N° d'ordre : 4448

THÈSE

PRÉSENTÉE A

L'UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DES SCIENCES

STIC (Sciences et technologies de l'information et de la communication)

Par Tushar, GUPTA

Équipe d'accueil : CEA LIST / DACLE / LFSE

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ: Électronique

HANDLING DESIGN ISSUES RELATED TO RELIABILITY IN MPSOC AT FUNCTIONAL LEVEL

Directeur de thèse: M. Thomas Zimmer

Soutenance prévue le : 15 Décembre 2011

Devant la commission d'examen formée de :

Mme. NAVINER, Lirida	Professeur à TELECOM Paris Tech	Rapporteur
M. GARDA, Patrick	Professeur à l'Univ. Pierre and Marie Curie	Rapporteur
M. GOGNIAT, Guy	Professeur à l'Univ. De Bretagne Sud	Examinateur
M. HUARD, Vincent	Reliability Section Manager à ST Microelectronics	Examinateur
M. ZIMMER, Thomas	Professeur à l'Univ. Bordeaux 1	Directeur de thèse
M. MARC, François	Maitre de Conférences, HDR à l'Univ. Bordeaux 1	Codirecteur de thèse
M. HERON, Olivier	Ingénieur-Chercheur CEA LIST	Encadrant de thèse
M. VENTROUX, Nicolas	Ingénieur-Chercheur CEA LIST	Encadrant de thèse

I would like to dedicate this thesis to my parents and their patience,

Acknowledgements

It has been a great 3 year journey came to an end. I would like to thank everyone who helped me prosper during this thesis at CEA LIST. I would like to mention few in this page.

Firstly, I would like to thank Olivier Heron, who chose me to work under his supervision for an interesting and ambitious subject, which is very new to the industry and enough potential for the future researchers to work on. I have grown and matured a lot during past three years under the guidance of Olivier and Nicolas Ventroux. Their enthusiasm to guide me in good direction made this work easy to compile. Thomas Zimmer, the director of thesis and Francois Marc, the co-director of thesis, I have not been able to see them that often, but they shared their knowledge with me using different means of communication. It would have had been impossible to prepare the state of the art without the effort from all the four of them.

In both laboratories LFSE and LCE inside CEA-LIST, I met people working on very different subjects that satisfied my curiosity to know how things work in industry. I would like to thank the members of jury for the PhD defense, who provided their valuable remarks to improve the quality of the report. I am more confident about the importance of this work after discussion with the jury, who belong to different domains of semiconductor industry.

It would have been quite boring to work if I was alone to work all the day everyday, I would like to give big cheers to Clement and Charly. Especially Clement, who was there with me for last 3 years (approximately), during all the good and bad moments. I am sorry; I cannot be there during last year of your PhD. I wish you all the best in life.

Life is not only work and studies. It is not possible to name everyone I made friends with in Ile de France during these quick three years. But a few to name, thank you Elodie, Mylene, Sabrina and special thanks to Annie, Sylvie and Frederic. I would have never been able to finish the French administration work without your help. I found my second family in the faces of Annie, Olivier and Clement. I respect you from my heart. You made the last three days so special that I can never be able to repay you.

My family and friends have shown great patience as I was complaining a lot about different things. I hope I have not bored you but cheered you up most of the times when we talked. Thank you France, Remi, Salima, Shivam, you were always there when I was tired from my work.

Finally, I thank my Mother, Father and Sister, who never said no to me. Always try to cheer me up, talk to me (in Hindi) whenever I was getting lost in English and French. As a small step to show my gratitude I dedicate this thesis to you. I could include the name of my future life partner, but I still do not know her, so let us not talk about her.

Contents

List of Figures

List of Tables

Introduction

Contents

Motivations and Objectives

Multi-Processor System-On-Chip (MPSoC) are complex digital circuits but are very attractive for embedded computing intensive applications. They are widely used in different type of industrial products, e.g., avionics, automobiles, electrical appliances, factory machines, and so on. As an example, in order to inform real-time traffic updates, a global traffic information system can be used. Automobiles can receive the information by using infrared communication. This example suggests that embedded systems become not only complex but also components in a huge complex system.

Such integrated circuits (IC) are composed of up to hundreds of processor cores, memories and interconnect. They constitute a complex embedded system with requirements such as high performance or real-time or low power etc. High performance is obtained by exploiting the benefit of transistor shrinking (vs. Moore's Law) and the available massive parallelism. Their design opens several challenges such as methods to parallelize the applications among the processors, the memory hierarchy organization, the communication latency between processors and memories, etc. Such System-on-Chips are manufactured with the most leading-edge technology. Die shrinking leads to faster devices and higher number of transistors per unit area but less reliable devices. ITRS roadmap [1] identifies the interconnect reliability as one of the 5 difficult issues that need to be solved before the 22nm node. Transistor reliability is affected by degradation and variation phenomena that cause a drift of their threshold voltage till the loss of their functionality. The reliability of an IC is generally defined as the likelihood that the IC provides the correct service for what it was intended after a specific period of functioning, in given operating and environment conditions. A common metric used in semiconductor industry is the failure rate that represents the frequency with which any engineered system or component fails, expressed generally in failures per hour or failures per billions of hours (failure-in-time or FIT). It can be written as in Eq.1.

$$1FIT = 10^{-9}/hours, \tag{1}$$

It was observed that the failure rate of the semiconductor devices in the field generally ranges from 10 to 100 FIT. We have many kinds of failure mechanisms that may result in intermittent and permanent errors in ICs. According to [1], the failure rate of devices used in an average IC can be explained by using the bathtub curve shown in figure 1. Taken from the standpoint of time, the device failures can be classified as early failure, random failure and wear-out failure periods. The 'product service life' is its expected lifetime or the acceptable period of use in service. It represents the time that any manufactured item can be expected to be 'serviceable' or supported by its manufacturer. Two points must be considered regarding the service life of a device; early and random failures rates and lifetime before wear-out. In fact, both failure rates of semiconductors gradually diminish as a factor of time as depicted in figure 1. In other words, a notable feature of semiconductor devices is that the longer a particular device has been used the more stable during the lifetime it will be before wear-out or aging comes into effect [2].

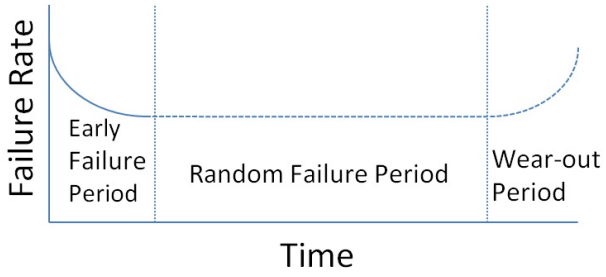

Figure 1: Observed failure rate in average vs. operation time for typical electronic and semiconductor devices

In this thesis, we focus on aging related failures. The major failures mechanisms are Electromigration (EM) in interconnect, hard/soft oxide breakdown, hot carrier injection (HCI), and bias temperature instability (BTI) in MOS (Metal-Oxide-Semiconductor) transistors. These failure mechanisms are still extensively studied at the transistor level and semiconductor industry provides ranges of values of the technology dependent parameters [3].

To keep the whole IC reliability constant, the failure rate per device must decrease as transistor density increases at each shrinking step or technology node. ITRS roadmap [4, 5] predicts that semiconductor industry knows solutions to reach the requirements till end of next year. After what, only interim solutions are known. Consequently, the reliability issue for MPSoC should no longer be a manufacturer problem but should become a design constraint/objective in the CAD flow. Today, the design flow includes a sign-off step before design tape-out that concludes on the reliability objective vs. design requirements. Existing commercial and academic simulation tools, described in Chapter 3, provide a solution to evaluate the performance drift and reliability hotspots in design in back-end flow i.e. at transistor- and layout-levels. However, these State-of-the-Art tools are not a sufficient answer to the design issue of MPSoC. In this thesis, we address the design issue of MPSoC regarding the reliability in the front-end i.e. before physical synthesis.

As the design of these systems becomes a more complex task [6], the first design step in a CAD flow has to start above the Register-Transfer abstraction Level (RTL). The design space exploration (memory sizes, processor pipeline depth, interconnect bandwidth, task scheduling, etc.) for performance or power consumption objectives requires fast and accurate simulators. Performance, power and temperature modeling and simulations at high level of abstraction for MPSoCs are still subject to intensive research works. At different levels of abstraction, there are different speed vs. accuracy trade-offs to evaluate these non-functional parameters. A more accurate data can be obtained at lower level of abstraction than higher. But the simulation is faster at higher level of abstraction. Functional abstraction level - or in a simpler way, functional level - is a representation level of the system that only describes the behavior of the digital blocks (i.e. bigger structure than one standard logic cell: whole processor or microarchitecture stage, peripheral, memory bank or circuit, etc.) without implementation details. As an example, only the functionality of a processor instruction is simulated at this level while the underlying microarchitecture is not described. One instruction is hence assumed to be executed in one clock cycle. The program execution time is less accurate but the simulation time is highly faster than the one obtained with a simulation of the same processor at gate-level, including the detailed implementation of the microarchitecture, with the same benchmarks. In this thesis, we propose a methodology to integrate reliability evaluation capabilities of MPSoC systems in a front-end design flow at functional level.

Relatively to the other recent works related to this topic, such as [7], the objective of the thesis is to develop a methodology to integrate reliability capabilities in a CAD flow which satisfies the following criteria:

1. Need of speed during simulation: the reliability of a digital block is simulated at functional level. We need to elaborate a modeling method of aging that fills the gap between process and front-end design;
2. Need of a 'powerful' language able to describe both the digital behavior of the block and the aging behavior within the block. In addition this language enables the integration of the augmented block model in a SystemC-based MPSoC simulator [8];
3. Need to distinguish the effect of different benchmarks on lifetime reliability of the processor and explore the effect of different task scheduling techniques in an MPSoC, very early in the design flow, taking into account various technology libraries.

The technical contribution of this thesis would be a trace-based tool-chain (power, temperature and reliability) that is fully parameterized for exploring the reliability in a single processor circuit, at functional level. The parameters are the design inputs, assembly technology and operating and environment conditions. User can plug any technology, packaging and failure libraries from manufacturers. Reliability of a digital block is expressed here as the Cumulative Failure Rate (CFR) over time for each failure mechanism [9]. It is important to clarify at this point that CFR should not be confused with the definition of constant failure rate. One great benefit of this simulator is the ability to highlight the main failure detractors and the weak parts of the design that are the most prone to these detractors.

The organization of manuscript

The thesis report has been organized in 5 chapters

The first chapter categorizes various types of errors that can occur during the useful life of the chip. Intrinsic errors are discussed in details in this part as this thesis is to provide a comparative study regarding these specific types of failures. Readers can get a basic idea about the physics behind these failure mechanisms.

The second chapter starts with applied mathematics in semiconductors. Various definitions are provided in the first part of this chapter that is used in generic failure language within the reliability field. The second part presents the common failure models at transistor level. These failure models are the inputs of complete methodology, which is developed in chapter 4. Finally, the 'macro' failure models of a digital block for 4 failure mechanisms are derived at functional level, using the existing transistor level models. These models display dynamic parameters and static parameters. Dynamic parameters are dynamic power consumption and mean temperature (spatial) of a digital block. In the context of a processor, these parameters depend on, but not limited to, the program executed. Static parameters are those related to manufacturer libraries.

The third chapter as the final part of State-Of-The-Art focuses on existing simulation tools and methodologies for aging. The simulators provide the values of the dynamic parameters of the models listed in Chapter 2. Most of reliability aware simulators are performance simulators extended with new capabilities. Different reliability simulators are categorized according to their level of abstraction. The pros and cons of each solution are studied carefully. Finally a comparison table is provided to make a synthesis of this study and to prove and conclude, how the proposed methodology will contribute to the scientific literature and industry needs.

The fourth chapter presents our methodology developed to predict the reliability of a RISC processor at functional level. Firstly, the chapter describes the instruction set simulator (ISS), used to simulate the behavior of a processor. In this work, we adopt ArchC language, an architectural description language that allows generating automatically an ISS, ready to be integrated in a SystemC based MPSoC simulator. As shown in Chapter 2, power and temperature values of the processor over time must be estimated and recorded during the simulation of applications. A State-of-the-Art discusses on the existing power consumption and temperature simulators and highlights the lack regarding our proposed methodology. Next, a reliability simulator, called real time MTTF evaluator (RTME), is developed: it estimates the reliability of a digital block (standard cell based) by reading power and temperature traces. Finally, a power simulator, called Power-ArchC, is developed to estimate the power consumption of the processor is developed. HotSpot, an already existing academic tool is used to estimate temperature. The various elements mentioned above forms the tool-chain named RAAPS (Reliability Aware ArchC based Processor Simulator) are explained in details. Finally, the various modes of RAAPS are discussed to satisfy different user's needs.

The fifth chapter presents a validation of RAAPS tool-chain. In the first part, we provide simulation time of RAAPS compared to other tools. Next, the standard deviation of CFR is discussed, where the deviation is due to the uncertainty in power and temperature values. Third, we discuss on the impact of energy consumption on the CFR of a 32-bit MIPS processor. Finally, a discussion presents two scenarios to improve the processor reliability (decrease the CFR level) at high level.

Finally, the conclusion section summarizes the main results. Also, various long/short term future perspectives are also discussed, which can help in improving the methodology.

As summarized in Figure 2, the first and third chapters are completely based on State-of-the-Art, whereas second and fourth chapter comprise of part of State-of-the-Art and part

of new contributions. The failure models are derived at functional level using State-of-the-Art transistor level failure models in Chapter 2. A State-of-the-Art about existing power and temperature simulators in front-end is provided in Chapter 4, to motivate the proposed tool-chain. The rest of Chapter 4 presents RAAPS and Chapter 5 a validation of it.

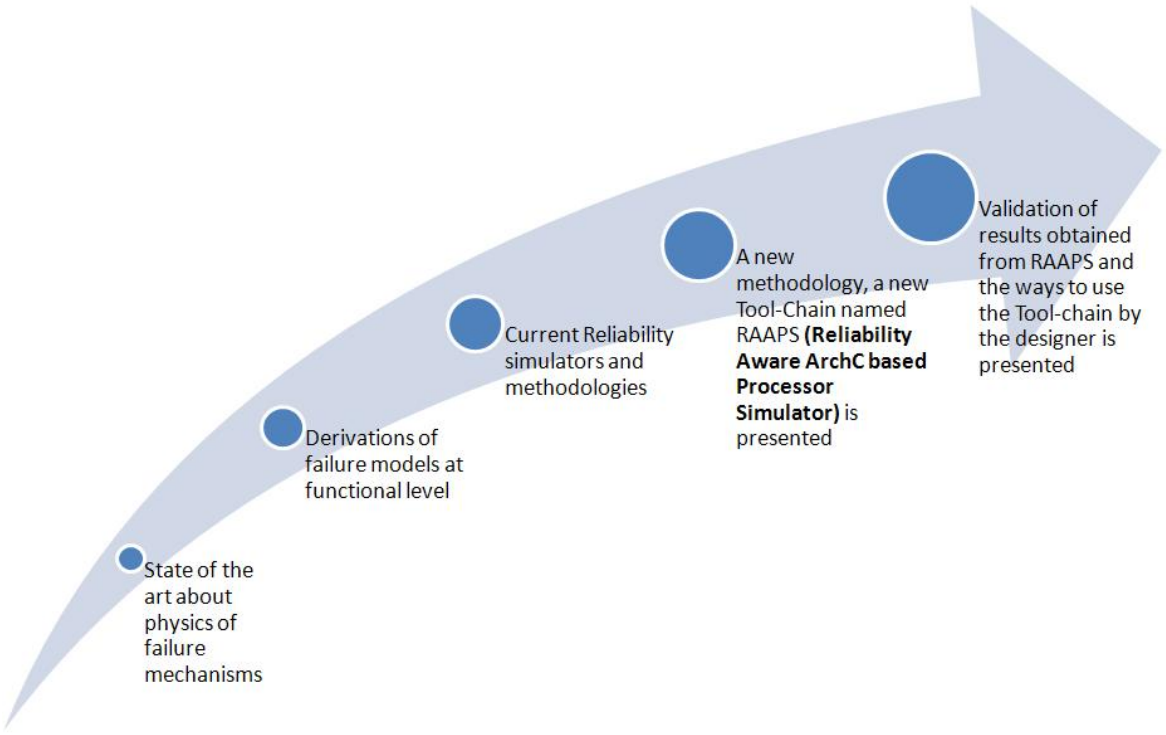

Figure 2: To remember before proceeding: Structure of the report, arranged in five chapters

Chapter 1

State of the art - Failure mechanisms in a chip

Contents

Every living or non-living thing degrades with time and the semiconductor devices are no different. It is important to define the level of degradation i.e., the point the device is considered to be failed. Also, the user of these devices wants to have knowledge about the life of the device he/she is going to buy. Current chapter is the base of this thesis report. It provides a discussion about various causes of failures or errors in a product manufactured using semiconductor devices. These errors occurring during the lifetime of a circuit, marks a question regarding reliability of the circuit. Various failures that occur during the aging are discussed and linked with working of a transistor. Since, various parameters affect various failure mechanisms in different manner, it is important to study the physics behind these mechanisms. In the following chapters, handling and modeling of these failure mechanisms at functional level of abstraction is shown. These failure mechanisms are chosen based on

the fact that they are dynamic and help in aging of the device. The following chapters will provide more mathematical details regarding some of the relevant failure mechanisms.

1.1 Errors in a system made of millions of these transistors

In [10], Srinivasan et al. gave a clear classification of different types of errors that can occur in a system, and is shown in figure 1.1

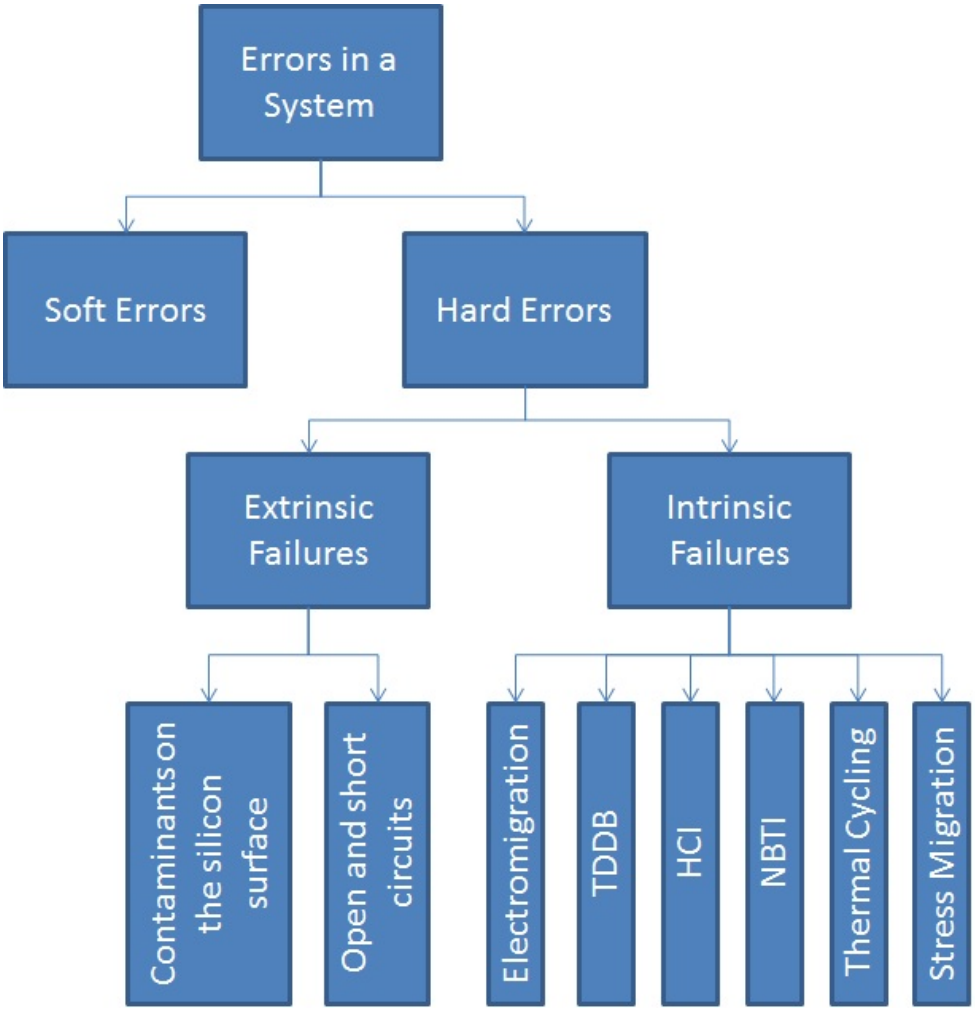

Figure 1.1: Classification of errors in a system

1.1.1 Soft Errors

Soft errors are mainly SEUs (Single Event Upsets) and are errors in processor execution due to electrical noise or external radiation, rather than design or manufacturing related defects [11, 12, 13]. SEUs are responsible for computer crashes, data corruption, and systems that

just suddenly stop working properly [14]. SETs are less likely to crash a system but can also manifest themselves in ways that are similar to SEUs. Such soft failures are also impossible to debug because they are gone after the chip is power-cycled or reset; consequently at some point, instead of being able to detect and fix the problem, the user must accept that the equipment is unreliable.

In [15], an architecture level model and tool named SoftArch is presented to determine soft error Mean Time To Failure (MTTF) for a processor with specific workload and to study the contribution of soft errors in different phases of runtime of an application.

In general soft errors can cause errors in computation and data corruption that do not cause the permanent failure in the circuit and hence are not viewed as a long-run reliability issue. Due to above reason, this thesis is mainly defined to focus on hard errors discussed in next section.

1.1.2 Hard Errors

According to Jedec [16], Hard error is an irreversible change in operation that is typically associated with permanent damage to one or more elements of a device or circuit (e.g., gate oxide rupture, destructive latch-up events). The error is called "hard" because the data is lost and the device or circuit may no longer function properly, even after power reset and re-initialization.

Hard errors or hard failures can be further divided into extrinsic failures or defects and intrinsic failures or wear outs. Extrinsic are like birth defects and most of them are detected during the Burn-In process whereas intrinsic (caused by wear and tear) are age related defects that increase over time.

The two have different characteristic lifetimes and two different characteristic parameters. An example is dielectric breakdown and is shown in figure 1.2, wear outs and defects can be clearly separated from each other. The above example is a special case of relatively thick dielectric. It is not always the case that two different failing types can be distinguished due to obvious failure analysis resources and total number of fails observed.

In figure 1.2, the slopes of both extrinsic and intrinsic errors (cumulative failures) are shown with time on x-axis. Useful life period can be seen as when no wear-outs occur, or only extrinsic errors exist. Similarly, wear out period or intrinsic failures occurs when the product reaches the end of its effective life and begins to degenerate and wear out. In detail, these can be classified as, aging, wear, degradation, fatigue, defects, poor servicing or maintenance etc. It is observed that the two types of errors can be easily distinguished in general.

The next two Sections 1.2 and 1.3 focus on the two types of hard errors named extrinsic and intrinsic errors.

1.2 Extrinsic Errors

Extrinsic failures are all the faults induced by process manufacturing or human-interactions which occur with a decreasing rate over time. For example, contaminants on the crystalline

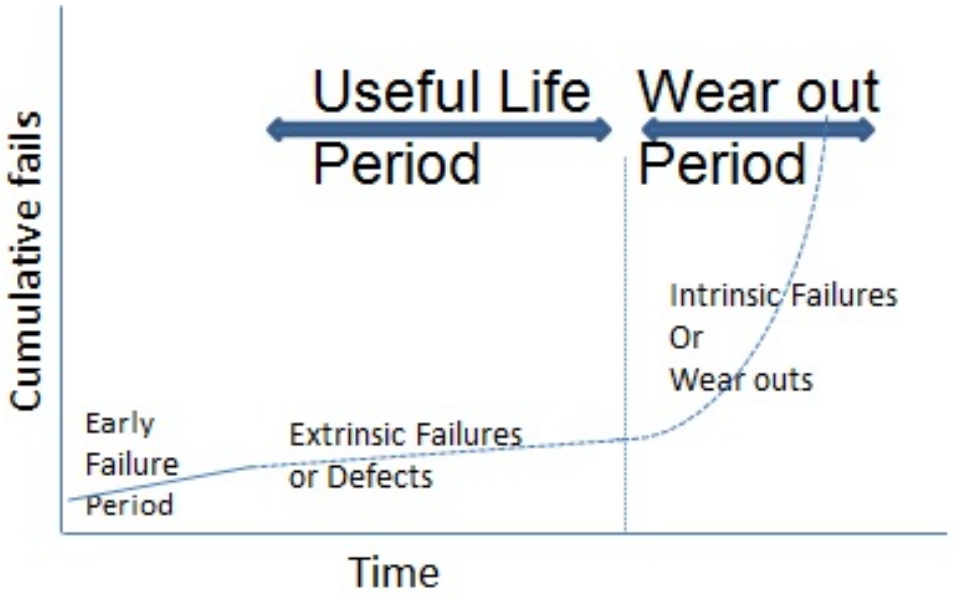

Figure 1.2: Slopes for extrinsic and intrinsic failures with time [2].

silicon surface and surface roughness can cause gate oxide breakdown. Other examples include short circuits and open circuits in interconnects due to incorrect metalization during fabrication. Extrinsic failures are mainly a function of the manufacturing process- the underlying micro architecture has very little impact on the extrinsic failure rate. Defects are typically expressed in terms of end-of-life failures. After manufacturing, using a technique called burn-in, the processors are tested at elevated operating temperatures and voltages in order to accelerate the manifestation of extrinsic failures. Since most of the extrinsic failures are weeded out during burn-in, shipped chips have a very low extrinsic failure rate. Semiconductor manufacturers and chip companies continue to extensively research methods for improving burn-in efficiency, and reduce extrinsic failure rates.

The next Section 1.3 will discuss the type of hard errors which are not created during manufacturing but occur during the useful lifetime.

1.3 Intrinsic Errors

Intrinsic failures are those related to processor wear-out and are caused over time due to operation within the specified conditions. A failure mechanism is caused by an error occurring during the design, layout, fabrication, or assembly process or by a defect in the fabrication or assembly materials. These failures are intrinsic to, and depend on the materials used to make the processor and are related to process parameters, wafer packaging, and processor design. If the manufacturing process was perfect and no errors were made during design and fabrication, all hard processor failures would be due to intrinsic failures. Intrinsic failures occur with an increasing rate over time. These kinds of failures are inherent material property of good and flawless dielectrics which eventually will wear out with time and finally fail at moment of breakdown. It is essential that these failures do not occur during the intended lifetime of the device when it is used under specified operating conditions.

Some of the failures can occur earlier than expected they are called early or defect-driven fails. In these fails, dielectric structure fails not from wear out, but from flaws in dielectric.

Although there are various wear out mechanisms existing in literature, some of them are becoming more important due to scaling. Some of these failure mechanisms are Electromigration, stress migration, hot-carrier injection, time dependent dielectric breakdown, thermal cycling and negative bias temperature instability, which are well documented in the state of the art and presented in following subsections.

1.3.1 Electromigration

On a chip, the wiring is used for number of reasons, including, routing signals in and out of the chip or from one part to another, routing power to various devices, making inductors and capacitors and as interface to external connections. Failure in wiring can be due to open circuit failure, resistance failure, short circuit failure and leakage. Electrical failures from macroscopic point of view, occurs due to applied current which is high enough to cause overheating and burnouts in wires or cause fire in adjacent materials. This heating is known as resistive or joule heating, is power dissipated in the wire, I^2R, Where R is resistance of wire, and I is applied current. This does not stand true in microscopic environment where wires are embedded into hard dielectrics which are connected to thermally conductive Si substrate. So, the wires are kept from burning out until higher current densities are reached. As IC technology increases device density, interconnects that carry signals are consequently reduced in size, specifically, in height and cross section. This leads to extremely high current densities, on the order of at least $10^6 A/cm^2$. At these current densities, momentum transfer between electrons and metal atoms becomes important. The transfer, which is called the electron-wind force, results in a mass transport along the direction of electron movement. Once the metal atoms are activated by the electron wind, they are subject to the electric fields that drive the current. Since the metal atoms are positively ionized, the electric field moves them against the electron wind once they have been activated. The interplay of these two phenomena determines the direction of net mass transfer. This mass transfer manifests itself in the movement of vacancies and interstitials. The vacancies coalesce into voids or micro cracks, and interstitials become hillocks. The voids, in turn, decrease the cross-sectional area of the circuit metalization and increase the local resistance and current density at that point in the metalization. Both the increase in local current density and in temperature increase EM effects. This positive feedback cycle can eventually lead to thermal runaway and catastrophic failure [17]. The above discussed phenomenon is summarized in Figure 1.3.

1.3.2 Time dependent dielectric breakdown

As shown in figure 1.4, TDDB occurs when the oxide breakdown resulting from prolonged electrical stress thus creating a conductive path in the dielectric is short-circuiting some signals. [18]

Although, the exact physical mechanism of TDDB is still an open question, the general belief is that a driving force such as the applied voltage or the resulting tunneling electrons create defects in the volume of the oxide film. The defects accumulate with time and eventually reach

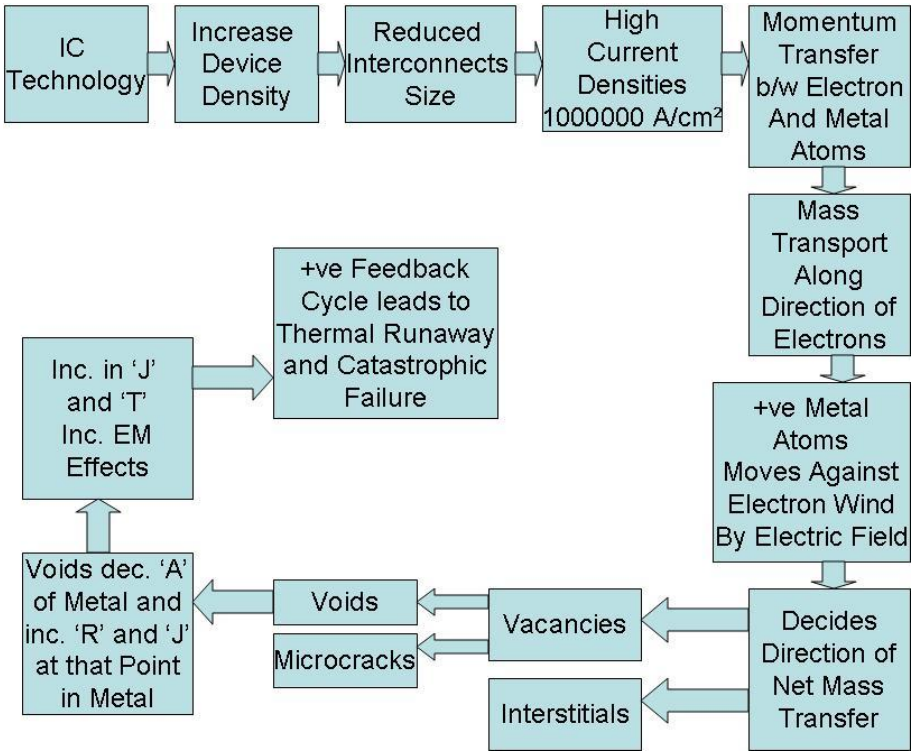

Figure 1.3: Summary EM

a critical density, triggering a sudden loss of dielectric properties. A surge of current produces a large localized rise in temperature, leading to permanent structural damage in the silicon oxide film. When an oxide is stressed electrically, structural defects are generated in oxide and its interface at a rate depending on stress conditions (i.e. voltage and temperature). With these changes in electrical properties of oxide that finally triggers the dielectric breakdown. Either time needed to break voltage stressed oxide is measured (CVS - Constant Voltage Stress), or time of current injection into the oxide after which oxide fails (CCS - Constant Current Stress). The standard TDDB reliability prediction methodologies consider statistics of the time to first breakdown. However, the first breakdown may not be the best definition for device failure, because many circuits ($CMOS$) remain functional after first failure. Some researchers have focused on studying the impact of breakdown on device performance, to finally establish relation between dielectric breakdown and device failure. The criteria to predict device failure is completely depending on the application. Electrons and holes can make transitions between the crystalline states near the silicon-oxide interface to the surface states. These charges will definitely affect the electrical characteristics of devices and are important factors in TDDB. Figure 1.4 shows the names and locations of charges inside silicon dioxide and at the silicon-oxide interface.

1. Interfacial oxide charge: This charge is located within 0.2 nm of the $SiO_2 - Si$ surface. The interfacial oxide charge arises from oxide vacancies, metal impurities and broken bonds due to charge injection.

2. Fixed oxide charge: Fixed oxide charge is a positive charge located some 3 to 5 nm from the $SiO_2 - Si$ interface. Due to the nature of modern electronics, bulk properties

of modern oxides are harder to define. Fixed and trapped oxide charges are generally likely to occur at oxygen vacancy sites.

3. Oxide trapped charge: This charge is also likely to occur at oxygen vacancy sites. The sources of this charge include the oxide growth process, fabrication of device [19], and high-energy electrons. A fabrication-introduced charge can be removed through low-temperature annealing.

4. Mobile Na+ and K+ ionic charge: These charges have been virtually eliminated as a source of reliability problems.

It is the generation of oxide charge states under high electric fields that ultimately leads to dielectric breakdown. There are processes such as Fowler-Nordheim tunneling; direct tunneling and trap-assisted tunneling that contribute to the overall creation and persistence of oxide charges.

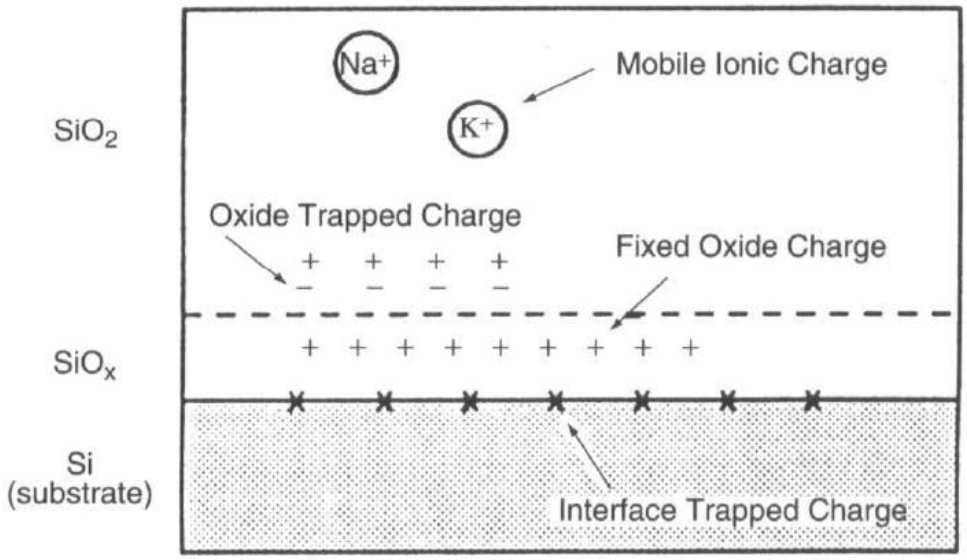

Figure 1.4: Location and identification of charges in $SiO_2 - Si$ and at the oxide-silicon surface

1.3.3 Stress migration

SM in interconnects is due to mechanical stress induced by the difference in thermal expansion rates between metal and oxide in a device. The atoms are then influenced by this mechanical stress gradient, which is proportional to the mechanical stress in a way vacancies moves from low hydrostatic stressed regions to higher ones. This metal movement causes voiding, and the resistance associated may engender electrical failures. It can be noted that little metal movement occurs until the stress exceeds the yield-point of the metalization.

1.3.4 Thermal cycling

TC and cracking can cause permanent damage. Damage from thermal cycling can also accumulate each time the device undergoes a normal power-up and power-down cycle. Such cycles can induce a cyclical stress that tends to weaken materials, and may cause a number of different types of failures. Solder connections are particularly common and important as they can fatigue to failure under thermo-mechanical stress, commonly driven by mismatch in thermal expansion coefficient and Young's modulus (In solid mechanics, Young's modulus (E) is a measure of the stiffness of an isotropic elastic material).

1.3.5 Hot Carrier Injection

The phenomenon "Hot Carrier Injection" (HCI) is due to the ionization caused by the impact of electrons on the silicon atoms at the drain. The ionization generates electron-hole pairs that enter the substrate and causes the increase of current in the substrate. Part of the carriers created can then cross the potential barrier layer of gate oxide. The HCI reduces the mobility of charge carriers which increases the switching time. Delays, if they appear on the critical path, lower the maximum switching frequency of the transistor. This phenomenon is more important at low than at high temperature because the electrons are more mobile and thus they have higher energy during ionization.

This can be assessed by measuring the saturation current of the drain I_{Dsat} which is one of the parameters affecting the speed of a transistor. Damage caused by HCI on the gate oxide increases the threshold voltage of the NMOS transistor which decreases the I_{Dsat} current. The current flowing in the channel is at the maximum during switching that is when the HCI phenomenon is maximal. The HCI is a failure that occurs when processor is active. It is important to note that this is not destructive: the structure of the circuit is not changed, so it can be regenerated [20].

1.3.6 Negative Bias Temperature Instability

NBTI is a wear out mechanism experienced by *PMOSFET*s with the channel in inversion. It is believed that NBTI is controlled by an electrochemical reaction where holes in the *PMOSFET* inverted channel interact with Si compounds (Si-H, Si-O, etc.) at the Si/SiO2 interface to produce donor type interface states and possibly positive fixed charges [3]. NBTI damage is generated by cold holes (thermalized) in the inverted channel. Attention must be paid not to confuse this mechanism with *PMOSFET* damage generated by possible impact ionization at high VG regime which produces hot holes damage. The relative contribution of the NBTI induced interface states generation and positive fixed charge formation is very sensitive to the gate oxide process used in the technology. The electrochemical reaction is strongly dependent on the gate oxide electric field (V_g/t_{ox}) and the channel temperature. The NBTI damage may lead to substantial *PMOSFET* parameter changes, in particular to an increase of the absolute value of the threshold voltage (transistor is harder to turn on) as well as mobility degradation with consequent reduction in drive current.

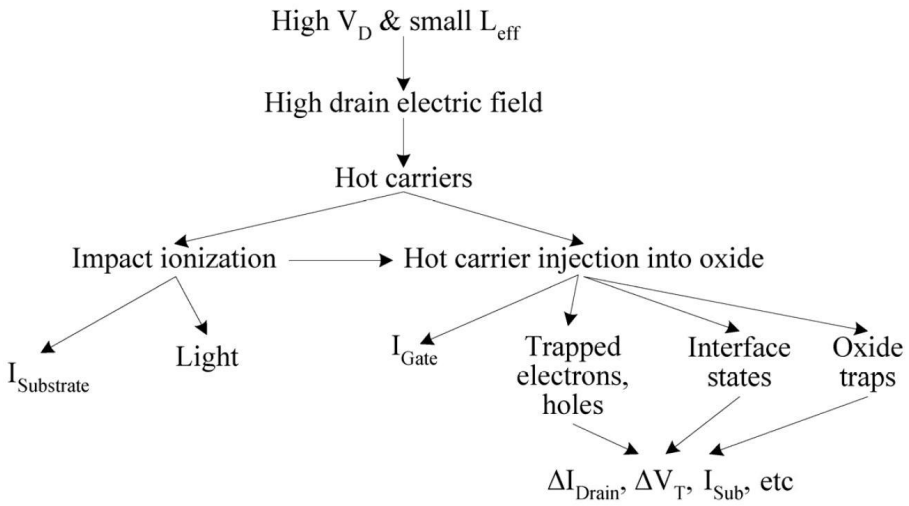

Figure 1.5: Hot carrier generation and degradation in MOSFETs

A given *PMOSFET* in a circuit is exposed to the NBTI damage as long as it operates in inversion. For this reason NBTI is sensitive to stand-by conditions ('0' input on an inverter for example), contrary to Hot Carrier Injection, which is typically only active during voltage transients.

This Section discussed about various type of intrinsic hard errors that occur during and after the useful lifetime of the chip in present day technologies. The next Section 1.4 will discuss a little about the coming technology, i.e., 3D ICs and which type of issue researchers should focus on, in the future technologies.

1.4 Reliability in 3D ICs

To finish the first chapter, let us see what future holds in terms of 3D IC technology and which reliability issues can/may occur in this promising technology. System-level integration is expected to gradually become a reality because of continuing aggressive device scaling for 2-D dies and emergence of 3-D integration technology. Chip power density, which is already a serious issue due to exponential increase every year (in comparison to exponential increase in number of transistors in a processor which is following Moore's law), and the related thermal issues in 3-D IC chips, may pose serious design problems unless properly addressed now. Such temperature-related problems include material as well as electrical reliability, leakage power consumption and possible regenerative phenomena such as avalanche breakdown. During the past four decades, semiconductor technology scaling has resulted in a sharp growth in transistor density. Figure 1.6 shows the number of transistors of Intel processors since 1971 [21].

