# Fault-tolerant integrated interconnections based on built-in self-repair and codes

Von der Fakultät für Mathematik, Naturwissenschaften und Informatik der Brandenburgischen Technischen Universität Cottbus

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing)

genehmigte Dissertation

vorgelegt von

Diplom-Elektrotechniker
Daniel Scheit
Geboren am 11.04.1981 in Frankfurt/Oder

Gutachter: Prof. Dr. H. T. Vierhaus
Gutachter: Prof. Dr. M. S. Reorda
Gutachter: Prof. Dr. M. Gössel

Tag der mündlichen Prüfung: 12.07.2011

# Abstract

The reliability of interconnects on integrated circuits (IC) has become a major problem in recent years because of the rise of complexity, the low-k-insulating material with reduced stability, and wear-out-effects from high current densities. The total reliability of a system on a chip is increasingly influenced by the reliability of the interconnections, which is caused by increased communication from the elevated number of integrated functional units. In recent years, studies have predicted that static faults will occur more often decreasing the reliability and the mean time to failure. The most published solutions aim to prevent dynamic faults and to correct transient faults. However, built-in self-repair (BISR) as a solution for static faults has not previously been discussed along with the other possible solutions. Theoretically, BISR can lead to higher reliability and lifetime. This is my motivation to implement BISR for integrated interconnects. Because BISR cannot repair transient and dynamic faults, I combine BISR with other approved solutions in this thesis. The results show that the combination leads to higher reliability and lifetime with less area and static power overhead compared to the existing solutions.

built-in self-repair, error correction code, integrated interconnection

# Kurzfassung

Die Zuverlässigkeit von Verbindungen integrierter Schaltungen (ICs) hat in den vergangenen Jahren an Bedeutung zugenommen. Dies liegt an der steigenden Komplexität der Schaltungen, an der verfrühten Alterung durch hohe Stromdichten und neuen Materialien, die zwar die Übertragungseigenschaften verbessern, aber die Zuverlässigkeit verringern. Die Chip-Zuverlässigkeit wird zunehmenden durch die Zuverlässigkeit der Leitungen beeinflusst, während der Einfluss der Logik-Zuverlässigkeit abnimmt. Dies liegt vor allem am steigenden Kommunikationsbedarf durch die steigende Anzahl integrierter Einheiten. Publikationen der letzten Jahre zeigen, dass vor allem mit einem Anstieg permanenter Fehler zu rechnen ist, welche sowohl die Zuverlässigkeit als auch die Lebensdauer verringern. Dem steht entgegen, dass die Vielzahl der Publikationen für fehlertolerante Verbindungen vor allem Lösungen für dynamische und transiente Fehler präsentieren. Der Einsatz von Selbstreparatur wurde nicht im gleichen Umfang diskutiert. Dabei kann sie zu höheren Zuverlässigkeiten hinsichtlich statischer Fehler führen. Da sich Selbstreparatur nicht für transiente Fehler und nur teilweise für dynamische Fehler eignet, wird in dieser Arbeit gezeigt, wie sich Selbstreparatur und Codes kombinieren lassen. Die Ergebnisse zeigen, dass die Kombinationen zu höheren Zuverlässigkeiten bei geringerem Schaltungsaufwand im Vergleich zu bestehenden Lösungen führen.

Selbstreparatur, Fehlerkorrektur-Codes, integrierte Verbindungen

# Contents

# List of Figures

# Chapter 1

# INTRODUCTION

According to the International Roadmap of the Semiconductor Industry (1), the total wire length on a chip will increase continuously in future developments. Simultaneously, the wire pitch and diameter will shrink, while the aspect ratio will increase. The current density will grow because the voltage cannot be reduced on a linear scale with the wire diameter. Hence, the RC delay will increase. These trends have a negative impact on the reliability of the chip and system. A longer wire has a higher probability of failing compared to a shorter wire, under the assumption that all of the other parameters are equal. The same is true for the number of wires. The decreased wire pitch makes fabrication more difficult, making faults more likely. While defects introduced at the time of production may be one cause, defects that may occur due to wear-out effects that are caused by high current density and subsequent metal migration effects seem to gain importance with current trend of feature size miniaturization. A high current density under higher temperatures or mechanical stress between metal and silicon can lead to a transport of metal atoms. This transport leads to voids and hillocks, which can result in a broken wire or shorts because of broken insulator layers. This increasing aspect ratio leads to larger capacitances between adjacent wires. Coupling capacitances between wires lead to statistical variations in signal delays, which can result in dynamic faults. Voltage drops on supply lines make the circuit more prone to transient faults, which are caused, for example, by the voltage supply noise or electro-magnetic interferences. In summary, it is estimated that the number of interconnection faults will increase and that static faults will decrease in mean time to failure.

In facing this problem, several solutions for reliable interconnections have been

published. The majority of the published solutions aim at transient and dynamic faults. Only a few solutions aim at static faults. One of these solutions is built-in self-repair (BISR) that can correct static faults with the use of switches, spare wires and administrative logic. This thesis continues research on interconnection built-in self-repair because it leads to higher reliability and higher mean times to failure than other solutions, as will be shown later. Built-in self-repair requires less power and area than code-based solutions, such as the Hamming code; but built-in self-repair can only correct static faults. For this reason, built-in self-repair has to be combined with codes for transient and dynamic faults. In this thesis, I show how the BISR has to be implemented to be compatible with existing codes and other existing solutions. The combination of BISR and appropriate codes results in fault-tolerant interconnections, which are especially useful for point-to-point interconnects and can be found in the upcoming network-on-chip technology. Similar to other solutions, only the metal layers of the interconnection are considered. The additional logic is not considered in reliability calculations.

The structure of this thesis is as follows. Subsequent to the introductory chapter, the necessary background is described. Chapter 2 begins with a description of the possible faults and the solutions that can be applied to prevent or correct these faults. Solutions with a high relevance are described in more detail. The relevance is given by the impact on the reliability and by the degree of familiarity. During the discussions at conferences in which I have participated, for example, the question of why it would not be sufficient to simply widen the wires to increase the reliability have often been asked. The description of the selected solutions is necessary for the discussion in chapter 3, which shows why wire widening and other existing solutions are sub-optimal. I discuss why it is worthwhile to develop fault-tolerant interconnections based on a combination of codes and built-in self-repair. The implementation and evaluation of built-in self-repair that is compatible with existing codes is described in chapter 4. The first two sub-chapters of chapter 4 address the reconfiguration, and the next two sub-chapters address the administrative implementation and evaluation. Chapter 5 addresses the Code/BISR combination. The results show that the combinations lead to higher reliability, less area and less power consumption. The conclusions and a future outlook are given in chapter 6.

# Chapter 2

# BACKGROUND

The reliability of interconnections depends on multiple factors. It depends on the materials used, on the manufacturing process, on the working conditions, and on the actual communication architecture and layout. This chapter contains a description of interconnection faults and a broad view of solutions that prevent or correct these faults. The purpose of this chapter is to provide an overview of state-of-the-art technology for reliable interconnections, with a focus on fault-tolerance. These topics are required for the subsequent discussion in chapter 3.

## 2.1 Interconnection faults

According to their time of occurrence, faults can be divided into four classes: transient, intermittent, dynamic and permanent. Examples are shown in figure 2.1. The four classes are described as follows.

*Transient faults* are temporary malfunctions that cause single error events. They occur randomly, and no permanent damage is inflicted. Transient faults on interconnects can be caused by internal and external noise, electromagnetic interference

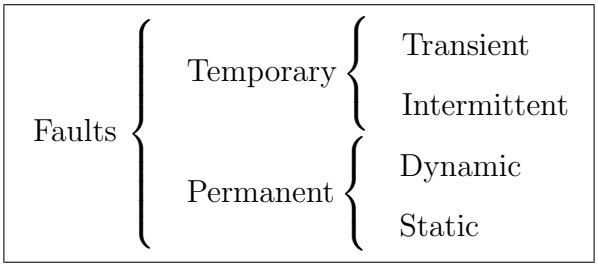


Figure 2.1: Time-related classification of faults

and electric discharges. The noise margin decreases because of supply voltage scaling and process variations. Process parameter fluctuations lead to variations in the transmission behavior of every single wire and, in addition, lead to parameter shifts between the repeaters that are used. This scenario reduces the signal integrity and may affect single faults, which are distributed statistically. Lowering the voltage supply leads to a growing impact of electromagnetic interference, which further increases with higher clock frequencies because the inductances on the wires become more important. External electric discharges can additionally lead to weakened circuits. Thus, they may ultimately lead to permanent faults (35).

*Intermittent faults* are error bursts that are activated by environmental changes or specific input combinations. If a wire crack, for example, changes the wire resistance as a function of the temperature, then this resistance change could lead to errors. Intermittent faults often precede permanent faults due to wear-out effects. A wire-resistant increase that results from electro-migration or an isolator-resistant decrease from time-dependent dielectric breakdown can cause signal delays and may eventually lead to stuck-at or bridging faults. If intermittent faults occur long enough to be testable, then they can be treated as permanent faults.

*Dynamic faults* are dependent on signal transitions. In figure 2.2, the multiple aggression fault model is depicted. It contains signal transitions that lead to the highest signal delay or that lead to glitches through capacitive coupling. Except for one wire (the victim), all of the other wires (aggressors) have the same value transition. If the victim has a constant value, then the transition of the aggressors leads to a temporary voltage drop or rise, which can result in a glitch. This glitch could lead to an error, for example, through faulty hand shaking. If the victim has a transition that is opposite to the transitions of the aggressors, this transition is delayed. If the delay is higher than the timing constraint, a fault occurs. For dynamic faults, there is a parameter called Lambda $\lambda$, which describes the strength of the coupling. The highest delay $D_{max}$ caused through capacitive coupling and signal transition is $D_{max} = d_0 + x \cdot \lambda$, with $d_0$ as the transition-independent delay and $x = 4$ the highest transition-dependent delay factor. The delay factor $x$ can be reduced through coding, which requires additional wires. The problem of dynamic faults increases with circuit down-scaling because of the increasing aspect ratio of the wires and the process variations during interconnection manufacturing. The transition-independent delay $d_0$ and the transition-dependent delay $\lambda$ are increasing.

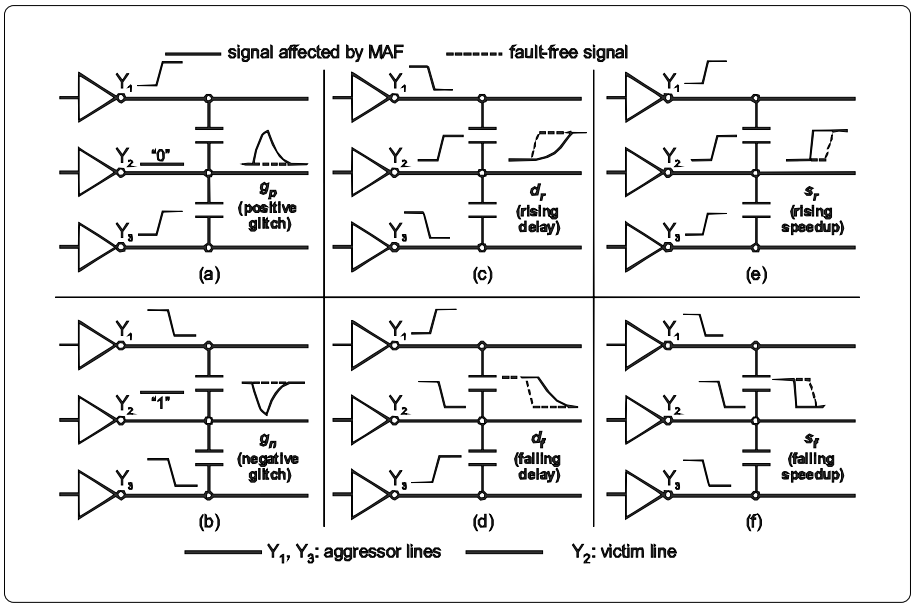


Figure 2.2: Multiple Aggression Fault Model (25)

*Static faults* are caused by local defects and are permanently present. These defects can occur during manufacturing or by wear-out effects during operations. Electro-migration (EM), stress-induced voiding (SM) and time-dependent dielectric breakdown (TDDB) are the main causes for static faults. Electro-migration describes the transport of metal atoms under high current densities and temperatures (17). It is enhanced by the growing number of metal layers, which lead to higher temperatures and higher current densities. Defects during manufacturing can narrow metal wires, which locally increase the current density. The connection area between vias and wires is also critical, especially where only single vias are used to connect wide wires (50). Stress-induced voiding or stress migration is a mechanism of metal atom transport caused by mechanical stress. The mechanical stress results from different thermal expansions of the materials used in the metal wiring. The stress leads to vacancy diffusion and further to void growth. Vias are the critical point because the highest stress gradient can be found between the wire-to-via contact (50). Time-dependent dielectric breakdown between wires can also limit the interconnection reliability. The problem of TDDB becomes more severe with the use of low-k materials, which are used to decrease the capacitive coupling. High potential differences between adjacent long wires, combined with high duty cycles, lead to a critical condition, which can cause bridging faults.

## 2.2 Fault prevention

Usually, there is an attempt to manufacture faultless interconnections, which means attempting to prevent faults and defects. In this section, I present the state-of-the-art technological advancement in error prevention. The solutions presented here are divided into three classes: solutions based on routing, solutions based on design methodologies, and architectural solutions. Routing-based solutions attempt to minimize capacitive and inductive coupling through various routing schemes. They also address static faults through a simulation-based reduction of electro-migration. Architecture-based solutions attempt to decrease the effects of electro-migration through a reversal process or through thermal management. The last class of solutions contains various design methodologies to overcome the problems of existing interconnection implementations.

### 2.2.1 Routing-based prevention

Dynamic and static faults depend on the shape of the wires and vias, the distance between adjacent wires, and their temperature. These parameters are considered during routing. To decrease the coupling capacitance between adjacent wires, the distance between them is increased, or a shielding wire connected to Vdd / Gnd is placed between them. Reducing crosstalk noise on interconnection trees using shielding wires is a solution that is currently being implemented (57). Interconnection shielding is applied iteratively, starting from the critical node segment towards the source. The delay could be reduced by an average of six percent in comparison to direct source shielding. Other possibilities for reducing delays are interconnection tuning and repeater sizing (42; 62). It has been shown that the optimal number of repeaters depends only on the wire shape and spacing. The total delay is a linear function of the path length using optimal repeater insertion. Spacing is more effective than shielding when using the same footprint.

One way to reduce inductive coupling effects is to twist the wires during the layout process, which was done in (66) and is called twisted bundle. The noise level is reduced by 4 to 76 percent, depending on the total wire length, the clock frequency, and the input pattern. The delay remains nearly the same in comparison with the original bus. To also address capacitive coupling, staggered twisted bundles have been implemented (65). The difference between these techniques is the use of

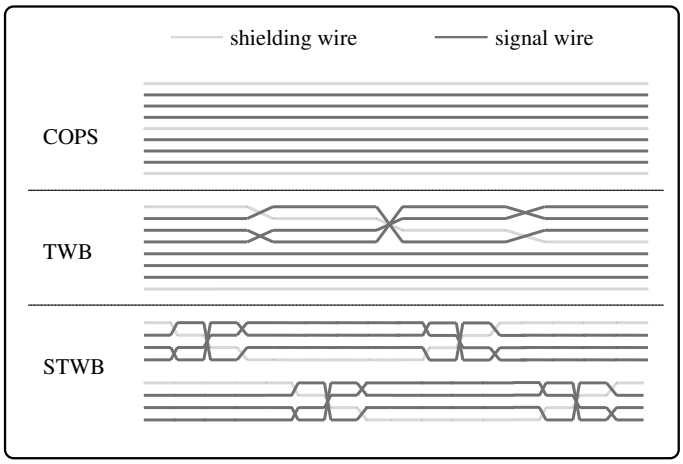


Figure 2.3: Comparison of Coplanar Shielding (COPS), Twisted Bundle (TWB), and Staggered Twisted Bundle(STWB) (65).

two groups of twisted bundles instead of one twisted and one normal group. This technique has been compared with coplanar shielding and twisted bundle, which can be seen in figure 2.3. The comparison shows that the staggered twisted bundle reduces the maximal noise and delay by approximately 6 to 20 percent. Using low-swing differential current-mode signaling with twisted differential lines (44) can also reduce crosstalk. Current-mode signaling uses a current source as a transmitter and a low impedance receiver. The received current-mode signal is isolated from the power supply. Energy is only consumed by charging and discharging wire capacitances. Current-mode signaling leads to a delay reduction of approximately 20 percent, compared with the optimal repeater scheme using voltage-mode signaling.

To prevent permanent faults such as the widening of interconnects, a reliability analysis by layout-based simulation (37; 51; 63; 64) is performed. Wider wires have a reduced current density and therefore a decreased electro-migration effect. The wire shaping and via usage depends on the simulation results. In figure 2.4, a part of an insufficient structure and the corrected layout is depicted. Figure 2.4 shows the simulation based widening of the wire and the use of additional vias (near T3).

Temperature has an exponential effect on electro-migration. Reducing the temperature is an effective way to increase the lifetime, which is otherwise limited by electro-migration. Thus, a good overall thermal management can result in reliable interconnections. Further steps can be a thermal-aware global routing such as the work performed with TAGORE (26). The interconnects are routed preferably on

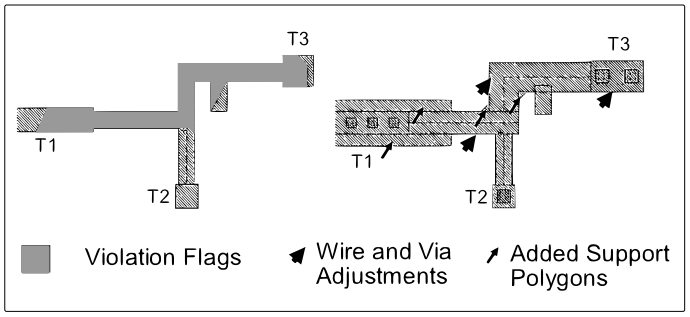


Figure 2.4: Electro-migration aware simulation of an interconnection layout (left) and the corrected layout (right) (37).

cold ship regions. However, lifetime increases lie between two and three percent, which could be traced back to the limited degree of freedom during routing.

### 2.2.2 Architecture-based prevention

Architecture-based solutions prevent faults during operation. One method is to change the workload, to decrease the temperature, which has a substantial impact on electro-migration. Thus, the dynamic thermal management (39) tracks reliability issues during operations. The chip temperature is measured periodically, to estimate the remaining lifetime. If the measured temperature is lower than the reliability-equivalent temperature, then the chip has saved almost a lifetime. This savings allows the chip to run with a temperature higher than the reliability-equivalent temperature for a certain time. Throttling is engaged only when it seems to be necessary, to prevent an reduction of lifetime. Architecture-based prevention, however, decreases the safety margins according to the wire width and decreases the performance penalty through throttling compared to previously published dynamic thermal management solutions.

Electro-migration describes the metal atom transport under high temperatures and current densities. This process is reversible. Abella et al. published an architecture that ensures that a bidirectional wire is used equally in both directions (2). This arrangement reduces the effect of electro-migration by a factor of up to $10^4$. Equalizing is accomplished by counting signal transitions for every wire in each direction. If equalizing (re-fueling) is necessary, then the wire is driven in the direction that has fewer transitions until the transition counts are equal. The slowdown is smaller than one percent for bidirectional wires and depends on the threshold when

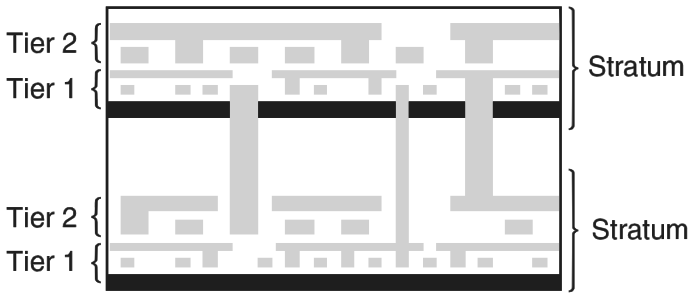


Figure 2.5: Cross-sectional structure of two stacked circuits connected with 3D interconnection (40)

the refueling process has started.

### 2.2.3 Design methodologies

There are several design methodologies that are designed to increase the reliability of interconnects. Using optical interconnects (18; 40) prevents crosstalk and facilitates satisfactory signal integrity. The delay of electrical interconnects seems to be constant with decreasing feature size. The delay of optical interconnects decreases feature sizes decrease because of the performance increase of the modulator driver and the receiver amplifier. The power consumption is less compared with the electrical interconnect. The bandwidth is higher if wavelength division multiplexing is applied. Of course, electro-migration does not take place. The main problems of optical interconnections are CMOS-compatible transmitters with small footprints (18).

To shorten the total wire length, 3D-interconnects have been implemented (38; 3; 40) . The concept behind this implementation is to stack chips, for example, to stack the memory of a processor on the processor die, as depicted in figure 2.5. Shorter interconnections lead to a higher throughput, less area and power consumption and a higher reliability, assuming that the reliability per wire length remains constant. Heat removal and the i/o interconnection are the main challenges. Temperature has an exponential impact on the mean time to failure; thus, reliability must also be considered.

Using new materials, such as single-wall carbon nanotubes, can reduce the problem of electro-migration through higher possible current densities (13). Carbon nanotubes would allow a decrease in feature size, power dissipation and delay, but they are not compatible with the CMOS process. The manufacturing process of

nanotubes underlies statistical variations, which require a selection or (built-in) self-repair process to ship faultless ICs.

## 2.3 Error correction

### 2.3.1 Codes

If, as in real life, other methods cannot prevent all of the faults, the remaining and occurring faults have to be corrected to ensure error-free system operation. Codes are one way to correct or prevent faults, especially dynamic and transient faults. There are three classes of codes: one to detect and correct transient faults (EDC/ECC), one to prevent dynamic faults (LXC/CAC), and one that combines the abilities of the codes to prevent dynamic and to correct transient faults (ECC+CAC).

**EDC and ECC** Error detection codes (EDC) and error correction codes (ECC) are mainly used to handle transient faults. Some of these codes can also handle permanent faults. The basic concept that is involved is to add redundant information using an encoder and to compare this information in the decoder circuit. The codes differ with respect to their overhead in terms of wires and logic, power consumption, signal delays, and the handling of errors.

A power-aware adaptive error protection has been published in (36). The power consumption of the coding logic depends on the numbers of transitions. The more signal transitions, the more power is consumed. The more faults that have to be detected, the more logic is required, and thus, the more power that is required. The immediate goal is to measure the noise and the density of fault events and to select the required protection. Depending on the measured signal integrity, one of three error detecting codes with different error detection capability is chosen. The power consumption could be fitted to the noise level, while ensuring the fault limits.

The Hsiao code and a less logic-consuming code are described in (34). The numbers of '1' values in the parity check matrix was decreased to simplify the encoding and decoding circuit. This code is mainly used for memory protection. To reduce the logic overhead, a lightweight hierarchical error correction code for multi-bit differential signaling (11) has been suggested. This code has been implemented especially for interconnects to increase noise immunity and to decrease the transient error rate. It uses multi-bit differential signaling, which is an alternative to low-voltage differ-

ential signaling with reduced power and area consumption. The data are encoded in such a way that half of the bits in each valid word are ones.

The trade-off between power consumption and reliability gain has been discussed in (16; 15). The discussion shows that the average energy per useful bit is lower for error detection codes in comparison with error correction codes. Further comparison between error recovery schemes according to power and usage in NOCs has been performed in (43). This study shows that end-to-end recovery is power-efficient for long link distances, and switch-level treatment is superior to short link distances. With respect to delays, a combination of both schemes is the best. To further increase the reliability of error detection/correction, coding in sections (bus guards) was implemented in (33). For this purpose, the interconnection is divided into several subsections, which contain an encoding and decoding circuit. The number of total errors that can be corrected grows linearly with the number of segments. The same is true for the delay.

