High-Performance, Energy-Efficient CMOS Arithmetic Circuits

by

Pierce I-Jen Chuang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2014

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In a modern microprocessor, datapath/arithmetic circuits have always been an important building block in delivering high-performance, energy-efficient computing, because arithmetic operations such as addition and binary number comparison are two of the most commonly used computing instructions. Besides the manufacturing CMOS process, the two most critical design considerations for arithmetic circuits are the logic style and micro-architecture. In this thesis, a constant-delay (CD) logic style is proposed targeting full-custom high-speed applications. The constant delay characteristic of this logic style (regardless of the logic type) makes it suitable for implementing complicated logic expressions such as addition. CD logic exhibits a unique characteristic where the output is pre-evaluated before the inputs from the preceding stage are ready. This feature enables a performance advantage over static and dynamic domino logic styles in a single cycle, multi-stage circuit block. Several design considerations including timing window width adjustment and clock distribution are discussed. Using a 65-nm general-purpose CMOS technology, the proposed logic style demonstrates an average speedup of 94% and 56% over static and dynamic domino logic, respectively, in five different logic gates. Simulation results of 8-bit ripple carry adders conclude that CD logic is 39% and 23% faster than the static and dynamic-based adders, respectively. CD logic also demonstrates 39% speedup and 64% (22%) energy-delay product reduction from static logic at 100% (10%) data activity in 32-bit carry lookahead adders. To confirm CD logic's potential, a 148 ps, single-cycle 64-bit adder with CD logic implemented in the critical path is fabricated in a 65-nm, 1-V CMOS process. A new 64-bit Ling adder micro-architecture, which utilizes both inversion and absorption properties to minimize the number of CD logic and the number of logic stage in the critical path, is also proposed. At 1-V supply, this adder's measured worst-case power and leakage power are 135 mW and 0.22 mW, respectively. A single-cycle 64-bit binary comparator utilizing a radix-2 tree structure is also proposed. This comparator architecture is specifically designed for static logic to achieve both low-power and high-performance operation, especially in low input data activity environments. At 65-nm technology with 25% (10%) data activity, the proposed design demonstrates 2.3× (3.5×) and 3.7× (5.8×) power and energy-delay product efficiency, respectively. This comparator is also 2.7× faster at iso-energy (80 fJ) or 3.3× more energy-efficient at iso-delay (200 ps)

than existing designs. An improved comparator, where CD logic is utilized in the critical path to achieve high performance without sacrificing the overall energy efficiency, is also realized in a 65-nm 1-V CMOS process. At 1-V supply, the proposed comparator's measured delay is 167 ps, and has an average power and a leakage power of 2.34 mW and 0.06 mW, respectively. At 0.3-pJ iso-energy or 250-ps iso-delay budget, the proposed comparator with CD logic is 20% faster or 17% more energy-efficient compared to a comparator implemented with just the static logic.

Acknowledgements

First of, I would like to thank Dr. Manoj Sachdev for his continuous mentoring, and guidance as my research supervisor. His unconditional support and faith in me, without a doubt, has allowed me to go beyond my limit. I am also thankful for my co-supervisor, Dr. Vincent Gaudet, who has provided tremendous insights, helped on several research projects, and proofread my papers numerous times. I would also like to thank Dr. Mark Aagaard, Dr. Jim Martin, and Dr. David Nairn for serving on my Ph.D committee and providing helpful suggestions to further improve the quality of this thesis. A special appreciation goes to Dr. Majid Ahmadi from University of Windsor for serving as an external examiner.

I can not express enough gratitude to David Li, who has always been a good friend and most importantly, as a mentor who first introduces me to this group and provides countless helps on various projects. I would also like to thank all the members of CMOS Design and Reliability Group for all the valuable discussions, and Phil Regier for the numerous technical issues I have encountered over the years

I would like to acknowledge the financial supports from National Sciences and Engineering Research Council of Canada, Ontario Graduate Scholarship program, and Waterloo Institute for Nanotechnology Nanofellowships program. A special thanks goes to Dr. Glenn Gulak's research group at University of Toronto, and especially Mario Milicevic, for all the valuable helps on the baseband communication circuit research project as well as chip testing. I would also like to acknowledge the technical and fabrication support from CMC Microsystems.

Finally, and most important of all, I would like to thank my parents and my wife for their support, encouragement, and understanding throughout my academic career at University of Waterloo.

Dedication

To my beloved wife, Xiaoxia, and parents.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ALU Arithmetic Logic Unit

ANT All-N-Transistor

BCL Bitwise Competition Logic

CAD Computer-Aided Design

CCD Compound Constant-Delay

CD Constant-Delay

CDL Compound Domino Logic

CLA Carry Lookahead Adder

CMOS Complementary Metal Oxide Semiconductor

CPU Central Processing Unit

DCDE Digitally Controlled Delay Element

EDP Energy-Delay Product

FA Full Adder

FO4 Fanout-of-4

FOM Figures-Of-Merit

FTL Feedthrough Logic

IC Integrated Circuits

LB Logic Block

MSB Most Significant Bit

MUX Multiplexer

nMOS N-Channel Metal Oxide Semiconductor

OPL Output Prediction Logic

PDN Pull-Down Network

PDP Power-Delay Product

PLL Phase-Locked Loops

pMOS P-channel Metal Oxide Semiconductor

PTL Pass Transistor Logic

PUN Pull-Up Network

PVT Process-Voltage-Temperature

RCA Ripple Carry Adder

SCL Source-Coupled Logic

SRAM Static Random Access Memory

TB Timing Block

TG Transmission Gate

TSMC Taiwan Semiconductor Manufactuering Company

TT Typical-Typical

VLSI Very-Large-Scale Integration

Chapter 1

Introduction

Thanks to integrated circuits (ICs), countless electronic devices, ranging from portable electronic devices to supercomputers that only existed in our imagination a few decades ago, can be realized and have become a life necessity in modern society. This is thanks to Moore's Law, which states that the number of transistors for a given die area doubles every eighteen months, as shown in Fig. 1.1 [1]. The continuous scaling of complementary metal oxide semiconductor (CMOS) technology is primarily responsible for such a rapid increased integration. Besides the increased number of transistors that can be packed in a given area, numerous advantages, including both delay and energy reduction, have been obtained as a result of the rapid shrinkage of a transistor's minimum dimensions. However, as CMOS technology further scales down to allow faster IC with less energy consumption, continuous circuit innovation, in particular, logic implementation, is a necessity in order to avail its benefits.

In the past decade, as the transistor dimension gradually approaches the fundamental physical limit, the scaling of CMOS technology has slowed down. Moreover, traditional constant field scaling, where all device dimensions as well as the supply voltage are reduced by a factor of 1/S, can no longer be applied. In fact, voltage supplies have remained approximately the same for several technology nodes and are expected to be approximately the same in the future [2]. This implies that the power reduction as a result of implementing a given design using an advanced technology node is expected to be more difficult to achieve,

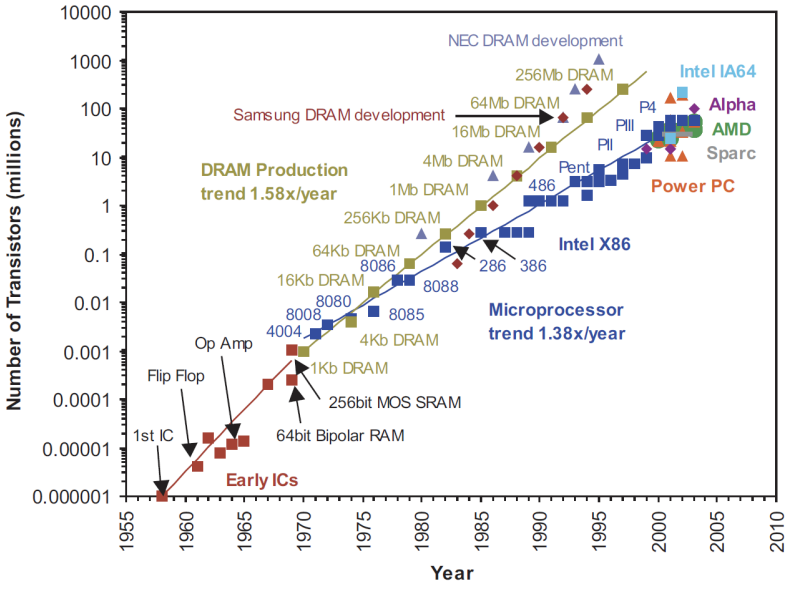


Figure 1.1: Transistors per integrated circuit trends.

because of the quadratic relationship between supply voltage and power dissipation.

For digital circuits, the shrinkage in transistor dimension is beneficial for both delay and power, since a smaller transistor implies less capacitance to be charged and discharged. While this benefit is expected to continue with device shrinkage; interconnect along with parasitic capacitance, on the other hand, has become the dominant factor in contributing the capacitive load on chip and does not scale well with each technology node [3, 4]. In other words, circuit designers are expected to observe less delay and power reduction as they move to a new technology node. This also increases both the difficulties and the effort for circuit designers to accurately evaluate and simulate the designs, as the simulation result discrepancies between schematic and post-layout simulations, where interconnect and parasitic capacitance are taken into consideration, have drastically increased. As CMOS device continues to scale down, this problem is expected to become even more prominent, and may result in a longer development time due to the increased complexity in design validation.

Despite both the supply voltage reduction and device dimension shrinkage, the total

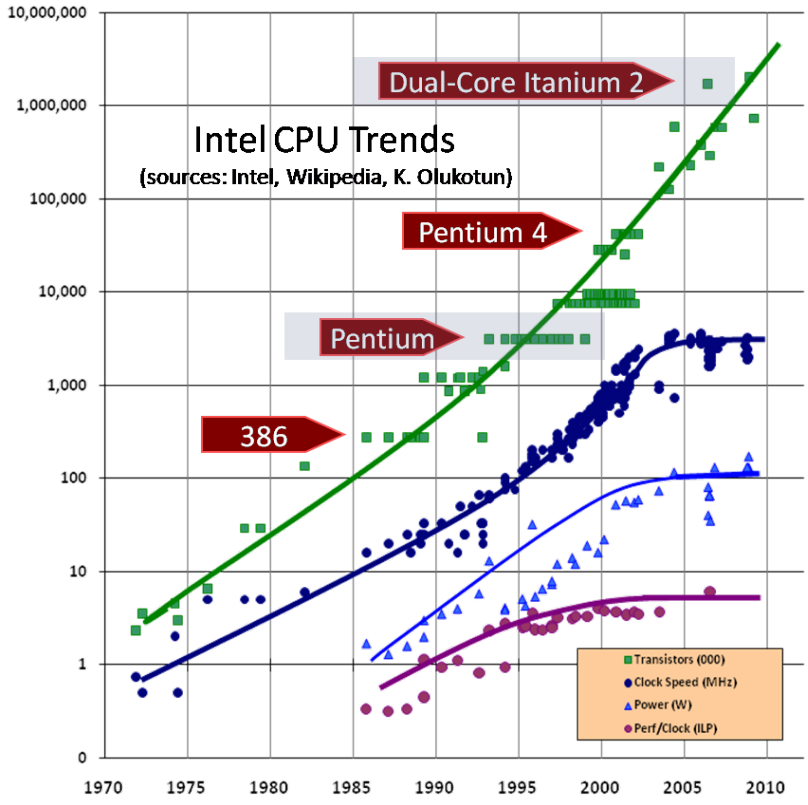


Figure 1.2: Intel CPU history trends [5].

power consumption of high-performance microprocessor remain approximately the same. In order to constrain the power budget and power density within an acceptable level without implementing expensive solutions such as liquid-cooling, the frequency scaling has stopped in recent years (Fig. 1.2). Instead, modern high-performance processors have transformed from high-speed, single-core structures to energy-efficient, many-core architectures to strive for more performance enhancement utilizing parallelism.

1.1 Motivation

Energy-efficiency has become one of the most important figures-of-merit (FOM) for digital circuits, primarily because of the continuous demands for longer battery life for portable electronic devices such as smart phones. Furthermore, the recent explosive demands of various processing-intensive tasks such as high-definition video streaming and decoding on mobile phones have created the need for energy-efficient digital arithmetic circuits without sacrificing the performance. In particular, digital arithmetic circuits such as adders and comparators are extremely important. Addition has always been one of the most commonly used arithmetic operations [6–9]. Binary comparator is one of the most fundamental components in digital systems with many applications such as the decoding of the x86 instruction sets, the renaming of the register files in a superscalar system, and the number magnitude comparison in an arithmetic logic unit. When designing high-performance, energy-efficient digital arithmetic circuits, the two most important design considerations for digital circuit designers are the logic styles and the architectures. In this thesis, an in-depth analysis and comparison of various logic styles and arithmetic architectures will be discussed.

1.2 Outline

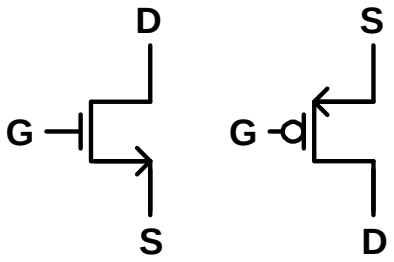
The rest of the thesis is organized in the following manner. Chapter 2 discusses the operations of CMOS transistors, digital circuit design FOM, and various CMOS logic families available for digital circuit designers. In Chapter 3, a constant-delay logic style is proposed with analytical and simulation results in various applications to demonstrate its performance advantage over existing designs. A 64-bit adder architecture with CD logic is described and its silicon measurement results are presented in Chapter 4. Chapter 5 introduces a 64-bit binary comparator architecture and demonstrates its energy efficiency over other comparator designs in three CMOS processes. An improved 64-bit comparator with CD logic is also proposed with silicon results in a 65-nm CMOS process to reveal its performance advantage. Finally, Chapter 6 concludes with future work that lies ahead in this line of research.

Chapter 2

Background

2.1 Complementary Metal Oxide Semiconductor Transistors

Conventional CMOS technology contains two types of transistors, n-channel (nMOS) and p-channel (pMOS), on the same silicon material. Both types of transistors are crucial for digital circuit designers, especially in logic implementation, since nMOS and pMOS transistors are responsible for delivering logic "0" and "1", respectively. The basic schematics of nMOS and pMOS transistors are shown in Figure 2.1. nMOS and pMOS devices are typically distinguished by two different types of representations. In the area of digital design, pMOS transistor is often represented with a circle at its gate terminal (g) while nMOS transistor does not have the circle. In terms of analog circuits, nMOS and pMOS transistors are identified by an arrow pointing away and toward the gate terminal, respectively, to indicate the direction of the current flow. Only the first method of representation is adapted in the rest of the thesis, since this thesis is mainly focused on the digital logic design. In the following section, detail descriptions of nMOS transistor's behaviour under different conditions are provided while pMOS transistor follows the same equations and arguments.



(a) (b)

Figure 2.1: Schematic of (a) nMOS and (b) pMOS transistor.

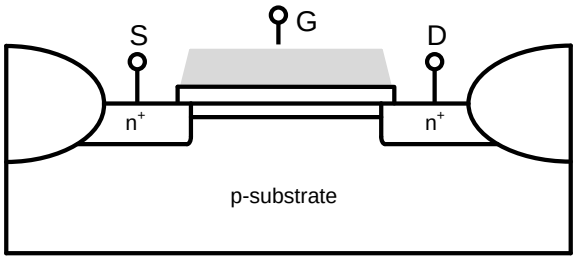


Figure 2.2: Cross-section of a nMOS device.

2.2 nMOS Transistor

A cross-section of an nMOS device on a silicon wafer is shown in Figure 2.2. Depending on the gate to source (s) voltage difference (V_{gs}) and drain (d) to source voltage difference (V_{ds}), the transistor can work at three different regions, namely subthreshold, triode (linear) and saturation regions.

2.2.1 Subthreshold Region

Subthreshold (weak-inversion) conduction takes place when V_{gs} is less than the threshold voltage (V_t). In this region, the transistor is considered "off", and the weak current conducting in the transistor channel is typically considered as leakage current. However, in recent years it has been demonstrated that circuit blocks operating in the subthreshold re-

gion with a supply voltage scaled near or below V_t can achieve significant energy-efficiency over super-threshold designs and are suitable for applications where delay performance is not the primary concern [10, 11]. The current in this region can be approximated by the following expression:

$$I_d = I_s e^{\frac{V_{GS}-V_t}{nkT/q}} \left(1 - e^{-\frac{V_{DS}}{kT/q}}\right) \tag{2.1}$$

where I_s and n are empirical parameters with n typically in the range of 1 to 1.5, kT/q is the thermal voltage and is equal to $26mV$ at 300K. Based on Equation 2.1, the sub-threshold current is exponentially dependent on V_{GS}. Moreover, if V_{DS} is sufficiently large (> 100mV), then $e^{\frac{V_{DS}}{kT/q}}$ can be neglected and the current is now independent of V_{DS}. This suggests that the transistor behaves like a current source, generating a constant current which is entirely dependent on V_{GS}.

2.2.2 Triode (Linear) Region

An nMOS transistor enters this region when $V_{GS} > V_t$ and $V_{DS} < V_{GS} - V_t$. The current in this region can be approximated by the following expression:

$$I_d = \mu_n C_{ox} \frac{W}{L} \left[(V_{GS} - V_t) V_{DS} - \frac{{V_{DS}}^2}{2} \right] \tag{2.2}$$

where μ_n is the charge-carrier effective mobility and C_{ox} is the gate oxide capacitance per unit area. $\mu_n C_{ox}$ is also known as the process transconductance and is often denoted as k'_n.

In this region the transistor is turned on and a channel has been created which allows current to flow between the source and drain terminal. When V_{DS} is small enough, $\frac{{V_{DS}}^2}{2}$ from Equation 2.2 can be neglected and the current is linearly proportional to V_{GS}. Hence the transistor behaves like a resistor in this region.

2.2.3 Saturation Region

When $V_{GS} > V_t$ and $V_{DS} > V_{GS} - V_t$, the transistor is operating at the saturation region. In this case, the current is no longer a linear function of V_{DS}; instead, it now has a squared dependency with respect to V_{GS}. The current in this region can be described as:

$$I_d = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{GS} - V_t)^2 \tag{2.3}$$

Equation 2.3 suggests that the transistor in the saturation region behaves like a perfect current source. The current flowing between source and drain terminal is constant regardless of the value of V_{DS} and is only dependent on V_{GS}. This simplified assumption is not entirely correct since V_{DS} modulates the current as well. A more accurate description of the MOS transistor current in the saturation region then becomes:

$$I_d = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{GS} - V_t)^2 (1 + \lambda V_{DS}) \tag{2.4}$$

where λ is an empirical parameter called channel-length modulation.

As transistor's channel length continues to shrink due to technology scaling, current behaviours begin to deviate considerably from Equation 2.4 and new physical phenomena, also known as short-channel effects, begin to influence transistor's current behaviour. Among all the short channel effects, the main culprit for this deviation is the velocity saturation.

Velocity Saturation

Velocity saturation happens due to a high lateral electric field between source and drain terminal. Consider the empirical Equation 2.5 which states that the average carrier drift velocity is directly proportional to carrier mobility μ and electric field E, which is the voltage difference between drain and source terminal (V_{DS}) divided by the channel length L.

$$\nu_n = \mu E \tag{2.5}$$

This simplified assumption only holds at low electric field. At high lateral field strength, the average carrier drift velocity does not follow this linear model but saturates at a constant value due to carrier scattering. In this regard, increasing electric field, hence the voltage difference between drain and source, no longer improves transistor's current output. Instead, the transistor's current is saturated at I_{DSAT}, and the current behaviour is better approximated by the following expression:

$$I_d = \mu_n C_{ox} \frac{W}{L} \left[(V_{GS} - V_t) V_{DSAT} - \frac{{V_{DSAT}}^2}{2} \right] (1 + \lambda V_{DS}) \tag{2.6}$$

where V_{DSAT} is the velocity saturation voltage. Equation 2.5 and 2.6 lead to three observations:

- Velocity saturation is more prominent in short-channel devices because at shorter channel length L, lower V_{DS} is required before the carrier drift velocity v_n saturates.
- Shorter channel devices therefore experience an extended saturation region, and tend to operate more in saturation conditions.
- The saturation current I_{DSAT} is linearly dependent on the gate to source voltage V_{GS} in the velocity saturation region instead of the squared dependence in the original saturation current expression. This reduces the amount of current a transistor can deliver for a given V_{GS}.

2.3 Digital Design Performance Merits

2.3.1 Delay

The delay determines how fast, hence the operating frequency, a particular digital circuit can respond when an input changes and is one of the most important performance merits for digital circuit designers. The delay metric can be further defined as the propagation delay, t_p, and is measured between the 50% transition points of the input and output waveforms, as shown in Figure 2.3. t_{pHL} defines the response time of a system for a high (input) to low

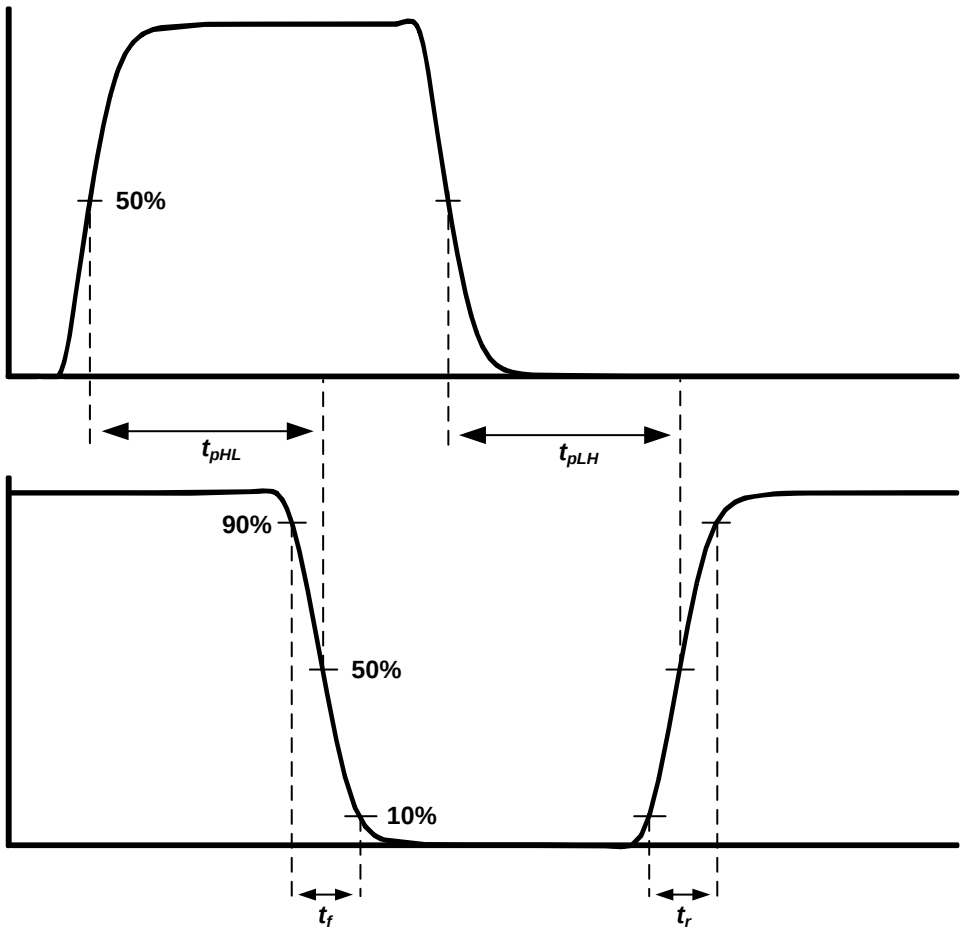


Figure 2.3: Definition of propagation delay and rise and fall time.

(output) transition while t_{pLH} refers to a low to high transition. The propagation delay t_p is defined as the average of the two and is expressed as:

$$t_p = \frac{t_{pLH} + t_{pHL}}{2} \tag{2.7}$$

In addition, the propagation delay is also a function of the slopes of the input and output signals, as shown in Figure 2.3. Therefore, two more FOMs, namely the rise and fall time, t_r and t_f, respectively, are introduced to measure the transition time between 10% and 90% of the rise and fall waveforms, respectively.

2.3.2 Dynamic Power Dissipation

Another important performance metric for digital system is the power consumption. Power consumption measures how much power a particular digital circuit needs to consume to perform an operation. The power consumption of MOS transistors can be further categorized as dynamic and static (leakage) power consumption.

Dynamic power consumption is defined as the energy a transistor requires to charge a capacitor. For instance, each time a capacitor is charged through the pMOS transistor, its voltage rises from GND to VDD and a certain amount of energy is drawn from the power supply. The amount of energy E_{supply} taken from the supply during this transition can be derived by integrating the instantaneous power over the period of transition:

$$E_{supply} = \int_0^\infty i_{VDD}(t) V_{DD}\, dt = V_{DD} \int_0^\infty C_L \frac{dv_{out}}{dt}\, dt = C_L V_{DD} \int_0^{V_{DD}} dv_{out} = C_L {V_{DD}}^2 \tag{2.8}$$

while the amount of energy E_C stored on the capacitor at the end of the transition is

$$E_C = \int_0^\infty i(t) V_{out}\, dt = \int_0^\infty C_L \frac{dv_{out}}{dt} v_{out}\, dt = C_L \int_0^{V_{DD}} v_{out}\, dv_{out} = \frac{C_L {V_{DD}}^2}{2} \tag{2.9}$$

where E_C is the amount of energy stored in the capacitor at the end of the transition and C_L is the load capacitor. Equation 2.8 and 2.9 suggest that only half of the energy supplied by the power source is stored in C_L, while the other half has been dissipated by the transistor, regardless of the transistor dimension. Also, E_C has a squared dependency on the supply voltage V_{DD}, hence reducing the supply voltage is the most effective way to decrease the dynamic energy consumption.

The capacitor charging event only takes place when the device is switched. In order to compute the dynamic power consumption, it is necessary to take into account how often the device is switched. Denotes the number of times a device is switched per second as f_{switch}, then the dynamic power consumption is given by:

$$P_{dynamic} = C_L {V_{DD}}^2 f_{switch} \tag{2.10}$$

Since $P_{dynamic}$ is linearly proportional to f_{switch}, increasing the f_{switch} leads to a higher dynamic power consumption. In addition, as explained in the following section, circuit style, such as dynamic logic, suffers from higher $P_{dynamic}$ due to higher data activity (f_{switch}) compared to static logic.

2.3.3 Static Power Dissipation (Leakage)

The other major source of power consumption is the static power dissipation and is expressed by the following relation:

$$P_{static} = I_{static}V_{DD} = I_{leakage}V_{DD} \tag{2.11}$$

where $I_{leakage}$ is the leakage current (subthreshold current) that flows between the supply rails when the device is not switching (turned off) and can be approximated by the subthreshold current expression shown in Equation 2.1. In larger CMOS manufacturing processes such as 0.25μm static power dissipation is not a major concern because dynamic power dissipation dominates the overall energy consumption. However, as technology scales down to nano-scale, the thin oxide thickness (for aggressive nanometer CMOS process, the thickness can be as thin as only few hydrogen molecules) along with other short channel effects significantly increases the leakage.

2.4 Logic Implementations: Circuit Families

2.4.1 Static Logic

Static logic is the most widely used logic style in CMOS technology and its basic structure is shown in Figure 2.4. It consists of a nMOS pull-down network (PDN) and a pMOS pull-up network (PUN). The primary advantages of static logic are robustness, low power dissipation especially at low data activity factor, and adequate performance with no static power dissipation. Its most distinct characteristic is that at any given time, the gate output is connected to either VDD or GND via a low-resistance path. While this unique feature

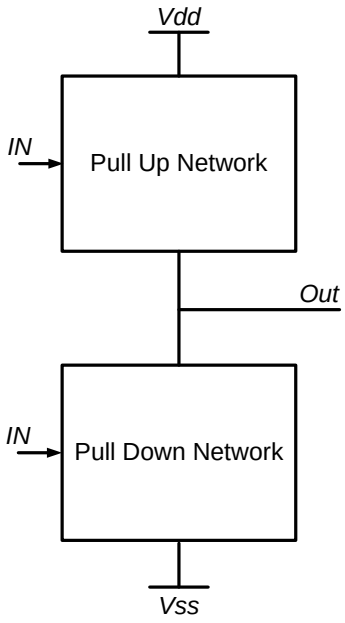
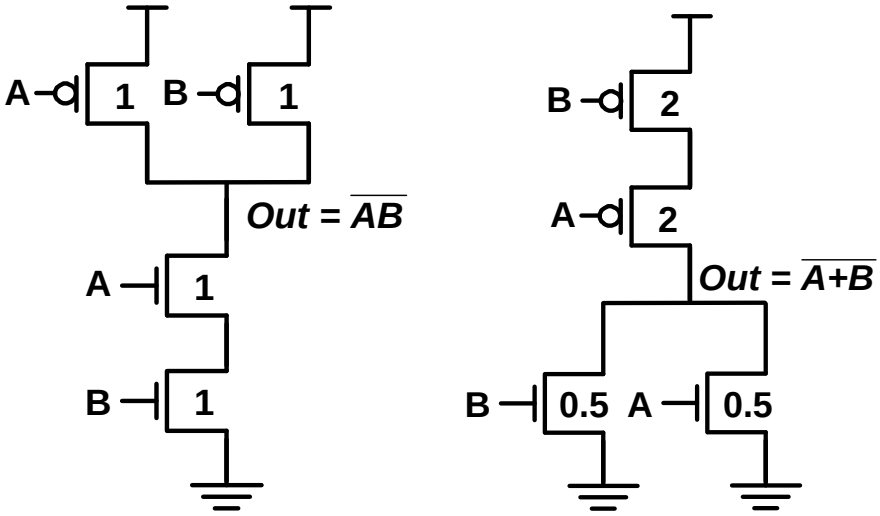


Figure 2.4: Static logic as a combination of a pull up and down network.

ensures static logic's robustness, it is also a major drawback since static CMOS requires both nMOS and pMOS transistors on each input. During a falling output transition, pMOS transistors do not contribute to the pull-down transition current but only add significant capacitance. Hence, static CMOS has a relatively large logical effort and area penalty and is slow when implementing complicated logic expression such as 4-input XOR. The schematic of a two-input static NAND and NOR gate is shown in Figure 2.5. For the two-input NAND gate, Out is connected to VDD when either A or B is logic "0" and is only connected to GND when both A and B are at logic "1". On the other hand, Out is connected to VDD only when both A and B are logic "0" and is connected to GND for the rest of the time in a two-input NOR gate. At every point in time, Out is not floating and is computed as the value of the Boolean function implemented by the PDN and PUN. The PDN and PUN are implemented using nMOS and pMOS devices because they can pass strong logic "0" and "1" respectively. pMOS devices are typically sized up two times larger than nMOS devices to provide equal rise and fall delay due to lower hole mobility. Therefore, pMOS transistors have to be up sized four times larger than nMOS transistors

to achieve equivalent rise and fall delay in the case of a two-input NOR gate. The up-sized pMOS transistors contribute additional capacitance for both transitions, while only helping the rise delay. Therefore, pMOS devices become the area bottleneck for static CMOS logic style when implementing NOR gate (pMOS devices in series). Furthermore, the up-sizing technique provides diminished rising delay improvement due to self-loading effect, since the additional drain capacitance introduced by up-sizing gradually offsets the performance enhancement contributed by higher pull-up current.

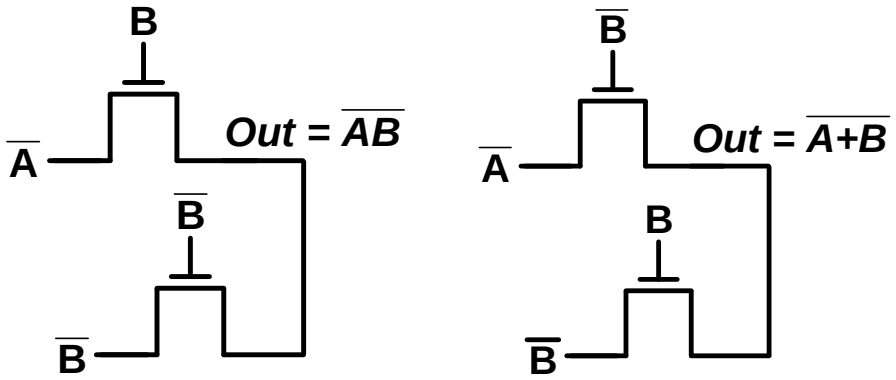


(a) (b)

Figure 2.5: Schematic of a two-input static (a) NAND gate and (b) NOR gate.

2.4.2 Pass Transistor Logic

In the previous section, static CMOS logic is described where the logic inputs are only applied to the gate terminals of the transistors. In this section, another conventional logic style is introduced where inputs are also applied to the source/drain diffusion terminals of the transistors to reduce area and power while not sacrificing performance. This type of



(a) (b)

Figure 2.6: Schematic of a two-input pass transistor (a) NAND gate and (b) NOR gate.

logic style is called pass transistor logic (PTL). Figure 2.6 depicts the schematic of a two-input pass transistor NAND and NOR gate. Compared to static logic, fewer transistors (hence lower capacitance) are required to implement the same function. For instance, the implementation of static NAND gate requires four transistors while PTL NAND gate only requires two transistors. Nevertheless, PTL requires complementary signals which are often generated using additional inverters in a single-ended system. This introduces additional hardware overhead and compromises the area advantage. In realistic designs the complementary signals are often shared among several pass transistor gates, hence the additional area overhead can be minimal, depending on the type of application.

