UNIVERSITY OF SOUTHAMPTON

# Ultra-Low-Power and Variation-Aware Digital Signal Processing

by

Yue Lu

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Faculty of Engineering, Science and Mathematics

Electronics and Computer Science

May 2019

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Yue Lu

Internet of Things is a rapidly emerging technology that allows people and things to be connected anywhere and any time, using any path or network. The "things" in the IoT applications need external power to perform any function. As a result, IoT devices must offer superior power efficiency compared with previous digital applications. In terms of power-saving techniques, voltage scaling is regarded as one of the most efficient method. However, the aggressive scaling of the CMOS size and supply voltage have posed a increase concern about circuit reliability. This thesis studies the effects of process variation on the low-voltage digital signal processing circuits, with the aim to develop efficient methods to further achieve power saving at the circuit-level and provide corresponding cost-effective error mitigation techniques.

This thesis presents three major contributions. First of all, the analysis of process variation on the adiabatic logic circuit (an energy recovery logic) is implemented and bit-serial implementation is proposed as an efficient method to improve the energy-efficiency and robustness against variation. After that, our research emphasis in this thesis is shifted to low power and reliable serial computing. The second contribution demonstrates the advantages of serial computing on a FFT implementation in terms of area minimisation, power-saving and energy-scalable operation. The third contribution proposes several novel error-mitigation techniques for serial computing via path isolation, variable latency or time borrowing. The FIR evaluation results show that the proposed techniques can effectively mask or prevent timing errors with very low hardware cost and make serial circuit as a promising candidate for power-efficient and reliable computing in the IoT devices.

# Contents

# List of Figures

# List of Tables

## Acknowledgements

I would like to thank my supervisor, Dr Tom Kazmierski, for his support and invaluable guidance through my PhD. I have been extremely fortunate to have such supervisor who cared so much about my work and responded to my questions so promptly. Also, I would like to offer my thanks to the staffs and students in CPS, for their help. Finally, my parent and wife have always been my strongest supporters. I am greatly thankful for their love, and I would like to dedicate this work to them.

# Chapter 1

# Introduction

In 1965, Gordon Moore of Intel Corporation observed that plotting the number of transistors that can be most economically manufactured on a chip gives a straight line on a semilog-arithmetic scale [1]. At the time, he found the transistor count doubling every 18 months. This observation has been called *Moore's Law*, which has become a self-fulfilling prophecy. Over the last half-century, the semiconductor world has been driving towards aggressive increases in chip density and performance. However, along with technology scaling, limited budgets for chip area and energy, and increasing requirements for circuit performance, functionality and reliability present more challenges for the next generation of designers. Accordingly, continuous innovation of advanced elements and processes are required for various design constraints.

## 1.1 Motivation

Because of advances in very large scale integration (VLSI) technology, the Internet of Things (IoT) [2][3] is growing rapidly, as huge numbers of connected meters and machines come online, all bristling with sensors to measure the world around them. The IoT also represents a turning point for VLSI design itself [4]. The combination of higher costs per unit area and thermal power dissipation limits useful chip integration. However, the IoT provides an alternative model through the deployment of vast numbers of small chips. As a result, these connected devices in the IoT must satisfy two criteria: be able to operate with very low power consumption, and be robust against variations.

Various methods, on all hierarchical levels, have been proposed in the past to reduce the power consumption in static complementary metal-oxide-semiconductor (CMOS) circuits [5][6][7]. One of the most effective solutions is voltage scaling [8][9]. For IoT-related applications without strict latency constraints, both the supply voltage and clock frequency can be scaled down as low as possible to minimise the dynamic energy

consumption. However, in this case, power consumption due to leakage might be the dominant factor. In order to reduce total power consumption further, some circuit-level approaches are urgently needed to minimise leakage power. IoT devices usually require adaptability to changing conditions. Accordingly, power-scalable operations should be taken into consideration for the relevant digital designs.

On the other hand, reliability issue [10] associated with ultra-low-power circuits is also a major concern for current designers. Low voltage designs are more affected by process variations compared to nominal designs, because the on-current in the low voltage region is highly sensitive to variations in threshold voltage. Increased process variations in advanced technology nodes further exacerbates the problem.

## 1.2 Research Aims and Contributions

This work aims to explore further the power-saving techniques at the circuit-level combined with voltage scaling, which is suitable for future IoT-related devices. Considering the moderate performance requirements in the IoT systems, the research also tries to trade-off the balance between power, area, and performance. On the other hand, with the implementation of low voltage operation, the effect of process variations would be a large concern. Accordingly, specific variation-aware approaches corresponding to the different circuit-level power saving techniques are urgently required.

The contributions of the research presented in this thesis can be summarised as follows:

- Evaluated and analysed the effect of process variations on the adiabatic logic combined with the near-threshold operation. To improve the robustness of the adiabatic adder further against process variation, a bit-serial structure is considered with an even lower energy consumption per cycle.
- Developed a new approach to bit-serial signal processing with ultra-low-power scalable multiplication. This is demonstrated by the design of an FFT in the 65 nm CMOS process.
- Developed several new approaches for timing-error tolerant serial computing, which modifies path delays for different bit results, and minimises the effects caused by timing violations. This is demonstrated by the design of an FIR filter or FFT implementation in the 65 nm CMOS process.

## 1.3 Thesis Organisation

Chapter 2 introduces the state-of-the-art low power techniques that would be introduced, namely adiabatic logic, serial computing, and voltage scaling. Moreover, a survey of timing error resilient approaches is also presented.

Chapter 3 firstly analyses the process variation effects on the adiabatic logic combined with the near-threshold operation. To improve the robustness of the ripple adder against process variation further, a bit-serial adiabatic adder is considered with an even lower energy consumption per cycle.

Chapter 4 investigates the possibility of further power reductions where a near-threshold bit-serial operation is used in signed multiplication. The ideal is demonstrated on a 64-point FFT, and the results were obtained using HSPICE simulations and the 65 nm ST technology library for a 600 mV supply voltage.

Chapter 5 proposes a novel approach to prevent timing errors at the circuit-level for serial computing. The proposed design places checkpoint flip-flops on each critical-path, and dynamically splits the clock cycle into two for error-resilience.

Chapter 6 demonstrates a timing error mitigation method at the circuit-level for serial computing. This is achieved by modifying the path delay distribution for different bit results. For instance, the path delay for the least significant bit (LSB) computation would be designed as the longest one, while the most significant bit (MSB) path should be the shortest. Accordingly, timing violations would weakly result in large computation accuracy losses.

Chapter 7 presents a method to bound the magnitude of timing errors and are demonstrated on a 16-tap FIR filter based on a distributed arithmetic (DA) circuit. The bit order in DA-based serial computations is initially reversed, thereby providing MSB paths with more guardband by time borrowing. This ensures the timing errors weakly affect the computation results. Further, a shifted-phase clock signal is applied on the end-point registers. Hence, extra time slack would be provided without any effects on system sampling rate.

Chapter 8 provides the conclusions regarding this thesis, including suggestions for future studies.

## 1.4 List of Publications

The contributions of this work have been published in 6 papers, 2 further papers are currently considered for publication after revisions, and 2 more papers have recently been submitted:

### 1.4.1 Conference papers

- Y. Lu, T. Kazmierski, "Error-Free Near-Threshold Adiabatic CMOS Logic in the Presence of Process Variation," in Forum on Specification and Design Languages (FDL), 2016.
- Y. Lu, T. Kazmierski, "An ultra-low-power variable-accuracy bit-serial FFT butterfly processing element for IoT sensors," in IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), 2016.
- Y. Lu, T. Kazmierski, "A Reconfigurable Bit-serial FFT/FIR Processor for Ultra-low-power," in Forum on Specification and Design Languages (FDL), 2017.
- Y. Lu, T. Kazmierski, "Variable-Accuracy Bit-serial Multiplication with Row Bypassing for Ultra Low Power," in IEEE Nordic Conference on Circuits and Systems (NORCAS), 2017.
- Y. Lu, T. Kazmierski, "A New Ageing-Aware Approach via Path Isolation," in Forum on Specification and Design Languages (FDL), 2018.
- Y. Lu, T. Kazmierski, "An Ultra-Low-Power Bit-Serial Variable-Accuracy FFT Processor," in IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), 2018.

### 1.4.2 Book section

- Y. Lu, T. Kazmierski, "Error-Free Near-Threshold Adiabatic CMOS Logic in the Presence of Process Variation," in Languages, Design Methods, and Tools for Electronic System Design, 2018, pp. 103-114.

### 1.4.3 Journal papers

- Y. Lu, T. Kazmierski, "A New Timing Error Prevention Method for Bit-serial Sequential Circuits," IET Computers & Digital Techniques, 2018. (Major revision)
- Y. Lu, S. Duan and T. Kazmierski, "A Cost-Efficient Error-Resilient Approach to Distributed Arithmetic for Signal Processing," Elsevier Microelectronics Reliability. (Second-round Revision)
- Y. Lu, S. Duan, H. Basel and T. Kazmierski, "Error-Resilient Low Power Serial FIR Filter via Adaptive Latency," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2018. (Submitted)
- Y. Lu, S. Duan and T. Kazmierski, "A Variation-aware Design Methodology for Distributed Arithmetic," MDPI Electronics, 2018. (Submitted)

# Chapter 2

# Literature Review

With the rapid development of the Internet of Things (IoT), its associated technologies are successfully integrating into everyday life. It is predicted that by 2020 approximately 50 billion electronic devices will be connected via the web. These devices, from cars and doorbells to kitchenware, will share this enormous amount of data. However, this will significantly increase the demand for power-efficient signal processing. On the other hand, the conscious downscaling of transistor dimensions also presents a severe challenge in CMOS readability issues. In this chapter, state-of-the-art low power and error-resilient techniques are introduced in Section 2.1 and Section 2.2, respectively.

## 2.1 State-of-the-art Low Power Techniques

The most effective strategy for improving power efficiency is to aggressively reduce the power supply voltage according to application performance demands [11]. Adiabatic logic [12] is another attractive approach for reducing dynamic power, which uses an AC (alternating current) voltage supply to recover power instead of being consumed in the ground. The energy dissipated in adiabatic circuits is considerably less than in transitional CMOS circuits. Therefore, adiabatic logic circuits are regarded as promising candidates for low-power circuits. To realise further power saving, voltage scaling is combined with the adiabatic logic circuit. Several papers regarding near-threshold adiabatic circuits have been published [13][14][15]. Bit-serial computing [16] is also regarded as a possible approach for reducing power consumption, or even total energy dissipation. In the bit-serial architecture, all the data is processed bit by bit to reduce hardware complexity. However, operation speeds are significantly affected due to the bit-serial architecture. The same operation may take much longer in terms of working clock cycles compared with conventional architecture.

In this section, an overview of promising power-saving techniques is presented. Specifically, Section 2.1.1 illustrates the principle and sources of CMOS power dissipation. Section 2.1.2 introduces the advantages and potential challenges in reliability of sub/near-threshold voltage designs. Section 2.1.4 describes the structure and types of serial circuits, and comparisons with their conventional parallel counterparts. The principle of the adiabatic logic circuit and the effects of process variations are presented in Section 2.1.3.

### 2.1.1 CMOS Power Dissipation

Power consumption is defined as the amount of energy used per unit of time. In general, power consumption can be classified into two types: dynamic power and static power [17]. The total power consumption in a CMOS circuit is given by putting this together, as shown in the following equations.

$$P_{dynamic} = P_{switching} + P_{short\ circuit} \tag{2.1}$$

$$P_{static} = I_{leakage} * V_{DD} \tag{2.2}$$

$$P_{total} = P_{dynamic} + P_{static} \tag{2.3}$$

#### 2.1.1.1 Dynamic power

Dynamic power mostly consists of the switching power, as seen in Equation. 2.1. The switching power denotes that during a certain time unit ($\frac{1}{f}$) or at a fixed frequency $f$, the power is consumed for charging and discharging the load capacitance in the circuit.

$$P_{swicthing} = \alpha C {V_{DD}}^2 f \tag{2.4}$$

Where $\alpha$, $C$ and $V_{DD}$ represents the corresponding switching activity, load capacitance, and supply voltage. According to these presented equations, it can be found that dynamic power can be saved by reducing the supply voltage $V_{DD}$ or switching activity $\alpha$. First of all, as $V_{DD}$ is a quadratic term in Equation. 2.4, the minimum $V_{DD}$ and working frequency can be selected when the digital circuit is not latency-constrained. Moreover, the activity factor is an efficient and simple lever for power reduction. When a chip turns off entirely, the activity factor and dynamic power become zero together. The easiest way of blocking the circuits is to stop the clock, which can be termed clock-gating.

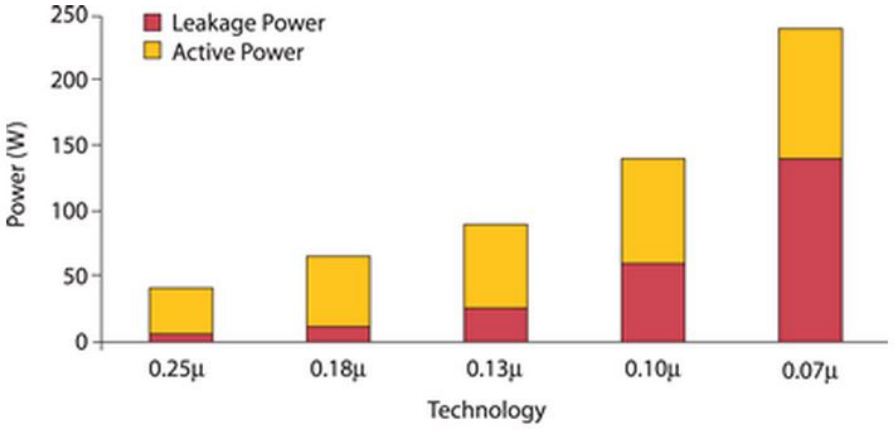



FIGURE 2.1: Power distribution with technology scaling [18].

#### 2.1.1.2 Static power

Static power refers to the power consumed when a chip is not working, where the sub-threshold leakage power is usually the dominants source. Basically, subthreshold leakage current flows when a transistor is OFF. The sub-threshold leakage current is problematic because it increases as transistor threshold voltages $V_{th}$ decrease. For 130-nm technology, the ratio of leakage power exhibits a significant increase. At sub-100-nm, leakage power accounts for as much as 50% of the total power consumed by a chip (as seen in Figure 2.1). Therefore, leakage will dominate total power dissipation in the future. Because the leakage power in a chip is approximately linearly correlated with the gate count, an effective method is to minimise the number of digital gates while maintaining correct circuit functionality.

### 2.1.2 Near-threshold Voltage Design

Research to date indicates that voltage scaling is the most effective solution for reducing power, and has been practically applied in many digital designs [11]. However, reducing the supply voltage causes a quadratic decrease in the dynamic power with an ensuing performance penalty. Scaling the voltage down to the sub-threshold region could achieve ultra-low power consumption, which was first proposed several decades ago [19]. However, the performance when operating at sub-threshold levels is unsuitable for most digital devices and the high-sensitivity to parameter variation is also a major concern. Near-threshold computing (NTC) [20][11] is an important research approach, which scales down the supply voltage to (approximately) the threshold voltage. Therefore, much of the energy savings typical of sub-threshold operation can be retained with an acceptable performance loss. In the rest of the chapter, the potential advantages, challenges, and corresponding solutions of ultra-low voltage operation are discussed.

#### 2.1.2.1 NTV Computing Analysis

Although subthreshold operation consumes ultra-low power, it can only relate to a very narrow market because of the severe performance loss. Comparatively, NTC (a design space where the supply voltage is approximately equal to the transistor threshold voltage), retains considerable performance with an acceptable energy penalty [11]. Therefore, it can be applied in a broad range of power-constrained devices from wireless sensor nodes to high-performance servers.

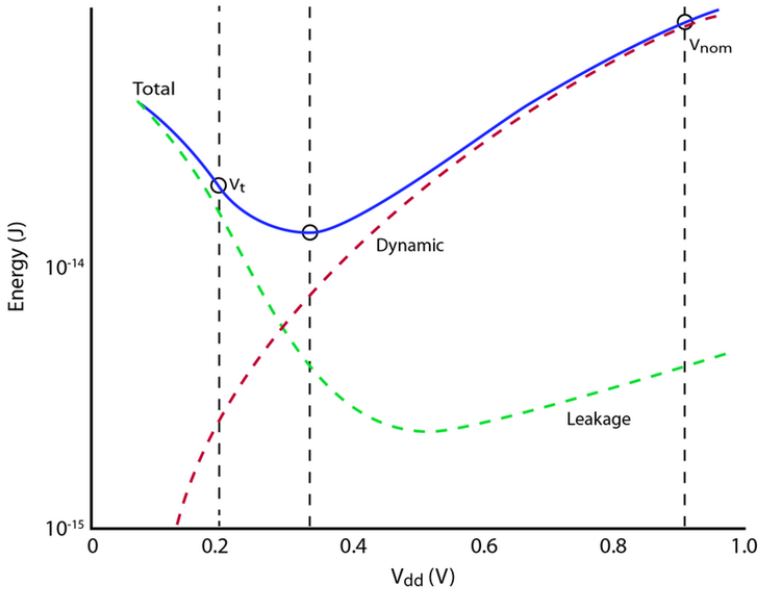



FIGURE 2.2: Energy minimum and contributors to total energy for a 32-nm process [11].

Given the Figure 2.2 showing the changes in energy and delay in the different voltage domains, the maximum energy-efficiency point can clearly observed. In the super-threshold region, the energy consumption is mainly determined by the supply voltage. This is because the majority of the energy cost is consumed by the dynamic power, which has a quadratic relationship with voltage.

Research by David Blaauw's group at the University of Michigan indicates that the energy minimum point is close to 0.3 V for a typical 32-nm CMOS process with a 0.2-V threshold voltage, where the leakage power has a significant contribution [21]. It is clear that the crossover between dynamic and static power consumption is only slightly below the minimum energy point.

#### 2.1.2.2 Preliminary Works in Near-threshold Computing

Recently, many advanced processors operating at sub-threshold or near-threshold voltages have been proposed [21][22][23]. The Subliminal [24] and Phoenix processors [25]

are the two most representative sub-threshold processors.

As discussed in the previous section, the minimum-energy point for both of these processors occurs in the sub-threshold region. Further, there is only a modest increase in energy consumption when the supply voltage reaches the near-threshold region. Meanwhile, the performance of the processors can be optimised (to a large extent). For the Subliminal processor, it costs 33.1 pJ/instruction at the nominal voltage with a 20.5 MHz working frequency, while near-threshold computing yields energy and frequency reductions in the order of 6.6 and 11.4, respectively. For the Phoenix processor, the nominal energy consumption and frequency are 29.6 pJ/instruction and 9.13 MHz, which are reduced by 9.8 and 9.1, respectively, at the near-threshold voltage. In terms of some specific-purpose processors, the JPEG co-processor designed by Yu Pu only costs 1.0 pJ/cycle with a 0.45 V supply voltage at 4.5 MHz in the near-threshold region [23]. In [26], an alternative processing platform for biomedical signal processing is presented, whereby the power and performance can be scaled to adapt to the application need. In the high-performance mode, the proposed system consumes 145 pJ/instruction, while the energy consumption reduces to 12.8 pJ/instruction with a 1 MHz working frequency in the low power operation mode.

#### 2.1.2.3 Challenges of Near-threshold Voltage Design

In the near-threshold regime, the on-current $I_{on}$ is increasingly sensitive to variations in threshold voltage $V_{th}$. Moreover, near-threshold designs are more affected by process variations compared to nominal designs [27]. Further, timing violations resulting from process variations are dramatically increased, which is a serious problem that needs to be addressed. Figure 2.3 shows that the gate delay variation is disproportional to the supply voltage. At nominal voltages, the frequency variation is stable at approximately 1.1, and the delay has a 5 increase at 400 mV. Moreover, near-threshold operation can also result in increased sensitivity to temperature and supply ripple, both of which would make the problem of performance variation even more serious [20][28].

### 2.1.3 Adiabatic Logic Circuit

In this section, the basic principles of the adiabatic circuit are presented. The word "adiabatic" is from the Greek language, and is used to describe a thermodynamics process where there is no energy loss or gain (in the form of dissipated heat) in a system. In 1994, this idea was first demonstrated on a digital circuit design by Koller and Athas [29]. They introduced an adiabatic latch circuit that could store the information on which node was charged, and the energy was recovered from the output nodes. However, because these designs use MOSFET devices as latching elements, the recovery only works until the threshold voltage is reached, and a part of the stored energy cannot be

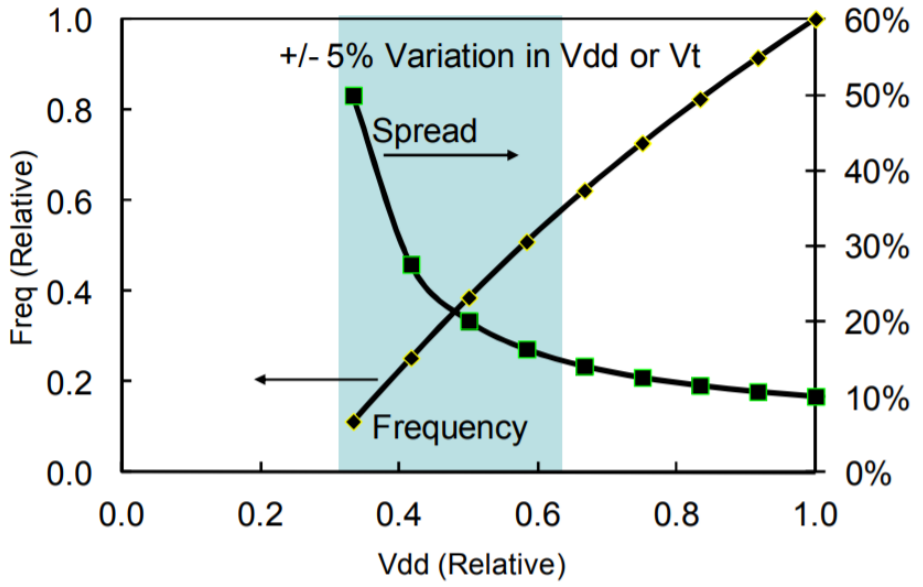



FIGURE 2.3: Variation analysis versus supply voltage [11].

recovered. This type of circuit is termed quasi-adiabatic, because of the unavoidable non-adiabatic losses. A variety of quasi-adiabatic families [30][31][32][33][34] have been introduced over the years.

Recently, adiabatic logic has been frequently applied in low-power digital designs [33]. Compared with traditional static CMOS circuits, it can minimise energy losses during the charging or discharging processes.

#### 2.1.3.1 The Charging Process in Adiabatic Logic

The energy consumption caused by switching of an inverter can be observed in Figure 2.4, where capacitor C represents the total load capacitance to which the pull-up and pull-down networks are both connected. When the input signal changes from zero to one, the charge from the voltage source transfers to charge the output capacitor. During this process, the total energy can be represented as follows:

$$E_{V_{DD}} = QV_{DD} = CV_{DD}^2 \tag{2.5}$$

Assuming the energy transferred from the supply voltage is equal to that supplied to the output capacitor, the energy stored in the output capacitor is equal to

$$E_C = \frac{1}{2}CV_{DD}^2 \tag{2.6}$$

The other half is dissipated in the PMOS transistor, because the transistor has a voltage across it when the current suddenly flows. Similarly, when an input transition from one

back to zero occurs, the energy stored in the capacitor is dissipated in the pull-down network [33].

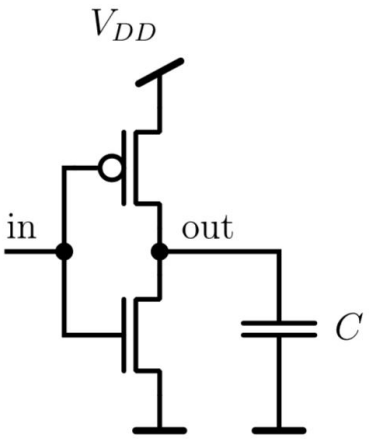



FIGURE 2.4: The schematic of an inverter.

The equivalent circuits in Figure 2.5 illustrate the energy dissipation during switching transitions. Compared with a static CMOS circuit, the energy saving is primarily due to the time-varying voltage source.

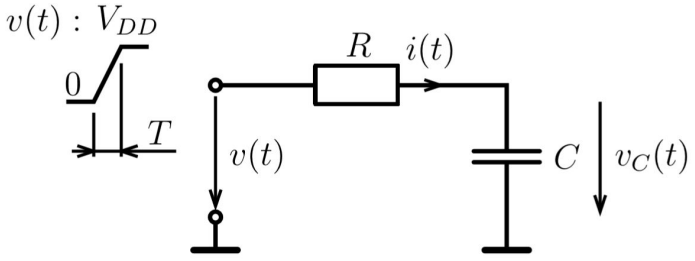



FIGURE 2.5: The equivalent circuits illustrating the energy dissipation during switching transitions

When the input signal transfers from one to zero, if the process is slow enough the voltage on the capacitor can follow the change of the power clock. Accordingly, the voltage on the capacitor is approximately equal to the voltage source. Therefore, the current flow pull-up network can be expressed by the following:

$$i(t) = C\frac{dv(t)}{dt} = \frac{CV_{DD}}{T} \tag{2.7}$$

where C represents the output capacitance, and T denotes the transition time of the power clock. By integrating the power p(t) during the transition time T, the energy for

the switching process can be determined by the following:

$$E = \int_0^T p(t)dt = \int_0^T v(t) * i(t)dt = \int_0^T (v_R(t) + v_C(t)) * i(t)dt \tag{2.8}$$

Because there is no energy consumed in the output capacitor, the total energy dissipation in the charging or discharging process is equal to the following:

$$E = \int_0^T R\frac{C^2V_{DD}^2}{T^2}dt = 2\frac{RC}{T}CV_{DD}^2 \tag{2.9}$$

In contrast to the charge/discharge process of the static logic circuit, when the T = 2RC the dynamic energy dissipation in these two logic circuits is the same. However, when the transition time is greater than 2RC, adiabatic logic appears more energy-efficient. It should be mentioned that all the previous analyses about energy dissipation are only relevant in an ideal adiabatic system, and the detailed loss mechanisms in adiabatic logic are discussed in Section 2.1.3.3.

#### 2.1.3.2 A Review of Adiabatic Logic Families

Over the last two decades, a large number of adiabatic logic families have been proposed. Among these, four of the most traditional families are discussed in the current section: ECRL [35], improved efficient charge recovery logic (IECRL) [36], and positive feedback adiabatic logic (PFAL) [30]. It is noteworthy that all of them are operated with a four-phase power clock, which can be seen in Figure 2.6.

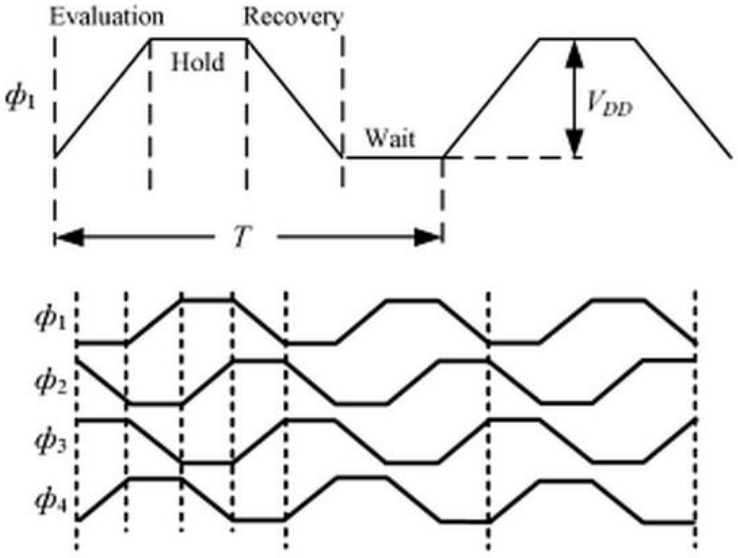
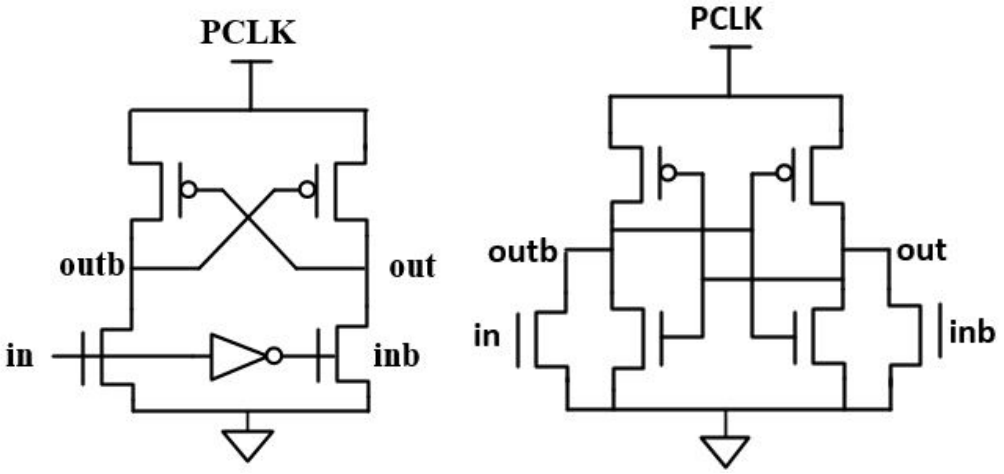



FIGURE 2.6: Four-phase power clock [33].

ECRL consists of a cross-coupled PMOS pair to store information. The source terminals are connected to the power clock, and the gates are driven by the drain of the other.

The drains of the PMOS transistors are proposed as outputs in the gate circuit. The pull-down network is composed of two NMOS transistors connecting the input signals. One disadvantage of ECRL is that when the charge in the previous stage is recovered, there would be no pull-down path to ground, causing floating output nodes. IECRL is designed to address this problem, whereby the two extra NMOS transistors can create a pull-down path during the idle phase. However, the number of transistors is increased, and the total area becomes larger. Similar to IECRL, PFAL consists of a latch element




(a) ECRL (b) IECRL

FIGURE 2.7: Structure of ECRL and IECRL gates [35][36].

formed by two cross-coupled inverters to store the output state. In PFAL, the NMOS transistors for evaluation are connected between the power clock and output. During the recovery phase, the charge stored in the output capacitor can completely

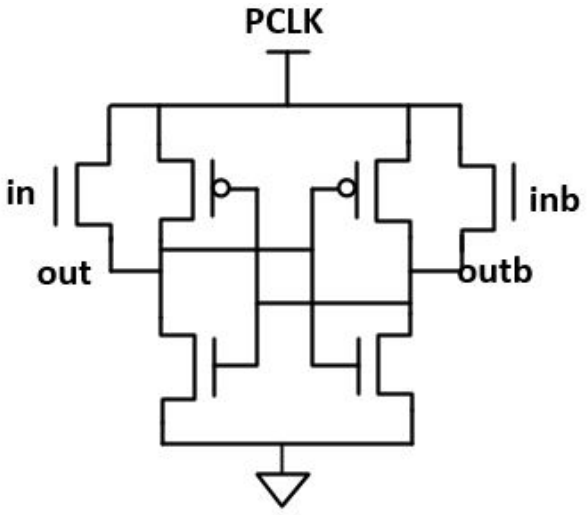



FIGURE 2.8: Structure of PFAL logic gate [30].

#### 2.1.3.3 Loss Mechanisms in Adiabatic Logic

In an ideal situation, the energy consumption of an adiabatic system should follow Equation 2.9. However, device shrinkage and the non-existence of zero-$V_{th}$ transistors means extra energy dissipation in adiabatic logic circuits.

First, with device shrinking, leakage power would not be negligible, which even dominates total power consumption when the supply voltage is scaled down to the sub-threshold voltage regime. The dominant portion of leakage current is termed the sub-threshold current, it continuously flows from the supply voltage to ground during evaluation, hold, and recovery phases, and the average current can be represented as $\overline{I_{leak}}$. Thus, the total energy consumption per cycle is as follows:

$$E_{leak} = V_{DD}\overline{I_{leak}}\frac{1}{f} \tag{2.10}$$

It can be observed in Equation 2.10 that leakage energy is inversely proportional to operation frequency. Therefore, in high-frequency designs the energy loss due to leakage current can be negligible.

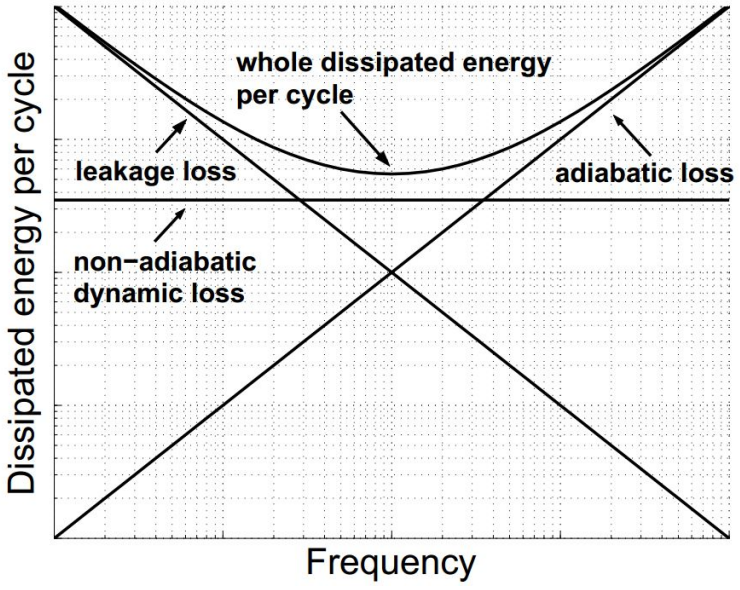



FIGURE 2.9: Energy loss of adiabatic logic with frequency [37].

Because energy can only be recovered when a charging path is formed, once the supply voltage is changed and below the threshold voltage, the pull-up PMOS device becomes switched off. Accordingly, a portion of charge is still stored in the output capacitor. In ECRL, the residual charge is possibly used in the next cycle; otherwise, it would flow to ground. Generally, this kind of energy loss is proposed as being non-adiabatic energy consumption, which can be determined by

$$E_{non-adia} = \frac{1}{2}CV_{th}^2 \tag{2.11}$$

The energy loss mechanisms are composed of three energy dissipations: adiabatic, leakage, and non-adiabatic losses.

$$E_{total} = E_{adia} + E_{leak} + E_{non-adia} \tag{2.12}$$

Moreover, non-adiabatic energy loss is independent of operating frequency. The total energy loss with different frequencies is shown in Figure 2.9. Further, the minimum energy point with a certain frequency can be observed in the figure, where the optimal operating frequency for adiabatic logic can be acquired.

#### 2.1.3.4 Impact of Process Variation on Adiabatic Logic

In CMOS circuits, process variation would be a major concern with technology scaling. Moreover, process variations in adiabatic circuits (especially variations in threshold voltage) could affect energy consumption and circuit performance.

As discussed in section 2.1.3.1, R is the resistance in the charging path of the circuit, which is mostly composed of the on resistance of PMOS transistors. Moreover, variations in threshold voltage are determinant to the value of resistance. For most of the time, the charging transistor is operated in the linear region, and the current flowing through the transistor can be expressed as follows:

$$I_D = \mu C_{OX} \frac{W}{L}((V_{GS} - V_{th})V_{DS} - \frac{V_{DS}^2}{2}) \tag{2.13}$$

where $\mu$ and $C_{OX}$ denote the mobility of the majority carrier and oxide capacitance, respectively. To minimise dynamic energy loss, the circuit is assumed to operate at a low frequency, ensuring the voltage difference between drain and source is very small. Because the on-resistance is equal to $\frac{V_{DS}}{I_D}$, it can also be represented as follows:

$$R_{on} = \frac{L}{\mu C_{OX} W}(V_{GS} - V_{th})^{-1} \tag{2.14}$$

Since $V_{GS} - V_{th}$ is inversely proportional to the on-resistance $R_{on}$, the threshold voltage affected by process variation could lead to the energy loss. Regarding leakage loss, the sub-threshold current has an exponential dependence on threshold voltage, and variations of threshold voltage could cause a change in leakage power dissipation. Similarly, non-adiabatic energy consumption is quadratically dependent on changes in threshold voltage.

The effects of parameter variations on adiabatic circuits are considered different to other circuit families. As mentioned in the previous publication [38], the threshold voltage variations on PMOS transistors have a large influence on energy dissipation. The impact of inter- and intra-die variations in PMOS transistor threshold voltages on energy loss can

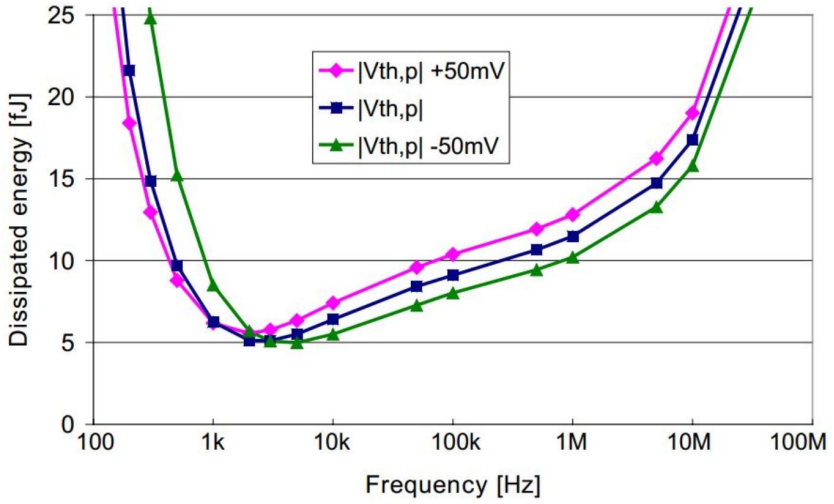



FIGURE 2.10: Effect of global Vth,p variations on the energy consumption [38].

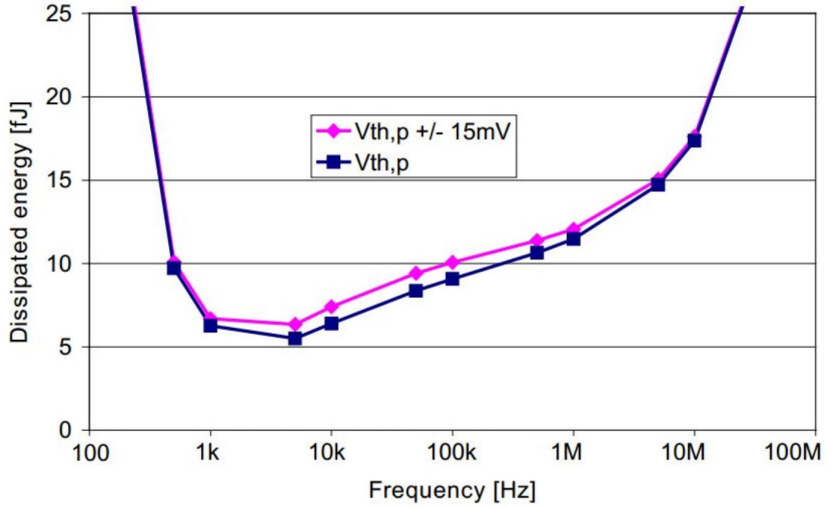



FIGURE 2.11: Effect of local Vth,p variations on the energy consumption [38].

be observed in Figure 2.10 and Figure 2.10, respectively. The simulation is implemented in an ECRL inverter with two global threshold voltage variations ($\triangle|V_{th,glob}| = +50\ mV$ and $\triangle|V_{th,glob}| = -50\ mV$) and operating frequency changes from 100 Hz to 100 MHz. In terms of the inter-die threshold voltage variation, when the circuit operates at low frequencies, positive variation decreases the total energy loss, and the reverse is also true. At medium and high frequencies, a circuit with -50 mV variation on the threshold obtains a better performance in terms of energy dissipation.

Comparatively, the effects of intra-die variations are smaller. Here, the threshold voltage difference between neighbouring transistors is set to 30 mV. In two symmetrical p-channel transistors, one threshold voltage has a +15 mV positive variation, and the other is decreased by 15 mV. According to simulation results, local variations cause more energy loss over all frequency ranges.

### 2.1.4 Serial Computing

Serial arithmetic [39] is a well-known digital design technique that minimises circuit area by processing a single bit (or digit) at a time. In word-parallel implementations, all bits of a word are processed at the same time by hardware working in parallel, whereas each bit is processed separately by the same hardware unit in bit-serial implementations.

Serial circuits are basically classified into two categories: bit-serial [40] and digit-serial [41][42] circuits. In a bit-serial circuit, both the operands are loaded bit by bit, reducing the number of data input pads to two. On the other hand, a digit-serial circuit loads one operand in a bit-serial fashion, and the other is always available for parallel operation. Bit-serial, digit-serial, and word-parallel architectures can be considered as extremities, as shown in the Figure 2.12.

