, their geometry and charge track production, the charge transport and collection, and circuit failure [46]. This is a two step process where the critical charge, Q_{crit}, is computed for the element and then this is mapped to a corresponding FIT rate. Determining Q_{crit} is usually achieved using a circuit level simulator, such as SPICE, to inject current pulses that mimic a particle strike into sensitive nodes of the circuit so that the smallest charge that changes the state is the critical charge. See Freeman [47] for a description of this for a bipolar SRAM circuit. Q_{crit} is different for DRAM and SRAM and varies with pulse shape, circuit parameters, and variation in technology parameters, temperature, power supply voltage, etc [25]. Srinivasan [25] describes three approaches to accomplishing the mapping of Q_{crit} to FIT rate in increasing order of sophistication: semi-empirical, device or circuit level, and chip level. The semi-empirical models are commonly based upon data from accelerated testing and have to be re-calibrated for each new technology generation. Popular models are the Burst Generation Rate (BGR) method from Ziegler and Landford [15], the Neutron Cross Section (NCS) method from Taber and Normand [14], and Hazucha and Svensson’s model [38]. IBM’s Soft Error Monte carlo Modeling (SEMM) tool is a chip-level predictive simulation method that does not require parameter fitting from empirical testing as it models from first principles [48] taking all device, process, and layout aspects into account. However, SEMM is targeted towards memory and sequential logic elements.

Measuring the SER is an expensive and time consuming process, because to gain credibility models must be based on empirical data from accelerated testing — which requires access to testing facilities, hardware, and people with the right skills — or be based upon nuclear particle reactions, like SEMM, which has taken many physicists many years to develop. This makes measuring the SER prohibitive for ARM.

2.7 Conclusion

Data on the occurrence of soft errors in the terrestrial environment are rare — because their transient nature makes it difficult to record and because companies are reluctant to release information on problems with their products. Since there are many sources of failure within modern computer systems, it is unlikely that soft errors dominate. However, it is a major concern of computer hardware manufacturers since faults caused by the environment have increased relatively in importance as permanent faults from production has fallen. With the continuing trend in technology scaling — shrinking device sizes and higher integration of components — it is predicted that the vulnerability of a computer system to failure from soft errors will rise. Although this will not be of substantial concern in the use of some technology devices, such as mobile phones, the increasing ubiquity of technology means that it is important for those companies that use computer devices in applications that have stringent requirements on dependability, as will be explored in the next chapter.

Chapter 3

Commercial Background

The purpose of business is to create and keep a customer

— Peter F. Drucker

A primary objective of this EngD is to develop an argument to support the commercialisation of research, which has at its core the target customer or market. Therefore, this chapter explores the macro and micro environment that predisposes an investigation into the impact of an unreliable IP interconnect. It represents information gathered from internal and external sources, including internal marketing, customers, and external publications. Before this examination, we give an overview of ARM in terms of what it produces and its business model. The second section examines the trends which effect the semiconductor industry and specifically at where the target markets — automotive and storage networking — sit on that line. The third section outlines what these target markets do, why they care about reliability, and examples of related current customer activity with ARM.

3.1 About ARM

Spun out from Acorn Computer Group in 1990 as a joint venture with Apple Computer to develop Reduced Instruction-Set Computer (RISC) microprocessors, the company now describes itself as the world's leading semiconductor IP supplier. From the ARM website [49]:

> The ARM business model involves the designing and licensing of IP rather than the manufacturing and selling of actual semiconductor chips. We licence IP to a network of Partners, which includes the world's leading semiconductor and systems companies. These Partners utilize ARM IP designs to create and manufacture system-on-chip designs, paying ARM a license fee for the original IP and a royalty on every chip or wafer produced. In addition to processor IP, we provide a range of tools, physical and systems IP to enable optimized system-on-chip designs.

To date, over fifteen billion ARM based chips have been shipped. More than 660 licences of IP have been sold; 87 of them in 2009 and of that amount, 30 of them for mobile applications. The rest are split across ARM's three other main markets: embedded (e.g. white goods, smart cards, and automotive), enterprise (e.g. WiFi routers and hard disks), and home (e.g. digital TV, digital cameras, and games consoles). The company reported a revenue of about \$490 million in 2009, of which about 50% is from royalty, 34% from licensing, and the rest from activities like support services [50]. In the same year, 63% of unit shipments made it into the mobile end-market, while 15% of units went into enterprise applications, 16% into the embedded end-market, and 6% into home applications. While ARM is seeing growth in smartphones and other mobile computers, it is beginning to see non-mobile usage increase as a proportion of its technology shipped and end-market diversity will be critical to its long term success.

3.2 Trends

3.2.1 Background

As of the third quarter of 2009, Intel's Atom processor represents the state-of-the-art in widely commercially available CMOS logic manufacturing at 45nm. By the end of 2009, Intel plans to release the first processors from their "Westmere" family which will be based on their second generation 32 nm high-κ metal gate process technology [51]. ARM's partners predominately use either TSMC or Common Platform technology who offer logic processes down to 45 nm as well, and are developing the capabilities for 32 nm and 28 nm, with the latter offered as a high-κ metal gate or silicon oxynitride process [52]. From 1999 to 2010, the electronics industry has seen the physical gate length of CMOS transistors used in micro-processor units fall from 120 nm to 27 nm [53, 10]. From then on, the 2009 edition of the ITRS projects that the gate length will half every seven to eight years.

Technology scaling had resulted in an exponential growth in the number of transistors and, in combination with further competition, has driven more product features. This has led to escalating design, verification, and implementation complexity and cost. While desktops and laptops have shifted to multiple processors in the last few years, many embedded computer systems have incorporated multiple heterogeneous processors for longer. In these embedded systems, the complexity and number of processor cores and other IP blocks has risen. Consequently, this has increased the complexity and size of the system that is responsible for the management of data transfer between these IP blocks, referred to as the IP interconnect. In 2003, ARM introduced the AXI protocol in the third edition of the AMBA IP interconnect standard to meet the higher performance required of future SoCs.

3.2.2 Automotive

There are many different electronic systems, commonly called Electronic Control Units (ECUs), within an automobile that vary not only in function but in semiconductor process, geometry, and complexity. Current mid-range automobiles contain more than seventy ECUs and this rises to over one hundred on higher-end vehicles. An ECU typically contains a micro-controller unit and a Application Specific Standard Product (ASSP) along with a number of power and signal converter devices to interface with the sensors and actuators. The micro-controller unit and ASSP will contain one or more processors while the other converter devices will tend not to. ARM IP is used in these micro-controller unit and ASSPs in nearly every system of a vehicle, e.g. airbags, chassis stability, brakes, dashboard, body (communications systems, energy management), smart sensors, adaptive cruise control, and infotainment. Examples of commercial products include: a Bosch airbag chip that provides an interface for the micro-controller unit to the sensors and can induce the firing of the airbags, which uses a 350 nm Bipolar CMOS DMOS (BCD) process technology [54]; while a 32-bit micro-controller unit from ST [55] designed for engine management and chassis control is on 90 nm. In this work, we consider those embedded systems that are responsible for safety critical functionality and are subject to international safety standards (described in Section 3.3.1). This includes the braking system (i.e. Anti-lock Braking System (ABS)), engine control, chassis stability control (i.e. Electronic Stability Control (ESC)), and airbags.

Automotive semiconductor suppliers tend to use a geometry several nodes behind the cutting edge because the design cycle is longer than other consumer products, safety requirements mean they tend to be more risk adverse to using the latest technology, but above all it is because automotive products need to be Automotive Electronics Council (AEC) Q100 certified [56]. This certification stipulates electric and electronic stress tests that must be met and requirements such as operating temperature (-40 C to +125 C for systems under the bonnet as opposed to 0 C to +70 C for standard commercial grade products), lifetime expectations of over ten years, protection from high loads, and long retention embedded flash memory. This extra cost to automotive semiconductor suppliers means they tend not to certify every process node and use an older one as long as possible.

The automotive industry has seen an increasing use of electronic components, not only to substitute for more expensive and less reliable mechanical counterparts, but in an effort to obtain higher performance, more efficient, and safer vehicles such as through the use of ABS for braking and ESC for the prevention of skidding, which will be mandatory by 2011 according to European Union (EU) legislation [57]. This trend is set to continue in the future with research into so called "x-by-wire" systems such as steer-by-wire and brake-by-wire where there is no longer a mechanical connection between the driver's controls and the system under control. A related trend is that of consolidation throughout the vehicle of electronic devices to reduce overall costs, whereby functionality from separate chips is conflated. An important development to help manage the growing electric and electronic complexity is Automotive Open System Architecture (AUTOSAR) [58] — a standardised automotive software architecture. The use of hybrid vehicles requires more complex engine control software, which will require higher performance processors. The increasing use of electronic systems and consolidation of those will lead to more complex SoCs and therefore larger and more complex IP interconnects.

3.2.3 Storage Networking

The storage networking market refers to the set of technologies that allow computers, such as servers, to access high capacity shared storage across a network, as apposed to storage directly attached to a computer. A Storage Area Network (SAN) is the most common remote storage architecture for large corporations and data centres. It comprises the network itself which is made up of switches, routers, and hubs and the computer and storage systems at the periphery. The devices within the network will incorporate a control processor, such as a PowerPC, and application specific hardware for the data path because throughput must be maximised. The network itself will either be Fibre Channel — can run on both twisted pair copper wire and fibre-optic cabling — or gigabit Ethernet. The storage system incorporates hundreds of Intel x86 or PowerPC processors that manage the transfer of data between the network and the storage array, i.e. hard disks, typically Redundant Array of Independent Disks (RAID) based, or tape drives. ARM IP is found in the network interface cards of the computer servers and storage servers, within the switches that connect the storage server to the storage array, and within the hard disks.

Storage network systems tend to lag a node behind the latest manufacturing processes available for SoCs because some elements such as the network interface cards contain analogue electronics for the optical I/O which does not scale easily, some devices incorporate a lot of IP which takes a significant amount of effort to design and verify, and the profit margins are high enough that the incentive to cut costs through scaling is not as strong as it is for companies such as Intel or AMD.

Most of the drive to increasing complexity is occurring at the periphery of the network. For instance, the network interface supports more protocols and there is convergence of protocols, the main one being Fibre Channel over Ethernet — an encapsulation of Fibre Channel frames over Ethernet networks. Moreover, the continual growth in storage capacity drives more complex storage switches and more complex and numerous processors for the software which manages data transferred between the network and storage array. The growth in complexity to increase performance will again result in larger and more complex IP interconnects.

Although slowing, the drive to continue shrinking manufacturing geometries has not diminished, and neither has the vigour of the semiconductor industry to produce higher performance and more functionality to remain competitive. The storage networking market is near the forefront of technology scaling and has seen growth in storage capacity, compute performance, and supported protocols. The automotive industry will face the challenges of scaling later and are consolidating an increasing amount of electronics. Both shrinking and increasing complexity for each market segment will increase the number of transistors susceptible to transient faults and potentially adversely impact the reliability of a device.

3.3 Markets

Set against the trends of reducing feature size and rising complexity, the following section will outline some background information on each target market and why both are concerned about reliability. This will include examples of important customers in contact with ARM who are developing products due for release within the next couple of years that specifically require techniques to detect and correct errors on their AXI bus.

3.3.1 Automotive

According to market analysts, automotive semiconductor sales peaked in 2007 at \$20 billion, but fell in 2009 to \$15.8 billion[1] and are forecast to rise 16% by the end of 2010 to \$18.4 billion [60]. Automotive semiconductor revenue tracks light vehicle production, which from the same source peaked in 2007 at around 67 million units, but fell to 54 million in 2009. Based on growth in vehicle sales from India, Brazil, and China, the authors estimate automotive semiconductor revenue to reach \$35 billion by 2017. From the earliest year that they present data (1997), the compound annual growth rate of automotive semiconductor revenue has been 5.9%, while it is 1.88% for light vehicle production[2], which indicates the increased use of electronics.

From this and other data we can gather some clues as to the cost of electronics in automobiles. A crude estimate would be to divide automotive semiconductor sales by light vehicle production, which from the previous source for 2009 would be \$300 per unit[3]. Gartner [61], another market research company, report that the power electronics module controller which manages the transfer of electric power to the motor to be worth about \$400 of semiconductor electronics, however, the cost of the vehicle is not given to put this in context. On top of this \$400, there would be the cost of the rest of the motor control, the safety systems, and infotainment. Gartner [62] provide further clues from estimate of the breakdown of automotive semiconductor revenue by end application equipment: 18% of revenue on airbag and ABS modules, 14% on engine control electronics, and 20% on satellite navigation.

Although it is difficult to acquire information on the proportion of revenue that ARM attributes to the automotive and storage networking markets, from the figures in Section 3.1 it will not be a major part of overall revenue. However, it is expected to increase and growth in non-mobile markets is being actively pursued. This works seeks to accelerate the growth in market share and revenue in automotive applications; chassis systems in particular.

The most important requirement for ARM's automotive customers is safety because the failure of their systems can result in injury or death, repair costs, legal action, and the damaging publicity. A recent example is from Toyota, where two problems surfaced this year: one with the ABS in which a software upgrade was needed and the other was a "stuck" accelerator pedal. Toyota announced that the former problem will result in around 360 000 cars to be recalled in the US, UK, and Japan alone [63]. The company estimate that it will loose about \$2 billion on warranty expenses and lost future sales, but this does not include potential litigation expenses [64]. For comparison, Toyota's net income for the financial year ending 2010 was \$2.5 billion [65].

Safety is difficult for end users to measure and compare when purchasing

[1]For comparison, total worldwide semiconductor revenue reached \$226 billion in 2009, down about 10% from 2008 [59].

[2]These figures take the 2009 data to be the average of the 2007 peak value and the 2008 trough.

[3]Using the short scale where one billion equals 10^9.

an automobile and because of the nature of this property there is an inherent minimum level expected. The automotive industry has to adhere to international standards on electronic safety such as IEC61508 and more recently ISO22626, which is specific to this sector. These standards specify targets for failure rates that companies have to meet for the different types of safety functions that their systems implement. In IEC61508, these are called Safety Integrity Levels (SILs) and there are four of them. The higher the SIL, the lower the acceptable failure rate. For example, the ABS in a car has to be implemented to SIL 3. SILs are used to determine what techniques should be used in order to meet the required failure rate. ARM does not certify its products to these standards, but rather provides the infrastructure in their IP to enable its partners to develop physical devices that will have to be certified.

Latency is the most important performance requirement from these customers, while they are much less sensitive to clock frequency and area overhead. The electronic safety systems embedded with the automobile, such as those that control the engine or brakes, are hard real-time systems, which means that they must complete their tasks within a certain time frame otherwise failure can result. These systems have a complex schedule of tasks and have to process many events activated by interrupts, so the hardware needs to minimise the time taken to react to an interrupt otherwise valuable time for controlling the actuator is lost.

A recent study by Allgemeiner Deutscher Automobil-Club e.V. (ADAC) (the German equivalent of the The Automobile Association (AA)) found electronics account for 40% of the breakdowns it attended [66]. Soft errors are one source of electronic failure among many and automobile manufacturers expect that the issue be addressed by it's semiconductor suppliers.

A specific customer request has helped to drive interest in ARM as to the value of increasing the reliability of the IP interconnect. They require protection of the data transported on the on-chip bus between the processor and main memory for their automotive platform, which is to be used for engine control. The main memory in their system will already have ECC protection and the new ARM processor to be used will support ECC on the internal Tightly Coupled Memories (TCMs) and caches. They now require similar ECC protection for the instructions and data read from and written to main memory by the processor while it is transported across the on-chip bus interconnect. This customer has asked ARM to implement the ECC logic in their new processor core to minimise the extra latency incurred and because a different bus protocol is used. It is hoped that the additional functionality offered by the product proposed in this work will strengthen ARM's case for this customer to switch to AXI. Until then the development of these features for this new processor core must be aligned with this work to ensure compatibility.

3.3.2 Storage Networking

Data about the storage networking market is more difficult to ascertain. This is due to the many different devices that the market encompasses: hard disks, network switches, and network interface adaptors. It was estimated for 2009 that the total number of hard disk drives shipped to be around 560 million units and for total industry revenue to be about \$30 billion [67]. By 2013, it is expected that total shipments will rise to 900 million units and revenue to increase to \$40 billion. Examining enterprise disks specifically, market research shows that

24 million units were shipped and produced nearly \$3 billion in revenue, which gives an average selling price of \$120 [68]. Enterprise disks account for about 3 to 4% of total hard disk shipments and about 10% of total revenue, which Gartner forecast to remain constant in the future. Their most likely scenario from 2009 to 2013 is a compound annual growth rate of 4% for shipments and 3% for revenue. Moving onto enterprise Ethernet switches, it was estimated that this industry took in about \$3 billion in 2009 (a 22% fall from the previous year) and shipped about 70 million units (a 16% fall) with mid to large companies making up nearly all of the customer base [69]. Finally, it was calculated that the SAN switch and adaptor market revenue was about \$2 billion in 2009 and is expected to rise to \$6.5 billion in 2014 [70]. Summing the enterprise figures, this gives about an \$8 billion industry, but does not include all appropriate devices. Although this is less than half of the automotive semiconductor industry, it is still a valuable market and growth area for ARM: it has seen its market share in storage grow from 19% in 2007 to 65% in 2009 while its networking market share has remained constant at 20%. This work seeks to accelerate market share in the storage networking market, thus it is important to understand the market's needs.

Availability and data integrity are very important issues for ARM's storage networking customers. Down time can cause the end customer, such as the financial sector, large sums of money. Availability is the fraction of time that a computer system is ready to complete work, while data integrity is to ensure that no bits of information are lost. These properties are used to differentiate between competitors. Performance, in terms of throughput, is also a major requirement.

In contrast to automotive manufacturers, these customers use semiconductor processes that are closer to the forefront of SoC technology. Soft errors have attracted the attention of storage networking companies because there is a body of research that finds soft errors to be a significant and growing reliability problem, which will adversely affect the availability and data integrity of their systems. Since these companies differentiate themselves from competitors on these metrics, they are interested in mitigating the impact of soft errors.

Several important customers have expressed an interest in adding features to the interconnect to improve data integrity. For instance, one of the world's largest hard disk drive manufacturers is currently engaged with ARM in the design of their new hard disk controller. This customer requires protection of the data transported between processor and storage because they are concerned about soft errors. Since they use ARM's interconnect fabric, they have requested ARM to implement this feature. Interest from these large companies provides credibility of the value in developing a product to improve the reliability of the IP interconnect.

Chapter 4

Soft Errors in Logic

The SER is an important metric for quantifying the reliability of a system. It allows comparison between different devices and with competitors' offerings, which can lead to a more informed choice by a customer and drive improvement in the device's reliability. It was envisaged that the SER of current or future ARM IP — at least the IP which is used by customers who care about reliability — is measured and would complement information already given on the performance, power, and area of a product.

Calculating the SER of a chip requires computing the intrinsic error rate of all elements of a chip — memory, sequential logic, and combinational logic — and the vulnerability factors for each different element, which reduce the intrinsic error rate since not all faults produce errors. Historically, the focus has been on calculating the SER of DRAM and SRAM because they were at the forefront of technology shrinking, memory composed most of the chip, and an error in memory is more likely to cause system failure. However, the impact of soft errors manifested in combinational and sequential logic on the system failure rate is predicted to become more important than those in memory [17, 2] because these elements are becoming more susceptible to soft errors and the failure rate of memory can be reduced several magnitudes with the use of ECCs. In combinational logic, three phenomena act to mask the SET before realising an SEU: Logical Masking (LM), Electrical Masking (EM), and Latch-Window Masking (LWM). The logical masking of a circuit is a function of the type of gates used, how they are connected, and their input values. The former two factors are controlled by the design of the component and the logic synthesis tool, which maps the design to a gate-level representation. Recent work found that LM had an equal or greater impact than EM and LWM combined on reducing the SER [71]. The influence of the gate-level design of a circuit on the logical masking of SETs was explored by performing SFI on different synthesised adder designs. Considering only LM provides an upper bound on the SER of a combinatorial circuit. Used in conjunction with other metrics, such as power, performance, and area, knowledge of the SER would help a designer to weigh the benefit between choosing a different implementation based on the error rate or applying a mitigation technique to the existing implementation.

This work makes the following contributions:

- linking measurable features of a gate-level representation of a circuit to the logical masking of soft errors in combinational logic

- identifying that SER of combinational logic may be underestimated if only a single chain of gates into a storage element is considered and not its fan-in logic cone
- exploration of the feasibility and value of techniques to measure the SER for ARM IP
- the development of a gate-level statistical fault injection methodology, which garnered interest within ARM and we believe is valuable as part of a future fault analysis methodology

The primary reasons for using adders are: their circuit complexity is a good balance between simulation time and meaningful analysis; there are many known different ways of implementing the same function; and that the synthesis tool can easily generate the different implementations. Thus providing a relatively inexpensive method of exploring the relationship between circuit design and LM. The adder has additional relevance because it a crucial unit of a microprocessor's pipeline, which is utilised for many operations other than explicit add instructions, such as in the computation of the address for the next instruction. Therefore, if the adder were to experience a fault and cause an incorrect result, the severity would be great. This study used statistical fault injection by simulation rather than accelerated testing because it benefits from higher controllability, observability, cost efficiency, and increases confidence by enabling testing before implementation.

