Adiabatic Circuits for Power-Constrained Cryptographic Computations

Raghav, H.

Adiabatic Circuits for Power-Constrained Cryptographic Computations

University of Westminster

Himadri Singh Raghav

A thesis submitted in partial fulfilment of the requirements of the University of Westminster for the degree of Doctor of Philosophy

January 2018

Author's Declaration

I hereby declare that the research work presented in this thesis has been completed by the author. Where other sources of information have been used, they have been quoted.

No part of this thesis has been submitted in support of an application for any other degree.

Himadri Singh Raghav

To my family

For their understandings, love, and support

Acknowledgements

I am indebted to my advisor Professor Izzet Kale for giving me the opportunity to carry out this work and for his great guidance and support throughout the course of my PhD. He is the one who always motivated me and helped me to get going when I felt stuck. He also gave me the opportunity to present my research in the conferences for which I am truly grateful.

I would also like to express my sincere gratitude to my second supervisor Dr Viv Bartlett for his guidance and valuable suggestions.

I would like to express my very great appreciation to Professor Alex Yakovlev for his valuable and constructive suggestions on the work presented in this thesis.

My grateful thanks are also extended to Dr. Adem Coskun for his valuable suggestions on the work of this thesis.

I would also like to thank,

Dr Andrej Tarczynaki, for his valuable suggestions and serving on my PhD committee.

Prof. Tajalli Keshavarz, for supporting and encouraging me in my toughest time.

I would like to thank my family for their unconditional support and encouragement throughout my research. Without them, I would have never reached this far.

My final and the most heartfelt acknowledgement go to Mr Sachin Maheshwari, who supported and encouraged me in the worst of my time.

This work was supported by Cavendish Research Scholarship whose contribution is gratefully acknowledged.

Abstract

This thesis tackles the need for ultra-low power operation in power-constrained cryptographic computations. An example of such an application could be smartcards. One of the techniques which has proven to have the potential of rendering ultra-low power operation is ‘Adiabatic Logic Technique’. However, the adiabatic circuits has associated challenges due to high energy dissipation of the Power-Clock Generator (PCG) and complexity of the multi-phase power-clocking scheme. Energy efficiency of the adiabatic system is often degraded due to the high energy dissipation of the PCG. In this thesis, n-step charging strategy using tank capacitors is considered for the power-clock generation and several design rules and trade-offs between the circuit complexity and energy efficiency of the PCG using n-step charging circuits have been proposed.

Since pipelining is inherent in adiabatic logic design, careful selection of architecture is essential, as otherwise overhead in terms of area and energy due to synchronization buffers is induced specifically, in the case of adiabatic designs using 4-phase power-clocking scheme. Several architectures for the Montgomery multiplier using adiabatic logic technique are implemented and compared. An architecture which constitutes an appropriate trade-off between energy efficiency and throughput is proposed along with its methodology. Also, a strategy to reduce the overhead due to synchronization buffers is proposed. A modification in the Montgomery multiplication algorithm is proposed.

Furthermore, a problem due to the application of power-clock gating in cascade stages of adiabatic logic is identified. The problem degrades the energy savings that would otherwise be obtained by the application of power-clock gating. A solution to this problem is proposed.

Cryptographic implementations also present an obvious target for Power Analysis Attacks (PAA). There are several existing secure adiabatic logic designs which are proposed as a countermeasure against PAA.

Shortcomings of the existing logic designs are identified, and two novel secure adiabatic logic designs are proposed as the countermeasures against PAA and improvement over the existing logic designs.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AL	Adiabatic Loss
CAL	Clocked Adiabatic Logic
CLA	Carry Look Ahead
CMOS	Complementary Metal Oxide Semiconductor
CPAL	Complementary Pass-transistor Adiabatic Logic
CS	Charge Sharing
CSA	Carry Save Adder
CSSAL	Charge-Sharing Symmetric Adiabatic Logic
CTT	Combined Tank Capacitance
DPA	Differential Power Analysis attack
DRSL	Dual-rail Random Switching Logic
EACRL	Efficient Adiabatic Charge Recovery Logic
ECC	Elliptic Curve Cryptography
EE-SPFAL	Energy Efficient Secure Positive Feedback Adiabatic Logic
FA	Full Adder
FSM	Finite State Machine
HA	Half Adder
IECRL	Improved Efficient Charge Recovery Logic
LL	Leakage Losses
LSB	Least Significant Bit

MADU	Modulus Addition Decision Unit
MDPL	Masked Dual-rail Pre-charge Logic
MM	Montgomery multiplier
MMM	Modified Montgomery multiplier
MSB	Most Significant Bit
NAL	Non-Adiabatic Loss
NED	Normalized Energy Deviation
NSD	Normalized Standard Deviation
PAA	Power Analysis Attacks
PC	Power-clock
PCG	Power-Clock Generator
PE	Processing Elements
PFAL	Positive Feedback Adiabatic Logic
PRU	Partial Residue Unit
IA1	Iterative approach Architecture 1
IA2	Iterative approach Architecture 2
IA3	Iterative approach Architecture 3
IA3_PG	Iterative approach Architecture 3 using Power Gating
IA3_PPG	Iterative approach Architecture 3 using Proposed Power Gating
ICMOS	Iterative approach architecture using CMOS
RSA	Rivest-Shamir-Adleman
SAA1	Systolic Array Architecture 1
SAA2	Systolic Array Architecture 2

SABL	Sense-Amplifier-Based Logic
SCA	Side Channel Attacks
SCC	Step Charging Circuit
SPGAL	Symmetric Pass Gate Adiabatic Logic
SQAL	Secure Quasi-Adiabatic Logic
SRAM	Static Random Access Memory
SyAL	Symmetric Adiabatic Logic
TDPL	Three-phase Dual-rail pre-charged logic
TG	Transmission Gate
TT	Typical-Typical
TSMC	Taiwan Semiconductor Manufacturing Company
WCS-QuAL	Without Charge Sharing Quasi-Adiabatic Logic
WDDL	Wave Dynamic Differential Logic

1. Introduction

1.1 Introduction

With an exceptional growth of smart cards in worldwide applications including electronic commerce and access control, the security and authentication of information have become a vitally important area of concern. Consequently, an important area of research in smart cards has been the development and implementation of energy efficient and secure cryptographic algorithms. Cryptography is the science of writing in secret code and is an ancient art predominantly been used by the military and governments for message confidentiality. When Julius Caesar sent messages to his generals, he didn't trust his messengers. So, in his message, he replaced every A with a D, every B with an E, and so on through the alphabets. Only someone who knew the "shift by 3" rule could decipher his messages. The art was transformed into a science by Shannon in 1948 [1]. This was arguably the beginning of modern cryptography, in which checking the integrity of the messages and authenticating the identities of the communicating parties have become as essential as ensuring message confidentiality. Cryptography is necessary when communicating over any untrusted medium, which includes just about *any* network, particularly the internet. It allows two people usually referred to as Alice and Bob, to communicate over an insecure channel in such a way that an adversary, usually referred to as Eve, cannot understand what is being communicated. Every day hundreds of thousands of people interact electronically, through e-mails, electronic commerce (business conducted over the Internet), or mobile phones. This perpetual increase of information transmitted electronically has led to an increased reliance on cryptography.

Within the context of any application-to-application communication, there are some specific security requirements, including:

- *Confidentiality:* The first and intuitive aim of cryptography is to provide the confidentiality; anyone trying to intercept an encrypted message must not be able to recover the original message, without having access to the ciphering key.
- *Integrity:* It is the next important feature which ensures the receiver that the received message is the original one and has not been altered in any way by an adversary.
- *Authentication:* This ensures the receiver that the message is really coming from the intended sender (The process of proving one's identity) who couldn't be mimicked by an adversary.
- *Identification:* This concept is close to the previous one. Here the aim is to directly authenticate the sender and not a message (the sender cannot deny having sent the message and also the receiver cannot deny having received the message). The person is generally authenticated with a secret that he or she possesses.

These features are provided by cryptographic algorithms. The field of cryptographic algorithms can broadly be divided into two types, symmetric and asymmetric, which have distinctly different properties. Symmetric or secret-key algorithms require two parties to share some secret piece of information (the key) that is then used to encrypt/decrypt messages between them. Encryption is the process of transforming message usually referred as *plain text* to the unreadable information referred as *ciphertext*. The ciphertext can only be converted to the plain text by using the key. The existence of a shared piece of secret information enables secret-key algorithms to be computationally efficient. Figure 1.1 depicts the symmetric encryption/decryption system.

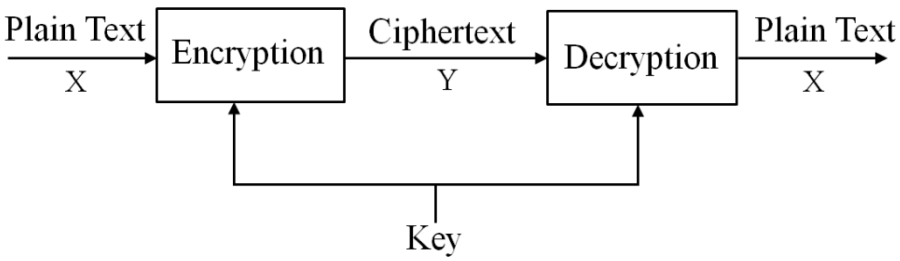


Figure 1.1: Symmetric encryption/decryption system

Asymmetric or public-key algorithms, on the other hand, rely on the presumed existence of hard number-theoretic problems that enable two keys to be generated; public (encryption) and private (decryption). Figure 1.2 depicts the asymmetric encryption/decryption system, where the sender uses the receiver's public key to encrypt while the receiver uses his own

private key to decrypt. Public-keys are stored in the open so that anyone can encrypt a message.

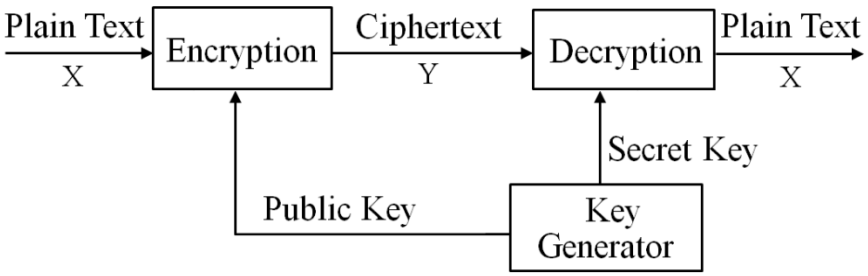


Figure 1.2: Asymmetric encryption/decryption system

However, because of the number-theoretic properties of the algorithms used, only the intended recipient who generated the public-private key pair can decode the message correctly. Hence, no secret needs to be shared by the communicating parties. Unfortunately, the underlying mathematics which enables this asymmetry requires a great deal more computation than symmetric-key algorithms. Thus, public-key cryptographic algorithms are computationally expensive and have a high energy cost. The two widely used public-key cryptographic schemes in smart card applications are the Rivest-Shamir-Adleman (RSA) [2] and Elliptic Curve Cryptography (ECC) [3]. RSA makes use of very large word length operands (e.g. 1024-bits, 2048-bits) to achieve the security required by many applications. Because key lengths correspond to the level of security; longer the key length is, stronger the security and higher the energy dissipation. ECC, on the other hand, uses relatively short operand length (e.g. 163, 224) compared to RSA. However, ECC is still considered as a computationally expensive algorithm.

Modular multiplication operation is common to both RSA and ECC. Because of its speed and efficiency when handling large word length operands, the modular multiplication is commonly implemented using the Montgomery algorithm [4]. This algorithm uses simple shift and add operations, avoiding the use of costly trial division operations otherwise required. However, such large operand lengths have associated costs such as computational complexity, hardware cost, large area as well as high energy dissipation. The latter makes the use of RSA/ECC in devices with limited power (such as smartcards) difficult. Such a problem is one of the motivations for this research.

An approach that has the potential to attain low power operation is the adiabatic circuit technique. Adiabatic circuits use a slowing changing power supply/clock called as power-

clock. A slowly changing power-clock allows approximately constant current charging and discharging [5], [6]. The power-clock also makes possible the recycling of charge, enabling energy to be recovered, thus reducing the overall energy drawn directly from the power supply. Therefore, adiabatic logic makes an attractive implementation method for cryptographic algorithms in smart cards.

Another concern for cryptographic implementations is that they present an obvious target for so-called Side Channel Attacks (SCA). These attacks depend on the relationship between information emitted (leaked) through the side-channels of the cryptographic implementation and the secret data processed. SCA such as Power Analysis Attacks (PAA) rely on monitoring of power supply currents/power fluctuations of cryptographic circuits during the execution of critical operations such as encryption/decryption; by analyzing these fluctuations, sensitive data, such as the secret key used in the encryption algorithm can be deduced. Differential Power Analysis (DPA) attack [7] is the widely deployed PAA. DPA attacks use statistical methods and digital processing techniques on a large number of monitored power signals. Such methods reduce noise and enhance the signals making it easier to distinguish between a zero and a one.

1.2 Motivation for this Thesis

Adiabatic logic has been proven to have the potential of rendering ultra-low-power operation but, has associated challenges arising due to the complexity of multiphase power-clocking scheme and high energy dissipation of the power-clock generator.
In this thesis, the aim is to tackle the need for ultra-low power operation in power-limited smartcards. In order to render ultra-low power operation, it is mandatory to jointly concentrate on energy efficient power-clock generation and on finding the architecture suitable for energy efficient adiabatic implementation for cryptographic algorithms. The energy efficiency of the complete adiabatic system is often degraded due to the high energy dissipation of the power-clock generator. Thus, one of the thesis objectives is to investigate adiabatic logic and its power-clock generation so that energy efficiency achievable by the complete adiabatic system and its associated cost can be known.

Another goal of the thesis is to make the cryptographic implementations secure. This includes identifying the shortcomings of the already existing secure adiabatic logic designs and developing new secure logic designs as a solution to the shortcomings and

countermeasure against PAA.

1.3 Original Contributions

The main contributions resulting from this research can be summarized as below:

1. For the implementation of energy efficient Power-Clock Generators (PCG) using n-step charging strategy (using tank-capacitors), several design rules and trade-offs between the circuit complexity and the energy efficiency of the PCG using n-step charging circuit are proposed.
    a. For the PCG using 2-step charging circuit, it was proposed that tank capacitor to load capacitor ratio of 10, keeping the width of the transistors minimum can be chosen as a convenient 'rule-of-thumb' in practical designs. This work is described in Chapter 3, is published in the proceedings of PATMOS 2016 [HSR2] and will be submitted to [HSR14].
    b. For PCG using 3, 4, 5, 6, 7, and 8-step charging circuits, it was established that combined tank capacitor to load capacitor ratio of 10 is appropriate. Also, PCG using 4-step charging circuit constitute appropriate trade-offs between circuit complexity and energy efficiency. This work is described in Chapter 3 and is published in the proceedings of PRIME 2016 [HSR1].
    c. When the energy dissipation of the Finite State Machine (FSM) controller was considered, it was proposed that 4-phase PCG using 3 and 4-step charging circuits constitute an appropriate trade-off between circuit complexity and energy efficiency. This work is described in Chapter 4 and is submitted to [HSR6].

2. Adiabatic implementation of cryptographic systems.
    a. Energy efficient and scalable systolic array architecture for the 8-bit Montgomery multiplier is proposed. Also, a strategy to reduce the overhead due to the synchronization buffers is proposed. Using the proposed strategy, the average energy per computation was reduced by approximately 21% and the throughput was improved by 4-power-clock cycles. Most importantly, the number of synchronization buffers was reduced by approximately 50%. This work is described in Chapter 5 and is submitted to [HSR7] and [HSR10].
    b. Scalable, area and energy efficient architectures based on iterative scheme for the 8-bit Montgomery multiplier using single, 2, and 3 adder stages in the datapath unit were proposed. These architectures outperformed the proposed systolic array

architectures in terms of area and energy efficiency. An optimum number of stages in the data path unit was identified. Montgomery multiplication algorithm was also modified. Due to the modification, latency was improved by one power-clock cycle and the area of one bit-slice was saved in the datapath unit. A methodology for designing architectures based on iterative scheme using 4-phase power-clocking scheme was proposed. This work is described in Chapter 5 and will be submitted to [HSR12].

c. A problem due to the application of power-clock gating in cascade stages of adiabatic gates was discovered. This problem degrades the energy savings that would otherwise be obtained by the application of power-clock gating. A solution to this problem is proposed and an improvement of about 13% and 34% in the energy dissipation was obtained in comparison to the Montgomery multiplier architecture with power-clock gating (without solution) and Montgomery multiplier architecture without power-clock gating. This work is described in Chapter 5 and will be submitted to [HSR13].

d. It was discovered that, for a large adiabatic core (load), the energy dissipation of the Finite State Machine (FSM) controller becomes negligible compared to the combined energy dissipation of the step charging circuit and adiabatic core. This work is described in Chapter 5 and is a part of [HSR6].

3. Two novel Power Analysis Attack (PAA) resilient adiabatic logic designs were proposed.

   a. Shortcomings of the existing secure adiabatic logic designs were identified. A novel PAA resilient adiabatic logic called Without Charge Sharing Quasi-Adiabatic Logic, WCS-QuAL was proposed as an improvement over the existing secure adiabatic logic designs. The proposed logic outperforms the existing logic designs in terms of resilience against PAA, Process corner variations, power supply scaling, and energy performance. This work is described in Chapter 6 and is published in the proceedings of ECCTD 2017 [HSR3], PATMOS 2017 [HSR4] and in Microelectronics journal (Elsevier) [HSR5].

   b. The condition when the two output nodes of WCS-QuAL remain unbalanced was identified. This condition results in variations in the negative current peaks. As a solution, another novel PAA resilient adiabatic logic was proposed. The proposed logic outperforms the existing logic designs in terms of resilience against PAA, power supply scaling, and current variations. This work is described in Chapter 7

and is published in Integration, The VLSI Journal (Elsevier) [HSR8], and in the Proceedings of PATMOS 2018 [HSR9] and will be submitted to [HSR11].

1.4 Author's Publications

1) **Himadri Singh Raghav**, Viv A. Bartlett and Izzet Kale, "Investigation of Stepwise Charging Circuits for Power-Clock Generation in Adiabatic Logic", in *Proc.* PRIME, Lisbon, Portugal, 2016, pp. 1-4. [HSR1]

2) **Himadri Singh Raghav**, Viv A. Bartlett and Izzet Kale, "Energy Efficiency of 2 Step Power-clocks for Adiabatic Logic", in 26th *Proc.* PATMOS, Bremen, Germany, 2016, pp. 176-182. [HSR2]

3) **Himadri Singh Raghav**, Viv A. Bartlett and Izzet Kale, "Novel Power Analysis Attack Resilient Adiabatic Logic", in 23rd Proc. ECCTD, Catania, Italy, 2017, pp. 1-4. [HSR3]

4) **Himadri Singh Raghav**, Viv A. Bartlett and Izzet Kale, "Robustness of Power Analysis Attack Resilient Adiabatic logic: WCS-QuAL under PVT variations", in 27th Proc. PATMOS, Thessaloniki, Greece, 2017, pp. 1-8. [HSR4]

5) **Himadri Singh Raghav**, Viv A. Bartlett and Izzet Kale, "Investigating the Effectiveness of Without Charge-Sharing Quasi-Adiabatic Logic for Energy Efficient and Secure Cryptographic Implementations", Microelectronics Journal, Elsevier, vol. 76, 2018, pp. 8-21. https://doi.org/10.1016/j.mejo.2018.04.004 [HSR5]

6) **Himadri Singh Raghav** and Izzet Kale "Symmetric Power Analysis Attack Resilient Adiabatic Logic for Smartcard Applications" in 28th Proc. PATMOS, Costa Brava, Spain, 2018. [HSR9]

7) **Himadri Singh Raghav** and Izzet Kale, "A Balanced Power Analysis Attack Resilient Adiabatic Logic using Single Charge Sharing Transistor", Integration, The VLSI Journal, Elsevier. https://doi.org/10.1016/j.vlsi.2018.07.010 [HSR8]

8) **Himadri Singh Raghav** and Izzet Kale, "Investigating the Trade-offs in Power-Clock Generators using Step Charging Circuits for Adiabatic Circuits", IEEE Trans. on Very Large Scale Integration (VLSI) Systems (Express Briefs). [HSR6] **(Submitted and under revision)**

9) **Himadri Singh Raghav** and Izzet Kale, "Investigating the Architecture Suitable to 4-phase Adiabatic System Implementation for Energy-Constraint Cryptographic Computations", DATE 2019. [HSR7] **(Submitted)**

10) **Himadri Singh Raghav** and Izzet Kale, "A New Synchronization Buffer Reduction Technique for Area and Energy Efficient 4-phase Adiabatic System for Smartcard Applications", ISCAS 2019. [HSR10] **(Submitted)**

11) **Himadri Singh Raghav** and Izzet Kale, "Influence of Charge-sharing and Structure Symmetry on the Secure Adiabatic Logic Designs", Microelectronics Journal, Elsevier, **(In preparation** [HSR11]

12) **Himadri Singh Raghav** and Izzet Kale, "Systolic vs Iterative Approach: Investigating the impact on 4-phase Adiabatic system and Power-Clock generator", IEEE Trans. on Circuits and Systems-1 (Regular) **(In preparation).** [HSR12]

13) **Himadri Singh Raghav** and Izzet Kale, "A New Power-Clock Gating Approach for Energy Efficient 4-phase Adiabatic Implementation for Smartcard Applications.", Microelectronics Journal, Elsevier, **(In preparation).** [HSR13]

14) **Himadri Singh Raghav** and Izzet Kale, "Tank Capacitor to Load Capacitor Ratio: Investigating the impact on the Energy Dissipation of the Step Charging Circuit and Adiabatic Load", IEEE Trans. on Very Large Scale Integration (VLSI) Systems (Express Briefs) **(In preparation)**. [HSR14]

1.5 Thesis Structure

Chapter 2 gives the general background of adiabatic technique, its power-clocking schemes, and loss mechanism. Also, a brief history of the adiabatic logic families followed by the discussion of the selected adiabatic logic families is presented. The design challenges associated with the adiabatic logic are also discussed.

Chapter 3 introduces the power-clock generation in adiabatic circuits, in particular, step charging strategy using tank-capacitor circuits is considered and several currently known step charging approaches are reviewed. Factors affecting the energy dissipation of step charging circuit are discussed. Several design rules for the implementation of energy efficient power-clock generators using step charging strategy are proposed. Also, suitable tradeoffs between step charging circuit complexity and energy performance are suggested.

Chapter 4 looks at the implementation of FSM controller to generate control signals for step-charging circuits. The energy consumption of the FSM controller for single and 4-phase power-clock generator using n-step charging circuits (where n=2, 3 and 4) is considered. Factors affecting the energy consumption of the FSM controller are discussed. Design rules and trade-offs between the complexity of FSM controller and the energy benefits in the 4-phase PCG using step charging circuit are proposed.

Chapter 5 introduces Montgomery multiplier and reviews several presently known implementations of Montgomery multiplier that applied energy reduction techniques. Several architectures for Montgomery multiplier using adiabatic logic are implemented and compared. A detailed performance evaluation of all the architectures is performed. An efficient strategy to reduce the overhead due to synchronization buffers in adiabatic logic implementation is proposed. A design methodology for scalable, area and energy efficient iterative approach architecture using 4-phase adiabatic logic is presented. In addition, Montgomery multiplication algorithm is modified. Lastly, to shut the periodically running units of the Montgomery multiplier architecture, power-clock gating is applied. A problem due to the application of power-clock gating in cascade stages of adiabatic gates is identified and a solution is proposed.

Chapter 6 introduces the general background of Power-Analysis Attack (PAA) resilient logic designs, followed by the review of currently known secure adiabatic logic designs and a summary of their shortcomings. As a solution to the shortcomings, a novel power PAA resilient adiabatic logic, WCS-QuAL is proposed. A detailed performance evaluation of the proposed and the existing secure adiabatic logic designs are performed.

Chapter 7 looks into the condition that leads to the variations in negative peak currents in the secure adiabatic logic, WCS-QuAL proposed in Chapter 6. As a solution, another novel PAA resilient adiabatic logic, proposed logic 2 is proposed. A detailed performance evaluation of the proposed logic 2, WCS-QuAL, and the existing secure adiabatic logic designs is performed.

Chapter 8 presents the conclusions drawn from this research and proposes future research directions.

2. Adiabatic Logic

This chapter presents the background material on adiabatic logic families. In addition, its switching principle, power-clocking schemes, loss mechanisms and a brief history of the adiabatic logic families are presented. The chapter ends with an overview of the design challenges associated with the adiabatic logic.

2.1 Introduction

In recent years, due to the remarkable success and growth of portable devices and use of smart cards for electronic commerce and access control, energy has become a critical concern and has perhaps superseded speed and area as the overriding implementation constraint. This has led to the increasing demand for a technique which can render ultra-low power operation. In a conventional static CMOS inverter, in the process of charging a load capacitance, $C_{L,}$ a charge of size $Q = C_L V_{DD}$ is delivered to the load. Therefore, the amount of energy supplied by the power supply, V_{DD} is $Q.V_{DD} = C_L V_{DD}^2$. The energy stored in the load capacitance C_L is only half of the energy supplied by the power supply i.e. $\frac{1}{2} C_L V_{DD}^2$. While the other half of the energy is dissipated by the pMOS transistor irrespective of the resistance of the pMOS transistor and the time taken to complete the charging. On the other hand, during the high-to-low transition, when the output capacitance starts discharging, all the energy stored in C_L is inevitably dissipated in the nMOS transistor as no energy can enter the ground rail $QV_{gnd} = Q.0 = 0$. Thus, from energy conservation viewpoint, $\frac{1}{2} C_L V_{DD}^2$ of energy is lost every time when the output node is discharged. All the charge enters at voltage V_{DD} and exits at voltage 0. The energy of the charge at the time of entry is $C_L V_{DD}^2$ and the energy at the exit is 0. Therefore, energy

dissipation in the entry to exit of the charge is $C_L V_{DD}^2$. All this energy is dissipated in the form of heat.

Since the energy dissipated during the charging and discharging of the load capacitance is fixed at twice the energy required to charge the load capacitance, one of the way to reduce the energy dissipation in a conventional CMOS logic is to charge the load capacitance slowly using a slowly changing ramp like AC power-clock rather than a DC.

2.2 Adiabatic Switching Principle

Adiabatic circuits are capable of operating with substantially less energy dissipation than conventional CMOS circuits [8]-[18] and have been in existence for more than 20 years. The term '*adiabatic'*, is of Greek origin and refers to a system in which a transition occurs without energy/heat being either lost to or gained from the system.

To have less dissipation, all the nodes should share the same principle of charging and discharging. These include (i) only turning switches off when no current is flowing through them, (ii) only turning switches on when there is no potential difference across them, and then using a slowly changing power- supply/clock -the so-called "power-clock" to evaluate the function. This can be achieved by using a slowly changing power-clock that allows approximately constant current charging/discharging and by avoiding current surges; the circuit dissipates less energy [5]. Thus, the adiabatic circuits would operate ideally with zero dissipation if the logic switching is slowed down. Decreased energy dissipation with increased switching time is, therefore, the defining property of an adiabatic switching. The use of power-clocks also makes possible the recycling of charge, enabling energy used in the computation to be recovered.

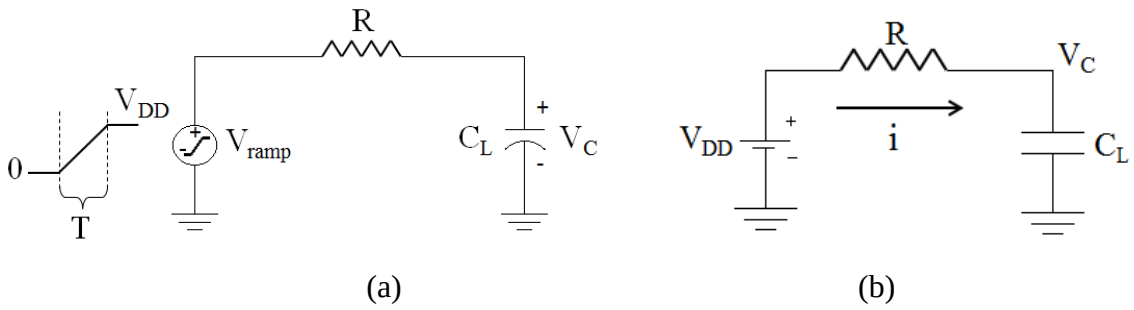


Figure 2.1: (a) Ideal adiabatic charging (b) Conventional charging

Adiabatic and conventional switching can be explained using a simple setup of the transient response with a ramp for the series *RC* circuit for adiabatic charging and with a constant DC supply for conventional charging as shown in Figure 2.1 (a) and (b) respectively.

Figure 2.1 (a) shows the adiabatic charging process. A voltage ramp (V_{ramp}) changes from 0 to V_{DD} generates a constant current I. Assuming the initial charge on the load capacitor, C_L is zero, the charge transfer through a resistance, R over a time, T by means of a voltage ramp generates a constant current I which is given as

$$I = \frac{Q}{T}$$

The total energy dissipation over a time T is therefore;

$$E_{diss} = PT = VIT = I^2RT$$

$$E_{diss} = \left(\frac{Q}{T}\right)^2 RT = \frac{Q^2R}{T}$$

In the process of charging the $C_{L,}$ up to the voltage level V_{DD}, the quantity of charge transferred is $Q=C_LV_{DD}$ and the energy dissipated in the charging path will be,

$$E_{diss} = \frac{RC_L}{T}\left(C_LV_{DD}^2\right) \qquad (2.1)$$

Similarly, during discharging, the same amount of energy will be dissipated. Therefore, the energy dissipated (Adiabatic Loss) in one cycle will be the total of energy dissipated in charging and discharging of the load capacitance:

$$E_{AL} = \frac{2RC_L}{T}\left(C_LV_{DD}^2\right) \qquad (2.2)$$

Equations 2.1 and 2.2 indicate that it is possible to reduce the energy dissipation by using a constant current source instead of a constant voltage source and by increasing the ramping time, *T* (charging and discharging time). The energy dissipation will be smaller than that of the conventional CMOS circuit if the ramping time is greater than 2*RC*. Also, the dissipated energy is inversely proportional to the ramping time, inferring that the energy dissipation can be made arbitrarily small by increasing the ramping time. From (2.1) it can be inferred that energy dissipation can be reduced to ideally zero if $T>>>RC_L$. This can be

done at the expense of increased operational time. On the other hand, for the condition $T<<<RC_{L,}$ the energy dissipation will approach that of the conventional CMOS circuit. It should also be noted that the energy dissipation in the adiabatic circuits depends on the resistance of the charging path.

In the conventional CMOS circuits, (Figure 2.1 (b)) the RC network is connected to a constant DC supply. The current, *i* in the circuit is given by

$$i = C_L \frac{dV_C}{dt}$$

The voltage across the resistor is given by

$$V_R = iR = RC_L \frac{dV_C}{dt}$$

From Kirchhoff's voltage law, the DC voltage source, V_{DD} equals the sum of the capacitor voltage (V_c) and voltage across the resistor.

$$V_{DD} = V_C + RC_L \frac{dV_C}{dt}$$

Solving the above equation, to find the voltage across the capacitor:

$$V_C = V_{DD}\left(1 - e^{\frac{-t}{RC_L}}\right)$$

Therefore, the current is given by

$$i = \frac{V_{DD}}{R}\left(1 - e^{\frac{-t}{RC_L}}\right)$$

When the load capacitor is charged through a resistance, the energy is lost in the resistor as heat and is given by:

$$E_{diss} = \int_0^\infty i^2 RT = \frac{1}{2} C_L V_{DD}^2 \qquad (2.3)$$

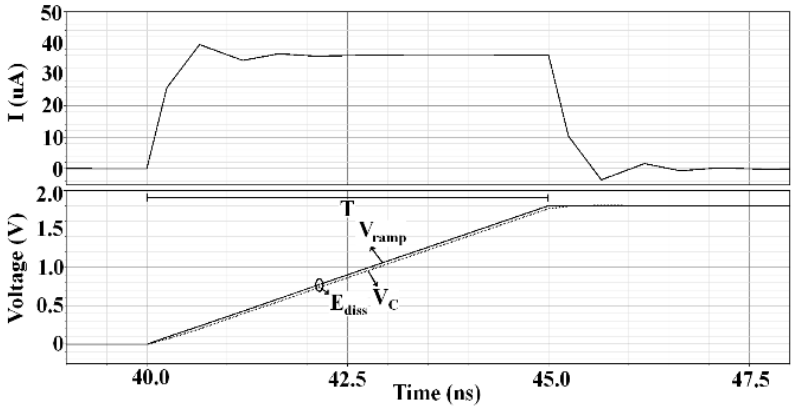
Whereas, the energy stored on the load capacitor is given by:

$$E_{stored} = \int_0^{V_{DD}} iV_C dt = \int_0^{V_{DD}} \left(C_L \frac{dV_C}{dt} \right) V_C dt$$

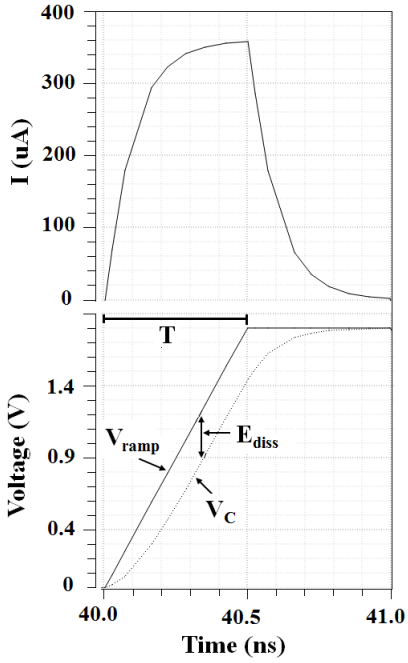
$$E_{stored} = \int_0^{V_{DD}} C_L V_C dV_C = \frac{1}{2} C_L V_{DD}^2 \qquad (2.4)$$

Thus, the energy dissipated in the conventional charging depends on the load capacitor and the supply voltage, whereas, the energy dissipated is proportional to *R* in adiabatic switching. Thus, as the resistance of the charging path decreases, the energy dissipation also decreases.

Fig 2.2 (a) and (b) shows the voltage curves and current peaks for the voltage ramp with longer and shorter ramping times respectively. From Fig 2.2(a) it can be seen that for a longer ramping time, the voltage Vc is able to follow the ramp voltage, resulting in lesser and constant peak current. For a shorter ramping time, on the other hand, the voltage, Vc is lagging behind the ramp voltage resulting in a higher current. The current graph in Figure 2.2 (b) is a typical exponential charging current graph for a conventional *RC* step response whose peak current is 10 times higher than the peak current value of Figure 2.2 (a). The same charging technique can also be used to discharge the logic from V_{DD} back to 0.



(a)



(b)

Figure 2.2: (a) Longer ramping time (b) Shorter (steeper) ramping time

2.3 Adiabatic Logic

The term '*adiabatic logic'* is used to refer logic designs that operate using adiabatic switching principle. In the literature, adiabatic logic is referred to under several different descriptors including "charge recovery logic" [6], "charge recycling logic" [9], "clock-powered logic" [13], "energy recovery logic" [12], and "energy recycling logic" [14]. These titles can be used interchangeably to refer to adiabatic logic. In recent years, the adiabatic logic design has been widely studied and exploited as an ultra-low power design technique for portable devices. Various energy recovery logic families have been proposed [8]-[27]. These can be divided into two classes, quasi-adiabatic (also known as partial energy recovery logic) and fully adiabatic [18]. The term fully adiabatic refers to the logic families that can operate theoretically entirely without losses i.e. which can in principle recover all the energy supplied. Alternatively, the term quasi-adiabatic describes logic that operates on the adiabatic switching principle but involves some theoretical energy losses. Such circuits can recover only a proportion of the energy supplied and are likely to be less complex and occupy less area than fully adiabatic logic designs. In this thesis, Quasi-Adiabatic Logic is considered.

2.4 Power-Clock Phases

Depending on the adiabatic logic family, different power-clocking schemes such as single-phase, 2-phase, 4-phase, and 8-phase are used to operate adiabatic gates connected in cascade. The adiabatic logic families requiring 8-phase power-clocking scheme are too complex and have large implementation overheads in terms of a number of transistors required and generation of power-clock phases. Consequently, these logic families were not considered. Figure 2.3 shows single-phase, 2-phase and 4-phase power-clocking scheme. For single-phase power-clocking scheme, signals C_X and C_{Xb} are the auxiliary clocks, and PC is the power-clock. For 2-phase power-clocking scheme, PC1 and PC2 are the two phases of the power-clock. Similarly for 4-phase power-clocking scheme, PC1, PC2, PC3 and PC4 are the four phases of the power-clock. The single phase and the 4-phase power-clocking scheme consists of four equal intervals namely evaluation (E), hold (H), recovery (R) and idle (I). 2-phase power-clocking scheme, on the other hand, has its evaluation, hold and recovery phase equal whereas, the idle phase thrice the length of the evaluation phase due to its non-overlapping power-clock requirement. The outputs are evaluated from the stable inputs during the evaluation phase. In the hold phase, outputs are kept stable in order to provide the stable inputs to the subsequent gate. Energy is recovered during the recovery phase of the power-clock. Power-clock remains zero during the idle phase. $T_{clk,1\text{-}phase}$, $T_{clk,2\text{-}phase}$ and $T_{clk,4\text{-}phase}$ are the durations of one power-clock phase of the single-phase, 2-phase and 4-phase power-clocking scheme respectively.

Adiabatic logic families using 4-phase power-clocking scheme has high throughput and low energy per computation in comparison to adiabatic logic families using single, and 2-phase power-clocking scheme [33].

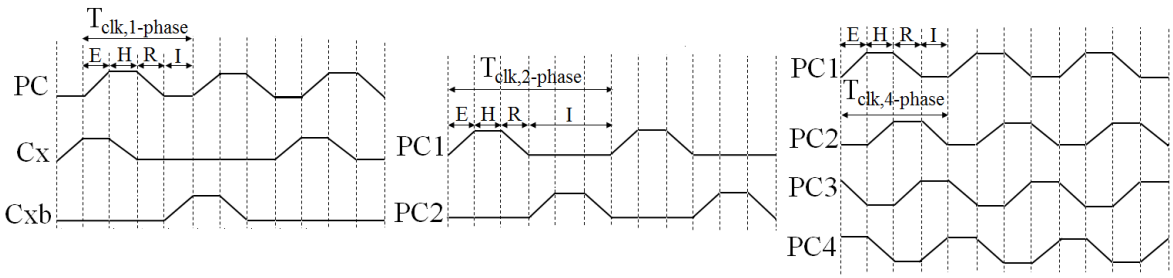


Figure 2.3: Comparison of single, 2-phase and 4-phase power-clocking scheme

2.5 Loss Mechanism in Adiabatic Logic

In an ideal adiabatic system, losses are governed by (2.2) and are called as Adiabatic Losse (AL). However, due to the shrinking devices into sub-µm regime and existence of V_{th} drop in transistors lead to additional losses. These effects can govern and also exhibit a lower bound for energy dissipation. With the shrinking devices, leakage currents can dominate the overall dissipation. One of the dominant leakage currents is the so-called sub-threshold current. It is expressed as [37]:

$$I_D = I_{D0} e^{\frac{V_{GS}-V_{th}}{\eta V_T}} \left(1 - e^{\frac{-V_{DS}}{V_T}}\right) \tag{2.5}$$

Where $I_{D0} = \frac{W \mu_0 C_{ox} V_T^2 e^{1.8}}{L}$, V_T is the thermal voltage equals to 25mV at room temperature, $V_{th,}$ is the threshold voltage of the device equal to 0.5V for nMOS and 0.44 for pMOS, V_{GS} and V_{DS} are the gate to source and drain to source voltages. W and L are the effective transistor width and length, respectively. C_{ox} is the gate oxide capacitance; μ_0 is the carrier mobility and η is the subthreshold swing coefficient (typically around 1.5). The typical value of I_{D0} for TSMC 180nm, having W=220nm, L=180nm and $\mu_0 C_{ox}$= 171 µA/V^2 is 0.79 µA. Similarly, the value of the subthreshold current, I_D at zero gate voltage (V_{GS}=0) is calculated using 2.5 is around 1.28 pA.

As long as V_{DS} is zero, no leakage current will flow. The leakage current increases to its maximum, only for values of V_{DS} that are multiples of the thermal voltage. In adiabatic logic, leakage current flows from the power-clock to ground during evaluation, hold and recovery phase. This leads to dissipation of charge that cannot be recovered. All the leakage loss can be summarized in a mean current, $I_{\overline{leak}}$, that leads to energy dissipation per cycle of:

$$E_{leak} = V_{DD} I_{\overline{leak}} \frac{1}{f} \tag{2.6}$$

Dissipation due to leakage increases for lower frequencies, as the Leakage Losses (LL), are accumulated over a longer time interval. Since in this thesis, all the work has been carried out at 180nm CMOS Process, and the frequency of operation of the application is 13.56MHz, the leakage losses will not contribute to energy dissipation significantly and therefore, are not dealt.

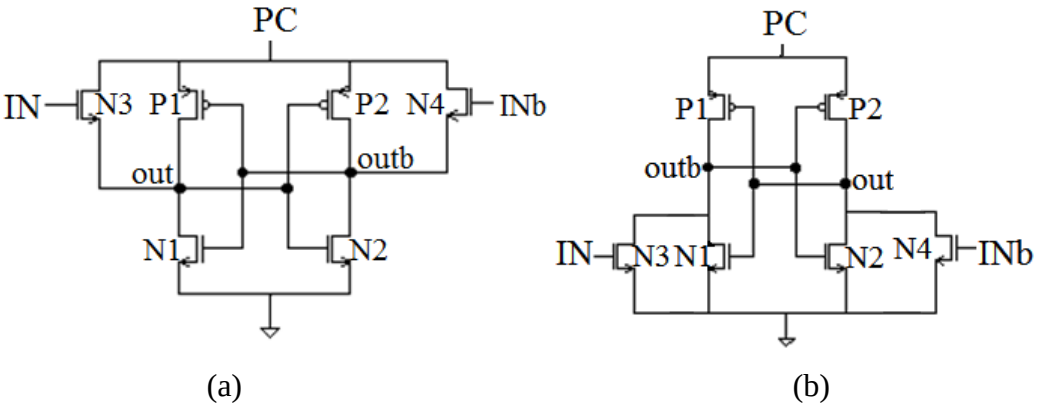


Figure 2.4: NOT/BUF gate using (a) PFAL [23] (b) IECRL [40].

Figure 2.4 (a) and (b) show the NOT/BUF gate using Positive Feedback Adiabatic Logic, PFAL [21], and Improved Efficient Charge Recovery Logic, IECRL [19] respectively. In PFAL and IECRL, during the recovery phase of the Power-Clock (PC), the charge at the output node is recovered till the power-clock doesn't go below the threshold voltage of the cross-coupled pMOS transistors leaving a residual charge at one of the output nodes. This charge is reused in the next cycle as long as the same inputs are evaluated; otherwise, it is discharged to ground. Also, in IECRL, (where the evaluation network is connected between the output nodes and the ground) during the evaluation phase, the output cannot instantly follow the rising power-clock. Only when the power-clock has reached the threshold voltage of the cross-coupled pMOS transistors, one of the output nodes follows the power-clock abruptly, leading to dissipation. All these losses are due to the threshold voltage and lead to Non-Adiabatic Loss (NAL) which is expressed as:

$$E_{NAL} = \frac{1}{2} C V_{th,p}^2 \qquad (2.7)$$

NAL is independent of the operating frequency. AL and LL, on the other hand, are dependent on the frequency of operation. An optimum frequency of operation exists for adiabatic logic, where minimum energy dissipation per cycle at a certain frequency is observed.

The three types of losses in the adiabatic circuit are presented in [37]. Adiabatic losses are proportional to frequency whereas, leakage losses are inversely proportional to frequency. Non-adiabatic losses, on the other hand, are independent of the frequency. The overall energy dissipation is obtained by summing the effects of all the three components and is given by (2.8). Where, E_{AL}, E_{leak} *and* E_{NAL} are mentioned in equations 2.2, 2.6 and 2.7

respectively.

$$E_{diss} = E_{AL} + E_{leak} + E_{NAL} \quad (2.8)$$

2.6 A brief history of Adiabatic Logic

Many different adiabatic logic families have been proposed over the last two decades. In early 1985, Seitz and co-workers were the first to formulate the relationship of adiabatic principle depicted in equation 2.1 [29]. It was in 1993 when Younis and Knight demonstrated the charge recovery process in a practical circuit [6]. Following this, in 1994, Athas showed that it is possible to achieve asymptotically zero dissipation in a CMOS design using adiabatic switching principle [30]. In 1994, Kramer and coworkers proposed a quasi-adiabatic family called 2N-2N2D [10]. It is a diode based complementary logic family. He also proposed a naming system for the adiabatic logic families he proposed. For instance, in 2N-2N2D, the first 2N represents the two nMOS transistors in the evaluation network and 2N2D denotes two nMOS transistors and two diodes connected back to back to form a latch. In the adiabatic logic family naming system, the first term suggests the transistors in the evaluation network and the latter suggests the transistors in the latch. In the same year, Dickinson and Denker proposed Adiabatic Dynamic logic which is implemented using alternate stages made of nMOS and pMOS transistors and recovery being carried out through diodes [8]. Since it is implemented using diodes and thus has Non Adiabatic Loss (NAL), consequently falls in the class of quasi-adiabatic logic. In 1995, Efficient Charge Recovery Logic family, ECRL [31], [32] and 2N-2P logic family [17] was independently proposed by Moon and Kramer respectively. ECRL and 2N-2P have identical structures and uses a pair of nMOS transistors to evaluate functions (denoted by 2N) and a pair of cross-coupled pMOS transistors (designated by 2P) to retain the state. Since complete recovery of the charge is not possible through the cross-coupled pMOS transistors, it remains a quasi-adiabatic logic. In 1994, Denker proposed an adiabatic logic family and described it as “Adiabatic logic gate” [18]. Later in 1995, it was named as 2N-2N2P [17] by Kramer. It was in 1998 when Liu named it as Improved Efficient Charge Recovery Logic, IECRL logic family [19]. In this, thesis these logic families will be referred as IECRL. It is an improvement over ECRL. The only difference is that IECRL has a pair of cross-coupled nMOS transistors in addition to the cross-coupled pMOS transistors. In 1995, Maksimovic et al. proposed Clocked Adiabatic Logic (CAL) [20] which is similar to IECRL logic family [17], [19] except for the fact that it has

clocked nMOS transistors connected between the evaluation network and the output. The clocked nMOS transistors use a pair of auxiliary-clocks which allows it to use single-phase power-clock. In 1996, Vetuli et al. presented an adiabatic logic family which makes use of CMOS positive feedback amplifier. The logic is called Positive Feedback Adiabatic Logic, PFAL [21]. It is very similar to IECRL, except that its evaluation network is connected between the power-clock and the outputs. In 1998, it was later called as PAL-2N [22] and was proposed by Liu. It can be made fully adiabatic if the recovery path is provided. PFAL will be discussed in detail later in this section.

Pass-transistor Adiabatic Logic, PAL [24] was proposed by Oklobdzija and Maksimovic in 1997. The circuit topology of PAL resembles PFAL without the pair of cross-coupled nMOS transistors. The logic works with 2-phase power-clocking scheme.

In 2001, Varga et al. proposed Efficient Adiabatic Charge Recovery Logic (EACRL) [26]. It has a pair of cross-coupled pMOS transistors and duplicate evaluation network, one connected between the power-clock and the output nodes and the other connected (with opposite assertion level) between the output nodes and the ground.

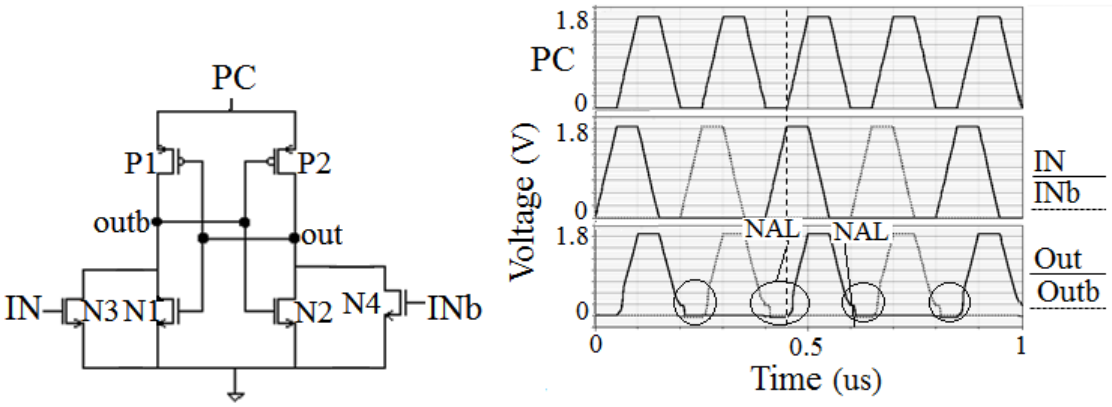
In 2003, Hu et al. proposed Complementary Pass-transistor Adiabatic Logic (CPAL) [29]. Its structure is made of PFAL buffer with the evaluation network designed using pass-transistors connected to the gates of the nMOS pull-ups also called bootstrapped transistors.

Adiabatic logic designs such as IECRL, EACRL, and PFAL require 4-phase power-clocking scheme and are the simplest of the quasi-adiabatic logic families. Also, these are considered to be the most energy efficient amongst the other single phase, 2-phase and 4-phase adiabatic logic families. Therefore, only these three will be discussed in detail.

2.6.1 Improved Efficient Charge Recovery Logic (IECRL)

As mentioned before, IECRL [19] is an enhancement of ECRL [31], [32]. Figure 2.5 (a) and (b) shows the schematic of NOT/BUF gate using IECRL and its operation waveform respectively. It is a logic based on latch having cross-coupled inverters which is a structure similar to the storage element of the Static Random Access Memory (SRAM). The addition of cross-coupled nMOS transistors provide a pull-down path during the recovery phase of the power-clock and thus remove the floating node condition reducing the coupling effect and decreasing the NAL during the recovery phase of the power-clock.

Like ECRL, IECRL also suffers from NAL during the evaluation phase of the power-clock.

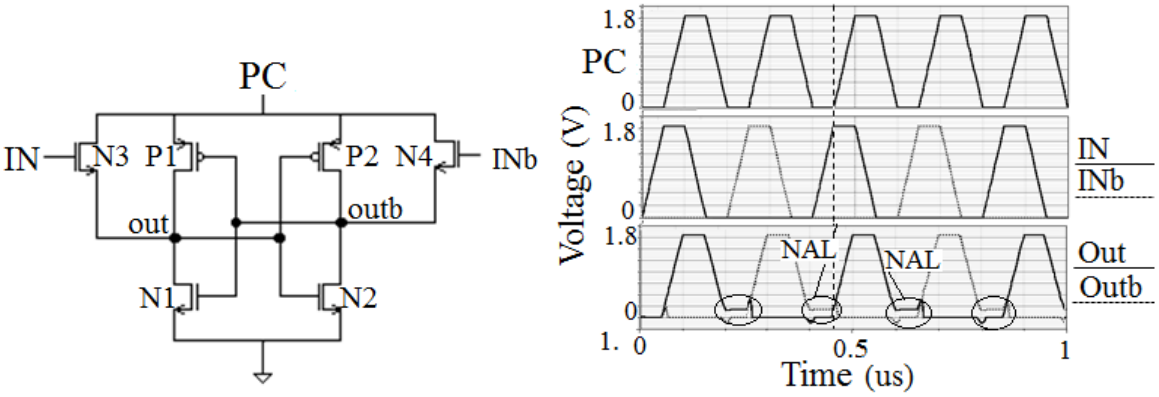


(a) (b)

Figure 2.5: (a) IECRL NOT/BUF gate [19] (b) Operation waveform

2.6.2 Positive Feedback Adiabatic Logic (PFAL)

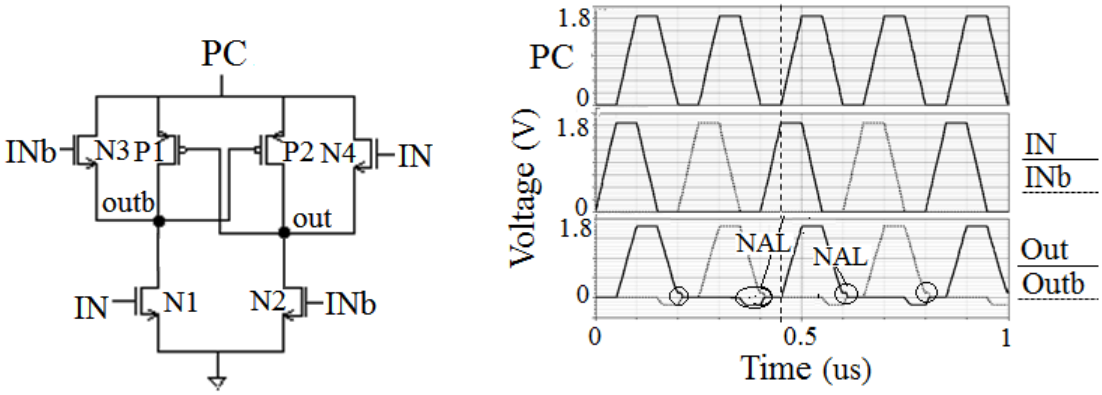
PFAL [21], like IECRL [19], also has a latch made of cross-coupled inverters. The only difference is the connection of the evaluation network. Figure 2.6 (a) and (b) shows the schematic of the NOT/BUF gate using PFAL and its operation waveform respectively. In PFAL, the evaluation network is connected between the power-clock and the output nodes. This helps to remove the NAL during the evaluation phase of the power-clock completely. The equivalent resistance at the two output nodes is also reduced due to the formation of the transmission gate pair (P1, N3, and P2, N4) with the cross-coupled pMOS transistors. PFAL suffers from NAL only for the part of recovery phase of the power clock, when the power-clock ramp down below the threshold voltage, $|V_{th,p}|$, of the pMOS transistors and the transistor is turned off leaving the residue charge on the output node. This charge is used in the next cycle if the same inputs arrive or discharged to ground non-adiabatically during the idle phase of the power-clock if different inputs arrive.



(a) (b)

Figure 2.6: (a) PFAL NOT/BUF gate [21] (b) Operation waveform

2.6.3 Efficient Adiabatic Charge Recovery Logic (EACRL)



(a) (b)

Figure 2.7: (a) EACRL NOT/BUF gate [26] (b) Operation waveform

As mentioned above, EACRL [26] was proposed by Varga et al. Figure 2.7 (a) and (b) shows the schematic of the NOT/BUF gate using EACRL and its operation waveform respectively. It is a mix of ECRL and PFAL. Its structure uses a pair of cross-coupled pMOS transistors and has pull down evaluation network similar to ECRL and a pull up evaluation network similar to PFAL. Like ECRL it also suffers from coupling effect during the part of the hold phase and complete recovery phase of the power-clock due to the absence of cross-coupled pull-down nMOS transistors. Though, having duplicate evaluation network makes it slightly better than ECRL.

Due to its energy-efficient operation, PFAL is chosen for the implementation of energy efficient cryptographic systems.

2.7 Design Challenges in Adiabatic Logic

Unfortunately, adiabatic circuits have limited potential to achieve significant energy savings at the same time as high performance. Optimizing large-scale systems developed using adiabatic technique raises many design challenges. Despite the development of several sub-systems based on adiabatic logic families, there are few papers existing that report the challenges associated with the development of adiabatic circuits or on the actual savings achievable/achieved by them [34]-[38].

Pipelining is inherent in adiabatic logic and thus, it can only perform one logic evaluation per clock phase. Therefore, every gate introduces a phase delay in propagating from input to output. Two signals originating at different phases must be synchronized before feeding to the same adiabatic gate as inputs. This requirement implies that synchronization buffers, each of which shifts a signal by one phase (a quarter of a clock period in a 4-phase system) need to be added in the circuit in order to maintain data synchronization.

In an adiabatic system implementation, the synchronization requirement can result in significant area overhead and degrades the energy benefits. Furthermore, the inherent pipelining leads to a rise in latency. The authors, M. C. Knapp et al. [34], [35] and P. Teichmann et al. [36]-[38] have discussed the design challenges (such as; circuit speed and overhead due to synchronization buffers) of arithmetic structures using adiabatic logic. The former suggested that by using an adiabatic logic family that works on 2-phase power-clocking scheme (where each gate introduces a half clock-period delay) would require fewer synchronization buffers but would get less work done per clock-period since only two logic functions could be performed per clock-period (Figure 2.3). The author also suggested an efficient method of buffer insertion. The details of this method can be found in [35]. P. Teichmann et al. on the other hand suggested that in certain arithmetic circuits nesting can reduce the need for synchronization buffers. The detailed discussion can be found in [36], [38]. He also proposed that use of complex gates can offer an overall energy reduction by reducing the overhead due to synchronization buffers. The detailed illustration of the method can be found in [37]. In this thesis, a solution to reduce the overhead due to synchronization buffers is proposed and is discussed in chapter 5.

The application of power-clock gating in adiabatic logic has been discussed in detail by P. Teichmann et al. [37], [39]-[41] and Jianping Hu et al. [42]. In cascaded stages of adiabatic

gates, the application of power-clock gating causes the problem of floating output nodes and degrades the energy efficiency that would otherwise be obtained. In this thesis, the solution is proposed. This is also discussed in detail in chapter 5.