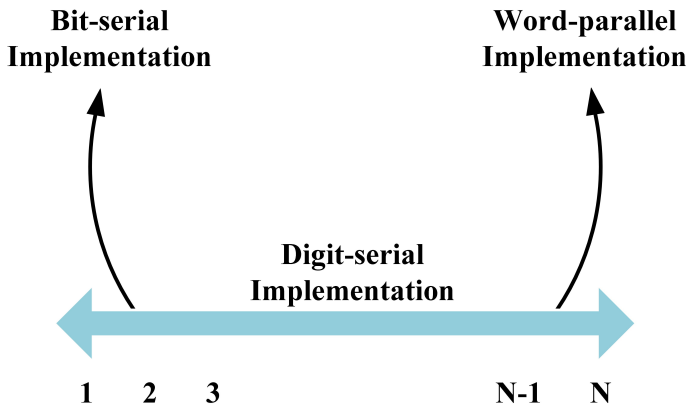



FIGURE 2.12: Indication of Bit-serial, Digit-serial and Word-parallel Implementations for a word length of N.

Depending on the supply voltage and the sampling data rates, serial computing may be more energy efficient compared with its parallel counterpart.

#### 2.1.4.1 Energy Analysis of Serial Computation

For conventional parallel designs, the dynamic and leakage energy in one clock cycle can be expressed as the following equations:

$$E_{dynamic} = \alpha C V_{DD}^2 \tag{2.15}$$

$$E_{leakage} = V_{DD} I_{off} T_d \tag{2.16}$$

Here $\alpha$ is the activity rate, $C$ is the total capacitance, $V_{DD}$ is the supply voltage, $I_{off}$ is the total leakage current, and $T_d$ is the clock period. The total energy consumption is equal to the sum of the dynamic and leakage energies, as shown in Equation 2.17.

$$E_{Total} = \alpha C V_{DD}^2 + V_{DD} I_{off} T_d \tag{2.17}$$

Serial designs cost less in terms of hardware compared with their conventional parallel counterparts. Therefore, it is expected that serial implementations will have less leakage current compared with parallel systems. In a serial circuit, auxiliary control logic would be used to maintain the system function, thereby increasing overall energy consumption in the system. Assuming $N$ digits exist in a word, the energy consumption of serial implementation in one clock cycle can be represented as follows:

$$E_{Serial} = (\alpha(C_a + C/N)V_{DD}^2 + rV_{DD}I_{off}T_d') * f(N) \tag{2.18}$$

Where $C_a$ is the capacitance of the extra circuit, $r$ is the ratio of leakage current between the parallel and serial design, $T_d$ is the reduced clock period of serial implementation, and $f(N)$ denotes the total operation cycles in the serial system. If we assume that both serial and parallel designs finish processing one sample data during the same operation period. Then,

$$T_d = T_d' * f(N) \tag{2.19}$$

In this case, the difference between the energy consumed in parallel and serial designs would be represented as Equation 2.20.

$$E_{diff} = \alpha V_{DD}^2 (C - (C_a + C/N) * f(N)) + (1 - r) V_{DD} I_{off} T_d \tag{2.20}$$

Clearly, it can be seen that serial designs mainly trade-off the balance between dynamic and leakage power consumption. For a specific system, if leakage energy consumes a large part of the total energy consumption, serial implementation will reduce the leakage energy significantly compared with its parallel counterpart. In this case, serial processing may save more energy compared to conventional parallel designs.

#### 2.1.4.2 Bit-serial Computing

The advantage of bit-serial architectures [43][44][45][46] is that all of the bits pass through the same arithmetic block in order, thereby resulting in a significant reduction in hardware complexity [18][47].

**Bit-serial adder**

For example, in a bit-serial adder as seen in Figure 2.13, each bit is fed through a 1-bit full adder, and the N-bit result would be generated after $N$ clock cycles. It can be seen that although the bit-serial design requires more operating time, the hardware complexity can be significantly reduced. Today, with the development of technology scaling and low-voltage designs, bit-serial computing becomes more significant in ultra-low-power system designs.

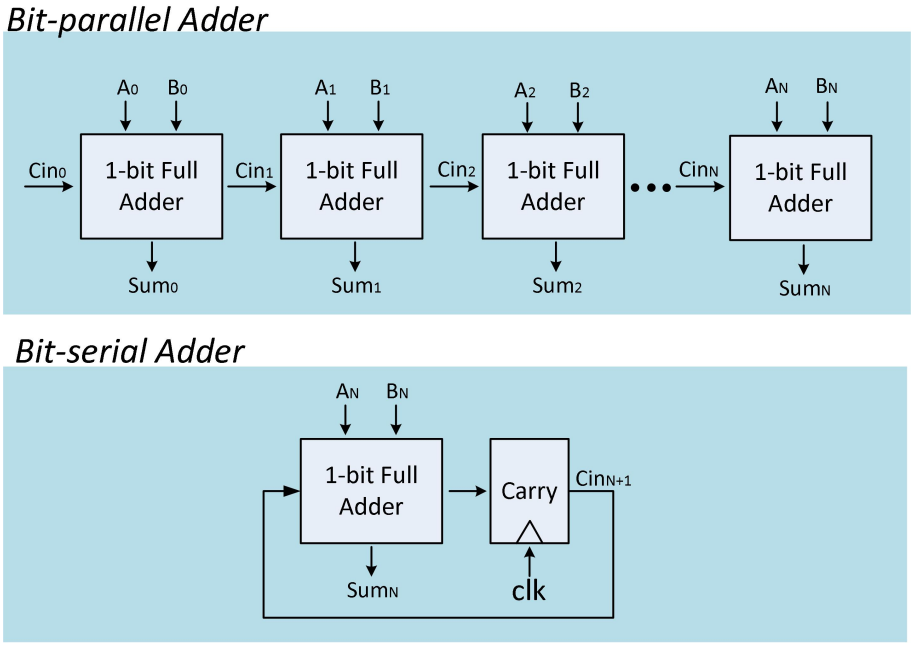



FIGURE 2.13: Comparison between bit-serial and parallel adder.

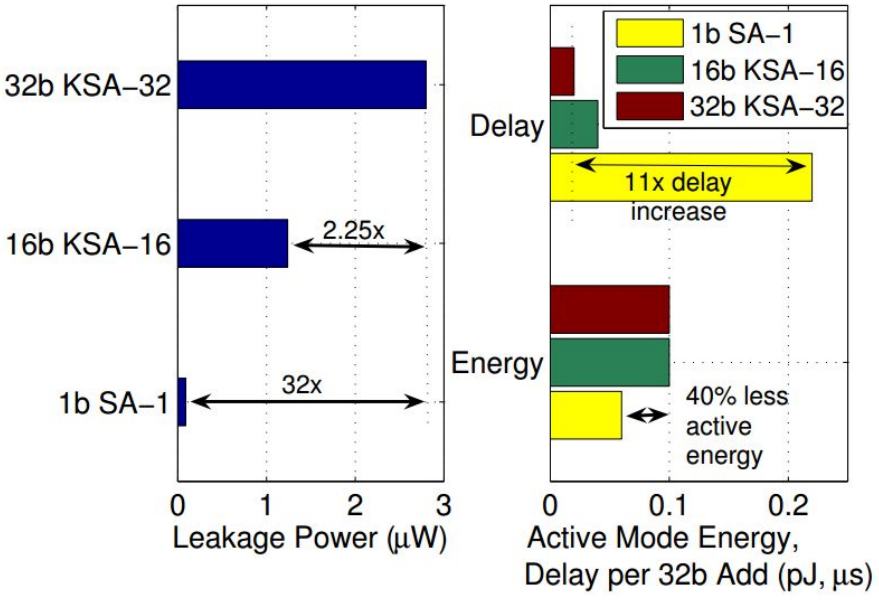
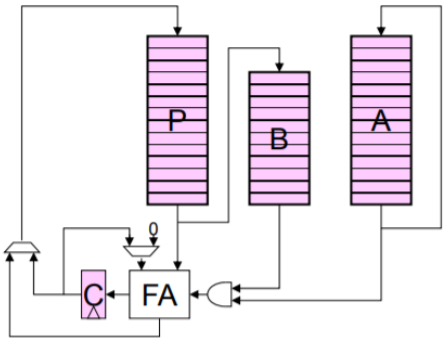



FIGURE 2.14: (a) Active mode leakage power, (b) energy (including dynamic energy), and delay per 32b added for different addition systems ($V_{DD}$=300mV) [48].

As mentioned in the previous section, in the low-voltage domain the ratio of leakage power should be increased, which benefits the implementation of bit-serial operation. Khanna [48] compared the simulation results of leakage power, active energy, and delay for both 1-bit and 32-bit adders operating at 300 mV. Figure 2.16 (a) shows that 1-bit and 16-bit adders demonstrate 32 and 2.25 leakage power reduction compared with a 32-bit adder, respectively. Further, Figure 2.16 (a) shows that the 1b SA-1 consumes 40% less active mode energy than the 32-bit KSA-32. In other words, the total energy consumed by a 1-bit system over 32 cycles is less than that consumed by a 32-bit system in one cycle. The reason for this is that complexity in terms of transistor count increases super-linearly with bit width, and leakage would dominate the total power consumption in low-voltage designs.

### Bit-serial multiplier

In terms of the bit-serial multiplier, the structure of a simple bit-serial multiplier and the corresponding pseudo codes are presented in Figure 2.15.




```
Init: P = 0; Load A,B

Repeat M times {
  Repeat N times {
    shift A,P:
    Amsb = Alsb
    Pmsb = Plsb + Alsb*Blsb + C/0
  }
  shift P,B: Pmsb = C, Bmsb = Plsb
}

(N+M)-bit result in P/B
```

FIGURE 2.15: Power comparison between the bit-serial and parallel multiplier with frequency scaling.

The bit-serial multiplier has a quite long history. In 1976, Lyon [49] proposed a bit-serial input/output multiplier that could improve the computational throughput at the expense of truncated output. Gnanasekaran [50] developed the first bit-serial multiplier that directly handles the negative weight of the most significant bit (MSB) in 2s complement representation, while n adder cells are needed for $n \times (n+1)$-bit multiplication. Dadda [51] extended Gnanasekaran's work to perform a signed multiplication, and two solutions were presented. First, the sign-extension of the input was used to produce a result which was correct up to the $2N^{th}$ cycle, where the $N^{th}$ bit represented the original sign-bit before sign-extension. This methodology was also used in [52][53]. Secondly, the negative weight of the most significant bit was directly accounted for. Later, Ibrahim analysed different structures of bit-serial multiplication in [54] and proposed an n-bit unsigned multiplier based on a single adder. Here, two registers are used to hold the

multiplier and multiplicand, and the outputs are fed to the arithmetic cell bit by bit. The multiplier register is cyclic, where the output is always fed back to its input. Another register shifts to the right every $(n+1)$ cycles. The final product is obtained from the partial-sum register after $n \times (n+1)$ cycles. Signed multiplication is based on the Baugh-Wooley (BW) algorithm [55]. The negative weight is taken into consideration rather than sign extension to save operating cycles.

Figure 2.16 shows the estimated total power dissipation for serial and parallel multipliers with the operating frequency and technology scale. At high sample rates in serial implementation, dynamic power dominates the total power because of the higher required clock frequencies. At sample rates, the low static power consumption of the implementation presents a significant advantage. Therefore, it can be seen that below a certain throughput threshold, serial computations require lower total power, and this threshold increases with technology scales due to increased leakage currents in deep-submicron CMOS.

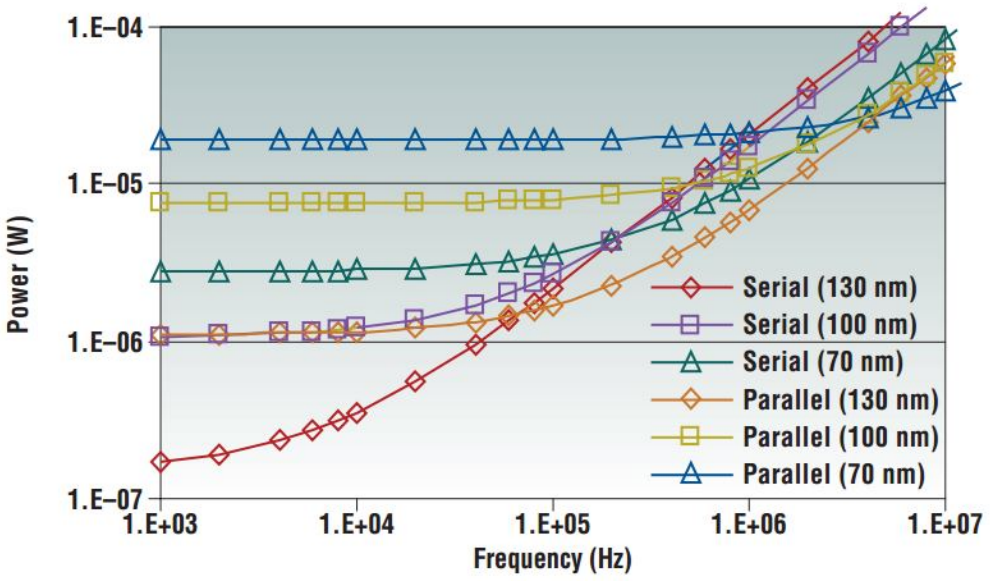



FIGURE 2.16: Power comparison between bit-serial and parallel multiplier with frequency scaling [18].

### Bit-serial processor

Based on the popularity of bit-serial arithmetic circuits, the bit-serial structure has been increasingly applied in many digital systems, such as the coordinate rotation digital computer (CORDIC) [56], LPDC decoder [47], and RISC processor [57].

A bit-serial MSP430 RISC processor [57] is highlighted here, which can outperform its parallel counterpart on the sensor benchmarks with low voltage operation. Initially, to serialise the microprocessor, various components are designed in serial structures from

top to bottom including the AND gate, ripple adder, binary coded decimal addition, and decoder. The corresponding circuit diagram can be seen in Figure 2.17.

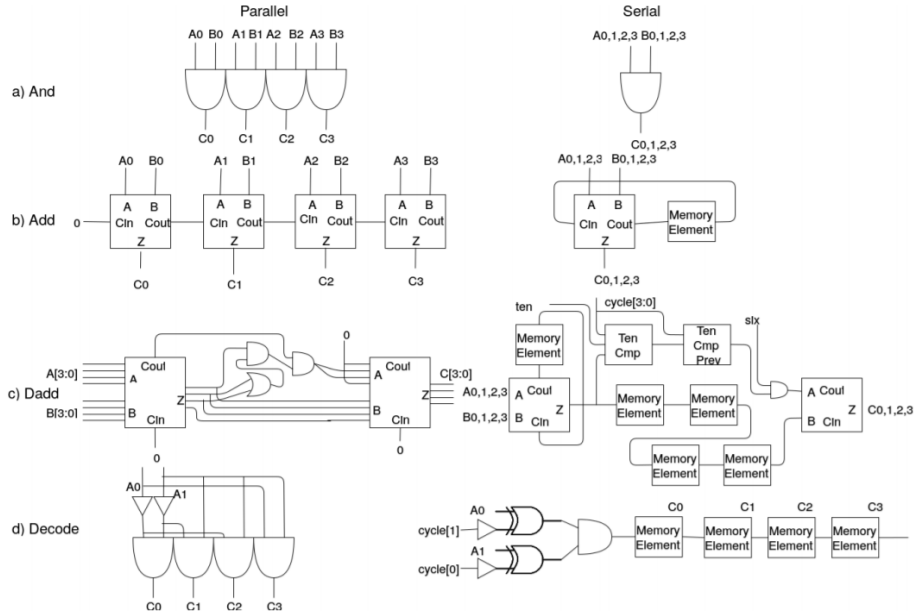



FIGURE 2.17: Serialised components in the RISC processor from top to bottom [57].

An overall area breakdown in openMSP430 by Verilog module is shown in Figure 2.18 (a), where it can be found that the execution unit consumes a dominant percentage. To be specific, the execution mostly consists of the register file and ALU, and their corresponding breakdowns can be seen in Figure 2.18 (b) and Figure 2.18 (c). In general, an MSP430 microprocessor has a three-stage pipeline architecture. The first stage is used to fetch the instructions from the program memory, and these instructions would be ready for decoding in the next stage. Here, the first two stages are summarised as the front-end. The remaining stage implements the computations and writes the result back to a data memory, which is called an execution.

According to the other sub-figures, it can be seen that the cost of this additional hardware logic in front-end modules outweighs the area benefit of the serialisation with a 1.5% area increase, as seen in Figure 2.18 (d). This is the result of the large number of parallel to serial converters required for the following executions. However, among execution units, both register files and the ALU enjoy significant benefits from circuit serialisation, as seen in Figure 2.18 (d) and Figure 2.18 (f).

Figure 2.19 (a) shows the power benefits of serial circuits over their parallel counterparts, where the advantages are increased along with the reduction of sample rates and supply voltage domains. Specifically, the maximum power saving can be up to 42%. Both Figure 2.19 (b) and Figure 2.19 (c) demonstrate that more leakage power in bit-serial design is saved with the voltage and sample rate scaling compared with that of parallel design, thereby resulting in a significant reduction in the overall power consumption.

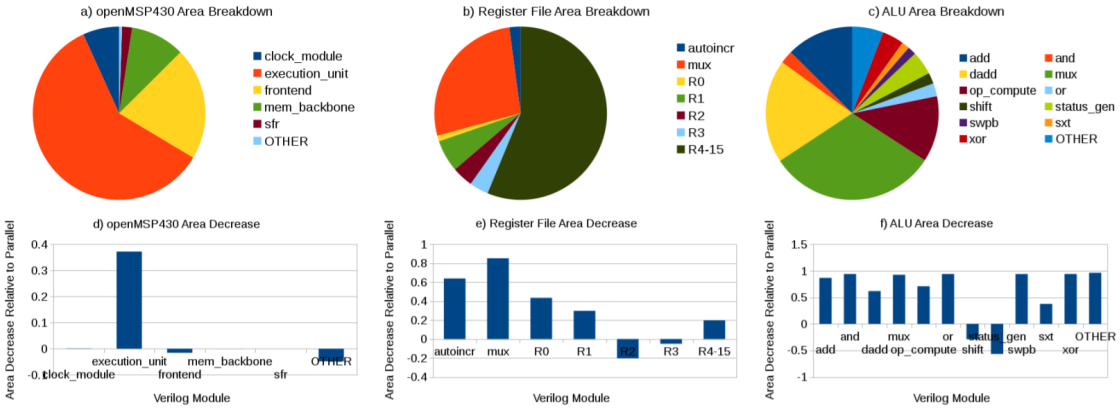



FIGURE 2.18: Serialised components in the RISC processor from top to bottom [57].

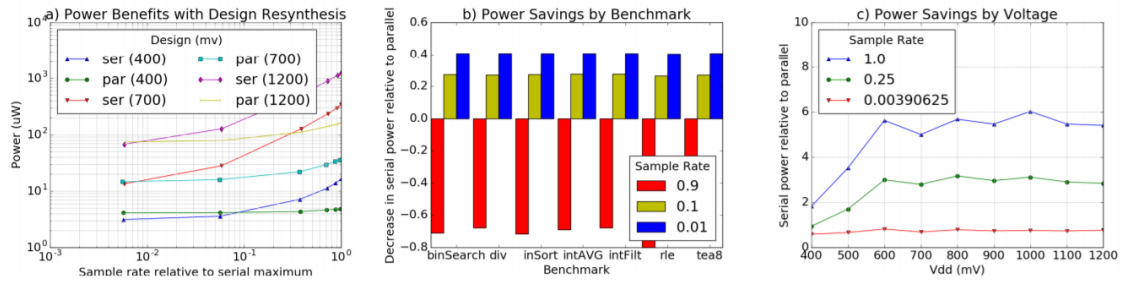



FIGURE 2.19: Power simulations versus sample rates, benchmarks and supply voltages [57].

### 2.1.4.3 Digit-serial Computing

In digit-serial arithmetic, the words are divided into digits of $d$ bits that are processed one digit at a time. The integer number $d$ is usually denoted the digit-size. This provides a trade-off between area, speed, and power consumption.

In terms of the circuit structure, digit-serial computing can be divided into two categories: conventional digit-serial circuits, and distributed arithmetic (DA) based circuits. Taking the multiplier as an example, conventional designs use combination logic to realise the generation of the partial product, while these are replaced by a look-up table in DA-based computation. The details are described as follows:

#### Conventional digit-serial arithmetic

As shown in the Figure 2.20, the conventional digit-serial multiplier is based on a carry save adder shift structure, which generates the partial product and accumulates with the previous results during each clock cycle. This digit-serial multiplier is extensively used in many signal processing applications [58][59]. Moreover, it can reduce the clock cycle from $n \times (n+1)$ to $n$ compared with its bit-serial counterpart, while approximately $\times n$ hardware overhead is used. In terms of critical path, the unsigned circuit has a full

adder, D flip-flop, and an AND gate, while one more XOR gate is used in 2's complement designs.

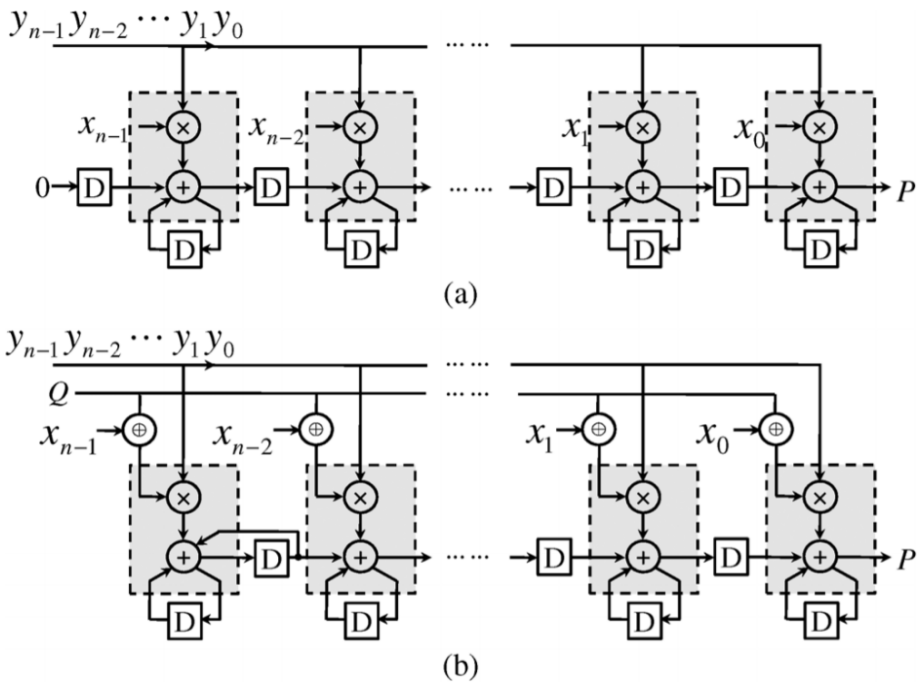



FIGURE 2.20: Conventional digit-serial multiplier. (a) Unsigned (b) 2's complement

As discussed in the previous section, digit-serial design could achieve even lower energy dissipation with technology scaling. This ideal is demonstrated on an 8-bit adder in Khan's paper [60]. Here, it is demonstrated that digit-serial computing can reduce overall energy consumption in ultra-low-voltage VLSI circuits compared with the bit-serial and conventional parallel implementations. For instance, approximately 73% and 92% energy reduction per clock cycle can be obtained with an 8-bit adder at 0.7 V voltage compared with bit-serial and conventional parallel implementations, respectively.

In his design, the total energy consumed per clock cycle against $N$ (the number of the bit being processed in one cycle) and bits produced per clock cycle at different values of the supply voltage is presented in Figure 2.21. It can be clearly seen that the total energy consumed per clock cycle is a minimum at two bits per clock cycle(N=8/2) for all values of supply voltage. Thus, the energy consumed in ultra-low-power circuits can be further reduced by shifting from word-parallel or bit-serial implementations to digit-serial implementations.

### Distributed arithmetic

Distributed arithmetic [61] is one of the traditional digit-serial operations that computes the inner product of two vectors in parallel. It has an inherent serial computation nature, and can realise multiply-less operation. Taking the multiply-accumulate (MAC) structure operation as an example, parallel implementation usually computes bits of a

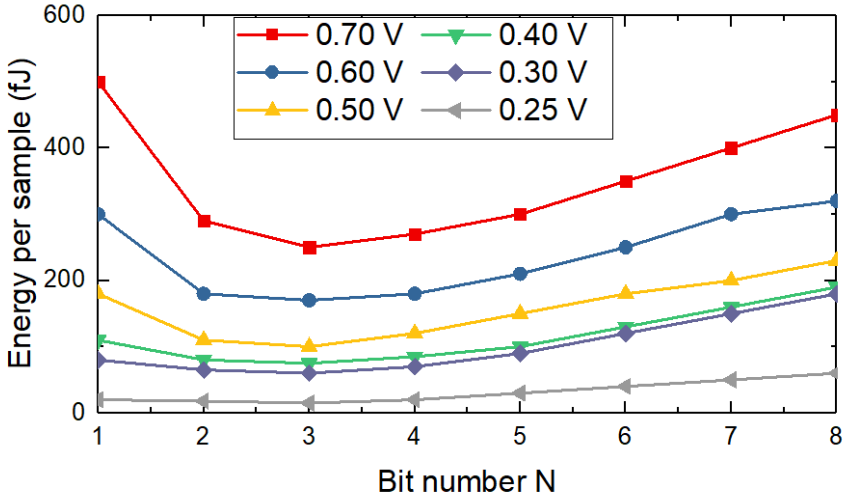



FIGURE 2.21: Energy per sample at different supply voltages against bit number $N$ (total bits per word) [60].

single input vector in parallel, while DA computes individual bits of multiple inputs serially. Therefore, when the number of filter taps exceeds the input vector width, fewer clock cycles would be consumed in a DA-based implementation. To understand how DA computation works, let us consider the following dot product:

$$y = \sum_{k=0}^{L-1} a_k * x_k \tag{2.21}$$

$x_k$ and $y_k$ denote the input and output data, respectively. $a_k$ is one of the constant coefficients. If we consider that each input data is an N-bit 2s complement binary number, and can be represented as follows:

$$x_k = -b_{k(N-1)}2^{N-1} + \sum_{n=0}^{N-2} b_{kn}2^n \tag{2.22}$$

where $b_{kj}$ is the $j^{thj}$ bit of $x_k$ and $b_{k(N-1)}$ is the sign bit of $x_k$. Combining Equations 2.21 and 2.22 together, we can obtain

$$y_k = \sum_{k=0}^{M-1} a_k \left[ -b_{k(N-1)}2^{N-1} + \sum_{n=0}^{N-2} b_{kn}2^n \right] \tag{2.23}$$

Rearranging the order of the summations in Equation 2.23, it can be modified as

$$y_k = -\sum_{k=0}^{M-1} a_k b_{k(N-1)} 2^{N-1} + \sum_{n=0}^{N-2} \left[ \sum_{k=0}^{M-1} b_{kn} 2^n \right] \tag{2.24}$$

Figure 2.22 shows the structure of a distributed arithmetic computing system, which can compute the partial product of a four-element vector X and a constant vector A. All the possible dot products are stored in a ROM in order.

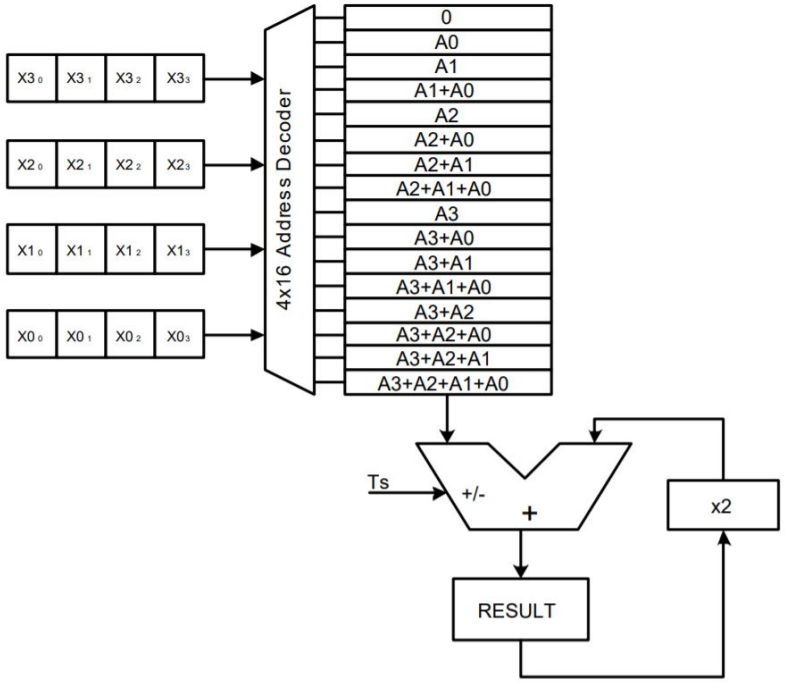



FIGURE 2.22: The structure of a DA-based computing system [62].

#### 2.1.4.4 Approximation Computing Based on Serial Structure

Serial computation is also a very promising candidate for an approximate-processing technique [18][63][64]. In a serial scheme, the word-length of computations can be increased simply by increasing the number of clock cycles. Using this approach, the computation precision can be made programmable just by truncating the input data without any extra hardware cost. Accordingly, the dynamic power could be reduced linearly with the input word-length.

An MIT research group explored the potential of serial processing in approximation computing approximately 10 years ago [62], and they proposed a DA-based approximate computation structure for low-power DSP design.

In DA, the computation is performed on a bit slice through the input data MSB first. This allows an alternative approach to realize approximate processing. The structure of

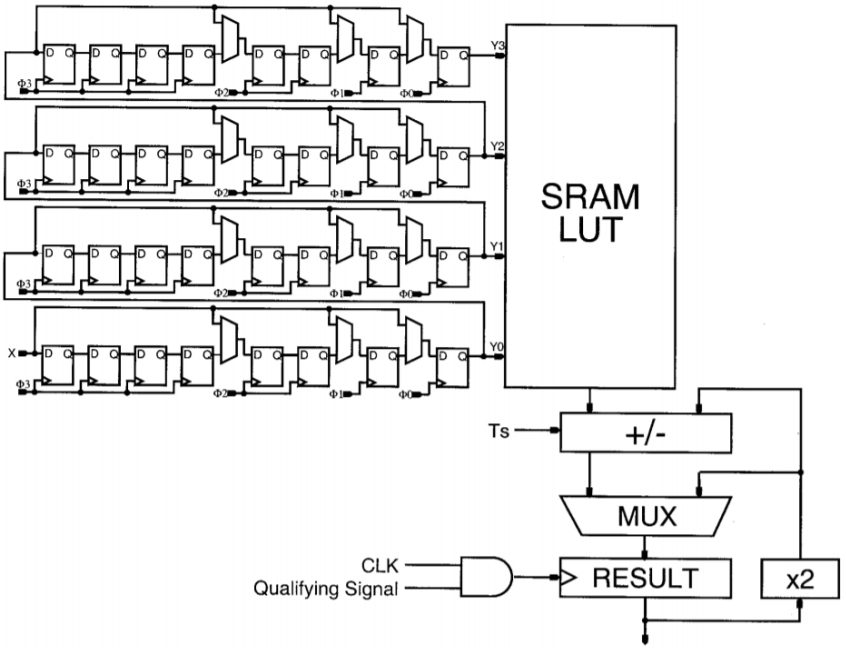



FIGURE 2.23: The structure of a DA-based approximate computing system [62].

a DA-based approximate computation is shown in Figure 2.23, the configurable input shift registers made using flip-flops and bypass MUXs addresses an SRAM lookup table of precomputed results. Therefore, the accuracy of input data can be decreased by shifting the less effective bit into the computation unit. The simulation results show that approximately 33% power can be saved with the decrease in computation accuracy.

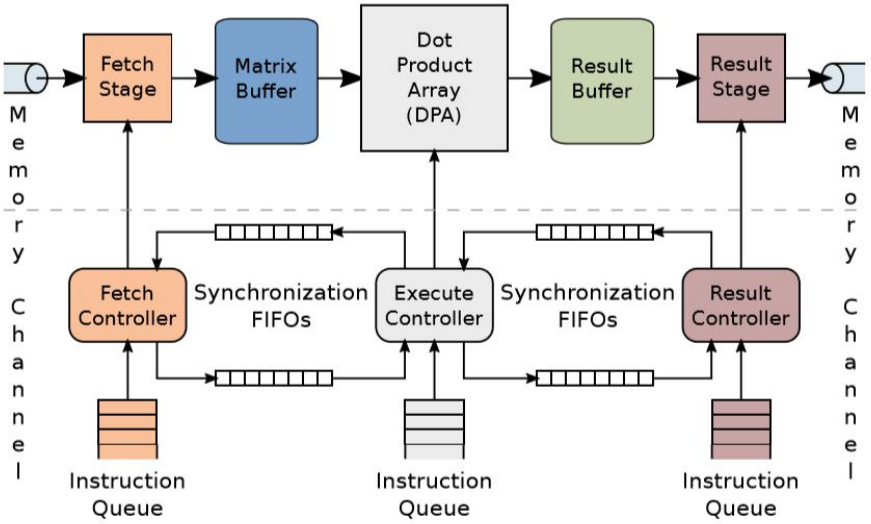



FIGURE 2.24: The hardware architecture of BISMO using serial computation [65].

Recently, the approximate serial computation attracted the IC designers' attention again. BISMO, a vectorized bit serial matrix multiplication overlay for recongurable

computing, is presented in [65]. Consder the fixed-precision matrix multiplication accelerators are not suitable for applications with variable precision since they may require multiple instances of the same accelerator. The bit-serial alternative provides the possibility to to realize matrix multiplications with any precision.

Figure 2.24 shows an overview of the BISMO hardware. The architecture is organized into three pipeline stages, fetch, execute, and result. Each stage sends data to the next stage via shared on-chip memory buffers in serial fusion. By expressing the fixed-point matrix multiplication as a weighted sum of binary matrix multiplications, the same hardware can be utilized for a range of different computation precisions.

## 2.2 Review of Timing-Error Resilient Techniques

According to the literature reviews in the previous section, it can be concluded that low-voltage serial implementation is a very promising method for power-saving. However, circuit reliability would become more serious at the low-voltage domain.

### 2.2.1 Timing Error Correction

A well-known approach for improving error resilience is timing-error correction technique. Typically, it uses double-sampling latches to detect timing errors on the critical path. Once an error is detected, an error recovery mechanism is activated to restore the system to a pre-error state.

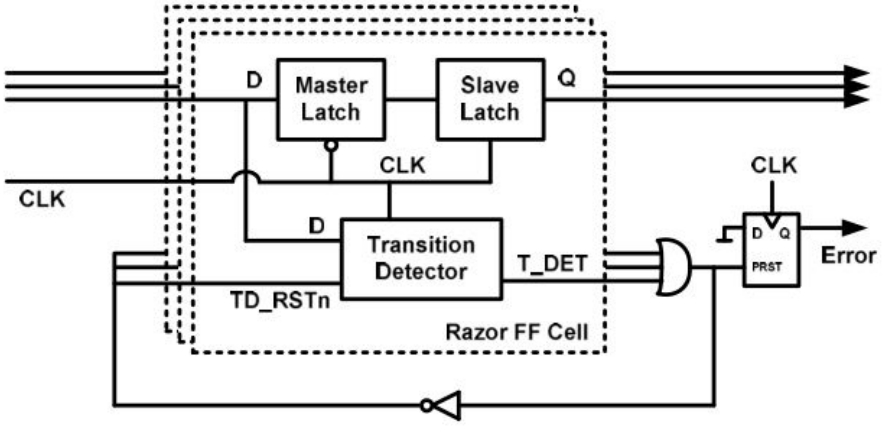



FIGURE 2.25: A basic Razor circuit [66].

#### 2.2.1.1 Razor

A typical sensor-based technique for timing error is called Razor [66], which combines circuit-level timing error detection with a microprocessor-level replay mechanism to correct the errors. As seen in Figure 2.25, an additional latch (shadow latch) is attached to the main flip-flop in the Razor, driven by a delayed clock. Accordingly, the correct data can always be fed into the shadow latch, even if a timing error occurs on the main flip-flop. If a timing violation occurs, the error can be detected by comparing the main flip-flop and shadow latch outputs, and the correct data in the shadow latch can be restored in the flip-flop in the next clock cycle.

The operation of a Razor flip-flop is illustrated in Figure 2.27. In the first clock cycle, the combination logic circuit meets the setup time and the correct data is captured by both the main flip-flop and the shadow latch. Accordingly, the error signal at the output of the XOR gate remains low, and the operation of the pipeline is not replayed. In the

following cycle, the combination logic exceeds the intended delay, resulting in a timing violation. Therefore, the main flip-flop does not successfully latch the correct data, but it is successfully captured by the shadow latch due to the delayed clock cycle. It should be mentioned that if timing errors occur regularly, the frequent replays in the Razor could affect system performance. In this case, the Razor-based technique may not be an efficient solution to overcome timing violations.

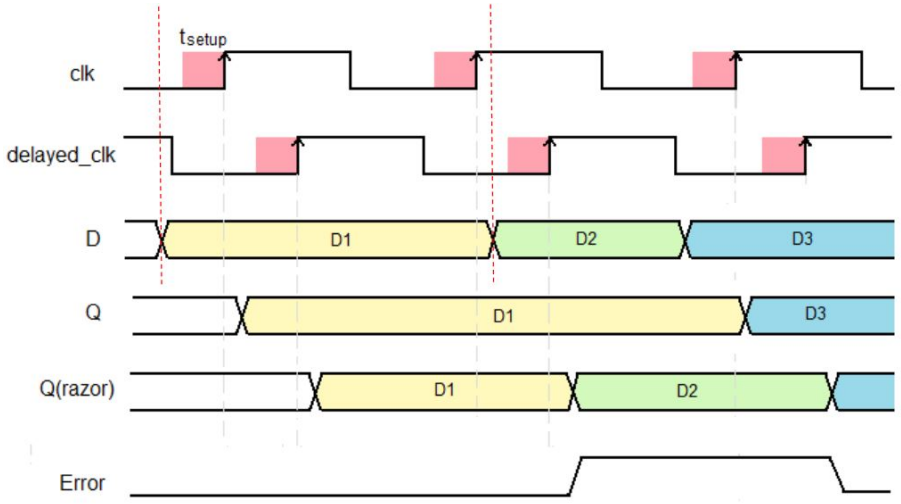



FIGURE 2.26: Error detection timing diagram in Razor [66].

Razor is also well known as an approach for implementing dynamic voltage scaling, by detection and correction of circuit timing errors in the microprocessor. Recently, the Razor flip-flop has been widely used in DSP accelerators to extend the power savings achievable by dynamic voltage scaling, trading accuracy and/or timing performance against power consumption [67][68].

An example of the Razor-based error-resilient DSP circuit is illustrated in Figure 2.27. Razor flip-flops are used to replace conventional flip-flops in the direct-form FIR architecture. In this case, there is a Razor flip-flop following each multiply-accumulate (MAC) operation, and an error flag indicates if a timing error has affected pipeline operations. Because the individual MAC operations are isolated by the Razor flip-flop stages, potential timing violations can be prevented through bypassing the output stage using an MUX and a delay register. Furthermore, the effects of each coefficient on the computation accuracy are different, and the MAC operation of the insignificant coefficients can be skipped to minimise the performance penalty. Therefore, the DSP can tolerate a small but non-zero error rate when the supply voltage is scaled down to a sub-critical operating point.

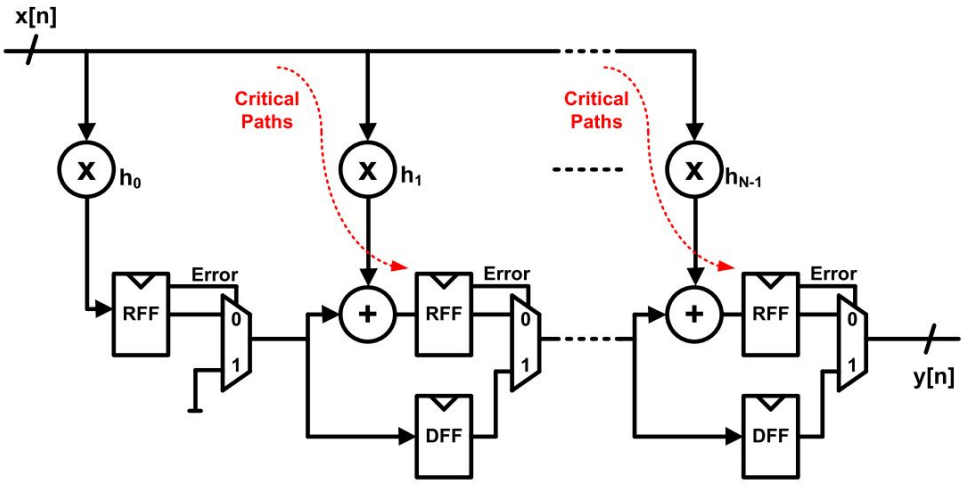



FIGURE 2.27: Error-resilient FIR filter with Razor flip-flops [68].

#### 2.2.1.2 Approximate Error Correction

With the popularity of approximation computing, another error-resilient technique has gained increasing attention, which is known as approximate error correction. Here, the basic concept is to allow intermittent timing errors but minimise their effects.

One of the efficient techniques is path delay shaping [69]. For conventional arithmetic data paths, such as adders and multipliers, the MSB usually takes place in the critical path. However, once a timing error occurs, system performance might be affected significantly due to sign-bit failure.