The next section describes the method in which logical masking is measured. The results are presented in Section 4.2, where we find that the largest difference in error rate between the 32-bit adder implementations is 3.1× and that the error rate decreases with higher synthesis frequency[1]. We also find that across all adders approximately 1.5 errors are caused on average for every fault and that nearly 30% of all faults cause more than one error. This is followed by a discussion on the relation between the circuit topology and logical masking, and then finally the conclusion.

4.1 Method

The SER of a circuit due to particle strikes can be calculated as [72]:

$$SER = R_{PH} \cdot \alpha \cdot P(SE) \tag{4.1}$$

Where R_{PH} is the particle hit rate, α is the fraction of hits that result in an SET and $P(SE)$ is the probability of that SET becoming a SEU or soft error. Neutron cross section [14, 73] and burst generation rate [15, 74] are two commonly used methods to calculate the effective particle hit rate ($R_{PH} \cdot \alpha$).

It is assumed that every particle hit is effective, i.e. $\alpha = 1$, which is modelled as a fault injection. In combinational logic, $P(SE)$ encapsulates the three

[1]For synchronous digital circuits, the synthesis frequency specifies to the synthesis tool the maximum delay that all paths of combinatorial gates between registers must meet, such that the delay of the slowest path is less than the target clock period minus the setup time of the register. The run frequency of a circuit specifies the frequency of the clock signal fed into the registers that determines the rate at which they store their next value.

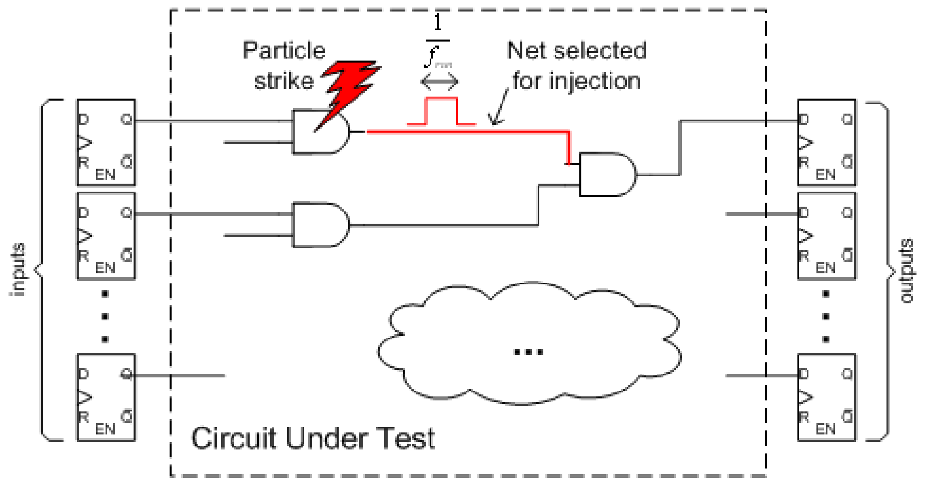


Figure 4.1: Model of fault injection on a gate-level circuit. Here, the width of the injected fault is equal to the run clock period, i.e. $\frac{1}{f_{run}}$.

masking phenomena. To only consider LM, the width of the SET pulse is set to the same time as the run clock period, i.e. the reciprocal of the run frequency, to discount LWM and EM is not inherently taken into account in a digital gate-level simulation (it requires additional modeling).

Faults are injected into a post-synthesised gate-level representation, which will have a strong correlation with the corresponding physical circuit. The adder implementation will influence the types of gates selected and the way that they are connected. The statistical fault injection tool injects the same amount of faults into each circuit and the location of an injection is uniformly random. This ensures that we only take into account the logical composition of the circuit.

A non-trivial software tool was created to perform fault injection on gate-level circuits, detect errors, and store the results. In a Verilog testbench the synthesised circuit is instantiated twice, one is the golden reference design and the other is subjected to fault injection. State elements in each circuit are compared to detect mismatches, i.e. errors in the circuit under injection. This software framework is written in the C programming language and interacts with a state-of-the-art Verilog simulator using the IEEE standardised Verilog Procedural Interface (VPI). The framework exposes a number of functions that can be called from the Verilog testbench to setup and control the fault injection. A fault is modelled as shown in Figure 4.1 by selecting the output net of the gate or register of the chosen target, inverting its value and holding it for the specified duration before releasing it. Faults are not injected on input wires, including the clock, because this would mean the fault was generated from an element outside the circuit.

A script was created to synthesise a given circuit and generate a file containing all potential target elements, which the software tool reads in. Another couple of scripts run the fault injection simulation and parse the error log report to store it in a database which allows quick and easy analysis of the results.

The input values have a uniform random distribution and change every clock cycle, which serves to discount its influence on the error rate. Each adder implementation completes the operation in one clock period. The system waits for long enough after a fault injection before performing another to ensure that the effects of a fault do not overlap. The software tool was designed to be as generic as feasible so that it can be used on any gate-level circuit, which allows it to be used by others within the company. The tool amounts to about twelve hundred source lines of C code and the scripts amount to about a thousand lines of source code. The software was tested by scheduling a number of faults at both a random and predetermined time and location, then examining debug messages and all relevant signals in the waveform output from the Verilog simulator.

Logical masking will be measured in terms of Incorrect Results (IRs), which is defined as the number of wrong results from the addition, no matter how many output bits are in error, i.e. if a fault corrupted three output bits, then this is counted as one incorrect result. The IR rate is normalised by the number of target gates, not area, because an injection site is randomly chosen from the set of gates rather than being influenced by sensitive area. Of course, α will be proportionate to sensitive area, but this work only analyses the effect of logical masking on $P(SE)$.

The adder implementations are created by Synopsys's Design Compiler synthesis tool with a TSMC 130 nm process. Using the Synopsys Design Ware IP Library, Design Compiler is forced to select an implementation during synthesis. The following seven adder implementations were compared:

- Ripple-carry (rpl)
- Carry-look-ahead (cla)
- Fast carry-look-ahead (clf)
- Brent-Kung (bk)
- Conditional sum (csm)
- Ripple-carry-select (rpcs)
- Delay-optimised flexible parallel-prefix (pparch)

4.2 Results

The results for the adders are based upon 100 000 samples, so that with a confidence level of 99%, this gives an estimation error of 0.816%. The results show the rate of errors that appear at the clock cycle immediately following the one in which the fault was injected because this is when the result would be forwarded onto the next stage in processing. Since the time at which the fault is injected is random and the width of the fault is $\frac{1}{f_{run}}$, a small number of errors show up in the next clock cycle. All the circuits within this section were synthesised at a frequency of 100 MHz. Later, we will analyse the effect of synthesis frequency on the error rate.

Figure 4.2 shows the variation of logical masking with each adder implementation. They have been sorted in descending order by the number of incorrect results per gate from left to right and have been normalised to 'pparch', i.e.'pparch' has the most number of wrong answers while 'csm' has the least. There is a 3.1× difference in the number of incorrect results per gate between the largest

Table 4.1: Size and error rates for each 32-bit adder implementation. Impl = Implementation; Area in synthesis units; MFO = Mean Fan-out; MPL = Mean Path Length.

Impl	Area	Gates	MFO	MPL	IR (%)	IR (%)/Gate
pparch	1249	160	1.39	23.96	88.36	0.55
rpl	1467	192	1.32	23.96	77.67	0.40
bk	1562	235	1.60	14.31	81.06	0.34
clf	1626	246	1.67	13.97	81.26	0.33
rpcs	1669	224	1.42	20.16	73.46	0.33
cla	1702	268	1.64	14.62	72.68	0.27
csm	3166	345	1.99	7.19	60.98	0.18

Table 4.2: Statistics on the errors per fault for each adder implementation. * Mean errors per fault. ** Percentage of faults that cause more than one error.

Impl	Mean EPF*	> 1**
pparch	1.67	35%
rpl	1.54	28%
bk	1.53	28%
clf	1.50	26%
rpcs	1.51	26%
cla	1.51	29%
csm	1.39	21%
Mean	1.52	28%

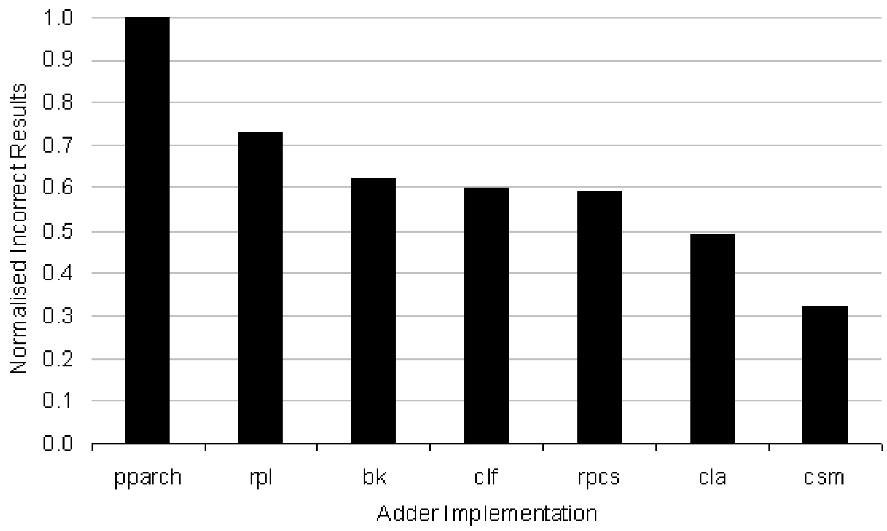
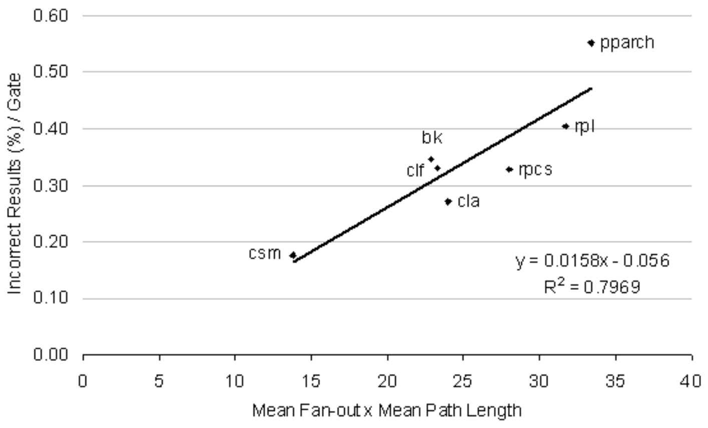
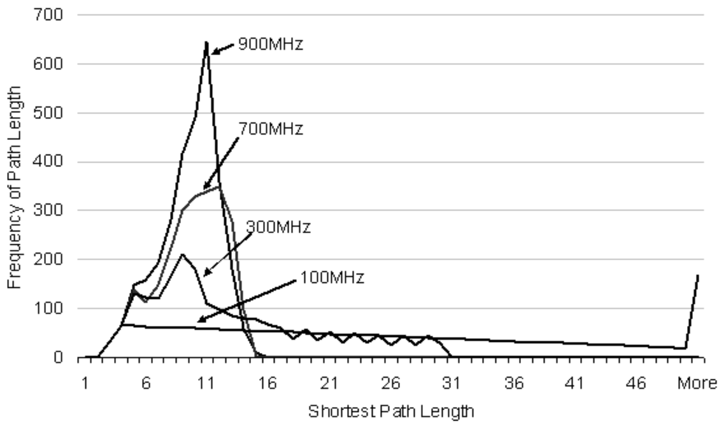


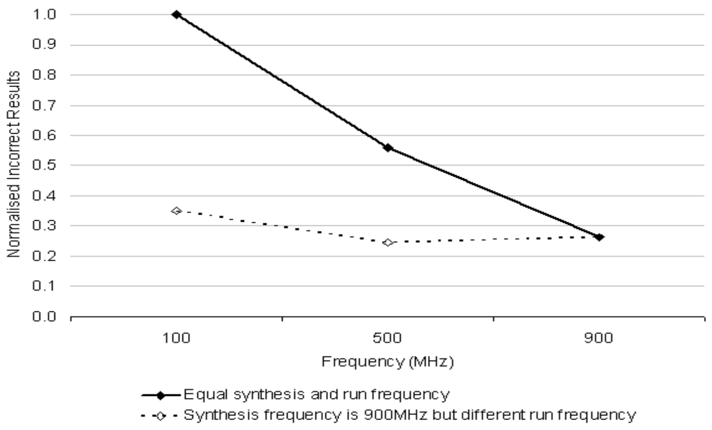
Figure 4.2: Incorrect result error rate per gate for each 32-bit adder implementation (normalised to 'pparch')



(a) Correlation between the product of the mean (shortest) path length and mean fan-out with the incorrect results for each 32-bit adder



(b) Shortest path length distribution for 'pparch' synthesised at different frequencies



(c) Variation in incorrect results with synthesis and run frequency for the 'pparch' adder

Figure 4.3: Data from SFI

implementation, 'csm', and the smallest 'pparch'. The largest difference between the remaining implementations is about half of that. From Table 4.1, 'csm' is about 2.5× larger than 'pparch' (because it is composed mostly of multiplexers).

The mean number of errors that result from each fault is given in Table 4.2 for each adder implementation. From this data we can see that every fault causes approximately 1.5 errors across all adders. Moreover, nearly 30% of all faults cause more than one error. This data is important because it shows that the combinational logic SER will be higher than some previous research estimates if that work only takes into account single chains of gates leading into flip-flops or latches and does not consider the impact of the fan-out logic cone. This is somewhat analogous to MBUs that occur in memory in that a single soft error in combinational logic can cause multi-bit upsets in sequential logic.

4.3 Discussion

Across all seven 32-bit adder circuits logical masking eliminates between 12% and 40% of all faults injected. These results show that LM accounts for a significant reduction in the SER but there is a wide range in impact. We can see from Table 4.1 that as the size, both area and gate count, of the adder implementation increases the IR rate decreases. However, normalising the IRs by the gate count reveals that there are other factors that influence the IR rate since if it were only number of gates that mattered, then it would be expected that the normalised IR rates would be similar.

Logical masking will be influenced by signal propagation probabilities (i.e. if a fault appears at the input of a gate then how likely is it to change the value of the output) and input values. We explore the influence that a circuit's complexity or composition may have on its logical masking by viewing a circuit as a directed acyclic graph where the gates are the vertexes and the interconnecting wires are edges. Then we can characterise the composition of a circuit using common measures of graph topology, such as the degree distribution and mean path length. In circuit terms, the degree distribution corresponds to the fan-in and fan-out distributions while we define the mean path length as the shortest distance (in terms of number of gates) between a target cell and an output register.

The fan-in distribution of a circuit will show the number of inputs a gate is likely to have. As the gates of all circuits under consideration are nearly all of two inputs this information does not help to reveal a relation with logical masking.

The majority, about 70%, of the gates in the most resilient circuit, 'csm', have a fan-out greater than or equal to two, while for the rest of the implementations the majority of gates have a fan-out of one. Counter intuitively, this would suggest that an increase in the mean fan-out of a circuit would give a lower error rate. To explain this, we need to look at the association of path length with the error rate.

Figure 4.3a shows that there is a fairly strong positive correlation between the product of mean fan-out and mean path length with the error rate. This would suggest that a lower mean fan-out and shorter distances between gates and outputs will result in higher masking. So although 'csm' has a higher mean fan-out than all other implementations, its effect on the error rate is outweighed by its low mean path length. Convergence of paths is likely to have an impact and is naturally taken into account because we are using circuit simulation, however it is not considered in the analysis.

At a synthesised frequency, f_{synth}, of 100 MHz, the distribution of path lengths is rather flat and wide for all implementations except 'csm'. To meet the target f_{synth} the synthesis tool only needs to ensure that the slowest path — the path with the greatest delay — meets the target maximum delay. At a synthesis frequency much lower that the maximum possible for the given circuit, the tool does not have to spend much effort optimising the paths for timing, which results in a wide distribution of path lengths. As the frequency at which the adders are synthesised is increased up until the maximum permissible more paths will have to be optimised to meet the target delay, therefore the distribution of path lengths will narrow and the mean path length reduces to meet the target delay. Figure 4.3b shows this for the implementation with the highest error rate, 'pparch'. Since the mean path length reduces with increasing f_{synth} the error rate decreases and the largest difference in error rate decreases too, such that at 900 MHz the difference between 'csm' and 'pparch' is $2.1\times$. The solid line in Figure 4.3c shows this change in error rate with synthesis frequency for the 'pparch' implementation with the run frequency, f_{run}, equal to f_{synth}.

If we consider that the objectives of a designer within this context are to maximise the performance of the circuit, minimise the area, and minimise the soft error rate, then the designer can control these constraints by varying f_{synth} and f_{run}, which reveals the following dependencies:

1. increasing f_{synth} increases the potential performance and lowers the mean path length and thus LM, but increases the area (which will make the circuit more likely to be hit by a particle)

2. increasing f_{run} improves the performance, but may decrease the latch-window masking effect, which will increase the error rate

If we synthesise 'pparch' at the maximum frequency of 900 MHz, but run it at various lower frequencies, we find that the error rate is less than the circuits where $f_{synth} = f_{run}$ (compare both lines in Figure 4.3c) due to the advantageous changes in circuit composition made by the synthesis tool at the higher speed. This benefit diminishes as f_{run} approaches f_{synth}. It was expected that the IR rate should decrease as f_{run} increases due to a rise in LWM. When $f_{run} \ll f_{synth}$ the width of the injected fault pulse will be significantly greater than the maximum delay between input and output registers such that nearly all fault injections located anywhere in the circuit and at any time during the clock period will be latched at the output register. As f_{run} approaches f_{synth} the delay through the circuit becomes comparable to the fault width, so the location and timing of the fault should have a greater influence on the probability of an error, that is LWM should increase. We can see from the dashed line in Figure 4.3c that the middle data point does not follow from this reasoning and thus further analysis of the impact of LWM is required.

It may be of benefit to a designer targeting a lower performance to run the circuit at a lower speed than that synthesised for, but at the cost of an increase in sensitive area. The overall gain will depend on the corresponding increase in the effective particle hit rate. This will also be important to designs that employ dynamic frequency scaling as the variation in run frequency will impact the logical masking of errors.

4.4 Related Work

Approaches to measuring the SER in combinational logic aim to determine the factors α and $P(SE)$ (Equation 4.1) for different circuits and technology nodes. The approaches can be categorised into dynamic and static methods, although some use a mixture of both.

Dynamic methods include SFI using circuit simulation tools such as SPICE or gate-level simulation and acceleration irradiation testing. Liden et al. [75] used heavy ions to induce SETs in a custom circuit using 1 μm technology to measure the proportion that cause soft errors, which was estimated to be between 0.7×10^{-3} and 2×10^{-3}. The circuit was of low complexity with about 2 400 transistors and 40 flip-flops and was clocked at 5 MHz. This testing constituted early work into the effect of EM and LWM and demonstrates that on an old process the probability was low, although not negligible. Using a laser and heavy ion source to perform accelerated testing of two inverter chains in 1 μm process at 5 V, electrical masking was observed to have a noticeable effect on the output error rate even in a circuit designed for minimum pulse attenuation [76]. Although not large enough to merit the calculation of the masking factor, the authors claim that the attenuation may have a significant effect when “slower” gates are used such as four-input NORs. In a separate paper, the same authors investigated the variation of LWM with clock frequency to find that the error rate in sequential logic was independent of frequency — although this was only tested up to 1 MHz — but was linearly dependent in combinational logic, which was tested up to 20 MHz [77]. The test circuit was a string of sixteen inverters that has a gate length of $0.7\mu m$. In another investigation on the scaling of LWM with clock frequency, Gadlage et al. [78] claim to corroborate the work of Buchner et al. [77]. This was conducted using two test chips manufactured on 180 nm CMOS irradiated with heavy ions to mimic alpha particles and protons to approximate neutrons and with the clock frequency varied from 1 MHz to 300 MHz. In the former case with alpha particles, there was a 10× increase in the cross-section of combinatorial logic for an equivalent rise in frequency. At the maximum frequency, the cross-section of the sequential logic was approximately an order of magnitude greater than combinatorial logic. Using linear extrapolation, the authors estimate the frequency cross-over point, i.e. where the cross-sections meet, to be 800 MHz. In the latter case for proton testing, they found the same relationship for sequential logic, but did not observe enough errors from combinatorial logic to be statistically significant. While there is convincing evidence that the SER of combinatorial logic increases dramatically with clock frequency, the evidence for sequential logic hides some important details, for instance, Buchner et al. only tested up to 1 MHz after which the error rate tailed off, but claimed the test chip was not well suited and Gadlage et al. performed their tests with the clock to the flip-flops turned off, thus only measuring the static error rate.