2.8 Chapter Summary

This chapter discussed the idea of adiabatic switching principle, fully and quasi-adiabatic logic families, power-clock phasing and loss mechanism in adiabatic logic. A brief history of adiabatic logic families is presented. Also, a brief discussion of the selected quasi-adiabatic logic families requiring 4-phase power-clocking scheme is presented along with their merits and demerits. PFAL has reduced NAL and the equivalent resistance of the charging path and thus was chosen on the basis of its energy efficiency. Lastly, design challenges associated with the development of a large-scale system using non-trivial adiabatic logic are discussed.

3. Power-Clock Generation

In order to realize the potential of adiabatic logic design, as accurately as is practical, the generation of power-clock is essential. In this Chapter, the power-clock generator using step charging strategy is considered. Factors affecting the energy dissipation of step charging circuit are presented. The work reported in this chapter relates principally to several design rules proposed for the implementation of energy efficient power-clock generator using step charging strategy. Also, suitable tradeoffs between step charging circuit complexity and energy performance are suggested.

3.1 Introduction

Adiabatic circuits work with power supply clock called the power-clock, which has both the functions of supplying power and timing the operations of the circuits. A complete adiabatic system consists of the adiabatic circuit and its power clock generator (PCG). Power-clock generation is one of the main concerns for the development of energy efficient adiabatic systems as it plays an important role in the overall system efficiency. A carefully designed power-clock generator can reduce the power consumption of the system significantly [46].

Also, a key aspect in the evaluation of the potential and the perspectives of adiabatic logic families is the performance of the complete system, including the PCG. In literature, most attention has been given to the logic operation and performance of the adiabatic core without considering the performance of the power-clock generator. Although these studies illustrate the energy efficiency of the adiabatic logic families, they are incomplete since the power-clock generator consumes the large fraction of the total energy consumed by the adiabatic system, therefore degrading the energy savings greatly, if not implemented efficiently. Only in a few papers have the PCG been considered [20], [43].

In literature, several methods for power-clock generation have been proposed [44]–[47]. Most of the papers have used either step charging [44] or resonant charging [45]–[47]. In resonant charging inductors, capacitors and MOS switches are used. These circuits approximate the ramp by generating a sinusoidal waveform. However, the use of inductors presents the problem of on-chip integration; therefore, step charging circuits offer a more promising solution. In step charging circuits, the load capacitance is charged from the power supply and the tank capacitors and a step charging waveform is produced. Step charging circuits will be discussed in detail in this chapter.

3.2 Background of Step Charging Circuits

In order to power adiabatic logic, the power-clock should ideally be a ramp which rises and falls linearly. This allows approximately constant current charging and discharging and by eliminating the current surges, the circuit dissipates less energy. Such a ramp can be approximated by a step charging circuit. A power-clock which rises and falls in n-steps is shown simplified in Figure 3.1 [44].

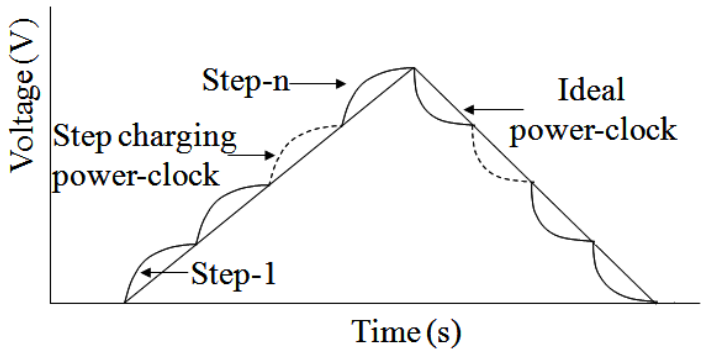


Figure 3.1: Approximation of an ideal ramp using n-step charging power-clock.

This approximation to a ramp can be improved by using more voltage steps thereby improving energy dissipation at the cost of increased circuit complexity. To implement step charging circuit with n-steps, n-1 “tank-capacitors” are required.

There are several papers that addressed the design of step charging circuits for adiabatic charging/discharging of the capacitive load. Consideration is mostly given to circuit topology and the stability of the step charging circuits [48]-[54], The authors in [48] have presented a step charging circuit which is independent of the tank-capacitor topology. However, the ratio of the tank-capacitors to load capacitor used in the step charging circuit

is 270. In [49] and [50] the authors have discussed the stability of a step charging circuit which uses tank-capacitors connected in series. However, the ratio of the tank-capacitors to load capacitor used is 750 which is quite big and will, therefore, consume a large silicon area. In [51] a step charging circuit with an equalizing capacitor that equalizes the node voltages of the tank-capacitors by connecting "touching" them is presented. The stability of the step charging circuit is also investigated by changing the order in which the tank-capacitor nodes are connected "touched" [51] with the equalizing capacitor. However, the ratio between the tank-capacitors and the equalizing capacitor used is 300. The authors in [52] have presented a step charging circuit and the stability of the step charging circuit is considered. The authors state that when the size of the tank-capacitor is much larger than the load capacitor, the step charging circuit offers stable operation even if the value of the load capacitor changes significantly. However, how large the size of the tank-capacitance in comparison to the load capacitance should be in order to ensure stability is not mentioned. In [53] and [54] the adiabatic stepwise charging and discharging of a capacitor with an inductor current that controls the switching transistors is demonstrated experimentally and the energy consumption is investigated as the function of the number of steps. So far all the above-cited references workaround using large tank-capacitor values for stability. Large tank-capacitors incur high silicon area cost. This can be a problem for the area constrained applications. Therefore, the relationship of total tank-capacitance to load capacitance that can deliver potential energy benefits with lower silicon area cost and stable operation is worth investigation.

Energy recovery determines the efficiency of the adiabatic circuit technique, therefore an important parameter to be considered for the design of adiabatic systems. As mentioned above, the power-clock in the adiabatic circuits make possible the recovery of charge, enabling energy to be recovered. Therefore, it is important to study the factors that affect the energy recovery achievable in step charging circuits.

The energy performance of the adiabatic circuits is additionally a function of ramping time. Therefore, it would be worth looking if increasing the ramping time of the step charging circuit influences the percentage energy recovery.

In a step charging circuit, pMOS transistor is used for charging the load capacitor from the power supply, CMOS transmission gates (TG) are used for the charging/discharging of the load capacitor to/from tank-capacitor and a nMOS transistor is used to discharge the load capacitor to the ground. Width of these transistors can affect the charging/discharging of

the load capacitor and in turn affect the energy recovery specifically at shorter ramping times (high frequency). Therefore, the transistor widths in the step charging circuit that can deliver potential energy benefits and how transistor widths influence the energy recovery achievable at different ramping times is worth investigating.

An easy and powerful way to reduce the energy dissipation in static CMOS is by reducing the power supply voltage, V_{DD}. It is because of the quadratic dependence of the energy dissipation on the V_{DD}.

$$E_{CMOS} \alpha V_{DD}^2 \tag{3.1}$$

Energy dissipation (charging) in adiabatic circuits is also proportional to the square of the supply voltage which is given by (2.1). Thus, energy dissipation reduces as the supply voltage is scaled down. With the decrease in supply voltage, energy supplied to the circuit will also decrease. Energy supplied, E_S to the circuit is measured for the evaluation phase of the power-clock, whereas, energy recovery, E_R is measured during the recovery phase of the power-clock. Energy dissipated is the difference of energy supplied; E_S and energy recovered, E_R

$$E_D = E_S - E_R \tag{3.2}$$

The percentage energy recovery is calculated as:

$$E_R = \left(\frac{E_R}{E_S} \right) \times 100 \tag{3.3}$$

For energy efficient operation, the impact of power supply voltage scaling on energy recovery is worth investigating.

Large tank-capacitors have a high silicon area cost but offer stable operation of the step charging circuit and therefore better energy efficiency. An appropriate trade-off is also a function of load capacitance, C_L. Therefore, it is important to investigate the appropriate ratio of tank-capacitance to load capacitance which can deliver potential energy benefits in step charging circuits. A metric called "C_T/C_L ratio" which denotes the ratio of tank-capacitance to load capacitance in a 2-step charging circuit was defined. Step charging circuits having a number of steps more than two; require more than one tank-capacitor. Therefore, another metric called "CTT/C_L ratio" which denotes the ratio of combined

(total) tank capacitance, C Total Tank (*CTT*) to load capacitance (C_L) in step charging circuit for a number of steps, n= 3, 4, 5, 6, 7, and 8 was defined. Total tank-capacitance (*CTT*) is the term used to denote the total of all the tank-capacitor values in a step charging circuit.

3.3 n-Step Charging Circuits

The basic structure of an n-step charging circuit using n-1 tank-capacitors is shown in Figure 3.2 (a), driving a capacitive load, C_L [44]. Each switch is momentarily closed in ascending order from S_1 to S_{n+1} and back to S_1. In steady state, this produces a step-like waveform as shown in Figure 3.2 (b)

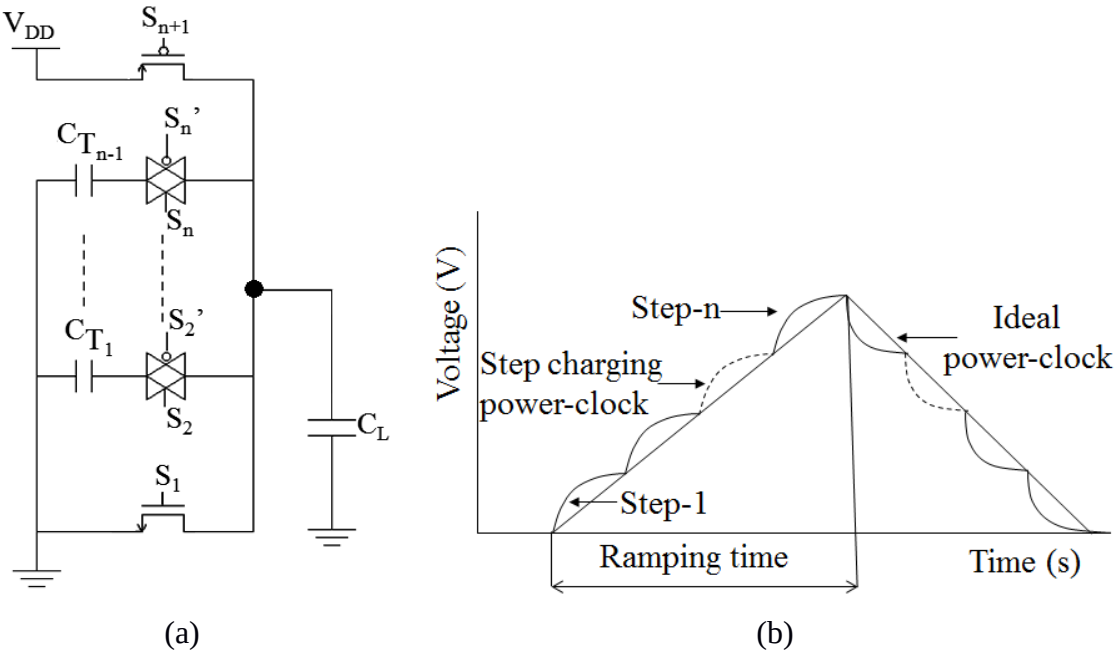


Figure 3.2: (a) n-step charging circuit [44] (b) Step charging output waveform as an approximation of a ramping power-clock

In a step charging circuit, the energy dissipation depends on the number of steps, n. assuming all the voltage steps are equal, during the i_{th} (where i =1, 2, 3, …n) step, the load capacitance is charged from voltage $\frac{(i-1)V_{DD}}{n}$ to $\frac{iV_{DD}}{n}$ under the voltage $\frac{iV_{DD}}{n}$, therefore the energy dissipation in this step is given by:

$$E_{diss,step} = \int_{\frac{(i-1)V_{DD}}{n}}^{\frac{iV_{DD}}{n}} \left(\frac{iV_{DD}}{n} - V \right) C_L dV = \frac{C_L V_{DD}^2}{2n^2} \tag{3.4}$$

Adding $E_{diss,\ step}$ for i from 1 to n, is given by the expression below

$$E_{diss,tot} = nE_{step} = \frac{C_L V_{DD}^2}{2n} \tag{3.5}$$

The above expression shows that the energy dissipation is reduced to $\frac{1}{n}$ in n-step charging compared to the conventional direct charging. The conventional direct charging corresponds to n=1. The energy stored in the capacitor in the i_{th} step is given as:

$$E_{store,step} = \frac{1}{2} C_L \left\{ \left(\frac{iV_{DD}}{n} \right)^2 - \left(\frac{(i-1)V_{DD}}{n} \right)^2 \right\} = \frac{(2i-1)C_L V_{DD}^2}{2n^2} \tag{3.6}$$

When the load capacitor is charged to V_{DD}, the total stored energy, $E_{store,\ tot}$ will be

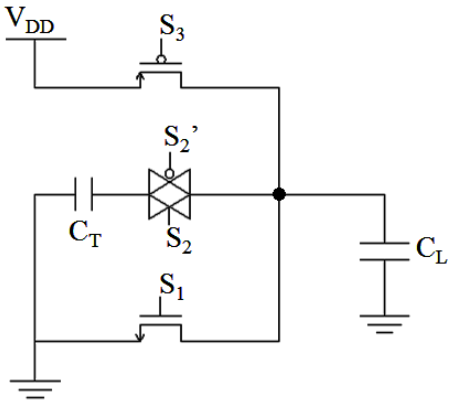
$$E_{store,tot} = \sum_{i=1}^{n} E_{store,step} = \frac{1}{2} C_L V_{DD}^2 \tag{3.7}$$

All the switches of the step charging circuit of Figure 3.2 (a) are CMOS transmission gates except for the switches to V_{DD} and ground which can be simple pMOS and nMOS transistors respectively. The switches of the step charging circuit are controlled using a Finite State Machine (FSM) controller which will be discussed in detail in the next chapter. As the number of steps increases, the waveform becomes a progressively better approximation to a ramp and energy performance is thereby improved [44].

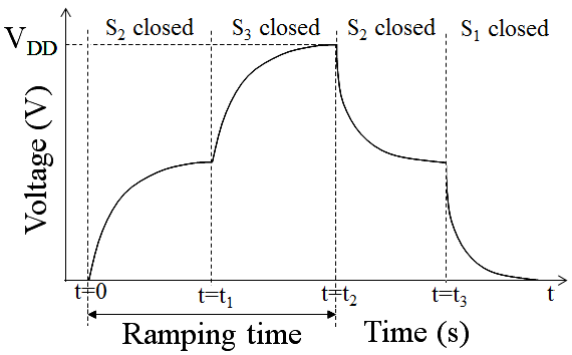
3.4 Analytical Modelling for Charge replenish in Tank capacitor of the 2-Step Charging Circuit

A 2-step charging circuit is shown driving a capacitive load, C_L, using a single tank-capacitor, C_T, in Figure 3.3(a). Each switch is momentarily closed in the sequence S_1, S_2, S_3, S_2, S_1… under the control of an FSM controller. In steady state, this produces a step-like waveform as shown in Figure 3.3(b). In Figure 3.3(a), the pMOS switch, S_3, is used

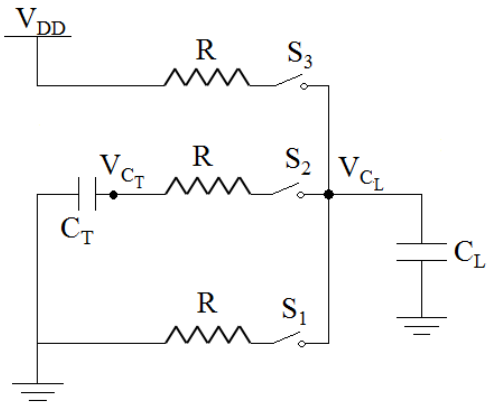
for charging the load capacitor, C_L, to V_{DD} and the nMOS switch, S_1, is used for discharging C_L to ground. A CMOS transmission gate is used for charging/discharging C_L to the intermediate voltage stored on the tank-capacitor, C_T.



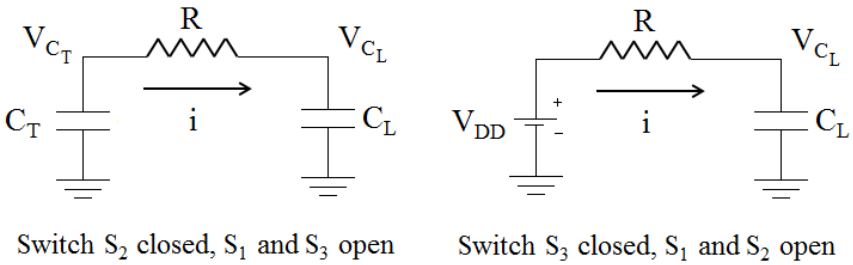
(a)



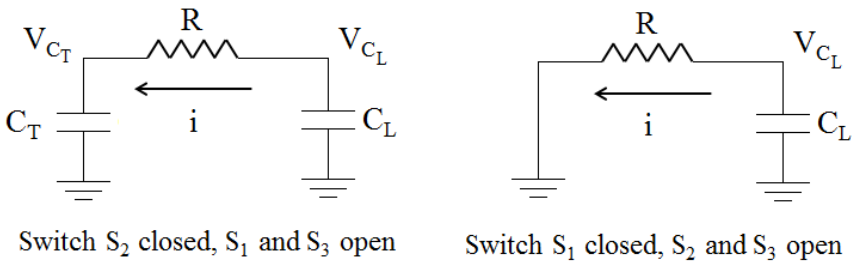
(b)



(c)



(d)



(e)

Figure 3.3: (a) 2-step charging circuit [44] (b) 2-step charging output waveform showing intervals of switches that are closed (c) 2-step charging circuit with MOS switches modelled as resistance R (d) model for charging of load capacitor (e) model for discharging of load capacitor.

Figure 3.3 (c), (d) and (e) Show the 2-step charging circuit with MOS switches modelled as resistance R, model for charging the load capacitor when switch S_2 (charging from the tank capacitor) and S_3 (charging from the supply voltage V_{DD}) are closed and model for discharging the load capacitor when switches S_2 (discharging to the tank capacitor) and S_1 (discharging to the ground) are closed respectively.

During charging:

It is assumed that the node V_{C_L} is discharged to zero and there is a voltage, V_1 on the node V_{C_T}.

$$\left.\begin{aligned} V_{C_L}(0) &= 0 \\ V_{C_T}(0) &= V_1 \end{aligned}\right\} \quad \text{Initial conditions}$$

Case: When switch S_2 is closed and S_1 and S_3 are open (Figure 3.3 (d)).

During the charging process, the tank capacitor, C_T charges the load capacitor, C_L till the voltages on the nodes V_{C_L} and V_{C_T} equalize. It is assumed that the closing time of the switch is sufficient such that both the nodes reach steady state.

$$V_{C_T} - V_{C_L} = iR$$

When C_L is charged

$$V_{C_T} - V_{C_L} = RC_L \frac{dV_{C_L}}{dt}$$

$$V_{C_T} = V_{C_L} + RC_L \frac{dV_{C_L}}{dt} \tag{3.8}$$

Taking Laplace transform of (3.8)

$$V_{C_T}(s) = V_{C_L}(s) + RC_L\left(sV_{C_L}(s) - V_{C_L}(0)\right) \tag{3.9}$$

When C_T is discharged

$$V_{C_T} - V_{C_L} = -RC_T \frac{dV_{C_T}}{dt}$$

$$V_{C_T} = V_{C_L} - RC_T \frac{dV_{C_T}}{dt} \tag{3.10}$$

Taking Laplace transform of (3.10)

$$V_{C_T}(s) = V_{C_L}(s) - RC_T\left(sV_{C_T}(s) - V_{C_T}(0)\right)$$

$$V_{C_T}(s) = V_{C_L}(s) - sRC_TV_{C_T}(s) + V_1RC_T$$

$$V_{C_L}(s) = V_{C_T}(1 + sRC_T) - V_1RC_T \tag{3.11}$$

Putting (3.9) in (3.11)

$$V_{C_L}(s) = V_{C_L}(1 + sRC_L)(1 + sRC_T) - V_1RC_T \tag{3.12}$$

Solving (3.12) for V_{C_L}

$$V_{C_L}(s) = \frac{V_1C_T}{s\left(sRC_LC_T + C_L + C_T\right)} = \frac{A}{s} + \frac{B}{sRC_LC_T + \left(C_L + C_T\right)} \tag{3.13}$$

Solving (3.13) for *A* and *B*

$$A = \frac{V_1C_T}{C_L + C_T}; \qquad B = \frac{V_1C_T}{C_L + C_T}\left(RC_LC_T\right)$$

$$V_{C_L}(s) = \frac{\frac{V_1C_T}{C_L + C_T}}{s} - \frac{\frac{V_1C_T}{C_L + C_T}\left(RC_LC_T\right)}{sRC_LC_T + \left(C_L + C_T\right)} \tag{3.14}$$

$$V_{C_L}(s) = \frac{\frac{V_1C_T}{C_L + C_T}}{s} - \frac{\frac{V_1C_T}{C_L + C_T}}{s + \frac{C_L + C_T}{RC_LC_T}} \tag{3.15}$$

Taking inverse Laplace of (3.15)

$$\frac{1}{s} = u(t);\ \frac{1}{s+\alpha} = e^{-\alpha t}u(t)$$

$$V_{C_L}(t) = \frac{V_1 C_T}{C_L + C_T} - \frac{V_1 C_T}{C_L + C_T} e^{-\left(\frac{C_L + C_T}{RC_L C_T}\right)t} \tag{3.16}$$

Replacing $\frac{RC_L C_T}{C_L + C_T}$ with τ , (3.16) is re-written as

$$V_{C_L}(t) = \frac{V_1 C_T}{C_L + C_T}\left(1 - e^{\frac{-t}{\tau}}\right) \tag{3.17}$$

Similarly for $V_{C_T}(t)$, from equations (3.9) and (3.11)

$$V_{C_T}(s) = \left(V_{C_T}(1 + sRC_T) - V_1 RC_T\right)(1 + sRC_L) \tag{3.18}$$

$$V_{C_T}(s) = \frac{V_1 C_T (1 + sRC_L)}{s(sRC_L C_T + C_L + C_T)} \tag{3.19}$$

Solving (3.19) similar to the equation (3.13)

$$V_{C_T}(s) = \frac{\frac{V_1 C_T}{C_L + C_T}}{s} + \frac{\frac{V_1 C_L}{C_L + C_T}}{s + \frac{C_L + C_T}{RC_L C_T}} \tag{3.20}$$

Taking inverse Laplace of (3.20) and replacing $\frac{RC_L C_T}{C_L + C_T}$ with τ

$$V_{C_T}(t) = \frac{V_1 C_T}{C_L + C_T} + \frac{V_1 C_L}{C_L + C_T} e^{\frac{-t}{\tau}}$$

$$V_{C_T}(t) = \frac{V_1 C_T}{C_L + C_T}\left(1 + \frac{C_L}{C_T} e^{\frac{-t}{\tau}}\right) \tag{3.21}$$

The charging current when the switch S_2 is closed

$$i(t) = \frac{V_{C_T}(t) - V_{C_L}(t)}{R} = \frac{V_1}{R} e^{\frac{-t}{\tau}} \tag{3.22}$$

Case: When switch S_3 is closed and the switches S_1 and S_2 are open

Boundary condition at $t=t_1$,

$$V_{C_L}(t_1) = \frac{V_1 C_T}{C_L + C_T}\left(1 - e^{\frac{-t_1}{\tau}}\right) \tag{3.23}$$

Charging current from V_{DD}

$$i = \frac{V_{DD} - V_{C_L}(t)}{R}$$

$$V_{C_L}(t) = V_{DD} - RC_L \frac{dV_{C_L}}{dt}$$

Integrating the above for the value of $V_{C_L}(t)$

$$\int \frac{dV_{C_L}(t)}{V_{DD} - V_{C_L}(t)} = \int \frac{dt}{RC_L}$$

$$\ln\left(V_{DD} - V_{C_L}(t)\right) = \frac{-t}{RC_L} + D \tag{3.24}$$

$$\left(V_{DD} - V_{C_L}(t)\right) = e^{\frac{-t}{RC_L}}.e^D$$

$$V_{DD} - e^{\frac{-t}{RC_L}}.e^D = V_{C_L}(t) \tag{3.25}$$

Substituting the boundary in (3.25) conditions t=t_1 to find e^D

$$\left(V_{DD} - V_{C_L}(t)\right) = e^{\frac{-t}{RC_L}}.e^D$$

$$e^D = \left(V_{DD} - \frac{V_1 C_T}{C_L + C_T}\left(1 - e^{\frac{-t}{\tau}}\right)\right)e^{\frac{t_1}{RC_L}} \tag{3.26}$$

Now substituting (3.26) in (3.25)

$$V_{C_L}(t) = V_{DD} - e^{\frac{-t}{RC_L}}\left(V_{DD} - \frac{V_1 C_T}{C_L + C_T}\left(1 - e^{\frac{-t_1}{\tau}}\right)\right)e^{\frac{t_1}{RC_L}}$$

$$V_{C_L}(t) = V_{DD} - e^{\frac{-(t-t_1)}{RC_L}}\left(V_{DD} - \frac{V_1 C_T}{C_L + C_T}\left(1 - e^{\frac{-t_1}{\tau}}\right)\right)$$

$$V_{C_L}(t) = V_{DD}\left(1 - e^{\frac{-(t-t_1)}{RC_L}}\right) + \frac{V_1 C_T}{C_L + C_T} e^{\frac{-(t-t_1)}{RC_L}} \tag{3.27}$$

Now at t=t_1, where t_1 is large, (3.27) can be approximated as;

$$V_{C_L}(t) \approx \frac{V_1 C_T}{C_L + C_T} \tag{3.28}$$

When t=t_2, t_2-t_1 is large and substituting the value of V_1 in (3.28)

$$V_{C_L}(t) \approx V_{DD} \tag{3.29}$$

$$i = \frac{V_{DD} - V_{C_L}(t)}{R} \approx \frac{e^{\frac{-(t-t_1)}{RC_L}}}{R}\left(V_{DD} - \frac{V_1 C_T}{C_L + C_T}\right) \tag{3.30}$$

Discharging Process:

Case: Switch S_2 is closed again and switches S_3 and S_1 are open. (Figure 3.3 (e)).

Now, C_L will discharge to C_T till the charge on the two capacitors equalises.

Putting t=t_2 in (3.27) for the initial condition on the node V_{C_L} .

$$V_{C_L}(t_2) = V_{DD}\left(1 - e^{\frac{-(t_2-t_1)}{RC_L}}\right) + \frac{V_1 C_T}{C_L + C_T} e^{\frac{-(t_2-t_1)}{RC_L}} \tag{3.31}$$

Assuming t_2-t_1 is large;

$V_{C_L}(t_2) \approx V_{DD}$

At node V_{C_T} when t=t_2, from (3.21)

$$V_{C_T}(t_2) = \frac{V_1 C_T}{C_L + C_T}\left(1 + \frac{C_L}{C_T} e^{\frac{-t_2}{\tau}}\right)$$

$$= \frac{V_1 C_T}{C_L + C_T} \approx V_2 \tag{3.32}$$

$$i = \frac{V_{C_L} - V_{C_T}}{R}$$

When C_L is discharging and using Kirchhoff's voltage low

$$V_{C_L} - V_{C_T} = -RC_L \frac{dV_{C_L}}{dt}$$

$$V_{C_L} = V_{C_T} - RC_L \frac{dV_{C_L}}{dt} \tag{3.33}$$

Using Laplace transform of (3.33)

$$V_{C_L}(s) = V_{C_T}(s) - RC_L\left(sV_{C_L}(s) - V_{C_T}(0)\right)$$

$$V_{C_L}(s) = V_{C_T}(s) - RC_L\left(sV_{C_L}(s) - V_{DD}\right)$$

$$V_{C_L}(s) = \frac{V_{C_T}(s) + RC_L V_{DD}}{(1 + sRC_L)} \tag{3.34}$$

When C_T is charging and using Kirchhoff's voltage low

$$V_{C_L} - V_{C_T} = RC_T \frac{dV_{C_T}}{dt}$$

$$V_{C_L} = V_{C_T} + RC_T \frac{dV_{C_T}}{dt} \tag{3.35}$$

Taking Laplace transform of (3.35)

$$V_{C_L}(s) = V_{C_T}(s) + RC_T\left(sV_{C_T}(s) - V_{C_T}(0)\right)$$

$$V_{C_L}(s) = V_{C_T}(s) + RC_T\left(sV_{C_T} - V_2\right)$$

$$V_{C_L}(s) = V_{C_T}(1 + sRC_T) - V_2RC_T \tag{3.36}$$

Solving (3.34) and (3.36) for $V_{C_L}(s)$ and $V_{C_T}(s)$

$$V_{C_T}(s) = \frac{\frac{C_LV_{DD} + V_2C_T}{C_L + C_T}}{s} - \frac{\frac{C_L(V_{DD} - V_2)}{C_L + C_T}}{s + \frac{C_L + C_T}{RC_LC_T}} \tag{3.37}$$

$$V_{C_L}(s) = \frac{\frac{C_LV_{DD} + V_2C_T}{C_L + C_T}}{s} + \frac{\frac{C_L(V_{DD} - V_2)}{C_L + C_T}}{s + \frac{C_L + C_T}{RC_LC_T}} \tag{3.38}$$

Taking inverse Laplace of (3.37) and (3.38)

$$V_{C_T}(t) = \frac{C_LV_{DD}\left(1 - e^{\frac{-t}{\tau}}\right) + V_2C_L\left(\frac{C_T}{C_L} + e^{\frac{-t}{\tau}}\right)}{C_L + C_T} \tag{3.39}$$

$$V_{C_L}(t) = \frac{C_LV_{DD}\left(1 + \frac{C_T}{C_L}e^{\frac{-t}{\tau}}\right) + V_2C_T\left(1 - e^{\frac{-t}{\tau}}\right)^2}{C_L + C_T} \tag{3.40}$$

For large t and substituting (3.32) in (3.39)

$$V_{C_T}(t) \approx \frac{C_LV_{DD} + V_2C_T}{C_L + C_T}$$

$$V_{C_T}(t) \approx \frac{C_LV_{DD}}{C_L + C_T} + V_1\left(\frac{C_T}{C_L + C_T}\right)^2 \tag{3.41}$$

For large t and substituting (3.32) in (3.40)

$$V_{C_L}(t) = \frac{V_{DD}C_L}{C_L + C_T} + \frac{V_2 C_T}{C_L + C_T}$$

$$V_{C_L}(t) \approx \frac{C_L V_{DD}}{C_L + C_T} + V_1\left(\frac{C_T}{C_L + C_T}\right)^2 \tag{3.42}$$

The discharge current from C_L to C_T (using (3.39) and (3.41))

$$i(t) = \frac{V_{C_L} - V_{C_T}(t)}{R} = \left(\frac{V_{DD} - V_2}{R}\right)e^{\frac{-t}{\tau}} \tag{3.43}$$

In this manner by periodic connection of load capacitance to the V_{DD} (equation (3.27)) and then sharing the charge (discharging to tank capacitor) with the tank capacitor in the step-wise charging/discharging manner (equations (3.17), (3.21), (3.39) and (3.40)), the tank capacitor in the steady state will eventually maintain the required voltage level.
Finally, when switch S_1 is closed and S_2 and S_3 are open. The remaining charge on C_L will be discharged to ground leading to energy dissipation.

3.5 Simulation Results

For the investigation, TSMC 180nm CMOS process was used. The load capacitance was chosen as 1pF and all the transistors were sized at minimum dimensions (W_{min}=220nm, L_{min}=180nm) except for the width of the pMOS switch, S_3 to V_{DD} which was sized at 440nm, with a view to equalizing its performance with respect to the nMOS switch, S_1. Simulations were carried out in a 'typical-typical' process corner using the CMOS process at 1.8V power supply. The tank-capacitors of the step charging circuits require a few cycles to settle. For this reason, all measurements were taken after the circuit had reached steady state. All the simulations were performed with equal L-H (Low-to-High) and H-L (High-to-Low) ramping times of 10ns, 25ns, 50ns, 100ns, 200ns, and 400ns.

To measure the energy recovery achievable with this strategy a 2-input Positive Feedback Adiabatic Logic (PFAL) [21] AND/NAND gate, as shown in Figure 3.4, was used as the test circuit of Figure 3.5. The power-clock generator comprises the 2-step charging circuit of Figure 3.2(a) together with its FSM. Figure 3.5 shows this generator driving the test circuit.

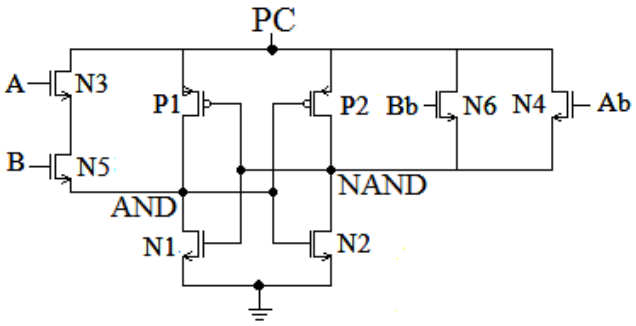


Figure 3.4: Test circuit: PFAL adiabatic AND/NAND gate [21].

The PFAL adiabatic AND/NAND gate was chosen for this study because as mentioned in chapter 2, amongst the energy efficient quasi-adiabatic logic designs such as EACRL [26], IECRL [19] and PFAL [21], the PFAL exhibits the most energy efficient operation.

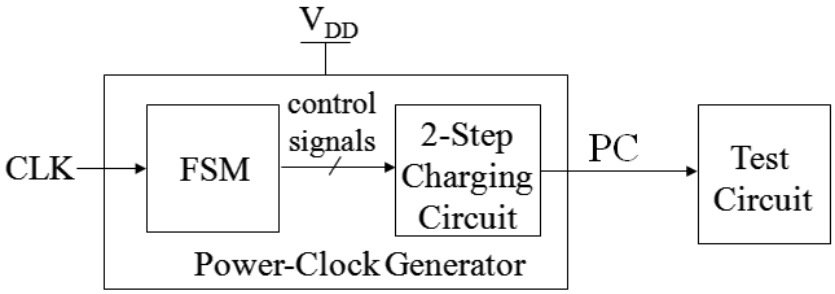


Figure 3.5: A general block diagram of an adiabatic system for single-phase power-clock.

The Simulations were performed for two cases; i) C_T/C_L ratio when C_T is varying and C_L is fixed; ii) C_T/C_L ratio when C_T is fixed and C_L is varying. Simulations were also performed to investigate if ramping time, width of the CMOS transmission gate (TG) and the power supply voltage scaling influence the energy recovery.

3.5.1 Energy Recovery vs C_T/C_L ratio at different Ramping Times

The simulation results shown in Figure 3.6 illustrate the relationship between C_T/C_L ratio (when C_T is varying, and C_L is fixed at 1pF) and percentage energy recovery at the simulated ramping times. The plot shows the "diminishing returns" of increasing C_T/C_L ratio. The "knee" of the curve occurs at around the C_T/C_L = 10 region and increasing the C_T/C_L ratio above 10, offers relatively slight improvement (less than 1%) in energy recovery in each case. This suggests that as a design rule, a C_T/C_L ratio of 10 is appropriate.

Energy performance of adiabatic circuits is additionally a function of ramping time. Figure 3.6 also compares energy recovery achievable by PFAL adiabatic logic using 2-step charging circuit at ramping times simulated. The ramping time is varied from 10ns to

400ns and not below 10ns; because the potential energy benefits of the adiabatic circuits can be obtained at low frequencies i.e. at longer ramping times. Moreover, they are intended to be used for applications such as smart cards which work on frequency around 13.56MHz. To set the ramping times of 10ns, 25ns, 50ns, 100ns, 200ns and 400ns, the time period of the Power-Clock (PC) was set to 40ns, 100ns, 200ns, 400ns, 800ns and 1600ns respectively. Figure 3.6 shows that across the whole range of C_T/C_L ratio, as the ramping time is increased above 100ns, the improvement in energy recovery is relatively small. But as the ramping time is reduced from 50ns to 10ns there is a significant decrement in energy recovery. This is supported by (2.1) which suggests that if the ramping time, T is reduced the energy dissipation increases which causes energy recovery to decrease. Thus, Adiabatic Losses (AL) dominates the energy dissipation at shorter ramping times (high frequency) and the energy recovery decreases. Additionally, for shorter ramping times, 10ns, 25ns and 50ns, the switching time of the switches is not enough to allow appropriate charging and discharging of the load capacitor and therefore, increased energy dissipation and reduced energy recovery. For ramping times, 100ns, 200ns, 400ns, the switching time is enough to allow complete charging and discharging of the load capacitor and therefore, better energy recovery compared to that of 10ns, 25ns and 50ns ramping times.

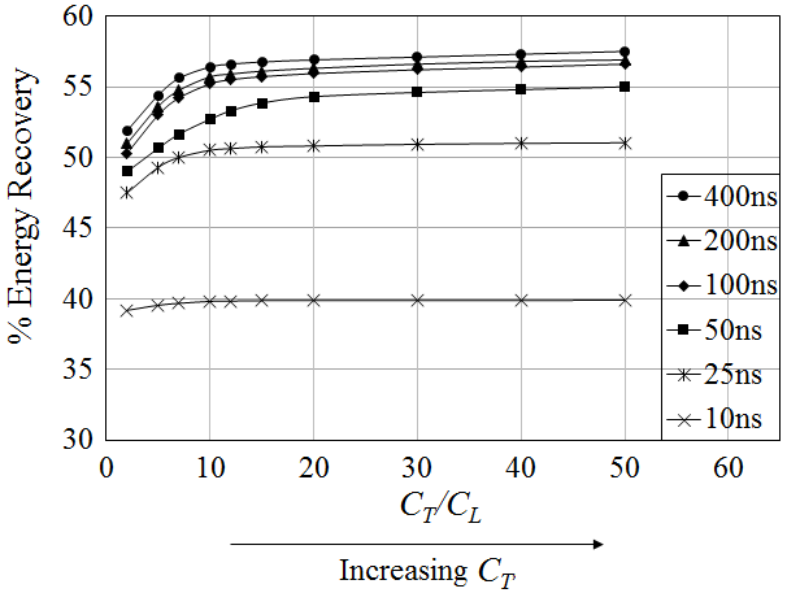


Figure 3.6: Energy recovery vs C_T/C_L ratio (when C_T is varying, and C_L is fixed) at different ramping times.

In these results, the energy cost of operating the switches in the 2-step charging circuit/FSM controller has not been included. It should also be noted that, from equation 3.5, the total energy dissipated by the 2-step charging circuit is half (50%) of the

conventional case. This suggests that a 2-step charging circuit can recover only up to 50% of the energy stored in the load capacitance. However, in Figure 3.6 the energy recovery at the ramping time of 400ns is about 57%. This is because the energy dissipation and energy recovery was measured at PC terminal of Figure 3.5, which measures the energy recovered by the PFAL adiabatic logic using 2-step charging circuit. This is the reason why energy recovery is decreased for the shorter ramping times because adiabatic losses increase for shorter ramping times. Increased losses result in increased energy dissipation and decreased energy recovery.

3.5.2 Energy Recovery vs C_T/C_L ratio at different Ramping Times

The simulation results shown in Figure 3.7 illustrate the relationship between C_T/C_L ratio (when C_T is fixed at 10pF and C_L is varying) and the percentage energy recovery at simulated ramping times. The plot shows that at C_T/C_L ratio of 1, 2 and 5 the energy recovery is less in comparison to the energy recovery at C_T/C_L ratio above 5. This is because the value of the C_L at C_T/C_L ratio of 1, 2, and 5 is 10pF, 5pF, and 2pF respectively. The large values of C_L increase the time constant of the circuit at the output node thus, preventing the output voltage of the step charging circuit to reach V_{DD}.

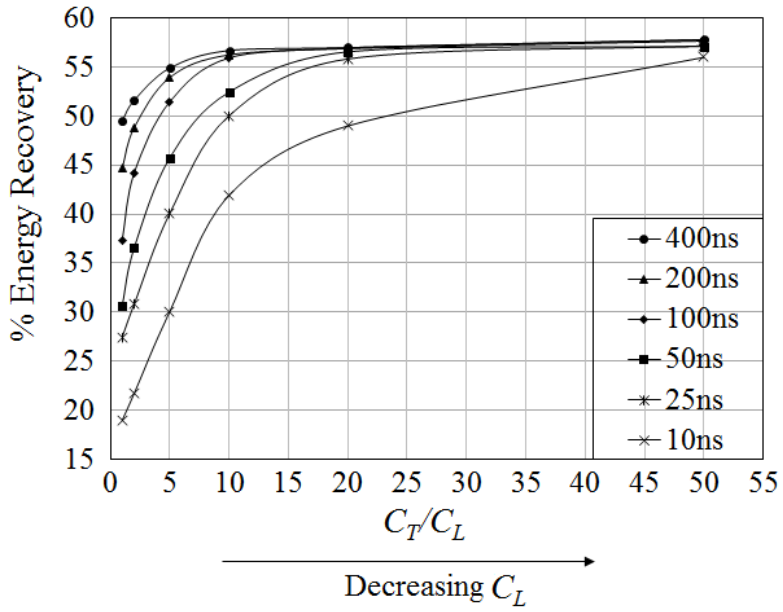


Figure 3.7: Energy Recovery vs C_T/C_L ratio (fixed C_T and varying C_L) at different ramping times.

There is no significant improvement in the energy recovery for the C_T/C_L ratio above 10 at the ramping times of 50ns, 100ns, 200ns, and 400ns. On the contrary, there is an improvement of about 12% and 6% approximately in energy recovery for the ramping

times of 10ns and 25ns respectively. This is because, for increasing C_T/C_L ratio (decreasing C_L) the time constant of the circuit decreases and the output voltage reaches to V_{DD} (complete charging/discharging), improving the energy recovery. However, in the previous case, the C_L was fixed at 1pF therefore; the switching time was not enough to charge the load capacitor to V_{DD} thus, degrading the energy recovery. For C_T/C_L ratio less than 10, the energy recovery degrades for ramping times, 100ns and above, whereas it is worst for ramping times less than100ns. This is because, for decreasing C_T/C_L ratio (increasing C_L) the time constant of the circuit is increased and the output voltage level does not reach V_{DD} (not allowing complete charging and discharging of the load capacitance), degrading the energy recovery.

3.5.3 Energy Recovery vs Ramping Time at different Transmission Gate (TG) widths

Simulations were performed to investigate the impact of CMOS TG, width on energy recovery. Here the widths of the pMOS and nMOS transistors which are connected to the supply voltage, V_{DD} and ground respectively were not changed. As on increasing the width of the pMOS transistor the current through the transistor will increase causing an increase in the total energy supplied and energy dissipation. Similarly increasing the width of the nMOS transistor, connected between the output and the ground, will cause more current to flow from output node to the ground, therefore dissipating more energy.

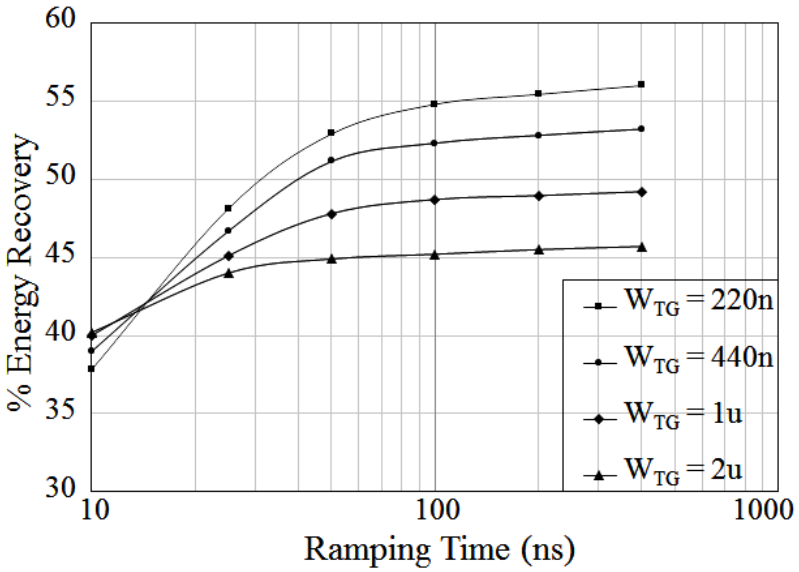


Figure 3.8: Energy recovery vs ramping time at different TG widths.

The simulations were performed at the C_T/C_L ratio of 10. Figure 3.8 shows the relationship between ramping time and percentage energy recovery at different TG widths (W_{TG}). The

graph also shows that for the ramping time of 10ns, the percentage energy recovery improves as the width of the transmission gate is increased from W_{TG} =220nm to W_{TG} =2µm. The shorter ramping time affects the switching time of the transistor; as a result, the load capacitance does not charge to the required level of voltage, causing a reduction in the energy supplied and recovery and thus, more energy dissipation. Therefore, for shorter ramping times, the energy recovery improves for increased TG width. For ramping times, 25ns and above, the switching time is enough to charge/discharge the load capacitance at minimum TG width. For this reason, the energy recovery improves as the width of the TG is reduced. Also, smaller TG width, cause the current to decrease which in turn reduces the dissipation in the switch (TG). The improvement in energy recovery for each TG width is significant for the ramping time, from 10ns to 50ns, in comparison to the ramping time from 100ns to 400ns.

3.5.4 Impact of Supply Voltage Scaling on Percentage Energy Recovery

For energy efficient operation, the influence of the supply voltage scaling on energy recovery is worth investigating. The simulations were performed at simulated ramping times keeping the C_T/C_L ratio at 10. The supply voltage was scaled from 1.8V down to 0.7V. It was not scaled down below 0.7V because the threshold voltage of the transistors used is approximately 0.5V. Figure 3.9 shows the relationship between supply voltage and energy recovery at simulated ramping times. The plot also shows that the energy recovery decreases as the supply voltage is scaled down from 1.8V to 0.7V for all the ramping times. It is because as the power-supply is scaled down, the ON-resistance of the charging path increases. In the conventional CMOS, the energy dissipation does not depend on the ON-resistance however, in the adiabatic logic, the energy dissipation depends on the ON-resistance of the charging path. As the supply voltage is scaled down, the ON-resistance (R_{ON} α 1/ (V_{GS}-V_{th})) increases because (V_{GS}-V_{th}) decreases.

There is no significant decrease in the energy recovery for the supply voltage range 1.8V to 1V for ramping times, 200ns and 400ns, whereas the decrease in the energy recovery for the supply voltage range 1V to 0.7V is about 30% and 10% respectively. For the ramping times, less than 200ns, the decrease in the percentage energy recovery is significant over the scaled voltage range.

There is a significant decrease in the energy recovery for the range, 1V to 0.7V at all ramping times. It is because as the supply voltage is scaled down the overdrive voltage

(V_{GS}-V_{th}) is reduced, causing the increase in the ON-resistance of the charging path and in the time constant $R_{ON}C_L$ (*RC* time constant increases). This increases the energy dissipation and decreases the energy recovery. In order to increase the energy recovery, either the widths of the switches in the step charging circuit should be increased or the ramping time (time of operation) should be increased.

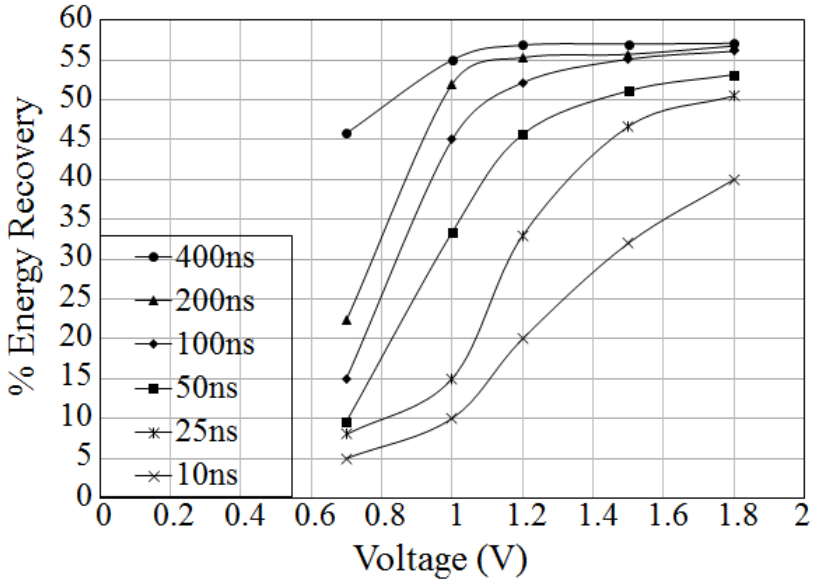


Figure 3.9: Energy recovery vs supply voltage scaling at different ramping times.

It can be seen that the decrease in the energy recovery with the supply voltage scaling is significant for the shorter ramping times (higher frequency). Therefore, as the supply voltage is scaled down, the ramping time should be increased in order to achieve high energy recovery.

3.5.5 Performance of 3, 4, 5, 6, 7 and 8-Step Charging Circuits

Next, the energy performance of the power-clock generator having more than 2-steps (3, 4, 5, 6, 7 and 8-step) was considered (for comparison point of view, the performance of 2-step charging circuit is also included in this investigation). Specifically, it was investigated that if the combined tank-capacitance to load capacitance ratio of '10' can be used in step charging circuits with more than 2-steps. Also, step charging circuits were compared in terms of their energy recovery properties and circuit complexity.

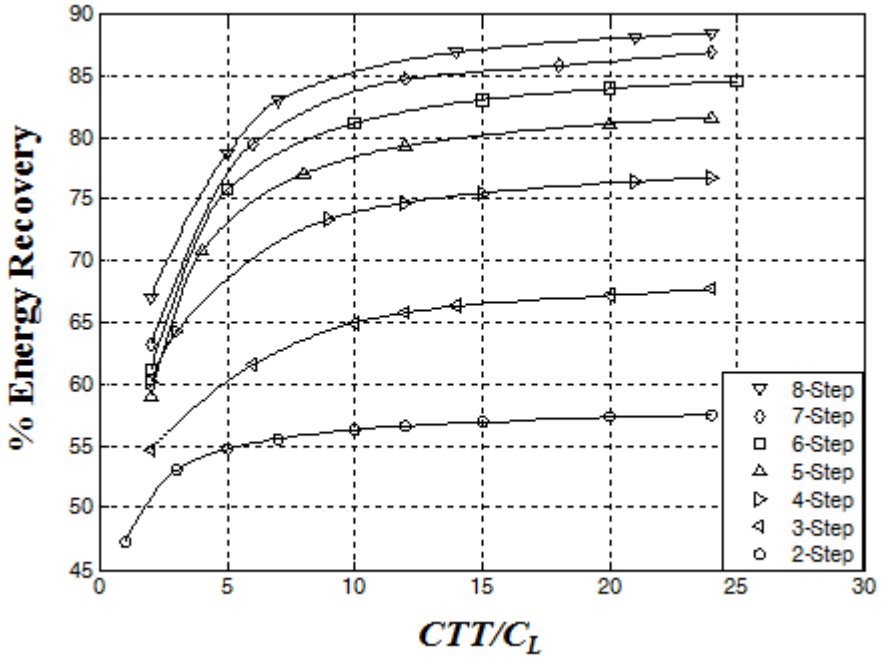
Simulations were carried out under the same environment as for the 2-step charging circuit, except that ramping times of 10ns and 25ns has not been considered as step charging circuits with a large number of steps doesn't perform well at such short ramping times. This

is because the frequency of CLK signal to FSM increases and the therefore, the switching time of the switches decreases. Energy recovery achievable by 3, 4, 5, 6, 7 and 8-step charging circuits at different CTT/C_L ratios was measured. The simulation results shown in Figure 3.10 (a), (b), (c), and (d), illustrate the relation between CTT/C_L ratios and energy recovery at all the ramping times simulated. The plots show that increasing the CTT/C_L ratio above, say, 10, offers relatively slight improvement in energy recovery. This suggests that as a design rule, a CTT/C_L ratio of 10 is appropriate in practical circuits.

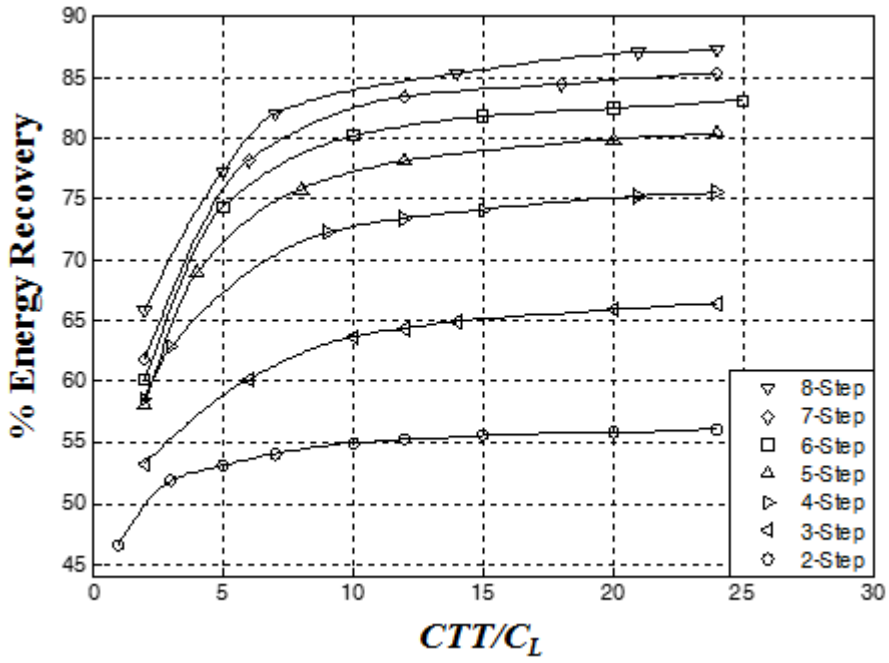
For small values of tank capacitor (in comparison to the load capacitor), the step charging circuit does not remain stable and the voltage level of the tank capacitor keeps on changing. Additionally, small values of tank capacitors cannot hold much charge and thus, cannot charge and discharge the load capacitance to the desired level and thus exhibits unequal steps in the step charging waveform and thus the adiabatic load dissipates high energy and therefore, less energy recovery.

It should also be noted that, after a particular ramping time, the improvement in the % energy recovery is not much. This is because as the ramping time is increased further, it increases the dissipation due to leakage and thus does not show any improvement in the % energy recovery.

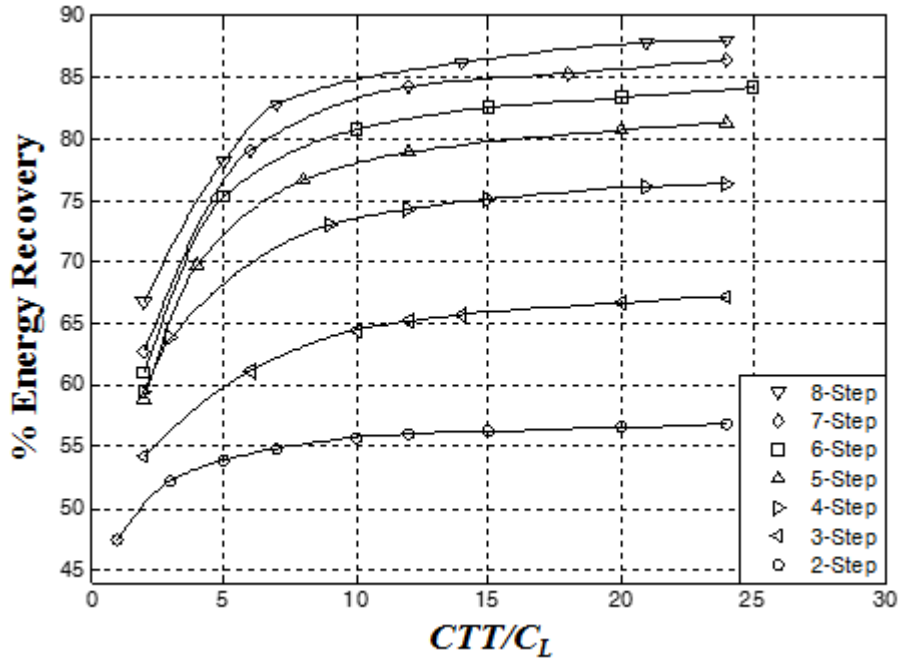
Figure 3.10 (a), (b), (c), and (d) also show the "diminishing returns" of increasing number of steps against improvements in energy recovery. Where "diminishing returns" means that the energy benefits of increasing the number of steps in the step charging circuit are less in comparison to the increase in the circuit complexity of the step charging circuit and the cost of generating the control signals for operating the switches of the step charging circuits. It shows an improvement of about 10%, 9%, 4%, 3%, 2% and 1% in energy recovery against increasing number of steps from 2 to 8 respectively at a CTT/C_L ratio of 10. The energy recovery goes through a progression of diminishing increments because from equation 3.5, it can be seen that the energy dissipation is decreased by $1/n$. This suggests that the energy dissipated by the 2, 3, 4, 5, 6, 7 and 8-step charging circuit will be 1/2, 1/3, 1/4, 1/5, 1/6, 1/7 and 1/8 of the conventional case respectively. This suggests that as the number of steps is increased from 2 to 8-steps, the energy benefits in moving from 2 to 3-steps, 3 to 4-steps, 4 to 5-steps, 5 to 6-steps, 6 to 7-steps and 7 to 8-steps goes through a progression of diminishing increments.


