Low-power Error-tolerant Digital Logic Circuit Design

by

Yufeng Li

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Integrated Circuits & Systems

Department of Electrical and Computer Engineering

University of Alberta

Abstract

As semiconductor process minimum linewidths have been scaled down to nanometers, digital computing circuits have become more and more complex. Meanwhile, these hardware circuits are required to be easily integrated into portable devices, be reliable even in a noisy environment, and have low power consumption to save energy. In other words, area, reliability and power consumption are three inevitable issues existing in modern Very-large-scale Integration (VLSI) designs. Therefore, researchers have focused effort on the low-power error-tolerant designs and have tried to find a better balance among the three issues, as sometimes there has to be a trade-off between one and another. Breakthroughs are sought in transistor-level designs, gate/cell-level designs and system-level designs. In this thesis, some new gate/cell-level designs and system-level designs are presented, including Markov random field (MRF)-based complementary dual modular redundancy (CDMR), discrete cosine transform (DCT) implementations based on MRF-stochastic logic, coding-based partial MRF (CPMRF) circuit design, and probabilistic-based complementary (PC) logic gate design.

MRF-based CDMR overcomes the large area cost and the vulnerability of the voting circuits in the triple modular redundancy (TMR) configuration and mitigates the effect of soft errors based on the MRF theory and the idea of the combination of stable logic signals. In the CDMR, the proposed two-stage voting circuit contains an MRF feedback structure in the first stage and a merging unit in the second stage. Compared with previous designs, it helps the whole system save the area of one module while lowering the large power consumption with a low error rate.

DCT implementation based on MRF-stochastic logic combines MRF theory and stochastic logic to simplify the hardware without losing high noise immunity for the DCT system in nanoscale conditions. MRF-based gate groups are designed to save area overhead for stochastic adders and multipliers used in one-dimensional DCT (1D-DCT). The proposed design not only achieves better noise-immunity but also saves hardware and power cost when compared with a master-slave version of the design.

CPMRF circuit design summarizes a general mapping method for logic operations. Unlike the conventional MRF designs, CPMRF gate pairs can easily achieve multi-logic operations by sharing a common MRF network. The coding structure complements the loss of the energy and strengthens the stability of the logic “1” and “0” states. An 8-bit carry lookahead adder (CLA) is built to measure the performance of the proposed method. It has relatively high noise immunity with low hardware cost and low power consumption, which corresponds to the theoretical analysis.

PC logic gate design separates logic “1” and “0” signals into either robust bits and weak bits from the perspective of probability. The design tolerates errors in terms of probability. By only choosing robust bits, the proposed logic gates can generate high quality output signals. Also, a general mapping rule is deduced to allow the idea to be easily implemented automatically within the existing Electronic Design Automation (EDA) flows.

Preface

This thesis is a final work for the degree of Doctor of Philosophy at the University of Alberta. Some chapters contain published articles authored or co-authored. All the authors' contributions on the related published articles are listed as follows:

Chapter 2 contains the research published in IEEE Transactions on Very Large Scale Integration as "Feedback-based Low-Power Soft-Error Tolerant Design for Dual Modular Redundancy". Yan Li and I are co-first authors. Yan Li mainly wrote Section I, Section II and a small part of Section III. I finished the remaining content. I was also in charge of simulation, data collection, processing, comparison, figures and tables in the article. Han Jie, Jianhao Hu, Fan Yang, Xuan Zeng, Bruce Cockburn, and Jie Chen all offered valuable suggestions and helped revise the manuscript.

Chapter 3 contains the research published in Proceedings of 2018 IEEE International Symposium on Circuits and Systems as "Low Power Area-Efficient DCT Implementation Based on Markov Random Field-Stochastic Logic". I am the first author. I wrote most of the sections and prepared figures and tables. I also completed simulation, data collection, processing and comparison. Yan Li contributed to some parts of Section II and III. The article was supervised and supported by Deqiang Cheng and Jie Chen. We are also preparing the submission of "Improved Low-Power Cost-Effective DCT Implementation Based on Markov Random Field and Stochastic Logic" to IEEE Transaction on Circuits and Systems

for Video Technology. I am the first author as well and responsible for the preparation of the manuscript.

Chapter 4 contains the research published in IEEE Journal of Solid-State Circuits as “Low-power Noise-Immune Nanoscale Circuit Design Using Coding-based Partial MRF Method”. Yan Li and I are co-first authors. Yan Li wrote Section I and some parts of Section II and III. I completed the remaining content, including some parts of Section II and III, Section IV, Section V, figures and tables. I-Chyn Wey provided the instruments and instructed me to test the chip. I was responsible for simulation, data collection, processing and comparison. Xiaoxue Jiang helped the RTL debugging and behavioral verification of the chip. I-Chyn Wey, Fan Yang, Xuan Zeng and Jie Chen all helped revise the manuscript and offered generous support.

Acknowledgements

I would like to acknowledge the financial support provided by China Scholarship Council (CSC) and Alberta Innovates Technology Futures (AITF). This funding helped me to focus on my research without having to be concerned with financial problems. I would also like to acknowledge my supervisor Dr. Jie Chen, who offers me the opportunity to do the research and other professors, who supported the projects. I couldn't forget expressing my sincere appreciation to the professors who teach me, including Dr. Bruce Cockburn, Dr. Jie Han, Dr. James Miller, Dr. Xihua Wang, Dr. Douglas Barlage and Dr. Yindi Jing. Thank you for helping me build solid background knowledge for my research. I also need to appreciate CMC microsystems for their support of chip manufacturing and CAD tools. In my four-year study, I'm honored to work with many excellent graduate students and postdoctoral researchers like Yan Li, Xiaojian Yu, Xiaoxue Jiang, Wei Zhang, Shi-ang Qi, Alexandra Savchenko, Donghai Lin and so on. Finally, I would like to acknowledge my grandparents and parents for their love and support. I wish the people I mentioned above good health and happiness always!

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Acronym	Definition
AI	Artificial Intelligence
ANT	Algorithmic Noise-Tolerant
ASET	Algorithmic Soft-Error Tolerance
AS	Adder-Subtractor
ASM	Adder-Subtractor with Multiplier
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BSIM4	Short-channel IGFET Model 4
CDMR	Complementary Dual Modular Redundancy
CLA	Carry-Lookahead Adder
CMOS	Complementary Metal Oxide Semiconductor
CPMRF	Coding-based Partial MRF
DCVS	Differential Cascode Voltage Switch
DCT	Discrete Cosine Transform
DMR	Dual Modular Redundancy
DSM	Deep Sub-Micron
ECC	Error Correcting Codes
EDA	Electronic Design Automation
EDP	Energy-Delay Product
FGSET	Fine-Grained Soft-Error Tolerance

FinFET	Fin-gate Field-Effect Transistor
FTL	Feedthrough Logic
IC	Integrated Circuit
KLD	Kullback-Leibler Distance
LEAP	Layout Design through Error-Aware Transistor Positioning
LET	Linear Energy Transfer
LSB	Least Significant Bit
MS	Master-Slave
MSB	Most Significant Bit
MUX	Multiplexer
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MRF	Markov Random Fields
1D-DCT	One-Dimensional Discrete Cosine Transform
PCL	Probabilistic-based Complementary Logic
PCMOS	Probabilistic CMOS
PDSOI	Partially Depleted Silicon-On-Insulator
PTMR	Partial Triple Modular Redundancy
RCA	Ripple Carry Adder
RPR	Reduced-Precision Redundancy
SC	Stochastic Computing
SER	Soft Error Rate
SET	Single-Event Transient
SEU	Single Event Upset

SVM	Support Vector Machine
SRAM	Static Random-Access Memory
SSI	Small-Scale Integration
TCAD	Technology Computer Aided Design
TL	Turtle Logic
TMR	Triple Modular Redundancy
ULSI	Ultra-Large-Scale Integration
VLSI	Very-Large-Scale Integration

Chapter 1

Introduction

1.1 Background

The concept of the Integrated Circuit (IC) was first described by G. W. A. Dummer as a single semiconductor "chip" that includes multiple components, such as transistors, together with wiring all formed as layers on the top surface of the chip [1]. It become reality through the contributions of numerous scientists and engineers. With the downscaling of Complementary Metal Oxide Semiconductor (CMOS) technology, the IC, be it analog, digital or mixed signal, developed from Small-scale Integration (SSI) through to Very-large-scale Integration (VLSI) or even Ultra-large-scale Integration (ULSI). The pace of IC improvement was predicted by the well-known Moore's Law that the number of transistors integrated in an IC would grow exponentially in linear time, doubling every 12 to 18 months [2]. Because of the shrinking dimensions, power dissipation becomes the bottleneck in IC design [3]. It limits the performance of a chip. For digital IC design, low power has been a mainstream trend to meet the demands of a "Digital Lifestyle" [4].

In order to solve the problem, we first need to understand the causes of CMOS power consumption. In general, CMOS power consumption can be divided to dynamic power consumption and static power consumption [5]. Static power consumption results from leakage such as gate-oxide power leakage. The main source of power consumption is dynamic power consumption since the static

power consumption of a CMOS device is typically very low [5]. Although it is found that static power consumption is catching up with the dynamic power consumption in Deep Sub-micron (DSM) CMOS circuits [6,7], as illustrated in [8], dynamic power consumption still contributes the majority of overall power consumption. For a logic gate, dynamic power consumption results from the switching behavior of the input signals from V_{DD} to V_{SS} and vice versa. The average dynamic power consumption of a single logic gate can be calculated by the following equation [9]:

$$p_{dynamic} = CV_{dd}^2 f \tag{1.1}$$

where C is the load capacitance on the output bus; V_{dd} is the supply voltage and f is the switching frequency of the input signals. It's a quadratic function of V_{dd}. For a microcontroller which consists of many CMOS devices, Equation (1.1) becomes

$$p_{dynamic} = \alpha CV_{dd}^2 f \tag{1.2}$$

where α is an activity factor. It's hard to control the load capacitance which is related to the internal lay-out and design of a chip, while the switching frequency can only be regulated by limiting the switching of input signals and clock signals. Since the supply voltage is a squared factor, the best way to cut down the dynamic power consumption is to reduce the supply voltage.

When a digital IC is working at a low supply voltage, reliability should be taken into consideration. This is because noise is a critical challenge for the reliability of a submicron chip. Noise is dynamic and random as follows:

- Thermal noise: As the increase of the chip density and downscaling of transistors, thermal noise voltage is more likely to cross the threshold voltage due to the increase of local temperature and the decrease of transistor capacitance. [10]
- Radiation-induced noise: Noise caused by radiation such as charged particles, neutrons and cosmic rays can result in soft errors. This commonly occurs in aerospace and military devices. Also, it can be induced by package materials which contain radioactive impurities. [11]
- Crosstalk noise: Neighboring wires in an IC chip are getting closer to each other due to the increasing chip density. Under these circumstances, the capacitive coupling becomes a noise source. [12]
- Other noise sources: It can be at the microscopic level such as shot noise, hot-carrier effect, threshold variation. [10]

Excessive noise, such as hot-carrier effect noise due to increasing drift velocity fluctuations and shot noise, because of carrier number fluctuations, cannot directly uses the conventional Johnson–Nyquist thermal noise model since excessive noise is non-equilibrium [13]. Researchers in [13] also demonstrated that the noise signal arising from thermal noise enhanced by threshold variation and crosstalk noise roughly follows a Gaussian distribution. Radiation-induced noise can be modeled as a double-exponential current source which result in quick-to-rise and slow-to-fall spikes for example [14]. Therefore, the noise model used in this thesis is mainly Gaussian. For the soft-error simulation, we also use the double-exponential current model.

In a VLSI circuit, noise can drive the voltage of a node to a lower or higher level which is away from the nominal value, as shown in Figure 1.1 [15]. An error causes a logical level upset when the deviated voltage of the node exceeds the logical threshold (typically 1/2Vdd). This situation gets worse when the supply voltage is very low, around the threshold voltage of a transistor.

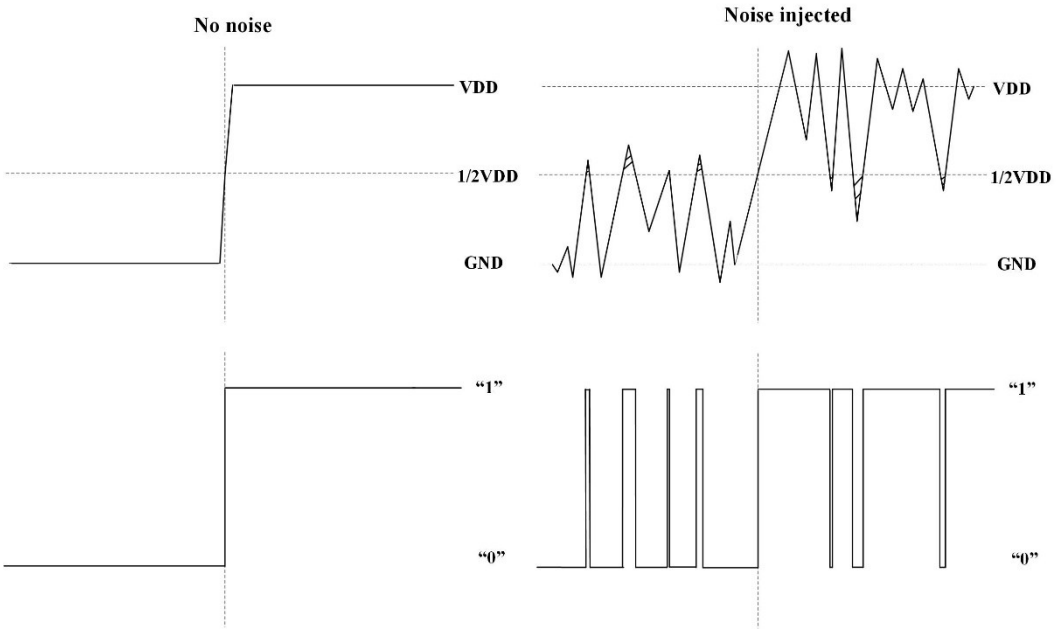


Figure 1.1: Noise can cause errors.

In addition to reliability, an IC's portability, which is determinated by the size and the weight of a chip, is another concern in digital IC design [16]. Therefore, researchers have been quite interested in low-power error-tolerant digital circuit designs. Generally, designers need to understand the trade-offs among power consumption, reliability and hardware cost. This becomes the driving force for designers to find a better balance as the most cost-effective solution.

1.2 Low power error-tolerant design overview

Low power error-tolerant logic circuit designs can be classified into three levels, including the transistor level, the gate/cell level and the system level. At the transistor-level, researchers try to enhance the error tolerance in terms of an IC's structure, dimension, material and layout design rules.

In recent years, the opinion has been expressed increasingly that Moore's Law is dying [17, 18]. On the contrary, some researchers believe that Moore's law will continue in different forms if we redefine it [19]. The redefined Moore's law identities a research direction, 3D technology [20]. This new direction drives the development of transistor-level low power error-tolerant circuit design. Hector Villacorta et al. [21] came up with an alternative: increasing the fin height of transistors, rather than increasing the numbers of fins, enhances the performance of Fin Field-Effect Transistor (FinFET) based Static Random-access Memory (SRAM) cells in terms of their ability to tolerate radiation-induced soft error. The results of their analysis indicate that increasing fin height can help the cell get enough critical charge without increasing the area overhead. P. Oldiges et al. [22] measured the soft error upset cross section for both Partially Depleted Silicon-On-Insulator (PDSOI) FinFET SRAMs and SOI FinFET SRAMs. It was found that SOI FinFET circuits have much lower soft error rates than PDSOI as the same as the result of Technology Computer Aided Design (TCAD) modeling. Jinhyun Noh et al. [23] compared the soft error rates of planar Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) and FinFET devices. They not only pointed out the better performance of FinFET but also explained the reason FinFET designs are less sensitive to soft errors due to the increase in the threshold

Linear Energy Transfer (LET) and the shape of the transistor. Norbert Seifert et al. [24] reported the invention of second-generation 3D tri-gate transistors that improve the radiation-induced upset rate in logic devices fabricated in 14-nm technology. In their previous research [25], it has been proved that 22-nm tri-gate devices have already had excellent Single Event Upset (SEU) benefits. As 14-nm second-generation 3D tri-gate transistors have taller and narrower fins than 22-nm first-generation trigate transistors, the Soft Error Rate (SER) improvement of the second-generation transistors, compared with that of the first-generation transistors, is due to the increase in the critical charge and the reduction of charge collection. Huichu Liu et al. [26] investigated the sea-level SER performance of Si FinFETs, III-V FinFETs and III-V Tunnel FETs based on experimental models for both SRAM cells and combinational logic. In their conclusion, III-V Tunnel FETs present the best soft error tolerance within the voltage range from 0.3V to 0.6V, which they attributed to the reduction of bipolar gain, superior Miller capacitance effect and better latching window masking. Huichu Liu et al. [27] furtherly evaluated the characteristics of steep switching Tunnel FETs at ultra-low power. They confirmed that the steep switching TFET is a promising device with better soft error performance, lower flicker noise level and less temperature drift. Different from the above research, Hsiao-Heng Kelin Lee et al. [28] presented a layout design through error-aware transistor positioning principle that resists the harmful effects of soft errors. Their special layout design was applied to three different kinds of master-slave flip-flops in a 5-mm×5-mm chip. Compared with other traditional designs, the Layout Design through Error-Aware Transistor

Positioning (LEAP) flip-flop experiences 2000 times fewer soft errors but costs more silicon area. These researchers established fundamental optimal methodologies which can be combined with any other higher-level designs in VLSI.

Some researchers have improved the performance of logic circuits by redesigning basic gates and cells or inserting some special structures between two traditional gates. Most of them address the problem from the perspective of noise immunity to tolerate errors caused by noise. Milos Stanisavljevic et al. [29] implemented a kind of Differential Cascode Voltage Switch (DCVS) logic to absorb errors in logic circuits. The DCVS logic realization consists of differential inputs for each logic cell. Mariem Slimani et al. [30] invented a new cross logic that tolerates errors caused by manufacturing defects based on standard logic cells. The new cross logic matches one logic with another complementary one-by-two crossed CMOS inverters to correct errors generated from a faulty path. One probabilistic approach is the use of Probabilistic CMOS (PCMOS) devices [31]. The supply voltage should be chosen based on Energy Delay Product (EDP) and probability of correctness. K. Nepal et al. [13, 32, 33] first mapped Markov Random Fields (MRF) probabilistic theory to logic circuits to maximize the probability of correct states by feedback from neighboring circuit nodes. They later optimized MRF elements by using an inverter instead of a three-input NOR gate for the feedback [34]. They also presented an MRF-based implementation of Error Correcting Codes (ECC) in [35]. The design in [13] was first implemented as a real silicon design by I-Chyn Wey et al. [36] as an 8-bit carry-lookahead

adder in a 180-nm CMOS process. The testing result achieved a 10^{-6} Bit Error Rate (BER) when injecting 10-dB SNR noise to demonstrate the feasibility of MRF circuit design for increased noise immunity. The related future research direction was pointed out by K. Nepal et al. [37], that the hardware cost of an MRF implementation could possibly be achieved with the help of supergates, implied dependence and clique variable sharing. Based on the above idea, I-Chyn Wey et al. [38] further improved the traditional direct-mapping MRF circuit design by simplifying the compatibility function and successfully reduced the area overhead using a new Master-Slave (MS) MRF structure. In 2013, a previous noise tolerant MRF combinational circuit design was extended to sequential design in [39]. The MRF latch combines H-tree structure and a cross-coupled feedback interlocking mechanism with the MRF theory to maintain high reliability in a noisy environment. Common-feedback Schmitt trigger MRF elements can further reduce the hardware cost without affecting the performance of noise-immunity [40]. Kaikai Liu et al. [41] simplified the master-slave MRF logic gates and proposed a general cost-effective MRF logic gate design. The cost-effective design has similar noise-tolerant performance as the MS one but saves some area. Zhenghao Lu et al. [42] inserted a DCVS structure into the middle of a cost-effective MRF inverter to enhance the noise-immunity capability. Xinghua Yang et al. [43] generalized Zhenghao Lu's idea for universal gates and measured Kullback-Leibler Distance (KLD) to evaluate the effectiveness for each DCVS based MRF noise-tolerant logic gate. Yan Li et al. optimized the MRF-based circuit design by extending the standard cells [44], proposing area-sharing

cyclic path [45] and designing partial clique energy pairs with cyclic paths [46]. Some researchers studied MRF-based circuits theoretically and found that MRF elements can effectively tolerate noise but have one order of magnitude bigger propagation delay than conventional non-MRF circuits [47, 48, 49]. Unlike I-Chyn Wey's latch design, the duration-observation master-slave flip-flop proposed by Yukiya Miura et al. [50] is intended to mitigate the influence of transient pulses induced by Single-Event Transients (SETs) and crosstalk. Two implementations were brought out using different effective cells for the second sampling time. Preetisudha Meher et al. [51] reports a new comparator design based on semi domino logic and footer cells for arithmetic circuits. From the simulation results in Cadence using UMC's 180-nm process, the new designed comparator is superior to other comparators in power, delay (75% to 90% less) and leakage current (30% to 50% less). R. Devi Sindhu et al. [52] used 12 transistors to form a SEU-protected memory cell. The proposed 12T SRAM cell overcomes the drawback of the previous quatro10-transistor SRAM cell while it can work perfectly even if single node or multi-mode upsets occur. Lancelot Garcia-Leyva et al. [53] introduced a probabilistic logics, called Turtle Logic (TL), aimed at tolerating errors induced by noise. TL tactfully utilizes redundant data as complementary components to force the output to keep the correct values. Sauvagya Ranjan Sahoo et al. [54] followed new design techniques to improve the performance of Feedthrough Logic (FTL) affected by noise. Stacked Technique and Mirror PMOS technique were used in an inverter and a NAND gate design, respectively as case studies. These techniques trade off area for the ability to

tolerate noise. Some of the above researchers introduced new proposed designs to traditional logic in order to improve the reliability of logic circuits.

Besides gate-level designs, some researchers have contributed strategies to increase circuit robustness from a system-level. Among them, the most well-known one, Triple Modular Redundancy (TMR), was first proposed by Von Neumann [55]. The idea describes a system structure which uses three replicated modules with a two-out-of-three majority voting circuit to decide the output to trade area for reliability. R. E. Lyons et al. [56] carried out the idea on a hypothetical computer to assess the effectiveness of the TMR concept. However, later researchers like Jacob A. Abraham et al. [57], threw doubt on the reliability of TMR because of two weaknesses, including module failures and voter failures, and gave an algorithm to evaluate the reliability. R.V. Kshirsagar et al. [58] also pointed out the weakness of the original voting design and presented a better fault-tolerant one as a replacement. The novel voter contains two XOR gates, a special encoder and a multiplexer to deal with the case where the voter fails. Rahul Parhi et al. [59] addressed the area and power consumption bottlenecks of TMR with a new concept called Partial Triple Modular Redundancy (PTMR). “Partial” indicates that the approach keeps the Most Significant Bits (MSB) but truncates the Lower Significant Bits (LSB). It was observed in their theoretical analysis that the probability of being at fault increases with the increase of the number of the used voting circuits. A similar idea was proposed by S. Baloch et al. [60] to harden circuits while successfully saving a certain amount of area and power. The difference lies in the objectives aimed by the partial idea that only

SEU sensitive gates are protected by TMR technique. On the contrary, some researchers, like Ryo Terada [61], paid more attention to the ability of error tolerance and prefer N-modular redundancy for high radiation environments. It was found that the number of the modules used in the redundant idea can be reduced as well. Therefore, Dual Modular Redundancy (DMR) is more area-efficient. J. Teifel designed a DMR scheme in [62] based on the gate-level circuit design of the C-element. The robust C-element DMR [63] is an improved transistor-level implementation for [22]. Another DMR design [64] uses a multiplexer in the voting circuit. An Algorithmic Noise-tolerant (ANT) Technique [65] consists of a main block, a reduced-precision redundancy (RPR) block, and a decision block to resist soft-errors. An extended ANT design, called Algorithmic Soft-error Tolerance (ASET) [66], can reduce soft errors caused by DSM noise. The Fine-grained Soft-error Tolerance (FGSET) design [67] is an effective approach to eliminate possible error propagation. Stochastic Computing (SC) is another algorithmic low-cost error-tolerant technique [68, 69, 70]. It can be applied on the whole system to tolerate errors because of its unweighted presentation in a sequence form of probabilities. Researchers have adopted SC in the field of Artificial Intelligence (AI) [71, 72, 73, 74] and signal processing [75, 76, 77, 78, 79]. In this thesis, we only focus on the gate/cell-level and system-level designs.

1.3 MRF-based Circuit Methodology

Since most of the designs in this thesis are based on the theory of MRF, this section will first introduce the procedure of how to convert a logic circuit to a

corresponding MRF-based circuit design and then give theoretical explanation to the reasons why researchers have made an effort to map MRF theory to circuit design.

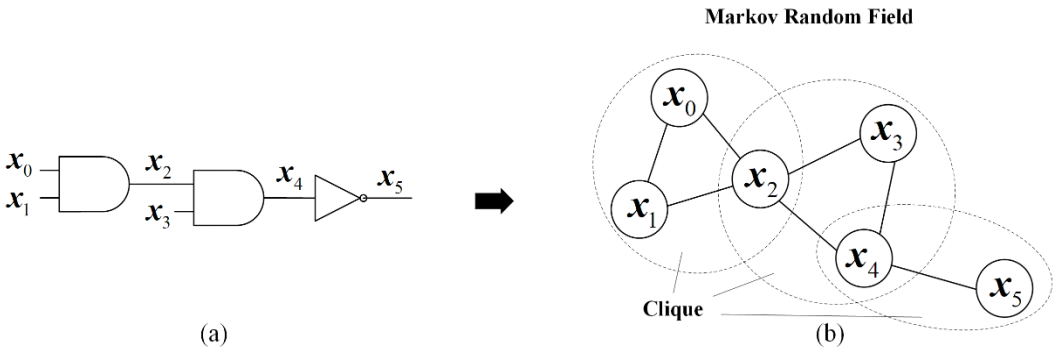


Figure 1.2: A mapping example: (a) a logic circuit; (b) the corresponding MRF network.

Figure 1.2 is used as an example to explain how to convert a logic circuit into an MRF network. There is a random logic circuit shown in Figure 1.2(a). It contains three logic gates and five input/output nodes x_0, x_1,…, x_5. The three logic gates divide the five nodes into three groups $\{x_0, x_1, x_2\}$, $\{x_2, x_3, x_4\}$ and $\{x_4, x_5\}$. In each group, nodes are directly connected through a logic gate. In order to build a mathematical model, we assume that these nodes belong to a set $\boldsymbol{X} = \{x_0, x_1, ..., x_5\}$. Therefore, the set X has three subsets $\{x_0, x_1, x_2\}$, $\{x_2, x_3, x_4\}$ and $\{x_4, x_5\}$. It is assumed that the probability of a node is always positive. Second, it is found that the state of a node is dependent only on the state of the nodes in the same group. The above two points satisfy positivity and markovianity as a variable must be independent of others except its neighbors in the same subset [80], so they can be converted to the corresponding MRF variables and form an MRF network. In MRF theory, the subsets have a special name, called a

Table 1.1: Energy Truth Table of an Inverter

Input x	Output y	State	Clique Energy
0	0	Invalid	0
0	1	Valid	-1
1	0	Valid	-1
1	1	Invalid	0

Table 1.2: Energy Truth Table of a Two-input NAND

Input x_1	Input x_2	Output y	State	Clique Energy
0	0	0	Invalid	0
		1	Valid	-1
0	1	0	Invalid	0
		1	Valid	-1
1	0	0	Invalid	0
		1	Valid	-1
1	1	0	Valid	-1
		1	Invalid	0

"clique". It means a group of variables which depend on each other. The three logic gates in Figure 1.2 (a) imply the three cliques shown in Figure 1.2(b). As an MRF graphical network reflects the relationship between the variables, it is marked by the short edges shown in Figure 1.2(b) representing whether a variable depends on another. In a clique, a variable is a neighbor of all the other variables [13]. Note that a variable only depends on its neighbors. In this way, we successfully transform a logic circuit to an MRF network. It indicates that MRF theory can be directly applied on the circuit design.

The next step is to design the MRF-based logic elements. Ideally, signals passing through logic gates should be correct all of the time. However, although

there are no defects in logic gates, signals can be easily corrupted in the real environment by the many possible sources of noise. The voltage of a node can be deviated from the original value, which results in errors and incorrect states. A state refers to a combination of input and output logic values. Although we only want to obtain correct outputs, the corresponding inputs must also be considered as each output node only depends on its neighbors (its inputs). This is why we focus on the concept of states. According to the MRF theory, a circuit is more likely to operate perfectly by working in a set of correct logic states when correct logic states have the lowest clique energy [32, 81]. The best way to investigate the situation is to combine Boolean algebra with energy functions. First, the correct states are converted into valid minterms, while the incorrect states are converted into invalid minterms. Second, we need to build an energy truth table which combines the correct and incorrect states with the corresponding inputs and outputs in order to measure whether the clique function for a logic gate is reasonable or not. If the clique energy of a logic gate is equal to the sum of all valid minterms, then the energy of correct states (1) is larger than that of incorrect states (0). However, this contradicts MRF theory. Therefore, the clique energy of a logic gate is defined to be -1 times the sum of all valid minterms. We use an inverter and a two-input NAND gate as examples. Table 1.1 is the energy truth table of an inverter. Table 1.2 is the energy truth table of a NAND gate. For an inverter, the energy function should be $U(x,y) = -(\bar{x}y + x\bar{y})$. In Table 1.1, the energy of a correct state is -1 which is lower than that of an incorrect state. For a NAND gate, the energy function should be $U(x_1, x_2, y) = -(\overline{x_1}\,\overline{x_2}y + \overline{x_1}x_2y +$

$x_1\overline{x_2}y + x_1x_2\overline{y})$. In Table 1.2, the energy of a correct state is also lower than that of an incorrect state.

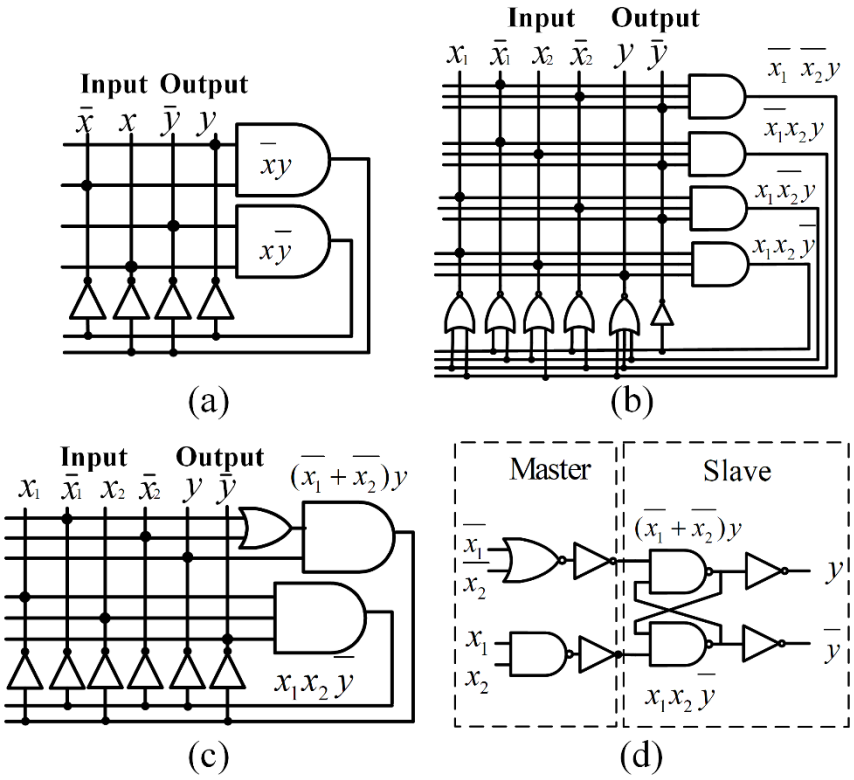


Figure 1.3: MRF circuits: (a) an MRF inverter [13] (b) an MRF NAND [13] (c) an Area-efficient MRF NAND [34] (d) an MS MRF NAND [38].

The following step shows the hardware implementation that exploits the clique energy function. Figure 1.3(a) is a conventional schematic diagram for an MRF-based inverter. The two AND gates implement the terms $\bar{x}y$ and $x\bar{y}$, respectively, in the energy function. The outputs will be fed back to all nodes in a grid network to strengthen the input and output signals. Similarly, Figure 1.3(b) implements an MRF-based NAND gate. Four AND gates perform the valid terms of $\overline{x_1}\ \overline{x_2}\ y$, $\overline{x_1}\ x_2\ y$, $x_1\ \overline{x_2}\ y$ and $x_1\ x_2\ \overline{y}$. The energy function of a NAND gate can

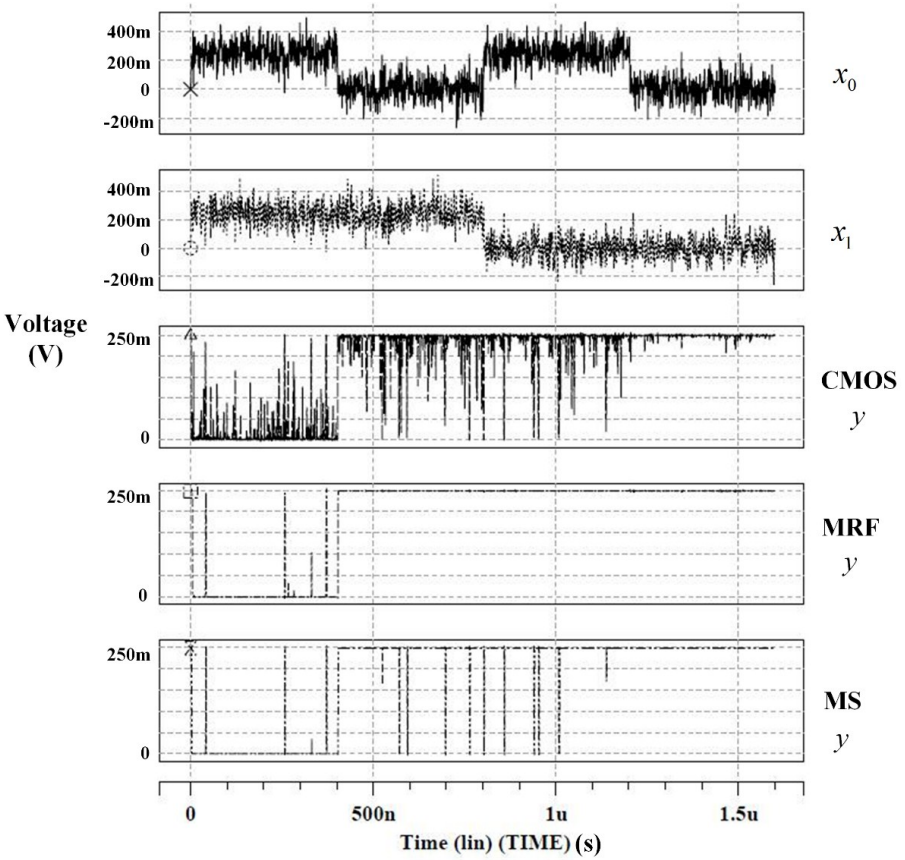


Figure 1.4: HSPICE simulation for a CMOS NAND gate, an MRF NAND [33] and an MS MRF NAND [38].

Table 1.3: Area and Critical Path Comparison for Three NAND Gates

Logic	NAND		
Scheme	CMOS	MRF	MS
Area	$2A$	$30A$	$14A$
Critical path	2τ	13τ	9τ

A is the area of an inverter; τ is an inverter delay

be simplified to $U(x_1, x_2, y) = -[(\overline{x_1} + \overline{x_2})y + x_1 x_2 \overline{y}]$. Based on the simplified function, there are two kinds of circuit designs: one is shown in Figure 1.3(c); the other is shown in Figure 1.3(d). Figure 1.3(d) is called as an MS MRF NAND gate.