While dynamic techniques are used to calculate both α and $P(SE)$, static methods predominately focus on $P(SE)$, taking into account all three constituent masking factors or concentrating on a subset. Rao et al. [79] describe an algorithm that propagates descriptions of current pulses across a circuit to model SETs that takes into account all masking. Their algorithm was tested on 22 circuits from the MCNC [80] and ISCAS-85 [81] benchmarks[2] and the SER was averaged over 1 000 random inputs vectors. They report that the SER ranged from 1.4×10^{-6}FIT for a small circuit with 222 gates and 1 output to 4.6×10^{-4}FIT for a 32x32 multiplier with approximately 25 500 gates and 64 outputs. Their work found that the number of outputs had a more significant effect than the number of gates, e.g. a circuit with 3 379 gates and 224 outputs has an SER comparable to that multiplier with 4.53×10^{-4}FIT. A major benefit of static techniques is the short run-times; their algorithm displays linear run-time with circuit size. To illustrate, the afore mentioned 32x32 multiplier took 200 seconds to calculate the SER over the 1 000 inputs values, however, the SFI framework developed in this work took several days on the same size multiplier. The authors claim that their method of characterising the shape of the current pulse using a Weibull probability density function is more accurate than a single parameter pulse width or simple trapezoidal shape. Although after traversing several gates a trapezoidal would be accurate, the authors claim that the accuracy of the initial strike requires the resolution their method provides. The large number of pulse shapes that result from their analysis of an entire cell library across a range of incident particle energies is reduced to a tractable amount by selecting some representative samples.

Another static probabilistic method is proposed by Miskov-Zivanov and Marculescu [82] who use Binary Decision Diagrams (BDDs) and Algebraic Decision Diagrams (ADDs) to perform an exhaustive symbolic analysis that avoids the repeated execution over a range of inputs values and the extensive cell library characterisation of Rao et al. [79]. Based on given input values probabilities, pulse duration, and pulse amplitude their framework can calculate the probability of error for each output, the impact of individual gates, and overall circuit error rate. The authors claim that the accuracy of their method is improved through joint treatment of the three masking effects, in contrast to other work that treats the phenomena independently and find that a separate treatment gives a measurement error between 4.5% to 40% (averaged across different input value distributions and for a pulse width of 80 to 125 ps, i.e. 33% to 50% of the clock period). The execution time is in the same order as Rao et al. In agreement with the results presented here, Miskov-Zivanov and Marculescu find that the output error probability decreases with greater circuit complexity and attribute this to more significant electrical and logical masking. From the testing of 22 different circuits from the same benchmarks as above, the authors report that the output error probability "can vary from less than 0.1%, for large circuits [560 gates and 8 outputs] and short glitches (20% of cycle time), to about 30% for very small circuits [45 gates and 4 outputs] and long enough glitches (50% of cycle time)". Both Rao et al. and Miskov-Zivanov and Marculescu have two test circuits in common, however the they do not provide enough parameters required to compare their SERs or derive $P(SE)$ to compare the level of masking to our results.

Miskov-Zivanov and Marculescu's work is further developed to include sequential logic [83] and to consider Multiple Event Transients (METs) [84], which occur when a single particle strike causes an SET in more than one combinatorial

[2] A number of circuits whose composition has been described to allow comparison

gate.

Some examples of work that use a combination of dynamic and static methods include Massengill et al. [85] who present an analytical technique that determines the shape of the current glitch based on collected charge and dynamic simulation of a gate-level representation of a circuit to measure the masking across input values. This is used to calculate the error cross-section of a circuit, which is the fraction of the circuit area that upon a particle strike will cause an observable error. This metric is given instead of the SER to allow comparison at different altitudes and locations, which affects the particle fluence. The authors tested their tool on a 4-bit bit-slice microprocessor circuit synthesised using a 250 nm technology library, whose layout area vulnerable to soft errors (the drain areas of the transistors) was determined to be about 8% of the total combinatorial layout area ($6\,500\mu\text{m}^2$). Varying the collected charge up to 1 pC, the error cross-section peaked at about 0.4 pC to be between 0.08% and 0.12% of the total combinatorial layout area, depending on the processor opcodes ran. If we use a neutron particle flux at sea level of $56.5m^{-2}s^{-1}$ [38], then the peak SER of the processor is $0.0012 \times 6500 \times 56.5 \approx 440$ FIT. Nieuwland et al. [86] propose a method that uses simulations to calculate Q_{crit} for different gates and different input values (they found that the failure rate of a NAND-2 gate varies up to 12× with input combination) and then input this into an analytical model to support the SER reduction that their mitigation technique provides. However, they do not take LWM into account and do not show results for comparison. A similar concept, but accompanied by more extensive analysis is carried out by Ramanarayanan et al. [87], who describe a methodology that pre-characterises transistors for SET sensitivity at the device level and then feeds that into an analytical method at the gate-level that calculates the SER of a comb circuit taking into all masking. Across five small circuits (less than ten gates) and eight larger circuits (92 to 1 200 gates), the level of masking ranged from 86% to 99.9%.

Assessing the relative importance of the masking effects, Miskov-Zivanov and Marculescu [71] found that in three of the four circuits tested the LM had approximately the same effect as EM, while the remaining circuit showed a 3.5× difference. The error probability due to only LM varied between 26% and 88% across three different input value distributions for two circuits tested. Varying the setup and hold times of the output flip-flops from 6 to 16 ps (2.4% to 6.4% of the clock period), the LWM did not significantly affect the error probability for three of the four circuits; for the other it doubles. Using a probabilistic method to purely consider LM, Kim et al. [88] find that for the carry output of a full-adder the relative probability of error varied up to nearly 3× for all input values, however, for an even smaller circuit (two inputs, one output, three gates) this was nearly 220×. This could suggest that the variation with input values decreases with greater circuit complexity, although the difference in complexity was small.

The greater complexity of some of the static methods outlined here creates a barrier for their adoption, while industry seems to prefer the credibility that accelerated testing affords. SFI is essentially simulated accelerated testing that offers a trade-off between these two. The methodology developed would need to incorporate models for the other masking phenomena and have some performance improvements to be able to comprehensively measure the SER of combinational logic circuits more complex than the adders used in this work.

4.5 Conclusion

It was found that the implementation of the circuit has a significant impact on logical masking in combinational logic as the largest difference between the 32-bit adders was 3.1×. Logical masking is only one component that determines whether an SET becomes a soft error in combinational logic and so presents the theoretical worst case. The sensitive area, which is used to calculate the effective particle hit rate, must also be taken into consideration by the designer when choosing an implementation based on the SER.

Measures of the circuit's topology were explored to relate its composition to logical masking. The data suggests that a lower mean path length and lower mean fan-out results in a higher amount of logical masking. As highlighted in the case of the 'csm' implementation, which had the highest masking, the effect of a lower mean path length dominates an increase in mean fan-out. It was found that increasing the synthesis frequency results in a lower mean path length and therefore lower error rate.

Additionally, we find that nearly 30% of faults injected cause more than one error and on average about 1.5 errors per fault. Research that does not consider the fan-out of combinational logic will considerably underestimate its SER relative to sequential logic and memory cells.

The logical masking, and thus the soft error rate, would help a designer to gauge the benefit between choosing a different implementation or applying a mitigation technique to the existing one, such as altering the composition of the circuit. It is worth noting that it may be possible to lower the SER by using Dual Modular Redundancy (DMR) or Triple Modular Redundancy (TMR) on a smaller adder which has a lower LM rate rather than using a larger adder with a higher LM rate. Using SFI is a time consuming method to calculate the logical masking, but using the properties of the circuit structure identified here and possibly others a static analysis method could be developed, which would be much quicker than fault simulation.

We have developed a methodology to estimate the logical masking for any digital circuit and applied it to a non-trivial and industrially relevant issue of logical masking variation and its causes, the results of which was published in a highly relevant and leading workshop on soft errors in logic (SELSE). This a first step in developing a comprehensive methodology for measuring the SER of any digital IP; the next steps would be to account for the other combinatorial masking, sequential logic, and memory. Doing so could use some of the masking models mentioned in the previous section as well as other known modelling frameworks, but would require additional resources to complete. Based upon the knowledge and capabilities gained from this work and from interaction with relevant people in the company, it was proposed and agreed that developing a comprehensive method for measuring the SER of ARM IP did not present a high enough return on investment for ARM for a number of reasons:

- ARM does not produce the hardware required to accurately measure the SER of its IP, such as a processor, using accelerated testing so it would be costly to assign the required labour for a significant period of time, to purchase a small batch of the chips, and access the testing facilities. There is also a lack of experience to design the necessary experiments.
- It would be difficult to test ARM IP using existing products from its

partners because there would not be adequate observability. Additionally, existing products contain older ARM products, but there would be more incentive for ARM to quantify the reliability of new IP to encourage uptake.

- Although modelling is not as valuable, it would still require significant time and effort to develop a comprehensive methodology that encompassed the many parameters of the process, memory, processor, and layout, which this work represents a part of.

- Ultimately, there was not a strong enough requirement from those customers who care about reliability that provides the financial incentive to do so.

Building upon the knowledge and capabilities gained, the subsequent research does present a high enough return as validated by significant customer interest in a product derived from the work. As described in the next chapter, the research continued to examine transient faults in combinatorial logic, but focused onto an investigation of the fault tolerance of a widespread ARM product: the AMBA on-chip bus communications system.

Chapter 5

The Impact of an Unreliable IP Interconnect

The work in this chapter continues to look at the impact of soft errors in logic, but focuses on an important ARM product, the AMBA IP interconnect, to create more commercial value for the company and relevance for the EngD. The AMBA IP interconnect is important to ARM because it provides leverage to get customers to use its other IP in conjunction, including processors and other peripherals, and it is present in the vast majority of SoCs in the embedded devices that ARM targets. It occupies a significant portion of the logic area, comparable to that of ARM's embedded series of processors, and is set to grow with the increasingly complex systems that it inhabits. As will be shown, the consequences of faults on the IP interconnect can be severe and warrant enough risk to be of a concern to those customers who care about dependability. This work seeks to examine the effect of decreasing reliability on current or future ARM interconnects and low-cost mitigation schemes to, at least, maintain levels of reliability.

This work makes the following contributions:

- demonstrates that the risk of faults on the IP interconnect is a concern because the impact on the reliability of the SoC can be severe
- argues that end-to-end EDAC codes cannot be employed for certain AXI signals and therefore other techniques are required
- development of several novel techniques that detect or correct faults arising on the logic path of certain AXI signals that would escape detection by end-to-end EDAC
- composition of a proposal for a product that increases the fault tolerance of the AXI bus matrix, which was successfully transferred out of the R&D department and could be commercially available in two years

This chapter is arranged as follows: Section 5.1 gives a technical overview of ARM's IP interconnect technology; Section 5.2 provides an explanation of why the fault tolerance of the interconnect is important; and Section 5.3 goes in to detail on why it is not straightforward to employ known techniques such as ECC and presents the features of the proposed product.

5.1 Background

ARM Fabric IP is a portfolio of products that consists of on-chip communication interconnects[1], memory controllers, Direct Memory Access (DMA) controllers, cache and interrupt controllers, and the software used to design and verify a SoC with the afore mentioned hardware IP. AMBA is an open on-chip communication standard, currently in its third version, that is an umbrella for four interface protocols: AXI, Advanced High-performance Bus (AHB), Advanced Peripheral Bus (APB), and Advanced Trace Bus (ATB). The ATB interface specification allows data to be traced for debug purposes. APB is a low bandwidth interface primarily designed to configure registers in a peripheral. AXI and AHB are protocols designed to allow on-chip components, such as a processor and memory controller, to communicate with each other. AXI is newer, more complex, and supports higher data traffic throughput than AHB. Since the AMBA standard is publicly available, it is possible for other companies to implement their own IP that adheres to it. ARM implements the AMBA 3 standard via it's NIC-301 product, which allows for a component that uses any of the four interface protocols to communicate with any other. ARM's AMBA Designer software tool provides a Graphical User Interface (GUI) to configure, connect, and generate the Verilog Register Transfer Level (RTL) for all Fabric IP.

We target the AXI protocol and it's implementation, NIC-301, because it is designed to replace AHB and so ARM wants to migrate customers to the newer protocol. AXI is described briefly in the next section.

5.1.1 Overview of the AMBA AXI Protocol

The AXI bus protocol is burst based and has five independent unidirectional channels – three for writing and two for reading – that carry the address/control and data phases. Every transaction has address and control information on the address channel (AW for the write channel and AR for the read channel) that describes the nature of the data to be transferred. The data is transferred between master and slave using a write data channel (W) to the slave or a read data channel (R) to the master. In write transactions, in which all the data flows from the master to the slave, the AXI protocol has an additional write response channel (B) to allow the slave to signal to the master the completion of the write transaction or an error. A burst is made up of a number of transfers, which is a unit of data whose width is specified on the AW or AR channel. See Table 5.2 for a description of signals on each channel. The signals have been grouped into types as they may feature on more than one channel, e.g. the signal type is prefixed with the channel abbreviation so ID features on all channels: AWID, ARID, WID, BID, and RID. Those signals types with a * suffix are referred to as the "control signals".

AXI uses a two wire handshake, using the VALID and READY signals, to exchange information between the sender and receiver. VALID is used to indicate when valid data or control information is available and the destination uses the READY signal to indicate when it can accept the data. Both the read data channel and the write data channel also include a LAST signal to indicate when the transfer of the final data item within a transaction takes place.

[1]"interconnect" is used to refer to the component that allows IP blocks to communicate, not to metal wires

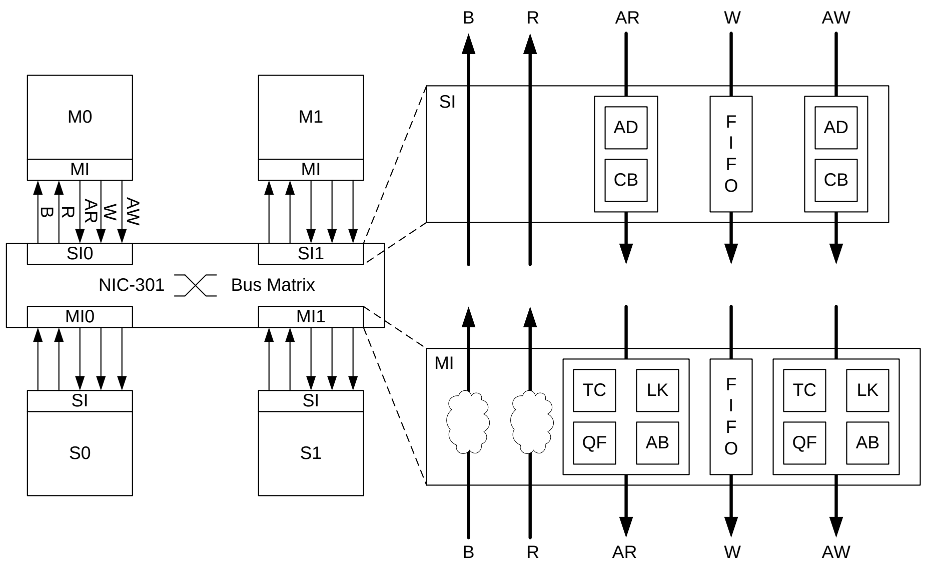


Figure 5.1: A block diagram of the NIC-301 bus matrix with two AXI masters (M0 and M1) and two AXI slaves (S0 and S1) connected to it. MI = Master Interface. SI = Slave Interface. Top right is a block diagram of a SI. Bottom right is a block diagram of a MI. Certain logic such as multiplexers are not shown for simplicity. See Table 5.1 for an explanation of the abbreviations used.

The standard also features a sideband signal, USER, up to 32 bits on each channel that allows the SoC designer to extend the protocol to carry implementation specific information, such as parity or ECC bits.

5.1.2 Inside the NIC-301 Bus Matrix

The NIC-301, commonly referred to as a bus matrix, is an ARM product that allows components using different AMBA interface protocols to communicate with each other. The left-hand block diagram in Figure 5.1 gives an example of two masters and two slaves connected to the NIC-301 bus matrix. A master could be another ARM product such as a Central Processing Unit (CPU) or Direct Memory Access Controller (DMAC), while a slave could be a customer's memory controller or a peripheral like a Universal Serial Bus (USB) controller. These masters or slaves need to implement one of the AMBA protocols — shown in the figure as a master interface (MI) or a slave interface (SI) — and can be verified against the available AMBA compliance tests. These protocol interfaces then are attached to the opposite interface on the bus matrix, for example a Master Interface (MI) connects to a Slave Interface (SI). The two diagrams on the right-hand side of the same figure give the main sub-components of the SI, top, and MI, bottom, of the bus matrix. Table 5.1 explains the abbreviations used and the function of each sub-component. It is important to get an overview of the construction of the bus matrix because it helps to understand some of the mitigation techniques described in Section 5.5.

Table 5.1: Explanation of the components in Figure 5.1

Abbreviation	Name	Function
MI	Master Interface	The logic that implements the master side of the protocol used. Connects to a SI.
SI	Slave Interface	The logic that implements the slave side of the protocol used. Connects to a MI.
AD	Address Decoder	The logic that maps the input address to an output destination. The memory map is specified during design time.
CB	Content addressable Buffer	A special memory unit that is used to store information about a transaction so that the protocol rules are adhered to.
FIFO	First-In First-Out	A queue that holds the write data when needed.
TC	Transaction Counter	Counts the transfers in a transaction which is used to control LK, QF, and the handshake.
LK	LocK logic	Implements the functionality of the AXI protocol that allows atomic access to a slave.
QF	Quality-of-service Filter	Allows some basic management of communication flow.
AB	Arbiter	The logic that controls which master can access which slave.

5.1.3 The Inherent Fault Tolerance of the AXI Bus Matrix

Before a transient fault progresses to an error it has to avoid being masked by the bus logic – either logically, electrically, or from the latch-window effect. Logical masking takes effect in various ways: the fault has to occur when the bus is being utilised, it has to occur at certain locations in the logic, and at certain times during a transaction. For instance, an erroneous AWLEN value received by a slave occurs only when there is a AW transaction in progress, i.e. AWVALID from the master is high and AWREADY from the slave is high, when a fault has occurred in the logic path of the AWLEN signal, and at such a place that ensures the glitch reaches the slave during the latching time of a register. During this window of vulnerability, the fault could cause an error to change the data value in one of two ways: to an invalid or incorrect value. An invalid value is not defined as possible or meaningful in the protocol specification, while an incorrect value is one that is valid but not what the sender intended. For example, AWBURST is a two bit signal where only three of its values are meaningful, so an error that results in a change to the unused value would be invalid. Invalid errors violate the protocol and are normally checked for at simulation using protocol checks to verify the correctness of the implementation against the specification. It is also possible to synthesise these protocol checks into hardware and have them active while the system is running to either catch errors due to incorrect implementation or for fault tolerance. Alternatively, if AWLEN were to change from zero to one, then this would still be valid, but is obviously incorrect and protocol checkers therefore cannot detect this error.

Finally, it is possible for a fault to occur on a VALID signal when the bus is not utilised, but this would only cause unknown data to be sampled if the receiver is waiting for a transaction, i.e. has READY high.

The inherent logical masking of the protocol and bus matrix further reduces the chance that a fault will manifest as an error and complicates the calculation of its probability since the type and pattern of traffic is dependent upon what components are connected to the bus matrix and how the software utilises the bus.

5.2 The Effect of Logic Soft Errors on the Interconnect

At a high level, the risk that a fault can induce failure is the product of the frequency or probability of occurrence and the severity of such an event. The complexity of robustly calculating the absolute SER of electronic devices was highlighted in the previous chapter. In that work, netlist-level SFI was used to measure the fault masking to compare the circuits, i.e. if a fault were to happen, then this data provides the likelihood of it propagating to an output where it is manifested as an error. Applying the same method to different configurations of bus matrices would yield how likely errors are and how they were manifested, which would be used to inform the creation of mitigation techniques. Performing SFI at the netlist level here was not achievable because the size of netlist is too large to simulate within a reasonable amount of time, there are numerous configuration options that affect the design, and the netlist is dependent upon its placement on a chip and thus on the placement and types of the other IP blocks present.

Instead, moving to a higher level of abstraction, errors were injected into the RTL of the bus interconnect, but only on the signals which appear in the AXI protocol standard, which we shall call protocol-level error injection. This is error, not fault, injection because it does not concern the implementation as the input and output signals between the bus interconnect and master or slaves are directly changed. This method allows a quicker discovery of the impact that errors originating from the interconnect have on the reliability of the system.

The protocol-level error injection framework was developed using the Specman/e language and software, which allows for a quick way to change the values of the AXI signals in Verilog during simulation, to monitor signals or events, and to interpret the results in a waveform of the traced signals. A simple AXI interconnect system was created and during simulation of standard tests an AXI input signal to a master or slave was inverted for a clock period to ensure an incorrect value was registered. Deadlock of the system was assumed if the test did not complete by its known time length, then on termination the system could be inspected for other effects of abnormal operation. The analysis shows that the impact of an error on an AXI signal can be severe, possibly Deadlock (DL), Memory Corruption (MC), or Undefined Behaviour (UB), which is a classification for when failure is dependent on the application. Memory corruption could cause failure later in time as it depends on how and if the data is subsequently used. For example, if the data is used to control an actuator in an automobile's brake system. Table 5.2 shows the impact for each signal and two examples are explained below.

5.2.1 LEN

The LEN signal specifies the number of transfers (i.e. data chunks) within a burst. We have to consider the effect of an error changing LEN to a higher value or a lower value. In the former case, it will cause deadlock because the slave expects more transfers than the master will send so the slave blocks which causes the master to block as it is waiting on a write response on the B channel. This prevents any other master from accessing that slave and thus blocks itself, which can cause the whole bus to deadlock. In the latter case, the slave uses the LAST signal to determine how many transfers to consume so it still receives the correct amount.

