UNIVERSITY OF SOUTHAMPTON

FACULTY OF PHYSICAL SCIENCES AND ENGINEERING

Electronics and Computer Science

An Investigation into Ageing-Resilient Processor Design

by

Haider Muhi Abbas

Thesis for the degree of Doctor of Philosophy

October 2018

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF PHYSICAL SCIENCES AND ENGINEERING
Electronics and Computer Science

Doctor of Philosophy

AN INVESTIGATION INTO AGEING-RESILIENT PROCESSOR DESIGN

by Haider Muhi Abbas

Microprocessors offer the ability for fast and reliable processing and are indispensable in many general and specific computing systems in different disciplines. In the past decades, CMOS technology size has progressively decreased, which has resulted in devices with smaller areas and lower power consumptions. However, side effects like ageing become more critical at smaller technology sizes, and this bottleneck challenges further improvements in CMOS technology. The main challenge of advanced technologies is at the physical level of the CMOS device, in which the device-level response time degrades over time due to higher-level stress (e.g. workload from a software). Advanced materials that cope with ageing effects could take years to be developed; therefore, this thesis focuses on the stress that is propagated from software-level and explores techniques to mitigate the ageing behaviour by reducing stress from its source.

The first part of the thesis targets single-core processors. We present a novel technique to mitigate the bias temperature instability (BTI) ageing effects on microprocessors. After intensive analysis of ageing and its sources from the program level to the device level, we found that an application may stress the critical paths of a circuit in a way that may have half of the nodes always negative BTI stressed, while the second half are positive BTI stressed. To mitigate this behaviour, we propose an application-level solution to reverse the stress and put the processor nodes into a relaxed mode.

In the second part of the thesis, we investigate how multi-core processors could be used to mitigate BTI ageing effects by using the fact that idleness is adverse to a processor core at high temperatures. We show that it is necessary to run an anti-ageing program on the idle core in order to relax the stress or proactively avoid the stress generated by a high-level application by analysing the workload and develop a learning model to estimate the stress and its distribution on the multi-core processor. Subsequently, this model is used in a frequency regulator to dynamically adjust the frequencies of the core based on the estimated stress. Results show that by applying the proposed techniques, the ageing stress could be reduced by a half.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Haider Muhi Abbas , declare that the thesis entitled *An Investigation into Ageing-Resilient Processor Design* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as: [1], [2] and [3]

Signed:...

Date:..

Acknowledgements

Thanks to my supervisor, Professor Mark Zwolinski, for all the advice and discussions, which have made me a much better researcher. Thanks to him for his insightful comments and constructive feedbacks. Also, I would like to thank my supervisor, Basel Halak, for all the time he spent with me to supervise my work and his openness which made my PhD study less stressful. He has always been welcoming when I need intensive discussions about my work.

I would like to thank my sponsor The Higher Committee for Education Development in Iraq (HCED Iraq) for the financial support and orientation. Without their support, I would not be able to come the UK and study. In the future, I wish I will be able to contribute for rebuilding the educational system and improve the research standards in Iraq and accomplish the aim that you sent me for it.

Not to forget my colleagues and friends. I would like to thank my lovely colleagues in the lab for being friendly and positive most of the time, which has made my PhD journey much more interesting. Thanks to Ibrahim Bello, Illani Mohd Nawi, Gaole Sai, Shengyu Duan and Mohd Syafiq Mispan. Also, I would like to thank my friends Ali Alameer, Zeyad Alkhafajy, Mahdi Alajwadi Ahmed Al-Zubaydi, Omar Adnan, Ahmed Allami and Ahmed Yashar for their advice on academic research and for the time we spent outside the university that helped me to find some relief from the more stressful days.

Last but not least I would like to thank my family. From the bottom of my heart I would like to thank my wife, Farah, for all the support, love and sacrifice. It was a long journey and I will never forget your support taking care of our beloved sons, Anas and Amjad, while studying at the same time. Back to my home country, thanks to my family especially my father and mother - I will never have reached this milestone without their support. All the love to you, thanks.

Nomenclature

AHL	Adaptive Hold Logic
ALU	Arithmetic Logic Unit
ASICs	Application Specific Integrated Circuits
ATPG	Automatic Test Pattern Generation
BTI	Bias Temperature Instability
CLK	Clock signal
DNM	Dynamic Wear-out/NBTI Management
DRM	Dynamic Reliability Management
EDA	Electronics Design Automation
EM	Electromigration
FFs	Flip-Flops
HCI	Hot-Carrier Injection
IoT	Internet of Things
IPC	Instructions Per Cycle
ISA	Instruction Set Architecture
IVC	Input Vector Control
LUTs	Look-up Tables
M-IVC	Multiple Input Vector Control
MOSFETS	Metal-Oxide-Semiconductor Field-Effect Transistors
MPSoC	Multiprocessor System on Chip
MSE	Mean Squared Error
NBTI	Negative-Bias Temperature Instability
NMOS	N-channel MOSFET (metal-oxide-semiconductor field-effect transistor)
NN	Neural Network
NOP	No-operation Instruction
PBTI	Positive-Bias Temperature Instability
PCPs	Potential Critical Paths or NBTI Critical Paths
PMOS	P-channel MOSFET (metal-oxide-semiconductor field-effect transistor)
RD	Reaction-Diffusion ageing model
RMG	Replacement Metal Gate Technology
SAA	Static Ageing Analysis
SBST	Software-Based Self-Test

SNM	Static Noise Margin
$SP(0)$	The probability of the gate-level signal being stressed with logic zero
$SRAM$	Static Random-Access Memory
STA	Static Timing Analysis
$TDDB$	Time-Dependent Dielectric Breakdown
$TSMC$	Taiwan Semiconductor Manufacturing Company
VCD	Value Change Dump
V_{dd}	Supply Voltage

Chapter 1

Introduction

Due to need for faster digital devices, transistors have been scaled over the years. These digital devices including microprocessors can now do more calculation in less time than before due to improvement in transistor performance with the shrinking technology sizes. However, scaling comes with reliability issues including ageing in which the transistor switches slower by the passage of time of the device operation. Thus, this transistor-level degradation could be seen as processor-level performance degradation (e.g., for 45nm technology size, processor could degrade in range 2% to 15% depending on the workload applied [4]). Transistor ageing is caused by wear-out mechanisms such as Negative Bias Temperature Instability (NBTI), Hot Carrier Injection (HCI), and Time-Dependent Dielectric Breakdown (TDDB). These phenomena result in performance degradation and reliability problems in the smallest unit of the microprocessor which is a complementary metal-oxide-semiconductor (CMOS) transistor. The first discovery of threshold voltage instability was in the 1960s, when it was discovered that both bias and temperature affect the threshold voltage of MOS transistors. Since then, Bias Temperature Instability (BTI) has remained largely an unimportant reliability concern [5]. However, with the introduction of nitrogen atoms into the gate oxide of CMOS transistor, negative BTI has continued to attract attention from both academia and industry [5, 6], and is now considered to be the most important ageing mechanism in nano-scale technology devices [7], [8]. Nowadays, Companies that produce devices (e.g. Smart phones, Tablets and Laptops) expecting ageing degradation in their devices but they may ignore the issue for encouraging their customers to buy new devices. However, with shrinking technology sizes, the ageing issue is progressive that makes the amount of degradation unacceptable in the future by the end users. So, this thesis trying to tackle the ageing issue as it is expected to have more attention in the near future.

Many mitigation techniques for NBTI have been proposed at different abstraction levels in recent years. At high levels of the abstraction hierarchy, a variety of protection mechanisms has proposed that may increase reliability even further when moving to the higher level of abstraction. These techniques depend on existing models, which predict the impact of NBTI on overall system performance, assume a generic signal probability of input signals of 50%. Such an assumption can cause misleading predictions about how the circuit's performance is going to degrade in time, and more importantly, which parts of the system will be most affected.

1.1 Research Motivation

The hypothesis of this work is that ageing stress can be mitigated from the application-level as it is being generated from source. However, the complexity of the processor design levels makes it difficult to model the ageing at the system level. Usually, ageing degradation needs to take into account different contributions at the lower levels of abstraction of time-dependent variations (e.g., signal probability, switching activity, temperature and supply voltage). There are many different approaches to control these ageing contributions using high-level mitigation techniques. Singh et al. [9] and Huard et al. [10] propose the Dynamic Wear-out/NBTI Management (DNM) technique, which reduces the power consumption by running the circuit with the least possible supply voltage and changes the supply voltage periodically based on readings from ageing sensors. However, the main challenge of this approach is that the accuracy of the sensors and the area overheads may exceed design constraints. In Tenentes et al. [11], techniques proposed to utilise existing power-gating circuitry to sense BTI-induced degradation to reduce the area overhead. However, since these sensors may be under stress as well, their readings could become less accurate over time.

Another approach to control ageing is to reduce the stress rather than predicting and adapting to it accordingly. Controlling the temperature and the switching activity of the processor (e.g., the thermal stress of the BTI effect) could be achieved by using established techniques used in reducing dynamic power consumption. In addition, the signal probabilities or the duty cycles (e.g., the electrical stress of the BTI effect) have been studied extensively in the literature. For example, signal probabilities can be controlled from the primary inputs of the circuit (Input Vector Control) or during the synthesis process by hiding nodes with high signal probability of being zero or changing the pin order of the gates on the critical paths [12, 13, 14].

Another method was proposed in Kiamehr et al. [15], where the rising and falling gate delays were balanced based on an extended standard cell library considering the signal probabilities of the nets during the gate mapping process. The signal probability of zero SP(0) is the percentage of having 0's on an input logic gate that may represented as a electrical stress on the PMOS transistor with negative bias voltage. However, the signal probability of any path in the circuit cannot be controlled, as there are always nodes that will be stressed if we try to relax other nodes due to the nature of CMOS circuit that have complementary switching behaviour. Karimi et al. [16] claim that the core of the processor can be put into a stressed state by executing programs to age the circuit; however, they did not consider the hidden nodes inside compound logic gates, which makes the analysis inaccurate.

Thus, a cross-layers analysis is required considering the stress from real workload mapped from the software-level to the transistor-level of the processor. Based on the analyses, mitigation techniques need to be proposed taking into consideration that the proposed techniques not required any intrusive and technology-dependency at the design stage of the processor. So, instead of reacting to the ageing behaviour, proactive techniques are required by reducing the BTI ageing stress.

1.2 Aim and Objectives

The aim of this work is to put the microprocessor into a low-stress ageing state. Thus, four objectives have been defined to accomplish this goal:

- Develop a tool that helps to propagate the high-level stress generated by an application to the low-level stress represented by signal probabilities at the gate-level of the microprocessor. Through this approach, the source of BTI stress on the physical-level is identified from the workload at the application-level.
- Identify the signal probability of the combinational logic circuits of the microprocessor and investigate the most vulnerable parts and the most critical stress state imposed on these units. Subsequently, we will propose a high-level technique for reducing ageing stress.
- The ageing stress of the memory units need to be identified and analysed to measure the relaxing and stress state in these units with respect to high-level workloads. We will then propose a technique for reversing stress state toward relax state.
- After having clear understanding about the stresses on the main components of the processor (combinational circuits and the memories), we will design a learning model to estimate the ageing stress and mitigate it before it is applied on the multi-core processor in which no core is put under static ageing stress.

1.3 Thesis Overview and Contributions

The structure of the remainder of the thesis is as follows:

Chapter 2 presents a background review about general digital system issues and transistor ageing factors in the first section of chapter and then NBTI modelling at the transistor and gate level is presented in the second section of chapter. After that, NBTI mitigation techniques are presented in the third section of the chapter.

Chapter 3 provides an analysis of signal probability for identifying NBTI critical paths and including the signal probability in the technology library.

Chapter 4 proposes a novel mitigation technique using an anti-ageing program to reverse the stress on the critical paths of the combinational logic circuit of the processor to alleviate both NBTI and PBTI effects.

Chapter 5 presents an ageing analysis and mitigation for the memory units of the processor, such as flip-flops and SRAMs. We also propose a software solution to mitigate the ageing on the register file of the processors.

Chapter 6 describes the possibility of reliability improvement to mitigate the ageing stress within multi-core processors and proposes a proactive technique for mitigating the stress by controlling the temperature and idleness of the processor core.

Chapter 7 concludes the thesis and present a number of areas for further research.

Parts of this thesis have been published as journal and conference papers. The list of publications are as follows:

- Abbas, Haider Muhi, Basel Halak, and Mark Zwolinski. "BTI mitigation by anti-ageing software patterns." Microelectronics Reliability 79 (2017): 79-90.
- Abbas, Haider, Mark Zwolinski, and Basel Halak. "Static Aging Analysis Using 3-Dimensional Delay Library." ERMAVSS@ DATE. 2016.
- Abbas, Haider Muhi, Mark Zwolinski, and Basel Halak. "An application-specific NBTI ageing analysis method." CMOS Variability (VARI), 2015 International Workshop on. IEEE, 2015.

Chapter 2

Background and Related Works

In this chapter, a background on the reliability issues including CMOS ageing of digital systems will be presented. We will also provide a mathematical model for the ageing of CMOS devices. Finally, we will review existing ageing mitigation techniques and identify the research gaps in the literature.

2.1 Digital Systems Issues

Digital systems refer to any software or hardware-based computing devices. The minimum requirement that enables us to solve a digital problem is the hardware circuit that is mostly designed with complementary metal oxide semiconductor (CMOS) technology. The basic element of CMOS circuit is the transistor that defines the performance of the general digital system at the higher level of abstraction (e.g., the system level). New technologies try to shrink the sizes of these transistors in order to improve the performance; however, as we shall come to see, there are bottlenecks limiting this trend. The scaling process of device geometries produces several issues including increase in temperature, power consumption and limiting reliability objectives. To optimise one issue (e.g., reliability) it is important to define the other design requirements (e.g., area-, performance- and power-cost) as they are correlated to each other [17]. In this section, the general definitions of the most important issues in CMOS devices are highlighted.

The increasing popularity of mobile devices such as cellular phones and tablets forces system designers into a low power domain. Thus, extending the operational times of these portable systems is limited by the capacity of the batteries and is a vital research area in digital systems. In addition, increasing power consumption correlated with increasing the temperature of the digital devices because increasing the power consumption would increase the temperature due to dynamic switching of the transistors [18], also, increasing the temperature of the devices would need to have a cooling part that would consume more power. So saving energy is one of the main targets of designing these systems. There are many optimisation techniques available for optimising power consumption at different levels of abstraction during the design flow from low level (e.g., transistor sizing [19]) to high level of abstraction (e.g., dynamic voltage and frequency scaling [20]).

Another important issue is the reliability of the digital systems because the consequences of unreliable digital systems design could be severe. For example, in 2013 EETimes reported that, perhaps a single fault in a memory bit caused a car accident which led to the death of a person [21]. Reliability techniques can be applied at different abstraction levels. Each level of granularity has its own techniques for boosting reliability. At lower levels, reliability can be provided by using technology innovations or by putting in margins (over-design). Moving up the abstraction hierarchy, other techniques, such as incorporating redundancies are available. The variety of protection mechanisms increases even further when moving to the software level [22]. In general, the number of faults per device increases with every new technology [23]. Nowadays, there are three main issues affecting the reliability of CMOS devices.

The first issue is the variability or non-idealities in process, supply voltage and temperature (PVT). Variations have always been an issue in integrated circuit (IC) design and have a direct impact on performance. PVT variations affect a transistor delay and consequently, the timing of the logic circuit [24]. Secondly, single-event effects (SEEs), also called transient faults or soft errors can cause errors in computation as single-event transients (SETs) and can corrupt data as single-event upsets (SEUs); however, they do not fundamentally damage the system and not considered as a lifetime reliability concern [25]. The source of these errors is from either electrical noise or external radiation, rather than manufacturing associated defects. In safety critical domains, such as automotive, aeronautics, and industrial automation, when a single bit is flipped as a result of a transient error, the impact can be severe. Therefore, in the design phase it is important to improve the resilience of a circuit to all possible transient errors [26]. Thirdly, ageing or time-dependent variations in CMOS devices represent a challenge to the design of ICs, along with power consumption, performance and other reliability issues (e.g., process variations, single event upsets (SEU) and Electromigration) [27]. Ageing-degradation can manifest itself as soft errors that become hard errors, bringing the system to a state where timing constraints are violated, and ultimately, the system fails to function properly. The mechanisms at the device level responsible for ageing degradation include: positive/negative bias temperature instability (PBTI/NBTI), hot carrier injection (HCI) and time-dependent dielectric breakdown (TDDB). These mechanisms manifest themselves as changes in the device threshold voltage, carrier mobility, and the insulating properties of gate dielectrics [28]. These phenomena result in a timing delay activated under temperature, stress and voltage fluctuations. However, transistor ageing is mostly caused by BTI, and is one of the main unreliability causes at nano-scale technology devices [7, 8].

2.2 Microprocessor Performance Degradation Factors

A transistor that is ageing gradually causes delay degradation in the logic gates due to an increase in the threshold voltage of the transistors. The threshold voltage shift leads to a decrease in the on-state-drain current. Ultimately, the circuit starts to exceed the timing limits and fails due to timing violations. Generally, transistors age due to NBTI, PBTI and HCI. NBTI affects PMOS transistors, while PBTI affects NMOS transistors. NBTI and PBTI phenomena cause the threshold voltage of the transistor to increase over time making less current to be passed between the source and drain of the transistor. HCI occurs when there is a transition in the gate-source of the transistor and results in a threshold voltage increase. Ageing factors and their models are described in more detail in the following subsections.

2.2.1 Hot Carrier Injection (HCI)

Interface traps are generated at the silicon/gate-oxide interface near the drain when there is a switching in the gate of the NMOS device that can eventually degrade device parameters [29]. Usually, hot carrier injection (HCI) happens in an NMOS device during gate input switching from low to high. HCI is highly switching-activity dependent unlike NBTI which is signal-probability dependent. Furthermore, the recovery in HCI is considered negligible, which makes HCI worse under dynamic stress. When the circuit is in the active state, a adaptive method is used by regulating the voltage and/or the frequency of the circuit based on measurement taken from sensors to measure the impact of HCI by measuring the drain current shift [30]. Finally, HCI has become less prominent than BTI with the reduction of supply voltages, but remains a serious concern due to the increase in device densities in deep sub-micron nodes [31].

2.2.2 Time-Dependent Dielectric Breakdown (TDDB)

Time-dependent dielectric breakdown (TDDB) is a physical process of SiO_2-based dielectrics breaking down with time, which happens when a constant electric field is applied to the dielectric. Thus, breakdown means that the dielectric transforms from an insulating state to a state that is more conductive. It was thought that the TDDB effect is a hard breakdown only as it breaks down the device and makes the oxide layer fully conducting, which leads to a catastrophic failure of the circuit [32, 33]. Furthermore, soft breakdown has emerged as another mode of TDDB. In this mode, the insulating properties of the SiO_2 layer degrade aggressively, but the device remains functional [32]. However, TDDB is still an open topic in semiconductor physics and its effect on digital devices is still not completely understood.

2.2.3 Bias Temperature Instability (BTI)

BTI is generated due to thermal and electrical stress leading to a change in the physical characteristics of the transistor by generation interface traps at the channel and dielectric. BTI in PMOS transistors is referred to as negative BTI (NBTI), since the gates of PMOS are negatively biased with respect to the source (i.e., $V_{gs} = -V_{dd}$). Meanwhile, BTI for NMOS transistors is referred to as positive BTI (PBTI), since the gates of PMOS are positively biased with respect to the source (i.e., $V_{gs} = V_{dd}$). Both NBTI and PBTI have the same consequences (i.e., they both increase the threshold voltages and decrease the driving currents); however, both have different physical causes. It is generally accepted that NBTI degradation in SiO_2 and High-κ dielectric process are due to dielectric interface traps. Besides, PBTI was negligible before introducing High-κ MOSFETS technology. In High-κ technology, PBTI degradation is a result of build-up of negative charges in the High-κ layer. Although High-κ technology is affected by both NBTI and PBTI degradations, NBTI is much more dominant according to results from replacement metal gate (RMG) technology [34, 35].

In this work, we are primarily concerned with the mitigation of Bias Temperature Instability (BTI). BTI has two different phases:

- Stress: Interface traps are generated at the interface of the substrate and gate oxide layers due to electrical stress (i.e., negative bias for PMOS and positive bias for NMOS) that leads to breaking of some of the Si-H or Si-O bonds as shown in Figure 2.1(a). Consequently, the threshold voltage of the transistor increases over time.

- Relaxation/Recovery: some of the generated traps are removed from the interface as shown in Figure 2.1(b). However, the relaxation phase cannot completely compensate for the effect of the stress phase and therefore the overall effect of BTI is degradation in the threshold voltage of each transistor. The amount of degradation depends on the ratio between the stress period and the total operating period (i.e., the duty cycle or the signal probability at the gate level).

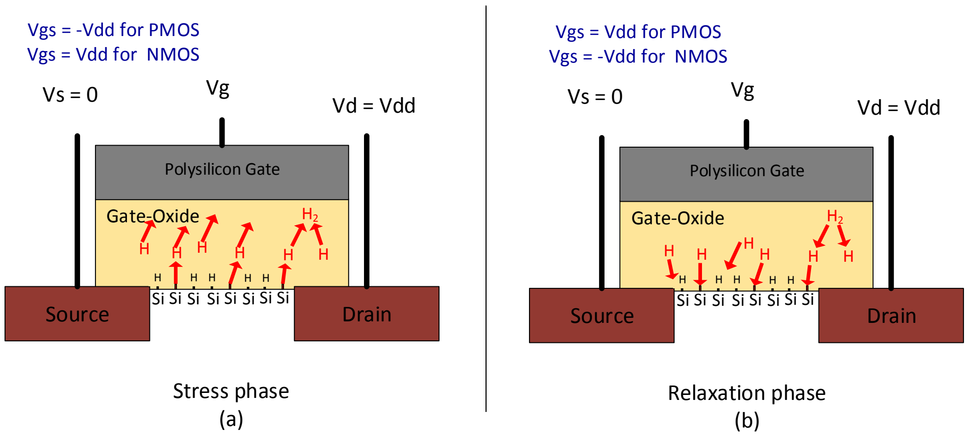


Figure 2.1: Illustration of (a) Stress Phase: Si-H bonds breaking, H and H_2 diffusions towards poly gate (b) Relaxation Phase: H and H_2 diffusions towards the Silicon/Gate-Oxide interface [29].

The characteristics of NBTI effects are highlighted as follows:

- NBTI is one of the most critical issues in nano-scale CMOS devices that affects system reliability. Presently, NBTI is gaining more attention for technologies beyond 45nm [36, 37].
- NBTI stress is increasing with increasing the electrical stress on the PMOS transistors of the logic gates [38, 39, 40].
- NBTI stress increases at high temperatures [38, 41, 42].
- NBTI recovers partially if the stress conditions are removed. In [43], for 0.13 μm technology size, the thickness of the gate oxide was approximately 22 Angstroms, gate length 0.13 μm, the temperature was $105°C$, it was found that up to 40% of the degradation is recovered after removing the stress.
- NBTI degradation could be AC (dynamic, alternating between stress and recovery times) or DC (static, continuous stress) [44].
- NBTI recovers rapidly after removing the stress and this recovery slows by time until it saturates making any measurement of the delay degradation inaccurate [45, 46].
- The physics behind NBTI is still a controversial topic but the results from both reaction-diffusion (RD) and defect-centric models that will be described in the next section match the experimental results of the degradation [38, 44].
- NBTI ageing could jeopardise the reputation of the company but at the same time, it could be an intended process. After manufacturing, the application could be used to stress the device (malicious ageing) either from the user, to retain

warranty of a used device and ask for replacement, or from the manufacturing company to encourage buying new released devices [16].

2.2.3.1 Transistor NBTI Degradation Model

This section presents the mathematical model of the NBTI effect and shows how to model the degradation of the low-level device parameters into transistor level. At device level, many models have been proposed as predictive models to simulate the real degradation in transistor performance. Until now, there is no theory or model universally accepted; therefore, all information is based on accepted experimental results. Reaction-diffusion (RD) is one of the most prevalent NBTI models in the literature [13, 29, 46, 47, 48]. Another NBTI model is the defect-centric model presented in [49], which differs from the RD model, as the defect-centric model tries to model the discrete behaviour of individual traps, while the RD model tries to describe the continuous behaviour of the trap as a mass rather than discrete particles. Both models have successfully explained NBTI experimental results [39].

Many developments have been proposed to increase the accuracy of these models. The short term and long term models for the RD model have been presented in detail in [13]. In general, NBTI causes time-dependent degradations that leads to deviations from the time-zero state of the physical parameters. These parameters include drain currents (linear and saturation) and threshold voltage [46, 50]. We reproduced the results from RD model to understand the relevant parameters of stress of the NBTI effect.

2.2.3.2 Short Term Cycle to Cycle Model of Degraded V_{th}

The amount of threshold voltage (V_{th}) degradation per cycle could be predicted using the RD-model presented in [48] that considers whether the stress is static (no recovery) or dynamic (recovery period heals some of the degradation). The following model is for the cycle-to-cycle V_{th} degradation for both the static and the dynamic operation modes.

- Static operation mode: here it assumed that the device is under full stress during the simulated period.

$$\Delta V_{th} = A\Big((1+\delta)t_{ox} + \sqrt{C(t-t_0)}\Big)^{2n} \tag{2.1}$$

$$A = \left(\frac{qt_{ox}}{\epsilon_{ox}}\right)\sqrt{K^3 C_{ox}(V_{gs}-V_{th})(\exp\frac{E_{ox}}{E_0})^2} \tag{2.2}$$

- Dynamic operation mode: here it assumed that there is a recovery period during the operation mode

$$\text{Stress}: \Delta V_{th} = \left(K_v(t-t_0)^{0.5} + \sqrt[2n]{\Delta V_{th0}}\right)^{2n} \tag{2.3}$$

$$\text{Recovery}: \Delta V_{th} = V_{th0}(1 - \frac{2\xi_1 t_{ox} + \sqrt{\xi_2 C(t-t_0)}}{2t_{ox} + \sqrt{Ct}} \tag{2.4}$$

where n is the time exponent, $n = 1/6$ for For H_2 diffusion based models, n = 1/4 for H based models (e.g., n is equal to 1/6 for the 90nm technology). q is the electron charge, V_{gs} is the transistor gate voltage, t_{ox} is the oxide thickness, C_{ox} is the oxide capacitance per unit area, $(t - t_0)$ is the stress or recovery duration starts at time t_0 and ends at time t, T is the temperature. The electric field $E_{ox} = (V_{gs} - V_{th})/t_{ox}$. K_v is a function of the electrical field, temperature and carrier concentration (C). k is the Boltzmann constant. The rest of the parameters are fitting parameters, for example for 90nm technology node, the parameters are given in Table 2.1.

$\boldsymbol{K_v}$	$\left(\frac{qt_{ox}}{\epsilon_{ox}}\right)^3 K^2 C_{ox}(V_{gs}-V_{th})\sqrt{C}exp\left(\frac{2E_{ox}}{E_0}\right)$		
$\boldsymbol{C}$	$T_0^{-1}\exp(-\frac{E_a}{kT})$		
$\boldsymbol{E_a}$	0.49	$\boldsymbol{E_0}$	0.335
$\boldsymbol{\delta}$	0.5	$\boldsymbol{K}$	8×10^4
$\boldsymbol{\xi_1}$	0.9	$\boldsymbol{\xi_2}$	0.5
$\boldsymbol{T_0}$	10^{-8}		

Table 2.1: RD Model Parameters for 90nm Technology Node [48]

This cycle-by-cycle model is complicated and slow for long stress time simulation. In the next section, we will present a model that can express V_{th} degradation values for long term degradations with maximum estimation error of less than 0.1% [48].

2.2.3.3 Long Term Model of Degraded V_{th}

In order to avoid using cycle-to-cycle model for degraded threshold voltages in the entire life time of large circuits, long term NBTI prediction model was presented in [48]. This model can correctly estimate V_{th} for a given time t (see Figure 2.2).

$$\Delta V_{th} = \left(\sqrt{K_v^2\, T_{clk}\, (\frac{\alpha}{1-\beta_t^{\frac{1}{2n}}})}\right)^{2n} \tag{2.5}$$

$$\beta_t = 1 - \frac{2\xi_1 t_{ox} + \sqrt{\xi_2 C(1-\alpha)T_{clk}}}{2t_{ox} + \sqrt{Ct}} \tag{2.6}$$

Where α is the duty cycle and T_{clk} is the clock period and the remaining parameters are the same as these shown in Table 2.1. In general, the long-term model is a function duty cycle, temperature, frequency and the technology parameters including the supply voltage and the threshold voltage. However, the ΔV_{th} is decreasing with increasing the frequencies and saturate at high frequencies making the degradation frequency-independent [48]. So, the BTI stress are mainly from the duty cycle and temperature that implicitly propagated from the activity at the higher level of abstraction.

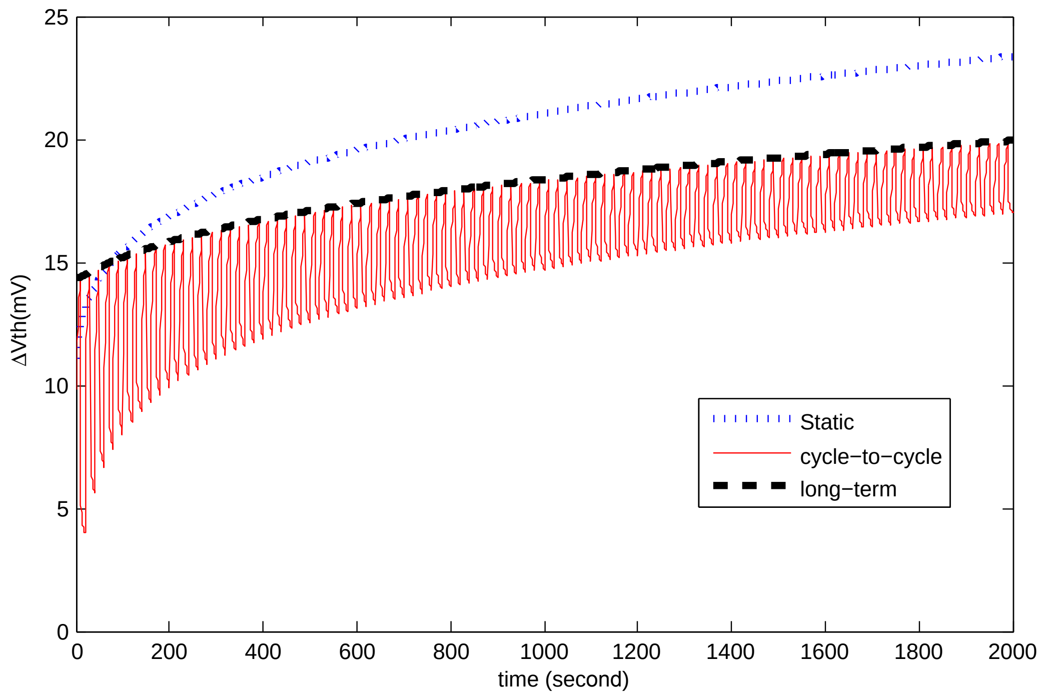


Figure 2.2: V_{th} degradation in dynamic and static operation modes

2.2.3.4 RD Model for 90nm Technology Node

In this section, we have simulated the RD model and simulation results will be shown using MATLAB simulation for the cycle-to-cycle and long-term RD models. The following parameters are used as defaults [48]: initial V_{th} = 0.276V, V_{gs} = 1.2V, T = 350K and t_{ox} = 2.15nm.

Figure 2.3 shows how duty cycles have a big impact on V_{th} degradation. The impact of the duty cycle, $1 - \alpha$, on the threshold voltage is summarised as follows:

- For $\alpha = 0.1$, V_{th} increases by 130mV increases 47% from the initial V_{th}
- For $\alpha = 0.5$, V_{th} increases by 180mV (65%)
- For $\alpha = 0.9$, V_{th} increases by 210mV (76%)

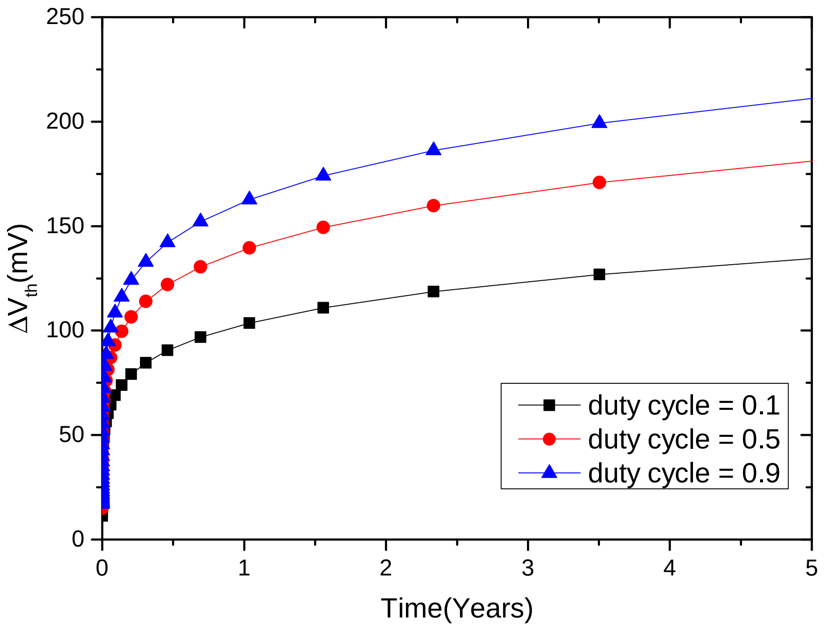


Figure 2.3: V_{th} degradation verses different duty cycles

Figure 2.4 shows how V_{th} generally decreases with increasing frequency. Furthermore, [48, 51] prove that V_{th} saturates with high frequency operation. It needs to be clarified that nodes with very high stress conditions (α=1) will induce a static V_{th} degradation and will be frequency-independent.

Figure 2.5 shows V_{th} increasing with increasing temperature, which is a function of power density that changes when a device switches between active and standby modes. For commercial devices, the temperature range is from $0°C$ to $85°C$.

Figure 2.6 shows V_{th} increasing with increasing supply voltage. Therefore, increasing V_{dd} as a solution for ageing effects could hasten V_{th} degradation.

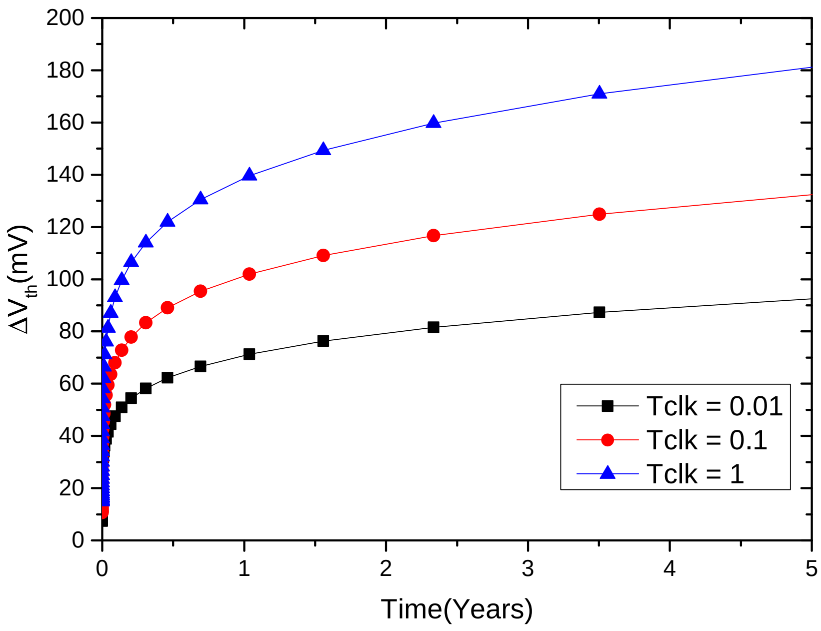


Figure 2.4: V_{th} degradation verses different frequencies

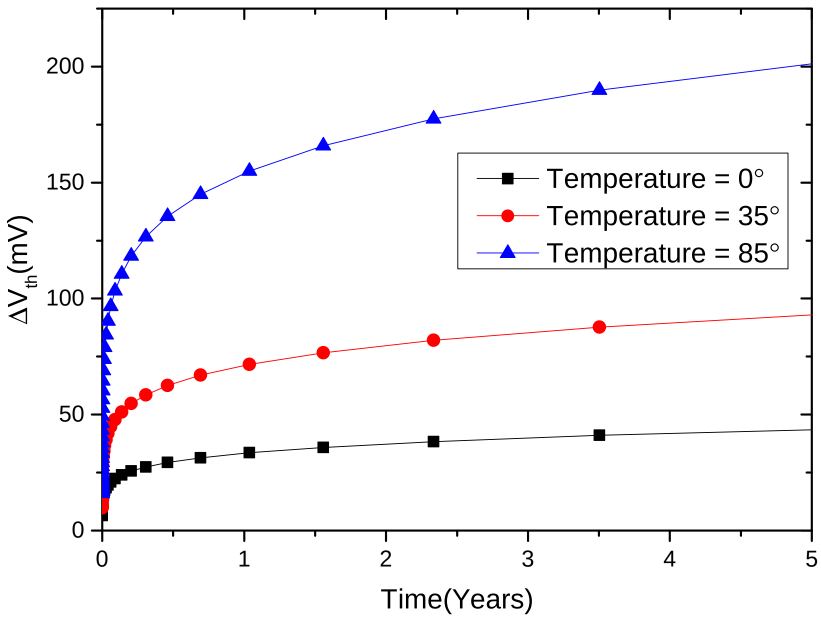


Figure 2.5: V_{th} degradation verses different temperatures

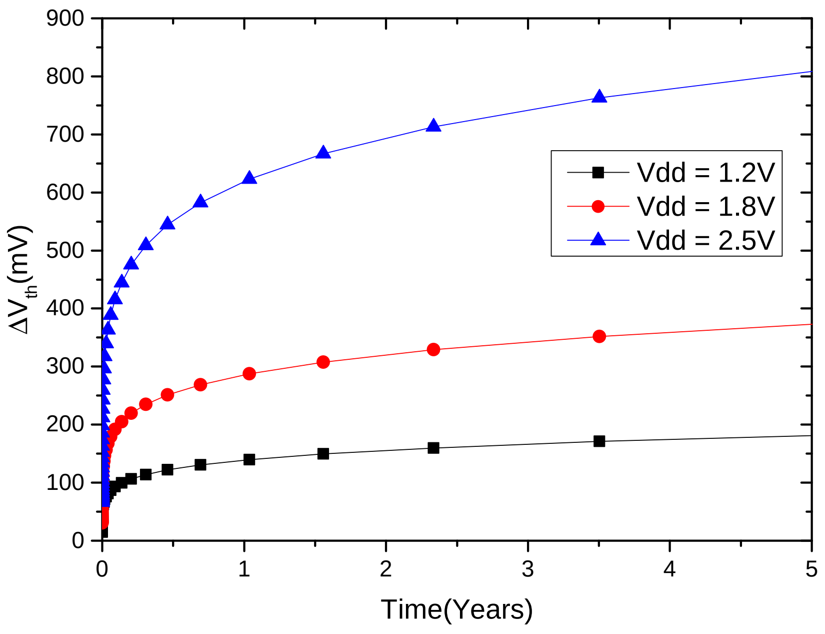


Figure 2.6: V_{th} degradation verses different supply voltages

2.2.3.5 Gate and Path Levels Delay Model

The gate delay shift can be calculated from NBTI-induced V_{th} degradation [52, 53, 54]. The first order of delay shift can be computed as follows:

$$\Delta\tau_{pd} = \Delta V_{th}\frac{\phi\,\tau_0}{V_{dd} - V_{th0}}, \tag{2.7}$$

where V_{th0} is the threshold voltage at any time instance, t, and τ_0 is the corresponding gate delay. ϕ is a constant, whose value ranges from 1 to 2. According to (2.8), a gate-level delay model for a specific gate depends on the transistor construction as shown in the following equation:

$$\Delta\tau = \begin{cases} \Delta\tau_{pd}, & \text{for NAND gates} \\ \sum_{i=1}^{n} \Delta\tau_{pd,i}, & \text{for NOR gates} \end{cases} \tag{2.8}$$

The delay shift of the gate is determined by the sum of delay shifts of the PMOS series transistors in the gate $\Delta\tau$. For example, for the NOR gate, the PMOS transistor are in series and the total degradation are represented by the sum of the individual propagation delay of the PMOS transistors ($\Delta\tau_{pd,i}$). Where n is the total number of the pull-up transistors. Therefore, the delay shift of a critical path (the longest path in a combinational logic circuit from either the primary input of the circuit or the output of the sequential circuit to either the primary output or the input of the sequential circuit) or potential critical paths is the sum of delay shifts of the pull-up transistors. $\Delta\tau_{cp}$ can be derived as:

$$\Delta\tau_{cp} = \sum_{i=1}^{l} \Delta\tau_i, \quad \text{where } l = \text{critical path depth} \tag{2.9}$$

Therefore, considering the delay of the series transistors is important for the accurate computation of the path delay degradations. In [55], the gate and path degradations have been modelled by considering the gate and its structure to avoid ignoring the stacking effect (when the gate structure has a series of transistors on the pull-up, e.g. NOR gate, the transistors may be stressed unevenly overestimating the degradation). In [55], estimating the degradation on NOR gate with consideration of stacking effect is found that having overestimation degradation rate up to 130% than the traditional gate delay degradation. So, ignoring the stacking effect could provide degradation results more pessimistic.

2.2.3.6 System-Level BTI Degradation Model

The complexity of a digital system makes it very hard or impossible to model the BTI degradation at the device-level without assumptions. In [56], the degradation was analysed for the whole microprocessor system based on standard software benchmarks (e.g., office work, gaming and general use). Thus, in [56], the degradation of the whole system have not considered, instead they only analysed the timing critical components and predicted the lifetime of these components of the logic unit combined with the lifetime of the memory units. Reliability calculated based on workload stress collected from running programs. Thus, this is not a model that can be used to predict the lifetime of the system, rather, it is an analysis for propagating the delay from the device level to the system level.

Cha et al. [57] estimate the NBTI parameters at the system level that model the threshold drift at the physical level using a specific usage of the system (e.g., add, divide or subtract). However, the microprocessor system is based on the collective operations of these components rather than their individual operations. So, these analyses only help to understand the key high-level parameters that affect lower level stress. For this reason, the aim of this thesis is to estimate these stresses and reduce them instead of modelling the ageing itself to predict the actual lifetime. This estimate will then be used to analyse the degradation and study its behaviour, which will help us to propose a mitigation technique and evaluate it.

2.2.4 Signal Probability

Duty cycle is the ratio of the system or signal being active (ON or '1') to the total period (ON plus OFF) [58]. Meanwhile, the probability of a signal being zero, SP(0), or the signal probability, reflects the fraction of time spent in the stress state (OFF or '0', from NBTI perspective). The duty cycle can be defined as [59]:

$$SP(1) = \frac{PW}{T} \times 100\%,$$

where SP(1) is the duty cycle in percentage, PW is the pulse width and T is the total period of the cycle (See the Figure 2.7). The signal probability of the signal being zero (SP(0)) can be defined as:

$$SP(0) = 1 - SP(1)$$

We simulated the effect of the duty cycle on the threshold voltage using a commercial reliability model, namely the HSPICE MOSRA Built-in Model Level 3 [60]. MOSRA can evaluate both NBTI and PBTI for a circuit at the SPICE level and obtain gate or path degradation, rather than just threshold voltage degradation or leakage current

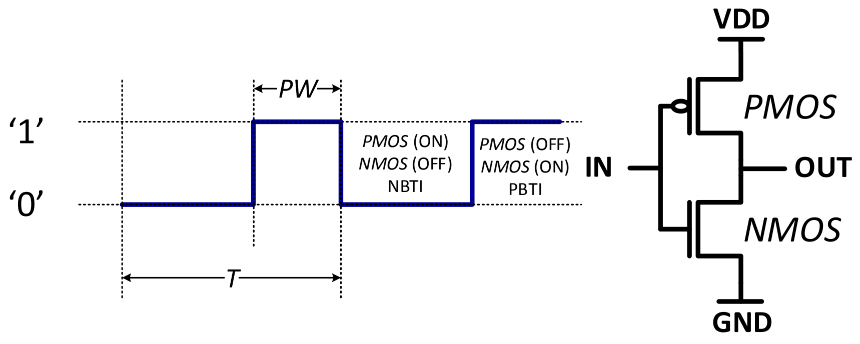


Figure 2.7: Example of the stress condition on inverter logic gate which if the input signal (IN) is '1', the pull-up transistor (PMOS) would be stressed with NBTI but if the input signal (IN) is '0', the pull-down transistor (NMOS) would be stressed with PBTI. These stresses period is defined by the pules width (PW) and the period (T).

increase at the transistor level, as given by the basic Reaction-Diffusion (RD) model [48]. Decreasing the duty cycle can impact positively on the V_{th} degradation of PMOS devices and negatively on the V_{th} degradation of NMOS devices. MOSRA simulations use degraded device parameters in HSPICE to calculate gate or path delay degradation in timing simulations [61].

Simulation parameters have been tuned to match the degradation given by statistical data, [62, 63]. We applied different SP(0) values to a circuit consisting of two inverters in series to calculate the maximum path delay after 10 years for two different technologies (90nm from Synopsys and 65nm from TSMC). We activate only the NBTI effects because PBTI should barely exist at these technology nodes and if modelled with smaller technologies would exaggerate the ageing effect. As can be seen from Figs. 2.8 and 2.9, the signal probability has an impact on path delay degradation and, thus, on the lifetime. While one node is highly stressed, it will tend to have more static NBTI and by balancing the signal probabilities, the static stress will be reduced as well. As the delay degradation is less at the end of the device lifetime than at the beginning, decreasing the path delay by a small amount could enhance the lifetime of a device by several years. Figure 2.10 shows the lifetime of the two inverter chain with a target maximum delay of 0.237ns for the 90nm technology from Synopsys and 0.149ns for the 65nm technology from TSMC.

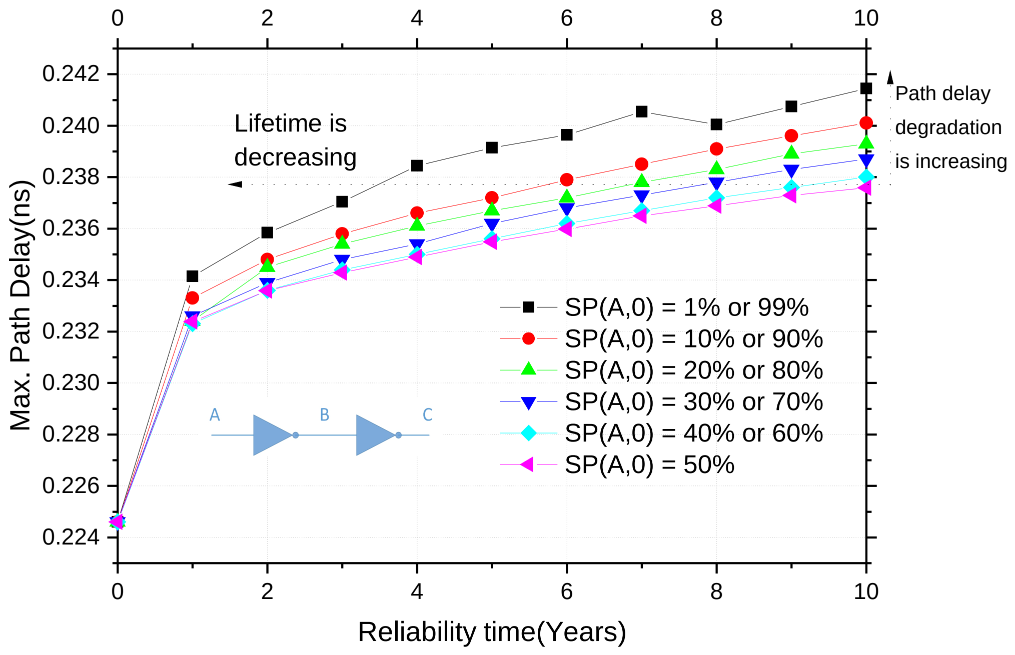


Figure 2.8: Path delay for a chain of two inverters for different SP(0) values at the primary input using 90nm Synopsys technology with NBTI effect.

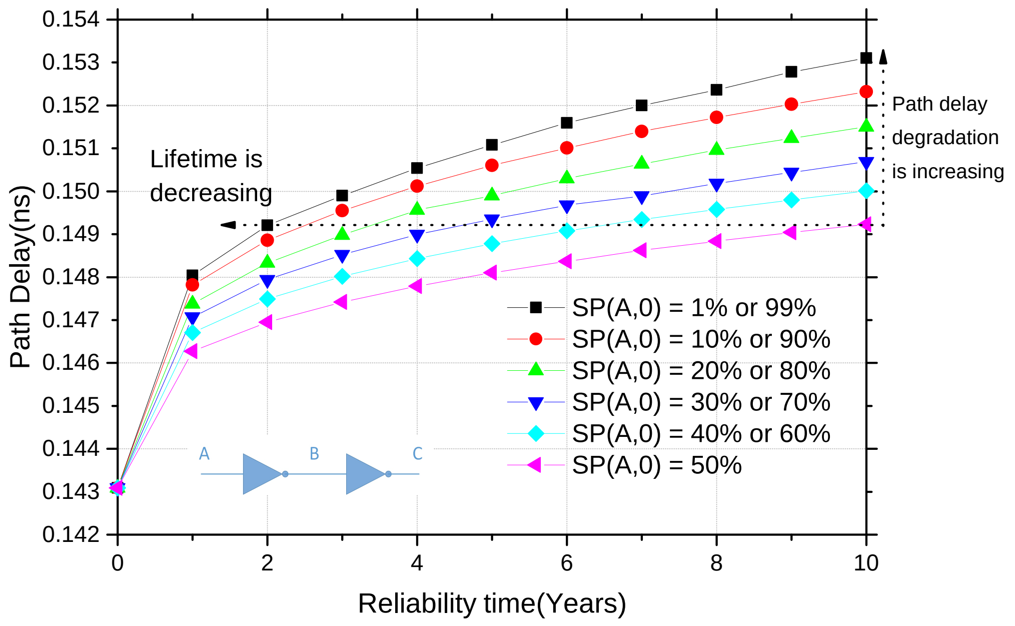


Figure 2.9: Path delay for a chain of two inverters for different SP(0) values at the primary input using 65nm TSMC Technology with NBTI effect.

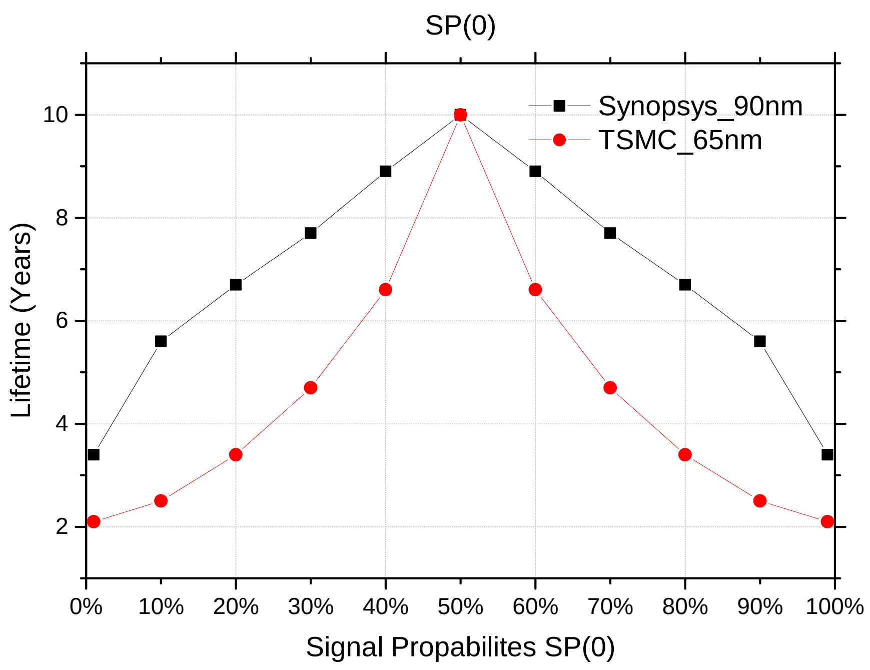


Figure 2.10: Lifetime for a chain of two inverters for different Signal Probabilities SP(0) values at the primary input.

2.3 Related Works

In this section, the literature of the mitigation techniques used for ageing is presented. A feasible solution to reliability problems, including ageing is to eliminate or even to reduce the design uncertainties that exist in current design technologies. However, in practice, there is more than one contributor to these uncertainties, including EDA tool limitations and complex environmental stress conditions [64]. Another solution is to design conservatively by using the worst-case scenarios at the design stage. However, circuits do not always run at the worst-case condition and such an over-designed approach is extremely costly in terms of power and area. The following subsections will investigate the available mitigation techniques based on the level of abstraction design they introduced.

2.3.1 Low-Level NBTI Mitigation Techniques

Different design-time techniques have been proposed to reduce NBTI degradation. Abadeer and Ellis [65], Zhang and Dick [66], Parihar et al. [67] explore the supply voltage and frequency scaling over time to reduce guard-bands and increase the lifetime of the circuit. Also, Wu and Marculescu [68], Butzen et al. [69] explore the signal probability by reordering the gate pins and restructuring the gates. In Butzen et al.

[69], the authors proposed a transistor network restructuring method to mitigate the NBTI degradation. An example of a different way to structure the CMOS transistors inside the gate is AND-OR-INV gate (AOI21). For instance, Figure 2.11 shows two logically-equivalent AOI21 gates with different structures for the pull-up transistors. In [69], it shows that these different structures have different behaviours in area and power consumption, and they have different NBTI degradation levels.

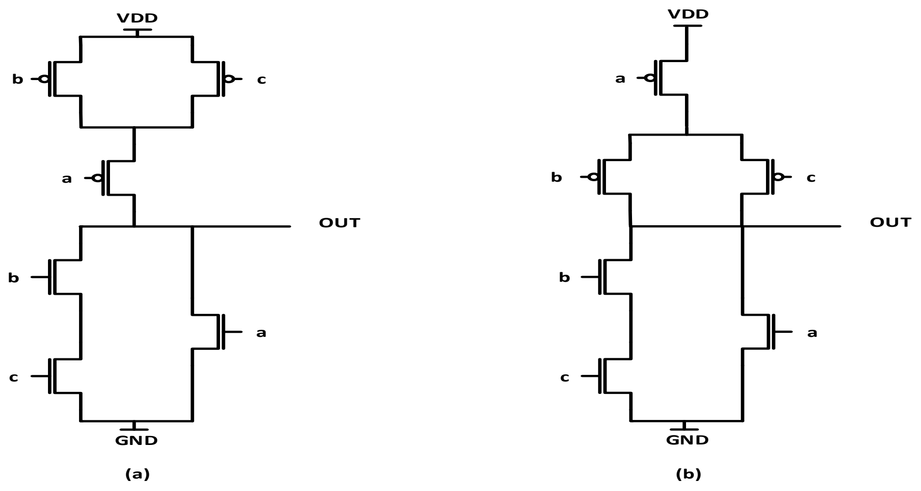


Figure 2.11: Two logically-equivalent AOI21 CMOS gates

At gate level, NBTI manifests itself as a time-dependent gate delay and finally leads to timing violations. Adding some margins to the critical path (the longest path from either the primary input or the sequential circuit output to either the primary output or the sequential circuit input, see Figure 2.12) is not a solution because the critical path at time zero may not still be the critical path after some years due to ageing [70], as we will show later in the thesis. Potential critical paths (PCPs) or NBTI critical paths take into account the effect of NBTI in the static path analysis. In [71], critical gates (gates within PCPs) are identified and optimisation methodologies (i.e. gate resizing or reducing temperature of this part of the circuit) are proposed for these critical gates.

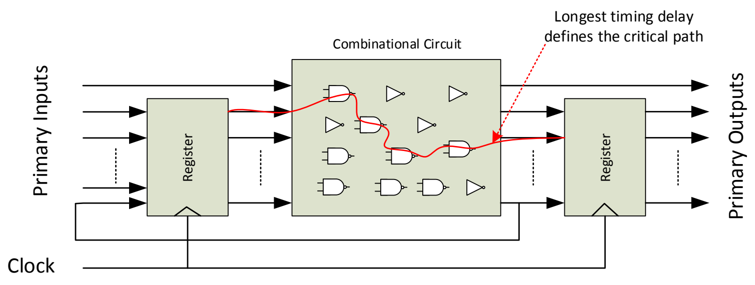


Figure 2.12: Critical path example in the combinational logic circuit

Gate sizing was presented in [72, 73, 74] for computing the optimal gate size taking into consideration the effect of NBTI stress conditions. Figure 2.13 illustrates the main idea of the gate-sizing technique. Initial over-design is used by sizing the gates to reduce the delay of the critical paths and relax the margin of the setup time. The dotted curve shown in Figure 2.13 represents the original design that fails to guarantee the functionality at the pre-defined lifetime (T_{REQ}) because the setup time margin reduction over time due to NBTI degradation finally sets below the worst case setup time constraint (D_{CONST}). The authors in [72] showed that by gate sizing, the lifetime of the ISCAS benchmark circuits could be guaranteed for three years with 8.7% average area overheads. In the same way, transistor sizing for the flip flops has been proposed in [75] and for the SRAM cell in [76] by up-sizing the transistors of these units.

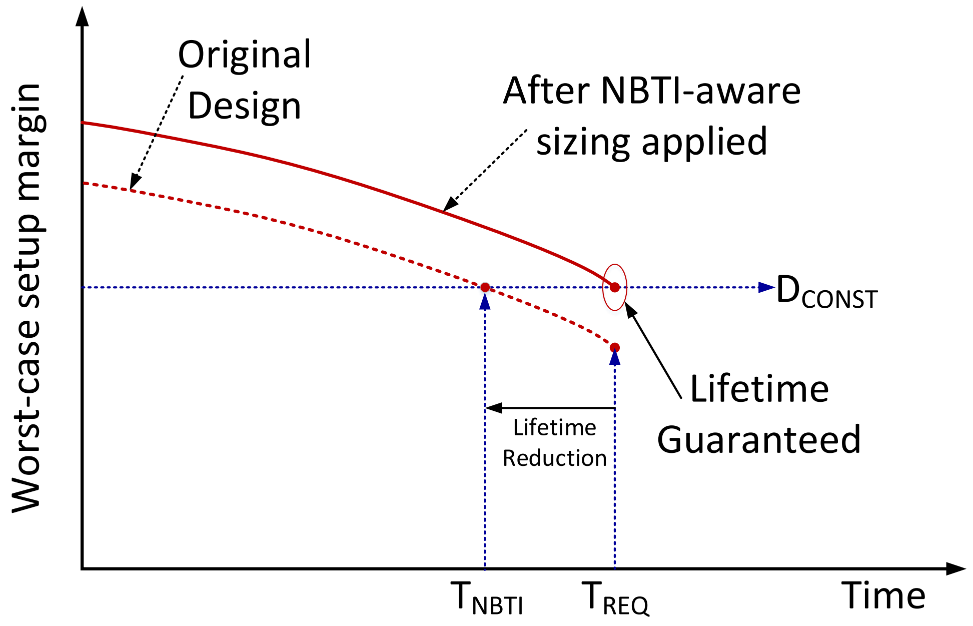


Figure 2.13: Lifetime extension with gate-sizing design [72]

In [15], it was shown that the area overheads of gate-sizing could be reduced by about 43% compared to the technique proposed in [72] by using different sizing rates for the NMOS and PMOS transistors of the logic gates. By this approach, the rising delays are targeted, which are mainly affected by NBTI rather than resizing both PMOS and NMOS transistors in the same rate. Furthermore, in [77], gate sizing has been utilised to mitigate NBTI effects on a group of representative critical paths (paths that contain the gates most vulnerable to ageing). Also, in [15], a technology cell library was extended for ageing mitigation. The aim of this work was to balance the rising and falling delays at the predefined lifetime rather than at time-zero of the operation using gate sizing for different signal probability distributions.

NBTI has a dependence on dynamic operation, such as supply voltage, spatial or temporal temperature and signal probability, and these parameters vary dynamically from one gate to another. Another solution, proposed in [12], uses signal probability to restructure the logic gates and arrival times of the input signal to reorder the pins. However, signal probabilities are assumed to be 50% for input signals and for larger systems, signal probability is dynamic and is based on the application.

Ashraf et al. [78] proposed a spatial hardware redundancy to the critical paths in which they adaptively alternate the selection between two identical paths. During the activation of one path, the concurrent path is power gated for lowering power and ageing stress. The main issue of this proposal is that the additional circuits would also be influenced by the ageing stress. Also, these additional circuits would increase the fan-out of some gates making the initial critical path delay larger than the one without the hardware redundancy.

So, most of the low-level techniques for NBTI mitigations have the advantage to prove their efficiency as modelling the ageing at this level is less complex than that at higher level of design abstractions. However, these low-level techniques have the following shortcomings:

1. They have many assumptions on the real workloads coming from the higher level of abstractions making the solutions not accurate.
2. They have strong dependence on the technology used making them hard to integrate into commercial tools.
3. They introduce additional circuitry for mitigating the NBTI ageing effects; however, these circuits could be under stress themselves making the system more vulnerable to ageing stress.

2.3.2 Component-Level NBTI Mitigation Techniques

Component-level mitigation techniques target a specific unit of the system (e.g. multiplier). For example, Lin et al. [79] proposed a novel design of a multiplier by considering the ageing effect. Variable latency designs have been used to reduce the wasted time in non-worst case possible delay. Figure 2.14 shows path delays for the basic design of the array multiplier (AM), and power optimised designs (column-bypassing and row-bypassing). More than 90% of the input patterns have a path delay less than half of the maximum delay of the multiplier. Thus, the variable latency design has been used to improve the circuit performance by implementing shorter paths within one cycle and longer paths within two cycles. To understand the principle of variable latency design, 8-bit ripple carry adder is used to demonstrate the principle of variable latency design (see Figure 2.15). The maximum propagation delay of the circuit is eight clock cycles, but normally, not all input patterns need eight clock cycles. Therefore, a hold circuit is added to indicate whether the addition needs four or eight clock cycles. If $A_4 = B_4$ or $A_5 = B_5$ then the hold signal needs to halt the clock for the next four cycles. Figure 2.16 shows the multiplier version of the variable latency design with adaptive hold logic (AHL) and Razor flip-flops [80]. A Razor flip-flop is used to detect timing violation and send an error signal to the AHL, which will indicate whether the error is due to delay degradation from ageing or it just needs two cycles due to its input pattern. The main limitation of this work is that it considers only the delay degradation of one-cycle patterns. However, two-cycle patterns are less frequent but might be more stressed due to NBTI.

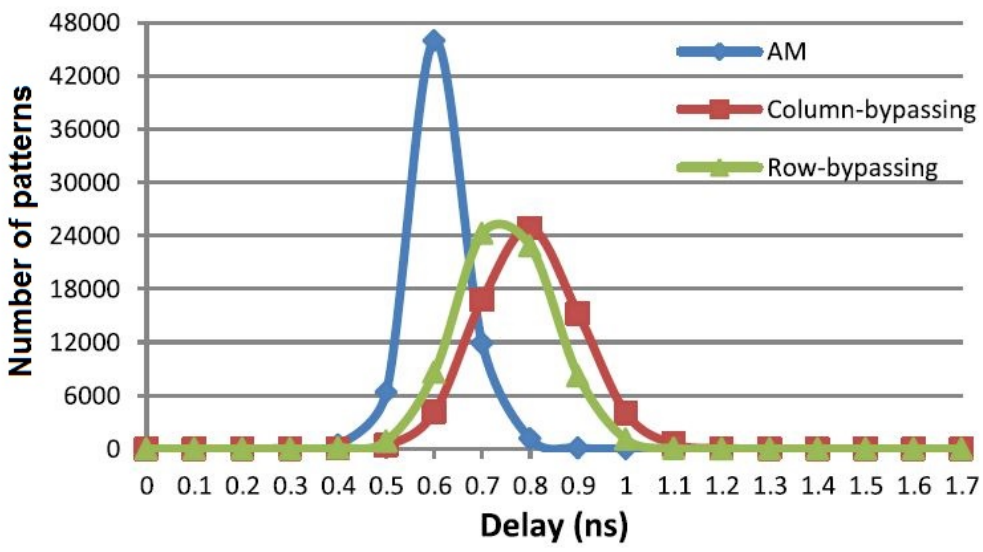


Figure 2.14: Path delay of different multipliers for different operands [79].

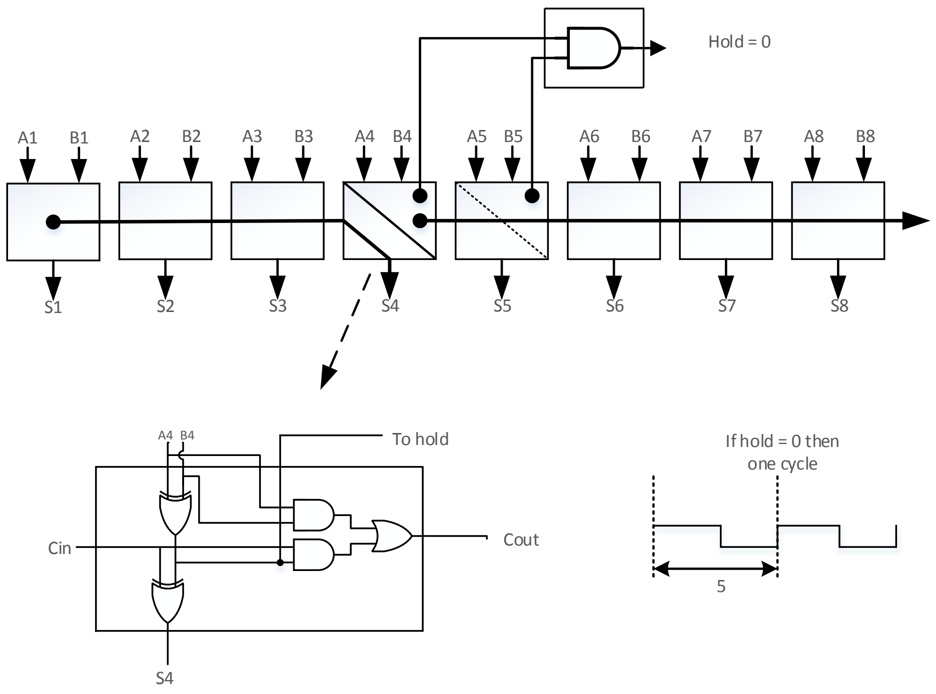


Figure 2.15: Variable latency RCA [79].

