Better Than Worst Case Timing Design With Latch Buffers On Short Paths

by

Ravi Kanth Uppu

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 14, 2013

Keywords: Process Variation, Hold Latches, Clock Period Scaling, Critical Functional Flops, Shadow Flops

Approved by

Adit D.Singh, Chair, James B Davis Professor, Electrical and Computer Engineering
Vishwani Agrawal, James J Danaher Professor, Electrical and Computer Engineering
Victor P. Nelson, Professor and Assistant Chair, Electrical and Computer Engineering

Abstract

With continued advances in CMOS technology, parameter variations are emerging as a major design challenge. Irregularities during the fabrication of a microprocessor and variations of voltage and temperature during its operation make it increasingly difficult to meet aggressive performance targets under strict power budgets. Traditional adaptive techniques that compensate for Process Voltage Temperature (PVT) variations need safety margins and cannot respond to rapid environmental changes. In this thesis, we present a novel Better-than-worst-case design (BTWC) technique, which eliminates worst-case safety margins through in situ error detection of variation-induced delay errors. In our design, we use a delay-error tolerant flip-flop for every functional critical flop to scale the clock period to the point of first failure of a die under low power operations, which was the concept adopted from Razor[12][13]. Thus, all margins due to global and local PVT variations are eliminated, resulting in significant energy savings. In addition, the clock period can be scaled even lower than the first failure point into the sub-critical region, deliberately tolerating a targeted error rate, thereby providing additional energy savings. Thus, in the context of this design, a timing error is not a catastrophic system failure but a trade-off between the overhead of error-correction and the additional performance benifit due to Clock Frequency Scaling.

Earlier BTWC designs such as Razor [12][13] introduce shadow flip-flops triggered by a delayed clock in parallel to the functional flip-flops for timing error detection through duplication and comparision. This arrangement suffers from the "short path" problem, whereby the activation of paths shorter than this timing skew can cause false errors to be flagged. The traditional solution is to add buffers to the short paths that are less than the clock skew between the duplicated error detection flip-flops. However, this approach adds considerable area and power overhead, particularly in the presence of significant process

variations [18]. The proposed design studies the use of latches to introduce extra delay on short paths; holding short paths stable for the first phase of the clock allows the design to achieve a skew of half a clock period between the functional and shadow flip-flops without short path errors. We present a generic algorithm that characterizes all the path groups and places latches in appropriate path segments of the circuit to ensure that all short paths driving duplicated flip-flops are delayed by half a clock cycle. Unit delay simulations for benchmark designs with and without process variations are presented. Average performance improvement (API) and best-case performance improvement (BPI) for designs are presented with an overall average performance improvement of about 15% and best case performance improvement of 32% at a cost of acceptable area overhead.

Error correction for the above stated approach is taken care by an architectural replay mechanism. Furthermore, this design can be proven effective for detecting spurious transitions that are caused due to Single Event Upsets. However, this requires augmenting all the functional flip flops with shadow flops leading to an additional area overhead.

Acknowledgments

First and foremost I offer my sincerest gratitude to my supervisor, Dr Adit Singh, who has supported me thoughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. I attribute the level of my Masters degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. His insightful suggestions in understanding and approaching a subject has not only helped in completion of my thesis but also attributed in confronting challenges on a larger scale. I also wish to thank my advisory committee members, Dr. Vishwani Agrawal and Dr. Fa Foster Dai for their guidance and advice on this work.

My masters at Auburn University has helped me gain valuable practical experience. Courses like CAD tools, VLSI Testing, Low power design and Computer Architecture has not only delivered the required intuition for an hardware engineer but also helped in getting accustomed to several simulating tools, for which I am grateful to each and every professor.

Graduate study at Auburn University has been a learning experience. I thank my brother Ravi Tej for his constant support and colleagues Suraj and Praveen for their valuable inputs and patients. Without them my graduate experience would not have been this enriching.

I am indebted to my parents, brother, sister and all friends for their love and support. I also thank everyone who directly or indirectly helped and inspired me during the course of my graduate study.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advent of the deep submicron (DSM) era, more and more system on-chip (SoC) designs are being fabricated in sub-100nm technologies. Process, Voltage and Temperature variations are systematic and random variations in process characteristics across dies, supply voltage droop, and temperature variation. Rising PVT variations at advanced process nodes widen worst-case timing margins of the design–degrading performance significantly. These variations differ significantly in temporal and spatial scales. Process variations are static in nature, while voltage and temperature variations are highly sensitive to workload behavior, albeit at very different time scales. All sources of variation affect different parts of a microprocessor die in myriad ways with complex interactions and differ in their temporal and spatial characteristics. Microarchitectural techniques designed to mitigate parameter variations must clearly account for these differing characteristics.

At each semiconductor process node, the impacts on performance of environmental and semiconductor process variations become a larger portion of the cycle time of the product. With continued scaling of CMOS technology however, the numbers of relevant sources of variation and their magnitude have increased. In an attempt to account for this, additional static timing corners (additional clock period for the actual clock period) are being added to the design flow. As additional sources of variation become important, either the total guard-band applied during static timing is increased, or the risk of impacting yield is increased. This comes about due to the different delay sensitivities of each path on a design and the inability of the currently available design automation tools to handle the unique sensitivities of each path on the chip without running 2^n timing corners, where n is the number of independent variables of interest. Some paths are predominately sensitive to metal delay

while others are predominately sensitive to silicon (device) delay. Assuming that these paths will track the same from one manufacturing lot to another can lead to silicon that fails its timing requirements. Clearly a better method for dealing with variations is required. Accounting for all these variations, processors are designed for worst-case operating margins, losing substantial performance.

Although processors are traditionally designed to tolerate worst-case conditions, it is unlikely that all nonidealities will take effect at once, pushing a processor to the brink of erroneous behavior. Thus, there exists a considerable potential to increase the power efficiency or performance of processors by relaxing traditional, conservative requirements for correctness in the worst-case and instead designing processors for the average-case. Such better-than-worst-case designs work normally in the average case and have recovery mechanisms to ensure correct operation when errors occur. Design methodologies with reasonable overhead that can exploit the skewed behavior of the combinational profile under extreme variability, which is much more apparent in low power operations, would reap a significant amount of performance benefits.

1.1 Background

Over the past four decades, the number of transistors on a chip has increased exponentially in accordance with Moore‘s law [1]. This has led to progress in diversified computing applications, such as health care, education, security, and communications. A number of societal projections and industrial roadmaps are driven by the expectation that these rates of improvement will continue, but the impediments to growth are more formidable today than ever before. The largest of these barriers is related to energy and power dissipation, and it is not an exaggeration to state that developing energy-efficient solutions is critical to the survival of the semiconductor industry. Extensions of today‘s solutions can only go so far, and without improvements in energy efficiency, CMOS is in danger of running out of steam.

When we examine history, we readily see a pattern: generations of previous technologies, ranging from vacuum tubes to bipolar-based to NMOS-based technologies, were replaced by their successors when their energy overheads became prohibitive. However, there is no clear successor to CMOS today. The available alternatives are far from being commercially viable, and none has gained sufficient traction, or provided the economic justification for overthrowing the large investments made in the CMOS-based infrastructure. Therefore, there is a strong case supporting the position that solutions to the power conundrum must come from enhanced devices, design styles, and architectures, rather than a reliance on the promise of radically new technologies becoming commercially viable. In our view, the solution to this energy crisis is the universal application of aggressive low-voltage operation across all computation platforms. This can be accomplished by targeting so-called "near-threshold operation" and by proposing novel methods to overcome the barriers that have historically relegated ultralow-voltage operation to niche markets.

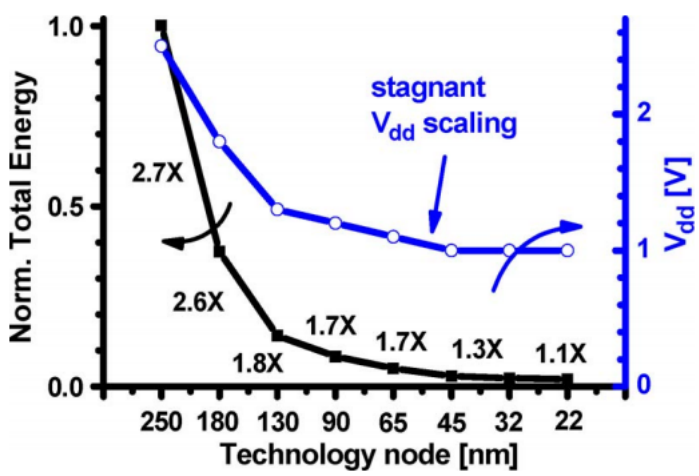


Figure 1.1: Technology scaling trends of supply voltage and energy

CMOS-based technologies have continued to march in the direction of miniaturization per Moore's law. New silicon-based technologies such as FinFET devices [2] and 3-D integration [3] provide a path to increasing transistor counts in a given footprint. However, using Moore's law as the metric of progress has become misleading since improvements in

packing densities no longer translate into proportionate increases in performance or energy efficiency. Starting around the 65 nm node, device scaling no longer delivers the energy gains that drove the semiconductor growth of the past several decades, as shown in Figure 1.1. The supply voltage has remained essentially constant since then and dynamic energy efficiency improvements have stagnated, while leakage currents continue to increase. Heat removal limits at the package level have further restricted more advanced integration. Together, such factors have created a curious design dilemma: *more gates can now fit on a die, but a growing fraction cannot actually be used due to strict power limits.*

At the same time, we are moving to a "more than Moore" world, with a wider diversity of applications than the microprocessor or ASICs of ten years ago. Tomorrow's design paradigm must enable designs catering to applications that span from high-performance processors and portable wireless applications, to sensor nodes and medical implants. Energy considerations are vital over this entire spectrum.

The aim of the designer in this era is to overcome the challenge of energy efficient computing and unleash performance from the reins of power to reenable Moore's law in the semiconductor industry.The use of ultralow-voltage operation, and in particular subthreshold operation ($V_{dd} < V_{th}$), was first proposed over three decades ago when the theoretical lower limit of Vdd was found to be 36 mV [4]. However, the challenges that arise from operating in this regime have kept subthreshold operation confined to a handful of minor markets, such as wristwatches and hearing aids. To the mainstream designer, ultralow-voltage design has remained little more than a fascinating concept with no practical relevance. However, given the current energy crisis in the semiconductor industry and stagnated voltage scaling we foresee the need for a radical paradigm shift where ultralow-voltage operation is applied across application platforms and forms the basis for renewed energy efficiency. Various design methodologies were proposed to achieve effective Dynamic Frequency or Voltage Scaling that include Adaptive Design Techniques, which tunes the system parameters (supply voltage and frequency of operation) during the dynamic operation of a processor, specific to the native

speed of each die and its run-time computational workload. The other class are the BTWC designs that not only overcome the additional margin that account for PVT variations but are also capable of running the clock at much aggressive rates under low power operations.

1.2 Motivation

For designs with aggressive voltage scaling, it is important to determine the V_{dd} at which the energy per operation (or instruction) is optimal. In the superthreshold regime ($V_{dd} > V_{th}$), energy is highly sensitive to V_{dd} due to the quadratic scaling of switching energy with V_{dd}. Hence voltage scaling down to the near-threshold regime ($V_{dd} \sim V_{th}$) yields an energy reduction on the order of 10X at the expense of approximately 10X performance degradation, as seen in Fig 1.2 [7]. However, the dependence of energy on V_{dd} becomes more complex as voltage is scaled below V_{th}. In subthreshold ($V_{dd} < V_{th}$), circuit delay increases exponentially with V_{dd}, causing leakage energy (the product of leakage current, V_{dd}, and delay) to increase in a near-exponential fashion. This rise in leakage energy eventually dominates any reduction in switching energy, creating an energy minimum seen in Figure 1.2.

The identification of an energy minimum led to interest in processors that operate at this energy optimal supply voltage [5],[6],[8] to the subthreshold regime, though delay rises by 50-100X over the same region. While acceptable in ultralow energy sensor-based systems, this delay penalty is not tolerable for a broader set of applications. Hence, although introduced roughly 30 years ago, ultralow-voltage design remains confined to a small set of markets with little or no impact on mainstream semiconductor products.

1.2.1 NTC Barriers

Although NTC provides excellent energy-frequency tradeoffs, it brings its own set of complications. NTC faces three key barriers that must be overcome for widespread use; performance loss, performance variation, and functional failure.

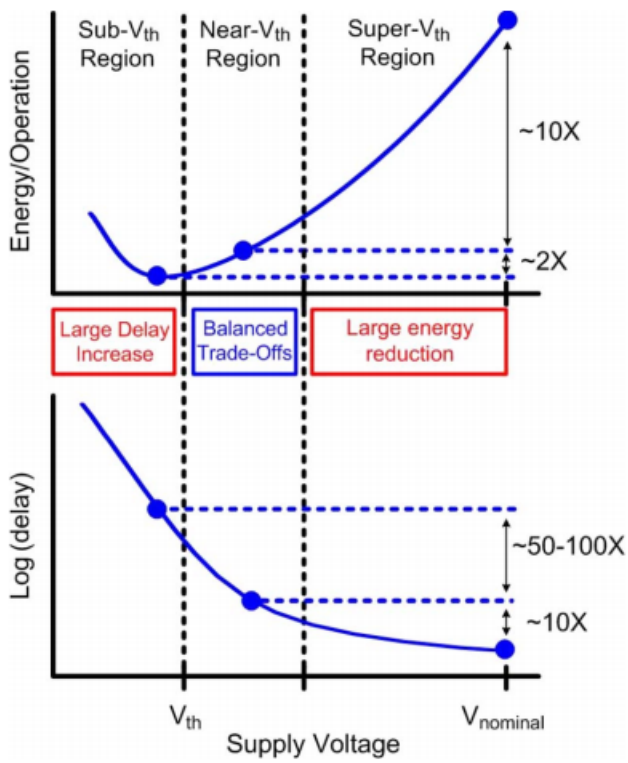


Figure 1.2: Energy and delay in different supply voltage operating regions.

A. Performance Loss

The performance loss observed in NTC, while not as severe as that in subthreshold operation, poses one of the most formidable challenges for NTC viability. In an industrial 45 nm technology the fanout-of-four inverter delay (FO4, a commonly used metric for the intrinsic speed of a semiconductor process technology) at an NTC supply of 400 mV is 10X slower than at the nominal 1.1 V. There have been several recent advances in architectural and circuit techniques that can regain some of this loss in performance. New technology optimizations that opportunistically leverage the significantly improved silicon wearout characteristics (e.g., oxide breakdown) observed in low voltage NTC can be used to regain a substantial portion of the lost performance.