The output of PTL should be protected by an inverter (buffer) before driving the next stage load. In other words, PTL logic cannot be cascaded by connecting the output of a PTL to the gate input of another PTL. This is because nMOS transistor can only deliver a weak logic "1" ($VDD - V_{th}$). Consider a pass transistor schematic as shown in Figure 2.7, where signal A, B and C all come from other pass transistor gates directly without protective inverters. If all signals are logic "1", then the maximum voltage at both X and Out will be $VDD - 2V_{th}$, since both A and B only have a maximum voltage swing of $VDD - V_{th}$. Hence in real practice an inverter is always inserted at the output of every PTL before driving the next stage logic.

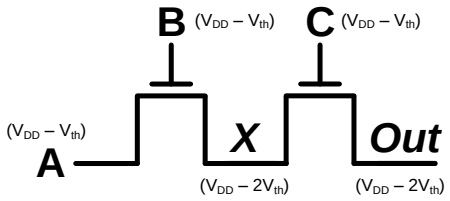


Figure 2.7: Multiple threshold voltage drops at the output of a pass transistor gate.

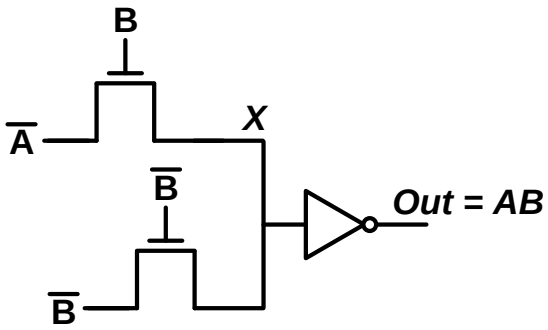


Figure 2.8: Schematic of a two-input pass transistor AND gate.

However, the addition of a protective inverter at the output of PTL causes direct power dissipation because the inverter's pMOS transistor is not completely turned off. Consider a two-input pass transistor AND gate with an inverter at the output, as shown in Figure 2.8. When A is at logic "0" and B at is logic "1", X is charged up to VDD - V_{th} and Out is discharged to GND through the inverter's nMOS transistor. While PTL successfully evaluates in this case, the gate to source voltage (V_{gs}) of the inverter's pMOS transistor is equal to V_{th} instead of zero. In this regard, the pMOS transistor is not completely off, and a direct current path exists.

A common solution to the voltage drop problem is the use of a level restorer. Figure 2.9 illustrates the schematic of a two-input AND gate with a feedback level restorer circuit. The single pMOS transistor's gate is connected to the output of the inverter and the drain terminal is connected to the input of the inverter. Consider the situation where A is at logic "0" and B is at logic "1". X is initially charged up to VDD - V_{th} and Out is discharged

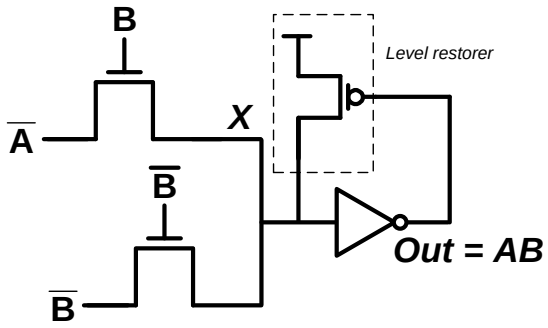


Figure 2.9: Schematic of a two-input pass transistor AND gate with feedback level restorer.

to GND. Once *Out* is at GND, the level restorer pMOS transistor turns on, continues to charge up X to full VDD, and eliminates the problematic static current path. Furthermore, no direct current path can exist through the level restorer and the pass transistor logic, since the pMOS transistor is only active when *Out* is low, which implies that X must be at logic "1".

While this solution mitigates the problem of static power dissipation, it contributes additional capacitance, adds layout complexity, and most importantly, makes pass transistor logic a ratioed logic. When X is to make a logic "1" to "0" transition, the nMOS pull down path now has to fight against the pMOS level restorer because initially the restorer is on. Therefore, the nMOS pull-down path must be stronger than the restorer and careful transistor sizing is necessary in order to make the circuit function properly.

2.4.3 Transmission Gate Logic

Another widely used solution to mitigate the threshold voltage drop problem associated with pass transistor logic is the use of transmission gates (TG). In a TG design, a pMOS device is often placed in parallel with a nMOS device to deliver both strong logic "1" and "0". The schematic of a TG multiplexer is shown in Figure 2.10. This configuration does not have V_{th} drop problem because the parallel pMOS device provides full voltage swing, at the expense of additional transistors and control signals. If transmission gates are used, it is a common practice to size both pMOS and nMOS transistors approximately the same

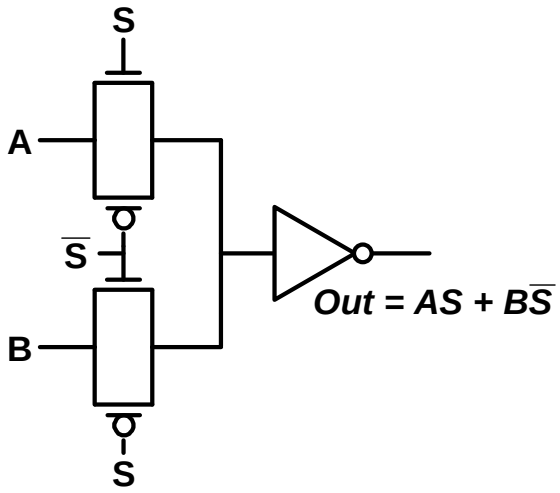


Figure 2.10: Schematic of a transmission gate based multiplexer.

width rather than using a double-width pMOS to nMOS size ratio. This is because both transistors are passing the signal in parallel, and the primary objective of the pMOS device is to provide a full voltage swing.

2.4.4 Dynamic & Compound Domino Logic

The invention of the dynamic logic in the 1980s is one of the answers to the request of ever increasing IC operating speed as it allows designers to implement high-performance circuit block, i.e., arithmetic logic unit (ALU), at an operating frequency that the traditional static and pass-transistor CMOS logic styles are difficult to achieve [12]. A generalized schematic of a dynamic gate with footer CLK transistor is shown in Figure 2.11. The operation of dynamic logic is as follows: When CLK is low (precharge period), transistor $M1$ is on, and nMOS PDN is off because $M2$ is off. X is charged to VDD by transistor $M1$ and Out is maintained at GND. Dynamic logic enters evaluation period when CLK rises to high. In this case, depending on the input patterns two possible scenarios can take place. If nMOS PDN is off, X will be floating because both $M1$ and PDN are off. Therefore, a small pMOS keeper ($M3$) is required to fight against the leakage and to help

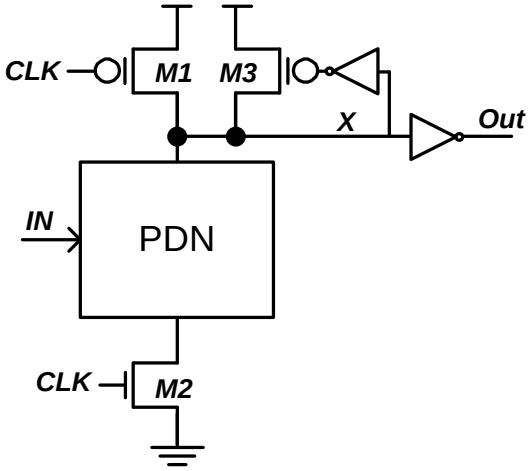


Figure 2.11: Dynamic logic with a footer transistor.

maintaining the voltage at node X to be VDD. On the other hand, if nMOS PDN is on, then X discharges to GND and Out is charged up to VDD via the inverter. Dynamic logic does not have the problem of static power dissipation because when X is at GND (Out is at VDD), pMOS keeper $M3$ is guaranteed to be off. When Out is discharged, it cannot be charged again until the next precharge period begins. Thus the inputs to the gate of nMOS PDN can make at most one transition during evaluation. In summary, the unique characteristics of dynamic logic are:

- The logic function is implemented with nMOS transistors only.
- The number of transistors for complicated logic expression implemented with dynamic logic is substantially lower than the static case.
- Dynamic logic has faster switching speed because fewer number of transistors (especially without any pMOS logic transistors) contributes to less load capacitance.
- It only consumes dynamic power since no static current path ever exists between VDD and GND. However, the overall power consumption can be significantly higher than the static design because of the higher switching activity.

The performance enhancement comes with several costs however, including reduced noise margin, charge-sharing noise, and higher power dissipation due to a higher data activity. In a traditional dynamic logic, an output inverter is required between dynamic logics to satisfy the data monotonicity requirement and to ensure proper logic evaluation [13]. This not only increases the overall delay but also the power consumption as well. Two variations of the dynamic logic have been proposed to mitigate this problem. NP domino, or also known as NORA domino [14] [15], replaces this inverter with pre-discharged dynamic gates using pMOS logic [16]. However, NORA is extremely susceptible to noise and has not been used extensively. Zipper domino [17] attempts to achieve the same objective by a slightly different implementation, but is widespread in the VLSI industry [16], [18].

Furthermore, dynamic logic has gradually lost its performance advantage over static logic due to the increased self-loading ratio in deep-submicron technology (65-nm and below) because of the additional nMOS CLK footer transistor (Figure 2.11). This phenomenon has been demonstrated in [19], which concludes that at processes such as 180nm and 130nm, the optimal adder architecture is radix-4 (5 transistors in series, including the footer transistor); however, radix-2 (3 transistors in series, including the footer transistor) configuration becomes optimal at 65-nm technology and beyond because the increased self-loading ratio has made radix-4 architecture slower than radix-2, even though radix-2 configuration requires more number of stages to complete the addition.

Compound domino logic (CDL) where dynamic and static CMOS gates alternating between each other mitigates the two aforementioned problems and has become the most popular logic style in high-performance circuit block, i.e., 64-bit adder in modern central processing unit (CPU) [20] [21] [22] [23]. In this design, the output inverter is replaced with a more complex inverting static CMOS gates (Figure 2.12), i.e., NAND or NOR, such that the monotonicity requirement is satisfied while conducting complex logic operations without wasting the one inverter delay [24]. Moreover, all the dynamic stages except the first stage can be footless (the footer transistor is eliminated) in CDL, thus reduce the total stack height by one. However, this implementation comes at the expense of increased power consumption due to the direct path current from VDD to GND during the precharge period [16]. While CDL offers higher performance and reduced power consumption over pure static and dynamic logic style, respectively [25], its noise margin is significantly degraded as in a

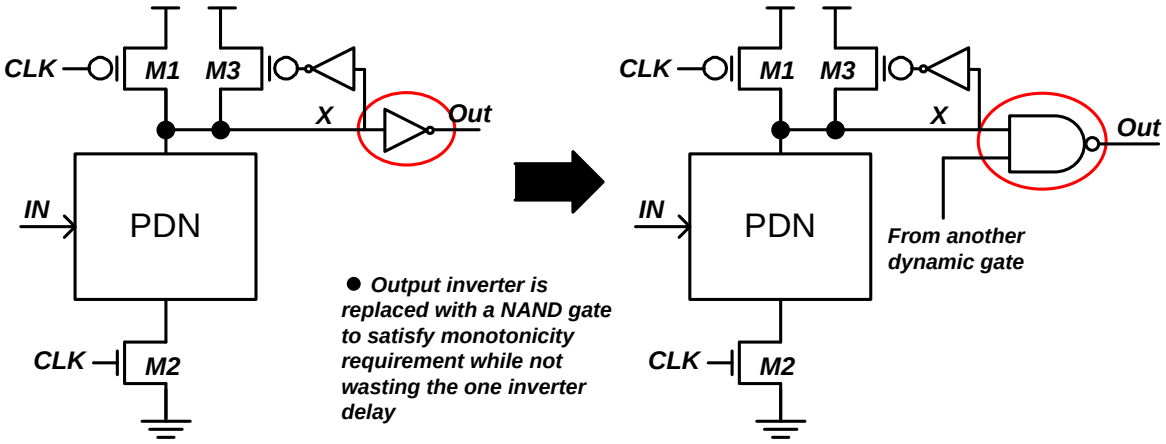


Figure 2.12: Dynamic logic vs. compound domino logic.

CDL design, the output of dynamic logic without any buffer is required to drive the next stage via a long interconnect and with other signal wires running in parallel. The crosstalk of the adjacent wire can potentially flip the state of the dynamic logic, and results in false logic evaluation [19]. As a result, extra distance among wires running in parallel has to be enforced in laying out such a design at the expense of increased total wire length. In the extreme case, power rails are placed in between adjacent wires to eliminate the crosstalk problem. This technique nevertheless, causes significant performance degradation and increased power consumption as a result of increased parasitic capacitance.

2.5 Conclusion

This section discussed the evolution of integrated circuits as well as its theory of operation and various FOMs including delay, dynamic and static power consumption. In addition, various logic styles are introduced and their advantages and disadvantages are analyzed. For a robust design, static logic is often the preferred choice. If higher performance is desired, circuit designers may choose dynamic logic or compound domino logic at the cost of power consumption.

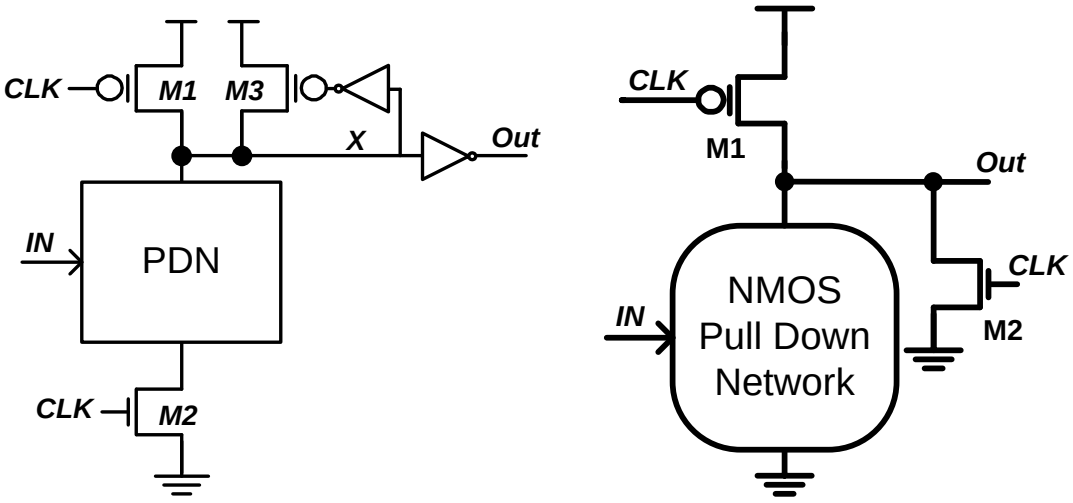
Chapter 3

Constant Delay Logic

3.1 Introduction

Significant research effort has been dedicated to explore new logic styles that go beyond dynamic domino logic and CDL. In particular, source-coupled logic (SCL) [26] has shown superior performance that is difficult to achieve using any other logic style. However, it suffers from high power dissipation due to constant current draw and its differential nature requires complementary signals. Pseudo-nMOS logic, which uses a single pMOS transistor as a pull-up device, provides high speed and low transistor count at the expense of high static power consumption as well as reduced output voltage swing [27, 28]. Output prediction logic (OPL) [29, 30] has also shown superior performance in high speed adders [31]. Nevertheless, OPL requires the generation and distribution of multi-phase clock signals with small timing separations and low skews, which is difficult to achieve. While numerous high speed logic styles have been proposed, dynamic and CDL still remain the most attractive choices when performance is the primary concern.

In recent years, a new logic family, known as feedthrough-type logic (FTL) [32] [33], has been proposed and demonstrated its high performance capability. Consider dynamic domino logic (Fig. 3.1(a)); the critical path consists of nMOS logic transistors. In FTL, however, the role of the clock and logic transistors are interchanged (Fig. 3.1(b)) and the



(a) (b)

Figure 3.1: Schematic of (a) dynamic domino logic with a footer transistor and (b) feedthrough logic (FTL).

clock transistor is now the critical path. Its basic operation is as follows: when CLK is high, the predischarge period begins and Out is pulled down to GND through $M2$. When CLK becomes low, $M1$ is on, $M2$ is off and the gate enters the evaluation period. If inputs (IN) are logic "1", Out enters the contention mode where M1 and transistors in the nMOS pull down network (PDN) are conducting current simultaneously. If PDN is off, then the output quickly rises to logic "1". In this case, FTL's critical path is always a single pMOS transistor.

Despite its performance advantage, FTL suffers from reduced noise margin, excess direct path current, and non-zero nominal low output voltage which are all caused by the contention between $M1$ and nMOS PDN during the evaluation period. Furthermore, cascading multiple FTL stages together to perform complicated logic evaluations is not practical. Consider a chain of inverters implemented in FTL are cascaded together and driven by the same clock, as shown in Fig. 3.2. When CLK is low, $M1$ of every stage turns on, and the output of every stage begins to rise. This will result in false logic evaluation at even number (i.e., 2, 4, 6, etc) stages since initially there is no contention

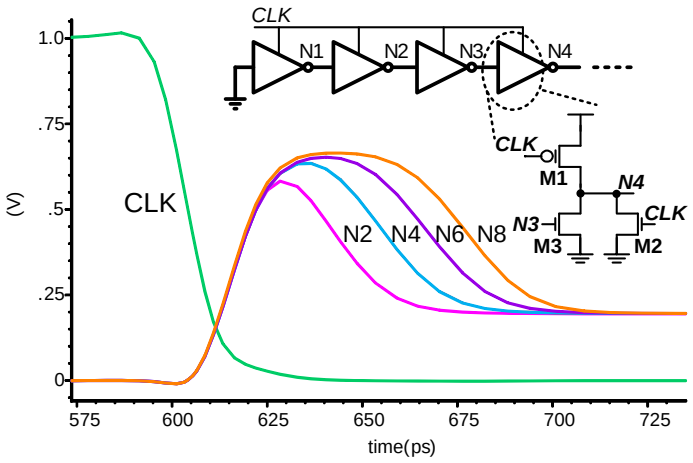


Figure 3.2: Simulated unwanted glitch at different logic depth in a chain of inverters with FTL implementation.

between $M1$ and nMOS PDN because all inputs to nMOS transistors are reset to logic "0" during the reset period. The first generation of FTL exhibits many shortcomings including excessive power dissipation and reduced noise margin. To mitigate these problems, we propose a new high-performance logic style that we call constant-delay (CD) logic. CD logic provides a local window technique and a self-reset circuit that enables robust logic operation with minimized power consumption while maintaining FTL's speed advantage. The most distinct characteristic of CD logic from previously proposed logic styles is that the delay is, to a first-order approximation, not affected by the logic expression[1]. Unlike SCL, CD logic does not require complementary signals and can be easily integrated with static and dynamic domino logics. Also, CD logic does not have the problem of constant static power dissipation similar to pseudo-nMOS. Furthermore, the clock timing requirement of CD logic is not as stringent as OPL. CD logic can achieve robust operation with optimal performance as long as CLK signal arrives earlier than the input signals. This thesis will demonstrate that CD logic has the potential to 1) outperform other logic styles with better energy-efficiency, and is particularly suitable for high-performance digital blocks; and 2) CD logic is robust under extreme process, voltage, and temperature (PVT) variations.

[1]The delay of CD logic is still a function of logic expression when all effects are considered, since a more complicated logic expression implies a larger capacitive load.

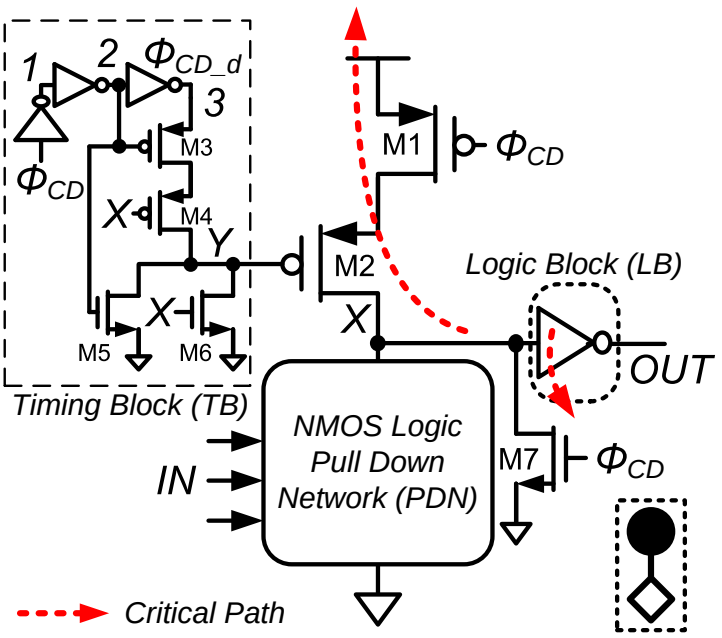


Figure 3.3: Constant delay (CD) logic block diagram.

3.2 CD Logic Operation

The proposed CD logic's schematic is shown in Fig. 3.3. *Timing block* (TB) creates an adjustable window period to reduce the static power dissipation. *Logic Block* (LB) helps to reduce the amplitude of unwanted glitch and also makes cascading CD logic feasible. Fig. 3.4 depicts the CD logic's timing diagram and flow chart. Without loss of generality it is assumed that IN signals come from dynamic domino logic gates. When ϕ_{CD} is high, CD logic pre-discharges both X and Y to GND. When ϕ_{CD} is low, CD logic enters the evaluation period and three scenarios, namely the contention mode, C-Q delay mode, and D-Q delay mode, where Q refers to OUT, can take place. The contention mode happens when ϕ_{CD} is low while IN remain at logic "1". In this case, X is at a non-zero voltage level which causes OUT to experience a temporary glitch. The duration of this glitch is determined by the local window width, which is defined as the 50% point of the ϕ_{CD} falling edge to the 50% point of the Y rising edge. C-Q delay mode takes place when IN make a transition from high to low before ϕ_{CD} becomes low. When ϕ_{CD} becomes low, X rises to

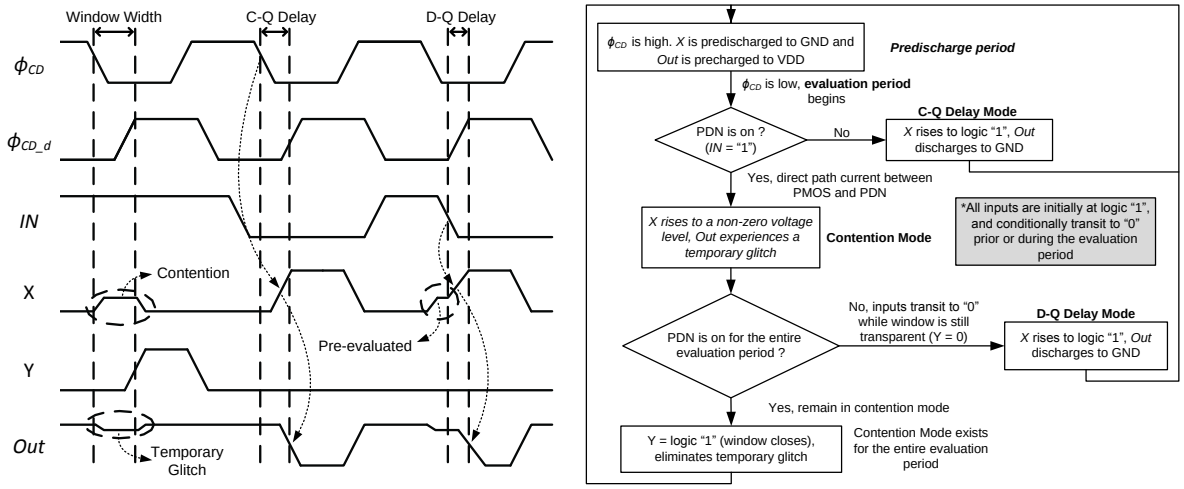


Figure 3.4: Timing diagram and flow chart of the proposed CD logic.

logic "1" and Y remains at logic "0" for the entire evaluation cycle. The delay is measured by the falling edge of both ϕ_{CD} and OUT, hence the name C-Q delay. D-Q delay utilizes the pre-evaluated characteristic of CD logic to enable high-performance operation. In this mode, ϕ_{CD} falls from high to low before IN transit; hence X initially rises to a non-zero voltage level. As soon as IN become logic "0", while Y is still low, then X quickly rises to logic "1". A race condition exists in this case between X and Y. If ϕ_{CD_d} rises much earlier than X, Y will go to logic "1", turn off *M1*, and result in a false logic evaluation. If ϕ_{CD_d} rises slightly slower than X, then Y will initially rise (thus slightly turns off *M1*) but eventually settle back to logic "0". CD logic can still perform correct logic operation in this case; however, its performance is degraded because of M1's reduced current drivability. This metastability phenomena is similar to the flip-flop setup time violation, which may result in a false logic evaluation. Therefore, it is important to maintain a sufficient window width under PVT variations. Table 3.1 presents a summary of CD logic's operations.

Compared to FTL, where the contention lasts for the entire evaluation period, TB effectively reduces CD logic's power consumption during the contention mode. The power, hence the energy saving, is primarily contributed by the cut off of the pMOS transistor to eliminate the direct path current. The normalized energy saving of the proposed window

Table 3.1: Summary of CD logic's operation.

Mode	**Scenario**	**Operation**
Predischarge	*CLK* is high	*X* and *Out* are predischarged and precharged to GND and VDD, respectively.
Contention	*IN* = "1" for the entire evaluation period.	Direct path current flows from pMOS to PDN. *X* rises to a non-zero voltage level and *Out* experiences a temporary glitch.
C-Q Delay	*IN* goes to "0" before *CLK* transits to low.	*X* rises to logic "1" and *Out* is discharged to VDD. Delay is measured from *CLK* to *Out*.
D-Q Delay	*IN* goes to "0" after *CLK* transits to low (while window is still open, i.e., Y is still "0").	*X* initially enters contention mode and later rises to logic "1". Delay is measured from *IN* to *Out*.

technique compared to the original feedthrough logic, where the contention lasts for the entire evaluation period, can be modeled as:

$$E_{saving} \approx V_{DD} I_{contention} \frac{t_{period} - t_{window\ width}}{t_{period}} \tag{3.1}$$

where $I_{contention}$ is the contention current between the pMOS transistors and nMOS PDN, t_{period} denotes the evaluation period, and $t_{window\ width}$ is the duration of the window. As depicted in Fig. 3.3, $t_{window\ width}$ is determined by the propagation delay of ϕ_{CD} to Y through three inverters and 2 pMOS transistors and can be approximated as:

$$t_{window\ width} \approx 0.69(R_1C_1 + R_2C_2 + R_3C_3 + R_4C_Y) \tag{3.2}$$

where R_1, R_2 and R_3 denote the equivalent resistance of the pMOS and nMOS transistors in the first, second, and third inverter, respectively, R_4 is the equivalent resistance of $M3$ and $M4$, and C_1, C_2, C_3, C_Y represent the lump capacitance at nodes $1, 2, 3$ and Y, respectively. In particular, the resistance can be modeled as [34]:

$$R \approx \frac{3}{4} \frac{V_{DD}}{I_{DSAT}} \left(1 - \frac{5}{6} \lambda V_{DD}\right) \tag{3.3}$$

where I_{DSAT} is the transistor' current in the velocity saturation region and λ is the channel-length modulation coefficient. From Equation 3.2, the window width is determined by four pMOS and one nMOS transistors. In the presence of global process variations, such a pMOS-dominant configuration will provide extended design margins to counter the undesirable V_t shift and to improve CD logic's reliability. Consider the case where all the pMOS transistors' threshold voltages (V_t) are shifted upwards, CD logic's delay will increase due to $M1$ and $M2$'s lower driving strength, while window width will also be stretched due to the same reason. Therefore, additional timing margin is available for CD logic during the evaluation period, and the likelihood of false logic evaluation is reduced. Similarly, if pMOS's V_t is reduced, the negative effect of shorter timing margin is compensated by $M1$ and $M2$'s higher sinking current. Another advantage of CD logic is that the internal node (X) is always connected to either VDD or GND, thus making the robustness of CD logic comparable to static logic, except during the contention mode.

CD logic eliminates the problem of false logic evaluation associated with cascaded FTL. Consider a cascaded CD logic system, the inputs to nMOS PDN are always at logic "1" when first entering the evaluation period, because X and Out are always predischarged and precharged to logic "0" and "1", respectively. Therefore, when ϕ_{CD} is low, CD gates will always first enter the contention mode and conditionally make a low-to-high transition depending on the inputs. This is not the case for the first-stage CD gate, however, as there is no guarantee that the inputs will always be at logic "1". In other words, designers need to ensure that the input signals to the first CD gate arrive earlier than the clock signal, i.e., operate in C-Q delay mode only.

3.3 CD Logic Design Considerations

3.3.1 CD Logic Transistor Sizing

The sizing of INV1-3 and M3-M6 in Fig. 3.3 are all close to the minimum size so that they do not create a huge area burden (e.g., less than 20% in a two-input AND gate). The length of INV1-3 can be altered to provide the required timing window duration based

on designer's choices. In the presence of aggravated process variation, INV1-3 can be further upsized with careful layout techniques to reduce the degree of process variation. To minimize the area overhead, only the transistors responsible for creating the window duration during the evaluation period in INV1-3 need to be upsized, while the non-critical transistors can remain close to the minimum size. M0 and M1 should also be properly sized such that the output's glitch is within an acceptable level.

CD Logic v.s. pseudo-nMOS

Both pseudo-nMOS and CD logic are ratioed circuits which rely on the correct pMOS-to-nMOS strength ratio to perform correct logic operations. The operations of pseudo-nMOS is similar to that of static CMOS logic gates where both type of transitions are possible. Therefore, pMOS transistor width is often selected to be about 1/4 the strength of the nMOS PDN as a compromise between noise margin and speed in pseudo-nMOS [16]. On the other hand, CD logic always discharges X to GND when ϕ_{CD}(CLK) is high; therefore, CD logic can be optimized for low-to-high transition speed only. Hence, pMOS clock transistors in CD logic can be up-sized larger to provide more speedup, as long as the output glitch is maintained at an acceptable level. Furthermore, pseudo-nMOS has a constant static power dissipation when its nMOS PDN is on. For CD logic, the static power dissipation occurs only during the contention mode.