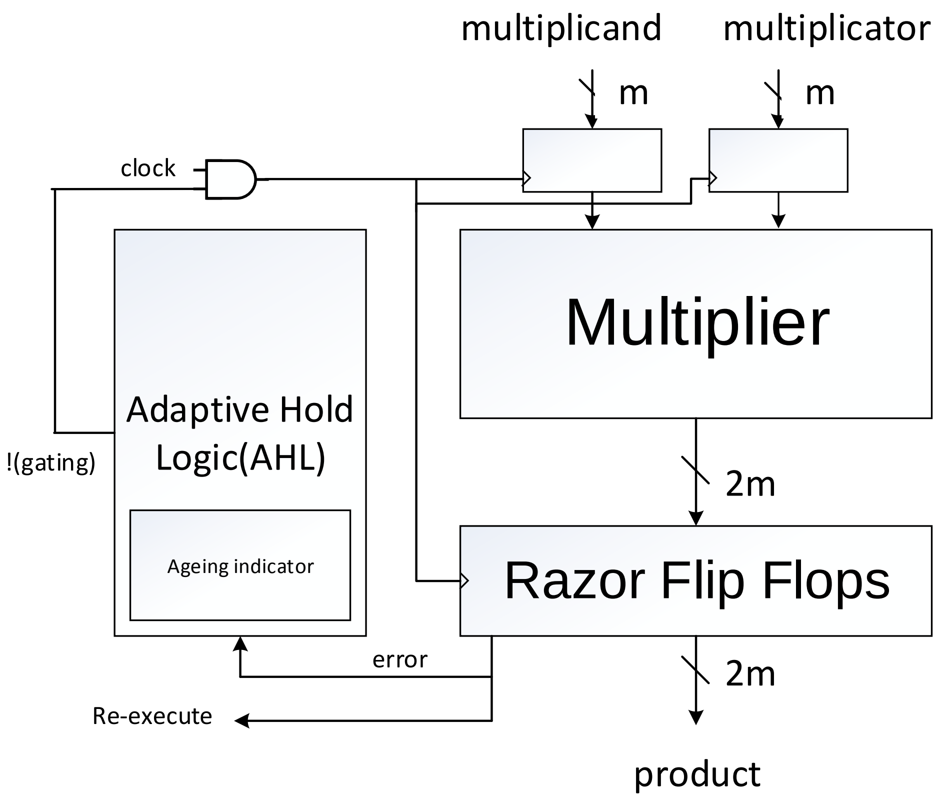


Figure 2.16: Ageing-aware multiplier [79].

2.3.3 System-Level NBTI Mitigation Techniques

Another high-level prediction technique is using ageing sensors. The authors of [9, 10] propose the dynamic wear-out/NBTI management (DNM) based on sensors with the aim of reducing design margins. This approach reduces the power consumption by running the circuit with the least possible supply voltage rather than the worst possible case (see Figure 2.17), and change the supply voltage periodically based on reading from ageing sensors. However, the main challenge of this approach is the accuracy of the sensors and the area overheads, which may possibly exceed design constraints.

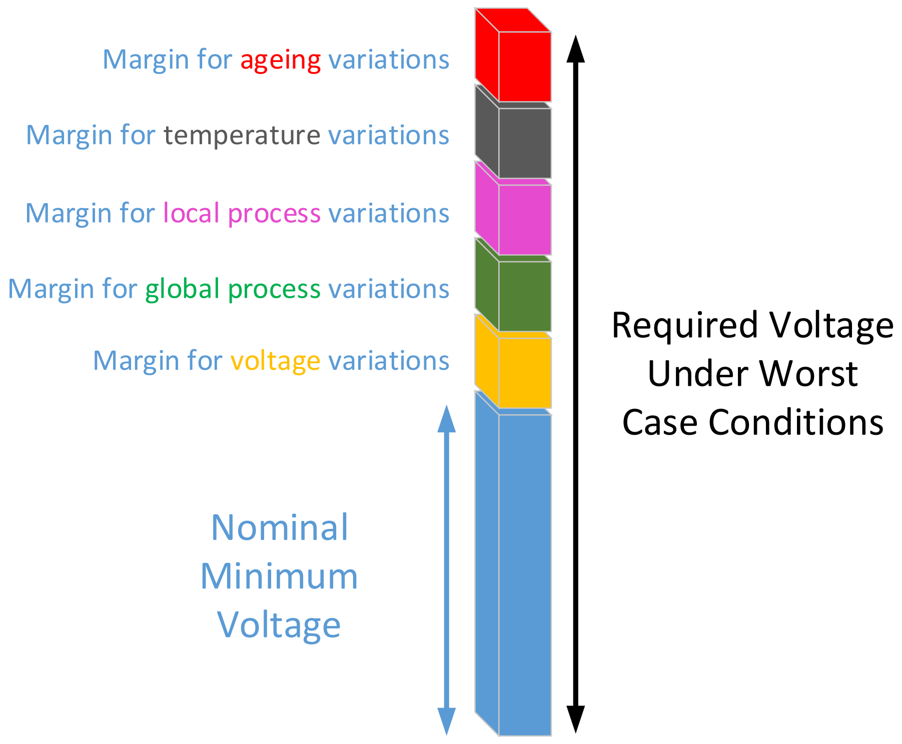

Margin for ageing variations
Margin for temperature variations
Margin for local process variations
Margin for global process variations
Margin for voltage variations
Nominal Minimum Voltage
Required Voltage Under Worst Case Conditions

Figure 2.17: Source of design margins [10].

Some of the works in the literature propose mitigation techniques targetted at specific technologies and sizes. For example, in [81], the authors design their sensors by utilising the fact that the relationship between the gate and subthreshold leakages are proportional at 45 nm. Figure 2.18 illustrates the degradation prediction, where $D(t_0)$ is the degradation reading at time t_0. An ageing model is used to predict the degradation at the lifetime ($D_{pred}(T_{LT})$). If the predicted degradation is less than the timing constraints, a higher temperature limit is allowed or the supply voltage is increased. Otherwise, the supply voltage is lowered. A challenge of this approach is that the sensor design needs to have low power consumption and small area overheads. Furthermore, there is the need to have an accurate ageing model that can predict without pessimistic or optimistic decisions.

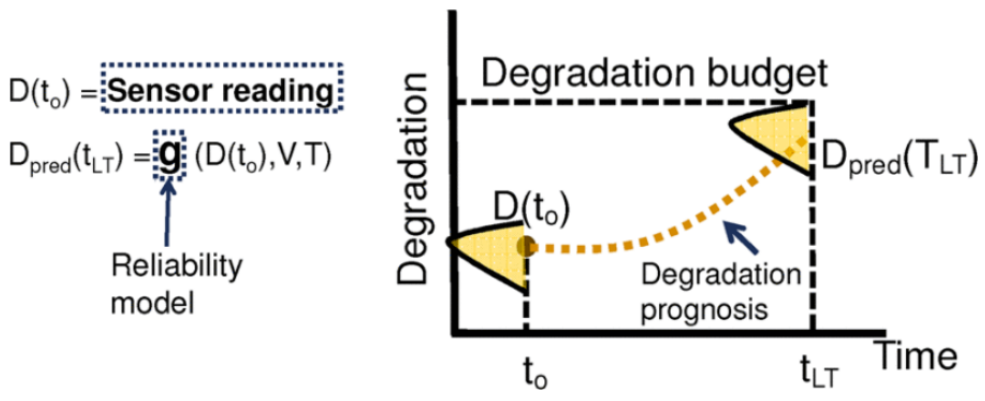


Figure 2.18: Dynamic NBTI Management (DNM) sensor based. $D(t_0)$ is the degradation reading at time t_0. Using the ageing model (g) to predict the degradation at the lifetime ($D_{pred}(T_{LT})$). [81].

2.4 Concluding Remarks

In general, there are three possible approaches of ageing mitigation techniques, namely, proactive, reactive or ageing-aware (protective). The proactive approach works as an estimator for ageing behaviour and is always based on a model that describes how ageing effects (e.g., NBTI, HCI and TDDB) are modelled by physicists at a low level. Usually, this approach needs to take into account different contributors of time-dependent variations (e.g., signal probability, switching activity, temperature and supply voltage). Another approach is to monitor the real behaviour of ageing through delay sensors on-line. This approach is more precise and avoids the complexity of modelling the ageing effects at the system level. However, the main problem of on-line sensors is the area overhead, and to reduce this drawback, only a limited number of nodes can be monitored. Furthermore, the delay sensors themselves suffer from degradation over time. The third approach tries to alleviate the source of ageing either by reducing the temperature or by reducing the stress probability by reversing it in the critical path.

In general, the already proposed techniques for mitigation the ageing effect have at least one of the following drawbacks:

- They are technology-dependent.
- They ignore the stress on the internal nodes of the compound gates.
- They assume that the stress coming from the high-level (e.g., the software) is represented as 50% signal probability of being zero (SP(0)) at the primary inputs of the gate-level circuit.
- They require design-time modification (intrusive) while the information about the workload and stress would not be available until finishing the design and test phase of the chip.

- They are trying to extend the lifetime by reacting to the degradation rather than reducing the stress.

In this research, we will focus on the protective approach to reduce ageing stress from the negative workloads that lead to circuit degradation rather than estimate the lifetime of the chip. These workloads are programs at the software-level running on the microprocessor, and represent the signal probability at the gate-level of abstraction. Therefore, a cross-layer analysis and tools are required for propagating these system-level workloads into gate-level signal probabilities. The stress probability distribution needs to be extracted as a source of ageing stress and analysed from the software level to the gate level to remove the assumptions made at the gate level about the real stress generated from the upper layers. A cross-layer mitigation technique needs to include the high-level stress in the low-level design of the processor, for example, by controlling the workload or using a multi-core architecture for optimising the workload distribution for lowering ageing stress. Thus, the first step of our proposed technique is to study the signal probability and subsequently create a tool for incorporating this important factor in the NBTI-induced delay calculations.

Chapter 3

BTI Stress State Analysis

3.1 Introduction

Chapter 2 provided an overview of the mitigation techniques for NBTI degradation at different levels of design. Most of these techniques assume that the workload is represented as 50% (i.e., balanced) signal probability at the gate level. In this chapter, we avoid making the assumption that the signal probability (stress state) of the processor components is balanced, which will allow us to learn whether tuning the workload is beneficial or not. NBTI ageing analysis will be conducted comparing two cases, namely, real signal probability propagated from the software-level, and balanced signal probability at the input of the timing critical paths of the processor circuit.

The objective of this chapter is to extract the ageing stress state to the gate-level of the processor driven from the software executed as a workload by developing a tool to facilitate the process. The extracted ageing stress state need to be analysed for identifying the lowest ageing stress state on the processors.

The chapter is organised as follows. Firstly, in Section 3.2, developed a tool to extract the signal probability from the level of running application on a processor to the gate-level of the processor. Firstly, These signal probability has been analysed on selected microprocessor (CORTEX M0 ARM processor) in Section 3.3. Secondly, in Section 3.4, we present an approach to statically analysing the ageing by building ageing-induced library.

3.2 Signal Probabilities SP(0) Generation Tool

To obtain the signal probabilities (duty cycles) of the nets being stressed with logic zero (SP(0)) for a specific application running on a specific processor, we use the value change dump (VCD) file, which can be extracted from many EDA tools (e.g. Modelsim and VCS) by running programs on simulated processor. The VCD file implicitly contains both switching activity that is used to estimate the dynamic power at the design phase and the signal probability that we use to estimate the NBTI effect on performance degradation. We built a compiler using open source software (JFLEX and CUP) available in [82] to generate a file containing the SP(0) for all the nets of the processor. In Appendix A, an example of a VCD and a generated SP(0) files with the code for compiling and translating from any VCD file to the SP(0) file is presented. The general procedure for SP(0) generation is:

- First, the VCD file are scanned using VCD Scanner tool build using CUP parser generator that is responsible to collect the tokens (e.g., keywords from VCD file, for example, '$scope', '$var' or '$comment'). These tokens are collected and saved in a dictionary file of a full list of keywords to be passed to the next step.
- Second, the VCD file are parsed and translated into a SP(0) file by using VCD Parser to SP(0) tool that is responsible to find the grammar from the input file to identify whether the line from the VCD is a time (e.g., '#2296' is the simulation time in nanosecond in which the following signal changes has happened) or a signal value that changed at the last time line (e.g., 1$ it means the signal $ has '1' between two time lines, the symbol '$' is defined in the header section of the VCD file, see the example shown in Figure 3.1 and for more details see the given example in the Appendix A).

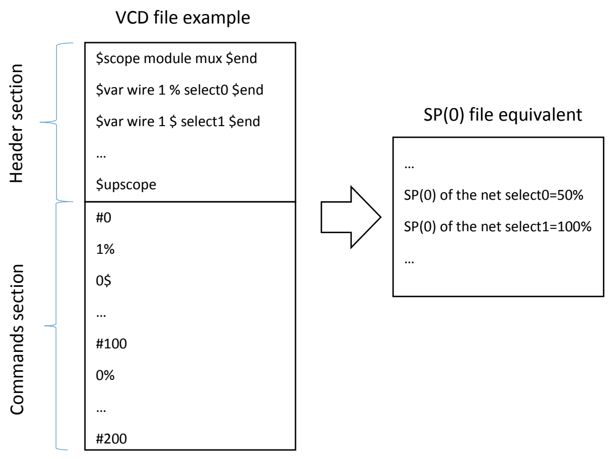


Figure 3.1: Example of value change dump VCD file and its equivalent signal probability SP(0) file

3.3 Case Study: NBTI Critical Paths Analyses

As described in 2.3.1, the NBTI critical paths are the time critical paths for the circuit that has been degraded with NBTI and may not necessarily be the same as the time-zero critical paths of the circuit. Having information about the signal probability from the application level could help in identifying NBTI critical paths. Figure 3.2 shows the proposed approach to identify NBTI critical paths, which can be summarised in four steps:

1. Consider a specific processor, then run different benchmarks on the simulated processor and obtain a VCD file that represents the application activity at gate level.
2. Obtain the signal probabilities of the nets from the VCD file.
3. Based on the predefined lifetime of the processor and SP(0), calculate the degradation value of V_{th} for a range of signal probabilities using the NBTI model.
4. Calculate the new delays for the paths and identify the worst degradation values that will represent the NBTI critical paths.

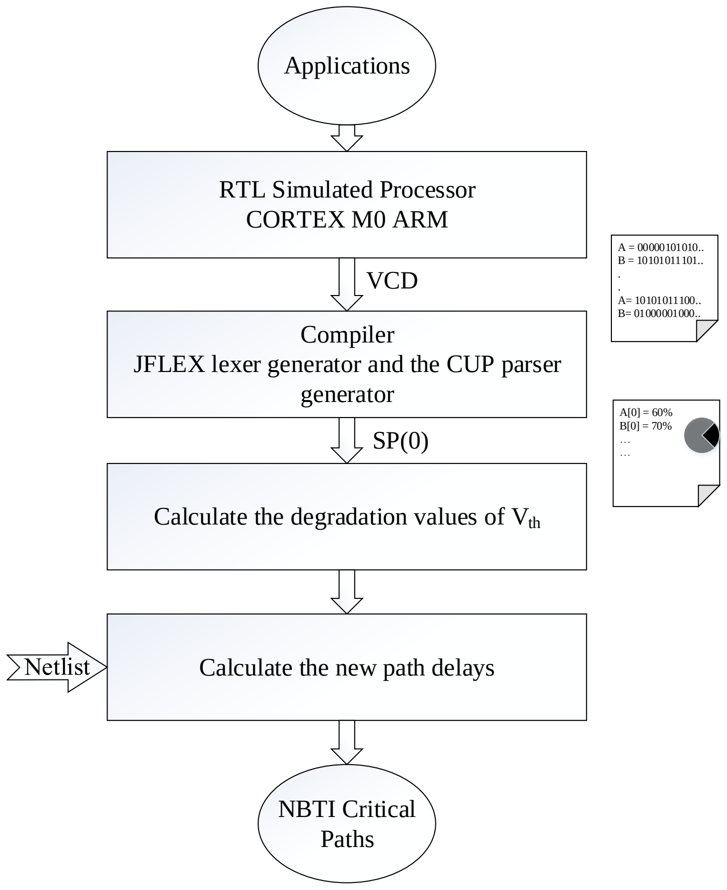


Figure 3.2: NBTI-critical-path identification steps

3.3.1 Analysis Setup and Evaluation

To get the SP(0) for the processor at the gate level we simulate using ModelSim simulation tool by running ‘hello world’ program on synthesised gate-level CORTEX M0 ARM processor, we have obtained the VCD file that compiled and translated into SP(0) file using the tool presented in Section 3.2. We can see the signal probability on the gate level of the processor, see Figure 3.3 and Figure 3.4. 14361 signals out of 49200 signals have SP(0) more than 90% (29% of the nets are mostly stressed), while 18606 signals out of 49200 signals have SP(0) less than 10% (38% of the nets are mostly unstressed) (see Figure 3.4). Furthermore, this is not the average activity of the system. Embedded systems that run a specific application or usually run different applications from the same domain (e.g. Internet of Things (IoT)) have a limited scope of applications. The VCD files for these applications will represent the average application activities that will contain the average signal probabilities of the system.

```
.......
========SCOPE= module======== MODULE= U1085
SP(0) of the net Q=96%
SP(0) of the net IN1=96%
SP(0) of the net IN2=2%
SP(0) of the net IN3=97%
SP(0) of the net IN4=83%
SP(0) of the net g_1_out=96%
SP(0) of the net g_2_out=99%
END_SCOPE
========SCOPE= module======== MODULE= U1086
SP(0) of the net Q=93%
SP(0) of the net IN1=93%
SP(0) of the net IN2=3%
SP(0) of the net IN3=96%
SP(0) of the net IN4=95%
SP(0) of the net g_1_out=93%
SP(0) of the net g_2_out=99%
END_SCOPE
.......
```

Figure 3.3: Part of the signal probabilities file

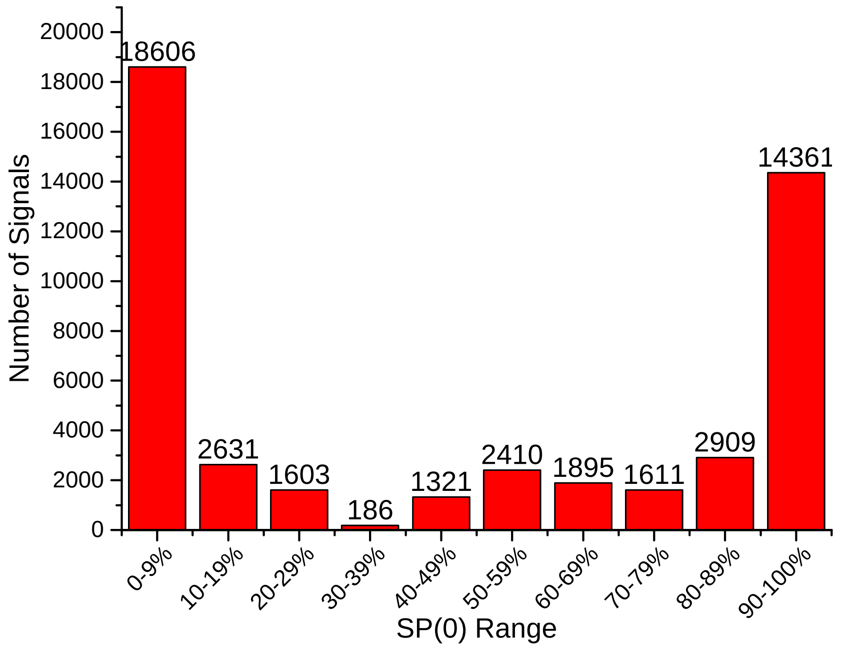


Figure 3.4: The SP(0) distribution of CORTEX M0 ARM for Hello World program

First, we simulated part of the critical path for the circuit shown in Figure 3.5 to see how the signal probability can make a difference to ageing effects in path delay. Considering three cases:

- CASE A – delay of a circuit at time zero (not aged circuit);
- CASE B – NBTI-induced delay under assumption of 50% duty cycles at the primary inputs on circuit degraded for five years;
- CASE C – NBTI-induced delay with duty cycles of all signals equal to SP(0) from the above analysis on circuit degraded for five years.

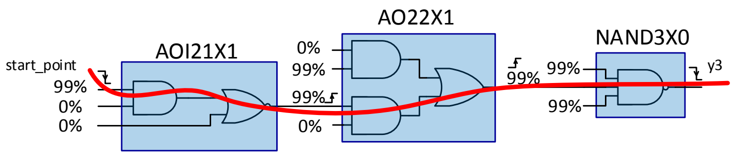


Figure 3.5: Critical path sub-circuit

To propagate the SP(0) values to transistor level, the long-term RD NBTI model has been used to calculate the degradation values of V_{th}. From this, we induced these values of V_{th} manually as instances (gates with different degradation levels) in the HSPICE. Then, the critical path netlists have been instantiated based on the equivalent SP(0) that helps us to calculate the path delay for 90nm technology node from Synopsys. The results show 21.33% difference between CASE B and CASE C, and that CASE C is more optimistic than CASE B (see Table 3.1 and Figure 3.6).

Table 3.1: Sub-circuit path delay (seconds) after five years

CASE A	CASE B (α=0.5)	CASE C ($\alpha = \text{SP}(0)$)	difference
1.1763E-10	1.6013E-10 (36.13% increased)	1.3503E-10 (14.79% increased)	-21.33%

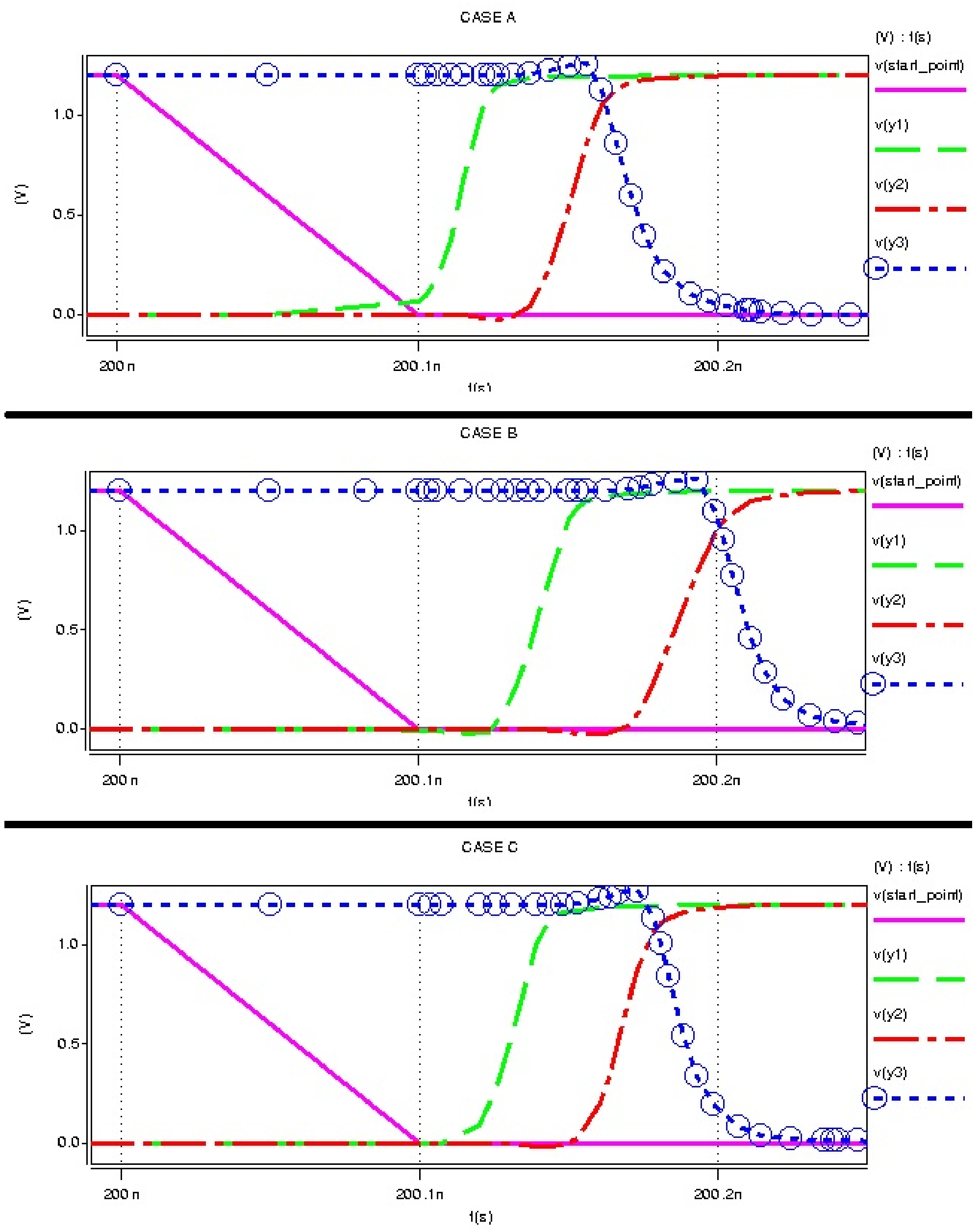


Figure 3.6: Sub-circuit path delay for the circuit shown in Figure 3.5: CASE A (Delay of new circuit (not yet degraded)); CASE B (Delay after 5 years with the assumption of having primary input signal probability 50%); CASE C (Delay after 5 years with signal probabilities extracted from real workload).

Using the signal probabilities obtained by running an application on a microprocessor, rather than using the default assumption (SP(0) = 50%), has a significant impact on the critical path delay computation of a system. We simulate the path delay for the two most critical paths using the cases listed in Table 3.1. For the first critical path as shown in Table 3.2, the degradation delay for CASE B is increased by 49.56%, and by 28.9% for CASE C. The results show that CASE C is more optimistic than CASE B with the same trend for the first three instances on the critical path. For the second critical path, as shown in Table 3.2, the degradation delay for CASE B is increased by 32.66%, and by 72.2% for CASE C. From these results, the first critical path is no longer the most critical path after five years and the second or other paths may now be considered to be the most critical path (see Figure 3.7 and Figure 3.8).

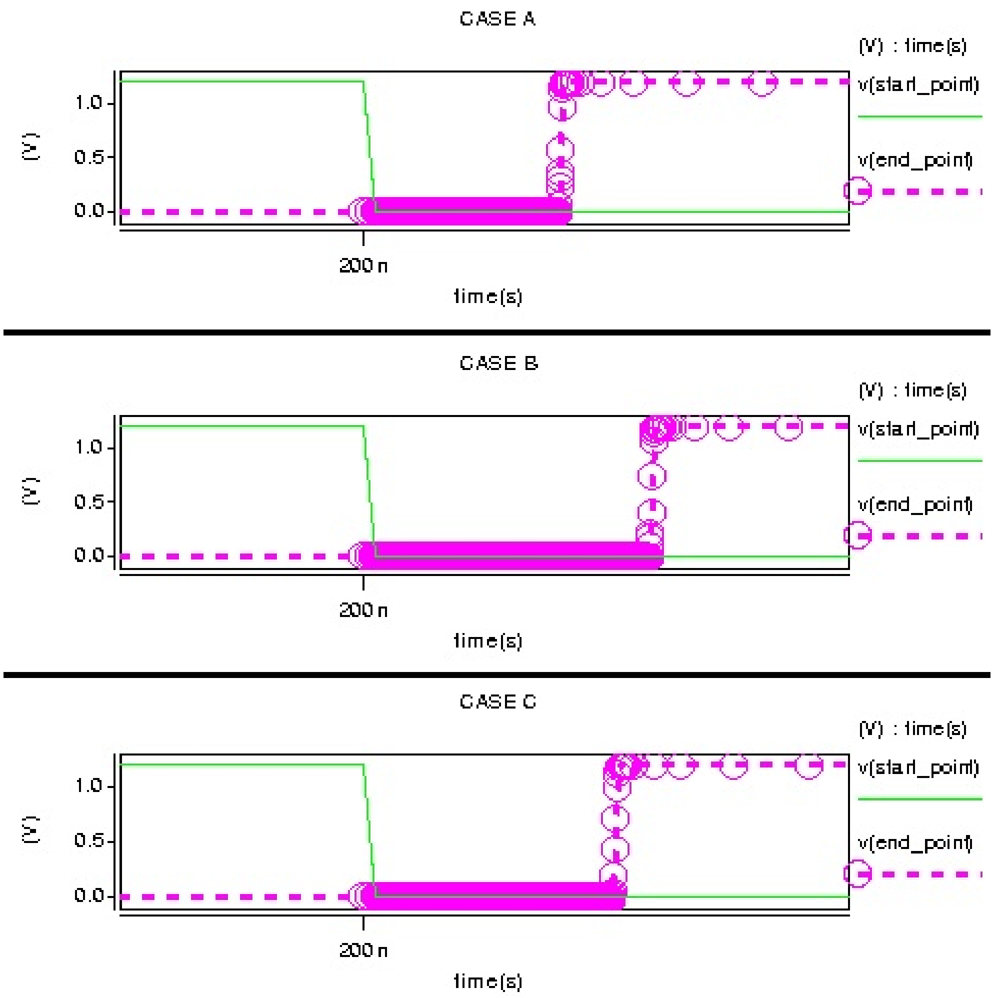


Figure 3.7: Path delay of the first critical path

Table 3.2: Critical path delay after five years

Critical Path Number	CASE A	CASE B (α=0.5)	CASE C ($\alpha = \text{SP}(0)$)	difference
1	1.5687E-09	2.3463E-09 (49.56% increased)	2.0221E-09 (28.9% increased)	-20.66%
2	1.5569E-09	2.0655E-09 (32.66% increased)	2.6811E-09 (72.2% increased)	39.54%

Estimating the path delay for all the possible paths of circuit is not feasible for complex systems. Therefore, classifying paths with the potential to be critical paths is possible if the maximum degradation delay is known for the predefined lifetime of a system. This may make some paths less critical. After reducing the number of paths that

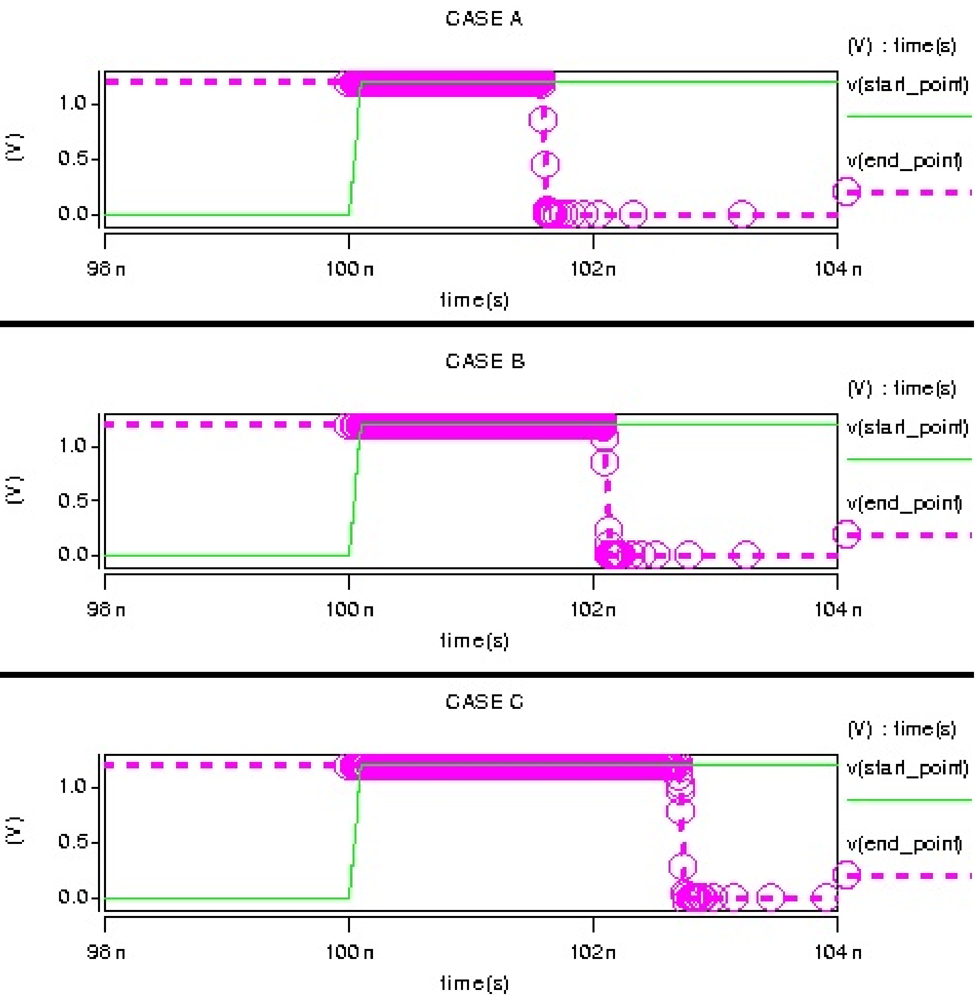


Figure 3.8: Path delay of the second critical path

could potentially be critical, further estimation can be applied using the proposed work for identifying NBTI-critical path by considering the real activity of the system. As signal probability is not the only observable parameter from the application level to transistor level, temperature is possibly observed as a function of switching activity of the transistor. However, the temperature is environment-dependent and the spatial deviation of the temperature in a small area is normally small. Therefore, considering the signal probability derived at the application level will not estimate the degradation delay with 100% accuracy, but it will increase the accuracy of the estimation.

3.4 Case Study: Static Ageing Analysis

Modelling NBTI-induced delay at gate level depends on the real stress activity of gate inputs, which are related to the workload applied from the higher level of abstraction (e.g., a running application). Having estimated values of the degradation delays can make this issue a design constraint and even to precisely allocate the on-line ageing sensors. This chapter proposes a method to include the stress probability within the technology library as three dimensional look-up tables for static timing analysis (STA) process of the design as an approach named static ageing analysis (SAA). The purpose of this approach is to estimate NBTI-induced delays for the predefined lifetime of the product instead of estimating only the timing delay at time zero during the logic synthesis.

The standard library that is used to estimate the timing and power at the synthesis stage is based on pre-calculated look-up tables considering the input transition delay and output capacitance load as two-dimensional LUTs for each standard cell in the library. To generate an NBTI-induced library for the purpose of estimating the effect of NBTI for a pre-defined lifetime of the product, a three-dimensional LUT delay library could be built considering the signal probability SP(0) as the third dimension along with input transition and output load capacitance.

The library file contains four look-up tables: two for propagation delays (rising and falling), and two for transition delays (rising and falling) that are used as input parameters to calculate the propagation delay for the next cell. To introduce signal probabilities SP(0) into the library, we used HSPICE simulation to generate the timing delays. For example, for the inverter gate of the Synopsys 90nm technology library, Figure 3.9 shows the output rising propagation delays when SP(0) is barely stressed and Figure 3.10 shows the output rising propagation delays when SP(0) is almost totally stressed. The results show up to 81% difference between the the stressed and relaxed gates. In the Appendix B, the full generated tables of the extended library for the main logic gates (invertor, nand and nor gates) has been presented.

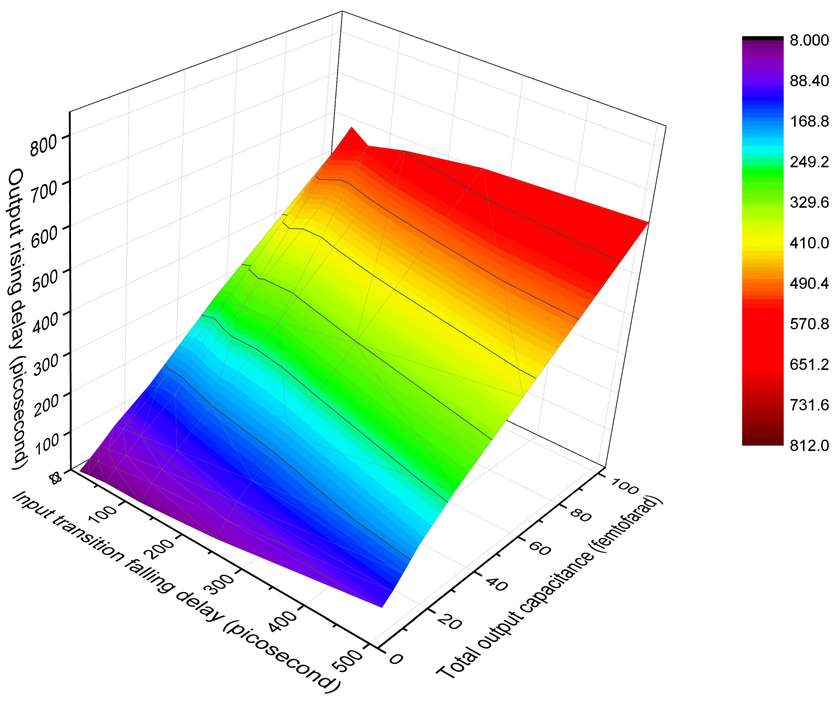


Figure 3.9: Output rising propagation delays for the 90nm Synopsys gate (INVX0) after 10 years when SP(0) = 0.01.

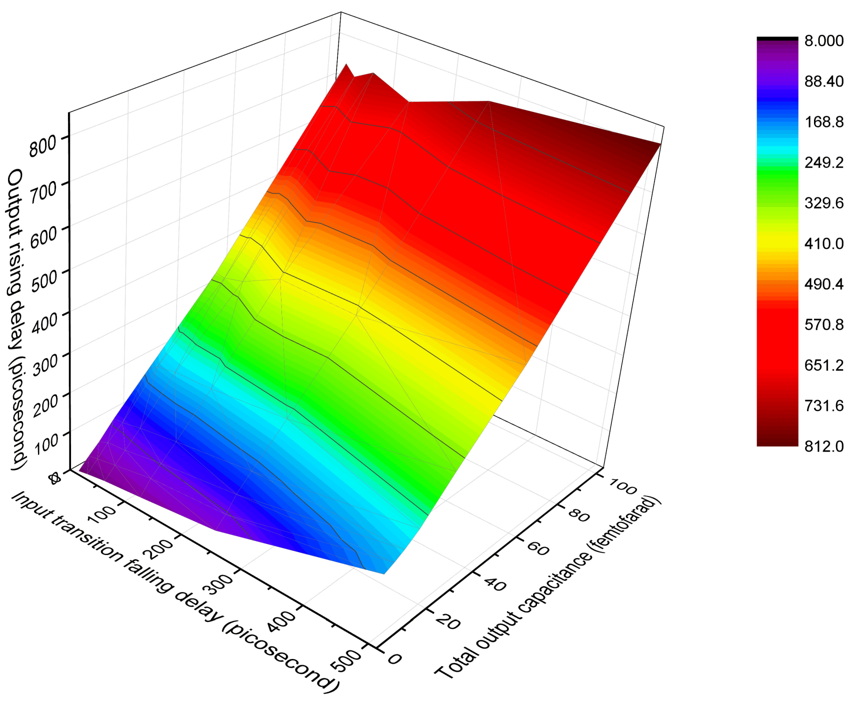


Figure 3.10: Output rising propagation delays for the 90nm Synopsys gate (INVX0) after 10 years when SP(0) = 0.99.

In general, the following steps are needed to extract these tables:

1. For each gates in the netlist of the given technology, the transistor-level of these gates need to be built using HSPICE
2. Define the number of years that required in which this extended library can estimate the delay degradation (e.g., in our case, 10 years).
3. The threshold voltage increasing for a range of input signal probability are calculated using RD model presented in Section 2.2.3.1. In our case, the signal probability of a range from 0.01 to 0.99 of step 0.1 has been considered[1].
4. Prepare input transitions using HSPICE to have at least one rising transition ('0' to '1') and one falling transition ('1' to '0').
5. After having the transistor-level of the gates, range of the possible threshold voltages, input transition delays and output capacitances are injected in a nested loops in HSPICE and generate the propagation gate delays and the transition delay of the output.
6. Save the lookup tables in files to be used in the design or post design analysis.

These tables could be used during the design stage of the logic circuits or could be used after designing for analysis and estimating the degradation of the circuits.

3.4.1 Analysis Setup and Evaluation

An example circuit shown in the Figure 3.11 has been used to extract its delay using the generated tables. The input transitions for the gates are the output transition of the previous gate that need to be calculated using the generated library. The input transition for the first gate has been assumed to be 16 picosecond. The output load capacitances C_b, C_c and C_d has been defined to be 104, 52 and 26 femtofarad, respectively. To propagate the delay from the input to the output, the side-inputs (off-path inputs) of the gates (e.g. in our example, NOR2X0 needs a '0' on the off-path input of the gate to propagate the transition on the on-path input, also, NAND2X0 needs a '1' on the off-path input to propagate the transition on the on-path input. We have here one path starts at net (a) and end with net (d) as shown in the Figure 3.11.

Considering the two scenarios of the net (a) when it is have falling and rising transition, each time, we calculated the time-zero propagated path delays from the standard 90nm Synopsys library and the 10-years path delays from the extended library. The available

[1]The number of steps required for SP(0) determine the complexity of the delay calculation using the static ageing analyser. Therefore, some specific steps need to be calculated and any other intermediate values could be either taken as the worst case or interpolated from two points.

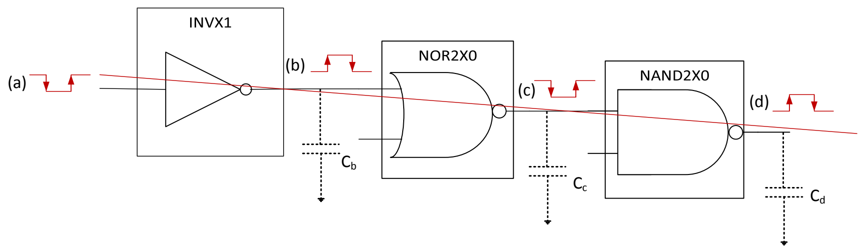


Figure 3.11: Example of circuit instantiate three logic gates from 90nm Synopsys technology.

static timing analyses tools (e.g., Primetime or Design Compiler) are not designed to read the extended library for ageing, so, the gate and path delay has been calculated using the same procedure that commercial tools use as fellow:

```
Data: Technology files contain delay data extracted earlier for all gates; Gates
      gates[i] that construct a path with its estimated output capacitance C[i];
      Primary inputs transition delays (Td); Input transition state (Tr_Inputstate);
Result: Path delay (Pd);
Pd = 0;
while i = zero do
    if Tr_Inputstate is 'falling' then
        calculate the output transition state (Tr_Outputstate);
        if Tr_Outputstate is 'falling' then
            Pd = Pd+ ReadPropagationDelay('falling',Td,C[i]);
            Td = ReadTransitionDelay('falling',Td,C[i]);
        else
            Pd = Pd+ ReadPropagationDelay('rising',Td,C[i]);
            Td = ReadTransitionDelay('rising',Td,C[i]);
        end
    else
        do the same as above but exchanging 'falling' with 'rising';
    end
end
ReadPropagationDelay(Tr_Outputstate,Td,C[i] );
# Function to read propagation delay from library delay tables
ReadTransitionDelay((Tr_Outputstate,Td,C[i] );
# Function to read transition delay from library delay tables
```

Algorithm 1: Procedure to calculate path delay from a technology library tables.

First, using the standard 90nm technology library from Synopsys, the path delay calculated using the worst case corner. when the input transition state was 'rising' the path delay was 0.69 nanosecond and when the input transition state was 'falling' the path delay was 0.91 nanosecond. Then, using the generated library that considers the signal probability of the input to the gate, the results for the gates delay and the total path delay shown in Table 3.3 when the input (a) is 'rising'. Similarly, in the results in Table 3.4 when the input (a) is 'falling'. The results for this example is presented not to prove that what is the best or worst input signal probability SP(0) but just to demonstrate the procedure of using the ageing-induced lookup tables. In the case of input (a) is 'falling', we have more degradation with increasing the signal probability of the input because we will have in this example two out of three gates highly stressed with NBTI that effect mainly on the rising delay of the transistors and that two gates are propagating rising states ('0' to '1').

Table 3.3: Propagation and transition delays (in picosecond) for the nets in the Figure 3.11 when the input (a) is rising from '0' to '1' extracted from the ageing-induced lookup tables.

net(a) rising	*net(b) falling*	*net(c) rising*	*net(d) falling*	
SP(0)	**propagation delay/transition delay**	**propagation delay/transition delay**	**propagation delay/transition delay**	**Total path delay**
0.01	339.35/693	225.14/701	160.10/315	724.59
0.5	339.26/694	226.60/640	159.33/301	725.19
0.99	339.18/694	226.60/560	158.55/295	724.33

Table 3.4: Propagation and transition delays (in picosecond) for the nets in the Figure 3.11 when the input (a) is falling from '1' to '0' extracted from the ageing-induced lookup tables.

net(a) falling	*net(b) rising*	*net(c) falling*	*net(d) rising*	
SP(0)	**propagation delay/transition delay**	**propagation delay/transition delay**	**propagation delay/transition delay**	**Total path delay**
0.01	531/849	230.93/341	250.12/315	1012.05
0.5	653/1020	238.16/519	327.89/378	1219.05
0.99	732/1090	242.51/530	560.63/414	1535.14

We extract signal probabilities (duty cycles) of the nets from different workloads using the MiBench benchmark as workloads on a simulated OpenRISC 1200 processor [2] (see Table 3.5 and Figure 3.12). To obtain the duty cycles (signal probabilities SP(0)) of the nets for a specific application, we obtain SP(0) from the VCD file. The results show the likely stress probability within the nets with different signal probability highly related to the architecture of the circuit Figure 3.12). However, using these signal probabilities to redesign the circuit could come with same number of stressed nodes in a different location. So, this technique could be used to analyse the ageing degradation rather than using it for optimising the circuit for lowering stress. In the following chapter we will show that controlling the circuit degradation by controlling the stress itself.

[2]The soft source code for OpenRISC 1200 processor is available as open source in RTL level that easily synthesised using the available technology to obtain the gate-level. ARM M0 is available for academic research only at gate-level which make it difficult to identify the components in this processor.

Table 3.5: Signal probabilities percentages for MiBench workloads

	Automotive		Telecomm			Network	Secuirty
SP(0) Range	qsort	susan	adpcm	crc	fft	dijkstra	sha
0-0.09	26%	20%	24%	20%	24%	20%	20%
0.1-	3%	3%	0%	3%	0%	3%	4%
0.2-	4%	3%	2%	2%	2%	3%	3%
0.3-	2%	2%	2%	2%	2%	1%	1%
0.4-	3%	3%	2%	2%	2%	3%	3%
0.5-	7%	5%	4%	5%	4%	4%	5%
0.6-	4%	2%	3%	3%	3%	2%	2%
0.7-	9%	6%	4%	6%	4%	3%	5%
0.8-	4%	6%	1%	7%	1%	8%	6%
0.9-1	39%	51%	58%	51%	58%	53%	51%

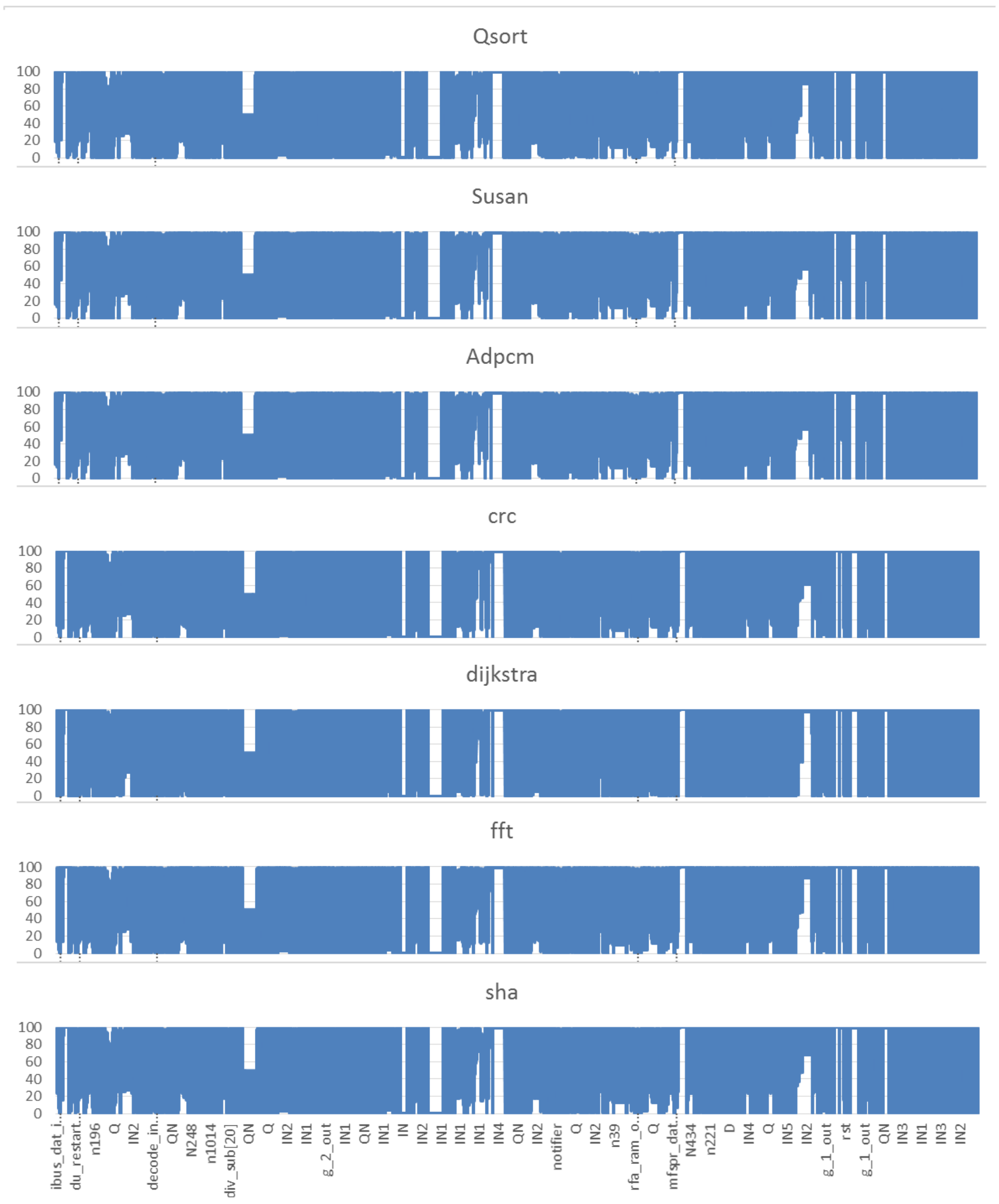


Figure 3.12: Horizontal axis: compressed information about the net names of OpenRISC Processor, vertical axis: percentages of signal probabilities SP(0) for MiBench workload.

3.5 Conclusion

This chapter has presented analyses on the stress state (signal probability) after being extracted from high-level workload. The high-level stress generated by the workload could be propagated down to the signal probabilities at the gate-level. Also, two case studies has been presented for using the signal probability extracted from the high level of abstraction (e.g., program) of the processor to analyse the NBTI ageing effect on the low level of abstraction (e.g, gate level). Firstly, the signal probability extracted used to analyse the critical path generation for aged circuit and showed how important to consider the real stress to analyse the ageing on the critical timing paths of the logic circuit. Secondly, we showed that these signal probabilities could be used in the design stage of digital system and during the synthesis phase that depends on the technology used by extended the libraries of the technology to include the signal probabilities of the gate-level design. By including the signal probability along with the input transitions, and the output load, the logic synthesis process could estimate the timing at the gate level. This approach could make the logic synthesis process estimate NBTI-induced delays more accurate during the design phase. However, the problem of this approach if it used for mitigation is that the ageing is considered at the design stage of the processor, while the stress state is only available when the whole system is fabricated and run, making this approach very complex to be implemented for mitigation but possible for analysis. For this reason, we will present in the following chapters how we could reduce the ageing stress propagated from the application level to the transistor level as a mitigation technique for the combinational and the memory parts of the processor.

Chapter 4

BTI Mitigation for the Combinational Logic Circuits of the Processor by Anti-Ageing Software Patterns

In chapter 3, the BTI stress state has been analysed and identified. In this chapter, we will try to reverse this stress by applying high-level workloads as anti-ageing patterns into the stressed component. This chapter presents a time-redundant technique to mitigate negative and positive bias temperature instability (NBTI/PBTI) ageing effects on the combinational units of a processor.

We have analysed the sources and effects of ageing from the device level to the instruction set architecture (ISA) level, and have found that an application may stress the critical paths in such a way that the combinational circuit has half of its nodes always NBTI-stressed. To mitigate this behaviour, we propose an application-level solution to balance the stress and put the timing-critical gates of the critical path into a relaxed (balanced) mode. The results show that the lifetime of the system can be doubled by applying anti-ageing patterns at the high level of abstractions and during the idle time of a processor system.

The starting hypothesis is that a combinational circuit needs to be actively relaxed to recover from stress, rather than simply doing nothing during idle periods. In modern applications, processors tend to have many short idle periods; thus, simple power gating would not be a good solution for stress optimisation [83]. During normal operational periods, the input states of the transistors may be constant, leaving the transistors stressed. NBTI/PBTI could then be mitigated by applying anti-ageing stimuli to the critical paths at the software-level. Running a program on the processor for a non-functional purpose has been used in on-line testing (i.e., software-based self-test (SBST) methods) as this does not require modification of the hardware design [84].

The main objective of this chapter is to design a high-level technology-independent mitigation technique to balance NBTI/PBTI effects on the combinational part of the processor. This technique bring the circuit into a recovery state by changing the nodes that are BTI-stressed to BTI-relaxed.

The organisation of this chapter is as follows. In section 4.1, the dependency of the data on stressing the processor is discussed. In section 4.2, NBTI/PBTI analysis is presented. The proposed technique for generating and applying the anti-ageing patterns is presented in section 4.3. The evaluation and discussion of running the balancing program are given in section 4.4. Section 4.5 concludes the chapter.

4.1 Data Dependency of Ageing-Induced Degradation of the Processor

Program data determines whether the nodes of the processor are stressed or not. The data includes both opcodes and operands. Firouzi et al. [85] looked at possible NOP instructions in the MIPS processor to reduce ageing. As well as the standard NOP instruction (sll r0, r0, 0) they considered other instructions that had no result (e.g., adding zero) to minimise stress. They proposed software and hardware techniques to assign the best input vector for these NOP instructions. This method would only be helpful, however, if the rate of NOP instructions is high with respect to the total number of operational instructions. To test this hypothesis, we ran different benchmarks from the MiBENCH suite [86] for two different architectures on the GEM5 simulator [87]. Figure 4.1 shows that the number of NOP instructions on the MIPS processor is significant, while on the ARM architecture it is negligible.

Data from other paths can also stress the critical path. For example, let us assume that the adder is in the critical path, and that during the execution of other operations data is routed to the adder, even though the result is not used. From a BTI perspective the critical path through the adder will be stressed. It has been claimed that the core of a processor can be brought to a failing state by executing a malicious program to age the circuit [16]; however, this does not consider the signal probabilities of intermediate nodes. In practice, it is not possible to put all critical path nodes into a fully relaxed state (i.e., a Signal Probability of zero, SP(0) = 0%) or a fully stressed state (SP(0) = 100%) as illustrated in Figure 4.2. On the other hand, it is possible to balance the stress by controlling the signal probabilities during the idle time of the processor.

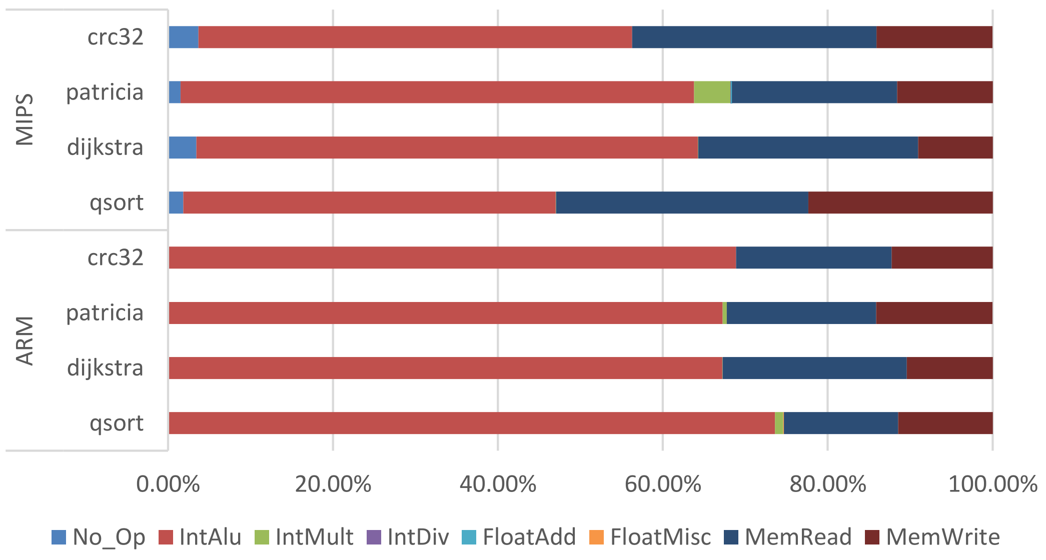


Figure 4.1: Class of instruction percentages for different benchmarks on MIPS and ARM.

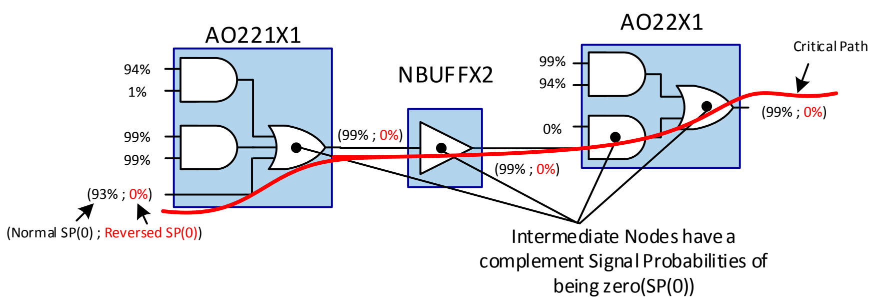

AO221X1
NBUFFX2
AO22X1
Critical Path
94%
1%
99%
99%
(93% ; 0%)
(Normal SP(0) ; Reversed SP(0))
(99% ; 0%)
(99% ; 0%)
99%
94%
0%
(99% ; 0%)
Intermediate Nodes have a complement Signal Probabilities of being zero(SP(0))

Figure 4.2: Zero-signal probabilities distribution on the critical path sub-circuit with normal and reversed state.

4.2 NBTI/PBTI Stress Analysis

4.2.1 Ageing-Sensitive Critical Path Selection

The selection of paths to reverse the stress needs to consider both the initial path delays from the post-synthesis analysis and the gate types in the paths. A non-critical path at time zero could become a critical path after a number of years because paths degrade according to different factors, for example, duty cycle, temperature, frequency and circuit topology. Estimating the path most sensitive to ageing depends on model parameters that would not be available until the system has been fabricated and tested in the target environment. Therefore, in this research, we have avoided using an ageing model to define the criteria for selecting specific paths that are potentially vulnerable to ageing. Instead, we define a threshold (θ) for the ageing-critical path delay. For example, the maximum critical path degradation has been measured by creating aged-aware gate model for different benchmark circuits (ISCAS85) for 10 years and found that the path delay increased for these circuits are between 12.3% and 19.5% [88]. The experiment conducted for 90nm standard cell technology at an effective temperature of 125°C and an effective supply voltage of 1.32V. However, this range of degradation around 20% is technology-dependant but we can define ageing sensitive critical paths for this technology size and for smaller technologies sizes, further analysis is required Thus, those paths that have slack in the range of zero to ($\delta_0 \times \theta$), inclusivse, or have a path delay between δ_0 and $\delta_0(1-\theta)$, inclusive. Where δ_0 is the slack of the path at time zero that could be found using static timing analysis tools (e.g., Design Compiler).

We also consider the effect of process variations on selected paths by defining the worst-case possible path deviation due to the process variations as $\Delta\delta_{pv}$. Then, an ageing- and process- sensitive critical path should be selected if its path delay is in the range:

$$\delta_0 \geq Path\,Delay \geq \delta_0(1 - \theta - \Delta\delta_{pv}) \tag{4.1}$$

These paths have nearly balanced path delays, but they could share instances with the first critical path (for example in an adder, if the carry chain path is shared between nearly-critical paths and the first critical path, then any degradation or reversed degradation on the shared part will also affect the ageing-sensitive critical paths). Alternatively, if the critical paths have instances that are independent from one path to another, then all the ageing sensitive critical paths need to be analysed individually for ageing and possible reversal.

4.2.2 SP(0) Distribution on the Critical Paths of the Processor

Combinational logic circuits may show different degradations in each PMOS transistor because different primary input patterns can lead to different inputs to the CMOS transistors. Some PMOS transistors may degrade more because they have SP(0) of 99%, but others may not degrade if they have SP(0) of 1% at their gates. To date, however, there has been no consideration of SP(0) of the intermediate nodes of the complex gates. For example, in [16, 89], the analysis was done only for the input transistors of the gates. In the OR gate of Figure 4.3, if IN1 is at “1”, Q would be “1” regardless of IN2, but there is a node, QN, inside the OR gate that would be stressed.

To measure the delay degradation on each net of the critical path, we need to consider the SP(0) of each input and of the internal nodes of the gate cells. We used the OpenRISC core for this analysis. Firstly, synthesis was done using Synopsys Design Compiler with the full set of cells available in the 90nm Synopsys library, including those cells that have internal nodes. There are 2888 critical paths in the OpenRISC

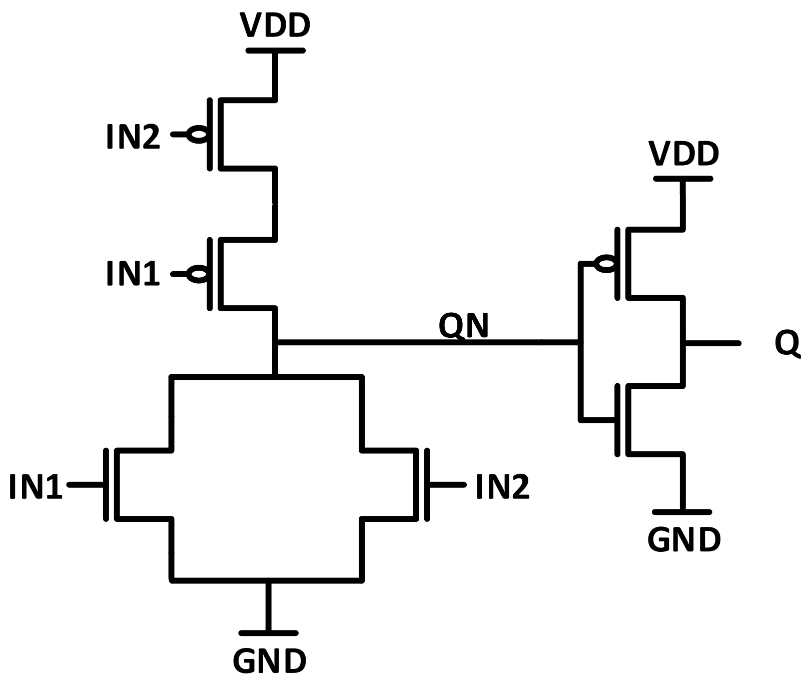


Figure 4.3: CMOS circuit for OR gate (NOR + INVERTER).

core, but various critical paths pass though the same gate cells. For example, the first 100 critical paths share more than 92% of the cells in the most critical path (see Table 4.1). Thus, if there is any degradation in the shared part, it would affect all these critical paths. Moreover, the average SP(0) for the first 100 critical paths is around 80% when executing a “Hello World” program. This means that the program will stress the critical path.

In this example, the nodes are totally NBTI stressed because the probabilities of signals being zero are close to 100%. However, this does not consider the hidden nodes of the compound gates and so the average SP(0) is not correct. These hidden nodes would have complementary values and therefore have no NBTI stress but could have PBTI stress. In other words, a circuit with SP(0) close to 0% could have hidden nodes with SP(0) close to 100%. To calculate a more accurate figure, we ran different instructions on a processor synthesised using only cells that have only one-level of transistors, in order to avoid any hidden nodes, as given in Table 4.3 and Table 4.2. The SP(0) at each node is generally the complement of the SP(0) of the previous node and the average SP(0) is around 50%. Therefore, the objective should be to reverse these signal probabilities to obtain signal probabilities that are as balanced as possible (around 50%), rather than reducing one signal probability to avoid NBTI stress. This example is only used to show the signal probabilities of the hidden nodes, as this case is not feasible in real synthesis.

Table 4.1: SP(0) distribution on the critical paths of the OpenRISC processor for Hello World program using compound gates.

	1st critical path		2nd critical path		3rd critical path		10th critical path		100th critical path	
	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)
1	DFFX1	50%	DFFX1	50%	DFFX1	50%	DFFX1	50%	DFFX1	50%
2	DFFX1	98%	DFFX1	98%	DFFX1	98%	DFFX1	98%	DFFX1	98%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
65	AO22X1	93%	AO22X1	93%	AO22X1	93%	AO22X1	93%	AO22X1	93%
66	OR2X1	93%	OR2X1	93%	OR2X1	93%	OR2X1	93%	XOR3X1	94%
67	AO22X1	93%	AO22X1	93%	AO22X1	93%	AO22X1	93%	AOI22X1	5%
68	XOR3X1	94%	XOR3X1	94%	XOR3X1	94%	XOR3X1	94%	NAND4X0	94%
69	AOI22X1	5%	AOI22X1	5%	AOI22X1	5%	AOI22X1	5%	AO221X1	93%
70	NAND4X0	94%	NAND4X0	94%	NAND4X0	94%	NAND4X0	94%	NBUFFX2	99%
71	AO221X1	93%	AO221X1	93%	AO221X1	93%	AO221X1	93%	AO22X1	14%
72	NBUFFX2	99%	NBUFFX2	99%	NBUFFX2	99%	NBUFFX2	99%	DFFX1	14%
73	AO22X1	99%	AO22X1	77%	AO22X1	99%	AO22X1	0%		
74	DFFX1	99%	DFFX1	77%	DFFX1	99%	DFFX1	0%		
Average SP(0)		83%		82%		83%		80%		80%

Table 4.2: SP(0) distribution on the 100th critical path of the OpenRISC processor for different instructions

		addi rD,rA,I				movhi rD,I			
	Cell type	$I = 0000_H$	5555_H	$AAAA_H$	$FFFF_H$	0000_H	5555_H	$AAAA_H$	$FFFF_H$
1	DFFX1	50%	50%	50%	50%	50%	50%	50%	50%
2	DFFX1	99%	99%	99%	99%	99%	99%	99%	99%
3	NAND2X0	0%	0%	0%	0%	0%	0%	0%	0%
4	NAND2X0	0%	0%	0%	0%	99%	99%	99%	99%
5	NAND2X0	99%	99%	99%	99%	0%	0%	0%	0%
6	NAND4X0	0%	0%	0%	0%	99%	99%	99%	99%
7	NOR2X0	99%	99%	99%	99%	0%	0%	0%	0%
8	AOBUFX1	99%	99%	99%	99%	0%	0%	0%	0%
9	INVX0	0%	0%	0%	0%	99%	99%	99%	99%
10	NAND2X0	0%	0%	0%	0%	0%	0%	0%	0%
11	NAND2X0	40%	40%	40%	40%	99%	99%	99%	99%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
142	NAND2X0	0%	40%	40%	40%	0%	0%	21%	23%
143	NAND4X0	99%	59%	59%	59%	99%	99%	78%	76%
Average SP(0)		50%	49%	49%	49%	49%	50%	49%	49%

4.2.3 Impact of Instruction/Program Level Workload on the Stress Probabilities

To study the effect of different instructions on the stress of the critical or nearly-critical paths, we ran two different instructions with four different operands each, to determine whether the opcode or the data have a significant impact. The synthesis is done using only cells that have no hidden nodes and the SP(0) at each node is generally the inverse of that of the previous node in the path. For the 155 nodes in the critical path, the average SP(0) was around 50%, as can be seen from the symmetry of the histograms in Figure 4.4. The average SP(0) of the paths does not change significantly with the

Table 4.3: SP(0) distribution on the first critical path of the OpenRISC processor for different instructions.

		addi rD,rA,I				movhi rD,I			
	Cell type	I=0000_H	5555_H	AAAA_H	FFFF_H	0000_H	5555_H	AAAA_H	FFFF_H
1	DFFX1	50%	50%	50%	50%	50%	50%	50%	50%
2	DFFX1	99%	99%	99%	99%	99%	99%	99%	99%
3	NAND2X0	0%	0%	0%	0%	0%	0%	0%	0%
4	NAND2X0	0%	0%	0%	0%	99%	99%	99%	99%
5	NAND2X0	99%	99%	99%	99%	0%	0%	0%	0%
6	NAND4X0	0%	0%	0%	0%	99%	99%	99%	99%
7	NOR2X0	99%	99%	99%	99%	0%	0%	0%	0%
8	AOBUFX1	99%	99%	99%	99%	0%	0%	0%	0%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
150	NAND3X0	50%	50%	50%	50%	34%	34%	34%	35%
151	NAND2X0	0%	0%	0%	0%	0%	0%	0%	0%
152	INVX0	99%	99%	99%	99%	99%	99%	99%	99%
153	NAND3X0	0%	0%	0%	0%	0%	0%	0%	0%
154	NAND2X0	99%	99%	99%	99%	99%	99%	99%	99%
155	DFFX1	99%	99%	99%	99%	99%	99%	99%	99%
Average SP(0)		49%	49%	49%	49%	48%	49%	48%	48%

opcodes or operands. However, the signal probability distributions on the critical path do depend on the opcode and operands, as shown in Figure 4.4("movhi rD,5555_H"), where the signal probabilities tend to the extremes (10% > SP(0) > 90%) even with symmetrical distributions that stress the critical paths with both NBTI and PBTI. Moreover, when the specific patterns do a change in some limited number of nodes, again that will be inverted in other nodes and the average SP(0) will not be affected. We run different benchmarks from the MiBench benchmarks to illustrate how different programs stress different PMOS transistor in the critical path or nearly-critical paths. As can be seen in Table 4.4 and Table 4.5, although different benchmarks have been executed, the SP(0) on the specific nodes did not changed significantly. Similarly, different MiBench benchmarks show nearly symmetrical average stress, as can be seen in Figure 4.5. Therefore, it would be desirable to reduce the number of nodes at the extremes of these histograms (e.g., 25% > SP(0) > 75%).