B. Increased Performance Variation

In the near-threshold regime, the dependencies of MOSFET drive current on Vth, Vdd, and temperature approach exponential. As a result, NTC designs display a dramatic increase

in performance uncertainty. Figure 1.3 shows that performance variation due to global process variation alone increases by approximately 5X from ∼30% (1.3X) [9] at nominal operating voltage to as much as 400%, (5X) at 400 mV. Operating at this voltage also heightens sensitivity to temperature and supply ripple, each of which can add another factor of 2X to the performance variation resulting in a total performance uncertainty of 20X. Compared to a total performance uncertainty of ∼1.5X at nominal voltage, the increased performance uncertainty of NTC circuits looms as a daunting challenge that has caused most designers to pass over low-voltage design entirely. Simply adding margin so that all chips will meet the needed performance specification in the worst case is effective in nominal voltage design. In NTC design this approach results in some chips running at 1/10th their potential performance, which is wasteful both in performance and in energy due to leakage currents. The proposed BTWC design presents a new architectural approach to dynamically adapting the performance of a design to the intrinsic and environmental conditions of process, voltage, and temperature that is capable of tracking over the wide performance range observed in NTC operation. This method is complemented by circuit-level techniques for diminishing the variation of NTC circuits and for efficient adaptation of performance.

C. Increased Functional Failure

The increased sensitivity of NTC circuits to variations in process, temperature and voltage not only impacts performance but also circuit functionality. In particular, the mismatch in device strength due to local process variations from such phenomena as random dopant fluctuations (RDF) and line edge roughness (LER) can compromise state holding elements based on positive feedback loops. Mismatch in the loop‘s elements will cause it to develop a natural inclination for one state over the other, a characteristic that can lead to hard functional failure or soft timing failure. This issue has been most pronounced in SRAM where high yield requirements and the use of aggressively sized devices result in prohibitive sensitivity to local variation. As our proposed design is confined to datapath units in a processor, the above discussed barrier can be neglected. However, we can make the design

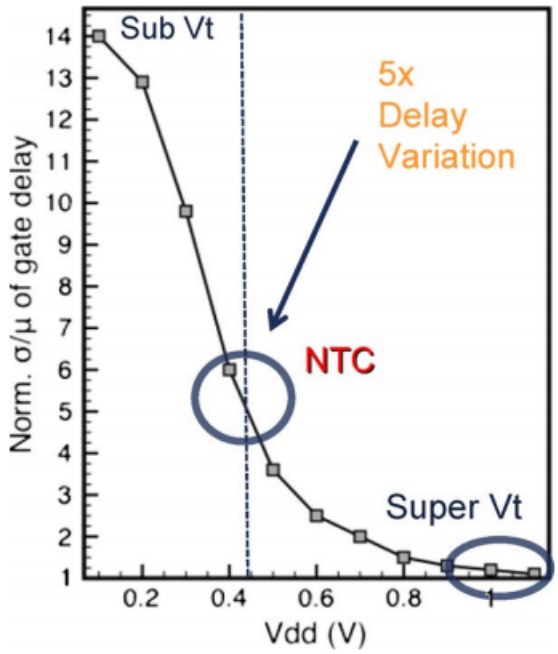


Figure 1.3: Impact of voltage scaling on gate delay variation.

robust to SEU's by augmenting every functional flop with a shadow flop, which can detect any Soft Errors with a penalty of increased hardware.

1.3 Problem Statement

From the above discussion, it is imminent that innovative design methodologies that not only overcome the timing margin in clocked sequential synchronous designs but also capable of aggressive frequency scaling operated at Near Threshold Voltage with reasonable hardware overhead are in great demand. One such design is RAZOR[12][13], which has its own limitations due to which the innate benefits of the design could not be exploited. In this thesis we try to overcome those limitations by using a novel hold mechanism at key internal nodes in a combinational profile. The following are the features of the proposed design.

1) Exploiting the skewed behavior of the combinational profile in a sequential design, under extreme timing variability (apparent in Low Power Operations).

2) Increased timing margin for the shadow flops ($\Delta T = T_{period}/2$), allowing aggressive frequency scaling under the face of extreme variability.

3) Reasonable hardware overhead, as dynamic latches (Tristate Buffers) are used to buffer the short paths, which effectively eliminate the short path violation without being susceptible to process variation.

4) Increase in the clock frequency, close to 20%, excluding the timing margins.

5) Effective for both the designs, with inherent skewed and balanced paths delays in a combinational profile.

1.4 Contribution of This Thesis

A novel design methodology was proposed that allows aggressive clock frequency scaling with a minimal area overhead and was analyzed for multiple circuits with and without timing variation. Unit delays were assigned to the gates in the design by characterizing the delay variation in an inverter for a 45nm process node, at multiple voltage corners with the threshold voltage varying from 0.1V to 0.9V. The proposed design was implemented in both combinational and sequential designs with inherent skewed and balanced path groups.

A generic algorithm was implemented, that characterizes all the path groups and places latches in appropriate path segments of the circuit to ensure that all short paths driving duplicated flip-flops are delayed by half a clock cycle. Unit delay simulations for benchmark designs with and without process variations are presented. Average performance improvement (API) and best-case performance improvement (BPI) for designs are presented with an overall average performance improvement of about 15% and best case performance improvement of 32% at a cost of acceptable area overhead. The key contribution of this thesis is achieving a timing window of $T_{period}/2$ by augmenting a negative edge triggered flop, which allows the clock to scale close to 33% of the critical path without including the PVT margin.

1.5 Organization of Thesis

The thesis is organized as follows. In Chapter 2, we discuss a general background of the earlier Better than Worst Case Design techniques, their pros and cons. Chapter 3 presents a discussion on short path constraints and overcoming them using latches at appropriate segment delays. Chapter 4 presents the latch placement algorithm and constraints for the latch placement for both the designs with and without timing variability. In Chapter 5, the proposed BTWC design is emphasized using an 8-bit Ripple Carry Adder. A design flow approach of the proposed BTWC design for a generic sequential circuit is discussed in Chapter 6. Simulation results with Conclusions and future work are discussed in Chapter 7.

Chapter 2

Literature Survey

This chapter presents a brief study of the present and past research work on better than worst case delay designs. Though BTWC designs [10-17] were implemented earlier either for performance improvement or for power aware computation the design has to make sure that a circuit operates without any errors even in the worst case scenario. Thus, the design of an efficient error resilient logic and a mechanism which can recover from a timing critical error can be thought of as a metric for the designs based on BTWC delay. The completion detection and correction in this work is novel, and can be implemented on designs whose delay distribution is skewed. Also, it is important to keep in mind that power and performance are compromising design metrics i.e. one needs to trade power for performance or vice versa.

2.1 Requirement for BTWC designs

After decades of astonishing improvement in integrated circuit performance, digital circuits have come to a point at which there are many problems ahead of us. Two main problems are power consumption and process variations.

In the past few decades, circuit designs have followed Moore‘s law and the number of transistors on a chip has doubled every two years. As we fit more transistors into a given area and clock them faster and faster, power density increases exponentially. The most effective way to reduce power density is to minimize the supply voltage, as predicted by CV^2f. Currently, we have been successful in containing the power density within tolerable levels, but this will not last. One barrier comes from the threshold voltage. In order to maintain the same performance, we have to reduce the threshold voltage together with the

supply voltage. However, reducing threshold voltage leads to an exponential increase in off-state leakage current. Leakage current has become so significant that further reductions in threshold voltage and supply voltage have slowed or even stopped. Without voltage scaling, the power density of a chip will increase without bound. If a solution to this ever increasing power consumption cannot be found, Moore‘s law will no longer apply and we will no longer be able to experience the tremendous increase in performance experienced in the past.

The other problem of the IC industry is process variations [17,18]. As transistor sizes approach atomic levels, it is very difficult to fabricate exactly what we specify. For example, a variation of dopant implantation on the order of a few atoms may translate to a huge difference in dopant concentration, and may cause a noticeable shift in the threshold voltage. Because traditional designs dictate that our circuit must always function correctly in all circumstances, the huge process variations present today force designers to allocate more voltage margin on top of the typical case in order to ensure proper operation. To make things worse, other temperature, die-to-die, and IR drop variations further increase the safety margins needed. The general practice of conservative overdesign has become a barrier to low power design. Because these large margins are used only to prevent the rare worst case scenario from failing, a large amount of energy can be saved if these margins are eliminated and instead utilize an error-resilient logic that can dynamically tune itself for all kinds of variations. We will then be able to run our chip at the lowest possible energy consumption or highest possible performance .

Power consumption and variations have become two of the most important roadblocks for extending Moores law and these problems must be addressed before we can continue to improve the performance of electronics at the amazing pace we enjoyed in the past decades. Thus even a minor improvement in performance without trading off power shall be a significant contribution in the face of the above mentioned bottlenecks of digital circuits.

2.1.1 Previous BTWC designs

A number of better-than-worst case (BTWC) designs have been proposed in the past to allow circuits to save power by operating under normal conditions rather than conservative worst case limits. One class of BTWC techniques specifies multiple safe voltage and frequency levels that a design may operate at and allows for switching between these levels. Examples in this class are correlating VCO (Voltage Controlled Oscillator) [28] [29] and Design-Time DVS (Dynamic Voltage Scaling) [32]. As voltage changes, correlating VCO adapts the frequency of a circuit to a level slightly below the critical frequency. The clock frequency for a given voltage is selected to incorporate a margin for process and temperature variations, as well as noise in the power supply network. Thus, scaling past the critical point is not allowed.

Similarly, Design-Time DVS provides the capability to switch between multiple voltage / frequency operating points to match user or application requirements. As with correlating VCO, each operating point incorporates a conservative margin to ensure that errors do not occur.Another class of BTWC designs uses canary circuits to detect when arrival at the critical point is imminent, thus revealing the extent of safe scaling. Delay line speed detectors [30] work by propagating a signal transition down a path that is slightly longer than the critical path of a circuit. Scaling is allowed to proceed until the point where the transition no longer reaches the end of the delay line before the clock period expires. While this circuit enables scaling, no scaling is allowed past the critical path delay plus a safety margin.

Another similar circuit technique uses multiple latches which strobe a signal in close succession to locate the critical operating point of a design. The third latch of a triple latch monitor [31] is always assumed to capture the correct value, while the first two latches indicate how close the current operating point is to the critical point. A design is considered to be tuned when the values in the first two latches do not match but the values in last two latches do match, indicating that the setup time of the third latch is longer than the critical delay of the circuit by a small margin.

All the BTWC techniques mentioned above have similar limitations. They allow for scaling up to, but never beyond,the critical operating point. However, with increasing variability in circuits, there is also high potential for benefit (e.g. in terms of power) when scaling is allowed to proceed past the critical point. Razor [12,13,14] actually allows voltage scaling past the critical point, since it incorporates error detection and correction mechanisms to handle the case when errors occur. While CRISTA [6] allows aggressive voltage scaling by Isolating and predicting the set of possible paths that may become critical under process variation and ensure that they are activated rarely, it avoids possible delay failures in the critical paths by dynamically switching to two-cycle operation.

On the other hand CSCD [10] provides carry completion signaling in low cost ripple carry adders which allows the control logic to schedule the next addition as soon as an earlier one is complete, thereby achieving the average case, rather than worst case addition delay over a set of computations.

While all the above mentioned designs can be considered as BTWC, only Razor, CRISTA and CSCD fall under a less conservative design as each design has unique way to sense the timing paths completion, and the challenge is to design a least conservation design technique. Instead of applying this approach to general purpose logic, as attempted by the Razor team and others, we have explored its use in specific widely used computations such as addition and multiplication.

2.2 Razor

The Razor approach, proposed in [12], aims at reducing the power consumption by minimizing the PVT timing margin to zero and beyond (since timing critical paths are not activated in every cycle), by building in a system capability to detect occasional errors due to slow signal paths and recover from them. The timing margins are removed by reducing the supply voltage to slow down the circuit to a point where a small, acceptable number of errors are observed. As long as the power saving from the reduced voltage operation exceeds the

extra power needed, on average, by the occasional error detection and recovery cycle, such a scheme can provide a net power saving. The challenge clearly is in designing an efficient low cost error detection and recovery capability to support this approach. The original Razor design [12] has changed and evolved [13,14] significantly over time in an attempt to achieve practicality. Even so the potential power savings from eliminating timing margins appear limited.

2.2.1 Razor I Overview

The key concept in Razor I [12] is to sample the input data of the flip-flop at two different points in time. The earlier, speculative sample is stored in a conventional positive-edge triggered, master-slave flip-flop. This main flip-flop is augmented with a so-called shadow latch which samples at the negative level of the clock. Razor I implementation shown in Figure 2.1.

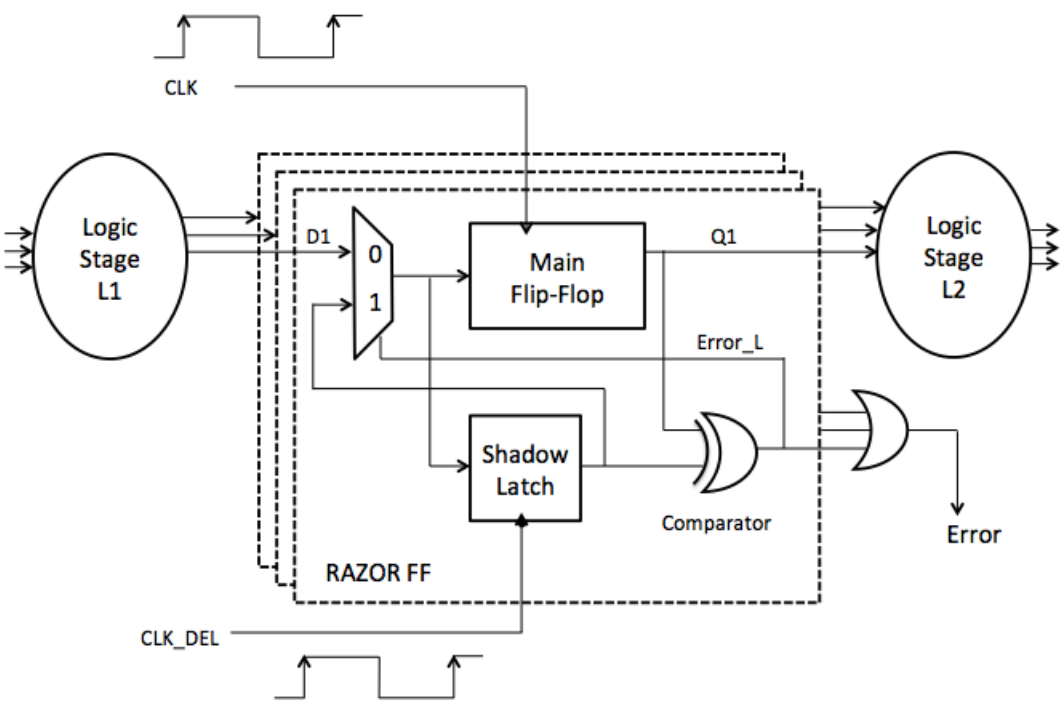


Figure 2.1: Razor flop-flop

Thus, the shadow latch gets additional time equal to the high phase of the clock to capture the correct state of the data. An error is flagged when data captured at the main flip-flop differs from the shadow-latch data. As the setup and hold constraints for the main flip-flop are allowed to be violated, an additional detector is required to flag the occurrences of metastability at the output of the main flip-flop. The error-pins of individual RazorI flip-flops are then OR-ed together to generate a pipeline restore signal which overwrites the correct data in the shadow latch into the main flip-flop, at the next positive edge of the clock. Since the shadow latch data is used to overwrite the state of the main flip-flop, it is required to ensure, using conventional worst-case techniques, that the data in the shadow latch is always correct.

