Circuit Techniques for Adaptive and Reliable High Performance Computing

by

Bharan Giridhar

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2014

Doctoral Committee:

Professor David T. Blaauw, Chair
Senior Principal Engineer Ram K. Krishnamurthy, Intel Corporation
Professor Trevor N. Mudge
Associate Professor Kevin P. Pipe
Professor Dennis M. Sylvester

"asato mA sadgamaya, tamasomA jyotir gamaya, mrityormAamritam gamaya"

From ignorance, lead me to truth; from darkness, lead me to light; from death, lead me to immortality

I dedicate this dissertation to my parents Kamesh and Lakshmi Giridhar, and my sister Poorna. Their constant love, support and encouragement made this possible.

ACKNOWLEDGEMENTS

I offer my most sincere gratitude to my advisor, Prof. David Blaauw for his valuable guidance and support in all my research endeavors. He has been a constant source of motivation and inspiration, and the driving force behind my all achievements during graduate school. I also thank Prof. Dennis Sylvester and Prof. Trevor Mudge for playing an important role in my research at graduate school. I also thank the other committee members Dr. Ram Krishnamurthy and Prof. Kevin Pipe for providing valuable support and feedback.

I thank all my research collaborators, Prof. Chaitali Chakrabarti at Arizona State University, and industry researchers at ARM and Intel for their valuable support and feedback. Thanks to all students and other researchers at the Michigan Integrated Circuits Laboratory (MICL) for proving me with a very dynamic and stimulating environment, making my PhD journey a very valuable and memorable experience.

I thank ARM Ltd., Intel Fellowship, Rackham Predoctoral Fellowship, NSF, DoE Blackcomb and DARPA PERFECT for funding my research through the years. Special thanks to Dr. Ram Krishnamurthy and Dr. Mondira Pant from Intel for their valuable mentorship, as a part of the Intel Fellowship. Thanks to AMD Research and Intel Circuits Research Lab (CRL) for providing me wonderful research opportunities and enabling me to work with some of the best VLSI researchers in the world.

Finally, I thank my family, friends and relatives for their support and encouragement, and the Almighty for blessing me with the right opportunities and the ability to achieve my goals.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

ABSTRACT

Circuit Techniques for Adaptive and Reliable
High Performance Computing

by

Bharan Giridhar

Chair: David T. Blaauw

Increasing power density with process scaling has caused stagnation in the clock speed of modern microprocessors. Accordingly, designers have adopted message passing and shared memory based multicore architectures in order to keep up with the rapidly rising demand for computing throughput. At the same time, applications are not entirely parallel and improving single thread performance continues to remain critical. Additionally, reliability is also worsening with process scaling, and margining for failures due to process and environmental variations in modern technologies consumes an increasingly large portion of the power/performance envelope. In the wake of multicore computing, reliability of signal synchronization between the cores is also becoming increasingly critical. This forces designers to search for alternate efficient methods to improve compute performance while addressing reliability. Accordingly, this dissertation presents innovative circuit and architectural techniques for variation-tolerance, performance and reliability targeted at datapath logic, signal synchronization and memories.

Firstly, a domino logic based design style for datapath logic is presented that uses Adaptive Robustness Tuning (ART) in addition to timing speculation to provide up to 71% performance gains over conventional domino logic in $32b \times 32b$ multiplier in 65nm CMOS. Margins are reduced until functionality errors are detected, that are used to guide the tuning.

Secondly, for signal synchronization across clock domains, a new class of dynamic logic based synchronizers with single-cycle synchronization latency is presented, where pulses, rather than stable intermediate voltages cause metastability. Such pulses are amplified using skewed inverters to improve mean time between failures by $\sim 1 \times 10^6 \times$ over jamb latches and double flip-flops at 2GHz in 65nm CMOS.

Thirdly, a reconfigurable sensing scheme for 6T SRAMs is presented that employs auto-zero calibration and pre-amplification to improve sensing reliability by up to $1.2\sigma_{Vth}$ in 28nm CMOS—this increased reliability is in turn traded for $\sim$42% sensing speedup.

Finally, a main memory architecture design methodology to address reliability and power in the context of exascale computing systems is presented. Based on 3D-stacked DRAMs, the methodology co-optimizes DRAM access energy, refresh power and the increased cost of error resilience, to meet stringent power and reliability constraints.

CHAPTER 1

Introduction

This introduction briefly discusses high-performance computing and why it is an important research topic. It shows how some of the major research problems came to be and how they are related, and highlights efforts in this area related to this dissertation.

1.1 High Performance Computing

Innovations in information technology have been fueled by a continuous and extraordinary increase in computer performance. High performance computer systems can be regarded as the most powerful research instruments today. They are employed to model phenomena in various fields such as climatology, quantum chemistry, computational medicine, high-energy physics, etc. The term High Performance Computing (HPC) was originally used to describe powerful, number-crunching supercomputers for scientific applications. However, over the last 50 years, with a remarkable turnover of technologies, architectures, vendors and the usage of systems, the definition has evolved to include systems with any combination of accelerated computing capacity, superior data throughput, and the ability to aggregate substantial distributed computing power [1].

The growth for HPC machines has been steady for decades, roughly following the so called "Moore's Law" [2] —famously observed by Gordon Moore, co-founder of Intel Corporation, in 1965. In his original paper, Moore predicted that the number of transistors per integrated circuit would double every year and the speed would double every 18 months.

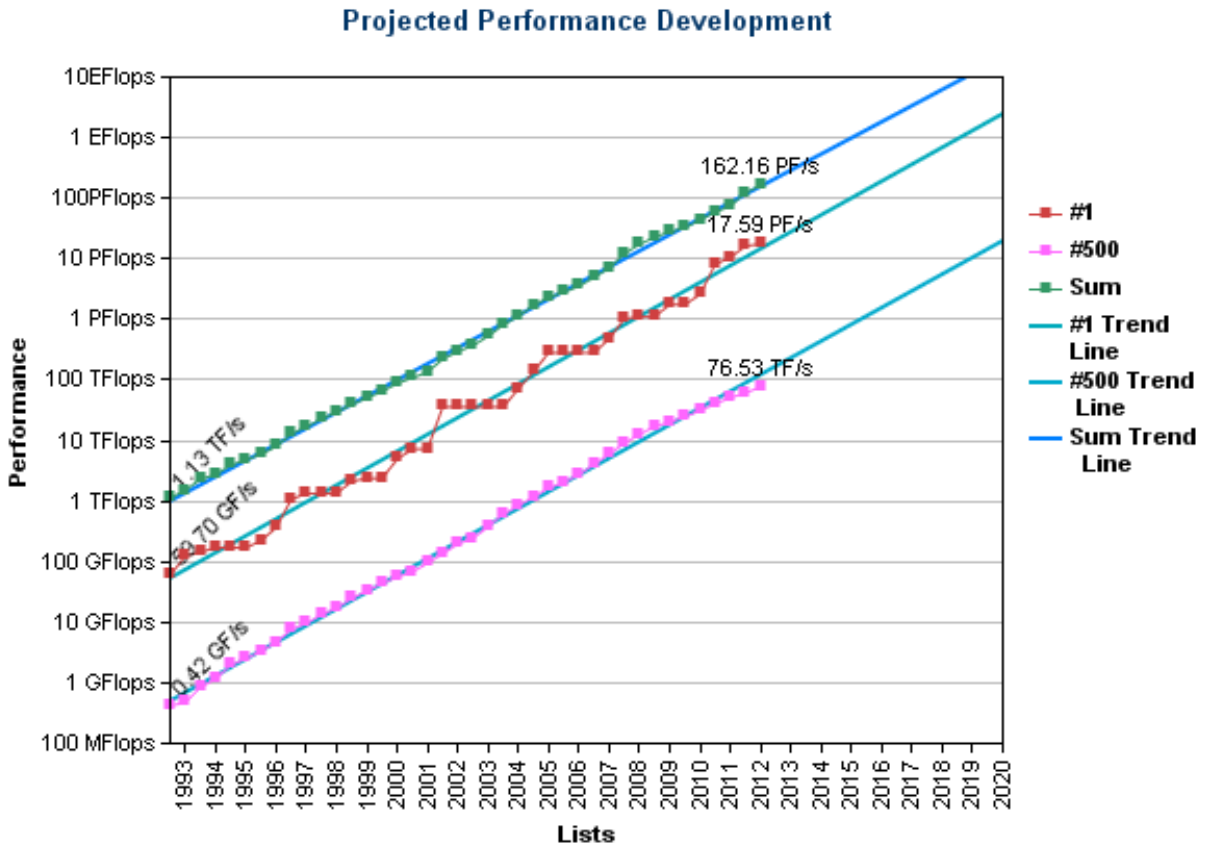


Figure 1.1: **Top 500 super computer performance projections.** Historic performance data of the top 500 performing non-distributed computers in the world. "#1" represents the fastest machine, "#500" represents the 500^{th} fastest machine, and "Sum" represents the aggregate performance of the top 500 machines. Trend lines are also drawn showing steady growth, with an exaFLOP computer projected to exist by 2018. [3]

This exponential growth, became a hallmark of computing power increases and a road-map used by the semiconductor manufacturing industry to coordinate improvements.

Figure 1.1 shows historic data for the top 500 machines in the world, showing the trends for the fastest machine, the 500th fastest machine, and the aggregate performance of the top 500 fastest machines. In contrast to specialized designs typically used through the 1990s, today's HPC systems are increasingly based on the cluster computing model [4] for cost-effectiveness—employing commodity processors, such as those from Intel and AMD instead of relying on custom processing elements. Additionally, according to these projections, an exaFLOP computer will exist by 2018-19.

Table 1.1: **Influence of scaling on MOS device characteristics.** Due to velocity saturation, device lifetime, and power density limitations, semiconductor manufacturers currently follow constant field scaling as best they can. [5]

Parameter	Sensitivity	Constant Field	Constant Voltage
Scaling Parameters			
Length: L		$1/S$	$1/S$
Width: W		$1/S$	$1/S$
Gate oxide thickness: t_{ox}		$1/S$	$1/S$
Supply voltage: V_{DD}		$1/S$	1
Threshold voltage: V_{thn}, V_{thp}		$1/S$	1
Substrate doping: N_A		S	S
Device Characteristics			
β	$(W/L)(1/t_{ox})$	S	S
Current: I_{ds}	$\beta(V_{DD} - V_{th})^2$	$1/S$	S
Resistance: R	V_{DD}/I_{DS}	1	$1/S$
Gate capacitance: C	WL/t_{ox}	$1/S$	$1/S$
Gate delay: τ	RC	$1/S$	$1/S^2$
Clock frequency: f	$1/\tau$	S	S^2
Switching energy: E	CV_{DD}^2	$1/S^3$	$1/S$
Switching power dissipation (per gate): P	Ef	$1/S^2$	S
Area (per gate): A		$1/S^2$	$1/S^2$
Switching power density	P/A	1	S^3
Switching current density	I_{ds}/A	S	S^3

1.2 Process Scaling

One of the most important enablers of Moore's law has been process scaling. Table 1.1 shows how CMOS devices perform under constant field and constant voltage style process scaling. Ideally, silicon manufacturers would like to follow constant field scaling, which improves performance but maintains the same power density.

Maintaining the same power density is important: since 2000, processors have reached the maximum power (per chip) that can be supported given the heat dissipation (*e.g.*, 125W). This is called the thermal design power (TDP). This is indicated by the flattening of the power curve in Figure 1.2. Additionally, perfect constant field scaling has not been achievable, due to increased leakage caused by reducing V_{th}, increased V_{th} variation, and V_{th} reduction via drain induced barrier lowering (DIBL). This results in an even greater power density, which designers are required to offset in other ways. This can include lowering frequency, increasing the amount of memories relative to active logic, or decreasing the die size (transistor count). This is also visible in Figure 1.2 as a decrease in the rate of frequency improvement since about 2004. As a consequence, there has been an increasing interest

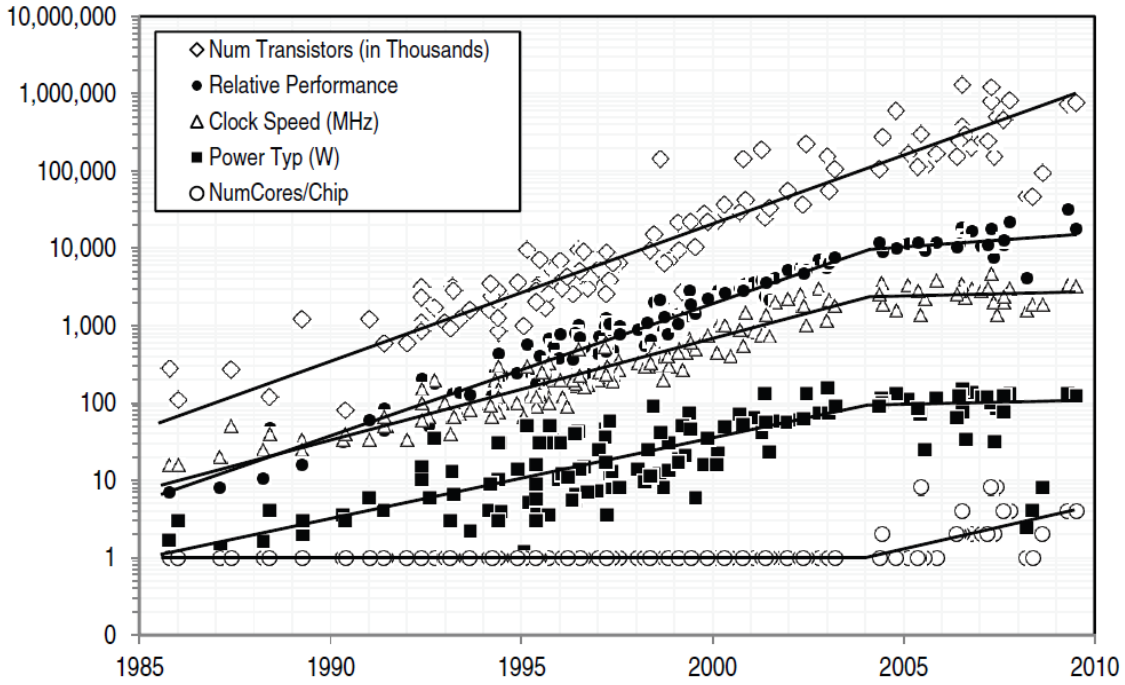


Figure 1.2: **The impact of process scaling on transistor count, power and performance [6].** While transistor count has continued to increase steadily, improvements in relative performance and clock speeds have plateaued. *Source: National Academy of Sciences*

to leverage multi-core processing since around the same time to provide performance gains depending on workload parallelism. In addition to improving performance of the individual cores, such architectures present additional challenges in the form of communication and data synchronization (signal reliability) between the cores. At the same time, with a reduction in the rate of delay scaling (that has traditionally provided "free speedup" to component designers), dynamic logic families are also making a comeback to implement speed critical paths in power constrained designs.

Process variation also increases with process scaling. Variation in effects such as line edge roughness, oxide thickness, and random dopant fluctuations do not scale directly with the feature length, causing a relative increase. Historically, systematic process variation has been of interest to semiconductor manufacturers. However, in recent years, random dopant fluctuation has emerged as a significant challenge. This is because average number of dopant atoms in modern processes is between 10 and 100 [7]. Figure 1.3 shows the increasing standard deviation of threshold voltage vs. channel length for square bulk transistors [8].

Process variation has a significant impact on circuit reliability. Various aging effects,

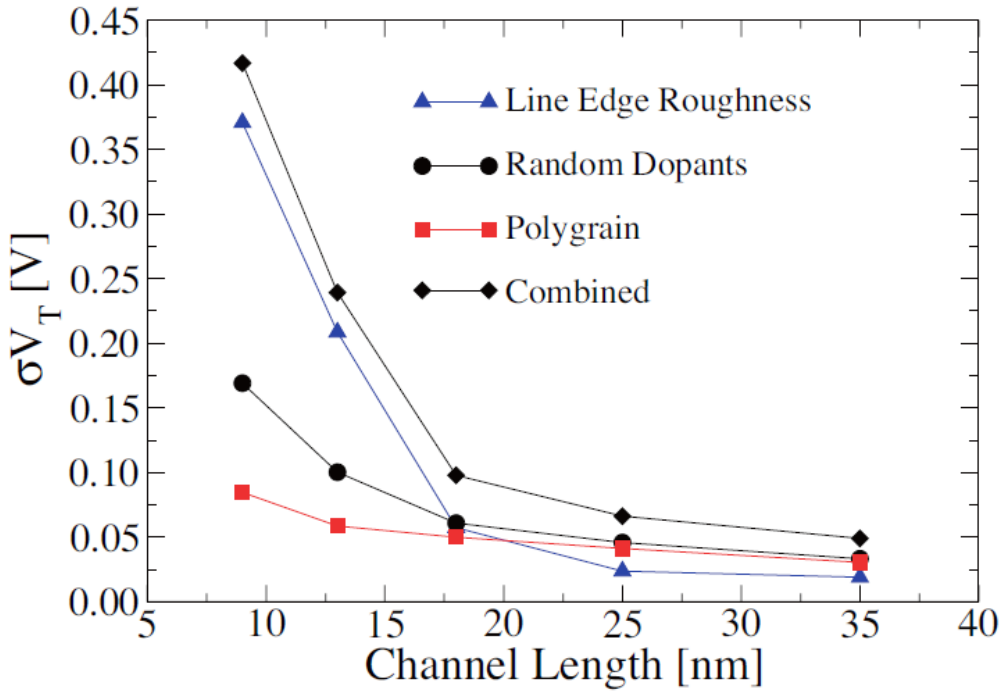


Figure 1.3: **Standard deviation of the threshold voltage vs. channel length for square bulk transistors.** Constant LER=4nm [8].

such as negative bias temperature instability and time-dependent gate oxide breakdown, cause continuous reliability degradation during circuit run-time usage. This has forced designers to increase margins for functionality and lifetime, resulting in an increase in power and/or a decrease in performance. Some of this margin can be recovered using equipment to test the performance of each chip and "bin" the chip for either power or performance. With chips having 10s of cores presently, and possibly 100s in the future, testing each core becomes expensive. Another approach is to make the circuits "adaptive", or self-adjust to the operating conditions. Thus, without the historical "free" improvement in performance as a consequence of clock frequency scaling, and with increasing process variation affecting sensitive circuits (like sense amplifiers in the memory), designers are required to develop new circuit (and architectural) techniques to improve reliability and continue performance scaling.

1.3 Adaptivity

With higher process variation in newer technologies, circuit designers have traditionally added higher timing margins at design time in order to tolerate uncertainty and ensure functionality under variation in process, voltage and temperature (PVT) as well as data and lifetime. However, under typical conditions, these margins are unnecessary and result in performance degradation.

Figure 1.4 shows how timing margins accumulate. With process scaling, such uncertainty gets a bit worse each generation. As mentioned in Section 1.2, the variation in process parameters does not scale directly with the parameters themselves, causing an increase in relative variation.

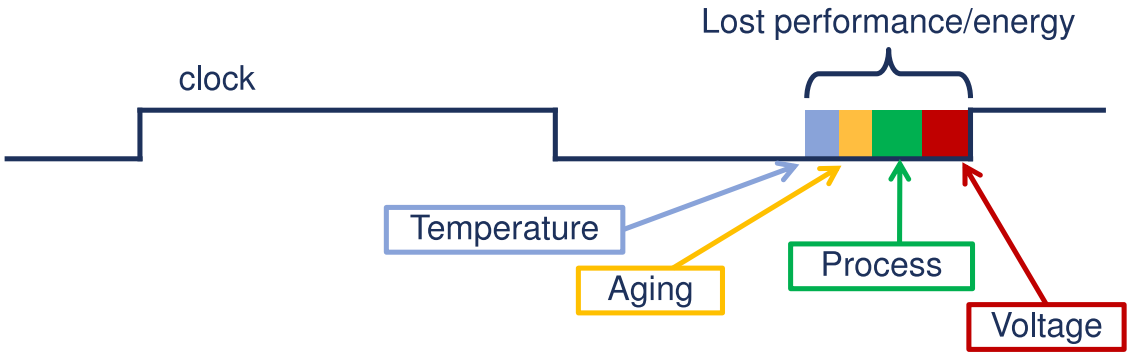


Figure 1.4: **Sources of timing margins in VLSI circuits [9].** Under typical conditions, these margins are unnecessary and result in performance degradation.

To address this, some manufacturers use a "binning" process where each integrated circuit (IC) is tested for power and performance and is assigned to a bin. Each bin is marketed and sold as a separate product,with specific voltage and frequency settings. This technique accounts for *static* variations (e.g. process) but it cannot account for *dynamic* variations (e.g., voltage and temperature) which occur during runtime. Additionally, the tester time required to perform binning is cost-intensive, so this technique is only suitable for expensive ICs.

Dynamic variations are becoming increasingly important with process scaling. As V_{DD} decreases, noise from outside sources increases relatively, and the relative effectiveness of decoupling capacitance decreases due to increased current draw to maintain the same TDP. Increases in power density also cause increases in voltage and temperature uncertainty, even

if the TDP is the same. In advanced processes it is possible for a relatively small structure to suddenly draw a large amount of current, which causes IR droop, Ldi/dt noise, and thermal hotspots. The event can be data driven, such as the case for "power viruses" run on processors, making the variation even greater since the designer often does not have control over the programs that a customer runs. An event can have complicated and cascading effects. For instance, if a driver to one half of a clock tree suddenly saw a voltage droop, then the two halves of the tree would be skewed for one or more cycles. In this case a purely local effect can have catastrophic global consequences.

Two strategies for reducing timing margins under dynamic variations are based on "detect and correct" circuits often known as "Razor" [10–14], and prediction based replica or "canary" circuits [15–18] or environmental sensors [19–21].

In a "detect and correct" style strategy or "Razor", the system operates at voltages and frequencies that may cause a timing failure, and a backup copy of any at-risk data is kept in case of emergency. If a timing failure occurs, then the backup copy of the data is restored, and operation continues. Such a scheme can account for global (e.g. process), local (e.g. hotspots) and fast (e.g. IR droop) variations. However, the technique has large overheads in terms of area and invasiveness.

The original Razor design used circuit-level data backup [22], where each flip flop had a "shadow latch" that stored known good data. Each cycle the shadow latch would need to be updated, meaning that a global rollback signal would need to be computed and distributed before the next clock cycle. It is common to have the following few clock cycles lengthened to give any undesirable electrical conditions time to pass; this also gives extra time to distribute the rollback signal. Forward progress is always made with this scheme since the program only stalls one cycle while the rollback occurs and computation continues at a guaranteed frequency.

Razor II introduced the now-prevalent architectural rollback which takes advantage of existing rollback mechanisms in modern processors [23]. Typically processors have a built-in rollback mechanism for branch mis-predicts which can be leveraged. Care must be taken to make sure that no architectural state is ever corrupted; this includes the register file, program counter, and any external memories. The rollback mechanism itself must also be

Table 1.2: **Methods of dynamic variation timing margin reduction.** Significant benefits are highlighted in *italics*. Static variation is accounted for by tester-based tuning in prediction techniques and intrinsically in others.

		Detect and Correct [10–14]	**Prediction via Canaries and Sensors [15–21]**	**Delay Slack Monitoring [9]**
Margins Reduced or Introduced	**Global Variation**	*Yes*	*Yes*	*Yes*
	Local Variation	*Yes*	No	*Yes*
	Fast Variation	*Yes*	No	No
	Introduced	None	Sensor Tracking & Sensor Variation	*TDC Accuracy*
Costs	**Area**	Large	*Small*	*Small*
	Invasiveness	Large	*Small*	*Small*
	Tester Time	*Small*	Typically Large	*Small*

considered reliable so that the program can be properly restarted. Forward progress can be stalled much longer in this technique since the rollback mechanism will undo multiple cycles of work. The lack of shadow latches at each critical flip-flop and the lack of a need for a global control signal makes this technique very popular.

Canary circuits, or "replica paths", use a string of inverters or other gates to create a known timing path that triggers every cycle. This path is used to adjust the clock frequency or system voltage such that there is never any timing failures in the main circuit. This strategy can account for changing global conditions, such as global temperature, global process variation, or global voltage scaling. It cannot respond to local variation, local hot-spots, or local supply noise, so a margin must still be included for this class of variation. Typically, the fastest delay of the canary must always be slower than the slowest critical path delay under all variation corners and conditions.

Recently, a hybrid timing speculation technique [9] was also proposed to recover some of the margins that canaries could not recover, by monitoring the delay slack of the critical paths themselves. To measure the delay slack, a time-to-digital converter (TDC) was connected to the data and clock inputs of each critical register. Note that while this technique largely avoids the design complexity introduced by traditional techniques, it introduces a new margin in terms of TDC accuracy. The technique can respond to global and local variations, but not fast variations. A summary of all these techniques is shown in Table 1.2.

While the discussed speculation techniques have been implemented for static logic, there is a need to investigate adaptivity in the context of dynamic logic families. With dynamic logic, in addition to timing speculation as performed in the discussed techniques, functionality margins can also be traded, providing an additional knob (robustness speculation) to provide additional performance benefits. However, this presents new circuit and architectural challenges in the form of detecting and recovering from gate functionality failures.

1.4 Reliability

1.4.1 Signal Reliability

With the rise in message-passing and shared memory based multi-core processing as mentioned in Section 1.2, a new challenge has emerged in the form of communication between the cores, as well as memory and cache coherence. These multicore processors are increasingly employing independent Dynamic Frequency and Voltage Scaling (DVFS) of the cores to improve energy efficiency [24, 25]. Thus data often crosses clock boundaries, and if the data from one core is not synchronized with the clock of another core, it might be changing exactly when sampled. As a result, fast and reliable on-chip communication is a key challenge in these systems due to the occurrence of metastability. Metastability is an issue because different downstream gates in a path can interpret the metastable value differently, which in turn can to lead to a system-wide functional failure.

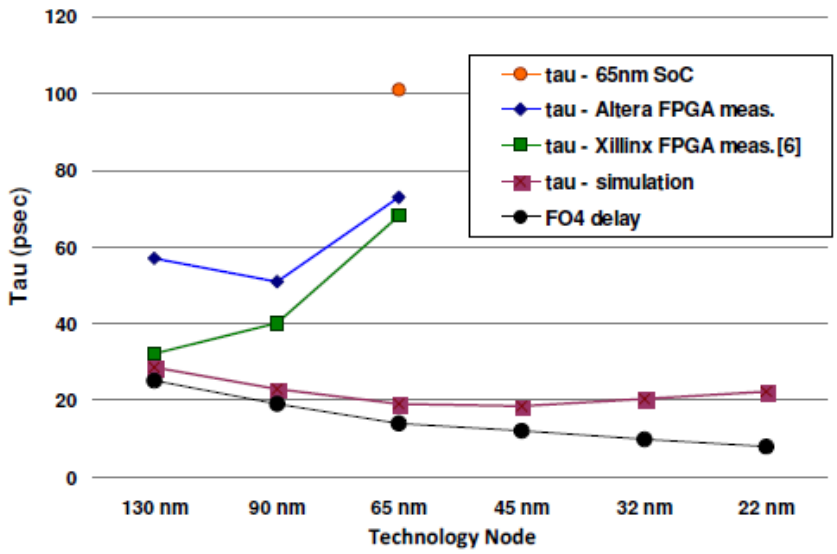


Figure 1.5: **Synchronizer performance (τ) is degrading with process scaling [26].**

Traditionally, designers have employed two to three flip-flops in series as a synchronizing element [27] at clock boundaries. The flip-flops in the path provides greater time-based amplification for the metastable signal to resolve and improves mean-time between failures (MTBF). Synchronizer performance is typically measured in terms of its resolution time constant (τ), which is a measure of the rate at which it resolves metastability [28]. With process

scaling, (τ) has been expected to scale proportionally to the gate delay 'FO4'. However, Beer *et al.* [26] recently reported that τ has actually been degrading with scaling (also shown in Figure 1.5).

One way to counter this degradation in synchronizer performance is by increasing the synchronization latency. However, this would degrade performance. Another approach has been to constrain the relationship between the clock frequencies [29, 30]. However, such an approach is not always feasible and increases clock design complexity. Thus, there is a growing demand for high MBTF, low latency synchronizing element.

1.4.2 Sensing Reliability for SRAMs

High performance SRAMs are crucial elements for high-performance microprocessor cache memory and SoC applications. A critical component of memory is the sense amplifier, which amplifies a a certain minimum signal differential (sense voltage) in the bitlines to detect and latch the stored data value. Creating the sense voltage involves discharging large parasitic capacitances on the bitlines and generally dominates the memory read access time. By reducing the minimum required sense voltage, the sense amplifier can be triggered faster, thereby improving memory read speed. Smaller sense voltage also reduces power dissipation, as the bitlines can be restored to their default state as soon as a value has been sensed and latched.

However, variation in transistor characteristics and particularly threshold voltage has emerged as a major challenge for circuit design in scaled technologies. Process variations result in increased mismatch among neighboring transistors which can affect the correct functionality of sense amplifiers by inducing offset into the cross-coupled inverter pair. In addition, increased I_{read} variation in the memory bitcell [31] further degrades sensing robustness.

The fundamental tradeoff between sensing time (time required to develop the sense voltage) and bitline read failures forces designers to heavily margin sensing time in order to guarantee sufficient sense voltage, prior to sense amplifier triggering. Previous research has proposed to improve sensing robustness by reducing offset using pre-amplification cir-

cuits [32], capacitance based offset cancellation [33, 34], and sense amplifier redundancy [35]. However, a majority of these schemes target single-ended sensing (losing the benefit of common-mode rejection), incurring up to 60% area overhead or post-silicon tuning costs. Thus, there is a need to develop high-speed, area-efficient, and robust differential sensing for 6T memories.

1.4.3 Device Reliability

In addition to signal reliability challenges introduced by architectural changes, the devices by themselves are also becoming unreliable. Device failures include hard and soft breakdown. Hard breakdown results in complete functional failure of a device. Some of these include oxide breakdown, where holes are punched in the oxide by leakage current passing through the oxide, and electromigration, where electrical current pulls metal atoms downstream until the wire is severed or it bulges causing a short. Soft breakdown causes devices to be slower, which can cause systematic timing failures, or functional failures in the case of structures like sense amplifiers and memory bitcells. Soft-breakdown failures include hot carrier effects, where electrons get injected and stuck inside the gate oxide, and negative bias temperature instability, where dangling bonds form underneath the gate oxide. In both cases the threshold voltage of the device is adversely affected [5]. In addition, high energy particle strikes (neutrons and alpha particles) also lead to failures in memory bitcells and sequential logic circuits.

Under normal operating conditions, the vast majority of devices in a system are expected to last many times longer than system's lifetime. Since a single device or wire failure can cause complete system failure, there is an inverse-exponential relationship between system size and lifetime. At the same time, process scaling is making devices smaller and less robust, while providing exponentially more of them. Together, these trends cause the lifetime margins to reduce the useful chip lifetime until it is nearly non-existent.

Accordingly, device reliability becomes even more important in the context of exascale computers, where the much higher number of components deployed will result in crippling failure rates.

1.5 Contributions of This Work

This dissertation presents circuit and architectural techniques to address adaptivity and reliability issues mentioned in this chapter. The remainder of this work is outlined below.

Chapter 2 discusses Adaptive Robustness Tuning (ART) for domino logic. It is a Razor-like speculation scheme that employs robustness speculation on domino logic in addition to timing speculation. The scheme dynamically tunes domino gates to trade surplus noise margins at nominal conditions for performance by detecting stability errors during runtime, while guaranteeing forward progress. This technique is demonstrated in a $32b \times 32b$ multiplier in 65nm CMOS technology, where it provides performance gains of up to 71% over conventional domino logic.

Chapter 3 discusses a new class of dynamic buffer based synchronizers, where pulses rather than stable intermediate voltages, cause metastability due to the one-sided operation of domino gates. This unique feature is exploited by amplifying such pulses to develop very high-MTBF, single-cycle synchronizers. This technique is demonstrated in 65nm CMOS technology, where it improves MTBF by $\sim 1 \times 10^6 \times$ ($\sim 5 \times 10^7 \times$) over jamb latches (double flip-flops) at 2GHz. A new technique to experimentally measure metastability in silicon is also proposed and used to measure results.

Chapter 4 discusses a variation tolerant sensing scheme targeting high performance 6T SRAMs. The scheme reconfigures a conventional sensing topology to additionally perform auto-zeroing based offset compensation, and bitline droop pre-amplification. The scheme is implemented in 28nm CMOS, where sensing reliability is improved by $1.2\sigma_{Vth}$ without added area overhead. This increased robustness is in turn traded for performance, providing up to 42% sensing speed improvement and 10% lower sensing power at 1.8GHz.

Chapter 5 discusses a main memory architecture design methodology to meet power and reliability for exascale computers. The methodology utilizes a 3D-stacked DRAM and proposes several optimizations to improve access energy and refresh power. In addition, a new fault tolerance strategy is proposed to combat soft and hard errors. The final design is obtained by co-optimizing error correction cost, access energy and refresh power. The resulting 3D-stacked memory uses a page size of 4kb and consumes 5pJ/bit. This is equivalent

to 4.7MW for a 100PB memory, which is well within the system power target (20MW), and also resilient to errors.

Chapter 6 summarizes the contributions of this dissertation and discusses some possible future directions.

CHAPTER 2

ART: Adaptive Robustness Tuning for High-Performance Domino Logic

In this chapter, a new domino logic design style called Adaptive Robustness Tuning (ART) is presented which provides performance gains of up to 71% over conventional domino logic. This technique is demonstrated in a $32b \times 32b$ multiplier in 65nm CMOS technology. The design dynamically tunes domino gates to trade surplus noise margins at nominal conditions for performance by detecting stability errors during runtime while guaranteeing forward progress. It also eliminates timing margins.

2.1 Motivation

While most modern chips are constrained by power, speed-critical datapaths continue to benefit from targeted use of high-performance logic design styles [36]. Domino logic [5,37] has been the mainstay for this purpose. It offers several advantages, including fewer transistors, faster switching speeds, and no contention or glitching-based power consumption. However, increasing process variation has made conventional domino design more complex and less beneficial, forcing designers to revert back to static CMOS [38].

Domino gates require safety margins to ensure correct operation under worst-case leakage, charge-sharing, and supply noise, which degrade their performance gains over static CMOS. In a Domino gate (Figure 2.1), the NMOS evaluation stack and the clock footer are upsized to speed up the falling transition (speed-critical). Similarly, the PMOS in the output inverter is

sized considerably more than the NMOS (3-4× for a fast rising transition, also speed-critical). The precharging PMOS device is sized to meet a reasonable precharge time and the PMOS keeper is sized to meet functionality under worst-case operating conditions. Figure 2.1 shows that margining the keeper for robustness under worst-case Process, Voltage and Temperature (PVT) conditions can result in a 32% delay increase.

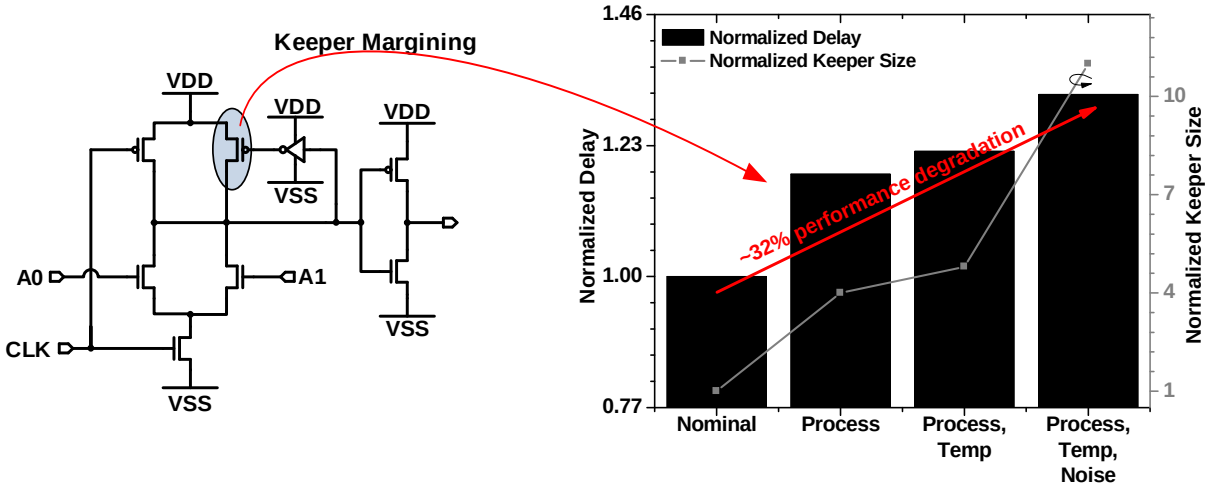


Figure 2.1: **Margining the keeper in a Domino gate for robustness under worst-case PVT conditions can degrade performance by ∼32%.** The data is simulated in a high-performance 65nm CMOS technology. The goal of ART is to be able to track PVT and trade the extra noise margins for performance under more typical conditions.

As PVT sensitivities increase with process scaling and more frequent use of low voltage operation, these margins and their resulting performance loss are expected to increase further. However, these margins can be reduced and traded for performance gains under more typical, nominal conditions. This motivates a new design style called Adaptive Robustness Tuning, (ART) [39] that shrinks robustness margins with minimal design overhead and enables performance gains of up to 34%. Similar to recently proposed adaptive approaches [40, 41], the robustness margins are reduced until functionality errors are detected. Failures are used to guide robustness tuning and are corrected to guarantee forward progress in the computation. Additionally, ART also removes timing margins, increasing the total gains up to 71% over conventional domino in a $32b \times 32b$ multiplier implemented in a 65nm CMOS process.

2.2 ART Domino Architecture

The basic concept of ART Domino is to perform two evaluations of the conventionally-sized domino gate: a fast, speculative evaluation followed by a slower, safe evaluation with sufficient margins to guarantee correct operation under worst-case conditions. The safe evaluation is performed in the background and does not impact latency of the computation. The results of the two operations are compared and in case of errors, the errant computation is flushed and the result of the safe evaluation is propagated, guaranteeing forward progress.

2.2.1 ART Domino Gate

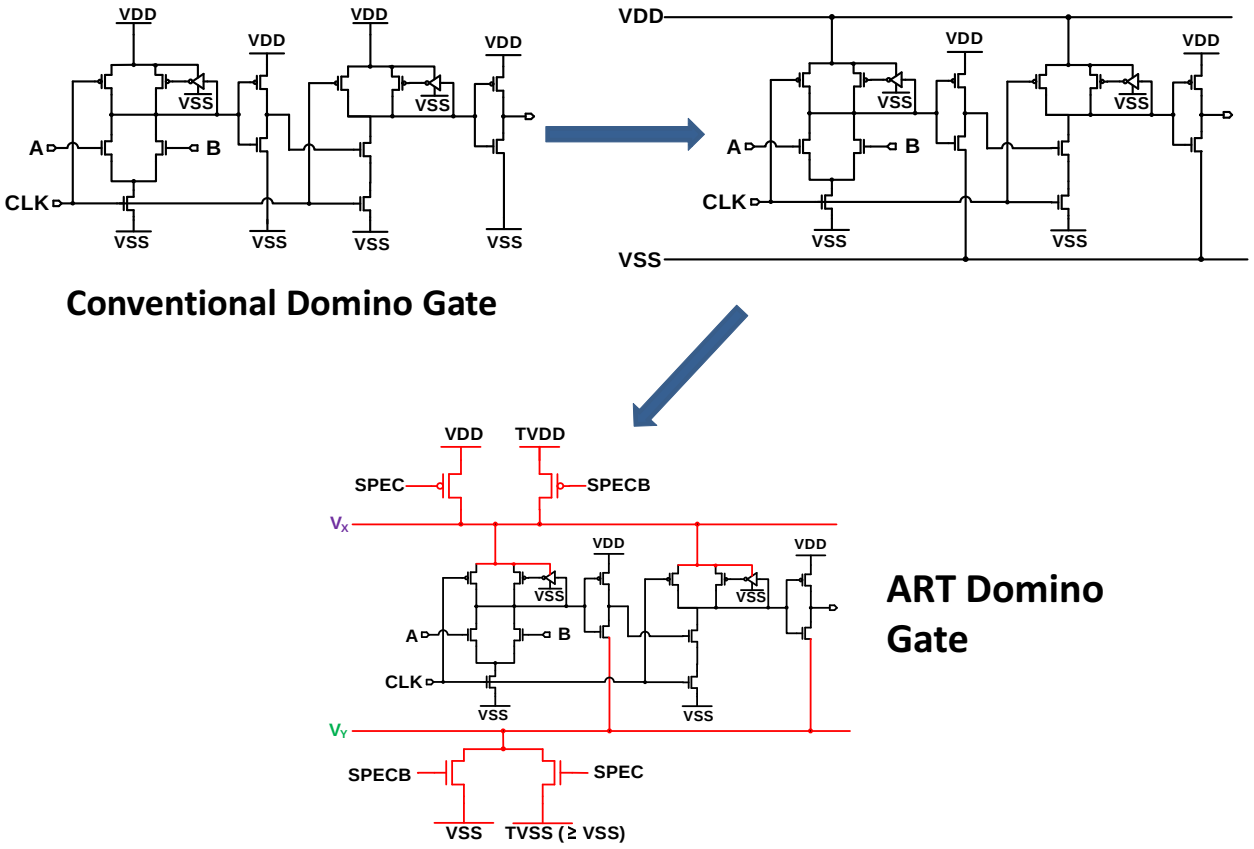


Figure 2.2: **Designing ART Domino gate starting from a conventional domino gate.** The VDD of the dynamic gate (connected to the precharge/keeper devices) and the VSS of the output inverter are converted to virtual rails controlled using headers/footers. The added headers/footers are shared across multiple gates to minimize overhead.

Figure 2.2 shows one implementation of an ART Domino gate and how it is obtained

from a conventional domino gate. The supply rails of the pull-up and pull-down networks are separated into two sets as shown and virtual supplies V_X and V_Y are introduced. In order to mitigate the area overhead of the two virtual supply rails, the two rails V_X and V_Y are laid out exactly over the primary rails VDD and VSS (which are in Metal 1) in a higher metal layer (Metal 3).