Hence, we conclude that a single instruction or program will not relax or stress 100% of the critical paths, but that a program-level solution could relax or stress some specific nodes. In other words, if there are some nodes in the processor that could face a continual stress while others are not stressed, it is possible to balance that effect at the application level.

Table 4.4: SP(0) distribution on the first critical path of the OpenRISC processor for different programs

	Cell type	hello world	bit cnts	basic math	jpeg
1	DFFX1	50%	50%	50%	50%
2	DFFX1	48%	47%	41%	47%
3	NAND2X0	22%	24%	20%	24%
4	NAND2X0	42%	45%	39%	45%
5	NAND2X0	40%	39%	34%	39%
6	NAND4X0	30%	29%	28%	29%
7	NOR2X0	75%	76%	76%	76%
8	AOBUFX1	75%	76%	76%	76%
9	INVX0	24%	23%	23%	23%
⋮	⋮	⋮	⋮	⋮	⋮
154	NAND2X0	36%	28%	25%	28%
155	DFFX1	36%	28%	25%	28%
Average SP(0)		45%	45%	45%	45%

Table 4.5: SP(0) distribution on the 100th critical path of the OpenRISC processor for different programs

	Cell type	hello world	bit cnts	basic math	jpeg
1	DFFX1	50%	50%	50%	50%
2	DFFX1	48%	47%	41%	47%
3	NAND2X0	22%	24%	20%	24%
4	NAND2X0	42%	45%	39%	45%
5	NAND2X0	40%	39%	34%	39%
6	NAND4X0	30%	29%	28%	29%
7	NOR2X0	75%	76%	76%	76%
8	AOBUFX1	75%	76%	76%	76%
9	INVX0	24%	23%	23%	23%
⋮	⋮	⋮	⋮	⋮	⋮
142	NAND2X0	4%	3%	4%	3%
143	NAND4X0	87%	89%	90%	89%
Average SP(0)		47%	46%	46%	46%

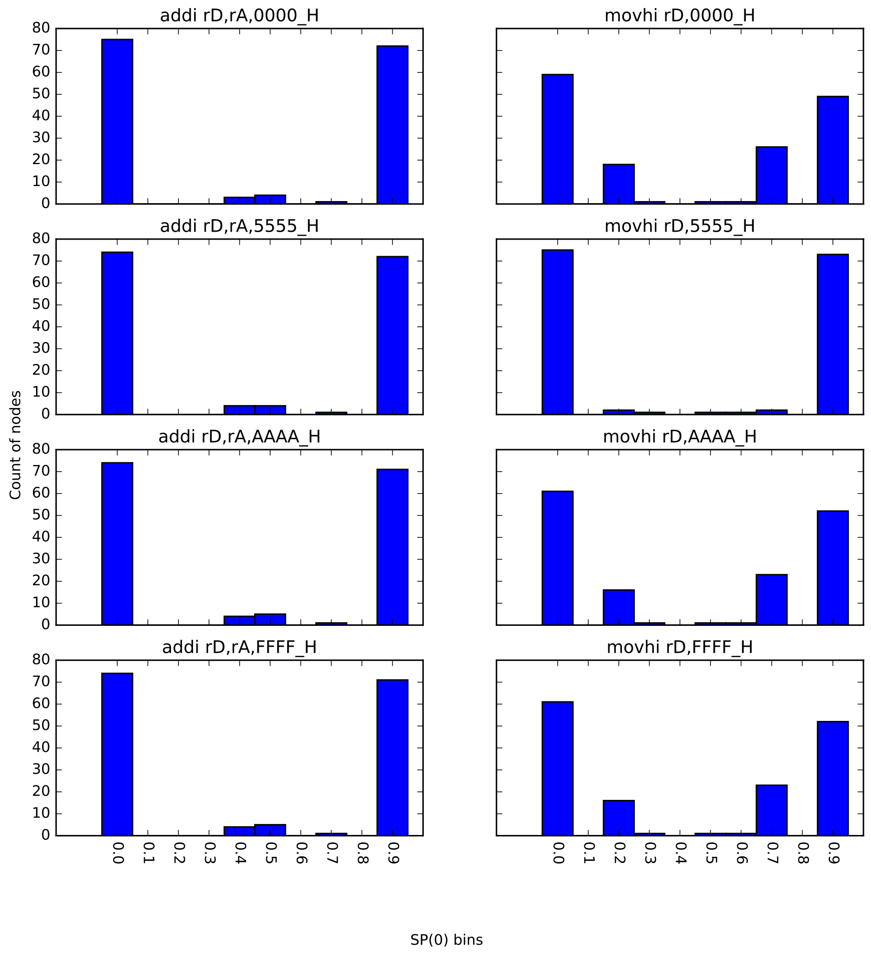


Figure 4.4: SP(0) distribution on the first critical path of the OpenRISC processor for different instructions.

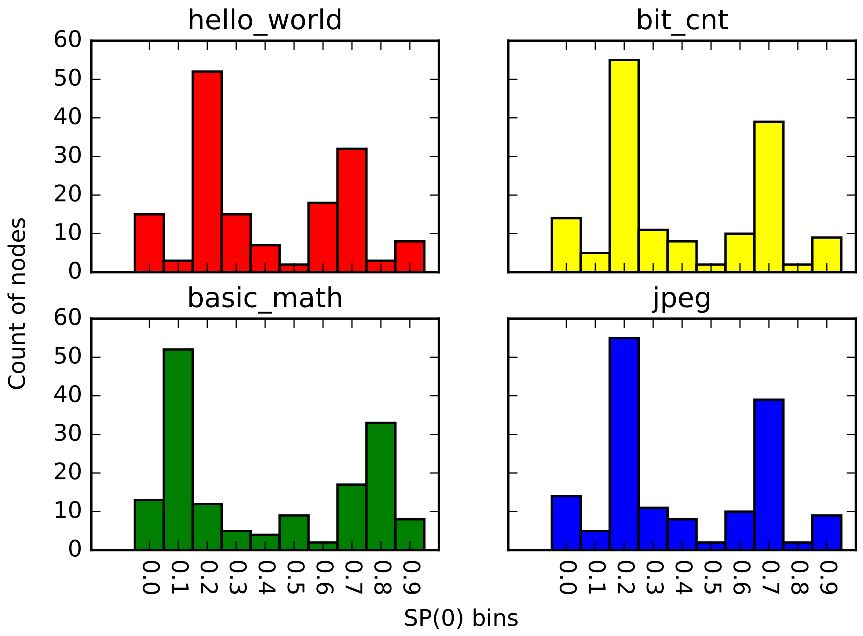


Figure 4.5: SP(0) distribution on the first critical path of the OpenRISC processor for different programs.

4.2.4 Gate Level Stress Balancing

We consider balancing the signal state of basic logic gates (inverter, NAND and NOR) compared with inverting the signal probability. We will examine how inverting the signal probability would affect the ageing degradation. We have used HSPICE for simulating path delays and modelled the NBTI using MOSRA Level 3 considering two different cases:

- CASE A: Unbalanced stressed nodes – the nodes of the critical paths are either significantly NBTI-stressed (SP(0) greater than 75%) or significantly NBTI-unstressed (or PBTI-stressed) (SP(0) less than 25%).
- CASE B: Balanced stressed nodes – the nodes of the critical path have SP(0) around 50%.

For the inverter, we simulated the degradation of a path of two inverters over ten years using the cases discussed above. The results show an advantage of 23.17% in the path delay and more than 50% in terms of time as shown in Figure 4.6.

For NAND and NOR gates, the same simulation as for the inverter is implemented. However, we consider two additional dependencies: the signal probabilities of the secondary inputs of the gates, and the input pin order. The results show that swapping input pins could decrease the advantage obtained from balancing the signal probabilities. Also, the signal probabilities of the secondary input of the gate will not significantly affect the benefits of balancing the signal probabilities over the critical path, Table 4.6 and Table 4.7.

Table 4.6: Path delay degradation advantages for a chain of two NAND2 gates considering balanced and unbalanced signal probabilities.

SP(0) of the 2nd input	Swap input order	CASE A degradation	CASE B degradation	Advantage over path delay	Advantage over years
50%	NO	9.27%	7.25%	21.75%	6 years
50%	YES	9.49%	7.62%	19.73%	5 years
99%	NO	7.35%	5.82%	20.80%	5 years
99%	YES	8.50%	6.88%	19.10%	5 years
1%	NO	10.38%	8.34%	19.69%	5 years
1%	YES	10.46%	8.18%	21.82%	5 years

We also considered how the anti-ageing patterns affect the remaining paths of the circuit. To answer this fundamental question, we examined the proposed technique on a two-bit adder. In this example, there is one target critical path and two nearly-critical paths as shown in Figure 4.7. In this example, we extracted the critical paths list after synthesising the circuit using Design Compiler. Again, we used the MOSRA Level 3 model in HSPICE simulations to model the degradation of the circuit using the two above-mentioned cases. The results show that for all paths, there will be an advantage

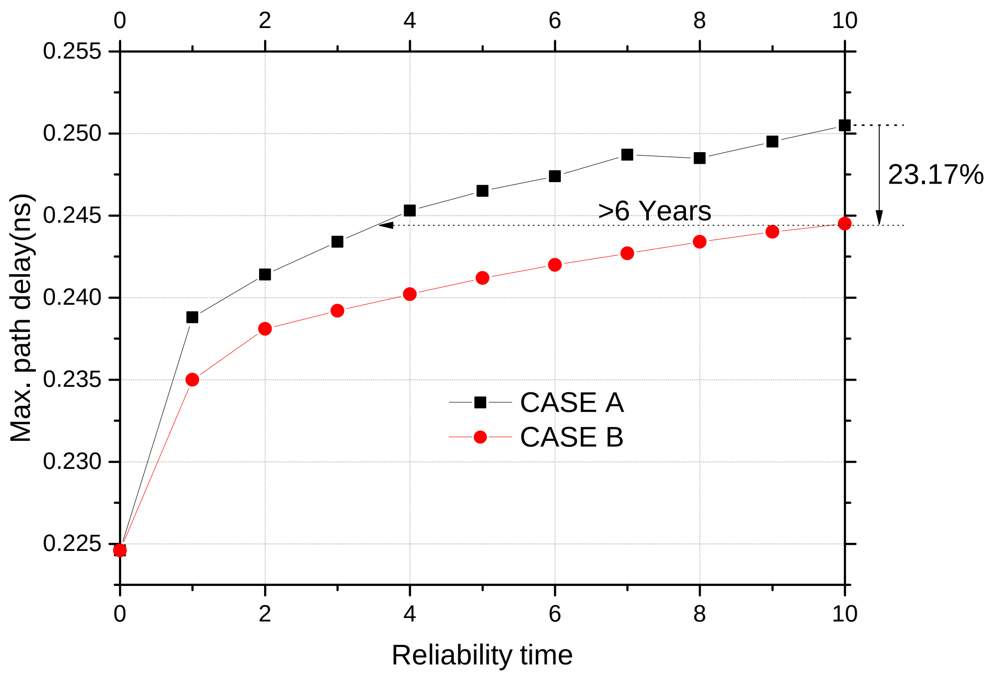


Figure 4.6: Path delay degradation for a chain of two inverters considering balanced and unbalanced signal probabilities.

of up to 50% in the expected lifetime from balancing the signal probabilities in the critical path (see Figure 4.18). Figure 4.18 also shows that nearly-critical paths share more than half of their nodes with the target critical path. Thus, any advantage in balancing the signal probabilities of the critical path will lead to an advantage in the remaining paths. If the nearly-critical paths do not share nodes with the critical path, then it is possible to control both the critical and the nearly-critical paths in parallel.

Table 4.7: Path delay degradation advantages for a chain of two NOR2 considering balanced and unbalanced signal probabilities.

SP(0) of the 2nd input	Swap input order	CASE A degradation	CASE B degradation	Advantage over path delay	Advantage over years
50%	NO	8.55%	6.81%	20.35%	5 years
50%	YES	9.10%	7.57%	16.80%	4 years
99%	NO	8.55%	6.81%	20.35%	5 years
99%	YES	9.76%	8.77%	10.18%	3 years
1%	NO	5.83%	4.54%	22.08%	6 years
1%	YES	7.38%	5.68%	23.08%	3 years

4.3 Proposed Technique

We propose a two-phase technique to mitigate the BTI ageing effects. In the first phase, anti-ageing patterns (the balanced states) are generated and these patterns are

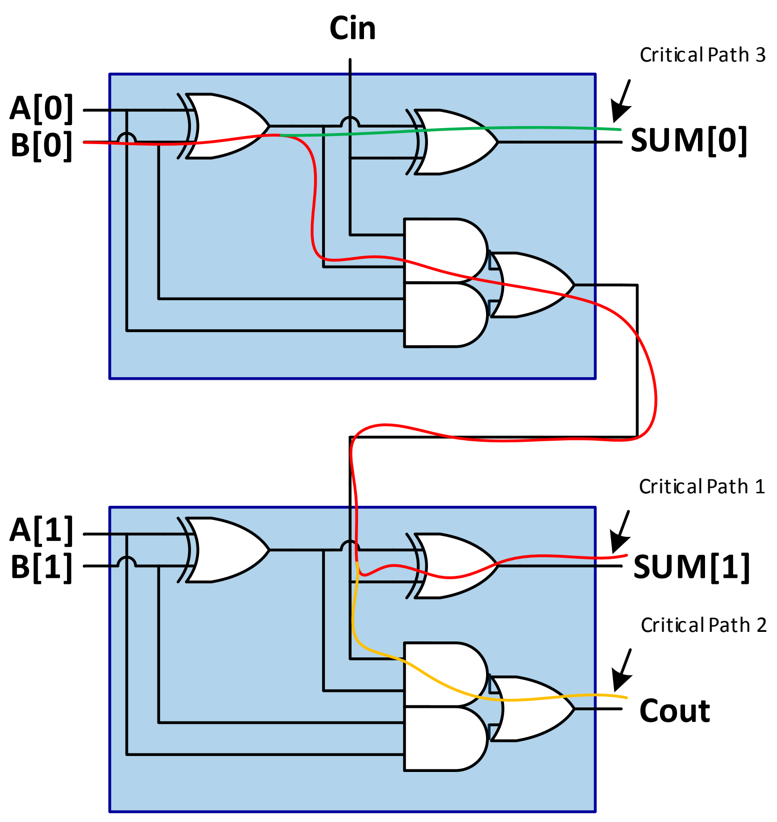


Figure 4.7: The three most critical paths in the two-bit adder.

applied in the second phase by executing an anti-ageing program instead of running a process idle task. The flow of the first phase of the proposed techniques is illustrated in Figure 4.9. Providing the hardware description of the processor, we can generate the netlist/gate-level design using specific technology node and synthesis tool (e.g. Design Compiler). Post-synthesis simulation are required to extract the VCD file by executing program in the simulated processor using for example, ModelSim simulation tool. The VCD file needs to be parsed and translated into SP(0) file using the tool we developed and presented in Chapter 3. The static timing analysis tool are needed to extract the timing critical paths. These paths need to be classified to obtain only potential critical paths as presented in Section 4.2.1. After obtain both the potential critical path and their signal probability, the stress state could be obtain for the nodes that having unbalanced SP(0). This stress state need to be reversed to obtain the reverse stress state (anti-ageing state). Finally, ATPG (Automatic Test Pattern Generator) Tool are required to obtain patterns that able to propagate the reverse stress state into the stressed nodes at the gate level. We find the normal states (i.e., the critical path stress states) of the nodes that need to be balanced by running different benchmarks and instructions. We obtain the signal probabilities of the nets being stressed to

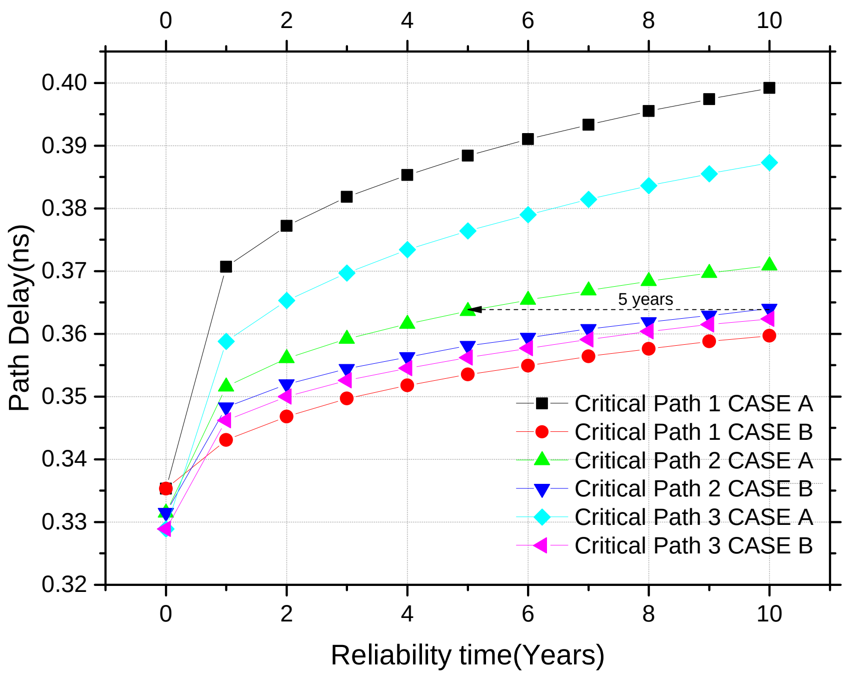


Figure 4.8: Path delay degradation of the most three critical paths in the two-bit adder considering unbalanced and balanced signal probabilities over the critical path.

logic zero (SP(0)) from gate-level simulations of the processor executing benchmarks. We calculate the SP(0) of the critical paths that have slacks less than the predefined maximum path delay degradation.

The second phase of the technique is to balance the effect of BTI by reversing the average signal probabilities by applying anti-ageing patterns to the timing-critical components in the functional unit of the processor during idle states.

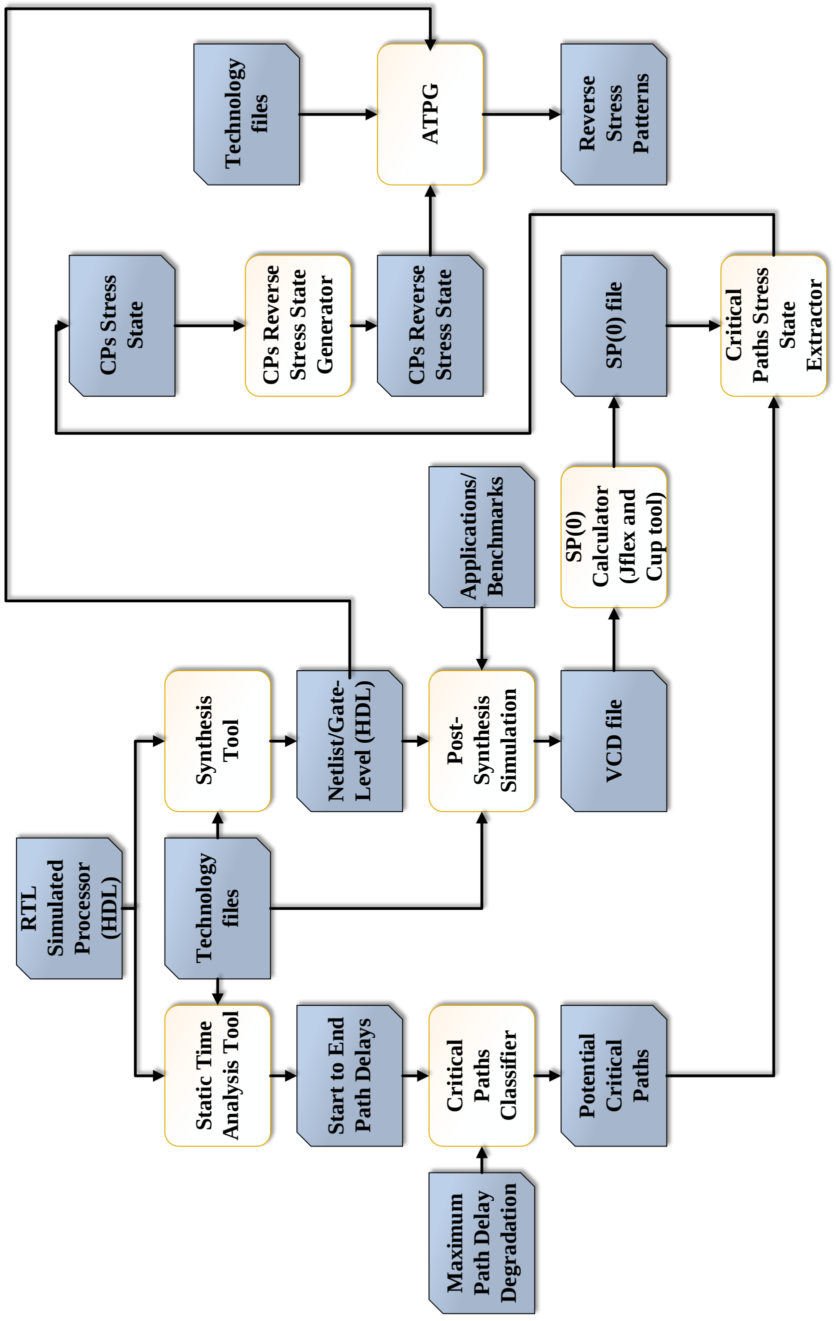

RTL Simulated Processor (HDL)
Technology files
Synthesis Tool
Static Time Analysis Tool
Netlist/Gate-Level (HDL)
Start to End Path Delays
Post-Synthesis Simulation
Critical Paths Classifier
Maximum Path Delay Degradation
Applications/ Benchmarks
VCD file
Potential Critical Paths
SP(0) Calculator (Jflex and Cup tool)
SP(0) file
Critical Paths Stress State Extractor
CPs Stress State
CPs Reverse Stress State Generator
CPs Reverse Stress State
Technology files
ATPG
Reverse Stress Patterns

Figure 4.9: BTI anti-ageing (anti-ageing) patterns generating flow

4.3.1 Case Study: Program-Level NBTI/PBTI Balancing

We synthesised an OpenRISC 1200 processor core using the 90nm Synopsys technology[1]. The VCD file from each post-synthesis simulation contains both the switching activity, that is used to estimate the dynamic power at the design phase, and the signal probability that we used to estimate the BTI effect on performance degradation. From the VCD file, we extracted the SP(0) for all the nets of the processor. To balance the effect of signal probability on the critical path, we need to find input patterns that will invert the signal states. This is effectively the same as generating test patterns for single-stuck faults. We used an ATPG tool (e.g., TetraMAX from Synopsys) to find test patterns for stuck-at-0 faults on nets that have high SP(0) so as to set those nets to ‘1’. Similarly, we generated patterns for stuck-at-1 faults on the nets that have low SP(0) so as to set those nets to ‘0’. Thus, these logic-level nodes that are stressed by ‘0’ (NBTI) or ‘1’ (PBTI) could be reversed by applying reversed stress patterns.

The critical paths of the OpenRISC 1200 processor are in the adder. 38 nodes have an SP(0) greater than 75%; 10 nodes have an SP(0) less than 25%. The ATPG tool found eight test patterns to set these nodes to anti-ageing conditions. Each pattern will apply anti-ageing to one or more nodes, while the full set is needed for every node. As the results given in section 4.4 show the percentages of stressed nodes will be reduced significantly after applying these patterns. Table 4.8 shows these patterns as they would be applied to inputs A[31 .. 0] and B[31 ..0] of the adder.

Table 4.8: OpenRISC anti-ageing patterns.

A[31:0]				B[31:0]			
48	CB	3E	67	D4	40	7A	4C
24	65	9F	2D	EA	20	3D	3C
DA	F9	F1	D1	21	50	64	F2
25	B7	C6	93	C4	E8	48	35
DE	4F	08	A3	F9	CD	6D	DE
BF	58	FA	23	6A	33	27	3F
67	5B	2E	9C	58	C3	65	7F
AC	8C	03	18	A4	DC	93	8D

The OpenRISC 1200 instruction set architecture has only a 16-bit immediate mode. These anti-ageing patterns have 32-bit widths and so are stored in consecutive memory locations starting with address K. The “balancing” program shown in Figure 4.10 transfers K (immediate value) to register 1 for use as an offset address. Then two patterns are loaded from memory to registers 2 and 3. The two patterns are applied to the adder with an ADD operation. The same sequence is applied for the remaining patterns. The program sits in a loop and is run during the idle states of the system.

[1]The technology is not important because the technique depends on reversing the SP(0) rather than estimating the ageing. Hence, it is also independent of the BTI model.

```
        l.movhi r1, K
        # Stimulate the first pattern
        l.lws r2, 0(r1)
        l.lws r3, 1(r1)
        l.add r4, r2, r3
        # Stimulate the second pattern
        l.lws r2, 2(r1)
        l.lws r3, 3(r1)
        l.add r4, r2, r3
        .
        .
        # Stimulate the eighth pattern
        l.lws r2, 14(r1)
        l.lws r3, 15(r1)
        l.add r4, r2, r3
        l.rfe # Return From Exception


```

Figure 4.10: Balancing program

Further optimisation is possible to the above program to reduce the memory access and thus to reduce the power consumption of the running program as shown in Figure 4.11.

```
        # Stimulate the first pattern
        l.movhi r2, 48CB
        l.ori r2, r2, 3E67
        l.movhi r3, D440
        l.ori r3, r3, 7A4C
        l.add r4, r2, r3
        # Stimulate the second pattern
        l.movhi r2, 2465
        l.ori r2, r2, 9F2D
        l.movhi r3, EA20
        l.ori r3, r3, 3D3C
        l.add r4, r2, r3
        .
        .
        # Stimulate the eighth pattern
        l.movhi r2, AC8C
        l.ori r2, r2, 0318
        l.movhi r3, A4DC
        l.ori r3, r3, 938D
        l.add r4, r2, r3
        l.rfe # Return From Exception

```

Figure 4.11: Optimised balancing program

Another consideration is that this program may not have the privileges to run while another program is running. So the scheduler should give the lowest priority to this program and run it when the system is idle. However, if it is decided that this program should run as a routine in response to an interrupt, then the context of interrupted processes needs to be saved. In this case, the program should push the registers onto the stack at the start of the routine and pop them off at the end of the routine to save the context of the interrupted program as shown in Figure 4.12.

```
        # Push r2,r3,r4 onto the stack
        push {r2-r4}
        # Stimulate the first pattern
        l.movhi r2, 48CB
        l.ori r2, r2, 3E67
        l.movhi r3, D440
        l.ori r3, r3, 7A4C
        l.add r4, r2, r3
        # Stimulate the second pattern
        l.movhi r2, 2465
        l.ori r2, r2, 9F2D
        l.movhi r3, EA20
        l.ori r3, r3, 3D3C
        l.add r4, r2, r3|
        .
        .
        # Stimulate the eighth pattern
        l.movhi r2, AC8C
        l.ori r2, r2, 0318
        l.movhi r3, A4DC
        l.ori r3, r3, 938D
        l.add r4, r2, r3
        # Pop r2,r3,r4 and the program counter(pc) from the stack,
    then branch to the new pc.
        pop {r2-r4,pc}

```

Figure 4.12: Optimised balancing program for saving the context of the interrupted program

4.4 Evaluation and Discussion

To evaluate the effect of running the balancing program and how the signal probability will be affected on the critical path, we ran the balancing program along with different benchmarks and varied the percentages of the running time of the balancing from 10% to 50%. To compare results, we calculated the number of stressed nodes as follows:

$$\text{Percentage of stressed nodes} = \frac{\#\text{ stressed nodes}}{\text{critical path nodes}}, \tag{4.2}$$

where the stressed nodes are those critical path nodes that have an SP(0) greater than 75% or less than 25%. As would be expected, to obtain a balanced state for the stressed nodes requires the balancing program to run for 50% of the time. Needless to say, it is not always possible to have this idle time or to add redundant time for the purpose of relaxing BTI. Figure 4.13 shows the effect on the stressed nodes of running the BTI balancing program for different percentages of the overall time.

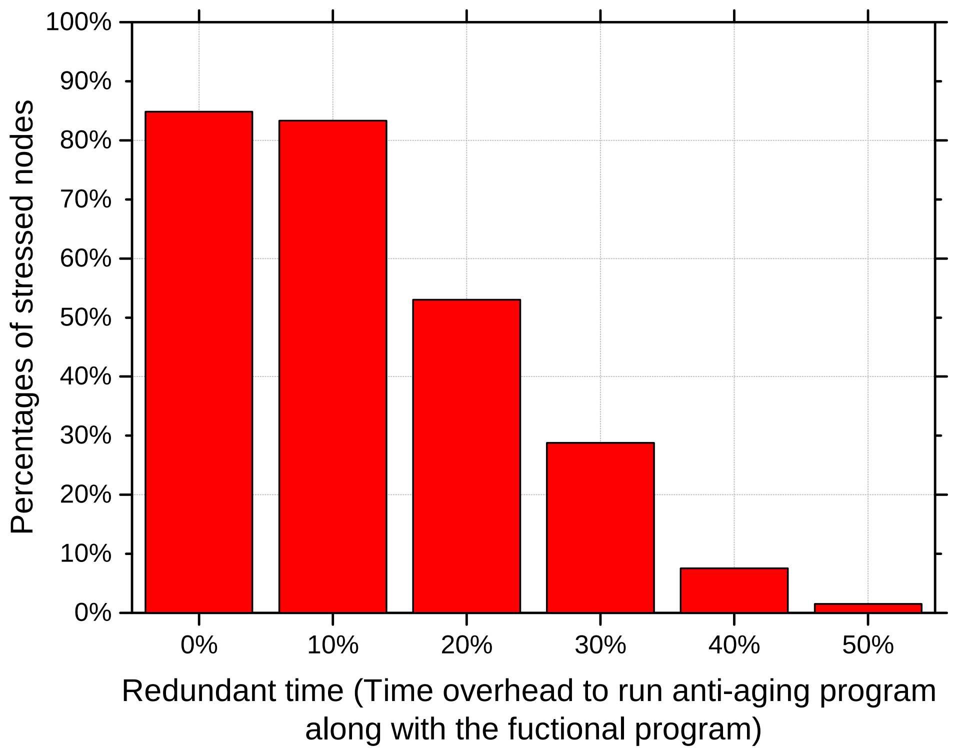


Figure 4.13: The effect on the stressed node percentage of running the BTI balancing program along with a “hello world” program with different run times of the balancing program.

Figure 4.14 shows the effect of running the BTI balancing program for different times on the percentages of the stressed nodes in critical paths of the OpenRISC processor. The results show that balancing one critical path will balance other near-critical paths as they share nodes with the first path. On the other hand, if the near-critical paths do not share many nodes with the targeted critical path, it is possible to apply anti-ageing patterns in the same way for the first critical path independently of the other paths. Although balancing signal probabilities would work with embedded systems that run specific applications, it is also possible to use the technique for a general-purpose processor. Figure 4.15 shows how the percentages of stressed nodes on the first critical path of the OpenRISC processor are reduced when executing the balancing program along with a different program from the MiBench benchmarks.

Next, to verify that the balancing program will reduce the degradation in the path delay of the processor, we simulated the adder using HSPICE and modelled NBTI using the MOSRA Level 3. We stimulated the circuit with two cases:

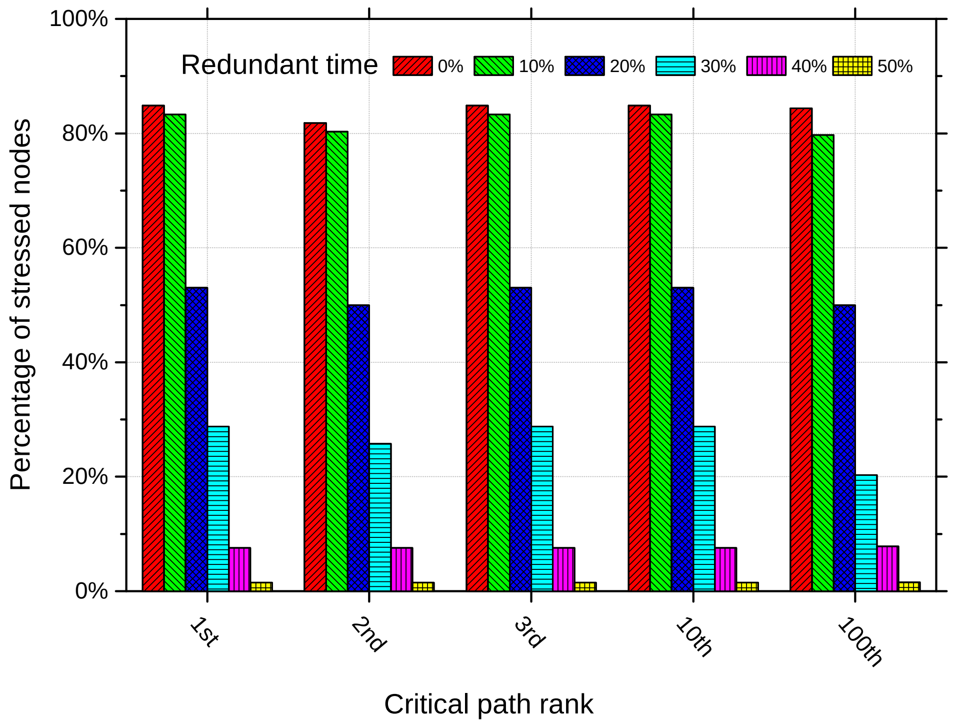


Figure 4.14: The effect of running the BTI balancing program along with a "Hello World" program on the stressed node percentage on different critical paths of the OpenRISC processor with various run times of the balancing (anti-ageing) program.

- CASE A (Normal Stressed Mode): Stress patterns with the equivalent signal probabilities of the Hello World program.
- CASE B (Balanced Mode): Balanced stressed nodes (i.e., the nodes in the critical path having an SP(0) around 50% by running the anti-ageing program along with the Normal Stressed Mode program).

The results show that running the balancing program would decrease the path delay by 20.24% compared with the normal operation and will double the expected lifetime as shown in Figure 4.16.

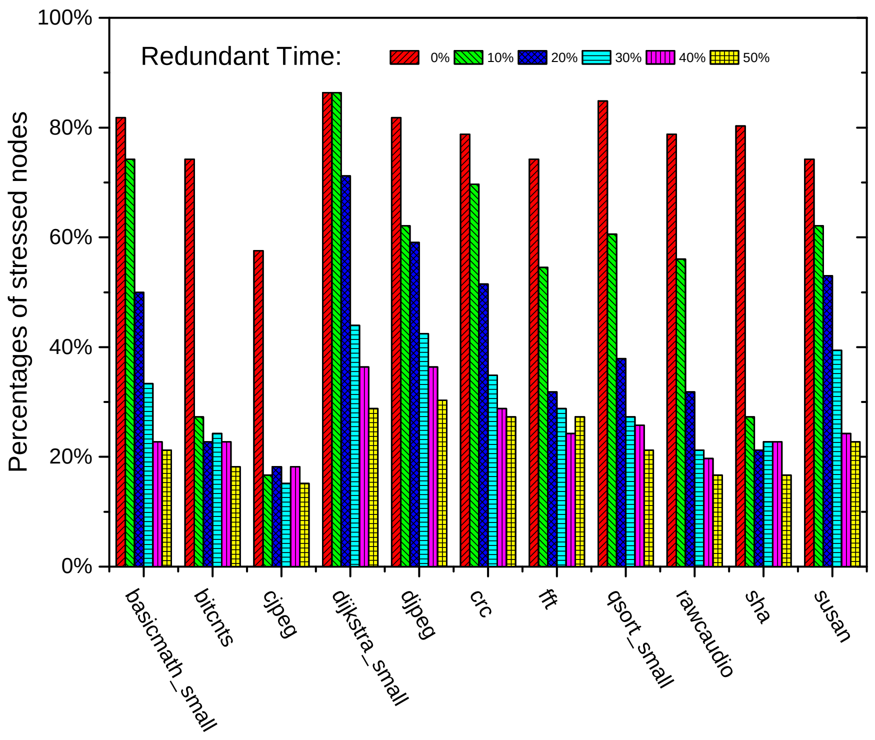


Figure 4.15: The effect of running BTI balancing program along with a different program from MiBench benchmarks on the stressed node percentage on the first critical path of the OpenRISC processor with different run times of the balancing program

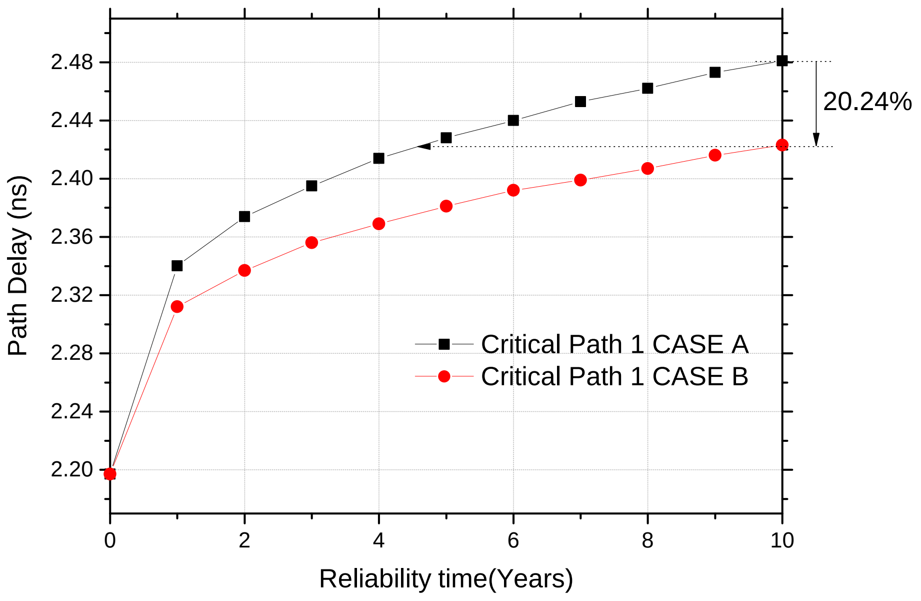


Figure 4.16: The advantage of running the anti-ageing program (CASE B) on the path delay of OpenRISC processor.

4.4.1 Discussion

In our analysis, we expect, for example, 11% degradation in six years as can be seen in Figure 4.16, so a simple solution could be guardbanding. Guardbanding is inevitable, not only for ageing but also for PVT variations. However, adding more guardbanding would negate the advantage of using a smaller technology size. So we have to find an active protective approach that will also estimate, sense and react to degradations.

The idea of this work in this chapter is to utilise the short idle periods in a processor [83] to reverse the BTI stress rather than running empty loops. In our case study of the OpenRISC processor, the critical paths are in the adder and we can propagate patterns simply by loading the patterns into a register and executing an addition operation. This program replaces the idle task and is executed whenever the operating system tries to schedule the idle task. In general, if the timing critical component is not the adder, then we have to replace the operation accordingly. If the critical paths are not controllable at the instruction level (e.g., in a control unit that may have many flip-flops), then we need an architectural solution rather than a software solution to propagate the patterns.

We also need to consider how process variations could affect the critical path ranking. If we get this wrong, we might heal a non-critical path and leave the real critical path unaffected. For this reason, we have to consider not only the critical path but also the nearly-critical paths that could become critical with PVT and time-dependent variations. In our case study, we have predefined $(\theta + \Delta\delta_{pv})$ to be 20% of the maximum path delay at time zero (δ_0), as described in section 4.2.1, which covers the first 100 critical paths. However, we found that the first 100 critical paths share more than 92% of the cells with the most critical path. Thus, balancing the most critical path also balances the nearly-critical paths. However, if the nearly-critical paths do not share a big percentage of their cells with the first path, then we have to consider every single path in our analysis and generate patterns for them to balance signal probabilities in parallel. So, even with process variations, this technique would target the nearly-critical paths. If the nearly-critical paths do not share cells with the most critical path, it is important to define a threshold that considers the process and ageing variation contributions and to control these paths in parallel.

The second phase of the proposed technique could be implemented during the normal operations of other components of the processor. This can be done at an architectural level by utilising the scan chain(s) but with a modification to control the balance chain (part of the scan chain) by inserting only one multiplexer and one OR gate. The multiplexer is used to separate the balance chain from the scan chain. As shown in Figure 4.17, two additional internal inputs fed by the control unit of the processor need to be added to the circuit (anti_aging_be), for enabling the balancing mode, and the anti-ageing patterns to be fed via the anti_aging_bi signal. The purpose of this separation is to shorten the path for controlling the primary inputs of the targeted module and applying the anti-ageing patterns in parallel with the normal operations of the processor when the circuit is not implementing any process using the targeted module.

The advantage of a hardware solution is to apply the anti-ageing patterns in parallel with the normal operations. For example, if the critical paths are in the multiplier and based on the multiplication instruction rate in the overall processor instruction rate, the balance state needs to be applied during the implementation of the remaining operations (e.g., addition division, memory/register read or write, etc.). To execute the balancing mode in parallel with the normal mode, a stand-alone architecture for the scan-chain that is dedicated to the balancing mode needs to be designed properly without conflict with the test or normal mode of the operation. The scan-in input needs to be fed with the anti-ageing patterns. Figure 4.17(b) shows the required modifications to the scan-chain circuit, which apply the balance-state patterns to component 3 (C3) during the normal operation of the circuit. However, additional signals are not additional pins to the integrated circuit as is the case for scan-in and scan-enable that are used during the test phase; instead, they are connected to the controller of the processor to apply the anti-ageing patterns during the normal operation of the circuit.

We demonstrate the correct functionality of the modified scan-chain with the gate-level circuit by applying NBTI/PBTI anti-ageing patterns. For simplicity, we designed three arithmetic components: 16 bit-adder, 16-bit-multiplier and 16 bit-divider with shared inputs and a multiplexed output with a 2-bit operation signal. The circuit has been functionally simulated then synthesised with Synopsys Design Compiler using the 90nm standard cell library from Synopsys. At that level, the OR gate and the multiplexer have been added to modify the scan chain for the purpose of using it to propagate the NBTI/PBTI anti-ageing patterns. Post-synthesis simulation is done to verify the functionality of the circuit after inserting the NBTI/PBTI balancing circuitry to the scan chain. Figure 4.18 shows the internal multiplication inputs in1_mult and in2_mult being fed with patterns serially though the anti-ageing balancing input (anti_ageing_bi), and in this example, are supplied with 1s instead of the balance-patterns that will be available after analysing the circuit along with the whole processor with real programs and benchmarks. The anti-aging balancing enable input (anti_aging_be) is used to activate the scan-chain by applying anti-ageing patterns during the normal operation of the circuit (during the addition operation in this example). Although the added circuitry increases the timing overhead of the scan chain, it will not affect the critical timing constraints that exist in the multiplier component.

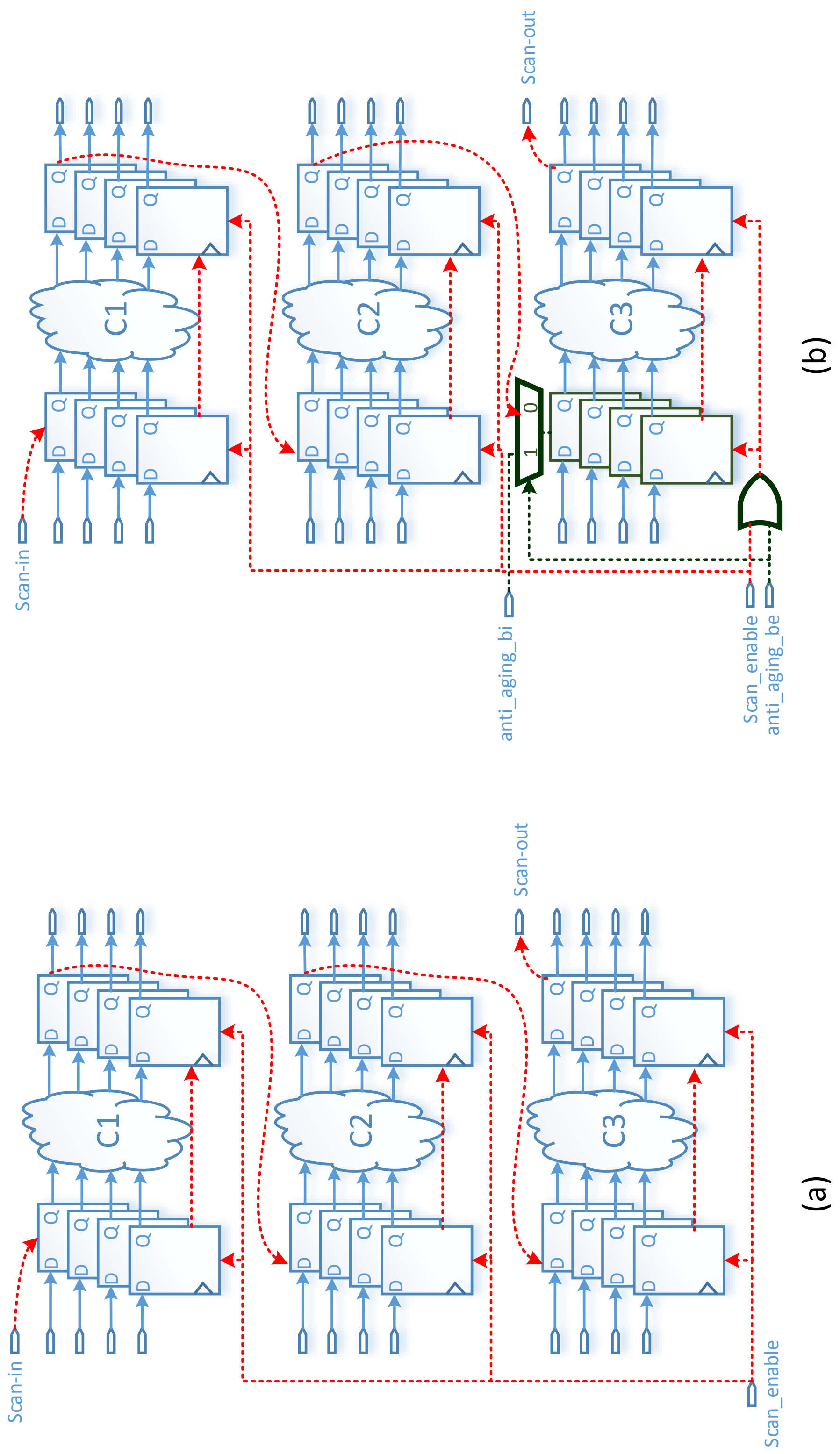


Figure 4.17: Modifying scan-chain circuit to support NBTI/PBTI balance state applying. (a) An example of scan-chain connecting three components. (b) Hardware-utilization approach to insert the anti-ageing patterns serially to the third component by modifying scan-chain.

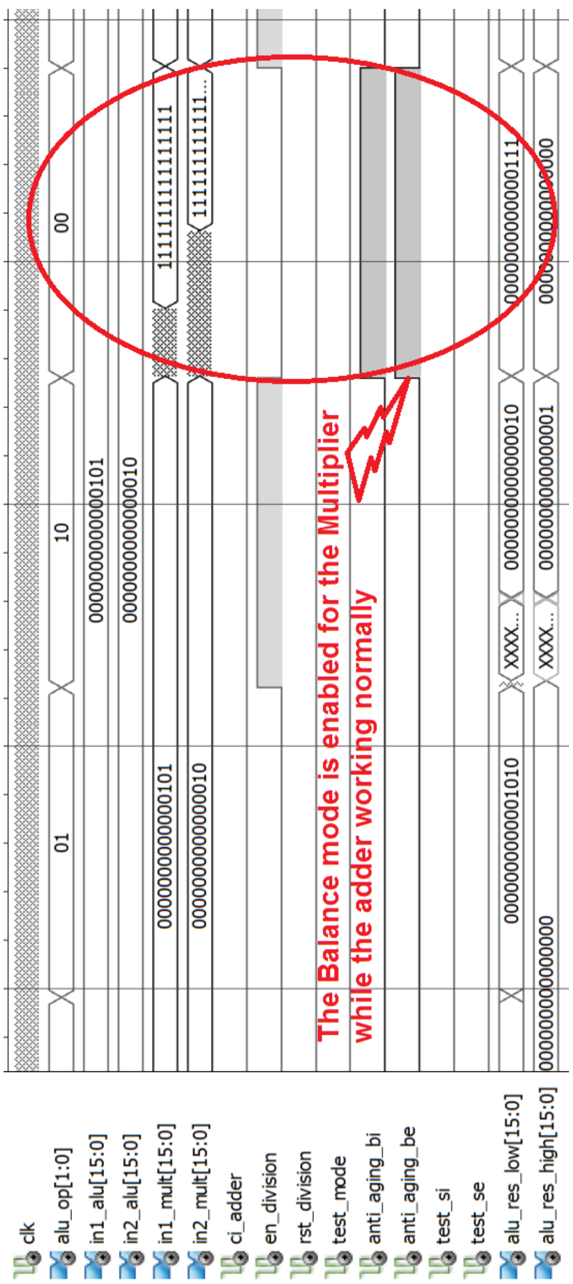


Figure 4.18: Signal waveform of the three arithmetic operations circuit alu_op = 00(add); =01(multiplication) and = 10(division) with modified scan chain for NBTI/PBTI balancing circuitry.

4.5 Conclusion

In this chapter, an application-specific high-level ageing analysis has been carried out to find a technique for CMOS ageing mitigation. The stress probability has been found at the application level down to the gate level. A cross-layer mitigation technique was proposed to apply anti-ageing patterns to the critical paths of a functional unit of a processor during idle times. This chapter presented technique using the processor BTI-balancing programs (Anti-ageing) to mitigate the BTI ageing effects. This technique generates anti-ageing patterns and apply these patterns by executing a program to balance the stress on the critical paths of the combinational part of the processor to alleviate BTI effects instead of running an empty process idle task. In the next chapter, we will deal with the BTI issue on the memory units of the processor including the flip-flops and SRAM-like circuits.

Chapter 5

An Application-Level BTI Ageing Mitigation Technique for Flip Flops and Register Files

In the previous chapter, BTI mitigation technique has been presented targeting the combinational part of the microprocessor. In this chapter, analyses and mitigation techniques for the memory unit of the processor are presented. The reliability of on-chip memory units, including flip-flops cells and register files, decreases with shrinking technologies due to CMOS ageing effects. As ageing negatively affects not only circuit performance but also the reliability of the circuit. The effect of BTI on the SRAM circuit has already been studied in the literature [90, 91, 92]. considering the stress as balanced in the SRAM circuit cells. However, the original of stress at the device level comes from the high-level control and data instructions. In our case study using an ARMv7 processor, the results show that the register file bits (SRAM cells) are stressed in a consistent way in which it is predictable to identify which bits tend to be more BTI stressed than others. To mitigate this behaviour, we propose a non-intrusive and technology-independent technique by executing a so-called anti-ageing program to balance the stress among register file bits. Although this technique does not control all the stressed bits; however, the results show that it is possible to put the stressed bits from a severe state to a balanced state, which reduce the ageing stress of the register file by 65%.

The objective of this chapter is to put the memory units of the processor into balanced BTI stress state using the same technique presented in Chapter 4 but for the flip-flops and register file of the processor.

The organisation of the chapter is as follow: In Section 5.1, we will introduce the issue of ageing on the memory units. Then, the trend of degradation on the flip-flops is presented in Section 5.2. In Section 5.3, the ageing trend and mitigation technique for the SRAM-like circuits of the processor will be presented. Finally in Section 5.4, the chapter will concluded.

5.1 Introduction

CMOS ageing affects the performance and reliability of digital systems due to degradation and stress. Memory units including flip-flops and SRAM circuits suffer from ageing effects like other combinational circuits; however, they have internal nodes whose states depend on previously stored states. Memory units could be timing critical (e.g., flip-flops) or data critical (e.g register file). Therefore, the consequences of ageing behaviour on these memory unit does not only affect the whole system performance but also the system reliability. In this chapter, the effect of ageing stress (specifically BTI stress) on these memory units will be studied for lowering ageing stress from the higher level of abstraction of the processor.

The flip-flops (FFs) that are usually used in the control unit of the processor are the most instantiated components in ASIC devices [93]. In this chapter, we will study the effect of high-level data on the timing degradation in order to identify the stress state. In the literature of flip-flops ageing mitigation, the proposals mostly suggest choosing the best device-level topology for slower degradation [93, 94, 95, 96, 97].

The register file, implemented using SRAMs, is considered to be the most data critical unit in the processor and in used by register, memory and manipulating instructions. In the literature [90, 91, 92], the register file utilisation has been balanced for reducing stress. The main problem of this approach is that the solution is not very practical, as it assumes that the register file is completely accessible or the whole registers are under stress, while as we will show in the later sections of this chapter that data tends to be distributed in predictable pattern among the bits inside the register files. Thus, running a non-intrusive data-dependent anti-ageing program could reduce the stress from these bits based on data analysis.

5.2 Flip-Flop Degradation Trend

Flip-flops or any clocked memory elements are considered to be one of the most critical components of the processor. In the past, several designs have been developed to optimise area, power and reliability of these components. Time-dependent variations could change the characteristics of flip-flops, and many design-stage processes based on these characteristics make the system less reliable and the timing constraints could be violated over time. These timing characteristics includes:

- The setup time (τ_s) is the minimum time required for the input data to be stable before the active edge (positive or negative) of the clock to generate a valid latching.
- The hold time (τ_h) is the minimum time required for the input data to be held stable after the active clock edge (positive or negative) of the clock to generate a valid latching.
- The clock-to-q (T_{ck-q}) delay refers to the propagation delay from the 50% transition of the active clock edge (positive or negative) to the 50% transition of the output data, q, of the latch/register.
- The setup skew (τ_{sw}) refers to the delay from the latest 50% transition edge of the input data signal to the 50% active clock transition edge (see Figure 5.1).
- The hold skew (τ_{hs}) refers to the delay from the 50% active clock transition edge to the earliest 50% transition of the input data signal (see Figure 5.1).

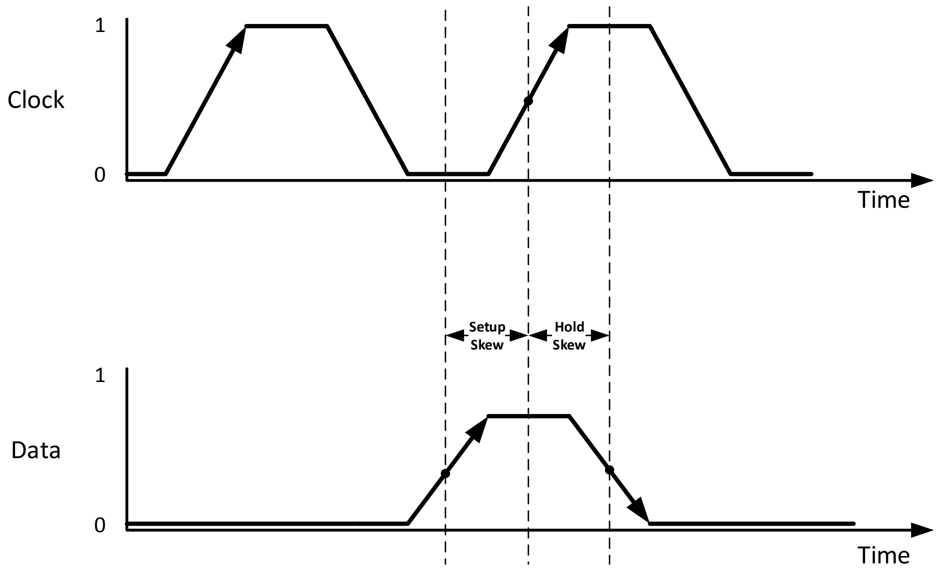


Figure 5.1: Setup and hold skews shown in the clock and data waveform.

In [93], the flip-flop degradation has been analysed considering the three ageing mechanisms: bias temperature instability (BTI), hot carrier injection (HCI) and time dependent dielectric breakdown (TDDB). Their finding focused on comparing different flop-flop architectures and their susceptibility to ageing effects. This would then be used to select the most appropriate structure of a flop-flop for a specific application. Regardless of the number of flip-flops per ASIC, one flop-flop degrading beyond the threshold timing, the entire system timing may be affected. NBTI-induced degradations change the flip-flops setup and hold time, and then this will tighten the setup/hold timing constraints in the design. Current techniques for mitigating the ageing effects on the flip-flops include sizing the transistors [98].

The setup and hold times depend strongly on each other. Typically, the setup time decreases as the hold skew increases and vice versa. Similarly, the hold time decreases as the setup skew increases and vice versa. The trade-off between setup and hold skews and the hold and setup times is an important consideration in flip-flop design. With ageing degradation, the setup and hold time may change with time and may lead to timing violations that will result in system errors. Using the HSPICE bisection tool to measure the setup time (see Figure 5.3) which the signal names are as these shown in the schematic representation of the D type flip flop used in 90nm Synopsys technology shown in Figure 5.4. Thus, if the degradation accumulated from the combinational part with the propagation delay of the flip-flops is exceeding the given slack, we will have setup time violation (see the too late input signal on input “data”, Figure 5.3). Using MOSRA Level 3 [60] to evaluate both BTI and HCI effect of the flip flop propagation delay and setup time.

Bisection is an optimisation method that uses a binary search method, to find the value of an input variable (target value). This variable is associated with a goal value of an output variable. Bisection could be used as an iterative analysis tool to define violation specification of the flip-flops by characterising the cell and finding early, optimal and late setup (or hold) times of the flip-flops. In Synopsys HSPICE, this can be achieved using the Bisection tool. Figure 5.2 illustrates how bisection method works by fixing one signal at T_2 (clock signal) and varying the second signal at T_1 (data signal) until it generates the setup time that represents the latest time that a data change could generate a valid output. Otherwise, it is too late and may generate an invalid output from the previous state of the input signal.

Firstly, the BTI effect has been analysed considering its effect on the flip-flop propagation time that represents the time degradation of the transition of the signal from clock active edge to the output Q or QN of the flip flop. This usually represents the starting or the ending of the critical path delay and represents a subset of the total delay of the combinational logic circuit. Using MOSRA, the flip-flop has been stressed using three states as shown in Figure 5.5 and measured two transition for each state considering the rising and falling of the Q signal (see Figure 5.4). It is clear that when we have unbalanced signal probability that tends to be always one (e.g., SP(0,D), the signal probability of the D input to be zero = 1%[1]) or tend to be zero must of the time (e.g., SP(0,D), the signal probability of the D input to be zero = 99%), the degradation would be the greatest on one of the falling or rising propagation delays. Meanwhile, when we have balanced signal probability on the D signal (e.g., SP(0,D) = 50%), the

[1]Not 0% because at least there is one transition to be not a constant value. also we need at least one transition to measure the propagation delays; the same with 99%

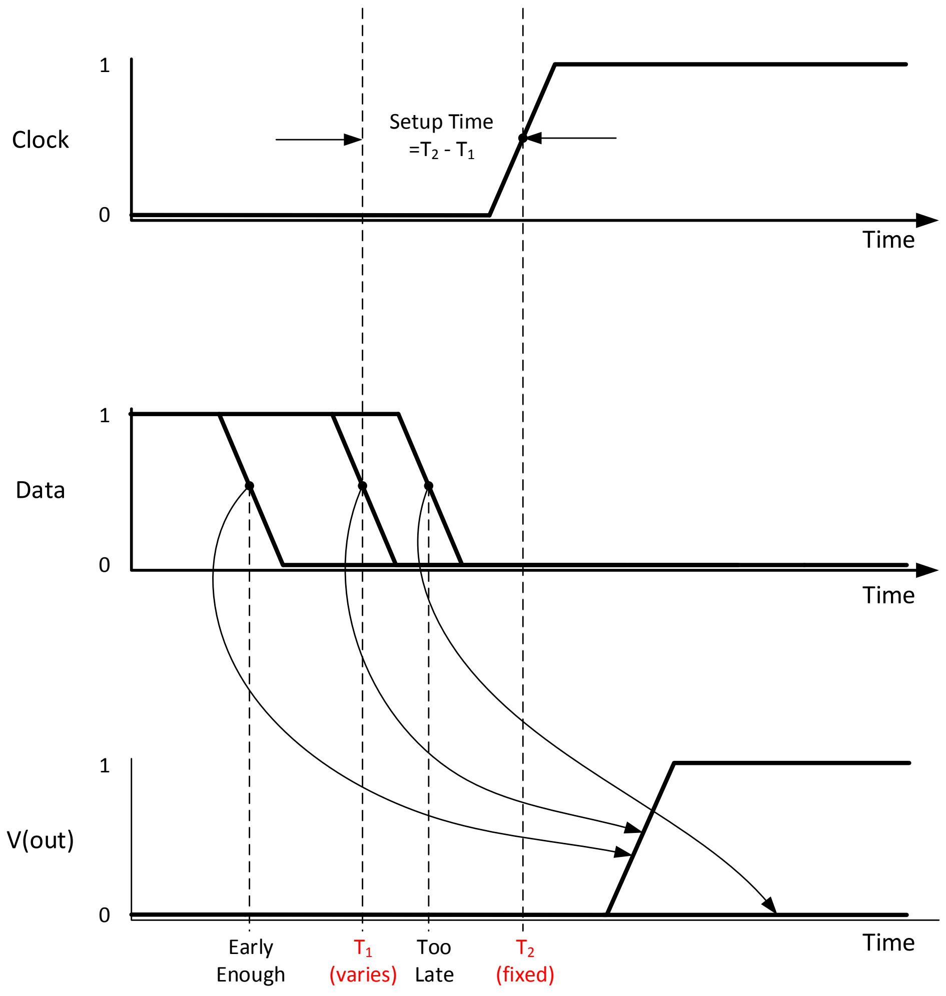


Figure 5.2: Determining setup time with bisection violation analysis from Synopsys HSPICE [99].

propagation delay for the rising and falling delays would increase equally (for a 10-year simulation period, this is about a 6% increase in the propagation time of the flip-flop).

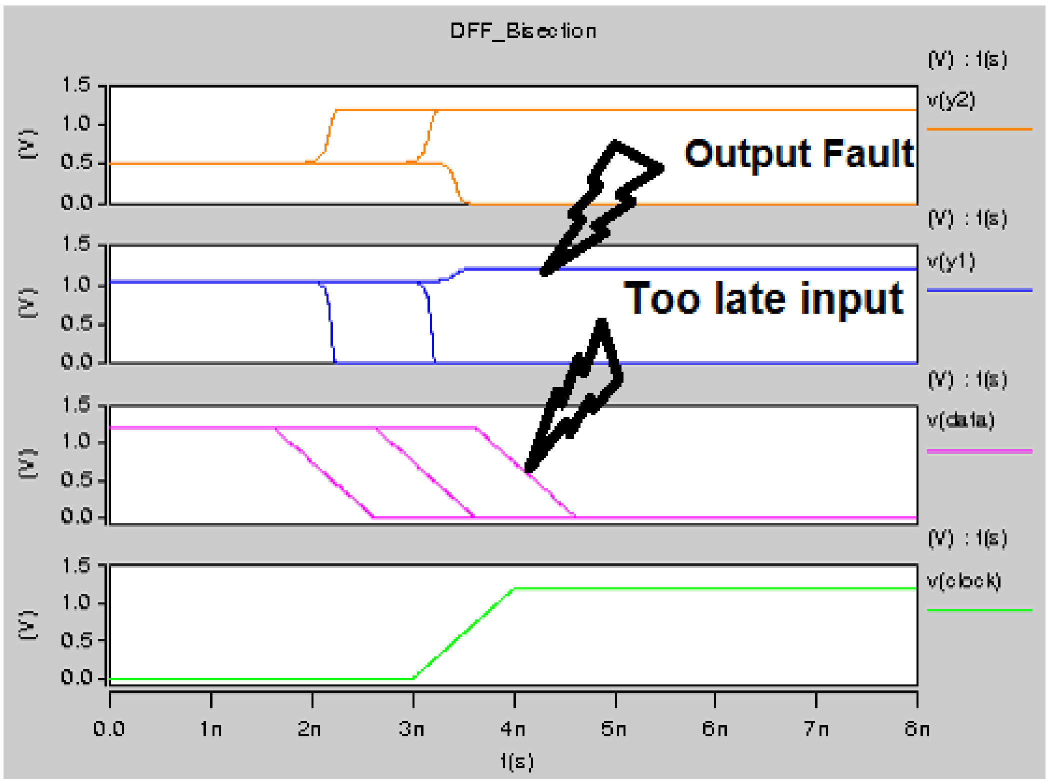


Figure 5.3: Violation of the setup time.

