2012

RAKSHA:Reliable and Aggressive frameworK for System design using High-integrity Approaches

Naga Durga Prasad Avirneni
Iowa State University

RAKSHA:Reliable and Aggressive frameworK for System design using High-integrity Approaches

by

Naga Durga Prasad Avirneni

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Program of Study Committee:
Arun K. Somani, Major Professor
Akhilesh Tyagi
Fernandez Baca
Joseph Zambreno
Phillip Jones
Zhao Zhang

Iowa State University

Ames, Iowa

2012

To my dear parents

To my loving sisters

To my beloved friends

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis.

First and foremost, I would like to thank my advisor Professor Arun Somani, for his invaluable guidance, support and encouragement throughout my PhD. In particular, he helped me to develop and establish good technical writing and presentation skills. His encouragement to explore to new ideas and push existing boundaries has helped me to develop this thesis. If not for my advisor, I would not have been able accomplish my research.

I would also like to thank my committee members, Dr. Akhilesh Tyagi, Dr. David Fernandez-Baca, Dr. Joseph Zambreno, Dr. Phillip Jones, and Dr. Zhao Zhang for serving on my committee and for the constructive feedback and suggestions during the course of the study. I am also thankful to them for providing me with sufficient knowledge to conduct this research. I am thankful to the ECpE department and Iowa State University for providing me with a great atmosphere for carrying out this research.

I thank all my past and present research lab members at Dependable Computing and Networking lab (DCNL): Mike, Srivatsan, Koray, David, Visu, Jinxu, Prem, Pavan, Lizandro, and Cory. It was great working with them and also learnt a lot from our 592 seminar discussions.

Finally, I would like to thank my parents, friends and family for their support and motivation during the writing of this work.

ABSTRACT

Advances in the fabrication technology have been a major driving force in the unprecedented increase in computing capabilities over the last several decades. Despite huge reductions in the switching energy of the transistors, two major issues have emerged with decreasing fabrication technology scales. They are: 1) increased impact of process, voltage, and temperature (PVT) variation on transistor performance, and 2) increased susceptibility of transistors to soft errors induced by high energy particles. In presence of PVT variation, as transistor sizes continue to decrease, the design margins used to guarantee correct operation in the presence of worst-case scenarios have been increasing.

Systems run at a clock frequency, which is determined by accounting the worst-case timing paths, operating conditions, and process variations. Timing speculation based reliable and aggressive clocking advocates going beyond worst-case limits to achieve best performance while not avoiding, but detecting and correcting a modest number of timing errors. Such design methodology exploit the fact that timing critical paths are rarely exercised in a design, and typical execution happens much faster than the timing requirements dictated by worst-case scenarios. Better-than-worst-case design methodology is advocated by several recent research pursuits, which propose to exploit in-built fault tolerance mechanisms to enhance computer system performance. Recent works have also shown that the performance lose due to over provisioning base on worst-case design margins is upward of 20% in terms operating frequency and upward of 50% in terms of power efficiency.

The threat of soft error induced system failure in computing systems has become more prominent as we adopt ultra-deep submicron process technologies. With respect to soft error susceptibility, decreasing transistor geometries lower the energy threshold needed by high-energy particles to induce errors. As this trend continues, the need for fault tolerance mechanisms to counteract this effect has moved from a nice to have, to be a requirement in current and future systems. In this dissertation,

RAKSHA (meaning to protect and save in Sanskrit), we take a multidimensional look at the challenges of system design built with scaled-technologies using high integrity techniques.

In RAKSHA, to mitigate soft errors, we propose lightweight high-integrity mechanisms as basic system building blocks which allow system to offer performance levels comparable to a non-fault tolerant system. In addition, we also propose to effectively exploit and use the availability of fault tolerant mechanisms to allow and tolerate data-dependent failures, thus setting systems to operate at typical case circuit delays and enhance system performance. We also propose the use of novel high-integrity cells for increasing system energy efficiency and also potentially increasing system security by combating power-analysis-based side channel attacks. Such an approach allows balancing of performance, power, and security with no further overhead over the resources needed to incorporate fault tolerance. Using our framework, instead of designing circuits to meet *worst-case* requirements, circuits can be designed to meet *typical-case* requirements.

In RAKSHA, we propose two efficient soft error mitigation schemes, namely Soft Error Mitigation (SEM) and Soft and Timing Error Mitigation (STEM), using the approach of multiple clocking of data for protecting combinational logic blocks from soft errors. Our first technique, SEM, based on distributed and temporal voting of three registers, unloads the soft error detection overhead from the critical path of the systems. SEM is also capable of ignoring false errors and recovers from soft errors using *in-situ* fast recovery avoiding recomputation. Our second technique, STEM, while tolerating soft errors, adds timing error detection capability to guarantee reliable execution in aggressively clocked designs that enhance system performance by operating beyond worst-case clock frequency. We also present a specialized low overhead clock phase management scheme that ably supports our proposed techniques. Timing annotated gate level simulations, using 45*nm* libraries, of a pipelined adder-multiplier and DLX processor show that both our techniques achieve near 100% fault coverage. For DLX processor, even under severe fault injection campaigns, SEM achieves an average performance improvement of 26.58% over a conventional triple modular redundancy voter based soft error mitigation scheme, while STEM outperforms SEM by 27.42%. We refer to systems built with SEM and STEM cells as reliable and aggressive systems.

Energy consumption minimization in computing systems has attracted a great deal of attention and

has also become critical due to battery life considerations and environmental concerns. To address this problem, many task scheduling algorithms are developed using dynamic voltage and frequency scaling (DVFS). Majority of these algorithms involve two passes: schedule generation and slack reclamation. Using this approach, linear combination of frequencies has been proposed to achieve near optimal energy for systems operating with discrete and traditional voltage frequency pairs. In RAKSHA, we propose a new slack reclamation algorithm, aggressive dynamic and voltage scaling (ADVFS), using reliable and aggressive systems. ADVFS exploits the enhanced voltage frequency spectrum offered by reliable and aggressive designs for improving energy efficiency. Formal proofs are provided to show that optimal energy for reliable and aggressive designs is either achieved by using single frequency or by linear combination of frequencies. ADVFS has been evaluated using random task graphs and our results show 18% reduction in energy when compared with continuous DVFS and over more than 33% when compared with scheme using linear combination of traditional voltage frequency pairs.

Recent events have indicated that attackers are banking on side-channel attacks, such as differential power analysis (DPA) and correlation power analysis (CPA), to exploit information leaks from physical devices. Random dynamic voltage frequency scaling (RDVFS) has been proposed to prevent such attacks and has very little area, power, and performance overheads. But due to the one-to-one mapping present between voltage and frequency of DVFS voltage-frequency pairs, RDVFS cannot prevent power attacks. In RAKSHA, we propose a novel countermeasure that uses reliable and aggressive designs to break this one-to-one mapping. Our experiments show that our technique significantly reduces the correlation for the actual key and also reduces the risk of power attacks by increasing the probability for incorrect keys to exhibit maximum correlation. Moreover, our scheme also enables systems to operate beyond the worst-case estimates to offer improved power and performance benefits. For the experiments conducted on AES S-box implemented using 45nm CMOS technology, our approach has increased performance by 22% over the worst-case estimates. Also, it has decreased the correlation for the correct key by an order and has increased the probability by almost 3.5X times for wrong keys when compared with the original key to exhibit maximum correlation. Overall, RAKSHA offers a new way to balance the intricate interplay between various design constraints for the systems designed using small scaled-technologies.

CHAPTER 1. INTRODUCTION

The progress made in fabrication technology over the last several decades has been the primary driving force in the corresponding explosion of computing capability. However, as fabrication technology continues to produce faster transistors at smaller geometries, the extraction of their full computing potential has become increasingly challenging. One of the biggest issues in such realization is the increasing impact that process, voltage, and temperature (PVT) variation have on designing systems built using these small-scale transistors. The state of the art in today's commercial designs use worst-case PVT variation analysis to help provide correct operation guarantees. Industry has been concerned about PVT variation for years and have combated this issue by employing post-fabrication speed binning techniques to allow them to sell faster chips at higher prices, thus increasing their profit margins. Even the post-fabrication tuning is governed by conservative principles and the working boundaries are still set by worst-case estimates. The result of *worst-case* analysis is conservative guard banding and resource over provisioning for the design to work properly. The gap in performance between designing for the worst-case versus the *typical-case* is becoming large. Brooks, in 2009, showed that designing for the worst-case condition as opposed to the typical-case condition can cost more than 20% in operating frequency performance, and more than 50% in power efficiency [33].

1.1 Fault Tolerance

Overclocking as a means to improve performance is a popular technique among enthusiasts [24]. Microprocessor vendors are even introducing capabilities in the chipset, examples being AMD's Overdrive and Advanced Clock Calibration techniques, to support overclocking. The latest 45nm AMD Phenom II processor has been overclocked to upwards of 5GHz from its rated frequency of 2.8GHz, using liquid nitrogen cooling [85]. Frequency binning and discreteness in artifact grading are con-

tributing to this improvement in performance through overclocking. Also, circuits exhibit worst case delay only when their longest delay paths are sensitized by the inputs. However, these worst case delay inducing inputs and operating conditions are rare, leading to room for performance improvement that overclockers exploit [4]. The problem is that timing errors may occur at overclocked speeds, so in order to reliably take advantage of this improvement in execution time, the processor must be made in some way tolerant to errors.

Overclocking without guaranteeing functional correctness leads to unpredictable system behavior and loss of data. Aggressive, but reliable, design methodologies employ relevant timing error detection and recovery schemes to prevent erroneous data from being used [25, 27]. In [9, 73], it has been shown that operating frequency can be increased beyond worst case limit, allowing systems to operate at optimal clock frequency, by adapting to the current set of instructions and environmental conditions. Performance gains as high as 50% was achieved for a small recoverable error rate of 1%.

Nano-sized transistors, coupled with deployment in hazardous environments, have magnified the reliability concerns plaguing modern computing systems. Rapid enhancements in VLSI technology have fueled the increasing apprehension of system hardware being susceptible to myriad of faults. Many fault tolerance techniques are proposed at different levels of design hierarchy, starting from the design of hardened latches to system-level fault tolerant architectures [1, 3, 29, 49]. All these techniques strive to provide high degrees of fault coverage by providing redundancy in either information, spatial or temporal domains. For example, on-chip memories, which are regularly structured arrays, are protected by Error Correcting Codes (ECC) against transient bit-flips, also known as, soft errors [20]. ECC applies information redundancy to mitigate soft errors, while incurring resource overhead, in terms of area and power.

Technology scaling has also radically impacted the soft error occurrence and fault tolerance is becoming increasingly important in today's and future designs due to their increasing susceptibility to soft errors. In the past, single event upsets (SEUs) were a major concern in space applications creating hard threats like loss of control, which often lead to catastrophic system failures. An SEU is caused when a high energy particle, either from cosmic radiation or decaying radioactive material, strikes the silicon substrate. If enough charge is deposited by the strike, it causes a bit flip in the memory cell or a

transient pulse in the combinational logic. The latter is referred to as a Single Event Transient (SET). A report by NASA, in September 2009, indicates that cosmic ray intensities have increased 19% above the previous space-age highs [58]. Study conducted by Normand in [60], provide recent evidence of upsets at ground level, which implies, terrestrial applications also require fault tolerant techniques to ensure their dependability.

With shrinking transistor feature sizes, supply voltages and node capacitance's of the circuits are getting smaller and smaller. However, this lowers the energy threshold needed by high-energy particles to induce errors. It results in rapid increase in number of particles in the flux that can induce a soft error. As indicated in [70], the problem of soft errors in combinational circuits is becoming comparable to that of unprotected memory elements in current and future technologies. Radiation induced SET pulses have widths in the range of $500ps$ to $900ps$ in the $90nm$ process, as compared to $400ps$ to $700ps$ in the $130nm$ process [56].

Providing fault tolerance capabilities for random and complex logic is expensive, both in terms of area and power. Techniques such as, duplication and comparison, and temporal triple modular redundancy (TMR) and majority voting have been proposed to mitigate soft error rate in logic circuits [48]. These approaches pay penalty for false errors and incur performance overhead even during error-free operation. Also at this juncture, when static power is comparable to dynamic power, logic replication is not a viable alternative. Therefore, to recover from such errors, it is therefore imperative that fault tolerance be built into systems using small-scale transistors.

1.2 Energy Efficiency

Power consumption has become an important concern in the design of computer systems today due to battery life considerations and environmental concerns. Dynamic voltage and frequency scaling (DVFS) is an effective method to control the trade-off between performance and power consumption of computing systems. DVFS schemes dynamically scale the voltage and frequency of the CPU to provide variable amount of energy to process the workload. In general, scaling down voltage or frequency of the processor, results in reducing dynamic power consumption. Intel SpeedStep, Advance Micro Devices (AMD) PowerNow!, and Transmeta LongRun [64, 2, 28] are few examples for DVFS in real

processors. Recent developments include providing support for deeper integration of DVFS like adding more power saving states (like deep p-states and c-states) [65].

Task scheduling play crucial role in multiprocessor systems which are widely deployed in high performance computing systems like super computers, computer clusters etc. Task scheduling algorithms for the multiprocessor system are usually developed based on the task graph. DVFS technique and task scheduling can be combined to form a two phase process for effective resource usage and energy consumption minimization. In this approach, first phase involve schedule generation, where tasks from the given task graph are scheduled on a given multi-processor system by attempting to maximize objective function which include minimizing makespan. In second phase, which involve slack reclamation, the output of first phase is post processed to minimize the energy consumption using DVFS technique and also without violating the deadline constraint.

The existing slack reclamation methods based on DVFS technique has a major shortcoming as they map each task to a single frequency among the available discrete set of (V, f) pairs of underlying DVFS platforms. As shown in [63], using only single frequency results in uncovered slack time, which reflect into energy wastage. To minimize uncovered slack time, a novel approach, multiple voltage-frequency selection dynamic voltage frequency scaling (MVFS-DVFS), has been proposed. MVFS-DVFS uses linear combination of frequencies for slack reclamation. With MVFS-DVFS technique, authors have shown that energy consumption can be pushed to the near optimal point of using (V, f) pairs supported by traditional DVFS schemes. However, MVFS-DVFS does not include the additional voltage and frequency swings that are possible by reliable and aggressive designs, which are built to operate beyond worst-case estimates..

1.3 Security

As implementation technology scales to nano-scale dimensions, electronic systems are now built with more complex and powerful integrated circuits. Along with the performance and power constraints, security also is emerging as a first class design constraint. This is especially true for embedded systems, which are deployed widely in domains ranging from mobile phones to smart cards. Rapid enhancements in VLSI technology, combined with innovations at various design hierarchies have con-

tributed to meet the power and the performance constraints. For all concerns related to security, cryptography is being applied to guard the systems from known attacks.

Power analysis attacks are cryptanalytic attacks that exploit information leaks from a cryptographic system. The idea to use cryptographic system power traces for extracting secret key was first presented by Kocher [39]. Power analysis attacks involve two basic techniques: Simple power analysis (SPA) and differential power analysis (DPA). SPA attacks involve first order power analysis and are mounted on systems by using the fact that different operations in secret-key crypto algorithms consume different power. Using SPA attacks, secret information, such as secret key or parts of it, is inferred directly through visual inspection of the power traces. DPA attacks are based on statistical computations and are much powerful than SPA attacks as no detailed knowledge about the cryptographic system is required. An alternative to DPA was suggested by Brier named Correlation Power Analysis (CPA), which offers more efficient power analysis based on linear correlation techniques [15].

We develop a new countermeasure using dynamic voltage and frequency scaling (DVFS). Earlier, a similar approach, named random dynamic voltage and frequency scaling (RDVFS), was investigated by Yang et. al. to reduce the correlation of input data to the power consumption. RDVFS reduce the correlation by varying the system voltage and frequency randomly [87]. But, later it was shown that RDVFS approach cannot prevent DPA/CPA power attacks [7]. Major limitation for RDVFS based countermeasure is on account of one-to-one mapping between voltage and frequency of DVFS (V, f) pairs. We overcome this inherent limitation of DVFS platforms by breaking the one-to-one mapping between voltage and frequency of (V, f) pairs.

1.4 Thesis Contributions

Given that future architectures will have fault tolerance mechanisms built in, our goal in RAKSHA (meaning to protect and save in Sanskrit) is to extract the full computing potential of small-scale technology effectively. We propose to develop and deploy lightweight high-integrity mechanisms that allow circuits to aggressively, yet reliably, operate beyond their conservative worst-case limits. In addition, we further develop slack reclamation algorithms for scheduling tasks in reliable and aggressive systems built using our high integrity techniques to increase system energy efficiency. Our mechanisms also of-

fer a potential solution to increase system security by combating power-analysis-based side channel attacks.

The main contributions of this thesis are:

- We propose two efficient soft error mitigation schemes, Soft Error Migigation (SEM) and Soft and Timing Error Mitigation (STEM), considering the approach of multiple clocking of data for tolerating soft errors in combinational logic. Both these techniques remove the error detection overhead from the circuit critical path. These specialized register cells provide near 100% fault tolerance against transient faults. Our schemes tolerate fast transient noise pulses, which is the principal characteristic of SETs. Both our schemes have no significant performance overhead during error-free operation. SEM cells are capable of ignoring false positives and recovers from errors using *in-situ* fast recovery avoiding recomputation. STEM cells has soft error mitigation characteristics similar to SEM. STEM cells allow reliable dynamic overclocking and are capable of tolerating timing errors as well. We also propose a scheme to manage phase shifts between clocks locally with constant delay values. Such an approach increases the possible frequency settings for aggressively clocked designs and also minimizes the clock routing resources.

- We address the energy consumption minimization issue by proposing a new task scheduling algorithm, which continue to play an important role in design of real-time systems and in high performance computing systems. In this thesis, we propose a new slack reclamation algorithm, aggressive dynamic and voltage scaling (ADVFS), using reliable and aggressive systems. ADVFS exploits the enhanced voltage frequency spectrum offered by reliable and aggressive designs for improving energy efficiency. We also provide guidelines on how to manage the enhanced voltage frequency spectrum and the constraints required for limiting the maximum attainable under ADVFS. We also provide formal proofs to show that significant energy savings are attainable either by using single frequency or by linear combination of frequencies. Our scheme can be integrated as post processing step with any of the existing static and dynamic task scheduling algorithms.

- We present an effective countermeasure to counteract DPA/CPA based attacks by employing

reliable and aggressive designs. Salient feature of our approach lays in enabling systems to operate beyond the worst-case estimates while breaking the one-to-one relationship between the voltage and the frequency of synchronous circuits. Our technique, Aggressive frequency scaling (AFS), also increases the entropy of power traces and pushes the performance to higher levels by exploiting data heterogeneity. Also, our technique reduces the probability to observe maximum correlation value for the correct key and prevents power attacks by increasing the probability for incorrect keys to exhibit maximum correlation value.

1.5 Organization

The remainder of this thesis is organized as follows. In Chapater 3, we present design and error recovery details of reliable and aggressive designs. In Chapter 4, we present our high integrity techniques, SEM and STEM. In Chapter 5, we present our task scheduling algorithm to improve energy efficiency for reliable and aggressive systems built using our high integrity techniques. In Chapter 6, we present our approach for preventing power attacks using reliable and aggressive designs and conclude thesis in Chapter 7.

CHAPTER 2. BACKGROUND

In this chapter, review of relevant previous work is presented. In Section 2.1, a brief review of better-than-worst-case designs is presented. Section 2.2 presents a brief review of soft error aware computing and Section 2.3 presents a brief review of work that aims at improving energy efficiency. In Section 2.4, we present background realted to our task scheduling algorithm. In Section 2.5, we present various security countermeasures that are developed to prevent power attacks and their assosciated limitations.

2.1 Better-Than-Worst-Case Designs

As the performance margin between designing circuits to meet worst-case constraints verses typically-case constraints has become increasing significant, a new area of research emerged that aimed to use fault tolerance to allow synchronous circuits to perform reliably at the highest possible clock frequencies. A good summary of the principles of this new area is presented in [4]. The methods reviewed in [4] are good initial attempts at pushing circuits close to safe limits for increasing performance; however, most approaches are not feasible in practice. For example, in [81] a method for increasing clock frequency is proposed by adding redundancy and clocking each stage of a pipeline at a phase shift of the original clock. However, the overheads associated with the redundancy and clocking resources of this approach were not addressed for deployment into real-world designs. A more recent approach is presented in [82], where the clock frequency of a simple in-order processor was tuned dynamically. An extra delay chain with a worst-case propagation delay greater than the critical path of the processor was used to guide frequency tuning. The delay chain was monitored constantly and detection of errors were used to adjust the processor's frequency to avoid errors within the processor as operating conditions changed. A benefit of this approach is that it avoided errors within the processor, thus eliminating the

performance penalties often associated with error recovery circuitry. However, this approach can adapt to changes in operating conditions, but can't exploit data dependent circuit delay. In the SPRIT^{3}E project, this attribute of circuit delay has been exploited to further extend the operation boundaries often set by worst case estimates [9, 73, 10].

Timing error tolerance techniques were employed by Razor [27] and SPRIT^{3}E [9, 73, 10] to operate beyond worst-case limits. While Razor focused on achieving lower energy consumption by reducing supply voltage in each pipeline stage, SPRIT^{3}E improved performance of a superscalar processor by reliably overclocking its pipeline. However, in the presence of soft errors extra redundancy, either in spatial or temporal domain, needs to be augmented with the above presented techniques in-order to design circuits for typical-case operation. To address these concerns, we propose an efficient way to organize this extra redundancy through our high integrity mechanisms presented in later sections of this thesis.

2.2 Soft Error Aware Computing

A multitude of fault tolerant architectures for tolerating soft errors have been developed in the past by the research community. Many schemes incur performance overhead even during error-free operation and do not support timing speculation. LEON-FT processor [29] uses temporal triple modular redundancy (TMR) approach and triplicates every flip flop in the processor and incurs a 100% area overhead. Redundant multi-threading based schemes exploit instruction level parallelism to provide fault tolerance [83]. SSD framework [68] consists of an integrity checking architecture for superscalar processors that can achieve fault tolerance capability of a duplex system at much less cost than the traditional duplication approach. The REESE architecture [34] takes advantage of spare elements in a superscalar processor to perform redundant execution. DIVA [5] uses spatial redundancy by providing a separate, slower pipeline processor alongside the fast processor. Reunion [71] exchanges control and data flow information between the cores to speed up execution, while leveraging the redundancy to provide partial fault coverage. These approaches trade performance and power for achieving soft error fault tolerant capabilities and none of these approaches support timing speculation.

2.2.1 Problem of False Errors

To combat the problem of soft errors, spatial redundancy or time redundancy or both have been used in previously proposed solutions [3, 48, 59, 52, 23, 66]. Dual Modular Redundant (DMR) systems employ two components to detect soft errors and the error recovery process is triggered whenever the comparison between these two components fails. There is an error in the system only if the primary component has an error and not in the redundant component. If we assume, probability for any component (out of the two components) to fail is equally likely, then the false error rate is as high as 50%. This can make systems to pay a huge penalty in the form of performance and power during the error recovery process. This penalty is aggravated in TMR systems as the false error rate can shoot up to 66%. Moreover, voting process degrades system performance. So an ideal situation would be to reduce power and performance penalty by lowering the false error rate or completely avoiding them.

2.2.2 Data Latching Schemes

Multiple clocking of data has been widely used to combat the issue of soft errors in combinational logic [3, 48, 59, 52, 23, 66]. These approaches involve latching data at different times or latching data and its delayed version using a single clock. Authors in [23] optimize a temporal TMR framework that is presented in [48]. Even with this approach, the clock frequency at which the circuit can be clocked must include the delays of combinational logic, double the pulse width under consideration and also the delay incurred for the majority voting. Moreover, this technique doesn't detect false errors. More recently, BISER, a DMR framework, using C-elements to tolerate soft errors has been proposed [52]. It offers two techniques - 1) using logic duplication (DMR in space) and 2) time shifted data (DMR in time). Using BISER technique, the power overhead is estimated to be between 7-11%. As discussed by the authors in [52], hardware duplication and time redundancy techniques, such as multi-threading for error detection and software implemented fault tolerance have very significant power overheads when compared to BISER. Also, using C-elements, authors in [66] propose soft error hardened flip flops to tolerate wide error pulses or hard errors and applies DMR in both time and space domains. Another piece of closely related work is Paceline [32], which employs a leader-checker architecture

(a DMR scheme) in a chip multiprocessor system and requires logic duplication, and checkpointing at regular intervals. All the above presented schemes employ either DMR or TMR [48, 52, 23, 66] and pay penalty for false errors. It is estimated that static power is comparable to dynamic power with nano-sized transistors. Hence, providing fault tolerance for energy constrained systems with logic replication is not a viable option. In RAKSHA, with the design of our high-integrity techniques, we propose to completely remove the overhead paid for the false errors. Our techniques offer soft error protection with out logic duplication and also enable timing speculation.

2.3 Power Efficiency

In [67], dynamic processor overclocking is proposed to utilize the dynamic voltage and frequency scaling (DVFS) capabilities of processors to offer increased performance in power-constrained systems. In [69], a multiple clock domain approach for processor design is presented for improving energy efficiency. Locally synchronous and globally asynchronous design techniques were used with the goal of improving energy efficiency by running different parts of the processor with different clocks. Existing microprocessor queue structures were then used for inter-clock domain communication. Initial exploration for using an aggressive framework for power savings is carried out by Razor [25], which achieves energy consumption savings by tuning the supply voltage in each pipeline stage. However, these schemes do not cover the solution space fully. In RAKSHA, using our high integrity techniques, we not only expand the solution space of DVFS, we also exploit it at finer granularity that leads to higher power efficiency and performance-power tradeoff.

2.4 Task Scheduling

A significant body of work has been published on task scheduling for real-time embedded systems using various forms of DVFS-enabled techniques. Key idea of most of the existing algorithms is to use processor slack time while meeting task requirements like deadline. Existing techniques can be divided into two groups: 1) static and 2) dynamic schemes.

In static scheduling schemes, all information about the task is made available during the compile-time and the scheduler generates schedule while meeting deadline of all tasks. Schedule generation

involve maximing a chosen objective funtion, which often involve maximizing processor utilization [91, 41, 90, 26]. In dynamic scheduling schemes, during compile time, only deadline information of the task is made available. Rest of the parameters, like release time, are estimated during the run-time and the scheduler uses this information to generate a schedule that maximizes the chosen objective function [22, 42, 53]. Dynamic scheme are more useful, where variable system operating conditions, like temperature, affect system performance and power. Such schemes are widely deployed in power-aware devices like laptops. Many scheduling schemes are also proposed to make scheduler aware of the operating temperature, which attempt to keep peak temperature below the maximum specified limit [18, 44, 19, 51, 12]. However, most of the schemes proposed assume that system operate at traditional voltage and frequencies.

Reliable and aggressive designs are proposed to exploit data dependence characterictics of the underlying system circuits. Such design methodology allows frequency and voltage swings beyond the worst case estimates. In [25], voltage swings beyond the worst case are used to reduce power consumption with out significant perfomance loss. In [9, 73, 10], frequency swings beyond the worst case are used to increase system performance with out significant increase in power consumption. Other work proposed using reliable and aggressive designs attempt to increase system reliabilty from soft errors [6]. To the best of our knowledge, no effort has been put for developing task scheduling algorithms in systems using reliable and aggressive designs.

From prior work, most relevant to the work is presented in [37, 63]. In [37], using DVFS, authors proposed an energy reduction algorithm, RDVFS, which selects a suitable set of (V, f) pairs for task execution as uniformly as possible with slight increase in the overall execution time. Authors in [63], propose MVFS-DVFS to minimize energy consumption by using linear combination of frequencies for slack reclamation. Our scheme, differes from the above mentioned schemes by utilizing the additional voltage and frequency swings that are provided by reliable and aggressive desings. Utilizing such swings, effectively utilize data dependence behavior of underlying system circuits for improving energy efficiency and also enable *typical-case* operation.

2.5 Security

Simple Power Analysis (SPA) and Differential Power Analysis (DPA) are common side channel attacks, that attempt to extract the secret information present in a system [45, 35]. SPA attacks involve first order power analysis and are mounted on systems by using the fact that different operations in crypto algorithms consume different power. Using SPA attacks, secret information, such as a secret key or parts of it, can be inferred directly through visual inspection of power traces. Countermeasures proposed for this attack are straight forward. DPA attacks involve higher order analysis of power traces like statistical analysis [50]. Hence, solutions proposed to prevent SPA attacks cannot be directly applied to prevent DPA attacks.