The MS version saves even more area than the area-efficient one proposed in [34]. It is a classical cost-effective design. Figure 1.4 shows the input and output signal waveforms of a NAND gate operating under the interference of Gaussian noise with the probability density function $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)}{2\sigma^2}}$ (μ=0V, σ =0.08V) at the supply voltage 0.25V. Noise is generated by the Matlab command $noise = std\_deviation \times \text{randn}(t,1)$, where t is the number of nodes affected by noise; and *std_deviation* is the standard deviation. The noise is injected to input signals by a .sp file in HSPICE. The assumed operating temperature is 50 ℃. The simulation library is the 65-nm CMOS library from Berkeley. In Figure 1.4, it is shown that the MRF NAND gate and the MS NAND gate can effectively tolerate noise, while the CMOS non-MRF NAND gate frequently switches between the correct and incorrect logic states along with the injected noise. However, MRF elements require more area and have longer propagation delay, as shown in Table 1.3. This has inspired researchers to optimize the mapped circuit. Based on the MS methodology, there are some other simplified MRF logic elements such as DCVS-MRF [42, 43], CENT_MRF [41], MRF-CL-Schmitt [40] and extensional MRF [44]. However, in these designs, an MRF network can only build an MRF-based logic element. This becomes a limitation to further simplifying the whole system. Although some researchers [45, 46] have proposed the idea of area-sharing to cut down the required hardware cost of the MRF elements, these designs have either the decreased performance due to the incomplete MRF networks or very long propagation delay.

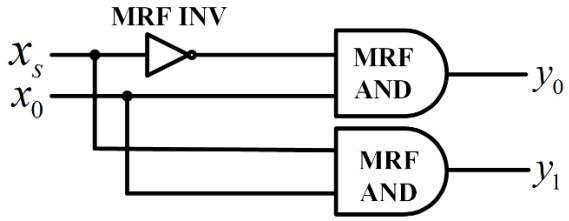


Figure 1.5: Local mapping based MRF DEMUX.

The final step is to replace each conventional logic gate by an MRF-based logic gate in a logic circuit. For example, Figure 1.5 is a logic circuit of a Demultiplexer (DEMUX). The design of an MRF version involves using the replacements of the logic gates by an MRF-based inverter and two MRF-based AND gate. Although MRF-based circuit design trades off area for reliability, the increase of the area won't increase the power consumption when the supply voltage is lowered to near the threshold voltage. On the contrary, under the minimal supply voltage when working properly, a traditional MRF circuit consumes 40.7% less power consumption than a conventional non-MRF design, while the area of the traditional MRF circuit is 13 times the area of conventional non-MRF design [38].

The reason why we apply the MRF theory on the circuit design comes from an important theorem called the Hammersley-Clifford theorem. It describes what determines the joint probability for a set of MRF variables [13]:

$$P(\boldsymbol{X}) = \prod_{c \in C} \frac{1}{Z} \exp\left(\frac{-U_c(x_c)}{k_b T}\right) \tag{1.3}$$

where C stands for the finite set of cliques; $U_c(x_c)$ is the energy function of the c-th clique; Z is a constant that is used to normalize values and $k_b T$ represents the thermal energy. Equation (1.3) shows that the value of the joint probability for a

set of variables depends on their clique energy. Since the reliability of a logic circuit can be regarded as the joint probability of the correct states, if there exists a method to build an MRF network for a logic circuit, improving the reliability of a circuit by increasing the joint probability of the correct states is simply converted to the minimization of the clique energy [13]. We do not need to directly map the theorem to a hardware circuit. We only need to map the energy function to a circuit design. Both the theoretical analysis and simulation results support the claim that MRF-based methodology is an effective alternative to constructing low-power error-tolerant digital circuits. The methodology, which belongs to space redundancy, trades off area for reliability and power consumption.

In conclusion, the function of MRF networks is to obtain correct logic values as much as possible in the presence of noise based on the reinforcement of the correct states with the highest probability. In other words, logic states are determined by considering the probability distribution of signals in a way of a multi-iteration process to maximum the probability of correct states. There are other options to achieve noise immunity such as low-pass filters. The frequency of noise depends on the noise itself. Only when the frequency of noise is higher than that of circuit signals can we use the low-pass filter to remove the noise. However, it's hard to predict when, where and what kind of noise will occur. The effectiveness of low-pass filters is limited. You may put filters at the inputs to increase the quality of input signals, but noise may not be only integrated in the input signals. Every circuit could have a chance to be interfered by random noise,

as well as the filter itself. The serious propagation delay caused by filters can be another problem. Also, designing a filter can be a lot of trouble. Cut-off frequency and impendence matching should be considered carefully. In contrast, probabilistic-based MRF designs, tolerating noise by maximizing the probability of valid states, not by the difference of frequency, are more cost-effective since they are easier to replace traditional CMOS components by prepared corresponding instances and are more comprehensive toward noise. In addition, probabilistic-based MRF designs have high commercial value. For a portable device such as a smartphone, a tablet and a wireless headphone, the battery power is limited. Customers look forward to long battery life device. One way is to increase the battery capacity. The other way is to reduce the power consumption. However, if we reduce the supply voltage to close to the transistor threshold, the background noise or wire crosstalk / coupling can impact circuit function. In this case, MRF can trade silicon area for noise-tolerance under low power operation. The other application is the hardware implementation of data mining and neutral computing algorithms. For example, Support Vector Machine (SVM) is well known for its pattern classification. Noise can lead to the misclassification when working at a low power supply voltage. In this case, MRF designs can help the circuit function well to avoid misclassification.

1.4 Contribution and Novelty of this Thesis

In this thesis, four low-power error-tolerant logic circuit design styles are presented and compared including MRF-based CDMR design, MRF-stochastic logic DCT design, CPMRF circuit design and PC Logic gate design. For the

MRF-based CDMR design, 1) it saves large area overhead being a DMR method; 2) the MRF-based voting circuit is more robust than the previous designs; 3) it is flexible to apply on different systems including single-stage application and multistage application. For the MRF-stochastic logic DCT design: 1) it combines MRF methodology and stochastic logic to alleviate the large hardware cost required by the MRF-based elements; 2) each proposed MRF-based element used in a stochastic adder and a stochastic multiplier can implement two logic operations by a shared MRF network to save more area; 3) the DCT algorithm is simplified to reduce the required number of SC multipliers to keep the high capability of noise-immunity unaffected. For the CPMRF circuit design: 1) it offers a general mapping method based on the idea of sharing for complementary and non-complementary logic pairs; 2) it integrates the coding method in the construction of MRF networks to transform weak logic bits to strong logic bits; 3) it is highly cost-effective in terms of hardware cost, power consumption and noise-immunity. For the PC Logic gate design, 1) the proposed error-tolerate unit can generate robust output bits from the perspective of probability based on the asymmetric output distribution of a NAND/AND gate and a NOR/OR gate; 2) the error-tolerate unit can be selectively applied to any blocks in a circuit according to different requirements. In general, the contribution and novelty of these designs lie in: 1) relatively large area and power savings; 2) cost-effectiveness as a better trade-off among hardware cost, power consumption and reliability, compared with previous designs. They offer alternatives other than conventional methods to tolerate errors at an ultra-low supply voltage in nanoscale digital VLSI designs.

They help make the digital VLSI designs easier to integrate in portable devices, more reliable in noisy environment, and eco-friendlier as relatively low power consumption is required.

1.5 Thesis Outline

This thesis is organized into six chapters to present four low-power error-tolerant logic circuit designs from the conceptual idea to the implementation. Chapter 1 explains the reason why low-power and error-tolerant designs are indispensable as a trend in VLSI design. Chapter 1 also reviews the research progress from the transistor level, the gate/cell level and the system level. It also states the contributions and novelty of this thesis. Chapter 2 focuses on the MRF-based CDMR design. It first introduces conventional TMR theory followed by a section that describes the proposed MRF-based two-stage design and the section that demonstrates its effectiveness. Chapter 3 is the DCT implementation based on MRF-Stochastic logic. It includes a detailed introduction to the DCT algorithm and stochastic logic. Based on the background knowledge, it should be easier to understand the proposed MRF-stochastic logic DCT design. Chapter 4 covers general CPMRF circuit design. There are four sections in this chapter, including a section that describes coding-based partial MRF methodology, a section that presents simulation results, a section that describes a chip implementation and a section for the conclusion. Chapter 5 investigates the idea of PC Logic gate design. It first analyses the behavior of noise-interfered logic gates. Based on the findings, the next section is concerned with the design procedure of PC logic gates. The remaining sections of Chapter 5 discuss the simulation results and

conclude the whole design. Finally, Chapter 6 summarizes the above designs and describes possible future work.

Chapter 2

MRF-based Complementary Dual Modular Redundancy

2.1 TMR

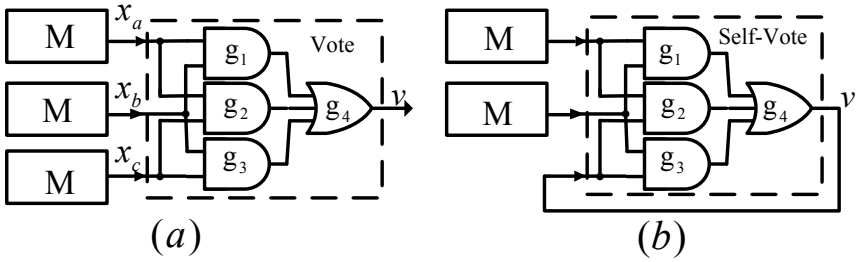


Figure 2.1: TMR, DMR circuits: (a) TMR [55], (b) DMR [62].

TMR configurations can tolerate errors because they can separate three modules into two correct modules and an incorrect module. If there is only one incorrect module coupling to the inputs of the voting circuit, that module's output bit cannot change the final outputs. Since the number of correct modules should be greater than the number of incorrect modules, at least three modules are required. The TMR system [55] shown in Figure 2.1(a) has three replicated module Ms and a majority voting circuit. The voting circuit selects a correct signal based on the logic function Equation (2.1):

$$v = x_a \cdot x_b + x_b \cdot x_c + x_a \cdot x_c \tag{2.1}$$

where v is the final output of the voting design. x_a, x_b and x_c are three individual results obtained from previous calculations. Assume that the error probabilities of the outputs of the three module Ms, x_a, x_b and x_c, are ε_a, ε_b and ε_c. We can calculate the erroneous probability of the output of the voting circuit v shown in

Table 2.1. If $\varepsilon_a = \varepsilon_b = \varepsilon_c = \varepsilon$, we can obtain the erroneous probability of TMR as in Equation 2.2.

$$P_{\text{TMR}} = \varepsilon^3 + 3(1-\varepsilon)\varepsilon^2 \tag{2.2}$$

Table 2.1: Erroneous Probability Table for TMR

M	x_a	x_b	x_c	v	Error probability
0	0	0	0	0	-
	0	0	1	0	-
	0	1	0	0	-
	0	1	1	1	$\frac{1}{2}(1-\varepsilon_a)\varepsilon_b\varepsilon_c$
	1	0	0	0	-
	1	0	1	1	$\frac{1}{2}\varepsilon_a(1-\varepsilon_b)\varepsilon_c$
	1	1	0	1	$\frac{1}{2}\varepsilon_a\varepsilon_b(1-\varepsilon_c)$
	1	1	1	1	$\frac{1}{2}\varepsilon_a\varepsilon_b\varepsilon_c$
1	0	0	0	0	$\frac{1}{2}\varepsilon_a\varepsilon_b\varepsilon_c$
	0	0	1	0	$\frac{1}{2}\varepsilon_a\varepsilon_b(1-\varepsilon_c)$
	0	1	0	0	$\frac{1}{2}\varepsilon_a(1-\varepsilon_b)\varepsilon_c$
	0	1	1	1	-
	1	0	0	0	$\frac{1}{2}(1-\varepsilon_a)\varepsilon_b\varepsilon_c$
	1	0	1	1	-
	1	1	0	1	-
	1	1	1	1	-

As one of the obvious drawbacks is the large area overhead, selective TMR [82] and partial TMR [59] were proposed to reduce the significant area overhead. In fact, these area-efficient designs reuse the key architectural element of TMR by using three identical modules. To sidestep the cost of three copies of module M, some researchers have proposed a so-called dual-modular redundancy (DMR)

[62]. These DMR schemes use only two modules M, but the voting circuit is almost the same size and cost as the voter in TMR. Although it has better performance than the conventional TMR, the structure in [62] is indeed a special version of self-purging-redundancy [83]. It has a self-voting unit shown in Figure 2.1(b) because it feeds back the output to one of its inputs, and thus removes the need of an extra redundant module. Therefore, it is in fact a subclass of a TMR design since it keeps the design rules of TMR. The above TMR [55] and DMR [62] both make the ideal assumption that voting circuits will never fail [84]. Generally, a TMR system with ideal voting circuits never fails if none of the x_a, x_b or x_c fails or if one of these outputs of the three modules fails. However, when an ultra-low supply voltage is applied, circuits fabricated with a deep sub-micrometer technology (DSM) are more likely to fail. Consequently, TMR has two drawbacks. First, the performance of a TMR system can be weakened when errors occur in the voting circuits. Second, replicated modules and voting circuits are big hardware costs. For a multi-stage application, the voting circuits should be embedded in each stage to improve the performance of the overall system shown in Figure 2.2. The required replicated modules and voting circuits are indeed great a significant source of hardware overhead to the whole system.

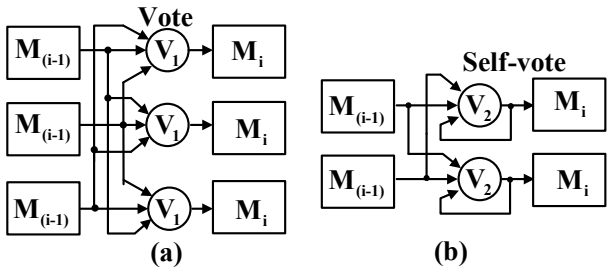


Figure 2.2: Multi-stage implementation: (a) TMR [55], (b) DMR [62].

2.2 MRF-based Two-Stage Design

Since TMR only tolerates one erroneous module [13], we proposed a real DMR, called Complementary Dual Modular Redundancy (CDMR) that tolerates the same single error condition. The single error condition refers to the condition where errors only affect one module at a time as the inputs to the voting circuit. The schematic diagram is shown in Figure 2.3. The new designed CDMR scheme not only saves large area overhead but also improves the reliability of the voting circuit. In Figure 2.3, two modules are separated into two paths with two stages. In the first stage, one module is connected to the structure A outputting a stable logic "1"; the other is connected to the structure B outputting a stable logic "0". The second stage is a feedback-based structure C which combines the stable logic "1" and stable logic "0". Structures A, B and C together form the voting circuit used in CDMR.

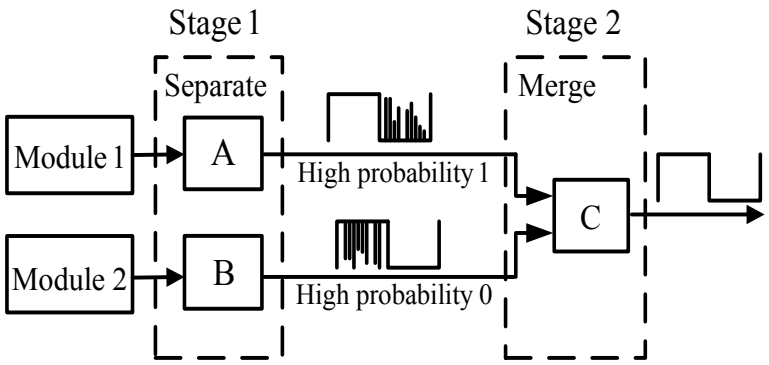


Figure 2.3: CDMR design.

In the idea of DMR shown in Figure 2.3, two identical modules are used. In the real application, modules are not replicated directly. We need to replace one of the identical modules with an inverting one to apply the MRF theory onto the

two-stage feedback voter. In Chapter 1, we have demonstrated that the MRF-based circuit design can lower the energy of the correct states to thus encourage the circuit to stay in correct states and thus tolerate errors. The 1 Stage circuit in Figure 2.3 is a NAND-NAND-based feedback structure. The design is inspired by the following theorem.

If the function of a module is an *n*-bit-input and one-bit-output function M , we can regard it as a clique, and $y = y_{out} + \sigma(\text{Noise})$ is the real output as the sum of the idea output y_{out} and a random noise source. Assume that the clique energy is $U(X_{in}, y)$, where $X_{in} = \{x_1, x_2, ..., x_n\}$ is a set of input signals and y_{out} is the ideal output of the logic function M.

Table 2.2: Energy Truth Table for M

y_{out}	y	State	Clique Energy $U(X_{in}, y)$
M(X_{in})=0	0	Valid	-1
M(X_{in})=0	1	Invalid	0
M(X_{in})=1	0	Invalid	0
M(X_{in})=1	1	Valid	-1

Theorem: As $y_{out} = \mathrm{M}(X_{in})$ represents the simplest form of the Kanaugh map simplification (canonical sum of minterms) for the module. If there exists noise，the clique energy can be deduced as

$$U(X_{in}, y) = -\mathrm{M}(X_{in}) \cdot y - \overline{\mathrm{M}(X_{in})} \cdot \bar{y} \tag{2.3}$$

Proof: Based on the conclusions in [31, 32, 37], when the clique energy of the valid states is lower than that of the invalid states, the MRF-based design is more likely to keep the outputs unaffected by the interference of soft errors as a random noise source. The goal of the MRF-based design is to make the ideal input y_{out} be

equal to the real output y, thus we have $y_{out} = \mathrm{M}(X_{in}) = y$. The energy truth table of M is built according to the equal relationship between y_{out} and y as shown in Table 2.2. Only when the clique energy is $U(X_{in}, y) = -\mathrm{M}(X_{in}) \cdot y - \overline{\mathrm{M}(X_{in})} \cdot \bar{y}$, can the clique energy of valid states have the lower energy "-1".

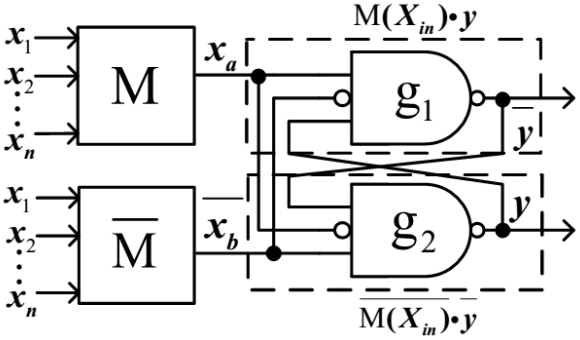


Figure 2.4: The proposed first-stage structure.

We extend the clique energy function Equation 2.3 to Equation 2.4:

$$U(X_{in}, y) = -\mathrm{M}(X_{in}) \cdot \mathrm{M}(X_{in}) \cdot y - \overline{\mathrm{M}(X_{in})} \cdot \overline{\mathrm{M}(X_{in})} \cdot \bar{y} \tag{2.4}$$

In this way, we can strengthen the influence of correct signals. In Figure 2.4, the NAND-NAND structure is the direct mapping of the clique energy function Equation 2.4. It can help the first stage tolerate soft errors. x_a and $\overline{x_b}$ correspond to the outputs of $\mathrm{M}(X_{in})$ and $\overline{\mathrm{M}(X_{in})}$ with the interference of noise. As the direct mapping of Equation 2.4, the feedback structure satisfies the clique-energy requirement. The valid states are all lower than the incorrect states and this constraint helps the structure stay in the correct states. When noise is injected in the circuits, g_1 in Figure 2.4 can tolerate a noisy "0" as an input signal since a NAND gate is more likely to output correct "1" when the inputs are {00,01,10}.

Proof: p_e ($0 \le p_e \le 0.5$) denotes as the probability of an incorrect input signal under

the effect of noise; the conditional correct probability of output signal y is assumed to be $p(y|x_1x_2)$ when the inputs of a NAND gate are x_1 and x_2:

$$p(1|00) = 1 - p_e^{\,2} \geq p(1|01) = p(1|10) = 1 - p_e(1-p_e) \geq p(0|11) = (1-p_e)^2 \qquad (2.5)$$

However, the first stage itself cannot tolerate all soft errors. It requires the help of the second stage to improve the performance of error tolerance. Figure 2.5 illustrates the detailed structure of the proposed voting circuit. The outputs of the modules M and $\overline{\mathrm{M}}$ are assumed to be $\{x_a, \overline{x_b}\}$. When their ideal values are {0, 1}, they can be corrupted to be {1, 1} as an error occurs in one module. In this situation, g_1 can output a correct bit "1" since the value "1" of $\overline{x_b}$ is inverted by an inverter, while g_2 generates an incorrect bit "1". Fortunately, the incorrect bit "1" will be corrected by the structure in the second stage shown in Figure 2.5, because Stage 2 latches the previous correct bits. Thus the proposed voting circuit can tolerate a transient "1" signal from M. A symmetrical argument shows that a transient "0" from $\overline{\mathrm{M}}$ will also be tolerated.

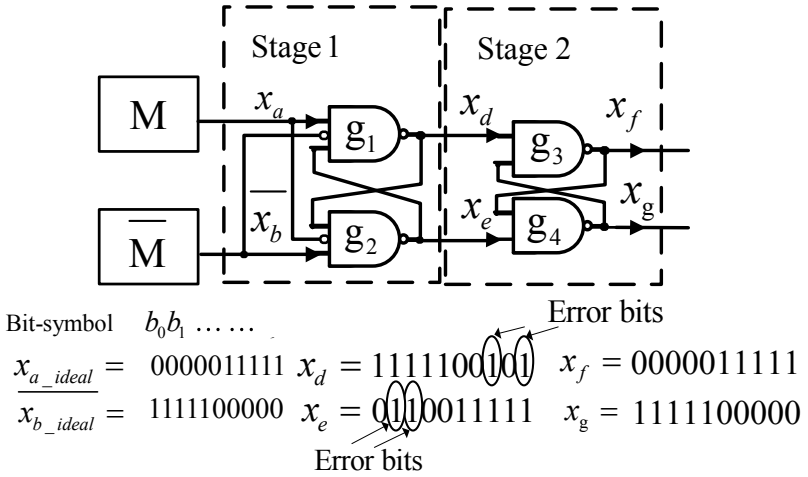


Figure 2.5: The proposed two-stage dual feedback structure.

One of the drawbacks of TMR is the vulnerability of the voting circuit itself. If errors occur inside the voting circuit, the whole system may lose the ability to tolerate errors. The following content will discuss what happens if errors occur inside the proposed voting circuits. Assume that there is an ideal input bit stream x_a (x_a= x_b) generated by modules M and $\overline{\text{M}}$. The values of x_a is {$b_0 \sim b_4$=0, $b_5 \sim b_9$=1}. Bit b_7, b_9 of x_d and b_1, b_2 of x_e are flipped to the opposite levels which are marked by ellipses in Figure 2.5. At the same time, their corresponding bits in the other branch are correct without upsets. The upsets in x_7, x_9, x_1 and x_2 make the structure in the second stage work in a latching mode shown in Table 2.3. Gate g_3 and g_4 hold the previous correct bits to protect the final outputs.

Table 2.3: States of g3-g4 Feedback

x_d		x_e		State	g3-g4
High 0 for x_a	1	0		correct	pass
	1	1		incorrect	hold
	0	1	High 1 for x_b	correct	pass
	1	1		incorrect	hold

2.3 Simulation and Discussion

2.3.1 Voter Behavior Simulation

A double-exponential current source was used to investigate the real behavior when a particle hits b_7, b_9 of x_d and b_1, b_2 of x_e in the outputs of the Stage 1 [14]:

$$I(t) = \frac{Q_{total}}{\tau_f - \tau_r}(e^{-t/\tau_f} - e^{-t/\tau_r}) \tag{2.6}$$

where Q_{total} is the total energy of the particle strike; τ_r and τ_f correspond to the rising time constant and the falling time constant. In general, τ_r and τ_f are assumed

to be 50 ps and 164 ps for a 65-nm process, and Q_{total} is set to 70 fC [85]. The supply voltage was set to 0.25 with the 65-nm Berkley CMOS library.

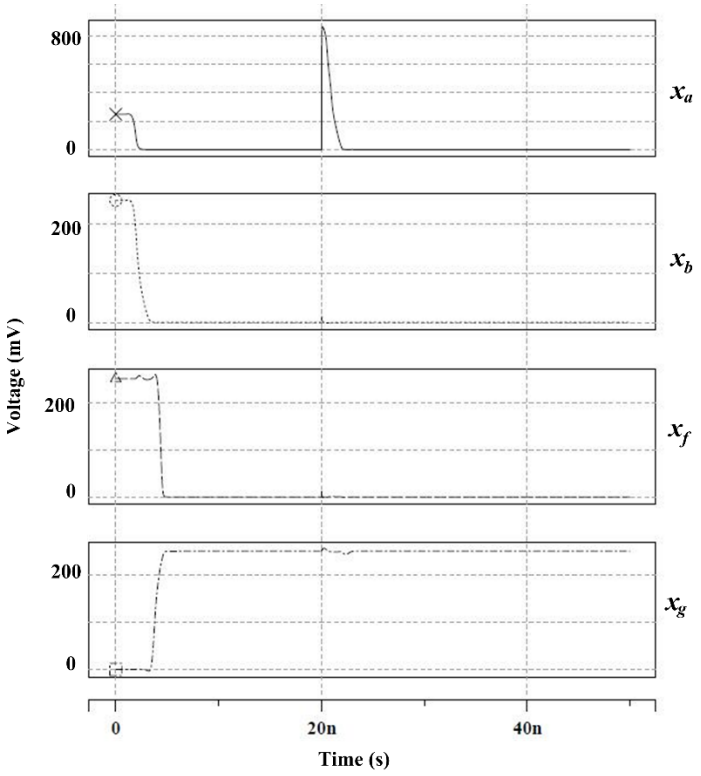


Figure 2.6: A particle disturbs x_a when $\{ x_a , x_b \}=\{0, 0\}$.

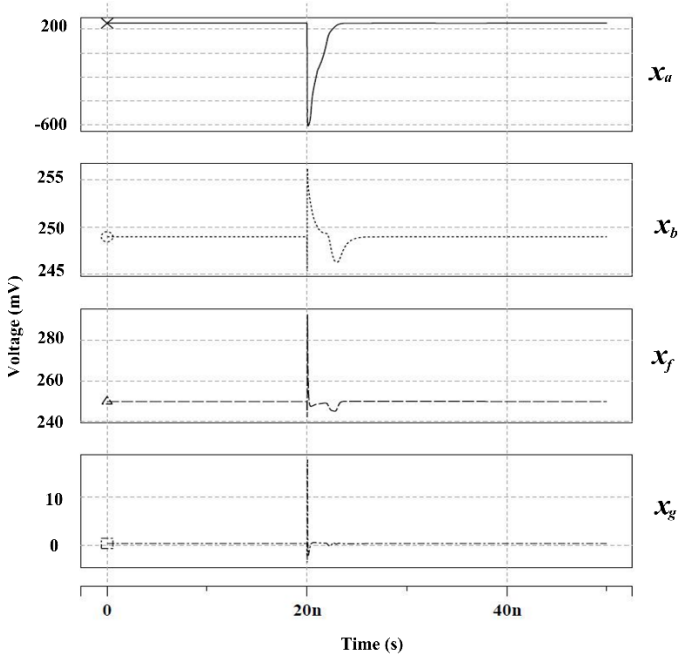


Figure 2.7: A particle disturbs x_a when $\{ x_a , x_b \}=\{1, 1\}$.

Figure 2.6 and Figure 2.7 show the cases when a particle disturbs the input x_a. In Figure 2.6, the outputs x_f and x_g have the correct values "0" and "1", respectively. The soft error in x_a is tolerated by the first stage. Although in Figure

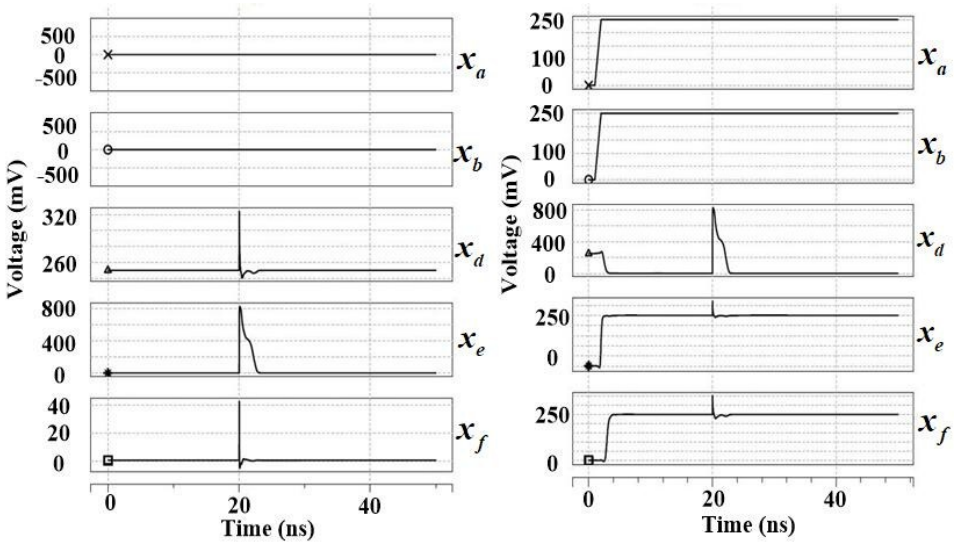


Figure 2.8: The simulation result of the intermediate propagation injected by a soft error in our proposed structure.

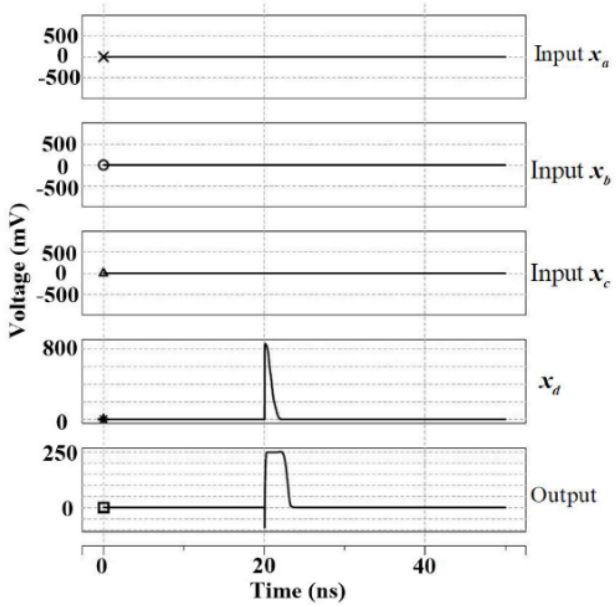


Figure 2.9: The simulation result of the intermediate propagation injected by a soft error in a TMR [55] voter.

2.7 there are sharp peaks in x_f and x_g, these peaks won't change the logic of the output signals as they don't last too long. Therefore, the proposed voting circuit can tolerate soft errors when one of the inputs is attacked.

Next, we would like to know what happen when a soft error corrupts the intermediate signals as they propagate through the voting circuit. Since x_d or x_e is robust in bit "1" as the output of a NAND gate demonstrated in Section 2.2, errors are more likely to occur in the weak bit "0" in x_d or x_e. If a particle charge is injected in the weak bit "0" in x_d or x_e, a sharp peak occurs in the final output x_f in Figure 2.8. When the same particle affects one of the three inner branches in the TMR voting circuit, there exists a short pulse at the output of a TMR voting circuit, as shown in Figure 2.9. Comparing the sharp peak in the proposed design with the short pulse in the TMR design, the upset in the proposed voting circuit is less harmless because it is too short to be sampled as errors. However, the pulse at the output of the TMR voting circuit can indeed cause many errors since it can be sampled many times because of the short duration in the signal. The results correspond to what we analyze from Figure 2.5. If errors won't occur at the two branches at the same time, then the final outputs of the system can be error-free. Therefore, the proposed voting design can help the whole system work correctly as long as the two inner signals are not affected by soft errors at the same time.

The above simulations are all based on the assumption that soft errors only occur in one of the propagation paths (one of the inputs x_a and x_b, or one of the inner propagation paths x_d and x_e) denoted as the single error case. A special case of the multiple-error condition occurs when one error appears on one of the M or

$\overline{M}$ modules and one in the voting system, as long as the error in the voting circuit happens to correct the erroneous output from the first stage caused by the error in one of the modules in Figure 2.10(a) (as the old saying goes, two negatives make a positive), or there is a time interval between the two errors in Figure 2.10(b), the

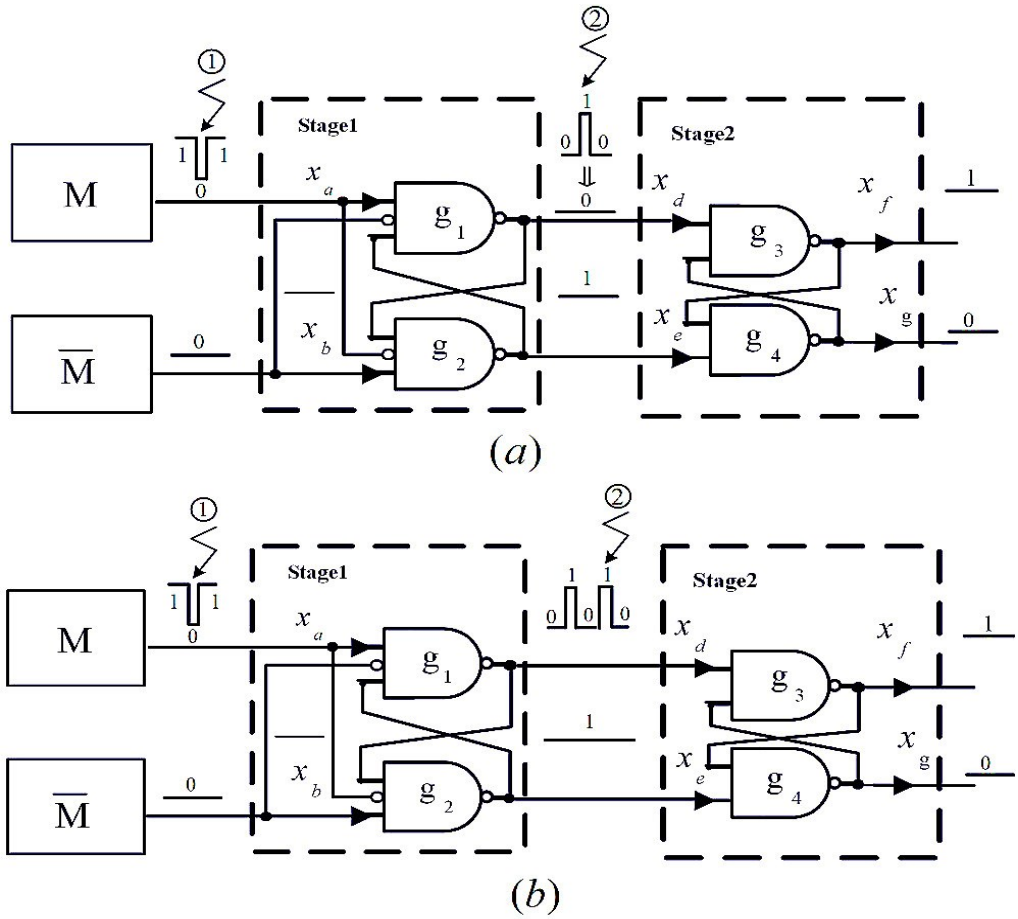


Figure 2.10: The special multiple-error cases.

final outputs are still correct. However, the voting circuit cannot tolerate errors which occur almost at the same time (for a time interval is less than 0.8 ns) on both propagation paths as the normal multiple error case as shown in Figure 2.11 and Figure 2.12. If there is a delay unit in one of the input branches, the proposed structure may tolerate the multiple-error cases where errors occur at the outputs of the two modules at the same time. Figure 2.13 shows that if the delay unit can

separate the two soft errors with at least a 0.8-ns time interval, the second stage can still help the circuit correct the soft errors.

We used the model in Equation 2.6 to describe the detailed behavior of the proposed structure when one of the propagation paths is impaired by a soft error. In the following content, we used random Gaussian noise injected to each input signals with respect to the period of input signals to mimic a realistic distribution of noise for the combination of different cases including the single error case and the multiple error case.