The ART Domino gate operates in four phases: (a) Speculative Precharge (SP), where the gate is precharged with margins removed (Figure 2.3); (b) Speculative Evaluate (SE), where the gate performs a fast, speculative evaluation (Figure 2.4); (c) Checker Precharge (CP), where the gate is precharged with restored margins (Figure 2.5); (d) Checker Evaluate (CE), where the gate performs a slower "always correct" evaluation (Figure 2.6). During the Speculate (SPEC) phase, precharge voltage V_X is lowered to TVDD and voltage V_Y on the output inverter is raised to TVSS speeding critical transitions at both nodes by reducing voltage swings. Raising V_Y also speeds the following gate by trading its noise margin for speed. During the Check (CHECK) phase, robustness margins are restored and a safe evaluation checks for errors. The values of TVDD and TVSS are tuned to operate the design at the edge of failure, thereby maximizing performance gains and automatically tracking PVT conditions.

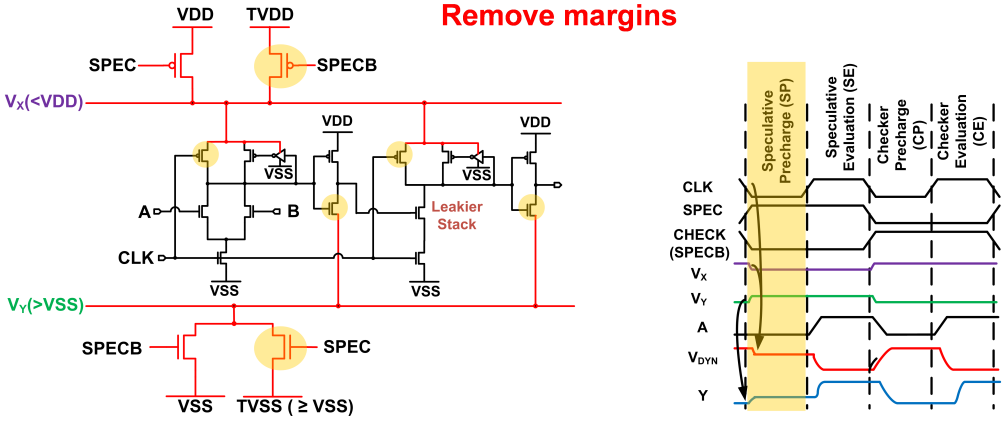


Figure 2.3: **Speculative Precharge (SP).** The gate is precharged with margins removed. V_X is lowered to TVDD and V_Y is raised to TVSS.

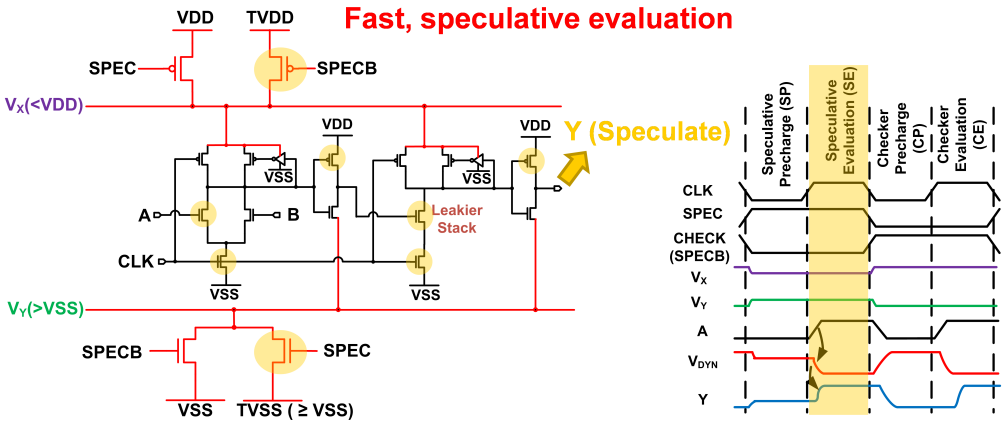


Figure 2.4: **Speculative Evaluate (SE).** The gate performs a fast, speculative evaluation. The result is recorded for in order to check for errors later.

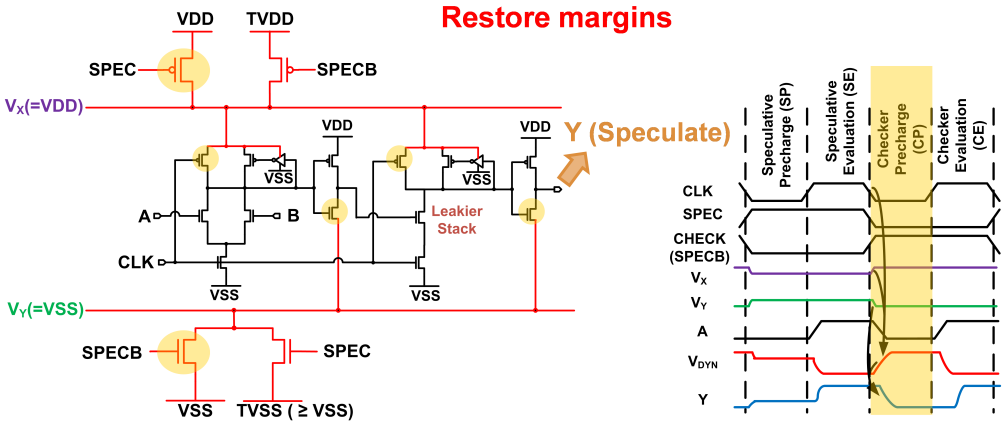


Figure 2.5: **Checker Precharge (CP).** The gate is precharged with restored margins. V_X is raised back to VDD and V_Y is lowered back to VSS.

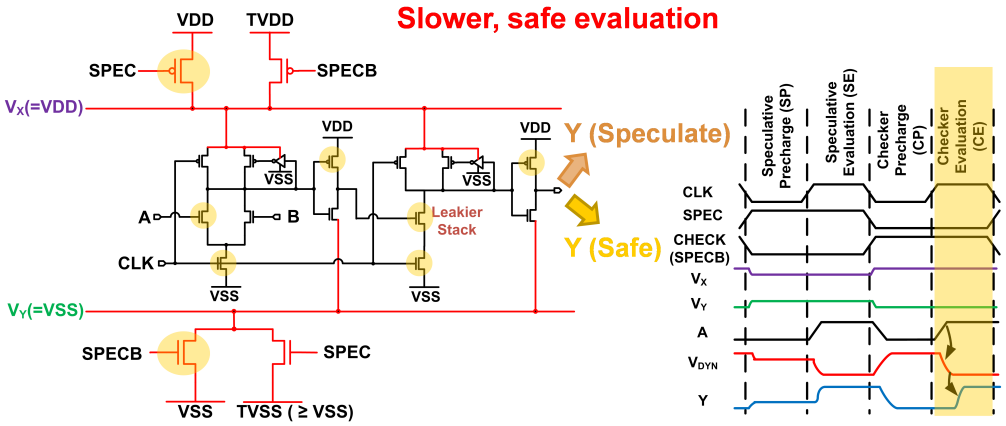


Figure 2.6: **Checker Evaluate (CE).** The gate performs a slower, "always correct" evaluation. The result of the safe evaluation is compared with the previously recorded result of the speculative evaluation to check for errors.

2.2.2 ART Pipeline and Clock Generation Design

To allow for the larger delay of the safe evaluation, we introduce a technique where we split each pipeline stage at its middle point during the safe evaluation phase (CE), effectively doubling the time for the slower, safe evaluation. Figure 2.7 shows how an ART Domino pipeline is obtained from a conventional Domino pipeline. A fully margined domino latch DOMBUF is inserted in the middle of each pipe stage. Figure 2.8 shows the circuits details in the ART Domino pipe stage. The headers/footers are shared across gates in a pipeline stage to minimize design overhead relative to conventional domino circuits.

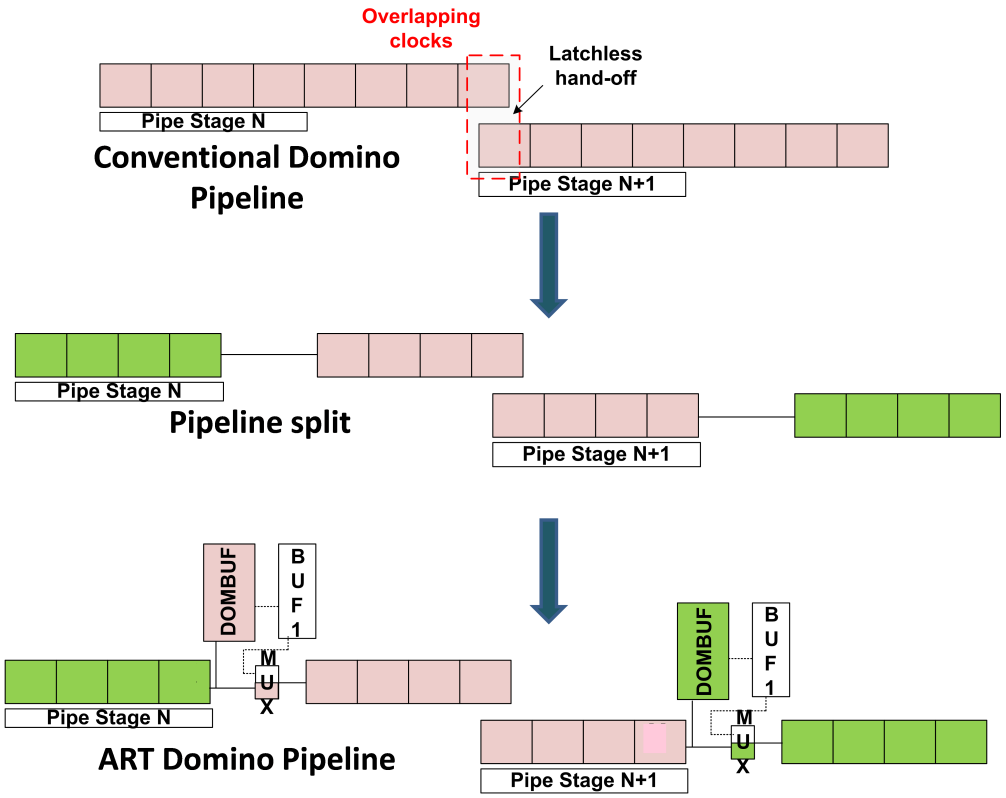


Figure 2.7: **Designing the ART Domino pipeline from a conventional Domino pipeline.** The extra logic added allows each pipeline stage to be split at its middle point during the slower, safe evaluation phase (CE).

Figure 2.9 shows the clock generation. The overlapping clock generator provides global clocks Φ1 and Φ2 which eliminate latches between pipe stages and provide skew tolerance [42]. Φ1, Φ2, and derived clocks Φ3 and Φ4 determine the required four phases for each stage. Φ3 and Φ4 must meet strict skew constraints and are generated locally in each pipe

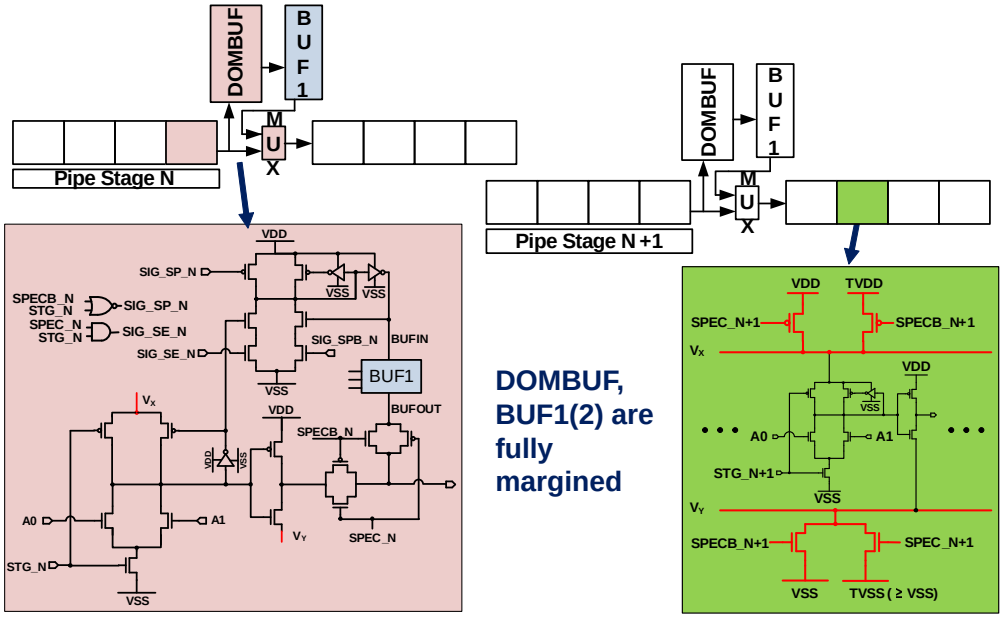


Figure 2.8: **ART Domino pipeline circuit details.** Headers/footers are shared across gates in a pipe stage. DOMBUF stores a copy of the previous gate's output during SE phase in order to split each pipe stage during the slower safe evaluation(CE).

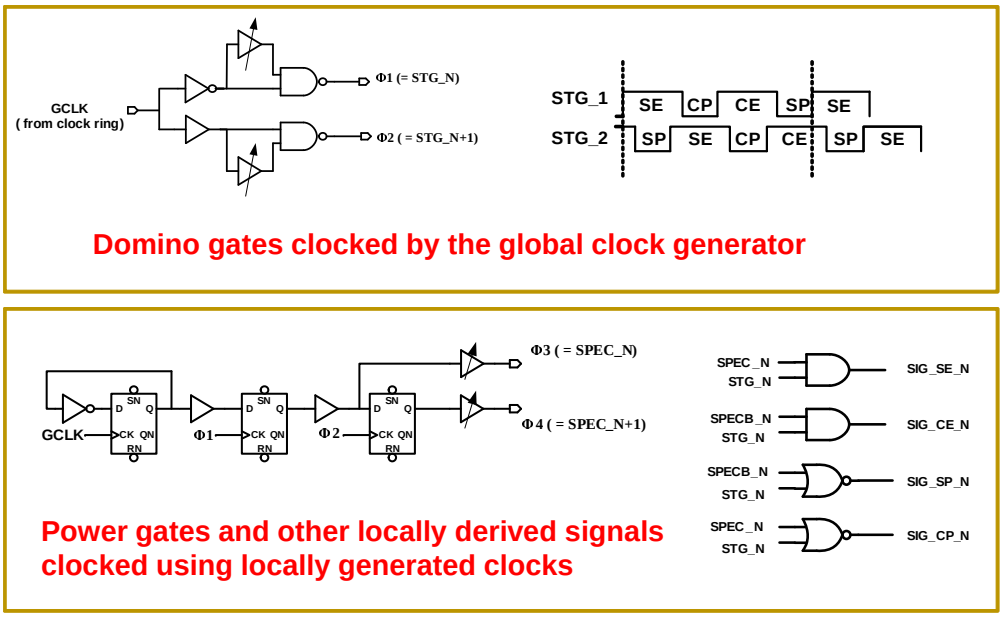


Figure 2.9: **ART Clock Generation.** Global clocks Φ1, Φ2 have relaxed overlap constraints while Φ3, Φ4 with stricter skew constraints are generated locally in each pipe stage.

stage.

Figure 2.10 contrasts the operation of the pipeline during speculation (SE) and safe (CE) evaluation phases. During SE, the domino latch DOMBUF is bypassed and the delay

overhead is limited to only a single transmission gate. The output of the gate preceding the latch is copied onto DOMBUF and during CE, this value is propagated forward cutting the stage depth by half. During CE, both halves of each pipe stage perform safe evaluations simultaneously. The second half passes its result via phase overlap to the next stage which then in turn performs simultaneous safe evaluations on its two halves.

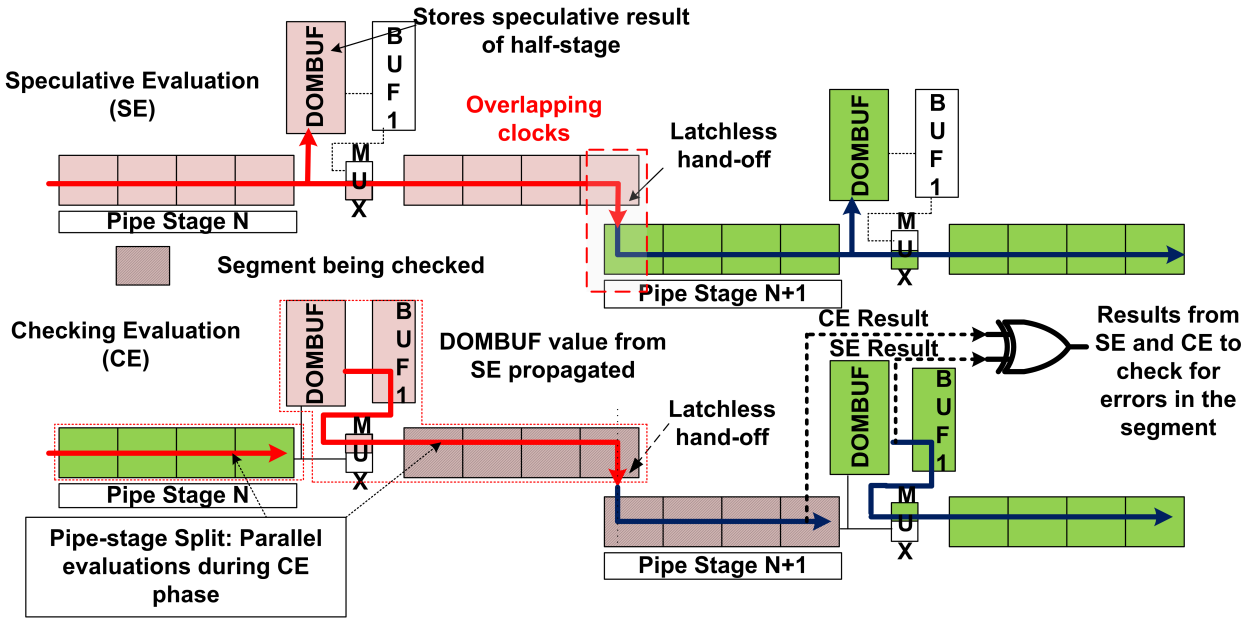


Figure 2.10: **ART Domino pipeline operations.** During SE, DOMBUF snoops on the value propagated forward though the mux. During CE, the value stored on DOMBUF is propagated forward, cutting the stage depth by half. Both halves of each stage perform safe evaluations simultaneously. The error detector at each DOMBUF checks the segment till the preceding DOMBUF for errors.

2.2.3 ART Pipeline Error Detection

The error detection in ART Domino is shown in Figure 2.11. Fully-margined gates are used in the error logic for "always correct" operation. The output of DOMBUF (the speculative result of the segment) and input of DOMBUF (the 'check' result of the segment) are copied to domino latches BUF1 and BUF2, respectively, to free DOMBUF for precharge during phase SP. The following SP and SE phases are used to XOR the two values, relaxing timing constraints on the comparison logic. Finally, all error logic is precharged during CP. Thus, errors are flagged during the next SYSCLK cycle in parallel with the subsequent set of

gate evaluations. The overall error detection timing for a four-stage ART Domino pipeline is shown in Figure 2.12.

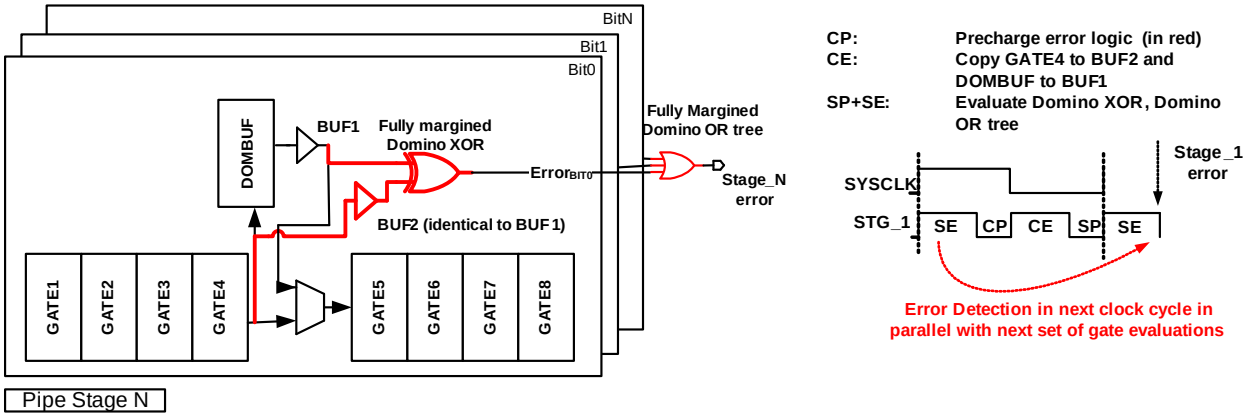


Figure 2.11: **ART Domino error detection.** Errors are flagged during the next SYSCLK cycle in parallel with the subsequent set of gate evaluations.

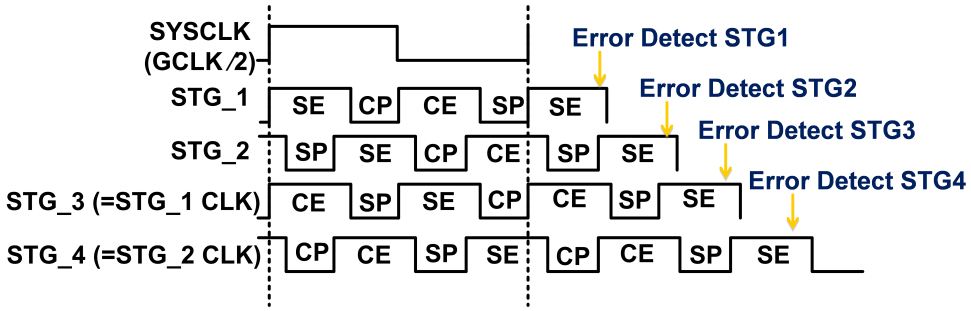


Figure 2.12: **Error detection timing in a four-stage ART Domino pipeline.** Errors are detected during the subsequent SE phase.

Note that not every gate is checked explicitly for errors. Since error detection only happens at the end of a pipeline segment, we have three scenarios as shown in Figure 2.13. In the event of segment X evaluating erroneously, BUF1 will store the incorrect value which will flag a real error during the safe evaluation phase (stand error detection). Another scenario is when multiple errors occur in the pipeline segments X and Y such that the final result is correct - flagging a false error. However, the functionality of the design is still maintained in such a case. A third scenario is when the value passed to segment X is incorrect and multiple errors in segments X and Y occur such that BUF1 stores the correct value while the final value passed out from segment Y is incorrect. Thus errors in such a scenario will not

be detected. In order to address this issue, an additional checker is added to detect errors in the segment following the last DOMBUF in the pipeline.

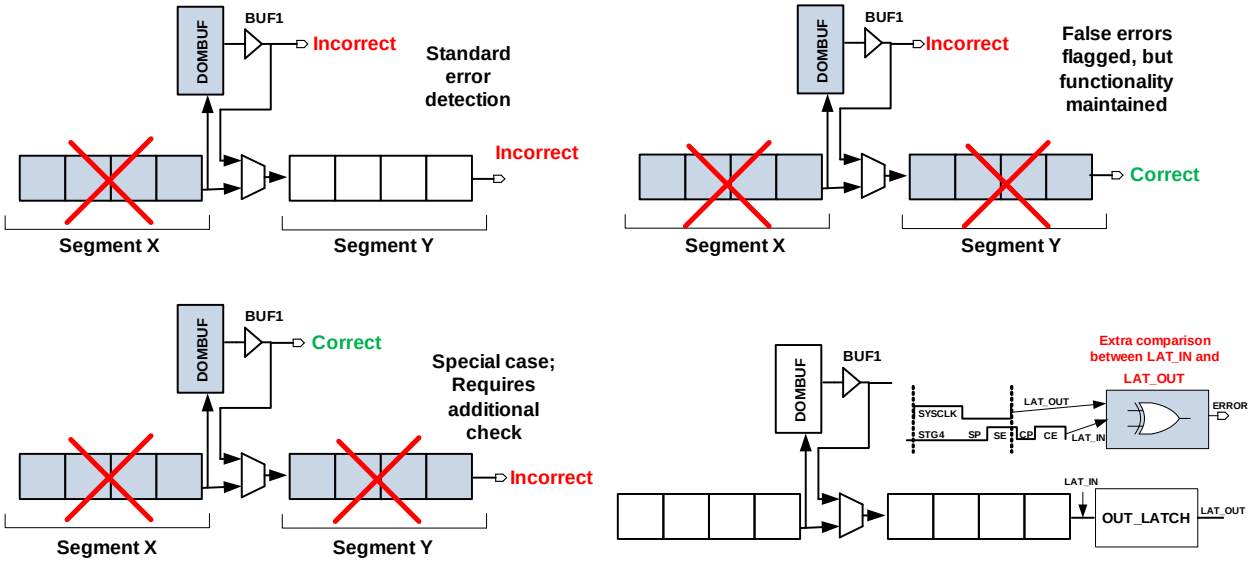


Figure 2.13: **Error detection scenarios in a ART Domino pipeline.** An additional checker is added to detect errors in the segment following the last DOMBUF in the pipeline.

2.2.4 ART Pipeline Error Recovery

We now discuss mechanisms for error recovery in the ART domino pipeline. This assumes a large system in which one of the pipelined stages (between flops) is implemented using ART Domino logic. The ART Domino logic itself is pipelined into four stages (using overlapping clocks) and the system is shown in Figure 2.14. Since errors are detected in the next SYSCLK cycle, the system will need to flush the pipeline and rollback to two cycles earlier. An alternative solution is shown in Figure 2.14. Suppose stage 2 of the ART Domino pipeline evaluates erroneously during its SE in Cycle0. This error would be detected during its SE phase in Cycle1. Once the error is detected, all registers in the system pipeline stages leading up to the ART Domino stage are stalled. Since stage 2 has not entered its CP phase, the correct value stored from the previous CE is still available and is propagated forward to the next SE of stage 3, thus ensuring forward progress. After the error has been resolved, the stall is resolved and the pipeline resumes normal operation.

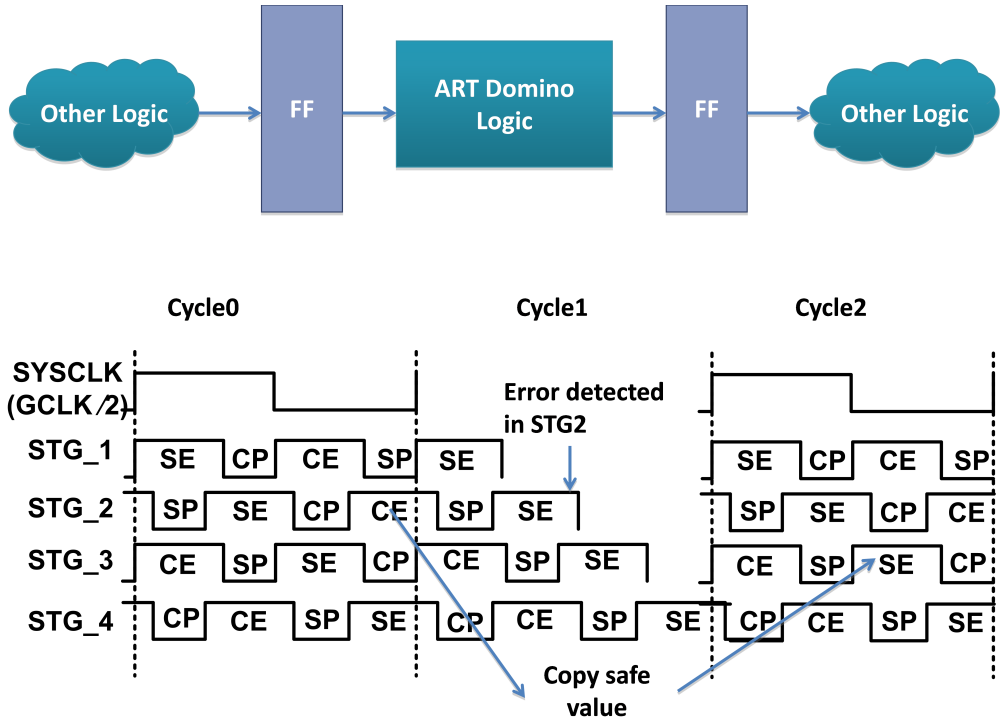


Figure 2.14: **Error recovery example in a ART Domino pipeline.** This example assumes a larger system in which ART Domino logic is implemented in one of the stages. The ART Domino logic is pipelined into four stages using overlapping clocks for latch-less pipelining. The example shows recovery in case an error occurs in stage 2 of the ART Domino pipeline.

2.2.5 Metastability in ART Domino Design

As in all design styles incorporating timing speculation, metastability can occur in ART Domino design on the error signal and cause failures in error detection. Two sources of metastability have been identified in the latch DOMBUF and we present solutions to minimize their occurrence: 1) Metastability due to genuine timing violations during SE is minimized by providing an additional half cycle of slack for the latch to evaluate. 2) Unintentional leakage in the preceding gate (Gate 4 in Fig. 2) during SE can also cause DOMBUF to go metastable. To address this, DOMBUF is given the full CP to resolve and is further latched through BUF1 during CE prior to using this value in the error and data paths, thereby minimizing the probability of metastability. Also, the intrinsic offset between the metastable input and output voltages of domino gates was found to reduce the probability of metastability to acceptable levels ($\sim 2.5 \times 10^{-21}$ or once in every 12,700 years).

2.3 ART Implementation Prototype

ART Domino logic was incorporated in a $32b \times 32b$ multiplier and implemented in 65nm CMOS technology. The design is split into two pipeline stages and four tunable voltage domains (TVDD1/TVSS1),...,(TVDD4/TVSS4). The multiplier architecture details are shown in Figure 2.15. The multiplier was an array multiplier with Booth encoding/decoding. A Radix-2 Kogge Stone adder was used as the final stage of the multiplier.

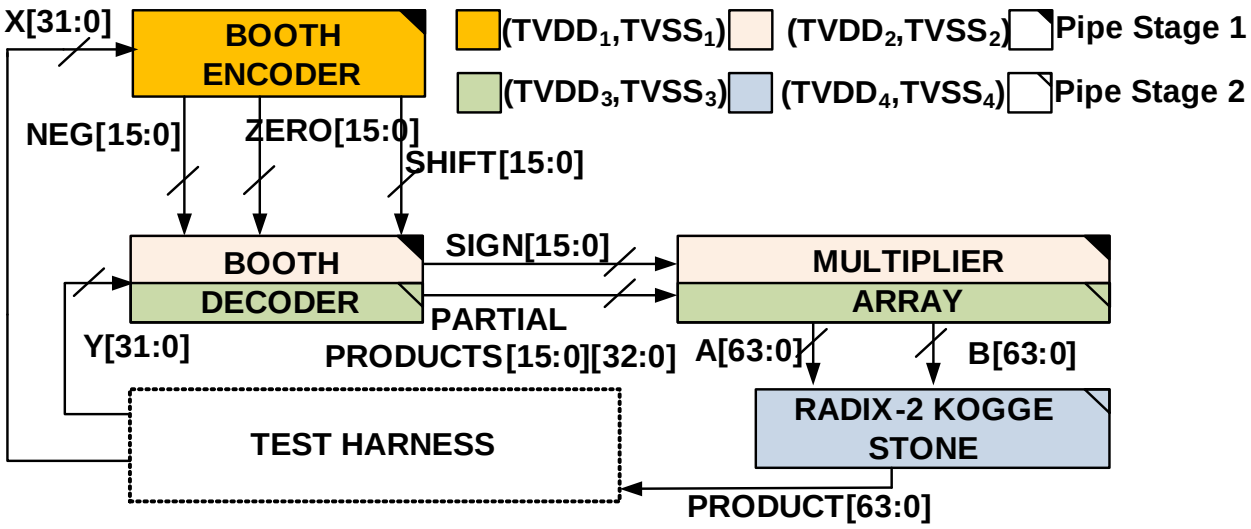


Figure 2.15: **Test prototype.** ART Domino was implemented on a $32b \times 32b$ multiplier in 65nm CMOS. The topology was an array multiplier with radix-2 Kogge Stone adder for final summation. The multiplier was partitioned into four tunable voltage domains and two pipeline stages as shown.

2.4 Measured Results

In this section, we discuss the measurement results for the ART Domino multiplier prototype. With ART disabled, the multiplier runs at 890MHz at 1.2V and 27°C, and consumes 184 mW. Measured frequency contours as a function of the tunable voltages are shown in Figure 2.16. This plot shows that performance with ART is improved to 1.192GHz (34% benefit) by eliminating robustness margins at nominal PVT conditions. Figure 2.17 plots measured minimum ART power overheads with achieved performance. The overhead initially reduces due to reduced voltage swing and increases at higher frequencies. Since every multiplication operation occurs twice (speculative and safe evaluation), energy overhead of ART accounting for both cycles is higher at 110% at 1.192GHz. Figure 2.18 shows measured voltage tuning to achieve these power-frequency points.

Measured error rate due to robustness failures (Figure 2.19) indicate higher sensitivity to TVSS tuning. The measured error rate due to timing failures is shown in Figure 2.20. Temperature dependence of gains due to robustness speculation (Figure 2.21) show higher gains for lower temperatures, as expected. Measured performance gain due to robustness speculation across 20 dies is shown in Figure 2.22. The average gain due to robustness speculation across the dies is ∼28%. The overall performance gains are shown in Figure 2.23. Measurement gains due to timing speculation (Fig. 6) at nominal temperature (27°C) and voltage (1.2V) range from 20% to 33% compared to performance of the slowest die at 85°C with 10% supply droop. Tuning robustness margins provides further gains (24% to 34%) resulting in measured total gains of 49% to 71% over conventionally margined designs. The die micrograph is shown in Figure 2.24 and the implementation is summarized in Table 2.1.

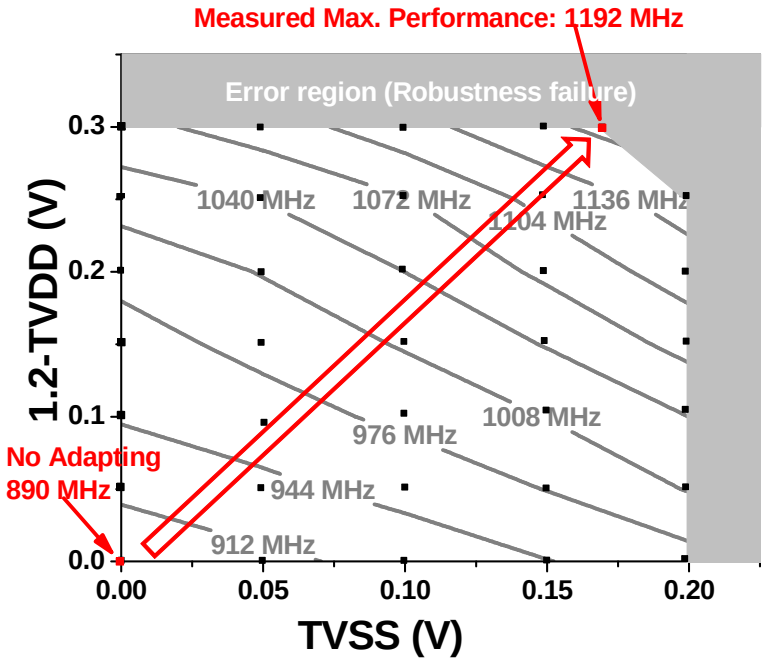


Figure 2.16: **Measured frequency contours as a function of the tunable voltages.** ART Domino improves performance by 34% by eliminating robustness margins at nominal PVT conditions.

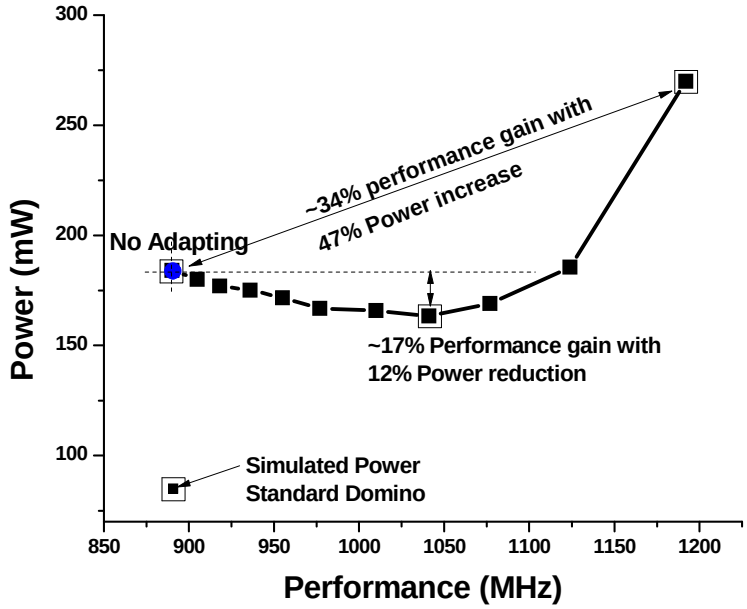


Figure 2.17: **Measured ART power as a function of performance.** The overhead initially reduces due to reduced voltage swing and increases at higher frequencies.

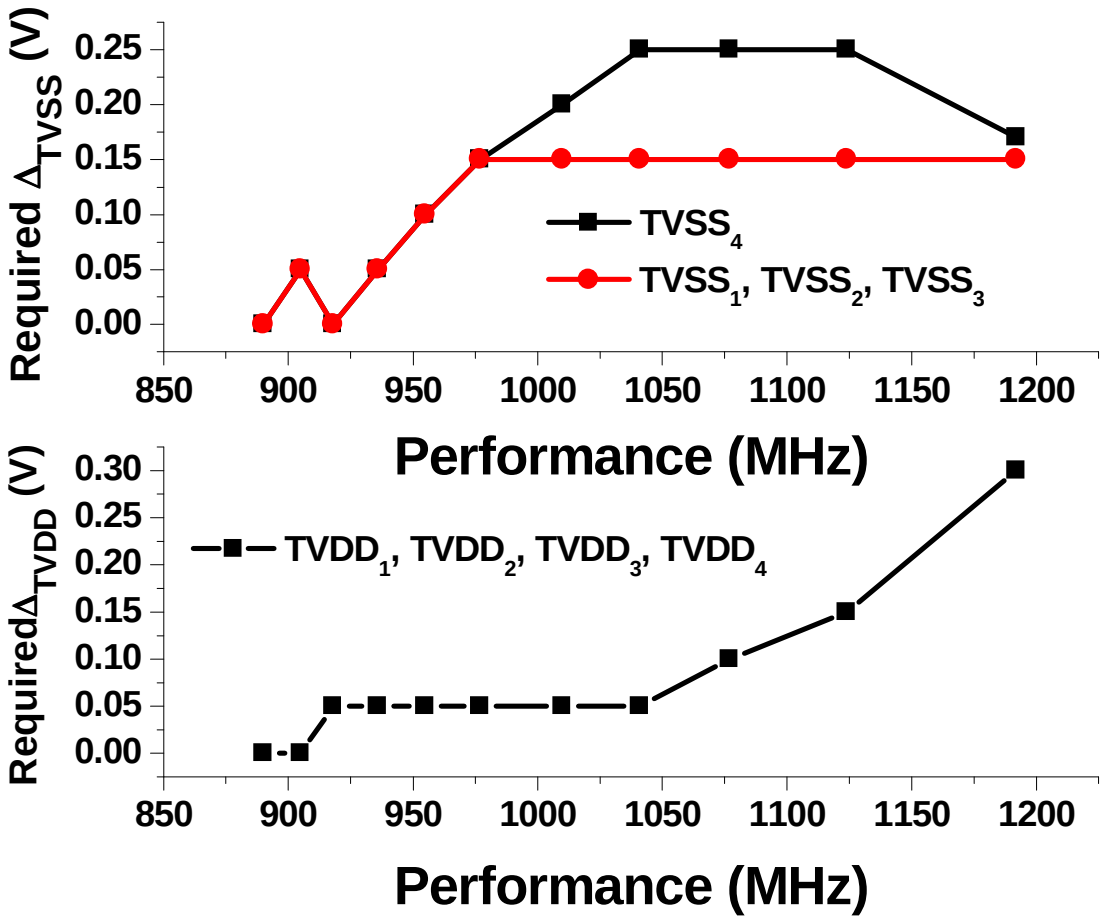


Figure 2.18: **Measured tunable voltage profiles as a function of achieved performance.** The step size for each voltage domain is 50mV.

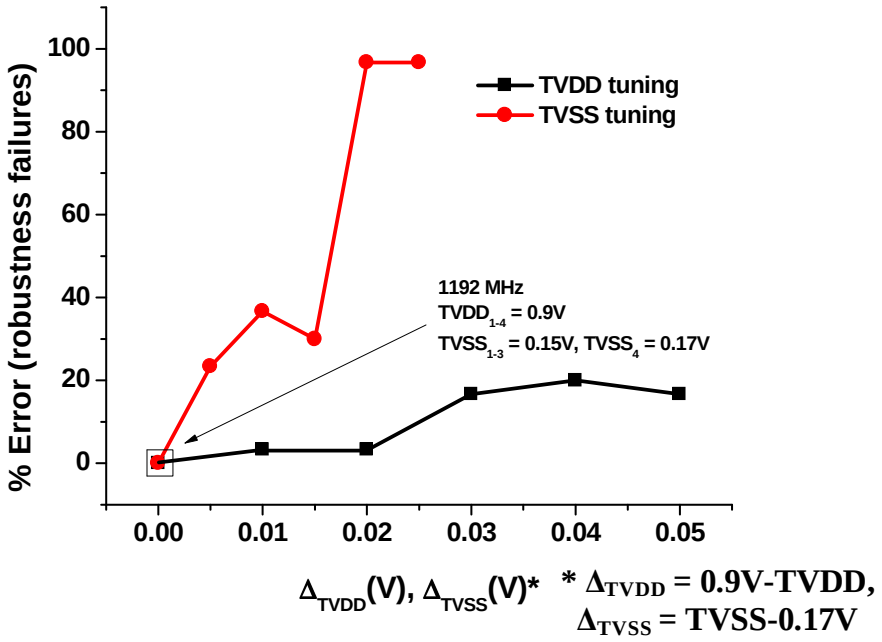


Figure 2.19: **Measured errors rates due to robustness failures.** Error rate shows a higher sensitivity to TVSS tuning.