Figure 2.28 presents a conventional combination logic stage with the maximum delay $t_{d,max}$. The illustration of path delay shaping can be seen in Figure 2.29, the circuits paths are separated into two groups, one group of M-LSBs, which will be considered the potential critical path, and the other group of (N-M)-MSBs, which will be non-critical. Here, the LSB group is intended to fail first, but the intermittent failure does not cause an unacceptable reduction in system performance. A timing guard-band between the critical path to the LSB group end-points ($t_{d,max,LSBs}$) and the critical path to the MSB group end-points ($t_{d,max,MSBs}$) can be achieved through a number of approaches, such as modifying the circuit structure of the arithmetic units.

### 2.2.2 Timing Error Prediction

As described in the previous section, Razor flip-flops can detect timing errors in multiple paths and correct them using the re-execution mechanism. However, check-pointing and re-execution can have significant overheads, because not all the micro-architecture can

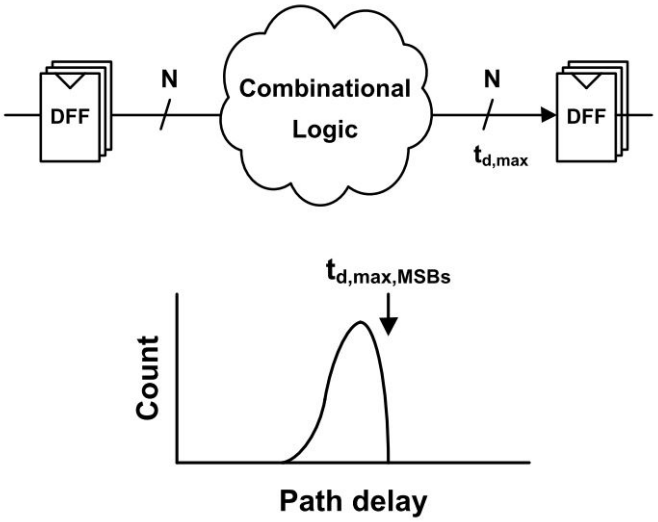



FIGURE 2.28: Conventional combinational logic stage with a critical path length of td, max, and associated diagrammatical path delay histogram [69].

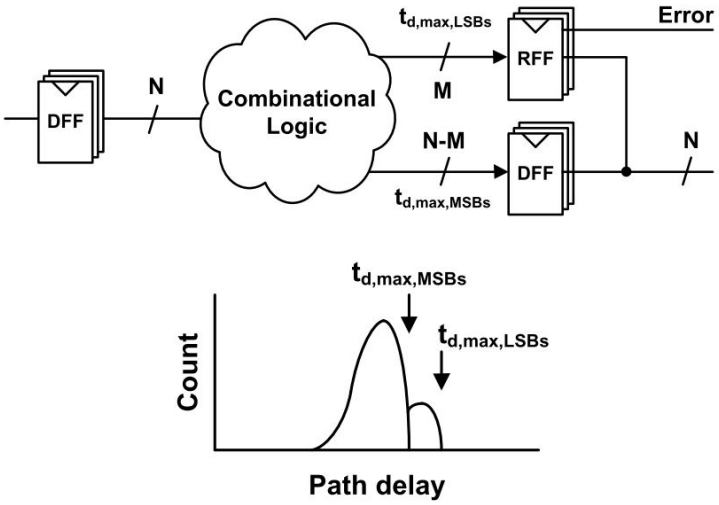



FIGURE 2.29: Illustration of the logic stage with path-shaping and corresponding diagrammatical slack histogram [69].

have prior support for speculative execution. For example, the error detection and correction (EDAC) technique is not practical for the conventional DSP.

Error prediction techniques are based on detecting signal transitions within a specified time period before the clock falling edge. If a transition is detected, a suspicious timing error might occur. Some corresponding solutions would be further implemented to prevent the errors, such as dynamic voltage scaling (DVS) or clock stretching [70][71][72].

#### 2.2.2.1 Canary Flip-flop

Unlike the error detection technique, delay-error prediction circuits can predict the occurrence of a timing error before its actual appearance. These design is called Canary FFs in the literature, [70], which detect suspicious timing errors due to a gradual increase in delay resulted from ageing effects.

As shown in Figure 2.30, an extra flip-flop is inserted, attaching the conventional flip-flop with a delay buffer. Accordingly, the delayed signal in the logic stage is captured by the canary flip-flop. Timing errors are predicted by comparing the value in the main flip-flop value with that in the canary flip-flop. Then, the error signal triggers the voltage or frequency adjustment.

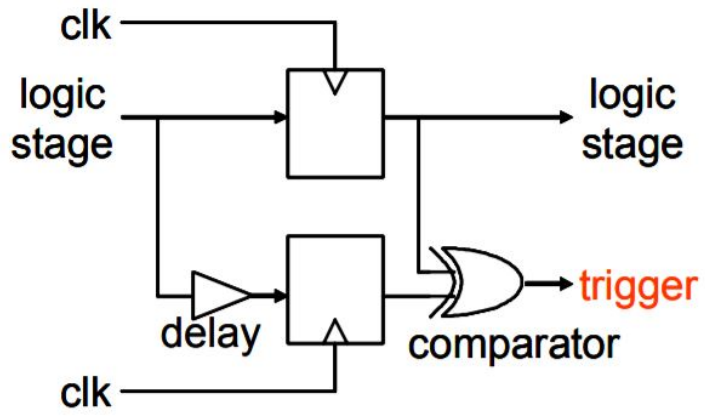



FIGURE 2.30: A canary flip-flop circuit [70].

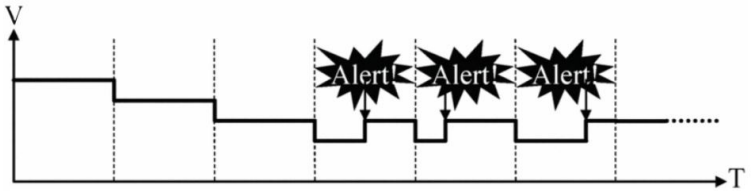



FIGURE 2.31: Canary's dynamic voltage scaling.

Figure 2.31 shows how the DVS technique is utilised with the canary flip-flops. The horizontal and vertical lines represent time and voltage, respectively. Initially, the delay element is considered; hence, the voltage is relevantly high. Because the critical path might be rarely activated owing to few variations, no timing error may occur even if the timing constraint is not met. Therefore, the supply voltage can be gradually decreased to achieve power reductions. Subsequently, the supply voltage is lower than that determined by the critical path, if an error signal is generated, this means a timing error is about to occur. Hence, control of the supply voltage is required to be triggered, and the voltage would be increased to a higher level, thereby avoiding the occurrence of suspicious errors.

#### 2.2.2.2 In Cycle Error Prediction and Prevention

In [73], a new in-situ timing error prediction and prevention technique is introduced. Here, the main concept is that by predicting timing errors through the detection of late-arriving signals at a half-path point, suspicious timing errors can be prevented using dynamic clock-gating.

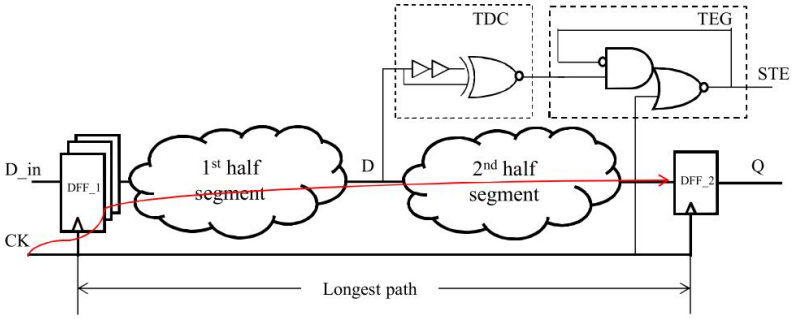



FIGURE 2.32: Timing error predication circuit

Figure 2.32 shows the circuit schematic of the error prediction system where the critical path is divided into two path segments. The circuit consists of a transition detector (TDC) and a timing error generator (TEG). The TDC contains a delay element and an XOR gate, which are used for detecting the late-arriving transition. Any transition after the clock signal falling edge is regarded as a possible timing error, and a high-level STE signal would be generated by the block TEG.

A corresponding timing diagram is displayed in Figure 2.32, where it can be seen that suspicious errors can be successfully predicted. When the signal D arrives later than the falling edge of the clock signal, the predicted error signal STE will be activated. Then, the STE would be kept high during the low clock phase, and the STE signal would be reset until the rising edge of CLK.

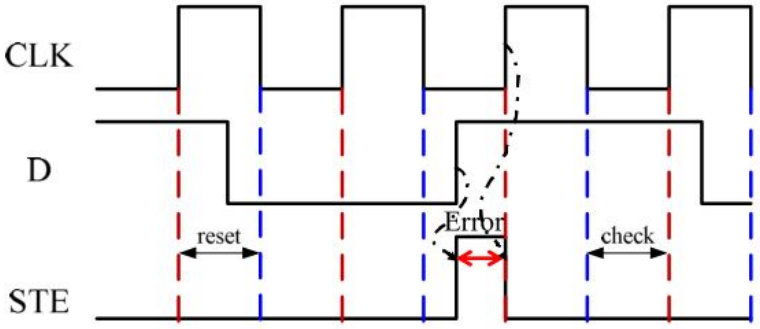



FIGURE 2.33: Conceptual timing diagram of error prediction technique [73].

Combined with state-of-the-art in cycle-error prediction methods, clock-gating techniques can dynamically isolate the critical paths once errors are predicted. Compared

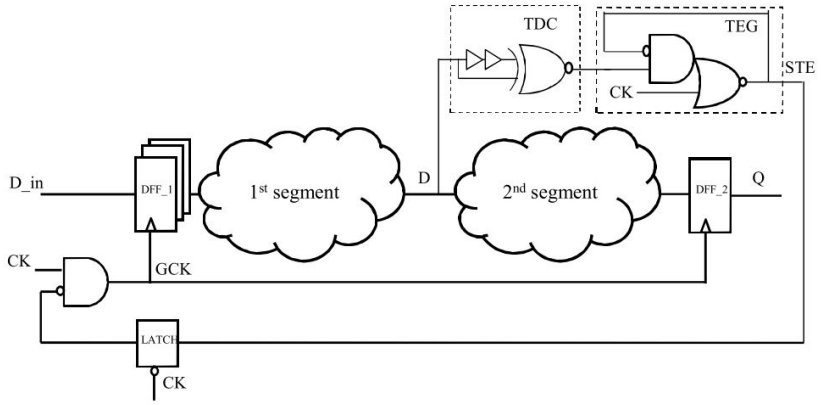



FIGURE 2.34: Timing error predication circuit with in-cycle clock-gating [73].

with prevention technique using voltage scaling, it is able to deal with errors caused by fast dynamic variations and prevent errors in the current clock cycle. Typically, the clock-gating error prevention method is easily applied in conventional digital designs. Figure 2.33 shows the circuit schematic of the error prediction and prevention system with a clock-gating technique. When a timing error is predicted, STE will be high and the clock will be gated to guarantee the correct operation with one more cycle.

#### 2.2.2.3 CRISTA

Conservative approaches for improving circuit robustness are ones that provide enough timing margins for uncertainties, such as process, voltage, temperature, and ageing variations. One efficient alternative option is critical path isolation for timing adaptiveness (CRISTA) [74]. The core idea includes the following: 1) isolate the set of possible critical paths under process variations and render them predictable; 2) guarantee that these paths are activated rarely; and 3) dynamically convert them to a two-cycle operation when they are activated. An example of a possible path delay distribution is presented in Figure 2.35. It can be seen that the original critical paths are isolated from the non-critical path with the presence of timing slack. Hence, if the original critical paths are activated rarely and predictably, then possible timing violations can be avoided by consuming an extra clock cycle.

An example of a variation-aware three-stage pipeline operation is presented in Figure 2.35, where the whole circuits are resynthesised using CRISTA methodology. The decoders $D_1$, $D_2$ and $D_3$ are used to predict the activation of the original critical paths of the individual stages. Whenever one of them is predicted, a two-cycle operation would be implemented by dynamic clock-gating.

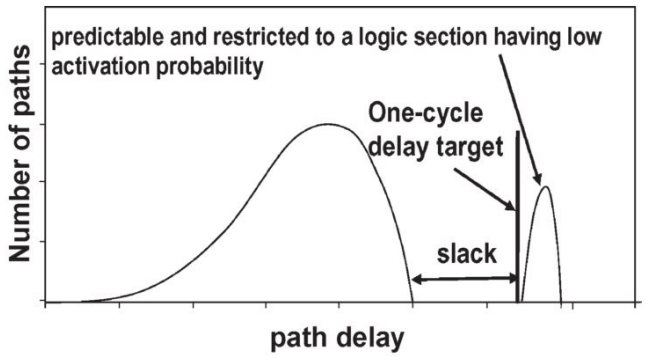



FIGURE 2.35: Path delay distribution required for CRISTA [74].

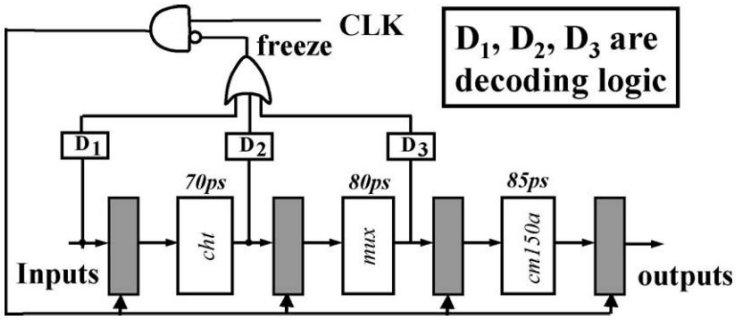



FIGURE 2.36: A case study of the Pipeline processor with CRISTA in [74].

### 2.2.3 Timing Error Masking

In error masking techniques, suspicious timing errors can be predicted and masked in the same clock cycle by tuning time references, such as clock-phase adjustment or time borrowing.

#### 2.2.3.1 Time borrowing

In the timing borrowing technique [75], if the data arrives before the set-up time of the flip-flop, it can be sampled correctly, no masking would be implemented. Conversely, if the data arrives later due to process variations or ageing effects, the flip-flop would sample correct data with a delay equal to the error masking latency.

The period of time after the clock falling edge used for error detection and masking is termed the checking period. When the first error is detected, it can be masked by borrowing the time interval in the next clock cycle equal to the recovery timing margin, which also equals the worst case timing margin. If the next clock cycle is also affected by the dynamic variations, another time interval (equal to the twice the recovery timing

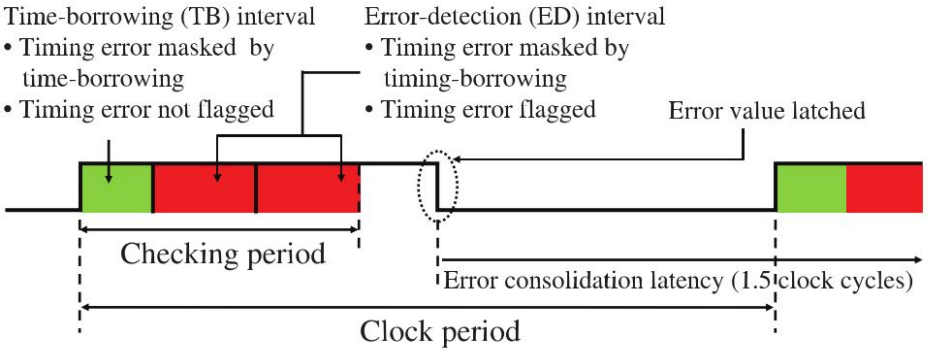



FIGURE 2.37: Conceptual timing diagram of error masking for metastability [75].

margin) would be borrowed from the following pipeline operation. Accordingly, given a checking period, $c$, and a recovered timing margin, $t$, if the error-free operation can be successfully implemented in a k-stage pipeline by time borrowing, then $c = k \times t$.

An example of a timing borrowing checking period can be seen in Figure 2.37, where it is divided into three intervals: one TB interval, and two ED intervals. The time intervals in the checking period can be classified into two types: time borrowing (TB) and error detection (ED). For a $k$-stage pipeline operation, if the number of TB intervals is $k_{TB}$, that of the ED intervals should be $k - k_{TB}$. The TB intervals can be used to mask detected errors in the $k$-stage pipeline operation without flagging to the central error control unit. However, if a timing violation occurs in the $k_{TB} + 1$-stage, ED intervals start to be borrowed with the flagging of timing errors. The detected timing error is avoided by reducing operation frequency at the system-level.

#### 2.2.3.2 Clock signal adjustment

Clock adjustment [76][77] is another common technique for realising error masking. As mentioned previously, time borrowing can prevent timing violations in the current pipeline stage. However, it would increase the path delay in the following stage, which could result in further timing errors. An efficient method of addressing this problem is clock stretching [76], which pays back the borrowed time in the next clock cycle by stretching the clock signal, as shown in Figure 2.38.

An example of error-masking flip-flop based on clock-phase adjustment was proposed in [77], termed a phase-adjustable error detection flip-flop (PEDFF). In VLSI system design, flip-flops that easily meet timing violations can be replaced by PEDFFs. The circuit diagram of a PEDFF can be seen in Figure 2.39, where an extra flip-flop $ExpReg$ attaches the conventional flip-flop $RefReg$. The detection scheme is realised by the insertion of an adjustable delay to the $ExpReg$ 's clock line. Once a timing violation is detected, it would be corrected during a one-cycle stall.

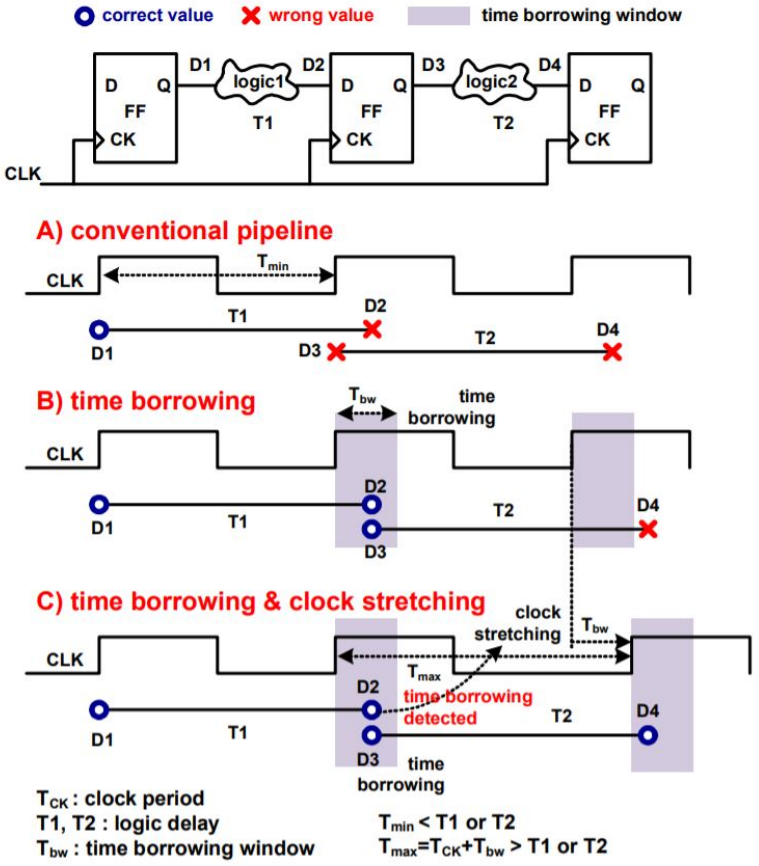



FIGURE 2.38: The operation mechanism of time borrowing and clock stretching [76].

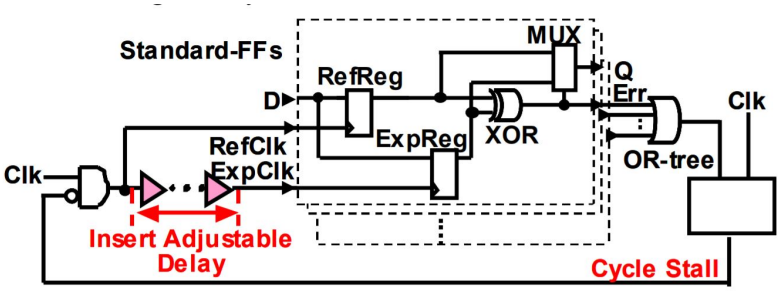



FIGURE 2.39: Phase-adjustable Error Detection Flip-Flop (PEDFF) [77].

## 2.3 Summary

The literature reviewed in the chapter includes the background about the state-of-the-art low power and timing-error resilient techniques.

Among the introduced power-saving techniques, we noticed that serial computation is regarded as an excellent technique for implementing both low power and approximate computing, which is widely used in designing energy-efficient IoT-related systems. However, the largest concern of serial computation is the too long operation time, which may

result in more energy consumption compared with parallel counterparts. Therefore, optimisation solution is needed to reduce the redundant operation cycles and demonstrated on the conventional signal processing hardware. Considering the area and power benefits brought by serial computation to digital designs relevant to the Internet-of-Things. Therefore, a new error resilient technique for serial computation is urgently required to improve robustness against process, voltage, and temperature variations.

# Chapter 3

# Error-Free Near-Threshold Adiabatic CMOS Logic in the Presence of Process Variation

In this section, the first analysis of process variation effect on the adiabatic logic combined with the near-threshold operation is provided. One of the major concerns is whether reliable performance is retained. We found that typical variations of process parameters do not affect error-free operation at the minimum-energy frequency. Monte Carlo simulations of a 4-bit full adder using ECRL structure with 0.45 V supply voltage show that, in the presence of typical process variations, energy consumption of the circuit operating at 25 MHz increases by 10.2% in the worst case while a 100% error-free operation is still maintained. The maximum operating frequency (208 MHz) is reduced to nearly half of the nominal value (385 MHz). To further improve the robustness of the adder against process variation, a bit-serial adiabatic adder is considered with an even lower power consumption per cycle.

## 3.1 ECRL Logic Operating at Near-threshold Voltage

The sensitivity to process variation is significantly increased when the supply voltage is reduced to the near-threshold region. As supply voltage gets close to the threshold voltage, both the dynamic and leakage powers decrease, but their slow-down rates are different. Leakage power shows a slower reduction. Thus its ratio in total power dissipation is increased. Since the leakage power is exponentially dependent on the threshold voltage, the variation of the threshold voltage may lead to serious power fluctuations in a near-threshold design. It is well known that near-threshold adiabatic designs can

save substantial energy. However, estimated values of energy consumption would be inaccurate if process variation is not taken into account [78]. Also, the optimal frequency, where the minimum energy point is attained, would be significantly affected.

When ECRL logic circuit operates in the near-threshold region, the supply voltage $V_{DD}$ is set slightly higher than the threshold voltage, thereby causing reduction of both adiabatic and leakage energy consumption. Accordingly, a substantial portion of energy can be saved. Because of different decrease rate with reduction of the supply voltage, the leakage energy consumption would take a larger part in the total energy dissipation. It is worth mentioned, although ECRL logic can work in the sub-threshold region, the operating frequency range is quite small and extremely sensitive to process variations. Once the switching probability is quite low, the optimal operating frequency would be very close to maximum frequency, easily affected by process variations. In contrast, near-threshold computing of ECRL circuit can not only save a large amount of energy consumption but also improve the robustness against process variations.

## 3.2 Analysis of Process Variation Effect on Near-threshold Adiabatic Operation

To demonstrate the performance of the adiabatic logic and near-threshold operation, a 4-bit ripple carry adder is implemented. The schematic of a ripple carry adder is shown in Figure 3.1. The sum and carry parts of the full adder are designed by ECRL logic circuit and the transistor-level circuits can also be seen in the figure.

### 3.2.1 Adiabatic adder without process variations

The whole design is simulated using Hspice and the functionality of the circuit can be confirmed through the simulations. In the experiment, the peak-peak supply voltage is set to 0.45 V, which is slightly larger than the absolute value of the PMOS threshold voltage (0.423 V). To compare the energy consumption of circuit operating at different voltage region, we adjust the peak-peak voltage to 0.7 V and 1.0 V respectively. Figure 3.2 shows the changes of energy dissipation with different supply voltage, it seems the near-threshold ECRL 4-bit full-adder operating at the peak-peak voltage of 0.45 V saves at least 60% energy consumption compared with the nominal voltage design when the operating frequency is below 125 MHz. Besides, optimal frequency, maximum frequency and minimum energy per cycle of the design have been figured out and shown in the Table 3.1.

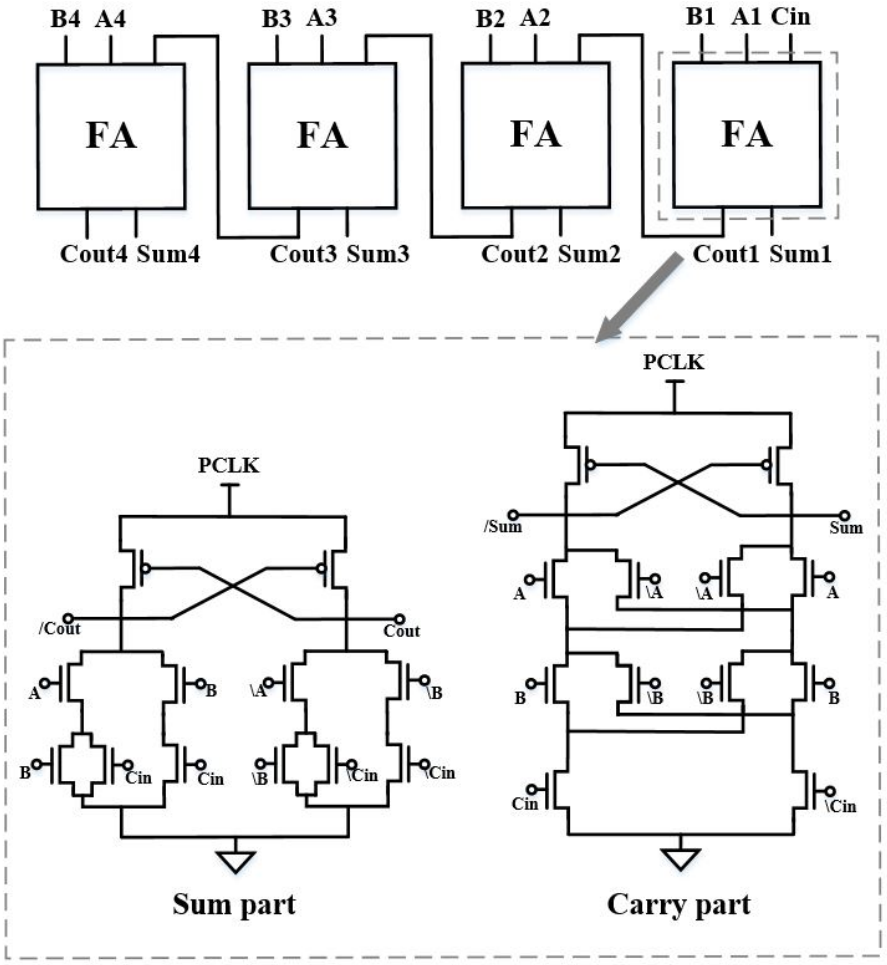



FIGURE 3.1: Topology of ECRL 4-bit full adder [33].

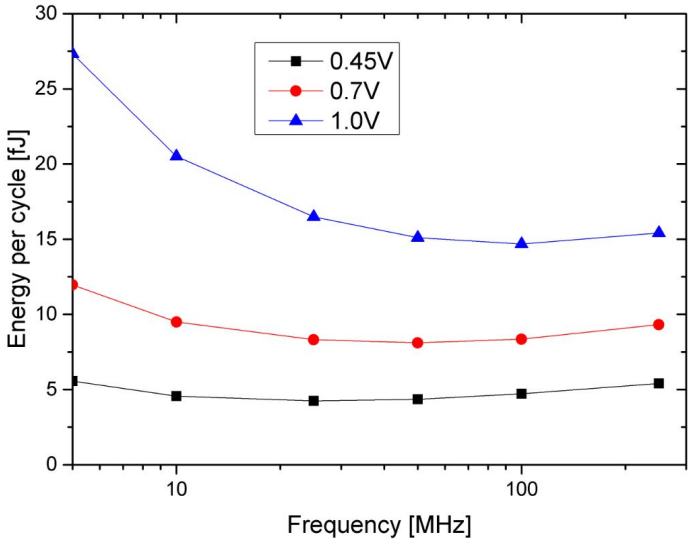



FIGURE 3.2: Energy dissipation of an adiabatic 4-bit full adder at 0.45 V, 0.7 V and 1.0 V voltage domains.

TABLE 3.1: Optimal frequency, maximum frequency and minimum energy per cycle of an adiabatic 4-bit full adder circuit operating in both super (1.0 V) and near-threshold (0.45 V) region.

| Voltage | Optimal Freq | Max Freq | Min Energy/cycle |
|---|---|---|---|
| 1.0 V | 125 MHz | 1500 MHz | 14.20 fJ |
| 0.45 V | 25 MHz | 384.6 MHz | 4.25 fJ |

TABLE 3.2: Varied Process Parameters [80]

| Parameter | Mean | Standard Deviation |
|---|---|---|
| Length | 70 nm | ±4% |
| $V_{thn}$ | 0.423 V | ±5% |
| $V_{thp}$ | -0.365 V | ±5% |
| $\mu_{effn}$ | 491 $cm^2/V.s$ | ±21% |
| $\mu_{effp}$ | 57.4 $cm^2/V.s$ | ±21% |

### 3.2.2 Adiabatic adder with process variations

To verify the robustness of full-adder design operating in near-threshold region, process variations are taken into consideration. According to the experiment results in [38], for ECRL circuits, the effect of inter-die variations is much smaller than that of intra-die variations. Therefore, only intra-die parameter variations are considered in the design. All the simulations are demonstrated based on 65 nm PTM technology [79], the relevant process variations are described in [80]. Three transistor parameters are recognized as the leading sources of process variation, including gate length, threshold voltage and mobility. The mean and standard deviation for both NMOS/PMOS transistors are shown in Table 3.1.

The influence of the length, threshold voltage and mobility on the energy consumption can be seen in Table 3.3 respectively, where threshold voltage has a stronger effect on the average, maximum and minimum energy consumption compared with other two parameters. To be specific, the variation for average energy per cycle is equal to 1.11%, but in the worst case, the process variations would lead to 10.21% extra energy dissipation.

As seen in Figure 3.3, the changes in maximum and minimum energy consumption in the presence of process variation are given, the difference with the nominal value of energy cost can be clearly observed. In general, the variation of energy dissipation is becoming more stable with the increase of the operating frequency. Besides, the optimal frequency of the whole design is largely affected by process variation. To demonstrate the reliability of near-threshold design in optimal frequency, the simulation waveforms of the fourth single-bit full-adder is shown in Figure 3.4. The input signals are power

TABLE 3.3: Effect of Process Variation on Energy Consumption of 0.45 V design operating in nominal optimal frequency

| Parameters | Avg Energy/cycle | Max. Energy/cycle | Min. Energy/cycle |
|---|---|---|---|
| Length | 4.29 fJ(+0.79%) | 4.36 fJ(+2.49%) | 4.21 fJ(−0.95%) |
| $V_{th}$ | 4.29 fJ(+0.99%) | 4.70 fJ(+10.63%) | 4.07 fJ(−4.187%) |
| Mobility | 4.28 fJ(+0.76%) | 4.43 fJ(+4.1%) | 4.18 fJ(−1.66%) |
| Total | 4.30 fJ(+1.11%) | 4.69 fJ(+10.21%) | 4.06 fJ(−4.55%) |

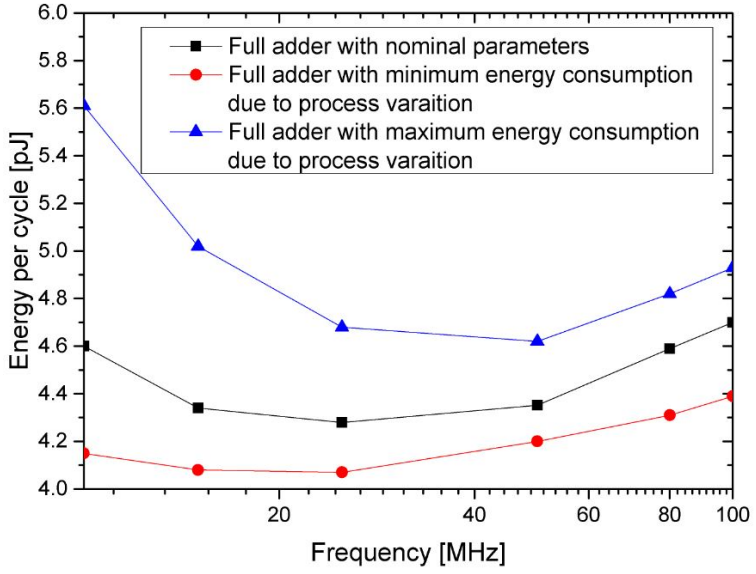



FIGURE 3.3: Energy dissipation of 0.45 V adiabatic full-adder design with nominal and varied parameters

clock, A4, B4 and the carry bit from the previous adder, while the results of the sum and carry bits can be observed in the waveforms. Besides, the orange, red and blue simulation waveforms are represented by design with nominal parameters and varied parameters(maximum and minimum energy dissipation per cycle) respectively. Clearly, the design with varied parameters can still realize correct function, therefore its reliability in the nominal optimal frequency can be confirmed.

With the presence of process variation, the error rate of the circuit operating in nominal maximum frequency (385 MHz) achieves 52.55%, and it approximately becomes zero when the working frequency is reduced to 208.3 MHz in Figure 3.5. Moreover, it is demonstrated that performance of the near-threshold adiabatic design is quite stable at the optimal frequency, even when the circuit is affected by process variation, the error rate in the optimal frequency remains zero.

As discussed before, the variation of the threshold voltage has the most significant influence on the performance of circuits. In order to further study the tolerance of near-threshold adiabatic logic against process variations, the standard deviation of Monte

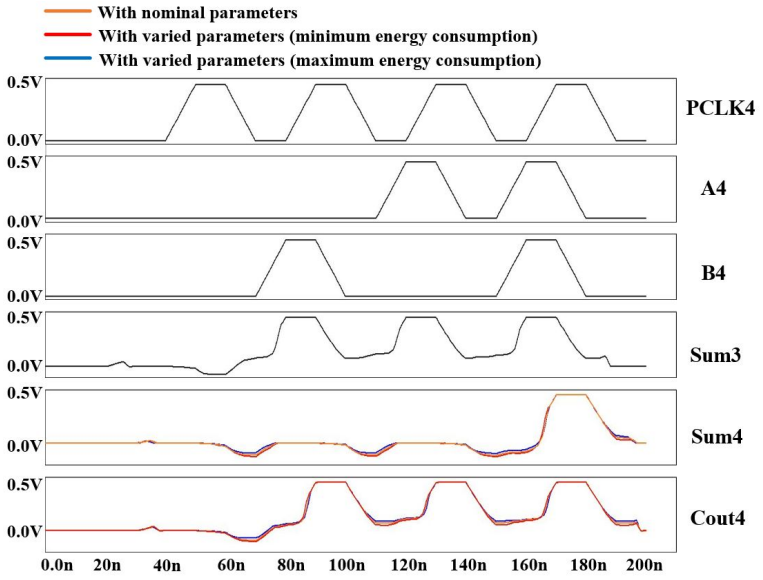



FIGURE 3.4: Simulation waveforms of an adiabatic 4-bit full adder with varied parameters at 0.45 V voltage

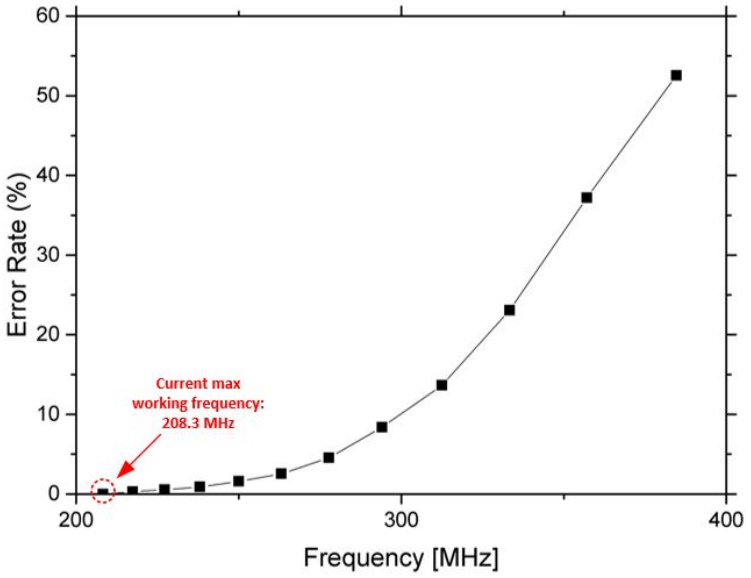



FIGURE 3.5: Error rate of near-threshold adiabatic full-adder design with process variation at 0.45 V voltage

Carlo simulations for threshold voltage is increased. According to the simulation results, when the standard deviation is set to 14%, the error rate of 4-bit full-adder operating in 200 MHz would be non-zero, approximately 0.36%. For simplicity, the waveforms of the fourth single-bit full-adder are presented in Figure 3.6 and the failed waveforms are highlighted using red cycle. With the increase of circuit complexity, reliability issues of the near-threshold adiabatic logic would be more serious.

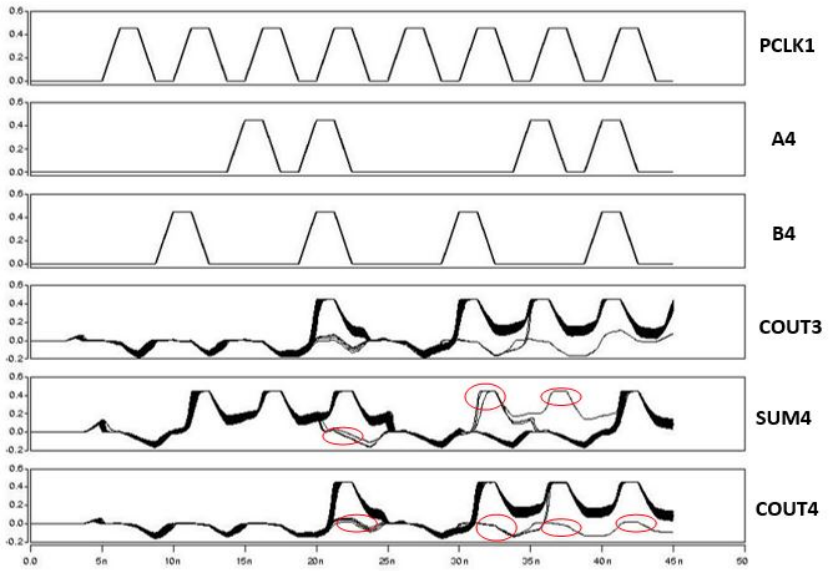



FIGURE 3.6: Monte Carlo simulation waveforms of an adiabatic 4-bit full adder with varied parameters at 0.45 V voltage

TABLE 3.4: The comparison of the 4-bit adder based on the nominal and bit-serial architecture in terms of energy consumption (operating at 25 MHz with 0.45 V supply voltage) and transistor count

| Architecture | Energy/cycle | Transistor count |
|---|---|---|
| Nominal | 4.25 fJ | 96 |
| Bit-serial | 1.10 fJ | 36 |

## 3.3 Variation-aware Adiabatic design based on Bit-serial Structure

### 3.3.1 VLSI implementation

As shown in the previous experiment results, once the threshold voltage variation becomes more serious, up to 14%, the adiabatic operation at near-threshold region would not be error-free any more. To further improve the robustness of the adiabatic logic circuit while ensuring its characteristics of low-power, a 4-bit adder is reconfigured based on the bit-serial structure. The bit-serial adiabatic adder consists of a single-bit full-adder and a flip-flop, where the flip-flop could be realized by three adiabatic logic buffers.