5.2.2 VALID

The VALID signal is one part of the two-way handshake, which informs the receiver that data is available. An error on VALID can cause deadlock if it transitions from high to low. For example, if the slave has AWREADY high (i.e. the slave is ready to accept information on the address write channel) and the master drives valid data, but an error causes the slave to miss AWVALID high on the following clock cycle then the master thinks there has been a handshake and de-asserts AWVALID so that the slave never realises there was a transfer. The master blocks because it does not get a write completion response.

If an error causes a transition on VALID from low to high and the slave is waiting on a handshake then it will sample garbage from the bus which could corrupt memory or lead to undefined behaviour.

5.3 Commercialising the Fault Tolerance of AXI

Although the chance of a fault is rare, the previous section has highlighted that the consequences may be severe, which can be enough of a concern to particular customers to warrant serious consideration of mitigation. The risk of failure from these types of faults is set to increase because technology scaling increases the likelihood of a fault and as other parts of the chip are protected, such as memory and the CPU, then protecting the bus interconnect will increase, relatively, in importance.

The number of possible mitigation schemes are vast. They can use temporal, information, or spatial redundancy and can be developed at many different layers of abstraction, for example, from circuits to software. Different parts a system can use different techniques and many different techniques can be employed to cover the same component. It is common for techniques that improve dependability to overlap in their coverage as to increase effectiveness. How the solution space is narrowed and the chance that a set of schemes are implemented into a product is dependent on factors internal and external to ARM. It is based upon what ARM provides now and in the future. And it is based on what customers think that they want and what they expect from ARM. Since most of ARM's revenue comes from selling semiconductor IP that are integrated by a customer onto SoC, a software solution or a change to the semiconductor manufacturing process has

Table 5.2: Source: M = Master, S = Slave. Impact: DL = Deadlock, MC = Memory Corruption, UB = Undefined Behaviour. Those signals types with a * suffix are referred to as the control signals.

AXI Signal Type	On Channel	Source	Description	Impact
ID*	All	M, S	The identification tag for the master	DL
ADDR	AW, AR	M	The read or write address	All
LEN*	AW, AR	M	The number of transfers within a burst	DL, MC
SIZE*	AW, AR	M	The width of each transfer in the burst	MC
BURST*	AW, AR	M	Specifies how the address for each transfer is calculated	MC
LOCK*	AW, AR	M	Additional information about the atomic characteristics of the transfer	DL, UB
CACHE*	AW, AR	M	Additional information about the cache-able characteristics of the transfer	UB, MC
PROT*	AW, AR	M	Protection unit information for the transaction	UB, MC
VALID*	All	M, S	Whether information is available from the sender	All
READY*	All	M, S	Indicates that the recipient is ready to accept the information	DL
DATA	W, R	M, S	The actual read or write payload data	MC, UB
STRB*	W	M	Which byte lanes to update in memory	MC
LAST*	W, R	M, S	Indicates the last transfer within a burst	All
RESP	B, R	S	Indicates the status of the write or read transaction	UB

less chance of adoption by ARM because this is not within the bounds of the level of abstraction the company is positioned at and so ARM does not have those capabilities nor do customers expect this from ARM.

The cost-effectiveness of a scheme can be quantified in terms of the decrease in the FIT rate or increase in the Mean Time Between Failure (MTBF) gained against the performance, power, and area overhead and the time and resources needed to implement and verify. This will vary with the level of abstraction at which the scheme is developed, for example multi-bit error correction of memory can be implemented in software rather than hardware because it does not require complex dedicated circuitry. However, performing accelerated irradiation testing or developing a comprehensive modelling program to quantify the improvement in reliability was prohibitively expensive for ARM. Instead, using known fault tolerance techniques like ECC seems to have meant interested customers have not required this qualification.

The following constraints that narrow the solution space were based upon information from engineers with experience of ARM Fabric products, marketers experienced with the target customers, and from requirements from those customers directly. The two target markets outlined previously place different priority on performance and area requirements. Storage networking customers are far more concerned about the adverse impact on clock frequency and throughput than automotive customers, who care more about latency overhead. Additionally, automotive customers are less sensitive to an increase in area than storage networking customers. Although methods were created that avoided extra latency, the cost of a drop in clock frequency and a rise in area, as detailed later, was deemed too high that it would not be acceptable for the storage networking market. Therefore, after careful analysis of the cost of some competing requirements, the following constraints apply:

- We will focus on methods that aim to tolerate transient faults, rather than avoid them because the control of manufacture is outwith ARM's influence (e.g. to lower the incidence of alpha particle strikes) and faults due to neutron strikes cannot be avoided.
- We will only consider online techniques, i.e. techniques that operate on the device while it is "in the field" because of the nature of the fault source: soft errors occur in the natural environment during device use and not only at the time of manufacture.
- We will focus on concurrent online methods, i.e. these methods are active while the device operates normally rather than having to temporarily suspend operation for testing to be performed because: it is expensive, often impractical, and the faults are transient in nature so that their effects may not be observable in an offline state.
- Hardware rather than software schemes are favoured as they present a lower barrier for adoption at ARM and its customers.
- Interfaces for software will be required to manage the hardware schemes.
- We cannot change the AMBA standard. Senior management does not yet consider the problem serious enough compared to the overwhelming commercial and organisational difficulties involved in changing the AMBA

standard (more than fifty companies were part of consultation process for the latest version of the standard).

- We cannot change the master or slave IP that connects to the bus matrix because they can come from any other company and we want to minimise the cost to remain compatible with the AMBA standard. These last two constraints restrict us to changing the bus matrix that implements the AXI protocol.

- We naturally want to minimise the frequency, area, and power overhead of any solution. A common empirical rule in ARM, is that the maximum overhead for each property should be no more than 20%. A side-effect of an increase in area is an increase in the probability of a fault.

- The focus should be on information redundant techniques because they represent a good trade-off between performance and area. The two other ways of increasing reliability through redundancy are spacial and temporal. The former employs physical duplication, from the transistor level up to the bus matrix, and therefore adversely increases area, which has to outweigh the consequent rise in the probability of a fault. Methods generally come under DMR, which duplicates the circuit to detect a fault, and TMR, which uses three instantiations of a circuit and majority voting to detect two faults or correct one. To avoid the respective 100% and 200% increase in area, only selective components can be duplicated based on sensitivity. The interconnect is not amenable to sensitivity analysis because of the wide variability in the manufactured circuity due to many configurable options and dependence on placement. Duplicating the logic between certain masters and slaves is rejected because it presents an open choice to customers to determine what traffic is more critical (they may be unable to make this choice) and a fault in the non-duplicated logic could still cause the whole bus to deadlock. Duplicating the control logic only is not affordable because it makes up most of the logic area. Temporal redundancy tends to infer repetition of work done, such as repeating a computation or sending a transaction across the interconnect again. Generally, these techniques reduce performance, which means it is unattractive for storage networking customers since it affects their need for high throughput and for automotive customers because it affects the requirement for low latency.

These high-level constraints were influenced from ARM sources who have experience and knowledge of the AMBA interconnect and the targeted customers. The solution space can be summarised as online and concurrent hardware techniques to detect or correct errors resulting from transient faults that employ information redundancy and which affects only the implementation of the interconnect. EDAC codes, specifically parity or ECC, are a group of information redundant methods that encode a set of data bits into a number of extra bits, sometimes called check bits, that are sent along with the original data. If the data are corrupted by the transmission medium, then the specific implementation guarantees that a certain number of erroneous bits can be detected or corrected. For example, a Hamming code is one which can detect up to two errors in the covered data and correct single bit errors and so is categorised as a Single Error Correcting-Double Error Detecting (SEC-DED) code. EDAC codes are proposed

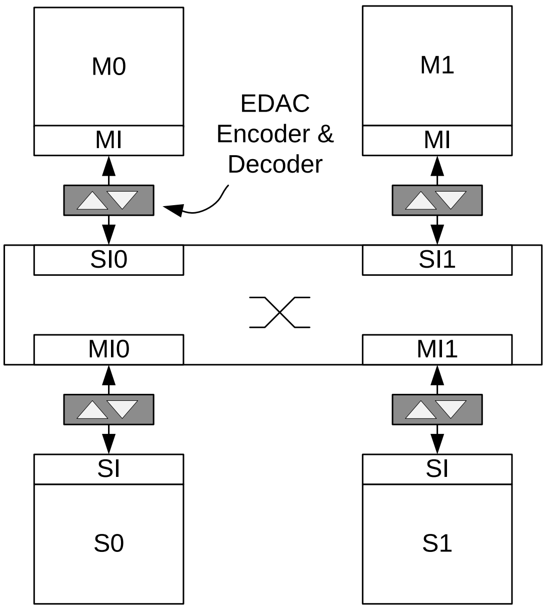


Figure 5.2: Block diagram of the NIC-301 bus matrix with end-to-end EDAC

to detect or correct errors resulting from transient faults because the techniques are familiar and straightforward to understand, which helps to lower the barrier for adoption. Depending on the specific implementation, EDAC codes benefit from low area and performance overheads, compared with spatial or temporal redundancy techniques. As shown in Figure 5.2, the encoding and decoding implementation logic for the EDAC codes are placed at the input and output ports of the interconnect, that is outside the logic of the bus interconnect and before the sender or destination, which is referred to as end-to-end EDAC. This means that no changes to the connecting IP is required, it minimises changes to the bus matrix logic, and simplifies verification and integration because the logic is modular. However, there are some signals for which end-to-end EDAC cannot detect faults.

5.4 Error Detection and Correction Codes are not Sufficient

The following analysis will show how end-to-end EDAC codes cannot be used on some AXI signals and therefore other methods are required. The premise was based on the possibility that a fault may affect the operation of the interconnect in such a way that the code is either not received by the destination to detect the problem or the fault does not manifest itself as an error in the data covered by the code. To find out in which logic a fault may induce this result, the following analysis was performed. For each AXI input signal, a graph was created with

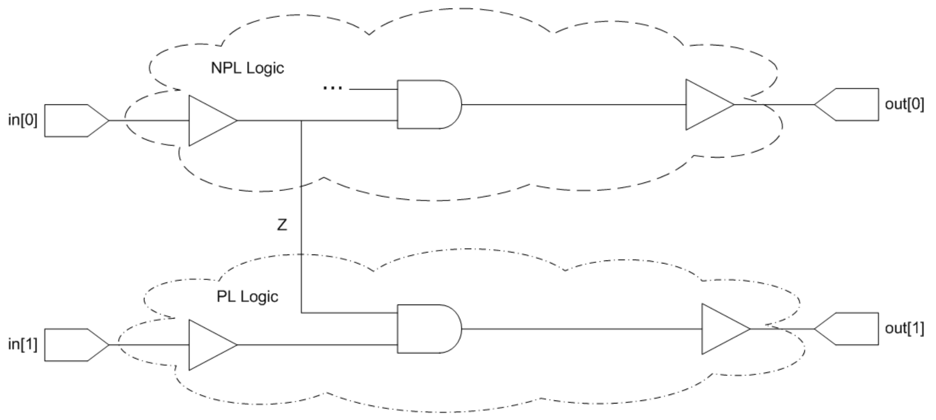


Figure 5.3: Example circuit depicting a PL signal (in[1]) and an NPL signal (in[0])

the input as the root vertex using a depth-first search algorithm. The input signal can be in either direction, i.e. from master to slave or slave to master. The number of output signals reachable in this graph, RO, are counted. If RO is equal to the number of destinations, D, that either the slave or master is connected to[2], then we classify the signal as a Payload (PL) signal. Otherwise, RO must be higher and is classified as a Non-Payload (NPL) signal. PL signals are those that can be encoded using end-to-end EDAC, while NPL signals cannot. A fault on the logic path of a PL signal can become at most a single-bit error at the output of the interconnect because the fault cannot deviate from that logic path and can only propagate to the same output signal. However, a fault on the logic path of an NPL signal, where $RO > D$, can fan-out to other logic not just on a path from its input to its corresponding output. Propagation of a fault on NPL logic may cause more disruption than a single bit error at an output signal and could potentially cause multiple errors at the outputs. For example, a fault on the select signal of a multiplexer can momentarily switch the outputs to the other data inputs, which would then propagate onwards and render a single error correcting ECC ineffective. This distinction between PL and NPL signals is illustrated in Figure 5.3. The input signal `in[1]` has only one reachable output, `out[1]`, and is therefore a PL signal. The logic circuitry on the path of `in[1]` is termed PL logic. On the other hand, the input signal `in[0]` has two reachable outputs and so is a NPL signal. The wire `Z` separates the PL logic from the NPL logic. The NPL logic area is calculated as the total logic area minus the area of the logic on the PL signal paths. The AND-gate at the end of wire Z is part of the PL logic because a fault there only affects `out[1]`. The example highlights that only a fault on the logic upstream from Z can propagate into the PL path.

An example specific to the AXI bus matrix will help to further illustrate this premise. The AXI LOCK signal, which is found on both address channels AW and AR, enables the implementation of atomic access in AXI: either locked access or exclusive access. From the AXI specification [89], an exclusive access

[2]A master or slave may not necessarily be connected to every slave or master, so D can vary.

is defined as:

> The exclusive access mechanism enables the implementation of semaphore type operations without requiring the bus to remain locked to a particular master for the duration of the operation. The advantage of exclusive access is that semaphore type operations do not impact either the critical bus access latency or the maximum achievable bandwidth.

While for a locked access:

> the interconnect must ensure that only that master is allowed access to the slave ... until an unlocked transfer from the same master completes. The arbiter within the interconnect is used to enforce this restriction. ... Locked accesses require that the interconnect prevents any other transactions occurring while the locked sequence is in progress and can therefore have an impact on the interconnect performance.

The slave must implement the functionality required to support the exclusive access, while the interconnect implements the locked access functionality. Therefore, this example will focus on faults that disrupt the operation of a locked access.

The lock logic within the interconnect (the LK block in Figure 5.1) makes this decision and an arbiter is used to enforce this restriction. The system could deadlock or perform undefined behaviour if a fault were to occur that breaks the atomicity of the access. Take the example system in Figure 5.4 of a bus matrix with two masters, M0 and M1, and one slave S0. At time zero, i.e. $t = 0$, M0 issues a write transaction to S0 with no intention of locking the slave to itself, so it sets LOCK = 0. Before the transaction has reached its destination, S0, a fault occurs in any of the logic that causes the interconnect to change its state to locked, i.e. LKST = 1. Because the fault did not occur on the logic path of the LOCK signal, S0 still receives the write transaction with LOCK = 0 at $t = 1$. The EDAC code does not detect the error because it has not occurred on the data it covers. Some time later ($t > 1$), M1 issues a write transaction to S0, but it cannot access it because the fault has caused the interconnect to only allow access to S0 from M0. Since M0 did not intend to lock the bus, it does not realise it has to unlock it, which can then cause M1 to deadlock as it waits for access to S0. The system can recover when M0 makes another write to S0 with LOCK = 0, which resets the lock logic, however it may be unknown when this will occur.

The ability to provide atomic access is part of a contract between masters and the interconnect, the breach of which due to faults does not represent illegal behaviour but a violation of the sender's intention. The lock state, LKST, is a property of the interconnect normally changed via the value of the LOCK signal on the AW or AR channel. The EDAC code covers the PL logic of the LOCK signal from master to slave, but not the NPL logic that controls the LKST property. Therefore other mechanisms are required and are described in the next section.

Iterating the above method for each AXI signal reveals the classification given in Table 5.3. Note that PROT is the only NPL signal not to cause

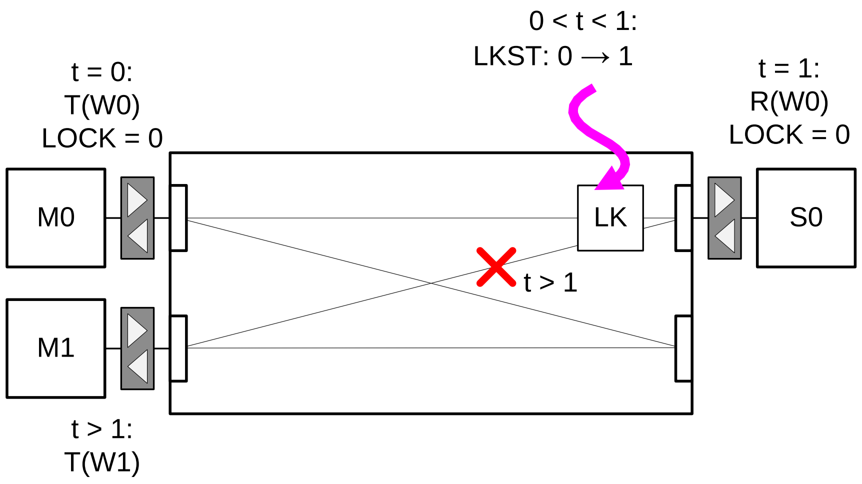


Figure 5.4: An example of a fault that disrupts a locked access on a NIC-301 bus matrix

the interconnect to deadlock because the protection level of the transaction is interpreted by the destination, therefore it is not immediately apparent if deadlock can result since it depends on the application. Also, LEN is the only PL signal that causes deadlock[3], but the difference is that an error on LEN causes the slave to deadlock not the interconnect as in the case with the other NPL signals.

This analysis presents two strands for investigation: firstly, end-to-end EDAC can be employed for PL signals, however, as explained later in Section 5.6, developing a proposal for a product for this is not straightforward and secondly, different methods of fault tolerance are required for NPL signals as described in the following section.

[3]When an error changes LEN to a higher value, as explained in Section 5.2.1 on page 50

Table 5.3: Taxonomy of AXI Signals with Impact. Impact: DL = Deadlock, MC = Memory Corruption, UB = Undefined Behaviour.

(a) AXI NPL Signals

AXI Signal Type	Impact
VALID	DL, MC, UB
READY	DL
ID	DL
ADDR[31:12]	DL, MC, UB
LOCK	DL, UB
LAST	DL, MC, UB
PROT	UB, MC

(b) AXI PL Signals

AXI Signal Type	Impact
ADDR[11:0]	MC, UB
LEN	DL, MC
SIZE	MC
BURST	MC
CACHE	MC, UB
DATA	MC, UB
STRB	MC
RESP	UB

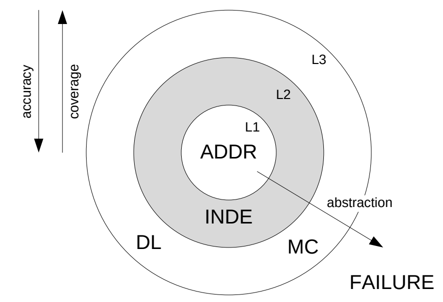


Figure 5.5: A fault-to-failure diagram for the ADDR signal

5.5 Techniques for the Fault Tolerance of Non-Payload Signal Logic

Methods for increasing the fault tolerance can be made at various levels of abstraction within the bounds of the constraints given previously. The concept of a fault-to-failure diagram is introduced to help visualise the manifestation of a fault at different layers of abstraction and thus aid the decision on the most cost-effective mitigation scheme. Using Figure 5.5 as an example, it has at its heart the fault location under consideration (on the logic path of the address, ADDR, signal) at layer one (L1). The level of abstraction at which we view the effect increases per layer until the final result: failure. As we increase the level of abstraction, we trade-off accurate knowledge of the fault source against fault coverage. We can see in layer three (L3) that these effects — DL, MC, and UB — were identified from the protocol-level fault injection analysis. Layer two (L2) effects encompass Incorrect Destination (INDE) and Disallowed behaviour (DISD). INDE means that a transaction is routed to the wrong destination. The DISD effect means that the fault has not broken the protocol, but the interconnect has performed some action that was not intended, such as blocking a secure transaction to a slave because a fault has changed its protection level to unsecure. The L2 effects were discovered from analysing the implementation code (Verilog RTL) and schematics to trace the path of the NPL signals and therefore determine what could happen due to a fault. Repeating this procedure for all the NPL signals, the impact taxonomy can be expanded to include L2 effects as shown in Table 5.4.

A fault can occur on any of the logic on the path of the NPL signal in question; its propagation to failure can be sliced into a finer granularity to create boundaries, or layers, in which to view its effect. The aim is to determine at which layer is the most cost-effective to mitigate its eventual consequence. The protocol-level error injection, described in Section 5.2, revealed three effects which have been assigned to L3. Mitigation at a hardware level is too late when the outcome is MC or UB, as there is not enough information at this level to detect and recover — this would have to be done in software if possible. The usual way to detect DL in hardware is to use a timer, often called a watchdog timer, that is periodically refreshed by a software task before it expires. Thus, it is assumed on its expiration that the software has deadlocked and recovery is normally enacted by resetting the system. This has two disadvantages: firstly, the

Table 5.4: Fault effect taxonomy of AXI signals at different layers of abstraction. L2 effects: INDE = incorrect destination and DISD = disallowed behaviour. L3 effects: DL= deadlock, MC = memory corruption, and UB = undefined behaviour.