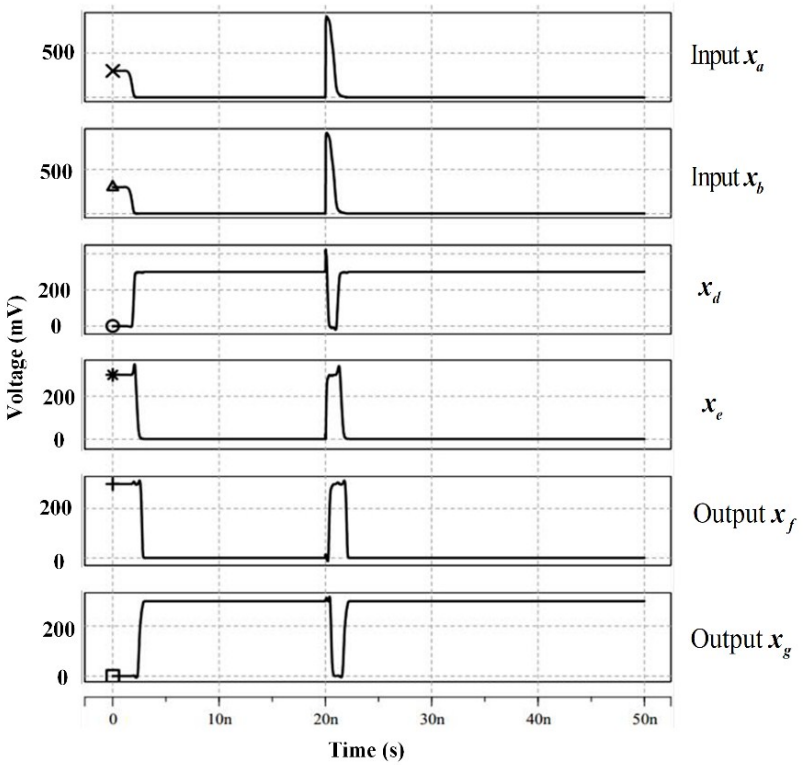


Figure 2.11: The double error case (x_a = x_b=0): time interval= 0 ns.

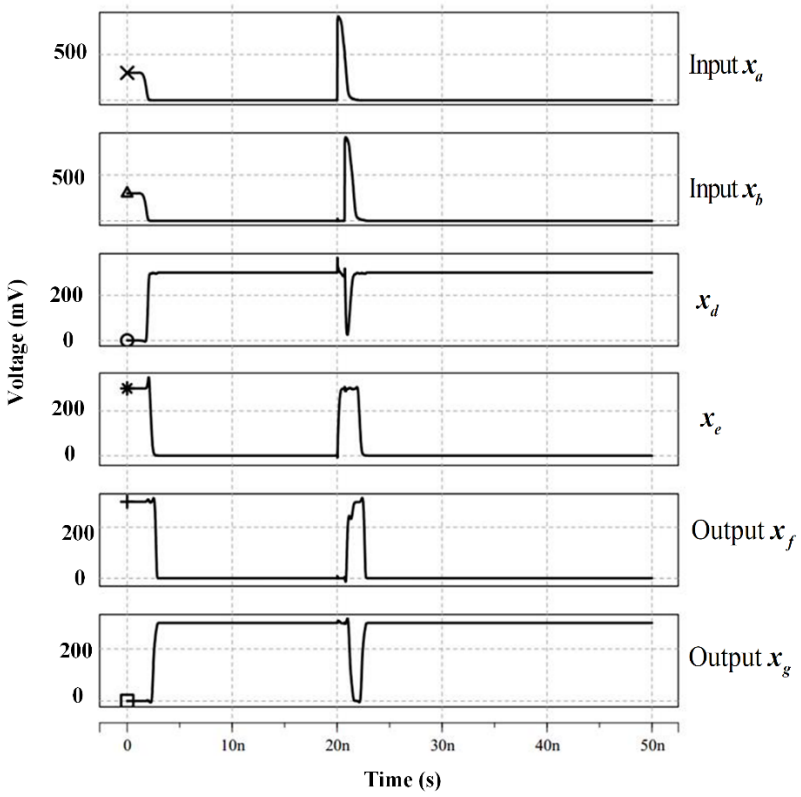


Figure 2.12: The double error case ($x_a = x_b$=0): time interval= 0.7 ns.

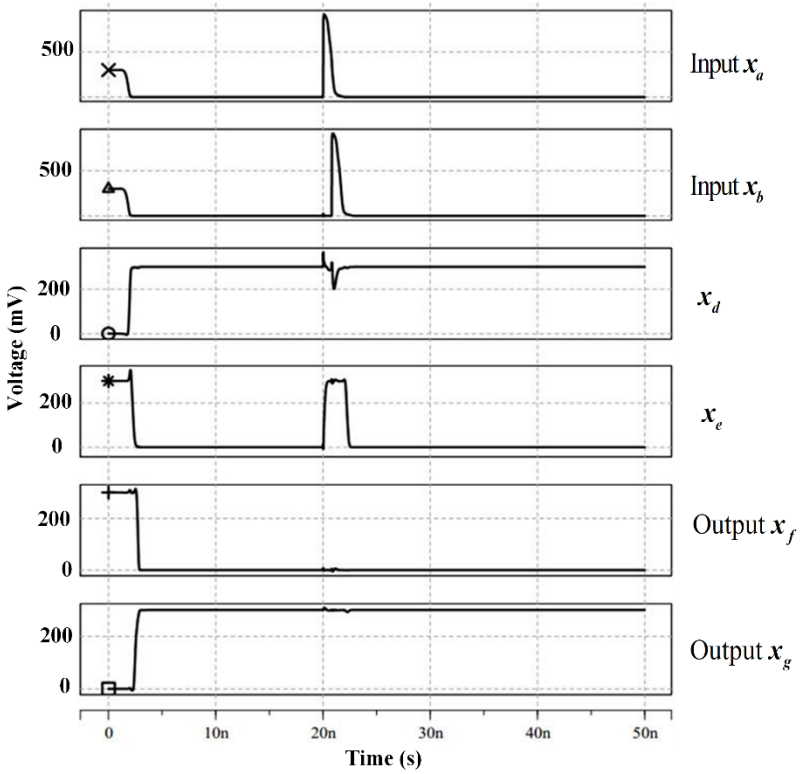


Figure 2.13: The double error case ($x_a = x_b$=0): time interval= 0.8 ns.

2.3.2 Error Rate Simulation

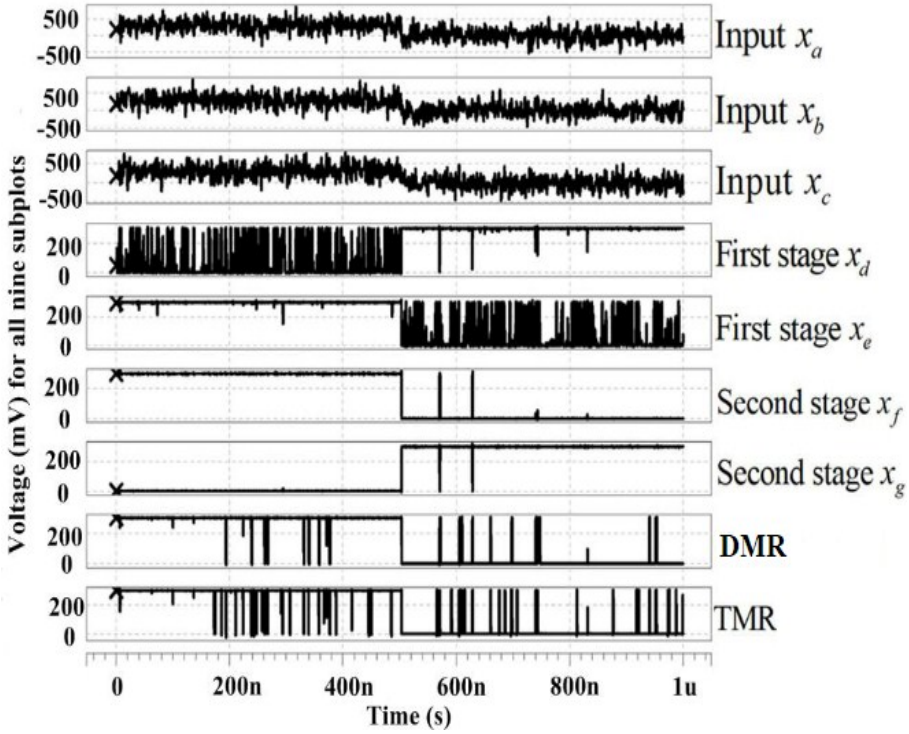


Figure 2.14: The simulation results: DMR [62] and TMR [55].

We used random Gaussian noise added to each input signal with respect to the period of input signals to mimic the distribution of soft errors. Simulations were all processed in 65-nm technology with a supply voltage 0.25 V. Figure 2.14 shows the output waveforms of different voting designs when an independent Gaussian noise with zero mean and 170-mV standard deviation was injected in the inputs. The first three curves in Figure 2.14 are three input signals with noise. The next four curves are the inner signals and output signals in the proposed design. The remaining two curves are the output signals of DMR voting circuits and TMR voting circuits. In the curve of the first stage x_d, bit "1" is clean without a lot of upsets. It shows that the first stage can tolerate a noisy "0" in x_a in Figure 2.5. Similarly, the fifth curve shows the first stage can tolerate a noisy "1" in x_b. Stage

2 of our design successfully combines the robust "1" and robust "0" as the second stage curves are clean in both high levels and low levels. The quality of the final output signals is the same after the processing of the two-stage voting circuit. However, there are a lot of upsets in the final outputs of DMR [62] and TMR [55] voting circuits.

For the following error rate simulation, we used transient analysis to simulate the output voltage under the interference of noise. Thermal noise was injected while setting the temperature to 50 °C. The sampling process was operated by HSPICE automatically using the .tran command. From 1 ns to 1000 ns, for 1-MHz input signals, it sampled about 30000 times (sampling rate≈30 GHz). Error rates were calculated in MATLAB from the .tr0 file. SNR was used to measure the intensity of the noise by $SNR = 20\log_{10}\frac{A_{Signal}}{A_{Noise}}$, where A_{signal} is the power supply voltage and A_{noise} is the standard deviation of the noise. The simulated designs are the proposed voter, self-vote [62], DCVS [86], DCVS-MRF [42], TMR [55], MRF-MS [38], MRF-efficient [13], Robust c-element [63] and Mux-vote [64].

Figure 2.15 shows the simulated output error rates under the effect of independent Gaussian noise. Figure 2.16, Figure 2.17 and Figure 2.18 present the error rates with the inteference of correlated Gaussian noise (weak correlation: ρ=0.1; medium correlation: ρ=0.3; strong correlation: ρ=0.5). Compared to TMR [55], the average error rate of the proposed design is reduced by 59.6% (68.2% when ρ=0.1; 61% when ρ=0.3; 49.5% when ρ=0.5) with 20% area saving and 8.33% delay reduction. When compared to the self-voting scheme in [62], the

reduction decreases to 30.2% (41.6% when ρ=0.1; 31% when ρ=0.3; 18% when ρ=0.5) with a reduction of 20% area and 15% delay. This indicates that the voting circuit tolerates more errors if the Gaussian noise sources are less correlated. The proposed voting citcuit has the lowest error rate regardless if the Gaussian noise sources are correlated or not.

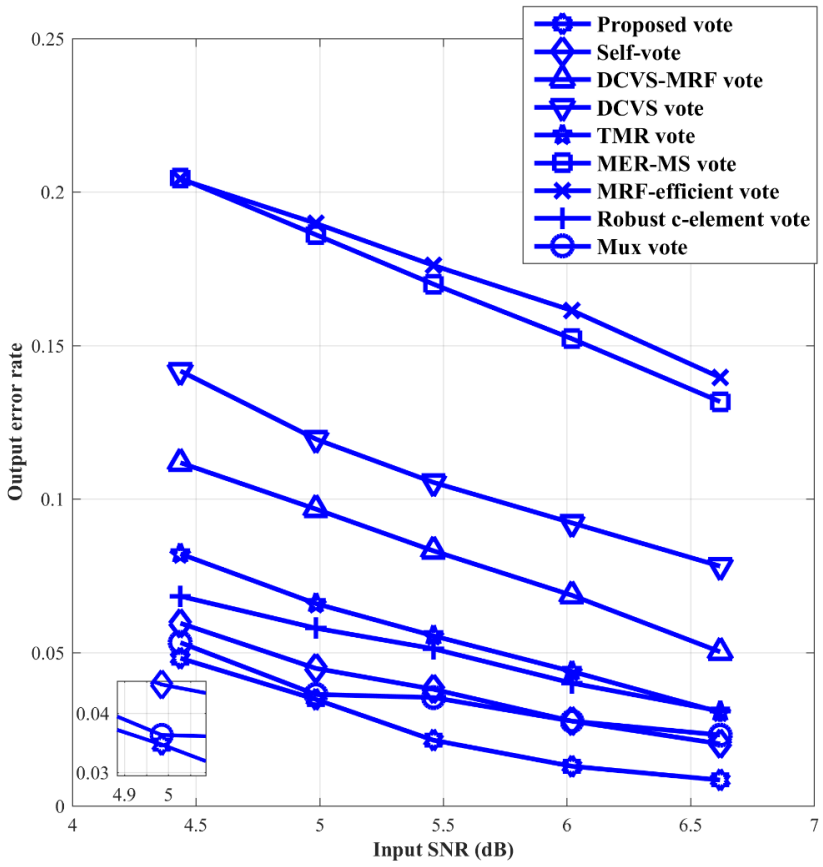


Figure 2.15: The simulation results under independent Gaussian noise.

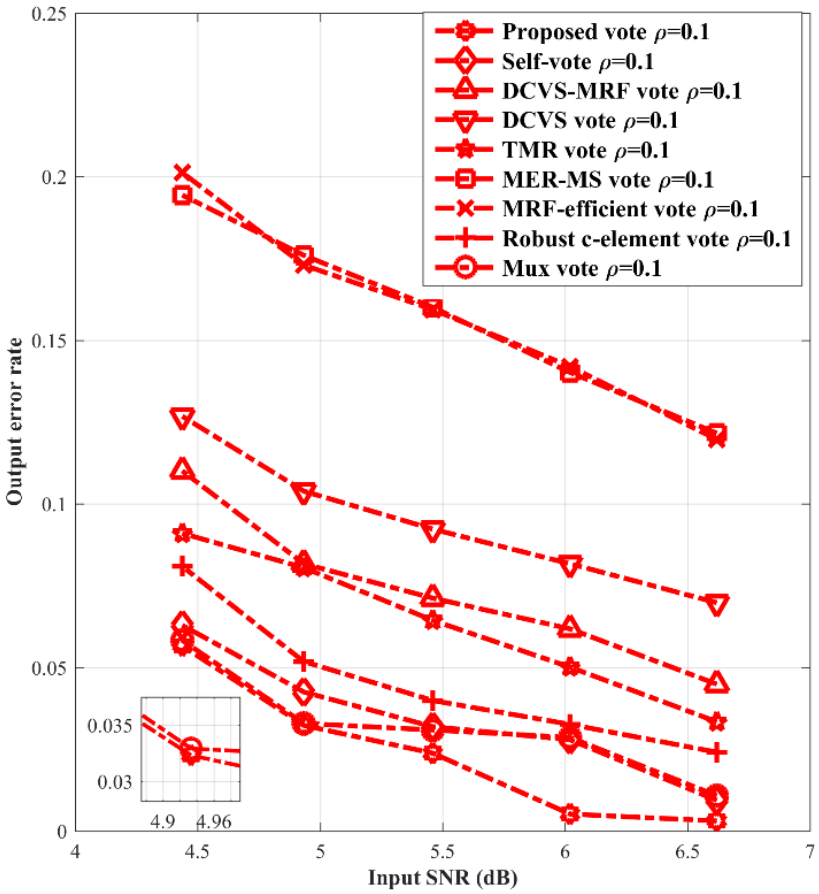


Figure 2.16: The simulation results under correlated Gaussian noise ρ=0.1.

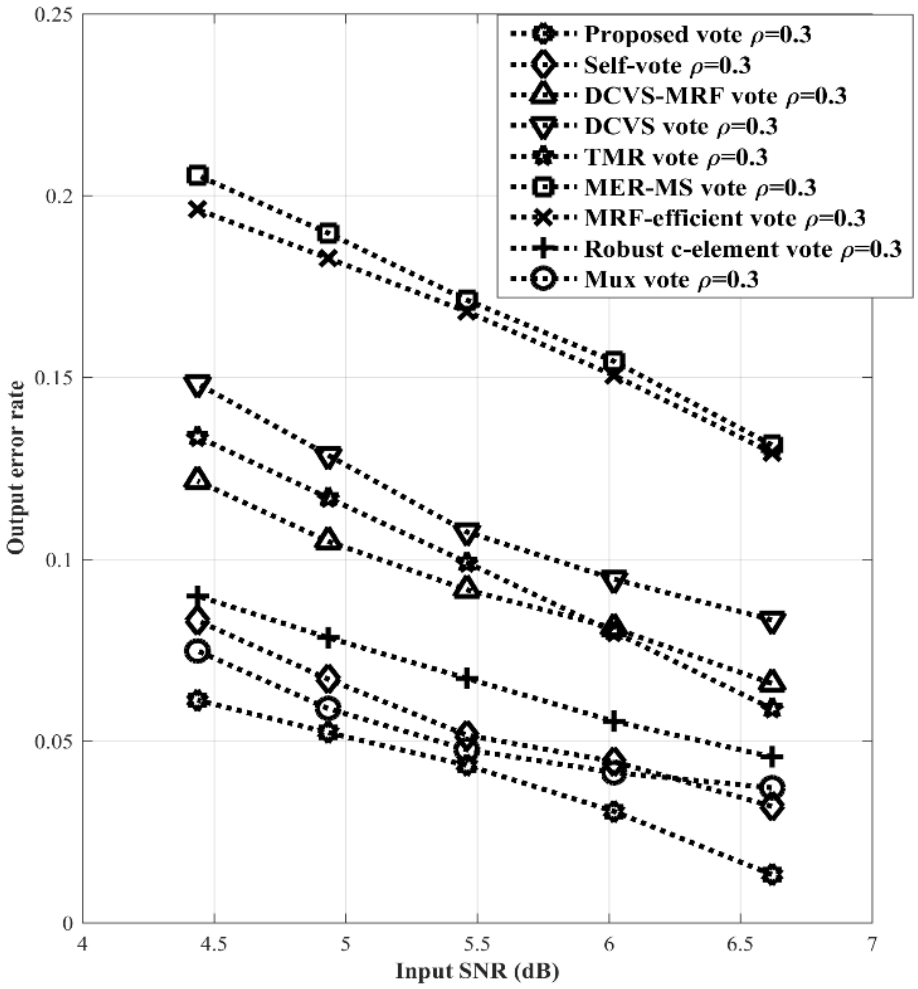


Figure 2.17: The simulation results under correlated Gaussian noise ρ=0.3.

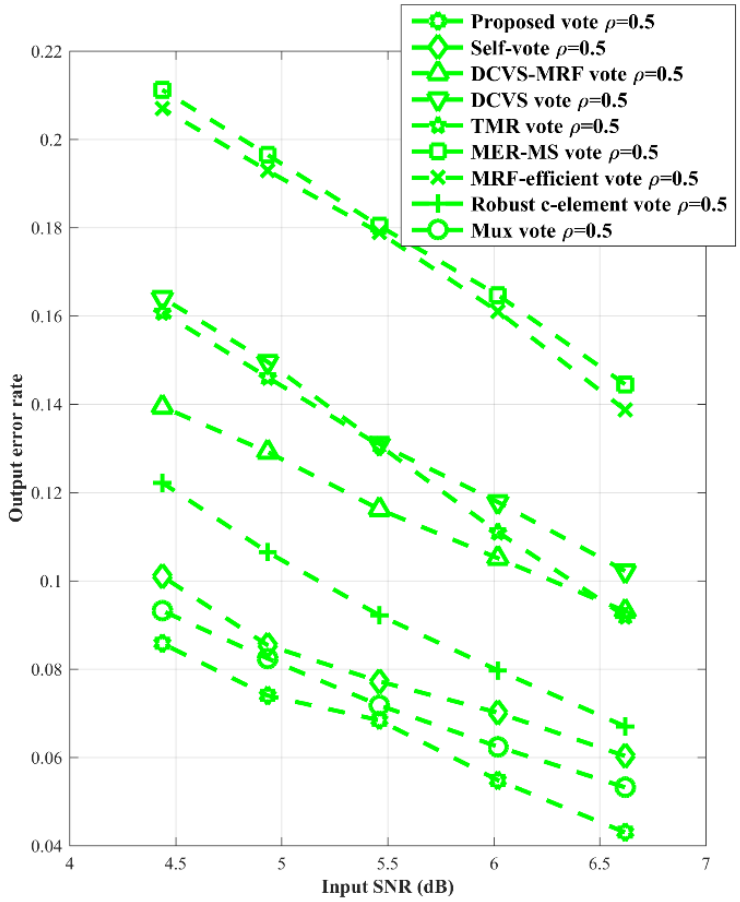


Figure 2.18: The simulation results with correlated Gaussian noise ρ=0.5.

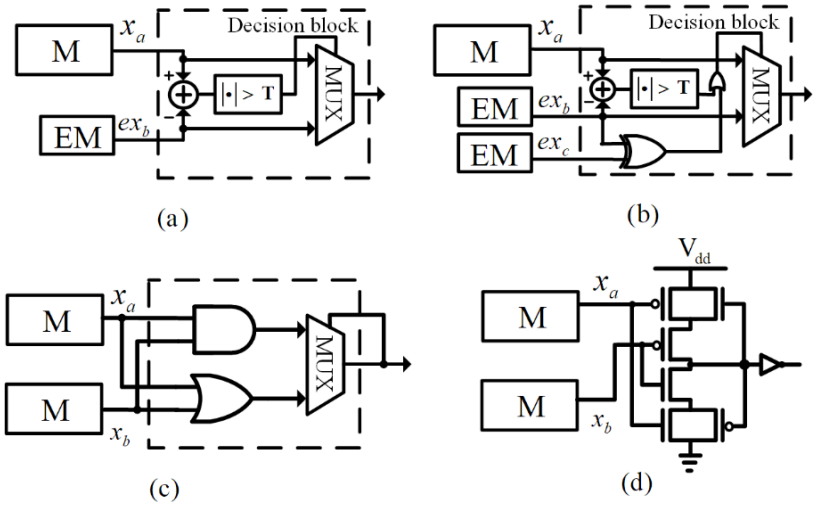


Figure 2.19: Different error tolerant systems: a) ANT [65] b) ASET [66] c) DMR with MUX-vote [64] d) DMR with robust c-element vote [63].

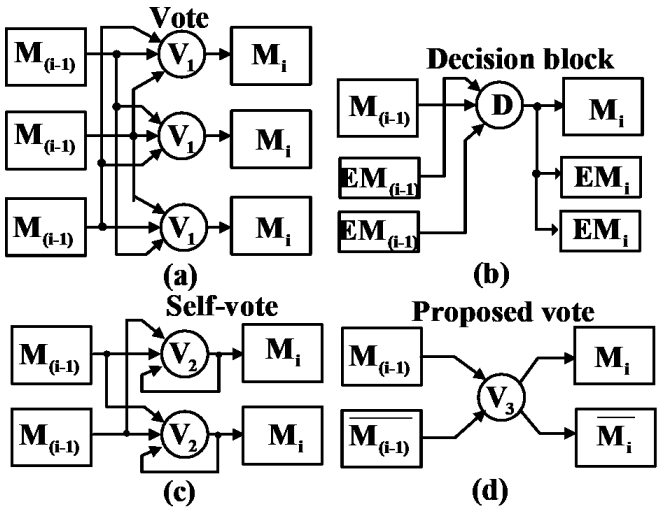


Figure 2.20: Multi-stage insertion of voters: a) TMR [55] b) FGSET [67] c) DMR [62] d) the proposed design.

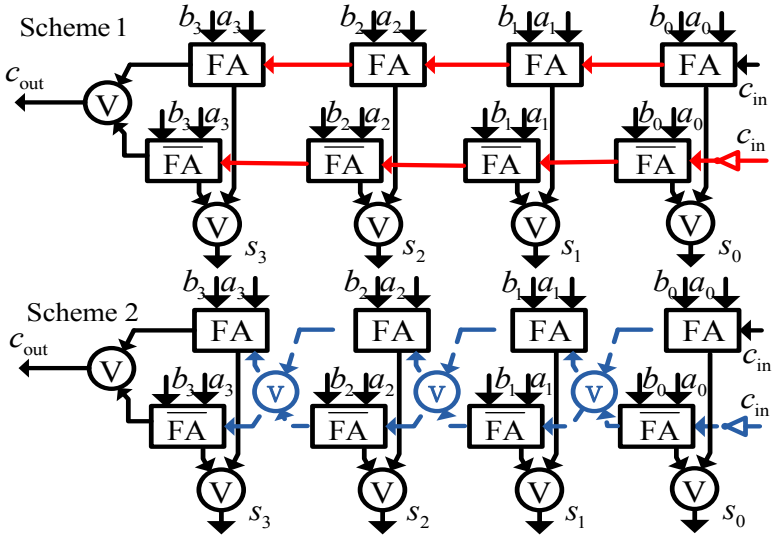
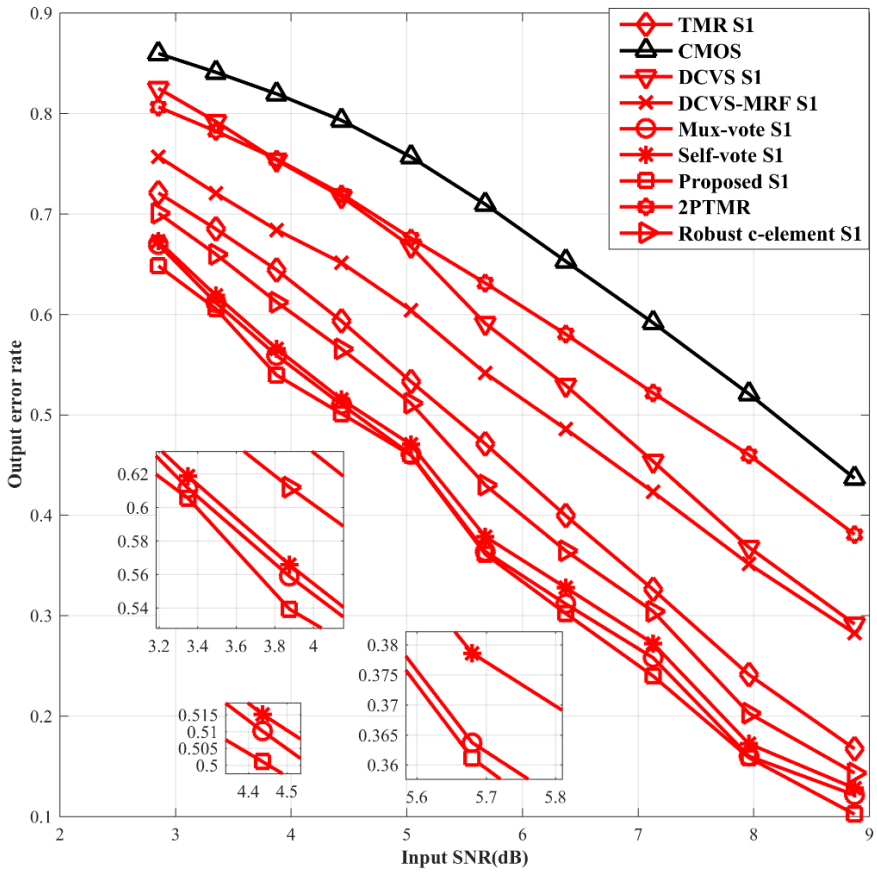


Figure 2.21: The proposed schemes for the ripple-carry adder structure.

We also simulated different designs as a whole system applied with different schemes. Figure 2.19 shows different error-tolerant systems, including ANT [65], ASET [66], DMR with MUX-vote [64] and DMR with robust c-element vote [63]. There are two schemes for applying the error-tolerant designs: a single-stage scheme and a multi-stage scheme. The single-stage scheme inserts a voting circuit

at the output of the whole system. The multi-stage scheme inserts a voting circuit between each stage (small blocks) of the whole system as shown in Figure 2.20.



*CMOS means a non-redundant design

Figure 2.22: Simulation results for 65-nm RCAs with Scheme 1.

Let us use a 4-bit Ripple Carry Adder (RCA) as the case study. There are two schemes for using the proposed CDMR shown in Figure 2.21. One is called Scheme 1, which uses the voting circuits at the final outputs; the other is called Scheme 2, which uses the voting circuits between each two full adders. Schemes 1 and 2 were simulated for a supply voltage of 0.25 V with different input SNR conditions assuming a 65-nm process. The results of Scheme 1 and Scheme 2 are

shown in the Figure 2.22 and Figure 2.23, respectively. The results indicate that the proposed voting designs help the system achieve at least 3.7 % and 12.5% improvement in error-tolerance compared to the Scheme 1 and Scheme 2 adder designs, with 1.8% timing saving and 29.4% area saving compared to TMR [55]. The proposed structure also has 3.7% less delay and 7.6% less area than the DMR [62] as shown in Table 2.4 and Table 2.5.

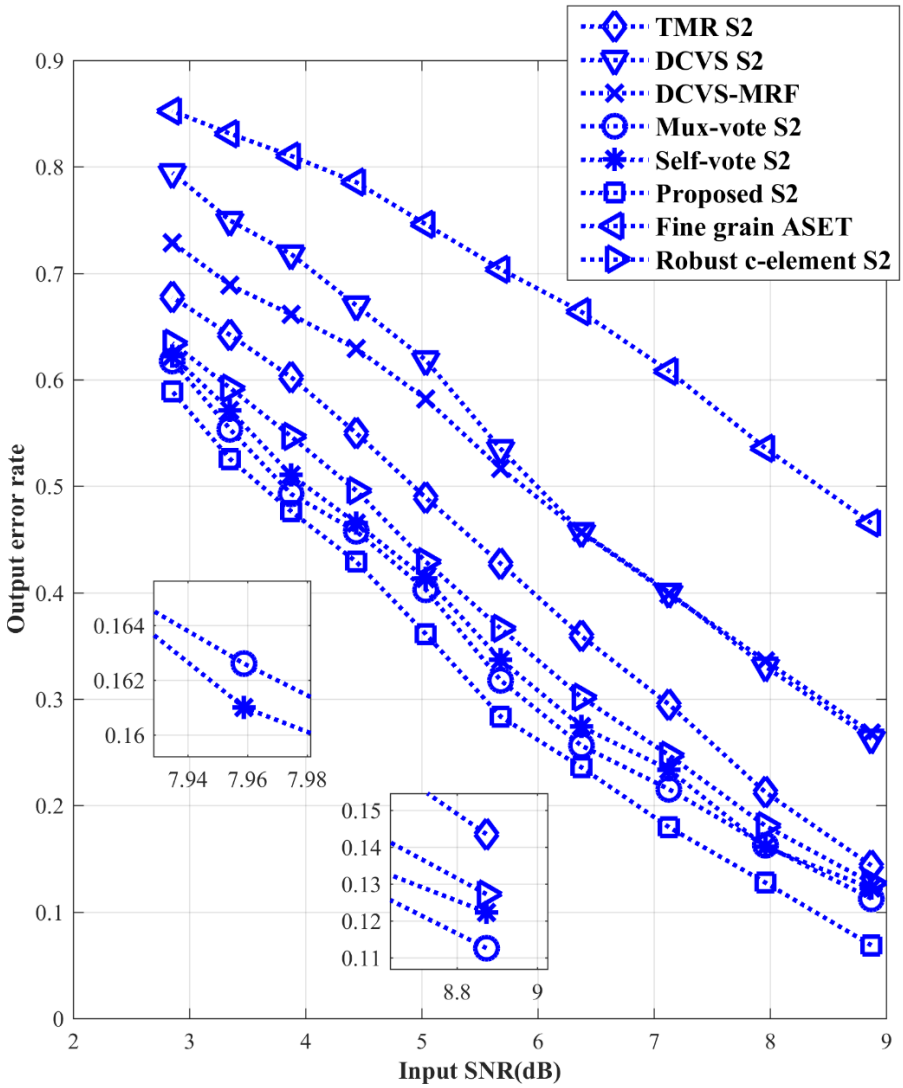


Figure 2.23: Simulation results for 65-nm RCAs with Scheme 2.

Table 2.4: Comparison of Area, Delay and Power consumption for Scheme 1

Design	CMOS	TMR [55]	DMR [62]	PTMR [59]	MUX [64]	This work
Process	TSMC 65 nm					
Scheme	S1					
Area (μm^2)	36	153	117	99	131	108
Delay (ns)	0.42	0.54	0.55	0.54	0.59	0.53
Vdd_{min} (V)	0.5	0.29	0.27	0.28	0.26	0.25
Power@Vdd_{min} (μW)	13.6	6.5	3.5	3.7	4.8	2.4
Error rate Improved (%)	46.88	21.04	5.89	41.39	3.71	-

Table 2.5: Comparison of Area, Delay and Power consumption for Scheme 2

	CMOS	TMR [55]	DMR [62]	FGSET [67]	MUX [64]	This work
Process	TSMC 65 nm					
Scheme	S2					
Area (μm^2)	36	324	216	141	279	150
Delay (ns)	0.42	1.20	1.21	1.57	1.81	1.01
Vdd_{min} (V)	0.45	0.28	0.26	0.29	0.255	0.25
Power@Vdd_{min} (μW)	10.4	6.9	3.7	6.5	6.8	2.6
Error rate Improved (%)	55	29.96	15.81	56.20	12.54	-

2.4 Conclusion

The novel CDMR method is flexible enough to apply to many digital computing systems. It can be applied to each stage or to the end of the whole system as a single stage. The method is inspired from the MRF theory, the inherent error tolerance of the logic gate and the traditional redundancy technique. It separates two modules into a robust “1” module and a robust “0” module and combines them together by the feedback-based voting circuit. One of the modules should be inverted to form the MRF network for the voting circuit. As a DMR technique, it saves one redundant module compared with previous TMR and

PTMR approaches. Meanwhile, the proposed voting circuit is more reliable than the previous designs. It can tolerate errors which occur in only one of two complementary propagation chains at the same time. Compared with TMR design, the proposed voting circuit reduces the area by at least 20% and propagation delay by at least 8.33%. It also achieves at least a 26% reduction in error rates at an ultra-low supply voltage 0.25V compared to other voter designs. When it is applied to a 4-bit RCA, the proposed CDMR method reduces by at least 12.5% error rate with at least 30% in area saving, compared with the previous self-voting DMR approaches if the voters are inserted in between every two full adders.

Since the proposed CDMRF method uses redundant modules, it complicates the testability of the whole system. The challenge is also strengthened by the latches in Stages 1 and 2 for some automatic test pattern generation (ATPG) algorithms. The simplest solution is the use of two bypass multiplexers. These multiplexers should be inserted to the end of Stage 2 for each output. When testing the function of the latches, these multiplexers should select the output signals of the voting circuit; when testing the modules, the two multiplexers need to select the outputs of the modules M and $\overline{M}$. In other words, the multiplexer control signal should first be fixed to temporarily omit the voting circuit for the running of the related ATPG algorithms. These generated test vectors can then test modules M and $\overline{M}$. Next, the multiplexer control signal will select the outputs of the voting circuits to test the whole DMR system. Therefore, the testability of

the design could be improved by the insertion of multiplexers at the cost of the related increased area.