If one combines error detection with retransmission, where the inverted pattern is transmitted, then it is possible to compensate even stuck-at-zero or stuck-at-one faults. This code was introduced by Shedletsky (58) as alternate data retry code (ADR code). Whenever a fault has been detected, a retransmission of the inverted pattern is triggered. Through the inversion, the effect of a stuck-at fault is compensated. If it was a transient fault, the retransmission leads also to a correct pattern. Shedletsky has shown how to implement a fault-tolerant data path using ADR. The use of ADR for fault-tolerant interconnections was not described explicitly. Further publications covering ADR and fault-tolerant interconnections together were not found in our literature search. Thus, the discussion of why this code is not the best solution for fault-tolerant interconnections, which will take place in chapter 3, is based on my own implementation of the ADR code.

**LXC and CAC** LXC and CAC Linear crosstalk codes (LXC) and crosstalk avoidance codes (CAC) are used to reduce the effects of capacitive coupling. Linear crosstalk codes such as wire duplication try to decrease the coupling capacitance or try to avoid signal patterns. Capacitive coupling can cause a signal delay. This delay depends on the pattern transitions. Crosstalk avoidance codes forbid either patterns or transitions that would cause the highest delay. They are called forbidden pattern code (FPC) and forbidden transition code (FTC). To reduce the logic over-

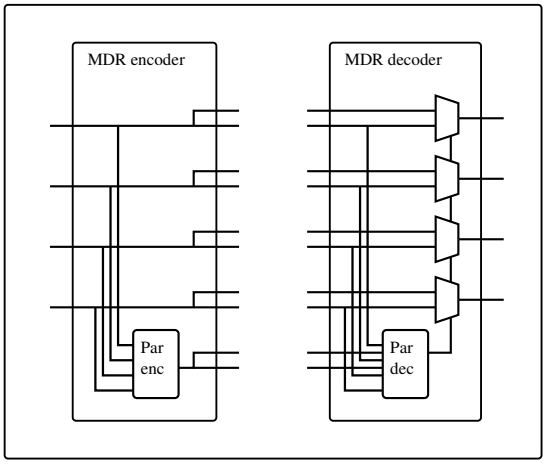


Figure 2.6: Modified dual rail

head, these codes have been overlapped, resulting in forbidden pattern/transition overlapping codes (FOC) (59). The usage of FPC, FTC and FOC in NOCs has been compared in (47). FTC is the most energy efficient scheme followed by FPC. FOC is the worst scheme according to energy efficiency but has the smallest area requirements. Instead of using only redundant wires, spatio-temporal coding uses fewer wires and time redundancy (28). This type of coding has been developed to decrease the crosstalk between a processor and memory, and the results show an improvement of up to 40 percent. A complex coding scheme that uses two cycles per transmission and local duplication to implement a one-lambda code is shown in (10). A one-lambda code has the smallest possible data dependent delay. The proposed code also detects one transient fault. Coplanar tapered interconnection wires have been combined with this spatio-temporal coding (56) to further reduce crosstalk.

**ECC with CAC** If transient and dynamic faults are present at the same time, joint crosstalk avoidance and error correction codes are one possible solution. Duplicate-add-parity code, modified-dual-rail code (53) and boundary-shift code (49) can correct one transient fault and limit the delay to two Lambda. Modified-dual-rail doubles every signal wire and adds a parity wire. Doubling the wire reduces the possible crosstalk, and the parity wire allows switching between the two groups of wires to correct one error. The comparison of Hamming and dual-rail code with further optimization has been performed by Rossi (54). The dual-rail code turns

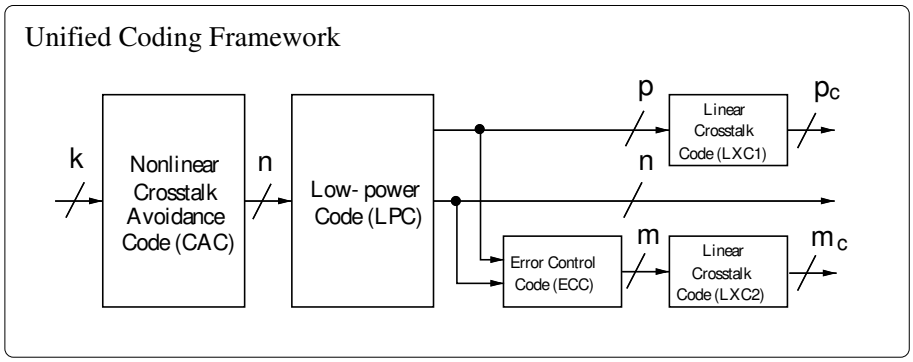


Figure 2.7: Unified coding framework (59)

out to have less coupling than the Hamming code implementation with the same footprint. The same relationship appears for the comparison of the Hamming code, the dual-rail code and the modified-eual-rail code (55) (figure 2.6). The modified-dual-rail code has a duplicated parity check bit, which leads to a decreased delay. DAP, MDR and BSC usage in NOCs hase been evaluated by Pande et al. (47). It was shown that they all reduce delays as well as power consumption. The MDR and DAP codes lead to nearly the same results and are better than BSC.

The crosstalk-aware double error correction code CADEC was published in (24). CADEC is the combination of the Hamming code and a Duplicate-Add-Parity code. The reliability is higher than for DAP, and the average energy per message is smaller than with DAP. A unified coding framework to combine ECC and CAC (figure 2.7) and a comparison of various combinations have been presented in (59; 61). There, it was possible to combine crosstalk-avoidance, error-correction, error-detection, and low-power codes. Figure 2.7 shows the general encoder for the combined codes.

### 2.3.2 Fault-tolerant communication architectures

Communication architectures can be divided into two classes: interconnection centric design and distributed interconnection design. Both classes are depicted in figure 2.8. A complex switching network handles the communication between the cores in the interconnection centric design. Multistage Interconnection Networks (MINs) are the most important implementation of this architecture. They consist of multiple switch stages, which work in serial. MINs allow a high bandwidth but are not that flexible to the number of cores like the distributed interconnection designs, which consist of independent switches. The most popular implementation is the

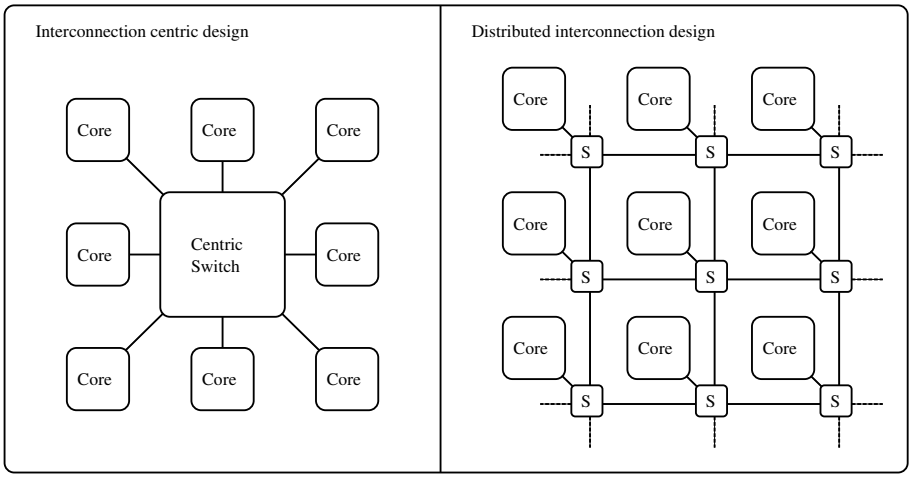


Figure 2.8: Interconnection centric and distributed interconnection design

network-on-chip (NOC) methodology. Both classes may suffer from switch and link failures. There are fault-tolerant implementations to handle this fault, which mainly use redundant switches or adaptive routing algorithms.

A fault-tolerant MIN using intrinsic redundancy and an FPGA reconfiguration technique has been published in (4). The basic switch element, called slice, has two properties. It is re-arrangeable and non-blocking. These properties allow it to compensate for faults in the slice through reconfiguration. Combined with FPGA reconfiguration techniques, they are able to compensate for faults outside of the slices. The combination of the fault-tolerant slices with FPGA reconfiguration allows compensation for multiple faults.

In (22), multiple switch faults can be tolerated with a minimal number of extra stages. An extra stage is an additional switch stage, which increases the degree of freedom to route the signal. This additional stage can make it possible to bypass a faulty switch. Fan and Bruck showed that their fault-tolerant MIN uses the extra stages optimally, which means that the extra stage is used efficiently.

A chip multi-processor switch with fault-tolerance and built-in self-repair (BISR) is proposed in (20), to fit the requirements of nano-technology. This switch provides system-level checking and recovery, component-level fault diagnosis, and spare-part reconfiguration. It is divided into clusters with equal sizes using a min-cut algorithm. This division is performed using spares or triple modular redundancy with a higher granularity. The investigators show that traditional techniques such as triple modular redundancy and error correction codes are not as efficient as end-to-end

error detection, resource sparing, and iterative diagnosis/reconfiguration.

An example of adaptive routing is published in (5). Ali et al. use a fault-tolerant protocol with retransmission for transient and dynamic routing for permanent faults. The packets are routed the shortest way possible. When a link or a switch is faulty because of a permanent fault, the routing tables are updated. This task is performed globally for all of the switches, to ensure that the network is stable. Because each switch has the same routing tables, the shortest path can be recalculated. This procedure allows for graceful degradation to occur and ensures that the bandwidth decreases only slowly with an increasing number of permanent faults, a scenario that does not occur often.

### 2.3.3 Test

Testing provides a mechanism with which faulty behavior can be addressed. Depending on the times and locations of the tests, testing can be divided into two classes: manufacturing tests and in-field tests. Manufacturing testing consists of all tests until a chip is shipped. In-field testing includes all in-field tests, such as the startup test to check for faultless operation and built-in self-test to diagnose faults as a prerequisite for built-in self-repair.

#### Manufacturing test

By analyzing the distribution of metal open resistances, weak open defects that cause delay faults can be detected (41). A weak open defect will eventually result in a stuck-at fault; thus, detecting these defects during production testing prevents an in-field failure. When the locations of full open defects are diagnosed, refinements of the layout can be made (52). First, an open defect is detected with a logic test. The position of this open defect is diagnosed by using adjacent wires to influence the logic level of the floating wire. Test pattern generation for signal integrity faults, which are designed to prevent hot carrier injection (HCI) and time-dependent dielectric breakdown (TDDB), are described in (8; 9). Skew and noise violations can be detected using detector circuits. These violations can be used to prevent overshoots, which can lead to HCI and TDDB.

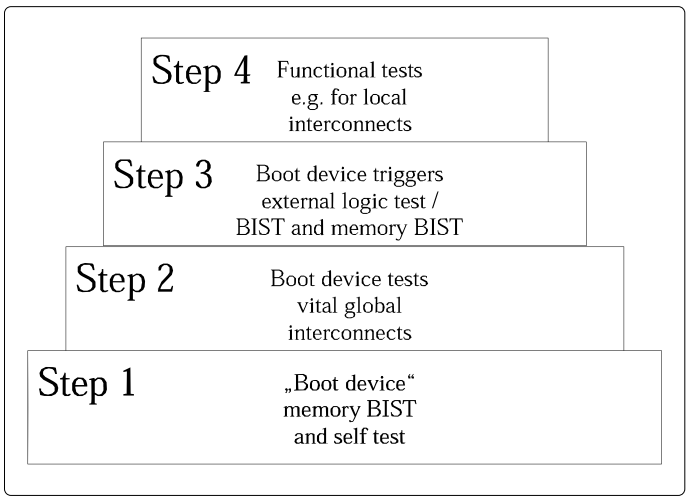


Figure 2.9: Hierarchical system-on-chip test (29)

**Built-in self-test**

There are built-in self-tests for static and dynamic faults. Testing for static faults is more common, but with stronger coupling, the need for dynamic tests increases. Global interconnects can be used as a test access mechanism (TAM); thus, they have to be tested before the integrated cores are tested. A solution of a hierarchical SOC test (29; 32) is shown in figure 2.9. Based on this solution, the interconnects are tested with data reflection (23). A test pattern is written on the interconnect, and on the other side, a Busreflector inverts the test pattern. This test finds all stuck-at and dynamic faults.

A built-in self-test architecture for network-on-chip has been presented in (25). This test is based on the maximum aggression fault model and tests for dynamic faults. The maximum aggression fault model assumes one victim wire and the remaining wires are assumed to be aggressor wires. The logic state of the victim and the aggressors are complementary. This setting causes the largest delay during the inversion of the whole pattern. This scheme is used to test the interconnection for dynamic faults. In figure 2.10, the test pattern for dynamic faults and the corresponding finite state machine of the built-in self-test controller are depicted. All dynamic faults for one wire can be tested in eight clock cycles by nesting the test patterns. The test pattern can also be generated using Busreflectors similar to the testing performed in (23); however, twelve clock cycles are required.

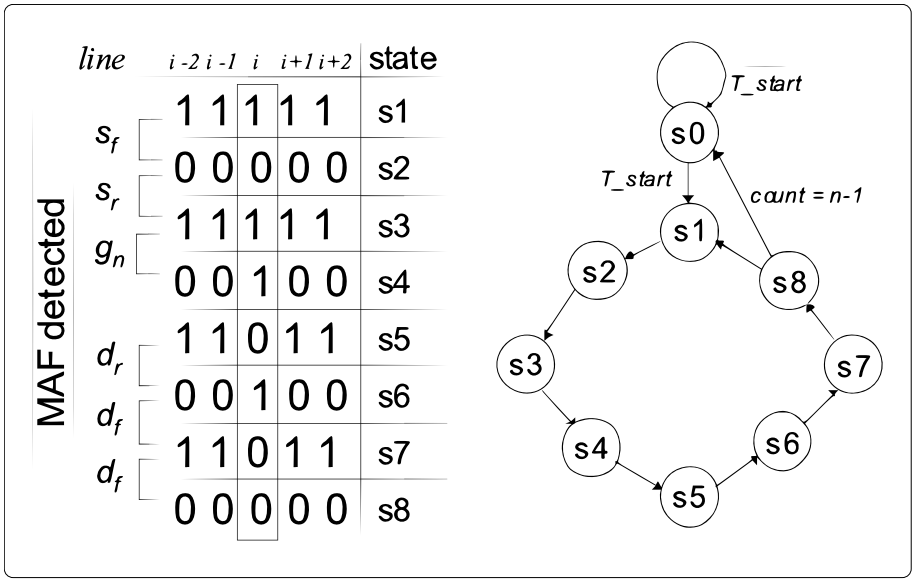


Figure 2.10: Test patterns for all possible dynamic faults on one wire using the multiple aggression fault model and the according finite state machine (25)

### 2.3.4 Built-in Self-Repair

Repair and built-in self-repair have been used mostly for regular structures such as memory and programmable logic arrays, to increase the yield. For this purpose, programmable and laser fuses have been used (19; 7). Laser fuses are used to (re-)configure the redundancy permanently after production testing. Programmable fuses are used to reconfigure the redundancy in the field of application. The yield of Memory BISR has been discussed with respect to its use in nanometer technology (46). A hierarchical approach is used, which combines block-level and bit-level repair to allow a repair of small blocks with fewer redundancy allocations overhead. The results show that, even under high defect densities of $10^{-3}$, a yield above 90 percent is possible. The overhead is approximately 70 percent.

Other regular structures such as a programmable logic array (PLA) and arithmetic modules have been also extended for built-in self-repair. A PLA BISR and a comparison between spare usage and duplication is performed in (6). It is shown that spare usage is beneficial to large PLAs and that duplication fits well for small PLAs. To increase the yield of PLAs, spare wires have been used (21). They use an M-choose-N sparing to cope with production defects. The numbers of necessary redundant wires are calculated with probability calculations. For an initial probability of 90 percent that the wire is fault-free, nearly 50 percent of the wires have

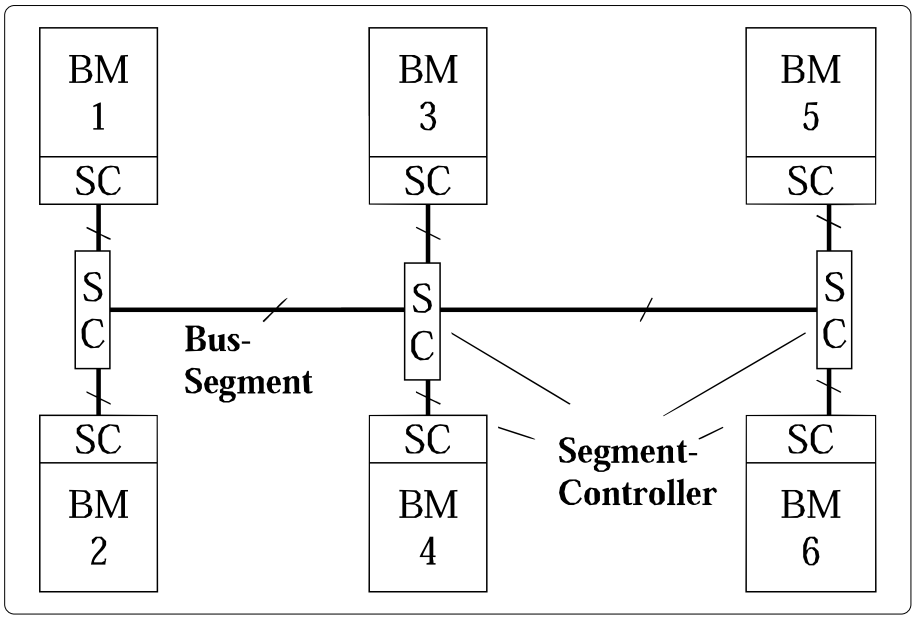


Figure 2.11: Global interconnection with several segments, each with built-in self-repair circuits (30)

to be redundant to achieve a yield of 0.999. The BISR of multiply accumulate cells (MACs) within a FIR filter is described in (14). Above 97 percent of the single stuck-at faults could be repaired at a cost of 33 percent logic overhead.

Only one publication was found on built-in self-repair for interconnects (30). This study depicts the following architecture, which I will discuss more in detail because the research is extended in this thesis. The basic concept is to use spare wires and additional circuits containing switches to change the wire utilization. The global interconnection is divided into several segments (figure 2.11). Each segment consists of wires and built-in self-repair circuits called segment couplers or Segment Controllers. A Segment Coupler consists of switches, memory to save the internal states, decoders, and configuration logic. The structure of a pair of segment couplers is depicted in figure 2.12. The switches are used to change the wire utilization. Unused wires are used as spare wires. The switches are arranged in such a way that, in every switch state, every wire has new neighbors. This structure causes crosstalk avoidance through capacitive balancing, which is combined with built-in self-repair. Capacitive balancing works in the following way. The interconnection is divided into several segments, and thereby, the capacitors between adjacent wires are also divided. Each segment can change the wire utilization, which can be used to

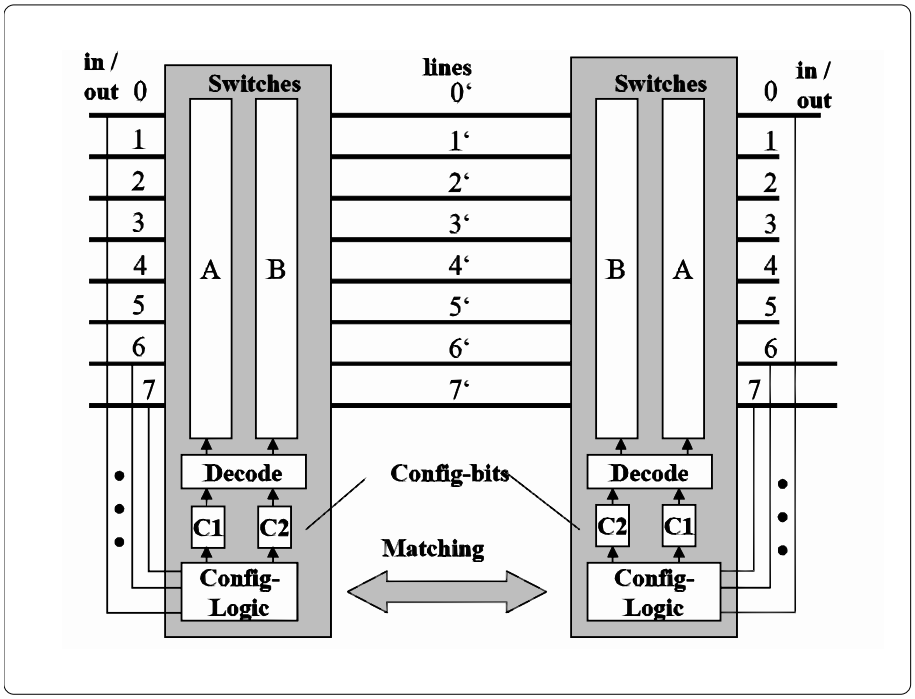


Figure 2.12: Structure of a pair of Segment Couplers (30)

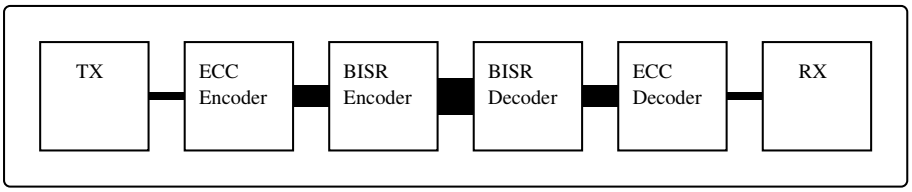


Figure 2.13: Combination of ECC and built-in self-repair

balance the segment capacitors. Error correction and error detection codes can be combined with the proposed built-in self-repair scheme, depicted in figure 2.13. The interconnection is encoded first, and the encoded interconnection can be repaired using BISR. The test of the interconnection is performed using a special purpose processor, called test processor and bus coupler or Busreflector. The bus coupler is able to link two independent buses; for example, a unidirectional address bus can be linked with a bidirectional data bus. This linkage is necessary to test the address bus by sending a test pattern. The pattern is inverted by the Bus Coupler and is transmitted back through the data bus. With the inversion of the test pattern, all stuck-at faults can be identified and located (29; 32). If the interconnection is bidirectional, then Busreflectors are used instead of bus couplers. A possible application is depicted in figure 2.14. Figure 2.14 shows a system of three bus masters

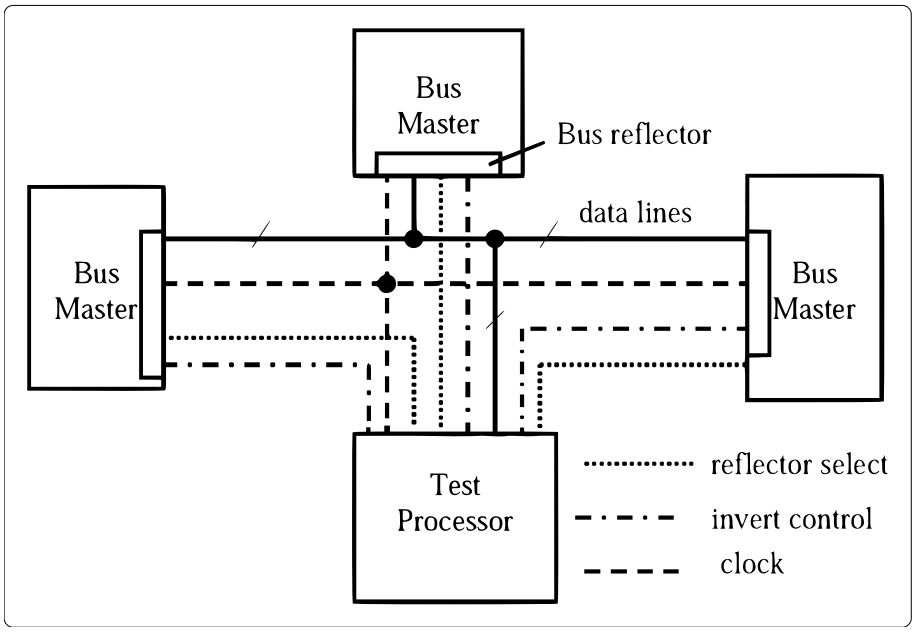


Figure 2.14: Bus system with Test Processor and Busreflector (30)

communicating over one bus. The test processors test the interconnection to the bus masters step by step. Therefore, they activate the corresponding Busreflector, send a test pattern and receive the inverted pattern. If a fault is detected, then built-in self-repair takes place, which is not depicted in this figure. Therefore, the test processor activates the segment couplers of the faulty segment and reconfigures the switch state until the fault can be corrected.