AXI Signal Type	Layer 2 Effect	Layer 3 Effect
VALID	INDE, DISD	DL, MC, UB
READY	–	DL
ID	INDE	DL
ADDR[31:12]	INDE	DL, MC, UB
LOCK	DISD	DL, UB
LAST	DISD	DL, MC, UB
PROT	DISD	UB, MC

value of the timer has to be set with accurate knowledge of worst case execution times, which can be difficult to obtain, so the timer is usually set to a very high value which means there is a long duration to detect deadlock; and secondly, the whole system has to be reset because it is not known which components have been affected. Clearly, it would be more beneficial for performance to detect deadlock earlier and only recover those components which were affected. This motivates moving to a lower layer of abstraction. At the lowest layer considered, L1, the fault is viewed as a transient glitch propagating through logic gates. Solutions here would involve duplicating the logic paths or using delayed sampling registers (which sample a logic signal at different times to detect a glitch). In the former method, these NPL signals span most of the logic of the interconnect, which would make duplication too expensive in silicon resource. The latter method would involve changing the technology cell library and would introduce extra latency by adding these registers throughout the logic of the bus matrix[4]. Therefore, a layer inbetween is introduced (L2), the effects here analysed, and solutions proposed. The Double Address Check (DAC) scheme for INDE and the scheme for DISD are built on the same idea of using a monitoring circuit to check that the bus matrix correctly carries out the task which the sender has instructed using the AXI protocol.

5.5.1 Solutions for Incorrect Destination (INDE) Errors

An erroneous value on the address signal can cause a transaction to address a different location within the same slave (when a fault occurs in the bottom part of the address — bits m in Figure 5.6 that specify the size of each memory region as 2^m) or re-route it to a different slave (when a fault occurs in the top part of the address — bits n in Figure 5.6 that specify the number of regions as 2^n), thus causing memory corruption and possibly further severe consequences later. Since the top part of the address is classed as an NPL signal, a fault on its logic path will not appear as an error in the address value that reaches the destination, so end-to-end EDAC will not detect this. Therefore, two different methods are required: one to detect for an incorrect destination and the other

[4]Only using these type of registers at the outputs of the bus matrix would suffer the same problems as end-to-end EDAC, i.e. that the fault could have an effect that is unnoticeable at the output.

to detect or correct an erroneous access to the correct destination. Two schemes are described here for the former effect, while for the latter we can use an EDAC code.

The address decoder is a significant part of the logic chain on the address signal and it determines the destination of a transaction. A fault on this logic could therefore incorrectly re-route it to the wrong place. Instead of duplicating the address decoder, the first mitigation technique proposed modifies its structure to make it an information redundant decoder, called the Hamming Address Decoder (HAD), that takes advantage of a clause in the AXI protocol. The memory map is re-arranged so that an error in the most significant bits, which specify the destination, causes the transaction to be routed to an unmapped region, which according to the AXI protocol will cause an error response to be returned to the master, thus providing detection of a fault that causes an INDE error. The following scheme is used to determine which regions to use. In Figure 5.7 the memory map for a system with two slaves is split into eight regions, so the top three bits of the address, n, index them. The remaining bits, m, are protected with an EDAC code such as parity or ECC. Only those region indexes separated by a Hamming distance of two are mapped, which means that a 1-bit error in any of the top three bits will cause the master to address an unmapped region, which will automatically induce an error response to the sender. This technique can be expanded to use a Hamming distance of three, so that 1-bit errors are corrected, but at the cost of quartering the available memory space. Consequently, this scheme requires a minimal change to the software to alter the header file to use the new memory map and at worst a re-compile, but not any change to the functionality. Thus, 1-bit errors on the address signal can be detected without adding extra latency, but at the cost of halving the addressable memory and a re-compile of the software. A computer program was developed that the user provides with information on the memory map and it outputs the Verilog code that implements the HAD. This code replaces the existing address decoder and the whole bus matrix is simulated to ensure correct operation and synthesised to obtain the area and frequency overheads as shown in Figure 5.8. The overheads for 1-bit parity across the address signal and the HAD-2 scheme (a Hamming distance of two provides 1-bit error detection) are shown. The x-axis gives the size of m, which is the number of bits allocated to region size, while the y-axis gives the maximum synthesis frequency or area overhead at that value of m relative to a bus matrix without either scheme. Using 1-bit of parity to cover the entire 32-bits of the address signal (i.e. $m = 32$) results in half the maximum synthesis frequency and a 10% increase in area. The large frequency overhead arises because the top bits of the address are on the critical path. This frequency penalty prevents the use of parity to detect an INDE error by placing the parity checker after the logic that determines the destination. At $m = 24$ (i.e. $n = 8$, meaning that there are 2^8 regions each of a size of 2^{24}) the HAD-2 scheme incurs less than a 20% drop in f_{max} and an area increase of 15%. Since the rest of the signal has to be protected with parity, f_{max} is limited by that of the parity scheme, which is a approximately 40% drop. Synthesis at $m = 16$ was not possible because the code generated by the computer program was too complex for the synthesis tool (there would be 2^{16} region addresses to check). This complexity is a result of the configuration used: more slaves and using less of the available memory space would reduce the number of regions to check in one Verilog statement. Extrapolating this worst case configuration the optimum

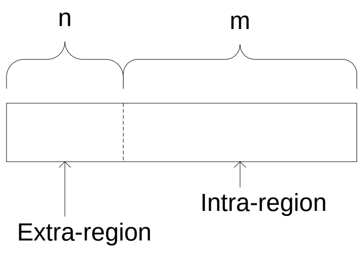


Figure 5.6: The address split into the n bits which specify which region and the m bits which address within a region

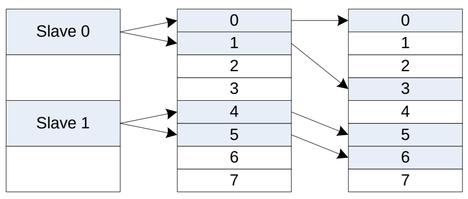


Figure 5.7: The memory map for two slaves split into smaller regions and then remapped to be separated by a Hamming distance of two

point to use both the HAD-2 and parity schemes is at $m = 18$ (where the two f_{max} lines intersect), which gives a frequency drop less than 30%. This linear extrapolation is a reasonable approximation as it is based on the ˜15% drop in f_{max} from the same increase in the number of regions from $m = 32$ to $m = 24$. Using a more realistic configuration or using a Hamming distance of three would reduce the frequency penalty and area overhead further. The HAD scheme was originally developed under the requirement from the automotive market that no extra latency should be added, but since this requires changes to the software, albeit very minimal, it was decided within ARM that this requirement could be relaxed and therefore schemes could be developed that introduced minimal latency overhead.

An alternative which only allows the INDE effect to be detected, but with changes only made to the implementation of the bus matrix, is to perform a second address check at the the receiver — called the DAC. This is placed in the MI block (see Figure 5.9) and compares the top bits of the address to what is expected by the slave, which is set in the DAC when the memory map is specifed at design time. This has the added advantage over the HAD scheme that it can detect an INDE error due to a fault on any logic that can cause this effect, which is more than the address decoder logic. A full address decode is not needed because at design-time the range of addresses for each slave is known, so the circuit which compares the received address with that expected will be much smaller. An error is flagged when these addresses do not match. This will incur extra latency because the transaction will have to be repeated, but only on the occurrence of such a mismatch, which may be acceptable since faults are rare.

5.5.2 Solutions for Disallowed Transaction (DISD) Errors

DISD effects arise from faults that occur on the logic path of the NPL signals LOCK, LAST, and PROT. LAST is included because its outputs affect the LK logic block, therefore a solution for LOCK will also cover faults on the LAST logic path that cause DISD errors. PROT is included because a fault can inadvertently change the secured nature of the transaction or recipient. The solution proposed here for both LOCK and PROT is to compare the corresponding property of the bus matrix, i.e. secure or lock state, with that which was intended via the

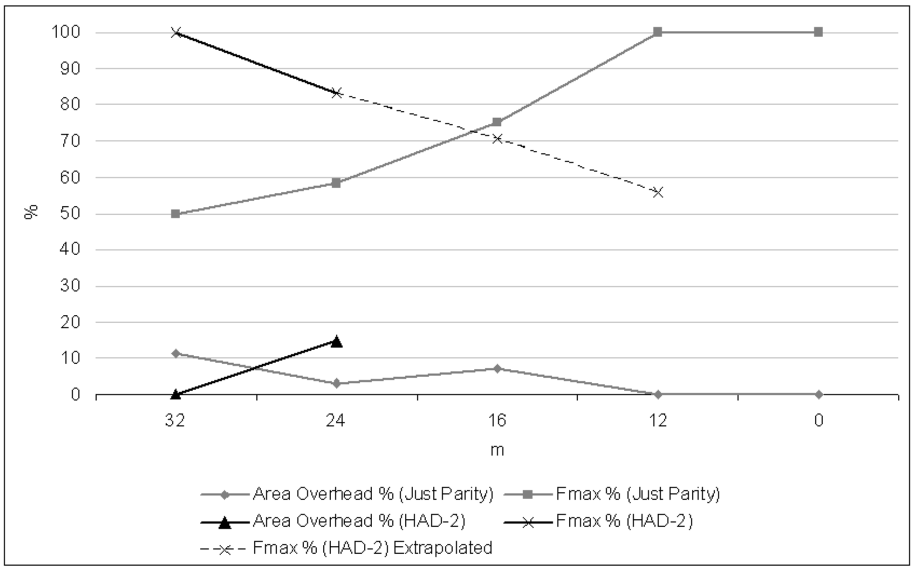


Figure 5.8: Maximum synthesis frequency (f_{max}) and area overhead of the HAD-2 scheme (Hamming distance of 2) compared with parity across the 32-bit ADDR signal. The bus matrix had two masters and two slaves and each slave had half the entire 2^{32} memory space.

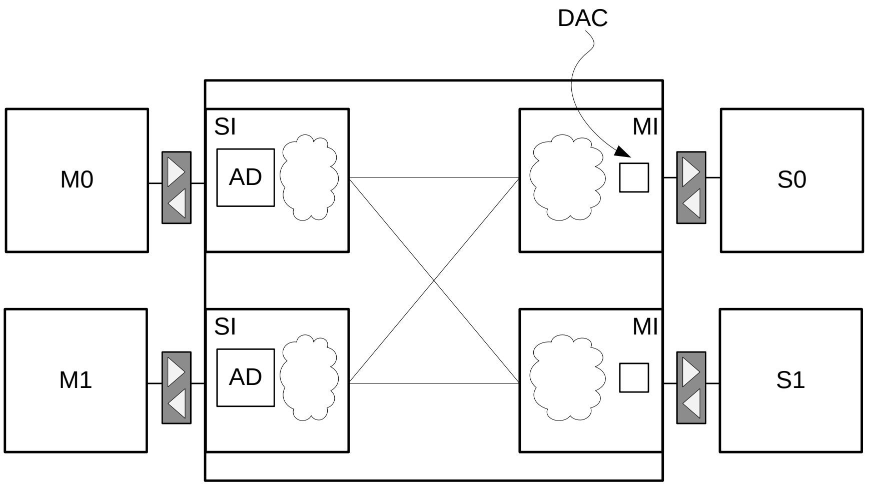


Figure 5.9: Placement of the DAC in scheme to detect INDE errors

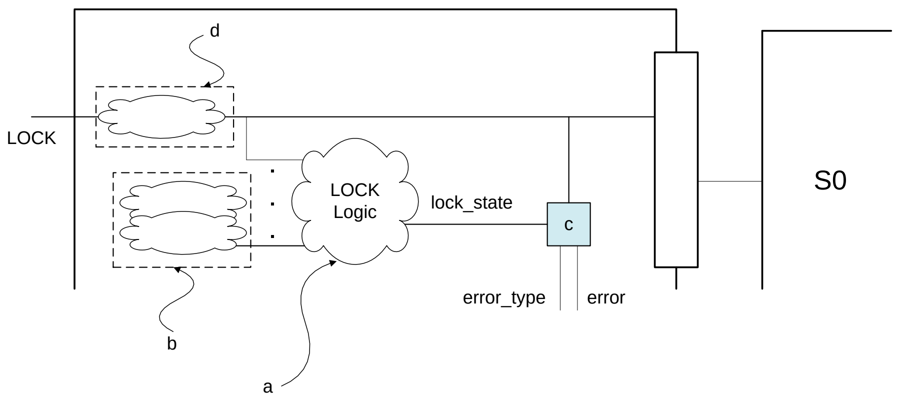


Figure 5.10: A micro-architecture monitor to detect unintended changes in the atomicity of a transaction

protocol. Since the scheme is the same for each signal, it is only described for the LOCK signal. The scheme is an online and concurrent hardware monitor embedded in the bus interconnect and is similar in concept to an assertion checker, but instead of asserting conditions on the protocol this makes assertions about the correct behaviour of the bus matrix.

As shown in Figure 5.10, the scheme uses circuitry *c* that monitors the state of the bus interconnect, in this case the `lock_state`, and compares it against the master's intention (the LOCK signal). This will then detect an unintended change in the lock state due to faults that occur in logic *a* and *b* (the rest of the logic that affects the lock state). The error can then be flagged to some other system component or software, where an appropriate recovery procedure can be initiated. Extra information on what type of error occurred (e.g. an unintended lock error) is provided using the `error_type` signal, which is useful for logging purposes and because the recovery procedure can be tailored to the type of error, e.g. a soft reset of the bus and/or master instead of a full system reset. The scheme is different from known spatial redundancy techniques such as DMR or TMR because no logic has to be duplicated since the checker component *c* compares the signals of different logic that already exist, thus taking advantange of the inherent redundancy in the NIC-301.

This analysis has only covered those faults on the NPL signals that affect its reliability and not its performance, for example, a fault on the LAST signal can also affect the Quality of Service (QoS) and arbiter blocks, which may block a transaction or release a transaction earlier than programmed. It is proposed that these schemes be verified using RTL error injection.

The final step to commercialising the enhancement of the fault tolerance of the AXI bus matrix is combining all the techniques into a proposal for a product that ARM can make a significant return-on-investment (ROI) — the significance is based upon the ROIs of other product proposals that require competing resources. The product takes the form of a separate extension, called an “add-on”, to NIC-301 AXI bus matrix, so that the techniques are only incorporated if the customer requires it. This proposal is described in the following section.

5.6 An Add-On for AXI to Improve its Reliability

To create value for ARM and commercial relevance for the EngD, a significant part of this work has consisted of composing an analysis for the viability of a product that will improve the fault tolerance of the AXI bus matrix. A summary of the recommended features for the proposed product are:

- The product will be an add-on to NIC-301 that will provide SEC-DED ECC protection for most of the AXI signals to correct transient faults that occur on the logic of the bus matrix during communication between masters and slaves.

- In the first revision of the add-on, ECC will cover all signals in which it is feasible, which excludes VALID, READY, LAST, LOCK, and PROT. Mechanisms for these will be introduced in the second revision.

- It will support frequency and data width conversion, but not protocol conversion.

- The product will form a super-set of the functionality of IDM-301[5], to allow information on the transactions where correctable and uncorrectable errors have occurred to be logged and the response action to be programmable – either reported via the AXI error response code Slave Error (SLVERR), an interrupt, accessed through a set of registers, or the destination can be soft reset.

Within ARM, a product proposal requires two key documents: the Product Requirement Specification (PRS) and Engineering Requirement Specification (ERS). Both these documents record the functional and non-functional requirements of the product, but are aimed at different audiences. The PRS is primarily for external presentation — at initial discussions with customers to convey and review the high-level features. The ERS elaborates in more detail, where needed, on the requirements specified in the PRS and is mainly for internal consumption. Combined, they should provide all that is necessary to implement the product. The product discussion document explains the reason why options were chosen over their alternatives, which is important when the product is implemented by other engineers as it will increase confidence in the chosen features so that work is not duplicated.

It is proposed that there are two revisions of the product whereby ECC for the PL signals is implemented in the first and techniques for the NPL are introduced in the second revision. A two-step approach is advocated to lower the barrier to product adoption. Automotive customers currently do not have any functionality to tolerate errors due to faults on the interconnect and the storage networking customers that have this functionality use parity or ECC on the data signals, so the risk of adversely affecting their deadlines is lowered by having known and straightforward features. Lower complexity reduces the risk and cost for ARM to implement it, which raises the return on investment and therefore increases the likelihood that it will be made into a product. The

[5]An existing add-on to the AXI bus matrix, which already provides the ability to log and report bus transaction errors, but does not specify how these errors are generated.

step up to revision two that includes techniques for the NPL signals, which is probably more difficult for customers to accept, is now easier to take than one larger step encompassing all the functionality, where the entire product could be dismissed because of the unfamiliarity and complexity of certain features.

IDM-301 is an existing ARM product that includes the functionality to record an erroneous transaction and to allow the customer to configure how this information is reported. An erroneous transaction is produced by a slave sending the standard AXI error response SLVERR to the master — the reason for the slave generating an error is specific to customer's implementation of the slave. Ramen[6] re-uses IDM-301 to form a super set of its functionality, so that there are two product offerings: IDM-301 on its own or Ramen with IDM-301. Using the same engineering resources will mean less risk of incompatibility.

The PRS and ERS contain the requirements for revision one of Ramen, but some of the details are not entirely straightforward. For example, AXI has the ability to interface between components with different data widths, referred to above as data width conversion, through the use of "upsizers" and "downsizers". This creates a problem of how to choose how many data bits an ECC code covers. Depending on what width conversion is used, this choice may inadvertently add extra latency as the ECC code has to be regenerated at the interface or add more area because more ECC codes are used than is necessary. This issue is covered in product discussion document. We note in detail which combinations of data width, converter type, and number of ECCs per data transfer require additional area or latency, and argue that the best option is to present this information to the user of the product so that they can make informed design decisions based on their own circumstances.

5.7 Conclusion

The impact that transient faults on the AMBA AXI bus matrix can have was examined and it was found that the consequences can be severe, possibly leading to deadlock, memory corruption, or undefined behaviour. The AMBA interconnect is an important ARM product that is present in a large majority of SoCs. It can occupy a significant part of the logic area on a SoC, comparable to an accompanying processor where the focus of improving the fault tolerance of electronic computer hardware has traditionally been. With the susceptibility to soft errors likely to increase with the shrinking of semiconductor technology, the rise in the complexity of the IP interconnect on a SoC, and the severe consequences that result make it a growing concern for some ARM customers, such as automotive and storage networking.

Based on requirements gathered from internal and external sources, a proposal for an add-on product to the AXI bus matrix, NIC-301, was made that increases its fault tolerance using ECC and other techniques. Some ARM customers have already expressed a desire to incorporate some of the techniques into their design to increase the robustness of their interconnect. This has helped the successful transfer of the proposal out of the R&D department and could be available as a product in the next couple of years. The current economic climate for business and the re-prioritisation of resources in ARM towards the recent competition from Intel has meant that the product will likely not be brought to market

[6]The codename of this product.

sooner. However, the work still has potential commercial value for ARM and met one of the chief aims of the EngD to develop research of direct relevance to industry.

Chapter 6

Conclusion

The erroneous perturbation of the logical values in current CMOS based ICs from naturally occurring phenomena, especially those from radioactive particle strikes, called soft errors, are of a concern to semiconductor hardware manufacturers due to the adverse impact on the reliability of their products. This is especially important to their customers who build dependable systems where the cost of failure is high, such as business critical data centres or the safety devices in automotive applications. With technology scaling and the increasing ubiquity of embedded computer devices, soft errors represent another source of failure that some of ARM's customers have to address. This is significant for automotive and networking customers, who have to achieve high levels of safety and availability, respectively.

At sea-level, the effect of soft errors on memory has been dramatically lowered by design changes and by using ECC. Consequently, the contribution of the remaining logic to the overall SER of a chip has grown and is predicted to rise due to the shrinking dimensions of devices and the increasing number of them integrated onto an chip. To evaluate the SER of combinational logic circuitry, one must not only calculate the intrinsic error rate but also the magnitude of the three mechanisms that reduce it. Part of this work looked at the variation of one of these processes, logical masking, with the construction of the circuit and developed a new methodology to carry out this analysis. It was found that the logical masking varied significantly between the different implementations of adder circuits under consideration, from 3.1× at a synthesis frequency of 100 MHz to 2.1× at 900 MHz. The results also reveal that every fault injected causes on average about 1.5 errors and nearly 30% of all faults cause more than one error. This has important ramifications on measuring the SER of combinational logic because the fan-in cone of storage elements has to be taken into account. In conjunction with modelling the other two masking mechanisms, this methodology can then be used to calculate the sensitivity of a circuit to soft errors. This methodology will be of particular importance during the design phase, along with other performance metrics, to evaluate different circuits and guide the development of mitigation techniques. Properties of mathematical graph theory were used to correlate the composition of the circuit with logical masking, which could be used in the development of a static analysis rather than the time consuming dynamic simulation.

It was envisaged that the SER of current or future ARM IP could be measured, however, from knowledge gathered during this investigation, developing the comprehensive methodology needed did not presently constitute a high enough return on investment for ARM. Although it would be costly to perform accelerated testing of hardware, which ARM does not produce, and construct a modelling framework, there was not a strong enough immediate financial incentive from customers to provide this data. However, customers will need such a tool in the future according to the trends identified in this work, and the first steps in providing such a facility have been made, and published, for when it becomes financially viable.