### 3.3.2 Simulation results

Simulation results show that, the energy per cycle can be reduced approximately four times compared with the parallel design (as seen in Table 3.4). Additionally, in the serial

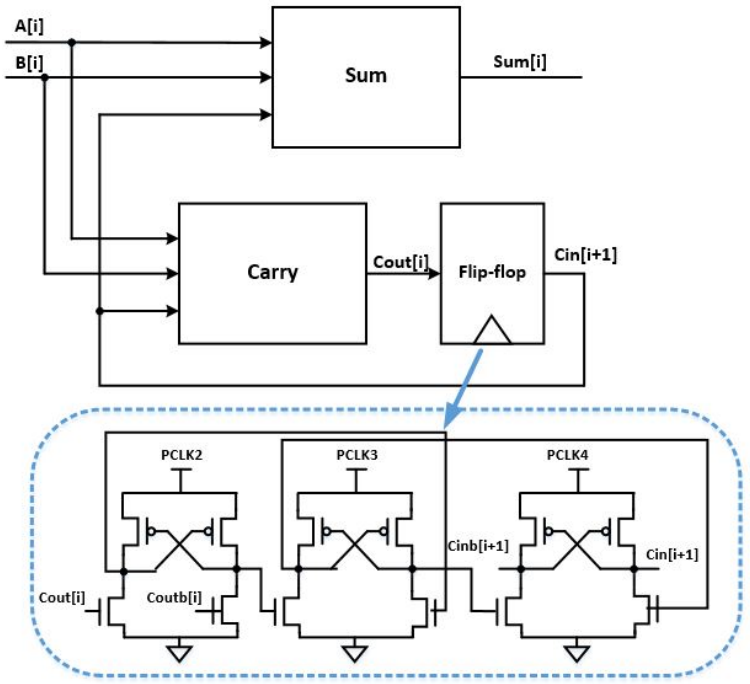



FIGURE 3.7: Adiabatic adder based on the bit-serial structure

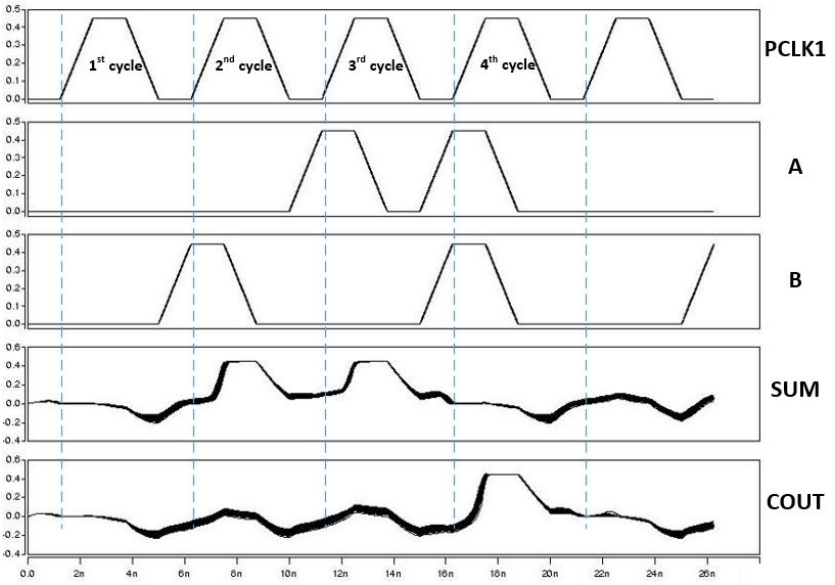



FIGURE 3.8: Monte Carlo simulation waveforms of an adiabatic 4-bit full adder with varied parameters at 0.45 V voltage based on bit-serial structure

arithmetic, 4-bit addition can be realized using only one single-bit adder, therefore the transistor count can be also significantly reduced. Only 36 transistors are required.

The bit-serial operation also improves the robustness against process variation. In the serial adiabatic adder, fewer transistors are connected in a stack, accordingly the effects of variation on the threshold voltage are alleviated. To demonstrate the robustness of the bit-serial design, Monte Carlo simulations with varying threshold voltage have been

implemented, the corresponding results can be seen in Figure 3.8. Compared with the equivalent parallel design, when the variation of the threshold voltage reaches to 14%, the bit-serial adiabatic design can still maintain the error-free operation.

## 3.4 Summary

The first analysis of process variation effects on the energy consumption and reliability of near-threshold adiabatic logic is presented in this section. As a case of study, a 4-bit full adder is tested and results show that, in the worst case, process variation could cause 10.2% extra energy dissipation at the optimal frequency, while the performance is not significantly affected. The maximum frequency is reduced by 45.8% and approximately equal to 208 MHz. In a bit-serial implementation, the energy per cycle can be further reduced while the robustness is also significantly improved, which makes it a promising candidate for future ultra-low-power digital designs.

# Chapter 4

# Bit-Serial Variable-Accuracy Computing for Ultra-Low-Power

So far, many variable-accuracy digital designs [81][82][83] have been published in order to realize energy-efficiency. However, all of the energy they save only refers to active energy, excluding static energy during the idle state. In many IoT-related devices, the sensors operating at the low sampling rates remain most of the time in the idle state with clock-gating. In this case, the static energy consumption is more important than the active counterpart. Furthermore, if a low voltage operation is adopted, leakage power would represent an increasing part of total energy consumption. Therefore, we investigate the potential of bit-serial computing, as a means of reducing the leakage power due to the small area cost of bit-serial designs.

In this chapter, a novel bit-serial signed multiplier with row bypassing is described, which outperforms the standard bit-serial multipliers as well as the main types of parallel multipliers in terms of both power and energy consumption in typical application scenarios at low voltage domains. Furthermore, by reducing the effective bit resolution of the input samples which are shifted into the proposed design, the overall energy consumption can be scaled down dynamically. The case study of a 64-point FFT implementation is described here, according to simulation results, when the effective word-length is reduced from 16 to 8 bits, the total energy consumption is reduced by up to 61%.

## 4.1 Background about bypassing technique in multiplier

The function of the bypassing technique is to turn off the arithmetic device when the bit of the multiplier is zero, thereby saving power consumption. In terms of architectural modification, the bypassing techniques can be classified into row bypassing [84] and column bypassing [85]. Based on the concept of the row and column bypassing for

the power reduction, a low power 2-dimensional bypassing multiplier [86] was proposed. Traditionally, bypassing has only been adopted in the array multipliers. Other parallel low-power multipliers, such as Booth encoding multiplier [87], cannot adopt this technique. Irrespective of whether row or column bypassing is used, their effects on power reduction are the same. In the next section we present how the row bypassing method can be adopted for implementation in serial multiplication but column bypassing can be also be implemented similarly.

## 4.2 Proposed Bit-serial bypassing multiplier

To explain the proposed bit-serial multiplier architecture, the principle of the Baugh-Wooley algorithm will be introduced first. Let two n-bit numbers to be multiplied, the multiplier Q and multiplicand M, be respectively expressed as:

$$Q = -q_{n-1}2^{n-1} + \sum_{i=0}^{n-2} q_i 2^i \tag{4.1}$$

$$M = -m_{n-1}2^{n-1} + \sum_{j=0}^{n-2} m_j 2^j \tag{4.2}$$

Where $q_i$ and $m_j$ are the bits of Q and M correspondingly while $i$ and $j$ denote the bit index. Therefore, the array partial product $P$ for bit $m_j$ can be expressed by equation 7.3. While when $m_j$ equals to zero, the partial product can be represented as $P+2^{n-1+j}$.

$$P = \begin{cases} P + \overline{q_{n-1}m_j}2^{n-1+j} + \sum\limits_{i=0}^{n-2} q_i m_j 2^{i+j}, & j < n-1 \\ P + q_{n-1}m_j 2^{n-1+j} + \sum\limits_{i=0}^{n-2} \overline{q_i m_j} 2^{i+j}, & j = n-1 \end{cases} \tag{4.3}$$

Figure 4.1 shows how this algorithm works in the case of 8-bit multiplication. The first seven rows are called PM (partial product with magnitude part) and generated by a NAND and seven AND operations. The eighth row is referred as PS (partial product with sign part) and generated by a AND and seven NAND operations. The ninth row represents the sign bit complementation. Therefore, we can simply bit-serialize the multiplication using AND, NAND and single-bit addition for accumulation.

The total operating cycle count in a conventional bit-serial multiplier is $(n+1)^2$, where the value of the product is renewed every $n+1$ cycles [88]. We observe that, if $m_j = 0$ while index $j < n-1$, the product can be accumulated by adding $2^{i+j}$ and, therefore, there is no need go through the entire sequence of $n+1$ cycles to update the partial

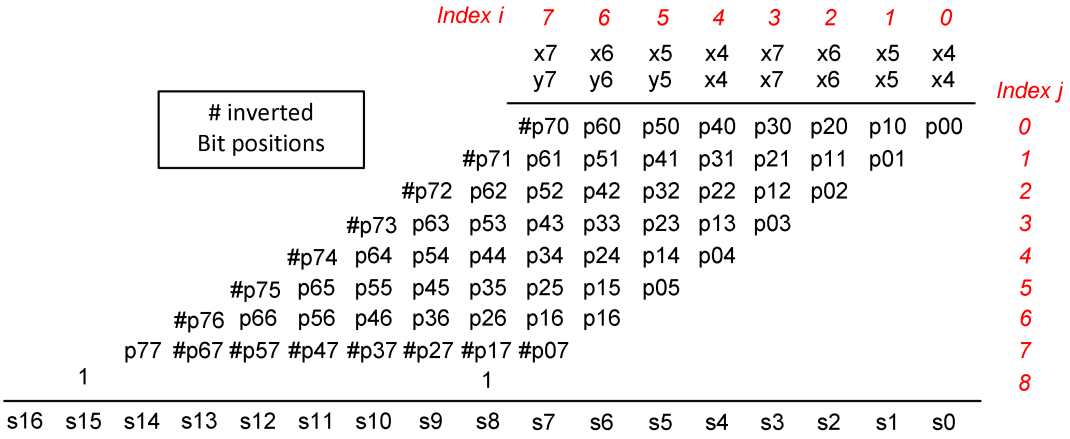



FIGURE 4.1: The case of an 8-bit multiplication using the Baugh-Wooley algorithm

product bit by bit. In other words, if there are zeros in the binary value of the multiplicand, the count of the working cycles can be reduced. The corresponding operations are presented in Figure 4.2.

In terms of hardware implementation, the proposed multiplier consists of a controller and datapath as shown in Figure 4.3. Signal *Start* initiates the multiplication sequence, the multiplier $Q$ and multiplicand $M$ are shifted into the *Datapath* bit by bit with the activation of $Next\_Q$ and $Next\_M$, to inform the outside world when the bits of the multiplier and multiplicand are required. The product is generated in serial fashion and signal *Ready* is enabled when the algorithm finishes. The responding ASM chart for the control block is presented in Figure 4.4.

## 4.3 Simulation Results

In this section, the proposed design as well as a number of traditional multipliers used in our comparisons, both serial and parallel, are synthesised using the ST 65nm CMOS process and the power and energy consumption are measured using HSPICE. The performance characteristics of the proposed serial multiplier with row bypassing are calculated with respect to the gate count, average power, total energy consumption and variable precision (energy-scalable) computation and compared against the following multipliers implemented using the same ST 65nm technology: standard bit-serial multiplier [88], parallel Baugh-Wooley multiplier [55], parallel bypassing multiplier [86] and Booth multiplier [87].

**Algorithm 1** Calculate $P = Q \times M$ in the bypassing bit-serial way based on Baugh-Wooley algorithm

$Q = -q_{n-1}2^{n-1} + \sum_{i=0}^{n-2} q_j 2^i$

$M = -m_{n-1}2^{n-1} + \sum_{i=0}^{n-2} m_i 2^i$

**for** $j$ *from* 0 *to* $n$ **do**
  **for** $i$ *from* 0 *to* $n-1$ **do**
    **if** $j < n-1$ *and* $m_j = 0$ **then**
      $P = P + 2^{n-1+j}$ ▷ Row bypassing
      **break**
    **else if** $(j < n-1$ *and* $i < n-1)$ *or* $(j = n-1$ *and* $i = n-1)$ **then**
      $P = P + q_i m_j 2^{i+j}$ ▷ Partial product accumulation
    **else if** $(j < n-1$ *and* $i = n-1)$ *or* $(j = n-1$ *and* $i < n-1)$ **then**
      $P = P + \overline{q_i m_j} 2^{i+j}$ ▷ Sign complementation 1
    **else if** $((j = n)$ *and* $(i =$ 1 *or* $n))$ **then**
      $P = P + 2^{i+j}$ ▷ Sign complementation 2
    **end if**
  **end for**
**end for**
**return** *Product* $P$

FIGURE 4.2: Algorithm of the proposed bit-serial bypassing multiplication.

### 4.3.1 Low area cost

In the proposed design, the bit-serial approach reduces the number of the combinational logic gates in the multiplier at the cost of some additional registers required by multi-cycle computing. Consequently, a bit-serial multiplier is more compact than its parallel counterparts, as it requires fewer adders and interconnects. The transistor count of the parallel Baugh-Wooley, bypassing parallel, Booth, conventional bit-serial and proposed bit-serial multipliers with different word-lengths can be acquired in the Hspice simulations, and equivalent gate-count comparisons are summarised in Table 4.1 where 100% represents the size of the conventional bit-serial circuit as a reference.

The complexity of parallel multipliers is approximately geometrically proportional to the word-length, but it can be seen that bit-serial multipliers do not grow significantly with the word-length. Although the proposed design increases slightly the complexity of the control block, the advantage in the area cost is still significant as it represents a saving of 366% and 38% compared with parallel Baugh-Wooley and Booth multipliers

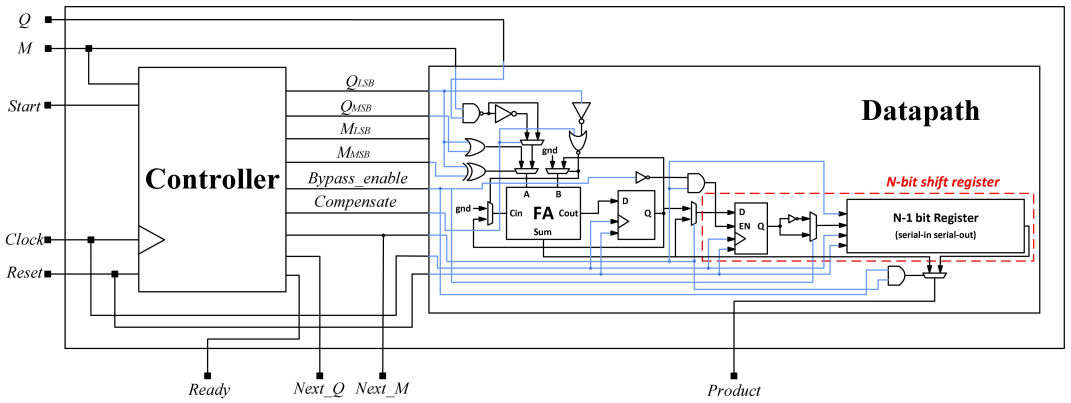



FIGURE 4.3: Structure units of the proposed bit-serial multiplier

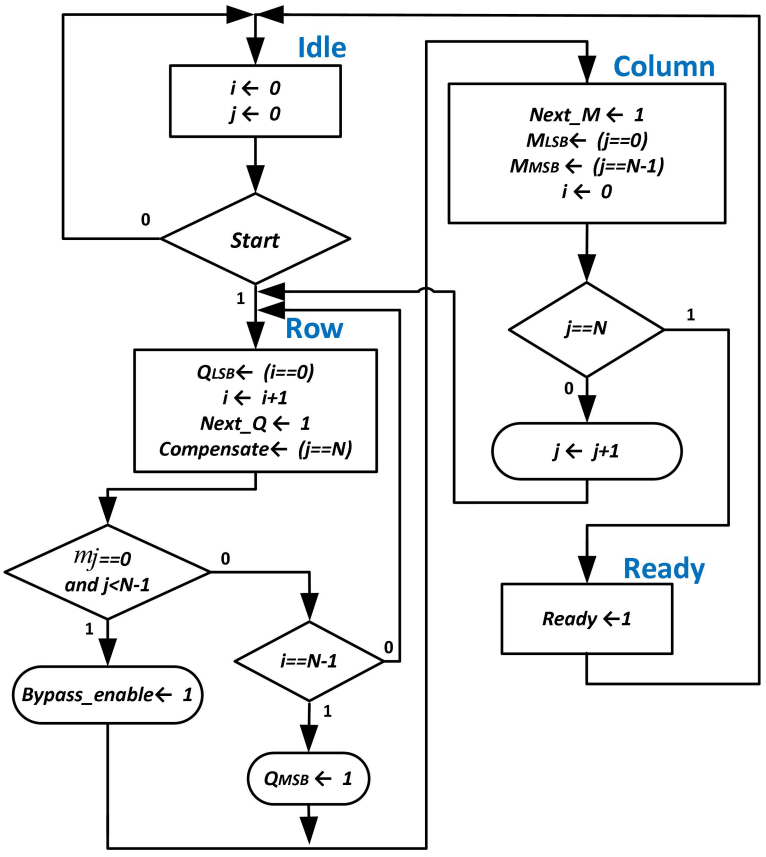



FIGURE 4.4: ASM chart of the controller.

TABLE 4.1: The gate count comparisons of conventional parallel, bit-serial and proposed bit-serial multipliers in terms of different word-lengths

| Structure | 8-bit | 12-bit | 16-bit |
|---|---|---|---|
| Parallel | 472 (130%) | 1111 (282%) | 2023 (466%) |
| Parallel with row-bypassing | 568 (157%) | 1327 (337%) | 2407 (555%) |
| Booth | 329 (91%) | 461 (117%) | 598 (138%) |
| Conventional Bit-serial | 323 (89%) | 355 (90%) | 395 (91%) |
| Proposed Bit-serial | 362 (100%) | 394 (100%) | 434 (100%) |

for 16-bit computation correspondingly. It is also worth noting that compared with the conventional serial multiplier, the proposed bypassing design increases the gate count by less than 12%.

To measure the power and energy consumption, extensive Hspice simulations have been carried out. All the results presented below have been obtained using, for each test, the averaged results for 1000 randomly generated pairs of numbers to be multiplied.

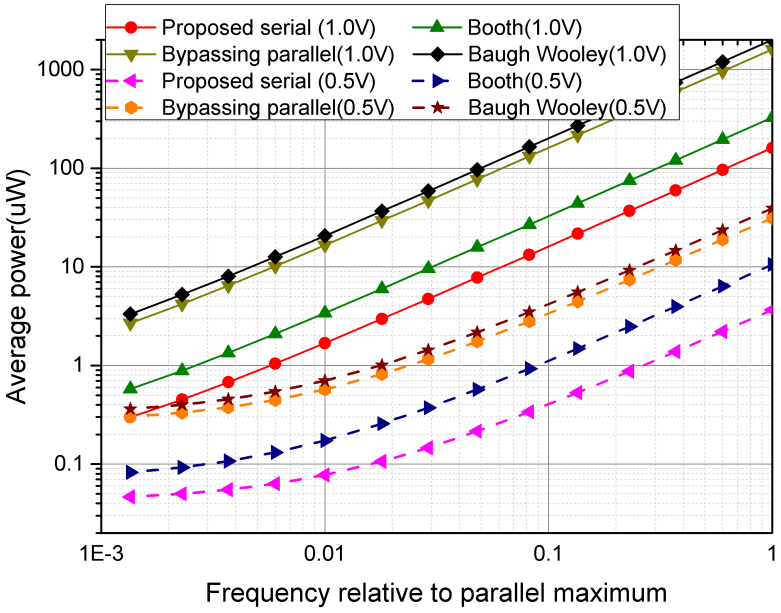



FIGURE 4.5: Average power consumption comparison vs frequency at super- and near-threshold regions (1 V and 0.5 V supply voltage).

### 4.3.2 Low power consumption

As mentioned above, serial designs have much smaller gate counts and interconnect capacitance compared with their parallel counterparts. This advantage allows serial operations to consume significantly less power at a fixed frequency. Figure 4.5 shows the average power of the proposed and other state-of-the-art multipliers versus frequency at both super and near-threshold voltage regions respectively. The frequency on the x-axis is normalized to the maximum frequency that can be processed by the Baugh-Wooley multiplier. It is evident that the proposed bit-serial design presents a significant power-saving compared with all the other types of multipliers, both serial and parallel, at both the super- and near-threshold supply voltage.

However, as parallel multipliers operate faster and require less time to complete a multiplication, it can be argued that a fairer power consumption comparison should be based on a scenario where operating frequencies of parallel designs are slowed down such that

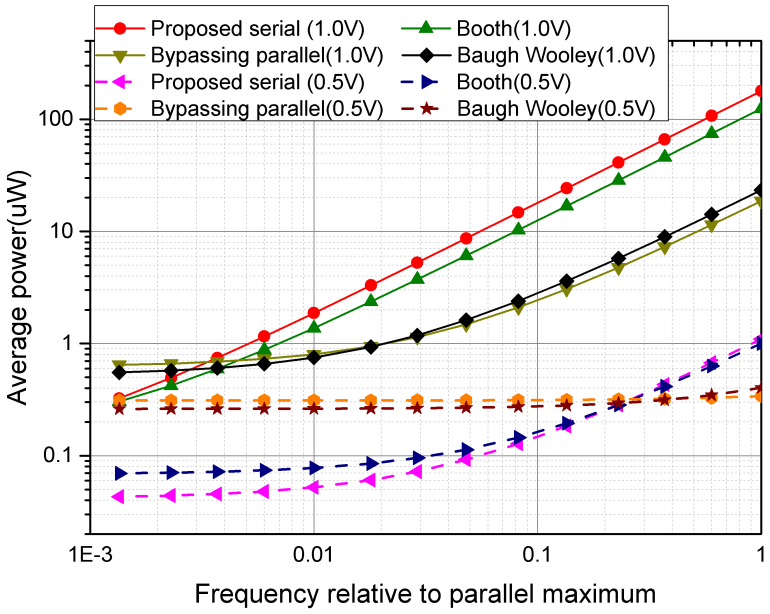



FIGURE 4.6: Average power consumption comparison vs sampling rate at super- and near-threshold regions (1 V and 0.5 V supply voltage).

all types of multipliers calculate their results at the same speed. Figure 4.6 shows the results of that scenario. The average power consumption of the proposed and other state-of-the-art multipliers are shown with the sample rate scaled to achieve the same computation time at both super and near-threshold voltage regions respectively. The sample rate on the x-axis is normalized to the maximum sample rate that can be processed by the conventional bit serial design, where the maximum frequencies are 1190 MHz and 33 MHz for 1.0V and 0.5 V voltage operations respectively. It is worth noting that all the designs are clock-gated during idle intervals. Simulation results show that the power benefits of the proposed serial design at a given sample rate is greater at the near-threshold supply voltage. This is because the leakage power consumes a higher proportion of the overall power at low voltages. For both super and near-threshold operations, at a high sample rate, the dynamic power dominates the total power because of higher required clock frequencies. At low sample rates, it is trivial that the serial implementation's low leakage power consumption has a significant advantage. It can be seen that the serial designs have lower total power consumption at the near-threshold supply voltage with the proposed bypassing bit-serial design being the best, better than the conventional bit-serial, as the proposed design skips redundant cycles and keeps the threshold sample rate (where less energy is consumed compared with parallel counterparts) in an acceptable range for conventional circuit devices. Specifically, the results show that the proposed bit-serial design uses up to 44%, 38%, 84% and 86% less power than the conventional bit-serial, Booth, bypassing parallel and Baugh-Wooley designs at near-threshold voltage respectively.

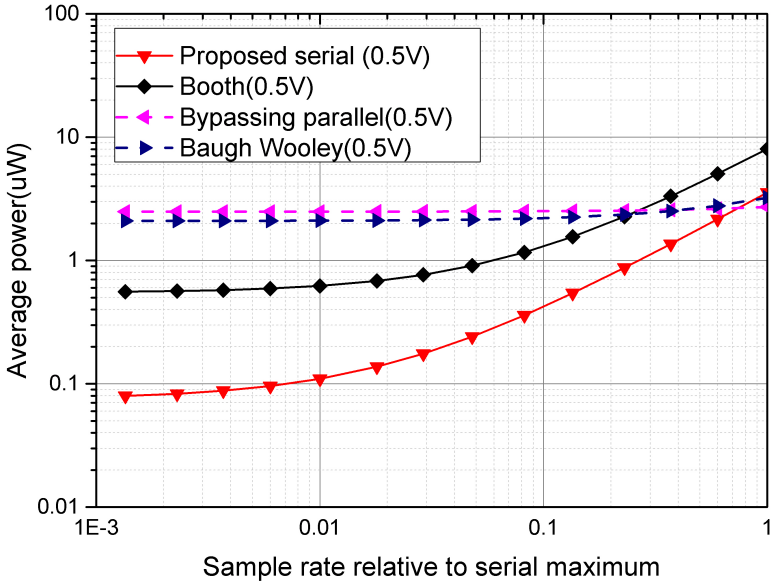



FIGURE 4.7: Average power consumption comparison vs sampling rate of the proposed design using eight processing elements an parallel designs at near-threshold region (0.5 V supply voltage).

The next scenario in our power consumption analysis tests the performance of the bit-serial design arranged as a vector multiplier using multiple processing elements driven by a single controller. It turns out that the proposed bit-serial design with bypassing using a vector of 8 processing elements and operating at 0.5 V near-threshold supply voltage outperforms, in terms of the average power consumption, all the standard parallel designs by about an order of magnitude when the calculation time is fixed and operating frequencies adjusted accordingly. This is shown in Figure 4.7. Calculations here were performed using 16-bit numbers.

It could also be observed that, the parallel bypassing multiplier consumes more power than the proposed bit-serial design for the same multiplication time and near-threshold region at all sampling rates. This is an interesting observation which can be explained by the much lower leakage power of the serial design at near threshold supply voltage.

Further, the proposed design was compared with a conventional bit-serial multiplier without row bypassing but with the operating frequencies adjusted to achieve the same multiplication times. Figure 4.8 shows that the proposed design outperforms the conventional bit-serial multiplier in terms of average power consumption for 8-bit, 12-bit and 16-bit arithmetic. It should be mentioned that the reduction of effective word-length would not save much power consumption in a standard bit-serial multiplier, that is because redundant clock cycle cannot be reduced without bypassing technique and

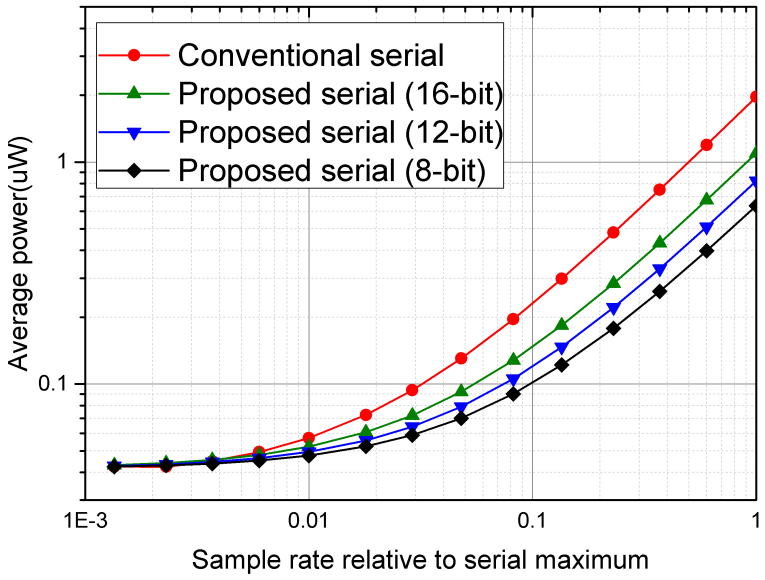



FIGURE 4.8: Average power consumption comparison vs sampling rate of the proposed design and standard bit-serial at near-threshold region (0.5 V supply voltage).

a majority of power is actually consumed in control blocks. Accordingly, power consumption for standard bit-serial design with 8-bit and 12-bit accuracy is not presented here.

### 4.3.3 Energy-efficiency and variable-accuracy

The additional benefit of the proposed bit-serial bypassing scheme is that it can also be used to trade bit-precision with overall energy consumption. In the proposed Algorithm, once $q_j = 0$ and $j < n-1$, the process of the partial product accumulation can be reduced from $n$ cycles to 1 cycle. In other words, the simple zeroing of leading or trailing bits of the multiplicand, the energy consumption of the system can be decreased in proportion to the bit precision, even if the bits of the multiplier remains unchanged. A simple modification of the controller can implement this bit-precision reduction technique and, in this way, the accuracy can be traded dynamically for lower energy consumption should the power supply levels become low. Such scenarios are typical in energy-harvester powered sensors.

Figure 4.9 presents the energy comparisons between conventional and proposed bit-serial multiplier at the minimum energy point (around 0.5 V supply voltage). Also, it demonstrates the variable accuracy behaviour of the proposed design. It can be seen that the proposed bit-serial multiplier can save around 44% energy through the bypassing technique. For the variable-accuracy operation, the energy consumption changes approximately linearly with the effective bit-width as expected. Approximately 23%

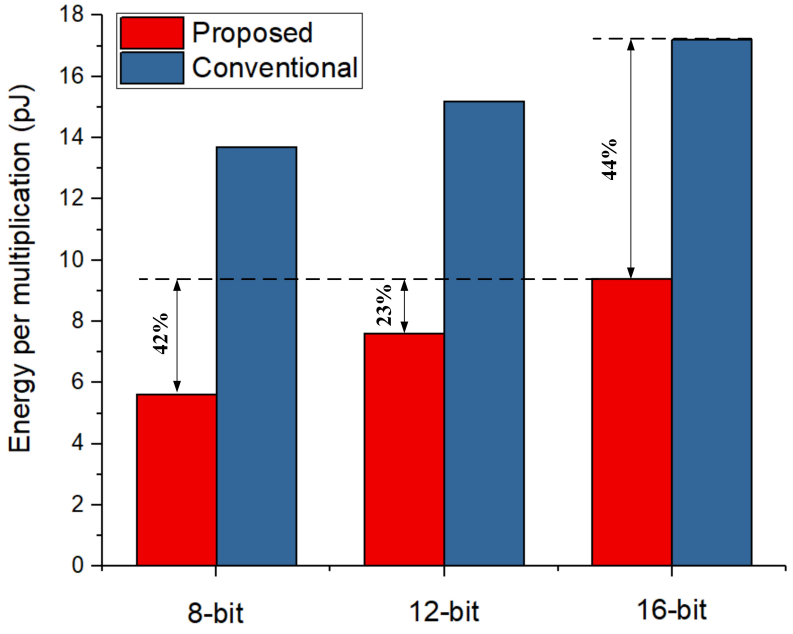



FIGURE 4.9: Energy comparison with convention and proposed bit-serial multiplier with variable-accuracy on near-threshold region

energy can be saved when the effective word-length is reduced from 16 to 12 bits while the 8-bit computation scales down the energy dissipation further, by 42%.

## 4.4 Demonstrate on FFT Implementation

### 4.4.1 FFT Datapath

The block diagram of the bit-serial FFT processor is shown in Figure 4.10, using a conventional radix-2 butterfly architecture. During a butterfly computation, two complex number is read from a data memory in serial fashion, and implement a butterfly operation with corresponding coefficients from ROMs. With the assistance of scalable shift memory, computation accuracy can be adjusted in the butterfly datapath to enable adaptive-accuracy operation.

In this design, The radix-2 butterfly datapath is adopted as seen in Figure 4.12, which contains a complex multiplication, followed by complex-value addition or subtraction. The corresponding equations are presented in Equation. 7.1, where $A$ and $B$ represent two complex number input data, $X$ and $Y$ represent two complex number output data, which would be sent back to data memory after butterfly computation and $W$ denotes the responding FFT coefficients. For the hardware design, four signed multiplications and six signed additions or subtractions are needed. Only 10 full-adder/subtracters are used in serial's implementation.

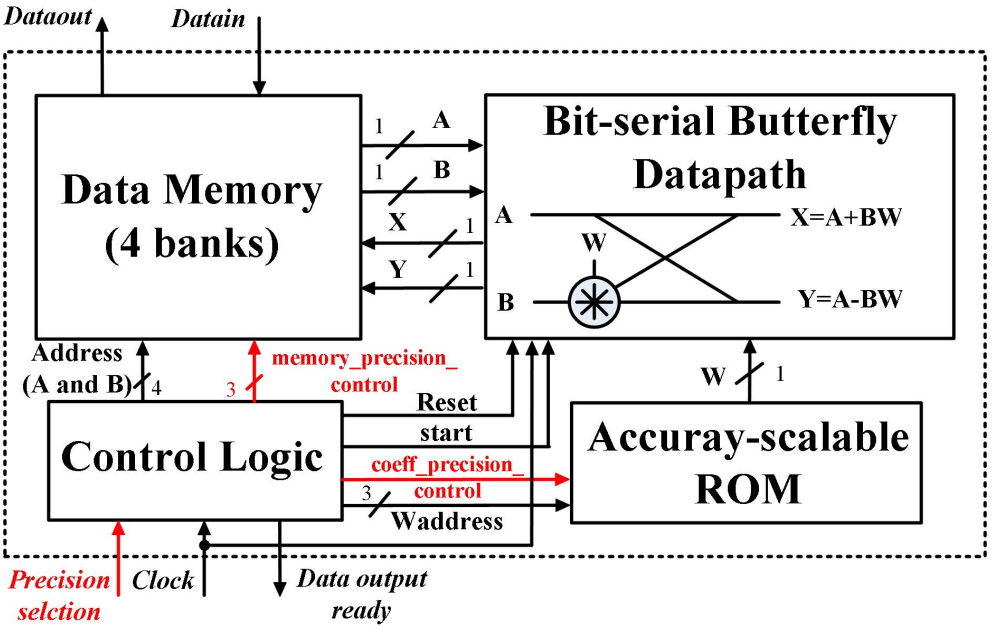
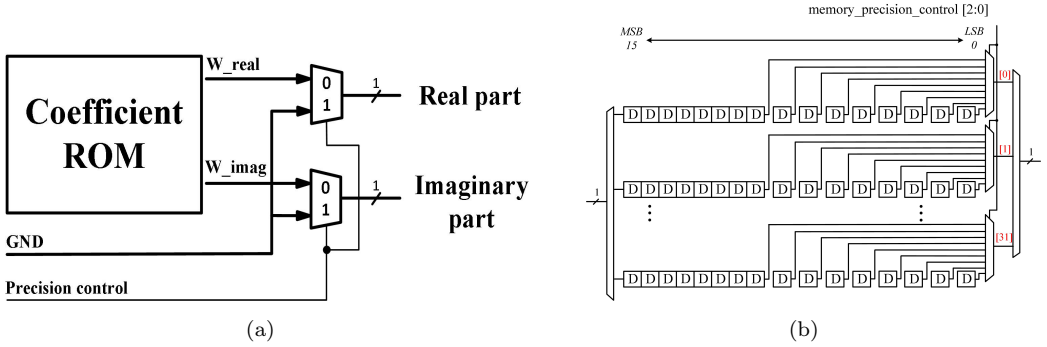



FIGURE 4.10: Structure of the proposed bit-serial FFT

$$X = A + B * W \quad and \quad Y = A - B * W \tag{4.4}$$

Illustrated in the previous section, the butterfly datapath using the proposed bit-serial multiplier, which can skip redundant clock cycles when the binary value of FFT coefficient is zero. In other words, the simple zeroing of leading or trailing bits of the coefficients, the clock cycle of butterfly operation can be decreased in proportion to the bit precision. As shown in Figure 4.11(a), the twiddle factor data output bit by bit from the ROM. The two MUXs can adjust the precision of twiddle factor value freely, one of the inputs is from the ROM output and the other is connected with ground, the two outputs would be sent to the butterfly processing block.




(a) (b)

FIGURE 4.11: Block diagram of memory (a) Accuracy-scalable ROM (b) Scalable data memory based on shift registers

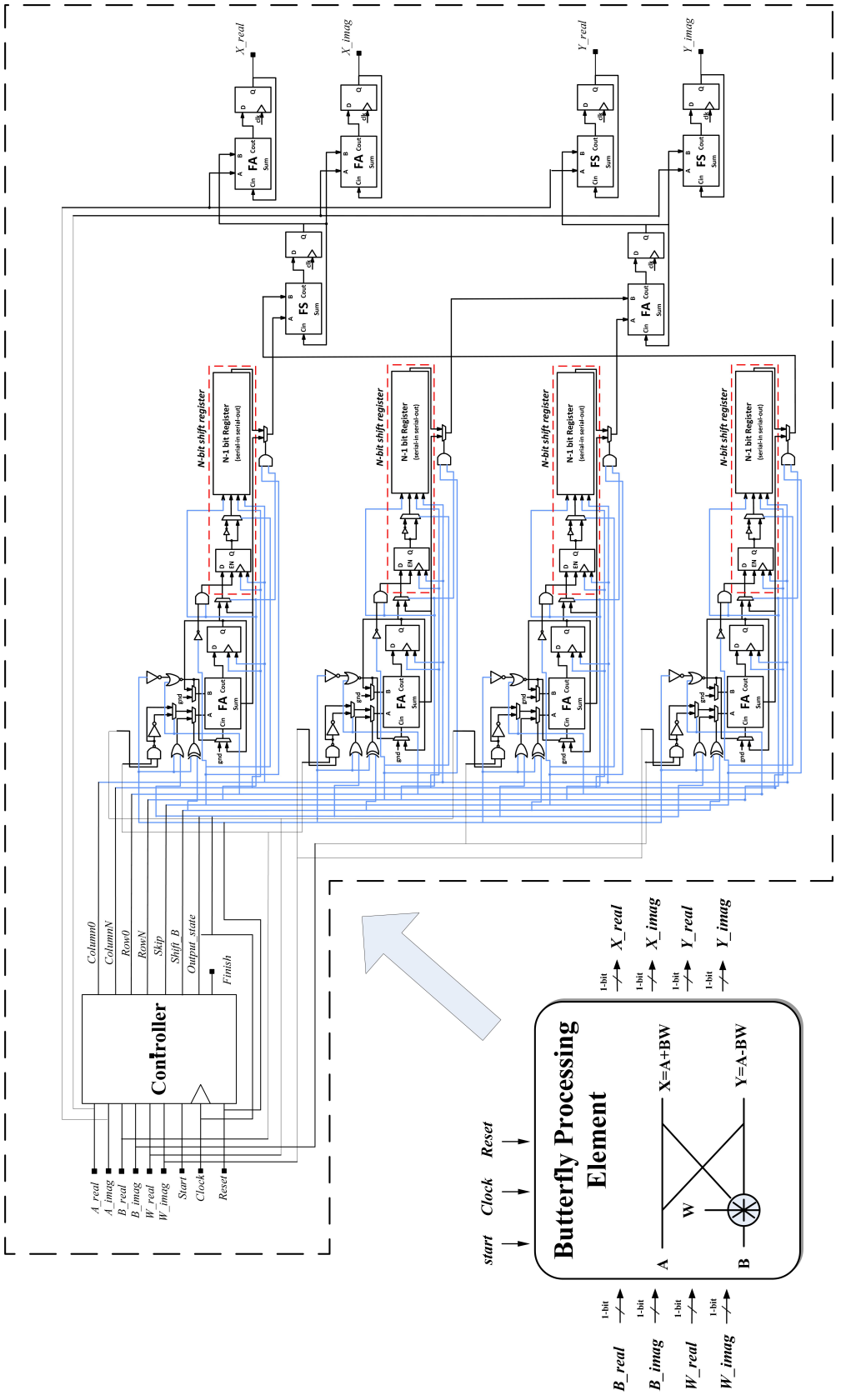



FIGURE 4.12: Block diagram of bit-serial butterfly processing element

Additionally, the word-length of input signals can be also reduced for further power-savings. As seen in Figure 4.11(b), the proposed data memory is designed based on shift registers, the accuracy of output data is dependent on the clock cycles for shifting. An 8:1 mux is inserted following each column of shift resisters to select the Least Significant Bit of the output data. Accordingly, the word-length of input signals can be easily adjusted by the signal *memory_precision_control* from control block.

Illustrated in the previous section, the butterfly datapath using the proposed bit-serial multiplier, which can skip redundant clock cycles when the binary value of the FFT coefficient is zero. In other words, the simple zeroing of leading or trailing bits of the coefficients, the clock cycle of butterfly operation can be decreased in proportion to the bit precision. Besides, the input complex numbers can also be scalable for 8-bit and 16-bit for further accelerated operation. Once 8-bit precision scaling is enabled, the data is selected from the $8^{th}$ register's output within a column of shift registers with the disability of the following 8 register. In this way, butterfly operating cycles can be further reduced by nearly half. Specifically, in the proposed hardware design, a 16-bit FFT computation requires 26110 clock cycles, while around 43% cycles can be saved when the effective word-length of coefficients is reduced to 8-bit. For an 8-bit operation, only 1000 cycles are consumed.

### 4.4.2 Simulation Results

In this section, the proposed design as well as a number of traditional FFT implementations used in our comparisons, both serial and parallel, are synthesised using the ST 65nm CMOS library and the power and energy consumption are measured using Hspice.

#### 4.4.2.1 Low area and low leakage

It is well known that the serial approach uses less area and has less interconnect capacitance, which results in smaller leakage power. The 64-point FFT processors based on conventional bit-serial and energy-aware parallel architecture [89] are synthesised in our comparisons [90][91], together with other state-of-the-art serial FFT implementations. From the Table. 6.2, it can be found that, the proposed serial design has significantly lower hardware cost than the parallel one. The total gate numbers in the 64-point FFT design is also smaller than designs based on other serial techniques, such as digit-serial [43] and distributed arithmetic based FFT implementations [91]. It is also worth noting that compared with the conventional bit-serial design, the proposed bypassing design increases the gate count by less than 10 %. In terms of leakage power, as the design operates at a near-threshold voltage (0.5 V), additional power saving is achieved. Leakage power consumption of the serial and parallel design is 1.83 $\mu w$ and 2.87 $\mu w$ respectively, shown in Table. 4.2.

TABLE 4.2: The comparisons between gate count and leakage power between proposed bit-serial, state-of-the-art bit-serial, conventional bit-serial and parallel FFT implementations.

| Architecture | Proposed | Parallel [89] | Conv serial [92] | DA [91] | Digit-serial [43] |
|---|---|---|---|---|---|
| Gate Count | 24.4K | 40.5K | 22K | 33.7K | 55K |
| Leakage Power | 1.83$\mu w$ | 2.87 $\mu w$ | 1.68$\mu w$ | 2.44$\mu w$ | N/A |

#### 4.4.2.2 Power and Energy Analysis

As mentioned above, serial designs have much smaller gate counts and interconnect capacitance compared with their parallel counterparts. This advantage allows serial operations to consume significantly less power at a fixed frequency. However, as parallel implementation operate faster and require less time to complete the same operation, it can be argued that a fairer power consumption comparison should be based on a scenario where operating frequencies of parallel designs are slowed down such that all types of designs calculate their results at the same speed.