# Chapter 3

# PROBLEM DEFINITION

In the previous chapter, I have shown that there are many different solutions for increasing the reliability of interconnections. This chapter has the aim of defining the research goal. Therefore, the general requirements for fault-tolerant interconnections are discussed, to be able to evaluate the different solutions. As a prerequisite for the evaluation, the reliability model for the original and fault-tolerant interconnection is derived. Subsequently, the existing solutions are discussed, to show why there is still a necessity to research a reliable interconnection. Finally, the evaluation results are summarized and the research goal is described.

## 3.1 Requirements for fault-tolerant interconnections

Looking at the trend of interconnection implementations and the existing solutions it is predicted, that static faults become more likely, aside from dynamic and transient faults. The total number of faults will increase and the mean time to failure will decrease. The yield decreases and the number of latent faults, which lead to early-life failures, increases. Besides the reliability issues, new architectures such as network-on-chip and globally asynchronous locally synchronous arise. There is a large variety of interconnection topologies and implementations. The ideal solution for interconnection reliability has to have the following attributes:

- Correcting all of the expected faults for high reliability,
- Universally usable to work with all interconnection architectures,

- No additional wires,
- No additional delay, and
- No additional power

Correcting all of the expected faults means that all of the types and all of the numbers of faults have to be corrected during the lifetime of the device. It would be optimal to define how many faults are expected, and the cad software would take care of the remainder of the task. Therefore, it must be possible to automatically insert circuits to ensure the required reliability. These circuits have to be compatible with the existing interconnection architectures. An ideal solution would support every type of clocking, physical implementation and topology. The additional overhead (area, delay, power) should be zero. In a real system, this scenario is impossible, which is why the overhead should be minimal.

## 3.2 Reliability model

The purpose of reliability modeling is to evaluate the benefit of the different fault-tolerant solutions. First, the reliability of an interconnection has to be modeled. Several parameters influence the reliability. For an adequate model, how the reliability is influenced must be determined, and which influence has the highest impact must also be found. These tasks are described in section 3.2.1. Then, section 3.2.2 describes the reliability model for the fault-tolerant interconnections.

### 3.2.1 Interconnection reliability

The reliability of interconnections can be modeled using the fault rate and the number of wires. The fault rate depends on various factors, which can be divided into three classes: layout parameters, interconnection materials, and operation conditions. These classes can be divided into subclasses such as those depicted in figure 3.1.

*Layout parameters* describe the shape, position and orientation of every single wire and the number and shape of the used vias. The shape of a simple wire is characterized by the wire length, height and width. The width is the most critical parameter because there is a trade-of between size and reliability. The smaller the

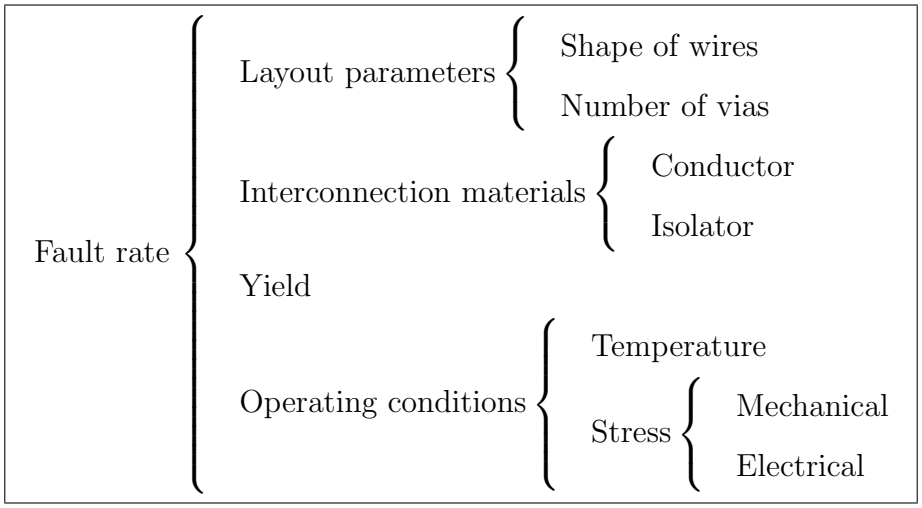


Figure 3.1: Fault-rate influencing factors

width, the more wires can be integrated and the fewer metal layers are required for all of the interconnections. The smaller the width is, the higher the impact is of the defects that are caused by particles during manufacturing. A particle during the lithographic process can lead to a wire narrowing. The narrowing decreases the cross-section, which leads to a locally increased current density. A higher current density leads, for example, to a reduced mean time to failure with respect to electro-migration. The longer the wire is, the higher the probability is that the wire contains at least one narrowing. To limit the current density and to compensate for the decrease in the width, the height is increased. This adjustment leads to an increase in the coupling capacitance between adjacent wires. The higher the capacitance is, the higher the impact is of the crosstalk. Vias are reliability critical (50). The higher the number of vias on a wire, the higher the probability that the wire is faulty, assuming that every via has the same constant probability of failure. This relationship occurs because of the difficulties during manufacturing. Two or more layers that have to be connected have to be aligned properly. When the cross section of the via is smaller, the alignment becomes worse. As a result, the current density tends to be highest in the vias. The higher the current density J is, the higher the electro-migration and the less the mean time to failure (17), as shown in equation 3.1. Parameter $A$ depends on the interconnect geometry and material, and the exponent $n$ lies between 1 and 2 according to the actual failure mechanism. The parameter $\phi$

is the activation energy.

$$MTTF_{EM} = \frac{e^{(\frac{\phi}{kT})}}{A \cdot J^2} \quad \text{from (17)} \tag{3.1}$$

The *interconnection materials* influence the reliability in the following way. Different materials are used for the conductors and isolators. Conductive materials such as aluminum and copper are used for the wires. Copper has a higher conductivity and allows a higher current density with respect to electro-migration. The disadvantage of this construct is that there is a more complex fabrication required because of the additional barrier layer. This layer is a diffusion barrier between copper and silicon dioxide ($Si_2$). For the inter-layer dielectric (ILD), several materials are used, which have low dielectric constants (called low-k dielectrics). The inter-layer dielectric has the task of insulating adjacent wires and providing mechanical stability. There are several problems that arise from the material choice. Time-dependent dielectric break down (TDDB) and mechanical stress due to the different thermal expansion coefficients of the materials lead to static faults.

*The yield* describes how reliable the manufacturing is. Reference (12) shows that it is legal to assume that a certain proportion of the defects are latent. Thus, you can predict the early lifetime fault rate from yield measurements. The worse the yield is, the more in-field faults can be expected.

*Operational conditions* with the highest reliability impact are temperature, mechanical, and electrical stress. The temperature has the largest impact on the interconnection reliability with respect to electro-migration. Temperature has an exponential impact on the mean time to failure, as can be seen in Equation 3.1. The temperature depends on the technology that is used and on the actual design. For electro-migration, the maximal temperature is critical. For stressinduced voiding, the temperature cycles are critical. Stress-induced voiding (50), describes the metal atom transport through mechanical stress, which results from different thermal expansion coefficients of adjacent materials. The thermal cycles lead to interconnection fatigue and are a result of power saving techniques and varying workloads.

The general reliability of a system that is only in an acceptable state as long as all of its $N$ subsystems are in an acceptable state is called a series system. A series system's reliability results from the product of all of the subsystem reliabilities. An interconnection is a series system of $N$ wires. Thus, the interconnection reliability $R_{icon}$ can be modeled as following, assuming that all of the wires have the same

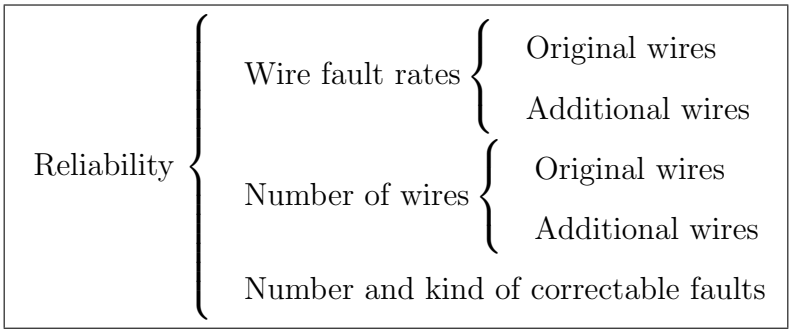


Figure 3.2: Reliability influencing factors of a fault-tolerant interconnection

reliability $R_{wire}$:

$$R_{series} = \prod_{i=1}^{N} R_i \tag{3.2}$$

$$R_{icon} = R_{wire}^{N} \quad \text{if all wires have the same reliability} \tag{3.3}$$

### 3.2.2 Fault-tolerant interconnection reliability

The reliability of a fault-tolerant interconnection depends on the numbers of wires, the wire fault rates, and the number and type of correctable faults (figure 3.2).

The *wire fault* rates depend on the parameters that are depicted in figure 3.1 and can be different for the original and the additional wires, which are used only for fault-tolerant implementations. If additional wires are used with the same intensity as the original signal wires, the fault rates can be equal. This scenario occurs when using the Hamming code, for example. The second case, where the fault rates differ, is relevant to built-in self-repair using cold spares. There is no current flowing through the spares, which is why no electro-migration takes place until the spare is used for repair.

The *numbers of* wires is the number of original and additional wires. The number of original wires is the number of wires that are necessary to transmit the data without any fault-tolerance. The number of additional wires counts the wires that are necessary to implement fault-tolerance. If the fault rates are equal for the additional and the original wires then it is legal to sum up both numbers. The *number and type of correctable faults* describe how many static faults can be repaired or how many transient faults can be corrected simultaneously. If one or more static or transient faults are allowed because of existing redundancies (codes or repair), the reliability

can be modeled using a k-out-of-n system.

$$R_{k/n} = \sum_{i=k}^{n} \binom{n}{i} R_{sub}^{i} [1 - R_{sub}]^{n-i} \tag{3.4}$$

For a 32-bit width interconnection with one spare, the reliability can modeled as follows:

$$\begin{aligned} R_{32+1} &= R_{k/n} \quad \text{with} \begin{cases} k = 32, n = 33 & \text{if spare has same failure probability} \\ k = 31, n = 32 & \text{if spare has zero failure probability} \end{cases} \\ R_{32+1} &= \begin{cases} 33R^{32} - 32R^{33} & \text{if spare has same failure probability} \\ 32R^{31} - 31R^{32} & \text{if spare has zero failure probability} \end{cases} \end{aligned} \tag{3.5}$$

The first part of equation 3.5 assumes that the spare has the same probability to fail. This assumption can be true for wires, which have the same workload, or for faults, which also effect inactive wires such as in mechanical stress. Considering electro-migration, this assumption is not accurate because electro-migration affects only live wires. The second part models this behavior. The reliability of both cases and the reliability of the 32-bit width bus are depicted in figure 3.3 as a function of the wire reliability. The two cases can be seen as reliability bounds. The real reliability lies between the two bounds. The bounds get closer for wider interconnects. In figure 3.4, the resulting interconnection reliability is plotted as a function of the original interconnection reliability. The difference between the case with zero or equal spare failure probability is so small that the curves overlap. For simplification, I will discuss only the case that the spare has the same probability for failure for the remainder of this paper.

With equation 3.4 above, it is possible to model the resulting interconnection reliability depending on the wire reliability. To predict the lifetime reliability, equation 3.4 must be extended with a reliability function. There are different reliability functions for each part of the bath tube curve. For a constant fault rate (CFR), the exponential function is suitable. For all mean times to failure calculations that are performed in this thesis, this reliability function is used.

$$R(t) = e^{-\lambda \cdot t} \tag{3.6}$$

Equation 3.4 together with equation 3.6 lead to equation 3.7.

$$R_{int}(t) = \sum_{i=k}^{n} \binom{n}{i} e^{-\lambda t i} [1 - e^{-\lambda t}]^{n-i} \tag{3.7}$$

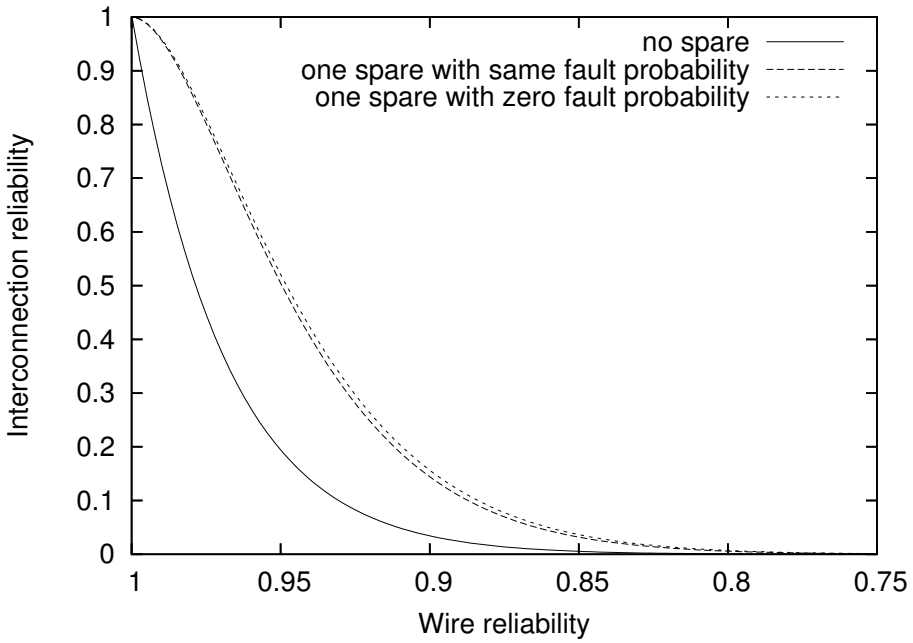


Figure 3.3: Reliability of a 32 bit interconnection for the cases of no spare, of one spare with equal failure probability, and one spare with zero failure probability dependent on the wire failure probability.

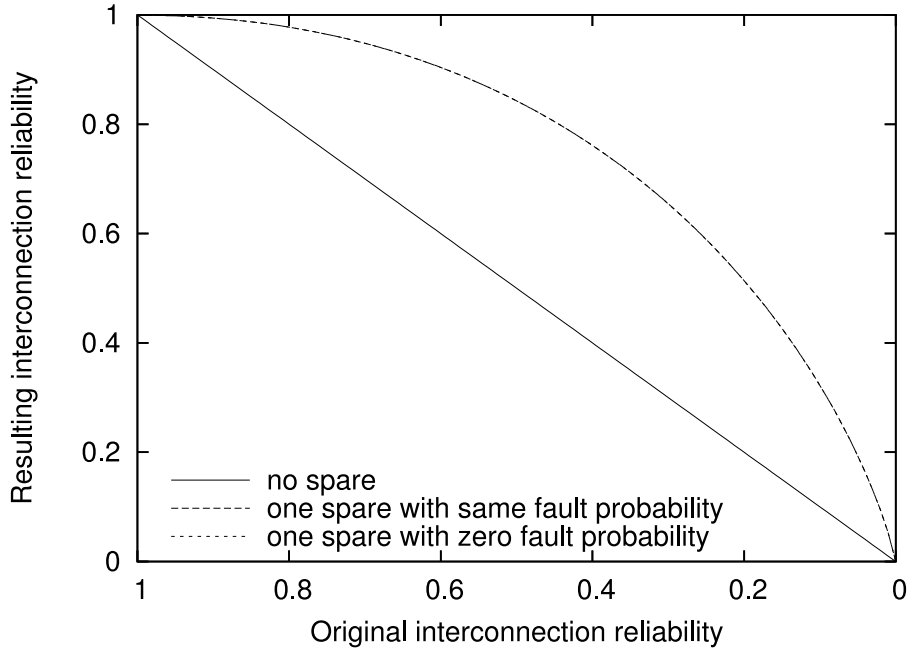


Figure 3.4: Interconnection reliability for the case of no spare, of one spare with equal failure probability and for the case of one spare with zero failure probability dependent on the original 32 bit-width interconnection failure probability.

With this equation, the mean time to failure (MTTF) can be modeled as follows:

$$\begin{aligned} MTTF &= \int_{t=0}^{\infty} R_{int}(t)\,\mathrm{d}t \\ &= \int_{t=0}^{\infty} \sum_{i=k}^{n} \binom{n}{i} e^{-\lambda t i}[1-e^{-\lambda t}]^{n-i}\,\mathrm{d}t \end{aligned} \tag{3.8}$$

For the original 32-bit width interconnection and for the case with one spare, the MTTFs look like the following two equations:

$$MTTF_{32} = \frac{1}{32\lambda} \tag{3.9}$$

$$MTTF_{32+1} = \frac{65}{1065\lambda} \tag{3.10}$$

$$LF_{32+1} = \frac{MTTF_{32+1}}{MTTF_{32}} = \frac{65}{33} \tag{3.11}$$

The quotient of the fault-tolerant interconnection MTTF and the MTTF of the original interconnection is called the lifetime factor (equation 3.11). A 32-bit width interconnection with one spare would lead to a lifetime factor of nearly 1.97.

## 3.3 Discussion of existing solutions

The first step towards obtaining reliable systems is having a reliable technology. This scenario includes the mastery of materials and their processing. Using better materials such as carbon nanotubes for wires has a larger influence than the choice of architectural solution. Therefore, the technology should be optimized first. If it is still not possible to ensure the required reliability, then architectural, code-based, and built-in self-repair-based solutions should be applied. In the following, some of the solutions that have been described in chapter 2 are described in more detail, to show their limits.

### 3.3.1 Wire widening

The interconnection layout is often adapted to a certain technology. The width of the wires is chosen, depending on the estimated current density and the intended lifetime. This step is performed with some additional factors of safety, which is necessary because manufacturing is not faultless. According to the reliability, this choice is not the best solution for any case. In figure 3.5, wire widening is compared

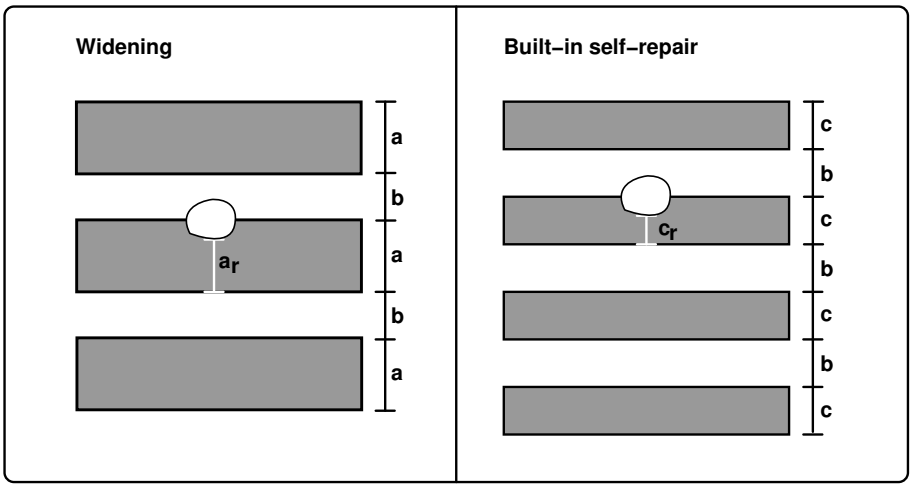


Figure 3.5: Wire widening versus built-in self-repair

with built-in self-repair for when one spare wire is used with the same footprint. For an interconnection with $n$ wires with an equal length $l$, the footprints $A_{Wide}$ and $A_{BISR}$ are calculated as follows:

$$A_{WIDE} = \big(n \cdot a + (n-1) \cdot b\big) \cdot l \geq A_{BISR} = \big((n+1) \cdot c + n \cdot b\big) \cdot l \quad . \tag{3.12}$$

The wire width in the case of widening is called $a$ and, in the case of built-in self-repair (BISR), $c$. The distance $b$ between adjacent wires is equal for both cases. The maximal wire width $c$ using BISR is expected to have an equal footprint compared to wire widening, which can be derived using equation 3.12. With the same wire length $l$ and the same space between adjacent wires $b$, the BISR wire width is as follows:

$$c \leq \frac{n \cdot a - b}{n+1} \quad . \tag{3.13}$$

In the case of faultless manufacturing, the widening of wires leads to a higher lifetime because the current density is smaller than the density using BISR. The wires of built-in self-repair would age faster. The first fault could be repaired. However, it is highly probable that another of the remaining aged wires will fail. The advantage of built-in self-repair decreases with an increasing fault-rate, which arises during aging.

In the case of imperfect manufacturing, built-in self-repair can lead to a higher lifetime and improved reliability. Assume that there is a defect with equal size in both cases. This assumption leads to the reduced widths $a_r$ and $c_r$. The smaller width $c_r$ leads to the highest current density and has the highest probability to fail. If it fails, it can be repaired using BISR. If the widened wire with the defect and the

width $a_r$ is smaller than the faultless wire using BISR with width $c$, then it will fail earlier. It cannot be repaired. Thus, built-in self-repair can contribute to a longer lifetime.

### 3.3.2 Refueling

Refueling works well when the wire temperature is equal over the whole length. If the temperature is equal, then equalizing the amount of current in both directions leads to self-repair. This result is possible because the energy required for the transport of metal atoms is equal over the whole wire. Electro-migration depends exponentially on the temperature. Thus, the higher the temperature is, the higher the mass transport. The effect of self-repair decreases with increasing temperature inhomogeneity. If there are hot spots on a wire, then self-repair will not work as well as is described in (2) and refueling will not lead to a lifetime extension of a factor up to $10^4$. A local defect resulting in a narrowed wire leads to a locally increased current density, which can lead to a locally increased temperature. The rise in current density and in temperature accelerates the electro-migration, which further leads to self-heating and ends up in a broken wire. The resulting error must be corrected using codes or built-in self-repair, or the circuit fails.

### 3.3.3 EDC and ECC

Error detection codes and error correction codes can correct transient faults. An error detection code is often combined with retransmission for the correction. Retransmission is not suitable for static faults because the fault is permanent. Thus, only the error correction code (ECC) can correct static faults. Compared to built-in self-repair, error correction codes have one disadvantage. These codes require more additional wires. This requirement leads to a lower reliability, as shown in the following:

Both the code and the BISR reliability can be modeled using a k-out-of-n system (equation 3.4). Only the number of wires and spare wires differ. Assume that we have a 32-bit interconnection, which we want to protect against a single static fault. We could use the Hamming code, which needs six additional wires and a total of 38 wires. The BISR scheme requires one wire as a spare and three wires for administration. Information about the implementation will follow in chapter 4.

Thus, BISR uses 36 wires for a 32-bit interconnection. The possibility that one of the 36 wires is faulty is smaller than one out of 38. For this reason, the reliability concerning static faults is higher when using built-in self-repair, even with pessimistic assumptions. The three wires used for administration are only used during testing, which takes place less frequently than the normal signal transmission. Furthermore, the spare wire is not used until the first fault. Electro-migration will not occur until it is used. Thus, BISR leads to an even higher reliability for permanent faults. With these optimistic assumptions, the mean time to failure can be increased by a factor of 2.0 using BISR. Using the Hamming code leads to a factor of 1.4.