Various countermeasures have been proposed to prevent DPA attacks, varying in their degree of protection. The key goal of countermeasures is to tune power traces to prevent information leakage. Techniques, like insertion of dummy instructions to misalign power traces [16] and masking techniques at the gate level [80], are proposed to counteract DPA. In [30, 46], it was shown that these techniques do not effectively prevent DPA attacks. Other countermeasures like, masked dual-rail pre-charge logic [62] and logic families whose power consumption is independent of processed data [78], have been proposed to prevent DPA attacks. But the increased area, power, performance, and design time overheads limit their adoptability. A countermeasure using a DVFS approach called RDVFS was proposed in [87], which randomly selects voltage scaling and associated frequency. This technique is area and power efficient. However, even with RDVFS, Baddam et. al. [7] were able to successfully compromise a system. This is due to the one-to-one mapping between voltage and frequency of RDVFS (V, f) pairs. In RAKSHA, by using our basic building blocks, we propose to break this one-to-one relationship used by current DVFS platforms. Such an approach does not involve algorithmic changes or require custom design flows.

CHAPTER 3. RELIABLE AND AGGRESSIVE DESIGNS

3.1 Reliable and Aggressive Framework

In this chapter, the design and operation details of reliable and aggressive system to understand the rest of the thesis is presented. Timing speculation of aggressive systems allows data computed to be used speculatively.To guage the perfomance enhancemnet of a processor, we have conducted a case study by exploiting frequency swings offered by reliable and aggressive systems. This study results are also presented in this chapter.

3.1.1 An opportunity for increasing performance

The clock frequency of a pipelined system is determined by the pipeline stage with the longest critical path, under worst-case conditions. Traditional design methodologies are too conservative, as the conditions that produce worst-case delays rarely occur in tandem. Moreover, circuit delay has a strong association with the data being processed and hence, not all inputs of a task cause its worst-case delay. For instance, in a carry-propagate adder, the worst-case delay occurs only when the carry is propagated through each bit-slice, which occurs only for specific inputs [4]. The infrequent occurrence of critical timing delays has opened a new domain of study for increasing system performance by using timing speculation at the circuit level. Operating a system faster than worst-case allows us to exploit data dependent circuit delay variance and execute more efficiently.

The main concern with working at frequencies past the worst-case limit is that timing errors may occur. In order to reliably take advantage of performance improvements, the system must be made tolerant to these potential timing errors. Aggressive, but reliable, design methodologies employ timing error detection and recovery schemes to prevent erroneous data from propagating throughout the system [25, 9, 73, 10].

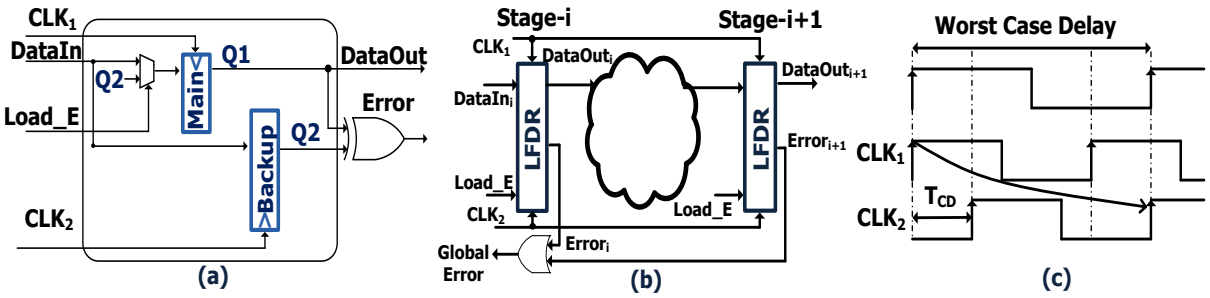

Figure 3.1 (a) Conceptual LFDR circuit to enable reliable overclocking (b) Designing a pipeline using LFDR (c) Illustration of aggressive clocking with worst case delay

3.1.2 Reliable overclocking mechanism

Adaptive and reliable overclocking approach uses circuitry placed between pipeline stages to locally detect and recover from timing errors [9, 73, 10]. The conceptual circuit element, Local Fault Detection and Recovery (LFDR), is shown in Figure 3.1(a). It consists of two registers, *Main* and *Backup*, clocked by clock signals CLK_1 and CLK_2 respectively. *DataIn* represents data coming from the combinational logic and *DataOut* represents the latched data that is sent to the next stages of computation. Both CLK_1 and CLK_2 have the same frequency at all times, but are phase shifted. Figure 3.1(b) explains how LFDR circuit is used in a pipeline. Figure 3.1(c) shows the timing relationship between the clock signals CLK_1 and CLK_2 with respect to the worst-case clock period. The contamination delay T_{CD} is the minimum amount of time between when the input to a logic circuit changes and the time when its output begins to change. The amount of phase shift is such that the time delay from the first rising edge of CLK_1 to the second rising edge of CLK_2 is not less than the maximum propagation delay of the circuit.

Two points to be noted are - 1) Computation that begins at the first rising edge of CLK_1 will produce a correct result by the second rising edge of CLK_2; and 2) If the input of the combinational circuit changes at the first rising edge of CLK_1, then the output of the combinational circuit will not change until the first rising edge of CLK_2. Thus *Backup* register always latches the correct value. It is also important to note that the extent of overclocking is limited by T_{CD}.

When data latched in the *Main* register and the *Backup* register do not match, a local error signal

Parameter	Value
Fetch/Decode/Issue/Commit width	4 inst/cycle
Functional units	4 INT ALUs, 1 INT MUL/DIV, 4 FP ALUs, 1 FP MUL/DIV
L1 D-cache	128K
L1 I-cache	512K
L2 Unified	1024K
Technology node	45*nm*
Base frequency	2.5GHz
No. of freq levels	32
Freq sampling	10μs
Freq penalty	0μs (Assuming Dual PLL)

Table 3.1 Simulator Parameters

is raised. A local recovery measure is then initiated by loading the value stored in *Backup* into *Main* during the next cycle. *Load_E* represents the control signal that is used to recover from timing error scenarios. In a pipelined system, a global error signal also is raised, as shown in Figure 3.1(b). All pipeline stages preceding this stage are stalled for a cycle, while all the stages following this stage process a bubble. Limiting the timing error budget to a reasonable number, SPRIT^{3}E has shown to enhance performance by up to 57% [73].

3.2 Performance Analysis

As the system is aggressively constrained, timing error rate can increase and can nullify any possible performance enhancement. Timings error rate introduce an upper bound for the possible frequency swings. To guage the frequency swings that are possible beyoond the traditional worst-case estimate, we have conducted a case study and results of the study are presented below.

3.2.1 Error rate characterization

To study the behavior of timing errors, we have conducted a simple study on Alpha processor model. Entire experiment framework has been developed as a part of our ongoing research efforts. We ran selected set of SPEC 2000 benchmark workloads on SimpleScalar - a cycle accurate simulator [17]. In order to embed timing aspects in SimpleScalar, we examined a hardware model of Alpha processor and obtained the number of timing errors occurring at different clock period, for each workload. We synthesized Illinois Verilog Model (IVM) for Alpha 21264 configuration using 45*nm* OSU standard cell

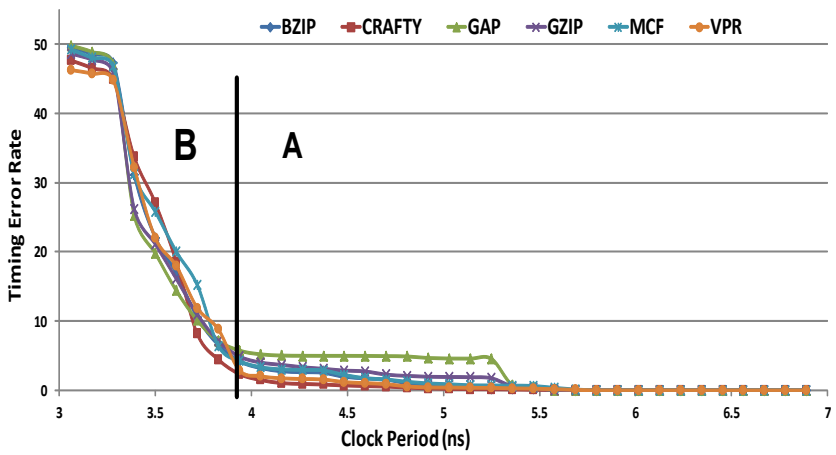

Figure 3.2 Cumulative error rate at different clock periods for the IVM Alpha processor executing instructions from SPEC 2000 benchmarks

library [72]. To match the RTL model, we adopt the same configuration for SimpleScalar simulations as well and the details of the simulation settings are presented in Table 3.1.

Figure 3.2 presents he cumulative error rate of six SPEC 2000 workloads for 32 equal intervals, for worst-case delay of $7ns$ and minimum contamination delay of $3.5ns$. The error profile obtained is the average values obtained by running the experiment for $100,000$ cycles, and repeating the experiment with different sequences of $100,000$ instructions for each workload. Although we are aware of the fact that the hardware configuration of the IVM pipeline is simplistic, we only use the study results to provide insights on timing error profile of a single core designed using reliable and aggressive design methodology.

As we notice from Figure 3.2, onset of timing errors doesn't begin until clock perios is $5.5ns$. It is resultant of the conservative guard band set for the system top operate at worst-case scenarios. Timing error profile presented can be divided into two regions, A and B. In region A, we see only moderate increase in timing error rate with frequency and in region B, we see an exponential increase in timing error rate. All applications have an error rate less than 5% in region A. In this region, clock frequency has been scaled up from $7ns$ to $3.9ns$, This translates into 79% increase in clock frequency for a 5% timing error rate. But for more complex processors such high frequency swings may not possible.

Recent trends show deeper integration of DVFS with increase in available voltage and frequencies [65]. It should be noted that increase in available voltages and frequencies are still estimated for worst-case scenarios. With this trend, distance between any two frequencies of (V, f) pair spectrum is less than 20-30% [28]. Therefore, for processors with high timing error rates, dynamic frequency swing should be limited to 20-30% of worst case clock frequency (region A).

CHAPTER 4. HIGH INTEGRITY TECHNIQUES

4.1 Introduction

Nano-sized transistors, coupled with deployment in hazardous environments, have magnified the reliability concerns plaguing modern computing systems. Rapid enhancements in VLSI technology have fueled the increasing apprehension of system hardware being susceptible to myriad of faults. Many fault tolerance techniques are proposed at different levels of design hierarchy, starting from the design of hardened latches to system-level fault tolerant architectures [1, 3, 29, 49]. All these techniques strive to provide high degrees of fault coverage by providing redundancy in either information, spatial or temporal domains. For example, on-chip memories, which are regularly structured arrays, are protected by Error Correcting Codes (ECC) against transient bit-flips, also known as, soft errors [20]. ECC applies information redundancy to mitigate soft errors, while incurring resource overhead, in terms of area and power.

In the past, single event upsets (SEUs) were a major concern in space applications creating hard threats like loss of control, which often lead to catastrophic system failures. An SEU is caused when a high energy particle, either from cosmic radiation or decaying radioactive material, strikes the silicon substrate. If enough charge is deposited by the strike, it causes a bit flip in the memory cell or a transient pulse in the combinational logic. The latter is referred to as a Single Event Transient (SET). A report by NASA, in September 2009, indicates that cosmic ray intensities have increased 19% above the previous space-age highs [58]. Study conducted by Normand in [60], provide recent evidence of upsets at ground level, which implies, terrestrial applications also require fault tolerant techniques to ensure their dependability.

With shrinking transistor feature sizes, supply voltages and node capacitance's of the circuits are getting smaller and smaller. However, this lowers the energy threshold needed by high-energy particles

to induce errors. It results in rapid increase in number of particles in the flux that can induce a soft error. As indicated in [70], the problem of soft errors in combinational circuits is becoming comparable to that of unprotected memory elements in current and future technologies. Radiation induced SET pulses have widths in the range of $500ps$ to $900ps$ in the $90nm$ process, as compared to $400ps$ to $700ps$ in the $130nm$ process [56].

Providing fault tolerance capabilities for random and complex logic is expensive, both in terms of area and power. Techniques such as, duplication and comparison, and temporal triple modular redundancy (TMR) and majority voting have been proposed to mitigate soft error rate in logic circuits [48]. These approaches pay penalty for false errors and incur performance overhead even during error-free operation. Also at this juncture, when static power is comparable to dynamic power, logic replication is not a viable alternative.

Increasing system wide integration force designers to adopt worst-case design methodologies, while designing individual system components. With such design practices, safety margins are added to address parameter variations, which include intra-die and inter-die process variations, and environmental variations, which include temperature and voltage variations [13, 14]. These additional guard bands are becoming non-negligible in nanometer technologies. Designers conservatively add these safety margins to salvage chips from timing failures and shortened lifetime. Most systems are characterized to operate safely below a particular vendor specified frequency. When they are operated beyond this rated frequency, timing errors and system failure may happen.

Overclocking as a means to improve performance is a popular technique among enthusiasts [24]. Microprocessor vendors are even introducing capabilities in the chipset, examples being AMD's Overdrive and Advanced Clock Calibration techniques, to support overclocking. The latest $45nm$ AMD Phenom II processor has been overclocked to upwards of $5GHz$ from its rated frequency of $2.8GHz$, using liquid nitrogen cooling [85]. Frequency binning and discreteness in artifact grading are contributing to this improvement in performance through overclocking. Also, circuits exhibit worst case delay only when their longest delay paths are sensitized by the inputs. However, these worst case delay inducing inputs and operating conditions are rare, leading to room for performance improvement that overclockers exploit [4]. The problem is that timing errors may occur at overclocked speeds, so in order

to reliably take advantage of this improvement in execution time, the processor must be made in some way tolerant to errors.

Overclocking without guaranteeing functional correctness leads to unpredictable system behavior and loss of data. Aggressive, but reliable, design methodologies employ relevant timing error detection and recovery schemes to prevent erroneous data from being used [25, 27]. In [9, 73], it has been shown that operating frequency can be increased beyond worst case limit, allowing systems to operate at optimal clock frequency, by adapting to the current set of instructions and environmental conditions. Performance gains as high as 50% was achieved for a small recoverable error rate of 1%.

Safety critical systems with hard real-time constraints require wide fault coverage with no compromise in performance. An interesting capability in nanometer design space, we believe, is to provide soft error tolerant reliable execution for high performance aggressive designs. In this chapter, we propose new ways of designing fault tolerant and reliably overclocked register cells that enable systems to improve both their performance and dependability.

4.1.1 Chapter Contributions

In this chapter, we address the issue of soft errors in random logic and develop solutions that provide fault tolerance capabilities. For this, we consider the approach of multiple clocking of data for detecting and correcting soft errors in combinational logic, an approach widely used in [3, 48, 59, 52, 23, 66]. Our first technique, SEM, replaces register elements in a circuit with **Soft Error Mitigation** (SEM) register cells. It allows systems to operate without the overhead of soft error detection circuitry. Unlike earlier approaches proposed using dual modular redundancy in space/time (DMR) and triple modular redundancy in time (TMR), SEM technique differs in error detection and recovery process and doesn't incur any performance penalty during error-free operation. SEM completes error detection process with two comparisons and also pays no penalty for false errors. Also on error detection, SEM avoids recomputation and recovers using local recovery. Our second technique, STEM, concurrently detects and corrects soft and timing errors using **Soft and Timing Error Mitigation** (STEM) register cells. STEM cells have soft error mitigation capabilities comparable to those of SEM cells, and they also support reliable overclocking. Both of our techniques employ a distributed and temporal voting scheme that

enables *in-situ* error detection and fast recovery. We support circuit level speculation in both SEM and STEM techniques. We allow data to move forward speculatively, and when an error happens we void the computation and perform re-computation. Since systems are protected from timing errors that may happen when systems are clocked aggressively, STEM approach allows systems to operate beyond design margins by providing support for reliable dynamic overclocking. Developed from better-than-worst-case design methodologies, dynamic overclocking exploits input data dependencies permitting the clock frequency to increase beyond the critical path delays and allowing systems to operate at optimal clock frequency [9, 73].

For error detection and correction, our temporal data sampling mechanisms, sample data at three different time intervals and thus require three clocks for proper operation. Clock distribution and routing are significant challenges in nano-scale technologies. Clock distribution network (CDN) consumes a significant portion of the power, area and metal resources in an integrated circuit. As a consequence, a specialized clock generation, distribution and routing scheme that minimizes the clock distribution overhead incurred by our fault mitigation techniques is important. Also, to support reliable dynamic overclocking, as discussed in [9, 73], it is important to precisely control the relative phase shifts of clock signals at high frequencies. Therefore, we focus on developing an efficient local clock manager (LCM), which helps in generating the required clock signals, with the desired phase shifts, locally. The clocks, so generated and distributed, satisfy the timing constraints required for proper working of our techniques. We also analyze the area overhead incurred for developing such LCMs.

For our experimental study, we integrated our data sampling mechanisms into a two stage pipeline consisting of an adder and a multiplier. Our results show that, with STEM cells, performance of this system can be increased by 55.93% over conventional TMR schemes, while providing near 100% fault coverage. For the experiments conducted on a DLX processor, we observed that SEM technique achieves an average performance improvement of 26.58% over the TMR scheme and STEM outperforms SEM by 27.42%, while providing near 100% fault coverage.

4.1.2 Chapter Organization

The remainder of this chapter is organized as follows. In Section 4.2, we describe our soft error mitigation technique and recovery mechanism. Section 4.3 describes how both timing error and soft error are concurrently detected and corrected. In Section 4.4, we discuss the issues in designing a pipeline system with our proposed soft/timing error mitigation techniques. Section 4.5 discusses the design aspects and area overheads of implementing a local clock manager. We present our results in Section 4.6 and related work in Section 4.7.

4.2 Soft Error Mitigation For High Performance

4.2.1 Problem of false errors

To combat the problem of soft errors, spatial redundancy or time redundancy or both have been used in previously proposed solutions [48]. Dual Modular Redundant (DMR) systems employ two components to detect soft errors and the error recovery process is triggered whenever the comparison between these two components fails. There is an error in the system only if the primary component has an error and not in the redundant component. If we assume, probability for any component (out of the two components) to fail is equally likely, then the false error rate is as high as 50%. This can make systems to pay a huge penalty in the form of performance and power during the error recovery process. This penalty is aggravated in TMR systems as the false error rate can shoot up to 66%. Moreover, voting process degrades system performance. So an ideal situation would be to reduce power and performance penalty by lowering the false error rate or completely avoiding them. With SEM cell design, we explain how the problem of false errors or false positives is avoided with the redundancy organization associated with systems employing them.

4.2.2 SEM cell Design

Prior soft error mitigation techniques at the circuit level are either based on temporal redundancy, spatial redundancy or a combination of both. These techniques achieve high degree of fault coverage, whilst degrading or trading performance, silicon area and other resources. For example, in [48], a spe-

cific design of a voting mechanism based on temporal triple modular redundancy is discussed, which mitigates all single event upsets. However, the overhead incurred is very high, as the operating frequency of a system built with such fault mitigation scheme must include the delays of combinational logic blocks, phase shifts of the clocks and the delay incurred by the voter. In this section, we present a variant of this scheme, and show that with a combination of local and global recovery, we can remove the additional overhead imposed, by the fault mitigation scheme, on the system operating frequency.

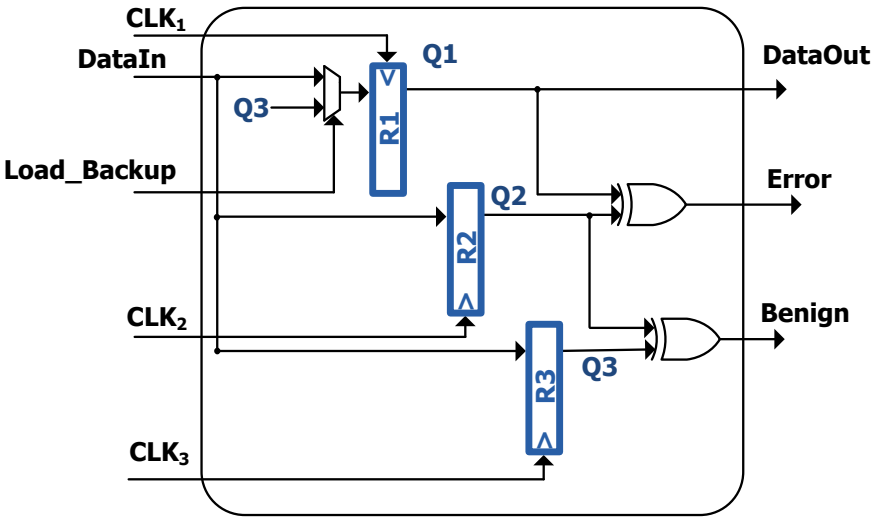

Figure 4.1 SEM Cell to Mitigate Soft Eerrors

The intent of our scheme is to make systems operate at frequencies same as that of non-fault tolerant designs, by unloading the error detection overhead from the circuit worst-case timing delay estimation. To keep the overhead of error detection off the critical path, we present the following redundancy organization using our *Soft Error Mitigation* (SEM) cells. Figure 4.1(a) shows a gate-level embodiment of a SEM cell. It consists of three registers R_1, R_2 and R_3, clocked by clock signals CLK_1, CLK_2 and CLK_3, respectively. *DataIn* represents data coming from the combinational logic and *DataOut* represents the latched data that is sent to the next stages of computation. *Error* and *Benign* represents the control signals generated by the voting process that help in soft error detection and recovery. *Load_Backup* represents the control signal that is used to recover from soft error scenarios. Data is sampled at three different time intervals T_1, T_2 and T_3, and are stored in registers R_1, R_2 and R_3, respectively.

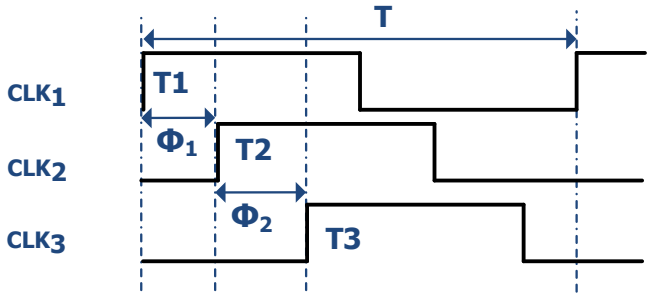

Figure 4.2 Timing Relationship between Clocks for Data Sampling

4.2.3 Timing Constraints

Figure 4.2 shows the timing relationship between the clock signals and the data sampling intervals. Clock signals, CLK_1, CLK_2 and CLK_3, have the same frequency, but they are out-of-phase by an amount governed by the timing constraints, explained below. Data is stored in registers at the rising edge of the clock signals, and strict timing constraints are required for efficient mitigation of soft errors. Notations that are used to explain the timing requirements are listed below. Contamination delay is the minimum amount of time beginning from when the input to logic becomes stable and valid to the time that the output of that logic begins to change. Propagation delay refers to the maximum delay of the circuit, under worst case conditions.

- T_{CD} = Contamination delay of the logic circuit
- T_{PD} = Propagation delay of the logic circuit
- T_{PW} = Expected soft error/noise pulse width
- Φ_1 = Phase shift between CLK_1 and CLK_2
- Φ_2 = Phase shift between CLK_2 and CLK_3
- T = Clock period

Equations (4.1) and (4.2) ensure that registers R_1, R_2 and R_3 are not corrupted by the same soft error. Since the system is running at CLK_1 frequency, data is forwarded speculatively to subsequent stages after latching in register R_1, and subsequent stages start their computation immediately.

$$\Phi_1 = T_2 - T_1 \geq T_{PW} \tag{4.1}$$

$$\Phi_2 = T_3 - T_2 \geq T_{PW} \tag{4.2}$$

Short paths present in the combinational circuit may corrupt the data before it gets latched in registers R_2 and R_3. Consequently, it is required to constrain short paths so that the same data registered in R_1 is also latched in registers R_2 and R_3, during error-free operation. Equation (4.3) ensure that these constraints are met, enabling timing speculation, by increasing the contamination delay above the desired combined phase shift values, given by Φ_1 and Φ_2. Equation (4.4) makes sure that temporal sampling happens only after the computation by the combinational logic is done. Our technique is capable of detecting all SEUs happening on registers, and all SETs having pulse duration less than T_{PW}.

$$T_{CD} \geq \Phi_1 + \Phi_2 \tag{4.3}$$

$$T \geq T_{PD} \tag{4.4}$$

4.2.4 Soft Error Detection and Recovery

Table 4.1 presents the possible soft error scenarios that a SEM technique is capable of detecting and recovering from. The table also lists the corresponding recovery mechanisms used. Once the data is latched in registers R_1, R_2 and R_3, they are compared with each other as shown in Figure 4.1(a) to produce ERROR and BENIGN signals. This comparison operation completes the voting process required to detect soft errors. On error detection, a single cycle system stall is all that is required for complete recovery. Below, we explain the different possible scenarios and the recovery mechanism used when an error happens.

- CASE I : No soft error occurs. Data latched in all three registers are correct. Both ERROR and BENIGN signals stay low, and no recovery mechanism is triggered. System operation continues without any interruption.

- CASE II : A soft error corrupts the data latched in register R_1. ERROR signal goes high after the data is latched in R_2. Since the next stage speculatively uses the data forwarded from R_1, re-computation is required next cycle to ensure functional correctness. The data stored in registers R_2 and R_3 are unaffected by the soft error. During the next cycle, value stored in R_2 or R_3 is loaded back into register R_1 with the help of the control signal LBKUP, completing the local recovery process. Figure 4.1 (a) shows R_3 being loaded into R_1. Global recovery, in the form of a stall signal sent to all other SEM cells that are unaffected by the soft error, is initiated and completed in one cycle.

- CASE III : A soft error corrupts the data latched in register R_2. Both the signals, ERROR and BENIGN, go high once temporal data sampling is completed. This is a false positive scenario. No recovery is required as data forwarded to the next stage is correct. System operation is not interrupted.

- CASE IV : This represents a case where register R_3 is corrupted with a soft error. In this case ERROR signal stays low, while BENIGN signal is asserted high. No recovery and interruption is required in this case too, as BENIGN signal is high.

As can be seen, SEM does not trigger error recovery for false positive scenarios. Also, since the data latched in R_1 is speculatively used by the succeeding stages, as soon as it is available, the error detection overhead is not incurred during normal system operation. This is also a low overhead solution, as it shuns the need for check pointing at regular time intervals. Thus, we enable systems to mitigate soft errors, using SEM cells, without any loss of performance, compared to a non-fault tolerant design.

Case	R_1	R_2	R_3	Error	Benign	Error Recovery
I	$\surd$	$\surd$	$\surd$	0	0	No recovery required
II	$\times$	$\surd$	$\surd$	1	0	Load R_2 or R_3 into R_1
III	$\surd$	$\times$	$\surd$	1	1	No recovery required
IV	$\surd$	$\surd$	$\times$	0	1	No recovery required

Table 4.1 Possible Soft Error Scenarios of SEM Cell ($\surd$ = No Error; $\times$ = Soft Error)

4.2.5 Fault Tolerance Analysis

The SEM technique detects and recovers from all possible soft error scenarios involving both SEUs and SETs. This scheme is well suited for fast transient pulses. Since fast transients typically correspond to soft errors with high strike rate probabilities, SEM cells have near 100% transient fault mitigation capability. Our scheme offers protection for pulses of widths less than the phase shifts provided between the clock signals. Any noise signal, whose pulse width exceeds this limit, cannot be detected by our scheme.

4.3 Soft Error Mitigation In Aggressive Designs

Aggressive designs are based on the philosophy that it is possible to go beyond worst-case limits to achieve best performance by not avoiding, but detecting and correcting a modest number of timing errors. Adaptive reliable overclocking technique proposed for synchronous implementation of logic blocks acts like an asynchronous system design technique exploiting data-dependent variance in delay. In this section, we further investigate the solution presented in previous section for soft error mitigation, and explain how it can be modified for soft error mitigation in aggressive designs, which uses adaptive reliable overclocking technique for improving system performance. Such a technique enables systems to operate beyond safety margins, while preserving data integrity.