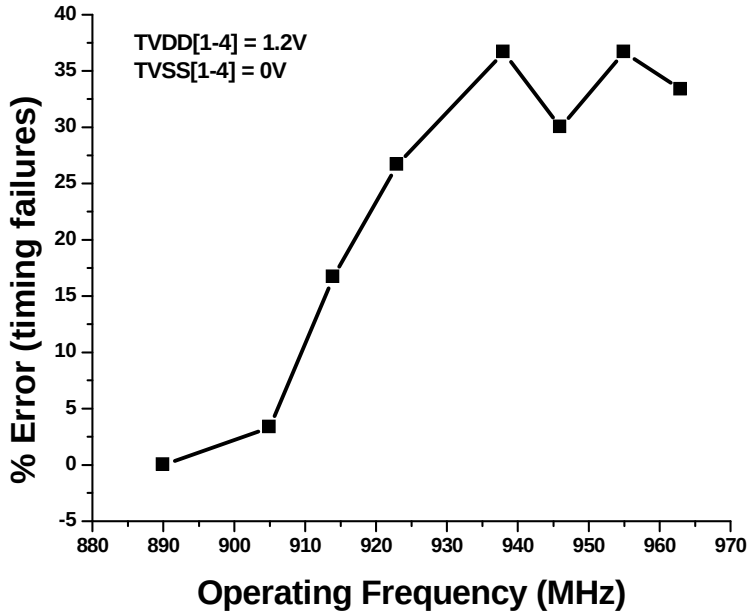


Figure 2.20: **Measured errors rates due to timing failures.** The non-monotonic data points are attributed to measurement setup limitations.

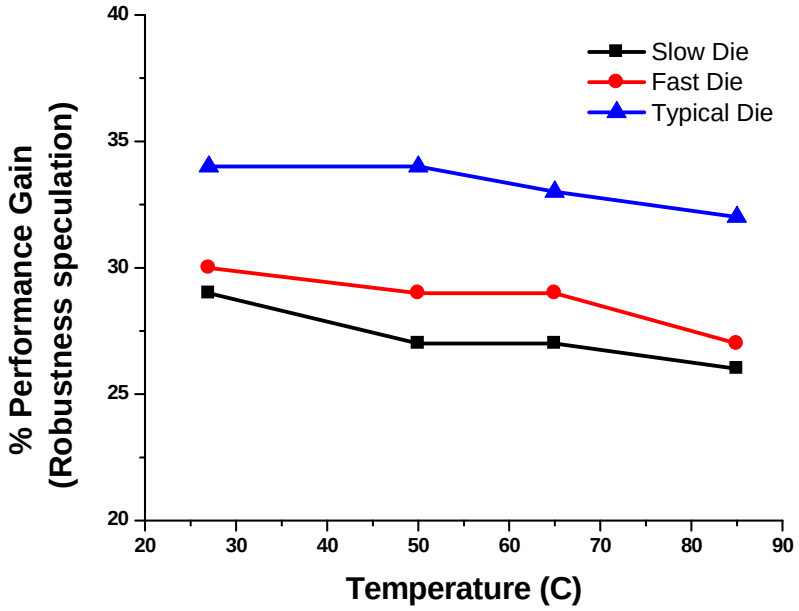


Figure 2.21: **Measured performance gain due to robustness speculation as a function of temperature.** The gain decreases at higher temperatures as gates become less robust.

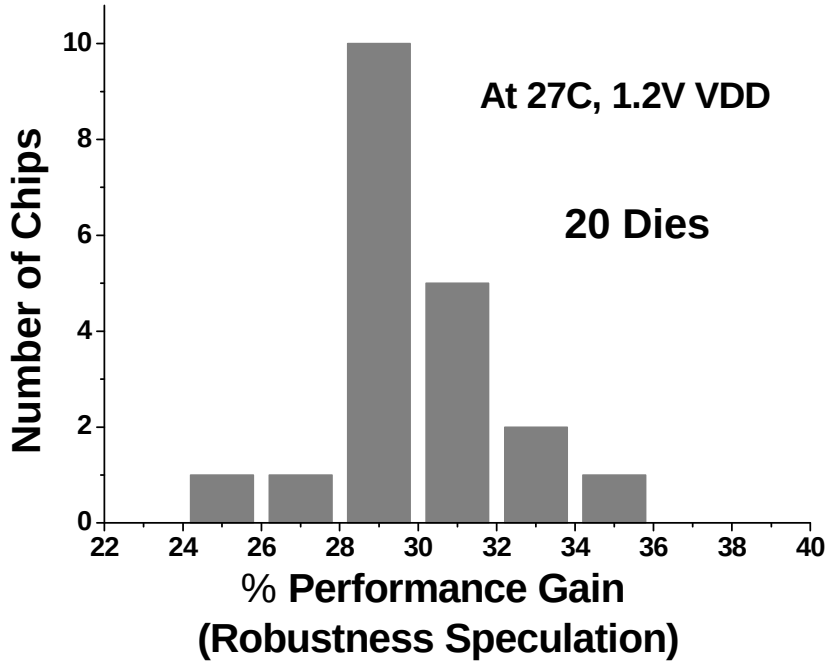


Figure 2.22: **Measured performance gain due to robustness speculation across dies.** The average gain due to robustness speculation across the dies is ∼28%.

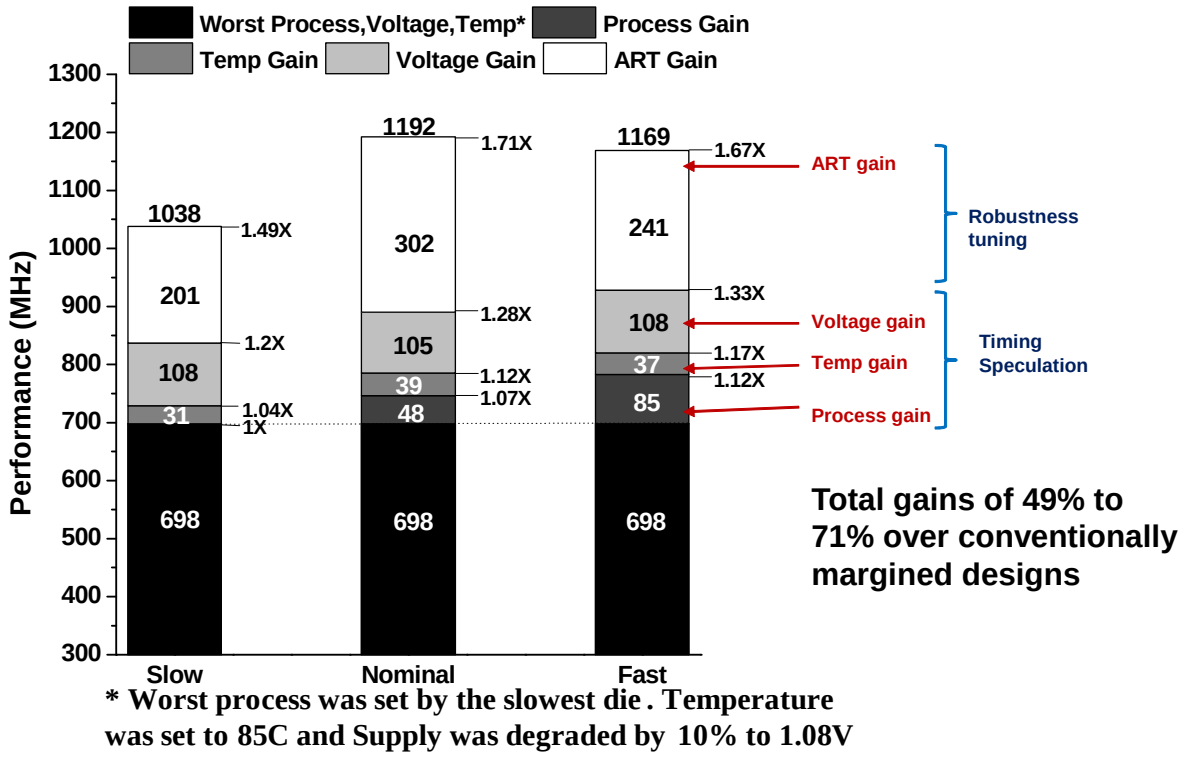


Figure 2.23: **Measured performance improvement due to robustness and timing speculation.** The performance was measured across 20 dies at 85°C with 10(1.2V) across the dies range from ∼20% to 33% compared to the slowest die. Tuning robustness margins provides further gains of ∼24% to 34% resulting in measured total gains of 49% to 71% over conventionally margined designs.

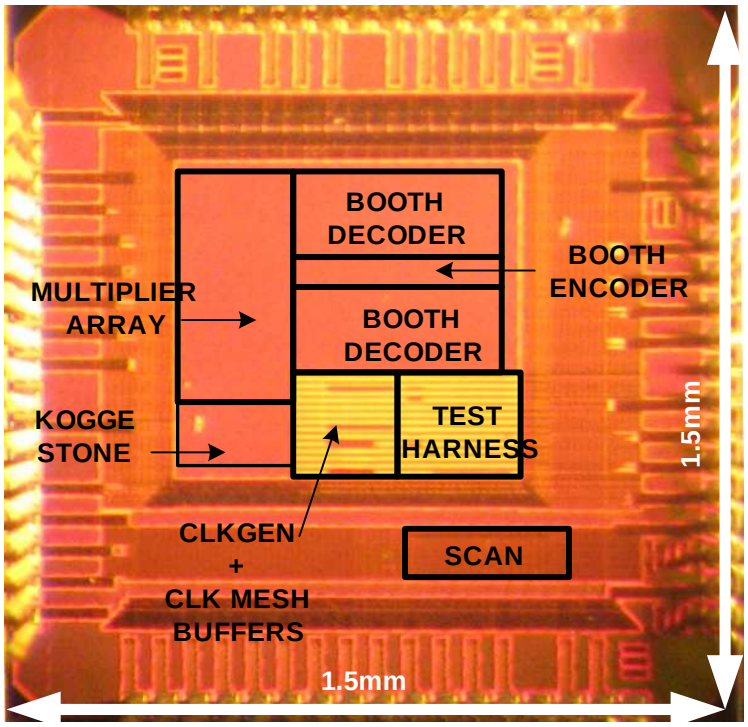


Figure 2.24: **Die micrograph in 65nm CMOS.** ART Domino was implemented on a $32b \times 32b$ multiplier.

Table 2.1: **ART design and performance summary.**

Technology	CMOS 65nm
Dimensions	$1496um \times 1496um$
Nominal Frequency	890MHz
Operating Voltage	1.2V
Max. Performance Gains (Robustness Speculation)	34%
Max. Performance Gains (Timing Speculation)	33%
Max. Performance Gains (Robustness + Timing Speculation)	71%

CHAPTER 3

Pulse-Amplification Based Dynamic Synchronizers with Metastability Measurement using Capacitance De-rating

In this chapter, a new class of dynamic buffer based synchronizers are presented where pulses, rather than stable intermediate voltages, cause metastability. We exploit this unique feature by amplifying such pulses to improve MTBF by $\sim 1 \times 10^6 \times$ *(*$\sim 5 \times 10^7 \times$*) over jamb latches (double flip-flops) at 2GHz in 65nm CMOS. The synchronizers provide single cycle synchronization with a MTBF of* $\sim 2 \times 10^{11}$ *years. A new technique to experimentally measure metastability in silicon is also presented and used to measure results.*

3.1 Motivation

Message-passing and shared-memory based multicore processors have risen in popularity and often employ independent DVFS of the cores to improve energy efficiency [24, 25]. As a result, fast and reliable on-chip communication is a key challenge in these systems and has spurred extensive research to reduce the occurrence of metastability during synchronization. Metastability is an issue because different downstream gates in a path can interpret the metastable value differently, which in turn can to lead to a system-wide functional failure. Figure 3.1 shows this scenario. Previous hardware approaches [29, 30, 43] have addressed this by either increasing synchronization latency or constraining the relationship between

the clock frequencies. This chapter presents dynamic buffer based synchronizers [44] where pulses, rather than stable intermediate voltages, generate metastability events due to the one-sided operation of dynamic gates. This unique feature enables the key advantage, that mean time between failures (MTBF) can be significantly improved by amplifying such pulses using skewed inverters in the synchronization path. In a 65nm test chip (FO4 delay = 11ps), this approach improves MTBF by $\sim 1 \times 10^6 \times$ over jamb latches and $\sim 5 \times 10^7 \times$ over double flip-flops (2-FFs) at 2GHz. The synchronizers provide single cycle synchronization with an MTBF of up to $\sim 2 \times 10^{11}$ years. In addition, a new silicon-confirmed metastability measurement method using capacitance de-rating is also demonstrated and used to evaluate synchronizer performance.

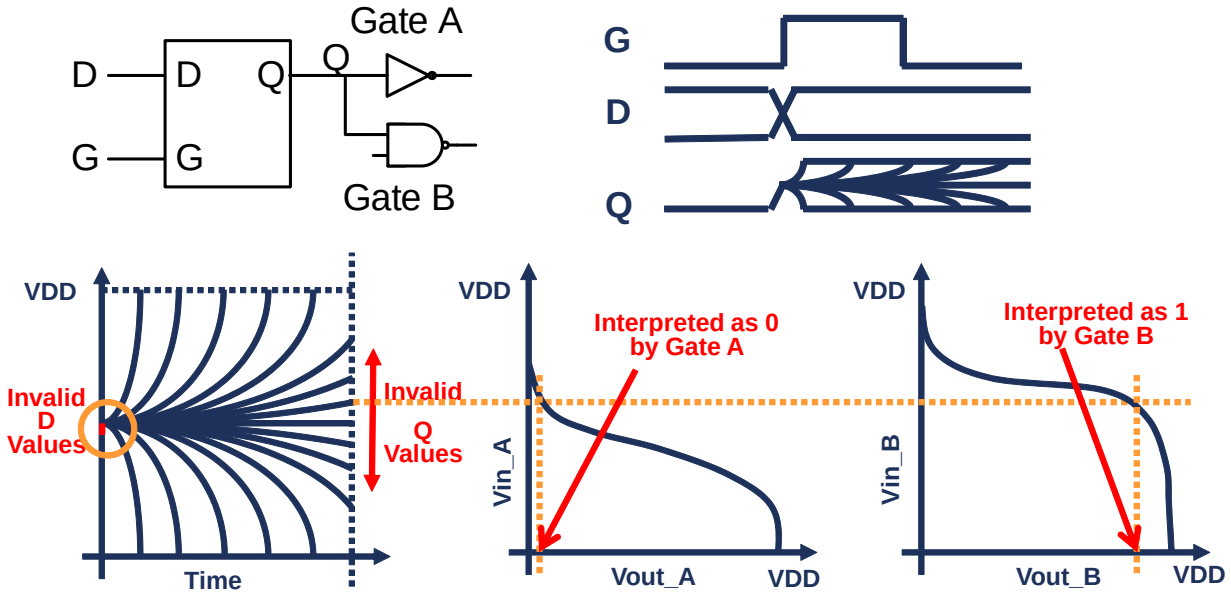


Figure 3.1: **The issue of metastability.** Different downstream gates can interpret the metastable value differently, which can lead to a system-wide functional failure.

3.2 Synchronizer Design

Typically, a synchronizer uses two series-connected flip-flops, as shown in Figure 3.2. The 1^{st} FF has a finite probability of sampling the input during a transition and becoming metastable due to the asynchronous relationship of the two clock domains. This event can cause an arbitrarily slow transition at Q1, which in turn can cause the 2^{nd} FF to become metastable during the subsequent cycle. As mentioned earlier, this output can be interpreted inconsistently by different downstream gates, potentially causing a functional failure. By increasing resolution time, the metastability probability reduces exponentially, giving rise to a fundamental latency/robustness trade-off [27].

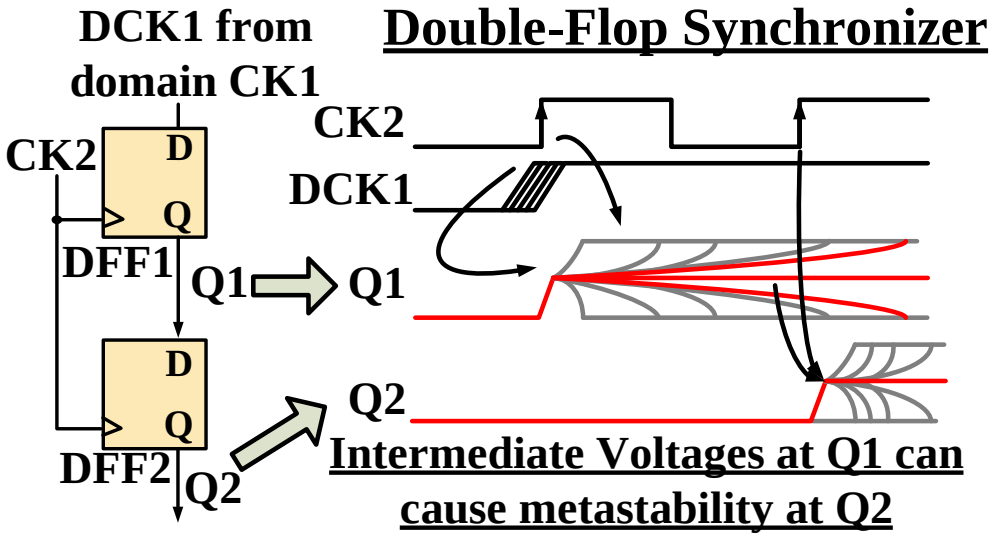


Figure 3.2: **Metastability in double-flop synchronizers.** Stable or slowly resolving intermediate voltages at Q1 can cause DFF2 to go metastable.

Dynamic buffers, however, exhibit one-sided evaluation. Figure 3.3 shows that a similar intermediate signal voltage at Y1 causes the final buffer G2 to fully evaluate (scenario 1), avoiding metastability. Instead, metastability occurs when buffer G1 generates a pulse at Y1 (resulting from its keeper) that causes partial evaluation at G2 and metastability at output Y2 (scenario 2). We make the key observation that such pulses, unique to dynamic logic, can be amplified using skewed inverters, thereby reducing metastability without adding synchronization cycle latency.

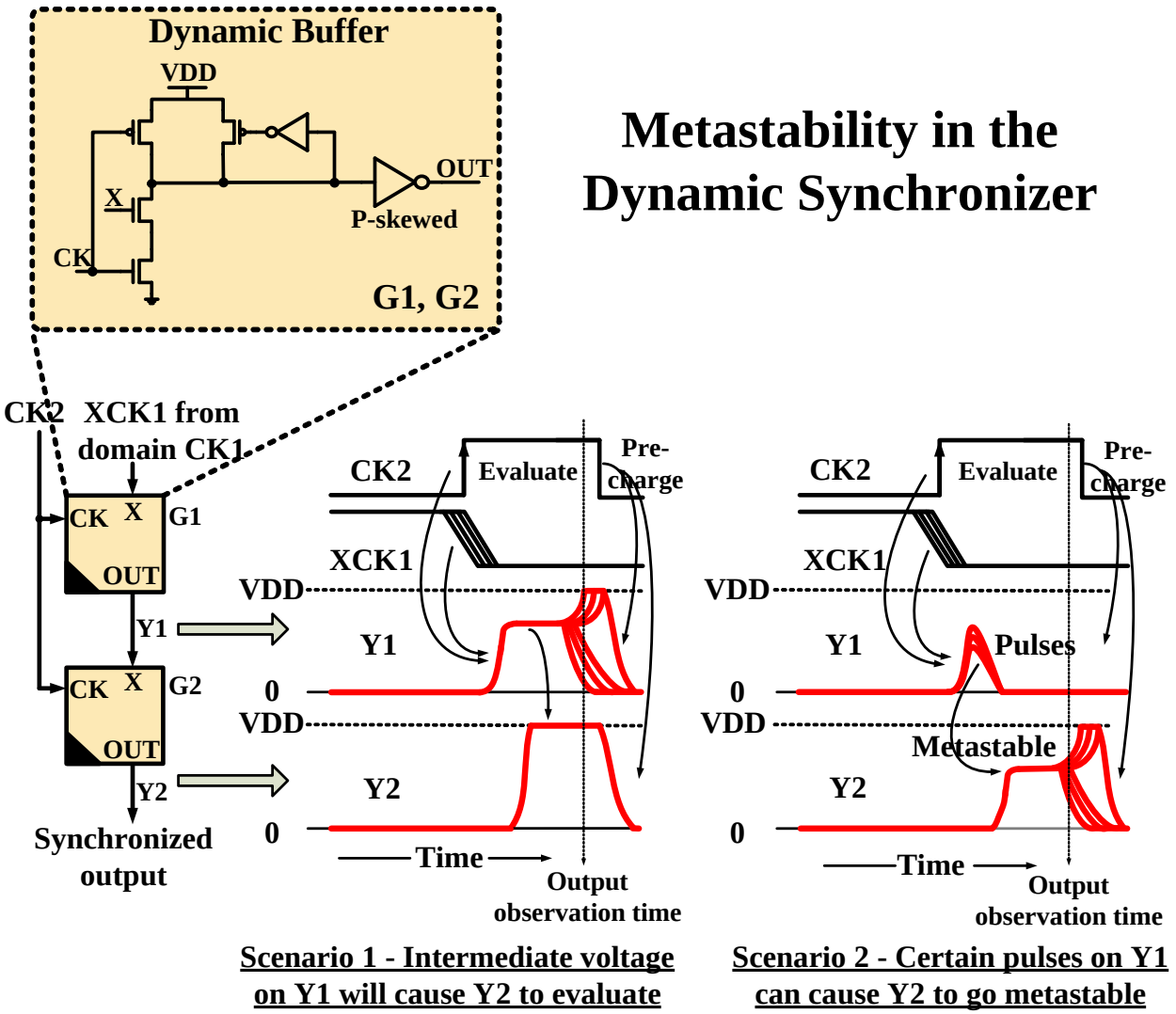


Figure 3.3: **Metastability in dynamic synchronizers.** In contrast to double-flop synchronizers, metastability is only caused by pulses, as shown in scenario 2. Such a pulse occurs due to data-clock alignment at buffer G1 (because of its keeper) that causes partial evaluation at G2 and metastability at Y2. These pulses can be amplified to significantly improve MTBF.

3.2.1 Pulse Amplification

Figure 3.4 shows how pulses generated at Y1 by dynamic buffer G1 are amplified using properly skewed inverters. If the pulse at the input of buffer G2 is within a specific height/width range, its output will fail to evaluate to a rail voltage by the required time, causing a metastability event at Y2. By providing amplification, the range of pulses at the output of G1 causing metastability in G2 is compressed. This in turn compresses the window (range) of data input to clock alignments at the input of G1 that yield metastability (metastability window). The amplifying inverters are skewed by aligning their DC transfer function with the input pulse height to maximize gain, as shown in Figure 3.5. Properly skewing the inverters in a 3-inverter chain by aligning their DC transfer functions with the input pulse height improves stage gain by 2.3×.

Figure 3.6 shows that adding skewed inverters improves MTBF by $\sim 2 \times 10^3 \times$ in a 2-stage synchronizer (based on simulation). In addition to inserting inverters, additional dynamic buffers, also clocked by CK2, can be inserted since they function much like inverters, providing gain for propagated pulses without adding cycle latency.

Figure 3.6 also shows that inserting additional inverters in FF-based synchronizers does not improve metastability. A properly skewed inverter chain with metastable input will drive its output to rail. However, since the metastable input can still resolve in either direction, the inverter output can still switch back at a later time, creating metastability in the capturing FF (in contrast to dynamic synchronizers with one-sided evaluation). Thus, inverter insertion only delays the metastability event and also worsens MTBF by reducing available resolution time (due to the additional delay of the inserted inverters). This is further explained in Figure 3.7.

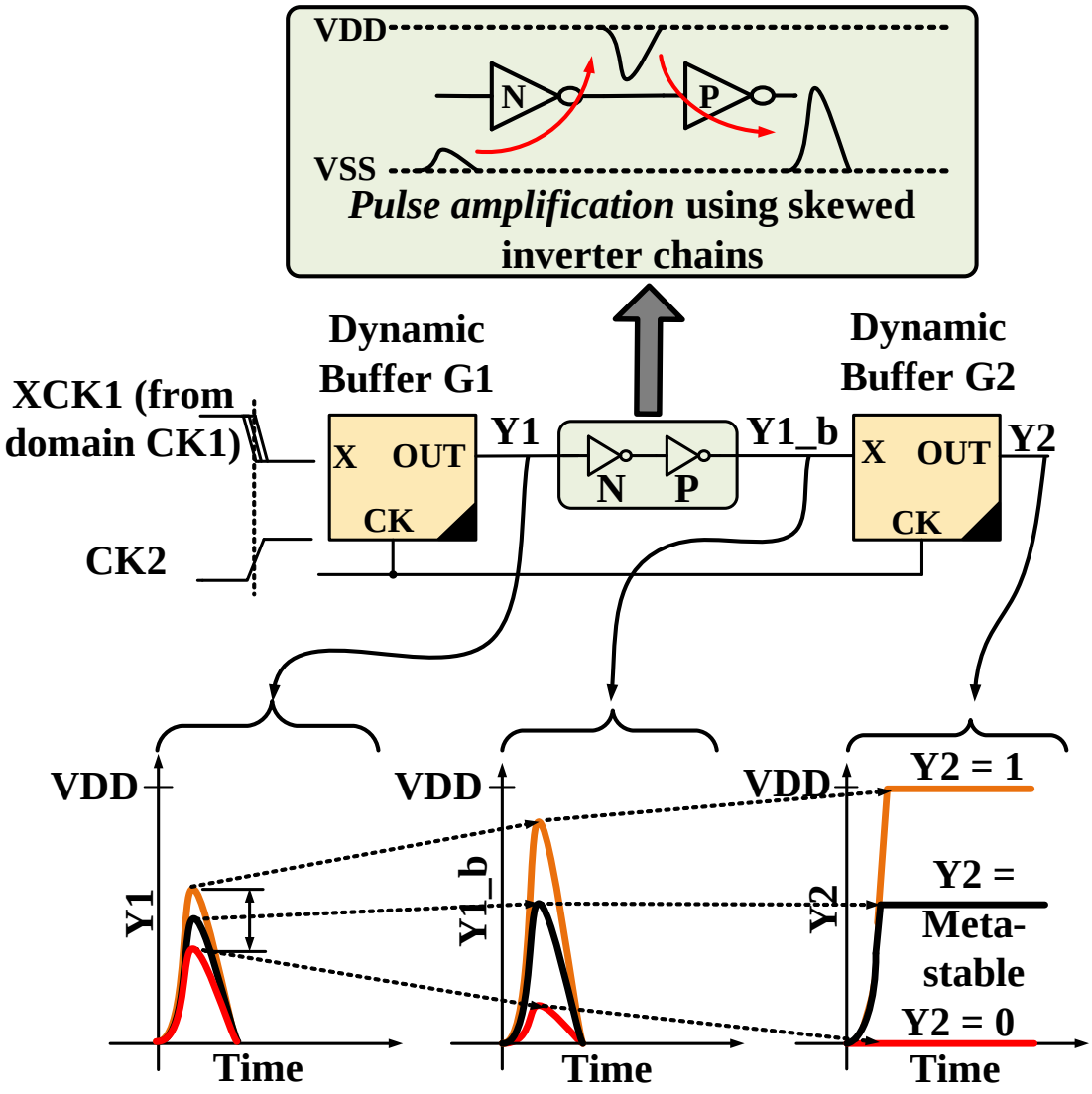


Figure 3.4: **Pulse amplification in Dynamic synchronizers.** Only pulses within a specific width/height range at the input of G2 can cause metastability at Y2. This range is compressed through pulse amplification using skewed inverters and added buffer stages.

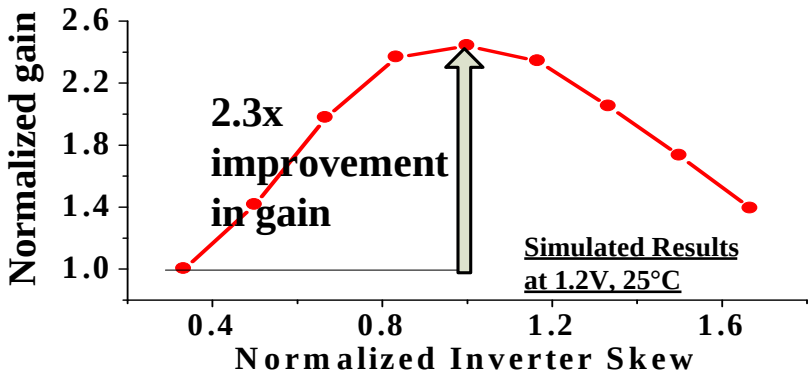


Figure 3.5: **Stage gain sensitivity to inverter skew.** Properly skewing the inverters in a 3-inverter chain by aligning their DC transfer functions with the input pulse height improves stage gain by 2.3×.

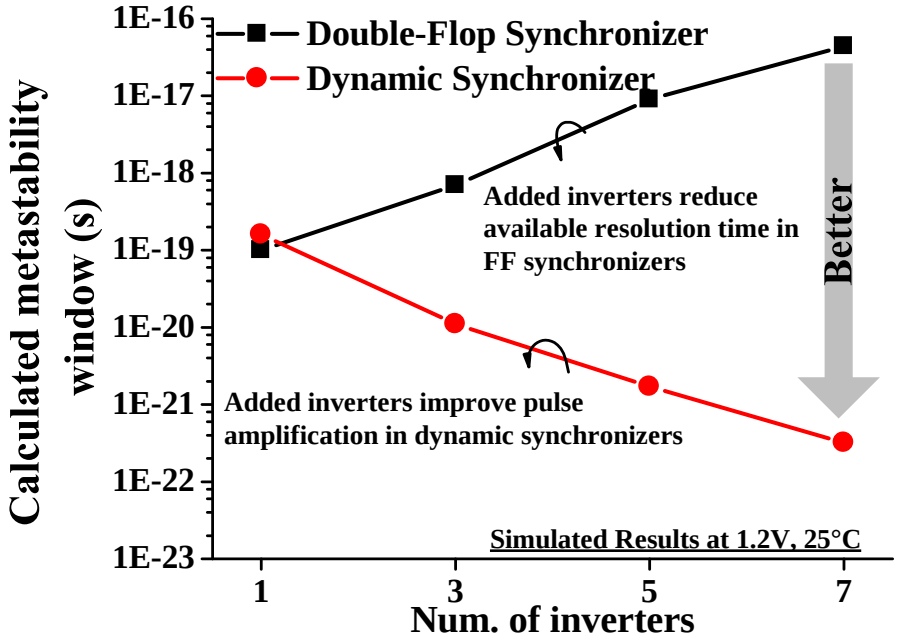


Figure 3.6: **Pulse amplification in dynamic synchronizers contrasted with FF-based synchronizers.** In contrast to FF-based synchronizers, skewed inverters improve MTBF by $\sim 2 \times 10^3 \times$ in a 2-stage dynamic synchronizer.

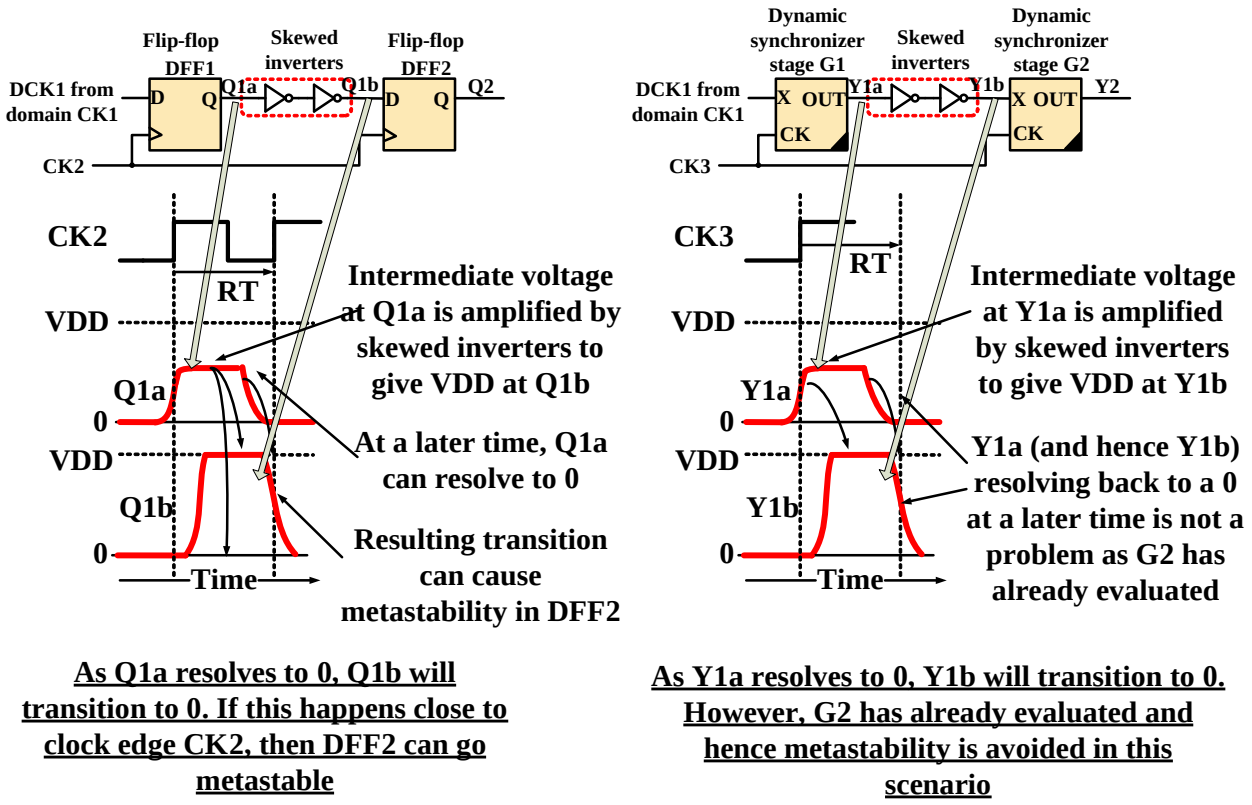


Figure 3.7: **Why pulse amplification uniquely benefits dynamic synchronizers.** The one-sided nature of dynamic gates ensures that late changing signals (possibly resolving from metastability) do not affect the following gate if it has already evaluated.

3.2.2 System-Level Performance Impact

In conventional flop-based synchronizers, designers add more flip-flops in series in order to exponentially reduce the probability of metastability. However, this degrades system performance. Figure 3.8 shows that a 3-cycle synchronization latency can degrade performance by 11% in a NoC. Dynamic synchronizers provide single-cycle synchronization and reduce this overhead to 4%. The simulated NoC configuration is shown in Figure 3.9. The 64 routers were partitioned into four frequency domains. Synchronizers were inserted at the frequency boundaries.

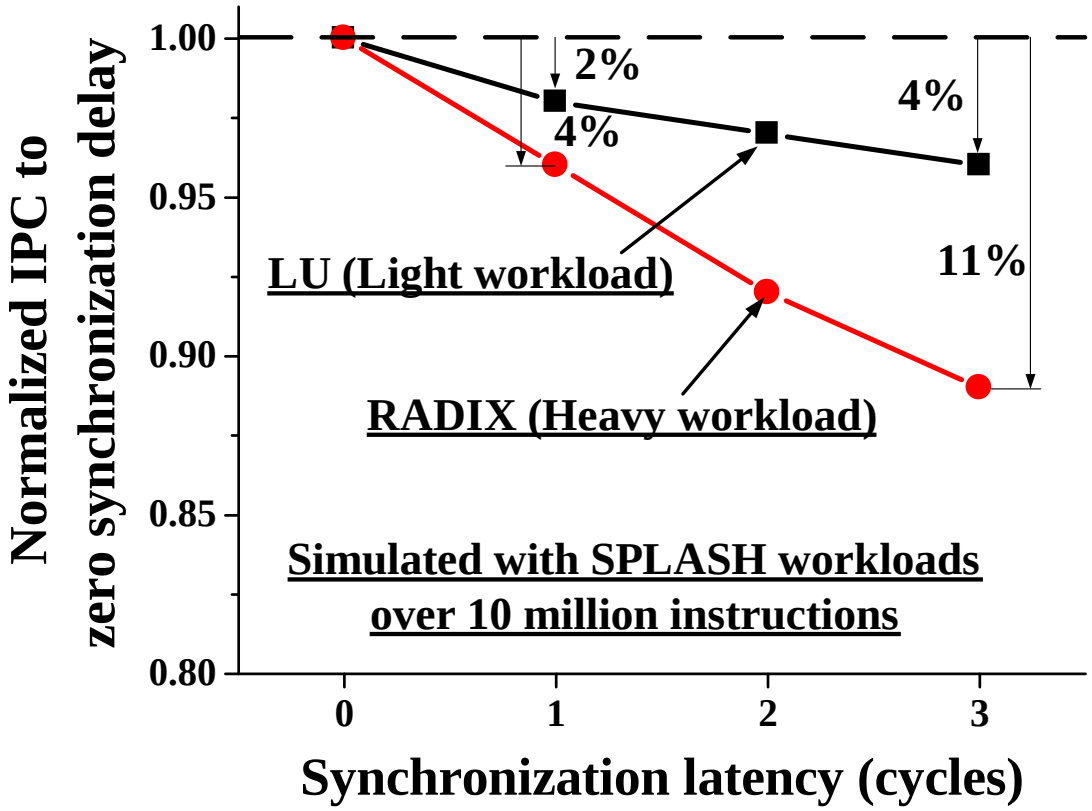
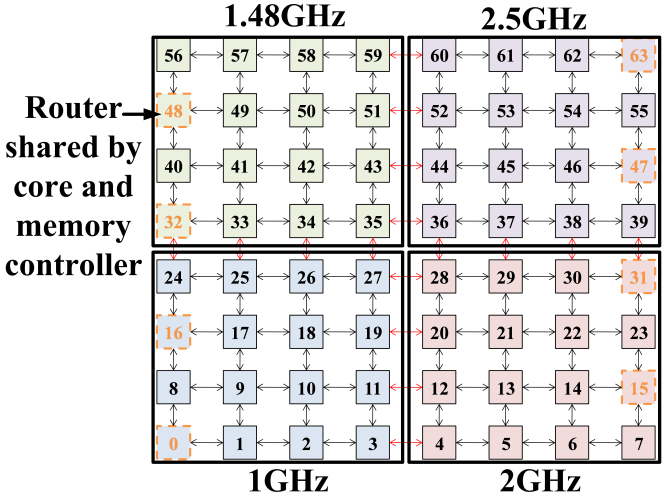


Figure 3.8: **Performance impact of synchronization latency.** 3-cycle synchronization latency can degrade performance by 11% in a NoC. Dynamic synchronizers provide single-cycle synchronization and reduce this overhead to 4%.



Cores	**64 cores, ARM Cortex A15, 2-way out of order at 2GHz**
L1 Cache	**32kB per core, private, 4-way set associative, 64B block size, 2-cycle latency, split I/D caches, 32 MSHRs**
L2 Cache	**56 banks, 256kB per bank, shared 8-way set associative, 64B block size, 6-cycle latency, 32 MSHRs**
Main Memory	**8 on-chip memory controllers with 4 DDR channels, each at 16GB/s, up to 16 outstanding requests per core, 50ns access latency**
Network	**128-bit datapath width, 64 routers Each router: 4flits/virtual channel (VC) 4 VCs per port**

Figure 3.9: **Simulated NoC configuration.** The 64 routers were partitioned into four frequency domains. Synchronizers were inserted at the frequency boundaries.

3.2.3 Dynamic Synchronizer Circuit Details

Figure 3.10 shows a single stage of a dynamic synchronizer. The full keeper improves gain at the dynamic node by 13.3× (simulated). Cutoff device M1 prevents short circuit current during precharge. Gate length in the inverters is increased to 70nm (L_{min} = 60nm), improving gain by 30%.

Like any other dynamic circuit, this synchronizer also has a precharge phase. Hence, in order to maintain throughput, two such synchronizers are operated in a ping-pong manner as shown in Figure 3.11. DS1 and DS2 are both complete, multi-stage dynamic synchronizers. When the synchronizer DS1 precharges, DS2 enters evaluation and hides this precharge latency.

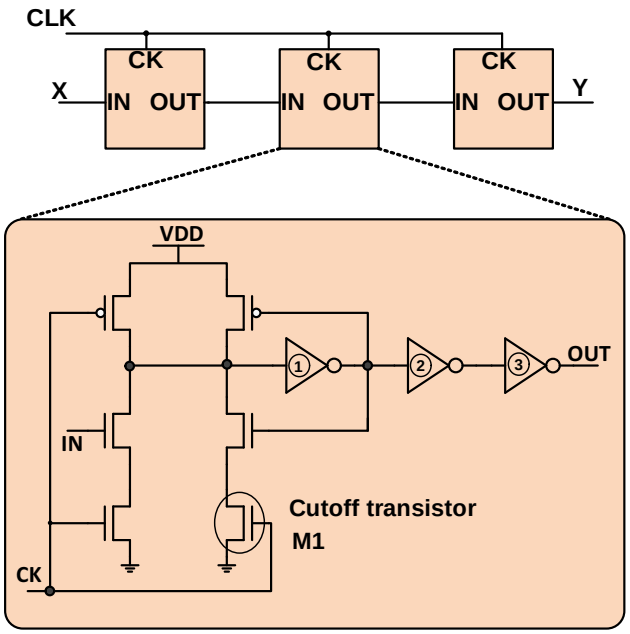


Figure 3.10: **3-stage, 3-inverter dynamic synchronizer.** Circuit details of a single stage are shown. Cutoff device M1 prevents short circuit current during precharge.