% Energy Recovery
90
85
80
75
70
65
60
55
50
45
0
5
10
15
20
25
30
CTT/C_L
8-Step
7-Step
6-Step
5-Step
4-Step
3-Step
2-Step

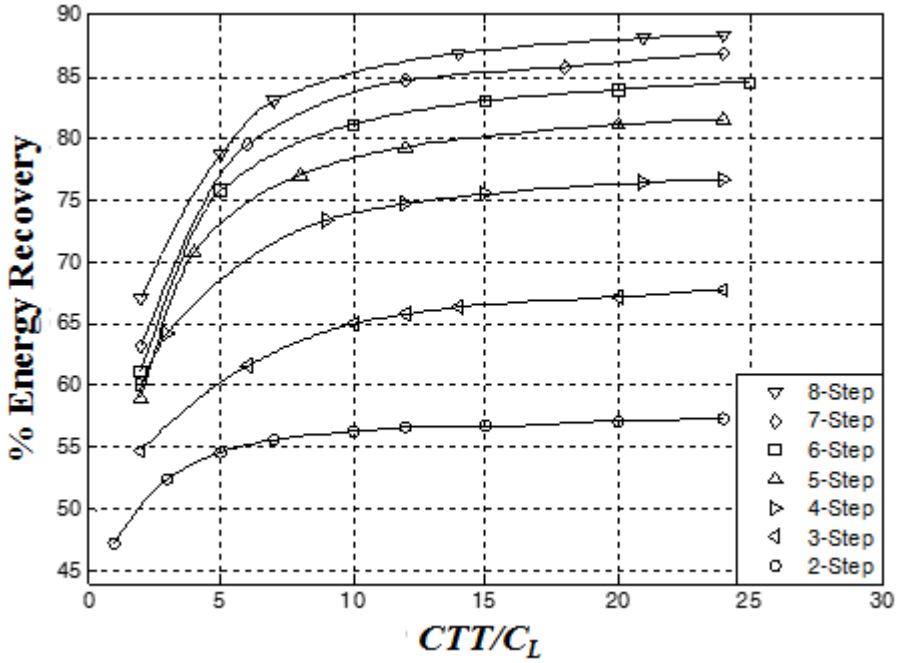
(a)


% Energy Recovery
90
85
80
75
70
65
60
55
50
45
0
5
10
15
20
25
30
CTT/C_L
8-Step
7-Step
6-Step
5-Step
4-Step
3-Step
2-Step

(b)



(c)



(d)

Figure 3.10: Energy recovery vs CTT/C_L ratio at ramping time (a) 50ns (b) 100ns (c) 200ns and (d) 400ns

The relatively small improvement between 4 and 5-step charging circuit suggests that 4-step charging circuit might be considered an adequate trade-off between complexity and energy

recovery. Also, keeping the CTT/C_L ratio of 10 ensures the stable operation of the step charging circuits for load capacitance of 1pF.

As a reminder, the reported results do not include the energy cost of operating the switches in the step charging circuit.

Because the results use "total tank-capacitance" C Total Tank (*CTT*) to load capacitor (C_L) ratios, it can be seen that step charging circuit having more number of steps with correspondingly smaller values of tank-capacitors deliver relatively better results in terms of energy recovery, in comparison to fewer, larger number of tank-capacitors in a step charging circuit having small number of steps. For e.g. according to Figure 3.10 (a), (b), (c), and (d) a 4-step charging circuit at CTT/C_L ratio of 10(three tank-capacitors) gives better energy recovery compared to the 2-step charging circuit with a CTT/C_L ratio of 10 (one tank-capacitor). This illustrates, that the tradeoffs are best decided on CTT/C_L ratios and also on the number of steps in the step charging circuit. Such a strategy has the advantage that the amount of silicon area dedicated to tank-capacitors in the step charging circuit/power-clock generator can remain largely constant regardless of the number of steps.

Energy recovery by the adiabatic load depends on the CTT/C_L ratio, as for small ratios, the energy dissipation increases, and the energy recovery decreases due to the unequal step sizes.

3.5.6 Energy Recovery vs Ramping Time

Energy performance of adiabatic circuits is additionally a function of ramping time. Figure 3.11 compares energy recovery achievable by 2, 3, 4, 5, 6, 7 and 8-step charging circuits at ramping times of 50ns, 100ns, 200ns and 400ns at a CTT/C_L ratio of 10. Figure 3.11 also shows that as the ramping time is increased above 100ns, the improvement in energy recovery is relatively small. But as the ramping time is reduced from 100ns to 50ns there is a decrement in energy recovery by approximately 3%. Adiabatic losses dominate the energy dissipation at shorter ramping times (higher speed) so energy recovery decreases.

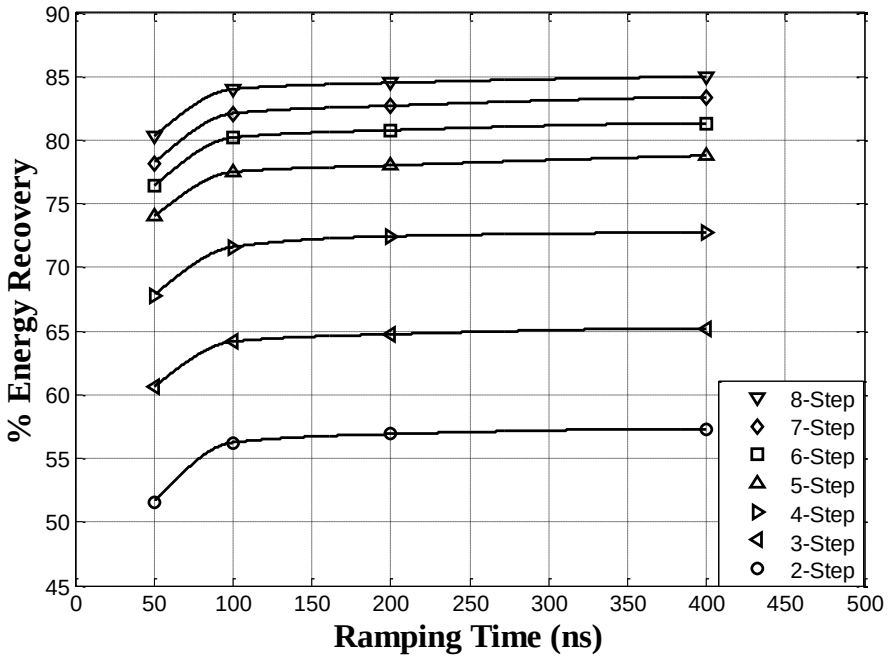


Figure 3.11: Energy recovery vs ramping time

3.6 Chapter Summary

In summary, the generation of power-clocks in adiabatic integrated circuits is investigated. Specifically, stepwise charging strategies (2, 3, 4, 5, 6, 7, and 8-step) based on tank-capacitor circuits are considered, comparing them in terms of their energy recovery properties and circuit complexity.

Impact of various parameters such as tank-capacitance to load capacitance ratio, ramping time, transistor widths and power supply scaling on the energy recovery achievable in the 2-step charging circuit was investigated. The simulation results show that energy recovery improves if the transistors widths in the step charging circuit are sized at their minimum dimensions. Also, energy recovery decreases as the power supply voltage was scaled down. Specifically, the decrease in the energy recovery with decreasing power supply was significant for shorter ramping times (higher frequencies).

In addition, it was identified that the energy recovery achievable in step charging circuits also depend on the tank-capacitor size and that can be reduced as the number of steps in a step charging circuit increases concluding that combined tank capacitance (*CTT*) versus load capacitance (C_L) ratio is the significant parameter.

Tradeoffs can be made on the basis of total tank-capacitance to load capacitor (CTT/C_L) ratios. Suitable tradeoffs have been suggested -specifically that a CTT/C_L ratio of 10 with a 4-step charging circuit is appropriate, increasing either parameter yielding relatively little benefits. Also, energy performance of adiabatic circuits improves at the longer ramping time (slower speed).

4. Finite State Machine Controller

As stated in Chapter 3, step charging circuits require control signals for operating the switches in order to generate the stepwise waveform. Having more number of steps increases the energy recovery of the steps charging circuit but can considerably increase the energy dissipation of the PCG due to the increased complexity of the Finite State Machine (FSM) controller. This chapter looks at the implementation of the synchronous FSM controller for generating the control signals for the step charging circuits. In particular, the design of FSM controller for single channel and 4-phase PCG using step charging circuit is considered and a comparison is made on the basis of circuit complexity and energy dissipation. The chapter ends proposing the design rules and trade-offs between the complexity of the FSM controller and the energy benefits in the single channel and 4-phase PCG using n-step charging circuit.

4.1 Introduction

The generation of the control signals for operating the switches of the step-charging circuits is also a source of energy consumption in an adiabatic system. For an n-step charging circuit, n+1 control signals are required. The control signals are generated using Finite State Machine (FSM) controller designed with conventional CMOS. Also, depending on the adiabatic logic family, one or more than one power-clocks are required to operate adiabatic gates in cascade [33]. Specifically, for a 4-phase adiabatic logic design, 4-phases of the power-clock, each having a phase difference of 90^o are required to be generated using four n-step charging circuits. In order to control the switches of four n-step charging circuits, a total of 4(n+1) control signals need to be generated. This suggests that having a multiphase requirement in the power-clock generator increases the complexity of the FSM controller many folds and makes the design of an energy efficient Power-Clock Generator (PCG) a challenging problem. Different topologies for generating the step

charging waveform have been proposed in the past [48]-[54] however, the energy dissipated by the controller in generating the control signals for the step charging circuits has not been reported.

As mentioned before, in literature, most attention has been given to the performance of the adiabatic core without considering the performance of the power-clock generator [46]. None of the papers in the literature have shown a number of steps in the PCG which presents an appropriate trade-off between circuit complexity and energy performance. This is the first, where a detailed investigation is performed which includes energy dissipated by the FSM controllers for PCG using 2, 3, 4, 5, 6, 7, and 8-step charging circuits.

4.2 FSM Controller for 2-Step Charging Circuit

A methodology based on state machine to generate the control signals are used here. Clocked with a signal 'CLK', the state machine will step through the states. A general block diagram of 4-phase power-clock generator using 2-step charging circuit is shown in Figure 4.1. For generating the 4-phases of the 2-step charging power-clock, four, 2-step charging circuits and an FSM controller is required.

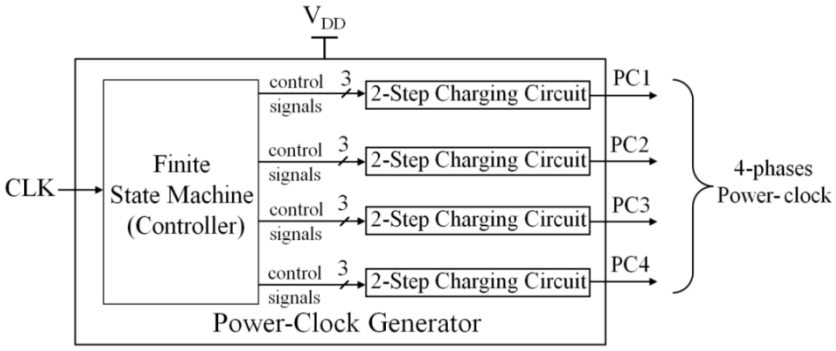


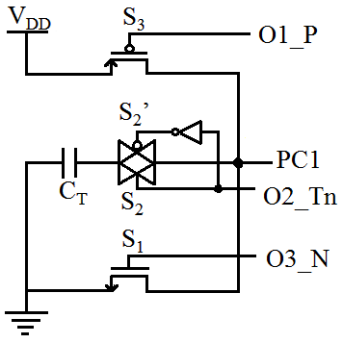
Figure 4.1: 4-phase PCG using the 2-step charging circuit

As step charging circuits have already been discussed in Chapter 3, this chapter will only be devoted to FSM controller.

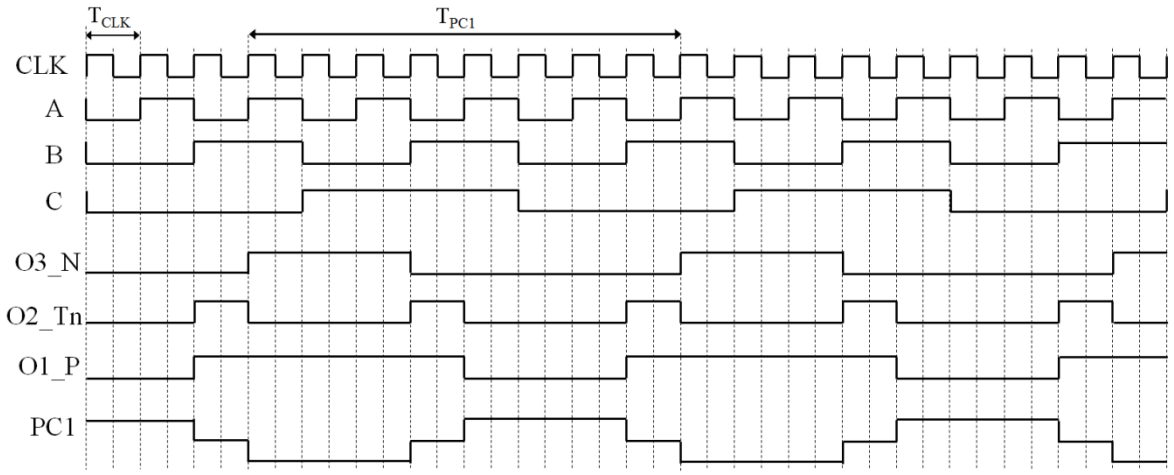
4.2.1 Timing Diagram

A 2-step charging circuit and the timing diagram of the control signals for single-phase power-clock (PC1) is shown in Figure 4.2 (a) and (b) respectively. A 2-step charging circuit requires 3 control signals (O1_P, O2_Tn, and O3_N, where P stands for pMOS and Tn and N stands for nMOS transistors) for generating single phase power-clock.

From Figure 4.2 (a) it can be seen that the control signal O1_P, turns ON the pMOS transistor switch, S_3 connected to the power supply, V_{DD}. At this time, switches S_2, S_2' and S_1 are OFF. When the switch, S_3 is turned OFF, control signal O2_Tn and its complement signal turn ON the transmission gate switches, S_2 and S_2'connected to the tank capacitor, C_T. The control signal for switch S_2' is the complement of control signal O2_Tn and is generated by passing O2_Tn through static CMOS inverter. While the switches S_2 and S_2'are ON, switches S_3 and S_1 are OFF. Next, control signal O3_N turns ON the nMOS transistor switch, S_1 connected to the ground. During this time, switches S_2, S_2' and S_3 are OFF. This sequence of turning ON the switches momentarily in forward and reverse manner produces a step-like waveform as shown in Figure 4.2 (b) and is named as PC1 (power-clock phase 1). The time period, T_{PC1} denotes one power-clock cycle which is eight times the clock cycle, T_{CLK}.



(a)



(b)

Figure 4.2: (a) 2-step charging circuit [44] (b) Timing diagram of the control signals for generating single phase power-clock

4.2.2 FSM Controller circuit

In order to generate the control signals O1_P, O2_Tn, and O3_N, an FSM control circuit was required. To realize a power-clock cycle using 2-step charging circuit, eight states of the FSM were required. Eight states were required to ensure that evaluation, hold, recovery and idle phase of the power-clock are equal in order to observe the adiabatic principle as stated in Chapter 2 of this thesis. Evaluation and recovery phases are equal due to the equal number of steps during rising and falling. However, to make hold and idle phase equal to evaluation and recovery phase, more states in the FSM were required. Using the timing diagram of Figure 4.2 (b) state diagram was drawn for the single phase power-clock (PC1) and is shown in Figure 4.3. O1_P is an active low signal as it is the input to the pMOS transistor, whereas O2_Tn and O3_N are active high signals. It should be noted that no dead time is required between the switches being turned ON.

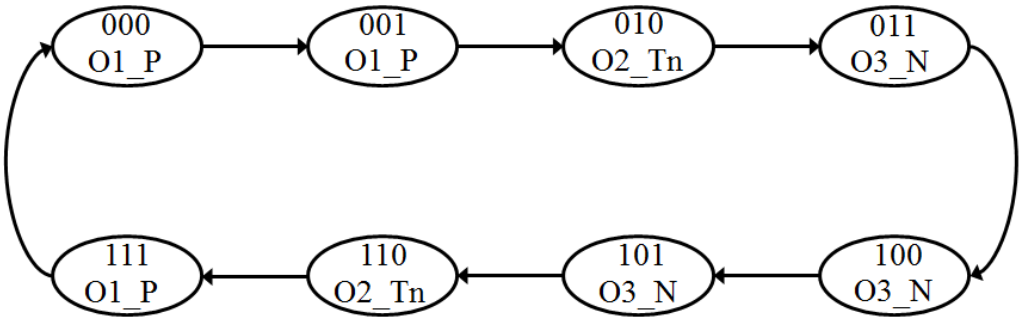


Figure 4.3: State diagram for generating control signals for single phase power-clock

The state diagrams and state tables were drawn for all the 4-phases of the 2-step charging power-clock. Table 4.1 shows the state table for generating the control signals for 4-phases of the power-clock (denoted by PC1, PC2, PC3 and PC4). The control signals for generating PC2 will have a phase difference of 90^{o} compared to the control signals of PC1. Similarly, control signals for generating PC3 will have a phase difference of 90^{o} compared to the control signals of PC2 and so on.

Table 4.1: State table for generating 4-phases of 2-step charging power-clock.

States	PC1			PC2			PC3			PC4		
CBA	O1_P	O2_Tn	O3_N	O1_P	O2_Tn	O3_N	O1_P	O2_Tn	O3_N	O1_P	O2_Tn	O3_N
000	0	0	0	1	1	0	1	0	1	1	1	0
001	0	0	0	1	0	1	1	0	1	0	0	0
010	1	1	0	1	0	1	1	1	0	0	0	0
011	1	0	1	1	0	1	0	0	0	0	0	0
100	1	0	1	1	1	0	0	0	0	1	1	0
101	1	0	1	0	0	0	0	0	0	1	0	1
110	1	1	0	0	0	0	1	1	0	1	0	1
111	0	0	0	0	0	0	1	0	1	1	0	1

From the state table, the Karnaugh-map was formed and solved to obtain the logical expressions for the control signals for each power-clock phase. Table 4.2 shows the logical expressions for control signals generating 4-phase power-clock using 2-step charging circuit.

Table 4.2: Logical expressions for control signals generating 4-phase power-clock using 2-step charging circuit.

	PC1	**PC2**	**PC3**	**PC4**
O_P	C’B+CB’+BA’	C’+B’A’	C’B’+CB+BA’	C+B’A’
O2_Tn	BA’	B’A’	BA’	B’A’
O3_N	(C’B’+CB+BA’)’	(C+B’A’)’	C’B’+CBA	(C’+B’A’)’

Figure 4.4 shows the circuit for FSM controller generating the control signals for single-phase (PC1) PCG using the 2-step charging circuit. Eight states of the FSM controller were generated using three True Single-Phase Clock, TSPC D flip-flops. CLK denotes the clock signal to the TSPC D flip-flop. This signal is responsible for setting the frequency of the generated power-clock. Here the CLK frequency, f_{CLK} is eight times the frequency of the power-clock, $f_{PC.}$

$$f_{CLK} = 8.f_{PC} \tag{4.1}$$

The outputs A and A’ forms the Least Significant Bits (LSB) whereas, C, and C’ forms the

Most Significant Bits (MSB) of the states in the state table. Using the three outputs and their complementary signals from the flip-flops, the control signals O1_P, O2_Tn, and O3_N for each power-clock phase were generated. It should be noted that the counter is used to generate the control signals. The signals were not produced by the direct encoding of FSM because to realize 4-phases of the 2-step charging power-clock, 8 states of the FSM controller were required. To generate the eight states of the FSM, pulses of different pulse width were required.

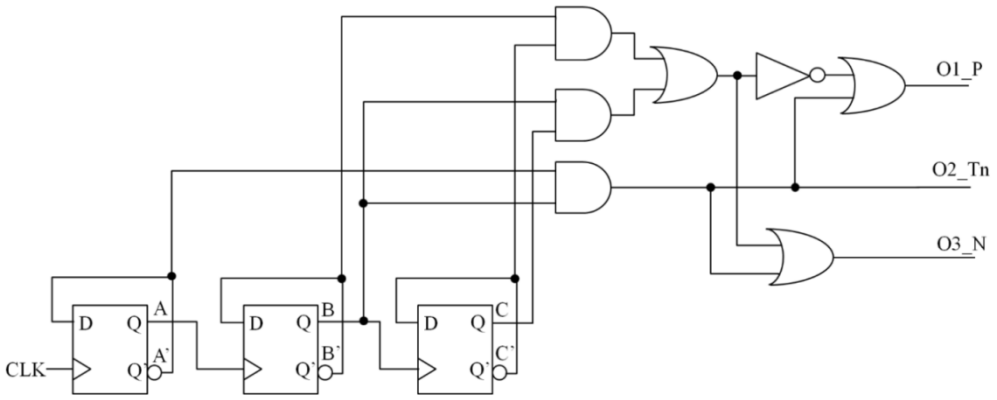
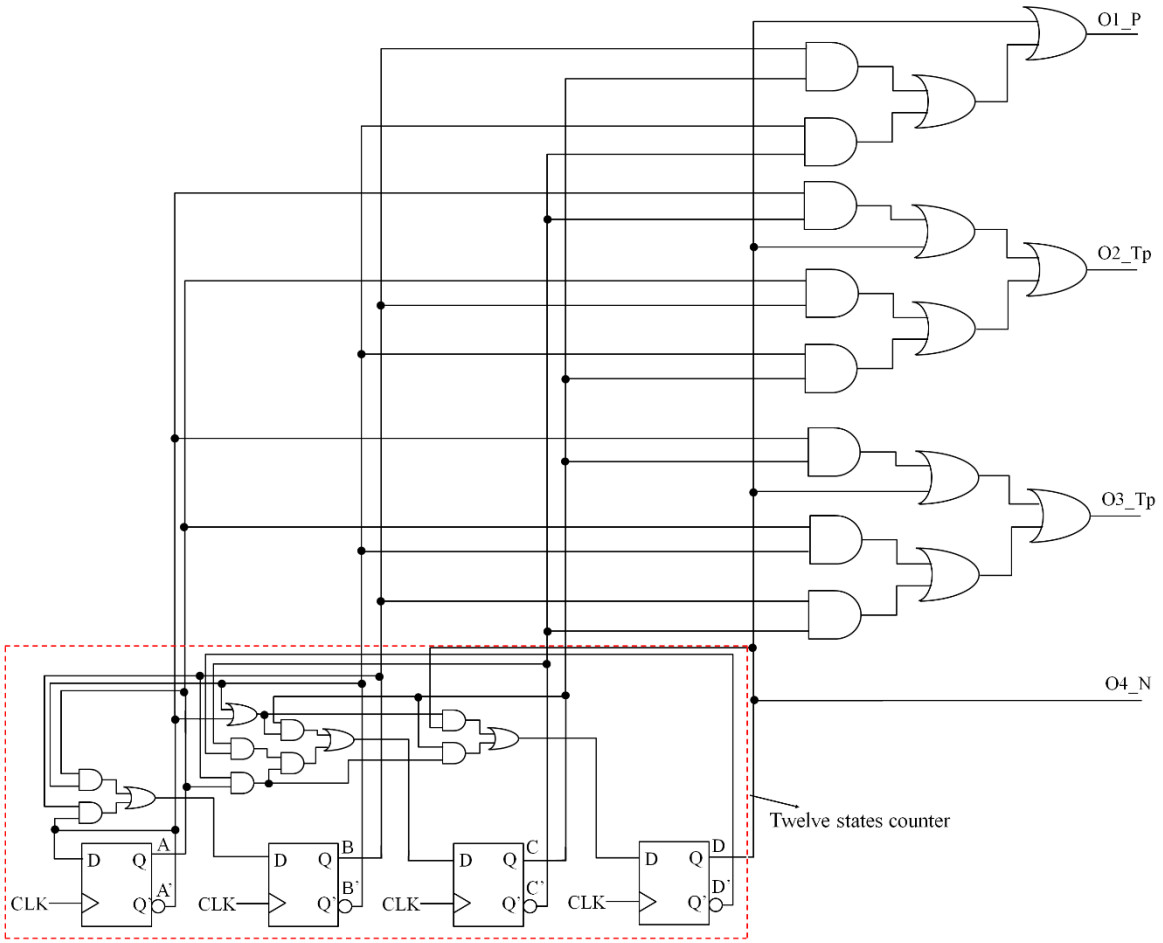
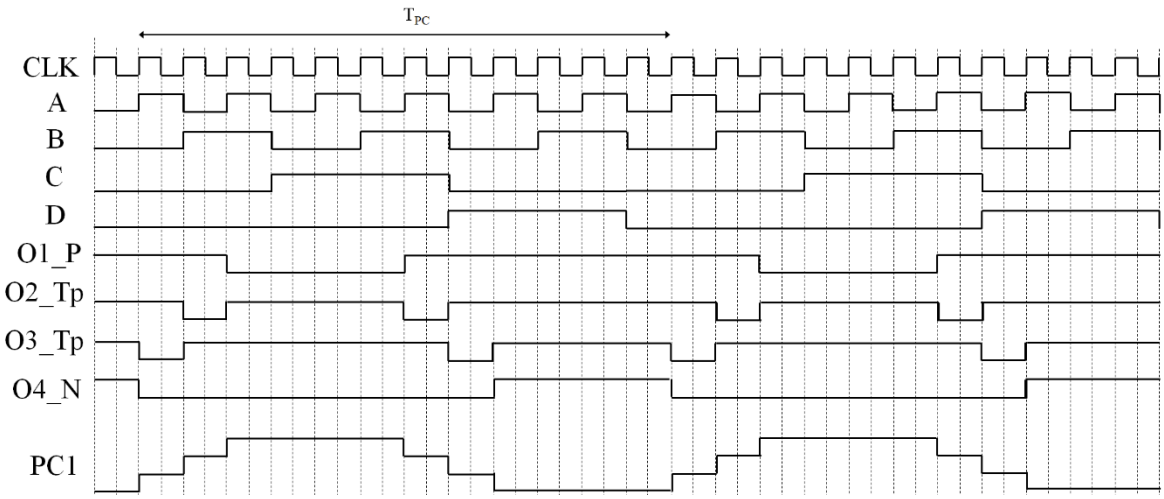


Figure 4.4: FSM controller for single phase PCG

The controller circuit and the timing diagram for generating single phase power-clock using 3-step charging circuit (PC1) are shown in Figure 4.5 (a) and (b) respectively. To construct the FSM controller circuit for 4-phase PCG using 3-step charging circuit, the same procedure as mentioned for the FSM controller circuit for 4-phase PCG using 2-step charging circuit using state diagram, state table, and Karnaugh-map was adopted. For a 3-step charging circuit, four control signals need to be generated. For realizing a 3-step charging waveform, twelve states were required in the FSM. For generating twelve states, a total of four TSPC D flip-flops, four OR gates and eight AND gates were required in the counter design as shown in Figure 4.5 (a). The outputs A and A' forms the Least Significant Bits (LSB) whereas, D, and D' forms the Most Significant Bits (MSB) of the states of the state table. Using the four outputs and their complementary signals from the flip-flops, the control signals, O1_P, O2_Tp, O3_Tp and O4_N for each power-clock phase were generated. Where P and Tp stand for the pMOS transistor and N stand for the nMOS transistor. Inverters were added as the delay elements for avoiding the glitches (arising due to the different path delays) in the control signals. The glitches cause current spikes and degrade the energy efficiency of the power-clock generator. For the simplicity, inverters (delay elements) are not shown in Figure 4.5 (a). Here the CLK frequency, f_{CLK} is twelve times the frequency of the power-clock, $f_{PC.}$



(a)



(b)

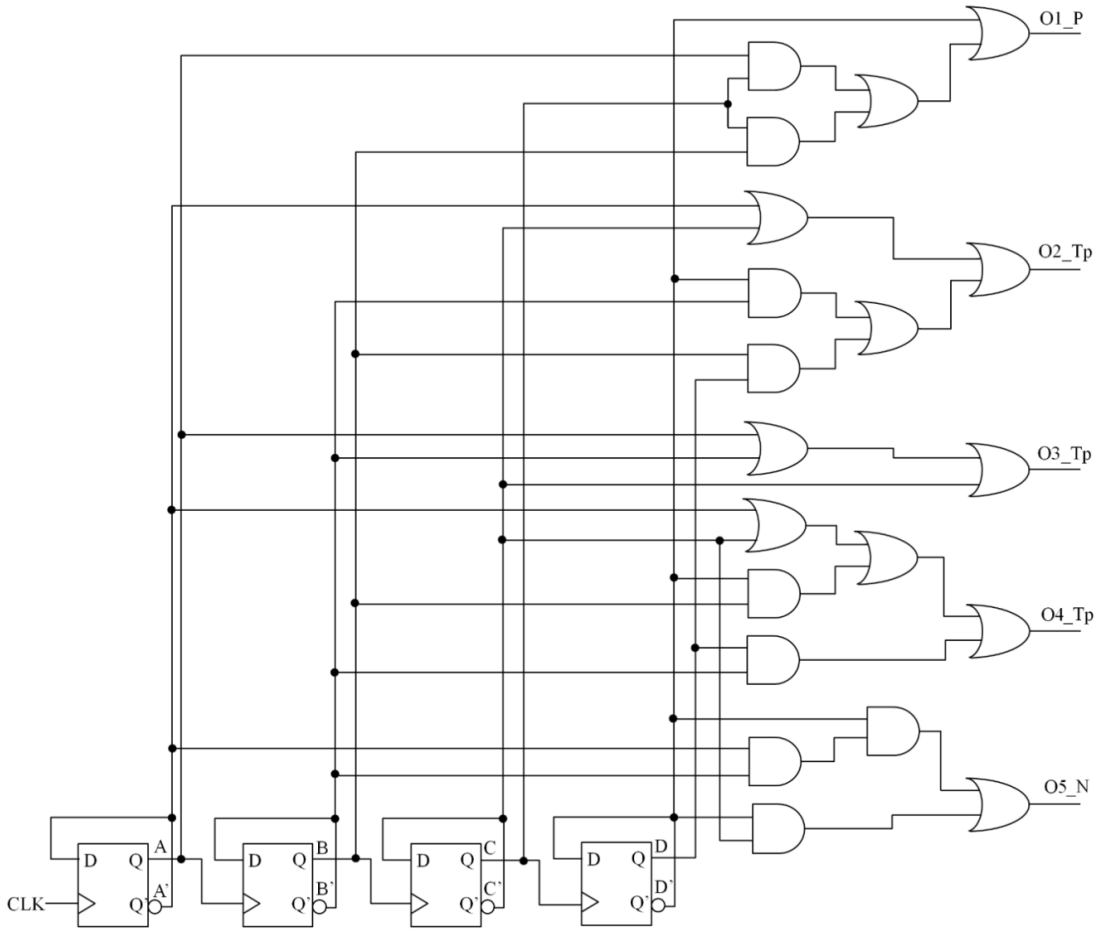
Figure 4.5: (a) FSM controller for 3-step charging circuit (b) Timing diagram of control signals for generating single phase power-clock

The logical expressions for generating 4-phase power-clock using 3-step charging circuit are shown in Table 4.3.

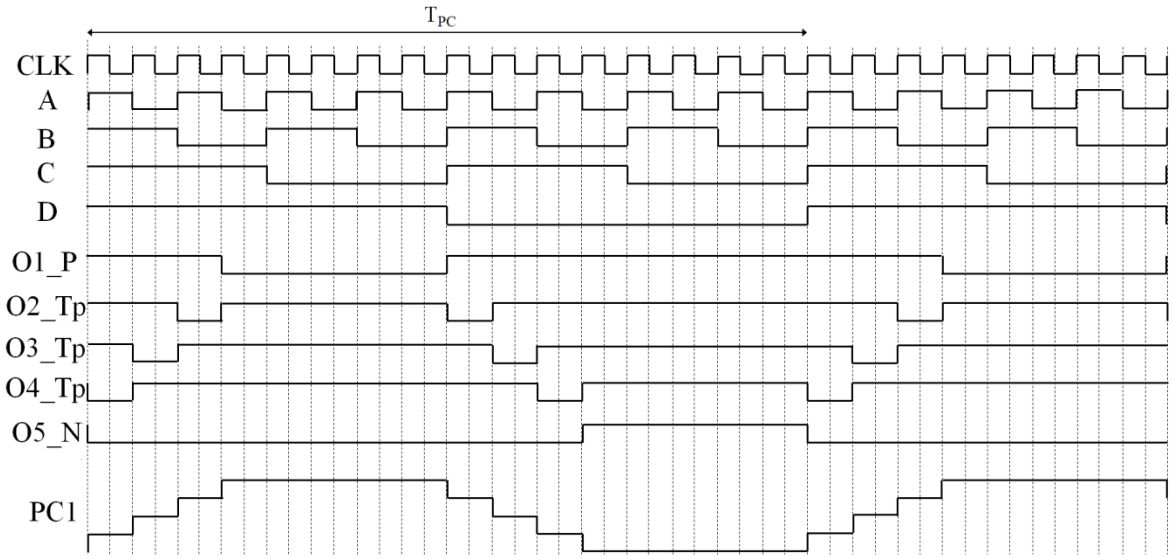
Table 4.3: Logical expressions for control signals generating 4-phase power-clock using 3-step charging circuit.

	PC1	**PC2**	**PC3**	**PC4**
O1_P	D+C'B'+CB	D'B'A'+C'(A+B)	D'	C+D'BA+D(B'+A')'
O2_Tp	D+C'A'+CB'+AB	B+C'A'+D'A	D+C'B+CA'+B'A	C+B'+DA+D'A'
O3_Tp	D+CA'+B'A+C'B	C+B'+D'A'+DA	D+C'A'+CB'+BA	B+C'A'+D'A
O4_N	D	D'C'(A'+B')+DBA	CB'+D'C'B	DB'A'+CB+CA

Figure 4.6 (a) and (b) shows the FSM controller circuit and the timing diagram for generating single phase 4-step charging power-clock (PC1) respectively. To construct the FSM controller circuit for 4-phase PCG using 4-step charging circuit, the same procedure as mentioned for the FSM controller circuit for 4-phase PCG using 2-step charging circuit using state diagram, state table, and Karnaugh-map was adopted. The 4-step charging circuit requires five control signals. For realizing a 4-step charging power-clock, sixteen states in the FSM controller were required. For realizing sixteen states, four flip-flops were used. Here the CLK frequency, f_{CLK} is sixteen times the frequency of the power-clock, $f_{PC.}$ The outputs A and A' forms the Least Significant Bits (LSB) whereas, D, and D' forms the Most Significant Bits (MSB). Using the four outputs and their complementary signals from the flip-flops, the control signals, O1_P, O2_Tp, O3_Tp, O4_Tp, and O5_N for each power-clock phase were generated. For the simplicity, inverters (delay elements) are not shown in Figure 4.6 (a).The logical expressions for generating 4-phases of the power-clock using 4-step charging circuit are shown in Table 4.4.



(a)



(b)

Figure 4.6: (a) FSM controller for 4-step charging circuit (b) Timing diagram of control signals for generating single phase power-clock

Table 4.4: Logical expressions for control signals generating 4-phases of the power-clock using 4-step charging circuit.

	PC1	**PC2**	**PC3**	**PC4**
O1_P	D'+CA+CB	D'(A+B+C)+DC'	D+CA+CB	D(A+B+C)+D'C'
O2_Tp	C'+A'+D'B'+DB	C+A'+DB'+D'B	C'+A'+DB'+D'B	C+A'+D'B'+DB
O3_Tp	A+B'+C'	A+B'+C	A+B'+C'	A+B'+C
O4_Tp	C'+A'+D'B+DB'	C+A'+D'B'+DB	C'+A'+D'B'+DB	C+A'+D'B+DB'
O5_N	D'C'+D'B'A'	D'C+DC'B'A'	DC'+DB'A'	DC+D'C'B'A'

For more than 4-steps in the step charging circuit, FSM controller will require five flip-flops to realize 20, 24, 28 and 32 states for 5, 6, 7, and 8-step charging circuits respectively. This suggests that complexity of the FSM controller increases as the number of steps in the step charging circuit is increased.

Tables 4.5 and 4.6 summarize the number of components required to design FSM controller for generating single channel PCG using 2, 3, 4, 5, 6, 7, and 8-step charging circuits and 4-phase PCG using 2, 3, and 4-step charging circuits. 4-phase PCG using 5, 6, 7, and 8-step charging circuits were not considered because in chapter 3 it was proposed that 4-step charging circuit presents an appropriate trade-off between circuit complexity and energy performance and going from 4-step to 5-step yields little benefits in terms of energy recovery.

From Table 4.5, for single phase PCG, the number of components required in the FSM controller increases as the number of steps is increased. In particular, the number of components in the FSM controller for 3-step charging circuit increases significantly compared to the 2-step charging circuit. The increase in the number of components is not significant as moving from the FSM controller for 3 to 8-step charging circuits. Similarly, from Table 4.6, it can be seen that the number of components in the FSM controller for 4-phase PCG using 4-step charging circuit increases considerably in comparison to FSM controller for 4-phase PCG using 2-step charging circuit.

Table 4.5: Characterization of the FSM controller for single phase PCG using n-step charging circuits.

No. of Gates	2-step	3-step	4-step	5-step	6-step	7-step	8-step
NOT	1	4	24	38	52	64	74
OR2	3	12	11	15	18	22	26
AND2	3	16	9	12	12	16	18
FF	3	4	4	5	5	5	5

Table 4.6: Characterization of the FSM controller for 4-phase PCG using 2,3 and 4-step charging circuits.

No. of Gates	2-step	3-step	4-step
NOT	19	8	56
OR2	5	26	37
AND2	4	30	22
FF	3	4	4

4.3 Simulation Results

For this investigation, 180nm CMOS process at 1.8V power supply was used. Simulations were performed using Spectre simulator in Cadence EDA tool at 'Typical-Typical (TT) process corner. Simulations were carried out at 13.56MHz power-clock frequency, f_{PC} for the capacitive load, C_L of 1pF. Due to the different number of steps in the step charging circuits, the CLK frequency is different for each of the n-step charging PCG. The frequency of the clock signal, CLK is calculated as:

$$f_{CLK} = N.f_{PC} \qquad (4.2)$$

Where, N is the number of states in the FSM controller. For the power-clock frequency of 13.56 MHz, the clock frequencies for n-step charging circuit were calculated using 4.2. Center frequency of the smart card applications is 13.56 MHz. Therefore, the simulations were performed at 13.56MHz.

The energy consumed per cycle by the FSM controller in generating a single and 4-phase

power-clock was measured for the time period denoted as $T_{PC,\ 1\text{-}\Phi,}$ and $T_{PC,\ 4\text{-}\Phi}$ (in Figure 4.7) respectively. $T_{PC,\ 1\text{-}\Phi,}$ is the time period for one power-clock phase. $T_{PC,\ 4\text{-}\Phi}$ on the other hand, is the time period that includes four phases of the power-clock (from the start of the power-clock phase 1 till the end of the power-clock phase 4). The impact of step charging circuit switch width and voltage scaling on the energy performance of the FSM controller was also investigated. FSM controller for single phase PCG using 2, 3, 4, 5, 6, 7, and 8 step charging circuits and 4-phase PCG using 2, 3, and 4-step charging circuits were implemented and compared on the basis of circuit complexity and energy performance.

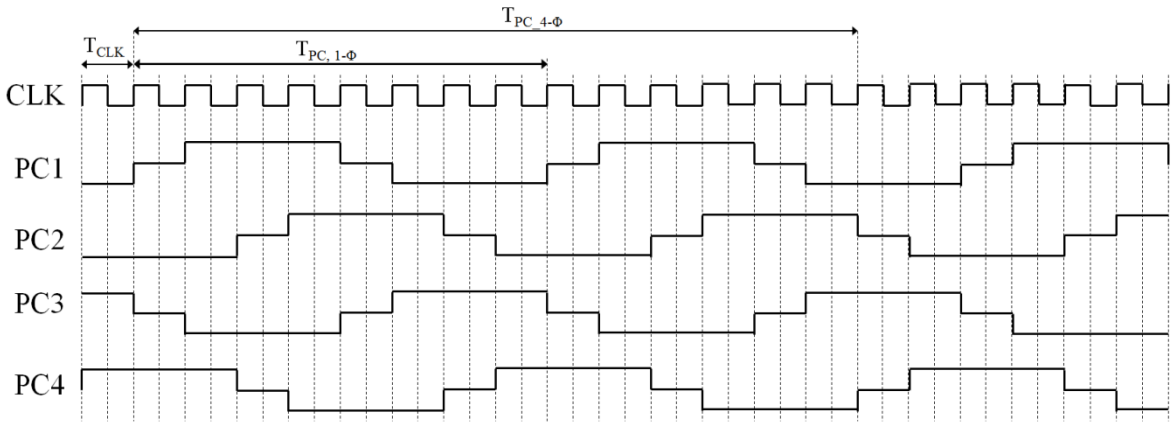


Figure 4.7: Time interval ($T_{PC,\ 1\text{-}\Phi,}$ and $T_{PC,\ 4\text{-}\Phi}$) of energy dissipation per cycle of FSM controller for single and 4-phase PCG using 2-step charging circuit.

4.3.1 Energy consumption of the FSM controller for single channel and 4-phase PCG

Figure 4.8 shows the block diagram of a 4-phase PCG having FSM controller and four n-step charging circuits. This setup was used to measure the energy dissipated by the FSM controller. The FSM controller generates the control signals for driving the switches of the four n-step charging circuits. Each step charging circuit is connected to the load capacitance, C_L of 1pF. PC1, PC2, PC3, and PC4 represent the 4-phases of the n-step charging power-clock. The value of the total tank capacitance (*CTT*) in the step charging circuits was set at 10pF. Sizes of the switches in the step charging circuits were set at 4µm for pMOS and 2µm for nMOS.

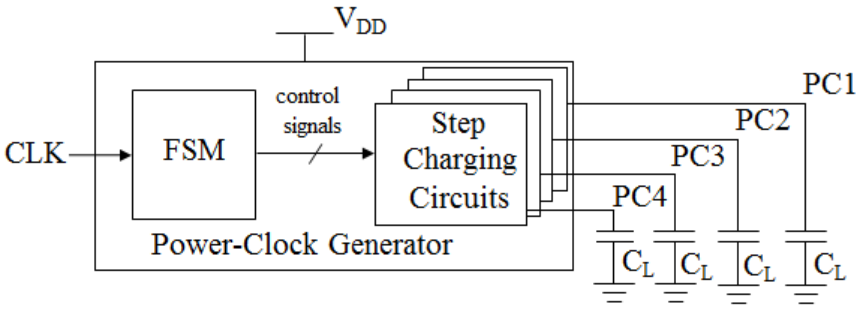


Figure 4.8: Setup for measuring energy dissipation of the FSM controller for 4-phase PCG

Table 4.7 illustrates the energy dissipated by the FSM controller for single channel PCG using 2, 3, 4, 5, 6, 7, and 8-step charging circuit. It can be seen that the energy dissipation of the FSM controller increases as the number of steps in the step charging circuit is increased. The increase in energy dissipation of the FSM controller for 5-step charging circuit and above is significant compared to 4-step charging circuit and below. This suggests that the energy benefits obtained by the increased number of steps diminish due to the energy dissipation of the FSM controller. This will increase the overall energy of the system, therefore, degrading the energy benefits obtained by the application of adiabatic technique.

Table 4.7: Energy consumption of the FSM controller for single channel PCG using n-step charging circuits.

	Energy consumption per power-clock cycle (pJ) @ 13.56MHz						
Steps	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Controller	1.082	3.82	4.69	11.14	19.52	30.44	41.99

Table 4.8 shows the energy dissipation per cycle by the FSM controller for the 4-phase PCG using 2, 3, and 4-step charging circuits. It can be seen that the energy consumption follows the same trend as shown in Table 4.7.

Table 4.8: Energy consumption of the FSM controller for 4-phase PCG using 2, 3 and 4-step charging circuit.

	Energy consumption per power-clock cycle (pJ) @ 13.56MHz		
Steps	**2**	**3**	**4**
Controller	3.542	15.06	18.19

The simulation results also suggest that PCG using more than 4-steps doesn't seem promising for energy and area efficient implementation of the adiabatic system. Also, PCG using 3 and 4-step charging circuits present an appropriate trade-off between circuit complexity and energy performance.

Although the use of n-step charging circuit reduces the energy dissipation of the step charging circuit to 1/n (theoretically), the energy dissipation of the FSM controller for generating n+1 and 4(n+1) control signals for single and 4-phase PCG respectively increases significantly, increasing the overall energy of the adiabatic system.

Therefore, for a thorough investigation of the energy performance of the complete adiabatic system, the energy dissipated by the PCG (FSM controller and step charging circuit) should be considered.

4.3.2 Impact of step charging circuit switch widths on the energy dissipation of the FSM controller

Next, the impact of step charging circuit switch widths on the energy dissipation of the FSM controller was investigated. Simulations were performed for single channel and 4-phase PCG using 2, 3, and 4-step charging circuits. The simulations were performed for five sets of the switch (pMOS and nMOS) widths shown in Table 4.9.

Table 4.9: Impact of step charging circuit switch widths on the energy dissipation of the FSM controller for single channel and 4-phase PCG using 2, 3, and 4-step charging circuits.

Switch Width	**Energy consumption per power-clock cycle (pJ) @ 13.56MHz**					
	2-Step		**3-step**		**4-Step**	
Wp : Wn	single phase	4-phase	single phase	4-phase	single-phase	4-phase
1u : 0.5u	1.014	3.231	3.696	14.190	4.548	17.180
2u : 1u	1.038	3.363	3.739	14.490	4.597	17.530
4u : 2u	1.082	3.542	3.820	15.060	4.690	18.190
8u : 4u	1.143	3.907	3.973	16.100	4.879	19.520
16u : 8u	1.216	4.658	4.264	18.010	5.238	22.120

It can be seen that the energy dissipation per cycle of the FSM controller increases as the switch sizes (transistors width) of the step charging circuit is increased. It is because increasing the switch widths, increases the capacitance of the switch and in turn load to the FSM controller. This suggests that step charging circuit switch sizes should be set to a value that allows just about enough time for charging and discharging of the load capacitor.

4.3.3 Impact of supply voltage scaling on the energy dissipation of the FSM controller

Simulations were performed to investigate the impact of supply voltage scaling on the energy dissipation of the FSM controller for single channel and 4-phase PCG using 2, 3 and 4-step charging circuit. The supply voltage was scaled from 1.8 V down to 0.8 V. The simulation results are summarised in Table 4.10. It can be seen that due to the square dependence of the supply voltage on energy, the energy dissipation per cycle of the FSM controller decreases as the supply voltage is scaled down.

Table 4.10: Impact of supply voltage scaling on energy dissipation of the FSM controller for single channel and 4-phase PCG using 2, 3, and 4-step charging circuit.

Supply Voltage	**Energy consumption per power-clock cycle (pJ) @ 13.56MHz**					
	2-Step		**3-step**		**4-Step**	
V_{DD} (V)	Single-Phase	4-phase	Single-Phase	4-phase	Single-Phase	4-phase
1.8	1.082	3.542	3.820	15.060	4.690	18.190
1.5	0.707	2.364	2.057	10.090	3.571	12.610
1.2	0.411	1.369	0.715	5.746	1.990	6.929
1	0.268	0.851	0.464	2.587	0.878	3.678
0.8	0.167	0.546	No Step	No Step	No Step	No Step

The steps in the step charging waveform were unequal for both charging and discharging when the supply voltage was scaled down. Also, at the supply voltage, 0.8V, both single and 4-phase PCG using 3 and 4-step charging circuit failed to deliver proper steps. However, to make the PCG work at 0.8V, the frequency of the CLK input to the FSM controller needs to be decreased.

4.4 Chapter Summary

The design of FSM controller for single and 4-phase PCG using step charging circuits is illustrated. A comparison of the area (in terms of logic gates), circuit complexity and energy dissipation of FSM controller for single channel PCG using 2, 3, 4, 5, 6, 7 and 8-step charging circuits and 4-phase PCG using 2, 3, and the 4-step charging circuit is presented. Also, the impact of step charging circuit switch widths and supply voltage scaling on the energy dissipation of the FSM controller for single and 4-phase PCG using 2, 3, and the 4-step charging circuit are illustrated. The Simulation results suggest that energy dissipation per cycle of the FSM controller increases as the step charging circuit switch widths are increased. Also, energy dissipation per cycle decreases as the supply voltage is scaled down. Overall, single and 4-phase PCG using 3 and 4-step charging circuit are promising in comparison to the PCG with a larger number of steps due to the increased circuit complexity and energy dissipation. However, which step charging circuit out of 3 or 4 will constitute an optimum setup for the PCG will depend on the energy benefits obtained by moving from 3-step to 4-step charging circuit and on the energy overhead of FSM controller for 3 and 4-step charging circuit. This implies that if the energy benefits of moving from 3 to 4-step charging circuit are less in comparison to the increase in the energy dissipation of the FSM controller for 4-step charging circuit, the 3-step charging circuit will be the optimum setup for the PCG.

5. Montgomery Multiplier in Adiabatic Logic

This chapter looks into the energy efficient adiabatic implementation of Montgomery multiplier and reviews several presently known implementations of Montgomery multiplier which apply energy reduction techniques. Several architectures for Montgomery multiplier using PFAL are implemented and compared. An efficient strategy to reduce the overhead due to synchronization buffers in adiabatic logic implementation is proposed.

A design methodology for scalable, area and energy efficient iterative approach architecture using 4-phase adiabatic logic is presented. In addition, Montgomery multiplication algorithm is modified. Lastly, power-gating is applied in the Montgomery multiplier architecture. A problem due to the application of power-clock gating in cascade stages is identified and a solution is proposed. A detailed performance evaluation of all the architectures is performed.

5.1 Introduction and Background

As outlined in Chapter 1 of this thesis, Montgomery multiplication [2] is an efficient method for performing modular multiplication with an arbitrary modulus, M. The algorithm replaces division by M, with simple addition and shift operations, which are particularly suitable for implementation in hardware.

In the literature, there are several papers that address the issue of a hardware implementation of Montgomery multipliers for operands of thousands of bits [55]-[58], [60]. Researchers have developed a lot of techniques to optimize the architectures for Montgomery multiplier, including, systolic arrays, scalability, and high-radix.

Systolic array architectures are less complex and are implemented to speed up the modular multiplication. These architectures offer a Processing Elements (PE) array where each PE

performs arithmetic addition and multiplication. Depending on the number of bits used, the architecture can employ a high number of PEs.

Scalability is an important feature of any architecture. Scalable architecture for the Montgomery multiplier was first introduced by A. Tenca, and C. Koc [61]. They introduced what is called word-based (scalable) Montgomery multiplication algorithm. Also, they extended their algorithm for high radix implementation. Since then, many hardware implementations have been reported dealing with scalability and high-radix features. High-radix implementations generally require a smaller number of iterations for completing one Montgomery modular multiplication than their radix-2 counterparts. However, the complex quotient determination and updating the intermediate results within one cycle may lead to circuit complexity.

An important feature that has not been considered much is the energy consumption/energy efficiency of the Montgomery multiplier. There are fewer papers which apply energy reduction techniques in their architectures. In [59] the authors attempt to reduce the energy consumption of Carry Save Adders (CSA) and registers in the CSA-based Montgomery multiplier by modifying the architecture to bypass the iterations that perform superfluous carry save addition and register write operations in the add-shift loop thus, saving the energy. In [62]-[65], the authors have presented a low power technique known as "glitch blockers" to reduce the glitches, thus reducing the power consumption. The glitch blockers are the latches and the outputs are forced to pass through them in order to avoid the glitches. In [66] a latch-based clock gating technique is used to reduce the energy consumption of the Montgomery multiplier. In [67] authors suggest a new clocking scheme where both positive and negative edge-triggered D flip-flops are used to exploit the rising and falling edge of the clock. This reduces the number of clock cycles and therefore, the switching activity which leads to a reduced power consumption.

All the above-cited references so far [59], [62]-[67] workaround using a low power technique such as glitch blockers, clock-gating and modifying the Montgomery algorithm to bypass the superfluous iterations. To the best of the author's knowledge, this is the first where Montgomery multiplier using adiabatic logic is implemented. Systolic and Iterative architectures using PFAL adiabatic logic are implemented to find an energy efficient architecture suitable for adiabatic implementation. To further reduce the energy dissipation, power-clock gating is applied and will be discussed later in this chapter.

5.2 Montgomery Multiplication (MM) Algorithm

The application of the Montgomery Multiplication (MM) algorithm on two integers, *X* and *Y*, with required parameters for *m*-bits of precision (where $0 \leq X, Y < M$), will result in the number $Z' = MM(X, Y) = X'Y'r^{-1} \bmod M$, where $r = 2^m$ and *M* is a modulus and an integer in the range $2^{m-1} < M < 2^m$ such that $gcd(r, M) = 1$. Since $r = 2^m$, it is sufficient that the modulus *M* be an odd integer. For cryptographic applications, *M* is usually a prime number or a product of primes, thus this condition is easily satisfied. The *image* or the *M-residue* of an integer *X*, is defined as $X' = Xr \bmod M$. It is easy to show that the Montgomery multiplication over the images X' and Y' computes the image $Z' = MM(X. Y)$, which corresponds to the integer $Z = XY \bmod M$. Figure 5.1 (a) shows radix-2 Montgomery multiplication algorithm. The transformation between the image and the integer set is accomplished using the Montgomery algorithm.

From the integer value to the *M*-residue: $X' = MM(X r^2) = Xr \bmod M$ and from the M-residue to the integer value: $X = MM(X', 1) = Xrr^{-1} \bmod M = X \bmod M$. Usually $r^2 \pmod M$ is pre-computed and saved. Thus, only a single Montgomery multiplication is needed to perform either one of these transformations. It should be noted that, for the Montgomery multiplier architectures presented in this chapter, the overhead of the conversion is not considered. The integers are converted to the M-residues manually (as the number is 8-bit) and are supplied to the architecture. Montgomery multiplication starts with the loop condition stated in step 2. The loop will be executed 0 to m-1 times. Initially, partial residue, S is 0. The algorithm computes a new partial residue for each bit of X, scanning all the bits of Y. Once Y, is completely read, another bit of X, is taken and the scan is repeated. The product of the X and Y will be added to the previous partial residue. If the sum of the partial product and the partial residue is even, the partial residue is shifted as depicted by step 4 of the algorithm of Figure 5.1 (a). However, if the sum of the partial product and the partial residue is odd, modulus, M will be added to make the sum even and the partial residue is shifted as depicted by step 5 of the algorithm. This process is repeated for 0 to m-1.

Algorithm MM:

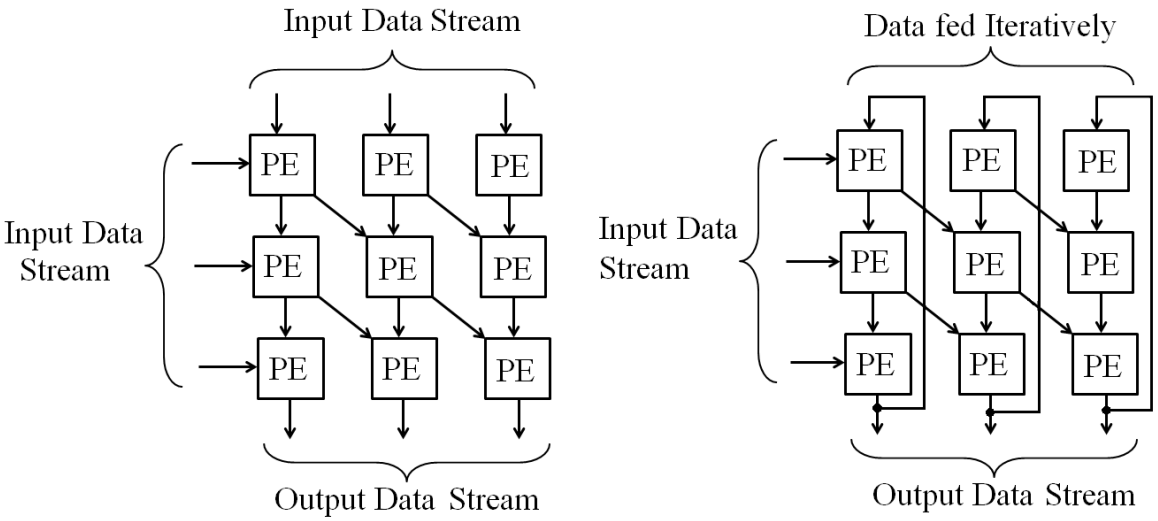
Radix-2 Montgomery modular multiplication [61]

Inputs **:** *(X, Y, M)*

Output **:** *S*

1. *S= 0*
2. *for i = 0 to m-1* (the effect of division by r (2^m)
3. *if ($S + x_i Y$) is even*
4. *then S: = ($S + x_i Y$)/2* (If even) (divide by 2, Shift operation)
5. *else S: = ($S + x_i Y + M$)/2* (If odd) (divide by 2, Shift operation)
6. *end if*
7. *end for*

(a)



(b) (c)

Figure 5.1: (a) Radix-2 Montgomery multiplication algorithm [61] (b) Simple block diagram of Systolic array approach and (c) Iterative approach

A simple block diagram of systolic and iterative architecture approach is shown in Figure 5.1 (b) and (c). All the adiabatic logic designs have differential input and output signals, see Chapter 2 of this thesis, but for the simplicity and better understanding, complementary signals are not shown in the Figures. Moreover, any signals followed by a letter 'b', is its complementary signal.