With a conventional voter design, to detect and correct n errors simultaneously, we need to have up to $2n+1$ data samples. In our case, we have $n = 2$, since we need to detect and correct both soft and timing errors. For this analysis, we consider soft errors to be of only type SET. A traditional fault tolerance technique requires five different data values for guaranteeing both soft error and timing error detection and correction. The overhead incurred by this approach is very high as it increases the number of registers by four times, and requires five different clocks to sample data at five different times. Our goal is to develop a soft and timing error mitigation scheme that incurs minimal overhead. The proposed *Soft and Timing Error Mitigation* (STEM) cell is similar to the SEM cell in area complexity. However, the error detection and recovery mechanism is significantly different to address the requirements of concurrent soft and timing error mitigation.

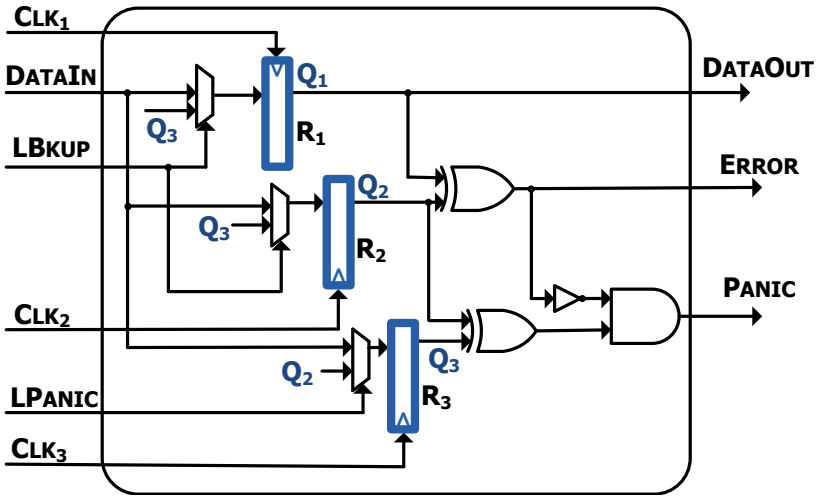

Figure 4.3 STEM Cell to Mitigate Soft and Timing Errors

4.3.1 Error Detection

Figure 4.3 shows a gate-level embodiment of a STEM cell, which acts as an on-line-fault monitor for soft and timing error mitigation. It also consists of three registers R_1, R_2 and R_3, clocked by clock signals CLK$_1$, CLK$_2$ and CLK$_3$, respectively. DATAIN represents data coming from the combinational logic and DATAOUT represents the latched data that is sent to the next stages of computation. ERROR and PANIC represents the control signals generated by the voting process that help in soft error and timing error detection and recovery. LBKUP and LPANIC represents the control signals that are used to recover from error scenarios as presented in Table 4.2. Under no error scenarios, incoming data is sampled at three different time intervals T_1, T_2 and T_3, and are stored in registers R_1, R_2 and R_3, respectively.

The working of a STEM cell is as follows. Once the data is latched in registers R_1 and R_2, they are compared with each other. This comparison operation completes the timing error detection process, since R_2 is timing safe [27, 9, 73]. But in the presence of soft errors, this comparison operation presents an ambiguous situation, as it is not possible to distinguish which one of these two registers is corrupted by an erroneous value. Also, value in R_2 is not to be trusted during the error recovery process.

If the comparison between R_1 and R_2 flags a mismatch, register R_3 is shielded from the incoming data value, and its content is used to recover the system state. This is done because any soft error that happens after comparing R_1 and R_2 has the potential to corrupt R_3 and push the system into an

unrecoverable state. Only when there is no mismatch between registers R_1 and R_2, register R_3 is allowed to latch the data safely. However, we have not yet ascertained whether R_3 is free from soft error. Therefore, we perform another comparison operation to complete the error detection process. After register R_3 is updated, we compare it with register R_2, to detect any error happening in register R_3. If there is no mismatch, register R_3 is trusted for error recovery purposes. If they mismatch, then that represents a case where register R_3 is corrupted by a soft error. At this point, it is possible to say that data latched in registers R_1 and R_2 are uncorrupted. The system is stalled for one cycle for flushing out the erroneous value from R_3, and loading either R_1 or R_2 value into R_3.

Case	R_1	R_2	R_3	ERROR	PANIC	RECOVERY
I	NE	NE	NE	0	0	No recovery required
II	SE	NE	NE	1	0	Load R_3 into R_1, R_2
III	NE	SE	NE	1	0	Load R_3 into R_1, R_2
IV	NE	NE	SE	0	1	Load R_2 into R_3
V	TE	NE	NE	1	0	Load R_3 into R_1, R_2
VI	TE	SE	NE	1	0	Load R_3 into R_1, R_2

Table 4.2 Possible Error Scenarios in using STEM Cell (NE = No Error; SE = Soft Error; TE = Timing Error)

4.3.2 Timing Constraints

As is the case with SEM cells, STEM cells also require strict timing constraints, to detect and correct soft and timing errors. STEM cells must satisfy Equations (4.1), (4.2) and (4.3). Equation (4.4) is modified as shown in Equation (4.5) for STEM cells. Equations (4.1) and (4.2) ensures that registers present in a STEM cell are not corrupted by the same SET. Equations (4.3) and (4.5) ensure that data latched in registers R_2 and R_3 are timing correct, i.e. free from timing errors. The timing relationships shown in Figure 4.2 still holds, with the caveat that Φ_1 also includes the extent of overclocking that is possible every cycle.

$$T + \phi_1 \geq T_{PD} \tag{4.5}$$

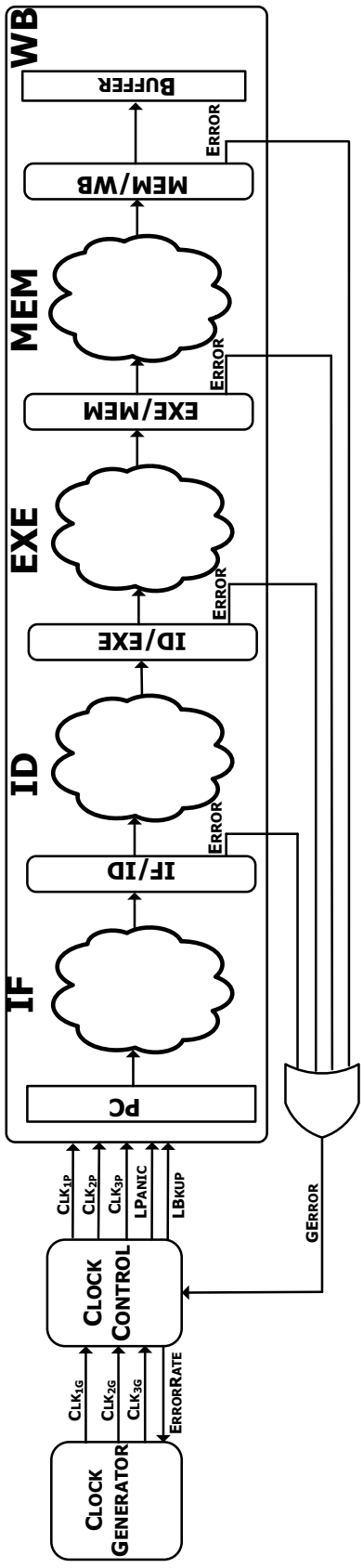

Figure 4.4 Pipeline Design with STEM Cells

4.3.3 Error Recovery

Table 4.2 lists all possible error scenarios with corresponding recovery mechanisms. In the following discussion, we explain the various possible events that take place in the STEM cell, and the associated recovery mechanism that is used in case of an error. It employs either a single cycle or three cycle fast local recovery based on the values of ERROR and PANIC signals, shown in Figure 4.3.

- CASE I : No error case. Both signals, ERROR and PANIC, stay low. System operation is not interrupted.

- CASE II, III, V, VI : This represents a case where one of the registers R_1 or R_2 is corrupted. In this case, ERROR $= 1$ and PANIC $= 0$. In this scenario R_3 is not updated, and the system recovers by loading R_3 in to R_1 and R_2 triggering re-computation. A three cycle global recovery process is initiated, which includes: one cycle stall for loading data back into the registers R_1 and R_2, using LBKUP signal, and two cycles for re-computation. This two cycle re-computation is required, as the error might have occurred because of overclocking, and this error will repeat in R_1, if sufficient time is not given for re-computation. This prevents recurrent system failures.

- CASE IV : Only R_3 is corrupted. In this case, ERROR $= 0$ and PANIC $= 1$. No re-computation is required. However, it is necessary to flush the erroneous data from R_3, to facilitate error recovery in subsequent cycles. As data in only R_3 is corrupted, "golden" data present in R_2 is loaded in to R_3. This requires a single cycle system stall, during which all STEM cells perform a local correction, using LPANIC signal.

4.3.4 Fault Tolerance Analysis

As is seen, the STEM technique detects and recovers from all possible soft and timing error scenarios, wherein the soft error is only of type SET. Also, the case where ERROR $= 1$ and PANIC $= 1$ never happens by design. Our technique leads to silent data corruption, if an SEU happens in R_3. However, since register R_3 is only used as a checkpointing register, a corrupted R_3 value may lead to failure, only if an error occurs in R_1 or R_2 in the next cycle. Consequently, the possibility of a system failure because of a SEU in R_3 is heavily mitigated. For Case VI, we expect that a TE or SE affects several STEM

cells, and the possibility of all cells having a TE in R_1 and SE in R_2 is insignificant. Hence, we hope one of the STEM cells will have the error signal triggered, preventing R_3 of all STEM cells from being loaded. If ERROR = 1, then we do not look at PANIC signal. The fault coverage is similar to that of the SEM technique, except that in case of false positives, we still need to take appropriate corrective action. In case of the SEM scheme, this value will be overwritten, as R_3 is used only for error detection. However, the STEM technique allows reliable overclocking, achieving higher performance than those systems incorporated with SEM cells.

4.4 Pipeline Design

The basic step in using SEM or STEM cells in a pipeline is to replace all pipeline registers with either one of them. Input clocks are to be constrained in a way, so as to provide fault tolerance capabilities to the pipeline from soft error, as well as, timing error when STEM is the cell of choice. In this section, our discussion is based on the use of STEM cells in place of pipeline registers. Using SEM cells follow straight forward.

Figure 4.4 illustrates how STEM cells are integrated into a processor pipeline. The figure depicts the data and control flow for a five-stage pipeline processor. To the last stage of the pipeline, which is writeback (WB), an extra write buffer, is added. This is to ensure that data written to the register file or memory is always free from timing errors. Every pipeline stage register is replaced with STEM cells, except for the write buffer registers. All error signals from a pipeline stage are logically OR-ed to generate the stage error for that pipeline stage. Global error signal,GERROR, is generated from all pipeline stage error signals, by combining them using another "OR" function. Similarly, global LPANIC signal is generated from individual PANIC signal from all STEM cells. Timing errors may occur once the operating frequency exceeds the worst-case frequency estimate. As explained in the previous section, our data latching scheme of STEM cell guarantees sufficient time before latching values in registers R_2 and R_3. However, data latched in all three registers are susceptible to soft errors that are uniformly distributed in time and space. Here, we explain the pipeline operation for ERROR = 1 and PANIC = 0 (Case II, III, V, VI), as this is the most complicated case. Once an error is detected in any one of the pipeline stages, the global error signal is asserted, and in every stage of the pipeline,

registers R_3 of the STEM cells are not updated with the incoming data. In the next clock cycle, the load backup signal, LBKUP, is asserted, and in each STEM cell, the content of register R_3 is loaded into corresponding R_1 and R_2 registers. After this, the clock to the pipeline is stalled for two cycles, completing the error recovery process.

4.4.1 Clock Control

Clock control monitors the error rate of the pipeline and communicates this error rate information with the clock generator for frequency tuning. Clock generator is connected to the pipeline in a feedback loop. It checks the pipeline error rate with a set target rate, which is programmable. Process of generating new frequency takes up to $10us$, depending upon the speed at which the phase locked loop (PLL) generates the new stable clock signal. Depending on the clock control scheme and error rate sampling scheme chosen [9, 73], clock frequency is adjusted to allow the pipeline to operate below a set target error rate. Once the new clock is generated, the main clock signal, CLK, is switched to that frequency and other clock signals CLK_{1G}, CLK_{2G}, CLK_{3G} are generated by providing the necessary phase shifts to CLK.

4.4.2 Dynamic Frequency Scaling

In the following discussion, we derive the limits of frequency scaling within which a system integrated with STEM cells operates reliably. Pipeline starts execution with a minimal phase shift required between the clocks, and the clock frequency is gradually increased, while satisfying the error rate constraint, as shown in Figure 4.5 (a). To support reliable dynamic overclocking, certain governing conditions need to be met at all times, during pipeline operation. Let us assume that the pipeline operates reliably between the clock frequencies, F_{MIN} and F_{MAX}, governed by time periods, T_{MAX} and T_{MIN} respectively. T_{MAX} is estimated by the worst-case design settings, and is equal to worst-case clock period, T_{WC}. The following clocking constraints decide T_{MIN}. Under overclocking conditions, the following constraints must be satisfied for proper error detection and recovery.

Let D_1 represent the phase shift that needs to be provided for CLK_2, with respect to CLK_1, for soft and timing error mitigation, when the system is clocked with clock period T_{MIN}, , as shown in Figure

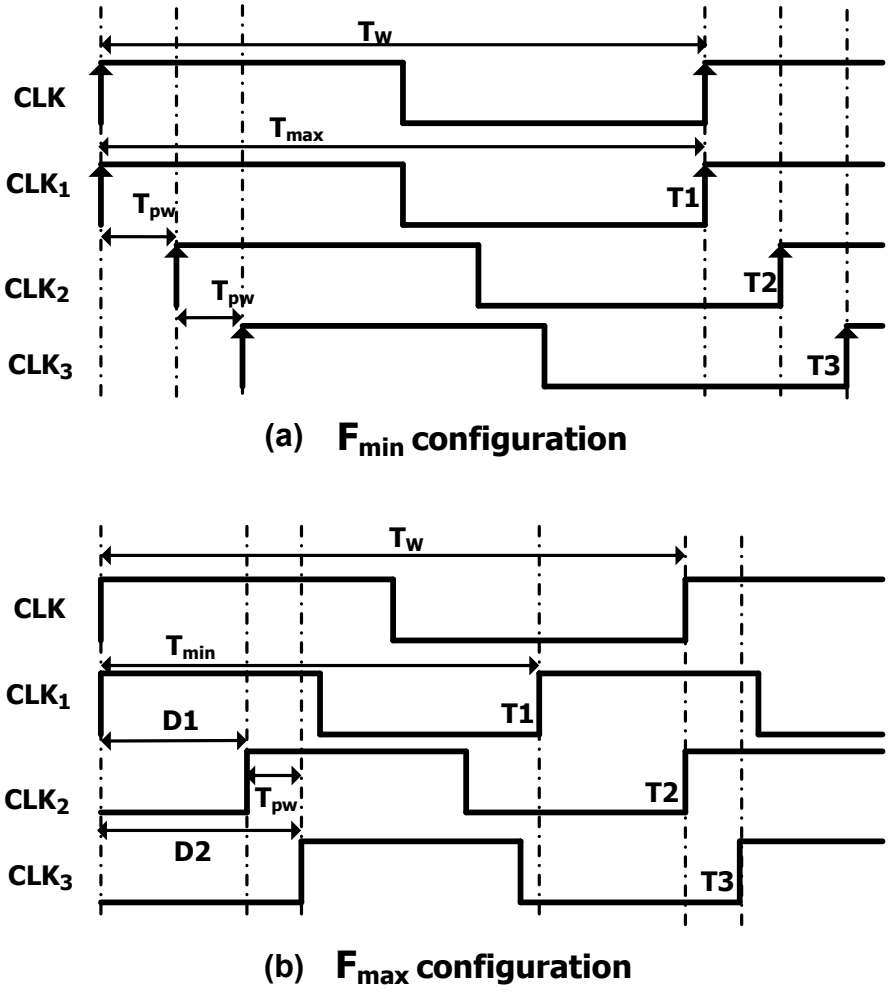

Figure 4.5 Dynamic Frequency Scaling

4.5 (b). Let D_2 represent the phase shift that needs to be provided for CLK_3, with respect to CLK_1, for proper error recovery, when the system is clocked with clock period T_{MIN}. Value of T_{MIN}, satisfying Equation (4.6), corresponds to the maximum frequency at which a system can possibly recover, after a timing error occurs.

$$T_{MIN} + D_1 \geq T_{PD} \tag{4.6}$$

$$D_2 - D_1 \geq T_{PW} \tag{4.7}$$

$$T_{CD} \geq D_2 \tag{4.8}$$

4.4.3 Fixed Frequency Operation

For operating without any run-time optimizations, the frequency of the main clock can be fixed at the desirable operating frequency satisfying the conditions described in Section 4.3. This operating frequency can also be set above the worst case estimate. Under these conditions, systems using STEM cells will see varying error rates based on the executing application. This configuration offer performance better than the system integrated with SEM cells and cannot achieve the best performance as the dynamic frequency adjustment is not allowed. If the frequency is fixed at the worst-case operating frequency, systems integrated with SEM and STEM cells are expected to offer quite high levels of reliability having identical performance levels, when compared to an unprotected system, as there is no overhead incorporated by error detection and correction circuitry on system operating frequency.

4.4.4 Pipeline Error Recovery

In this section, we present the error recovery scheme in detail for a pipeline using STEM cells. Various events involved in the recovery process are illustared with the help of a timing diagram. Figure 4.6 shows how our global error recovery scheme rescues a system when an error happens in the pipeline. The figure also depicts the timing relationship between various control signals required for the recovery process. The following description explains the events that take place during the recovery process for the cases : $\text{ERROR} = 1$, $\text{PANIC} = 0$ and $\text{ERROR} = 0$,$\text{PANIC} = 1$.

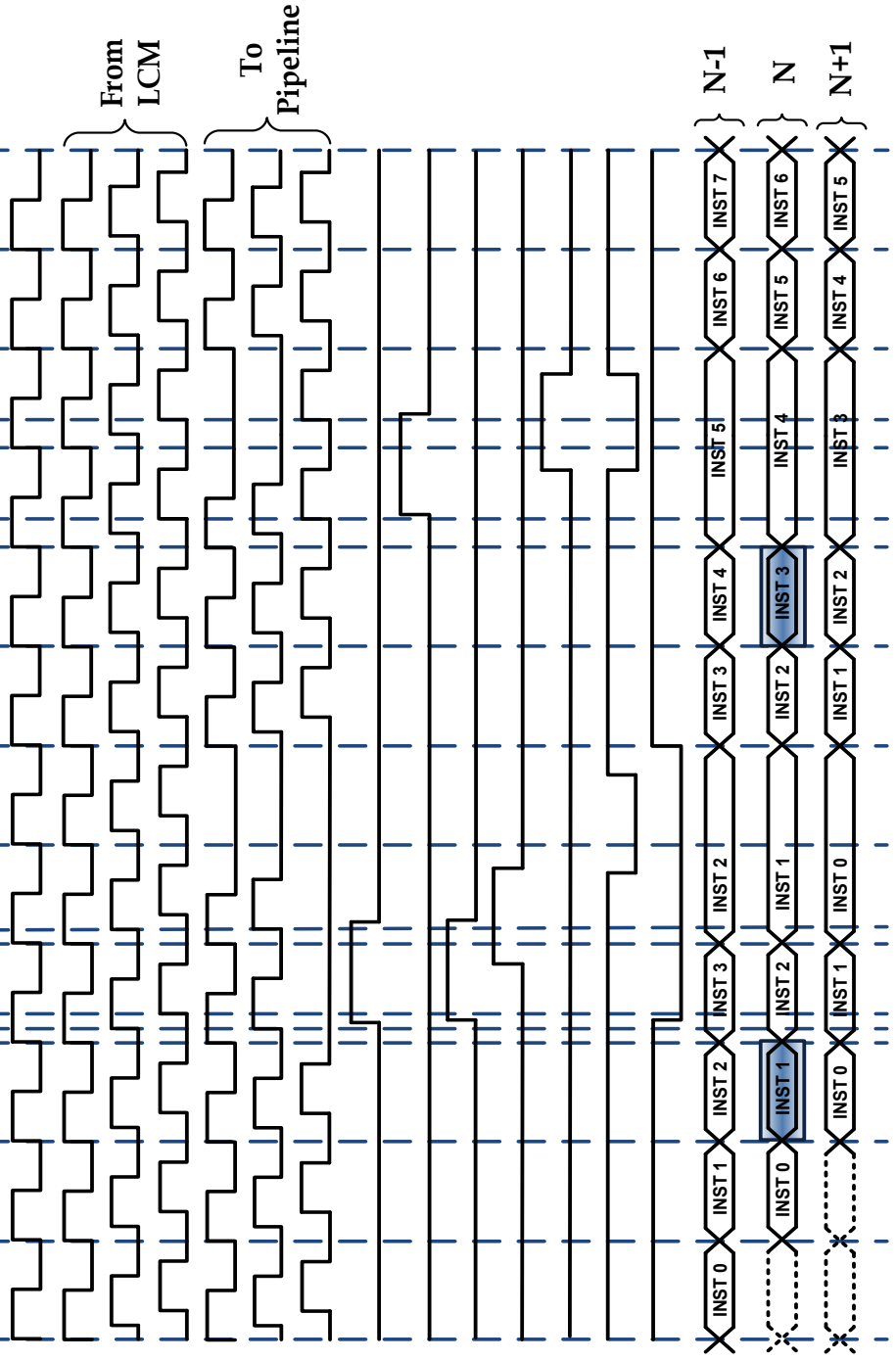

Figure 4.6 Timing Diagram Illustrating STEM Error Recovery Process

Figure 4.6 shows a set of clock signals, CLK_{1G}, CLK_{2G} and CLK_{3G}, that are generated from the main clock signal, CLK, using a LCM. Next, it shows a set of clock signals, CLK_{1P}, CLK_{2P} and CLK_{3P}, that are routed to the pipeline. These clock signals, which are gated versions of CLK_{1G}, CLK_{2G} and CLK_{3G} respectively, are stalled in a manner that enables the pipeline to recover from different error scenarios. Signal ERROR_N indicates an error happening in the pipeline stage N. ERROR signals from all the pipeline stages are OR-ed together to generate the global error signal, GERROR, which is latched in the clock control unit. Similarly, to initiate global recovery when ERROR = 0,PANIC = 1 i.e. for panic cases, all stage PANIC signals are used.

For case, ERROR = 1, PANIC = 0, once an error is detected, the very next clock edge of clock signal CLK_{3G} is gated and in the next cycle, LBKUP signal is asserted high for one clock cycle. In the same clock cycle, using CLK_{1P} and CLK_{2P}, recovery data from register R_3 is loaded back into registers R_1 and R_2. During the next cycle, all clock signals, CLK_{1G}, CLK_{2G} and CLK_{3G}, are clock gated to give the pipeline sufficient time for re-computation. Clock gating is achieved through control signals CLKSTALL_{12} and CLKSTALL_3, which are generated by the clock control unit.

For case, ERROR = 0, PANIC = 1, once an error is detected, the very next clock edge of clock signals, CLK_{1G} and CLK_{2G}, is gated and in the same cycle, LPANIC signal is asserted high for one clock cycle. Also, recovery data present in register R_2 or R_1 is loaded back into register R_3 during this cycle using CLK_{3P}. Clock gating of clock signals, CLK_{1G} and CLK_{2G} is achieved through control signal CLKSTALL_{12}, which are generated by the clock control unit.

To illustrate our error recovery mechanism, error occurrences corresponding to cases, ERROR = 1,PANIC = 0 and ERROR = 0,PANIC = 1, are highlighted in Cycle 3 and Cycle 8 respectively. First error occurs during the execution of INST 1 of pipeline stage N. This event triggers the error recovery mechanism that spans from Cycle 4 to Cycle 6. During Cycle 4, data is loaded into register R_1 and R_2 from the corresponding stage golden register R_3. Pipeline is allowed to perform the computation during Cycles 5 and 6. Results are again checked at the end of Cycle 6. Since no error is detected in this cycle, normal pipeline operation resumes. From the waveforms, we can see that on error detection, the entire pipeline goes back by one instruction. Second error occurs during the execution of INST 3 of pipeline stage N. This event triggers the error recovery mechanism that spans from Cycle 9 to Cycle 10. During

Cycle 9, data is loaded into register R_3 from the corresponding stage golden registers, R_1 or R_2. At the end of Cycle 9, no new results are stored in all stage registers R_1 and R_2. New results are latched and checked at the end of Cycle 10. Since no error is detected in this cycle, normal pipeline operation resumes. In this case, the pipeline does not roll back and just the corresponding stage R_3 register is updated. As we can see from Figure 4.6, for 12 clock cycles, pipeline computes only 8 instructions as 4 clock cycles are accounted for the error recovery process.

4.4.5 Performance Analysis

A key factor that limits frequency scaling is error rate. As frequency is scaled higher, the number of input combinations that result in delays greater than the new clock period also increases. The impact of error rate on frequency scaling is analyzed as follows:

Let t_{wc} denote the worst-case clock period. Let t_{ov} denote the clock period after overclocking the circuit. Let n be the number of cycles needed to recover from an error. Let us assume that a particular application takes N clock cycles to execute, under normal conditions. Let t_{diff} be the time difference between the original clock period and the new clock period. Then the total execution time is reduced by $t_{diff} \times N$, if there is no error. Let us assume that the application runs at the overclocked frequency of period t_{ov} with an error rate of $k\%$. To achieve any performance improvement at this frequency, Equation (4.9) must be satisfied. It states that even after accounting for error recovery penalty, execution time required is still less than that required for worst-case frequency operation.

$$N \times t_{ov} + n \times N \times k \times t_{ov} < N \times t_{wc} \tag{4.9}$$

$$k < \frac{(t_{wc} - t_{ov})}{n \times t_{ov}} \tag{4.10}$$

For the STEM technique, an error can happen in five different scenarios, as mentioned in Table 4.2, and also the error recovery penalty paid is not the same for all the cases. If we assume that all these error scenarios are equally likely, then the average error penalty in cycles is: $n = \frac{4\times3+1\times1}{5} = 2.6$. According to Equation (4.10), for a frequency increase of 15%, the error rate must not be higher than 5.76%, for the STEM technique to yield no performance improvement. For error rates less than *1%*, a frequency

increase of *2.6%* is enough for the STEM scheme to have a performance improvement over non fault tolerant designs.

4.4.6 Impact of Process Variation

Process variation can alter gate delays of a combinational circuit and hence can alter the distribution of path delays. Both our techniques, SEM and STEM, sample data at three different times, namely T_1, T_2, and T_3, after the computation is completed. Paths that can affect this sampling window are short paths that are present in the circuit. These paths are already padded with buffers to allow sufficient time for data sampling. In presence of process variation, in particular, only sampling interval T_3 can be affected. Since, these paths are present locally, inter-die variation has no impact on these paths. To mitigate intra-die variation, conservative padding of buffers needs to be done on the short paths to make sure that same data is sampled. This will only result in slight increase of the circuit area.

4.4.7 Overheads

One of the main overheads incurred by our schemes is fixing the circuit contamination delay to a required value. Increasing this delay involves rapid increase in silicon area, as buffers need to be inserted in the short circuit delay paths. This problem has to be addressed from different design perspectives that include developing new synthesis algorithms and delay buffer design with minimal area consumption. Both SEM and STEM cells require metastability mitigation circuits, as flip-flops may enter a metastable state when overclocked, or when a soft error reaches the registers during the latching window. We envisage the incorporation of a metastability detection circuit, similar to the one developed in [27].

4.5 Local Clock Management

Reliable dynamic overclocking technique has been proposed earlier, in [9, 73], to improve system performance by tuning the clock frequency beyond the conservative worst-case clock period. It requires a dynamic phase shift (DPS) between the clock signals to support aggressive dynamic clock tuning. This implies the system has to be stalled for a few cycles whenever the clock frequency is

changed. This is done for two purposes: 1) the time needed to switch to a new frequency and 2) to appropriately phase shift for the respective clock signals, as mandated by the timing constraints. At higher frequencies, controlling the phase shift precisely is a challenge and often the phase shift step size can restrict the possible system operating frequencies. Also, SEM and STEM cells would require three clock distribution networks (CDN), thus increasing clock routing complexity and resources needed for it. The increase in clock resources would in turn increase system energy consumption. To avoid dynamic phase shift between the clock signals, we incorporate a constant phase shift (CPS) between the clocks that are configured to run between frequencies corresponding to the time periods, T_{MAX} and T_{MIN}.