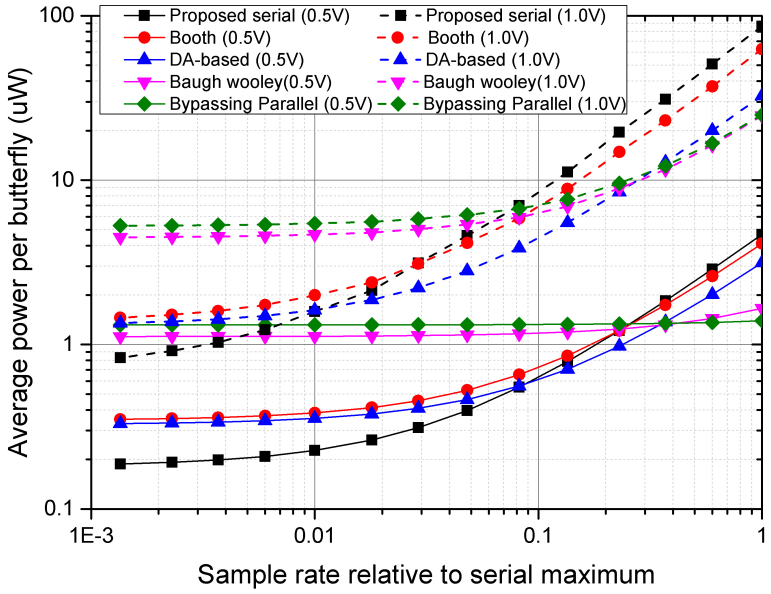



FIGURE 4.13: Average power consumption comparison in butterfly datapath with various multipliers vs sampling rate at super- and near-threshold regions (1 V and 0.5 V supply voltage).

In Figure 4.13, the proposed bit-serial multiplier butterfly processing unit is firstly demonstrated on the butterfly processing unit, compared with other state-of-the-art multipliers. The average power consumption of the proposed bit-serial and parallel counterparts are shown with the sample rate scaled to achieve the same computation time at both super and near-threshold voltage regions respectively. The sample rate on

the x-axis is normalized to the maximum sample rate that can be processed by the conventional bit serial design, where the maximum sample rate is 4.36 MHz and 116 KHz for 1.0 V and 0.5 V voltage operations respectively. It is worth noting that both the designs are clock-gated during idle intervals. Simulation results show that the proposed bit-serial design outperforms parallel counterparts at low sample rate.

The next scenario in our power consumption analysis tests the performance of the bit-serial design in FFT implementation. Both the serial and parallel FFT run at serial's maximum sample rate, 639 KHz and 1.2 KHz for 1.0 V and 0.5 V voltage operations respectively. Simulation results in Figure 4.14 shows that the power benefits of the proposed serial design at a given sample rate is greater at the near-threshold supply voltage. This is because the leakage power consumes a higher proportion of the overall power at low voltages. For both super and near-threshold operations, at a high sample rate, the dynamic power dominates the total power because of higher required clock frequencies. At low sample rates, it is trivial that the serial implementation's low leakage power consumption has a significant advantage. Specifically, the results show that the proposed bit-serial design uses up to 36% less power than the parallel counterpart. In general, it turns out that the power advantage of serial butterfly datapath remains in whole FFT computation.

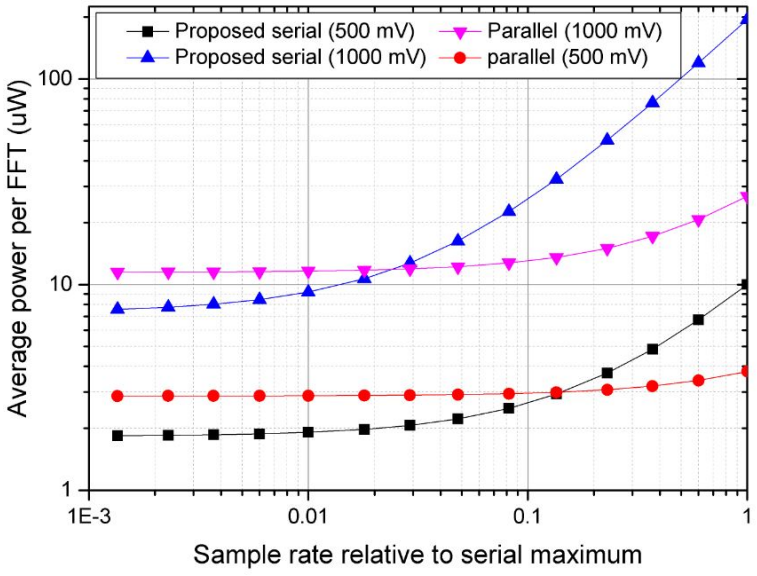



FIGURE 4.14: Average power consumption of both serial and parallel FFT implementations at 1.0 V and 0.5 V

Figure 4.15 displays the power comparison between the proposed serial and parallel implementations operating at 0.5 V supply voltage, with sample rate scaling, for 8-bit and 16-bit arithmetic. It can be observed that, at low sample rate, variable-accuracy is more efficient for proposed serial approach. For parallel implementation, although the accuracy reduction can significantly save the dynamic power in arithmetic unit. As leakage power still domain the total power consumption due to long operation time, power

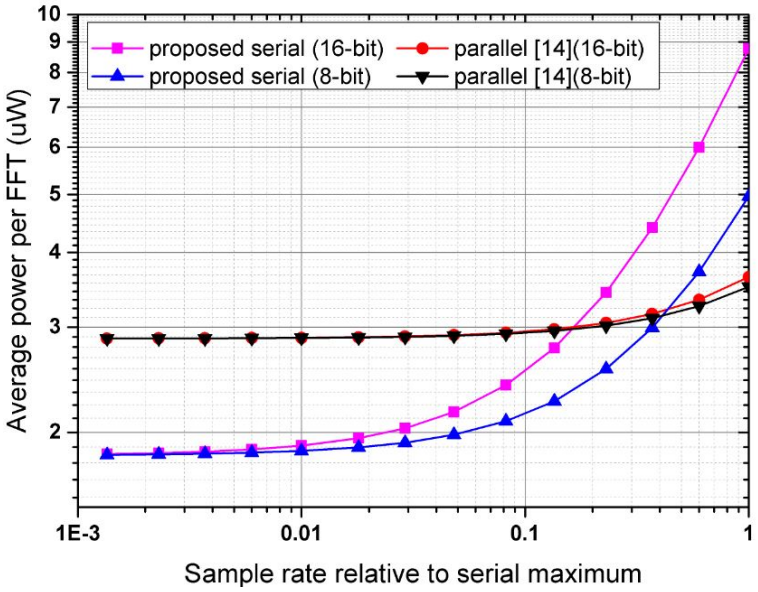



FIGURE 4.15: Average power consumption of both serial and parallel FFT implementations at 0.5 V with 8-bit and 16-bit bitwdith

saving by reduced accuracy computation in parallel FFT processor is negligible. On the other hand, serial implementation can save around 62% clock cycles. To maintain the same computation time, operating frequency of serial design is slowed down, accordingly the average power consumption can be reduced significantly.

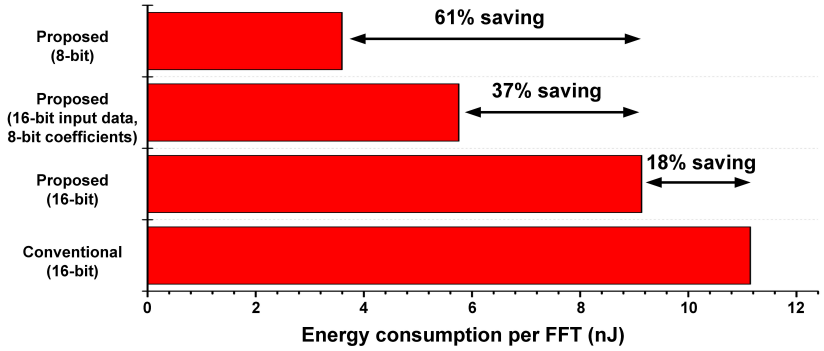



FIGURE 4.16: Energy reduction with bypassing technique and accuracy loss

In addition to the advantages of low-area and low-leakage, serial implementation lends itself easily to energy-scalable computing. Bit-serial computing can remain ultra-low power consumption during operation, while the clock cycles are reduced with the degraded computation accuracy, the energy consumption of FFT computation can also be saved. Figure **??** presents the energy comparisons between conventional and proposed bit-serial multiplier at the minimum energy point. Also, it demonstrates the variable accuracy behaviour of the proposed design. It can be seen that the proposed bit-serial FFT design can save around 18% energy through the bypassing technique in terms of 16-bit computation. For the variable-accuracy operation, the energy consumption changes

approximately linearly with the effective bit-width as expected. Approximately 37% energy can be saved when the effective word-length of the coefficient is reduced from 16 to 8 bits while the 8-bit computation can further scales down the energy dissipation by 61%.

## 4.5 Summary

We investigate the possibility of further power reductions where a near-threshold bit-serial operation is used in the signed multiplication. The main novelty of the design is the adoption of the row bypassing technique, hitherto only used in parallel multiplication, which can be implemented very simply in the bit-serial control state machine. Moreover, the row bypassing can also be used to implement variable bit precision with a minimal additional circuitry. Furthermore, to stay alive under intermittent power supply, inherent, for example, in energy-harvester powered sensor systems, the proposed design can dynamically scale down the energy consumption at the cost of computational accuracy. The simulation results demonstrated on a 64-point FFT implementation show that the energy consumption can be saved up to 61% when the word-length is shifted from 16-bit to 8-bit. These results prove that the near-threshold operation and bit-serial techniques are a promising solution to implement ultra-low power designs powered by energy harvesters, especially in IoT sensor applications where ultra-low power and energy consumption are usually of greater concerns than the computational performance.

# Chapter 5

# Timing-Error-Prevention for Bit-Serial Computing by Dynamic Path Isolation

Due to the area and power benefits bit-serial computing brings for IoT-relevant ultra-low-voltage digital designs (illustrated in the previous chapter), a new in-situ timing error prevention technique is proposed for mitigating the impact of process variation on bit-serial sequential circuits.

The proposed design places checkpoint flip-flops on each critical-path and dynamically split the clock cycle into two for error-resilience. Compared with prior error prevention for parallel design, the proposed method is able to deal with frequent suspicious timing error with a very low performance loss. In general, the approach can be both utilized in the conventional parallel and serial design.

## 5.1 Conventional Clock-gating Approach for Error Prevention

Combined with state-of-the-art half-path error prediction method [93][72], clock-gating techniques can dynamically isolate the critical path once an error is predicted. Compared to the prevention technique using voltage scaling, it is able to deal with errors caused by fast dynamic variations and prevent the errors that happen in the current clock cycle. In general, clock-gating error prevention method is easily applied in the conventional digital designs. Figure 5.1 shows the circuit schematic of the existing EPAP (Error Prediction And Prevention) system with a clock-gating technique where the critical path is divided into two path segments and the signal *error* indicating the detected potential timing errors that can be correctly generated before the rising edge of *clk*. A corresponding

timing diagram is displayed in Figure 5.2, it can be seen that suspicious errors can be successfully prevented with the isolation of critical paths.

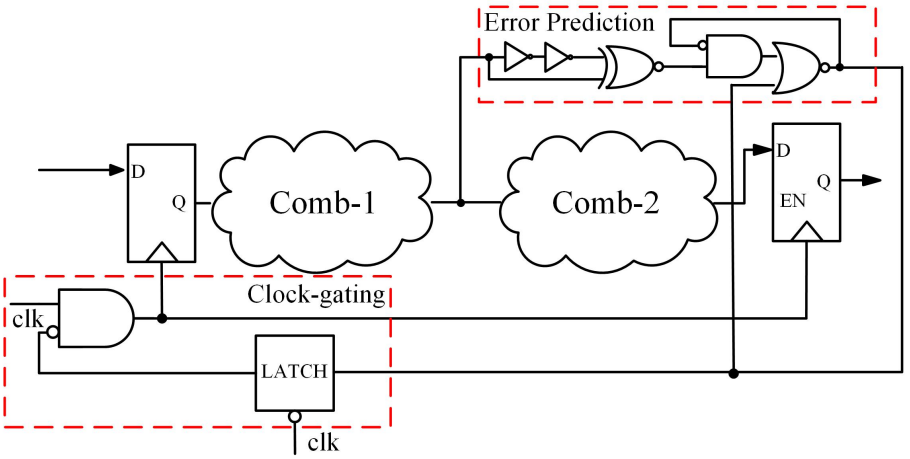



FIGURE 5.1: Circuit schematic of in-cycle clock gating

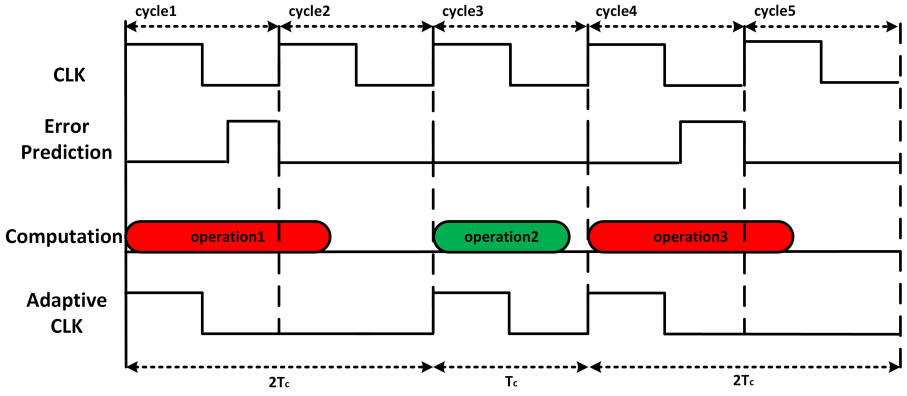



FIGURE 5.2: Timing diagram of critical path isolation paradigm

Basically, it can be concluded that clock-gating is a more efficient error prevention method compared with voltage/frequency scaling for conventional parallel designs. However, the following potential problems may constrain the use of the clock gating technique in bit-serial computing. Initially, it poses a challenge on driving ability of the clock gating circuits, delay induced by clock gating may also lead to synchronization problems. In order to avoid this problem, the driving ability has to be improved at the cost of area. In this case, the area overhead of the clock-gating technique would be a heavy burden for bit-serial sequential, which is well-known for low area cost. Additionally, for clock gating technique, the performance penalty can be huge if suspicious timing errors are frequently detected. This situation would be more serious for bit-serial computing, which consumes much more clock cycles compared with conventional parallel.

## 5.2 Critical Path Isolation

The proposed timing error prevention method for bit-serial sequential circuits is described in Figure 5.3, where the error prediction circuits are designed based on Shi's previous work [73]. It should be noted that the half-path-point is selected as the location for detecting suspicious timing errors, any transition after the falling edge of the clock signal can be identified as a potential timing error. The half-path point can be determined from the timing report generated by timing analysis tools.

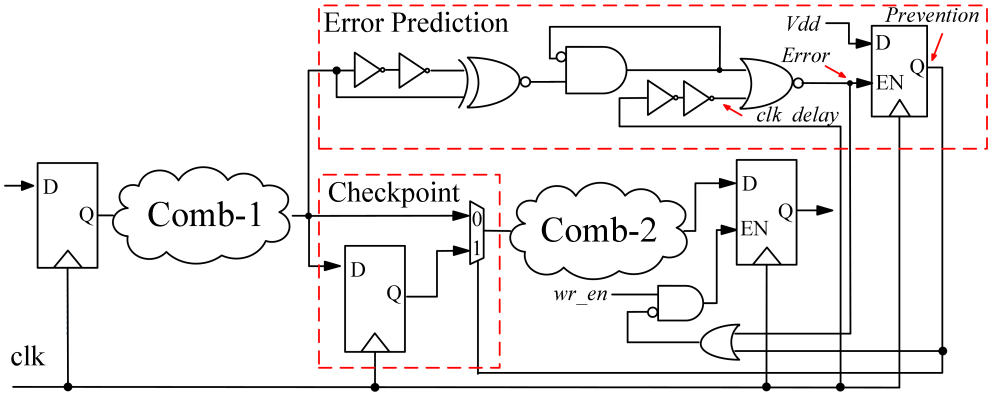



FIGURE 5.3: Circuit schematic of the proposed checkpoint mechanism

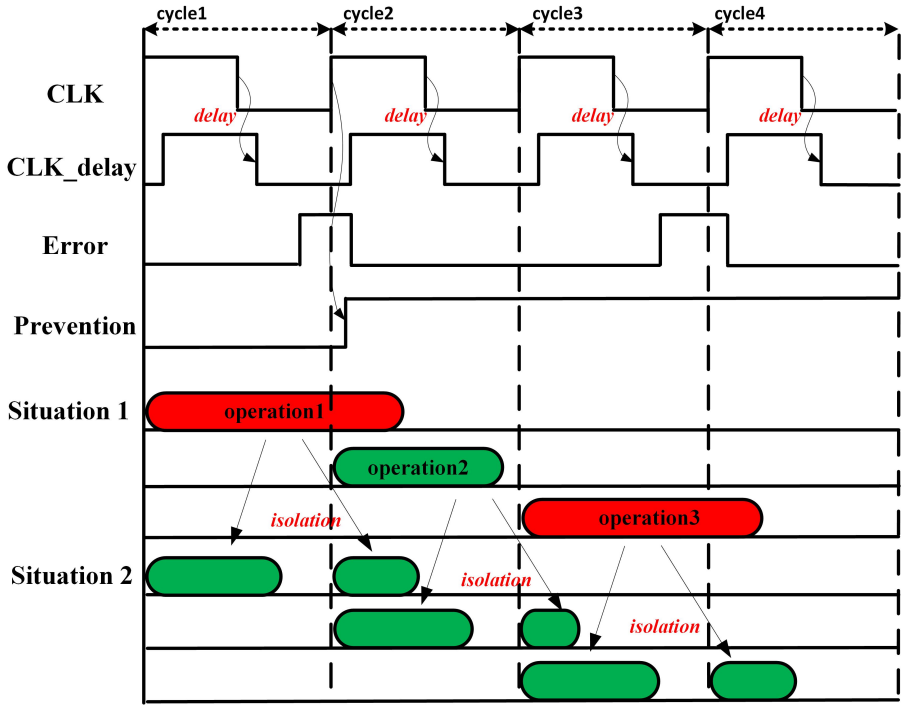



FIGURE 5.4: Timing diagram of the proposed critical path isolation paradigm with pipeline operation

As shown in Figure 5.3, the circuit can be divided into two parts: half-path prediction and error prevention. An XOR gate is used to detect the late transition in half-path point and generate a high-level signal if a timing error is predicted. The following

inverter and AOI gate can keep the generated signal unchanged during the low clock phase. In this design, the global clock signal is delayed and fed into an OR gate, it can not only cover the mismatch between half-path point and corresponding gate, but also guarantee the transition of generated *error* signal be later than the falling edge of the clock signal and allow a high-level signal to store into the following D-flipflop. Accordingly, the signal *Prevention*, the output of the D-flipflop, would be kept high until the D-flipflop resets. In check-point node, data is stored into a flipflop. It should be noted that, in the circuit schematic above, checkpoint node is not chosen in the same location as a half-path point. The checkpoint node can be located anywhere in the longest path just for feasibility and cost-efficiency. A MUX is inserted connecting the check-point node and the flipflop's output as two input signals, the selection is determined by signal *Prevention*. If *Prevention* remains low, that means no timing error is predicted, the operation would be finished in one cycle. Otherwise, checkpoint data is fed into the gates followed by the MUX in the next clock cycle, a re-execution is implemented while the predict incorrect result in the last cycle does not store into the end-point register. As shown in Figure 5.4, once suspicious timing error is detected, *error* would keep high during the low phase of the delayed clock signal. The original single-cycle operation is split into two and implemented in a pipelined way. If the timing error is detected frequently, only one clock cycle is consumed at the cost, therefore, the system performance would not be degraded seriously.

## 5.3 Utilization of the Path Isolation in the Conventional Parallel Design

NBTI (Negative Bias Temperature Instability is becoming one of the major circuit reliability issues in nano-scale technologies. BTI can cause a threshold voltage shift in CMOS devices and consequently increase circuit delay. A novel ageing aware approach to improve circuit's lifetime is proposed and demonstrated on the conventional parallel design. The vulnerable circuit paths against ageing effects are isolated. In addition, minimum area overhead is consumed by adopting proposed synthesis algorithm.

### 5.3.1 Ageing-aware approach via path isolation

In this section, a novel approach is proposed to efficiently increase circuit robustness against ageing with negligible area and delay overheads. The basic idea is to isolate potential timing-violated paths with the presence of ageing into two cycles, thereby providing more guardband to mitigate ageing-issued timing violations. Depend on different design requirements about ageing, the circuit guradband can be adjusted ranging from 10% to 25%. Compared with standard two-cycle pipeline operation, the proposed

approach inserts D-flipflops as checkpoints into fewer number of circuit paths while acceptable timing margin is acquired to prevent ageing-issued problems.

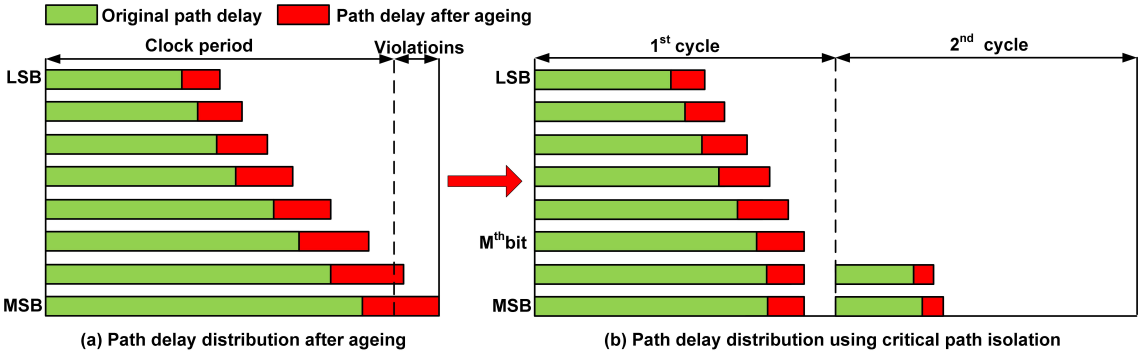



FIGURE 5.5: Design time and post-ageing data arrival times before and after ageing-aware optimization for an 8-bit adder

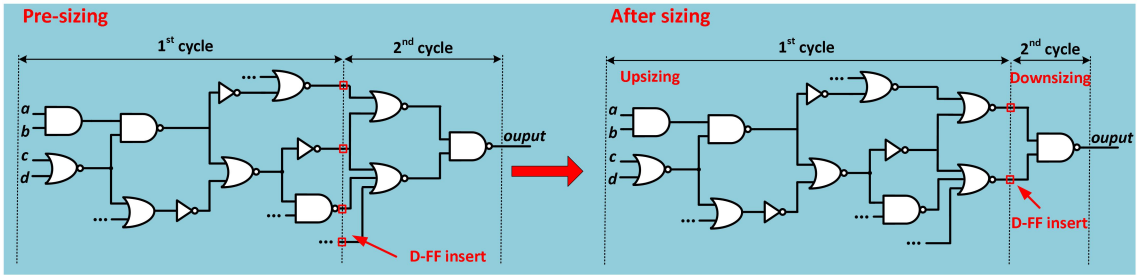



FIGURE 5.6: An example of the proposed synthesis approach

For the sake of simplicity, we choose an 8-bit ripple carry adder as an example, the path delay distribution across sum bits is showed in Figure 5.5. The green bars represent the original path delay without ageing effects and the red bars denotes the increased circuit delay due to circuit ageing, respectively. It can be clearly seen that the path delay of MSB violates timing constraints with the presence of ageing firstly. Once the MSB paths start to fail, it would lead to a rapid decline in the computation accuracy. With the adoption of the proposed approach, this concern can be significantly eliminated. For a certain amount of guardband these violated paths would be separated into two part to extend the circuit lifetime. In the other words, the previous one-cycle operation is modified as two-cycle pipeline computation. In terms of performance loss, only one extra clock cycle is required without any effects on system sampling or throughput rate.

#### 5.3.1.1 Algorithm for Path Isolation

In the proposed approach, our main objective is to assign ageing-aware timing constraints to the circuit via the path isolation method during the synthesis stage to extend the circuit lifetime according to a certain amount of guardband. In order to realize the path isolations, a large number of flip-flops are inserted to cover all potential circuit paths. Take the circuit's cost-efficiency into consideration, we propose a synthesis method via

gate sizing to reduce the number of inserted flip-flops while maintains the same path coverage. Specifically, for the circuit paths with the same end-point, the number of their shared path nodes grows with the increase of circuit depth while that of the total path node is reduced. Therefore, once a certain amount of guardband is given, we can move the path isolation nodes forward, to the deeper circuit paths by upsizing and downsizing the different part of gates. Accordingly, the required insertion nodes can be significantly reduced, while the total area overhead is very small.

As seen in Figure 5.7, we propose Algorithm 1 to isolate some paths and put in a specific timing margin, compensating for the delay shift caused by the BTI effect. Specifically, the required intrinsic circuit delay $D_{t0}$ is first computed based on the maximum delay allowed by the system and the percentage timing margin. In each iteration of the optimization, we focus on the most critical path, which has the largest delay and thus requires the greatest delay reduction. To isolate the path, the gates of the path are divided into a lower-depth group ($G_l$) and a greater-depth group ($G_g$) based on the signal arrival time on their output nodes: a gate with the output signal arrival time smaller/larger than $D_{t0}$ is considered to be at the lower-/greater-depth and is categorized into $G_l/G_g$. One flip-flop (FF) is then inserted in the path, to isolate the gates of $G_l$ and $G_g$. In this way, the most critical path is broken down into two paths, both having the path delay smaller than $D_{t0}$. In order to ensure the correct function, other paths need also be changed to support a two-cycle operation: flip-flops are inserted at the specific internal nodes of all paths that end at the same output as the most critical one, while one FF is required to be put in the each of the rest outputs, delaying the output signals for one clock cycle. We then use an algorithm to reduce the area overhead, as will be explained later. The above steps are repeated until the intrinsic circuit delay is not greater than $D_{t0}$, indicating that all paths have the required timing margin.

As has been described, the number of the required flip-flops may be reduced, by moving the flip-flops to a greater circuit depth. This can be realized by resizing the gates of a signal path. In specific, the flip-flops are inserted at the internal node of the critical path, where the signal arrival time is just smaller than $D_{t0}$, according to Algorithm 1. Therefore, by up-sizing some logic gates, the signal arrival times at the specific internal nodes may become smaller, moving the two-cycle operation point to a greater circuit depth. While the up-sizing process introduces more area on the combinational logic, it also potentially reduces the number of flip-flops required to isolate the paths, reducing the area of the sequential part (*i.e.* the flip-flops). Therefore, the overall circuit area needs to be evaluated to ensure a smaller area cost. We present Algorithm 2 in Figure 5.8 to minimize the area cost. This algorithm iteratively tightens the timing constraint to up-size the gates of the critical path. Only the gates of $G_l$ can be up-sized. This is because up-sizing the gates of $G_g$ would not change the signal arrival time of any gates of $G_l$ and thus, the flip-flops would be inserted at the same nodes as before the circuit is up-sized, causing more area cost. The flip-flops are re-inserted on the up-sized circuit

**Algorithm 1** Path isolations for lifetime extension

**Require:** all paths have required timing margin;

```
1:  procedure PATHISO()
2:      D_t0 = max delay/(1+margin%);
3:      critPath = path with the largest delay;
4:      while D_critPath > D_t0 do
5:          for each gate ∈ critPath do
6:              if arrival time < D_t0 then
7:                  G_l = G_l ∪ gate;
8:              else
9:                  G_g = G_g ∪ gate;
10:         Insert FFs based on G_l and G_g;
11:         AREAMIN();
12:         Identify new critPath;
13:     return optCircuit
```

FIGURE 5.7: Algorithm of the path isolations for lifetime extension

according to the signal arrival time, similar to Algorithm 1. We compute the cost based on the area increase of the combinational logic and the area decrease of the sequential part. The optimization ends when the area cost becomes larger than 0, indicating the current design has the minimum area and any further change would induce more area cost.

**Algorithm 2** Area cost minimization

```
1:  procedure AREAMIN()
2:      Timing constraint = D_critPath;
3:      repeat
4:          Reduce timing constraint;
5:          upSize(G_l) to meet timing constraint;
6:          Re-insert FFs;
7:          Cost = AreaInc(comb) - AreaDec(seq);
8:      until Cost > 0;
```

FIGURE 5.8: Algorithm of minimizing the area cost

Figure 5.6 shows a simple circuit to demonstrate the difference between the selections of D-flipflop insertion nodes before and after gate sizing. It can be seen clearly, before

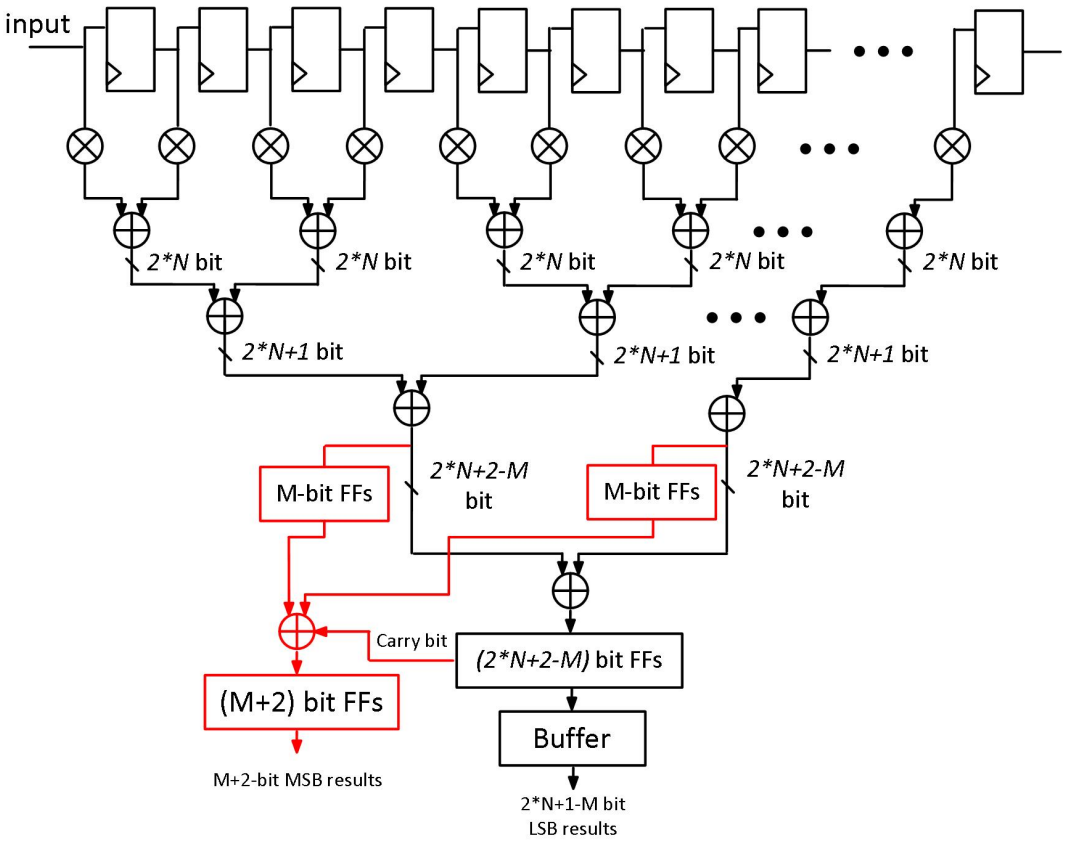



FIGURE 5.9: An example of FIR datapath with 15% guardband

circuit optimization, there are 4 two cycle D-flipflop insertion nodes. After carefully gate sizing, the number of inserted register are reduced by double while the guradband margin is maintained. In general, for the proposed ageing-aware synthesis approach, these constraints provide almost the same result as offered by conventional techniques, however, with much less D-flipflop insertion nodes.

#### 5.3.1.2 Case study of an FIR filter

To investigate the proposed ageing aware approach, we implemented a 16-tap digital filer datapath, which is an important application in hardware accelerations. With respect to the different amount of guardband against ageing $(5\% - 25\%)$, the ageing-aware FIR circuit is modified respectively and the relevant simulation results are analysed in this section.

#### 5.3.1.3 VLSI implementation

For the sake of simplicity, we choose an FIR architecture with a 15% gurdband as an example. Figure 5.9 shows a modified datapath of a 16-tap FIR filter with $N$-bit computation accuracy, where the red highlighted parts represent the auxiliary circuits

used to realize path isolation. The optimal isolation nodes are found around the third adder stags, the particular amount of 15% guardband can be acquired by carefully gate sizing.

In this design, $2 \times M$ flip-flops are inserted to store the intermediate results, where the value of $M$ is defined by the total number of timing violated paths affected by ageing. Since FIR filtering computation is based on multiplication and addition and the critical paths are in the MSB of results and the path delay distribution for each individual end-point of all paths shows a degrade trend across result bits (from MSB to LSB groups). In terms of an n-bit FIR filtering operation, fewer circuit paths would be seriously affected by ageing and lead to timing errors. If we assume all the circuit paths have 15% delay shift resulted from ageing problem. The results show that around $M+2$-bit result would suffer from timing errors.

As seen in Figure 5.9, 2*N+4-bit results are generated in a N-bit FIR filter operation to avoid overflows. The whole computation is implemented in a pipeline way. The 2*N+2-bit LSB group results are initially computed in the first clock cycle while some immediate results of MSB computation are stored to prevent timing errors. Until the next clock cycle, the complete results could be generated. It should be mention is that during the second clock cycle, the MSB in 2*N+2-bit LSB group results would be sent to realize uncompleted operation as a carry bit.

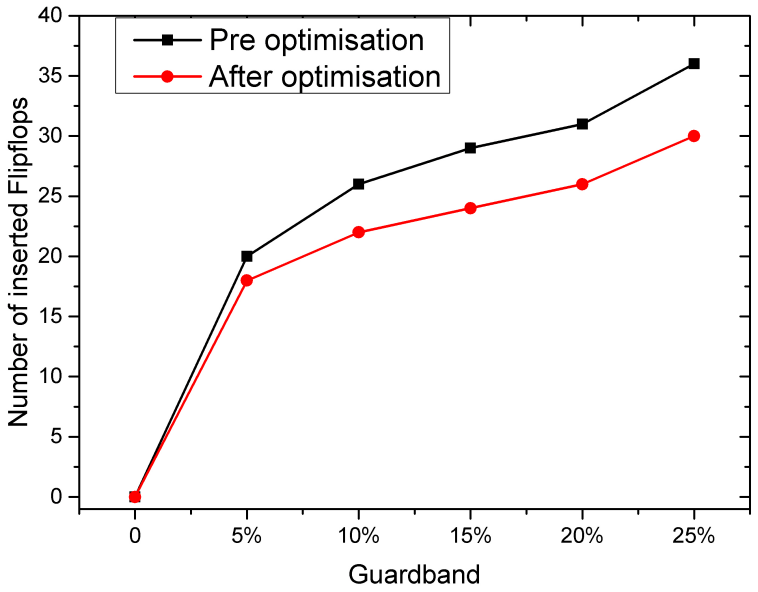



FIGURE 5.10: Number of inserted flipflops with the increase of guardband for both before and after the proposed synthesis approach

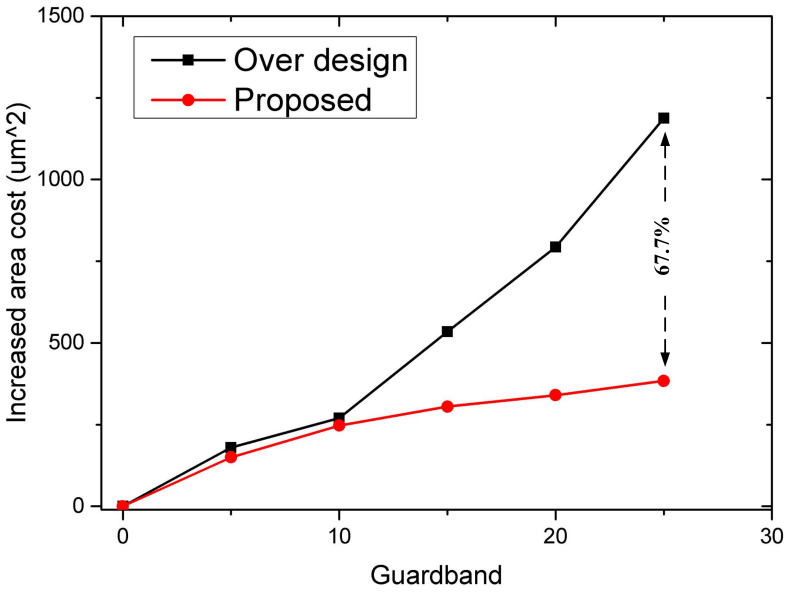



FIGURE 5.11: Increased area cost with the increase of guardband for both proposed and over designs

#### 5.3.1.4 Experimental Results

The proposed approach has been validated by applying it to a 16-tap digital FIR filter. We used Synopsys Design Compiler for logic-level optimization and our ageing-aware approach can provide guardband against ageing ranging from 5% to 25%.

Figure 5.10 shows the benefits of our proposed synthesis approach. As can be seen, the number of inserted flip-flops is significantly saved after circuit optimisation. In addition, these savings are kept rising with the increase of provided guardband. Compared with the conventional method, the number of flip-flops can be reduced by 17%.

As mentioned, practical approaches to improve circuit lifetime at the design stage is to leave enough timing margins against ageing effects, which increases the overheads of area and power. Thus, the circuit is over-designed. Here, we compare our proposed design with the ones are over-designed for the same amount of gurandband in terms of area overhead. The simulation results in Figure 5.11 show that, with the increase of the provided timing margin, our proposed design is more advantageous in area saving. As can be noted, our approach can save 35.3% area on average to guarantee the same lifetime reliability. When the timing guardband increase to 25%, the area saving can reach to 67.7%.

## 5.4 Utilization of the Path Isolation in the Serial Design

Based on the path isolation method, a new in-situ timing error prevention technique for mitigating the impact of process variation in bit-serial sequential circuits is described in this section. A case study of a bit-serial FFT processors shows that error prevention can be achieved with negligible area overhead and performance loss.

### 5.4.1 Error-resilient bit-serial FFT processor

Several publications mentioned in the literature point out that bit-serial design outperforms parallel counterpart in terms of area and energy-efficiency when performance is not the priority [57][94][95][96]. Timing error prevention is readily applied to DSP accelerator as they are normally datapath-dominated. To investigate the circuit-level error prevention approach, we implemented a bit-serial Fast Fourier Transform(FFT) processor, which is a common choice for hardware acceleration in Wireless and Multimedia SoCs.