3D integrated circuit technology is an emerging technology for the near future, and has received tremendous attention in the semiconductor community. With the 3D integrated circuit, the temperature and thermo-mechanical stress in the various parts of the Integrated Circuit (IC) are highly dependent on the surrounding materials and their materials properties,

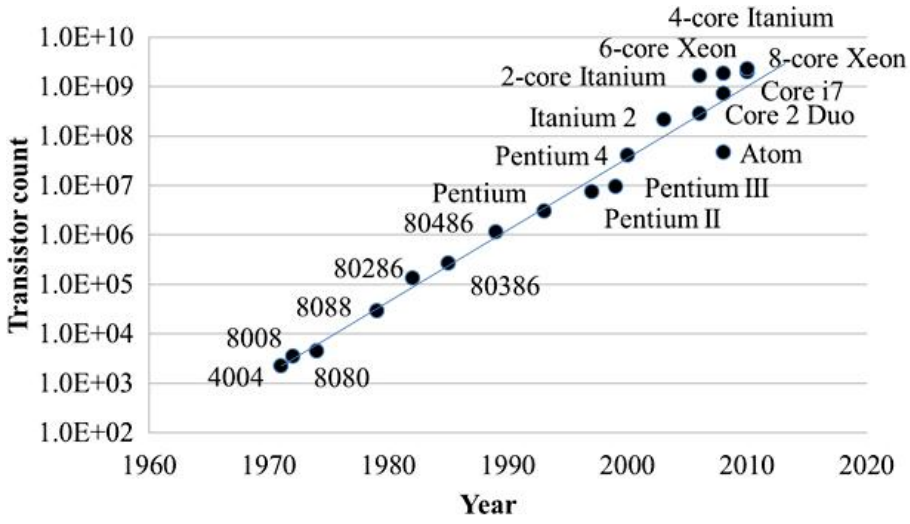

Figure 1.6: Count of transistors in Intel processors 1970-2010

including their thermal conductivities, thermal expansions, Young modulus, Poisson ratio etc. Also, the architecture of the 3D IC will also affect the current density, temperature and thermo-mechanical stress distributions in the IC. In relation of the above-mentioned, the electrical thermo-mechanical modeling of integrated circuit can no longer be done with a simple 2D model. The distributions of the current density, temperature and stress are important in determining the reliability of an IC.

3D stacking of dies is a very promising technique to allow scaling i.e., miniaturization and performance enhancement through the reduction of interconnect lengths in microelectronic systems [22]. Problems related to thermal management in the 3D stacks are believed to be the main challenges for 3D integrations [23]. The use of adhesives that are poor thermal conductors, the vertical integrations of the chips and the reduced thermal spreading due the aggressively thinned dies cause these thermal management issues. When there are hotspots, these thermal effects are even more considerable. Due to this, as in 2D IC, the same power dissipation in a 3D stack will lead to even higher temperatures and more pronounced temperature peaks in a stacked die package compared to a single die package. Due to the complexity of the interconnection structures and through-Si vias, the thermal behavior of a stacked die structure becomes more complicated [24]. To conclude, the 3D ICs need new power management techniques because of their different thermal characteristics (i.e., heterogeneous thermal coupling and cooling efficiency) compared with 2D ICs.

1.5 Conclusion

In this chapter, first of all we discussed the basics of a MOS transistor and its working. Then, we studied various errors already known to researchers that cause problems for the user of device and question its reliability. Power and temperature are two important stress factors in semiconductor device reliability analysis. At the functional level, due to the complexity of VLSI circuit and dynamic operating conditions, it is a very complicated process to estimate power and temperature and hence to predict reliability. In the next chapter, we will see how

the reliability mathematics works and how to handle the different parameters from physics of each failure mechanism are modeled. Various models have been proposed to describe the effect for a single failure mechanism, because of the unique physical process underlying each failure mechanism, e.g., Electromigration (EM), hot carrier injection (HCI), time dependent dielectric breakdown (TDDB), and negative bias temperature instability (NBTI). A failure-mechanism-based qualification methodology using specifically designed stress conditions over traditional approaches (i.e., one voltage and one temperature) can lead to improved reliability predictions for targeted applications and optimized burn-in, screening, and qualification test plans.

Reader may observe the focus on intrinsic errors and not on extrinsic errors. It is due to the fact that effect of run-time applications causes fails in a device which are intrinsic in nature. Also, most of the extrinsic errors are removed during burn-in process before shipping the chips.

Out of various presented failure mechanisms 4 failure mechanisms are considered and explained in details, Electromigration, TDDB, HCI and NBTI. The following chapters will continue to provide mathematical models existing at transistor level of abstraction and derived models for the four failure mechanism at functional level of abstraction. These derivations will make clear the relation between power consumption, temperature and these four failure mechanisms. Also, readers will understand that it is possible to simulate power consumption and temperature (with enough accuracy) at higher level of abstraction. All models are based on the physics provided in current chapter.

Chapter 2

Failure models at functional level

Contents

In a single transistor, all the simulations are done at the logic level through the use of four different values: 0 (logic 0), 1 (logic 1), Z (high impedance) and X (unknown logic value). Transistors are represented by ideal switches that can be either conducting or non-conducting. Semiconductor devices made of these transistors are very sensitive to impurities and particles [25]. Therefore, to manufacture these devices it is necessary to manage many processes while accurately controlling the level of impurities and particles. The finished device quality depends upon the many layered relationship of each interacting substance in the semiconductor, including metallization, chip material (list of semiconductor materials) and package. Due to the rapid advances in technology, many new devices are developed using new materials and processes, and design calendar time is limited due to non-recurring engineering constraints, plus time to market concerns. Consequently, it is not possible to base new designs on the reliability of existing devices. To achieve economy of scale, semiconductor products are manufactured in high volume. Furthermore repair of finished semiconductor products is impractical. Therefore incorporation of reliability at the design stage and reduction of variation in the production stage have become essential. Reliability of semiconductor devices may depend on assembly, use, and environmental conditions. In this chapter, we begin with a small discussion about mathematical parameters that are used in estimating and predicting reliability. The mathematics discussed provide the reason to define a new parameter which suits our needs to analyze the effect of past and present stress on a complete chip and predict the reliability at specified time in future for different failure mechanisms separately.

In the next Section 2.2, the mathematical relations have been studied about various stress factors that affect aging at transistor level. Then, failure modeling techniques at present era at higher level of abstraction are discussed. The last section gives the new relations that have been derived to use and embed the transistor level models in models at higher level of abstraction.

2.1 Reliability Mathematics

The following discussion of reliability mathematics is limited to semiconductor reliability mechanism, more details and derivations for given parameters can be found in [26]. Let us start the discussion with definitions given for failures that are used in reliability domain. Failures due to wear, intrinsic failures and end-of-life failures are considered at the end of the expected life of the product. Defects are typically expressed in terms of end-of-life and different points of time throughout the life. Terminology generally used for cumulative fails is parts per million (ppm), and for failure rates is fails per 1000hrs and fails per billion device hours (FITs). The provided mathematics will be used to define a new parameter called CFR (Cumulative Failure Rate) in Section 2.4. The following subsections are provided to discuss existing parameters that are used by designers to estimate and predict reliability.

2.1.1 Cumulative Distribution Function (CDF)

CDF (F(t)) is cumulative sum of failing population for discrete function. The CDF for continuous function is the integral of Probability Density Function (PDF is the function to

distinguish fails in each period of time). It provides the fails that occurred in the past. An expression is used to describe the fails in these past time steps, and this expression is used to predict fails that would be expected in future.). The CDF is related to continuous PDF.

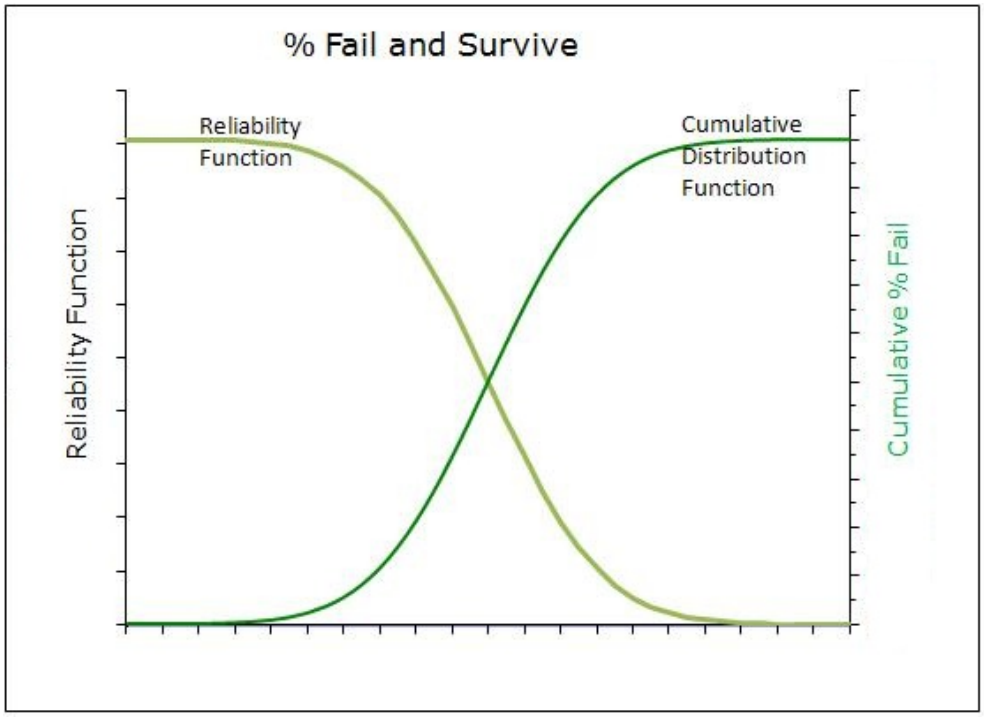

Figure 2.1: Fail and survive

2.1.2 Reliability Function (RF)

It is the cumulative surviving population. It is calculated by subtracting the CDF from 1. So, $R(t) = 1 - F(t)$. The reliability function should be equal to 0, after the last fail, since by definition there should be no survivors in the end. In figure 2.1, the relation between CDF and RF is shown for arbitrary values.

2.1.3 Hazard Function

Hazard function, λ(t) or Instantaneous failure rate, another important function that is used in reliability community can be defined as probability of those parts that have not failed until time t, but will fail in next time interval, Δt. Mathematically, λ(t) can be expressed in probability divided by time interval i.e., fails per time or fraction failing per time. Now, λ(t) can be defined as in Eq.2.1:

$$\lambda(t) = \frac{f(t)}{1 - F(t)} = \frac{f(t)}{R(t)}, \tag{2.1}$$

where f(t) is the time to first failure distribution and R(t) is 1 - F(t). The Hazard function is mainly used to describe defects during the normal life of the device. As PDF is to CDF, Hazard function is to Cumulative hazard. Cumulative hazard is the integral of hazard function.

2.1.4 Mean-Time-To-Failure (MTTF)

MTTF is nothing more than the expected value of time to failure and is derived from basic statistical theory as in Eq.2.2:

$$MTTF = \int_0^\infty t \cdot f(t) \cdot dt, \tag{2.2}$$

Integrating Eq.2.2 by parts and applying "Hopital's rule," as derived in [27] gives Eq.2.3:

$$MTTF = \int_0^\infty R(t) \cdot dt, \tag{2.3}$$

The above Eq.2.3, in general cases, allows the simplification of MTTF calculations. If user knows (or can model from the data) the reliability function, R(t), the MTTF can be obtained by direct integration of R(t), by graphical approximation, or by Monte Carlo simulations. For repairable equipment MTTF is defined as the mean time to first failure.

2.1.5 Mean Life

The mean life (θ) refers to the total population of items being considered. For example, given an initial population of n items, if all are operated until they fail, the mean life (θ) is merely the arithmetic mean time to failure of the total population given by Eq.2.4:

$$\theta = \frac{\sum_{i=1}^{n} t_i}{n}, \tag{2.4}$$

where, t_i = time to failure of the i_{th} item in the population and n = total number of items in the population.

While studying the different mathematical parameters, it is important to study the configurations of the system for which the mathematics has to be provided. These configurations are studied in following subsection.

2.1.6 Architectural configurations of Electronic systems

Some major architectural configurations of electronic systems are very common, and the analysis of their reliability behavior forms the foundation of the analysis of any complex system. In the serial configuration, depicted in the left part of figure 2.2, several blocks, n, with failure rates $R_1(t), ..., R_n(t)$ considered independent of each other are cascaded. The correct operation of the system depends on the reliability of each block and is mathematically expressed as in Eq.2.5:

$$R_{system} = R_1(t) \cdot R_2(t) \cdot \bullet \bullet \bullet \cdot R_n(t) = \prod_{i=1}^{n} R_i(t), \tag{2.5}$$

In the parallel configuration, depicted in the right part of figure 2.2, considering redundant circuits (all blocks are same), malfunction of all composing blocks is necessary to cause the system to fail. Naming the probability of failure or unreliability of the components $F_i = 1 - R_i$ and omitting the expression of time (t) for clarity, the probability of failure of the system is expressed as in Eq.2.6:

$$F_{system} = \prod_{i=1}^{n} F_i, \tag{2.6}$$

The reliability of the system composed of parallel implementation is expressed as in Eq.2.7:

$$R_{system} = 1 - F_{system} = 1 - \prod_{i=1}^{n} (1 - R_i), \tag{2.7}$$

and can be higher than the reliability of individual components because redundancy is applied. Realistic designs are typically composed of hybrid arrangement of parallel and serial configurations, where the system reliability can be obtained by iterative decomposition of the network into its series and parallel components and step-by-step solving. Finally, a system in a $k - out - of - n$ configuration consists of n components. Only k components need to function properly to enable the full system to operate.

2.1.7 Synthesis

Concluding this section, we have studied different mathematical parameters that are somehow related to each other as shown by the use of equations. These parameters are used by researchers and will be discussed in next section. In Section 2.4, a new parameter is defined to be more suitable at higher level of abstraction in comparison to the ones used by tools and methodologies in Section 2.3.

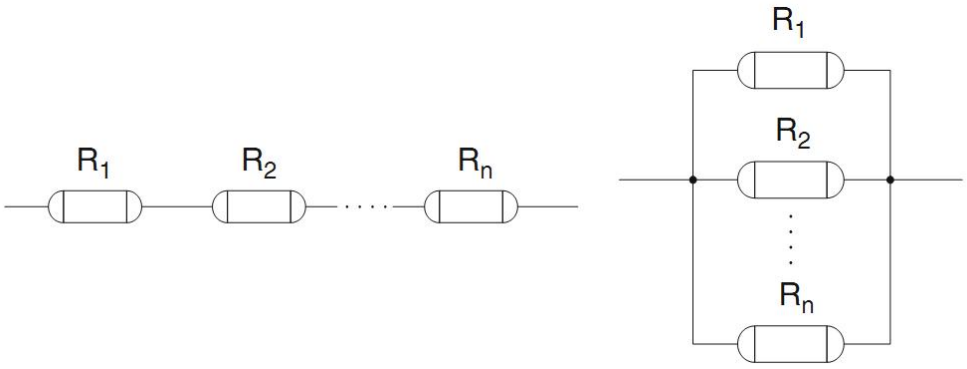

Figure 2.2: Electrical component configurations: serial in the left and parallel in the right [25]

2.2 Failure rate models at Transistor level (for EM, HCI, TDDB, NBTI)

Failure modeling is a key to reliability engineering. Validated failure rate models are essential to the development of prediction techniques, allocation procedures, design and analysis methodologies and test and demonstration procedures. Or, we may say, all of the elements needed as inputs for sound decisions to insure that an item can be designed and manufactured so that it will perform satisfactorily and economically over its useful life. Inputs to failure rate models are operational field data, test data, engineering judgment, and physical failure information. These inputs are used by the reliability engineer to construct and validate statistical failure rate models (usually having one of the distributional forms described previously) and to estimate their parameters. In previous chapter, physics of failure mechanisms have been discussed. In this section, we will read about some existing failure models at transistor level of abstraction that have been validated for specific conditions.

In this section, mathematics of different failure mechanisms is given at transistor level that are affected by one or more of above discussed stress factors.

2.2.1 Black's Law for Electromigration (EM)

EM, the dominating failure mode of interconnects, is characterized by the migration of metal atoms in a conductor through which large direct-current densities pass [28]. Although EM has been intensely studied for more than 40 years, many aspects of EM are still not well understood. This lack of understanding is caused by two related issues: the existence of many factors that influence EM and the inability to isolate the effect of these factors experimentally. These factors include grain structure, grain texture, interface structure, stresses, film composition, physics of void nucleation and growth, thermal and current density dependencies, etc. [28]. According to experimental research, current density and temperature are among the most important factors. Black [29] developed an empirical model relating the median time to failure (t_{50}) of a metal line to the temperature (T) and current density (J); the model has the form as in Eq.2.8:

$$t_{50} = \frac{A}{J^2} \cdot exp(\frac{E_a}{K \cdot T}), \tag{2.8}$$

where A is a material and process-dependent constant and E_a is the activation energy for the diffusion processes that dominate the temperature range of interest. The importance of current density and temperature is shown in this equation. Also, as expected, the scaling of interconnects, in last 40 years has increased current densities and temperature, thereby greatly reducing the median time. The reliability of the IC has decreased simultaneously. To better understand the interconnect-scaling effect, physical models and statistical models have to be carefully developed.

A Generalized Black Model has been proposed to characterize EM failures [30]. According to Lloyd [30], Black's equation is not always strictly obeyed. Where, often n could be found to vary substantially from 2, ranging from as low as 1, but in fact rising without limit in various extreme cases. Very high values of n can be attributed to over stressing with too high a current density, lead to meaningless test results. The failure mode in these cases would be due to the presence of temperature gradients that should not be present in properly designed products in ideal world. There are, however, other reasons that n may be high, approaching infinity, unrelated to temperature gradients that need to be considered. Lloyd generalized Black's law as in Eq.2.9:

$$MedianTimetoFailure = t_{50} = A \cdot J^{-n} \cdot exp(\frac{E_a}{K \cdot T}), \tag{2.9}$$

The various values of n and m are determined by the particular failure physics and the conductor's geometry. If $n = 2, m = 0$, we have the original Black model.

2.2.2 Takeda's Model for Hot-Carrier Injection (HCI)

As discussed in Chapter 1, in Section 1.3.5 concerning the physics behind HCI, the discussion will be continued in this subsection. Takeda gave failure model based on the physics, according to Takeda [31, 32, 33], there are three main types of hot carrier injection modes: 1. Channel hot electron (CHE) injection. 2. Drain avalanche hot carrier (DAHC) injection. 3. Secondary generated hot electron (SGHE) injection.

CHE injection is due to the escape of "lucky" electrons from the channel, causing a significant degradation of the oxide and the $Si - SiO_2$ interface, especially at low temperature (77 K). On the other hand, DAHC injection results in both electron and hole, gate currents due to impact ionization, giving rise to the most severe degradation around room temperature. SGHE injection is due to minority carriers from secondary impact ionization or, more likely, bremsstrahlung radiation, and becomes a problem in ultra-small metal oxide semiconductor (MOS) devices. Fowler-Nordheim tunneling and direct tunneling might also cause hot carrier injection. For deep sub-micrometer devices, it is important to attempt to account for the effects resulting from combinations of some if not all of these injection processes.

Power law model is an empirical model which was proposed by Takeda [34] based on the following assumptions:

1. Avalanche hot carrier injection due to impact ionization at the drain, rather than channel hot electron injection composed of "lucky electrons," imposes the severest constraints on device design;

2. Device degradation (V_{th} shift and G_m (trans-conductance) change) resulting from drain avalanche hot carrier injection has a strong correlation to impact ionization induced substrate current.

The V_{th} shift, ΔV_{th}, or G_m degradation, $\Delta G_m/G_{m0}$, can be empirically expressed as in Eq. 2.10:

$$\Delta V_{th} = \Delta G_m/G_{m0} = A \cdot (t^n), \tag{2.10}$$

This expression is particularly valid for short stress times, while for long stress times, ΔV_{th} and/or $\Delta G_m/G_{m0}$ begins to saturate. The slope n or ΔV_{th}, in a log plot is strongly dependent on V_G, but has little dependence on V_D. This suggests that n changes according to the hot carrier injection mechanism. The magnitude of degradation, A, is strongly dependent on V_D and has little dependence on V_G. We can write the Eq. 2.11:

$$A \propto exp(\frac{-\alpha}{V_D}) \tag{2.11}$$

Therefore, the lifetime (or time to failure) τ can be expressed as in Eq. 2.12:

$$\tau \propto exp(\frac{e}{V_D}), \tag{2.12}$$

where $e = \alpha/n$.

Takeda and Hu [34, 35] both reported $\tau \propto I_{sub}^m$, while m ranging between 3.2 and 3.4 given by Takeda and 2.9 by Hu. Also, hot carrier effects are enhanced at low temperature. The main reason is an increase in electron mean free path and impact ionization rate at low temperature. As shown in [36, 37], substrate current at 77 K is five times greater than that at room temperature, and CHE gate current is approximately 1.5 orders of magnitude greater than that at room temperature. At low temperature, the electron trapping efficiency increases and the effect of fixed charges becomes large [34]. This accelerates the degradation of Gm at low temperature. The degradation of V_{th} and G_m at low temperatures is more severely accelerated for CHE-induced effects than for DAHC. Hu [35] showed the temperature coefficient of CHE gate and substrate current to be negative. In [31], time dependence of device degradation by drain avalanche hot-carrier injection is explained.

To conclude, with the help of above discussion, we can rewrite time to failure at transistor level for HCI using Takeda model as in Eq. 2.13:

$$ttf_{tr-HCI} = \frac{A_{HCI}^n}{exp(\frac{-e}{V_{DD}}) \cdot exp(\frac{-E_a}{kTi})}, \tag{2.13}$$

where, A_{HCI} is technology dependent constant.

2.2.3 E-Model for time dependent dielectric breakdown (TDDB)

The thermo-chemical model (E model) is a widely accepted and cited dielectric breakdown model. McPherson [38] reviewed the development of this model and proposed a physical explanation. This model proposes that defect generation is a field-driven process, and the current flowing through the oxide plays at most a secondary role. The interaction of the applied electric field with the dipole moments associated with oxygen vacancies (weak $Si - Si$ bonds) in SiO_2 lowers the activation energy required for thermal bond breakage and accelerates the dielectric degradation process. Eventual charge trapping at the broken bond sites and their wave function overlap lead to a conduction subband formation. Consequently, severe Joule heating occurs at the stage of oxide breakdown. McPherson [38] also showed that allowing for a distribution of energies of the weak bonds could account for a wide range of observations of the temperature and field dependence of dielectric breakdown times. The E model suggests time-to-breakdown (T_{BD}) is given by Eq. 2.14:

$$T_{BD} = A_0 \cdot exp(-\gamma \epsilon_{ox}) \cdot exp(\frac{E_a}{kT}), \tag{2.14}$$

where, A_0 is arbitrary scale factor, dependent upon materials and process details, γ is field acceleration parameter, temperature dependent with $\gamma(T) = a/kT$, where a is the effective dipole moment for the molecule, ϵ_{OX} is externally applied electric field across the dielectric.

The E model has attained widespread acceptance on the basis of experimentally verified exponential dependence of T_{BD} on the electric field [39].

2.2.4 Power Law for Negative Bias Temperature Instability (NBTI)

NBTI occurs to p-channel MOS (PMOS) devices stressed with negative gate bias at elevated temperatures [40]. Bias temperature stress under constant voltage (DC) causes the generation of interface traps (N_{IT}) between the gate oxide and silicon substrate, which translate to device threshold voltage (V_{th}) shift and loss of drive current (I_{on}). The time dependence of the threshold voltage shift (ΔV_{TH}) is found to follow a power-law model provided in Eq. 2.15:

$$(\Delta V_{TH})(t) = A \cdot (t^n), \tag{2.15}$$

where A is a constant that depends on oxide thickness, field, and temperature. The theoretical value of the time dependence parameter n is 0.25 according to the solution of diffusion equations [41]. Reported value of n is in the range from 0.2 to 0.3. According to Chakravarthi [42], the values of n vary around 0.165, 0.25, and 0.5 depending on the reaction process and the type of diffusion species. The temperature dependence of NBTI follows the Arrhenius law with activation energies ranging from 0.18 to 0.84 eV [43]. Improved models have been proposed after the simple power-law model. Considering the temperature and gate voltage, ΔV_{th} can be expressed as in Eq. 2.16:

$$\Delta V_{th}(t) = A \cdot exp(\beta \cdot V_G) \cdot exp(\frac{-Ea}{kT}) \cdot t^{0.25}, \tag{2.16}$$

where A and β are constants and V_G is the applied gate voltage. From above discussion we can conclude that failure due to NBTI depends upon oxide thickness, field, temperature and gate voltage as shown in Eq. 2.17:

$$MTTF_{NBTI} = A \cdot {V^n}_G \cdot exp(\frac{-Ea}{kT}), \tag{2.17}$$

A summary is presented in Table 2.1, to compile different failure mechanisms and their failure models at transistor level, as it has been studied in Section 2.2

Table 2.1: Failure models at transistor level of abstraction

Failure Mechanism	Failure Model (TTF)	Parameters Involved
Electromigration	$A \cdot J^{-n} \cdot T^{-m} \cdot exp(\frac{E_a}{K \cdot T})$	A, n, m: Process related constant; J: Current density; T: Temperature in Kelvins; E_a: Activation energy; k: Boltzmann's constant
Hot carrier injection	$\frac{A^n_{HCI}}{exp(\frac{-e}{V_{DD}}) \cdot exp(\frac{-E_a}{kT})}$	e: factor involving shift in threshold voltage; V_D: Drain Voltage; A_{HCI}: Field acceleration parameter
Time dependent dielectric breakdown	$A_0 \cdot exp(-\gamma \epsilon_{ox}) \cdot exp(\frac{E_a}{kT})$	A_0: Arbitrary scale factor; γ: Field acceleration parameter; ϵ_{ox}: externally applied electric field
Negative bias temperature instability	$A \cdot {V^n}_G \cdot exp(\frac{-Ea}{kT})$	A: Technology parameters; V_G: Gate Voltage

2.2.5 Synthesis

Now, all the parameters are discussed that are believed to be involved in causing intrinsic type errors using physics and mathematics. It is a good point to look at present day failure modeling techniques at higher level of abstraction.

2.3 Failure modeling at higher level of abstraction in present days

Reliability system simulations are an integral part of the designing to make the end-product immune to factors that could adversely affect reliability. These simulators use the models of the most significant physical failure mechanisms in modern electronic devices, such as TDDB, NBTI, EM, and HCI. In present, integrated chips are composed of millions of transistors.

So, functional level reliability prediction methods are based on statistics obtained through experiments. These reliability prediction tools model the time to failure (TTF). But, they do not predict lifetime due to random post burn-in errors. Various semiconductor companies provides an expected TTF for specific operating voltage and frequency, heat dissipation, etc.

In recent years, without a doubt there is improvement in the semiconductor industry's understanding of the reliability physics of semiconductor devices. Each failure mechanism has been studied in details and very accurately modeled for a given technology. The High Temperature Operating Life (HTOL) or steady-state life test, generally shows that during testing there can be multiple intrinsic failure mechanisms, it means, none of the discussed failure mechanisms can dominate during operation of a device. So, to derive a model for reliability, a good approximation is to assume that all mechanisms have equal probability to occur [44]. Since, failure rate is very slow, and it takes years for a device to fail, hence, for experimental studies industry use the parameter called Acceleration Factor (AF). The acceleration of a single failure mechanism is a highly non-linear function of temperature and/or voltage. The temperature acceleration factor (AFT) and voltage acceleration factor (AFV) can be calculated separately or together [18], this is the subject of most studies concerning reliability physics. The acceleration factor can be calculated as combinations of the product of the acceleration factors of temperature and voltage.

Each mechanism competes with the others to cause an eventual failure. For more than one mechanism, the relative acceleration of each one must be defined and averaged for specific conditions. For each mechanism identified, its unique AF be calculated at given conditions of temperature and voltage, each mechanism leads to an expected failure rate. These failure mechanisms are as already explained not uniformly accelerated by HTOL test, due to which, the manufacturer has to model a single acceleration factor that cannot be combined with the known physics-of-failure models.

Although we will see the various reliability prediction methodologies and simulators in next Chapter, some of these macro-modeling techniques are introduced here that gave new dimension to circuit level reliability modeling.

2.3.1 FaRBS

One of the circuit-level methodology that is FaRBS (Failure Rate Based Simulation Program with Integrated Circuit Emphasis) [45] is based on the physics-of-failure and sum-of-failure-rates (SOFR) models. FaRBS combines modules of SPICE (simulation program with integrated circuit emphasis), semiconductor wear-out models, IC system reliability models, AF models, and the SOFR reliability model.

2.3.2 MaCRO

An integrated circuits emphasis (SPICE) simulation method is MaCRO (Maryland Circuit-Reliability Oriented) [46] that was developed based on the failure equivalent circuit-modeling techniques. MaCRO uses a series of accelerated lifetime models and failure-equivalent circuit models for common silicon intrinsic wear out mechanisms, including HCI, TDDB, and NBTI [18]. For system designers MaCRO seems very promising for future-generation technologies

to better prepare for the reliability challenges. Flow in MaCRO is simple; the SPICE is only called for a limited number of times to simulate the impact of the device wear out on circuit functionality.

2.3.3 RAMP

RAMP [47] uses the Arrhenius model to show the dependence of processor failures on temperature; due to the direct processor reliability relation to the operating temperature, it is expected that many reliability problems are the result of elevated processor temperature. To calculate MTTF, RAMP assumes all failure mechanisms have constant failure rates. This is an inaccurate assumption; however, it allows RAMP to combine different failure mechanisms and provides a unified MTTF. The MTTF is calculated as the inverse of the failure rate, assuming a constant failure rate. The reliability model in RAMP is the sum-of-failure-rates (SOFR) model. RAMP considers each block on a chip as a separate component that can fail in different ways corresponding to various failure mechanisms. The dominant component failure mechanism is determined by "competing risk model", and the "series model" estimates the system failure rate (based on the failure rate of each block). In competing risk model to calculate the failure rate of a component, RAMP assumes:

1. Each failure mechanism proceeds independently of every other, at least until a failure occurs.
2. The component fails when the first of all competing failure mechanisms reaches a failure state.
3. Each of the failure mechanisms has a known life distribution model.

The "series model" is applied to estimate the systems reliability based on different blocks. With same assumptions used for the "competing risk model," i.e., a system consisting of j components fails when the first component fails, the MTTF of the system is given by Eq. 2.18:

$$MTTF_{SYS} = \frac{1}{\sum_{i=1}^{j} MTTF_i} = \frac{1}{\sum_{i=1}^{j} \lambda_i} = \frac{1}{\sum_{i=1}^{j} \sum_{l=1}^{k} \lambda_{il}}, \tag{2.18}$$

where λ_i is the failure rate of the i_{th} component and λ_{il} is the failure rate of the i_{th} component due to the l_{th} failure mechanism.

2.3.4 Synthesis

In the author's knowledge, there exist some tools and methodology developed in the past to simulate reliability models at high level. It is still a new science that has to be developed. The science needs more inputs from researchers to provide results close to real world. In the following section, a new mathematical parameter is defined and used to derive models for various failure mechanisms that are already modeled at transistor level.

2.4 Derivations for failure models at functional level

Previous section 2.3 provided some existing methodologies to model reliability at circuit level and has given a brief overview of various parameters existing and used in general for reliability prediction. However, for our need and conditions, a simple but effective and easily understandable parameter is defined named Cumulative Failure Rate. CFR represents the cumulative failure rate over the time of a failure mechanism. As discussed, the mathematical definitions for the existing parameters like Cumulative Distribution Function and Hazard Function, it is important to define CFR, in a manner such that, it is not misunderstood by researchers or engineers from different domains. CFR takes into consideration all the past behavior, hence, it reflects the real stress on the device during the life. It can be extrapolated to show the effect in the future after long time continuous stress. CFR also takes into account, each failure mechanism independent to each other. CFR can be mathematically represented as in Eq. 2.19:

$$CFR_x(n, bl) = \sum_{i=1}^{n} (\lambda_x(a, bl) \cdot t_a), \tag{2.19}$$

where λ is the failure rate as discussed in Section 2.1.3, a is the simulation time step of duration t_a, bl is the current block (a block in the floorplan, can be defined at various abstraction levels) failure rate at time t_i, n is the total number of steps (or instructions executed), bl is the block reference and x is the failure mechanism reference. It must be noted at this point that CFR in present form is defined specifically for applications running on processor(s).

2.4.1 Assumptions to switch from transistor level of abstraction to functional level

The failure rate is derived of a digital block composed of more than one transistor with N transistors switching their output at a particular instant, by applying the following assumptions:

1. the failure rate of a block is a constant value during a cycle t_i. Hence, the Mean Time to Failure (MTTF) is the reciprocal of the failure rate;
2. the equivalent failure rate of tr components is based on a Series model in which the first device failure always causes the block failure, i.e., $CFR_{bl} = \sum CFR_{tr}$, where, tr is the number of transistors in a device;
3. same transistor geometries and doping with same via/contact cross-section areas and hence same capacitances, i.e., all transistors age in a same manner considering no process variations;
4. from previous assumption, it is easy to understand next assumption, that all CMOS gates have an identical fan-out;
5. block area is proportional to the number of transistors and hence via/contact (N);

6. and the number of switching transistors are equal to the switching probability, i.e., all transistors age irrelative to the functionality of the block they belong to.

The provided assumptions are used for the next subsections for different failure mechanisms. It should be understood that these assumptions are necessary to provide a probability factor while switching from transistor level to higher level of abstraction.

2.4.2 Electromigration

Let us see, how the block failure rate is modeled from the knowledge of physics and failure models of EM at device level discussed in Section 1.2 and Section 2.2 respectively. From the physics of EM, it is clear that there are various parameters involved. Some of these parameters are not modeled and only the parameters and conditions that activate EM are taken into account. One important assumption in modeling EM at functional level is that Electromigration is predominant in contacts/vias located in power rails and CMOS gate outputs and the effects of EM are neglected in inter-gate wiring. To conclude, EM occurs when transistor switches, and depends on current operating voltage, frequency and stress time. The failure rate for EM of a single wire or via or contact is given using the well-known Black's law from Eq. 2.20:

$$\lambda_{tr-EM}(a) = \frac{j^n(a)}{A_0} \cdot exp(-\frac{E_{aEM}}{K \cdot T_{tr}(a)}), \tag{2.20}$$

Where j is the instantaneous current density that flows in the item, A_0 is the combination of technology-dependent constants and the second part is from Arrhenius equation, where, E_a is activation energy, K is Boltzmann constant and T is the junction temperature. The instantaneous current density j that flows through a via/contact during a clock cycle can be expressed as in Eq. 2.21:

$$j(a) = \frac{i(a)}{S} = \frac{P_{tr}(a)}{V_{DD} \cdot S}, \tag{2.21}$$

Where i is the instantaneous current, S is the wire/via/contact cross-section area, V_{dd} is the operating voltage and P_{tr} is the power consumption (sum of static and dynamic).

Therefore, the instantaneous current density (Black's law) of a block is assumed to be replaced by the mean current density bringing into play the switching probability, the mean dynamic power and the number of items. From that, the failure rate of block bl for failure mechanism x i.e., Electromigration at time t_a can be expressed as in Eq. 2.22:

$$\lambda_{bl-EM}(a) = N \cdot \frac{j^n(a)}{A_0} \cdot e^{-\frac{E_{aEM}}{K \cdot T_{bl}(a)}}, \tag{2.22}$$

Where N is the total number of via/contact of the block that are active at time t_a. Here, an identical current density flows through all the via/contact at each cycle.

After making relation between instantaneous current density and instantaneous power consumption, let us understand how to estimate the power consumption of digital CMOS circuits. As also shown in Figure 2.3, it is generally considered in terms of three components [48]:

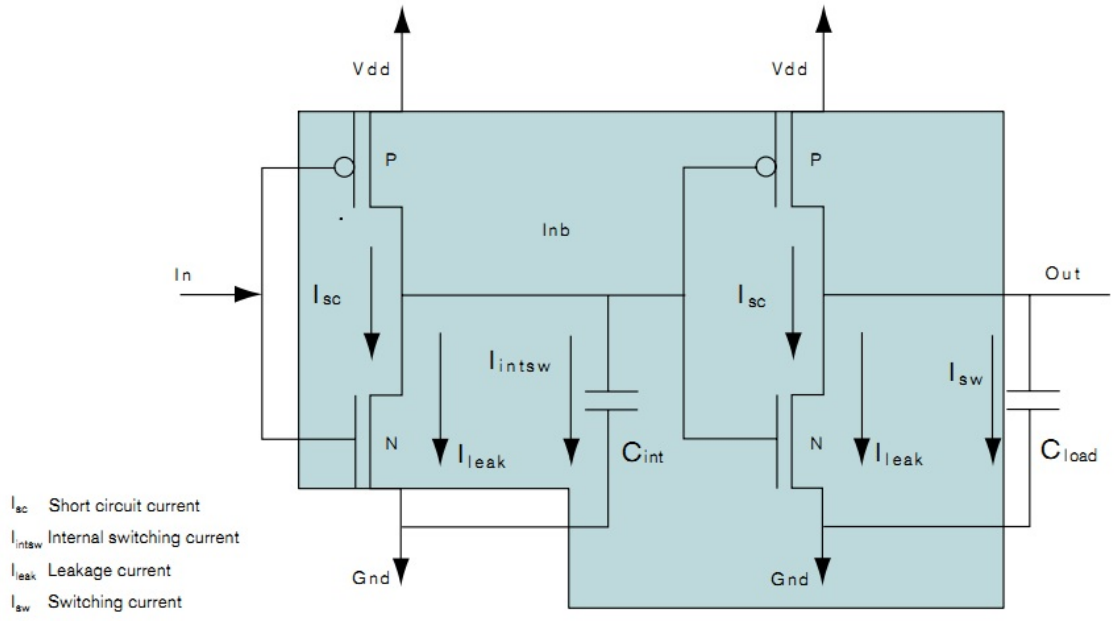

Figure 2.3: Power dissipation sources in a buffer cell [49].

1. The dynamic power component, related to the charging and discharging of the load capacitance at the gate output.

2. The short-circuit power component. During the transition of the output line (of a CMOS gate) from one voltage level to the other, there is a period of time when both the PMOS and the NMOS transistors are on, thus creating a path from VDD to ground.

3. The static power component, due to leakage, that is present even when the circuit is not switching. This, in turn, is composed of two components - gate to source leakage, which is leakage directly though the gate insulator, mostly by tunneling, and source-drain leakage attributed to both tunneling and sub-threshold conduction.

For short-circuit power consumption, careful design to minimize low slope input ramps, namely through the appropriate sizing of the transistors, can keep this component as a small fraction of the total power, and hence it is generally considered only as a second-order effect. With an estimate of the average amount of charge, Q_{short}, which is carried by the short-circuit current per output transition, the short-circuit power is obtained with Eq. 2.23:

$$P_{tr-short} = Q_{short} \cdot V_{dd} \cdot f, \tag{2.23}$$

Also, the contribution of the static power component to the total power number is growing very rapidly in the current era of Deep Sub-Micrometer (DSM) Design can be expressed as in Eq. 2.24

$$P_{tr-Leak} = V_{dd} \cdot I_{tr-Leak}, \tag{2.24}$$

where $I_{tr-Leak}$ is sum of leakage currents gate to source leakage and source to drain leakage currents.

Dynamic power consumption for a particular instant is given by Eq. 2.25:

$$P_{tr-dyn} = C_{ox} \cdot V^2{}_{dd} \cdot f, \tag{2.25}$$

where C_{ox} is the equivalent gate capacitance of the transistor and capacitance of a block $C_{bl} = N \cdot Cox$.

If multiple power sources are used, the power will be the sum of the power drawn from the different power sources. Similar to failure rate (λ), Assuming the number of transistors N that a circuit contains, the instantaneous power consumption P_{bl} of a digital block can be expressed as in Eq. 2.26

$$P_{bl} = N \cdot (P_{tr-dyn} + P_{tr-short} + P_{tr-Leak}), \tag{2.26}$$

Hence, from Eq. 2.21 and Eq. 2.26 the current density is derived of a given block as in the following Eq. 2.27,

$$j_{bl}(a) = \frac{N \cdot P_{tr-dyn}(a)}{V_{DD} \cdot S}, \tag{2.27}$$

And from above stated equations, the failure rate of the block is derived as an expression of the dynamic power of the block and the input switching probability as expressed in Eq. 2.28,

$$\lambda_{bl-EM}(a) = \frac{P^n_{bl-dyn}(a)}{B_0} \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}), here, \frac{1}{B_0} = (\frac{1}{V_{DD} \cdot S})^n \cdot \frac{1}{A_0} \tag{2.28}$$

Hence, the CFR for EM of a digital block b at time n can be derived as in Eq. 2.29:

$$CFR_{bl-EM}(n) = \sum_{i=1}^{n} (\frac{P_{bl-dyn}(a)^2}{B_0} \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a), \tag{2.29}$$

EM results depend on power consumption and temperature variations. Similarly to power and temperature, CFR is computed at each time instant. Note here that the static power contribution is not yet considered. Parameter t_a is a constant value (let say l) related to the frequency at which power and temperature are recorded. The a^{th} line in a power or temperature values corresponds to the value measured at time step $(a-1) * l$. Power and temperature are assumed to remain constant during time l.