Let us consider a case where $T_{MAX} = 10ns$, $T_{MIN} = 6ns$ and $T_{CD} = 4ns$. Considering a dynamic phase shift between the clock signals, when we scale the system clock period down to $8ns$, then we need to provide a phase shift of $2ns$. Similarly a phase shift of $3ns$ is required for a $7ns$ clock period. Since the circuit contamination delay is increased to $4ns$ to aggressively clock the system, computed data will remain stable for $(T+4)ns$, where T is the current operating frequency of the system. Instead of requiring a dynamic phase shift along with frequency scaling, we provide a constant phase shift of at most T_{CD} at all times.

With a CPS scheme, aggressive systems can be built with a single CDN, thus tremendously reducing the amount of clock resources required. Using a constant phase shift that is valid for all frequencies between F_{MIN} to F_{MAX}, allows phase management to be decoupled from frequency switching. This completely eliminates the constraint imposed by the phase shift step size in DPS, allowing CPS to support more operating configurations than DPS. Local clock managers (LCM) handle the phase shift management in CPS. Figure 4.7 shows a conceptual implementation of a LCM that can distribute CLK_1, CLK_2, and CLK_3. Employing CPS, delay values D_1 and D_2, are set to constant values to satisfy the timing constraint explained earlier in Section 4.4. Figure 4.7 also depicts how these LCMs could be integrated into an H-tree clock distribution network.

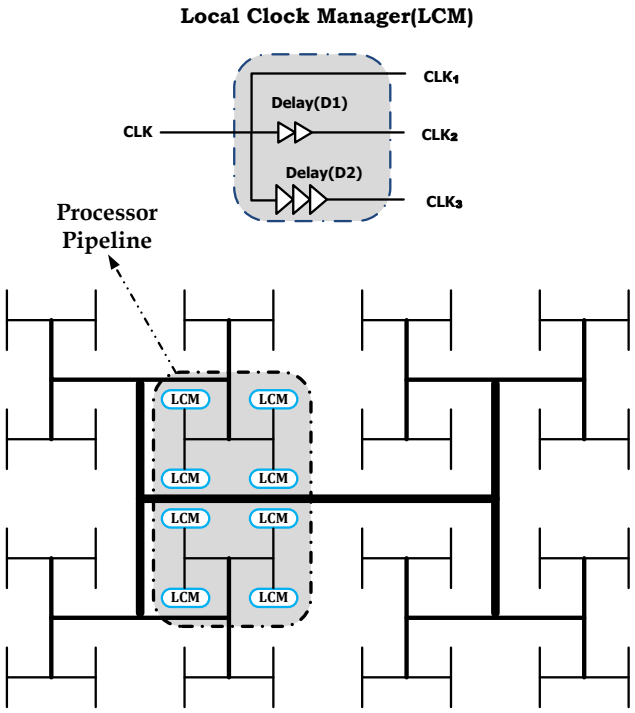

Figure 4.7 Local Clock Phase Management with Single Clock Routing

4.5.1 Case Study: Local Clock Management using buffers.

For generating clock signals required by SEM and STEM schemes locally, we present a possible implementation using buffers. We perform this study using 45*nm* spice models distributed by Nangate Technologies [55]. Post layout spice models containing parasitic information are used. Area overhead, incurred for generating constant phase shift clocks, is analyzed by applying a load of 128 STEM cells. From this study, we observe that, even for a 2.5*ns* phase shift, only 14 clock buffers are needed. This overhead is much lower than a having a second and third clock tree networks. Study results are summarized in Table 4.3.

DELAY(*ns*)	BUFFERS	DELAY(*ns*)	BUFFERS
1.0	6	1.5	8
2	11	2.5	14

Table 4.3 Number of Buffers vs Delay

Figure 4.8 shows the transient response of clock signals generated with a LCM designed with delay buffers. A load of 32 STEM cells is applied at room temperature (25°C). From this we notice that, sufficient phase shift between the clock signals is provided across all the cycles. Also the rise time and

the fall time of CLK_2 and CLK_3 signals are around $569ps$ and $222ps$ respectively, as illustrated in the Figure 4.8. Table 4.4 shows the rise time and the fall time for LCM outputs at different temperatures. From these values, we notice that, all the clock signals track each other very well in terms of fall time even at very high temperatures. They can also be made to have similar characteristics in terms of rise time at the expense of designing sophisticated buffers.

OUTPUT	70°C		75°C		80°C	
	risetime	falltime	risetime	falltime	risetime	falltime
CLK_1	$200ps$	$200ps$	$200ps$	$200ps$	$200ps$	$200ps$
CLK_2	$875ps$	$285ps$	$913ps$	$292ps$	$956ps$	$300ps$
CLK_3	$861ps$	$278ps$	$898ps$	$286ps$	$935ps$	$293ps$

Table 4.4 Temperature Effects on LCM Outputs

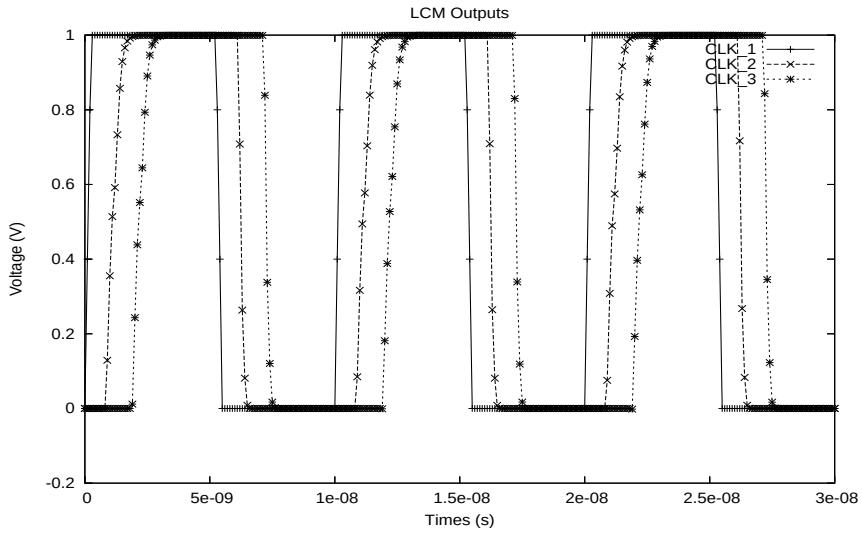

Figure 4.8 Transient Response of LCM Outputs

Impact of process variations : In this case study, we presented a simple approach, using delay buffers for managing phase relationship between clock signals that are required for SEM and STEM cells. It is presented to give a conceptual picture of our CPS scheme. In presence of process variations, delay of these buffer cells can vary and hence the phase difference between the clocks can also be altered. In a CPS scheme, it is important to realize the difference in phase difference requirements between CLK_1 and CLK_2, and CLK_2 and CLK_3. In CPS, the phase difference between CLK_1 and CLK_2 at max needs to be D1 (where D1 $\gg T_{PW}$) and for CLK_2 and CLK_3, it has to be at least T_{PW}. Because of process variations, any variation in D1 will alter only limit the frequency scaling limits and not the soft error

mitigation capabilities of our techniques. For CLK_2 and CLK_3 phase relationship, post fabrication delay tuning approaches [57] or using tunable/programmable delay elements [38] can be used to mitigate the effect of process variation (intra-die only) on our techniques.

4.6 Experiments and Results

In this section, we present our results based on the experiments conducted on a two stage arithmetic pipeline and a five stage DLX in-order pipeline processor, wherein pipeline registers are augmented with our fault detection and correction circuitry. It should be pointed out that soft error rate depends on electrical masking and logical masking which inherently masks transient faults. Across process technology nodes, logical masking remains the same but the electrical masking differs. This is because electrical properties of gates differ between technology nodes. Electrical masking could diminish significantly as feature size decreases. In order to model electrical masking that happens in nanometer technologies, we have implemented system in $45nm$ technology. Also fault mitigation capabilities exhibited for same T/T_{PW} ratio but for different clock frequencies and pulse widths are expected to be same. For example, fault mitigation capabilities exhibited for $100ps$ pulse at $1GHz$ are expected to be the same as those offered for $1000ps$ pulse at $100MHz$. As presented in Section 1, soft error pulses have width in the range of $500ps - 900ps$ in the $90nm$ process. In our experiments, we have considered this *extreme-case* pulse width for $45nm$ process to demonstrate transient fault detection and correction capabilities of our techniques. Without lose of generality, our results can be improved by relaxing the soft error pulse width (T_{PW}) constraint and insights derived from our experiments will still hold.

4.6.1 Experimental Methodology

To estimate the performance gains and fault tolerant capabilities offered by SEM and STEM techniques, simulations are carried out on a two stage arithmetic pipeline. This circuit performs a 64-bit addition in the first stage and a 32-bit multiplication in the second stage. Adder output is fed to the multiplier as multiplicand and multiplier. RTL level models are developed for both the circuits, and are synthesized using the $45nm$ OSU standard cell library [72]. Timing-annotated gate level simulations are then carried out by extracting timing information in standard delay format (SDF), and back

annotating them on the design.

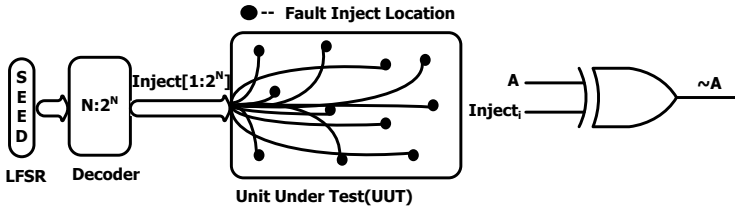

Figure 4.9 Fault Injector Framework

Figure 4.9 illustrates our fault injection methodology. The working of our fault injector is as follows: A total of 2^N (N being 7 in our experiments) fault injection test nodes that are spread uniformly across the area of the logic circuit are selected. To make sure that our injected fault has indeed produced a SET, we modified the circuit netlist to insert XOR gates at all selected nodes, as shown in Figure 4.9. If a location i is chosen for fault injection, $Inject_i$ is made high to invert the signal A driven by the fault injection node i. Out of 2^N locations, one location is chosen randomly for fault injection at a time, by using the output of a N-bit random number generator. For our experiments, we used a linear feedback shift register (LFSR) for generating the N-bit random number. Final fault location is then selected with the help of a N:2^N decoder.

4.6.2 Results for Arithmetic Pipeline

For the arithmetic pipeline, from static timing analysis reports, we estimated the value of T_{MAX} to be $9ns$. For aggressively clocking the design, we increased the contamination delay to $3ns$. Combinational area of the arithmetic pipeline is increased by 38%, for fixing the contamination delay to $3ns$. This area overhead is in addition to the area consumed by SEM/STEM cells. Pulses of varying widths ranging from $500ps$ to $900ps$ are injected in the unit under test (UUT). Each cycle, results are checked for correctness after the computation is over to ensure that the recovery mechanism works. Whenever recovery is triggered, we logged the occurrence of an error.

For evaluating STEM technique, we performed our experiments for a set error rate target of 1% over 10000 cycles. During run time, the number of errors that happened during a sampling interval is communicated to the clock controlling unit at the end of each interval. The clock controlling unit

	STEM (MAXOC)			STEM (DYNOC)			STEM (NOOC)			SEM			TMR		
	TE	Transient Faults		TE	Transient Faults		Transient Faults			Transient Faults			Transient Faults		
		Injected	Detected		Injected	Detected	Injected	Detected		Injected	Detected		Injected	Detected	
RUN1	14	2031	432	14	2033	421	2030	325		2031	334		2026	256	
RUN2	14	2031	450	12	2033	414	2025	315		2028	323		2026	268	
RUN3	14	2031	449	15	2032	424	2025	307		2030	311		2034	273	

Table 4.5 Fault injection results for the arithmetic pipeline

makes a decision based on the error rate, during the previous sampling interval, and the set target error rate. We considered a linear control scheme for switching clock frequency between the worst-case clock frequency, F_{MIN} and the overclocked frequency, F_{MAX}. For our design, T_{MIN} is set at $7ns$. This range is divided into 32 steps, and if the error rate is less than 1%, clock frequency is increased by one step size, otherwise it is decreased. Our fault injection results for the arithmetic pipeline are presented in Table 4.5. We initialized the LFSR with different seeds, and the fault injection results are presented for three different runs.

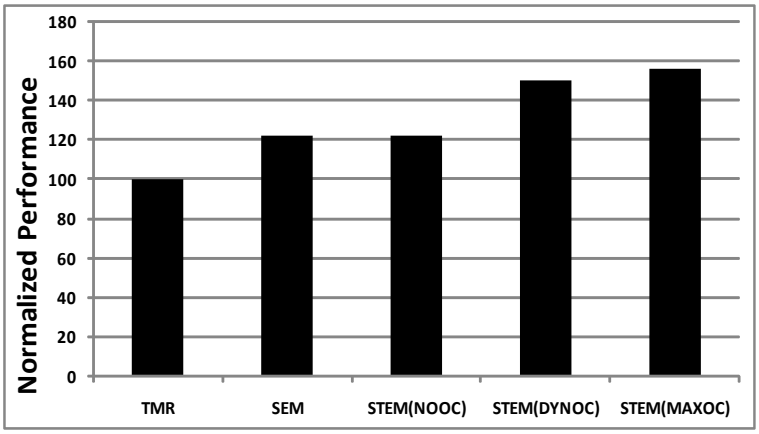

Figure 4.10 Normalized Arithmetic Pipeline Execution Time

We configured the arithmetic pipeline designed with STEM cells to operate in three different modes. They are no overclocking (NOOC), wherein $T_{MAX} = T_{MIN} = 9ns$, maximum overclocking (MAXOC), wherein $T_{MAX} = T_{MIN} = 7ns$, and dynamic overclocking (DYNOC), wherein $T_{MAX} = 9ns$ and $T_{MIN} = 7ns$. For DYNOC mode, we started with a low frequency setting. For TMR system, worst-case frequency, T_{MAX}, is set at $11ns$. We evaluate SEM scheme at a constant clock period of $9ns$. Performance improvements offered by both SEM scheme and different modes of STEM are shown in Figure 4.10. From this, we can see that DYNOC mode offers 49% improvement over TMR, while MAXOC mode offers 55% improvement. Performance of NOOC mode is comparable to that of SEM and SEM offers 23% performance improvement over TMR. From Table 4.5, we can see that fault masking rate is high in TMR design when compared with SEM and STEM designs. This is because; its operating frequency includes the phase shifts of the clocks and voter delay. Hence, TMR operates with a longer clock period compared to SEM and STEM, resulting in more SET pulses attenuating before

reaching the latching window.

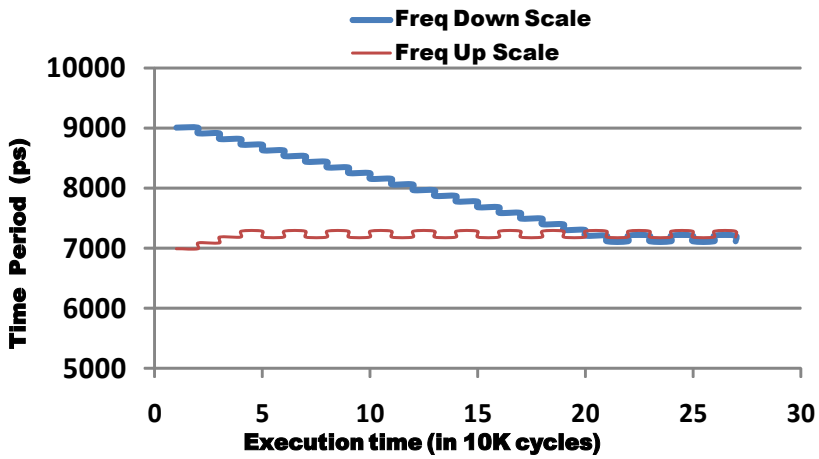

Figure 4.11 Dynamic Frequency Scaling

Figure 4.11 shows the transient response of our arithmetic pipeline configured with STEM cells in DYNOC mode. It shows how the frequency of the system changes dynamically for high to low frequency switching mode and low to high frequency switching mode. From figure, we see that the system tunes it optimal clock period to 7.2*ns*. For high to low frequency switching mode, the optimal value is reached after 4 sampling intervals, for low to high mode, optimal value is reached after 22 such intervals. High to low frequency mode ran more operations than the other mode, and the performance difference was about 3%. Even though contamination of the system is increased to 3*ns*, the available dynamic frequency range is restricted to 2*ns* due to pulse width considerations.

4.6.3 Results for DLX Processor

We also simulated three different micro benchmarks to evaluate the performance improvement and fault coverage of both SEM and STEM (DYNOC mode) schemes on a five stage in-order pipelined processor. This processor, implemented in 45*nm* technology, is based on the DLX instruction set architecture. First application, RandGen, calculates a simple random number generation to give a number between 0 and 255. The MatrixMult application multiplies two 50x50 integer matrices and the BubbleSort program implements bubble sort algorithm on 5,000 half-word variables. Here, we followed the same fault injection strategy and clock control used for two stage arithmetic pipeline. For each benchmark, processor state is checked to verify the correctness of the computed results after

simulation. From timing reports, the worst-case clock period, T_{MAX}, is estimated as $6ns$. Contamination delay is increased by $3ns$ and the system operates at an optimal clock period of $4ns$. At chip level, area overhead incurred is less than 15% for the processor because significant area consumption of the system comes from the memory system. The results for the three different benchmarks are presented in Figure 4.12, showing relative execution times for conventional TMR, SEM and STEM schemes. From this, we found that SEM offers 26.58% performance improvement over TMR and STEM offers 27.42% over SEM.

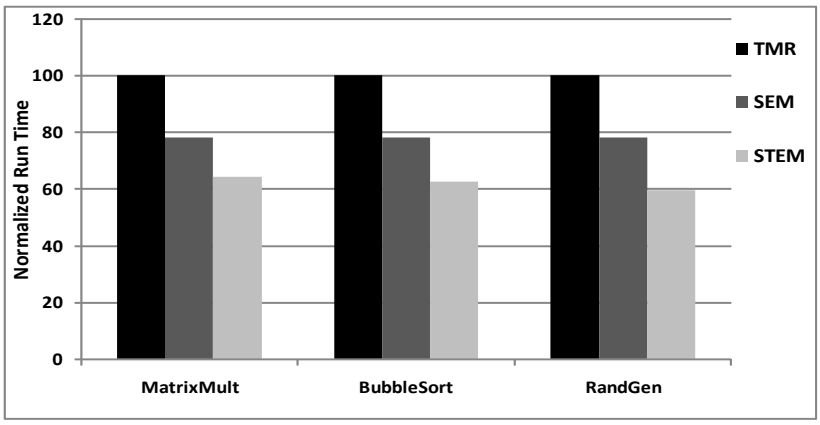

Figure 4.12 DLX Execution Rime for various benchmarks

4.7 Related Work

A multitude of fault tolerant architectures for tolerating soft errors have been developed in the past by the research community. Many schemes incur performance overhead even during error-free operation and do not support timing speculation. LEON-FT processor [29] uses TMR approach and triplicates every flip flop in the processor and incurs a 100% area overhead. The REESE architecture [34] takes advantage of spare elements in a superscalar processor to perform redundant execution. DIVA [5] uses spatial redundancy by providing a separate, slower pipeline processor alongside the fast processor. Reunion [71] exchanges control and data flow information between the cores to speed up execution, while leveraging the redundancy to provide partial fault coverage. These approaches trade performance and power for achieving soft error fault tolerant capabilities and none of these approaches support timing speculation. Our technique, SEM, is the first technique in this class, that effectively

exploits timing speculation and enhances performance without any logic duplication.

Multiple data latching schemes: Multiple clocking of data has been widely used to combat the issue of soft errors in combinational logic [3, 48, 59, 52, 23, 66]. These approaches involve latching data at different times or latching data and its delayed version using a single clock. Authors in [23] optimize a temporal TMR framework that is presented in [48]. Even with this approach, the clock frequency at which the circuit can be clocked must include the delays of combinational logic, double the pulse width under consideration and also the delay incurred for the majority voting. Moreover, this technique doesn't detect false errors.

More recently, BISER, a DMR framework, using C-elements to tolerate soft errors has been proposed [52]. It offers two techniques - 1) using logic duplication (DMR in space) and 2) time shifted data (DMR in time). Also, using C-elements, authors in [66] propose soft error hardened flip flops to tolerate wide error pulses or hard errors and applies DMR in both time and space domains. As we explained, SEM requires only two comparison operations for error detection, as opposed to full voting of multiple samples. With our distributed and temporal voting, SEM completely eliminates the error recovery penalty paid for false errors. This is a big contribution when compared to previously proposed techniques, which use multiple data latching, by employing either DMR in space/time or temporal TMR. Moreover, the fault detection overhead is taken off the critical path, which enables systems to have performance levels comparable to a non-fault tolerant system in error-free operation. Previously proposed schemes, using multiple data latching, employ either DMR or TMR [48, 52, 23, 66] and pay penalty for false errors. SEM effectively combines timing speculation with temporal TMR framework and completely removes the overhead paid for false errors. Also, when an error is detected, unlike previous DMR solutions, we need not perform re-computation to recover the system state. This is a salient feature of our approach, which has *in-situ* fast error recovery. Hence, SEM does not require any checkpointing, thereby saving the time and space required to store the system state. Also, it is estimated at static power is comparable to dynamic power with nano-sized transistors. Hence, providing fault tolerance for energy constrained systems like embedded systems, with logic replication is not a viable option. Systems designed with SEM cells improve reliability without logic replication.

As the performance margin between designing circuits to meet worst-case constraints verses typically-

case constraints has become increasing significant, a new area of research emerged that aimed to use fault tolerance to allow synchronous circuits to perform reliably at the highest possible clock frequencies. A good summary of the principles of this new area is presented in [4]. The methods reviewed in [4] are good initial attempts at pushing circuits close to safe limits for increasing performance; however, most approaches are not feasible in practice. For example, in [81] a method for increasing clock frequency is proposed by adding redundancy and clocking each stage of a pipeline at a phase shift of the original clock. However, the overheads associated with the redundancy and clocking resources of this approach were not addressed for deployment into real-world designs. A more recent approach is presented in [82], where the clock frequency of a simple in-order processor was tuned dynamically. An extra delay chain with a worst-case propagation delay greater than the critical path of the processor was used to guide frequency tuning. The delay chain was monitored constantly and detection of errors were used to adjust the processor's frequency to avoid errors within the processor as operating conditions changed. A benefit of this approach is that it avoided errors within the processor, thus eliminating the performance penalties often associated with error recovery circuitry. However, this approach can adapt to changes in operating conditions, but can't exploit data dependent circuit delay. This attribute of circuit delay has been exploited in SPRIT^{3}E to further extend the operation boundaries often set by worst case estimates [9, 73].

Razor [25, 27] and SPRIT^{3}E [9, 73] architectures employ timing error tolerance techniques to operate beyond worst-case limits. Both these architectures use temporal fault tolerance by replicating critical pipeline registers to improve performance beyond worst case limits. While Razor focuses on achieving lower energy consumption by reducing supply voltage in each pipeline stage, SPRIT^{3}E improves performance of a superscalar processor by reliably overclocking the pipeline. Other closely related works is Paceline [32], which employs leader-checker configuration in a chip multiprocessor system and tolerates both timing and soft errors. Systems designed with STEM cells looks to improve the reliability and performance by enabling reliable overclocking without logic duplication. Table 4.6 summarizes how our schemes differs from previously proposed techniques.

Design	LD	SEP	AC	ES
Razor	✗	×	×	√
SPRIT^{3}E	✗	×	√	×
Paceline	√	√	×	×
SEM	✗	√	×	×
STEM	✗	√	√	×

Table 4.6 Comparison with other schemes in terms of Logic Duplication (LD), Soft Error Protection (SEP), Aggressive Clocking (AC) and Energy Savings (ES)

4.8 Conclusion

In this chapter, we have presented two efficient soft error mitigation schemes, SEM and STEM, considering the approach of multiple clocking of data for tolerating soft errors in combinational logic. Both these techniques remove the error detection overhead from the circuit critical path. These specialized register cells provide near 100% fault tolerance against transient faults. Our schemes tolerate fast transient noise pulses, which is the principal characteristic of SETs. Both our schemes have no significant performance overhead during error-free operation. SEM cells are capable of ignoring false positives and recovers from errors using *in-situ* fast recovery avoiding recomputation. STEM cells has soft error mitigation characteristics similar to SEM. STEM cells allow reliable dynamic overclocking and are capable of tolerating timing errors as well. We also propose a scheme to manage phase shifts between clocks locally with constant delay values. Such an approach increases the possible frequency settings for aggressively clocked designs and also minimizes the clock routing resources.

CHAPTER 5. TASK SCHEDULING

5.1 Task scheduling

Power consumption has become an important concern in the design of computer systems today due to battery life considerations and environmental concerns. Dynamic voltage and frequency scaling (DVFS) is an effective method to control the trade-off between performance and power consumption of computing systems. DVFS schemes dynamically scale the voltage and frequency of the CPU to provide variable amount of energy to process the workload. In general, scaling down voltage or frequency of the processor, results in reducing dynamic power consumption. Intel SpeedStep, Advance Micro Devices (AMD) PowerNow!, and Transmeta LongRun [64, 2, 28] are few examples for DVFS in real processors. Recent developments include providing support for deeper integration of DVFS like adding more power saving states (like deep p-states and c-states) [65].

Task scheduling play crucial role in multiprocessor systems which are widely deployed in high performance computing systems like super computers, computer clusters etc. Task scheduling algorithms for the multiprocessor system are usually developed based on the task graph. DVFS technique and task scheduling can be combined to form a two phase process for effective resource usage and energy consumption minimization. In this approach, first phase involve schedule generation, where tasks from the given task graph are scheduled on a given multi-processor system by attempting to maximize objective function which include minimizing makespan. In second phase, which involve slack reclamation, the output of first phase is post processed to minimize the energy consumption using DVFS technique and also without violating the deadline constraint.

The existing slack reclamation methods based on DVFS technique has a major shortcoming as they map each task to a single frequency among the available discrete set of (V, f) pairs of underlying DVFS platforms. As shown in [63], using only single frequency results in uncovered slack time, which reflect

into energy wastage. To minimize uncovered slack time, a novel approach, multiple voltage-frequency selection dynamic voltage frequency scaling (MVFS-DVFS), has been proposed. MVFS-DVFS uses linear combination of frequencies for slack reclamation. With MVFS-DVFS technique, authors have shown that energy consumption can be pushed to the near optimal point of using (V, f) pairs supported by traditional DVFS schemes.

5.1.1 Chapter Contribution

For a given task, optimal energy under traditional DVFS schemes is achieved by using a single frequency if the frequency of the system is assumed to be continuous. But real processors work with a discrete set of freqeuncies and voltages, which we name as available DVFS (V, f) spectrum. Linear combination of frequencies has been proposed to achieve near optimal energy for systems operating with discrete frequencies and voltages[63]. It is important to note that (V f) pairs available from the DVFS spectrum are set by conservative estimates.

In this chapter, we propose a new slack reclamation algorithm, aggressive dynamic and voltage scaling (ADVFS), using reliable and aggressive systems. ADVFS exploits the enhanced voltage frequency spectrum offered by reliable and aggressive designs for improving energy efficiency. We provide formal proofs to show that optimal energy for reliable and aggressive designs is achieved either by using single frequency operation or by linear combination of frequencies. We evaluate the performance of ADVFS using random task graphs and our simulations indicate that significant energy savings are possible under ADVFS when compared with MVFS-DVFS and also continuous traditional DVFS.

5.1.2 Chapter Organization

The chapter is organized as follows. In Section 5.2, we present details on how to manage (V, f) spectrum of reliable and aggressive systems and its impact on slack reclamation phase of task scheduling algorithms. We present our algorithm, ADVFS, in Section 5.3 along with formal proofs. Our simulation results are presented in Section 5.4.

5.2 Energy Management Using Reliable And Aggressive Framework

5.2.1 Managing the (V, f) spectrum

Available (V, f) spectrum for reliable and aggressive systems is enhanced as additional voltage and frequency swings are allowed beyond the conservative guard bands. Figure 5.1 presents the changes to the (V, f) spectrum, which is available for traditional DVFS algorithms. The dots on the solid diagonal lines in Figure 5.1(a) and 5.1(b) represent the limited (V, f) pairs available to traditional DVFS. In this scenario, system voltage can be varied from V_{min} to V_{max}, while the frequency can be varied from f_{min} to f_{max} in discrete steps (four possible pairs are shown in the figures). Using reliable and aggressive designs, (V, f) space is expandable by two orthogonal mechanisms: Aggressive Voltage Scaling (AVS) [25] and Aggressive Frequency Scaling (AFS) [9, 73, 10]. Figures 5.1(a) and 5.1(b) illustrate enhanced (V, f) spectrum for AFS and AVS respectively.