3.3.2 Output Glitch

Fig. 3.5 depicts a simplified schematic of CD logic during the contention mode, where both transistors *P1* and *N1* are on simultaneously and induce a glitch voltage ΔV_1 which in turn generates another smaller glitch, ΔV_2. By design, ΔV_1 should be small (i.e., less than V_t). Hence, *P1* operates in the velocity saturation region while *N1* is in the linear region. The current equation is given as:

$$\mu_p C_{ox} \frac{W_{p1}}{L_{p1}} \left[(V_{gsp1} - V_{tp})\, V_{dsatp} - \frac{(V_{dsatp})^2}{2} \right] = \mu_n C_{ox} \frac{W_{n1}}{L_{n1}} \left[(V_{gsn1} - V_{tn})\, V_{dsn1} - \frac{(V_{dsn1})^2}{2} \right] \tag{3.4}$$

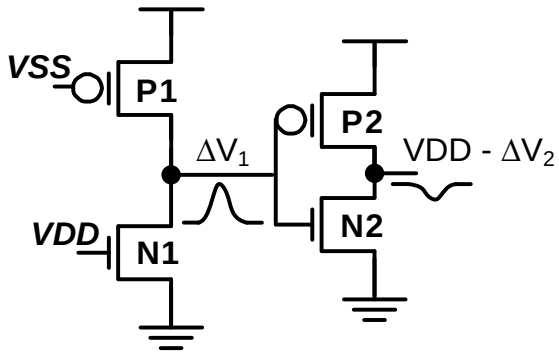


Figure 3.5: A simplified schematic of CD logic during contention mode.

where μ_p and μ_n are the hole and electron mobility of pMOS and nMOS transistors respectively, C_{ox} is the oxide capacitance, W and L are the transistor width and length respectively, V_{gs} and V_{ds} are the transistor gate-to-source and drain-to-source voltages respectively, and V_{dsatp} is the velocity saturation voltage of *P1*. Rearranging Eq. 3.4 and assuming $L_{n1} = L_{p1}$ gives

$$\Delta V_1 = V_{gsn1} - V_{tn} - \sqrt{(V_{gsn1} - V_{tn})^2 - \frac{2\left(\left(V_{DD} - V_{tp}\right)V_{dsatp} - V_{dsatp}^2/2\right)}{(\mu_n W_{n1})/(\mu_p W_{p1})}} \tag{3.5}$$

By Taylor expansion, the square root term can be approximated to first-order as:

$$\sqrt{N^2 + d} = N + \frac{d}{2N} - \frac{d^2}{8N^3} + \frac{d^3}{16N^5} \cdots \approx N + \frac{d}{2N} \tag{3.6}$$

Hence Equation 3.5 can be approximated as:

$$\Delta V_1 \approx \frac{\left(V_{DD} - V_{tp}\right)V_{dsatp} - V_{dsatp}^2/2}{\left(\mu_n W_{n1}\right)\left(V_{gsn1} - V_{tn}\right)/(\mu_p W_{p1})}. \tag{3.7}$$

ΔV_2 can also be found through a similar approach. Consider Fig. 3.5 again, transistor *N2* operates in the subthreshold region while *P2* is working in the linear mode. Equating the two current equations yields:

$$\frac{W_{n2}}{L_{n2}} I_t e^{\frac{V_{gsn2} - V_{tn}}{\eta V_T}} \left(1 - e^{\frac{-V_{dsn2}}{V_T}}\right) = \mu_p C_{ox} \frac{W_{p2}}{L_{p2}} \left[\left(V_{gsp2} - V_{tp}\right)V_{dsp2} - \frac{(V_{dsp2})^2}{2}\right] \tag{3.8}$$

where [35], [36]

$$I_t = \mu_o C_{ox} (V_T)^2 e^{1.8}, \quad \eta = 1 + \frac{3T_{ox}}{W_{dm}}, \quad V_T = \frac{KT}{q} \tag{3.9}$$

where μ_o is the zero bias mobility, η is the subthreshold swing coefficient, W_{dm} is the maximum depletion layer width, V_T is the thermal voltage, K is the Boltzman constant, T is the temperature in Kelvin, and q is the electron charge. In the case of nMOS transistors, μ_o is simply μ_n. Rearranging Equation 3.8 and assuming $L_{n2} = L_{p2}$ gives:

$$\frac{W_{n2} I_t e^{\frac{\Delta V_1 - V_{tn}}{\eta V_T}}}{\mu_p C_{ox} W_{p2}} = (V_{DD} - \Delta V_1 - V_{tp}) \Delta V_2 - \frac{(\Delta V_2)^2}{2} \tag{3.10}$$

where $e^{\frac{-V_{dsn2}}{V_T}} \approx 0$ since $(V_{DD} - \Delta V_2) >> V_T$. Solving ΔV_2 then yields:

$$\Delta V_2 = V_{DD} - \Delta V_1 - V_{tp} - \sqrt{(V_{DD} - \Delta V_1 - V_{tp})^2 - A e^{\frac{\Delta V_1 - V_{tn}}{\eta V_T}}} \tag{3.11}$$

$$A = \frac{2 W_{n2} I_t}{\mu_p C_{ox} W_{p2}} \tag{3.12}$$

Apply Taylor expansion,

$$\Delta V_2 \approx V_{DD} - \Delta V_1 - V_{tp} - \left((V_{DD} - \Delta V_1 - V_{tp}) - \frac{A e^{\frac{\Delta V_1 - V_{tn}}{\eta V_T}}}{2 (V_{DD} - \Delta V_1 - V_{tp})} \right) \tag{3.13}$$

Finally,

$$\Delta V_2 \approx \frac{A e^{\frac{\Delta V_1 - V_{tn}}{\eta V_T}}}{2 (V_{DD} - \Delta V_1 - V_{tp})} \tag{3.14}$$

Equation 3.7 and 3.14 provide several first-order design insights for CD logic. For a given ΔV_1, designers can quickly estimate the required W_{p1} to W_{n1} ratio. Moreover, ΔV_1 is linearly proportional to the shift of V_t and transistor width in the presence of process variations. When ΔV_1 is sufficiently small, ΔV_2 is approximately zero. As ΔV_1 increases, both numerator and denominator in Equation 3.14 contribute to ΔV_2's exponential increase. In a multi-stage CD logic circuitry, ΔV_1 of each stage will slowly increase, due

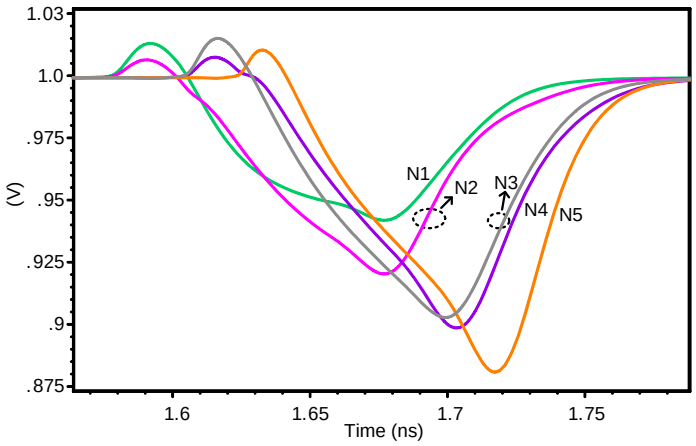


Figure 3.6: Simulated output glitch of a five stage two-input AND gate implemented with CD logic.

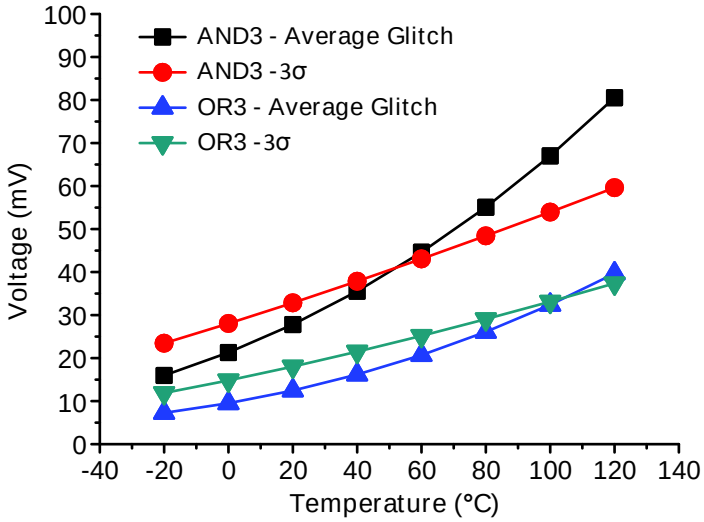


Figure 3.7: Temporary glitch mean and standard deviation at the output of 3-input CD AND and OR gate vs. temperature in a Monte-Carlo simulation with 7500 iterations.

to the reduced V_{gs} of *N1* as a result of ΔV_2 from the preceding stage. This phenomena is demonstrated in Fig. 3.6, where the glitch level aggravates as it traverses through a series of 2-input AND gate implemented with CD logic. Equation 3.14 also suggests that

ΔV_2 is a strong function of temperature, which is also demonstrated in Fig. 3.7. Fig. 3.7 illustrates the mean and three standard deviation (σ) of the temporary glitch at the output of a 3-input CD AND and OR gate vs. temperature. The mean and σ are calculated from a Monte-Carlo simulation with 7500 iterations. As the temperature increases from 20°C to 120°C, the 3σ glitch level of a 3-input AND gate raises from 55mV to approximately 140mV. Therefore, designers who wish to employ CD logic need to enforce more stringent design guidelines in order to sustain the system's reliability, i.e. simulate the circuits under extreme temperature and corner conditions.

3.3.3 Power Consumption

Data activity measures how frequently signals toggle and is defined as

$$data\ activity = \frac{\#\ of\ signal\ transitions}{\#\ of\ signals \times \#\ of\ clock\ cycles} \tag{3.15}$$

Static logic has an empirical α of $0.1 \sim 0.2$ [16] and dynamic domino logic has an activity factor of 0.5. While CD logic's α is also 0.5, it always consumes power when it enters the evaluation period. During the evaluation period, CD logic always dissipates power via either dynamic power dissipation (X goes to VDD and Out is discharged to GND) or direct path current (contention mode)). While CD logic consumes more power, CD logic may still be an attractive choice in a high-performance, full-custom design because 1) CD logic is only intended to replace the critical path and 2) power management techniques such as clock gating [37] [38], where the clock connection to idle module is turned off (gated), will significantly reduce CD logic's dynamic power consumption.

3.3.4 CD Logic Family

CD logic's LB (Fig. 3.3) can be modified such that the inverter is replaced by a static gate to achieve even higher performance, since the inverter delay is not wasted. Such a variation will be referred to as compound CD logic (CCD) in this thesis, analogous to the case of CDL of the dynamic domino logic. Another family of CD logic was proposed

in [39], where the output inverter is replaced by a dynamic domino logic. In other words, CD logic is domino compatible. The analysis in [39] shows that a 64-bit parallel-prefix adder employing this type of logic is superior than CDL-based counterpart; however, it requires additional design considerations due to the degraded noise margin.

3.4 CD Logic Characterization

Unless otherwise specified, all simulation runs in this section are done in schematic level (transistor netlists) with extracted parasitic capacitance in the Cadence design environment using a 65-nm CMOS technology. All CD logic gates are designed such that the worst case glitch level under 6σ deviation with nominal VDD (1 V) is less than 300 mV at 110°C. A 300 mV constraint is set based on the V_t (approximately 320 mV) of the chosen technology to ensure that no false logic evaluation will occur. The measured power consumption includes clock trees and data buffers, which are both sized to drive a Fanout-of-4 (FO4) load. The outputs of all the logic gates are driving an identical 20fF load. The window duration (width) is defined as the 50% point of the falling edge of ϕ_{CD} to the 50% point of the rising edge of node Y. The delay is measured at the 50% switching point of either ϕ_{CD} or data to the 50% switching point of the latest output.

All logic transistors have a 1μm effective nMOS width. For CD logic, the pMOS CLK transistors' width is 2.4μm. The transistor sizings are optimized primarily for delay, because the main objective of this section is to explore CD logic's performance advantage. The clock and data frequencies are set to 2 GHz.

3.4.1 Noise Margin vs. Window Width

Noise margin is defined as the dc-noise level at the input that generates a false logic evaluation at the output of the same gate in this thesis and can be computed based on the following formula:

$$Noise\ Margin = |V_{original} - V_{noise}| \tag{3.16}$$

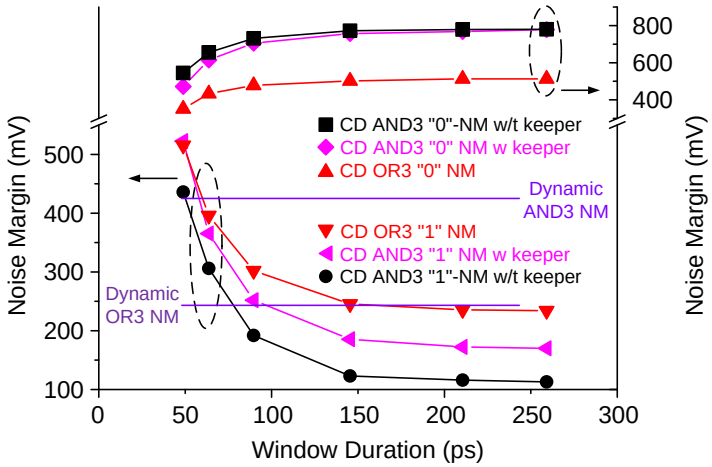


Figure 3.8: Simulated logic "1" and "0" noise margin vs. window duration for a 3-input AND gate and OR gate.

where $V_{original}$ is the expected voltage level without any input noise interference and V_{noise} is the input dc-noise voltage that causes the false logic evaluation. For CD logic, two types of noise margin are defined: Logic "1" and "0" noise margin. Logic "1" noise margin refers to the input dc-noise level that causes CD logic to fail to remain in the contention mode. If the voltage of *IN* (V_{in}, Fig. 3.3) which is supposed to be at full VDD, is now degraded due to noise, then the glitch level at X may be too high such that *Out* is falsely discharged. In this case, the noise margin can be calculated as $1V - V_{in}$. Similarly, logic "0" noise margin refers to the input dc-noise level that causes CD logic to fail evaluating. If an input, which is supposed to be at GND, is now much higher due to noise, the contention between pMOS transistors and nMOS PDN will cause X to settle at an intermediate voltage instead of VDD. When ϕ_{CD_d} rises to VDD (window closes), Y will also be charged up through M3 and M4, since M3 is on and M4 is partially on because X is not at VDD. If the voltage level at X is too low, then Y will be charged to VDD through positive feedback and X will be discharged to GND through M7 which is driven by the noise source.

Fig. 3.8 shows the simulated worst case logic "1" and "0" noise margin of a 3-input AND gate and OR gate implemented with CD and dynamic domino logic. CD logic's logic "0" noise margin is always much higher than the logic "1" noise margin, suggesting that

CD logic is more robust during the C-Q delay and the D-Q delay than the contention mode. Moreover, as window width becomes longer, logic "0" noise margin improves while logic "1" noise margin degrades for both logic types. Therefore, reducing the window duration not only minimizes the power consumption but also improves CD logic's overall robustness[2] . For noise-margin-sensitive applications, a minimum-size nMOS keeper (gate connected to *Out*, drain connected to X) can be added to improve its overall robustness. In this case, the minimum size keeper improves logic "1" noise margin by approximately 60mV with virtually no degradation in logic "0" noise margin.

3.4.2 CD Logic Performance

Fig. 3.9 and 3.10 illustrate the normalized delay and average power consumption of static, dynamic, and CD logic, respectively, in five logic expressions with various input data activities. The average power is calculated by summing up the power consumption of every possible input vector, then dividing by the number of input vector combinations.

CD logic demonstrates superior performance, especially for complicated logic expressions, such as $Y = AB+CD$ (AOI22), in D-Q mode due to the pre-evaluated characteristic. This is demonstrated in Fig. 3.9, where CD logic is approximately two times faster than dynamic domino logic. This is contributed by 1) the pre-evaluated characteristic and 2) the fewer number of transistors in the critical path (3N1P for dynamic, while only 2P1N for CD logic). On the other hand, CD logic's performance is only approximately the same or even worse than that of dynamic domino logic during C-Q mode. Therefore, it is advantageous to implement CD logic in a single cycle, multi-stage datapath since only then the pre-evaluated feature (D-Q delay) of CD logic can be fully utilized.

The power consumption of CD logic at 50% data activity is at least 3× and 5× higher than that of static logic in AOI22 and the rest of logic expressions, respectively. This suggests that CD logic should only be used to replace the critical path in any circuit block, since it is not energy-efficient to implement any system with CD logic only. Table 3.2 summarizes the total transistor width of static, dynamic, and CD logic. Despite CD

[2]As long as logic "0" noise margin is higher than logic "1" noise margin, improving logic "1" noise margin is equivalent to improving CD logic's overall robustness.

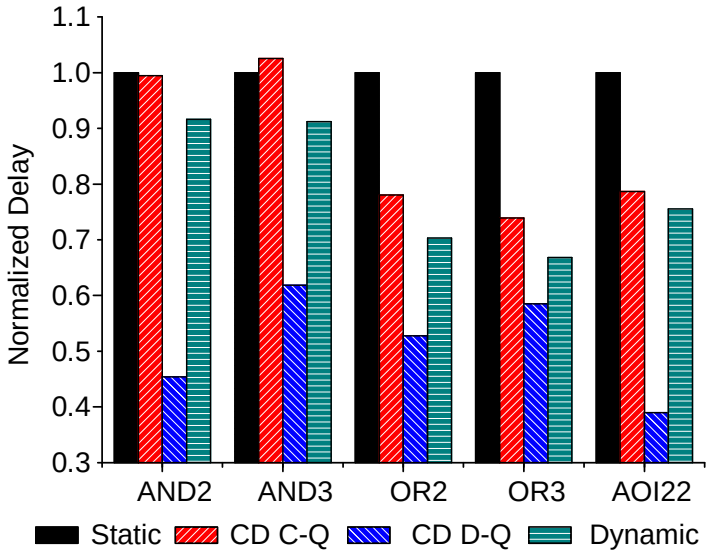


Figure 3.9: Normalized delay of five logic expressions implemented in static, dynamic, and CD logic.

logic's additional transistors overhead, the average area of CD logic is 13% smaller and 4.5% larger than that of static and dynamic domino logic, respectively.

3.5 Performance Analysis

3.5.1 8-bit Ripple Carry Adders

The simulation setup in this section is similar to that of Section 3.4. Three 8-bit ripple carry adders (RCAs) using static, dynamic, and CD logic style are simulated to compare their performances. RCA with FTL on the critical path is also implemented; however, our analysis indicates that FTL-based RCA generates false outputs at the later bits because of the false-evaluation phenomena described earlier. NP-FTL (equivalent to NP-domino, where nMOS-FTL and pMOS-FTL alternate) is also difficult to realize because the output glitch is significant and easily exceeds 500mV under process variations.

The basic static full adder (FA) is implemented with 28 transistors with sizing strongly

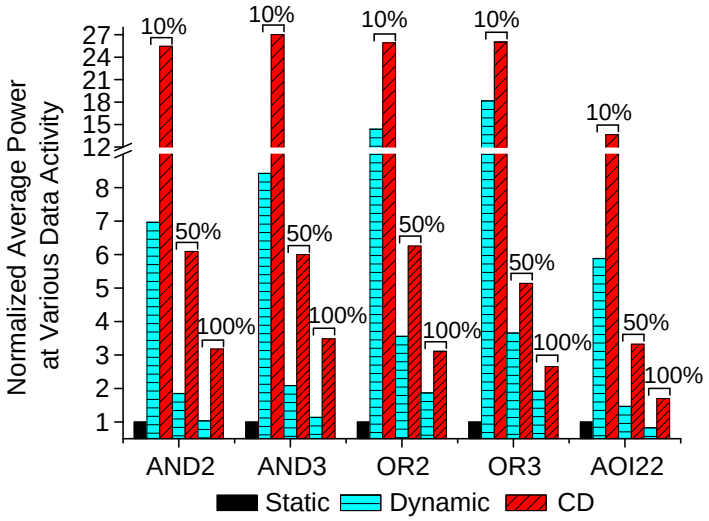


Figure 3.10: Normalized average power of five logic expressions at various data activities implemented in static, dynamic, and CD logic.

in favor of C_{out} computation [16]. The main purpose of this 8-bit RCA is to demonstrate CD logic's performance advantage and to discuss the design considerations that should be taken into account when using CD logic. A more energy-efficient pass-transistor FA design [40] will be implemented in the subsequent analysis to provide a more realistic comparison.

Only the timing-critical carry generation is replaced with dynamic and CD logic while non-critical sum computation remains static in all three RCAs. 10000 random input vectors are applied to RCAs to compute the average power consumption. The clock timing is designed in such a way that all the CD logic gates except the first stage operate in the D-Q mode with a window duration of approximately 115ps[3]. Fig. 3.11(a) depicts the RCA block diagram and FA schematic. Fig. 3.11(b) shows the corresponding worst-case timing

[3]The window duration is a function of logic expression, number of preceding stages that are driving by the same phase clock signal, maximum glitch level constraint, and the robustness of the overall system. Extensive simulation results have indicated that a window duration of 115ps provides excellent delay and power performance while maintaining a sufficient timing margin against PVT and transistors mismatch variations (Table 3.6).

Table 3.2: Number of transistors and area comparison for Static, dynamic, and CD logic.

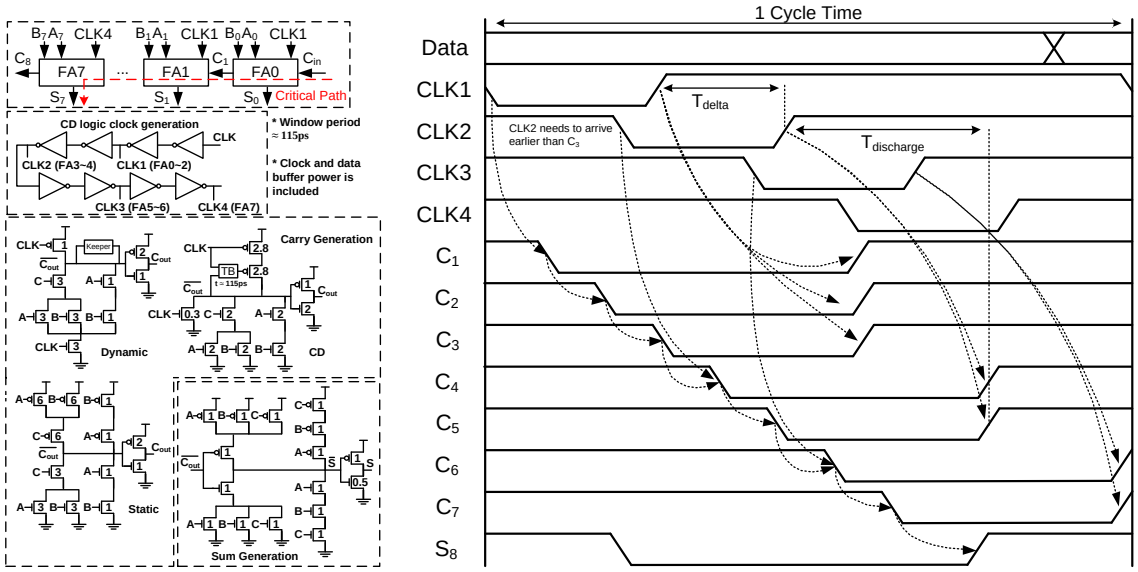
	Total Transistor Width (μm)			Number of Transistors		
	Static	Dynamic	CD	Static	Dynamic	CD
AND2	11	13.6	14.96	6	7	17
AND3	18	20.9	19.96	8	8	18
OR2	13	10.6	12.96	6	7	17
OR3	24	12.6	13.96	6	7	18
AOI22	27	19.6	18.96	10	9	19
Average	18.6	15.46	16.16	7.2	7.6	17.8

diagram for CD logic, which occurs when $\{C_{in}, A_0 \cdots A_7, B_0 \cdots B_7\} = \{0, 0 \cdots 0, 1 \cdots 1\}$.

Design Considerations in a Multi-Stage System

Fig. 3.11(b) provides several insights in designing single cycle, multi-stage CD circuitries. When CD logic is to be used with other logic styles and when the gate preceding the CD logic is not a pre-charge type logic (i.e., dynamic domino logic), then the inputs (*Data*) can only make transitions when all CD logic gates are in the pre-discharge mode (i.e., CLK1-4 are high). CD logic may suffer from additional power consumption due to the possible direct path current during the predischarge mode. Consider a CD-based RCA with the worst case input vector, FA0-2's CD logic carry circuitries enter predischarge mode when *CLK1* goes to high. The internal nodes before the output inverter of all CD logic gates are all discharged to logic "0" by nMOS clock transistors, and the outputs (C_1 to C_3) are charged to logic "1". If C_3 becomes logic "1" before FA3 enters predischarge mode (i.e., CLK2 is still high), then a direct path takes place[4]. To avoid this condition, it is necessary for $T_{discharge} \geq T_{delta}$, where $T_{discharge}$ is the time that CD logic takes to charge its output and T_{delta} is the delay between two adjacent clock signals (CLK2 to CLK1, or CLK3 to CLK2, etc.).

[4]Consider the CD carry generation circuitry shown in Fig. 3.11(a), under worst case vector condition B = 1 and A = 0. If input C (C_{in} from the preceding stage) rises to logic "1" (originally at logic "0") before CLK is high, then both pMOS and nMOS transistors are on.



(a) (b)

Figure 3.11: Ripply-carry adder: (a) block diagram and (b) timing diagram for CD logic.

CD logic sizing strategy

To guarantee a 6-sigma glitch level of 300mV at 110°C, the following sizing strategy is employed:

1. An equally weighted variable is assigned to the width of CD logic's pMOS pull-up transistors.
2. The entire circuitry (e.g., 8-bit RCA) is then simulated under typical corner at 110°C to determine the glitch level. Extensive simulation results reveal that if this glitch level is approximately 65mV, then the 6 sigma glitch level will be less than 300mV.
3. Iterative simulations are performed by sweeping this variable until the glitch level is around 65mV.

The equally weighted scheme may not be the optimal solution. However, different sizing schemes have been explored and simulation results indicate that no apparent performance

Table 3.3: Performance comparison of RCAs implemented with various logic styles.

	Static			Dynamic			CD		
Data Activity	10%	50%	100%	10%	50%	100%	10%	50%	100%
Delay (ps)	369.4 (1.00)			292.6 (0.79)			224.4 (0.61)		
Power (μW)	50.95	254.77	509.54	142.70	336.05	401	231.85	533.96	602.82
Power-Delay Product (fJ)	18.8	94.1	188.2	41.8	98.3	117.3	52	119.8	135
Energy-Delay Product (fJ·ps)	6944	34761	69521	12231	28763	34322	11669	26883	30294

improvement is achieved compared to the sizing strategy described above.

RCA Performance

Table 3.3 compares static, dynamic, and CD logic RCA with various FOMs at different data activity factor. CD-based RCA is approximately 39% and 23% faster than the static and dynamic counterparts, respectively. On the other hand, the power consumption of CD logic ranges from 4.55× to 1.18× higher than that of static logic. In terms of power-delay product (PDP), CD logic is 2.78× more and 0.72× less than static logic at 10% and 100% data activity, respectively. CD logic provides a speed advantage that logic styles such as static and dynamic have difficulty achieving. Therefore, CD logic is suitable in a system where performance is the most critical factor.

3.5.2 32-bit Carry Lookahead Adder

32-bit carry lookahead adders (CLA) are implemented to further analyze CD logic's performance. The detailed operations of CLA are described in [16] and the schematic is displayed in Fig. 3.12. The 32-bit CLA uses eight 4-bit FAs with dedicated circuitry to facilitate carry generation. The energy-efficient FA used in this analysis utilizes pass transistor logic styles with only 24 transistors for sum generation [40]. For the carry generation, only the

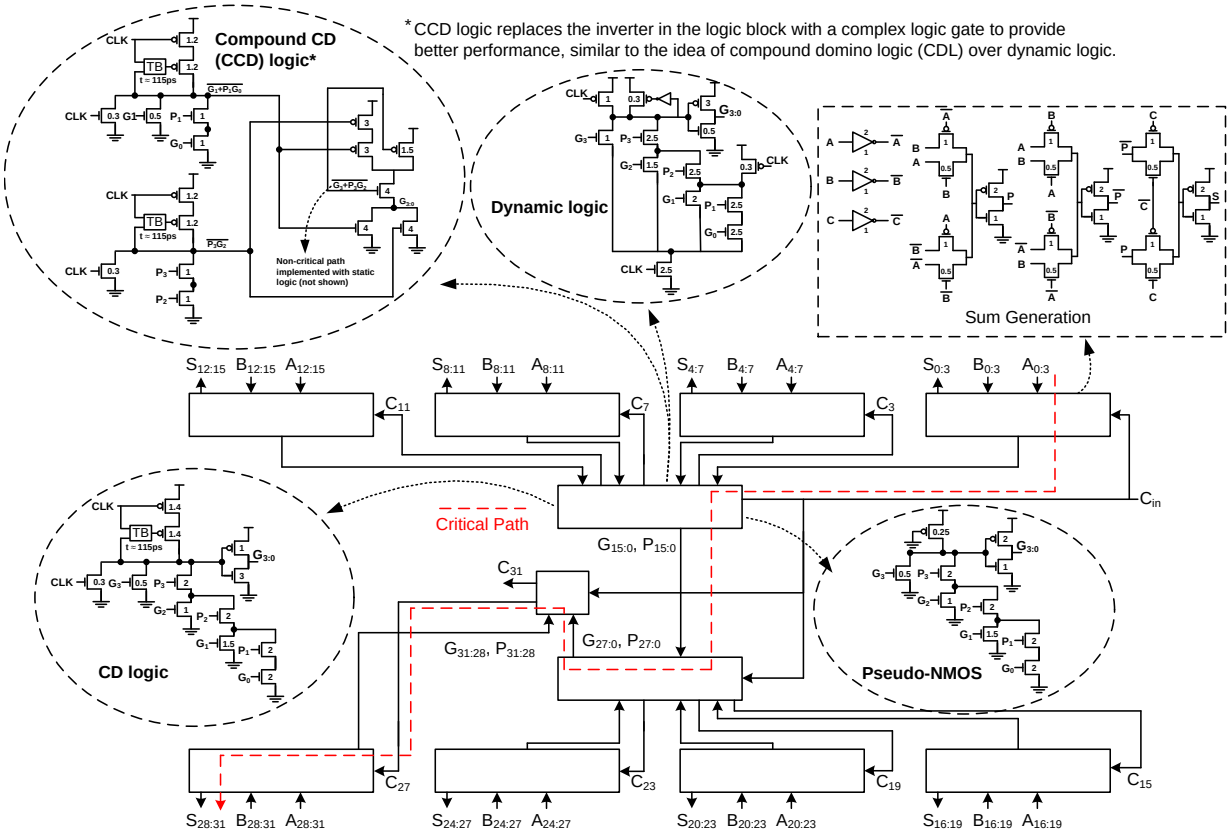


Figure 3.12: 32-bit carry lookahead adder with critical path implemented using various logic styles.

critical path is replaced with different logic style. The maximum fan-in is limited to four except the case of dynamic domino logic due to the footer transistor. In this case, the 4-bit critical carry generation path of CLA is:

$$G_{3:0} = G_3 + P_3(G_2 + P_2(G_1 + P_1(G_0))) \tag{3.17}$$

where G and P are the generate $(A \cdot B)$ and propagate $(A \oplus B)$ signals, respectively. CDL and CCD logic are implemented to reduce the fan-in. One can utilize the inversion property

Table 3.4: 32-bit CLAs performance comparison.

		Static	Dynamic	CDL	pseudo-nMOS	CD logic	CCD logic
Delay (ps)	Data Activity	448 (1.00)	316 (0.71)	287 (0.64)	313 (0.70)	**272 (0.61)**	**239 (0.53)**
Power (mW)	100%	5.13	5.27	5.29	5.34	5.02	5.09
	50%	1.94	2.22	2.21	2.21	2.31	2.34
	25%	0.99	1.32	1.33	1.25	1.43	1.46
	10%	0.42	0.8	0.81	0.68	0.9	0.94
PDP (pJ)	100%	2.3	1.67	1.52	1.67	1.37	1.22
	50%	0.87	0.7	0.63	0.69	0.63	0.56
	25%	0.45	0.42	0.38	0.39	0.39	0.35
	10%	0.19	0.25	0.23	0.21	0.25	0.22
EDP (pJ·ps)	100%	1030	526	437	523	371	291
	50%	390	222	182	216	171	134
	25%	199	132	110	122	106	84
	10%	85.1	79.8	66.7	66.3	66.8	53.5
Worst Leakage (μA)		32.9	32.1	32.7	551.2	33.5	33.4

and rearrange Equation 3.17 to

$$\overline{G_{1:0}} = \overline{G_1 + P_1 G_0}, \quad \overline{P_{3:2}} = \overline{P_3 P_2}, \quad \overline{G_{3:2}} = \overline{G_3 + P_3 G_2}$$
$$G_{3:0} = \overline{\overline{G_{3:2}}(\overline{P_{3:2}} + \overline{G_{1:0}})} \tag{3.18}$$

Therefore, a maximum fan-in of two and three can be achieved with CCD logic (Chapter 3.3.4) and CDL, respectively. The critical pMOS transistors' width of CCD logic is slightly smaller than that of CD logic to satisfy the 300mV glitch constraint because the pull-up path in the *Logic Block* now consists of two pMOS transistors. Footless dynamic domino logic cannot be applied in this analysis because not all the circuits are implemented with dynamic domino logic. Therefore, some of the inputs to the dynamic domino logic in the critical path can come from non-critical static gates. In order to satisfy the monotonicity requirement and to avoid the possible direct path current, a footer transistor is required for all dynamic domino logics. On the other hand, if the entire 32-bit CLA is implemented with dynamic domino logic, then the footless scheme can be applied. However, simulation results indicate that the power consumption in this case is much higher than that of the current setup, thus making it a less attractive design.

Table 3.4 summarizes the simulation results of the 32-bit CLAs. Power consumption is

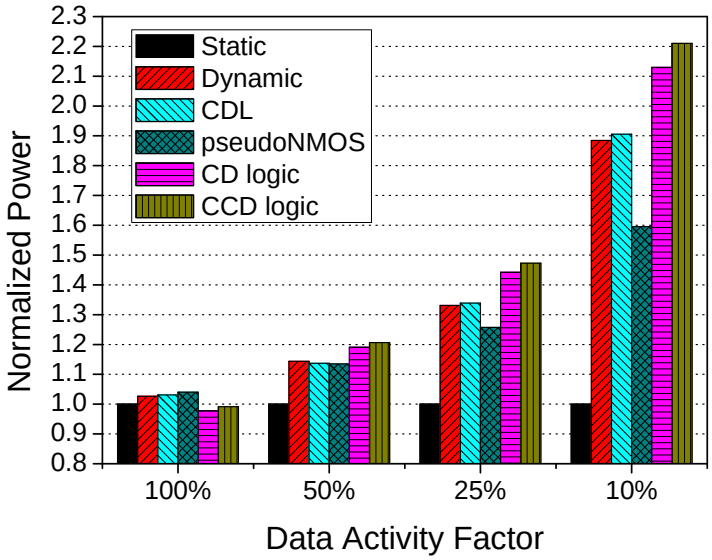


Figure 3.13: Normalized power of 32-bit CLAs.