2.4.3 Hot Carrier Injection

As explained in Subsection 2.2.2, Hot carrier injection describes the phenomena by which carriers gain sufficient energy to be injected into the gate oxide. Interface-state generation and charge trapping induced by this mechanism result in transistor parameter degradation, typically as switching frequency degradation rather than a "hard" functional failure. HCI occurs during the low-to-high transition of the gate of an NMOSFET, therefore the degradation increases for high switching activity and/or higher frequency of operation [50]. Furthermore, the recovery in HCI is negligible, making it worse for AC stress conditions. The HCI failure rate at transistor level is given using Takeda model:

$$\lambda_{tr-HCI}(a) = \frac{exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{-E_a}{kT})}{A^n_{HCI}}, \tag{2.30}$$

Dependence of $\lambda_{tr-HCI}(a)$ on $I_{sub}(a)$ is considered to be constant in present derivation, since leakage power is assumed to be constant throughout. $\lambda_{tr-HCI}(a)$ depends on the switching probability α of MOS transistors and the duration of a transistor in the saturation region. Since hot carrier degradation occurs when transistor is in the saturation region, which in CMOS circuits happens only during transitions, it follows that higher the signal activity at the gate output, the more damage a transistor accumulates. The duration of a transistor in the saturation region depends on input slew rate and the load capacitance of a gate. In turn, the load capacitance depends on the fan-out of the gate [51]. The temperature acceleration is often treated as a minor effect in most HCI models; however, to consider possible large temperature excursions, some reliability models include a temperature acceleration effect based on the HCI lifetime model given in [3]. Hence we can rewrite Eq. 2.30 as in Eq. 2.31:

$$\lambda_{tr-HCI}(a) = \frac{\alpha \cdot exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{E_a}{K \cdot T_{tr}(a)})}{A^n_{HCI}}, \tag{2.31}$$

In the case of Electromigration, the relation has been derived between current density and dynamic power consumption. It is easy to observe relation between switching probability and dynamic power consumption using Eq. 2.25 as α is a function of transition probability and operating frequency, and can be written as, $\alpha = \beta \cdot P_{dyn_{tr}}(a)$. As a result Eq. 2.31 can be rewritten as in Eq. 2.32

$$\lambda_{tr-HCI}(a) = \frac{\beta \cdot P_{dyn\_tr}(a) \cdot exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{E_a}{K \cdot T_{tr}(a)})}{A^n_{HCI}}, \tag{2.32}$$

The HCI failure model for a block using Eq. 2.26 can be given as in Eq. 2.33:

$$\lambda_{bl-HCI}(a) = N \cdot \lambda_{tr-HCI}(a), \tag{2.33}$$

Combining Eq. 2.26, 2.19, 2.32 and 2.33, CFR for HCI at block level is derived as in Eq. 2.34:

$$CFR_{HCI}(n,bl) = \sum_{i=1}^{n}(\frac{\beta \cdot P_{dyn-bl}(a) \cdot exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{E_a}{K \cdot T_{bl}(a)})}{A_{HCI}^{n}} \cdot t_a), \tag{2.34}$$

2.4.4 Time Dependent Dielectric Breakdown

Time-Dependent Dielectric Breakdown is an important failure mechanism in ULSI devices. As discussed in Subsection 2.2.3, the dielectric fails when a conductive path forms in the dielectric, shorting the anode and cathode. According to [52], assuming an Arrhenius law and an "E" model to describe the time-to-breakdown evolution with voltage and temperature parameters, the relationship is given as in Eq. 2.35:

$$\lambda_{tr-TDDB} = A_0 \cdot exp(\gamma_{TDDB} \cdot V_{DD}(a)) \cdot exp(\frac{-E_a}{K \cdot T_{tr}(a)}), \tag{2.35}$$

where E_a corresponds to the activation energy and γ_{TDDB} to the voltage acceleration factor. Since, half of the transistors experience TDDB at a given instant. The failure model for a block can be given as in Eq. 2.36:

$$\lambda_{bl-TDDB}(a) = \frac{N}{2} \cdot \lambda_{tr-TDDB}(a), \tag{2.36}$$

By combining Eq 2.19, 2.35 and 2.36, Eq. 2.37 is obtained:

$$CFR_{TDDB}(n,b) = \sum_{i=1}^{n} \frac{N}{2} \cdot exp(\gamma_{TDDB} \cdot V_{DD}(a)) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a, \tag{2.37}$$

2.4.5 Negative Bias Temperature Instability

Negative Bias Temperature Instability (NBTI) - It is a key reliability issue that is of immediate concern in p-channel MOS devices stressed with negative gate voltages. NBTI manifests as an increase in the threshold voltage and consequent decrease in drain current and transconductance. The degradation exhibits power law dependence with time. From [3], Eq.2.17 is rewritten as in Eq. 2.38:

$$\lambda_{tr-NBTI}(a) = A \cdot V_{DD-bl}^{n}(a) \cdot exp(\frac{-E_a}{K \cdot T_{tr}(a)}), \tag{2.38}$$

Assuming half of transistors switching at any given instant with p as probability that a transistor gets affected with NBTI and depend on dynamic power. Since, there is no easy way to calculate p, hence, to remain close to reality and give fast simulation results a worst case scenario is assumed, $p = 1$. The failure model for a block can be given as in Eq. 2.39:

$$\lambda_{bl-NBTI}(a) = A \cdot \frac{N}{2} \cdot \lambda_{tr-NBTI}(a), \tag{2.39}$$

By combining Eq 2.19, 2.38 and 2.39, Eq. 2.40 is obtained:

$$CFR_{NBTI}(n,b) = \sum_{i=1}^{n} \frac{N}{2} \cdot A \cdot V_{DD}^{n}(a) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a, \tag{2.40}$$

Table 2.2 gives the summary of all the derived models; these models will be used as the base in our reliability prediction tool.

Table 2.2: Failure models at functional level of abstraction derived using transistor level models. For better understanding parameters involved are defined in the third column from left. Refer to Table 2.1 for the definition of rest of the parameters.

Failure Mechanism	Failure Model (CFR)	Parameters Involved
Electromigration	$\sum_{i=1}^{n}(\frac{P_{bl-dyn}(a)^2}{B_0} \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a)$	P_{bl-dyn}: Dynamic power consumed by a block; a: Simulation time step of duration t_a; T_{bl}: Average temperature of a block; B_0: Constant depending on technology and initial operating conditions
Hot carrier injection	$\sum_{i=1}^{n}(\frac{\beta \cdot P_{dyn-bl}(a) \cdot exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{E_a}{K \cdot T_{bl}(a)})}{A_{HCI}^{n}} \cdot t_a)$	β: Proportionality constant
Time dependent dielectric breakdown	$\sum_{i=1}^{n} \frac{N}{2} \cdot exp(\gamma_{TDDB} \cdot V_{DD-bl}(a)) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a$	N: Total number of transistors in a block
Negative bias temperature instability	$\sum_{i=1}^{n} \frac{N}{2} \cdot A \cdot V_{DD-bl}^{n}(a) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a$	

2.5 Conclusion

In Section 2.4, a method is showed to build functional level reliability models using transistor level models. The accuracy of these models can be calculated by a qualitative point of view, recalling the assumptions in subsection 2.4.1. Besides, [53] showed that all transistors age in a same manner, hence we can extrapolate the transistor level models to higher level of abstraction. Of course, this extrapolation is not accurate but is close to reality, as the stress on each transistor can be different during useful life but in all cases, it gets stressed due to dynamic or static consumption. To justify the reason why this modeling is useful to research and reliability prediction methodologies, in Chapter 3, different methodologies are

presented that use different modeling ways. We will see in Section 3.2 that [54, 55, 56] have provided ways of modeling failure models at higher level of abstraction, but they have their own limitations. We will use the derived functional level failure models in our reliability prediction tool presented in Chapter 4, and also validate these models in Chapter 5 by case studies using TSMC 40nm technology library.

To conclude this section, we may say that the accurate statistics of following parameters can provide a good reliability prediction of useful life of the device: Probability of number of transistors switching at a given instant, Current Density, Substrate Current, Temperature, Area or number of transistors. In following Chapters, we will use the models derived in the current chapter and embed them in a reliability simulator, with help of power and temperature simulators.

Chapter 3

Previous work - Reliability simulation methodologies

Contents

From previous chapter, it is clear that there is a strong relationship between power consumption (includes current density and stress), temperature, environmental conditions (humidity, and ambient temperature), process variations, operating conditions (operating supply voltage and frequency) and failures (permanent faults) in an integrated chip (IC). Brooks et al. [57] explained this relationship as shown in figure 3.1 and discussed that most system-level

(one block or a complete processor) reliability models are empirical models which can benefit greatly from calibration and validation, i.e., to set a standard at lower level of abstraction for reliability for a given specification. Both dynamic and leakage power increase temperature. In general, temperature profiles depend on the temporal and spatial distribution of power, the size and plan of the chip as well as the microprocessor's cooling and packaging solutions. High temperature increases charge-carrier concentrations, resulting in increased sub-threshold leakage power consumption. In addition, it decreases charge-carrier mobility, decreasing transistor and interconnect performance, and decreases threshold voltage. Finally, temperature has a tremendous impact on the errors that lead to the majority of IC permanent faults. Process variation greatly influences the other power-related integrated circuit (IC) characteristics shown in figure 3.1. It is a cause of changes in critical timing paths that affects transient fault rate; and changes in various parameters such as wire and oxide dimensions to increase permanent fault rate, also, leakage power consumption by causing changes in dopant concentration; and finally affecting dynamic power consumption. As Figure 3.1 show the relationships among power, temperature, and reliability are complex. Controlling power, temperature, and reliability requires modeling and optimization at different abstraction levels. It will be clear from the following discussion that there is a clear lack of EDA tools and methodologies for reliability prediction at higher level of abstraction. Even the existing methodologies have some avoidable and some unavoidable limitations. It is important to take a look on the tools that emerged from the various works. The aim is to study their environment and define their characteristics. In the end, a conclusion has been made to show the pros and cons of these tools and the motivation to why we need a new methodology for an MPSoC environment.

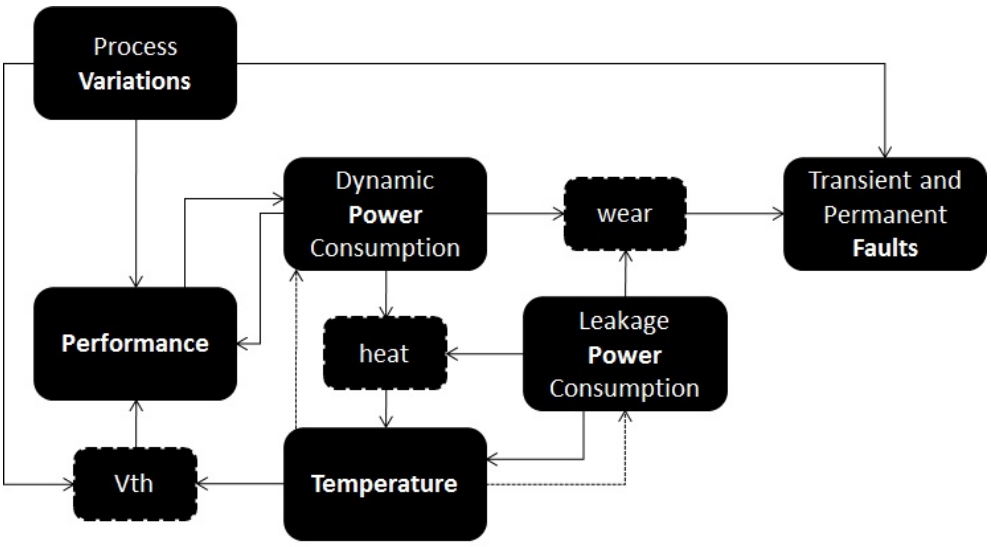

Figure 3.1: Relation between Reliability and various parameters involved

3.1 Outline of the current chapter

The chapter is divided into three main sections according to abstraction levels. This synthesis is the result from already existing literature for both industrial and academic reliability

simulators at various levels of abstraction. This work allowed us to make a comparison of all the features of the simulators and their limitations. The main objective is to highlight the methods put in place to integrate these tools into the loop of the flow of Design-for-Reliability (DFR), designed to meet the needs of designers. The following sections take a look at the various features offered by each tool in terms of electrical simulation of aging at device level. The features of each tool, their input requirements and their limitations are presented.

3.2 Transistor level reliability simulation methodology

For the background, Figure 3.2 describes a simulation environment aging at device level [58]. The simulator requires input data that are the electrical parameters of the transistor, or the degradation models (Degradation modeling is an effective reliability analysis tool for products with failures caused by degradation in various parameters), the integrated circuit architecture and specifications and can be obtained via Technology libraries. Management of all these data allows the simulator to evaluate the reliability of the integrated circuit by expressing output of electrical characteristics in the form of curves or variations obtained in tabular form. The interpretation of these simulation points and a transmission feedback can act on the input data and re-evaluate the parameter values from electrical models of transistors.

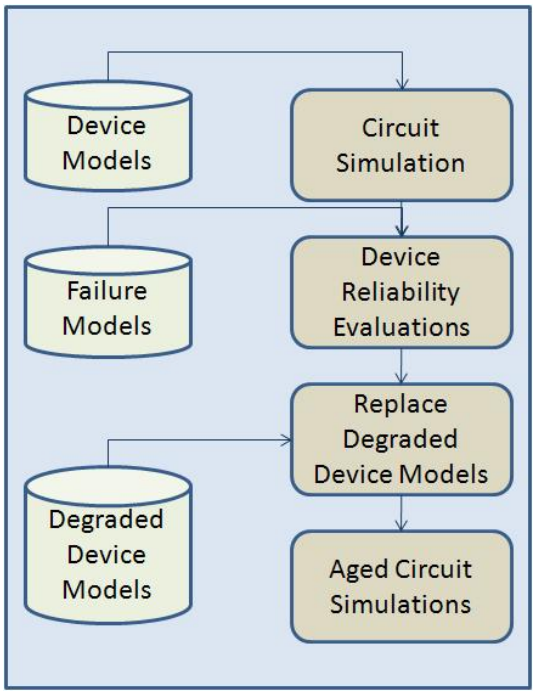

Figure 3.2: A general transistor level reliability modeling and simulation flow

Let us discuss the general method of degradation calculation at lower level of abstraction. The transient analysis algorithms perform one or more estimates of the degradation of transistors during the simulation. According to Figure 3.2, degradation can be studied and defined in two phases. After a first simulation with constraints, the simulator should perform:

1. an estimate of the degradation
2. and calculation of the drift of electrical parameters of transistors due to aging

The estimate of the degradation is calculated from the constraints of static current in a transistor. The estimate of the degradation is a function of the stresses applied to the individual device. To calculate this quantity of stress, the pattern of life of the transistor is used. Degradation suffered by the transistor is represented by the variable AGE or STRESS according to the characteristics of the tools [59], can be given as in Eq. 3.1:

$$Age(T) = \int g(V_{DS}(t), V_{GS}(t), I_{DS}(t), I_B(t)) \cdot dt \tag{3.1}$$

The calculation of the variable AGE can be given as an integration of electrical stress applied to elementary device over a given age.

The second phase calculates the drift of electrical parameters of transistors. It is calculated from the amount of accumulated damage on the aging period T. Drift parameters expressed by Eq. 3.2,

$$\Delta P(T) = f(AGE(T)), \tag{3.2}$$

The function f can represent various laws of variation: the power law, the logarithmic law and the exponential law.

After calculating the drift of electrical parameters, the simulator updates all electrical parameters of each transistor. The last step is to make an electrical simulation taking into account the degraded transistors. Degradation performance of the circuit is finally seen by comparing the electrical characteristics of the circuit not degraded and of the circuit that is aged.

3.2.1 BERT - Berkeley Reliability Tools

BERT is the tool developed by the University of Berkeley, California [60, 61] and serves as the base for the various simulator's methodologies described below. This is the reason that BERT is discussed in more details than the other simulators. BERT is developed to simulate EM, HCI and TDDB.

BERT is a software widely used, for the simulation of reliability of integrated circuits (MOSFETs and bipolar). It is used for several years among large groups of semiconductor developers in its commercial form: RelXpert [62, 63]. This software package provides the development of reliability processing for industrial applications. RelXpert can be used to combine with the SPICE-like simulators to simulate the wear-out degradation on the CMOS integrated circuits.

This tool can work either stand-alone or added to Cadence's Analog Artist environment. It simulates the age of the devices based upon actual circuit operating waveforms using degraded or "aged" models.

It is developed by the Celestry (now part of Cadence Design Systems) and is an industry standard simulation tool for simulating the effect of HC and NBTI degradation on circuit performances [64, 63]. Degraded circuit performance waveforms are calculated and used to optimize the circuits with the SPICE netlist. When combined with the Analog Artist environment, RelXpert "pick up" the key devices, which were responsible for circuit degradation [64]. The designer can also specify the devices inside the integrated circuit, which can be redesigned to improve the CMOS IC reliability. The tool generates the netlist to allow an aged circuit simulation. RelXpert can also be used to drive the commercial SPICE-like simulator for the lifetime and degradation simulations. A good point of RelXpert is that it is designed to estimate the amount of degradation experienced by each transistor in real environment, i.e., in a functional circuit and to accomplish this function, the simulated dynamic stress levels must be linked to degradation levels of transistors already determined, which can be obtained by experimental measurements in static conditions. The tool RelXpert is divided into several modules, each implementing a simulated physical mechanism of degradation, all based on a mechanism for pre/post treatment of SPICE simulation.

The module that simulates the effects of hot carrier is known as CAS (Circuit Aging Simulator), it is aging simulator and it implements the model of Hu [65]. The module CORS (Circuit Oxide Reliability Simulator) is intended to study the degradation mechanism, TDDB (Time-Dependent Dielectric Breakdown). The simulator also includes a module for the analysis of Electromigration. These models can be used together or separately during the simulation aging. It is possible to extend the simulation of the reliability to the mechanism of catastrophic degradation ESD (Electro-Static Discharge) or to the degradation phenomenon of radiation-induced SEU (Single Event Upset). A block diagram of the BERT simulator is presented in Figure 3.3. The inputs to the simulator BERT comprise of a range of information on SPICE models used, on the aging parameters of transistors, which provides a basis for data from experiments on several components and for different conditions over constraints regarding aging. These inputs must also contain the status of various degradation mechanisms which are useful for the BERT output information. The preprocessor's role is to treat input information and to put it in the standard form suitable for SPICE. After electrical simulation, the post processor calculates the time to reach failure TTF (Time to Failure). Then it calculates the table of aging, i.e., the status of various degraded transistor parameters and communicates this information to pre-processor as a feedback. With these results of aging calculation and the generation of a new simulation, it is possible to simulate the effect of a degradation mechanism in the time. Note that the simulator BERT uses the electrical simulator SPICE containing standard models of transistor components. All the information needed for the study of a failure mechanism must be provided as an input to pre-processor.

The BERT-CAS module calculates the parametric variations of the MOSFET in V_{th} threshold voltage, mobility μ_0 or trans-conductance g_m and determines the time to reach failure, TTF. The module of aging by hot carriers CAS uses Hu model based on the relationship between age and the circuit substrate current generated by stress. The main effects of hot carriers are derivatives of the parametric threshold voltage and conductance. The destructive effects are revealed by the maximum substrate current, and this for a gate bias of the neighbor transistor is close to half of the drain-source voltage.

CORS (Circuit Oxide Reliability Simulator) is a part of BERT. It projects the probability of oxide breakdown induced circuit failure as a function of operating time, temperature, power supply voltage and input waveforms. CORS can also simulate the effects of burn-in on subsequent yield and lifetime. Test capacitor breakdown statistics and input decks which

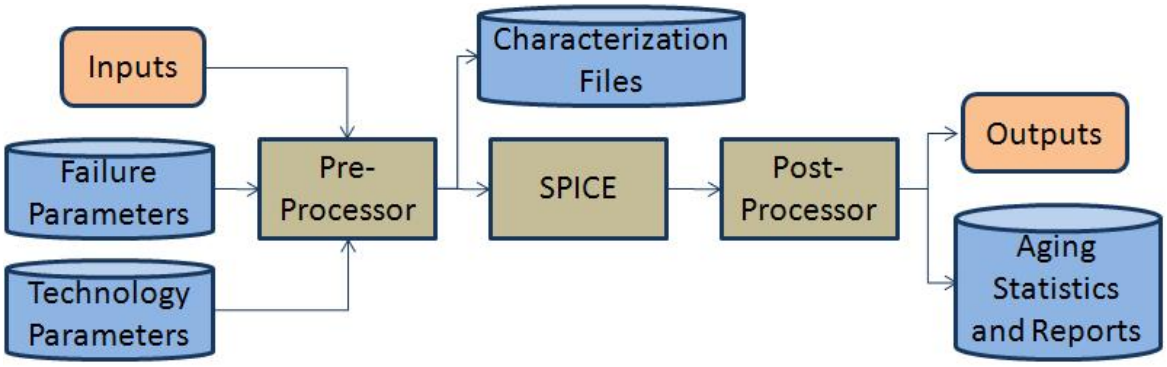

Figure 3.3: BERT Simulator: Block diagram

describe the circuit using SPICE commands have to be provided by the user to the simulator. In addition, input deck also contains commands which instruct BERT to execute the CORS module. It may also contain CAS (Hot electron module) and EM (Electromigration module) specific commands. CORS consists of a pre- and post-processor for SPICE. BERT-CORS module directly determines the time to reach failure. Figure 3.4 shows a block diagram of CORS flow.

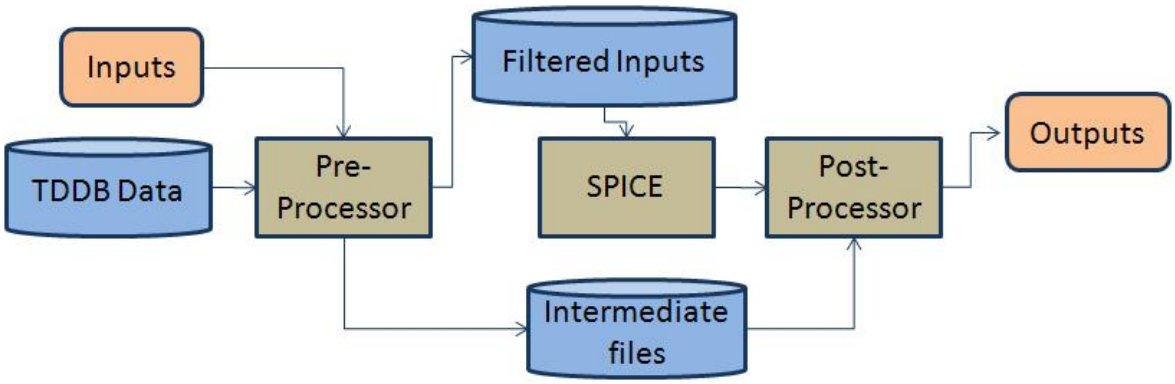

Figure 3.4: CORS: Flow chart

The BERT-EM module has two modes of operation. It can inform us by prevention on the layout and it highlights the weak points. This module is the only one to offer an opportunity to optimize the design of a circuit by pointing to the defects of the layout phase. Other modules do not have this opportunity and are limited only to the calculation of the TTF.

To facilitate the simulation by using a classical electric behavior of transistor aging is represented by a set of models fitted to the electrical characteristics of components aged experimentally. It is desirable to simulate the behavior of using the full circuit model parameters extracted from electrical components degraded in order to predict circuit performance after aging.

To conclude the discussion of BERT, BERT reliability simulator has the following features:

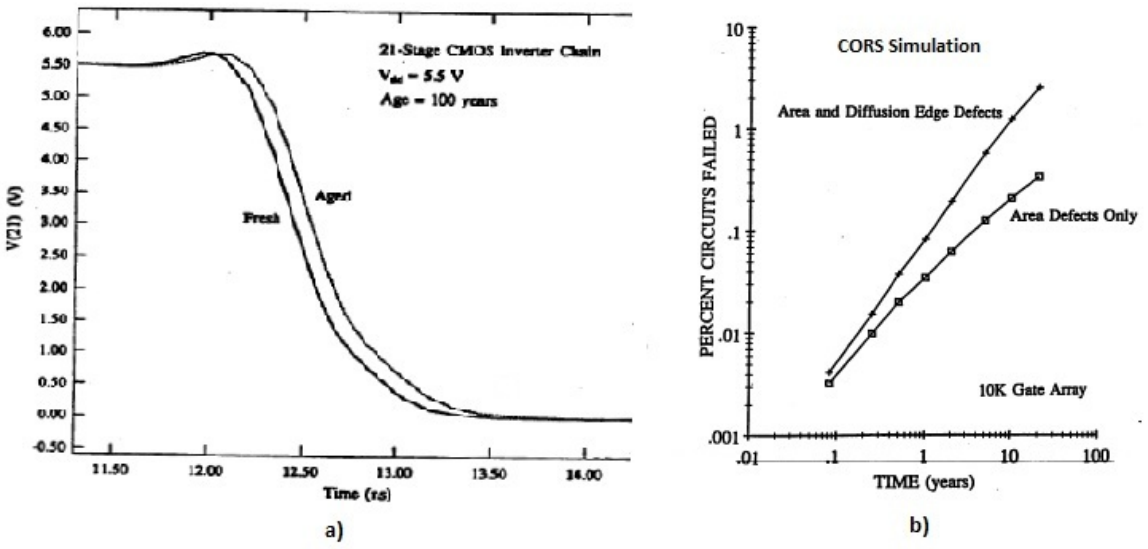

Figure 3.5: (a) The voltage waveform at the output of the 20th stage showing the propagation difference between the fresh and aged inverter chain. (b) Effects of device defects.

1. modeling the degradation of the gate oxide of the MOS transistor according to operating conditions;
2. modeling the electrical behavior of the MOS transistor according to damage localized oxide;
3. simulating the long-term degradation of the oxide;
4. determining the overall performance of the circuit after hot carrier stress.

Although it offers great accuracy, the only limitation of BERT in our case working at higher level of abstraction is its speed which is too slow to simulate a complete processor.

3.2.2 HOTRON

The reliability simulator HOTRON, a commercial tool was designed by Texas Instruments and uses a similar approach to the module BERT-CAS, and it simulates HC. The heart of the simulator is the Electrical SPICE simulator.

The function block of the simulator HOTRON is given by Figure 3.6. An important additional feature is the ability to generate sensitivity analysis [66]. The semiconductor components are strongly constrained and do not have necessarily a significant effect on circuit performance, which is why it is very important for both types of evaluations. First, it is necessary to determine about the transistors that undergo the most severe constraints, then we must also determine which of these transistors are the most susceptible to degradation, and then determine the sensitivity characteristics of the circuit to variations in performance transistor, i.e., determine the transistors that have the highest probability of modifying the electrical characteristics of the circuit, one of the results are shown in Figure 3.7.

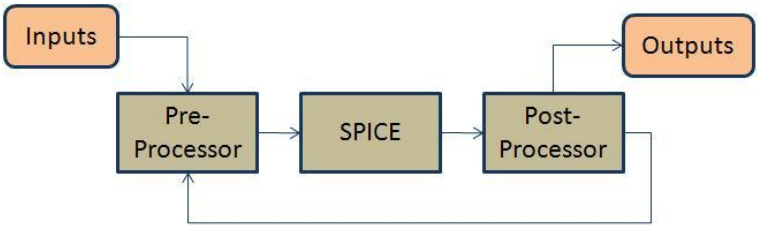

Figure 3.6: HOTRON Simulator: Block diagram

The post processor HOTRON simulator offers the possibility to make an analysis of sensitivity for a given MOSFET under strong constraints, and analyzes the effects of circuit performance. In addition, this simulator is the first of its kind to take into account the degradation of several parameters of the transistor instead of one. Parameters are the initial threshold voltage V_{th0} (zero back bias threshold voltage), the trans-conductance gm, the modulation factor of the length channel, θ_3 the speed factor of saturation of the field and the parameter of substrate effect.

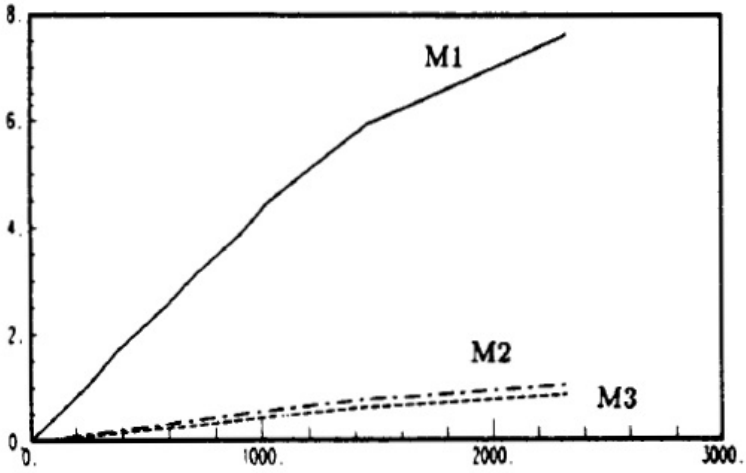

Figure 3.7: Sensitivity of the MOSFET's in the pre-charge circuit, the x-axis is the DC stress time (hours) and y-axis is the pre-charge time increase (%) M_4 is practically zero

To conclude the discussion, Hotron works at transistor level like BERT, and hence has similar limitation to BERT. It considers variation in V_{th} and g_m, which was important for being the base of simulators that were developed after Hotron.

3.2.3 UltraSim

Hot carrier (HC) reliability simulation can also be simulated in Virtuoso UltraSim. Virtuoso UltraSim is the Cadence FastSPICE circuit simulator capable of predicting and validating timing, power and reliability of mixed-signal, complex digital and System-on-Chip (SoC) designs in advanced technology of 0.13 μm and below.

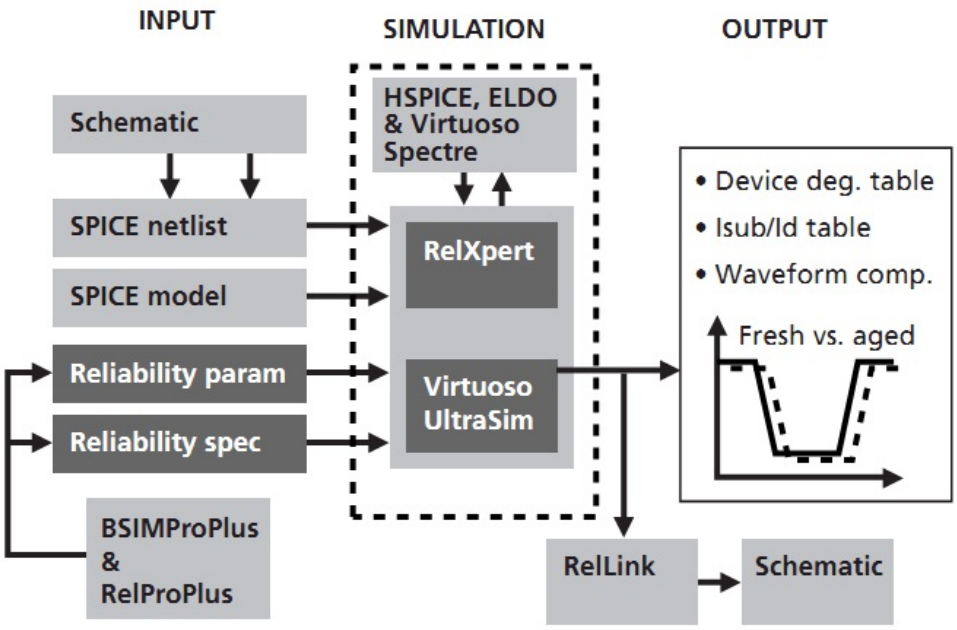

Figure 3.8: UltraSim Simulator: Block diagram [45]

It has a set of specialized reliability models (AgeMos) for HCD and NBTI simulation [67, 63]. In the simulation, an *Age* parameter is calculated for each NMOS device with the Eq. 3.3:

$$Age(\tau) = \int_{t=0}^{t=\tau} [\frac{I_{sub}}{I_{ds}}]^m \times \frac{I_{ds}}{W \times H} dt, \tag{3.3}$$

where W refers to the width of the transistor; m and H are technology dependent parameters and determined from experiments; I_{sub} is the substrate current; I_{ds} is the drain current; t is the time for stress. For PMOS devices, the gate current I_{gate} is used instead of I_{sub} to determine the *Age* parameter. The simulation starts with device parameter extraction and modeling. From the SPICE model parameters of fresh devices, some other device parameters are added to accurately model I_{sub}. Saturation current I_{dsat} , threshold voltage, V_{th}, or the maximum trans-conductance, g_{max}, can be used as a degradation monitoring parameter. According to Li *et al.* [45] I_{dsat} is a good degradation monitor for digital circuits, while V_{th} is suitable for analog applications. The stress time resulting in 10 percent decrease of one of these degradation monitoring parameters is arbitrarily set as the device lifetime. Finally, it performs AgeMos extraction. Using the *Age* parameter calculated after the simulation, the AgeMos module applies the degradation models, which can be provided as input to most SPICE-like simulators, for the aged circuit simulation. Reliability simulation with Virtuoso UltraSim is an iterative process; several iterations are needed to get accurate modeling. The simulations output the degradation results to predict the lifetime of each MOS instance. Complete flow of UltraSim is depicted in Figure 3.8.

UltraSim has following features:

1. speeds up full-chip simulation 10 times to 1,000 times, with minimum of 1% accuracy;

2. virtually provides unlimited simulation capacity and can handle large memory circuits;
3. the only complete transistor-level silicon accurate solution, it includes timing, power, noise, and reliability analysis.

UltraSim has some very interesting features, but being an industrial tool, not available for researchers as open source to develop further. Although UltraSim can simulate complete chips and much faster than BERT, since it performs transistor-level simulations, it is not a solution to simulate MPSoC.

3.2.4 PRESS

PRESS reliability simulator has been developed in the Netherlands in cooperation between Philips and MESA Research Institute, University of Twente [68]. PRESS is a simulator for HC.

The simulator PRESS is developed based on the electrical simulator PSTAR, Philips is the owner of this reliability simulator deals primarily with the failure mechanism of hot carriers in the MOS technology. It can be extended to other mechanisms of wear. Figure 3.9 shows the block diagram of the simulator PRESS. Block Pre-Press provides filtering of specific commands to the circuit from the input and adapts them to those subject to the reliability of the technology, and then the result of this treatment is used directly by the simulator PSTAR. For this simulator, it should be noted that PRESS uses the script for the treatment of reliability unlike other simulators, with electrical simulator PSTAR as an integral part. This has the effect that it is not necessary to add, to create or to modify the original code of the simulator. PRESS evaluates the parametric variations of internal models, i.e., during the transient simulation, instead of using a loop of information actively. It is understandable that the simulator PRESS uses a coupled solution, unlike other simulator that uses a decoupled approach to calculate the effects of wear mechanism. In this simulator, the degradations are renewed at every step of the transient analysis; the duration of the analysis is not adjustable by the user of the simulator. In the case of analysis with a single simulation step, this tool can sometimes appear slower than other functionalities. But the case of a multiple-step analysis, PRESS is much faster than iterative simulations generated as a result of a feedback loop where you have to solve the quasi-static equations many times while changes between two parametric transient analyses are very low or nonexistent. In addition, the simulator PRESS uses the automatic management of the duration of the transient analysis step in PSTAR electrical simulator. As for the other simulators, the duration of the transition analysis is chosen by the user and the management of the analysis is not conducted by an additional program.

For simple calculations like the duration of life, which requires little information and time simulation, a special option is used. Post-press output generates the lifetime of transistors and some additional information on the degradation, if requested. In fact, the blocks Pre-Press and Post-Press are programs that do not conduct reliability analysis. PRESS, the simulator can be used in two modes, the indicator mode and complete simulator mode. The indicator mode calculates lifetimes compared to a failure criterion given and highlights the transistors potentially low. No backup of parameters is performed during the transient analysis which aims to reduce the CPU time. The full mode, however, performs backup of degradation

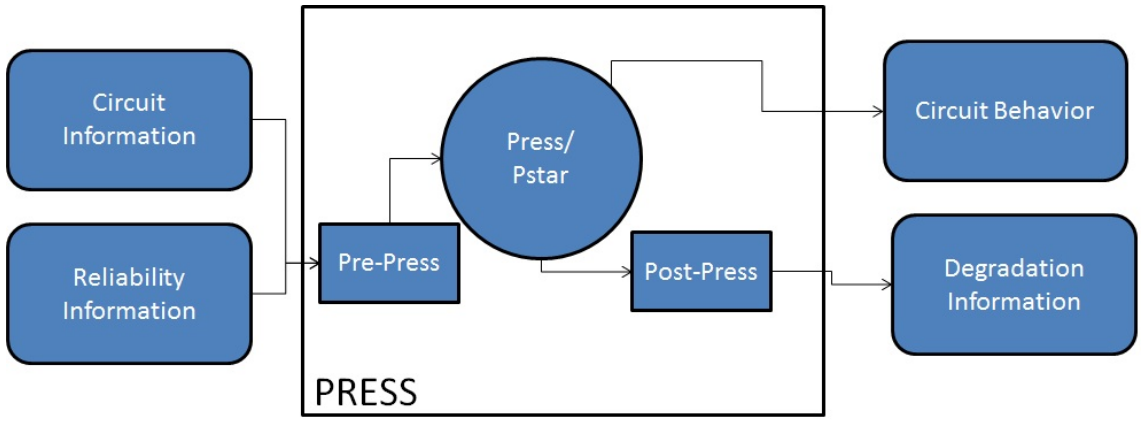

Figure 3.9: Block diagram of simulator PRESS

parameters and calculate all the effects of degradation on electrical behavior of the MOSFET circuit. This mode of operation is similar to features offered by HOTRON. But it should be noted that the effects of wear mechanism on all the MOSFETs, then on all the electrical characteristics of the circuit can be analyzed simultaneously so that the HOTRON simulator provides a sensitivity analysis of electrical performance of the circuit using only one transistor degradation.

The output file has no information regarding the optimization of circuit's parameters. However, regarding the mechanism of hot carriers, only the voltages and the channel length L are taken into account. In addition, with the full simulation mode and the use of a specific behavioral modeling language PSTAR, it makes it easy to get information about the sensitivity of circuits with respect to the degradation of MOSFETs due to injection of hot carriers.

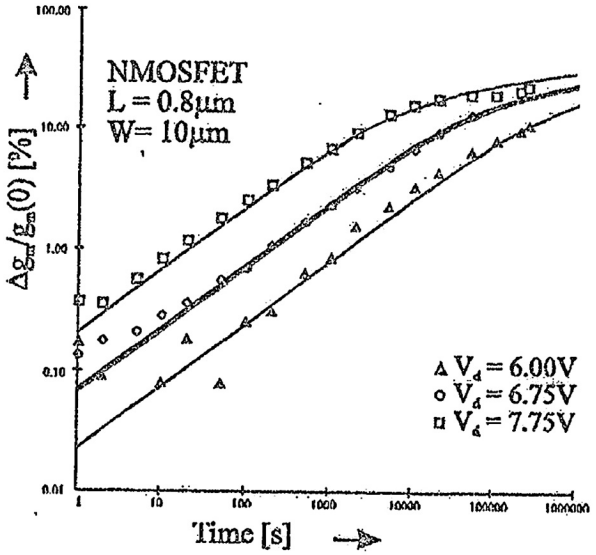

Figure 3.10: For different drain stress voltages (W=10 μm, L=0.8 μm and for 10% g_m shift) to predict lifetime, measurements of an NMOS for verification of parameter shift in PRESS is shown.

To conclude the discussion of PRESS, it provides script based solution hence no need to adapt original simulator. There is no way to manually manage the transient analysis step. Hence, there is no way to make the simulations faster for specific needs of user. Also, Press has a limitation that it provides results only for fixed input and not for variable inputs. Hence, no complex simulations can be performed.

3.2.5 IMS methodology

Ghosh et al. have developed a methodology in [69], it use standard simulators with no external processing. They use modified models for aging that use behavioral and compact modeling. In contrary to Press (discussed in previous subsection), they have shown the simulation results using Verilog-A with variable signals and hence they provided reliability results for complex simulations. Verilog-A code is used to simulate the impact of device failure mechanisms on the circuit in operating conditions. This methodology is newer than other reliability aware transistor level methodologies and very interesting for emerging technologies.

3.2.6 Synthesis of State-of-the-Art tools at transistor level of abstraction

The useful life of CMOS circuit depends on the degree of degradation caused by the wear-out and should be confirmed before the marketing of the product. Degradation is mainly caused by the transistor's characteristic shift over time with high electric field near drain edge generating high energy hot-carriers that are out of thermal equilibrium. This phenomenon was well managed by previous generation manufacturing technologies. According to [70] acceleration in miniaturization of the device, makes it very difficult to assure HCI reliability as scaling of the device dimension passes scaling of the voltage. Some macro-modeling is available with slightly less accuracy than the transistor-level approach. For example, iRULE [71] uses macro-models to approximate degradation of the logic gate, it transforms transistors in series in multi-input logic into an equivalent transistor for simplification. It can handle larger circuits than transistor-level ones, but it was still difficult to study the behavior of the complex logic correctly.

3.3 Gate level reliability simulation methodology

The functionality available at gate level from the works of [71] and before were limited to a small part of a chip. It should be clear that the understanding of full-chip behavior is required to help the designers decide whether to fix the circuit or not. Hence, a full-chip aged timing simulation tool is required.

3.3.1 GLACIER

GLACIER is a gate level circuit characterization and simulation system for hot carrier effects and developed by BTA Technology for VLSI design [72]. It is an accurate tool that can handle large circuitry. It can be integrated in design flow as shown in Figure 3.11 for the purpose of circuit Hot-Carrier (HC) reliability simulation. The methodology is mainly based on two

major sub-systems, HCLib for cell characterization and HCSim for simulation and analysis. The authors of Glacier have introduced a new ratio based gate level HC degradation model. It considers T_{fresh} and T_{aged} as the fresh and aged pin-to-pin signal delays as shown in Figure 3.12. α is the aged-to-fresh signal delay ratio which characterizes the overall degradation of all transistors in the gate due to hot carrier effect. These variables are defined for each transition type (rise or fall) of each signal path of the logic gates. Equation 3.4 shows the relationship between T_{fresh}, T_{aged} and α.

$$T_{aged} = \alpha \times T_{fresh}, \tag{3.4}$$

It also includes a powerful graphical user interface (GUI) provides friendly user interface to facilitate cell characterization, timing library verification and timing analysis and to enhance productivity.

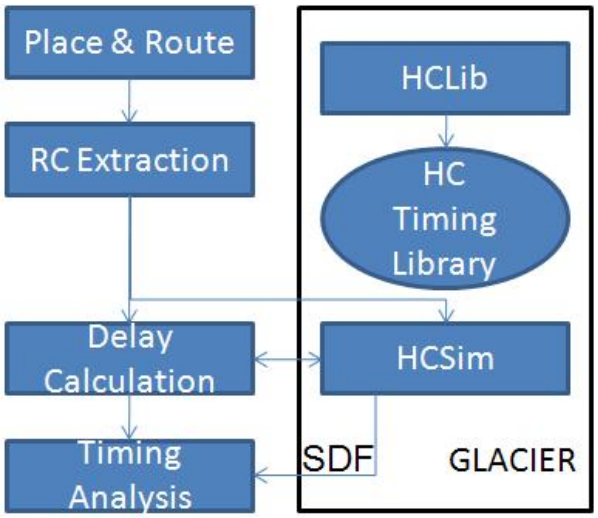

Figure 3.11: Glacier Simulator: Block diagram

To conclude, by virtue of a unique ratio based modeling technique the accuracy in Glacier is kept within 1 % error range as compared to transistor level circuit reliability simulation. Depending on the circuit size, Glacier can be 10 to 1000 times faster than BERT.