There are key design issues that complicate the deployment of RazorI in high-performance, aggressively-clocked microprocessors. The primary difficulty is the generation and propagation of the pipeline restore signal. The restore signal is evaluated at the output of a high fan-in OR-tree and is suitably buffered and routed to every flip-flop in the pipeline stage before the next rising edge of the clock. This imposes significant timing constraints on the restore signal and the error recovery path can itself become critical when the supply voltage is scaled. This limits the voltage headroom available for Razor, especially in aggressively clocked designs. The design of the metastability detector is also difficult under rising process variations as it is required to respond to metastable flip-flop outputs across all process, voltage and temperature corners. Consequently, it requires the use of larger devices which adversely impacts the area and power overhead of the RazorI flip-flop. There is the additional risk of metastability at the restore signal which can propagate to the pipeline control logic, potentially leading to system failure.

2.2.2 RazorII

Razor II [13], Figure 2.2, was implemented to effectively address the design and timing issues in RazorI which moves the responsibility of recovery entirely to the micro-architectural

domain. The RazorII approach introduces two novel components which are described in the following paragraphs.

Instead of performing both error detection and correction in the flip-flop, RazorII performs only detection in the flip-flop, while correction is performed through architectural replay. This allows significant reduction in the complexity and size of the Razor flip-flop. although at the cost of increased IPC penalty during recovery. Architectural replay is a conventional technique which often already exists in high-performance microprocessors to support speculative operation such as out-of-order execution and branch prediction. Hence, it is possible to overload the existing framework to support replay in the event of timing errors. In addition, this technique precludes the need for a pipeline restore signal, thereby significantly relaxing the timing constraints on the error recovery path. This feature makes RazorII highly amenable to deployment in high-performance processors.

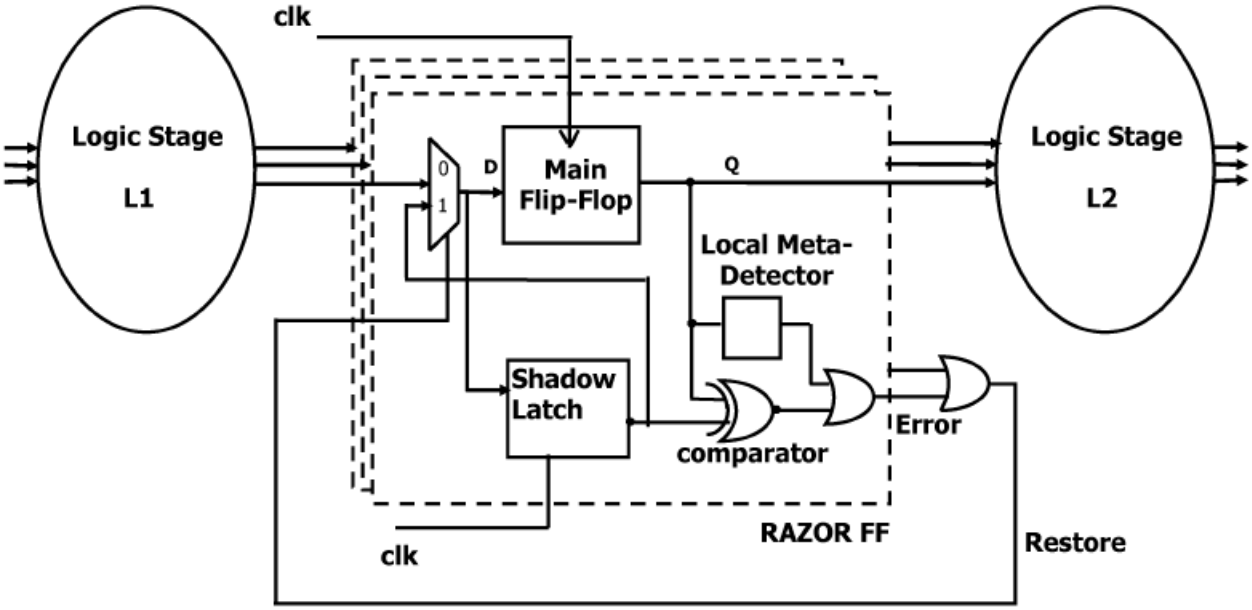


Figure 2.2: Pipeline augmented with Razor latches and control lines

Besides, the design of the RazorII flip-flop uses a positive level-sensitive latch instead of a master-slave flip-flop. The flip-flop operation is enforced by flagging any transition on the input data in the positive clock phase as a timing error. Elimination of the master latch

significantly reduces the clock pin capacitance of the flip-flop, bringing down its power and area overhead.

2.2.3 Razor Limitations

Designs such as Razor that allow scaling past the critical operating point [25] must be mindful of two aspects of error recovery - error detection and correction. Razor detects an error when the value latched by the shadow latch differs from the value latched by the main flip-flop. This happens when the logic signal has not settled to its final value before the setup time of the main flip-flop. If the signal transitions again before the shadow latch latches, an error will be detected.

For error correction, the Razor flip-flop must not only detect a timing violation, but must also latch the correct value in the shadow latch. This simply implies that the correct value must arrive by the setup time of the shadow latch for all Razor flip-flops in a design. So, Razor may not be able to correct errors if a) detection fails (i.e., both the main flip-flop and the shadow latch have the same incorrect value), or b) detection succeeds, but the value latched in the shadow latch is not the correct value.

To guarantee correctness, Razor requires two conditions to be met on the circuit delay behaviour which are, the short path constraint and the long path constraint. The long path constraint (eqn. 2.1), states that the maximum delay through a logic stage protected by Razor must be less than the clock period (T) plus the skew between the two clocks (the clock for the main flip-flop and the clock for the shadow latch).

$$delay < T + skew \tag{2.1}$$

If the long path constraint is not satisfied, false negative detections can occur when a timing violation causes both the main flip-flop and shadow latch to latch the incorrect value. The short path constraint (eqn. 2.2) states that there must not be a short path through a

logic stage protected by Razor that can cause the output of the logic to change before the shadow latch latches the previous output.

$$delay_{min} > skew + hold \tag{2.2}$$

Failure to satisfy the short path constraint leads to false positive error detections when the logic output changes in response to new circuit inputs before the shadow latch has sampled the previous output. Combination of the short and long path constraints (eqn. 2.4) demonstrates that Razor can only guarantee correctness when the range of possible delays for a circuit output falls within a window of size

$$skew + hold < delay < T + skew \tag{2.3}$$

$$delay_{max} - delay_{min} < T - hold \tag{2.4}$$

Note that equation 2.4 implies a tradeoff between the limit of Razor protection and the range of Razor usability. While increasing skew can reduce the number of uncorrectable errors by protecting longer path delays, this also leads to a reduction in the range over which Razor can be applied to correct errors due to violation of the short path constraint.

The authors of [18] try to demonstrate the voltage scaling limitations of Razor based design using two circuits which are chosen to be canonical examples of two contrasting design philosophies i.e., a Kogge-Stone adder (KSA) and a Ripple Carry Adder (RCA)

The KSA architecture has timing paths of nearly the same length, and therefore exhibits a critical operating point akin to traditional high performance processor designs. Due to this property of KSA, it was observed that in [18] scaling beyond a certain voltage point leads

to a catastrophic failure of the adder (i.e., 100% error rate). Aggressive voltage scaling, therefore, is not possible for such designs, even with an efficient error correction mechanism, as the power consumption may actually increase drastically in spite of voltage scaling. This is because the absolute error rate is high (close to 100%) and the overhead of error recovery for Razor is roughly an order of magnitude more expensive than the overhead of executing an instruction normally [18]. So, for designs like KSA where timing paths are bunched up (like in traditional high performance processor designs), Razor may not be very effective in terms of power reduction through undervolting (1.e., scaling beyond the voltage for which the first timing violation appears). While some power can be saved by eliminating the voltage guardband, scaling past the critical operating point results in nearly 100% erroneous computations.

The second circuit, i.e., ripple carry adder is not subject to catastrophic failure in response to scaling past the point of first error as the circuit has wide range of path groups. Although the minimum delay for any path of the RCA equals the delay of the sum path of a full adder, operational delay ultimately depends on adder inputs, which generate carry chains from lower to higher order bits. The RCA exhibits maximum delay when the carry chain extends from the least significant bit to the most significant bit. However, on average, carry chains are much shorter, leaving extensive room for aggressive scaling past the point where errors begin to occur. In fact, the error rate reaches close to 100% only at very low voltages.

It might be obvious that Razor should perform well for architectures that fail gracefully, since such designs do not have a wall of criticality. However, analysis of the results in [18] reveals some serious limitations of using Razor, even in such architectures.

Limitations arise due to the potential short path and long path constraint violations discussed earlier. If the long path constraint is not satisfied, false negative detection can occur when the main flip-flop and shadow latch both latch the incorrect value. Similarly, the failure to meet short path constraints makes Razor unusable over a range of voltages.

In fact, the same factor that makes the error behavior of RCA graceful (skewed delay distribution) makes Razor less effective. This is because Razor relies on the variation in delay to be less than a threshold. The variation in delay is significantly larger for an RCA design than a KSA design. In order to make Razor work for circuits that fail gracefully, buffering must be used to increase the delay of short paths, thus shifting them into the window of correction. This buffering adds area and power overheads in a design, negating some of the power savings afforded by better than worst-case design. Secondly, required buffering increases the delay on short paths, transforming a circuit from one that fails gracefully to one that fails catastrophically, thus limiting the extent of possible scaling.

So, while Razor is ineffective for circuits like KSA because of massive timing violations in the face of undervolting, it is also not very effective for circuits like RCA due to a large span between the maximum and minimum circuit delays. The results in [18] demonstrate the inadequacies of current better than worst-case design methodologies (like Razor) in terms of voltage/ frequency scaling, motivating the need for new techniques for processor design and error handling.

2.3 CRISTA

CRISTA [15] is a low-power variation-tolerant circuit design called CRitical path ISolation for Timing Adaptiveness, which allows aggressive voltage scaling. The principal idea includes the following:

1.) Isolates the critical paths and makes them predictable (based on few primary inputs) under parametric variation so that with reduced supply voltage, possible delay errors under single-cycle operation are deterministic and can be avoided by a two-cycle operation.

2.) Restricts the occurrences of the previous two-cycle operations by reducing the activation probability of critical paths.

3.) Increases the delay margin between critical and noncritical paths by both logic synthesis

and proper gate sizing for improved yield, reliability of operations, and aggressive voltage scaling.

As mentioned above CRISTA induces the skewed distribution in completion delays for a given pipeline stage logic and further downsizes the critical path gates to increase the timing slack between non-critical and critical paths for performance/power benefits in a circuit. The occurrence of an error is detected by a Decode logic which monitors the activation probability of critical path vectors.

It can be understood intuitively that the performance or power benefit from such a design technique for designs such as a 32-bit RCA can be minimal. The primary reason is that RCA has an innate skewed distribution besides the decode logic used to flag for two cycle operation could impose an additional hardware penalty.

Therefore, though the design can be considered as a less conservative design for generic circuits with balanced critical paths it might not gain huge performance benefits from designs with skewed delay distribution such as RCA.

2.4 CSCD for High Speed and Area Efficient Arithmetic

This design [10] equips a Ripple Carry Adder with a current sensing capability which observes late settling carry signal nodes in the circuit and indicates when they reach a quiescent state. The incorporation of a computation completion signal into a RCA offers a way for improving the "average case" RCA to signal to the higher level circuitry controlling it that it has completed the operation. Thus, for example, if 32 repeated additions are to be performed to multiply two 32 bit numbers, using completion signalling to initialize the next addition can cut down the total multiplication time from 32 worst case addition delays, to 32 average case delays.

The proposed method for implementing the current sensor involves using a sense-inverter such as the one shown in Figure 2.3 . A CMOS inverter draws current from the power supply as long as its input is midrange between a high and a low voltage, such that both the P

and N transistors see a gate source voltage above their respective threshold voltages. The Supply current does not flow once the input reaches within a threshold of either the high or low supply voltages. Thus monitoring the supply current provides a means to determine if the input has stabilized close (within a threshold voltage) to a high or low.

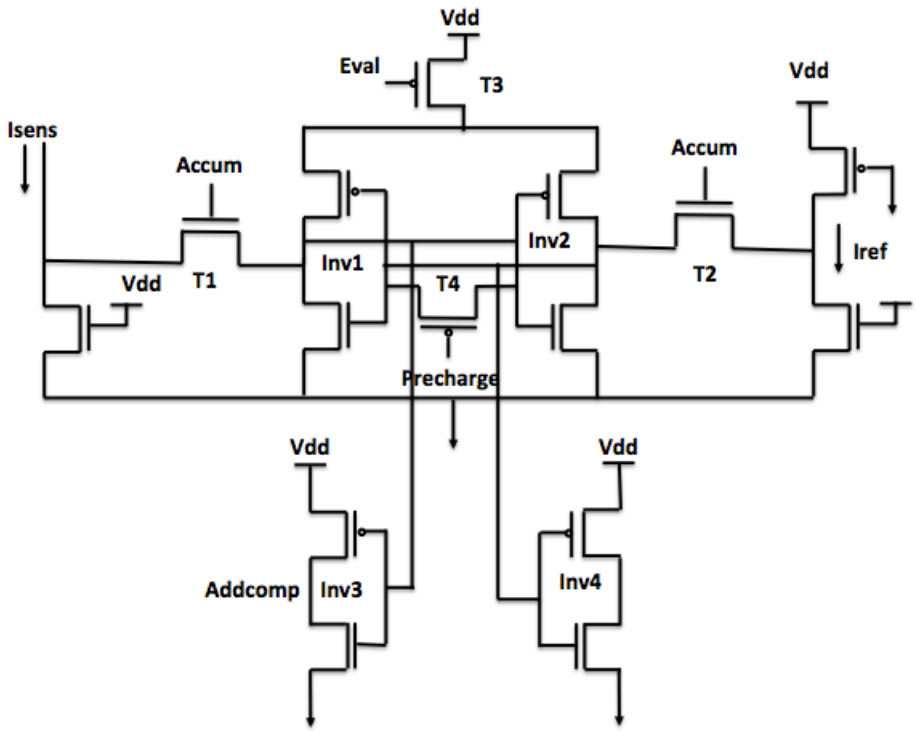


Figure 2.3: Current Sensor Circuitry

In Figure 2.3 the sense current is drawn from a transient current generator, i.e., an inverter whose IDD quiescent current is monitored with respect to the rising clock edge of each capture flop. The Addcomp signal is flagged in the case of an addition completion.