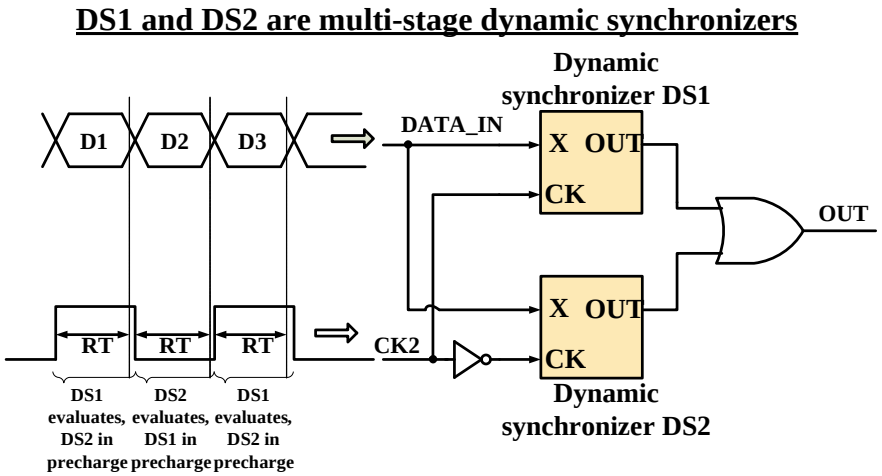


Figure 3.11: **Ping-pong operation of dynamic synchronizers.** The two synchronizers operate in a ping-pong fashion in order to hide each others precharge latency.

3.3 Metastability Measurement and Simulation Techniques

In this section, we discuss the CAD methodology to simulate these synchronizers in Section 3.3.1. We also explain the capacitance de-rating in Section 3.3.2 which is used to characterize these synchronizers in silicon.

3.3.1 Transfer Function Based Simulations

SPICE reliably simulates metastability windows down to the 1×10^{-18}s range. To optimize the synchronizer design, we developed an analysis flow in which each inverter/dynamic stage is characterized using a dynamic transfer function to map a range of input pulses to output pulses. Figure 3.12 shows this methodology applied to a 2-stage dynamic synchronizer. A single stage is characterized in SPICE to generate to sets of mappings: 1) Input data-clock alignment to output (Y1) pulse amplitude, and 2) Output (Y2) amplitude at resolution time (RT) to input pulse (Y1) amplitude. By interpolating these characterization tables, it is possible to accurately predict the metastability window size for windows as small as 1×10^{-40}s.

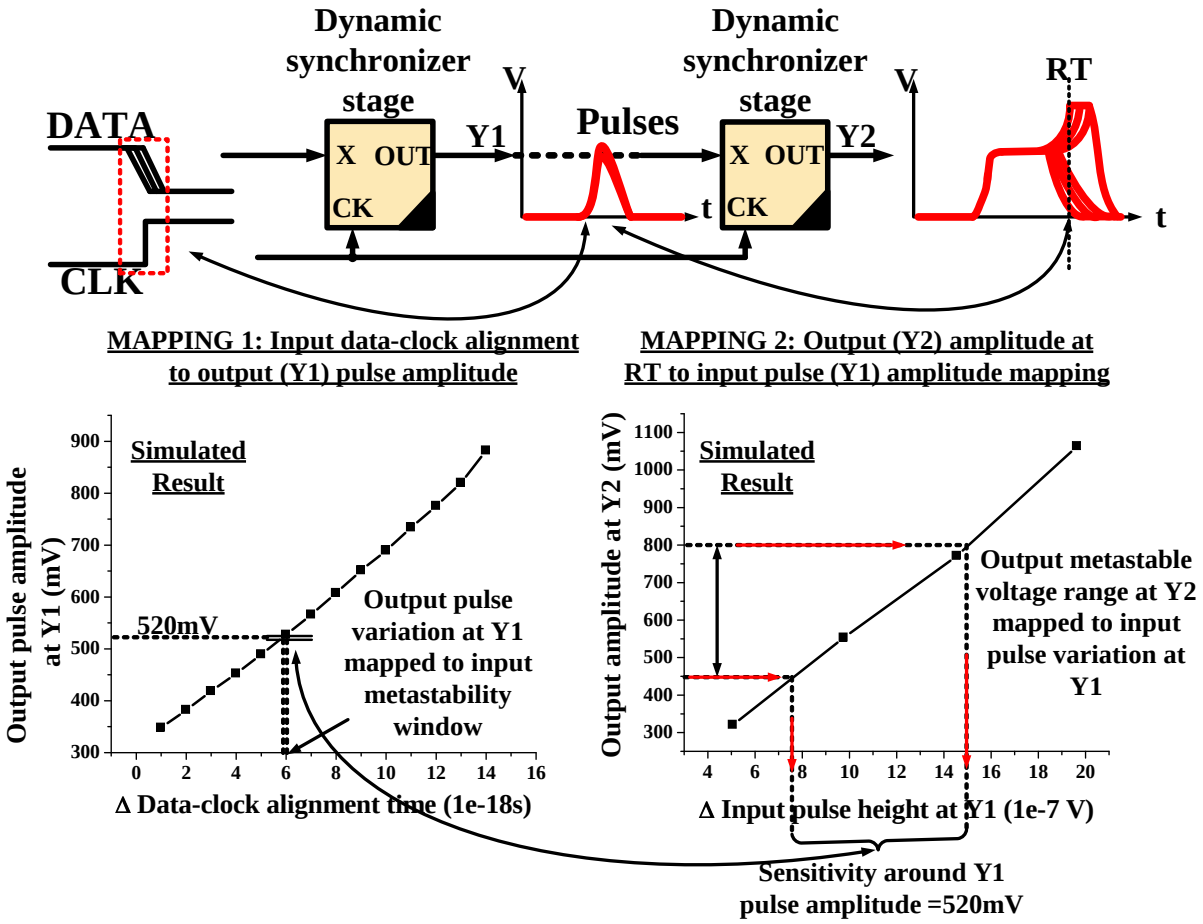


Figure 3.12: **Transfer function based simulation methodology.** The mappings have been generated by characterizing a single synchronizer stage in SPICE.

3.3.2 Capacitance De-rating

To experimentally characterize MTBF, we present a new measurement method where DUTS are de-rated (slowed) by connecting their internal nodes to selectable MIM capacitors (Figure 3.13). By increasing node capacitance, gain-bandwidth product is reduced and the metastability window increases. Such windows are then measured and results are finally extrapolated to the actual metastability window under native (self-loaded) conditions. Slowing the gates also requires resolution time (RT) to be de-rated (increased) due to the slower transitions of nodes to their steady-state values. The de-rating of RT and capacitance must be coordinated to obtain a linear dependence, which is critical to facilitate accurate extrapolation.

Figure 3.14 shows that linearly scaling RT with capacitance does not yield a linear change

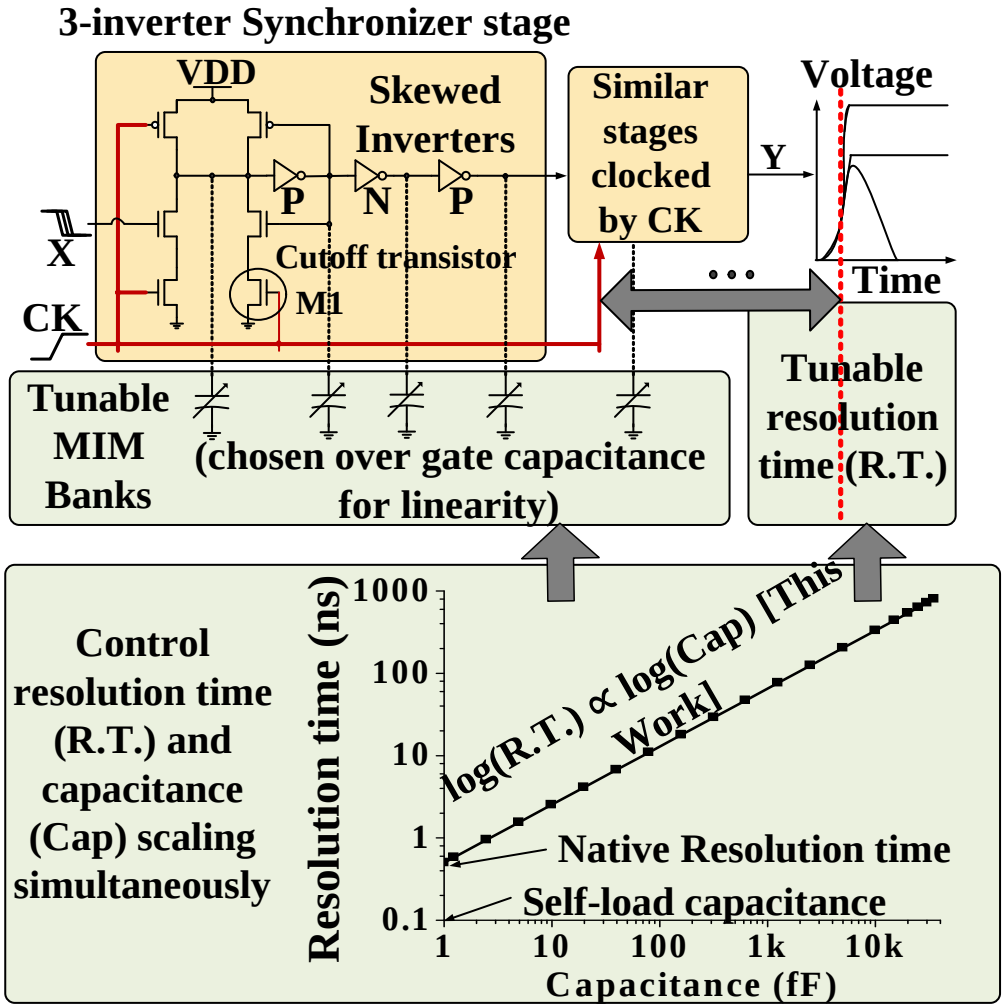


Figure 3.13: **Measurement technique for determining intrinsic metastability window using capacitance de-rating (I).** The de-rating of RT and capacitance must be coordinated to obtain a linear dependence, which is critical to facilitate accurate extrapolation.

in the metastability window. Instead, we find that log-log proportionality between RT and capacitance provides linearity and enables accurate extrapolation to native RT (500ps) under self-loading conditions (1fF for the simulation in Figure 3.14). This result is confirmed by SPICE simulations, transfer function-based calculations, and silicon measurements.

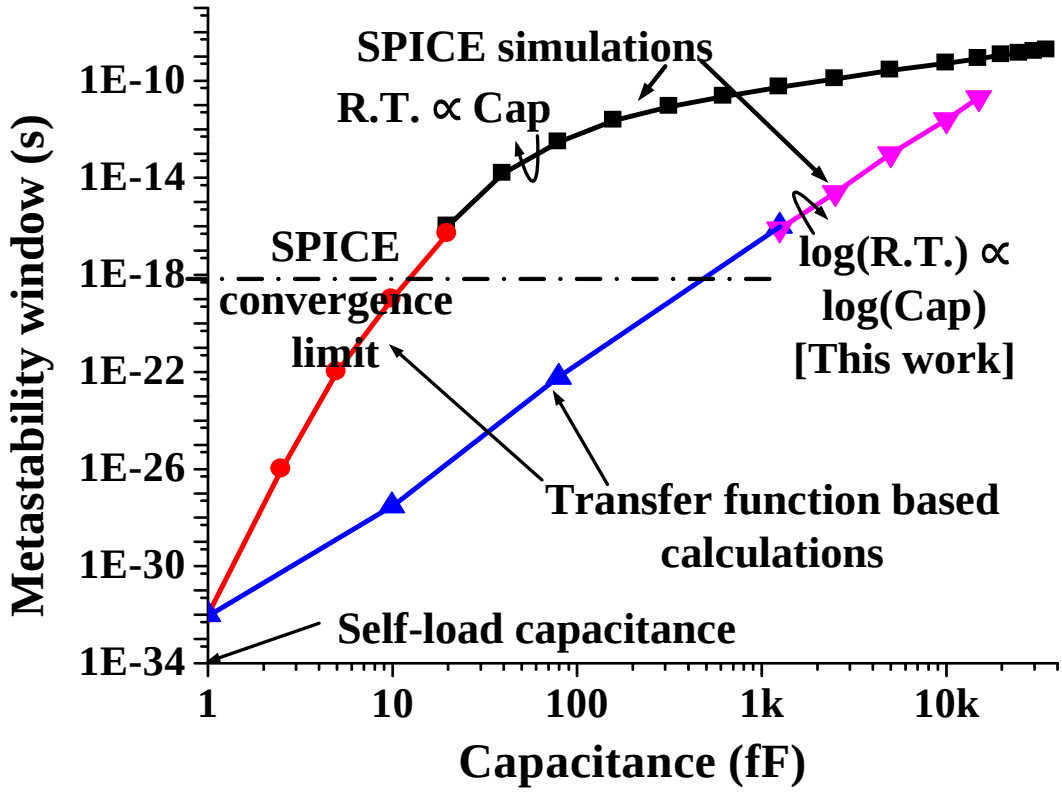


Figure 3.14: **Measurement technique for determining intrinsic metastability window using capacitance de-rating (II).** Scaling capacitance and RT with log-log proportionality results in linear dependence of metastability window on capacitance, enabling accurate extrapolation.

3.3.3 Test Harness

The test harness to measure metastability in silicon using this de-rating scheme is shown in Figure 3.15. The data-clock alignment was controlled using a 3-stage delay chain: 1) a counter-based coarse delay chain with measured steps of 0.5ns; 2) a fine delay chain with measured steps of 18ps; and 3) a Vernier delay chain with measured mean resolution of 1.2ps. A statistical TDC [45] averaging 10^6 results was used to measure the data-clock alignment with 1ps accuracy. The DUTs (at 1.2V VDD) were de-rated by connecting their nodes to calibrated, binary-weighted, selectable MIM capacitors. All switches were double-stacked to remove leakage effects. Two comparators were used to flag a metastable event by comparing the selected DUT output to off-chip references (0.8V and 0.4V) that define the metastable

voltage range. Averaging counters recorded the number of 1, 0, and metastable events over several trials. Data was increasingly delayed with respect to clock and the metastability window was defined as the time range when metastability count dominated over 0- and 1-counts.

In order to accurately measure the data-clock alignment, the fine and Vernier delay chains were initially characterized using the TDC. Figure 3.16 shows the measured variations in step size for the fine delay chains. Measured delays for the fine and Vernier delay chains as a function of TDC output code are shown in Figures 3.17 and 3.18 respectively.

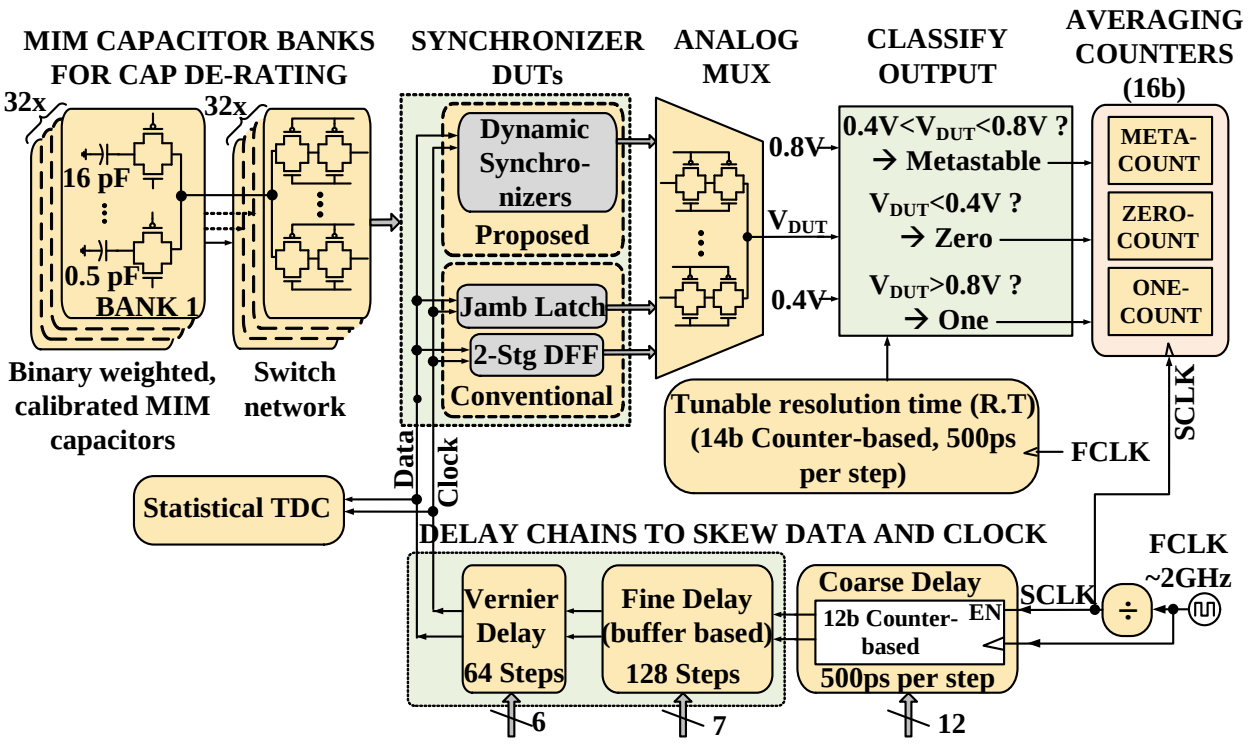


Figure 3.15: **Test harness to measure metastability using capacitance de-rating.** The data-clock alignment is controlled using a 3-stage delay chain and measured using a statistical TDC. The DUT output is compared to off-chip references (0.8V and 0.4V) that define the metastable voltage range. All switches were double-stacked to remove leakage effects.

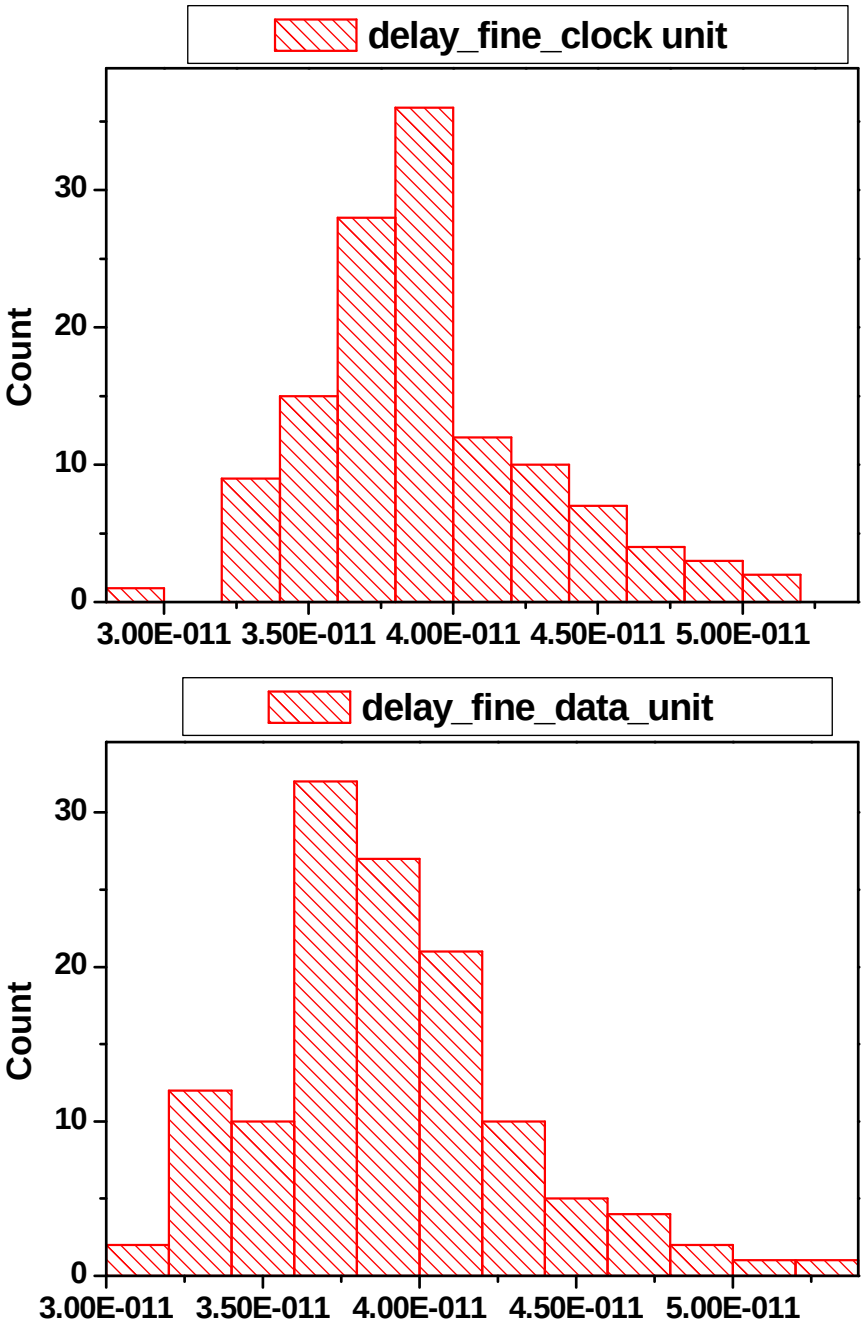


Figure 3.16: **Measured variations in step size (in ps) for the fine delay chains.**

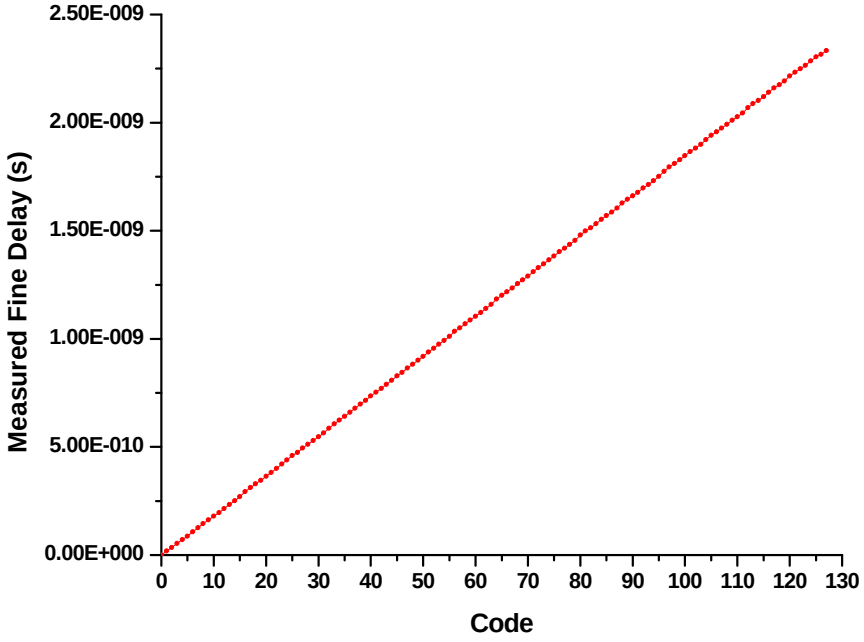


Figure 3.17: **Measured fine delay vs. TDC output code.**

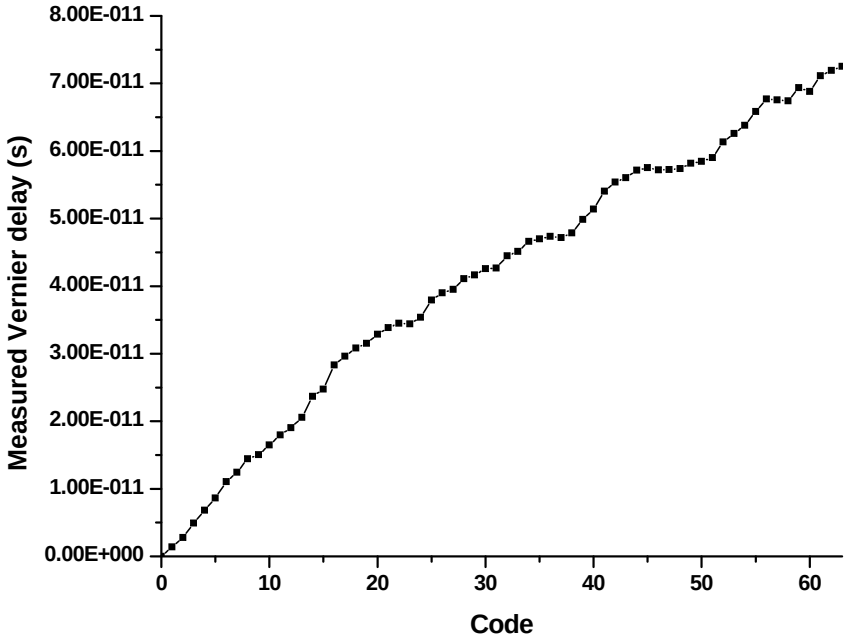


Figure 3.18: **Measured Vernier delay vs. TDC output code.**

3.4 Measured Results

Fourteen dynamic synchronizer configurations differing in the number of stages and inverters per stage were tested at 2GHz and compared to 2-FF, 3-FF and jamb latch synchronizers.

Figure 3.19 shows measured metastability windows for several dynamic synchronizer configurations (de-rated conditions) along with their extrapolated windows at native conditions ($\sim$2fF, 500ps RT). This is the first work where these windows are measured using capacitance de-rating with linear extrapolation. Figure 3.20 corroborates the extrapolation approach by measuring windows using distinct RT / capacitance scaling ratios; results converge to a relatively small range at native conditions, as desired.

Extrapolated windows for all measured configurations are shown in Figure 3.21, confirming that metastability reduces as inverters/dynamic buffers are inserted until their propagation delay becomes prohibitive. The 3-stage, 7-inverter synchronizer provides the best performance and MTBF improvement of 8$\times$ over the jamb latch (Figure 3.22) at the smallest measure-able de-rating condition (9.1pF loading, identical RT of 307ns). This translates to an improvement of $\sim 1 \times 10^6\times$ at native conditions. Figure 3.24 shows that dynamic synchronizers show temperature dependence similar to jamb latches and 2-FF synchronizers.

Figure 3.23 confirms that inserting additional FFs does not improve metastability in FF-based synchronizers, unless RT between the end-point FFs is also increased.

Measurement-based extrapolated windows are also compared with their respective theoretical estimates calculated by measuring τ and t_w [46] from simulation (Figure 3.25). Figure 3.26 shows that the extrapolation error due to measurement and fit limitations is relatively small compared to improvement over jamb latch. This synchronizer has a MTBF of $\sim 2 \times 10^{11}$ years at 2GHz, mapping to a system failure rate of $\sim 8.7 \times 10^{-4}$/year for a CMP with 10^3 synchronized signals at 2.5GHz (jamb latch rate = 55.8/year). Figure 3.27 shows the die micrograph and Table 3.1 shows the implementation summary.

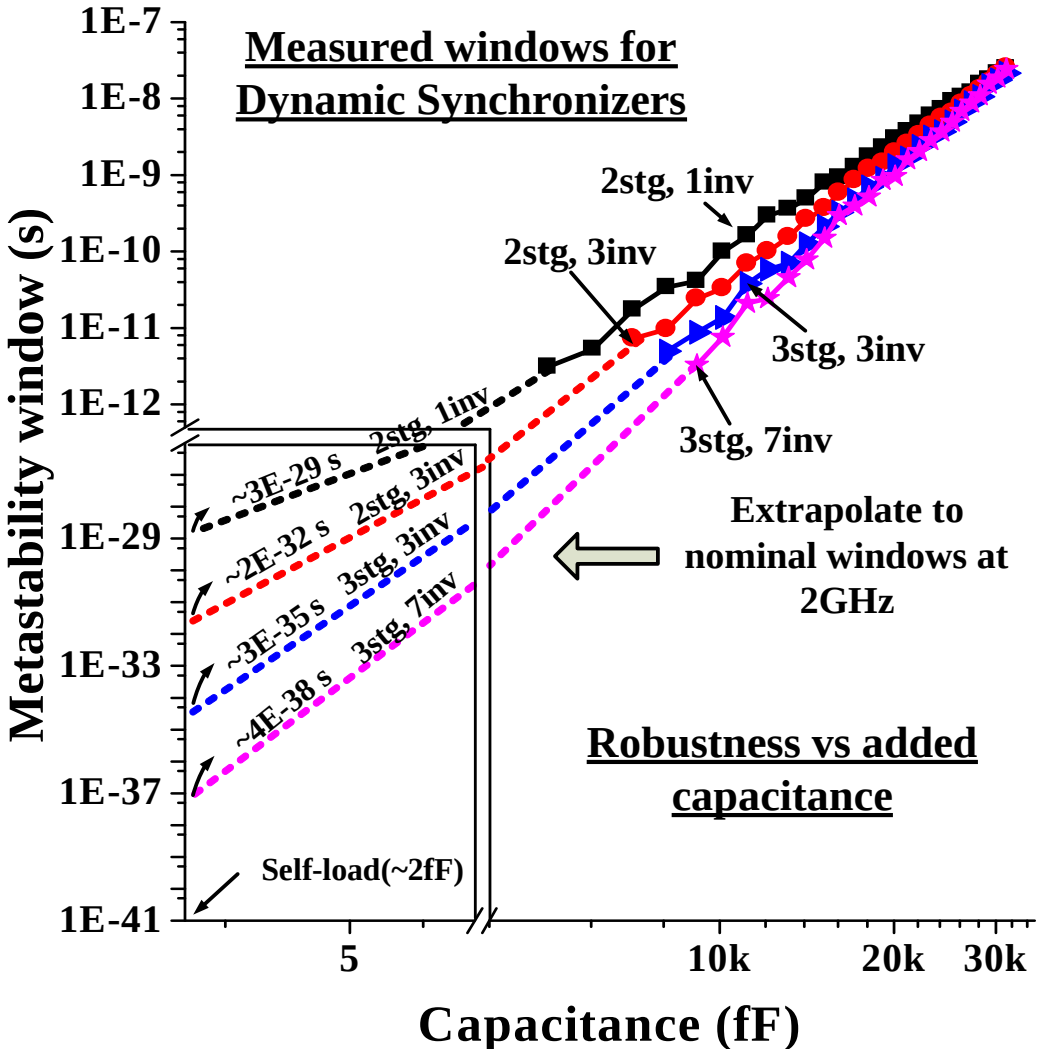


Figure 3.19: **Measured metastability windows for several dynamic synchronizer configurations (de-rated conditions) along with their extrapolated windows at native conditions (∼2fF, 500ps RT)**

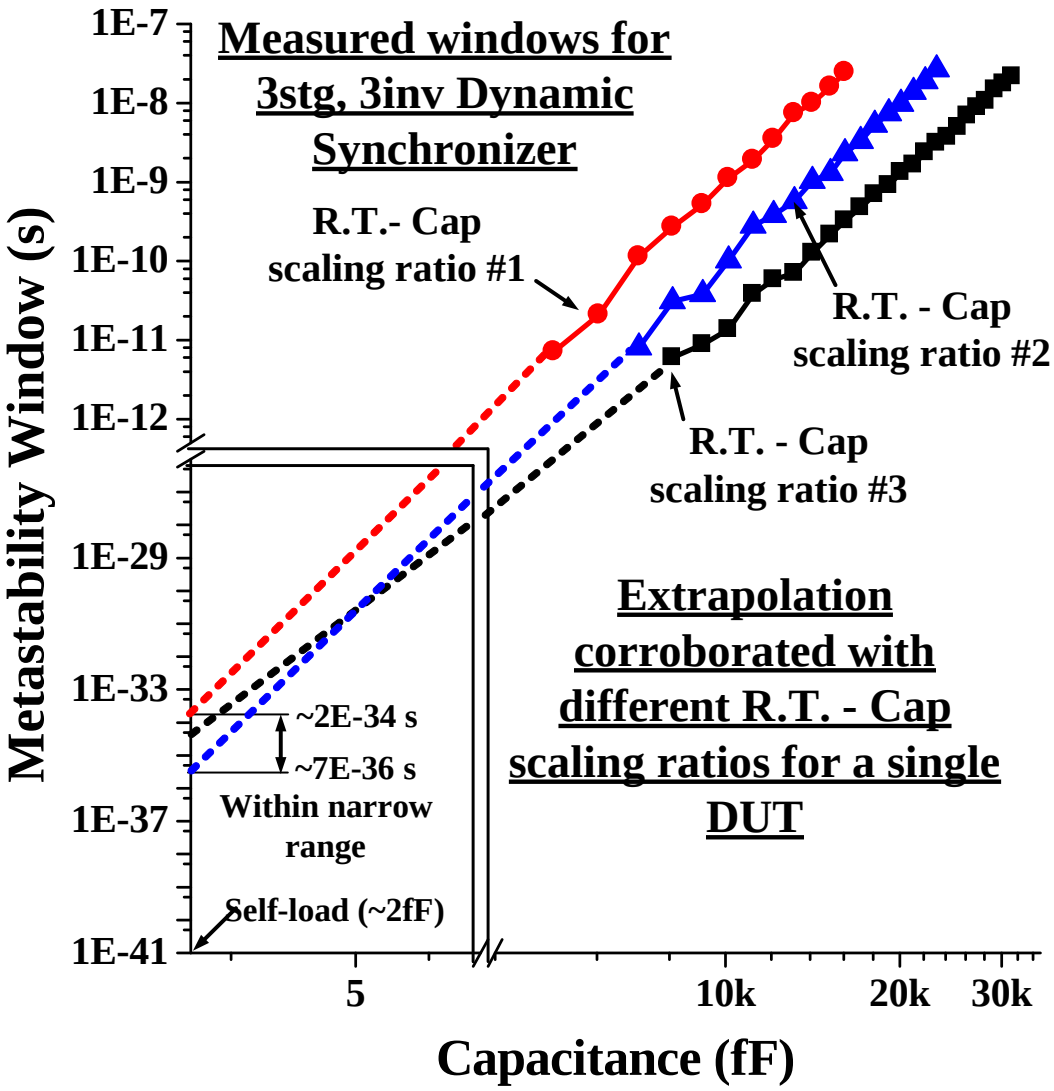


Figure 3.20: **The extrapolation approach by measuring windows using distinct RT / capacitance scaling ratios.** Results converge to a relatively small range at native conditions, as desired.

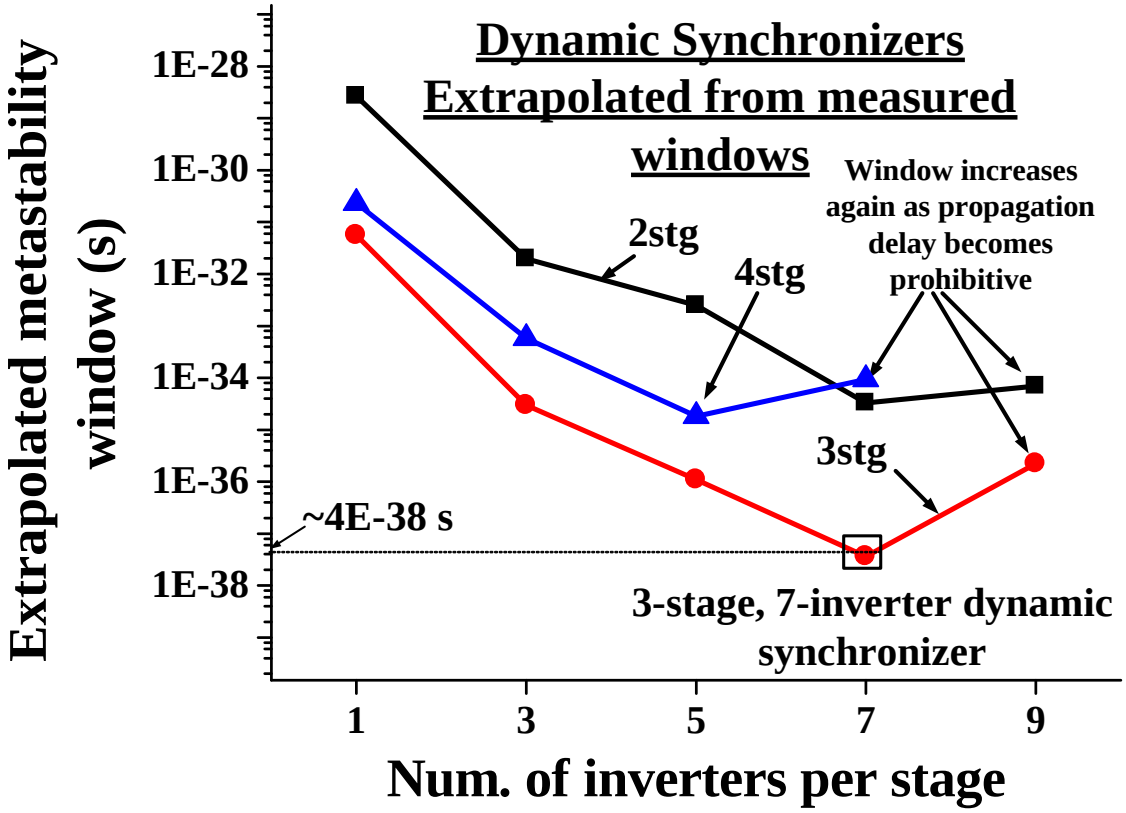


Figure 3.21: **Extrapolated windows for all measured dynamic synchronizer configurations.** Metastability reduces as inverters/dynamic buffers are inserted until their propagation delay becomes prohibitive

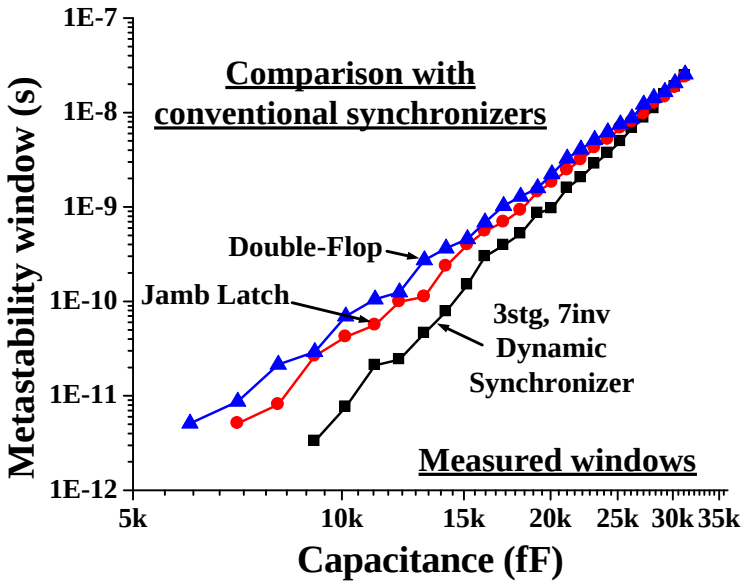


Figure 3.22: **The 3-stage, 7-inverter synchronizer provides the best performance and MTBF improvement of 8× over the jamb latch at the smallest measure-able de-rating condition (9.1pF loading, identical RT of 307ns).** This translates to an improvement of $\sim 1 \times 10^6 \times$ at native conditions.

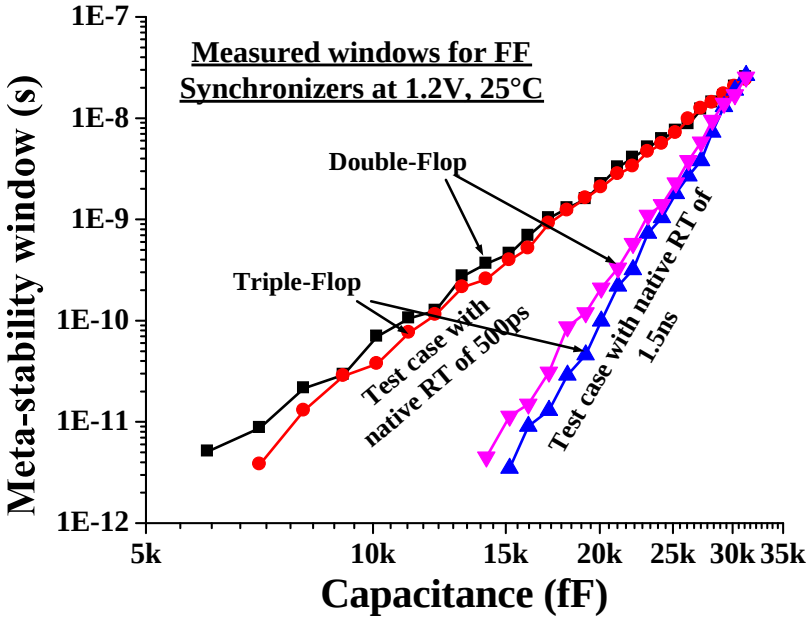


Figure 3.23: **Inserting additional FFs does not improve metastability in FF-based synchronizers, unless RT between the end-point FFs is also increased.**

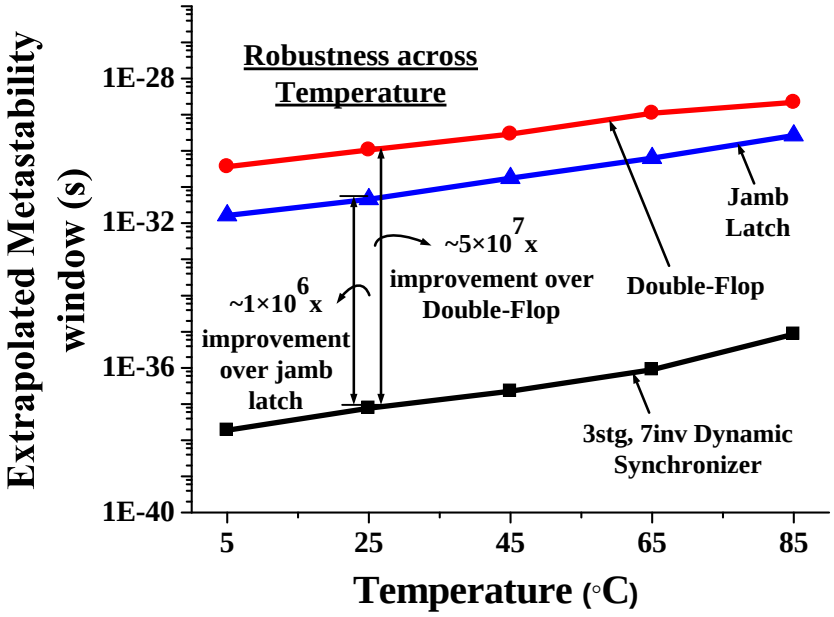


Figure 3.24: **Dynamic synchronizers show temperature dependence similar to jamb latches and 2-FF synchronizers.**

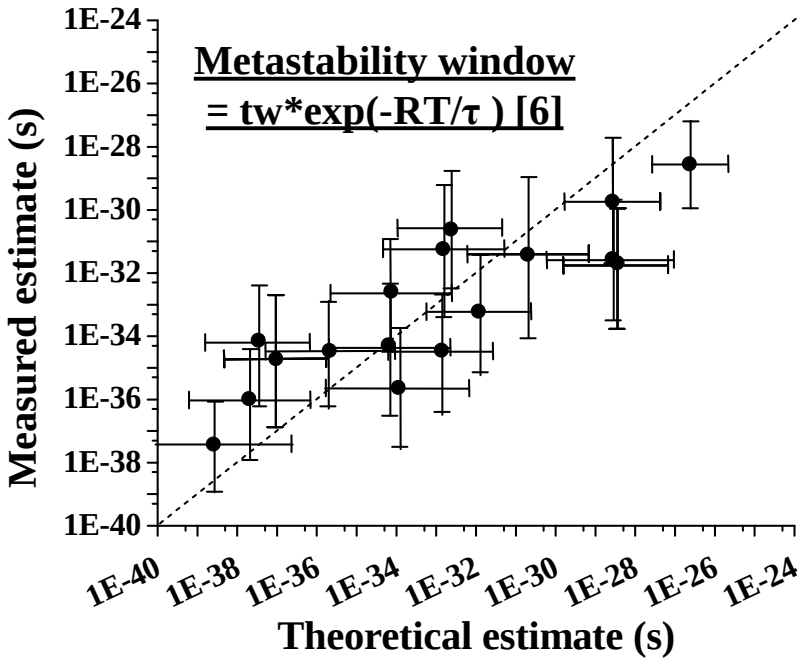


Figure 3.25: **Measurement-based extrapolated windows are also compared with their respective theoretical estimates calculated by measuring τ and t_w [46] from simulation.** Error bars show confidence bounds in both estimates.