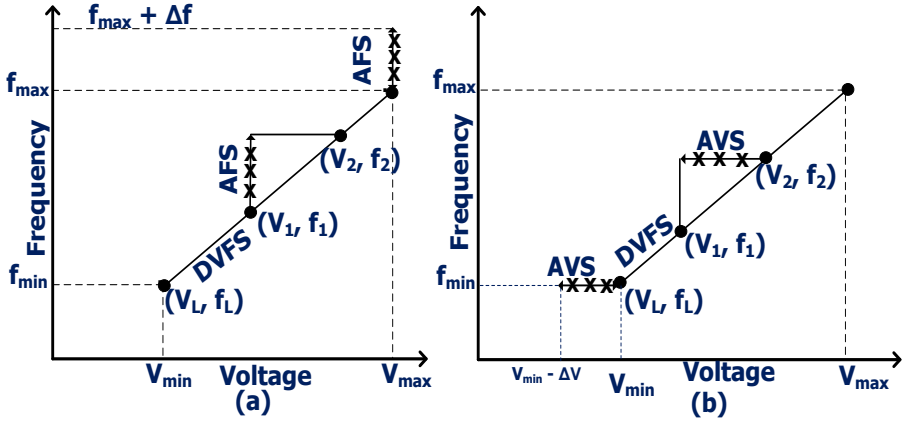

Figure 5.1 (a) AFS enhances the worst case frequency of DVFS (V, f) pair by Δf
(b) AVS decreases the worst case voltage of DVFS (V, f) pair by ΔV

At each (V, f) pair of traditional DVFS, the voltage range available under AVS is V - ΔV to V. Similarly, the frequency range available under AFS is f to $f + \Delta f$. This range can be varied by managing the contamination delay (T_{CD}) of the system. As an example, using AFS, starting from a traditional DVFS pair, (V_L, f_L), the circuit's T_{CD} should satisfy the constraint given by Equation 5.1 to give a frequency range of Δf. Similar area overheads can also be paid under AVS to allow voltage swing beyond the conservative guard band. The area overhead associated with adjusting a circuit's T_{CD} to satisfy this equation comes in addition to the timing overhead associated with error detection and correction

circuitry.

$$T_{CD} \geq 1/f_L - 1/(f_L + \Delta f) \tag{5.1}$$

5.2.2 Comparing AFS and AVS

For reliable and aggressive systems, the more aggressively the system is pushed beyond worst-case constraints, the higher the probability of timing errors occurring. These error rates can become a bounding constraint on performance/power gains of reliable and aggressive systems. Because AFS and AVS use different physical mechanisms to achieve the desired operating point, the observed error rates can differ. A given target (V, f) pair can be reached in two ways: (1) overclocking using AFS from (V, $f - \Delta f$) and (2) using AVS from $(V + \Delta V, f)$. Figure 5.2(a) and Figure 6.3(b) illustrate the nature of the timing error rates observed for AFS and AVS.

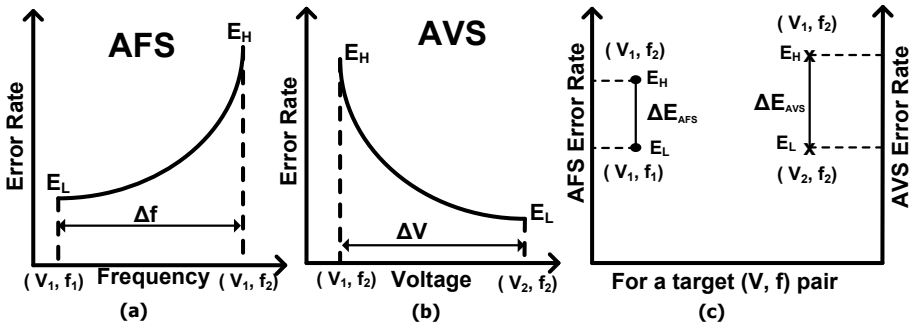

Figure 5.2 (a) Timing error profile of AFS (b) Timing error profile of AVS (c) Depicts the difference in observed timing error rate for AFS and AVS

For example, (V_1, f_2) pair can be reached by overclocking from (V_1, f_1) or undervolting from (V_2, f_2) and observed timing error profile is shown in Figure 5.2(a) and Figure 5.2(b) respectively. For a given set of operating conditions and a given target (V, f) pair, the differences in error rates of AFS and AVS need can be characterized and be compared to determine the most appropriate approach, as conceptually illustrated in Figure 5.2(c). Figure 5.2(c) represents the difference in timing error profile (bounded between E_L and E_H) observed for AFS and AVS in reaching a target pair - (V_1, f_2) from corresponding worst-case voltage and frequency pairs. In most cases, error rate lower limit i.e E_L of AFS and AVS match while upper limit i.e E_H can differ. Assuming (V_1, f_1) and (V_2, f_2) to be adjacent

pairs of traditional DVFS spectrum, it is safe to assume that difference in timing error rates of AFS and AVS to match i.e. $\Delta E_{AVS} = \Delta E_{AFS}$. It is because circuit characteristics differ only marginally for adjacent (V, f) pairs and also the circuit will have same critical paths, under both AFS and AVS that can cause a timing error.

5.2.3 Impact on slack reclamation

To study the impact of AFS and AVS on slack reclamation phase, we introduce the following notation. We denote a given task with (T, K), where T is the task deadline and K is the number of clock ticks required for the task execution. If system support continuous voltage and frequency, (V, f) pair for optimal enery consumption is calculated as follows. Frequency of the optimal (V, f) pair, f_{opt}, is given by 5.2.

$$f_{opt} = K/T \tag{5.2}$$

It is important to note that voltage and frequency of the traditional DVFS spectrum are one-to-one mapped. This implies, once frequency is known, voltage can also be known. Voltage of the optimal (V, f) pair can be calculated once f_{opt} is known and we denote this voltage as V_{opt}. But most of the system support discrete voltage and frequency and realization of is not possible. As pointed out in [63], linear combination of frequencies can be used to reach optimum energy consumption. Also, assuming idle power of the system to be negligible, power consumption, P at a (V, f) pair is estimated as:

$$P \propto c * f * v^2 \tag{5.3}$$

where c is the effective capacitance. Using the above model, we have

$$If \;\; f_i < f_j \;\; then \;\; P(V_i, f_i) < P(V_j, f_j) \tag{5.4}$$

Since AFS increases frequency from worst case estimate, using AFS at a given (V, f) pair can enable the task as long as the error rate is less than the change in frequency. While system performance, at the same time, energy consumption of the system increases but only slightly [73]. However, as AVS

decreases voltage, using AVS at a given (V, f) pair can cause the task to violate deadline constraint and thus making the schedule invalid. Therefore, AVS cannot be used by slack reclamation phase.

Theorem 1. *Given a schedule, consisting of (V, f) pair information, that minimizes the makespan for a given task graph, AFS can be used to replace any (V, f) pair in the schedule as long as the timing error rate doesn't exceed the change in frequency.*

Proof. Lets say a task (T,K) is scheduled to run at (V, f) to meet the deadline. Let t_{wc} denote the worst-case clock period at (V, f). Let t_{ov} denote the clock period after overclocking the circuit. Let n be the number of cycles needed to recover from an error. Let us assume that the application runs at the overclocked frequency of period t_{ov} with an error rate of $k\%$ under AFS. To achieve any performance improvement at this frequency, Equation (5.5) must be satisfied. It states that even after accounting for error recovery penalty, execution time required is still less than that required for worst-case frequency operation. Equation (5.6) must be satisfied to have any performance improvement using reliable and aggressive designs.

$$K \times (t_{ov} + n \times k \times t_{ov}) < K \times t_{wc} \tag{5.5}$$

$$k < \frac{(t_{wc} - t_{ov})}{n \times t_{ov}} \; or \; k < \frac{(f_{ov} - f)}{n \times f} \tag{5.6}$$

As explained in Section 3.1.2, n=1 for reliable and aggressive systems. From Equation 5.6, we notice that as long as the error rate is less than the change in frequency, AFS offers performance enhancement and can be used for slack reclamation as it doesn't violate deadline constraint. □

According to Equation (5.6), for a frequency increase of 15%, the error rate must not be higher than 15%, for AFS to yield no performance improvement. For error rates less than *5%*, a frequency increase of only *5%* is required to offer performance advantage over worst case designs.

5.3 Frequency Selection Using Reliable And Aggressive Designs

In this section, frequency selection for optimizing energy consumption using DVFS in reliable and aggressive is investigated. Since, systems only support discrete voltage and frequencies, as shown in

[63], optimal energy is always obtained two frequencies and also these two frequencies are adjacent when processor energy consumption is a convex function of frequency. We also assume, energy consumption is a convex function of frequency as it is valid for most systems. This scheme is named multiple voltage frequency selection dynamic voltage frequency scaling (MVFS-DVFS). But, (V, f) pairs considered by MVFS-DVFS are from conservative DVFS spectrum and doesn't take additional voltage and frequency swings allowed by reliable and aggressive designs into consideration. To overcome this limitation, we propose, aggressive dynamic voltage and frequency scaling (ADVFS) for optimizing energy consumption using reliable and aggressive systems. Details of ADVFS are presented below.

5.3.1 Frequency selection using ADVFS

Let us assume, that a task (T,K) is to be scheduled and (V_1,f_1) and (V_2,f_2) are the adjacent voltage-frequency pairs for (V_{opt},f_{opt}). Let us say timing error rate at (V_{opt},f_{opt}) is $k\%$. From, MVFS-DVFS, we have

$$t_1 = (T * f_2 - K)/(f_2 - f_1) \tag{5.7}$$

$$t_2 = (K - T * f_1)/(f_2 - f_1) \tag{5.8}$$

where t_1 is the amount of time spent at (V_1,f_1) and t_2 is the amount of time spent at (V_2,f_2). Also $f_1 < f_{opt} < f_2$. Following Theorem 1, let us denote f_{AFS} as the maximum frequency that can be reached under AFS without violating the deadline constraint. Since amount of frequency swing is limited by timing error rate, we assume f_{AFS} is below the midpoint of f_1 and f_2 i.e. 0.5* (f_1 + f_2). Based on f_{AFS}, frequency selection problem is divided into two cases: (1) $f_{AFS} > f_{opt} * (1+k)$ and (2) $f_{AFS} \leq f_{opt} * (1+k)$. Figure 5.3 illustrates these two cases. Figure 5.3 (a) present the f_{opt} relative to (V_1,f_1) and (V_2,f_2), Figure 5.3 (b) present first scenario i.e. $f_{AFS} > f_{opt} * (1+k)$ and Figure 5.3 (c) present second scenario i.e. $f_{AFS} \leq f_{opt} * (1+k)$

Theorem 2. *If $f_{AFS} > f_{opt} * (1+k)$, significant energy savings are always obtained by operating at a single and aggressive (V, f) pair as opposed to using linear combination of traditional (V, f) pairs.*

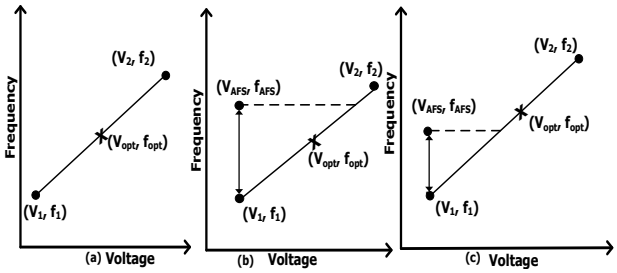

Figure 5.3 Frequency selection using AFS

Proof. From MVFS-DVFS, energy consumption for task (T,K) is given by

$$E_{MVFS-DVFS} = P_1 * t_1 + P_2 * t_2 \tag{5.9}$$

Using ADVFS, energy consumption for task (T,K) is given by

$$E_{ADVFS} = P_1^{'} * T \tag{5.10}$$

Inorder to have single frequency operation,

$$E_{ADVFS} < E_{MVFS-DVFS}$$

$$P_1^{'} * T < P_1 * t_1 + P_2 * t_2$$

$$P_1^{'} * (t_1 + t_2) < P_1 * t_1 + P_2 * t_2$$

Rearranging terms, we have

$$t_2 * (P_2 - P_1^{'}) > t_1 * (P_1^{'} - P_1)$$

$$t_2/t_1 > (P_1^{'} - P_1)/(P_2 - P_1^{'})$$

$$(t_2/t_1) > (x-1)/(y-x)$$

where $P_2 = y * P_1$ and $P_1^{'} = x * P_1$

Assuming power consumption of the adjacent (V, f) pairs differ by an order (at least more than couple times) and assuming 5% power penalty under AFS [73], we have:

$$t_1 > t_2$$

From equation 5.7 and 5.8, we have:

$$T * f_2 - K > K - T * f_1$$

$$f_{opt} < (f_1 + f_2)/2 \tag{5.11}$$

As $f_{AFS} < 0.5 * (f_1 + f_2)$ and $f_{opt} < f_{AFS}$, Equation 5.11 will be satisfied. For this scenario, operation at aggressive (V, f) pair is preferred. Therefore, task (T, K) need to operate at $(V_1, f_{opt} * (1 + k))$ and energy savings when compared to MVFS-DVFS is given by $E_{MVFS-DVFS}/E_{ADVFS}$. □

Theorem 3. *If $f_{AFS} \leq f_{opt} * (1 + k)$, significant energy savings are always obtained by using linear combination of a aggressive (V, f) pair and a traditional (V, f) pair.*

Proof. From MVFS-DVFS, energy consumption for task (T, K) is given by

$$E_{MVFS-DVFS} = P_1 * t_1 + P_2 * t_2 \tag{5.12}$$

Using ADVFS, energy consumption for task (T, K) is given by

$$E_{ADVFS} = P_1^{'} * t_1^{'} + P_2^{'} * t_2^{'} \tag{5.13}$$

In order to have linear combination of aggressive and traditional frequency pairs,

$$E_{ADVFS} < E_{MVFS-DVFS}$$

$$P_1^{'} * t_1^{'} + P_2 * t_2^{'} < P_1 * t_1 + P_2 * t_2$$

$$P_2 * (t_2 - t_2^{'}) > P_1^{'} * t_1^{'} + P_1 * t_1$$

As $f_1^{'} > f_1$, $f_2 > f_1$, and $V_2 > V_1$, from equation 5.7 and 5.8, we have, $t_1^{'} > t_1$ and $t_2^{'} < t_2$. Rearranging terms, we have:

$$P_2 * (t_2 - t_2^{'}) / (P_1^{'} * t_1^{'} + P_1 * t_1) > 1$$

$$(t_2 - t_2^{'}) / ((f_1^{'} * t_1^{'} / f_1) - t_1) > 1$$

$$((f_1^{'} * t_1^{'} / f_1) - t_1) > 1 \tag{5.14}$$

As $f_1^{'} > f_1$ and $t_1^{'} > t_1$, Equation 5.14 will be satisfied. Therefore, when $f_{AFS} \leq f_{opt} * (1+k)$, linear combination of aggressive and traditional frequency pairs is preferred. Task (T, K) need to operate at $(V_1, f_{opt} * (1+k))$ and (V_2, f_2) and energy savings when compared to MVFS-DVFS is given by $E_{MVFS-DVFS} / E_{ADVFS}$. □

5.3.2 Impact on design implementation

From implementation perspective, more aggressively the system is constrained, more design time for getting timing closure, thereby increasing the cost. Thefore, it is advantageous to build a system with frequency swing less than 0.5* (f_1 + f_2). Therefore, we recommend to build reliable and aggressive system with frequency swings upto the mid point of traditional DVFS (V, f) pairs.

5.4 Experiments and Results

This section presents simulation results of our proposed algorithm (ADVFS) and compare energy savings with schemes RDVFS and MVFS-DVFS [63]. We use continuous voltage and frequency scaling (CDVFS) as our base case. CDVFS gives optimal energy consumption for a given task set [63].

5.4.1 Example

In this section, we present an example to explain how each algorithm uses task's slack time for energy consumption minimization. We chose power model similar to the example presented in [63], where $P(V,f) = 1.367 * 10^{-24} f^3$. Table shows the task information, such as deadline constraint, clock ticks required, and available frequencies for task execution.

Parameter	Value
f_1	60MHz
f_2	80MHz
T	130 ms
K	9 million cycles

Table 5.1 Task parameters

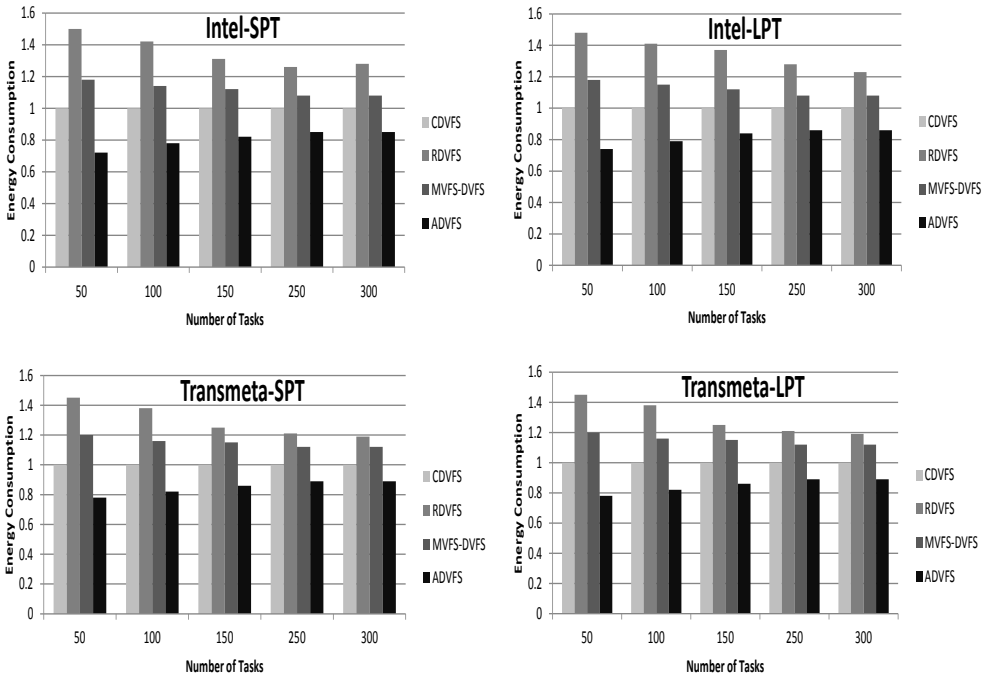

Figure 5.4 Comparing energy consumption of frequency selection schemes CDVFS, RDVFS, MVFS-DVFS, and ADVFS

Based on the parameters listed in the above table, we have:

- From Equation 5.2, we can calculate f_{opt}, which is K/T = 69.2 MHz, which is not supported by the underlying DVFS platform of the given system. The optimal energy consumption, which is observed under CDVFS, is 58.96 mW.

- Under RDVFS, processor executes the task at the higher frequency close to f_{opt}. In this example, given task is executed at 80 MHz and the corresponding energy consumption is 90.98mW.

- Under MVFS-DVFS, where linear combination of frequencies are used for task execution, given task is executed for 0.07 ms at 60 MHz and for 0.06 at 80 MHz. The corresponding energy consumption is 62.66 mW.

- Under ADVFS, as f_{opt} is near the mid frequency of f_1 and f_2, assuming 5% timing error rate and 5% increase in power consumption under AFS, linear combination of aggressive and traditional (V, f) pairs is preferred and the corresponding energy consumption is close 39.48 mW.

From above discussion, we observe that RDVFS allocates more energy than the required amount by nearly 50%. MVSFS-DVFS reduces the energy consumption close to the optimal energy consumption, which is observed under CDVFS. However, as DVFS (V, f) pairs are enhanced by reliable and aggressive designs, ADVFS reduces the energy consumption by nearly less than 33% of the optimal energy that can achieved by using traditional (V, f) pairs.

5.4.2 Experimental Settings

In our simulations, two schedulers: (1) list scheduler with Longest Processing Time first (LPT) and (2) list scheduler with Shortest Processing Time first (SPT) are used to produce task schedules minimizing makespan. We have used MATLAB torsche toolbox to generate schedule using LPT and SPT for random task graphs [74]. Once the schedule is generated, we ran our slack reclamation algorithm to compare ADVFS with other schemes.

For our study, we have chosen voltage frequency of two real processors: (1) Transmeta Crusoe [28] and (2)Intel Xscale [86]. Table 5.2 presents the available (V, f) pairs on Transmeta processor and Table 5.3 presents the available (V, f) pairs on Intel Xscale processor.

Frequency (MHz)	Voltage (V)	Power (W)
667	1.6	5.3
600	1.5	4.2
533	1.35	3.0
400	1.225	1.9
300	1.2	1.3

Table 5.2 Available (V, f) pairs for Transmeta processor

Frequency (MHz)	Voltage (V)	Power (W)
1000	1.8	1.6
800	1.6	0.9
600	1.3	0.4
400	1	0.17
150	0.75	0.085.3

Table 5.3 Available (V, f) pairs for Intel Xscale processor

To evaluate the performance of ADVFS, we have generated random task graphs consisting of different number of tasks. Required clock cycles of the randomly generated tasks is varied uniformly between 5-10 million cycles and execution time is also uniformly sampled between 100-300 ms. Using the available (V, f) pairs as shown in Table 5.2 and Table 5.3, task schedule has been obtained by minimizing the makespan. We assume the system is comprised of two processors, where each processor supports AFS. As timing error rate can bound the achievable frequency swings under AFS, we have considered scenario where $f_{AFS} \leq f_{opt} * (1+k)$ for all tasks. Out of two possible scenarios, case where $f_{AFS} \leq f_{opt} * (1+k)$ is preferred case if large frequency swings are not possible under AFS.

For all the random tasks, we assume timing error rate increases uniformly upto 5% at f_{AFS}. As reported in [73], we have modelled the power penalty for running at higher frequency and also to recover from occasional timing error to be under 5% . Also, we ignore the overhead of voltage and frequency transitions as they are relatively much smaller than the execution time of the tasks.

5.4.3 Results

Figure 5.4 presents our simulation results. It compares energy consumption of frequency selection schemes CDVFS, RDVFS, MVFS-DVFS, and ADVFS. As we can notice from figure, LPT and SPT did not have any significant effect in terms of energy consumption. On average, RDVFS when compared with CDVFS, consume more than 33% energy. MVFS-DVFS energy consumption is smaller than RDVFS for all cases and its average energy consumption is more 12% of CDVFS scheme. Of all schemes including CDVFS, our scheme, ADVFS, consume lower energy. On average, ADVFS consumes less than 18% energy when compared with CDVFS.

Figure 5.5 compare energy consumption of RDVFS, MVFS-DVFS and ADVFS schemes for Intel processor using SPT list scheduling. As we notice from the figure, with increase in number of tasks, possible energy savings decrease for all schemes. This is due to the increase in system utilization

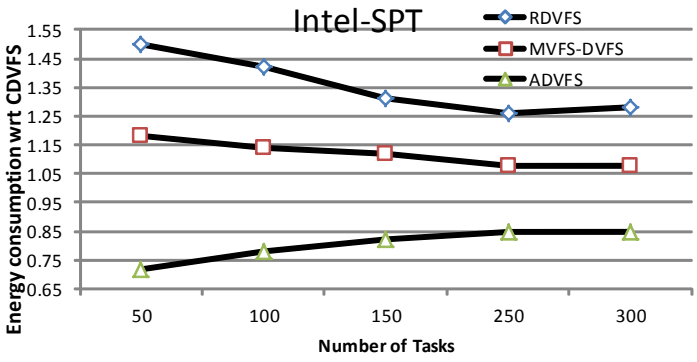

Figure 5.5 Comparing energy comsumption of RDVFS, MVFS-DVFS and ADVFS for Intel Xscale processor

as number of processors has remained constant. AFS has been proposed earlier as a performance enhancing scheme [73]. But ADVFS uses AFS from a different angle, where in benefits from operating beyond conservative worst case estimate are effectively translated into energy savings.

5.4.4 Overheads

Timing error recovery incurs power and performance overheads. Based on studies conducted in [73, 25], we account these power and performance overheads in our simulations. However, the main overhead incurred by reliable and aggressive designs is fixing the circuit contamination delay (CD) to a required value. ADVFS is beneficial when relatively small timing error rates are observed, similar to the error rates presented for Alpha processor in Section 3.2. Increasing CD involves rapid increase in silicon area, as buffers need to be inserted in the short circuit delay paths. As shown in [73, 25], these area overheads, along with timing error detection circuitry of reliable and aggressive designs, are about 3% at chip-level for a microprocessor implementation.

5.5 Conclusion

In this chapter, we propose a new slack reclamation algorithm, aggressive dynamic and voltage scaling (ADVFS), using reliable and aggressive systems. ADVFS exploits the enhanced voltage frequency spectrum offered by reliable and aggressive designs for improving energy efficiency. We also

provide guidelines on how to manage the enhanced voltage frequency spectrum and the constraints required for limiting the maximum attainable under ADVFS. We also provide formal proofs to show that significant energy savings are attainable either by using single frequency or by linear combination of frequencies. Our scheme can be integrated as post processing step with any of the existing static and dynamic task scheduling algorithms. Our simulations on random task graphs show 18% reduction in energy when compared with continuous DVFS schemes and over more than 33% when compared with scheme using linear combination of traditional voltage frequency pairs.

CHAPTER 6. SECURITY

6.1 Introduction

As implementation technology scales to nano-scale dimensions, electronic systems are now built with more complex and powerful integrated circuits. Along with the performance and power constraints, security also is emerging as a first class design constraint. This is especially true for embedded systems, which are deployed widely in domains ranging from mobile phones to smart cards. Rapid enhancements in VLSI technology, combined with innovations at various design hierarchies have contributed to meet the power and the performance constraints. For all concerns related to security, cryptography is being applied to guard the systems from known attacks.

6.1.1 Side-channel Power Attacks

Power analysis attacks are cryptanalytic attacks that exploit information leaks from a cryptographic system. The idea to use cryptographic system power traces for extracting secret key was first presented by Kocher [39]. Power analysis attacks involve two basic techniques: Simple power analysis (SPA) and differential power analysis (DPA). SPA attacks involve first order power analysis and are mounted on systems by using the fact that different operations in secret-key crypto algorithms consume different power. Using SPA attacks, secret information, such as secret key or parts of it, is inferred directly through visual inspection of the power traces. DPA attacks are based on statistical computations and are much powerful than SPA attacks as no detailed knowledge about the cryptographic system is required. An alternative to DPA was suggested by Brier named Correlation Power Analysis (CPA), which offers more efficient power analysis based on linear correlation techniques [15].

6.1.2 Chapter Contribution

We develop a new countermeasure using dynamic voltage and frequency scaling (DVFS). Earlier, a similar approach, named random dynamic voltage and frequency scaling (RDVFS), was investigated by Yang et. al. to reduce the correlation of input data to the power consumption. RDVFS reduce the correlation by varying the system voltage and frequency randomly [87]. But, later it was shown that RDVFS approach cannot prevent DPA/CPA power attacks [7]. Major limitation for RDVFS based countermeasure is on account of one-to-one mapping between voltage and frequency of DVFS (V, f) pairs. As pointed out in [7], even under RDVFS, system operating frequency can be easily estimated from the power traces collected. Once the frequency is estimated correctly, corresponding voltage can also be known and attacker can use this (V, f) pair to estimate the secret key. We overcome this inherent limitation of DVFS platforms by breaking the one-to-one mapping between voltage and frequency of (V, f) pairs.

We propose to use reliable and aggressive designs as building blocks, which also enable *typical-case* operation. Reliable and aggressive designs enable *typical-case* operation by using circuit-level timing speculation effectively [73, 25]. They support **Aggressive Frequency Scaling** (AFS) [73] and **Aggressive Voltage Scaling** (AVS) [25]. Such systems not only enable AFS or AVS but also break one-to-one mapping between voltage and frequency of DVFS (V, f) pairs. Using Fisher's Z-transform, we demonstrate that our technique reduce the correlation present between two given signals at statistically significant (vulnerable to DPA/CPA) levels to statistically insignificant levels (secure from DPA/CPA) under 95% degree of confidence interval.