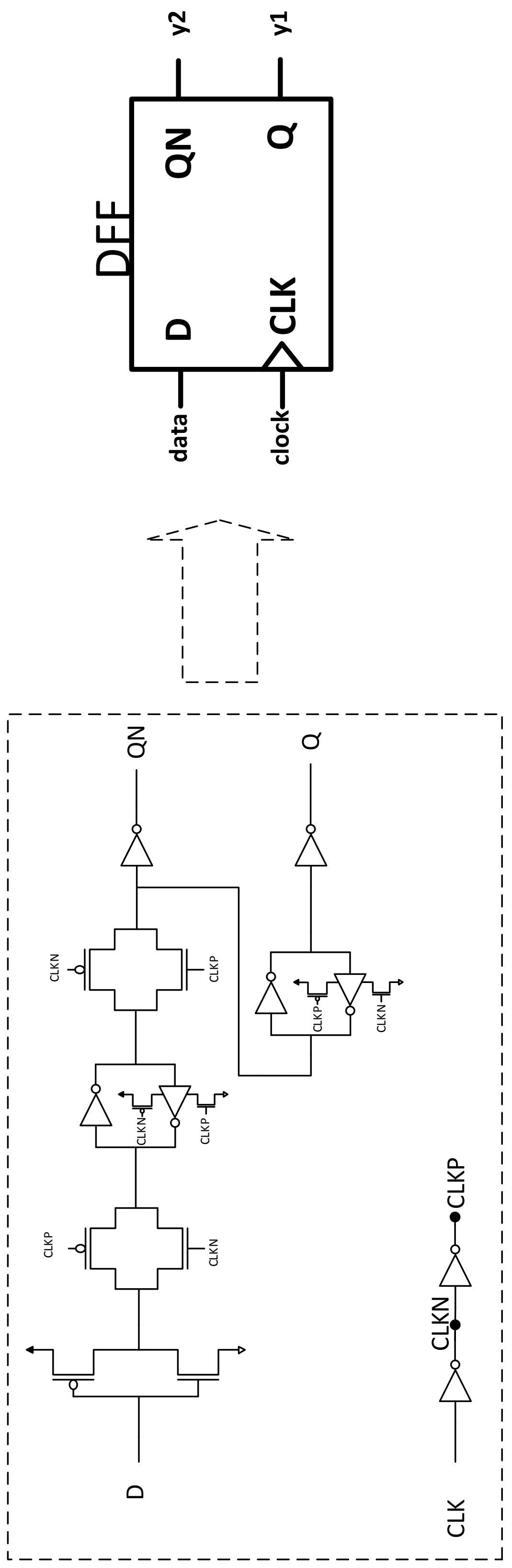


Figure 5.4: Schematic representation of D type flip-flop.

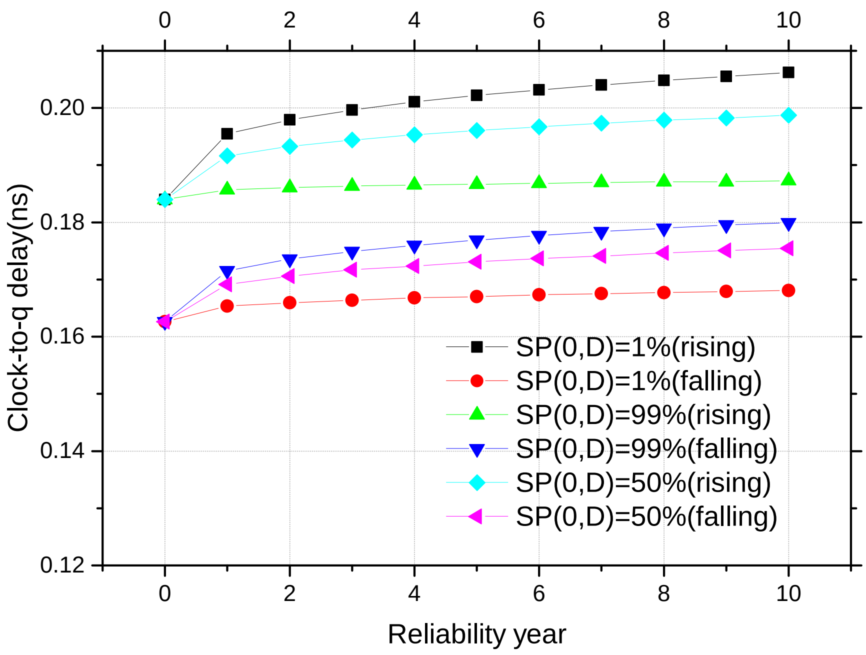


Figure 5.5: Propagation delay degradations versus different signal probabilities on the input data.

Secondly, the flip-flop setup time has been analysed to find the effect of BTI degradation considering balanced signal probabilities on the D input signal and the two extreme states of 99% and 1% of the signal probabilities. We used MOSRA Level 3 for circuit degradation and bisection HSPICE method for measuring the required setup time with the change of CMOS threshold voltage due to ageing. Although, the percentage increase in the setup time reached 25% of its initial setup time when the input D's signal probability is mostly 99%. However, the percentage of setup time degradation with the critical path degradation is negligible as adding additional slack could mitigate the problem of the required setup time increasing due to ageing. Moreover, if the balancing solution adopted at the combinational critical path also considers the D signal of the flip flop then the circuit would suffer less from the setup time increase, as shown in Figure 5.6.

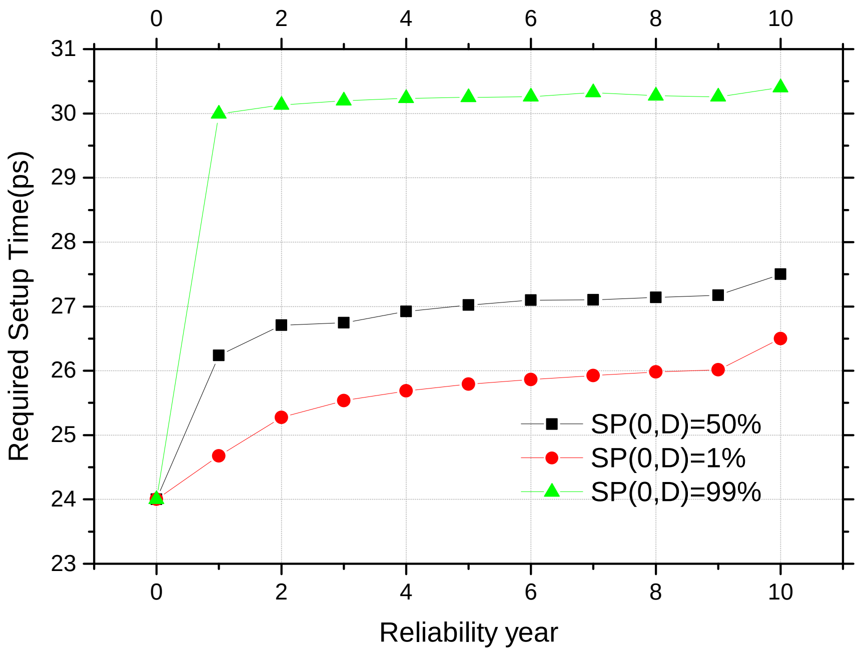


Figure 5.6: Increasing the required setup time by time versus different signal probabilities on the input data

Thirdly, the number of transistors on the flip-flop shown in the Figure 5.4 is 26, of which 12 are biased to the clock signal CLK (CLKN or CLKP) as this signal is an inherently balanced signal with 50% signal probability. Thus, considering HCI effects along with the BTI effects is crucial for identifying the prominent effect of ageing in the flip-flop circuitry. Figure 5.7 shows that the falling propagation delay from the clock to the Q signal where the degradation reaches 20% from the initial delay is the worst case when the input D's signal probability of being zero is 99%. In the same trend for the rising propagation delay as shown in Figure 5.8 with the worst case when the input D's signal probability of being zero is 1%. However, balancing signal probability for the input D of the flop-flop generates balanced degradation between the two worst cases for the falling and rising propagation delays.

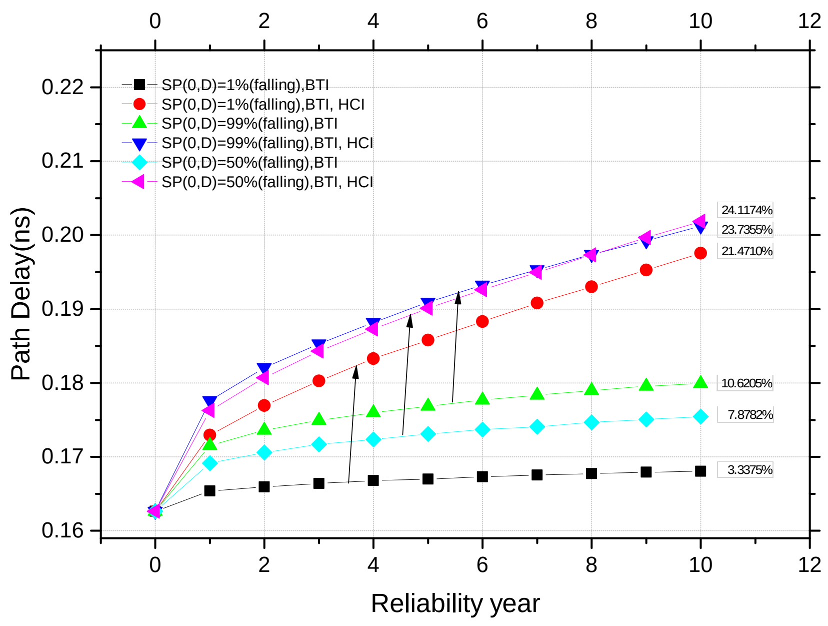


Figure 5.7: The effect of BTI and HCI on the falling delay of the flip flops

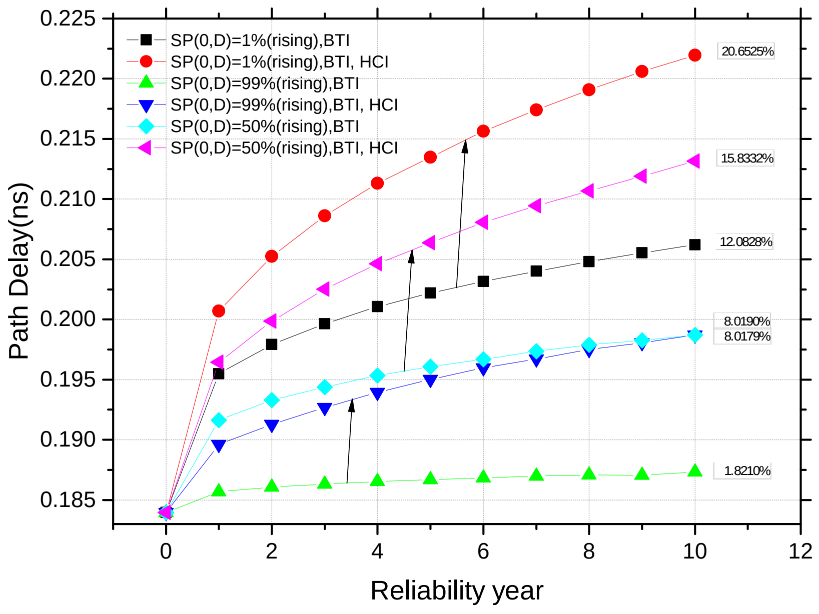


Figure 5.8: The effect of BTI and HCI on the rising delay of the flip flops

5.3 Ageing in SRAM cells

The symmetric nature of the typical 6-transistors SRAM circuit shown in Figure 5.9 makes part of it to be always under stress. Considering the most prominent cause of ageing which is NBTI, the device could be under stress when it is negatively biased and has a high temperature. For example, when a cell stores a '1' state, node V_R should retain a high voltage higher than its threshold voltage to represent '1' and that makes M_2 and M_4 positively and negatively biased, respectively. This would make V_L to retain the complement state of V_R to have '0' state. Thus, any device-level degradation would manifest itself as read sensitivity in the the memory cell. However, read or write latency are not considered to be the most harmful effect of ageing on the SRAM cell [100, 101]. The main parameter that is affected by ageing is the static noise margin (SNM) [100, 101, 102] .

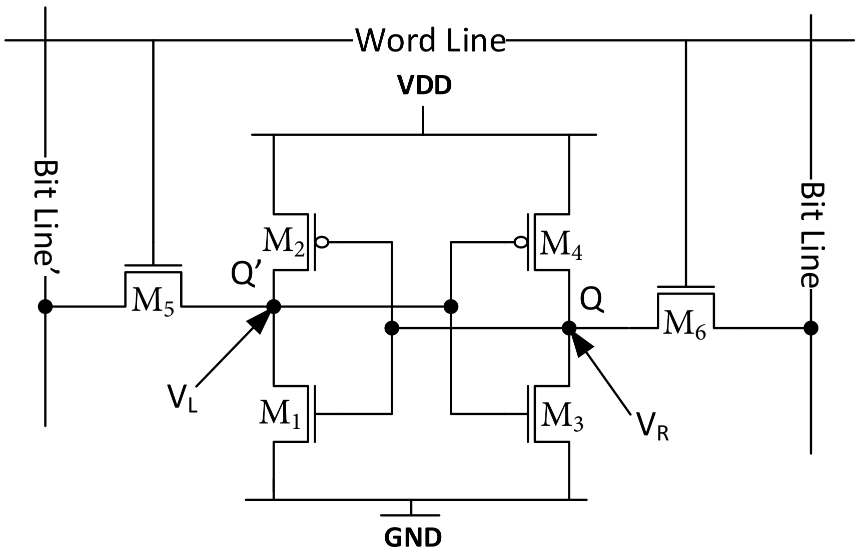


Figure 5.9: 6-Transistors SRAM cell

SNM is defined as the minimum voltage required to change the state of the SRAM cell [103]. The graphical representation of the SNM is shown in Figure 5.10 and is the maximum square that fits between the inverters characteristics in the SRAM cell. In [100], from the basic transistor level of the SRAM circuitry, the threshold voltage has been measured for the 100nm and 70nm technology sizes. The results in Kumar et al. [100] show that both read and write delay are not reduced over time which means that NBTI degradation would not negatively affect the speed of reading/writing from/into the SRAM cell. However, the minimum voltage for state inversion is reduced, which puts the reading stability at risk, and which can potentially produce failure at the higher level of abstraction. Due to the symmetric structure of the SRAM circuit, there is at least one inverter under stress with negative BTI and the other one under positive

BTI. Thus, the best zero signal probability is 50%, which ensures that both sides of the SRAM circuit are balanced [102].

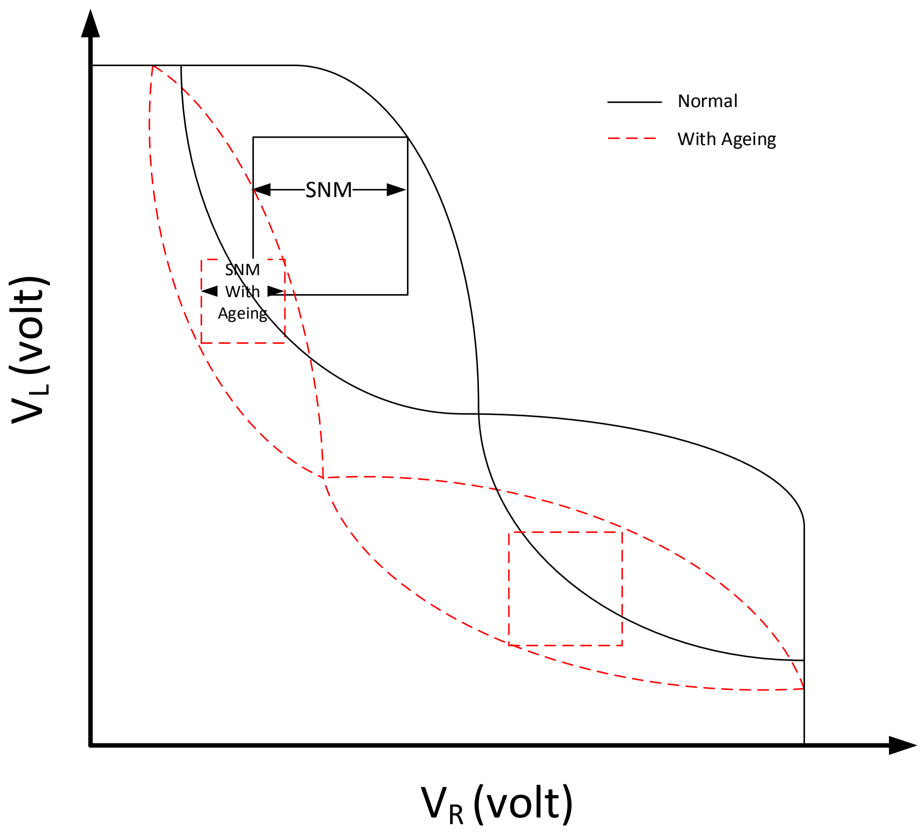


Figure 5.10: Static Noise Margin

5.3.1 Register File BTI Mitigation Techniques

In this section, existing ageing mitigation techniques for register files will be presented. In [100], periodic bit inversion has been proposed using hardware and software approaches for balancing the signal probability inside an SRAM array to have zero signal probability equal to 50%. For the hardware approach, additional physical circuitry and control signals are used. However, this additional hardware could be also under stress and affected by ageing. For the software solution, they proposed that on specific days to have always inverted instructions before storing and after reading any data. However, exhaustive usage of inverter instructions would increase the processing time and will defeat the purpose of shrinking technology to have faster devices.

In [102], the register file or 6-transistor SRAM circuit has been studied for mitigating the SNM degradation and the work have focused on the register file component as it has no specific patterns of stress in the SRAM array (bit level or register level). So, register and bit level rotation has been proposed to avoid keeping one register or one bit within that register to be stressed by a dominant stored state ('0' or '1'). However, the main drawback of this technique is to consider that all the registers within the register file have the same operating system privilege and usage, while in practice, there are registers dedicated for the operating system kernel that have values cannot be updated or moved. Otherwise, they could could activate/deactivate interrupt bits and lose the current context of the program.

Another work, Abella et al. [89], that dealt with a generic strategy for processor NBTI mitigation in the register file, proposes to have shadow registers that carry the inverted value of the original register file and update the original register with the values from the shadow register for 50% of the operation time to obtain 50% zero signal probability. Although this technique ensures that the stress among the cells of the register file is balanced; however, the shadow register file would be under stress as well. To avoid stressing the shadow register file, the original values need to be swapped with these shadow values. However to implement the swapping operation, a temporary physical unit is required and this could tripling the hardware overhead.

5.3.2 Case Study: ARMv7 Register File

We use the GEM5 simulator to simulate the data distribution on the register file of an ARMv7 by running a number of the MiBench benchmarks from different applications. The benchmarks are "hello world", "basic math", "jpeg", "typeset", "dijkstra", "patricia", "qsot", and "fft" that represent applications from image processing, signal processing and mathematical operations. For each benchmark, the bit probability of each core register in the simulated processor has been collected, and the probability of the bit of being zero has been computed, as illustrated in Figure 5.12. Although most of the high significant bits of most of the registers in the register files tend to be a '0' state; however, running one application is not enough to conclude that it is true for most of the time. Therefore, we run a suite of applications to obtain the average distribution of data and find the area that are common to be either one or zero.

From the data collected, an area or bit state classifier has been used to find the tendency of each bit in the register file. One thing we need to consider is that these registers are not always controllable by non-privileged user programs. Therefore, the first step is to identify the stressed bits (i.e., the bits that tend to be always '0' or always '1' or in our case, we have considered that having probability of a bit being zero either less than 30% or greater than 70% as stressed and any thing else as not stressed because the bits not tending to the extreme stress). Figure 5.13 shows the probability of a bit being zero for a few benchmarks from MiBench and their average distributions. Furthermore, the percentages of stressed bits in the ARMv7 register file are more shown in Table 5.1. After classifying the register bits between these that needs to be reversed or not, an anti-ageing program needs to be run to relax the stressed bit. For this purpose, and to prove the concept, a reverse stress (anti-ageing program that writes complement values to that found in the analyses phase on the register files) is needed.

Figure 5.14 shows how interleaving the "Hello World" (stressed program) with the reverse_stress (anti-ageing program, that propagate the reverse pattern to the general propose registers R1 to R12, See Figure 5.11) can relax up to 67% of the registers in the register file. For optimisation purposes, the instructions of the anti-ageing program could be interleaved with the stressed program at the time that these registers to be controlled are not in use but to be considered from data hazards that could be happen while trying to reverse the stress.

```
        # Stimulate the reverse patterns to the register R0
        MOV r0 ,#0x0000
        MOVT r0 ,#0xFFF0
        MOV r0 ,#0xFFFF
        MOVT r0 ,#0xFFFF
        # here we have propagated the reverse patterns to R0
        # Stimulate the reverse patterns to the register R1
        MOV r1 ,#0x0000
        MOVT r1 ,#0xFFF0
        MOV r1 ,#0xFFFF
        MOVT r1 ,#0xFFFF
         .
         .
        # Stimulate the reverse patterns to the register R12
        MOV r12 ,#0x0000
        MOVT r12 ,#0xFFF0
        MOV r12 ,#0xFFFF
        MOVT r12 ,#0xFFFF
```

Figure 5.11: Anti-ageing program for balancing BTI stress on ARMv7 register file.

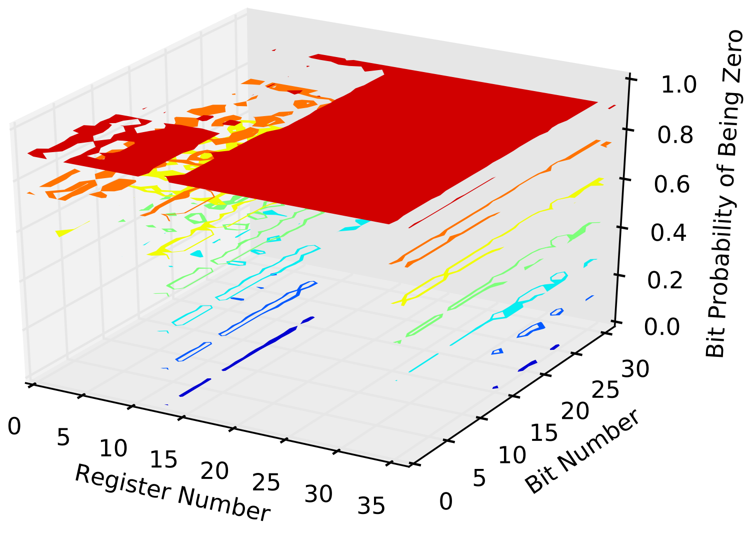


Figure 5.12: SP stress on the register file of ARM7 when running Basicmath benchmark

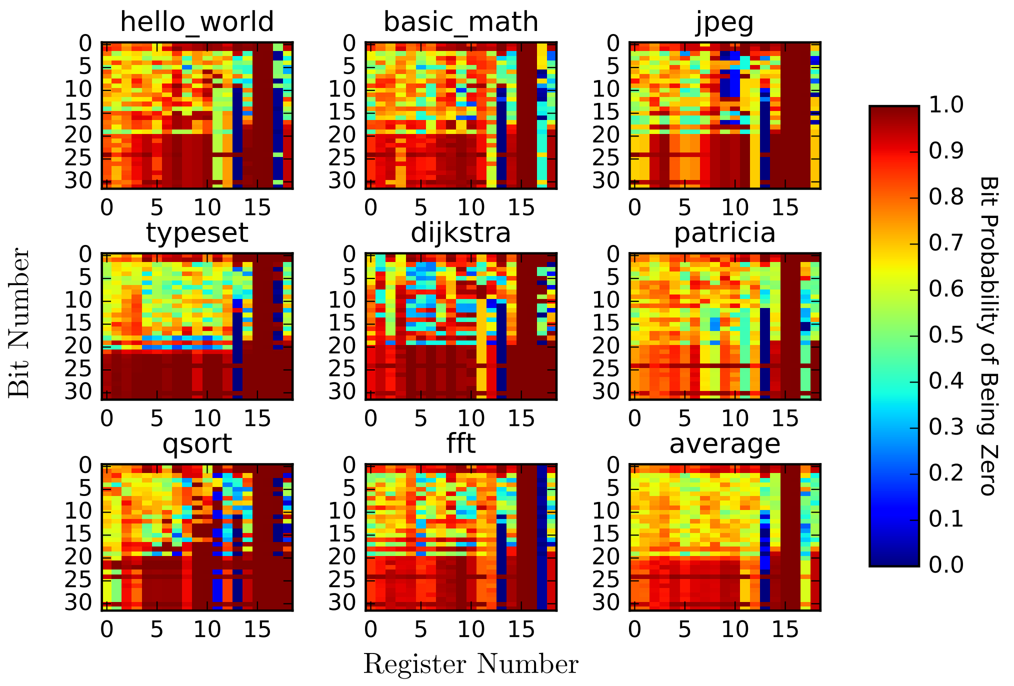


Figure 5.13: Register file stress distribution with different benchmarks running

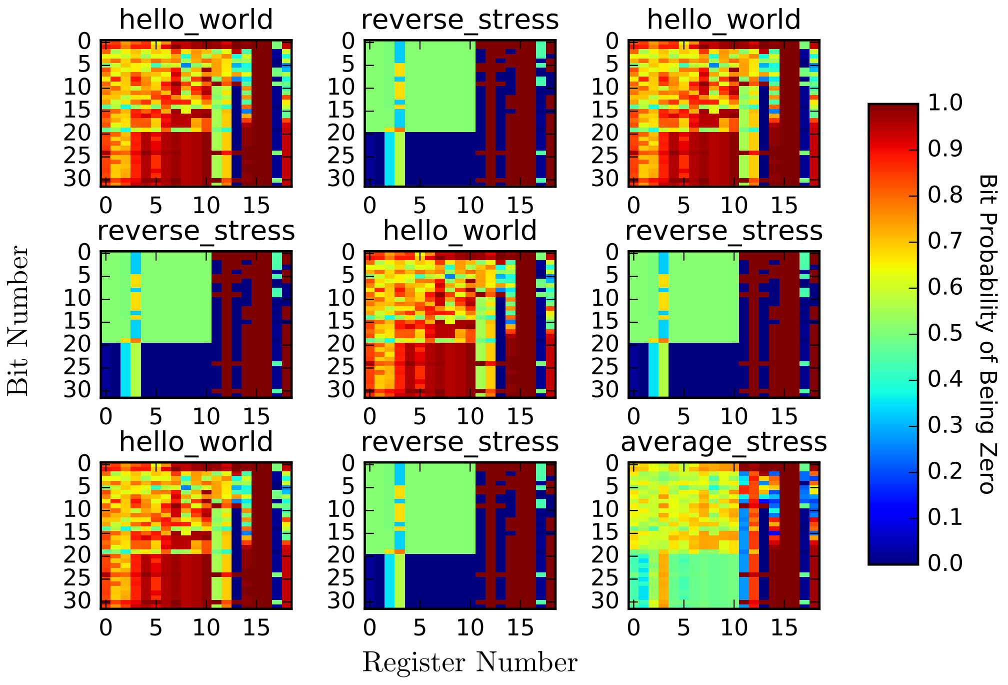


Figure 5.14: Balancing signal probability of the register file. Some of the general purpose registers are not propgated with the anti-ageing patterns because they are used during the task initialisation by the system.

Table 5.1: Percentages of stressed bit in the ARMv7 register file when executing different benchmarks.

Benchmark	hello_world	basic_math	jpeg	typeset	dijkstra	patricia	qsort	fft	**the average**
Stressed bits	66.78%	65.63%	61.18%	61.02%	71.88%	52.63%	71.38%	71.05%	65.19%

5.4 Conclusion

This chapter has focused on the problem of ageing on flip-flops and register files. For flip-flops, the BTI and HCI effects have been analysed, and we find that balancing the signal probability of the input signals reduces the BTI stress, which will consequently reduce the setup and propagation time degradations. However, the HCI is more prominent due to clock signal transition activity. The anti-ageing solution presented in the previous chapter could mitigate the BTI effect on the flip-flops, but to mitigate the HCI, already-existing dynamic power optimimisation techniques (e.g., clock gating) are needed. For the register file, the analysis of the workload effect on the degradation behaviours was conducted by implementing programs from the MiBench benchmarks. We find that SNM (read sensitivity) varies from one area (group of bits or registers) to another making particular registers or bits in that register to be in a more stressed state than others. Finally, a non-intrusive and technology-independent solution has been implemented for the register file by using a balancing stress program that could relax the stress on the register file by up to 67%.

Chapter 6

Learning-Based BTI Stress Estimation and Mitigation in Multi-core Processor Systems

In the previous chapters, the workload-based stress analyses, and associated ageing mitigation techniques, on the single core processor system have been presented. This chapter will use a multi-core processor system for mitigating the ageing effects by considering the power and temperatures of the cores. Previous work in the literature has mainly focused on modelling the ageing behaviour at the device level and developing ageing sensors for on-line delay detection at the system level. In this chapter, we will present a method to estimate the ageing stresses (e.g., temperature, ageing-stress activity etc.), rather than modelling ageing (i.e., performance degradation) itself. The essence of the proposed approach is to proactively estimate and minimise the ageing stress at the system level by adjusting frequencies and utilisations of cores. The model of estimating ageing stress is constructed using machine-learning algorithms based on artificial neural networks, relying on application-specific data extracted from high-level workloads (e.g., parsec and splash2 benchmarks). The problem of estimating stress from discrete data collected from real or simulated system is a regression problem that could be modelled using machine learning techniques that have minimum approximation error [104]. Our experimental analysis, performed on a four-core Xeon X5550 processor indicates that the proposed model is capable of identifying the cores that are most susceptible to ageing stress with less than 0.1% error and is able to proactively reduce the ageing stress by 50%.

6.1 Introduction

To achieve high performance and energy efficiency, multi-core architectures are used widely in modern computer systems. When multi-core architectures are powered by batteries, reducing the power consumption is crucial, since an energy-efficient design means slower depletion of batteries and results in lower chip temperatures that in turn improve performance and reliability.

With the increasing susceptibility of logic cells, memory and interconnections to time-dependent degradations, especially at smaller technologies, an adaptive proactive reliability management approach is needed. Multi-core processor systems, like other digital devices, are affected by ageing, more specifically by BTI, as the system performance degrades over time due to the change in the physical characteristics of the operational transistors. Ageing models are either hard to simulate for the whole system, or the model's inputs are hard to predict during the design stage of the digital system. Therefore, an adaptive proactive solution is vital to minimise the factors that could worsen the ageing effects, which is more useful than simply predicting the lifetime of the chip.

In this chapter, we will focus on two factors that lead to BTI stress: idleness and temperature. As BTI is partially recoverable [105], the target is to increase the period of recovery instead of keeping the state of the system unchanged (i.e., idle or statically negative/positive biased). If we do not consider the spatial and temporal effect of temperature, the high temperature does not occur in the idle state. In single-core processors, it is rare to have both idleness and high temperature occur at the same time. However, in multi-core processors, ageing is accelerated with increased temperature, especially during the idle state. This could happen in the case of increased temperature from a previous non-idle state, or when temperature is transferred from adjacent locations (see Figure 6.1). We will focus on the spatial effect of temperature and its effect on BTI-induced degradation in multi-core systems.

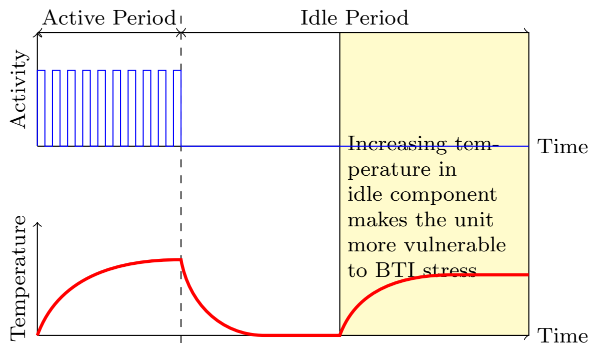

Active Period
Idle Period
Activity
Time
Increasing temperature in idle component makes the unit more vulnerable to BTI stress
Temperature
Time

Figure 6.1: The effect of temperature on an idle core

The main objective of this chapter is design a model to predict the ageing stress on the multi-core processor systems and use this model to mitigate the high-level stress by regulating the core frequencies based on proactive ageing stress prediction.

6.2 Multi-core Power Model

Saving power means reducing the temperature, which in turn reduces the ageing stress. Modelling power could be used for different reasons including estimating the system efficiency or to optimise the energy and reliability of the system. In general, the source of power consumption is classified either as dynamic power, which occurs as a result of transistor switching, or static power, which is dissipated even when the system is idle.

The first source of total power consumption is from the dynamic power consumption, which is defined at the transistor level as follows:

$$P_{\text{dynamic}} = \alpha \cdot C_{\text{L}} \cdot V_{\text{dd}}^2 \cdot F, \tag{6.1}$$

where α is the switching activity of the transistor, C_{L} is the total load capacitance, V_{dd} is the supply voltage, and F is the clock frequency.

The second source is the static power consumption, which is defined at the transistor level as follows:

$$P_{\text{static}} = V_{\text{dd}} \cdot I_{\text{static}} = V_{\text{dd}} \cdot (I_{\text{sub}} + I_{\text{gate}})), \tag{6.2}$$

where I_{static} is the current that leaks through the transistors during the idle state. The static current is negligible for technology sizes above 100 nanometres [106]; however, with the increasing market demands for higher chip densities, the static power consumption is now becoming significant. The main components of the static current are subthreshold drain current, which is the current that leaks between the source and drain of the transistor when the transistor is in the subthreshold or off region, and gate-oxide current, which is the current that leaks between the gate and oxide insulation [106]. With high-κ technology, gate capacitance is increased, which makes gate-oxide leakage current negligible [107].

However, for multi-core systems, the power cannot be simulated using the simplified models presented in (6.1) and (6.2) [108]. Instead, and with the help of performance counters, the instruction per cycle (IPC) per cores could represent the activity rate or the utilisation of the core that defined as the number of instructions executed and committed per clock cycle. Therefore, similar to [109] and [106], we test two models using non-linear and linear relationships between the core switching activity and its IPC for modelling dynamic power in multi-core systems and compare the results. The dynamic power can be estimated using a given IPC (which implicitly captures the switching activity and idleness) and clock frequency. To accurately compute the total power, we need to consider the static power as well, which can be computed for a given temperature, supply voltage and threshold voltage.

Power data could be collected from physical or simulated processor to extract the dataset for modelling the power using either non-linear regression (e.g., Neural Network) or linear regression machine learning methods depending on the output data is linear or non-linear dependent with the input data. The dataset that contains the input and the target power for the model is defined as follows:

$$Data : \{X_n, Y_n\}_{n=1}^{N}, \tag{6.3}$$

where X_n is the input data that includes: frequency (f), supply voltage (V_{dd}) and the IPC for each core, c. The input data can be represented mathematically as follows:

$$X_n : \{f, V_{dd}, IPC^c\}. \tag{6.4}$$

Y_n is the output data that represents the target power for each core, and is defined as follows:

$$Y_n : \{P_n^c\} \tag{6.5}$$

The input and output data has been collected from simulated 4 cores processor using simulation tool (Sniper Simulator [110] and McPAT [111], defined in Appendix E) that allows us to define the configurations as presented in Table 6.1. We considered to model the nominal, minimum and maximum corners for the Xeon x5550 Gainestown x86 microprocessor. Examples of the IPC and power consumption for the cores of the Xeon multi-core processor running Black-Scholes benchmark are shown in Figs. 6.2 and 6.3 respectively. In Appendix C, the complete input and output data collected from running various benchmarks from parsec and splash2 available in Sniper multi-core simulator.

Table 6.1: Xeon processor settings

Parameters	**Settings**
Frequency	2.66, 3.06, 1.7 GHz
Vdd	1.2, 0.85, 1.5 V
Technology_node	45nm
Area	42.5mm x 45mm
Number of cores	4

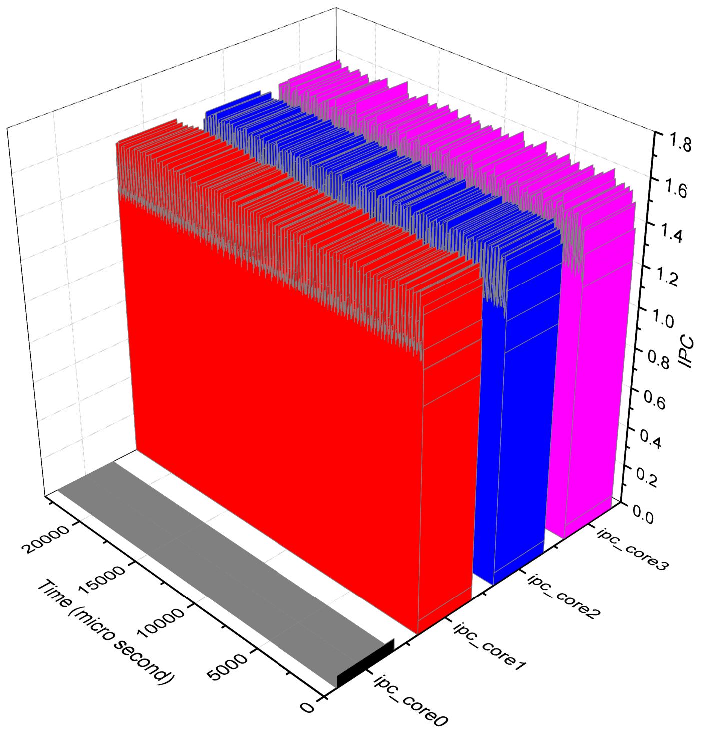


Figure 6.2: IPC traces extracted from running Black-Scholes benchmark on simulated Xeon multi-core processor.

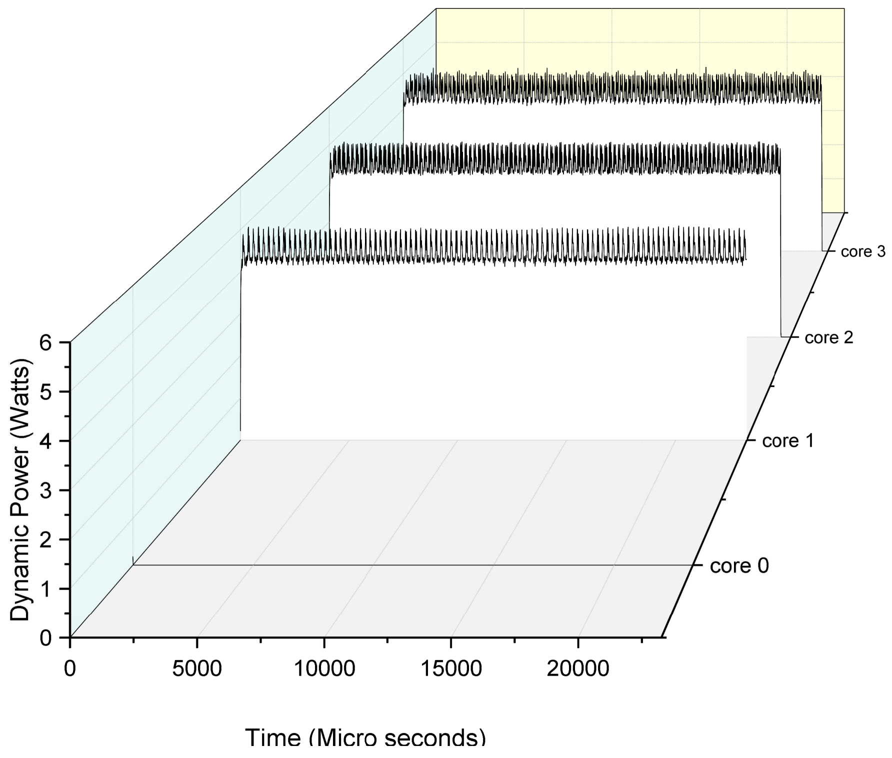


Figure 6.3: Power traces extracted from running Black-Scholes benchmark on simulated Xeon multi-core processor. Core0 is in an idle state while other cores are executing processes.

Two techniques are employed to model the power consumption. The data and the MATLAB code is provided in appendix C.

Firstly, we use a non-linear modelling method based on a neural network that are able to use priori information hidden in data. The process of extracting these hidden information is called "learning" [112]. Feed-forward neural network is used because relationship between the input and output data is forwarded from the input to the output data (no loop needed as a feedback from the output data). In our case, data extracted from executing parsec and splash2 benchmarks on a Xeon 4-core processor into the training phase of the machine learning. Using a neural network consisting of 20 hidden layers[1] and four output layers (one for each core) estimates the power with least mean squared error (MSE[2]) of 4.67% (see Figure 6.4).

Secondly, we model the power using linear regression to obtain the weights and bias of the model as follows:

$$P_{\text{dyn}} = V_{\text{dd}}^2 \cdot f \cdot (\omega \cdot \text{IPC} + b)\,, \tag{6.6}$$

where ω and b are the model parameters that represent the weight vector and bias of the linear regression parameters to be determined. The least mean squared validation error is 1.76% which is better than that found using feed-forward neural network. To measure the uncertainty of the estimation (i.e., the standard deviation or how a new unseen data fed into the network affects the estimation performance), we use a cross-validation method, which divides the training data into N folds and trains the system using $N-1$ folds. The remaining one fold is used for testing, and is unseen during training. For example, if we have data set of 8000 samples and N is 8, we will have eight folds each of 1000 samples. A cross-validation method allows us to traverse the whole data for training and testing while validating data that was not fed during training phase [113]. Figure 6.5 shows the mean squared error with eight different validation folds to have only one validation fold has mean squared error greater than the one obtain using neural network.

6.3 Multi-core Thermal Model

The thermal model extracted for our multi-core system is based on compact thermal modelling presented in [114, 115]. This method for modelling the temperature attempts to reduce the complexity of the lower level of abstraction and achieve a high level of accuracy. For temperature modelling of multi-core systems, we have used the same

[1]Neural Network (NN) hidden layers try to convert the non-linear relationship into linear relationship to the next layer; NN size defines the performance of the modelling but increasing the size into a limit which will see a slight performance improvement; in our case, any network size greater than 20 produced the same result as 20 network size.

[2]MSE is defined as the average of squared differences between the target (desired) vector and the estimated output vector from the model.

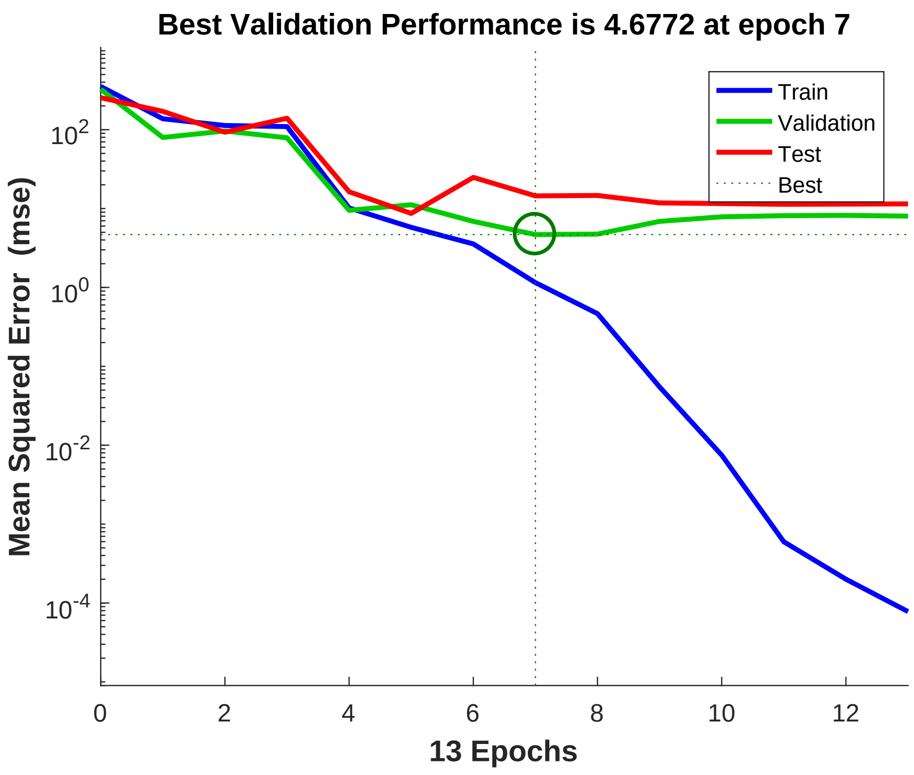


Figure 6.4: Prediction error using feed-forward neural network training; X axis (Epochs) represents the number of times that all training set are used once to generate new weights for the neural network layers.

technique used in [116] and [115] by dividing the chip into cubic temperature cells of silicon and copper layers. Thus, given the floorplan of the multi-core system, the thermal model of the cores and core units could be represented by one or more thermal silicon cells. The thermal model has been extracted based on cell conductances and capacitances as calculated in [114], and the cell geometries (height, width and thickness) for the silicon and copper layers, which are extracted from the chip layout. From the equivalent RC model, the temperature can be modelled from the power consumption and the layout of the chip using the following equation:

$$T_i[k+1] = T_i[k] + \sum_{\forall j \epsilon Adj_i} \alpha_{i,j}(T_j[k] - T_i[k]) + \beta_i P_i[k], \tag{6.7}$$

where $T_i[k]$ and $P_i[k]$ are the temperature and power consumption of the cell, i, at time step k respectively. α_i and β_i are constants that represent the thermal characteristics of the chip which could be extracted as in [114]. Adj_i is the adjacency matrix of the cell i that represents the spatial thermal transfer between the cell i and its adjacent cells. In Appendix D, the MATLAB code for simulating the multi-core thermal model is presented. Figure 6.6 shows the change of the simulated temperature with the change in the power of the corresponding core and its adjacent cores. The input power could be regulated dynamically by adjusting the frequency of the core from a higher-level of

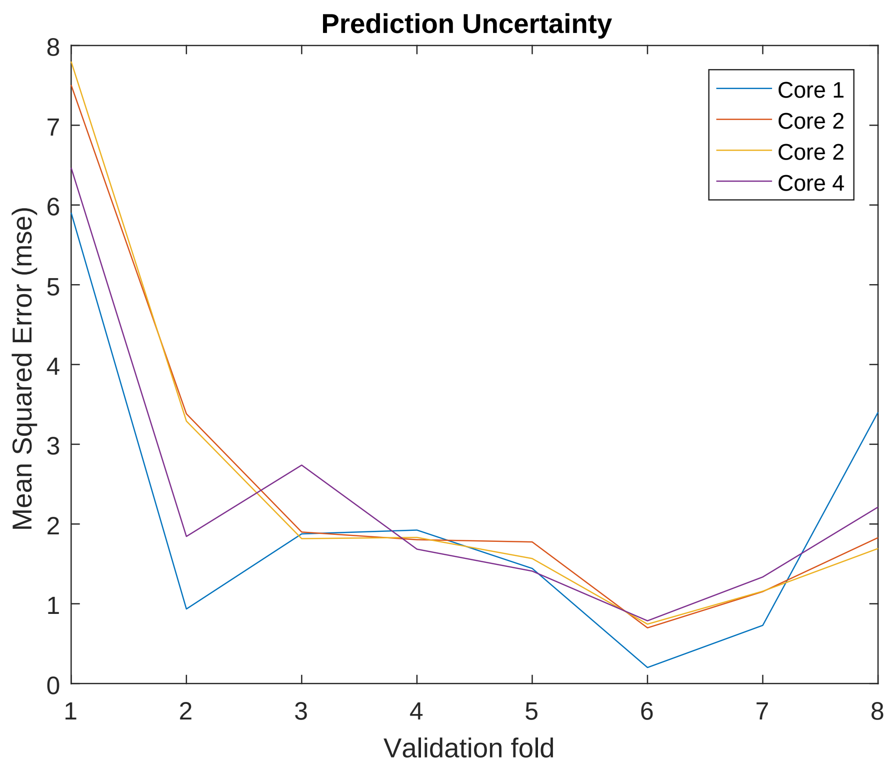


Figure 6.5: Prediction error and uncertainty using linear regression and cross validation. Prediction errors is changing with changing the validation data (uncertainty).

abstraction (e.g., the operating system) by scaling the frequency of the cores. However, this lowers the temperature of the core but incurs a timing overhead.

6.4 Multi-core Ageing Stress

In dynamic power management, the different states of the processor are utilised to optimise the power consumption by putting an idle core or processor into a low power state. We showed in chapter 4 that the BTI stress is worsened when the processor is in an idle state (i.e., static BTI to have one node at the gate level statically stressed with NBTI and another with PBTI). The processor is described as idle when it is powered on but has not been utilised by any useful program. During the idle state, most operating systems run a no-operation instruction in a loop and assign the lowest priority to this task. Therefore, when the processor is "idle", it means that the processor is not actually switched off, power-gated or off-lined. The implication is that the core could be under stress from an ageing perspective. A simple solution like aggressive off-lining could increase the power consumption [117]. The processor could be forced into an idle state by using the clock gating technique for dynamic power reduction in which the clock is disabled for the flip-flops and registers when a high-level signal is enabled. In the

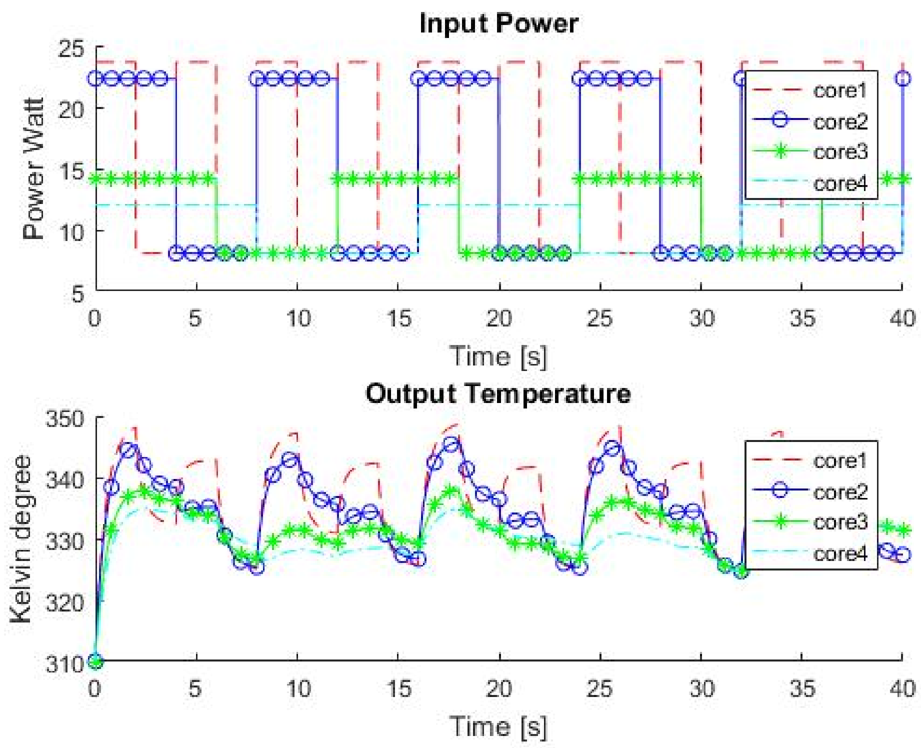


Figure 6.6: Simulated temperatures from the input power of the multi-core system.

literature [118, 119], the idle state has been shown to have a negative impact on the leakage current and power, while it is not considered as a factor that may jeopardise the reliability of the system. The key issue in multi-core systems is that one core could be under static BTI stress (idle) and simultaneously having high temperature generated by adjacent cores. For example, we have simulated the case of idle core as an zero input power to the thermal model (Appendix D) as shown in Figure 6.7, core 2 has no workload (i.e., it is idle) while its adjacent cores are running and processing workloads. As a result, the average temperature of the idle core is affected by the fluctuation of the temperatures of its adjacent cores.

The ageing stress from the BTI perspective at the system level is affected by electrical stress (static stress with idle state) and thermal stress (temperature) as shown in Figure 6.8. Thus, the overall stress is represented as an average stress between the idleness (e.g., implicitly in the input power) and the temperature at the system level to model the margins of stress and named as "normalised data"[3].

[3]e.g., when "normalised data" equals '1', it means that the stress is maximum, which occurs when the temperature is at the highest level and the core is in the idle period. When "normalised data" equals to '0', it means that the stress is minimum, which occurs when the temperature is at the minimum level while the core is not in the idle state. The stress has been normalised to be between '1' and '0'.

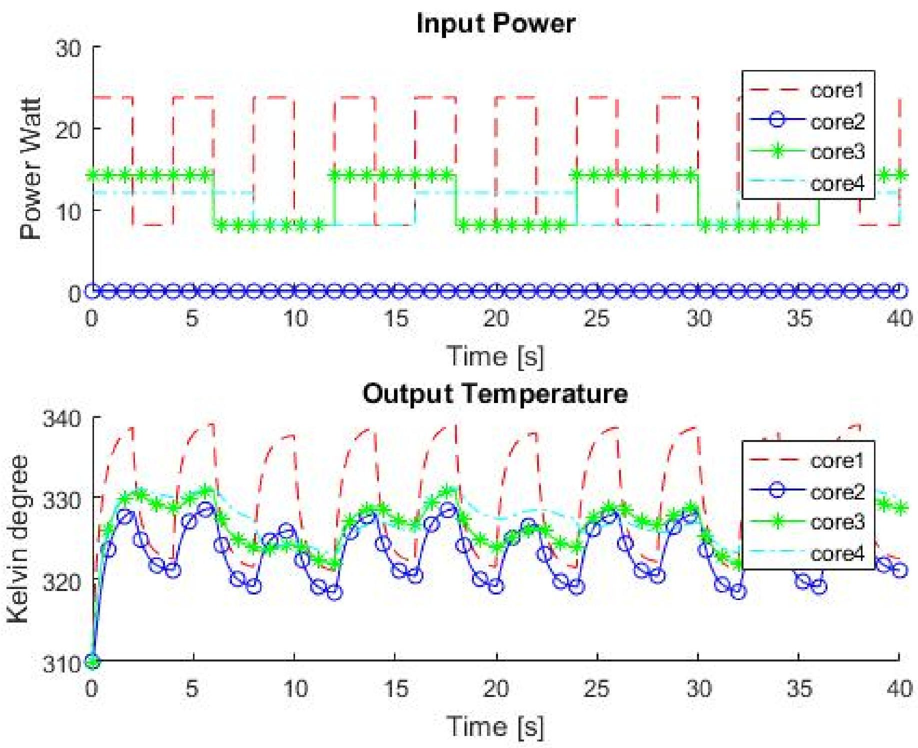


Figure 6.7: Simulated temperature with the effect of idleness of the cores. Core2 is in an idle state but its corresponding temperature is effected by its adjacent cores.

6.5 Proposed Technique

The aim of this section is to reduce the ageing stress not the ageing-induced degradation. The ageing stress model is used to adaptively adjust the workload in order to lower the ageing stress. A simple solution for BTI stress reduction on the stressed core is to swap the core priorities for running an application in order to avoid having one core always under BTI stress and another not stressed. Although it is possible that one core could be relieved from a continuous static stress[4]; however, the overall stress may not necessarily be reduced if the swapping activities and the new scheduling produce the same stress as in the unmitigated scenario.

A proactive approach is proposed to reduce the temperature and idleness in a multi-core system. The technique consists of two phases as shown in Figure 6.10. The temperature of the stressed core is reduced by dynamically adjust the frequency of the adjacent cores and the idleness is reduced by replacing idle process with an activity of low power consumption.

[4]This is illustrated in Figure 6.9. At time 20 seconds, the activity has been migrated from core 2 to core 3, which swaps the idle core with the core under the BTI stress.

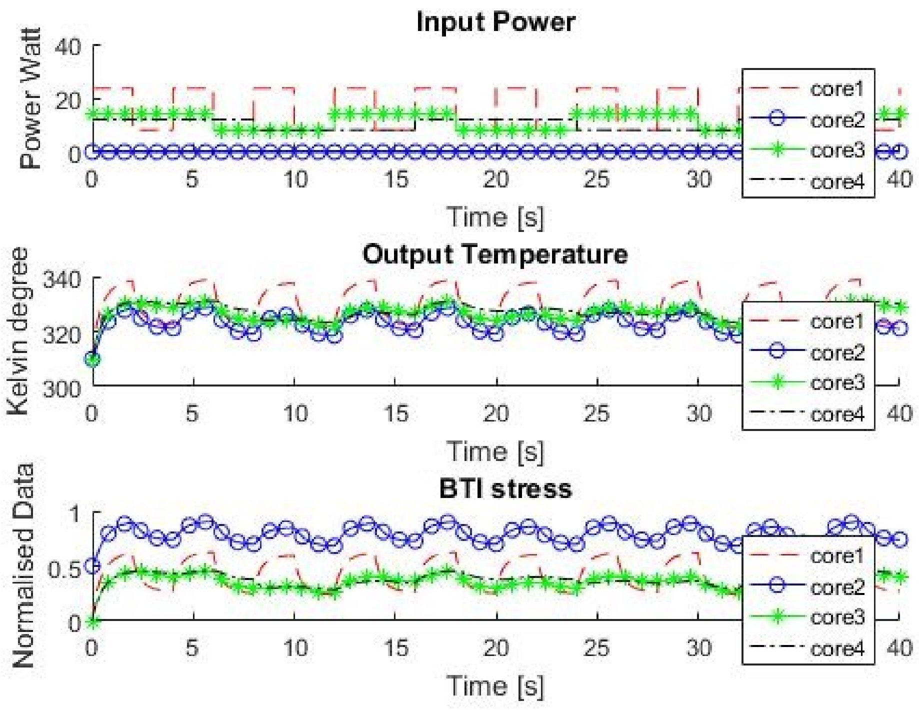


Figure 6.8: System-level simulated BTI stress from core temperatures and activities. The input power represent the activity and idleness of the cores; BTI stress represents the average effect from the electrical stress (idleness) and the thermal stress (core temperature).

The first phase is done offline to model BTI-induced stress at the system level of the multi-core system. The model does not predict the delay or the overall performance degradation of the core; instead, it predicts the input stress that could put the core under the BTI ageing effect. As mentioned earlier in Section 6.1, ageing stress at the system level is affected by two factors: idleness and temperature. Thus, the model should predict these two factors for each core based on the input workload. The proposed ageing stress model needs a profile of inputs of workload sets and outputs of temperature and idleness period in order for each core to be learned. This profile can be collected by running a range of programs from standard benchmarks on a physical or simulated multi-core system. After collecting the profile data, a neural network could be used to model the stress because its ability to model the non-linear behaviour of data.

The second phase of the proposed technique is done on-line by adaptively responding to the ageing stress and feeding the estimated stress to the frequency regulator proposed in [120, 121, 122, 123] that in turn dynamically adjusts the frequency of the adjacent cores so as to lower the temperature of the core under stress.

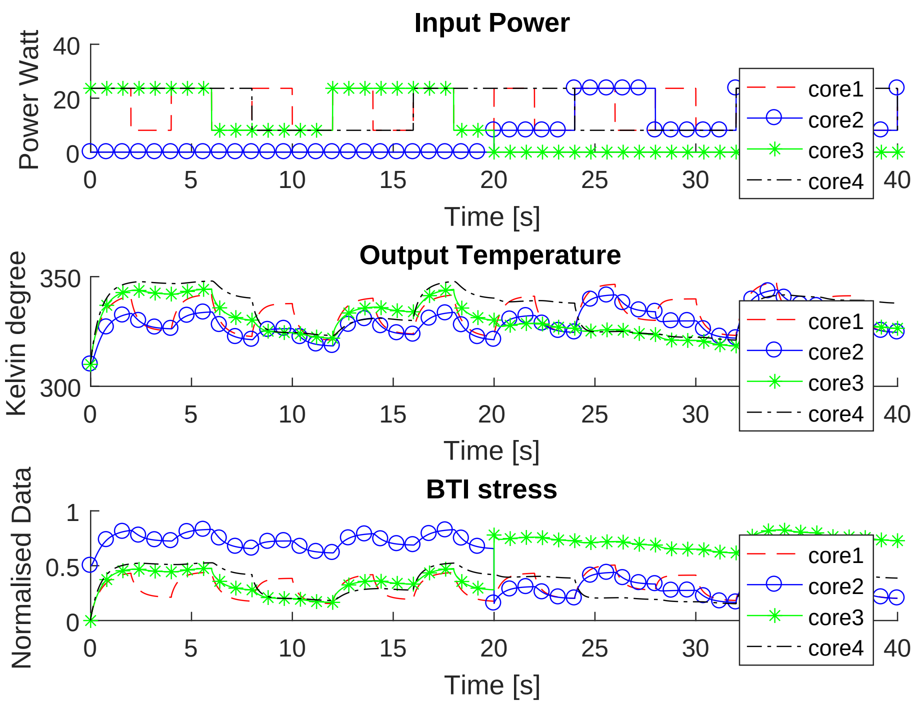


Figure 6.9: Migrating activities from core 2 to core 3.

6.6 Experimental Setup and Simulation Results

To model the ageing stress as a first phase of the proposed technique, the power and idleness are collected using the Sniper Simulator [110] and McPAT [111] from the simulated four-core processor, Xeon x5550 Gainestown, having a standard floorplan as shown in Figure 6.11. These open source tools installed and run on linux Ubuntu 14.04 to simulate The IPC and the power per core according to the configuration presented in Section 6.2. The temperatures and ageing stress per core have been calculated as presented in Sections 6.3 and 6.4, respectively.

Figure 6.12 shows the calculated temperature and ageing stress for parsec and splash2 benchmarks. This ageing stress indicator has been trained using a feedforward neural network that have ability to produce accurate output on data outside its seen training set [112]. The Figure shown in 6.13 with a network size of 10 (i.e., in our case, increasing the network size would not improve the estimation performance), obtained from feeding the temperature and the power consumption per core as training set into the machine learning, which implicitly provide estimates for the workload and idleness. Therefore, the ageing stress models the temperature and combines it with the idleness to generate the normalised ageing stress per core. The estimated ageing stress was found with mean square error of less than 0.185×10^{-3} for the test dataset (see Figure 6.14). Every training round (epoch), the data is divided into three equally data sets (training,

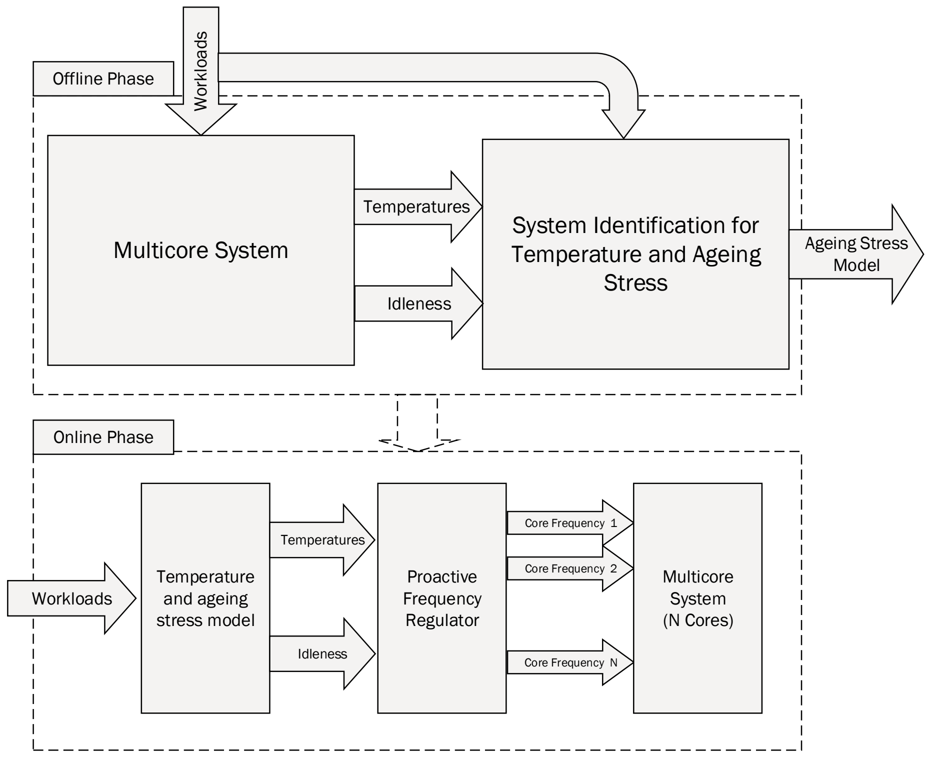


Figure 6.10: The proposed technique.

validating, testing data sets). Data is selected randomly from the data given to be used as a training set during the training phase to find the weights (w) and the bias (b) for the hidden and output layers. The validation set is used to select the training parameters and test set that are used to measure the performance characteristics of the neural network and needs to be unseen during the training phase. Regression (R) has been obtained for each of the datasets for the estimated output to the target as shown in Figure 6.15.

The stress model is generated to estimate the ageing stress proactively, and a frequency regulator is used as the second phase of the proposed technique to adjust the core frequencies for a predefined ageing-stress threshold. In our case, the ageing stress threshold is defined to be 80% (compromising value with the time overhead). In this case, The time overhead is 11.12%, which prevents the ageing stress from going over 80% of its maximum value (see Figure 6.16 (b)). Further optimisation has been done by replacing any idle period of the cores with an activity running at the minimum frequency to reduce the ageing stress by more than 50% and having no timing overhead (see Figure 6.16 (c)).

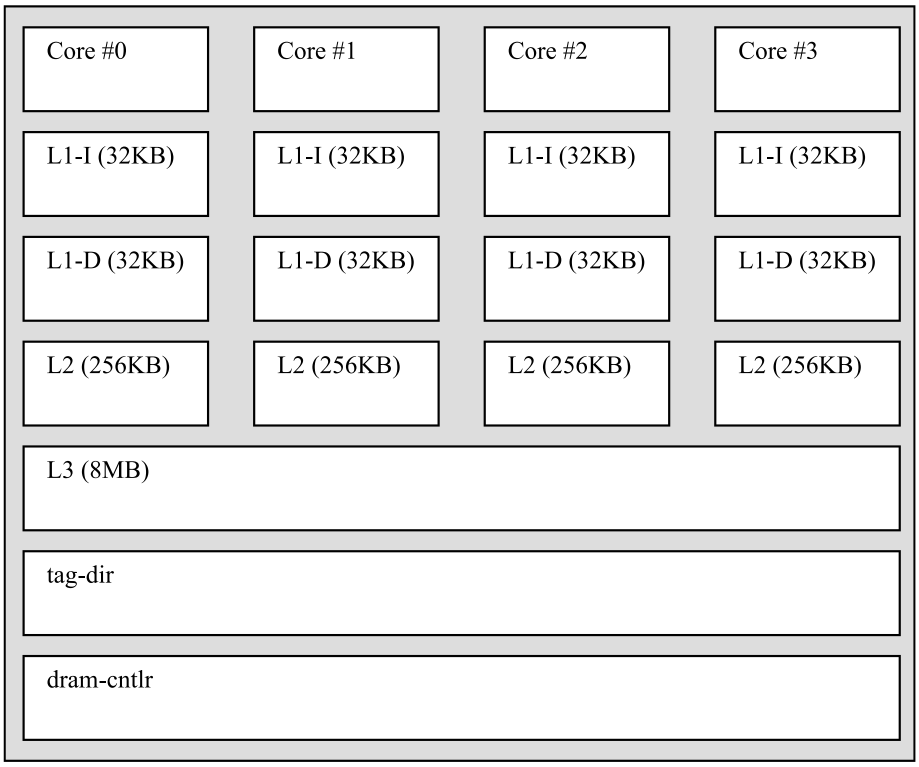


Figure 6.11: Floorplan of the Xeon multi-core processor.