Chapter 3

DCT Implementation Based on MRF-Stochastic Logic

3.1 DCT Algorithm

The Discrete Cosine Transform (DCT) is a common compression algorithm used today for images and video [87]. N. Ahmed et al. first proposed the algorithm in 1974 [88]. It has been widely adopted as the core algorithm for video coding standards such as H.263 [89], H.264 [90], MPEG-1[91] and MPEG-2[92], and image coding standards such as JPEG [93] and JPEG2000 [94]. It deals with real numbers. Assume that $x(n),\ n = 0,1,...,N\text{-}1$ is a sequence of real numbers. A DCT of the real numbers is calculated by the following expression [95]:

$$X(k) = e(k)\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1} x(n)\cdot\cos[\frac{k\pi(2n+1)}{2N}],\ k = 0,1,...,N-1 \tag{3.1}$$

where $e(k) = \begin{cases} \frac{1}{\sqrt{2}} & k = 0 \\ 1 & k \neq 0 \end{cases}$. The matrix form of its DCT can be defined as

$$X = T\cdot x \tag{3.2}$$

where

$$T = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{2N} & \cos\frac{3\pi}{2N} & \cos\frac{5\pi}{2N} & \cdots & \cos\frac{(2N-1)\pi}{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \cos\frac{(N-2)\pi}{2N} & \cos\frac{3(N-2)\pi}{2N} & \cos\frac{5(N-2)\pi}{2N} & \cdots & \cos\frac{(2N-1)(N-2)\pi}{2N} \\ \cos\frac{(N-1)\pi}{2N} & \cos\frac{3(N-1)\pi}{2N} & \cos\frac{5(N-1)\pi}{2N} & \cdots & \cos\frac{(2N-1)(N-1)\pi}{2N} \end{bmatrix}$$

In this thesis, we only consider the most common 8-point DCT, that is $N = 8$, and thus T can be expressed by an 8×8 cosine matrix where $C_i = \cos\frac{i\pi}{16}, i = 1,...,7.$

$$T = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 \\ C_1 & C_3 & C_5 & C_7 & C_9 & C_{11} & C_{13} & C_{15} \\ C_2 & C_6 & C_{10} & C_{14} & C_{18} & C_{22} & C_{26} & C_{30} \\ C_3 & C_9 & C_{15} & C_{21} & C_{27} & C_{33} & C_{39} & C_{45} \\ C_4 & C_{12} & C_{20} & C_{28} & C_{36} & C_{42} & C_{50} & C_{58} \\ C_5 & C_{15} & C_{25} & C_{35} & C_{45} & C_{55} & C_{65} & C_{75} \\ C_6 & C_{18} & C_{30} & C_{42} & C_{54} & C_{66} & C_{78} & C_{90} \\ C_7 & C_{21} & C_{35} & C_{49} & C_{63} & C_{77} & C_{91} & C_{105} \end{bmatrix} \tag{3.3}$$

T can be simplified to Equation (3.4) by exploiting the properties of the trigonometric functions:

$$T = \begin{bmatrix} C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 & C_4 \\ C_1 & C_3 & C_5 & C_7 & -C_7 & -C_5 & -C_3 & -C_1 \\ C_2 & C_6 & -C_6 & -C_2 & -C_2 & -C_6 & C_6 & C_2 \\ C_3 & -C_7 & -C_1 & -C_5 & C_5 & C_1 & C_7 & -C_3 \\ C_4 & -C_4 & -C_4 & C_4 & C_4 & -C_4 & -C_4 & C_4 \\ C_5 & -C_1 & C_7 & C_3 & -C_3 & -C_7 & C_1 & -C_5 \\ C_6 & -C_2 & C_2 & -C_6 & -C_6 & C_2 & -C_2 & C_6 \\ C_7 & -C_5 & C_3 & -C_1 & C_1 & -C_3 & C_5 & -C_7 \end{bmatrix} \tag{3.4}$$

Note that the absolute values of the elements in T are symmetrical in rows.

$$\begin{cases} X(1) = M_0 C_1 + M_1 C_7 + M_2 C_3 + M_3 C_5 \\ X(3) = M_0 C_3 - M_1 C_5 - M_2 C_7 - M_3 C_1 \\ X(5) = M_0 C_5 + M_1 C_3 - M_2 C_1 + M_3 C_7 \\ X(7) = M_0 C_7 - M_1 C_1 - M_2 C_5 + M_3 C_3 \\ X(2) = M_{10} C_2 + M_{11} C_6 \\ X(6) = M_{10} C_6 - M_{11} C_2 \\ X(4) = M_{100} C_4 \\ X(0) = P_{100} C_4 \end{cases}, \tag{3.4}$$

where

$$M_0 = x_0 - x_7;\ P_0 = x_0 + x_7;$$
$$M_1 = x_3 - x_4;\ P_1 = x_3 + x_4;$$
$$M_2 = x_1 - x_6;\ P_2 = x_1 + x_6;$$
$$M_3 = x_2 - x_5;\ P_3 = x_2 + x_5;$$
$$M_{10} = P_0 - P_1;\ P_{10} = P_0 + P_1;$$
$$M_{11} = P_2 - P_3;\ P_{11} = P_2 + P_3;$$
$$M_{100} = P_{10} - P_{11};\ P_{100} = P_{10} + P_{11}.$$

Therefore, Equation (3.2) can be simplified further to the above set of 8 equations.

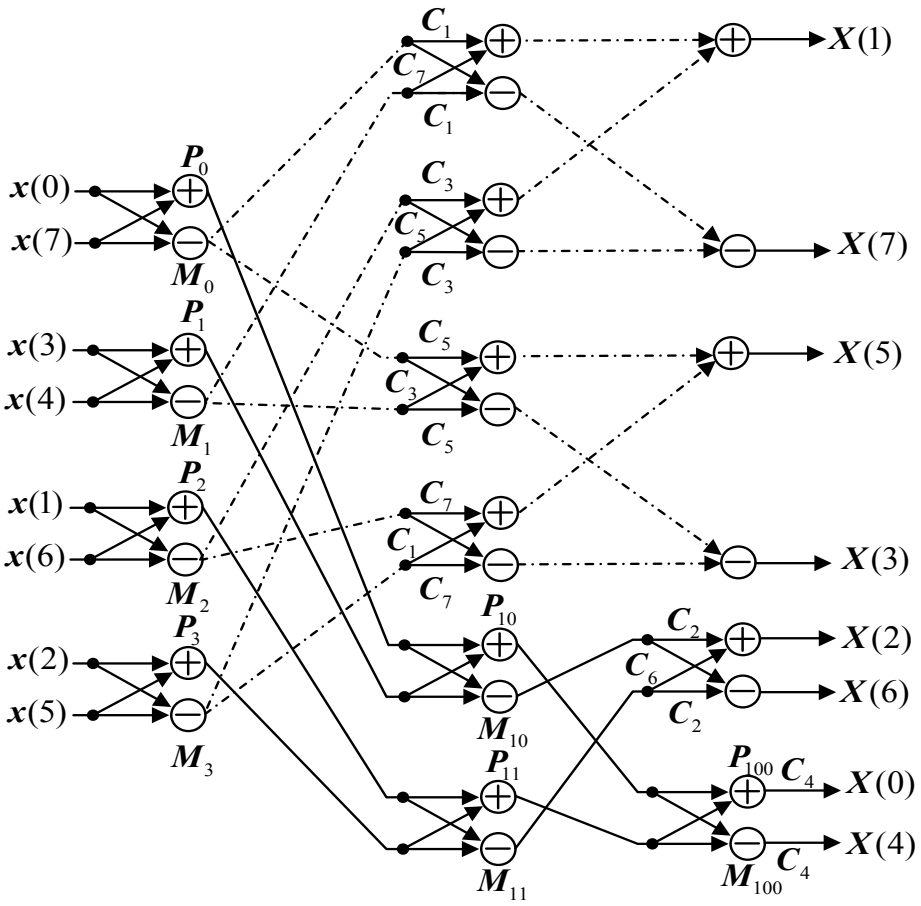


Figure 3.1: The 8-point DCT butterfly structure.

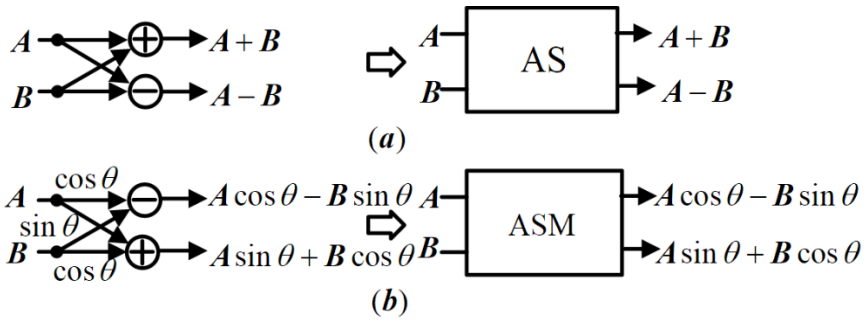


Figure 3.2: DCT computing units: (a) an AS unit (b) an ASM unit.

In this way, an 8-point DCT can be easily implemented by a hardware circuit. The schematic diagram of the structure shown in Figure 3.1 consists of adders, subtractors and multipliers arranged in a butterfly shape. Given that an adder can also perform subtraction, in total it requires 28 adders and 22 multipliers to compute the DCT function in Figure 3.1. The butterfly-shaped structures are separated into two kinds of DCT computing units, including the ADD-SUB (AS) units in Figure 3.2(a) and ADD-SUB with MUL (ASM) units in Figure 3.2(b).

3.2 Stochastic Logic

In 1956, the idea of stochastic computing (SC) was first proposed by Von Neumann [55]. The idea has contributed to different research areas such as control applications [96, 97], image processing [98, 99] and Artificial Intelligence (AI) [100, 101]. The main advantages of SC focus on low cost and low complexity as a digitized probability form of number representation [102]. In conventional binary computing, hardware cost is involved with the number of binary bits, bit length. For example, a 4-bit ripple carry adder consumes 4 full adders. This means that the number of bits determines the corresponding number of required full adders. In

modern VLSI circuit design, we want to shrink the required computing devices as much as possible. Furthermore, binary representation involves the concept of bit weight. The higher order bits are called the Most Significant Bits (MSB) and the bits starting at the lowest order are called the Least Significant Bits (LSB). If MSBs contain errors caused by noise, the corresponding value will be far from the ideal correct one. Therefore, MSBs are more vulnerable to noise. Corresponding to the above two weaknesses in binary computing, SC is a possible solution. As a low-cost alternative, arithmetic operations can be achieved by simple computing units. For example, scaled stochastic addition only requires a multiplexer (MUX) [103] and multiplication only requires an XNOR gate [104]. In addition, since stochastic numbers in SC are represented in a form of non-weighted sequences of bits called bit streams, it has the inherent capability of error tolerance especially for transients or soft errors. Meanwhile, the form of digitized probabilities affects accuracy and computation time. Longer bit streams can improve accuracy at the cost of consuming more time.

In SC, real numbers are not directly used. They should be converted into stochastic numbers that encode a sequence of probabilities. In this thesis, we only consider the conversion between the signed real domain and unsigned stochastic domain. Assume that $a \in [-k, k]$ is a real number. It can be normalized into the stochastic domain by the formula:

$$P_a = \frac{\frac{a}{k} + 1}{2} \tag{3.5}$$

We know that P_a varies from 0 to 1. For example, a=6 and k=2^n=8 (let n=3; n is the least number of the bits in the two's complement system); we have P_a=0.875. In this case, the corresponding bit stream can be "1,1,1,1,1,0,1,1". There are 0.875×k 1s and (1-0.875)×k 0s. The order of the "1s" and "0s" in the sequence is random, so other possible bit steams are "1,1,0,1,1,1,1,1", "1,1,1,1,0,1,1,1" and so on. Usually, the conversion circuit from binary numbers to stochastic numbers consists of a random number generator and a comparator, while the conversion circuit from stochastic numbers to binary numbers requires a counter to count the number of 1s [105, 106]. Since DCT computing units only need to implement addition and multiplication, the following examples are for stochastic addition and multiplication. Assume that there are two real numbers a and b ($a \in [-k_1, k_1]$; $b \in [-k_2, k_2]$); the stochastic value of the addition of a and b is shown in Equation (3.6).

$$P_{a+b} = \frac{\frac{a+b}{k_1+k_2}+1}{2} = \frac{k_1}{k_1+k_2}P_a + \frac{k_2}{k_1+k_2}P_b \tag{3.6}$$

When $k_1 = k_2$, we have

$$P_{a+b} = \frac{1}{2}(P_a + P_b) \tag{3.7}$$

Therefore, Equation (3.7) can be implemented by a multiplexer (MUX) with a fixed selection input P_s =0.5 [107]. Figure 3.3 is an example of the stochastic addition between stochastic numbers 0.5 and 0.25 (decimal numbers 0 and -4). Since $P_{a+b} \in [-2k_1, 2k_1]$, 0.375 corresponds to decimal number -4. Similarly, the stochastic multiplication of a and b is

$$P_{a\times b}=\frac{\frac{a\times b}{k_1\times k_2}+1}{2}=(1\text{-}P_a)(1\text{-}P_b)+P_a\times P_b \tag{3.8}$$

Therefore, stochastic multiplication can be implemented by an XNOR gate based on Equation (3.8) [108]. Figure 3.4 is an example of the stochastic multiplication between stochastic numbers 0.5 and 0.25 (decimal numbers 0 and -4). Since $P_{a+b} \in [\text{-}2k_1, 2k_1]$, 0.5 corresponds to decimal number 0.

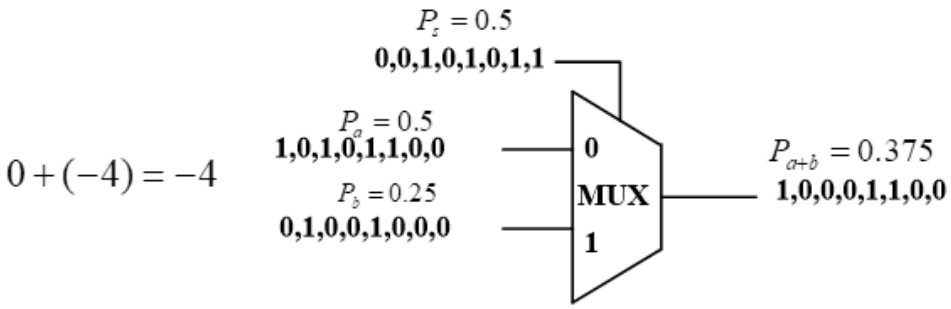


Figure 3.3: An example of stochastic addition.

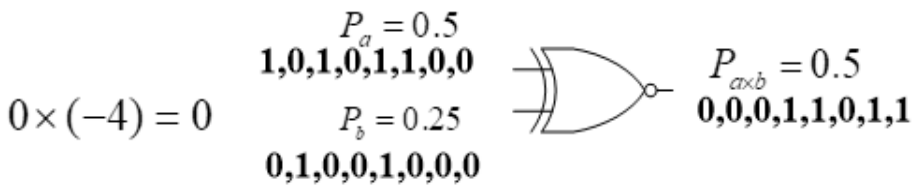


Figure 3.4: An example of stochastic multiplication.

3.3 MRF-Stochastic-based DCT Design

3.3.1 Stochastic DCT with MRF-SC Computing Units

One of the advantages of using SC is that arithmetic operations can be performed by simple digital hardware. This advantage can compensate for the

disadvantage of the MRF-based circuit designs. By combining SC with the MRF-based circuit design, we can obtain a cost-effective DCT computing circuits with high noise immunity. A stochastic adder costs at least three logic gates. Direct mapping to an MRF-based stochastic adder is not cost-effective. Therefore, we need to improve the traditional mapping method for the stochastic adder. A simple optimized way to simplify the MRF network by combining two MRF-based logic gates is shown in Figure 3.5. In other words, an MRF network can perform two kinds of logic operations.

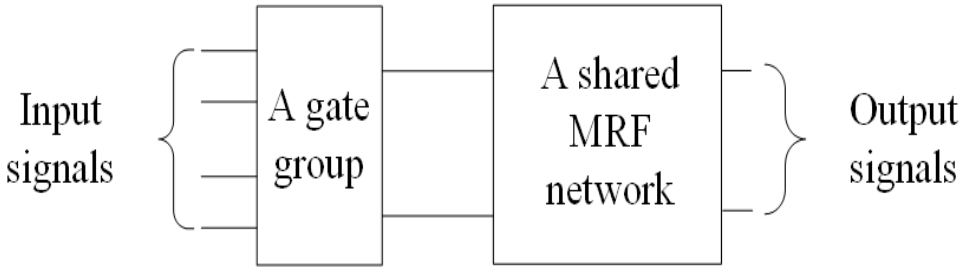


Figure 3.5: The idea of a shared MRF network.

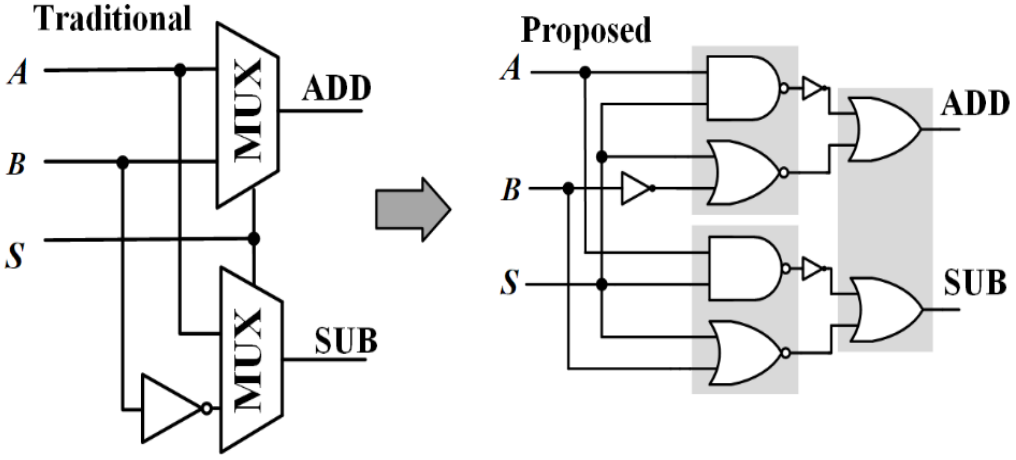


Figure 3.6: An AS unit.

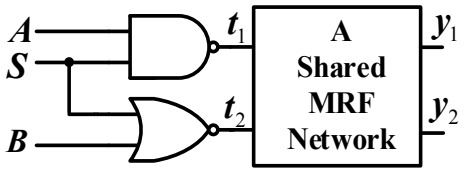


Figure 3.7: The schematic of a NAND-NOR group.

Table 3.1: Truth Table for the Proposed NAND-NOR Unit

A	B	S	t_1	t_2	y_1	y_2	Valid states
0	0	0	1	1	1	1	$t_1 t_2 y_1 y_2$
0	0	1	1	0	1	0	$t_1 \overline{t_2} y_1 \overline{y_2}$
0	1	0	1	0	1	0	$t_1 \overline{t_2} y_1 \overline{y_2}$
0	1	1	1	0	1	0	$t_1 \overline{t_2} y_1 \overline{y_2}$
1	0	0	1	1	1	1	$t_1 t_2 y_1 y_2$
1	0	1	0	0	0	0	$\overline{t_1} \overline{t_2} y_1 \overline{y_2}$
1	1	0	1	0	1	0	$t_1 \overline{t_2} y_1 \overline{y_2}$
1	1	1	0	0	0	0	$\overline{t_1} \overline{t_2} y_1 \overline{y_2}$

The first step is to separate the logic gates in a MUX into two groups. Inspired by the MS NAND gate [38], a NAND gate can be grouped with a NOR gate. The other OR gate can be grouped with another OR gate in an AS unit. The NAND-NOR groups and the OR-OR group are marked with a gray color as shown in Figure 3.6. These gate groups will share a common MRF network in each group. First, we need to design a common MRF network for the NAND-NOR group shown in Figure 3.7. Assume that A, B and S are the inputs of the NAND gate and the NOR gate; t_1 and t_2 are the outputs of the NAND gate and the NOR gate while

y_1 and y_2 are the outputs of the shared MRF network. Since we know the intended logic operations among these variables, we can construct a truth table for the group. Unlike normal truth tables, Table 3.1 contains a column for the valid states formed by t_1, t_2, y_1 and y_2. This can help us write out the compatibility function (the opposite of the energy function) more easily. According to Table 3.1, the compatibility function of t_1, t_2, y_1, y_2 with t_1, t_2, y_2 as inputs and y_1 as an output is

$$\begin{aligned} C(t_1,t_2,y_2,y_1) &= (t_1t_2y_2y_1 + t_1\overline{t_2}\,\overline{y_2}y_1) + \overline{t_1}t_2y_2\overline{y_1} \\ &= \overline{t_1}\,\overline{t_2}\,\overline{y_2}y_1 + \overline{t_1}t_2y_2\overline{y_1} \end{aligned} \tag{3.9}$$

If we regard t_1, t_2, y_1 as inputs and y_2 as an output, the compatibility function will be changed to

$$\begin{aligned} C(t_1,t_2,y_1,y_2) &= t_1t_2y_1y_2 + (t_1\overline{t_2}y_1\overline{y_2} + \overline{t_1}\,\overline{t_2}y_1y_2) \\ &= t_1t_2y_1y_2 + \overline{t_1}t_2y_1\overline{y_2} \end{aligned} \tag{3.10}$$

We can choose one term from Equation (3.9) and one term from Equation (3.10) to build a new compatibility function:

$$\begin{aligned} C(t_1,t_2,y_1,y_2) &= C(t_1,t_2,y_2,y_1) \\ &= \overline{t_1}\,\overline{t_2}\,\overline{y_2}y_1 + \overline{t_1}t_2y_1\overline{y_2} \end{aligned} \tag{3.11}$$

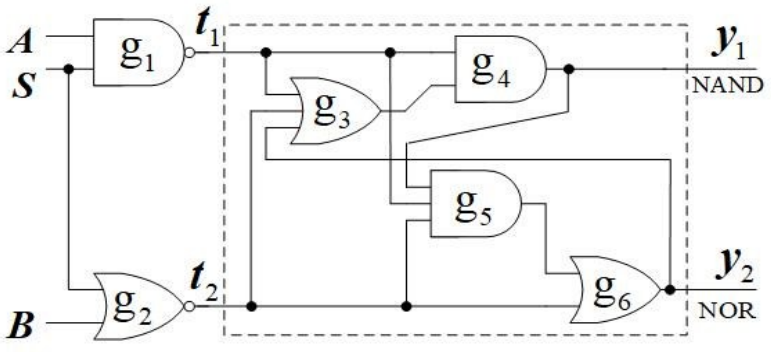


Figure 3.8: The proposed NAND-NOR group with a shared MRF network.

We can then map the Equation (3.11) to an MRF network as shown in Figure 3.8. In Section 2.2, we have presented a simple proof that an AND gate can produce a stable output "0" with the noisy inputs "0". Similarly, an OR gate outputs a stable logic "1" with the noisy inputs "1". Therefore, the unstable bits in t_1 and t_2 from g_1 and g_2 are more likely to produce correct outputs with the help of g_4 and g_6. In this way, the final output y_1 and y_2 will have fewer upsets in both "1" and "0".

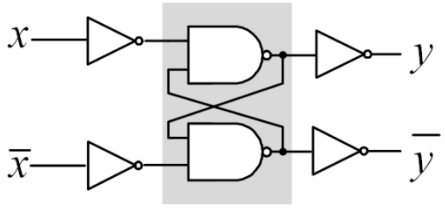


Figure 3.9: An MS inverter [38].

Second, we need to design the common MRF network for the OR-OR group. Unlike the NAND-NOR group, the OR-OR group doesn't share a common input signal. It is difficult to simplify the compatibility function as there are no overlapped valid terms. Figure 3.9 shows an MS inverter. The MRF network is based on the compatibility function:

$$C(x, y) = x\bar{y} + \bar{x}y \tag{3.12}$$

Both the MS inverter and the MS NAND gate have the NAND-NAND feedback structures which help stabilize the output signals. We try to design the MRF network for the OR-OR group by improving the MS feedback structures. As the outputs of two OR gates, t_3 and t_4 are unstable in bit "0". The Unstable bit "0" can be transferred to a stable bit "0" by an AND gate. Figure 3.10 illustrates the shared MRF network for the OR-OR group. g_9 and g_{11} are the NAND gates used

to strengthen the unstable bit "0" from signals t_3 and t_4, respectively. g_{10} is used to help determine the output logic. Without g_{10}, the feedback structure would enter a latch mode when t_3 and t_4 are both "0s".

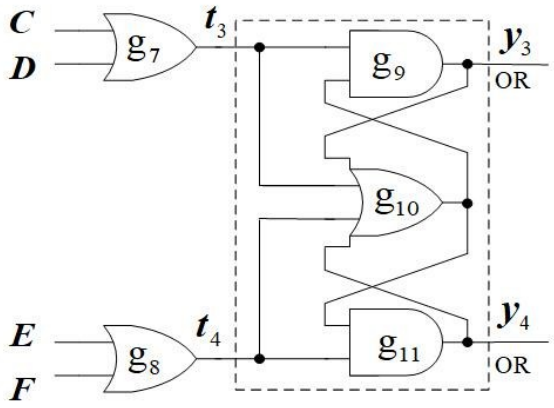


Figure 3.10: The proposed OR-OR group with a shared MRF network.

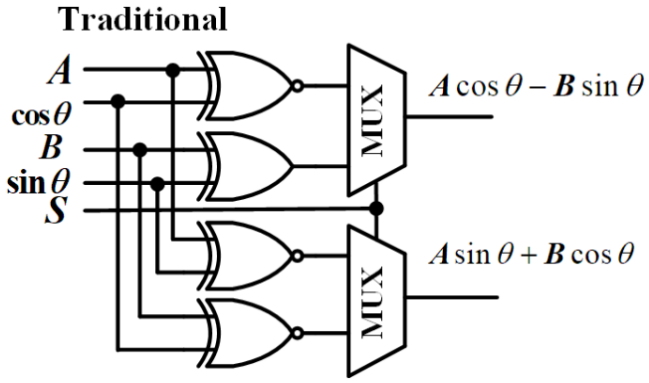


Figure 3.11: An ASM with traditional non-MRF CMOS gates.

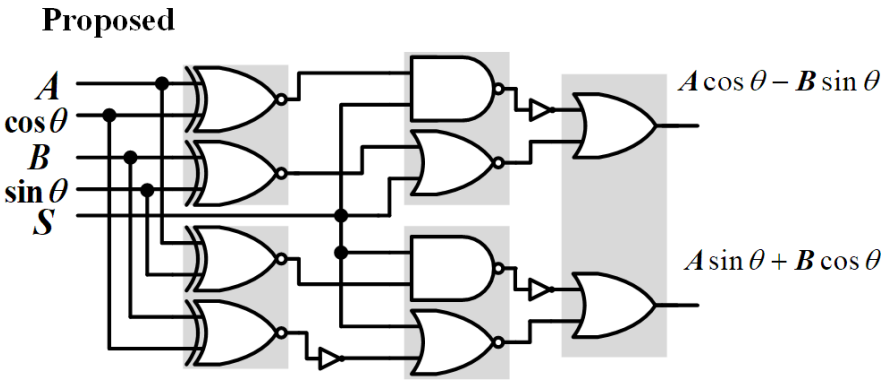


Figure 3.12: An ASM with proposed MRF gates.

The other computing unit for an 8-point DCT system is the ASM shown in Figure 3.2(b). It contains four stochastic multipliers implemented by XNOR gates and two stochastic adders implemented by MUXs shown in Figure 3.11. We still separate gates into groups, including two XNOR-XNOR groups, two NAND-NOR groups and one OR-OR group shown in Figure 3.12. The detailed design of the proposed MRF-based stochastic adder has been presented in Figure 3.8 and Figure 3.10. We only need to design the shared MRF network for the XNOR-XNOR group. The XNOR-XNOR group is like the OR-OR group. It doesn't have a common input signal as well. Therefore, it can directly use the same feedback structure as the one shown in Figure 3.13. However, as the outputs of two XNOR gates, t_5 and t_6 have the same probability to be incorrect in "0" and "1". In other words, t_5 and t_6 are both unstable in "0" and "1". The shared structure may be not so that helpful to tolerate noise as the one used in the OR-OR group. It can only transfer the unstable bit "0" to the stable bit "0" by g_{14} and g_{16}.

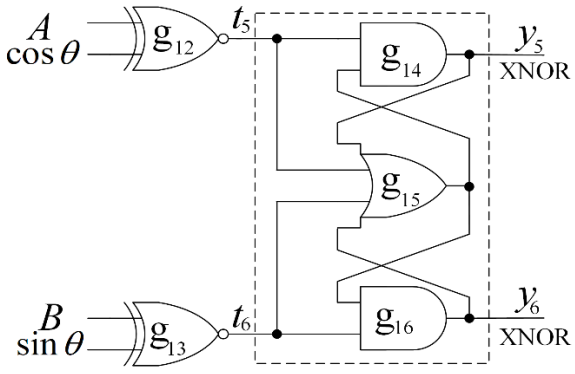


Figure 3.13: The proposed XNOR-XNOR group with a shared MRF network.

3.3.2 MRF-based Stochastic DCT circuit

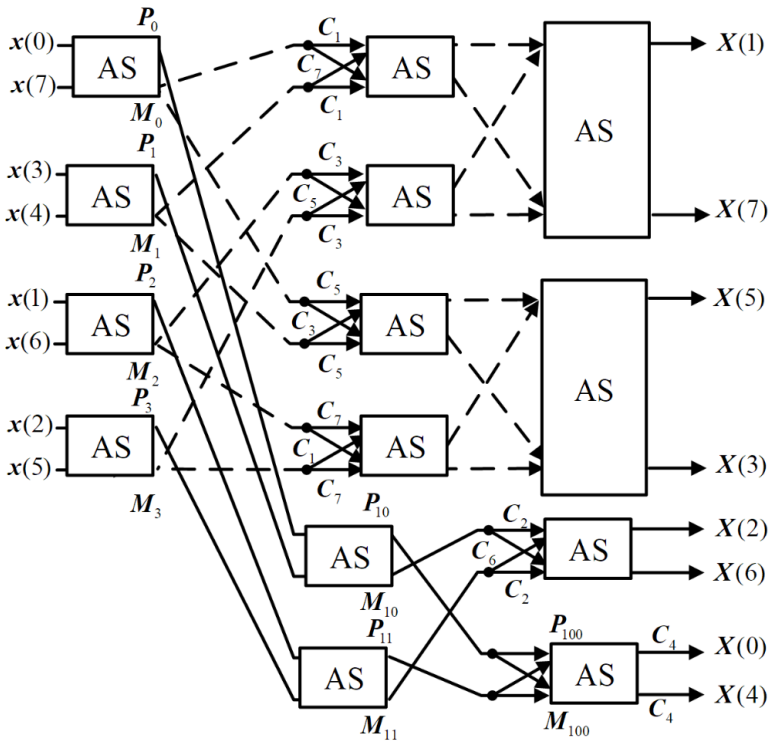


Figure 3.14: The 8-point 1D-DCT circuit with the proposed AS units.

In order to design an MRF-based stochastic 1D-DCT circuit, we need to use the proposed MRF-based stochastic computing units to implement the 8-point 1D-DCT algorithm. Figure 3.1 is the direct implementation of Equation (3.4). More stochastic adders are used in the system. It is also found that a stochastic multiplier requires only one XNOR gate. We have mentioned in Section 3.3.2 that the shared structure in a stochastic multiplier may be not so that helpful to tolerate noise as the one used in the OR-OR group of a stochastic adder. If we don't make any changes to the schematic shown in Figure 3.1, it's more cost-effective to only use the MRF-based stochastic adders as shown in Figure 3.14. The stochastic multipliers used in the ASM unit are implemented by traditional non-MRF CMOS gates. In this way, the system can obtain relatively high noise immunity without

consuming too much area. However, we then lose the noise immunity of the ASM units. It is necessary to find a better solution to solve this problem.

In Figure 3.14, there are 22 multipliers. Considering that the noise-immunity performance of the XNOR-XNOR group is not as good as that of the other groups, in order to maximize the capability of the noise immunity, the required number of stochastic multipliers should be reduced as much as possible. We must simplify the 8-point 1D-DCT algorithm by using the angle sum and difference identities. Equation (3.4) is the original set of equations mentioned in Section 3.1:

$$\begin{cases} X(1) = M_0C_1 + M_1C_7 + M_2C_3 + M_3C_5 \\ X(3) = M_0C_3 - M_1C_5 - M_2C_7 - M_3C_1 \\ X(5) = M_0C_5 + M_1C_3 - M_2C_1 + M_3C_7 \\ X(7) = M_0C_7 - M_1C_1 - M_2C_5 + M_3C_3 \\ X(2) = M_{10}C_2 + M_{11}C_6 \\ X(6) = M_{10}C_6 - M_{11}C_2 \\ X(4) = M_{100}C_4 \\ X(0) = P_{100}C_4 \end{cases}$$

where

$$\begin{aligned} &M_0 = x_0 - x_7;\ P_0 = x_0 + x_7; \\ &M_1 = x_3 - x_4;\ P_1 = x_3 + x_4; \\ &M_2 = x_1 - x_6;\ P_2 = x_1 + x_6; \\ &M_3 = x_2 - x_5;\ P_3 = x_2 + x_5; \\ &M_{10} = P_0 - P_1;\ P_{10} = P_0 + P_1; \\ &M_{11} = P_2 - P_3;\ P_{11} = P_2 + P_3; \\ &M_{100} = P_{10} - P_{11};\ P_{100} = P_{10} + P_{11}. \end{aligned}$$

By studying these equations, the equations for $X(1)$, $X(3)$, $X(5)$ and $X(7)$ can be simplified to reduce the use of multiplication. Specifically, these equations can be simplified to the following equations (3.13).

$$\begin{cases} X(1) = C_4[(K_1 + K_3) + (K_2 + K_4)] \\ X(3) = K_1 - K_2 \\ X(5) = K_3 - K_4 \\ X(7) = C_4[(K_1 - K_3) - (K_4 - K_2)] \end{cases} \tag{3.13}$$

where

$$\begin{aligned} K_1 &= M_0 C_3 - M_1 C_5; \\ K_2 &= M_2 C_7 + M_3 C_1; \\ K_3 &= M_0 C_5 + M_1 C_3; \\ K_4 &= M_2 C_1 - M_3 C_7; \end{aligned} \tag{3.14}$$

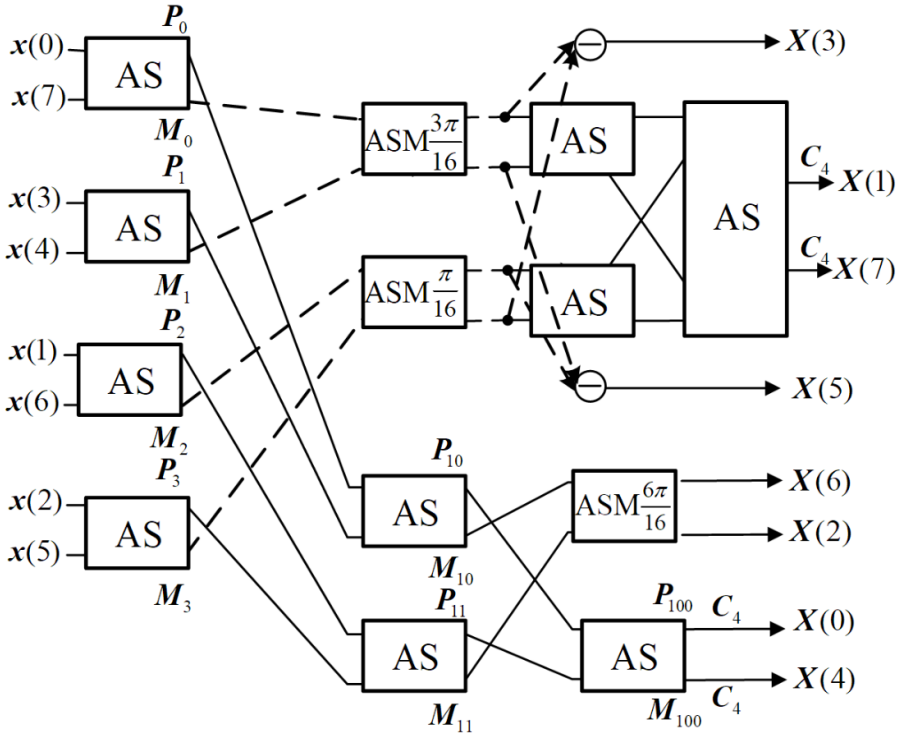


Figure 3.15: The 8-point 1D-DCT circuit with the proposed AS and ASM units.