The second part of this work continued to investigate soft errors in logic, but with more immediate commercial value, to show that faults on the ARM AMBA AXI bus interconnect can have a serious impact on the reliability of an entire SoC. Protocol-level error injection was performed on the AXI bus, which found that errors could cause deadlock, memory corruption, and undefined behaviour. Moreover, the AMBA bus standard features in a vast array of embedded devices and can occupy an area comparable to typical ARM processors that it is connected to. With an increase in the complexity and number of components in an IC, the interconnect will become a more complex and important part of the SoC. Interest from automotive and storage networking customers helped solidify the motivation to develop a product proposal for an add-on to the AXI bus interconnect that incorporates ECC to detect, correct, and report transient faults in the bus logic without changing the AMBA standard or the connecting customer IP. Within industry, a solution that uses proven techniques and limits the subsequent changes a customer has to make to their system lowers the cost and risk for adoption. However, it is argued that there are faults that cannot be detected by ECC and yet can have severe consequences; thus several novel techniques are proposed that are needed in addition to ECC. The appropriate requirement specification documents were created with approval from the internal stakeholders, which signifies the successful transition of a research idea out of ARM R&D. With a robust foundation established by this work, the product can be brought to market within the next two years.

6.1 Future Work

In the next twelve months, ARM customers need to understand how the functionality of the proposed product will be implemented and verified. Critical to the verification is to quantify the improvement in the error rate that the proposed fault tolerant techniques bring. This would be done using the SFI methodology developed, but would need some performance improvements to make simulation computationally tractable for a bus interconnect netlist. The first step would be to ensure the number of faults injected is the minimum needed for the confidence level, which could be lowered too. The second step would extract sub-circuits or sub-graphs, using the same graph traversal algorithm for identifying fault paths, which are smaller and can be processed in parallel. Indeed this approach can be applied to larger circuits, such as ARM processors, by extracting combinatorial circuits between registers and simulating them individually before combining the error rates. If the computation time is still too large, then a static approach may be required, such as those identified in Section 4.4 on page 40 potentially incorporating the topology properties of the circuit for calculating logical masking. For both dynamic and static approaches the methodology would have to be augmented with models for electrical and latch-window masking. There needs to be more investigation into the accuracy of the two graph metrics identified and whether other metrics need to be incorporated, which would involve performing experiments with other circuits.

As shown in this work the variation of one of the factors in calculating the SER can be significant, therefore, it is crucial to examine the sensitivity of the SER to variation of other terms. Industry requires knowledge of which factors have most influence to allow them to work out the most cost-effective way to reduce the error rate to the desired level.

Over the next five years, it will be important to research energy-efficient dependability, the reliability of wires with technology scaling, and the interaction of manufacturing process variation with reliability.

ARM is particularly known for designing IP which are used in systems with small power budgets, relative to desktop or server computers. The combination of the concepts of low power and reliability reveals two avenues of interest: how to design and test low power fault tolerance techniques and how does the use of methods to reduce power affect the SER? When companies such as Intel test the dependability of their high-end servers there is an expectation that these computers will be in continuous operation for their life-time, but the systems in which ARM IP is present will likely employ various levels of sleep modes to reduce active power dissipation. Given the same length of life-time, one might expect fewer failures in the latter case because the system is not performing as much useful work. However, how does the use of clock gating, power gating, or varying the voltage to memory cells impact the SER? The contribution of the clock tree was found to be about 10% to 20% of the total SER of non-hardened latches and dominates the hardened latch SER [90][1]. How often do memory elements have to be off to mitigate the increase in susceptibility of the extra logic that clock gating adds? One would expect that using power gating to turn off elements would reduce the error rate. However, if a flip-flop or memory cell is put into a retention or sleep state by lowering the voltage, then this could make the elements much more susceptible to soft errors, since voltage scaling is seen as the primary concern for the increase in soft errors as technology shrinks [41]. Additionally, the use of different voltage domains may mean that some memory is more susceptible than others. The effect of techniques to lower energy consumption on dependability could become more important in the future when computer designers, which includes those designing servers, look to reduce the energy usage for cost and environmental reasons.

With the bus interconnect increasing in size and complexity along with the systems it inhabits, the length and the number of metal wires will rise. It will be important for ARM to consider the particular faults that affect these metal wires, such as electromigration, cross-talk, substrate coupling, etc.

An area that requires further investigation is the link between manufacturing process variation and reliability. Chip-to-chip process variation from IBM manufacturing introduced a spread in the SER of about 2× while measurements of chips from other companies showed this to be greater than 200× [91]. More recent data found that the spread in the SER of SRAM increased from 3× to

[1]For comparison, the same author reports four years later that the total contribution of combinational logic in a microprocessor chip is no more than 30% of the total unhardened latch SER[29].

4.3× between 180 nm to 130 nm, which indicates it is likely that the SER spread will widen in the future because controlling variation is ever more challenging and expensive [10]. This source of uncertainty in the SER alone is enough to warrant mechanisms to reduce at least the spread. Since reducing process variation increases cost, would it be cheaper at some point to allow greater variation, but to deal with the consequences with fault tolerant mechanisms? Does this change the character of the fault sources such that different mechanisms are required? A second aspect to this link is: do effects such as device ageing change the SER of the system over time? If it does, then the SER will be a dynamic property of the system rather than the static property as it is currently viewed. This would introduce a further source of uncertainty that would require examination. Whilst it may be that ageing effects have a negligible impact on the SER of previous technology (such as IBM's 65 nm SOI process [92]), indications of increasing variation at future technology nodes [93], especially in threshold voltage, would suggest further research is needed.

Bibliography

[1] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, Jan. 2004.

[2] R. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 305–316, 2005.

[3] D. Lyons. (2000) Sun Screen. Last accessed: 26/02/10. [Online]. Available: http://www.forbes.com/forbes/2000/1113/6613068a.html

[4] L. Spainhower and T. A. Gregg, "IBM S/390 Parallel Enterprise Server G5 fault tolerance: A historical perspective," *IBM Journal of Research and Development*, vol. 43, no. 5.6, pp. 863–873, 1999.

[5] ICKnowledge. (2002) Defect Density Trends. Last accessed: 22/03/10. [Online]. Available: http://www.icknowledge.com/trends/defects.pdf

[6] C. Constantinescu, "Impact of deep submicron technology on dependability of VLSI circuits," in *Proceedings International Conference on Dependable Systems and Networks*, 2002, pp. 205–209.

[7] D. Siewiorek, R. Chillarege, and Z. Kalbarczyk, "Reflections on industry trends and experimental research in dependability," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 2, pp. 109–127, 2004.

[8] C. Constantinescu, "Neutron SER Characterization of Microprocessors," in *International Conference on Dependable Systems and Networks.* IEEE, 2005, pp. 754–759.

[9] J. F. Ziegler, H. Curtis, H. Muhlfeld, C. Montrose, B. Chin, M. Nicewicz, C. Russell, W. Wang, L. Freeman, P. Hosier, and Others, "IBM experiments in soft fails in computer electronics (1978-1994)," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 3–18, 1996.

[10] ITRS, "International Technology Roadmap for Semiconductors 2009."

[11] B. Stackhouse, S. Bhimji, C. Bostak, D. Bradley, B. Cherkauer, J. Desai, E. Francom, M. Gowan, P. Gronowski, D. Krueger, C. Morganti, and S. Troyer, "A 65 nm 2-Billion Transistor Quad-Core Itanium Processor," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 18–31, 2009.

[12] T. C. May and M. H. Woods, “Alpha-particle-induced soft errors in dynamic memories,” *IEEE Transactions on Electron Devices*, vol. 26, no. 1, pp. 2–9, 1979.

[13] R. Silberberg, C. H. Tsao, and J. R. Letaw, “Neutron Generated Single-Event Upsets in the Atmosphere,” *IEEE Transactions on Nuclear Science*, vol. 31, no. 6, pp. 1183–1185, 1984.

[14] A. Taber and E. Normand, “Single event upset in avionics,” *IEEE Transactions on Nuclear Science*, vol. 40, no. 2, pp. 120–126, 1993.

[15] J. F. Ziegler and W. A. Lanford, “Effect of Cosmic Rays on Computer Memories,” vol. 206, no. 4420, 1979, pp. 776–788.

[16] E. Normand, “Single event upset at ground level,” *IEEE Transactions on Nuclear Science*, vol. 43, no. 6, pp. 2742–2750, 1996.

[17] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, “Modeling the effect of technology trends on the soft error rate of combinational logic,” in *International Conference on Dependable Systems and Networks.* IEEE Comput. Soc, 2002, pp. 389–398.

[18] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, B. Gill, and J. Maiz, “Radiation-induced soft error rates of advanced CMOS bulk devices,” in *IEEE International Reliability Physics Symposium Proceedings*, vol. 44, 2006, pp. 217–225.

[19] R. Baumann, T. Hossain, E. Smith, S. Murata, and H. Kitagawa, “Boron as a primary source of radiation in high density DRAMs,” in *IEEE Symposium on VLSI Technology*, 1995, pp. 81–82.

[20] “JEDEC JESD89 Standard - Measurement and Reporting of Alpha Particles and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices,” 2001.

[21] H. Nguyen, Y. Yagil, N. Seifert, and M. Reitsma, “Chip-level soft error estimation method,” in *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3. IEEE, Sep. 2005, pp. 365–381.

[22] S. Mukherjee, *Architectural Design for Soft Errors.* Morgan Kaufmann, 2008.

[23] M. Gordon, P. Goldhagen, K. Rodbell, T. Zabel, H. Tang, J. Clem, and P. Bailey, “Measurement of the Flux and Energy Spectrum of Cosmic-Ray Induced Neutrons on the Ground,” *IEEE Transactions on Nuclear Science*, vol. 51, no. 6, pp. 3427–3434, 2004.

[24] J. F. Ziegler, “Terrestrial cosmic ray intensities,” *IBM Journal of Research and Development*, vol. 42, no. 1, pp. 117–139, 1998.

[25] G. R. Srinivasan, “Modeling the cosmic-ray-induced soft-error rate in integrated circuits: An overview,” *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 77–89, 1996.

[26] P. Bradley and E. Normand, “Single event upsets in implantable cardioverter defibrillators,” *IEEE Transactions on Nuclear Science*, vol. 45, no. 6, pp. 2929–2940, 1998.

[27] J. Ziegler and H. Puchner, *SER – History, Trends and Challenges.* Cypress Semiconductor, 2004.

[28] P. Dodd, M. Shaneyfelt, J. Schwank, and G. Hash, “Neutron-induced soft errors, latchup, and comparison of SER test methods for SRAM technologies,” in *International Electron Devices Meeting.* IEEE, 2002, pp. 333–336.

[29] B. Gill, N. Seifert, and V. Zia, “Comparison of alpha-particle and neutron-induced combinational and sequential logic error rates at the 32nm technology node,” in *International Reliability Physics Symposium.* IEEE, 2009, pp. 199–205.

[30] ITRS, “International Technology Roadmap for Semiconductors 2005.”

[31] Intel. (2010) Intel Pentium 4 Processor 670. Last accessed: 29/09/10. [Online]. Available: http://ark.intel.com/Product.aspx?id=27487&processor=670&spec-codes=SL7Z3,SL8PY,SL8Q9

[32] C. Constantinescu, “Dependability analysis of a fault-tolerant processor,” in *Proceedings of Pacific Rim International Symposium on Dependable Computing*, 2001, pp. 63–67.

[33] C. Tuner. (2010) Cortex-R4 Whitepaper. Last accessed: 28/09/10. [Online]. Available: http://www.arm.com/files/pdf/Cortex-R4-white-paper.pdf

[34] S. Mitra, N. Seifert, M. Zhang, Q. Shi, and K. Kim, “Robust system design with built-in soft-error resilience,” *Computer*, vol. 38, no. 2, pp. 43–52, Feb. 2005.

[35] R. Baumann, “The impact of technology scaling on soft error rate performance and limits to the efficacy of error correction,” *International Electron Devices Meeting*, pp. 329–332, 2002.

[36] S. Hareland, J. Maiz, M. Alavi, K. Mistry, and S. Walsta, “Impact of CMOS process scaling and SOI on the soft error rates of logic processes,” in *Symposium on VLSI Technology.* Japan Soc. Appl. Phys, 2001, pp. 73–74.

[37] P. Hazucha, T. Karnik, J. Maiz, S. Walstra, B. Bloechel, J. Tschanz, G. Dermer, S. Hareland, P. Armstrong, and S. Borkar, “Neutron soft error rate measurements in a 90-nm CMOS process and scaling trends in SRAM from 0.25-m to 90-nm generation,” in *IEEE International Electron Devices Meeting.* IEEE, 2003, pp. 21.5.1–21.5.4.

[38] P. Hazucha and C. Svensson, “Impact of CMOS technology scaling on the atmospheric neutron soft error rate,” *IEEE Transactions on Nuclear Science*, vol. 47, no. 6, pp. 2586–2594, 2000.

[39] N. Seifert, B. Gill, K. Foley, and P. Relangi, "Multi-cell upset probabilities of 45nm high-k + metal gate SRAM devices in terrestrial and space environments," in *International Reliability Physics Symposium.* IEEE, 2008, pp. 181–186.

[40] R. Baumann, "Soft Errors in Advanced Computer Systems," *IEEE Design and Test of Computers*, vol. 22, no. 3, pp. 258–266, 2005.

[41] N. Cohen, T. Sriram, N. Leland, D. Moyer, S. Butler, and R. Flatley, "Soft error considerations for deep-submicron CMOS circuit applications," in *International Electron Devices Meeting.* IEEE, 1999, pp. 315–318.

[42] T. Karnik, B. Bloechel, K. Soumyanath, V. De, and S. Borkar, "Scaling trends of cosmic ray induced soft errors in static latches beyond 0.18 μ," *Symposium on VLSI Circuits*, pp. 61–62, 2001.

[43] ITRS, "International Technology Roadmap for Semiconductors 2001."

[44] J. F. Ziegler, "Terrestrial cosmic rays," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 19–40, 1996.

[45] J. Ziegler, "Review of Accelerated Testing of SRAMs," *Annual Topical Research Conference On Reliability*, 2000.

[46] G. Srinivasan, P. Murley, and H. Tang, "Accurate, predictive modeling of soft error rate due to cosmic rays and chip alpha radiation," in *International Reliability Physics Symposium.* IEEE, 1994, pp. 12–16.

[47] L. B. Freeman, "Critical charge calculations for a bipolar SRAM array," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 119–129, 1996.

[48] P. C. Murley and G. R. Srinivasan, "Soft-error Monte Carlo modeling program, SEMM," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 109–118, 1996.

[49] ARM. (2010) Company Profile. Last accessed: 09/11/10. [Online]. Available: http://www.arm.com/about/company-profile/index.php

[50] ARM. (2009) Annual Report. Last accessed: 09/11/10. [Online]. Available: http://www.arm.com/annualreport09

[51] Intel. Whitepaper: Introduction to Intel's 32nm Process Technology. [Online]. Available: http://download.intel.com/pressroom/kits/32nm/westmere/Intel_32nm_Overview.pdf

[52] (2009) TSMC sees Q3 demand as it solidifies roadmap. Last accessed: 21/10/09. [Online]. Available: http://www.eetimes.com/news/semi/showArticle.jhtml?articleID=219501118

[53] ITRS, "International Technology Roadmap for Semiconductors 2000."

[54] Bosch. Product Information Airbag Chip Set 4th Generation CG101, CG102, CG103. Last accessed: 21/10/09. [Online]. Available: http://www.semiconductors.bosch.de/pdf/CG10x_Product_Info_2007_06.pdf

[55] STMicro. ST Microelectronics 32-bit Microcontrollers for Automotive. Last accessed: 21/10/09. [Online]. Available: http://www.st.com/stonline/products/families/automotive/microcontrollers/32bit_micro.htm

[56] The Automotive Electronics Council (AEC). [Online]. Available: http://www.aecouncil.com

[57] (2009) Electronic stability control to be mandatory from 2011. Last accessed: 21/10/09. [Online]. Available: http://www.profeng.com/archive/archive+2009/2206/22060027.htm

[58] AUTOSAR homepage. [Online]. Available: http://www.autosar.org

[59] Gartner, "Market Share Analysis: Preliminary Total Semiconductor Revenue, Worldwide, 2009," 2009.

[60] Semicast Research Ltd, "Automotive Semiconductor Market Forecast to Recoverin 2010," 2010.

[61] (Gartner), "Competitive Landscape: Automotive Semiconductors," 2009.

[62] Gartner, "Forecast Analysis: Automotive Semiconductors, 2Q09," 2009.

[63] BBC. (2010) Q&A: Toyota recalls. Last accessed: 06/11/10. [Online]. Available: http://news.bbc.co.uk/1/hi/business/8496902.stm

[64] Businessweek. (2010) Toyota Recall Cost to Exceed $2 Billion, Lawyers Say (Update2). Last accessed: 06/11/10. [Online]. Available: http://www.businessweek.com/news/2010-02-09/toyota-recall-cost-to-exceed-2-billion-lawyers-say-update2-.html

[65] Toyota. (2010) Toyota Finanical Results. Last accessed: 06/11/10. [Online]. Available: http://www.toyota.co.jp/en/ir/financial_results/2010/year_end/summary.pdf

[66] ADAC, "ADAC Pannenstatistik 2008," 2008.

[67] (Gartner), "Forecast Analysis: Hard-Disk Drives, Worldwide, 2004- 2013, December 2009 Update," 2010.

[68] Gartner, "Dataquest Insight: Global Hard-Disk Drive Forecast Scenarios, 2009-2013, 3Q09 Update," 2009.

[69] (Gartner), "Market Share: Enterprise Ethernet Switches, Worldwide, 2Q09," 2009.

[70] Infonetics. (2010) SAN switch and adapter market forecast to hit $6.5 billion in 2014. Last accessed: 06/11/10. [Online]. Available: http://www.infonetics.com/pr/2010/1Q10-Storage-Network-Equipment-Market-Highlights.asp

[71] N. Miskov-Zivanov and D. Marculescu, "Modeling and Analysis of SER in Combinational Circuits," in *SELSE*, no. 1, 2010.

[72] M. Zhang and N. Shanbhag, "A soft error rate analysis (SERA) methodology," in *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design.* IEEE Computer Society, 2004, p. 118.

[73] P. Hazucha, C. Svensson, and S. Wender, "Cosmic-ray soft error rate characterization of a standard 0.6-& CMOS process," *Solid-State Circuits, IEEE Journal of*, vol. 35, no. 10, pp. 1422–1429, 2000.

[74] Y. Tosaka, H. Kanata, S. Satoh, and T. Itakura, "Simple method for estimating neutron-induced soft error rates based on modified BGR model," *Electron Device Letters, IEEE*, vol. 20, no. 2, pp. 89–91, 1999.

[75] P. Liden, P. Dahlgren, R. Johansson, and J. Karlsson, "On latching probability of particle induced transients in combinational networks," in *Fault-Tolerant Computing Symposium*, 1994, pp. 340–349.

[76] M. Baze and S. Buchner, "Attenuation of single event induced pulses in CMOS combinational logic," *IEEE Transactions on Nuclear Science*, vol. 44, no. 6, pp. 2217–2223, 1997.

[77] S. Buchner, M. Baze, D. Brown, D. McMorrow, and J. Melinger, "Comparison of error rates in combinational and sequential logic," *IEEE Transactions on Nuclear Science*, vol. 44, no. 6, pp. 2209–2216, 1997.

[78] M. Gadlage, P. Eaton, J. Benedetto, and T. Turflinger, "Comparison of heavy ion and proton induced combinatorial and sequential logic error rates in a deep submicron process," *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2120–2124, Dec. 2005.

[79] R. R. Rao, K. Chopra, D. Blaauw, and D. Sylvester, "An efficient static algorithm for computing the soft error rates of combinational circuits," in *Design, Automation, and Test in Europe*, 2006, pp. 164–169.

[80] S. Yang, "Logic synthesis and optimization benchmarks user guide," 1991.

[81] F. Brglez and H. Fujiwara, "A neural netlist of ten combinational benchmark circuits and translator in Fortran," in *ISCAS*, 1985.

[82] N. Miskov-Zivanov and D. Marculescu, "Circuit Reliability Analysis Using Symbolic Techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 12, pp. 2638–2649, 2006.

[83] Miskov-Zivanov, N. and Marculescu, D., "Modeling and Optimization for Soft-Error Reliability of Sequential Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 803–816, 2008.

[84] ——, "A systematic approach to modeling and analysis of transient faults in logic circuits," in *2009 10th International Symposium on Quality of Electronic Design*. Ieee, Mar. 2009, pp. 408–413.

[85] L. Massengill, A. Baranski, D. Van Nort, J. Meng, and B. Bhuva, "Analysis of single-event effects in combinational logic-simulation of the AM2901 bitslice processor," *Nuclear Science, IEEE Transactions on*, vol. 47, no. 6, pp. 2609–2615, 2000.

[86] A. Nieuwland, S. Jasarevic, and G. Jerin, "Combinational Logic Soft Error Analysis and Protection," in *IEEE International On-Line Testing Symposium*. IEEE, 2006, pp. 99–104.