### 3.3.4 Alternate Data Retry

Alternate Data Retry (ADR) in combination with an error detecting code allows for compensation of transient and permanent faults. When a fault has been detected, the inverted data are transmitted to compensate for the fault. This action has some disadvantages. The most important disadvantage is the fact that the bandwidth decreases by 50 percent when a permanent fault occurs. For example, a stuck-at-one fault would cause a retransmission of a zero. Thus, to transmit a one and a zero, one needs three instead of two cycles. If zeros and ones are distributed equally, then there is a 50 percent reduction in bandwidth. Shedletsky (58) has proposed the principles of ADR and an implementation for a fault-tolerant data path. The application of Alternate Data Retry for a fault-tolerant interconnection has not been published. For this reason, I propose an implementation that can evaluate the ADR code. Bandwidth reduction can be prevented if the faulty wire can be diagnosed. If the faulty wire is known, then the inversion of the data could be done and would be dependent on the wire value. Additionally, the information, whether it is the original or the inverted data, has to be transmitted. The resulting system is depicted in figure 3.6. The drawback of retransmitting in case of a static fault can be solved. However, this step comes with an additional signal delay during a normal operation. Thus, either possibility has a 50 percent decrease in bandwidth from the time that a permanent fault has occurred, or there is always an additional signal delay due to the multiplexers.

Another drawback is that ADR cannot isolate the fault. A wire that is grounded could lead to higher currents compared to the original wire. Higher currents mean a higher temperature and a larger amount of aging. What happens if it is not a clear

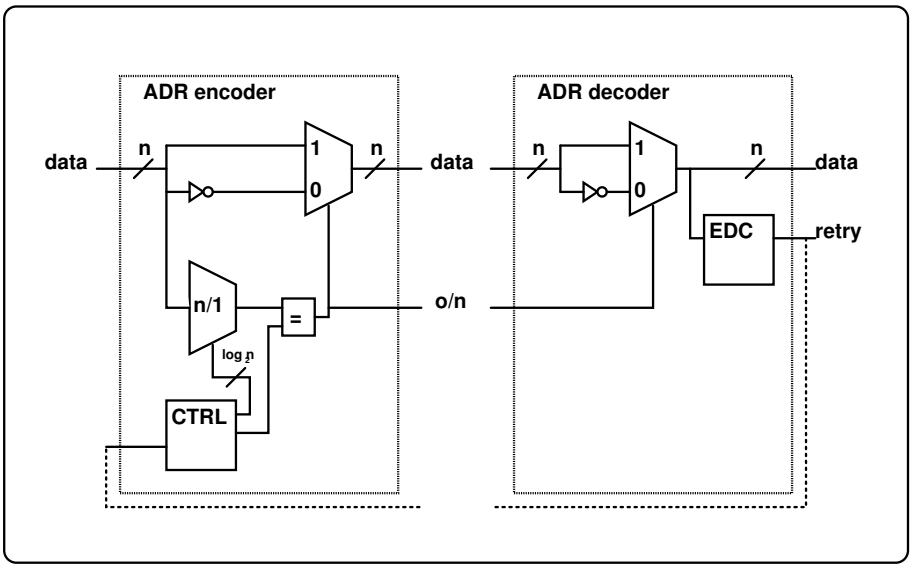


Figure 3.6: Stand-alone alternate-data retry system to ensure bandwidth

stuck-at-one or stuck-at-zero fault? The fault could be a resistive short leading to an undefined value. The literature could not be found on how the ADR would react in this case. Thus, additional research is required. Furthermore, incompatibility with crosstalk avoidance codes is an additional problem. The group of forbidden transition codes (FTCs) attempts to avoid transitions that lead to high signal delays. The worst case transition for a three-bit-width interconnection would be “010” to “101”, and vice versa. The value of the middle wire would be delayed. Normally, an FTC would prevent this transition, but an inverted pattern through an ADR would directly lead to the highest possible delay, which could lead to dynamic faults.

### 3.3.5 Fault-tolerant communication architectures

Fault-tolerant communication architectures enable fault-tolerance on a higher level than coding and built-in self-repair. A higher level means a decreased granularity. On this level, the interconnection is considered as a link, which can be faulty or not. Whether or not a wire is faulty is not considered. Only the whole interconnection is considered. Because of the higher level of fault-tolerance, fault-tolerant communication architectures are compatible with coding and BISR. Thus, a fault, for example, that cannot be corrected using codes or BISR can be corrected using adaptive routing. The focus in this thesis lies on a deeper level, and therefore, fault-tolerant communication architectures are not considered in the following sections.

### 3.3.6 Built-in self-repair

The built-in self-repair of interconnects is described in (31). Testing is performed using a special purpose processor (a test processor) and additional circuits. These additions can be used for bidirectional bus-like interconnects and are compatible with error detection and error correction codes. The replacement scheme is fixed. For every seven wires, there is one spare. The scheme allows capacitive balancing when there are segment couplers used in serial. The implementation partly covers the existing interconnection topologies and implementations. It requires a tri-state implementation of the communication architecture, which is not the best solution because of the higher energy consumption (48). Mixed-based and and-or-based implementations are not supported. The replacement scheme is not compatible with crosstalk avoidance codes because the reconfiguration could lead to forbidden signal patterns and transitions, eliminating the benefits of these codes. The administration of the Busreflector and segment couplers is performed centrally and unprotected. Local administration is not implemented. Pass transistors and transmission gates are used for switching. For a 32-bit interconnection, five spares are required. Thus, more than 37 wires and additional wires for administration would be required. In comparison, 38 wires would be required when using a Hamming code, which enables immediate error correction. Using built-in self-repair, testing and repairs occur periodically or upon start-up. The BISR configuration must be saved permanently or the interconnection must be tested again after power down. From this point of view, built-in self-repair is not competitive with existing codes. However, built-in self-repair would lead to higher reliability with respect to permanent faults, and there would be more efficient spare utilization.

## 3.4 Research goal

To ensure a reliable interconnection, all of the types of faults must be addressed. A decreasing yield leads to an increasing number of latent faults, which will result in static faults in the field. This type of change leads to an increase in the static fault rate. Past solutions such as widening and re-fueling can reduce aging-based faults. In the case of imperfect manufacturing, these strategies are not the best solution because they cannot compensate for in-field faults. The static faults can be corrected using codes or built-in self-repair. BISR leads to a higher reliability

because it requires fewer additional wires, and the additional wires are not as stressed as the wires used for coding. BISR cannot correct transient faults or prevent dynamic faults similar to CACs. However, a combination of BISR and codes could enable reliable interconnections, which could address all types of faults.

---

*In this thesis, the combination of built-in self-repair and codes is implemented and evaluated according the reliability and the costs.*

---

To achieve the research goal, I have proceeded using the following two steps:

1. Extension and evaluation of the existing built-in self-repair circuits, and
2. Evaluation of selected combinations of built-in self-repair circuits and codes.

*First*, the existing BISR circuits must be developed further so that a combination of codes is possible and they are applicable to existing and upcoming interconnection implementations. The BISR circuits proposed in (31) support bus-like, bidirectional interconnections. They are administrated centrally with a test processor and an additional circuit called Busreflector. The circuits are compatible with error correction and error detection codes. However, they are incompatible with codes for dynamic fault prevention, which neglect a BISR-based interconnection with protection against dynamic faults. To reach the research goal, it is necessary to make them have the following characteristics:

- compatible with crosstalk avoidance and joint crosstalk avoidance error correction codes,
- scalable with the number of static faults and the interconnection width,
- supporting different interconnection structures, and
- supporting different clocking implementations.

After the implementation, the resulting built-in self-repair circuits must be evaluated for reliability and cost. This evaluation is necessary for deciding which codes are suitable for combinations with respect to maximum reliability.

*Second*, the combinations must be evaluated to show how expensive the achieved reliability is. I have evaluated only some of the possible combinations. The selection of the combinations depend on the results of the BISR circuit evaluation. I have chosen one code from every group of codes (EDC, ECC, and CAC).

# Chapter 4

# BUILT-IN SELF-REPAIR

In this chapter, the implementation and evaluation of the built-in self-repair circuits, mainly the segment couplers, takes place. The function of the segment couplers is to change the wire utilization, which allows us to replace a faulty wire with a spare wire. The couplers consist of switches, memory to save the switch configuration, and a state machine for communication and reconfiguration. The implementation of the switches is crucial for the compatibility with crosstalk avoidance codes and will be described in the following section. The structure of the switches determines the reliability that can be achieved. The structure is discussed in the section called the segmentation scheme and describes how multiple faults can be repaired. The subsequent two sections describe the two administration schemes (central, local) and the two clocking schemes (synchronous, asynchronous). The results show that codes with a high code rate result in small BISR circuits, which require less power.

## 4.1 Switching scheme

The switching scheme describes the way that wire utilization is accomplished. In this scheme, it is possible to exchange the use of two wires or even to permute them. In general, the use of a switching scheme depends on whether or not there are constraints according to the reconfiguration.

### 4.1.1 Compatibility to crosstalk avoidance codes

In figure 4.1, the two switching schemes *bypass* and *rotate* are depicted; these two schemes are used in this thesis.

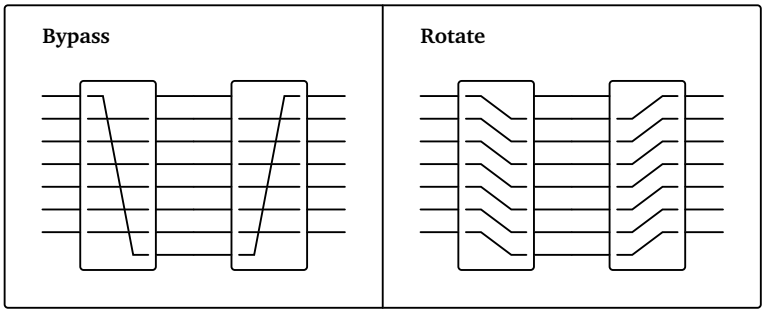


Figure 4.1: Bypass and rotate switching scheme

If no crosstalk avoidance code (CAC) is used and no capacitive balancing is required, then the wire can simply be bypassed. The switches have to be controlled in such a way that the input signal of the faulty wire is switched with the spare wire. In the segment coupler decoder, the bypassed signal is switched back to the original output.

If no CAC is used and capacitive balancing is required, then the wire can be permuted (30) or rotated. Capacitive balancing works if the interconnection is divided into two or more segments. The wires are permuted in such a way that, after permuting them, every wire has new neighbors.

If CAC is used, then the wires have to be rotated because the neighborhood of adjacent wires must be preserved. For example, in a bundle of 8 wires, suppose that one spare wire (no. 9) and a fault on wire 3 occurs, so that 3 is shifted to 4, 4 to 5, and so on. This occurrence is repeated until wire 3 is routed onto the spare wire. Rotating the wire means that, in the case of a fault, all of the wires will be routed by one position in the same order. Routing the wires rotation-like requires a higher number of switches because every single wire has to be routed onto every other wire. Thus, the complexity is of quadratic order $O(n^2)$ whereas the complexity of signal bypassing is linear. Bypass replacement is possible when the spare wire is isolated from the adjacent wires using shield wires. The interconnection segmentation schemes are described in the subsequent section.

### 4.1.2 Cost comparison

To obtain area, power, and delay information, I have created generic VHDL models of the built-in self-repair and coding circuits. These models have been synthesized using the *Cadence RTL Compiler* and the 180$\mu$m library that comes with the pro-

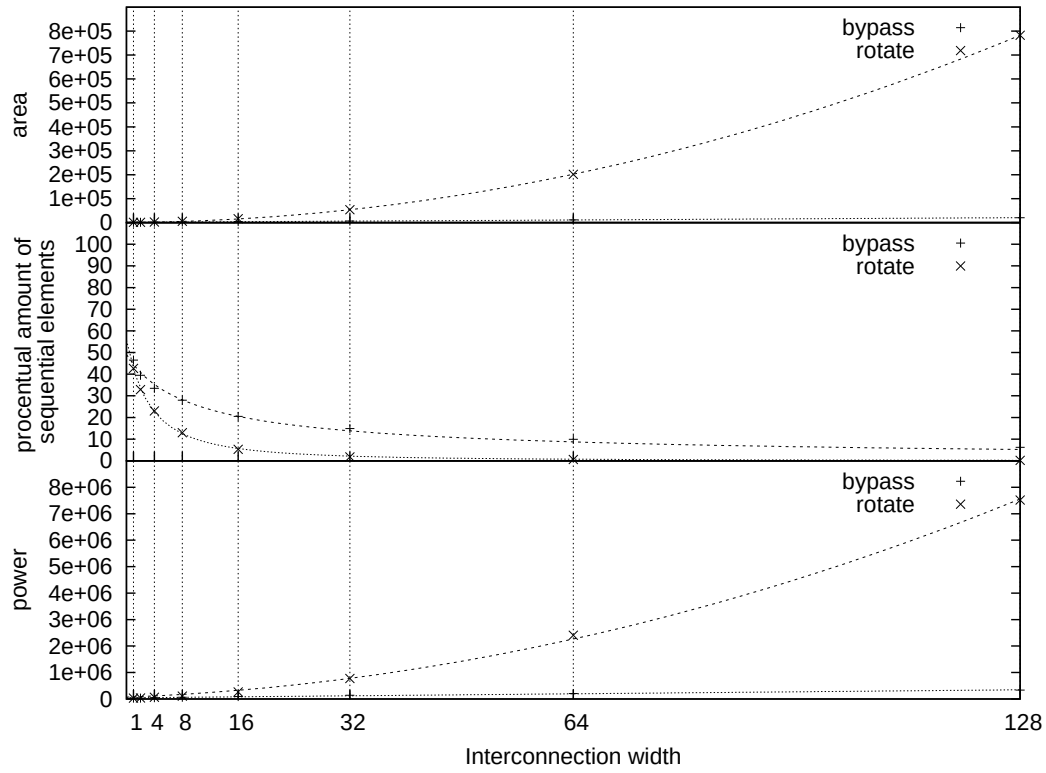


Figure 4.2: Area consumption of bypass or rotate reconfiguration

gram. In other words, every data point in the following diagrams required one synthesis or calculation. The synthesis results depend mainly on the library used. Thus, absolute values are less important than relative ones. I will always use relative values so that it is easier to obtain general statements. With relative values, it is easier to estimate what the results would be using another library. For this reason, this library is well-suited for the task, and there is no need for other libraries.

The switches can be uni- or bidirectional. For bidirectional switches, pass transistors or transmission gates can be used. The same scenario is valid for unidirectional transmission, but additionally tri-state gates can be used. In the following, unidirectional switches using tri-state gates have been used. Other switches have not been supported by the library, and from the results obtained it is possible to estimate the area required when using pass transistors or transmission gates.

The relationship between area, power, the number of sequential elements and the interconnection width is depicted in figure 4.2. For each data set, a curve has been

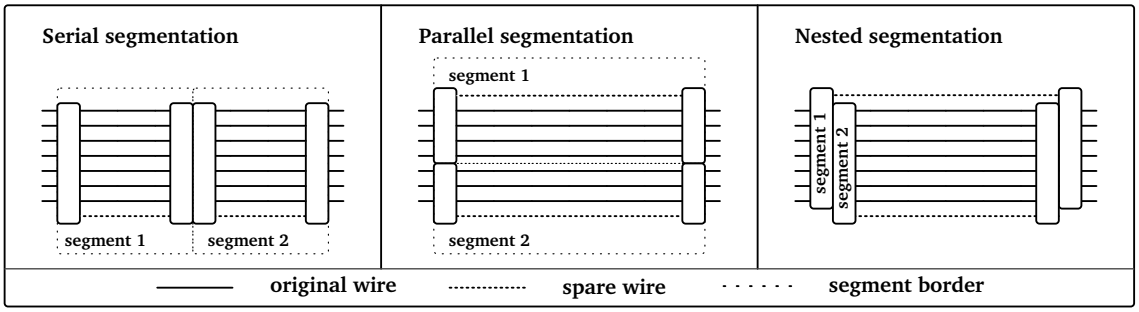


Figure 4.3: Possibilities to repair more than one fault

fitted. The synthesis results confirm that the bypass reconfiguration has a linear behavior, and the rotate reconfiguration has a quadratic behavior, according to the area consumption. The behavior is the same for power consumption. The percent of sequential elements decreases with increasing interconnection width. There are sequential elements because of the counters, which save the reconfiguration state. The counter grows with a complexity of $log_2(N)$. The number of switches grows either linearly (bypass) or quadratically (rotate). The percent of sequential elements decreases because the area of the switches increases faster than the area of the counters. The percent of the curve that is fitted for bypass is $A_{se} = \frac{\log_2(N+a)}{b*(N+c)}$ and the curve fitted for rotate is $A_{se} = \frac{\log_2(N+d)}{e*(N+f)^2}$ with $a, b, c, d, e$ and $f$ as constant fitting parameters. The functions fit well and confirm the above explanation.

## 4.2 Segmentation scheme

To repair multiple faults, there are three segmentation schemes, which are depicted in figure 4.3. These schemes are called serial, parallel and nested segmentation.

*Serial segmentation* means that there are multiple segments along the interconnection. In every segment, one fault can be repaired. The reliability of a segment depends on the segment length. The longest segment is the most unreliable. For maximum reliability, the lengths of the segments should be equal. In addition to the repair of multiple faults, serial segmentation allows capacitive balancing to decrease the effect of capacitive coupling.

*Parallel segmentation* describes the division across the interconnection. Again, one fault per segment can be repaired. The segment with the highest wire count is the most unreliable. If one segment fails, then the whole interconnection is faulty.

Thus, the parallel segments should have equal size for maximum reliability. Parallel segmentation is compatible with crosstalk avoidance codes when there is a shielding wire between adjacent segments.

*Nested segmentation* describes the structure of switch layers when the switches overlap. This structure is the only scheme that enables the repair of all of the multiple faults. In contrast, when there is more than one fault per segment using serial or parallel segments, the interconnection fails. Nested segmentation is compatible with crosstalk avoidance codes, assuming that the spares are shielded with respect to adjacent wires.

The reliability of the segmentation schemes is described in the following subsections.

### 4.2.1 Serial segmentation

The reliability of the serial segmentation can be modeled as a serial system of $S$ times a k-out-of-n system. Under the assumption that the serial segments have the same reliability, the reliability $R_{serial}$ depends on the segment reliability $R_{seg}$, as follows.

$$R_{serial} = (R_{seg})^S \tag{4.1}$$

The segment reliability $R_{seg}$ depends on the reliability of the wires. Through serial segmentation, the wires are divided into several segments. Thus, the segment reliability depends on the reliability of the wire segment $R_{sow}$. There is always one spare per serial segment, which is why $n = N + 1$ and $k = n - 1$.

$$R_{seg} = R_{k/n} = (N+1)(R_{sow})^{(N)} - (N)(R_{sow})^{(N+1)} \tag{4.2}$$

The reliability of the wire segments $R_{sow}$ depends on the reliability of the whole wire $R_w$ and the number of segments $S$.

$$R_w = R_{sow}^S \tag{4.3}$$

The wire reliability $R_w$ can be calculated using the original interconnection reliability $R_{oi}$ and the interconnection width $N$.

$$R_{oi} = R_w^N \tag{4.4}$$

The wire segment reliability can be calculated using equation 4.3 and equation 4.4.

$$R_{sow} = (R_w)^{\frac{1}{S}} = (R_{oi})^{\frac{1}{NS}} \tag{4.5}$$

The resulting reliability of the serial segmentation $R_{serial}$ can be calculated using equation 4.1, equation 4.2 and equation 4.5.

$$R_{serial} = \left((N+1)(R_{oi})^{\frac{1}{S}} - N(R_{oi})^{\frac{N+1}{NS}}\right)^{S} \tag{4.6}$$

### 4.2.2 Parallel segmentation

The reliability of parallel segmentation can be modeled similar to serial segmentation reliability. However, there are some differences. First, the segments do not have to have the same reliability because the interconnection is divided across. If the total number of wires $N$ is not a whole-number factor of the number of segments $S$, then the numbers of wires per segment differ. With the same reliability for every wire, the segment reliabilities differ also. The parallel segmentation reliability $R_{parallel}$ is the product of the segment reliabilities $R_{seg}$.

$$R_{parallel} = \prod_{i=1}^{S} R_{seg_i} \tag{4.7}$$

The reliability of a serial system is influenced mainly by the most unreliable subsystem. Thus, to get the highest reliability, the difference between the subsystem reliabilities should be small. For parallel segmentation, where the interconnection is divided across, the difference depends on how the wires are allocated. The allocation should always be performed so that the maximal difference is one wire. For example, a 32-bit-width interconnection divided into three parallel segments should be divided into two 11-bit and one 10-bit segment. Thus, there are only two different numbers of wires per segment $n_1$ and $n_2$. As a result, there are also only two different segment reliabilities $R_{n_1}$ and $R_{n_2}$. The number of spares $S$ is the sum of all of the segments because there is always one spare per segment. There are, at a maximum, two types of segments, which have different numbers of wires. The number of each segment is denoted by $s_1$ and $s_2$, with $S = s_1 + s_2$. The parallel segmentation reliability can be calculated as follows:.

$$R_{parallel} = R_{n_1}^{s_1} \cdot R_{n_2}^{s_2} \tag{4.8}$$

Because only one fault per segment can be repaired, the reliabilities of the segments $R_{n1}$ and $R_{n2}$ can be modeled as k-out-of-n systems with $n-k = 1$ (see equation 4.2).

This construct leads to the following equation:

$$R_{parallel} = \left(n_1(R_{oi})^{\frac{n_1-1}{N}} - (n_1-1)(R_{oi})^{\frac{n_1}{N}}\right)^{s_1} \cdot \left(n_2(R_{oi})^{\frac{n_2-1}{N}} - (n_2-1)(R_{oi})^{\frac{n_2}{N}}\right)^{s_2} \tag{4.9}$$

The parameters $s_1$, $s_2$, $n_1$, and $n_2$ can be determined using the following algorithm 1. The basic concept in this algorithm is to subtract the integer part $G$ of the quotient of the number of original wires $N$ and the number of segments $S$ from $N$ until the remainder is zero or the remainder is a whole-number factor of $G+1$.

$$N = s_1 G + s_2(G+1) \quad \text{with} \quad S = s_1 + s_2 \tag{4.10}$$

**Algorithm 1** Divide nearly equally

**Require:** $S \neq 0$
  $Z = N$
  $G = \left\lfloor \frac{N}{S} \right\rfloor$ with $G \in \mathbb{N}$
  $i = 0$
  **if** ($N \mod S = 0$) **then**
    $i = S$
  **else**
    **while** ($Z \neq 0$) **do**
      **if** ($Z \mod (G+1) \neq 0$) **then**
        $Z = Z - G$
        $i = i + 1$
      **else**
        break
      **end if**
    **end while**
  **end if**
  $n_1 = G + 1$ and $n_2 = G + 2$
  $s_1 = i$ and $s_2 = \frac{N - G s_1}{G+1}$
  **return** $n_1, n_2, s_1, s_2$

### 4.2.3 Nested segmentation

Nested segmentation allows us to repair every fault until the spares are depleted. The resulting reliability $R_{nested}$ can be calculated using a single k-out-of-n system and the relationship between the original interconnection reliability $R_{oi}$ and the wire reliability $R_w$.

$$R_{nested} = \sum_{i=N}^{N+S} \binom{N+S}{i} (R_{oi})^{\frac{i}{N}} (1 - (R_{oi})^{\frac{i}{N}})^{N+S-i} \quad \text{with } R_w = (R_{oi})^{\frac{1}{N}} \tag{4.11}$$

### 4.2.4 Reliability comparison

To compare the reliability of serial, parallel, and nested segmentations, the reliabilities are depicted in figure 4.4 , 4.5 and 4.6. In figure 4.4, the resulting reliability uses built-in self-repair, and two spares are depicted. The figure shows the equations 4.6, 4.9 and 4.11 with $N = 64$ and $S = 2$. This figure shows that serial and parallel segmentations lead to nearly the same reliability, which is smaller than the reliability using nested segmentations. The embedded figure shows the upper-right curves in detail. Again, nested segmentation leads to a much higher reliability. Nested segmentation is superior to serial and parallel segmentation because every fault can be repaired until all of the spares are used. Serial and parallel segmentation have different behaviors. Only one fault per segment can be repaired. Using two spares, two faults can be repaired. For serial and parallel segmentation, these two faults must occur in two segments. If they were to occur in one segment, the repair would fail. Using nested segmentation, every double fault can be repaired.