In Equation (3.14), K_1 and K_3 can be implemented by an ASM unit, while K_2 and K_4 can be implemented by another ASM unit. Mapping Equation (3.4) and Equation (3.13) into the 8-point 1D-DCT circuit, we obtain the new schematic shown in Figure 3.15. It contains 10 AS units and 3 ASM units. In terms of the number of stochastic adders and multipliers, it contains 28 adders and 16 multipliers. Compared with Figure 3.14, the number of the multipliers is reduced from 22 to 16. In the mapping of the proposed MRF-based stochastic adders and multipliers, the remaining two independent subtractors can be grouped together, while the remaining four independent multipliers can be combined into two groups to share a common MRF network in each group.

3.4 Simulation and Discussion

To investigate the performance of the AS unit and the ASM unit in the presence of noise, we measured the output error rates of the traditional non-MRF version, the MRF version [33], the MS version [38] and the proposed shared MRF version in HSPICE. In the non-MRF version, all of the logic gates are conventional designs without the enhancements of MRF theory. All parameters are based on the 65-nm Berkeley Short-channel IGFET Model4 (BSIM4). The supply voltage of each component was as low as 0.25 V. The value is near to the threshold voltage of the transistors used in the 65nm library. In this case, more errors will occur in the simulated circuit with the interference of noise. Additive White Gaussian Noise (AWGN) is chosen as the noise model injected to the circuit. Thermal noise is considered by setting the temperature to 40 ^{o}C for the entire circuit. The Gaussian

noise is zero mean and standard deviation σ. The intensity of the AWGN varied from the value 3.3 to 5.7 in the form of SNR calculated by $SNR = 20\log_{10}\frac{A_{Signal}}{A_{Noise}}$, where A_{signal} is the power supply voltage and A_{noise} is the standard deviation of the noise. Since there are two outputs in the AS unit, both of the outputs ADD and SUB are simulated. The results are marked using different symbols in Figure 3.16.

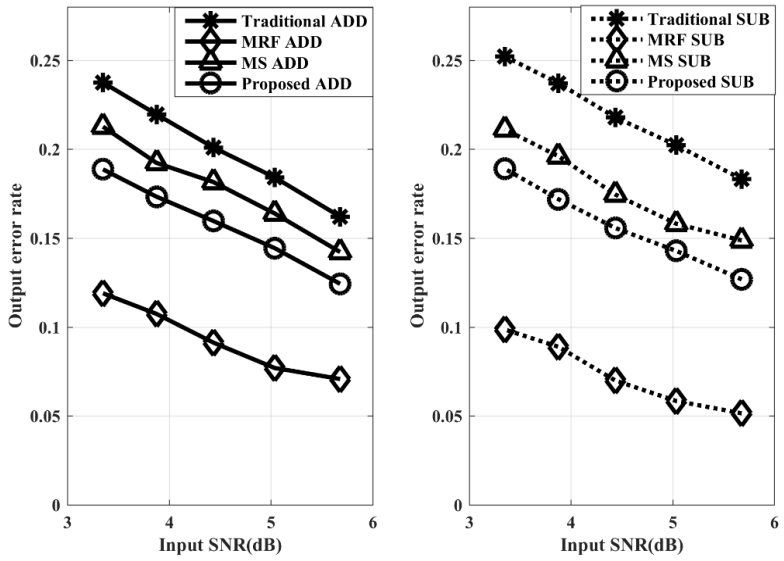


Figure 3.16: Error rate results of the AS units.

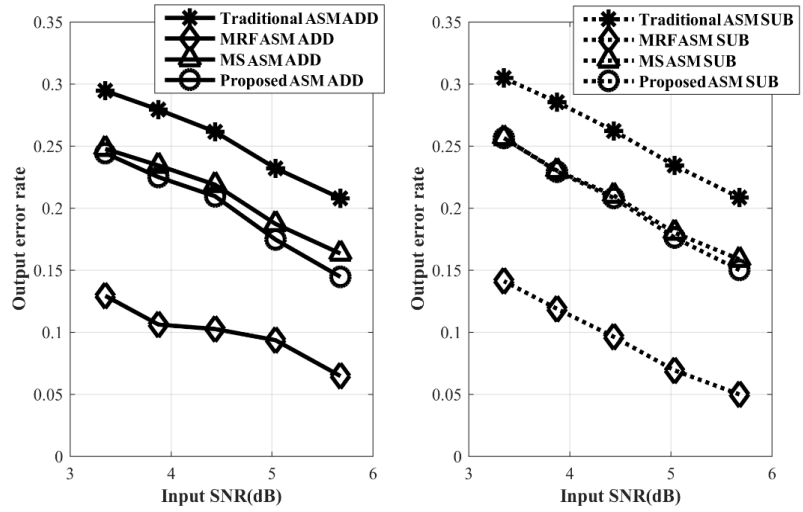


Figure 3.17: Error rate results of the ASM units.

The quality of the output signals is the best in the MRF version [33] as the output error rates are the lowest ones. The proposed version has the second-lowest output error rates. This is intended to save area. Figure 3.17 shows the error rate results of the ASM units. The rank of each version in the ASM units corresponds to that in the AS units. However, the results of the proposed ASM are very close to the results for MS. This confirms what we have mentioned in Section 3.3.2. The noise-immunity performance of the XNOR-XNOR group is not as good as that of the other groups like the OR-OR group. The symmetric distribution of the outputs in an XNOR gate, that an XNOR gate has the same probability to output "1" and "0", indeed has an adverse effect on the noise-immunity performance from the perspective of probability. The error rate results of the AS units are generally smaller than these in the ASM units. This is because more noise is injected into the ASM units by the extra inputs $\cos\theta$ and $\sin\theta$.

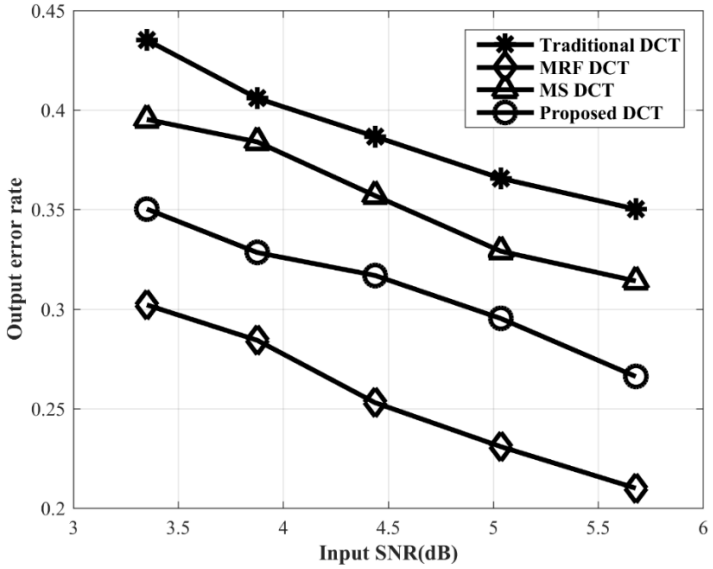


Figure 3.18: Error rate results of the 8-point 1D-DCT system.

We also simulated the structure of the 8-point SC 1D-DCT circuit shown in Figure 3.15. All of the logic gates in the required AS units and ASM units were built by the non-MRF version, the MRF version [33], the MS version [38] and the proposed version, respectively. As there are 8 outputs to the 8-point 1D-DCT system, the average output error rates of the 8 outputs were calculated to have a fair comparison shown in Figure 3.18. The order of the output error rates from the lowest to the highest is the MRF version [33], the proposed version, the MS version [38] and the non-MRF version, which is the same with what is shown in Figure 3.16 and Figure 3.17. The proposed MRF version have better results than MS version. Although the output error rates of the MRF version [33] are always the best ones, it costs the largest area as well. The MRF version [33] is the performance limitation for all MRF-based design, in terms of the noise immunity.

Table 3.2: Comparison of the Gate Count

Structure	Traditional	MRF-based		
	Standard CMOS	MRF [33]	MS [38]	proposed
AS	9	107	50	28
ASM	21	167	90	51
1D-DCT	167	1726	856	484

Table 3.3: Comparison of Power, Area and Delay

Scheme	Traditional SC 1D DCT	MRF-based SC 1D DCT		
	Standard CMOS	MRF [33]	MS [38]	proposed
Power (μW) ER=0.21	2.1@0.48V	1.6@0.25V	2.3@0.34V	1.1@0.29V
Area (μm^2)	128.2	1852.2	1180.0	620.3
Delay (ns)	0.5	3.4	1.7	2.9

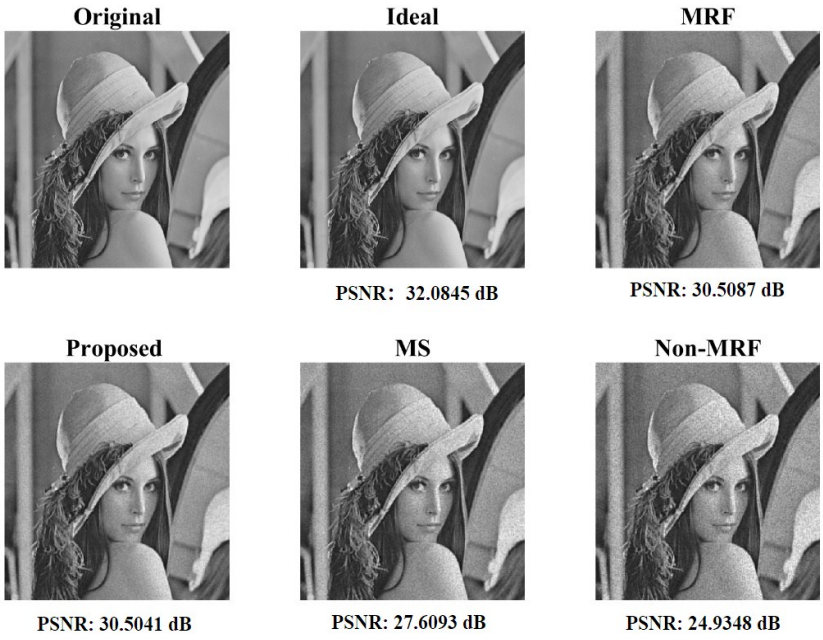


Figure 3.19: Noise-immunity comparison with/without 11.5 dB SNR noise.

Note that according to TSMC 65-nm core library databook, the gate count of a two-input NAND/NOR gate is 1; the gate count of a two-input AND/OR gate is 1.5; the gate count of a three-input AND/OR gate is 2; the gate count of a four-input OR gate is 2.5; and the gate count of a two-input XNOR gate is 3. We can calculate the gate count results of the computing units and the 1D DCT circuit with different gate designs. The comparison of gate count is given in Table 3.2. Compared with the MRF version [33], the average gate count of the proposed version is reduced by 71%. The reduced average gate count changes to 43%, when it is compared with the MS version [38]. Table 3.3 illustrates the comparison of power, area and delay. Power was simulated in HPICE while area and delay were simulated in Design Compiler. Compared with the MS version [38], the average output error rate of the proposed design under different SNRs is

reduced by 13% while it saves about 47% hardware cost and 52% power when the error rate is 0.21. Although the average output error rate of the proposed version is higher than that of the MRF version [33], the proposed design cost 67% less area, 31% less power (ER=0.21) and 15% less delay. Figure 3.18 is the visual reflection of the capability of noise-immunity when the circuits are injected with 11.5dB SNR noise or in the ideal condition without any noise interference and 5% sequence calculation errors. These images are processed by repeating 1D DCT twice and the Inverse 1D DCT (IDCT) twice in MATLAB with/without the effects of noise. While the MRF-based gate designs tolerate most of noise, the quality of the traditional non-MRF gate design is strongly weakened by noise.

3.5 Conclusion

The MRF approach offers us a way to trade off area and operation speed for error tolerance for logic circuits. The direct mapping of MRF theory to hardware provides benefits but at a great cost. It is necessary to find a more cost-effective way to express logic circuits with the approach. It should not be just the simplification of the MRF networks like the MS design. Fortunately, a low-cost method, SC, attracts our attention. SC is a perfect partner since it can directly reduce the complexity of a circuit and as an unweighted presentation it can tolerate errors inherently. Therefore, the combination of the two approaches is a win-win situation. The idea is applied on the logic circuit to implement an 8-point 1D DCT. We first simplify MRF networks by grouping two logic gates together and integrating the idea of transferring unstable bits to stable ones. It is found that

the XNOR-XNOR group has relatively limited noise-immunity performance compared with other groups. We have to simplify the DCT algorithm to cut down the use of multiplications. The final design requires only 10 multipliers which can be paired to share the MRF networks. The ASM Simulation results correspond to the finding about the performance of the XNOR-XNOR group. The proposed circuit achieves 20% decrease of error rate at a low supply voltage of 0.25V compared with the conventional non-MRF stochastic design. It outperforms the MS design by saving 47% area and 52% power consumption under the same error rate 0.21, but at a cost of 1.7 times the propagation delay. Although the capability of noise immunity is the best with complete MRF mapping, its area consumption increases by 67% with 1.6 μW power consumption and it increases the propagation delay by 3.4 ns. Therefore, the proposed approach can process an 8-point 1D DCT at relatively low power, low area cost and high noise immunity.

Chapter 4

Coding-based Partial MRF Circuit Design

4.1 Coding-based Partial MRF Methodology

In Chapter 3, we discussed the idea of building a common MRF network to save hardware cost. In this section we propose a general cost-effective MRF mapping method, specifically called a Coding-based Partial MRF (CPMRF) method, that aims to improve upon the previous work. The direct mapping of the MRF theory to digital logic circuits considers all possible states, and thus the clique energy function is complete. If we prefer to group two logic gates together to build a shared MRF network, the combination of the complete clique energy functions of the two logic gates is unnecessary. It is hard to save hardware cost in this way. However, what we can do is to choose some of the terms in the complete clique energy function of each gate and form a new energy function. In other words, only partial MRF clique energy terms are used in the new function.

Before introducing the CPMRF design method, some concepts are defined based on the properties of logic gates. Since logic gates perform Boolean functions and their outputs are Bernoulli random variables, the terms present in their clique energy can be grouped with either output logic “1” and “0”. From this point of view, the distribution of these “1s” and “0s” deserves our attention. We observe that logic gates can be divided into two groups with respect to their output values. Asymmetric logic gates are referred to as the gates outputting more “1s” or “0s” in their truth tables. The distribution of the outputs is asymmetric. In

other words, the two possible outcomes occur with different probabilities. For example, a two-input NAND gate is an asymmetric gate. If the probabilities of the input cases ("00", "01", "10", "11") are equal, a NAND gate is more likely to output "1" as it outputs "1" with probability $\frac{3}{4}$ and "0" with $\frac{1}{4}$. Similarly, NOR, OR and AND are all asymmetric gates. On the contrary, symmetric gates have symmetric output distributions such as NOT, XOR and XNOR. They output "0" and "1" equally. For some asymmetric gates and symmetric gates, their partial clique energy can be combined with others to compensate for the lost terms. These gates can be grouped in pairs denoted as complementary logic groups, such as a NAND-OR group. However, some gates could only form non-complementary logic groups such as a NOR-NOR group. Based on these observations, we summarized the specialized CPMRF mapping methods for the complementary logic groups and non-complementary logic groups, respectively.

First, we present the CPMRF mapping method for the complementary logic groups. Let us take an AND-NOR group for example. Table 4.1 gives the truth table of an AND-NOR group. The partial clique energy of an AND gate is denoted by $U_{\text{p-AND}}$. The valid minterms which have higher probability to occur in terms of the outputs are assigned to be the terms in the partial clique energy function. Consequently, the energy function of $U_{\text{p-AND}}$ is

$$\begin{aligned} U_{\text{p-AND}}(x_1, x_2, y) &= -\overline{x_1} \cdot x_2 \cdot \overline{y} - x_1 \cdot \overline{x_2} \cdot \overline{y} - \overline{x_1} \cdot \overline{x_2} \cdot \overline{y} \\ &= -(\overline{x_1 \cdot x_2}) \cdot \overline{y} \end{aligned} \tag{4.1}$$

Table 4.1: Energy Truth Table of an AND-NOR Group

Input		Valid output				
x_1x_2		AND y_{AND}		NOR y_{NOR}		
0	0	0	$U_{\text{p-AND}}$	$\overline{x_1}\cdot\overline{x_2}\cdot\overline{y_{\text{AND}}}$	U_1	1
0	1	0		$\overline{x_1}\cdot x_2\cdot\overline{y_{\text{AND}}}$	$\overline{x_1}\cdot x_2\cdot\overline{y_{\text{NOR}}}$	0
1	0	0		$x_1\cdot\overline{x_2}\cdot\overline{y_{\text{AND}}}$	$x_1\cdot\overline{x_2}\cdot\overline{y_{\text{NOR}}}$	0
1	1	1	U_2	$x_1\cdot x_2\cdot\overline{y_{\text{NOR}}}$		0

The energy function of $U_{\text{p-NOR}}$ is

$$\begin{aligned} U_{\text{P-NOR}}(x_1,x_2,y) &= -\overline{x_1}\cdot x_2\cdot\overline{y} - x_1\cdot\overline{x_2}\cdot\overline{y} - x_1\cdot x_2\cdot\overline{y} \\ &= -(x_1+x_2)\cdot\overline{y} \end{aligned} \tag{4.2}$$

Compared with the complete energy function of NOR, we need a term from the $U_{p\text{-AND}}$ to complement the partial clique energy of NOR:

$$U_{\text{NOR}} = U_{\text{P-NOR}} - \overline{x_1}\cdot\overline{x_2}\cdot\overline{y_{\text{AND}}} \tag{4.3}$$

Similarly, we choose one term from the $U_{p\text{-NOR}}$, $x_1\cdot x_2\cdot\overline{y_{NOR}}$, to complement the partial clique energy of AND:

$$U_{\text{AND}} = U_{\text{P-AND}} - x_1\cdot x_2\cdot\overline{y_{\text{NOR}}} \tag{4.4}$$

$U_{\text{C-AND-NOR}}$ is denoted to complete the partial energy terms of an AND-NOR group as

$$U_{\text{C-AND-NOR}} = -(x_1 \cdot x_2 \cdot \overline{y_{\text{NOR}}} + \overline{x_1} \cdot \overline{x_2} \cdot \overline{y_{\text{AND}}}) \tag{4.5}$$

Equation (4.5) is the sum of a term in Equation (4.2) for a NOR gate and a term in Equation (4.1) for an AND gate. Since we want to build a common MRF network for an AND gate and a NOR gate, it is necessary to choose one term from the complete energy function of a logic gate to complete the partial energy of the other. The terms are $\overline{x_1} \cdot \overline{x_2} \cdot \overline{y_{\text{AND}}}$ from AND and $x_1 \cdot x_2 \cdot \overline{y_{\text{NOR}}}$ from NOR. In this way, the simplified clique energy function of an AND-NOR group can be deduced from Equations (4.3), (4.4) and (4.5):

$$\begin{aligned} U_{\text{AND-NOR}} &= -(x_1 \cdot x_2 \cdot \overline{y_{\text{NOR}}} + \overline{x_1} \cdot \overline{x_2} \cdot \overline{y_{\text{AND}}}) \\ &\quad -(\overline{x_1} \cdot x_2 \cdot \overline{y_{\text{NOR}}} + x_1 \cdot \overline{x_2} \cdot \overline{y_{\text{NOR}}}) \\ &\quad -(\overline{x_1} \cdot x_2 \cdot \overline{y_{\text{AND}}} + x_1 \cdot \overline{x_2} \cdot \overline{y_{\text{AND}}}) \\ &= U_{\text{C-AND-NOR}} - (x_1 \oplus x_2) \cdot \overline{y_{\text{NOR}}} - (x_1 \oplus x_2) \cdot \overline{y_{\text{AND}}} \end{aligned} \tag{4.6}$$

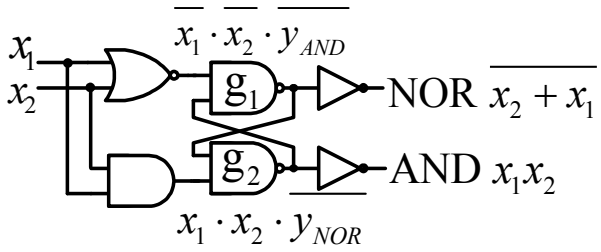


Figure 4.1: A compensation clique energy-based structure for the AND-NOR group.

Mapping Equation (4.5) to a logic circuit for the lost terms in a AND-NOR group is shown in Figure 4.1. It can be added the partial terms to form Equation (4.6). The corresponding circuit is shown in Figure 4.2. Gates g_3, g_4 and g_5 form a

coding unit in Figure 4.2 to the compensate for the loss in clique energy in the partial clique energy. First, the coding unit helps the entire MRF network implement the complete energy for a NOR-AND group. In detail, gates g4 and g5 shown in Figure 4.2 are used to perform the clique energy terms $(x_1 \oplus x_2)\cdot\overline{y_{\text{NOR}}}$ and $(x_1 \oplus x_2)\cdot\overline{y_{\text{AND}}}$. Gates g1 and g2 shown as the feedback structure can implement the energy function of $U_{\text{C-AND-NOR}}$.

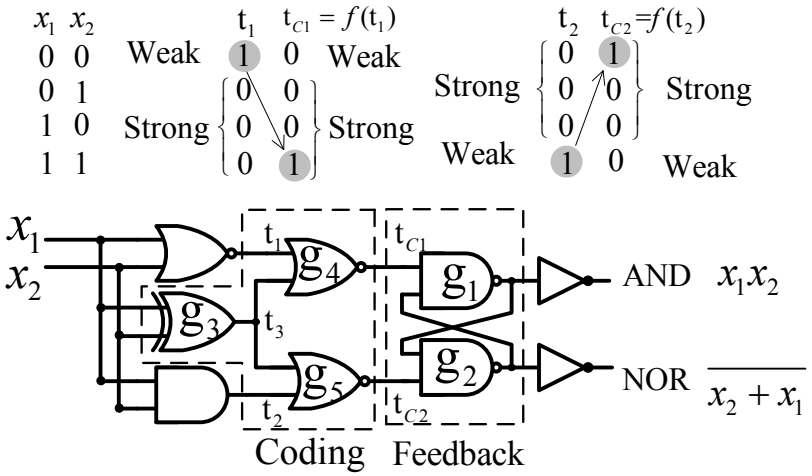


Figure 4.2: The coding-based structure for a complementary AND-NOR group.

On the other hand, the coding unit can transform weak bits to strong bits. The MS-MRF structure [38] performs a simplified MRF network, but it fails to compensate for the lost terms in the master and the feedback loop. Thus it trades off too much performance for area-saving according to the study [46]. From the perspective of probability, NAND has an asymmetric output distribution. Considering the interference from noise, both bits "1" and "0" have a certain probability of being erroneous for input and output signals. In order to investigate the conditional probability of outputs being errorless, x_1, x_2 are denoted as the inputs of a two-input NAND gate, while y stands for the output. Assume that an

input of a NAND gate has the probability being erroneous and being errorless with the effect of noise. $p(y|x_1 x_2)$ is denoted as the conditional probability of output y being errorless. We have $p(1|00) \geq p(1|01) = p(1|10) \geq p(0|11)$. Under the condition that the inputs are {00,01,10}, a NAND gate is more likely to generate correct outputs. In other words, outputs being "1" has fewer upsets than outputs being "0". In the MS structure, almost all gates are asymmetric, but there is no probability match in both the master stage and the slave stage. It results in failures to compensate for the performance difference, compared with that of the direct MRF mapping method.

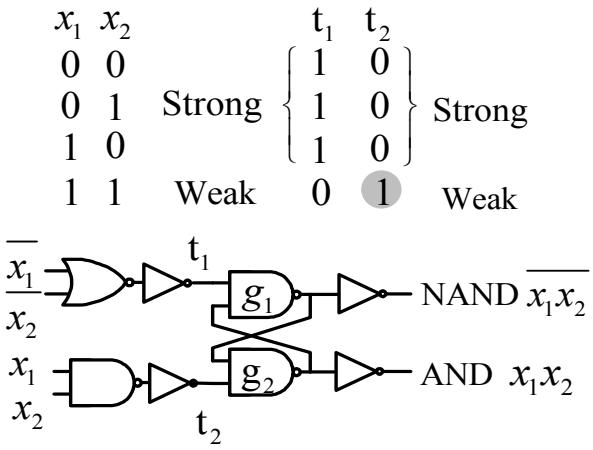


Figure 4.3: Bit analysis for the MS [38] structure.

Figure 4.3 illustrates the situation. Both t_1 and t_2 are the output signals of NOR-NOT and NAND-NOT gates. They have different probabilities of being errorless when being "1" and "0". In our view, strong bits have a high probability to be errorless and weak bits has a high probability to be erroneous. It is a win-win situation when strong bits from the master stage are connected to a following structure to which the bits are still strong. However, it can also result in double-

weakened function for the weak bits. In Figure 4.3, a weak bit "1" of t_2 is connected to a NAND gate. Since a NAND gate can only transform a weak input "0" to a strong output "1", a weak bit "1" is weakened again. The weak input-pair of the next level such as {11} is weak for g_2. Therefore, the MS sacrifices the signal quality of the output of g_2 in bit "0". Differently, the proposed coding unit shown in Figure 4.2 transforms the weak bit "1" of t_1 and t_2 to strong bit "1" of t_{c1} and t_{c2}. In this way, the proposed structure has better noise tolerance than the conventional MS structures. For the function of a half adder, the proposed coding unit for a complementary AND-XOR group is shown in Figure 4.4. An MRF network can perform Boolean operations other than the functions of the input gates. Based on the same idea, we can also propose a common CPMRF network for NAND-OR, AND-XOR (NXOR), OR-XOR (NXOR), NAND-XOR (NXOR), NOR-XOR (XNOR), NAND-AND, NOR-OR and XOR-NXOR groups. The CPMRF networks consist of the same feedback structure and a different coding struture aiming at completing the logic function and complementing the lost energy terms.

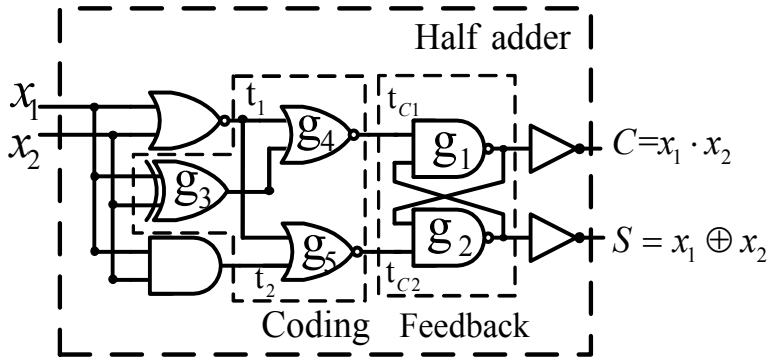


Figure 4.4: Coding based structure for a complementary AND-XOR group.

In a fast tree-based carry-lookahead adder, a MUX-AND block is used to produce the carry bit. In this block, we do not use two AND gates and an OR gate to implement the function of a MUX since an AND-AND group is a non-complementary logic group. Instead, we choose an AND gate, a NOR gate and an OR gate based on Equation (4.7).

$$y_{mux} = x_a \cdot x_s + x_b \cdot \overline{x_s} = x_a \cdot x_s + \overline{\overline{x_b} + x_s} \tag{4.7}$$

Gates g_1 and g_2 shown in Figure 4.5(a) perform $x_a \cdot x_s$ and $\overline{\overline{x_b} + x_s}$ in Equation (4.7) by our proposed coding-based AND-NOR group shown in Figure 4.5(b). Gate g_4 shown in Figure 4.5(a) combines the results of g_1 and g_2 for the MUX function $x_a \cdot x_s + \overline{\overline{x_b} + x_s}$. Gates g_3 and g_4 are grouped together in order to build a cost-effective MRF shown in Figure 4.5(b). Therefore, in the application of the CPMRF method, we should consider complementary logic groups first and try to convert non-complementary logic groups to complementary logic groups by Boolean algebra.

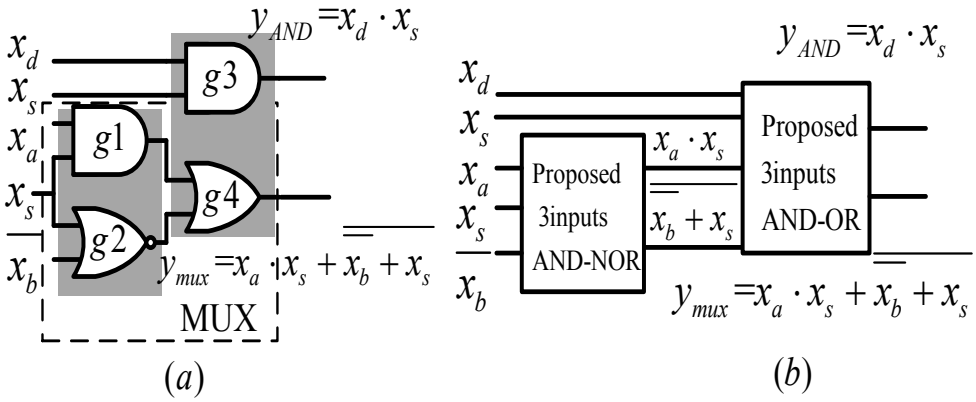


Figure 4.5: Mixed mapping and CPMRF MUX (a) Mixed mapping MUX (b) CPMRF pair-based MUX structure.

For non-complementary logic groups, their CPMRF networks cannot be deduced from the energy truth tables. It is hard to simplify the energy functions since there are no common terms. In this case, we need to consider the properties of logic gates to complement the weak bits for the gate groups. A NOR-NOR group is used as a case study, shown in Figure 4.6. The signals t_1 and t_2 are denoted as the outputs of gate g_5 and g_6. As the outputs of a NOR gate, both of them are strong in bit "0" denoted as "S_1" in the first stage and weak in bit '1' denoted as "W_1". Gate g_4 is used to code to prevent double-weakened cases. In the second stage feedback structure, g_1 and g_2 can compensate the weak bits "1" in the first stage as the output of a NOR gate. For t_1, t_2 and t_c, none of them are doubly weakened in the second stage in bit "1" and "0".

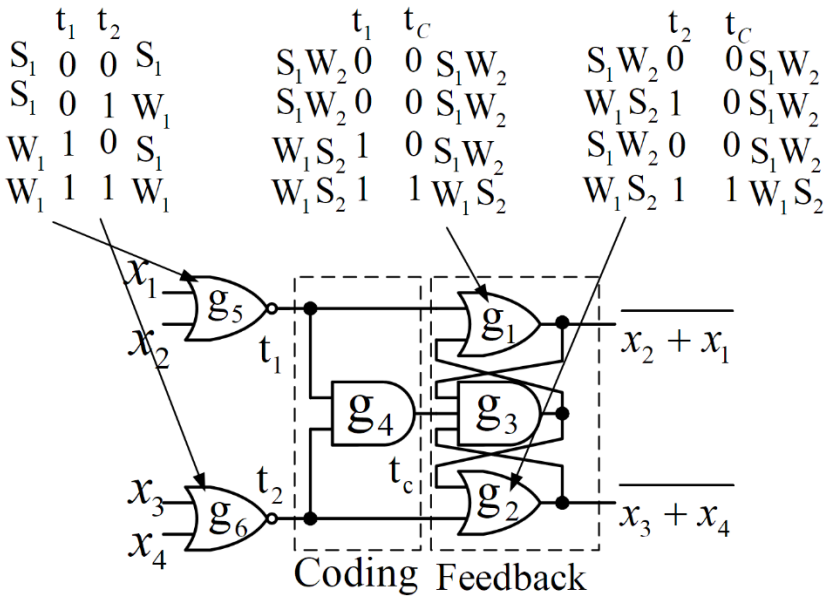


Figure 4.6: Non-complementary CPMRF group: NOR-NOR.

Another case study is for a NAND-NAND group, as shown in Figure 4.7. Similarly, the signals t_1 and t_2 are denoted as the outputs of two NAND gate g_5 and g_6. As the outputs of a NAND gate, t_1 and t_2 are weak in bit "0" denoted as

"W_1" in the first stage and strong in bit '1' denoted as "S_1". In the coding structure, an OR gate g_4 is used to code t_1 and t_2 to an aided signal t_c. The AND gates, g_1 and g_2, can transfer the weak bits "0" in the first stage to the strong bits "0" as the outputs of the AND gates. In Figure 4.7, the weak bits in t_1, t_2 and t_c in the first stage are complemented by the feedback structure in the second stage. In conclusion, there are two kinds of CPMRF networks for non-complementary logic groups. If the weak bits are "1", the CPMRF networks should be the same as the one in Figure 4.6. If the weak bits are "0", we can use the same CPMRF network shown in Figure 4.7. Since the CPMRF networks for non-complementary logic groups are separated into two fixed structures, they are much simpler to implement with the assist of EDA tools than complementary logic groups.

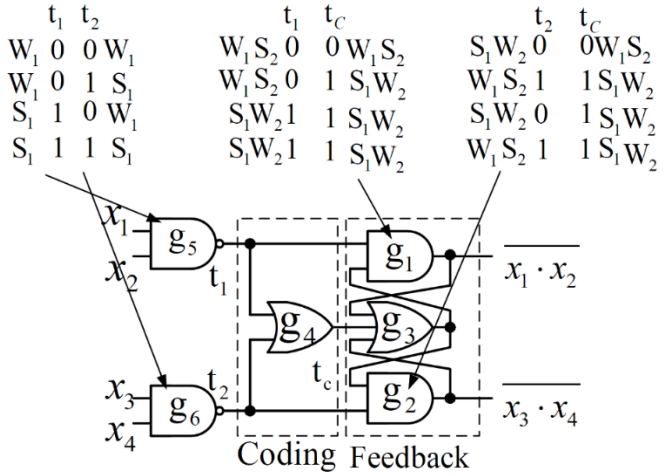


Figure 4.7: Non-complementary CPMRF group: NAND-NAND.

4.2 Comparison and Simulation

The proposed CPMRF method constructs a common MRF network for a logic group to save hardware cost. In one network, multiple Boolean operations are performed which is not limited to the input logics. In Section 4.1, Figure 4.2,

Figure 4.4, Figure 4.5, Figure 4.6 and Figure 4.7 illustrate the proposed structures for AND-NOR, AND-XOR, MUX-AND, NOR-NOR and NAND-NAND. More examples are shown in Figures 4.8, 4.9 and 4.10. Figure 4.8 is the schematic of two 4-bit even parity generators. Two 4-bit even parity generators contain six XOR gates shown in Figure 4.8 (a). In the proposed version, g_1 and g_2 are grouped together; g_3 and g_4 are grouped together; and finally g_5 and g_6 are grouped together. Each group are performed by the proposed XOR-XOR pair in Figure 4.8(b). Another two examples are two binary comparators in Figure 4.9 and a 3-line to 8-line decoder Figure 4.10. Generally, the proposed method involves the construction of CPMRF components. The functions of these components correspond to the logic operations of the non-MRF ones. The proposed shared MRF network in each CPMRF component won't change the logic operations.

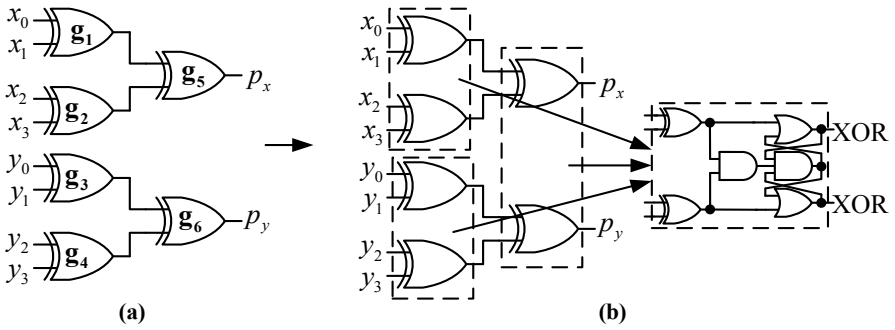


Figure 4.8:Two 4-bit even parity generators: (a) non-MRF version; (b) the proposed version.