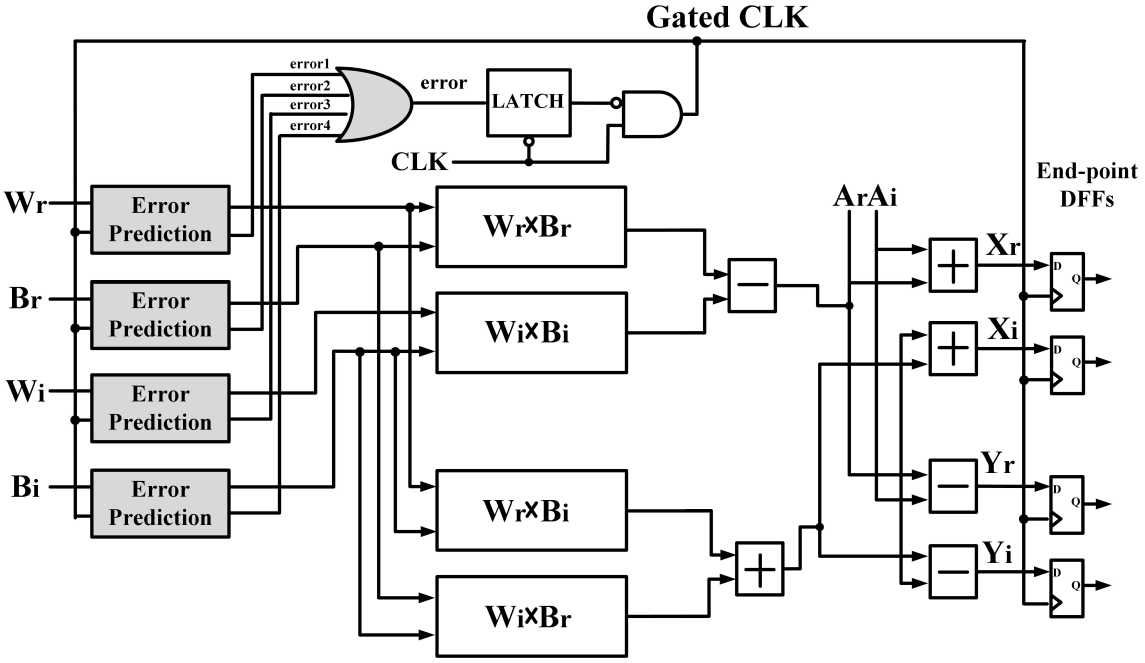



FIGURE 5.12: Circuit schematic of bit-serial butterfly datapath with clock gating based timing error prevention technique

#### 5.4.1.1 Clock-gating based design

In this paper, a 64-point memory-based FFT processor is adopted, which relies on 192 butterfly operations to realize FFT function. According to our previous publication results, a $n$-bit butterfly computation would cost $(n+1)^2$ clock cycle, accordingly, for a

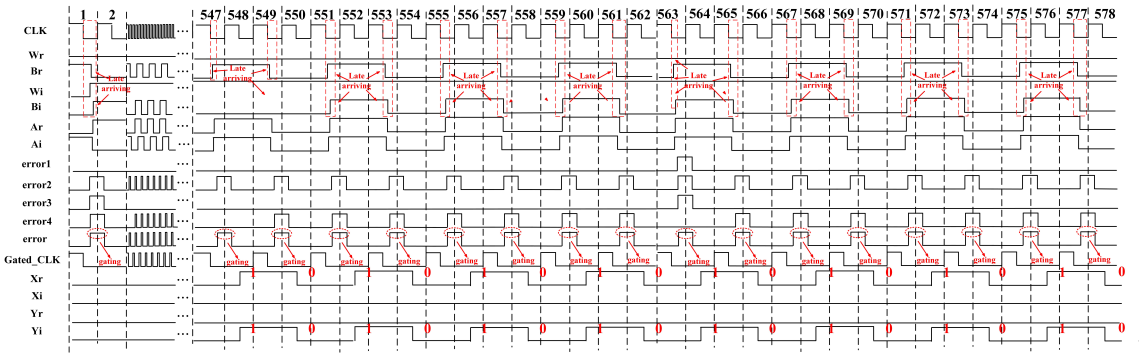



FIGURE 5.13: Timing diagram of the implementation with clock gating based timing error prevention technique

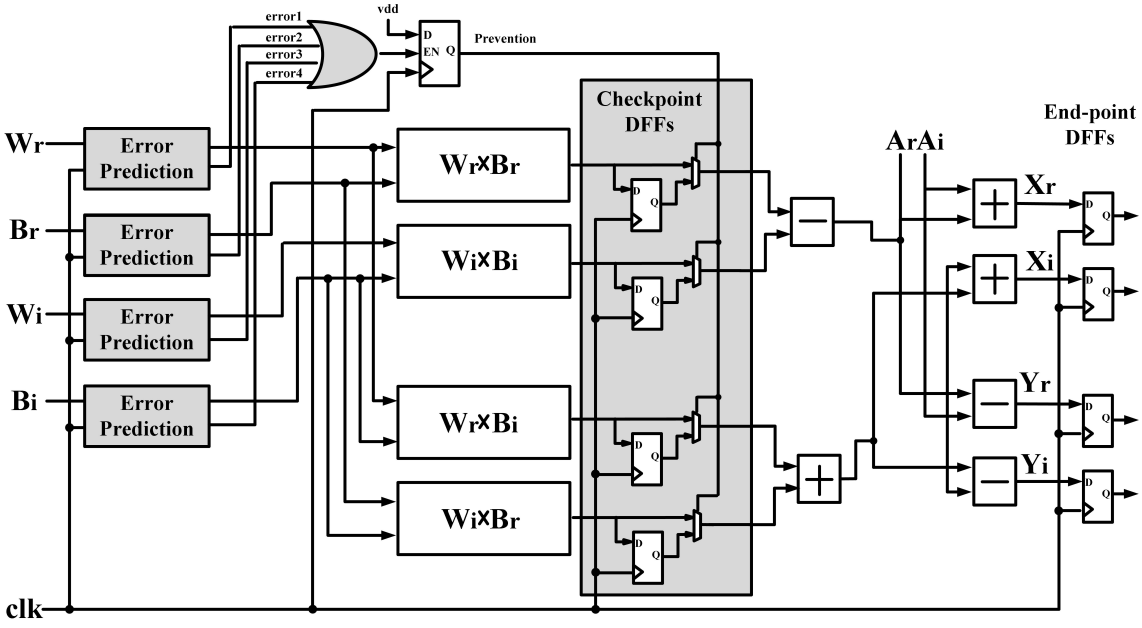



FIGURE 5.14: Circuit schematic of bit-serial butterfly datapath with the proposed timing error prevention technique

16-bit implementation, 279 cycles are required. The Figure 5.12 shows the schematic of bit-serial butterfly datapath with conventional clock gating technique. According to the measurement results from timing analysis tools, half-path points exist near the outputs of data memory, which are selected as the locations for detecting suspicious timing errors. Since all potential critical paths must pass the arithmetic gates to end-point flipflops, the selected nodes connecting bit-serial multipliers' inputs can cover all potential critical paths for detection. Considering that majority of operation time is consumed in partial product generations of the multiplier, critical-path is activated only when the multiplication results are generated for the following addition/subtraction operations. Therefore, checkpoint nodes are selected in the outputs of bit-serial multipliers, the guard-band for timing error prevention can be maximised in this case.

The corresponding timing diagram for one butterfly operation can be seen in Figure 5.13. In order to take the worst case into consideration, It is assumed that the control logic of

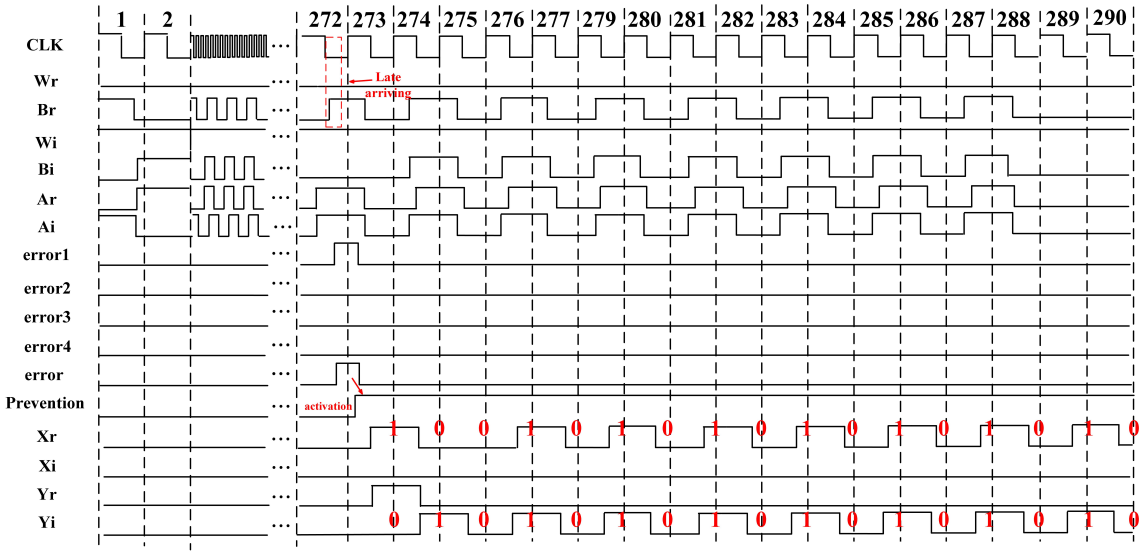



FIGURE 5.15: Timing diagram of the implementation with the proposed timing error prevention technique with only one error being predicted

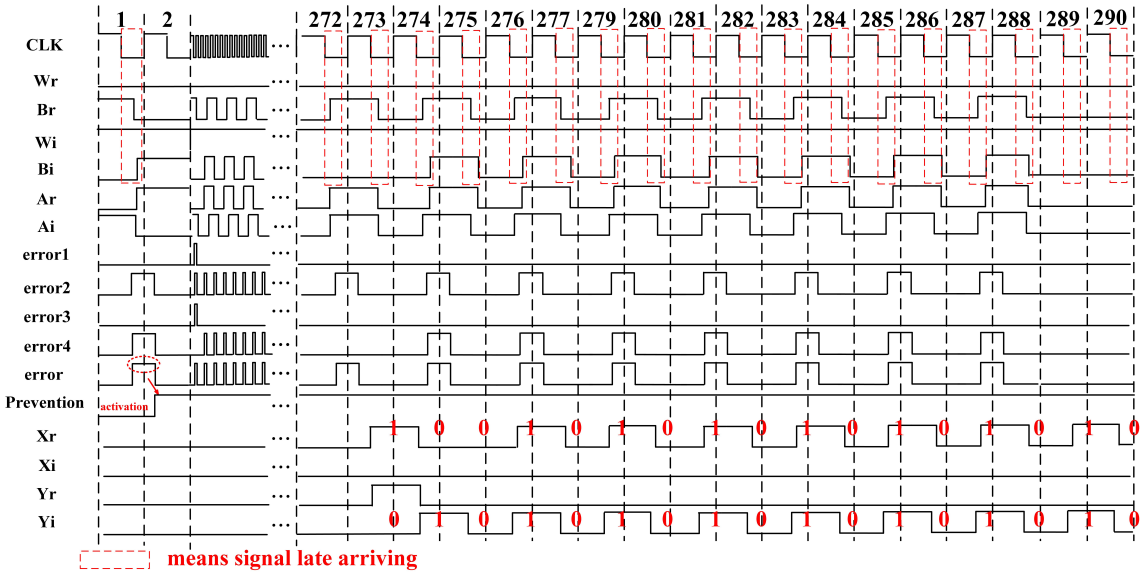



FIGURE 5.16: Timing diagram of the implementation with the proposed timing error prevention technique with massive errors being predicted

FFT design suffer from serious CMOS process variations or ageing effects, the input signals in data memory are fetched lately. Accordingly, massive timing errors are predicted. In this simulation, the input signals $A$ and $B$ are "21845+21845i" and "21845+21844i". The corresponding twiddle coefficient is "0-i". The computation results should be "43689" and "1+43690i", and the binary value of real and imaginary parts, $X_r$, $X_r$, $Y_r$ and $Y_i$ should be "0101010101010100", "0000000000000000", "0000000000000001" and "0101010101010101" respectively. The simulation result shows that signal *error* is activated in each clock cycle, accordingly the clock signal is gated frequently to prevent the suspicious timing errors and each bit of the correct results is generated after two clock cycles. A bit-serial butterfly computation is required 279 cycles as mentioned.

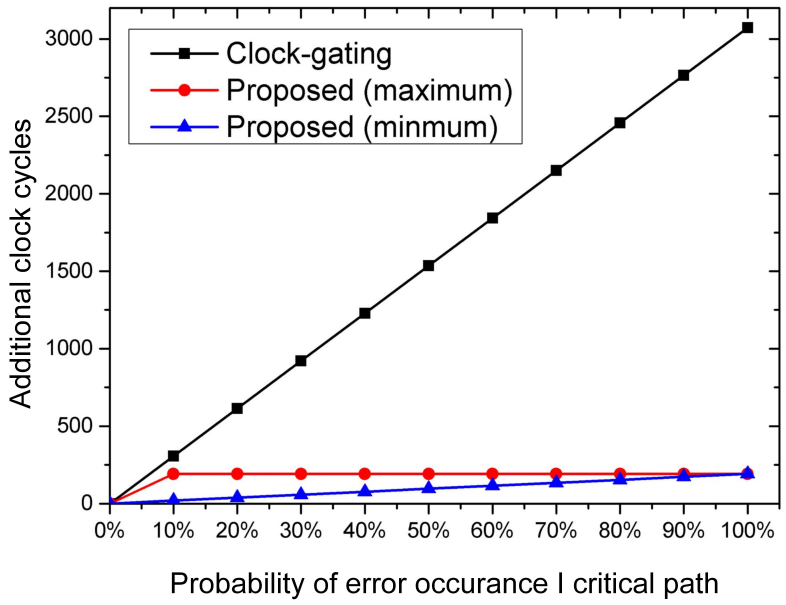



FIGURE 5.17: Clock cycle count versus error rate in the critical path for two methods

With the clock-gating prevention method, the total lock cycles used are increased to 578, which is twice compared with original value.

#### 5.4.1.2 Proposed critical path isolated design

A simulated timing diagram for the proposed approach is presented in Figure 5.15. As an example, it can explain how the proposed approach prevents predicted timing errors in a bit-serial implementation. It can be observed that, the only suspicious timing error is predicted in the $272^{th}$ clock cycle, the signal *Prevention* are changed from low to high with the rising edge of the clock, which represents the activation of critical-path isolation. Then, the MUXs inserted in the checkpoint node would output stored data in the flipflop rather than the multiplier outputs. It should be mentioned that the total clock cycle with the proposed approach consumes only 290 cycles, which only consume one more cycle compared with the original value. Figure 5.16 shows the timing diagram for the proposed approach with massive timing errors being predicted. From the first cycle, probable timing errors has already been detected while the two-cycle operation is implemented. The total clock cycle consumed is still 290, which present the most advantages of our approach, no matter how many errors are predicted, only one extra clock cycle is required. In conclusion, if the timing errors occur frequently, the circuit performance would be degraded significantly with clock-gating prevention method while it may not be a large concern using the proposed prevention technique.

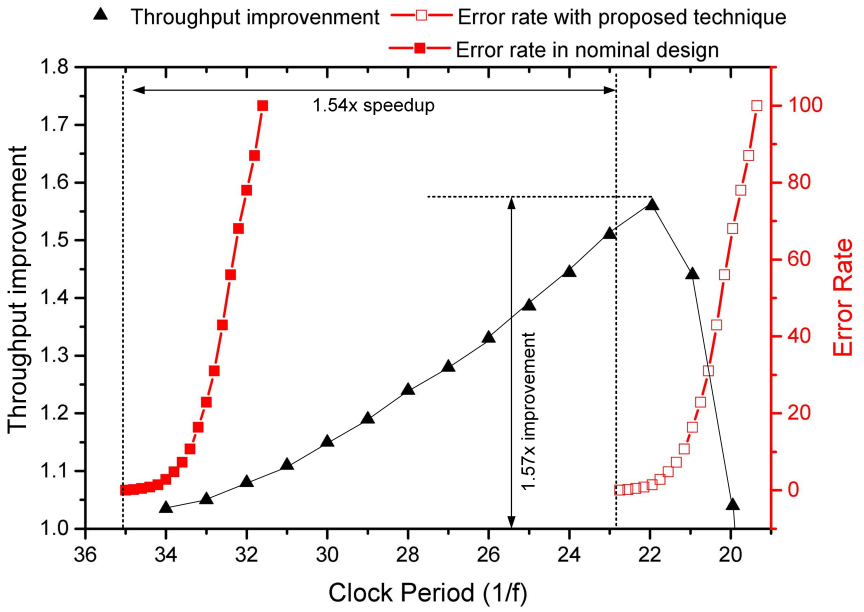



FIGURE 5.18: Throughput and error rate versus frequency characteristics

#### 5.4.1.3 Simulation results and comparison to published approaches

The comparison between the clock-gating and proposed method in terms of increased cycles with error rates occurs in critical-path can be seen in Figure 5.17. It can be observed that the number of increased clock cycles increase linearly with error rate by clock-gating methods. For the proposed technique, once timing error has been predicted in the current butterfly computation, the critical path was isolated and no more clock cycle is needed to maintain error-free operation. In the Figure, red and blue lines represent the change of cycle numbers in terms of worst and best case, respectively. Suffering from serious ageing-induced problems, the circuit operation may be significantly impacted while error rate in the critical path may reach up to 100%. In this worst case, 192 extra clock cycles are required to prevent predicted timing errors using proposed technique, which is in accordance with the number of total butterfly operations. In contrast, the clock-gating method consumes 16x clock cycles to implement error-free computation.

Since the proposed method can dynamically isolate the critical path, reducing its path length, in other words, it provides circuit an extra "guard-band" to mitigate the effect of process variations, there is an increase in the minimum clock period. Figure 5.18 details the timing error-rate vs. operation frequency in the FFT processor after Monte Carlo simulations with varied parameters at 0.6 V supply voltage and 25 C temperature. The experimental results show that in the nominal design, the *First failure point* appears around 35 ns while it could have a 12.3 ns delay with the proposed technique. In other words, the circuit can be overclocked by 1.54X without any errors. With

the frequency scaling, more timing errors are predicted, therefore, more recovery clock cycles are required. As mentioned, if the variation-induced delay is not large enough and only affect critical paths, the proposed method can consume negligible number of clock cycle for error-prevention. Hence, the hardware throughput presents an increased trend. However, when operating frequency scales down to a threshold value, the sub-critical paths are gradually affected, the performance improvement due to over-clocking may be cancelled. As shown in the figure, throughput improvement rapidly decreases from 23.1 ns. In general, maximum throughput improvement can achieve up to 1.57X.

TABLE 5.1: The comparisons of gate count and leakage power between proposed bit-serial, conventional parallel and stochastic designs.

| | Intel [97] | [98] | iRazor [99] | HEPP | Proposed |
|---|---|---|---|---|---|
| Process | 45nm | 65nm | 40nm | 180nm | 65nm |
| Type | EDAC | EDAC | EDAC | EPAP | EPAP |
| Performance penalty | Low | Medium | Low | High | Low |
| Area overhead | 6.9% | 8.3% | 11.9% | 7.8% | 0.5% |

## 5.5 Summary

Initially, a novel ageing-aware approach via path isolation is proposed. Unlike the state-of-the-art techniques, the additional timing margin does not only rely on gate sizing. We isolate these paths vulnerable to ageing effects by flip-flops insertion. Through carefully gate-level optimisation, we can reduce the number of inserted flip-flops while maintaining the same amount of guardband. The simulation results based on the FIR filter design shows that our approach is much cost-efficient The area cost of our approach is at most 67.7% less compared with a conventional over-design technique.

A cost-efficient timing error prevention method for bit-serial sequential circuits is presented in the section. Unlike previous works, clock-gating or voltage/frequency techniques are not adopted. We place checkpoint flip-flops in each critical-path and isolate the paths once the suspicious error is detected. Experimental results show that our design has very low performance penalty compared with the clock-gating technique. In addition, the proposed technique is applied to a bit-serial FFT processor, the circuit can be over-clocked by up to 1.54X and achieve 1.57X throughput improvement maximumly with a small area overhead of 0.5%.

# Chapter 6

# Timing-Error-Tolerant Variable Latency Design for Serial Computing

In this chapter, a novel approach is proposed to mitigate timing errors at the circuit-level for serial computing. This is achieved by modifying the path delay distribution for different bit results. For instance, the path delay for LSB (Least Significant Bit) bit computation would be designed as the longest one while the MSB (Most Significant Bit) path should be the shortest. Accordingly, timing violations would weakly result in large computation accuracy loss. This approach is demonstrated on an FIR filter implementation in 65 nm CMOS.

## 6.1 Delay Distribution in the Serial Datapth

The structure of arithmetic circuits usually reflects relatively deterministic path delay characteristics. Take an example of the conventional Baugh-Wooley multiplier, the critical path for an $N$-bit multiplier consists of $2 * N - 1$ full adder stage. The path delay difference between each product bit equals to one adder delay. The path delay distribution of a 8-bit multiplier can be seen in Figure 6.1. In parallel multiplier, critical paths usually occur in the MSBs, accordingly, MSBs would start to fail once the timing violations happen.

As mentioned in the literature review, in a serial structures, each result bit is generated from the same arithmetic datapath, MSBs and LSBs paths have very close latencies. For instance, a msb-first serial multiplier is considered, which is realized based on the shift-add operations. In an $N$-bit MSB first computation, the first cycle only consume $N$ adders delay, which increments by an adder delay after each cycle. As seen in Figure 6.2,

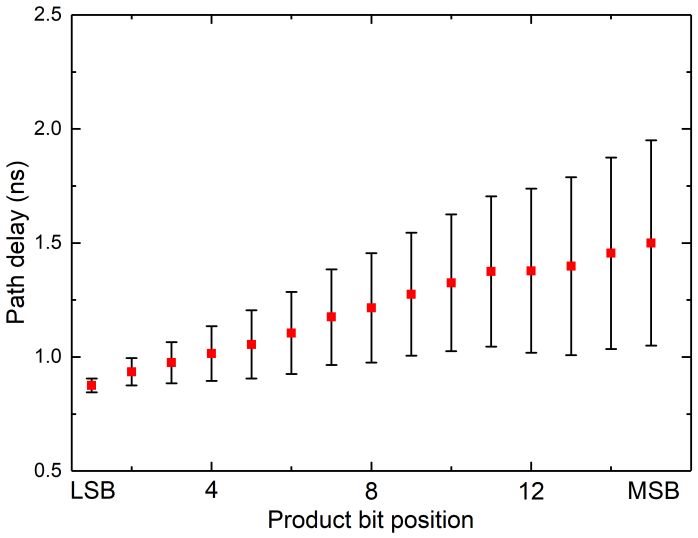



FIGURE 6.1: Path delay distribution of a conventional multiplier across product bit positions

critical paths occur in the LSBs, in this case, LSBs would start to fail once the timing violations happen.

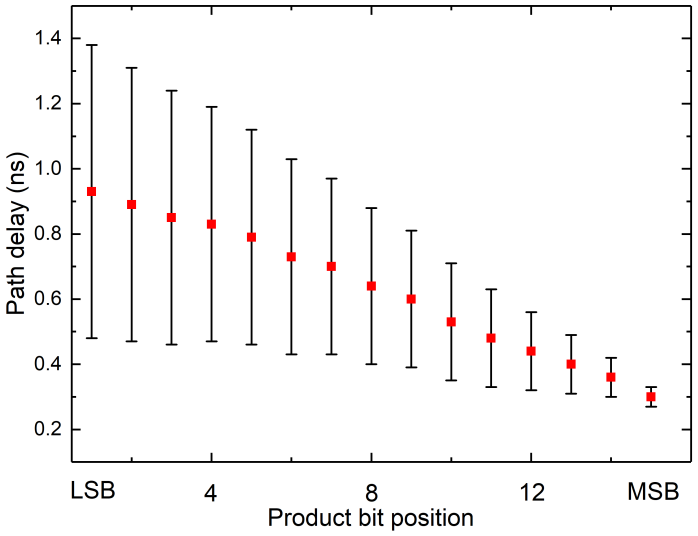



FIGURE 6.2: Path delay distribution of a msb-first multiplier across product bit positions

## 6.2 Realization of Variable Latency in the Serial Datapath

A novel timing error mitigation technique for serial computing is proposed in the section. The critical path of datapath is modified to be sensitive to input signal value. This can easily create path delay imbalance for each result bit with the adjustment of input data, more path slack can be achieved to provide a guard-band for timing error during MSB computation. The basic idea is demonstrated on a multi-input adder.

### 6.2.1 General Approach

Figure 6.3 presents a conventional combination logic circuit with its critical path, $t_{d,max}$. It should be mentioned that, in bit-serial structure, each result bit is generated from the same arithmetic datapath, thus, the MSBs (Most Significant Bits) and LSBs (Most Least Bits) of computation results have a very close latency.

To illustrate the basic idea of the error-mitigation variable latency approach in the serial system, a combination logic circuit with an adjustable critical path can be seen in Fig. 6.4. When the MSB computation is implemented, the signal is activated, and part of the combination gates can be bypassed. Therefore, the MSB group end-points ($t_{d,max,MSBs}$) can have more timing slack than the LSB group end-points ($t_{d,max,LSBs}$). In our work, the timing guardband between the two groups is achieved using a modified multi-input adder tree, where the redundant addition can be bypassed during the MSB operations.

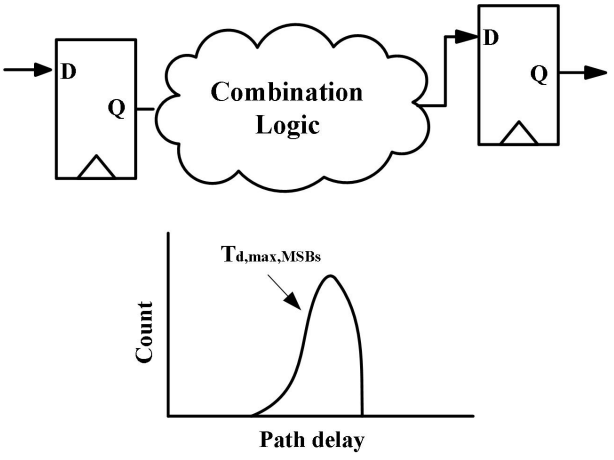



FIGURE 6.3: Carry save adder based multiple input adder tree with the proposed structure

### 6.2.2 Modified Adder Tree with Variable Latency

Multi-operand addition forms the core of many arithmetic operations. These computations can be useful for energy efficient circuits that use multiplication and division and

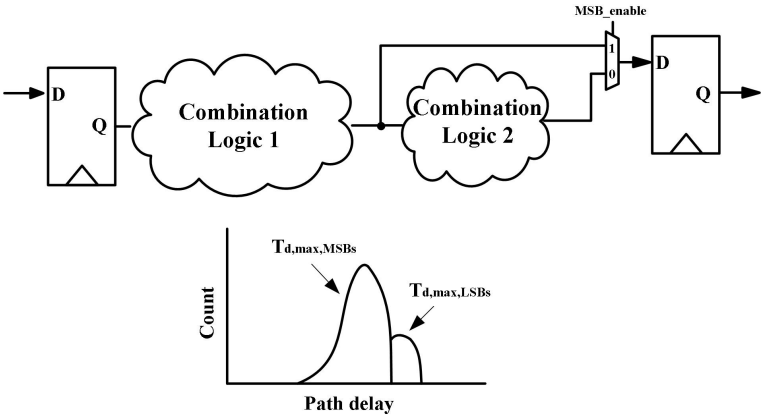



FIGURE 6.4: Carry save adder based multiple input adder tree with the proposed structure

for summing large numbers of data that are applicable in wireless sensor nodes and in environmental and remote health monitoring. The function of multi-operand adders can be expressed as Equation 6.1 , where $O$ is and operand and n is the total number of operands.

$$S = \sum_{p=0}^{n} O_p \tag{6.1}$$

In terms of hardware implementation, traditional multi-operand adders use ripple carry adder to calculate two of operands and sum up the result for a final summation. The delay of the adders stages increases with the number of stages due to the carry propagation, a $m$ operands $n$-bit multi-input adders will cost $O(n + 2m)$ gate delay, while overflow issue is considered. Accordingly, ripple adder is not a primary choice for IoT wireless sensor nodes, which requires energy-efficient computation.

The multi-operand carry-save adder (CSA) uses a number of CSAs interconnected as a multilevel adder tree to add more than one number per cycle. The number of adder tree levels determines the operation time. The conventional $m$ inputs carry-save addition is based on an array structure, which repeats the single carry-save addition $m - 2$ times. An 18-input array CSA is displayed in Figure 6.5(a) where 16 CSA blocks are needed. Instead of arranging the CSA blocks as an array, a better way to reduce the operation time is to organize CSAs in a tree formation, which is commonly known as a Wallace tree. This arrangement is illustrated in Figure 6.5(b). The number of levels of the CSA Wallace tree is $\log_{\frac{3}{2}} m$ for m input operations; thus, it can be seen that 16 CSA blocks and 6 tree levels are needed in the 18 inputs CSA tree.

As mentioned, n-bit serial computation is realized by repeating a single-bit operation $n$ times. Thus, each bit of the final result is generated in order using the same datapath. Since the input signals may affect the circuit latency, computation time in each cycle should be varied. Take the array-based CSAs for instance, once an input operand equals zero, a half adder operation can be skipped. However, for the Wallace-based CSAs, the

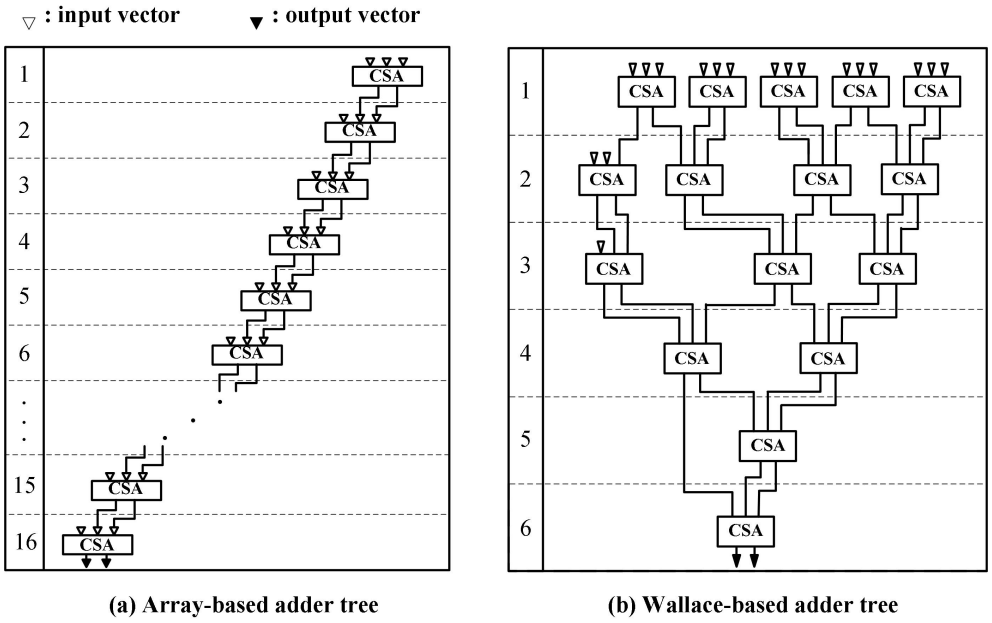



FIGURE 6.5: Carry save adder based multiple input adder tree with array and Wallace

circuit latency would not be considerably affected by the input data; that is, because many CSA blocks are implemented simultaneously in each tree level, the possibility to bypass all CSA computations in the same level would be quite low. In this case, we propose a hybrid tree based on array and Wallace tree structures; an 18-operand adder tree design can be seen in Figure 6.6, where the first 4 layers can be viewed as a 9-operand Wallace-based CSA, while the remaining adder blocks are organized in an array structure. To realize variable-latency computing, the input operands that have a high possibility of being "l" are fed into the Wallace tree-based circuit, while other operands likely to have a "0" value are connected with an array-based CSA in each tree level. Therefore, once the input operands from the $5^{th}$ layer equal zero, the corresponding redundant computations can be bypassed to reduce the circuit delay. Conventionally, the proposed multi-operand CSA can have a moderate speed between the array and the Wallace tree structures. This CSA can also demonstrate a shorter path delay than the Wallace tree design when half of the input signals are equal to zero.

## 6.3 Case Study of An FIR filter Implementation

In this section, the proposed approach is demonstrated on a linear low-pass FIR filter implementation, which is one of the important applications in wireless sensor nodes. Since the differences between the values of the FIR coefficients are quite significant, the described approach in section 6.2 is very suitable for the FIR filter implementation using a serial datapath.

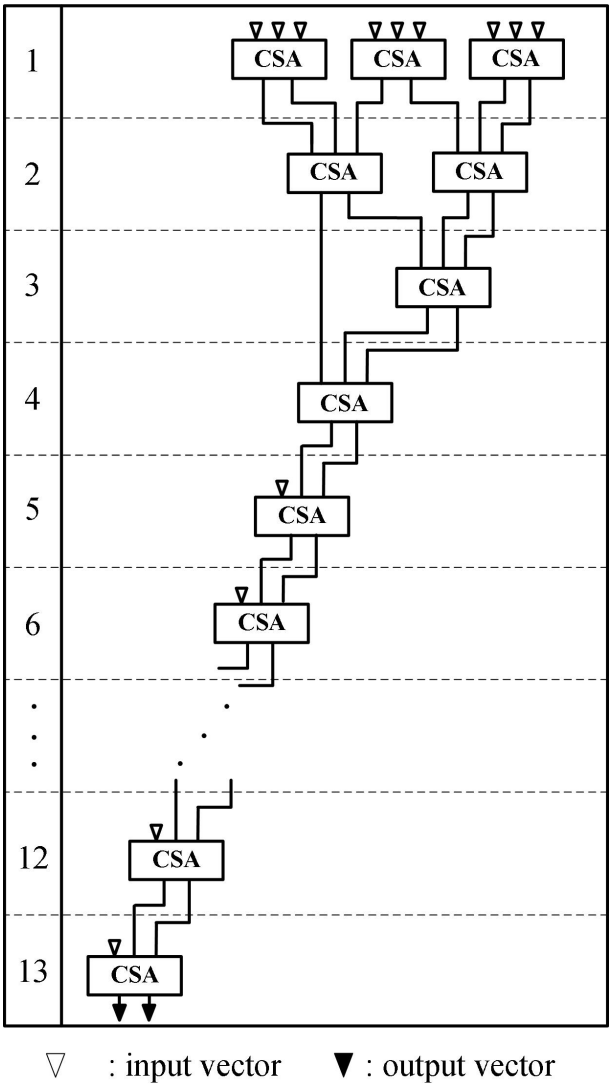



FIGURE 6.6: Carry save adder based multiple input adder tree with the modified structure

### 6.3.1 Linear FIR Filter Coefficients Analysis

The impulse response of an ideal low-pass filter can be represented by the Equation 6.2 , where $\Omega_c$ denotes the cut-off frequency.

$$h_D[n] = \frac{\Omega_c}{\pi} * \frac{sin(n\Omega_c)}{\Omega_c} \tag{6.2}$$

It can also be rewritten by replacing $\Omega_c = 2\pi F_c$ in the Equation above, to obtain it in terms of the normalised cut-off frequency $F_c$, as shown in Equation 6.3 .

$$h_D[n] = 2F_c * \frac{sin(n\Omega_c)}{\Omega_c} \tag{6.3}$$

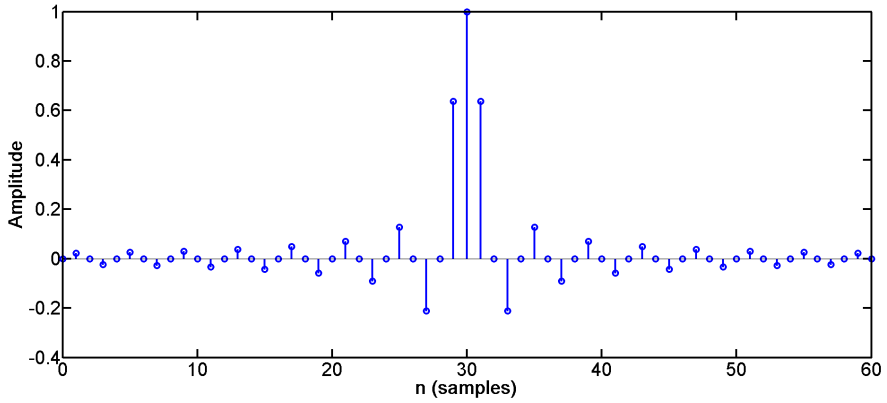



FIGURE 6.7: Impulse response of an ideal low-pass filter to 2M+1 samples (M=30)

As seen in Equation 6.3, the impulse response of a low-pass filter is a *sinc* function, and the maximum filter coefficients should be approximately equal to $2F_c$. If the coefficients are divided by $2F_c$, their quotient results can be expressed as a $\frac{1}{sinc}$ function, as shown in Equation 6.4. In this design, the concept of 2's complement is utilized for the binary representation, and the decimal range of the filter coefficient is determined by the largest coefficient data. Since the differences between the absolute values of the coefficients are very large, a large number of the significant bits in the coefficient binary data are equal to zero. As illustrated in the section 6.2.2, the variable latency multi-operand adder can bypass these additions when the corresponding input signals equal zero.

$$Q_c = \sum_{n=1}^{2M+1} \frac{\Omega_c}{\mid sin(n\Omega_c) \mid} \tag{6.4}$$

For fixed-point FIR digital design with $n$-bit computation accuracy, a group of $\frac{n}{2}$ MSBs can have timing guardbands when the quotients $Q_c$ between $2F_c$ and the corresponding FIR coefficients are larger than $2^{\frac{n}{2}}$. Accordingly, if the computation bit-width is 8-bit and 16-bit, $Q_c$ needs to be larger than $2^3$ and $2^7$, respectively. The $Q_c$ distribution in a symmetric low-pass FIR implementation with variable tap numbers can be seen in Figure 6.8, where the normalized cut-off frequency $\Omega_c$ varies from $0.1 * \pi$ to $0.9 * \pi$ radians/sample. With the increase in tap number, $Q_c$ *larger than* 8 takes a growing proportion and begins to rank first in the 17-tap FIR implementation, which meets our design requirement in 8-bit computation. Using the 129 tap number, the difference between the coefficient values becomes very large, and $Q_c$ *larger than* 128 consumes the largest inside part. In this case, a 16-bit FIR design computation can also be modified with the proposed error-tolerant technique.

To further analyse the linear FIR filter coefficients, 800 groups of random FIR coefficients for 17-tap implementation are sampled. Figure 6.9 plots the average possibility of each bit in these coefficients being 1. This plot provides a method to identify that more than half of the coefficients may have a 0 value in the most significant 4-bit position,

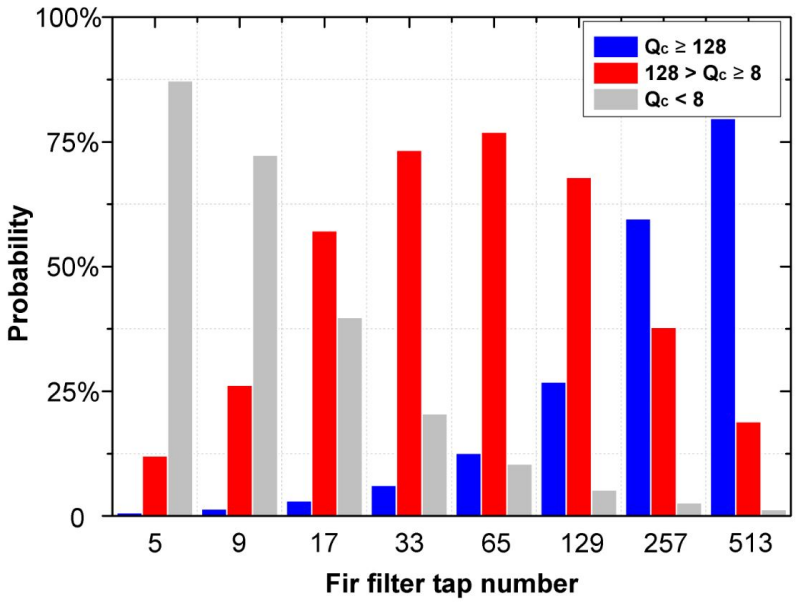



FIGURE 6.8: Distribution of $Q_c$ among FIR filter implementation with various tap number

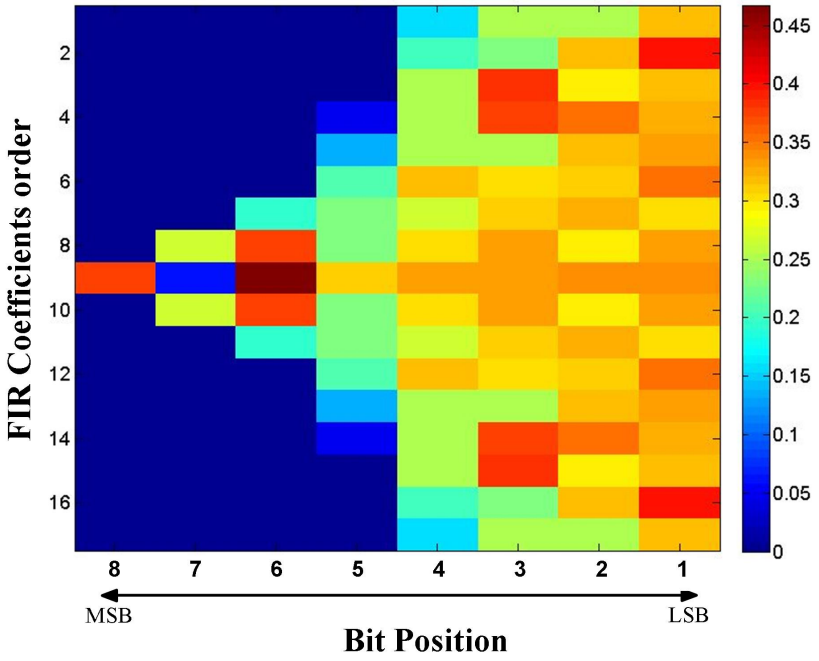



FIGURE 6.9: The probability of each bit in the sampled FIR coefficients being one for a 17-tap implementation

which reduces the redundant computation times using the proposed adder tree circuits, regardless of the input signal value. Therefore, the MSB computation should have more slack than the nominal computation. It should be mentioned that all the negative coefficients are converted into absolute values during the analysis, while the corresponding full adder is replaced by a subtractor for correct function.