3.3.2 ILLIADS - Illinois Analogous Digital Simulator

The analytical macro-modeling for the MOS logic gates first proposed by University of Illinois for the event-driven fast timing simulation of MOS digital circuits [73, 74]. This model was developed by mapping the MOS logic gates onto a generalized MOS circuit. The output waveform for this circuit with linear input waveforms for each component can then be processed by solving a nonlinear (Ricatti) differential equation with time-varying coefficients. In [73, 74], the equation describing the MOS transistor I-V characteristics is assumed to be piecewise-quadratic, and hence, can only be applied to technology modeled by the simplified

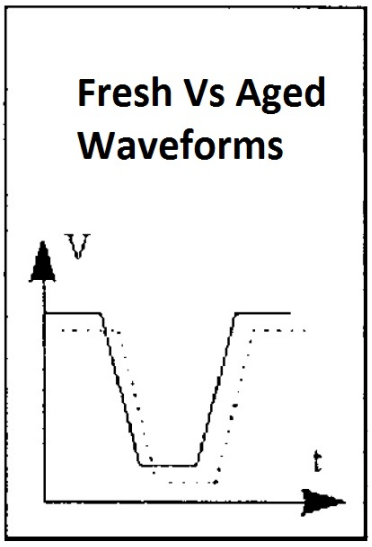

Figure 3.12: Glacier: Fresh vs Aged waveforms

level 1 SPICE model. ILLIADS displayed similar results as obtained by SPICE but with a lower time for simulation. The RWQ (Region-Wise Quadratic) models are used in ILLIADS (ILLinois Analogous Digital Simulator), a fast timing simulator for hot-carrier reliability simulations. According to Shih et al, these models provide the same level of accuracy as SPICE but with a lesser time for simulation as shown in Figure 3.13.

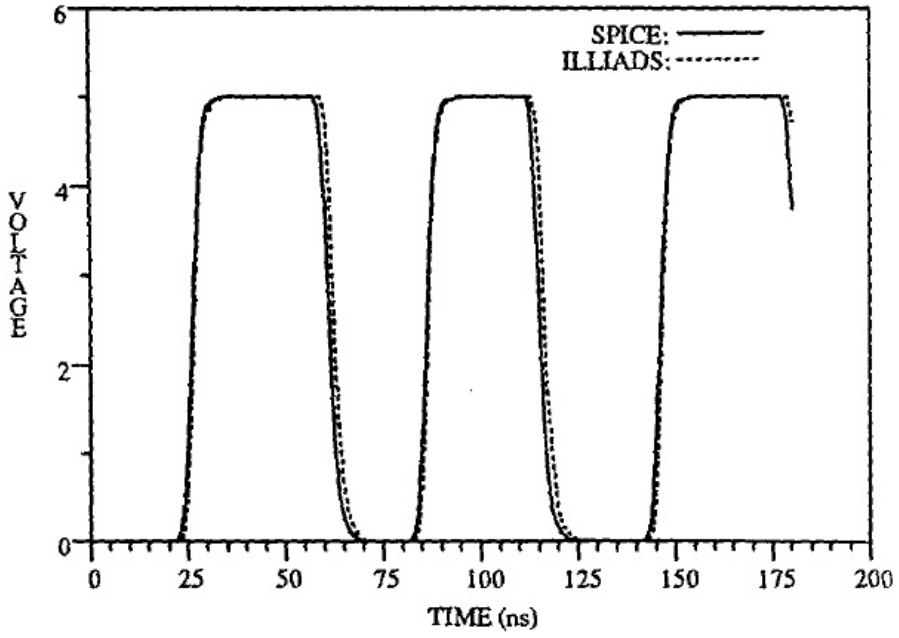

Figure 3.13: An output of a 4-bit full adder, the difference between outputs from SPICE and ILLIADS is small

The latest version of ILLIADS, named ILLIADS-T, is an electro-thermal simulation methodology for temperature profile estimation, hotspot identification and circuit reliability prediction for CMOS chips. ILLIADS-T is applied to Electromigration (EM) reliability diagnosis and timing analysis. It considers both transistor and interconnects temperatures. According to authors, the temperature sensitive failure mechanisms induced MTTF and critical path timing are estimated by ILLIADS-T, and is also used to simulate the temperature profile of a commercial microprocessor with the most advanced packaging structure. In Figure 3.14, the statistical measurement block (Fast Timing Simulation) is needed in order to collect and update the statistical data. Then, the stopping criterion is used to determine whether the average power estimated thus far is close enough (converged) to the real value. This simulation is then terminated and it reports the average power value. Else, another input pattern will be generated for the next round of circuit simulation. To be more efficient, it is necessary that the stopping criterion will result in the sample size as small as possible.

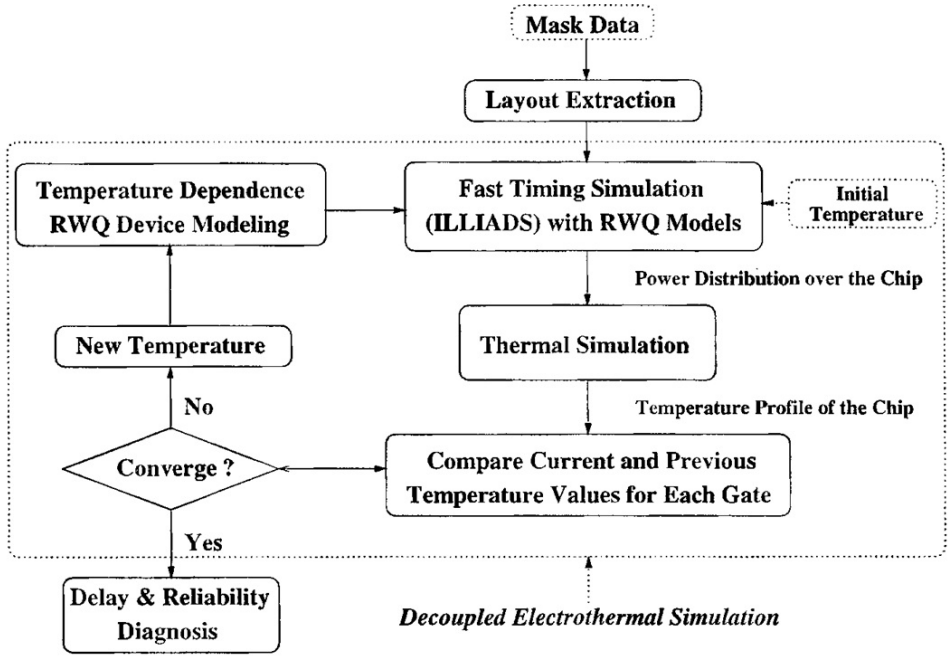

Figure 3.14: Flowchart of ILLIADS-T simulator

To conclude, ILLIADS is not developed specific to any failure mechanism. Authors claimed that it is much faster than transistor level reliability simulation tools. It validates results for a circuit with 235,000 transistors that simulate 10 minutes of real clock for one clock cycle output. But it is not enough for large simulations on a processor due to speed and memory constraints.

3.3.3 Synthesis of State-of-the-Art tools at gate level of abstraction

Semiconductor design and manufacturing is suffering through changes which is causing the (supposed) unlimited lifetime that customers expected to be reduced significantly. Hence, ITRS predict the onset of significant reliability problems in the future via roadmaps. Due

to new materials, processes and devices, the voltage scaling limitations and increasing power consumption are causing many new reliability challenges. The gate level reliability simulators provide accurate enough information with higher speed of simulations. The discussion gives researchers base to go to higher level than gate level. The following section discusses methodologies that exist at architectural level, which provide less accurate but fast enough results to simulate a complete processor.

3.4 Architecture level reliability simulation methodology

According to [10], there are the two main challenges to maintain reliability, which are, the continued increase in die size and number of transistors, and the constant scaling of transistors for performance. More the transistors result in more failures which results in lower lifetimes. Increasing power consumption and increasing transistor density are causing higher temperatures on chip again resulting in fast failure. Also, before the work of Srinivasan et al [10], techniques for enhancing reliability focused on fault-tolerant computing methods like redundancy and efficient failure recovery methods. This and other techniques are mainly focused on error detection, recovery, minimizing down time, and increasing yield. But they had no interest in the rate of wear-out of processors. Before [10], there were already many techniques that can maximize energy and thermal performance by exploiting architectural features and adaptation capabilities. Using these approaches it is possible to predict long-term reliability to track application behavior and can be used to increase processor reliability.

3.4.1 RAMP

In beginning of 21st century, the Reliability Aware Micro-Processor (RAMP) model for processor reliability was published by IBM. RAMP provides chip mean time to failure (MTTF) as a function of the failure rates (λ) of individual blocks of the chip due to different failure mechanisms i.e., Electromigration, stress migration, time-dependent dielectric breakdown, temperature cycling, and can be used to evaluate the reliability due to different applications, architecture and processor designs" [47].

It is claimed that the RAMP model is a self-standing module that could be attached to simulators to make power and temperature predictions [10]. RAMP only models intrinsic processor failures because long-term processor reliability is dominated by wear-out or intrinsic failures. However, IBM published that RAMP can be extended to model soft errors. It should be noted that RAMP uses the Arrhenius model to show the dependence of processor failures on temperature. The brief overview of the methodology (for a case study) is given in Figure 3.15 [56]. RAMP has been integrated with two different simulators first is IBM's 'Turandot' architectural simulator which provides inbuilt power estimation capabilities and uses 'HotSpot' for temperature simulation, and the second is 'RSIM' simulator which uses 'Wattch' for power measurements and again 'HotSpot' for temperature measurements. These simulators also have possibilities to model leakage power. As already said, according to Srinivasan et al. RAMP is a separate module, and easy to port to other simulators. In Figure 3.16, Srinivasan used the results obtained using RAMP to show the effect of scaling on reliability. The lifetime of a processor is reducing rapidly with technology scaling. It seems, there is no further development of RAMP.

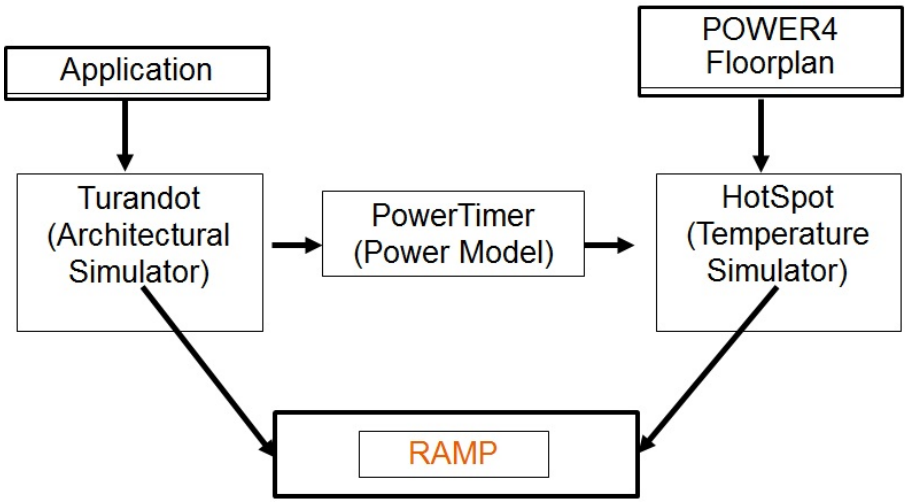

Figure 3.15: Block diagram of simulator RAMP

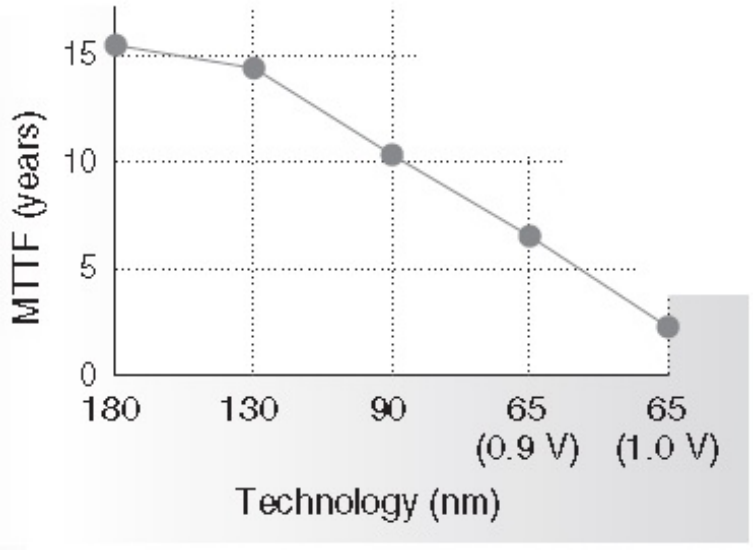

Figure 3.16: Using RAMP results, Srinivasan et al. has shown effect of scaling on reliability

To conclude, RAMP is very interesting beginning for reliability simulation at higher level of abstraction. The already discussed assumptions out of which some are accurate and some inaccurate, make it difficult to use RAMP for different specific needs.

3.4.2 A functional level reliability simulation methodology by Coskun et al.

Coskun et al [7], from University of California, San Diego, have provided methodologies that use DVS (Dynamic Voltage Scaling) and DPM (Dynamic Power Management) to improve the reliability of MPSoCs shown in Figure 3.17.

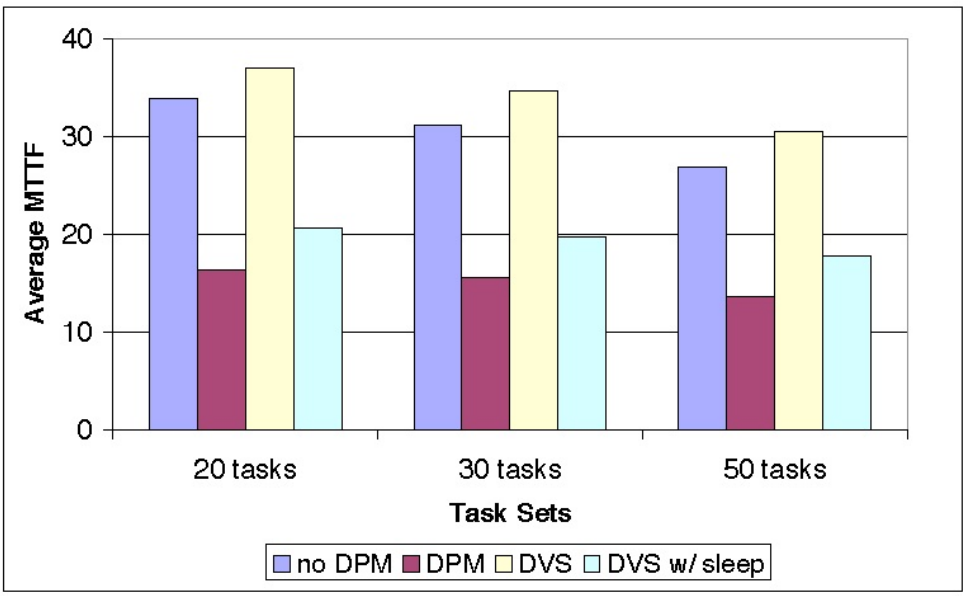

Figure 3.17: Coskun compared various power management policies

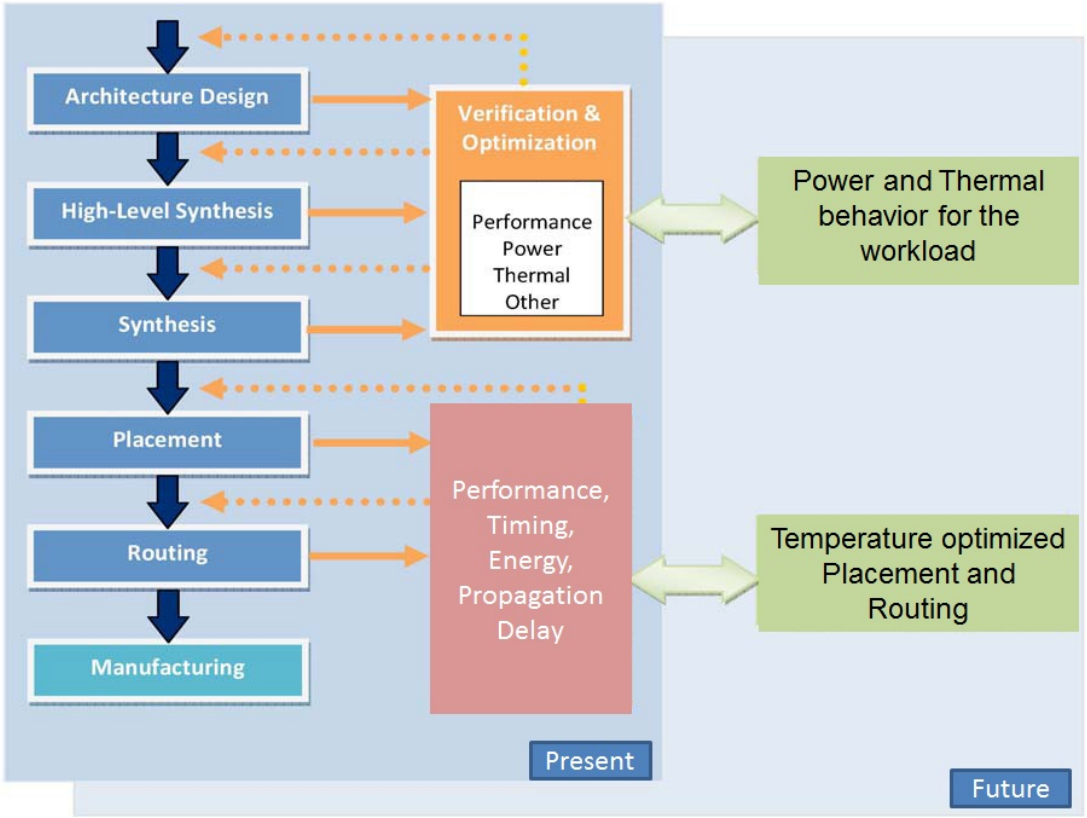

Figure 3.18: Coskun's present and future flow diagram of IC design

They have shown theoretically that using DVS can highly improve the lifetime of the multi-processor. But they assume the failure rate of the circuit to be dependent on its instantaneous behavior and to be independent of circuit usage in the past. The methodology is said to provide simulation speed much faster than at architecture level even when simulations run at fine grain level. To simplify, Coskun assumed an exponential lifetime distribution for different failure mechanisms with the failure rate of the processor supposed to be dependent on its instantaneous parameters (e.g., temperature and voltage), and considered as independent of the past values. It obviously seems inaccurate as a wear-out mechanism has increasing failure rate as the circuit age increases even if the parameters remains same or even when there is now application running on processor actively. In a very recent article [55], Coskun et al have shown current and future IC design flows in Figure 3.18. Only average and peak power and temperature evaluation at higher level designs are included in the current flow and power estimation during the lower level of abstraction. For multi-core architecture to be more reliable, it has been proposed to extend the design flows with two additional features; one is, to evaluate the transient power and temperature during system architecture design and synthesis and the second is to consider temperature constraints and optimize placement and routing in the design of microarchitecture units at chip level.

To conclude the methodology from Coskun is not free of limitations like other discussed methodologies. It assume exponential lifetime distribution, and did not implement the methodology in design flow hence results (with arbitrary values) are not validated.

3.4.3 Agesim

In [54], Huang provided many details about mathematics of calculating aging rate, a new reliability metric but only numerical examples for Electromigration. The integration of AgeSim in a real MPSoC design flow with different technology libraries is not yet addressed. In contrast to existing work, AgeSim is able to simulate arbitrary lifetime for different failure mechanisms. According to authors, AgeSim does not require tracing the system's reliability related parameters during its lifetime, and supposed to be more efficient and accurate. Simplified AgeSim framework is provided in Figure 3.19.

Different simulators are used to simulate power and temperature and obtain performance; aging rate and energy are obtained as outputs. Figure 3.20 shows one main result obtained using AgeSim, which is the impact of DVFS (Dynamic Voltage Frequency Scaling) on the failure rate. Different DVFS techniques can provide different level of reduction in failure rate.

To conclude in AgeSim, failure models are provided as inputs to calculate reliability using the obtained aging rate. AgeSim does not include new failure models. The methodology is very strong and according to authors can also be extended to MPSoCs. The methodology, even at single processor level is not yet implemented in a design flow, i.e., no technology library is taken into consideration. Hence, results are provided with arbitrary values.

3.4.4 Synthesis - Requirements to predict reliability at higher level of abstraction

At a high level, the environment can be characterized in terms such as ground, mobile, airborne, space, etc. This provides a rough indication of the severity ranges of stresses to be

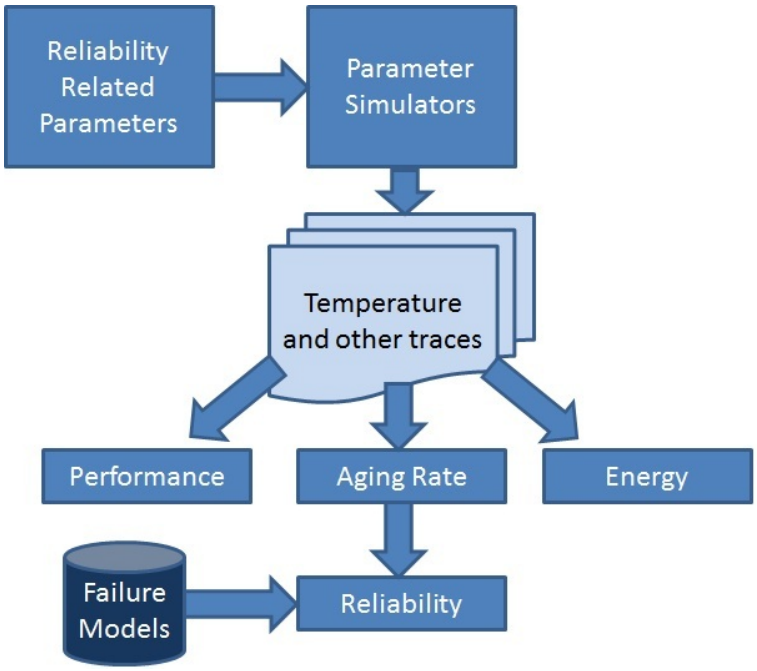

Figure 3.19: AgeSim simulation framework

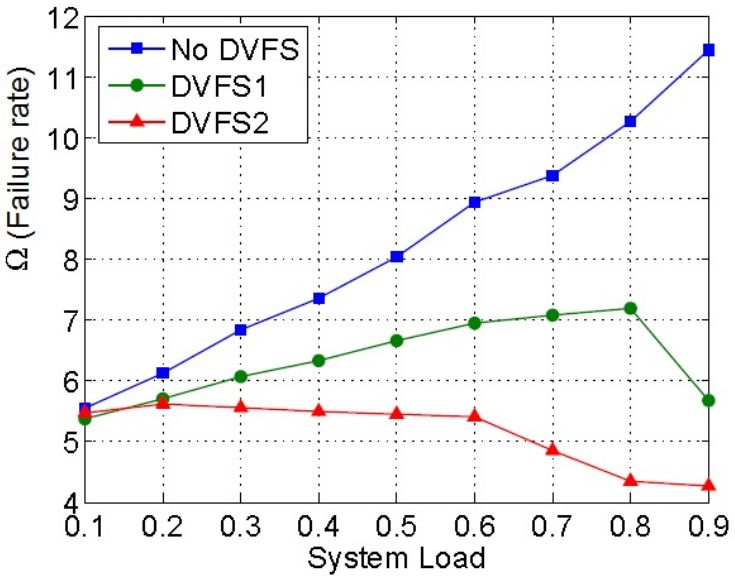

Figure 3.20: Huang et al shows the impact of DVFS on aging using arbitrary results (no technology libraries are used) from AgeSim simulations

experienced. As the design process starts, more detailed information may be necessary. For example, Time Stress Measurement Devices are available that can be used to measure and record stresses such as temperature, humidity, shock, vibration and power.

A number of reliability-oriented tasks are useful in deriving performance reliability require-

ments from the needs. Modeling and Simulation is an effective technique to determine a level of reliability, or range of reliability, necessary to meet a more general need or requirement. It enables the trade-off of various product characteristics to achieve a more general requirement. Simulation makes use of computer automation for various solutions until an optimized solution is achieved. It provides means of identifying solutions without the costly design, build, and test process. As the product evolves during the Design/Development phase, the models are updated to reflect the current product design configuration.

In developing reliability requirements using mission and support models, the relationships among reliability and the various mission and support figures of merit are defined using mathematical relationships. These figures of merits may be the number of spares required for a given scenario, the number of flights that can be generated in a given time period, the average number of products down for service at any one time, etc. By varying the operational measures of reliability, the effect on the figures of merits can be determined. Those values of operational reliability which give the "best" results, all other factors being held constant, are then selected as the reliability requirements.

Ideally, each product requirement would be optimized. Instead, the overall product must be optimized, with the product's requirements addressed properly, as a set. Whether requirements are conflicting or complementary, they must be viewed as a whole, and not as independent requirements. Reliability requirements should be developed within the context of the overall requirements for the product. The focus must be on the overall product performance. So other performance requirements, and even form and fit requirements, can complement or conflict with the reliability requirements. To reach a "best" compromise, trade-offs are conducted in which an increase in one requirement is traded for a decrease in another. In a given situation, reliability might be traded off (i.e., the requirement is reduced) to achieve better performance.

The design requirements, operational requirements, or both may need to be adjusted to account for the improvement in technology, different operating environments, different duty cycles, and so forth. Temperature is one of the most important parameter that influence reliability. Temperature effects are associated with electronics. By conducting a thermal analysis, designers can determine heat transfer paths and modes, temperature extremes experienced by individual components and parts, and the impact of thermal shock caused by rapid changes in temperature. Like temperature analysis, power analysis is also very important as it provides the knowledge of stress time during the product's life. This stress also cause the rise in temperature.

3.5 Conclusion

In this chapter, we have discussed reliability simulators and methodologies from past 25 years. Understandably most of the existing tools work at transistor and gate level of abstraction as prediction of reliability needs high accuracy. High level simulations mainly focus on complex architectures with millions of transistors. Hence, the prediction of reliability cannot be very accurate due to complexity involved. At the higher-level of design stage, power and temperature are taken into account by estimating the average and peak power and the corresponding thermal behavior. There are already some thermal simulation tools that are provided by EDA companies: e.g., FloTherm by Mentor Graphics and Encounter by Cadence for Power/Thermal Analysis. These commercial tools are based on applied fluid dynamics;

Level of Abstraction	Name of Tool	Author and University/Co.	Pros	Cons
Transistor level	BERT	Univ. of Berkeley	Very accurate, EM, HCI and TDDB	Aging parameters, Small-step iterative algorithms, Very slow, if we consider a complete chip (not feasible) (In PRESS - automatic management of transient analysis step)
	HOTRON	Texas Instruments	HC, Similar approach to BERT-CAS, considers variations in Vth and gm	
	RelXpert	Celestry	HC , NBTI, Stand-alone tool, use accurate degradation models	
	ULTRASIM	Cadence	Specialized models for HC and NBTI, Isub is the primary reliability parameter	
	PRESS	University of Twente, Philips	HC, script based solution, no changes to adapt original simulator	
Gate level	GLACIER	BTA Technology	HC, accurate to handle large circuitry, 1% error compared to BERT, GUI	Considers each transition, hence very slow in a circuit with billions of transistors
	ILLIADS	University of Illinois	Can perform, timing, temperature and reliability simulations, approximately 1000 times faster than spice oriented tools	No specific failure models, 10 minutes of real clock for one clock cycle output for 235,000 transistors (not enough for large simulations on a processor)
Functional level	RAMP	University of Illinois	Good starting point at high level of abstraction, methodology in design flow, EM, TDDB	Lots of assumptions, some accurate, some inaccurate, uniform device density over chip
	-	Coskun *et al.*, University of California, San Diego	Methodology using DTM and DTM in MPSoCs	Assume exponential lifetime distribution, Not implemented the methodology in design flow, results with arbitrary values
	AGESIM	Chinese Univ. of Hong Kong, China	Good new mathematics for calculation of aging rate	Not implemented the methodology in design flow, results with arbitrary values, i.e., no technology libraries are taken in consideration

Figure 3.21: Relation between different methodologies

hence, they can accurately model chip-level temperatures. However, a simulation framework that integrates runtime workload profiles with the power and thermal models and hence predicts reliability is not utilized in this flow. At lower-level of abstraction, with information about placement and routing of the circuits, it is possible to compute detailed timing and performance characteristics to optimize the design accordingly.

In [75] it is said that performing architecture-level simulation is very costly with today's tools for emulating only a few seconds of real-life execution takes hours to days, depending on the system's complexity and due to this all the techniques that can provide efficient techniques for evaluation of fine-grained performance, temperature, and energy simultaneously are very important for current and future. The stress on a processor changes a lot during runtime with different workloads, to make the right choice for the various design issues is a difficult task, without an accurate lifetime reliability simulator as shown in results in [54]. It is a challenging problem to design a performance efficient yet accurate lifetime reliability simulator. In authors knowledge there are no work in the literature in this domain except [54] who are also working on similar methodology in parallel as presented in Chapter Reliability estimation at functional level, with the big differences are the embedding in design flow and the parameters fitting with close to real values in our methodology.

In Figure 3.21, a quick comparison is provided between all the simulators and methodologies provided above. It is clear to see that methodologies from [54, 55, 56] are new generation methodologies to predict reliability at high level of abstraction. As shown in Figure 3.21, all the existing methodologies have some issues that should be taken care of, i.e., missing some key points and still required to be included in reliability simulation methodology. In the following chapter, a new methodology is introduced that takes many of the good points from existing methodologies and especially close to RAMP and AgeSim simulation frameworks. It will be clear in Chapter 5, that this methodology has also been implemented in design flow at functional level using case studies with 40 nm TSMC technology library and functional level MIPS 32 processor architecture.

Due to all above discussed limitations of methodologies from Srinivasan, Coskun and Huang et al., it is shown that no tool exist for reliability prediction accurate enough that can provide high speed and exploration capabilities for many core architecture at functional level of simulation, which can be embedded in design flow. To conclude, it is required that this tool must be able to handle the complexities that exist in MPSoCs, with the consideration of timings and should take into concern the technology libraries while integrated in the design flow.

Chapter 4

Reliability estimation at higher level of abstraction

Contents

The emergence of new embedded applications for telecom, automotive, digital television and multimedia applications has fueled the demand for architectures with higher performances, more chip area and high power efficiency. These applications are usually computation-intensive, which prevents them from being executed by general-purpose processors. Thus, designers are showing interest in a System-on-Chip (SoC) paradigm composed of multiple processors and a network that is highly efficient in terms of latency and bandwidth. The resulting new trend in architectural design is the Multiprocessor SoC (MPSoC) [76]. MPSoC architectures can have homogeneous or heterogeneous processors, depending on the application requirements. Choosing the best processor among hundreds of available architectures, or even designing a new processor, requires the evaluation of many different features (pipeline structure, ISA (Instruction Set Architecture) description, register files, processor size etc.), and the architect needs to explore different solutions in order to find the best trade-off. Semiconductor industry has an immense pressure for improving performance, increasing functionality, decreasing cost and reducing design and development time. For all these improvements, the solution is to minimize device feature size in nanometer scale range and further, which affects the lifetime of a chip drastically.

This chapter is organized to provide clear details of the tool-chain developed in order to predict reliability of a processor. This chapter is divided into 5 main subsections. Section 4.1

provides motivation to our methodology and the tool chain developed at functional level of abstraction. Section 4.2 provides a state of the art for existing power simulation tools and also a new power simulator which is developed as a part of this thesis in order to predict reliability. Section 4.3 discusses about an academic open source temperature simulator which is accurate enough and famous among researchers. Section 4.4 is the heart of the complete methodology that will be presented in this chapter, it presents a reliability simulator that can take inputs from power and temperature simulators at any abstraction level (Power-ArchC and HotSpot in this case) and other inputs which are provided by designer and manufacturer to predict reliability of a design. Final Section 4.5 shows how to integrate all power, temperature and reliability simulators into a tool-chain named RAAPS (Reliability Aware ArchC based Processor Simulator).

In Chapter 5, using RAAPS, it will be shown that, aging depends on applications executing during lifetime of the processor, and some blocks are much more prone to failures than others.

4.1 Reliability simulation at Functional level

At different levels of abstraction, there are different trade-offs to calculate power consumption, temperature and predict reliability. More accurate data can be obtained at lower level of abstraction than higher. But the simulation is faster at higher level of abstraction. Multi-Processor System-On-Chips (MPSoCs) are composed of hundreds of processor cores, memories and interconnect. Their design space exploration (memory sizes, processor pipeline depth, interconnect bandwidth, task scheduling, etc.) for performance or power consumption objectives require fast and accurate simulators. Performance, power and temperature modeling and simulations for MPSoCs are still subject to intensive research works, such as SESAM [77].

At this point, it should be clear in the mind of the reader that the main aim of this thesis is to give means to predict reliability at functional level of abstraction to help designers. Designer needs to verify if a design is robust and can handle memory sizes, task scheduling etc. for performance and reliability. Hence, this thesis provides a tool that will provide designer an enhanced capability to analyze reliability and take quick decisions to improve the lifetime of his design.

In Figure 4.1 which is based on all the discussed requirements in Section 3.4.4 and as shown in Figure 3.1, an introduction to a new methodology is provided. This introduction should help the reader to make links between following sections and subsections that explains the various blocks of this tool-chain. In Figure 4.1, it is shown that power consumption, i.e., both dynamic and leakage power increase temperature. In general, temperature profiles depend on the temporal and spatial distribution of power, the size and plan of the chip as well as the microprocessor's cooling and packaging solutions. High temperature increases charge-carrier concentrations, resulting in increased sub-threshold leakage power consumption. In addition, it decreases charge-carrier mobility, decreasing transistor and interconnect performance, and decreases threshold voltage. Finally, temperature has a tremendous impact on the errors that lead to the majority of IC permanent faults. The relationships among power, temperature, and reliability are complex. Controlling power, temperature, and reliability requires modeling and optimization at different abstraction levels to achieve different level of accuracy. A reliability simulator needs different information. There is information about static inputs that do not change while the design is undergoing stress during its lifetime such as, floorplan,

spreader, heat sink, packaging, technology library, failure models. There are also some dynamic inputs that may or may not change during the time, design is under stress such as, operating voltage, operating frequency, consumed power and maximum temperature of the design (or just some part of design). There are various failure mechanisms that may result in temporary, intermittent and permanent errors in integrated circuits as discussed in Chapter 1. The main ones are Electromigration (EM) in interconnects, TDDB in the gate oxides, hot carrier injection (HCI) in NMOS transistors, and negative bias temperature instability (NBTI) in PMOS transistors. These failure mechanisms have been extensively studied at the transistor level in the past, and researchers have provided values for the technology dependent parameters for each failure mechanism, brief discussion was provided in Chapter Chapter 1 - State of the art - Failure mechanisms in a chip. The important permanent failure mechanisms are modeled and a method to build functional level reliability models using transistor level failure models has been shown in Chapter 2.

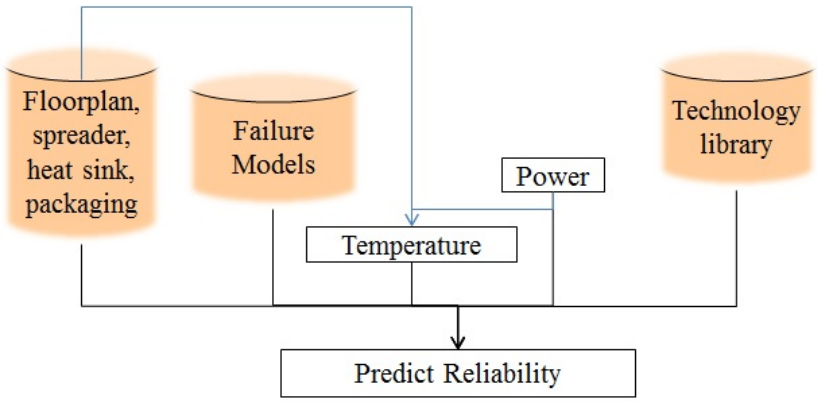

Figure 4.1: A general methodology to predict reliability - Reliability can be predicted by designer using knowledge about power consumed, temperature on the chip, mathematics of failure mechanisms and information obtained via technology library

Relatively to other works related to reliability simulation at front-end as discussed in Chapter 3, the reasons a reliability aware processor simulator is developed are:

1. Need of speed during simulation: the processor lifetime reliability was simulated at cycle accurate level (pipeline step) which was too slow for MPSoC simulation.
2. Need of a powerful language description for processor cores at functional level with lifetime reliability evaluation capabilities to be readily integrated in an MPSoC simulator.
3. Need to distinguish the effect of different benchmarks on lifetime reliability of the processor and explore the effect of different task scheduling techniques in an MPSoC, very early in the design flow.

Starting points of this work to develop this reliability aware processor simulator are (i) RTME (Real Time MTTF Evaluation, a new tool developed) [78], a tool to get failure models and a reliability simulator at block level, (ii) Power-ArchC [79], an enhanced ArchC based Instruction Set Simulator (ISS) that embeds block-level power estimation capabilities at functional level (A new methodology is given to perform functional-level power estimation, will

be discussed in details in following sections), and (iii) Hotspot [80, 81], a block-level temperature simulator (an academic tool, source-code available). The technical contribution of this methodology to the scientific world is a trace-based tool chain (power-temperature-reliability) - that can be fully parameterized - for exploring reliability in a processor, at functional level. In this reliability aware processor simulator, any technology library from manufacturers, packaging library and failure library (failure models as derived in Chapter 2) can be plugged. Power values at functional level are obtained by an extensive Instruction-Level Power Characterization for a given technology library [79]. Relatively to a packaging library, temperature values before synthesis are estimated by solving electrical equations in an equivalent RC network that represents the lateral and vertical paths of temperature dissipation over the integrated circuit [80]. Reliability is expressed as the cumulative failure rate (CFR) over time for each failure mechanism as explained in Section 2.4. This reliability aware processor simulator could highlight the main failure detractor and the weak part of the design that is prone to this detractor. In the following sections, it is shown, how to build this tool chain named RAAPS (Reliability Aware ArchC based Processor Simulator) at functional level for exploring the effects of different benchmarks. The proposed methodology is applied to the MIPS processor for a case study and provides results in Chapter 5.

As discussed in Chapter 3, for reliability at functional level, the first and foremost requirements are to simulate power consumption and temperature behavior. In Section 4.2, Power-ArchC is detailed that is a powerful ISS enabling the power analysis of processor cores at functional abstraction level. In the Subsection 4.2.1, already existing methodologies are presented regarding power simulators.

4.2 Power-ArchC

As a corollary to Moore's law, power consumption of computing nodes doubles every 18 months [82] and needs to be evaluated as soon as possible in the design stages. Due to this, there are many tools at present to assess power at different levels of abstraction. Most commercial tools start working at the design after synthesis. Such tools are not candidates for MPSoC power analysis. The different power analysis methodologies and tools will be discussed in this Subsection. Lot of research has been performed in power estimation techniques and tools at transistor level, gate level and register-transfer level [83, 84]. Also in recent days, more research is performed at cycle-accurate and behavioral levels [85]. Since they target different levels of details, they make different trade-offs between simulation time and accuracy.

Compared to previous works, it is the first work that enables the generation of a power aware ISS ready to be integrated in a complex SystemC based System-on-Chip (SoC) design with a short development time. The technical contributions can be given as follows:

1. a semi-automatic design flow extended with power exploration capabilities at High-Level;

2. a Power aware ISS generated by an Architecture description language;

The following subsections will provide motivation that make designing of this tool more obvious and present some already existing power estimation methodologies at different levels of abstraction. Then a new methodology is presented to estimate power at functional level in

Subsection 4.2.2. The proposed methodology is applied to the MIPS case study and results are provided in Chapter 5.

4.2.1 Power Modeling and Simulations

Some recent academic tools like 'Wattch' [85], 'SoftExplorer' [86, 87] enable the power analysis at front end design. Most simulators are parameterized, so they can be used to estimate the energy consumption of different system configurations. In the rest of this section, some already existing methodologies and tools (commercial and academic) are discussed.

At Transistor level, power estimation is typically performed as a by-product of circuit simulation [83]. It enables the most accurate estimation but in detriment of a very slow speed because the simulation is performed late in the design cycle. These simulators characterize models of transistors and estimate voltage and current behaviors over time. Power dissipation of transistors comes from three sources: switching power, short-circuit power, and leakage power as already discussed in Section 2.4.2. Such simulation is time-consuming but useful in integrated circuit design. Transistor-level simulators such as 'HSPICE', 'HSIM' [88, 89, 90], are not suitable in evaluating power consumption of large programs on complex systems, as they need large amount of memory space for storing all the details.

Gate-level approaches simulate a gate-level design, and calculate power by considering the switching activity (average number of toggling per time unit), the time, the equivalent capacitance and voltage of internal nodes [91, 92]. Compared to the previous approach, power values are less accurate but simulation speed increases. Techniques for power estimation and switching units of a circuit can be mapped to predefined minimized activities. At gate and circuit levels, it is usually classified as (1) statistical, or (2) probabilistic. In statistical methods, circuit simulation is performed using a set of randomly chosen input vectors, while monitoring the switching activity on each circuit node. The simulation runs until the switching activity converges to the average switching activity. Convergence is tested by a statistical mean estimation technique, such as the Monte Carlo procedure [92, 84]. In probabilistic methods, user-supplied input probabilities are propagated into the circuit to produce signal probabilities at every node. The switching activity may be determined once the signal probabilities at each node are computed. An example is 'PrimeTime' [49], which delivers a dynamic and leakage power analysis for design geometries at 90-nm and below. Designers have a single, unified analysis environment for timing, signal integrity and power analysis that is anchored by the 'PrimeTime' static timing solution.