Though this design seems to be promising for circuits with a skewed delay distribution such as a RCA, the loading on the sense circuitry increases linearly with the number of capture flops to be monitored for transition completion, thus imposing an upper bound on performance or power benefits.

2.5 Need for reasonable area, power overhead and Resilient Error Detection Circuit

Though designs such as RAZOR, CRISTA and CSCD are less conservative compared to other BTWC designs, the performance/power benefits for a design with skewed delay distribution (like a RCA) using any of the above techniques is limited.

Thus, it is important to design a circuit that overcomes the short path constraints and also aids significant clock period scaling under the face of extreme timing variation. One such design technique is to buffer the short paths using dynamic latches at approprite segment delays of the combinational profile.

Chapter 3

Level-sensitive latch based scheme for hold-fixing

A key requirement for the proposed design is that during error-free operation, the delay and power overhead due to the error detection is minimal. Otherwise, the power gain from more aggressive clocking is overcome by the power overhead due to the presence of the error detection circuitry. The power consumption in this design is reduced significantly as a single clock is routed for both functional critical flops and shadow flops (negative edge triggered flop is used to achieve a timing window of $T_{period}/2$), avoiding the need for a second delayed clock. Also, many non-critical flip-flops, that do not capture signals from any long paths capable of exceeding the clock period, do not need this error detection design. If the maximum delay at the input of a flip-flop is guaranteed to meet the required cycle time under aggressive clock, the flip-flop cannot fail and does not need to be augmented with a shadow flop, thereby reducing the area as well as power.

3.0.1 Short Path Constraint

The use of a delayed clock at the shadow latch raises the possibility that a short path in the combinational logic will corrupt the data in the shadow latch. Figure 3 shows how a short-path allows data launched at the start of a cycle to be latched into the shadow latch, instead of the data launched from the previous cycle. To prevent this corruption of the shadow latch data, a minimum-path length constraint is added at the input of each functional critical flip-flop in the design. Figure 3.1 shows that the minimum path constraint is equal to the clock delay t_{delay} plus the hold time t_{hold} of the shadow latch (which is typically a small negative value).

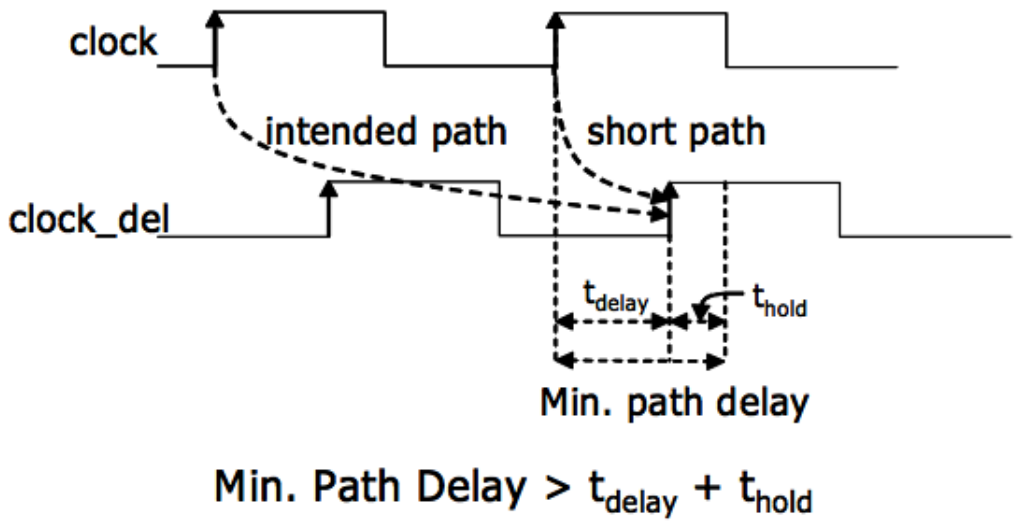


Figure 3.1: Short Path Constraints.

Earlier designs which include the Razor[12,13] delays the short paths using buffers. However addition of buffers during logic synthesis to slow down fast paths introduces a certain power overhead and with a large clock delay the severity of the short path constraint increases the power overhead due to the need for additional buffers. On the other hand, a small clock delay reduces the margin between the main flip-flop and the shadow latch, and hence reduces the amount by which the clock can be scaled below the critical clock frequency.

For most processors, buffer-insertion based hold-fixing effectively achieves the minimum-delay constraint without adding significant overhead. For example, the power overhead of hold-fixing was less than 3% of the total chip power for both the RazorI and the RazorII prototype processors. However, at aggressive process nodes, increasing variability in the short-paths can significantly worsen the hold-time constraint through the requirement of large number of delay buffers. In addition to the area and power overhead, excessive buffer insertion adversely affects the critical-path timing as well, thereby impacting performance.The limitations of Razor in the face of aggressive voltage scaling, coupled with the expectation of high variability in coming technology generations, motivates the need for alternative techniques that will allow full extraction of the power benefits available from voltage scaling.

3.0.2 Latch Based Buffering

The novelty of the proposed design lies in buffering the short paths using latches. Instead of adding buffers to increase the delay of short paths, we use latches (implemented as low cost dynamic latches by adding tristate controls at appropriate gates) to insert this delay. Specifically, we hold the signals in the short paths steady using the latches for a time period, which is equivalent to the clock skew between the main flops and the shadow flops. As, in this design a clock skew of $T_{period}/2$ (T_{period} being the aggressive clock) is achieved by using a negative edge triggered shadow flop, a negative level sensitive latch, placed at appropriate segment delays buffers the short paths that are less than $T_{period}/2$. These latches are transparent in the negative phase of the clock and sample in the positive phase of the clock. Figure 3.2 conceptually illustrates this scheme of pipeline transformation. The inputs

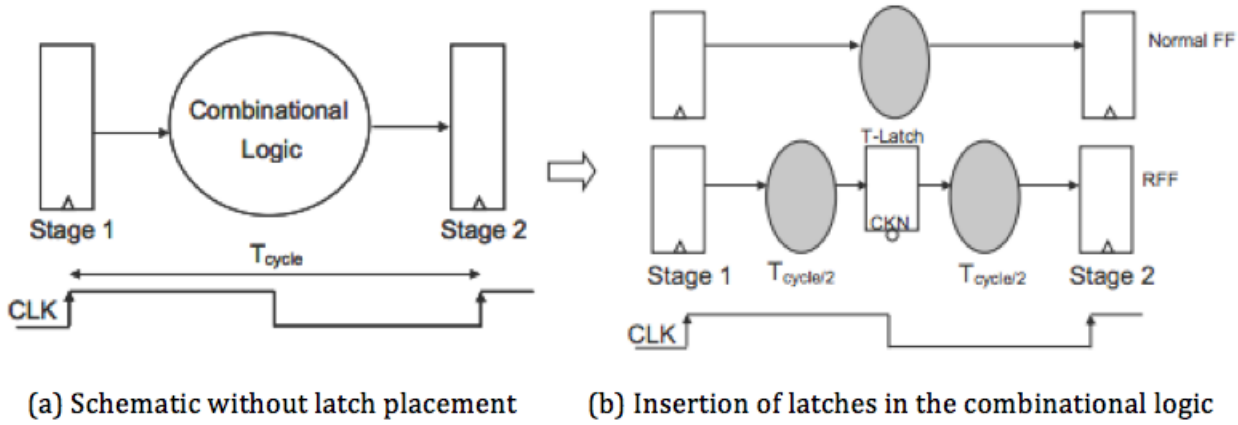


Figure 3.2: Latch insertion scheme for satisfying the short path constraint

to the functional and shadow flip-flops are prevented from being updated by short-path computations in the positive phase, due to the opaque latches. This guarantees, by construction, that the transitions which occur at the duplicated flip-flop inputs in the high clock-phase are valid timing errors caused due to setup violations. A point to note here is that the level-sensitive latches are required only in the logic cones that fan out to duplicated flip-flops. Maintaining a consistent clock skew of $T_{period}/2$ in the design guarantees a 33% scaling of the clock in an ideal scenario. Figure 3.3 illustrates the error detection mechanism in the

proposed design, (a) is the actual clock, which is subject to clock frequency scaling, and (b)

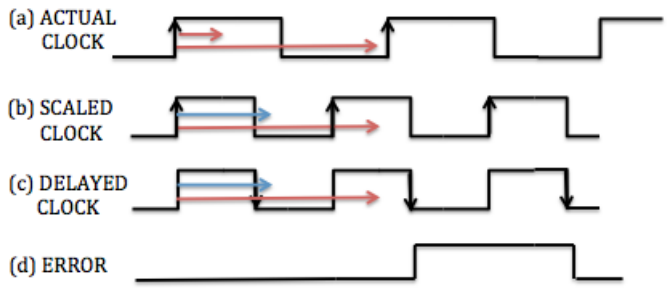


Figure 3.3: Timing model waveforms for error detection using a delayed clock

is the scaled clock. It is evident that the critical path is violating the timing in (b) and is captured unconditionally by the delayed clock (c), the delay which in this case is achieved using a negative edge triggered flop. As the output of the two flops are compared at every falling edge of the clock (error is captured using a negative edge triggered flop), an error is flagged when the polarity at the output of the two flops contradict. It can also be observed that the short path constraint is met, as all the short paths that are below the scaled clock period are buffered using negative level sensitive latches and thus are held for a time period of $T_{period}/2$. It has to be noted that the delayed clock cannot be scaled further below the critical path, as in this case both the functional and shadow flops capture the wrong value. Figure 3.4 illustrates the proposed design in a Huffman model with latches placed at key internal nodes. It can also be observed that short paths that encounter a latch (paths marked in red) are buffered to overcome the short path constraint and few short paths (paths that are not fed to shadow flop) are void of latches and hence there is no increase in the path delay.

The key advantage of this scheme is that conventional fifty percent duty cycle clocking can be used. This greatly simplifies the clock-generation and propagation. In addition, it is highly resilient to variability on the short-paths of the processor and eliminates the safety margins required to meet the minimum delay constraint.

Buffering all the short paths that converge to a critical path and thereby to a functional critical flop using latches is a meticulous task, as any inappropriate placement of a latch not

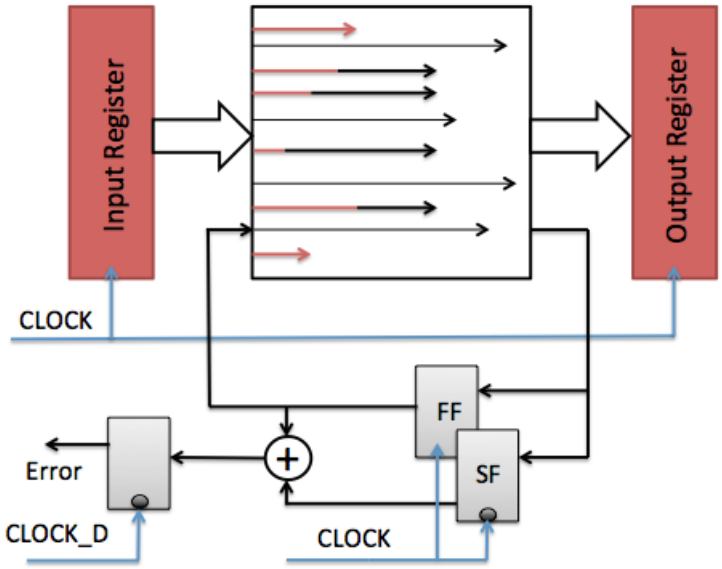


Figure 3.4: Schematic of the proposed design

only increases the delay of a long path that consequently triggers redundant errors but also corrupts the functionality of the circuit if the delay increase due to the latch placement is greater than 3/2 scaled clock period (greater than the delayed clock). Note however, that any latches introduced to handle the short paths do increase clock loading, and therefore their number must be minimized. This leads to certain constraints for the placement of latches. We present latch placement methodologies for the proposed design for two cases:

1) Clock Frequency Scaling without random variability.

2) Clock Frequency Scaling under the face of significant random variation

Chapter 4

Latch Placement for designs with and without timing variation

4.1 Latch Placement Algorithm

The paths in a combinational profile are very intricate, and include a wide range of path delays with interconnected paths. All these paths converge at different nodes and further to different functional flops. The combinational profile of a combinational circuit is very simple when compared to that of a sequential circuit. Placing latches amidst such a complex structure requires meticulous understanding of each and every path with its respective gate delays and path delays. For a given cone of logic that converges to a flop, it is always the longest path activated that gets latched in the flip-flop, and the probability for the longest path in that cone to activate is very low when compared to the short paths. A critical path is a long path in one of the logic cones which determines the clock period and so all the short paths that are activated within a clock period always meet the timing.

Latch placement constraints

1) All short paths with potential delays less than Tperiod/2 must include a hold latch to avoid comparison errors from capture of an updated signal in the shadow flop (driven by the next state captured at the end of the regular clock period).

2) No latch can be placed on any long path beyond the point where the segment delay from the flip-flop to the hold latch can potentially exceed Tperiod (under extreme variability). This is to avoid blocking and holding signals that do not reach the primary flip-flop at the end of Tperiod, during the subsequent Tperiod to 3/2 Tperiod window. Allowing this could result in the same incorrect old values being captured in both primary and shadow flip-flops.

3) No latch can be placed on any long path before the point where the segment delay from the hold latch to any flip flop can potentially exceed Tperiod (under extreme variability). This is to avoid blocking and holding signals during the first 1/2 Tperiod of a cycle to the point that they do not reach a final stable logic value even over 3/2 Tperiod necessary for error detection.

4) A Desired condition for performance is to ensure that the segment delay from flip-flop to the hold latch always exceeds Tperiod/2 (under extreme variablity), to ensure signal propagation is not delayed on long paths because of hold latches. Violation of this condition can potentially result in additional timing errors; but these will be reliably detected as long as the required conditions are met.

5) No path should encounter more than one latch, i.e., all the paths that are fed to funtional and shadow flops should have a unique latch. This is to ensure that no path under extreme variability is held twice when the path is activated.

Having mentioned the above constraints, it is obvious that a latch can effectively be placed on a long path that violates the aggressive clock but should satisfy the constraints. With all the path delays and their respective gate delays in a given circuit, latches can be placed using the latch placement algorithm. After targeting an aggressive clock, which is discussed in the subsequent sections, the steps shown in Figure 4.1 are implemented for the placement of latch.