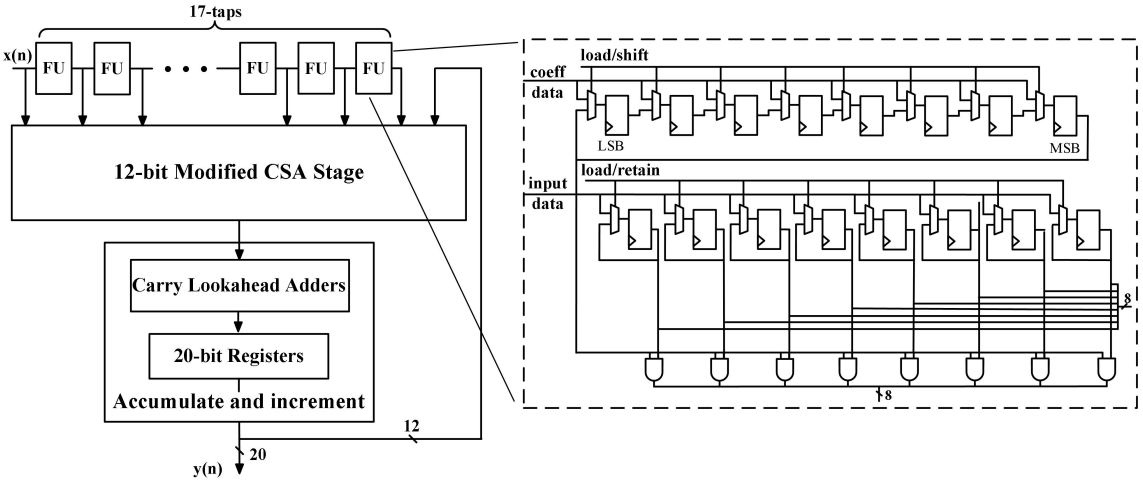



FIGURE 6.10: Modified datapath of FIR implementation based on serial architecture

### 6.3.2 VLSI Design

The proposed approach is based on achieving a guardband between the MSB and LSB computation delay across the design. The guardband is achieved using a modified carry-save adder tree with a bypassing function. In this section, we describe how this is realized at the circuit level with a very low area and power overheads.

Figure 6.10 shows the datapath for a 17-tap FIR filter with 8-bit computation, containing three main parts, function units, the adder stage and the accumulate and increment block. Inside the function unit, coefficient bits are shifted left in each clock cycle so that the partial product is ANDed from the most significant bit to the least significant bit. In each cycle, 17 partial products are generated in the function unit (FU) and delivered to the adder stage. To avoid overflow, a 20-bit wider adder structure is implemented for accumulation and incrementation. Therefore, a complete 20-bit sample result is generated every eight clock cycles.

In our design, the critical path starts from FU and ends in the accumulation registers and the circuit latency is progressively reduced along with the increase of result bit number. First of all, MSB computation can save up to nine half adder gate delay compared with LSB group's by using the modified CSA tree. This is because, there are more zero-valued bit during MSB operation and some of the additions can be bypassed or simplified for reduced path delay, which is illustrated in section 6.2.2. Secondly, the proposed FIR adopts MSB-first operation, after the first clock cycle, the maximum effective word-length of accumulation register would not be more than 13-bit. In this case, no half adder is required for increment and another 8 half adder gate delay is saved. In 8-bit computation, if we define the first four bit as MSB group, they would have around 17, 16, 15 and 14 half adder gate delay as an additional guard band respectively.

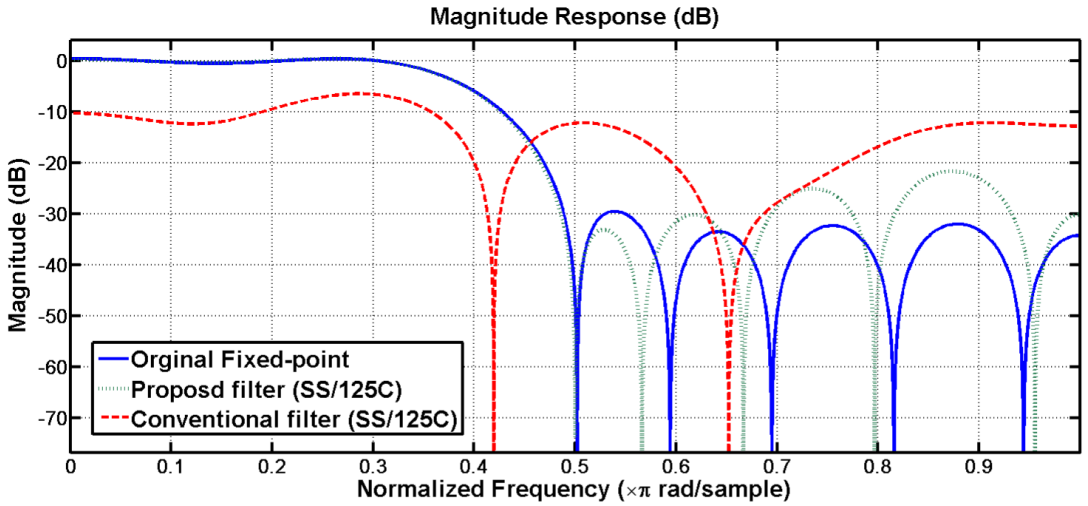



FIGURE 6.11: Frequency response of the conventional design versus proposed design in the worst case (SS process corner and 125C temperature)

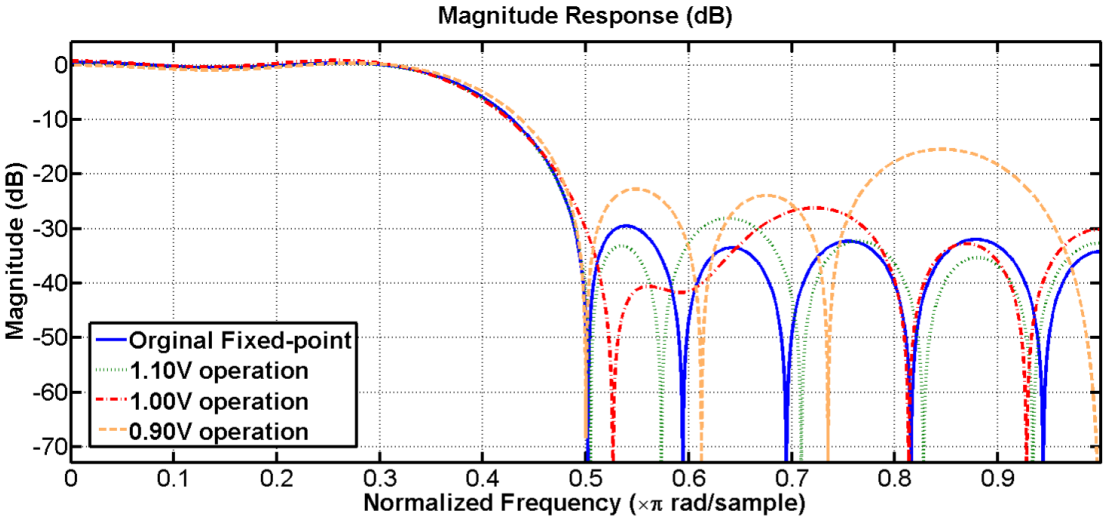



FIGURE 6.12: Frequency response of the proposed design with the different supply voltage in the typical case (TT process corner and 25C temperature)

## 6.4 Simulation Results

As mentioned in the previous section, proposed design can improve its robustness against timing errors at the cost of computation accuracy. To illustrate this, we consider a 17-tap low-pass FIR filter with the following specifications ($F_{pass} = 0.3, and F_{stop} = 0.5, equirpple$).

### 6.4.1 Robustness Against Timing Error

For the filter design, the passband and stopband can contain oscillations, which can be called ripples here. In a fixed-point implementation, timing violations may lead to the degradation of filter performance, thereby increasing the ripples of both the passband

TABLE 6.1: Maximum ripple for the conventional and proposed filters in the worst case

| **17-tap** | **Pass Band $\delta$** | **Stop Band $\delta$** |
|---|---|---|
| Original | 0.0505 | 0.0195 |
| Proposed | 0.0559 | 0.0794 |
| Conventional | 0.7488 | 0.2512 |

TABLE 6.2: Maximum ripple for the proposed filter with different voltage.

| **Proposed 17-tap** | **Pass Band $\delta$** | **Stop Band $\delta$** |
|---|---|---|
| Original | 0.0505 | 0.0195 |
| 1.10V | 0.0777 | 0.0398 |
| 1.00V | 0.0965 | 0.0562 |
| 0.90V | 0.1220 | 0.1585 |

and the stopband. To better present the performance change of the proposed filter, we calculate the stopband and passband ripples.

Initially, the proposed and conventional ripple adder-based designs are both simulated in the worst case (slow corner, 125°C) without a timing margin. The filter characteristics under the timing variations are shown in Figure 6.12, and the corresponding error metrics (passband and stopband ripple) are shown in Table 7.1. The simulation results show that the conventionally designed filter meets the function failure, while the proposed design still works with slightly degraded performance.

If the worst case margin is taken into consideration, the proposed design can retain error-free operation at a nominal voltage domain. To identify the dynamic voltage scaling method on the proposed circuit, the simulations are implemented with different voltage domains. The results in Figure 6.11 and Table 6.2 show that the proposed filter can maintain acceptable performance with the supply voltage scale from 1.20 V to 0.90 V (Figure 6.12). Beyond 0.90 V, the MSB computations begin to fail, which results in significant performance loss.

### 6.4.2 Power Saving with Dynamic Voltage Scaling

Power consumption savings in our proposed FIR filter are obtained as shown in Figure 6.13 under different circumstances of SS, TT and FF corners and -40°C, 25°C and 125°C. Compared with the TT process corner, 25°C and 1.2 V supply voltage, this proposed approach achieves 21%, 18% and 15% power savings for the best case (FF corner, -40°C), nominal case (TT corner, 25°C) and worst case (SS corner, 125°C), respectively at a voltage of 1.10 V. At the 1.00 V voltage domain, the approach has a power benefit

of 21%, 18% and 15% at different operation modes. When the voltage scales down to 0.90 V, the maximum power reduction can be obtained by up to 46%.

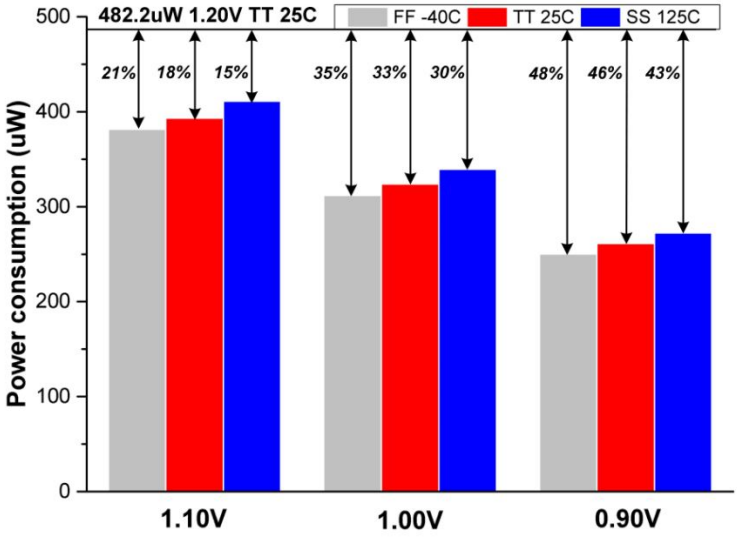



FIGURE 6.13: Power consumption saving in different operating modes

### 6.4.3 Overheads of Proposed Approach for FIR Filters

The proposed variable latency approach trades delay for error resilience over a given timing guardband and therefore increases the minimum clock period. To illustrate this, a tentative comparison to a conventional implementation is made. An FIR implementation in serial architecture achieves a minimum delay of 5.42 ns (136 MHz) in a 65 nm process technology. The proposed approach has a conservative timing guardband of 1.72 ns, which is a 32% delay overhead, and no extra hardware cost is required based on the modified adder tree structure. As shown in Table 6.3, compared with the conventional implementations using the ANT and path shaping approaches, our design can provide more guardband, thereby saving more power with voltage scaling. Although the proposed design does not outperform conventional razor-based voltage scaling methods in terms of power saving, a 25% acceptable delay overhead is required without any extra hardware cost.

In addition, the 8-bit FIR implementation with the proposed approach shows significant advantages in terms of hardware. With the voltage over-scaling, the design has only a 0.58 pJ energy cost per operation with acceptable performance, which outperforms other counterparts. In general, These findings suggest the proposed adaptive latency approach is well suited for serial design.

TABLE 6.3: Comparisons between the state-of-the-art error-resilient techniques demonstrated on an 8-bit FIR filter

| Approach | Razor [68] | ANT [100] | PDS [101] | Proposed |
|---|---|---|---|---|
| Process | 90 nm | 32 nm | 32 nm | 65 nm |
| Type | EDAC | EDAC | EM | EM |
| Delay overhead | 24% | 72% | 20% | 25% |
| Area overhead | 26% | 20% | 22% | 0% |
| Guardband | $N/A$ | 20% | 20% | 32% |
| Power saving | 53% | N/A | 23% | 46% |
| Area/tap ($\mu m^2$) | 525 | N/A | 516 | 362 |
| Min. Energy/tap (pJ) | 2.01 | N/A | 0.67 | 0.58 |

EDAC: Error Detection and Correction
EM: Error Masking

## 6.5 Summary

A cost-efficient timing error prevention method for serial sequential circuits is presented in this paper. A novel approach is proposed to mitigate the timing errors at the circuit level for serial computing. This is achieved by modifying the path delay distribution for different bit results. For instance, the path delay for the LSB computation is designed as the longest one, while the MSB path is the shortest. Accordingly, timing violations weakly result in large computational accuracy loss. In general, the proposed design can maintain graceful performance without any timing margin and acquire significant power saving up to 46% with different operating conditions with the worst case margin.

# Chapter 7

# Timing-Error-Masking and Prevention Approach for Serial Computing

In this chapter, two novel approaches are proposed to bound the magnitude of timing errors and demonstrated on a 16-tap FIR filter using 65-nm CMOS process. We initially reverse the bit order in DA-based serial computations and provide MSB paths more guardband by time borrowing, which ensures the timing errors weakly affect the computation results. In addition, we apply the shifted-phase clock on the end-point registers, thereby providing extra timing slack globally without any effects on the system's sampling rate. We show that this error-masking approach is more cost-efficient compared with state-of-the-art error-resilient techniques. Besides, we also combine the clock-gating with the conventional error prediction technique to generate a DA-based error-prevention approach, more circuit flexibility and latency saving is achieved.

## 7.1 Distributed Arithmetic

Distributed Arithmetic computes the inner product of two vectors (one of which is a constant) in parallel, that is a bit-serial operation in nature. The main advantages of DA computation is that no multiply operations are required, which are replaced by precomputed look-up tables. The DA-based multiplication and accumulation (MAC) can be expressed as:

$$y_k = \sum_{k=0}^{L-1} h_k * x_k \tag{7.1}$$

where $x_k$ and $y_k$ represent the input and output data respectively. If we consider that each input data is a $N$-bit two's complement binary number and can be represented as:

$$x_k = -b_{k(N-1)}2^{N-1} + \sum_{i=1}^{N-1} b_{k(N-1-i)}2^{N-1-i} \tag{7.2}$$

where $b_{ki}$ is the $i$ th bit of the input data $x_k$. Accordingly, $b_{k0}$ and $b_{k(N-1)}$ are the least significant bit and most significant bit (sign bit) respectively. Substituting equation. 7.1 in equation. 7.2:

$$y_k = -\sum_{k=0}^{L-1} h_k b_{k(N-1)}2^{N-1} + \sum_{i=1}^{N-1}\left[\sum_{k=0}^{L-1} h_k b_{k(N-1-i)}2^{N-1-i}\right] \tag{7.3}$$

If we replace the term in bracket by

$$q_i = \sum_{k=0}^{L-1} h_k * b_{k(N-1-i)} \tag{7.4}$$

The MAC operation can be represented by equation. 7.5, which shows the $i^{th}$ intermediate result of a MSB-first DA operation. In terms of hardware implementation, the bit-serial input data $b_{ki}, ..., b_{2i}, b_{1i}, b_{0i}$ is used to form the look-up tables address, and all possible linear combinations of the partial products are stored in the look-up tables.

$$y_k = -q_{N-1}2^{N-1} + \sum_{i=1}^{N-1} q_i 2^{N-1-i} \tag{7.5}$$

## 7.2 Proposed Error-Resilient Approaches

With the presence of PVT variations, the path delay increases relative to the fixed clock period, eventually resulting in failure when the critical paths are activated. Therefore, the robustness of the circuit can be improved via an increased timing margin. As illustrated in section 7.1, a $N$-bit DA-based computation consumes $N$ clock cycles, and each bit of the final result is generated every cycle using the same addressing, fetch, and accumulation circuits. Hence, delay imbalance across result bits should be very close, which is different from conventional arithmetic circuits. Based on this characteristic, we propose to trade a small amount of area overhead to allow the critical path to correspond to the LSB computation, thereby bounding the error magnitude of timing errors. In this section, two error-resilient approaches are proposed.

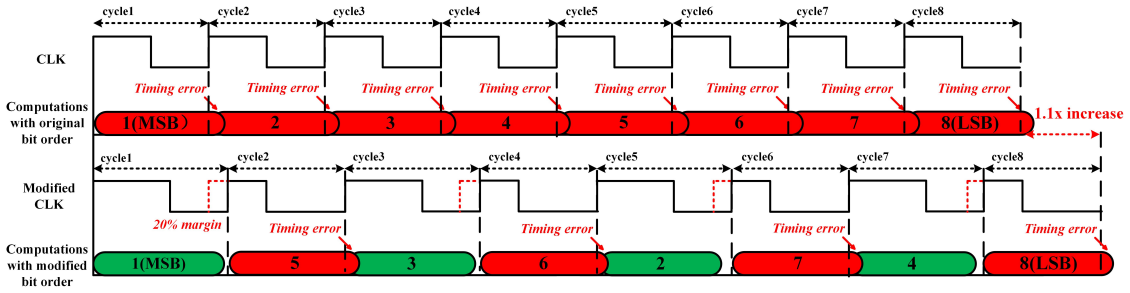



FIGURE 7.1: Timing diagram of the reversed bit approach

### 7.2.1 Bit reversed approach for time borrowing

The magnitude of an error is strongly correlated to the bit order of the result, and large logic errors are generated resulting from sign-bit (MSB) failure. The basic idea of the first approach is to rearrange bit order in serial computing (DA-based circuits), allowing a significant computation (MSB group) to be followed by a relevantly insignificant one (LSB group). Accordingly, any late arriving transition during MSB computations can be elegantly masked by borrowing time from the next cycle. To better illustrate this approach, the DA-based computation is modified as follows:

$$y_k = -q_{N-1}2^{N-1} + q_{\frac{N}{2}}2^{\frac{N}{2}-1} + \sum_{i=1}^{\frac{N}{2}-1}\left[q_{i+\frac{N}{2}}2^{\frac{N}{2}-1-i} + q_i 2^{N-1-i}\right] \tag{7.6}$$

where the $N$-bit pipeline end-points are separated into two groups, one group of $\frac{N}{2}$ LSBs, and the other group of $\frac{N}{2}$ MSBs, respectively. Because the MSB group has an increased time period through time borrowing, the LSB group is intended to fail first due to timing violations, and their intermittent failure would not cause an unacceptable reduction in system performance. Figure 7.1 displays the timing diagram of the proposed approach with the presence of process variations, which is demonstrated on an 8-bit DA-based circuit. It can be seen that with the original clock signal, the timing violations occur in every single stage due to dynamic variability. Hence, the computation accuracy is significantly affected. To reduce the magnitude of timing errors efficiently, a small amount of operation time is traded. In this design, the whole operation latency is increased by 10%, while 20% extra timing margin is provided for the MSB group using the modified clock signal. Therefore, the majority of variation-induced timing violations in MSB computations can be masked. Regarding the delay overhead, 10% more operation latency is required, which can be alleviated by sizing the arithmetic unit at the expense of hardware cost. According to the different reliability requirements, the circuit error-resilience can be correspondingly adjusted during the RTL design stage.

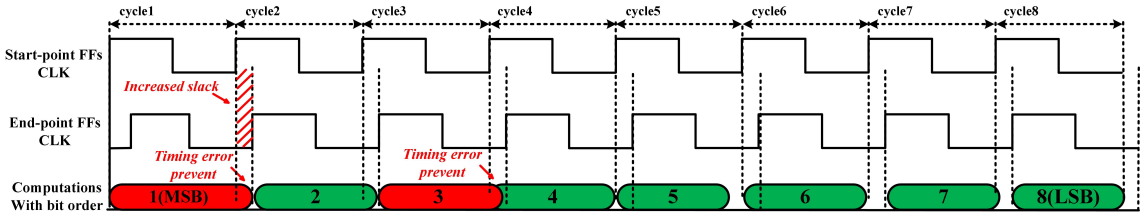



FIGURE 7.2: Timing diagram of the shifted phase clock approach

### 7.2.2 Shifted-phase clock approach

The DA-based computation can be implemented with the following stages: addressing, data fetch, and accumulation, where the output data of the arithmetic unit would not be changed until the new look-up table address is provided. In other words, the computation results in the previous clock cycle remain unchanged before the implementation of accumulation operation in the current cycle. Therefore, a shifted-phase clock signal can be applied on the end-point registers in the DA-based circuit to increase the timing margin. However, the shifted clock phase must be less than the time period used for addressing and data fetch, otherwise incorrect results would be fed into the end-point registers. As described in approach I, the arithmetic unit is up-sized to shorten the clock period, thereby avoiding increased total operation time. Accordingly, the addressing and fetching operations may consume increased portions among the total clock period, although in this case more guardband can be provided by using the shifted clock signal.

The corresponding timing diagram is shown in Figure 7.2, and it is also demonstrated on an 8-bit DA-based circuit, where we assume that the timing errors are detected in the first and third clock cycles. Although the correct results are generated after the falling edge of the original clock signal, the shifted phase clock provides enough time period to complete the computations and help the end-pint resister capture the correct data. Therefore, timing violations can be successfully prevented with the assistance of the proposed method.

### 7.2.3 VLSI Implementation

The structure of the proposed error-resilient distributed arithmetic architecture is presented in Figure 7.3, which consists of a look-up table with $2^L * M$ bits for an $L$-tap FIR filter. Each of the $2^L$ look-up table entries has a corresponding register for storing input data, where the multiplexors are used to select the bit number dependent on the $log_2N$-bit counter and generate the address for the look-up table. The select signals are modified according to the counter value, and the example of a 3-bit counter case can be seen in Table 7.1. The low right shows a $N + M$-bit accumulator, and the computation results are generated every $N$ clock cycles.

Figure 7.4 shows the circuit diagram of the proposed clock generator and corresponding waveforms. The first modified clock signal (termed $Mclk0$) is generated by periodical selection between the global clock signal and its delayed signal using a multiplexer. A simple frequency divider (based on a D flip-flop and an inverter) is used to provide the select signal, where the high-level output means the significant (MSB) computation is being implemented and the global clock signal should be selected. Another modified clock signal $Mclk1$ is easily generated by delaying the $Mclk0$ signal using a number of buffers. This clock signal is only fed into the end-point registers, while the other sequential circuits in the DA system are all driven by signal $Mclk0$.

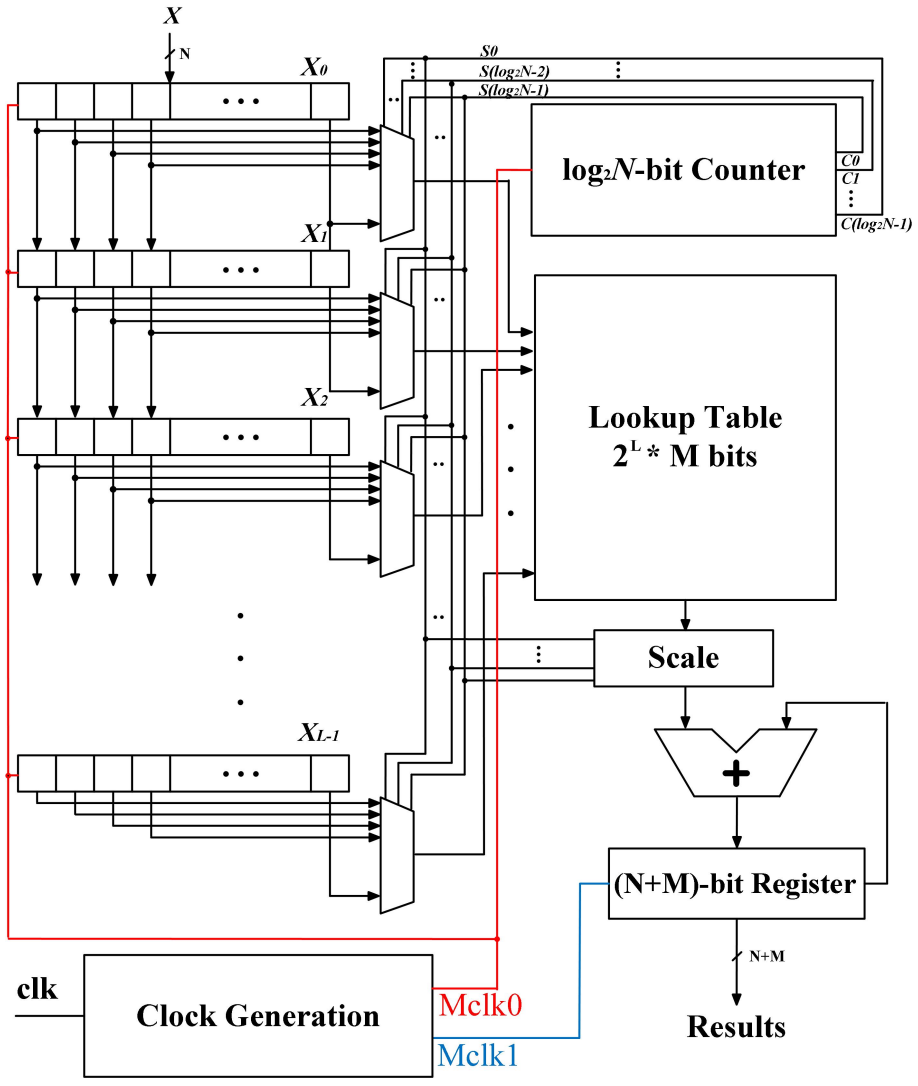



FIGURE 7.3: The proposed error-masking DA-based FIR filter architecture

Figure 7.4 shows the circuit diagram of the proposed clock generator and corresponding waveforms. The first modified clock signal named $Mclk0$ is generated by periodically selecting the global clock signal and its delayed signal. A simple frequency divider (based on a D-flipflop and an inverter) fed into delayed clock signal is used to generate the select signal, while the high-level output means the significant (MSB) computation is

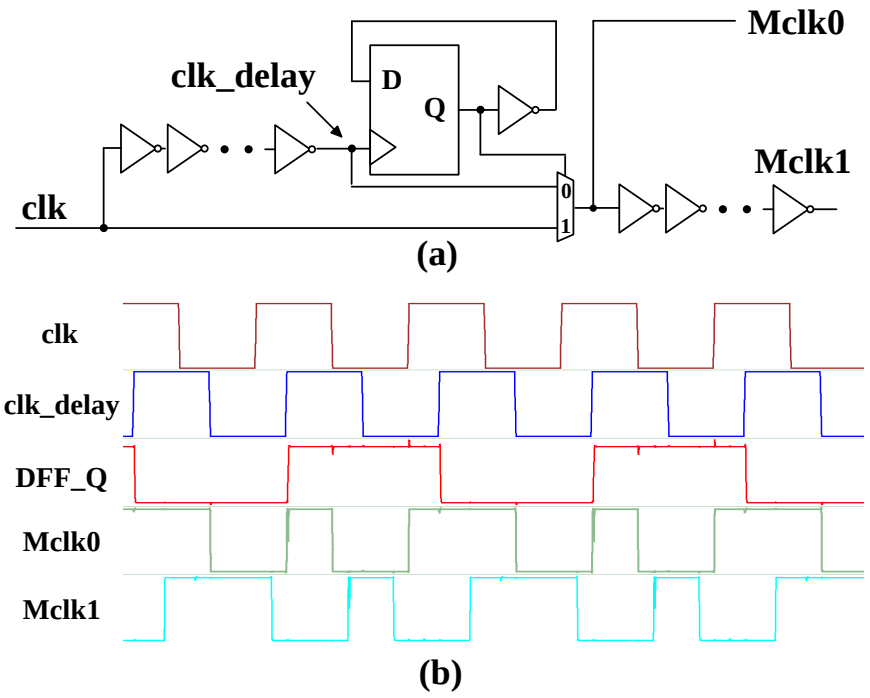



FIGURE 7.4: The circuit block and corresponding spice simulated waveforms of the proposed clock generator

TABLE 7.1: Reversed bit generation table.

| Counter ($b_2b_1b_0$) | Decimal (Counter) | Select ($b_0b_1b_2$) | Decimal (Select) |
|---|---|---|---|
| 000 | 0 | 000 | 0 |
| 001 | 1 | 100 | 4 |
| 010 | 2 | 010 | 2 |
| 011 | 3 | 110 | 6 |
| 100 | 4 | 001 | 1 |
| 101 | 5 | 101 | 5 |
| 110 | 6 | 011 | 3 |
| 111 | 7 | 111 | 7 |

being implemented. Another modified clock signal $Mclk1$ is easily produced by delaying $Mclk0$ using an appropriate number of buffers, which is only connected with end-point registers, while the other sequential circuits in the DA circuits are all driven by signal $Mclk0$.

### 7.2.4 Simulation Results

As mentioned in the previous section, the proposed approach can improve circuit robustness against timing errors at the cost of hardware area and delay. To demonstrate this, the proposed architecture was applied on a digital FIR filter. This design is synthesized using the Synopsys Design Compiler with the ST 65nm CMOS process.

#### 7.2.4.1 Robustness Against Timing Error

To realize comparisons with conventional designs fairly, a 16-tap low-pass FIR filter with the following specifications (16 *taps*, $F_{pass} = 0.3$, and $F_{stop} = 0.5$, *equirpple*) is considered. These filters are designed using the Matlab FDA tool and the fixed-point coefficients are generated. Both the proposed DA-based and conventional designs are simulated in the worst case (slow corner, 125°C) without any timing guardband. The corresponding filter characteristics are presented in Figure 7.5, compared with the magnitude response of the original fixed-point filter. Simulation results indicate that the conventionally designed digital filter meets the function failure while the proposed design can still work without any accuracy loss, owing to the delayed-phase clock signals on the end-point registers.

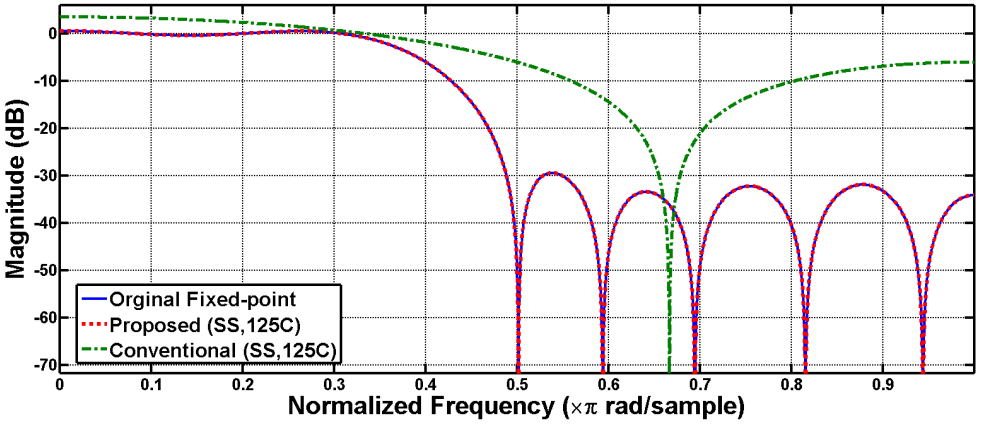



FIGURE 7.5: Frequency response of conventional design versus proposed design in worst case (SS process corner and 125C temperature)

It was proved that the proposed design can retain error-free operation at a nominal voltage domain with the presence of process and temperature variations. Therefore, the dynamic voltage scaling (DVS) method is applied on the proposed design, while the worst-case margin is taken into consideration. Specifically, simulations with different voltage domains in terms of filter magnitude response are implemented. The corresponding results in Figure 7.6 indicate that the proposed filter can maintain acceptable performance when the supply voltage reduces from 1.20 to 0.90 V. Beyond 0.90 V, MSB computations begin to fail, which results in significant performance degradation.

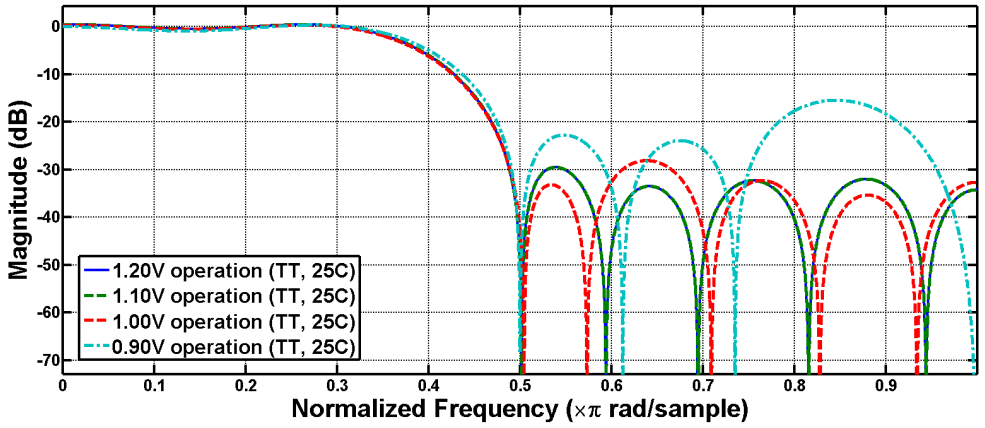



FIGURE 7.6: Frequency response of the proposed design with the different supply voltage in typical case (TT process corner and 25C temperature)

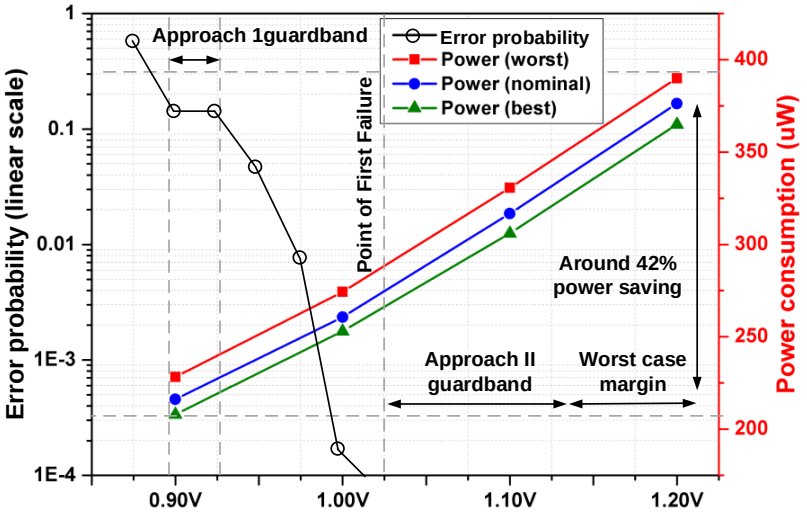



FIGURE 7.7: Power dissipation and the probability of error versus supply voltage with different process corners and temperature.

Because the simulation results of power consumption and computation accuracy are dependent on the input samples and the coefficients, random stimulus is implemented over the full input dynamic range. The power consumption savings in our proposed FIR filter are shown in Figure 7.7 under different circumstances of SS, TT, FF corners, and -40°C, 25°C, 125°C. Compared with the TT process corner, 25°C, and 1.2 V supply voltage, the proposed approach achieves 19%, 17%, and 12% power savings for best (FF corner, -40C), nominal (TT corner, 25°C) and worst(SS corner, 125°C) cases, respectively (at 1.10 V). At the 1.00 V domain, it can achieve a power benefit of 32%, 30% and 27% at different operation modes. When the voltage is reduced to 0.90 V, a maximum power

reduction of up to 42% can be obtained.

To better present the trade-off between the power consumption and computation accuracy, the error probability ($P_e$) serves as the measurement metric to evaluate the accuracy loss, which means it is probable that the output of computation differs from the correct one. Figure 7.7 also shows the error probability versus supply voltage, starting from the nominal 1.20 V, which includes the full worst-case design margin. If this margin is removed in the nominal case (TT, 25°C) using DVS, error-free operations can still be retained. The guardband provided by the variable clock phase approach allows the nominal operating point to be moved from 1.13 V to the point of first failure (PoFF) at 1.03 V. At this point, paths from the LSB group start to fail, but the corresponding error rate is extremely low. Beyond 0.90 V, MSB paths also begin to fail, which leads to a rapid decline in system performance.

Figure 7.7 also shows the probability of error versus supply voltage, starting from the nominal 1.20 V, which includes full worst-case design margin. If this margin is removed at the nominal case (TT, 25C) using DVS, error-free operations can still be retained. The guardband provided by the variable clock phase approach allows the nominal operating point to be moved from 1.13 V to the point of the first failure (PoFF) at 1.03 V. At this point, paths from the LSB group start to fail, but the corresponding error rate is extremely low. Beyond 0.90 V, MSB paths also begin to fail, which leads to a rapid decline in system performance.

#### 7.2.4.2 Overheads of the Proposed Approach

TABLE 7.2: Comparisons between state-of-the-art error-masking techniques.

| Approach | Razor [68] | ANT [100] | PDS [101] | Proposed |
|---|---|---|---|---|
| Process | 90 nm | 32 nm | 32 nm | 65 nm |
| Type | EDAC | EDAC | Error masking | Error masking |
| Delay overhead | 24% | 72% | 20% | 6.3% |
| Area overhead | 26% | 20% | 22% | 7.3% |
| Guardband | $N/A$ | 20% | 20% | 50% + 20% (MSBs only) |
| Power saving | 53% | N/A | 23% | 42% |

EDAC: Error Detection and Correction

In this design, our proposed approach trades extra delay and hardware cost for error resilience over a given timing guardband. To illustrate this, tentative comparisons to state-of-the-art conventional implementations demonstrated on a 16-tap FIR filter are implemented, as shown in Table. 7.2. The proposed approach provides approximately 50% of global guardband by shifting the clock-phase of end-point resisters, and has an

extra conservative timing guard-band for the MSB group, which is approximately 20% of the clock period. Compared to conventional implementations using ANT and path delay shaping approaches, our design can provide more guardband, thereby saving more power with voltage scaling.

In terms of circuit overheads, the increased operation time equals the provided guard-band by the proposed shifted-phase clock approach. For an 8-bit pipeline-based DA computation, the overall delay overhead is acquired by dividing 50% by 8, which is 6.3%. Although the proposed design does not outperform conventional razor-based DVS methods in terms of power saving, only 7.3% extra hardware cost is required. That is because the error-resilient method is mainly realized by modification of clock signal without using extra hardware circuits. Based on the comparisons mentioned before, the proposed method is significantly competitive in terms of cost-efficiency.

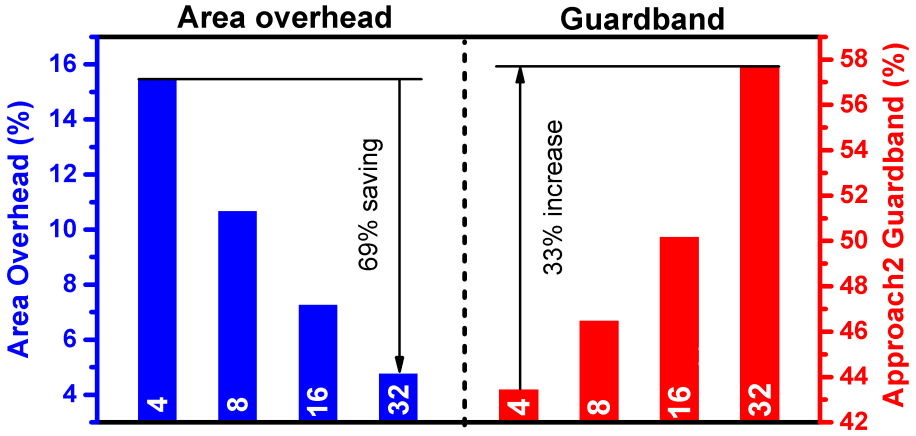



FIGURE 7.8: Area overhead and provided guardband versus FIR filter tap number

Since more operation time would be consumed in the addressing and data fetch with the increased look-up table size. It can be found that the proposed shifted clock-phase approach can bring more benefits along with the filter's tap number, as seen in Figure 7.8. In terms of area overhead, a 32-tap FIR filter can achieve approximately 69% savings compared with a 4-tap design, and only 5% extra hardware cost is required. On the other hand, up to 57% guardband can be provided by a proposed 32-tap design, which increases by approximately 33%.

## 7.3 Proposed Error-Prevention FIR filter

In the proposed error-masking design, the shifted-phase clock leads to a delay in the generation of the computation results. For the systems with global clock signal, the output data from DSP accelerators could be directly sent to MCU or other DSP block for further computations. In this case, the delay may result in timing violations in

the next clock cycle. To satisfy the requirements in the different digital systems, we present an error-prevention DSP design based on the clock-gating and shifted-phase clock, which can dynamically increase the circuit's timing slack according to the error prediction results, while the acceptable computation accuracy is achieved.