At Cycle-accurate level, power model considers the equivalent capacitance of a macro logic block. Compared to the previous approach, accuracy is lower because the internal switching activity of the block is not considered, only I/O toggling activity. Conversely, simulation speed is faster. At Cycle-accurate-level simulator simulates the execution at the level of individual cycles, allowing keeping track of power behavior changes across cycles. Examples of cycle-level simulator are 'Wattch' [85], 'SimplePower' [93] and 'Sim-Panalyzer' [94]. 'Wattch' involves analytical capacitance models which have to be developed for each block of a processor by the microarchitecture which is sometimes very difficult to perform. Indeed, at the cycle-accurate level, the processor's behavior is simulated cycle by cycle. In these tools, it is not a problem when only a small portion of the code (a few instructions) is simulated, but this may be very time consuming for large programs. Moreover, cycle-level simulations necessitate a low-level description of the architecture.

Instruction-level simulators provide coarser power behavior than the cycle-accurate ones. It is possible to analyze very quickly the power consumption of the circuit but with a rather low accuracy than descriptions made at lower abstraction level. The simulation is based on the instruction-level energy profiling of the instruction set of the target processors and the assumption that the energy consumption of an instruction is mostly independent of the addressing mode or operands or previous instructions. Some methodologies about Instruction level power analysis are discussed in recent papers [95, 96]. One of the instruction-level simulators is 'Joule-Track' [97]. Joule-Track is available as an online resource and has various estimation levels. It acts as a predictor for a set of benchmark programs evaluated on the StrongARM SA-1100 and Hitachi SH-4 microprocessors.

At Application-level, to estimate whole-program power-consumption, the methodologies generally work as predictor of power consumption for a program [98]. An application-level simulator, 'SoftExplorer' [87] works at C-level and relies on the Functional-Level Power Analysis, which results in a power model of the processor.

'SoftExplorer' is an interesting tool that works at C-level and relies on the Functional-Level Power Analysis, which results in a power model of the processor. F. Klein et al. propose 'PowerSC' [99, 100], a power-aware extension of 'SystemC' classes. It allows power estimation of circuits described in 'SystemC' language. 'PowerSC' is based on a new multi-model power estimation engine. It selects the macro-modeling technique leading to the least estimation error for a given system component depending on the properties of its input-vector stream. Design effort for power insertion in SystemC design is relatively limited: add of macros in user code and, technology and power libraries that are automatically built by 'PowerSC' flow.

A power aware ISS is proposed in the subsection 4.2.2, called Power-ArchC because the tools at transistor, gate and cycle-accurate levels are not a practical solution to perform power and hence reliability explorations for a complete processor, whereas Joule-Track is not suitable for SystemC based processor designs written in ArchC language. In addition estimation of power consumed of dynamic applications is required that will be executed on a processor; this is not possible with 'SoftExplorer'. Although PowerSC is a very powerful framework that would fit to our needs, its availability is currently limited to gate-level description. Power-ArchC is based on a power-reuse model approach. Power values of instructions are obtained at gate level for better accuracy while simulation time for characterization remains acceptable. The instruction set architecture is annotated with obtained power values in the simulator. It should be noted that the goal is to have an accurate power simulator and not to derive new methodology which is enough for reliability analysis.

4.2.2 Power-ArchC using ArchC and ILPC

Power-ArchC is based on a methodology for functional and instruction level power analysis given by [101] and can be seen in Figure 4.2. Qu et al, have provided methodology for a given microprocessor and build a 'power data bank' that has power information about basic instructions and then they evaluate total energy consumption and execution time using data bank obtained. The components connected by the dotted arrows show the procedure to build the 'power data bank'. Instructions executed only once for one hardware configuration without any information about the benchmarks to be executed on the core. The power estimation tool is shown and connected via solid edges. It takes the user's program and input data,

and estimates the power of the execution of such benchmarks on the given processor. Power estimates are given for 3 % error.

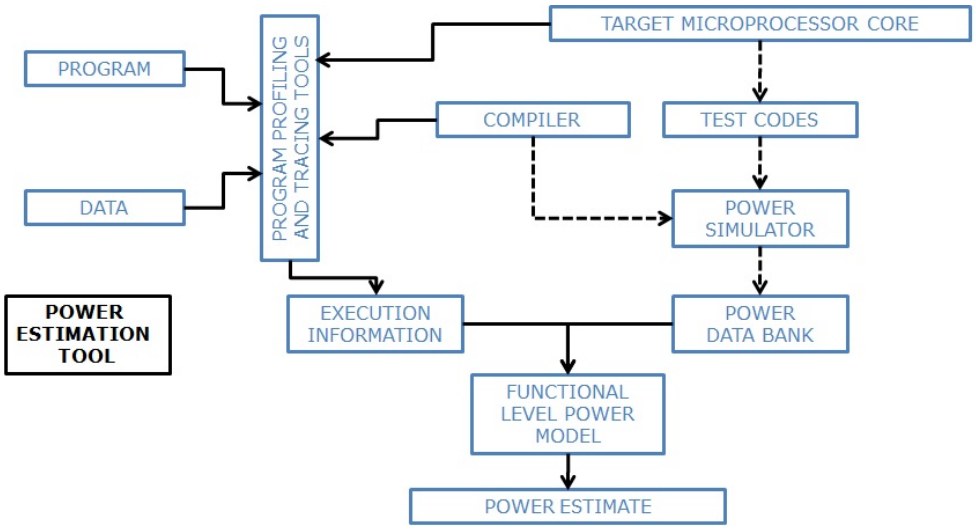

Figure 4.2: Functional level Power estimation methodology

ArchC based ISS

An Instruction Set Simulator (ISS) is a simulation model, usually coded in a high-level programming language, which mimics the behavior of a mainframe or microprocessor by "reading" instructions and maintaining internal variables which represent the processor's registers [102]. The ISSs enable fast design space exploration and simulation of complex MpSoCs while accuracy remains in acceptable level.

The processor Instruction Set Simulator (ISS) has an important role, and must have the following features: it should be able to parameterize, fast and accurate, and able to be easily integrated in the MPSoC simulation environment. The ISS emulates the behavior of a processor by executing the instructions of the target processor while running on a host computer. Depending on the abstraction level, it can be modeled at the functional or cycle-accurate level. On the one hand, the functional ISS model abstracts the internal hardware architecture of the processor (pipeline structure, register files etc.) and simulates only the ISA. Therefore, it can be available in the early phase of the MPSoC design for the application software development, where the simulation speed and the model development time are important factors for a fast design space exploration. On the other hand, the cycle-accurate ISS model simulates the processor at an abstraction level between the RTL and the functional model. It presents most of the architectural details that are necessary for processor sizing, in order to evaluate in advance its performance capabilities in the MPSoC design. All these advantages come at the expense of a slower simulation speed and longer development time.

They can be obtained basically from three sources: standalone simulators, third party components and created by 'ADL' (Architecture Description Language) based tools. 'ADLs' modeling levels are classified into three categories: structural, behavioral, and mixed. Structural or cycle-accurate ADLs describe the processor at a low abstraction level (RTL) with a detailed description of the hardware blocks and their interconnection. These tools, such as

'MIMOLA' [103], are mainly targeted for synthesis and not for design space exploration due to their slow simulation speed and lack of flexibility. On the contrary, behavioral or functional ADLs abstract the microarchitecture details of the processor, and provide a model at the instruction set level. Their low accuracy is compensated by their fast simulation speed. Many languages exist such as 'nML' [104] and 'ISDL' [105]. Therefore, mixed 'ADLs' provide a compromise solution and combine the advantages of both the structural (accuracy) and behavioral (simulation speed) 'ADLs'. It is the best abstraction layer for design space exploration. 'EXPRESSION' [106], 'MADL' [107], 'LISA' [108], and 'ArchC' [109] are examples of mixed 'ADLs'. A recent type of processor description language called 'ArchC' [109] is gaining special attention from the research communities [110, 111, 112]. ArchC v2.0 is an open-source ADL, developed by the University of Campinas in Brazil. It generates from processor and ISA description files, a functional or cycle-accurate ISS in SystemC. The ISS is ready to be integrated with no effort in a complete SoC design based on 'SystemC' [8]. In addition, the ISS can be easily deployed in a multiprocessor environment thanks to the interruption mechanism based on 'TLM', which allows the preemption and migration of tasks between the cores. The main distinction of 'ArchC' is its ability to generate a cycle-accurate ISS with little development time. Only the behavior description of the ISA requires accurate description. As for the microarchitecture details, they are generated automatically according to the architecture resource description file. Since 'ArchC' is an open-source language, the simulator generator can be modified to produce a processor with customized microarchitecture enhancements, which makes it a great tool for computer architecture research [113]. To our knowledge, the processor model cannot be synthesized because it is not yet supported by 'ArchC'.

Instruction Level Power Characterization (ILPC)

Power-ArchC is built on the top of the ArchC methodology. ArchC provides an efficient framework for describing a processor architecture and ISA at behavioral or cycle accurate level. In addition, it automatically generates an ISS with a short development time, but it does not have any way to estimate its power consumption. The ArchC methodology is extended with power capabilities. Using power model from Instruction Level Power Characterization (ILPC) into Power-ArchC, the total energy used and average power consumed can be provided for a given benchmark, design implementation and technology node. The flow for instruction level power characterization is illustrated in Figure 4.3. Chain of tools is shown in black boxes. Input and output files are represented by grey boxes. The flow is built into four steps:

1. ArchC based description - ArchC 2.0 is used to generate the MIPS ISS that supports the full R3000 ISA. Binary codes are generated with a MIPS cross compiler provided in ArchC framework. From two input description files (ac, isa), the tool acsim generates a SystemC based ISS that can be easily integrated in a complex MPSoC model. The ArchC Simulator Generator (acsim) is composed by a simulator generator and a decoder generator. It uses the programs to extract the information from the model description files, in order to create all C++ classes and/or SystemC modules necessary to build the architecture simulator. The decoder generated by ArchC is capable of handling ISAs from simple RISC machines and multi-word of variable length instructions, like in many DSPs. The general simulator generation flow is shown in Figure 4.4.

2. RTL design and synthesis - An open source RTL description is used to obtain same functionality as from processor described from ArchC. Hence, to characterize power,

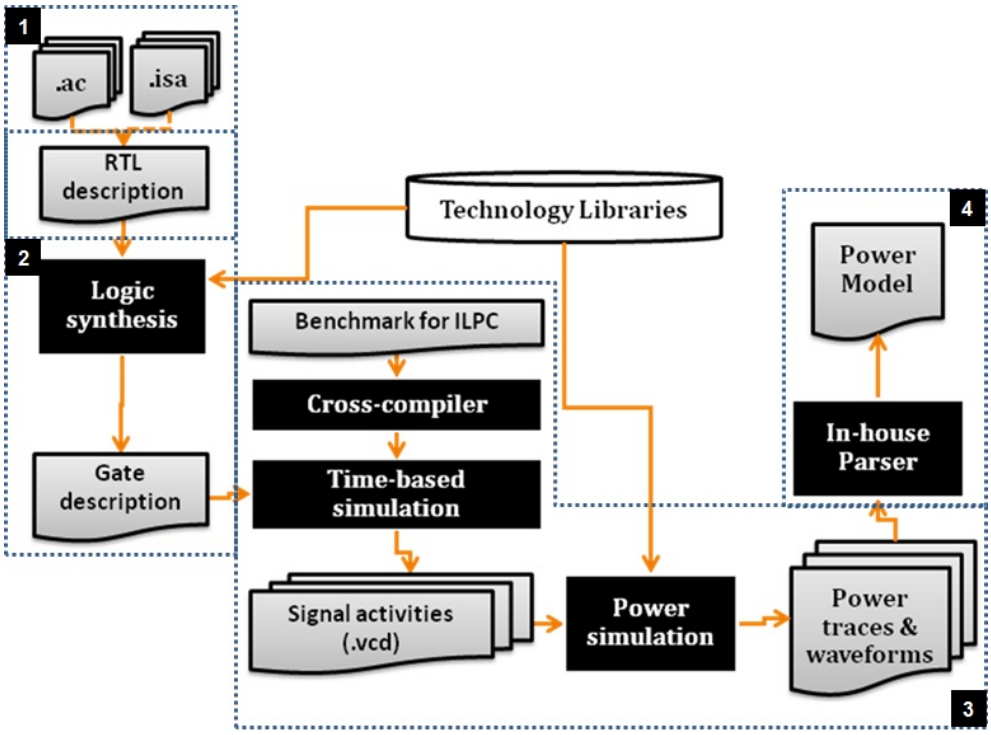

Figure 4.3: Instruction level power characterization to obtain power model at Functional level

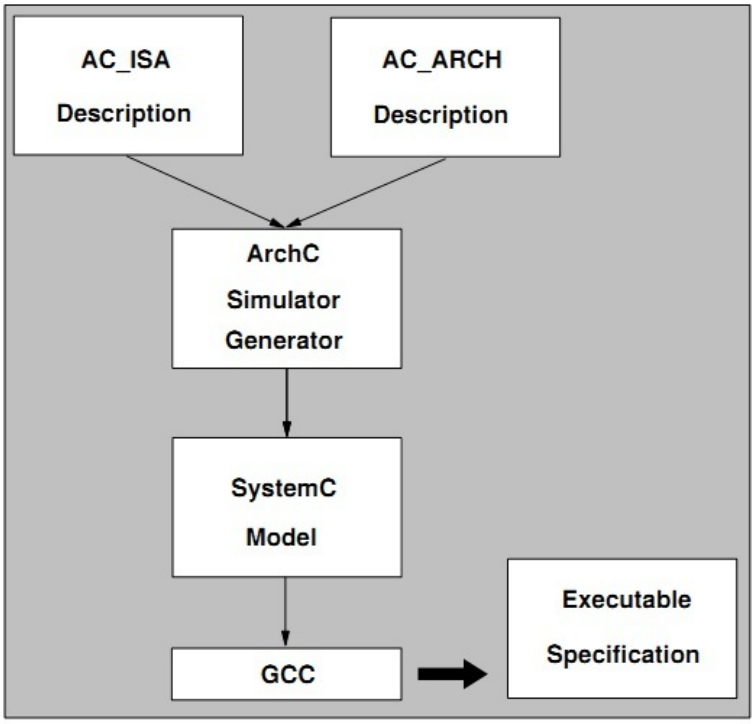

Figure 4.4: ArchC simulator generation flow

the design with same functionality at least at gate-level is required to provide sufficient details and accuracy to power values. Consequently, the RTL description of the processor is manually performed relatively to the architecture description made in ArchC. From that, commercial tools like Synopsys Design Compiler v2009.06 [114] generate a gate level generation, based on a chosen technology library. This explains why the design flow is currently semi-automatic.

3. ILPC - Power simulation tools at gate level can provide accurate power consumption of each block, at each clock cycle. Since power simulation tools at gate-level such as Synopsys PrimeTime [115] only provide power consumptions of hardware blocks, a parser tool is designed that outputs the average power consumption of each instruction from power and program traces provided by the simulation tool. The parser tracks the power value in the different pipeline stages flowed by the instruction. Based on the data path in each stage of the pipeline, it gathers the power value caused by the execution of one instruction and stores it in a model. Each instruction with same name will only have one entry in the previously defined model, and their different power values are averaged for each stage. One should note that the ILPC can be performed automatically whenever hardware implementation of microarchitecture or technology library changes.

4. Back-Annotation - Let's consider a set of three instructions (2 stores (sw) and 1 load (lw)) walking in the different stages of pipeline as shown in Figure 4.5 with the power values P1 to P9 generated by the gate level tool (other values are not shown for better clarity).

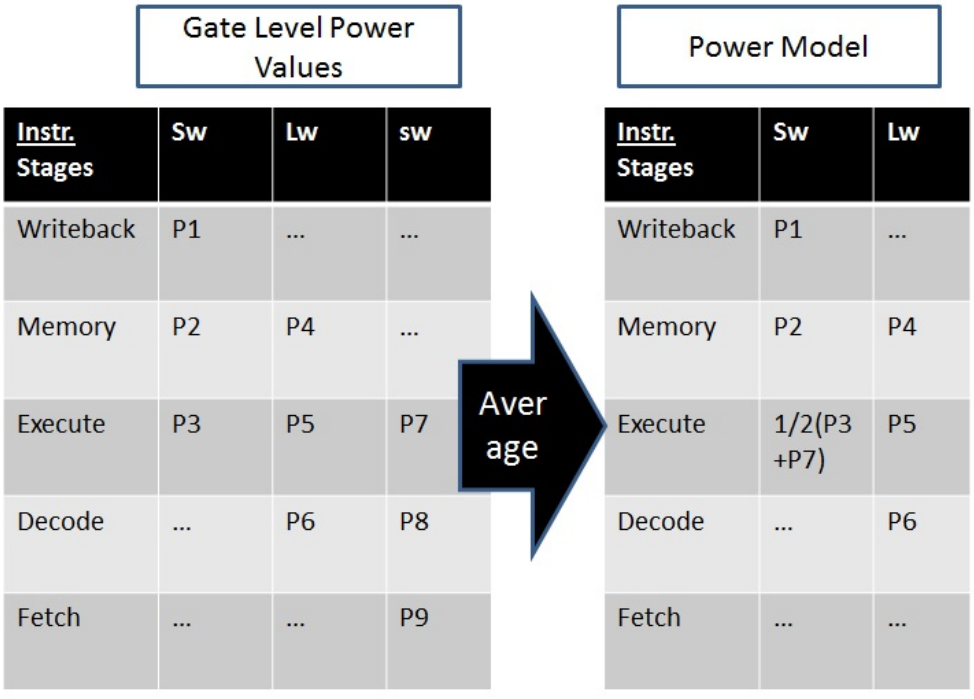

Figure 4.5: An example of generating power model for MIPS processor

In this case, ILPC outputs a power model that provides an average total power value for each instruction. In Figure 4.6, a random benchmark is shown at the top with random values of program counter (PC), opcode, instruction name and operands. On the right hand side is shown a power model generated from this benchmark as also explained in Figure 4.5. Finally, this power model can be back annotated to the benchmark at bottom with its corresponding values in power model. From this example, the building of power model is explained based on sample of instructions that is by computing an average value of gate level power values for same instructions. A characterization campaign has to evaluate all of the instructions for different operands and instructions interrelations. Since it is not possible to cover all the possible cases in a reasonable time, only a subset of possibilities is usually considered. In our case, ILPC outputs a power model that provides an average total power value (static and dynamic) for each instruction. The power model is next used to back-annotate the ISS generated by ArchC with power values for each instruction. The behavior description of the instruction contains now a variable that points to the corresponding instruction in the power model. The ISS is now able to output both instruction and power traces and total consumed power of the executed program.

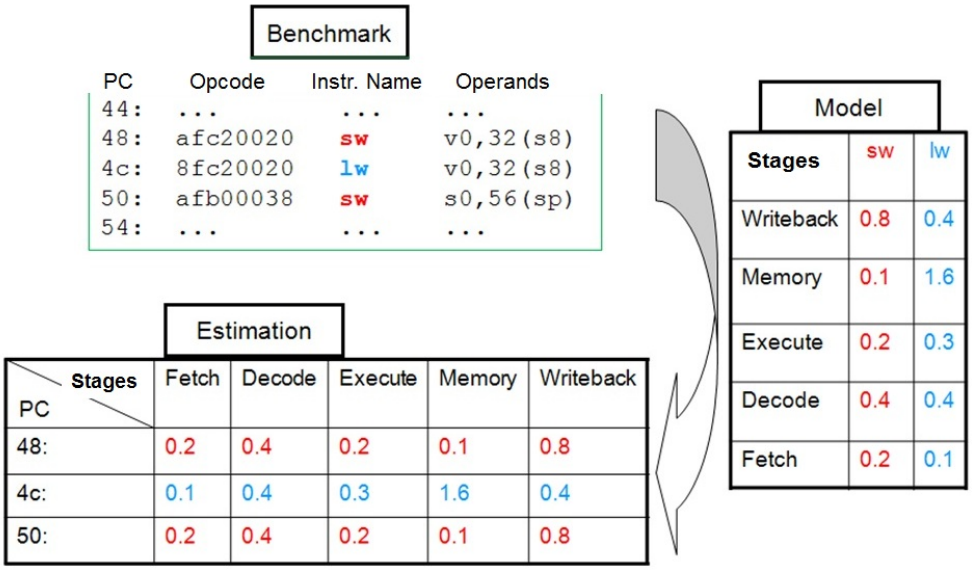

Figure 4.6: An example of generating power model for MIPS processor and using it to obtain power estimation for a random benchmark

Given an input power model, the compiler for host machine generates the modified ISS with power capabilities. A new ISS output provides power traces at the end of simulation of a benchmark. From that, it is possible to explore the power consumption at high level in a quick fashion. As we experience with performance accuracy at instruction-level, the resulting power accuracy will be obviously lower than the one obtained after synthesis. As previously explained, this is mainly due to the interrelations between the instructions during execution and the value of the operands that both affect the bit toggling activity of the processor micro-architecture and hence, the dynamic power consumption. As compared to Figure 4.4 that

provided overview of original ArchC simulator flow, Figure 4.7 shows Power-ArchC framework which has modified architecture of ISS generated using power model and ArchC simulator.

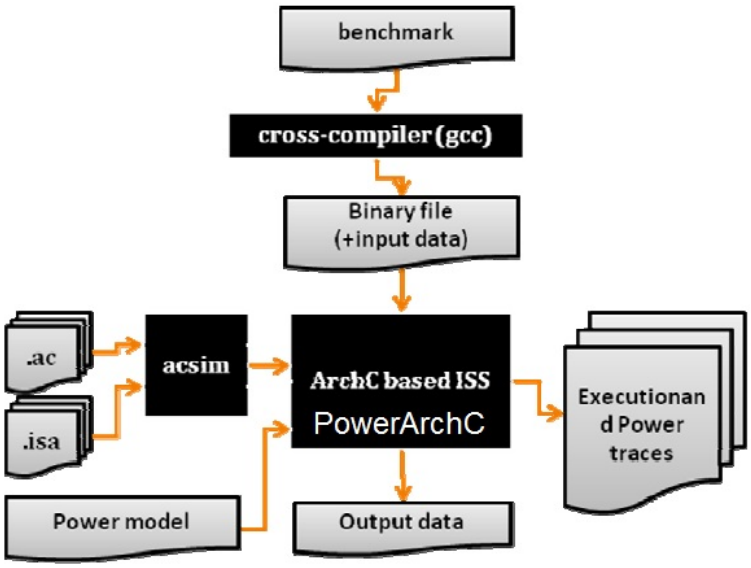

Figure 4.7: Power-ArchC Framework

4.3 Temperature modeling and simulations

Although there are some temperature simulation tools, it should be worth noting at this point that HotSpot is one open source academic tool which is famous among researchers (with citation count of 76 till date) for the flexibility it provides. Hence, this tool is used to complete the Tool chain and to provide temperature traces as input to RTME.

HotSpot [81, 80] is a tool developed by University of Virginia that can track temperatures at the granularity of individual micro-architectural units, so the equivalent RC circuit must have at least one node for each unit. It must be parameterized, in the sense that a new compact model is automatically generated for different micro-architectures; and portable, making it easy to use with a range of power/performance simulators. It is also BICI, that is, boundary- and initial-condition independent: the thermal model component values do not depend on initial temperatures or the particular configuration being studied. The software is a simple library that generates the equivalent RC circuit automatically (an example is shown in Figure 4.8, and, supplied with power dissipations over any chosen time step, computes temperatures at the center of each block of interest. The model is BICI by construction since the component values are derived from material, physical, and geometric values. Chips today are typically packaged with the die placed against a spreader plate, often made aluminum, copper, or some other highly conductive material, which is in turn placed against a heat sink aluminum or copper that is cooled by a fan. This is the configuration modeled by HotSpot.

The steady state temperatures are however a good enough estimate for the correct set of initial temperatures. HotSpot can be used for the steady state temperatures produced as the set of initial temperatures for the next 'true' run. It is possible to make HotSpot compute the steady state temperatures directly without going through the transient simulation, and hence making the simulations much faster and less memory consuming. HotSpot has steady state mode to compare temperatures predicted by HotSpot, and a simplistic model that eliminates the lateral portion of the RC circuit. HotSpot shows errors (with respect to the ambient temperature, 45°C or 318°K) always less than 5.8% and usually less than 3% [81]. HotSpot also include transient mode, the evolution of temperature in one block on the chip over time. Thermal capacitance is also necessary for modeling transient behavior to capture the delay before a change in power results in the temperatures reaching steady state.

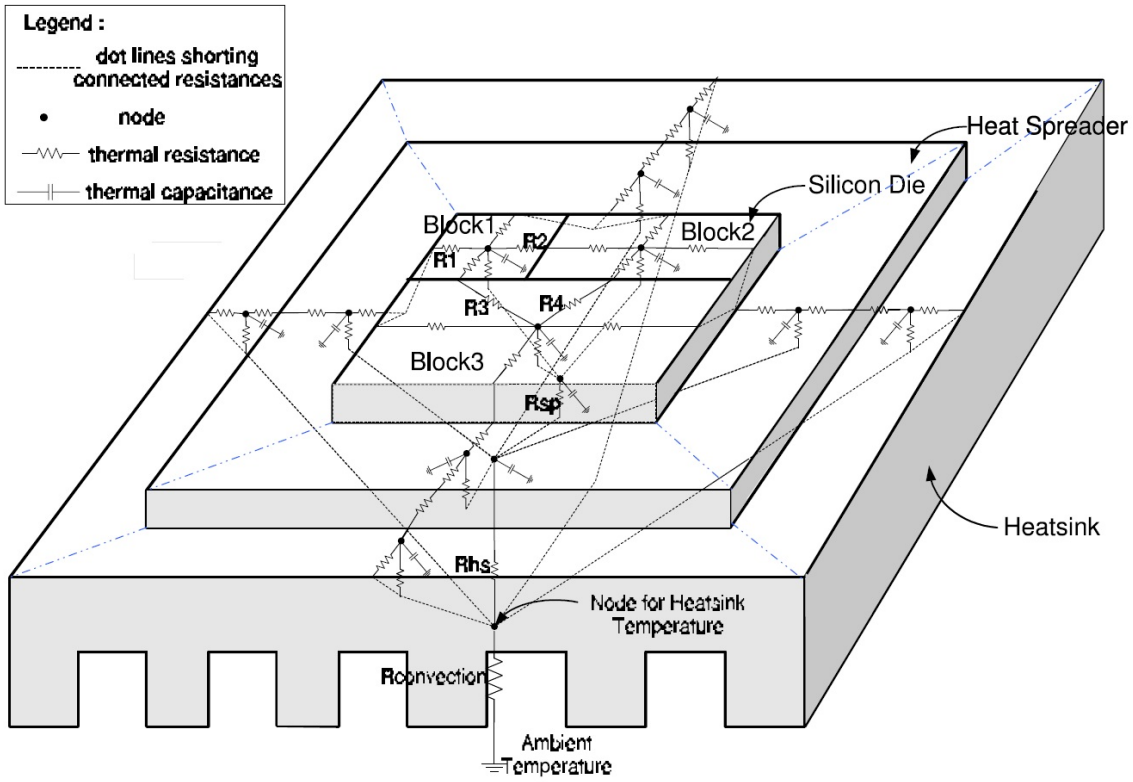

Figure 4.8: Example HotSpot RC model for a floorplan with three architectural units, a heat spreader, and a heat sink [81].

4.4 RTME (Real Time MTTF Evaluation)

RTME (Real-Time MTTF Evaluation) is an original work and a new simulation tool developed by our team for predicting Cumulative Failure Rate (CFR) of different blocks of the floorplan of the processor [78]. RTME as the name suggests is build to estimate real time MTTF values and with the help of these values CFR is obtained using the relation between MTTF and failure and Equation 2.19. It is capable of comparing aging behavior for different benchmarks and architecture choices, not bound to any specific technology or

power/temperature simulators. For now, it reads operating voltage-frequency (V-F) configuration and power/temperature traces of the floorplan blocks for different failure mechanisms. CFR represents the cumulative failure rate over the time of a failure mechanism. The failure library considers 4 failure mechanisms: NBTI, HCI, TDDB and EM. Their failure model at block level was detailed in Section 2.4. Relatively to power and temperature at block level, EM and HCI results depend on power consumption and temperature variations, while NBTI and TDDB results are based on temperature variations. Similarly to power and temperature, CFR is computed at each instruction execution i.e. each time step of Power-ArchC. Parameter n of CFR formula is so equal to the number of instructions in the executed benchmark. Parameter t_i is a constant value related to the frequency at which power and temperature are recorded. The i^{th} line in a power or temperature trace corresponds to the value measured at time step $(i-1) \times T$. Power and temperature are assumed to remain constant during time T. At each time step, RTME produces a CFR value for each failure mechanism and for each block of the chip floorplan. RTME is a simulation tool for predicting CFR (cumulative failure rate) using failure rate (λ) of different blocks of the processor at architectural level. The objective is to check the feasibility of the proposed design, before even synthesizing the circuit. The advantage of RTME is that it is a flexible tool, capable to compare aging behavior for different benchmarks and architecture choices, not bound to any specific simulator. RTME is believed to be hundreds of times faster than already existing tools at transistor level, but with reduced accuracy.

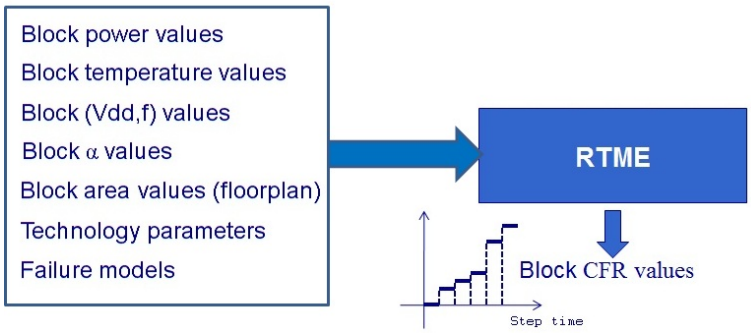

Figure 4.9: Real time MTTF evaluation

Table 2.1 listed the analytical TTF equations that model the behavior of studied failure mechanisms. It presents E-model for TDDB, Black's Law for Electromigration, Takeda model for HCI, and one of the phenomenological models for NBTI. In these models, the global factor of each model is not given: these factors are technology dependent and difficult to obtain. Hence, the resulting failure rate values must be considered as given in different arbitrary units. In RTME, to deal with CFR at functional level, the respective failure models are shown in Table 4.1 as derived in Section 2.4.

These failure models are provided as an input to RTME and in addition they also provide the dynamic but accurate statistics of following parameters to provide a good reliability prediction in useful life of the device: Probability of number of transistors switching at a given instant, Current Density, Substrate Current, Temperature, Area or number of transistors.

Finally, in RTME, failure models can be modified to remain true for a particular technology

Table 4.1: Failure models at functional level of abstraction derived using transistor level models. For better understanding parameters involved are defined in the third column from left. Refer to Table 2.1 and 2.2 for the definition of rest of the parameters.

Failure Mechanism	Failure Model (CFR)	Parameters Involved
Electromigration	$\sum_{i=1}^{n}(\frac{P_{bl-dyn}(a)^2}{B_0} \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a)$	P_{bl-dyn}: Dynamic power consumed by a block; a: Simulation time step of duration t_a; T_{bl}: Average temperature of a block; B_0: Constant depending on technology and initial operating conditions
Hot carrier injection	$\sum_{i=1}^{n}(\frac{\beta \cdot P_{dyn-bl}(a) \cdot exp(\frac{-e}{V_{DD}(a)}) \cdot exp(\frac{E_a}{K \cdot T_{bl}(a)})}{A_{HCI}^n} \cdot t_a)$	β: Proportionality constant
Time dependent dielectric breakdown	$\sum_{i=1}^{n} \frac{N}{2} \cdot exp(\gamma_{TDDB} \cdot V_{DD-bl}(a)) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a$	N: Total number of transistors in a block
Negative bias temperature instability	$\sum_{i=1}^{n} \frac{N}{2} \cdot A \cdot V_{DD-bl}^n(a) \cdot exp(\frac{-E_a}{K \cdot T_{bl}(a)}) \cdot t_a$	

library by modifying technology dependent parameters in header file of RTME. The header file contains information of parameters as shown in Table 4.2:

Table 4.2: Various parameters used in RTME

Designer Specific	Manufacturer Specific Technology Dependent Constants
k: Boltzmann's Constant: $8.62e^{-5}eV/K$	E_a: Activation energy for each failure mechanism
V_{DD}: Operating voltage in Volts	CFR_{th}: Threshold value of CFR at which the device is considered as fail
Freq: Operating frequency in MHz	t_{ox}: oxide thickness in μm^2
$Area_{blk}[num]$:Area of floorplan blocks	γ_{HCI} and γ_{TDDB}: Field acceleration parameter
t_s: Time step in seconds	n: Gate voltage exponent

Figure 4.10 gives the brief overview of RTME. The calculation of λ and CFR is performed for each time step as already explained. The steady values of CFR for each failure mechanism and for each block is generated at the end of simulation (n^{th} step) as shown in Figure 4.10. Finally, the tool provides values of CFR at user defined time using extrapolation. A point to be noted is that any high level power, temperature and performance simulator can be used to connect to RTME and fed their outputs to RTME. Hence, it is possible to show the variations in lifetime of various blocks of the chip using above discussed Power and Temperature Traces and other technology related parameters. The block can be a specific part of a processor or a complete processor in a multi-core architecture. Power and temperature traces are

provided for each simulation step generated by power and temperature simulators for each block. Power values provided for each simulation step define the stress on the chip, and during the simulation it may get varied due to DVFS i.e., due to dynamic voltage frequency scaling. RTME can easily handle these changes and provides the slope of CFR including the change in stress due to change in V-F set. The temperature values and hence the reliability also depend on the area, dimension and location of the specified block, it implies that the reliability gets effected by the floorplan of the chip. By making changes in the floorplan and analyzing the changes in temperature reliability of the chip can be enhanced.

Mode 1: Based on real time values

This mode of RTME is very important and is the base for other modes. In this mode the RTME simulator is integrated with power and temperature simulator to provide a single stand alone simulator that can provide power, temperature and CFR values per unit time in real time.

Mode 2: Based on trace values

As the name suggests, it is based on power and temperature values obtained per unit time and printed in a trace file. Corresponding to each power and temperature set failure rate (λ) and cumulative failure rate (CFR) are calculated. This mode comes into effect after the power and temperature simulations and not at real time.

Mode 3: Based on average steady values

RTME calculate the average values of power consumption and temperature during the execution of an application on the processor. According to P_{avg} and T_{avg} RTME calculate and provide the final value of CFR at the end of execution. Mode3 is much faster than Mode1 and Mode2 as it performs less complex calculations. This mode is very interesting in case of linear extrapolation as shown in Figure 4.10 even if it is a trade-off between accuracy and speed.

It is understandable that even at functional level of abstraction we cannot simulate power and temperature (and hence CFR) for long time of executions (for example 1 year) due to memory size limitations and time to simulate. Hence, extrapolation is required to predict the future values of CFR. The extrapolation is done using the slope from initial and final CFR values for the execution of various benchmarks for one or many voltage-frequency sets and different ambient temperatures. For example, designer can predict CFR for 1 year of execution on a processor considering data in Table 4.3. Consider a processor which is used, by application (app) 1 for 6 months and for maximum operating conditions of V-F set, by app 2 for 3 months for high performance V-F set and, by app 3 for 3 months in slow speed mode. The assumption is that the order of execution has no effect because the transition effect is neglected between different applications. Using RTME, designer can find the final CFR after 1 year of usage for different operating conditions using extrapolation.

Mode 4: Based on maximum values

This mode is similar to mode 3, here, RTME only search for the maximum values of power consumption and temperature during the execution of an application on the processor. According to P_{max} and T_{max} RTME calculate and provide the final value of CFR at the end of execution.

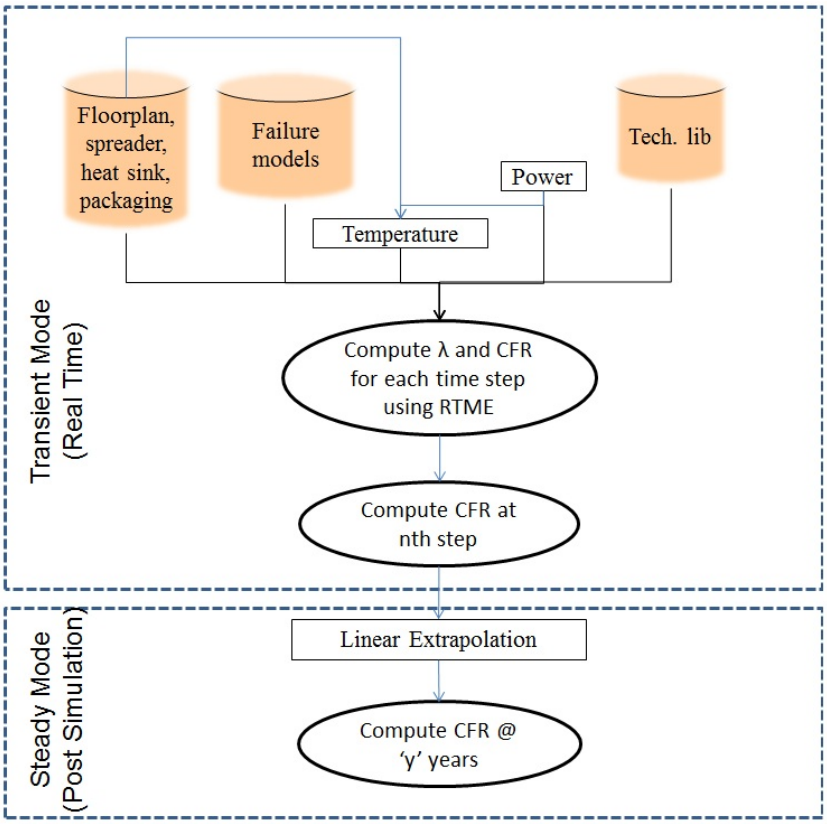

Figure 4.10: Real time MTTF evaluation

Table 4.3: An example to explain linear extrapolation of CFR results using RTME in steady mode

Total Time Duration = 1 Year

Time usage	Application	V-F
50 %	App 1	1.21V-373Mhz
25 %	App 2	1.1V-267Mhz
25 %	App 3	0.81V-54Mhz

The accuracy of CFR for each block is affected by the assumptions made for estimation at each abstraction layer. Other factors that may affect the RTME accuracy are the different technology parameters and the accuracy of other tools used to simulate power consumption and temperature. It will be shown that state-of-the-art tools are used with their own level of accuracy, the error is estimated in different ways and in addition failure models have different dependence with various parameters.

4.5 RAAPS - The Tool-Chain

This section describes the whole reliability simulator including RTME at functional level called Reliability Aware ArchC based Processor Simulator (RAAPS) methodology as illustrated in Figure 4.11 and explained below. To estimate the reliability of a processor at functional level, power and temperature values at functional level are required, as well. For

temperature, again power consumption values are needed, packaging characteristics and the processor floorplan. Power at functional level is obtained through Instruction Level Power Characterization ILPC [79] performed with a power simulation tool applied to a gate level description of the processor. The behavior of instructions at functional level is only simulated; the first step is to model the power contribution of each instruction. As shown, in the left part of Figure 4.11, the RTL design of the processor corresponds to the microarchitecture and instruction set architecture (ISA) descriptions, made in ArchC language at functional level. From that, a synthesis tool like Design Compiler [114] generates a gate level description, based on a chosen technology library.

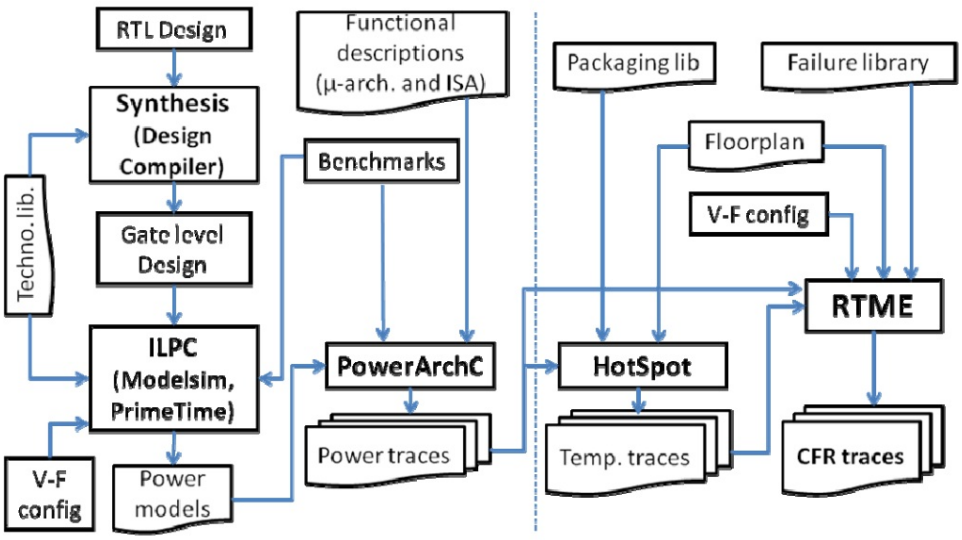

Figure 4.11: Reliability simulation methodology at functional level using RTME

To characterize power, ILPC at gate level is performed with ModelSim and PrimeTime [116, 115]. It can provide an accurate power consumption of each block, at each clock cycle. A parser is designed that outputs the average power consumption of each instruction from power and program traces provided by the simulation tool. An assumption is taken to achieve the Power model i.e., not all possible combinations of instructions and operands are considered. The power model is next used to back-annotate the ISS generated by ArchC with power values for each instruction. The behavior description of the instruction now contains a variable that points to the corresponding instruction in the power model. The ISS is now able to output both instruction and power traces and total consumed power of the executed program using the modified ISS version with power capabilities i.e., using Power-ArchC. A Power model refers to chosen operating condition (Voltage, Frequency) and characterization campaign (Benchmark). As shown in the right part of Figure 4.11, temperature traces are obtained using the tool named HotSpot. Using a floorplan, a Packaging library and Power traces of the whole processor, HotSpot can derive steady and transient temperature values of each block of the floorplan.