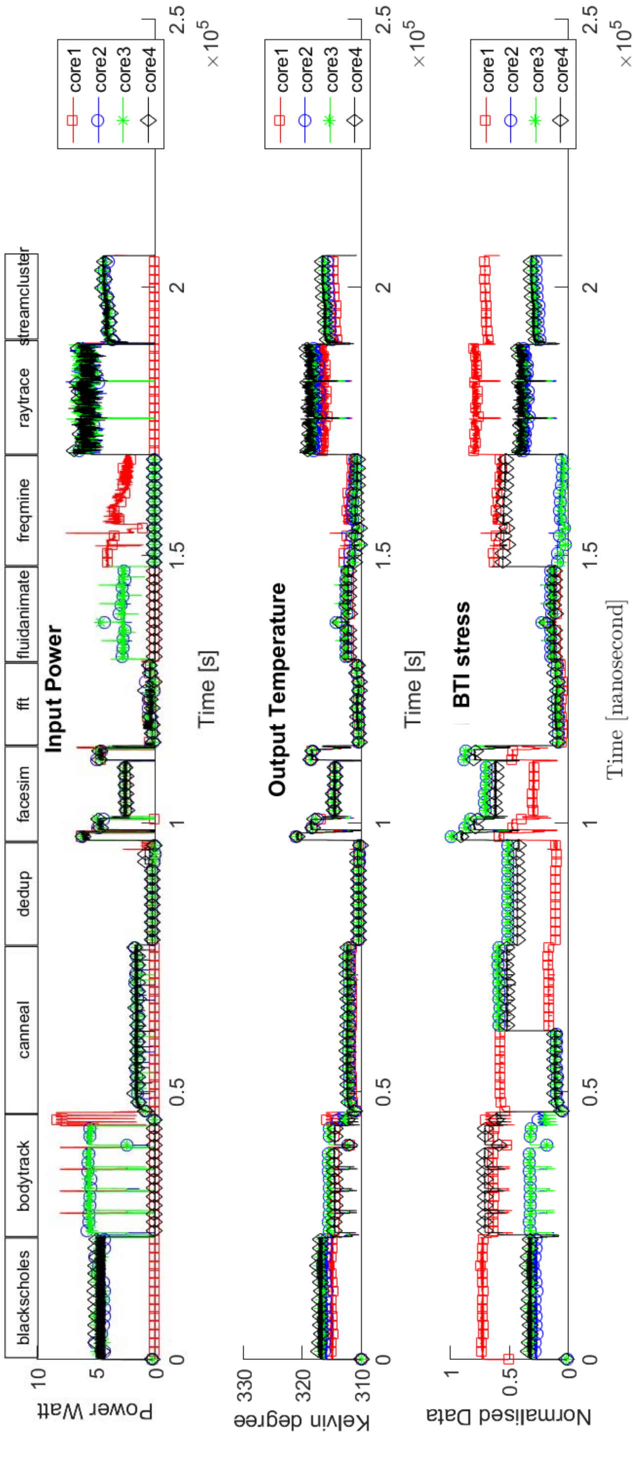


Figure 6.12: Calculated temperatures and ageing stresses for Xeon four-core processor running benchmarks from parsec and splash2.

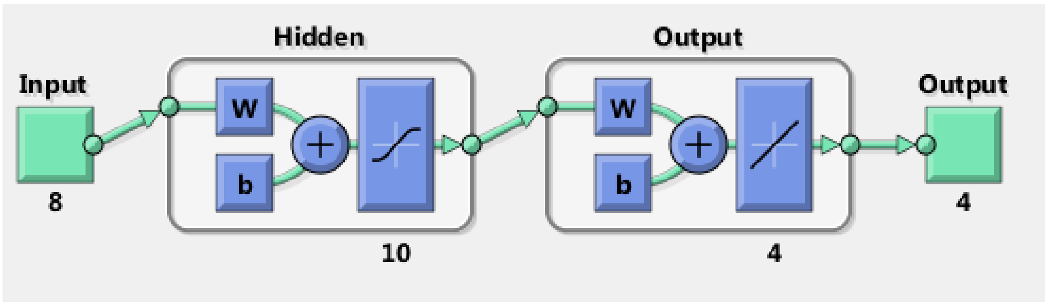


Figure 6.13: Feed-forward neural network; The input of the neural network is the idleness and the temperature for each core (4 cores × 2 = 8 Inputs) and the output is ageing stress per cores (4 Outputs); The hidden layers are responsible to transfer the relationship into linear by multiplication input vector with weights (w) and adding with scaler value (b), finally, pass the results into activation function (e.g, sigmoid) to limit the output between 0 and 1; Output layers are responsible to limit the number of outputs to the required output data (regression).

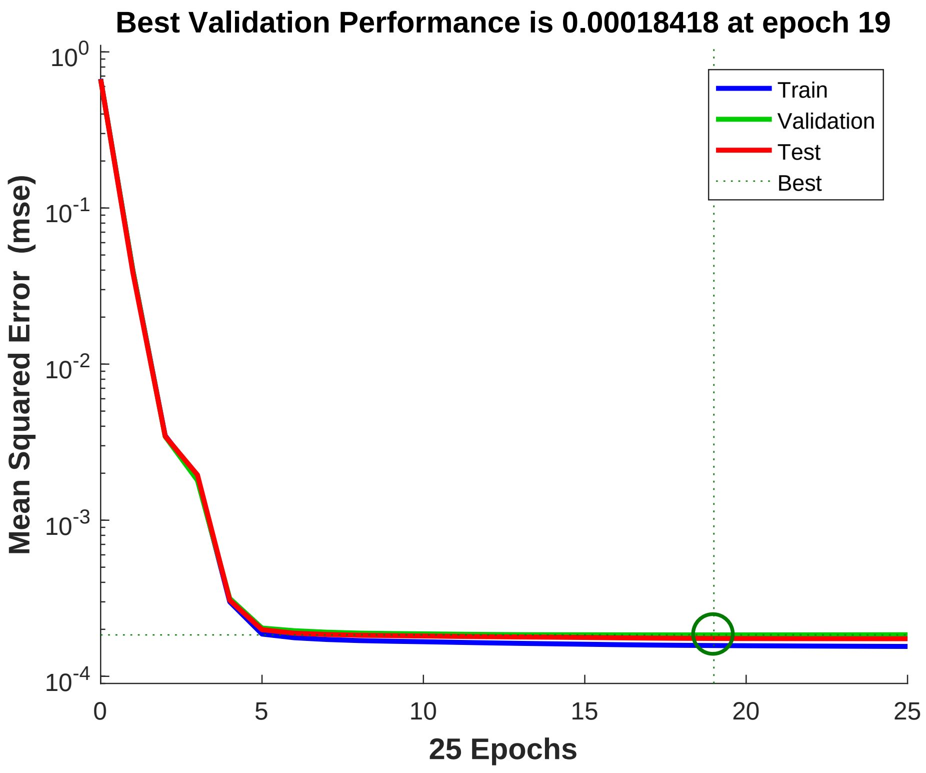


Figure 6.14: Ageing model performance. Every epoch (new training data used), the training update the weights (w) and (b) of the neural network to fit the generated outputs with the target outputs. Best performance is found when the mean squared errors on the validation data is stop decreasing.

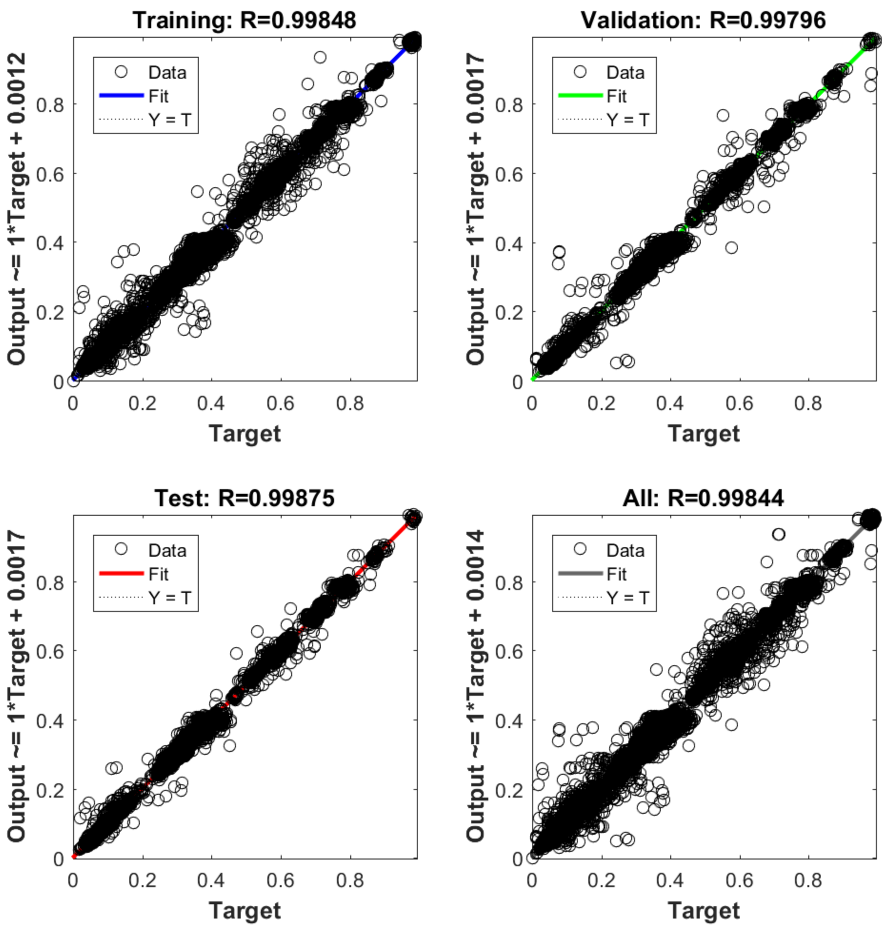


Figure 6.15: Neural network regression of the estimated data to the target data for the training, validation and testing data sets. (y axes) estimated outputs (Y); (x axes) target outputs (T).

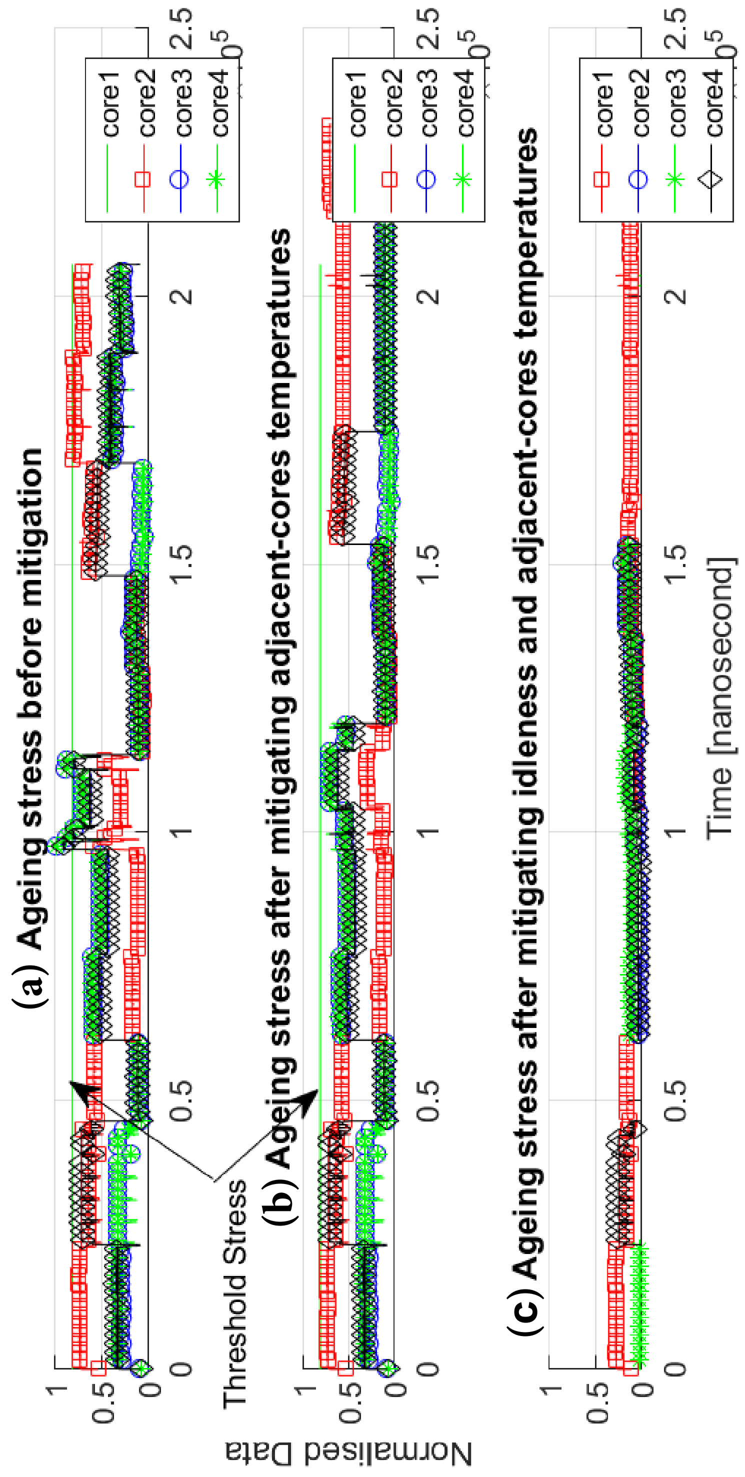


Figure 6.16: Ageing stress mitigation.

6.7 Discussion and Conclusion

Current processors are designed to have temperature sensors. These temperature sensors could be used to collect more accurate data and collect the the temperature profile instead of using temperature simulators. However, the main purpose of this work is to react to the stress proactively. Obtaining the current temperature using temperature sensors could be useful but the solution would be reactive and after the core or the processor must have been put under stress. Thus, it makes sense to predict the temperature using a model rather than obtain it from the readings of temperature sensors.

It should be mentioned that estimating the ageing stress does not necessarily have to be done using a feed-forward neural network, but it is also possible to use other techniques. For example, the radial basis approximation [124, 125] has been used as an alternative technique to the feed-forward neural network. The advantage of using radial basis approximation is that the learning parameters (e.g., network size) do not need to be defined, but it is not the best choice in terms of the approximation accuracy. In Figure 6.17, which the target performance was defined based on the best performance obtained from the feed-forward neural network from the unseen data. However and during the training phase, it never reached for this target and the best performance is 0.25×10^{-3}. Thus, feed-forward neural network has outperformed the radial basis approximation to model ageing stress data. However, we are not trying to prove that the feed-forward neural network is the best method to model ageing stress data but only to prove that it is possible be trained with small estimation errors.

This chapter has proposed an ageing-aware mitigation technique at the system level for multi-core processors. The technique consists of a learning neural network to estimate the high-level ageing stress and then to use this network to adjust the frequencies and workloads among the cores of the processor in a proactive way. The results of the proposed technique show that the ageing stress could be controlled by a limit (e.g., 80% on the whole system with 11.12% time overhead). This time overhead could be utilised to replace any idle process with another activity to reduce the ageing stress to the half.

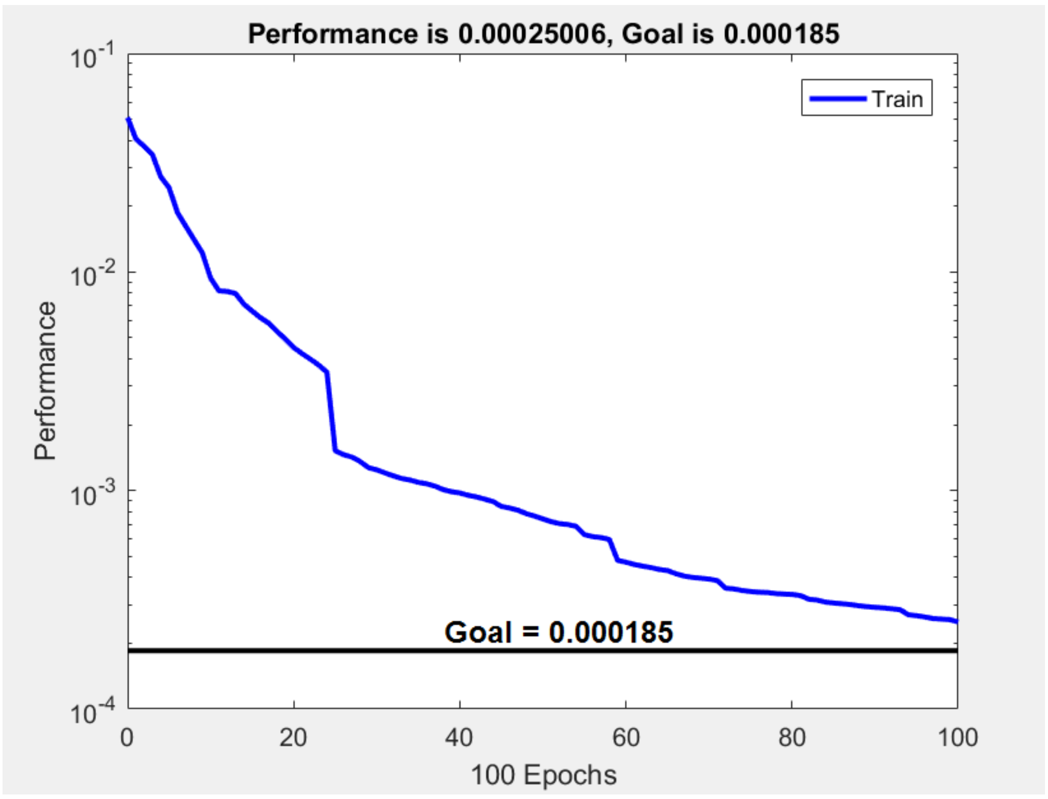


Figure 6.17: The performance of Radial Basis Approximation. The goal is defined from the best performance obtained from the feed-forward neural network.

Chapter 7

Conclusions and Future Work

7.1 Summary

High level workload on application-specific microprocessors could be used to optimise system reliability. High-level proactive methods that consider the real system data inputs can make changes to the low-level device stress and can be used to reduce the ageing-induced delays. Software methods have been used to mitigate ageing behaviours on microprocessor systems. The software approach has been proposed for the units that are considered to be data controllable.

The work described in this thesis has proposed technology-independent and non-intrusive techniques for mitigating CMOS ageing on microprocessors. To accomplish this goal, the workload has been analysed for identifying the real stress and its distribution on the microprocessor units. In both combinational circuits and memory units, workload data propagates as signal probability in the the lower level of abstractions making ageing stress. These signal probabilities have been extracted to be used for reversing the stress and make the stress on the units balanced.

In chapter 3, a signal probability extractor has been developed ans used in two cases. One to evaluate the timing critical path and second to build a library , which considers the BTI-induced delay for aged circuits. The aim of this chapter was to extract the signal probability of the gate-level from the workload that was applied at the application-level and shows that considering the 50% assumption of the primary inputs on the gate-level could not be a good assumption as shown in the example of ageing-induced technology library.

In chapter 4, we focus on the combinational logic circuit to develop a technique for mitigating the BTI-induced delay. Using the signal probability extractor as presented in the previous chapter, we analyse the data-dependent stress and propose two techniques (software and architectural) for balancing the stress on the device-level of the processor.

The results show that by balancing the stress on the timing-critical gates, the lifetime of the processor can be increased to double its normal lifetime.

In chapter 5, we analyse the stress on flip-flops and SRAM-like circuits like the register file. Due to property of the flip-flops that connect with highly switching signal of clock, the HCI- along with the BTI-induced delays have been considered in the analyses. It was found that for the flip-flop, the best technique for mitigating the ageing effect is by balancing its data inputs with the combinational logic nodes as described in the previous chapter. For the register file, an anti-ageing program was proposed to balance the data storage in the SRAM cells using an anti-ageing program for mitigating static noise margin (SNM) degradation. We note that the negative impact of ageing on the memory units is not only on the performance but also on the sensitivity of reading from the memory cell (SNM). The results shows that by applying anti-ageing patterns the reliability of the register file can be improved by up to 65%.

Finally, chapter 6 analyses the impact of temperature on the ageing effect and how ageing stress could be reduced pro-actively in a multi-core processor system. Firstly, temperature and idleness data need to be collected from the real system or simulated one. From an ageing preservative, the combination of idleness and high temperature represents the highest possible stress on a processor core. Therefore, the next step is to use this idle and temperature profile data to estimate the ageing stress using a non-linear estimation model implemented as a neural network. After obtaining the model off-line, we proposed to use the model on-line to predict the ageing stress on-line and pro-actively adapt the frequency of the core for lowering the ageing stress based on the workload assigned.

In conclusion, the contributions of this work can be summarised as follows:

- This work has developed a tool for extracting the low-level ageing stress from the high-level workloads generated by an application.
- This work has proposed a software solution for mitigating the ageing effect on combinational units and the memory units of the microprocessor system.
- Finally, this work has introduced a novel technique to reduce the ageing stress on the multi-core processor system.

7.2 Future Research Directions

For multi-core processors, the available optimisation techniques to find the optimal place-and-route that minimise area, power and temperature could be extended to include time-dependant degradations, as well as idleness and temperature. For example, units having long idle periods, and which are timing critical, can be separated from the highly-active units, in order to avoid idle units from being subjected to high temperatures. The main problem is that the data about the core or units inside the cores and their idle periods can be only be available after fabrication and operation and the aim is to find the optimal floorplan. Otherwise, complex simulated system is required to have these data and re-optimise until optimal solution is found. If no such complex simulator is available, specific units that inherently consider both active and hot (e.g., the flip-flops) or idle and critical (e.g., divider) units could be considered during the optimisation process.

The motivation for the first research direction was to implement the work described in chapter 4 of balancing BTI stress using software on the multi-core processor. A multi-threaded anti-ageing program needs to be developed for multi-core processors. As long as there is an idle core, and current core utilisation is not 100%, an anti-ageing program could be assigned to the idle core to reverse the stress, and at the same time, avoid BTI-induced stress.

Another possible future work is to use the software level to test the ageing degradation. One of the techniques used to mitigate the ageing effects of BTI is dynamic wear-out/NBTI management as presented in Chapter 2. This technique uses delay sensors that usually need to be always active which in turn leads increased power consumption and degradation over time. Ageing sensors need to continuously monitor the targeted paths and should always be active whether the path delay is sensitised or not by the applied data. Software-based solution could be used to support the ageing sensors with an effective test phase by applying application-level test patterns to sensitise the monitored paths. Activating ageing sensors is power consuming, and when there is no alarm from the sensor, it does not mean that there is no ageing because path delay propagation are data-dependent. Ageing test is possible by using high-level data to sensitise the critical path, which makes the ageing sensors active only during the on-line ageing test (this time could be during the idle time of the circuit).

Also, with the growing demand for Internet of Things (IoT) devices and application specific integrated circuits (e.g., Antminer S9 Bitcoin mining processor), studying user-behaviour, by collecting data on-line, could be used for future design optimisations. To implement this, big data techniques will be required to give meaning to this data in order to enable it to be used in predicting future stresses on the system. Ethical issues are important as well and need to be considered with such techniques and people data need to be taken with permission, otherwise, technology could breach human data privacy.

Appendix A

VCD and SP(0)

A.1 VCD Example

The following code is an example of VCD file that generally consists of two parts: header section to define signals and command section to define the values of the signals in sequence of time stamps.

```
$scope module buffer8 $end
$var wire 8 a1 data_bus [7:0] $end
$var wire 1 $ data_bus_valid $end
$var wire 1 % enable $end
$var wire 1 & rx_enable $end
$var wire 1 ’ tx_enable $end
$var wire 1 ( empty $end
$var wire 1 ) under_run $end
$upscope $end
$enddefinitions $end

#0
a11000010 a1
0$
1%
0&
1’
0(
0)
#2211
b0 a1
0’
#2296
a1000 a1
1$
```

```
#2302
0$
#2303
```

A.2 SP(0) Generated File

This is the resulted SP(0) file of the above VCD example.

```
========SCOPE= module======== MODULE= buffer8
SP(0) of the net data_bus[0]=100%
SP(0) of the net data_bus[1]=3%
SP(0) of the net data_bus[2]=100%
SP(0) of the net data_bus[3]=99%
SP(0) of the net data_bus[4]=100%
SP(0) of the net data_bus[5]=100%
SP(0) of the net data_bus[6]=3%
SP(0) of the net data_bus[7]=3%
SP(0) of the net data_bus_valid=99%
SP(0) of the net enable=0%
SP(0) of the net rx_enable=100%
SP(0) of the net tx_enable=3%
SP(0) of the net empty=100%
SP(0) of the net under_run=100%
END_SCOPE
```

A.3 VCD Scanner

The following code is the scanner part of the compiler which written for JFlex scanner genertor inviroment for java [82]. this step will collect the tokens to CUP parser generator.

```
/* This is the source code of VCD file lexer */
import java_cup.runtime.*;
%%
%class Lexer
/* Cup compatibility*/
%cup
/* Macros */
ws = " " | "\t" | "\r"| "\n"
```

```
xz = (x | X | z | Z |0 |1)([!-\~])+
bin_num = (b|B)(x|X|z|Z|"0"|"1")+
dec_num = [0-9]+
ml_comment = ("$"comment)([!-\~|\n | \r | \r\n])*("$"end)
date = ("$"date)(.|\n | \r | \r\n)*("$"end)
version = ("$"version)(.|\n | \r | \r\n)*("$"end)
vector = ("[")[0-9]+(":")[0-9]+("]")
scalar = ("[")[0-9]+("]")
identifier_code = ([!-\~])+
%%

{ws}                {return new Symbol(sym.WS,new String(yytext()); }
{xz}                {return new Symbol(sym.XZ,new String(yytext()); }
{bin_num}              {return new Symbol(sym.BIN_NUM,new
    String(yytext()); }
{dec_num}              {return new Symbol(sym.DEC_NUM,new
    String(yytext()); }
"$"xxxdefinitions         {return new Symbol(sym.ENDDEFNS,new
    String(yytext()); }
{ml_comment}           {return new Symbol(sym.ML_COMMENT,new
    String(yytext()); }
{date}                 {return new Symbol(sym.DATE,new String(yytext());
    }
{version}              {return new Symbol(sym.VERSION,new
    String(yytext()); }
{vector}            {return new Symbol(sym.VECTOR,new String(yytext()); }
{scalar}            {return new Symbol(sym.SCALAR,new String(yytext()); }
"$"end                 {return new Symbol(sym.END,new String(yytext()); }
"$"timescale           {return new Symbol(sym.TIMESCALE,new
    String(yytext()); }
"$"scope            {return new Symbol(sym.SCOPE,new String(yytext()); }
"$"var                 {return new Symbol(sym.VAR,new String(yytext()); }
"$"upscope             {return new Symbol(sym.UPSCOPE,new
    String(yytext()); }
("#")[0-9]+            {return new Symbol(sym.OCTOTHORPE,new
    String(yytext()); }
"$"dumpall             {return new Symbol(sym.DUMPALL,new
    String(yytext()); }
"$"dumpon              {return new Symbol(sym.DUMPON,new
    String(yytext()); }
"$"dumpoff             {return new Symbol(sym.DUMPOFF,new
    String(yytext()); }
"$"dumpvars            {return new Symbol(sym.DUMPVARS,new
    String(yytext()); }
{identifier_code}         {return new Symbol(sym.IDENTIFIER_CODE,new
    String(yytext()); }
```

A.4 VCD Parser to SP(0)

After obtaining the tokens from the scanner, the second step is to parse the VCD file and generate a formal file named SP(0) using a parser generator CUP for java.

```
..........
/* TERMINALS */
terminal  BIN_NUM, DEC_NUM, OCTOTHORPE, XZ, VECTOR, WS, ML_COMMENT,
    DATE, VERSION, IDENTIFIER_CODE, ENDDEFNS, END, TIMESCALE, SCOPE,
    VAR, UPSCOPE, DUMPALL, DUMPON, DUMPOFF, DUMPVARS;
terminal  String SCALAR;
/* NON TERMINALS*/
non terminal U, vcd_file, pattern, vcd_header, decl_command_list,
    decl_command, end_defns, vcd_decl_date, vcd_decl_version,
    vcd_decl_timescale, vcd_scope, vcd_decl_scope, vcd_decl_upscope,
    scope_type,vcd_decl_vars, var_type, simulation_commands,
    simulation_command, simulation_command_p, value_change_p,
    simulation_keyword, simulation_time, value_change,
    scalar_value_change, value, vector_value_change;
non terminal String identifier_code_all;
/* PRECEDENCES:
Associativity values: left | right | nonassoc
Precedences order : precedences that have been declared first have
    lesser priority than the following ones  */

/* Grammar starting symbol */
start with vcd_file;

vcd_file ::= pattern  {: System.out.println("VCD file correctly
    recognized\n"); :};

pattern ::= vcd_header
         | simulation_commands {:
         if(parser.found_scalar2==1)
         parser.print_file("SP(Z) of the net
    "+parser.nam+"="+(int)parser.sp+"%\n");
         //parser.print_file((int)parser.sp+"\n");
         //parser.print_file(parser.nam+"\n");
         if(parser.found_vector2==1)
         //print vector here with loop
         {
         int i = parser.w;
         i--;
         while(i>=0){
         parser.print_file("SP(Z) of the net
    "+parser.nam+"["+(parser.w-1-i)+"]"+"="+(int)parser.sp_array[i]+"%\n");
         //parser.print_file((int)parser.sp_array[i]+"\n");
         //parser.print_file(parser.nam+"\n");

         i--;}
         }
         :}

;


/* HEADER */

vcd_header ::= decl_command_list end_defns
;
end_defns
    ::= ENDDEFNS WS END WS
    ;
decl_command_list ::=  decl_command_list decl_command | decl_command
;

decl_command
    ::= vcd_decl_date         {: System.out.println("data file
    correctly recognized\n"); :}
    | vcd_decl_version        {: System.out.println("version file
    correctly recognized\n"); :}
    | vcd_decl_timescale      {: System.out.println("timescale file
    correctly recognized\n"); :}
    | vcd_scope            {: System.out.println("scope file correctly
    recognized\n"); :}
    | vcd_decl_vars           {: System.out.println("var file correctly
    recognized\n"); :}
   | ML_COMMENT WS            {: System.out.println("comment file
    correctly recognized\n"); :}
;

vcd_decl_date
    ::= DATE WS
    ;
vcd_decl_version
    ::= VERSION WS
    ;
vcd_decl_timescale
```

```
    ::= TIMESCALE WS DEC_NUM WS IDENTIFIER_CODE WS END WS
    ;
vcd_scope
    ::= vcd_decl_scope decl_command_list vcd_decl_upscope |
    vcd_decl_scope vcd_decl_upscope
    ;
vcd_decl_scope
    ::= SCOPE WS identifier_code_all:S WS identifier_code_all:M WS END
    WS  // bypass the RESULT value from down
   {:
   parser.print_file("========SCOPE= " + S +"======== MODULE= "+M+"\n");
   System.out.println("SCOPE= " + S +" MODULE= "+M+"\n");
   :}
    ;
vcd_decl_upscope
    ::= UPSCOPE WS END WS  {:
   parser.print_file("END_SCOPE\n");
   System.out.println("END_SCOPE\n"); :}
    ;

............

/*COMMANDS*/
simulation_commands
   ::=  simulation_commands value_change
   | value_change
   ;

value_change
   ::=     BIN_NUM:BN WS OCTOTHORPE:OC WS
   {:
   if((OC.toString()).equals(parser.searchSym))
{
parser.found_vector=1;
int w = parser.w;
parser.STR2 = (parser.delete_first_char(BN)).toString();

String alignedSTR= new String(parser.stringAlign(parser.STR2,w));

char[] charArray = alignedSTR.toCharArray();

w--; // to start with w= 7 which is equal to [0]
while(w>=0)
{

if (charArray[w]=='0')  //stress
                                     {
```

```
                                    parser.found_zero_array[w] = 1;
                                    // this means between a timestamp
    and a timestamp if there is no zero in the signal do not add
    timestamp to the num of zeros above

    //parser.num_zero_array[w]=parser.num_zero_array[w];
                                    }
                                 else
                                    {
                                    parser.found_zero_array[w] = 0;
                                    }

w--;
}}
   ...........
   ;
```

Appendix B

3-dimentional Lookup Tables

This appendix shows the lookup tables (LUT) extracted for the main logic gates (INV, NAND, NOR) for the 90nm Synopsys technology library. **tr\C_load** means the column represents the input transition delay (picosecond) and the row represent the output capacitance (femtofarad). the contain of the tables are the delays that changes with the changes of input transition delay, output capacitance load and the signal probability of the input SP(0).

B.1 Invertor gate (INVX1)

B.1.1 Falling propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4589	17.159	28.957	47.705	83.99	166.59	339.35
32	8.1522	18.952	30.386	46.579	85.274	167.02	339.56
64	0.67648	21.986	34.1	50.837	91.372	169.91	324.75
128	-9.0481	20.975	38.432	57.461	96.503	177.31	336
256	-1.6807	17.417	43.668	69.5	112.64	189.7	352.68
512	-38.453	-4.2075	31.712	66.654	128.92	220.12	373.79
1024	-66.23	-69.631	-54.923	-12.701	90.981	235.67	426.36

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4579	17.024	28.948	47.706	83.971	166.57	339.32
32	4.4659	18.956	30.377	46.566	85.259	167	339.53
64	0.19567	21.878	34.044	50.787	91.337	169.85	324.73
128	-9.1273	20.748	38.265	57.335	96.409	177.23	335.91
256	-16.879	17.18	43.428	69.299	112.54	189.64	352.62
512	-40.312	-5.3748	30.726	65.967	128.6	220.03	373.75
1024	-71.599	-73.042	-57.324	-14.029	90.538	235.14	425.95

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4392	17.028	28.944	47.713	83.958	166.56	339.29
32	8.188	18.96	30.384	46.557	85.248	166.98	339.52
64	0.33031	21.813	34.01	50.754	91.314	169.82	324.72
128	-9.1688	20.626	38.177	57.267	95.837	177.19	335.84
256	0.18695	17.029	43.263	69.175	112.47	189.61	352.59
512	-44.337	-6.0619	30.08	65.571	128.41	219.98	373.72
1024	-101.02	-75.311	-58.847	-14.741	90.393	234.83	425.75

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4549	17.031	28.942	47.707	83.948	166.55	339.28
32	4.3922	18.963	30.378	46.55	85.239	166.97	339.5
64	0.4481	21.765	33.984	50.728	91.296	169.79	324.7
128	-9.196	20.544	38.119	57.22	95.806	177.15	335.8
256	1.1021	16.912	43.129	69.071	112.42	189.58	352.56
512	-45.379	-6.5564	29.971	65.287	128.28	219.95	373.7
1024	-120.91	-77.107	-60.007	-15.212	90.35	234.62	425.61

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.471	17.034	28.942	47.699	83.939	166.54	339.26
32	4.3698	18.965	30.373	46.544	85.232	166.96	339.49
64	0.53033	21.728	33.964	50.705	91.28	169.76	324.7
128	-10.008	20.484	38.077	57.185	95.782	177.13	335.76
256	-18.476	16.812	43.014	68.983	112.37	189.56	352.53
512	-43.527	-6.9465	29.62	65.063	128.18	220	373.69
1024	-80.705	-78.646	-60.968	-15.554	90.358	234.47	425.52

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4728	17.042	28.942	47.691	83.931	166.53	339.25
32	8.2228	18.968	30.368	46.538	85.225	166.96	339.48
64	0.5433	21.696	33.946	50.685	91.265	169.73	324.69
128	-9.2306	20.436	38.044	57.157	95.762	177.1	335.72
256	3.1307	16.72	42.906	68.904	112.33	189.54	352.51
512	-44.366	-7.2744	29.32	64.874	128.09	219.96	373.67
1024	-87.372	-80.044	-61.812	-15.817	90.397	234.35	425.44

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4746	17.044	28.942	47.684	83.923	166.52	339.24
32	8.2335	18.97	30.364	46.533	85.219	166.95	339.47
64	2.2123	21.669	33.93	50.666	91.252	169.71	324.68
128	-9.2426	20.396	38.017	57.133	95.745	177.08	335.69
256	4.3587	16.632	42.801	68.828	112.29	189.51	352.49
512	-45.185	-7.5658	29.048	64.705	128.01	219.92	373.66
1024	-109.35	-81.382	-62.594	-16.029	90.458	234.25	425.38

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4766	17.047	28.944	47.676	83.915	166.51	339.22
32	7.1888	18.973	30.36	46.527	85.212	166.94	339.46
64	0.54991	21.643	33.915	50.647	91.238	169.68	324.67
128	-9.2525	20.361	37.993	57.112	95.73	177.06	335.66
256	5.87	16.541	42.693	68.753	112.25	189.49	352.47
512	-46.029	-7.8381	28.788	64.544	127.93	219.89	373.65
1024	-92.053	-82.733	-63.356	-16.205	90.543	234.16	425.32

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4792	17.051	28.94	47.667	83.906	166.5	339.21
32	6.7932	18.975	30.356	46.521	85.205	166.93	339.44
64	1.8657	21.617	33.9	50.626	91.223	169.65	324.66
128	-9.2611	20.327	37.972	57.091	95.714	177.04	335.62
256	7.9709	16.439	42.572	68.67	112.21	189.47	352.44
512	-46.985	-8.1148	28.514	64.377	127.85	219.86	373.63
1024	-94.697	-84.227	-64.164	-16.357	90.658	234.07	425.27

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.4859	17.06	28.928	47.651	83.889	166.48	339.18
32	6.2046	18.981	30.348	46.509	85.191	166.91	339.42
64	0.52788	21.575	33.873	50.587	91.195	169.6	324.65
128	-9.2709	20.279	37.941	57.059	95.689	177.01	335.56
256	13.912	15.61	42.339	68.517	112.13	189.43	352.4
512	-48.845	-8.5675	28.038	64.09	127.71	219.8	373.61
1024	-99.735	-87.025	-65.576	-16.533	90.927	233.94	425.19

B.1.2 Rising propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.58E+00	2.70E+01	4.61E+01	7.79E+01	1.35E+02	2.80E+02	5.31E+02
32	1.23E+01	3.24E+01	4.96E+01	8.10E+01	1.43E+02	2.73E+02	5.79E+02
64	1.78E+01	4.07E+01	5.93E+01	8.84E+01	1.48E+02	2.88E+02	5.48E+02
128	1.94E+01	5.55E+01	7.51E+01	1.02E+02	1.69E+02	2.89E+02	5.73E+02
256	3.80E+01	6.87E+01	9.89E+01	1.30E+02	1.92E+02	3.27E+02	6.04E+02
512	9.98E+01	1.22E+02	1.49E+02	1.92E+02	2.65E+02	3.86E+02	6.29E+02
1024	1.52E+02	1.92E+02	1.84E+02	2.59E+02	3.64E+02	5.23E+02	7.68E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.82E+00	2.80E+01	4.79E+01	8.13E+01	1.40E+02	2.90E+02	5.60E+02
32	1.27E+01	3.37E+01	5.21E+01	8.40E+01	1.51E+02	2.83E+02	6.02E+02
64	1.88E+01	4.22E+01	6.14E+01	9.22E+01	1.56E+02	2.98E+02	6.16E+02
128	2.18E+01	5.77E+01	7.77E+01	1.07E+02	1.77E+02	3.02E+02	5.93E+02
256	4.43E+01	7.30E+01	1.03E+02	1.34E+02	2.00E+02	3.40E+02	6.39E+02
512	1.10E+02	1.32E+02	1.58E+02	1.98E+02	2.73E+02	4.00E+02	6.57E+02
1024	1.67E+02	1.57E+02	2.08E+02	2.78E+02	3.83E+02	5.42E+02	8.02E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.98E+00	2.87E+01	4.91E+01	8.36E+01	1.45E+02	2.97E+02	5.78E+02
32	1.30E+01	3.46E+01	5.40E+01	8.60E+01	1.56E+02	2.90E+02	6.17E+02
64	1.96E+01	4.32E+01	6.29E+01	9.49E+01	1.61E+02	3.05E+02	6.34E+02
128	2.47E+01	5.93E+01	7.95E+01	1.09E+02	1.83E+02	3.12E+02	6.07E+02
256	4.91E+01	7.55E+01	1.05E+02	1.37E+02	2.05E+02	3.48E+02	6.59E+02
512	1.17E+02	1.39E+02	1.66E+02	2.04E+02	2.83E+02	4.09E+02	6.81E+02
1024	1.77E+02	1.50E+02	2.22E+02	2.89E+02	3.96E+02	5.56E+02	8.26E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.10E+00	2.93E+01	5.01E+01	8.54E+01	1.50E+02	3.03E+02	6.39E+02
32	1.33E+01	3.54E+01	5.55E+01	8.77E+01	1.71E+02	2.96E+02	6.30E+02
64	2.01E+01	4.41E+01	6.41E+01	9.70E+01	1.66E+02	3.11E+02	6.48E+02
128	2.68E+01	6.05E+01	8.10E+01	1.12E+02	1.87E+02	3.21E+02	6.19E+02
256	5.33E+01	7.64E+01	1.07E+02	1.40E+02	2.09E+02	3.55E+02	6.75E+02
512	1.24E+02	1.46E+02	1.74E+02	2.21E+02	2.87E+02	4.16E+02	6.99E+02
1024	1.84E+02	1.60E+02	2.33E+02	2.98E+02	4.07E+02	5.67E+02	8.46E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.22E+00	2.99E+01	5.10E+01	8.71E+01	1.54E+02	3.08E+02	6.53E+02
32	1.35E+01	3.60E+01	5.92E+01	8.92E+01	1.75E+02	3.01E+02	6.41E+02
64	2.06E+01	4.47E+01	6.52E+01	9.88E+01	1.70E+02	3.17E+02	6.61E+02
128	2.83E+01	6.16E+01	8.24E+01	1.14E+02	1.91E+02	3.29E+02	6.30E+02
256	5.72E+01	7.73E+01	1.09E+02	1.42E+02	2.12E+02	3.61E+02	6.89E+02
512	1.30E+02	1.53E+02	1.81E+02	2.22E+02	2.90E+02	4.23E+02	7.16E+02
1024	1.88E+02	1.70E+02	2.43E+02	3.06E+02	4.17E+02	5.79E+02	8.63E+02

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.32E+00	3.07E+01	5.18E+01	8.86E+01	1.58E+02	3.13E+02	6.66E+02
32	1.37E+01	3.66E+01	6.07E+01	9.06E+01	1.78E+02	3.06E+02	6.51E+02
64	2.10E+01	4.53E+01	6.62E+01	1.00E+02	1.75E+02	3.22E+02	6.73E+02
128	2.95E+01	6.25E+01	8.39E+01	1.15E+02	1.94E+02	3.37E+02	6.40E+02
256	5.91E+01	7.82E+01	1.10E+02	1.44E+02	2.16E+02	3.67E+02	7.02E+02
512	1.36E+02	1.59E+02	2.02E+02	2.24E+02	2.94E+02	4.29E+02	7.32E+02
1024	1.86E+02	1.79E+02	2.52E+02	3.13E+02	4.26E+02	5.88E+02	8.78E+02

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.43E+00	3.14E+01	5.27E+01	9.01E+01	1.61E+02	3.18E+02	6.78E+02
32	1.40E+01	3.72E+01	6.20E+01	9.20E+01	1.82E+02	3.11E+02	6.61E+02
64	2.14E+01	4.60E+01	6.72E+01	1.02E+02	1.78E+02	3.27E+02	6.84E+02
128	3.06E+01	6.35E+01	8.51E+01	1.17E+02	1.97E+02	3.45E+02	6.50E+02
256	5.91E+01	7.91E+01	1.12E+02	1.46E+02	2.19E+02	3.72E+02	7.14E+02
512	1.43E+02	1.66E+02	2.03E+02	2.25E+02	2.97E+02	4.35E+02	7.47E+02
1024	1.72E+02	1.89E+02	2.61E+02	3.21E+02	4.35E+02	5.96E+02	8.93E+02

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.53E+00	3.22E+01	5.36E+01	9.16E+01	1.65E+02	3.23E+02	6.91E+02
32	1.42E+01	3.78E+01	6.34E+01	9.35E+01	1.85E+02	3.17E+02	6.72E+02
64	2.18E+01	4.68E+01	6.83E+01	1.04E+02	1.92E+02	3.32E+02	6.96E+02
128	3.15E+01	6.45E+01	8.64E+01	1.19E+02	2.00E+02	3.53E+02	6.60E+02
256	5.92E+01	8.01E+01	1.13E+02	1.48E+02	2.22E+02	3.78E+02	7.26E+02
512	1.49E+02	1.74E+02	2.04E+02	2.26E+02	3.00E+02	4.41E+02	7.62E+02
1024	1.78E+02	1.97E+02	2.70E+02	3.28E+02	4.43E+02	6.17E+02	9.08E+02

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.66E+00	3.30E+01	5.46E+01	9.33E+01	1.70E+02	3.29E+02	7.05E+02
32	1.45E+01	3.84E+01	6.48E+01	9.52E+01	1.89E+02	3.23E+02	6.83E+02
64	2.23E+01	4.77E+01	6.99E+01	1.06E+02	1.97E+02	3.38E+02	7.09E+02
128	3.25E+01	6.56E+01	8.77E+01	1.21E+02	2.04E+02	3.62E+02	6.72E+02
256	5.93E+01	8.13E+01	1.15E+02	1.50E+02	2.26E+02	3.84E+02	7.41E+02
512	1.58E+02	1.91E+02	2.04E+02	2.28E+02	3.04E+02	4.48E+02	7.79E+02
1024	1.84E+02	2.16E+02	2.79E+02	3.37E+02	4.53E+02	6.27E+02	9.24E+02

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.89E+00	3.46E+01	5.70E+01	9.67E+01	1.79E+02	3.41E+02	7.32E+02
32	1.50E+01	3.97E+01	6.76E+01	9.86E+01	1.96E+02	3.38E+02	7.07E+02
64	2.32E+01	4.94E+01	7.35E+01	1.09E+02	2.05E+02	3.50E+02	7.34E+02
128	3.45E+01	6.77E+01	9.10E+01	1.25E+02	2.11E+02	3.99E+02	6.96E+02
256	5.94E+01	8.41E+01	1.19E+02	1.54E+02	2.34E+02	3.97E+02	7.68E+02
512	1.81E+02	1.92E+02	2.06E+02	2.31E+02	3.11E+02	4.62E+02	8.12E+02
1024	1.95E+02	2.37E+02	2.95E+02	3.54E+02	4.72E+02	6.46E+02	9.57E+02

B.1.3 Falling transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.38E+00	2.76E+01	5.41E+01	8.55E+01	1.85E+02	3.25E+02	6.93E+02
32	1.25E+01	3.06E+01	5.09E+01	9.09E+01	1.83E+02	3.26E+02	6.81E+02
64	1.94E+01	3.25E+01	5.14E+01	8.31E+01	1.62E+02	3.51E+02	6.70E+02
128	3.24E+01	5.82E+01	6.41E+01	9.16E+01	1.66E+02	3.21E+02	6.86E+02
256	6.41E+01	8.31E+01	9.76E+01	1.19E+02	1.73E+02	3.38E+02	6.53E+02
512	8.09E+01	1.25E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.15E+02
1024	2.45E+02	2.42E+02	2.88E+02	3.30E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.35E+00	2.91E+01	5.37E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.23E+01	3.04E+01	5.09E+01	9.07E+01	1.83E+02	3.26E+02	6.81E+02
64	1.85E+01	3.24E+01	5.14E+01	8.30E+01	1.62E+02	3.51E+02	6.70E+02
128	3.11E+01	5.87E+01	6.43E+01	9.17E+01	1.64E+02	3.21E+02	6.87E+02
256	6.94E+01	8.15E+01	9.80E+01	1.19E+02	1.73E+02	3.40E+02	6.53E+02
512	7.55E+01	1.22E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.16E+02
1024	2.25E+02	2.32E+02	2.82E+02	3.25E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.38E+00	2.80E+01	5.40E+01	8.56E+01	1.85E+02	3.25E+02	6.94E+02
32	9.95E+00	3.06E+01	5.09E+01	9.09E+01	1.83E+02	3.26E+02	6.81E+02
64	1.93E+01	3.25E+01	5.14E+01	8.30E+01	1.62E+02	3.51E+02	6.70E+02
128	3.17E+01	5.84E+01	6.42E+01	9.16E+01	1.66E+02	3.21E+02	6.87E+02
256	4.61E+01	8.25E+01	9.78E+01	1.19E+02	1.73E+02	3.39E+02	6.53E+02
512	7.92E+01	1.24E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.15E+02
1024	2.40E+02	2.39E+02	2.85E+02	3.29E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.34E+00	2.91E+01	5.36E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.23E+01	3.04E+01	5.09E+01	9.06E+01	1.83E+02	3.26E+02	6.81E+02
64	1.72E+01	3.24E+01	5.14E+01	8.29E+01	1.62E+02	3.51E+02	6.70E+02
128	3.10E+01	5.87E+01	6.43E+01	9.17E+01	1.63E+02	3.21E+02	6.87E+02
256	7.07E+01	8.13E+01	9.81E+01	1.19E+02	1.73E+02	3.40E+02	6.53E+02
512	7.48E+01	1.21E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.16E+02
1024	1.97E+02	2.31E+02	2.81E+02	3.25E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.38E+00	2.84E+01	5.40E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.24E+01	3.05E+01	5.09E+01	9.08E+01	1.83E+02	3.26E+02	6.81E+02
64	1.90E+01	3.24E+01	5.14E+01	8.30E+01	1.62E+02	3.51E+02	6.70E+02
128	3.13E+01	5.85E+01	6.42E+01	9.16E+01	1.64E+02	3.21E+02	6.87E+02
256	6.62E+01	8.22E+01	9.78E+01	1.19E+02	1.73E+02	3.39E+02	6.53E+02
512	8.82E+01	1.24E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.15E+02
1024	2.05E+02	2.37E+02	2.84E+02	3.27E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.34E+00	2.91E+01	5.35E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.10E+01	3.04E+01	5.09E+01	9.06E+01	1.83E+02	3.26E+02	6.81E+02
64	1.82E+01	3.24E+01	5.14E+01	8.29E+01	1.62E+02	3.51E+02	6.70E+02
128	3.10E+01	5.87E+01	6.44E+01	9.17E+01	1.63E+02	3.21E+02	6.87E+02
256	7.24E+01	8.12E+01	9.82E+01	1.19E+02	1.73E+02	3.40E+02	6.53E+02
512	7.40E+01	1.21E+02	1.43E+02	1.81E+02	2.42E+02	3.39E+02	7.16E+02
1024	2.20E+02	2.30E+02	2.81E+02	3.24E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.37E+00	2.88E+01	5.39E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.00E+01	3.04E+01	5.09E+01	9.08E+01	1.83E+02	3.26E+02	6.81E+02
64	1.88E+01	3.24E+01	5.14E+01	8.30E+01	1.62E+02	3.51E+02	6.70E+02
128	3.12E+01	5.86E+01	6.43E+01	9.16E+01	1.64E+02	3.21E+02	6.87E+02
256	6.71E+01	8.19E+01	9.79E+01	1.19E+02	1.73E+02	3.39E+02	6.53E+02
512	8.72E+01	1.23E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.15E+02
1024	2.32E+02	2.35E+02	2.83E+02	3.27E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.33E+00	2.85E+01	5.35E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.07E+01	3.04E+01	5.09E+01	9.05E+01	1.83E+02	3.26E+02	6.81E+02
64	1.50E+01	3.24E+01	5.14E+01	8.29E+01	1.62E+02	3.51E+02	6.70E+02
128	3.09E+01	5.88E+01	6.44E+01	9.17E+01	1.63E+02	3.21E+02	6.87E+02
256	7.47E+01	8.10E+01	9.82E+01	1.19E+02	1.73E+02	3.41E+02	6.53E+02
512	7.32E+01	1.21E+02	1.43E+02	1.81E+02	2.42E+02	3.39E+02	7.16E+02
1024	2.18E+02	2.28E+02	2.80E+02	3.23E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.35E+00	2.91E+01	5.38E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.00E+01	3.04E+01	5.09E+01	9.07E+01	1.83E+02	3.26E+02	6.81E+02
64	1.87E+01	3.24E+01	5.14E+01	8.30E+01	1.62E+02	3.51E+02	6.70E+02
128	3.45E+01	5.86E+01	6.43E+01	9.17E+01	1.64E+02	3.21E+02	6.87E+02
256	4.46E+01	8.17E+01	9.80E+01	1.19E+02	1.73E+02	3.40E+02	6.53E+02
512	7.62E+01	1.22E+02	1.43E+02	1.80E+02	2.42E+02	3.39E+02	7.16E+02
1024	2.31E+02	2.34E+02	2.82E+02	3.26E+02	3.82E+02	5.42E+02	7.10E+02

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	5.32E+00	2.84E+01	5.35E+01	8.57E+01	1.85E+02	3.25E+02	6.94E+02
32	1.04E+01	3.04E+01	5.09E+01	9.04E+01	1.83E+02	3.26E+02	6.81E+02
64	1.79E+01	3.24E+01	5.14E+01	8.29E+01	1.62E+02	3.51E+02	6.70E+02
128	3.08E+01	5.88E+01	6.44E+01	9.17E+01	1.64E+02	3.21E+02	6.87E+02
256	8.13E+01	8.24E+01	9.84E+01	1.20E+02	1.73E+02	3.41E+02	6.53E+02
512	7.16E+01	1.20E+02	1.43E+02	1.81E+02	2.42E+02	3.39E+02	7.17E+02
1024	2.13E+02	2.26E+02	2.80E+02	3.22E+02	3.82E+02	5.42E+02	7.10E+02

B.1.4 Rising transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.77E+00	3.44E+01	6.24E+01	1.12E+02	2.09E+02	4.15E+02	8.49E+02
32	1.17E+01	3.53E+01	6.43E+01	1.07E+02	2.15E+02	4.06E+02	8.62E+02
64	1.90E+01	4.00E+01	6.61E+01	1.08E+02	2.10E+02	4.12E+02	8.57E+02
128	3.81E+01	6.61E+01	8.01E+01	1.16E+02	2.13E+02	4.12E+02	8.21E+02
256	7.51E+01	1.13E+02	1.33E+02	1.57E+02	2.55E+02	4.23E+02	8.72E+02
512	1.37E+02	1.63E+02	2.06E+02	2.36E+02	3.05E+02	4.68E+02	8.81E+02
1024	1.54E+02	2.89E+02	4.32E+02	4.48E+02	5.78E+02	6.36E+02	9.23E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.81E+00	3.60E+01	6.54E+01	1.15E+02	2.19E+02	4.32E+02	8.94E+02
32	1.17E+01	3.68E+01	6.75E+01	1.12E+02	2.27E+02	4.22E+02	8.83E+02
64	1.96E+01	4.10E+01	6.86E+01	1.13E+02	2.20E+02	4.28E+02	9.35E+02
128	3.93E+01	6.63E+01	8.00E+01	1.20E+02	2.20E+02	4.32E+02	8.53E+02
256	8.14E+01	1.17E+02	1.35E+02	1.81E+02	2.61E+02	4.36E+02	9.19E+02
512	1.47E+02	1.72E+02	2.09E+02	2.39E+02	3.15E+02	4.89E+02	9.28E+02
1024	1.52E+02	3.28E+02	4.37E+02	5.21E+02	5.72E+02	6.23E+02	9.53E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.82E+00	3.73E+01	6.75E+01	1.17E+02	2.26E+02	4.44E+02	9.24E+02
32	1.16E+01	3.78E+01	6.99E+01	1.15E+02	2.34E+02	4.36E+02	8.98E+02
64	1.92E+01	4.15E+01	7.17E+01	1.16E+02	2.28E+02	4.39E+02	9.50E+02
128	3.92E+01	6.62E+01	8.16E+01	1.23E+02	2.27E+02	4.47E+02	8.76E+02
256	8.62E+01	1.18E+02	1.37E+02	1.84E+02	2.65E+02	4.44E+02	9.35E+02
512	1.55E+02	1.80E+02	2.13E+02	2.44E+02	3.40E+02	4.96E+02	9.52E+02
1024	1.51E+02	3.34E+02	4.37E+02	5.26E+02	5.65E+02	6.07E+02	9.75E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.83E+00	3.84E+01	6.93E+01	1.19E+02	2.33E+02	4.54E+02	1.01E+03
32	1.16E+01	3.86E+01	7.20E+01	1.17E+02	2.55E+02	4.47E+02	9.10E+02
64	1.89E+01	4.24E+01	7.33E+01	1.18E+02	2.35E+02	4.48E+02	9.63E+02
128	3.89E+01	6.62E+01	8.29E+01	1.25E+02	2.33E+02	4.60E+02	8.95E+02
256	9.04E+01	1.19E+02	1.38E+02	1.87E+02	2.70E+02	4.54E+02	9.48E+02
512	1.61E+02	1.87E+02	2.19E+02	2.77E+02	3.43E+02	5.02E+02	9.71E+02
1024	1.49E+02	3.38E+02	4.36E+02	5.31E+02	5.58E+02	6.01E+02	9.94E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.83E+00	3.95E+01	7.09E+01	1.20E+02	2.38E+02	4.62E+02	1.02E+03
32	1.15E+01	3.93E+01	7.98E+01	1.20E+02	2.58E+02	4.58E+02	9.22E+02
64	1.85E+01	4.31E+01	7.48E+01	1.21E+02	2.41E+02	4.57E+02	9.74E+02
128	3.82E+01	6.61E+01	8.39E+01	1.26E+02	2.38E+02	4.72E+02	9.12E+02
256	9.42E+01	1.19E+02	1.39E+02	1.89E+02	2.74E+02	4.62E+02	9.59E+02
512	1.67E+02	1.93E+02	2.25E+02	2.78E+02	3.46E+02	5.08E+02	9.88E+02
1024	1.55E+02	3.42E+02	4.35E+02	5.33E+02	5.51E+02	6.20E+02	1.01E+03

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.83E+00	4.06E+01	7.24E+01	1.21E+02	2.43E+02	4.71E+02	1.03E+03
32	1.16E+01	3.99E+01	8.11E+01	1.22E+02	2.61E+02	4.67E+02	9.43E+02
64	1.80E+01	4.37E+01	7.61E+01	1.23E+02	2.47E+02	4.65E+02	9.84E+02
128	3.74E+01	6.58E+01	8.48E+01	1.28E+02	2.42E+02	4.83E+02	9.29E+02
256	9.61E+01	1.20E+02	1.40E+02	1.90E+02	2.78E+02	4.69E+02	9.70E+02
512	1.73E+02	1.99E+02	2.45E+02	2.79E+02	3.48E+02	5.13E+02	1.00E+03
1024	1.94E+02	3.45E+02	4.32E+02	5.33E+02	5.43E+02	6.24E+02	1.02E+03

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.83E+00	4.13E+01	7.40E+01	1.24E+02	2.48E+02	4.79E+02	1.04E+03
32	1.22E+01	4.05E+01	8.20E+01	1.24E+02	2.63E+02	4.76E+02	9.62E+02
64	1.79E+01	4.43E+01	7.75E+01	1.25E+02	2.53E+02	4.74E+02	9.95E+02
128	3.64E+01	6.54E+01	8.62E+01	1.30E+02	2.47E+02	4.94E+02	9.45E+02
256	9.61E+01	1.21E+02	1.41E+02	1.92E+02	2.82E+02	4.77E+02	9.81E+02
512	1.80E+02	2.06E+02	2.46E+02	2.80E+02	3.51E+02	5.18E+02	1.02E+03
1024	1.32E+02	3.47E+02	4.99E+02	5.32E+02	5.37E+02	6.27E+02	1.04E+03

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.84E+00	4.18E+01	7.56E+01	1.27E+02	2.53E+02	4.87E+02	1.05E+03
32	1.22E+01	4.12E+01	8.29E+01	1.26E+02	2.66E+02	4.86E+02	9.82E+02
64	1.77E+01	4.48E+01	7.90E+01	1.27E+02	2.76E+02	4.82E+02	1.01E+03
128	3.67E+01	6.48E+01	8.75E+01	1.32E+02	2.51E+02	5.05E+02	9.62E+02
256	9.62E+01	1.22E+02	1.41E+02	1.94E+02	2.85E+02	4.85E+02	9.92E+02
512	1.87E+02	2.13E+02	2.46E+02	2.81E+02	3.53E+02	5.24E+02	1.03E+03
1024	1.31E+02	3.49E+02	4.99E+02	5.30E+02	5.28E+02	6.56E+02	1.05E+03

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.89E+00	4.24E+01	7.76E+01	1.30E+02	2.59E+02	4.98E+02	1.06E+03
32	1.23E+01	4.19E+01	8.39E+01	1.29E+02	2.69E+02	4.97E+02	1.00E+03
64	1.76E+01	4.54E+01	8.23E+01	1.29E+02	2.80E+02	4.93E+02	1.02E+03
128	4.25E+01	6.40E+01	8.90E+01	1.35E+02	2.56E+02	5.19E+02	9.83E+02
256	9.62E+01	1.22E+02	1.42E+02	1.95E+02	2.95E+02	4.94E+02	1.00E+03
512	1.95E+02	2.31E+02	2.46E+02	2.82E+02	3.56E+02	5.34E+02	1.05E+03
1024	1.30E+02	4.28E+02	4.99E+02	5.26E+02	5.16E+02	6.61E+02	1.07E+03

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.98E+00	4.35E+01	8.14E+01	1.36E+02	2.75E+02	5.18E+02	1.09E+03
32	1.17E+01	4.35E+01	8.59E+01	1.34E+02	2.76E+02	5.18E+02	1.05E+03
64	1.76E+01	4.66E+01	8.55E+01	1.34E+02	2.86E+02	5.15E+02	1.04E+03
128	4.07E+01	6.29E+01	9.11E+01	1.40E+02	2.65E+02	5.81E+02	1.02E+03
256	9.63E+01	1.24E+02	1.42E+02	1.99E+02	3.02E+02	5.16E+02	1.03E+03
512	2.19E+02	2.32E+02	2.47E+02	2.84E+02	3.62E+02	5.56E+02	1.07E+03
1024	1.27E+02	4.23E+02	4.95E+02	5.13E+02	4.84E+02	6.69E+02	1.10E+03

B.2 NAND gate (NAND2X0)

B.2.1 Falling propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8334	23.645	39.53	61.36	117.3	226.36	429.13
32	8.2114	25.161	40.484	62.719	117.58	233.31	429.25
64	6.5065	28.021	43.236	65.509	120.01	235.14	431.89
128	-6.1402	28.461	48.455	71.367	124.88	236.84	435.44
256	-16.154	28.226	56.289	85.688	138.21	241.56	451.07
512	-46.004	11.457	47.192	87.444	158.51	270.5	477.87
1024	-102.24	-82.859	-30.043	41.259	145.52	294.49	523.47

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.835	23.633	39.508	61.342	117.28	226.32	429.1
32	8.4237	25.154	40.472	62.703	117.56	233.28	429.22
64	6.5031	27.939	43.188	65.471	119.98	235.1	431.87
128	-6.2319	28.273	48.303	71.252	124.8	236.78	435.4
256	-16.934	28.048	55.994	85.541	138.12	241.49	451.02
512	-42.325	9.979	46.073	86.698	158.2	270.33	477.74
1024	-121.33	-85.972	-32.821	38.882	144.26	293.96	523.07

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8364	23.625	39.492	61.329	117.26	226.3	429.09
32	8.3837	25.15	40.464	62.692	117.54	233.26	429.21
64	6.3888	27.889	43.158	65.447	119.96	235.08	431.85
128	-6.2758	28.173	48.22	71.189	124.75	236.74	435.37
256	-2.6615	27.931	55.801	85.444	138.06	241.45	450.99
512	-48.995	9.1147	45.407	86.165	158.03	270.24	477.67
1024	-119.68	-87.914	-34.405	37.626	143.57	293.69	522.87

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8379	23.618	39.48	61.319	117.25	226.28	429.08
32	8.1953	25.146	40.457	62.683	117.53	233.25	429.19
64	6.1382	27.852	43.136	65.429	119.94	235.06	431.84
128	-6.3011	28.106	48.165	71.145	124.71	236.72	435.36
256	-4.0823	27.839	55.649	85.369	138.02	241.42	450.97
512	-49.96	8.4948	44.921	85.845	157.91	270.18	477.62
1024	-111.6	-89.375	-35.512	36.807	143.12	293.51	522.74

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8391	23.613	39.47	61.31	117.24	226.26	429.06
32	8.191	25.143	40.452	62.676	117.52	233.23	429.18
64	6.2172	27.822	43.118	65.414	119.93	235.05	431.82
128	-6.3164	28.057	48.124	71.112	124.69	236.69	435.34
256	-15.553	27.759	55.519	85.304	137.98	241.39	450.95
512	-45.344	8.0065	44.532	85.591	157.82	270.14	477.58
1024	-113.58	-90.572	-36.358	36.224	142.78	293.38	522.64

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8606	23.607	39.46	61.302	117.23	226.25	429.05
32	8.1872	25.14	40.447	62.669	117.52	233.22	429.17
64	6.1499	27.797	43.102	65.401	119.92	235.03	431.81
128	-2.7531	28.018	48.091	71.085	124.66	236.67	435.33
256	-6.4054	27.685	55.401	85.246	137.94	241.37	450.93
512	-46.12	7.5961	44.199	85.374	157.74	270.1	477.55
1024	-129.1	-91.612	-37.045	35.785	142.51	293.28	522.56

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8416	23.602	39.45	61.294	117.22	226.23	429.05
32	8.2821	25.137	40.442	62.662	117.51	233.21	429.16
64	6.0797	27.775	43.087	65.389	119.91	235.02	431.8
128	-3.055	27.986	48.063	71.062	124.64	236.66	435.31
256	-18.66	27.613	55.289	85.19	137.91	241.34	450.91
512	-52.249	7.231	43.897	85.177	157.67	270.06	477.52
1024	-117.14	-92.565	-37.633	35.442	142.28	293.2	522.5

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.843	23.597	39.44	61.286	117.21	226.22	429.04
32	8.1893	25.134	40.436	62.655	117.5	233.2	429.15
64	5.7843	27.754	43.074	65.377	119.9	235	431.79
128	-3.3445	27.957	48.038	71.041	124.63	236.64	435.3
256	-18.91	27.496	55.175	85.134	137.88	241.32	450.89
512	-52.993	6.8889	43.608	84.989	157.6	270.03	477.5
1024	-132.2	-93.482	-38.157	35.169	142.08	293.12	522.45

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8446	23.591	39.429	61.277	117.2	226.2	429.02
32	8.1917	25.131	40.43	62.647	117.49	233.18	429.14
64	5.701	27.733	43.059	65.363	119.88	234.99	431.78
128	-6.3223	27.93	48.014	71.021	124.61	236.62	435.29
256	-9.7641	27.334	55.05	85.072	137.84	241.3	450.88
512	-53.809	6.861	43.305	84.791	157.54	269.99	477.47
1024	-133.73	-94.442	-38.658	34.874	141.88	293.05	522.39

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	7.8694	23.579	39.408	61.26	117.18	226.17	429
32	8.2633	25.124	40.419	62.632	117.47	233.16	429.12
64	5.5589	27.699	43.034	65.34	119.86	234.96	431.76
128	-6.2972	27.891	47.977	70.988	124.57	236.59	435.26
256	-12.025	27.012	54.816	84.956	137.77	241.25	450.84
512	-55.369	6.1456	42.781	84.448	157.42	269.93	477.43
1024	-137.35	-96.075	-39.381	34.277	141.57	292.93	522.31