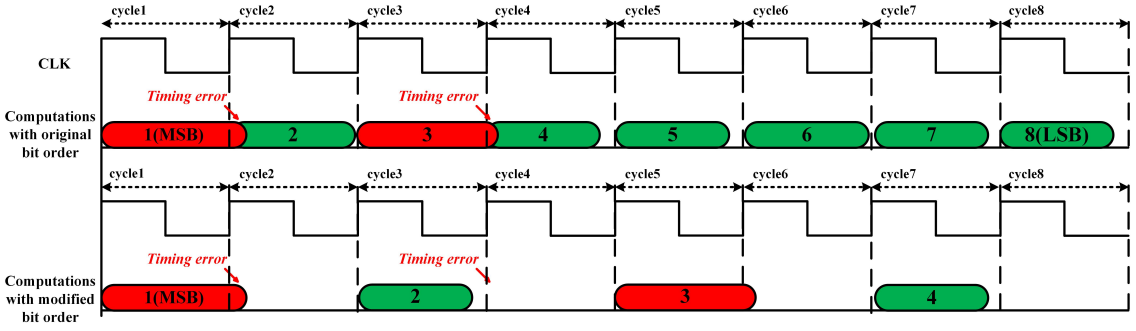



FIGURE 7.9: Timing diagram of the dynamic clock-gating approach

In this section, we propose a new concept called two-stage error-prevention. The basic idea is that repeat the implementations of the error prediction and prevention twice. Specifically, once the first error is predicted, the shifted-phase clock signal is utilized to provide timing slack with the accuracy loss of LSB. After that, if the suspicious timing error is detected again, it can be concluded that the variation-induced timing violations are very serious. In this case, the clock signals are dynamically gated and the clock period is doubled. For example, an 8-bit computation would be shifted into 4-bit's automatically.

The timing diagram of the clock gating approach can be seen in Figure 7.9, it is also demonstrated on an 8-bit serial circuit where we assume that the timing errors are detected in the first and third clock cycle. With the assistance of the proposed method, the clock period time is doubled. Accordingly, the timing errors can be successfully masked.

### 7.3.1 VLSI Implementation

Figure 7.10 shows the circuit schematic of the EPAP (Error Prediction And Prevention) system with two error generation blocks where the critical path is divided into two path segments. The two error generation blocks have different clock phases, a delayed clock signal is fed into the *Error generation*1 block. Specifically, once an error is predicted initially, signal *Error prevention*0 is activated and the shifted-phase clock would be applied on the end-point registers, thereby providing enough timing slack (named $d0$) to mitigate the suspicious timing violations. The delay of the clock signal is theoretically equal to $\frac{d0}{2} - T_{Hold}$, the difference between half value of the provided guardband and the Flip-Flop hold time. Afterwards, if a suspicious timing error is detected again and it

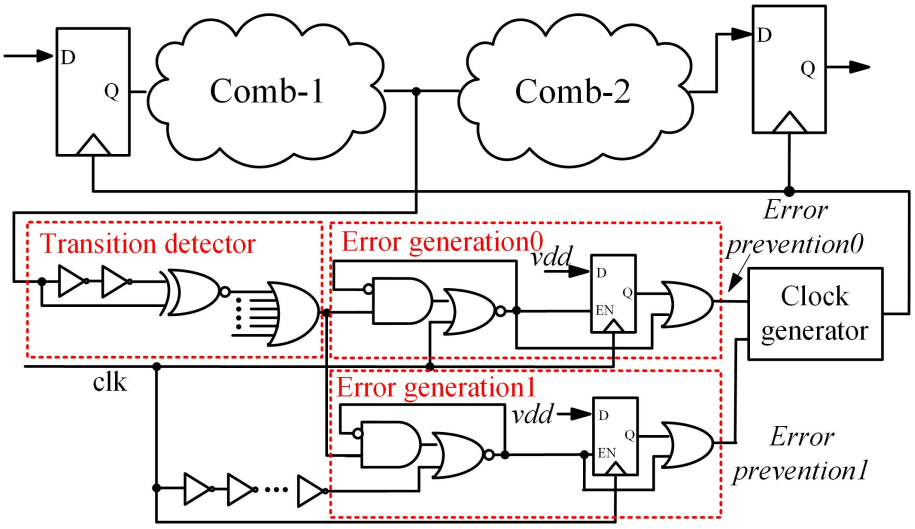



FIGURE 7.10: Error-prevention system with two Error generation blocks

occurs in the significant (MSBs group) computations, signal *Error prevention*1 becomes high-level and the time borrowing operation would be implemented.

The structure of the proposed error-prevention distributed arithmetic architecture is presented in Figure 7.11, which shares the same DA-based circuit with the error-masking design except for the *Error Prediction* and *Clock Generation* blocks.

Figure 7.12 shows the circuit diagram of the proposed clock generator and corresponding waveforms. The first modified clock signal is named $Mclk0$ and equals to the global clock signal when the signal *Error prevention*0 is zero. Once the select signal *Error prevention*0 becomes high-level, $Mclk0$ is generated by periodical combination between the global clock signal and its delayed signal. It should be noticed here the delayed time $d0$ equals to the guardband provided to MSBs group computations. A simple frequency divider (based on a D-Flipflop and an inverter) fed into the delayed clock signal are used to generate the select signal, while the high-level output means the significant (MSB) computations are being implemented. Another modified clock signal $Mclk1$ is generated by selecting between global clock signal $clk$ and the $Mclk0$ signal with a $d1$ delay, where $d1$ means the maximum timing guardband can be gained by shifting clock phase. The relationships between generated clock signals and *Error prevention* signals can be seen in Table. 7.3.

TABLE 7.3: Relationships between generated clock signals and *Error prevention* signals

| *Error prevention*0&1 | 00 | 10 | 11 |
|---|---|---|---|
| $Mclk0$ | clk | clk | *TB clk |
| $Mclk1$ | clk | delayed clk | delayed *TB clk |

*TB: Time Borrowing

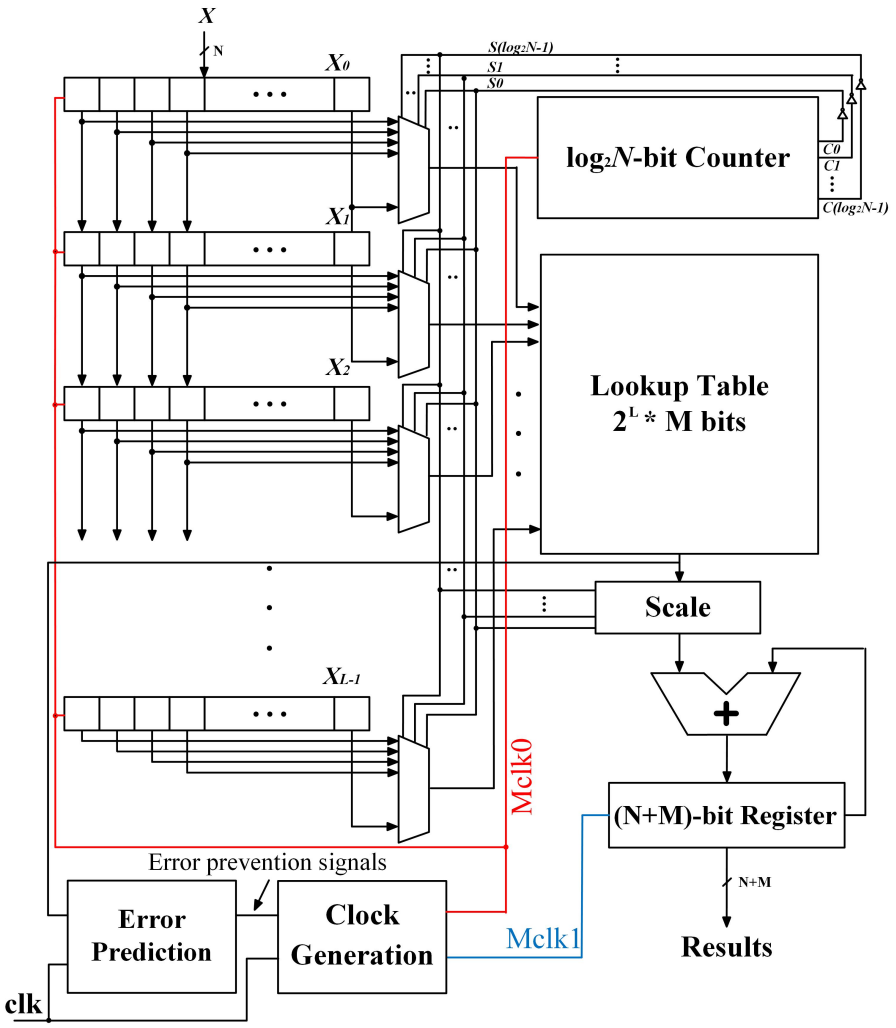



FIGURE 7.11: The proposed error-masking DA-based FIR filter architecture

### 7.3.2 Simulation Results

To demonstrate this, the proposed architecture is also applied to a 16-tap low-pass FIR filter with the following specifications ($F_{pass} = 0.3$, and $F_{stop} = 0.5, equirpple$). This design has been synthesised using the Synopsis Design Compiler with the ST 65nm CMOS process.

Same as the previous error-masking approach, the DVS (dynamic voltage scaling) method is applied to the proposed design, while the worst case margin is taken into consideration. The only difference is that the error-prevention design cannot retain error-free operation at nominal voltage domain with the presence of process and temperature variations. Once a suspicious error is predicted. Figure 7.13 shows the corresponding probability of error versus supply voltage, starting from the nominal 1.20 V, which includes full worst-case design margin. If this margin is removed in the nominal case (TT, 25C) using DVS, the effective computation word-length would be 7-bit. Beyond 0.90 V, MSB paths also begin to fail, which leads to a rapid decline in system performance.

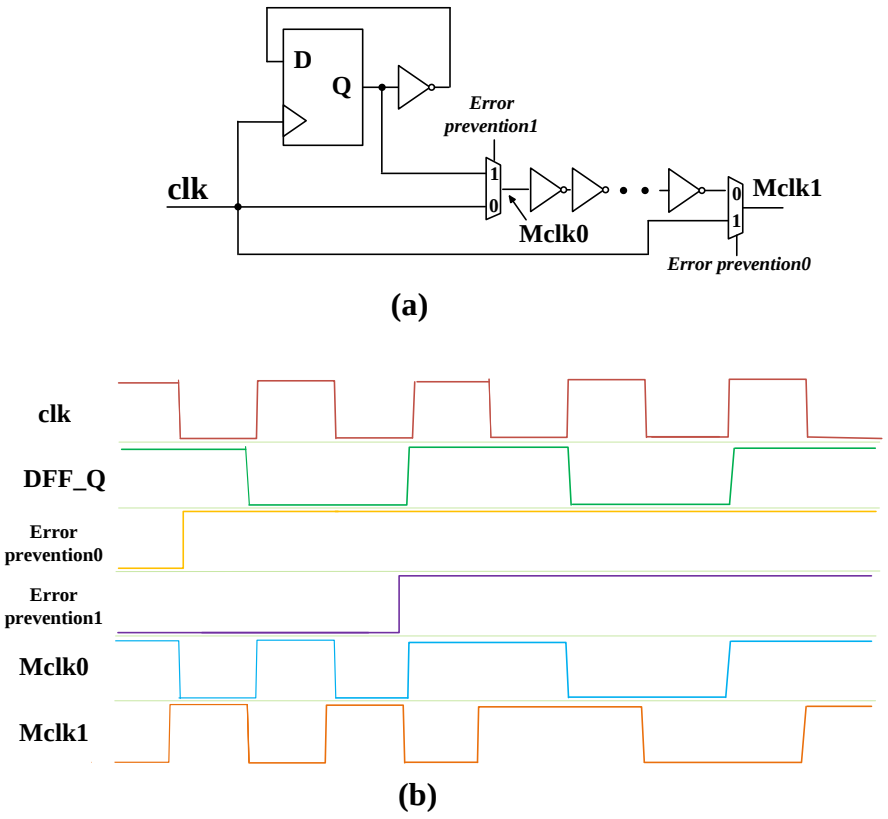



FIGURE 7.12: The circuit block and corresponding spice simulated waveforms of the proposed clock generator

TABLE 7.4: Comparison between the proposed error-masking and error-prevention techniques in terms of 8-bit DA computation.

| Approach | Error-masking | Error-prevention |
|---|---|---|
| Area overhead | 7.3% | 16% |
| Execution time | $8.5 * clk$ | $8 * clk$ |

EDAC: Error Detection and Correction

Compared with the error-masking technique, error-prevention approach consumes more hardware cost, as seen in Table 7.4. However, more circuit flexibility is brought by this approach and the execution time used for each sample processing is reduced, the delay resulted from the shifted-phase clock could be eliminated. In this case, if the output data from our proposed DSP accelerators is directly sent to MCU or other DSP block for further computing. We do not worry about the potential timing violation problem happens in the following clock cycles.

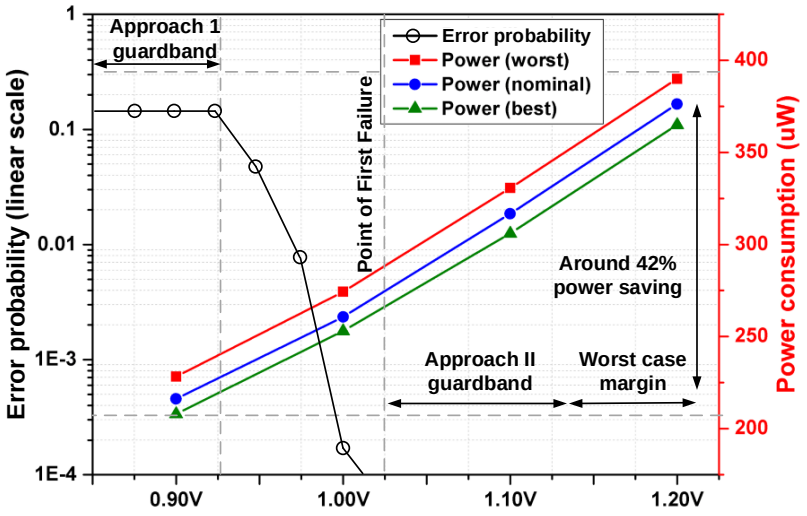



FIGURE 7.13: Power dissipation and the probability of error versus supply voltage with different process corners and temperature.

## 7.4 Summary

The cost-efficient error-resilient distributed arithmetic filter is presented in the section. Two novel approaches (error-masking and prevention) are proposed to mitigate the effects of PVT variations, which are achieved by modifying the path delay latency.

In terms of error-masking approach, extra timing slack is added for the paths of MSB group by time borrowing technique. In addition, the shifted-phase clock is adopted in the end-point register, thereby increasing the global guardband against variations. The error prevention design combines the shifted-phase clock and dynamic clock gating with the conventional error prediction circuit to realize a two-stage error-prevention scheme. In general, both the designs can acquire significant power saving up to 42% with different operating conditions when the worst case margin is considered. They also outperform other state-of-the-art error-resilient techniques in terms of cost-efficiency.

# Chapter 8

# Conclusions and Future Works

## 8.1 Conclusions

In this thesis, efficient methods to realize the low power and error-resilient signal processing are presented. Specifically, research aims are introduced in Chapter 1, as follows. (1) effective circuit-level power-saving methods at ultra-low voltage. (2) trade-off the balance between power, area and performance according to the design requirements of IoT-related devices. (3) cost-efficient timing error mitigation methods corresponding to the proposed power saving techniques.

The first two research aims are realized in Chapter 4, where the potential of the bit-serial circuit in terms of power-saving and variable-accuracy computing is investigated. In many IoT-related systems, the sensors operating at low sampling rates remain most of the time in the idle state with clock-gating. Since that the static energy consumption is more important than the active counterpart in this case. Furthermore, the bit-serial circuit can realize variable-accuracy at very low hardware overhead. Therefore, a novel bit-serial signed multiplier with row bypassing is proposed, which outperforms the standard bit-serial multipliers as well as the main types of parallel multipliers in terms of power and energy consumption in typical application scenarios at low voltage domains. Furthermore, by reducing the effective bit resolution of the input samples which are shifted into the proposed design, the overall energy consumption can be scaled down dynamically. A case study of FFT implementation based on the proposed bit-serial multiplier is also described. The simulation results show that when the effective word-length is reduced from 16 to 8 bits, the total energy consumption is reduced by up to 61%.

The third research aim is achieved in Chapter 3 and Chapter 5-7. In Chapter 3, the analyse of process variation effect on the adiabatic logic combined with the near-threshold operation is presented. In the case study of a 4-bit ripple adder, process variation could

cause at least 10.2% extra energy dissipation while the maximum frequency is reduced by 45.8% and approximately equal to 208 MHz. With the utilization of a bit-serial implementation, the energy per cycle can be further reduced while the robustness can be also significantly improved.

In Chapter 5, an error-prevention method by dynamic path isolation suitable with both conventional parallel and serial circuits is presented. Specifically, checkpoint flip-flops are placed in each critical path and dynamic isolate these paths once the error is detected in the bit serial sequential circuit. Compared with conventional clock-gating technique, our methods use smaller area and less performance penalty to achieve the same error-resilience.

In Chapter 6, it is initially demonstrated that the path delay in a serial datapath across bit positions is imbalanced. If a MSB-first computation is implemented, the critical path occurs in the LSBs. Then a variable latency approach to further create path delay imbalance is proposed. The main idea is to modify the critical path to be sensitive to the input signals and create more time slack for the MSB computation by adjustment of input data. The case study of an FIR filter presents that the proposed approach can maintain graceful performance without any timing margin and acquire significant power saving up to 46% with different operating conditions with worst-case margin.

In Chapter 7, two approaches to mitigate timing errors are proposed. The first one is achieved by time borrowing, which provides more time slack to MSB group. Accordingly, when time violations occur, less computation accuracy would be a lose. In additional, a shifted-phase clock is adopted on the end-point registers, thereby increasing the global guardband against variations. For the second design, it combines the shifted-phase clock and dynamic clock gating with the conventional error prediction circuit to realize a two-stage error-prevention scheme. The case study of an FIR filter shows the proposed approaches can achieve significant power saving up to 42% with different operating conditions when the worst case margin is considered. They also outperform other state-of-the-art error-resilient techniques in terms of cost-efficiency.

## 8.2 Future works

Based on the findings presented in this thesis, several directions for future research are described as follows:

### 8.2.1 Energy-harvesting-aware computing based on serial sequential circuit

A major challenge in the design of IoT sensors is the lack of reliable and continuous power sources. With advances in energy harvesting techniques, the application of self-powered sensors has become possible. However, it also poses a challenge to the current VLSI designers, how to keep circuits alive with the intermittent power supply as the supply can no longer be considered to be battery-like.

In this thesis, the potential of the serial sequential circuit in terms of power saving and energy-scalable operation are explored based on an FFT implementation. Future work includes a further demonstration on the system-level. First of all, implement the bit-serialization of the micro-controller and other hardware accelerators to minimise the area cost in a DSP chip. Then the performance is tested and optimised under the variable power supply source from the energy harvester. As known, in the conventional energy harvesting system, if the power is not large enough, the system would stop work and make a check-point. When the harvested power is restored to supply the load, the application execution can be continued. Since the main advantage of serialization is minimization of the leakage power which is a major power consumption during the system idle state. If the constant idle power can be always less than the minimum harvested power in any case, the powered circuit can naturally and smoothly meet the power requirements of energy harvesters. Accordingly, the system does not need to switch on/off or use the check-point mechanism frequently to deal with the variable power source. Besides, an energy harvesting estimation system is required, the accurate prediction of future changes can drive the DSP block to dynamically adjust the computation accuracy, thereby keeping alive in the intermittent environment.

### 8.2.2 Utilization of the error-resilient serial circuit in the reliable or DVAS system

In this thesis, two effective error-mitigation methods for serial computing are proposed and both of them are successfully demonstrated on an FIR implementation with very low hardware cost, which is introduced in Chapter 6 and Chapter 7, respectively. Based on the fact that timing error can be more easily and cost-efficiently prevented and masked in the serial datapath. The next step should be replacing the conventional hardware accelerators with serial counterparts in the reliable VLSI application. In addition, further explore the potential of serial design in error-resilience, such as improve the robustness against radiation-induced soft errors.

Since the path delay in a serial datapath across bit positions is imbalanced. If a MSB-first computation is implemented, the LSBs path start to fail first when the supply voltage is scaled down. This characteristic of serial computing makes itself compatible

with Dynamic Voltage Accuracy Scaling (DVAS) technique. Therefore, another of our future directions is to propose the generalization of the DVAS concept to the serial circuit and demonstrate its cost-efficiency compared with parallel counterparts in realizing approximate computing. In order to clearly present the advantages of DAVS based on the serial system, applications such as JPEG decoder would be considered.

# Bibliography


[1] Scott E Thompson and Srivatsan Parthasarathy. Moore's law: the future of si microelectronics. *Materials today*, 9(6):20–25, 2006.

[2] Feng Xia, Laurence T Yang, Lizhe Wang, and Alexey Vinel. Internet of things. *International Journal of Communication Systems*, 25(9):1101–1102, 2012.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[4] Massimo Alioto. *Enabling the Internet of Things: From Integrated Circuits to Integrated Systems*. Springer, 2017.

[5] Jan M Rabaey and Massoud Pedram. *Low power design methodologies*, volume 336. Springer Science & Business Media, 2012.

[6] Abdellatif Bellaouar and Mohamed Elmasry. *Low-power digital VLSI design: circuits and systems*. Springer Science & Business Media, 2012.

[7] Massimo Alioto. Ultra-low power vlsi circuit design demystified and explained: A tutorial. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 59(1):3–29, 2012.

[8] Alice Wang, Benton H Calhoun, and Anantha P Chandrakasan. *Sub-threshold design for ultra low-power systems*, volume 95. Springer, 2006.

[9] Dejan Markovic, Cheng C Wang, Louis P Alarcon, Tsung-Te Liu, and Jan M Rabaey. Ultralow-power design in near-threshold region. *Proceedings of the IEEE*, 98(2):237–252, 2010.

[10] Cristian Constantinescu. Trends and challenges in vlsi circuit reliability. *IEEE micro*, (4):14–19, 2003.

[11] Ronald G Dreslinski, Michael Wieckowski, David Blaauw, Dennis Sylvester, and Trevor Mudge. Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits. *Proceedings of the IEEE*, 98(2):253–266, 2010.

[12] Victor I Starosel'skii. Adiabatic logic circuits: A review. *Russian Microelectronics*, 31(1):37–58, 2001.

[13] Haiyan Ni and Jianping Hu. Near-threshold sequential circuits using improved clocked adiabatic logic in 45nm cmos processes. In *Circuits and Systems (MWSCAS), 2011 IEEE 54th International Midwest Symposium on*, pages 1–4. IEEE, 2011.

[14] Yangbo Wu and Jianping Hu. Near-threshold computing of clocked adiabatic logic with complementary pass-transistor logic circuits. *Journal of Low Power Electronics*, 7(3):393–402, 2011.

[15] Jianping Hu and Xiaoying Yu. Near-threshold adiabatic flip-flops based on pal-2n circuits in nanometer cmos processes. In *Circuits, Communications and System (PACCS), 2010 Second Pacific-Asia Conference on*, volume 1, pages 446–449. IEEE, 2010.

[16] Peter B Denyer and David Renshaw. *VLSI signal processing; a bit-serial approach*. Addison-Wesley Longman Publishing Co., Inc., 1985.

[17] David Harris and Sarah Harris. *Digital design and computer architecture*. Morgan Kaufmann, 2010.

[18] Rajeevan Amirtharajah, Jamie Collier, Jeff Siebert, Bicki Zhou, and Anantha Chandrakasan. Dsps for energy harvesting sensors: applications and architectures. *IEEE Pervasive Computing*, (3):72–79, 2005.

[19] Scott Hanson, Bo Zhai, Kerry Bernstein, David Blaauw, Andres Bryant, Leland Chang, Koushik K Das, Wilfried Haensch, Edward J Nowak, and Dinnis M Sylvester. Ultralow-voltage, minimum-energy cmos. *IBM Journal of Research and Development*, 50(4.5):469–490, 2006.

[20] Himanshu Kaul, Mark Anders, Steven Hsu, Amit Agarwal, Ram Krishnamurthy, and Shekhar Borkar. Near-threshold voltage (ntv) design: opportunities and challenges. In *Proceedings of the 49th Annual Design Automation Conference*, pages 1153–1158. ACM, 2012.

[21] Shailendra Jain, Surhud Khare, Satish Yada, V Ambili, Praveen Salihundam, Shiva Ramani, Sriram Muthukumar, M Srinivasan, Arun Kumar, Shasi Kumar Gb, et al. A 280mv-to-1.2 v wide-operating-range ia-32 processor in 32nm cmos. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 66–68. IEEE, 2012.

[22] Sven Luetkemeier, Thorsten Jungeblut, Mario Porrmann, and Ulrich Rueckert. A 200mv 32b subthreshold processor with adaptive supply voltage control. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 484–486. IEEE, 2012.

[23] Yu Pu, Jose Pineda De Gyvez, Henk Corporaal, and Yajun Ha. An ultra-low-energy multi-standard jpeg co-processor in 65 nm cmos with sub/near threshold supply voltage. *IEEE Journal of Solid-State Circuits*, 45(3):668–680, 2010.

[24] Bo Zhai, Sanjay Pant, Leyla Nazhandali, Scott Hanson, Javin Olson, Anna Reeves, Michael Minuth, Ryan Helfand, Todd Austin, Dennis Sylvester, et al. Energy-efficient subthreshold processor design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 17(8):1127–1137, 2009.

[25] Mingoo Seok, Scott Hanson, Yu-Shiang Lin, Zhiyoong Foo, Daeyeon Kim, Yoonmyung Lee, Nurrachman Liu, Dennis Sylvester, and David Blaauw. The phoenix processor: A 30pw platform for sensor applications. In *VLSI Circuits, 2008 IEEE Symposium on*, pages 188–189. IEEE, 2008.

[26] Jos Hulzink, Mario Konijnenburg, Maryam Ashouei, Arjan Breeschoten, Torfinn Berset, Jos Huisken, Jan Stuyt, Harmke de Groot, Francisco Barat, Johan David, et al. An ultra low energy biomedical signal processing system operating at near-threshold. *Biomedical Circuits and Systems, IEEE Transactions on*, 5(6):546–554, 2011.

[27] Mingoo Seok, Gregory Chen, Scott Hanson, Michael Wieckowski, David Blaauw, and Dennis Sylvester. Cas-fest 2010: Mitigating variability in near-threshold computing. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 1(1):42–49, 2011.

[28] Benton H Calhoun and David Brooks. Can subthreshold and near-threshold circuits go mainstream? *IEEE Micro*, 30(4):80–85, 2010.

[29] William C Athas, Lars J Svensson, Jefferey G Koller, Nestoras Tzartzanis, and E Ying-Chin Chou. Low-power digital systems based on adiabatic-switching principles. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2(4):398–407, 1994.

[30] A Vetuli, SD Pascoli, and LM Reyneri. Positive feedback in adiabatic logic. *Electronics Letters*, 32(20):1867–1869, 1996.

[31] Dragan Maksimovic, Vojin G Oklobdzija, Borivoje Nikolic, and K Wayne Current. Clocked cmos adiabatic logic with integrated single-phase power-clock supply. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(4):460–463, 2000.

[32] Ettore Amirante, Jurgen Fischer, Markus Lang, Agnese Bargagli-Stoffi, Jörg Berthold, Christoph Heer, and Doris Schmitt-Landsiedel. An ultra low-power adiabatic adder embedded in a standard 0.13/spl mu/m cmos environment. In *Solid-State Circuits Conference, 2003. ESSCIRC'03. Proceedings of the 29th European*, pages 599–602. IEEE, 2003.

[33] Philip Teichmann. *Adiabatic logic: future trend and system level perspective*, volume 34. Springer Science & Business Media, 2011.

[34] Nan Liao, XiaoXin Cui, Kai Liao, KaiSheng Ma, Di Wu, Wei Wei, Rui Li, and DunShan Yu. Low power adiabatic logic based on finfets. *Science China Information Sciences*, 57(2):1–13, 2014.

[35] Yong Moon and Deog-Kyoon Jeong. An efficient charge recovery logic circuit. *Solid-State Circuits, IEEE Journal of*, 31(4):514–522, 1996.

[36] F Liu and KT Lau. Improved structure for efficient charge recovery logic. *Electronics Letters*, 34(18):1731–1732, 1998.

[37] Anish Muttreja, Niket Agarwal, and Niraj K Jha. Cmos logic design with independent-gate finfets. In *Computer Design, 2007. ICCD 2007. 25th International Conference on*, pages 560–567. IEEE, 2007.

[38] Ettore Amirante, Agnese Bargagli-Stoffi, Jürgen Fischer, Giuseppe Iannaccone, and Doris Schmitt-Landsiedel. Variations of the power dissipation in adiabatic logic gates. In *Proceedings of the 11th International Workshop on Power And Timing Modeling, Optimization and Simulation, PATMOS*, volume 1, pages 9–1, 2001.

[39] Hanho Lee and Gerald E Sobelman. Fpga-based fir filters using digit-serial arithmetic. *architecture*, 1:5, 1997.

[40] Xin Cai. *A Serial Bitstream Processor for Smart Sensor Systems.* PhD thesis, 2010.

[41] Richard Hartley and Peter Corbett. Digit-serial processing techniques. *IEEE Transactions on Circuits and Systems*, 37(6):707–719, 1990.

[42] Keshab K Parhi. A systematic approach for design of digit-serial signal processing architectures. *IEEE Transactions on Circuits and Systems*, 38(4):358–375, 1991.

[43] Lang Yang and Thomas W Chen. A low power 64-point bit-serial fft engine for implantable biomedical applications. In *Digital System Design (DSD), 2015 Euromicro Conference on*, pages 383–389. IEEE, 2015.

[44] Levent Aksoy, Cristiano Lazzari, Eduardo Costa, Paulo Flores, and José Monteiro. Design of digit-serial fir filters: Algorithms, architectures, and a cad tool. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(3):498–511, 2013.

[45] K Jebin Roy and R Ramya. Low-power and low-area adaptive fir filter based on distributed arithmetic and lms algorithm. *International Journal of Scientific and Research Publications*, 4(3), 2014.

[46] Andrew J Miller. Fpga implemented bit-serial multiplier and infinite impulse response, August 20 2002. US Patent 6,438,570.

[47] Ahmad Darabiha, Anthony Chan Carusone, and Frank R Kschischang. A bit-serial approximate min-sum ldpc decoder and fpga implementation. In *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, pages 4–pp. IEEE, 2006.

[48] Sudhanshu Khanna and Benton H Calhoun. Serial sub-threshold circuits for ultra-low-power systems. In *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*, pages 27–32. ACM, 2009.

[49] Richard Lyon. Two's complement pipeline multipliers. *IEEE Transactions on Communications*, 24(4):418–425, 1976.

[50] R Gnanasekaran. On a bit-serial input and bit-serial output multiplier. *IEEE Transactions on Computers*, 32(9):878–880, 1983.

[51] Luigi Dadda. Fast multipliers for two's-complement numbers in serial form. In *Computer Arithmetic (ARITH), 1985 IEEE 7th Symposium on*, pages 57–63. IEEE, 1985.

[52] Paolo Ienne and Marc A. Viredaz. Bit-serial multipliers and squarers. *IEEE Transactions on Computers*, 43(12):1445–1450, 1994.

[53] Poras T Balsara and David T Harper. Understanding vlsi bit serial multipliers. *IEEE Transactions on Education*, 39(1):19–28, 1996.

[54] MK Ibrahim. Radix-2n multiplier structures: A structured design methodology. *IEE Proceedings E (Computers and Digital Techniques)*, 140(4):185–190, 1993.

[55] Charles R Baugh and Bruce A Wooley. A two's complement parallel array multiplication algorithm. *IEEE Transactions on Computers*, 22(12):1045–1047, 1973.

[56] Xin Cai and Martin Brooke. A compact cpu architecture for sensor signal processing. In *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, pages 4–pp. IEEE, 2006.

[57] Matthew Tomei, Henry Duwe, Nam Sung Kim, and Rakesh Kumar. Bit serializing a microprocessor for ultra-low-power. In *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, pages 200–205. ACM, 2016.

[58] Biswajit Mishra and Bashir M Al-Hashimi. Subthreshold fir filter architecture for ultra low power applications. In *International Workshop on Power and Timing Modeling, Optimization and Simulation*, pages 1–10. Springer, 2008.

[59] Alicia M Klinefelter, Yanqing Zhang, Brian Otis, and Benton H Calhoun. A programmable 34 nw/channel sub-threshold signal band power extractor on a body sensor node soc. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 59(12):937–941, 2012.

[60] Muhammad Umar Karim Khan and Chong Min Kyung. Energy reduction of ultra-low voltage vlsi circuits by digit-serial architectures. In *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, pages 3012–3017. IEEE, 2013.

[61] Stanley A White. Applications of distributed arithmetic to digital signal processing: A tutorial review. *IEEE Assp Magazine*, 6(3):4–19, 1989.

[62] Rajeevan Amirtharajah and Anantha P Chandrakasan. A micropower programmable dsp using approximate signal processing based on distributed arithmetic. *IEEE Journal of Solid-State Circuits*, 39(2):337–347, 2004.

[63] Jie Han and Michael Orshansky. Approximate computing: An emerging paradigm for energy-efficient design. In *2013 18th IEEE European Test Symposium (ETS)*, pages 1–6. IEEE, 2013.

[64] Vaibhav Gupta, Debabrata Mohapatra, Anand Raghunathan, and Kaushik Roy. Low-power digital signal processing using approximate adders. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(1):124–137, 2013.

[65] Yaman Umuroglu, Lahiru Rasnayake, and Magnus Själander. Bismo: A scalable bit-serial matrix multiplication overlay for reconfigurable computing. In *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, pages 307–3077. IEEE, 2018.

[66] Dan Ernst, Shidhartha Das, Seokwoo Lee, David Blaauw, Todd Austin, Trevor Mudge, Nam Sung Kim, and Krisztián Flautner. Razor: circuit-level correction of timing errors for low-power operation. *IEEE Micro*, 24(6):10–20, 2004.

[67] Manuel de la Guia Solaz and Richard Conway. Razor based programmable truncated multiply and accumulate, energy-reduction for efficient digital signal processing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23(1):189–193, 2015.

[68] Paul N Whatmough, Izzat Darwazeh, David M Bull, Shidhartha Das, and Danny Kershaw. A robust fir filter with in situ error detection. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 4185–4188. IEEE, 2010.

[69] Paul Whatmough, Shidhartha Das, David Bull, and Izzat Darwazeh. Error-resilient low-power dsp via path-delay shaping. In *Proceedings of the 48th Design Automation Conference*, pages 1008–1013. ACM, 2011.

[70] Toshinori Sato and Yuji Kunitake. A simple flip-flop circuit for typical-case designs for dfm. In *Quality Electronic Design, 2007. ISQED'07. 8th International Symposium on*, pages 539–544. IEEE, 2007.

[71] Weiwei Shan, Xinning Liu, Minyi Lu, Liang Wan, and Jun Yang. A low-overhead timing monitoring technique for variation-tolerant near-threshold digital integrated circuits. *IEEE Access*, 6:138–145, 2018.

[72] Jun Zhou, Xin Liu, Yat-Hei Lam, Chao Wang, Kah-Hyong Chang, Jingjing Lan, and Minkyu Je. Hepp: A new in-situ timing-error prediction and prevention technique for variation-tolerant ultra-low-voltage designs. In *Solid-State Circuits Conference (A-SSCC), 2013 IEEE Asian*, pages 129–132. IEEE, 2013.

[73] Youhua Shi, Hiroaki Igarashi, Nozomu Togawa, and Masao Yanagisawa. Suspicious timing error prediction with in-cycle clock gating. In *Quality Electronic Design (ISQED), 2013 14th International Symposium on*, pages 335–340. IEEE, 2013.

[74] Swaroop Ghosh, Swarup Bhunia, and Kaushik Roy. Crista: A new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(11):1947–1956, 2007.

[75] Mihir Choudhury, Vikas Chandra, Kartik Mohanram, and Robert Aitken. Timber: Time borrowing and error relaying for online timing error resilience. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010*, pages 1554–1559. IEEE, 2010.

[76] Kwanyeob Chae, Chang-Ho Lee, and Saibal Mukhopadhyay. Timing error prevention using elastic clocking. In *IC Design & Technology (ICICDT), 2011 IEEE International Conference on*, pages 1–4. IEEE, 2011.

[77] Masanori Kurimoto, Hiroaki Suzuki, Rei Akiyama, Tadao Yamanaka, Haruyuki Ohkuma, Hidehiro Takata, and Hirofumi Shinohara. Phase-adjustable error detection flip-flops with 2-stage hold driven optimization and slack based grouping scheme for dynamic voltage scaling. In *Proceedings of the 45th annual Design Automation Conference*, pages 884–889. ACM, 2008.

[78] Ulya R Karpuzcu, Nam Sung Kim, and Josep Torrellas. Coping with parametric variation at near-threshold voltages. *Micro, IEEE*, 33(4):6–14, 2013.

[79] Arizona State Univ. PTM. Temp, Mar, 2010. [Online]. Available: http://ptm.asu.edu.

[80] Wei Zhao, Frank Liu, Kanak Agarwal, Dhruva Acharyya, Sani R Nassif, Kevin J Nowka, and Yu Cao. Rigorous extraction of process variations for 65-nm cmos design. *Semiconductor Manufacturing, IEEE Transactions on*, 22(1):196–203, 2009.

[81] Rangharajan Venkatesan, Amit Agarwal, Kaushik Roy, and Anand Raghunathan. Macaco: Modeling and analysis of circuits for approximate computing. In *Proceedings of the International Conference on Computer-Aided Design*, pages 667–673. IEEE Press, 2011.

[82] Parag Kulkarni, Puneet Gupta, and Milos Ercegovac. Trading accuracy for power with an underdesigned multiplier architecture. In *VLSI Design (VLSI Design), 2011 24th International Conference on*, pages 346–351. IEEE, 2011.

[83] Liping Guo, Mackenzie Scott, and Rajeevan Amirtharajah. An energy scalable computational array for sensor signal processing. In *Custom Integrated Circuits Conference, 2006. CICC'06. IEEE*, pages 317–320. IEEE, 2006.

[84] Jun-ni Ohban, Vasily G Moshnyaga, and Koji Inoue. Multiplier energy reduction through bypassing of partial products. In *Circuits and Systems, 2002. APCCAS'02. 2002 Asia-Pacific Conference on*, volume 2, pages 13–17. IEEE, 2002.

[85] Ming-Chen Wen, Sying-Jyan Wang, and Yen-Nan Lin. Low power parallel multiplier with column bypassing. In *2005 IEEE International Symposium on Circuits and Systems*, pages 1638–1641. IEEE, 2005.

[86] Jin-Tai Yan and Zhi-Wei Chen. Low-power multiplier design with row and column bypassing. In *SOC Conference, 2009. SOCC 2009. IEEE International*, pages 227–230. IEEE, 2009.

[87] Wen-Chang Yeh and Chein-Wei Jen. High-speed booth encoded parallel multiplier design. *IEEE transactions on computers*, 49(7):692–701, 2000.

[88] Joseph T Scanlon and W Kent Fuchs. High performance bit-serial multiplication. *Proc. IEEE ICCD'86*, pages 114–117, 1986.

[89] Alice Wang and Anantha Chandrakasan. A 180-mv subthreshold fft processor using a minimum energy design methodology. *Solid-State Circuits, IEEE Journal of*, 40(1):310–319, 2005.

[90] Bo Yuan, Yanzhi Wang, and Zhongfeng Wang. Area-efficient scaling-free dft/fft design using stochastic computing. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 63(12):1131–1135, 2016.

[91] M Jiang, B Yang, R Huang, TY Zhang, and YY Wang. Multiplierless fast fourier transform architecture. *Electronics Letters*, 43(3):191–192, 2007.

[92] Johan Melander, Torbjörn Widhe, Peter Sandberg, Kent Palmkvist, Mark Vesterbacka, and Lars Wanhammar. Implementation of a bit-serial fft processor with a hierarchical control structure. In *Proc. European Conf. on Circuit Theory and Design, ECCTD 95, Istanbul, Turkey*, pages 423–426, 1995.

[93] Shinnosuke Yoshida, Youhua Shi, Masao Yanagisawa, and Nozomu Togawa. An effective suspicious timing-error prediction circuit insertion algorithm minimizing area overhead. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98(7):1406–1418, 2015.

[94] Liping Guo, Mackenzie Scott, and Rajeevan Amirtharajah. An energy scalable functional unit for sensor signal processing. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–73. IEEE, 2007.

[95] Yue Lu and Tom J Kazmierski. An ultra-low-power variable-accuracy bit-serial fft butterfly processing element for iot sensors. In *Circuits and Systems (APCCAS), 2016 IEEE Asia Pacific Conference on*, pages 13–16. IEEE, 2016.

[96] Lu Yue and Tom Kazmierski. Variable-accuracy bit-serial multiplication with row bypassing for ultra low power. In *2017 IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP and International Symposium of System-on-Chip (SoC)*, pages 1–6. IEEE, 2017.

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

[98] Seongjong Kim and Mingoo Seok. Variation-tolerant, ultra-low-voltage microprocessor with a low-overhead, within-a-cycle in-situ timing-error detection and correction technique. *IEEE Journal of Solid-State Circuits*, 50(6):1478–1490, 2015.

[99] Ricky Liou and Matthew Beach. irazor: a low overhead error detection and correction scheme to improve processor performance.

[100] Sai Zhang and Naresh R Shanbhag. Embedded algorithmic noise-tolerance for signal processing and machine learning systems via data path decomposition. *IEEE Transactions on Signal Processing*, 64(13):3338–3350, 2016.

[101] Paul N Whatmough, Shidhartha Das, David M Bull, and Izzat Darwazeh. Circuit-level timing error tolerance for low-power dsp filters and transforms. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(6):989–999, 2013.