5.3 Montgomery Multiplier (MM) Implementation using Systolic Array Architecture

Two systolic array architectures were implemented. Architecture 1 (MM_SAA1) suffered from overhead due to synchronization buffers. Therefore, architecture 2 (MM_SAA2) is proposed with a strategy to reduce the overhead and improve the energy efficiency and throughput.

5.3.1 Montgomery Multiplier using Systolic Array Architecture 1 (MM_SAA1)

An 8-bit Montgomery multiplier using systolic array architecture was implemented. The architecture is shown in Figure 5.2. Partial residues are generated by Partial Residue Units (PRUs). Each PRU has its associated Modulus Addition Decision Unit (MADU). For an 8-bit architecture, a total of eight PRUs and eight MADUs are required. Each PRU computes the partial residue for the next PRU. $X_{0\text{-}7,}$ $Y_{0\text{-}7}$, represent the two 8-bit inputs (one byte each) and $M_{0\text{-}7}$ represents the 8-bit (one byte) Modulus respectively. The bits of X and Y are represented as X0-X7, and Y0-Y7 respectively. The LSBs of the partial products X0Y0, X1Y0…..X7Y0 are sent to the MADUs. The output of the MADUs $M1_{0\text{-}7,}$ $M2_{0\text{-}7}$,…, $M8_{0\text{-}7}$ are supplied to the respective PRUs. A1-A9, B1-B9,……., G1-G9 are the partial residues calculated by the first seven PRUs. The final residue (S1-S9) is generated by the 8^{th} PRU.

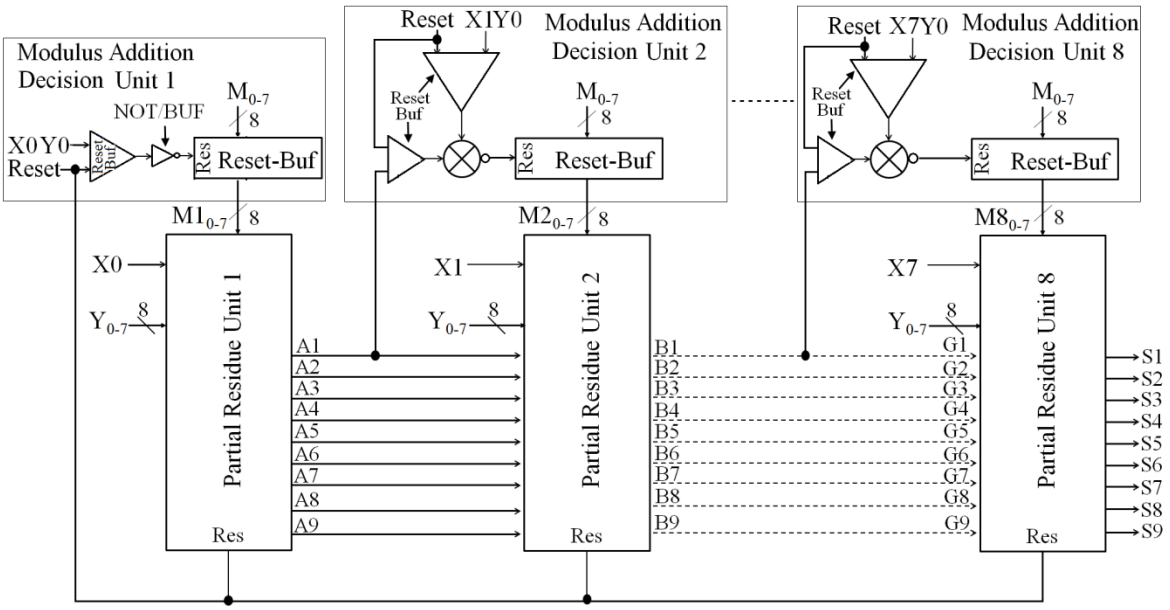
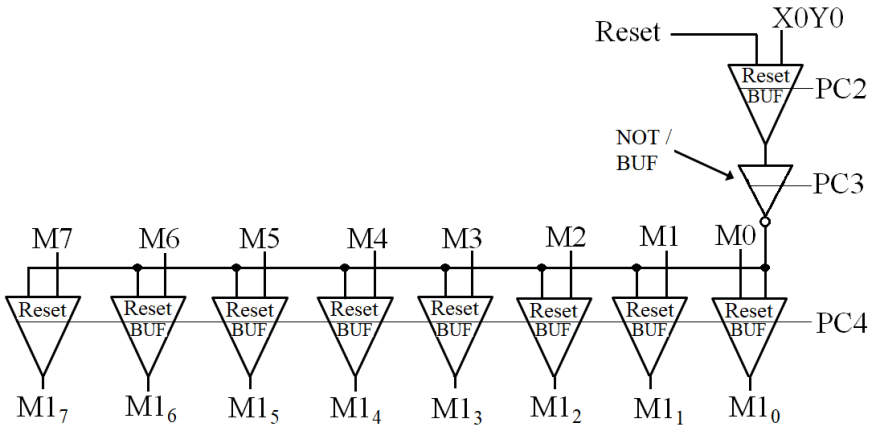


Figure 5.2: An 8-bit Montgomery multiplier using systolic array architecture.

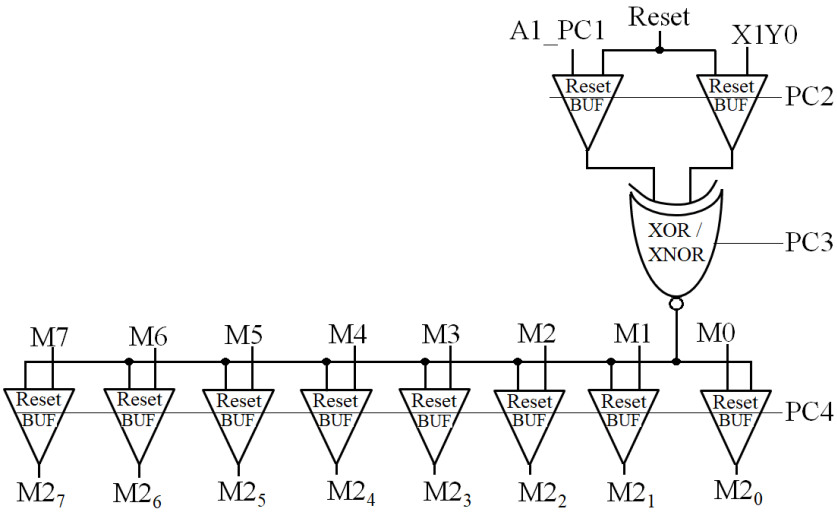
Except for the first and the last PRU, rest of the six PRUs are identical having same latency and number of gates. Also, apart from the first MADU, the structures of the rest of the seven MADUs are identical.

Each PRU is incorporated with the resettable buffer. When Reset is logic ‘1’, all the PRUs and MADUs are cleared to zero. The architecture starts the computation when Reset signal is logic ‘0’. The first set of inputs X0 and $Y_{0\text{-}7}$ are provided to the first PRU. The partial residue to the first PRU is always zero. The value of the modulus, M is passed to the first PRU based on the LSB value of the partial product (X0Y0). For the subsequent PRUs, the next set of the inputs are provided along with the partial residues calculated by the previous PRUs. The Modulus from MADUs is passed if the sum of the partial residue and partial product is odd. However, if the sum of the partial residue and partial product is even, M will not be added and the calculated partial residue will be shifted.

The detailed diagrams of the first and second MADU along with the power-clock phasing are shown in Figure 5.3 (a) and (b) respectively. For MADU1, the LSB of the partial product (X0Y0) is fed to the resettable buffer [104] and its output is passed through the synchronization NOT/BUF gate. The resettable buffers [104] are active when asserted high, hence the inverted output of the synchronization NOT/BUF is fed to the reset input of the resettable buffer stage. Modulus (M0-M7) is passed to PRU1 if X0Y0 is logic ‘1’, else zeros are passed.



(a)



(b)

Figure 5.3: (a) MADU1 (b) MADU2

MADU2 is implemented using XOR/XNOR and resettable buffers as shown in Figure 5.3 (b). The LSBs of the partial residue (A1_PC1) from PRU1 and the partial product (X1Y0) are fed to the two resettable buffers [104] and their output is fed as input to the XOR/XNOR gate. XNOR output is fed to the reset input of the resettable buffer stage. If either of the two XOR/XNOR gate input is logic '1', then M is passed to the PRU2 else zeros are passed. The structures of the rest of the six MADUs are similar to MADU2.

Figure 5.4 shows the block diagram of PRU1. It is made of a number of Processing Elements (PE), AND/NAND, XOR/XNOR, synchronization and resettable NOT/BUF gates which work in the cascade manner to calculate the residue. All PEs are made of Half Adders (HA). Half adders are implemented using AND/NAND gate and XOR/XNOR gate working in the same power-clock phase.

Due to the limitation of the 4-phase power-clocking scheme, each horizontal gate in the architecture uses the same power-clock phase. Consequently, the implementation is not possible using ripple carry addition, where the carry is propagated to the next stage on the left; instead the carry bits from each stage are passed diagonally downwards. The resulting addition is called as a carry-save addition.

The partial products are generated through an array of AND/NAND gates. Reset (active high signal), is used to clear the PEs of all PRUs to '0' before the computation starts. The synchronization NOT/BUF gates between the resettable buffers stage and HA stage are used to synchronize the arrival of the modulus, $M1_{0\text{-}7}$ from the first MADU. The last stage

of synchronization buffers is used to synchronize the arrival of the partial products and the partial residues in the next PRU. As one of the inputs is always zero, XOR gates are used in the MSB position of PRU instead of the HAs, saving the area of carry circuitry. The output ($M1_{0\text{-}7}$) from the MADU1 is added based on the LSB (even, if 0 or odd, if 1) of the partial product (X0Y0). The addition is performed until all the carry outputs are resolved to zero. To resolve the carry output to zero in the addition of two 8-bit numbers (i.e. the partial product and the modulus, M), a maximum of nine stages of HAs in cascade are required.

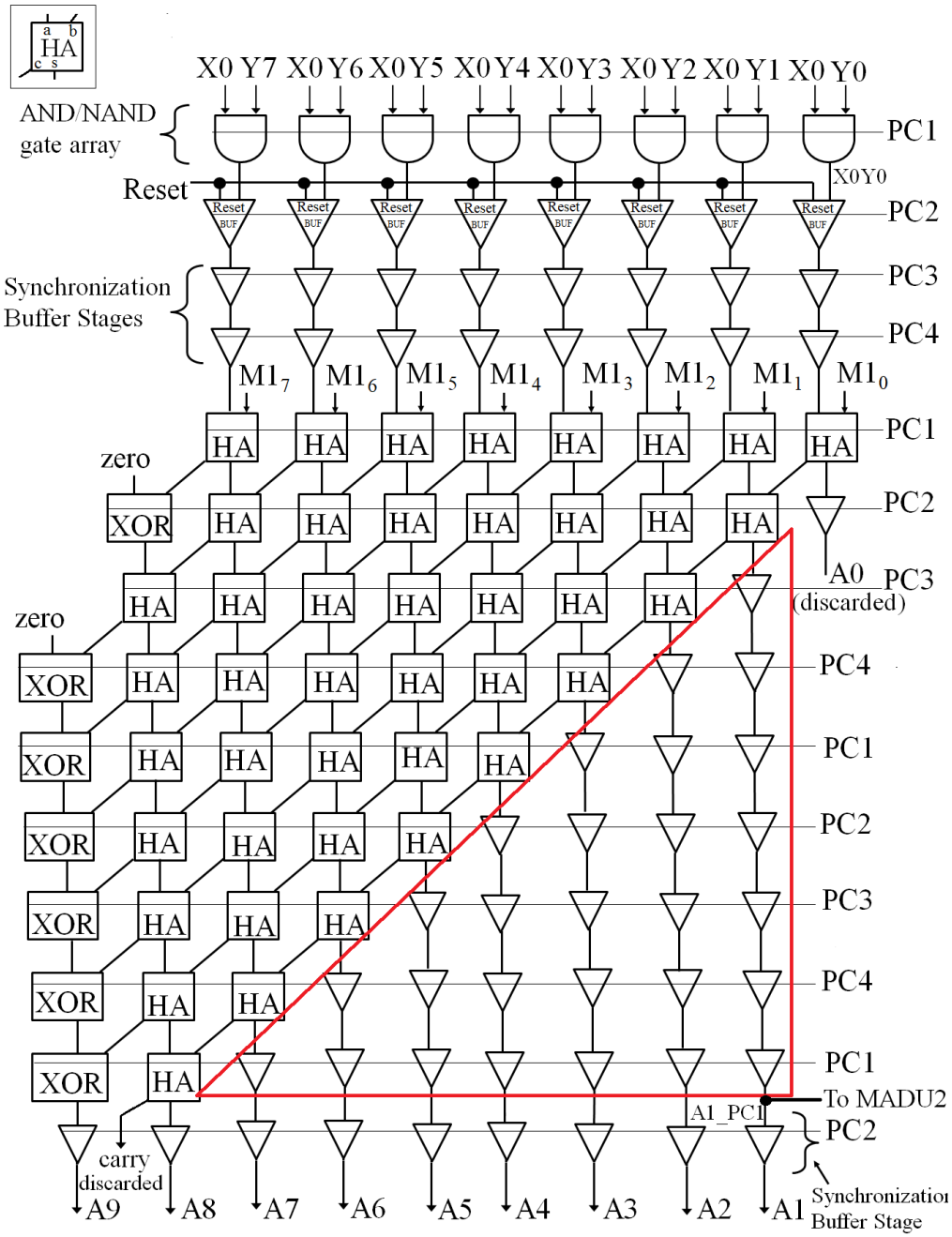


Figure 5.4: Partial Residue Unit 1 (PRU1) for MM_SAA1

The LSB of the generated partial residue, A0 is shifted and the next significant bit, A1 will become the new LSB of the partial residue of PRU1 (A1-A9). PRU1 has a latency of 3.5 power-clock cycles or 14 power-clock phases. Figure 5.5 shows PRU2 for MM_SAA1. To save the power-clock phases, the first two phases, PC1 and PC2 in the PRU2 are generated in parallel to the last two phases of PRU1.

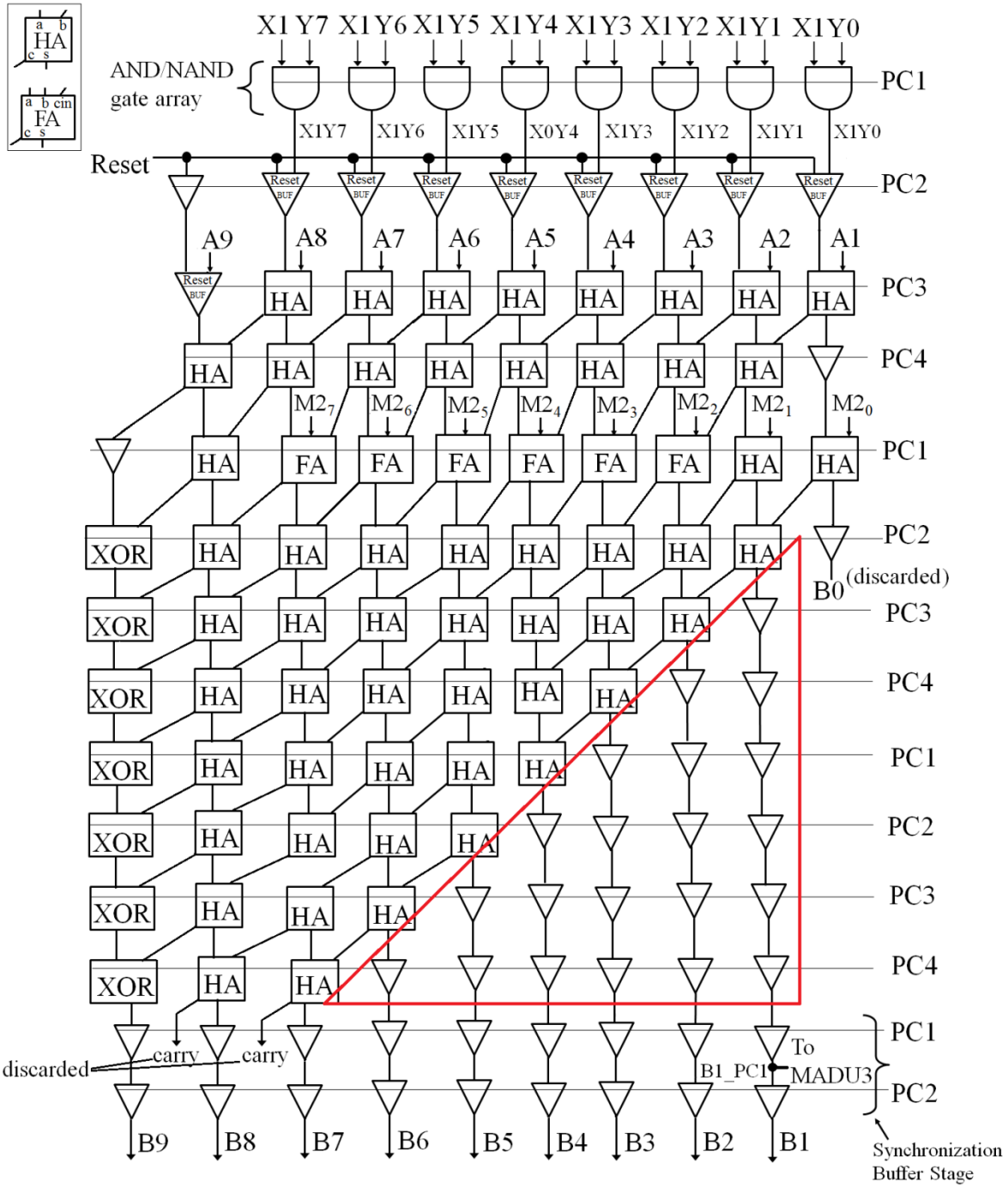


Figure 5.5: Partial Residue Unit 2 (PRU2) for MM_SAA1

The residue from PRU1 (A1-A9) will be added to the next set of partial products in PRU2. The output ($M2_{0-7}$) from the MADU2 is added in PRU2, based on the LSBs (even, if 0 or odd, if 1) of the partial product (X1Y0) and the partial residue (A1_PC1).

Since the partial residue from PRU1 is of 9-bits (A1-A9) and is added to the partial product which is 8-bit long, the maximum number of adder stages required for PRU2 is ten for the carry outputs to resolve to '0'. The LSB, B0 is discarded and the next significant bit, B1 becomes the new LSB of the calculated partial residue (B1-B9). As mentioned before, the two stages of synchronization buffers in the last are added to synchronize the arrival of the partial residue and the partial product. PRU2 has the latency of 3 power-clock cycles or 12 power-clock phases. The partial residue generated by PRU2 will then be sent to PRU3 and added to the third set of partial products. The final partial residue is calculated by PRU8, therefore, the requirement of the synchronization buffer stages in PRU8 is obviated.

For an 8-bit Montgomery multiplier, the final residue is available at the end of PRU8. The throughput of the design is 24 power-clock cycles and the total computation time is 1.7664us at 13.56MHz. In this implementation, fifteen stages of the synchronization buffers are used.

5.3.2 Montgomery Multiplier Systolic Array Architecture 2 (MM_SAA2): Synchronization Overhead Reduction

In adiabatic logic design, the primary focus is on achieving the lowest energy dissipation possible. Therefore, it is crucial to find an architecture that does not call for many buffer stages due to synchronization reasons. Synchronization buffer stages in MM_SAA1 are required because all the partial products were generated in the first phase of the power-clock, PC1 (see Figure 5.4 and 5.5). Therefore, the arrival of the partial residues must be synchronized. If the condition of the generation of the partial products in the same phase for each of the PRUs is relaxed, synchronization buffer stages can be avoided. Based on this, MM_SAA1 is modified.

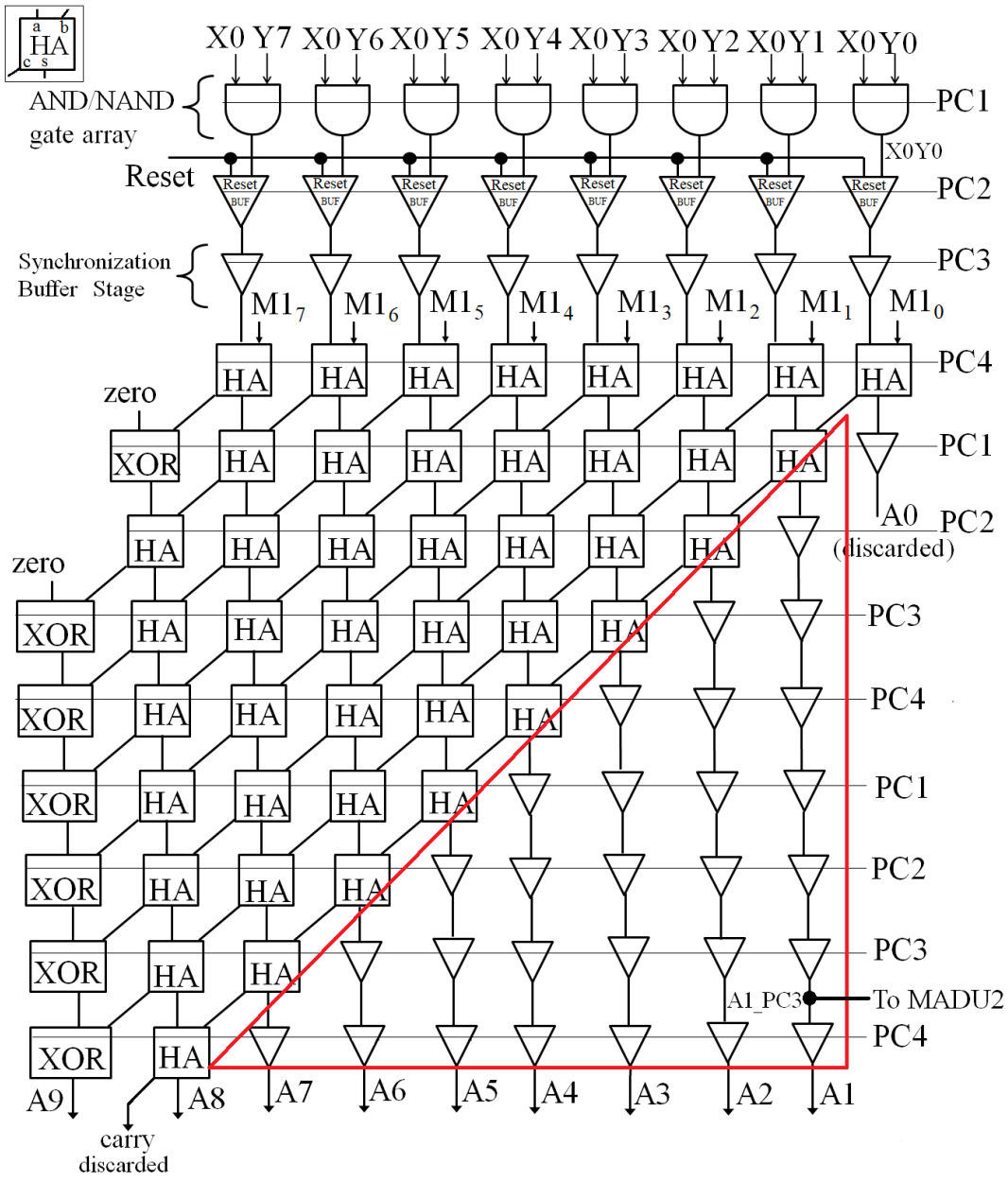


Figure 5.6: Partial Residue Unit 1 (PRU1) for MM_SAA2

For PRU1 of MM_SAA2, partial products and the partial residue are generated in phase, PC1 and PC4 respectively (Figure 5.6). Therefore, in PRU2, to synchronize the arrival of partial products with respect to partial residues, the partial product needs to be generated in power-clock phase, PC3 (Figure 5.7). Consequently, the inputs X and Y are passed through two synchronization buffer stages working in phases PC1 and PC2. The inputs, X and Y generated in PC2 can be fed as input to the AND gate array working in PC3 (in PRU2) which generates partial products in power-clock phase, PC3. The partial residues in PRU2 are generated in phase PC2. Similarly, for PRU3, the partial product need to be generated in phase, PC1. For the rest of PRUs, the partial products need to be generated either in phase, PC1 or PC3.

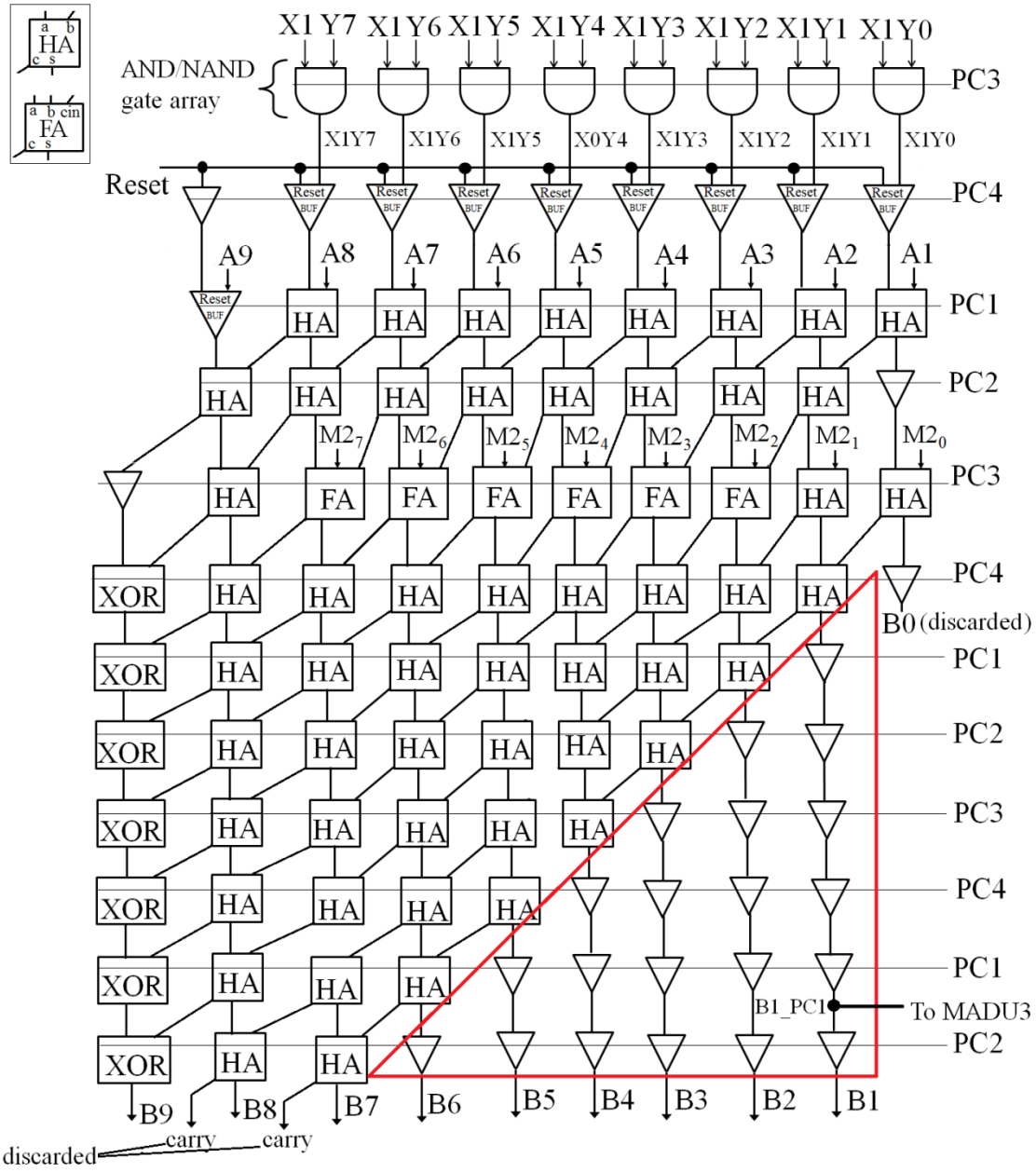


Figure 5.7: Partial Residue Unit 2 (PRU2) for MM_SAA2

Figure 5.6 shows PRU1 of MM_SAA2. The synchronization buffer stage between the resettable buffer stage and the HA stage is used to synchronize the modulus ($M1_{0\text{-}7}$) from the MADU1. Like MM_SAA1, the LSB bit, A0 is shifted and the next significant bit, A1 will now become the LSB of the partial residue (A1-A9). The calculated partial residues will then be added to the next partial products in PRU2. Figure 5.7 shows PRU2 of MM_SAA2. It can be seen that no synchronization buffer stages are used in PRU2. The calculated partial residue (B1-B9) will be sent to PRU3. Figure 5.8 shows PRU3 of MM_SAA2. PRUs from 3 to 8 of MM_SAA2 have identical structures as PRU2, except the power-clock phase requirements.

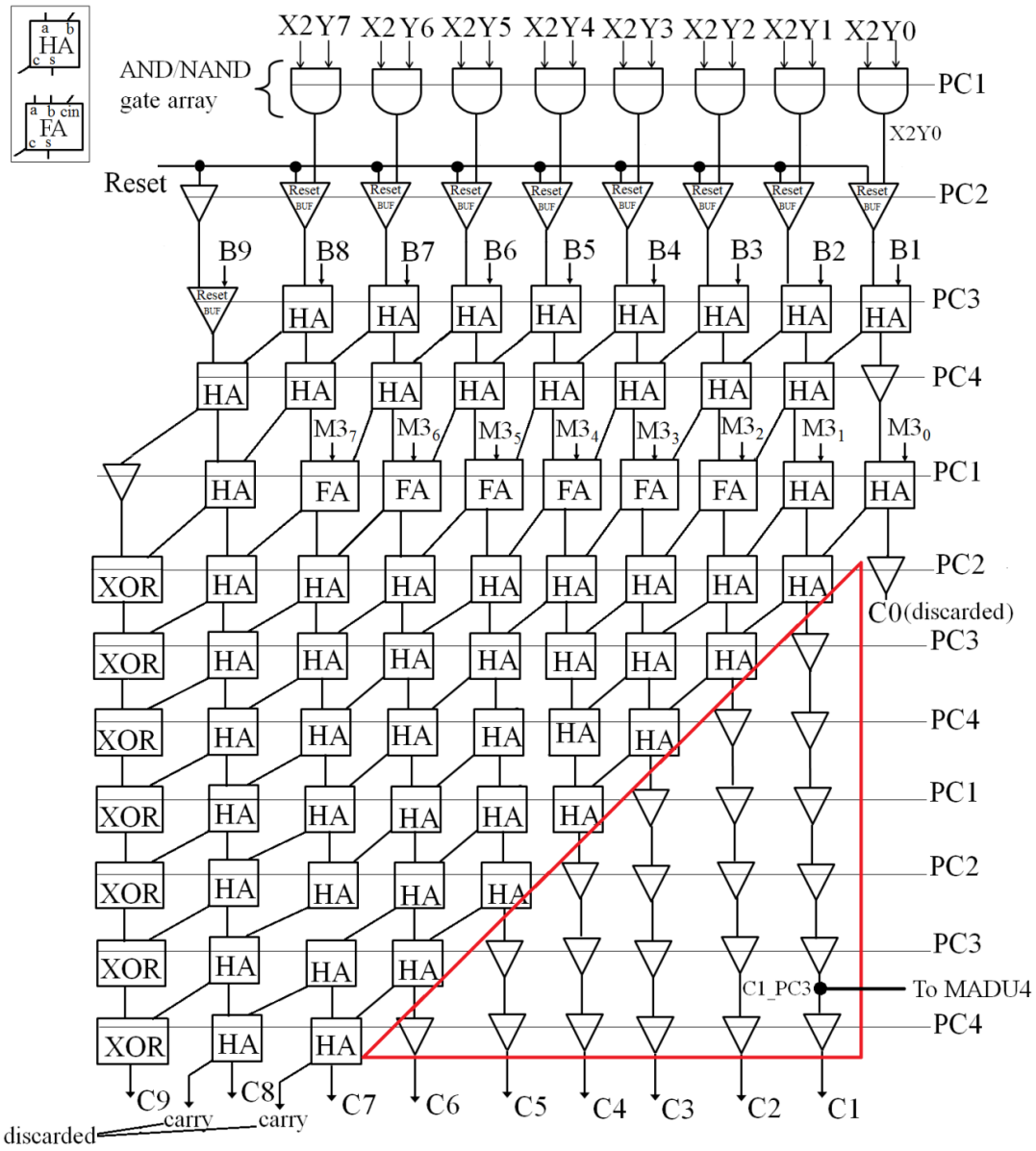
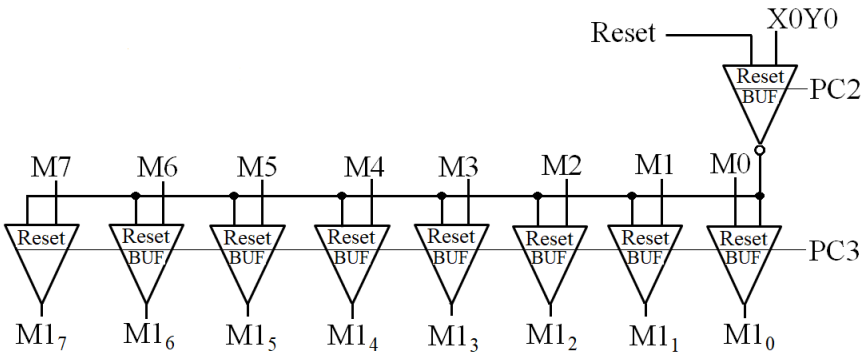


Figure 5.8: Partial Residue Unit 3 (PRU3) for MM_SAA2

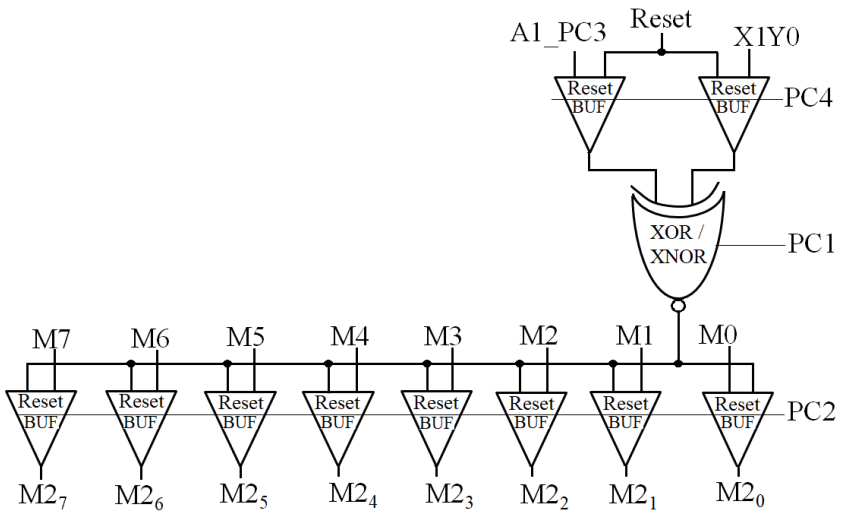
Figure 5.9 (a), (b) and (c) shows the detailed diagram of MADU1, MADU2, and MADU3 respectively for MM_SAA2. For MADU1, the condition of the addition of M depends on the LSB of the partial product (generated in phase PC1). Therefore, M inputs are fed to the resettable buffer stage working in phase, PC3. It should be noted that in MADU1 of MM_SAA2 no synchronization NOT/BUF gate is used unlike MADU1 of MM_SAA1 of Figure 5.3 (a).

In MADU2, the condition of the addition of M depends on the LSB of the partial product (generated in PC3) and the partial residue (A1_PC3, generated in PC3). M inputs are fed to the resettable buffer stage working in power-clock phase, PC2.

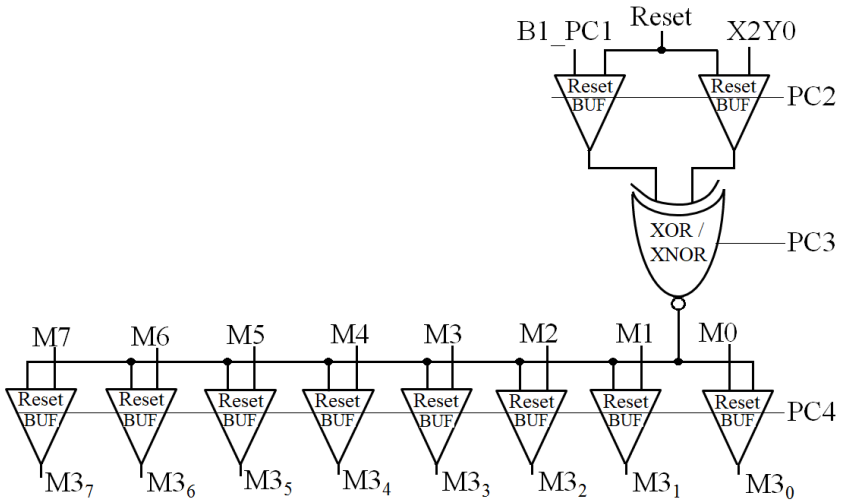
For MADU3, the condition of the addition of M depends on the LSB of the partial product (generated in PC1) and the partial residue (C1_PC1, generated in PC1). M inputs are fed to the resettable buffer stage working in phase, PC4. Therefore, M input was generated in phase PC1, PC2, and PC3 of the power-clock. The phases of the 8-bit, M were generated by passing it through three buffer stages. The structure of the rest five MADUs used in MM_SAA2 is identical to MADU2/MADU3, except the power-clock phase requirements.



(a)



(b)



(c)

Figure 5.9: (a) MADU1 (b) MADU2(c) MADU3

In total, eight buffer stages having eight buffers each are used in MM_SAA2. Out of that only one buffer stage used in PRU1 add to the latency/throughput of MM_SAA2. An area equivalent of approximately 414 transistors is saved using MM_SAA2. The savings due to the synchronization overhead reduction technique are expected to increase as the architecture is scaled to the higher number of bits. This is due to the excessive overhead of synchronization buffer stages that comes with the increased number of bits. More importantly, the latency of each PRU is reduced by 2 power-clock phases which will also reduce the overall energy dissipation.

It should be noted that in worst case, nine buffer stages having 8-bits each will be required to generate all the phases of X, Y and M. Therefore, the benefit of this strategy is that even if the number of inputs in the systolic array architecture for Montgomery multiplier are scaled up to m-bits, the overhead of this technique will still be nine buffer stages of m-bit each.

To give a more realistic comparison in terms of area and throughput the complexity analysis is performed on both the architectures. Furthermore, the CMOS architecture is also implemented and compared with the adiabatic implementation in the next subsection.

5.3.3 Area and Throughput Complexity Analysis of Systolic Array Architectures

In this section, the area and throughput complexity analysis of MM_SAA1, MM_SAA2 and systolic array architecture using conventional CMOS is performed.

In MM_SAA1 the numbers of components used are:

AND	= 64
Reset Buffer (ResBuf)	= 150
Buffer (synchronisation stage)	= 133
XOR	= 63
Full Adder (FA)	= 42
Half Adder (HA)	= 421
Buffers (adder stage)	= 191

In general, for an m-bit MM_SAA1, the number of components required can be approximated as; for generating the partial products, m stages are required each having m 2-input AND gates. Thus, the total number of AND gates required will be m^2. Since the first and the rest *(m-1)* PRUs require m and *(m+1)* resettable buffers respectively, the total number of reset buffers in PRUs are $m + (m-1)(m+1) = m^2+m-1$. Also, the first and rest *(m-1)* MADUs use *(m+1)* and *(m+2)* reset buffers respectively, the total number of reset buffers in MADUs will be $(m+1)+(m-1)(m+2) = m^2 +2m-1$. In total, $2m^2 + 3m - 2$ resettable buffers are required for m-bit MM_SAA1.

Considering the number of synchronization buffer stages, PRU1 requires two stages of m buffers and one stage of *(m+1)* buffers. In PRU8, no buffer stages are required. The remaining (m-2) PRUs require two stages of *(m+1)* buffers each. In total, $2m+ (m+1) + 2(m-2)(m+1) = 2m^2+m-3$ synchronization buffer stages are required. In the adder stages of PRU1, *(m-1)* stages of buffers are used in the triangular pattern (red) and one buffer for the discarded bit, A0 (see Figure 5.4). Also, one NOT/BUF gate is used in MADU1. The total number of buffers required in PRU1 (excluding the synchronization buffer stages) will be $2+ \frac{m(m-1)}{2}$. For the rest *(m-1)* PRUs, the number of buffers required is $4-(m-1) + \frac{m(m-1)}{2}$ in each PRU. Therefore, $6m - m^2 - 3 + m \left(\frac{m(m-1)}{2}\right)$ buffers, excluding the

buffers in the synchronization buffer stages are required. Overall, MM_SAA1 architecture requires $m^2 + 7m - 6 + m\left(\frac{m(m-1)}{2}\right)$ buffers.

It should be noted that all adiabatic logic gates have differential output. Therefore, NOT and XNOR gates are considered BUF and XOR gates respectively. In case of XOR gates, each PRU requires (*m-1*) gates and *m* MADUs also require (*m-1*) gates. Therefore, in total **m**(*m-1*) + (*m-1*)= (m^2*-1*) XOR gates are required.

FAs are not required in PRU1 as the initial value of the partial residue is always zero. The rest (*m-1*) PRUs require (*m-2*) FAs. In total, (*m-1*)(*m-2*) = m^2*-3m+2* FAs are required.

To add two m-bit numbers in the PRU1, *(m+1)* stages of HAs are required to resolve the carry to zero. With each addition, the number of HAs in the successive stages is reduced by one as can be seen from Figure 5.4. Thus, *(m-1)* + $\frac{m(m+1)}{2}$ HAs are required in PRU1. As the output of PRU1 is *(m+1)* bit long and is added to m-bit partial products and M, the rest of the (*m-1*) PRUs require *(m+2)* stages of HAs to resolve the carry. Therefore, (*m-1*) PRUs require *2(m+1)* + $\frac{m(m+1)}{2}$ HAs in each. Therefore, in total, $2m^2 + m - 3 + m\left(\frac{m(m+1)}{2}\right)$ HAs are required.

As stated before for MM_SAA2, eight stages of synchronization buffers having eight buffers each are used. Therefore, in total 64 synchronization buffers are used in MM_SAA2 in comparison to 133 buffers in MM_SAA1. This saves an area equivalent of approximately 51.88%. The rest of the components in MM_SAA2 are same as in MM_SAA1. Taking the worst condition, m-bit MM_SAA2 requires 10m synchronization buffer stages.

All the implementations require 4-phase power-clocking scheme. Therefore, one power-clock cycle is equivalent to four power-clock phases. For m-bit MM_SAA1, PRU1 and PRU8 have the latency (in terms of power-clock phases) of *(m+6)* and *(m+2)* respectively. The rest *(m-2)* PRUs have the latency of *(m+4)* each. Therefore, the throughput for m-bits MM_SAA1 can be summed up to be m^2*+4m*. Similarly, for m-bit MM_SAA2, PRU1 has the latency of *(m+4)* and the remaining *(m-1)* PRUs' have the latency of *(m+2)* each. Therefore, the throughput of MM_SAA2 is m^2*+2m+2* power-clock phases.

Regarding the number of synchronization buffer stages used, it is worth mentioning that in comparison to MM_SAA1, two stages of buffers were saved in each *(m-1)* PRUs of MM_SAA2. Thus, increasing the throughput by *0.5(m-1)* power-clock cycles. Table 5.1 shows the complete analysis of the area and throughput complexities in terms of logic gate count and throughput for MM_SAA1, MM_SAA2, and conventional CMOS.

Table 5.1: Comparison of area-throughput complexity

Designs	**Area Requirement (no. of logic gates)**	**Computation time (power-clock cycles)**
MM_SAA1	$m_{AND}^2 + (2m^2 + 3m - 2)_{ResBuf} + (m^2 - 3m + 2)_{FA} + (m^2 - 1)_{XOR} + \left(m^2 + 7m - 6 + m\left(\frac{m(m-1)}{2}\right)\right)_{TotBuf} + \left(2m^2 + m - 3 + m\left(\frac{m(m+1)}{2}\right)\right)_{HA}$	$0.25m^2 + m$
MM_SAA2	$m_{AND}^2 + (2m^2 + 3m - 2)_{ResBuf} + (m^2 - 3m + 2)_{FA} + (m^2 - 1)_{XOR} + \left(16m - m^2 - 3 + m\left(\frac{m(m-1)}{2}\right)\right)_{TotBuf} + \left(2m^2 + m - 3 + m\left(\frac{m(m+1)}{2}\right)\right)_{HA}$	$0.25m^2 + 0.5m + 0.5$
Conventional CMOS	$m_{AND}^2 + (m^2 + m - 1)_{XOR} + (m^2 - m)_{FA} + m_{NOT} + (2m^2 + 1)_{MUX2} + \left(3m^2 + m + \frac{m(m+1)}{2}\right)_{FF} + \left(2m^2 + m - 3 + m\left(\frac{m(m+1)}{2}\right)\right)_{HA}$	$m + 1$

5.4 Montgomery Multiplier Implementation using Iterative Approach

Since pipelining is inherent in adiabatic logic, careful selection of architecture is essential, as otherwise overhead in terms of area and energy due to synchronization buffers is

induced specifically, in the case of 4-phase adiabatic designs. The area and energy of the systolic array architectures proposed in the previous section are expected to increase enormously as the number of bits is scaled up (Table 5.1). Therefore, an architecture is required which can be scaled up to a higher number of bits without increasing area and energy enormously. Iterative approach architecture can be used to reduce the area and energy dissipation at the cost of circuit complexity. The complexity increases due to the synchronization requirement of the 4-phase power-clocking scheme. Three iterative approach architectures using single, two and three adder stages in the datapath unit are proposed and the optimum number of adder stages in the datapath unit is investigated.

5.4.1 Montgomery Multiplier using Iterative Architecture 1 (MM_IA1)

Iterative approach architecture for the 8-bit Montgomery multiplier was designed based on the Montgomery algorithm of Figure 5.1. The block diagram of the iterative approach architecture 1 is shown in Figure 5.10. It comprises of Partial Products Generator Unit (PPGU), 3-bit counter, controller unit, Datapath Unit (DU) with single adder stage, Modulus Addition Decision Unit (MADU), synchronization buffer stages and Residue Register Unit (RRU).

5.4.1.1 Working of MM_IA1

The architecture is initialized using the 'Reset' input set to logic '1'. It initializes the counter to "000" state and RRU outputs (S0-S8) to zero. Signal 'Z0' is generated from the AND operation between the signal Resetb and the control signal 'CS5'. 'Resetb' is the complement of the 'Reset' signal. 'Z0' is a control signal used in MADU and initializes it to zero before the start of the computation. Signal 'Z1' is generated from signal 'Z0'. Both the signals, 'Z0' and 'Z1' initialize the datapath unit to zero.

The signal 'Rin' is generated from OR operation between 'CS1' and 'Reset' signal and is supplied to the resettable buffer stage in RRU. It resets the RRU to zero before the computation starts (when reset is logic '1') and also during the computation of the partial residues when carry outputs are not resolved.

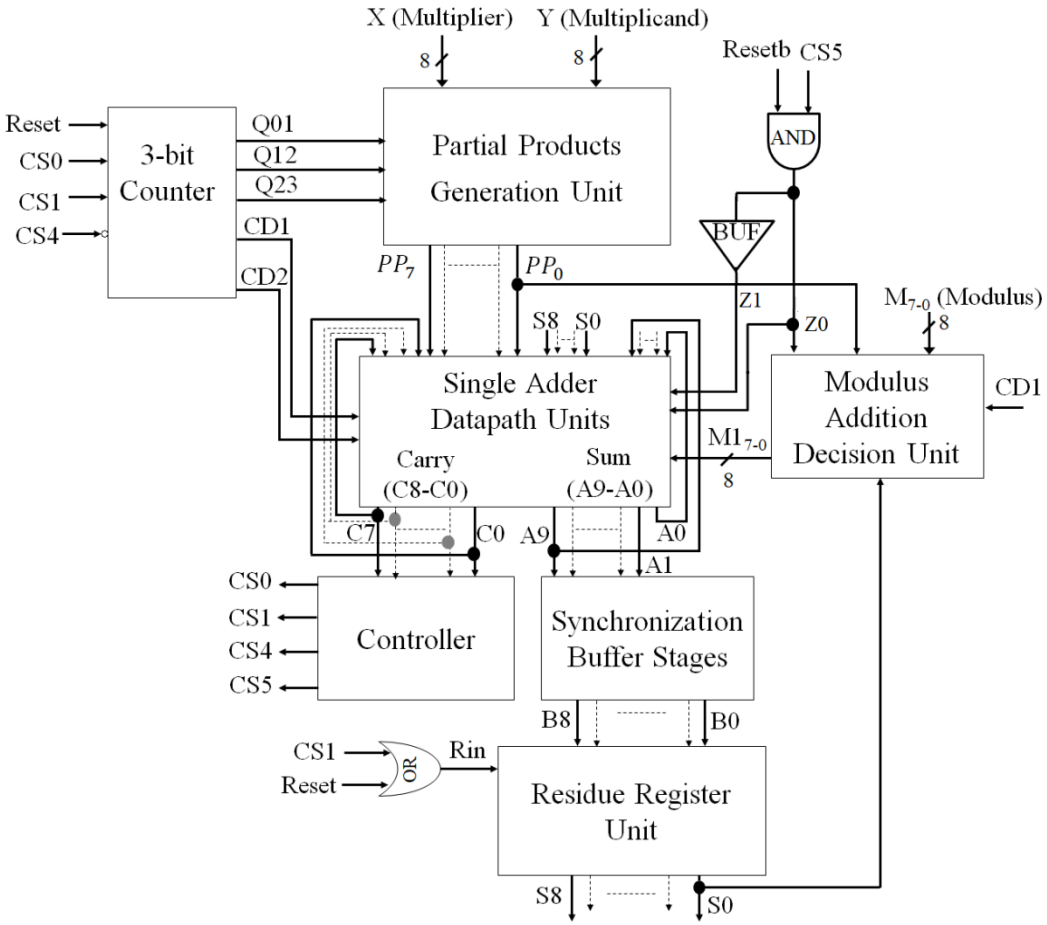


Figure 5.10: Block diagram of 8-bit MM_IA1

The computation starts when the reset signal is logic ‘0’. The “000” state of the counter triggers the first set of partial products from PPGU to be available to the datapath unit as input. The decision of adding M is taken by the Modulus Addition Decision Unit (MADU). If the sum of the partial products and the partial residue is even, the partial residue is shifted. However, if the sum of the partial residue and the partial product is odd, modulus M will be added in order to make the sum even and then the sum is shifted.

In the datapath unit, modulus operation is performed. The carry outputs from the datapath unit are fed as inputs to the controller unit. The Controller unit generates the control signals (‘CS0’, ‘CS1’, ‘CS4’, and ‘CS5’) to synchronize the counter unit, datapath unit, MADU and RRU. The Control signals ‘CS1’, ‘CS4’, and ‘CS5’ are generated by passing the control signal ‘CS0’ through one, three and four synchronization buffers respectively. If the carry outputs are not resolved to zero, the control signals will be logic ‘1’ and the carry and the sum outputs of DU are fed back as inputs to DU recursively until all the carry outputs are resolved to zero and control signals are logic ‘0’. During the time when control signals are logic ‘1’, the controller performs three major tasks for ensuring the correct

operation of the architecture; 1) It does not allow the inputs M, partial products and partial residues to be supplied to the datapath unit. 2) Retains the counter in the same state so that no new partial products are supplied. 3) Reset RRU so that the values of its output (S0-S8) are zero.

When the carry outputs are resolved to zero, the control signals are set to logic '0' allowing the sum outputs of the datapath unit (A1-A9) to be stored in RRU after passing through the synchronization buffer stages. At the same instant, the control signals trigger the counter to change state. The counter outputs (Q01, Q12, and Q23) enable PPGU to send the next set of partial products to the datapath unit. Also, the partial residue (S0-S8) along with the modulus value is provided to the datapath unit for the computation of the next partial residue. The process continues until the final residue is calculated. RRU retain the final value for one power-clock cycle. At this instant, the completion detection signals, 'CD1' and 'CD2' generated by the counter are asserted logic '1' which clears the datapath unit and MADU for the next new computation (with the same input).

The main feature of the architecture is that its datapath unit is implemented using single adder stage, where partial products, partial residues and the Modulus, M all are added in the same power-clock phase.

5.4.1.2 Implementation of MM_IA1

The complete Montgomery multiplier architecture of Figure 5.10 is implemented using PFAL. For simplicity, the complementary signals, both inputs, and outputs are not shown in the Figures. The main parts of the architecture are:

1. Counter
2. Partial Product Generator Unit (PPGU)
3. M addition decision unit (MADU)
4. Datapath Unit (DU)
5. Controller Unit (CU)
6. Synchronization buffer Stages
7. Residue Register Unit (RRU)

Counter: For an 8-bit Montgomery multiplier, a 3-bit binary counter is required. The function of the counter unit is implemented using AND/NAND, OR/NOR, XOR/XNOR gates. Using the function, NOT/BUF and resettable NOT/BUF gates [104] the counter is

designed. Figure 5.11 shows the counter implementation along with the power-clock phasing.

When the ‘Reset’ is logic ‘1’, the counter is in “000” state. When ‘Reset’ is logic ‘0’, the counter starts counting or remain in the same state depending on the value of the control signals ‘CS0’/ ‘CS1’ from the controller. It should be noted that if ‘CS0’/ ‘CS1’ is logic ‘1’, the counter remains in the same state, else the counter changes to the next state. The outputs from the counter serve as the select line inputs to the MUXs in PPGU. For each count, a new set of partial products is provided to DU. In order to save power-clock phases, signals, Q01, Q12, and Q23, were sent to PPGU. Once the count reaches “111” and the carry outputs from the DU are resolved to zero (final residue computed), all the control signals are asserted logic ‘0’ (complementary signals logic ‘1’). At this instant, the completion detection signals ‘CD0’/‘CD1’/‘CD2’ are asserted high. After the count “111” the counter restarts the count from “000”.

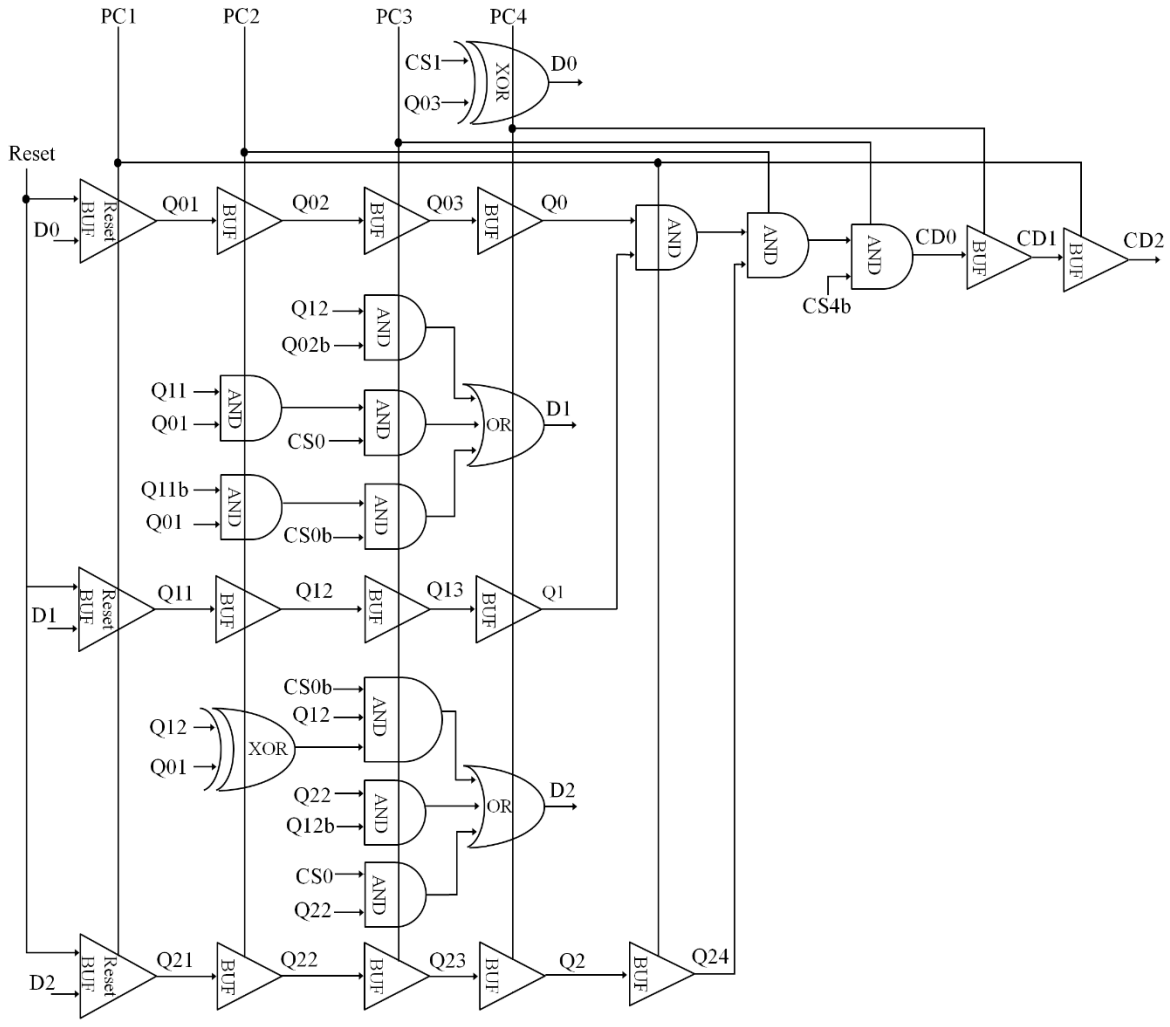
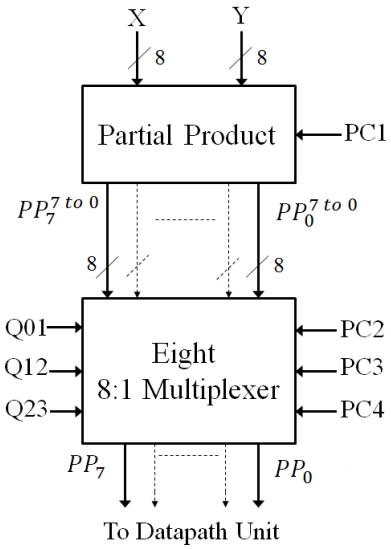
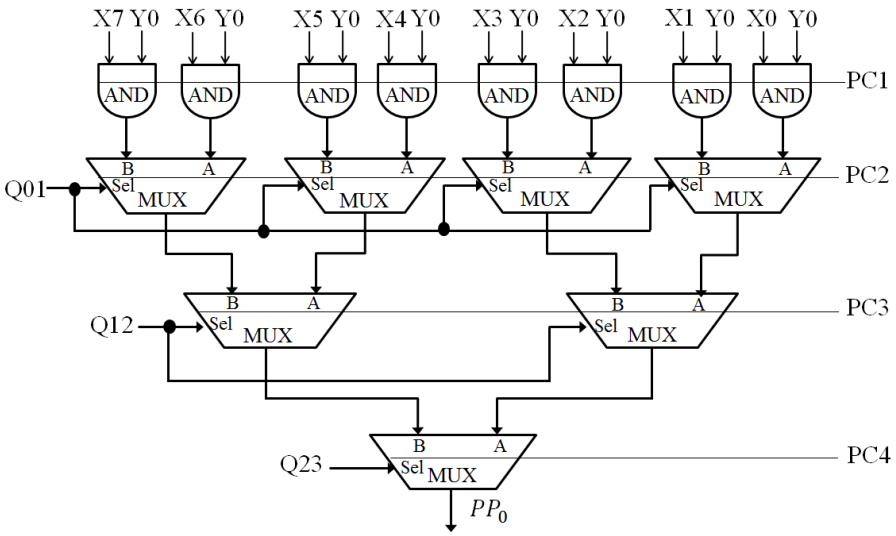


Figure 5.11: 3-bit counter generating the select lines for the MUXs in PPGU.