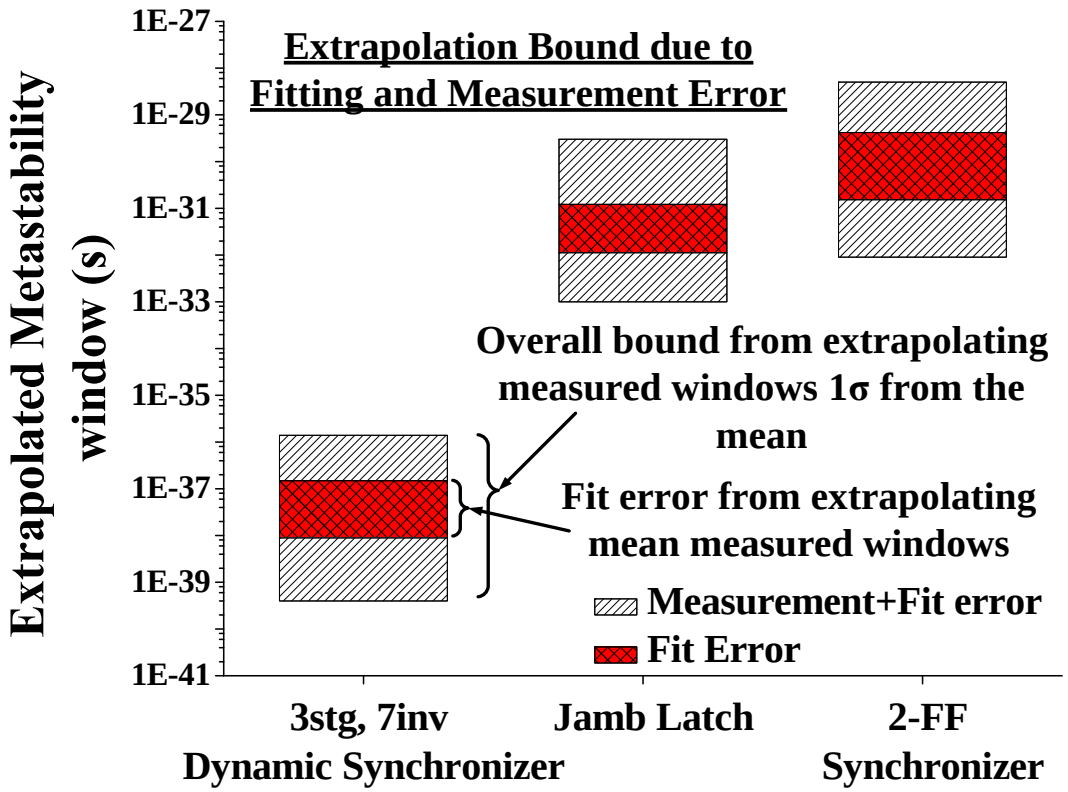


Figure 3.26: **Extrapolation error due to measurement and fit limitations is relatively small compared to improvement over jamb latch.**

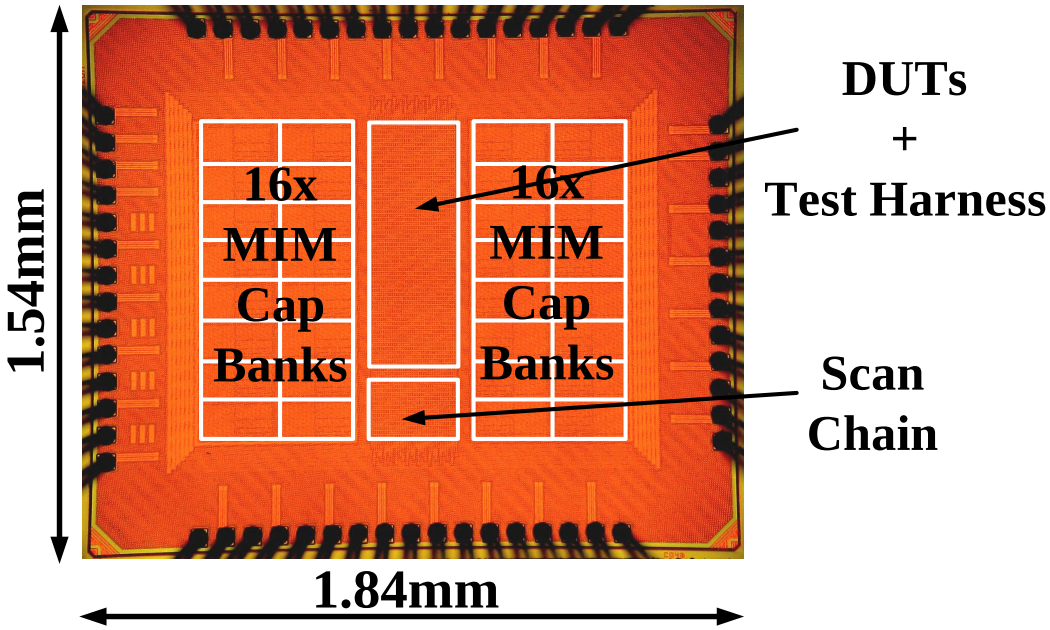


Figure 3.27: **Die micrograph in 65nm CMOS.**

Table 3.1: **Synchronizer design and performance summary.**

Technology	CMOS 65nm		
Dimensions	$1840um \times 1540um$		
Operating Voltage	1.2V		
Operating Frequency	2GHz		
Synchronizer	Normalized MTBF	Power (simulated)	Area
3-stage, 7-inverter dynamic synchronizer (This work)	$\sim 5 \times 10^7 \times$	106uW	$52um^2$
2-FF	$1\times$	60uW	$24um^2$
Jamb latch	$\sim 50\times$	24uW	$8um^2$

CHAPTER 4

VTS: Variation Tolerant Sensing with Auto-Zero Calibration and Pre-amplification for High Performance Memories

In this chapter, a variation tolerant sensing scheme targeting high performance 6T SRAMs is presented. The scheme reconfigures a conventional sensing topology to additionally perform auto-zeroing based offset compensation, and bitline droop pre-amplification. The scheme is implemented in 28nm CMOS, where sensing reliability is improved by 1.2σ_{Vth} without added area overhead. This increased robustness is traded for performance, providing up to 42% sensing speed improvement and 10% lower sensing power at 1.8GHz.

4.1 Motivation

High performance SRAMs are critical elements in microprocessors and SoCs. Fast and robust bitline sensing is a key requirement in such memories. However, process scaling has degraded sensing robustness due to increased mismatch in the sense amplifier (SA) circuit (Figure 4.1). In addition, increased I_{read} variation in the memory bitcell [31] further degrades sensing robustness. The fundamental tradeoff between sensing time and bitline read failures (seen in Figure 4.2) forces designers to heavily margin sensing time in order to guarantee sufficient bitline differential voltage prior to SA triggering. Previous research has proposed to improve SA robustness using pre-amplification circuits [32], capacitance based

offset cancellation [33,34], and redundancy [35]. However, a majority of these schemes target single-ended sensing (losing the benefit of common-mode rejection), incurring up to 60% area overhead or post-silicon tuning costs.

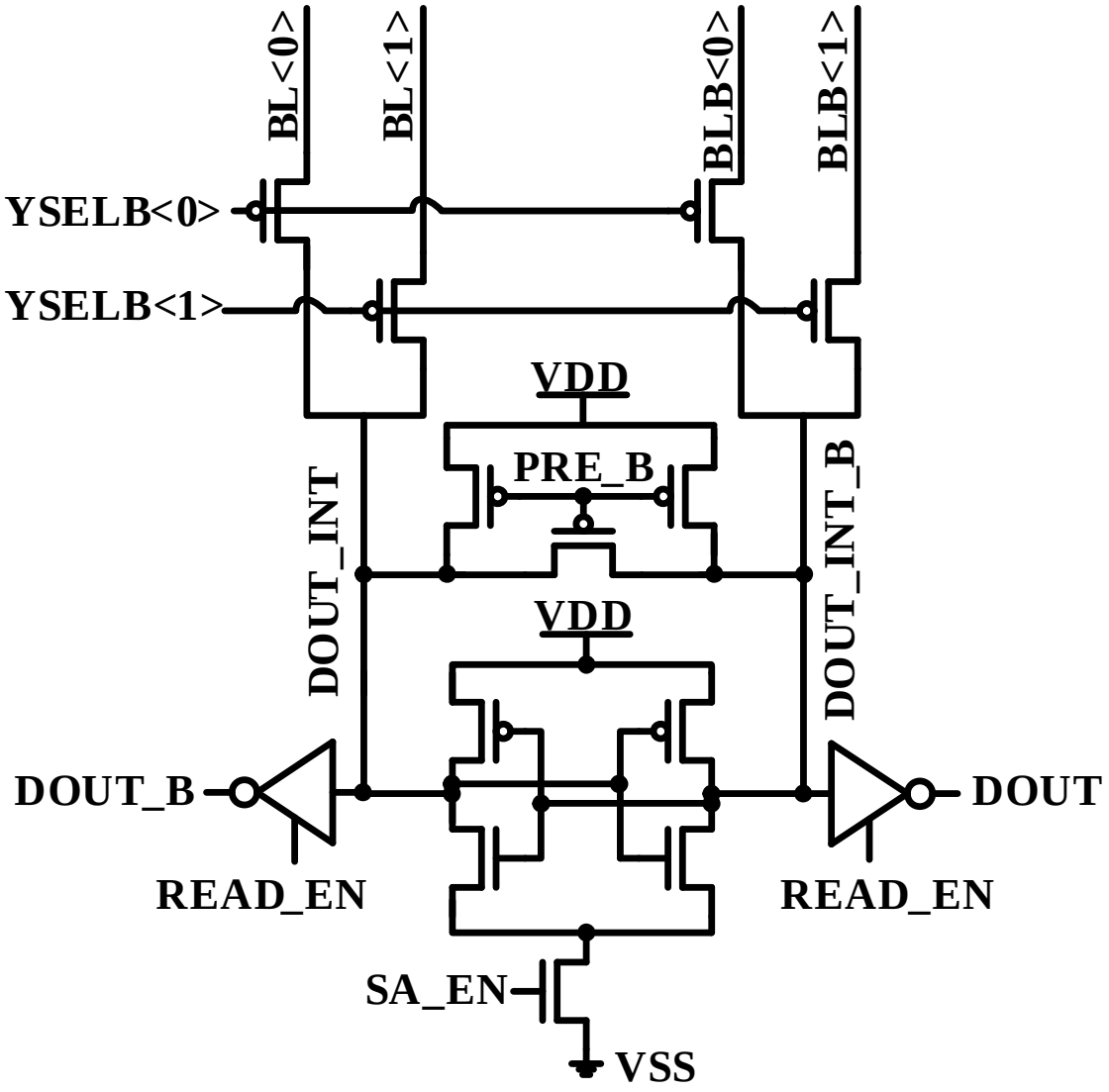


Figure 4.1: **Voltage-type conventional sensing scheme.** A sufficient bitline differential is allowed to be developed between the bitlines, after which the sense amplifier is enabled and the differential is amplified and latched using regenerative feedback.

The main contribution of this work [47] is an area-efficient and variation-tolerant small-signal differential sensing (VTS) scheme that modifies the conventional SA circuit to include: 1) a structure for on-the-fly, auto-zeroing offset compensation, 2) pre-amplification of bitline differential by reconfiguring the SA inverter pair as amplifiers, and 3) latching of the ampli-

fied voltage differential by returning the SA to its conventional cross-coupled configuration. The approach is demonstrated to improve SA robustness at iso-sensing time without area overhead (Figure 4.2). Conversely, sensing time can be reduced at iso-robustness and area. Measurements of a 28nm CMOS test chip show that an iso-area VTS scheme improves offset noise tolerance by $\sim 1.2\sigma_{Vth}$ or sensing speed by up to 42% at iso-robustness (<0.3% failure rate).

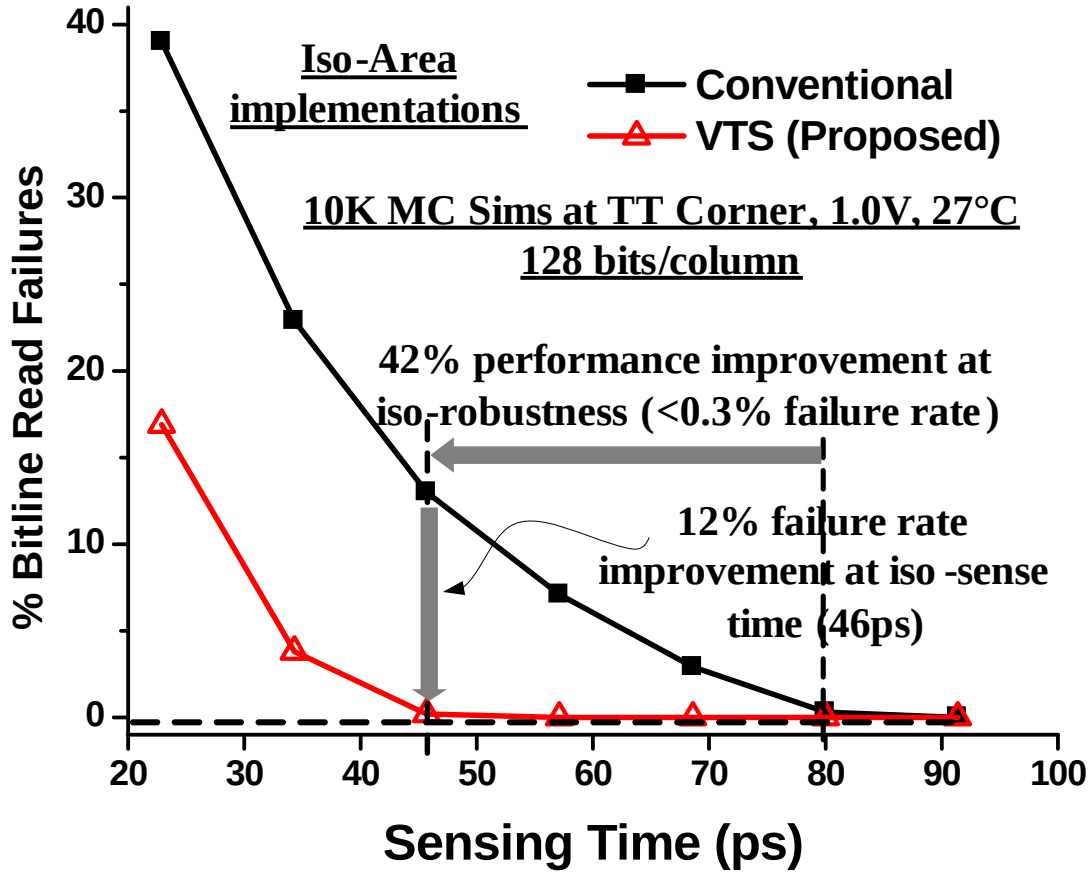


Figure 4.2: **VTS sensing speed/robustness advantage over conventional sensing.** VTS provides 42% sensing speed improvement over an iso-area, iso-robustness conventional sensing scheme in 28nm CMOS (simulated).

4.2 VTS Design

The basic concept of the VTS scheme is to reconfigure the inverter pair of the SA, effectively putting it to use during all phases of operation to provide offset cancellation and additional amplification. Figure 4.3 illustrates the high-level operation of VTS, which consists of three successive SA configurations:

1) During bitline precharge, the SA does not have to detect bitline droop allowing the inverters to be decoupled from each other and biased in their high-gain regions close to their ideal trip-points. AC-coupling capacitors C1 and C2 enable independent biasing of the bitlines and inverters. In addition, the capacitors also compensate for mismatch in the inverter trip-points via auto-zeroing.

2) During reads, the bitcell wordline is activated and the inverters function as offset-compensated pre-amplifiers for the bitline differential (in contrast to conventional SAs, where they remain idle).

3) Finally, the inverters are cross-coupled to further amplify and latch the data using regenerative feedback, as in a conventional SA.

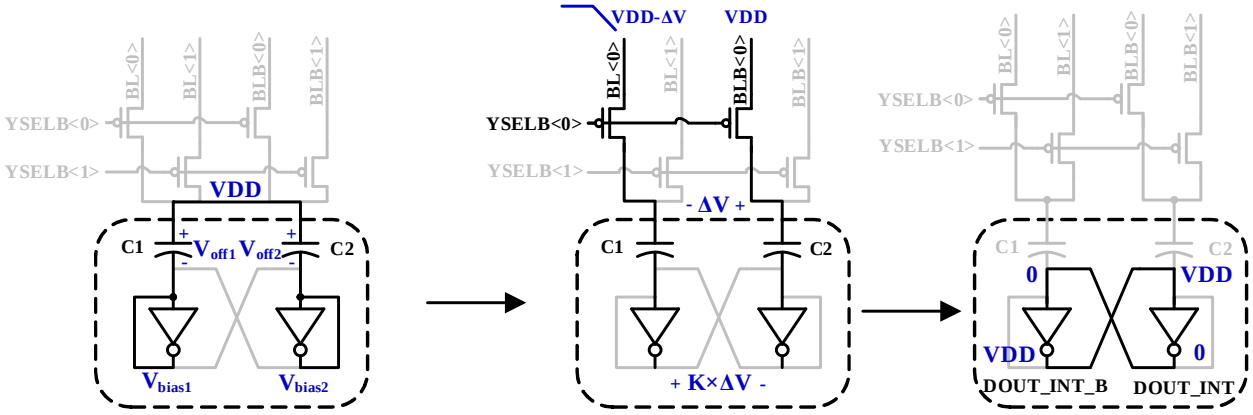


Figure 4.3: **High level operation of VTS.** VTS modifies conventional sensing by reconfiguring the SA inverters through 1) auto-zeroing based offset compensation, 2) pre-amplification of bitline differential ($\Delta V \rightarrow K \times \Delta V$), and 3) latching the amplified differential voltage to recover SA robustness at shorter sensing times.

4.2.1 VTS Circuit Schematic

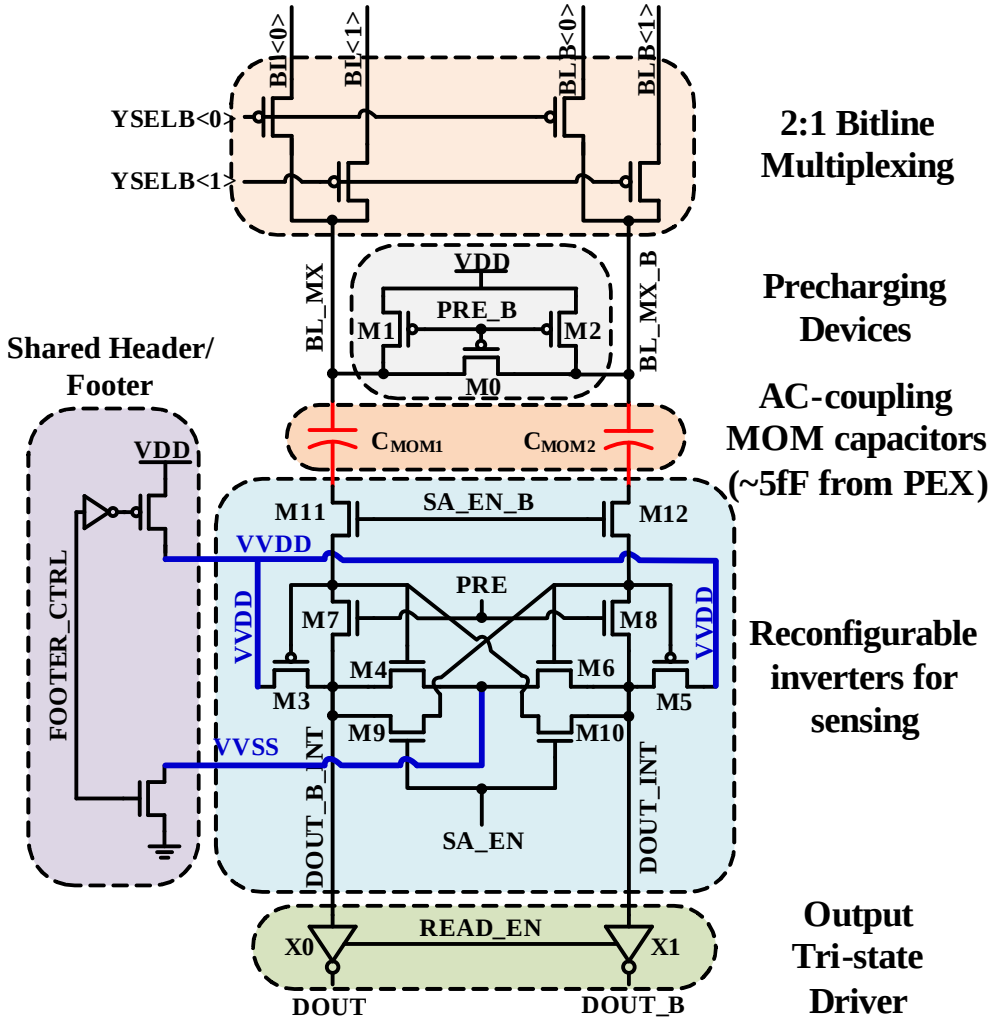


Figure 4.4: **VTS-SA circuit schematic.** The VTS-SA is designed to support 128 bits/column with 2:1 bitline multiplexing. The reconfigurable inverters are coupled to the multiplexed bitlines (BL_MX/BL_MX_B) using capacitors C_{MOM1} and C_{MOM2} that store inverter trip point offsets for auto-zeroing based compensation. Header/footer units (shared across 16 VTS-SAs) are used to duty-cycle auto-zeroing during precharge to reduce short-circuit power draw and provides up to 26% measured power savings.

Figure 4.4 shows the circuit schematic of the VTS-SA. The 2:1 bitline multiplexing, precharge and output driver circuits are similar to those in the conventional SA. The 10-transistor reconfigurable inverter circuits are coupled to the multiplexed bitlines using capacitors C_{MOM1} and C_{MOM2}. Transistors M3-4 and M5-6 form the SA inverters and NMOS switches M7-10 are used to reconfigure inverter connections for auto-zeroing, pre-amplification,

and latching modes. NMOS switches M11 and M12 isolate the MOM capacitors during regeneration, preventing full rail voltage swing at nodes BL_MX/BL_MX_B that could turn on bitline mux switches and severely degrade performance. Since this scheme incorporates automatic offset compensation, the SA is not highly sensitive to mismatch. Hence, all devices in the VTS-SA are near minimum-sized and can fully leverage density improvements from technology scaling. This is in contrast to conventional SAs, which require large devices to reduce mismatch and therefore have not tracked with feature size improvements [48].

4.2.2 VTS Circuit Operation Phases

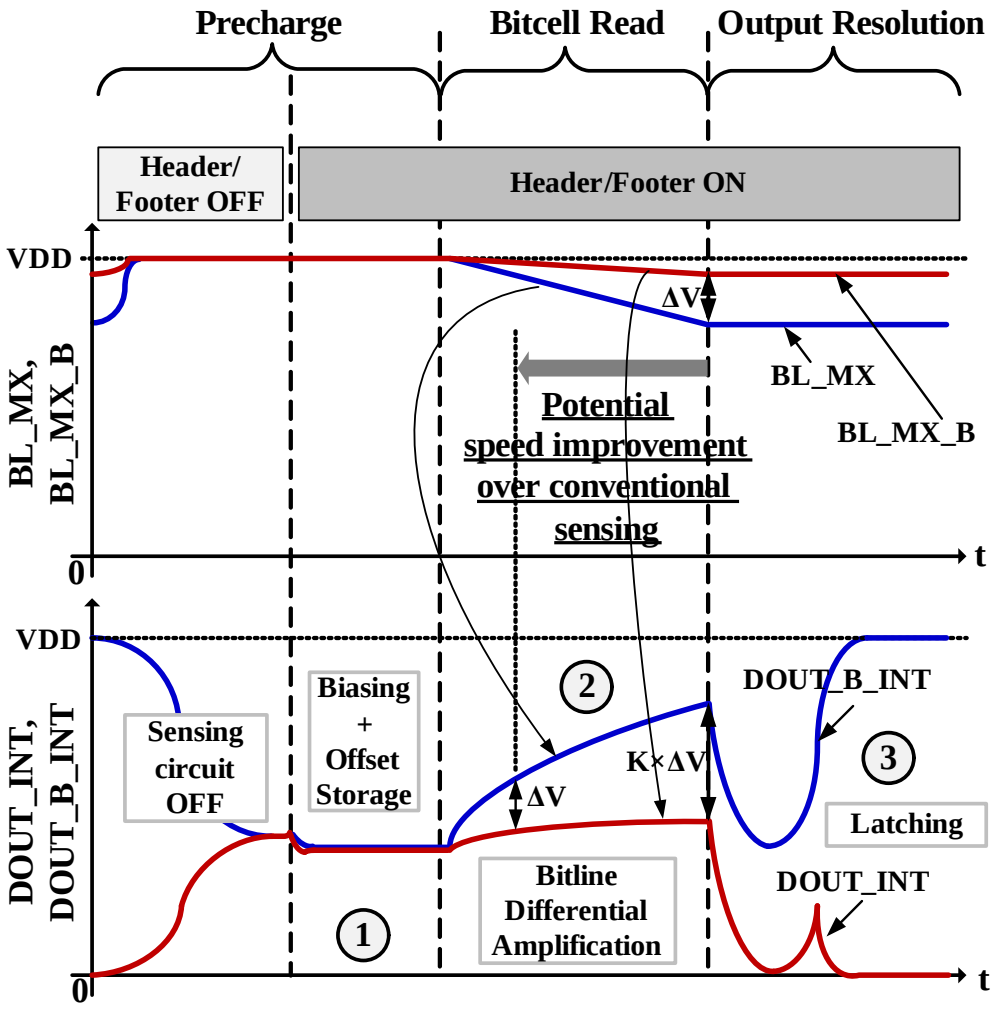


Figure 4.5: **VTS-SA operation phases (I).** The biased inverters provide added amplification during bitcell-reads improving sensing reliability that can conversely be traded for sensing speed.

Figure 4.5 shows simulated waveforms for the selected bitlines and SA inverter outputs through various phases of operation. During biasing/offset storage, the input and output of the SA inverters are shorted together, which creates a 14μA (measured) short circuit current that would increase power consumption in the current scheme. However, biasing and offset storage require only ~60% of the precharge phase to complete and are therefore duty-cycled, resulting in 26% measured SA power savings (compared to no duty-cycling). Headers and footers for duty-cycling are shared across 16 SAs. During the bitcell read phase, the capacitors connect the bitlines to the inverter inputs while their outputs are disconnected. This compensates for inverter trip-point offset and enables pre-amplification of bitline droop (~3.2× larger bitline swing at 60ps sensing time, simulated at TT corner, 1V, 27°C). Finally, the inverters are cross-coupled when SA_EN is enabled for latching. The various SA configurations through its operation phases are shown in Figure 4.6.

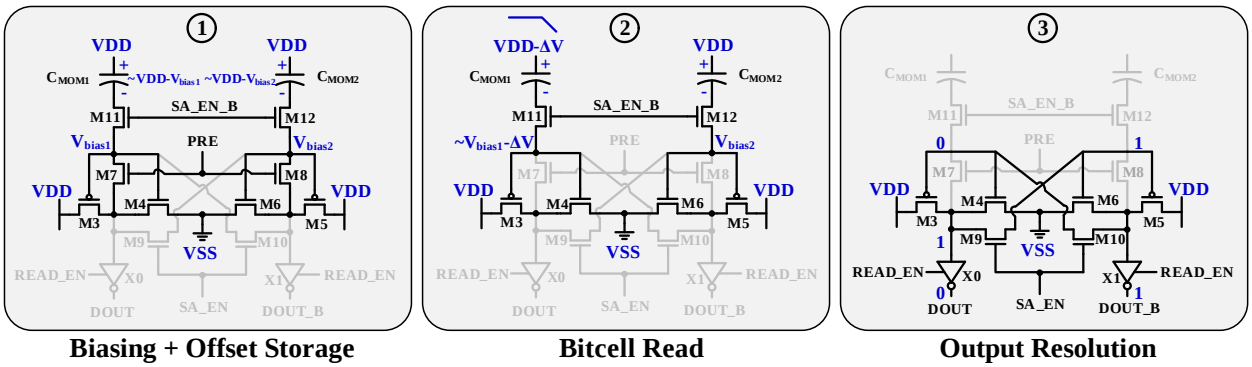


Figure 4.6: **VTS-SA operation phases (II).** The various configurations of the sensing circuit through the operation phases are shown.

4.2.3 VTS Array Design

The VTS-SA is implemented in an 8kb SRAM array composed of high-density 6T bitcells (Figure 4.7). The bitlines are interleaved 2:1 with 128 bits on each column. The capacitors are implemented as 7.8μm × 0.76μm MOM (metal-oxide-metal) devices, rather than: 1) MIM (metal-insulator-metal) capacitors, which have larger minimum size constraints, or 2) MOS (metal-oxide-semiconductor) capacitors, which undergo weak inversion during auto-zeroing, increasing coupling loss. The MOM capacitors are pitch-matched to the SA and

placed on top of two bitcell columns in Metals 5 and 6 (Figure 4.7).

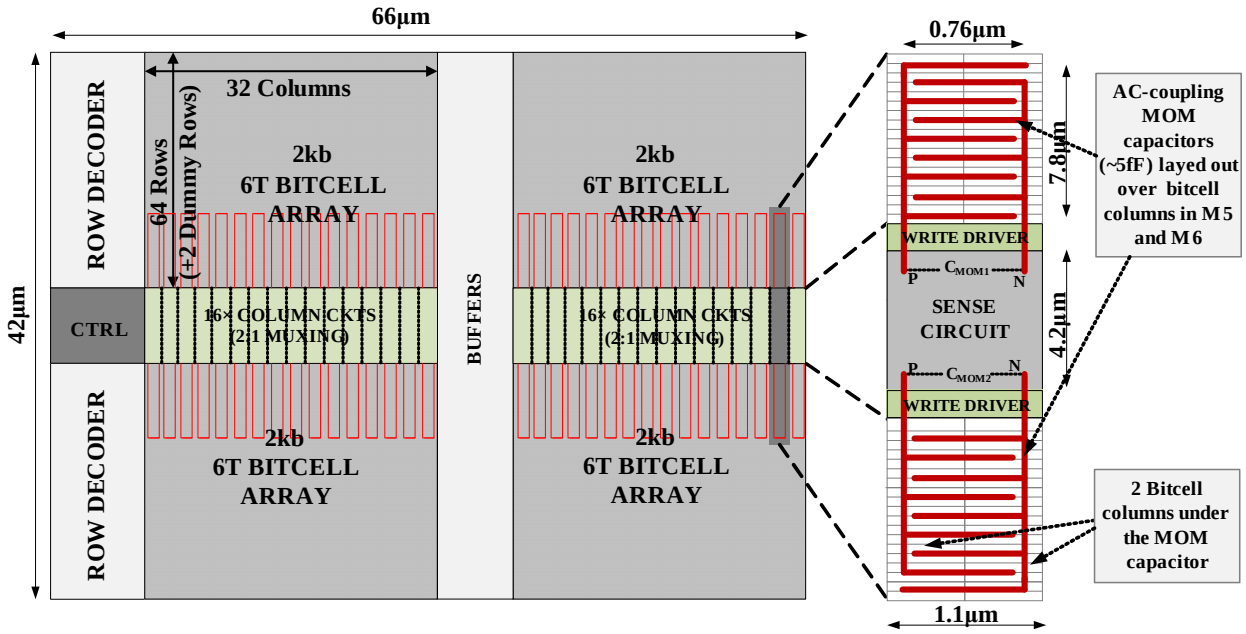


Figure 4.7: **VTS-based array design.** VTS is evaluated with an 8kb 6T array consisting of 128 rows and 64 columns. Each 5fF capacitor is pitch-matched to the column circuit and placed on top of two bitcell columns in Metals 5-6.

The timing diagram for read control signals in the VTS scheme is shown in Figure 4.8.

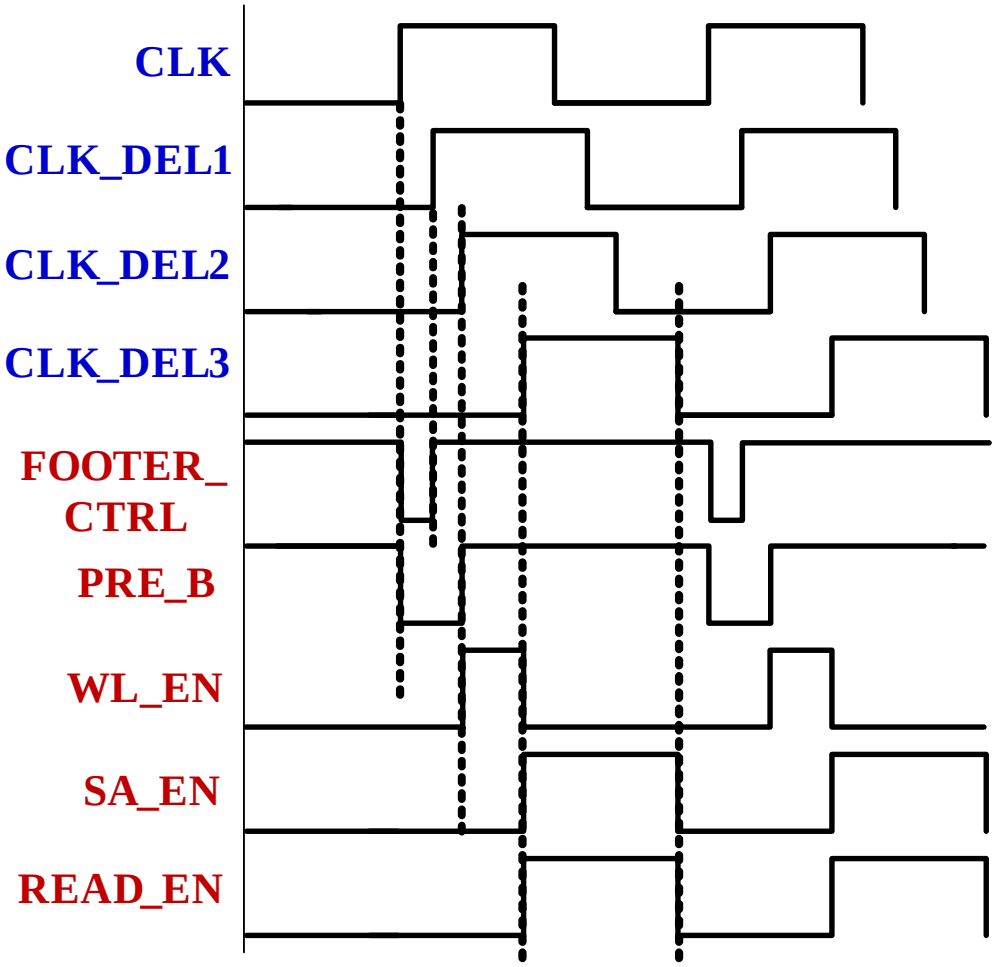


Figure 4.8: **VTS-based array read timing.**

4.2.4 VTS Capacitor Design

The sizes of C_{MOM1} and C_{MOM2} are a critical design parameter in the VTS scheme. Increasing these capacitances both degrades sensing time (due to larger bitline capacitance) and requires up-sizing of the inverter transistors (M3-M6) to charge the capacitors within a given precharge time. In contrast, smaller C_{MOM1} and C_{MOM2} result in reduced coupling, attenuating the input bitline swing and negating the benefit of pre-amplification. Figure 4.9 shows the simulated design-space, which was used to determine capacitor size. In the test chip implementation, ~5fF capacitors are used to maximize gain-bandwidth product, striking a balance between coupling ratio and total bitline capacitance while minimizing area.

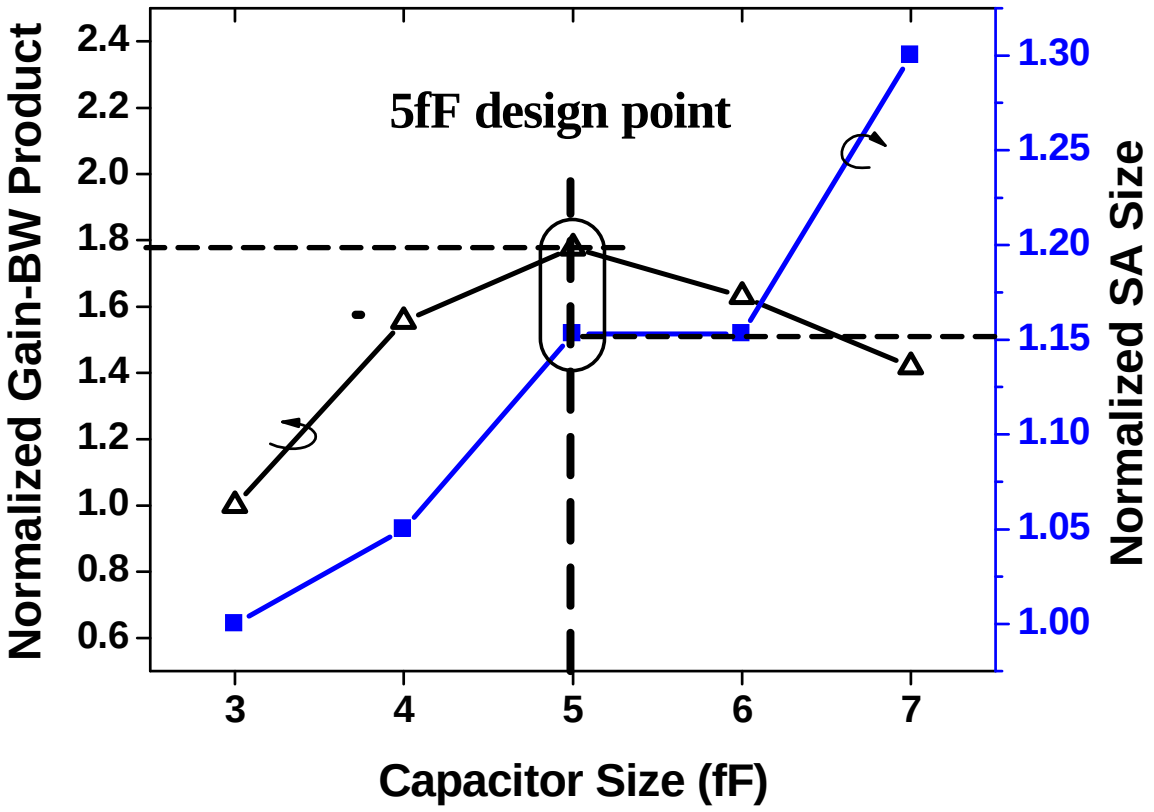


Figure 4.9: **Capacitor sizing design space.** In the current implementation ∼5fF capacitors are used to maximize gain-bandwidth product.

4.3 Test Chip Implementation

To evaluate robustness and speed improvements, a conventional SA-based array is also implemented, where the SA is sized for 4.5σ yield and has an area of $4.62\mu m^2$. Placing the MOM capacitors over the bitcells and using near-minimum sized devices enables an iso-area implementation of the VTS-SA, despite 2× higher transistor count. Because of the additional 2 routing layers used by the MOM capacitor placement strategy, it may not be feasible in some routing resource limited cases (e.g., generic memory compilers). However, custom memory design applications such as processors often have sufficient ($\leq$9) metal layers where a similar implementation can be achieved with little impact on overall routing.

The test harness used to characterize SA performance is shown in Figure 4.10. The arrays

are programmed with pseudo-random data using a 32b LFSR. To measure sensing speed, the WL_EN to SA_EN delay is swept using a two-stage delay chain and any read failures are recorded over 2^{32} experiments operating at 1.8GHz. Similarly, SA robustness (offset noise tolerance) is characterized by skewing the supply voltages of the cross-coupled inverters (to induce mismatch) at a fixed nominal sensing time.