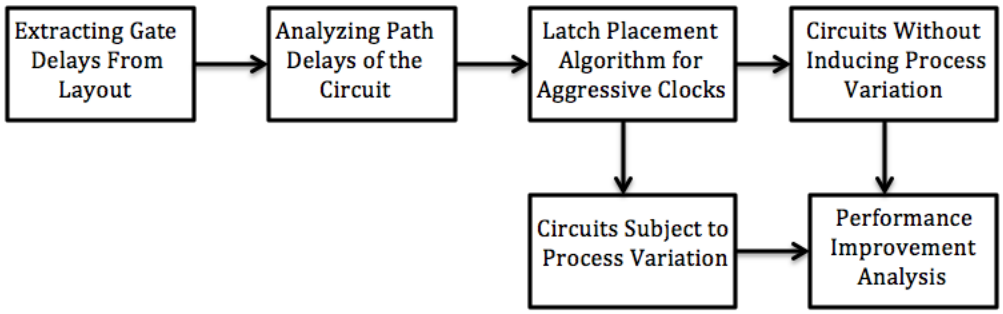
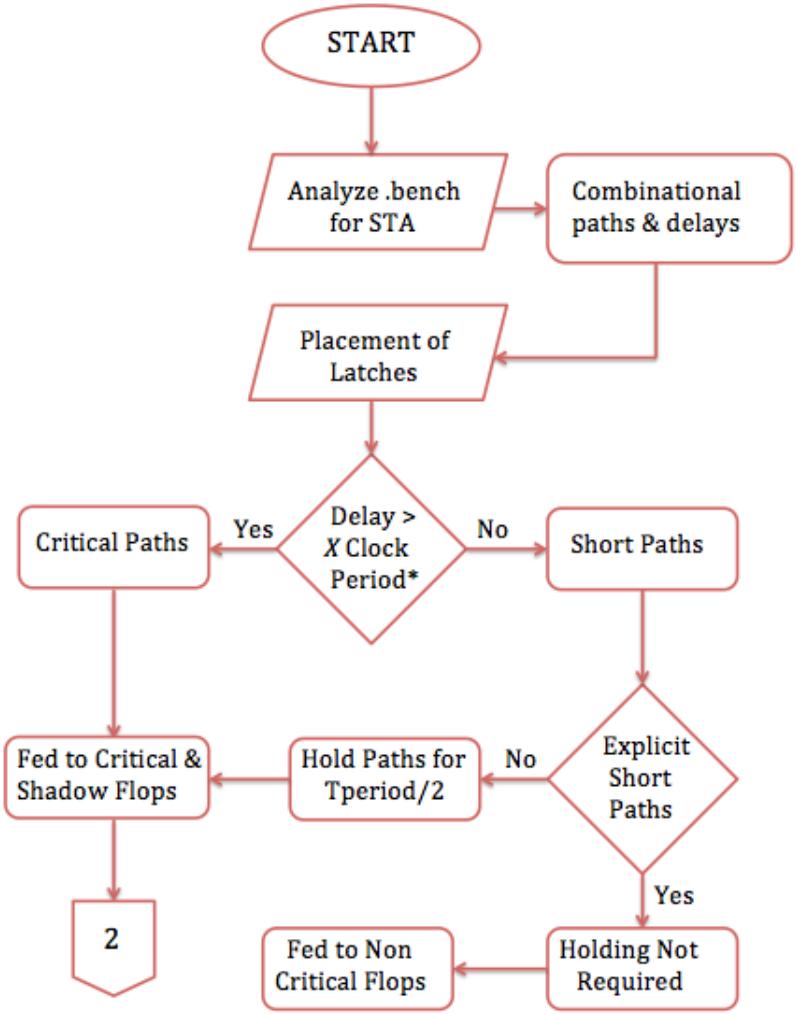


Figure 4.1: Latch Placement Flow

True path delays for a design would be extracted from the layout of that particular design and it is this delay file to which the latch placement is implemented. However, the following research was done using STA path delays for a given circuit and mimicing the characteristics of the extracted delays by adding a random delay of 1-4 units to the gate delay (an n input gate is assumed to have n units of gate delay) that accounts for parasitic capacitance and fanout parameters. STA path delays are given by an algorithm that accumlates the path delays from all the path groups in a sequential design. This is done by analysing the .bench file (ISCAS Benchmark representation) for a given circuit. A .bench file is a structural model for a given circuit at its abstract level. It depicts all the elements in the circuit with their respective input and output nets which are further fed to another element in the circuit. For example; G10 = NOR(G14, G11) is the .bench representation for a two input NOR gate with G14 and G11 as its inputs and G10 as the output.

A path delay file is given that has all the paths form different path groups with its respective gate delays. It is this file which is further analyzed for the placement of latches. After targetting an agressive clock, for a design without variability, all the flops that are fed with paths whose path delays are greater than the agressive clock period are to be augmented with shadow flops. However, for a design with variability there will be paths that after fabrication might fail the timing with the aggressive clock (constraints on the variabilty factor are assumed which are discussed in the further sections) and hence all the flops that are fed with potential paths that fail to meet timing under extreme timing variability are augmented with shadow flops. As discussed earlier, all the paths with path delay less than aggressive clock or 2/3 agressive clock (for designs without timing variability/with timing variability) that do not converge to the paths that violate timing need not be held. Figure 4.2 shows a generic flow chart for the placement of latches in a design with and without variability. It is further elucidated in the following sections.



4.2 Latch Placement for a Design without inducing Variability

The combinational profile in a sequential circuit can be characterized into two different models. The first one is the design with inherent skew where the difference between the mean path delay to the critical path is significantly large, while the second one is a design with balanced path delays, which has small range of timing paths. The benefits using the proposed design can be limited if the underlying circuit has a balanced path delay profile, as such circuits produce catastrophic failures in the face of clock frequency scaling [18]. However, with stringent latch placement in a cricuit with inherent skew, significant performance benefits can be assured.

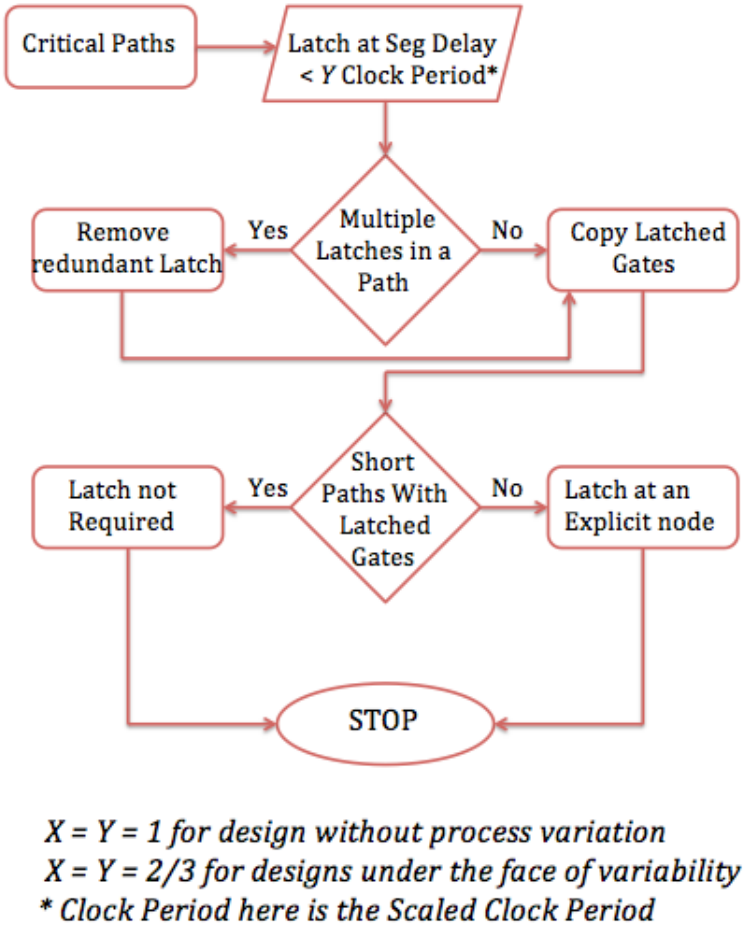


Figure 4.2: Flow Chart for Latch Insertion

Latch placement depends on the scaled clock frequency, as the primary criteria is to hold all the paths whose path delay is less than Tperiod/2 (Tperiod being the scaled clock period). To determine the aggresive clock, the delay distribution profile of a given circuit has to be analyzed. Depending on the hardware implementation, the computations can display a wide range of completion delays, depending on the applied inputs. For example a simple low cost (in hardware and power) 32-bit ripple carry adder (RCA) can generate a result with no carry propagation delays for some input cases, while requiring 32 stages of carry propagation in the worst case [11]. Importantly, for random inputs, this delay distribution is highly skewed, which can allow a significant speedup in the operating clock period while still ensuring only a small error rate. This is seen in Figure 4.3, which simulates addition completion delays for a 32bit RCA designed at the gate level and, for simplicity, assumes

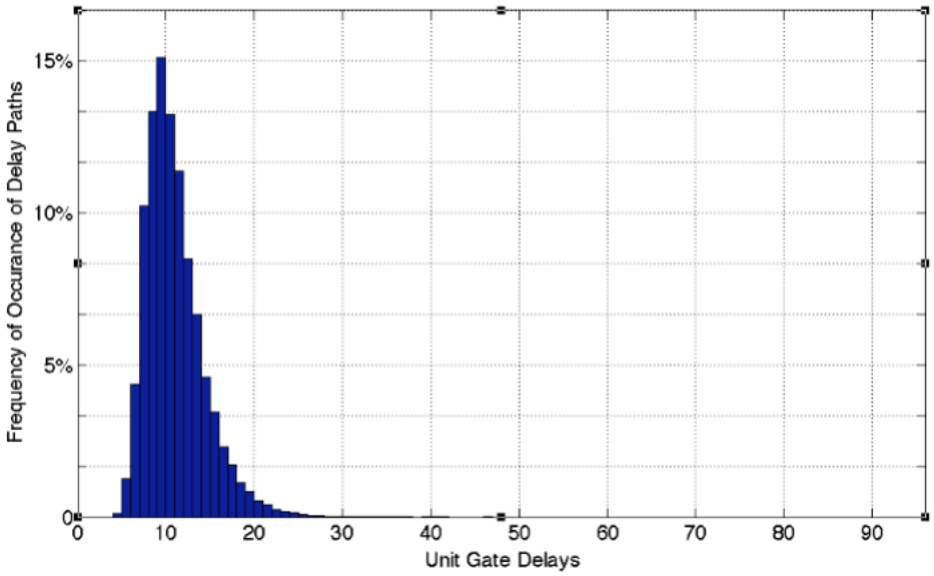


Figure 4.3: 32-bit Ripple Carry Adder Delay Distribution for Random Inputs.

fixed unit delay for every gate. Observe that the median addition delay is only 10 units when compared to the worst-case delay of 64 units. Note this does not mean that only 3-4 full adder stages generate a carry for the median case because strings of carries are generated and propagate in parallel at the same time. From the point of generation, a carry needs to see 01 or 10 inputs in the subsequent full adder stage for propagation, resulting in only a 50% chance that it will propagate through the next stage. The probability of a carry propagating n stages is 2^n , which dies out quickly with increasing n. Consequently, it if observed in Figure 2.2 it is very evident that, when compared to a worst case delay of 64 units, less than 0.1% of random inputs result in an addition delay of greater than 25 units. This suggests that the addition can potentially be run at 2.5X the worst case speed, with only one in a thousand operations, on average, requiring correction and recovery [6]. However, from the above stated constraints it is apparent that the clock cannot be scaled beyond 2/3rd the critical path and hence the aggressive clock is targeted at 50 units. This placement would also suffice if the clock period is scaled further as with a latch placement for 50 units clock, all the short paths below 25 units are efficiently buffered and hence scaling down further would still buffer the short paths below half the new clock period. The clock in this case

cannot be scaled below 43 units and hence the maximum performance improvement using this design is 33%.

With the estimated agressive clock, we now augment all the flops with shadow flops to which timing violated paths are fed. A latch is placed in all these timing violating paths by meeting the latch placement constraints. An effective latch placement for the design without variability would be placing a latch in the timing violating paths at a segment delay (from the flop output to the latch input) close to the scaled clock period. Doing so, a unique latch for all the paths converging to a flop would be sufficient to hold all the short paths. Placing a latch in a timing violating path without violating the constraints not only holds most of the short paths that converge to the critical path but also reduces the total number of latches in a design. Since all the short paths that converge to the timing violating path before the latch node are held, we now need to hold the short paths that converge beyond the latch node. Hence, we place a latch for all the short paths at explicit nodes of convergence.

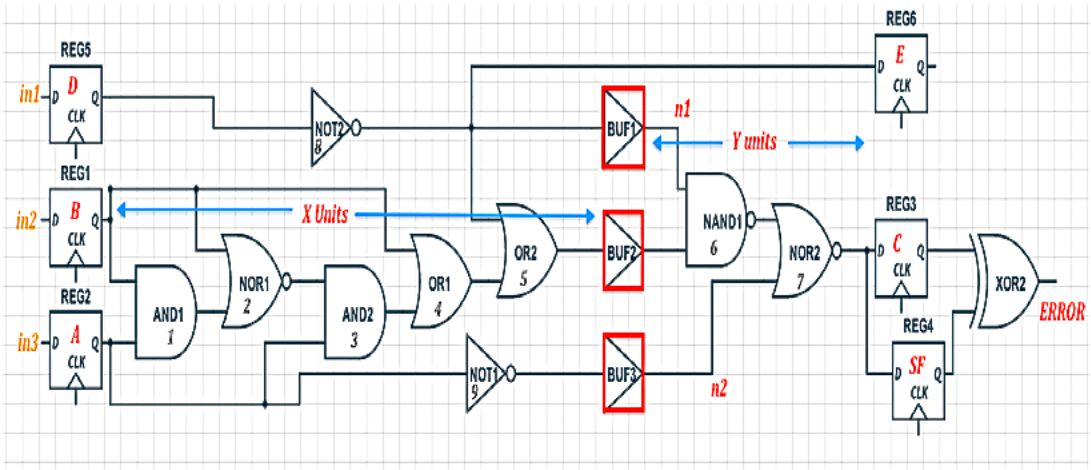


Figure 4.4: Effective Placement of a latch for a design without variation

For the design in the Figure 4.4, the critical path is assumed to be the path that starts from launch flop A to capture flop C, and the critical path to be x + y units. If the aggressive clock is targeted at n units (n < x + y), then a latch should be placed at a segment delay just less than n units (n = x + Δ), which in this case is x units. With this latch placement the clock can further be scaled without violating the condition that the segment delay from

the launch flop to the latch input is always less than the scaled clock period. The latch is released after $T_{period}/2$, and hence all the paths, which are greater than $T_{period}/2$ and less than T_{period}, always meet the timing, however the paths, which are greater than T_{period}, flag a timing error. With this placement, all the short paths (B - gate4 - gate5 - gate6 - gate7 - C and D - gate8 - gate5 - gate6 - gate7 - C) are inevitably held. All the other short paths (A - gate9 - gate7 - C; D - gate8 - gate6 - gate7 - C) that converge into the critical path beyond the latch are to be held at explicit nodes (nodes that does not fall in the critical path), which in this case are n1 and n2. The path from flop D to flop E is an explicit path (a short path that does not converge to a critical path) and hence is void of a shadow flop.