To estimate the performance/power advantages and to study the changes to (V, f) spectrum of DVFS platforms, we conducted our experiments on an AES 8-bit S-box. Our results show that using AFS technique, an enhanced (V, f) spectrum can be realized without any further additional resources to combat power attacks. Moreover, the performance of S-box is increased by 22% over the worst-case estimate. Also, it has decreased the correlation for the correct key by an order and has increased the probability by almost 3.5X times for wrong keys when compared with the original key to exhibit maximum correlation. Thus, our technique add a new dimension to power attack countermeasures by effectively exploiting *typical-case* operation.

6.1.3 Chapter Organization

The chapter is organized as follows. In Section 6.2, we present background details of power analysis attacks and related work. We present DVFS based countermeasure details in Section 6.3. In Section 6.5, we explain how such designs break one-to-one mapping between voltage and frequency of DVFS (V, f) pairs. In Section 6.6, we model and analyze the power behavior of systems using our approach. Then we present security analsyis to demonstrate the resistance of our countermeasure in Section 6.7. We present our results in Section 6.8.

6.2 Background

6.2.1 Theory behind Power Attacks

The most effective power analysis attack is differential power analysis (DPA). It combines an actual power trace, obtained for several encryptions rounds with known inputs and an estimated power trace, which is generated using a hypothesized key and predictions about the transitions of internal nodes. Next, using statistical evaluation, most likely value of the key is determined. As already demonstrated, DPA is powerful enough to successfully retrieve the key from implementations of AES [39, 40].

In CMOS technology, for an applied stimulus, when a value inside a component (e.g. a register) of digital electronic system changes from logic 0 $\rightarrow$ 1 or logic 1$\rightarrow$0, associated power consumption is significantly higher than when the values remains constant i.e. logic 0$\rightarrow$0 or logic 1$\rightarrow$1. This difference in power values leads to data dependent power characteristics for a system implemented in CMOS technology.

Initially the DPA attack was developed using a distance-of-mean test, which simply takes the difference between the mean of two sets of data. DPA was later extended to correlation power analysis (CPA) which again involves hypothesizing a secret key and then analyzing correlation coefficients between the actual power trace measured and the estimated power trace [15]. Models like hamming weight, hamming distance, or toggle count have been used for estimating the power consumption of the cryptographic system [54]. Search over the entire key space is then conducted and the correlation coefficients are determined. Correct key is the one which maximizes correlation factor, ρ, which is

given by Equation 6.1.

$$\rho = \frac{E(P_A, P_E) - E(P_A)E(P_E)}{\sigma_{P_A}\sigma_{P_E}} \tag{6.1}$$

where P_A denotes the actual power trace measured, P_E denotes the estimated power trace, $E()$ denotes the expectation, and σ denotes the variance. For N power traces recorded for N different inputs, each trace having M recording points, attacker calculates the correlation trace, $\rho_{K,NM}$, for hypothesized key K. For example, in AES, all possible keys for the S-box is 256. For all possible 256 key values, correlation traces are then constructed with the help of hypothetical power model. In each correlation trace, $\rho_{K,NM}$, there is a highest peak, $\rho_{K,N}$, and largest ρ among all $\rho_{K,N}$ indicates the correct key.

6.2.2 Related Work

The past countermeasures for power analysis attacks can broadly be classified into three categories: balanced logic styles, masking techniques at gate-level and algorithmic-level, and dynamic voltage and frequency varying techniques. Balanced logic style techniques, such as Sense Amplifier Based Logic (SABL), and Wave dynamic and differential logic (WDDL) [76, 79], attempt to avoid the leakage of information by reducing the variance in the power consumed for different logic transitions. Logic replication is essential in order to balance the logic so that there is always one transition. For combinational logic, it requires doubling the amount of logic and to balance flip-flops, four normal ones are normally required. Therefore, overall area of the design will be more than double.

Using simulation, Tiri and Verbauwhede found that WDDL logic style is DPA resistant when layout parasitic are ignored [78]. But ASIC developed by using single and dual rail AES core [77], is found to reveal 11 out of 16 secret key bytes. Dual rail designs come at a high cost in terms of performance, area, and power. For the WDDL ASIC, the chip area has increased by 3 times, the speed is reduced by a factor of 4, and the power consumption has increased by 4 times[Goodwin]. Approaches like pre-charge logic and gate-level masking are conjoined to develop Random Switching Logic (RSL) [75] and masked dual-rail pre-charge logic (MDPL) [62]. Use of these logic styles improves DPA resistance but must be applied at the transistor or even deeper design levels. Consequently, overheads such as design effort, area, power consumption, and performance limit their adoptability. Asynchronous design

methodology like clock less design [88] offer no DPA resistance unless it is combined with dual rail balanced logic techniques.

Algorithm-level making techniques attempt to hide internal variable. Components performing linear operations are easy to mask. Therefore, many approaches were focused to develop masked AES S-box [11, 61], as it performs non-linear operation. But later in time, Mangard et al presented a practical power analysis attack, by changing the hypothetical power model from hamming weight/distance model to toggle count model. Using this power model, two masked AES implementations which revealed the secret key [47]. Similarly, other masking approaches are breakable by using more precise power models. On average, it was found that, by using masking techniques, speed of the system reduces twice and area increases by 3 times and still being susceptible to DPA attacks [Goodwin].

Third class of countermeasures adopt dynamic changes to voltage and frequency of the system such as RDVFS [87]. Unlike other countermeasures, this approach has increased the entropy of power traces and provided 27% energy reduction and has only 16% time overhead for DES encryption and decryption algorithms. However, later it was shown that RDVFS cannot prevent CPA attacks [7]. To overcome this limitation, Baddam et al propose to vary voltage randomly at a fixed frequency, as on-the-fly frequency changes are detectable [7]. Using this approach, they were able to reduce the correlation by five times. But power consumption of the system increases heavily in-order to provide high voltage swings and also cannot operate beyond worst-case estimates. Also, at higher voltages transistors can enter saturation region and hiding voltage may not be effective.

6.3 Dynamic Voltage and Frequency Scaling

Dynamic voltage and frequency scaling (DVFS) is an approach to provide variable amount of energy for a task/process by changing the operating voltage and frequency of a system dynamically. Its primary goal is to provide optimal power required by the underlying task/process, thus reducing the system energy consumption. Systems supporting DVFS has a set of predefined discrete voltage levels. As the circuit delay has strong association with the supply voltage, system frequency is also adjusted to the corresponding worst-case circuit delay. The voltage-frequency, (V, f), pairs are estimated off-line during the design phase and determined after fine tuning the system characteristics after fabrication.

This process of (V, f) pair estimation involves studying system characteristics under worst-case settings.

Table 6.1 shows the (V, f) pairs for the Intel's speedstep technology [64]. As we notice from the table, frequency can be changed from 600MHz to 2GHz in steps of 200MHz, while corresponding voltage level is varied from 0.988 V to 1.340 V. Such a spectrum contain (V, f) pairs that offer high performance and high power levels as well as (V, f) pairs that offer reduced performance and lower power consumption. The later typically are used when system is idle. For example, the voltage-frequency pair (1.34V, 2GHz) has higher performance and power levels when compared with the voltage-frequency pair (0.988V, 600MHz).

With using such (V, f) spectrum, DVFS platforms provide the necessary computation power for systems as applications become increasingly sophisticated by exploiting hardware characteristics. DVFS algorithms alternate between (V, f) pairs at a constant or dynamic rate and switching to a new configuration involves some time penalty. However, as shown in [36], this penalty is in the order of *ns* and can be be compensated by carefully choosing the DVFS rate along with appropriate choice of DVFS algorithm.

6.3.1 DVFS based countermeasure

DPA attacks discover correlations in time between system power traces under the assumption that the cryptosystem operates at a constant frequency. This observation is leveraged to develop an area and power efficient countermeasure, random DVFS (RDVFS), that breaks this key assumption [87]. It used traditional DVFS mechanisms to randomly change frequencies during system operation. However, this simple and area efficient technique has limitations. In [7] it was shown that successful DPA/CPA attacks could still be mounted on systems using RDVFS countermeasure.

6.3.2 Characteristics of correlation under RDVFS

The relative difference between the maximum and second maximum value of ρ over the entire key space play a dominant role in discovering the secret key, which is introduced formally in [84]. It is important to note that the uncertainty in distinguishing the correct key depend on the degree of

Frequency f (in MHz)	Voltage V (in Volt)
0600	0.988
0800	1.052
1000	1.100
1200	1.140
1400	1.196
1600	1.244
1800	1.292
2000	1.340

Table 6.1 Intel Speedstep voltage-frequency (V, f) pairs

difference between these values. Any countermeasure increases the number of traces to collect by minimizing the degree of difference between these two values.

Baddam et. al. performed extensive transistor-level simulations of AES and conducted in-depth study using toggle count power model to validate RDVFS countermeasure [7]. We summarize characteristics of ρ observed in this study below.

- **Case-I:** When (V, f) pair is *known*, maximum ρ is observed for the correct key and Pearson correlation coefficients are in range [0.06, -0.02]. Another important parameter to note is the second maximum value of ρ. This value is estimated to be around 0.03 and this plays an important role in identifying the correct key.

- **Case-II:** When (V, f) pair is *unknown*, maximum value of ρ is observed for an incorrect key and Pearson correlation coefficients are in range [0.03, -0.02]. Also, second maximum value of ρ is close to 0.022.

6.3.3 Changes in correlation due to misprediction of (V, f) pair

As we notice from above discussion, while mounting a power attack, voltage and frequency information cannot be ignored for a system using DVFS. Since power is directly dependent on voltage and frequency of the system, any misprediction of (V, f) pair will have a big impact on ρ and it can lead the attacker to hypothesize a wrong key. For example, Figure 6.1 presents a case where misprediction of (V, f) pair introduce changes in correlation calculation. In Figure 6.1, P_{E1} represents the estimated power trace by the attacker when (V, f) pair is *known* and P_{E2} represents the estimated power trace when (V,

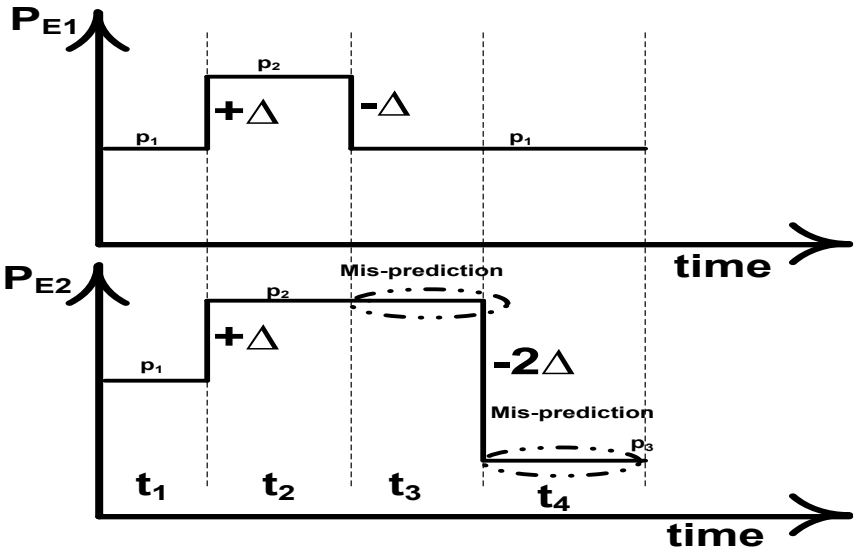

Figure 6.1 Figure showing the trend changes in the estimated power trace due to (V, f) pair misprediction

f) pair is *unknown*. Lets say, ρ_1 and ρ_2 are the estimated values of correlation using these traces. We assume inputs are fed in the same order for estimating P_{E1} and P_{E2}. We also assume, attacker estimates the correct (V, f) pair while estimating P_{E2} at time t_1 and t_2 (assuming 50% success probability).

As shown in the figure, while constructing trace P_{E2}, (V, f) pair misprediction happen at time t_3 and t_4 and change the trend in the estimated power trace. Originally, change that is expected between time t_2 and t_3 is nullified due to the first misprediction at time t_3, while the second misprediction at time t_4 introduces a big jump in the power value when no change in expected. Since inputs are same at the respective time intervals, such misprediction influence ρ_2, causing ρ_2 to differ from ρ_1. Degree of difference between ρ_1 and ρ_2 will depend on misprediction rate. Such changes in correlation can ultimately lead the attacker to hypothesize a wrong key. Also, under RDVFS, when (V, f) pair is *unknown*, it is not possible for a single key to exhibit maximum correlation value. It can be observed for different keys depending on (V, f) pair misprediction. Under such scenarios, attacker needs to do multiple attempts to guess the secret key.

Main vulnerability of RDVFS is the one-to-one mapping that is present between voltage and frequency of traditional DVFS (V, f) pairs, which can be exploited by attackers to recover correlations between power traces. Glitches in a power trace can be used to estimate frequency. Once frequency is

known, voltage can also be trivially found. This assumes, of course, that the attacker has knowledge of the one-to-one map implemented by the DVFS mechanism (contrary to security through obscurity). We propose an alternative to the work in [87] and use aggressive and reliable building blocks to break the one-to-one mapping between the voltage and frequency of (V, f) pairs used by conventional DVFS algorithms.

6.4 Reliable and Aggressive Designs

In a pipelined system, the clock frequency is determined based on the critical path across all stages, under adverse operating conditions. Traditional design methodologies for the worst-case operating conditions are too conservative as the critical timing delays rarely occur in tandem, during typical circuit operation. Moreover, circuit delay has a strong association with the data being processed. For instance, in a carry-propagate adder, the worst-case delay occurs only when the carry is to be propagated through each bit-slice, which occurs only for a specific input data set [4]. Such infrequent occurrence of critical timing delays has opened a new domain of study that allows improvement of systems performance to a greater extent through timing speculation at circuit level. Speculation at this level allows voltage/frequency swings beyond worst-case estimates.

In Razor [25], timing speculation has been exploited in voltage domain using adaptive voltage scaling. We refer to this technique as **Aggressive Voltage Scaling** (AVS). In adaptive and reliable overclocking [73], timing speculation has been exploited in frequency domain using adaptive frequency scaling. We refer to this technique as **Aggressive Frequency Scaling** (AFS). Impressive results are achieved using AVS and AFS and it enables systems to work past the worst-case limitations. Both these complementary techniques are well suited for improving resistance against power analysis attacks. Security strength offered by AVS is similar to that of AFS. We keep our discussion in the rest of the chapter to AFS and the analysis can be easily extended to AVS.

6.4.1 System Architecture

Since the data forwarded at higher frequencies can be faulty (*DataOut* signal in Figure 3.1 (a)), wrong cipher texts can be forwarded and the security of the whole system can be compromised. This

can be avoided by inserting an extra buffer at the last pipeline stage of the system and the modified timing error detection circuit is shown in Figure 6.2. It should be noted that timing error detection circuitry of only last pipeline stage need to modified as shown in Figure 6.2. Registers $Main_1$ and *Backup* are clocked similarly to registers *Main* and *Backup* shown in Figure 3.1(a) and register $Main_2$ will always have correct data. During normal operation, register $Main_2$ loads from $Main_1$. If a timing error is detected, it loads from register *Backup* and the system need to be stalled for a cycle. This operation is identical to the recovery mechanism of a timing error and system level recovery is identical to the recovery mechanisms developed in [6, 73].

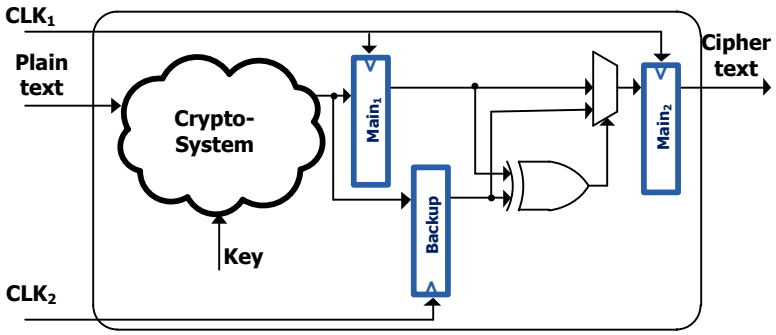

Figure 6.2 Figure highlighting the architecture of the cryptographic system and how the timing critical pipeline registers of last pipeline stage need to be modified to prevent forwarding of faulty cipher texts.

6.4.2 Performance Analysis

A key factor that limits frequency scaling of AFS is the timing error rate. As frequency is scaled higher, the number of input combinations that result in delays greater than the new clock period also increases. The impact of error rate on frequency scaling is analyzed as follows:

Let t_{wc} denote the worst-case clock period. Let t_{ov} denote the clock period after overclocking the circuit. Let n be the number of cycles needed to recover from an error. Let us assume that sytem takes N clock cycles to execute, under normal conditions. Let us assume that the application runs at the overclocked frequency of period t_{ov} with an error rate of $k\%$. Because of the extra buffer added at the last pipeline stage, there will be latency of one clock cycle when the system is operating at clock period t_{ov}. To achieve any performance improvement at this frequency, Equation (6.2) must be satisfied. It

states that even after accounting for error recovery penalty, execution time required is still less than that required for worst-case frequency operation. For large value of N, Equation (6.3) must be satisfied to have any performance improvement using reliable and aggressive designs.

$$(N+1) \times (t_{ov} + n \times k \times t_{ov}) < N \times t_{wc} \tag{6.2}$$

$$k < \frac{(t_{wc} - t_{ov})}{n \times t_{ov}} \tag{6.3}$$

As explained above, for our technique n=1. According to Equation (6.3), for a frequency increase of 15%, the error rate must not be higher than 15%, for our technique to yield no performance improvement. For error rates less than *1%*, a frequency increase of *1%* is required for our scheme to offer performance advantage over worst case designs.

6.5 Breaking connection between DVFS (V, f) pairs

In this section, we first explain how the one-to-one mapping between voltage and frequency of DVFS (V, f) pairs is broken using aggressive designs. Then, using aggressive designs, we demonstrate how to manage the (V, f) spectrum used by the traditional DVFS. Figure 6.3 illustrates the solution space of (V, f) pairs, available for traditional DVFS. The voltage can be varied from V_{max} to V_{min} (where $V_{max} > V_{min}$), while the frequency can be varied from f_{max} to f_{min} (where $f_{max} > f_{min}$). The voltage-frequency pair (V_{max}, f_{max}), has the highest energy consumption while the voltage-frequency pair (V_{min}, f_{min}), has the lowest energy consumption.

In Figure 6.3 (a) and (b), (V, f) pairs present on the straight line (connected through •), are the available pairs for traditional DVFS approach. We denote such pairs having one-to-one mapping or linear relationship by (V_L, f_L). In such a sample space, once the voltage V or the frequency f is known, other unknown parameter is also known because of their relationship. Since frequency f can be estimated from the power traces, the (V, f) pair is no longer an unknown and RDVFS countermeasure is therefore vulnerable to DPA/CPA power attacks.

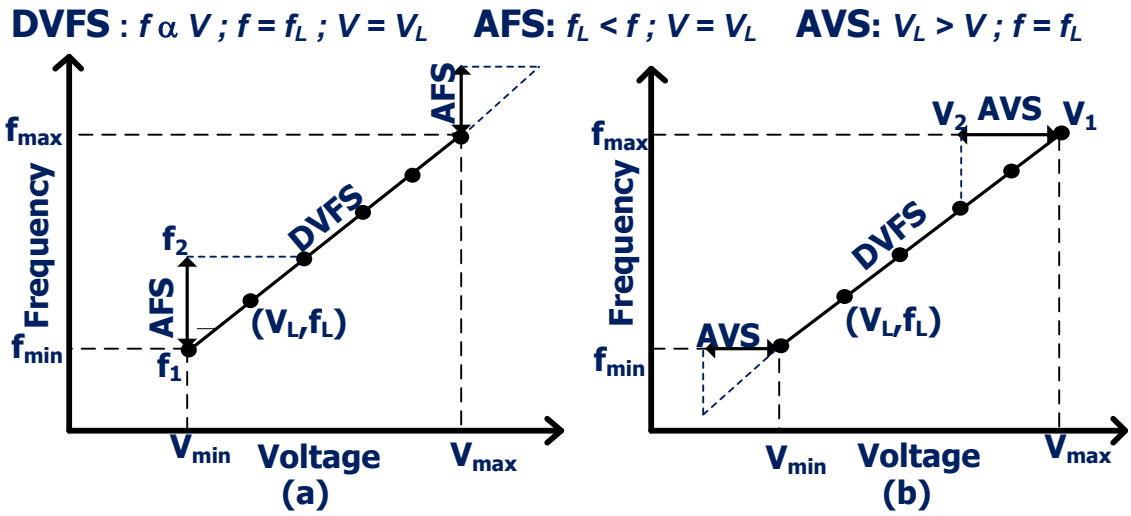

Figure 6.3 Figure showing possible voltage frequency pairs for (a) DVFS and AFS schemes and (b) DVFS and AVS schemes

6.5.1 Changes in DVFS (V, f) spectrum

Using reliable and aggressive designs, voltage-frequency solution space is expandable by two mechanisms - Aggressive Voltage Scaling (AVS) [25] and Aggressive Frequency Scaling (AFS) [73]. With AVS, at each voltage and frequency pair, given by (V_L, f_L), operating configuration can be changed by varying voltage V_L, between (V_1, V_2), i.e., deviating from the linear relationship and allowing to traverse horizontally, as illustrated in Figure 6.3 (b). With AFS, at each voltage and frequency pair, given by (V_L, f_L), operating configuration can be changed by varying frequency f_L, between (f_1, f_2), i.e., deviating from the linear relationship and allowing to traverse vertically, as illustrated in Figure 6.3 (a). At each (V_L, f_L) configuration, the voltage range available for AVS i.e., $\Delta V = V_1 - V_2$ or the frequency range available for AFS i.e., $\Delta f = f_2 - f_1$, can be increased to enhance the available (V, f) spectrum. This is achieved by managing the contamination delay (T_{CD}) of the system. Using AFS, at a (V_L, f_L), to have a frequency range of Δf , where Δf = f_2 - f_1, T_{CD} of the circuit should satisfy the constraint given by Equation 6.4.

$$T_{CD} \geq 1/f_L - 1/(f_L + \Delta f) \tag{6.4}$$

Using the data presented in Table 6.1, at voltage-frequency pair (1.1V, 1000MHz), to have a fre-

quency swing of 200 MHz, T_{CD} of the circuit should be increased to 0.167ns. For 400 MHz frequency swing, T_{CD} of the circuit need to be increased to 0.286ns. This area overhead comes in addition to the overhead caused by timing error detection and correction circuitry.

For example, let us assume that $\{(V_1, f_1), (V_2, f_2), (V_3, f_3), (V_4, f_4)\}$ are the supported voltage-frequency configurations with the traditional DVFS approach and also the frequencies are equidistant i.e. $\Delta f = f_2 - f_1 = f_3 - f_2 = f_4 - f_4$. Figure 6.4(a) presents possible state transitions between states (V_1, f_1) and (V_2, f_2). Also, let us assume that, T_{CD} of the circuit is increased beyond twice the maximum of frequency differences between the adjacent (V, f) pairs, i.e., $2\Delta f$. Under such a scenario, Figure 6.4(b) presents all possible state transitions for voltage pair (V_1, f_1) under both AFS and DVFS. Under DVFS only one state transition is possible from (V_1, f_1). This is the state transition between (V_1, f_1) and (V_2, f_2) as shown in the figure. But under AFS, two more additional state transitions are possible from (V_1, f_1). These are represented as state transition between (V_1, f_1) and (V_1, f_2), and (V_1, f_1) and (V_1, f_3). It should also be noted that state transition between (V_1, f_2) and (V_1, f_3) is also possible under AFS and state transition between (V_1, f_2) and (V_2, f_2) is also possible under AVS as shown in the figure.

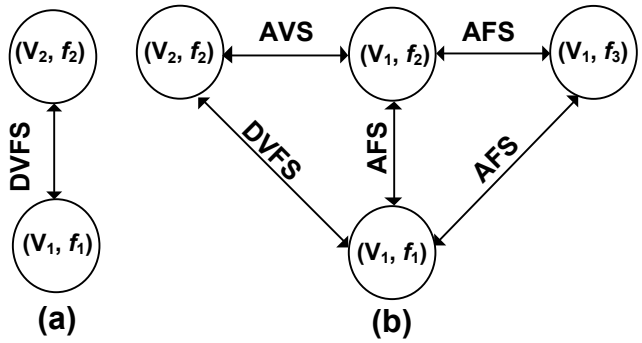

Figure 6.4 Possible state transitions for (V_1, f_1)pair (a) under DVFS and (b) under reliable and aggressive techniques AFS and AVS

For the system configuration presented above, Table 6.2 presents possible (V, f) pairs for each voltage level under DVFS and AFS approaches. For frequency f_3, from the table, we notice that the system can operate at three voltage levels, namely V_2, V_3, or V_4. Even under random DVFS, if the attacker estimates the frequency by observing glitches on the power line, it is no longer possible to estimate the actual voltage level, as the one-to-one mapping between voltage and frequency is broken. Under such situations, as shown in [7], attacker observes an incorrect key. Therefore, we propose to

prevent power attacks using reliable and aggressive designs assisted with random DVFS. It should also be noted that similar changes to (V, f) spectrum can be observed by employing AVS.

In addition to breaking the one-to-one mapping between the voltage and frequency, it also increases the randomness present in the power traces. Let us assume that a system built with reliable and aggressive designs is set to operate with $k\%$ timing error rate. This implies that out of every N operations executed, on an average k extra operations are also executed in the crypto-system. This increases the entropy of the power traces collected roughly in the order of $ln(2^{N(1+k)}/2^N) = Nk$, assuming every operation has a unique value of power consumption. For example, for every 100 operations executed in the system, entropy of power traces can be increased as high as 25 times while operating at 25% error rate.

Voltage Level	DVFS	AFS
V_1	(V_1, f_1)	(V_1, f_1),(V_1, f_2),(V_1, f_3)
V_2	(V_2, f_2)	(V_2, f_2),(V_2, f_3),(V_2, f_4)
V_3	(V_3, f_3)	(V_3, f_3),(V_3, f_4),(V_3, f_5)
V_4	(V_4, f_4)	(V_4, f_4),(V_4, f_5),(V_4, f_6)

Table 6.2 Possible (V, f) pairs using DVFS and AFS

6.5.2 Impact of timing errors

When a system is operating at single (V, f) pair using AFS/AVS, it may be possible to detect the occurrence of a timing error. But under random DVFS scheme where (V, f) pairs are selected randomly; it may not be possible to identify the timing error events precisely. This is due to the following. Let us assume that timing error detection and recovery events are high power consuming operations. At a given frequency, when high power values are detected in the trace, it can be attributed to two events: 1) a timing error event or 2) normal system operation executing at the same frequency but for a higher voltage as such configurations are supported by reliable and aggressive designs. This increases the fault positive rate in (V, f) pair estimation and causes trend changes in ρ calculation. It is important to note that, these trend changes cannot be cancelled out using frequency-based attacks, as the system is using random DVFS. Similarly, trend changes in estimated power trace happen even if we consider the timing error events as low energy operations. Therefore, side effect of aggressive designs i.e. timing

errors detection do not leak any useful information.

In worst-case scenario, if all the timing error occurrences are detected in the power trace, leakage about (V, f) pair can be completely eliminated by making sure that timing errors happen at more than one voltage for a given frequency or vice versa. Therefore, the challenge in using reliable and aggressive designs is to come up with an enhanced (V, f) spectrum that eliminates one-to-one mapping and also maps frequency to atleast two different voltage levels where timing errors can happen. For the (V, f) spectrum presented in Table 6.2, we can notice that for frequency f_3, timing errors can happen at (V_1, f_3) and (V_2, f_3) pairs. Traversal through such an enhanced (V, f) spectrum effectively eliminates possible information leakage through timing error detection.

In Section 6.3.3, we have assumed that only 50% of time correct (V, f) pair is predicted accurately and also assume trend change in estimated power trace on (V, f) pair misprediction. Main insight of this example is to show how (V, f) pair misprediction impact correlation value. Even, if we assume trend change to happen for only 50% of time on misprediction, trend changes still happen in the estimated power trace. For example, assuming misprediction during time t_3 and time t_4 (50% misprediction rate) and trend change to happen only for time t_3 misprediction (trend change only for 50% of mispredicted time), still influence correlation. This is equal to 0.5*(1) and 0.5*(0.5)=75% prediction accuracy. Following our above discussion, using our solution, any frequency needs to be mapped to more than two voltages. So for (V, f) spectrum presented in Table 6.2, assuming all (V, f) pair for a given frequency to be equally likely and trend change to cause for only 50% of time on (V, f) pair misprediction, prediction accuracy is not greater than 0.33*(1) + 0.33*(0.5) + 0.33*(0.5) = 66%, which is still lower than the prediction accuracy we assumed in Section 6.3.3.