To highlight the differences between serial, parallel, and nested segmentation, the necessary reliability of the original interconnection to achieve a reliability of 0.95, 0.99, and 0.999999 using different numbers of spares is depicted in figure 4.5. Therefore, the resulting reliability $R$ was set to 0.95, 0.99 or 0.999999, and the equations 4.6, 4.9 and 4.11 were solved numerically to find the minimal reliability of the original interconnection. The higher the target reliability is, the smaller the reliability increase through serial and parallel segmentation. In other words, using nested segmentation allows one to use much more unreliable interconnections to achieve the same reliability.

The mean time to failure is defined as the integral of the reliability from zero to infinity over time. Using the exponential fault distribution $R = e^{-\lambda t}$ while integrat-

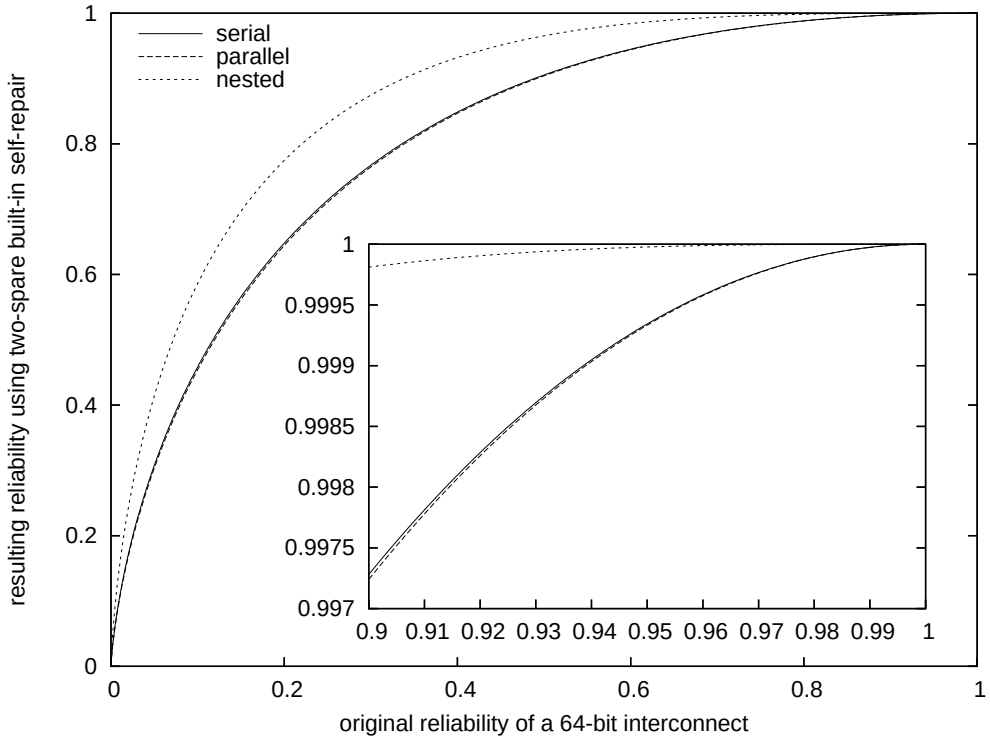


Figure 4.4: Achievable reliability of a 64-bit interconnection using two spares and different segmentation schemes

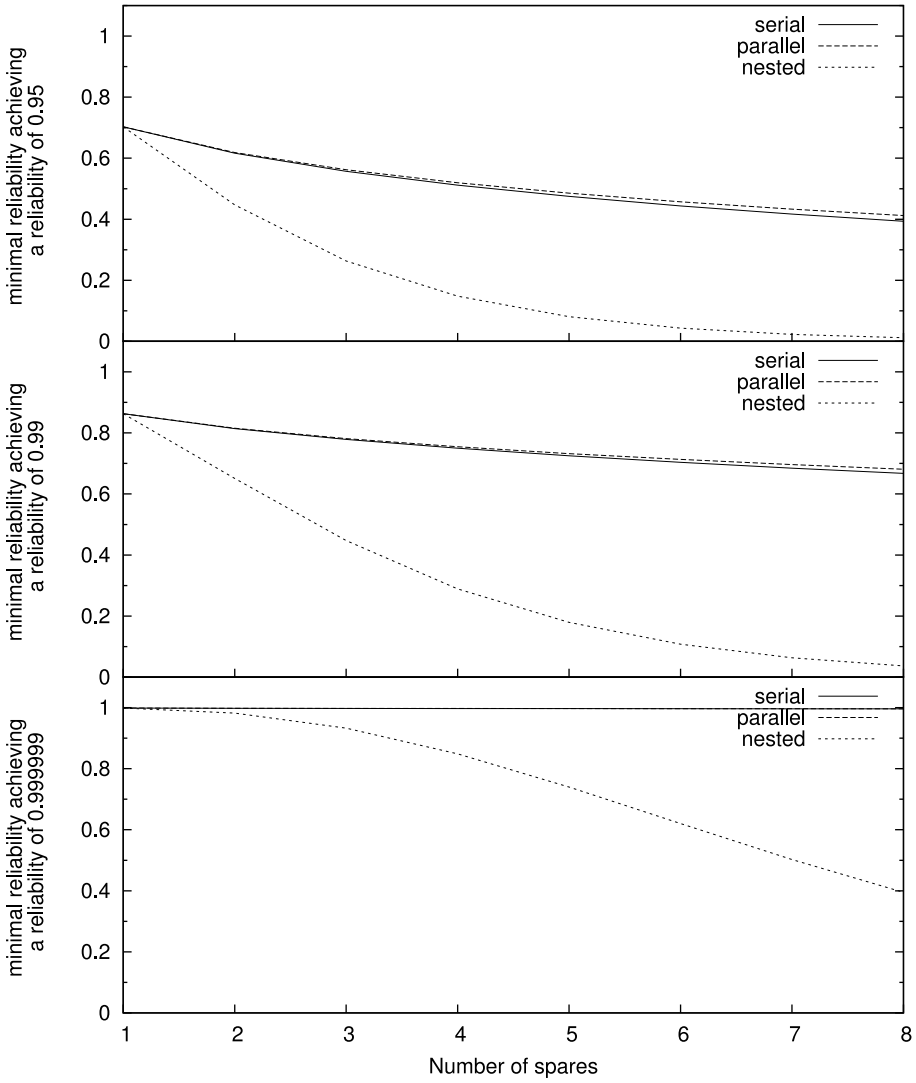


Figure 4.5: Minimal necessary reliability of the original 64-bit interconnection to achieve a 0.95, 0.99, or 0.999999 reliability using different segmentation schemes and different numbers of spares

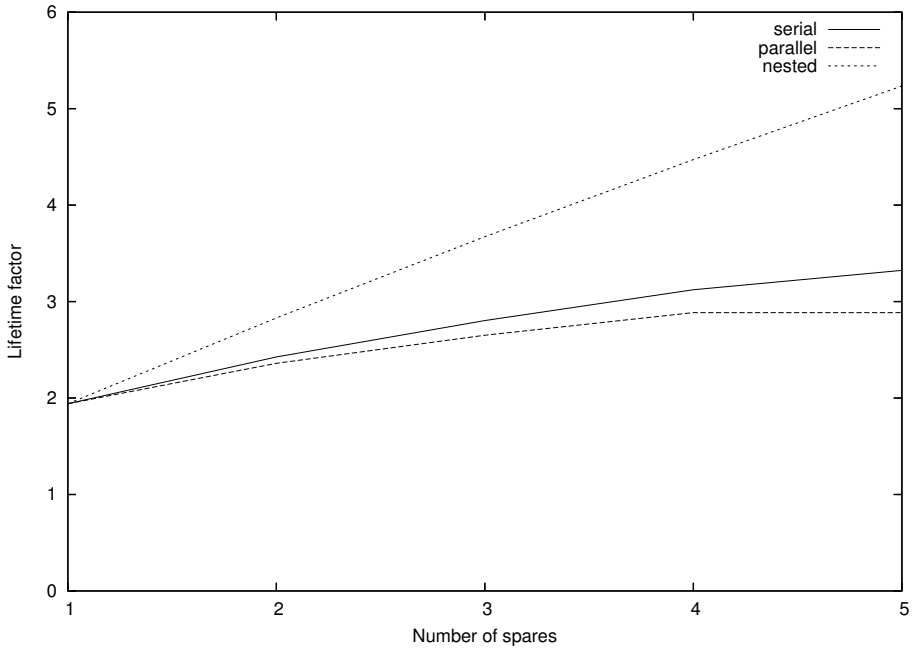


Figure 4.6: Lifetime factor (quotient of resulting and original MTTF) for the three segmentation schemes and different numbers of spares for a 16-bit-width interconnection

ing the three reliabilities leads to the mean times to failure (MTTFs). The quotient of the resulting MTTFs and the original MTTF is called the lifetime factor. The lifetime factors using serial, parallel and nested segmentation with different numbers of spares are depicted in figure 4.6. Nested segmentation leads to the highest MTTF, which confirms the results from figure 4.4. The relationship between the number of spares and the lifetime factor is nearly linear for nested segmentation. Serial and parallel segmentation lead to lower lifetime improvement. The relationship shows a decreasing slope. These relationships can be explained from the properties of the spare utilization. With nested segmentation, every multiple fault can be repaired until the spares are exhausted. For serial and parallel segmentation, multiple faults can be repaired until a segment has two faults. Thus, not all multiple faults can be repaired, which results in a lower lifetime improvement. Parallel segmentation needs a spare for every segment. Serial segmentation needs only one additional wire, which is divided into several segments. The total wire count is the highest count for parallel segmentation, which leads to the lowest reliability and the lowest mean

time to failure compared to serial and nested segmentation.

### 4.2.5 Cost comparison

To determine how expensive the repair of multiple faults is, the number of spares has been changed, while the original number of wires was constantly 64. The results are depicted in figure 4.7. The middle part depicts the same area results as the top part, but with a higher resolution. Serial, parallel and nested segmentation have been combined with rotate and bypass reconfiguration schemes. All combinations with the bypass reconfiguration scheme require less area than the combination with the rotate reconfiguration scheme, with one exception. Parallel segmentation in combination with rotate reconfiguration consume less area than bypass/nested and bypass/serial if more than four spares are used because the area of rotate reconfiguration depends quadratically on the width $n$ of the interconnection $A_r = n^2$. Parallel segmentation leads to smaller sub segments, and thus, $n^2 > S * (\frac{n}{S})^2$. The area consumption decreases with an increasing number $S$ of sub-segments. Except for parallel/rotate, the area use grows linearly with the number of spares.

Rotate reconfiguration, combined with nest segmentation, is the most expensive. Bypass/parallel is the combination with the lowest area consumption. The power consumption behavior is equivalent to the area consumption behavior because the static power consumption depends on the basic elements used. If the distribution is not changing, i.e., if the percentage of switches remains nearly the same, then there is a linear relationship between area and power consumption. This linear relationship is why I depicted only two data sets.

## 4.3 Administration

Whenever a fault has been detected, the associated SCs have to change their configuration. The tasks of the administration are to test the interconnection, to trigger the reconfiguration and to ensure that all involved SCs reconfigure synchronously. The communication used for administration has to be fault-tolerant. Additional requirements include the support of different topologies and interconnection implementations. To reduce the overhead and to increase the reliability, only a few additional wires for administration should be used.

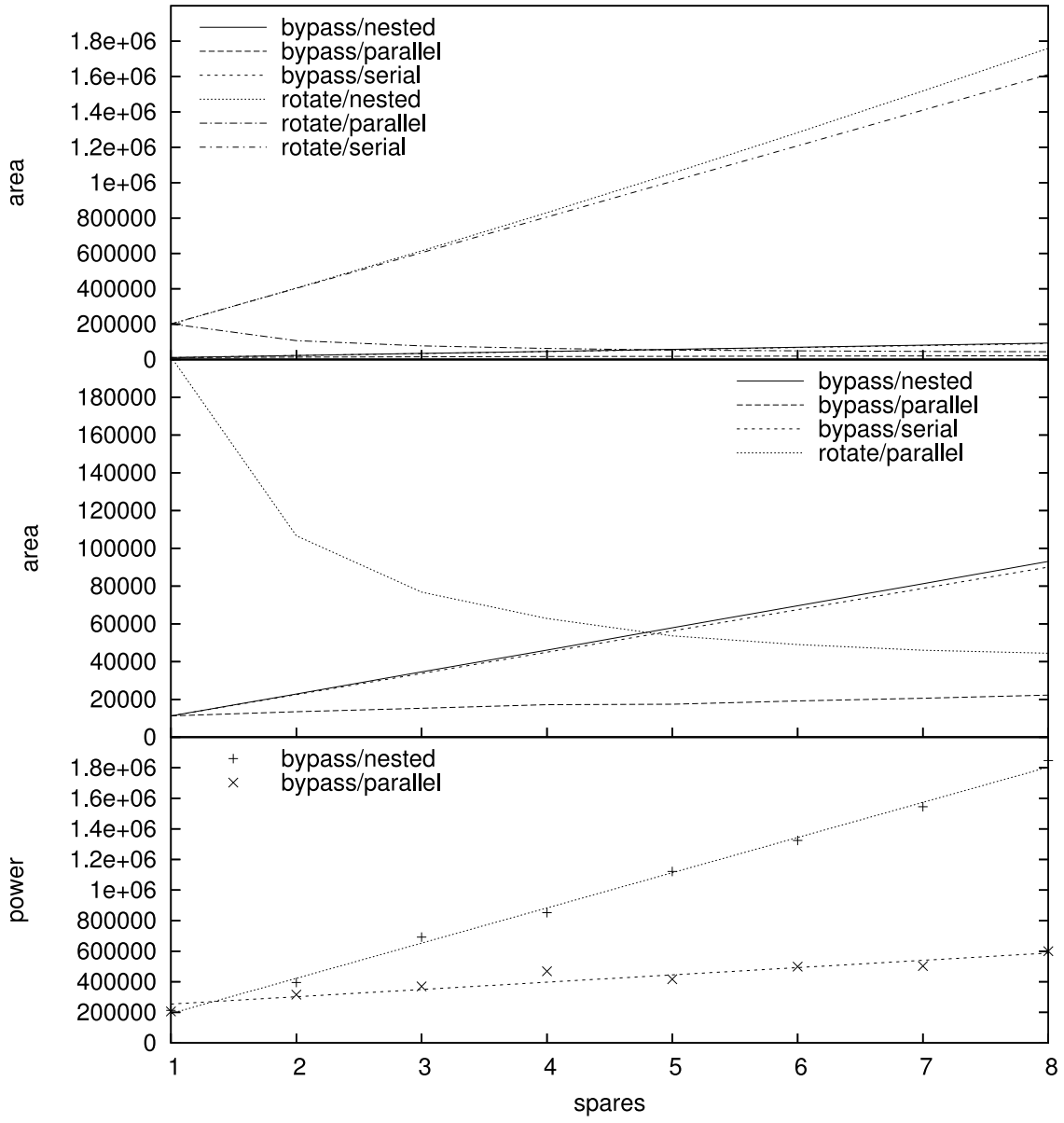


Figure 4.7: Area and power consumption of the combinations of reconfiguration schemes for a 64-bit width interconnection with different numbers of spares

### 4.3.1 Behavior of central and local administration

The test and repair of interconnections can be administrated centrally or locally. Central administration means that the whole administration is done from a single point. It is equivalent to BISR circuits. Local administration can be compared with self-checking circuits. There is a test-and-repair administration circuit for every segment of the interconnection. Both approaches have different behaviors and features.

*Central administration* is based on special-purpose logic or a built-in test processor for the test-and-repair routines. The test processor, which can also be used for a built-in self-test of the memory, administrates the bus reflectors/bus couplers for testing and the segment couplers for repairing. As such, the administration device has to be implemented only once. The use of a test processor allows for a large variety of test patterns that enable dynamic tests, in addition to the test for stuck-at faults. A further advantage of central administration is that all fault and configuration information about the interconnection is stored at one point. This central location eases fault diagnosis in the case of chip failure. The administration device is a single point of failure and has to be fault-tolerant itself, which is assumed to be true in this thesis. A further disadvantage is that central administration needs either bidirectional interconnections, or two unidirectional interconnections of the same size. Unidirectional on-chip interconnections are more common.

*Local administration* has to incorporate test-and-repair procedures for every segment. The advantage is that the test-and-repair procedures can be executed in parallel, which increases the speed of the test-and-repair tasks. Furthermore, there is no need for a fully-linked interconnection. Independent buses can be repaired independently. They do not have to be connected to a centralized test processor, which reduces the overhead for communication. Due to the distributed administration, there is no single point of failure. If a segment cannot be repaired, the fault could be compensated on a higher hierarchy, such as adaptive routing. The other segments would be still able to repair themselves. The test of locally administrated segments can be done using an error detection/correction code. Using error detection/correction code allows for online error detection instead of periodic tests, but comes with additional signal delay. Due to the code used for error detection, local administration is best suited for unidirectional interconnections.

The implementation of the two administration schemes are described in the fol-

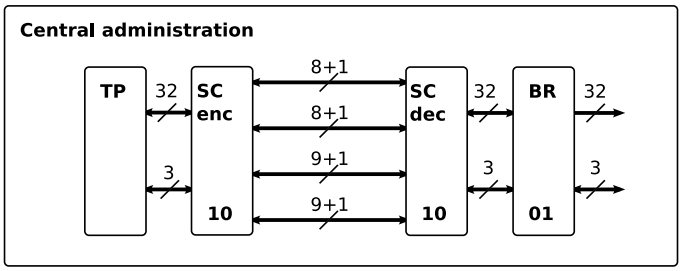


Figure 4.8: Centrally administrated BISR architecture for one segment of a 32-bit interconnection; the BISR architecture uses four spares (+1) and parallel segmentation

lowing two sections. In addition to the structure of the SCs and the additional BISR circuits, the test-and-repair procedures are described.

### 4.3.2 Central administration

A single segment of a centrally administrated BISR architecture with four spares and parallel segmentation is depicted in figure 4.8. The test processor (TP) controls the bus reflector (BR) for testing and the segment couplers (SC) for repairing. It has to be possible to activate them in the case of a faulty wire. The control is done using identification numbers (ID) and a control signal. The ID is transmitted using the original $n$ wires. The control signal requires an additional wire. For fault-tolerance, the m-bit ID and an additional control signal is combined with N-modular redundancy (NMR). The ID is transmitted N-times over the existing wires with the requirement that $N \cdot m \leq n$. The control signal is implemented N-times, thus requiring $N$ additional wires in total. The redundant IDs and control signals are compared in the SCs and the BR, which allows for transient and static fault masking. In the following, triple modular redundancy is used.

To test and repair the segment depicted in figure 4.8, the test processor activates the BR by sending the ID “01” three times in parallel and by activating the three-bit control signal. For every test pattern, the BR has to be activated. If a reconfiguration is necessary, the SCs are also activated by the test processor. The test processor sends their IDs “10” and activates the control signal. After reconfiguration, test takes place again to check whether the fault was repaired or not. This test process is repeated until the fault is repaired, or until all reconfiguration states have been tried. If there are several segments, the segments are tested and repaired along the

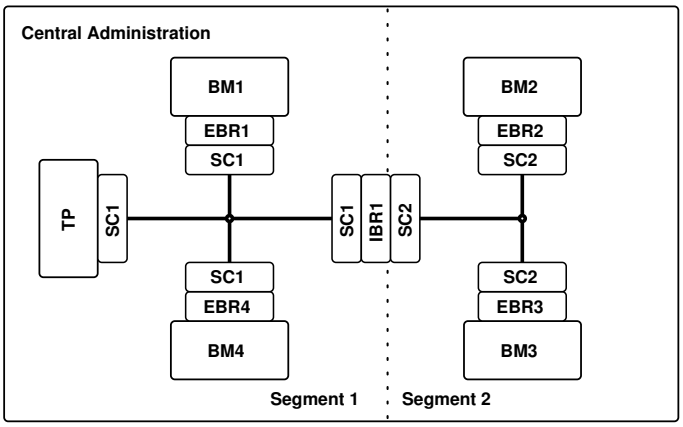


Figure 4.9: Interconnection with two segments using centrally administrated BISR

interconnection beginning with the segment in front of the test processor. The faults are repaired one after another to prevent fault accumulation.

### Internal and external Busreflector

The BR used for testing has two tasks. First, it has to isolate the segment from all connected bus masters (BM). Second, it enables interconnection testing. Therefore, the activated BR sets its output to high impedance and receives the test pattern from the test processor in the next clock cycle. In the following clock cycle, the BR sends the inverted test pattern. This procedure is done until all test patterns have been used, or a fault is detected. Through reflecting, one can detect all static faults and large-scale delay faults under worst-case switching conditions (31). The outputs facing the BMs have high impedance during both tasks.

In figure 4.9, an interconnection with two segments is depicted. The BRs can be used either directly in front of a bus master or anywhere else. They are called external BRs for the first case, and internal BRs for the second. The difference between internal and external BRs is how they are set into their isolation modes. External BRs (eBR1..4) are set to isolation mode whenever the control signal is active. Internal BRs (iBR1) are set to isolation mode when a neighbor BR is set to reflection mode. Both types of BRs are set to reflection mode when the control signal is active and the transmitted ID matches the ID of the according BR. If the test processor is going to test the interconnection part between the processor and eBR1, iBR1 and eBR4 will isolate segment 1. If segment 2 is tested, then all BRs (eBR1..4) will isolate the interconnection. Only the internal BR (iBR1) will not be

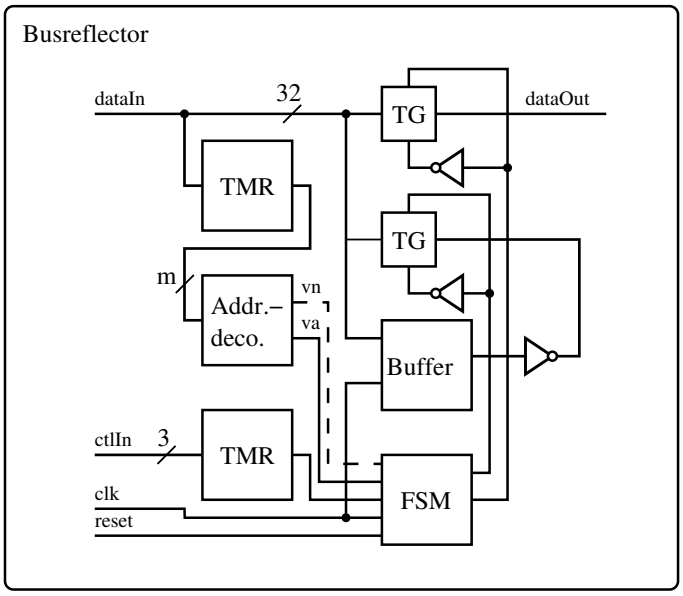


Figure 4.10: Structur of internal ($va\&vn$) and external (only $va$) BR

isolated and will bypass the test pattern.

The structure of the internal and external BR is depicted in figure 4.10. The signal *dataIn* and the signal *ctlIn* are compared using triple modular redundancy for fault-tolerance. The address decoder of the internal BR checks whether there is a valid ID ($va$) or a valid ID of an adjacent BR ($vn$). Depending on the two signals $va$ and $vn$, the isolating or reflection mode is activated if the control signal *ctlIn* is active. The buffer saves the test pattern that is inverted and sent back one clock cycle later. Transmission gates are used for the switches. Pass transistors would also be possible.