Partial Product Generator Unit (PPGU): Figure 5.12 (a) shows the block diagram of the 8-bit PPGU with power-clock phasing. It consists of AND gate arrays for generating the partial products and eight 8:1 MUXs for selecting the partial products. For an 8-bit MM, a total of 64 AND gates are required. The part of PPGU generating the LSB of the partial product is shown in Figure 5.12 (b). The partial products from the AND gates are sent to the 8:1 MUX implemented using 2:1 MUXs. The output from the counter unit Q01, Q12, and Q23 are the select lines to the 2:1 MUXs which selects the required partial products. PPGU uses four power-clock phases.



(a)



(b)

Figure 5.12: (a) Block diagram of PPGU (b) PPGU generating the LSB of the partial product.

M Addition Decision Unit (MADU): Figure 5.13 shows MADU with power-clock phasing for an 8-bit MM. It comprises of XOR/XNOR gate, OR gate, 2:1 MUX, and eight resettable buffers. The condition for the addition of M is calculated by performing the XOR/XNOR operation on the LSB of the partial product (PP_0) and the partial residues (S0). The XNOR output is the input to the 2:1 MUX along with logic '1' on the other input. The OR operation is performed on the signals 'Z0' and 'CD1'. The output of the gate is the select input to the 2:1 MUX. The MUX allows the addition of M to DU only once based on the select input.

The completion detection signal, 'CD1' will be logic '1' only when the final residue is computed (count "111"). At this instant, logic '1' is passed through the MUX, resetting the output of the resettable buffer stage. For the rest, 'CD1' remains logic '0' and signal 'Z0' decides whether the output of the XOR/XNOR gate or the logic '1' is passed to the reset input of the resettable buffer stage.

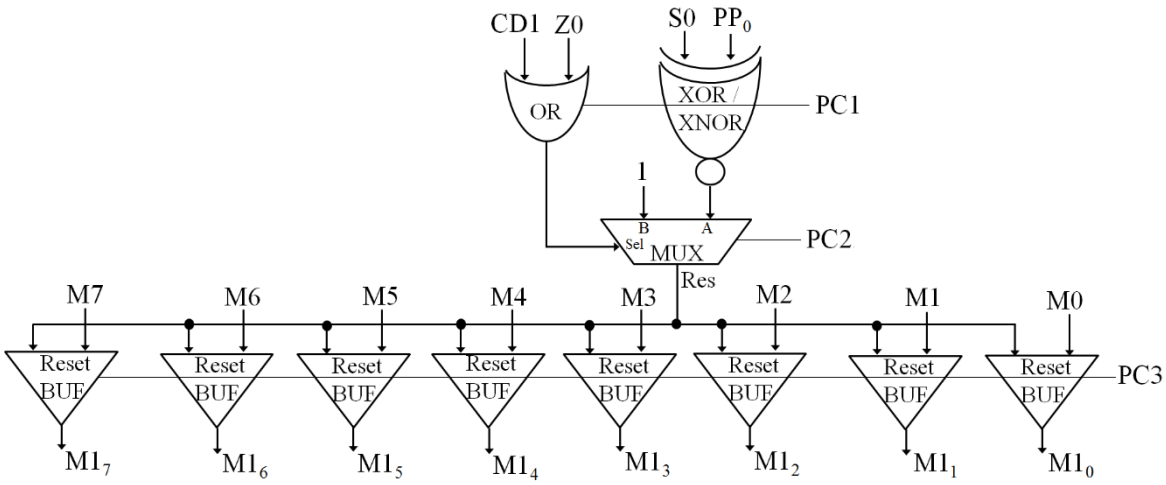


Figure 5.13: Modulus Addition Decision Unit (MADU) for MM_IA1

Datapath Unit: The datapath is the main unit where Montgomery multiplication operation is performed. It is implemented in a bit-slice manner. In this case, the bit-slice manner implies that each bit is executed in parallel. The addition of two m-bit numbers (partial products and M) and one (m+1) bit number (partial residue) results in (m+2) bit length number, therefore 10 bit-slices are required (bit-slices 0-9). Figure 5.14 shows the detailed diagram of the datapath unit. It shows the bit-slice 0, 8 and 9. The bit-slices 0-7 are identical.

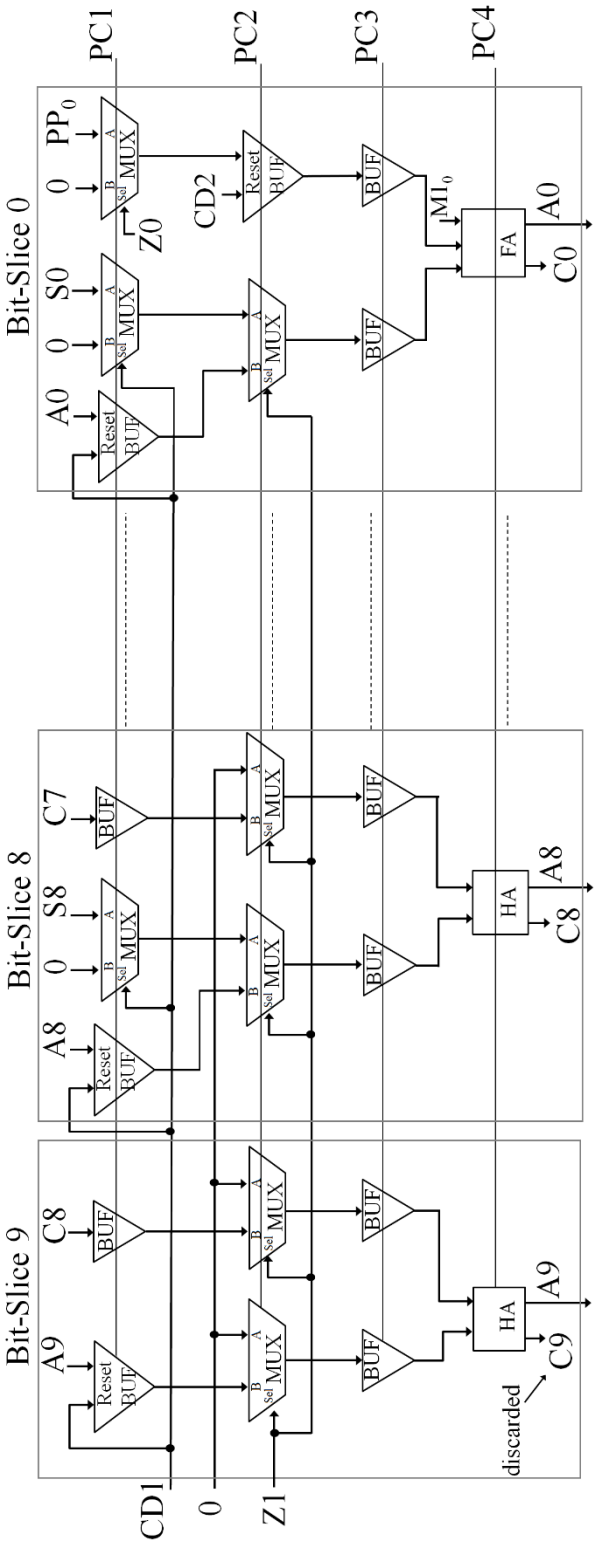


Figure 5.14: Datapath unit for MM_IA1

The inputs to the bit-slices 0-7 are partial products (PP_0-PP_7), partial residues (S0-S7), Modulus, M, intermediate sum (A0-A7) and carry outputs (C0-C6) from the datapath unit. The bit-slices 0-7 are implemented using 2:1 MUXs, NOT/BUF gates, resettable buffers and Full Adder (FA).

Inputs to the bit-slice 8 are the partial residue (S8), intermediate sum (A8) and carry output (C7) from the datapath unit. As the partial residue is of 9 bits, the bit-slice 9 has intermediate sum (A9) and carry output (C8) as the input. Both the bit-slices are implemented using 2:1 MUXs, NOT/BUF gates and resettable buffers and Half Adders.

The signals, 'Z0', 'Z1', 'CD1' and 'CD2' are the select lines to the 2:1 MUXs. Where, 'Z0', 'Z1' allows the partial products and the partial residues to be added only once when the carry outputs in the datapath unit are resolved to zero and the computation of the partial residue is completed. Whereas, 'CD1' and 'CD2' are the completion detection signals and reset the datapath unit after the calculation of the final residue is completed at the end of the count "111".

The computation starts when the select line inputs to the 2:1 MUXs are zero, allowing the partial residues, partial products, and M (from MADU) to be added. The signals 'Z0' and 'Z1' remain logic '1', till the intermediate carry outputs of DU are not resolved, ensuring partial residues, partial products, and M to be disconnected and only the intermediate sum and the carry outputs to be fed to DU recursively. DU works recursively until all the carry outputs are resolved to zero and the computation of the partial residue is completed. When the intermediate carry outputs (C0-C8) are resolved, the intermediate sum outputs (A1-A9) will be passed through the synchronization buffer stages and will be stored in RRU for the calculation of the next partial residue. Also, the signals 'Z0' and 'Z1' become '0', ensuring new partial residues (from RRU) and the new partial products (from PPGU) along with M (from MADU) to be added in DU.

The entire process is repeated eight times for calculating the final residue in 8-bit Montgomery multiplier. When the counter count reaches "111" and the final residue is computed, the signals 'CD1' and 'CD2' are asserted high. At this instant, DU is reset to zero in order to start the new computation.

If the Montgomery Multiplier is scaled up to a higher number of bits, the bit slices in the datapath unit can be replicated making it scalable. Also, the area doesn't increase enormously unlike the systolic array architecture, MM_SAA1 and MM_SAA2.

Controller Unit: The controller unit is designed using OR/NOR gates and is shown in Figure 5.15 with power-clock phasing. The carry outputs from the datapath unit (C8 to C0) are fed as inputs to the controller and OR operation is performed to generate a control

signal, ‘CS0’. For an 8-bit Montgomery multiplier two power-clock phases are required to generate ‘CS0’. For synchronizing different units (working in distinct phases) in the architecture, different phases of the signal ‘CS0’ are required. The phases of ‘CS0’ are generated by passing it through buffer gates (shown in Figure 5.15). The controller unit performs several tasks for ensuring the correct operation of the complete architecture. It repeatedly checks if all the carry outputs in DU are resolved to zero or not. If the carry outputs are resolved, the control signals reset to logic ‘0’ indicating that the partial residue is computed. Signal ‘CS1’ is used in RRU. It allows the intermediate sum outputs (A1-A9) of the DU to be stored in RRU. The signals, ‘CS0’, ‘CS1’ and ‘CS4’ are used in the counter unit for updating the state and generating the completion detection signal. Signal ‘CS5’ allows the partial residues, partial product and modulus, M to be passed to DU, ensuring they are added only once in DU. Also, to ensure that the control signals are logic ‘0’ only for one power-clock cycle, the AND operation between ‘CS0’ and ‘CS4’ is performed. The output of the gate is passed through the NOT/BUF gate used for synchronization. The inverted output of the BUF is fed as input to the controller.

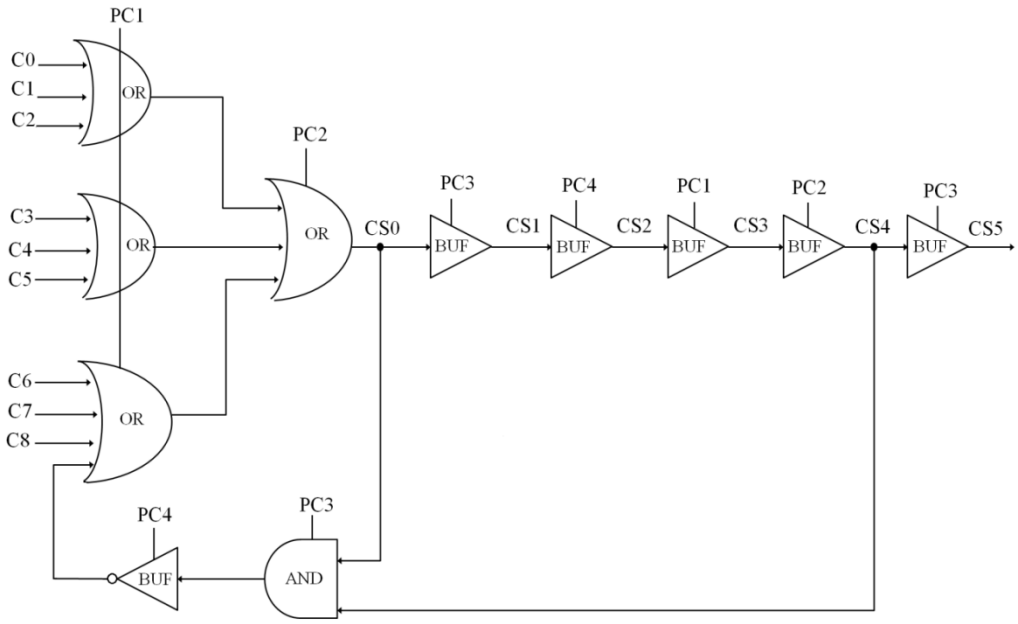


Figure 5.15: Controller unit generating the control signals

Synchronization buffer Stages: Figure 5.16 shows the synchronization buffer stages used in 8-bit Montgomery multiplier architecture. To synchronize the arrival of the partial products and the partial residues in DU, sum outputs (A1-A9) from DU are passed through the synchronization buffer stages. As mentioned before, total of eight buffer stages are required to synchronize the arrival of the new partial residues and partial products in DU.

Since, RRU requires four stages to store the partial residues for one power-clock cycle, only four stages of synchronization buffers are required.

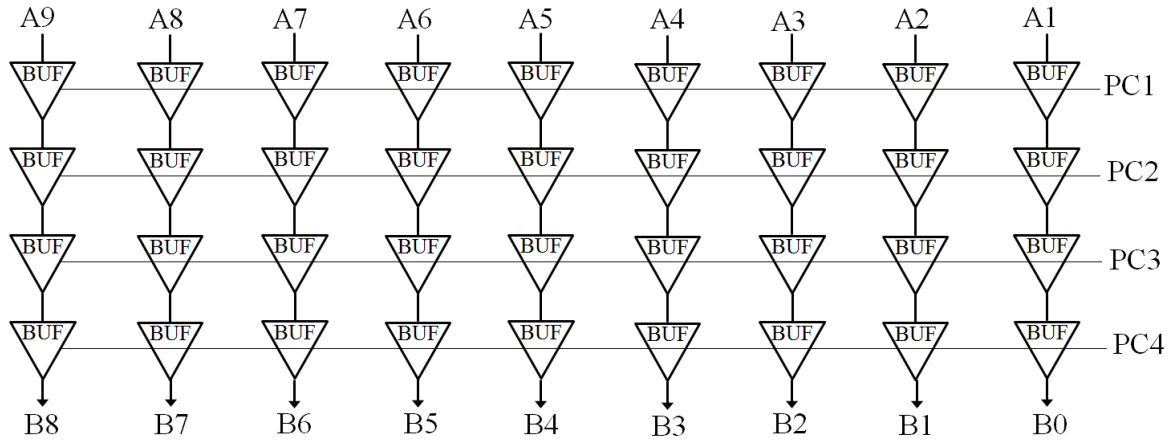


Figure 5.16: Synchronization buffer stages

Residue Register Unit (RRU): Figure 5.17 shows the detailed diagram of RRU along with power-clock phasing. It consists of one resettable buffer stage and three buffer stages. 'Rin' is the reset signal for the resettable buffer stage, generated using the OR operation between the control signal 'CS1' and 'Reset'. During the reset operation, ('Reset' is logic '1') RRU is cleared to zero. Only when both the signals ('CS1' and 'Reset') are zero the sum outputs from the datapath unit are stored in RRU for the calculation of the next partial residue. At the end of every count a newly calculated residue will be loaded to RRU. The final residue is the one which gets computed and stored at the end of the final count.

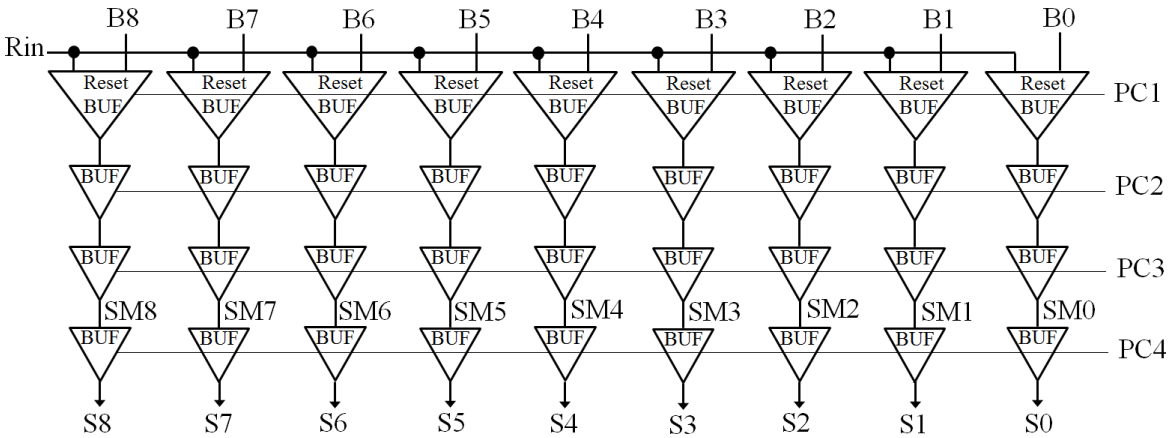


Figure 5.17: Residue Register Unit (RRU)

5.4.1.3 Methodology: Power-clock phasing in MM_IA1

Iterative approach architectures are highly complex specifically using adiabatic logic that works with 4-phase power-clocking scheme due to the synchronization of power-clock phases. The methodology presented is based on 8-bit Montgomery multiplier architecture. This methodology can be used for Montgomery multiplier architectures having higher bit operands.

Montgomery multiplier architecture consists of datapath unit, controller unit, counter unit, PPGU, and M addition decision unit. The main functional block of the Montgomery multiplier is the datapath unit which is implemented in bit-slice form. The datapath uses four levels of logic gates working in four power-clock phases. The first two power-clock phases are used by 2:1 MUXs and resettable buffers, the third phase is used by the buffer stage for synchronization and the fourth phase by the adder stage. Therefore, the sum and the carry outputs of the datapath unit are generated in phase 4 of the power-clock.

The carry outputs from DU are fed to the controller. The controller requires two power-clock phases to generate the control signal, 'CS0'. Therefore, 'CS0' is available in PC2 of the power-clock. The 'CS0' signal is sent to the counter unit which takes three phases of the power-clock to provide select line inputs to the eight 8:1 MUXs in PPGU. The 8:1 MUXs takes three power-clock phases to provide the set of partial products to the datapath unit. Therefore, the total of eight phases of the power-clock are required by the controller, the counter and the 8:1 MUXs to send the next set of partial products to the datapath unit. Moreover, the partial products and the partial residues should be added in the same phase in the datapath unit, thus the sum outputs (A1-A9) are delayed by eight power-clock phases using four synchronization buffer stages and four stages in RRU. This suggests that the number of synchronization buffer stages depends on the number of phases consumed by the controller, counter unit and MUX in PPGU.

The control signals ('CS0', 'CS1', 'CS4', 'CS5') are used in various modules; such as datapath unit, counter unit, RRU, and MADU for synchronization. Each data coming to DU travels eight power-clock phases. From Figure 5.10, it can be seen that 'CS1' is used in RRU. To generate 'CS1' the controller takes three power-clock phases. Then one power-clock phase is taken by the 'Rin' signal followed by four phases in RRU. In total, eight power-clock phases are required to save the sum outputs (partial residues to the RRU).

Similarly, control signals ‘Z0’/ ‘Z1’ are the select inputs to the MUXs in DU. The signal ‘Z0’ is generated by performing AND operation on signal ‘CS5’ and ‘Resetb’. The generation of ‘CS5’ requires seven power-clock phases and one power-clock phase is consumed by the AND operation for generating signal ‘Z0’. Again, eight power-clock phases are required to generate the control signal to synchronize the data processed in the datapath unit. This suggests that each data coming to the datapath unit has consumed eight power-clock phases.

Overall, five synchronization buffer stages (one in DU and four in synchronization buffer stages) are used in this architecture which is less in comparison to the number synchronization buffer stages used in the systolic array-based architectures.

5.4.2 Modified Montgomery Multiplier (MMM) Iterative Architecture 2 (MMM_IA2)

The Montgomery multiplier algorithm is modified. It obviates the need for the LSB bit-slice (bit-slice 0) by pre-computing the carry generated by the addition of the LSBs of the partial residues and the partial products. This architecture (MMM_IA2) requires nine bit-slices in total compared to ten bit-slices used in MM_IA1. Also, to increase the throughput of the architecture, the datapath unit of MMM_IA2 is implemented using two adder stages.

5.4.2.1 Modified Montgomery Multiplication (MMM) Algorithm

The two operands *Y* (multiplicand) and *X* (multiplier) and the modulus, *M*, all are m-bits long. These can be represented as $X = (X_{m-1}, \ldots, X_1, X_0)$, $Y = (Y_{m-1}, \ldots, Y_1, Y_0)$, $M = (M_{m-1}, \ldots, M_1, M_0)$, where the bits are marked with the subscripts, representing leftmost bit as Most Significant Bit (MSB) and rightmost bit as Least Significant Bit (LSB). The partial residue, *S*, *i*s represented as $S = (S^{m-1}_{m\ to\ 0}, \ldots, S^{1}_{m\ to\ 0}, S^{0}_{m\ to\ 0})$, where the bits are marked with the subscripts and the words (each partial residues) are marked with superscripts. The partial residue, *S*, is m+1 bits long because the total carry-out value generated in two additions corresponds to $C_a + C_b$ and it is in the range [0, 2]. This range for the carry values satisfies the condition imposed by the addition of three m-bit numbers.

The modified *radix-2* Montgomery multiplication algorithm is shown in Figure 5.18. The algorithm computes a new partial residue for each bit of *X*, scanning all the bits of *Y*. Once *Y*, is completely read, another bit of *X*, is taken and the scan is repeated.

Algorithm MMM:

Radix-2 Modified Montgomery Multiplication (MMM) algorithm

Inputs *: (X, Y, M)*

Output *: S*

1. *S= 0*
2. *for i = 0 to m-1*
3. *if* ($S_0^i + X_i Y_0$) *is one* [Condition for carry generation]
4. *then* (C_a, *S'): =* (($S^i_{m\ to\ 1}$) + $X_i Y_{(m-1\ to\ 1)}$ + 1)
5. *else* (C_a, *S'): =* (($S^i_{m\ to\ 1}$) + $X_i Y_{(m-1\ to\ 1)}$ + 0)
6. *if* (S_0^i *XOR* $X_i Y_0$) *is one* [Condition for addition of M]
7. *then* (C_b, *S'): =* (($S^i_{m\ to\ 1}$) + $X_i Y_{(m-1\ to\ 1)}$) + 1 + $M_{(m-1\ to\ 1)}$ [If odd]
8. *else* (C_b, *S'): =* ((C_a,*S'*) + 0) [If even]
9. *end if*
10. *end if*
11. *end for*

Figure 5.18: Modified radix-2 Montgomery multiplication algorithm.

In this algorithm, to avoid the addition of the LSBs of the partial products, partial residues and the modulus, M and then right shifting the calculated sum (as it is done in algorithm of Figure 5.1), an OR operation is performed between the LSBs of the partial product, and the partial residue to pre-compute the carry for the next significant bit. If either of the LSB is one, then M will be added, and this will generate carry for the next significant bits. In addition, if both, the LSBs are one, then M will not be added, but carry will still be generated. These conditions are checked in the algorithm of Figure 5.18 in steps 3 and 4. On the other hand, if both the LSBs are zero, then carry generated will be zero. This operation is described in step 5 of the modified algorithm.

Next, the condition for the addition of modulus, M, is checked. If *XOR* operation between the LSBs of partial product and the partial residue is logic '1', then M is added else zero is added. The operation is shown from steps 6 to 8.

The functionality of the modified Montgomery algorithm was validated by the VHDL code provided in the appendix section of this thesis.

5.4.2.2 Implementation of MMM_IA2

The architecture of the Montgomery multiplier based on the modified Montgomery algorithm is shown in Figure 5.19. An additional signal 'Sel' (shown in red) from MADU is used in DU. The rest of the signals and power-clock phasing is similar to MM_IA1.

The addition of LSBs of the partial product and partial residue is skipped and the condition for the generation of carry from the addition of the LSBs of the partial product and partial residue is pre-calculated. The area equivalent and hardware for one bit-slice are saved at the cost of one OR gate and one 2:1 MUX. This modification also reduces the latency by one power-clock cycle, because the carry that would have taken four phases of the power-clock in the datapath unit to get added to the next significant bits has been added in the beginning of the computation.

The working of MMM_IA2 is similar to the working of MM_IA1. Only the datapath unit and MADU are modified in MMM_IA2. Therefore, only datapath unit and MADU are discussed in detail.

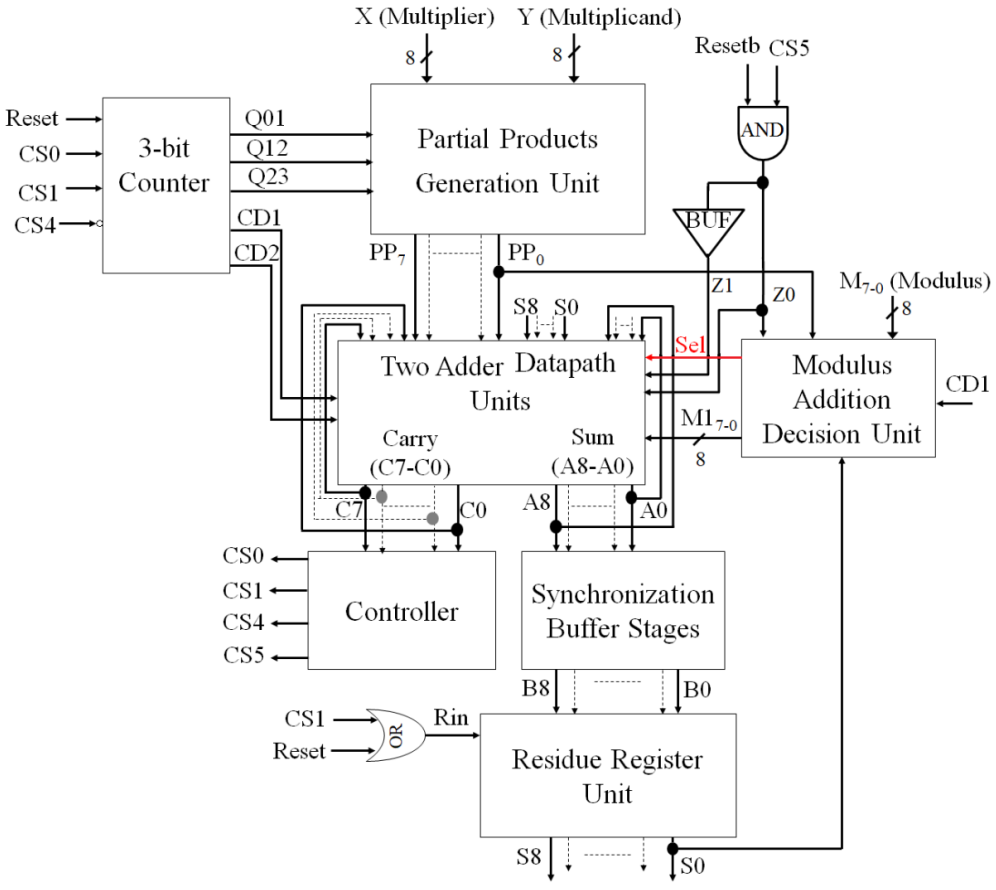


Figure 5.19: Block diagram of 8-bit MMM_IA2.

Datapath Unit: The bit-slices 1-6 are identical. Figure 5.20 shows the detailed diagram of the datapath unit for MMM_IA2 with bit-slices 0, 1, 7 and 8. The datapath unit requires four phases of the power-clock to generate the sum and the carry outputs. Each bit slice is implemented using four levels of cascade logic gates (2:1 MUXs, NOT/BUF gates, resettable NOT/BUF gate, Half Adders, and Full Adders) working in power-clock phases, PC1, PC2, PC3, and PC4. The completion detection signals 'CD1'/'CD2' and control signals 'Z0'/'Z1' are the select lines to the 2:1MUXs.

Compared to the previous architecture, MM_IA1, the synchronization buffer stage working in phase, PC3 is replaced with adder stage in MMM_IA2. Inputs to the bit-slice 0 are the carry generated by the LSBs of the partial product (PP_0) and partial residue (S0), next significant bits of the partial product (PP_1), next significant bits of the partial residue (S1) and intermediate some output (A0) all fed to the full adder depending on the select lines of the 2:1 MUXs working in PC2. The sum of the full adder is fed to the half adder and is added to the modulus M. The carry of the full adder is fed to the full adder of the bit-slice 1. The intermediate sum (A0), generated by bit-slice 0 is fed as input to the datapath unit in the same bit-slice whereas, the carry output (C0) is fed as input to the bit-slice 1 of the datapath unit.

It should be noted that, in order to stop the carry generated by the OR gate (indicated in the red box) from being fed to the full adder as input, 'Sel' signal generated in MADU is used as the select line to the 2:1 MUX. The 'Sel' signal ensures that the output of the OR gate should be fed to the full adder only once when the calculation for each partial residue starts. It also ensures that zero is passed to the full adder after the computation of the final residue is over.

In the bit-slice 1, either the partial product (PP_2) and partial residue (S2), or the carry (C0) from the bit-slice 0 and intermediate sum (A1) are fed to the half adder. The sum output of the half adder, M and the carry generated from the full adder of the bit-slice 0 are added to the full adder of the bit-slice 1. Because the partial products are 8-bit, in the bit-slice 7, either partial residue (S8) or the sum (A7) and the carry (C6) output are fed to the half adder in the datapath unit. The sum of the half adder is then added in the next adder stage to the carry from the half adder of the bit-slice 6.

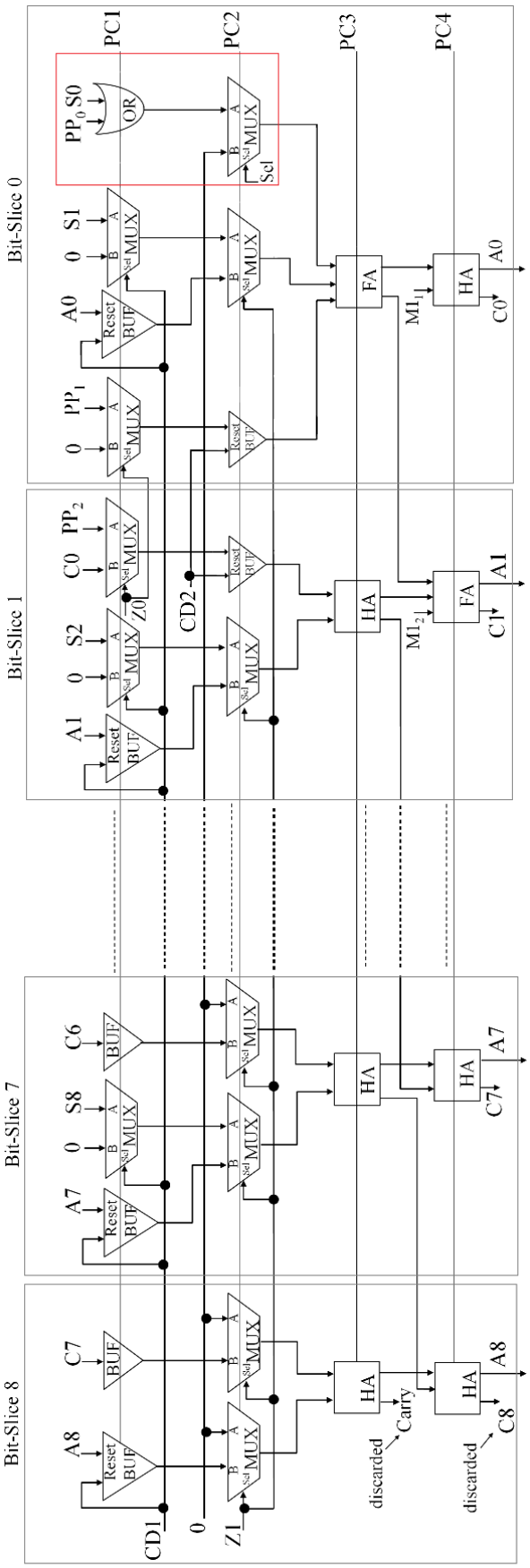


Figure 5.20: Datapath unit for MMM_IA2

In the bit-slice 8, the sum (A8) and the carry (C7) output are fed to the half adder in the datapath unit. The sum of the half adder is then added in the next adder stage to the carry from the half adder of the bit-slice 7. The carry outputs from the two HA stages are discarded as the addition of three 7-bit numbers will always give a maximum of a 9-bit number whose carry from the last bit addition will always be zero.

The sum (A0-A8) and the carry outputs (C0-C7) of the second adder stage are fed back to the datapath unit and the datapath unit works recursively until all the carry outputs are resolved to zero and the calculation of the partial residue is completed. When all the carry outputs are resolved, the datapath unit is fed with the next set of partial products and the calculated partial residue. The entire process is repeated eight times for calculating the final residue.

M Addition Decision Unit (MADU): The structure and working of MADU of MMM_RA2 (shown in Figure 5.21) are similar to that of MADU of MM_IA1. Due to the modification in the algorithm, the signal 'Sel' generated from the OR operation between 'CD1' and 'Z0' signal is used as the MUX select line in the bit-slice 0 of DU.

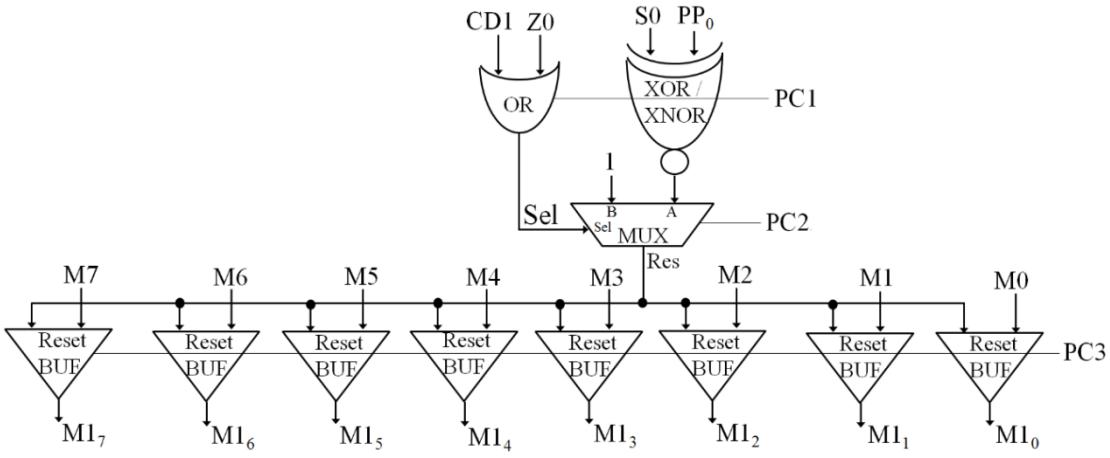


Figure 5.21: Modulus Addition Decision Unit (MADU) for MMM_IA2

5.4.3 Modified Montgomery Multiplier Iterative Architecture 3 (MMM_IA3)

From the previous two architectures, it is concluded that energy and throughput can be improved by adding more adder stages in the datapath unit. The architecture for the Montgomery multiplier presented in the previous section was further modified to improve the throughput. The number of phases the datapath unit can have (without increasing the number of synchronization buffers stages) in the architecture, depends on the power-clock

phases required by the controller unit, counter unit and 8:1MUXs in PPGU to provide partial products to the datapath unit. From the architecture of MMM_IA2, it can be observed that the datapath unit works in four power-clock phases (PC1, PC2, PC3, PC4). The addition of one more stage in the datapath unit will generate the carry and sum outputs in power-clock phase, PC1. In order to provide the carry and the sum output as the input to DU, the carry and the sum outputs have to be delayed by three stages of the synchronization buffers working in PC2, PC3 and PC4 hence increasing the synchronization buffer stages, energy and decreasing the throughput.

Therefore, to add another stage of adders in the datapath unit without increasing the number of synchronization buffer stages, the 8:1MUXs in PPGU were modified. In the previous two architectures, the 8:1 MUXs were implemented using three cascade stages of 2:1 MUXs requiring three power-clock phases. To save a power-clock phase which can be used in DU, a complex gate needs to be designed. Therefore, 8:1 MUX was implemented using 4:1MUXs and 2:1 MUX. This new design of 8:1 MUX required only two phases of the power-clock, thus saving one phase which can be used in the datapath unit to add an additional adder stage.

Moreover, the addition of an adder stage in the datapath unit removes one stage of the synchronization buffers before the RRU. Thus, MMM_IA3 uses seven stages of the synchronization buffers including RRU.

This is the optimized number of adder stages in the datapath unit for the implementation of 8-bit Montgomery multiplier. The MUXs in the PGGU has already been changed and there is no other way of saving a phase. Therefore, if the number of adder stages in the datapath unit is increased further, it will lead to an overhead due to synchronization buffers and will also increase energy dissipation and decrease the throughput.

5.4.3.1 Implementation of MMM_IA3

Figure 5.22 shows the detailed block diagram of MMM_IA3. It is similar to the architecture of MMM_IA2 shown in Figure 5.19, except for the changes in the power-clock phases of the control signals due to the modifications in PPGU and DU. The changes in signals are shown in red in the Figure 5.22.

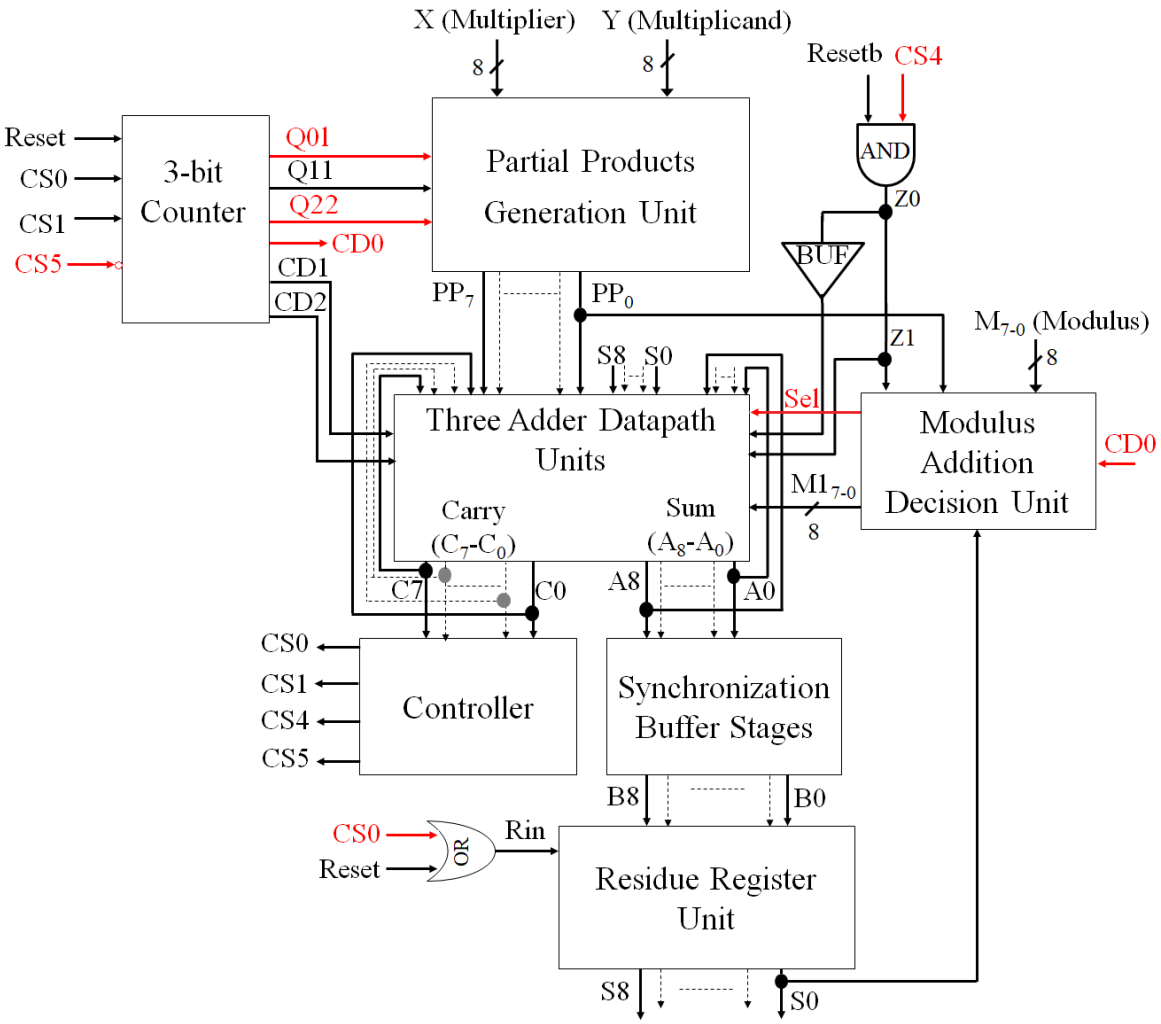


Figure 5.22: Block diagram of 8-bit MMM_IA3

Partial Product Generator Unit (PPGU): Similar to the last two architectures, PPGU consist of AND gate arrays for generating the partial products and eight 8:1 MUXs for selecting the respective partial products. For this architecture, the 8:1 MUXs are implemented using two 4:1 MUXs and one 2:1 MUXs and therefore, saves one power-clock phase. Figure 5.23 shows the part of PPGU generating LSB of the partial product with required power-clock phasing. The outputs from the counter are used as the select input to the MUXs.

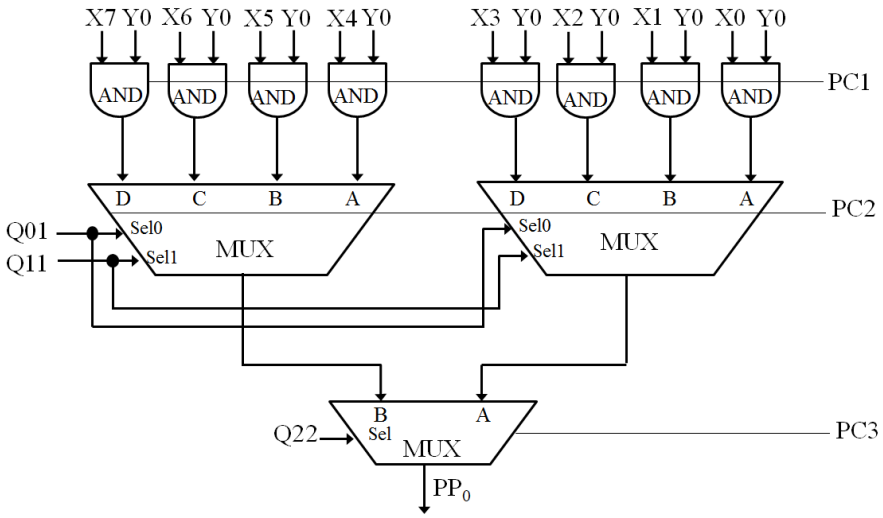


Figure 5.23: PPGU generating the LSB of the partial product

Datapath Unit: The power-clock phase saved in PPGU is used in DU of MMM_IA3 to speed up the calculation of partial residues. Figure 5.24 shows the detailed diagram of the datapath unit of MMM_IA3. DU uses five power-clock phases to generate the sum (A0-A8) and carry outputs (C0-C7). Due to the addition of an adder stage, the control signals 'Z0'/'Z1' are generated using the control signal 'CS4' (Figure 5.22), whereas the completion detection signals 'CD0' and 'CD1' are used. The working of DU is similar to the working of DU of MMM_IA2.

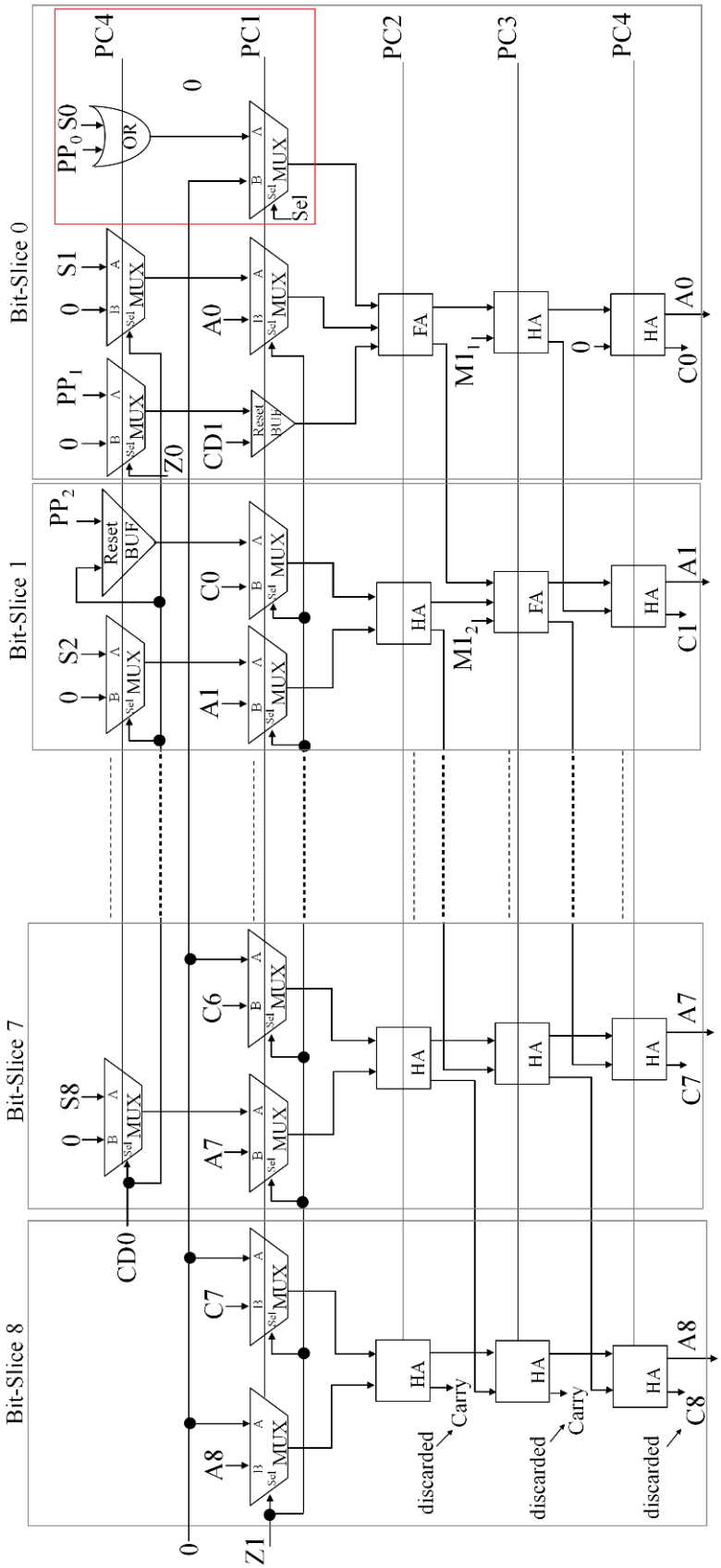


Figure 5.24: Datapath unit for MMM_IA3

Modulus Addition Decision Unit (MADU): The structure and working of MADU are similar to that of MADU of MMM_IA2. Due to the change in the power-clock phasing, signal ‘CD0’ is used in OR gate along with Z0. Figure 5.25 shows the detailed diagram of MADU along with the power-clock phasing.

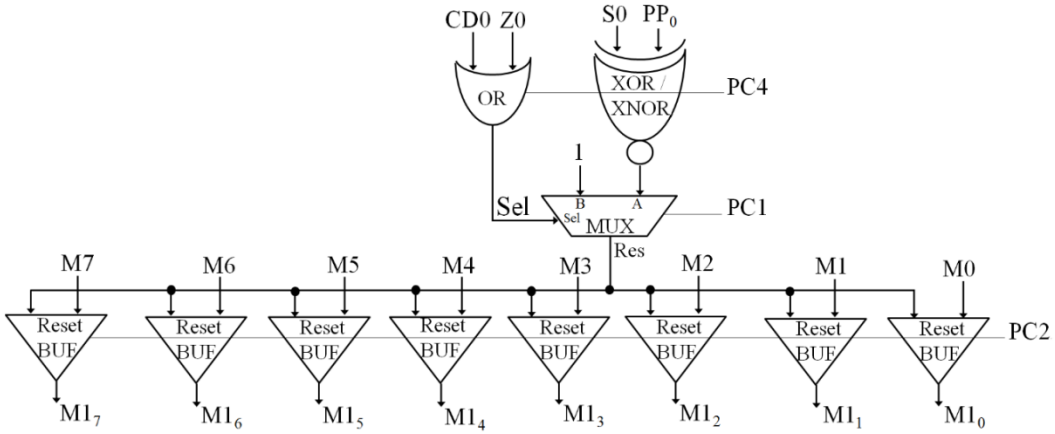


Figure 5.25: Modulus Addition Decision Unit (MADU) for MMM_IA3

Synchronization buffer Stage: As stated before, to synchronize the arrival of the partial products and partial residues, the sum outputs (A0-A8) from the DU are passed through the synchronization buffer stages. Because DU uses five level of cascade stages, the number of synchronization buffer stage is reduced by one compared to MMM_IA2. Figure 5.26 shows the detailed diagram of the synchronization buffer stages along with the power-clock phasing.

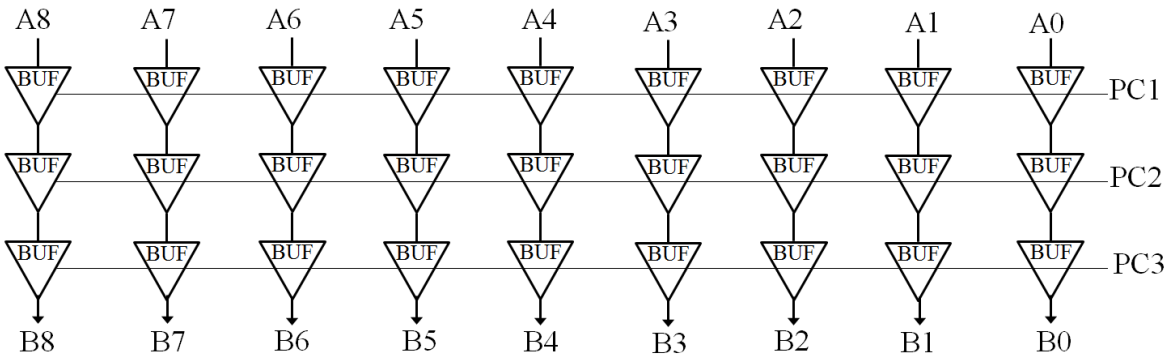


Figure 5.26: Synchronization buffer stages

Residue Register Unit (RRU): Because the synchronization buffer stages are reduced by one, the partial residues in the RRU are generated in PC3, which is a phase before in comparison to RRU of MMM_IA2. Here the reset input ‘Rin’ to the first stage of the RRU is generated by the OR operation between the control signal ‘CS0’ and ‘Reset’ shown in

Figure 5.22. The structure and working are similar to RRU of MMM_IA2. Figure 5.27 shows the detailed diagram of RRU with the power-clock phasing.

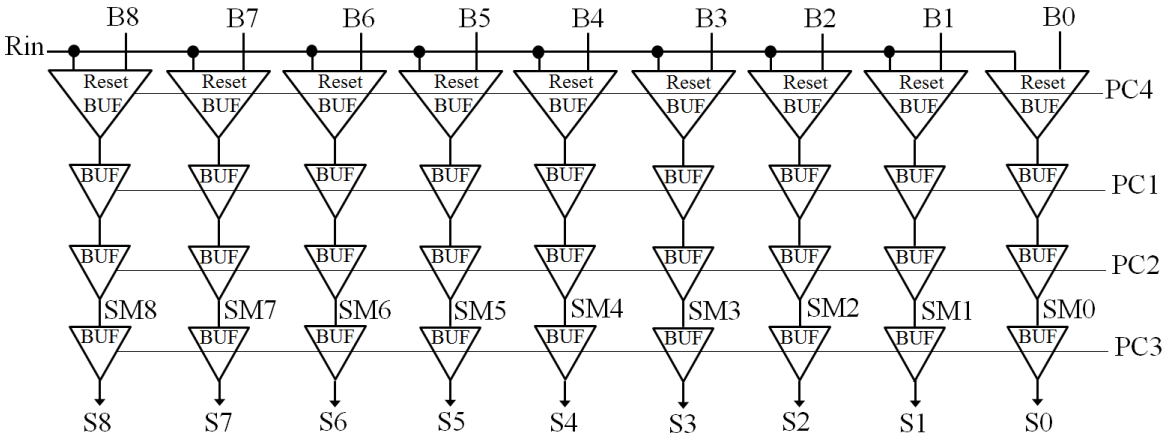


Figure 5.27: Residue Register Unit (RRU)

5.4.3.2 Methodology: Power-clock phasing in MMM_IA3

The addition of one more adder stage in DU, leads to the changes of power-clock phasing in PPGU, DU, MADU, and RRU of MMM_IA3. Whereas, the power-clock phases of the controller and the counter remain unchanged. PPGU is modified to save an extra phase that is used to add an additional adder stage in DU. DU has five levels of cascade logic gates, thus uses five power-clock phases (PC4, PC1, PC2, PC3 and PC4). The first two phases (PC4 and PC1) are used by 2:1 MUXs; and the next three phases (PC2, PC3, and PC4) are used by the adder stages. The select input, 'CD0' for the MUX working in PC4 is taken from the counter a phase before compared to MMM_IA2. The signal 'CS4' is used instead of 'CS5' for generating signal 'Z0'/'Z1' for DU and MADU. Similarly, signal 'CS0' is used instead of 'CS1' for generating reset signal 'Rin' for RRU. The sum and the carry outputs of the datapath unit are generated in phase PC4 of the power-clock which are fed back to the second stage of 2:1 MUX working in PC1 in DU for correct phasing. The total of seven phases of the power-clock are required by the controller, the counter unit and 8:1 MUXs in PPGU to send the next set of partial products to the datapath unit after the carry outputs have been resolved. For synchronization, the partial residues should also be supplied to DU, for next partial residue calculation, only after the seven power-clock phases from the sum outputs of DU. Therefore, one synchronization buffer stage was removed in MMM_IA3 and the sum outputs (A0-A8) will take seven power-clock phases, three in synchronization buffer stages and four in RRU.

Similarly, the control signals ‘Z0’/’Z1’ controls the datapath unit. The signal ‘Z0’ is generated by performing AND operation on signals ‘CS4’ and ‘Resetb’. Signal ‘Z1’ is generated by passing signal ‘Z0’ through the buffer gate. The generation of ‘CS4’ requires six power-clock phases and one power-clock phase is consumed by the generation of ‘Z0’ signal in the AND gate. In total seven power-clock phases are required to generate the control signal to synchronize the data processed in the datapath unit. This suggests that each data coming to the datapath unit has consumed seven power-clock phases.