B.2.2 Rising propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.14E+01	3.56E+01	5.84E+01	9.91E+01	1.71E+02	3.50E+02	6.73E+02
32	1.57E+01	3.95E+01	6.44E+01	9.79E+01	1.84E+02	3.60E+02	6.53E+02
64	2.33E+01	4.82E+01	7.09E+01	1.04E+02	1.90E+02	3.76E+02	6.61E+02
128	2.85E+01	6.44E+01	8.74E+01	1.22E+02	2.05E+02	3.88E+02	6.86E+02
256	4.48E+01	8.16E+01	1.14E+02	1.49E+02	2.27E+02	3.92E+02	7.53E+02
512	1.02E+02	1.36E+02	1.70E+02	2.18E+02	2.98E+02	4.51E+02	7.92E+02
1024	1.59E+02	2.28E+02	2.70E+02	3.10E+02	4.22E+02	5.94E+02	9.14E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.17E+01	3.75E+01	6.06E+01	1.03E+02	1.80E+02	3.62E+02	7.05E+02
32	1.63E+01	4.12E+01	6.79E+01	1.02E+02	1.91E+02	3.77E+02	6.78E+02
64	2.42E+01	5.03E+01	7.42E+01	1.09E+02	1.96E+02	3.91E+02	6.86E+02
128	3.05E+01	6.67E+01	9.07E+01	1.26E+02	2.13E+02	4.05E+02	7.11E+02
256	6.04E+01	8.41E+01	1.18E+02	1.54E+02	2.35E+02	4.06E+02	7.82E+02
512	1.11E+02	1.48E+02	1.84E+02	2.24E+02	3.13E+02	4.66E+02	8.28E+02
1024	1.75E+02	2.18E+02	2.90E+02	3.30E+02	4.43E+02	6.18E+02	9.49E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.19E+01	3.87E+01	6.22E+01	1.06E+02	1.87E+02	3.71E+02	7.85E+02
32	1.67E+01	4.23E+01	7.01E+01	1.04E+02	1.96E+02	3.90E+02	6.96E+02
64	2.49E+01	5.17E+01	7.64E+01	1.13E+02	2.01E+02	4.02E+02	7.04E+02
128	3.23E+01	6.83E+01	9.31E+01	1.29E+02	2.18E+02	4.17E+02	7.28E+02
256	6.06E+01	8.61E+01	1.20E+02	1.57E+02	2.41E+02	4.16E+02	8.02E+02
512	1.19E+02	1.59E+02	2.08E+02	2.36E+02	3.18E+02	4.77E+02	8.53E+02
1024	1.87E+02	1.97E+02	3.04E+02	3.43E+02	4.58E+02	6.46E+02	9.72E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.21E+01	3.97E+01	6.35E+01	1.08E+02	1.92E+02	3.78E+02	8.04E+02
32	1.70E+01	4.32E+01	7.19E+01	1.06E+02	2.00E+02	3.99E+02	7.11E+02
64	2.54E+01	5.28E+01	7.82E+01	1.17E+02	2.05E+02	4.10E+02	7.27E+02
128	3.34E+01	6.97E+01	9.49E+01	1.32E+02	2.22E+02	4.26E+02	7.43E+02
256	6.07E+01	8.79E+01	1.22E+02	1.60E+02	2.46E+02	4.24E+02	8.18E+02
512	1.26E+02	1.70E+02	2.09E+02	2.38E+02	3.23E+02	4.87E+02	8.72E+02
1024	1.95E+02	2.09E+02	3.16E+02	3.54E+02	4.69E+02	6.58E+02	9.92E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.23E+01	4.05E+01	6.51E+01	1.10E+02	1.97E+02	3.84E+02	8.20E+02
32	1.73E+01	4.41E+01	7.35E+01	1.08E+02	2.04E+02	4.08E+02	7.30E+02
64	2.59E+01	5.37E+01	7.98E+01	1.21E+02	2.09E+02	4.18E+02	7.47E+02
128	3.44E+01	7.08E+01	9.66E+01	1.34E+02	2.26E+02	4.35E+02	7.56E+02
256	6.08E+01	8.95E+01	1.24E+02	1.62E+02	2.50E+02	4.32E+02	8.32E+02
512	1.32E+02	1.94E+02	2.10E+02	2.40E+02	3.27E+02	4.97E+02	8.89E+02
1024	2.01E+02	2.18E+02	3.25E+02	3.64E+02	4.78E+02	6.68E+02	1.01E+03

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.25E+01	4.13E+01	6.66E+01	1.12E+02	2.02E+02	3.91E+02	8.35E+02
32	1.75E+01	4.48E+01	7.49E+01	1.10E+02	2.07E+02	4.15E+02	7.49E+02
64	2.63E+01	5.46E+01	8.12E+01	1.24E+02	2.12E+02	4.25E+02	7.66E+02
128	3.53E+01	7.18E+01	9.84E+01	1.36E+02	2.30E+02	4.42E+02	7.69E+02
256	6.09E+01	9.12E+01	1.26E+02	1.65E+02	2.54E+02	4.39E+02	8.45E+02
512	1.39E+02	1.95E+02	2.11E+02	2.42E+02	3.31E+02	5.06E+02	9.04E+02
1024	2.08E+02	2.32E+02	3.34E+02	3.72E+02	4.87E+02	6.77E+02	1.03E+03

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.26E+01	4.21E+01	6.81E+01	1.13E+02	2.07E+02	3.97E+02	8.50E+02
32	1.78E+01	4.56E+01	7.63E+01	1.12E+02	2.11E+02	4.23E+02	7.67E+02
64	2.68E+01	5.54E+01	8.26E+01	1.26E+02	2.16E+02	4.32E+02	7.85E+02
128	3.62E+01	7.28E+01	1.00E+02	1.38E+02	2.34E+02	4.49E+02	7.84E+02
256	6.10E+01	9.29E+01	1.28E+02	1.67E+02	2.59E+02	4.45E+02	8.58E+02
512	1.45E+02	1.95E+02	2.12E+02	2.43E+02	3.35E+02	5.15E+02	9.19E+02
1024	2.13E+02	2.41E+02	3.41E+02	3.81E+02	4.95E+02	6.86E+02	1.04E+03

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.28E+01	4.28E+01	6.97E+01	1.15E+02	2.12E+02	4.03E+02	8.65E+02
32	1.81E+01	4.63E+01	7.77E+01	1.14E+02	2.14E+02	4.31E+02	7.87E+02
64	2.72E+01	5.62E+01	8.40E+01	1.29E+02	2.19E+02	4.39E+02	8.05E+02
128	3.71E+01	7.38E+01	1.02E+02	1.40E+02	2.37E+02	4.57E+02	8.04E+02
256	6.12E+01	9.48E+01	1.29E+02	1.69E+02	2.65E+02	4.53E+02	8.72E+02
512	1.52E+02	1.96E+02	2.13E+02	2.45E+02	3.39E+02	5.24E+02	9.34E+02
1024	2.16E+02	2.50E+02	3.49E+02	3.89E+02	5.03E+02	6.96E+02	1.06E+03

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.30E+01	4.37E+01	7.16E+01	1.17E+02	2.17E+02	4.11E+02	8.81E+02
32	1.83E+01	4.72E+01	7.93E+01	1.16E+02	2.18E+02	4.39E+02	8.10E+02
64	2.78E+01	5.72E+01	8.56E+01	1.32E+02	2.23E+02	4.47E+02	8.28E+02
128	3.80E+01	7.50E+01	1.04E+02	1.44E+02	2.42E+02	4.65E+02	8.27E+02
256	6.13E+01	9.70E+01	1.31E+02	1.72E+02	2.71E+02	4.61E+02	8.87E+02
512	1.61E+02	1.96E+02	2.14E+02	2.48E+02	3.43E+02	5.34E+02	9.50E+02
1024	2.14E+02	2.61E+02	3.58E+02	3.99E+02	5.13E+02	7.07E+02	1.08E+03

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.34E+01	4.54E+01	7.86E+01	1.22E+02	2.43E+02	4.26E+02	9.14E+02
32	1.88E+01	4.90E+01	8.23E+01	1.21E+02	2.26E+02	4.56E+02	8.52E+02
64	2.91E+01	5.90E+01	8.87E+01	1.38E+02	2.32E+02	4.62E+02	8.73E+02
128	3.97E+01	7.78E+01	1.08E+02	1.51E+02	2.50E+02	4.81E+02	8.74E+02
256	6.15E+01	1.01E+02	1.36E+02	1.77E+02	2.83E+02	4.81E+02	9.18E+02
512	1.18E+02	1.97E+02	2.16E+02	2.53E+02	3.52E+02	5.54E+02	9.82E+02
1024	2.08E+02	2.81E+02	3.76E+02	4.16E+02	5.31E+02	7.28E+02	1.11E+03

B.2.3 Falling transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.80E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.16E+01	4.45E+01	7.24E+01	1.17E+02	2.30E+02	4.74E+02	9.48E+02
128	3.88E+01	6.35E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.64E+01	8.91E+01	1.14E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	9.12E+01	1.28E+02	1.67E+02	2.10E+02	2.77E+02	4.78E+02	9.10E+02
1024	2.12E+02	2.67E+02	2.96E+02	3.33E+02	4.33E+02	5.98E+02	9.19E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.80E+02	9.14E+02
32	1.41E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.05E+01	4.45E+01	7.24E+01	1.17E+02	2.30E+02	4.74E+02	9.48E+02
128	3.82E+01	6.37E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.55E+01	8.86E+01	1.14E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.19E+01	1.28E+02	1.67E+02	2.10E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.87E+02	2.64E+02	2.93E+02	3.32E+02	4.33E+02	5.98E+02	9.19E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.80E+02	9.14E+02
32	1.39E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.05E+01	4.45E+01	7.24E+01	1.17E+02	2.30E+02	4.74E+02	9.48E+02
128	3.78E+01	6.38E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.53E+01	8.87E+01	1.14E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.69E+01	1.27E+02	1.67E+02	2.10E+02	2.78E+02	4.78E+02	9.10E+02
1024	2.34E+02	2.61E+02	2.92E+02	3.31E+02	4.33E+02	5.99E+02	9.19E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.03E+01	4.45E+01	7.24E+01	1.17E+02	2.30E+02	4.74E+02	9.48E+02
128	3.75E+01	6.39E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.37E+01	8.88E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.57E+01	1.27E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	2.03E+02	2.60E+02	2.91E+02	3.30E+02	4.33E+02	5.99E+02	9.19E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.04E+01	4.45E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.72E+01	6.40E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.16E+01	8.88E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	7.84E+01	1.26E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	2.01E+02	2.58E+02	2.90E+02	3.30E+02	4.33E+02	5.99E+02	9.20E+02

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.05E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.04E+01	4.45E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.63E+01	6.40E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.13E+01	8.89E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	7.76E+01	1.26E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.79E+02	2.57E+02	2.89E+02	3.29E+02	4.33E+02	5.99E+02	9.20E+02

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.04E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.36E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.04E+01	4.45E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.63E+01	6.41E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.38E+01	8.90E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.30E+01	1.26E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.97E+02	2.56E+02	2.88E+02	3.29E+02	4.33E+02	5.99E+02	9.20E+02

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.04E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.04E+01	4.46E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.63E+01	6.41E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	5.36E+01	8.91E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.23E+01	1.25E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.76E+02	2.56E+02	2.87E+02	3.29E+02	4.33E+02	5.99E+02	9.20E+02

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.04E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.04E+01	4.46E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.62E+01	6.41E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	4.86E+01	8.92E+01	1.13E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.14E+01	1.25E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.75E+02	2.55E+02	2.86E+02	3.29E+02	4.33E+02	5.99E+02	9.20E+02

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	9.03E+00	4.00E+01	7.46E+01	1.19E+02	2.34E+02	4.81E+02	9.14E+02
32	1.35E+01	4.06E+01	7.18E+01	1.18E+02	2.31E+02	4.74E+02	9.29E+02
64	2.05E+01	4.46E+01	7.25E+01	1.16E+02	2.30E+02	4.74E+02	9.48E+02
128	3.57E+01	6.42E+01	8.28E+01	1.21E+02	2.25E+02	4.60E+02	9.70E+02
256	4.77E+01	8.94E+01	1.12E+02	1.48E+02	2.43E+02	4.60E+02	9.29E+02
512	8.00E+01	1.25E+02	1.67E+02	2.08E+02	2.78E+02	4.78E+02	9.10E+02
1024	1.71E+02	2.53E+02	2.84E+02	3.28E+02	4.33E+02	5.99E+02	9.20E+02

B.2.4 Rising transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.10E+01	4.63E+01	8.18E+01	1.38E+02	2.67E+02	5.22E+02	1.08E+03
32	1.36E+01	4.52E+01	8.64E+01	1.34E+02	2.63E+02	5.62E+02	1.03E+03
64	2.37E+01	4.95E+01	8.38E+01	1.42E+02	2.61E+02	5.49E+02	1.03E+03
128	4.27E+01	6.97E+01	9.37E+01	1.42E+02	2.65E+02	5.59E+02	1.02E+03
256	8.20E+01	1.23E+02	1.44E+02	2.00E+02	3.05E+02	5.12E+02	1.03E+03
512	1.40E+02	1.84E+02	2.21E+02	2.56E+02	3.43E+02	5.55E+02	1.08E+03
1024	1.68E+02	3.00E+02	4.46E+02	5.33E+02	5.79E+02	6.59E+02	1.11E+03

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.12E+01	4.81E+01	8.57E+01	1.42E+02	2.80E+02	5.43E+02	1.13E+03
32	1.34E+01	4.69E+01	8.95E+01	1.40E+02	2.74E+02	5.77E+02	1.07E+03
64	2.25E+01	5.12E+01	8.73E+01	1.49E+02	2.71E+02	5.61E+02	1.08E+03
128	4.30E+01	6.97E+01	9.63E+01	1.47E+02	2.74E+02	5.73E+02	1.06E+03
256	9.75E+01	1.24E+02	1.45E+02	2.03E+02	3.12E+02	5.37E+02	1.07E+03
512	1.49E+02	1.91E+02	2.30E+02	2.63E+02	3.73E+02	5.76E+02	1.11E+03
1024	1.65E+02	3.25E+02	4.15E+02	5.20E+02	5.71E+02	6.77E+02	1.14E+03

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.14E+01	4.89E+01	8.85E+01	1.46E+02	2.90E+02	5.57E+02	1.22E+03
32	1.38E+01	4.81E+01	9.11E+01	1.44E+02	2.82E+02	5.87E+02	1.11E+03
64	2.19E+01	5.23E+01	8.96E+01	1.55E+02	2.78E+02	5.70E+02	1.11E+03
128	4.32E+01	6.94E+01	1.01E+02	1.51E+02	2.81E+02	5.81E+02	1.10E+03
256	9.75E+01	1.25E+02	1.47E+02	2.06E+02	3.17E+02	5.54E+02	1.11E+03
512	1.57E+02	2.01E+02	2.57E+02	2.94E+02	3.77E+02	5.92E+02	1.13E+03
1024	1.62E+02	3.51E+02	3.85E+02	5.06E+02	5.60E+02	7.07E+02	1.17E+03

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.15E+01	4.97E+01	9.10E+01	1.50E+02	2.98E+02	5.69E+02	1.24E+03
32	1.37E+01	4.91E+01	9.24E+01	1.48E+02	2.89E+02	5.96E+02	1.14E+03
64	2.13E+01	5.31E+01	9.14E+01	1.60E+02	2.85E+02	5.77E+02	1.15E+03
128	4.32E+01	6.88E+01	1.02E+02	1.54E+02	2.86E+02	5.89E+02	1.13E+03
256	9.75E+01	1.26E+02	1.48E+02	2.08E+02	3.21E+02	5.68E+02	1.14E+03
512	1.63E+02	2.12E+02	2.58E+02	2.95E+02	3.79E+02	6.04E+02	1.16E+03
1024	1.61E+02	3.56E+02	3.50E+02	4.90E+02	5.48E+02	7.15E+02	1.19E+03

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.16E+01	5.03E+01	9.32E+01	1.54E+02	3.05E+02	5.80E+02	1.25E+03
32	1.35E+01	5.00E+01	9.36E+01	1.52E+02	2.94E+02	6.03E+02	1.17E+03
64	2.09E+01	5.38E+01	9.30E+01	1.63E+02	2.92E+02	5.83E+02	1.17E+03
128	4.32E+01	6.81E+01	1.02E+02	1.61E+02	2.91E+02	5.95E+02	1.16E+03
256	9.76E+01	1.27E+02	1.48E+02	2.10E+02	3.25E+02	5.81E+02	1.16E+03
512	1.70E+02	2.36E+02	2.59E+02	2.96E+02	3.82E+02	6.16E+02	1.18E+03
1024	1.61E+02	3.57E+02	3.21E+02	4.74E+02	5.37E+02	7.27E+02	1.22E+03

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.18E+01	5.13E+01	9.54E+01	1.57E+02	3.12E+02	5.91E+02	1.27E+03
32	1.33E+01	5.08E+01	9.47E+01	1.55E+02	3.00E+02	6.10E+02	1.19E+03
64	2.10E+01	5.45E+01	9.45E+01	1.65E+02	2.98E+02	5.95E+02	1.20E+03
128	4.31E+01	6.76E+01	1.04E+02	1.64E+02	2.96E+02	6.01E+02	1.18E+03
256	9.77E+01	1.27E+02	1.49E+02	2.12E+02	3.28E+02	5.93E+02	1.18E+03
512	1.76E+02	2.36E+02	2.60E+02	2.97E+02	3.86E+02	6.27E+02	1.21E+03
1024	1.60E+02	4.28E+02	3.24E+02	4.56E+02	5.26E+02	7.38E+02	1.24E+03

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.20E+01	5.24E+01	9.75E+01	1.60E+02	3.18E+02	6.01E+02	1.28E+03
32	1.32E+01	5.16E+01	9.58E+01	1.58E+02	3.05E+02	6.17E+02	1.22E+03
64	2.11E+01	5.51E+01	9.59E+01	1.66E+02	3.04E+02	6.08E+02	1.22E+03
128	4.31E+01	6.81E+01	1.05E+02	1.67E+02	3.02E+02	6.06E+02	1.21E+03
256	9.77E+01	1.28E+02	1.49E+02	2.15E+02	3.31E+02	6.05E+02	1.20E+03
512	1.82E+02	2.36E+02	2.61E+02	2.98E+02	3.91E+02	6.37E+02	1.23E+03
1024	1.58E+02	4.25E+02	3.27E+02	4.35E+02	5.12E+02	7.48E+02	1.26E+03

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.22E+01	5.36E+01	9.98E+01	1.63E+02	3.27E+02	6.12E+02	1.29E+03
32	1.32E+01	5.26E+01	9.69E+01	1.62E+02	3.10E+02	6.24E+02	1.24E+03
64	2.12E+01	5.58E+01	9.74E+01	1.68E+02	3.11E+02	6.21E+02	1.25E+03
128	4.34E+01	6.87E+01	1.07E+02	1.73E+02	3.08E+02	6.16E+02	1.23E+03
256	9.78E+01	1.29E+02	1.50E+02	2.17E+02	3.35E+02	6.18E+02	1.23E+03
512	1.90E+02	2.36E+02	2.62E+02	2.99E+02	3.96E+02	6.49E+02	1.25E+03
1024	1.55E+02	4.22E+02	3.31E+02	4.11E+02	4.95E+02	7.58E+02	1.28E+03

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.23E+01	5.49E+01	1.02E+02	1.66E+02	3.36E+02	6.25E+02	1.31E+03
32	1.34E+01	5.40E+01	9.94E+01	1.66E+02	3.17E+02	6.32E+02	1.27E+03
64	2.18E+01	5.68E+01	9.92E+01	1.70E+02	3.19E+02	6.35E+02	1.28E+03
128	5.10E+01	6.92E+01	1.09E+02	1.77E+02	3.16E+02	6.32E+02	1.26E+03
256	9.79E+01	1.29E+02	1.50E+02	2.20E+02	3.38E+02	6.34E+02	1.25E+03
512	1.98E+02	2.36E+02	2.63E+02	3.00E+02	4.02E+02	6.62E+02	1.27E+03
1024	1.99E+02	4.88E+02	3.35E+02	3.80E+02	5.01E+02	7.68E+02	1.30E+03

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.26E+01	5.73E+01	1.18E+02	1.73E+02	3.76E+02	6.52E+02	1.34E+03
32	1.39E+01	5.67E+01	1.04E+02	1.74E+02	3.30E+02	6.58E+02	1.35E+03
64	2.23E+01	5.91E+01	1.03E+02	1.74E+02	3.34E+02	6.62E+02	1.35E+03
128	5.19E+01	7.27E+01	1.12E+02	1.83E+02	3.32E+02	6.61E+02	1.31E+03
256	9.81E+01	1.30E+02	1.76E+02	2.26E+02	3.49E+02	6.66E+02	1.30E+03
512	7.77E+01	2.37E+02	2.66E+02	3.03E+02	4.13E+02	6.87E+02	1.32E+03
1024	1.41E+02	4.85E+02	3.46E+02	3.83E+02	5.03E+02	7.89E+02	1.35E+03

B.3 NOR gate (NOR2X0)

B.3.1 Falling propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1683	20.276	31.727	48.257	87.854	167.39	327.81
32	9.0674	22.484	34.001	50.485	90.463	167.38	321.13
64	10.018	25.566	37.256	53.795	92.476	169.54	323.67
128	6.7771	28.332	42.768	60.868	100.05	177.21	334.21
256	0.81039	19.979	47.336	72.211	115.77	193.33	347.7
512	-33.066	3.8689	38.454	72.087	132.58	223.84	377.53
1024	-41.238	-70.616	-49.403	9.6085	111.07	240.05	430.24

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1656	20.184	31.672	48.191	87.744	166.79	327.69
32	9.0567	22.468	33.965	50.445	90.607	166.79	321.22
64	9.245	25.462	37.188	53.725	92.414	169.44	324.84
128	5.8319	27.943	42.477	60.688	99.9	177.03	333.8
256	0.54566	19.758	47.055	72.084	115.75	193.3	347.63
512	-34.666	2.3662	37.188	71.177	132.13	223.7	377.46
1024	-41.515	-74.116	-52.028	6.9874	109.28	239.28	429.62

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1392	20.149	31.635	48.147	87.672	166.43	327.24
32	9.0479	22.456	33.944	50.412	90.59	166.32	321.31
64	8.8144	25.387	37.144	53.678	92.371	169.38	325.48
128	5.2113	27.673	42.292	60.555	99.81	176.91	333.53
256	0.36061	19.626	47.063	71.955	115.74	193.29	347.54
512	-35.749	1.4826	36.439	70.666	131.89	223.63	377.42
1024	-41.672	-76.418	-53.605	5.5248	108.3	238.88	429.3

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.137	20.125	31.605	48.113	87.617	166.17	326.74
32	9.0395	22.446	33.925	50.385	90.548	166.34	321.41
64	8.514	25.325	37.108	53.641	92.336	169.37	325.91
128	4.731	27.456	42.284	60.482	99.744	176.82	333.32
256	0.20577	19.528	47.055	71.9	115.73	193.28	347.48
512	-36.64	0.84709	35.896	70.267	131.72	223.58	377.39
1024	-41.783	-78.224	-54.753	4.5241	107.63	238.62	429.1

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1352	20.103	31.58	48.082	87.571	165.97	326.23
32	9.031	22.437	33.908	50.363	90.515	166.38	321.72
64	8.2825	25.282	37.078	53.609	92.306	169.41	326.23
128	4.3305	27.272	42.143	60.425	99.691	176.75	333.14
256	0.064904	19.448	47.039	71.856	115.73	193.27	347.43
512	-37.396	0.34629	35.465	69.984	131.59	223.54	377.37
1024	-44.458	-79.759	-55.667	3.7752	107.12	238.43	428.95

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1337	20.086	31.557	48.054	87.546	165.79	325.73
32	9.022	22.428	33.893	50.342	90.486	166.33	321.92
64	8.0921	25.248	37.052	53.58	92.278	169.45	326.47
128	3.9791	27.106	42.027	60.379	99.646	176.69	332.98
256	-0.07145	19.377	46.941	71.819	115.72	193.27	347.38
512	-38.091	-0.07426	35.098	69.748	131.48	223.51	377.36
1024	-47.878	-81.143	-56.439	3.1804	106.72	238.29	428.83

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1324	20.07	31.534	48.027	87.506	165.81	325.23
32	9.0119	22.419	33.901	50.322	90.459	166.32	322.12
64	7.927	25.217	37.027	53.553	92.242	169.49	326.68
128	3.649	26.952	41.927	60.339	99.606	176.63	332.8
256	-0.2114	19.312	46.888	71.785	115.72	193.26	347.34
512	-38.766	0.071763	34.771	69.541	131.39	223.48	377.34
1024	-51.13	-82.458	-57.126	2.6855	106.37	238.17	428.73

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1312	20.053	31.511	47.999	87.458	165.64	325.17
32	8.9999	22.41	33.884	50.3	90.433	166.23	322.59
64	7.7772	25.186	37.002	53.526	92.214	169.52	326.85
128	3.3044	26.8	41.836	60.302	99.569	176.58	332.53
256	-0.36418	19.248	46.828	71.754	115.72	193.31	347.3
512	-39.458	-0.26684	34.462	69.348	131.3	223.46	377.33
1024	-64.11	-83.776	-57.767	2.2636	106.06	238.06	428.65

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1299	20.034	31.485	47.969	87.415	171.49	318.25
32	8.9843	22.399	33.863	50.276	90.406	166.17	325.33
64	7.6315	25.219	36.975	53.496	92.184	169.2	327.01
128	2.9393	26.64	41.809	60.265	99.53	176.52	332.27
256	-0.54968	19.178	46.757	71.721	115.71	193.29	347.26
512	-40.235	-0.608	34.142	69.154	131.21	223.43	377.31
1024	-67.665	-85.222	-58.415	1.8696	105.77	237.96	428.57

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.1278	19.998	31.435	47.91	87.335	171.29	317.96
32	8.947	22.379	33.823	50.226	90.356	166.06	324.73
64	7.4151	25.092	36.927	53.442	92.127	169.13	324.35
128	2.3022	26.363	41.69	60.207	99.465	176.45	331.82
256	-0.95879	19.05	46.877	71.667	115.71	193.28	347.19
512	-41.732	-1.1568	33.601	68.834	131.06	223.48	377.3
1024	-74.268	-87.894	-59.453	1.3545	105.33	237.8	428.45

B.3.2 Rising propagation delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.17E+00	2.03E+01	3.17E+01	4.83E+01	8.79E+01	1.67E+02	3.28E+02
32	9.07E+00	2.25E+01	3.40E+01	5.05E+01	9.05E+01	1.67E+02	3.21E+02
64	1.00E+01	2.56E+01	3.73E+01	5.38E+01	9.25E+01	1.70E+02	3.24E+02
128	6.78E+00	2.83E+01	4.28E+01	6.09E+01	1.00E+02	1.77E+02	3.34E+02
256	8.10E-01	2.00E+01	4.73E+01	7.22E+01	1.16E+02	1.93E+02	3.48E+02
512	-3.31E+01	3.87E+00	3.85E+01	7.21E+01	1.33E+02	2.24E+02	3.78E+02
1024	-4.12E+01	-7.06E+01	-4.94E+01	9.61E+00	1.11E+02	2.40E+02	4.30E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.17E+00	2.02E+01	3.17E+01	4.82E+01	8.77E+01	1.67E+02	3.28E+02
32	9.06E+00	2.25E+01	3.40E+01	5.04E+01	9.06E+01	1.67E+02	3.21E+02
64	9.25E+00	2.55E+01	3.72E+01	5.37E+01	9.24E+01	1.69E+02	3.25E+02
128	5.83E+00	2.79E+01	4.25E+01	6.07E+01	9.99E+01	1.77E+02	3.34E+02
256	5.46E-01	1.98E+01	4.71E+01	7.21E+01	1.16E+02	1.93E+02	3.48E+02
512	-3.47E+01	2.37E+00	3.72E+01	7.12E+01	1.32E+02	2.24E+02	3.77E+02
1024	-4.15E+01	-7.41E+01	-5.20E+01	6.99E+00	1.09E+02	2.39E+02	4.30E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.14E+00	2.01E+01	3.16E+01	4.81E+01	8.77E+01	1.66E+02	3.27E+02
32	9.05E+00	2.25E+01	3.39E+01	5.04E+01	9.06E+01	1.66E+02	3.21E+02
64	8.81E+00	2.54E+01	3.71E+01	5.37E+01	9.24E+01	1.69E+02	3.25E+02
128	5.21E+00	2.77E+01	4.23E+01	6.06E+01	9.98E+01	1.77E+02	3.34E+02
256	3.61E-01	1.96E+01	4.71E+01	7.20E+01	1.16E+02	1.93E+02	3.48E+02
512	-3.57E+01	1.48E+00	3.64E+01	7.07E+01	1.32E+02	2.24E+02	3.77E+02
1024	-4.17E+01	-7.64E+01	-5.36E+01	5.52E+00	1.08E+02	2.39E+02	4.29E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.14E+00	2.01E+01	3.16E+01	4.81E+01	8.76E+01	1.66E+02	3.27E+02
32	9.04E+00	2.24E+01	3.39E+01	5.04E+01	9.05E+01	1.66E+02	3.21E+02
64	8.51E+00	2.53E+01	3.71E+01	5.36E+01	9.23E+01	1.69E+02	3.26E+02
128	4.73E+00	2.75E+01	4.23E+01	6.05E+01	9.97E+01	1.77E+02	3.33E+02
256	2.06E-01	1.95E+01	4.71E+01	7.19E+01	1.16E+02	1.93E+02	3.47E+02
512	-3.66E+01	8.47E-01	3.59E+01	7.03E+01	1.32E+02	2.24E+02	3.77E+02
1024	-4.18E+01	-7.82E+01	-5.48E+01	4.52E+00	1.08E+02	2.39E+02	4.29E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.14E+00	2.01E+01	3.16E+01	4.81E+01	8.76E+01	1.66E+02	3.26E+02
32	9.03E+00	2.24E+01	3.39E+01	5.04E+01	9.05E+01	1.66E+02	3.22E+02
64	8.28E+00	2.53E+01	3.71E+01	5.36E+01	9.23E+01	1.69E+02	3.26E+02
128	4.33E+00	2.73E+01	4.21E+01	6.04E+01	9.97E+01	1.77E+02	3.33E+02
256	6.49E-02	1.94E+01	4.70E+01	7.19E+01	1.16E+02	1.93E+02	3.47E+02
512	-3.74E+01	3.46E-01	3.55E+01	7.00E+01	1.32E+02	2.24E+02	3.77E+02
1024	-4.45E+01	-7.98E+01	-5.57E+01	3.78E+00	1.07E+02	2.38E+02	4.29E+02

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.13E+00	2.01E+01	3.16E+01	4.81E+01	8.75E+01	1.66E+02	3.26E+02
32	9.02E+00	2.24E+01	3.39E+01	5.03E+01	9.05E+01	1.66E+02	3.22E+02
64	8.09E+00	2.52E+01	3.71E+01	5.36E+01	9.23E+01	1.69E+02	3.26E+02
128	3.98E+00	2.71E+01	4.20E+01	6.04E+01	9.96E+01	1.77E+02	3.33E+02
256	-7.14E-02	1.94E+01	4.69E+01	7.18E+01	1.16E+02	1.93E+02	3.47E+02
512	-3.81E+01	-7.43E-02	3.51E+01	6.97E+01	1.31E+02	2.24E+02	3.77E+02
1024	-4.79E+01	-8.11E+01	-5.64E+01	3.18E+00	1.07E+02	2.38E+02	4.29E+02

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.13E+00	2.01E+01	3.15E+01	4.80E+01	8.75E+01	1.66E+02	3.25E+02
32	9.01E+00	2.24E+01	3.39E+01	5.03E+01	9.05E+01	1.66E+02	3.22E+02
64	7.93E+00	2.52E+01	3.70E+01	5.36E+01	9.22E+01	1.69E+02	3.27E+02
128	3.65E+00	2.70E+01	4.19E+01	6.03E+01	9.96E+01	1.77E+02	3.33E+02
256	-2.11E-01	1.93E+01	4.69E+01	7.18E+01	1.16E+02	1.93E+02	3.47E+02
512	-3.88E+01	7.18E-02	3.48E+01	6.95E+01	1.31E+02	2.23E+02	3.77E+02
1024	-5.11E+01	-8.25E+01	-5.71E+01	2.69E+00	1.06E+02	2.38E+02	4.29E+02

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.13E+00	2.01E+01	3.15E+01	4.80E+01	8.75E+01	1.66E+02	3.25E+02
32	9.00E+00	2.24E+01	3.39E+01	5.03E+01	9.04E+01	1.66E+02	3.23E+02
64	7.78E+00	2.52E+01	3.70E+01	5.35E+01	9.22E+01	1.70E+02	3.27E+02
128	3.30E+00	2.68E+01	4.18E+01	6.03E+01	9.96E+01	1.77E+02	3.33E+02
256	-3.64E-01	1.92E+01	4.68E+01	7.18E+01	1.16E+02	1.93E+02	3.47E+02
512	-3.95E+01	-2.67E-01	3.45E+01	6.93E+01	1.31E+02	2.23E+02	3.77E+02
1024	-6.41E+01	-8.38E+01	-5.78E+01	2.26E+00	1.06E+02	2.38E+02	4.29E+02

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.13E+00	2.00E+01	3.15E+01	4.80E+01	8.74E+01	1.71E+02	3.18E+02
32	8.98E+00	2.24E+01	3.39E+01	5.03E+01	9.04E+01	1.66E+02	3.25E+02
64	7.63E+00	2.52E+01	3.70E+01	5.35E+01	9.22E+01	1.69E+02	3.27E+02
128	2.94E+00	2.66E+01	4.18E+01	6.03E+01	9.95E+01	1.77E+02	3.32E+02
256	-5.50E-01	1.92E+01	4.68E+01	7.17E+01	1.16E+02	1.93E+02	3.47E+02
512	-4.02E+01	-6.08E-01	3.41E+01	6.92E+01	1.31E+02	2.23E+02	3.77E+02
1024	-6.77E+01	-8.52E+01	-5.84E+01	1.87E+00	1.06E+02	2.38E+02	4.29E+02

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	8.13E+00	2.00E+01	3.14E+01	4.79E+01	8.73E+01	1.71E+02	3.18E+02
32	8.95E+00	2.24E+01	3.38E+01	5.02E+01	9.04E+01	1.66E+02	3.25E+02
64	7.42E+00	2.51E+01	3.69E+01	5.34E+01	9.21E+01	1.69E+02	3.24E+02
128	2.30E+00	2.64E+01	4.17E+01	6.02E+01	9.95E+01	1.76E+02	3.32E+02
256	-9.59E-01	1.91E+01	4.69E+01	7.17E+01	1.16E+02	1.93E+02	3.47E+02
512	-4.17E+01	-1.16E+00	3.36E+01	6.88E+01	1.31E+02	2.23E+02	3.77E+02
1024	-7.43E+01	-8.79E+01	-5.95E+01	1.35E+00	1.05E+02	2.38E+02	4.28E+02

B.3.3 Falling transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.41E+00	2.83E+01	5.04E+01	9.12E+01	1.69E+02	3.24E+02	6.59E+02
32	9.77E+00	2.85E+01	5.07E+01	8.53E+01	1.63E+02	3.28E+02	6.88E+02
64	2.03E+01	3.28E+01	5.20E+01	8.29E+01	1.61E+02	3.19E+02	6.49E+02
128	2.71E+01	5.06E+01	6.35E+01	8.96E+01	1.65E+02	3.33E+02	6.66E+02
256	5.21E+01	8.80E+01	9.90E+01	1.20E+02	1.73E+02	3.23E+02	6.89E+02
512	8.53E+01	1.27E+02	1.48E+02	1.82E+02	2.42E+02	3.42E+02	6.42E+02
1024	2.68E+02	2.46E+02	2.98E+02	3.12E+02	3.89E+02	5.18E+02	6.93E+02

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.32E+00	2.83E+01	5.04E+01	9.09E+01	1.69E+02	3.25E+02	6.50E+02
32	9.73E+00	2.84E+01	5.07E+01	8.52E+01	1.64E+02	3.30E+02	7.00E+02
64	1.95E+01	3.29E+01	5.19E+01	8.31E+01	1.61E+02	3.17E+02	6.46E+02
128	2.68E+01	5.07E+01	6.35E+01	8.95E+01	1.64E+02	3.33E+02	6.69E+02
256	5.22E+01	8.79E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.82E+02
512	8.37E+01	1.26E+02	1.47E+02	1.82E+02	2.42E+02	3.42E+02	6.38E+02
1024	2.68E+02	2.41E+02	2.95E+02	3.10E+02	3.88E+02	5.18E+02	6.93E+02

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.36E+00	2.83E+01	5.03E+01	9.06E+01	1.70E+02	3.26E+02	6.44E+02
32	9.72E+00	2.84E+01	5.07E+01	8.52E+01	1.64E+02	3.31E+02	7.09E+02
64	1.94E+01	3.29E+01	5.19E+01	8.30E+01	1.61E+02	3.18E+02	6.43E+02
128	2.66E+01	5.08E+01	6.35E+01	8.94E+01	1.63E+02	3.29E+02	6.71E+02
256	5.23E+01	8.78E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.77E+02
512	8.26E+01	1.25E+02	1.47E+02	1.82E+02	2.42E+02	3.42E+02	6.39E+02
1024	2.68E+02	2.39E+02	2.93E+02	3.09E+02	3.88E+02	5.19E+02	6.93E+02

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.34E+00	2.83E+01	5.03E+01	9.06E+01	1.70E+02	3.26E+02	6.39E+02
32	1.02E+01	2.84E+01	5.07E+01	8.52E+01	1.64E+02	3.32E+02	7.12E+02
64	1.93E+01	3.29E+01	5.18E+01	8.31E+01	1.61E+02	3.18E+02	6.41E+02
128	2.65E+01	5.09E+01	6.33E+01	8.93E+01	1.62E+02	3.26E+02	6.73E+02
256	5.25E+01	8.78E+01	9.88E+01	1.20E+02	1.73E+02	3.23E+02	6.72E+02
512	8.18E+01	1.24E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.39E+02
1024	2.68E+02	2.37E+02	2.92E+02	3.08E+02	3.87E+02	5.19E+02	6.93E+02

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.34E+00	2.83E+01	5.03E+01	9.05E+01	1.70E+02	3.27E+02	6.37E+02
32	1.02E+01	2.84E+01	5.07E+01	8.52E+01	1.64E+02	3.33E+02	7.09E+02
64	1.90E+01	3.30E+01	5.18E+01	8.31E+01	1.61E+02	3.19E+02	6.39E+02
128	2.64E+01	5.10E+01	6.34E+01	8.93E+01	1.62E+02	3.23E+02	6.75E+02
256	5.26E+01	8.78E+01	9.88E+01	1.20E+02	1.73E+02	3.23E+02	6.69E+02
512	8.11E+01	1.24E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.39E+02
1024	2.65E+02	2.35E+02	2.91E+02	3.07E+02	3.87E+02	5.19E+02	6.93E+02

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.34E+00	2.83E+01	5.03E+01	9.03E+01	1.70E+02	3.28E+02	6.38E+02
32	1.02E+01	2.83E+01	5.06E+01	8.52E+01	1.64E+02	3.33E+02	7.05E+02
64	1.88E+01	3.30E+01	5.17E+01	8.31E+01	1.61E+02	3.19E+02	6.37E+02
128	2.64E+01	5.12E+01	6.35E+01	8.92E+01	1.62E+02	3.23E+02	6.77E+02
256	5.28E+01	8.77E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.66E+02
512	8.04E+01	1.24E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.40E+02
1024	2.62E+02	2.34E+02	2.90E+02	3.06E+02	3.87E+02	5.19E+02	6.93E+02

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.34E+00	2.82E+01	5.03E+01	9.02E+01	1.70E+02	3.27E+02	6.40E+02
32	1.02E+01	2.83E+01	5.04E+01	8.52E+01	1.63E+02	3.33E+02	7.01E+02
64	1.87E+01	3.30E+01	5.17E+01	8.32E+01	1.61E+02	3.19E+02	6.35E+02
128	2.63E+01	5.13E+01	6.36E+01	8.92E+01	1.63E+02	3.24E+02	6.79E+02
256	5.29E+01	8.77E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.63E+02
512	7.98E+01	1.22E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.40E+02
1024	2.59E+02	2.33E+02	2.89E+02	3.06E+02	3.87E+02	5.19E+02	6.93E+02

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.34E+00	2.82E+01	5.03E+01	9.01E+01	1.70E+02	3.28E+02	6.39E+02
32	1.02E+01	2.83E+01	5.04E+01	8.52E+01	1.63E+02	3.39E+02	6.75E+02
64	1.86E+01	3.30E+01	5.18E+01	8.32E+01	1.61E+02	3.19E+02	6.34E+02
128	2.63E+01	5.14E+01	6.37E+01	8.92E+01	1.63E+02	3.24E+02	6.83E+02
256	5.31E+01	8.77E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.60E+02
512	7.91E+01	1.22E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.40E+02
1024	2.46E+02	2.32E+02	2.88E+02	3.05E+02	3.87E+02	5.19E+02	7.10E+02

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.33E+00	2.82E+01	5.03E+01	8.99E+01	1.70E+02	3.31E+02	7.26E+02
32	1.02E+01	2.84E+01	5.05E+01	8.53E+01	1.64E+02	3.54E+02	6.67E+02
64	1.85E+01	3.30E+01	5.19E+01	8.32E+01	1.61E+02	3.20E+02	6.32E+02
128	2.62E+01	5.15E+01	6.36E+01	8.93E+01	1.63E+02	3.25E+02	6.82E+02
256	5.33E+01	8.77E+01	9.89E+01	1.20E+02	1.73E+02	3.23E+02	6.57E+02
512	7.84E+01	1.21E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.40E+02
1024	2.43E+02	2.31E+02	2.87E+02	3.04E+02	3.86E+02	5.19E+02	7.10E+02

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	6.33E+00	2.82E+01	5.03E+01	8.97E+01	1.70E+02	3.30E+02	7.22E+02
32	1.02E+01	2.84E+01	5.07E+01	8.54E+01	1.64E+02	3.52E+02	6.72E+02
64	1.90E+01	3.31E+01	5.20E+01	8.32E+01	1.61E+02	3.19E+02	6.47E+02
128	2.63E+01	5.17E+01	6.37E+01	8.93E+01	1.63E+02	3.24E+02	6.76E+02
256	5.38E+01	8.76E+01	9.86E+01	1.20E+02	1.73E+02	3.23E+02	6.52E+02
512	7.70E+01	1.20E+02	1.46E+02	1.82E+02	2.42E+02	3.42E+02	6.40E+02
1024	2.36E+02	2.29E+02	2.85E+02	3.03E+02	3.86E+02	5.19E+02	7.10E+02

B.3.4 Rising transition delays table

SP(0)=0.01							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.49E+01	4.85E+01	8.55E+01	1.36E+02	2.68E+02	5.20E+02	1.01E+03
32	1.72E+01	4.85E+01	8.33E+01	1.37E+02	2.60E+02	5.15E+02	1.02E+03
64	2.65E+01	5.31E+01	8.69E+01	1.40E+02	2.59E+02	5.12E+02	1.01E+03
128	4.38E+01	7.25E+01	9.57E+01	1.44E+02	2.69E+02	5.07E+02	1.02E+03
256	7.27E+01	1.14E+02	1.35E+02	1.81E+02	2.76E+02	5.12E+02	1.01E+03
512	1.34E+02	1.74E+02	2.19E+02	2.52E+02	3.34E+02	5.45E+02	1.01E+03
1024	2.53E+02	2.37E+02	4.37E+02	4.58E+02	5.98E+02	6.98E+02	1.09E+03

SP(0)=0.2							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.52E+01	5.07E+01	9.08E+01	1.43E+02	2.77E+02	5.54E+02	1.05E+03
32	1.75E+01	5.05E+01	8.72E+01	1.44E+02	2.73E+02	5.40E+02	1.09E+03
64	2.64E+01	5.53E+01	8.99E+01	1.42E+02	2.73E+02	5.40E+02	1.05E+03
128	4.45E+01	7.54E+01	1.02E+02	1.50E+02	2.83E+02	5.33E+02	1.07E+03
256	7.65E+01	1.17E+02	1.41E+02	1.91E+02	2.88E+02	5.37E+02	1.05E+03
512	1.42E+02	1.80E+02	2.21E+02	2.56E+02	3.42E+02	5.67E+02	1.05E+03
1024	1.65E+02	2.87E+02	4.42E+02	5.36E+02	5.94E+02	6.96E+02	1.13E+03

SP(0)=0.3							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.55E+01	5.22E+01	9.48E+01	1.48E+02	2.86E+02	5.77E+02	1.08E+03
32	1.73E+01	5.18E+01	9.00E+01	1.49E+02	2.83E+02	5.58E+02	1.14E+03
64	2.62E+01	5.59E+01	9.19E+01	1.46E+02	2.84E+02	5.62E+02	1.08E+03
128	4.46E+01	7.63E+01	1.02E+02	1.57E+02	2.97E+02	5.52E+02	1.10E+03
256	8.12E+01	1.20E+02	1.41E+02	1.97E+02	2.96E+02	5.54E+02	1.08E+03
512	1.48E+02	1.84E+02	2.23E+02	2.59E+02	3.47E+02	5.90E+02	1.09E+03
1024	1.63E+02	3.24E+02	4.42E+02	5.45E+02	5.91E+02	6.95E+02	1.16E+03

SP(0)=0.4							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.57E+01	5.34E+01	1.02E+02	1.52E+02	2.93E+02	5.96E+02	1.11E+03
32	1.71E+01	5.29E+01	9.21E+01	1.54E+02	2.91E+02	5.73E+02	1.19E+03
64	2.59E+01	5.70E+01	9.37E+01	1.50E+02	2.91E+02	5.82E+02	1.11E+03
128	4.47E+01	7.63E+01	1.03E+02	1.60E+02	3.05E+02	5.68E+02	1.12E+03
256	8.43E+01	1.23E+02	1.45E+02	2.01E+02	3.02E+02	5.69E+02	1.11E+03
512	1.54E+02	1.88E+02	2.25E+02	2.62E+02	3.52E+02	6.04E+02	1.12E+03
1024	1.63E+02	3.42E+02	4.41E+02	5.48E+02	5.89E+02	7.01E+02	1.19E+03

SP(0)=0.5							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.59E+01	5.45E+01	1.02E+02	1.56E+02	3.00E+02	6.12E+02	1.14E+03
32	1.69E+01	5.38E+01	9.40E+01	1.59E+02	2.99E+02	5.88E+02	1.24E+03
64	2.61E+01	5.80E+01	9.61E+01	1.53E+02	2.97E+02	6.01E+02	1.14E+03
128	4.51E+01	7.61E+01	1.05E+02	1.62E+02	3.09E+02	5.82E+02	1.14E+03
256	8.73E+01	1.26E+02	1.46E+02	2.05E+02	3.15E+02	5.83E+02	1.13E+03
512	1.59E+02	1.91E+02	2.27E+02	2.64E+02	3.57E+02	6.20E+02	1.15E+03
1024	1.62E+02	3.47E+02	4.40E+02	5.49E+02	5.85E+02	7.06E+02	1.21E+03

SP(0)=0.6							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.61E+01	5.55E+01	1.03E+02	1.59E+02	3.06E+02	6.28E+02	1.16E+03
32	1.70E+01	5.47E+01	9.58E+01	1.63E+02	3.06E+02	6.00E+02	1.24E+03
64	2.62E+01	5.88E+01	9.82E+01	1.55E+02	3.04E+02	6.19E+02	1.16E+03
128	4.56E+01	7.24E+01	1.07E+02	1.60E+02	3.12E+02	5.97E+02	1.16E+03
256	9.02E+01	1.29E+02	1.48E+02	2.09E+02	3.25E+02	6.11E+02	1.16E+03
512	1.64E+02	1.96E+02	2.29E+02	2.67E+02	3.64E+02	6.43E+02	1.18E+03
1024	1.62E+02	3.50E+02	4.38E+02	5.50E+02	5.81E+02	7.19E+02	1.24E+03

SP(0)=0.7							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.62E+01	5.65E+01	1.04E+02	1.63E+02	3.12E+02	6.43E+02	1.19E+03
32	1.71E+01	5.56E+01	9.75E+01	1.68E+02	3.14E+02	6.11E+02	1.36E+03
64	2.63E+01	5.96E+01	1.00E+02	1.58E+02	3.11E+02	6.36E+02	1.19E+03
128	4.59E+01	7.48E+01	1.11E+02	1.68E+02	3.15E+02	6.11E+02	1.18E+03
256	9.32E+01	1.30E+02	1.50E+02	2.13E+02	3.36E+02	6.28E+02	1.19E+03
512	1.69E+02	2.01E+02	2.32E+02	2.70E+02	3.72E+02	6.50E+02	1.22E+03
1024	1.60E+02	3.52E+02	5.05E+02	5.49E+02	5.75E+02	7.28E+02	1.26E+03

SP(0)=0.8							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.64E+01	5.77E+01	1.05E+02	1.66E+02	3.19E+02	6.59E+02	1.22E+03
32	1.74E+01	5.67E+01	9.94E+01	1.70E+02	3.22E+02	6.23E+02	1.37E+03
64	2.64E+01	6.12E+01	1.08E+02	1.62E+02	3.18E+02	6.97E+02	1.22E+03
128	5.30E+01	7.46E+01	1.13E+02	1.71E+02	3.18E+02	6.26E+02	1.21E+03
256	9.64E+01	1.30E+02	1.52E+02	2.18E+02	3.49E+02	6.50E+02	1.23E+03
512	1.75E+02	2.08E+02	2.36E+02	2.74E+02	3.79E+02	6.50E+02	1.28E+03
1024	1.58E+02	3.53E+02	5.06E+02	5.47E+02	5.69E+02	7.57E+02	1.28E+03

SP(0)=0.9							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.66E+01	5.92E+01	1.06E+02	1.71E+02	3.26E+02	6.76E+02	1.24E+03
32	1.78E+01	5.80E+01	1.02E+02	1.73E+02	3.32E+02	6.37E+02	1.39E+03
64	2.65E+01	6.25E+01	1.11E+02	1.67E+02	3.25E+02	7.03E+02	1.25E+03
128	5.34E+01	7.45E+01	1.14E+02	1.75E+02	3.21E+02	6.44E+02	1.24E+03
256	9.92E+01	1.31E+02	1.55E+02	2.23E+02	3.48E+02	6.65E+02	1.27E+03
512	1.82E+02	2.15E+02	2.41E+02	2.77E+02	3.86E+02	6.65E+02	1.25E+03
1024	1.71E+02	3.57E+02	5.06E+02	5.44E+02	5.61E+02	7.71E+02	1.31E+03

SP(0)=0.99							
tr\C_load	**0**	**3.75**	**7.5**	**13**	**26**	**52**	**104**
16	1.73E+01	6.23E+01	1.10E+02	1.80E+02	3.42E+02	7.60E+02	1.30E+03
32	1.84E+01	6.07E+01	1.07E+02	1.78E+02	3.51E+02	6.65E+02	1.43E+03
64	2.66E+01	6.48E+01	1.14E+02	1.77E+02	3.39E+02	7.15E+02	1.33E+03
128	5.37E+01	7.63E+01	1.18E+02	1.82E+02	3.38E+02	6.93E+02	1.31E+03
256	9.94E+01	1.32E+02	1.78E+02	2.29E+02	3.58E+02	6.75E+02	1.36E+03
512	1.97E+02	2.39E+02	2.68E+02	3.05E+02	4.17E+02	6.92E+02	1.35E+03
1024	1.43E+02	4.27E+02	5.08E+02	5.34E+02	5.37E+02	7.96E+02	1.37E+03

Appendix C

Matlab Code for Modelling Dynamic Power

This Chapter give details about the data and code for modelling the the dynamic power in a multi-core system.

C.1 Data sets

Table C.1: Input data

Benchmark	**Corner**	**Frequency**	**Vdd**	**IPC_core_0**	**IPC_core_1**	**IPC_core_2**	**IPC_core_3**
barnes	Nominal	2.66	1.2	0.69	0.71	0.7	0.72
blackscholes	Nominal	2.66	1.2	0	1.38	1.38	1.38
bodytrack	Nominal	2.66	1.2	0.31	1.38	1.38	0
canneal	Nominal	2.66	1.2	0	0.54	0.54	0.54
cholesky	Nominal	2.66	1.2	2.06	1.95	2	1.92
dedup	Nominal	2.66	1.2	0	0.04	0.02	0.24
facesim	Nominal	2.66	1.2	0.83	0.83	0.84	0.81
fft	Nominal	2.66	1.2	1.43	1.43	1.43	1.43
fluidanimate	Nominal	2.66	1.2	0	0.87	0.87	0
fmm	Nominal	2.66	1.2	2.43	2.47	2.4	2.4
freqmine	Nominal	2.66	1.2	1.17	0	0	0
lu	Nominal	2.66	1.2	2.13	2.03	1.94	2.03
ocean	Nominal	2.66	1.2	0.69	0.69	0.69	0.69
radiosity	Nominal	2.66	1.2	0.65	0.72	0.57	0.66
radix	Nominal	2.66	1.2	0.41	0.41	0.41	0.41
raytrace	Nominal	2.66	1.2	0	1.74	1.74	1.75
streamcluster	Nominal	2.66	1.2	0.03	1.12	1.14	1.12
swaptions	Nominal	2.66	1.2	0	1.28	1.28	1.45
vips	Nominal	2.66	1.2	0.1	2.57	2.58	0
water	Nominal	2.66	1.2	1.25	1.25	1.34	1.27
barnes	Max	3.06	0.85	0.69	0.7	0.71	0.72
blackscholes	Max	3.06	0.85	0	1.39	1.39	1.39
bodytrack	Max	3.06	0.85	0.31	1.37	1.37	0
canneal	Max	3.06	0.85	0	0.51	0.51	0.51
cholesky	Max	3.06	0.85	1.96	1.88	1.92	1.8
dedup	Max	3.06	0.85	0	0.03	0.02	0.2
facesim	Max	3.06	0.85	0.78	0.78	0.79	0.76
fft	Max	3.06	0.85	1.34	1.34	1.34	1.34
fluidanimate	Max	3.06	0.85	0	0.87	0.87	0
fmm	Max	3.06	0.85	2.4	2.47	2.4	2.43
freqmine	Max	3.06	0.85	1.16	0	0	0
lu	Max	3.06	0.85	2.11	2.01	2.01	1.92
ocean	Max	3.06	0.85	0.64	0.64	0.64	0.65
radiosity	Max	3.06	0.85	0.6	0.67	0.62	0.52
radix	Max	3.06	0.85	0.41	0.41	0.41	0.41
raytrace	Max	3.06	0.85	0.83	0.83	0.82	0.8
streamcluster	Max	3.06	0.85	0.03	1.12	1.13	1.13
swaptions	Max	3.06	0.85	0	1.28	1.28	1.53
vips	Max	3.06	0.85	0.1	2.57	2.58	0
water	Max	3.06	0.85	1.25	1.25	1.33	1.26
barnes	Min	1.7	1.5	0.69	0.71	0.72	0.7
blackscholes	Min	1.7	1.5	0	1.39	1.39	1.39
bodytrack	Min	1.7	1.5	0.31	1.38	1.38	0
canneal	Min	1.7	1.5	0	0.61	0.61	0.61
cholesky	Min	1.7	1.5	2.27	2.18	2.11	2.13
dedup	Min	1.7	1.5	0	0.06	0.04	0.38
facesim	Min	1.7	1.5	1	0.99	0.97	1
fft	Min	1.7	1.5	1.78	1.78	1.78	1.78
fluidanimate	Min	1.7	1.5	0	0.87	0.87	0
fmm	Min	1.7	1.5	2.41	2.44	2.41	2.48
freqmine	Min	1.7	1.5	1.19	0	0	0
lu	Min	1.7	1.5	2.13	2.03	1.94	2.03
ocean	Min	1.7	1.5	0.91	0.92	0.92	0.91
radiosity	Min	1.7	1.5	0.7	0.83	0.8	0.89
radix	Min	1.7	1.5	0.41	0.41	0.41	0.41
raytrace	Min	1.7	1.5	0.9	0.89	0.87	0.9
streamcluster	Min	1.7	1.5	0.03	1.13	1.12	1.13
swaptions	Min	1.7	1.5	0	1.27	1.27	1.52
vips	Min	1.7	1.5	0.1	2.58	2.58	0
water	Min	1.7	1.5	1.27	1.27	1.35	1.28

Table C.2: Output power data

Benchmark	**Corner**	**Power_core_0**	**Power_core_1**	**Power_core_2**	**Power_core_3**
barnes	Nominal	5.83391	6.00652	5.90775	6.09756
blackscholes	Nominal	0.609271	9.21822	9.21974	9.22056
bodytrack	Nominal	2.4765	8.87871	8.87207	0.608066
canneal	Nominal	0.608614	3.69783	3.6974	3.69646
cholesky	Nominal	15.0519	14.4279	14.5864	14.2249
dedup	Nominal	0.608077	0.802223	0.730435	2.09423
facesim	Nominal	6.82507	6.79908	6.8569	6.66062
fft	Nominal	8.74034	8.74091	8.73868	8.73987
fluidanimate	Nominal	0.608158	6.01221	6.04471	0.608066
fmm	Nominal	14.493	14.6599	14.2872	14.2375
freqmine	Nominal	7.80965	0.608066	0.608066	0.608066
lu	Nominal	14.763	14.1046	13.5074	14.1118
ocean	Nominal	5.51386	5.52044	5.5484	5.55415
radiosity	Nominal	4.9981	5.44468	4.30746	5.10956
radix	Nominal	2.98451	2.98571	2.98746	2.99175
raytrace	Nominal	6.08335	6.07725	5.88249	6.03322
streamcluster	Nominal	0.781616	8.8005	8.87477	8.79467
swaptions	Nominal	0.60841	8.87352	8.87339	10.5255
vips	Nominal	1.26051	14.1932	14.2343	0.608195
water	Nominal	8.73526	8.7359	9.13314	8.79854
barnes	Max	6.95043	7.0445	7.16909	7.28514
blackscholes	Max	0.306792	12.2438	12.2458	12.2472
bodytrack	Max	2.81746	12.0601	12.0418	0.305089
canneal	Max	0.305826	4.52742	4.52549	4.52696
cholesky	Max	17.9512	17.4513	17.4985	16.8023
dedup	Max	0.305103	0.563126	0.464837	2.03517
facesim	Max	7.56771	7.52408	7.5886	7.36634
fft	Max	10.7774	10.775	10.7743	10.7761
fluidanimate	Max	0.305219	7.77345	7.80799	0.305089
fmm	Max	19.1478	19.7281	19.1718	19.4216
freqmine	Max	10.6396	0.305089	0.305089	0.305089
lu	Max	20.4227	19.4875	19.4978	18.6372
ocean	Max	6.28895	6.29569	6.33168	6.33856
radiosity	Max	5.71774	6.28923	5.87377	4.84491
radix	Max	3.61756	3.62017	3.62267	3.62641
raytrace	Max	7.52126	7.54186	7.46465	7.25612
streamcluster	Max	0.547362	11.0434	11.092	11.1204
swaptions	Max	0.305573	11.6529	11.653	13.9208
vips	Max	1.18506	20.7169	20.7777	0.305273
water	Max	11.1322	11.1333	11.7444	11.2336
barnes	Min	6.08094	6.25025	6.33976	6.1534
blackscholes	Min	0.951294	9.47964	9.48109	9.48197
bodytrack	Min	2.79168	9.10322	9.09661	0.950103
canneal	Min	0.950715	4.40532	4.40463	4.40584
cholesky	Min	16.626	15.9491	15.6977	15.8435
dedup	Min	0.950121	1.26094	1.14593	3.32769
facesim	Min	8.27336	8.24203	8.08025	8.30909
fft	Min	10.9592	10.9595	10.9595	10.9575
fluidanimate	Min	0.950194	6.28703	6.30485	0.950103
fmm	Min	14.4325	14.683	14.4805	14.8494
freqmine	Min	8.14927	0.950103	0.950103	0.950103
lu	Min	14.8699	14.2296	13.635	14.2227
ocean	Min	7.34822	7.40197	7.394	7.35654
radiosity	Min	5.49373	6.47301	6.27457	6.85276
radix	Min	3.30932	3.31344	3.31076	3.31626
raytrace	Min	6.63399	6.61448	6.46107	6.65301
streamcluster	Min	1.12092	9.02578	9.00644	9.06014
swaptions	Min	0.950438	8.99819	8.99808	10.6068
vips	Min	1.59225	14.3268	14.3671	0.95023
water	Min	9.03519	9.03542	9.43032	9.09786

C.2 MATLAB Code

```
clear all;close all;
power_data =
    importdata('C:\Users\hma1g14\Dropbox\MachineLearning\profile.csv')
power_data.data(:,1)
[N, p1] = size(power_data.data);  % N = No. of rows; p1 = No of columns
p = p1-4;  % p = no. of inputs
Y = [power_data.data(:,1:p)];
f = power_data.data(:,p1-3:p1);
Xtr = Y
ytr = f
Xts = Y
yts = f
X = Xtr
y = ytr;
% Non-linear Neural Network
net = feedforwardnet(10);
[net tr] = train(net, X', y');
figure
hold all
plotperform(tr)
net2 = feedforwardnet(20);
[net2 tr2] = train(net2, X', y');

plotperform(tr2)
hold off

f2 = net2(Xts');
%view(net);
perf = perform(net2,f2,yts');


% linear regression with cross-validation
Nfold = 8;
indices = crossvalind('Kfold',f(:,1),Nfold);
MSPE_linear = zeros(Nfold,4);
YL = [Y ones(N,1)];
fL= f
for i = 1:Nfold
    test = (indices == i); train = ~test;
    w = inv(YL(train,:)'*YL(train,:))*YL(train,:)'*fL(train,:);
    fh_linear = YL(test,:)*w;
    MSPE_linear(i,:) = mean((fL(test,:)- fh_linear).^2); % to compare
    the training output with the test data
end
```

```
%disp(['The accuracy of predition: ' num2str(LSE)]);
MSPE_linear
 %average prediction error
avg = sum(MSPE_linear)/Nfold
 %uncertainty on it.
 uncertinity= std(MSPE_linear)
figure;
x = [1:Nfold];
bar(x,MSPE_linear)
legend('Core 1' , 'Core 2', 'Core 2', 'Core 4')
title('Prediction Uncertainty ' )
%legend('X2','Y2')
xlabel('Validaition fold') % x-axis label
ylabel('Mean Squared Error (mse)') % y-axis label
```

powerLearning.m

Appendix D

Thermal Model Generation

This appendix describes how the temperature and BTI model of the multi-core systems been modelled. MATLAB has been used to simulate both the temperature and BTI for each core in the system by simulating the following steps:

1. Input pattern generation (we define the input power and the frequency of each core in this step)
2. Thermal model generation (build the thermal model from the floorplan)
3. Core temperature simulation ()
4. Core stress simulation ()

Matlab Code

```
% This script allows to simulate a selected thermal model:
% Required steps:
% 1 - Input Pattern Generation
%    a) Power input
%    b) CPI, Freq Input - Internally converted in Power through a non
%    linear Power model.
% 2 - Thermal Model Generationn
% 3 - Core temperature Simulation
% 4 - Ageing Simulation

% ********************************************************** %
% *************** GLOBAL PARAMETERS ************************** %
NC = 4 ;       % Number Of Cores

TS = 1  ;               % Discrete Time model - sample time defined in
    Time.Step
    stepsim = true();       % To simulate step by step - required with non linear
    %stepsim = false();     % To simulate with dlsim

%TS = -1;                % Continuos Time model - uses lsim

Type  =             ;   % 1 Layer 1 cell per core
%Type  = 'Reduced2L';   % 2 Layer 1 cell per core
%Type  = 'Full2L';       % 2 Layer 1 cell per core
%Type  = 'NL';

flp = false(); % Put one if you want to generate model from the flp file!

Tenvironment = 310; % [K] Environmetal temperature

if (strcmp(Type,    )) % Parameters for non linear model
    TS = 1   ;
    stepsim = true();
end


PI = false();    % Power Input Vector ??
Plots = true();    % Do you want plot ??

plotcolor =[   ;   ;   ;   ;   ;   ;   ;   ];
markers =[    ;    ;    ;    ;    ;    ;    ;    ];


% ************************************************************ %
% ************ (1) INPUT PATTERN GENERATION **************** %

Time = struct();        % Contains the input vector Time information
Time.Start = 0;         % [s]
Time.End = 40;          % [s]
Time.Points =  5000;    % Number of points
Time.Step = (Time.End - Time.Start)/Time.Points; % Step 20/5000
Time.array = (Time.Start:Time.Step:Time.End)'; % Time vector
Time.array2 = (Time.Start:Time.Step:Time.End/2)';
Time.array3 = ((Time.End/2)+1:Time.Step:Time.End)';

Tenv = ones(length(Time.array),1)*Tenvironment;
% Input vector with environmental temperature

if (PI) % Plot input power vectors
    % Example generation of power stress pattern
    Power = zeros(length(Time.array),NC);
    PMax = 25;
    PMin = 10;
    for i=1:1:2
        %Power(:,i) = square(2*pi*(1/(i*NC))*Time.array);
        Power(:,i) = ones(length(Time.array),1);
    end
    Power(Power==1)=PMax;
    Power(Power==-1)=PMin;
        Power(Power==0)=PMin;

else
    % Example generation of CPI, Freq stress pattern
    CPI = zeros(length(Time.array),NC);
    Freq = zeros(length(Time.array),NC);
    Idleness = ones(length(Time.array),NC);
    % Idleness(:,2)= zeros(length(Time.array),1);
   % let core 2 to be always idle
    half_period = length(Time.array)/2;
    for i=1:1:2499
        Idleness(i,2)= 0;
    end

    for i=2500:1:5001
        Idleness(i,3)= 0;
    end

    Temperature = ones(length(Time.array),NC)*Tenvironment;
    FMax = [2970 2970 2970 2970];
    FMin = [1600 1600 1600 1600];
    %print CPI
    for i=1:1:NC

        CPI(:,i) = ones(length(Time.array),1)*0.5;
        %Freq(:,i) = square(2*pi*(1/(i*NC))*Time.array);
        %high frequency phase
        for j=1:1:2500
        Freq(j,i) = 1;
        end
        %low frequency phase
        for j=2501:1:3500
        Freq(j,1) = -1;
        Freq(j,2) = -1;
        Freq(j,3) = -1;
        Freq(j,4) = 1;
        end
        for j=3501:1:5001
        Freq(j,i) = -1;
        end

        Freq(Freq==1)=FMax(i);
        Freq(Freq==-1)=FMin(i);
    end

    Vdd = Freq2Vdd(Freq); % Convert Freq to corresponding Vdd
    Power = CPI2Pow(Freq,CPI,Idleness,Vdd,Temperature); % Power Model
end


if (Plots) % if u want to see input pattern plot
   figure;
   tmp =   ;
   subplot(3,1,1);
% x = [0.35 0.35];
% y = [0.82 0.77];
% annotation('textarrow',x,y,'String','Core 2 is idle ')
   hold on
   TA = decimate(Time.array,4);
   Pow = decimate(Power,4);
   Pow = transpose(Pow)

   for i=1:1:NC
       %'LineWidth',2,'MarkerSize',4,
    plot(Time.array,Power(:,i),strcat(plotcolor(i),markers(i,:)),
                  ,1:100:length(Time.array));
    tmp = strcat(tmp,        ,num2str(i),     );
   end
   hold off
   eval(strcat(         ,tmp(1:end-1),    ));
   title(              )
   ylabel(             )
   xlabel(           )
end


% ************************************************************ %
% ************** (2) THERMAL MODEL GENERATION ***************** %

if (TS == -1)
    [A,B,C,D]=modmatrices(Type,TS,flp);
    dimA = size(A);
    X0 = ones(dimA(1,1),1)*Tenvironment;
    TModel = ss(A,B,C,D);
else if (strcmp(Type,    ))
        X0=Tenvironment*ones(192,1);
        [A,B,C,D]=modmatricesnl(X0,Time.Step,flp);
        dimA = size(A);
        X0 = ones(dimA(1,1),1)*Tenvironment;
    else
        [A,B,C,D]=modmatrices(Type,Time.Step,flp);
        dimA = size(A);
        X0 = ones(dimA(1,1),1)*Tenvironment;
    end
end

if (flp==1)
% Adapt Power vector to include the component
sz_p = size(Power);
sz_b = size(B);
    if ((sz_b(1,2)-sz_p(1,2)-1)>0)
        Power = [Power(),zeros(sz_p(1,1),sz_b(1,2)-sz_p(1,2)-1)];
    end
end


% ************************************************************ %
% ************** (3) TEMPERATURE SIMULATION ***************** %


if (TS == -1)       % Continuous time
    Temp = lsim(TModel,[Power,Tenv],Time.array,X0);
else                % Discrete time
    if (stepsim)    % Step by Step simulation
        Temp = zeros(length(Time.array),NC);
        tmp =  C*X0;
        Temp(1,:) = tmp(tmp~=0);
        T = X0;
        if (strcmp(Type,    ))  % Non Linear Model
            for j=2:1:length(Time.array)
                [A,B,C,D]=modmatricesnl(T,Time.Step,flp);
                T = A*T + B*[Power(j-1,:),Tenv(j-1,:)]';
                tmp =  C*T;
                Temp(j,:) = tmp(tmp~=0);
            end
        else % Linear model
             for j=2:1:length(Time.array)
                T = A*T + B*[Power(j-1,:),Tenv(j-1,:)]';
                tmp =  C*T;
                Temp(j,:) = tmp(tmp~=0);
             end
        end
    else % Simulate with dlsim
```

```
            Temp = dlsim(A,B,C,D,[Power,Tenv],X0);
            Temp = C(:,1:end/2)*Temp';
            Temp = Temp';
        end
end
% ************************************************************ %
% *************** (4) AGEING SIMULATION ****************** %
%idleness means before here: 1 not idle; 0 idle


% normilasing Temp
for i=1:1:NC
norm_Temp(:,i) = (Temp(:,i) - min(Temp(:,:))) / ( max(Temp(:,:)) -
    min(Temp(:,:)) );
end

NBTI_stress = 0.5*(abs(Idleness-1)+(norm_Temp));
fileID = fopen(                    ,   );
fprintf(fileID,     ,
                                   );
fprintf(fileID,       ,max(NBTI_stress(:,1)));
fprintf(fileID,     ,
                                   );
fprintf(fileID,       ,max(NBTI_stress(:,2)));
fprintf(fileID,     ,
                                   );
fprintf(fileID,       ,max(NBTI_stress(:,3)));
fprintf(fileID,     ,
                                   );
fprintf(fileID,       ,max(NBTI_stress(:,4)));
% ************************************************************ %
if (Plots) % Plot output tempeartures
   %figure;
   tmp =   ;
   subplot(3,1,2);
   hold on

   for i=1:1:NC

    plot(Time.array,Temp(:,i),strcat(plotcolor(i),markers(i,:)),               ,1:100:leng
    tmp = strcat(tmp,        ,num2str(i),     );
   end
   hold off
   eval(strcat(         ,tmp(1:end-1),    ));
   title(                   )
   ylabel(              )
   xlabel(         )
```

```
    subplot(3,1,3);
    hold on

    for i=1:1:NC

     plot(Time.array,NBTI_stress(:,i),strcat(plotcolor(i),markers(i,:)),               ,1:1
     tmp = strcat(tmp,        ,num2str(i),     );
    end
    hold off
    eval(strcat(         ,tmp(1:end-1),    ));
    title(             )
    ylabel(                 )
    xlabel(          )
end
```

ThermalANDNBTI_PROACTIVE_sim_4cores_v2.m

Appendix E

Tools Used

E.1 GEM5

GEM5 is a system-level computer system architecture simulator. In this work, I have used GEM5 to simulate ARM and MIPS processors during the analysis of the running applications to obtain the register file stress states in chapters 3 and 4. Also, GEM5 has been used for analysing the register file ageing stress state in chapter 5.