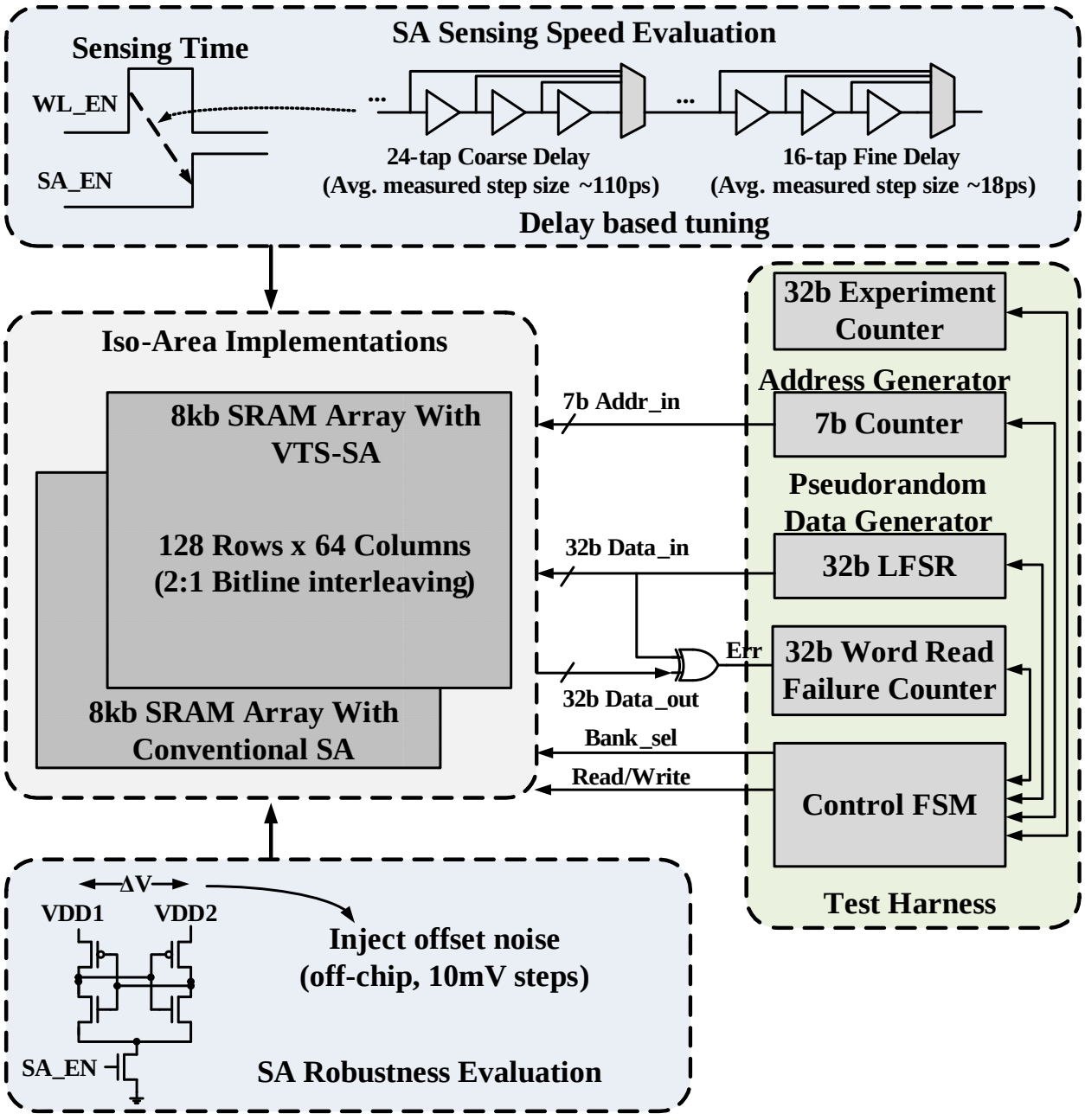


Figure 4.10: **Test chip implementation.** The test harness used to characterize SA performance is also shown.

4.4 Measured Results

Measurements were carried out across 22 dies to characterize sensing time and robustness for conventional and VTS implementations. Figure 4.11 shows that for a typical die, VTS improves sensing time by 34% over conventional sensing at an iso-failure rate of <0.3%. Alternatively, this corresponds to $\sim 0.9\sigma_{Vth}$ higher offset noise tolerance (Figure 4.12).

Figures 4.13 and 4.14 respectively show the sensing speed and robustness characterization across all the 22 dies tested. Across dies, sensing speed improvements range from 25% to 42% (Figure 4.15), corresponding to robustness improvements of $0.6\sigma_{Vth}$ to $1.2\sigma_{Vth}$ (Figure 4.16).

Figure 4.17 shows the VTS-based sensing speed/robustness improvement across temperatures. The VTS sensing circuit consumes 5.1uW at 1.8GHz, which is 10% lower than the conventional SA . Table 4.1 compares the key characteristics of VTS with the conventional sensing approach. Figure 4.18 shows the die micrograph in 28nm CMOS.

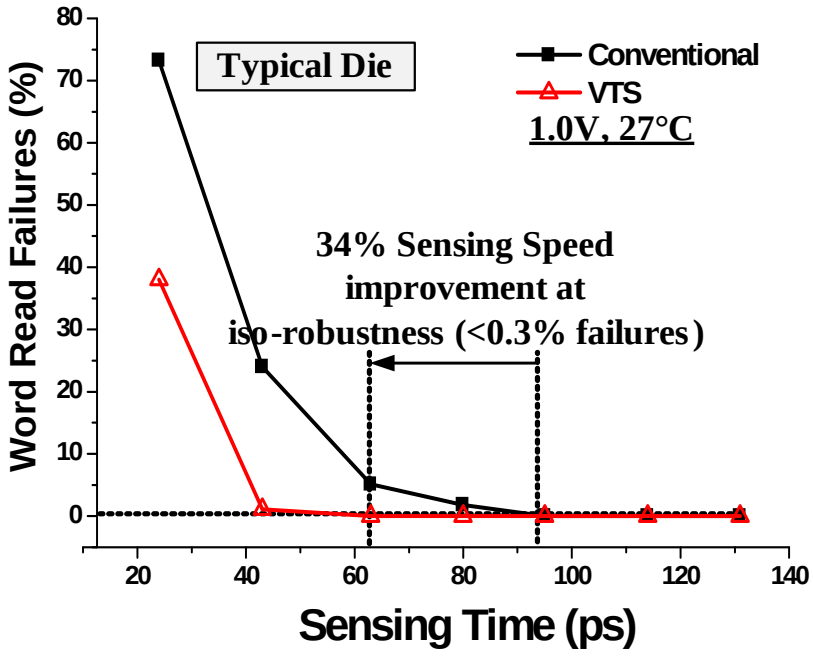


Figure 4.11: **Measured VTS vs. conventional sensing time characterization for a typical die.** VTS improves sensing time by 34% over conventional scheme at an iso-failure rate of <0.3%.

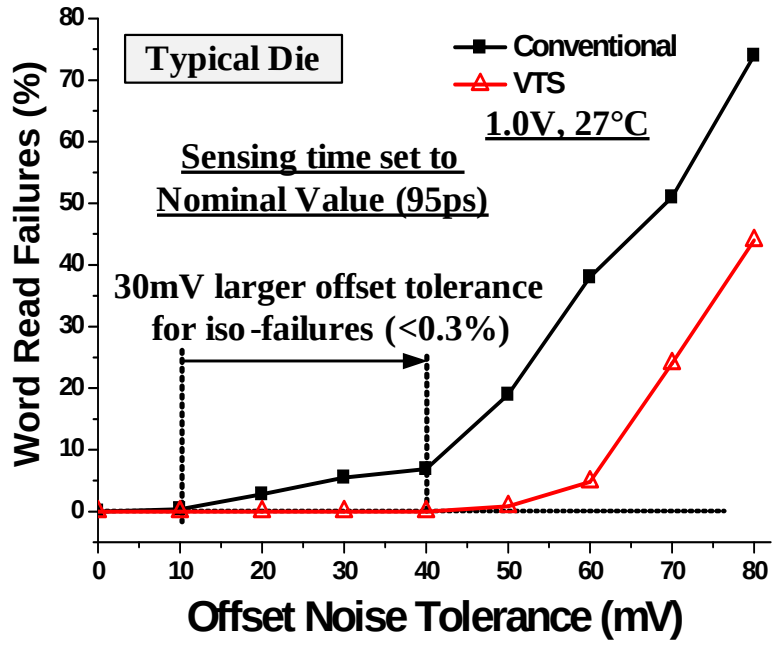


Figure 4.12: **Measured VTS vs. conventional sensing robustness characterization for a typical die.** VTS improves sensing robustness by $\sim 0.9\sigma_{Vth}$ over conventional scheme at iso-sensing time.

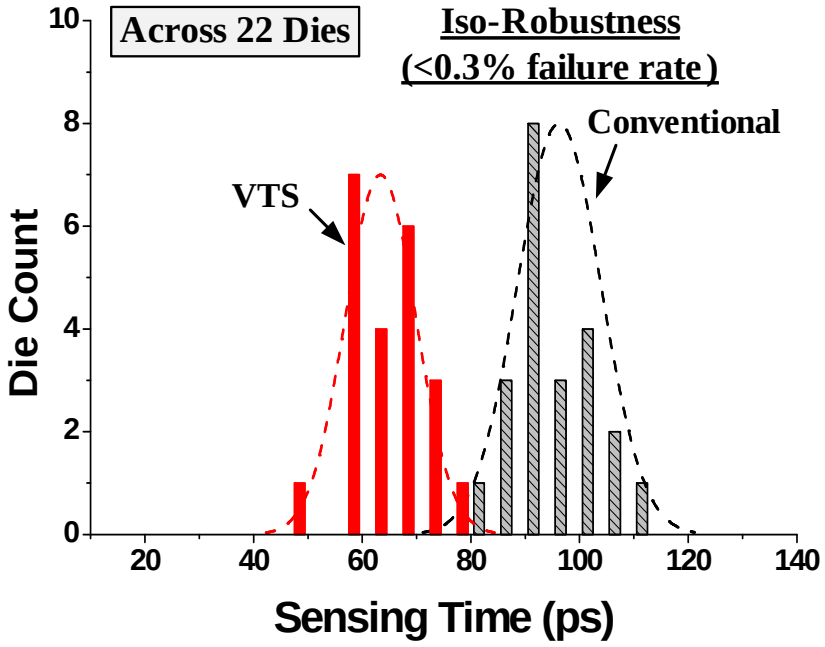


Figure 4.13: **Measured sensing speed characterization across 22 dies.**

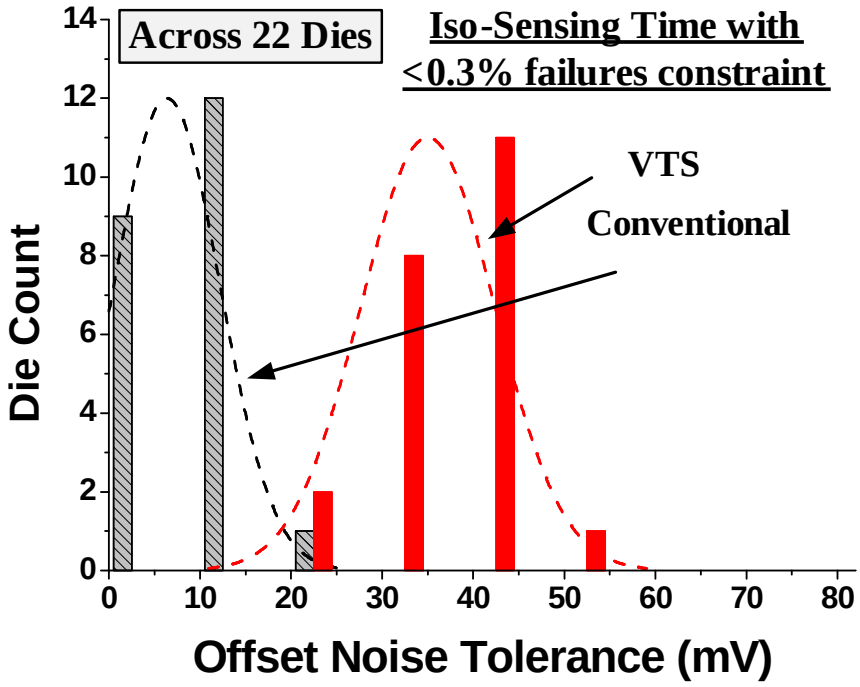


Figure 4.14: **Measured sensing robustness characterization across 22 dies.**

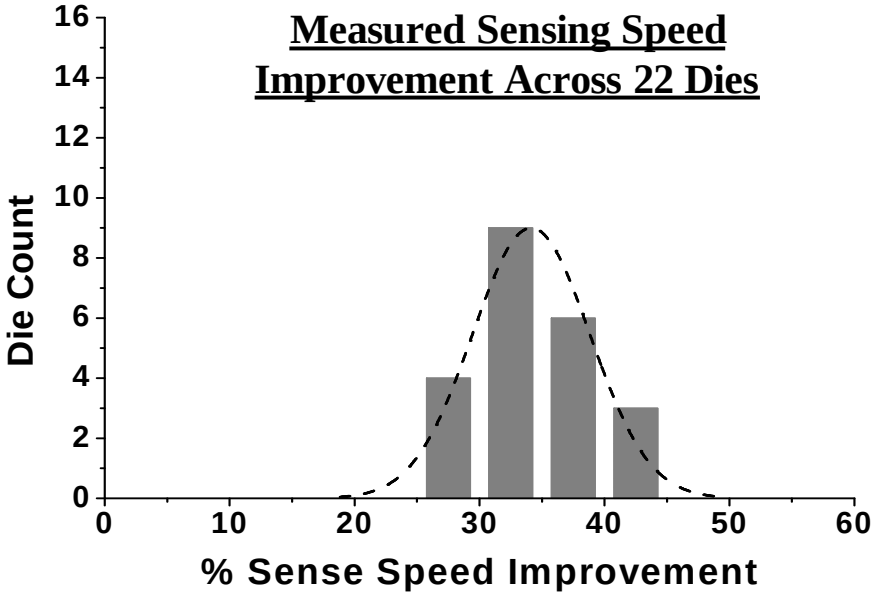


Figure 4.15: **Measured VTS sensing speed improvement across 22 dies.** The improvement ranges from 25% to 42%.

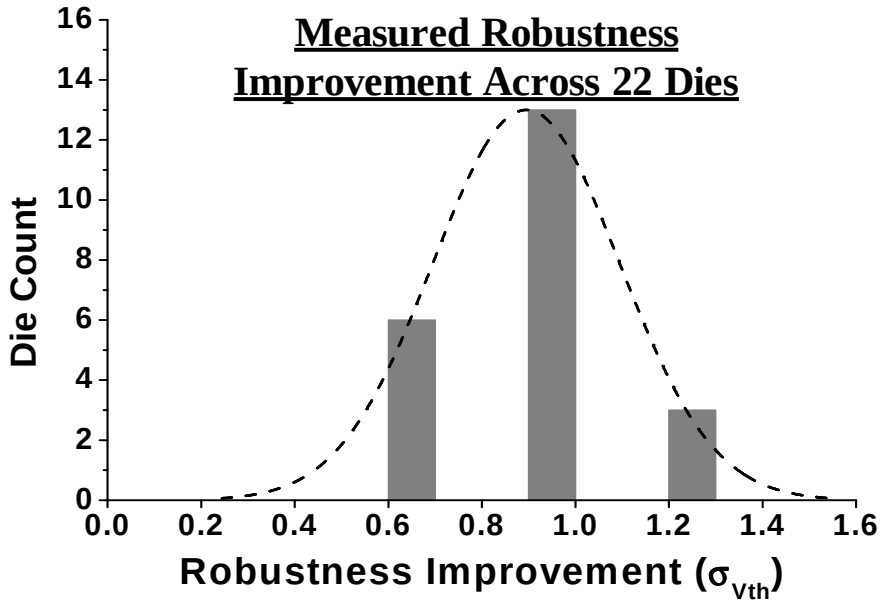


Figure 4.16: **Measured VTS sensing robustness improvement across 22 dies.** The improvement ranges from $0.6\sigma_{Vth}$ to $1.2\sigma_{Vth}$.

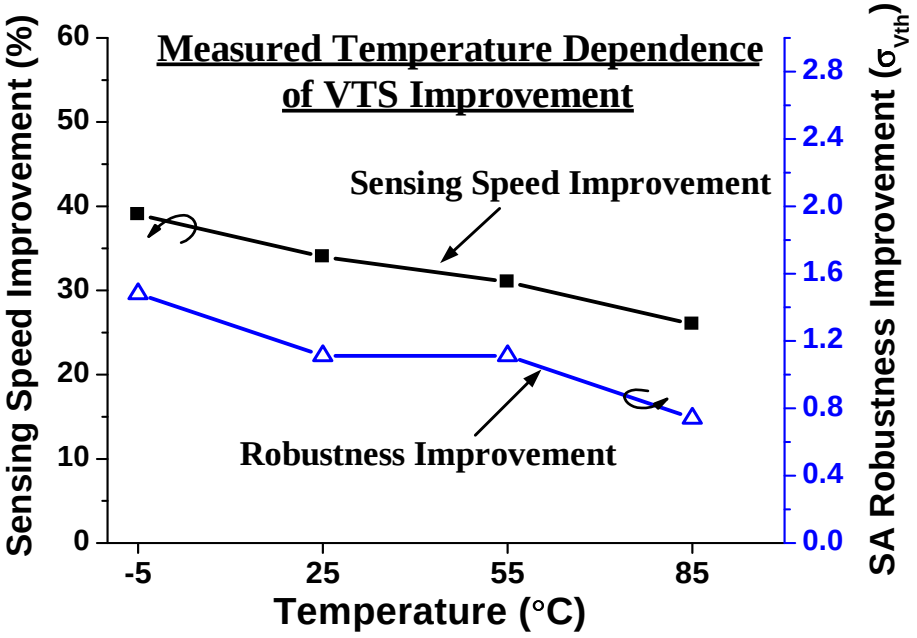


Figure 4.17: **Measured VTS-based sensing speed/robustness improvement across temperatures.** The improvements are relatively stable across temperatures.

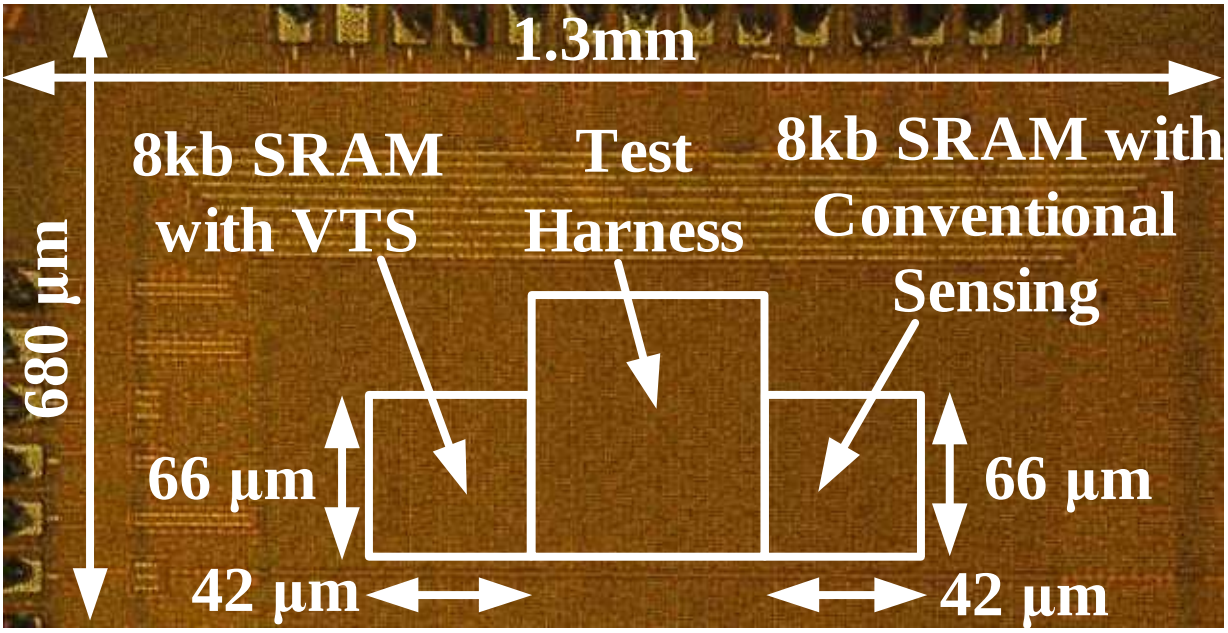


Figure 4.18: **Die micrograph in 28nm CMOS.** The 8kb SRAM arrays with VTS and conventional sensing are highlighted.

Table 4.1: **Key characteristics of VTS compared with conventional sensing.**

	Conventional SA	VTS-SA
Technology	28nm CMOS	
Supply Voltage	1.0V	
Power (1.8GHz)	5.6μW	5.1μW
Sensing Circuit Area	4.62μm^2	
Resert Power	-	14μW
Configuration	128 6T bitcells per column with 2:1 bitline muxing	
Max. Sensing Speed Improvement (at iso-robustness)	-	42%
Max. Robustness Improvement (at iso-sensing speed)	-	1.2σ_{Vth}

CHAPTER 5

Exploring DRAM Organizations for Energy-Efficient and Resilient Exascale Memories

The power target for exascale supercomputing is 20MW, with about 30% budgeted for the memory subsystem. Commodity DRAMs will not satisfy this requirement. Additionally, the large number of memory chips (>10M) required will result in crippling failure rates. Although specialized DRAM memories have been reorganized to reduce power through 3D-stacking or row buffer resizing, their implications on fault tolerance have not been considered. In this chapter, we show that addressing reliability and energy is a co-optimization problem involving tradeoffs between error correction cost, access energy and refresh power—reducing the physical page size to decrease access energy increases the energy/area overhead of error resilience. Additionally, power can be reduced by optimizing bitline lengths. The presented 3D-stacked memory uses a page size of 4kb and consumes 5.1pJ/bit based on simulations with NEK5000 benchmarks. Scaling to 100PB, the memory consumes 4.7MW at 100PB/s which, while well within the total power budget (20MW), is also error-resilient.

5.1 Introduction

The exascale supercomputing program has set a goal of producing an exaFLOP-class computer (capable of executing $>10^{18}$ FLoating point Operations Per Second) within a power budget of 20MW—today's supercomputers perform in petaFLOPS, consuming ∼10MW [49]. GPGPUs have greatly improved processor efficiency in supercomputers [50], and emerging

technologies such as near-threshold computing [51,52] promise to improve this further. However, memory efficiency has been improving at a much slower rate, and exascale machines are projected to require at least 100PB of main memory capable of sustaining a bandwidth of 100PB/s (0.1B/FLOP) [53]. Main memory in today's petascale systems consumes ∼30% of the total power [54], and projections show that a simple scaling of today's DDR3-based memory [55,56] to 100PB will result in a power consumption of ∼52MW. Thus, the memory subsystem by itself is equivalent to the power consumption of about 50,000 homes and far exceeds the 20MW power budget set for the entire system. Additionally, such a memory will suffer from soft and hard errors so frequently that rollback will take longer than the mean-time-to-failure [57].

DDR4, a newer JEDEC standard, only provides 2× improvement [55] in efficiency which also fails to meet this target. While mobile DRAM standards like LPDDR2/3 [58] provide larger improvements (6-7×), they offer 3-5× less bandwidth compared to DDR4, thus requiring a much larger number of chips for the same performance. This in turn increases their fault tolerance requirements. With 3D-stacking, DRAM chips can be integrated very tightly with logic dies. These logic dies can take over some of the operations of the memory controller and allow much of the internal organization details to be hidden from the processor. This opens up flexibility in defining the bitcell array configurations, such as the size of the row buffer. High-performance computing (HPC) memories such as the Hybrid Memory Cube (HMC) [55] have taken a step in this direction, showing an increasing willingness to move away from pre-existing DRAM standards. Recently, JEDEC also proposed the Wide I/O standard [58] for mobile DRAM that is also based on 3D-stacking and is projected to provide ∼12× improvement in efficiency over DDR3 - however, like other mobile DRAMs, it offers 10× less bandwidth than HMC-class solutions.

Memory errors in current systems contribute more than 40% of the total hardware-related failures and are projected to further increase in exascale systems [59], making error correction capability increasingly important. Hence, any changes to the DRAM organization must consider the impact on error resilience. While several recent DRAM architectures have proposed reducing access energy through 3D-stacking [60, 61] and row buffer resizing/rank subsetting [62–65], they have not considered their implications on fault tolerance, which

needs to be included as an integral part of the power optimization problem.

The focus of this chapter is energy and reliability for exascale memories. We co-optimize error resilience costs, access energy and refresh power to arrive at an energy-efficient and resilient 3D-stacked memory for exascale computing [66]. In addition to its area and power advantage, 3D integration allows us to stack conventional bitcells fabricated in an existing DRAM technology (50 nm) over a 28nm CMOS logic die. This also limits the design risk to just the stacking technology (already demonstrated in commercial products) and is an alternative to more speculative low-power non-volatile memory technologies. In order to address power and reliability, we make the following key contributions:

1. **Reduce DRAM refresh power by restructuring subarrays to minimize bitline capacitance.** DRAM bitcells must be periodically refreshed in order to retain data and a number of efforts have been presented to reduce the associated power consumption [67,68]. In a 100PB memory built using DDR3 chips, refresh power alone can be as high as 3-4MW, consuming 20% of the total power budget for the system, even in standby mode. We optimize subarray column height (bits per bitline) to minimize the total bitline capacitance using a tradeoff between decreased local bitline capacitance and increased muxing/routing capacitance to reduce refresh power. This achieves $\sim$4.6$\times$ savings in refresh power with a 9.7% increase in subarray area.

2. **Optimize the energy/area overhead of including stronger resilience mechanisms.** Traditional DIMM-based solutions use Chipkill [69] to protect against single DRAM chip failures. By analogy, we describe *Subarraykill*—a fault tolerance mechanism implemented on subarrays in a bank. The scheme protects against soft errors (occuring primarily due to particle strikes causing burst errors in a subarray), as well as hard errors (such as multi-bit faults along columns/rows and 3D technology-specific faults such as TSV failures). While most existing schemes use Single-bit Error Correction Double-bit Error Detection (SECDED) ECC in conjunction with Chipkill, we find that for smaller pages, such schemes significantly increase check-bit area/ power overheads. In this work we use rotational Single Byte Error Correction Double Byte Error Detection (SBCDBD) ECC with 4-8b per byte[1] [70] to reduce these overheads. For instance, accessing a 128b data word in a 4kb page

[1]'Byte' in ECC terminology represents a symbol of multiple bits, not necessarily the customary 8 bits.

using a (144, 128) SBCDBD[2] (B=4b) ECC decoder instead of 4×(39, 32) SECDED decoders reduces access energy by 26% and check-bit storage and refresh power overheads from 21.9% to 12.5% without decreasing error coverage.

3. **Include the impact of data locality on the optimal page size.** A physical page consists of data from multiple subarrays of bitcells in a bank. Access energy primarily results from activating rows of bitcells with a RAS (Row Address Strobe). Reducing the page size decreases the energy spent per RAS by activating fewer subarrays. On the other hand, if workloads exhibit good data locality, larger pages are desirable as higher reuse of the page contents reduces the number of RASs and results in greater energy savings. We include this tradeoff in the optimization study by simulating our DRAM model with NEK5000 [71] benchmarks representing anticipated exascale applications.

The presented solution [] is a 32Gb 3D-stacked DRAM with a page size of 4kb, access energy of 5.1pJ/bit and standby power of 0.75pW/bit. For 100PB, the total power consumption is ∼4.7MW at a data bandwidth of 100PB/s. This is an improvement of ∼6.5× over DDR4 DIMM-based solutions and ∼1.8× over the first generation HMC. This leaves 15MW for processors, interconnect, cooling and the other sources of power losses in an exascale system.

[2](144,128) represents 128 data bits and 16 check-bits for ECC, totaling 144 bits of storage.

5.2 Background and Motivation

5.2.1 Power Challenge

Commodity DRAMs, primarily driven by volume, use very few data pins per chip to reduce packaging/test costs. As a consequence, such architectures open a page across multiple chips in a DIMM in order to meet bandwidth requirements. Every access opens an 8kb row buffer in each chip (constituting an 8kB page) in order to fetch a cache line, which is typically only 64B [64]. The spatial locality of single-core workloads takes advantage of this large row buffer; however, with reduced locality in future multi-cores [62] this over-fetch renders commodity DRAMs very energy-inefficient and unsuitable for exascale computing [54]. Thus, using today's DDR3-1333 chips (517.63mW/GB/s [55]), an exascale memory with a 100PB/s data bandwidth would consume ∼52MW, far exceeding the 20MW target for the entire system. With DDR4-2667 chips (309.34mW/GB/s [55]), this decreases to 31MW, which also fails to meet this target. With mobile DRAMs like LPDDR2 (80mW/GB/s) and LPDDR3 (70mW/GB/s) [58] the total memory power further reduces to 8MW and 7MW respectively. However, they provide much less bandwidth (4.3-6.4GB/s) compared to DDR4 (21.34GB/s), thus requiring a much larger number of chips to achieve the same performance. This, in turn, impacts cost, area and fault tolerance requirements.

To address these issues, we explore 3D integration. In addition to its inherent area and power benefits, 3D-stacking also decouples bitcell and logic design, which are known to have contrasting process needs: (1) DRAM bitcells benefit from a low-leakage process for longer data retention; (2) the peripherals (such as sense amplifiers, wordline drivers and I/O) are designed for high speed and benefit from a high performance process. 3D-stacking allows DRAM designers to integrate these distinct processes, as well as reduce latency and increase bandwidth through the use of a large number of high-speed TSVs. The first generation HMC is one such example of a 3D-stacked memory that improves the power efficiency to 86.5mW/GB/s while providing a per-stack bandwidth of 128GB/s [55]. With over 7× power savings compared to DDR3, a 100PB main memory built using HMCs will consume ∼8.8MW. While this is within the power budget of the whole system, it still amounts to a sizeable 44%—more than the 30% used as today's rule of thumb [54] for memory.

Recently proposed Wide I/O mobile DRAM, also based on 3D-stacking, is projected to further improve the efficiency to 40mW/GB/s [58] (4MW for an exascale memory) - however, it only provides a bandwidth of 12.8GB/s (10× lesser than HMC) and also does not give any special consideration to error resilience.

DRAM memories also consume power due to refresh. With the 64ms refresh interval, a 1Gb DDR3-1333 chip consumes 4mW of refresh power [72]. An exascale main memory built using such chips will consume as much as 3-4MW for refresh alone, even in standby mode. More recently, several approaches [67, 68] have focused on optimizing refresh interval as a function of process and temperature variations. However, there is scope for additional power savings, orthogonal to such techniques, by reorganizing subarray layout.

5.2.2 Resiliency Challenge

In addition to meeting stringent power constraints, large scale machines also require stronger mechanisms for error resilience. Errors can be broadly classified as soft and hard. Soft errors are generated by energetic particle interactions with semiconductor devices and are transient in nature. Such particle strikes typically affect multiple bits and cause burst errors in modern DRAM processes. This failure mode has traditionally been considered the dominant fault mechanism in DRAM DIMMs. Errors are detected, and may also be corrected, using ECC. Commodity ECC DIMMs use a (72, 64) SECDED ECC to protect 64 bits of data using 8 check-bits and are constructed using 18 x4 chips (x4 ECC DIMM), or 9 x8 chips (x8 ECC DIMM) [73]. They use a 72b wide data path where the additional DRAM chips are used to store both information and check-bits. The 4 (or 8) bits coming from each chip are spread apart spatially such that no more than one bit is affected by a particle strike—allowing for SECDED ECC to correct such errors.

Hard errors, on the other hand, are related to manufacturing process variations and device wearout. They can be intermittent (data pattern dependent) or permanent in nature [74]. 3D-stacking technology introduces another source of hard errors in the form of TSV faults. Recent studies suggest hard errors are at least as significant as soft errors in large-scale systems [74–76]. Accordingly, most server-grade DIMMs use stronger protection mechanisms

which can tolerate a whole-chip failure, known as Chipkill. One way of implementing Chipkill is by sending each data bit of a DRAM chip (e.g. each of the 4 bits of an x4 chip) to a separate ECC word (a set of data and check-bits over which the ECC algorithm performs error detection/correction). The other method of implementing Chipkill employs the use of stronger codes, such as SBCDBD ECC, that can correct multiple-bit errors. Such codes use Galois Field (GF) arithmetic with b-bit symbols (or bytes) to tolerate up to an entire memory chip failure. The (144,128) SBCDBD with 4b per byte is one such code which can correct up to 4 adjacent bit errors. This code has the same check-bit storage overhead as the (72, 64) SECDED ECC (12.5%), but has much higher detection and correction capability. This ECC scheme also has a highly parallel implementation, with at most 3-4 gates in the critical path. Thus the delay overhead of this scheme is ¡0.4ns in 28nm CMOS.

Although such mechanisms must ideally be able to correct for all such DRAM failure modes, this is not always possible. In such cases a rollback to the last checkpoint is performed (which incurs additional power and computation latency penalties). Finally, software intervention is required to retire pages with permanent failures [77].

5.3 Preliminary 3D Architecture

5.3.1 Basic Organization

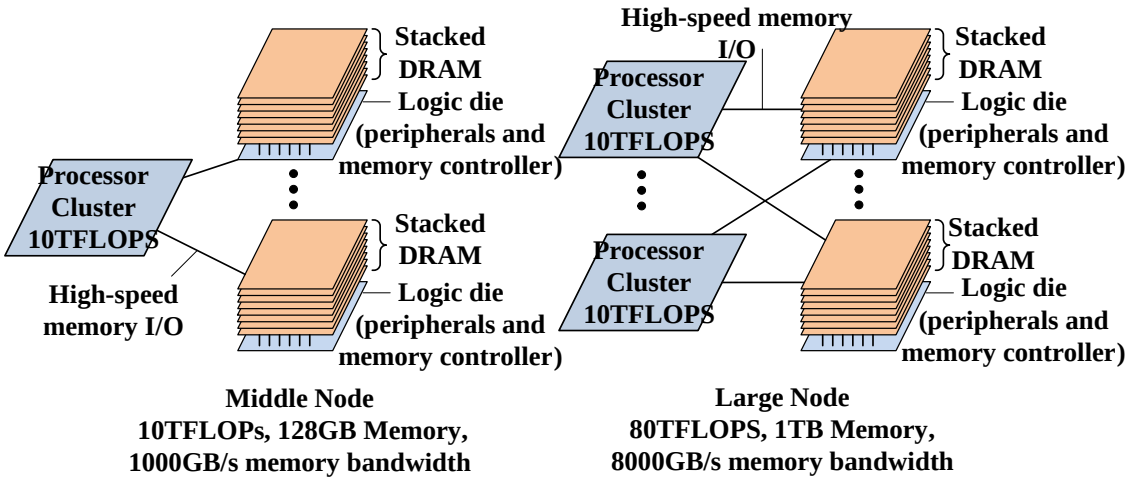


Figure 5.1: **Middle and large node architectures for exascale computing [78].**

This research proposes building a 100PB main memory using smaller 3D-stacked devices based on Tezzaron's 3D-stacking process [79] that allows us to stack 8 memory layers on a logic die through finely-spaced (1.75μm pitch) low-power TSVs. The TSVs have a feedthrough capacitance of 2-3fF and a series resistance of $<3\Omega$. This allows as much as 4GB of data in each individual stack. We assume a middle to large node architecture (Figure 5.1) where these 3D-stacked devices will be connected to processor dies through high-speed I/O [78] and then multiple such nodes will be organized into racks across the entire system to make 100PB of total memory ($\sim 2.5 \times 10^7$ of these 3D chips in total in the overall system).

Building such a system has a number of design challenges. As a starting point, the focus of this work is on the power and resilience of the building block—the 32Gb 3D-stacked DRAM. The 32Gb stack's logical organization is shown in Figure 5.2. This will serve as the base architecture for co-optimizing energy and resilience. Note that the '32Gb' capacity only accounts for data bits and does not include the check-bit storage area overhead which depends on the choice of ECC (discussed in Section 5.4). Each 32Gb 3D chip consists of 8 4Gb DRAM memory dies stacked on top of a logic die. The organization of the 4Gb DRAM

die is based on Tezzaron's existing Octopus [80] DRAM solution. Each 3D stack has 16 128-bit data ports, with each port accessing an independent 2Gb address space. Each address space is further subdivided into 8 256Mb banks. Each bank, in turn, is physically organized as 64×64 matrix of subarrays (not including subarrays for storing ECC check-bits). Each subarray is a 256×256 arrangement of bitcells and is 60μm×35μm (Figure 5.4).

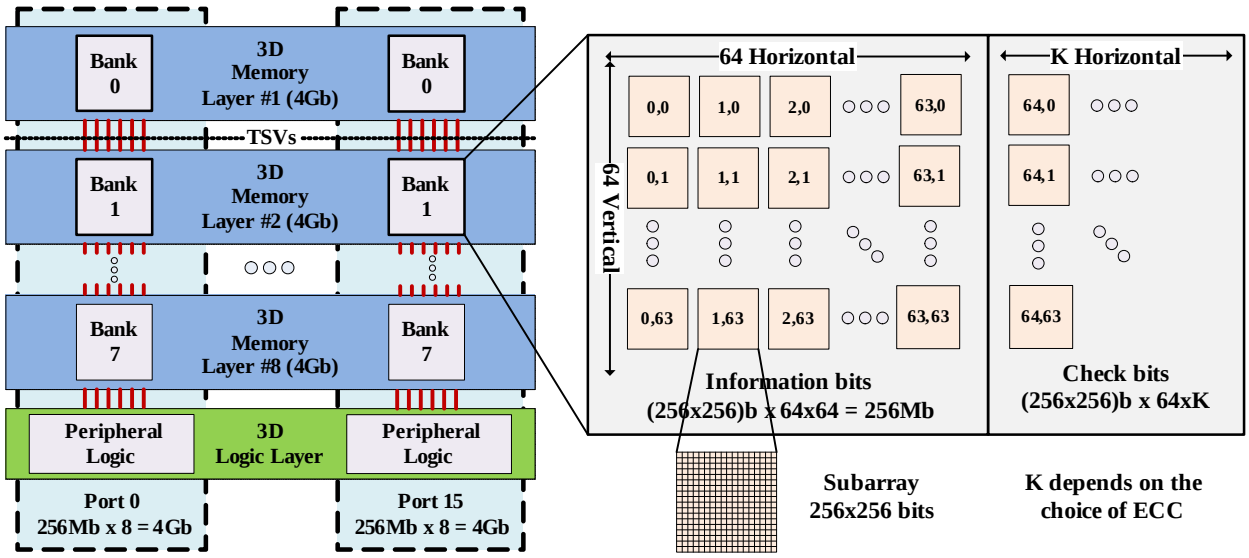


Figure 5.2: **Logical organization of the 32Gb 3D-stacked DRAM.** The DRAM capacity (32Gb) only accounts for information bits and does not include check-bit storage overhead, which depends on the choice of ECC.

Figure 5.3 shows the physical floorplan of each 4Gb DRAM memory die and the logic die. The logic die is fabricated in a 28nm CMOS process and consists of address-decoding logic, global wordline drivers, sense amplifiers, row buffers, error correction logic and low-swing I/O logic with pads. Each memory die is partitioned into 16 ports with each port serving 1 of the 16 banks on a die (Figure 5.3). The memory die is fabricated in a 50nm DRAM process and consists of the DRAM subarrays along with some logic such as local wordline drivers and pass-gate muxes. While there are more advanced DRAM processes (e.g. 20nm), TSV yield in existing 3D-stacked prototypes has only been proven up to the 50nm DRAM process node [55,81]. All subarrays in a vertical stack share the same row buffer using TSVs and hence at most one row of subarrays in a vertical stack can have its contents in the row buffer, which corresponds to a physical page. Thus, assuming an 8kb page, a maximum of 2048 pages can be simultaneously open per device (128 8kb pages per bank × 16 banks

per physical layer), providing concurrency similar to Sub-Array Level Parallelism [82]. The device provides a sustained bandwidth of 6.25GB/s per port (100GB/s total).

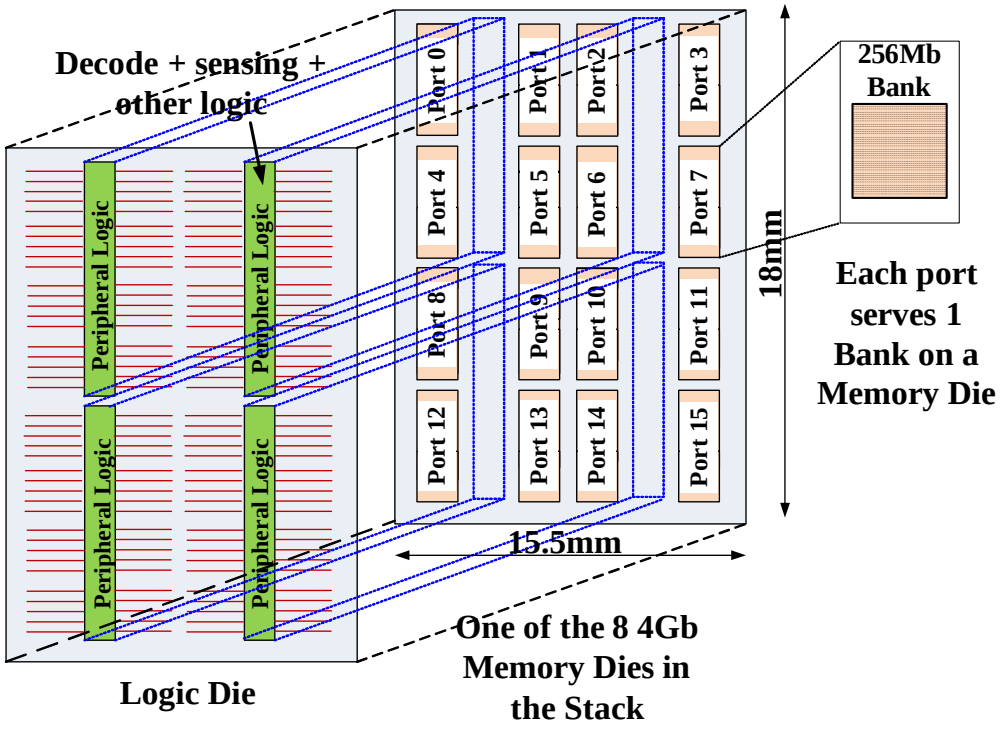


Figure 5.3: **Physical floorplan of the logic die and a 4Gb memory die in the 3D stack (ignoring ECC overhead).** The top-down view of the stack shows that DRAM banks are arranged around 'spines' of peripheral logic.