4.3 Latch Placement for a Design with Timing Variability

As high-performance processors move beyond 45nm technologies, designers face the major roadblock of parameter variation the deviation of Process, Voltage, and Temperature (PVT) [22-27] values from nominal specications. Variation makes designing processors harder because they have to work under a range of parameter. A key process parameter subject to variation is the transistor threshold voltage (Vth). Variation in Vth directly impacts two major properties of the processor, namely the frequency it attains and the leakage power it dissipates. The delay of an inverter gate is given by the alpha-power model as:

$$T_g \propto \frac{L_{eff}}{\mu(V - V_{th})^{\alpha}} \tag{4.1}$$

where L_{eff} is the effective channel length of a transistor, α is typically 1.3, and μ is the mobility of carriers which, as a function of T, is $\mu(T) \propto T^{-1.5}$. As V_{th} decreases, V - V_{th} increases and the gate becomes faster. As T increases, V_{th} decreases and, as a result, V - $V_{th}(T)$ increases. However, $\mu(T)$ decreases [21] . The second factor dominates and, with higher T, the gate becomes slower.

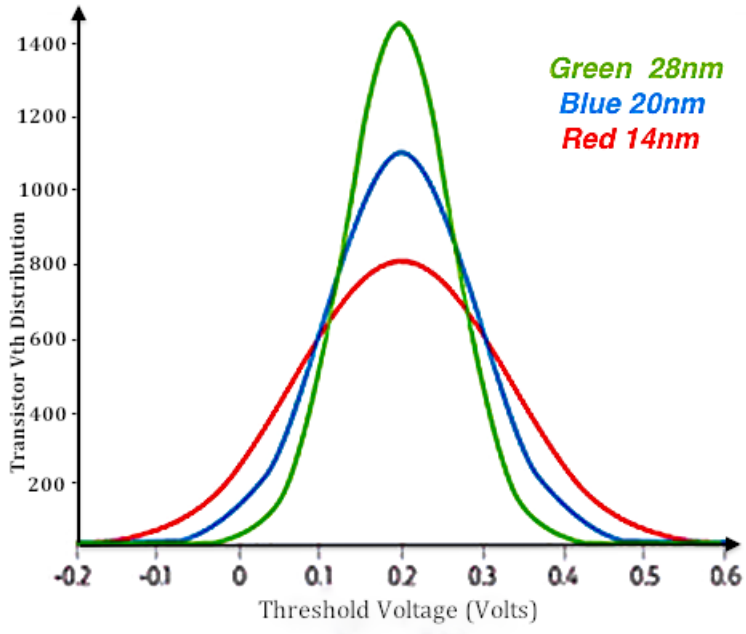


Figure 4.5: Process Variation effects on Vth at different technology nodes

Threshold voltage ideally varies as little as possible from device to device. However, as we move to smaller geometries, threshold voltage has increasingly larger variance; this larger variance in device performance translates to more variance in circuit performance and hence the path delay profile, which is skewed when compared to that of a design with zero variance in threshold voltage. It is also this skewed behavior that is exploited for improving the performance in a given circuit. Latch placement for the design with variabilty requires certain conditions on the variability parameter, as the latch placement should be unique for all the circuits under the face of extreme variation. From Figure 4.5 it is obvious that process variation in Vth is responsible for both speeding up and slowing down the delay of a transistor. Slowing down of paths is more obvious than speeding up of any path, as the increase in delay of a gate is exponential with higher Vth, whereas with lower Vth the decrease is the delay is not that significant. Following are the conditions for process variation factor in speeding up and slowing down of paths.

Slow down of paths

1) No path less than 2/3 T_{period} can slow down more than T_{period}.

2) Under extreme variability, paths longer than 2/3 T_{period} should not exceed 3/2 T_{period}.

Speed up of paths

1) Under extreme variability, paths longer than 2/3 T_{period} cannot speed up more that 33% (to be considered when there is no latch in a long path that might speed up and fall below $T_{period}/2$).

Note that T_{period} here is the scaled clock period. The aggressive clock in this case is determined by the critical path of a nominal circuit whose gate delays are the weighted average of n circuits gate delays that are subject to extreme process variation. It has to be noted that the clock cannot be scaled beyond 33% of the outlier circuits critical path. From the above limitations over the impact of variation on the gate delays, it is evident that all the nominal path delays greater than 2/3 T_{period} might potentially exceed T_{period} thereby failing to meet the timing. Hence, all these paths are fed to both functional and shadow flops. It is also obvious that no latch can be placed in a path where the segment delay from the flop output to the latch input is greater that 2/3 T_{period} (from the latch placement constraints). Considering the above-mentioned limitations, the latch placement is similar to that of a design without variation.

4.4 Inducing Variation in a given circuit

The effect of variation on the delay of a gate is characterized by its respective transistors threshold voltages. There are two possible transitions at the output of a gate: Low-to-High (rising) and High-to-Low (falling). Although the gates may be designed for same low-to-high and high-to-low delays in the nominal case, under random process variations these two delays can be different. Effect of variation on Vth of a pmos/nmos device impacts the rising/falling transition of that particular gate. Table 4.1 shows the average of rising and falling delays

Table 4.1: Effects of variation on the average of rising and falling transitions of an inverter operated at multiple voltages

Δ Vth	Average of rise and fall time transition of an inverter ps					
	1V	0.9V	0.8V	0.7V	0.6V	0.5V
0.1	3.44	3.98	4.51	5	6.14	9.75
0.2	4.21	4.73	5.12	5.94	8.19	13.9
0.3	5.1	5.45	5.96	8.08	12.77	26.57
0.4(nom)	6.35	7.79	9.35	11.32	20.62	48.21
0.5	8.2	9.32	13.15	16.45	36.65	111.05
0.6	9.34	11.7	15.86	25.85	56.01	191.7
0.7	10.77	14.73	19.52	38.79	100.62	274.5
0.8	12.73	17.26	23.28	72.38	198.26	427.45
0.9	16.33	22.32	29.47	129.84	428.88	697.25

of an inverter, simulated in HSPICE for 45nm technology for a threshold voltage range of 0.1V to 0.9V (with a nominal threshold voltage being 0.4V) operated at multiple voltage corners. The delay of the inverter was characterized by analyzing the falling transition for variation in nmos and rising transition for variation in pmos. Figure 4.6 shows the impact of variation on the average of rising and falling transition of an inverter; it is obvious that the delay increase is exponential when the device is operated at a voltage close to the threshold voltage of the device.

For a circuit with x gates operated at a given voltage, a Gaussian distribution is plotted with a mean of 0.4 (mean at nominal threshold voltage) and a variance of 0.05. The threshold voltage from the distribution is rounded off to the values in the table and the corresponding delay factor is multiplied with the gate delay.

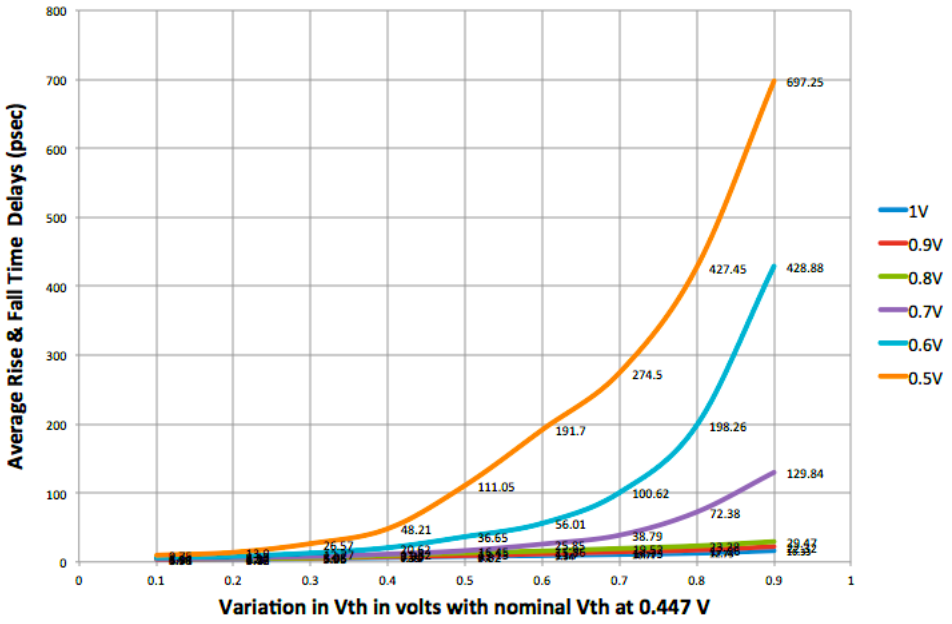


Figure 4.6: Increase in delay of an inverter for multi Vth operated at various voltage corners

Chapter 5

Latch placement for an 8-bit ripple carry adder subject to timing variation

In this section we study a simple circuit that displays a wide spread in input-based path delays to illustrate the latch placement and the potential power/performance gains. The combinational profile of a ripple carry adder is elementary. An n bit ripple carry adder has n full adder modules connected in series. The carry bit of one full adder block is fed to another and so it is obvious that the critical path will be from the carry in bit to the final carry out bit. From Figure 5.1 the critical path would be from one of the three inputs A0, B0, Cin to Cout.

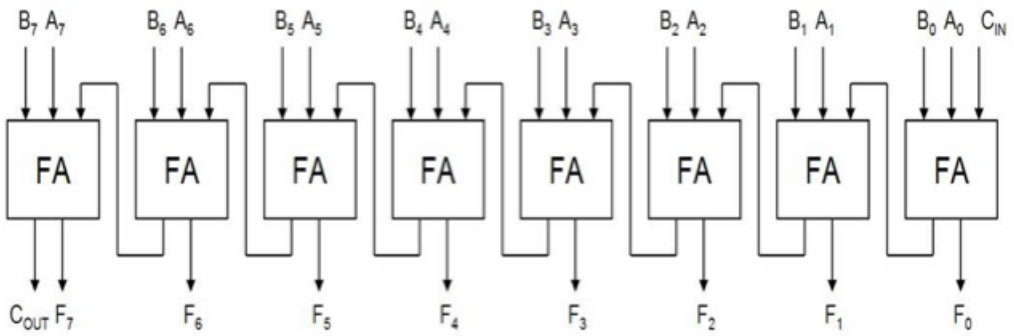


Figure 5.1: 8-bit Ripple Carry Adder

The ripple carry adder in this section was implemented using primitive gates, and the critical path, measured in unit gate delays, was observed to be 68 units from the input pin A0 to carry out. Figure 5.2 shows the final four full adder modules of a an 8-bit ripple carry adder and Figure 5.3 illustrates the final four full adder modules of ripple carry adder with latches placed in the design. As discussed earlier, a unique latch placement will be implemented considering the path delay profile of this nominal design, with an aggressive clock of 68 units (neglecting PVT timing margin). Hence, under the face of variation all the circuits are run at a clock of 68 units with the error detection mechanism. For the

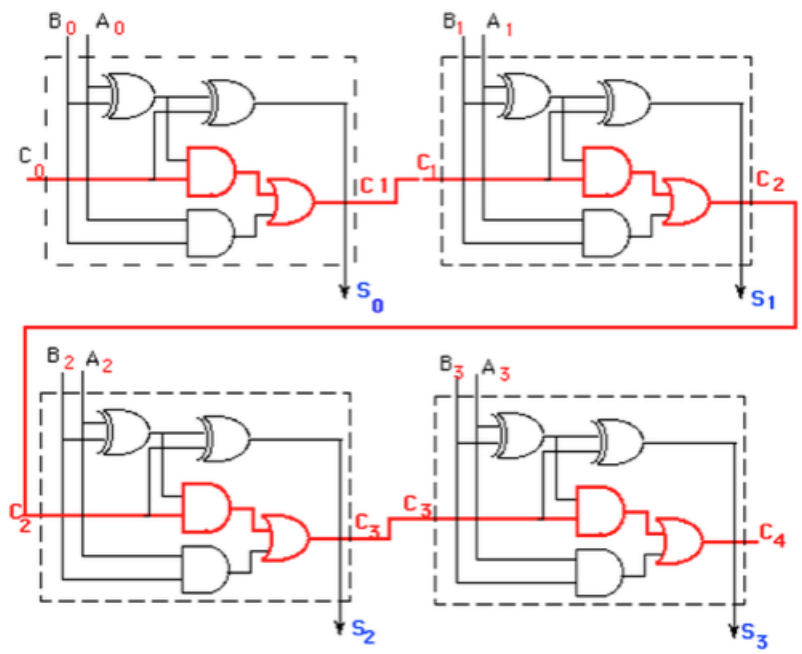


Figure 5.2: Last four full adder modules of an 8-bit Ripple Carry Adder

nominal design, all the paths that are greater that 2/3 aggressive clock (45.33 units) are probable paths that under extreme variation might fail to meet the timing. Hence, all the flops to which these paths feed are augmented with shadow flops. For the above design, it was observed that all the paths that converge to the outputs F4 to F7 and Cout (at least one path to these outputs have a path delay greater than 45 units) under extreme variation fail to meet the timing and so the flops are augmented and latches are to be placed to hold all the potential short paths that are less that $T_{period}/2$ (34 units). Rest of the flops need not be augmented, as all the paths feeding these flops will meet the timing under extreme variation. Figure 9 shows the final 4 full adder modules of a ripple carry adder, with the xor gates in the full adder block flattened for the placement of latches. Latches in this figure are depicted using buffers. The critical path is from the carry out bit C3 to the final carry out (A0 to Cout). A unique latch is placed in all the paths that fail to meet timing under extreme variation and henceforth hindering the progress of any short paths that are less than $T_{period}/2$. For the critical path, a latch is placed on the carry out path of the fourth full adder module at a segment delay of 45 units from the launch flop (A0) to the latch input