The structure of the external BR differs only in two points. First, there is no signal $vn$ to check if an adjacent BR has been activated. Second, the state machine has a different behavior. Isolation mode is activated whenever the control signal *ctlIn* is active.

### Segment Coupler

The implementation of the centrally administrated SCs is depicted in figure 4.11. The state machine is a simple counter with a one-hot output. The counter is incremented when the ID matches the SC ID and the control signal *ctl* is active. The ID is checked using triple modular redundancy. The same is true for the control signal

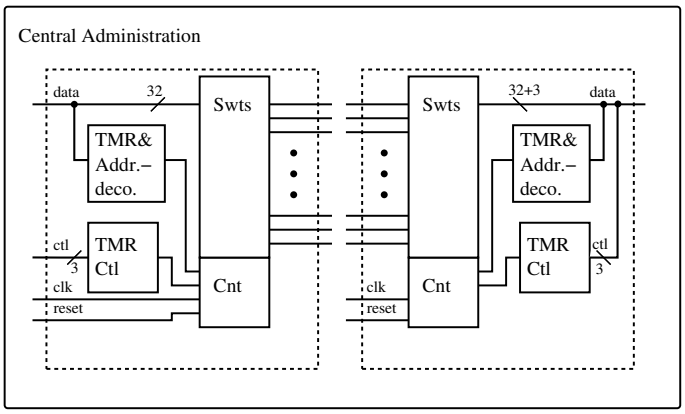


Figure 4.11: RTL-level implementation of the centrally administrated SCs

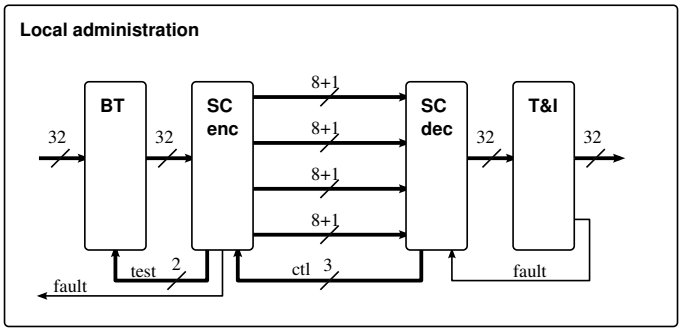


Figure 4.12: Centrally administrated BISR architecture for one segment of a 32-bit interconnection using four spares (+1) and parallel segmentation

*ctl*. The switches can be implemented as serial, parallel and nested segmentation for parallel or rotate switching.

### 4.3.3 Local administration

To test the interconnection using local administration, two additional circuits, in addition to the SCs, are necessary (figure 4.12). The bus tester (BT) is a circuit that can send test patterns dependent on the mode signal *test*. The test-and-isolate circuit (T&I) can be either a normal circuit with test mode or simply an error detection/correction code circuit in combination with a high impedance circuit. The task of the T&I circuit is to detect faults and to have high impedance outputs during test and repair to prevent fault propagation. In general, the SC decoder will initialize the test-and-repair process when the signal *fault*, which is controlled through the T&I circuit, is active. Test patterns are sent until the fault is detected or all patterns

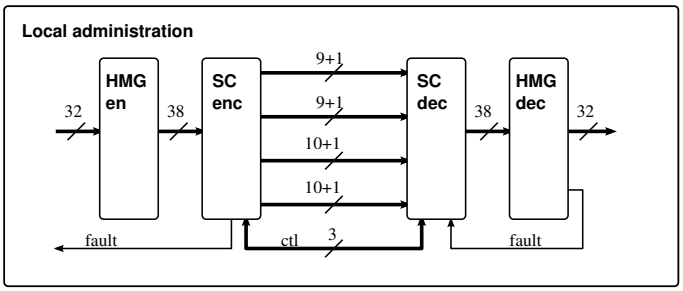


Figure 4.13: Locally administrated 32-bit segment using a Hamming code for testing and fault propagation prevention.

have been used. If the fault is not detected or corrected, the signal *fault* is bypassed to correct the fault on a higher hierarchy.

If testing is done using an error correction code, as in figure 4.13, an isolation circuit can be omitted because the error correction prevents fault propagation. In this example, the SCs divide the segment into four parallel subsegments. The test is started if a fault is detected through the Hamming code. The SC decoder waits for a fault signal that is longer than one clock cycle. If this signal appears, the SC decoder sends a fault signal to the encoder, and the repair process is started. The SCs will reconfigure as long as the fault is present. The BT is not necessary because testing is done with the error correction code.

The structure of the locally administrated SCs from figure 4.13 with four spares and parallel segmentation is depicted in figure 4.14. They consist of four switching circuits with local memory to save the configuration. The switching circuits consist of either tristate gates, pass transistors, or transfer gates. The global counter (*Glob Cnt*) is used for administration of the four switching circuits. Four spare wires are used; accordingly, four switching circuits are implemented. Thus, the global counter has four one-hot encoded states. The configuration of the switches is done by incrementing the local counter. When the local count has an overflow, the global counter is increased by one, and the next switching circuit is configured. The finite state machine is used for communication and administration of the reconfiguration process. The SC-encoder informs the SC-decoder via the three-bit wide signal *ctl* about the occurrence of a fault. The signal is implemented using triple modular redundancy with the ability to mask one fault. Triple modular redundancy is necessary to prevent unilateral reconfiguration, which would make the interconnection useless until the next global reset. The *ctl* signal is bidirectional to support handshaking

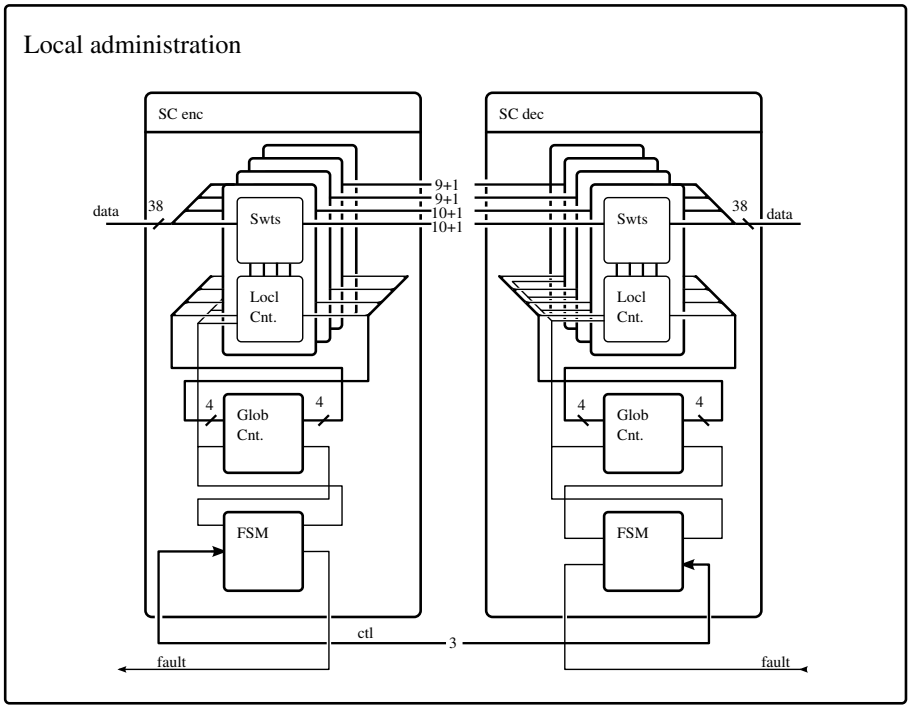


Figure 4.14: Implementation of the locally administrated SCs with four spares and parallel segmentation for a 32-bit interconnection encoded with Hamming code

protocols.

### 4.3.4 Cost comparison

As can be seen in figure 4.15, the choice of administration has little impact on the area and power consumption of the SCs. Central administration needs less area and power. Both administration schemes show a linear relationship between the interconnection width and the area/power consumed. The administration effort is constant over the interconnection width. This constant effort results in a constant offset, which decreases with increasing interconnection widths relative to the total area/power consumed. The difference in power consumption can be explained with the fitting of linear functions. As shown in the figure, the difference between the power consumed depends on the interconnection width. I could not find a relationship and assumed that it is related to the synthesis optimization algorithms. Central administration requires an administration circuit, such as a test processor. The area/power effort has to be added. The effort for the additional circuits required for BISR with local administration also has to be added.

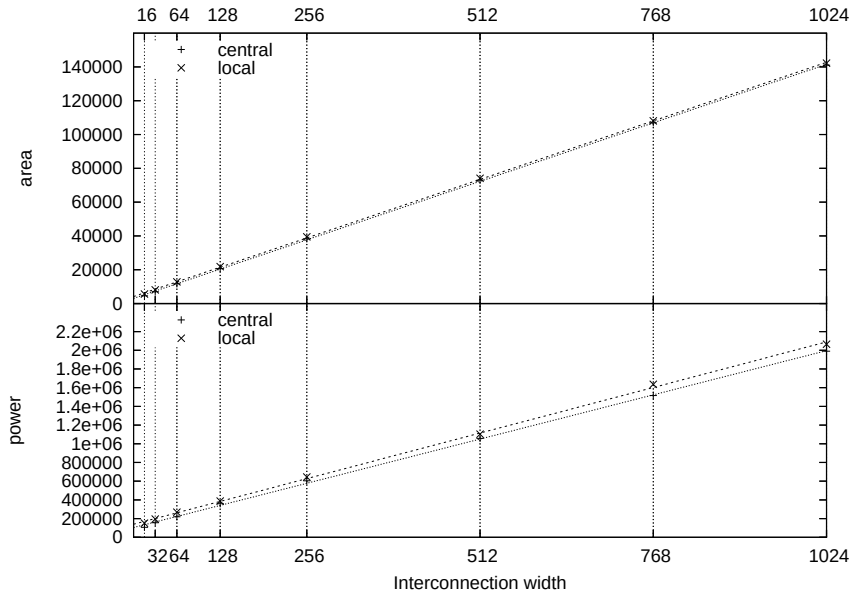


Figure 4.15: Area consumption of a centrally administrated and a locally administrated SC pair using bypass reconfiguration and one spare

## 4.4 Clocking scheme

Locally administrated SCs have to be synchronized. To ensure that the SCs reconfigure at the same time, they have to communicate with each other. This communication can be synchronous or asynchronous. Two possible state machines, one for synchronous and one for asynchronous communication, are depicted in figure 4.16.

When the decoder of the *synchronous* SC pair has received an active error signal, the decoder has to check whether it is a transient or permanent fault. This check is done by simply waiting for one or more clock cycles. If the error signal is still active after a certain amount of time, the test-and-repair process is started. The decoder sets an active fault signal, and the encoder sends the according ID for handshaking, which can be done using the BT. For synchronization, the decoder sets the fault signal to the inactive state for one clock cycle. Both circuits start the test. Various test patterns are used to find the error. Cycling through the test pattern is done until the error is found or the test patterns are exhausted. If the error is detected, the interconnection reconfiguration is changed until the error is corrected or until all reconfiguration states have been tried. In the latter case, the fault cannot be

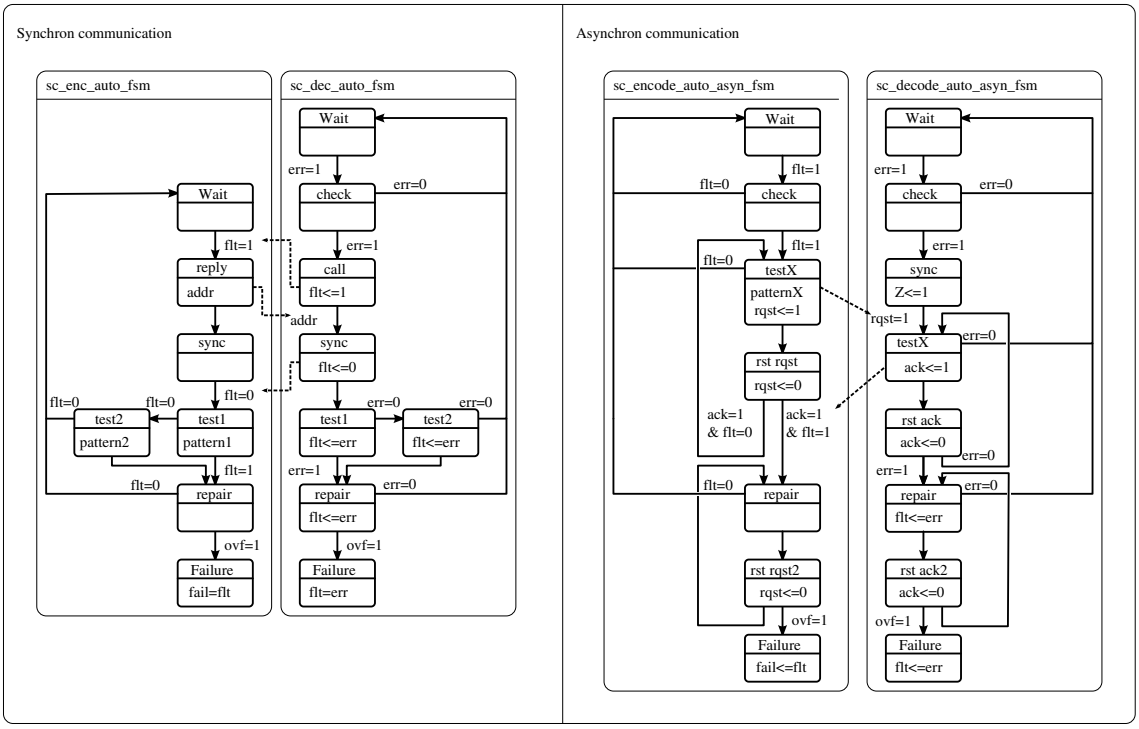


Figure 4.16: State machine for synchronous and asynchronous communication

repaired and is propagated permanently for error correction on a higher hierarchy.

The *asynchronous* communication starts when the encoder and decoder have detected an error longer than one or more clock cycles. If only one of the two circuits detects a fault, the synchronization is started and, due to a missing handshake, aborted after a certain time. This behavior is not depicted to reduce the schematic complexity. After the detection of the static error, the encoder starts the synchronization and testing. The decoder isolates the interconnection and sends the acknowledge signal. Testing is done until the error is detected or all test patterns have been used. If the fault is detected, the interconnection is reconfigured until the fault is corrected or all reconfigurations have been tried. The reconfiguration ends with success or failure. If the error could not be corrected, the error signal is bypassed to allow correction on a higher hierarchy.

The synthesis results for synchronous and asynchronous communication are depicted in figure 4.17. The difference between area and power consumption is small. Synchronous communication needs less area and power because the communication protocol is simpler, resulting in a simpler state machine.

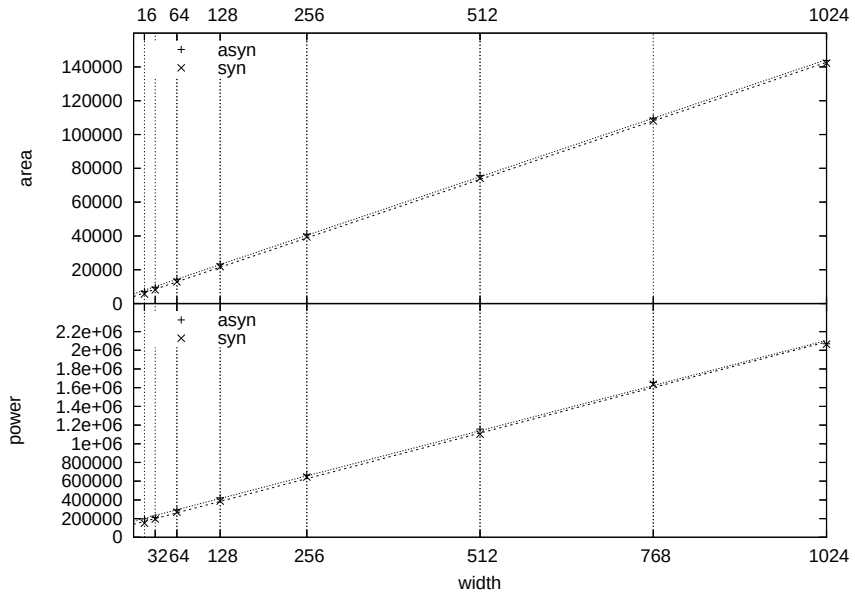


Figure 4.17: Area consumption using synchronous or asynchronous communication

## 4.5 Summary

In this chapter, the BISR circuits have been implemented and evaluated. The costs for reconfiguration depend on the type of replacement (bypass/rotate) and on the type of segmentation (serial/parallel/nested). Bypass replacement has a linear dependency on the interconnection width, while rotate replacement has a quadratic dependency; as such, codes with a high code rate lead to smaller BISR circuits than codes with a low code rate. For multiple faults, nested segmentation leads to the highest reliability. However, serial segmentation allows for capacitive balancing. If facing multiple faults, parallel segmentation can reduce the cost for rotate replacements. The administration costs are dependent on where the administration occurs (central/local) and what the communication protocol is (synchronous/asynchronous). However, the results show only small differences between the various implementations. Thus, the influence of these design decision on costs is small; i.e., costs are not the main criteria for these design decisions. The minor effect on cost is an advantage because the design space of the communications architecture is not limited by the administration of the proposed BISR circuits.

# Chapter 5

# BISR-CODE COMBINATIONS

The proposed BISR is now compatible with existing error detection, error correction, and crosstalk avoidance codes. In this section, the general BISR code architecture (BISR+C) is described. BISR+C, is an extension of the Unified Coding Framework (60). I have implemented combinations of BISR with a code for transient faults and a code for dynamic faults. The results identify which codes have to be used in combination with BISR for maximum reliability and lifetime.

## 5.1 BISR+C architecture

The BISR+C architecture is based on the Unified Coding Framework (60). The encoder is depicted in figure 5.1. The decoder works in the opposite direction. Where the test patterns have to be applied depends on whether the testing is done using an error correction/detection code or not.

If *no ECC/EDC is used* for testing, the circuits used for testing require full read and write access to all wires that might have to be repaired. No ECC/EDC is the

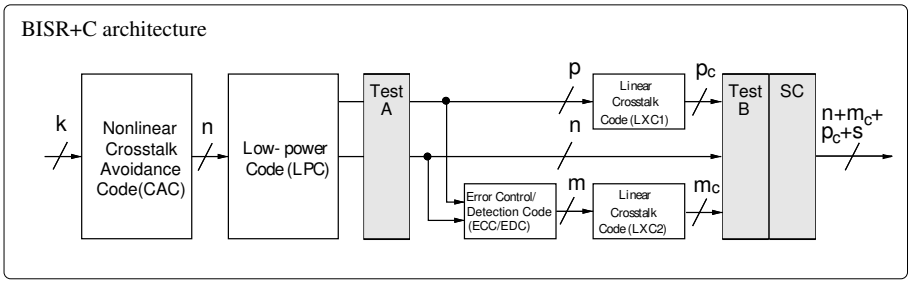


Figure 5.1: Encoder of the BISR+C architecture

case for central administration, which needs a test processor and a BR/bus coupler. The test pattern has to be available at point *Test B*, behind all coding circuits, to test the interconnection. If there are several serial interconnection segments through serial segmentation, it must be possible to bypass all coding circuits. This bypass is costly because it requires multiplexers.

If *an ECC/EDC is used* for testing, there are two possibilities that depend on whether the fault has to be repaired at once or in several steps. For the first possibility, whenever a fault occurs for more than one clock cycle, test patterns (*Test A*) are used to make the fault visible and to reconfigure until the fault is repaired. During test and repair, the interconnection is out of order. The possibility of a fault accumulation between a static fault and a transient fault is being minimized by the repair. The disadvantages are the need for a circuit to generate test patterns (BT) and the halting of normal communication. If the repair is done in several steps, the disadvantages are avoided at the cost of longer time, until the static fault has been repaired. Whenever a fault longer than one clock cycle is detected, only one reconfiguration step is executed. No additional test patterns are needed to make the fault visible. The probability that a stuck-at fault is visible within two clock cycles is 25% when the zeros and ones are distributed equally. The one-step reconfiguration works with error correction codes and error detection codes using immediate automatic retransmission request (ARQ). In this thesis, the condition that there is a static fault whenever a fault occurs for more than one clock cycle has been used. Of course, there are other possible conditions for detecting static faults. For example, the reconfiguration could be triggered whenever there are two faults within 4 clock cycles, which would increase the detection speed at the cost of additional flip-flops and a decoder.

## 5.2 Results

In the following section, the BISR+C architecture is compared with a purely code-based solution. First, the achievable reliability is discussed, followed by a comparison of lifetime and costs. In the last subsection, the influence of the code rate of crosstalk avoidance codes (CAC) on lifetime and costs is discussed in more detail.

All combinations can correct one static fault. A point-to-point interconnection consisting of an encoder and a decoder circuit is considered. The number of resulting

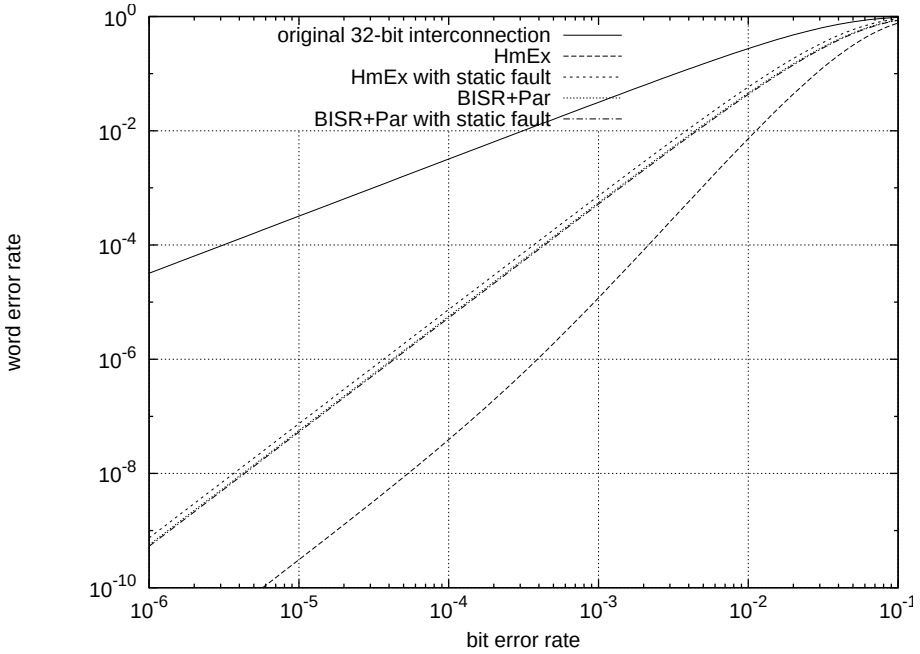


Figure 5.2: Remaining fault rate using BISR and codes to compensate transient and static faults for a 32-bit width interconnection

wires has been calculated and the resulting area has been gathered from synthesis results. With the exception of the Hamming/FTC combination, a triple signal for an automatic repeat request (ARQ) has been implemented to ensure a fully fault-tolerant interconnection. Triple modular redundancy is used to correct either one static or one transient fault. If this is not enough, n-modular redundancy can be used. The number of ARQ wires is small compared to the total number of wires and continues to decrease relative to increasing interconnection widths. Thus, the interconnection reliability is only slightly influenced.

### 5.2.1 The influence of static faults on the transient fault rate

In figure 5.2, the transient word error rate for a 32-bit wide interconnection is depicted as a function of the bit error rate. BISR combined with parity code (Par) is compared with the extended Hamming code (HmEx). The BISR has one spare. Thus, BISR+Par and HmEx are able to correct one transient and one static fault at the same time. The word error rate has been depicted for two cases. In the first case, no static fault is present; in the second case, a static fault has occurred. The

extended Hamming code leads to the lowest word error rate as long as no static fault is present. In this case, two transient faults can be corrected through retransmission. If a static fault is present, only one transient fault can be detected. This fault detection results in a word error rate that is higher than the BISR-based solution. The extended Hamming code needs additional wires to correct one transient fault, which increases the word error rate. However, this increased error rate is better than the unprotected interconnection. The BISR+Par combination leads to nearly constant word error rates whether there is a repaired static fault or not. In both cases, one transient fault can be detected. Fewer additional wires are necessary, which leads to a lower word error rate in the case of a static fault compared to the extended Hamming code. This result confirms the discussion from chapter 3: purely code-based solutions are not the best solutions for static faults due to the higher number of additional wires.