6.5.3 Overheads

One of the main overheads incurred by reliable and aggressive schemes is fixing the circuit contamination delay to a required value and the register duplication. Not all registers need to be duplicated. Duplication of registers that fall only on timing critical paths is required. As shown in previous research, area overheads can be kept to minimal and is about 3% at chip-level [25]. Also, flip-flops may enter a metastable state when operated with a overclocked frequency (AFS) or with undervolting

(AVS). Incorporation of a metastability detection circuits, similar to the one developed in [25], are needed.

6.6 Power Analysis

Attackers use power models like hamming weight model, toggle count model etc. to deduce the correlation between the hypothesized key and the actual key [54]. They assume that voltage and frequency are kept constant and develop or use already existing power models. For DVFS based countermeasures, power models need accurate voltage and frequency information as there is very high dependence of power on system voltage and frequency. Any error in (V, f) pair estimation leads to trend changes as discussed in Section 6.3.2. In this section, we first model the power consumed by a system operating under AFS and discuss how each component can influence the correlation under random DVFS.

6.6.1 Static Power

The power dissipated by a circuit can be classified into two broad categories - static power (P_{stat}) and dynamic power (P_{dyn}). Static power (P_{stat}) is the power dissipated by a circuit when it is not switching, i.e. when it is inactive or static. The largest percentage of P_{stat} is due to the source to drain sub-threshold leakage current and P_{stat} can be modeled as,

$$P_{stat} \propto I_L \times V \tag{6.5}$$

where I_L is the total leakage current of the circuit and V is the voltage level at which the circuit is being operated. From Equation 6.5, we notice that P_{stat} is independent of clock frequency, f and depends on voltage V only. Reliable and aggressive designs tolerate timing errors by employing a one-cycle low overhead error recovery mechanism [25, 73]. For small error rates, its impact on static power is minimal and can be ignored while accounting for the static power of systems employing AFS. Even if frequency is estimated from the power traces, error in voltage estimation has a direct impact on correlation factor. P_{stat} can be visualized as a component providing random bias each cycle.

6.6.2 Dynamic Power

Dynamic power (P_{dyn}) is the power dissipated when a circuit is active. A circuit is active anytime the voltage on an internal node changes due to the input stimulus applied to the circuit and it can be modeled as,

$$P_{dyn} \propto \alpha \times V^2 \times f \tag{6.6}$$

where α represents the average node switching probability. As we notice that from Equation 6.6, P_{dyn} is strongly dependent on the circuit voltage and frequency i.e (V, f) pair. The power dissipated in detecting and correcting the timing errors (TE) should be accounted in P_{dyn} for systems using AFS. This additional component, P_{TE}, can be modeled as,

$$P_{TE} \propto k \times HD \times I_{0\rightarrow 1} \times V \tag{6.7}$$

where k represents the timing error rate and HD represents the hamming distance between values stored in registers *Main* and *Backup*, and $I_{0\rightarrow 1}$ is the average current required to change a 1-bit register logic state '0' to state logic '1'. $I_{0\rightarrow 1}$ can be easily estimated for a given CMOS technology and the parameters $\{k, HD\}$, depend on the (V, f) pair.

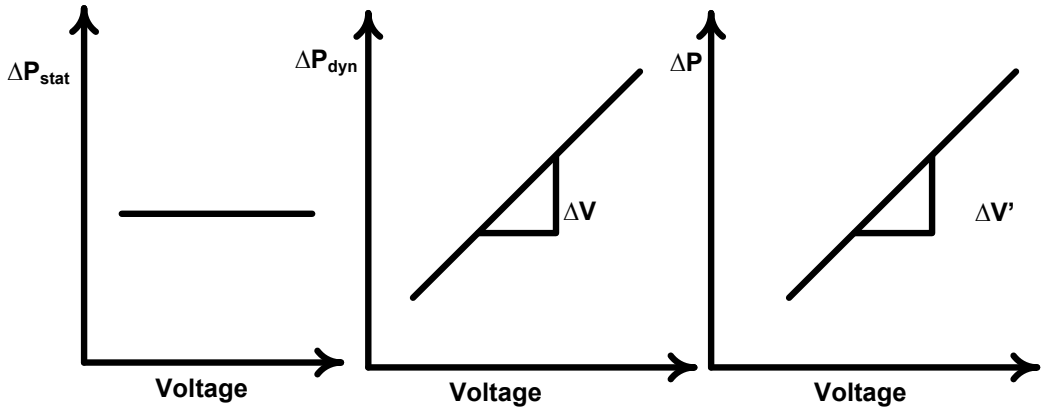

Figure 6.5 Figure showing the dependence of error in P_{stat} and P_{dyn} estimation with voltage

Equation 6.8 and equation 6.9 model the error in P_{stat} and P_{dyn} due to (V, f) pair misprediction.

Figure 6.5 presents the dependence of error in power estimation versus voltage. In figure, ratio of $\Delta V'$ to ΔV is equal to the ratio of P_{dyn} to P_{stat}. Large the distance between (V, f) pairs, higher is the error in both P_{stat} and P_{dyn}. Also, error in P_{dyn} increases linearly with voltage. Therefore, it is preferable to operate at high energy (V, f) pairs.

$$\Delta P_{stat} \propto \Delta V \tag{6.8}$$

$$\Delta P_{dyn} \propto \Delta V \times V \tag{6.9}$$

With random DVFS approach, where voltage and frequency are one-to-one mapped, P_{stat} and P_{dyn} can be easily estimated, once the (V, f) pair and the circuit topology are known. But with systems using random AFS or AVS, P_{stat} and P_{dyn} can no longer be estimated accurately for a given circuit topology. As we notice from Equations 6.8 and 6.9, error in the voltage estimation has direct impact on correlation factor (ρ). If the timing error events are not detected from the power trace, P_{TE} also contribute to the deviation in ρ estimation. It is important to note that, at chip-level, power overhead introduced by timing error circuitry is only 3% of the total power consumption [25]. Therefore, the impact of P_{TE} on ρ is minimal or negligible. Even if the timing error occurrences are detected and completely removed from power traces, components P_{stat} and P_{dyn} still impact ρ.

Even though P_{TE} contribution towards ρ is negligible, it can still influence ρ estimation by leaking (V, f) pair information. As discussed in Section 6.5.2, detection of timing error events will be not reveal (V, f) pair information if timing errors happen at more than one voltage for a given frequency or vice versa. Therefore, each frequency need to support more than two voltage levels. One such example of (V, f) spectrum is presented in Table 6.2.

Since, timing error rate at different (V, f) pairs differ, theoretically it is possible to estimate operating (V, f) pair for a module. For a single functional unit like adder, because of data dependency, it takes more than 'k' (we estimate k=10,000 based on our experiments) clock cycles to estimate timing error rate at a given (V, f) pair. But at system level, attacker can only estimate global error rate. Since each frequency is mapped to three or more voltages (say V voltages), assuming a module contribution to global timing error rate is g% and attacker detects timing errors with a% accuracy, for 1GHz

clock, required DVFS period is the order of (V-1)*10us/(g*a). In a balanced pipeline with 'p' pipeline stages and assuming each pipeline stage to produce a timing error equally likely, required DVFS period is in the order of (V-1)*p*10us/a. For V=3 and p=25, with 66% detection accuracy (as explained in Section 6.5.2, we need to change (V, f) pair for every 750 us, which can be easily supported by a coarse-grained system level switching mechanism. For 1GHz clock, this is equal to 1.5 million cycles. Typical configurations of cryptosystem are in MHz range. So, even DVFS rate periods larger than 500us are sufficient.

6.7 Security Analysis

To test the resistance offered by our countermeasure, we conducted a series of simulations and then analyzed the results using Fisher's Z-transform to test the statistical significance of changes observed in correlation. After describing Fisher's Z-transform, we present the changes observed in ρ using our technique. Next, we discuss how the changes observed in correlation reduce the risk of side-channel power attacks.

6.7.1 Fisher's Z-Transform

Using Fisher's Z-Transform, the correlation can be transformed to a value z_ρ which has a normal distribution and the transformation is given by Equation 6.10,

$$z_\rho = 0.5 * ln(\frac{1+\rho}{1-\rho}) \tag{6.10}$$

where z_ρ has an approximate normal distribution with variance $\sigma_\rho^2 = \frac{1}{N-3}$, where N in the number of samples collected to observe ρ. It should be noted that this model is only accurate only when N is greater than 30. All reported successful attacks used traces much higher than 30. Therefore, the model is valid all the time. To assess the statistical significance of our results, we have used the procedure listed as in [43]. We have calculated the normalized difference between ρ_{max} and ρ_{2nd_max} values using Equation 6.11 (from [43])

$$\Delta z = z_{max} - z_{2nd\_max} * \sqrt{\frac{N-3}{2}} \tag{6.11}$$

where N is the number of samples collected until ρ_{max} and ρ_{2nd_max} are observed. Once the difference between ρ_{max} and ρ_{2nd_max} under Z-transformation has been calculated, their difference is used to assess the statistical significance under the chosen confidence degree. If the difference between ρ_{max} and ρ_{2nd_max} under Z-transformation calculated, using Equation 6.11, is larger than the critical value, then we recognize the difference as statistically significant. The larger the difference is, the easier it is to discern the right guess and therefore, less secure is the system from power attacks. The critical value at a 95% degree of confidence (typically chosen) is 1.96.

6.7.2 Changes in Correlation Coefficient

Authors in [21] presents a detailed study on the statistical distribution of static and dynamic power. Statistical power analysis has been performed using 45-nm SOI technology and has estimated the probability density function of full-chip power. We approximate this distribution as a normal distribution. Without loss of generality, we also assumed the total power consumption of the system (Eg: a unicore processor) using our technique also follows normal distribution. Specifically, we assumed it to be $N(0,1)$ to simplify our analysis.

For our study, we assume the system can have (V, f) spectrum similar to one listed in Table 6.2. We also assume that all (V, f) pair to occur with equal probability i.e. equally likely. We notice that there are multiple frequencies which are mapped to more than two voltages. As discussed in Section 6.3.3, wrong (V, f) pair changes the trend in the estimated power traces and leads to error accumulation in ρ calculation, which eventually can lead the attacker to hypothesize a wrong key value. Using $N(0,1)$ as sampling distribution, for a given ρ, we have generated two different power traces which are independent and of size N. For every frequency detected, as the attacker can map it to three different voltages and we assume the probability of success to be 33%. For the rest of time, we assume the attacker maps it to wrong (V, f) pair with equal probability.

We also assume the secret key is 8 bit long and our analysis can be easily extended for other key lengths. Since there are 255 different incorrect keys and one correct key, we have generated 256

different power traces in the following fashion. Power trace corresponding to correct key is generated with ρ value of 0.3 i.e $\rho_{max} = 0.3$. Also one of the incorrect key power traces is generated with ρ value of 0.05 (representing ρ_{2nd_max}) and the remaining power traces generated with ρ values uniformly chosen from [-.02, 0.02]. We have selected ρ_{max} and ρ_{2nd_max} in such way that the difference between them is statistically significant even for $N = 100K$.

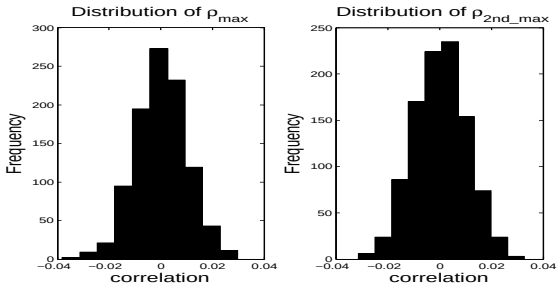

Figure 6.6 Distribution of ρ_{max} and ρ_{2nd_max} due to (V, f) pair misprediction

As discussed in Section 6.3.3, wrong (V, f) pair changes the trend in the power traces and leads to error accumulation in ρ calculation which can lead the attacker to hypothesize a wrong key. For our study, we assumed incorrect (V, f) pair prediction lead to trend change in the original power trace for 50% of the time with 10% error in the power value. We introduced these changes in the estimated power traces and calculated the new correlation values for 1000 times and the distribution for ρ_{max} and ρ_{2nd_max} is shown in Figure 6.6. Figure 6.6(a) presents ρ distribution for the correct key and Figure 6.6(b) presents the ρ_{2nd_max} distribution which is observed for one of the 255 different keys with (V, f) pair misprediction. From the figure, we notice that mean of new ρ_{max} is smaller than original ρ_{max} by an order. Also, mean of ρ_{2nd_max} is greater than ρ_{max} distribution, implying, on average peak value of ρ is observed for a wrong key.

6.7.3 Testing changes in ρ

To test if the changes observed in ρ values are of statistical significance, we have applied Fisher's Z-transform on ρ as explained in Section 6.7.1. To explain our results, we introduce two probabilities, p_1 and p_2, which are defined as:

- p_1 = Ratio of number of times ρ_{max} is observed for correct key to total number of attempts.

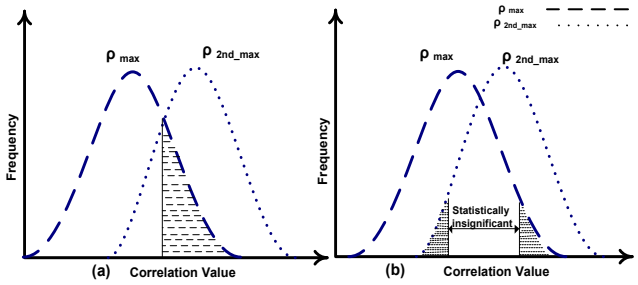

Figure 6.7 Figure highlights the area for (a) Probability p_1 (b) Probability p_2 from ρ_{max} and ρ_{2nd_max} distribution

- p_2 = Ratio of number of times ρ_{max} is observed for correct key and the difference between ρ_{max} and ρ_{2nd_max} is statistically significant to total number of attempts.

From the definition, we notice that p_1 only considers cases where $\rho_{max} > \rho_{2nd\_max}$ and doesn't take Fisher's test into account and p_2 considers cases where $\rho_{max} > \rho_{2nd\_max}$ while taking Fisher's test into account. Probability p_2 represents the probability that the attacker will observe the correlation peak for the correct key. The highlighted area in Figure 6.7 (a) represents p_1 and the highlighted area in Figure 6.7 (b) represents p_2. Table 6.3 presents the observed values of p_1 and p_2 for our study presented in Section 6.7.2. In each attempt power trace is collected for 10K inputs.

Number of attempts	p_1	p_2
1K	489/1K	24/1K
5K	2429/5K	126/5K
10K	4953/10K	258/10K

Table 6.3 Values of p_1 and p_2 for N=10K

From Table 6.3, we notice that p_1 is less than 0.5, which implies, peak value of ρ is observed for wrong key for more than 50% of the time. Also, probability p_2 is less than 3% and also it is less than p_1 by an order. This means, with our countermeasure, the attacker will observe peak correlation for the correct key for less than 3% of the time. Also, for a 10X increase in the number of traces to collect, p_2 increases by just 7%.

Reducing p_2 alone doesn't guarantee complete protection from power attacks. This is because, after multiple attempts if the probability of observing ρ_{max} for correct key is greater than all other keys, attacker still can extract the secret key and the system can be compromised. We repeated our

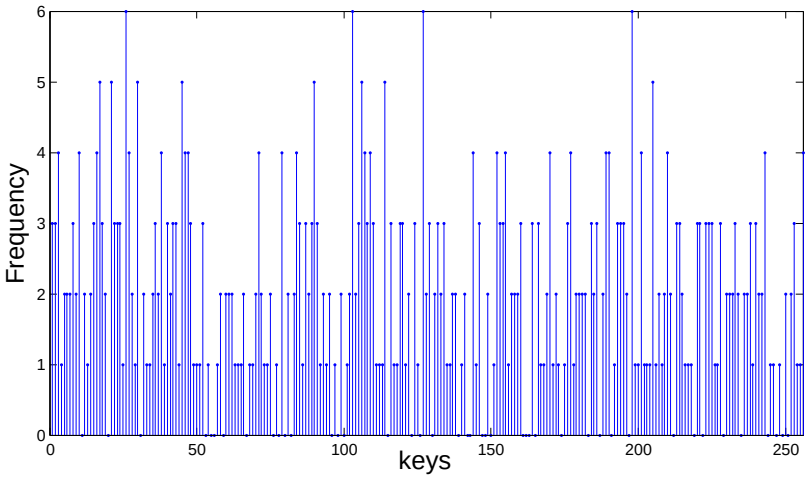

Figure 6.8 Plot showing the frequency for which maximum value of correlation is found for all possible keys in 500 attempts

experiment for 500 times and the frequency for each key to exhibit ρ_{max} is shown in Figure 6.8. We notice that the probability of observing maximum rho value for correct key (which is 148) is much smaller than the other keys. The probability for the actual key to exhibit ρ_{max} is almost smaller by three times when compared to the wrong keys exhibiting ρ_{max}.

Therefore, from above discussion, we conclude that our countermeasure decreases correlation for the correct key by an order and reduces the probability to observe ρ_{max} for the correct key. Also, our technique reduce the risk of power attacks by increasing the probability for incorrect keys to exhibit ρ_{max}.

Ring Oscillator		**S-box**			
Voltage (V)	Frequency (GHz)	Voltage (V)	Frequency (MHz)	T_{CD} (ns)	Δf (MHz)
0.8	5.26	0.8	440	0.9	260
0.9	6.54	0.9	540	0.73	320
1	8.06	1	670	0.6	400
1.1	9.21	1.1	760	0.5	450
1.2	10	1.2	830	0.48	500

Table 6.4 DVFS (V, f) pairs of 7-stage ring oscillator and 8-bit S-box

6.8 Experiments and Results

In this section, we present our results based on the experiments performed on 8-bit S-box. First, we measure the timing error rate of S-box and quantify its performance enhancement under AFS. Next, we estimate the (V, f) pairs of the 7-stage ring oscillator for given voltage levels. Using this data, we scale delay values of S-box across different voltage levels. Then, we present the (V, f) spectrum available for S-box under DVFS approach. We also present the enhanced (V, f) spectrum available for S-box under AFS and present how such enhanced (V, f) spectrum reduce the risk of power attacks.

6.8.1 Performance enhancement under AFS

For this study, we have modeled 8-bit S-box, part of AES, as a combinational circuit. Input data is logically XORed with the key and the net result is provided as input to the S-box. For this circuit, RTL level model is developed and synthesized using the 45nm OSU standard cell library [72]. Timing-annotated gate level simulations are carried out by extracting timing information in standard delay format (SDF), and back annotating delay information on the design. From static timing analysis reports, we estimated the worst case clock period to be 1.5ns and T_{CD} to be 0.56ns. Also estimated area overhead for integration of timing error detection circuits is around 3%.

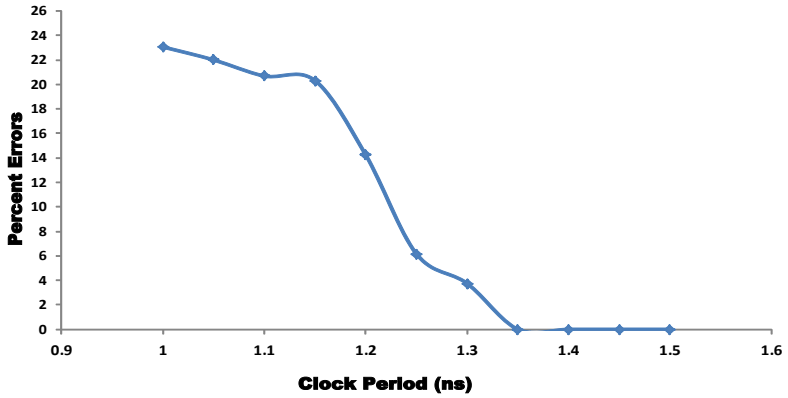

Figure 6.9 Timing error rate versus the clock period for S-box

A key factor that limits the frequency scaling under AFS is the timing error rate. As frequency is scaled higher beyond the worst case estimate, the number of input combinations that result in delays greater than the new clock period also increases. So, the timing error rate has been measured to gauge

the performance improvements provided by AFS approach. At each clock frequency, the total errors are counted for an execution run of 10,000 cycles and clock period is changed from 1.5ns to 1ns in steps of 0.05ns. Voltage of the circuit has been kept constant at 1V. Figure 6.9 depicts the percentage of cycles that produce an error for different clock periods. As shown, although the worst case delay was estimated at 1.5ns, timing errors do not begin to occur until a period of 1.35ns. Inspite of increasing timing error rate, the rate of increase in clock frequency is more than the rate of increase in timing error rate. For clock period at 1ns, timing error is about 23%, giving a net performance enhancement of 22% over the worst-case estimate. Even though cryptographic circuit outputs have equal toggle probability, our countermeasure exploits the timing behavior of the output signals. As demonstrated in our experiment, AES S-box benefits from AFS by improving the clock frequency by almost 33%. Also, S-box contamination delay is greater than 0.5ns and therefore it can be operated at 1ns without constraining the shorter paths.

6.8.2 Enhancement of (V, f) spectrum

To scale delay values across different voltage levels, we measured the worst-case frequency of a 7-stage ring oscillator for different voltage levels. It has been assumed the S-box will have seven levels of combinational logic. Using a ring oscillator, delay of an inverter can be accurately modeled even in the presence of temperature variations. Also, it reflects how gate delay varies with the operating voltage. Using the 7-stage ring oscillator, the *worst-case* frequency for the single stage was determined from 0.8 V to 1.2 V in steps of 0.1 V. These results are shown in the first two columns of Table 6.4. The worst-case delay through the S-box was then determined by appropriately scaling these values, shown in column 4. Finally the timing constraints from Section 6.5 and the values of T_{CD} given in column 5 were used to compute the maximum frequency increase (Δf) that can be realized using our AFS approach, given in column 6. Since T_{CD} of S-box is greater than 0.5ns, the frequency swing Δf, presented in column 5, is achievable with no additional area overhead.

From Table 6.4, we notice that for S-box operating at (1V, 670 MHz), frequency swing of 400 MHz is possible under AFS. This implies that two new operating pairs - (1V, 760MHz) and (1V, 830MHz) can be created using AFS. Now using the Δf values given in Table 6.4, generic example of mapping

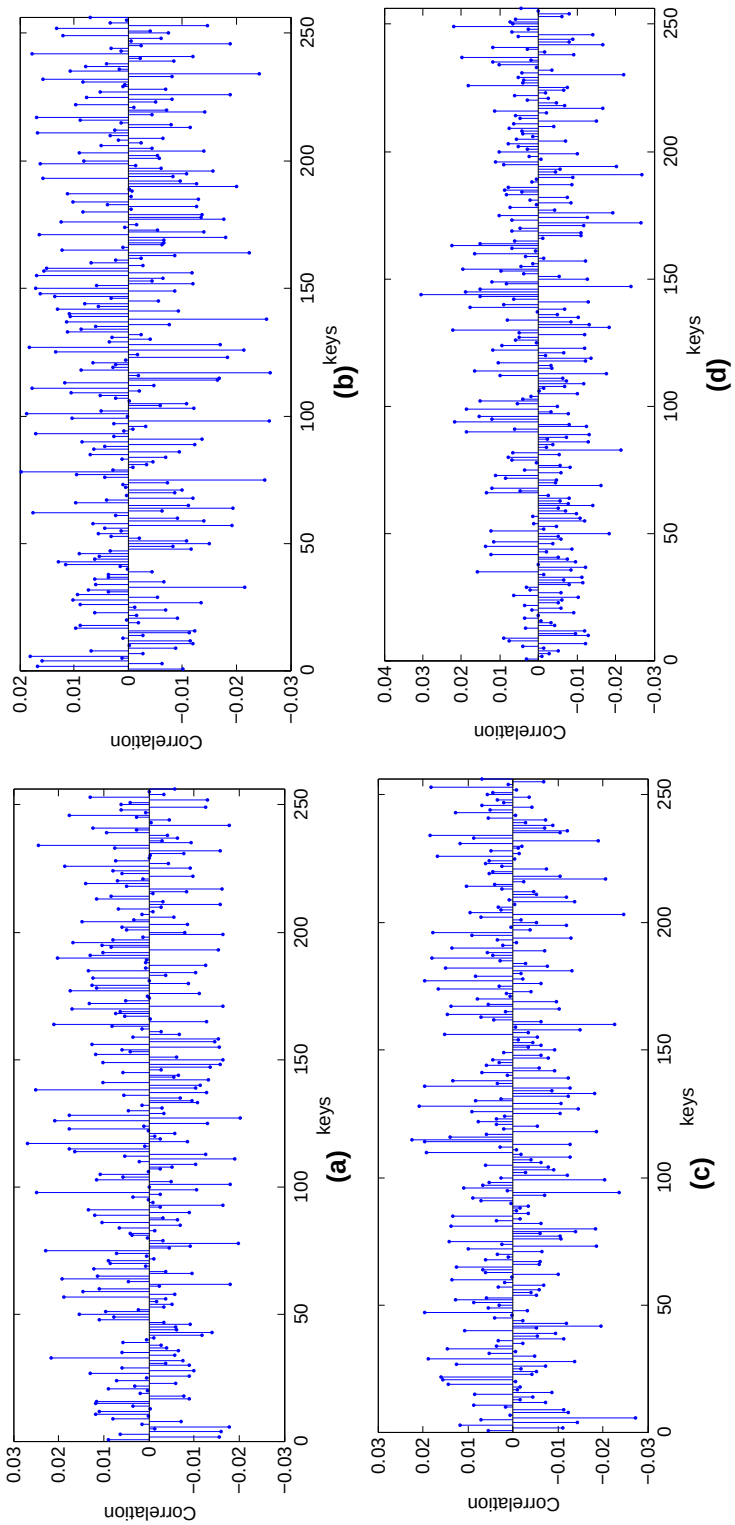

Figure 6.10 Correlation values for all keys with the voltage schemes (a) High Voltage (HV) (b) Low Voltage (LV) (c) Median Voltage (MV) and (d) Random Voltage (RV) used for (V, f) pair estimation

multiple frequencies to multiple voltage levels in Table 6.2 is transformed to specific voltages and frequencies, for an AES S-box implemented using 45 nm fabrication technology [72]. It should be noted while this example only maps three frequency to each voltage level, AFS is not inherently limited to this number. Table 6.5 lists (V, f) pairs supported by AFS that break one-to-one mapping between voltage and frequency. From Table 6.5, we notice that from a total of five DVFS (V, f) pairs, AFS has enhanced (V, f) spectrum to nine pairs. It should be noted that AFS (V, f) pairs listed in Table 6.5 are the only pairs that break one-to-one mapping between DVFS (V, f) pairs and AFS is not inherently limited to just these pairs. AFS can support many other (V, f) pairs, for example (1.2V, 1330 MHz). But these pairs leak voltage information as the frequency associated with them in one-to-one mapped.

Voltage (V)	DVFS (f in MHz)	AFS (f in MHz)
0.8	[0.8, 440]	[0.8, 670]
0.9	[0.9, 540]	[0.9, 670], [0.9, 760]
1	[1, 670]	[1, 670],[1, 760], [1, 830]
1.1	[1.1, 760]	[1.1, 760], [1.1, 830]
1.2	[1.2, 830]	[1.2, 830]

Table 6.5 Enhanced (V, f) spectrum for S-box under AFS

6.8.3 Effectiveness of the countermeasure

To test the effectiveness of our countermeasure, we have mounted an attack on 8-bit S-box without any countermeasure. The system is shown in Figure 6.11. Although the chosen system is small when compared to a complete crypto system, it enables us to test the effectiveness in a shorter simulation time. Moreover if a countermeasure is proven to be strong on a system like S-box, it will be more effective in preventing power attacks on big system like AES.

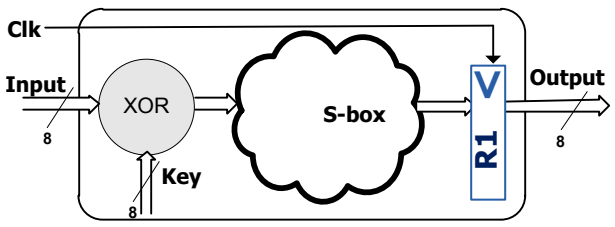

Figure 6.11 Block diagram of the attacked system

Spice-level transistor simulations are conducted on a 8-bit S-box implemented using 45nm CMOS

implementation and the total current is monitored for the attack interval. Hypothetical power consumption is chosen as function of toggle activity at register R1 and the results are shown in Figure 6.12. We have added noise sampled from $N(0,1)$ to the actual power trace. With the experimental setup chosen, we are able to detect the correct key with just 10,000 input texts.