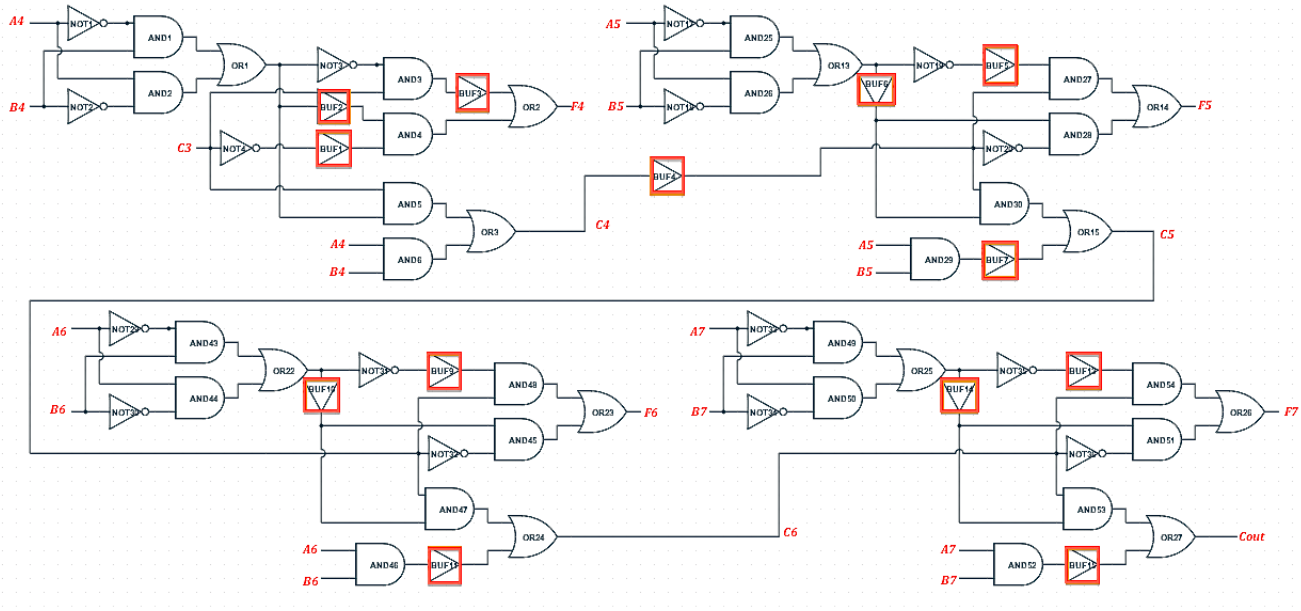


Figure 5.3: Last four full adder modules of an 8-bit Ripple Carry Adder with latches placed (Buf4). Looking at the conditions on the effects of variation on delay it is obvious that this segment delay can never violate the timing ((45x1.5/67.5 $<$ 68) and this this latch holds all the short paths that converge before the latch node (A4, B4 to F5, F6, F7 and Cout). However, paths that converge beyond the latch are held at explicit nodes (A5, B5; A6, B6; A7; B7 to F5, F6, F7, Cout).

Chapter 6

Design Flow Approach For Latch Insertion in a Given Verilog Netlist

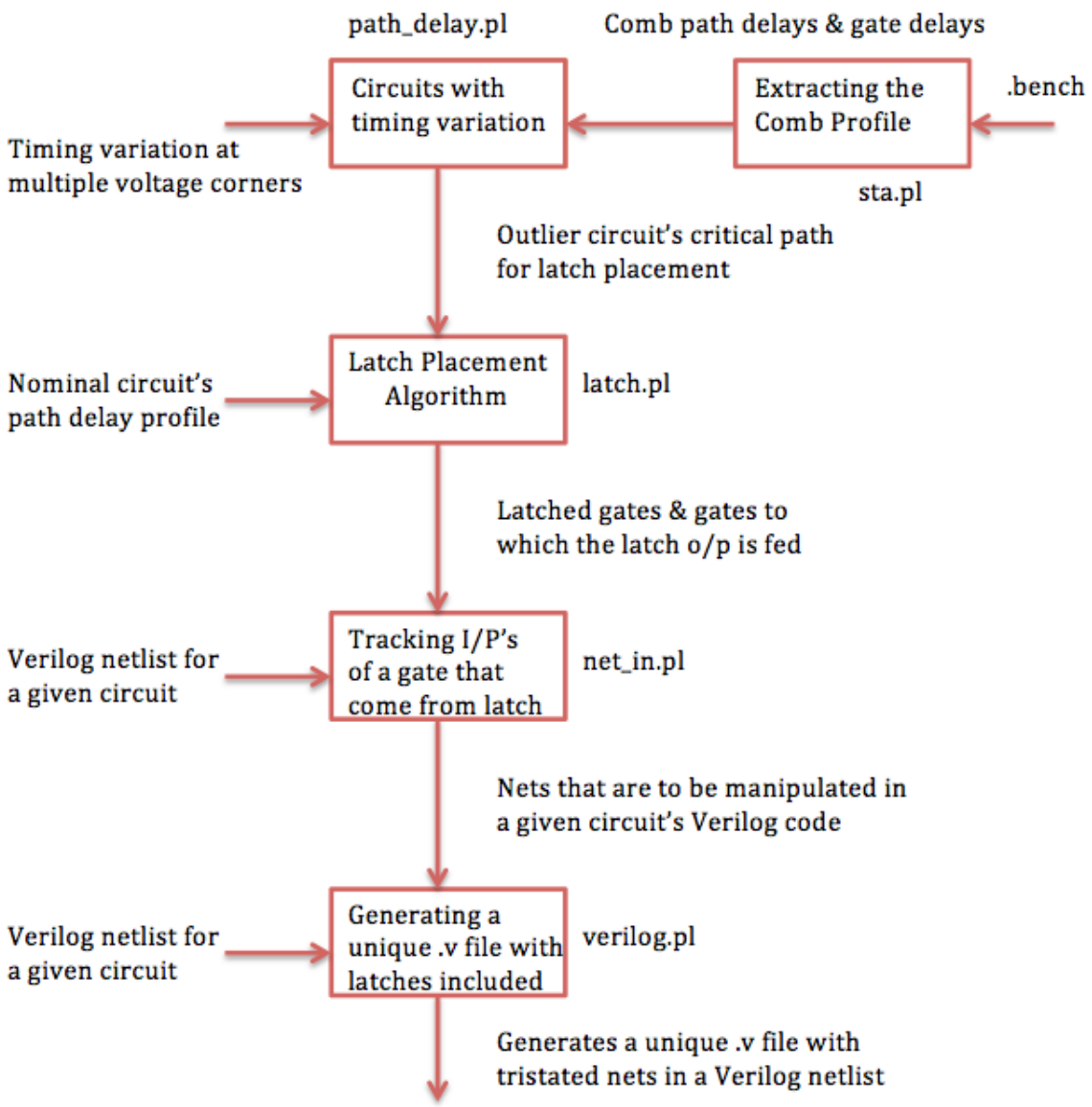


Figure 6.1: Design flow for the latch insertion

```
\\.bench representation for s27 sequential benchmark circuit
#4inputs/1output
#3 internal D-typeflipflops
#10gates(2NOTs+1ANDs+1NANDs+2ORs+4NORs)

INPUT(G_0)
INPUT(G_1)
INPUT(G_2)
INPUT(G_3)
OUTPUT(G_17)

G5=DFF(G10) # Internal Registers
G6=DFF(G11)
G7=DFF(G13)

G0=DFF(G_0) # Input Register
G1=DFF(G_1)
G2=DFF(G_2)
G3=DFF(G_3)

G_17=DFF(G17) # Output Register

G14=NOT(G0)
G17=NOT(G11)
G8=AND(G14,G6)
G15=OR(G12,G8)
G16=OR(G3,G8)
```

```
G9=NAND(G16,G15)
G10=NOR(G14,G11)
G11=NOR(G5,G9)
G12=NOR(G1,G7)
G13=NOR(G2,G12)
```

Part of the combinational profile with gate, nets and respective path delays

```
-------flop-q to flop-d paths-------------
3 paths for flop-q G5
 {G5 G11
 [4-4
 (NOR2_1
```

For the above path, nets G5 and G11 are the inputs and outputs of the gate NOR2_1 with a gate delay of 4 units. Since this path has a single gate from flop to flop, the path delay is same as the gate delay which is 4 units.

```
 {G5 G11 G17
 [6-4 2
 (NOR2_1 NOT_1

 {G5 G11 G10
 [8-4 4
 (NOR2_1 NOR2_0
```

```
6 paths for flop-q G6
 {G6 G8 G15 G9 G11
 [15-3 4 4 4
 (AND2_0 OR2_0 NAND2_0 NOR2_1

 {G6 G8 G15 G9 G11 G17
 [17-3 4 4 4 2
 (AND2_0 OR2_0 NAND2_0 NOR2_1 NOT_1

 {G6 G8 G15 G9 G11 G10
 [19-3 4 4 4 4
 (AND2_0 OR2_0 NAND2_0 NOR2_1 NOR2_0

 {G6 G8 G16 G9 G11
 [15-3 4 4 4
 (AND2_0 OR2_1 NAND2_0 NOR2_1

 {G6 G8 G16 G9 G11 G17
 [17-3 4 4 4 2
 (AND2_0 OR2_1 NAND2_0 NOR2_1 NOT_1

 {G6 G8 G16 G9 G11 G10
 [19-3 4 4 4 4
 (AND2_0 OR2_1 NAND2_0 NOR2_1 NOR2_0
```

Latch placement amidst the combinational profile; each and every path in the

```
combinational profile encounters a unique latch

AND2_0 OR2_0 NAND2_0 LATCH NOR2_1
3 4 4 4   15
AND2_0 OR2_1 NAND2_0 LATCH NOR2_1
3 4 4 4   15
NOR2_2 OR2_0 NAND2_0 LATCH NOR2_1
2 4 4 4   14
NOT_0 AND2_0 OR2_0 NAND2_0 LATCH NOR2_1
1 3 4 4 4   16
NOT_0 AND2_0 OR2_1 NAND2_0 LATCH NOR2_1
1 3 4 4 4   16
NOR2_2 OR2_0 NAND2_0 LATCH NOR2_1
2 4 4 4   14
AND2_0 OR2_0 NAND2_0 LATCH NOR2_1 NOT_1
3 4 4 4 2   17
AND2_0 OR2_1 NAND2_0 LATCH NOR2_1 NOT_1
3 4 4 4 2   17
NOR2_2 OR2_0 NAND2_0 LATCH NOR2_1 NOT_1
2 4 4 4 2   16
NOT_0 AND2_0 OR2_0 NAND2_0 LATCH NOR2_1 NOT_1
1 3 4 4 4 2   18
```

Chapter 7

Simulation Results

Unit delay simulations were done for both the designs with and without timing variation. For the design without variation a 32-bit ripple carry adder was simulated, and for the design with process variation multiple sequential benchmark circuits were implemented along with a 32-bit ripple carry adder. For a 32 bit ripple carry adder without variation, the critical path was observed to be 230 units and the latch placement was done for an aggressive clock of 175 units. The ripple carry adder in this simulation was constructed using primitive gates, and hence the total number of gates in the adder are 416 (128 not gates, 292 and gates, 96 or gates) and the total flops that are used to capture the result are 33. The total number

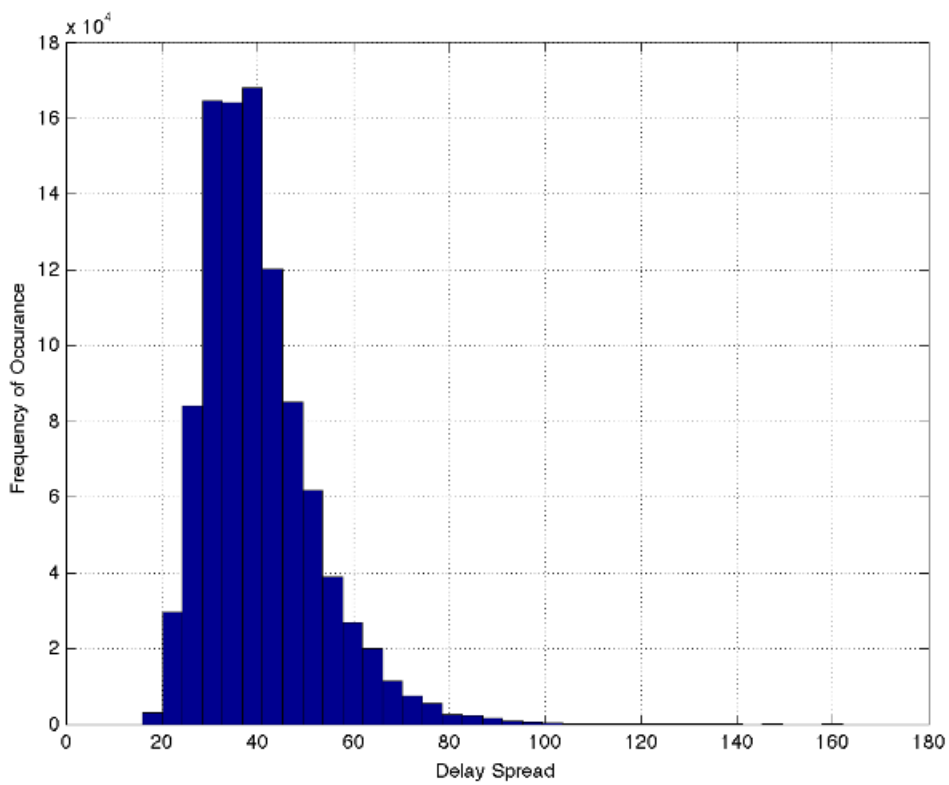


Figure 7.1: Delay Spread of a 32-bit Ripple Carry Adder for 1 million vectors

Table 7.1: Unit Delay Simulations for a 32-bit Ripple Carry Adder with the Latch Placement and Clock Frequency Scaling

FF/SF	Total no. of Latches	Critical Path	Aggressive Clock	Average Clock (3 clock cycle penality)	Performance Improvement
97/12	38	230 units	175 units	175 units	1.31x
97/12	38	230 units	165 units	165 units	1.39x
97/12	38	230 units	154 units	154 units	1.49x

of flops to which timing violated paths (path delay greater than 175 units) feed are 12 and hence all these flops are augmented with shadow flops. Total number of latches in this design given by the latch placement algorithm was 38. Hence, to run a 32 bit ripple carry adder at an aggressive clock of 175 units, requires an additional area that accounts for shadow flops and latches. Further the clock can be scaled beyond the aggressive clock with the same latch placement, however it cannot be scaled beyond 2/3rd the critical path of the circuit (153.33 units) as in this case, if the critical path is triggered, both the shadow flop and the functional flop capture wrong values leading to the failure of the design. The aggressive clock was estimated using the delay spread of the 32-bit ripple carry adder. The delay distribution of the circuit is shown in Figure 7.1.