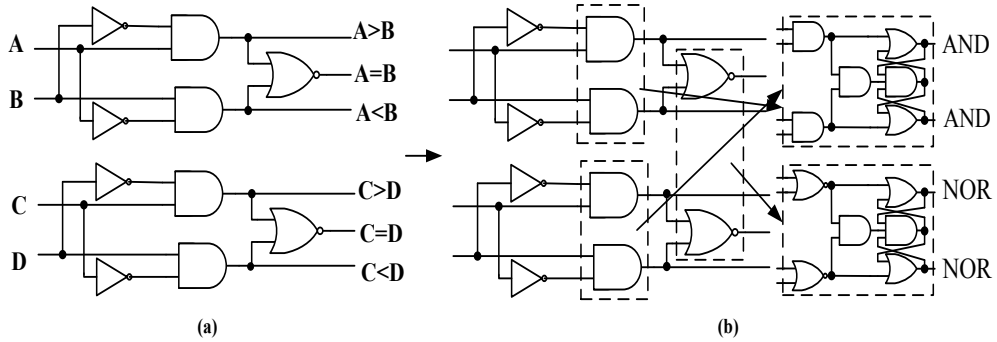


Figure 4.9: Two binary comparators: (a) non-MRF version; (b) the proposed version.

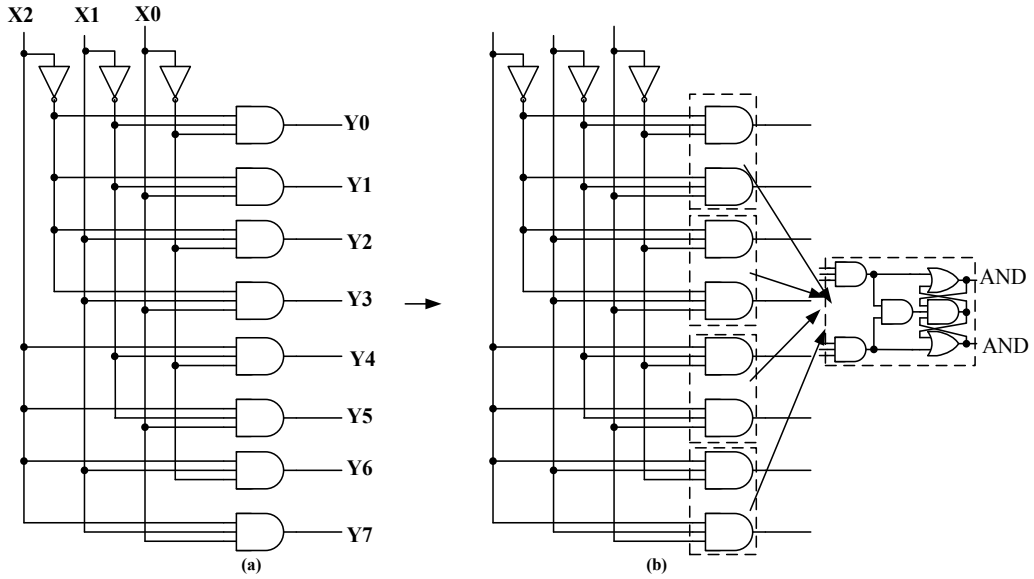


Figure 4.10: A 3-line to 8-line decoder: (a) non-MRF version; (b) the proposed version.

Table 4.2: Transistor Count Comparison

	MRF	MS	MRF-Schmitt	Proposed
AND-NOR	120	56	64	46
AND-XOR	120	82	86	46
MUX-AND	242	114	94	76
NOR-NOR	120	56	64	34
NAND-NAND	120	56	64	34
Two 4-bit even parity generators	360	324	276	174
Two binary comparators	368	88	216	118

A 3-line to 8-line decoder	550	294	326	174

By counting the number of the required transistors, we can know how much area can be reduced in the logic groups. The transistor numbers are counted in Table 4.2, while the transistor count reductions are shown in Table 4.3. In Table 4.2, the direct MRF mapping method [33] for each group requires many more transistors than the others. The average reduction of the required transistors is 65.4%, compared with the direct MRF mapping method [33], while it saves on average 36.7% transistors compared with the MS [38]. Compared with the MRF-Schmitt [40], the average reduction is 39.5%.

Table 4.3: Transistor Count Reduction

	MRF	MS	MRF-Schmitt	Proposed
AND-NOR	61.7%	17.9%	28%	-
AND-XOR	61.7%	43.9%	46.5%	-
MUX-AND	68.6%	33.3%	19.1%	-
NOR-NOR	71.7%	39.3%	46.9%	-
NAND-NAND	71.7%	39.3%	46.9%	-
Two 4-bit even parity generators	51.6%	46.2%	36.9%	-
Two binary comparators	67.9%	33.0%	45.4%	-
A 3-line to 8-line decoder	68.4%	40.8%	46.6%	-

*Reduction = (other structure-proposed)/other structure

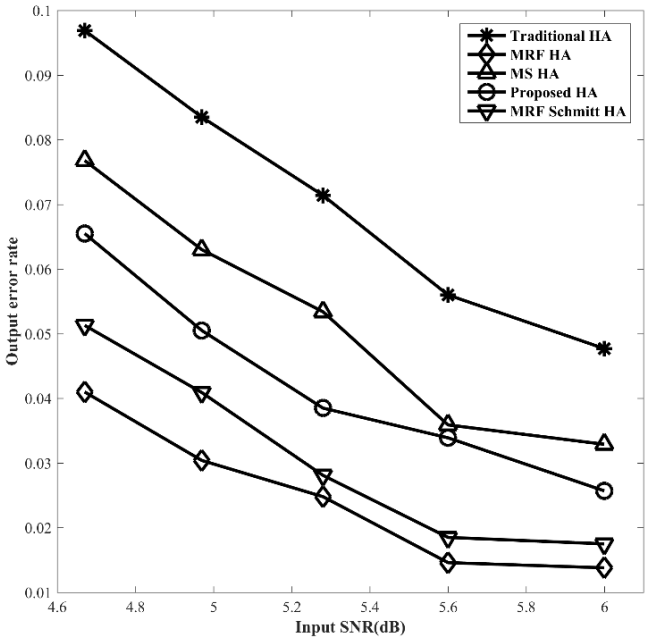


Figure 4.11: BER results for the XOR output in a half adder.

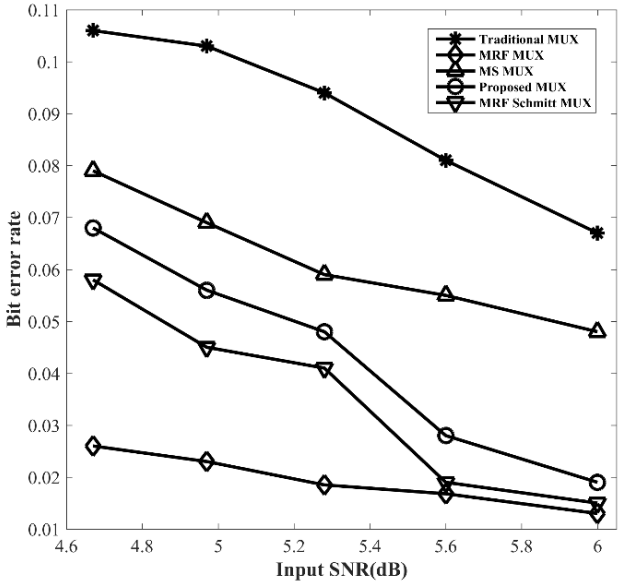


Figure 4.12: BER results for a MUX.

The performance of the error-tolerance against noise is investigated for the XOR output in the half adder shown in Figure 4.4 and the MUX output in the

MUX-AND group shown in Figure 4.5 as examples. The performance is measured by the bit error rate (BER) under the injection of noise with different Signal-to-noise ratios (SNR). We used the 130-nm BSIM4 library under a supply voltage of 0.25 V. Additive white Gaussian noise (AWGN) was generated every 1 ns by MATLAB and it was injected to the input noise-free signals. From 1 ns to 1000 ns, for 2-MHz input signals, the output signals were sampled about 15000 times (sampling rate≈15 GHz). Error rates were calculated in MATLAB from .tr0 file. Figure 4.11 and Figure 4.12 give the BER results of the non-MRF mapping XOR/MUX, the direct MRF mapping [33] XOR/MUX, the MS mapping [38] XOR/MUX, the MRF-Schmitt mapping [40] XOR/MUX and the proposed CPMRF mapping XOR/MUX. Compared with the MS mapping XOR, the number of errors is reduced by 18% in the proposed CPMRF design. The reduction increases to 41% when it is compared with the non-MRF version. Compared with the MS mapping MUX, the proposed CPMRF mapping MUX achieves a 32% average reduction in the BER. Errors in the non-MRF version are 54% higher than the proposed one. The above results correspond to the analysis of the general mapping process of the CPMRF mapping method in Section 4.1.

In general, the CPMRF mapping method can be summarized into two major steps:

- Design a common MRF network for a logic group based on the combination of the partial clique energy functions of each gate.
- Design a coding structure to complement the losing clique energy terms and prevent double-weakened cases.

These two steps consider the internal valid states which are used in the MS mapping method as well. Moreover, the coding unit in the proposed CPMRF method considers the asymmetric distribution of bits “1” and “0” in asymmetric logic gates to transform weak bits into strong bits to avoid double-weakened cases. This is what the MS mapping method neglects. The BER results comfirm that the proposed method can improve the performance in the presence of noise. Since two gates share a common MRF network in the proposed method, it costs less hardware than the MS mapping method. In Figure 4.11, the performance of the direct MRF mapping [33] design is the best with respect to BER under different SNRs, while the performance of the MRF-Schmitt mapping [40] is the second-best. The MRF-Schmitt mapping [40] is better than the proposed design because it adds classical Schmitt trigger inverters to resist the effect of noise. The idea of the Schmitt trigger inverter is different with MRF-based designs. In a Schmitt trigger inverter, there is a feedback inverter to regenerate the input signal with hysteresis to tolerate noise. The slight time difference indicates the Schmitt trigger inverter is based on time redundancy. However, MRF designs are based on probabilistic networks which result in space redundancy. In other words, noise immunity of MRF designs is at a cost of space redundancy. The drawback of using Schmitt trigger circuits is the increase in the power consumption and time delay. The other MRF mapping methods all consume more transistors than the porposed design. Regardless of whether it is a simple or complex design, the general methodology for mapping fundamental logic gates to their corresponding CPMRF logic groups saves area by sharing the same coding-based MRF network. Compared with the

direct MRF mapping method, the MS mapping method and the MRF-Schmitt [40] method, the proposed CPMRF method has a more cost-effective trade-off between hardware cost and the performance of the error tolerance against noise.

The functional verification of the proposed design can be achieved by simply comparing the outputs of the traditional non-MRF auto-transformation design with the outputs of the CPMRF auto-transformation design. The comparisons can be performed by XOR gates. When the outputs of these XOR gates are all "0", it indicates that the proposed CPMRF design has passed the functional verification. Consider the structure in Figure 4.2 as an example. Figure 4.13 is the schematic diagram for the functional verification. Two components should be built, including a non-MRF version formed by traditional logic gates (CMOS_AND_NOR.vhd in the Appendix) and a proposed version shown in Figure 4. 13 (proposed_AND_NOR.vhd in the Appendix). We use XOR gates to compare the outputs of the two components, shown in the functional_AND_NOR.vhd in the Appendix. Second, we should write the testbench code (functional_AND_NOR_tb.vhd in the Appendix) to perform a behavioral simulation to do the functional verification. Then, we run the simulation. If the outputs (test_AND and test_NOR) are both "0", as shown in Figure 4.14, then the outputs of the proposed version are the same as those of the non-MRF version. This indicates that the proposed structure passes the functional verification.

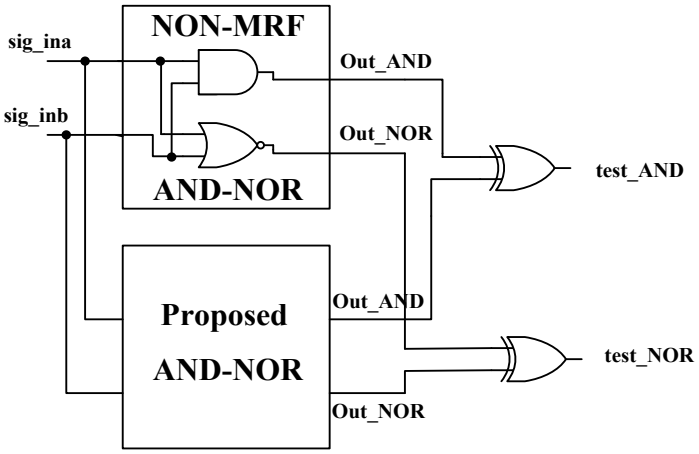


Figure 4.13: Schematic diagram for the functional verification.

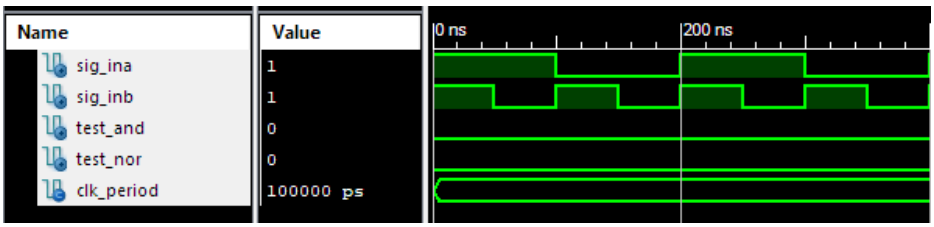


Figure 4.14: Simulation results.

4.3 Chip Implementation

Previous researchers in [36, 38] all chose carry-lookahead adders (CLA) to demonstrate their proof of concept. Similarly, we integrated the proposed CPMRF methodology into an 8-bit MUX-based CLA with IBM 130-nm technology. The tape-out service was offered by CMC Microsystems and manufactured by MOSIS. Figure 4.15 is the architecture of the 8-bit MUX-based CLA. We replaced the traditional non-MRF logic groups with the proposed CPMRF groups, including AND-XOR, AND-NOR, OR-XOR, OR-OR, AND-OR and XOR-XOR. These logic groups all have shared MRF networks. The layout in Cadence is shown in Figure 4.16, while Figure 4.17 is the silicon die photo.

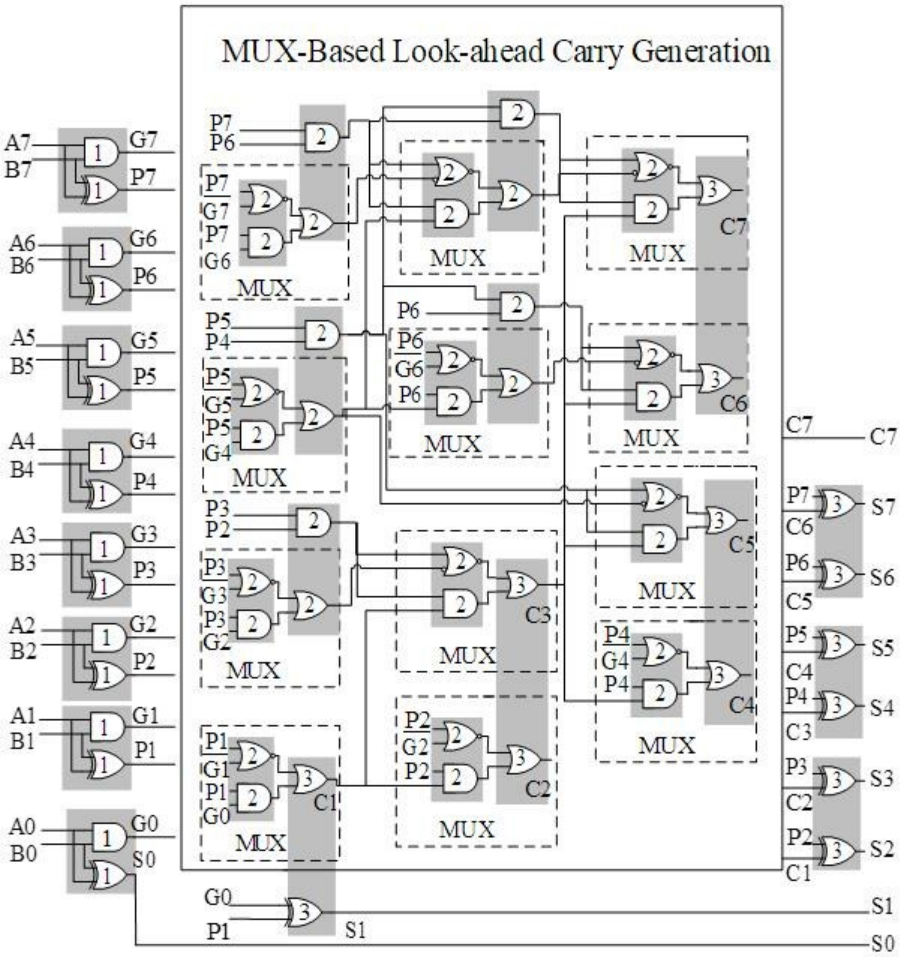


Figure 4.15: The architecture of the 8-bit MUX-based CLA.

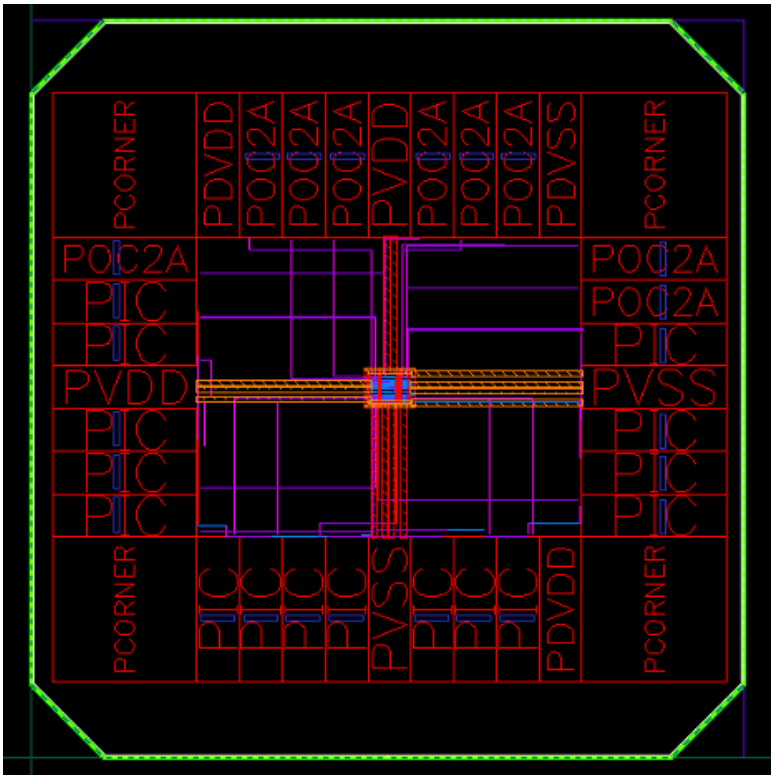


Figure 4.16: The layout in Cadence.

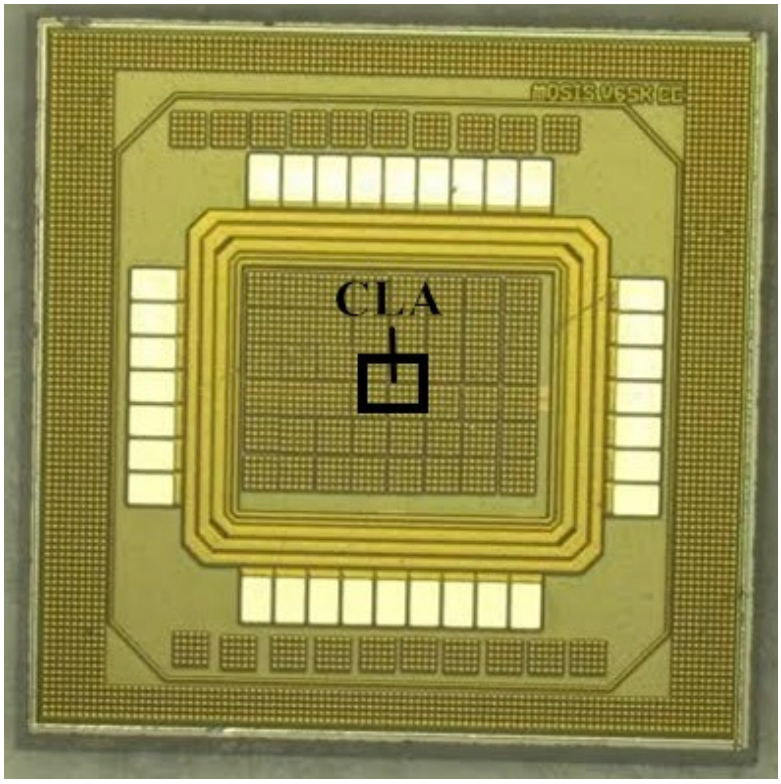


Figure 4.17: The die photo.

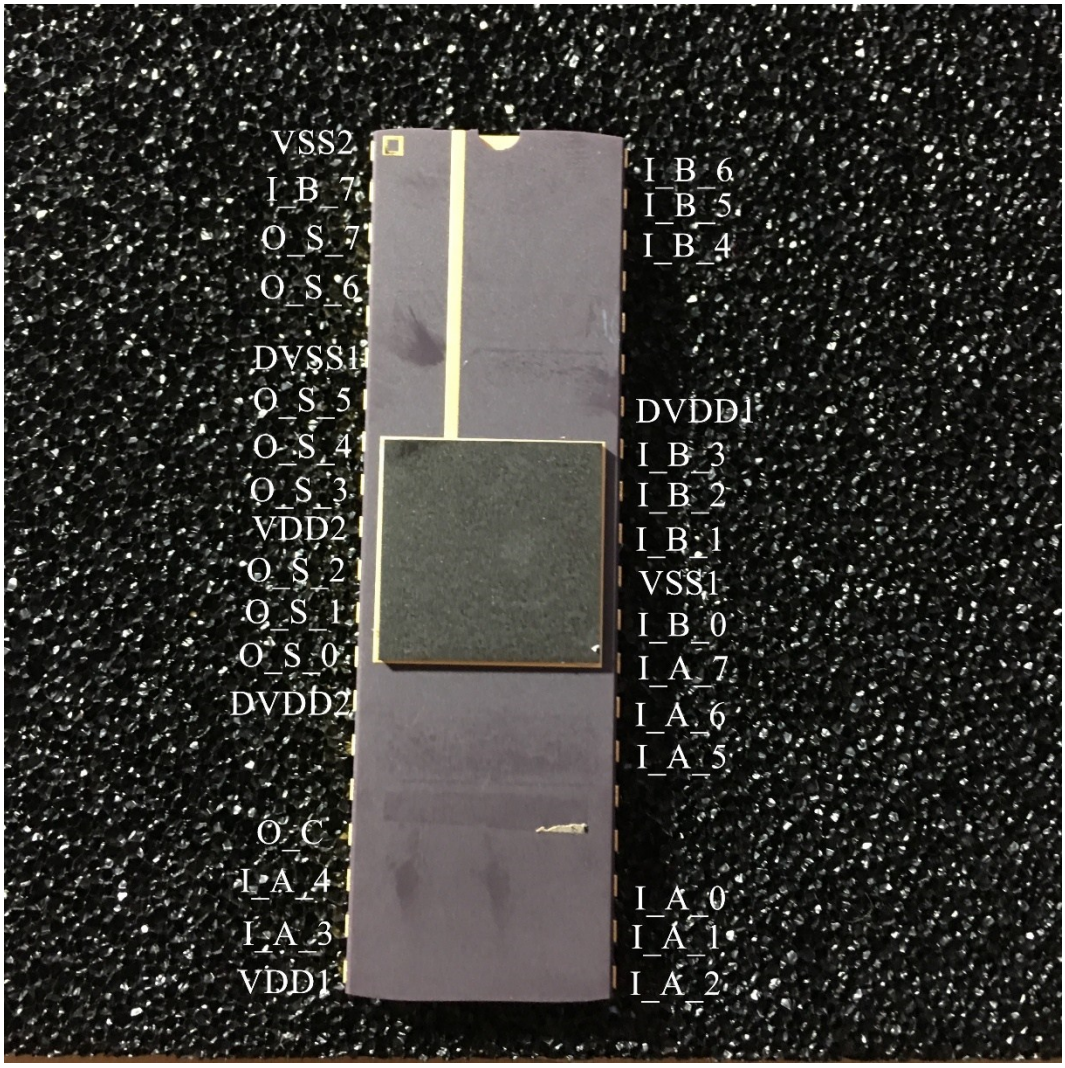


Figure 4.18: The chip with DIP40 package.

The package that we use is the 40-pin DIP shown in Figure 4.18. The pin information is labeled in Figure 4.18 as well. The input pins are I_A_0 to I_A_7 and I_B_0 to I_B_7, and the output pins are O_S_0 to O_S_7 and O_C. The power and ground pins are DVDD1, DVDD2, DVSS1 for pad and VDD1, VDD2, VSS1, VSS2 for core. IBM 130-nm CMRF8SF process is a technology with 1.2V core and 2.5V pad supply voltage. Figure 4.19 is the testing platform for noise tolerance. The instruments we used are listed in Table 4.4.

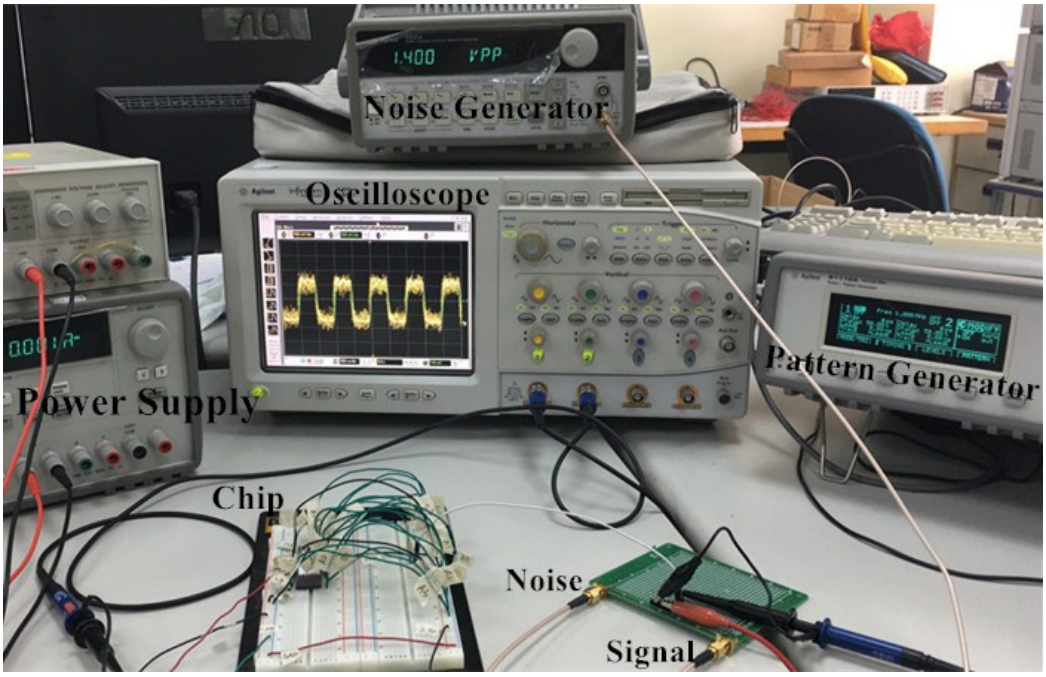


Figure 4.19: The testing platform.

Table 4.4: Testing Instruments

Instruments
Agilent 33120A Function/Arbitrary Waveform Generator
Agilent 81110A Pattern Generator
Agilent 54835A Digital Oscilloscope
Agilent E3630A Triple Output DC Power Supply
Agilent E3631A Triple Output DC Power Supply

Before we tested the noise-tolerance performance, we first tested the functional correctness of the chip. The testing circuit was connected as shown in Figure 4.20. We tried different input patterns from “00000000” to “11111111” for both input bits I_A and I_B to confirm it can work correctly. Figure 4.21 is an example. I_B_0 to I_B_7 were all set to “1”, while only I_A_0 was set to “1”. The O_C was “1” as shown in Figure 4.21.

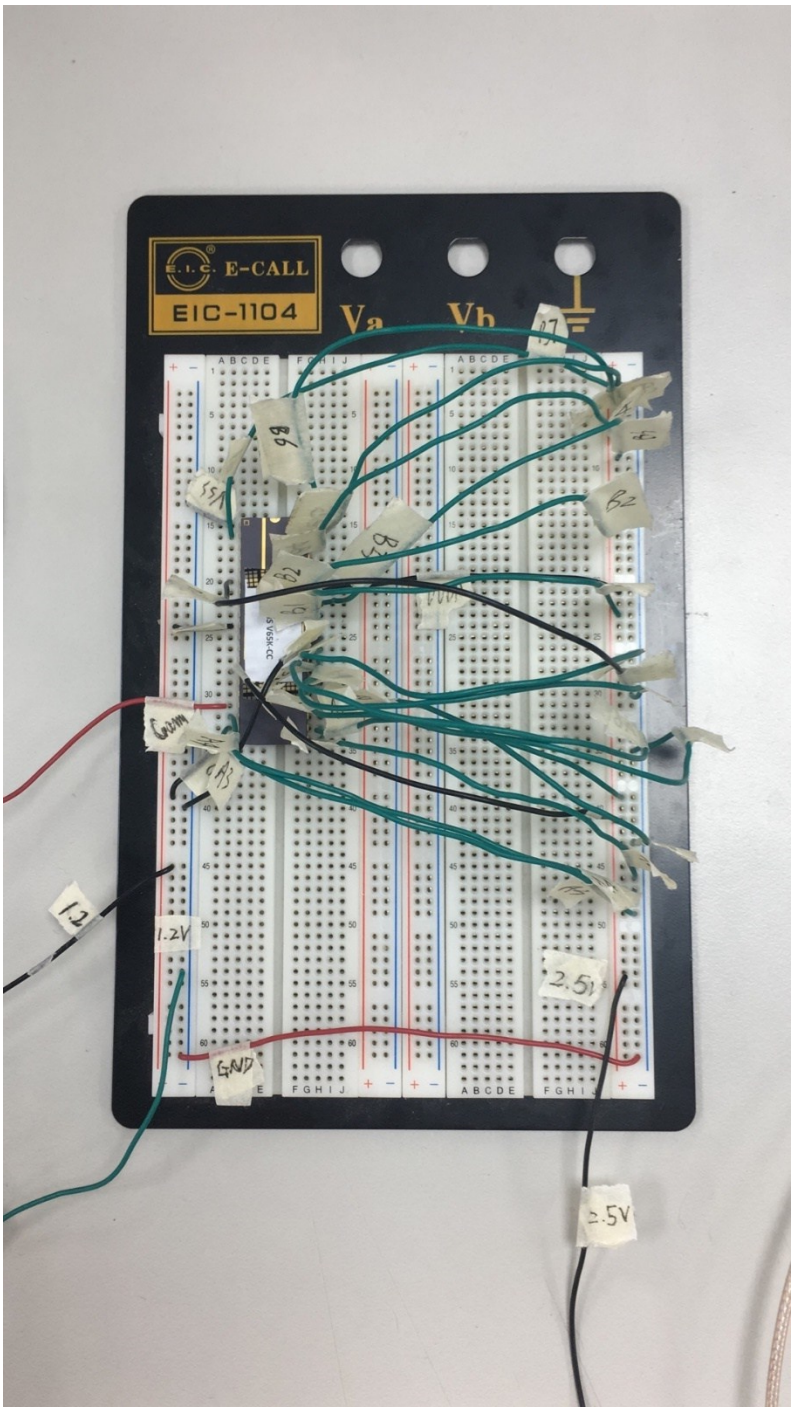


Figure 4.20: The testing circuit.

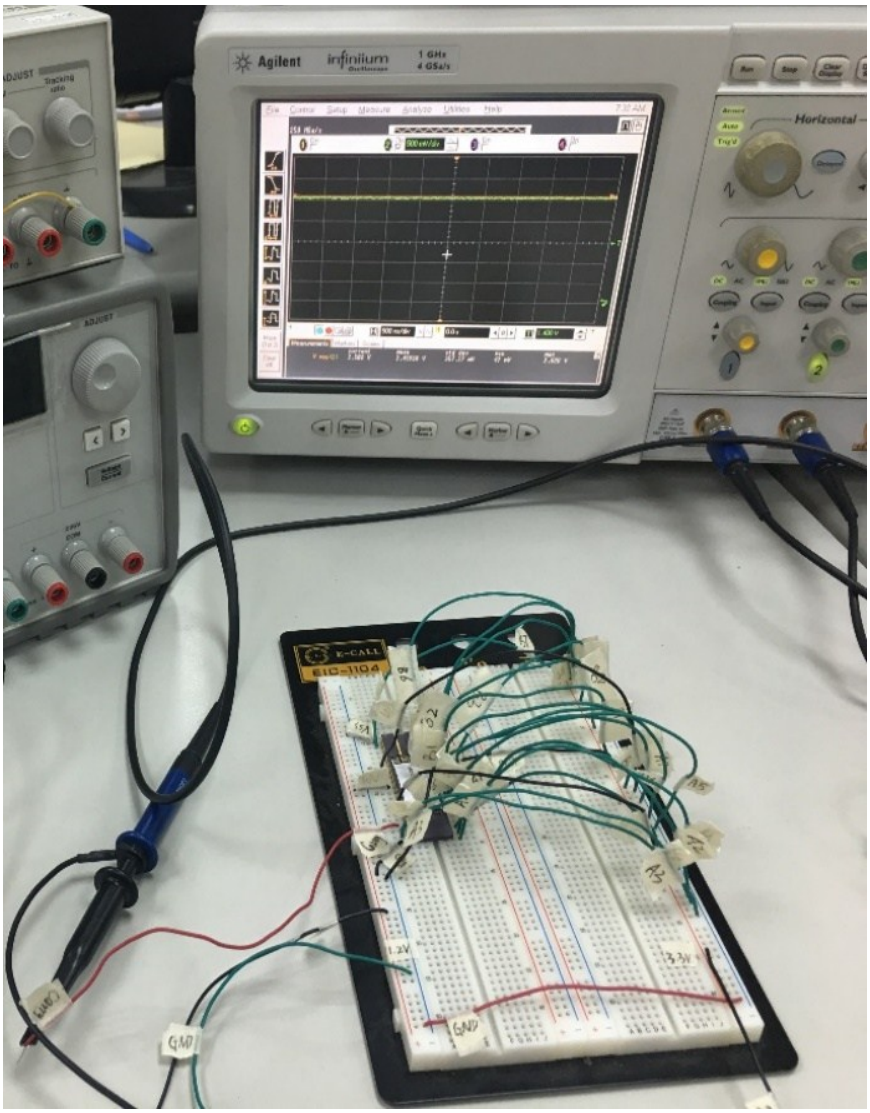


Figure 4.21: The functional test set-up.

For BER measurements, we injected the noise source through Agilent 33120A Function/Arbitrary Waveform Generator by regulating noise V_{pp} from 1.4 V to 0.95 V to the input signal by a combiner as shown in Figure 4.19. The input and output signals are shown in Figure 4.22. The input peak variance is improved by the chip as reflected by the small output variance. Figure 4.23 and Figure 4.24 are the input and output eye pattern diagrams under the interference of 6.02 dB

SNR noise. The quality of the output signal is much better than that of the input signals. It shows that the chip can tolerate some noise.

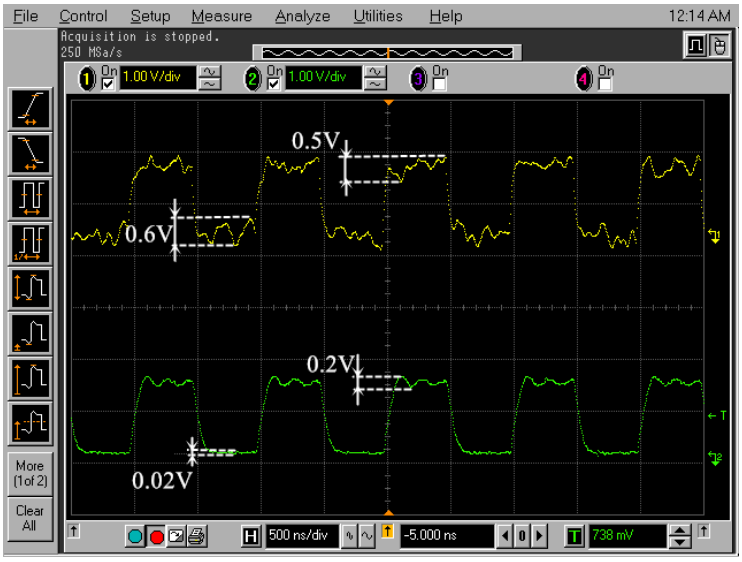


Figure 4.22: Input (upper) and output (lower) signals under 6.02 dB SNR.