[87] R. Ramanarayanan, V. S. Degalahal, R. Krishnan, J. Kim, V. Narayanan, Y. Xie, M. J. Irwin, and K. Unlu, "Modeling Soft Errors at the Device and Logic Levels for Combinational Circuits," *IEEE Transactions on Dependable and Secure Computing*, vol. 6, no. 3, pp. 202–216, 2009.

[88] J. S. Kim, C. Nicopoulos, N. Vijaykrishnan, Y. Xie, and E. Lattanzi, "A Probabilistic Model for Soft-Error Rate Estimation in Combinational Logic," *1st International Workshop on Probabilistic Analysis Techniques For Real Time And Embedded Systems (PARTES)*, 2004.

[89] "ARM AMBA 3 AXI Specification (available at http://www.arm.com/products/solutions/axi_spec.html)."

[90] N. Seifert, P. Shipley, M. Pant, V. Ambrose, and B. GiIl, "Radiation-induced clock jitter and race," in *IEEE International Reliability Physics Symposium*. IEEE, 2005, pp. 215–222.

[91] J. F. Ziegler, H. P. Muhlfeld, C. Montrose, H. Curtis, T. O'Gorman, and J. Ross, "Accelerated testing for cosmic soft-error rate," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 51–72, 1996.

[92] E. H. Cannon, A. J. Kleinosowski, R. Kanj, D. D. Reinhardt, and R. V. Joshi, "The Impact of Aging Effects and Manufacturing Variation on SRAM Soft-Error Rate," *IEEE Transactions on Device and Materials Reliability*, vol. 8, no. 1, pp. 145–152, 2008.

[93] M. Bukhori, A. Brown, S. Roy, and A. Asenov, "Simulation of statistical aspects of reliability in nano CMOS transistors," in *International Integrated Reliability Workshop*, 2009, pp. 82–85.

Appendix A

Acronyms

AA	The Automobile Association
ABS	Anti-lock Braking System
ADD	Algebraic Decision Diagram
ADAC	Allgemeiner Deutscher Automobil-Club e.V.
AEC	Automotive Electronics Council
AHB	Advanced High-performance Bus
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
ASSP	Application Specific Standard Product
ATB	Advanced Trace Bus
AUTOSAR	Automotive Open System Architecture
AXI	Advanced eXtensible Interface
BCD	Bipolar CMOS DMOS
BDD	Binary Decision Diagram
BGR	Burst Generation Rate
BPSG	Borophosphosilicate Glass
CMOS	Complementary Metal-Oxide-Semiconductor
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DAC	Double Address Check
DEC	Double Error Correction

DISD	Disallowed behaviour
DL	Deadlock
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
DMOS	Doublediffused MetalOxideSemiconductor
DMR	Dual Modular Redundancy
DRAM	Dynamic Random Access Memory
DUE	Detected Uncorrectable Error
ECC	Error Correcting Code
ECU	Electronic Control Unit
EDAC	Error Correction or Detection Code
EM	Electrical Masking
ERS	Engineering Requirement Specification
ESC	Electronic Stability Control
EU	European Union
FIT	Failures In Time (number of failures per billion hours in which a device is in operation)
GeV	Giga electron Volt
GUI	Graphical User Interface
HAD	Hamming Address Decoder
Hκ-MG	High-κ-Metal Gate
IC	Integrated Chip
IEC	International Electrotechnical Commission
INDE	Incorrect Destination
IP	Intellectual Property
IR	Incorrect Result
ISO	International Organization for Standardization
ITRS	International Technology Roadmap for Semiconductors
JEDEC	JEDEC Solid State Technology Association
LET	Linear Energy Transfer
LM	Logical Masking

LSI	Large Scale Integration
LWM	Latch-Window Masking
Mbit	Mega-bit (one million bits)
MBU	Multi-Bit Upset
MC	Memory Corruption
MCU	Multi-Cell Upset
MET	Multiple Event Transient
MeV	Mega electron Volt
MI	Master Interface
MPU	Microprocessor Unit
MTBF	Mean Time Between Failure
NCS	Neutron Cross Section
NPL	Non-Payload
OCP	Open Core Protocol
PL	Payload
PRS	Product Requirement Specification
QoS	Quality of Service
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
R&D	Research and Development
RISC	Reduced Instruction-Set Computer
ROI	Return On Investment
RTL	Register Transfer Level
SAN	Storage Area Network
SBU	Single-Bit Upset
SDC	Silent Data Corruption
SEC-DED	Single Error Correcting-Double Error Detecting
SEE	Single Event Effect
SEL	Single Event Latch-up
SELSE	Workshop on Silicon Errors in Logic - System Effect

SEMM	Soft Error Monte carlo Modeling
SER	Soft Error Rate
SET	Single Event Transient
SEU	Single Event Upset
SFI	Statistical Fault Injection
SI	Slave Interface
SIA	Semiconductor Industry Association
SIL	Safety Integrity Level
SLVERR	Slave Error
SoC	System-on-a-Chip
SOI	Silicon on Insulator
SPICE	Simulation Program with Integrated Circuit Emphasis
SRAM	Static Random Access Memory
TCM	Tightly Coupled Memory
TMR	Triple Modular Redundancy
UB	Undefined Behaviour
USB	Universal Serial Bus
VPI	Verilog Procedural Interface

Appendix B

SELSE Paper

Design For Reliability: An Analysis of Logical Masking on Transient Faults

Derek Graham*, Daryl Bradley†, Scott Roy‡, Per Strid†, Fernando Rodriguez‡
*Institute for System Level Integration
(Universities of Edinburgh, Glasgow, Heriot-Watt and Strathclyde)
†ARM Ltd
‡University of Glasgow

***Abstract*—The propagation of transient faults within a building block of a microprocessor was studied to ascertain the influence its design may have on its reliability. Statistical fault injection was performed on various implementations of adders to study the effect that the micro-architecture has on the logical masking of the circuit. This provides an upper bound on the likelihood of a fault appearing at the output of a component because the two other masking phenomena within combinational logic - electrical masking and latch-window masking - are not considered. The soft error rate, which includes logical masking, used in conjunction with other metrics, such as power, performance, and area, would help a designer to weigh the benefit between choosing a different implementation based on the error rate or applying a mitigation technique to the existing implementation. We find that the largest difference in error rate between the 32-bit adder implementations is 3.1X and that the error rate decreases with higher synthesis frequency. We then relate this to measures of circuit composition, such as fan-out and mean path length.**

I. Introduction

Compared to physical experimentation, performing statistical fault injection by simulation to analyse the impact of errors on a system benefits from higher controllability, observability, cost efficiency, and increases confidence by enabling testing before implementation. This work uses this method to model the effect of transient faults, which can occur due to a number of reasons, including power supply and interconnect noise, electromagnetic interference, and particle strikes due to radiation from the packaging materials and the atmosphere [1], [2]. The impact of single-event upsets (SEUs) on system reliability due to particle strikes has been predicted to increase with shrinking process feature size [3]–[5] and those originating from non-memory logic areas (i.e. combinational and sequential logic) are envisaged to become more important than those from memory [6], [7].

The adder is a crucial unit of a microprocessor's pipeline, that is utilised for many operations other than explicit add instructions, such as in the computation of the address for the next instruction. Therefore, if the adder were to experience a fault and cause an incorrect result, the severity would be great.

Various implementations of an adder at widths of 8 to 64 bits were used to explore the effect that the composition of the circuit has on logical masking.

This paper is organised as follows: the next section describes the method in which logical masking is measured, then the results are presented for various adder implementations at different widths in section III, followed by a discussion on the relation between the circuit topology and logical masking, and then finally the conclusion.

II. Method

The soft error rate (SER) of a circuit due to particle strikes can be calculated as [8]:

$$SER = R_{PH} \cdot \alpha \cdot P(SE)$$

Where R_{PH} is the particle hit rate, α is the fraction of hits that result in a single-event transient (SET) and $P(SE)$ is the probability of that SET becoming a SEU or soft error. Neutron cross section [9], [10] and burst generation rate [11], [12] are two commonly used methods to calculate the effective particle hit rate ($R_{PH} \cdot \alpha$).

In this report we assume that every particle hit is effective, which is modelled as a fault injection. In combinational logic, $P(SE)$ encapsulates three phenomena that act to mask the SET before realising a SEU: logical masking, electrical masking and latch-window masking. We consider the factors that effect the logical masking properties of a circuit. The width of the SET pulse is set to the same size as the clock period to discount latch-window masking. These factors provide a worst case analysis of the impact of the propagation of transient faults.

Faults are injected into a post-synthesised gate-level representation, which will have a strong correlation with the corresponding physical circuit. The implementation will influence the types of gates selected and the way that they are connected. The statistical fault injection tool injects the same amount of faults into each circuit and the location of a injection is uniformly random. Only gates are targeted and not the input or output registers, which would be part of the pipeline. This ensures that we only take into account the logical composition of the circuit. In reality, of course, the probability of a fault would vary due to environmental and technology process factors.

Logical masking will be measured in terms of incorrect results (IR), which is defined as the number of wrong results from the addition, no matter how many output bits are in error, i.e. if a fault corrupted three output bits, then this is counted as one incorrect result. The rate of incorrect results, or just simply the error rate for this paper, is normalised by the number of

TABLE I
SIZE AND ERROR RATES FOR EACH 32-BIT ADDER IMPLEMENTATION

Implementation	Area (synthesis units)	Gates	IR (%)	IR (%)/Gate
pparch	1249	160	88.36	0.55
rpl	1467	192	77.67	0.40
bk	1562	235	81.06	0.34
clf	1626	246	81.26	0.33
rpcs	1669	224	73.46	0.33
cla	1702	268	72.68	0.27
csm	3166	345	60.98	0.18

target gates, not area, because an injection site is randomly chosen from the set of gates rather than being influenced by sensitive area. Of course, α will be proportion to sensitive area, but this work only analyses the effect of logical masking on $P(SE)$.

The adder implementations are created by Synopsys' Design Compiler synthesis tool with a TSMC 130nm process. Using the Synopsys Design Ware IP Library, Design Compiler is forced to select an implementation during synthesis. The following seven adder implementations were compared:

- Ripple-carry (rpl)
- Carry-look-ahead (cla)
- Fast carry-look-ahead (clf)
- Brent-Kung (bk)
- Conditional sum (csm)
- Ripple-carry-select (rpcs)
- Delay-optimised flexible parallel-prefix (pparch)

The input values have a uniform random distribution and change every clock cycle. Each adder implementation completes the operation in one clock period. The system waits for long enough after a fault injection before performing another to ensure that the effects of a fault do not overlap.

III. RESULTS

The results for the adders are based upon 100,000 samples, so that with a confidence level of 99%, this gives an estimation error of 0.816%.

The results show the rate of errors that appear at the clock cycle immediately following the one in which the fault was injected because this is when the result would be forwarded onto the next stage in processing. Since the time at which the fault is injected is random, a small number of errors show up in the next clock cycle. All the circuits within this section were synthesised at a frequency of 100MHz. Later, we will analyse the effect of synthesis frequency on the error rate.

Figure 1 shows the variation of logical masking with each adder implementation. They have been sorted in descending order by the number of incorrect results per gate from left to right and have been normalised to 'pparch', i.e. 'pparch' has the most number of wrong answers while 'csm' has the least. There is a 3.1X difference in the number of incorrect results per gate between the largest implementation, 'csm', and the smallest 'pparch'. The largest difference between the remaining implementations is about half of that. From Table I, 'csm' is about 2.5 times larger than 'pparch' (because it is composed mostly of multiplexers).

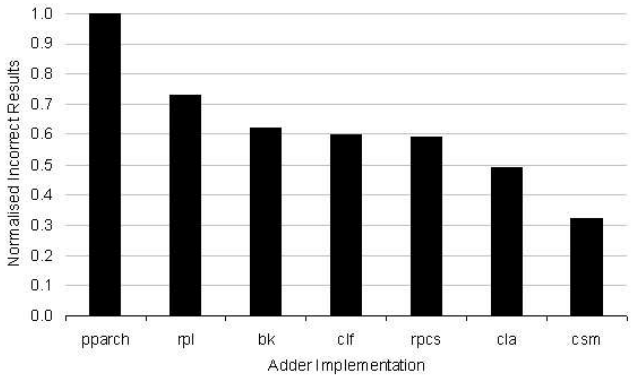


Fig. 1. Incorrect result error rate per gate for each 32-bit adder implementation (normalised to 'pparch')

The error rate per gate decreases for all implementations with increasing width. Across all the widths tested, 'pparch' has the largest error rate while 'csm' retains the smallest and the difference between them increases.

IV. DISCUSSION

From Figure 1, we can see that the implementation does have a significant impact on logical masking and is not just due to the number of gates (and thus the size). Logical masking will be influenced by signal propagation probabilities (i.e. if a fault that appears at the input of a gate, how likely is it to change the value of the output) and input values. Other measures of a circuit's complexity or composition may influence logical masking. If we view a circuit as a directed acyclic graph where the gates are the vertexes and the interconnecting wires are edges, then we can characterise the composition of a circuit using common measures of graph topology, such as the degree distribution and mean path length. In circuit terms, the degree distribution corresponds to the fan-in and fan-out distributions while we define the mean path length as the shortest distance (in terms of number of gates) between a target cell and an output register.

The fan-in distribution of a circuit will show the number of inputs a gate is likely to have. As the gates of all circuits under consideration are nearly all of two inputs this information does not help to reveal a relation with logical masking.

The majority, about 70%, of the gates in 'csm' have a fan-out greater than or equal to two, while for the rest the majority of gates have a fan-out of one. Counter intuitively, this would suggest that an increase in the mean fan-out of a circuit would give a lower error rate. To explain this, we need to look at the association of path length with the error rate.

Figure 2 shows that there is a fairly strong positive correlation between the product of mean fan-out and mean path length with the error rate. A circuit with a larger mean path length will imply that a fault can propagate to more outputs and therefore would be more likely to cause an incorrect

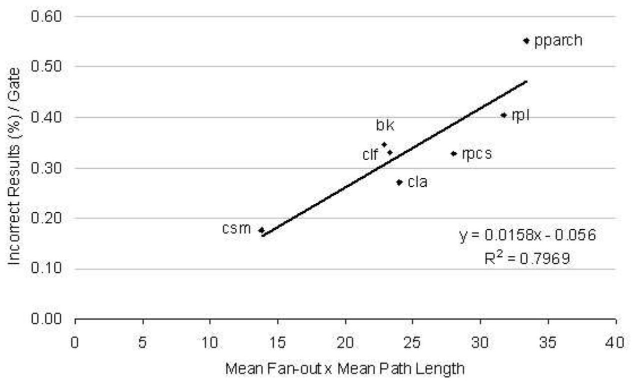


Fig. 2. Correlation between the product of the mean (shortest) path length and mean fan-out with the incorrect results for each 32-bit adder

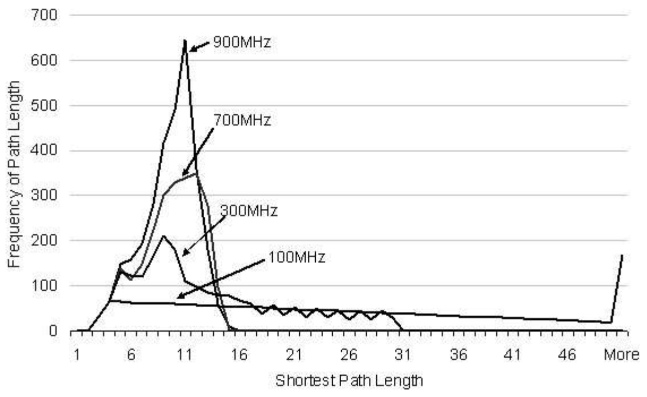


Fig. 3. Shortest path length distribution for 'pparch' synthesised at different frequencies

result. This would suggest that a lower mean fan-out and shorter distances between gates and outputs will result in higher masking. So although 'csm' has a higher mean fan-out than all other implementations, its affect on the error rate is outweighed by its low mean path length. Convergence of the paths is likely to have an impact, but is not taken into account in this work.

At a synthesised frequency of 100MHz, the distribution of path length frequency is rather flat and wide for all implementations except 'csm'. As the frequency at which the adders are synthesised is increased up until the maximum permissible, the distribution of path lengths normalises and the mean path length reduces. Figure 3 shows this for the implementation with the highest error rate, 'pparch'.

Therefore the error rate decreases with increasing synthesis frequency and the largest difference in error rate decreases too, such that at 900MHz the difference between 'csm' and 'pparch' is 2.1X. The bold line in Figure 4 shows this change in error rate with synthesis frequency for the 'pparch' implementation (with equal run frequency).

This increase in synthesis frequency, however, leads to more errors appearing in the cycle after the adder operation

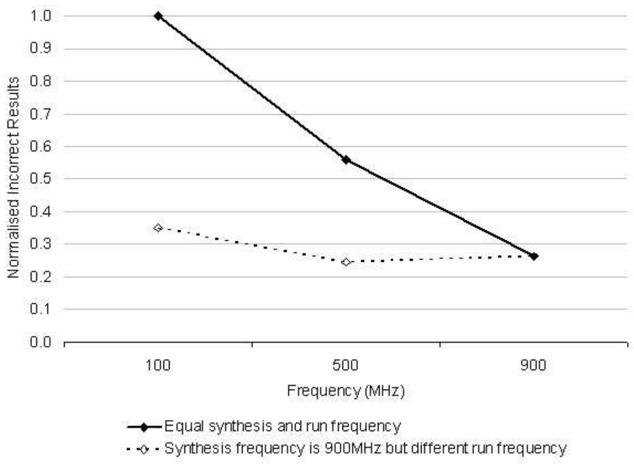


Fig. 4. Variation in incorrect results with synthesis and run frequency

completes, i.e. two cycles after the one in which the fault was injected. This can have serious consequences as errors will spill over into other operations that utilise the adder, which is clearly undesirable. For 'pparch' synthesised and ran at 100MHz, the number of errors that appear in the second cycle total only 1% of those in the first, although this increases to 12% at 900MHz.

The designer then has to trade-off how much the synthesis frequency is increased since this decreases the error rate for that operation (by lowering the mean path length), but increases the likelihood that an error will affect other operations.

If we consider that the objectives of a designer within this context are to maximise the performance of the circuit, minimise the area, and minimise the soft error rate, then the designer can control these constraints by varying the synthesis frequency (f_{synth}) and the run frequency (f_{run}), which reveals the following dependencies:

1) increasing f_{synth} increases the potential performance and lowers the mean path length and thus the error rate, but increases the area
2) increasing f_{run} improves the performance, but increases the likelihood of errors in subsequent operations

If we synthesise 'pparch' at the maximum frequency of 900MHz, but run it at various lower frequencies, we find that the error rates over time are less than the circuits where $f_{synth} = f_{run}$ (compare both lines in Figure 4) due to the advantageous changes in circuit composition made by the synthesis tool at the higher speed. This benefit diminishes as f_{run} approaches f_{synth}.

It may be of benefit to a designer targeting for a lower performance to run the circuit at a lower speed than that synthesised for, but at the cost of an increase in sensitive area. The overall gain will depend on the corresponding increase in the effective particle hit rate. This will also be important to designs that employ dynamic frequency scaling as the variation in run frequency will impact the logical masking of errors.

V. Conclusions

The effect that the micro-architecture of a circuit has on the logical masking was studied by using statistical fault injection on various adders.

It was shown that the implementation of the circuit has a significant impact on logical masking as the largest difference between the 32-bit adders was 3.1X. Logical masking is only one component that determines whether a SET becomes a soft error in combinational logic and so presents the theoretical worst case. The sensitive area, which is used to calculate the effective particle hit rate, must also be taken into consideration by the designer when choosing an implementation based on the SER.

Measures of the circuit's topology were explored to relate the composition to logical masking. The data suggests that a lower mean path length and lower mean fan-out results in a higher amount of logical masking. As highlighted in the case of the 'csm' implementation, which had the highest masking, the effect of a lower mean path length dominates an increase in mean fan-out. It was found that increasing the synthesis frequency results in a lower mean path length and error rate, but has the repercussion that more errors could materialise in subsequent operations that utilise the adder and thus increase the chance of system failure. It is proposed that the circuit be ran at a lower frequency than synthesised for to lower the error rate, which has implications for systems that employ dynamic frequency scaling.

The logical masking, and thus the soft error rate, of a crucial part of a microprocessor would help a designer - along with other metrics - to gauge the benefit between choosing a different implementation or applying a mitigation technique to the existing one, such as altering the composition of the circuit.

References

[1] J. Ziegler, "Terrestrial cosmic rays," in *IBM Journal of Research and Development*, vol. 40, no. 1, January 1996, pp. 19–39.

[2] C. Constantinescu, "Impact of deep submicron technology on dependability of vlsi circuits," in *Dependable Systems and Networks, 2002. Proceedings. International Conference on*, 23-26 June 2002, pp. 205–209.

[3] R. Baumann, "The impact of technology scaling on soft error rate performance and limits to the efficacy of error correction," in *Electron Devices Meeting, 2002. IEDM '02. Digest. International*, 8-11 Dec. 2002, pp. 329–332.