4.6 Conclusion

After discussion in Chapter 3 about already existing reliability simulators and methodologies, a new reliability simulator is presented called RTME. It is developed to help designers and manufacturers by calculating Cumulative Failure Rate (CFR) parameter. CFR can be calculated for various failure mechanisms like, Electromigration, Hot Carrier Injection, Negative Bias Temperature Instability and Time Dependent Dielectric Breakdown. RTME can calculate CFR for different blocks of the processor, according to the provided floorplan and for different applications that can run on a processor separately or together. A Tool-chain is presented that is built around RTME at functional level of abstraction. This Tool-chain also provides details regarding integration of RTME with a functional level power simulator and a power trace dependent temperature simulator. Here, it must be noted that developed reliability simulator, RTME, can be integrated with other power and temperature simulators at various levels of abstraction by easily adapting the inputs. An example is the integration of RTME in a cycle accurate Tool-chain with Wattch and Hotspot as in [78]. At cycle accurate level, designer can have more detailed results, like CFR for each block of the processor separately. Figure 4.12 includes RAAPS Tool-chain in the state-of-the-art. It can be seen that RAAPS has certain advantages and limitations in respect to other existing tools. One most important advantage is the integration of reliability simulator in a design flow. Chapter 5 take first step towards validation of Tool-Chain and also verify the accuracy of Power-ArchC (a power simulator at functional level developed as a part of RAAPS). A case study will be presented to help both designer and manufacturer and will be explained in details in Chapter 5. Some of the points are explained to show, how designers and manufacturers can use the results of the developed Tool-chain, RAAPS.

Level of Abstraction	Name of Tool	Author and University/Co.	Pros	Cons
Transistor level	BERT	Univ. of Berkeley	Very accurate, EM, HCI and TDDB	Aging parameters, Small-step iterative algorithms, Very slow, if we consider a complete chip (not feasible) (In PRESS - automatic management o f transient analysis step)
	ULTRASIM	Cadence	Specialized models for HC and NBTI, Isub is the primary reliability parameter	
	PRESS	University of Twente, Philips	HC, script based solution, no changes to adapt original simulator	
Gate level	GLACIER	BTA Technology	HC, accurate to handle large circuitry, 1% error compared to BERT, GUI	Considers each transition, hence very slow in a circuit with billions of transistors
	ILLIADS	University of Illinois	Can perform, timing, temperature and reliability simulations, approximately 1000 times faster than spice oriented tools	No specific failure models, 10 minutes of real clock for one clock cycle output for 235,000 transistors (not enough for large simulations on a processor)
Functional level	RAMP	University of Illinois	Good starting point at high level of abstraction, methodology in design flow, EM, TDDB	Lots of assumptions, some accurate, some inaccurate, uniform device density over chip
	-	Coskun *et al.*, University of California, San Diego	Methodology using DTM and DTM in MPSoCs	Assume exponential lifetime distribution, Not implemented the methodology in design flow, results with arbitrary values
	AGESIM	Chinese Univ. of Hong Kong, China	Good new mathematics for calculation of aging rate	Not implemented the methodology in design flow, results with arbitrary values, i.e., no technology libraries are taken in consideration
	RAAPS	IMS, University of Bordeaux 1/ CEA LIST	Mathematics for Failure models (EM, HCI, NBTI, TDDB) at Functional level, Integrated power and reliability simulators in design flow , Power results with close to real world values, 15 times faster than cycle-accurate	RTL description performed manually in relation to architecture description made in ArchC, hence, semi-automatic design flow. Not all the modes are finished. Variability is not considered yet

Figure 4.12: RAAPS advantages and disadvantages in relation to state-of-the-art as discussed in Figure 3.21

Chapter 5

Validation and use cases

Contents

Previous chapter discussed about RTME, developed to help designers by calculating Cumulative Failure Rate (CFR) parameter which was defined and explained in Chapter 2. RTME is able to calculate CFR for different blocks of the processor or for complete processor, according to the details provided in the floorplan and for different applications that can run on a processor separately or together as explained in Section 4.5. A tool-chain was presented that was built around RTME at functional level of abstraction called RAAPS (Reliability Aware ArchC based Processor Simulator). RAAPS is developed using RTME, HotSpot and a new power simulator called Power-ArchC (based on ArchC- Architecture description language), to simulate power consumption at functional level of abstraction. It is shown that RAAPS has certain advantages and limitations in respect to other existing tools. One most important

advantage was the integration of reliability simulator in the design flow. In the current chapter first steps towards validation of RAAPS tool-chain are taken. It also verifies the accuracy of Power-ArchC. The RAAPS tool-chain is used to predict accurate reliability values of a processor at functional level. The effect of different power scenarios on the reliability of a MIPS processor will be shown for TSMC 40nm technology library. The MIPS processor is chosen as it is a very common processor among researchers. It is easy to obtain open source code of MIPS-ISA already realized at various abstraction levels and hence very useful for comparing results between these levels, for example HMC-MIPS [117] at RTL and MIPS-IV ISA using SimpleScalar [118] at cycle-accurate level. The results will be presented for specific set (until and unless specified otherwise) of Voltage-Frequency, i.e., 1.21V-373MHz for high performance and high energy. Finally, some uses of the RAAPS tool-chain are given that can be used by designers to control the processor reliability.

5.1 RAAPS in design flow

Before starting the discussion about the quality evaluation of RAAPS and its uses and limitations, readers must understand who can use RAAPS and where is the place of RAAPS tool chain in the design flow. RAAPS target specific type of semiconductor companies, as it needs information at different levels of the design flow, to provide more accurate reliability prediction data. In Figure 5.1, the design flow is explained. To keep up the speed of the design flow, various tools and methods were introduced in chip design to automate the flow and produce quicker and better results, such as, IC design moved from putting transistors by hand through designing the logic using standard gates with schematic entry to describe the logic at a higher level of abstraction. System designer's work at front end involves defining software interface, defining the contents of the chip, and defining the interfaces to communicate with external components. It also involves modeling Hardware/Software systems at higher level and run simulations to see if the performance requirements are met. Job of a front-end designer is to provide technology independent design entry and design verification. In most cases, the work can be reused for various technology nodes and fabrication plants. Whereas, a back-end designer performs technology related implementation and requires additional effort for implementation in different technology nodes or fabrications and hence limited re-usability. According to [119], there exist different categories of enterprise in semiconductor industry, named 'Fabless,' 'Design House,' 'Foundry,' and 'IDM'. Fabless, receive the specifications from their customer, and work completely on the designing part from system level design until layout design and perform quality assurance, i.e., verifying if the product meets the customer requirements. Design house, are generally very similar to Fabless, but they do not perform back end designing and qualitative assurance. Foundries work completely at back-end design i.e., fabrication and mask making. Finally, there is integrated device manufacturer, IDM, an enterprise who work from beginning to end, on all the aspects of design flow, from creating design specifications to manufacture and provide the product to the customer.

Intuitively, RAAPS target IDM, as it needs information at different levels of design flow, to provide more accurate reliability prediction data. As shown in Figure 5.2, how a designer at high level of abstraction can use RTME and hence RAAPS to predict reliability of his design using parameter values provided by back-end designer. The graphs that follow in this section are the results that can be used by IDM to provide their designs more reliability.

It can also be used by a fabless enterprise to explore different design choices based on ITRS

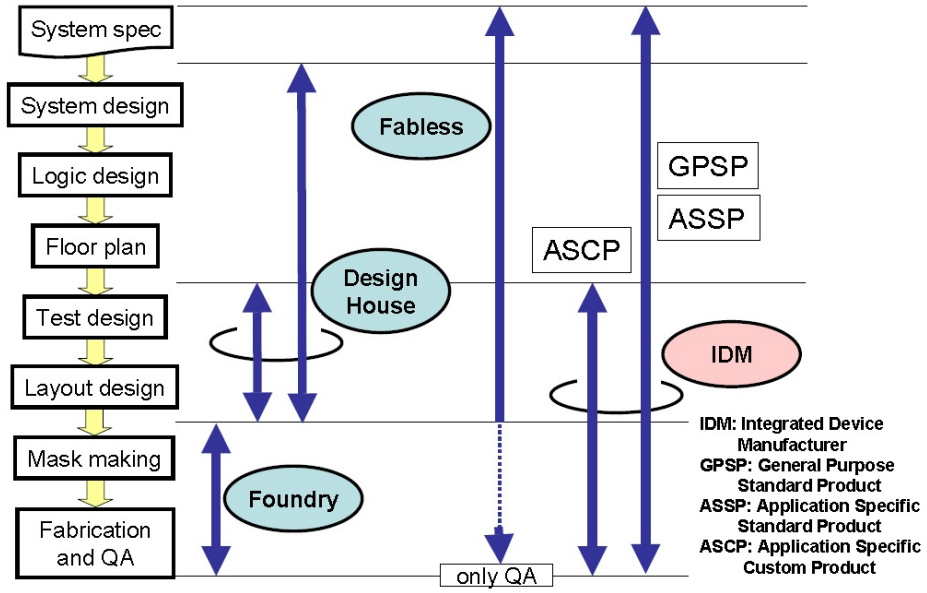

Figure 5.1: Design flow and business model [119]

or literature based technology related parameters.

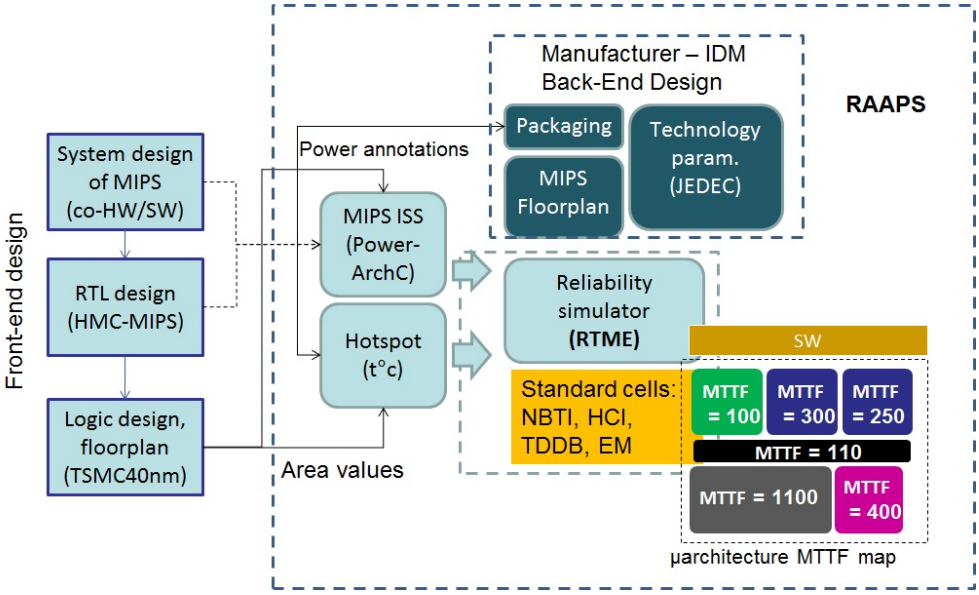

Figure 5.2: RAAPS integrated in the design flow, various modules of RAAPS need inputs from various abstraction levels, such as, HotSpot needs floorplan and packaging information as an input from logical design level and MIPS design at RTL and system level

In the next Section 5.2, the complete framework is explained to make understandable for the readers the requirements for the complete analysis of results and validation from RAAPS Tool-Chain.

5.2 Framework Design

The methodology was discussed in Section 4.5 i.e., RAAPS tool chain for a MIPS 32-bit processor. This section is a first step to validate the quality of RAAPS tool chain and discuss about its performance and accuracy. To do the validation, a processor design is required at functional level and RTL as explained in Section 4.5.

5.2.1 Case study using MIPS processor designed at RTL and functional level

ArchC v2.0 is used to generate the MIPS ISS at functional level that supports the full R3000 ISA. AC_{ARCH} (ArchC based architecture element description) is used to simulate the instruction set of the MIPS architecture, without considering its pipeline. This makes the instruction behavior description very simple and also demands few structural information. AC_{ISA} (ArchC based Instruction set architecture) description of MIPS model contains instructions, formats, decoding information and assembly syntax declarations, illustrating the main characteristics of an ISA description in ArchC. Every instruction must have a previously declared format associated to it. The designer declares an instruction through the keyword ac_{instr}. The generated simulator executes one instruction per cycle. The data is stored into arrays and is accessible through read and write methods.

An open source RTL description of MIPS that is HMC-MIPS [117] is considered. It is a project handled by HMC's and Adelaide's Universities to create a MIPS in Verilog language. It includes a 5-stage pipelined processor that mostly supports MIPS ISA. It includes thirty-two general-purpose 32-bit registers and fifty-eight instructions, each 32 bits long. It does not include support for an FPU. It also includes data and program cache memories and a RAM. Some modifications are applied in HMC-MIPS. Especially cache memories are deactivated that are not modeled in MIPS ArchC. An overview of MIPS pipeline implemented in HMC-MIPS is given in Figure 5.3, and the floorplan is divided into 7 blocks as, fetch, decode, execute, memory, writeback, control and clock. Two examples are shown in Figure 5.4. Using such floorplan, power simulations and hence temperature simulations can also be performed at block level.

A Processor designed at Instruction level or RTL has much less details about the design. With these libraries and Synopsys Design Compiler v2009.06 [114], at gate-level the processor description from HMC-MIPS is synthesized. Then, some of the MiBench benchmarks [120] executed with Mentor Graphics ModelSim v6.5b [116] are simulated to generate the execution trace of the entire processor and verifies the correct functionality at RTL. The simulations are assumed to be under consideration of ideal environment, with no humidity and no variability. The results are provided for the whole processor without system memories (i.e., icache and dcache are not present on the chip).

After that, the MiBench benchmarks are simulated with Mentor Graphics ModelSim v6.5b to generate the execution trace of the entire processor. Binary codes are generated with a MIPS cross compiler provided in ArchC framework. A specific link script is made for HMC-MIPS target. Then, Synopsys PrimeTime PX v2008.12 is used to generate a time-based power analysis and to report power traces.

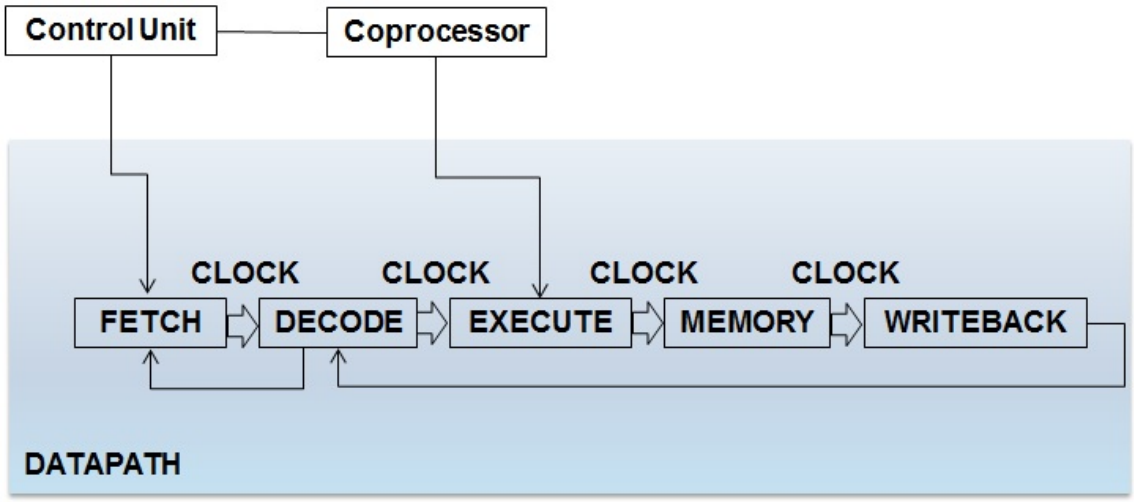

Figure 5.3: An overview of MIPS processor block diagram

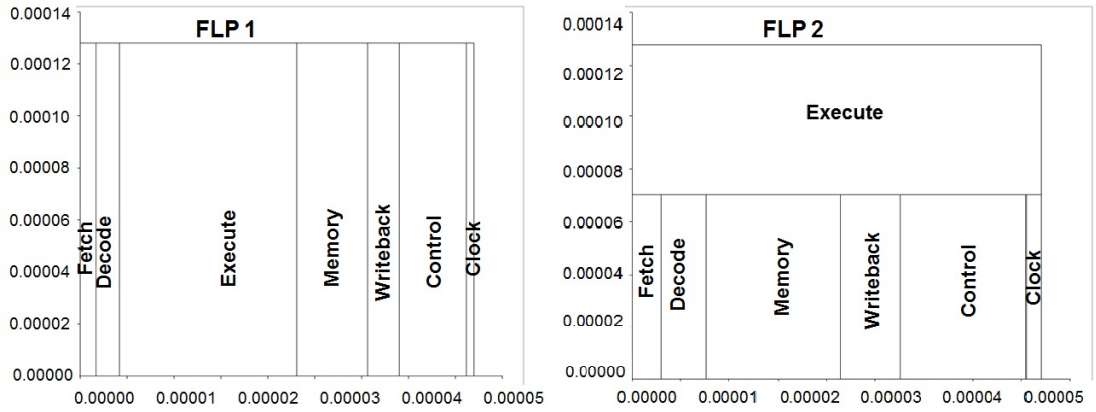

Figure 5.4: Floorplans of HMC-MIPS processor. Two floorplans are shown with different placements

5.2.2 Standard benchmarks

MiBench [120] is a benchmark suite representative of embedded computing software that mostly runs on real applications. It is including automotive, consumer, network, office, security and telecomm application examples that will help us to prove that our work is consistent for the embedded domain. The following benchmarks are used:

1. 'Quick sort' of words, 'Susan' and 'bit count'ing software
2. 'JPEG' compression
3. 'Dijkstra' and 'Patricia' network examples
4. 'CRC32' and 'FFT' computing and 'GSM' encoding

5. 'Rijndael' security tool

The instruction distribution for each benchmark is shown in Figure 5.5. Although, all these benchmarks, include lot of calculations, and hence include maximum of 'arithmetic' type instructions, the instruction distribution still vary a lot between two benchmarks.

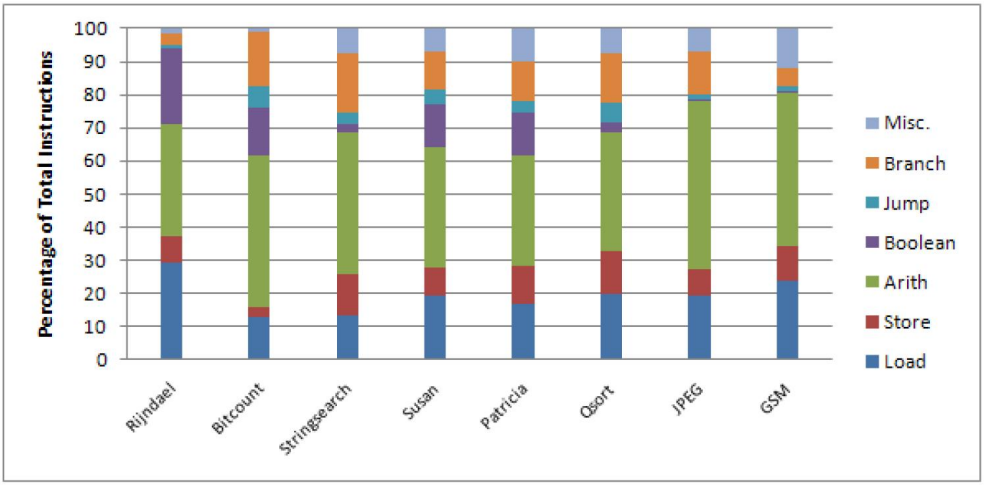

Figure 5.5: Instruction distribution in MiBench Benchmark suite

5.2.3 Random benchmarks

The power consumed in executing an instruction is also related to the instruction executed before it. There is relation between the instructions executed before a particular instruction. Hence, it is important to study different combination(s) of instructions executed together. Since, the benchmarks in any benchmark suite are very specific and do not cover all these combinations, some random benches are required. The random benches (written in assembly language) are generated using a script in Perl. A weight table is made in the script to define the probability of occurrence of each instruction type in the random benchmark. The number of instructions and operands in the bench varies every time. The bench ends in such a manner that it can be executed on both pre-synthesized and post-synthesized HMC-MIPS processor. Each bench is with approximately 16,000 instructions. All set of operands for each instruction cannot be executed due to time limitation. The cache memory limits the number of instructions in one benchmark. Hence 20 different random benchmarks are generated and hence provide one final power model to improve accuracy in power model. The results use a more accurate power model, obtained by executing approximately 300,000 instructions (20 random benches). The bench ends in a manner such that, it can be executed on both pre-synthesized and post-synthesized HMC-MIPS processor.

5.3 Performance and accuracy evaluations of RAAPS

5.3.1 A quick comparison between two abstraction levels using standard benchmarks

In this section, it is shown, why functional level of abstraction is chosen to study and analyze the results over cycle-accurate level. It will be shown that functional level can be used when very high performance is required. Two simulators are used, Power-ArchC (based on ArchC) and Wattch (based on SimpleScalar). The MiBench benchmarks come with their source code and need to be compiled for SimpleScalar (simulator generator at cycle-accurate level of abstraction) or ArchC and Design Compiler simulators. Concerning Wattch, it uses an old version of gcc (v 2.7.2.3) and needs some bug fixes that are reported in a script. It should be noted that Wattch needs little endian binaries to run; hence a little endian cross compiler is used. For ArchC, executables are directly taken from ArchC website, compatible with the MIPS32 model. This difference implies that benchmark binaries are not exactly comparable because instruction sets are different. To avoid this problem, it is decided to relate the 2 ISAs, as Wattch instruction set is made of 111 elements whereas ArchC has 223 (ArchC MIPS32 instruction set is more up to date than MIPS-I in Wattch). Then a link is created between new instructions in ArchC and older ones in Wattch. These relations can be found in an ArchC simulator parameter file named OPCODES.relation. It is a text file where each line gives the opcode of the Wattch instruction related with ArchC instruction name. The 2 instructions have the same power characterization values. Running the benchmarks on a Intel(R) Xeon(TM) CPU at 3.60GHz produces the statistics as shown in Figure 5.6

	SimpleScalar+Wattch		Power-ArchC		
Benchmarks	No. Of kInstr.	Execution time	No. Of kInstr.	Execution time	Performance gain
qsort	10337	2min55s	14413	5s	35
susan	4925	11s	6988	2s	5.5
Bitcount	50321	1min54s	45594	11s	10.36
JPEG	20919	2m40s	29475	6s	26.6
Dijkstra	75906	3min21s	59354	12s	16.75
Patricia	268888	9min8s	289206	1min14s	7.4
GSM	44425	1min45s	32664	8s	13.12

Figure 5.6: MiBench: benchmark execution information

Here, it can be concluded that Power-ArchC (power simulator based on ArchC) is around 10-15 times faster than Wattch (power simulator based on SimpleScalar). The two tools, Power-ArchC and Wattch are very different, it is further analyzed by studying the results that both provide similar level of accuracy, although, no comparative results could be provided, since the two MIPS generated using SimpleScalar and ArchC are different. In the subsection

5.4.1, the results obtained using Power-ArchC are analyzed and these results will be further used to predict CFR and accuracy of RAAPS.

The results are also provided for random benchmark generated in Subsection 5.2.3 and the power model generated using these random benches named ILPC4. Figure 5.7 shows the power simulator performances and accuracy. Third column compares the execution time of a benchmark at RTL (PrimeTime) and at functional level (Power-ArchC). The last column shows the percent error, where Dynamic power from RTL provides the theoretical value and Dynamic power from Power-ArchC is calculated using power model obtained with the help of the random program.

Benchmark	# instr.	Simulation time (min)		%error
		PowerArchC	PrimeTime	
Qsort	4741	<1	~60	6.24
Motion	30558	<1	~240	1.57

Figure 5.7: Performance and accuracy comparison between Power-ArchC and PrimeTime

5.3.2 Impact of standard deviation of Power and t°C on CFR- EM, HCI, NBTI, TDDB

Figure 5.8 shows the dynamic power values for the different classes of MIPS instructions in our power model. Figure 5.9 shows the dynamic power of a collection of MiBench benchmarks [120] which are typically used in embedded systems. Using benchmark profiling (Figure 5.5), 'Rijndael' has a higher percentage of 'Boolean' instructions and less of 'arithmetic' instructions that explains the high dynamic power of 'Rijndael' benchmark. In contrary, 'GSM' has high percentage of 'arithmetic' instructions and other 'miscellaneous' instructions (e.g., nop, mfhi), and less of 'Boolean', and hence less dynamic power compared to others. To conclude, with these results it can be said that, the higher the percentage of arithmetic and miscellaneous instructions is, the lower the dynamic power consumption is.

As already discussed, Power-ArchC generates power model using ILPC with average values of dynamic power consumption for each instruction with different operands. To estimate the accuracy and the range of power values for one type of instruction, the standard deviation is calculated: it shows how much variation or "dispersion" there is from the "average" value of each instruction. Two separate power models are generated with average plus deviation and average minus deviation power values using a random benchmark as discussed in Section 5.2.3. Percent deviation for a subset of instructions is shown in Figure 5.10.

The deviation is approximately between 2% to 5% from average. It is also seen that percent deviation from average is same irrespective of change in operating conditions. Deviation is approximately between 2% to 4% for a complete benchmark as shown in figure 5.11.

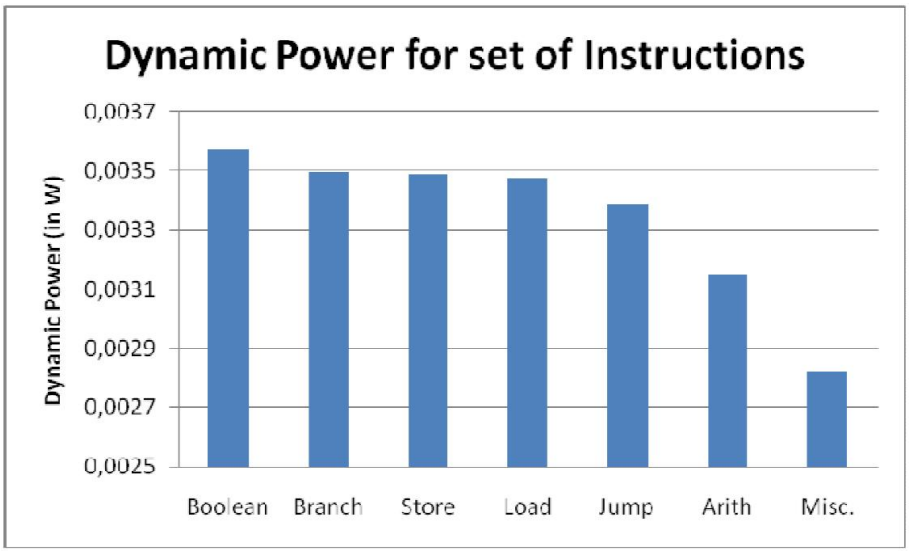

Figure 5.8: Dynamic power for each type of instruction set in average

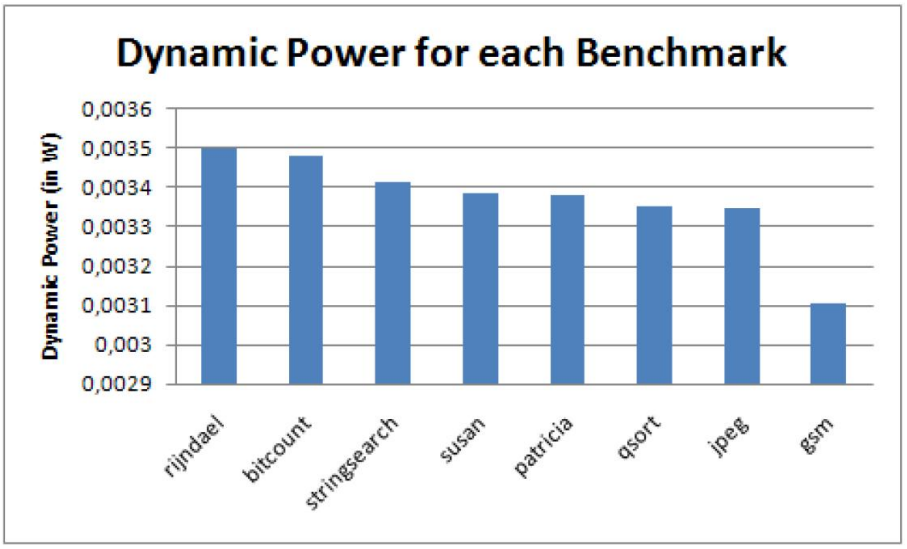

Figure 5.9: Total dynamic power for each benchmark

In Figure 5.12, the accuracy of the Tool-chain, RAAPS to calculate CFR is shown in terms of percentage error: it is the estimated value minus the true value divided by the true value and multiplied by 100. In the case of 'Patricia' there are more samples to estimate percent deviation which generally improve accuracy, even though the precision may be worse.

Power and temperature should be studied using left hand y-axis and CFR should be studied using right hand y-axis. The error is calculated in relation to accuracy obtained from power and temperature simulators. Percentage error is calculated using the average value as reference or true value and instantaneous values as experimental values. In Figure 5.12, the graphs provide knowledge of percentage error migration inside the tool chain, i.e., to show, how the

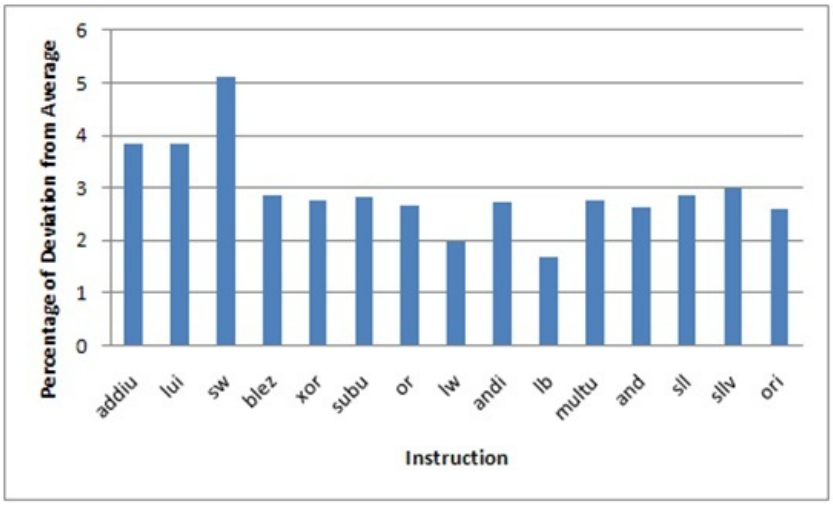

Figure 5.10: Percent deviation from average for individual instructions

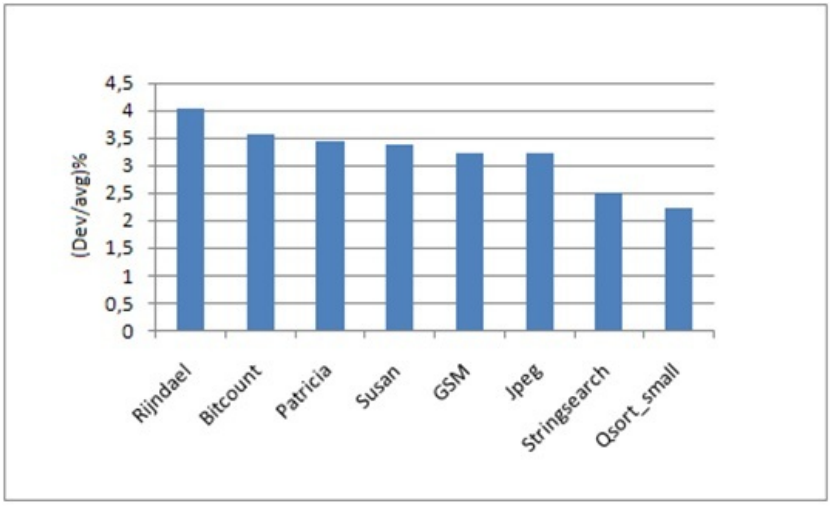

Figure 5.11: Percent deviation from average for different MiBench benchmarks

accuracy of Power-ArchC affect the accuracy of HotSpot and hence the accuracy of RTME. A point to note in Figure 5.12, it is assumed the maximum percentage error in HotSpot to be 6% as provided in [80]. The percentage error in calculation of: CFR_{EM} is approximately 31% to 34%, CFR_{HCI} is approximately 8% to 10%, CFR_{NBTI} is approximately 11.8% to 12.7% and CFR_{TDDB} is approximately 21.8% to 23%. The accuracy is depending on many factors such as power and temperature simulators attached to RTME, the run length of a given benchmark and so on. Hence, the accuracy results provided may vary in other conditions. Until now, the accuracy of RAAPS cannot be compared with other tools as it is based on very different frameworks and abstraction levels. Around 30% of standard deviation seems to be very high to be accurate but it is not the case, because RAAPS is currently used to provide comparative studies between two designs with same functionality.

The dynamic power traces for each benchmark are used to estimate CFR, and the deviation

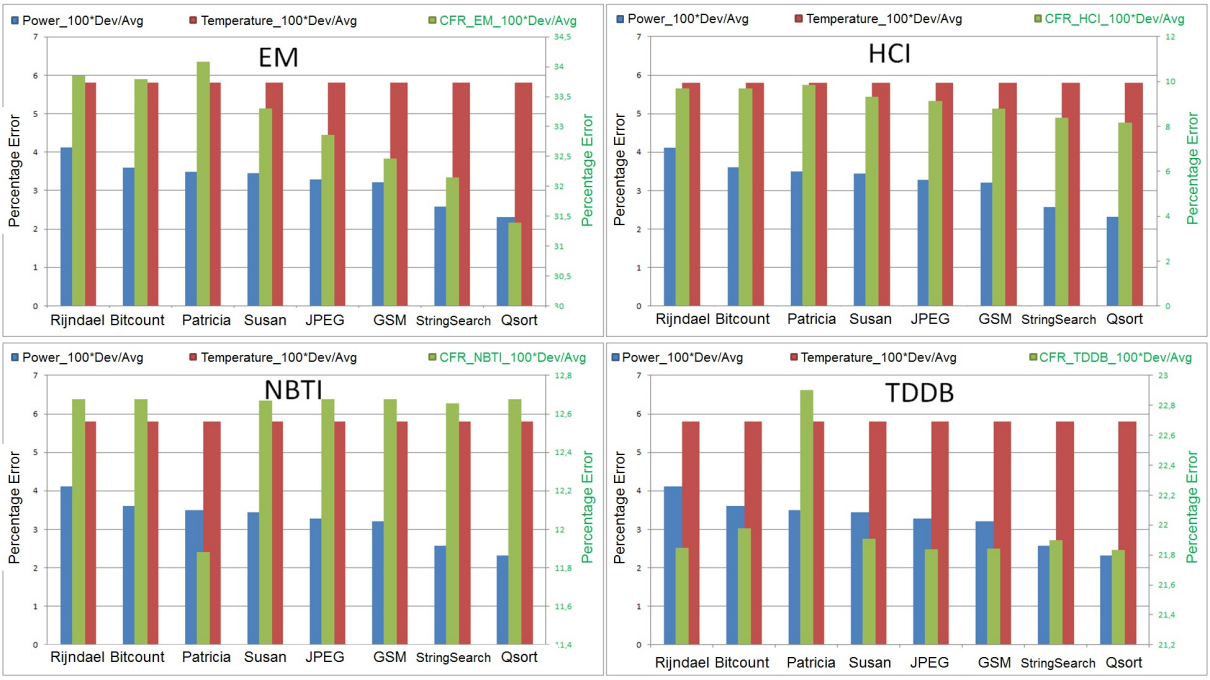

Figure 5.12: Migration of percentage error from Power-ArchC to HotSpot to RTME. The left hand y-axis is used to show percentage error for power and temperature, whereas the right hand y-axis is used to show percentage error for CFR.

migrates into CFR. Also, effect of exponential relation to temperature is observed.

5.4 Impact of energy consumption on MIPS CFR

The study of results obtained using ILPC and hence Power-ArchC is divided into two parts as in subsection 5.4.1 and 5.3.2 to provide different level of ease and accuracy:

1. At first, three power models are generated using three different ILPC campaigns using standard benches (i.e., Motion, Qsort and average of both) to provide the power and energy for each instruction separately and then to find the relation between different obtained power and energy models. Motion and Qsort being small and understandable benchmarks provide a nice overview of the results obtained. Results are provided in Subsection 5.4.1

2. In the second part, a big but random benchmark as explained in Subsection 5.2.3 is generated to obtain a wide variety of operands to be executed on a processor for each instruction. This will help to find a relation between each benchmark executed on MIPS (generated using Power-ArchC) and the power values for each instruction in the power model generated using ILPC. Also, the results will be used to obtain percentage error in Power-ArchC with respect to PrimeTime in Subsection 5.3.1.

5.4.1 Power and Energy models using Power-ArchC for MIPS processor

As a first step to verify the accuracy of Power-ArchC, three power models are built from three ILPC (Instruction Level Power Consumption) campaigns, ILPC1, ILPC2 and ILPC3 to provide comparative analysis. In a first ILPC campaign, 'Motion' benchmark [121] is used. Second ILPC campaign is performed with modified version of 'Qsort' benchmark, which is one of the benchmarks from MiBench [120] and last ILPC (ILPC3) campaign is performed with the average of both previous benchmarks. Motion and Qsort are chosen with small input files due to their simplicity. As an example, the three power models for a subset of MIPS ISA is shown in Figure 5.13. ILPC 2 v/s ILPC 1 has an average relative deviation percentage of 6.7%, ILPC 3 v/s ILPC 1 with 3.3%, and ILPC3 v/s ILPC2 with (−)2.7%. The extremities are with instruction 'jr' which has maximum relative deviation percentage for ILPC 2 v/s ILPC 1 of 32.2% and instruction 'negu' has minimum for ILPC 2 v/s ILPC 1 of (-)0.9%.

Instr Name	ILPC 1 (in µW)	ILPC 2 (in µW)	ILPC 3 (in µW)
addiu	865.9	953.5	909.7
li	866.6	923.2	894.9
jal	783.1	932.5	857.8
sw	872.2	913.3	892.7
move	867.6	903.6	885.6
lw	849	939.5	894.3
nop	856.3	925.4	890.9
mult	828.6	915.5	872.1
mflo	831.2	845	838.1
j	854.8	913.8	884.3
bgez	899.8	936.6	918.2
sll	866.1	882.9	874.5
addu	871.7	936.3	904.0
slt	868.5	687.3	777.9
bnez	869.7	903.9	886.8
jr	695	918.8	806.9
slti	870.3	989.3	929.8
subu	871.9	915.3	893.6
beqz	866.8	909.9	888.4
negu	908.1	899.9	904.0

Figure 5.13: ILPC campaigns of MIPS (power models)

From the three ILPC campaigns, Figure 5.14 shows the energy consumption of 8 MiBench benchmarks whereas Figure 5.15 shows the average power consumption. MiBench benchmarks with six suites, each suite targeting a specific area of the embedded market. The six categories are Automotive and Industrial Control, Consumer Devices, Office Automation, Networking,

Telecommunications. The difference in average power for different benchmarks is due to instruction distribution, memory behavior, and available parallelism. The difference in total energy is mainly due to the complexity and the number of instructions (hence the execution time of a benchmark) and also due to interactions between instructions. This information is useful to designers to consider the effect of these design constraints on power consumption and eventually may result in reducing discharge rate of batteries in portable computing devices. 'Qsort' has the maximum relative deviation percentage for ILPC 2 v/s ILPC 1 of 58.2%, whereas 'gsm' has the minimum of 5.3%. Figure 5.14 and Figure 5.15 show the difference between different campaigns.

Benchmarks	ILPC 1 (in µJ)	ILPC 2 (in µJ)	ILPC 3 (in µJ)
Bitcount	251.8	349.2	339.7
Qsort	80.6	127.6	124.0
Susan	19.4	30.0	29.3
Jpeg	216.9	250.7	244.3
Gsm	223.6	235.6	228.9
Stringsearch	1.8	2.5	2.4
Rijndael	172.1	264.0	257.3
Patricia	1672.3	2549.5	2487.1

Figure 5.14: Total energy in μJ of benchmarks vs. ILPC campaigns

Benchmarks	ILPC 1 (in µW)	ILPC 2 (in µW)	ILPC 3 (in µW)
Bitcount	552.4	765.9	745.2
Qsort	559.6	885.3	860.6
Susan	561	867.8	848.6
Jpeg	736	850.6	829
Gsm	684.7	721.3	700.8
Stringsearch	666.2	905.4	878.9
Rijndael	510.4	783.1	763.2
Patricia	578.2	881.5	859.9

Figure 5.15: Average power in μW of benchmarks vs. ILPC campaigns

Figure 5.16 shows the energy consumption of both Motion and Qsort benchmarks at instruction level and gate level (synthesized MIPS from Design Compiler). For each ILPC campaign, the relative energy error is computed. For Motion benchmark, the energy value in ILPC1 column is equal to gate level value. Actually, ILPC1 at gate level is performed by the execution of this benchmark (with same input data).

Bench-mark	Energy at RTL (in µJ)	Energy at Instruction Level		
		ILPC1 (in µJ)	ILPC2 (in µJ)	ILPC3 (in µJ)
Motion	0.2664	0.2628	0.2827	0.2726
Qsort	0.0438	0.0344	0.0441	0.0426

Figure 5.16: Total energy in μJ of MIPS at instruction level vs. gate level for three ILPC campaigns

However, a relative deviation of 2.31% appears instead of 0. This difference is due to two reasons: one is the difference in linker scripts and second is due to the HMC-MIPS difference with the MIPS-ISA for the 'move' instruction. For ILPC2 and ILPC3, a relative deviation of resp. 1.3% and 6.1% appear. In a similar way, Qsort benchmark may display an identical value in gate-level and ILPC2 columns. As explained above, a difference appears for the same reasons. With ILPC1 and ILPC3 based power models, a relative deviation of 21% and 0.5% respectively, is measured. Additional simulations at instruction-level with different operand values are performed for a better validation.

As an additional feature, Power model also includes the power values of each instruction per pipeline stage. Figure 5.17 shows the total energy of each MIPS pipeline stage for each benchmark.