We now describe DRAM operations for this architecture for an 8kb page (information bits), which is the typical size today [64]. Note that for smaller page sizes, the decoding and other peripheral logic will change accordingly. DRAM operations are pipelined and each port can perform up to three tasks in each clock cycle. Hence, at each port, one bank could be doing a RAS while another one could be simultaneously doing a Column Address Strobe (CAS); and a third one could be doing a Page Close (PC). Thus, in each cycle, we can open 16 new pages, read/write 16 data words and close 16 other pages [80].

5.3.2 RAS Operation

During RAS (Figure 5.4), the address is first decoded in the logic layer to determine the bank and the subarrays to be activated. The decoded address is sent through the TSVs to the

corresponding memory die containing the row of subarrays to be accessed. This ultimately activates one wordline in 32 subarrays out of the 64 subarrays in a row, (i.e. $32 \times 256 = 8192$ bitcells), causing them to charge share with their corresponding bitlines. In an error-resilient memory with ECC additional subarrays will have to be activated due to the check-bit storage overhead. The final charge-shared values are sensed using sense amplifiers on the logic layer (through TSVs) and the data is latched onto the row buffer (also on the logic layer) and a page is opened.

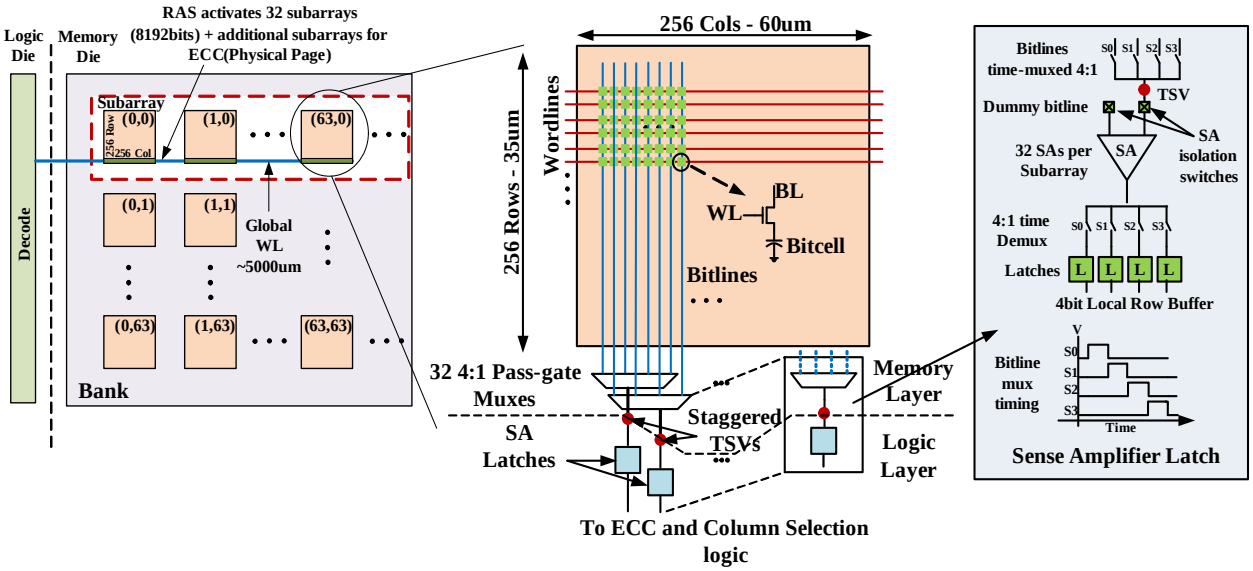


Figure 5.4: **Diagram showing how RAS operations are performed in each bank.** The bitlines are 4:1 time multiplexed to one TSV as the TSV pitch (1.75μm) is much larger than the bitline pitch (0.5μm).

Sensing Scheme

In a conventional DRAM memory, the bitlines are initially precharged to VDD/2 and are allowed to charge share with the accessed bitcells. The sense amplifier then compares the new bitline voltage with a dummy/reference bitline at VDD/2 and stores this value onto a latch. The sense amplifier also causes a full swing on the bitline, thereby re-enforcing the bitcell data. Since the TSV pitch (1.75μm) is much larger than the bitline pitch ($\sim$0.24μm), we stagger the TSVs and employ time-multiplexing on the bitlines to limit the number of TSVs and meet the minimum TSV pitch requirement. Note that the keep-out area around

the TSVs is 500nm, allowing logic to be very close to the TSVs. The bitlines are 4-way time-multiplexed to one sense amplifier. The mux output drives a TSV which also serves as a global bitline vertically running through the stacked memory layers. The global bitline terminates on the logic die where it drives a sense amplifier and sensed data is stored into a 4-bit latch local to this amplifier (Figure 5.4, SA Latch). This latch serves as a row buffer local to the sense amplifier. An 8kb row buffer consists of 2048 SA Latch units.

During every RAS, the sense amplifier evaluates 4 times in order to fill the row buffer with the corresponding bitcell values. Since full swing on the bitline involves charging/discharging large capacitances (∼100fF), the standard sensing scheme is slow and does not meet our high speed sensing requirement. We make an optimization by decoupling the sense amplifier from the bitlines during evaluation using isolation switches (Figure 5.4) in order to boost the sensing speed. Additionally, this also allows the sense amplifier to be reset more quickly before the next firing. The RAS latency (t_{RCD}) with such a sensing scheme is only 5ns [80] compared to 13.5ns in DDR3-1333 chips [83]. Unlike conventional schemes, the full swing on the bitlines is deferred and performed during the next Page Close (PC). This is acceptable as the data is available in the row buffer throughout this period. The shorter RAS latency also offsets the longer PC (also 5ns).

5.3.3 CAS Operation

During CAS, data is moved from the row buffer to the I/O logic. The address bits are decoded to select the relevant 128 bits from the row buffer to be sent out to the I/O port. Multiple CAS operations can be performed per RAS as long as the access is to the row already stored in the buffer. In order to protect against errors, ECC decoding and correction is also performed in this phase (Section 5.4). The corrected data is then sent to the I/O port using a low swing interface consuming 1pJ/bit at 1GT/s [80, 84]. The CAS read latency is 2.5ns. Finally, during PC, the row buffer contents are written back to the subarray row and the buffer is released.

Table 5.1: **Initial break-down of access energy in the 3D DRAM architecture for an 8kb page size (no optimizations included).**

RAS+PC Energy (pJ/b)	12.7
CAS Energy (pJ/b)	2.85
Low-swing I/O Energy (pJ/b)	1.00
Total Access Energy (pJ/b)	16.6

5.3.4 Access Energy Reduction

Table 5.1 illustrates how RAS energy dominates the total access energy per bit; it assumes that every access results in both a CAS and a RAS. There are two competing approaches to reduce total RAS energy—(1) reduce the total number of RAS operations; or, (2) reduce the energy per RAS operation. While large page sizes suffer from higher energy per RAS, spatial data locality will reduce the number of RASs by increasing the reuse of page contents. In contrast a smaller page size (activating fewer subarrays per RAS) reduces the energy per RAS, but suffers more RAS operations due to loss of data locality. Further complicating the tradeoff is that smaller pages tend to have higher ECC check-bit storage/power overhead (Section 5.4), which also increases access energy. We use these tradeoffs to arrive at the final, energy-optimal page size in Section 5.6.1.

5.3.5 Refresh Power Reduction

Tezzaron's 256 × 256 subarray has total bitline load of ∼100fF, bitcell capacitance of 25fF and VDD of 1.2V. Each bitcell needs to be refreshed within 64ms—a common DRAM standard. This implies that if a bitcell is read at the end of 64ms (just before a refresh), the final charge-shared voltage on the bitline, $V_{chargeshare}$ should still be sensed correctly. Typically, the sense amplifier requires a certain voltage margin between the sensed and the dummy bitline (at VDD/2 = 0.6V) in order to guarantee correct operation. This is around 100mV accounting for process variation and mismatch. Thus the minimum $V_{chargeshare}$ for sensing a '1' in our scheme is 0.7V.

If the subarray layout is changed such that the number of bits on a bitline is reduced (Figure 5.5), the total capacitance charge-sharing with the bitcell during reads can be reduced. With reduced charge-sharing, $V_{chargeshare}$ retains a higher voltage value and hence the bitcell

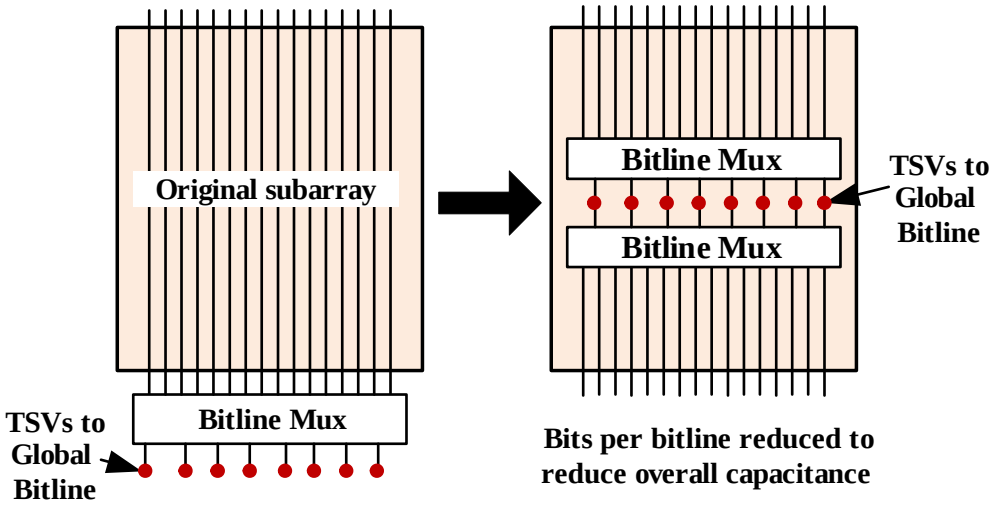


Figure 5.5: **Reorganizing subarrays to shorten local bitlines and reduce refresh power.**

can also leak longer while meeting the minimum voltage requirement on $V_{chargeshare}$, before requiring refresh. This concept can be used to reduce the refresh power by (1) reducing the total capacitance charging/discharging during refresh, and (2) increasing the mean interval between refreshes. This technique is also effective in reducing variations in required refresh intervals among the bitcells, because charge-sharing between the bitcell and the bitline is linearly proportional to the bitline capacitance, thus the mean as well as the standard deviation of the refresh interval distribution improve linearly.

However, as the number of bits per bitline reduces, the capacitance of muxing (including routing through TSVs) increases (Figure 5.6) which eventually negates further refresh savings (Figure 5.7). Also, the increased area overhead of the extra TSVs and bitline muxing also reduces the subarray area efficiency. We choose to employ 64 bits on a bitline to get a $\sim$4.6$\times$ savings in refresh power. This increases the subarray area by 9.7% due to additional TSVs and muxing. Note that reducing bitline capacitance also reduces the energy spent on RAS during an access. For an 8kb page, moving to 64 bits on a bitline (from 256 bits) reduces RAS energy from 12.7pJ/bit to 8.6pJ/bit.

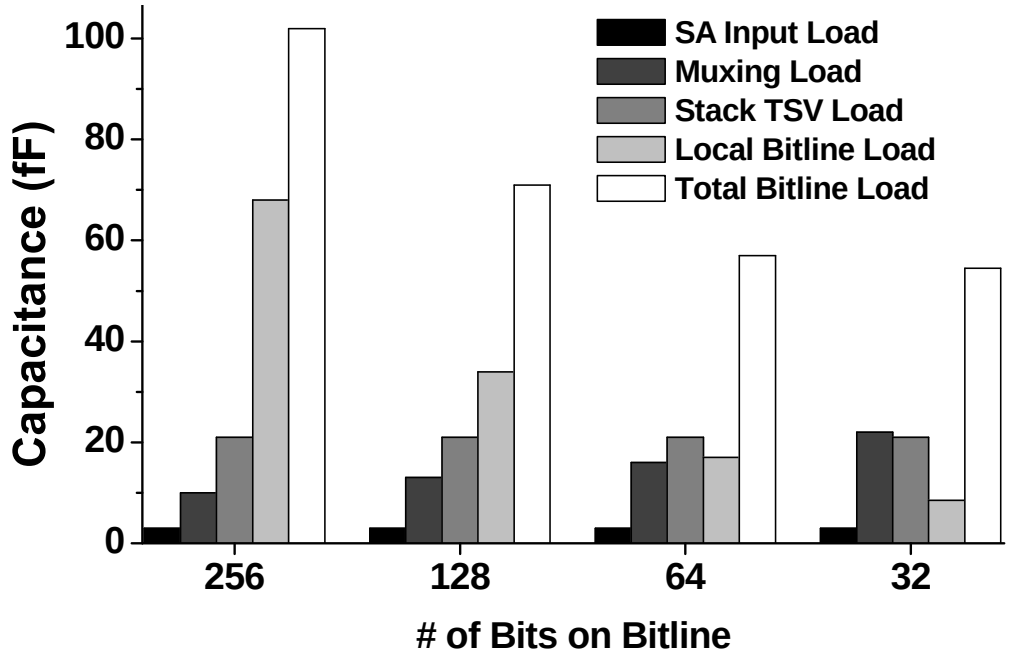


Figure 5.6: **Components of charge-sharing capacitance as a function of number of bits on a bitline.** While bitline capacitance reduces, the muxing capacitance increases.

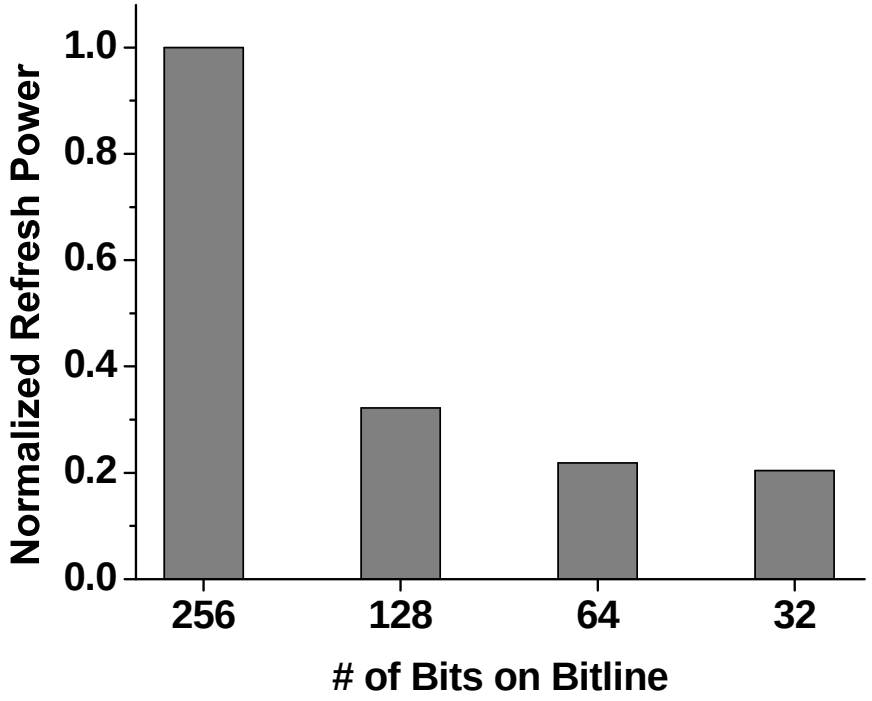


Figure 5.7: **Refresh power savings as a function of number of bits on a bitline.** The increase in muxing and routing power offsets the savings in bitline swing power as we move to 32 or fewer bits on a bitline.

5.4 3D Memory with Subarraykill

In this section we propose Subarraykill—a resilience approach intended to protect against soft errors caused by particle strikes and hard errors caused by multiple errors along rows and/or columns in a subarray, referred to as subarray failures. This is analogous to how typical server-grade memory uses Chipkill [69] to protect against single memory chip failures or multiple errors from any portion of a single memory chip on a DIMM. We include hard error protection, because, as noted earlier, hard errors are at least as significant as soft errors [74–76]. In particular, row failures are the most frequent [85]. Subarraykill is implemented by spreading a data word evenly across a whole accessed page in order to tolerate multiple errors in the same subarray. Such errors can be detected and corrected using either SECDED or SBCDBD-based codes, which we compare in the next two sections. In summary, Subarraykill is designed to handle multi-bit faults, column faults and row faults (up to whole subarrays). In addition, since the ECC is on the logic die at the base of the 3D stack, we also protect against TSV failures.

5.4.1 SECDED-based Subarraykill

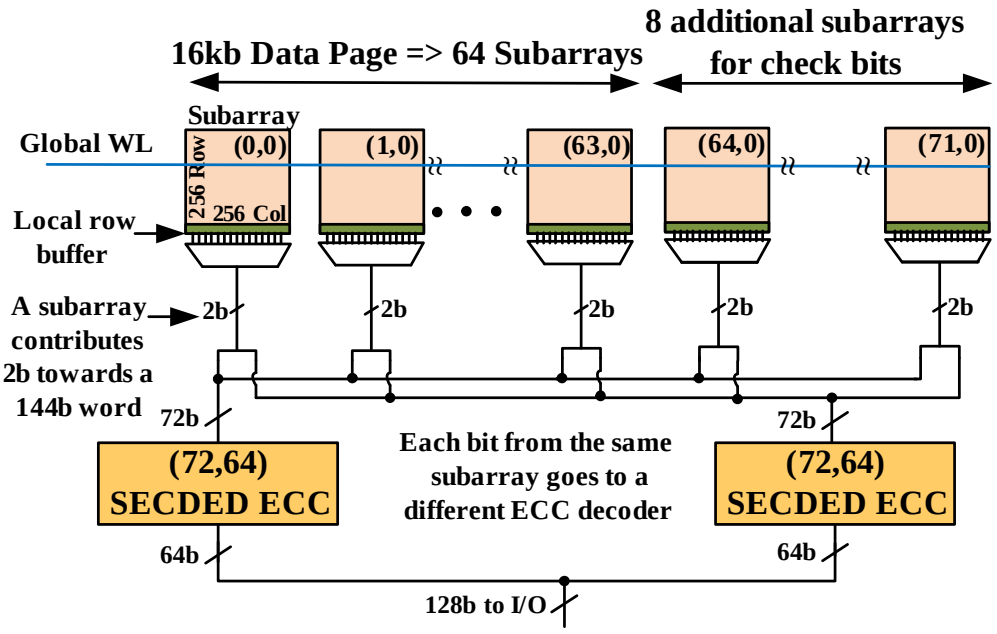


Figure 5.8: **SECDED-based Subarraykill error correction for 16kb page.**

Table 5.2: **Comparing Subarraykill configurations for accessing a 128b information word from different page sizes using SECDED and SBCDBD ECC codes that have the same error correction performance.**

Page size (info. bits)	# accessed data arrays	# sel. bits per sub-array	SECDED Config.	SECDED Check-bit Over-head	SECDED Latency Over-head in 28nm	SBCDBD Config.	SBCDBD Check-bit Over-head	SBCDBD Latency Over-head in 28nm
16kb	64	2	2×(72,64)	12.5%	0.62ns	1×(144,128), (B=4b)	12.5%	0.88ns
8kb	32	4	4×(39,32)	21.9%	0.49ns	1×(144,128), (B=4b)	12.5%	0.88ns
4kb	16	8	4×(39,32)*	21.9%*	0.49ns*	1×(144,128), (B=4b)*	12.5%*	0.88ns*
			8×(22,16)	37.5%	0.38ns	1×(152,128), (B=8b)	18.75%	1.08ns
2kb	8	16	8×(22,16)*	37.5%*	0.38ns*	1×(152,128), (B=8b)*	18.75%*	1.08ns*
*code guarantees hard error detection (not correction) with soft error correction.								

We discuss the SECDED-based scheme using a 16kb page as an example as shown in Figure 5.8. Borucki et al. [86] showed that a particle strike can affect almost 20 bits in the 110nm process. Since our DRAM uses the 50nm process, we conservatively protect against burst errors of up to 64 bits due to particle strikes. Additionally, we must also protect against hard errors such as row failures. In order to correct burst errors due to particle strikes and row failures, the 144-bit code word is spread across 64+8 subarrays and each subarray only contributes 2 bits towards the data word. During a read, each of these 2 bits is sent to a separate ECC unit so that in case of a soft error, or a subarray failure, each ECC word can have at most 1 bit in error which can be corrected using a conventional (72,64) SECDED code.

Table 5.2 shows that as the page size is reduced, fewer subarrays get activated and more bits are pulled from each subarray to form a word. With the SECDED scheme, these bits need to be sent to separate ECC words for decoding. Thus, as the page size is reduced, the SECDED code also becomes smaller which increases the storage overhead due to check bits. Table 5.2 also shows the storage overhead of error correction as a function of page size. Note that for a page size of 4kb, storage overhead with SECDED-based Subarraykill is 37.5%,

which reduces the useful DRAM density to an unacceptable amount. This increase in storage overhead in turn increases the energy spent in retrieving these check bits during access and more prominently during refresh (Figure 5.11). So, we propose switching to SBCDBD-based ECC codes which have lower energy and storage overheads as will be discussed in the next section.

5.4.2 SBCDBD-based Subarraykill

We noted earlier (Section 5.3.4) that opening smaller pages directly improves total access energy, because it is dominated by RAS. However, SECDED-based Subarraykill incurs a high energy and area overhead for smaller page sizes, negating the advantages of smaller pages. As a solution, we propose using rotational SBCDBD-based codes which have lower area and energy overheads for the same error correction performance. Rotational codes have the additional merit that hardware implementation has low latency overheads (Table 5.2).

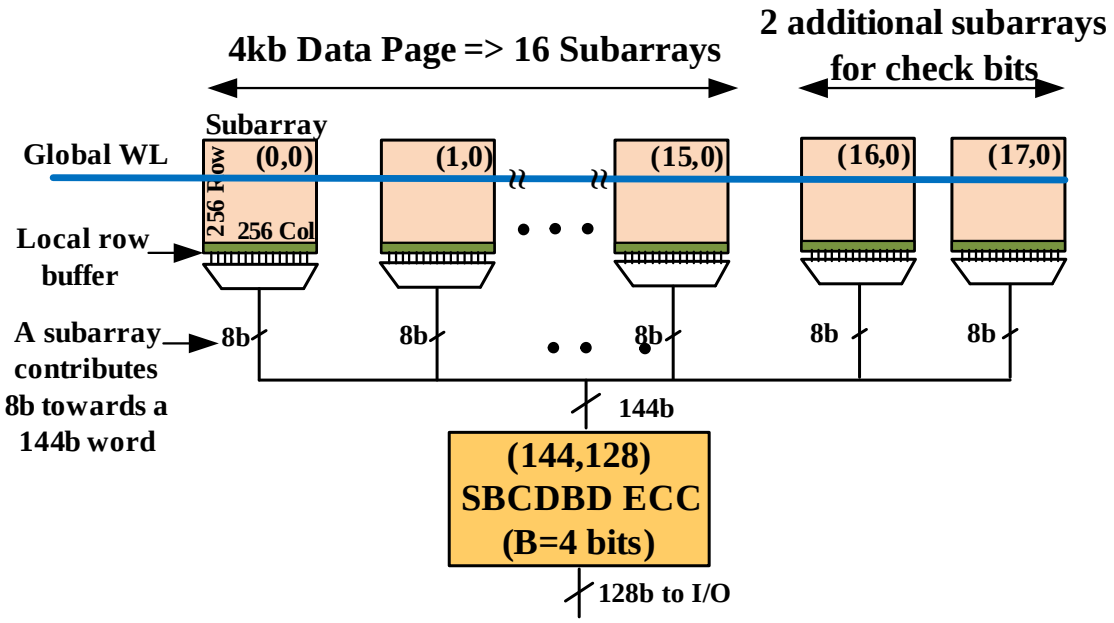


Figure 5.9: **SBCDBD-based Subarraykill error correction option I for 4kb page (information bits).**

Consider the 4kb page (information bits) as shown in Figure 5.9. The page is opened across 16+K subarrays (where K is the number of additional subarrays to store check bits) and the 144bit word is spread across this page with 32b spacing. This implies that each

subarray supplies 8 bits towards the word. Because the bits are spaced 32 bits apart, this ensures that a particle strike would not cause a burst error of greater than 2 bits. Using a (144,128) rotational SBCDBD (B=4b) code [70], as shown in Figure 5.9, will ensure that no more than one 'byte' is in error, and hence can be corrected. Thus we have reduced the check bit storage area overhead to 12.5% (versus 21.9% with SECDED).

While the scheme explained in Figure 5.9 protects against soft errors, it cannot guarantee correction in case of hard errors, such as whole subarray row failure, although it will detect them. This is because a subarray row failure would corrupt up to 8bits, or 2 bytes for B=4b. The (144,128) SBCDBD code with B=4b can detect errors spanning 2 bytes but cannot correct them.

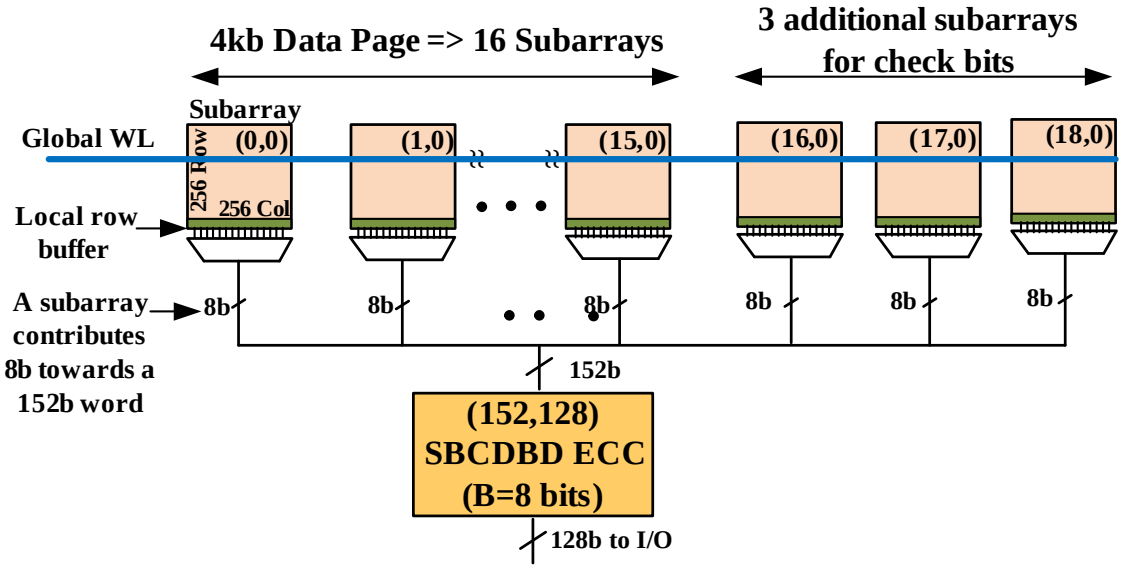


Figure 5.10: **SBCDBD-based Subarraykill error correction option II for 4kb page (information bits).**

If we need to guarantee correction in the case of subarray row failures, stronger SBCDBD codes are required at the cost of higher area, power and latency overheads. For example, we can modify the scheme from Figure 5.9 to Figure 5.10 and use a (152,128) SBCDBD (B=8b) ECC unit instead of a (144,128) SBCDBD (B=4b) ECC unit. The (152,128) SBCDBD (B=8b) ECC is a shortened RS(19,16) code over GF(2^8) with minimum distance 4 that is derived from RS (255,252) code over GF(2^8). Any subarray row failure will now affect at most 1 byte (8b) in an ECC word, and can be corrected. Table 5.2 compares SBCDBD

configurations for different page sizes with their SECDED counterparts that have the same error correction performance. For instance, for the 4kb page (information bits), the (152, 128) SBCDBD (B=8b) ECC has 18.75% storage overhead compared to the 37.5% for the SECDED configuration and can correct whole subarray failures. While the reduction in storage overhead is significant, in memory systems this may still be too large. In such cases, we propose the use of the (144,128) code which has a 12.5% storage overhead and guarantees soft error correction, but only hard error detection. These configurations have been labeled with a '*' in the table. We end our analysis with the 2kb page size, as with smaller pages this storage overhead becomes prohibitive, increasing die cost.

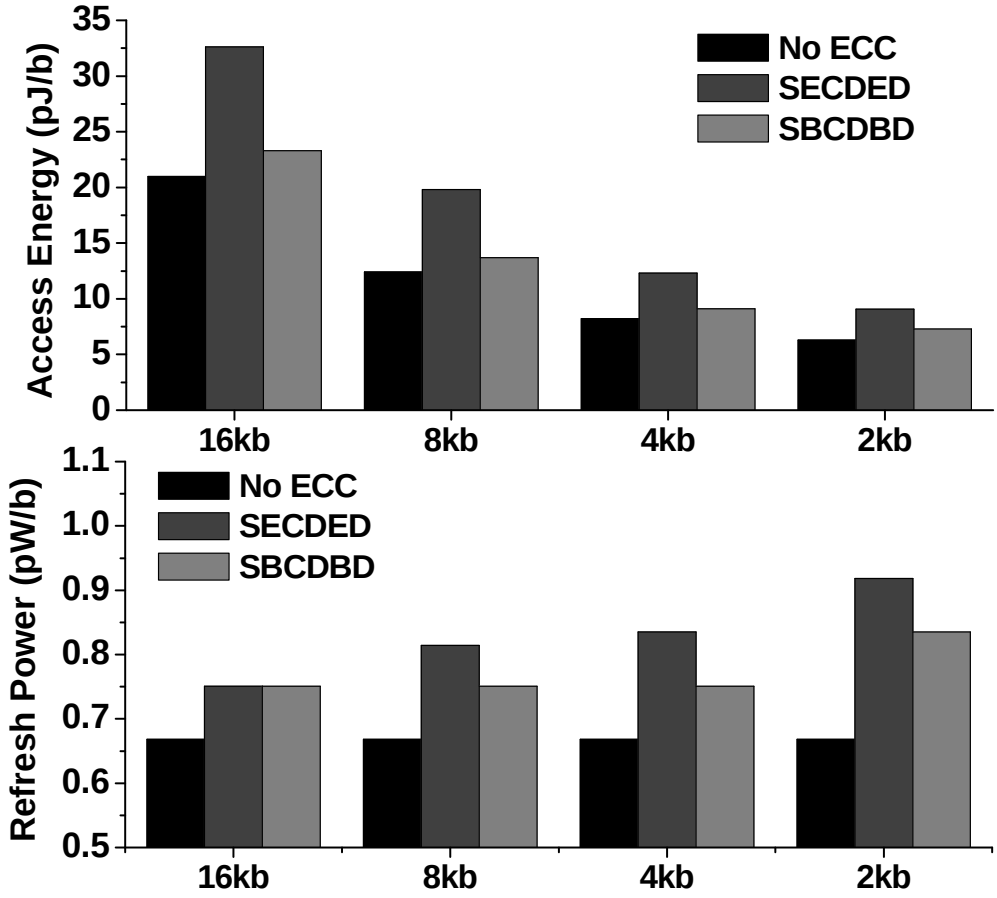


Figure 5.11: **Comparing the impact of error correction on access energy and refresh power across page sizes.**

Now consider the case when multiple subarrays have hard and soft errors. In other words, there are more than 2 bytes of errors in the code words that are being processed by the SBCDBD units. The mentioned SBCDBD codes have more than a 99% probability

of detecting 3 or 4 bytes of errors. In contrast, the Hsiao code, which is a popular (72, 64) SECDED code, has a 43% probability of triple error detection and more than a 99% probability of quadruple error detection [70]. Thus, the use of the SBCDBD codes allows us to detect more errors reliably and flag them for further processing by the OS. Even in the event of more severe faults (multiple subarrays, whole-chip, memory stack), we can still detect such failures using the same scheme by OR-ing the 'error-detect' signal [70] across subarrays.

5.4.3 Background Scrubbing

Typical cosmic particle strike rates are 0.005/(cm^2-h) [86]. Any particle strike can affect an area as much as 15μm in diameter (which is equivalent to 64 bits in a single row). Since our ECC scheme is designed to handle a single particle strike between scrubs, we are interested in a scrubbing rate which can keep the probability of two such strikes in close proximity fairly low. We choose an interval of 1 hour between same row scrubs, as this results in a fairly low probability of a double strike (Table 5.3) and a low power overhead (Figure 5.12). The scrubbing power slightly increases for smaller pages due to a corresponding small increase in CAS energy per bit (due to changes in muxing, error correction scheme).

Table 5.3: **Probability of double particle strikes per hour.**

Page size	Area sensitive to a particle strike	Prob. of double particle strikes per hour
16kb	13.44e-4	4.52e-11
8kb	6.72e-4	1.13e-11
4kb	3.36e-4	2.82e-12
2kb	1.68e-4	7.06e-13

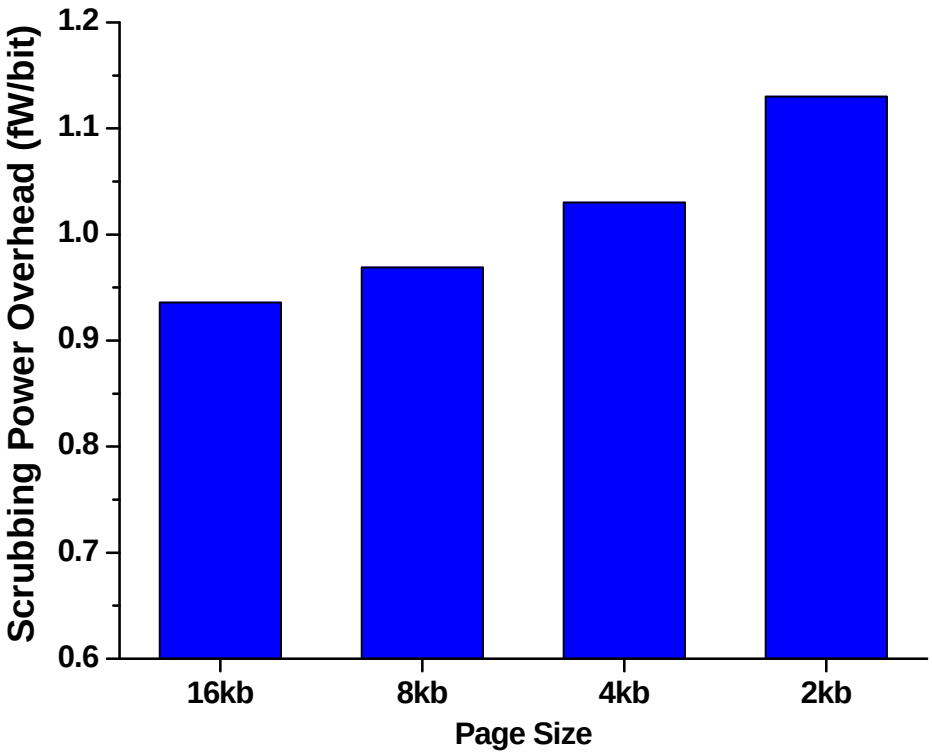


Figure 5.12: **Scrubbing power overhead as a function of page size. Refresh and ECC optimizations are in place.**

5.5 Evaluation Methodology

5.5.1 Power Model

The logic die model accounts for the power of the address decoders, sense amplifiers, row buffers and the low-swing I/O. The memory die model accounts for the power of bitlines and the global/local wordline drivers. All mixed-signal peripherals (sense amplifiers, precharge logic, etc.) were custom-designed in a 28nm industrial process and simulated in SPICE. All other logic layer components were synthesized in 28nm and the power was modeled using Synopsys Primetime©. All wire loads (including TSV routing capacitances) were estimated using the floorplan in Figure 5.3. The sense amplifiers (along with bitline columns) were simulated across process, voltage and temperature. We performed 1M Monte Carlo simulations to design for process mismatch. The subarray is in a 50nm DRAM process and the model was provided by Tezzaron. The subarray model included the capacitances of the bitcell, the bitline and column muxing, as well as bitcell leakage current for accurate power

Table 5.4: **NEK5000 benchmarks used for this study.**

Benchmark	Description	Threads
blasius	Boundary layer analysis	64
conj_ht	2D example of conjugate heat transfer	64
eddy	Navier-Stokes Equation solver	64
ext-cyl	Flow past a cylinder in 2D	64
lowMach_test	Chemically reactive flow	64
solid	Linear elasticity steady 3D solid solver	24
axi	Axisymmetric boundary case	8

modeling.

5.5.2 Performance Model

In order to obtain DRAM access patterns in large scale systems, we used main memory access traces from NEK5000 benchmarks [71]. These benchmarks characterize the applications expected in exascale machines. The benchmarks use a Message Passing Interface (MPI) [87] to communicate between cores. The individual benchmarks are described in Table 5.4. Since the total access latency t_{RCD} (RAS latency)+t_{RP} (Precharge latency)+t_{CL} (CAS latency) of the proposed 3D-stacked chip, including ECC enhancements, is much less than that of conventional DIMMs (13.6ns vs. 45ns for DDR3-1333 [83]), there is no negative impact on the total system runtime. Thus, the focus of our performance analysis is on the impact of row buffer locality on energy.

While it is impractical to evaluate an entire exascale system, we develop a methodology to simulate a single 32Gb 3D DRAM chip that is part of a blade unit in a larger exascale system. This is accomplished by running the benchmarks on a cluster with up to 64 cores per benchmark. Locality measurements are gathered on the access pattern for each benchmark. Additional traffic from other blades in the system would only serve to reduce the row buffer

Table 5.5: **Cache parameters.**

	L1	L2
Size	64KB	16MB
Block Size	64B	64B
Associativity	4	16
Replacement	LRU	LRU
Prefetch Policy	None	None
Write Back	Yes	Yes

hit rate and, as such, the results presented here are optimistic for the amount of locality that will be experienced in exascale systems.

The evaluation cluster consisted of twelve 6-core Intel Xeon X5670 CPU nodes providing a total of 72 total cores. Each node contained 24GB of RAM and is connected to the network using 10Gbps Ethernet. We instrumented all reads / writes of the benchmarks with PIN [88]. PIN traces all memory accesses and instructions on each of the cores. Each process filtered its memory accesses through an L1 cache simulator before writing L1 misses out to a merged trace file. This merged file was run through an L2 cache simulator to generate a memory trace. The cache parameters are presented in Table 5.5.

Finally, this memory trace was run through a memory simulator to yield cache line locality in the memory row buffers. The memory simulator has two memory controllers: one that issues requests in-order and one that issues requests out-of-order with a scheduling window of 10. Since the out-of-order scheduler does not have a concept of time between accesses, the simulator gives the best case performance for locality with a scheduling window of size 10. We assume a standard DDR channel width of 128 bits and every memory access results in 4 consecutive DRAM accesses (burst of four CAS instructions) to fill a cache line (64B). On a subsequent access, if the row buffers already contain the data correlating to that address, we have a hit and can avoid a RAS access. On a row buffer miss a RAS is required in addition to a CAS. We studied the percentage of main memory accesses that resulted in a RAS as a function of row buffer (DRAM page) size in order to understand the impact of page size on locality.

5.6 Results

The results section is broken into two parts. First we include the impact of benchmark locality on page size, and then we explore the total power consumption of our proposed design for a 100PB system, comparing it to conventional DIMMs and current 3D DRAM designs.

5.6.1 Locality

Figures 5.13 and 5.14 respectively show the percentage of cache misses that result in a DRAM RAS operation across several NEK5000 benchmarks with (a) in-order and (b) out-of-order (window size 10) memory scheduling. As the page size becomes smaller, it is less likely that a subsequent access will hit the row buffer (lower locality). Overall, the aggregate locality and its dependence on page size is at best moderate, because multiple cores interleave accesses to the DRAM, evicting each others' data from the row buffer. Even with optimistic memory scheduling, the trend barely changes, and at 2kb (information page size), almost 80% of DRAM cache line accesses result in a RAS operation.

The low locality of these benchmarks suggests using smaller pages to minimize energy. However, smaller pages have higher energy/area costs of error resilience. Taking locality and error resilience costs jointly into consideration, we minimize the energy of 3D DRAM chip. The storage-optimal ECC scheme for each page size is selected from the configurations in Table 5.2. The power/energy numbers were calculated with the methodology described in Section 5.5. Figure 5.15 shows the energy of DRAM accesses for each page size. The energy is broken down by RAS+PC energy and CAS energy that includes all ECC overheads, and low swing I/O. The fourth bar shows the total energy per bit without considering locality—all access result in a RAS+PC and a CAS. The locality of the benchmarks reduces the likelihood of an access requiring a RAS+PC, so the last bar shows a reduction resulting from the average locality across the benchmarks. For the 8kb page size, the locality helps to reduce the average access energy per bit from 13.7pJ to 5.5pJ. It is important to note that each cache line miss that results in a RAS operation performs 4 CAS operations to return the entire 64B cache line—this improves the locality by 4×. The total access energy of the

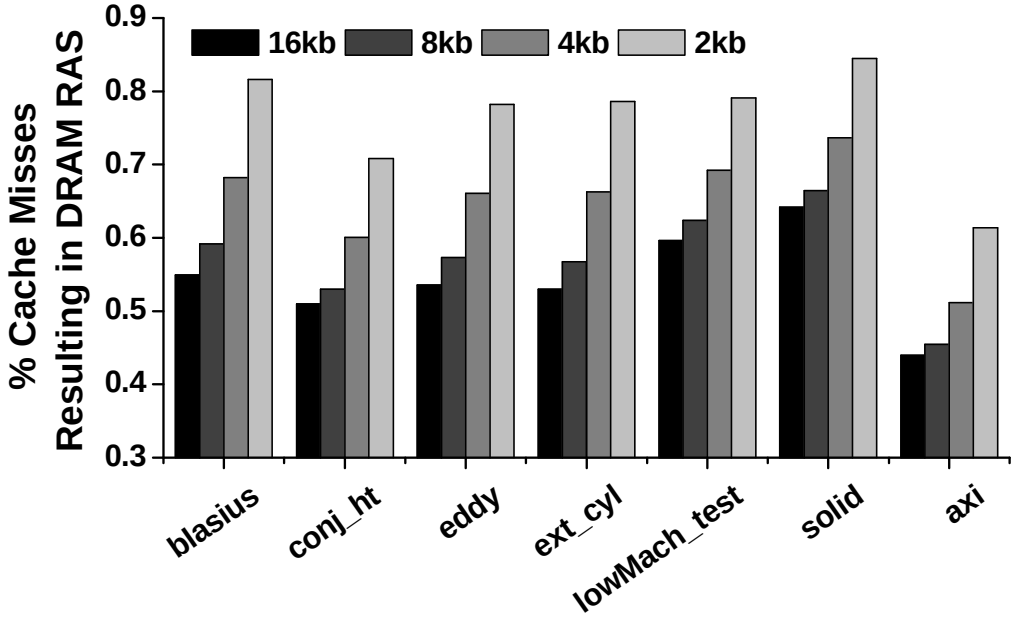


Figure 5.13: **Percentage of cache misses resulting in a DRAM RAS operation across NEK5000 benchmarks with in-order memory scheduling.**

system experiences a slight minima at the 4kb page size. Smaller page sizes incur slightly higher energy due to poor locality and additional area overhead for parity bit storage. The 4kb page has a low energy of 5.1pJ/bit with a storage overhead of 12.5% for ECC check-bits. The 12.5% overhead is quite common in today's ECC DIMMs [73]. Note that this particular implementation only guarantees hard error detection—if hard error correction is required, the 4kb page consumes 5.4pJ/bit with 18.75% storage overhead.