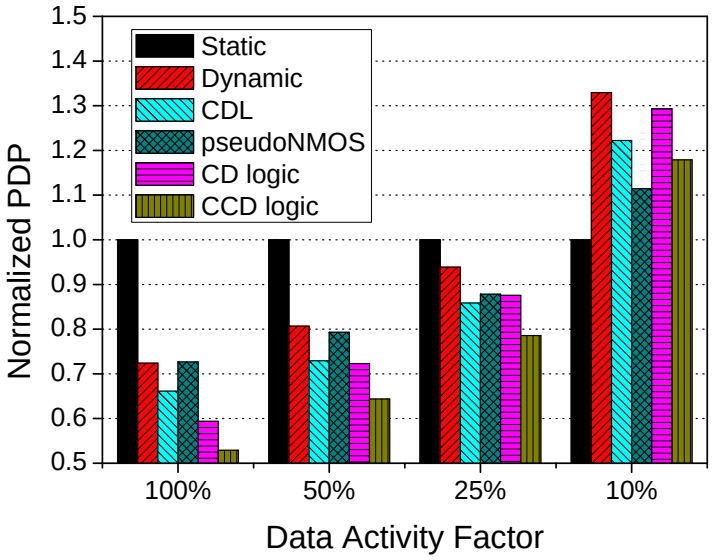


Figure 3.14: Normalized power-delay-product of 32-bit CLAs.

calculated with 5000 random input vectors. The performance enhancement of CD and CCD logic are evident in this case with 39% and 47% speedup over static design, respectively.

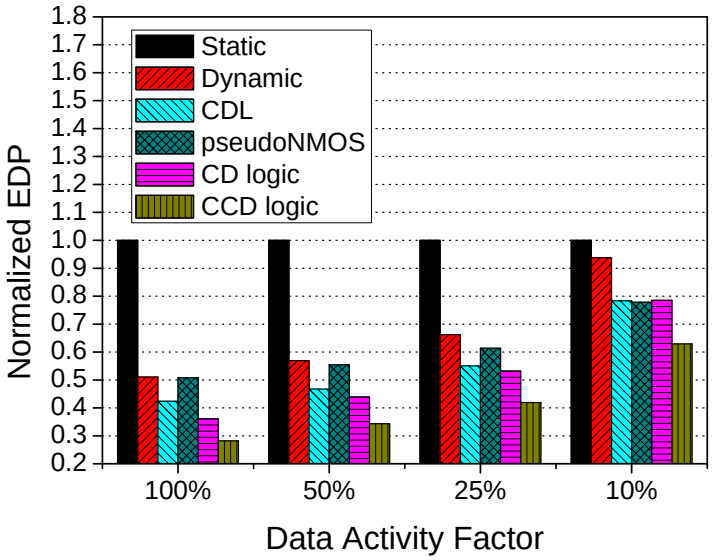


Figure 3.15: Normalized energy-delay-product of 32-bit CLAs.

Table 3.5: CD and CCD logic's glitches in 32-bit CLAs at different temperature settings.

	CD logic			CCD logic		
Temp. (°C)	Mean (mV)	σ (mV)	Mean + 6σ (mV)	Mean (mV)	σ (mV)	Mean + 6σ (mV)
85	65.3	23.2	204.5	63.6	26.2	220.8
110	88.5	29.8	267.3	86.8	36.1	303.4

Both CD and CCD logic are also faster than pseudo-nMOS, primarily because of the larger effective pMOS width. The worst-case leakage of all the designs are comparable except for pseudo-nMOS, which is caused by the contention between nMOS PDN and the weak pMOS pull-up transistor. In this case, pseudo-nMOS's leakage (static power dissipation) is at least 15× higher than the rest of the designs.

Figs. 3.13-3.15 show the normalized power, PDP, and EDP of all the CLAs analyzed in this work, respectively. At 10% (100%) α, the power consumption of CD logic is 2.1× (1.05×) higher than that of the static logic. Compared to the case of 8-bit RCA, the power consumption discrepancy between static and CD logic at low data activity is less obvious,

Table 3.6: PVT and Monte-Carlo performance analysis of the CD and CCD logic based designs.

Corner	Temperature (°C)	VDD (V)	8-bit RCA (CD)	32-bit CLA (CD) (ps)	32-bit CLA (CCD) (ps)
FF	110	0.9	199	238	200
		1.1	152	189	157
	-30	0.9	244	288	250
		1.1	182	222	190
FS	110	0.9	244	288	250
		1.1	182	222	190
	-30	0.9	265	289	263
		1.1	178	206	185
SF	110	0.9	220	259	218
		1.1	164	205	169
	-30	0.9	217	248	214
		1.1	152	186	156
SS	110	0.9	369	439	388
		1.1	262	321	280
	-30	0.9	392	437	405
		1.1	249	294	266
Monte Carlo	mean	1	226	274	240
	σ	1	16.2	17.8	18.2

because CD logic only accounts for approximately 10% of the entire circuitry. CCD logic achieves lowest PDP at all α except at 10%. While higher than CCD logic, PDP of CD logic is comparable to CDL and pseudo-nMOS and is lower than the rest of the designs. At 25% α, EDP reduction of CD and CCD logic from other designs is at least 4% and 24%, respectively. At 10% data activity, CCD logic achieves the lowest EDP with at least 19% improvement. Notice that the power consumption of a CLA implemented with CD logic is lower than that of a CLA with dynamic domino logic at 100% data activity. This is because the width (1μm) of CD logic's pull-up pMOS transistors in this CLA is much smaller than that of CD logic in a single stage logic gate. Hence the power consumption of dynamic domino logic is higher than that of CD logic due to higher internal capacitance at 100% data activity. Similar reasonings can easily be applied to a CLA with CCD logic.

Table 3.5 summarizes the mean and σ of CD and CCD logic's worst glitch level in a Monte-Carlo simulation with 2000 samples. The worst case glitch is calculated by summing

up the mean and the extrapolated six sigma deviation[5]. CCD logic exhibits higher worst case glitch level than CD logic at both 85°C and 110°C, despite its already lower critical pMOS effective width. Both designs are approximately within the glitch constraint set earlier, and demonstrate that they are still able to function properly under extreme process and temperature conditions.

Table 3.6 shows the process-voltage-temperature (PVT) analysis and Monte-Carlo simulation results (2000 iterations, 27°C) of the proposed CD and CCD-logic-based designs. All the designs are functional under extreme PVT variations, hence demonstrating the proposed CD logic's robustness.

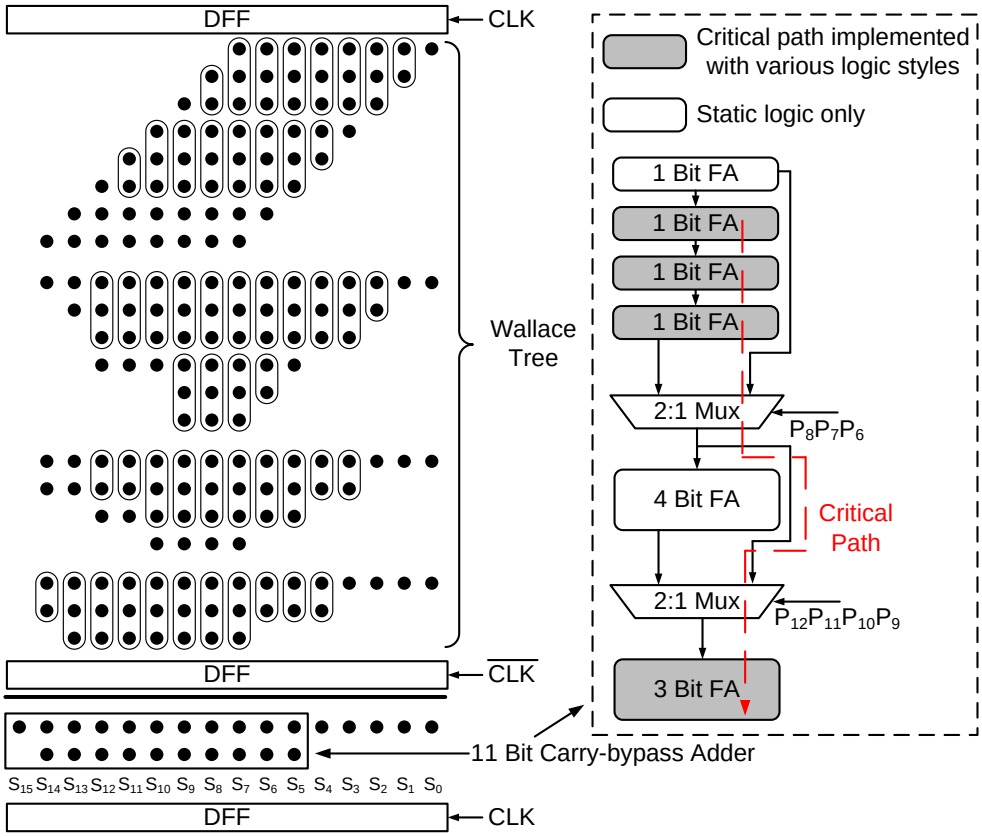


Figure 3.16: Schematic of an 8-bit Wallace tree multiplier.

[5]It is assumed that the glitch variation as a result of process variations follows a Gaussian distribution.

3.5.3 8-bit Wallace Tree Multiplier

Single-cycle, two-phase, 8-bit Wallace tree multipliers are implemented and analyzed. The first phase (CLK is high) is a Wallace tree utilizing 3:2 compressors to reduce the number of partial products and during the second phase, final addition is carried out by a 11-bit carry bypass adder, as shown in Fig. 3.16. Only the critical path of the final adder is implemented with various logic styles with the exception of multiplexors, while the rest of the circuits remain static. Simulation setups are similar to that of 32-bit CLAs.

Table 3.7 summarizes the normalized power, PDP, and EDP performance results of the various 8-Bit multipliers under different data activities. CD logic achieves a similar speedup (40%) over static logic compared to 32-Bit adders. At 25% α, CD logic consumes 16% more power, but is 30% and 58% more PDP and EDP efficient than static logic, respectively. Compared to dynamic domino logic, CD logic is approximately 15% faster, and achieves a minimum 25% EDP reductions. In 8-Bit multipliers, because critical path is only a small portion of the entire circuitry, CD logic has the lowest PDP and EDP values among all the logic styles across all data activities. CCD and CDL logic are not included in this analysis because they are not particularly suitable for this setup. This is because G and P signals are not generated in this case; instead, direct inputs (A,B) similar to 8-Bit RCA are used to compute the carry.

3.6 Conclusion

A new high-performance logic style with constant-delay characteristic and self-reset circuitry is proposed. The pre-evaluated feature of CD logic makes it particularly suitable in a circuit block where a unique critical path exists and performance is the primary concern. Several advantages of CD logic over previously proposed feedthrough type logic styles have been explored, including i) better noise margin, ii) ability to cascade multiple stages to perform complicated logic evaluation, and iii) reduced power consumption via a local timing window technique. In addition, CD logic does not require complementary input signals and the clock timing requirement is comparable to that of dynamic domino logic. The performance advantage of CD logic has been demonstrated in five logic types. In D-Q mode,

Table 3.7: Performance comparison of 8-bit Wallace tree multipliers implemented with various logic styles.

Data Activity		100%			50%			25%			10%		
	Delay (ps)	Power (mW)	PDP (pJ)	EDP (pJ·ps)	Power (mW)	PDP (pJ)	EDP (pJ·ps)	Power (mW)	PDP (pJ)	EDP (pJ·ps)	Power (mW)	PDP (pJ)	EDP (pJ·ps)
Static	404	3.52	1.42	575	2.29	0.93	375	1.29	0.52	211	0.67	0.27	109
Dynamic	294	3.57	1.05	309	2.36	0.7	205	1.39	0.41	121	0.80	0.23	69
pseudo-nMOS	292	3.82	1.11	325	2.56	0.75	218	1.57	0.46	134	0.96	0.28	82
CD logic	243	3.59	0.87	212	2.46	0.60	145	1.49	0.36	88	0.88	0.22	52

CD logic achieves an average speed up of 94% and 58% with an average area overhead of -13% and 4.6%, respectively, compared to static and dynamic domino logic. Performance analysis of 8-bit ripple carry adders reveals that CD logic is 39% and 23% faster than static and dynamic domino logic, respectively. CD logic achieves PDP and EDP reduction of 28% and 56%, respectively, compared to static logic at 100% data activity. At low data activity however, CD logic is no longer energy-efficient. The excessive power consumption of CD logic as a result of inevitable direct path current during the contention mode indicates that CD logic should only be used to implement the critical path. Simulation results of 32-bit carry lookahead adders show similar speed advantage of CD logic compared to static logic. In this setup, CCD logic achieves the lowest PDP at all data activities except 10%. Also, CCD logic achieves the best EDP results, with 66%(37%) reduction compared to static logic at 50%(10%) data activity. CD logic has also demonstrated its robustness under extreme process, voltage, and temperature variations. The proposed CD and CCD logic-based 32-bit CLAs are functional under all PVT variations with 6σ worst-case glitches of 220.8 mV and 303.4 mV at 110°C, respectively.

CD logic's advantages in terms of delay and EDP are also demonstrated in 8-bit Wallace tree multipliers. Compared to 32-bit adders, CD logic achieves a similar delay improvement, but has an even better EDP reduction, primarily because the final adder which makes up the critical path of the multiplier is a relative small circuit block of the overall circuitry. At 25% α, CD logic is 52%, 25%, and 37% more EDP efficient than static, dynamic, and pseudo-nMOS logic, respectively.

Chapter 4

64-Bit High-Performance Adder with Constant-Delay Logic

Addition has always been one of the most commonly used arithmetic operations. Consequently, high-performance, energy-efficient addition, the core of the Arithmetic Logic Unit (ALU) in every microprocessor, has been one of the research focuses of digital circuits. Besides the manufacturing CMOS process, the two most critical design considerations of adders are the logic style and the carry-merge tree architecture [41]. For non-timing-critical applications, the static/pass-transistor logic style is often the preferred choice due to its reasonable performance, low power consumption especially at low data activity environment, and robustness. For timing-sensitive adders, however, designers may prefer the dynamic domino or CDL, owing to its superior switching speed compared to static/pass-transistor logic [20] [21] [22] [23] [42]. Nevertheless, the performance enhancement comes with several costs, including reduced noise margin, charge-sharing, and higher power dissipation. Therefore, CD logic has been proposed as a method to provide circuit designers an alternative in designing full-custom, high-speed digital circuits. CD logic is exclusively employed in an adder's critical path to confirm its performance potential.

Today's high-performance VLSI adders have adopted the carry-merge architectures based on the CLA method, developed by Weinberger and Smith [43–48] in generating the carry bits. The CLA structure is theoretically one of the fastest schemes since the delay

to add two N-bit numbers only depends on the logarithm of N, the number of bits in the adder [49–53]. Recently, Ling's algorithm has been considered to be a more energy-efficient adder architecture than Weinberger's [41] [19]. Ling's pseudo-carry equation reduces the number of critical transistors in the carry tree at the expense of an increased complexity in the sum pre-computation. Sparse trees, which compute only every second (S-2) or fourth (S-4) carry signals, have been extensively used due to their energy savings and performance enhancements [41] [19] [22]. A higher degree of sparseness implies a faster carry tree, since the complexity is effectively shifted from the carry tree to the sum pre-computation block [54–56]. While S-4 trees have been reported in Weinberger adders [22], only an S-2 tree has been realized for the Ling adders [19], as the combination of a Ling adder and an S-4 tree creates a new critical path in the sum block. In this work, an S-4 tree with Ling's algorithm implemented with CD logic, and redesign of the sum pre-computation block to avoid the new critical path, are described. In summary, the contributions of this chapter are:

- For the first time, CD logic is implemented and verified in silicon.
- For the first time, silicon results of an S-4 Ling's adder are reported in a deep sub-micron (65-nm) CMOS technology.

4.1 Adder Architecture

4.1.1 Carry-Merge Tree with Ling's Recurrence Algorithms

Assuming a_i and b_i are the input operands to the adder, the intermediate signals *generate* (g) and *propagate* (p) and the sum signal S can be obtained at every bit as:

$$g_i = A_i B_i \tag{4.1}$$

$$p_i = A_i + B_i \tag{4.2}$$

$$S_i = A_i \oplus B_i \oplus C_{in} \tag{4.3}$$

where $i = 0...63$ for a 64-bit adder, and C_{in} is the input carry to bit i. The g and p signals can be further combined to form *group* G and P signals:

$$G_{i:j} = C_{i:j} = G_{i:k} + P_{i:k} \cdot G_{k-1:j} \tag{4.4}$$

$$P_{i:j} = P_{i:k} \cdot P_{k-1:j} \tag{4.5}$$

where $C_{i:j}$ is the carry out from j_{th} to i_{th} stage. In Ling's transformation [57], the complexity of the above (Weinberger's) recurrence (hence the complexity of the carry-merge tree) is reduced by only generating the group pseudo-carry signal, $H_{i:j}$, instead of $C_{i:j}$. By identifying that $g_i p_i = g_i$, the generate term $G_{i:j}$ can be expressed as:

$$G_{i:j} = g_i + p_i \cdot G_{i-1:j} = p_i \left(g_i + G_{i-1:j}\right) = p_i \cdot H_{i-1:j} \tag{4.6}$$

In Ling's adder [58], only $H_{i-1:j}$ is computed and propagated through the carry-merge tree [50] [19] [41]. Hence,

$$\begin{aligned} H_{i:j} &= H_{i:k} + T_{i:k} \cdot H_{k-1:j} \\ &= g_i + gi - 1 + t_{i-1} \cdot g_{i-2} + t_{i-1} \cdot t_{i-2} \cdot g_{i-3} + \ldots + t_{i-1} \cdot t_{i-2} \ldots t_{j+1} \cdot g_i \end{aligned} \tag{4.7}$$

$$T_{i:j} = t_i \cdot t_{i-1} \ldots t_j \tag{4.8}$$

where t_i and $T_{i:j}$ are analogues to p_i and $P_{i:j}$ in Weinberger's recursions. It can be observed that one less term is required in generating $H_{i:j}$ than that of $G_{i:j}$, which implies one less transistor in the critical path. Consider $C_{3:0}$, which can be represented as:

$$C_{3:0} = g_3 + p_3 \left(g_2 + p_2 \left(g_1 + p_1 \left(g_0 + p_0 C_{in}\right)\right)\right) \tag{4.9}$$

In the worst case, the generation of the signal $C_{3:0}$ requires C_{in} to propagate through p_0 to p_3 and results in a critical path of five transistors. In Ling's equations however, the pseudo-carry $H_{3:0}$ is defined as:

$$H_{3:0} = g_3 + g_2 + t_2 \left(g_1 + t_1 \left(g_0 + t_0 C_{in}\right)\right) \tag{4.10}$$

which only has a critical path of four transistors. It is important to realize that Ling's equations only reduces the number of transistors in the entire carry-merge tree's critical path by one. This is because in the subsequent merging stages, such as:

$$H_{6:C_{in}} = H_{6:5} + T_{6:5} \left(H_{4:3} + T_{4:3} \left(H_{2:1} + T_{2:1} H_{0:c_{in}}\right)\right) \tag{4.11}$$

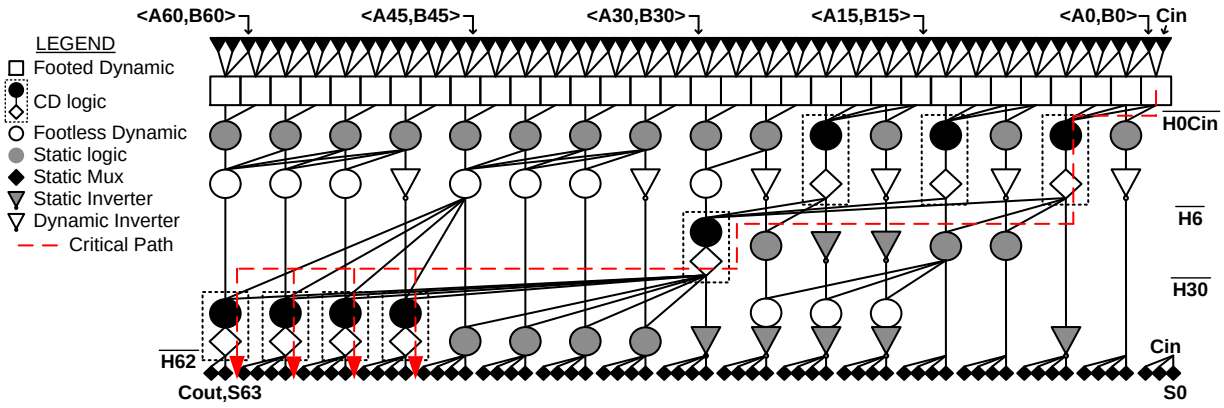


Figure 4.1: Proposed 64-bit adder carry-merge tree.

Ling's adder still results in a critical path of four transistors per stage, similar to that of the Weinberger's configuration.

The proposed hybrid S-4 carry-merge tree is shown in Fig. 4.1 and the corresponding schematics of CD and dynamic logic are depicted in Fig. 4.2. The carry-merge tree is carefully redesigned such that the number of stages is minimized by inversion property for energy-efficiency [41] and the number of CD logic blocks required for implementing the critical path is only eight. The first stage implements radix-2[1] merging with a footed dynamic logic and results in a maximum stack height of three, as shown in Fig. 4.2(b). A footer transistor is only required in the first stage, since the monotonicity of the global inputs A[0:63], B[0:63], and C_{in} cannot be guaranteed. A small pMOS CLK transistor precharges the internal node to minimize the charge-sharing problem. Notice that this first-stage configuration is only possible with Ling's configuration because the first recursion stage is simplified to

$$H_{i:i-1} = A_i B_i + A_{i-1} B_{i-1} \tag{4.12}$$

with three transistors in the critical path (including a footer transistor) from Weinberger's adder

$$G_{i:i-1} = A_i B_i + (A_i + B_i) \cdot (A_{i-1} B_{i-1}) \tag{4.13}$$

[1]Radixness implies the number of bits that are merged at every stage.

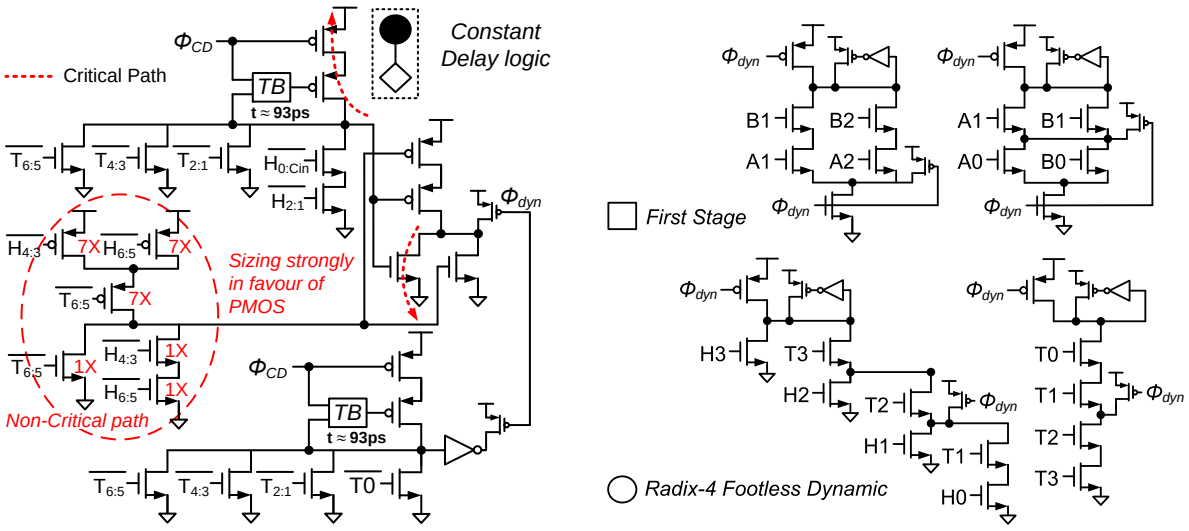


Figure 4.2: Carry-merge tree circuit schematic of (a) critical 4-bit merging with CD logic (b) first stage footed dynamic logic and non-critical radix-4 footless dynamic logic.

which results in a stack height of four. In the subsequent stages, non-critical paths are implemented with delayed-precharge compound domino logic with static transistor sizing strongly in favor of pMOS. Critical paths are implemented with CD logic along with absorption property [59]. The inversion and absorption property transform Equation 4.11 to

$$H'_{6:C_{in}} = \left\{ \left(T'_{6:5} + T'_{4:3} + T'_{2:1} + H'_{2:1}H'_{0:C_{in}}\right)' \cdot \left(T'_{6:5} + H'_{6:5}H'_{4:3}\right)' \right\}' \tag{4.14}$$

In this case, the number of critical transistors is reduced from 4 nMOS and 1pMOS (footless dynamic logic with an output inverter) to just 2 pMOS and 1 nMOS for a 4-bit merging operation. This is only possible with CD logic, since the critical path does not depend on the logic transistors, as highlighted in Fig. 4.2. The non-critical path of the 4-bit merging operation is also implemented with skewed static logic style, with pMOS transistors' width at least 7× larger than that of nMOS transistors. Fig. 4.3 displays the critical path of the proposed carry-merge tree, which consists of footed dynamic logic and three stages of CD logic. The *LB* block of CD logic employs pseudo-static logic to

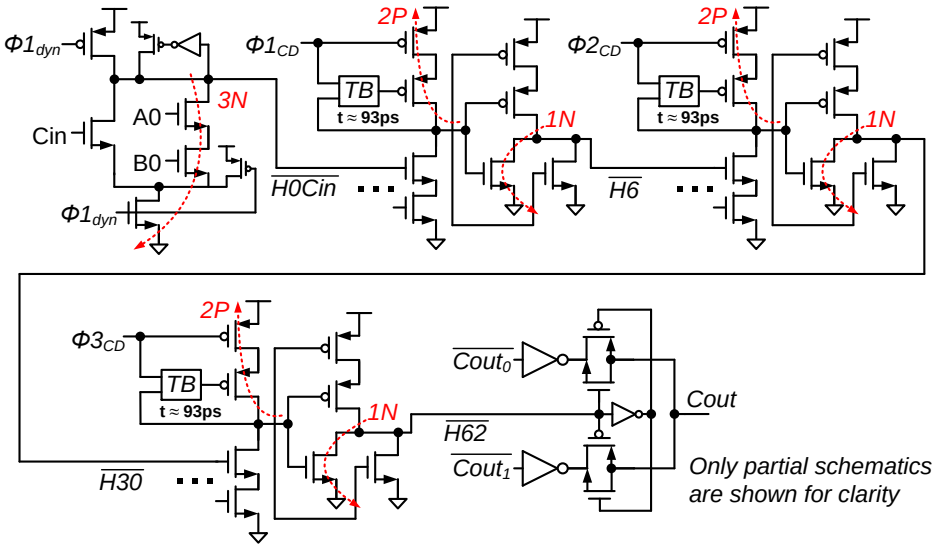


Figure 4.3: 64-bit adder critical path highlighted. The critical path consists of a footed dynamic logic and three stages of CD logic with a total of 12 transistors in the critical path.

maintain sufficient noise margin and achieve short precharge time. The proposed tree along with CD logic with both absorption and inversion properties have resulted in only twelve transistors in the carry-merge tree's critical path. Compared to the designs in [19] and [22], the number of transistors is reduced by 50% and 58%, respectively. The output of the last stage CD logic drives transmission gate multiplexers to derive the final outputs.

4.1.2 Sum Computation

The reduced complexity of Ling's carry-merge tree comes at the cost of a more complex sum pre-computation block. In this case, the sum generation in Ling's adder becomes:

$$S_i = A_i \oplus B_i \oplus (H_{i-1} \cdot p_i) \tag{4.15}$$

where H_{i-1} comes from the carry-merge tree. Because of the sparse-4 configuration, intermediate carry and conditional sum signals assuming input carries of 0 and 1 [22] both have

to be generated within the sum computation block, thus significantly increases its complexity. Consider a 4-bit ($S_{3:6}$) static conditional sum generator with input pseudo-carry $H_{2:0}$; the lower 2-bits (${S_3}^0, {S_3}^1, {S_4}^0, {S_4}^1$) can be generated as:

$$S_3 = A_3 \oplus B_3 \oplus (H_{2:0} \cdot p_2) \tag{4.16}$$

$${S_3}^0 = A_3 \oplus B_3 \tag{4.17}$$

$${S_3}^1 = A_3 \oplus B_3 \oplus p_2 \tag{4.18}$$

$$S_4 = A_4 \oplus B_4 \oplus (g_3 + p_3 (H_{2:0} \cdot p_2)) \tag{4.19}$$

$${S_4}^0 = A_4 \oplus B_4 \oplus g_3 \tag{4.20}$$

$${S_4}^1 = A_4 \oplus B_4 \oplus (g_3 + p_3 \cdot p_2) \tag{4.21}$$

which can be implemented with static logic only without creating a new critical path; however, for the upper 2-bits, a purely static design will violate the timing requirement. Hence, dynamic signals coming from the carry-merge tree are utilized to facilitate the sum generation, at the expense of higher power consumption. S_5 is then computed as:

$$\begin{aligned} {S_5}^0 &= A_5 \oplus B_5 \oplus (g_4 + p_4 \cdot g_3) \\ {S_5}^1 &= A_5 \oplus B_5 \oplus (g_4 + p_4 (g_3 + p_3 \cdot p_2)) = A_5 \oplus B_5 \oplus [p_4' + \underbrace{{P_{3:2}}' (g_4 + g_3)'}_{\text{from carry-merge tree}}]' \end{aligned} \tag{4.22}$$

Finally, S_6 is calculated as:

$${S_6}^0 = A_6 \oplus B_6 \oplus (g_5 + p_5 (g_4 + p_4 \cdot g_3)) = A_6 \oplus B_6 \oplus [p_5' + g_5'(p_4' + \underbrace{(g_4 + g_3)'}_{\text{from carry-merge tree}})]' \tag{4.23}$$

$$\begin{aligned} {S_6}^1 &= A_6 \oplus B_6 \oplus \Big(g_5 + p_5 (g_4 + p_4 (g_3 + p_3 \cdot p_2))\Big) \\ &= A_6 \oplus B_6 \oplus \{[p_5' + g_5'(p_4' + (g_4 + g_3)')]' + \underbrace{P_{5:2}}_{\text{from carry-merge tree}}\} \end{aligned} \tag{4.24}$$

where ${S_6}^1$ reuses part of ${S_6}^0$'s logic to reduce the area. The detailed schematic of a 4-bit

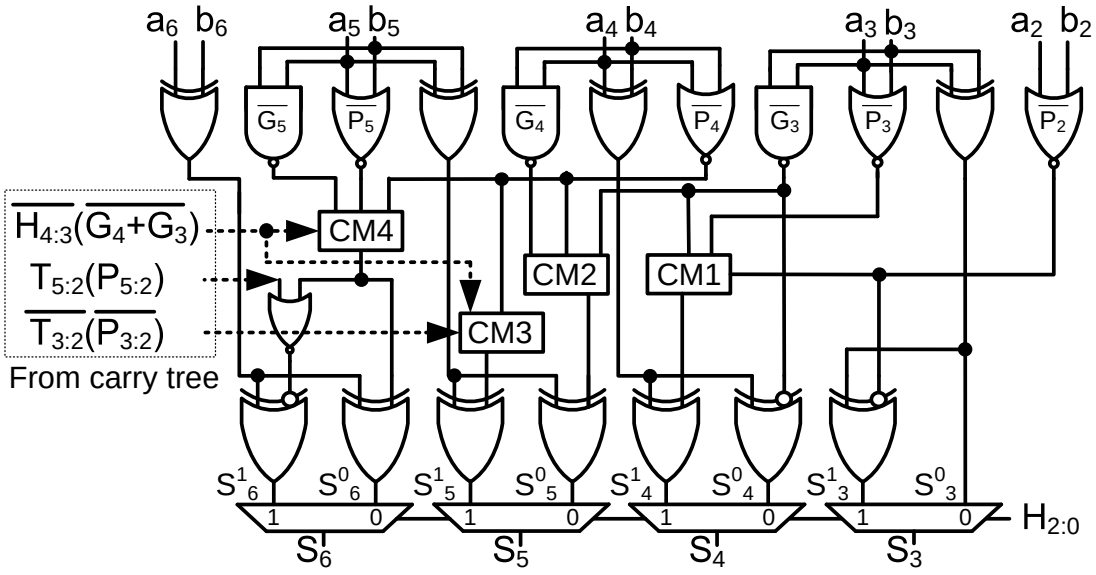


Figure 4.4: Schematic of a 4-bit semi-dynamic conditional-sum generator.

conditional sum generation circuitry is shown in Fig. 4.4, where

$$CM1 = \left(\overline{G_3}\left(\overline{P_2} + \overline{P_3}\right)\right)' \tag{4.25}$$

$$CM2 = \left(\overline{G_4}\left(\overline{G_3} + \overline{P_4}\right)\right)' \tag{4.26}$$

$$CM3 = \left(\overline{P_4} + \left(\overline{H_{4:3}} \cdot \overline{T_{3:2}}\right)\right)' \tag{4.27}$$

$$CM4 = \left(\overline{P_5} + \overline{G_5}\left(\overline{H_{4:3}} + \overline{P_4}\right)\right)' \tag{4.28}$$

Post-layout simulations indicate that the longest delay of the sum block is about 110ps at nominal supply.