5.5 Power-clock Gating

Clock-gating is a well-known technique to reduce the energy consumption. Because individual circuit/module usage in a system varies, not all the circuitry is used all the time, giving rise to energy reduction opportunity. One of the ways is by ANDing the clock with a gate-control signal which disables the clock to the circuit whenever the circuit is not in use, thus, avoiding energy dissipation due to unnecessary switching. Effective clock-gating, however, requires a methodology that determines which part of the system should be gated, when, and for how long.

The clock-gating scheme that either results in the frequent toggling of the clock-gated circuit between enabled and disabled states or has the clock-gating control circuitry almost as large as the blocks themselves, incur large overhead. This overhead may result in higher energy dissipation than that without clock-gating.

In adiabatic circuits, the power-clock gating is applied to switch OFF the power-clock. The straightforward implementation of power-clock gating in adiabatic circuits at circuit level is; firstly, by adding switches in the power-clock line and cutting-off the power-clock from the circuit when in the idle state [39]-[42] and Secondly, by gating the power-clock generator [68].

Power-clock gating is applied in MMM_IA3 as it is the most efficient implementation compared to other architectures implemented. In MMM_IA3, not every module is required to provide the data at all time. After the partial products and partial residues are provided to the datapath unit, it works recursively until all the carry outputs are resolved to zero. So, while the datapath is working recursively, the data from PPGU and RRU are not required. Therefore, power-clock gating is applied in these modules. Thus, by disconnecting PPGU

and RRU from the power-clock with the help of the control signal from the controller, the energy losses are reduced.

Figure 5.28 and 5.29 shows the power-clock gated PPGU and RRU respectively. The power-clocks are gated using the complementary signals of ‘CS2’, ‘CS3’, ‘CS4’ and ‘Rin’ depending on the power-clock phases of PPGU and RRU. For gating PPGU, the complementary signals of ‘CS2’ ‘CS3’ and ‘CS4’ are fed to the buffer gate to produce the gated power-clocks. Similarly, for gating the four stages of RRU, the complementary signal of ‘Rin’ is fed to the input of the buffer gate. The output of the buffer is used to power gate the first stage of RRU. For power gating, the rest of the three stages, the gated power-clock of the first stage is passed through the three buffer stages and their outputs are fed as gated power-clock to the respective RRU stages.

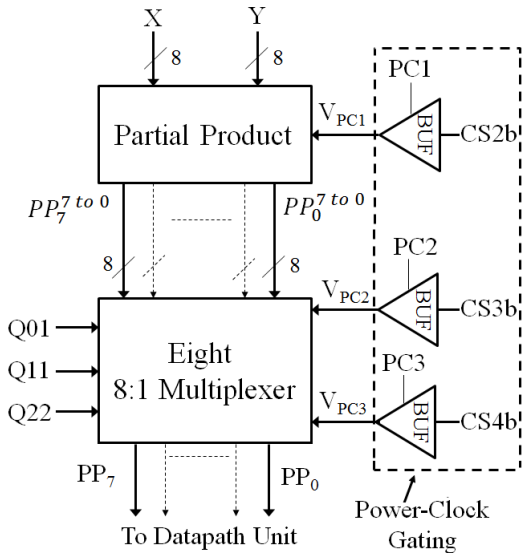


Figure 5.28: Power-clock gating applied in PPGU

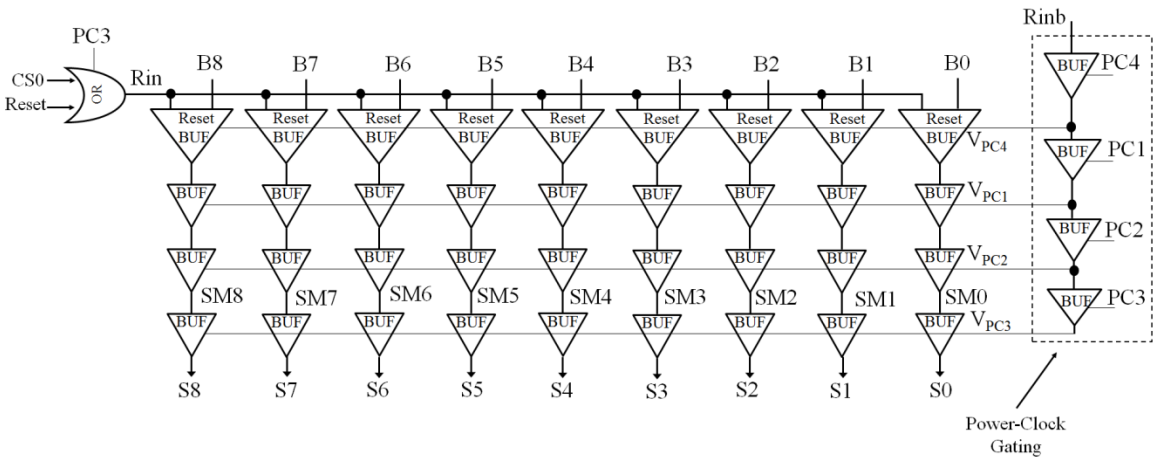


Figure 5.29: Power-clock gating applied in RRU

The output of the buffer gates (BUF) providing the gated power-clock is active only when the inputs ‘Rinb’, ‘CS2b’, ‘CS3b’ and ‘CS4b’ to the buffers are active, that is, when the carry outputs are resolved, and the calculation of partial residue is completed.

The PFAL NOT/BUF gate is used as the switch for gating the power-clock. The advantage of using the NOT/BUF gate is that it has high fan-out. Except the buffer gate providing gated power-clock to 64 AND/NAND gates (partial products) in PPGU, the rest six are sized to the minimum dimension. If a nMOS or transmission gate switch is used, the width must be increased which results in increased capacitance leading to increased adiabatic losses [37].

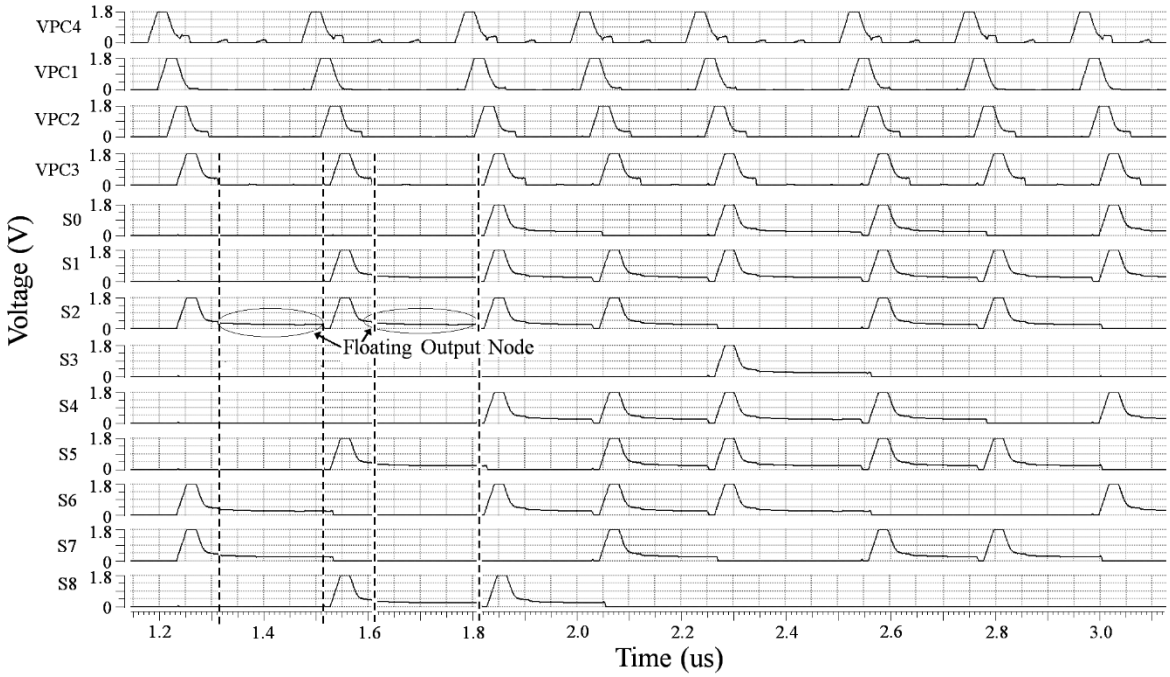
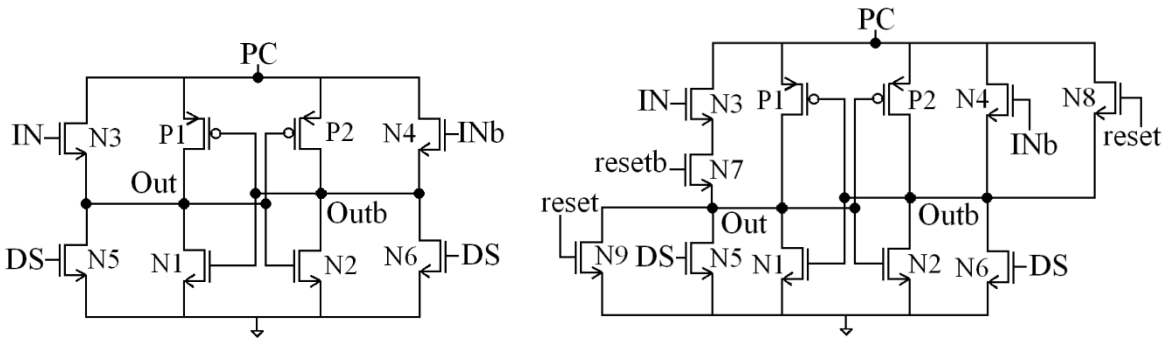


Figure 5.30: Gated power-clock and RRU outputs

In PFAL, during the recovery phase, when the power-clock falls below the threshold voltage of the pMOS transistor, the transistor is turned off and the left-over charge does not recover back to the power-clock and remains trapped at the output node. This leftover charge will discharge to the ground upon the arrival of next inputs. When the power-clock gating is applied in the cascade stages of RRU, the gated power-clock is not continuous as the power-clock. Because the output follows the gated power-clock and is fed as input to the next stage, therefore, the input and the gated power-clock to the next stage will not be continuous. As a result, the charge remains trapped causing the output node to float until the next gated power-clock and input arrives. Figure 5.30 shows the floating output node due to the leftover charges in the power-clock gated Montgomery multiplier. If the power-

clock is gated for a long time, the output nodes will remain floating for a long time too. This leads to the non-adiabatic losses and diminishes the energy savings otherwise obtained by the application of the power-clock gating.

To solve this problem, the resettable/non-resettable buffer gates used in RRU were modified such that the charge at the output nodes is discharged to ground, during the idle phase of the power-clock. For this, two nMOS transistors (N5 and N6 in Figure 5.31) each connected between the two output nodes and ground are used. The discharge transistors are turned ON during the idle phase of the power-clock and thus, allows the trapped charges at the two output nodes 'Out' and 'Outb' to discharge to ground. The modified buffer gates with discharge transistors are shown in Figure 5.31. These gates are used in the cascade stages where the power-clock gating is applied. Figure 5.32 shows the output of the power-gated Montgomery multiplier with the solution proposed.



(a) (b)

Figure 5.31: Modified PFAL NOT/BUF gate with discharge transistors (a) Non-resettable (b) Resettable.

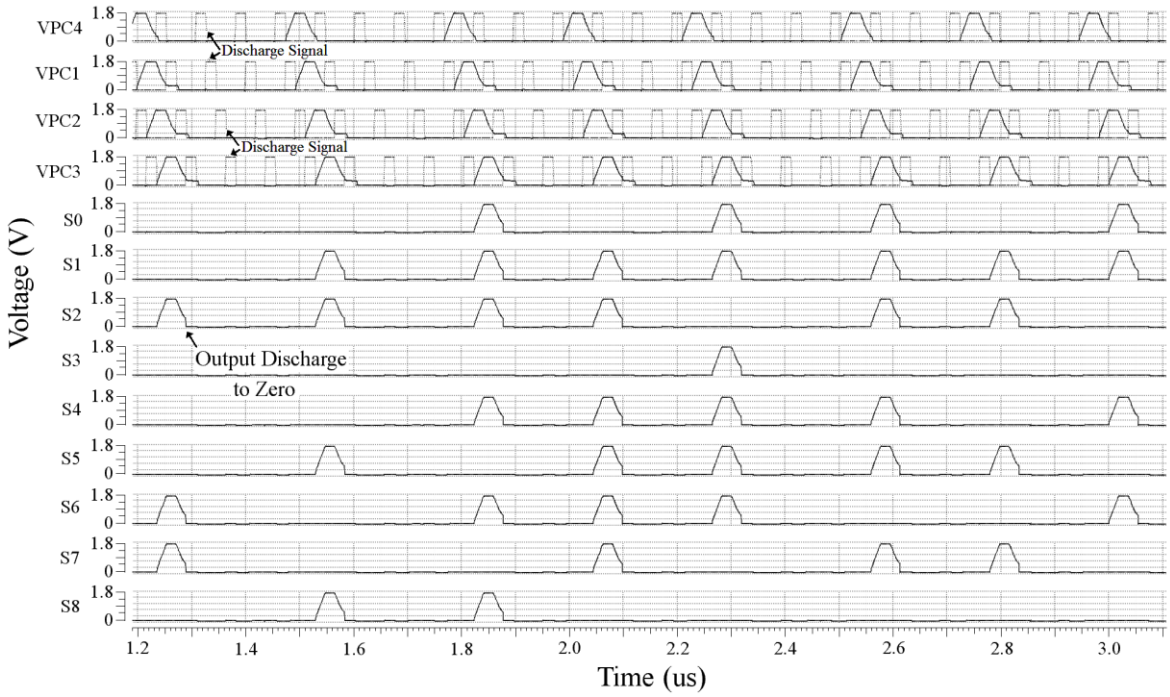


Figure 5.32: Gated power-clock and discharge input with RRU outputs.

This problem has not been reported in the literature. To the best of my knowledge, this is the first time this problem has been identified and the solution is proposed.

5.6 Simulation Results

All the adiabatic implementations, systolic and iterative approach are compared with and without considering the power-clock generator. For comparison, systolic version and iterative approach version architectures using conventional CMOS logic were also implemented.

First, the systolic array architectures, MM_SAA1, MM_SAA2 and systolic array architecture using conventional CMOS (MM_SCMOS) are compared in terms of energy, area, and throughput. Second, all the iterative approach architectures namely, MM_IA1, MMM_IA2, MMM_IA3, iterative approach architecture with power-clock gating, (MMM_IA3_PG), iterative approach architecture with proposed power-clock gating, (MMM_IA3_PPG) and iterative approach architecture using conventional CMOS (MM_RCMOS) are compared based on energy, area, and throughput.

The simulation results are shown for the 8-bit Montgomery multiplier implemented using 180nm CMOS technology at 1.8V power supply for a power-clock frequency of 1MHz, 13.56 MHz and 25MHz using Spectre simulator in Cadence EDA tool at 'Typical-Typical (TT) process corner. The output load capacitance is taken as 10fF for all the architectures. Power-clock frequency is determined by setting the clock signal (CLK) in the FSM controller which is based on Chapter 4 of this thesis. All the transistors in the adiabatic implementation of Montgomery multiplier were kept at minimum dimensions (W_{min}=220nm, L_{min}=180nm).

For the verification of the functionality, the behavioural model of the modified Montgomery multiplication algorithm in VHDL is done. The functionality is tested by providing ten random test vectors to the test bench. The average energy per computation for each of the Montgomery multiplier implementation is calculated using Spectre simulation for ten test vectors.

5.6.1 Systolic Array Architectures without PCG

Table 5.2 shows the comparison of energy dissipation per computation at 13.56MHz of MM_SAA1, MM_SAA2 and MM_SCMOS. Here the 4-phase power-clocks for the adiabatic architectures were generated using a four trapezoidal voltage source each having a 90^o phase difference. From Table 5.2, it can be seen that both adiabatic implementations outperform MM_SCMOS with proposed MM_SAA2 (using synchronization buffer reduction technique) being the most energy efficient. It shows a reduction of about 25.8% in average energy dissipation and an improvement of 3.5 power-clock cycles compared to MM_SAA1

However, MM_SCMOS consumes 3x and 4x more energy compared to MM_SAA1 and MM_SAA2 respectively, it outperforms the adiabatic implementations in terms of throughput.

Table 5.2: Comparison of energy consumption and throughput of 8-bit systolic array-based Montgomery multipliers using trapezoidal power-clock at 13.56MHz and 10fF load capacitance

Logic	Energy Consumption per Computation (pJ)	Throughput (power-clock cycles)
MM_SAA1	64.44	24
MM_SAA2	47.82	20.5
MM_SCMOS	209.6	9

5.6.2 Systolic Array Architectures with 4-phase PCG using 2, 3 and 4-step charging circuit

Table 5.3 shows the comparison of energy dissipation per computation at 13.56MHz of MM_SAA1 and MM_SAA2. Here the 4-phase PCG using 2, 3 and 4-step charging circuits are used. The energy dissipation per computation for FSM controller, Step Charging Circuit (SCC), Adiabatic Core (AC) and the total energy consumed by the adiabatic system is measured. From Table 5.3, it can be seen that MM_SAA2 is more energy efficient and shows the reduction of approximately 13%, 16% and 17% in total energy using 2-step, 3-step, and 4-step charging circuits respectively, in comparison to MM_SAA1. The energy figures confirm the energy benefits gained by the proposed synchronization buffer stage reduction technique. Also, the energy dissipation of the FSM controller increases significantly as the number of steps is increased from 2 to 4. FSM controller using 4-step charging circuit consumes the maximum energy. On increasing the number of steps, the increase in the total energy dissipation of the system diminishes the advantage of decreased energy dissipation in SCC.

Table 5.3: Comparison of energy consumption per computation of different components of the adiabatic system using systolic array architecture.

	2-step		3-step		4-step	
Energy Consumption (pJ) @ 13.56MHz	**MM_SAA1**	**MM_SAA2**	**MM_SAA1**	**MM_SAA2**	**MM_SAA1**	**MM_SAA2**
Controller	47.41	42.40	94.57	81.85	267.60	229.70
SCC	355.04	306.15	271.55	220.72	261.06	209.99
AC	30.15	29.05	27.58	23.93	26.44	23.51
Total	432.60	377.60	393.70	326.50	555.10	463.20

In other words, though the advantage of increasing the number of steps can be seen in SCC, the controller energy increases enormously, consuming the most part of the total energy in an adiabatic system.

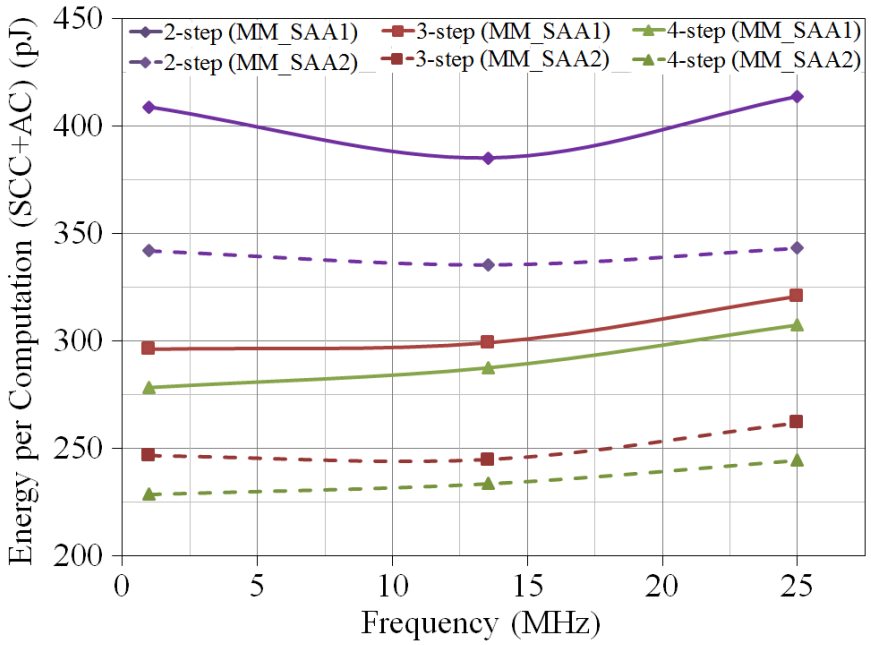


Figure 5.33: Energy consumption of step-charging circuit along with the adiabatic core for systolic array architecture

Figure 5.33 and 5.34 show the plot of energy dissipation per computation of step charging circuit including adiabatic core and the total energy dissipation per computation of the adiabatic system respectively at 1MHz, 13.56MHz and 25MHz. From Figure 5.33, it can be seen that energy dissipation of the adiabatic core along with the SCC is lowest for 4-step charging circuit and highest for 2-step charging circuit at all the simulated frequencies.

The synchronization reduction technique proves to be efficient as MM_SAA2 dissipates less in comparison to MM_SAA1. Contrary to the results shown in the plot of Figure 5.33, the energy figures shown in Figure 5.34 are opposite. Because of the high energy dissipation of the FSM controller for 4-step charging circuit, the total energy dissipation is highest for the adiabatic system using 4-step charging circuit whereas, the energy dissipation of the adiabatic system using 3-step charging circuit is the lowest. Here the 3-step constitutes an appropriate trade-off between complexity and energy dissipation.

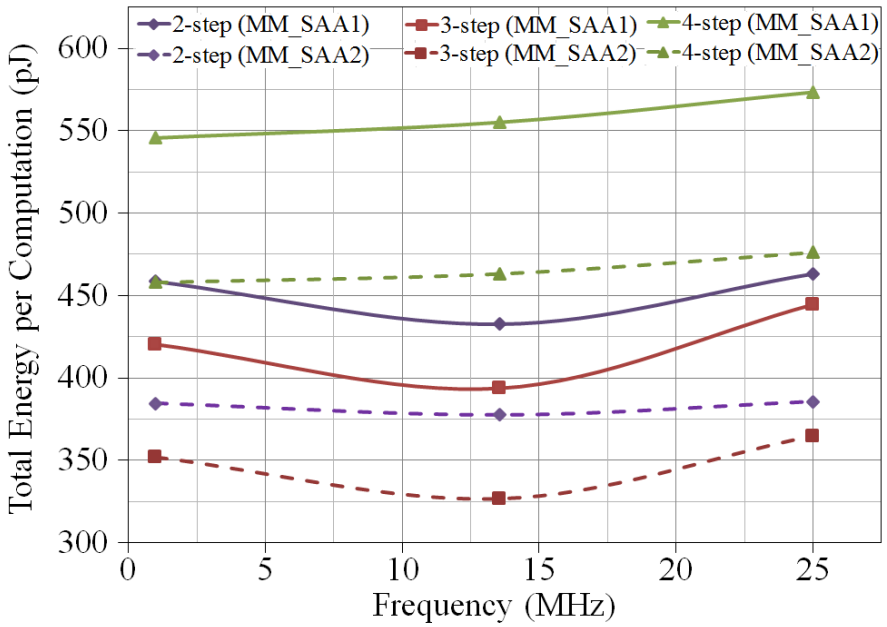


Figure 5.34: Total energy consumption of the adiabatic system for systolic array architecture.

Table 5.4 compares MM_SAA1, MM_SAA2, and MM_SCMOS based on the number of logic gates and the total number of transistors used in the implementation of 8-bit Montgomery multiplier. It can be seen that systolic array architecture using conventional CMOS (MM_SCMOS) uses the lowest transistor count approximately 28% and 26% less compared to MM_SAA1 and MM_SAA2 respectively. It is followed by MM_SAA2 which shows a reduction of approximately 3% compared to MM_SAA1. It should be noted that the data in Table 5.4 includes number of transistors used in the core.

Table 5.4: Comparison of transistor count of 8-bit Montgomery multiplier systolic array architecture.

Logic	Number of logic gates in Montgomery Multiplier		Number of transistors per gate	Total number of transistors (Approx.)
MM_SAA1	NOT/BUF AND/NAND2 XOR/XNOR2 Reset NOT/BUF HA FA	338 64 63 150 421 42	6 8 10 9 18 34	13,526
MM_SAA2	NOT/BUF AND/NAND2 XOR/XNOR2 Reset NOT/BUF HA FA	269 64 63 150 421 42	6 8 10 9 18 34	13,112
MM_SCMOS	NOT AND2 XOR2 Flip-flops 2:1 MUX HA FA	8 64 71 236 129 421 56	2 4 12 14 6 8 20	9,682

5.6.3 Iterative Architectures without PCG

The comparison of energy dissipation per computation of 8-bit Montgomery multiplier of MM_IA1, MMM_IA2, MMM_IA3, MMM_IA3_PG, MMM_IA3_PPG, and MM_RCMOS is shown in Table 5.5. The results are based on the 4-phase power-clocks, each having a 90^{o} phase difference, generated using a trapezoidal source at 13.56MHz. From Table 5.5, all the adiabatic implementations outperform MM_RCMOS in terms of energy dissipation. Amongst non-power-gated architectures, MMM_IA3 is the most energy efficient followed by MMM_IA2 and MM_IA1. It shows an energy reduction of 26% and 11% and improvement of 10 and 3 power-clock cycles when compared to MM_IA1 and MMM_IA2 respectively.

As mentioned before architectural level power-clock gating was applied on MMM_IA3. MMM_IA3_PG (power-clock gated MMM_IA3) shows an improvement of about 24% in energy dissipation compared to MMM_IA3. Energy dissipation was further reduced by applying the proposed solution to the power gating. MMM_IA3_PPG consumes the lowest

energy in comparison to all the adiabatic implementations shown in Table 5.5. MMM_IA3_PPG shows an improvement of about 13.25% and 34.13% in the energy dissipation in comparison to MMM_IA3_PG and MMM_IA3 respectively. Also, energy dissipation of all the iterative approach architectures is much less in comparison to the energy dissipation of the two systolic array architectures.

However, MM_RCMOS outperforms all the adiabatic implementations in terms of throughput, it consumes 16x and 10x more energy compared to MMM_IA3_PPG and MMM_IA3 respectively.

Table 5.5: Comparison of energy consumption and throughput of 8-bit Iterative Montgomery multipliers using trapezoidal power-clock at 13.56MHz and 10fF load capacitance

Logic	Energy Consumption per Computation (pJ)	Throughput (power-clock cycles)
MM_IA1	24.743	38
MMM_IA2	20.504	31
MMM_IA3	18.17	28
MM_RCMOS	180	14
MMM_IA3_PG	13.796	28
MMM_IA3_PPG	11.968	28

5.6.4 Iterative Architectures with 4-phase PCG using 2, 3 and 4-step charging circuit

Table 5.6 shows the comparison of energy dissipation per computation of MM_IA1, MMM_IA2, and MMM_IA3. Here the 4-phase PCG using 2, 3 and 4-step charging circuits are used. The energy dissipation per computation for FSM controller, Step Charging Circuit (SCC), Adiabatic Core (AC) and the total energy consumed by the adiabatic system is measured at 13.56MHz. From Table 5.6, it can be seen that MMM_IA3 exhibits the lowest energy dissipation using 2, 3 and 4-step charging circuit. Compared to MM_IA1, MMM_IA2 and MMM_IA3 show an improvement of approximately 19% and 28% respectively in energy dissipation for all the given step charging circuits.

It can also be seen that the total energy dissipation increases as the number of steps in the

step charging circuit is increased. FSM controller using 4-step charging circuit consumes the maximum energy and thus has the highest total energy of the adiabatic system. However, there is no significant difference in the total energy dissipated by the adiabatic system using 2 and 3 step charging circuit. Overall, the adiabatic system using 2 step charging circuit has the lowest energy dissipation.

From Tables 5.3 and 5.6, it should be noted that the energy dissipated by SCC is more for systolic array architectures compared to iterative approach architectures. It is because, systolic array architectures having more number of transistors compared to iterative approach architectures, presents a large load to SCC. Thus, in a large system (systolic array architecture) the energy consumption of the SCC dominates over the energy of synchronous FSM controller for 2 and 3-step charging circuit and comparable to 4-step charging circuit. Whereas, in case of iterative approach architectures, the energy dissipation of the synchronous FSM controller dominates the energy dissipation of SCC in 3 and 4-step charging circuit. This suggests that in a large adiabatic system the losses due to the FSM controller in comparison to the overall losses will be negligible.

Table 5.6: Comparison of energy consumption per computation of different components of adiabatic system using iterative architecture

Energy Consumption (pJ) @ 13.56MHz	2-Step			3-Step			4-Step		
	MM_IA1	MMM_IA2	MMM_IA3	MM_IA1	MMM_IA2	MMM_IA3	MM_IA1	MMM_IA2	MMM_IA3
Controller	80.54	65.98	59.28	141.30	118.40	108.10	425.80	348	313.1
SCC	145.95	118.94	104.23	105.87	86.02	75.60	87.30	70.71	62.65
AC	20.01	16.38	14.99	16.23	13.48	12.10	14.70	12.49	11.05
Total	246.50	201.30	178.50	263.40	217.90	195.80	527.80	431.20	386.80

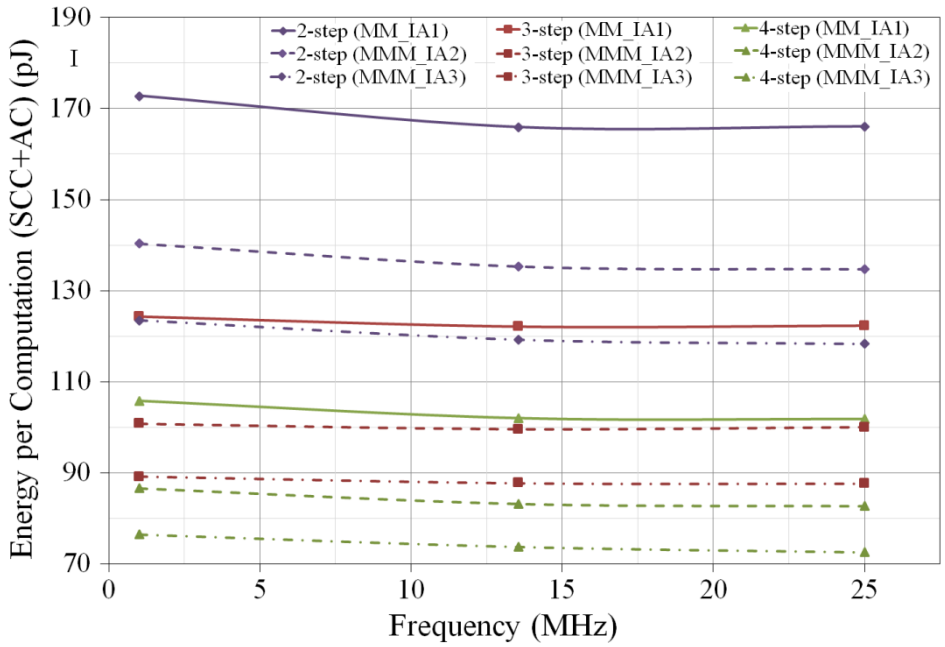


Figure 5.35: Energy consumption of step-charging circuit along with the adiabatic core for iterative architecture.

Figure 5.35 and 5.36 shows the plot of energy dissipation per computation of step charging circuit including adiabatic core and the total energy dissipation of the adiabatic system respectively at 1MHz, 13.56MHz and 25MHz. From Figure 5.35, energy dissipation per computation of SCC+AC is lowest for 4-step charging circuit and highest for 2-step charging circuit at all the simulated frequencies.

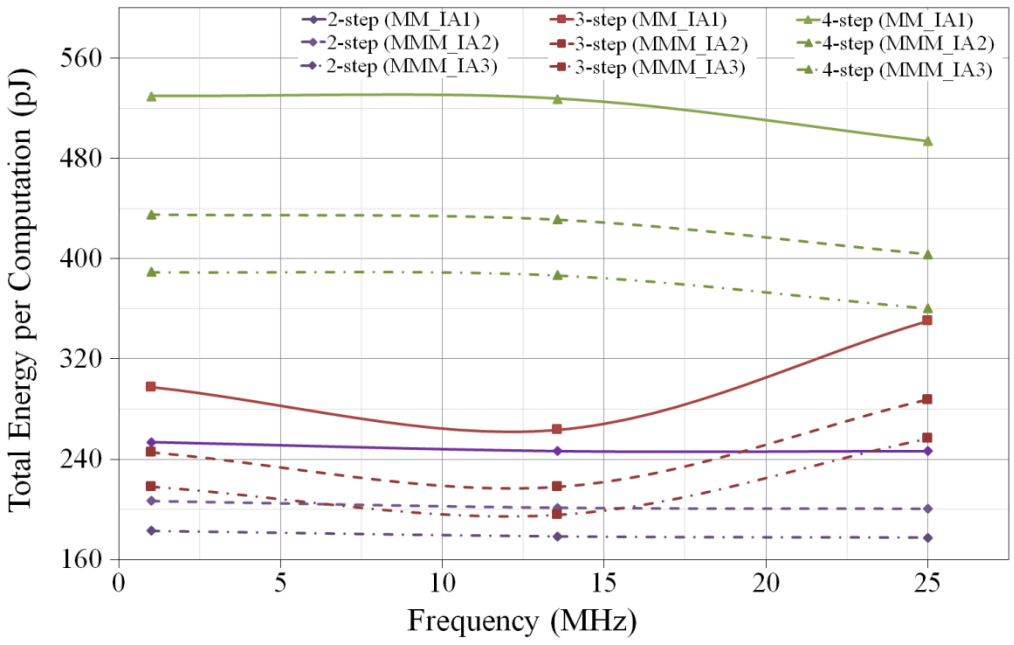


Figure 5.36: Total energy per computation of the adiabatic system for iterative architecture.

Contrary to the results shown in the plot of Figure 5.35, the energy dissipation shown in Figure 5.36 is opposite. Because of the high energy dissipation of the FSM controller for 4-step charging circuit, the total energy dissipation of the adiabatic system using 4-step charging circuit is highest whereas, the energy dissipation using 2-step charging circuit is the lowest. Although the comparison of the adiabatic system to the conventional CMOS is unfavourable as latter doesn't include the energy dissipation due to the clock generation and distribution network, MMM_IA3 with 4-phase PCG using 2-step charging circuit dissipates almost same energy as that of conventional CMOS.

Table 5.7 compares MM_IA1, MMM_IA2, MMM_IA3, MMM_IA3_PG, and MM_RCMOS based on number of transistors used in the implementation of 8-bit Montgomery multiplier. It can be seen that MM_RCMOS uses the lowest transistor count followed by MMM_IA3, MMM_IA3_PG, MMM_IA2, and MM_IA1. The proposed power-clock gated (MMM_IA3_PPG) uses 72 more transistors in RRU compared to MMM_IA3_PG.

Table 5.7: Comparison of transistor count of 8-bit Montgomery multiplier iterative approach Architectures.

Logic	Number of logic gates in Montgomery Multiplier	Number of transistors per gate	Total number of transistors (Approx.)
MM_IA1	NOT/BUF 108 AND/NAND2 76 AND/NAND3 1 OR/NOR2 2 OR/NOR3 6 XOR/XNOR2 3 Reset NOT/BUF 38 2:1 MUX 86 HA 2 FA 8	6 8 10 8 10 10 9 10 18 34	2,882
MMM_IA2	NOT/BUF 89 AND/NAND2 76 AND/NAND3 1 OR/NOR2 2 OR/NOR3 6 XOR/XNOR2 3 Reset NOT/BUF 35 2:1 MUX 84 HA 11 FA 7	6 8 10 8 10 10 9 10 18 34	2,857
MMM_IA3	NOT/BUF 76 AND/NAND2 76 AND/NAND3 1 OR/NOR2 2 OR/NOR3 6 XOR/XNOR2 3 Reset NOT/BUF 25 2:1 MUX 37 4:1 MUX 16 HA 20 FA 7	6 8 10 8 10 10 9 10 22 18 34	2,733
MMM_IA3_PG	NOT/BUF 82 AND/NAND2 76 AND/NAND3 1 OR/NOR2 2 OR/NOR3 6 XOR/XNOR2 3 Reset NOT/BUF 25 2:1 MUX 37 4:1 MUX 16 HA 20 FA 7	6 8 10 8 10 10 9 10 22 18 34	2,775
MMM_CMOS	NOT 71 NAND2 68 NAND3 7 NOR2 3 NOR3 7 XOR2 3 Flip-flops 20 2:1 MUX 101 HA 11 FA 7	2 4 6 4 6 12 14 6 8 20	1,660

5.7 Chapter Summary

In this chapter, several adiabatic implementations of Montgomery multiplication algorithm are presented. In particular, systolic array and iterative approach architectures are considered. A synchronization overhead reduction technique to reduce the area and energy overhead due to synchronization buffers in the systolic array architecture is proposed. Due to the solution, the area, throughput and energy are improved by 414 transistor count, 3.5 power-clock cycles and 25.8% respectively.

Also, three scalable, area and energy efficient iterative approach architectures using single, 2 and 3 adder stage in the datapath unit are proposed along with their methodology and the optimum number of adder stages in the datapath unit for 8-bit Montgomery multiplier is investigated. In MM_IA1, the replacement of the synchronization buffers into adders is the clever step to increase the speed of the computation. Also, an additional adder stage in the datapath unit of the MMM_IA3 is made possible by carefully examining the phases used by the controller unit, counter unit and 8:1MUXs in PPGU. The modification in Montgomery modular multiplication algorithm proved to be area and energy efficient. Amongst the three iterative approach architectures, MMM_IA3 is the most energy efficient followed by MMM_IA2 and MM_IA1.

Lastly, to shut the periodically running units of the Montgomery multiplier architecture, power-clock gating is applied. A problem due to the application of power-clock gating in cascade stages of adiabatic gates is identified and a solution is proposed. Power-clock gating was applied on MMM_IA3, and a reduction of about 24% is obtained in comparison to the energy dissipation of MMM_IA3. Using the proposed solution in the power-clock gating (MMM_IA3_PPG), a reduction of about 34.13% was obtained in comparison to the energy dissipation of MMM_IA3.

In comparison to systolic array architectures, iterative approach architectures are more energy efficient. FSM controller for 4-phase PCG using 4-step charging circuit consumes the maximum energy and thus has the highest total energy for both systolic array architecture and iterative approach architectures. The energy of the controller for 4-step charging circuit increases enormously compared to the energy dissipated by the FSM controllers of 2 and 3-step charging circuits.

Also, the energy dissipated by the step charging circuits is less for iterative approach architectures compared to the systolic array architectures. It is because the large adiabatic core in systolic array architectures presents a large load to the step charging circuits. Thus, the energy consumption of the SCC in the systolic array architecture dominates over the energy of synchronous FSM controller for 2 and 3-step charging circuit and comparable to 4-step charging circuit. Whereas, in iterative approach architectures, the energy dissipation of the synchronous FSM controller dominates the energy dissipation of the step charging circuit (SCC) in 3 and 4-step charging circuit. This suggests that in a large adiabatic system the losses due to the FSM controller in comparison to the overall losses will be negligible.

6. Power Analysis Attack Resilient Adiabatic logic

This chapter introduces the general background of Power-Analysis Attack (PAA) resilient logic designs, followed by the review of currently known secure adiabatic logic designs and a summary of their shortcomings. As a solution to the shortcomings, a novel power PAA resilient adiabatic logic is proposed. A detailed performance evaluation of the proposed logic is performed. The proposed logic is compared with the three existing secure adiabatic logic designs at gate level and in a complex circuit design (Montgomery multiplier design). The impact of frequency variations, process corner variations and power-supply scaling on the resistance of the proposed and the existing secure diabatic logic designs are investigated.

6.1 Introduction

In the present information and communication technology-based world, security of the information is a fundamental requirement. The security is usually ensured by the cryptography algorithms which are based on hard to solve mathematical problems. However, a hardware implementation of the cryptographic algorithms can leak information that can be exploited by the adversary to know the secure information stored on the device. Such attacks are called as Side Channel Attacks (SCA) and are defined as attacks based on the information leaked from the hardware implementation of the cryptosystem, rather than the theoretical weaknesses in the algorithm. For example, power consumption [7], timing information [69], or electromagnetic emissions [70], [71] provide a source of information that can be exploited to reveal the secret key used during the critical operations such as encryption and decryption. These attacks have been demonstrated extremely powerful to defeat various cryptographic algorithm implementations (e.g. secret key and public key) on different platforms such as smart cards, ASICs and FPGAs [72].

The Power Analysis Attacks (PAA) [7], [73] are the type of SCA that have received the most attention in recent years. In PAA, the adversary attempts to reveal secret information such as secret key, on the basis of the cryptographic device's power consumption during the execution of the critical operations such as encryption and decryption. In particular, the attacker attempts to find the relationship between the instantaneous power consumption and the internal states of the cryptographic device. PAA such as Simple Power Analysis (SPA) and Differential Power Analysis (DPA) [73], [74] have become a major threat to the security of cryptographic implementations. SPA attack involves monitoring power fluctuations of the cryptographic circuit. DPA attacks, on the other hand, are more advanced than SPA attacks and use statistical methods and digital processing techniques on a large number of monitored power signals. Such methods reduce noise and enhance the signals making it easier to distinguish between zero and one.

The strength of the PAA comes from the fact that the power consumption of the cryptographic device depends on the intermediate values processed in it. Therefore, if the power consumption of the cryptographic device can be made independent of the intermediate values, the PAA can be made difficult. There are various countermeasures that have been proposed in the literature [68]-[86] to protect cryptographic implementations against PAA and are employed at the cell (gate) level. Hiding [75], [76] and masking [77]-[79] are amongst the most common countermeasures applied at the cell level. The goal of hiding is to make the power consumption of the cryptographic device independent of the data processed. This means that the cryptographic device's power consumption characteristics are changed so that every operation consumes approximately same amount of energy. Masking, on the other hand, relies on randomizing the input dependent (key-dependent) intermediate values processed during the execution of the cryptographic device. This method randomizes the power consumption of the cryptographic device and thus makes it largely independent of the actual intermediate values.

6.2 Background

The idea of using a countermeasure at the cell level is to build the cryptographic device using gates which are resilient to PAA. It is because the power consumption of the cryptographic device is the total of the power consumption of its gates. If the power consumption of the gates is made independent of the input data, the cryptographic device

can be made resilient to PAA. There are several papers that have addressed the design of PAA resistant logic designs [76], [80]-[94] such as Sense-Amplifier-Based Logic (SABL) [76], and Wave Dynamic Differential Logic (WDDL) [80], are dual-rail pre-charge logic styles. The power consumption of SABL [76] is constant under the assumption that the differential outputs of the gate drive the same capacitive load. Thus, it needs a full-custom design tool to equalize the capacitances of the complementary wires. WDDL [80], on the other hand, was designed to avoid the usage of a full custom design tool. Masked Dual-rail Pre-charge Logic (MDPL) [81], [82], Dual-rail Random Switching Logic (DRSL) [83] was implemented by combining the masking scheme and the dual-rail pre-charge logic in order to use semi-custom design tools without routing constraints. These logic styles use a random number generator to prepare the mask bits for the masking operation. Three-phase Dual-rail pre-charged logic (TDPL) [84] is based on a three-phase operation and needs an additional discharge phase after pre-charge and evaluation in order to obtain constant energy consumption. The power consumption of the (TDPL) is insensitive to unbalanced load conditions at the output nodes. Thus allows adopting semi-custom design flow. TDPL gate contains three control signals, and hardware implemented using TDPL style needs a separate unit to schedule control signals in order to prevent the glitches. All of the above-mentioned PAA resistant logic designs applied conventional CMOS logic operation and thus dissipate high energy.

There are several energy efficient PAA resistant logic designs based on the adiabatic logic [87]-[92] such as Charge-Sharing Symmetric Adiabatic Logic (CSSAL) [87]-[90], Symmetric Adiabatic Logic (SyAL) [91], and Secure Quasi-Adiabatic Logic (SQAL) [92]. These design styles use charge sharing technique at the output/internal nodes and load balancing at the two output nodes to guarantee data-independent energy dissipation. SyAL and SQAL are based on Efficient Charge Recovery Logic (ECRL) [31], [32]. The difference between SyAL and SQAL is in the number of charge sharing transistors used. Alternatively, CSSAL is based on 2N-2N2P [17] and is an enhancement of SyAL. CSSAL consumes more energy and has a complex structure (uses two additional inputs in the gate). SyAL, SQAL, and CSSAL suffer from Non-Adiabatic Losses (NAL) during the evaluation phase of the power-clock. Also, their structure is asymmetric. Asymmetric structure makes the logic vulnerable to PAA. As a solution to the problems, a logic called as Without Charge Sharing Quasi Adiabatic Logic (WCS-QuAL) is proposed which has symmetric structure and does not require any charge sharing input.

For measuring the resistance against PAA, percentage Normalised Energy Deviation (%NED) and percentage Normalised Standard Deviation (%NSD) are used as the security metrics [86].

PAAs are performed at the physical level. The aim of a simulation-based security evaluation is to get some insights on a physical attack without performing the actual attacks, by carefully investigating the higher abstraction levels. Each time the abstraction level is decreased, new imperfections may appear in the design, possibly increasing the amount of information leaked to the adversary. Thus, it is important to note that resistance to PAA cannot be isolated at one abstraction level. A countermeasure that can be confirmed secure at a high abstraction level and at a certain environmental condition is not necessarily secure when gate delays, load capacitances, process variations, and frequency of operation are taken into account [95]. For example, certain masking logic schemes work fine at the logic level but fall under attack due to glitching activities if the circuit environmental conditions are changed [96]. Similarly, dual-rail logic designs are highly dependent on the output load balancing and thus on how perfectly the capacitances and the resistances of the interconnect wires are balanced [72], [97]-[99]. The best security evaluation is performed at the physical level, using actual measurements. However, this doesn't mean that the simulation-based evaluation is meaningless; rather it only presents the picture of a part of the physical reality which needs to be confirmed by subsequent analysis at lower abstraction level. Thus, it is important to perform the simulation-based evaluations exhaustively. Under process variations [72], [95], [97]-[99], many existing gate level countermeasures against PAA can fail and is an additional factor that can deteriorate the resistance against PAAs of all kinds of PAA resilient logic styles. Therefore, it is important to evaluate the performance of the proposed logic and the existing secure adiabatic logic designs under process corner variations and the impact of process corner variations on the %NED and %NSD.

In adiabatic circuits, adiabatic losses (AL) increase as the frequency of operation is increased (shorter ramping time). Therefore, it would be worth evaluating the performance of the proposed logic, WCS-QuAL and the existing secure adiabatic logic designs at different frequencies to analyze the effects on %NED and %NSD.

As mentioned earlier the constant power consumption in secure adiabatic logic designs is guaranteed by charge-sharing and the output nodes load balancing. In schematic design, the resistances and capacitances of the wires are not taken into consideration. Therefore, it is

important to analyze and evaluate the effect of resistances and capacitances of the wires on the %NED and %NSD in the post-layout simulations.

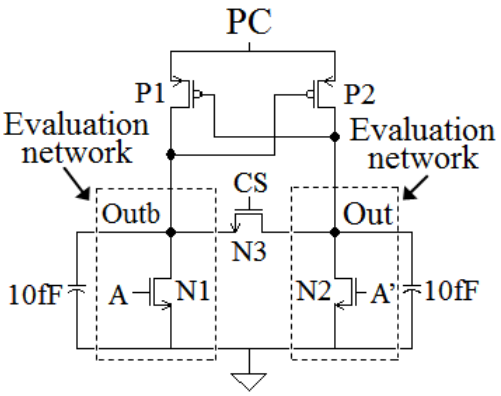
For a realistic and fair evaluation, it is also important to evaluate the performance of WCS-QuAL and existing logic in a complex system. Thus, an 8-bit Montgomery multiplier using WCS-QuAL, CSSAL, SQAL, and SyAL was implemented as a vehicle to evaluate and compare the resistance against PAA.

6.3 Existing Secure Adiabatic Logic Designs

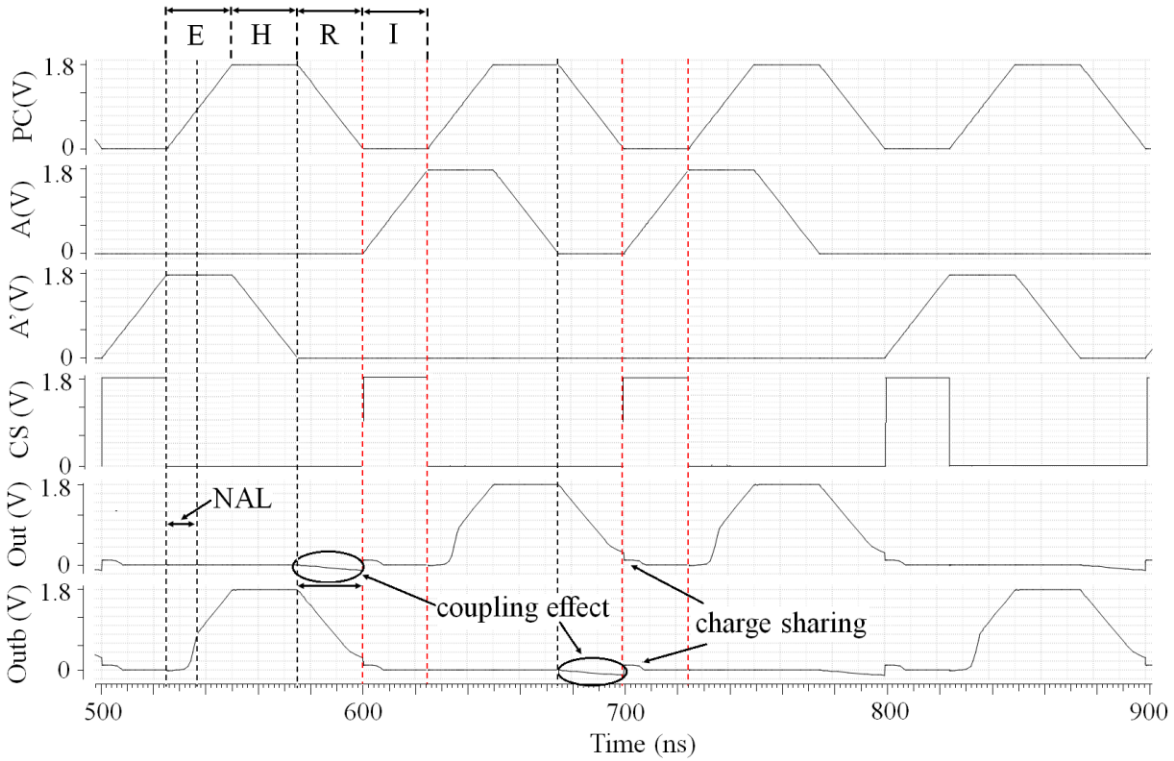
Charge sharing transistors are used to remove the remaining charge (from the evaluation of the previous inputs) from the output nodes. They are turned ON during the Idle (I) phase of the power-clock before the evaluation of the next input takes place. Existing secure adiabatic logic designs use a different number of charge sharing transistors to achieve data-independent energy dissipation.

6.3.1 Secure Quasi-Adiabatic Logic (SQAL)

The NOT/BUF gate using SQAL [102] and SyAL [101] has identical structure. Figure 6.1 (a) and (b) shows the schematic of the SQAL/SyAL NOT/BUF gate and its simulation results at 10MHz respectively. The operation of the SQAL/SyAL gate is explained through the design of a NOT/BUF gate. The simulation results shows the Power-Clock (PC), input A, its complement A', charge sharing input, CS and the output nodes 'Out' and 'Outb'.



(a)



(b)

Figure 6.1: (a) SQAL/SyAL NOT/BUF gate[92], [91] (b) Simulation results at 10MHz

As mentioned before, SQAL is based on ECRL [31], [32] adiabatic logic and works on 4-phase power-clocking scheme. The evaluation networks are connected between the two output nodes and the ground and are indicated in the schematic of the NOT/BUF gate of Figure 6.1 (a). The Transistors N1 and N2 are input transistors whereas; N3 is the charge sharing transistor. P1 and P2 are the cross-coupled transistors responsible for holding the output nodes to their respective voltages. The operation is explained for A='1', A'= '0'.

During the Idle (I) phase of the Power-Clock (PC), when the input A is rising, transistor N1 turns ON after the input reaches the threshold voltage and drives the output node 'Outb' to zero. Also, the charge sharing transistor, N3, turns ON which connects the output node, 'Out' to ground through the transistor, N1. In this manner, both the output nodes are discharged to ground before the evaluation of the next inputs.

During the Evaluation (E) phase, the charge sharing transistor N3 is turned OFF. The input A is logic '1' (A' is logic '0') and the PC ramps up. The output node, 'Outb' will be connected to ground and the node 'Out' will follow the PC through transistor P2. The output node cannot instantly follow the rising PC, only when the PC has reached the

threshold voltage of the cross-coupled pMOS transistor P2, the output node follows the PC abruptly, leading to Non Adiabatic Loss (NAL).

During the Hold (H) phase, the input A ramps down and the transistor N1 is switched OFF when the gate-to-source voltage falls below the threshold voltage, $V_{tn.}$ The output nodes 'Out' and 'Outb' are held at their respective voltages due to the cross-coupled transistors P1and P2.

During the Recovery (R) phase, the input transistors are OFF and the PC ramps down. The charge on output node 'Out' will be recovered back to the PC through P2. The charge is recovered till PC falls below the threshold voltage, $|V_{tp}|$ of P2. The node 'Out' stays at V_{tp} when P2 is turned OFF, leading to NAL. Thus, SQAL and SyAL suffer from NAL during the evaluation and the recovery phase of the PC. The leftover charge will be discharged to ground in the idle phase when the charge sharing transistor is turned ON.

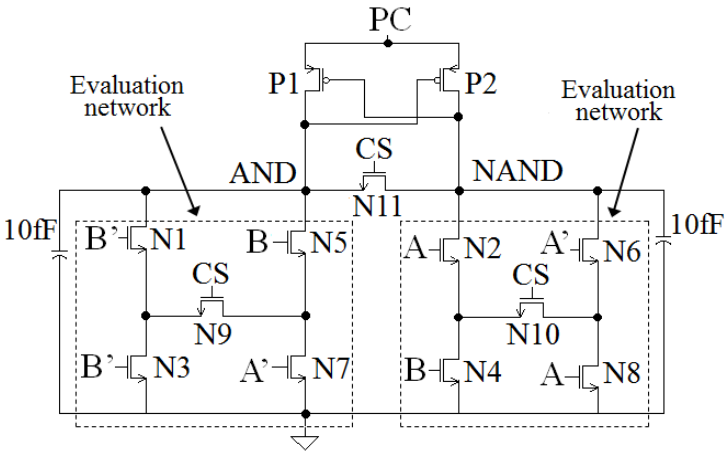
Also, during the Recovery phase, when the input transistor N1 is turned OFF, the output node 'Outb' gets coupled to the node 'Out' (following the PC) due to the parasitic capacitance between the gate and the source of the pMOS transistor and thus suffers from coupling effect.

If the charge sharing transistor, N3 is removed in SQAL/SyAL NOT/BUF gate, the charge will remain trapped on the output node. For instance, From Figure 6.1(a), during the idle phase, when input A is logic '1', the output node, 'Outb' will be connected to ground. Whereas, the output node, 'Out' will have the left-over charge on it. When in the next cycle of PC, if the input doesn't change, the leftover charge on the output node, 'Out' will remain trapped. The charge will be discharged to zero if the inputs change in the next cycle.

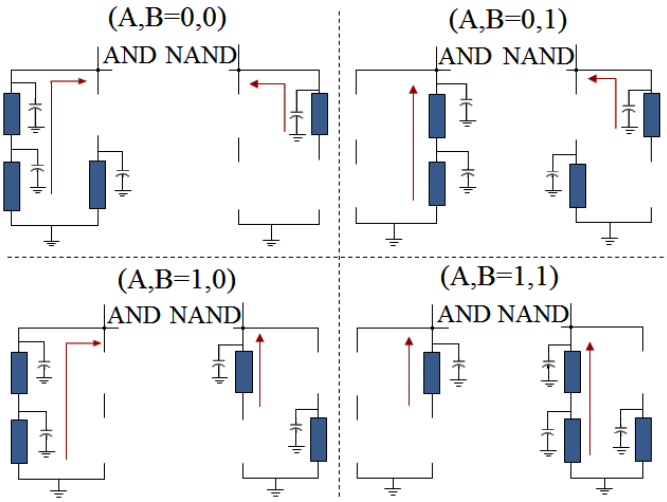
Charge sharing input is a clock input of 25% duty cycle and used to discharge the two output nodes to the ground before the next inputs are evaluated. As mentioned before, because SQAL works on 4-phase power-clocking scheme; 4-phases of the charge sharing input need to be generated. This incurs the overhead of generation and routing of the additional input. More the number of charge sharing transistors used in the secure logic design, more the overhead of routing the 4 phases of the charge sharing input.

Figure 6.2 (a) and (b) shows the AND/NAND gate using SQAL and its equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. It can be seen that, for each of the 4 input combinations, the capacitance at the

two output nodes is different. For instance, From Figure 6.2 (a) it can be seen that for input combination AB=00, the output node, 'AND' has three transistors ON whereas, output node 'NAND' has one transistor ON. For AB=01 and 10, the output nodes, 'AND' and 'NAND' each have 2 transistors ON. For AB=11, the output node, 'AND' has one transistor ON whereas, output node 'NAND' has three transistors ON. This makes the power consumption of the SQAL data dependent and the structure of the gate asymmetric.