Figure 5.3 depicts the same word error rate to bit error rate relationship as in figure 5.2 with an additional crosstalk avoidance code FTC. This additional code requires additional wires and results in an increased word error rate which can be seen in the second part of the diagram. In this part of the diagram, the quotient between the upper part of this diagram and figure 5.2 is depicted. $ST$ stands for solutions for transient and static faults, and $SDT$ is an abbreviation for solutions with additional dynamic fault handling. The additional wires through the crosstalk avoidance code lead to an increased word error rate. The extended Hamming code without a static fault can correct two faults. Thus, for smaller bit error rates the influence of the additional wires decreases because the chance of a triple fault decreases quickly with decreasing bit error rates.

### 5.2.2 Lifetime comparison

The achievable lifetime factors are depicted in figure 5.4. The lifetime factor ($LF$) is the quotient of the mean time to failure of the fault-tolerant interconnection ($MTTF_{ft}$) to the MTTF of the original interconnection ($MTTF_{orig}$).

$$LF = \frac{MTTF_{ft}}{MTTF_{orig}} \tag{5.1}$$

The figure shows that the lifetime factor increases with an increasing interconnection width. The combinations without the crosstalk avoidance code FTC trend toward a factor of 2.0, whereas the combinations with FTC trend toward 1.5. The increasing

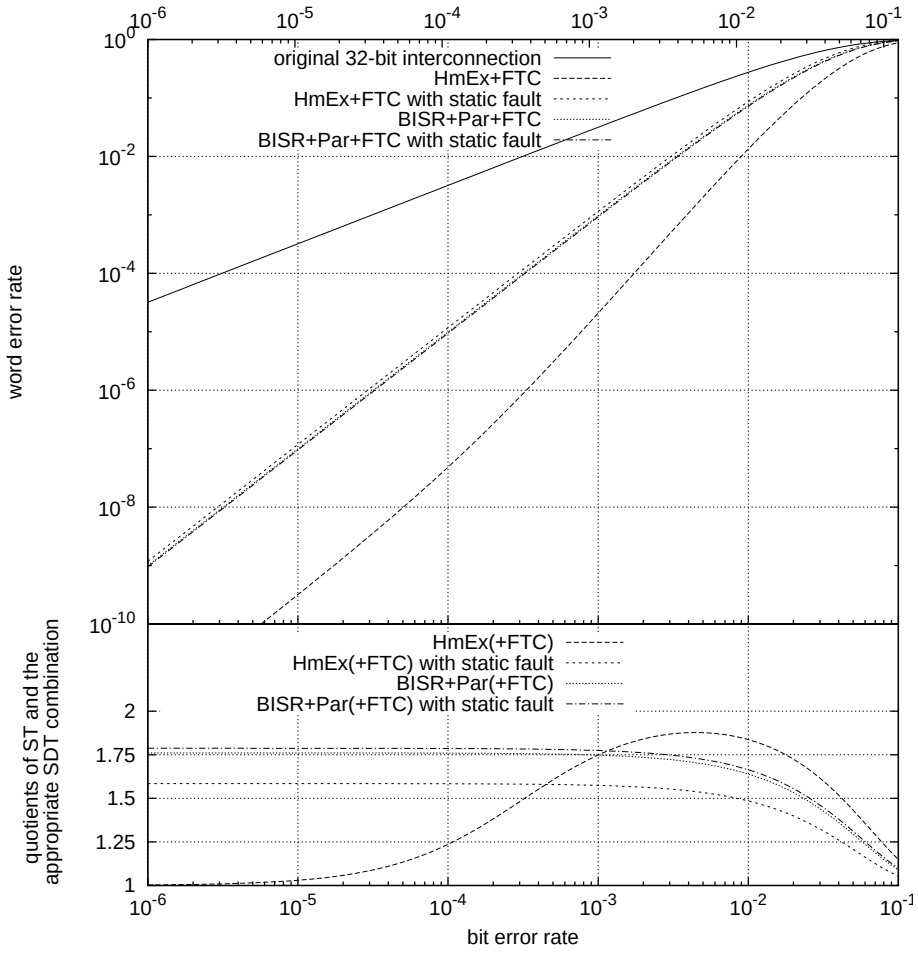


Figure 5.3: Remaining fault rates using BISR and codes to compensate transient, dynamic and static faults for a 32-bit wide interconnection

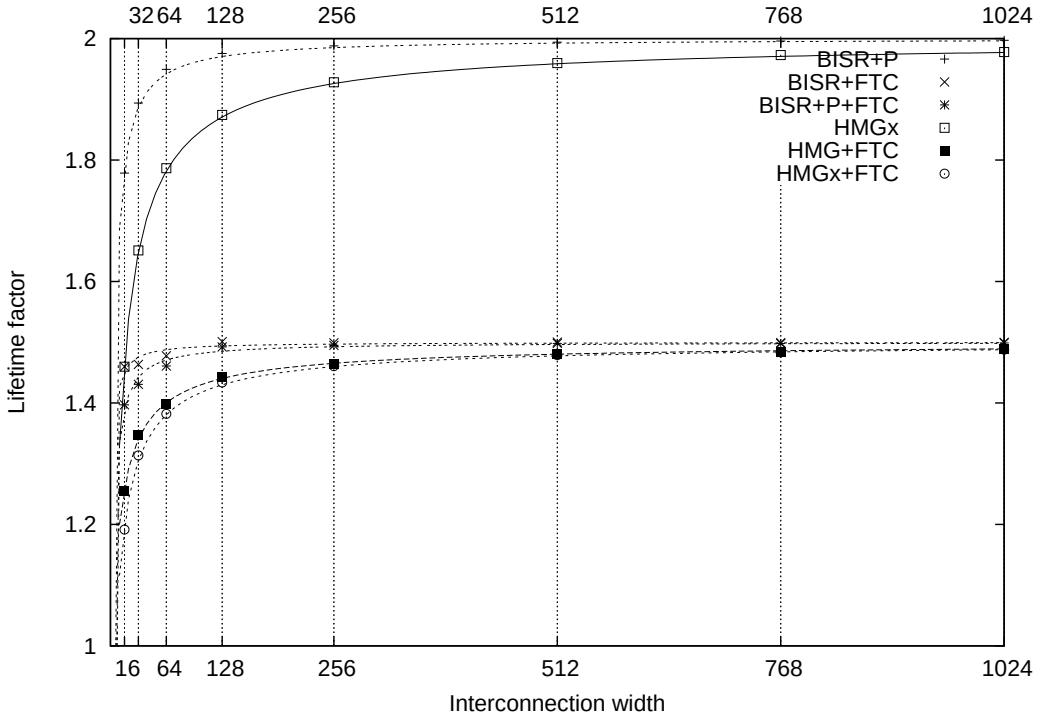


Figure 5.4: Lifetime factor (quotient of resulting and original MTTF) for different combinations and interconnection widths

lifetime factor can be explained as follows. Static faults can be corrected with the use of additional wires. In this case, one static fault should be correctable. In the ideal case, the MTTF can be doubled. The ideal case exists when the additional wires are faultless. In the used lifetime model, the additional wires have the same probability of failing as the original ones. Thus, the MTTF is reduced by these wires. The more wires per correctable fault required, the smaller the lifetime factor. BISR in combination with Parity (BISR+P) needs only two wires in addition to those for the interconnection. The extended Hamming code (HMGx) needs $log_2 n + 1$ wires. That is why the lifetime factor is always smaller than the one for BISR+P. The wider the interconnection is, the smaller the percentage of the additional wires required for both static and transient (ST) fault-tolerant interconnections. Therefore, the lifetime factor increases with the interconnection width. This relationship is also valid for static/dynamic (SD) and static/dynamic/transient (SDT) fault-tolerant interconnections. The difference is the maximal achievable lifetime factor. The forbidden transition code (FTC) has a constant code rate of 4/3. The maximal lifetime factor decreases from the ideal of 2 to $\frac{3}{4} \cdot 2 = \frac{3}{2}$.

### 5.2.3 Cost comparison

Figure 5.5 shows a linear relationship between the number of wires for the fault-tolerant interconnection and the original interconnection. The combinations using the crosstalk avoidance code FTC need more wires than the combination without FTC. The solution with the fewest wire count is the BISR+P combination, followed by HMGx. The number of wires consists of the number of original wires and the number of additional wires. The percentage of additional wires depends on the interconnection width. Some wires, such as those used as spares, have constant width. Other signals, such as ARQ and Parity, are also constant width. Additionally, there are signals with widths that depend on the interconnection width linearly, as is the case for the FTC(4,3). Some signal widths have a nonlinear dependency; for example, those used for the Hamming code follow a base-two logarithm. The original wires are the main component of the wire count. If it is used, the second largest component is the FTC. Thereafter, the logarithmic and the constant wire counts follow. The original and the crosstalk avoidance code wires have a linear relationship with the interconnection width. The influence of the other parts decreases with increasing interconnection widths. Thus, there is a linear relationship between the

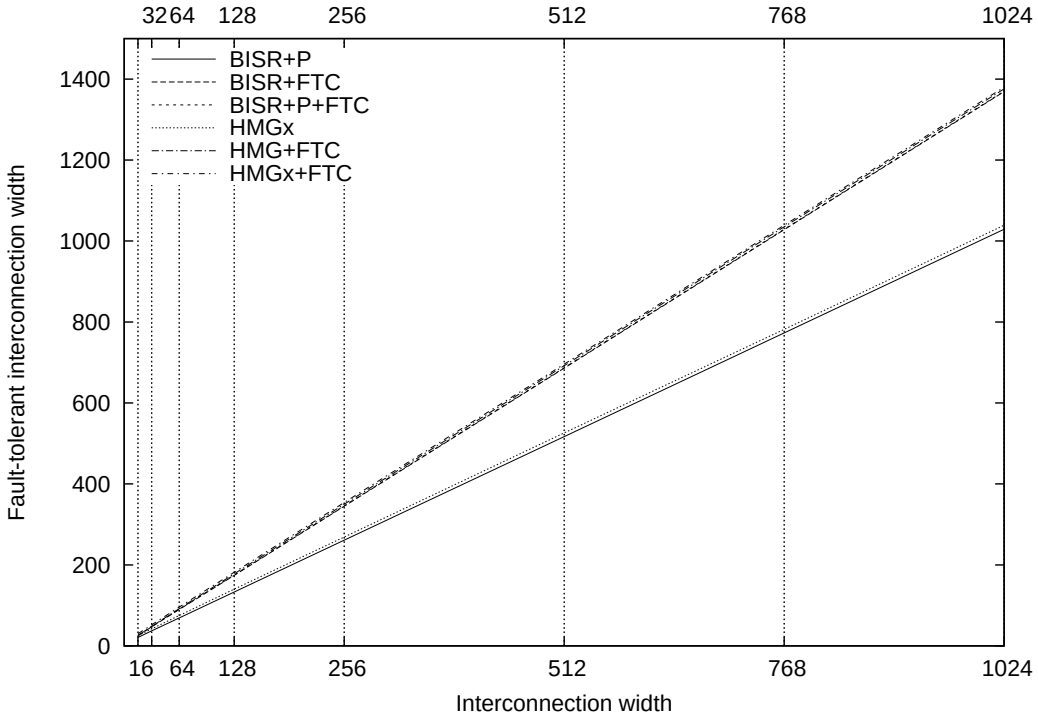


Figure 5.5: Resulting numbers of wires for different combinations and interconnection widths

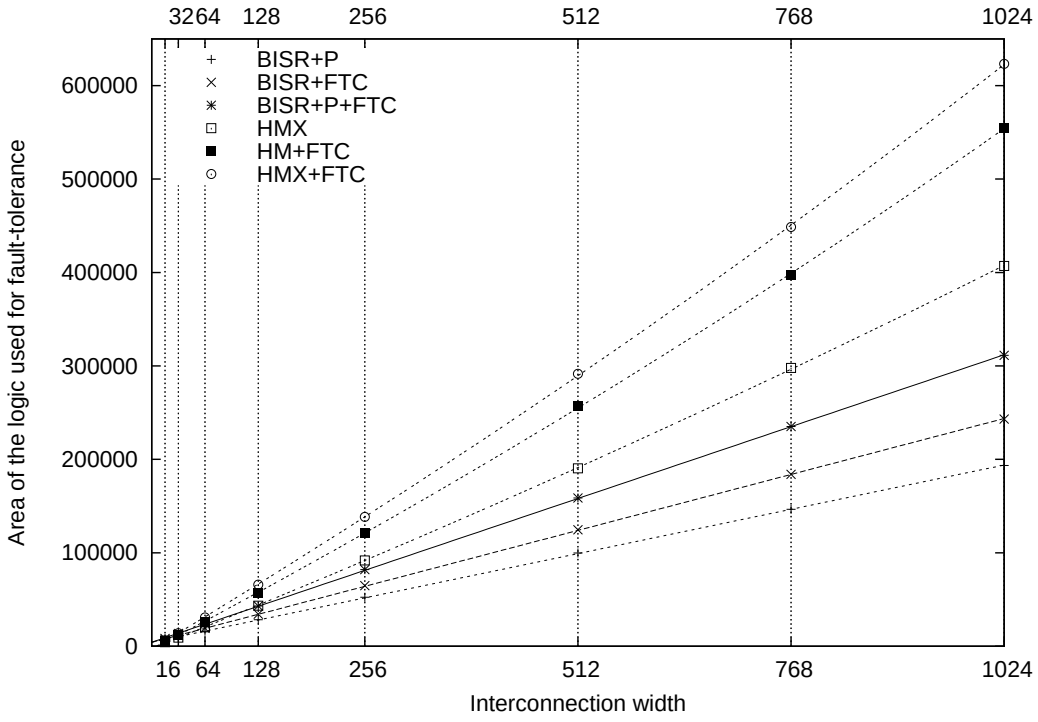


Figure 5.6: Area consumption for different combinations and interconnection widths

wires of the fault-tolerant interconnection and the original interconnection.

The area consumption of the fault-tolerant interconnections is depicted in figure 5.6. The interconnections that use BISR need less area than the comparable Hamming-code-based interconnection. The area of BISR-based interconnections increases linearly with the interconnection width. The area of Hamming-code-based interconnections increases exponentially. The linear area increase results from the linear behavior of the BISR circuits and the linear behavior of the FTC en- and decoder. The exponential area increase is caused by the Hamming code decoder. Again, curve fitting confirms the linear and exponential behavior.

### 5.2.4 The influence of crosstalk avoidance codes on lifetime and costs

In the previous figure 5.4 on page 66, it can be seen that the crosstalk avoidance code greatly influences the achievable lifetime. The fewer the number of required additional wires, the higher the reliability. In figure 5.7, the lifetime factor, the area consumption and the area ratio of the combination of BISR and two crosstalk avoidance codes (CAC), the Forbidden Transition Code (FTC[4,3]) and the Forbidden Pattern Code (FPC[5,4]) are depicted. The area ratio is the quotient of the area required only for the CAC and the area required for the combination.

In the first part of the figure, the lifetime factor is depicted. As discussed in the previous figure, the maximum achievable lifetime factor depends on the code rate and the number of spares. Thus, the maximum lifetime for the FTC(4,3) is $\frac{3}{4} \cdot 2 = 1.5$, and for the FPC(5,4) is $\frac{4}{5} \cdot 2 = 1.6$.

The second part shows the area consumption. For both combinations, the area depends linearly on the interconnection width. The BISR+FTC combination needs slightly less area. The BISR and the CAC circuits have a linear relationship between their areas and interconnection widths; thus, the sum of both is also linear. An interesting point is the fact that the area difference is quite small. The FTC has a smaller code rate, but this was done to decrease the logic required for the en- and decoder. The area difference should be larger.

The area difference can be explained using the third part: the area ratio. The area ratio shows a large slope from 16 to 128 wires. This slope is due to the communication unit of the SCs used for BISR, which has a constant area. Thus, the influence is decreasing with increasing interconnection width. Later, the ratio trends toward a constant value. The percentage amount of the FPC is higher than the one from the FTC. Although the combinations differ only slightly, that the FPC en- and decoder requires more area than the FTC en- and decoder. The FTC, on the other hand, needs more wires. The area of the SCs depends linearly on the number of wires that need to be repaired. Thus, the FTC needs less area, but due to the higher wire count the combined SCs need more area and negate nearly all of the FTC area savings. In summary, to achieve high MTTFs, it is better to use a CAC with a higher code rate and higher area consumption because the area savings through a reduced code rate is nearly compensated through the BISR circuits. The higher MTTF comes at

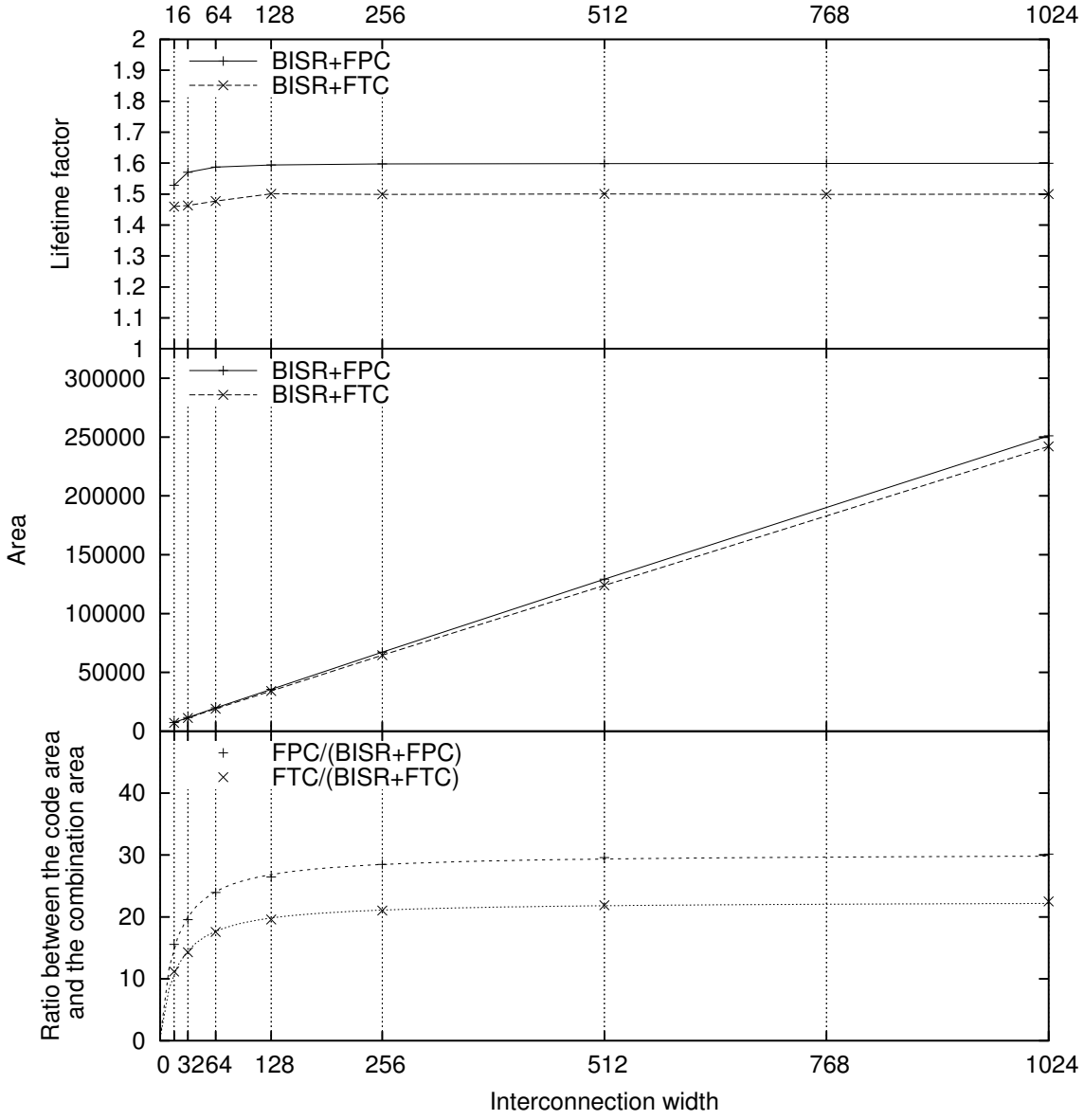


Figure 5.7: Lifetime factor (quotient of resulting and original MTTF), area consumption and area ratio of BISR and crosstalk avoidance codes (FTC/FPC) combinations

the cost of higher dynamic power consumption due to the increased active logic in the CAC en- and decoder.

### 5.2.5 Conclusions

The conclusions of the results are the following. If one wants to achieve high lifetime factors, then it is necessary to keep the number of additional wires as small as possible. If a crosstalk avoidance code is used, a code with a high code rate should be used. The code rate limits the maximum achievable lifetime factor. Of course, the usage of additional spares increases the MTTF. BISR allows the use of as many spares as necessary with linear area and wire costs, but comes with further signal delay when using nested or serial segmentation. Nested segmentation leads to the highest reliability increase (see figure 4.6 on page 47). The crosstalk avoidance code is also crucial to the total wire count. Again, a CAC with a higher code rate would fit best. Shielding wires have not been modeled because it was not clear to me what failure rate to assume. If the failure rate were the same, shielding wires would decrease the reliability greatly due to their high wire count. Concerning area consumption, the results show that the BISR-based solutions need less area than Hamming-code-based ones. The consumption can be only 50% of the Hamming code solutions. Furthermore, the Hamming code solutions have higher dynamic power consumption because the logic state depends on the data input. This operation stands in contrast to the BISR, where the state only changes when the repair takes place. In summary, BISR-based solutions need less energy, less area and result in higher MTTF compared to the Hamming-code-based solutions. Furthermore, BISR-based solutions can correct more than one static fault. Double error correction codes or triple error correction codes come with a high expense in wires and logic (45; 24; 27).

# Chapter 6

# CONCLUSION AND OUTLOOK

There are several solutions that can help ensure reliable interconnections. The point to start at is optimizing technology to prevent faults. Simulation-based routing, shaping of the wires, and a good thermal design result in fewer problems with electro- and stress-migration. Keeping the interconnects as simple as possible will also result in improved reliability. If fault prevention is impossible, then the system has to correct or repair the occurring faults. Codes are the best solution for dynamic and transient faults. They are mature, well known, and easy to implement. Unfortunately, they are not the best solution for static faults. BISR uses less wires and can lead to higher reliability. But transient and dynamic faults occur more often. Thus, BISR has to be combined with codes.

The implementation and evaluation of the BISR code combination (BISR+C) is the main topic of this thesis. I have extended the existing BISR solutions to be compatible with solutions against dynamic faults; for example, using crosstalk avoidance codes. I had to change the art and manner of spare wire utilization, and I had to implement a fault-tolerant test-and-repair administration. Compared to existing solutions, this new and possible combination leads to higher reliability. The developed BISR circuits are flexible, according to the number of wires, spares, the type of clocking and the place of administration. This flexibility ensures a high applicability of the proposed circuits.

The logic effort for the BISR interconnections depends mainly on the interconnection topology and the number of wires that are repairable. The administration implementation (central/local and synchronous/asynchronous) has only a small impact on the logic effort. This minor impact on logic effort is why codes with only

a few additional wires, and interconnections with only a few segments based on the topology, is best suited for the BISR+C architecture. The inter-switch interconnections in network-on-chip systems, for example, are well suited. They consist of unidirectional peer-to-peer connections using block and cyclic redundant codes. The test-and-repair process can either be organized by the local administration or by the switch circuit itself. The used codes allow online error detection. Using BISR+C in inter-switch interconnections would result in fault-tolerant NOC communications on a deeper hierarchical level than existing fault-tolerant NOC routing algorithms, which would extend the total NOC lifetime and increase the NOC reliability.