4.1.3 Adder Layout

The adder core consists of 4,687 transistors (of which 516 are for input signal buffers) and occupies 120x90 μm^2 in a 65-nm TSMC 1V 1P9M multi-threshold CMOS process. The core is organized in 16 rows with a row height of 3.31 μm. The critical path of the carry-merge tree is implemented with low V_t transistors when necessary to facilitate the carry generation while non-critical path, sum-precomputation blocks, and clock tree are

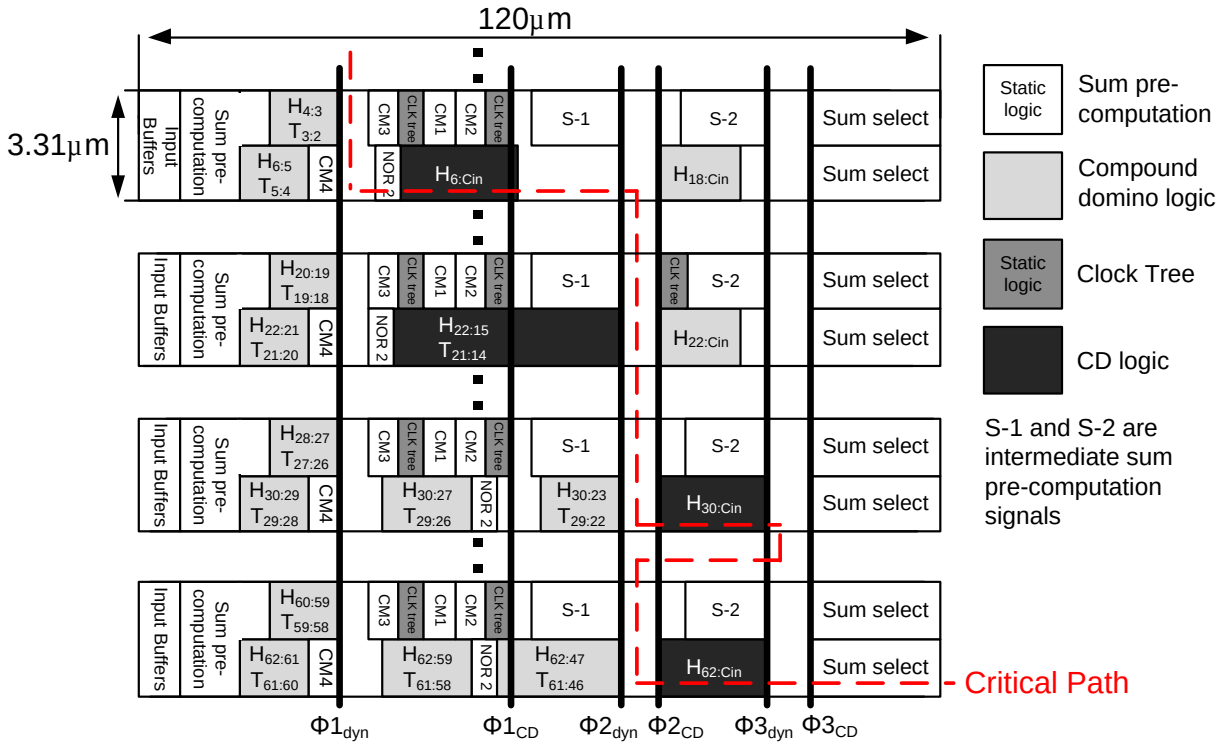
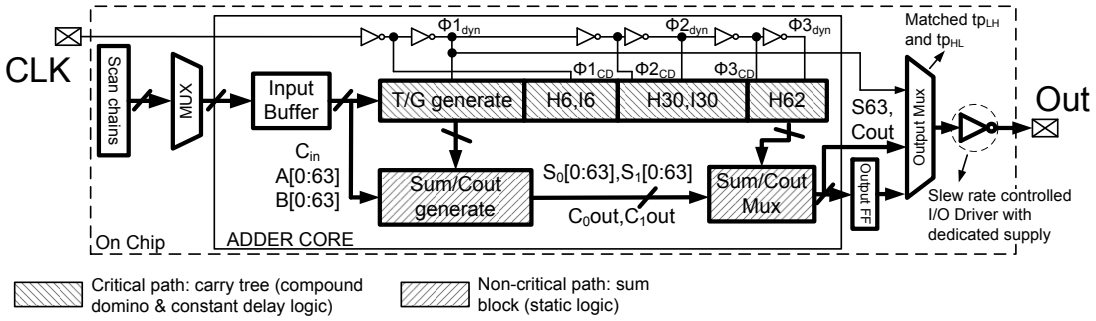


Figure 4.5: Floor plan of the proposed 64-bit adder with CD logic.

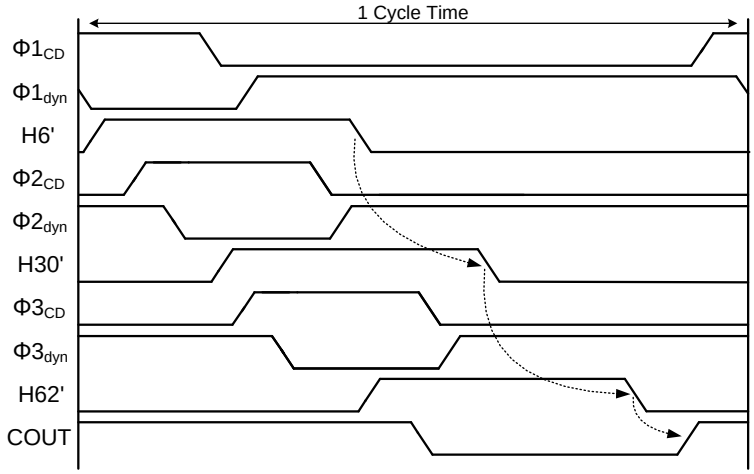
implemented with standard V_t transistors to minimize the leakage current. Multi-cut vias are utilized whenever possible to improve the yield. Horizontal and vertical cell-to-cell routings are done via M4,M6 and M3, M5, respectively, with at least 4× the minimum pitch between adjacent wire on the same metal layer to reduce the coupling effect. For sensitive nodes such as the output of an unprotected dynamic logic (i.e., $T'_{45:30}$), a minimum of 1μm spacing is reserved. Fig. 4.5 shows the floor plan of the proposed adder. Fig. 4.6 shows the chip block diagram and the corresponding timing waveforms.

4.2 Post-Layout Simulation Results

Post-layout simulations of the adder core at extreme PVT conditions are performed, with the results summarized in Table 5.4. The worst glitch is determined by i) supplying the



(a)



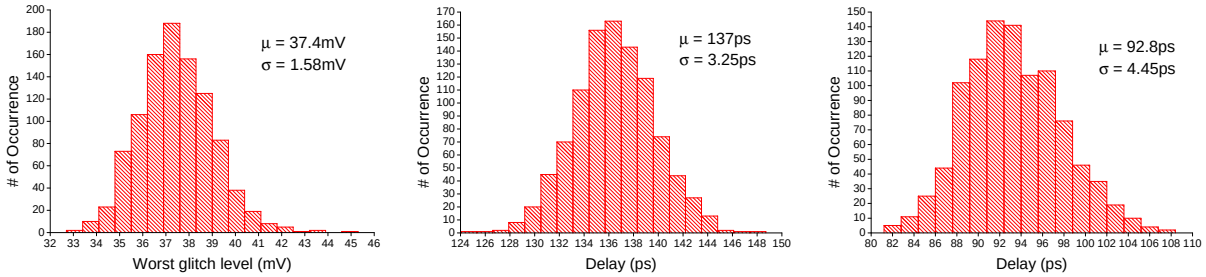
(b)

Figure 4.6: 64-bit adder chip implementation (a) block diagram (b) timing waveform.

worst contention input vectors (A[0:63] = 0000...0, B[0:63] = 111...1) to the adder, ii) comparing the outputs of all the CD logic, and iii) selecting the CD logic output node that has the most excess glitch. With the worst contention input vectors, only one branch of nMOS logic transistors (consider Fig. 4.2(a), only inputs $H_{0:Cin}$ and $H_{2:1}$ remain at logic "1") in CD logic is on. From Table 5.4, it is evident that the adder core is functional in the presence of global variations along with excess static power supply noise and temperature fluctuations. The worst-case glitch level is 172mV, which takes place at FF corner, 110°C, and 1.1V supply. At nominal condition, this adder's delay with the worst delay input vectors (A[0:63] = 1000...0, B[0:63] = 111...1) and worst glitch level are 136ps and 38mV,

Table 4.1: PVT post-layout analysis of the 64-bit adder.

Corner	Temperature (°C)	VDD (V)	Delay (ps)	Worst glitch (mV)	Window width (ps)
TT	27	1	136	38	93
SF	-40	0.9	151	34	115
		1.1	112	42	72
	110	0.9	168	41	119
		1.1	132	81	81
SS	-40	0.9	206	33	158
		1.1	140	41	93
	110	0.9	216	30	156
		1.1	160	40	100
FS	-40	0.9	158	35	124
		1.1	114	43	77
	110	0.9	167	43	123
		1.1	130	95	83
FF	-40	0.9	134	85	97
		1.1	93	55	61
	110	0.9	134	85	97
		1.1	110	172	69

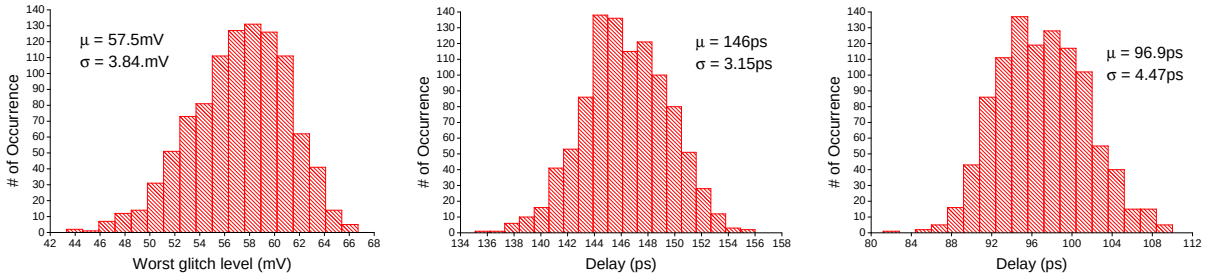


(a) (b) (c)

Figure 4.7: Performance results of (a) worst case glitch level (b) adder delay (c) window width in a Monte-Carlo simulation with 1000 iterations at 27°C. The proposed adder is functional (i.e., no errors) under all iterations.

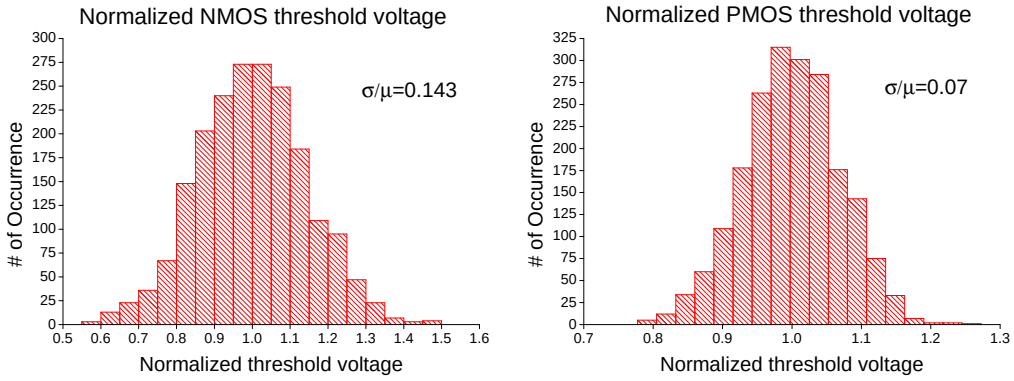
respectively.

Beside global process variations, analysis on local transistor process and mismatch variations is also conducted. Due to the complexity of the overall circuitry, it is not feasible to perform Monte-Carlo simulations on the entire adder core. Instead, non-critical blocks



(a) (b) (c)

Figure 4.8: Performance results of (a) worst case glitch level (b) adder delay (c) window width in a Monte-Carlo simulation with 1000 iterations at 110°C. The proposed adder is functional (i.e., no errors) under all iterations.



(a) (b)

Figure 4.9: Monte-Carlo simulations of normalized threshold voltage of (a) nMOS PDN and (b) pMOS pull-up (*M1*) transistors implemented in a CD logic carry generation block (Fig. 4.2(a)).

such as the sum pre-computation of lower 32-bit (S[0:31]) have been removed to reduce the simulation time while not sacrificing the accuracy of the results. Fig. 4.7 and 4.8 show the performance results of Monte-Carlo post-layout simulations of the partial adder core at 27°C and 110°C with 1000 iterations, respectively. In this setup, each transistor's key parameters, such as the threshold voltage, will randomly deviate from the default value at

TT corner in each Monte-Carlo run. Fig. 4.9 shows the normalized V_t distribution of the nMOS PDN and pMOS pull-up (*M1*, *M2*) transistors implemented in a CD logic block (Fig. 4.2(a)) in a Monte-Carlo simulation with 2000 iterations in order to better understand the degree of the fluctuation. With this transistor variation profile, the proposed adder core is functional (i.e., no errors) in all iterations with an average delay of 137 (146)ps and a standard deviation of 3.13 (3.01)ps at 27 (110)°C.

4.2.1 Post-Layout Simulations with Dynamic Power Supply Noise

Dynamic power supply noise in this thesis refers to the undesirable voltage droop and overshoot at the supply grid contributed by the switching noise of the digital circuitries [60–64]. Typically, the maximum allowable voltage droop/overshoot of the supply grid in a modern IC is ± 10% of the nominal supply voltage (i.e., ± 100mV for a 1-V supply) [65–68]. In this setup, different levels of power supply noise is generated by placing an inductor and a resistor of different values in series between the ideal voltage source and the adder core. The three main components of switching noise are [65, 69, 70]: i) L(dI/dt) noise, which is a consequence of the inductive parasitics (ex, bondwire), ii) IR noise, which is a consequence of the resistive parasitics, and iii) resonance which is a consequence of both the inductive and capacitive parasitics on the supply grid.

Fig. 4.10 displays the first stage clock and final Cout with worst-case delay input vectors at the presence of power supply noise. Fig. 4.10 is generated by superimposing several iterations of simulations with various degree of noise. Clearly, even with a voltage fluctuation of more than 200mV, the adder core is still functional, indicating that CD logic is robust with the presence of voltage droop/overshoot, where the window duration increase/decrease accordingly.

4.3 Silicon Results

Fig. 4.11 shows the die photo of the proposed 64-bit Ling adder. The delay of the proposed 64-bit adder is measured using the following approach: The adder is triggered by an off-chip

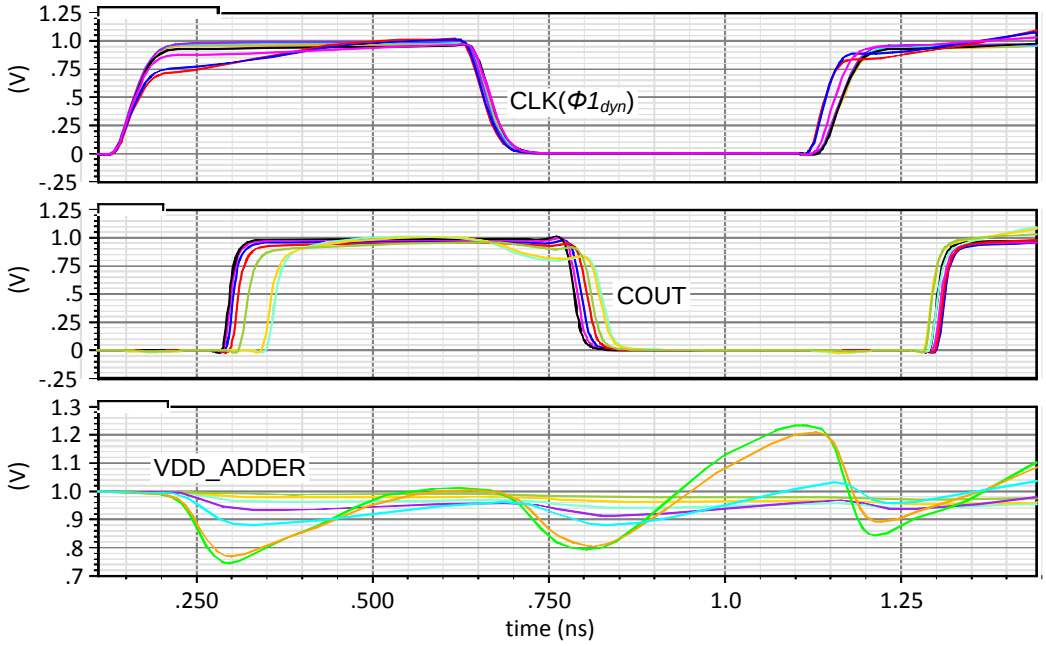


Figure 4.10: Waveforms of first stage clock to final Cout of a 64-bit Ling adder with CD logic under various levels of power supply noise. The adder can function correctly even with more than 200mV of voltage droop/overshoot.

100-MHz reference signal. The timing difference of both the first stage clock signal and the critical path, Cout, signal to this reference signal are measured, and then subtracted from each other to obtain the adder delay. Both signals traverse through the same output driver and matched MUX (Fig. 4.6(a)). The slew-rate-controlled output driver has a dedicated power supply that is isolated from the rest of the chip to minimize supply noise. The conceptual and actual delay measurement waveforms are shown in Fig. 4.12. Notice that during the actual testing only one of the signals ($\phi 1_{dyn}$ or Cout) can be observed, since both of them traverse through the same MUX and output driver. Fig. 4.12(b) is generated by superimposing both signals' waveforms on the oscilloscope. A fixed delay offset between CLK (IN_CLK_PAD) to both output signals are also applied to achieve a better visual representation. The worst power consumption is also measured at 100 MHz,

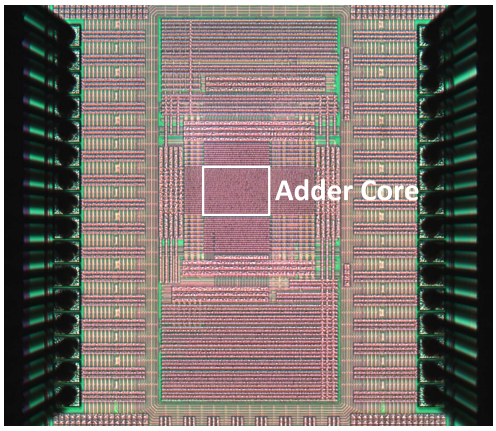
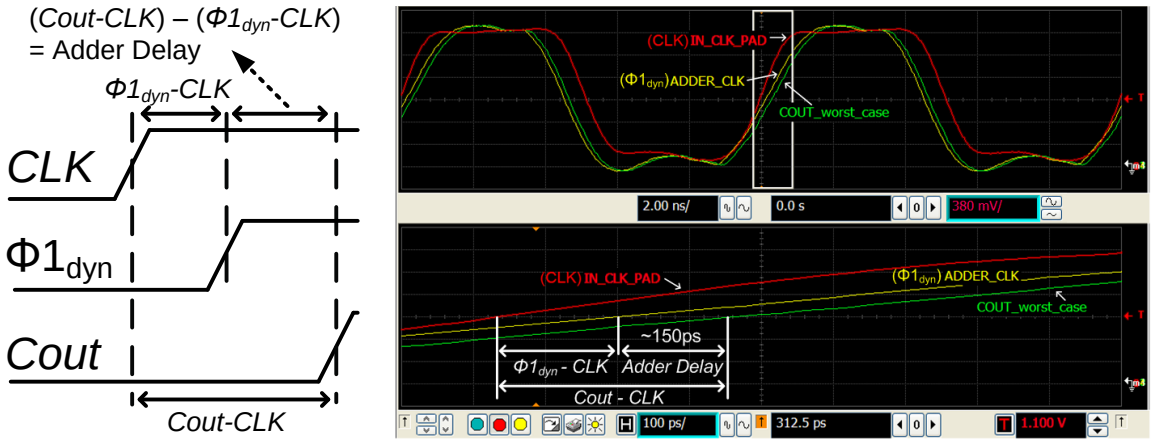


Figure 4.11: Die photo of the 64-bit adder with CD logic in a 65-nm CMOS process.



(a) (b)

Figure 4.12: (a) Conceptual and (b) actual delay measurement waveforms. The zoom-in window (bottom) indicates that the proposed 64-bit adder delay is approximately 150ps.

and then extrapolated to full-speed power consumption by multiplying it by a scaling factor, calculated as:

$$\frac{1/(adder\ delay)}{100MHz} \tag{4.29}$$

which assumes that dynamic power is the dominant component.

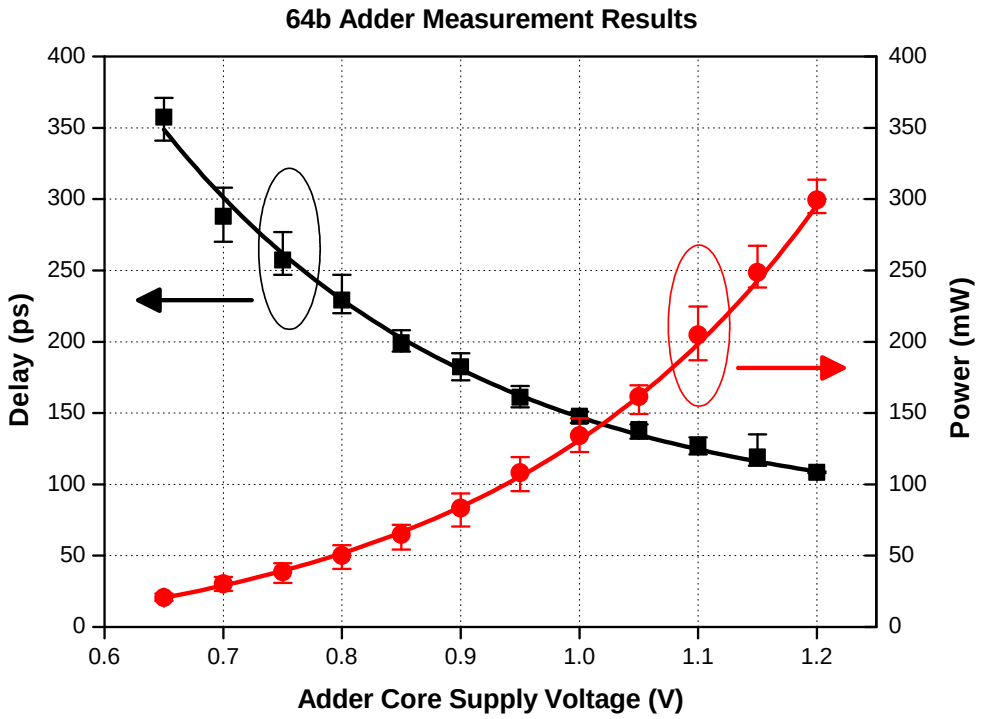


Figure 4.13: Delay and power vs. adder core supply voltage measurement results.

All eight fabricated chips are tested and functional, with measurement results summarized in Fig. 4.13. *No* post silicon calibrations were carried out during the measurement process. At the 1-V nominal supply, the adder core runs at an average delay of 148 ps for the slowest input vector with a worst-case power consumption of 135 mW and a leakage of 0.22 mW at room temperature. The delay result is approximately 8% slower than that of post-layout simulation, most likely due to the power supply noise during the evaluation period as a result of bondwire's inductance. The power includes the adder core, input signal buffers, and clock generation circuitries. At the 1.2-V supply, the delay decreases to 108 ps, with a worst-case power and a leakage power of 300 mW and 0.58 mW, respectively. Finally, Table 4.2 compares this work with state-of-the-art designs. To ensure a fair comparison, the delay and power consumptions are normalized to a 65-nm equivalent

Table 4.2: 64-bit adder chip performance comparisons.

	[71]	[59]	[22]	[72]	[19]	This work
Technology (nm)	250	225	90	180	90	**65**
Supply Voltage (V)	2.5	1.8	1.3	1.8	1	**1**
Architecture	Adder	Adder	ALU[1]	Adder	Adder[3]	**Adder**
Test Chip	*Yes*	*Yes*	*Yes*	*No*	*Yes*	***Yes***
Number of bits	64	64 64	64	64	**64**	
Delay (ps)[†]	1500(390)	470(136)	250(181)	1500(542)	240(173)	**148**[4]
Power (mW)[†]	300(12.5)	N/A	300(128)[2]	N/A	260(188)[2]	**135**[2,4]
PDP (pJ)[†]	450(4.9)	N/A	75(23)	N/A	62.4(33)	**20**
Area (μm^2)	1600 X 275	N/A	280 X 260	13250	417 X 75	**120 X 90**
Leakage Power (mW)	N/A	N/A	9.6	9.3	2.3	**0.22**

[†] Numbers in brackets are scaled to a 65-nm equivalent CMOS process

[1] require two different voltage supplies [2] worst power consumption

[3] requires a post calibrated clock signal to feed the last stage dynamic MUXs

[4] average measurement results of eight tested chips

CMOS process with 1V supply voltage using the following formulas:

$$Delay_{norm} = Delay \times \frac{65nm}{Technology} \tag{4.30}$$

$$Power_{norm} = Power \times \left(\frac{1V}{Voltage}\right)^2 \times \frac{65nm}{Technology} \tag{4.31}$$

The designed 64-bit Ling adder with CD logic demonstrates lowest leakage power while achieving the shortest delay compared to recently published silicon results. The power consumption of the proposed adder is also comparable with others, leading to a lower power-delay product.

4.4 Conclusion

A 64-bit single-cycle S-4 Ling adder is described in this thesis. The carry-merge tree of this adder has been redesigned to utilize CD logic to facilitate the carry generation and results in only twelve transistors in the carry-merge tree's critical path. CD logic has demonstrated speed advantage and energy-efficiency over conventional logic styles and its robustness has been verified in extensive post-layout simulations under various process, voltage, and temperature conditions. A redesign of a partial static sum-precomputation block which utilizes dynamic signals coming from the carry-merge tree is also described.

The proposed adder is fabricated in TSMC 1-V 1P9M multi-threshold CMOS process. All eight test chips are functional with no post calibration involved. At the nominal supply, the proposed adder shows a delay of 148 ps with a worst case power consumption and a leakage power of 135 mW and 0.22 mW, respectively.

Chapter 5

64-Bit Energy-Efficient Tree Comparator

5.1 Introduction

Binary comparators are one of the most fundamental components in digital systems with many applications such as the decoding of the x86 instruction sets, the renaming of the register files in a superscalar system, and the number magnitude comparison in an arithmetic logic unit. Conventionally, high-performance binary comparison is achieved using a high speed adder, at the expense of both power consumption and area. Wang *et al.* [73] proposed a high-performance, tree-structure comparator using all-N-Transistor (ANT) dynamic CMOS logic. This design however, may not be suitable for a single-cycle operation, because of the heavy pipelining (3.5 clock cycles) required for ANT logic. Huang and Wang [74] proposed a single-cycle, two-phase comparator using a priority-encoding algorithm and has shown 16% performance enhancement over [73]. A parallel-MSB-checking algorithm is proposed in [75] and [76] by introducing a new static priority encoder and a MUX-based comparator structure. This implementation achieves superior performance compared to the previous designs at the expense of twice the number of transistors. In 2007, Kim and Yoo [77] proposed a new comparator with bitwise Competition Logic (BCL). This comparator utilizes BCL to detect the earliest first "1" away from the most-significant-bit

(MSB) after pre-encoding the inputs. This design achieves the lowest transistor count and shows a 16% delay improvement. Recently, tree-based comparators have been proposed in [78] and [79] where dynamic Manchester structures are used to facilitate the comparison process.

All of the above works achieve high-performance operations using dynamic logic. While dynamic logic has demonstrated superior performance compared to static logic, it is not suitable for low-power operation because its data activity factor (α) is always 0.5. On the other hand, static logic has an empirical α of close to 0.1 [16], making it advantageous in terms of power consumption. Designs in [74], [78], [77] and [79] may not be suitable for static logic implementation, primarily because of the tall transistor stack height. In addition, a higher stack height is also less attractive in a deep sub-micron process, where the V_{DD}/V_t ratio (≈ 3 for 65nm) is lower compared to an earlier technology, because transistors will exit the saturation mode sooner and be forced to operate in the linear region [80].

In this chapter, a new 64-bit tree structure comparator with a pre-encoding scheme to achieve a maximum stack height of two is described. This design is particularly suitable for implementation with pass transistor and/or static logic to ensure low-power consumption. Moreover, this chapter provides a comprehensive performance analysis of state-of-the-art comparators in 180-nm, 90-nm, and 65-nm CMOS processes. This chapter demonstrates that the proposed static logic implementation achieves similar delay performance compared to other papers' dynamic logic designs. Since static logic is more power efficient, especially at lower data activity factors, the proposed comparator shows significant energy efficiency compared to others. Both the static 64-bit comparator and an improved comparator, where CD logic is exclusively implemented in timing-critical stages to reduce the delay without sacrificing the energy consumption, are realized in a 1-V, 65-nm CMOS process. At 1-V supply, the proposed comparators measured delay is 167 ps, and has an average power and a leakage power of 2.34 mW and 0.06 mW, respectively. At 0.3-pJ iso-energy or 250-ps iso-delay budget, the proposed comparator with CD logic is 20% faster or 17% more energy-efficient compared to a comparator implemented with just the static logic.

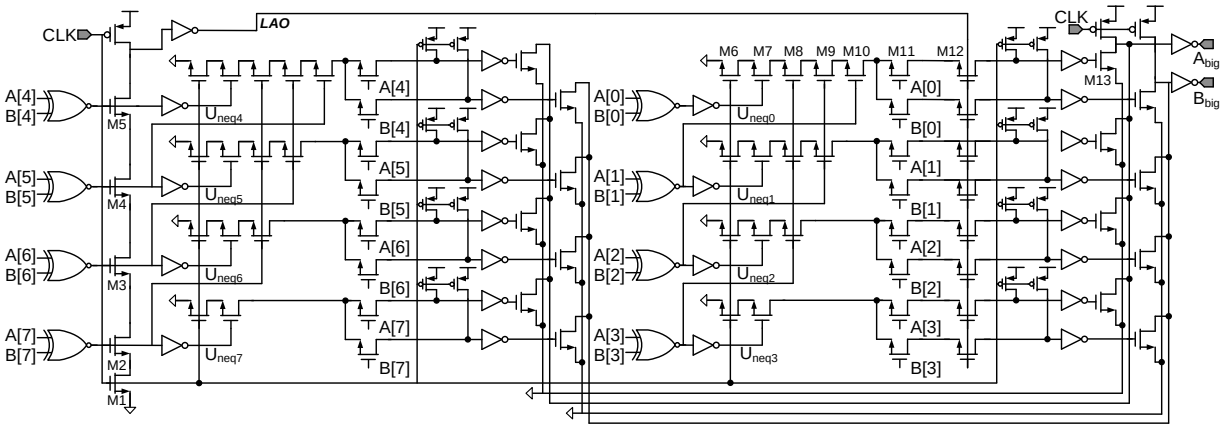


Figure 5.1: Schematic of an 8-bit priority encoder [74].

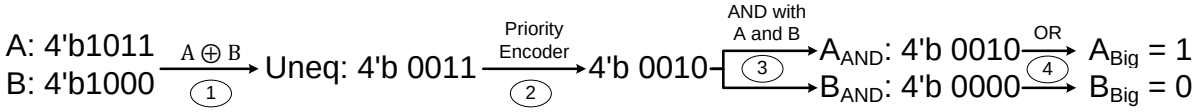


Figure 5.2: A 4-bit numerical example of the algorithm in [74].

5.2 Existing Comparator Designs

5.2.1 Priority-Encoding-Based Comparator

A priority-encoding-based comparator [74] [76] relies on a priority encoder to decode the first bit away from the MSB that can distinguish the relationship of the two numbers. An example of an 8-bit priority encoder design and a 4-bit comparison algorithm [74] are shown in Fig. 5.1 and 5.2, respectively.

Consider a scenario where $A_{7:0} = 000...1$ and $B_{7:0} = 000...0$. In this case, U_{neq0} is logic "1", and the signal LAO is pulled high through M1-5 and an inverter. Consequently, M13 is turned on via transistor M6-12 and an inverter, thus discharging the internal node of the 8-input dynamic OR gate and pulling A_{big} high. The 64-bit comparator consists of two stages, with eight priority encoders in parallel in the first stage driving another identical priority encoder in the second stage. In [76], a parallel-MSB-checking algorithm with pre-

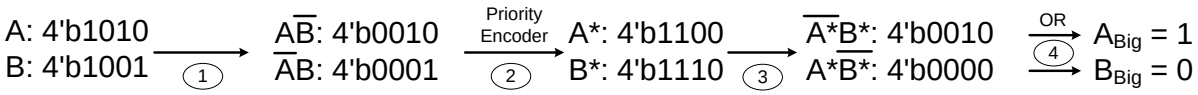


Figure 5.3: A 4-bit numerical example of the algorithm in [76].