(a)



(b)

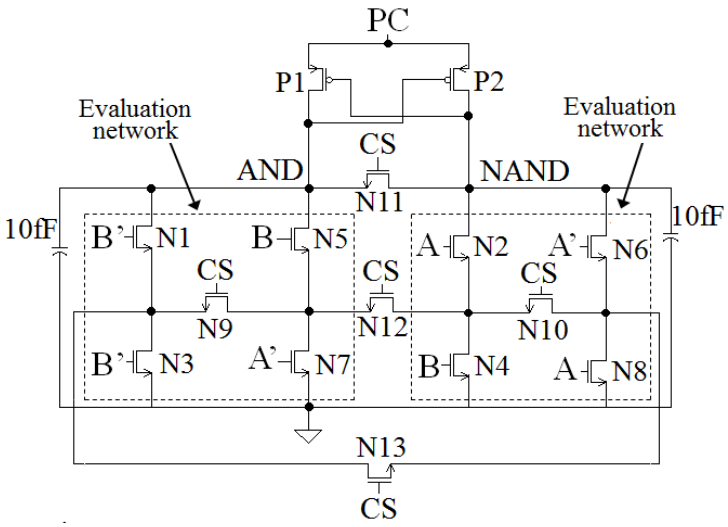
Figure 6.2: (a) AND/NAND gate using SQAL [92] (b) Equivalent RC models during evaluation phase

6.3.2 Symmetric Adiabatic Logic (SyAL)

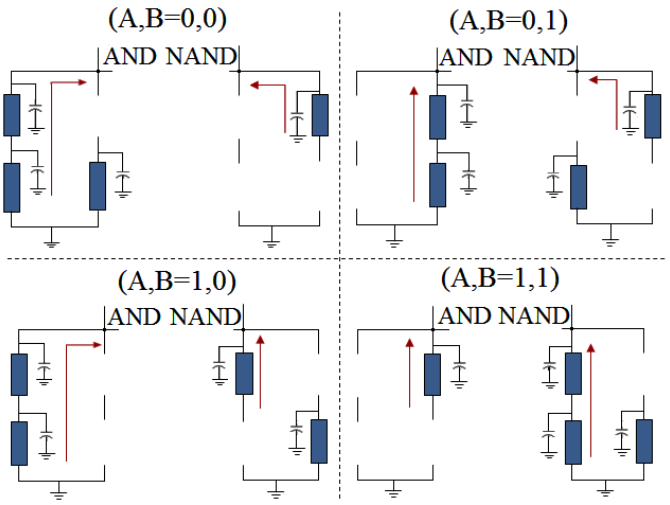
Like SQAL [102], SyAL [91] is also based on ECRL [31], [32] adiabatic logic and works on 4-phase power-clocking scheme. The difference between SQAL and SyAL adiabatic

logic lies in the number of charge sharing transistors used. SyAL uses more charge sharing transistors in comparison to SQAL. It also suffers from NAL during the evaluation and the recovery phase of the PC and from the coupling effect during the recovery phase.

Figure 6.3 (a) and (b) shows the AND/NAND gate using SyAL and its equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. It can be seen that SyAL uses five charge sharing transistor compared to three in the SQAL AND/NAND gate. From Figure 6.3 (b) it can be seen that, for each of the 4 input combinations, the capacitance at the two output nodes is different leading to data-dependent energy dissipation.



(a)



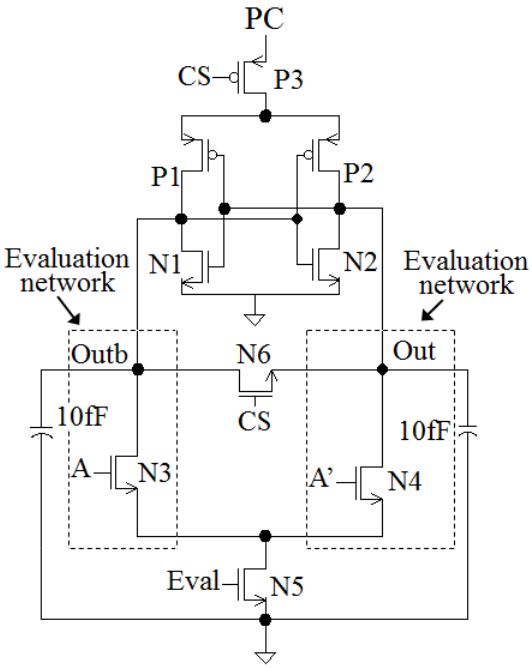
(b)

Figure 6.3: (a) AND/NAND gate using SyAL[91] (b) Equivalent RC models during evaluation phase.

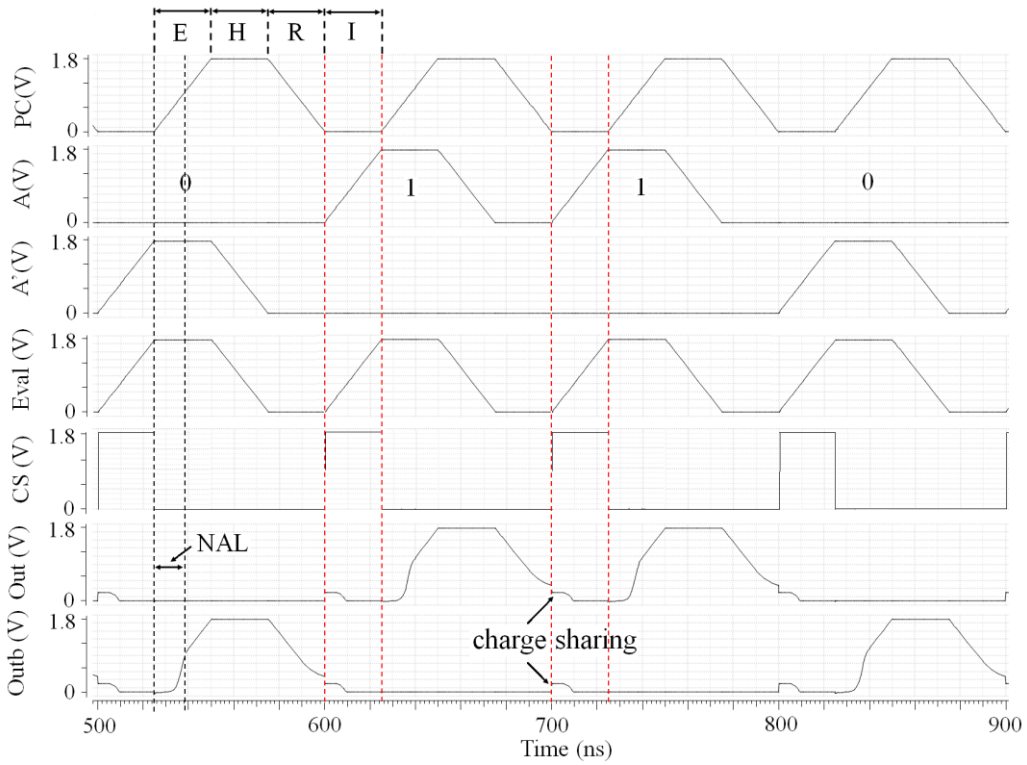
6.3.3 Charge Sharing Symmetric Adiabatic Logic (CSSAL)

CSSAL [87]-[90] is based on 2N-2N2P [17] and is an enhancement of SyAL [91] adiabatic logic. Figure 6.4 (a) and (b) shows the schematic of the CSSAL NOT/BUF gate and its simulation result at 10MHz respectively. The operation of CSSAL gate is explained through the design of a NOT/BUF gate. The simulation result shows the PC, input A, its complement A', evaluation input, Eval, charge sharing input, CS, and the output nodes 'Out' and 'Outb'.

The evaluation networks are connected between the two output nodes and the ground and are indicated in the schematic of the NOT/BUF gate of Figure 6.4 (a). Apart from charge sharing input, CSSAL also uses evaluation input. The evaluation input rises simultaneously with the input signals. N3 and N4 are the input transistors, N5 is the evaluation transistor, N6 and P3 are the charge sharing transistors and P1, P2, N1, and N2 forms the cross-coupled latch responsible to hold the two output nodes to their respective voltages. The operation is explained for A='1', A'= '0'.



(a)



(b)

Figure 6.4: (a) CSSAL NOT/BUF [87]-[90] gate (b) Simulation result at 10MHz.

During the Idle (I) phase of the PC, when the input A is rising, transistor N3 turns ON after the input reaches the threshold voltage. Also, the evaluation transistor N5 turns ON and drives the output node 'Outb' to ground. Also, the charge sharing transistor, N6, turns ON and connects the output node, 'Out' to ground through transistors N3 and N5. In this manner, both the output nodes are discharged to ground before the evaluation of the next inputs.

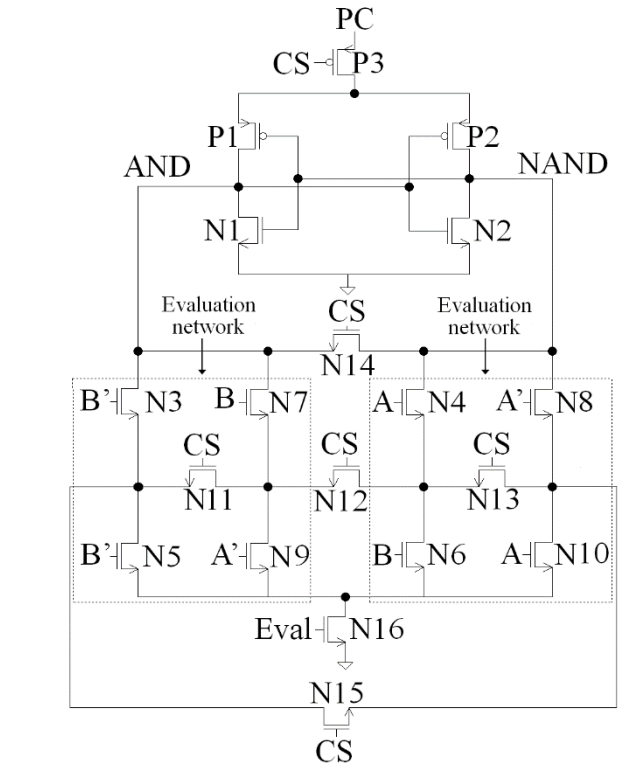
During the Evaluation (E) phase, the charge sharing transistor N6 is turned OFF. The input A and Eval is at logic '1'(complementary signal, A' is at logic '0') and the PC ramps up. The output node, 'Outb' will be connected to ground and the node, 'Out' will follow the PC through transistor P2 and the charge sharing transistor P3. The output node will follow the rising PC, only when the PC has reached the threshold voltage of the pMOS transistors P3 and P2, leading to NAL. It should be noted that due to the charge sharing transistor connected between the cross-coupled pMOS transistors and the PC, the evaluating output node in CSSAL experiences increased lag in following the PC and thus increased NAL and energy dissipation compared to SQAL and SyAL.

During the Hold (H) phase, the input A and Eval ramp down and the transistors, N3 and N5 are switched OFF when the gate-to-source voltage falls below the threshold voltage, $V_{tn.}$ The output nodes 'Out' and 'Outb' are held at their respective voltages due to the cross-coupled pMOS and nMOS transistors, P1, P2, N1, and N2.

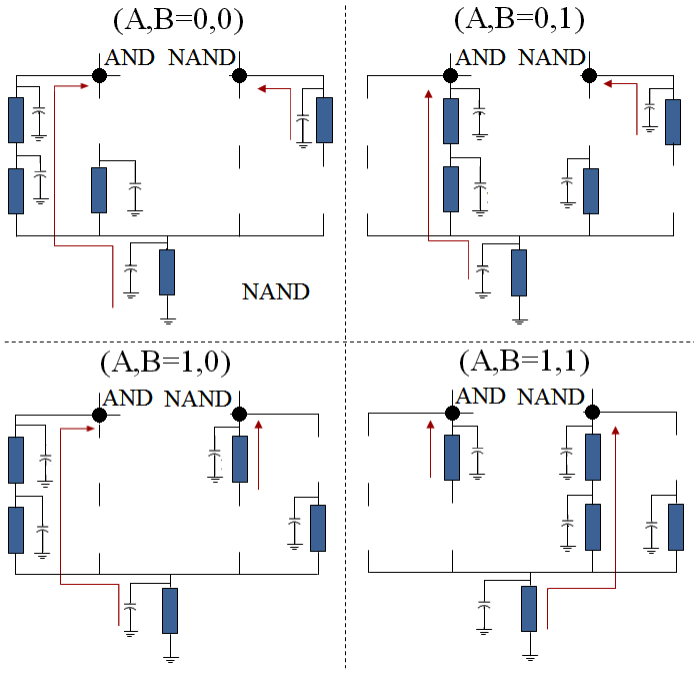
During the Recovery (R) phase, the input transistors are OFF and the PC ramps down. The charge on the output node 'Out' will be recovered back to the PC through transistors P3 and P2. The charge is recovered till PC falls below the threshold voltage, $|V_{tp}|$ of P3 and P2. When transistors P3 and P2 are turned off, the left-over charge remains at output node 'Out', leading to NAL. Also, the two output nodes remain floating and one of the nodes gets coupled to the evaluating node leading to the coupling effect. Unlike SQAL and SyAL, due to the presence of cross-coupled nMOS transistors (N1 and N2), the two output nodes remain floating for the part of recovery phase when the PC falls below the threshold voltage of the pMOS transistors P2 and P3. This leads to coupling effect only for the part of recovery phase.

The leftover charge from the recovery phase will be discharged to ground in the idle phase when the charge sharing transistor is turned ON. Figure 6.5 (a) and (b) shows the AND/NAND gate using CSSAL and its equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. It can be seen that, for each of the 4 input combinations, the capacitance at the two output nodes is different leading to the asymmetric structure.

CSSAL also works on 4-phase power-clocking scheme thus; 4-phases of the charge sharing and evaluation inputs need to be generated. This incurs the overhead of generation and routing of the two additional inputs.



(a)



(b)

Figure 6.5: (a) AND/NAND gate using CSSAL[87]-[90] (b) Equivalent RC models during evaluation phase.

6.4 Proposed Logic: Without Charge Sharing Quasi Adiabatic Logic (WCS-QuAL)

For data-independent power consumption, the two output nodes of the adiabatic gate should charge equal capacitance for each input transitions. This can be achieved by having a symmetric structure, where an equal number of transistors is turned ON on the two output nodes for each input transition. In proposed logic, Without Charge Sharing Quasi Adiabatic Logic (WCS-QuAL), this is achieved by using dual duplicate evaluation network (shown in Figure 6.6 (a), one connected between the power-clock and the two output nodes and the other connected between the two output nodes and the ground. This allows an equal number of transistors to be turned ON in the diagonally opposite evaluation networks on the two output nodes for each input transition. Having dual duplicate evaluation network helped to make the circuit symmetric and to get the data-independent energy-consumption.

The dual duplicate evaluation network in the WCS-QuAL also helps the two output nodes to discharge to zero (without using charge sharing input) before the evaluation of the next inputs.

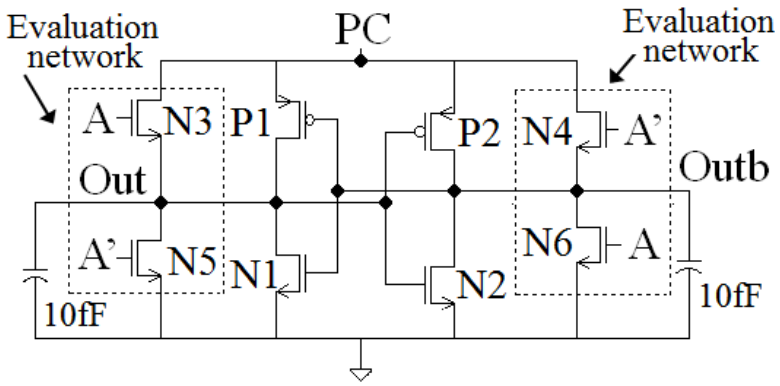
Figure 6.6 (a) and (b) shows a NOT/BUF gate using WCS-QuAL and its simulation results at 10MHz respectively. The operation of WCS-QuAL gate is explained through the design of a NOT/BUF gate. N3, N4, N5, and N6 are the input transistors and P1, P2, N1 and N2 forms the cross-coupled latch responsible for holding the output nodes to their respective voltages. The simulation result shows the PC, input A, its complement A’ and the output nodes ‘Out’ and ‘Outb’. WCS-QuAL works on 4-phase power-clocking scheme. The operation is explained for A= ‘1’, A’= ‘0’.

During the Idle phase (I) when input A is rising transistors N3 and N6 are turned ON after the input reaches the threshold voltage. The PC is zero during the idle phase, therefore, the output node; ‘Out’ is connected to PC through transistor N3 and is zero. Similarly, transistor N6 causes the output node, ‘Outb’ to connect to ground. This way, two output nodes are discharged to zero before the evaluation phase of the PC begins. Therefore, no charge sharing transistors are required and the overhead of generation and routing of the 4 phases of the charge sharing input is saved.

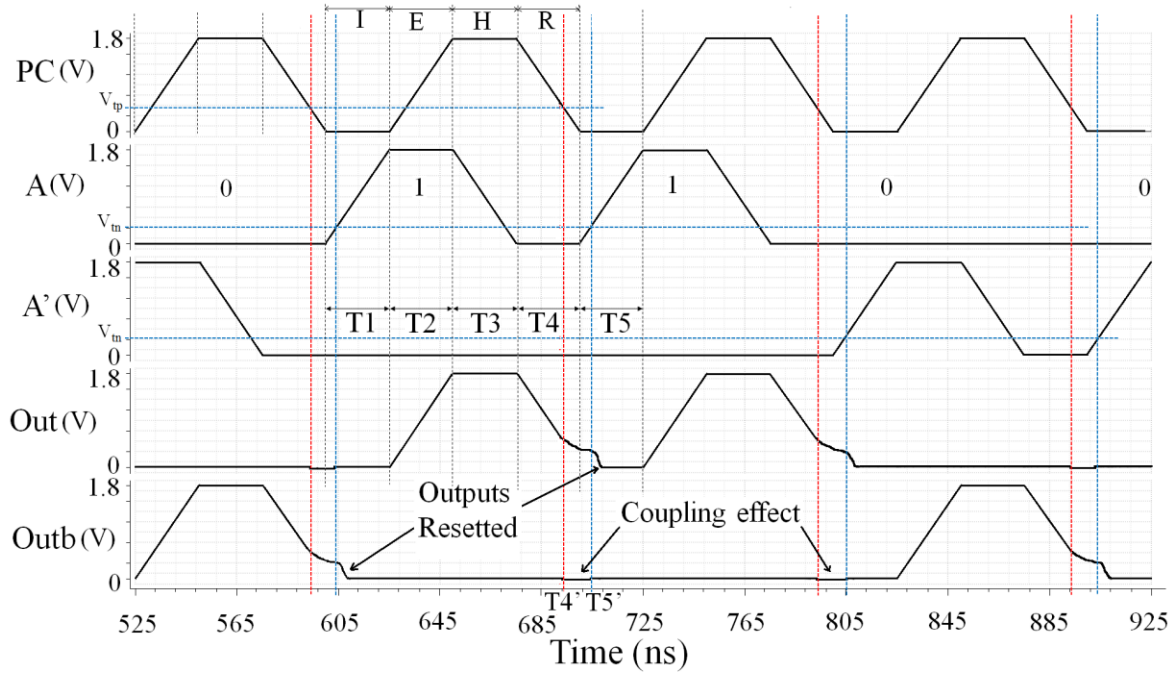
During the Evaluation phase (E), input A is logic ‘1’ (A’ is logic ‘0’) and the PC rises up. The output node, ‘Out’ follows the PC through N3 and P1 from 0 to V_{DD}-V_{tn} and V_{tp} to V_{DD} respectively and thus, does not suffer from NAL.

During the Hold phase (H), input A ramps down and the transistor N3 and N6 are switched OFF when the gate-to-source voltage falls below $V_{tn.}$ The output nodes, Out and Outb are held at their respective voltages due to the cross-coupled transistors (P1, P2, N1, and N2).

During the Recovery phase (R), the PC ramps down and the charge on the output node Out is recovered back to the PC through the transistor, P1. The charge is recovered till PC falls below the threshold voltage, $|V_{tp}|$ of P1. At the time, T4’, P1 is turned off and the node ‘Out’ stays at V_{tp}. The left-over charge will be discharged to ground in the idle phase at time T5’ when the next input arrives, and its gate voltage exceeds the threshold voltage (V_{tn}). From T4’ to T5’, the output nodes are floating, thus the complementary node ‘Outb’ gets coupled to node Out and goes below zero, leading to the capacitive coupling effect.



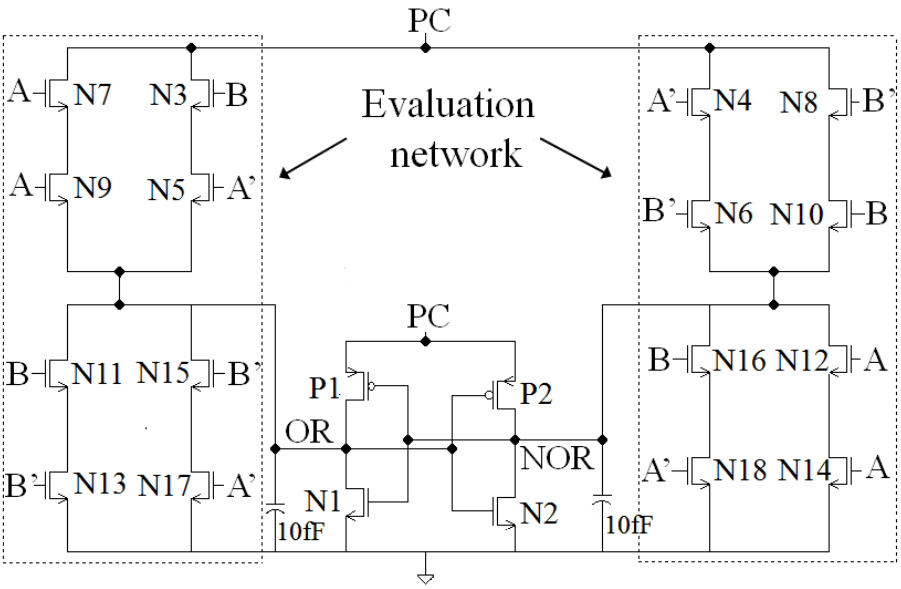
(a)

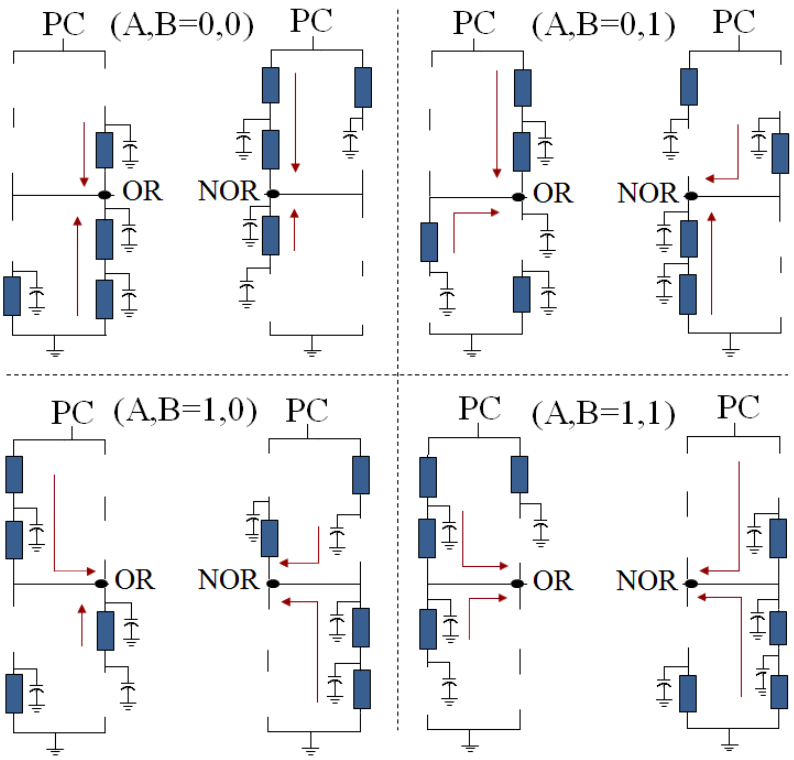


(b)

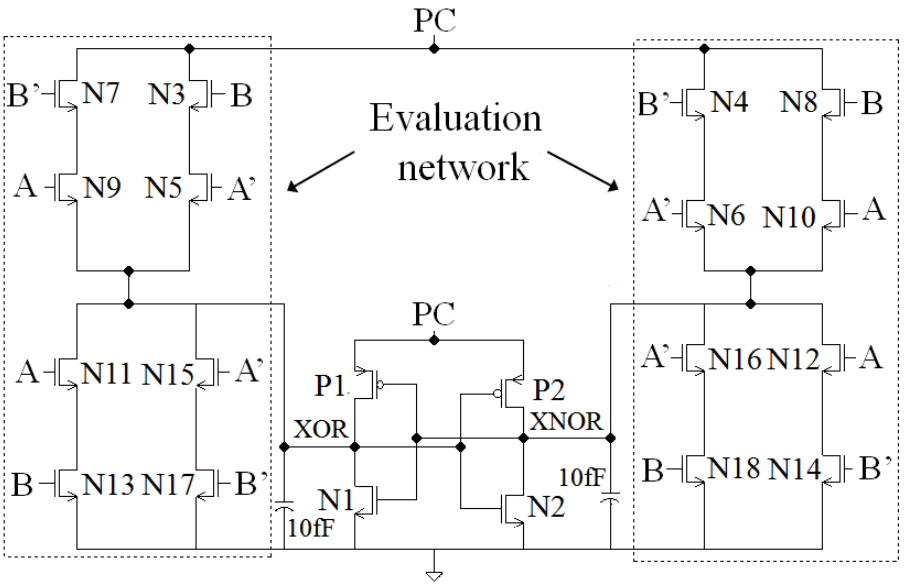
Figure 6.6: (a) WCS-QuAL NOT/BUF gate (b) Timing diagram at 10MHz.

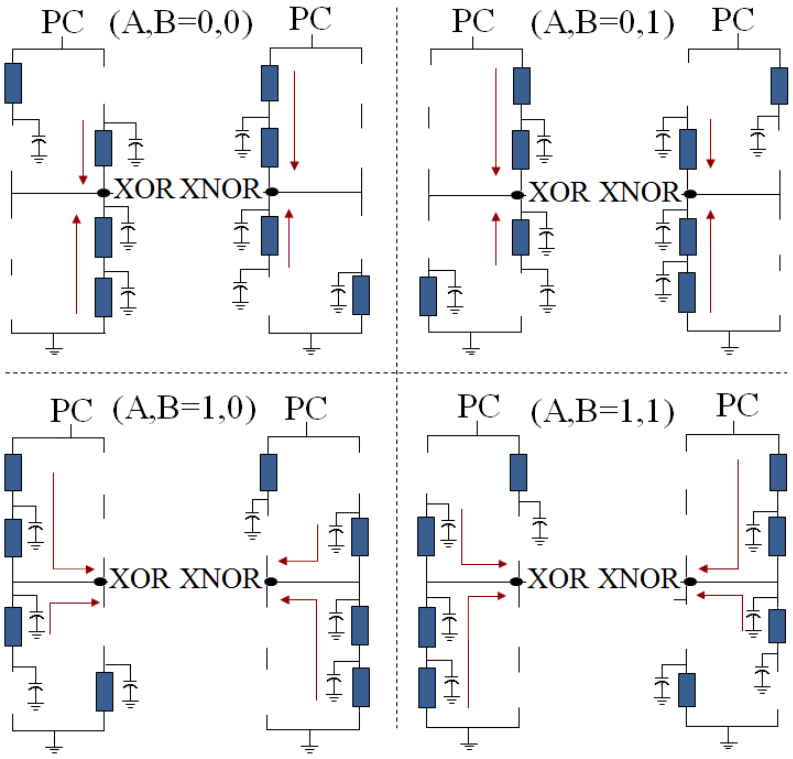
Figure 6.7 (a), (b) (c) and (d) shows the schematics of OR/NOR, XOR/XNOR and AND/NAND, gates using WCS-QuAL and their equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. It can be seen that, in each gate, for each of the 4 input combinations, the two output nodes charge the same capacitance leading to data-independent behaviour, unlike SyAL, SQAL, and CSSAL. All the 2-input logic gates using WCS-QuAL have the same structure and an equal number of transistors, except the position of the input signals.


PC
A
N7
N3
B
Evaluation
network
A'
N4
N8
B'
A
N9
N5
A'
B'
N6
N10
B
PC
B
N11
N15
B'
P1
P2
B
N16
N12
A
OR
NOR
B'
N13
N17
A'
N1
N2
A'
N18
N14
A
10fF
10fF

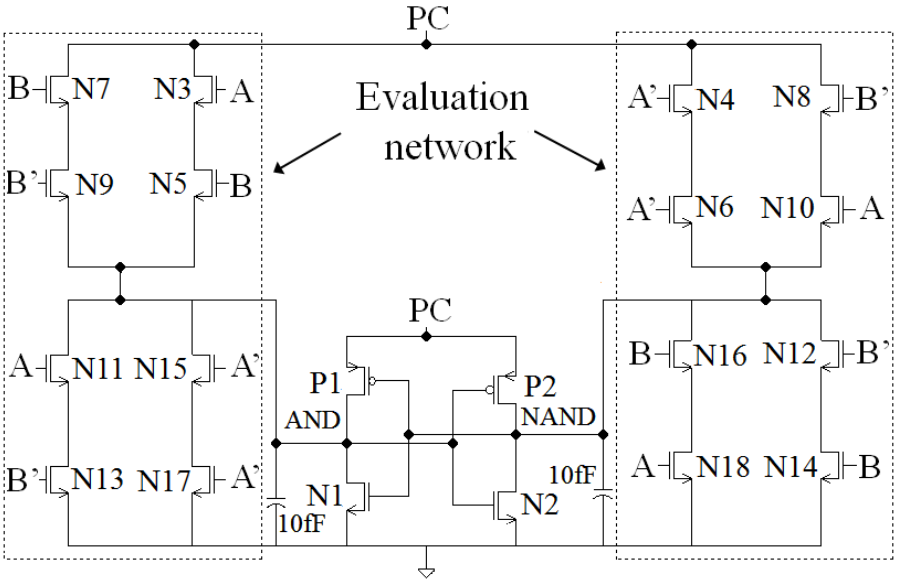

PC
(A,B=0,0)
PC
OR
NOR
PC
(A,B=0,1)
PC
OR
NOR
PC
(A,B=1,0)
PC
OR
NOR
PC
(A,B=1,1)
PC
OR
NOR

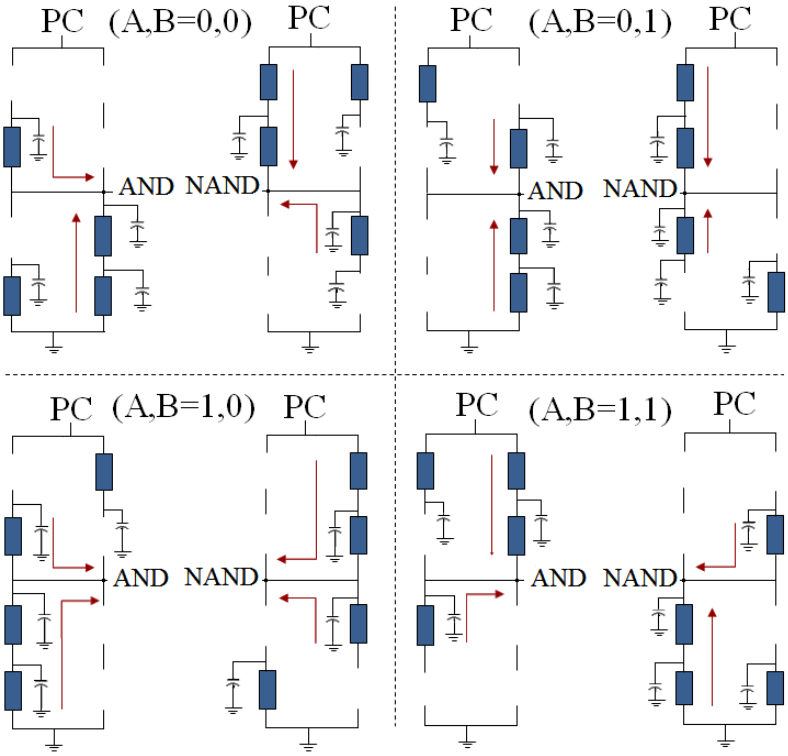
(a)


PC
B'
N7
N3
B
A
N9
N5
A'
Evaluation
network
B'
N4
N8
B
A'
N6
N10
A
PC
A
N11
N15
A'
P1
P2
XOR
XNOR
A'
N16
N12
A
B
N13
N17
B'
10fF
N1
N2
10fF
B
N18
N14
B'


PC
(A,B=0,0)
PC
XOR
XNOR
PC
(A,B=0,1)
PC
XOR
XNOR
PC
(A,B=1,0)
PC
XOR
XNOR
PC
(A,B=1,1)
PC
XOR
XNOR

(b)





(c)

Figure 6.7: WCS-QuAL 2-input gates and their equivalent RC model for evaluation phase (a) OR/NOR (b) XOR/XNOR. (c) AND/NAND

6.5 Simulation Results

Simulations for all the secure adiabatic logic designs were performed with Spectre simulator using Cadence EDA tool in a 'typical-typical' (TT) process corner using TSMC 180nm CMOS process at 1.8V power supply. The load capacitance was chosen as 10fF and the transistor sizes for all the designs were set to the technology minimum ($W_{min}=W_n=W_p=220nm$, $L_{min}=L_n=L_p=180nm$).

6.5.1 Impact of Frequency on NED and NSD

The simulations were performed at 1MHz, 10MHz and 100MHz frequencies. The energy dissipation was measured per cycle for 4 and 16-input transitions for NOT/BUF and 2-input gates using SyAL, SQAL, CSSAL, and WCS-QuAL.

To evaluate the resistance of WCS-QuAL and the existing secure adiabatic logic designs against PAA, the maximum energy (E_{max}), minimum energy (E_{min}), the average energy (E_{av}), and the standard deviation (σ) for 4 and 16-input transitions of single and 2-input gates were measured. From these, Normalised Energy Deviation (NED) and Normalised Standard Deviation (NSD) were obtained according to (6.1) and (6.2). Greater the difference between the maximum and minimum energy values higher the %NED and %NSD and higher the cell's susceptibility to PAA. NED and NSD are the two widely used security metrics to evaluate the resistance of the secure logic designs against PAA. The metrics were introduced by Tiri in [76] and was used to evaluate the resistance of SABL logic. NED gives the measure of variation between the maximum and the minimum energy dissipation of the gate (for all the possible input transitions) with respect to the maximum energy dissipated by the gate. NSD on the other hand, gives the deviation of energy dissipation of the gate for all the possible input transitions with respect to the average energy dissipated by the gate. Since the proposed logic is based on the principle that if the energy dissipation of the logic gate can be made independent of the inputs, the gate can be made resistant to PAA. In other words, if the gate dissipates equal energy for all the possible input transitions, the energy dissipation of the gate can be referred as input independent. Because the two metrics give the measure of variations in the energy dissipated per input transition, they are appropriate to be used as the security metrics to measure the resistance of the proposed logic against PAA. In addition, the existing secure adiabatic logic designs are also based on the fact that if the energy dissipation of the secure

adiabatic logic gate can be made input independent, the logic gate can be made resilient to PAA.

The NED [76] is defined as:

$$NED = \left(E_{\max} - E_{\min}\right) / E_{\max} \tag{6.1}$$

NSD [76] is defined as:

$$NSD = \sigma / E_{av} \tag{6.2}$$

Standard Deviation is defined as:

$$\sigma = \sqrt{\sum_{i=1}^{n} \left(E_i - E_{av}\right)^2} \Big/ n \tag{6.3}$$

The pre-layout simulation results of the evaluated gates are summarized in Table 6.1. It shows that WCS-QuAL exhibits the best (i.e. least) value of %NED and %NSD at all simulated frequencies than the existing logic designs. It also shows that at 1MHz, 2-input gates using WCS-QuAL dissipate more energy than the 2-input gates using SQAL and SyAL and slightly less than CSSAL. However, former (WCS-QuAL) suffers from NAL only during the recovery phase, whereas, all the three-existing logic designs suffer from NAL in both the evaluation and recovery phase of the power-clock. Also, CSSAL has 2 stacked transistors in the evaluation and recovery path (as shown in Figure 6.4 (c)) thus, has higher NAL than SQAL and SyAL.

Since, the energy dissipated by the adiabatic logic in general, is dominated by leakage energy and not by Adiabatic Losses (AL) and NAL at low frequencies [37], [100] WCS-QuAL, having more transistors than SQAL and SyAL dissipate more energy. On the other hand, CSSAL has approximately equal transistors compared to WCS-QuAL but has higher NAL thus, consumes more energy.

The NOT/BUF gate using WCS-QuAL consumes the lowest energy at all the simulated frequencies. Because the existing logic also suffers from NAL in the evaluation phase they dissipate more energy than WCS-QuAL.

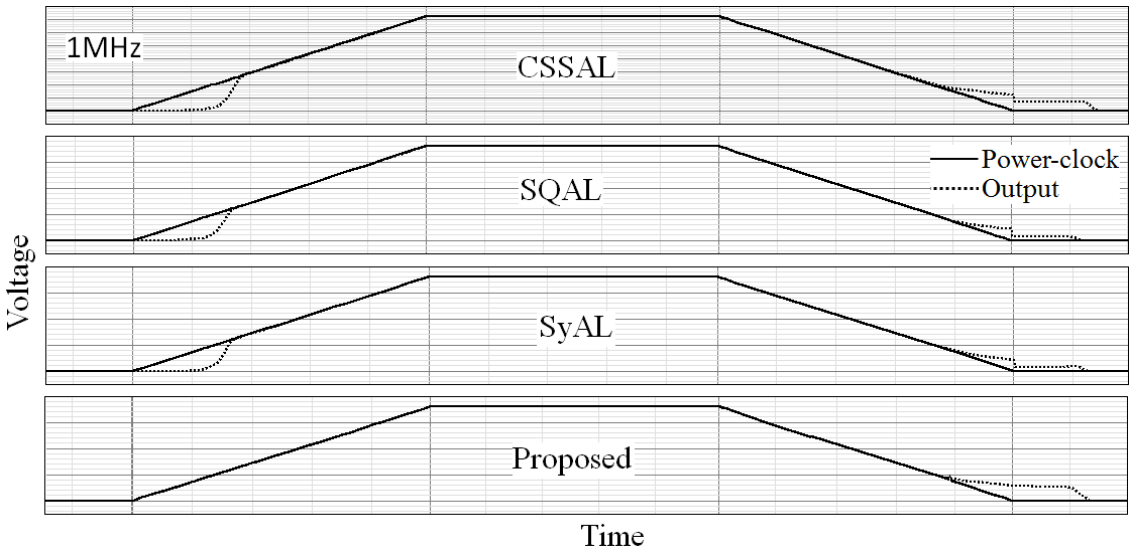
Table 6.1: Pre-layout simulation results of the gates using WCS-QuAL and the existing logic

Logic	1 MHz				10MHz				100MHz			
Designs	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
Eav (fJ)	3.314	2.415	2.415	1.792	6.340	4.276	4.276	2.479	19.540	12.180	12.180	5.685
%NED	0.781	2.050	2.050	0.445	1.223	1.163	1.163	0.523	0.814	0.735	0.735	0.351
%NSD	0.453	0.920	0.920	0.257	0.710	0.675	0.675	0.281	0.377	0.358	0.358	0.176
AND/NAND												
Eav (fJ)	6.500	4.892	5.434	5.837	10.350	7.137	7.760	6.438	28.710	17.870	19.253	10.674
%NED	1.115	2.384	1.933	0.562	1.914	2.985	1.332	0.186	1.073	3.685	1.546	0.187
%NSD	0.458	1.169	0.810	0.167	0.599	1.033	0.409	0.047	0.456	1.505	0.619	0.076
OR/NOR												
Eav (fJ)	6.499	4.890	5.435	5.838	10.360	7.129	7.765	6.439	28.710	17.840	19.233	10.674
%NED	1.161	2.384	1.988	0.528	1.820	3.065	0.922	0.124	1.010	3.630	1.597	0.187
%NSD	0.483	1.169	0.813	0.165	0.596	1.109	0.330	0.034	0.442	1.444	0.610	0.076
XOR/XNOR												
Eav (fJ)	6.503	5.152	5.444	5.840	10.370	7.368	7.761	6.440	28.720	17.090	19.235	10.676
%NED	0.964	0.658	1.808	0.545	1.726	0.095	1.589	0.047	1.040	0.992	1.444	0.187
%NSD	0.477	0.179	0.573	0.183	0.474	0.032	0.385	0.019	0.428	0.318	0.592	0.068

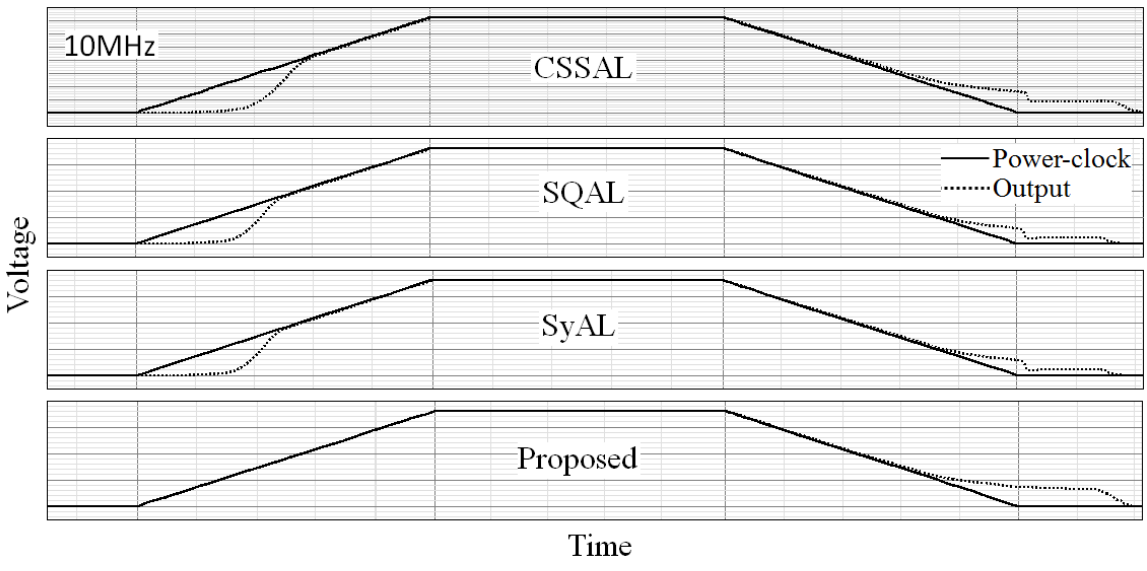
At higher frequencies (shorter ramping time), the effect of Adiabatic Loss (AL) is more prominent rather than leakage loss [37], [100]. As the frequency is increased, AL combined

with NAL leads to more energy dissipation in existing logic designs than WCS-QuAL as can be seen from Table 6.1.

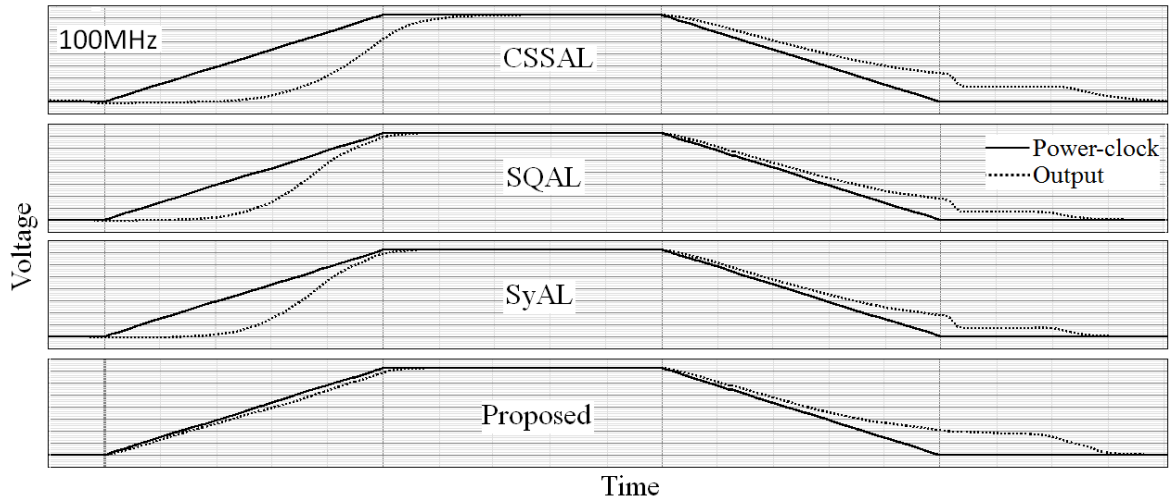
Figure 6.8 (a), (b) and (c) shows the output waveforms (dotted line) of AND/NAND gates using CSSAL, SQAL, SyAL, and WCS-QuAL overlapped on the power-clock (solid line) at 1MHz, 10MHz, and 100MHz respectively. For CSSAL, SQAL, and SyAL, NAL exists for both the evaluation and the recovery phase of the power-clock, however, for WCS-QuAL, NAL exists only for the recovery phase of the power-clock. As the frequency increases, AL combined with NAL leads to large energy dissipation in CSSAL, SQAL and SyAL compared to WCS-QuAL.



(a)



(b)



(c)

Figure 6.8: Output waveforms of 2-input AND/NAND gate using CSSAL, SQAL, SyAL and WCS-QuAL overlapped on the Power-clock at (a) 1MHz, (b) 10MHz and (c)100MHz.

Table 6.1 also shows that the performance (on the basis of %NED and %NSD) of existing logic designs changes with frequency. CSSAL is second best followed by SyAL and SQAL At 1MHz, whereas, at 10MHz, SyAL is second best followed by CSSAL and SQAL. The order of performance at 100MHz is same as at 1 MHz.

6.5.2 Intra-Operation Energy Variability

Another advantage of WCS-QuAL is that all its 2-input gates dissipate nearly the same energy at all simulated frequencies. Since PAA is based on the principle that where energy consumption is data-dependent, sensitive data can be inferred from analysis of the power supply currents. The main benefit of the logic reported, is that any particular gate's energy consumption is data-independent.

However, an additional level of protection is offered by ensuring, as far as possible, that an AND gate, say, uses the same energy as an OR gate; thereby making it difficult to infer what logic operation is being performed at any one time. In other words, "gate-independence" as well as data-independence is achieved. Table 6.2 shows the average energy dissipated for all possible input transitions of AND/NAND, OR/NOR and XOR/XNOR gates using WCS-QuAL and existing logic.

Table 6.2: Comparison of the standard deviation of average energy dissipated by 2-input gates

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
AND/NAND												
Eav (fJ)	6.500	4.892	5.434	5.837	10.350	7.137	7.760	6.438	28.710	17.870	19.235	10.674
OR/NOR												
Eav (fJ)	6.499	4.890	5.435	5.838	10.360	7.129	7.765	6.439	28.710	17.840	19.233	10.674
XOR/XNOR												
Eav (fJ)	6.503	5.152	5.444	5.840	10.370	7.368	7.761	6.440	28.720	17.090	19.235	10.676
Eav,gates(fJ)	6.500	4.978	5.437	5.838	10.360	7.211	7.762	6.439	28.713	17.600	19.234	10.674
σ (fJ)	0.002	0.150	0.005	0.001	0.010	0.135	0.002	0.001	0.005	0.441	0.001	0.001

It also shows the standard deviation (σ) of average energy dissipated by the three logic gates at all the simulated frequencies. WCS-QuAL shows the lowest value of standard deviation compared to existing logic designs at all frequencies simulated.

6.5.3 Impact of Process Corner Variations

To further evaluate the robustness of WCS-QuAL, CSSAL, SQAL, and SyAL under process corner variations. The simulations were performed at 'Fast-Fast' (FF), 'Slow-Slow' (SS), 'Fast-Slow' (FS) and 'Slow-Fast' (SF) process corners. The pre-layout simulation results for FF, SS, FS and SF process corners are summarised in Tables 6.3, 6.4, 6.5 and 6.6 respectively.

The simulation results show that WCS-QuAL exhibits the least (i.e. best) value of %NED and %NSD for each process corner at frequencies simulated. However, the ranking of performance (level of security based on %NED and %NSD) of CSSAL, SQAL, and SyAL is not independent of process corner variations.

At FF process corner and 1) at 1MHz, WCS-QuAL exhibits the least value of %NED and %NSD followed by SyAL, SQAL, and CSSAL. 2) At 10MHz, WCS-QuAL exhibits the least value of %NED and %NSD followed by CSSAL, SyAL, and SQAL. 3) At 100MHz, same order as mentioned in point 2 is followed.

At SS process corner, and 1) at 1MHz, WCS-QuAL exhibits the least value of %NED and %NSD followed by SyAL, CSSAL, and SQAL. 2) At 10MHz, WCS-QuAL exhibits the least value of %NED and %NSD followed by CSSAL, SyAL, and SQAL. 3) At 100MHz, same order as mentioned in point 2 is repeated.

At FS corner and at 1MHz, 10MHz and 100MHz, WCS-QuAL shows the least value of the %NED and %NSD followed by CSSAL, SyAL, and SQAL.

Lastly at SF corner, and 1) at 1MHz, WCS-QuAL has the least value of %NED and %NSD followed by SyAL, SQAL, and CSSAL logic. 2) At 10MHz, the order changes to WCS-QuAL followed by SyAL, CSSAL, and SQAL. 3) At 100MHz, the sequence is changed to WCS-QuAL followed by CSSAL, SyAL, and SQAL.

Table 6.3: Pre-layout simulation results of gates at FF corner.

Logic	1 MHz				10MHz				100MHz			
Designs	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	2.858	0.808	0.808	0.548	0.165	0.406	0.406	0.136	0.182	0.174	0.174	0.083
%NSD	1.439	0.362	0.362	0.245	0.082	0.150	0.150	0.049	0.091	0.078	0.078	0.035
AND/NAND												
%NED	1.693	1.077	0.815	0.521	1.241	3.115	1.548	0.329	1.571	4.058	2.781	0.524
%NSD	0.555	0.514	0.199	6.901	0.484	1.188	0.487	0.077	0.583	1.890	0.877	0.159
OR/NOR												
%NED	1.369	1.348	0.830	0.517	1.174	3.066	1.305	0.392	1.571	3.792	1.365	0.565
%NSD	0.428	0.522	0.190	0.177	0.427	1.232	0.545	0.093	0.575	1.829	0.614	0.156
XOR/XNOR												
%NED	0.579	1.070	2.212	0.488	2.014	1.548	1.228	0.314	1.610	0.979	1.731	0.442
%NSD	0.299	0.270	0.567	0.125	0.525	0.590	0.401	0.115	0.559	0.284	0.634	0.144

Table 6.4: Pre-layout simulation results of gates at SS corner.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	0.554	2.045	2.045	0.050	0.111	0.105	0.105	0.035	0.270	0.337	0.337	0.073
%NSD	0.271	0.857	0.857	0.022	0.055	0.047	0.047	0.019	0.118	0.177	0.177	0.030
AND/NAND												
%NED	1.812	3.422	1.726	0.254	1.184	2.955	2.145	0.030	0.677	2.258	2.063	0.083
%NSD	0.641	1.215	0.534	0.074	0.350	0.924	0.609	0.007	0.307	1.043	0.505	0.028
OR/NOR												
%NED	1.812	2.259	1.507	0.372	0.948	2.868	2.062	0.045	0.677	2.306	1.492	0.167
%NSD	0.641	0.957	0.488	0.109	0.339	1.030	0.673	0.001	0.307	1.029	0.471	0.052
XOR/XNOR												
%NED	1.681	2.077	1.197	0.679	1.104	4.442	2.021	0.045	0.967	1.298	1.580	0.084
%NSD	0.810	0.479	0.444	0.163	0.359	1.119	0.591	0.011	0.333	0.312	0.596	0.043

Table 6.5: Pre-layout simulation results of gates at FS corner.

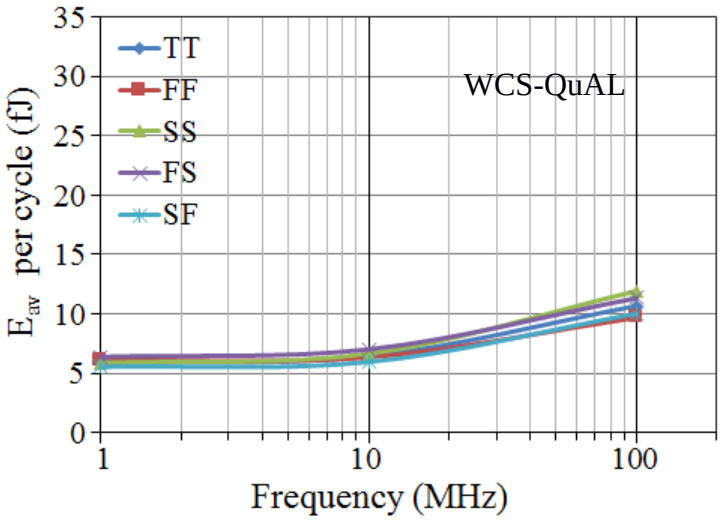
Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	0.070	0.617	0.617	0.043	0.529	0.602	0.602	0.033	0.047	0.217	0.217	0.032
%NSD	0.029	0.262	0.262	0.024	0.026	0.291	0.015	0.015	0.023	0.125	0.125	0.011
AND/NAND												
%NED	0.700	2.222	0.860	0.546	1.334	2.283	2.136	0.156	1.140	3.882	2.457	0.352
%NSD	0.347	0.921	0.347	0.210	0.427	0.704	0.527	0.039	0.561	1.625	0.736	0.068
OR/NOR												
%NED	0.700	1.954	0.736	0.432	1.334	2.164	1.063	0.213	1.140	3.882	2.030	0.264
%NSD	0.347	0.899	0.304	0.101	0.490	0.623	0.300	0.061	0.552	1.625	0.635	0.066
XOR/XNOR												
%NED	0.996	1.326	0.813	0.670	1.252	0.513	2.535	0.199	1.172	1.304	2.319	0.176
%NSD	0.311	0.484	0.332	0.152	0.363	0.126	0.742	0.062	0.537	0.424	0.767	0.050

Table 6.6: Pre-layout simulation results of gates at SF corner.

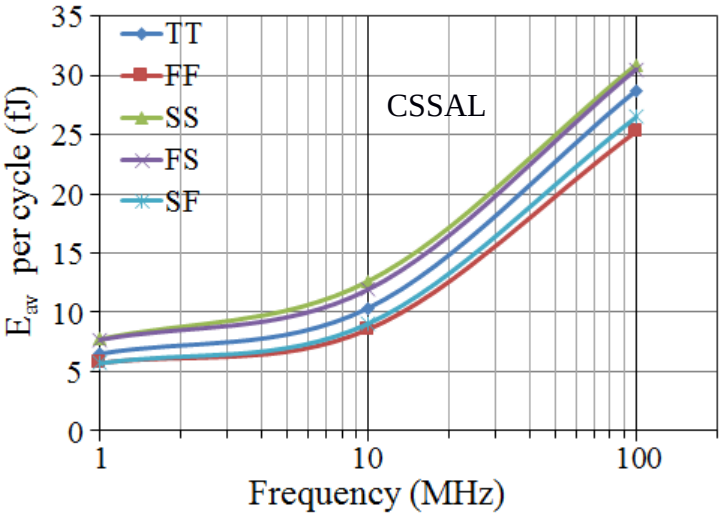
Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	0.763	2.209	2.209	0.465	0.495	1.215	1.215	0.378	0.056	1.685	1.685	0.019
%NSD	0.383	1.021	1.021	0.208	0.248	0.488	0.488	0.169	0.032	0.822	0.822	0.010
AND/NAND												
%NED	3.848	3.293	0.888	0.575	1.792	2.122	1.155	0.834	1.163	2.789	1.772	0.319
%NSD	1.327	0.848	0.408	0.121	0.591	0.944	0.379	0.185	0.424	1.246	0.744	0.081
OR/NOR												
%NED	3.848	2.999	0.867	0.593	1.728	2.690	1.497	0.251	1.200	2.541	1.435	0.329
%NSD	1.327	0.837	0.407	0.156	0.582	1.004	0.446	0.073	0.417	1.203	0.558	0.127
XOR/XNOR												
%NED	3.731	1.268	0.762	0.686	1.71	1.564	1.403	0.251	1.163	0.848	1.659	0.338
%NSD	1.074	0.445	0.344	0.252	0.633	0.548	0.490	0.083	0.448	0.271	0.509	0.112

Figure 6.9 (a), (b), (c) and (d) shows the average energy dissipation per cycle of the AND/NAND gate at all five process corners at 1MHz, 10MHz and 100MHz for WCS-QuAL, CSSAL, SQAL, and SyAL respectively. For WCS-QuAL, the average energy

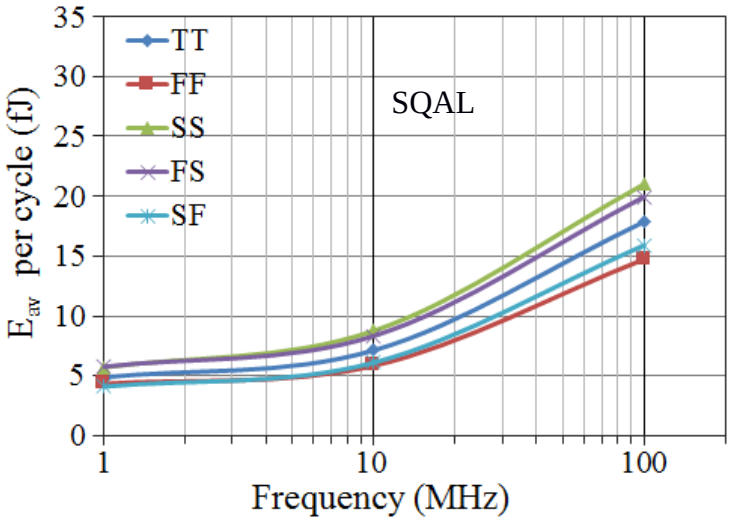
dissipation is largely independent of the process corners. The changes in threshold voltage have an impact on the NAL in adiabatic circuits and thus on energy dissipation. As WCS-QuAL completely removes the NAL from the evaluation phase, it shows the least sensitivity to process corners. By contrast, CSSAL, SQAL, and SyAL show greater sensitivity to process corners, especially at a higher frequency. Also, the average energy dissipation increases significantly for CSSAL, SQAL, and SyAL as moving from 1MHz to 100MHz.