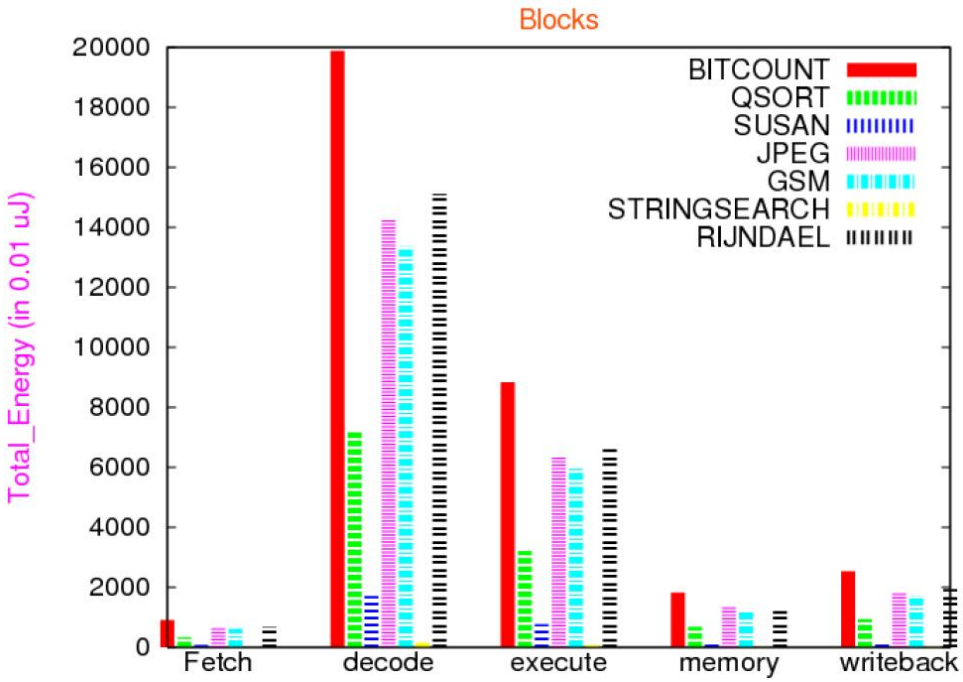

Figure 5.17: A feature of Power-ArchC: Energy consumption of MIPS pipeline at instruction level

It is specific to the architecture, but it can be easily seen which block is consuming more energy and which ones consume less. The reason for the different energy consumption values

in each block is the varied instruction distribution in different benchmarks. For example, 'Decoder' consumes more energy than other stages of pipeline. A logical explanation is that most of the instruction pass by the decoder stage and then processed to move further in the pipeline. Also, Execute block has high energy consumption; actually, the major instruction class from these benchmarks is the computing instructions. This information is very important to a design more efficient architecture and to study the impact of applications on the energy consumption.

5.4.2 Temperature and CFR results using RAAPS for MIPS processor

The variation in temperature in each benchmark is due to difference in type of instructions executed in time. Each benchmark is executed in a loop for approximately 122ms. When loops in benchmarks are created, it is observed that the temperature values from HotSpot approximately follow the same pattern for all iterations. The difference between minimum and maximum of temperature is found while changing some internal parameters in HotSpot, but it follows the same pattern in all cases. These parameters are shown in Figure 5.18. The values shown are for very specific to one configuration and are used in RAAPS. When, the results from HotSpot are analyzed, behavior of steady temperature values of each benchmark is found to be similar to the one of total power. It is understandable as power values are one of the main inputs of HotSpot.

<table>
<tr><th>Basic Specifications for Heat sink</th><th>Chip Specifications</th><th>Heat Spreader Specifications</th><th>Interface Material Specifications</th><th>Miscellaneous</th></tr>
<tr><td>Convection capacitance in J/K - 140.4</td><td rowspan="5">Chip thickness in meters - 0.00015</td><td>Spreader side in meters - 0.03</td><td rowspan="5">Interface material thickness in meters -2.0e-05</td><td>Ambient temperature in Kelvin - 318.15</td></tr>
<tr><td>Convection resistance in K/W - 0.1</td><td rowspan="4">Spreader thickness in meters - 0.001</td><td>Initial temperatures</td></tr>
<tr><td>Heat sink side in meters - 0.06</td><td>Initial temperature (Kelvin) - 333.15</td></tr>
<tr><td rowspan="2">Heat sink thickness in meters - 0.0069</td><td>Hotspot calling interval - 10K cycles at 3GHz</td></tr>
<tr><td>Base processor frequency in Hz - 3e+09</td></tr>
<tr><td></td><td>Model type - block or grid block model</td></tr>
</table>

Figure 5.18: Parameters used in HotSpot when integrated in RAAPS are provided. These parameters can be changed for different operating conditions.

Figures 5.19 and 5.20 are the graphs depicting effect of power on temperature and hence on CFR_{EM}. The graphs are shown with respect to time. Each sample on x-axis corresponds to approximately 27 μsec.

All the CFR values are normalized for clear understanding with the maximum CFR value (normalized to 1) out of all the benchmarks that are being analyzed. This is because of the

Figure 5.19: Temperature profiling for all benchmarks

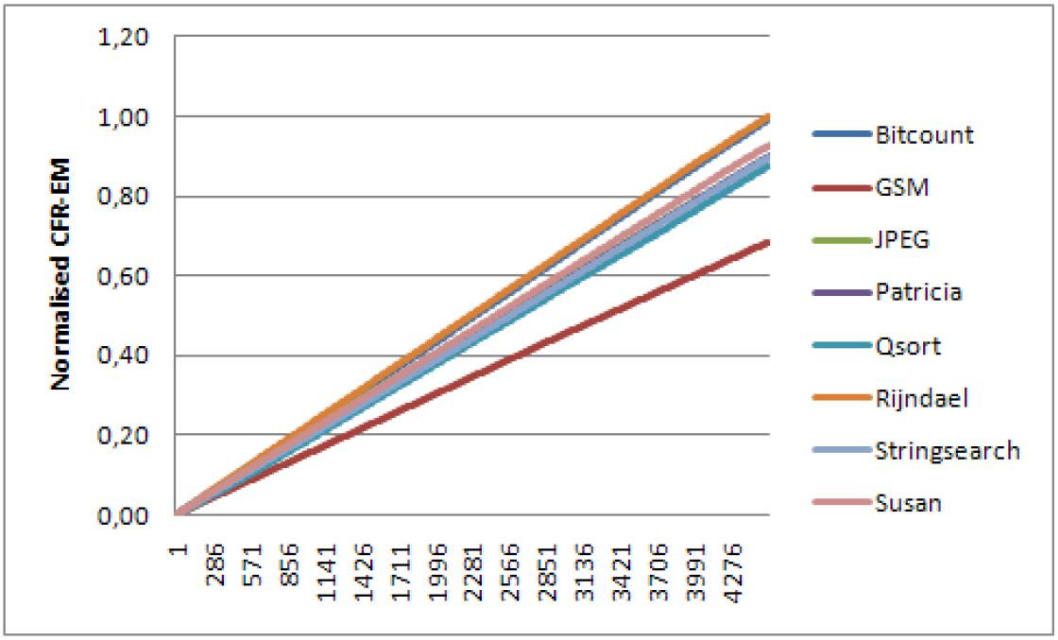

Figure 5.20: Normalized CFR_{EM} for all benchmarks, from 0 to 122ms

lack of knowledge about reliability related parameters from the manufacturer, and hence the constant values are replaced with typical constants found in recent studies [3]. Failure rate involves the variations due to power and temperature both, but largely due to exponential dependency on temperature. CFR_{EM}@122ms tends to increase in linear manner, with slight curve in the beginning (not visible in the graph). The user of such reliability simulator

can analyze CFR due to different benchmarks and can demand a threshold (CFR) from manufacturer according to the purpose of the specific processor. The equal time of simulations are shown for clarity, since Patricia is very long in comparison to other benches, but this does not change the behavior of CFR and will continue in the same manner, if the user simulates a loop of same application. Then, the results of Figure 5.20 can be easily extended to one year or other times. Whatever the time is, 'Rijndael' has the most effect on processor EM compared to 'GSM' due to their instruction distributions, i.e., Rijndael has much high number of Boolean instructions (which has high power consumption) and GSM has high number of Miscellaneous instructions (which has low power consumption in average).

5.4.3 Comparison of CFR results for each benchmark for different failure mechanisms

Figures 5.21, 5.22, 5.23 and 5.24 are showing CFR results for EM, HCI, NBTI and TDDB respectively. Results are given for the whole processor (without considering memories) regarding the 3 different power models (PM1, PM2 and PM3) and the 8 different benchmarks from MiBench benchmark suite. For each failure mechanism, the CFR values are normalized respectively to the maximum CFR obtained from the 8 benchmarks. For each benchmark, parameter 'n' of CFR formula is equal to the number of instructions divided by 100. Parameter 'ti' is a constant value equal to T= 0.27μs. For all failure mechanisms, 'patricia' has much higher CFR than other benchmarks; this is solely due to much higher no. of instructions in 'patricia' than others, as shown in Figure 5.6.

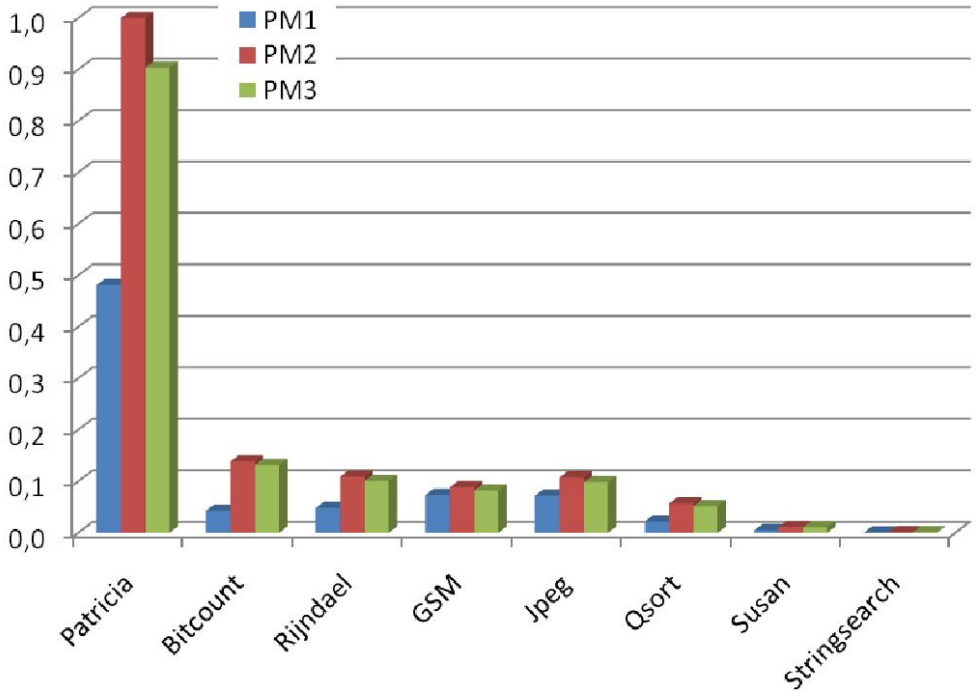

Figure 5.21: CFR EM for MIPS processor

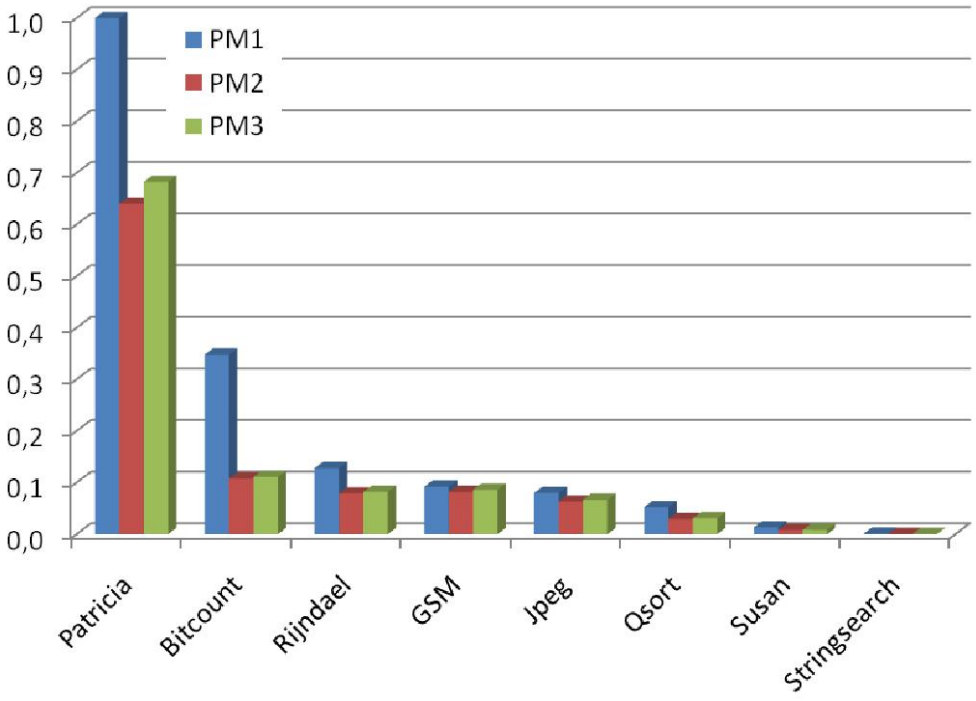

Figure 5.22: CFR HCI for MIPS processor

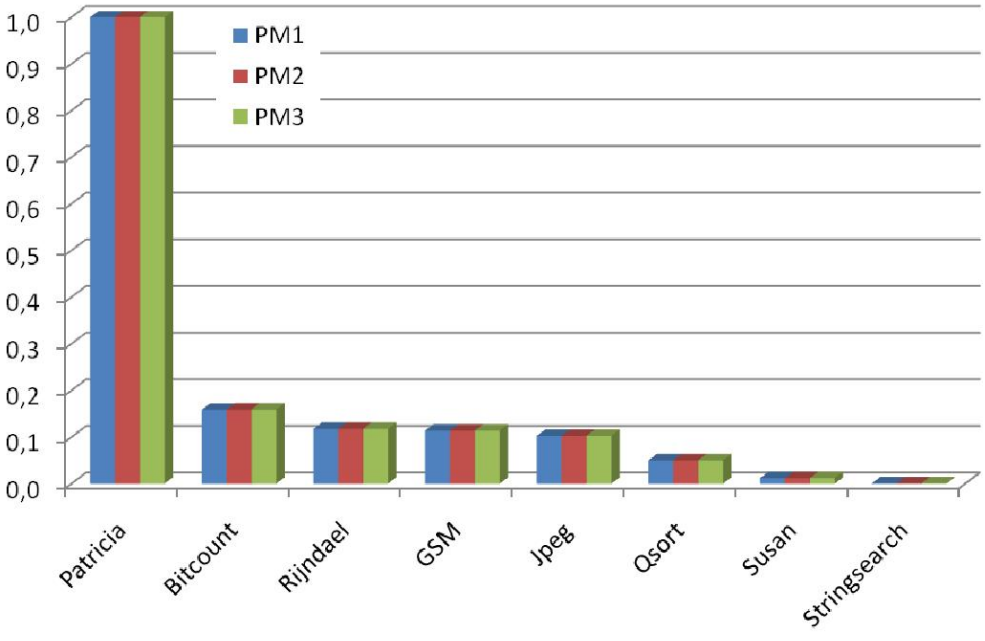

Figure 5.23: CFR NBTI for MIPS processor

Figure 5.21 shows results for EM. We observe effect of power consumption on EM, since the MTTF for EM is inversely proportional to current density; CFR is directly proportional to power consumption. Temperature variations, as already discussed, do not cause an observable

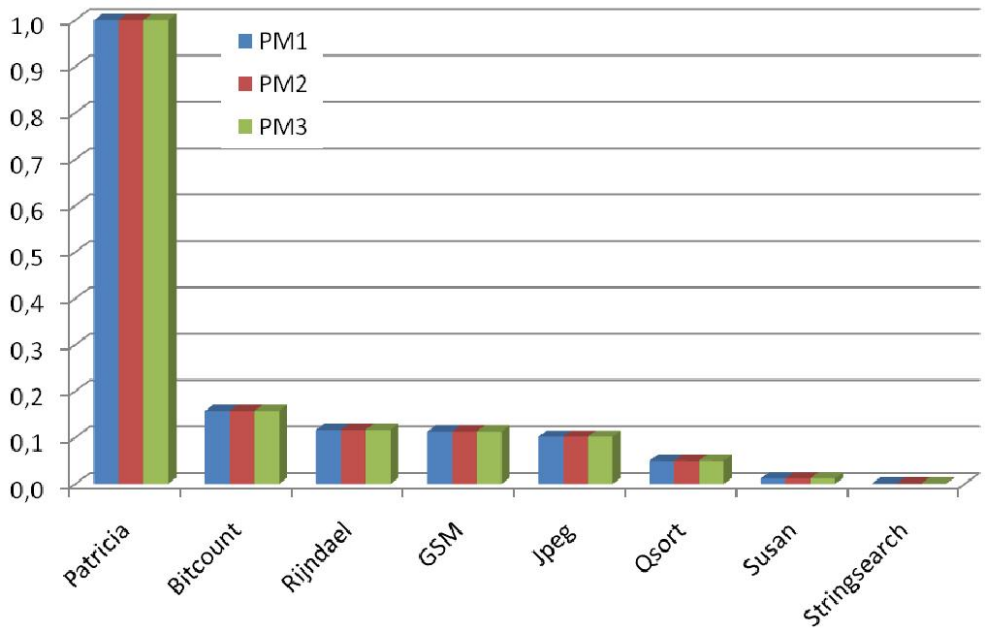

Figure 5.24: CFR TDDB for MIPS processor

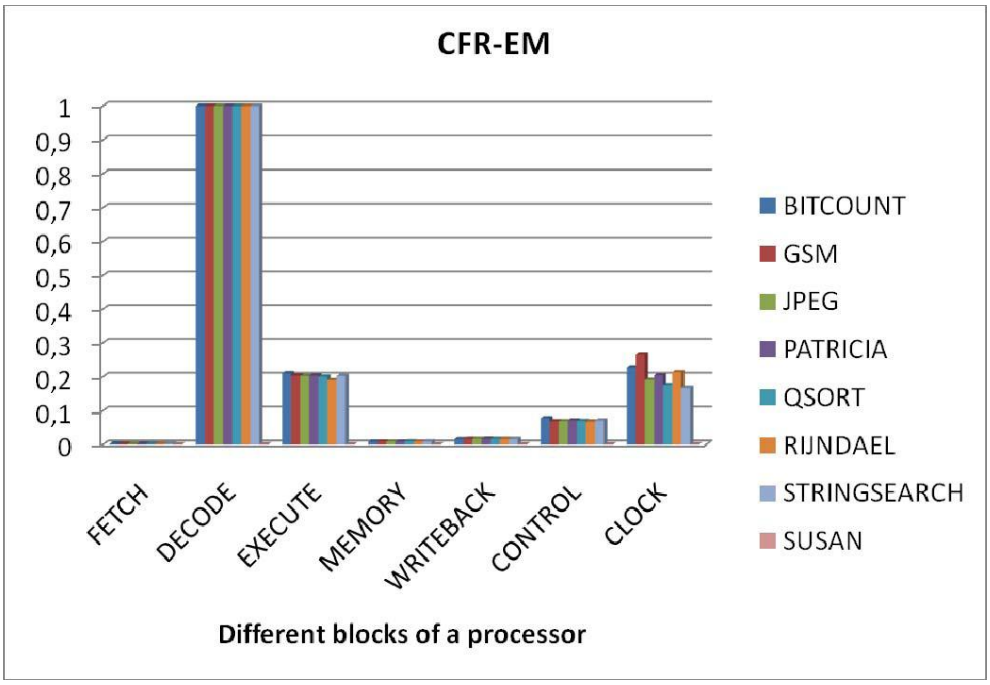

Figure 5.25: CFR EM for different blocks of a processor

effect in this graph. As expected 'Patricia' due to long time of execution and high temperature (46°C in average), put more stress on the processor. When we compare effect of 'GSM' and 'Rijndael' benchmarks (with almost similar no. of instructions and temperature of 45°C) on EM, we observe 'Rijndael' has more effect for PM1 and PM3 because it consumes more power

(i.e., due to the type and executions occurrences of instructions executed).

Figure 5.22 shows CFR results for HCI. As discussed in [78], it is inversely proportional to power consumption and temperature but proportional to time. HCI also has an inverse effect of temperature than all other failure mechanisms. Due to small change in temperature, we are not able to highlight this effect. Similar to discussion for EM, when we compare 'Bitcount', 'Rijndael', 'GSM and 'Jpeg' (with almost similar range of number of instructions), we show that 'Bitcount' for PM1 has much more effect on HCI than others, as power consumption for PM1 of 'Bitcount' is less.

Figure 5.23 and 5.24 show results for NBTI and TDDB respectively. Degradation exhibits dependence on time and temperature. Since there is no effect of power consumption on NBTI, hence for all two power models, we have same results. As expected, TDDB presents the similar behavior as NBTI, since we did not consider all the parameters from the failure models.

As an additional feature, it is also possible to generate CFR results for each block of a processor shown in Figure 5.25, to analyze deeply the effects of power consumption and temperature on CFR, but with a higher speed than at RTL.

5.5 Reliability improvements in MIPS processor using RAAPS

Figure 5.26 provide an overview of various techniques to the designer to use RAAPS, and improve the reliability of his design. In general, a designer could make designs according to the user defined problem and provide it to the manufacturer to be carried out in the form of hardware.

Manufacturer considers different reliability issues then make process refinements as per required lifetime and finally supply the finished product to the user. In Figures 5.26 and 5.27, new steps are added that can help designers to make quick decisions to improve the reliability lifetime of the product. As already explained, CFR can be calculated using RTME based RAAPS tool-chain. After obtaining values of CFR from RTME for various failure mechanisms and different parts of the processor, they can be used by designers and manufacturers separately. As shown in Figure 5.27, using standard designs, manufacturers can decide a threshold value of CFR and check the current design for this threshold.

If the design is not good enough and does not pass the threshold test, manufacturer can take extra precautions for the specific failure mechanism for which the design did not pass the threshold test. On the other hand, since RTME is developed considering higher level of abstraction, the designer has more degree of freedom as he can change the design to adapt for better reliability. As shown in Figure 5.26, using CFR values from RTME, designer can understand which part of the design should be redesigned and which extra features he can include in his design to get better lifetime reliability.

Let us discuss in more details about designer's approach. Using CFR values, designer can use one or more of the following 5 provided options to improve the values of CFR, depending on the budget and time to market.

1. Use Task migration

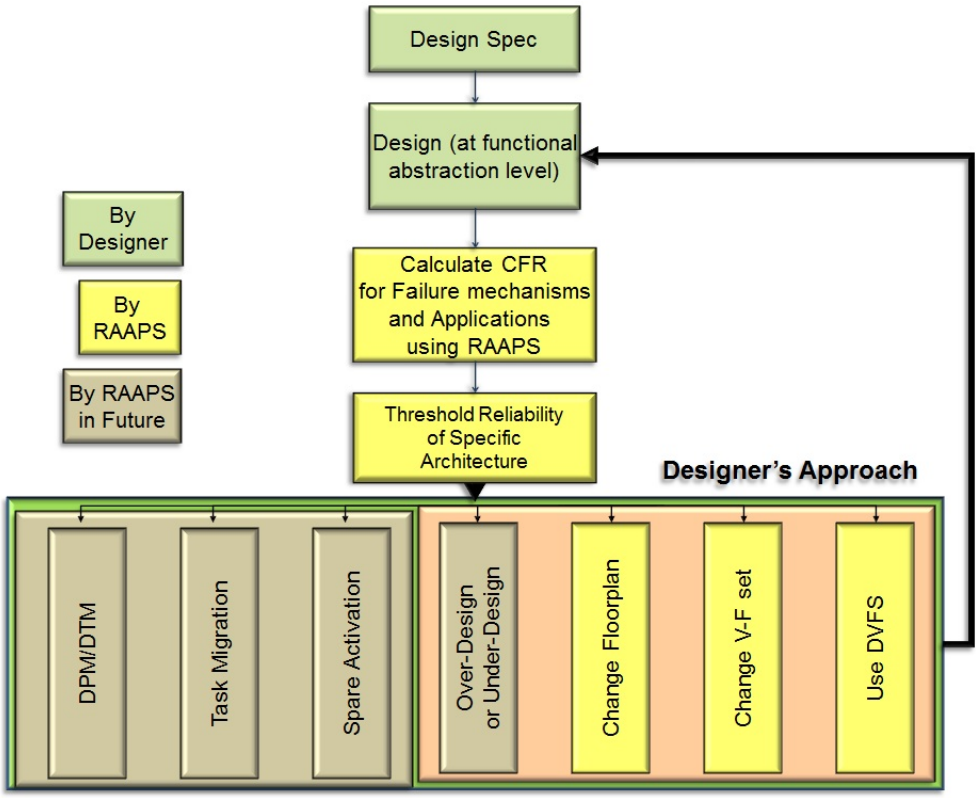

Figure 5.26: How to use the Tool-Chain: Designer's approach

2. Use Over/Under design - such as, use different Voltage-Frequency set
3. Use different Floorplan
4. Use DPM/DTM/DVFS
5. Use Spare activation

The first two of the above techniques will be analyzed using RAAPS Tool-Chain in following subsections and three others are discussed theoretically as short term perspectives in Section 5.6. Here, some particular examples are provided to explain the usage of RAAPS in design flow. A processor may pass a test to verify requirements in specifications under certain conditions.

The scenario is given for a dual-core system that has to run two specific benchmarks Rijndael and GSM at ambient temperature of 318.15°K with required lifetime of each processor to be 34 weeks. The threshold of CFR for Electromigration ($CFR_{EM}(th)$) is fixed at 10 (in arbitrary units).

Task migration

From Figure 5.20, CFR is projected (and continued to extrapolate) in a linear manner for each benchmark to approximately 1 year (365 days), with each sample on x-axis representing 1 week. It is clear from Figure 5.28, that the processor will surely fail the threshold test in

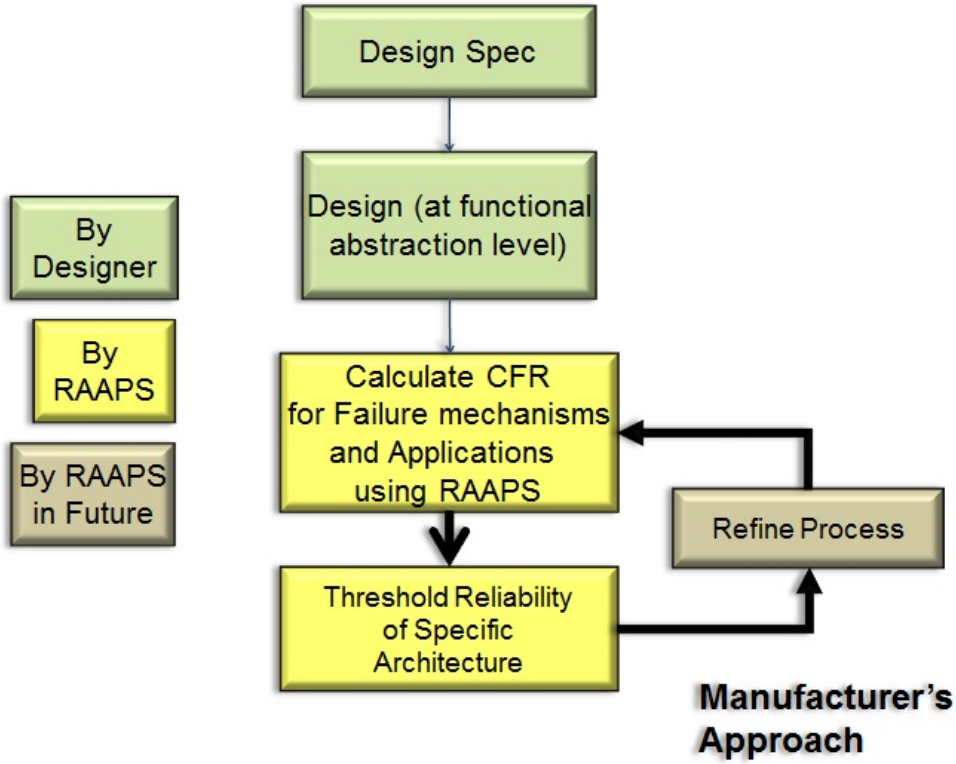

Figure 5.27: How to use the Tool-Chain: Manufacturer's approach

case of Rijndael and Bitcount, and is also on the margin in case of Susan. In case of Qsort, Patricia, Jpeg and GSM, the processor easily pass the threshold criteria and hence is suitable for these applications.

Considering the case from the scenario with Rijndael and GSM to be the applications obliged to work on a processor. One possible condition can be provided as in Figure 5.29.

The solution is to generate a trigger after 20 weeks of execution of Rijndael and to execute GSM for rest of life and move Rijndael to another processor which is probably less aged. This can be used as one trade-off to pass threshold limit.

Use Over/Under design - Use different Voltage-Frequency set

The results presented until now are specific to one set of Voltage-Frequency, i.e., 1.21V-373MHz for high performance. If the designer needs to design the processor reliable for all proposed benchmarks, he may tune the operating conditions such that CFR-EM comes in the limit and hence pass the threshold test. At this moment, RAAPS can be used with 4 different V-F sets as shown in Figure 5.30.

As ILPC4 was obtained in Subsection 5.3.2 for 1.21V-373MHz, it is possible to obtain different power models for different operating conditions. Four power models are obtained for four different VF sets. Power traces are generated using Power-ArchC using different power models and for the specified benchmarks. Then various power values are controlled at task level, i.e., it is only allowed to change V-F set, after finishing a task. Although this process

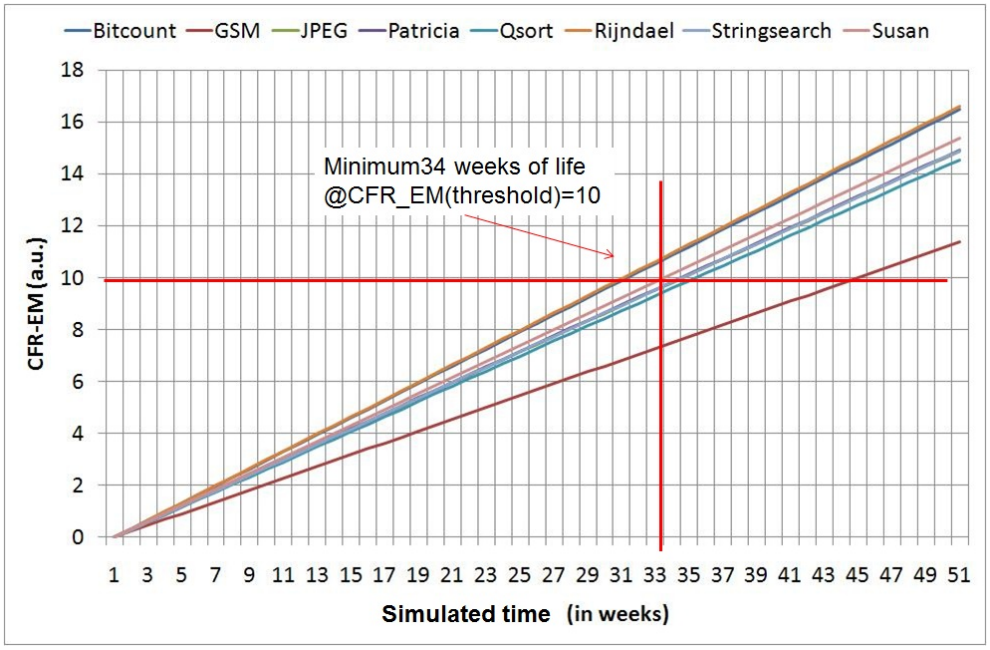

Figure 5.28: Normalized CFR_{EM} for all benchmarks, from 0 to 1 year

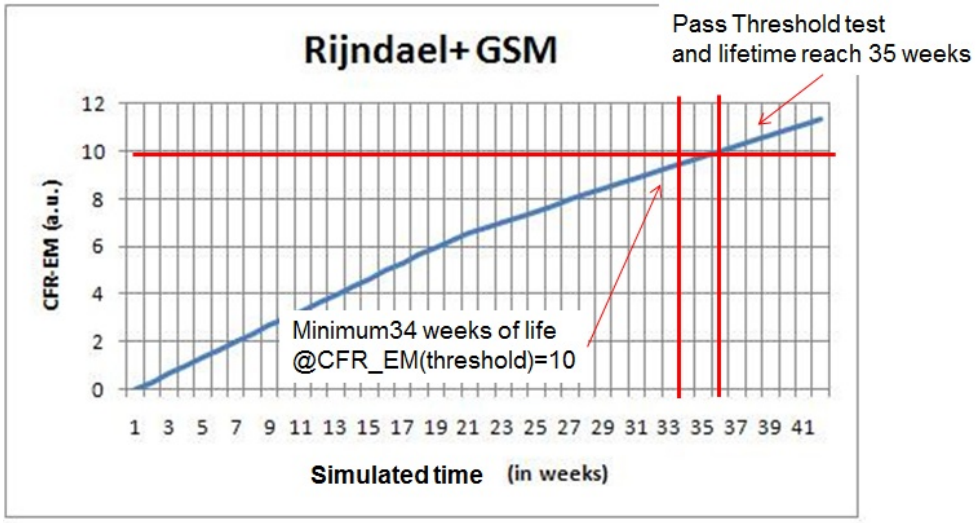

Figure 5.29: Normalized CFR_{EM} for Rijndael + GSM benchmarks, from 0 to 1 year

will also affect the performance, it is still a good trade-off. Figure 5.31 depict the effect of two different V-F set on CFR. The lifetime of the process improves significantly by using different V-F set, while making trade-off with the performance.

Mode	Operating Voltage (in V)	Operating Frequency (in MHz)
Low performance	0.81	54
Slow	0.99	142
Half speed	1.1	267
High performance	1.21	373

Figure 5.30: Different V-F sets

5.6 Conclusion

In this chapter, the methodology presented in Chapter 4 is first used to explore power and energy consumption for an ISS generated by ArchC. The originality of this work (Power-ArchC) is that it is the first work that incorporates power capabilities in the ArchC framework and so it can provide immediate power estimation in MPSoC with the execution of a benchmark. In the second part, the RAAPS tool-chain is used to predict accurate reliability values of a processor at functional level. In previous work, RTME was used to estimate processor reliability at RTL abstraction, using Wattch as a power estimator of a generic RISC processor [78]. Originality in RAAPS is that it incorporates reliability capabilities in the ArchC framework and so it can be easily incorporated in an MPSoC simulator (SystemC language) and will provide similar analysis with the execution of a parallel application, using for e.g. different floorplans and scheduling policies. In this Chapter, the effect of different power scenarios on the reliability of a MIPS processor is given. It has been shown that power consumption and the type and number of occurrences of executed instructions have a big effect on MIPS EM and HCI. Here, temperature has not much effect due to small size of benchmarks. While calculating percentage error for the results obtained using RAAPS, it is assumed that HotSpot has +/- 6% error. The error for RAAPS is between 8% and 34% for all failure mechanisms. Finally, as an additional feature, reliability results are shown for different blocks of MIPS and hence shown that 'Decoder' is much more prone to failures than other blocks. Hence designer should take extra precautions to improve the reliability of decoder block.

In the Section 5.5, the analyses with RAAPS are continued, providing some details regarding the failure model used for Electromigration at system level. RAAPS can be used by designer to improve his design's overall reliability, using various techniques provided such as task migration, changing floorplan, changing operating conditions using DVFS. Designer can find a good trade-off between reliability, performance and the cost of the whole product very fast and very early in the design flow.

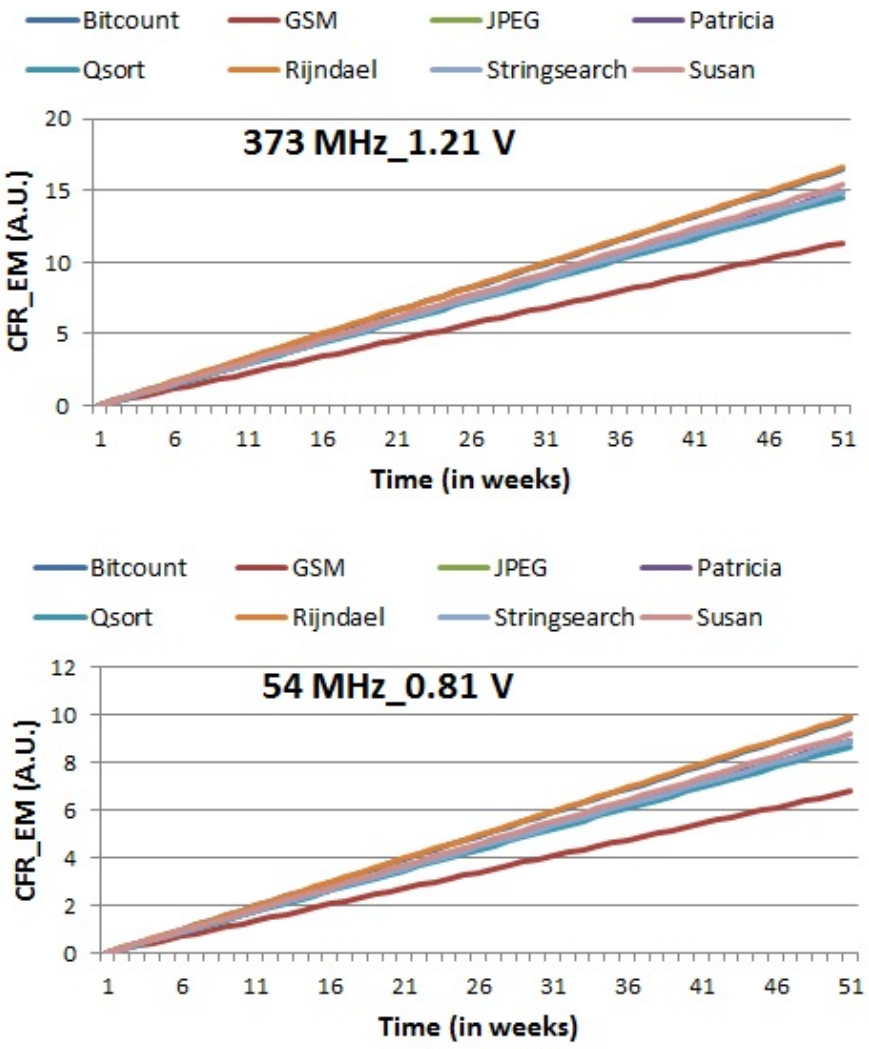

Figure 5.31: Effect of changing operating conditions on the CFR EM for different benchmarks with respect to time

Conclusions and perspectives

Contents

Hard intrinsic errors are still important because of die shrinking, have been widely investigated for one single transistor, it is still new for researchers to predict these errors for a complete chip which is composed of millions of transistors. The important point is to find the interrelations between transistors and also interrelations between different blocks, the chip is divided into. In the present thesis report, the studies have been started at transistor level of abstraction and move on to higher level of abstraction to predict reliability of millions of transistors on an average. To this aim, a new tool named RAAPS is developed. The results show that RAAPS can help to decide if the design is reliable enough to be synthesized or to go to design cycles at other level of abstraction.

Summary of contributions

The first chapter of this thesis report provided a brief overview of the MOS transistor and its working. Then various types of errors that can occur in a transistor and already known to researchers were discussed. As most of the extrinsic errors are removed during burn-in process before shipping the chips, the focus was mainly on intrinsic errors. The physics behind different hard intrinsic errors was given to explain why there are errors during and after the useful lifetime of a semiconductor device. Also, variability was linked to reliability, it was explained that the feedback of variability results should be provided to improve the reliability. Finally, the problem of reliability in the emerging technology 3D-IC was discussed.

Chapter 1 was used as the base for Chapter 2 to study mathematics of reliability and failure models at transistor level. All models were based on the physics provided in Chapter 1. Chapter 2 focuses on failure modeling first at transistor level of abstraction and then it provided a method to build functional level reliability models using transistor level models.

The accuracy of these models was estimated by a qualitative point of view, using the assumptions that help creating transition from transistor level to functional level. Researchers showed in the past that all transistors age in a same manner, hence we can extrapolate the transistor level models to higher level of abstraction. It was not accurate but was close to reality, as the stress on each transistor can be different during useful life but in all cases, it gets stressed due to dynamic or static consumption. Chapter 2 ended with derivations for failure rate (λ) and hence cumulative failure rate (CFR) for EM, HCI, NBTI and TDDB type failure mechanism at functional level of abstraction. From these derivations, it can be said that following parameters can provide a good reliability prediction in useful life of the device: Probability of number of transistors switching at a given instant, Current Density, Substrate Current, Temperature, Area or number of transistors.

The modeling in Chapter 2 at transistor level and then at functional level was useful to research and reliability prediction methodologies discussed in Chapter 3. In Chapter 3, different methodologies were presented that use different modeling ways in past 25 years. Existing tools work at transistor and gate level of abstraction as prediction of reliability needs high accuracy. High level simulations mainly focus on complex architectures with millions of transistors. Hence, the prediction of reliability could not be very accurate due to complexity involved. At lower-level of abstraction, with information about placement and routing of the circuits, it was possible to compute detailed timing and performance characteristics to optimize the design accordingly. It was a challenging problem to design a performance efficient yet accurate lifetime reliability simulator for complete system. In authors knowledge there were no work in the literature in this domain except [54] who are also working on similar methodology at the same time as presented in Chapter 4, with the big differences are the embedding of the methodology in design flow and the parameters fitting with close to real values in given methodology. In the end, a quick comparison between all the simulators and methodologies was provided at various abstraction levels. It was also shown in Chapter 3 that no tool exists for reliability prediction accurate enough that can provide high speed and exploration capabilities for multi-core architecture at functional level of simulation, which can be embedded in design flow. In Chapter 4, a new methodology was introduced that takes many of the good points from existing methodologies and especially close to 'RAMP' and 'AgeSim' simulation frameworks.