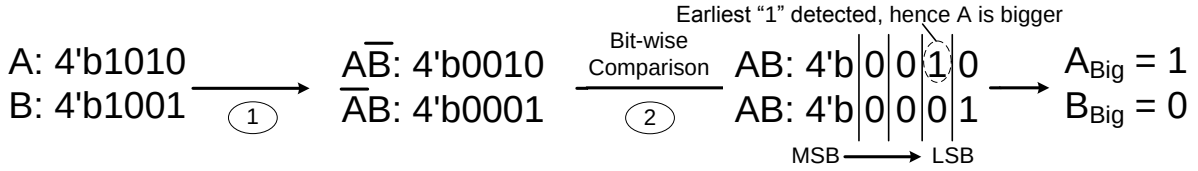


Figure 5.4: A 4-bit numerical example of the algorithm in [77].

encoding scheme and a static priority decoder is implemented. A 4-bit numerical example of this scheme is shown in Fig. 5.3. The key difference of this design compared to [74] is that it sets all the preceding bits of the MSB (but not including the MSB itself) to "1" and reset all the other bits to "0" [76], as shown in Step 2 of Fig. 5.3. In order to speed up this process, [76] uses many OR dynamic logic in parallel at the expense of excess power dissipation. Another encoding process then takes place (step 3) followed by a dynamic OR gate to determine the larger number. Similar to [74], the 64-bit comparator is consisted of two stages, with the first stage being eight 8-bit comparator running in parallel. Another key difference in this design is that a static priority decoder runs in parallel with the first stage to determine the first group of bits away from the MSB that can determine the comparison process. Therefore, the second stage is simplified to be an 8 to 1 dynamic MUX and thus achieves high-performance operation.

Designers who wish to implement [74] and [76] with static logic will incur a substantial hardware overhead because both designs heavily rely on dynamic logic to achieve high-speed operations. For instance, static implementations of 8-input dynamic OR gates and MUXs required for both architectures will result in significant area and delay penalties.

5.2.2 Bit-Wise Competition Logic (BCL) Based Comparator

The BCL based comparator compares two integers using the location of the first "1" away from the MSB. A 4-bit numerical example is shown in Fig. 5.4. After the encoding process, the bit comparison is performed with the initial position at MSB to detect the first "1". If "1" is detected, that input is decided to be the larger one; otherwise, BCL moves to the next lower bit and repeats the process until the first "1" occurs.

BCL is not particularly suitable for static logic, primarily because its principle of operation is very similar to that of a dynamic MUX. Its high-performance operation relies on the precharge of the internal node, thus eliminating the need for a very tall pMOS transistor stack. Hence, converting BCL to a static-compatible design for the low-power purpose will most likely cause performance degradation similar to that of a dynamic MUX.

5.2.3 Tree Structure Based Comparator

[78] and [79] proposed a tree-based comparator that utilizes dynamic Manchester adders to reduce the number of critical stages to three. A static implementation of this design inevitably incurs performance and area penalties. For instance, a static Manchester adder requires an additional *delete* signal and has a tall pMOS transistor stack.

5.3 Proposed Radix-2 Tree Structure Comparator

The proposed high-performance tree-based comparator is inspired by the fact that G (*generate*) and P (*propagate*) signals can be defined for binary comparisons, similar to the G and P signals for binary additions. Hence, the following key observation is made:

- A binary comparator is essentially a subset of the carry-merge tree in a parallel-prefix adder, where only the final carry-out signal is necessary to interpret the result.

5.3.1 Basic Design Principle

A two 2-bit binary number (A_1A_0 and B_1B_0) comparison can be realized with Equation (5.1):

$$B_{Big} = \overline{A_1}B_1 + \overline{(A_1 \oplus B_1)}\left(\overline{A_0}B_0\right) \tag{5.1}$$

$$EQ = \overline{(A_1 \oplus B_1)} \cdot \overline{(A_0 \oplus B_0)} \tag{5.2}$$

If $B > A$, then "B_{Big}, EQ" is "1,0". "B_{Big}, EQ" is "0,0" if $A > B$, and "0,1" if $A = B$. A closer look at Equation (5.1) reveals that it is analogous to the carry signal generation in binary additions. Consider the carry generation:

$$C_{out} = AB + (A \oplus B)C_{in} = G + PC_{in} \tag{5.3}$$

where A, B are the binary inputs, C_{in} is the carry input, C_{out} is the carry output, and G and P are the *generate* and *propagate* signals, respectively. Compared Equation (5.1) and (5.3), one can then define $G_1 = \overline{A_1}B_1$, $EQ_1 = \overline{(A_1 \oplus B_1)}$, and $C_{in} = \left(\overline{A_0}B_0\right)$ for B_{Big}. Equation (5.1) may not be suitable for high-performance operation when implementing with static logic, due to the tall transistor stack height and a complicated XNOR gate. An encoding scheme is employed to mitigate this problem. The encoding equation is given as:

$$G_{[i]} = \overline{A_{[i]}}B_{[i]}, \qquad EQ_{[i]} = \overline{A_{[i]} \oplus B_{[i]}} \tag{5.4}$$

where $i = 0...63$. The radix-2 comparison in Equation (5.1) and (5.2) can then be simplified to:

$$B_{Big[2j+1:2j]} = G_{[2j+1]} + EQ_{2j+1}G_{[2j]} \tag{5.5}$$

$$EQ_{[2j+1:2j]} = EQ_{[2j+1]} \cdot EQ_{[2j]} \tag{5.6}$$

for $j = 0...31$, which only requires ten transistors with a maximum stack height of two.

The G and P signals can be further combined to form *group* G and P signals.

$$\begin{aligned} B_{Big[3:0]} &= \overline{A_3}B_3 + \left(\overline{A_3 \oplus B_3}\right) \cdot \left\{ \overline{A_2}B_2 + \left(\overline{A_2 \oplus B_2}\right) \left[\overline{A_1}B_1 + \left(\overline{A_1 \oplus B_1}\right) \left(\overline{A_0}B_0\right) \right] \right\} \\ &= G_3 + EQ_3 \left(G_2 + EQ_2 \left(G_1 + EQ_1 C_{in}\right)\right) \\ &= B_{Big[3:2]} + EQ_{3:2} B_{Big[1:0]} \qquad (5.7) \\ B_{Big[7:4]} &= B_{Big[7:6]} + EQ_{7:6} B_{Big[5:4]} \qquad (5.8) \\ B_{Big[7:0]} &= B_{Big[7:4]} + EQ_{7:4} B_{Big[3:0]} \qquad (5.9) \end{aligned}$$

and for the equal (EQ) function,

$$\begin{aligned} EQ_{[3:0]} &= \left(\overline{A_3 \oplus B_3}\right) \cdot \left(\overline{A_2 \oplus B_2}\right) \cdot \left(\overline{A_1 \oplus B_1}\right) \cdot \left(\overline{A_0 \oplus B_0}\right) \\ &= EQ_{[3]} \cdot EQ_{[2]} \cdot EQ_{[1]} \cdot EQ_{[0]} \\ &= EQ_{[3:2]} \cdot EQ_{[1:0]} \qquad (5.10) \\ EQ_{[7:4]} &= EQ_{[7:6]} \cdot EQ_{[5:4]} \qquad (5.11) \\ EQ_{[7:0]} &= EQ_{[7:4]} \cdot EQ_{[3:0]} \qquad (5.12) \end{aligned}$$

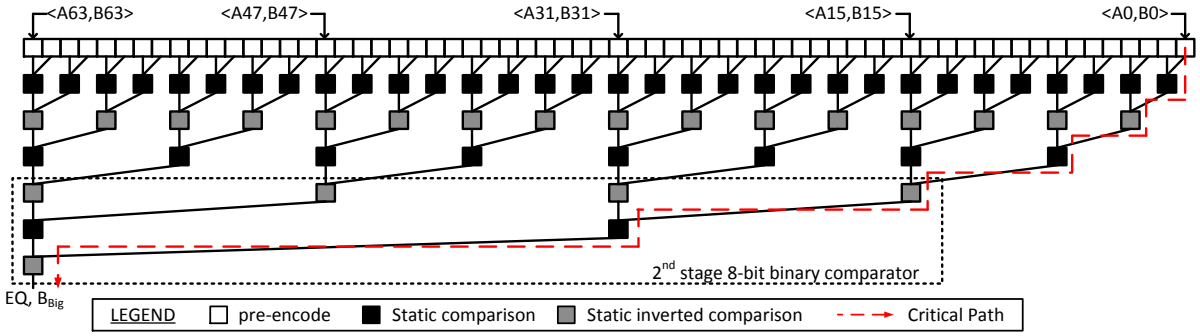
Finally, B_{Big} and EQ in a 64-bit comparator are computed using Equation (5.13) and (5.14).

$$B_{Big[63:0]} = G_{63} + \sum_{k=0}^{62} \left(G_k \cdot \prod_{m=k+1}^{63} EQ_m \right) \qquad (5.13)$$

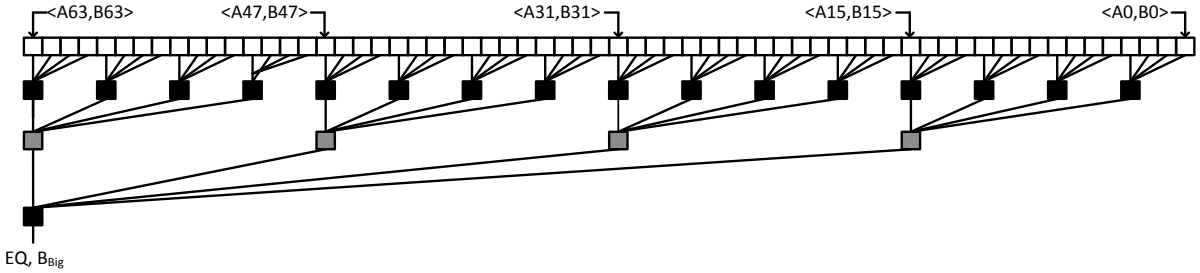
$$EQ_{[63:0]} = \prod_{m=0}^{63} EQ_m \qquad (5.14)$$

5.3.2 Comparator Tree Design Analysis

Several 64-bit comparator tree designs are analyzed and implemented in order to determine the most energy-efficient tree structure. Notice that unlike the carry-merge tree in an adder, the variety of tree structure in a comparator is relatively less, because comparator belongs to the family of "parallel-reduction structure" [81] where only nodes B_{BIG} and EQ are derived at the end.



(a)



(b)

Figure 5.5: (a) Radix-2, and (b) radix-4 64-bit comparator tree diagrams.

Comparator Tree Radix

The radix (also known as the valency [82]) defines the number of bits that is merged for a given stage. In a radix-2 (radix-4) configuration, two (four) bits are compared at every stage and results in a $log_2 64 = 6$ ($log_4 64 = 3$) stages comparison for a 64-bit comparator. Compared to a radix-2 structure, a radix-4 design reduces the number of stages by half at the expense of 2× the transistor stack height per stage. In this work, both radix-2 and radix-4 comparators are analyzed, with the corresponding tree diagrams shown in Fig. 5.5(a) and 5.5(b).

Encoding scheme

The encoding scheme (Equation (5.4)) simplifies the first stage comparison and reduces the maximum transistor stack height at the expense of adding an additional stage in the tree. Without the encoding scheme however, implementations of Equation (5.13) and (5.14) result in a maximum transistor stack height of four, similar to the case of a radix-4 configuration.

5.3.3 Comparator Sizing Strategy

The four 64-bit comparator trees that are analyzed are as follows:

1. A radix-2 tree with the encoding scheme implemented in pass transistor logic style.
2. A radix-2 tree with the encoding scheme implemented in static logic style.
3. A radix-2 tree without the encoding scheme.
4. A radix-4 tree with the encoding scheme implemented in pass transistor logic style.

To ensure a fair comparison, it is important to size each tree appropriately to achieve the minimum delay for a given energy constraint. Such a multi-dimensional space optimization problem can lead to considerable run time even for a simple circuit, and may not be computationally feasible for a large circuit such as a 64-bit binary comparator. In order to speedup the optimization process, we reduce the design-freedoms without compromising the optimization results by employing the following heuristic approaches.

Group Sizing

Binary comparator, similar to an adder, is a suitable candidate for group-sizing strategy, where identical logic gates in each stage are grouped together and sized the same. For a radix-2 comparator tree with the encoding scheme, this approach constraints the number of sizing variables to seven.

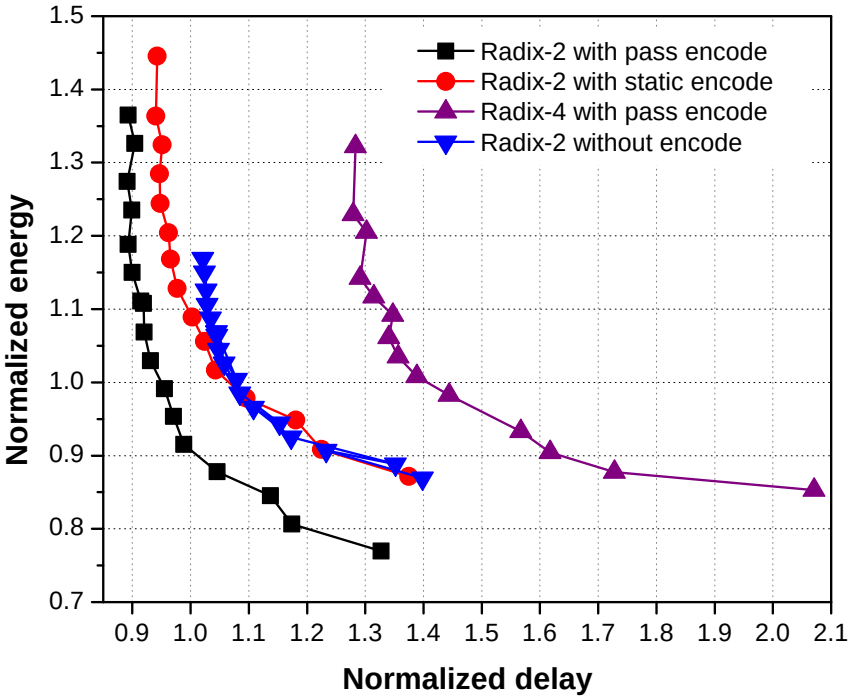


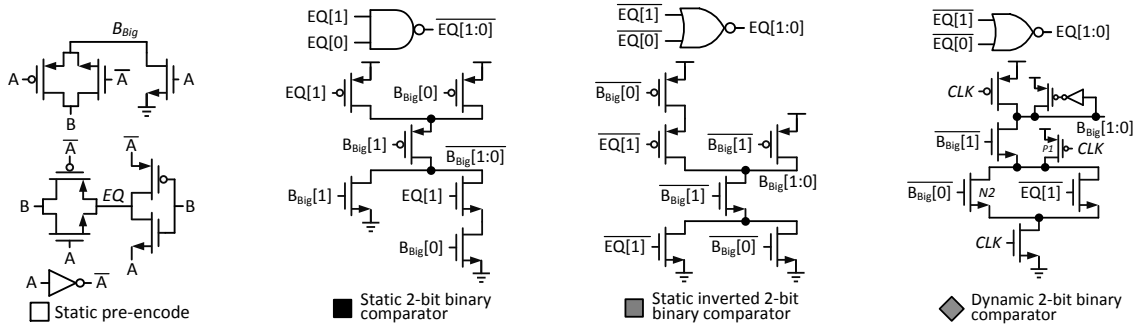
Figure 5.6: Energy-delay tradeoff curves of 64-bit binary comparators implemented with various tree designs in a 65-nm CMOS process.

Sizing Constraints

Due to the parallel-reduction characteristic of a comparator tree, the first two stages of the tree occupy approximately 75% of the overall circuit area. Therefore, a suitable energy-efficient sizing approach with delay performance taken into consideration is to set the first two stages with small transistor sizing, and then progressively increases the transistor sizes as the stage increases.

5.3.4 Optimization Results

Fig. 5.6 shows the energy-delay tradeoff curves of the four 64-bit binary comparators being analyzed, where each point is generated using the Cadence Virtuoso Analog Circuit Optimizer with the sizing constraint/strategy as described above. For the chosen CMOS



(a) (b) (c) (d)

Figure 5.7: Schematics of (a) pre-encode, (b) static, (c) static inverted, and (d) dynamic 2-bit binary comparator.

process, radix-2 configuration is a better alternative than its radix-4 counterpart. The radix-2 tree with static encoding scheme exhibits similar energy consumption compared to the radix-2 tree without the encoding scheme at longer delay constraints; however, if the required delay constraint reduces, the former provides a better design tradeoff. This is because direct implementation of Equation (5.13) results in a tall transistor stack at the first stage, similar to the case of a radix-4 merging. By implementing the encoding scheme in pass transistor style, an additional 10% delay improvement can be obtained under the same energy constraint. On average, the proposed radix-2 comparator with the encoding stage implemented with pass transistor logic style achieves 45% delay improvement at any given energy constant compared to a radix-4 design.

5.3.5 Proposed 64-bit, Radix-2 Binary Comparator

The proposed 64-bit comparator tree is shown in Fig. 5.5(a) and the schematics of corresponding circuits are shown in Fig. 5.7(a)-5.7(c). The critical path consists of one stage of pre-encoding and six stages of 2-bit binary comparison circuits. A pass transistor logic style is employed in the first stage pre-encoding circuitry to reduce the number of transistors required from sixteen (static logic) to nine (including inverters). For the comparison

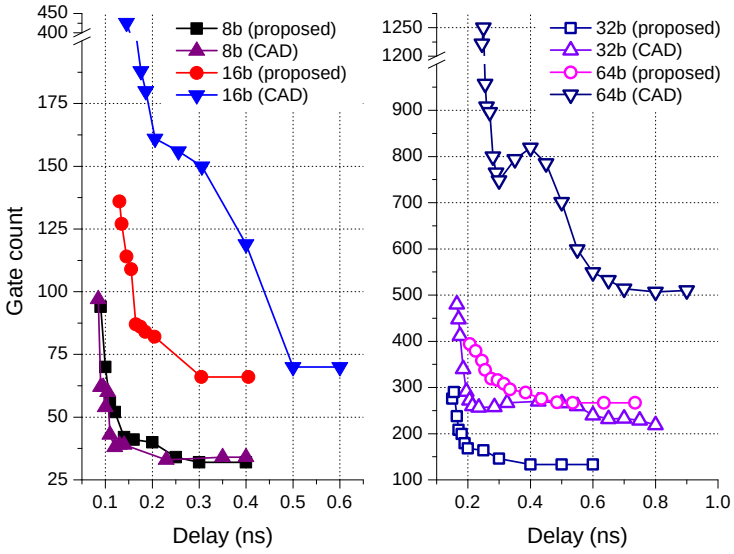


Figure 5.8: Delay vs. area tradeoff curves of the proposed comparator and the comparator generated automatically by the CAD tool using TSMC's 65-nm standard-cell library.

generation circuitry, inversion property is utilized to minimize the number of stages in the critical path and to also reduce the total power consumption by eliminating unnecessary inverters. Unlike other designs [74] [76] [77] [78], the proposed comparator is static logic compatible and has a maximum stack height of only two. As discussed previously, static logic is particularly attractive for low-power applications due to its lower data activity, α, which is defined as

$$\alpha = \frac{\#\,of\ signal\ transitions}{\#\,of\ signals \times \#\,of\ clock\ cycles} \tag{5.15}$$

α essentially represents the number of signal transitions divided by the product of number of signals and number of clock cycles.

5.3.6 Proposed Comparator vs. Synthesized designs

The standard-cell based implementation of binary comparators with different bit-widths utilizing the proposed radix-2 comparator architecture is implemented and compared with

the binary comparators that are generated by the computer-aided design (CAD) tool in a TSMC 65-nm 1-V CMOS process. All designs are first described using Verilog and then synthesized using Synopsys Design Compiler. The only difference is that for binary comparators utilizing the proposed comparator architecture, we enforce the CAD tool to implement the proposed radix-2 architecture while for the other cases, Design Compiler will choose a suitable implementation using DesignWare® IP portfolios based on a given delay constraint. The delay vs. area tradeoff curves of the binary comparators with different bit-width are illustrated in Fig. 5.8. For low bit-width applications (ex., 8-bit) the proposed design is not the preferred choice. As the bit-width increases, the proposed comparator architecture outperforms the designs generated by the CAD tool. For instance, for a 32-bit comparator with a delay requirement of 0.2 ns, the proposed comparator requires 40% less number of gates than the comparator generated automatically by the CAD tool.

5.4 Power and Delay Comparative Analysis

This section compares the proposed 64-bit tree comparator with the existing comparator designs. All simulation runs are done in schematic level (transistor netlists) in the Cadence design environment using three different CMOS technologies at nominal supply voltage at 27°C with TT corner. To ensure a fair and accurate comparison, all comparator architectures are reproduced from prior works and transistor sizings are determined through iterative simulation processes aiming to optimize PDP. The measured power consumption includes the clock tree and data buffers, which are both sized to drive a FO4 load. The clock and data frequencies are set to 100MHz. Table 5.1 summarizes the key parameters and the fixed output load of the three CMOS technologies. The delay is measured at the 50% point of the rising edge of either the CLK or data to the 50% point of the rising edge of the comparator output with the worst-case delay vector: $B_{[63:0]} = 000....0 \rightarrow 000...1$. $A_{[63:0]} = 111....1 \rightarrow 000...0$.

Table 5.1: CMOS technology key parameters and output load.

Technology (nm)	180	90	65
V_{DD} (V)	1.8	1	1
$V_{tn}(V_{tp})$ (V)	0.41(0.43)	0.35(0.32)	0.32(0.34)
Output Load	$25f$	$20f$	$15f$

Table 5.2: Summary of simulated delay and power results of various 64-bit comparators.

Publication	Process (nm)	Delay (ps)	Worst Power (μW)	50% Power (μW)	25% Power (μW)	10% Power (μW)	# of Transistors	Total Width	Leakage (μA)
This work	180	642	1224	783	386	153	1206	2988	—
	90	240	207	132	68	31		2445	5.4
	65	166	189	123	70	40		2013	13.7
Frustaci [79]	180	633	1133	964	780	675	1365	2334	—
	90	352	283	261	220	192		1856	4.6
	65	211	216	198	168	148		1467	11.3
Lam [76]	180	453	3102	4268	3939	3731	3386	5119	—
	90	180	844	737	679	643		4227	10.4
	65	124	608	511	466	439		3340	17.5
Kim [77]	180	1005	2194	2115	1850	1691	964	2469	—
	90	386	401	322	280	255		1934	5.5
	65	268	339	263	226	203		1691	13.5
Huang [74]	180	752	1364	1020	775	652	1640	2382	—
	90	311	307	259	199	163		2014	7.8
	65	212	234	202	163	139		1737	17.3

5.4.1 Experimental Results

Table 5.2 summarizes the delay and power results for various 64-bit comparators. Average power consumption except the worst case is measured with 500 random input vectors at various input data activities (α).

Worst case power is measured with the following input vectors: $A_{[63:0]} = 111....1 \rightarrow 000...0$, $B_{[63:0]} = 000....0 \rightarrow 111...1$. Power measurement at 10% α is considered as a more realistic situation, since most of the logic blocks in digital systems nowadays are implemented with static logic. Simulation results at various α demonstrate that, even at

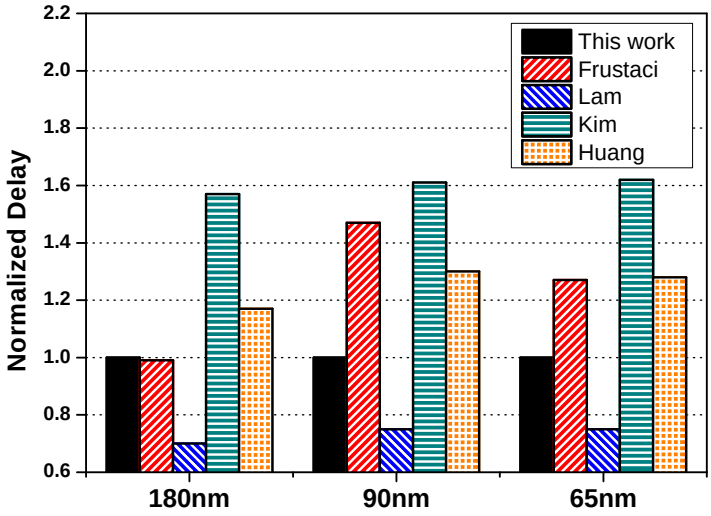


Figure 5.9: Normalized delay of various comparators.

25% α, the proposed comparator still outperforms the rest of the designs in terms of power, PDP and EDP.

Fig. 5.9 illustrates the normalized delay of the comparators across the three technology nodes considered. [76] demonstrates the best performance, largely due to the numerous dynamic logic blocks required for this design. The proposed design is approximately 30% slower than [76] but is at least 17% faster than the rest. This design is slower than [76] primary because energy-efficiency is also taken into consideration. If performance is the primary concern, the proposed comparator can also be implemented with dynamic logic as well as a larger transistor sizing. Simulation results indicate that in this setup, the proposed comparator is 16% faster than [76]. All the designs exhibit similar performance trends across the three technology nodes considered, with the exception of [79]. As the technology node scales down, the delay of [79] degrades, because of its tall stack height (6 transistors). [74] is less susceptible to this problem, despite its tallest 7-transistor stack height. With the worst case delay vectors, signals driving transistors M6-11 in the priority encoder (Fig. 5.1) will arrive much earlier than M12. Hence, when LAO is pulled high, M6-11 already discharge the internal nodes to ground. A similar situation also takes place in the priority encoder in [76]. On the other hand, the Manchester adder in [79] can not

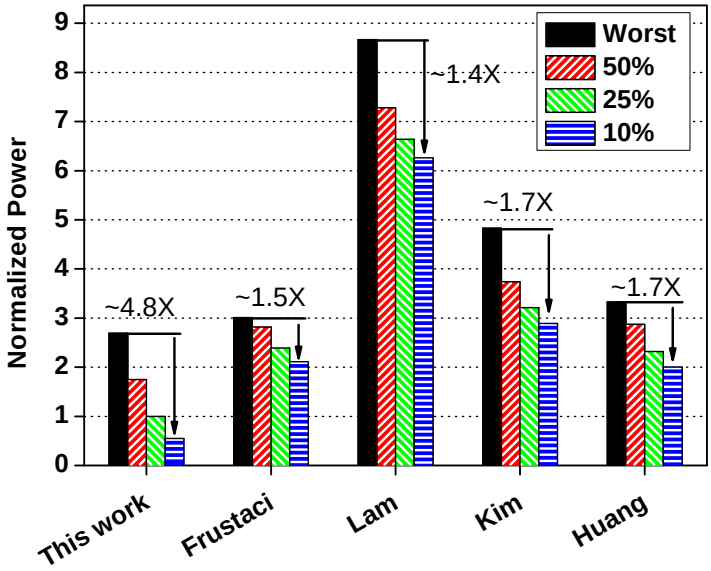


Figure 5.10: Normalized power of various comparators with respect to the proposed 64-bit comparator at 25% α vs. different input data activity in 65-nm.

begin to discharge until the slowest signal (CLK) arrives at the footer transistor.

Fig. 5.10 reveals the normalized power of various comparators with respect to the proposed work at 25% α in a 65-nm technology. Clearly, the proposed comparator's power consumption is a strong function of α, with approximately 4.8× power reduction from the worst case to average power dissipation at 10% α. The rest of the designs are less affected by α, because of the dynamic logic's higher data activity. At 25% α, the power consumption of the proposed design is 2.3× to 6.6× lower than the other designs. At 10% α, the power reduction of the proposed work ranges from 3.5× to 11×. This is largely because the leakage power has become a more significant portion of the overall power consumption.

Fig. 5.11 and 5.12 illustrate the normalized PDP and EDP of various 64-bit comparators for 25% α, respectively. The proposed comparator achieves the lowest PDP and EDP across all the technology nodes considered. At 65 nm with 25% α, the proposed design is approximately 3× and 3.7× more PDP and EDP efficient, respectively. The proposed comparator's leakage power is approximately 21% more than that of [79] at 65 nm (Table 5.5). It is expected that the leakage power discrepancy will diminish when low-leakage

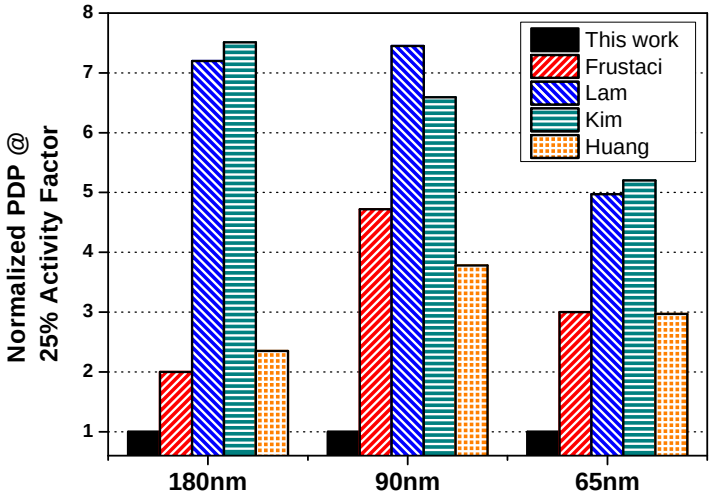


Figure 5.11: Normalized PDP of various comparators at 25% input data activity.

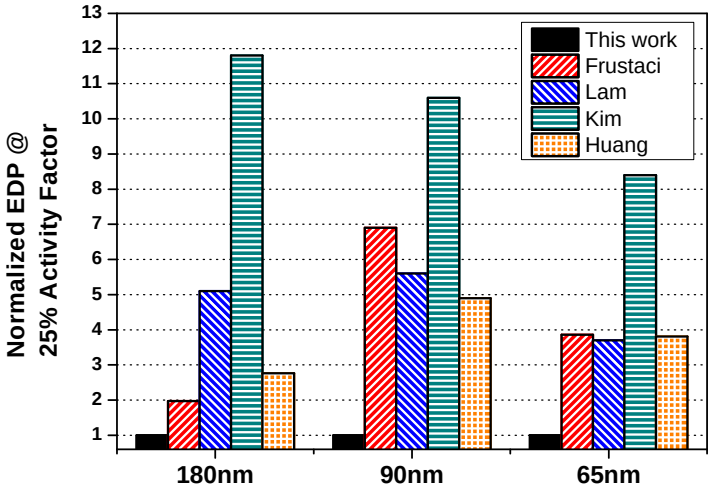


Figure 5.12: Normalized EDP of various comparators at 25% input data activity.

circuit techniques such as power gating are enforced. The total number of transistors and total transistor width of each design in the three CMOS processes are also summarized in Table 5.5. Even though the total number of transistors of the proposed work is the second smallest among all the designs, the proposed comparator's total transistor width

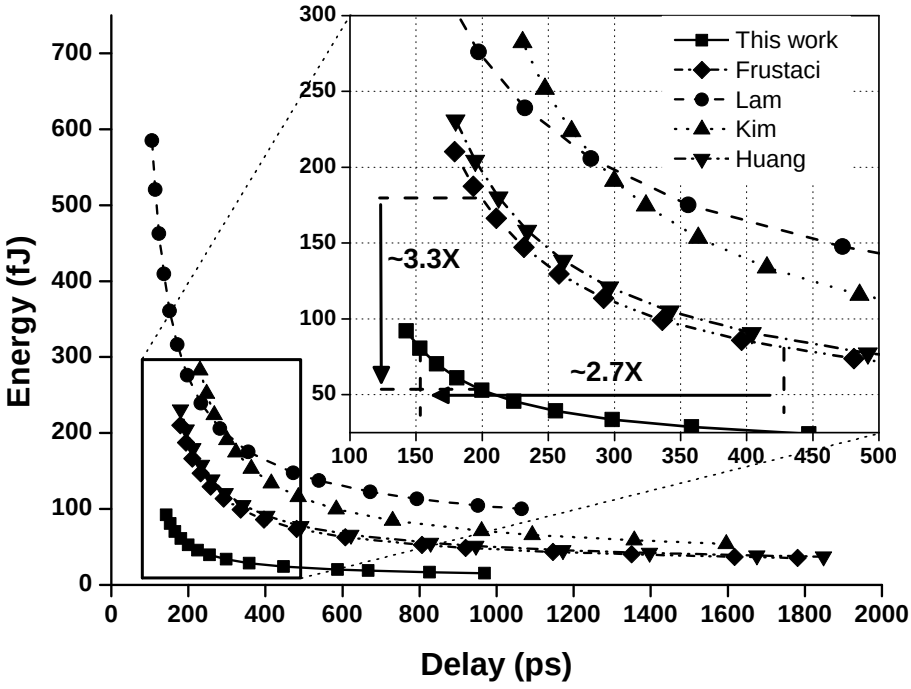


Figure 5.13: Energy-delay tradeoffs of various comparators in a 65-nm CMOS process.

in the 65-nm CMOS process is about 37%, 19%, and 16% larger but 66% smaller than that of [79], [77], [74], and [76], respectively. This is primarily because static logic requires pMOS transistors, which need to be sized larger compared to nMOS transistors due to lower hole mobility, for the critical path logic evaluation.

Fig. 5.13 shows the impact in the energy-delay space of various 64-bit comparators. The proposed comparator is the optimal design in the energy-delay constraint space. At 80fJ iso-energy or 200ps iso-delay budget, the proposed comparator is 2.7× faster or 3.3× more energy efficient than the next best design respectively.