Of course, there are constraints for the proposed solution. I have assumed that the clock and reset signals and the used logic are faultless. All modeled reliabilities are best case reliabilities according to these constraints. Faults on the clock and reset signals and in the administration and switch logic decrease system reliability.

The focus of this thesis was to discuss the principles of interconnection BISR in combination with codes. For real applications, the following points should be considered. First, there is a need for optimized communication protocols. The protocols proposed here are simple because the only purpose was to show that it is principally possible to implement different clocking schemes for BISR. In the case of multiple faults, the question of how to save and restore the SC reconfiguration status must be solved. Restoring the configuration status could be done either with more complex communication protocols or with local permanent memory. The most effort has to be put into the extension of existing EDA software for the purposes of reliability calculation and automatic placement. Reliability calculation on the layout level, combined with the automatic placement of BISR circuits, is necessary to find the best solution for a specific circuit. Reliability calculation is necessary because the additional BISR circuits for interconnection reliability enhancement decrease the total logic reliability. Additional logic leads to additional possibilities for failures. There is an optimum between logic and interconnection reliability that leads to the highest total reliability, dependent on the logic and interconnection fault rates. In the future, this optimum could be found by synthesis and reliability software. Then, the use of the proposed BISR+C architecture could be automated, dependent on the reliability calculation, like scan chain insertion, to enable rapid implementation of fault-tolerant interconnections.

# Bibliography


[1] Sematech: International Technology Roadmap for Semiconductors 2007 Interconnect. Version: 2007. http://www.itrs.net/Links/2007ITRS/2007_Chapters/2007_Interconnect.pdf. 2007. – Forschungsbericht

[2] Abella, Jaume ; Vera, Xavier: Electromigration for microarchitects. In: *ACM Comput. Surv.* 42 (2010), Nr. 2, S. 1–18. http://dx.doi.org/http://doi.acm.org/10.1145/1667062.1667066. – DOI http://doi.acm.org/10.1145/1667062.1667066. – ISSN 0360–0300

[3] Alam, Syed M.: *Design Tool and Methodologies for Interconnect Reliability analysis in integrated circuits*, Massachusetts Institute of Technology, Diss., September 2004

[4] Alderighi, M. ; Casini, F. ; D'Angelo, S. ; Salvi, D. ; Sechi, G. R.: A fault-tolerant FPGA-based multi-stage interconnection network for space applications. In: *Proc. First IEEE International Workshop on Electronic Design, Test and Applications*, 2002, S. 302–306

[5] Ali, Muhammad ; Welzl, Michael ; Hessler, Sven: A Fault tolerant mechanism for handling Permanent and Transient Failures in a Network on Chip. In: *Information Technology: New Generations, Third International Conference on* 0 (2007), S. 1027–1032. http://dx.doi.org/http://doi.ieeecomputersociety.org/10.1109/ITNG.2007.5. – DOI http://doi.ieeecomputersociety.org/10.1109/ITNG.2007.5. ISBN 0–7695–2776–0

[6] Alsaiari, Uthman ; Saleh, R.: Power, Delay and Yield Analysis of BIST/BISR PLAs Using Column Redundancy. In: Saleh, R. (Hrsg.): *Proc.*

*8th International Symposium on Quality Electronic Design ISQED '07*, 2007, S. 703–710. – printed

[7] Anand, D. ; Cowan, B. ; Farnsworth, O. ; Jakobsen, P. ; Oakland, S. ; Ouellette, M.R. ; Wheater, D.L.: An on-chip self-repair calculation and fusing methodology. In: *IEEE Design & Test of Computers* 20 (2003), Nr. 5, S. 67–75. http://dx.doi.org/10.1109/MDT.2003.1232258. – DOI 10.1109/MDT.2003.1232258. – ISSN 0740–7475

[8] Attarha, A. ; Nourani, M.: Testing interconnects for noise and skew in gigahertz SoCs. In: *Proc. International Test Conference*, 2001, S. 305–314

[9] Attarha, A. ; Nourani, M.: Test pattern generation for signal integrity faults on long interconnects. In: *Proc. 20th IEEE VLSI Test Symposium (VTS 2002)*, 2002, S. 336–341

[10] Avinash, L. ; Krishna, M. K. ; Srinivas, M. B.: A Novel Encoding Scheme for Delay and Energy Minimization in VLSI Interconnects with Built-In Error Detection. In: *Proc. IEEE Computer Society Annual Symposium on VLSI ISVLSI '08*, 2008, S. 128–133

[11] Bakos, J. D. ; Chiarulli, D. M. ; Levitan, S. P.: Lightweight Error Correction Coding for System-Level Interconnects. 56 (2007), Nr. 3, S. 289–304. http://dx.doi.org/10.1109/TC.2007.49. – DOI 10.1109/TC.2007.49. – ISSN 0018–9340

[12] Barnett, Thomas S. ; Grady, Matt ; Purdy, Kathleen G. ; Singh, Adit D.: Combining Negative Binomial and Weibull Distributions for Yield and Reliability Prediction. In: *IEEE Design and Test of Computers* 23 (2006), S. 110–116. http://dx.doi.org/http://doi.ieeecomputersociety.org/10.1109/MDT.2006.38. – DOI http://doi.ieeecomputersociety.org/10.1109/MDT.2006.38. – ISSN 0740–7475

[13] Baughman, Ray H. ; Zakhidov, Anvar A. ; Heer, Walt A.: Carbon Nanotubes–the Route Toward Applications. In: *Science* 297 (2002), Nr. 5582, 787-792. http://dx.doi.org/10.1126/science.1060928. – DOI 10.1126/science.1060928

[14] Benso, A. ; Di Carlo, S. ; Di Natale, G. ; Prinetto, P.: Online self-repair of FIR filters. In: *IEEE Design & Test of Computers* 20 (2003), Nr. 3, S. 50–57. http://dx.doi.org/10.1109/MDT.2003.1198686. – DOI 10.1109/MDT.2003.1198686. – ISSN 0740–7475. – printed

[15] Bertozzi, D. ; Benini, L. ; De Micheli, G.: Low power error resilient encoding for on-chip data buses. In: *Proc. Design, Automation and Test in Europe Conference and Exhibition*, 2002, S. 102–109

[16] Bertozzi, D. ; Benini, L. ; De Micheli, G.: Error control schemes for on-chip communication links: the energy-reliability tradeoff. 24 (2005), Nr. 6, S. 818–831. http://dx.doi.org/10.1109/TCAD.2005.847907. – DOI 10.1109/TCAD.2005.847907. – ISSN 0278–0070

[17] Black, J.R.: Electromigration - A brief survey and some recent results. In: *IEEE J ED* 16 (1969), Nr. 4, S. 338–347. – ISSN 0018–9383

[18] Chen, Guoqing: *Design and Modeling of High Speed Global On-Chip Interconnects*, University of Rochester, Diss., 2007

[19] Choi, M. ; Park, N. ; Lombardi, F. ; Kim, Y. B. ; Piuri, V.: Optimal spare utilization in repairable and reliable memory cores. In: *Proc. Records of the 2003 International Workshop on Memory Technology, Design and Testing*, 2003. – ISSN 1087–4852, S. 64–71

[20] Constantinides, K. ; Plaza, S. ; Blome, J. ; Zhang, B. ; Bertacco, V. ; Mahlke, S. ; Austin, T. ; Orshansky, M.: BulletProof: a defect-tolerant CMP switch architecture. In: *Proc. Twelfth International Symposium on High-Performance Computer Architecture*, 2006. – ISSN 1530–0897, S. 5–16

[21] DeHon, André ; Naeimi, Helia: Seven Strategies for Tolerating Highly Defective Fabrication. In: *IEEE Design and Test of Computers* 22 (2005), Nr. 4, S. 306–315. http://dx.doi.org/http://doi.ieeecomputersociety.org/10.1109/MDT.2005.94. – DOI http://doi.ieeecomputersociety.org/10.1109/MDT.2005.94. – ISSN 0740–7475

[22] Fan, C. C. ; Bruck, J.: Tolerating multiple faults in multistage interconnection networks with minimal extra stages. 49 (2000), Nr. 9, S. 998–1004. http:

//dx.doi.org/10.1109/12.869334. – DOI 10.1109/12.869334. – ISSN 0018–9340

[23] Galke, C. ; Grabow, M. ; Vierhaus, H. T.: Perspectives of combining online and offline test technology for dependable systems on a chip. In: *Proc. 9th IEEE On-Line Testing Symposium IOLTS 2003*, 2003, S. 183–187

[24] Ganguly, A. ; Pande, Partha P. ; Belzer, B. ; Grecu, C.: Addressing Signal Integrity in Networks on Chip Interconnects through Crosstalk-Aware Double Error Correction Coding. In: Pande, Partha P. (Hrsg.): *Proc. IEEE Computer Society Annual Symposium on VLSI ISVLSI '07*, 2007, S. 317–324

[25] Grecu, C. ; Pande, P. ; Ivanov, A. ; Saleh, R.: BIST for network-on-chip interconnect infrastructures. In: *VLSI Test Symposium, 2006. Proceedings. 24th IEEE*, 2006, S. 6pp.

[26] Gupta, A. ; Dutt, N.D. ; Kurdahi, F.J. ; Khouri, K.S. ; Abadir, M.S.: Thermal Aware Global Routing of VLSI Chips for Enhanced Reliability. In: *Proc. 9th International Symposium on Quality Electronic Design ISQED 2008*, 2008, S. 470–475

[27] Hao, Li ; Yu, Lixin: A Systematic (48, 32) Code for Correcting Double Errors and Detecting Random Triple Errors. In: *Computational Intelligence and Security, International Conference on* 2 (2008), S. 355–358. http://dx.doi.org/http://doi.ieeecomputersociety.org/10.1109/CIS.2008.161. – DOI http://doi.ieeecomputersociety.org/10.1109/CIS.2008.161. ISBN 978–0–7695–3508–1

[28] Hsieh, Wen-Wen ; Chen, Po-Yuan ; Hwang, TingTing: A bus architecture for crosstalk elimination in high performance processor design. In: *Proc. 4th international conference Hardware/software codesign and system synthesis CODES+ISSS '06*, 2006, S. 247–252

[29] Kothe, R. ; Galke, C. ; Vierhaus, Heinrich T.: A multi-purpose concept for SoC self test including diagnostic features. In: *On-Line Testing Symposium, 2005. IOLTS 2005. 11th IEEE International*, 2005, S. 241–246

[30] Kothe, R. ; Vierhaus, Heinrich T.: Flip-Flops and Scan-Path Elements for Nanoelectronics. In: *Design and Diagnostics of Electronic Circuits and Systems, 2007. DDECS '07. IEEE*, 2007, S. 1–6

[31] Kothe, R. ; Vierhaus, Heinrich T.: Repair Functions and Redundancy Management for Bus Structures. In: *ARCS '07 - 20th International Conference on Architecture of Computing Systems 2007*, 2007

[32] Kretzschmar, C. ; Galke, C. ; Vierhaus, H. T.: A hierarchical self test scheme for SoCs. In: *Proc. 10th IEEE International On-Line Testing Symposium IOLTS 2004*, 2004, S. 37–42

[33] Lajolo, M. ; Reorda, M.S. ; Violante, M.: Early evaluation of bus interconnects dependability for system-on-chip designs. In: Reorda, M.S. (Hrsg.): *Proc. Fourteenth International Conference on VLSI Design*, 2001, S. 371–376

[34] Lala, P. K.: A single error correcting and double error detecting coding scheme for computer memory systems. In: *Proc. 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2003. – ISSN 1063–6722, S. 235–241

[35] Lehtonen, Teijo ; Plosila, Juha ; Isoaho, Jouni: On Fault Tolerance Techniques towards Nanoscale Circuits and Systems / TUCS. 2005. – Forschungsbericht

[36] Li, Jin-Fu ; Yeh, Jen-Chieh ; Huang, Rei-Fu ; Wu, Cheng-Wen ; Tsai, Peir-Yuan ; Hsu, Archer ; Chow, Eugene: A Built-In Self-Repair Scheme for Semiconductor Memories with 2-D Redundancy. In: *itc* 00 (2003), S. 393. `http://dx.doi.org/http://doi.ieeecomputersociety.org/10.1109/TEST.2003.1270863`. – DOI http://doi.ieeecomputersociety.org/10.1109/TEST.2003.1270863. – ISSN 1089–3539

[37] Lienig, J. ; Jerke, G.: Electromigration-aware physical design of integrated circuits. In: *Proc. 18th International Conference on VLSI Design*, 2005. – ISSN 1063–9667, S. 77–82

[38] Loh, Gabriel H. ; Xie, Yuan ; Black, Bryan: Processor Design in 3D Die-Stacking Technologies. 27 (2007), Nr. 3, S. 31–48. http://dx.doi.org/10.1109/MM.2007.59. – DOI 10.1109/MM.2007.59. – ISSN 0272–1732

[39] Lu, Z. ; Lach, J. ; Stan, M. R. ; Skadron, K.: Improved thermal management with reliability banking. 25 (2005), Nr. 6, S. 40–49. http://dx.doi.org/10.1109/MM.2005.114. – DOI 10.1109/MM.2005.114. – ISSN 0272–1732

[40] Meindl, J.D.: Interconnect opportunities for gigascale integration. 23 (2003), Nr. 3, S. 28–35. http://dx.doi.org/10.1109/MM.2003.1209464. – DOI 10.1109/MM.2003.1209464. – ISSN 0272–1732

[41] Montanes, R. R. ; Gyvez, J. P. ; Volf, P.: Resistance characterization for weak open defects. In: *IEEE Design & Test of Computers* 19 (2002), Nr. 5, S. 18–26. http://dx.doi.org/10.1109/MDT.2002.1033788. – DOI 10.1109/MDT.2002.1033788. – ISSN 0740–7475

[42] Muddu, S. ; Sarto, E. ; Hofmann, M. ; Bashteen, A.: Repeater and interconnect strategies for high-performance physical designs. In: *Proc. XI Brazilian Symposium on Integrated Circuit Design*, 1998, S. 226–231

[43] Murali, S. ; Theocharides, T. ; Vijaykrishnan, N. ; Irwin, M. J. ; Benini, L. ; De Micheli, G.: Analysis of error recovery schemes for networks on chips. In: *IEEE Design & Test of Computers* 22 (2005), Nr. 5, S. 434–442. http://dx.doi.org/10.1109/MDT.2005.104. – DOI 10.1109/MDT.2005.104. – ISSN 0740–7475

[44] Narasimhan, A. ; Kasotiya, M. ; Sridhar, R.: A low-swing differential signalling scheme for on-chip global interconnects. In: *Proc. 18th International Conference on VLSI Design*, 2005. – ISSN 1063–9667, S. 634–639

[45] Naseer, Riaz ; Draper, Jeff: Parallel Double Error Correcting Code Design to Mitigate Multi-Bit Upsets in SRAMs. In: *Proceedings of the European Solid-State Circuits Conference*, 2008, S. 222–225

[46] Nicolaidis, M. ; Achouri, N. ; Anghel, L.: Memory built-in self-repair for nanotechnologies. In: *Proc. 9th IEEE On-Line Testing Symposium IOLTS 2003*, 2003, S. 94–98

[47] Pande, P.P. ; Ganguly, A. ; Feero, B. ; Belzer, B. ; Grecu, C.: Design of Low power & Reliable Networks on Chip through joint crosstalk avoidance and forward error correction coding. In: *Defect and Fault Tolerance in VLSI Systems, 2006. DFT '06. 21st IEEE International Symposium on*, 2006, S. 466–476

[48] Pasricha, Sudeep ; Dutt, Nikil: *On-Chip Communication Architectures System on chip interconnect*. Morgan Kaufmann Publishers, 2008. – 522 S.

[49] Patel, K.N. ; Markov, I.L.: Error-correction and crosstalk avoidance in DSM busses. 12 (2004), Oct., Nr. 10, S. 1076–1080. http://dx.doi.org/10.1109/TVLSI.2004.827565. – DOI 10.1109/TVLSI.2004.827565

[50] Pompl, T. ; Schlunder, C. ; Hommel, M. ; Nielen, H. ; Schneider, J.: Practical aspects of reliability analysis for IC designs. In: *Proc. 43rd ACM/IEEE Design Automation Conference*, 2006. – ISSN 0738–100X, S. 193–198

[51] Rochel, S. ; Nagaraj, N. S.: Full-chip signal interconnect analysis for electromigration reliability. In: *Proc. IEEE 2000 First International Symposium on Quality Electronic Design ISQED 2000*, 2000, S. 337–340

[52] Rodriguez-Montanes, R. ; Arumi, D. ; Figueras, J. ; Einchenberger, S. ; Hora, C. ; Kruseman, B. ; Lousberg, M. ; Majhi, A. K.: Diagnosis of Full Open Defects in Interconnecting Lines. In: *Proc. 25th IEEE VLSI Test Symposium*, 2007. – ISSN 1093–0167, S. 158–166

[53] Rossi, D. ; Cavallotti, S. ; Metra, C.: Error correcting codes for crosstalk effect minimization [system buses]. In: *Proc. 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2003. – ISSN 1063–6722, S. 257–264

[54] Rossi, D. ; Metra, C. ; Nieuwland, A.K. ; Katoch, A.: New ECC for crosstalk impact minimization. In: *IEEE Design & Test of Computers* 22 (2005), July–Aug., Nr. 4, S. 340–348. http://dx.doi.org/10.1109/MDT.2005.91. – DOI 10.1109/MDT.2005.91

[55] Rossi, D. ; Omana, M. ; Toma, F. ; Metra, C.: Multiple transient faults in logic: an issue for next generation ICs? In: Omana, M. (Hrsg.): *Proc.*

*20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems DFT 2005*, 2005, S. 352–360. – printed

[56] Sainarayanan, K.S. ; Raghunandan, C. ; Srinivas, M.B.: Delay and Power Minimization in VLSI Interconnects with Spatio-Temporal Bus-Encoding Scheme. In: Raghunandan, C. (Hrsg.): *Proc. IEEE Computer Society Annual Symposium on VLSI ISVLSI '07*, 2007, S. 401–408

[57] Semerdjiev, B. ; Velenis, D.: Optimal Crosstalk Shielding Insertion along On-Chip Interconnect Trees. In: *Proc. th International Conference on VLSI Design Held jointly with 6th International Conference on Embedded Systems*, 2007. – ISSN 1063–9667, S. 289–294

[58] Shedletsky, J. J.: Error Correction by Alternate-Data Retry. C-27 (1978), Nr. 2, S. 106–112. `http://dx.doi.org/10.1109/TC.1978.1675044`. – DOI 10.1109/TC.1978.1675044. – ISSN 0018–9340

[59] Sridhara, S.R. ; Ahmed, A. ; Shanbhag, N.R.: Area and energy-efficient crosstalk avoidance codes for on-chip buses. In: *Proc. IEEE International Conference on Computer Design: VLSI in Computers and Processors ICCD 2004*, 2004, S. 12–17

[60] Sridhara, S.R. ; Shanbhag, N.R.: Coding for system-on-chip networks: a unified framework. In: *DAC'04*, 2004, S. 103–106

[61] Sridhara, S.R. ; Shanbhag, N.R.: Coding for reliable on-chip buses: fundamental limits and practical codes. In: *Proc. 18th International Conference on VLSI Design*, 2005. – ISSN 1063–9667, S. 417–422

[62] Tsuchiya, Akira: *A Study on Modeling and Design Methodology for High-Performance On-Chip Interconnection*, Kyoto University, Diss., November 2005

[63] Xuan, Xiangdong: *ANALYSIS AND DESIGN OF RELIABLE MIXED-SIGNAL CMOS CIRCUITS*, Georgia Institute of Technology, Diss., December 2004

[64] Xuan, Xiangdong ; Singh, A.D. ; Chatterjee, A.: Reliability evaluation for integrated circuit with defective interconnect under electromigration. In: *Proc. Fourth International Symposium on Quality Electronic Design*, 2003, S. 29–34

[65] Yu, Hao ; He, Lei: Staggered twisted-bundle interconnect for crosstalk and delay reduction. In: He, Lei (Hrsg.): *Proc. Sixth International Symposium on Quality of Electronic Design ISQED 2005*, 2005, S. 682–687

[66] Zhong, G. ; Koh, C.-K. ; Roy, K.: A twisted-bundle layout structure for minimizing inductive coupling. In: *Proc. ICCAD-2000 Computer Aided Design IEEE/ACM International Conference on*, 2000, S. 406–411

# First author publications


[1] T. Koal, D. Scheit, and H. T. Vierhaus. Reliability estimation process. *Digital Systems Design, Euromicro Symposium on*, 0:221–224, 2009.

[2] D. Scheit and H. T. Vierhaus. Fehlertolerante Busse basierend auf Codes und Selbstreparatur. In *Zuverlässigkeit und Entwurf: 2. GMM/GI/ITG-Fachtagung vom 29. September bis 1. Oktober 2008 in Ingolstadt*, page 157.158, 2008.

[3] D. Scheit and H. T. Vierhaus. Fehlertolerante integrierte Verbindungsstrukturen. In *Dresdner Arbeitstagung Schaltungs- und Systementwurf*, pages 119–124. Fraunhofer Institut Integrierte Schaltungen, 2008.

[4] D. Scheit and H. T. Vierhaus. Fehlertoleranz und Selbstreparatur von Verbindungsstrukturen auf Socs. In *TuZ 2008, 20. Workshop für Testmethoden und Zuverlässigkeit von Schaltungen und Systemen*, pages 121–126, 2008.

[5] D. Scheit and H. T. Vierhaus. Zentrale und dezentrale selbstreparatur von bussen. In *TuZ 2009, 21. Workshop für Testmethoden und Zuverlässigkeit von Schaltungen und Systemen*, pages 55–60, 02 2009.

[6] D. Scheit and H. T. Vierhaus. Zentrale und dezentrale Selbstreparatur von Bussen. In *EDA 2009. Electronic Design Automation workshop*, 05, 2009.

# Contributing author publications


[1] T. Koal, D. Scheit, and H. T. Vierhaus. Möglichkeiten und Grenzen der Selbstreparatur für Logik. In *Proc. 2. GMM/GI/ITG- Arbeitstagung Zuverlässigkeit und Entwurf (ZuE)*, 2008.

[2] T. Koal, D. Scheit, and H. T. Vierhaus. A comprehensive scheme for logic self repair. In *Proc. IEEE Conf. On Signal Processing Architectures (SPA)*, 2009.

[3] T. Koal, D. Scheit, and H. T. Vierhaus. A concept for logic self repair. In *Proc. 12th Euromicro Conference on Digital System Design (DSD)*, 2009.

[4] T. Koal, D. Scheit, and H. T. Vierhaus. A scheme of logic self repair including local interconnects. In *Proc. IEEE DDECS*, 2009.

[5] T. Koal, D. Scheit, and H. T. Vierhaus. Selbstreparatur durch Regularisierung von Logik-Schaltungen. In *Proc. 3. GMM/GI/ITG Fachtagung Zuverlässigkeit und Entwurf (ZuE)*, 2009.

[6] T. Koal, D. Scheit, and H. T. Vierhaus. Schwachstellen und Engpässe bei Verfahren zur Fehlerkompensation und Selbstreparatur für hochintegrierte Schaltungen. In *Proc. 4. GMM/GI/ITG- Arbeitstagung Zuverlässigkeit und Entwurf (ZuE)*, 2010.

[7] R. Kothe, D. Scheit, and H. T. Vierhaus. Angepasste Fehlerdiagnose für die Selbstreparatur in logischen Schaltungen. In G. Elst, editor, *Proc. Dresdner Arbeitstagung für Schaltungs- und Systementwurf (DASS)*, May 2008.