[4] P. Hazucha *et al.*, "Neutron soft error rate measurements in a 90-nm cmos process and scaling trends in sram from 0.25-um to 90-nm generation," in *Electron Devices Meeting, 2003. IEDM '03 Technical Digest. IEEE International*, 8-10 Dec. 2003, pp. 21.5.1–21.5.4.

[5] ITRS, "International technology roadmap for semiconductors 2005," ITRS. [Online]. Available: http://www.itrs.net/

[6] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Dependable Systems and Networks, 2002. Proceedings. International Conference on*, 23-26 June 2002, pp. 389–398.

[7] R. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *Device and Materials Reliability, IEEE Transactions on*, vol. 5, no. 3, pp. 305–316, Sept. 2005.

[8] M. Zhang and N. Shanbhag, "A soft error rate analysis (sera) methodology," in *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, 7-11 Nov. 2004, pp. 111–118.

[9] A. Taber and E. Normand, "Single event upset in avionics," *Nuclear Science, IEEE Transactions on*, vol. 40, no. 2, pp. 120–126, 1993.

[10] P. Hazucha, C. Svensson, and S. Wender, "Cosmic-ray soft error rate characterization of a standard 0.6-& cmos process," *Solid-State Circuits, IEEE Journal of*, vol. 35, no. 10, pp. 1422–1429, Oct. 2000.

[11] J. F. Ziegler and W. A. Lanford, "Effect of cosmic rays on computer memories," in *Science*, vol. 206, no. 4420, November 1979, pp. 776–788.

[12] Y. Tosaka, H. Kanata, S. Satoh, and T. Itakura, "Simple method for estimating neutron-induced soft error rates based on modified bgr model," *Electron Device Letters, IEEE*, vol. 20, no. 2, pp. 89–91, Feb. 1999.

Appendix C

DSN Paper

A low-tech solution to avoid the severe impact of transient errors on the IP interconnect

Derek Graham
Institute for System Level Integration
Derek.Graham@sli-institute.ac.uk

Per Strid
ARM Ltd

Scott Roy and Fernando Rodriguez
University of Glasgow

Abstract

There are many sources of failure within a System-on-Chip (SoC), so it is important to look beyond the processor core at other components that affect the reliable operation of the SoC, such as the fabric included in every one that connects the IP together. We use ARM's AMBA 3 AXI bus matrix to demonstrate that the impact of errors on the IP interconnect can be severe: possibly causing deadlock or memory corruption. We consider the detection of 1-bit transient faults without changing the IP that connects to the bus matrix or the AMBA 3 standard and without adding extra latency while keeping the performance and area overhead low. We explore what can be done under these constraints and propose a combination of techniques for a low-tech solution to detect these rare events.

1 Introduction

Extensive work has been invested in the design [8, 2] and verification [11, 1] of dependable computers for high RAS (Reliability, Availability and Serviceability) applications, however, these techniques are not likely to be suitable or required for embedded SoCs, where dependability requirements vary widely with market segments – from mobile to automotive. Much work in embedded systems has focussed on the protection of microprocessors, with ARM offering processors with support for ECC on various types of memory and support for a redundant processor [5]. However, these only form part of a complete SoC, so it is important to look at other components that affect its reliable operation.

This paper examines the impact that transient faults can have on ARM's AMBA 3 AXI fabric – the standard bus based IP interconnect in embedded SoCs. When qualifying the reliability of a device, it is important to realise that we cannot just consider the overall failure rate, we need to know the severity because the consequence of each failure may not be the same. Although the chances of such an error occurring are small, the risk of not considering the interconnect fabric is still high due to the severity of the impact.

There are many sources that can induce transient faults, such as electromagnetic interference, electrostatic discharge [9], power supply noise and crosstalk [14], radioactive particle strikes (soft errors) [10, 15], and aging (e.g. NBTI [12]). However, before it progresses to an error it has to avoid being masked by bus logic – either logically, electrically, or from the latch-window effect. Logical masking takes effect in various ways: the fault has to occur when the bus is being utilised, it has to occur at certain locations in the logic, and at certain times during a transaction. These type of events are random, rare, and depend on many factors including the surrounding environment, process technology, and circuit layout. This uncertainty makes estimating the probability of these errors difficult, but as their occurance is thought to be increasing with further integration (e.g. [4, 13] for soft errors) and as their impact can be high they warrent enough concern in certain applications to design systems so that the time between failure is at least maintained.

It is not uncommon in a SoC for the AXI bus matrix to constitute around 20-30% of the die area, which with increasing bus matrix sizes mean that a significant amount of logic is susceptible to the types of faults mentioned above.

2 Context

Within this work we make the following assumptions:

- we consider transient faults only
- 1-bit error
- can't change the master and slave IP that connects to the bus matrix
- can't change the AMBA 3 standard
- no extra latency
- low frequency overhead

- minimal area overhead

We explore what can be done under these constraints and aim for a low-tech solution to detect these rare events because low overheads will decrease the friction for adoption and a highly complex scheme may actually make the device more unreliable.

We believe that the automotive and networking, specifically storage networking, will be most likely interested in this work as they both care about dependability, but in different ways: safety and availability, respectively. This affects their requirements where automotive are sensitive to latency but less so to performance and area and as exemplified in ISO26262, the automotive version of IEC61508 [6] in development, error detection is satisfactory. For networking, performance is critical and can have extra register stages to keep throughput high, but are likely to require error correction.

3 AXI Overview

The AXI bus protocol [3] is burst based and has five independent unidirectional channels – three for writing and two for reading – that carry the address/control and data phases. Every transaction has address and control information on the address channel (AW for the write channel and AR for the read channel) that describes the nature of the data to be transferred. The data is transferred between master and slave using a write data channel (W) to the slave or a read data channel (R) to the master. In write transactions, in which all the data flows from the master to the slave, the AXI protocol has an additional write response channel (B) to allow the slave to signal to the master the completion of the write transaction or an error. A burst is made up of a number of transfers, which is a unit of data whose width is specified on the AW or AR channel. See Table 1 for a description of signals on each channel. The signals have been grouped into types as they may feature on more than one channel, e.g. the signal type is prefixed with the channel abbreviation so ID features on all channels: AWID, ARID, WID, BID, and RID. Those signals types with a * suffix are referred to as the "control signals".

AXI uses a two wire handshake, using the VALID and READY signals, to exchange information between the sender and receiver. VALID is used to show when valid data or control information is available and the destination uses the READY signal to show when it can accept the data. Both the read data channel and the write data channel also include a LAST signal to indicate when the transfer of the final data item within a transaction takes place.

The standard also features a sideband signal, USER, up to 32 bits on each channel that allows the SoC designer to extend the protocol to carry implementation specific information.

4 Impact of Errors on the AXI Bus Matrix

We can classify the impact of an error on each AXI signal into those that cause deadlock (DL), memory corruption (MC), and undefined behaviour (UB) (so failure may result depending on the application). Memory corruption could cause failure later in time, if the data is used at some point or if it incorrectly controls an actuator. This was tested by injecting faults at the RTL and observing the consequences. Lin et al. [7] perform fault injection on the AMBA AHB protocol (targeted for lower performance) and similarly categorised errors. They also invented a scheme to predict the effect of a single bit error on a signal for any application with high accuracy.

Table 1 shows the impact for each signal and two examples are given below. These cases were tested by injecting faults into the RTL of the bus matrix.

4.1 LEN

The LEN signal specifies the number of transfers (i.e. data chunks) within a burst. We have to consider the effect of an error changing LEN to a higher value or a lower value. In the former case, it will cause deadlock because the slave expects more transfers than the master will send so the slave blocks which causes the master to block as it is waiting on a write response on the B channel. This prevents any other master from accessing that slave and thus blocks itself, which can cause the whole bus to deadlock. In the latter case, the slave uses the LAST signal to determine how many transfers to consume so it still receives the correct amount. However, the slave uses LEN to calculate the address (which is only sampled once at the begining of a transaction) so can cause memory corruption. In the test setup, we wrote two 32-bit transfers to a 64-bit wide memory and changed LEN from two to one, which did not corrupt memory because of the size of the transfer and memory alignment. This highlights the subtle masking that depends on the properties of the transfer and configuration of the bus matrix.

4.2 VALID

The VALID signal is one part of the two-way handshake, which informs the receiver that data is available. An error on VALID can cause deadlock if it transitions from high to low. For example, if the slave has AWREADY high (i.e. the slave is ready to accept information on the address write channel) and the master drives valid data, but an error causes the slave to miss AWVALID high on the following clock cycle then the master thinks there has been a handshake and de-asserts AWVALID so that the slave never realises there was a transfer. The master blocks because it

Table 1. Source: M = Master, S = Slave. Impact: DL = Deadlock, MC = Memory Corruption, UB = Undefined Behaviour. Those signals types with a * suffix are referred to as the control signals.

AXI Signal Type	On Channel	Source	Description	Impact
ID*	All	M, S	The identification tag for the master	DL
ADDR	AW, AR	M	The read or write address	All
LEN*	AW, AR	M	The number of transfers within a burst	DL, MC
SIZE*	AW, AR	M	The width of each transfer in the burst	MC
BURST*	AW, AR	M	Specifies how the address for each transfer is calculated	MC
LOCK*	AW, AR	M	Additional information about the atomic characteristics of the transfer	DL, UB
CACHE*	AW, AR	M	Additional information about the cacheable characteristics of the transfer	UB, MC
PROT*	AW, AR	M	Protection unit information for the transaction	UB, MC
VALID*	All	M, S	Whether information is available from the sender	All
READY*	All	M, S	Indicates that the recipient is ready to accept the information	DL
DATA	W, R	M, S	The actual read or write payload data	MC, UB
STRB*	W	M	Which byte lanes to update in memory	MC
LAST*	W, R	M, S	Indicates the last transfer within a burst	All
RESP	B, R	S	Indicates the status of the write or read transaction	UB

doesn't get a write completion response.

If an error causes a transition on VALID from low to high and the slave is waiting on a handshake then it will sample garbage from the bus which could corrupt memory or lead to undefined behaviour.

5 Proposal

Addressing reliability issues can be tackled using redundancy in three main ways: space, time, and information redundancy. Time redundancy is disregarded because of the no extra latency and low performance overhead constraints. Looking for the simplest techniques, we first see how space redundancy can't be used and then how we conclude with a mix of information redundancy through three techniques.

Since we primarily care about detection, in the simplest case for space redundancy we can't duplicate the entire bus matrix as this obviously doubles the area and will increase the chance of an error occurring in the first place. If a customer believes that only the traffic between certain master and slave IP is important, then we can save area by duplicating those communications by instantiating another matrix connected to only those IP. However, there is a limit on the size of this smaller duplicated bus matrix relative to the original to ensure that area overhead requirements are met and, more importantly, an error that occurs between the other IP – whose traffic is not duplicated – can cause the whole bus matrix to deadlock. Another approach would be to duplicate only the control signals on which an error can cause deadlock, however, the area increase is too much: using the sideband signals of the AMBA standard to carry these duplicated signals – which has a maximum width of 32 bits – increases the area of a realistically sized bus matrix with eight masters and six slaves by 42%, so it's not scalable.

Three techniques are proposed, 1) for the control signals, 2) address signals, and 3) the data payloads. These different approaches come about because placing the parity encoders inline reduces the maximum frequency too far for the wide address (32 bits) and data (up to 1024 bits) signals. Even though the likelihood of failure in some of these cases is rare, the severity is still high and the proposal is that parity and ECC can detect them at a low cost.

5.1 Control Scheme

These measurements are made on bus matrix with two masters and two slaves with both masters able to access both slaves. This is intentionally small because if the overheads are acceptable on this, then they will be acceptable on larger matrices since the growth in area due to the size will dominate the growth in area needed for covering more channels with parity.

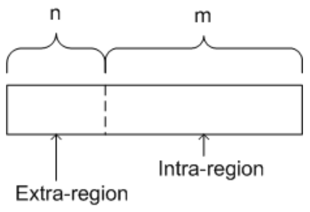


Figure 1. Address split into the n bits which specify which region and the m bits which address within a region

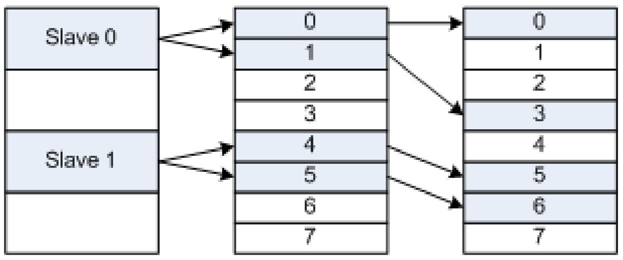


Figure 2. The memory map for two slaves split into smaller regions and then remapped to be separated by a Hamming distance of two

1-bit inline parity on the control signals (14 on the AW channel) results in a negligible area overhead and no frequency drop. However, applied on the 32-bit address signal there is drop in the maximum synthesisable frequency of 60%, which is clearly too high. If four bits of parity are used the maximum frequency halves with a 25% increase in area mainly due to the extra sideband signals that are needed to accommodate the extra wires needed for each parity bit.

5.2 Address Scheme

An erroneous value on the address signal can cause the transaction to address a different location within the same slave or re-route it to a different one, thus causing memory corruption and possibly further severe consequences later. This requires that the error code to detect an error must be sent with the address. Since the overheads for protecting the entire address width with inline parity is too great, it is proposed that only a portion of the least significant bits are parity protected (which defines the memory region size - bits 'm' in Figure 1) and the memory map is rearranged so that a bit error in the most significant bits (which specifies the region - bits 'n' in Figure 1) causes the transaction to be routed to an unmapped region, which according to the AXI protocol will cause an error response to be returned to the master. Although this limits the SoC to half of the original memory, in real automotive safety systems more than half is rarely used. This will likely not be the case in networking systems, but then the extra cycle of latency introduced with a register slice can be tolerated, therefore this technique is not needed.

The following scheme is used to determine which regions to use. In Figure 2 we have split the memory map for two slaves into eight regions, so the top three bits of the address, *n*, index them. The remaining bits, *m*, are protected with parity. Only those region indexes separated by a Hamming distance of two are mapped, which means that a 1-bit error in any of the top three bits will cause the master to address an unmapped region, which will automatically induce an error response to the sender. This technique can be expanded to use a Hamming distance of three, so that 1-bit errors are corrected, but at the cost of quartering the available memory space. Consequently, this scheme requires a minimal change to the software to alter the header file to use the new memory map and at worst a re-compile, but not any change to the functionality. The area and frequency overheads are acceptable on realistically sized bus matrices. Thus, 1-bit errors on the address signal can be detected without adding extra latency, but at the cost of halving the addressable memory and a re-compile of the software.

5.3 Data Payload Scheme

The constraint on the address was that the error had to be detected before it reached its destination. With the data payload it is proposed that this is relaxed so that the data is allowed to pass through without being checked immediately. An ECC is sent on the sideband signals in the subsequent cycle so that it can correct the value if there was an error, which is then re-written. It is sent on the following cycle because the ECC takes some time to decode and check and we don't want to introduce latency.

In some cases, the payload will be checked by the destination, for example when it is transmitted across an external transmission line to another processor, however there are cases where it won't be as it is not expected that an error can occur on the data when sent across the bus. Although memory may have ECC, it is only to protect errors that occur while that value is held there. An incorrect data payload can incorrectly configure a peripheral, such as a brake controller, or send it a wrong command. The assumption here is that the erroneous data value written will not have an immediate impact. The reason we can make this assumption is that some peripherals operate far slower, so there is time to correct either by re-writing the value or by simply resetting the peripheral. This may not be applicable to all types of memory, e.g. a FIFO, but at least resetting the device allows the failure to be controlled.

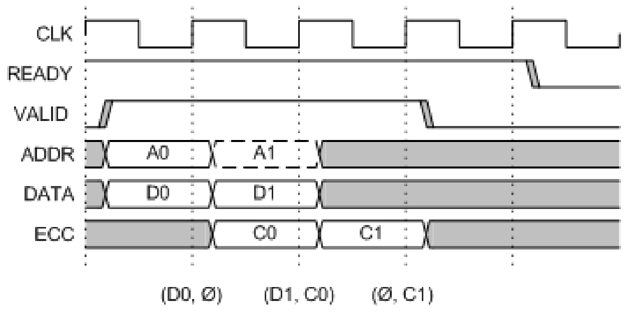


Figure 3. Data payload scheme timing with no error. The address is only sent at the start so A1 (address one) in a dotted outline is only for illustration purposes. The tuple at the bottom of the diagram shows what information the slave has received, i.e. the data and the checksum (ϕ for null)

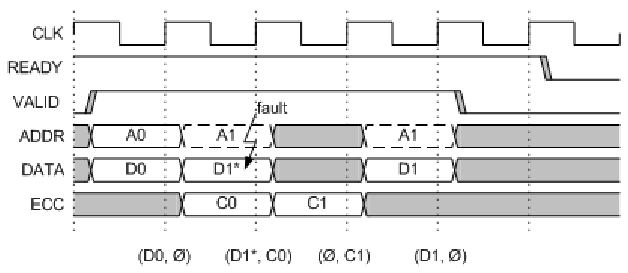


Figure 4. Data payload scheme timing with an error on D1 (data value 1) indicated by the * suffix

Figure 3 shows an example of the timing in the case of no error, while Figure 4 highlights what happens when there is an error. There needs to be an extra transfer at the end of the transaction for the last ECC, but this extra cycle will only impact the latency of the subsequent transaction if it occurs immediately after the last. When an error is detected the master is simply stalled by one cycle to allow for correct, so that latency is only introduced when there is an error.

The address of the unchecked payload is buffered until the ECC check is made. If it passes then the value is cleared from the buffer, but if it doesn't then the master is stalled, and the correct data is written to the stored address before it is cleared. The area overhead is small since it includes the ECC decode and a buffer which needs to store two 32-bit addresses at most (one for the address of the new unchecked payload and the other for last payload whose address is needed to re-write the data value if there was an error). Thus an error can be corrected on the data payload with low cost and without introducing extra latency, but only where it is deemed safe that the error will not have an immediate impact.

6 Conclusion

The impact that 1-bit transient faults on the AMBA AXI bus matrix can have was examined and it was found that the consequences can be severe, possibly leading to deadlock, memory corruption, or undefined behaviour. Under the constraints that the master and slave IP that connect to the bus matrix can't be changed nor the AMBA standard, that no extra latency should be introduced, and the frequency and area overheads minimised, a combination of three low-tech schemes were proposed: parity on the control signals, re-arranging the memory map using a Hamming distance of two, and delayed ECC on the data payload. This a first step in improving the reliability of the AXI interconnect which represents an attractive choice now for the target markets, automotive and networking, and with increasing silicon unreliability due to shrinking may lead to further techniques in the future.

References

[1] H. Ando, K. Seki, S. Sakashita, M. Aihara, R. Kan., and K. Imada. Accelerated testing of a 90nm sparc64 v microprocessor for neutron ser. *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2007.

[2] H. Ando, Y. Yoshida, A. Inoue, I. Sugiyama, T. Asakawa, K. Morita, T. Muta, T. Motokurumada, S. Okada, H. Yamashita, Y. Satsukawa, A. Konmoto, R. Yamashita, and H. Sugiyama. A 1.3-ghz fifth-generation sparc64 microprocessor. *IEEE Journal Solid State Circuits*, 38(11):1896–1905, 2003.

[3] ARM. Amba 3 axi specification. (http://tinyurl.com/dfxt62).

[4] R. Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *Device and Materials Reliability, IEEE Transactions on*, 5(3):305–316, Sept. 2005.

[5] D. Bradley and J. Penton. A perspective on developing ip for embedded reliability. *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2007.

[6] IEC. Functional safety and iec 61508.

[7] I. Lin, S. Srinivasan, and N. Vijaykrishnan. Transaction level error susceptibility model for bus based soc architectures. In *Quality Electronic Design, 2006. ISQED '06. 7th International Symposium on*, pages 6pp.–780, 27-29 March 2006.

[8] M. Mack, W. Sauer, S. Swaney, and B. Mealey. IBM POWER6 reliability. *IBM Journal of Research and Development*, 51(6):763–774, 2007.

[9] D. Marsh. Silicon safeguards automotive circuits. EDN Europe, http://tinyurl.com/d3bl5c (retrieved 2008/12/15).

[10] T. May and M. Woods. Alpha-particle-induced soft errors in dynamic memories. *Electron Devices, IEEE Transactions on*, 26(1):2–9, Jan 1979.

[11] P. Sanda, J. Kellington, P. Kudva, R. Kalla, R. McBeth, J. Ackaret, R. Lockwood, J. Schumann, and C. Jones. Soft-error resilience of the ibm power6 processor. *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, 52(3):275, 2008.

[12] D. Schroder and J. Babcock. Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing. *Journal of Applied Physics*, 94:1, 2003.

[13] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, B. Gill, and J. Maiz. Radiation-induced soft error rates of advanced cmos bulk devices. In *IEEE International Reliability Physics Symposium Proceedings, 2006*, pages 217–225, March 2006.

[14] K. L. Shepard and V. Narayanan. Noise in deep submicron digital design. In *Proc. IEEE/ACM International Conference on Computer-Aided Design ICCAD-96. Digest of Technical Papers*, pages 524–531, 1996.

[15] J. Ziegler. Terrestrial cosmic rays. In *IBM Journal of Research and Development*, volume 40, pages 19–39, January 1996.