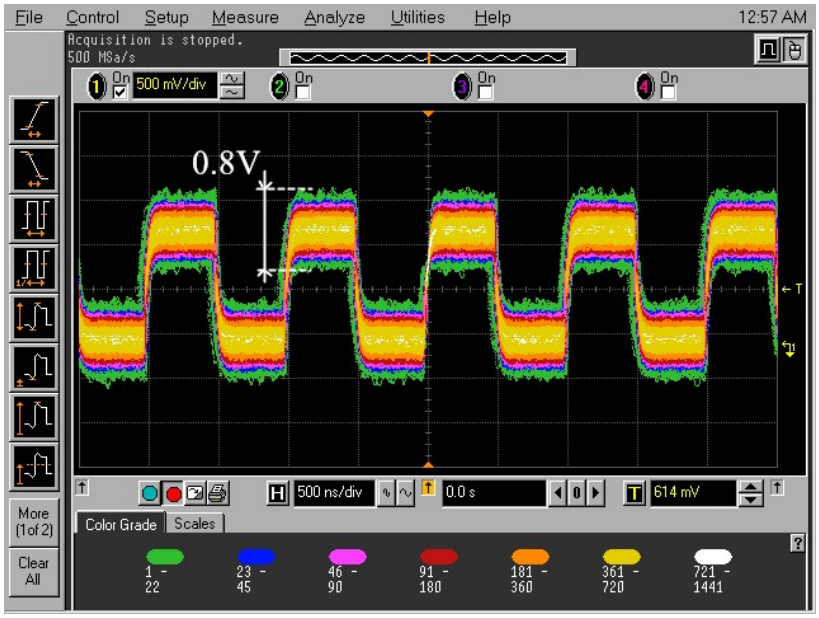


Figure 4.23: Input eye pattern under 6.02 dB SNR.

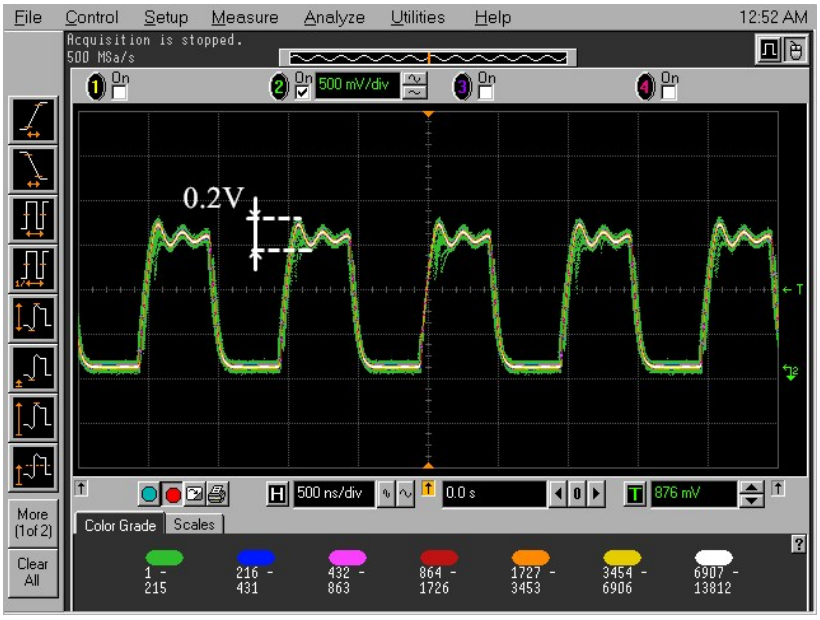


Figure 4.24: Output eye pattern under 6.02dB SNR.

We also measured some BER results shown in Figure 4.25. Since other researchers [36, 38, 40] have presented measured results, all comparisons and analysis based on the other designs in this section are refereed and compared. Table 4.5 is the performance comparisons based on previous results in [38]. Compared with the MS CLA [38], the proposed CLA saves 37.7% hardware cost while it reduces about 20% BER. In Figure 4.25, the direct-mapping MRF CLA [33] can tolerate the most of noise with the lowest BER. It has 28% lower BER than the proposed CLA. However, it consumes 64.4% more area as a trade-off. Similarly, although the MRF-Schmitt CLA [40] achieves 29% lower BER, it requires 35.1% more area. These findings are consistent with the conclusions presented in [36, 38, 40]. Therefore, the proposed design has relatively good noise tolerance with less hardware cost and less power consumption.

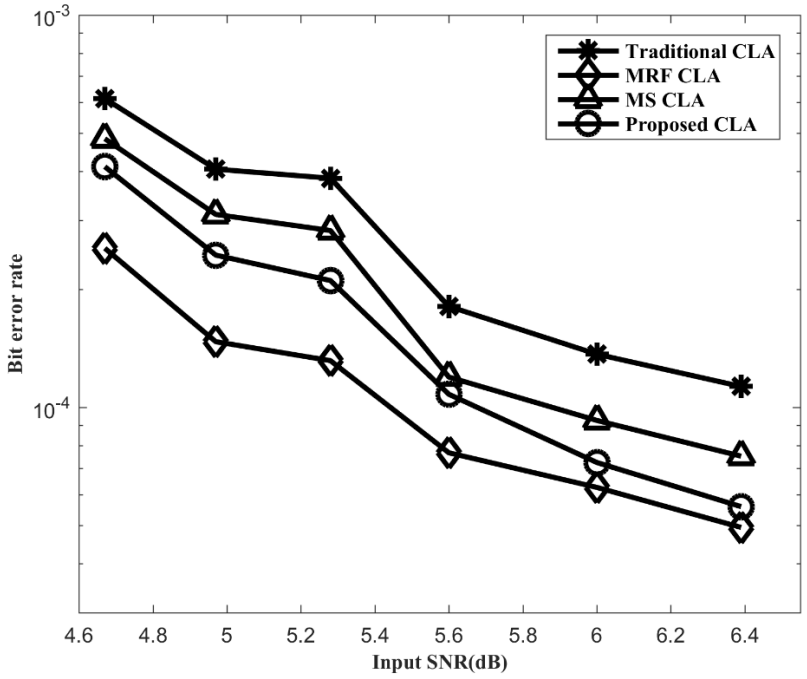


Figure 4.25: BER results.

Table 4.5: Performance Comparisons

Design	Proposed	MS [38]
Process	IBM130nm	TSMC130nm
Area	84.8um×67.6 um	140um×140 um
Power Consumption	125nW/MHz@0.25V 540nW/MHz@0.9V 700nW/MHz@1.0V 880nW/MHz@1.1V 1080nW/MHz@1.2V	1.9uW/MHz@0.25V 84uW/MHz@1.2V
BER	4.11e-04@4.68dB 7.24e-05@6.02dB 4.77e-07@8.05dB	7.00e-5@10.6dB 1.21e-05@11.5dB 1.25e-08@13.7dB

4.4 Conclusion

The MRF theory offers logic gates a way to tolerate errors generated from noise. For modern digital VLSI, it effectively ameliorates one of the major

concerns, the potential loss in reliability, as logic circuits can work correctly under a low supply voltage without the interference of random noise. However, MRF-based logic circuits sacrifice simplicity in terms of hardware cost. The direct mapping method trades off at least 10 times area for the capability of noise tolerance. Cost-effectiveness becomes the main concern for researchers. Based on the previous idea of circuit overhead sharing, we proposed a general simplified mapping method, CPMRF, to solve the above problem. Logic gates are separated into two categories: asymmetric gates and symmetric gates based on their output distribution. Asymmetric gates have the unequal capability of noise tolerance with respect to bit "0" and "1". In addition, logic gates are combined into complementary groups and non-complementary groups. The criterion is defined by whether the logic gates in a group can complement clique energy terms for each other. The shared MRF networks for these complementary groups should be built on the clique energy functions and the noise-tolerance characteristics of the asymmetric gates. Unlike complementary groups, non-complementary groups could only consider the noise-tolerant characteristics of asymmetric gates. They could only rely on coding structures to transfer a weak bit to a strong one. The effectiveness of the proposed CPMRF mapping method is demonstrated in silicon with an 8-bit carry-lookahead adder fabricated with IBM 130-nm technology. The measurement results show that the proposed CPMRF CLA achieves 20% improvement of noise-tolerance and 37.3% of area savings, compared with the MS CLA. It also costs 93% less power consumption than the MS one. Although, the traditional MRF CLA and the MRF-Schmitt CLA have less BER than the

proposed one, they both require much more area. In conclusion, the proposed CPMRF method is more cost-effective than the previous MRF designs. It achieves a relatively high noise tolerance capability without a large cost of area and power consumption.

Chapter 5

Probabilistic-Based Complementary Logic Gate Design

5.1 Noise-interfered Logic Gate Behavior Analysis

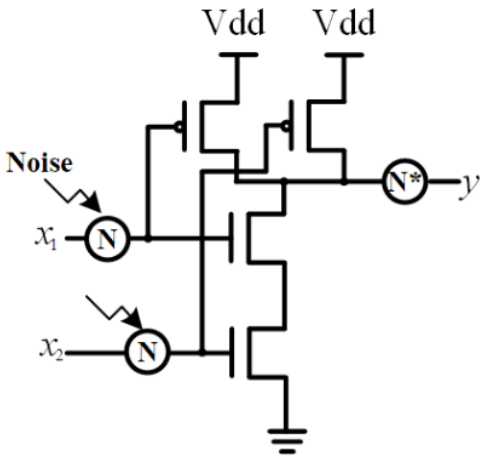


Figure 5.1: An analysis model for a NAND gate.

In the presence of noise, the output signal of a logic gate contains transitions from low to high or from high to low. It is necessary to investigate the distribution of the transitions from the perspective of probability. Let us use a NAND gate as an example. Figure 5.1 is a simple analysis model for a NAND gate. Dynamic noise is injected into the input signals x_1 and x_2 marked by a circle with a big letter N. It leads to the noisy output signal y whose noise is marked by a circle with N*. Denote the correct probability of the input signal as P_{in} when it is interfered by the noise. The output probability can be calculated by P_{in} as shown in Table 5.1. Assume that the occurrence probability of the four input sets are the same. The correct probability of the output signal being "1" is shown in Equation (5.1).

Table 5.1: Probability Table for a Noisy NAND Gate

Input x_1 x_2	Output y	Probability
0 0	0	$(1-P_{in})^2$
	1	$P_{in}^2 + 2P_{in}(1-P_{in})$
0 1	0	$P_{in}(1-P_{in})$
	1	$P_{in}^2+P_{in}(1-P_{in})+(1-P_{in})^2$
1 0	0	$P_{in}(1-P_{in})$
	1	$P_{in}^2+P_{in}(1-P_{in})+(1-P_{in})^2$
1 1	0	P_{in}^2
	1	$2P_{in}(1-P_{in})+(1-P_{in})^2$

$$\begin{aligned} P_{y=1} &= \frac{1}{4}(P_{in}^2 + 2P_{in}(1-P_{in}) + 2(P_{in}^2+P_{in}(1-P_{in})+(1-P_{in})^2)) \\ &= \frac{1}{4}P_{in}^2 + \frac{1}{2} \end{aligned} \tag{5.1}$$

The correct probability of the output signal being "0" is shown in Equation (5.2).

$$P_{y=0} = \frac{1}{4}P_{in}^2 \tag{5.2}$$

For $P_{in} \in (0,1]$, we have

$$P_{y=1} > P_{y=0} \tag{5.3}$$

It can also be explained by the asymmetric distribution of the output values as a NAND gate will output "1" as long as one of the inputs is "0". Therefore, in the presence of noise, a NAND gate is more likely to output a correct bit "1".

Similarly, consider the probability table for a noisy NOR gate shown in Table 5.2. The correct probability of the output signal being "0" is the same with the correct probability of the output signal being "1" of a noisy NAND gate:

$$P_{y=0} = \frac{1}{4}(P_{in}^{\ 2} + 2P_{in}(1-P_{in}) + 2(P_{in}^{\ 2} + P_{in}(1-P_{in}) + (1-P_{in})^2)) \\ = \frac{1}{4}P_{in}^{\ 2} + \frac{1}{2} \tag{5.4}$$

The correct probability of the output signal being "1" is shown in Equation (5.5).

$$P_{y=1} = \frac{1}{4}P_{in}^{\ 2} \tag{5.5}$$

For $P_{in} \in (0,1]$, we have

$$P_{y=0} > P_{y=1} \tag{5.6}$$

Therefore, for a NOR gate: $P_{y=0} > P_{y=1}$.

Table 5.2: Probability Table for a Noisy NOR Gate

Input x_1 x_2	Output y	Probability
0 0	0	$2P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}^{\ 2}$
0 1	0	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}(1-P_{in})$
1 0	0	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}(1-P_{in})$
1 1	0	$P_{in}^{\ 2}+2P_{in}(1-P_{in})$
	1	$(1-P_{in})^2$

Table 5.3: Probability Table for a Noisy AND Gate

Input x_1 x_2	Output y	Probability
0 0	0	$P_{in}^{\ 2}+2P_{in}(1-P_{in})$
	1	$(1-P_{in})^2$
0 1	0	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}(1-P_{in})$
1 0	0	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}(1-P_{in})$
1 1	0	$2P_{in}(1-P_{in})+(1-P_{in})^2$
	1	$P_{in}^{\ 2}$

According to Table 5.3 and Table 5.4, we can deduce the relationship between $P_{y=0}$ and $P_{y=1}$ for an AND gate and an OR gate in the same way: 1) for an AND gate: $P_{y=0} > P_{y=1}$; 2) for an OR gate: $P_{y=1} > P_{y=0}$.

Table 5.4: Probability Table for a Noisy OR Gate

Input x_1 x_2	Output y	Probability
0 0	0	$P_{in}^{\ 2}$
	1	$2P_{in}(1-P_{in})+(1-P_{in})^2$
0 1	0	$P_{in}(1-P_{in})$
	1	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
1 0	0	$P_{in}(1-P_{in})$
	1	$P_{in}^{\ 2}+P_{in}(1-P_{in})+(1-P_{in})^2$
1 1	0	$(1-P_{in})^2$
	1	$P_{in}^{\ 2}+2P_{in}(1-P_{in})$

In order to demonstrate the mathematical analysis, we simulate the cases in HSPICE. Transistor models are from the 65-nm CMOS Berkeley library. We injected Gaussian noise into the input signals with zero mean and 170-mV standard deviation. The circuit is operated at a supply voltage 0.25V at 50℃ to inject thermal noise. Signal waves are presented in Figure 5.2 and Figure 5.3. We also load the HSPICE file in MATLAB to calculate the error rate for each logic gate. The results are shown in Table 5.5. Both Figure 5.2, Figure 5.3 and Table 5.5 correspond to the conclusions of the mathematic analysis. In conclusion, a noisy NAND gate and a noisy OR are more likely to output correct "1", defined to be a Robust "1", while a noisy AND gate and a noisy NOR are more likely to output correct "0" defined to be a Robust "0".

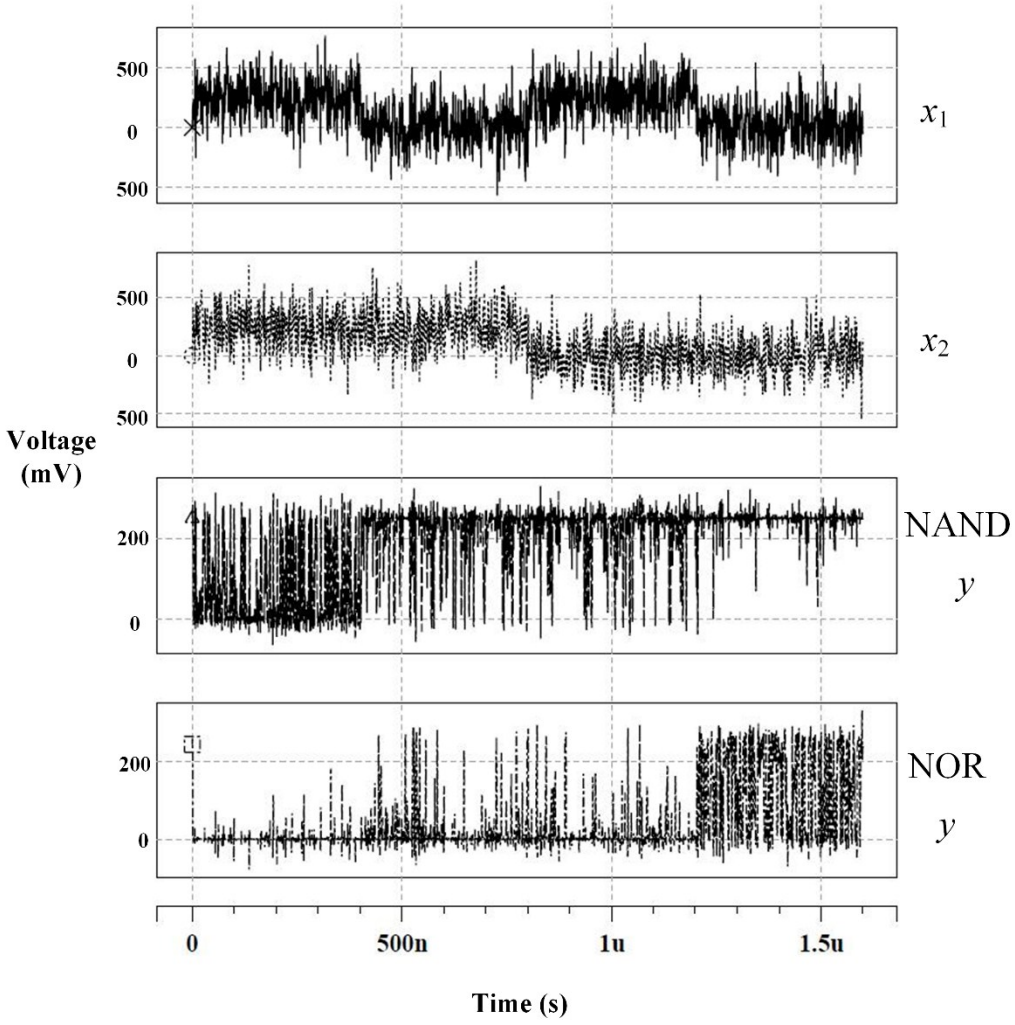


Figure 5.2: HSPICE simulation for NAND and NOR logic gates.

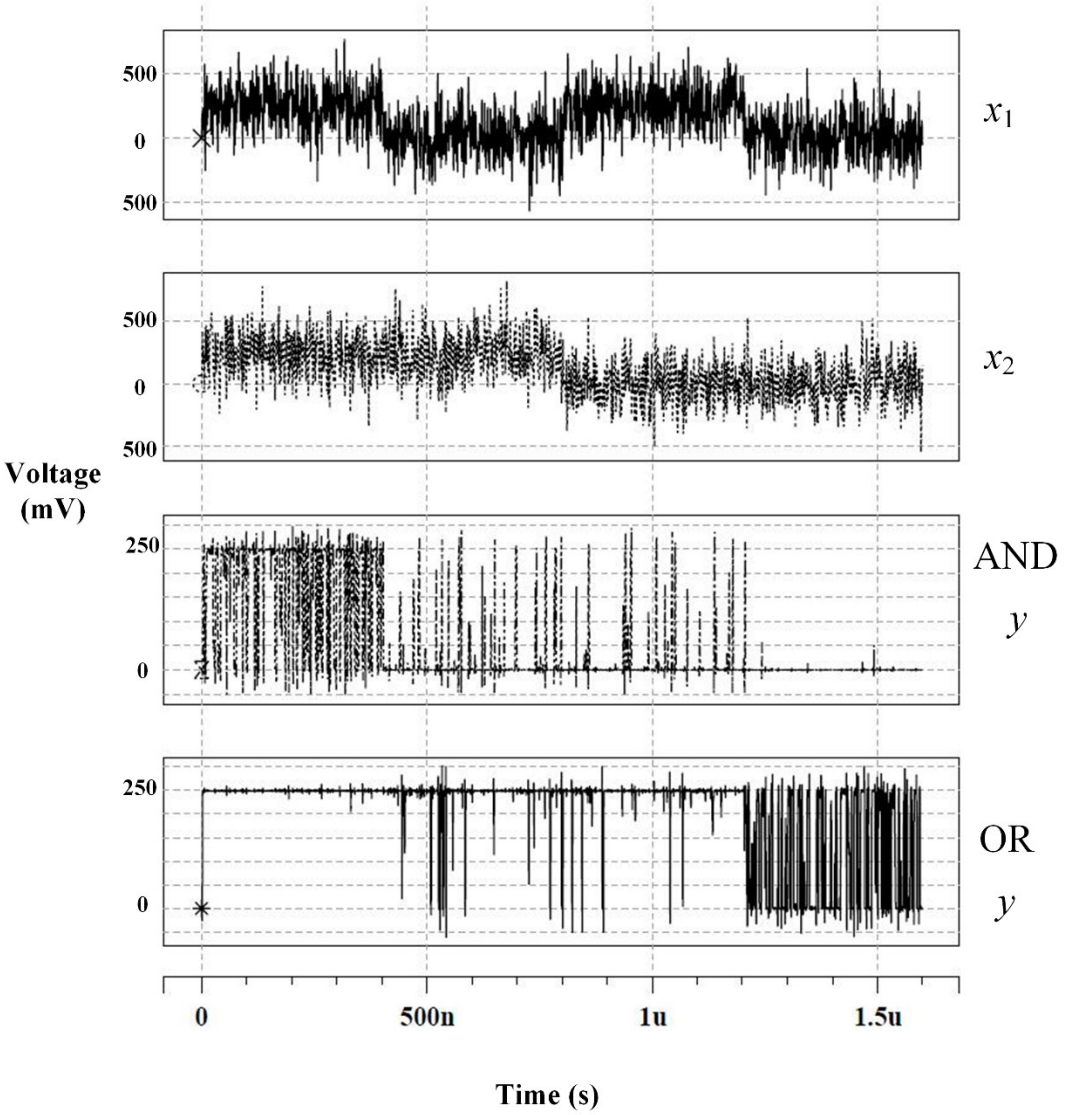


Figure 5.3: HSPICE simulation for AND and OR logic gates.

Table 5.5: Error Rate Results

Gate	Error rate	
	"0"	"1"
NAND	0.2804	0.0735
NOR	0.0324	0.4786
AND	0.2804	0.0351
OR	0.0249	0.4203

5.2 Probabilistic-Based Complementary Logic Gates

In Section 5.1, we demonstrated the properties of the outputs of a noisy NAND/NOR/AND/OR gate. We summarize the main conclusions in Table 5.6. Robust outputs are highlighted in grey. From Table 5.6, it is observed that 1) a NAND gate and an OR gate have the robust output "1" which is opposite to the robust output "0" of a NOR gate and an AND gate; 2) when the inputs x_1 x_2 are complementary "01" or "10", a NAND gate grouped with a NOR gate or an AND gate grouped with an OR gate only generate robust complementary outputs. Inspired from the observations, we can design a self-error-tolerant logic gate which have both robust output "1" and "0".

Table 5.6: The Truth Table for NAND, NOR, AND and OR

Inputs		Output y			
x_1	x_2	NAND	NOR	AND	OR
0	0	1	1	0	0
0	1	1	0	0	1
1	0	1	0	0	1
1	1	0	0	1	1

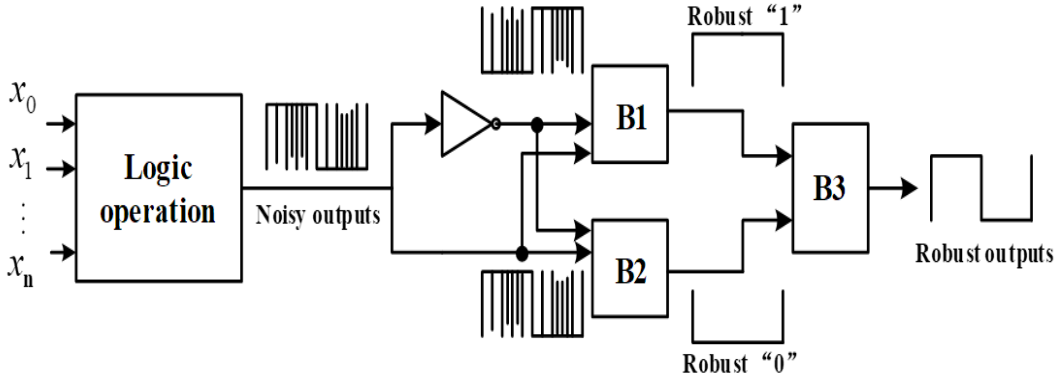


Figure 5.4: The idea of a self-error-tolerant logic gate.

Figure 5.4 shows the idea of a self-error-tolerant logic gate. The noisy outputs of a logic operation can be separated into two noisy complementary outputs by an inverter as the inputs of two blocks B1 and B2. B1 and B2 will separately generate robust "1" and robust "0". Finally, a block B3 will combine the robust "1" and robust "0" to form robust outputs for the logic operation. As a NAND gate grouped with a NOR gate or an AND gate grouped with an OR gate only generate robust complementary outputs when the inputs are complementary, B1 and B2 can be a NAND gate and a NOR gate/ an AND gate and an OR gate. In other words, a NAND gate can be paired with a NOR gate, while an AND gate can be paired with an OR gate. Next, we only need to find an element to choose the right robust outputs for the logic operation. A multiplexer (MUX) can play the role of B3 to finish the logic operation correctly. The selection input of the MUX can be the noisy output of the logic operation which will select the robust "1" or "0" to the output.

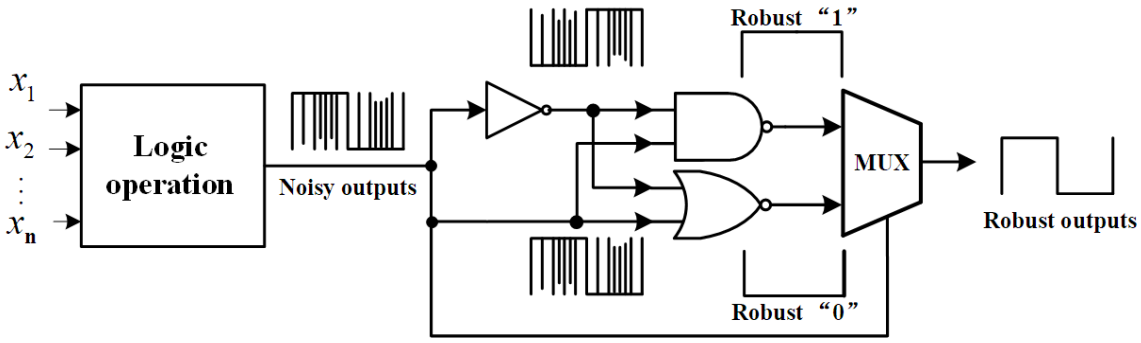


Figure 5.5: The detailed structure of a self-error-tolerant logic gate.

Figure 5.5 illustrates the detailed structure of a self-error-tolerant logic gate which is defined as a Probabilistic-based Complementary Logic (PCL) gate. An inverter, a NAND gate, a NOR gate and a MUX together form an error-tolerant

structure. The logic operation in Figure 5.4 can be replaced by logic gates. For example, a PCL NAND gate consist of a NANG gate and the error-tolerant structure shown in Figure 5.6. The NAND gate and the NOR gate can be replaced by an AND gate and an OR gate together. The error-tolerant structure used in the PCL gate design is easy to apply to a logic circuit by the EDA flow since it can be created as a basic cell in the library. In this way, it is flexible to use as a replicable component. The error-tolerant structure belongs to the technique of redundancy. It trades hardware cost for reliability. If hardware cost is of primary concern, the design can be selected to use in some critical parts. For example, the PCL gates can be used in the computing circuits which create MSB, while the circuits creating LSB are implemented using the normal logic gates without the error-tolerant structure.

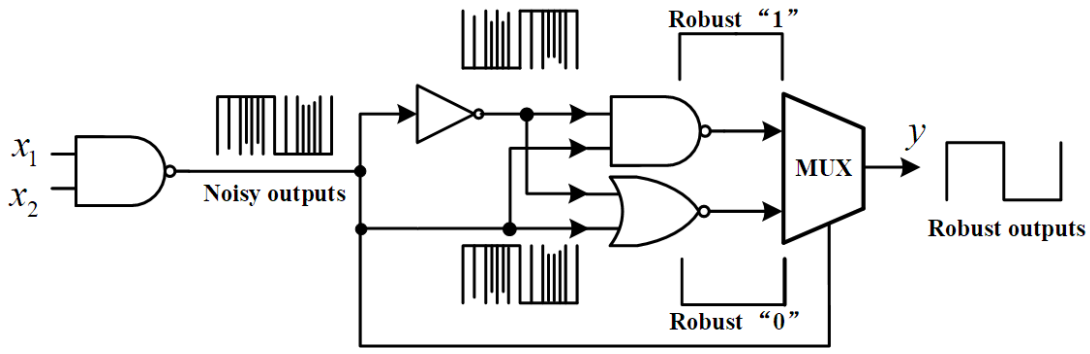


Figure 5.6: A PCL NAND gate.

5.3 Simulation and Discussion

Sections 5.1 and 5.2 introduce the idea of PCL gates theoretically. In this section, the simulation results of some designs based on PCL gates are shown to demonstrate their noise immunity. A noise source, random Gaussian noise, was injected to the input signals. The basic components are from 65-nm Berkeley library. The supply voltage was set to 0.25 V, while the simulation temperature was 50 °C. The sampling process was operated by HSPICE automatically using .tran command. From 1 ns to 800 ns, for 2.5-MHz input signals, it sampled the output signals about 19000 times (sampling rate ≈ 23 GHz). The error rates of the output signals were calculated by MATLAB.

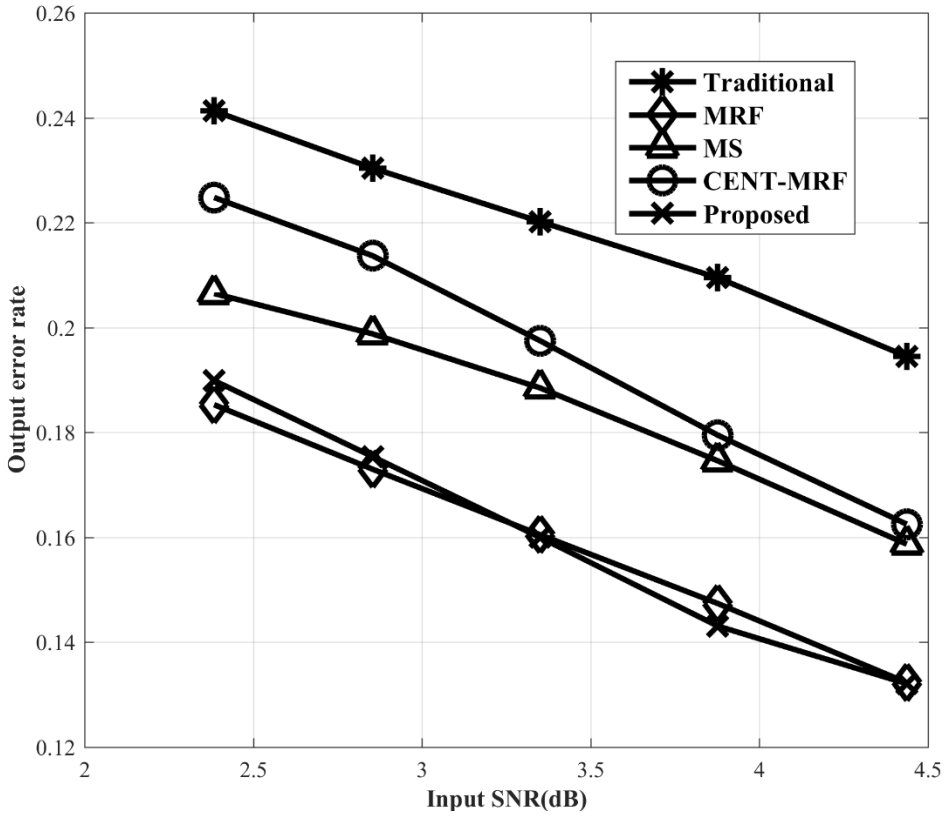


Figure 5.7: Simulation results of different NAND gate designs.

Figure 5.7 illustrates the ERs of different NAND gate designs, including Traditional (normal CMOS), MRF [33], MS [38], CENT-MRF [41] and the proposed PCL NAND design. Both the MRF-based NAND gate and the proposed PCL NAND gate have good noise-immunity performance. The CENT-MRF-based NAND gate is not as good as the MS one because CENT-MRF uses the simplified energy function based on the MS design. The proposed PCL NAND gate has 27.3% lower ER than the traditional NAND gate. Compared with the MS-based NAND gate, the output signal contains 13.9% fewer errors, while it has 18.3% fewer errors than the CENT-MRF-based design.

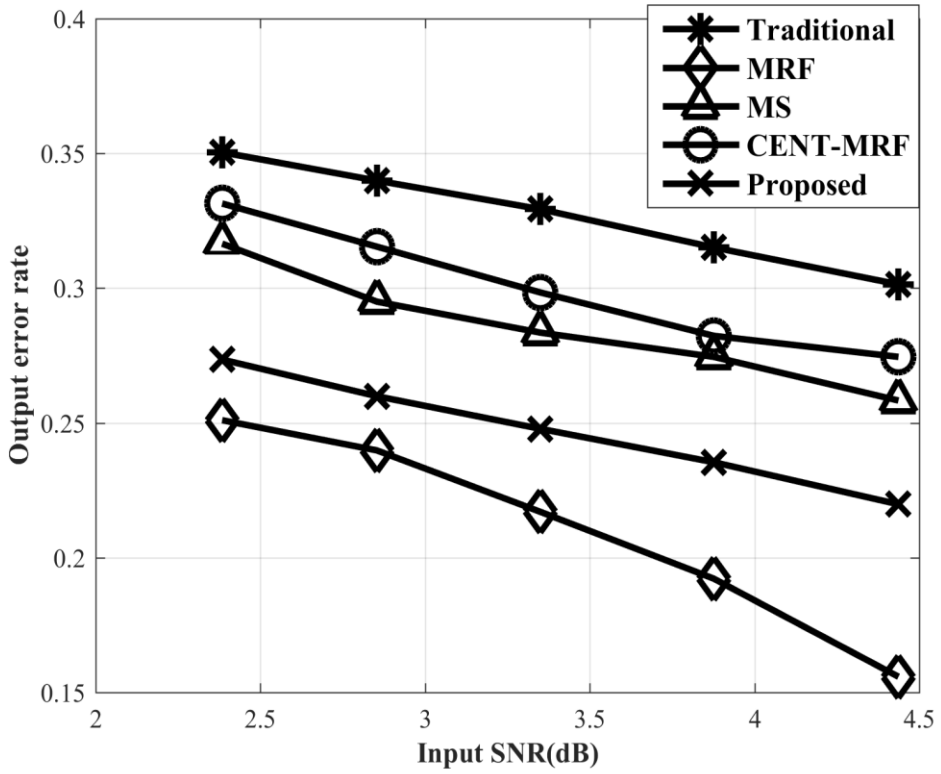


Figure 5.8: Simulation results of different XOR gate designs.

Figure 5.8 shows the ERs of different XOR gate designs, including Traditional (normal CMOS), MRF [33], MS [38], CENT-MRF [41] and the proposed PCL XOR design. For the XOR gates, the proposed PCL XOR gate cannot tolerate noise as well as the MRF-based XOR gate. This is because the output distribution of an XOR gate is symmetric. This limits the noise-immunity performance of the NAND gate and NOR gate in the PCL XOR design. The proposed PCL XOR gate has 24.5% lower ER than the traditional XOR gate. Compared with the MS-based XOR gate, the output signal contains 13.5% fewer errors, while it has 17.7% fewer errors than the CENT-MRF-based design.