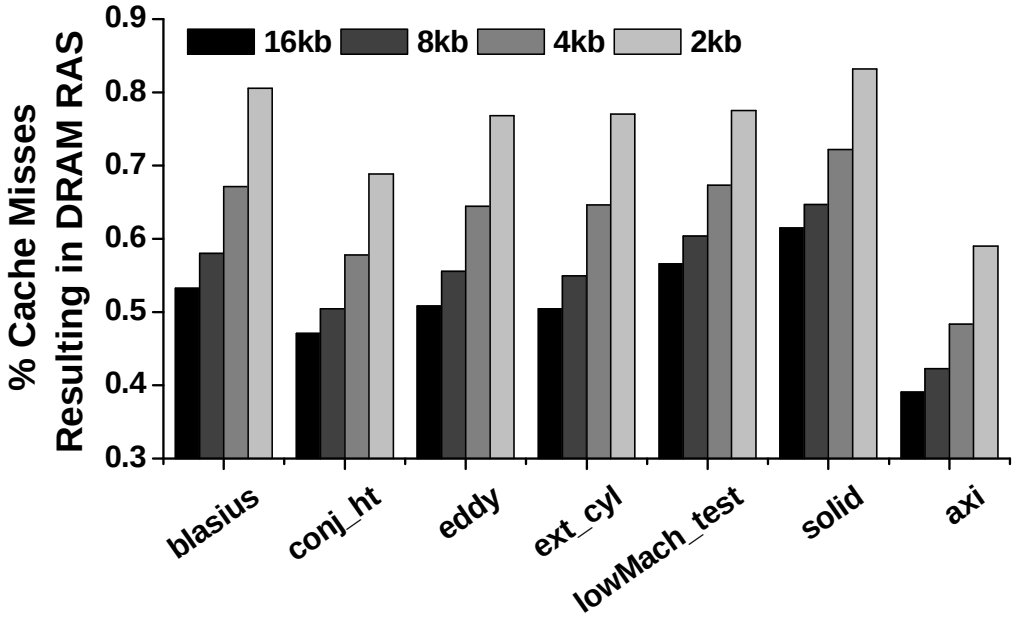


Figure 5.14: **Percentage of cache misses resulting in a DRAM RAS operation across NEK5000 benchmarks with optimistic out-of-order memory scheduling.**

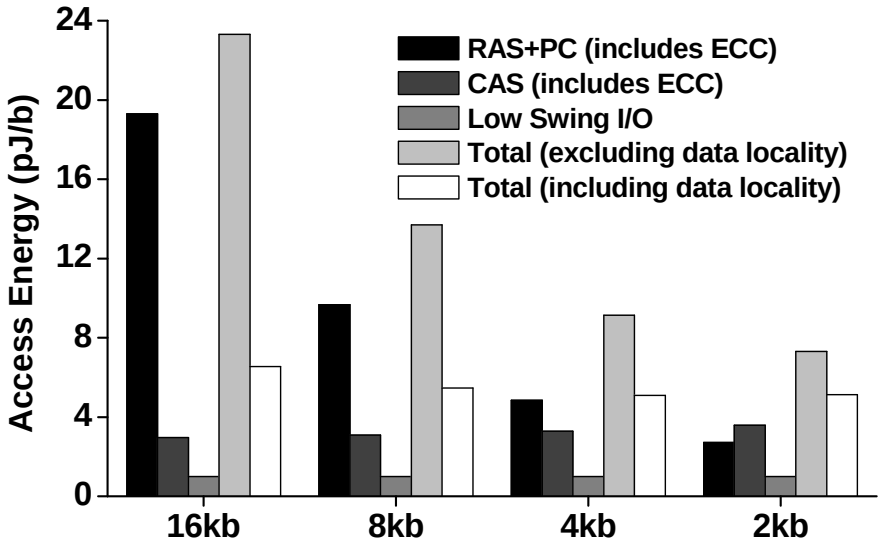


Figure 5.15: **DRAM access energy across page sizes.** This includes the energy cost of error correction using our proposed scheme.

5.6.2 Complete 100PB DRAM Power Analysis

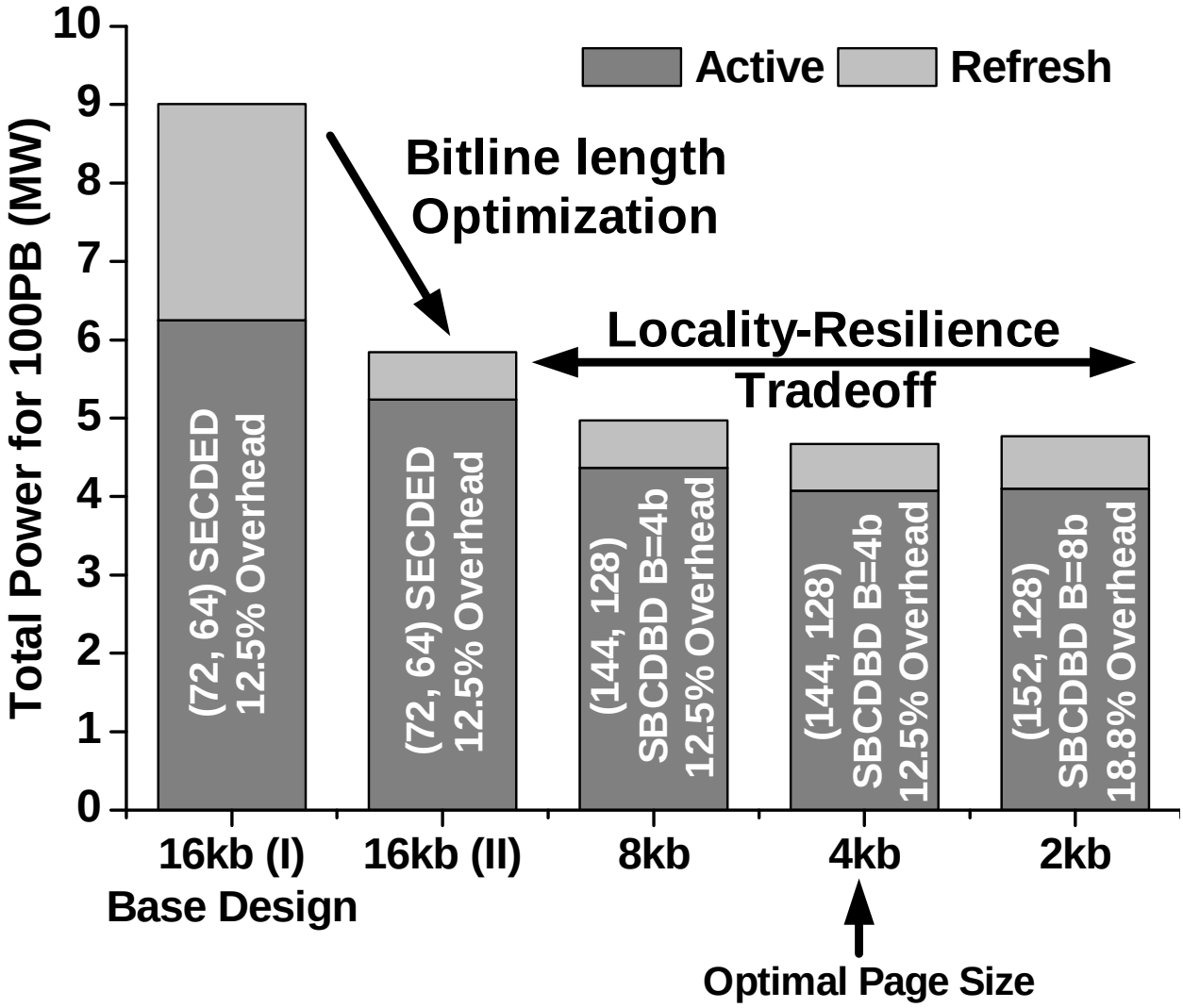


Figure 5.16: **Power consumption of 100PB DRAM constructed using 32Gb 3D-stacked chips.**

Taking into consideration the effect of locality on access energy, we calculate the total power of the exascale memory. To do this we scale the energy numbers to a 100PB memory with 100PB/s data bandwidth and include the power from refresh and error detection. Overall, we find that the total power of the DRAM system scales well to the 4kb page size (Figure 5.16) without any additional ECC storage overhead (Table 5.2). The total power of the DRAM in the exascale system is 4.7MW (with hard error detection). This increases to 4.9MW if hard error correction is required (with 18.75% storage overhead). An exascale

Table 5.6: **Comparison with current DIMM-based and 3D-stacked DRAM memories.**

DRAM Memory	**BW (GB/s)**	**Configuration**	**Power at 100PB/s**
DDR3-1333 DIMM [55]	10.66	5E7× 2GB	52MW
DDR4-2667 DIMM [55]	21.34	2.5E7× 4GB	31MW
LPDDR3 [89] (30nm)	6.4	2E8× 512MB	6.3MW
HMC I [55] (3D-stack, 50nm DRAM, 90nm logic)	128	2E8× 512MB	8.6MW
Wide I/O [81] (3D-stack, 50nm)	12.8	2E8× 512MB	2.6MW*
This work (3D-stack, 50nm DRAM, 28nm logic)	100	2.5E7× 4GB	4.7MW
*Assumes 100% locality in a 16kb page.			

system built with conventional DDR3 DIMMs would consume ∼52MW and one built with a 3D design such as the HMC would consume ∼8.8MW. Overall, our proposed design results in a 6.5× reduction in power compared to DDR4 DIMMs, and a 1.8× reduction compared to the first generation HMC. We compare our solution with current DIMM-based and 3D-stacked DRAM memories in Table 5.6.

5.7 Related Work

In this chapter we analyzed the implications of error correction, page resizing, and the subarray design itself on access and refresh energies to converge on a memory targeted for Exascale computing. Below we summarize various classes of related work.

DRAM Reorganization and Active Power: Several works have proposed reorganizing DRAM through rank sub-setting and subarray reorganization [62–64] and analyzed their impact on performance, power and reliability [90]. Our work focuses on reorganizing DRAM for an exascale system where we are constrained by stringent requirements on power, performance and reliability. Udipi et al. [62] reorganized the DRAM to enable single access filling of a 64B cache line. This approach reduced power consumption while trading off transfer times by having lower bandwidth. 3D-stacking enables us to have increased transfer bandwidth (through a large number of TSVs) and hence we did not have to make any major tradeoffs while scaling back on power. Research in photonic interconnects [91, 92] has demonstrated increased bandwidth which will further optimize 3D DRAM in the future. Work by Loh [60] on 3D DRAM architecture has mainly focused on design points involving multiple memory controllers and ranks; our reorganization is on a finer micro-architectural granularity. In [82], Kim et al. exploited parallelism in DRAM sub-arrays to reduce latency by overlapping memory accesses to different banks and reduce power by operating at a sub-array granularity. More recently, Weis et al. [93] also performed a design space exploration for 3D-stacked DRAM, where they co-optimized the memory and the controller architecture to minimize energy. However, both approaches have not considered implications on error resilience costs.

Refresh Power: Recently, there have been attempts to reduce refresh rates as low as possible [67, 68] without introducing errors. These proposals make the observation that process variations and temperature conditions result in each DRAM cell having a different refresh rate. Accordingly, they refresh different portions of the DRAM at different rates after profiling the impact of different sources of variation. In addition, the controller can smartly avoid refreshing inactive DRAM rows. Both of these techniques to reduce refresh power are orthogonal to the technique presented in this work and could be used to further reduce

refresh power in future DRAM organizations. However, they would incur more storage and computation overheads to track DRAM cell refresh rates or the inactive rows. Recently Kim et al. [94] also proposed tiering DRAM bitlines to reduce capacitance and improve latency. However, they have not considered its implications for refresh power savings.

Error Correction cost: Several works [65, 95] have evaluated the cost of error correction on the overall system in the context of soft errors. However, more recent studies [75, 76] have shown that DRAM failures in large scale systems are dominated by hard errors. We implement Subarraykill to protect against hard errors such as whole-subarray failures. In addition, large scale systems use a memory scrubber that periodically walks through the memory and corrects the data with an ECC mechanism [96].

CHAPTER 6

Conclusion and Future Directions

This chapter summarizes the contributions of this dissertation and and highlights future research directions motivated by the current work.

6.1 Summary

Improving performance efficiently, while addressing reliability has become a critical computing bottleneck in modern technologies. This dissertation presented circuit and architectural techniques for addressing these challenges targeted at datapath logic, signal synchronization and memories.

In chapter 2, a domino logic design style called Adaptive Robustness Tuning (ART) was presented. ART architecture, pipelining, clock generation, error detection and recovery were discussed. This technique was demonstrated in a $32b \times 32b$ multiplier fabricated in 65nm CMOS technology where it provided performance gains of up to 71% over conventional domino logic. The technique dynamically tunes domino gates to trade surplus noise margins at nominal conditions for performance by detecting stability errors during runtime while guaranteeing forward progress. It also eliminates timing margins.

In chapter 3, we described dynamic buffer based synchronizers, where metastability is only caused by pulses, rather than stable, intermediate voltages. This unique feature was exploited by amplifying such pulses using simple elements such as skewed inverters. The synchronizers were fabricated in 65nm CMOS and were shown to improve MTBF by $\sim 1 \times 10^6 \times$

($\sim 5\times10^7\times$) over jamb latches (double flip-flops) at 2GHz. The synchronizers provided single cycle synchronization with a MTBF of up to $\sim 2\times10^{11}$ years. A capacitance de-rating based technique to experimentally measure metastability in silicon was also presented and used to characterize synchronizer performance.

In chapter 4, a variation tolerant sensing (VTS) scheme targeting high performance 6T memories was presented. The scheme modified the conventional sensing topology to additionally perform auto-zeroing based offset compensation, and bitline droop pre-amplification using AC-coupling MOM capacitors. VTS circuit schematic, operation phases, array design and capacitor design were discussed. The scheme was implemented in 28nm CMOS, where sensing reliability was measured to be improved by $1.2\sigma_{Vth}$ without added area overhead. This increased robustness was in turn traded for performance, providing up to 42% sensing speed improvement and 10% lower sensing power at 1.8GHz. The characterization methodology to measure SA performance was also discussed.

In chapter 5, we showed that addressing reliability while meeting exascale power budgets is a co-optimization problem involving multiple aspects of memory design. Contrary to popular belief, we showed that an all-DRAM solution is feasible, provided the traditional interfaces are re-thought. We presented a resilient architecture for a main memory building block based on a Tezzaron prototype. It employs a 3D-stacked DRAM organization to meet the stringent power, bandwidth and reliability demands. We showed that minimizing power requires a tradeoff between row buffer size, refresh, and the ECC mechanisms. Using NEK5000 benchmarks, we showed that the optimal solution for the 32Gb 3D-stacked building block uses a 4kb page with (144,128) SBCDBD (B=4b) ECC scheme to protect against soft/hard errors. Scaling this design to 100PB would result in a memory power consumption of 4.7MW, which is well within the exascale power budget (20MW).

6.2 Future Directions

The work presented in this dissertation opens up many directions for future research. The work on ART domino logic presented in chapter 2 can be extended to include *error correction*, as already described in section 2.2.4. The current prototype implements error detection. Also, a feedback loop for robustness speculation to automatically dial in optimal TVDD/TVSS voltages to maximize performance gains based on a desired error rate, and generating these voltages on chip is another area in which this work can be extended. Additionally, the concept of ART can be extended to included other more power-efficient dynamic logic families, such as Limited Switch Dynamic Logic (LSDL) [97], that do not consume constant data power for a static input. Finally, the technique was implemented on a standalone multiplier —the work can be extended included the ART-based component into a larger system to study its system level implications.

The work on synchronizers can also be extended to implement them in a larger NoC system, rather as standalone components. As the synchronizers are dynamic, the static to dynamic and vice-versa signal conditioning in the context of these synchronizers can be studied in greater detail. In addition, skewed inverters are currently used as amplifying elements in the synchronizer —there is scope to explore other amplifying elements such as analog amplifiers and Schmitt triggers. Synchronizer performance (MTBF) can be also be studied using a different characterization method, where multiple copies of the synchronizer are run concurrently (without any de-rating) and failure statistics are recorded. The performance measured using this method can be compared with the results obtained using capacitance de-rating.

The work on VTS for 6T memories can be extended to include a feedback loop to automatically trade off improved robustness to increase sensing speed. This is similar to a future direction as also suggested in [98] to implement Razor for SRAM memories.

Finally, the work on DRAM memories for exascale computing can be extended to include the cost overhead of checkpointing. One approach can be to stack a layer of nonvolatile memory (NV, such as flash, STT-RAM, memristors, etc.) onto the DRAM stack for local checkpointing. Additionally, stronger error correction codes (such as erasure codes [99]) can

be deployed to leverage the known location of faulty bits to provide stronger error resilience for the remainder bits in the physical page.

BIBLIOGRAPHY

[1] FPGA Acceleration in HPC: A Case Study in Financial Analytics. Technical report, XtremeData Inc., Nov. 2006.

[2] Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38, Apr. 1965.

[3] TOP500 Supercomputing Sites. `http://www.top500.org/`.

[4] JJ Dongarra and AJ Van der Steen. High-performance computing systems: Status and outlook. *Acta Numerica*, 21(1):379–474, 2012.

[5] Weste Neil HE et al. *CMOS VLSI design: a circuits and systems perspective*. Pearson Education India, 2006.

[6] Samuel H Fuller, Lynette I Millett, et al. *The Future of Computing Performance: Game Over or Next Level?* National Academies Press, 2011.

[7] Kelin J Kuhn, Martin D Giles, David Becher, Pramod Kolar, Avner Kornfeld, Roza Kotlyar, Sean T Ma, Atul Maheshwari, and Sivakumar Mudanai. Process technology variation. *Electron Devices, IEEE Transactions on*, 58(8):2197–2208, 2011.

[8] Asen Asenov. Simulation of statistical variability in nano mosfets. In *VLSI Technology, 2007 IEEE Symposium on*, pages 86–87. IEEE, 2007.

[9] David A Fick. *Power, Interconnect, and Reliability Techniques for Large Scale Integrated Circuits*. PhD thesis, The University of Michigan, 2012.

[10] Shidhartha Das, David Roberts, Seokwoo Lee, Sanjay Pant, David Blaauw, Todd Austin, Krisztián Flautner, and Trevor Mudge. A self-tuning DVS processor using delay-error detection and correction. *Solid-State Circuits, IEEE Journal of*, 41(4):792–804, 2006.

[11] Shidhartha Das, Carlos Tokunaga, Sanjay Pant, W-H Ma, Sudherssen Kalaiselvan, Kevin Lai, David M Bull, and David T Blaauw. RazorII: In situ error detection and correction for PVT and SER tolerance. *Solid-State Circuits, IEEE Journal of*, 44(1):32–48, 2009.

[12] Keith A Bowman, James W Tschanz, Nam Sung Kim, Janice C Lee, Chris B Wilkerson, S-LL Lu, Tanay Karnik, and Vivek K De. Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. *Solid-State Circuits, IEEE Journal of*, 44(1):49–63, 2009.

[13] David Bull, Shidhartha Das, Karthik Shivashankar, Ganesh S Dasika, Krisztian Flautner, and David Blaauw. A power-efficient 32 bit ARM processor using timing-error detection and correction for transient-error tolerance and adaptation to PVT variation. *Solid-State Circuits, IEEE Journal of*, 46(1):18–31, 2011.

[14] Keith A Bowman, James W Tschanz, SL Lu, Paolo A Aseron, Muhammad M Khellah, Arijit Raychowdhury, Bibiche M Geuskens, Carlos Tokunaga, Chris B Wilkerson, Tanay Karnik, et al. A 45 nm resilient microprocessor core for dynamic variation tolerance. *Solid-State Circuits, IEEE Journal of*, 46(1):194–208, 2011.

[15] Thomas D Burd, Trevor A Pering, Anthony J Stratakos, and Robert W Brodersen. A dynamic voltage scaled microprocessor system. *Solid-State Circuits, IEEE Journal of*, 35(11):1571–1580, 2000.

[16] Kevin J Nowka, Gary D Carpenter, Eric W MacDonald, Hung C Ngo, Bishop C Brock, Koji I Ishii, Tuyet Y Nguyen, and Jeffrey L Burns. A 32-bit PowerPC system-on-a-chip with support for dynamic voltage scaling and dynamic frequency scaling. *Solid-State Circuits, IEEE Journal of*, 37(11):1441–1447, 2002.

[17] Masakatsu Nakai, Satoshi Akui, Katsunori Seno, Tetsumasa Meguro, Takahiro Seki, Tetsuo Kondo, Akihiko Hashiguchi, Hirokazu Kawahara, Kazuo Kumano, and Masayuki Shimura. Dynamic voltage and frequency management for a low-power embedded microprocessor. *Solid-State Circuits, IEEE Journal of*, 40(1):28–35, 2005.

[18] Alan Drake, R Senger, H Deogun, G Carpenter, S Ghiasi, T Nguyen, N James, M Floyd, and V Pokala. A distributed critical-path timing monitor for a 65nm high-performance microprocessor. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 398–399. IEEE, 2007.

[19] Kyoungho Woo, Scott Meninger, Thucydides Xanthopoulos, Ethan Crain, Dongwan Ha, and Donhee Ham. Dual-DLL-based CMOS all-digital temperature sensor for microprocessor thermal monitoring. In *Solid-State Circuits Conference-Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*, pages 68–69. IEEE, 2009.

[20] Jong Sung Lee, Kevin Skadron, and Sung Woo Chung. Predictive temperature-aware DVFS. *Computers, IEEE Transactions on*, 59(1):127–133, 2010.

[21] Nandish Mehta and Bharadwaj Amrutur. Dynamic supply and threshold voltage scaling for cmos digital circuits using in-situ power monitor. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, (99):1–10, 2011.

[22] Dan Ernst, Nam Sung Kim, Shidhartha Das, Sanjay Pant, Rajeev Rao, Toan Pham, Conrad Ziesler, David Blaauw, Todd Austin, Krisztian Flautner, et al. Razor: A low-power pipeline based on circuit-level timing speculation. In *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, pages 7–18. IEEE, 2003.

[23] David Blaauw, Sudherssen Kalaiselvan, Kevin Lai, Wei-Hsiang Ma, Sanjay Pant, Carlos Tokunaga, Shidhartha Das, and David Bull. Razor II: In situ error detection and correction for PVT and SER tolerance. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 400–622. IEEE, 2008.

[24] Jason Howard, Saurabh Dighe, Yatin Hoskote, Sriram Vangal, David Finan, Gregory Ruhl, David Jenkins, Howard Wilson, Nitin Borkar, Gerhard Schrom, et al. A 48-core IA-32 message-passing processor with DVFS in 45nm CMOS. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*, pages 108–109. IEEE, 2010.

[25] Saurabh Dighe, Sriram R Vangal, Paolo Aseron, Shasi Kumar, Tiju Jacob, Keith A Bowman, Jason Howard, James Tschanz, Vasantha Erraguntla, Nitin Borkar, et al. Within-die variation-aware dynamic-voltage-frequency-scaling with optimal core allocation and thread hopping for the 80-core teraflops processor. *Solid-State Circuits, IEEE Journal of*, 46(1):184–193, 2011.

[26] Salomon Beer, Ran Ginosar, Michael Priel, R Dobkin, and Avinoam Kolodny. The devolution of synchronizers. In *Asynchronous Circuits and Systems (ASYNC), 2010 IEEE Symposium on*, pages 94–103. IEEE, 2010.

[27] Ran Ginosar. Fourteen ways to fool your synchronizer. In *Asynchronous Circuits and Systems, 2003. Proceedings. Ninth International Symposium on*, pages 89–96. IEEE, 2003.

[28] Ran Ginosar. Metastability and synchronizers: A tutorial. *Design & Test of Computers, IEEE*, 28(5):23–35, 2011.

[29] A Shibayama, K Nose, Sunao Torii, M Mizuno, and M Edahiro. Skew-tolerant global synchronization based on periodically all-in-phase clocking for multi-core SOC platforms. In *VLSI Circuits, 2007 IEEE Symposium on*, pages 158–159. IEEE, 2007.

[30] Peter Caputa and C Stevenson. An on-chip delay-and skew-insensitive multicycle communication scheme. In *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pages 1765–1774. IEEE, 2006.

[31] Joseph Wang, Ping Liu, Yandong Gao, Pankaj Deshmukh, Sam Yang, Ying Chen, Wing Sy, Lixin Ge, Esin Terzioglu, Mohamed Abu-Rahma, et al. Non-gaussian distribution of sram read current and design impact to low power memory using voltage acceleration method. In *VLSI Technology (VLSIT), 2011 Symposium on*, pages 220–221. IEEE, 2011.

[32] Daniel Schinkel, Eisse Mensink, E Kiumperink, E Van Tuijl, and B Nauta. A double-tail latch-type voltage sense amplifier with 18ps setup+ hold time. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 314–605. IEEE, 2007.

[33] Naveen Verma and Anantha P Chandrakasan. A high-density 45 nm sram using small-signal non-strobed regenerative sensing. *Solid-State Circuits, IEEE Journal of*, 44(1):163–173, 2009.

[34] Masood Qazi, Kevin Stawiasz, Leland Chang, and Anantha P Chandrakasan. A 512kb 8t sram macro operating down to 0.57 v with an ac-coupled sense amplifier and embedded data-retention-voltage sensor in 45 nm soi cmos. *Solid-State Circuits, IEEE Journal of*, 46(1):85–96, 2011.

[35] Naveen Verma and Anantha P Chandrakasan. A 65nm 8t sub-vt sram employing sense-amplifier redundancy. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 328–606. IEEE, 2007.

[36] Sapumal Wijeratne, Nanda Siddaiah, Sanu Mathew, Mark Anders, Ram Krishnamurthy, Jeremy Anderson, Seung Hwang, Matthew Ernest, and Mark Nardin. A 9GHz 65nm Intel Pentium 4 processor integer execution core. In *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pages 353–365. IEEE, 2006.

[37] Gin Yee and Carl Sechen. Clock-delayed domino for adder and combinational logic design. In *Computer Design: VLSI in Computers and Processors, 1996. ICCD'96. Proceedings., 1996 IEEE International Conference on*, pages 332–337. IEEE, 1996.

[38] Rajesh Kumar and Glenn Hinton. A family of 45nm IA processors. In *Solid-State Circuits Conference-Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*, pages 58–59. IEEE, 2009.

[39] Bharan Giridhar, David Fick, Matthew Fojtik, Sudhir Satpathy, David Bull, Dennis Sylvester, and David Blaauw. Adaptive robustness tuning for high performance domino logic. In *VLSI Circuits (VLSIC), 2011 Symposium on*, pages 190–191. IEEE, 2011.

[40] James Tschanz, Keith Bowman, Shih-Lien Lu, Paolo Aseron, Muhammad Khellah, Arijit Raychowdhury, Bibiche Geuskens, Carlos Tokunaga, Chris Wilkerson, Tanay Karnik, et al. A 45nm resilient and adaptive microprocessor core for dynamic variation tolerance. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*, pages 282–283. IEEE, 2010.

[41] Shidhartha Das, David Roberts, Seokwoo Lee, Sanjay Pant, David Blaauw, Todd Austin, Krisztián Flautner, and Trevor Mudge. A self-tuning DVS processor using delay-error detection and correction. *Solid-State Circuits, IEEE Journal of*, 41(4):792–804, 2006.

[42] David Harris and Mark A Horowitz. Skew-tolerant domino circuits. *Solid-State Circuits, IEEE Journal of*, 32(11):1702–1711, 1997.

[43] Zhiyi Yu, Kaidi You, Ruijin Xiao, Heng Quan, Peng Ou, Yan Ying, Haofan Yang, Xiaoyang Zeng, et al. An 800MHz 320mW 16-core processor with message-passing and shared-memory inter-core communication mechanisms. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 64–66. IEEE, 2012.

[44] Bharan Giridhar, Matthew Fojtik, David Fick, Dennis Sylvester, and David Blaauw. Pulse amplification based dynamic synchronizers with metastability measurement using capacitance de-rating. In *Custom Integrated Circuits Conference (CICC), 2013 IEEE*, pages 1–4, 2013.

[45] David Fick, Nurrachman Liu, Zhiyoong Foo, Matthew Fojtik, Jae-sun Seo, Dennis Sylvester, and David Blaauw. In situ delay-slack monitor for high-performance processors using an all-digital self-calibrating 5ps resolution time-to-digital converter. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*, pages 188–189. IEEE, 2010.

[46] Jun Zhou, David J Kinniment, Charles E Dike, Gordon Russell, and Alexandre V Yakovlev. On-chip measurement of deep metastability in synchronizers. *Solid-State Circuits, IEEE Journal of*, 43(2):550–557, 2008.

[47] Bharan Giridhar, Nathaniel Pinckney, Dennis Sylvester, and David Blaauw. A reconfigurable sense amplifier with auto-zero calibration and pre-amplification in 28nm cmos. In *Solid-State Circuits Conference, 2014. ISSCC 2014. Digest of Technical Papers. IEEE International*, pages 1–2. IEEE, 2014.

[48] Kevin Zhang, Ken Hose, Vivek De, and Borys Senyk. The scaling of data sensing schemes for high speed cache design in sub-0.18 μm technologies. In *VLSI Circuits, 2000. Digest of Technical Papers. 2000 Symposium on*, pages 226–227. IEEE, 2000.

[49] Jeffrey Vetter. On the Road to Exascale: Lessons from Contemporary Scalable GPU Systems. In *Proceedings of the ATIP/A* CRC Workshop on Accelerator Technologies for High-Performance Computing: Does Asia Lead the Way?*, page 27. A* STAR Computational Resource Centre, 2012.

[50] Song Huang, Shucai Xiao, and W Feng. On the Energy Efficiency of Graphics Processing Units for Scientific Computing. In *Parallel and Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1–8. IEEE, 2009.

[51] Ronald G Dreslinski, Michael Wieckowski, David Blaauw, Dennis Sylvester, and Trevor Mudge. Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits. *Proceedings of the IEEE*, 98(2):253–266, 2010.

[52] Himanshu Kaul, Mark Anders, Steven Hsu, Amit Agarwal, Ram Krishnamurthy, and Shekhar Borkar. Near-Threshold Voltage (NTV) Design: Opportunities and Challenges. In *Proceedings of the 49th Annual Design Automation Conference*, pages 1153–1158. ACM, 2012.

[53] Vivek Sarkar, S Amarasinghe, D Campbell, et al. Exascale Software Study: Software Challenges in Extreme Scale Systems. *DARPA Information Processing Techniques Office, Washington DC*, 14:159, 2009.

[54] Alan Gara. Energy Efficiency Challenges for Exascale Computing. In *ACM/IEEE Conference on Supercomputing: Workshop on Power Efficiency and the Path to Exascale Computing*, 2008.

[55] J Thomas Pawlowski. Hybrid Memory Cube: Breakthrough DRAM Performance with a Fundamentally Re-Architected DRAM Subsystem. In *Proceedings of the 23rd Hot Chips Symposium*, 2011.

[56] Peter Kogge, Keren Bergman, Shekhar Borkar, Dan Campbell, W Carson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Kerry Hill, et al. ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems. 2008.

[57] Franck Cappello, Al Geist, Bill Gropp, Laxmikant Kale, Bill Kramer, and Marc Snir. Toward Exascale Resilience. *International Journal of High Performance Computing Applications*, 23(4):374–388, 2009.

[58] S Dumas. Mobile Memory Forum: LPDDR3 and WideIO. In *JEDEC Mobile Forum*, 2011.

[59] Sheng Li, Ke Chen, Ming-Yu Hsieh, Naveen Muralimanohar, Chad D Kersey, Jay B Brockman, Arun F Rodrigues, and Norman P Jouppi. System Implications of Memory Reliability in Exascale Computing. In *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for*, pages 1–12. IEEE, 2011.

[60] Gabriel H Loh. 3D-Stacked Memory Architectures for Multi-Core Processors. In *ACM SIGARCH Computer Architecture News*, volume 36, pages 453–464. IEEE Computer Society, 2008.

[61] Taeho Kgil, Ali Saidi, Nathan Binkert, Steve Reinhardt, Krisztian Flautner, and Trevor Mudge. PicoServer: Using 3D Stacking Technology To Build Energy Efficient Servers. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 4(4):16, 2008.

[62] Aniruddha N Udipi, Naveen Muralimanohar, Niladrish Chatterjee, Rajeev Balasubramonian, Al Davis, and Norman P Jouppi. Rethinking DRAM Design and Organization for Energy-Constrained Multi-Cores. In *ACM SIGARCH Computer Architecture News*, volume 38, pages 175–186. ACM, 2010.

[63] Hongzhong Zheng, Jiang Lin, Zhao Zhang, Eugene Gorbatov, Howard David, and Zhichun Zhu. Mini-Rank: Adaptive DRAM Architecture for Improving Memory Power Efficiency. In *Microarchitecture, 2008. MICRO-41. 2008 41st IEEE/ACM International Symposium on*, pages 210–221. IEEE, 2008.

[64] Jung Ho Ahn, Jacob Leverich, Robert S Schreiber, and Norman P Jouppi. Multicore DIMM: an Energy Efficient Memory Module with Independently Controlled DRAMs. *Computer Architecture Letters*, 8(1):5–8, 2009.

[65] Jung Ho Ahn, Norman P Jouppi, Christos Kozyrakis, Jacob Leverich, and Robert S Schreiber. Improving System Energy Efficiency with Memory Rank Subsetting. *ACM Transactions on Architecture and Code Optimization (TACO)*, 9(1):4, 2012.

[66] Bharan Giridhar, Michael Cieslak, Deepankar Duggal, Ronald Dreslinski, Hsing Min Chen, Robert Patti, Betina Hold, Chaitali Chakrabarti, Trevor Mudge, and David

Blaauw. Exploring dram organizations for energy-efficient and resilient exascale memories. In *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 23:1–23:12, New York, NY, USA, 2013. ACM.

[67] Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *Computer Architecture (ISCA), 2012 39th Annual International Symposium on*, pages 1–12. IEEE, 2012.

[68] Mrinmoy Ghosh and Hsien-Hsin S Lee. Smart Refresh: An Enhanced Memory Controller Design for Reducing Energy in Conventional and 3D Die-Stacked DRAMs. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 134–145. IEEE Computer Society, 2007.

[69] Timothy J Dell. A White Paper on the Benefits of Chipkill-Correct ECC for PC Server Main Memory. *IBM Microelectronics Division*, pages 1–23, 1997.

[70] Thammavarapu RN Rao and Eiji Fujiwara. Error-Control Coding for Computer Systems. *Prentice-Hall Inc.*, 1989.

[71] NEK5000. `http://nek5000.mcs.anl.gov/`.

[72] Micron. Technical note tn-41-01: Calculating memory system power for ddr3, 2007.

[73] Doe Hyun Yoon and Mattan Erez. Virtualized ECC: Flexible Reliability in Main Memory. *Micro, IEEE*, 31(1):11–19, 2011.

[74] Vilas Sridharan and Dean Liberty. A Study of DRAM Failures in the Field. In *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, pages 1–11. IEEE, 2012.

[75] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. DRAM Errors in the Wild: A Large-Scale Field Study. In *Proceedings of the 11th International Joint Conference on Measurements and Modeling of Computer Systems*, pages 193–204. ACM, 2009.

[76] Andy A Hwang, Ioan A Stefanovici, and Bianca Schroeder. Cosmic Rays Don't Strike Twice: Understanding the Nature of DRAM Errors and the Implications for System Design. In *Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 111–122. ACM, 2012.

[77] Dong Tang, Peter Carruthers, Zuheir Totari, and Michael W Shapiro. Assessment of the Effect of Memory Page Retirement on System RAS Against Hardware Faults. In *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, pages 365–370. IEEE, 2006.

[78] Masaaki Kondo. Report on Exascale Architecture Roadmap in Japan, IESP Meeting, 2012.

[79] Tezzaron Semiconductor. http://www.tezzaron.com/.

[80] Octopus 8-Port DRAM for Die-Stack Applications. http://www.tezzaron.com/.

[81] Jung-Sik Kim, Chi Sung Oh, Hocheol Lee, Donghyuk Lee, Hyong-Ryol Hwang, Sooman Hwang, Byongwook Na, Joungwook Moon, Jin-Guk Kim, Hanna Park, et al. A 1.2 V 12.8 GB/s 2Gb mobile Wide-I/O DRAM with 4× 128 I/Os using TSV-based stacking. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2011 IEEE International*, pages 496–498. IEEE, 2011.

[82] Yoongu Kim, Vivek Seshadri, Donghyuk Lee, Jamie Liu, and Onur Mutlu. A case for exploiting subarray-level parallelism (SALP) in DRAM. In *Computer Architecture (ISCA), 2012 39th Annual International Symposium on*, pages 368–379. IEEE, 2012.

[83] Micron 1Gb DDR3 SDRAM Datasheet. http://www.micron.com/.

[84] Koji Fukuda, Hiroki Yamashita, Goichi Ono, Ryo Nemoto, Eiichi Suzuki, Noboru Masuda, Takashi Takemoto, Fumio Yuki, and Tatsuya Saito. A 12.3-mW 12.5-Gb/s complete transceiver in 65-nm CMOS Process. *Solid-State Circuits, IEEE Journal of*, 45(12):2838–2849, 2010.

[85] Xin Li, Michael C Huang, Kai Shen, and Lingkun Chu. An empirical study of memory hardware errors in a server farm. In *The 3rd Workshop on Hot Topics in System Dependability (HotDep07)*, 2007.

[86] Ludger Borucki, Guenter Schindlbeck, and Charles Slayman. Comparison of accelerated DRAM soft error rates measured at component and system level. In *Reliability Physics Symposium, 2008. IRPS 2008. IEEE International*, pages 482–487. IEEE, 2008.

[87] Edgar Gabriel, Graham E Fagg, George Bosilca, Thara Angskun, Jack J Dongarra, Jeffrey M Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, et al. Open MPI: Goals, concept, and design of a next generation MPI implementation. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 97–104. Springer, 2004.

[88] Chi-Keung Luk, Robert Cohn, Robert Muth, Harish Patil, Artur Klauser, Geoff Lowney, Steven Wallace, Vijay Janapa Reddi, and Kim Hazelwood. Pin: building customized program analysis tools with dynamic instrumentation. In *ACM SIGPLAN Notices*, volume 40, pages 190–200. ACM, 2005.

[89] Yong-Cheol Bae, Joon-Young Park, Sang Jae Rhee, Seung Bum Ko, Yonggwon Jeong, Kwang-Sook Noh, Younghoon Son, Jaeyoun Youn, Yonggyu Chu, Hyunyoon Cho, et al. A 1.2 v 30nm 1.6 gb/s/pin 4gb lpddr3 sdram with input skew calibration and enhanced control scheme. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 44–46. IEEE, 2012.

[90] Jung Ho Ahn, Norman P Jouppi, Christos Kozyrakis, Jacob Leverich, and Robert S Schreiber. Future scaling of processor-memory interfaces. In *High Performance Computing Networking, Storage and Analysis, Proceedings of the Conference on*, pages 1–12. IEEE, 2009.

[91] Scott Beamer, Chen Sun, Yong-Jin Kwon, Ajay Joshi, Christopher Batten, Vladimir Stojanović, and Krste Asanović. Re-architecting DRAM memory systems with monolithically integrated silicon photonics. In *ACM SIGARCH Computer Architecture News*, volume 38, pages 129–140. ACM, 2010.

[92] Dana Vantrease, Robert Schreiber, Matteo Monchiero, Moray McLaren, Norman P Jouppi, Marco Fiorentino, Al Davis, Nathan Binkert, Raymond G Beausoleil, and Jung Ho Ahn. Corona: System implications of emerging nanophotonic technology. *ACM SIGARCH Computer Architecture News*, 36(3):153–164, 2008.

[93] Christian Weis, Igor Loi, Luca Benini, and Norbert Wehn. Exploration and optimization of 3-d integrated dram subsystems. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 32(4):597–610, 2013.

[94] Donghyuk Lee Yoongu Kim, Vivek Seshadri Jamie Liu, Lavanya Subramanian, and Onur Mutlu. Tiered-latency dram: A low latency and low cost dram architecture.

[95] Jangwoo Kim, Nikos Hardavellas, Ken Mai, Babak Falsafi, and James Hoe. Multi-bit error tolerant caches using two-dimensional error coding. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 197–209. IEEE Computer Society, 2007.

[96] Shubhendu S Mukherjee, Joel Emer, Tryggve Fossum, and Steven K Reinhardt. Cache scrubbing in microprocessors: Myth or necessity? In *Dependable Computing, 2004. Proceedings. 10th IEEE Pacific Rim International Symposium on*, pages 37–42. IEEE, 2004.

[97] Wendy Belluomini, Damir Jamsek, Andrew K Martin, Chandler McDowell, Robert K Montoye, Hung C Ngo, and Jun Sawada. Limited switch dynamic logic circuits for high-speed low-power circuit design. *IBM journal of research and development*, 50(2.3):277–286, 2006.

[98] Shidhartha Das. *Razor: A Variability-Tolerant Design Methodology for Low-power and Robust Computing*. ProQuest, 2009.

[99] Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A Spielman. Efficient erasure correcting codes. *Information Theory, IEEE Transactions on*, 47(2):569–584, 2001.