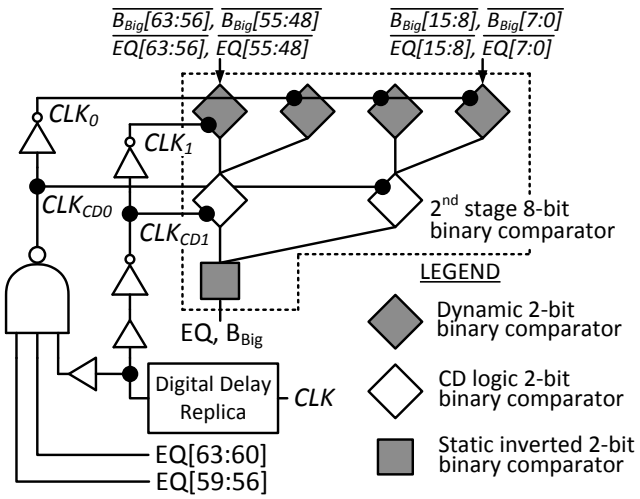


Figure 5.14: Proposed 2^{nd} stage 8-bit binary comparator with dynamic and CD logic.

5.5 Proposed High-Performance 8-bit Comparator with CD Logic

The proposed tree-based comparator can be divided into two stages, where the first stage consists of eight 8-bit comparators in parallel along with input signal buffers and encoding circuitries, and the second stage contains only one 8-bit comparator. Simulation results indicate that the second stage comparator only accounts for 3.1% of the total energy consumption[1], while the input buffers/encoding circuits and the eight 8-bit comparators in the first stage constitute 63.1% and 33.8% of the energy consumption ($\alpha = 12.5\%$), respectively. Based on this observation, we believe the design objective of the second stage comparator should focus on high performance, even at the expense of power consumption. As demonstrated by the silicon results in Section 5.7, such a design philosophy leads to an improved performance with a comparable energy consumption.

[1] The second stage comparator, however, is responsible for 43% of the critical path. The critical path of a 64-bit comparator consists of seven logic levels, of which three of them belong to the second stage comparator.

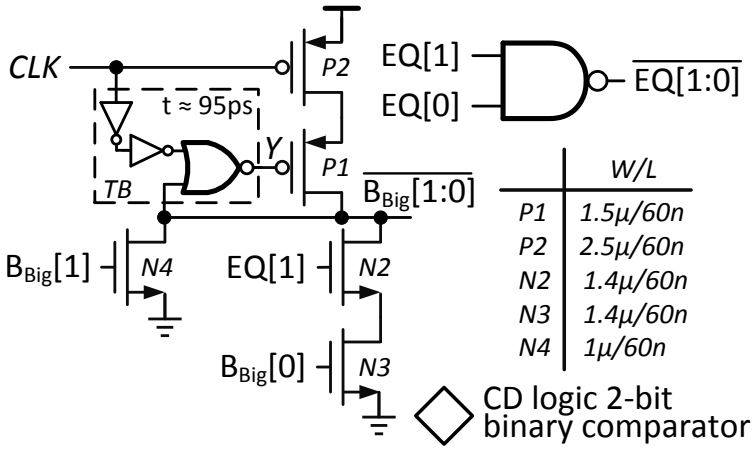


Figure 5.15: Improved CD logic schematic.

The proposed second stage high-speed 8-bit comparator architecture along with the clock generation circuits is shown in Fig. 5.14. The first stage implements a radix-2 merging with footed dynamic logic and results in a maximum stack height of three, as shown in Fig. 5.7(d). A small pMOS *CLK* transistor *P1* precharges the internal node to minimize the charge-sharing problem. CD logic is utilized in the second stage due to its domino-compatibility. Notice that even though only two logic gates are implemented with CD logic, it accounts for 14.3% of the critical path in the proposed 64-bit comparator and also acts as a high-performance logic interface between dynamic and static logic.

The clock tree is arranged such that CD logic always operates in the high-performance D-Q mode. Fig. 5.15 shows the schematic of the CD logic comparison circuitry with transistor sizing. Notice that this improved CD logic reduces the transistor overhead for the *TB* block by 20% over the previously introduced design. Since the clock has an activity factor of 1, this reduction helps to reduce the CD logic's overall power consumption. For a two input NAND gate, the power and area saving is approximately 5% and 2%, respectively. The predischarge nMOS transistor is no longer required, because the first stage dynamic logic always precharge to logic "1" during the precharge period and consequently pulls down the internal node of CD logic to logic "0" through transistor *N4*. The static inverted

comparison circuit acts as a *LB* which reduces the unwanted glitch seen at the output while computing the final stage comparison. *TB* is designed to have a window duration of approximately 95 ps. As shown in Section 5.6, a 95-ps window duration is sufficient for the CD logic to perform robust operations under PVT variations. It is often not a straightforward task to identify the proper window duration especially considering both energy consumption and yield requirement. Designers may have to go through iterative design process between post-layout simulations and layout to obtain this value, which can be very time-consuming. In this work, the following design approach is adapted in order to speedup this process:

1. The high-performance 8-bit comparator is lay out, except for the CD logic's *TB* block. Instead, the window duration of CD logic at this design phase is directly controlled by an external input. A long window period, i.e., 150ps, is used as a starting reference point..
2. Perform post-layout Monte-Carlo and corner simulations. If there is any failure, proceeds to step 4).
3. Reduce the window period by a fixed step size (i.e., 5ps). Go to step 2).
4. Lay out the *TB* block to create the window duration. In this case, the appropriate window duration is (current window period + 5ps).

5.5.1 Design Considerations

Other designs have been explored for the second stage high-performance 8-bit comparator and we have concluded that the dynamic-CD-static logic style combinations is a more suitable design when performance is the primary concern. The 8-bit comparator with only CD logic is not a preferred design due to the following two reasons:

1. There is no performance advantage by replacing the dynamic logic gates in the first stage of the 8-bit comparator with CD logic, since these CD-logic gates always operate in C-Q mode only. As demonstrated in [83], CD logic's C-Q delay is only comparable with that of a dynamic logic gate with a much higher energy consumption.

2. If CD logic is implemented in the third (last) stage of the 8-bit comparator, its output should be protected by a static inverter in case it needs to drive a long interconnect or multiple fan-ins. The additional inverter delay will compromise the delay improvement introduced by the CD logic and defeats the purpose of employing it in the first place.

In addition, compound domino logic style, where dynamic and static logic alternate in every stage, is also not applicable. This is because the last stage dynamic logic also needs to be protected by a static inverter, and same argument in reason (2) can be applied here. One possible solution is to expand the design to become a 16-bit high-performance binary comparator[2]. Simulation results reveal that this design achieves similar delay improvement compared to the proposed high-performance design at a much higher energy consumption. Furthermore, the dynamic logic gates'outputs in the first stage have to traverse through long interconnects, which may be a reliability concern.

5.5.2 Clock Generation Circuits with Clock Gating Capability

The clock generation circuits consist of a digital tunable delay replica and a clock gating circuitry which is controlled by two EQ signals. A digital tunable delay replica [84, 85] is used instead of an analog delay replica to prevent excess static power dissipation, since the power consumption of the delay replica is included in the comparator's power dissipation. The delay replica simulates the critical path delay and ensures that the input signals to the first stage dynamic logic will always arrive earlier than CLK_0 and CLK_1 under all PVT variations. In the proposed 8-bit comparator, one dynamic and one CD logic gate will be triggered every clock cycle by CLK_1 and CLK_{CD1}, respectively. On the other hand, the rest of three dynamic and one CD logic gates will only enter evaluation period if both EQ[63:60] and EQ[59:56] signals are at logic "1". Assuming that the input vectors are random numbers, then the probability that CLK_0 goes to high and CLK_{CD0} goes to low (enter evaluation period) is only $(1/2)^8 = 0.4\%$. Fig. 5.16 illustrates the normalized

[2]In this case, the number of stages becomes four. If compound domino logic style is implemented, then the last stage is a static gate.

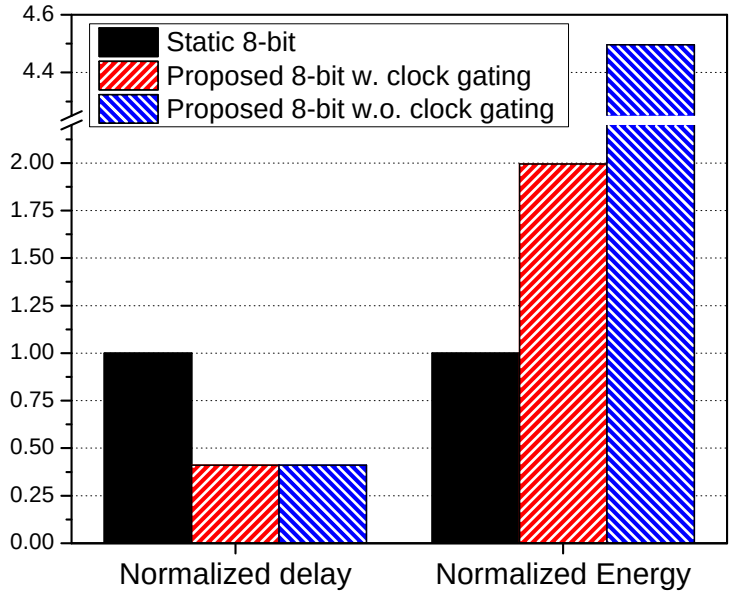


Figure 5.16: Normalized delay and energy consumption ($\alpha = 12.5\%$) of the proposed high-performance comparator with CD logic vs. an 8-bit static comparator.

delay and energy consumption of the proposed 8-bit comparator with and without the clock gating vs. the static 8-bit comparator which is sized for high performance. Clock gating effectively reduces the energy consumption by 2.25× without compromising the performance. Compared to the 8-bit static comparator, the proposed design achieves 60% delay reduction while dissipating 2× the energy. Table 5.3 summarizes the delay distribution of each stage in the proposed 8-bit high-performance comparator for the worst delay input vector. In this configuration, CD logic is approximately 19.5% faster than the dynamic logic.

5.5.3 Digital Tunable Delay Replica

Fig. 5.17 shows the block diagram and schematics of the proposed digital tunable delay replica with detailed transistor sizing. The replica consists of several stages of digitally controlled delay element (DCDE). DCDE has been used extensively in various CMOS

Table 5.3: Proposed 8-bit high-performance comparator delay distribution.

Stage	Logic style	Normalized delay (%)
1	Dynamic logic	33.8
2	CD logic	27.2
3	Static logic	39

applications, including phase-locked loops (PLLs) , time-to-digital converter, and Static Random Access Memory (SRAM) [86–91] to provide precise timing signals. Compared to the conventional current-starved inverter, DCDE is attractive for low-power applications since it reduces the static power dissipation. The proposed DCDE is controlled by a 3-bit binary code. When S[2:0] is "000", transistors *N3*, *N4*, *N6*, and *N7*, are off, transistors *N8*, and *P3* are on, and DCDE has the longest falling edge delay. As the code increments, additional nMOS pull-down branch will be on and increases the pull-down current, which in term reduces DCDE's falling edge delay. An additional nMOS branch consisted of transistors *N5-N7*, which is only on when both code S[0] and S[1] are at logic "1", is necessary to ensure DCDE's monotonic decreasing delay characteristic. When S[2] is at logic "1", the gate capacitance of *P2* is disconnected from the node OUT since the transmission gate *N8* and *P3* are off; hence, DCDE's delay is further reduced. A pMOS gate capacitance is preferred over an additional nMOS pull-down path since in this case, DCDE's layout area can be reduced (better overall area utilization, since the proposed DCDE is already dominated by nMOS transistors). Furthermore, extensive simulation results indicate that an additional nMOS pull-down path provides diminishing delay tunability; hence, to achieve the same delay decrement as that of a pMOS gate capacitance, the nMOS transistors of the pull-down path have to be widened, which contribute to bother larger area and higher power consumption. Notice that this design focuses only on DCDE's falling edge delay, since four stages of DCDE in series ensure that both CLK_{delay}'s falling and rising edge delays are approximately the same. *Buf1* (Fig. 5.17) is also carefully designed (by incorporating post-layout simulation information) to make sure that the delay is matched when S[3] switches. Each DCDE occupies an area of 3.83 μm $\times$ 3.73 μm = 14.3 μm^2 and the

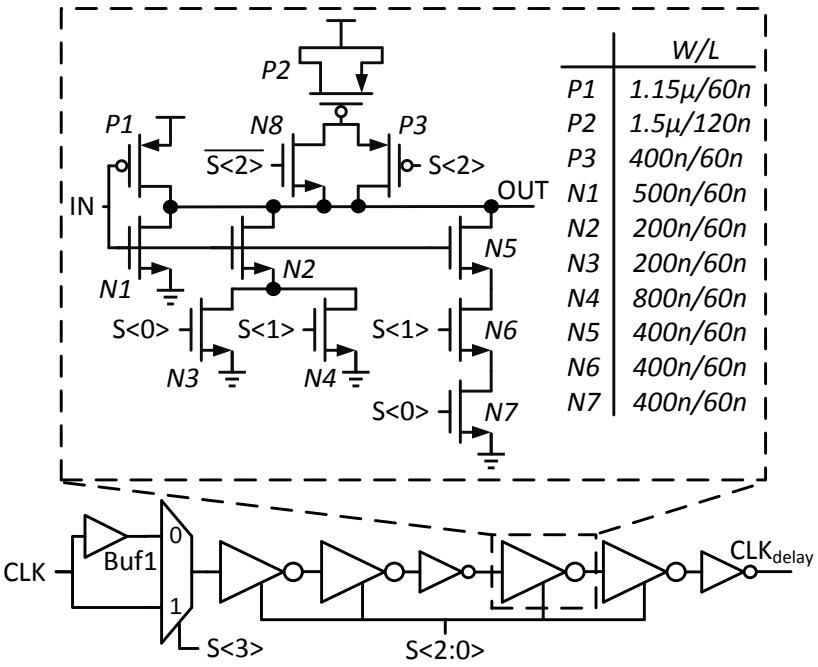


Figure 5.17: Digital tunable delay replica block diagram and schematic.

delay replica occupies an area of 21.5 μm $\times$ 3.73 μm = 80.2 μm^2.

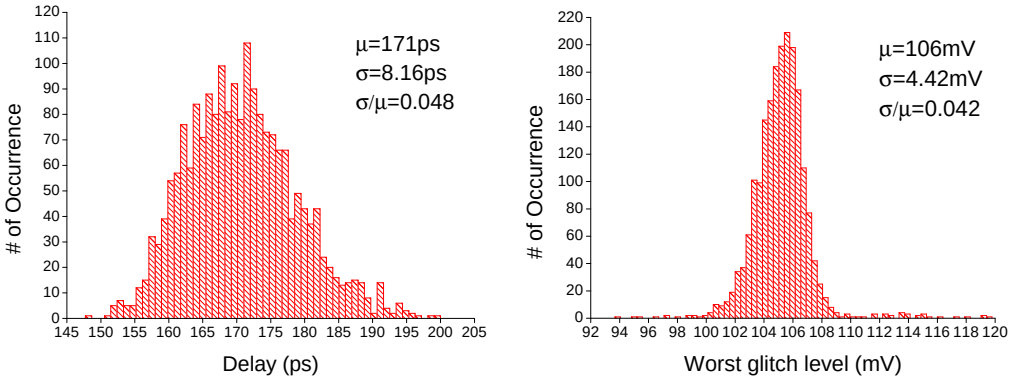
5.6 Robustness Analysis

5.6.1 Process, Voltage, and Temperature (PVT) Variations

Post-layout simulations of the comparator with CD logic at extreme process, temperature, and voltage conditions are performed, with the results summarized in Table 5.4. The worst-case glitch level is determined by first supplying the worst contention input vector (A[63:0]: 000...1 $\rightarrow$ 000...0, B[63:0]: 000...0 $\rightarrow$ 000...1) to the comparator, and then measuring the glitch at node B_{Big}. With the worst contention input vectors, only one branch of nMOS logic transistors (consider Fig. 5.15, only inputs EQ[1] and $\mathrm{B_{Big}}$[0] remain at logic "1") in

Table 5.4: PVT post-layout analysis of the 64-bit comparator with CD logic.

Corner	TT	SF				SS				FS				FF			
Temperature (°C)	27	-40		110		-40		110		-40		110		-40		110	
VDD (V)	1	0.9	1.1	0.9	1.1	0.9	1.1	0.9	1.1	0.9	1.1	0.9	1.1	0.9	1.1	0.9	1.1
Delay (ps)	163	193	135	199	150	247	161	250	187	189	128	191	148	153	112	160	126
Worst glitch (mV)	50	45	56	84	151	45	56	72	90	44	52	74	112	43	60	99	163



(a) (b)

Figure 5.18: (a) Delay and (b) worst-case glitch level of the proposed comparator in a Monte-Carlo post-layout simulation with 2000 iterations at 110°C.

CD logic is on. From Table 5.4, it is evident that the comparator core is functional in the presence of global variations along with excess static power supply noise and temperature fluctuations. The worst case glitch level is 163 mV, which takes place at FF corner, 110°C, and 1.1-V supply. At nominal condition, this comparator's delay with the worst delay input vectors (A[63:0]: 000...0 → 000...1, B[63:0]: 000...1 → 000...0) and worst glitch level are 163 ps and 50 mV, respectively.

5.6.2 Monte-Carlo Post-Layout Simulations

Beside global process variations, Monte-Carlo analysis with local transistor process and mismatch variations (each transistor's key parameters, such as the threshold voltage, will

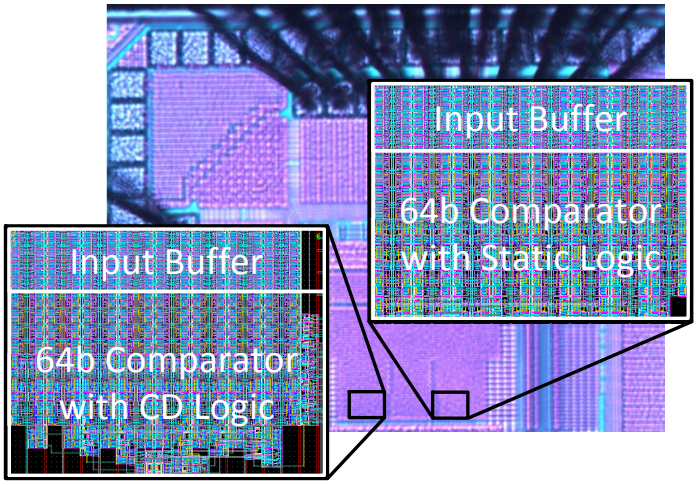


Figure 5.19: Die photo of the two proposed 64-bit comparators. The proposed 64-bit static comparator occupies an area of 1718 μm^2 and the proposed 64-bit comparator with CD logic occupies an area of 2160 μm^2.

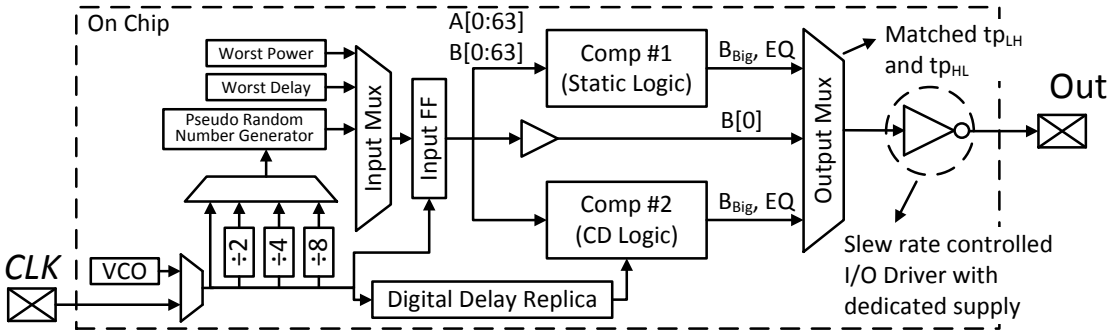


Figure 5.20: 64-bit comparator chip implementation block diagram.

randomly deviate from the default value at TT corner) is also conducted. In this setup, the σ/μ ratio of the nMOS and pMOS transistors' threshold voltage with W/L ratio of 500 nm/60 nm are 0.144 and 0.134, respectively. Fig. 5.18 shows the performance results of Monte-Carlo post-layout simulations of the comparator with CD logic at 110°C with 2000

iterations. With the aforementioned transistor variation profile, the proposed comparator core has an average delay (worst glitch) of 171 ps (106 mV) with a standard deviation of 8.16 ps (4.42 mV). The σ/μ ratios of the delay and worst glitch level are 0.048 and 0.042, respectively.

5.7 Measurement Results

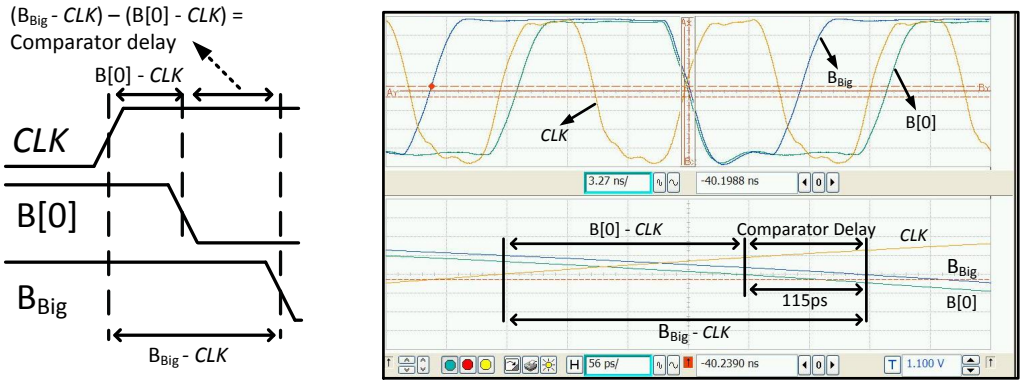
Unless otherwise specified, all the numbers reported in this section were obtained from silicon measurement results based on five packaged dies. The comparator cores were realized in a TSMC 65-nm 1-V 1P9M multi-threshold CMOS process. Each core was organized in 16 rows with a row height of 3.73 μm. The critical path was implemented with low-V_t transistors when necessary to facilitate the comparison generation while non-critical path, delay replica, and clock tree were implemented with standard-V_t transistors to minimize the leakage current. Fig. 5.19 shows the die photo of the two proposed 64-bit binary comparators and Fig. 5.20 shows the chip block diagram.

The delay of the 64-bit comparator with static logic was measured using the following approach:

1. The comparator was triggered by an off-chip 100-MHz reference signal from a Tektronix DG2020A data generator to drive the worst delay input vectors (A[63:0]: 000...0 $\rightarrow$ 000...1, B[63:0]: 000...1 $\rightarrow$ 000...0).

2. The timing difference of both the input vector's falling edge (B[0]) and the critical path, B_{Big}, signal's falling edge to the off-chip reference signal were measured using an Agilent 91304ADSA oscilloscope, and then subtracted from each other to obtain the delay.

Measuring the delay of the 64-bit comparator with CD logic was not as straight-forward. It is important to ensure that the *CLK* signal which feeds the second stage comparator is properly timed, since an early arrival of the signal can lead to a false logic evaluation; on the other hand, a late arrival of the signal leads to a performance degradation. Hence,



(a) (b)

Figure 5.21: (a) Conceptual and (b) actual delay measurement waveforms of the 64-bit comparator with CD logic.

measuring the delay in this case involves two extra steps. The complete procedure is outlined as follows:

1. The comparator was triggered by an off-chip 100-MHz reference signal to drive the following input vector A[63:0]: 000...1 → 000...0, B[63:0]: 000...0 → 000...1. This set of data transition causes $\overline{B_{Big}[7:0]}$ to traverse from logic "1" to "0". In this case, the first stage dynamic logic (controlled by CLK_0) can only enter the evaluation period after $\overline{B_{Big}[7:0]}$ is settled to ensure correct operations (Fig. 5.14).

2. Initially set the digital delay replica's code S[3:0] to "0000" (i.e., longest delay) then incremented the code until the output signal B_{Big} falsely evaluating from logic "1" to "0". (i.e., CLK_0 arrives before $\overline{B_{Big}[7:0]}$ is settled)

3. Once the desired control code was determined, the comparator delay can be measured using the same approach as described for the comparator with static logic.

Both signals *B[0]* and B_{Big} traverse through the same output driver and matched MUX (Fig. 5.20). The slew-rate-controlled output driver has a dedicated power supply that is

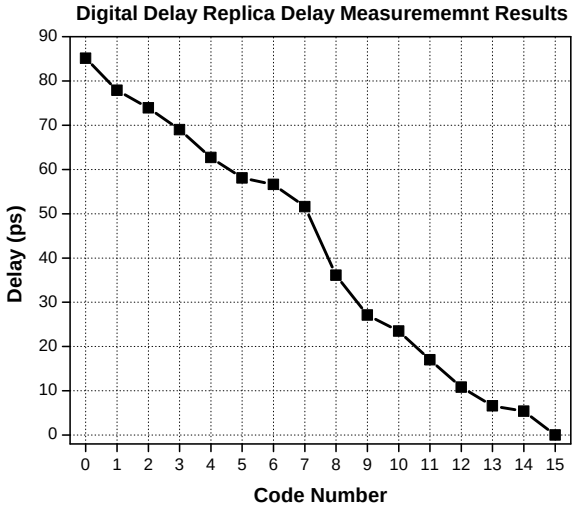


Figure 5.22: Digital delay replica delay measurement results.

isolated from the rest of the chip to minimize supply noise. The conceptual and actual delay measurement waveforms of the proposed comparator with CD logic are shown in Fig. 5.21. It has been demonstrated in [92] that this approach can accurately measure the delay with picosecond resolution. Notice that a buffer delay for the signal *B[0]* before it reaches the output MUX (Fig. 5.20) needs to be included in the calculation of the overall comparator delay. At the nominal 1-V supply, this buffer delay was measured to be 52 ps in a post-layout simulation. Hence, the proposed comparator with CD logic's actual delay was 115 ps (Fig. 5.21) + 52 ps = 167 ps. For the non-timing critical *EQ* signal, the measured worst case delay was approximately 143 ps.

The worst and average power consumptions of the two comparators were measured using the on chip voltage controlled ring oscillator, which was running at approximately 500 MHz. For the worst power consumption, the following input pattern A[63:0]: 000...0 $\rightarrow$ 111...1, B[63:0]: 111...1 $\rightarrow$ 000...0 was supplied. The average power consumption was measured by creating pseudo-random inputs and feeding them to the device under test. The pseudo-random number generator was implemented using a 129-bit (only 128 bits are used) linear feedback shift register (LFSR) with taps on bit locations 124 and 129 [93],

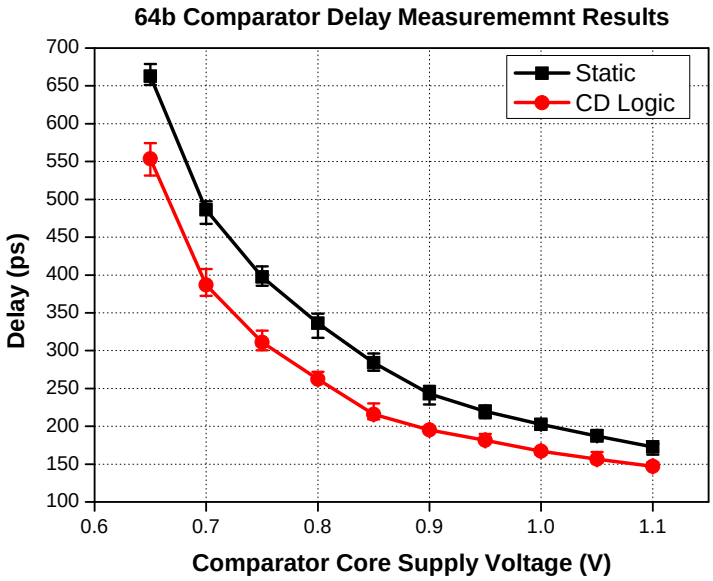


Figure 5.23: Delay vs. comparator core supply voltage measurement results.

and resulted in an average data activity of 0.5, assuming that the flip flop was triggered every clock cycle. Pseudo-random inputs representing different average data activities were created by manipulating the *CLK* frequency of the LFSR (Equation (5.15)).

Fig. 5.22 shows the measured delay vs. controlled code of the digital delay replica. The proposed digital delay replica provided a 85 ps tuning rage, with an average step size of 5.67 ps.

Fig. 5.23 and 5.24 summarize the delay and full-speed power consumption ($\alpha = 12.5\%$) measurement results of the two proposed 64-bit comparators vs. core supply voltage, respectively. The full-speed power consumption is extrapolated by multiplying the measured power consumption at CLK frequency of 500 MHz by a scaling factor, calculated as:

$$\frac{1/comparator\ delay}{500\ MHz} \tag{5.16}$$

which assumes that dynamic power is the dominant component. The power includes the comparator core, input signal buffers, clock trees, and the digital delay replica. At the 1-V nominal supply, the proposed comparator with CD logic runs at a delay of 167 ps for

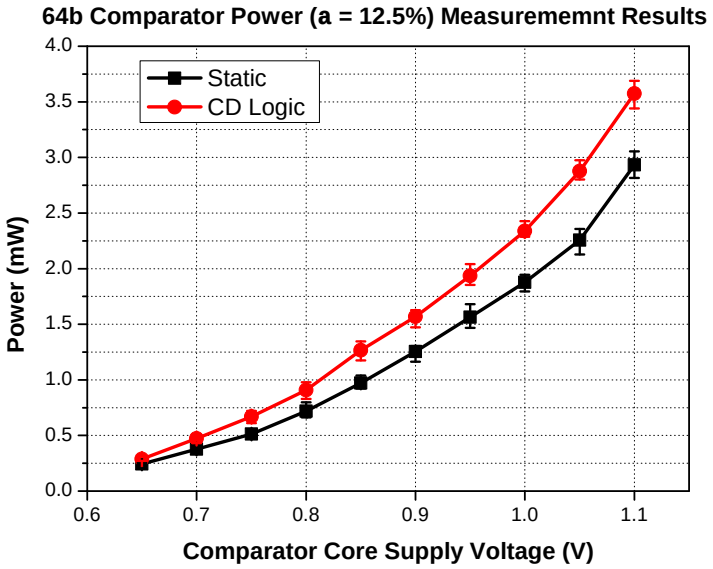


Figure 5.24: Power ($\alpha = 12.5\%$) vs. comparator core supply voltage measurement results.

the worst-case delay input vector with an average power ($\alpha = 12.5\%$) consumption of 2.34 mW, and a leakage of 0.06 mW at room temperature. The estimated standard deviation of the proposed 64-bit comparator with CD logic's delay at nominal supply voltage is 3.3 ps. At the 1.1-V supply, the proposed comparator's delay decreases to 147 ps, with an average power consumption of 3.57 mW, and a leakage power of 0.1 mW. Compared to the original 64-bit comparator, the proposed comparator with CD logic is approximately 18% faster.

The energy consumption of the two proposed comparators are approximately the same across all data activities, as demonstrated in Fig. 5.25. The CD logic based comparator is 2.2% (2.7%) more (less) energy-efficient than the static logic based comparator when the worst power (average power at α=12.5%) input vector is supplied. Between the two proposed 64-bit comparators, the data activity factor influences both designs' energy consumption in a similar fashion, since only one 8-bit comparator (second stage) out of the nine 8-bit comparators (in total) is different. In addition, as discussed in Section 5.5.3, the clock gating circuitry ensures that only one dynamic and one CD logic gate dissipate power every cycle, while the rest of dynamic and CD logic gates only have a 0.4% prob-

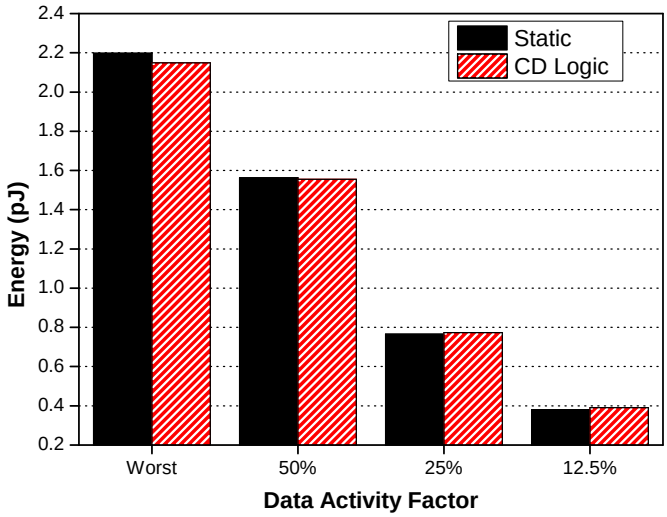


Figure 5.25: Measured energy consumption of the two comparators at various data activity factors at 1-V supply.

ability to dissipate dynamic power. Fig. 5.26 shows the energy-delay product (EDP) of the two proposed 64-bit comparators at various α. The proposed comparator with CD logic achieves lower EDP across all data activity factors. At $\alpha = 12.5\%$, the EDP of the comparator with CD logic is approximately 15% lower than that of the comparator with static logic.