The discussion in Chapter 3 about already existing reliability simulators and methodologies lead to a new reliability simulator called RTME explained in Chapter 4. It was developed to help designers and manufacturers by calculating Cumulative Failure Rate (CFR) parameter which was defined and explained in Chapter 2. RTME can calculate CFR for different blocks of the processor or for complete processor, according to the details provided in the floorplan and for different applications that can run on a processor separately or together. As a case study, a tool-chain was presented that was built around RTME at functional level of abstraction called RAAPS (Reliability Aware ArchC based Processor Simulator). In complement to RAAPS, a new power simulator called Power-ArchC (based on ArchC- Architecture description language) was also presented, to simulate power consumption at functional level of abstraction. RAAPS has certain advantages in respect to other existing tools. One most important advantage was the integration of reliability simulator in a design flow.

Chapter 5 takes first step towards validation of RAAPS tool-chain. It also verified the accuracy of Power-ArchC. Power-ArchC was the first work that incorporated power capabilities in the ArchC framework and so it can provide immediate power estimation in MPSoC with the execution of a benchmark. The limitations of this work were: 1.) To reduce the

time of obtaining power model(s), only a subset of all the possible operands and instructions interrelations were taken into account to characterize power consumption, and 2.) There was no tool at present, able to translate an ArchC description to RTL. The RAAPS tool-chain was used to predict accurate reliability values of a processor at functional level. The effect of different power scenarios on the reliability of a MIPS processor was shown for TSMC 40nm technology library. Also, it has been shown that power consumption and the type and number of occurrences of executed instructions had a big effect on MIPS EM and HCI. Here, temperature did not have much effect due to small size of benchmarks. The percentage error in calculation of CFR_{EM} is approximately 31% to 34%, CFR_{HCI} is approximately 8% to 10%, CFR_{NBTI} is approximately 11.8% to 12.7% and CFR_{TDDB} is approximately 21.8% to 23%. The accuracy is depending on many factors such as power and temperature simulators attached to RTME, the run length of a given benchmark and so on. Hence, the accuracy results provided may vary in other conditions.

RTME and hence RAAPS were developed so that they can be used by the designer to improve his design's overall reliability, using various measures provided such as different floorplan or V-F set. Designer can find a good trade-off between reliability, performance and the cost of the whole product very fast and very early in the design flow.

Perspectives

Although, a lot of work is performed and presented in this thesis report, there is still lot of work left in the methodology presented using RTME and RAAPS. Most of the development time was used to develop power simulator at functional level of abstraction named PowerArchC and in integration of RAAPS in design flow. Rest of the duration was used to analyze and explore the results of reliability obtained using RAAPS. In the following subsections, some of the potential future perspectives are provided. These perspectives see the future of this methodology.

Short term perspectives

The short term perspectives are the future work which could have been done in the next few months of work and analysis. The complete report is focused on 8 MiBench benchmarks with different instruction configurations. The use of other benchmark suites must provide more generality to validation of the Tool-chain. The main effort in analyzing other benchmark suites is to obtain the source code and make them executable on given ArchC based MIPS processor. The limitation of this methodology is to reduce the time of obtaining power model using ILPC, only a subset of all the possible operands and instructions interrelations were taken into account to characterize power consumption. The second limitation is that there is no tool at present, able to translate an ArchC description to RTL. The first limitation could be overcome by making more simulations, although it may take year(s) to simulate all the possible operands for each instruction type. The second limitation is more complicated, since it is a manual step to generate RTL and functional level design of a given processor. Although, there seems to be an update from ArchC that may be able to reduce the designer's work to write RTL code for the processor he/she has designed using ArchC at functional level [122].

To improve the design, designer can use various techniques provided in [123] and are also discussed in Section 5.5. Some of these techniques are discussed here (which are also in close relation to measures given by [7]) and can be used to improve the values of CFR, depending on the budget and time to market.

1. Use DPM/DTM/DVFS: Research [124] in Dynamic Power Management (DPM) and Dynamic Thermal Management (DTM) has attracted different industrial enterprise. The DPM controller dynamically activates idle mode or turns off the computing device when it is idle or under-utilized to reduce the system power consumption. By the DTM controller also, similar actions are taken when the die is (or is predicted to be) over heated. Reducing energy consumption to the required levels ensures correct and useful operation of the integrated systems. DPM also affects the reliability of the system components. Curbing power dissipation helps lowering the device temperatures and reducing the effect of temperature-driven failure mechanisms, thus making components more reliable. In cases, where the usage of the processor is very complex, the designer has the option to use DVFS to achieve good energy saving and provide improved reliability by controlling operating voltage and frequency that can help controlling power consumption. But it is a very expensive option for the user [125, 126].

2. Spare activation: Two units that run hot by themselves will tend to run even hotter when adjacent [80]. Separating them will introduce additional communication latency that is incurred regardless of operating temperature. This suggests the use of spare units located in cold areas of the chip, to which computation can migrate only when the primary units overheat and hence may fail in the future.

3. Decision about Floorplan: The temperature values and hence the reliability also depend on the area, dimension and location of the specified block, it implies that the reliability gets affected by the floorplan of the chip. By making changes in the floorplan and analyzing the changes in temperature, reliability of the chip can be enhanced.

These techniques could be used by designer to improve the reliability of his design. Of course, to implement each of the above steps in a design will increase the cost of the whole chip and hence the designer will have to make trade-offs between one or more steps. An EDA tool can be made to provide the designer with functionality to switch on or off, one or more of steps, to find the good trade-off according to his need.

Average term perspectives

For average term perspectives, these are the points that could be taken care of, in next year or two and, which includes some intense research before proceeding for development. In the following Subsections, different modes of RAAPS are discussed, which can improve the performance of design or accuracy of the reliability results. To satisfy the needs of various users, it is necessary to have a flexible methodology. In following subsection, two steps are discussed to improve the accuracy of RAAPS. These steps can be used to increase and decrease the speed and accuracy respectively (or vice-versa) of RAAPS.

RAAPS considering variability and static power consumption values

There is a strong relationship between power consumption (both dynamic and static), temperature, environmental conditions (humidity, and ambient temperature), process variations and operating conditions (operating supply voltage and frequency) causing failures (permanent faults) in an integrated chip. Relatively to power and temperature at block level, EM results depend on power consumption and temperature variations. At present, the simulations in RAAPS are assumed to be under consideration of ideal environment, with no humidity and no variability and results for the whole processor are considered without system memories. In the future version of RAAPS, variability will also be introduced to increase the accuracy in predicting reliability of the device. Variability is a field that is subjected to extensive research. It may need few months of research to completely understand, what variability is and which parameters exactly affect variability. Then the focus should be on developing a module to include variability. Finally, this module would be integrated in RAAPS tool-chain and may be output of CFR processed through variability module and fed back to Power-ArchC, due to which power consumption values will be modified from the values in power models generated by ILPC. This will again directly impact values of CFR. Like dynamic power, the static power values are obtained using PrimeTime and ILPC use the values to generate power model for static power also. But the static power consumption can be assumed to be constant only in an ideal world with no variability. PrimeTime provides information about static power consumption, in the current version of RAAPS only dynamic power consumption is considered. This is very important in future technologies. If static power is considered, it will improve the accuracy for predicting lifetime. Effect of static power on reliability, this study should be performed after obtaining the manufacturer related values via industrial partner (Integrated Device Manufacturer - IDM) as explained in Section 5.3.

Long term perspectives

It is a difficult task for the designer to propose a general algorithm or failure model that can predict reliability of any given circuit for any given technology as there are parameters that should be considered that may not change with time but changes for different technologies. A long term perspective of RTME and hence RAAPS like tool-chain could be developing it in more general form and not dependent on present day's existing failure mechanisms only. For example, in few years from now, by studying the industrial data (from IDM or foundry) regarding the occurrence of failures in a chip, designer should be able to create or obtain a general failure library with all the technology and manufacturer dependent data. From this library, designer should obtain/provide all reliability related required inputs in 'New-RTME'. 'New-RTME' should be able to predict the lifetime of any design using these inputs. It is very complicated to develop this proposed futuristic tool, 'New-RTME', at this moment due to lack of information flow between designer and manufacturer. But with the help of future guidelines from ITRS regarding technological requirements, and partnerships between IDMs, foundries and fabless enterprises to share the reliability related data, a researcher in the future should be able to find a suitable algorithm that can make this tool a reality.

Glossary

AF ..... *Acceleration Factor*
AFT ..... *Temperature Acceleration Factor*
AFV ..... *Voltage Acceleration Factor*
BERT ..... *Berkeley Reliability Tools*
BICI ..... *Boundary- and Initial-Condition Independent*
CAD ..... *Computer Aided Design*
CCS ..... *Constant Current Stress*
CDF ..... *Cumulative Distribution Function*
CFR ..... *Cumulative Failure Rate*
CMOS ..... *Complementary Metal Oxide Semiconductor*
CORS ..... *Circuit Oxide Reliability Simulator*
CVS ..... *Constant Voltage Stress*
D2D ..... *Die-to-Die*
DFR ..... *Design-for-Reliability*
DPM ..... *Dynamic Power Management*
DTM ..... *Dynamic Thermal Management*
DVFS ..... *Dynamic Voltage Frequency Scaling*
Ea ..... *Activation Energy*
EDA ..... *Electronic Design Automation*
EM ..... *ElectroMigration*
FaRBS ..... *Failure-Rate-Based Simulator*
FIT ..... *Failure-in-Time*
HCI ..... *Hot Carrier Injection*
IC ..... *Integrated Circuit*
IDM ..... *Integrated Device Manufacturer*
ILLIADS ..... *Illinois Analogous Digital Simulator*
ILPC ..... *Instruction level power consumption*
ISS ..... *Instruction Set Simulator*
MaCRO ..... *Maryland Circuit-Reliability Oriented*
MPSoC ..... *Multi-Processor System-On-Chip*
MTTF ..... *Mean Time To Failure*

NBTI ..... *Negative Bias Temperature Instability*
PC ..... *Program Counter*
PDF ..... *Probability Density Function*
PVT ..... *Process-Voltage-Temperature*
RAAPS ..... *Reliability Aware ArchC based Processor Simulator*
RAMP ..... *Reliability Aware MicroProcessor*
RF ..... *Reliability Function*
RTL ..... *Register Transfer Level*
RTME ..... *Real Time MTTF Evaluation*
SEE ..... *Soft Error Effect*
SEU ..... *Single Event Upset*
SOFR ..... *Sum-of-Failure-Rates*
TDDB ..... *Time Dependent Dielectric Breakdown*
TLM ..... *Transactional Level Modeling*
WID ..... *Within-Die*

Bibliography

[1] Itrs. 2009 international technology roadmap for semiconductors. [online] itrs. [cited: 01/28, 2010.] http://www.itrs.net/links/2009itrs/home2009.htm.

[2] Mitsubishi high power semiconductors, semiconductor device reliability, Aug 1998.

[3] Failure mechanisms and models for semiconductor devices. Technical report, JEDEC Publication, March, 2009.

[4] A. Allan, D. Edenfeld, Jr. Joyner, W. H., A. B. Kahng, M. Rodgers, and Y. Zorian. 2001 technology roadmap for semiconductors. *Computer*, 35(1):42–53, 2002.

[5] Itrs 2007: Process integration, devices and structures.

[6] Kuang Wei. Cmos ic design for reliability - a review. 20 November 2009.

[7] Ayse K. Coskun, Tajana Simunic Rosing, Yusuf Leblebici, and Giovanni De Micheli. A simulation methodology for reliability analysis in multi-core socs. In *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, GLSVLSI '06, pages 95–99, New York, NY, USA, 2006. ACM.

[8] Open systemc initiative:.

[9] T. Gupta, C. Bertolini, O. Heron, N. Ventroux, T. Zimmer, and F. Marc. Raaps: Reliability aware archc based processor simulator. In *Proc. IEEE Int. Integrated Reliability Workshop Final Report (IRW)*, pages 153–156, 2010.

[10] P. Bose J. Rivers J. Srinivasan, S. Adve and C. Hu. Ramp: A model for reliability aware microprocessor design. Technical report, IBM Research Report, 2003.

[11] R. Baumann. Soft errors in advanced computer systems. *IEEE Design & Test of Computers*, 22(3):258–266, 2005.

[12] N. Bidokhti. Seu concept to reality (allocation, prediction, mitigation). In *Proc. - Annual Reliability and Maintainability Symp. (RAMS)*, pages 1–5, 2010.

[13] Subhasish Mitra, Pia Sanda, and Norbert Seifert. Soft errors: Technology trends, system effects, and protection techniques. In *Proc. 13th IEEE Int. On-Line Testing Symp. IOLTS 07*, 2007.

[14] Austin Lesea and Peter Alfke. Xilinx fpgas overcome the side effects of sub-90 nm technology. Technical report, Xilinx, Inc., March 2007.

[15] X. Li, S. V. Adve, Pradip Bose, and J. A. Rivers. Softarch: an architecture-level tool for modeling and analyzing soft errors. In *Proc. Int. Conf. Dependable Systems and Networks DSN 2005*, pages 496–505, 2005.

[16] Failure mechanisms and models for semiconductor devices. Technical report, JEDEC Solid State Technology Association, 2003.

[17] D. Young and A. Christou. Failure mechanism models for electromigration. 43(2):186–192, 1994.

[18] White Mark & Joseph B. Bernstein. Microelectronics reliability: Physics-of-failure based modeling and lifetime evaluation. Technical report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, JPL publication., 2008.

[19] Yuan Taur and Tak H. Ning. *Fundamentals of modern VLSI devices.* Cambridge University Press, New York, NY, USA, 1998.

[20] T.-C. Ong, M. Levi, P.-K. Ko, and C. Hu. Recovery of threshold voltage after hot-carrier stressing. 35(7):978–984, 1988.

[21] Transistor counts on wikipedia [online].

[22] E. Beyne. The rise of the 3rd dimension for system intergration. In *Proc. Int. Interconnect Technology Conf*, pages 1–5, 2006.

[23] S. S. Sapatnekar. Addressing thermal and power delivery bottlenecks in 3d circuits. In *Proc. Asia and South Pacific Design Automation Conf. ASP-DAC 2009*, pages 423–428, 2009.

[24] H. Oprins, V. Cherman, C. Torregiani, M. Stucchi, B. Vandevelde, and E. Beyne. Thermal test vehicle for the validation of thermal modelling of hot spot dissipation in 3d stacked ics. In *Proc. 3rd Electronic System-Integration Technology Conf. (ESTC)*, pages 1–6, 2010.

[25] Rev Aug. Semiconductor reliability handbook, 2006.

[26] *Reliability Stress and Failure Rate Data for Electronic Equipment, Mil-HDBK 217B.* 20 September 1974.

[27] Mil-hdbk-338b, electronic design reliability handbook. Technical report, Department of Defense of USA, 1988.

[28] D. G. Pierce and P. G. Brusius. Electromigration: A review. *Microelectronics and Reliability*, 37(7):1053 – 1072, 1997. Reliability Physics of Advanced Electron Devices.

[29] James R. Black. Mass transport of aluminum by momentum exchange with conducting electrons. In *Proc. Sixth Annual Reliability Physics Symp*, pages 148–159, 1967.

[30] J. R. Lloyd. New models for interconnect failure in advanced ic technology. In *Proc. 15th Int. Symp. the Physical and Failure Analysis of Integrated Circuits IPFA 2008*, pages 1–7, 2008.

[31] A. Hamada and E. Takeda. Ac hot-carrier effect under mechanical stress [mosfet]. In *Proc. Digest of Technical Papers VLSI Technology 1992 Symp*, pages 98–99, 1992.

[32] E. Takeda, K. Ikuzaki, H. Katto, Y. Ohji, K. Hinode, A. Hamada, T. Sakuta, T. Funabiki, and T. Sasaki. Vlsi reliability challenges: from device physics to wafer scale systems. 81(5):653–674, 1993.

[33] E. Takeda, R. Izawa, K. Umeda, and R. Nagai. Ac hot-carrier effects in scaled mos devices. In *Proc. Int Reliability Physics Symp., 29th Annual*, pages 118–122, 1991.

[34] E. Takeda and N. Suzuki. An empirical model for device degradation due to hot-carrier injection. 4(4):111–113, 1983.

[35] Chenming Hu, Simon C. Tam, Fu-Chieh Hsu, Ping-Keung Ko, Tung-Yi Chan, and K. W. Terrill. Hot-electron-induced mosfet degradation – model, monitor, and improvement. 20(1):295–305, 1985.

[36] A. Hamada and E. Takeda. Hot-electron trapping activation energy in pmosfet's under mechanical stress. 15(1):31–32, 1994.

[37] A. Hamada and E. Takeda. Hot-electron trapping activation energy under mechanical stress. In *Proc. Digest of Technical Papers VLSI Technology 1993 Symp*, pages 15–16, 1993.

[38] J. W. McPherson and H. C. Mogul. Underlying physics of the thermochemical e model in describing low-field time-dependent dielectric breakdown in sio2 thin films. *Journal of Applied Physics*, 84(3):1513–1523, 1998.

[39] J. H. Stathis. Physical and predictive models of ultrathin oxide reliability in cmos devices and circuits. 1(1):43–59, 2001.

[40] N. K. Jha, P. S. Reddy, D. K. Sharma, and V. R. Rao. Nbti degradation and its impact for analog circuit reliability. 52(12):2609–2615, 2005.

[41] Kjell O. Jeppson and Christer M. Svensson. Negative bias stress of mos devices at high electric fields and degradation of mnos devices. *Journal of Applied Physics*, 48(5):2004–2014, 1977.

[42] S. Chakravarthi, A. Krishnan, V. Reddy, C. F. Machala, and S. Krishnan. A comprehensive framework for predictive modeling of negative bias temperature instability. In *Proc. IEEE Int. 42nd Annual Reliability Physics Symp*, pages 273–282, 2004.

[43] G. Haller, M. Knoll, D. Braunig, F. Wulf, and W. R. Fahrner. Bias-temperature stress on metal-oxide-semiconductor structures as compared to ionizing irradiation and tunnel injection. *Journal of Applied Physics*, 56(6):1844–1850, 1984.

[44] Joseph B. Bernstein, Moshe Gurfinkel, Xiaojun Li, Jörg Walters, Yoram Shapira, and Michael Talmor. Electronic circuit reliability modeling. *Microelectronics and Reliability*, 46(12):1957 – 1979, 2006.

[45] Xiaojun Li, B. Huang, J. Qin, X. Zhang, M. Talmor, Z. Gur, and J. B. Bernstein. Deep submicron cmos integrated circuit reliability simulation with spice. In *Proc. Sixth Int. Symp. Quality of Electronic Design ISQED 2005*, pages 382–389, 2005.

[46] Xiaojun Li, Jin Qin, Bing Huang, Xiaohu Zhang, and J. B. Bernstein. Sram circuit-failure modeling and reliability simulation with spice. 6(2):235–246, 2006.

[47] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. The case for lifetime reliability-aware microprocessors. In *Proc. 31st Annual Int Computer Architecture Symp*, pages 276–287, 2004.

[48] Luciano Lavagno, Grant Martin, and Louis Scheffer. *Electronic Design Automation for Integrated Circuits Handbook - 2 Volume Set*. CRC Press, Inc., Boca Raton, FL, USA, 2006.

[49] Gordon Yip. Expanding the synopsys primetime solution with power analysis. Technical report, Synopsys, Inc., June 2006.

[50] S. Ghosh and K. Roy. Parameter variation tolerance and error resiliency: New design paradigm for the nanoscale era. 98(10):1718–1751, 2010.

[51] K. Roy and S. Prasad. Logic synthesis for reliability: an early start to controlling electromigration and hot-carrier effects. 44(2):251–255, 1995.

[52] G. Ribes, S. Bruyere, F. Monsieur, D. Roy, and V. Huard. New insights into the change of voltage acceleration and temperature activation of oxide breakdown. *Microelectronics Reliability*, 43(8):1211 – 1214, 2003.

[53] D. Lorenz, G. Georgakos, and U. Schlichtmann. Aging analysis of circuit timing considering nbti and hci. In *Proc. 15th IEEE Int. On-Line Testing Symp. IOLTS 2009*, pages 3–8, 2009.

[54] Lin Huang and Qiang Xu. Agesim: A simulation framework for evaluating the lifetime reliability of processor-based socs. In *Proc. Design, Automation & Test in Europe Conf. & Exhibition (DATE)*, pages 51–56, 2010.

[55] Ayse K. Coskun, Richard Strong, Dean M. Tullsen, and Tajana Simunic Rosing. Evaluating the impact of job scheduling and power management on processor lifetime for chip multiprocessors. In *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*, SIGMETRICS '09, pages 169–180, New York, NY, USA, 2009. ACM.

[56] P. Bose J. A. Rivers. Poster to appear at the IBM Austin Conference on Energy-Efficient Design (ACEED) J. Srinivasan, S.V. Adve. Lifetime reliability awareness for microprocessors. Mar. 2005.

[57] D. Brooks, R. P. Dick, R. Joseph, and Li Shang. Power, thermal, and reliability modeling in nanometer-scale microprocessors. 27(3):49–62, 2007.

[58] Zhihong Liu, Bruce W. McGaughy, and James Z. Ma. Design tools for reliability analysis. In *Proceedings of the 43rd annual Design Automation Conference*, DAC '06, pages 182–187, New York, NY, USA, 2006. ACM.

[59] Horst Rempp, Oliver Thalau, Andrea Scorzoni, Gerard Ghilbaudo, Emanuel Vincent, and Sean Minehane. Experiences on reliability simulation in the framework of the prophecy project. *Microelectronics and Reliability*, 39(5):661 – 672, 1999.

[60] R.H. Tu, E. Rosenbaum, C.C. Li, W.Y. Chan, P.M. Lee, B-K. Liew, J.D. Burnett, P.K. Ko, and Chenming Hu. Bert - berkeley reliability tools. Technical Report UCB/ERL M91/107, EECS Department, University of California, Berkeley, 1991.

[61] R. H. Tu, E. Rosenbaum, W. Y. Chan, C. C. Li, E. Minami, K. Quader, P. K. Ko, and C. Hu. Berkeley reliability tools-bert. 12(10):1524–1534, 1993.

[62] M. Ruberto, T. Maimon, Y. Shemesh, A. B. Desormeaux, Weiquan Zhang, and Chune-Sin Yeh. Consideration of age degradation in the rf performance of cmos radio chips for high volume manufacturing. In *Proc. Digest of Papers Radio Frequency integrated Circuits (RFIC) Symp. 2005 IEEE*, pages 549–552, 2005.

[63] Reliability simulation in integrated circuit design a whitepaper.

[64] The solution for circuit-level, hot-carrier / nbti simulation.

[65] P. M. Lee, M. M. Kuo, K. Seki, P. K. Lo, and C. Hu. Circuit aging simulator (cas). In *Proc. Int. Electron Devices Meeting IEDM '88. Technical Digest*, pages 134–137, 1988.

[66] S. Aur, D. E. Hocevar, and Ping Yang. Circuit hot electron effect simulation. In *Proc. Int. Electron Devices Meeting*, volume 33, pages 498–501, 1987.

[67] *Users' Manuals for BSIMPro+/RelXpert/UltraSim tools, www.cadence.com.*

[68] M. M. Lunenborg, J. D. Boter, and R. M. D. A. Velghe. Mos model 9 parameter extraction with realistic time-dependence for hot-carrier reliability simulation. In *Proc. Proceeding of the 28th European Solid-State Device Research Conf*, pages 248–251, 1998.

[69] S. Ghosh, F. Marc, C. Maneux, B. Grandchamp, G.A. Koné, and T. Zimmer. Thermal aging model of inp/ingaas/inp dhbt. *Microelectronics Reliability*, 50(9-11):1554 – 1558, 2010. 21st European Symposium on the Reliability of Electron Devices, Failure Physics and Analysis.

[70] Y. Kawakami, Jingkun Fang, H. Yonezawa, N. Iwanishi, Lifeng Wu, A. I-Hsien Chen, N. Koike, Ping Chen, Chune-Sin Yeh, and Zhihong Liu. Gate-level aged timing simulation methodology for hot-carrier reliability assurance. In *Proc. Asia and South Pacific Design Automation Conf the ASP-DAC 2000*, pages 289–294, 2000.

[71] Chin-Chi Teng, Weishi Sun, Sung-Mo Kang, Peng Fang, and J. Yue. irule: fast hot-carrier reliability diagnosis using macro-models. In *Proc. Custom Integrated Circuits Conf. the IEEE 1994*, pages 421–424, 1994.

[72] Lifeng Wu, Jingkun Fang, Hirokazu Yonezawa, Yoshiyuki Kawakami, Nobufusa Iwanishi, Heting Yan, Ping Chen, Alvin I-Hsien Chen, Norio Koike, Yoshifumi Okamoto, Chune-Sin Yeh, and Zhihong Liu. Glacier: a hot carrier gate level circuit characterization and simulation system for vlsi design. In *Proc. IEEE 2000 First Int. Symp. Quality Electronic Design ISQED 2000*, pages 73–79, 2000.

[73] Y.-H. Shih and S.-M. Kang. Analytic transient solution of general mos circuit primitives. 11(6):719–731, 1992.

[74] Y.-H. Shih and S. M. Kang. Illiads: a new fast mos timing simulator using direct equation-solving approach. In *Proc. 28th ACM/IEEE Design Automation Conf*, pages 20–25, 1991.

[75] Ayse K. Coskun and Tajana Simunic Rosing. Improving energy efficiency and reliability through workload scheduling in high-performance multicore processors. *DAC.COM Knowledge Center Article*, 2010.

[76] A. A. Jerraya and W. Wolf. *Multiprocessor Systems-on-Chips*. Elsevier, 2005.

[77] N. Ventroux, A. Guerre, T. Sassolas, L. Moutaoukil, G. Blanc, C. Bechara, and R. David. Sesam: An mpsoc simulation environment for dynamic application processing. In *Proc. IEEE 10th Int Computer and Information Technology (CIT) Conf*, pages 1880–1886, 2010.

[78] T. Gupta, C. Bertolini, O. Heron, N. Ventroux, T. Zimmer, and F. Marc. Effects of various applications on relative lifetime of processor cores. In *Proc. IEEE Int. Integrated Reliability Workshop Final Report IRW '09*, pages 132–135, 2009.

[79] T. Gupta, C. Bertolini, O. Heron, N. Ventroux, T. Zimmer, and F. Marc. High level power and energy exploration using archc. In *Proc. 22nd Int Computer Architecture and High Performance Computing (SBAC-PAD) Symp*, pages 25–32, 2010.

[80] K. Skadron, M. R. Stan, W. Huang, Sivakumar Velusamy, Karthik Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *Proc. 30th Annual Int Computer Architecture Symp*, pages 2–13, 2003.

[81] K. Skadron. The importance of computer architecture in microprocessor thermal design. In *Proc. Ninth Intersociety Conf. Thermal and Thermomechanical Phenomena in Electronic Systems ITHERM '04*, volume 2, pages 729–730, 2004.

[82] Wu-chun Feng. Making a case for efficient supercomputing. *Queue*, 1:54–64, October 2003.

[83] J. Coburn, S. Ravi, and A. Raghunathan. Power emulation: a new paradigm for power estimation. In *Proc. 42nd Design Automation Conf*, pages 700–705, 2005.

[84] E. Macii, M. Pedram, and F. Somenzi. High-level power modeling, estimation, and optimization. In *Proc. 34th Design Automation Conf*, pages 504–511, 1997.

[85] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. In *Proc. 27th Int Computer Architecture Symp*, pages 83–94, 2000.

[86] J. Laurent, N. Julien, E. Senn, and E. Martin. Functional level power analysis: an efficient approach for modeling the power consumption of complex processors. In *Proc. Design, Automation and Test in Europe Conf. and Exhibition*, volume 1, pages 666–667, 2004.

[87] Eric Senn, Johann Laurent, Nathalie Julien, and Eric Martin. Softexplorer: estimating and optimizing the power and energy consumption of a c program for dsp applications. *EURASIP J. Appl. Signal Process.*, 2005:2641–2654, January 2005.

[88] H. A. Mantooth and I. E. Getreu. The development of new power semiconductor models. In *Proc. IEE Colloquium CAD of Power Electronic Circuits*, 1992.

[89] D. F. Haslam. Modelling power mosfets for spice circuit simulation; a review and discussion. In *Proc. IEE Colloquium Large Signal Device Models and Parameter Extractions for Circuit Simulation*, 1988.

[90] S. Sangameswaran and S. Yamauchi. Post-layout parasitic verification methodology for mixed-signal designs using fast-spice simulators. In *Proc. IEEE Dallas/CAS Workshop: Architecture, Circuits and Implementtation of SOCs DCAS '05*, pages 211–214, 2005.

[91] T. H. Krodel. Power play-fast dynamic power estimation based on logic simulation. In *Proc. Conf. IEEE Int Computer Design: VLSI in Computers and Processors ICCD '91*, pages 96–100, 1991.

[92] R. Tjarnstrom. Power dissipation estimate by switch level simulation [cmos circuits]. In *Proc. IEEE Int Circuits and Systems Symp*, pages 881–884, 1989.

[93] W. Ye, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin. The design and use of simplepower: a cycle-accurate energy estimation tool. In *Proc. 37th Design Automation Conf*, pages 340–345, 2000.

[94] Mudge T. & Grunwald D. Austin, T. Sim-panalyzer.

[95] V. Tiwari, S. Malik, A. Wolfe, and M. T.-C. Lee. Instruction level power analysis and optimization of software. In *Proc. Conf. Ninth Int VLSI Design*, pages 326–328, 1996.

[96] H. Mehta, R. M. Owens, and M. J. Irwin. Instruction level power profiling. In *Proc. Conf. IEEE Int Acoustics, Speech, and Signal Processing ICASSP-96*, volume 6, pages 3326–3329, 1996.

[97] A. Sinha and A. P. Chandrakasan. Jouletrack-a web based tool for software energy profiling. In *Proc. Design Automation Conf*, pages 220–225, 2001.

[98] C. Krintz, Ye Wen, and R. Wolski. Application-level prediction of battery dissipation. In *Proc. Int. Symp. Low Power Electronics and Design ISLPED '04*, pages 224–229, 2004.

[99] F. Klein, G. Araujo, R. Azevedo, R. Leao, and L. C. V. dos Santos. A multi-model power estimation engine for accuracy optimization. In *Proc. ACM/IEEE Int Low Power Electronics and Design (ISLPED) Symp*, pages 280–285, 2007.

[100] F. Klein, R. Azevedo, and G. Araujo. High-level switching activity prediction through sampled monitored simulation. In *Proc. Int System-on-Chip Symp*, pages 161–166, 2005.

[101] Gang Qu, N. Kawabe, K. Usarni, and M. Potkonjak. Function-level power estimation methodology for microprocessors. In *Proc. 37th Design Automation Conf*, pages 810–813, 2000.

[102] Cristiano Araujo, Millena Gomes, Edna Barros, Sandro Rigo, Rodolfo Azevedo, and Guido Araujo. Platform designer: An approach for modeling multiprocessor platforms based on systemc. *Design Automation for Embedded Systems*, 10:253–283, 2005. 10.1007/s10617-006-0654-9.

[103] Rainer Leupers and Peter Marwedel. Retargetable code generation based on structural processor descriptions. design automation for embedded systems. In *In Design Automation for Embedded Systems*, pages 1–36. Kluwer Academic Publishers, 1998.

[104] A. Fauth, J. Van Praet, and M. Freericks. Describing instruction set processors using nml. In *In Proceedings on the European Design and Test Conference*, pages 503–507, 1995.

[105] G. Hadjiyiannis, S. Hanono, and S. Devadas. Isdl: An instruction set description language for retargetability. In *Proc. 34th Design Automation Conf*, pages 299–302, 1997.

[106] Ashok Halambi, Peter Grun, Vijay Ganesh, Asheesh Khare, Nikil Dutt, and Alex Nicolau. Expression:a language for architecture exploration through compiler/simulator retargetability. In *DATE '99: Proceedings of the conference on Design, automation and test in Europe*, page 100, New York, NY, USA, 1999. ACM.

[107] Wei Qin, Subramanian Rajagopalan, and Sharad Malik. A formal concurrency model based architecture description language for synthesis of software development tools. In *LCTES '04: Proceedings of the 2004 ACM SIGPLAN/SIGBED conference on Languages, compilers, and tools for embedded systems*, pages 47–56, New York, NY, USA, 2004. ACM.

[108] Stefan Pees, Andreas Hoffmann, Vojin Zivojnovic, and Heinrich Meyr. Lisa-machine description language for cycle-accurate models of programmable dsp architectures. In *DAC '99: Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, pages 933–938, New York, NY, USA, 1999. ACM.

[109] S. Rigo, G. Araujo, M. Bartholomeu, and R. Azevedo. Archc: a systemc-based architecture description language. In *Proc. 16th Symp. Computer Architecture and High Performance Computing SBAC-PAD 2004*, pages 66–73, 2004.

[110] N. Kavvadias and S. Nikolaidis. Elimination of overhead operations in complex loop structures for embedded microprocessors. 57(2):200–214, 2008.

[111] M. R. de Schultz, A. K. I. Mendonca, F. G. Carvalho, O. J. V. Furtado, and L. C. V. Santos. Automatically-retargetable model-driven tools for embedded code inspection in socs. In *Proc. 50th Midwest Symp. Circuits and Systems MWSCAS 2007*, pages 245–248, 2007.

[112] G. Beltrame, C. Bolchini, L. Fossati, A. Miele, and D. Sciuto. Resp: A non-intrusive transaction-level reflective mpsoc simulation platform for design space exploration. In *Proc. Asia and South Pacific Design Automation Conf. ASPDAC 2008*, pages 673–678, 2008.

[113] Sandro Rigo, Marcio Juliato, Rodolfo Azevedo, Guido Araújo, and Paulo Centoducatte. Teaching computer architecture using an architecture description language. In *WCAE '04: Proceedings of the 2004 workshop on Computer architecture education*, page 6, New York, NY, USA, 2004. ACM.

[114] Synopsys, design compiler,.

[115] Synopsys, primetime px, v.2007.12.

[116] Mentor graphics, modelsim v6.5b.

[117] Google code hmc-mips.

[118] T. Austin, E. Larson, and D. Ernst. Simplescalar: an infrastructure for computer system modeling. *Computer*, 35(2):59–67, 2002.

[119] Tsuneo Funubashi. Soc design flow. Technical report, 2007.

[120] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown. Mibench: A free, commercially representative embedded benchmark suite. In *Proc. WWC-4 Workload Characterization 2001 IEEE Int. Workshop*, pages 3–14, 2001.

[121] Lai-Man Po and Wing-Chung Ma. A novel four-step search algorithm for fast block motion estimation. 6(3):313–317, 1996.

[122] Samuel Goto. Adl synthesis using archc, February 2010.

[123] T. Gupta, C. Bertolini, O. Heron, N. Ventroux, T. Zimmer, and F. Marc. Method for selecting a resource from a plurality of processing resources such that the likely time lapses before resource failure evolve in a substantially identical manner, July 2011.

[124] Hao Shen and Qinru Qiu. An fpga-based distributed computing system with power and thermal management capabilities. In *Proc. 20th Int Computer Communications and Networks (ICCCN) Conf*, pages 1–6, 2011.

[125] H. Kimura, M. Sato, T. Imada, and Y. Hotta. Runtime dvfs control with instrumented code in power-scalable cluster system. In *Proc. IEEE Int Cluster Computing Conf*, pages 354–359, 2008.

[126] Hideaki Kimura, Takayuki Imada, and Mitsuhisa Sato. Runtime energy adaptation with low-impact instrumented code in a power-scalable cluster system. In *Proc. 10th IEEE/ACM Int Cluster, Cloud and Grid Computing (CCGrid) Conf*, pages 378–387, 2010.

Personal publications

International conferences

Publications in Journal

[JOLPE2011] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., Impact of Power Consumption and Temperature on Processor Lifetime Reliability (Accepted), American Scientific Publishers in JOLPE Vol. 7 N° 5, December 2011

Publications in Conferences with reviews for presentation

[SBACPAD2010] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., High Level Power and Energy Exploration Using ArchC, Proc. 22nd Int Computer Architecture and High Performance Computing (SBAC-PAD) Symp, Petropolis, Rio de Janeiro, Brazil, October 2010

Publications in Comferences with reviews for poster presentation

[IRW2009] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., Effects of various applications on relative lifetime of processor cores (IRW), Lake Tahoe, California, US, May 2010

[IRW2010] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., RAAPS: Reliability Aware ArchC based Processor Simulator, Proc. IEEE Int. Integrated Reliability Workshop Final Report (IRW), Lake Tahoe, California, US, October 2010

Communication without reviews

[HIPEAC2010] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., Predicting Lifetime using power consumption from 'Wattch', Proceedings of the 6th International Summer School on Adv. Computer Arch. and Compilation for Embedded Systems, ACACES, Terrassa, Spain (pp. 197-198). Ghent: HiPEAC, July 2010

[VARI2011] Gupta, T. and Bertolini, C. and Heron, O. and Ventroux, N. and Zimmer, T. and Marc, F., System Level Analysis and Accurate Prediction of Electromigration, in European Workshop on CMOS Variability (VARI), Grenoble, France, May 2011

Résumé

Les systèmes multiprocesseurs sur puce (MPSoC) sont une solution de calcul de plus en plus adoptée dans de nombreux domaines d'applications de l'embarqué tels que l'avionique, l'automobile, la production industrielle, etc. Ces systèmes sont composés jusqu'à plusieurs centaines de processeurs, de mémoires et de réseaux sur puce. La miniaturisation des dispositifs (transistor et interconnexion) a pour effet d'améliorer leur performance et d'augmenter leur nombre par unité de surface mais cela au détriment de leur fiabilité qui au contraire tend à diminuer. La roadmap ITRS identifie la fiabilité des interconnexions comme un des 5 défis majeurs à adresser avant le noeud 22nm. La fiabilité des transistors est affectée par des mécanismes de dégradation et de variation qui causent un changement de leur tension de seuil voire la perte de leur fonctionnalité. Cette thèse s'intéresse aux phénomènes de vieillissement dans les dispositifs qui se manifestent pendant la vie d'un circuit intégré et qui causent une dégradation des performances. Pour maintenir constant le niveau de fiabilité globale d'un circuit intégré, le taux de défaillance par dispositif doit diminuer avec l'augmentation de la densité d'intégration à chaque saut technologique. De nouvelles méthodologies de conception sont nécessaires en complément des solutions existantes ou envisagées dans les fonderies. Cette thèse propose une méthodologie de conception pour analyser et améliorer la fiabilité de systèmes MPSoC dans le cycle "front-end" du flot de conception, i.e. avant la synthèse physique. La première contribution de cette thèse est une méthode pour élaborer une métrique et des macro-modèles de vieillissement pour un circuit intégré. Ces modèles permettent d'estimer la probabilité de défaillance d'un circuit dû au mécanisme d'"electromigration", de NBTI, d'injection de porteurs chauds et de claquage d'oxyde de grille en fonction de données de conception, d'assemblage et de fabrication et des conditions de fonctionnement et d'environnement. La deuxième contribution de cette thèse est une chaîne d'outils de simulation qui permet l'estimation de la probabilité de défaillance dans un processeur de type MIPS. Plus précisément, il s'agit d'un simulateur d'instructions (ISS) augmenté avec des capacités d'estimation de la puissance consommée, de température dissipée et de fiabilité du processeur. Ce simulateur permet au concepteur d'analyser rapidement l'impact de choix de conception sur la fiabilité du processeur. Un intérêt majeur de ces contributions est la possibilité d'identifier la partie de la micro-architecture la plus vulnérable aux différents mécanismes de défaillance, en fonction du design et du programme exécuté. Ce nouveau ISS est décrit en langage SystemC, ce qui le rend apte à être intégré dans un simulateur de MPSoC. Enfin, cette thèse présente plusieurs scénarios montrant la manière d'améliorer la fiabilité d'un processeur avec l'aide de ces contributions, au niveau système du cycle de développement.

mots-clés : MPSoC, vieillissement, NBTI, electromigration, HCI, TDDB, processeur, ISS, ArchC, SystemC, MIPS.

Summary

Multi-Processor System-On-Chip (MPSoC) are complex digital circuits but are very attractive for embedded computing intensive applications. They are widely used in different type of industrial products, e.g., avionics, automobiles, electrical appliances, factory machines, and so on. Such integrated circuits (IC) are composed of up to hundreds of processor cores, memories and interconnect. Die shrinking leads to faster devices and higher number of transistors per unit area thus increasing performance of ICs, but in detriment of device reliability that tends to decrease. ITRS roadmap identifies the interconnect reliability as one of the important difficult issues that need to be solved before the 22nm node. Transistor reliability is affected by degradation and variation phenomena that cause a drift of their threshold voltage till the loss of their functionality. In this thesis, we focus on aging related failures that occur during the IC life and cause performance till functionality losses. To keep the whole IC reliability constant, the failure rate per device must decrease as transistor density increases at each shrinking step or technology node. New design methodologies are required in complement to process-level solutions so as IC costs remain at acceptable level. This thesis proposes a methodology for analyzing and optimizing the reliability of MPSoC systems in a design flow starting from system-design level i.e. before synthesis. The first contribution is a method to derive a metric and macro-ageing models for computing the probability of electromigration, NBTI, hot carrier injection and time-dependent dielectric breakdown failures in a digital circuit regarding design and assembly inputs, operating and environment conditions and process information. The second contribution is a simulation tool-chain that enables the estimation of failure probability in a MIPS based processor. More precisely, we enhance an instruction set simulator (ISS) of MIPS with power, temperature and ageing estimation capabilities so as a designer can get evaluations quickly regarding some design choices. One great benefit of this tool-chain is the ability to highlight which part of the processor microarchitecture is the most prone to a failure mechanisms and for which program executed. The augmented ISS is ready to be integrated in a SystemC based MPSoC simulator. Finally, we present several scenarios to improve the reliability of the processor at system-design level.

Keywords: MPSoC, vieillissement, NBTI, electromigration, HCI, TDDB, processeur, ISS, ArchC, SystemC, MIPS.