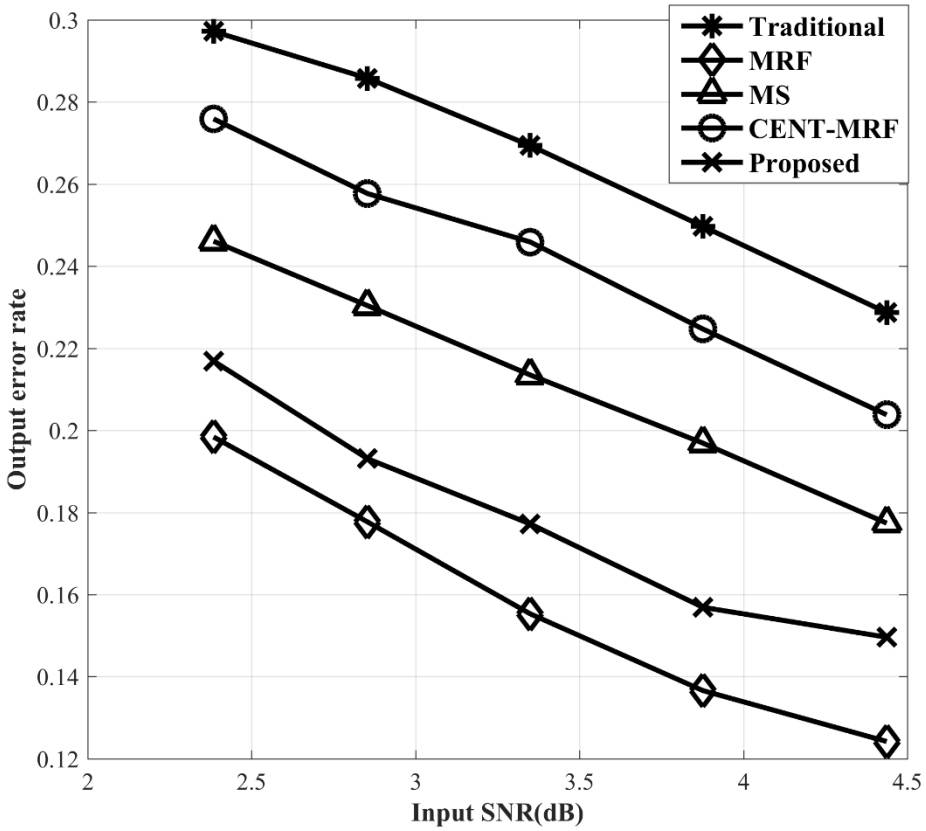


Figure 5.9: Simulation results of different MUX designs

We also simulated different MUX designs. The traditional MUX design was implemented with two NAND gates, an inverter and an OR gate. Other MUX designs correspond to the MRF [33] version, the MS [38] version, the CENT-MRF [41] version and the proposed PCL version. For example, the proposed PCL MUX was implemented by two PCL NAND gates, an inverter and an PCL OR gate. Figure 5.9 shows the ERs of different MUX designs. The proposed PCL MUX has 33.1% lower ER than the traditional MUX. Compared with the MS-based MUX gate, the output signal contains 16.2% fewer errors, while it has 26.2% fewer errors than the CENT-MRF-based design. The fact that the proposed PCL MUX gate is not as good as the single gate design (PCL NAND gate for example) may result from the property of the output signals. In a PCL design, each PCL gates contains a NAND gate and a NOR to filter noise from input signals. Therefore, the output signals of these PCL gates will have the similar noise distribution. It limits the noise-immunity performance of the PCL design.

Table 5.7: Transistor Counts of Different Designs

	NAND	XOR	MUX
Traditional	4	12	20
MRF [33]	60	60	202
MS [38]	28	46	92
CENT-MRF [41]	14	22	50
PCL	34	42	110

Table 5.7 calculates the transistor counts of different designs. It was found that PCL design saves at least 30% of the transistors compared with MRF-based designs. For the MUX design, it requires 16.3% more transistors while achieving

16.2% lower ER than MS design. Compared with the CENT-MRF MUX design, it has 26.2% lower ER at the cost of 54.5% more transistors.

5.4 Conclusion

When noise interferes with logic circuits, logic gates have the ability to tolerate errors to varying degrees. By building models for different logic gates, we showed that a NAND gate or an OR gate is more likely to output a correct "1" in the presence of noise, while an AND gate or a NOR gate is more likely to output a correct "0". The simulation results correspond to the theoretical analysis. On the other hand, if the input signals are complementary "01" or "10", a NAND gate can be paired with a NOR gate to generate robust complementary outputs as an AND gate pairs with an OR gate. Based on these findings, a kind of self-error-tolerant logic gates named PCL gates was designed to strengthen the ability of noise immunity. Each PCL gate contains a main logic gate to implement the target logic operation, a pair of complementary logic gates like a NAND gate and a NOR gate to generate robust complementary signals, and a MUX to control the robust final outputs. In the ER simulation, the PCL NAND gate has much lower ERs than the MS NAND gate and the CENT-MRF NAND gate. However, the noise immunity performance of a PCL XOR gate is not as good as that of the PCL NAND gate. When using PCL gates to implement logic functions, the noise-immunity performance is limited as the output signals of the PCL gates have similar noise distribution. Considering the transistor count, the PCL designs are not very cost-effective. The ER of the PCL MUX is 26.2% lower than that of the CENT-MRF MUX design at the cost of 54.5% more transistors. In future research,

the potential of the PCL gate design could be explored by optimizing the current structure.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The increasing chip density in modern VLSI designs drives researchers to become more concerned with power consumption. Dynamic power consumption as the main source of power consumption is a quadratic function of the supply voltage. Low-power operation becomes a good choice to reduce the dynamic power consumption to save the overall power consumption. However, when the lowered supply voltage is close to the threshold voltage, errors are more likely to occur in the circuit in the presence of noise. Low-power operation requires more attention to be paid to reliability. Meanwhile, we still need to consider hardware cost as price-sensitive consumer portable devices are popular and attractive in today's market. Therefore, hardware cost, reliability and power consumption are three key points in modern VLSI designs. As trade-offs always exist, how to keep a better balance among the three key priorities becomes an important topic. In this thesis, some low-power error-tolerant gate-level and system-level designs are proposed, including MRF-based CDMR design, DCT implementation based on MRF-stochastic logic, CPMRF circuit design and PCL gate design.

MRF-based CDMR combines MRF theory with DMR. Unlike the traditional DMR, the modules used by CDMR have complementary outputs which means one is the inverted version of the other. MRF theory is involved in the design of the two-stage voting circuit. The first stage contains an MRF-based feedback circuit. The second stage can work like a latch to compensate for the loss of the

error tolerance in the first stage. Since it only requires one redundant module, hardware cost can be reduced with relatively low power consumption and high error tolerance.

DCT implementation based on MRF-stochastic logic uses stochastic logic implemented by MRF-based circuits to implement the algorithm for a 1D DCT. Logic gates in stochastic adders and multipliers are paired in groups to share common MRF networks to achieve area saving. When injecting the same noise source, the proposed design can tolerate more noise with low hardware cost and low power consumption compared with the MS-stochastic-based 1D DCT.

CPMRF circuit design offers a way to simplify the traditional MRF-based circuit design by following a general mapping rule for logic operations. Logic gates are separated into complementary pairs and non-complementary pairs. For a complementary pair, a shared MRF network can be designed based on PMRF energy terms. For a non-complementary pair, asymmetric gates are used to compensate for the loss of noise immunity in bit "1" and "0". The coding structure compensates for the loss of the energy and strengthens the stability of the logic "1" and "0" states. An 8-bit CLA is taped out to measure the performance of the proposed method. It cost less hardware and power consumption with high noise immunity than the MS-based version.

Finally, PCL gate design investigates the property of asymmetric gates in terms of the robustness of logic "1" and "0" based on the perspective of probability. It takes the advantage of asymmetric gates to generate robust "1" and

"0" signals by inputting complementary signals. In each PC logic gate, a MUX is used to choose robust bits as the final outputs. The proposed self-error-tolerant component is compatible with any logic gates. It helps logic gates to tolerate errors from noise. Since the self-error-tolerant component is a fixed design, any logic gates can be transformed to PC versions automatically within the existing Electronic design automation (EDA) flows.

6.2 Future Work

Low-power error-tolerant digital circuit design is an interesting field to investigate. Although my current research takes me four years to have some accomplishments, it still has a long way to work on.

First, the current designs still have some potential to exploit. For example, MRF-based CDMR rely on the single-error assumption that errors won't occur at the same time in the complementary propagation chains. In Chapter 2, we point out that a delay unit can be added in one of the input branches to tolerate errors occurring at the outputs of the two modules at the same time. Future research can be carried out toward the finding of a suitable delay unit. Meanwhile, the current two-stage voting circuit may need to be transformed into other versions. Another example can be an optimized design of the PC logic gates. Chapter 5 indicates that the current PC logic gates might not be that cost-effective as they still require more transistors than MS and CENT-MRF designs. It may possible to improve the PC logic design by combining other techniques.

On the other hand, low-power error-tolerant digital circuit designs can be extended to a broad area. The current designs are limited to pure combinational circuits. In future work, it would be interesting to research on low-power error-tolerant sequential circuits. We can investigate whether it is possible to integrate MRF theory with sequential circuit designs. In addition, we can also consider the application of the current designs to other applications. We have successfully applied MRF theory on the implementation of an 1D DCT system in Chapter 4. However, there are other computing systems which can be combined with the idea of MRF, MRF-based CDMR, MRF-stochastic logic, CPMRF and PCL. The MRF-based circuit designs are able to correct errors intelligently by using output signals and valid states. This inspires us that it may have potentials to implement other AI algorithms with the advantages of low power consumption and high error tolerance.

References

[1] J. S. Kilby, "Invention of the integrated circuit," in *IEEE Transactions on Electron Devices*, vol. 23, no. 7, pp. 648-654, July 1976.

[2] P. K. Bondyopadhyay, "Moore's law governs the silicon revolution," in *Proceedings of the IEEE*, vol. 86, no. 1, pp. 78-81, Jan. 1998.

[3] B. Nikolic, "Design in the Power-Limited Scaling Regime," in *IEEE Transactions on Electron Devices*, vol. 55, no. 1, pp. 71-83, Jan. 2008.

[4] Ki Won Lee, "Low power requirements for future digital life style," *Proceedings of the 2003 International Symposium on Low Power Electronics and Design, 2003, ISLPED '03*, Seoul, South Korea, 2003, pp. 1-.

[5] Instruments, T. (1997), CMOS power consumption and CPD calculation, *SCAA035B June*.

[6] Meng, Yan, Timothy Sherwood, and Ryan Kastner, "On the limits of leakage power reduction in caches," *High-Performance Computer Architecture, 2005, HPCA-11, 11th International Symposium on* IEEE, 2005.

[7] J. W. Chun and C. Y. R. Chen, "A novel leakage power reduction technique for CMOS circuit design," *2010 International SoC Design Conference*, Seoul, 2010, pp. 119-122.

[8] N. S. Kim *et al*., "Leakage current: Moore's law meets static power," in *Computer*, vol. 36, no. 12, pp. 68-75, Dec. 2003.

[9] Ivey, B. (2011). Low-Power Design Guide, *AN1416, Microchip Technology Inc*.

[10] Tangim, Golam, *Noise-Immune Digital Circuit Design Based on Probabilistic Models*, PhD dissertation. University of Calgary, 2012.

[11] P. Larsson and C. Svensson, "Noise in digital dynamic CMOS circuits," in *IEEE Journal of Solid-State Circuits*, vol. 29, no. 6, pp. 655-662, June 1994.

[12] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," *Proceedings of International Conference on Computer-Aided Design*, San Jose, CA, USA, 1996, pp. 524-531.

[13] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson and A. Zaslavsky, "Designing logic circuits for probabilistic computation in the presence of noise," *Proceedings. 42nd Design Automation Conference, 2005.*, Anaheim, CA, 2005, pp. 485-490.

[14] G. R. Srinivasan et al., "Accurate, predictive modeling of soft error rate due to cosmic rays and chip alpha radiation," in *Proc. IEEE Int. Rel. Phys. Symp*., 1994, pp. 12–16.

[15] Lei Wang and N. R. Shanbhag, "Noise-tolerant dynamic circuit design," *ISCAS'99, Proceedings of the 1999 IEEE International*

Symposium on Circuits and Systems VLSI (Cat. No.99CH36349), Orlando, FL, 1999, pp. 549-552 vol.1.

[16] M. I. Elmasry, "Low power VLSI CMOS circuit design," *ICM 2000, Proceedings of the 12th International Conference on Microelectronics, (IEEE Cat. No.00EX453)*, Tehran, Iran, 2000, pp. 4-.

[17] Vardi, M. Y. (2014), Moore's law and the sand-heap paradox, *Communications of the ACM*, *57*(5), 5-5.

[18] Huang, A. (2015), Moore's Law is Dying (and that could be good), *IEEE Spectrum*, *52*(4), 43-47.

[19] E. P. DeBenedictis, "It's Time to Redefine Moore's Law Again," in *Computer*, vol. 50, no. 2, pp. 72-75, Feb. 2017.

[20] A. B. Kahng, "Scaling: More than Moore's law," in *IEEE Design & Test of Computers*, vol. 27, no. 3, pp. 86-87, May-June 2010.

[21] Villacorta, Hector, et al., "FinFET SRAM hardening through design and technology parameters considering process variations," *Radiation and Its Effects on Components and Systems (RADECS), 2013 14th European Conference on*, IEEE, 2013.

[22] Oldiges, Phil, et al. "SOI FinFET soft error upset susceptibility and analysis." *Reliability Physics Symposium (IRPS), 2015 IEEE International*, IEEE, 2015.

[23] Noh, Jinhyun, et al., "Study of neutron soft error rate (SER) sensitivity: investigation of upset mechanisms by comparative simulation of FinFET and planar MOSFET SRAMs," *IEEE Transactions on Nuclear Science* 62.4 (2015): 1642-1649.

[24] Seifert, Norbert, et al., "Soft error rate improvements in 14-nm technology featuring second-generation 3D tri-gate transistors," *IEEE Transactions on Nuclear Science* 62.6 (2015): 2570-2577.

[25] Seifert, Norbert, et al., "Soft error susceptibilities of 22 nm tri-gate devices," *IEEE Transactions on Nuclear Science* 59.6 (2012): 2666-2673.

[26] H. Liu, M. Cotter, S. Datta and V. Narayanan, "Technology assessment of Si and III-V FinFETs and III-V tunnel FETs from soft error rate perspective," *2012 International Electron Devices Meeting*, San Francisco, CA, 2012, pp. 25.5.1-25.5.4.

[27] Liu, Huichu, Suman Datta, and Vijaykrishnan Narayanan, "Steep switching tunnel FET: A promise to extend the energy efficient roadmap for post-CMOS digital and analog/RF applications," *Proceedings of the 2013 International Symposium on Low Power Electronics and Design*, IEEE Press, 2013.

[28] Kelin, Lee Hsiao-Heng, et al., "LEAP: Layout design through error-aware transistor positioning for soft-error resilient sequential cell design," *Reliability Physics Symposium (IRPS), 2010 IEEE International*, IEEE, 2010.

[29] Stanisavljevic, Milos, Alexandre Schmid, and Yusuf Leblebici, "Fault-tolerance of robust feed-forward architecture using single-ended and differential deep-submicron circuits under massive defect density," *Neural Networks, 2006, IJCNN'06. International Joint Conference on*, IEEE, 2006.

[30] Slimani, Mariem, Arwa Ben Dhia, and Lirida Naviner, "Cross logic: A new approach for defect-tolerant circuits," *IC Design & Technology (ICICDT), 2014 IEEE International Conference on*, IEEE, 2014.

[31] P. Korkmaz, B. E. S. Akgul and K. V. Palem, "Energy, Performance, and Probability Tradeoffs for Energy-Efficient Probabilistic CMOS Circuits," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 8, pp. 2249-2262, Sept. 2008.

[32] J. Chen, J. Mundy, Y. Bai, S.-M. C. Chan, P. Petrica, and R. I. Bahar, "A probabilistic approach to nano-computing", *Proc. IEEE Non-Silicon Comput. Workshop*, pp.Chen. 1 -Chen. 8, 2003.

[33] R. I. Bahar, J. Chen, and J. Mundy, "A probabilistic-based design for nanoscale computation", *Nano, quantum and molecular computing, Springer US*, pp.133-156, 2004.

[34] Nepal, Kundan, *et al.*, "Optimizing noise-immune nanoscale circuits using principles of Markov random fields," *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, ACM, 2006.

[35] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson and A. Zaslavsky, "Designing MRF based Error Correcting Circuits for Memory Elements," *Proceedings of the Design Automation & Test in Europe* Conference, Munich, 2006, pp. 1-2.

[36] I. C. Wey, Y. G. Chen, C. Yu, J. Chen and A. Y. Wu, "A 0.18μm Probabilistic-Based Noise-Tolerate Circuit Design and Implementation with 28.7dB Noise-Immunity Improvement," *2006 IEEE Asian Solid-State Circuits Conference*, Hangzhou, 2006, pp. 291-294.

[37] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson and A. Zaslavsky, "Techniques for Designing Noise-Tolerant Multi-Level Combinational Circuits," *2007 Design, Automation & Test in Europe Conference & Exhibition*, Nice, 2007, pp. 1-6.

[38] I. C. Wey, Y. G. Chen, C. H. Yu, A. Y. Wu and J. Chen, "Design and Implementation of Cost-Effective Probabilistic-Based Noise-Tolerant VLSI Circuits," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 11, pp. 2411-2424, Nov. 2009.

[39] Wey, I-Chyn, Yi-Jung Lan, and Chien-Chang Peng, "Reliable ultra-low-voltage low-power probabilistic-based noise-tolerant latch design," *Microelectronics Reliability* 53.12 (2013): 2057-2069.

[40] Wey, I-Chyn, and Ye-Jhih Shen, "Hardware-efficient common-feedback Markov-random-field probabilistic-based noise-tolerant VLSI circuits," *INTEGRATION, the VLSI journal* 47.4 (2014): 431-442.

[41] K. Liu, T. An, H. Cai, L. Naviner, J. F. Naviner and H. Petit, "A general cost-effective design structure for probabilistic-based noise-tolerant logic functions in nanometer CMOS technology," *Eurocon 2013*, Zagreb, 2013, pp. 1829-1836.

[42] Z. Lu, X. P. Yu and K. S. Yeo, "Design of probabilistic-based Markov Random Field logic gates in 65nm CMOS technology," *2010 International SoC Design Conference*, Seoul, 2010, pp. 311-314.

[43] X. Yang, F. Qiao, Q. Wei and H. Yang, "A general scheme for noise-tolerant logic design based on probabilistic and DCVS approaches," *2015 IEEE 13th International New Circuits and Systems Conference (NEWCAS)*, Grenoble, 2015, pp. 1-4.

[44] Y. Li and J. Hu, "Extensional design for noise-tolerate MRF standard cells via global mapping," *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, Melbourne VIC, 2014, pp. 1728-1731.

[45] Yan Li; Xiaoqian Li; Jianhao Hu; Sheng Yang, "Area-sharing cyclic structure MRF cirucits design in ultra-low supply voltage," *in Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, vol., no., pp.2353-2356, 24-27 May 2015

[46] Y. Li, J. Hu, H. Lu and J. Chen, "Area-efficient partial-clique-energy MRF pair design with ultra-low supply voltage," *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, Montreal, QC, 2016, pp. 261-264.

[47] A. R. Kermany, N. H. Hamid and Z. A. Burhanudin, "A study of MRF-based circuit implementation," *Electronic Design, International Conference on,* Penang, 2008, pp. 1-4.

[48] J. Anwer, U. Khalid, N. Singh, N. H. Hamid and V. S. Asirvadam, "Highly noise-tolerant design of digital logic gates using Markov Random Field modelling," *2010 2nd International Conference on Electronic Computer Technology*, Kuala Lumpur, 2010, pp. 24-28.

[49] W. Jian, Y. Sheng and H. Jian-hao, "Timing performance for MRF-based circuits with low supply voltage," *2016 International Conference on Integrated Circuits and Microsystems (ICICM)*, Chengdu, 2016, pp. 61-64.

[50] Y. Miura and Y. Ohkawa, "A noise-tolerant master-slave flip-flop," *2014 IEEE 20th International On-Line Testing Symposium (IOLTS)*, Platja d'Aro, Girona, 2014, pp. 55-61.

[51] Meher, Preetisudha, and Kamala Kanta Mahapatra, "High-performance noise tolerant comparator design for arithmetic circuits," *Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on*. IEEE, 2016.

[52] Sindhu, R. Devi, and S. Jayanthi. "A soft error tolerant 12T hardened memory cell." *Intelligent Systems and Control (ISCO), 2016 10th International Conference on*, IEEE, 2016.

[53] Garcia-Leyva, Lancelot, et al., "Novel redundant logic design for noisy low voltage scenarios," *Circuits and Systems (LASCAS), 2013 IEEE Fourth Latin American Symposium on*, IEEE, 2013.

[54] Sahoo, Sauvagya Ranjan, and Kamala Kanta Mahapatra. "Noise tolerant circuit techniques for high performance feedthrough logic." *Emerging Trends in Science, Engineering and Technology (INCOSET), 2012 International Conference on*, IEEE, 2012.

[55] Von Neumann, John. "Probabilistic logics and the synthesis of reliable organisms from unreliable components." *Automata studies* 34 (1956): 43-98.

[56] R. E. Lyons and W. Vanderkulk, "The Use of Triple-Modular Redundancy to Improve Computer Reliability," in *IBM Journal of Research and Development*, vol. 6, no. 2, pp. 200-209, April 1962.doi: 10.1147/rd.62.0200.

[57] J. A. Abraham and D. P. Siewiorek, "An Algorithm for the Accurate Reliability Evaluation of Triple Modular Redundancy Networks," in *IEEE Transactions on Computers*, vol. C-23, no. 7, pp. 682-692, July 1974.doi: 10.1109/T-C.1974.224016.

[58] Kshirsagar, Ravindra V., and Rajendra M. Patrikar, "Design of a novel fault-tolerant voter circuit for TMR implementation to improve reliability in digital circuits," *Microelectronics Reliability* 49.12 (2009): 1573-1577.

[59] Parhi, Rahul, Chris H. Kim, and Keshab K. Parhi, "Fault-tolerant ripple-carry binary adder using partial triple modular redundancy (PTMR)," *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, IEEE, 2015.

[60] Baloch, S., T. Arslan, and A. Stoica, "Probability based partial triple modular redundancy technique for reconfigurable architectures," *Aerospace Conference, 2006 IEEE*. IEEE, 2006.

[61] Terada, Ryo, and Minoru Watanabe, "Error injection analysis for triple modular and penta-modular redundancies," *Next Generation Electronics (ISNE), 2017 6th International Symposium on*, IEEE, 2017.

[62] J. Teifel, "Self-Voting Dual-Modular-Redundancy Circuits for Single-Event-Transient Mitigation," in *IEEE Transactions on Nuclear Science*, vol. 55, no. 6, pp. 3435-3439, Dec. 2008.

[63] Wey, I-Chyn, et al., "Robust C-element design for soft-error mitigation," *IEICE Electronics Express* 12.10 (2015): 20150268-20150268.

[64] Smith, Farouk, "A new methodology for single event transient suppression in flash FPGAs," *Microprocessors and Microsystems* 37.3 (2013): 313-318.

[65] I. C. Wey, C. C. Peng and F. Y. Liao, "Reliable Low-Power Multiplier Design Using Fixed-Width Replica Redundancy Block," in *IEEE*

Transactions on Very Large Scale Integration (VLSI) Systems, vol. 23, no. 1, pp. 78-87, Jan. 2015.

[66] Byonghyo Shim and N. R. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 4, pp. 336-348, April 2006.

[67] Y. H. Huang, "High-Efficiency Soft-Error-Tolerant Digital Signal Processing Using Fine-Grain Subword-Detection Processing," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 2, pp. 291-304, Feb. 2010.

[68] B. Gaines, "Stochastic computing," in *Proc. Spring Joint Comput. Conf.*,1967, pp. 149–156.

[69] W. Poppelbaum, C. Afuso, and J. Esch, "Stochastic computing elements and systems," in *Proc. Fall Joint Comput. Conf.*, 1967, pp. 635–644.

[70] P. Knag, W. Lu and Z. Zhang, "A Native Stochastic Computing Architecture Enabled by Memristors," in *IEEE Transactions on Nanotechnology*, vol. 13, no. 2, pp. 283-293, March 2014.

[71] Da Zhang, H. Li and S. Y. Foo, "A simplified FPGA implementation of neural network algorithms integrated with stochastic theory for power electronics applications," *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, Raleigh, NC, 2005, pp. 6 pp.-.

[72] I. Yeo, S. Gi, B. Lee and M. Chu, "Stochastic implementation of the activation function for artificial neural networks," *2016 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Shanghai, 2016, pp. 440-443.

[73] A. Zhakatayev, S. Lee, H. Sim and J. Lee, "Sign-Magnitude SC: Getting 10X Accuracy for Free in Stochastic Computing for Deep Neural Networks*," *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, 2018, pp. 1-6.

[74] H. Sim, S. Kenzhegulov and J. Lee, "DPS: Dynamic Precision Scaling for Stochastic Computing-based Deep Neural Networks*," *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, 2018, pp. 1-6.

[75] J. Chen, J. Hu and S. Li, "Low power digital signal processing scheme via stochastic logic protection," *2012 IEEE International Symposium on Circuits and Systems*, Seoul, 2012, pp. 3077-3080.

[76] J. Chen and J. Hu, "A novel FIR filter based on stochastic logic," *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, Beijing, 2013, pp. 2050-2053.

[77] L. Miao and C. Chakrabarti, "A parallel stochastic computing system with improved accuracy," *SiPS 2013 Proceedings*, Taipei City, 2013, pp. 195-200.

[78] A. Ardakani, F. Leduc-Primeau and W. J. Gross, "Hardware implementation of FIR/IIR digital filters using integral stochastic

computation," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, 2016, pp. 6540-6544.

[79] N. Onizawa, S. Koshita, S. Sakamoto, M. Kawamata and T. Hanyu, "Design of stochastic asymmetric compensation filters for auditory signal processing," *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Montreal, QC, 2017, pp. 1315-1319.

[80] Li, Stan Z, "Mathematical MRF models," Markov random field modeling in image analysis (2009): 1-28.

[81] Li, Yan, Jianhao Hu, and Yufeng Li. "Supply voltage analysis for MRF circuits design based on information theory." *IEICE Electronics Express* (2016): 13-20161080.

[82] P. K. Samudrala, J. Ramos and S. Katkoori, "Selective triple Modular redundancy (STMR) based single-event upset (SEU) tolerant synthesis for FPGAs," in *IEEE Transactions on Nuclear Science*, vol. 51, no. 5, pp. 2957-2969, Oct. 2004.

[83] J. Losq, "A Highly Efficient Redundancy Scheme: Self-Purging Redundancy," in *IEEE Transactions on Computers*, vol. C-25, no. 6, pp. 569-578, June 1976.

[84] Hagbae Kim and K. G. Shin, "Design and analysis of an optimal instruction-retry policy for TMR controller computers," in *IEEE Transactions on Computers*, vol. 45, no. 11, pp. 1217-1225, Nov 1996.

[85] Qi, Chunhua, et al., "Low cost and highly reliable radiation hardened latch design in 65nm CMOS technology," *Microelectronics Reliability* 55.6 (2015): 863-872.

[86] K. M. Chu and D. L. Pulfrey, "Design procedures for differential cascode voltage switch circuits," in *IEEE Journal of Solid-State Circuits*, vol. 21, no. 6, pp. 1082-1087, Dec. 1986.

[87] H. El-Banna, A. A. El-Fattah and W. Fakhr, "An efficient implementation of the 1D DCT using FPGA technology," *Proceedings of the 12th IEEE International Conference on Fuzzy Systems (Cat. No.03CH37442)*, Cairo, Egypt, 2003, pp. 278-281.

[88] N. Ahmed, T. Natarjan, and K. R. Rao, "Discrete Cosine Transform," *IEEE Trans. Comput.*, vol. 100, no. 1, pp. 90–93, Jan. 1974.

[89] F. Bartolini, A. Manetti, A. Piva and M. Barni, "A data hiding approach for correcting errors in H.263 video transmitted over a noisy channel," *2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No.01TH8564)*, Cannes, 2001, pp. 65-70.

[90] H. Wang and S. Kwong, "Hybrid Model to Detect Zero Quantized DCT Coefficients in H.264," in *IEEE Transactions on Multimedia*, vol. 9, no. 4, pp. 728-735, June 2007.

[91] Cintra, Renato J., et al., "Low-Complexity DCT Approximations for Biomedical Signal Processing in Big Data," *Signal Processing and Machine Learning for Biomedical Big Data*. CRC Press, 2018. 151-176.

[92] Yulin Wang and A. Pearmain, "Blind MPEG-2 video watermarking robust against geometric attacks: a set of approaches in DCT domain," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1536-1543, June 2006.

[93] Abd-Elhafiez, Walaa M., and Wajeb Gharibi, "A Novel Color Image Compression Method Using the Adaptive Threshold," *TELKOMNIKA (Telecommunication Computing Electronics and Control)* 17.2 (2018).

[94] Kapranova, Ekaterina A., Vadim A. Nenashev, and Mikhail B. Sergeev, "Compression and coding of images for satellite systems of Earth remote sensing based on quasi-orthogonal matrices," *Image and Signal Processing for Remote Sensing XXIV*. Vol. 10789, International Society for Optics and Photonics, 2018

[95] Watson, Andrew B., "Image compression using the discrete cosine transform," *Mathematica Journal* 4.1 (1994):81.

[96] S. L. Toral, J. M. Quero and L. G. Franquelo, "SRC passivation controller implementation using stochastic computing," *ISCAS 2001, The 2001 IEEE International Symposium on Circuits and Systems (Cat. No.01CH37196)*, Sydney, NSW, 2001, pp. 723-726 vol. 2.

[97] Dinu, A., M. N. Cirstea, and M. McCormick, "Stochastic implementation of motor controllers," *Proceedings of the 2002 IEEE International Symposium on Industrial Electronics (ISIE)*, Vol. 2. 2002.

[98] R. Seva, P. Metku, K. K. Kim, Y. Kim and M. Choi, "Approximate stochastic computing (ASC) for image processing applications," *2016 International SoC Design Conference (ISOCC)*, Jeju, 2016, pp. 31-32.

[99] C. Ma, S. Zhong and H. Dang, "High Fault Tolerant Image Processing System Based on Stochastic Computing," *2012 International Conference on Computer Science and Service System*, Nanjing, 2012, pp. 1587-1590.

[100] J. Li, C. Yang, P. Culverhouse and H. Ma, "Stochastic neural network control of rigid robot manipulator with passive last joint," *Proceedings of 2012 UKACC International Conference on Control*, Cardiff, 2012, pp. 662-667.

[101] Y. Xie, S. Liao, B. Yuan, Y. Wang and Z. Wang, "Fully-Parallel Area-Efficient Deep Neural Network Design Using Stochastic Computing," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 12, pp. 1382-1386, Dec. 2017.

[102] Gaba, Siddharth, et al., "Stochastic memristive devices for computing and neuromorphic applications," *Nanoscale* 5.13 (2013): 5872-5878

[103] Te-Hsuan Chen and J. P. Hayes, "Equivalence among stochastic logic circuits and its application," *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, San Francisco, CA, 2015, pp. 1-6.

[104] K. K. Parhi, "Analysis of stochastic logic circuits in unipolar, bipolar and hybrid formats," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4.

[105] P. Jeavons, D. A. Cohen and J. Shawe-Taylor, "Generating binary sequences for stochastic computing," *IEEE Transactions on Information Theory*, vol. 40, no. 3, pp. 716-720, May 1994.

[106] S. Zhang, Y. Shen and C. Yang, "A stochastic computation based integer DCT implementation in HEVC," *2014 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, Guilin, 2014, pp. 153-157.

[107] A. Alaghi and J. P. Hayes, "Survey of Stochastic Computing," *ACM Transactions on Embedded computing systems (TECS)* 12.2s (2013): 92.

[108] Y. Li and J. Hu, "A novel implementation scheme for high area-efficient DCT based on signed stochastic computation," *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013),* Beijing, 2013, pp. 990-993. doi: 10.1109/ISCAS.2013.6572015.

Appendix: CPMRF Code Examples

functional_AND_NOR.vhd:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity functional_AND_NOR is
   Port ( sig_ina : in  STD_LOGIC;
          sig_inb : in  STD_LOGIC;
                    test_AND : out  STD_LOGIC;
          test_NOR : out  STD_LOGIC);
end functional_AND_NOR;
architecture Behavioral of functional_AND_NOR is
COMPONENT proposed_AND_NOR is
          PORT(
                    in_a  : IN std_logic;
                    in_b  : IN std_logic;
                    out_AND: OUT std_logic;
                    out_NOR: OUT std_logic);
          END COMPONENT;

COMPONENT CMOS_AND_NOR is
          PORT(
                    in_a  : IN std_logic;
                    in_b  : IN std_logic;
                    out_AND: OUT std_logic;
```

```
                out_NOR: OUT std_logic);
        END COMPONENT;

signal t1,t2,t3,t4   : std_logic;

begin
I1: proposed_AND_NOR PORT MAP(
                in_a => sig_ina,
                in_b => sig_inb,
                out_AND=> t1,
                out_NOR=> t2);

I2: CMOS_AND_NOR PORT MAP(
                in_a => sig_ina,
                in_b => sig_inb,
                out_AND=> t3,
                out_NOR=> t4);

test_AND<= t1 xor t3;
test_NOR<= t2 xor t4;
end Behavioral;
```

---

proposed_AND_NOR.vhd:

```
library IEEE;
```

```
use IEEE.STD_LOGIC_1164.ALL;

entity proposed_AND_NOR is
        PORT(
                in_a  : IN std_logic;
                in_b  : IN std_logic;
                out_AND: OUT std_logic;
                out_NOR: OUT std_logic);
end proposed_AND_NOR;

architecture Behavioral of proposed_AND_NOR is
        signal t1,t2,t3,tc1,tc2,y1,y2   : std_logic;

begin
t1<=in_a nor in_b ;
t2<=in_a and in_b ;
t3<=in_a xor in_b ;
tc1<=t1 nor t3 ;
tc2<=t2 nor t3 ;
y1<=tc1 nand y2;
y2<=tc2 nand y1;
out_AND<=not y1;
out_NOR<=not y2;
end Behavioral;
```

CMOS_AND_NOR.vhd:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity CMOS_AND_NOR is
        PORT(
                in_a  : IN std_logic;
                in_b  : IN std_logic;
                out_AND: OUT std_logic;
                out_NOR: OUT std_logic);
end CMOS_AND_NOR;

architecture Behavioral of CMOS_AND_NOR is

begin
out_AND<=in_a and in_b ;
out_NOR<=in_a nor in_b ;
end Behavioral;
```

---

functional_AND_NOR_tb.vhd:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY functional_AND_NOR_tb IS
```

```
END functional_AND_NOR_tb;

ARCHITECTURE behavior OF functional_AND_NOR_tb IS
    COMPONENT functional_AND_NOR
    Port ( sig_ina : in  STD_LOGIC;
           sig_inb : in  STD_LOGIC;
                     test_AND : out  STD_LOGIC;
           test_NOR : out  STD_LOGIC);
    END COMPONENT;
   signal sig_ina : std_logic;
   signal sig_inb : std_logic;
   signal test_AND : std_logic;
   signal test_NOR : std_logic;
   constant clk_period : time := 100 ns;

BEGIN

   uut: functional_AND_NOR PORT MAP (
          sig_ina  => sig_ina,
          sig_inb => sig_inb,
          test_AND => test_AND,
          test_NOR => test_NOR
        );
 sig_ina_process :process
```

```
        begin
                sig_ina <= '1';
                wait for clk_period;
                sig_ina <= '0';
                wait for clk_period;
end process;

sig_inb_process :process
        begin
                sig_inb <= '1';
                wait for clk_period/2;
                sig_inb <= '0';
                wait for clk_period/2;
end process;


  -- Stimulus process
  stim_proc: process
begin
   wait;
 end process;
END;
```