Fig. 5.27 shows the measured energy ($\alpha = 12.5\%$) vs. delay curve of the two proposed comparators. Each point on the two curves is generated by multiplying the measured power consumption and delay of the two 64-bit comparators at a specific supply voltage. CD logic is the more attractive design in all of the energy-delay constraint space. At 0.3 pJ iso-energy or 250 ps iso-delay budget, CD logic is 20% faster or 17% more energy-efficient than the static logic, respectively. As shown previously, both comparators achieve similar energy consumption at nominal supply voltage, while the CD logic based comparator has a shorter delay. In other words, for iso-delay comparison, CD logic based comparator becomes more energy-efficient than the static logic based comparator. For different α, the

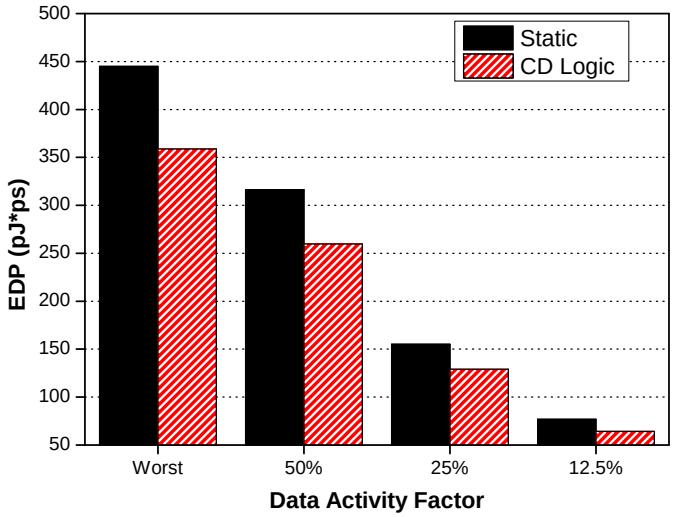


Figure 5.26: Measured energy-delay product (EDP) of the two comparators at various data activity factors at 1-V supply.

proposed comparator with CD logic also exhibits similar delay improvement or energy-efficiency at iso-energy or iso-delay constraint, respectively. Finally, Table 5.5 compares this work with state-of-the-art designs. The numbers in brackets are normalized to a 65-nm CMOS process, with the conversion formulas shown below the table. The normalization formulas work well for technology nodes that are within few generations (i.e., 180 nm, and 90 nm) and may not be suitable for older processes such as 350 nm and 600 nm. Also, all the other designs do not specify the data activity factor, which as demonstrated in this thesis plays a significant role in determining the energy consumption. [94] and [78] achieve better numbers than the proposed work; however, they are not based on silicon results. The proposed 64-bit comparator with CD logic at $\alpha = 25\%$ is 42% and 76% more energy and EDP efficient, respectively, than the recently published silicon results [77]. Notice that the measurement results are consistent with the simulation numbers. As shown in Fig. 5.16, the energy consumption of the proposed 8-bit high-performance comparator is 2× more than that of the 8-bit static comparator, which accounts for 3.1% of a 64-bit

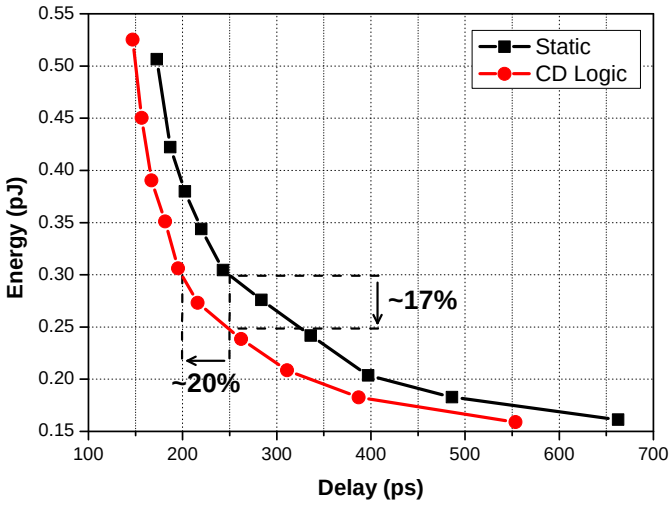


Figure 5.27: Measured energy (α=12.5%) vs. delay curve of the two proposed comparators under various supply voltages.

comparator's energy consumption at α = 12.5%. Hence, based on simulation results the 64-bit comparator with CD logic's energy consumption will be 3.1% higher than its static counterpart. As summarized in Table 5.5, the measured energy consumption of the 64-bit comparator with CD logic is 2.63% higher than that of the static 64-bit comparator at α = 12.5%.

5.8 Conclusion

A new single-cycle, radix-2 tree based comparator is developed. This design is static logic compatible, and has demonstrated low-power, high-performance operation compared to state-of-the-art works. Detailed analysis across three technology nodes reveal that it is the most power-efficient design with approximately 2.3× and 3.7× reduction in terms of power and EDP at 65nm respectively. Furthermore, this design is attractive with both low-delay and low-energy-constraint. When implemented with a 80 fJ energy budget, the proposed comparator achieves 2.7× better performance. With a 200 ps delay constraint, the proposed work is 3.3× more energy efficient than the next best design. A 64-bit single-

Table 5.5: 64-bit comparator chip performance comparisons.

	This work[1] (CD Logic)	**This work[1] (Static)**	[94]	[78]	[77]	[76]
Process (nm)	**65**	**65**	90	90	180	350
VDD (V)	**1**	**1**	1	1	1.8	*N/A*
Silicon Results	***Yes***	***Yes***	*No*	*No*	*Yes*	*Yes*
Delay(ps)	**167**	**203**	220 (159)	230 (166)	1120 (404)	1550 (288)
Energy (pJ)	**2.15 (Worst)** **1.55 (α = 50%)** **0.77 (α = 25%)** **0.39 (α = 12.5%)**	**2.2 (Worst)** **1.56 (α = 50%)** **0.77 (α = 25%)** **0.38 (α = 12.5%)**	0.77 (0.56)[2]	1 (0.72)[2]	12.65 (1.32)[2]	*N/A*
EDP (pJ × ps)	**129 (α = 25%)**	**156 (α = 25%)**	169 (89)	230 (120)	14168 (533)	*N/A*
Area (μm^2)	**2160**	**1718**	*N/A*	*N/A*	4416 (576)	199576 (6883)
Leakage (mW)	**0.06**	**0.06**	0.007	*N/A*	*N/A*	*N/A*

[1] Measurement results based on five packaged dies.

[2] data activity factor (α) not specified.

[3] Numbers in brackets are normalized to a 65-nm CMOS process.

[4] Normalization formulas are: $t_{d_{norm}} = t_d \times \left(\frac{65nm}{tech.}\right)$, $E_{norm} = E \times \left(\frac{65nm}{tech.}\right)\left(\frac{1V}{V_{tech.}}\right)^2$, $Area_{norm} = Area \times \left(\frac{65nm}{tech.}\right)^2$

cycle, tree-based comparator with CD logic is also described. The proposed comparator with CD logic implemented exclusively in the timing-critical path achieves additional speed advantage with comparable energy consumption over the same design with static logic only and its robustness has been verified in extensive post-layout simulations under various process, voltage, and temperature conditions. The proposed comparator is fabricated in a TSMC 1-V 65-nm 1P9M multi-threshold CMOS process. At the nominal 1-V supply, the proposed comparator with CD logic show a delay of 167 ps, an average power consumption of 2.34 mW when the inputs toggle at a data activity factor of 12.5%, and a leakage power of 0.06 mW. At an iso-energy or iso-delay constraint of 0.3 pJ and 250 ps, the proposed comparator with CD logic is 20% faster or 17% more energy-efficient than the proposed comparator with static logic only, respectively.

Chapter 6

Conclusions

Arithmetic circuits such as adders and comparators are crucial components in today's microprocessors and are also used extensively in many areas of CMOS digital systems. For high-performance central processing units, arithmetic circuits in the arithmetic logic unit are often the performance bottleneck. This thesis analyzes arithmetic circuits and provides both transistor-level and micro-architecture level techniques to improve arithmetic circuit's performance and energy-efficiency. Extensive simulation results and silicon data from two test chips have verified the effectiveness and robustness of the proposed techniques.

6.1 Summary of Contributions

The main contribution of this thesis is to provide high-performance, energy-efficient CMOS arithmetic circuit designs at both micro-architecture and circuit levels. Details of the contributions are summarized below:

Constant-Delay Logic

CD logic, where the critical path, to a first-order approximation, does not depend on the logic expression. The performance advantage of CD logic has been demonstrated over other

logic styles in various applications in both simulations and silicon results presented in this thesis. By implementing CD logic exclusively in the critical path while adapting static logic for non-critical paths, high-performance, energy-efficient operations can be achieved. This design strategy is especially suitable for applications where there exists a unique critical path, such as adders and comparators.

Detailed theoretical and simulations on the design considerations of CD logic are formulated and presented. Design choices including maximum glitch requirement, and window width are discussed and analyzed. Designers can follow the procedures outlined in this thesis to determine the optimal design parameters based on the yield requirement and the degree of PVT variations.

64-bit Ling Adder with CD logic

A 64-bit hybrid-radix, sparse-4 adder architecture using Ling's algorithm is presented in this thesis. This adder architecture utilizes CD logic in the critical path with a modified sparse-4 sum precomputation circuitry to achieve high-performance addition. Silicon results in a 65-nm CMOS technology indicate that this adder with CD logic achieves shortest delay with lower power consumption, leading to a better energy-efficiency. This adder is particular suitable for server processors where performance is the primary concern.

64-bit Binary Tree-Structure Comparator with CD logic

A 64-bit binary comparator with a radix-2 tree structure is presented in this thesis. This comparator design targets at deep-submicron CMOS process by limiting the maximum number of stack height to be two and can be implemented with static logic only to achieve low-power operation, especially in low data activity environments. If higher performance is desired, the second stage 8-bit comparator can be implemented with CD logic based on the observation that the second stage 8-bit comparator accounts for 43% of the critical path while only contributing to a small fraction of the total power consumption. Silicon results in a 65-nm CMOS process indicate that the 64-bit comparator with CD logic is

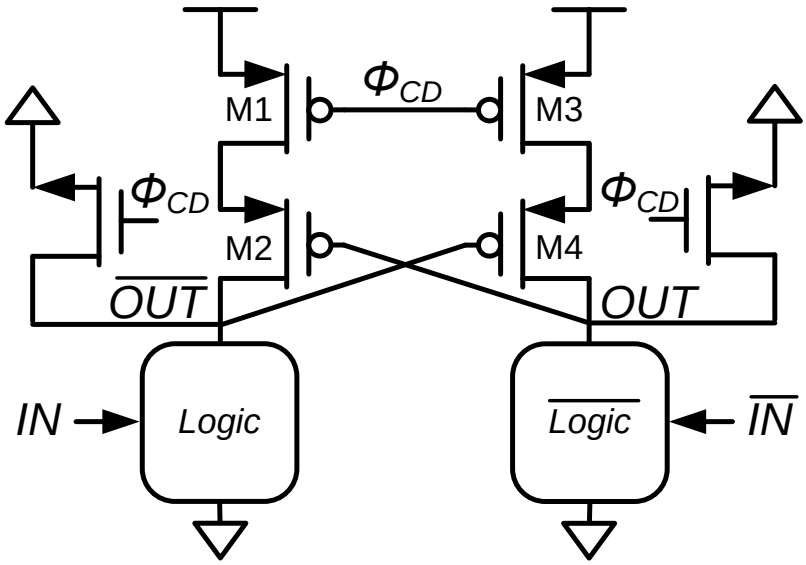


Figure 6.1: Differential constant delay (CD) logic block diagram.

20% faster or 17% more energy-efficient compared to a 64-bit comparator with static logic only at 0.3 pJ iso-energy or 250 ps iso-delay budget, respectively.

6.2 Future Work

This thesis introduces a new CMOS logic style, constant-delay (CD) logic, and how it can be utilized in applications with a unique critical path, such as adders and comparators. Full-custom designs in a 65-nm CMOS technology has demonstrated CD logic's performance potentials. Looking forward, several research directions can be investigated to further demonstrate CD logic's performance advantage and to encourage other circuit designers in adapting this logic style.

The single-ended CD logic demonstrated in this thesis utilizes a timing window to reduce the power consumption. The timing window needs to be carefully analyzed under process, voltage, and temperature variations to ensure high-performance operations with excellent yield. If robustness is the primary concern, differential CD logic can be imple-

mented at a cost of additional power consumption. A schematic of a differential CD logic is shown in Fig. 6.1.

Assuming the inputs to the NMOS logic block are coming from a precharge-type logic (i.e., dynamic logic), both OUT and $\overline{OUT}$ are initially at logic "0" when ϕ_{CD} goes from high to low, since all the inputs entering the differential CD logic are at logic "1". When ϕ_{CD} is low, both OUT and $\overline{OUT}$ enter the contention period and rise to a non-zero voltage level due to direct path current. When the preceding logic gates make the transitions and turn off either $Logic$ or $\overline{Logic}$ block (assuming $Logic$ block is turned off in this case), $\overline{OUT}$ rises to logic "1" and turns off transistor $M4$. In this case, OUT goes back to logic "0" and the direct path current is eliminated. Compared to a single-ended CD logic, both the area and power consumption of differential CD logic are increased due to a greater number of transistors and the number of direct current path. On the other hand, a timing block is no longer required for the differential CD logic, as the differential nature provides a self-timing window that is robust under process, temperature, and voltage variation. For deep sub-micron CMOS technologies (i.e., 28 nm and below) with high-performance and stringent yield requirements, differential CD logic may be an attractive design choice over single-ended CD logic and other logic families. Additional analysis in terms of performance, power consumption, and robustness compared to other logic families should be conducted to further explore the potential of differential CD logic.

Another important research direction is to integrate CD logic with CAD tools such that circuit designers working in a digital standard design flow environment can utilize CD logic in performance-critical circuit blocks without sacrificing the overall development cycle time. This will require (1) implementation of various logic gates using CD logic, (2) careful characterization of these logic gates at different corner and (3) circuit blocks with timing and power information in standard-cell library compatible format.

In addition, to ensure high yield for robust designs, careful characterization of CD logic's glitch and window width requirement under different process variation in advanced CMOS processes such as 28nm and below will be necessary. Finally, integrating CD logic into an even more complex circuit block, i.e., arithmetic logic unit (ALU), is an important research direction that should be taken into consideration. In particular, the following work will be required:

1. Investigate the 64-bit ALU design, especially at low supply voltages for energy efficiency.
2. Design of non-critical, low-power logic and shifting units
3. Design of the ALU peripheral circuits
4. Low-leakage standby mode and efficient clock gating for the adder unit

References

[1] "IC Knowledge," *IC Knowledge LLC, http://www.icknowledge.com/trends/trends.html*.

[2] M. Horowitz, E. Alon, D. Patil, S. Naffziger, R. Kumar, and K. Bernstein, "Scaling, power, and the future of CMOS," *IEEE International Electron Devices Meeting*, pp. 7–15, Dec. 2005.

[3] R. Ho, K. Mai, and M. Horowitz, "The future of wires," *Proceedings of the IEEE*, vol. 89, pp. 490–504, Apr 2001.

[4] R. Ho, K. Mai, and M. Horowitz, "Managing wire scaling: a circuit perspective," in *IEEE 2003 International Interconnect Technology Conference*, pp. 177–179, June 2003.

[5] H. Sutter, "The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software," *http://www.gotw.ca/publications/concurrency-ddj.htm*.

[6] E. Fetzer, M. Gibson, A. Klein, N. Calick, C. Zhu, E. Busta, and B. Mohammad, "A fully bypassed six-issue integer datapath and register file on the Itanium-2 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1433–1440, Nov 2002.

[7] S. Naffziger, B. Stackhouse, T. Grutkowski, D. Josephson, J. Desai, E. Alon, and M. Horowitz, "The implementation of a 2-core, multi-threaded itanium family processor," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 197–209, Jan 2006.

[8] S. Rusu, S. Tam, H. Muljono, D. Ayers, and J. Chang, "A Dual-Core Multi-Threaded Xeon Processor with 16MB L3 Cache," in *IEEE International Solid-State Circuits Conference*, pp. 315–324, Feb 2006.

[9] M. Golden, S. Arekapudi, G. Dabney, M. Haertel, S. Hale, L. Herlinger, Y. Kim, K. McGrath, V. Palisetti, and M. Singh, "A 2.6ghz dual-core 64bx86 microprocessor with ddr2 memory support," in *IEEE International Solid-State Circuits Conference*, pp. 325–332, Feb 2006.

[10] N. Verma and A. Chandrakasan, "A 65nm 8T Sub-Vt SRAM Employing Sense-Amplifier Redundancy," in *IEEE International Solid-State Circuits Conference*, pp. 328 –606, 11-15 2007.

[11] M. Sinangil, N. Verma, and A. Chandrakasan, "A Reconfigurable 8T Ultra-Dynamic Voltage Scalable (U-DVS) SRAM in 65 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 44, pp. 3163 –3173, nov. 2009.

[12] R. Zimmermann and W. Fichtner, "Low-power logic styles: CMOS versus pass-transistor logic," *IEEE Journal of Solid-State Circuits*, vol. 32, pp. 1079 –1090, July 1997.

[13] R. Krambeck, C. Lee, and H.-F. Law, "High-speed compact circuits with CMOS," *IEEE Journal of Solid-State Circuits*, vol. 17, pp. 614 – 619, June 1982.

[14] N. Goncalves and H. De Man, "NORA: a racefree dynamic CMOS technique for pipelined logic structures," *IEEE Journal of Solid-State Circuits*, vol. 18, pp. 261 – 266, June 1983.

[15] V. Friedman and S. Liu, "Dynamic logic CMOS circuits," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 263 – 266, April 1984.

[16] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective (4th Edition)*. Addison Wesley, 4 ed., March 2010.

[17] C. Lee and E. Szeto, "Zipper CMOS," *IEEE Circuits and Systems Magazine*, pp. 10 – 16, May 1986.

[18] K. Bernstein, K. Carrig, C. M. Durham, P. R. Hansen, D. Hogenmiller, E. J. Nowak, and N. J. Rohrer, *High Speed CMOS Design Styles*. Springer, 1st ed., Aug. 1998.

[19] R. Zlatanovici, S. Kao, and B. Nikolic, "Energy–Delay Optimization of 64-Bit Carry-Lookahead Adders With a 240 ps 90 nm CMOS Design Example," *IEEE Journal of Solid-State Circuits*, vol. 44, pp. 569–583, Feb. 2009.

[20] S. Mathew, R. Krishnamurthy, M. Anders, R. Rios, K. Mistry, and K. Soumyanath, "Sub-500-ps 64-b ALUs in 0.18μm SOI/bulk CMOS: design and scaling trends," *IEEE Journal of Solid-State Circuits*, vol. 36, pp. 1636–1646, Nov. 2001.

[21] S. Mathew, M. Anders, R. Krishnamurthy, and S. Borkar, "A 4 GHz 130 nm address generation unit with 32-bit sparse-tree adder core," *IEEE Symposium on VLSI Circuits*, pp. 126–127, 2002.

[22] S. Mathew, M. Anders, B. Bloechel, T. Nguyen, R. Krishnamurthy, and S. Borkar, "A 4-GHz 300-mW 64-bit integer execution ALU with dual supply voltages in 90-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 44–51, Jan. 2005.

[23] S. Wijeratne, N. Siddaiah, S. Mathew, M. Anders, R. Krishnamurthy, J. Anderson, S. Hwang, M. Ernest, and M. Nardin, "A 9GHz 65nm Intel Pentium 4 Processor Integer Execution Core," *IEEE International Solid-State Circuits Conference*, pp. 353–365, 2006.

[24] I. Sutherland, R. F. Sproull, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann, 1st ed., Feb. 1999.

[25] V. G. Oklobdzija, B. R. Zeydel, H. Dao, S. Mathew, and K. Ram, "Energy-delay estimation technique for high-performance microprocessor VLSI adders," *IEEE Symposium on Computer Arithmetic*, pp. 272–279–, 2003.

[26] S. Kiaei, S.-H. Chee, and D. Allstot, "CMOS source-coupled logic for mixed-mode VLSI," *IEEE International Symposium on Circuits and Systems*, pp. 1608 –1611 vol.2, May 1990.

[27] F. Lu and H. Samueli, "A bit-level pipelined implementation of a CMOS multiplier-accumulator using a new pipelined full-adder cell design," in *International Phoenix Conference on Computers and Communications*, pp. 49–53, March 1989.

[28] F. Lu and H. Samueli, "A high-speed CMOS full-adder cell using a new circuit design technique-adaptively-biased pseudo-NMOS logic," in *IEEE International Symposium on Circuits and Systems*, pp. 562–565 vol.1, May 1990.

[29] L. McMurchie, S. Kio, G. Yee, T. Thorp, and C. Sechen, "Output prediction logic: a high-performance CMOS design technique ," *International Conference on Computer Design*, pp. 247 –254, 2000.

[30] S. Sun, L. McMurchie, and C. Sechen, "A high-performance 64-bit adder implemented in output prediction logic," in *Conference on Advanced Research in VLSI*, pp. 213–222, 2001.

[31] K. Chong, L. McMurchie, and C. Sechen, "A 64b adder using self-calibrating differential output prediction logic," in *IEEE International Solid-State Circuits Conference*, pp. 1745–1754, Feb 2006.

[32] V. Navarro-Botello, J. A. Montiel-Nelson, and S. Nooshabadi, "Low Power Arithmetic Circuit in Feedthrough Dyanmic CMOS Logic," *IEEE International Midwest Symposium on Circuits and Systems*, vol. 1, pp. 709–712, 2006.

[33] V. Navarro-Botello, J. A. Montiel-Nelson, and S. Nooshabadi, "Analysis of High-Performance Fast Feedthrough Logic Families in CMOS," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, no. 6, pp. 489–493, 2007.

[34] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital integrated circuits : a design perspective*, vol. 2nd. Upper Saddle River, N.J.: Pearson Education, 2003.

[35] Y. Taur and T. Ning, *Fundamentals of Modern VLSI Devices*. Cambridge University Press, 1998.

[36] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer cmos circuits," *Proceedings of the IEEE*, vol. 91, pp. 305 – 327, Feb. 2003.

[37] N. Kurd, J. Barkarullah, R. Dizon, T. Fletcher, and P. Madland, "A multigigahertz clocking scheme for the Pentium(R) 4 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 36, pp. 1647 –1653, Nov. 2001.

[38] S. Vangal, N. Borkar, E. Seligman, V. Govindarajulu, V. Erraguntla, H. Wilson, A. Pangal, V. Veeramachaneni, M. Anders, J. Tschanz, *et al.*, "5GHz 32b integer-execution core in 130nm dual-VT CMOS," *IEEE International Solid-State Circuits Conference*, pp. 334–335, 2002.

[39] P. Chuang, D. Li, and M. Sachdev, "Design of a 64-bit low-energy high-performance adder using dynamic feedthrough logic," *IEEE International Symposium on Circuits and Systems*, pp. 3038–3041, 2009.

[40] M. Aguirre-Hernandez and M. Linares-Aranda, "CMOS Full-Adders for Energy-Efficient Arithmetic Applications," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 19, pp. 718–721, April 2010.

[41] B. Zeydel, D. Baran, and V. Oklobdzija, "Energy-Efficient Design Methodologies: High-Performance VLSI Adders," *IEEE Journal of Solid-State Circuits*, vol. 45, pp. 1220–1233, June 2010.

[42] Z. Huang and M. Ercegovac, "Effect of wire delay on the design of prefix adders in deep-submicron technology," in *Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1713–1717 vol.2, Oct 2000.

[43] A. Weinberger and J. Smith, "A one-microsecond adder using one-megacycle circuitry," *IRE Transactions on Electronic Computers*, vol. EC-5, pp. 65–73, June 1956.

[44] P. M. Kogge and H. S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations," *IEEE Transactions on Computers*, vol. C-22, pp. 786–793, Aug 1973.

[45] R. P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders," *IEEE Transactions on Computers*, vol. C-31, pp. 260–264, March 1982.

[46] T. Han and D. Carlson, "Fast area-efficient VLSI adders," in *IEEE Symposium on Computer Arithmetic (ARITH)*, pp. 49–56, May 1987.

[47] R. E. Ladner and M. J. Fischer, "Parallel prefix computation," *Journal of the ACM (JACM)*, vol. 27, no. 4, pp. 831–838, 1980.

[48] S. Knowles, "A family of adders," in *IEEE Symposium on Computer Arithmetic*, pp. 30–34, 1999.

[49] S. Winograd, "On the Time Required to Perform Addition," *Journal of the ACM (JACM)*, vol. 12, pp. 277–285, Apr. 1965.

[50] R. Doran, "Variants of an improved carry look-ahead adder," *IEEE Transactions on Computers*, vol. 37, no. 9, pp. 1110–1113, 1988.

[51] M. Lehman and N. Burla, "Skip techniques for high-speed carry-propagation in binary arithmetic units," *IRE Transactions on Electronic Computers*, no. 4, pp. 691–698, 1961.

[52] J. Sklansky, "Conditional-Sum Addition Logic," *IRE Transactions on Electronic Computers*, vol. EC-9, pp. 226–231, June 1960.

[53] V. Oklobdzija and E. R. Barnes, "Some optimal schemes for ALU implementation in VLSI technology," in *IEEE Symposium on Computer Arithmetic (ARITH)*, pp. 2–8, June 1985.

[54] H. Dao, B. Zeydel, and V. Oklobdzija, "Energy optimization of pipelined digital systems using circuit sizing and supply scaling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, pp. 122–134, Feb 2006.

[55] V. Oklobdzija, B. Zeydel, H. Dao, S. Mathew, and R. Krishnamurthy, "Comparison of high-performance VLSI adders in the energy-delay space," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, pp. 754–758, June 2005.

[56] B. Zeydel, T. Kluter, and V. Oklobdzija, "Efficient mapping of addition recurrence algorithms in CMOS," in *IEEE Symposium on Computer Arithmetic*, pp. 107–113, June 2005.

[57] H. Ling, "High-speed binary adder," *IBM Journal of Research and Development*, vol. 25, no. 3, pp. 156–166, 1981.

[58] G. Dimitrakopoulos and D. Nikolos, "High-speed parallel-prefix vlsi ling adders," *IEEE Transactions on Computers*, vol. 54, pp. 225–231, Feb 2005.

[59] J. Park, H. Ngo, J. Silberman, and S. Dhong, "470 ps 64-bit parallel binary adder [for CPU chip]," *IEEE Symposium on VLSI Circuits*, pp. 192–193, 2000.

[60] S. Pant, D. Blaauw, V. Zolotov, S. Sundareswaran, and R. Panda, "Vectorless analysis of supply noise induced delay variation," in *International Conference on Computer Aided Design*, pp. 184–191, Nov 2003.

[61] A. Strak and H. Tenhunen, "Investigation of Timing Jitter in NAND and NOR Gates Induced by Power-Supply Noise," in *IEEE International Conference on Electronics, Circuits and Systems*, pp. 1160–1163, Dec 2006.

[62] T. Hook, M. Breitwisch, J. Brown, P. Cottrell, D. Hoyniak, C. Lam, and R. Mann, "Noise margin and leakage in ultra-low leakage SRAM cell design," *IEEE Transactions on Electron Devices*, vol. 49, pp. 1499–1501, Aug 2002.

[63] J. Rius and M. Meijer, "A high-frequency nonquasi-static analytical model including gate leakage effects for on-chip decoupling capacitors," *IEEE Transactions on Advanced Packaging*, vol. 29, pp. 88–97, Feb 2006.

[64] M. Gowan, L. Biro, and D. Jackson, "Power considerations in the design of the Alpha 21264 microprocessor," in *IEEE Design Automation Conference*, pp. 726–731, June 1998.

[65] H. Chen, J. Neely, M. Wang, and G. Co, "On-chip decoupling capacitor optimization for noise and leakage reduction," in *Symposium on Integrated Circuits and Systems Design*, pp. 251–255, Sept 2003.

[66] J. Gu, R. Harjani, and C. Kim, "Design and Implementation of Active Decoupling Capacitor Circuits for Power Supply Regulation in Digital ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, pp. 292–301, Feb 2009.

[67] P. Larsson, "Resonance and damping in CMOS circuits with on-chip decoupling capacitance," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications,*, vol. 45, pp. 849–858, Aug 1998.

[68] M. Xiongfei, R. Saleh, and K. Arabi, "Layout of Decoupling Capacitors in IP Blocks for 90-nm CMOS," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, pp. 1581–1588, Nov 2008.

[69] S. Mukhopadhyay, C. Neau, R. Cakici, A. Agarwal, C. Kim, and K. Roy, "Gate leakage reduction for scaled devices using transistor stacking," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, pp. 716–730, Aug 2003.

[70] Y. Chen, H. Li, K. Roy, and C.-K. Koh, "Gated Decap: Gate Leakage Control of On-Chip Decoupling Capacitors in Scaled Technologies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, pp. 1749–1752, Dec 2009.

[71] W. Hwang, G. Gristede, P. Sanda, S. Wang, and D. Heidel, "Implementation of a self-resetting CMOS 64-bit parallel adder with enhanced testability," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 1108 –1117, Aug 1999.

[72] M. Mirhassani, M. Ahmadi, and G. Jullien, "Low-Power Mixed-Signal CVNS-Based 64-Bit Adder for Media Signal Processing," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 16, pp. 1141 –1150, Sept. 2008.

[73] C.-C. Wang, C.-F. Wu, and K.-C. Tsai, "1 GHz 64-Bit High-speed Comparator using ANT Dynamic Logic with two-phase Clocking," *IEE Proceedings on Computers and Digital Techniques*, vol. 145, pp. 433–436, Nov. 1998.

[74] C.-H. Huang and J.-S. Wang, "High-Performance and Power-Efficient CMOS Comparators," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 254 – 262, Feb. 2003.

[75] H.-M. Lam and C.-Y. Tsui, "High-performance Single clock Cycle CMOS Comparator," *Electronics Letters*, vol. 42, no. 2, pp. 75–77, 2006.

[76] H.-M. Lam and C.-Y. Tsui, "A Mux-based High-Performance Single-Cycle CMOS Comparator," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, pp. 591 –595, July 2007.

[77] J.-Y. Kim and H.-J. Yoo, "Bitwise Competition Logic for Compact Digital Comparator," *IEEE Asian Solid-State Circuits Conference*, pp. 59 –62, Nov. 2007.

[78] S. Perri and P. Corsonello, "Fast Low-Cost Implementation of Single-Clock-Cycle Binary Comparator," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, pp. 1239 –1243, Dec. 2008.

[79] F. Frustaci, S. Perri, M. Lanuzza, and P. Corsonello, "A New Low-power High-speed Single-clock-cycle Binary Comparator," *IEEE International Symposium on Circuits and Systems*, 2010.

[80] J. D. Warnock, J. M. Keaty, J. Petrovick, J. G. Clabes, C. J. Kircher, B. L. Krauter, P. J. Restle, B. A. Zoric, and C. J. Anderson, "The circuit and physical design of the POWER4 microprocessor," *IBM Journal of Research and Development*, vol. 46, no. 1, pp. 27–51, 2002.

[81] K. Furuya, "Design methodologies of comparators based on parallel hardware algorithms," in *International Symposium on Communications and Information Technologies*, pp. 591 –596, Oct. 2010.

[82] D. Harris, "A Taxonomy of Parallel Prefix Networks," *Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 2213 – 2217 Vol.2, 2003.

[83] P. Chuang, D. Li, and M. Sachdev, "Constant delay logic style," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 554–565, March 2013.

[84] M. Dooghabadi, H. Hjortland, and T. Lande, "High precision calibrated digital delay element," *Electronics Letters*, vol. 47, pp. 564–565, April 2011.

[85] C. W. Farrow, "A continuously variable digital delay element," in *IEEE International Symposium on Circuits and Systems*, pp. 2641–2645 vol.3, Jun 1988.

[86] M. Maymandi-Nejad and M. Sachdev, "A monotonic digitally controlled delay element," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 2212 – 2219, nov. 2005.

[87] D. Sheng, C.-C. Chung, and C.-Y. Lee, "An Ultra-Low-Power and Portable Digitally Controlled Oscillator for SoC Applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, pp. 954 –958, nov. 2007.

[88] M. Figueiredo and R. L. Aguiar, "Time precision comparison of digitally controlled delay elements," in *IEEE International Symposium on Circuits and Systems*, pp. 2745–2748, May 2009.

[89] M. Figueiredo and R. L. Aguiar, "Noise and Jitter in CMOS Digitally Controlled Delay Lines," in *IEEE International Conference on Electronics, Circuits and Systems*, pp. 1356–1359, Dec 2006.

[90] K. Blutman, J. Angevare, A. Zjajo, and N. van der Meijs, "A 0.1pJ Freeze Vernier time-to-digital converter in 65nm CMOS," in *IEEE International Symposium on Circuits and Systems*, pp. 85–88, June 2014.

[91] F. Baronti, D. Lunardini, R. Roncella, and R. Saletti, "Experimental characterization of a self-calibrating delay-locked delay-line," in *Southwest Symposium on Mixed-Signal Design*, pp. 94–98, Feb 2003.

[92] P.-J. Chuang, D. Li, M. Sachdev, and V. Gaudet, "A 148ps 135mW 64-bit adder with Constant-Delay logic in 65nm CMOS," in *IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1 –4, Sept. 2012.

[93] P. Alfke, "Efficient Shift Reigsters, LFSR Counters, and Long Pseudo-Random Sequence Generators," *Xilinx Application Note*, 2007.

[94] F. Frustaci, S. Perri, M. Lanuzza, and P. Corsonello, "Energy-Efficient Single-Clock-Cycle Binary Comparator," *International Journal of Circuit Theory and Applications*, vol. 40, no. 3, pp. 237–246, 2012.