E.2 Design Compiler

Design Compiler is used for logic synthesis and static timing analysis at the gate-level. A soft core processor from OpenRisc and ARM M0 has been synthesised to obtain the critical path at time zero in chapter 4. Design Compiler is also used to obtain the VCD file for any test-bench program executed on the processor mentioned above.

E.3 TetraMAX ATPG

TetraMAX is an Automatic Test Pattern Generator tool from Synopsys. It is used to generate anti-ageing patterns as test patterns in chapter 4.

E.4 HSPICE

HSPICE is a comprehensive low-level circuit simulator. It was used in chapters 2, 3 and 4 for modelling the path delay using its ageing simulation tool, MOSRA Level 3. Also, in chapter 5, MOSRA was used for measuring the setup time for the flip-flop using the bisection tool available in HSPICE.

E.5 Sniper Simulator

Sniper is a multi-core simulator support timing simulations for multi-program workloads and generate Instruction-Per-Cycle (IPC) profile for the executed program. It is been used in chapter 6 to extract the IPC and the idleness for Black-Scholes Benchmark running on Xeon x5550 Gainestown x86 microprocessor.

E.6 McPAT

McPAT is a multi-core simulator support power, area and timing simulation. It is used in chapter 6 to extract power dataset with the support from Sniper Simulator for Black-Scholes Benchmark running on Xeon x5550 Gainestown x86 microprocessor.

Appendix F

Published Papers

Microelectronics Reliability 79 (2017) 79–90

Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/microrel

BTI mitigation by anti-ageing software patterns

Haider Muhi Abbas[*], Basel Halak, Mark Zwolinski

Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK

A R T I C L E I N F O

Keywords:
Negative/Positive Bias Temperature Instability (NBTI/PBTI)
Workload
Application specific systems
Idle time
Anti-ageing

A B S T R A C T

This paper presents a time-redundant technique to mitigate Negative and Positive Bias Temperature Instability (NBTI/PBTI) ageing effects on the functional units of a processor. We have analysed the sources and effects of ageing from the device level to the Instruction Set Architecture (ISA) level, and have found that an application may stress the critical paths in such a way that the circuit has half of its nodes always NBTI-stressed. To mitigate this behaviour, we propose an application-level solution to balance the stress and put the timing-critical gates of the critical path into a relaxed (balanced) mode. The results show that the lifetime of the system can be doubled by applying balanced stress patterns at the software level during the idle time of a processor system.

1. Introduction

Ageing or time-dependent variations in CMOS devices represent a challenge to the design of integrated circuits, along with power consumption and performance. With technology feature sizes scaling down, critical issues related to system reliability, such as soft errors, hard errors, and process variations, [1], are emerging. Ageing-degradation can manifest itself as soft errors that become hard errors, bringing the system to a state where timing constraints are violated, and finally, the system fails to function properly. The mechanisms at the device level responsible for that degradation include: Positive/Negative Bias Temperature Instability (PBTI/NBTI), Hot Carrier Injection (HCI) and Time-Dependent Dielectric Breakdown (TDDB) and manifest themselves as changes in the device threshold voltage, the carrier mobility, and the insulating properties of gate dielectrics, [2].

In this work, we are primarily concerned with the mitigation of Bias Temperature Instability (BTI) and particularly NBTI. BTI has two different phases:

- Stress: Interface traps are generated at the interface of the substrate and gate oxide layers due to electrical stress (i.e. negative bias for PMOS and positive bias for NMOS) that leads to breaking of some of the Si–H or Si–O bonds. Consequently, the threshold voltage of the transistor increases over time.
- Relaxation/Recovery: some of the generated traps are removed from the interface. However, the relaxation phase cannot completely compensate for the effect of the stress phase and therefore the overall effect of BTI is degradation in the threshold voltage of each transistor. The amount of degradation depends on the ratio between the stress period and the total operating period (the duty cycle).

Techniques for reducing stress and increasing recovery include controlling the signal probabilities. This can be done from the inputs of the circuit (Input Vector Control) or during the synthesis process by hiding those nodes with high probabilities of being zero, or by changing the pin order of the gates on the critical paths, [3–5]. However, if we try to relax one node, other nodes in the signal path may be stressed.

The hypothesis of this work is that a processor needs to actively relax to recover from stress, rather than simply doing nothing during idle periods. In modern applications, processors tend to have many short idle periods; thus, simple power gating would not be a good solution for stress optimization, [6]. During normal operational periods, the input states of the transistors may be constant, leaving the transistors stressed. NBTI/PBTI could then be mitigated by applying balanced-stress stimuli to the critical paths at the software-level. Running a program on the processor for a non-functional purpose has been used in on-line testing (i.e. Software-Based Self-Test (SBST) methods) as this does not require modification of the hardware design [7].

The main contributions of this work are:

- A high-level ageing prediction model is proposed which takes into account the actual signal probabilities of the system. To achieve this we have developed a tool to derive the actual stress-recovery ratio of each logic gate.
- An application-level technology-independent mitigation technique is proposed to balance NBTI/PBTI effects. This brings the circuit into a recovery state by changing the nodes that are BTI-stressed to BTI-relaxed.

The organisation of this paper is as follows: ageing causes and mitigation techniques are covered in Section 2. In Section 3, NBTI/PBTI

[*] Corresponding author.

http://dx.doi.org/10.1016/j.microrel.2017.10.009
Received 28 November 2016; Received in revised form 28 September 2017; Accepted 11 October 2017
Available online 23 October 2017

stress analysis is presented. The proposed technique for generating and applying the balancing patterns is presented in Section 4. The results of running the balancing program are given in Section 5. Section 6 concludes the paper.

2. Background and related work

2.1. BTI stress-recovery

BTI in a transistor is caused by the generation of traps at the channel and dielectric interface. BTI in PMOS transistors is referred to as Negative BTI (NBTI), since the gates of PMOS devices are negatively biased with respect to the source (i.e. $V_{gs} = -V_{dd}$). BTI for NMOS transistors is referred to as Positive BTI (PBTI), since the gates of NMOS transistors are positively biased with respect to the source (i.e. $V_{gs} = V_{dd}$). NBTI and PBTI have the same consequences, namely that they increase the threshold voltages and decrease the driving currents of the devices, but for different physical reasons. It is generally accepted that NBTI degradation in SiO_2 and High-κ dielectric is due to dielectric interface traps. PBTI was considered to be negligible before the introduction of High-κ MOSFET technology [8]. In High-κ processes, PBTI degradation is due to a build-up of negative charges in the High-κ layer. Although High-κ technology is impacted by both NBTI and PBTI degradation, NBTI is still much more dominant according to recent results for Replacement Metal Gate (RMG) Technology, [9,10].

The probability of a signal being zero, SP(0), or the duty cycle, reflects the fraction of time spent in the stress state. We simulated the effect of the duty cycle on the threshold voltage using a commercial reliability model, namely the HSPICE MOSRA Built-in Model Level 3 [11]. MOSRA can evaluate both NBTI and PBTI for a circuit at the SPICE level and obtain gate or path degradation, rather than just threshold voltage degradation or leakage current increase at the transistor level, as given by the basic Reaction-Diffusion (RD) model [12]. Decreasing the duty cycle can impact positively on the V_{th} degradation of PMOS devices and negatively on the V_{th} degradation of NMOS devices. MOSRA simulations use degraded device parameters in HSPICE to calculate gate or path delay degradation in timing simulations, [13].

Simulation parameters have been tuned to match the degradation given by statistical data, [14,15]. We applied different SP(0) values to a circuit consisting of two inverters in series to calculate the maximum path delay after 10 years for two different technologies (90 nm from Synopsys and 65 nm from TSMC). We activate only the NBTI effects because PBTI should barely exist at these technology nodes and if modelled would exaggerate the ageing effect. As can be seen from Figs. 1 and 2, the signal probability has an impact on path delay degradation and, thus, on the lifetime. While one node is highly stressed, it will tend to have more static NBTI and by balancing the signal probabilities, the static stress will be reduced as well. As the delay degradation is less at the end of the device lifetime than at the beginning, decreasing the path delay by a small amount could enhance the lifetime of a device by several years. Fig. 3 shows the lifetime of the two inverter chain with a target maximum delay of 0.237 ns for the 90 nm technology from Synopsys and 0.149 ns for the 65 nm technology from TSMC.

2.2. NBTI mitigation techniques

One approach to mitigation is to reduce or even to eliminate design uncertainties. However, in practice, there is more than one contributor to these uncertainties, including EDA tool limitations and complex environmental stress conditions, [16]. Another solution is conservative design under worst-case scenarios, but circuits do not always run at the worst-case condition, and such an over-designed approach is extremely costly with respect to power and area.

At gate level, BTI manifests itself as a time-dependent gate delay, leading eventually to timing violations. In Ref. [17], critical gates are identified and optimization methodologies, e.g. gate resizing, have been proposed for these critical gates. However, NBTI has a dependence on the dynamic operation, such as the supply voltage, spatial or temporal temperature and signal probability and these parameters vary dynamically from gate to gate. Another solution, [3], uses signal probabilities to restructure the logic gates and arrival times of the input signals to reorder the pins. However, signal probabilities are assumed to be 50% for input signals, but in larger systems, the signal probability is dynamic and application-dependent. A technique to mitigate NBTI has been proposed, [18,19], in which the signal probability is modified using Input Vector Control (IVC) or Multiple Input Vector Control (M-IVC) during the stand-by mode of the circuit. However, saving these vectors in memory has costly overheads in terms of area and power and the techniques do not consider the stress probabilities during the operational mode of the circuit.

At circuit and gate levels, different methods have been proposed to reduce NBTI degradation. Supply voltage scaling over time to reduce guard-bands and increase the lifetime of the circuit has been explored, [20]. Scaling the clock frequency has been proposed to detect and mask late transitions generated due to ageing, [21]. Reordering the gate pins and restructuring the transistor network to mitigate the NBTI degradation has also been suggested, [3,22].

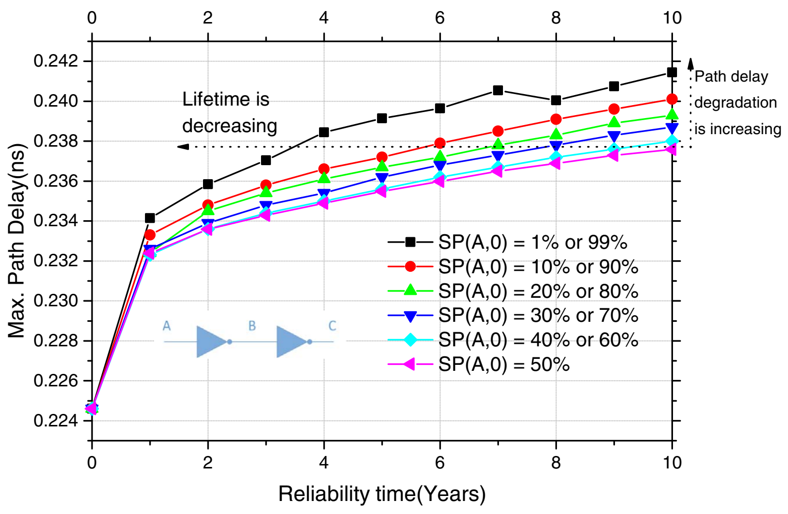


Fig. 1. Path delay for a chain of two inverters for different SP(0) values at the input using 90 nm Synopsys Technology with NBTI effect.

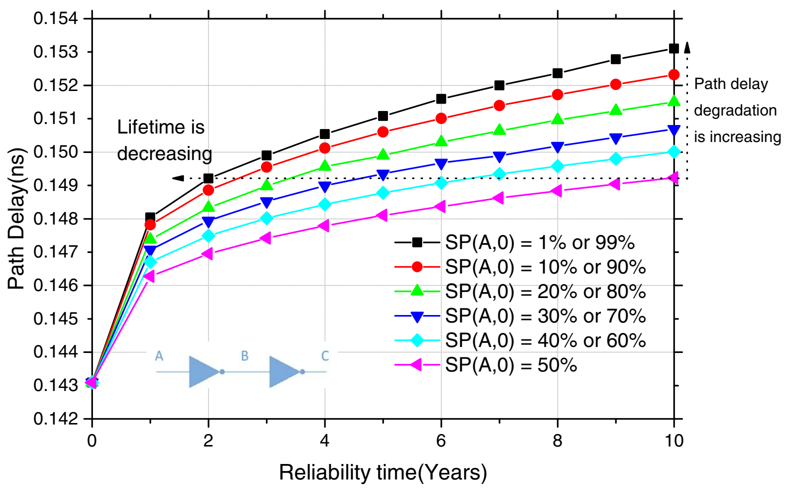


Fig. 2. Path delay for a chain of two inverters for different SP(0) values at the input using 65 nm TSMC Technology with NBTI effect.

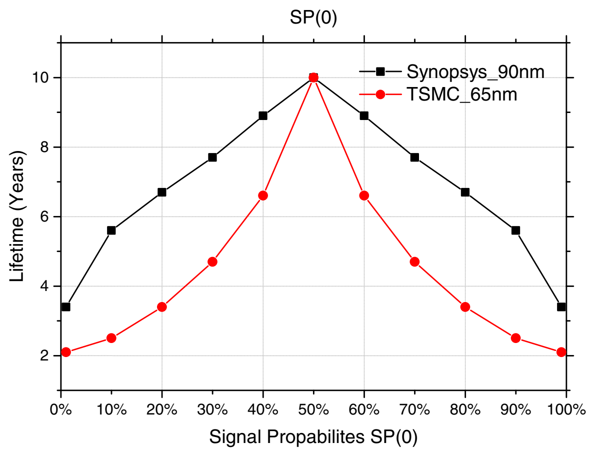


Fig. 3. Lifetime for a chain of two inverters for different Signal Probabilities SP(0) values at the input.

There are many different approaches to controlling these effects. A technique called Dynamic Wear-out/NBTI Management (DNM) has been proposed, with the aim of reducing design margins [23,24]. The DNM approach reduces the power consumption by running the circuit with the least possible supply voltage and changes the supply voltage periodically based on readings from delay sensors. However, the main challenge of this approach is the accuracy of the sensors and the area overheads that could exceed design constraints.

In general, there are three possible approaches to ageing mitigation: proactive, reactive and ageing-aware (protective). The proactive approach works as an estimator for ageing behaviour and is based on a physical model that describes ageing effects (NBTI, HCI and TDDB) at a low level, [4,12,25]. Usually, this approach needs to take into account the different contributions of time-dependent variations (e.g. signal probability, switching activity, temperature and supply voltage). The reactive approach is to monitor on-line the real behaviour resulting from ageing by using delay sensors [23,24]. This approach is more precise and bypasses the complexity of modelling the ageing effects at the system level. However, the main problem of on-line sensors is area and power overheads and to moderate this only a limited numbers of nodes can be monitored. The third, protective, approach tries to alleviate the source of ageing either by reducing the operating temperature or reducing the stress probability in the critical path, [17,26].

2.3. Data dependency of ageing-induced degradation of the processor

Program data determines whether the nodes of the processor are stressed or not. The data includes both opcodes and operands. Firouzi et al., [26], looked at possible NOP instructions in the MIPS processor to reduce ageing. As well as the standard NOP instruction (sll r0, r0, 0) they considered other instructions that had no result (e.g. adding zero) to minimise stress. They proposed software and hardware techniques to assign the best input vector for these NOP instructions. This method would only be helpful, however, if the rate of NOP instructions is high with respect to the total number of operational instructions. To test this hypothesis, we ran different benchmarks from the MiBENCH suite, [27], for two different architectures on the gem5 simulator, [28]. Fig. 4 shows that the number of NOP instructions on the MIPS processor is significant, while on the ARM architecture it is negligible.

Data from other paths can also stress the critical path. For example, assume the adder is in the critical path, and that during the execution of other operations data is routed to the adder, even though the result is not used. From a BTI perspective the critical path through the adder will be stressed. It has been claimed, [29], that the core of a processor can be brought to a failing state by executing a malicious program to age the circuit; this does not consider the signal probabilities of intermediate nodes, however (see Fig. 5). In practice, it is not possible to put all critical path nodes into a fully relaxed state (a Signal Probability of zero, SP(0) = 0%) or a fully stressed state (SP(0) = 100%), Fig. 5. On the other hand, it is possible to balance the stress by controlling the signal probabilities during the idle time of the processor.

3. NBTI/PBTI stress analysis

3.1. Ageing-sensitive critical path selection

The selection of paths to reverse the stress needs to consider both the initial path delays from the post-synthesis analyses and the gate types in the paths. A non-critical path at time zero could become a critical path after a number of years because paths degrade according to different factors, for example, duty cycle, temperature, frequency and circuit topology. Estimating the path most sensitive to ageing depends on model parameters that would not be available until the system has been fabricated and tested in the required environment. So, in this research, we have tried to avoid using an ageing model to define the criteria for selecting specific paths that are potentially vulnerable to ageing. Instead, we define a threshold (θ) for the ageing-critical path delay. For example, the maximum critical path degradation has been

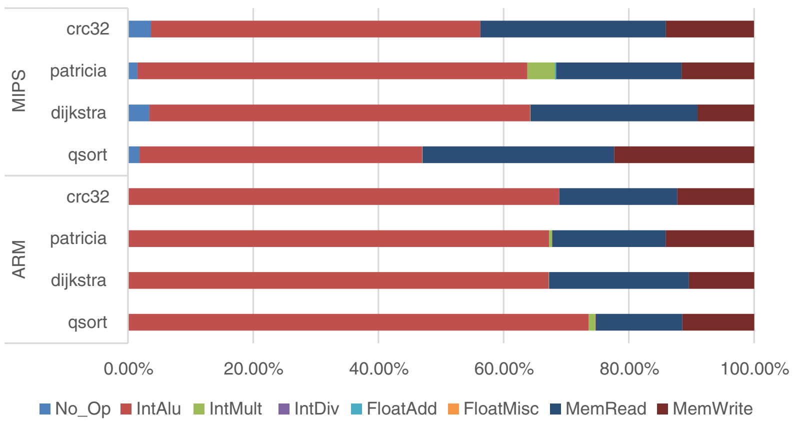


Fig. 4. Percentages of classes of instruction for different benchmarks on MIPS and ARM processors.

measured for different benchmark circuits for 10 years and found to be between 12.3% and 19.5%, [30]. Thus, we can define ageing sensitive critical paths as those paths that have slack in the range of zero to ($\delta_0 \times \theta$), inclusive, or have a path delay between δ_0 and $\delta_0(1-\theta)$, inclusive.

Also we have to consider the effect of process variations on selected paths by defining the worst case possible path deviation due to the process variations as $\Delta\delta_{pv}$. Then, an ageing- and process-sensitive critical path should be selected if its path delay is in the range:

$$\delta_0 \geq \textit{Path Delay} \geq \delta_0(1-\theta-\Delta\delta_{pv}). \quad (1)$$

These paths have nearly balanced path delays, but they could share instances with the first critical path (for example in an adder, if the carry chain path is shared between nearly-critical paths and the first critical path, then any degradation or reversed degradation on the shared part will also affect the ageing-sensitive critical paths). Alternatively, if the critical paths have instances that are independent from one path to another, then all the ageing sensitive critical paths need to be individually analysed for ageing and possible reversal.

3.2. SP(0) distribution on the critical paths of the processor

Combinational logic circuits may show different degradations in each PMOS transistor because different input patterns can lead to different inputs to the CMOS transistors. Some PMOS transistors may degrade more because they have a SP(0) of 99%, but others may not degrade if they have a SP(0) of 1% at their gates. To date, however, there has been no consideration of the SP(0) of the intermediate nodes of the complex gates. For example, in Ref. [29,31], the analysis was done only for the input transistors of gates. In the OR gate of Fig. 6, if IN1 is at “1”, Q would be “1” regardless of IN2, but there is a node, QN, inside the OR gate that would be stressed.

To measure the delay degradation on each net of the critical path, we need to consider the SP(0) of each input and of the internal nodes of

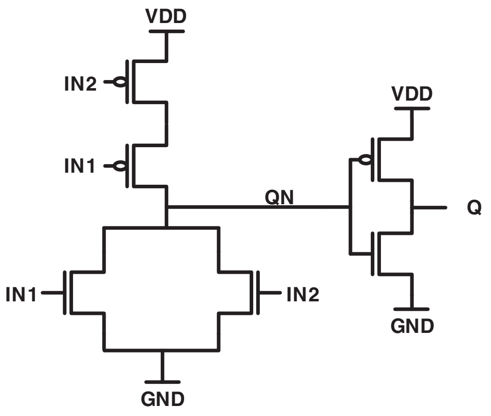


Fig. 6. CMOS circuit for OR gate (NOR + INVERTER).

the gate cells. We used the OpenRISC core for this analysis. First, synthesis was done with the full set of cells available in the 90 nm Synopsys library, including those cells that have internal nodes. There are 2888 critical paths in the circuit, but various critical paths pass though the same gate cells. For example, the first 100 critical paths share more than 92% of the cells in the most critical path, Table 1. Thus, if there is any degradation in the shared part, it would affect all these critical paths. Moreover, the average SP(0) for the first 100 critical paths is around 80% when executing a “Hello World” program. This means the program will stress the critical path. In this example, the nodes are totally NBTI stressed because the probabilities of signals being zero are close to 100%. However, this does not consider the hidden nodes of the compound gates and so the average SP(0) is not correct. These hidden nodes would have complementary values and therefore have no NBTI stress but could have PBTI stress. In other

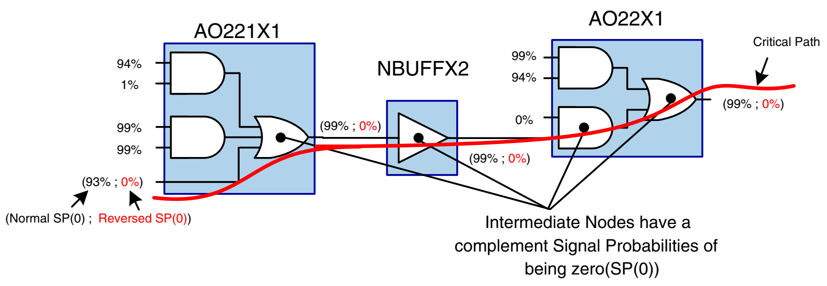

AO221X1
NBUFFX2
AO22X1
Critical Path
94%
1%
99%
99%
(99% ; 0%)
99%
94%
0%
(99% ; 0%)
(99% ; 0%)
(93% ; 0%)
(Normal SP(0) ; Reversed SP(0))
Intermediate Nodes have a complement Signal Probabilities of being zero(SP(0))

Fig. 5. Zero-signal probabilities distribution on the critical path sub-circuit with Normal and Reversed state.

Table 1
SP(0) distribution on the critical paths of the OpenRISC processor for Hello World program, using compound gates.

	1st critical path		2nd critical path		3rd critical path		10th critical path		100th critical path	
	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)	Cell type	SP(0)
1	DFFX1	50%	DFFX1	50%	DFFX1	50%	DFFX1	50%	DFFX1	50%
2	DFFX1	98%	DFFX1	98%	DFFX1	98%	DFFX1	98%	DFFX1	98%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
65	AO22 × 1	93%	AO22 × 1	93%	AO22 × 1	93%	AO22 × 1	93%	AO22 × 1	93%
66	OR2 × 1	93%	OR2 × 1	93%	OR2 × 1	93%	OR2 × 1	93%	XOR3 × 1	94%
67	AO22 × 1	93%	AO22 × 1	93%	AO22 × 1	93%	AO22 × 1	93%	AOI22 × 1	5%
68	XOR3 × 1	94%	XOR3 × 1	94%	XOR3 × 1	94%	XOR3 × 1	94%	NAND4 × 0	94%
69	AOI22 × 1	5%	AOI22 × 1	5%	AOI22 × 1	5%	AOI22 × 1	5%	AO221 × 1	93%
70	NAND4 × 0	94%	NAND4 × 0	94%	NAND4 × 0	94%	NAND4 × 0	94%	NBUFFX2	99%
71	AO221 × 1	93%	AO221 × 1	93%	AO221 × 1	93%	AO221 × 1	93%	AO22 × 1	14%
72	NBUFFX2	99%	NBUFFX2	99%	NBUFFX2	99%	NBUFFX2	99%	DFFX1	14%
73	AO22 × 1	99%	AO22 × 1	77%	AO22 × 1	99%	AO22 × 1	0%		
74	DFFX1	99%	DFFX1	77%	DFFX1	99%	DFFX1	0%		
Average SP(0)		83%		82%		83%		80%		80%

Table 2
SP(0) distribution on the first critical path of the OpenRISC processor for different instructions.

		addi rD,rA,I				movhi rD,I			
	Cell type	I = 0000_H	5555_H	AAAA_H	FFFF_H	0000_H	5555_H	AAAA_H	FFFF_H
1	DFFX1	50%	50%	50%	50%	50%	50%	50%	50%
2	DFFX1	99%	99%	99%	99%	99%	99%	99%	99%
3	NAND2 × 0	0%	0%	0%	0%	0%	0%	0%	0%
4	NAND2 × 0	0%	0%	0%	0%	99%	99%	99%	99%
5	NAND2 × 0	99%	99%	99%	99%	0%	0%	0%	0%
6	NAND4 × 0	0%	0%	0%	0%	99%	99%	99%	99%
7	NOR2 × 0	99%	99%	99%	99%	0%	0%	0%	0%
8	AOBUFX1	99%	99%	99%	99%	0%	0%	0%	0%
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
150	NAND3 × 0	50%	50%	50%	50%	34%	34%	34%	35%
151	NAND2 × 0	0%	0%	0%	0%	0%	0%	0%	0%
152	INVX0	99%	99%	99%	99%	99%	99%	99%	99%
153	NAND3 × 0	0%	0%	0%	0%	0%	0%	0%	0%
154	NAND2 × 0	99%	99%	99%	99%	99%	99%	99%	99%
155	DFFX1	99%	99%	99%	99%	99%	99%	99%	99%
Average SP(0)		49%	49%	49%	49%	48%	49%	48%	48%

words, a circuit with SP(0) close to 0% would have hidden nodes with SP(0) close to 100%. To calculate a more accurate figure, we ran different instructions on a processor synthesized using only cells that have only one-level of transistors, in order to avoid any hidden nodes, as given in Table 2. The SP(0) at each node is generally the complement of that in the previous node and the average SP(0) is around 50%. So, the object should be to reverse these signal probabilities to obtain signal probabilities that are as balanced as possible (around 50%), rather than reducing one signal probability to avoid NBTI stress. This example is only used to show the signal probabilities of the hidden nodes, as this case is not feasible in real synthesis.

3.3. Impact of instruction /program level workload on the stress probabilities

To study the effect of different instructions on the stress of the critical or nearly critical paths, we ran two different instructions with four different operands each, to determine whether the opcode or the data have a significant impact. The synthesis was been done using only cells that have no hidden nodes and the SP(0) at each node is generally the inverse of that of the previous node in the path. For the 155 nodes in the critical path, the average SP(0) was around 50%, as can be seen from the symmetry of the histograms in Fig. 7. The average SP(0) of the paths does not change significantly with the opcodes or operands. However, the signal probability distributions on the critical path do depend on the opcode and operands, as shown in Fig. 7 ("movhi rD,5555_H"), where the signal probabilities tend to the extremes (10% > SP(0) > 90%) even with symmetrical distributions that stress the critical paths with both NBTI and PBTI. Similarly different MiBench benchmarks show nearly symmetrical average stress, as can be seen from Fig. 8. Therefore, it would be desirable to reduce the number of nodes at the extremes of these histograms (e.g. 25% > SP(0) > 75%).

Hence, we conclude that a single instruction or program will not relax or stress 100% of the critical paths, but that a program-level solution could relax or stress some specific nodes. In other words, if there are some nodes in the processor that could face a continual stress while others are not stressed all, it possible to balance that effect at the application level.

3.4. Gate level stress balancing

We considered balancing the signal state of basic logic gates (inverter, NAND and NOR) compared with inverting the signal probability. We examined how inverting the signal probability would affect the ageing degradation. We have used HSPICE for simulating path delays and modelled the NBTI using MOSRA Level 3 considering two different cases:

- CASE A: Unbalanced stressed nodes — the nodes of the critical paths are either significantly NBTI-stressed (SP(0) greater than 75%) or significantly unstressed (or PBTI-stressed) (SP(0) less than 25%).
- CASE B: Balanced stressed nodes — the nodes of the critical path have SP(0) around 50%.

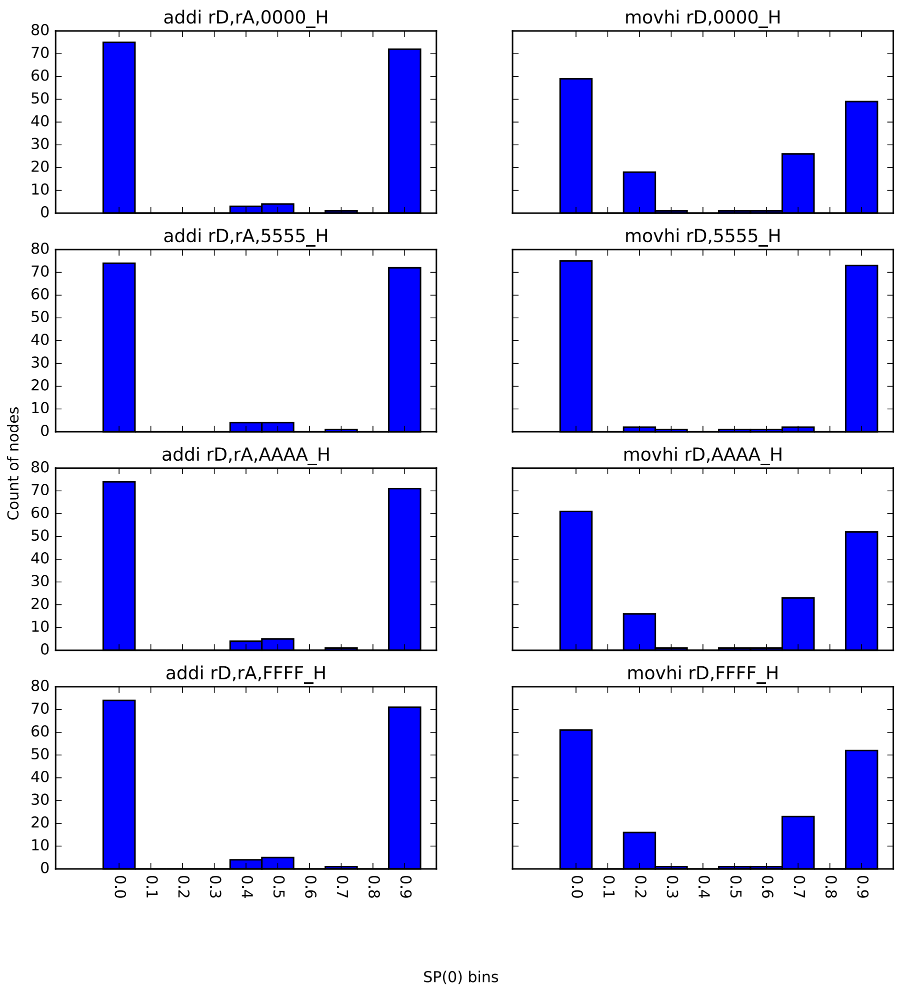


Fig. 7. SP(0) distribution on the first critical path of the OpenRISC processor for different instructions.

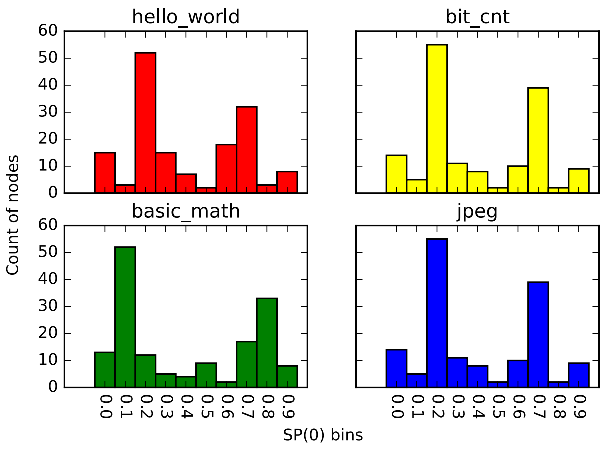


Fig. 8. SP(0) distribution on the first critical path of the OpenRISC processor for different programs.

For the inverter, we simulated the degradation of a path of two inverters over ten years using the cases discussed above. The results show an advantage of 23.17% in the path delay and more than 50% in terms of time, Fig. 9.

For NAND and NOR gates, the same simulation as for the inverter was done but also considering two further dependencies: the signal probabilities of the secondary inputs of the gates, and the input pin order. The results show that swapping input pins can decrease the advantage obtained from balancing the signal probabilities. Also, the signal probabilities of the secondary input will not significantly affect the benefits of balancing the signal probabilities over the critical path, Tables 3 and 4.

We also considered how the balanced stress patterns affect the remaining paths of the circuit. To answer this fundamental question, we examined the proposed technique on a two-bit adder. In this example, there is one target critical path and two nearly-critical paths, Fig. 10. In this example, we extracted the critical paths list after synthesizing the circuit using Design Compiler. Again we used the MOSRA Level 3 model in HSPICE simulations to model the degradation of the circuit, using the two above-mentioned cases. The results show that for all paths, there will be an advantage up to 50% in the expected lifetime from balancing the signal probabilities in the critical path (Fig. 11). Fig. 11 also shows that nearly critical paths share more than half of their nodes with the target critical path. Thus, any advantage in balancing the signal probabilities of the critical path will lead to an advantage in the remaining paths. If the nearly-critical paths do not share nodes with the critical path, then it is possible to control both the critical and the nearly-critical paths in parallel.

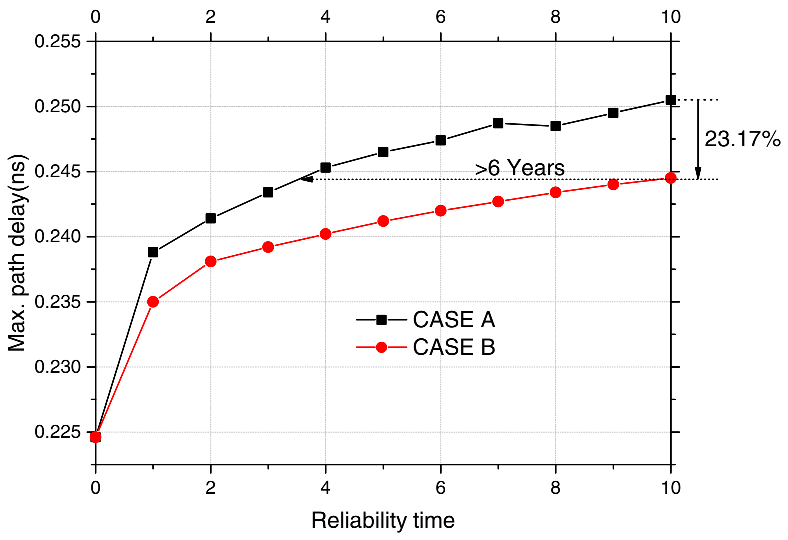


Fig. 9. Path delay degradation for a chain of two Inverters considering balanced and unbalanced signal probabilities.

Table 3
Path delay degradation advantages for a chain of two NAND2 considering balanced and unbalanced signal probabilities.

SP(0) of the 2nd input	Swap input order	CASE A degradation	CASE B degradation	Advantage over path delay	Advantage over years
50%	No	9.27%	7.25%	21.75%	6 years
50%	Yes	9.49%	7.62%	19.73%	5 years
99%	No	7.35%	5.82%	20.80%	5 years
99%	Yes	8.50%	6.88%	19.10%	5 years
1%	No	10.38%	8.34%	19.69%	5 years
1%	Yes	10.46%	8.18%	21.82%	5 years

Table 4
Path delay degradation advantages for a chain of two NOR2 considering balanced and unbalanced signal probabilities.

SP(0) of the 2nd input	Swap input order	CASE A degradation	CASE B degradation	Advantage over path delay	Advantage over years
50%	No	8.55%	6.81%	20.35%	5 years
50%	Yes	9.10%	7.57%	16.80%	4 years
99%	No	8.55%	6.81%	20.35%	5 years
99%	Yes	9.76%	8.77%	10.18%	3 years
1%	No	5.83%	4.54%	22.08%	6 years
1%	Yes	7.38%	5.68%	23.08%	3 years

4. Proposed technique

We propose a two-phase technique to mitigate the BTI ageing effects. In the first phase anti-ageing patterns (the balance states) are generated and these patterns are applied in the second phase by executing a stress-relief program instead of running a process idle task.

The flow of the first phase of the proposed techniques is illustrated in Fig. 12. We find the normal states (the Critical Path Stress States) of the nodes that need to be balanced by running different benchmarks and instructions. We obtain the signal probabilities of the nets being stressed to logic zero (SP(0)) from gate-level simulations of the processor executing benchmarks. We calculate the SP(0) of the critical paths that have slacks less than predefined maximum path delay degradation.

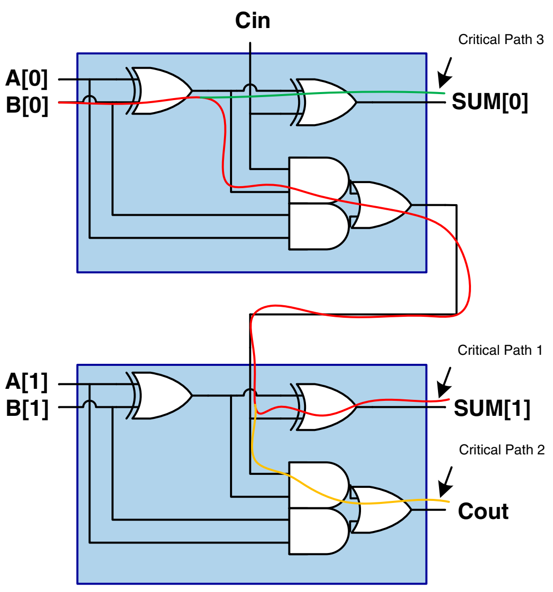


Fig. 10. The most three critical paths on the two-bit adder.

The second phase of the technique is to balance the effect of BT by reversing the average signal probabilities by applying stress-relaxing patterns to the timing-critical components in the functional unit of the processor during idle states.

4.1. Case study: program-level NBTI/PBTI balancing

We synthesized an OpenRISC 1200 processor core using the 90 nm Synopsys technology.[1] The VCD (Value Change Dump) file from each post-synthesis simulation contains both switching activity, that is used to estimate the dynamic power at the design phase, and the signal probability that we used to estimate the BTI effect on performance degradation. From this we extracted the SP(0) for all the nets of the

[1] The technology is not important because the technique depends on reversing the SP (0) rather than estimating the ageing. Hence, it is also independent of the BTI model.

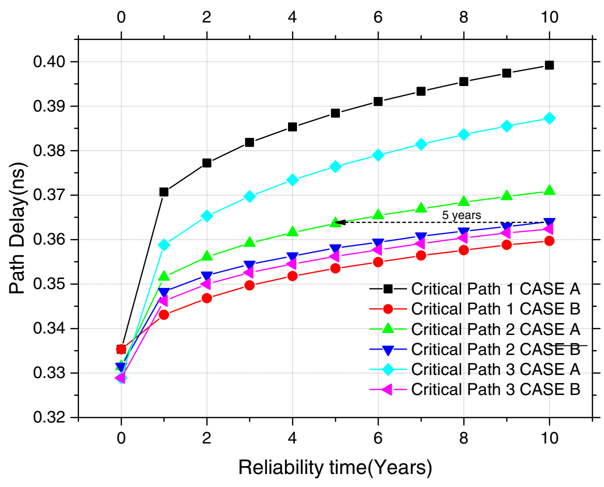
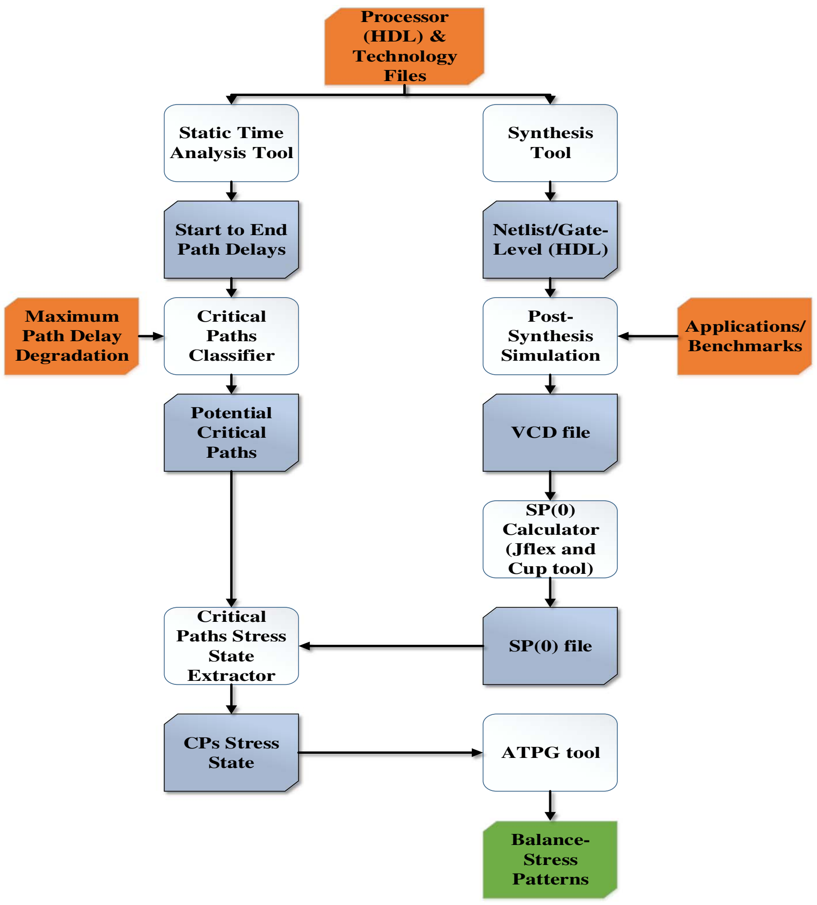


Fig. 11. Path delay degradation of the most three critical paths in the two-bit adder considering unbalanced and balanced signal probabilities over the critical path.



Processor (HDL) & Technology Files
Static Time Analysis Tool
Synthesis Tool
Start to End Path Delays
Netlist/Gate-Level (HDL)
Maximum Path Delay Degradation
Critical Paths Classifier
Post-Synthesis Simulation
Applications/ Benchmarks
Potential Critical Paths
VCD file
SP(0) Calculator (Jflex and Cup tool)
Critical Paths Stress State Extractor
SP(0) file
CPs Stress State
ATPG tool
Balance-Stress Patterns

Fig. 12. BTI balance-stress-patterns generating flow.

processor. To balance the effect of signal probability on the critical path, we need to find input patterns that will invert the signal states. This is effectively the same as generating test patterns for single stuck faults. We used an ATPG tool to find test patterns for stuck-at-0 faults on nets that have high SP(0) so as to set those nets to ‘1’. In the same way, we generated patterns for stuck-at-1 faults on the nets that have low SP(0).

Table 5
OpenRISC balance-stress patterns.

A[31:0]				B[31:0]			
48	CB	3E	67	D4	40	7A	4C
24	65	9F	2D	EA	20	3D	3C
DA	F9	F1	D1	21	50	64	F2
25	B7	C6	93	C4	E8	48	35
DE	4F	08	A3	F9	CD	6D	DE
BF	58	FA	23	6A	33	27	3F
67	5B	2E	9C	58	C3	65	7F
AC	8C	03	18	A4	DC	93	8D

The critical paths of the OpenRISC 1200 processor are in the adder. 38 nodes have an SP(0) greater than 75%; 10 nodes have an SP(0) less than 25%.

The ATPG tool found 8 test patterns to set these nodes to balanced stress conditions. Each pattern will apply balanced stress to one or more nodes; the full set is needed for every node. As the results given in Section 5 show, the percentages of stressed nodes will be reduced significantly after applying these patterns. Table 5 shows these patterns as they would be applied to inputs A[31 … 0] and B[31 …0] of the adder. The patterns could be applied either in a test mode or by writing a program.

The OpenRISC 1200 Instruction Set Architecture has only a 16-bit

```
l.movhi r1, K
# Stimulate the first pattern
l.lws r2, 0(r1)
l.lws r3, 1(r1)
l.add r4, r2, r3
# Stimulate the second pattern
l.lws r2, 2(r1)
l.lws r3, 3(r1)
l.add r4, r2, r3
     .
     .
# Stimulate the eighth pattern
l.lws r2, 14(r1)
l.lws r3, 15(r1)
l.add r4, r2, r3
l.rfe # Return From Exception
```

Fig. 13. Balancing program.

immediate mode. These balance-stress patterns have a 32-bit widths and so are stored in consecutive memory locations starting with address K. The program, Fig. 13, transfers K (immediate value) to register 1 for use as an offset address. Then two patterns are loaded from memory to registers 2 and 3. The two patterns are applied to the adder with an ADD operation. The same sequence is applied for the remaining patterns. Finally, this program sits in a loop to be run during the idle states of the system.

Further optimization is possible to the above program to reduce the memory access and thus to reduce the power consumption of the running program, Fig. 14.

Another consideration is that this program may not have the privileges to run while another program is running. So the scheduler should give the lowest priority to this program and run it when the system is idle. However, if it is decided that this program should run as a routine in response to an interrupt, then the context of interrupted process needs to be saved. In this case the program should push the registers onto the stack at the start of the routine and pop them off at the end of the routine to save the context of the interrupted program, Fig. 15.

```
# Stimulate the first pattern
l.movhi r2, 48CB
l.ori r2, r2, 3E67
l.movhi r3, D440
l.ori r3, r3, 7A4C
l.add r4, r2, r3
# Stimulate the second pattern
l.movhi r2, 2465
l.ori r2, r2, 9F2D
l.movhi r3, EA20
l.ori r3, r3, 3D3C
l.add r4, r2, r3
     .
     .
# Stimulate the eighth pattern
l.movhi r2, AC8C
l.ori r2, r2, 0318
l.movhi r3, A4DC
l.ori r3, r3, 938D
l.add r4, r2, r3
l.rfe # Return From Exception
```

Fig. 14. Optimized balancing program.

```
# Push r2,r3,r4 onto the stack
push {r2-r4}
# Stimulate the first pattern
l.movhi r2, 48CB
l.ori r2, r2, 3E67
l.movhi r3, D440
l.ori r3, r3, 7A4C
l.add r4, r2, r3
# Stimulate the second pattern
l.movhi r2, 2465
l.ori r2, r2, 9F2D
l.movhi r3, EA20
l.ori r3, r3, 3D3C
l.add r4, r2, r3
     .
     .
# Stimulate the eighth pattern
l.movhi r2, AC8C
l.ori r2, r2, 0318
l.movhi r3, A4DC
l.ori r3, r3, 938D
l.add r4, r2, r3
# Pop r2,r3,r4 and the program counter(
    pc) from the stack, then branch to
    the new pc.
pop {r2-r4,pc}
```

Fig. 15. Optimized balancing program for saving the context of the interrupted program.

5. Evaluation and discussion

To evaluate the effect of running the balancing program and how the signal probability will be affected on the critical path, we ran the balancing program along with different benchmarks and varied the percentages of the running time of the balancing from 10% to 50%. To compare results, we calculated the number of stressed nodes as follows:

$$\text{Percentage of stressed nodes} = \frac{\text{\# stressed nodes}}{\text{critical path nodes}}$$

where the stressed nodes are those critical path nodes that have an SP (0) greater than 75% or less than 25%. As would be expected, to obtain a balanced state for the stressed nodes requires the balancing program to run for 50% of the time. Needless to say, it is not always possible to have this idle time or to add redundant time for the purpose of relaxing BTI. Fig. 16 shows the effect on the stressed nodes of running the BTI balancing program for different percentages of the overall time.

Fig. 17 shows the effect of running the BTI balancing program for different times on the percentages of the stressed nodes in critical paths of the OpenRISC processor. The results show that balancing one critical path will balance other near-critical paths as they share nodes with the first path. On the other hand, if the near-critical paths do not share many nodes with the targeted critical path, it is possible to apply balancing patterns in the same way for the first critical path independently of the other paths. Although balancing signal probabilities would work with embedded systems that run specific applications, it is also possible to use the technique for a general-purpose processor. Fig. 18 shows how the percentages of stressed nodes on the first critical path of the OpenRISC processor reduce when executing the balancing program along with a different program from the MiBENCH benchmarks.

Next, to verify that the balancing program will reduce the degradation in the path delay of the processor, we simulated the adder using HSPICE and modelled NBTI using the MOSRA Level 3. We stimulated the circuit with two cases:

- CASE A (Normal Stressed Mode): Stress patterns with the equivalent

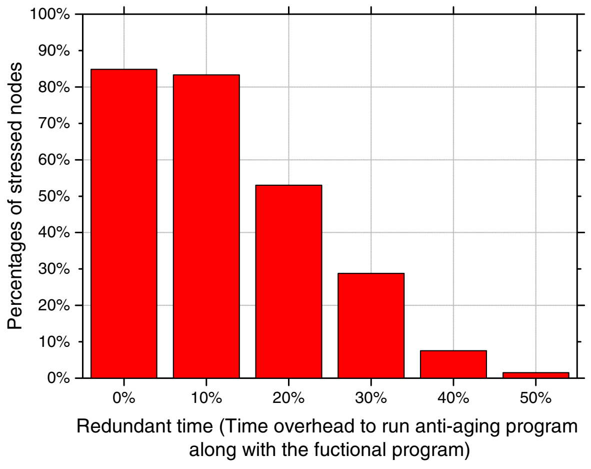


Fig. 16. The effect on the stressed node percentage of running the BTI balancing program along with a "hello world" program with different run times of the balancing program.

signal probabilities of the Hello World program.

- CASE B (Balanced Mode): Balanced stressed nodes (i.e. the nodes in the critical path having an SP(0) around 50% by running the anti-ageing program along with the Normal Stressed Mode program).

The results show that running the balancing program would decrease the path delay by 20.24% compared with normal operation and double the expected lifetime as shown in Fig. 19

5.1. Discussion

In our analysis, we expect, for example, 11% degradation in six years as can be seen in Fig. 19, so a simple solution could be guardbanding. Guardbanding is inevitable, not only for ageing but also for PVT variations. However, adding more guardbanding would negate the advantage of using a smaller technology size. So we have to find an active protective approach as well as estimating, or sensing and reacting to degradations.

The idea of this work is to utilise the short idle periods in a processor, [6]. These are used to reverse the BTI stress rather than running empty loops. In our case study, the OpenRISC processor, the critical paths are in the adder and we can propagate patterns simply by loading the patterns into a register and executing an addition operation. This program should replace the idle task and should be executed whenever the operating system tries to schedule the idle task. In general, if the timing critical component is not the adder, then we have to replace the operation accordingly. If the critical paths are not controllable at the instruction level (e.g. in a control unit that may have many flip-flops) then we need an architectural solution rather than a software solution to propagate the patterns and currently we are working on this issue.

We also need to consider how process variations could affect the critical path ranking. If we get this wrong, we might heal a non-critical path and leave the real critical path unaffected. For this reason, we have to consider not only the critical path but also the nearly critical paths that could become critical with PVT and time-dependent variations. In our case study, we have predefined ($\theta + \Delta\delta_{pv}$) to be 20% of the maximum path delay at time zero (δ_0), as described in Section 3.1, which covers the first 100 critical paths. However, we found that the first 100 critical paths share more than 92% of the cells with the most critical path and in this case balancing the most critical path also includes the 92% shared with the nearly critical paths. However, if the nearly critical paths do not share a big percentage of their cells with the first path, then we have to consider every single path in our analysis and generate patterns for them to balance signal probabilities in parallel. So, even with process variations, this technique would target the nearly-critical paths. If the nearly-critical paths do not share cells with the most critical path, it is important to define a threshold that considers the process and ageing variation contributions and to control these paths in parallel.

Finally, running a program to balance the BTI stress raises other design issues. Time overheads and power consumption need to be optimized by reducing the memory access or by using a program that has only immediate mode operands, as memory accesses increase the power consumption. Another issue that needs to be considered is when and for how long the program needs to run. The obvious answer is during the idle time of the processor but also we need to consider the operating system actions during the idle state.

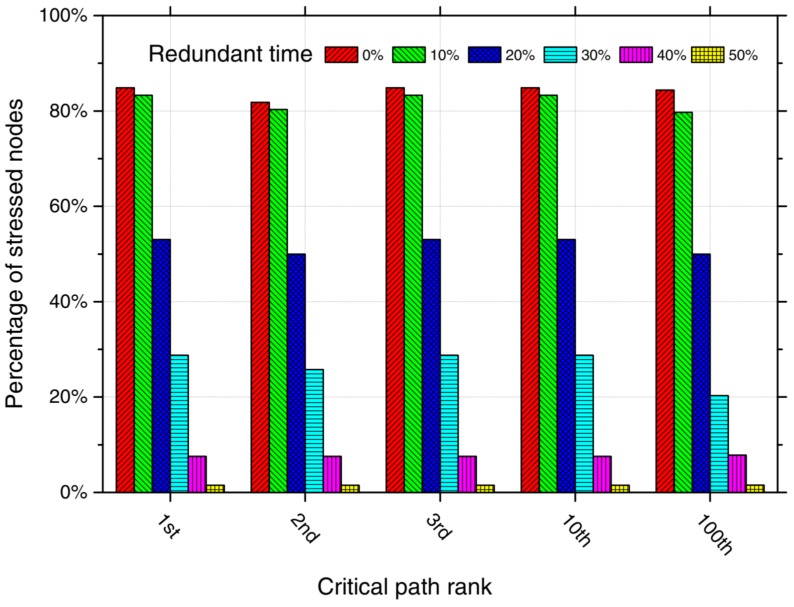


Fig. 17. The effect of running the BTI balancing program along with a "Hello World" program on the stressed node percentage on different critical paths of the OpenRISC Processor with various run times of the balancing (anti-ageing) program.

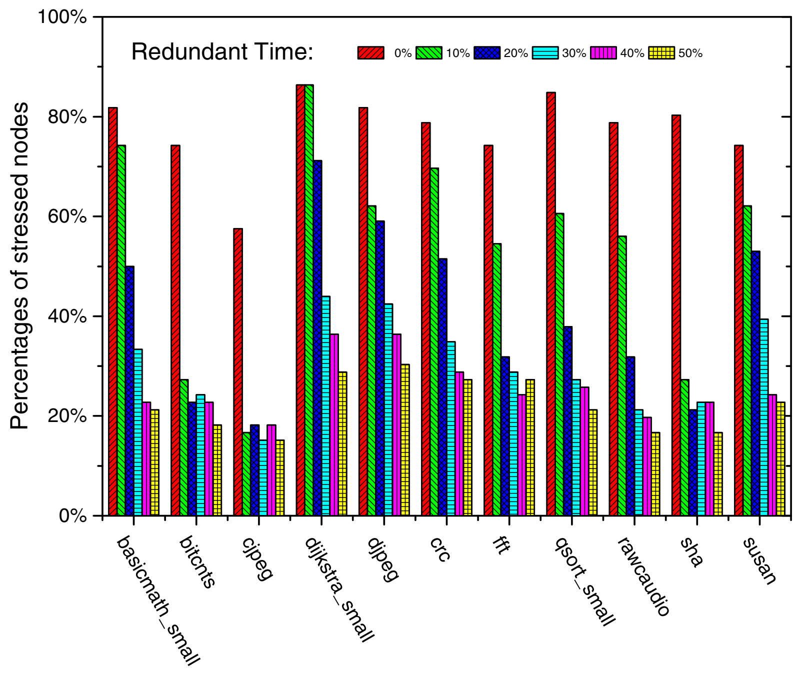


Fig. 18. The effect of running BTI balancing program along with a different program from MiBENCH benchmarks on the stressed node percentage on the first critical path of the OpenRISC Processor with different run times of the balancing program.

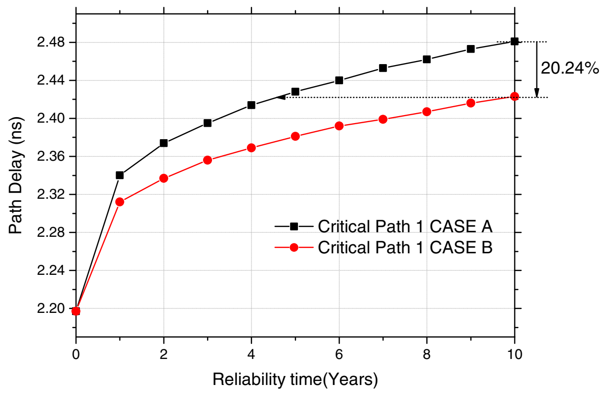


Fig. 19. The advantage of running the anti-ageing program (CASE B) on the path delay of OpenRISC processor.

6. Conclusion

Application-specific high-level ageing analysis has been done to find a technique for CMOS ageing mitigation. In this work, the stress probability has been found at the application level down to the gate level. A cross-layer mitigation technique is proposed to apply stress-relaxing patterns to the critical paths of a functional unit of a processor during idle times. This paper presents a two phase technique to mitigate the BTI ageing effects. The first phase generates balancing patterns. The second phase applies these patterns by executing a program to balance the stress on the critical paths of the embedded systems to alleviate BTI effects instead of running an empty process idle task. In future work, we will apply this technique as an architectural solution to control the paths in the non-software-controllable units of the processor. Also, we will apply stress balancing in multiprocessors or many-processors systems. The operating system scheduler of these systems will have a higher opportunity to run anti-ageing programs by assigning an anti-ageing process to a processor in an idle state, concurrently with other processors that are running user or system tasks.

Acknowledgements

This work was supported by the Higher Committee of Education Development in Iraq (HCEDIraq) as a PhD scholarship program.

References

[1] M. Kamal, A. Afzali-Kusha, S. Safari, M. Pedram, Design of NBTI-resilient extensible processors, Integr. VLSI J. 49 (2015) 22–34 http://linkinghub.elsevier.com/retrieve/pii/S016792601400087Xhttp://dx.doi.org/10.1016/j.vlsi.2014.12.001.
[2] J. Li, M. Seok, Robust and in-situ self-testing technique for monitoring device aging effects in pipeline circuits, 2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC), 2014, pp. 1–6 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6881529http://dx.doi.org/10.1109/DAC.2014.6881529.
[3] K.-C. Wu, D. Marculescu, Joint Logic Restructuring and Pin Reordering against NBTI-Induced Performance Degradation, Proceedings of the Conference on Design, Automation and Test in Europe, European Design and Automation Association, 2009, pp. 75–80.
[4] Y. Wang, H. Luo, K. He, R. Luo, H. Yang, Y. Xie, Temperature-aware NBTI modeling and the impact of standby leakage reduction techniques on circuit performance degradation, IEEE Trans. Dependable Secure Comput. 8 (5) (2011) 756–769.
[5] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, H. Yang, On the efficacy of input vector control to mitigate NBTI effects and leakage power, Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009, 2009, pp. 19–26, , http://dx.doi.org/10.1109/ISQED.2009.4810264.
[6] M. Arora, S. Manne, I. Paul, N. Jayasena, D.M. Tullsen, Understanding Idle Behavior and Power Gating Mechanisms in the Context of Modern Benchmarks on CPU-GPU Integrated Systems, 2015 IEEE 21St International Symposium on High Performance Computer Architecture (HPCA), IEEE, 2015, pp. 366–377.
[7] S. Di Carlo, M. Gaudesi, E. Sanchez, M. Sonza Reorda, A Functional Approach for Testing the Reorder Buffer Memory, J. Electron. Test. 30 (4) (2014) 469–481, http://dx.doi.org/10.1007/s10836-014-5461-9.
[8] T. Grasser, Bias Temperature Instability for Devices and Circuits, Springer Science & Business Media, 2013, p. 6 chap. 1.
[9] S. Mahapatra, Fundamentals of Bias Temperature Instability in MOS Transistors: Characterization Methods, Process and Materials Impact, DC and AC Modeling, vol. 52, Springer, 2015.
[10] A. Rahman, P. Bai, G. Curello, J. Hicks, C.-H. Jan, M. Jamil, J. Park, K. Phoa, M.S. Rahman, C.-Y. Tsai, et al., Reliability Studies of a 22Nm Soc Platform Technology Featuring 3-D Tri-Gate, Optimized for Ultra Low Power, High Performance and High Density Application, Reliability Physics Symposium (IRPS), 2013 IEEE International, IEEE, 2013, pp. I–2.

[11] B. Tudor, J. Wang, W. Liu, H. Elhak, MOS Device aging analysis with HSPICE and customism, Synopsys, White Pap. (2011).
[12] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, S. Vrudhula, Predictive Modeling of the NBTI Effect for Reliable Design, Custom Integrated Circuits Conference, 2006. CICC'06. IEEE, IEEE, 2006, pp. 189–192.
[13] B. Tudor, J. Wang, Z. Chen, R. Tan, W. Liu, F. Lee, An Accurate and Scalable MOSFET Aging Model for Circuit Simulation, Quality Electronic Design (ISQED), 2011 12Th International Symposium On, IEEE, 2011, pp. 1–4.
[14] W. Wang, V. Reddy, A.T. Krishnan, R. Vattikonda, S. Krishnan, Y. Cao, An Integrated Modeling Paradigm of Circuit Reliability for 65 Nm Cmos Technology, Custom Integrated Circuits Conference, 2007. CICC'07. IEEE, IEEE, 2007, pp. 511–514.
[15] J.B. Velamala, K.B. Sutaria, T. Sato, Y. Cao, Aging Statistics Based on Trapping/Detrapping: Silicon Evidence, Modeling and Long-Term Prediction, Reliability Physics Symposium (IRPS), 2012 IEEE International, IEEE, 2012, pp. 2F–2.
[16] G. Jerke, A.B. Kahng, Mission Profile Aware IC Design: a Case Study, Proceedings of the Conference on Design, Automation & Test in Europe, European Design and Automation Association, 2014, p. 64.
[17] W. Wang, Z. Wei, S. Yang, Y. Cao, An efficient method to identify critical gates under circuit aging, 2007 IEEE/ACM International Conference on Computer-Aided Design, 2007, pp. 735–740 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4397353http://dx.doi.org/10.1109/ICCAD.2007.4397353.
[18] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, H. Yang, On the efficacy of input vector control to mitigate NBTI effects and leakage power, Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009, 2009, pp. 19–26, , http://dx.doi.org/10.1109/ISQED.2009.4810264.
[19] S. Jin, Y. Han, L. Zhang, H. Li, X. Li, G. Yan, M-IVC: Using Multiple Input Vectors to Minimize Aging-Induced Delay, 2009 Asian Test Symposium, IEEE, 2009, pp. 437–442.
[20] L. Zhang, R.P. Dick, Scheduled Voltage Scaling for Increasing Lifetime in the Presence of NBTI, Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific, IEEE, 2009, pp. 492–497.
[21] M. Omana, D. Rossi, N. Bosio, C. Metra, Low cost NBTI degradation detection and masking approaches, IEEE Trans. Comput. 62 (3) (2013) 496–509.
[22] P.F. Butzen, V. Dal Bem, A.I. Reis, R.P. Ribas, Transistor network restructuring against NBTI degradation, Microelectron. Reliab. 50 (9) (2010) 1298–1303.
[23] P. Singh, E. Karl, D. Sylvester, D. Blaauw, Dynamic NBTI management using a 45 nm multi-degradation sensor, IEEE Trans. Circuits Syst. Regul. Pap. 58 (9) (2011) 2026–2037, http://dx.doi.org/10.1109/TCSI.2011.2163894.
[24] V. Huard, F. Cacho, F. Giner, M. Saliva, a. Benhassain, D. Patel, N. Torres, S. Naudet, a. Jain, C. Parthasarathy, Adaptive Wearout Management with in-situ aging monitors, 2014 IEEE International Reliability Physics Symposium, 2014 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6861106 http://dx.doi.org/10.1109/IRPS.2014.6861106 6B.4.1–6B.4.11.
[25] W. Wang, V. Reddy, A.T. Krishnan, R. Vattikonda, S. Krishnan, Y. Cao, Compact modeling and simulation of circuit reliability for 65-nm CMOS technology, IEEE Trans. Device Mater. Reliab. 7 (4) (2007) 509–517.
[26] F. Firouzi, S. Kiamehr, M.B. Tahoori, NBTI mitigation by optimized NOP assignment and insertion, 2012 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012, pp. 218–223 http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6176465http://dx.doi.org/10.1109/DATE.2012.6176465.
[27] M.R. Guthaus, J.S. Ringenberg, D. Ernst, T.M. Austin, T. Mudge, R.B. Brown, Mibench: A Free, Commercially Representative Embedded Benchmark Suite, Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop On, IEEE, 2001, pp. 3–14.
[28] N. Binkert, B. Beckmann, G. Black, S.K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D.R. Hower, T. Krishna, S. Sardashti, The gem5 simulator, ACM SIGARCH Comput. Architect. News 39 (2) (2011) 1–7.
[29] N. Karimi, A.K. Kanuparthi, X. Wang, O. Sinanoglu, R. Karri, MAGIC: malicious aging in circuits/cores, ACM Trans. Archit. Code Optim. (TACO) 12 (1) (2015) 5.
[30] D. Lorenz, G. Georgakos, U. Schlichtmann, Aging Analysis of Circuit Timing considering NBTI and HCI, On-Line Testing Symposium, 2009. IOLTS 2009. 15Th IEEE International, IEEE, 2009, pp. 3–8.
[31] J. Abella, X. Vera, A. González, Penelope: The NBTI-aware processor, Proc. Annu. Int. Symp. Microarchitecture, MICRO (2007) 85–96, http://dx.doi.org/10.1109/MICRO.2007.11.