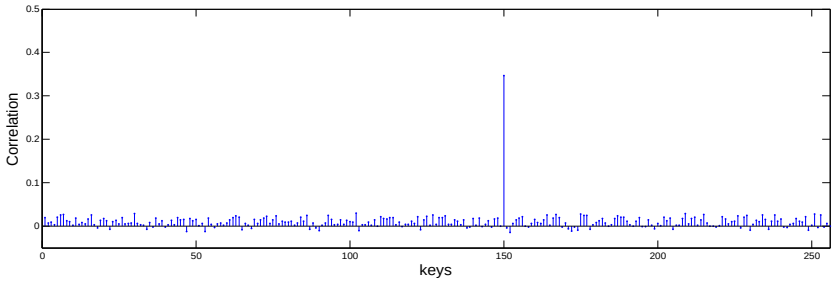

Figure 6.12 Successful power analysis attack on AES S-box without any countermeasure

To test the effectiveness of our countermeasure, we have used four schemes for estimating operating (V, f) pair of S-box. They are 1) High Voltage (HV), 2) Low Voltage (LV), 3) Median Voltage (MV) and 4) Random Voltage (RV). For a given frequency, HV scheme uses the highest possible voltage, LV scheme uses the lowest possible frequency, MV scheme uses the median frequency, and RV scheme uniformly randomly selects one of the possible voltages for (V, f) pair estimation. For our DVFS experiments, we used (V, f) spectrum presented in Table 6.5. Also, for all schemes,we assume that all (V, f) pair to occur with equal probability i.e. equally likely. In our experiments, we have assumed (V, f) pair is changed every clock cycle. In our attempt using 10,000 plain texts, the four voltage schemes described couldn't reveal the secret key and the correlation values are shown in Figure 6.10. From the figure, we notice that maximum correlation value is less than 0.04 and all possible key correlation values are in range [-0.03, 0.04]. Also, none of the schemes has revealed the secret key even for one million plain texts.

For static voltage schemes, with increasing the number of plain texts, error in correlation coefficient also increases. Error accumulation is dependent on (V, f) pair misprediction. Increasing number of traces has little impact on them. But with dynamic voltage scheme, RV, secret key can be revealed if the (V, f) misprediction doesn't change the trend in the estimated power trace. Therefore, we have repeated the (V, f) pair estimation using RV scheme for 500 times. Figure 6.13 present the frequency

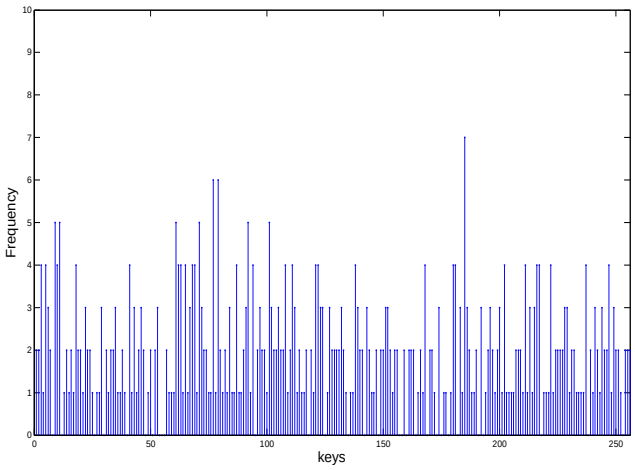

Figure 6.13 Plot showing the frequency for which ρ_{max} is found for of all possible keys using RV voltage scheme for AES S-box

for all possible keys to exhibit peak correlation. In our experiments, we have found that key value 184 (which is an incorrect key) has high probability to exhibit ρ_{max} when compared with all possible keys. It is found seven times out of 500 attempts. Also, actual key (key value 150) is found only two times out of 500 attempts and this probability is 3.5X times less than the probability for which ρ_{max} is found for incorrect key value 184.

6.8.4 Comparing with other countermeasures

Using reliable and aggressive designs, we overcome the limitation of RDVFS and also enable systems to operate beyond worst-case estimates and enable *typical-case* operation. As pointed out by Brooks, this gap is becoming huge in today's technology [33].

Error recovery of reliable and aggressive system involves loading golden data present in BACKUP registers (as shown in Figure 3.1(a)) and doesn't involve recomputation. Therefore, power overheads do not increase heavily with increase in operating timing error rates, as data loading is a low power operation when compared to data recomputation. It was shown that, area overheads of aggressive designs are only about 3% for a microprocessor implementation [25]. Therefore, at chip-level, we can expect area and power overhead for a cryptosystem also to be in this order. Other countermeasures involve logic replication, as presented in Section 6.2.2. This will result in increased combinational area. Our technique involves adding BACKUP registers for timing critical paths resulting only in slight

increase in sequential area. Also, for our experiments on S-box presented in Section 6.8.1, we didn't have to increase contamination delay, implying its inherent support for overclocking

For successful power analysis attacks relatively low clock frequency is needed in the actual experiments [89]. Therefore countermeasures like AFS, which increases clock frequency beyond worst-case estimate, are preferred. Even though our technique hides (V, f) pair from the attacker, it differs from other hiding techniques. Secret key can be recovered if actual (V, f) pair information is made available. Therefore, similar to techniques developed in [31, 8], our solution can also be used to detect watermarks in power traces for intellectual property (IP) detection. This is contrary to all other countermeasures where IP detection is not possible after the countermeasure deployment.

Recent developments in information technologies demand secure transmission of data, often, at high encryption rates that require hardware implementation of the security algorithms/protocols. Even though power attacks have been demonstrated practical on systems like smart cards, they still pose a threat to complex systems as well like microprocessors, which attempt to offload crypto operations to dedicated hardware units like a co-processors or reconfigurable hardware. Our solution is applicable to both custom ASIC and FPGA solutions. Also, support for multiple voltages is being widely deployed to achieve power savings. Given these future trends, our solution using reliable and aggressive designs, unify design approaches that aim to improve power, performance, security, and PVT tolerance and add a new dimension to power attack countermeasures.

Scheme	Area	Power	Performance	PVT Tolerance
Logic Style (WDDL)	3X	4X	.25X	×
Masking	3X	-	0.5X	×
RDVFS[87]	-	0.73X	0.85X	×
AFS [73]	1.03X	1.05X	1.57X	√
AVS [25]	1.03X	0.5X	0.95X	√

Table 6.6 Comparison with other schemes in terms of area, power, performance, and PVT tolerance

6.9 Conclusion

In this chapter, we present an effective countermeasure to counteract DPA/CPA based attacks by employing reliable and aggressive designs. Salient feature of our approach lays in enabling systems to

operate beyond the worst-case estimates while breaking the one-to-one relationship between the voltage and the frequency of synchronous circuits. Our technique, Aggressive frequency scaling (AFS), also increases the entropy of power traces and pushes the performance to higher levels by exploiting data heterogeneity. Also, our technique reduces the probability to observe maximum correlation value for the correct key and prevents power attacks by increasing the probability for incorrect keys to exhibit maximum correlation value. For the experiments conducted on 8-bit S-box implemented using 45nm CMOS technology, the size of (V, f) spectrum has increased by 1.8X times while offering 22% performance improvement over the *worst-case* estimate. Also, our technique has decreased the correlation for correct key by an order and has increased the probability by almost 3.5X times for wrong keys, when compared with the original key, to exhibit maximum correlation.

CHAPTER 7. CONCLUSION AND FUTURE WORK

Advances in semiconductor technology have led to profound integration of devices into many aspects of our lives. Along with explosive growth, emphasis on performance and power efficiency also continue to increase to lead a smoother integration of devices of different form factors. As CMOS device scaling is reaching its limits, reliable and aggressive systems has the potential and capability to extend the roadmaps of all computing systems. Also, fabrication technology continues to produce faster transistors at smaller geometries, the extraction of their full computing potential has become increasingly challenging due to PVT variation. Technology scaling has also radically impacted the soft error occurrence and fault tolerance is becoming increasingly important due to their increasing susceptibility to soft errors.

In this work, we have developed two efficient soft error mitigation schemes, SEM and STEM, considering the approach of multiple clocking of data for tolerating soft errors in combinational logic. Both these techniques remove the error detection overhead from the circuit critical path. These specialized register cells provide near 100% fault tolerance against transient faults. Our schemes tolerate fast transient noise pulses, which is the principal characteristic of SETs. Both our schemes have no significant performance overhead during error-free operation. SEM cells are capable of ignoring false positives and recovers from errors using *in-situ* fast recovery avoiding recomputation. STEM cells has soft error mitigation characteristics similar to SEM. STEM cells allow reliable dynamic overclocking and are capable of tolerating timing errors as well. We also propose a scheme to manage phase shifts between clocks locally with constant delay values. Such an approach increases the possible frequency settings for aggressively clocked designs and also minimizes the clock routing resources.

Task scheduling continue to play an important role in design of real-time systems and also in high performance computing systems. In this work, we propose a new slack reclamation algorithm, aggres-

sive dynamic and voltage scaling (ADVFS), using reliable and aggressive systems. ADVFS exploits the enhanced voltage frequency spectrum offered by reliable and aggressive designs for improving energy efficiency. We also provide guidelines on how to manage the enhanced voltage frequency spectrum and the constraints required for limiting the maximum attainable under ADVFS. We also provide formal proofs to show that significant energy savings are attainable either by using single frequency or by linear combination of frequencies. Our scheme can be integrated as post processing step with any of the existing static and dynamic task scheduling algorithms.

Along with the performance and power constraints, security also is emerging as a first class design constraint. In this work, we also present an effective countermeasure to counteract DPA/CPA based attacks by employing reliable and aggressive designs. Salient feature of our approach lays in enabling systems to operate beyond the worst-case estimates while breaking the one-to-one relationship between the voltage and the frequency of synchronous circuits. Our technique, Aggressive frequency scaling (AFS), also increases the entropy of power traces and pushes the performance to higher levels by exploiting data heterogeneity. Also, our technique reduces the probability to observe maximum correlation value for the correct key and prevents power attacks by increasing the probability for incorrect keys to exhibit maximum correlation value.

Overall, our research results using circuit level timing speculation are very encouraging and looked at the possibility of unifying various system design constraints like performance, power, reliability and security. This work also lays a strong foundation that in deep submicron era systel design constraints should not be viewed in isolation and unification of them can lead to much efficient and adaptive systems. The techniques we introduced and explored in reliable and aggressive framework holds good prospect for further research.

BIBLIOGRAPHY

[1] Alam, M. (2008). Reliability- and process-variation aware design of integrated circuits. *Microelectronics Reliability*, 48(8-9):1114 – 1122. 19th European Symposium on Reliability of Electron Devices, Failure Physics and Analysis (ESREF 2008).

[2] AMD (2000). Technology, AMD white paper.

[3] Anghel, L., Nicolaidis, M., and TIMA, F. (2000). Cost reduction and evaluation of a temporary faults detectingtechnique. In *Design, Automation and Test in Europe Conference and Exhibition 2000. Proceedings*, pages 591–598.

[4] Austin, T., Bertacco, V., Blaauw, D., and Mudge, T. (2005). Opportunities and challenges for better than worst-case design. In *ASP-DAC '05: Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pages 2–7, New York, NY, USA. ACM.

[5] Austin, T. M. (1999). Diva: a reliable substrate for deep submicron microarchitecture design. In *Int'l Symp. Microarchitecture*, pages pp. 196–207.

[6] Avirneni, N. D. P. and Somani, A. K. (2012). Low overhead soft error mitigation techniques for high-performance and aggressive designs. *IEEE Transactions on Computers*, 61:488–501.

[7] Baddam, K. and Zwolinski, M. (2007). Evaluation of dynamic voltage and frequency scaling as a differential power analysis countermeasure. In *International Conference on VLSI Design held jointly with International Conference: Embedded Systems*, pages 854–862. Citeseer.

[8] Becker, G., Kasper, M., and Paar, C. (2010). Side-channel based watermarks for ip protection. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*.

[9] Bezdek, M. and Somani, A. (2006). *Utilizing timing error detection and recovery to dynamically improve superscalar processor performance*. MS Thesis, Iowa State.

[10] Bezdek, M. and Somani, A. (2010). US Patent, http://www.freepatentsonline.com/7671627.html. Number 7671627.

[11] Blomer, J., Guajardo, J., and Krummel, V. (2005). Provably secure masking of AES. In *Selected Areas in Cryptography*, pages 69–83. Springer.

[12] Borkar, S. (2007). Thousand core chips: a technology perspective. In *Proceedings of the 44th annual Design Automation Conference*, pages 746–749. ACM.

[13] Borkar, S. et al. (2003). Parameter variations and impact on circuits and microarchitecture. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 338–342, New York, NY, USA. ACM.

[14] Bowman, K. et al. (2007). Impact of die-to-die and within-die parameter variations on the throughput distribution of multi-core processors. In *Proceedings of the 2007 international symposium on Low power electronics and design*, pages 50–55. ACM New York, NY, USA.

[15] Brier, E., Clavier, C., and Olivier, F. (2004). Correlation power analysis with a leakage model. *Cryptographic Hardware and Embedded Systems-CHES 2004*, pages 135–152.

[16] Bucci, M., Luzzi, R., Guglielmo, M., Trifiletti, A., AG, I., and Graz, A. (2005). A countermeasure against differential power analysis based on random delay insertion. In *IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005*, pages 3547–3550.

[17] Burger, D. and Austin, T. M. (June 1997). *The SimpleScalar Tool Set, Version 2.0*. University of Wisconsin-Madison Computer Sciences Department Technical Report #1342.

[18] Chantem, T., Hu, X., and Dick, R. (2009). Online work maximization under a peak temperature constraint. In *Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design*, pages 105–110. ACM.

[19] Chaparro, P., González, J., Magklis, G., Qiong, C., and González, A. (2007). Understanding the thermal implications of multi-core architectures. *Parallel and Distributed Systems, IEEE Transactions on*, 18(8):1055–1065.

[20] Chen, C. (1992). Symbol error-correcting codes for computer memory systems. *IEEE transactions on computers*, 41(2):252–256.

[21] Chen, H., Neely, S., Xiong, J., Zolotov, V., and Visweswariah, C. (2009). Statistical Power Analysis for High-Performance Processors. *Journal of Low Power Electronics*, 5(1):70–76.

[22] Chen, J., Yang, C., Kuo, T., and Shih, C. (2007). Energy-efficient real-time task scheduling in multiprocessor dvs systems. In *Design Automation Conference, 2007. ASP-DAC'07. Asia and South Pacific*, pages 342–349. IEEE.

[23] Chen, M. and Orailoglu, A. (2007). Improving circuit robustness with cost-effective soft-error-tolerant sequential elements. In *Asian Test Symposium, 2007. ATS'07. 16th*, pages 307–312.

[24] Colwell, B. (2004). The zen of overclocking. *IEEE Compututer*, 37(3):9–12.

[25] Das, S., Roberts, D., Lee, S., Pant, S., Blaauw, D., Austin, T., Flautner, K., and Mudge, T. (2006). A self-tuning DVS processor using delay-error detection and correction. *IEEE Journal of Solid-State Circuits*, 41(4):792–804.

[26] De Langen, P. and Juurlink, B. (2007). Trade-offs between voltage scaling and processor shutdown for low-energy embedded multiprocessors. *Embedded Computer Systems: Architectures, Modeling, and Simulation*, pages 75–85.

[27] Ernst, D., Kim, N., Das, S., Pant, S., Rao, R., Pham, T., Ziesler, C., Blaauw, D., Austin, T., Flautner, K., et al. (2003). Razor: A low-power pipeline based on circuit-level timing speculation. In *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society Washington, DC, USA.

[28] Fleischmann, M. (2001). Longrun power management. In *Transmeta Corporation, Whitepaper*.

[29] Gaisler, J. (2002). A portable and fault-tolerant microprocessor based on the SPARC v8 architecture. In *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, pages 409–415.

[30] Gebotys, C., Ho, S., and Tiu, C. (2005). EM Analysis of Rijndael and ECC on a Wireless Java-based PDA. *Cryptographic Hardware and Embedded Systems–CHES 2005*, pages 250–264.

[Goodwin] Goodwin, J. Novel countermeasures and techniques for differential power analysis.

[31] Goodwin, J. and Wilson, P. (2010). Power analysis detectable watermarks for protecting intellectual property. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 2342–2345. IEEE.

[32] Greskamp, B. and Torrellas, J. (2007). Paceline: Improving single-thread performance in nanoscale cmps through core overclocking. In *PACT*, pages 213–224.

[33] Gupta, M., Rivers, J., Bose, P., Wei, G., and Brooks, D. (2009). Tribeca: design for PVT variations with local recovery and fine-grained adaptation. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 435–446. ACM.

[34] J.B., N. and Somani, A. (2001). Reese: A method of soft error detection in microprocessors. In *Dependable Systems and Networks, 2001. DSN 2001. International Conference on*, pages 401–410.

[35] Joye, M., Paillier, P., and Schoenmakers, B. (2005). On second-order differential power analysis. *Lecture notes in computer science*, 3659:293.

[36] Kim, W., Gupta, M., Wei, G.-Y., and Brooks, D. (2008). System level analysis of fast, per-core dvfs using on-chip switching regulators. In *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pages 123–134.

[37] Kimura, H., Sato, M., Hotta, Y., Boku, T., and Takahashi, D. (2006). Emprical study on reducing energy of parallel programs using slack reclamation by dvfs in a power-scalable high performance cluster. In *Cluster Computing, 2006 IEEE International Conference on*, pages 1–10. IEEE.

[38] Kobenge, S. and Yang, H. (2009). Power optimized digitally programmable delay element. In *Proceedings of the 9th WSEAS international conference on Robotics, control and manufacturing technology*, pages 21–24. World Scientific and Engineering Academy and Society (WSEAS).

[39] Kocher, P., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *Advances in Cryptology-CRYPTO99*, pages 789–789. Springer.

[40] Kocher, P., Jaffe, J., Jun, B., and Rohatgi, P. (2011). Introduction to differential power analysis. *Journal of Cryptographic Engineering*, pages 1–23.

[41] Krishna, C. and Lee, Y.-H. (2003). Voltage-clock-scaling adaptive scheduling techniques for low power in hard real-time systems. *Computers, IEEE Transactions on*, 52(12):1586 – 1593.

[42] Lee, Y. and Zomaya, A. (2009). Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling. In *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*, pages 92–99. IEEE.

[43] Li, H., Chen, T., Wu, K., and Yu, F. (2009). Quantitative Evaluation of Side-Channel Security. In *Information Processing, 2009. APCIP 2009. Asia-Pacific Conference on*, volume 2, pages 456–460. IEEE.

[44] Li, J. and Martínez, J. (2005). Power-performance considerations of parallel computing on chip multiprocessors. *ACM Transactions on Architecture and Code Optimization (TACO)*, 2(4):397–422.

[45] Mangard, S. (2002). A simple power-analysis (SPA) attack on implementations of the AES key expansion. *Lecture notes in computer science*, pages 343–358.

[46] Mangard, S., Popp, T., and Gammel, B. (2005a). Side-channel leakage of masked CMOS gates. *Topics in Cryptology–CT-RSA 2005*, pages 351–365.

[47] Mangard, S., Pramstaller, N., and Oswald, E. (2005b). Successfully attacking masked AES hardware implementations. *Cryptographic Hardware and Embedded Systems–CHES 2005*, pages 157–171.

[48] Mavis, D. and Eaton, P. (2002). Soft error rate mitigation techniques for modern microcircuits. In *Reliability Physics Symposium Proceedings, 2002. 40th Annual*, pages 216–225.

[49] Meixner, A., Bauer, M., and Sorin, D. (2007). Argus: Low-Cost, Comprehensive Error Detection in Simple Cores. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 210–222. IEEE Computer Society.

[50] Messerges, T., Dabbish, E., and Sloan, R. (2002). Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers*, pages 541–552.

[51] Michaud, P., Seznec, A., Fetis, D., Sazeides, Y., and Constantinou, T. (2007). A study of thread migration in temperature-constrained multicores. *ACM Transactions on Architecture and Code Optimization (TACO)*, 4(2):9.

[52] Mitra, S., Zhang, M., Seifert, N., Mak, T., and Kim, K. (2008). Soft error resilient system design through error correction. *VLSI-SoC: Research Trends in VLSI and Systems on Chip*, pages 143–156.

[53] Mochocki, B., Hu, X., and Quan, G. (2004). A unified approach to variable voltage scheduling for nonideal dvs processors. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 23(9):1370–1377.

[54] Moradi, A., Salmasizadeh, M., Manzuri Shalmani, M., and Eisenbarth, T. (2009). Vulnerability modeling of cryptographic hardware to power analysis attacks. *Integration, the VLSI Journal*, 42(4):468–478.

[55] Nangate (2007). Nangate. page http://www.nangate.com.

[56] Narasimham, B., Bhuva, B., Schrimpf, R., Massengill, L., Gadlage, M., Amusan, O., Holman, W., Witulski, A., Robinson, W., Black, J., et al. (2007). Characterization of digital single event transient pulse-widths in 130-nm and 90-nm cmos technologies. *IEEE Transactions on Nuclear Science*, 54(6 Part 1):2506–2511.

[57] Narendra, S., Keshavarzi, A., Bloechel, B., Borkar, S., and De, V. (2003). Forward body bias

for microprocessors in 130-nm technology generation and beyond. *IEEE Journal of Solid-State Circuits*, 38(5).

[58] NASA (2009). Nasa news report. http://science.nasa.gov/headlines/y2009/29sep_cosmicrays.htm.

[59] Nicolaidis, M. (2005). Design for soft error mitigation. *IEEE Transactions on Device and Materials Reliability*, 5(3):405–418.

[60] Normand, E. (1996). Single event upset at ground level. *IEEE Transactions on Nuclear Science*, 43(6 Part 1):2742–2750.

[61] Oswald, E., Mangard, S., Pramstaller, N., and Rijmen, V. (2005). A side-channel analysis resistant description of the AES S-box. In *Fast Software Encryption*, pages 413–423. Springer.

[62] Popp, T. and Mangard, S. (2005). Masked dual-rail pre-charge logic: DPA-resistance without routing constraints. *Lecture notes in computer science*, 3659:172.

[63] Rizvandi, N., Taheri, J., and Zomaya, A. (2011). Some observations on optimal frequency selection in dvfs-based energy consumption minimization. *Journal of Parallel and Distributed Computing*, 71(8):1154–1164.

[64] Rotem, E., Naveh, A., Moffie, M., and Mendelson, A. (2004). Analysis of thermal monitor features of the intel pentium m processor. In *TACS Workshop at ISCA-31*. Citeseer.

[65] Rotem, E., Naveh, A., Rajwan, D., Ananthakrishnan, A., and Weissmann, E. (2012). Power-management architecture of the intel microarchitecture code-named sandy bridge. *Micro, IEEE*, 32(2):20–27.

[66] Ruan, S., Namba, K., and Ito, H. (2008). Soft Error Hardened FF Capable of Detecting Wide Error Pulse. In *Proceedings of the 2008 IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems*, pages 272–280. IEEE Computer Society.

[67] Rubio, J., Rajamani, K., Rawson, F., Hanson, H., Ghiasi, S., and Keller, T. (2005). Dynamic processor overclocking for improving performance of power-constrained systems.

[68] S., K. and Somani, A. (2001). SSD: An affordable fault tolerant architecture for superscalar processors. In *Pacific Rim International Symposium on Dependable Computing*, pages 27–34.

[69] Semeraro, G., Albonesi, D., Dropsho, S., Magklis, G., Dwarkadas, S., and Scott, M. (2002). Dynamic frequency and voltage control for a multiple clock domain microarchitecture. In *35th Annual IEEE/ACM International Symposium on Microarchitecture, 2002.(MICRO-35). Proceedings*, pages 356–367.

[70] Shivakumar, P., Kistler, M., Keckler, S., Burger, D., and Alvisi, L. (2002). Modeling the effect of technology trends on the soft error rate of combinational logic. In *Dependable Systems and Networks, International Conference on*, pages 389–398.

[71] Smolens, J. C., Gold, B. T., Falsafi, B., and Hoe, J. C. (2006). Reunion: Complexity-effective multicore redundancy. In *IEEE Micro*, pages 223–234.

[72] Stine, J. et al. (2007). FreePDK: An Open-Source Variation-Aware Design Kit. In *Proc. of the 2007 IEEE Intl Conference on Microelectronic Systems Education*, pages 173–174.

[73] Subramanian, V., Bezdek, M., Avirneni, N. D., and Somani, A. (2007). Superscalar processor performance enhancement through reliable dynamic clock frequency tuning. In *DSN*, pages 196–205.

[74] Šůcha, P., Kutil, M., Sojka, M., and Hanzálek, Z. (2006). Torsche scheduling toolbox for matlab. In *IEEE Computer Aided Control Systems Design Symposium (CACSD'06)*, pages 1181–1186, Munich, Germany.

[75] Suzuki, D., Saeki, M., and Ichikawa, T. (2004). Random switching logic: a countermeasure against DPA based on transition probability. *IACR ePrint, rep*, 346:2004.

[76] Tiri, K., Akmal, M., and Verbauwhede, I. (2005a). A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In *Solid-State Circuits Conference, 2002. ESSCIRC 2002. Proceedings of the 28th European*, pages 403–406. IEEE.

[77] Tiri, K., Hwang, D., Hodjat, A., Lai, B., Yang, S., Schaumont, P., and Verbauwhede, I. (2005b). Prototype IC with WDDL and Differential Routing–DPA Resistance Assessment. *Cryptographic Hardware and Embedded Systems–CHES 2005*, pages 354–365.

[78] Tiri, K. and Verbauwhede, I. (2003). Securing encryption algorithms against DPA at the logic level: Next generation smart card technology. *Cryptographic Hardware and Embedded Systems-CHES 2003*, pages 125–136.

[79] Tiri, K. and Verbauwhede, I. (2004). A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation. In *Proceedings of the conference on Design, automation and test in Europe-Volume 1*, page 10246. IEEE Computer Society.

[80] Trichina, E., Korkishko, T., and Lee, K. (2004). Small size, low power, side channel-immune AES coprocessor: design and synthesis results. *Advanced Encryption Standard–AES*, pages 113–127.

[81] Uht, A. K. (2000). Achieving typical delays in synchronous systems via timing error toleration. Technical report 032000-0100, University of Rhode Island.

[82] Uht, A. K. (2005). Uniprocessor performance enhancement through adaptive clock frequency control. *IEEE Transactions on Computers*, 54(2):132–140.

[83] Vijaykumar, T., Pomeranz, I., and Cheng, K. (2002). Transient-fault recovery using simultaneous multithreading. pages 87–98.

[84] Whitnall, C. and Oswald, E. (2011). A comprehensive evaluation of mutual information analysis using a fair evaluation framework. *Advances in Cryptology–CRYPTO 2011*, pages 316–334.

[85] Wire, C. (2008). Amd's 45nm phenom ii chips overclocked to 6+ghz. http://www.crn.com/hardware/212101254.

[86] Xian, C., Lu, Y., and Li, Z. (2008). Dynamic voltage scaling for multitasking real-time systems with uncertain execution time. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 27(8):1467–1478.

[87] Yang, S., Wolf, W., Vijaykrishnan, N., Serpanos, D., and Xie, Y. (2005). Power attack resistant cryptosystem design: a dynamic voltage and frequency switching approach. In *Design, Automation and Test in Europe, 2005. Proceedings*, pages 64–69.

[88] Yu, A. and Brée, D. (2005). A clock-less implementation of the AES resists to power and timing attacks. In *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, volume 2, pages 525–532. IEEE.

[89] Zhi-bo, D., Yun, C., and Ai-dong, C. (2011). The impact of the clock frequency on the power analysis attacks. In *Internet Technology and Applications (iTAP), 2011 International Conference on*, pages 1 –4.

[90] Zhu, D., Melhem, R., and Childers, B. (2003). Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems. *Parallel and Distributed Systems, IEEE Transactions on*, 14(7):686–700.

[91] Zhuo, J. and Chakrabarti, C. (2008). Energy-efficient dynamic task scheduling algorithms for dvs systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 7(2):17.