The simulation table shows the total number of flops that are to be augmented and the number of latches required for Clock Period Scaling. The average clock is the total clock period required to compute an addition operation that also accounts for a 3-clock cycle penalty for the error recovery mechanism. For the above simulations the errors observed were negligible at 175, 164 and 154 units for 1 million vectors and hence the average clock is observed to be the same as the aggressive clock. PI reffered in the table 7.1 is the Performance improvement. Hence, this design when equipped in a circuit with inherent skew would improve the performance. In the above case, a 1.5x performance improvement is guaranteed with considerable area overhead. It is evident that the clock can be scaled further beyond 2/3rd the critical path, however the design constraints limit the clock frequency

Table 7.2: Simulation Results For s444 with the proposed design operated at Multiple Voltage Corners

Supply Voltage	FF/SF	Total no. of Latches/ Gates	Nominal CP/ Outlier CP	Avg CP subject to PV	Aggressive Clock (3 Cycle penality)	Avg performance improvement	Best case improvement
1V	30/15	26/171	250/315	272.5	233.39	14.40%	26.1%
0.9V	30/15	25/171	306/376	342.33	290.21	13.3%	22.9%
0.8V	30/14	25/171	367/507	425.09	357.54	16%	20.6%
0.7V	30/14	23/171	445/877	723	646.3	10.7%	26.4%

Table 7.3: Simulation Results s510 with the proposed design operated at Multiple Voltage Corners

Supply Voltage	FF/SF	Total no. of Latches/ Gates	Nominal CP/ Outlier CP	Avg CP subject to PV	Aggressive Clock (3 Cycle penality)	Avg performance improvement	Best case improvement
1V	32/4	14/201	326/377	347.33	318.4	9.40%	13.7%
0.9V	32/4	13/201	381/459	424.07	369.21	13.00%	19.30%
0.8V	32/4	14/201	479/594	537.79	468.54	12.90%	21.3%
0.7V	32/3	12/201	625/795	690.76	613.32	11.20%	23.9%

scaling beyond a certain point. Thus with this approach in an ideal scenario, the maximum performance benefit that can be assured is no more than 33%.

Unit delay simulation results for the design with variability were done for sequential benchmark circuits that include s444, s510, s641, s1196 and a 32-bit Ripple Carry Adder. The unit delay simulations for these circuits were done with the gate delays that mimic the true gate delays of the circuit operated at multiple voltage corners. NCP and OCP referred to in the tables 7.2 to 7.6 to VII are Nominal Critical Path and Outlier Critical Path, respectively. Simulation results show average performance (API) and best-case performance improvement (BPI) in a design with the latch placement. Average Performance Improvement (API) is the performance improvement for circuits operated at the aggressive clock (including a 3 clock cycle penalty) to the average of individual clocks, which in this simulation is the weighted average of the circuits critical path whose path delay is greater than the aggressive clock (Avg

Table 7.4: Simulation Results For s641 with the proposed design operated at Multiple Voltage Corners

Supply Voltage	FF/SF	Total no. of Latches/ Gates	Nominal CP/ Outlier CP	Avg CP subject to PV	Aggressive Clock (3 Cycle penality)	Avg performance improvement	Best case improvement
1V	78/16	43/379	1329/1457	1385.58	1192.30	14.00%	18.20%
0.9V	78/16	45/379	1644/1842	1705.12	1426.61	16.40%	22.68%
0.8V	78/16	43/379	1962/2268	2145.42	1743.43	18.80%	23.25%
0.7V	78/15	46/379	2384/3083	2658.62	2247.60	19.30%	27.22%

Table 7.5: Simulation Results For s1196 with the proposed design operated at Multiple Voltage Corners

Supply Voltage	FF/SF	Total no. of Latches/ Gates	Nominal CP/ Outlier CP	Avg CP subject to PV	Aggressive Clock (3 Cycle penality)	Avg performance improvement	Best case improvement
1V	46/19	72/529	547/711	636.35	523.39	17.80%	26.50%
0.9V	46/19	72/529	669/ 886	758.75	632.21	16.78%	28.73%
0.8V	46/17	78/529	805/1210	998.69	835.65	16.32%	27.87%
0.7V	46/17	74/529	977/2089	1824.45	1508.23	17.33%	27.90%

CP). The best-case performance improvement (BPI) is the performance improvement for the outlier circuit (the circuit with the largest critical path among the 100 circuits) operated at aggressive clock to the actual clock frequency of operation (critical path of the outlier circuit). In all of the designs that are operated at 1v to 0.8v the outlier circuit‘s critical path is less than 1.5x of the nominal circuit‘s critical path. Hence, the aggressive clock was assumed to be close to 2/3rd that of the outliers critical path. For the design operated at 0.7v the outlier‘s critical path is greater that 1.5x the nominal design, and hence the aggressive clock is greater than the nominal design‘s critical path. This is because of the fact that the aggressive clock cannot be scaled beyond 2/3rd the outlier circuit‘s critical path. The simulation results for a 32-bit ripple carry adder are significant to that of the sequential designs, as this circuit has inherent skew and with the additional parameter (process variation) that exaggerates the already existing skew in the design reaping a significant performance improvement.

Table 7.6: Simulation Results For 32-Bit RCA with the proposed design operated at Multiple Voltage Corners

Supply Voltage	FF/SF	Total no. of Latches/ Gates	Nominal CP/ Outlier CP	Avg CP subject to PV	Aggressive Clock (3 Cycle penality)	Avg performance improvement	Best case improvement
1v	97/15	51/416	260/340	306.41	227	23.90%	32%
0.9V	97/15	52/416	302/415	373.34	277	23.74%	32%
0.8V	97/14	50/416	406/620	518.10	414	20.10%	32%
0.7V	97/14	54/416	548/1042	814.41	696	14.53%	32%

Simulations for the designs at voltages 0.6 and 0.5 could not be presented as the outlier circuit‘s critical path was observed to be 3-5 times that of the nominal circuit‘s critical path. Thus, it is evident that paths that are less than the aggressive clock increase with a factor greater than 5. This complicates the assumption of augmenting very few functional flops with shadow flops. For this simulation we need to augment all the flops in the design with shadow flops, which might improve the performance of the circuit at a cost of increased area overhead.

7.1 Conclusion

In this thesis we studied latch-based buffering at selected circuit locations to overcome the short path problem associated with duplication-based error detection in better-than-worst-case-timing designs. Our approach, which employs dynamic latches implemented as tristate gate outputs, not only avoids the significant overhead of conventional buffer chains used in conventional Razor designs, but is also more robust to the case of extreme timing variations observed in scaled technologies. Using realistic gate level delay simulations that incorporate random gate delay variability drawn from actual SPICE level simulations, we studied multiple sequential benchmark circuits. The results presented here show an average performance improvement of 15% and best case performance improvement of about 33% in both the designs. It is important to note that better-than-worst case designs also allow

elimination of the 10-20% traditional timing margins employed, so the average performance gain compared to conventional design can be expected to be 25% or greater.

Bibliography

[1] Moore, G.E., "No exponential is forever: but "Forever" can be delayed! [semiconductor industry]," Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC. 2003 IEEE International , vol., no., pp.20,23 vol.1, 13-13 Feb. 2003

[2] Xuejue Huang; Wen-Chin Lee; Kuo, C.; Hisamoto, D.; Chang, L.; Kedzierski, J.; Anderson, Erik; Takeuchi, H.; Choi, Yang-Kyu; Asano, K.; Subramanian, Vivek; Tsu-Jae King; Bokor, J.; Chenming Hu, "Sub-50 nm P-channel FinFET," Electron Devices, IEEE Transactions on , vol.48, no.5, pp.880,886, May 2001

[3] Topol, A.W.; Tulipe, D.C.La; Shi, L.; Frank, D.J.; Bernstein, K.; Steen, S.E.; Kumar, A.; Singco, G.U.; Young, A.M.; Guarini, K.W.; Ieong, M., "Three-dimensional integrated circuits," IBM Journal of Research and Development , vol.50, no.4.5, pp.491,506, July 2006

[4] Swanson, R.M.; Meindl, J.D., "Ion-implanted complementary MOS transistors in low-voltage circuits," Solid-State Circuits, IEEE Journal of , vol.7, no.2, pp.146,153, Apr 1972

[5] Wang, A.; Chandrakasan, A., "A 180mV FFT processor using subthreshold circuit techniques," Solid-State Circuits Conference, 2004. Digest of Technical Papers. ISSCC. 2004 IEEE International , vol., no., pp.292,529 Vol.1, 15-19 Feb. 2004

[6] Bo Zhai; Nazhandali, L.; Olson, J.; Reeves, A.; Minuth, M.; Helfand, R.; Pant, S.; Blaauw, D; Austin, T., "A 2.60pJ/Inst Subthreshold Sensor Processor for Optimal Energy Efficiency," VLSI Circuits, 2006. Digest of Technical Papers. 2006 Symposium on , vol., no., pp.154,155, 0-0 0

[7] Bo Zhai; Blaauw, D; Sylvester, D; Flautner, K., "Theoretical and practical limits of dynamic voltage scaling," Design Automation Conference, 2004. Proceedings. 41st , vol., no., pp.868,873, 7-11 July 2004

[8] Hanson, S.; Bo Zhai; Mingoo Seok; Cline, B.; Zhou, K.; Singhal, M.; Minuth, M.; Olson, J.; Nazhandali, L.; Austin, T.; Sylvester, D; Blaauw, D, "Performance and Variability Optimization Strategies in a Sub-200mV, 3.5pJ/inst, 11nW Subthreshold Processor," VLSI Circuits, 2007 IEEE Symposium on , vol., no., pp.152,153, 14-16 June 2007

[9] Borkar, S.; Karnik, T.; Narendra, S.; Tschanz, J.; Keshavarzi, A.; De, V., "Parameter variations and impact on circuits and microarchitecture," Design Automation Conference, 2003. Proceedings , vol., no., pp.338,342, 2-6 June 2003

[10] Gadamsetti, B.; Singh, A.D.; , "Current Sensing Completion Detection for high speed and area efficient arithmetic," *Circuits and Systems (APCCAS), 2010 IEEE Asia Pacific Conference on* , vol., no., pp.240-243, 6-9 Dec. 2010.

[11] Uppu, R.T.; Uppu, R.K.; Singh, A.D.; Chatterjee, A., "A High Throughput Multiplier Design Exploiting Input Based Statistical Distribution in Completion Delays," VLSI Design and 2013 12th International Conference on Embedded Systems (VLSID), 2013 26th International Conference on , vol., no., pp.109,114, 5-10 Jan. 2013

[12] Ernst, D.; Nam Sung Kim; Das, S.; Pant, S.; Rao, R.; Toan Pham; Ziesler, C.; Blaauw, D.; Austin, T.; Flautner, K.; Mudge, T.;"Razor: a low-power pipeline based on circuit-level timing speculation," *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, vol., no., pp. 7-18, 3-5 Dec. 2003.

[13] Blaauw, D.; Kalaiselvan, S.; Lai, K.; Wei-Hsiang Ma; Pant, S.; Tokunaga, C.; Das, S.; Bull, D.; , "Razor II: In Situ Error Detection and Correction for PVT and SER Tolerance," *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International* , vol., no., pp.400-622, 3-7 Feb. 2008.

[14] Fojtik, M.; Fick, D.; Yejoong Kim; Pinckney, N.; Harris, D.; Blaauw, D.; Sylvester, D.; "Bubble Razor: An architecture-independent approach to timing-error detection and correction, *"Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, vol., no., pp.488-490, 19-23 Feb. 2012.

[15] Ghosh, S.; Bhunia, S.; Roy, K.; , "CRISTA: A New Paradigm for Low-Power, Variation-Tolerant, and Adaptive Circuit Synthesis Using Critical Path Isolation," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.26, no.11, pp.1947-1956, Nov. 2007.

[16] Choudhury, M.; Chandra, V.; Mohanram, K.; Aitken, R.; , "TIMBER: Time borrowing and error relaying for online timing error resilience," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010* , vol., no., pp.1554-1559, 8-12 March 2010.

[17] Gupta, M.S.; Rivers, J.A.; Bose, P.; Gu-Yeon Wei; Brooks, D., "Tribeca: Design for PVT variations with local recovery and fine-grained adaptation," Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on , vol., no., pp.435,446, 12-16 Dec. 2009

[18] J Sartori, R Kumar; ,"Characterizing the voltage scaling limitation s of Razor-based Designs"

[19] Austin, T.; Bertacco, V.; Blaauw, D; Mudge, T., "Opportunities and challenges for better than worst-case design," Design Automation Conference, 2005. Proceedings of the ASP-DAC 2005. Asia and South Pacific , vol.1, no., pp.I/2,I/7 Vol. 1, 18-21 Jan. 2005

[20] Dreslinski, R.G.; Wieckowski, M.; Blaauw, D; Sylvester, D; Mudge, T., "Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits," Proceedings of the IEEE , vol.98, no.2, pp.253,266, Feb. 2010

[21] J.M. Rabaey, *Digital Integrated Circuits*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[22] Borkar, S.; Karnik, T.; Vivek De; , "Design and reliability challenges in nanometer technologies," *Design Automation Conference, 2004. Proceedings. 41st* , vol., no., pp.75, 7-11 July 2004.

[23] Saha, K. S., Modeling Process Variability in Scaled CMOS Technology, IEEE Design and Test of Computers, Vol. 27, Number 2, pp. 8-16, March/ April 2010.

[24] Victoria Wang, Kanak Agarwal, Sani R. Nassif, Kevin J. Nowka, Dejan Markovic , A Design Model for Random Process Variability Proceeding 2008 ISQED pp. 734-737

[25] J. Patel. CMOS process variations: A critical operation point hypothesis, 2008.

[26] R. Kumar. Stochastic processors. In NSF Workshop on Science of Power Management, 2009.

[27] S. Narayanan, G. Lyle, R. Kumar, and D. Jones. Testing the critical operating point (cop) hypothesis using fpga emulation of timing errors in over-scaled soft-processors. In In SELSE 5 Workshop - Silicon Errors in Logic - System Effects, 2009.

[28] T. D. Burd, S. Member, T. A. Pering, A. J. Stratakos, and R. W. Brodersen. A dynamic voltage scaled microprocessor system. IEEE Journal of Solid-State Circuits, 35:15711580, 2000.

[29] V. Gutnik and A. Chandrakasan, An efficient controller for variable supply-voltage low power processing. pages 158159, Jun 1996.

[30] S. Dhar, D. Maksimovic, and B. Kranzen. Closed-loop adaptive voltage scaling controller for standard-cell asics. In ISLPED '02: Proceedings of the 2002 international symposium on Low power electronics and design, pages 103107, New York, NY, USA, 2002. ACM.

[31] T. Kehl. Hardware self-tuning and circuit performance monitoring. pages 188192, Oct 1993.

[32] I. Corporation. Enhanced intel speed step technology for the intel pentium M processor, 2004.