An Application-Specific NBTI Ageing Analysis Method

Haider Muhi Abbas, Mark Zwolinski and Basel Halak
Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
Email:{hma1g14,mz,bh9}@ecs.soton.ac.uk

***Abstract*—There is growing concern about time-dependent performance variations of CMOS devices due to ageing-induced delay degradation. One of the main causes of ageing is Negative Bias Temperature Instability (NBTI). Existing models which predict the impact of NBTI on overall system performance assume a generic stress-recovery ratio of input signals of 50%. Such an assumption can cause misleading predictions about how a circuit's performance will degrade over time and more importantly which parts of the system will be most affected. This work develops a novel NBTI ageing analysis which is based on accurate calculations of the stress-recovery ratios for application-specific systems. The proposed method is employed to predict the ageing of an ARM processor synthesised to 90nm technology. Our results show the proposed ageing analysis techniques can significantly reduce prediction errors (e.g. 39% for one of the critical paths) compared to the generic models, it can also identify more accurately the parts of the system which are most vulnerable to ageing.**

Keywords*—*Negative Bias Temperature Instability (NBTI); Signal Probability; Application-Specific Systems

I. Introduction

Ageing or time-dependent variations are caused by wear-out mechanisms such that Negative Bias Temperature Instability (NBTI), Hot Carrier Injection (HCI), Electromigration (EM), and Time-Dependent Dielectric Breakdown (TDDB). These phenomena result in performance degradation and reliability problems. However, NBTI is the most important ageing mechanism in Nano-scale technology devices [1], [2].

NBTI can lead to an increase in threshold voltage over time. The first observations of a threshold voltage instability were made in the 1960s in MOS transistors, in which it was found that both bias and temperature affect the threshold voltage. From that time and until the year 2000, when introduction of nitrogen atoms into the oxide was achieved, Bias Temperature Instability (BTI) remained unimportant phenomenon. Since then, negative BTI has become a more important concern, both in academia and industry [3].

In general, many mitigation techniques for NBTI have been proposed at different abstraction levels. At high levels of the abstraction hierarchy, a variety of protection mechanisms may increase reliability even further when moving to the software level [4]. Existing models which predict the impact of NBTI on overall system performance assume a generic signal probability of input signals of 50%. Such an assumption can cause misleading predictions about how the circuit's performance is going to degrade in time and, more importantly, which parts of the system will be most affected.

The aim of this work is to develop a more reliable ageing prediction model which takes into account the actual signal probability of the system. To achieve this we developed a tool to derive the actual stress-recovery ratio of each logic gate from application data. To the best of our knowledge this is the first ageing analysis model which is based on accurate calculation of the duty cycles derived from the application level.

The structure of the remainder of the paper is as follows. A brief review of the stress and recovery of NBTI, NBTI mitigation techniques and NBTI modelling are presented in Section II. Analysis of signal probability in identifying NBTI critical paths is described in Section III. Experimental setup and results are given in section IV. These results are analyzed in Section V and finally Section VI concludes the paper.

II. Background And Related Work

A. NBTI Stress-Recovery sources

Basically, NBTI is generated due to a change in the physical characteristics of a transistor by generating interface traps at the channel and dielectric. The gates of PMOS transistors are negatively biased with respect to the source (i.e. $V_{gs} = -V_{dd}$). Generally, BTI consists of two different phases:

1) Stress: Some interface traps are generated at the interface of substrate/gate oxide layers due to applying electrical stress (i.e. negative bias for PMOS) that leads to breaking of some of the SiH or SiO bonds as shown in Fig. 1(a). Consequently, the threshold voltage of the transistor increases over the time.

2) Relaxation/Recovery: some of the generated interface traps are removed from the interface as shown in Fig. 1(b). However, the relaxation phase cannot completely compensate for the effect of the stress phase and therefore the overall effect of NBTI is a degradation in the threshold voltage of the transistor. The amount of this degradation depends on the ratio between the stress period and the total period (duty cycle).

B. NBTI Mitigation Techniques

In this section, the literature of the mitigation techniques used for ageing is presented. A feasible solution to reliability problems including ageing is to eliminate or even to reduce the design uncertainties that exist in current design technologies. However, in practice, there is more than one contributor to

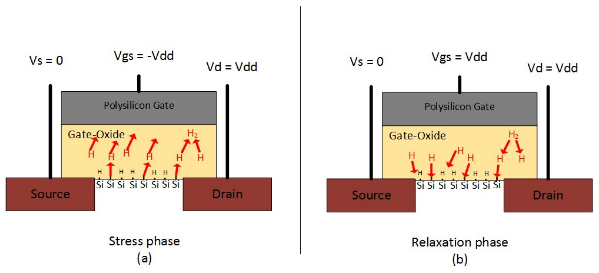


Fig. 1. Representation view of (a) Stress Phase: Si-H bonds breaking, H and H_2 diffusions towards poly gate (b) Relaxation Phase: H and H_2 diffusions towards the Silicon/Gate-Oxide interface.

these uncertainties including EDA tool limitations and complex environmental stress conditions [5]. Another solution is conservative design under the worst-case scenarios at the design stage. However, circuits do not always run at the worst-case condition and such an over-designed approach is extremely costly in terms of power and area. At gate level, NBTI manifests itself as time-dependent gate delay and finally leads to timing violations. Adding some margins to the critical path is not a solution because the critical path at time zero may not still be the critical path after some years due to ageing, as we show in the following section of this paper. Potential Critical Paths (PCPs) or NBTI Critical Paths take into account the effect of NBTI in static path analysis. In [6], critical gates (gates within PCPs) are identified and optimization methodologies (i.e. gate resizing or reducing temperature) proposed for these critical gates. However, NBTI has a dependence on dynamic operation, such as supply voltage, spatial or temporal temperature and signal probability and these parameters vary dynamically from gate to gate. Another solution, proposed in [7], uses signal probability to restructure the logic gates and arrival times of the input signal to reorder the pins. However, signal probabilities are assumed to be 50% for input signals and for larger systems, signal probability is dynamic and based on the application. Another technique to control NBTI is to control the signal probability using the Input Vector Control (IVC) technique. However, both NBTI and leakage power have a cross dependency on input patterns [8], [9].

C. Long-Term NBTI Model

At device level, many models have been proposed as predictive models to simulate the real degradation in transistor performance. Until now, there is no universally accepted theory or model, therefore all information is based on accepted experimental results. Reaction-diffusion (RD) is one of the most prevalent NBTI models in the literature [6], [8], [10], [11]. Many developments have been proposed to the model to increase the accuracy of the model. The following formulae describe the NBTI-induced degradation threshold voltage using long-term RD model:

$$\Delta V_{th} = \left(\sqrt{K_v^2\, T_{clk}\, (\frac{\alpha}{1-\beta_t^{\frac{1}{2n}}})}\right)^{2n} \tag{1}$$

$$K_v = \left(\frac{qt_{ox}}{\epsilon_{ox}}\right)^3 K^2 C_{ox}(V_{gs}-V_{th})\sqrt{C}exp\left(\frac{2E_{ox}}{E_0}\right) \tag{2}$$

$$\beta_t = 1 - \frac{2\xi_1 t_{ox} + \sqrt{\xi_2 C(1-\alpha)T_{clk}}}{2t_{ox}+\sqrt{Ct}} \tag{3}$$

α is the signal probability and it reflects the fraction of time spent in the stress state over a period of time, and T_{clk} reflects the frequency of stress and recovery phases. The rest of the parameters are described in detail in [11]. We simulate the 90nm technology node using MATLAB with the following parameters: initial $V_{th} = 0.276V, V_{gs} = 1.2V, T = 350K, t_{ox} = 2.15nm$. Fig. 2. shows how duty cycles have a big impact on V_{th} degradation.

- For duty cycle (α)=0.1, V_{th} increases by 130mV (increases 47% from the initial V_{th})
- For duty cycle (α)=0.5, V_{th} increases by 180mV (65%)
- For duty cycle (α)=0.9, V_{th} increases by 210mV (76%)

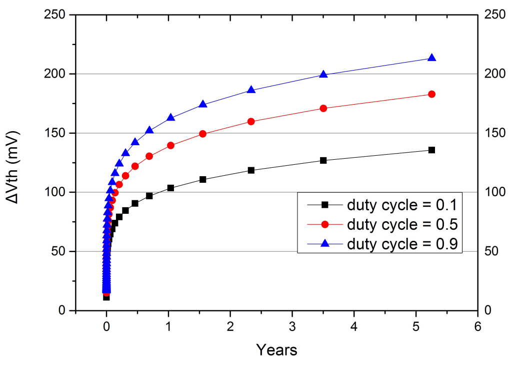


Fig. 2. V_{th} degradation for different duty cycles.

III. SIGNAL-PROBABILITY-NBTI AGEING ANALYSIS

Having information about the signal probability from the application level could help in identifying NBTI critical paths. Fig. 3. shows the proposed approach to identify NBTI critical paths which can be summarized in four steps:

1) Consider a specific processor, then run different benchmarks on the simulated processor and obtain a VCD (value Change Dump) file that represents the application activity at gate level.
2) Obtain the signal probabilities of the nets being stressed with logic zero (SP(0)) from the VCD file.
3) Based on the predefined lifetime of the processor and SP(0), calculate the degradation value of V_{th} for a range of signal probabilities using the NBTI model.
4) Calculate the new delays for paths and identify the worst degradation values that will represent the NBTI critical paths.

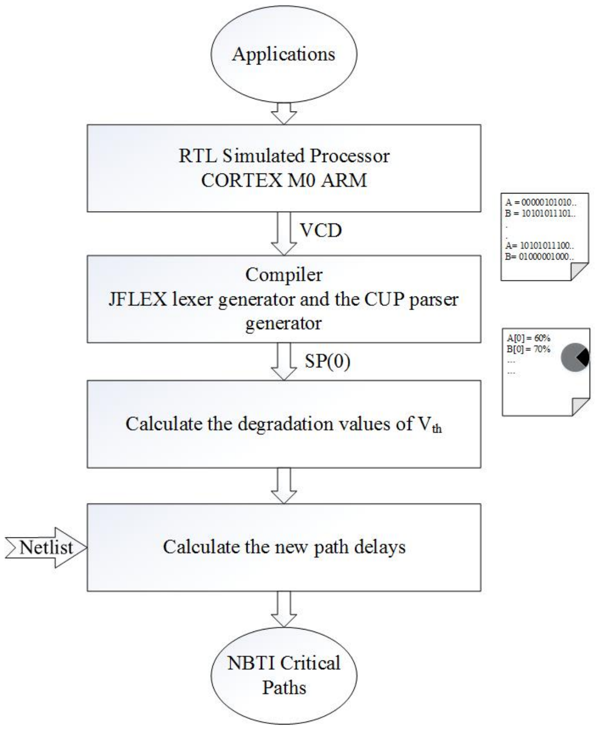


Fig. 3. NBTI-critical-path identification steps.

IV. Experimental Setup And Simulation Results

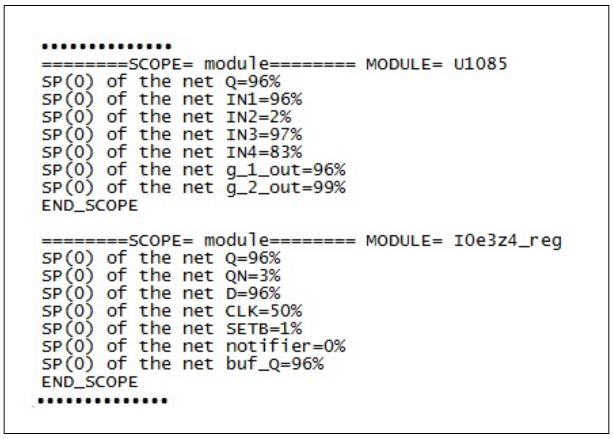
A. VCD to SP(0) Experimental Setup

To obtain the signal probabilities (duty cycles) of the nets for a specific application, we started from the VCD file to obtain SP(0). The VCD file implicitly contains both switching activity that is used to estimate the dynamic power at the design phase and the signal probability that we use to estimate the NBTI effect on performance degradation. So, we built a compiler using open source software (JFLEX and CUP) available in [12] to generate a file contains the SP(0) for all the nets of the processor (see Fig. 4. and Fig. 5.). it can be seen from Fig. 5. that 29% of the nets are mostly stressed while 38% are mostly unstressed. Furthermore, this is not the average activity of the system. Embedded systems that run a specific applications or usually run different applications from the same domain (e.g. Internet of Things (IoT)) have a limited number of applications. VCD file for these applications can be extracted from many EDA tools (e.g. Modelsim and VCS) and then this VCD file will represent the average application activity that will contain the average probabilities of the system's signals.

B. Ageing with real signal probabilites SP(0)

We used a CORTEX M0 ARM processor as a case study to identify critical paths and to compare results. First, we simulated part of the critical path as shown in Fig. 6. to see how signal probability can make a difference to ageing effects in path delay. We considered three cases:

- CASE A – delay of new circuit;



```
..............
========SCOPE= module======== MODULE= U1085
SP(0) of the net Q=96%
SP(0) of the net IN1=96%
SP(0) of the net IN2=2%
SP(0) of the net IN3=97%
SP(0) of the net IN4=83%
SP(0) of the net g_1_out=96%
SP(0) of the net g_2_out=99%
END_SCOPE

========SCOPE= module======== MODULE= I0e3z4_reg
SP(0) of the net Q=96%
SP(0) of the net QN=3%
SP(0) of the net D=96%
SP(0) of the net CLK=50%
SP(0) of the net SETB=1%
SP(0) of the net notifier=0%
SP(0) of the net buf_Q=96%
END_SCOPE
..............
```

Fig. 4. Part of the signal probabilities file.

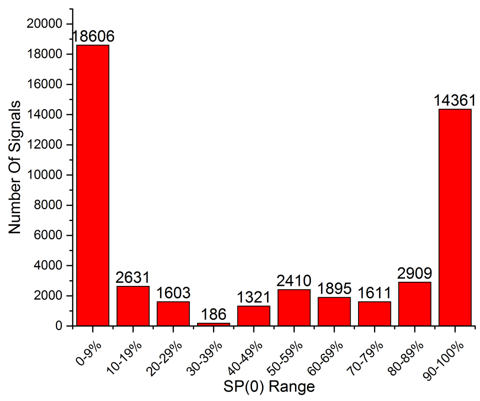


Fig. 5. The SP(0) of CORTEX M0 ARM for Hello World program.

- CASE B – NBTI-induced delay under assumption of 50% duty cycles at the primary inputs;
- CASE C – NBTI-induced delay with duty cycles of the all signals equal to SP(0) from the above analysis.

To propagate the SP(0) values to to transistor level, we used the long-term RD NBTI model to calculate the degradation values of V_{th}. From this, we calculate the path delay using HSPICE simulation for 90nm technology node from Synopsys. The results show 21.33% difference between CASE B and CASE C, and that CASE C is more optimistic than CASE B (see TABLE I).

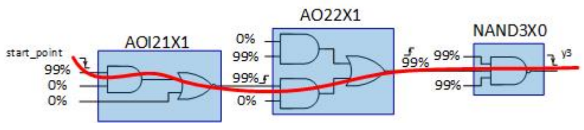


Fig. 6. Critical path subcircuit.

TABLE I. Subcircuit path delay after five years

CASE A	CASE B (α=0.5)	CASE C (α = SP(0))	error
1.1763E-10	1.6013E-10 (36.13% increased)	1.3503E-10 (14.79% increased)	-21.33%

C. Ageing critical paths

Considering the signal probabilities that come from simulating an application, rather than the default assumption, to measure the path delay of a critical path sub circuit, makes a significant difference to the critical path delay of the system. In the same way, we simulated the path delay for the two most critical paths using the three above mentioned cases. For the first critical path as shown in the TABLE II, the degradation delay for CASE B has increased 31.66% and 28.9% for CASE C. The results show that CASE C is more optimistic than CASE B with the same trend for the first three instances on the critical path. For the second critical path, the degradation delay for the CASE B has increased 32.66% and 72.2% for the CASE C. From these results, the first critical path is no longer the most critical path after five years and the second or other path may now be considered to be the most critical path.

TABLE II. Crtical path delay after five years

Critical Path Number	CASE A	CASE B (α=0.5)	CASE C (α = SP(0))	error
1	1.5687E-09	2.3463E-09 (49.56% increased)	2.0221E-09 (28.9% increased)	-20.66%
2	1.5569E-09	2.0655E-09 (32.66% increased)	2.6811E-09 (72.2% increased)	39.54%

V. Discussion

Estimating path delay for all the possible paths of circuit is not feasible for complex systems. So, classifying paths with the potential to be critical paths is possible if the maximum degradation delay is known for the predefined lifetime of a system. This may make some paths less critical. After reducing the number of paths that could potentially be critical, further estimation can be applied using the proposed work for identifying NBTI-critical path by considering the real activity of the system. As signal probability is not the only observable parameter from the application level to transistor level, temperature is possibly observed as a function of switching activity of the transistor. However, the temperature is environment dependent and spatial deviation of the temperature in a small area is normally small. So, considering the signal probability derived at the application level will not estimate the degradation delay with 100% accuracy but it will increase the accuracy of the estimation. Further optimisation needs to be considered: for example during the design phase, by restructuring the gates with high signal probabilities. Delay sensors might also be inserted in a limited number of paths to consider other sources of ageing that are difficult to observe during the design phase (e.g. Temperature).

VI. Conclusion

This work has introduced an analysis of ageing effects by considering the signal probabilities derived from the application level in an automated way. Our work demonstrates that NBTI-critical paths can be identified using the proposed approach, which will help to accurately place ageing sensors. Also this work can help to improve the design offline by pin reordering or gate restructuring for gates that have high stress. Future work will automate the process of finding the age-induced critical paths considering both NBTI and HCI.

References

[1] S. Kiamehr, F. Firouzi, M. Ebrahimi, and M. B. Tahoori, "Aging-aware standard cell library design," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, vol. 3, pp. 1–4, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6800475

[2] W. Dweik, M. Annavaram, and M. Dubois, "Reliability-Aware Exceptions: Tolerating intermittent faults in microprocessor array structures," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, pp. 1–6, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6800315

[3] T. Grasser, *Bias Temperature Instability for Devices and Circuits*. Springer Science & Business Media, 2014.

[4] U. Schlichtmann, "Connecting different worldsTechnology abstraction for reliability-aware design and Test," *Design, Automation ...*, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6800466

[5] G. Jerke and A. Kahng, "Mission profile aware IC designA case study," *Design, Automation and Test in Europe ...*, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6800278

[6] W. Wang, Z. Wei, S. Yang, and Y. Cao, "An efficient method to identify critical gates under circuit aging," *2007 IEEE/ACM International Conference on Computer-Aided Design*, pp. 735–740, Nov. 2007. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4397353

[7] D. Marculescu, "Joint logic restructuring and pin reordering against NBTI-induced performance degradation," *2009 Design, Automation & Test in Europe Conference & Exhibition*, pp. 75–80, Apr. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5090636

[8] Y. Wang, H. Luo, K. He, R. Luo, H. Yang, and Y. Xie, "Temperature-aware NBTI modeling and the impact of standby leakage reduction techniques on circuit performance degradation," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 2005, pp. 756–769, 2011.

[9] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, and H. Yang, "On the efficacy of input vector control to mitigate NBTI effects and leakage power," *Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009*, pp. 19–26, 2009.

[10] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The impact of NBTI effect on combinational circuit: Modeling, simulation, and analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 2, pp. 173–183, 2010.

[11] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, "Predictive modeling of the NBTI effect for reliable design," *Proceedings of the Custom Integrated Circuits Conference*, no. Cicc, pp. 189–192, 2006.

[12] G. Klein, "Jflex users manual," *Available on-line at www. jflex. de. Accessed January*, 2015.

Static Aging Analysis Using 3-Dimensional Delay Library

Haider Muhi Abbas, Mark Zwolinski and Basel Halak
Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
Email:{hma1g14,mz,bh9}@ecs.soton.ac.uk

***Abstract*—The growing concern about time-dependent performance variations of CMOS devices due to aging-induced delay degradation has increased with shrinking technology dimensions of the devices . One of the main causes of aging is Negative Bias Temperature Instability (NBTI). Modeling NBTI-induced delay at gate level depends on the real stress activity of gate inputs which are related to the workload applied from the higher level of abstraction (e.g. Application). Having estimated values about the degradation delays can make design stage to consider this issue as a design constrain and even to precisely allocate the online aging sensors. this paper propose a method to include the stress probability within technology library as three dimensional look-up tables for Static Timing Analysis(STA) process of the design as an approach named Static Aging Analysis(SAA). the purpose of this approach is instead of estimating only the timing delay at time zero, estimating NBTI-induced delays for predefined lifetime of the product.**

***Keywords*—Negative Bias Temperature Instability (NBTI); Signal Probability; Static Timing Analysis.**

I. Introduction

NBTI can lead to an increase in threshold voltage over time. The first observations of a threshold voltage instability were made in the 1960s in MOS transistors, in which it was found that both bias and temperature affect the threshold voltage. From that time and until the year 2000, when introduction of nitrogen atoms into the oxide was achieved, Bias Temperature Instability (BTI) remained unimportant phenomenon. Since then, negative BTI has become a more important concern, both in academia and industry [1].

In general, there are two possible approaches of aging mitigation techniques either proactive or reactive. Proactive approach works as an estimator for aging behaviour and always based on a model that describes how aging effects (NBTI, HCI and TDDB) are modeled by physicists in low level [2]–[4]. Usually this approach needs to take into account different contributors of time-dependent variations (e.g. signal probability, switching activity, temperature and supply voltage). Another approach is to monitor online the real behaviour of aging through delay sensors [5], [6]. The last approach is more precise and short-cutting the complexity of modeling the aging effects on the system level. However, the main problem of online sensors is area overheads and to moderate this drawback, limited numbers of nodes are needed to be monitored. In order to define the locations and how many sensors to be inserted, offline analysis of aging is inevitable.

This paper propose a method to inject the accurate calculation of the duty cycles derived from the application level to design new library targeting the aging prediction as a method called Static Aging Analysis(SAA). To achieve this we have used a tool to derive the actual stress-recovery ratio of each logic gate from application then build three-dimensional look-up tables for the NBTI-induced timing delays. To the best of our knowledge this is the first aging analysis method which is based on predefined stress-probability library.

The organization of the remainder of the paper is as follows. A concise review of the stress and recovery of NBTI, NBTI mitigation techniques and NBTI modeling are conferred in Section II. Analysis of three-dimensional delay library is presented in Section III. Experimental setup and results are given in section IV and finally Section V. concludes the paper.

II. Background And Related Work

A. NBTI Stress-Recovery sources

Basically, NBTI is generated due to a change in the physical characteristics of a transistor by generating interface traps at the channel and dielectric. The gates of PMOS transistors are negatively biased with respect to the source (i.e. $V_{gs} = -V_{dd}$). Generally, BTI consists of two different phases:

1) Stress: Some interface traps are generated at the interface of substrate/gate oxide layers due to applying electrical stress (i.e. negative bias for PMOS) that leads to breaking of some of the SiH or SiO bonds. Consequently, the threshold voltage of the transistor increases over the time.

2) Relaxation/Recovery: some of the generated interface traps are removed from the interface. However, the relaxation phase cannot completely compensate for the effect of the stress phase and therefore the overall effect of NBTI is a degradation in the threshold voltage of the transistor. The amount of this degradation depends on the ratio between the stress period and the total period (duty cycle).

B. NBTI Mitigation Techniques

In this section, the literature of the mitigation techniques used for aging is presented. A feasible solution to reliability problems including aging is to eliminate or even to reduce the design uncertainties that exist in current design technologies. However, in practice, there is more than one contributor to these uncertainties including EDA tool limitations and complex environmental stress conditions [7]. Another solution is conservative design under the worst-case scenarios at the design stage. However, circuits do not always run at the worst-case condition and such an over-designed approach is extremely costly in terms of power and area. At gate level, NBTI manifests itself as time-dependent gate delay and finally leads to timing violations. Adding some margins to the critical path is not a solution because the critical path at time zero may not still be the critical path after some years due to aging. Potential Critical Paths (PCPs) or NBTI Critical Paths take into account the effect of NBTI in static path analysis. In [8], critical gates (gates within PCPs) are identified and optimization methodologies (i.e. gate resizing or reducing temperature) proposed for these critical gates. However, NBTI has a dependence on dynamic operation, such as supply voltage, spatial or temporal temperature and signal probability and these parameters vary dynamically from gate to gate. Another solution, proposed in [9], uses signal probability to restructure the logic gates and arrival times of the input signal to reorder the pins. However, signal probabilities are assumed to be 50% for input signals and for larger systems, signal probability is dynamic and based on the application. Another technique to control NBTI is to control the signal probability using the Input Vector Control (IVC) technique. However, both NBTI and leakage power have a cross dependency on input patterns [4], [10].

C. Long-Term NBTI Model

At device level, many models have been proposed as predictive models to simulate the real degradation in transistor performance. Until now, there is no universally accepted theory or model, therefore all information is based on accepted experimental results. Reaction-diffusion (RD) is one of the most prevalent NBTI models in the literature [8], [4], [11], [3]. Many developments have been proposed to the model to increase the accuracy of the model. The following formulas describe the NBTI-induced degradation threshold voltage using long-term RD model:

$$\Delta V_{th} = \left(\sqrt{K_v^2 \, T_{clk} \, (\frac{\alpha}{1 - \beta_t^{\frac{1}{2n}}})} \right)^{2n} \tag{1}$$

$$K_v = \left(\frac{qt_{ox}}{\epsilon_{ox}}\right)^3 K^2 C_{ox}(V_{gs} - V_{th})\sqrt{C} exp\left(\frac{2E_{ox}}{E_0}\right) \tag{2}$$

$$\beta_t = 1 - \frac{2\xi_1 t_{ox} + \sqrt{\xi_2 C(1-\alpha)T_{clk}}}{2t_{ox} + \sqrt{Ct}} \tag{3}$$

α is the signal probability and it reflects the fraction of time spent in the stress state over a period of time, and T_{clk} reflects the frequency of stress and recovery phases. The rest of the parameters are described in detail in [3]. We simulate the 90nm technology node using MATLAB with the following parameters: initial $V_{th} = 0.276V, V_{gs} = 1.2V, T = 350K, t_{ox} = 2.15nm$. Fig. 2. shows how duty cycles have a big impact on V_{th} degradation.

- For duty cycle (α)=0.1, V_{th} increases by 130mV (increases 47% from the initial V_{th})
- For duty cycle (α)=0.5, V_{th} increases by 180mV (65%)
- For duty cycle (α)=0.9, V_{th} increases by 210mV (76%)

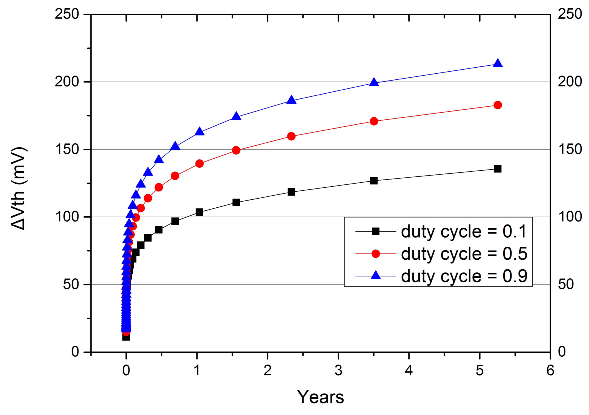


Fig. 1. V_{th} degradation for different duty cycles.

III. 3-Dimensional Delay Library

The standard library that used to estimate the timing and power at the synthesis stage based on pre-calculated look-up tables considering the input transition delay and output capacitance load as two-dimensional LUTs for each standard cell in the library. To generate NBTI induced library for the purpose of estimating the effect of NBTI for pre-defined lifetime of the product, three-dimensional LUT delay library could be built considering the Signal Probability SP(0) as the third dimension along with input transition and output load capacitance. Debatably, signal probability is not the only observable parameter from the application level to transistor level, temperature is possibly observed as a function of switching activity of the transistor. However, the temperature is environment dependent and the spatial deviation of the temperature in a small area is normally small. So, considering the signal probability derived at the application level will not estimate the degradation delay with 100% accuracy, but it will increase the accuracy of the estimation. Further optimization needs to be considered: for example, during the design phase, by restructuring the gates with high signal probabilities. Delay sensors might also be inserted in a limited number of paths to consider other sources of ageing that are difficult to observe during the design phase (e.g. Temperature).

IV. Experimental Setup And Simulation Results

A. Signal Probabilities Extraction

We extract signal probabilities (duty cycles) of the nets from different workloads using Mibench benchmark as workload on OpenRisc 1200 processor (see TABLE I and Fig. 3.).To obtain the duty cycles (signal probabilities SP(0)) of the nets for a specific application, we started from the VCD file to obtain SP(0). The VCD file implicitly contains both switching activity that is used to estimate the dynamic power at the design phase and the signal probability that we use to estimate the NBTI effect on performance degradation. So, we built a compiler using open source software (JFLEX and CUP) available in [12] to generate a file contains the SP(0) for all the nets of the processor. Getting VCD file for the gate level needs to simulate the post-synthesized processor. We synthesis OpenRisc using 90nm Technology from Synopses using limited number of netlists that have only one level of transistors to avoid getting hidden signal probabilities inside the multilevel cells. The results show that there is likely stress probability within the nets with different workloads and the signal probability highly related to the architecture rather than the workload as shown in the Fig. 3.).

TABLE I. Signal Probabilities Percentages for MiBench Workloads

	Automotive		Telecomm			Network	secuirty
SP(0) Range	qsort	susan	adpcm	crc	fft	dijkstra	sha
0-0.09	26%	20%	24%	20%	24%	20%	20%
0.1-	3%	3%	0%	3%	0%	3%	4%
0.2-	4%	3%	2%	2%	2%	3%	3%
0.3-	2%	2%	2%	2%	2%	1%	1%
0.4-	3%	3%	2%	2%	2%	3%	3%
0.5-	7%	5%	4%	5%	4%	4%	5%
0.6-	4%	2%	3%	3%	3%	2%	2%
0.7-	9%	6%	4%	6%	4%	3%	5%
0.8-	4%	6%	1%	7%	1%	8%	6%
0.9-1	39%	51%	58%	51%	58%	53%	51%

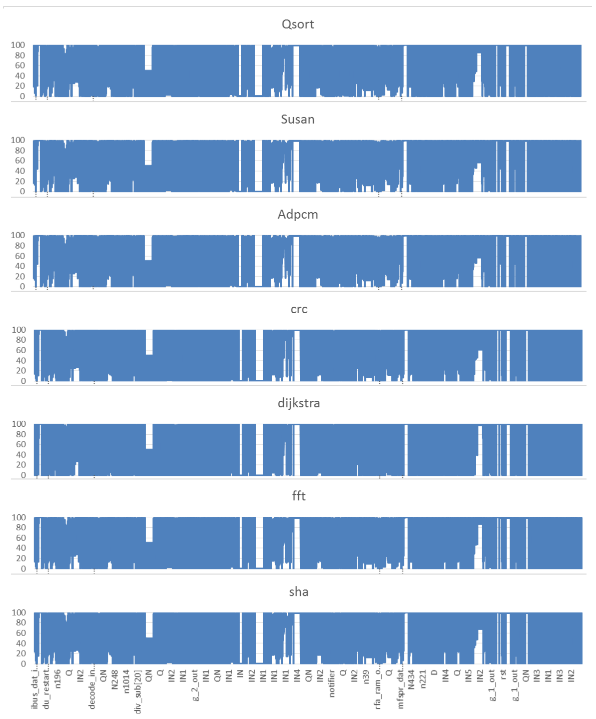


Fig. 2. Horizontal axises: Compressed information about the net names of OpenRisc Processor, vertical axises: Percentages of Signal Probabilities SP(0) for MiBench Workload.

B. Three-Dimensional LUT Generation

Basically, the library file contains four look-up tables, two for rising and falling propagation delays, and two for transition delays that used as an input parameter to calculate the propagation delay for the next cell. To induce signal probabilities SP(0) into the library, we used HSpice simulation to generate the timing delays. the figure Fig. 4. shows the output rising delays when the SP(0) barely stressed and the figure Fig. 5. shows the output rising delays when the SP(0) almost totally stressed. the results show that up to 81% difference between the the cases mentioned above. Choosing how many steps for SP(0) would define the complexity of the delay calculation inside the Static Aging Analyzer. So, some specific steps need to be calculated and any other intermediate values could be either taken the worst case or interpolated from two points.

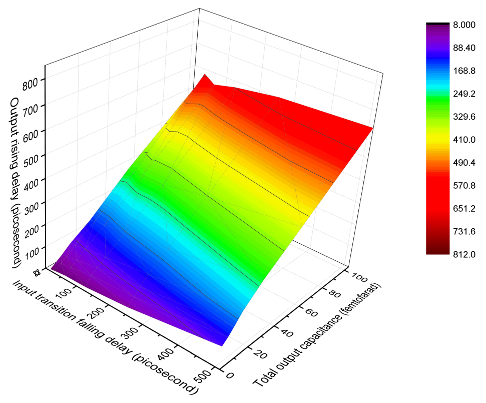


Fig. 3. Output Rising Delays when SP(0)=0.1.

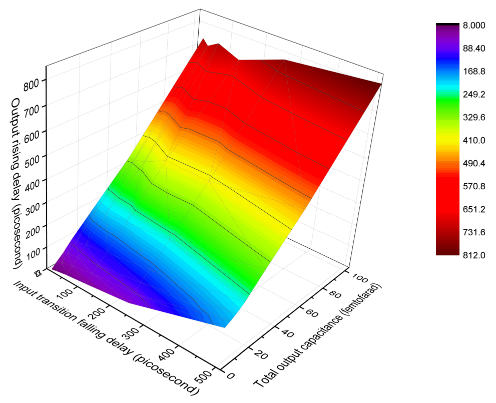


Fig. 4. Output Rising Delays when SP(0)=0.9.

V. Conclusion and Future Work

This work has proposed an approach to include the signal probabilities driven from the workload to the gate-level timing library. This work could make estimating NBTI-induced delay more feasible during the design and could help to abate the number of aging sensor and location. However, this approach required a logic synthesizer to read these three-dimensional LUT and may make optimization during cell mapping to reduce highly stressed signal in the critical path. Future work will consider the use this table by building this logic synthesizer to identify the most vulnerable part for aging in the processor.

References

[1] T. Grasser, *Bias Temperature Instability for Devices and Circuits*. Springer Science & Business Media, 2014.

[2] W. Wang and V. Reddy, "Compact modeling and simulation of circuit reliability for 65-nm CMOS technology," *…Materials Reliability, …*, vol. 7, no. 4, pp. 509–517, 2007. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4359941

[3] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, "Predictive modeling of the NBTI effect for reliable design," *Proceedings of the Custom Integrated Circuits Conference*, no. Cicc, pp. 189–192, 2006.

[4] Y. Wang, H. Luo, K. He, R. Luo, H. Yang, and Y. Xie, "Temperature-aware nbti modeling and the impact of standby leakage reduction techniques on circuit performance degradation," *Dependable and secure computing, IEEE transactions on*, vol. 8, no. 5, pp. 756–769, 2011.

[5] V. Huard, F. Cacho, F. Giner, M. Saliva, a. Benhassain, D. Patel, N. Torres, S. Naudet, a. Jain, and C. Parthasarathy, "Adaptive Wearout Management with in-situ aging monitors," *2014 IEEE International Reliability Physics Symposium*, pp. 6B.4.1–6B.4.11, Jun. 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6861106

[6] R. Baranowski, F. Firouzi, S. Kiamehr, C. Liu, M. Tahoori, and H.-j. Wunderlich, "On-Line Prediction of NBTI-induced Aging Rates," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, no. 2, pp. 589–592, 2015.

[7] G. Jerke and A. Kahng, "Mission profile aware IC designA case study," *Design, Automation and Test in Europe …*, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6800278

[8] W. Wang, Z. Wei, S. Yang, and Y. Cao, "An efficient method to identify critical gates under circuit aging," *2007 IEEE/ACM International Conference on Computer-Aided Design*, pp. 735–740, Nov. 2007. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4397353

[9] D. Marculescu, "Joint logic restructuring and pin reordering against NBTI-induced performance degradation," *2009 Design, Automation & Test in Europe Conference & Exhibition*, pp. 75–80, Apr. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5090636

[10] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, and H. Yang, "On the efficacy of input vector control to mitigate NBTI effects and leakage power," *Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009*, pp. 19–26, 2009.

[11] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The impact of NBTI effect on combinational circuit: Modeling, simulation, and analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 2, pp. 173–183, 2010.

[12] G. Klein, "Jflex users manual," *Available on-line at www. jflex. de. Accessed January*, 2015.

References

[1] Haider M. Abbas, Basel Halak, and Mark Zwolinski. BTI mitigation by anti-ageing software patterns. *Microelectronics Reliability*, 79:79–90, 2017.

[2] Haider Abbas, Mark Zwolinski, and Basel Halak. Static Aging Analysis Using 3-Dimensional Delay Library. In *ERMAVSS@ DATE*, pages 34–37, 2016.

[3] Haider Abbas, Mark Zwolinski, and Basel Halak. An application-specific nbti ageing analysis method. In *CMOS Variability (VARI), 2015 International Workshop on*, pages 1–4. IEEE, 2015.

[4] Evelyn Mintarno, Vikas Chandra, David Pietromonaco, Robert Aitken, and Robert W Dutton. Workload dependent NBTI and PBTI analysis for a sub-45nm commercial microprocessor. In *Reliability Physics Symposium (IRPS), 2013 IEEE International*, pages 3A–1. IEEE, 2013.

[5] Tibor Grasser. *Bias Temperature Instability for Devices and Circuits.* Springer Science & Business Media, 2014.

[6] Giuseppe La Rosa and Anand Krishnan. Introduction to the special issue on negative bias temperature instability. *IEEE Transactions on Device and Materials Reliability*, 7(4):500–501, 2007.

[7] Saman Kiamehr, Farshad Firouzi, Mojtaba Ebrahimi, and Mehdi B. Tahoori. Aging-aware standard cell library design. *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, pages 1–4.

[8] Waleed Dweik, Murali Annavaram, and Michel Dubois. Reliability-aware exceptions: Tolerating intermittent faults in microprocessor array structures. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6. IEEE, 2014.

[9] Prashant Singh, Eric Karl, Dennis Sylvester, and David Blaauw. Dynamic nbti management using a 45 nm multi-degradation sensor. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(9):2026–2037, 2011.

[10] Vincent Huard, F Cacho, F Giner, M Saliva, A Benhassain, Dinesh Patel, N Torres, S Naudet, Abhishek Jain, and C Parthasarathy. Adaptive wearout management with in-situ aging monitors. In *Reliability Physics Symposium, 2014 IEEE International*, pages 6B–4. IEEE, 2014.

[11] Vasileios Tenentes, Daniele Rossi, Sheng Yang, Saqib Khursheed, Bashir M Al-Hashimi, and Steve R Gunn. Coarse-grained online monitoring of BTI aging by reusing power-gating infrastructure. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(4):1397–1407, 2017.

[12] Kai-Chiang Wu and Diana Marculescu. Joint logic restructuring and pin reordering against NBTI-induced performance degradation. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 75–80. European Design and Automation Association, 2009.

[13] Yu Wang, Hong Luo, Ku He, Rong Luo, Huazhong Yang, and Yuan Xie. Temperature-aware NBTI modeling and the impact of standby leakage reduction techniques on circuit performance degradation. *Dependable and secure computing, IEEE transactions on*, 8(5):756–769, 2011.

[14] Yu Wang, Xiaoming Chen, Wenping Wang, Varsha Balakrishnan, Yu Cao, Yuan Xie, and Huazhong Yang. On the efficacy of input vector control to mitigate NBTI effects and leakage power. *Proceedings of the 10th International Symposium on Quality Electronic Design, ISQED 2009*, pages 19–26, 2009. doi: 10.1109/ISQED.2009.4810264.

[15] Saman Kiamehr, Mojtaba Ebrahimi, Farshad Firouzi, and Mehdi B Tahoori. Extending standard cell library for aging mitigation. *IET Computers & Digital Techniques*, 9(4):206–212, 2015.

[16] Naghmeh Karimi, Arun Karthik Kanuparthi, Xueyang Wang, Ozgur Sinanoglu, and Ramesh Karri. Magic: Malicious aging in circuits/cores. *ACM Transactions on Architecture and Code Optimization (TACO)*, 12(1):5, 2015.

[17] Pradipkumar Dixit and D Manimekalai. Analysis of reliability for fault tolerant design in nano cmos logic circuit. *Journal of Experimental and Theoretical Nanotechnology Specialized Researches (JETNSR) VOL*, 2(1), 2018.

[18] Piotr Kocanda and Andrzej Kos. Static and dynamic energy losses vs. temperature in different cmos technologies. In *Mixed Design of Integrated Circuits & Systems (MIXDES), 2015 22nd International Conference*, pages 446–449. IEEE, 2015.

[19] Benton H Calhoun, Alice Wang, and Anantha Chandrakasan. Modeling and sizing for minimum energy operation in subthreshold circuits. *IEEE Journal of Solid-State Circuits*, 40(9):1778–1786, 2005.

[20] Marcus T Schmitz, Bashir M Al-Hashimi, and Petru Eles. *System-level design techniques for energy-efficient embedded systems.* Springer Science & Business Media, 2004.

[21] Junko Yoshida. Toyota Case: Single Bit Flip That Killed.

[22] Ulf Schlichtmann, Veit B Kleeberger, Jacob A Abraham, Adrian Evans, Christina Gimmler-Dumon, Michael Glas, Andreas Herkersdorf, Sani R Nassif, and Norbert Wehn. Connecting different worldstechnology abstraction for reliability-aware design and test. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–8. IEEE, 2014.

[23] A. Kumar, A., Veeravalli, B., Bolchini, C. , Miele. Combined DVFS and mapping exploration for lifetime and soft-error susceptibility improvement in MPSoCs. *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, pages 1–6.

[24] Martin Wirnshofer. *Variation-aware adaptive voltage scaling for digital CMOS circuits*, volume 41. Springer, 2013.

[25] J. Srinivasan, S.V. Adve, P. Bose, and J.a. Rivers. The case for lifetime reliability-aware microprocessors. *Proceedings. 31st Annual International Symposium on Computer Architecture, 2004.*, pages 276–287.

[26] Sujan Pandey and Bart Vermeulen. Transient errors resiliency analysis technique for automotive safety critical applications. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 9. European Design and Automation Association, 2014.

[27] Mehdi Kamal, Ali Afzali-Kusha, Saeed Safari, and Massoud Pedram. Design of NBTI-resilient extensible processors. *Integration, the VLSI Journal*, pages 22–34, March . ISSN 01679260. doi: 10.1016/j.vlsi.2014.12.001.

[28] Jiangyi Li and Mingoo Seok. Robust and in-situ self-testing technique for monitoring device aging effects in pipeline circuits. *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, June .

[29] Wenping Wang, Vijay Reddy, Anand T Krishnan, Rakesh Vattikonda, Srikanth Krishnan, and Yu Cao. Compact modeling and simulation of circuit reliability for 65-nm cmos technology. *Device and Materials Reliability, IEEE Transactions on*, 7(4):509–517, 2007.

[30] Deepashree Sengupta and Sachin S Sapatnekar. Estimating circuit aging due to BTI and HCI using ring-oscillator-based sensors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(10):1688–1701, 2017.

[31] John Keane, Xiaofei Wang, Devin Persaud, and Chris H Kim. A high resolution on-chip beat frequency detection system for measuring bti, HCI, and tddb. In *IC Design and Technology (ICICDT), 2010 IEEE International Conference on*, pages 142–145. IEEE, 2010.

[32] Hong Luo, Xiaoming Chen, Jyothi Velamala, Yu Wang, Yu Cao, Vikas Chandra, Yuchun Ma, and Huazhong Yang. Circuit-level delay modeling considering both TDDB and NBTI. *Proceedings of the 12th International Symposium on Quality Electronic Design, ISQED 2011*, pages 14–21, 2011. ISSN 1948-3287.

[33] Shireen Warnock, Allison Lemus, Jungwoo Joh, Srikanth Krishnan, Sameer Pendharkar, and Jesús A del Alamo. Time-dependent dielectric breakdown in high-voltage gan mis-hemts: The role of temperature. *IEEE Transactions on Electron Devices*, 64(8):3132–3138, 2017.

[34] Souvik Mahapatra. *Fundamentals of Bias Temperature Instability in MOS Transistors: Characterization Methods, Process and Materials Impact, DC and AC Modeling*, volume 52. Springer, 2015.

[35] Aminur Rahman, P Bai, G Curello, J Hicks, C-H Jan, M Jamil, Jongho Park, K Phoa, Md Saifur Rahman, Chia-Yin Tsai, et al. Reliability studies of a 22nm soc platform technology featuring 3-d tri-gate, optimized for ultra low power, high performance and high density application. In *Reliability Physics Symposium (IRPS), 2013 IEEE International*, pages PI–2. IEEE, 2013.

[36] Simone Corbetta and William Fornaciari. NBTI mitigation in microprocessor designs. In *Proceedings of the great lakes symposium on VLSI*, pages 33–38. ACM, 2012.

[37] Joseph W McPherson. Reliability challenges for 45nm and beyond. In *Proceedings of the 43rd annual Design Automation Conference*, pages 176–181. ACM, 2006.

[38] Souvik Mahapatra and Narendra Parihar. A review of NBTI mechanisms and models. *Microelectronics Reliability*, 81:127–135, 2018.

[39] James H Stathis, Souvik Mahapatra, and Tibor Grasser. Controversial issues in negative bias temperature instability. *Microelectronics Reliability*, 81:244–251, 2018.

[40] Shimpei Tsujikawa, Toshiyuki Mine, Kikuo Watanabe, Yasuhiro Shimamoto, Ryuta Tsuchiya, Kazuhiro Ohnishi, Takahiro Onai, Jiro Yugami, and Shin'ichiro Kimura. Negative bias temperature instability of pmosfets with ultra-thin sion gate dielectrics. In *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International*, pages 183–188. IEEE, 2003.

[41] Seyab Khan and Said Hamdioui. Temperature dependence of NBTI induced delay. In *On-Line Testing Symposium (IOLTS), 2010 IEEE 16th International*, pages 15–20. IEEE, 2010.

[42] Dieter K Schroder. Negative bias temperature instability: What do we understand? *Microelectronics Reliability*, 47(6):841–852, 2007.

[43] M Ershov, S Saxena, H Karbasi, S Winters, S Minehane, J Babcock, R Lindley, P Clifton, M Redford, and A Shibkov. Dynamic recovery of negative bias temperature instability in p-type metal–oxide–semiconductor field-effect transistors. *Applied physics letters*, 83(8):1647–1649, 2003.

[44] James H Stathis. The physics of NBTI: What do we really know? In *Reliability Physics Symposium (IRPS), 2018 IEEE International*, pages 2A–1. IEEE, 2018.

[45] Hans Reisinger, O Blank, Wolfgang Heinrigs, A Muhlhoff, Wolfgang Gustin, and C Schlunder. Analysis of NBTI degradation-and recovery-behavior based on ultra fast vt-measurements. In *Reliability Physics Symposium Proceedings, 2006. 44th Annual., IEEE International*, pages 448–453. IEEE, 2006.

[46] S Mahapatra, N Goel, S Desai, S Gupta, B Jose, S Mukhopadhyay, K Joshi, A Jain, AE Islam, and MA Alam. A comparative study of different physics-based NBTI models. *IEEE Transactions on Electron Devices*, 60(3):901–916, 2013.

[47] Rakesh Vattikonda, Wenping Wang, and Yu Cao. Modeling and minimization of pmos NBTI effect for robust nanometer design. In *Proceedings of the 43rd annual Design Automation Conference*, pages 1047–1052. ACM, 2006.

[48] Sarvesh Bhardwaj, Wenping Wang, Rakesh Vattikonda, Yu Cao, and Sarma Vrudhula. Predictive modeling of the NBTI effect for reliable design. In *Custom Integrated Circuits Conference, 2006. CICC'06. IEEE*, pages 189–192. IEEE, 2006.

[49] Tibor Grasser. Stochastic charge trapping in oxides: From random telegraph noise to bias temperature instabilities. *Microelectronics Reliability*, 52(1):39–70, 2012.

[50] Narendra Parihar, Nilesh Goel, Subhadeep Mukhopadhyay, and Souvik Mahapatra. BTI Analysis Tool-Modeling of NBTI DC, AC Stress and Recovery Time Kinetics, Nitrogen Impact, and EOL Estimation. *IEEE Transactions on Electron Devices*, 2017.

[51] Chengbin Ma, Hans Jurgen Mattausch, M Miura-Mattausch, K Matsuzawa, Takeshi Hoshida, M Imade, R Koh, and Takeshi Arakawa. Universal NBTI model and its application for high frequency circuit simulation. In *Reliability Physics Symposium, 2014 IEEE International*, pages CA–4. IEEE, 2014.

[52] Yu Wang, Hong Luo, Ku He, Rong Luo, Huazhong Yang, and Yuan Xie. Temperature-aware NBTI modeling and the impact of standby leakage reduction techniques on circuit performance degradation. *IEEE Transactions on Dependable and Secure Computing*, 8(2005):756–769, 2011. ISSN 15455971. doi: 10.1109/TDSC.2010.41.

[53] Shengqi Yang, Wenping Wang, Mark Hagan, Wei Zhang, Pallav Gupta, and Yu Cao. NBTI-aware circuit node criticality computation. *ACM Journal on Emerging Technologies in Computing Systems*, (3):1–19, September . ISSN 15504832. doi: 10.1145/2491681.

[54] Yao Wang, Sorin D Cotofana, and Liang Fang. Lifetime reliability assessment with aging information from low-level sensors. In *Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI*, pages 339–340. ACM, 2013.

[55] Hong Luo, Yu Wang, Ku He, Rong Luo, Huazhong Yang, and Yuan Xie. A novel gate-level NBTI delay degradation model with stacking effect. In *International Workshop on Power and Timing Modeling, Optimization and Simulation*, pages 160–170. Springer, 2007.

[56] Chang-Chih Chen, Taizhi Liu, and Linda Milor. System-level modeling of microprocessor reliability degradation due to bias temperature instability and hot carrier injection. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(8):2712–2725, 2016.

[57] Soonyoung Cha, Chang-Chih Chen, and Linda S Milor. System-level estimation of threshold voltage degradation due to NBTI with i/o measurements. In *Reliability Physics Symposium, 2014 IEEE International*, pages PR–1. IEEE, 2014.

[58] Steven F Barrett and Daniel J Pack. Microcontrollers fundamentals for engineers and scientists. *Synthesis Lectures on Digital Circuits and Systems*, 1(1):1–124, 2005.

[59] James F Cox. *Fundamentals of linear electronics: integrated and discrete*. Cengage Learning, 2002.

[60] Bogdan Tudor, Joddy Wang, Weidong Liu, and Hany Elhak. Mos device aging analysis with hspice and customsim. *Synopsys, White Paper*, 2011.

[61] Bogdan Tudor, Joddy Wang, Zhaoping Chen, Robin Tan, Weidong Liu, and Frank Lee. An accurate and scalable mosfet aging model for circuit simulation. In *Quality Electronic Design (ISQED), 2011 12th International Symposium on*, pages 1–4. IEEE, 2011.

[62] Wenping Wang, Vijay Reddy, Anand T Krishnan, Rakesh Vattikonda, Srikanth Krishnan, and Yu Cao. An integrated modeling paradigm of circuit reliability for 65nm cmos technology. In *Custom Integrated Circuits Conference, 2007. CICC'07. IEEE*, pages 511–514. IEEE, 2007.

[63] Jyothi B Velamala, Ketul B Sutaria, Takashi Sato, and Yu Cao. Aging statistics based on trapping/detrapping: Silicon evidence, modeling and long-term prediction. In *Reliability Physics Symposium (IRPS), 2012 IEEE International*, pages 2F–2. IEEE, 2012.

[64] Goeran Jerke and Andrew B Kahng. Mission profile aware ic design: a case study. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 64. European Design and Automation Association, 2014.

[65] W Abadeer and W Ellis. Behavior of NBTI under ac dynamic circuit conditions. In *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International*, pages 17–22. IEEE, 2003.

[66] Lide Zhang and Robert P Dick. Scheduled voltage scaling for increasing lifetime in the presence of NBTI. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*, pages 492–497. IEEE, 2009.

[67] Narendra Parihar, Nilesh Goel, Ankush Chaudhary, and Souvik Mahapatra. A modeling framework for NBTI degradation under dynamic voltage and frequency scaling. *IEEE Transactions on Electron Devices*, 63(3):946–953, 2016.

[68] Kai-Chiang Wu and Diana Marculescu. Joint logic restructuring and pin reordering against nbti-induced performance degradation. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 75–80. European Design and Automation Association, 2009.

[69] Paulo F Butzen, Vinícius Dal Bem, André I Reis, and Renato P Ribas. Transistor network restructuring against NBTI degradation. *Microelectronics Reliability*, 50 (9):1298–1303, 2010.

[70] Kai-Chiang Wu and Diana Marculescu. Joint logic restructuring and pin reordering against NBTI-induced performance degradation. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 75–80. European Design and Automation Association, 2009.

[71] Wenping Wang, Zile Wei, Shengqi Yang, and Yu Cao. An efficient method to identify critical gates under circuit aging. *2007 IEEE/ACM International Conference on Computer-Aided Design*, pages 735–740, November . ISSN 1092-3152. doi: 10.1109/ICCAD.2007.4397353.

[72] Bipul C Paul, Kunhyuk Kang, Haldun Kufluoglu, Muhammad Ashraful Alam, and Kaushik Roy. Temporal performance degradation under NBTI: Estimation and design for improved reliability of nanoscale circuits. In *Design, Automation and Test in Europe, 2006. DATE'06. Proceedings*, volume 1, pages 1–6. IEEE, 2006.

[73] Xiangning Yang and Kewal Saluja. Combating NBTI degradation via gate sizing. In *Quality Electronic Design, 2007. ISQED'07. 8th International Symposium on*, pages 47–52. IEEE, 2007.

[74] Saman Kiamehr, Farshad Firouzi, Mojtaba Ebrahimi, and Mehdi B Tahoori. Aging-aware standard cell library design. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 261. European Design and Automation Association, 2014.

[75] Mohammad Saber Golanbari, Saman Kiamehr, Mojtaba Ebrahimi, and Mehdi B Tahoori. Aging guardband reduction through selective flip-flop optimization. In *Test Symposium (ETS), 2015 20th IEEE European*, pages 1–6. IEEE, 2015.

[76] Saurabh Kothawade, Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy. Mitigating NBTI in the physical register file through stress prediction. In *Computer Design (ICCD), 2012 IEEE 30th International Conference on*, pages 345–351. IEEE, 2012.

[77] Jifeng Chen, Shuo Wang, and Mohammad Tehranipoor. Efficient selection and analysis of critical-reliability paths and gates. In *Proceedings of the great lakes symposium on VLSI*, pages 45–50. ACM, 2012.

[78] Rizwan A Ashraf, Navid Khoshavi, Ahmad Alzahrani, Ronald F DeMara, Saman Kiamehr, and Mehdi B Tahoori. Area-energy tradeoffs of logic wear-leveling for BTI-induced aging. In *Proceedings of the ACM International Conference on Computing Frontiers*, pages 37–44. ACM, 2016.

[79] Chao Lin, Yu-Hung Cho, and Yi-Ming Yang. Aging-aware reliable multiplier design with adaptive hold logic. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23(3):544–556, 2015.

[80] Dan Ernst, Nam Sung Kim, Shidhartha Das, Sanjay Pant, Rajeev Rao, Toan Pham, Conrad Ziesler, David Blaauw, Todd Austin, Krisztian Flautner, et al. Razor: A low-power pipeline based on circuit-level timing speculation. In *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, page 7. IEEE Computer Society, 2003.

[81] Prashant Singh, Eric Karl, Dennis Sylvester, and David Blaauw. Dynamic NBTI management using a 45 nm multi-degradation sensor. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(9):2026–2037, 2011.

[82] Gerwin Klein. Jflex users manual. *Available on-line at www. jflex. de. Accessed July*, 2018.

[83] Manish Arora, Srilatha Manne, Indrani Paul, Nuwan Jayasena, and Dean M Tullsen. Understanding idle behavior and power gating mechanisms in the context of modern benchmarks on cpu-gpu integrated systems. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 366–377. IEEE, 2015.

[84] S. Di Carlo, M. Gaudesi, E. Sanchez, and M. Sonza Reorda. A functional approach for testing the reorder buffer memory. *Journal of Electronic Testing*, 30(4):469–481, 2014. ISSN 1573-0727. doi: 10.1007/s10836-014-5461-9.

[85] F. Firouzi, S. Kiamehr, and M. B. Tahoori. NBTI mitigation by optimized NOP assignment and insertion. *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 218–223. ISSN 15301591. doi: 10.1109/DATE.2012.6176465.

[86] Matthew R Guthaus, Jeffrey S Ringenberg, Dan Ernst, Todd M Austin, Trevor Mudge, and Richard B Brown. Mibench: A free, commercially representative embedded benchmark suite. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 3–14. IEEE, 2001.

[87] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R Hower, Tushar Krishna, Somayeh Sardashti, et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2):1–7, 2011.

[88] Dominik Lorenz, Georg Georgakos, and Ulf Schlichtmann. Aging analysis of circuit timing considering NBTI and HCI. In *On-Line Testing Symposium, 2009. IOLTS 2009. 15th IEEE International*, pages 3–8. IEEE, 2009.

[89] Jaume Abella, Xavier Vera, and Antonio Gonzalez. Penelope: The NBTI-aware processor. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 85–96. IEEE Computer Society, 2007.

[90] Majid Namaki-Shoushtari, Abbas Rahimi, Nikil Dutt, Puneet Gupta, and Rajesh K Gupta. Argo: aging-aware gpgpu register file allocation. In *Hardware/-Software Codesign and System Synthesis (CODES+ ISSS), 2013 International Conference on*, pages 1–9. IEEE, 2013.

[91] Shuai Wang, Tao Jin, Chuanlei Zheng, and Guangshan Duan. Low power aging-aware register file design by duty cycle balancing. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 546–549. EDA Consortium, 2012.

[92] Jaume Abella, Xavier Vera, and Antonio Gonzalez. Penelope: The NBTI-aware processor. In *Microarchitecture, 2007. MICRO 2007. 40th Annual IEEE/ACM International Symposium on*, pages 85–96. IEEE, 2007.

[93] Cícero Nunes, Paulo F Butzen, André I Reis, and Renato P Ribas. Bti, HCI and tddb aging impact in flip–flops. *Microelectronics Reliability*, 53(9-11):1355–1359, 2013.

[94] Krishnan Ramakrishnan, Xiaoxia Wu, Narayanan Vijaykrishnan, and Yuan Xie. Comparative analysis of NBTI effects on low power and high performance flip-flops. In *Computer Design, 2008. ICCD 2008. IEEE International Conference on*, pages 200–205. IEEE, 2008.

[95] Junyoung Park and Jacob A Abraham. An aging-aware flip-flop design based on accurate, run-time failure prediction. In *VLSI Test Symposium (VTS), 2012 IEEE 30th*, pages 294–299. IEEE, 2012.

[96] Vikram G Rao and Hamid Mahmoodi. Analysis of reliability of flip-flops under transistor aging effects in nano-scale cmos technology. In *Computer Design (ICCD), 2011 IEEE 29th International Conference on*, pages 439–440. IEEE, 2011.

[97] Hamed Abrishami, Safar Hatami, and Massoud Pedram. Design and multicorner optimization of the energy-delay product of cmos flip–flops under the negative bias temperature instability effect. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(6):869–881, 2013.

[98] Hamed Abrishami, Safar Hatami, Behnam Amelifard, and Massoud Pedram. NBTI-aware flip-flop characterization and design. In *Proceedings of the 18th ACM Great Lakes symposium on VLSI*, pages 29–34. ACM, 2008.

[99] HSPICE. Reference manual: Commands and control options. Technical report, 2012.

[100] Sanjay V Kumar, KH Kim, and Sachin S Sapatnekar. Impact of NBTI on sram read stability and design for reliability. In *Quality Electronic Design, 2006. ISQED'06. 7th International Symposium on*, pages 6–pp. IEEE, 2006.

[101] Kunhyuk Kang, Haldun Kufluoglu, Kaushik Roy, and Muhammad Ashraful Alam. Impact of negative-bias temperature instability in nanoscale sram array: Modeling and analysis. *IEEE Tranfsactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(10):1770–1781, 2007.

[102] Saurabh Kothawade, Koushik Chakraborty, and Sanghamitra Roy. Analysis and mitigation of NBTI aging in register file: An end-to-end approach. In *Quality Electronic Design (ISQED), 2011 12th International Symposium on*, pages 1–7. IEEE, 2011.

[103] Evert Seevinck, Frans J List, and Jan Lohstroh. Static-noise margin analysis of mos sram cells. *IEEE Journal of solid-state circuits*, 22(5):748–754, 1987.

[104] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[105] Tibor Grasser and Siegfried Selberherr. Modeling of negative bias temperature instability. *Journal of Telecommunications and Information Technology*, pages 92–102, 2007.

[106] Nam Sung Kim, Todd Austin, David Baauw, Trevor Mudge, Krisztián Flautner, Jie S Hu, Mary Jane Irwin, Mahmut Kandemir, and Vijaykrishnan Narayanan. Leakage current: Moore's law meets static power. *Computer*, 36(12):68–75, 2003.

[107] Mark T Bohr, Robert S Chau, Tahir Ghani, and Kaizad Mistry. The high-k solution. *IEEE spectrum*, 44(10):29–35, 2007.

[108] Ermao Cai and Diana Marculescu. Temperature effect inversion-aware power-performance optimization for FinFET-based multicore systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(11): 1897–1910, 2017.

[109] W Bircher, Jason Law, Madhavi Valluri, and Lizy K John. Effective use of performance monitoring counters for run-time prediction of power. *University of Texas at Austin Technical Report TR-041104*, 1, 2004.

[110] Wim Heirman, Trevor Carlson, and Lieven Eeckhout. Sniper: Scalable and accurate parallel multi-core simulation. In *8th International Summer School on Advanced Computer Architecture and Compilation for High-Performance and Embedded Systems (ACACES-2012)*, pages 91–94. High-Performance and Embedded Architecture and Compilation Network of Excellence (HiPEAC), 2012.

[111] Sheng Li, Jung Ho Ahn, Richard D Strong, Jay B Brockman, Dean M Tullsen, and Norman P Jouppi. Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, pages 469–480. IEEE, 2009.

[112] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[113] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[114] Kevin Skadron, Mircea R Stan, Karthik Sankaranarayanan, Wei Huang, Sivakumar Velusamy, and David Tarjan. Temperature-aware microarchitecture: Modeling and implementation. *ACM Transactions on Architecture and Code Optimization (TACO)*, 1(1):94–125, 2004.

[115] Wei Huang, Shougata Ghosh, Sivakumar Velusamy, Karthik Sankaranarayanan, Kevin Skadron, and Mircea R Stan. Hotspot: A compact thermal modeling methodology for early-stage vlsi design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14(5):501–513, 2006.

[116] Francesco Zanini, David Atienza, Luca Benini, and Giovanni De Micheli. Multicore thermal management with model predictive control. In *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*, pages 711–714. IEEE, 2009.

[117] Aaron Carroll and Gernot Heiser. Unifying dvfs and offlining in mobile multicores. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2014 IEEE 20th*, pages 287–296. IEEE, 2014.

[118] James W Tschanz, Siva G Narendra, Yibin Ye, Bradley A Bloechel, Shekhar Borkar, and Vivek De. Dynamic sleep transistor and body bias for active leakage power control of microprocessors. *IEEE Journal of Solid-State Circuits*, 38(11): 1838–1845, 2003.

[119] Ravindra Jejurikar, Cristiano Pereira, and Rajesh Gupta. Leakage aware dynamic voltage scaling for real-time embedded systems. In *Proceedings of the 41st annual Design Automation Conference*, pages 275–280. ACM, 2004.

[120] Taeyoung Kim, Xin Huang, Hai-Bao Chen, Valeriy Sukharev, and Sheldon X-D Tan. Learning-based dynamic reliability management for dark silicon processor considering em effects. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016*, pages 463–468. IEEE, 2016.

[121] Pietro Mercati, Andrea Bartolini, Francesco Paterna, Tajana Simunic Rosing, and Luca Benini. Workload and user experience-aware dynamic reliability management in multicore processors. In *Design Automation Conference (DAC), 2013 50th ACM/EDAC/IEEE*, pages 1–6. IEEE, 2013.

[122] Tajana Simunic Rosing, Kresimir Mihic, and Giovanni De Micheli. Power and reliability management of socs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(4):391–403, 2007.

[123] Eric Karl, David Blaauw, Dennis Sylvester, and Trevor Mudge. Reliability modeling and management in dynamic microprocessor-based systems. In *Design Automation Conference, 2006 43rd ACM/IEEE*, pages 1057–1060. IEEE, 2006.

[124] Jooyoung Park and Irwin W Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

[125] Xi Meng, Pawel Rozycki, Jun-Fei Qiao, and Bogdan M Wilamowski. Nonlinear system modeling using rbf networks for industrial application. *IEEE Transactions on Industrial Informatics*, 14(3):931–940, 2018.