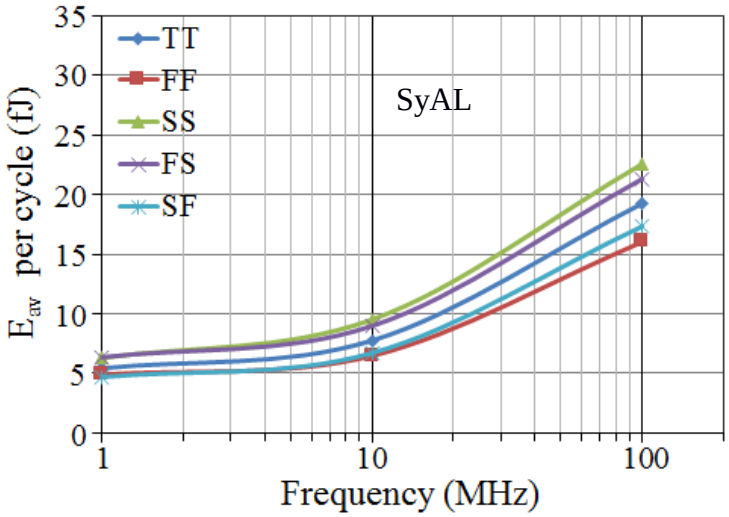
(a)



(b)



(c)



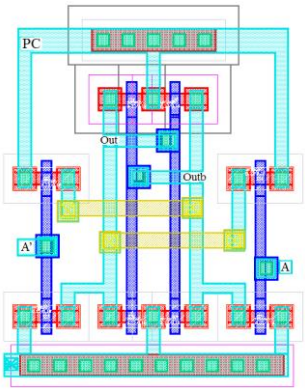
(d)

Figure 6.9: Average energy dissipation per cycle under TT, FF, SS, FS and SF process corners of AND/NAND gate at 1MHz, 10MHz and 100MHz for (a) WCS-QuAL (b) CSSAL (c) SQAL and (d) SyAL.

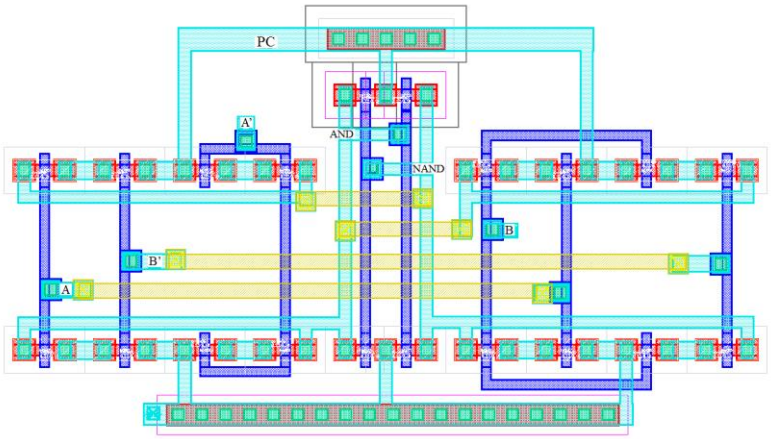
6.5.4 Post-Layout Results

In order to get more realistic simulation results, full-custom layouts were drawn using Cadence Virtuoso™ layout editor. The post-layout simulations were carried out using the av-extracted file from the layout design with the resistance and capacitance (RC) parasitic parameters. The layouts for each of the logic gates using existing logic designs and WCS-QuAL were drawn and the simulations were performed for all five process corners. Figure 6.10 (a), (b), (c) and (d) shows the layout designs for NOT/BUF, AND/NAND, OR/NOR

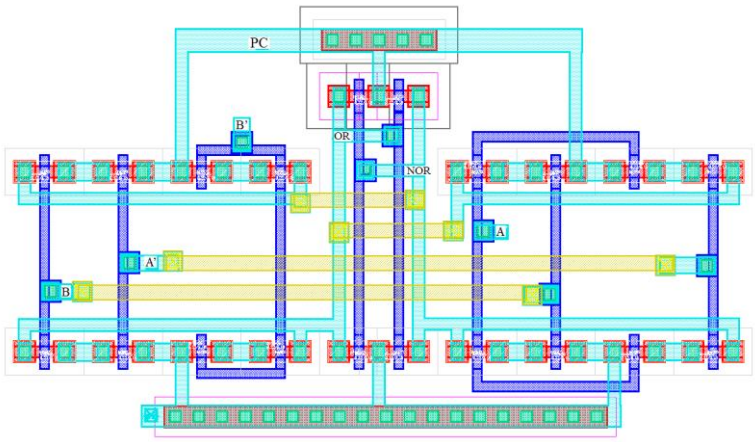
and XOR/XNOR gates respectively using WCS-QuAL. The layouts for all the gates were drawn bearing in mind the need to maintain the symmetry and load balancing on the output nodes [107], [109]. The area is often sacrificed while maintaining the symmetry of the layouts.



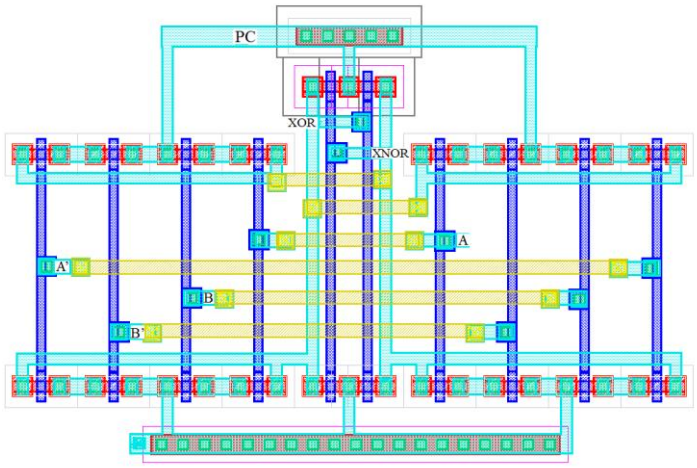
(a)



(b)



(c)



(d)

Figure 6.10: Layout designs of WCS-QuAL (a) NOT/BUF (b) AND/NAND (c) OR/NOR (d) XOR/XNOR gates.

The numerical values of the layout area of the logic gates using WCS-QuAL and the existing logic designs are shown in Table 6.7. Due to the less number of transistors, SQAL exhibits the lowest layout area for all the logic gates. Except NOT/BUF gate using WCS-QuAL, all 2-input gates consumes less area in comparison to the area consumed by 2-input gates using CSSAL and SyAL. CSSAL exhibits the highest layout area for all the logic gates.

Table 6.7: Layout area comparison of logic gates using WCS-QuAL and the existing logic designs.

Adiabatic Logic Gates	**Layout Area (μm)^2**			
	CSSAL	**SQAL**	**SyAL**	**WCS-QuAL**
NOT/BUF	6.68 x 7.97	6.28 x 5.42	6.28 x 5.42	6.44. x 7.13
AND/NAND	15.39 x 11.30	12.70 x 8.80	11.43 x 11.88	15.02 x 8.41
OR/NOR	15.39 x 11.30	12.70 x 8.80	11.43 x 11.88	15.02 x 8.41
XOR/XNOR	14.40 x 10.51	8.70 x 6.52	11.86 x 11.88	15.02 x 9.95

Table 6.8: Post-layout simulation results of gates using WCS-QuAL and existing logic designs.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
Eav (fJ)	4.191	3.085	3.085	2.093	7.803	5.097	5.097	3.101	23.910	14.630	14.630	6.945
%NED	8.804	7.873	7.873	2.505	6.760	6.775	6.775	2.080	5.690	6.775	6.775	1.202
%NSD	5.316	4.376	4.376	1.310	3.990	3.367	3.367	1.017	3.310	3.367	3.367	0.439
AND/NAND												
Eav (fJ)	8.900	5.980	7.331	6.670	13.280	8.677	10.279	7.403	36.380	22.530	25.848	12.810
%NED	13.18	14.97	14.02	7.483	13.910	14.37	13.760	8.099	12.450	13.840	13.230	7.106
%NSD	5.120	5.813	6.487	2.281	6.060	6.507	6.273	3.416	5.660	6.226	5.422	2.851
OR/NOR												
Eav (fJ)	8.818	6.093	6.752	6.043	13.260	8.645	9.444	6.998	36.410	22.498	24.461	12.720
%NED	13.64	14.66	14.09	7.888	13.740	14.45	12.956	7.656	12.740	13.84	13.185	6.555
%NSD	5.400	6.320	6.822	3.718	6.170	6.019	6.645	3.352	5.860	6.299	5.715	2.738
XOR/XNOR												
Eav (fJ)	9.302	6.369	6.720	5.928	14.670	8.747	9.762	7.045	40.570	20.100	25.254	12.890
%NED	10.65	9.360	10.78	6.431	11.860	4.023	11.472	7.461	9.515	8.469	9.851	4.397
%NSD	4.130	2.741	4.970	2.405	5.200	1.441	3.770	2.974	4.010	2.779	4.116	1.606

The post-layout simulation results are summarized in Table 6.8. The results show that there is a large difference in %NED and %NSD for WCS-QuAL and the existing secure adiabatic logic designs in comparison to their corresponding pre-layout simulation results. This large difference is due to the complexity and difficulty of making layouts symmetric. Routing of charge sharing transistors and evaluation transistors make the balancing of the two output nodes difficult which leads to higher values of %NED and %NSD in CSSAL, SQAL, and SyAL. WCS-QuAL exhibits the lowest value of %NED and %NSD compared to CSSAL, SQAL, and SyAL at all the frequencies simulated.

The post-layout simulation results for FF, SS, FS and SF process corners are summarised in Table 6.9, 6.10, 6.11 and 6.12 respectively. The post-layout simulation results show that WCS-QuAL exhibits the least value of %NED and %NSD followed by SQAL, SyAL and CSSAL under FF, SS, FS and SF process corners at 1MHz, 10MHz, and 100MHz. %NED and %NSD for SQAL are better in comparison to SyAL and CSSAL because it has less complex circuit compared to SyAL and CSSAL. 2-input gates using SQAL (except XOR/XNOR gate which uses single charge sharing transistor) uses three charge sharing transistors compared to five charge sharing transistors in SyAL and six charge sharing transistors and a single evaluation transistor in CSSAL. The complexity of the circuit makes it difficult to draw layouts symmetrical and balance the capacitive loads at the two output nodes. For ensuring the constant power consumption for every input transition, it is required that the capacitive loads at the two output nodes are balanced. Interconnect wires, resistances, and capacitances should be balanced on the two output nodes. Routing of charge sharing transistors and evaluation transistors makes this difficult which leads to higher values of %NED and %NSD. The values of %NED and %NSD can further be improved for WCS-QuAL, CSSAL, SQAL and SyAL by improving the layout structures by experimenting with different structures.

Table 6.9: Post-layout simulation results of gates at FF corner.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	5.767	4.968	4.968	2.321	6.298	4.638	4.638	2.631	6.175	3.916	3.916	1.088
%NSD	2.691	2.610	2.610	1.097	0.227	2.203	2.203	0.985	3.555	2.189	2.189	0.475
AND/NAND												
%NED	15.05	12.883	15.70	8.243	16.50	15.13	15.72	8.386	16.399	15.566	15.32	8.883
%NSD	4.472	6.197	7.029	3.259	6.596	6.862	7.006	3.452	7.497	7.147	5.956	3.712
OR/NOR												
%NED	14.64	14.901	16.95	8.877	16.59	13.52	15.506	7.842	16.755	15.566	15.391	7.932
%NSD	6.644	6.960	8.397	4.253	8.255	6.531	6.105	3.808	7.580	7.231	6.606	3.132
XOR/XNOR												
%NED	15.48	14.899	15.68	8.706	15.46	10.08	15.695	7.192	16.84	9.302	15.692	5.236
%NSD	6.165	3.713	7.375	3.388	6.736	2.559	6.780	3.154	7.846	3.234	6.674	1.874

Table 6.10: Post-layout simulation results of gates at SS corner.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	6.364	5.099	5.099	2.313	6.529	4.038	4.038	1.314	5.054	3.788	3.788	0.435
%NSD	3.608	2.611	2.611	1.226	3.580	2.225	2.225	0.666	2.972	2.134	2.134	0.201
AND/NAND												
%NED	15.40	14.463	14.60	8.678	16.27	12.72	13.43	8.550	16.27	12.473	14.810	5.553
%NSD	5.336	6.967	6.961	2.479	8.159	5.506	6.04	3.735	7.720	5.535	5.424	2.155
OR/NOR												
%NED	14.92	14.937	15.39	8.523	15.92	12.20	14.526	8.602	15.388	12.110	14.572	5.810
%NSD	7.173	7.464	7.523	3.992	6.820	4.227	6.860	4.022	7.525	5.318	5.980	2.429
XOR/XNOR												
%NED	15.31	15.536	15.73	8.982	16.61	11.38	15.75	7.498	15.172	8.998	14.898	3.753
%NSD	6.588	3.750	5.687	3.529	7.689	2.803	7.196	2.895	7.420	2.370	6.265	1.413

Table 6.11: Post-layout simulation results of gates at FS corner.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	6.180	5.161	5.161	2.620	6.503	5.556	5.556	1.312	5.149	3.699	3.699	0.944
%NSD	3.558	2.857	2.857	1.348	3.641	2.834	2.834	0.722	3.029	1.981	1.981	0.337
AND/NAND												
%NED	15.63	14.840	15.78	9.230	16.40	12.665	13.398	8.798	15.767	14.425	15.048	7.503
%NSD	7.323	7.015	6.840	3.301	7.483	5.764	6.289	3.263	7.484	6.263	5.414	3.085
OR/NOR												
%NED	14.51	15.677	15.86	8.650	16.94	13.32	15.676	8.133	16.142	14.358	15.568	6.451
%NSD	6.983	7.535	7.545	3.703	6.853	5.979	7.325	3.755	7.498	6.276	6.540	2.642
XOR/XNOR												
%NED	15.55	14.085	15.24	8.053	15.18	9.689	15.634	6.977	15.793	8.417	15.619	4.002
%NSD	6.586	3.869	6.900	3.279	6.386	2.484	7.180	2.773	7.532	2.384	6.377	1.514

Table 6.12: Post-layout simulation results of gates at SF corner.

Logic Designs	1 MHz				10MHz				100MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
NOT/BUF												
%NED	5.965	5.594	5.594	2.414	6.237	6.793	6.793	2.065	5.877	3.887	3.887	0.233
%NSD	3.099	2.212	2.212	1.307	3.588	3.638	3.638	0.890	3.456	2.266	2.266	0.110
AND/NAND												
%NED	15.84	16.296	17.20	8.781	16.97	15.29	15.68	8.791	16.816	14.037	15.248	8.108
%NSD	6.887	7.773	8.140	2.840	7.729	7.446	6.992	2.871	6.659	6.401	5.819	3.440
OR/NOR												
%NED	15.08	15.20	16.28	8.434	16.26	14.89	15.547	8.054	16.803	14.317	15.904	7.206
%NSD	6.857	6.898	7.373	2.800	6.537	6.973	6.831	3.139	6.643	6.422	6.020	3.049
XOR/XNOR												
%NED	15.26	12.282	16.95	8.524	16.82	9.894	14.049	8.127	17.511	9.184	15.737	4.673
%NSD	6.502	2.938	8.090	3.096	7.342	2.396	5.286	3.253	8.526	2.980	6.747	1.727

6.5.5 Case Study: 8-bit Montgomery Multiplier

To evaluate the performance of WCS-QuAL an 8-bit Montgomery multiplier was implemented. For comparison, CSSAL, SQAL and SyAL logic versions were also implemented. 8-bit Montgomery multipliers were implemented using Systolic Array Architecture 1 (SAA1) discussed in Chapter 5 of this thesis.

6.5.5.1 Impact of Frequency on NED, NSD, and Average Energy

Simulations for the Montgomery multipliers were performed at 1MHz, 13.56MHz and 100MHz frequencies at 10fF load capacitance. The energy dissipation is measured per cycle for 10 random input patterns. The simulation results for the Montgomery multiplier using WCS-QuAL and the existing logic designs are summarized in Table 6.13.

From Table 6.13, WCS-QuAL exhibits the lowest value of %NED and %NSD for all the simulated frequencies. The ranking of performance (based on %NED and %NSD) changes for existing logic designs at simulated frequencies. At 1MHz and 100MHz, CSSAL is second best followed by the SyAL and SQAL, whereas, at 13.56MHz, SyAL is second best followed by CSSAL and SQAL.

Figure 6.11 shows the graph of average energy dissipated per cycle by WCS-QuAL, CSSAL, SyAL and SQAL at 1MHz, 13.56MHz and 100MHz. As expected from the pre-layout and post-layout simulation results of the gates, WCS-QuAL exhibits the minimum energy dissipation at frequencies above 20MHz. It can be seen that at low frequencies (~1MHz) the four logic families show broadly similar energy consumption with SQAL consuming around 20% less than the others. However, as operating frequency increases, Adiabatic Losses (AL) start to become significant albeit less so in the case of WCS-QuAL because of the low ON-resistance (due to the formation of transmission gate pair N3, P1, and N4, P2 in Figure 6.6(a)), which at 100MHz dissipates the least energy. As the frequency of operation is increased, the AL combined with NAL in SQAL, SyAL, and CSSAL makes the energy dissipation worst in comparison to WCS-QuAL. CSSAL consumes the maximum energy at all simulated frequencies.

Table 6.13: Simulation results comparing the %NED and %NSD of 8-bit Montgomery Multiplier using WCS-QuAL and existing logic.

	1 MHz				13.56 MHz				100 MHz			
	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL	CSSAL	SQAL	SyAL	WCS-QuAL
%NED	1.728	3.646	2.113	0.725	2.829	3.453	1.828	0.749	2.475	3.835	2.894	0.673
%NSD	0.693	0.947	0.592	0.205	0.805	0.666	0.664	0.254	0.795	1.390	0.801	0.189

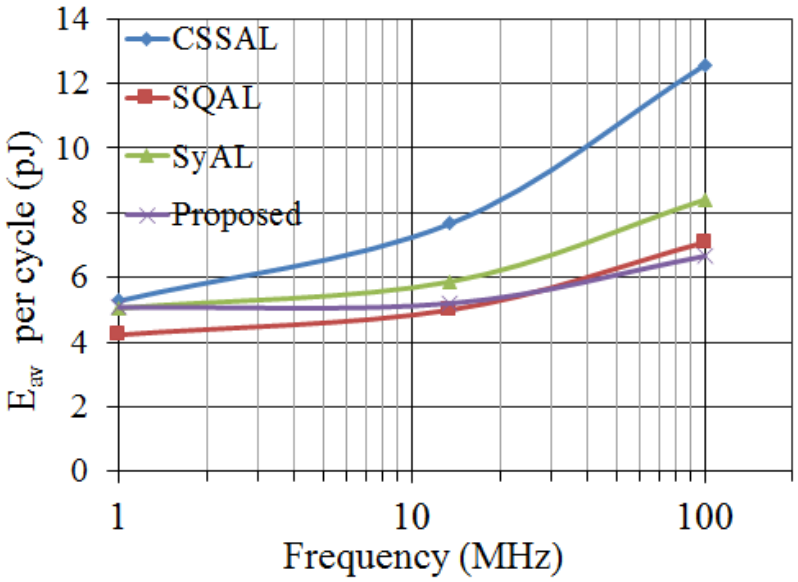


Figure 6.11: Average energy per cycle of 8-bit Montgomery multiplier using CSSAL, SQAL, SyAL, and WCS-QuAL at frequencies ranging from 1MHz to 100MHz.

6.5.5.2 Impact of Power Supply Scaling

Since one of the dominant components of the energy dissipation in adiabatic logic is the supply voltage, energy can be reduced by reducing the power supply. Reduction in supply voltage affects the gate overdrive voltage, V_{GS}-$V_{th,}$ and on-resistance of the transistors in the charging path. With the reduction in supply voltage, an increase in on-resistance is observed [37]. Therefore, it is important to evaluate the impact of power-clock scaling on the performance of the secure adiabatic logic designs.

The power-clock was scaled from 1.8V down to 0.6V. The simulation results of the power-supply/clock scaling at 13.56 MHz and 10fF load capacitance for 10 random inputs are summarized in Table 6.14. Since the simulation results for 1.8V power supply were included in Table 6.13, they are omitted in Table 6.14. It can be seen that WCS-QuAL exhibits the least value of %NED and %NSD compared to CSSAL, SQAL, and SyAL at all the power-clock values.

Table 6.14: Comparison of energy performance of Montgomery multiplier under power supply scaling.

Designs	Power-clock scaling @ 13.56MHz				
	V=0.6	**V=0.8**	**V=1.0**	**V=1.2**	**V=1.5**
CSSAL					
E_{av}(pJ)	1.322	2.803	3.553	4.243	5.249
%NED	1.320	1.732	1.703	1.683	1.209
%NSD	0.342	0.707	0.680	0.716	0.338
SyAL					
E_{av}(pJ)	1.205	2.171	2.619	3.210	4.236
%NED	1.809	1.417	1.666	1.790	1.731
%NSD	0.962	0.752	0.885	0.952	0.920
SQAL					
E_{av}(pJ)	0.999	1.770	2.258	2.786	3.609
%NED	2.941	2.793	2.451	3.169	3.493
%NSD	1.251	1.205	1.224	1.665	1.483
WCS-QuAL					
E_{av} (pJ)	1.297	1.617	2.016	2.637	3.681
%NED	0.643	0.678	0.691	0.793	0.622
%NSD	0.196	0.309	0.304	0.373	0.329

Table 6.15 compares WCS-QuAL and existing logic on the basis of the number of voltage sources required and the number of transistors used in the implementation of 8-bit Montgomery Multiplier. Because WCS-QuAL works on 4-phase power-clocking scheme in cascade stages, 8-bit Montgomery multiplier design using WCS-QuAL requires four voltage sources. On the other hand, SQAL and SyAL also work on 4-phase power-clocking scheme and use charge sharing input, therefore, requiring four voltage sources each for 4-phases of the power-clocks and for generating 4-phases of the charge sharing input. In total, eight voltage sources are required. In [92] the authors omitted, four voltage sources required for generating the 4-phases of the charge sharing inputs in the results. Similarly, CSSAL works on 4-phase power-clocking scheme requiring four voltage sources. Additionally, it uses charge sharing and evaluation input, therefore four phases of each charge-sharing and evaluation inputs need to be generated which requires additional eight voltage sources. In total, twelve voltage sources are required in the design using CSSAL.

Although for the design of the 8-bit Montgomery multiplier, WCS-QuAL uses the highest number of transistors (≈36,000) which is 75.6%, 34.8% and 4% more in comparison to SQAL, SyAL, and CSSAL respectively, it consumes the lowest energy at frequencies ranging from 20MHz to 100MHz.

Table 6.15: Comparison of required voltage sources and transistor counts of WCS-QuAL and the existing logic designs.

Logic	Required number of voltage sources	Number of logic gates in Montgomery Multiplier	Number of transistors per gate	Total number of transistors (Approx.)
WCS-QuAL	4	NOT/BUF 297 AND/NAND 734 OR/NOR 49 XOR/XNOR 716 Reset BUF 199	8 20 20 20 20	36,336
SQAL	8	NOT/BUF 297 AND/NAND 734 OR/NOR 49 XOR/XNOR 716 Reset BUF 199	5 13 13 9 13	20,695
SyAL	8	NOT/BUF 297 AND/NAND 734 OR/NOR 49 XOR/XNOR 716 Reset BUF 199	5 15 15 15 15	26,955
CSSAL	12	NOT/BUF 297 AND/NAND 734 OR/NOR 49 XOR/XNOR 716 Reset BUF 199	9 19 19 19 19	34,935

6.6 Chapter Summary

A background study of PAA resilient logic design is presented. Shortcomings of the existing adiabatic logic designs are identified. As a solution, a novel PAA resilient adiabatic logic is proposed which does not require any charge sharing between the two output nodes of the gate removes the NAL during the evaluation phase of the power-clock and has the symmetric structure. To evaluate and compare the performance of WCS-QuAL, and the existing PAA resilient adiabatic logic designs, simulations were performed

to investigate the impact of frequency and process corner variations on the %NED and %NSD of the single input and 2-input gates. Also, the impact of resistance and capacitances of the wires is investigated by performing post-layout simulations of the single and 2-input gates.

To evaluate and compare the performance of WCS-QuAL, and the existing secure adiabatic logic designs in the complex circuit, an 8-bit Montgomery multiplier is designed and the impact of frequency variations and power-clock scaling on the %NED and %NSD of the Montgomery multiplier is investigated. The pre-layout and post-layout simulation results of the gates show that WCS-QuAL outperforms the existing secure adiabatic logic at all process corners at all simulated frequencies and shows the least sensitivity to process corners. In addition, all the 2-input gates using WCS-QuAL consume nearly equal energy thus, exhibiting data-independence and gate-function independence. Also, the Montgomery multiplier using WCS-QuAL exhibits the least (i.e. best) value of %NED and %NSD under frequency variations and power supply scaling and dissipates the lowest energy at frequencies ranging from 20MHz to 100MHz.

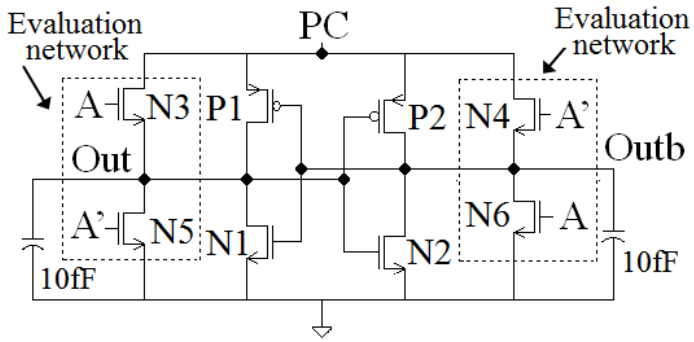
7. Power Analysis Attack Resilient Adiabatic logic with Single Charge Sharing Transistor

In this chapter, a novel PAA resilient adiabatic logic with single charge sharing transistor is proposed as a solution to the problem of negative peak current variations in the previously proposed, WCS-QuAL. The condition for which the two output nodes of WCS-QuAL remain unbalanced is identified. A detailed performance evaluation of the proposed logic 2 is performed. The proposed logic 2 is compared with WCS-QuAL and the two recently proposed secure adiabatic logic designs at gate level and in a complex circuit design (Montgomery multiplier). Simulations are performed to investigate the impact of frequency variations and power-supply scaling on the resistance of the proposed logic 2, WCS-QuAL and the existing secure diabatic logic designs against PAA.

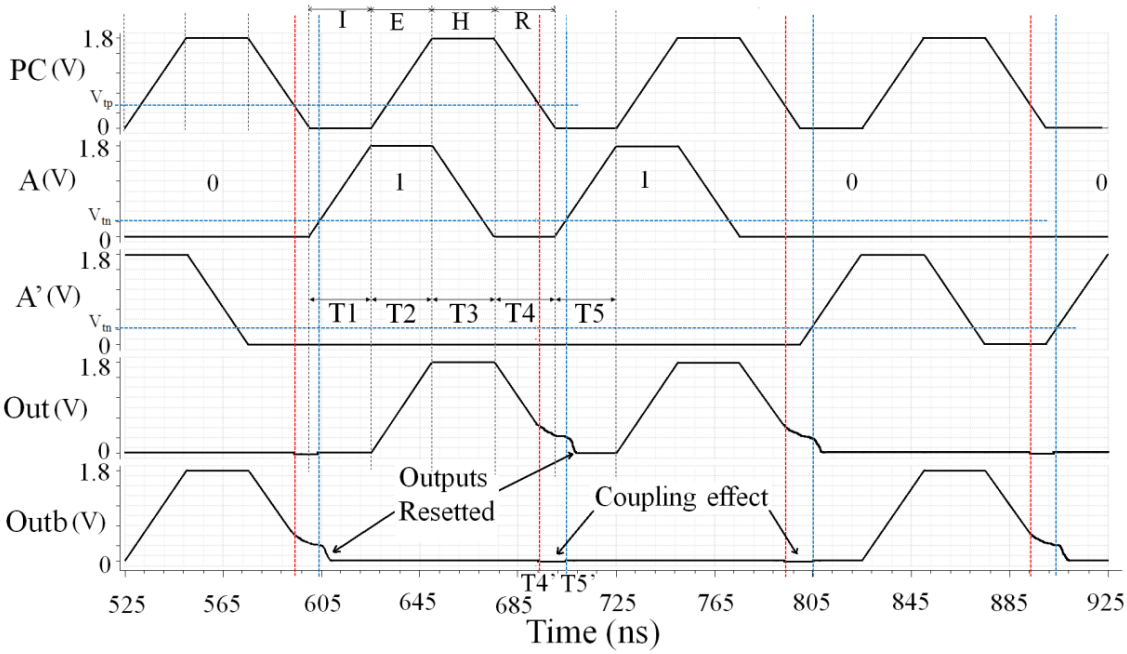
7.1 Introduction

WCS-QuAL proposed in Chapter 6 has been proven to outperform the existing secure adiabatic logic designs, CSSAL, SQAL, and SyAL. It has a symmetric structure, performs well under process corner variations, power-supply scaling and exhibits the least value of %NED and %NSD. However, it was identified that WCS-QuAL exhibits variations in negative peak currents due to the unequal charge on the two output nodes for the duration T4' to T5' (indicated in Figure 7.1 (b)). As outlined in Chapter 6, T4' corresponds to the duration in the recovery phase, when the PC falls below the threshold voltage of the cross-coupled pMOS transistors and charge is left on the evaluating output node. On the other hand, T5' corresponds to the time in the idle phase of the PC when the next input arrives, and its gate voltage has not yet reached the threshold voltage (V_{tn}). From T4' to T5', the two output nodes have a different charge on them, leading to variations in negative peak currents. The variations in the negative peak currents can be reduced if the charge at the two output nodes is shared. Therefore, charge sharing transistor can be used between the

two output nodes to share the charge during the idle phase of the PC. This way, the difference in charge on the two output nodes for the Time T5' can be removed. As a result, another PAA resilient adiabatic logic using single charge sharing transistor (proposed logic 2) is proposed. It should be noted that due to the symmetric structure of the proposed logic 2 only single charge sharing transistor is required, unlike the existing secure adiabatic logic designs which use two or more than two charge sharing transistors to share the charge on the two output nodes.



(a)



(b)

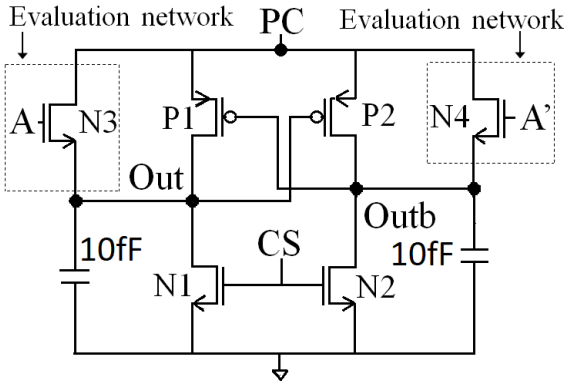
Figure 7.1: (a) WCS-QuAL NOT/BUF gate (b) Simulation result at 10MHz.

Recently, two secure adiabatic logic designs called as Symmetric Pass Gate Adiabatic Logic (SPGAL) [101], [102] and Energy Efficient Secure Positive Feedback Adiabatic Logic (EE-SPFAL) [103] were proposed. SPGAL and EE-SPFAL are based on Positive

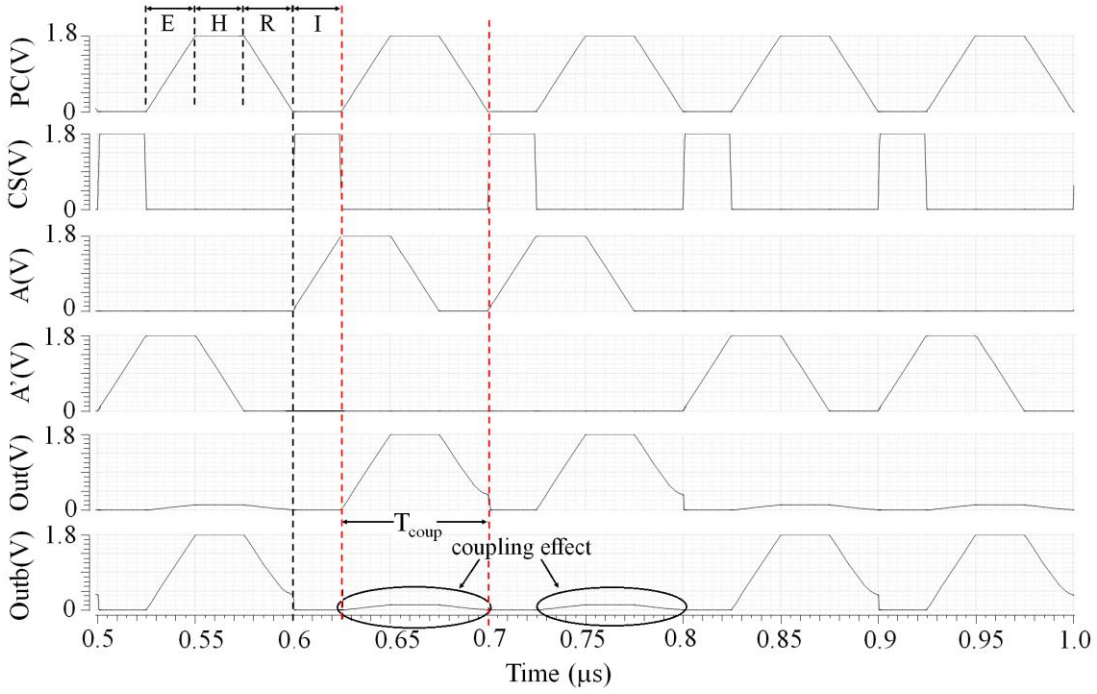
Feedback Adiabatic Logic (PFAL) [21] and do not suffer from NAL during the evaluation phase of the power-clock. EE-SPFAL is the enhancement of SPGAL. EE-SPFAL [103] and SPGAL [101], [102] are proven to outperform the three existing secure logic designs, CSSAL [87]-[90], SyAL [91] and SQAL [92]. Therefore, in this chapter, proposed logic 2 will be compared to WCS-QuAL, SPGAL, and EE-SPFAL.

7.1.1 Symmetric Pass Gate Adiabatic Logic (SPGAL)

Figure 7.2 (a) and (b) show the schematic of the SPGAL NOT/BUF gate and its simulation result at 10MHz respectively.



(a)



(b)

Figure 7.2: (a) SPGAL NOT/BUF [101], [102] gate (b) Simulation result at 10MHz.

The operation of SPGAL gate is explained through the design of a NOT/BUF gate. The simulation result shows the PC, charge sharing input, CS, input A, its complement A', and the output nodes, 'Out' and 'Outb'. As mentioned before, SPGAL [101], [102] is based on PFAL [21] and works on 4-phase power-clocking scheme. The evaluation networks are connected between the power-clock and the two output nodes and are indicated in the schematic of the NOT/BUF gate of Figure 7.2 (a). The Transistors N3 and N4 are the input transistors whereas; N1 and N2 are the charge sharing/discharge transistors. P1 and P2 are the cross-coupled transistors responsible for holding the output nodes to their respective voltages. The operation is explained for A= '1', A'= '0'.

During the Idle (I) phase of the PC, input A is rising (A' is logic '0') and turns ON the transistor N3 after the input reaches the threshold voltage. Also, the charge sharing transistors, N1 and N2 are turned ON which connect the two output nodes, 'Out' and 'Outb' to ground. This way, both the output nodes are discharged to ground before the evaluation of the next inputs.

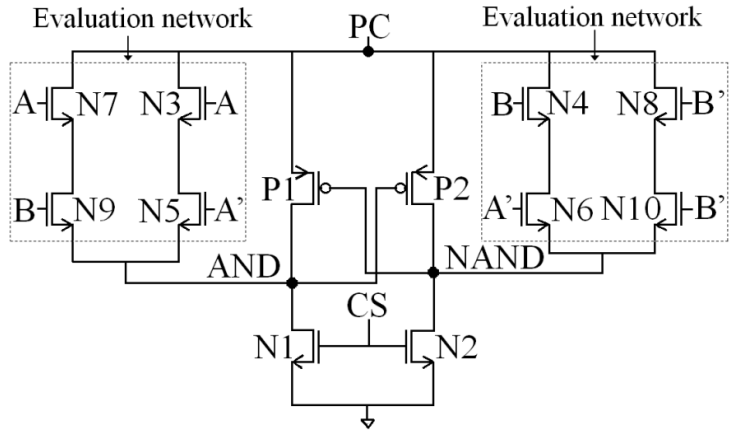
During the Evaluation (E) phase, the charge sharing transistors N1 and N2 are turned OFF. The input A is at logic '1' (A' is '0') and the PC ramps up. The output node 'Out' will follow the PC through transistors N3 and P1 from 0 to V_{DD}-V_{tn} and V_{tp} to V_{DD} respectively and thus, does not suffer from NAL.

During the Hold (H) phase, the input, A ramps down and the transistor N3 is switched OFF when the gate-to-source voltage falls below the threshold voltage, $V_{tn.}$

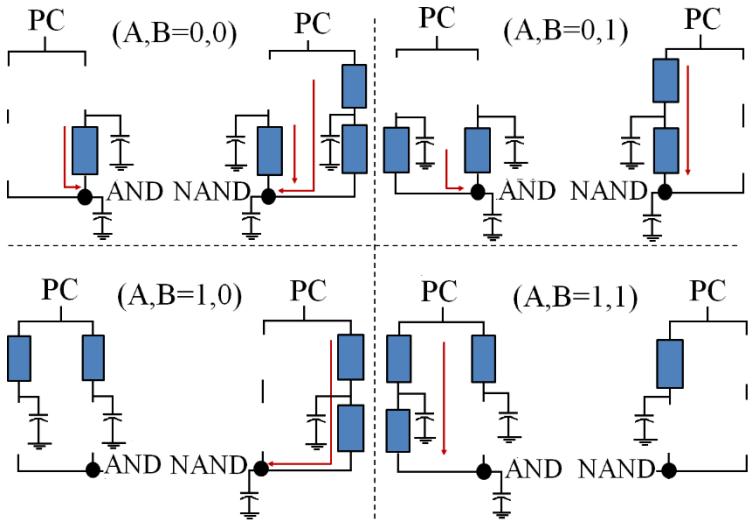
During the Recovery (R) phase, the input transistors are OFF and the charge on the output node 'Out' will be recovered back to the PC through P1. The charge is recovered till PC falls below the threshold voltage, $|V_{tp}|$. The node 'Out' stays at V_{tp}, leading to NAL. Thus, SPGAL suffers from NAL only during the recovery phase of the PC. The leftover charge will be discharged to ground in the idle phase when the charge sharing transistors are turned ON.

The structure of SPGAL is unstable due to the absence of cross-coupled nMOS. It is because when one of the output nodes follows the PC, the complementary node gets coupled to it during evaluation, hold, and recovery phase of the power-clock (T_{coup}), leading to sever coupling effect. This results in the complementary node voltage to rise above the threshold voltage (V_{tn}). The coupling effect is indicated in the timing diagram.

Figure 7.3 (a) and (b) show the AND/NAND gate using SPGAL and its equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively.



(a)



(b)

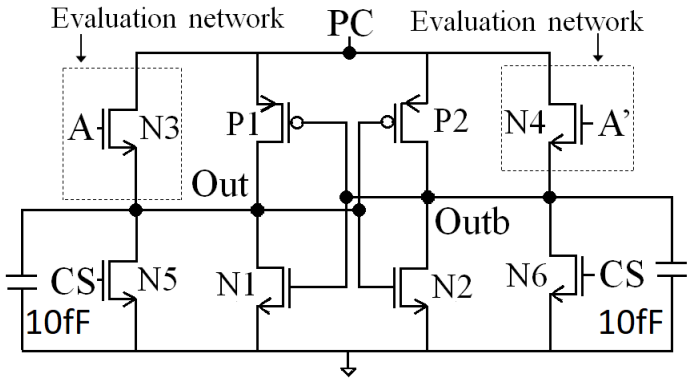
Figure 7.3: (a) AND/NAND gate using SPGAL [101], [102] (b) Equivalent RC models during evaluation phase.

It can be seen that, for each of the 4 input combinations, the capacitance at the two output nodes is different leading to asymmetric structure and data-dependent behaviour.

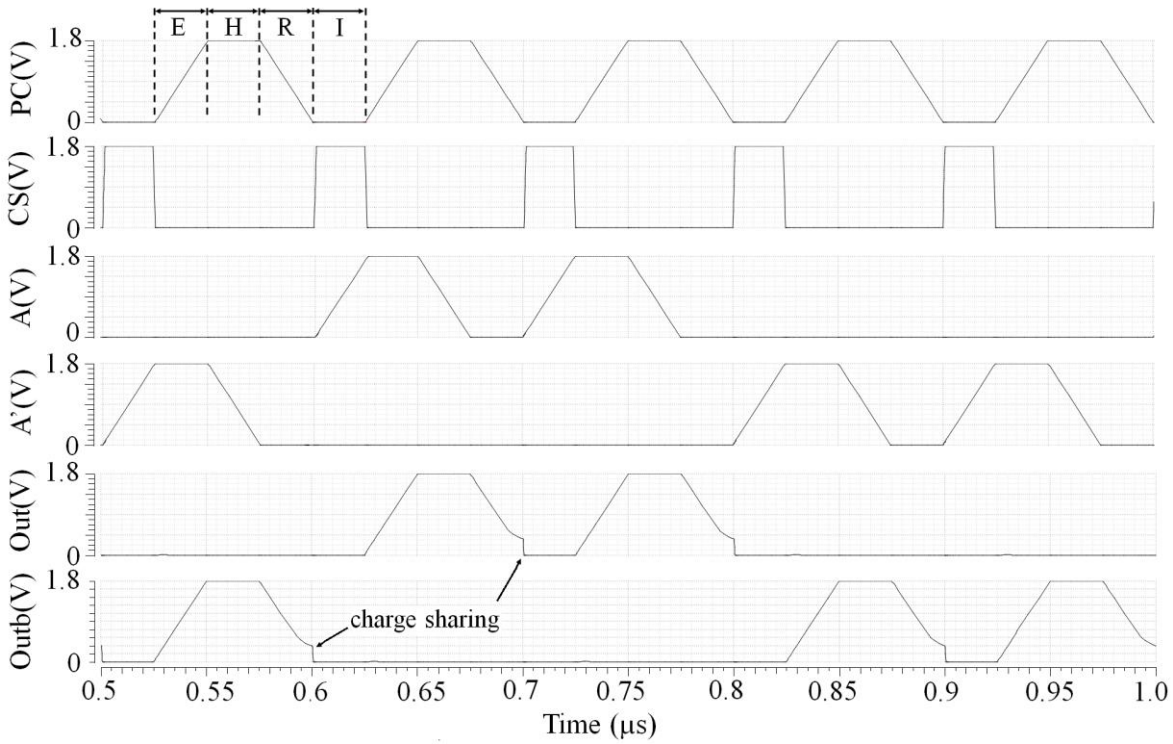
7.1.2 Energy Efficient Secure Positive Feedback Adiabatic Logic (EE-SPFAL)

Figure 7.4 (a) and (b) shows the schematic of the NOT/BUF gate using EE-SPFAL [103] and its simulation result at 10MHz respectively. The operation of EE-SPFAL gate is explained through the design of a NOT/BUF gate. The simulation result shows the PC,

charge sharing input, CS, input A, its complement A', and the output nodes, 'Out' and 'Outb'. EE-SPFAL is also based on PFAL [21] and is an enhancement of SPGAL with an addition of cross-coupled nMOS transistors. It also works on 4-phase power-clocking scheme. The evaluation networks are connected between the power-clock and the two output nodes and are indicated in the schematic of the NOT/BUF gate. The transistors N3 and N4 are the input transistors whereas; N5 and N6 are the charge sharing/discharge transistor. P1, P2, N1, and N2 form the cross-coupled latch responsible for holding the output nodes to their respective voltages.



(a)



(b)

Figure 7.4: (a) EE-SPFAL NOT/BUF [103] gate (b) Simulation result at 10MHz.

The operation is explained for A= ‘1’, A’= ‘0’.

During the Idle (I) phase of the PC, input A is rising and turns ON the transistor N3 after input reaches the threshold voltage. Also, the charge sharing transistors, N5 and N6 are turned ON which connect the two output nodes, ‘Out’ and ‘Outb’ to ground. In this manner, both the output nodes are discharged to ground before the evaluation of the next inputs.

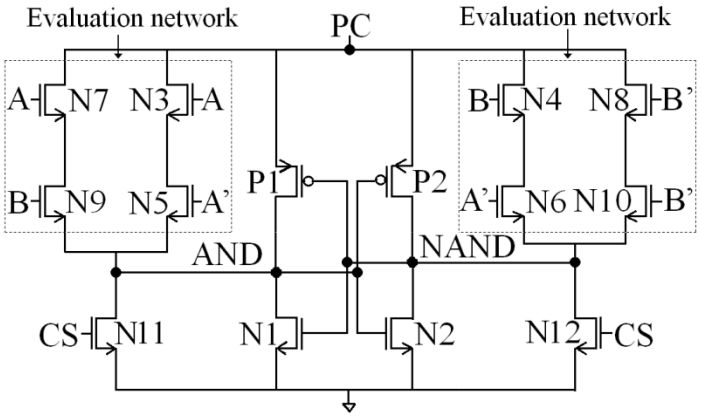
During the Evaluation (E) phase, charge sharing transistors N5 and N6 are turned OFF. The input A is at logic ‘1’ (A’ is ‘0’) and the PC ramps up. The output node ‘Out’ will follow the PC through transistors N3 and P1 from 0 to V_{DD}-V_{tn} and V_{tp} to V_{DD} respectively and thus, does not suffer from NAL.

During the Hold (H) phase, the input A ramps down and the transistor N3 is switched OFF when the gate-to-source voltage falls below the threshold voltage, $V_{tn.}$ The output nodes ‘Out’ and ‘Outb’ are held at their respective voltages due to the cross-coupled latch.

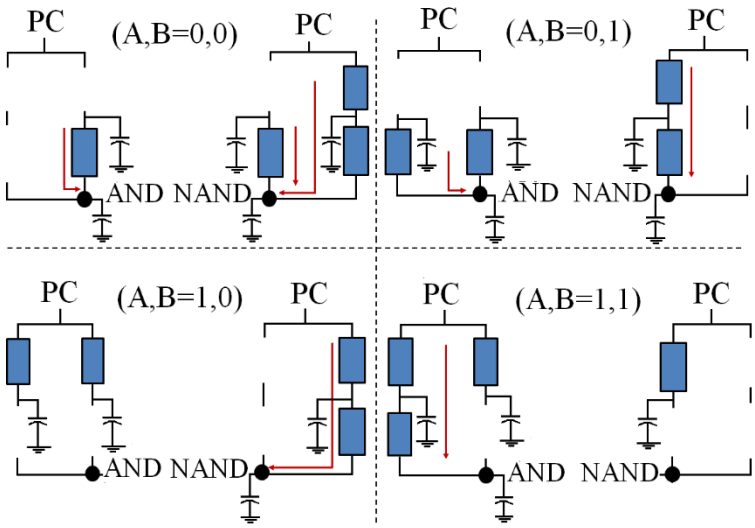
During the Recovery (R) phase, the input transistors are OFF and the charge on the output node ‘Out’ will be recovered back to the PC through transistor P1. The charge is recovered till PC falls below the threshold voltage, $|V_{tp}|$. The node ‘Out’ stays at V_{tp}, leading to NAL. Like SPGAL, EE-SPFAL also suffers from NAL during the recovery phase of the PC. The left-over charge on the output node will be discharged to ground in the idle phase when the charge sharing transistors are turned ON.

Unlike SPGAL, EE-SPFAL, due to the presence of cross-coupled nMOS transistors, N1 and N2, the two output nodes remain floating only for the part of the recovery phase when the PC falls below the threshold voltage of the pMOS transistor and therefore suffers from coupling effect only for the part of recovery phase of the power-clock.

Figure 7.5 (a) and (b) show the AND/NAND gate using EE-SPFAL and its equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. For each of the 4 input combinations, the capacitance at the two output nodes is different leading to data-dependent behaviour and asymmetric structure.



(a)



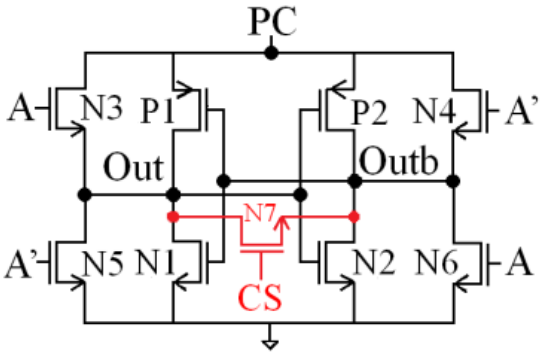
(b)

Figure 7.5: (a) AND/NAND gate using EE-SPFAL [103] (b) Equivalent RC models during the evaluation phase.

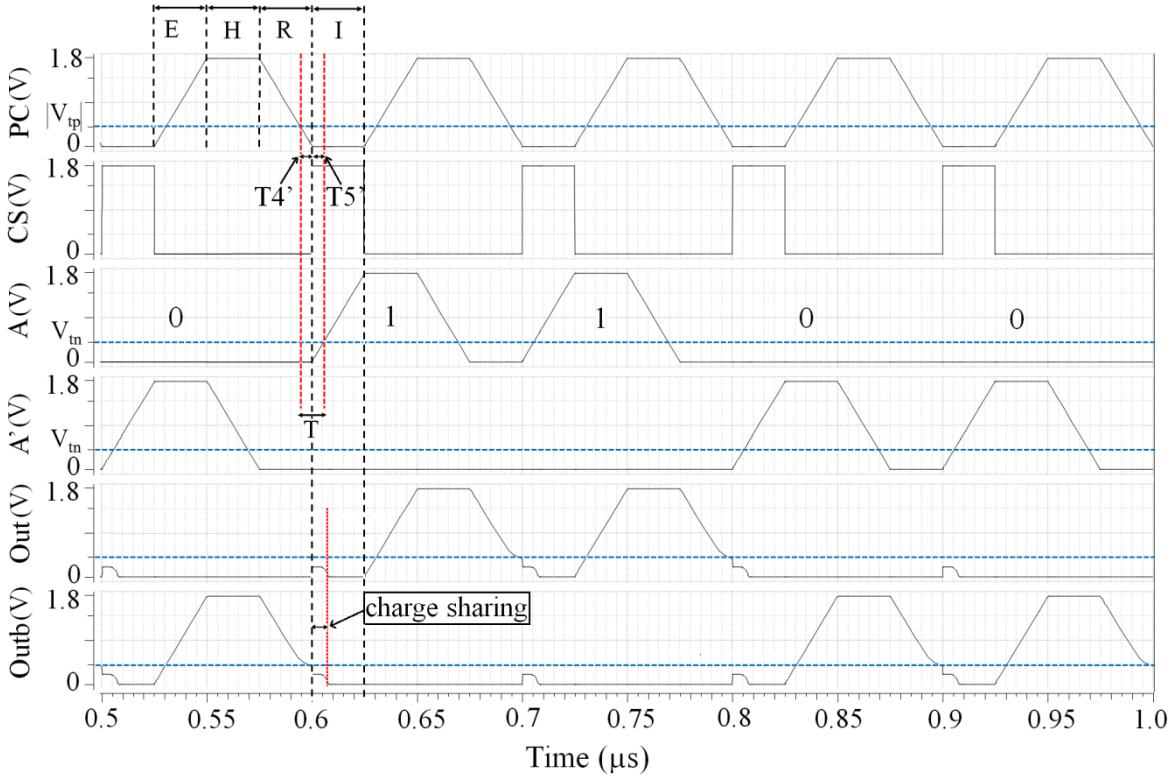
7.2 Proposed logic 2 using Single Charge Sharing Transistor

Figure 7.6 (a) and (b) show a NOT/BUF gate using proposed logic 2, its simulation result at 10MHz and the current peaks for 4 input transitions of the NOT/BUF gate respectively. It also uses dual duplicate evaluation network one connected between the power-clock and the output nodes and the other connected between the output nodes and the ground. The charge sharing transistor balances the charge during the idle phase of the PC when the inputs have not yet reached the threshold voltage of the transistors. It also connects the two output nodes to the ground before the evaluation of the next inputs. Charge sharing transistor uses a clock signal having 25% duty cycle with steeper rise and fall time.

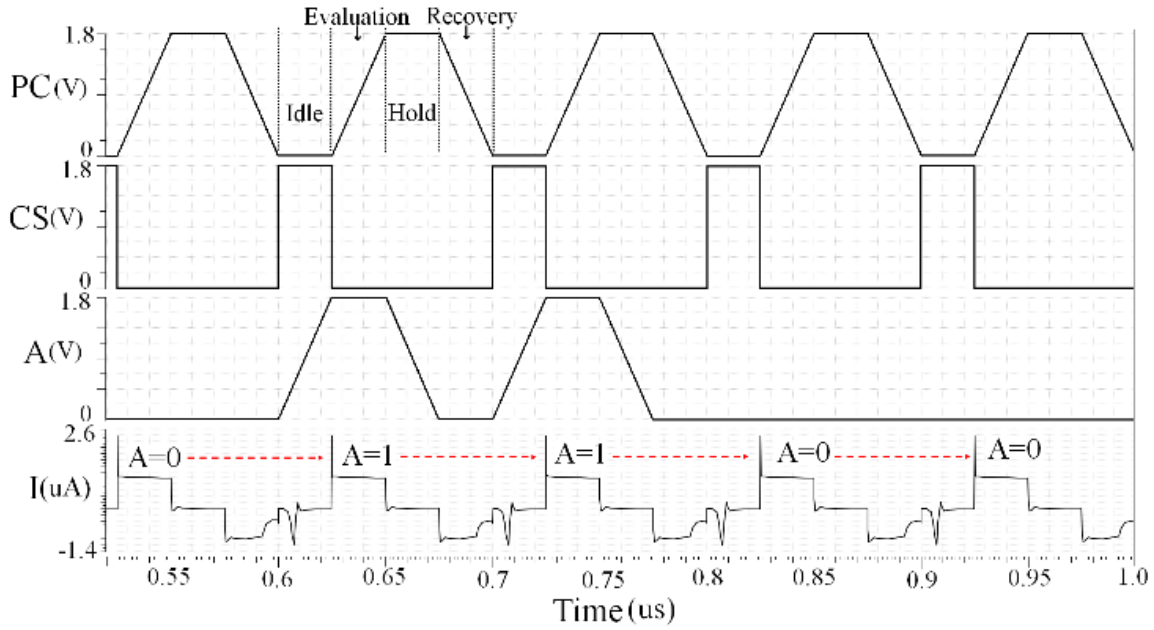
The operation of proposed logic 2 is explained through the design of a NOT/BUF gate. N3, N4, N5, and N6 are the input transistors, P1, P2, N1 and N2 forms the cross-coupled latch responsible for holding the output nodes to their respective voltages and N7 is the charge sharing transistor. The timing diagram shows the PC, charge sharing input, CS, input A, its complement A', and the output nodes, 'Out' and 'Outb'. Current peak graph shows the PC, charge sharing input, CS, input A, and current peaks for 4-input transitions of the NOT/BUF gate. Proposed logic 2 also works on 4-phase power-clocking scheme. The operation is explained for A= '1', A'= '0'.



(a)



(b)



(c)

Figure 7.6: (a) Proposed logic 2 NOT/BUF gate (b) Simulation result at 10MHz (c) Current peaks for 4 input transitions.

During the Idle phase (I) when input A is rising, transistors N3 and N6 are turned ON after they reach the threshold voltage. Also, charge sharing transistor N7 is turned ON. For the interval, the inputs are not ON; the charge sharing transistor ensures that both the output nodes have the same charge. When transistors N3 and N6 are turned ON, the input transistor, N3 connects the output node 'Out' to PC (which is at logic '0') and makes it zero. Also, transistor N6 causes the output node 'Outb' to connect to ground. Also, transistor N7 is connected between 'Out' and 'Outb' thus, both the output nodes are discharged to '0' before the evaluation of the next input.

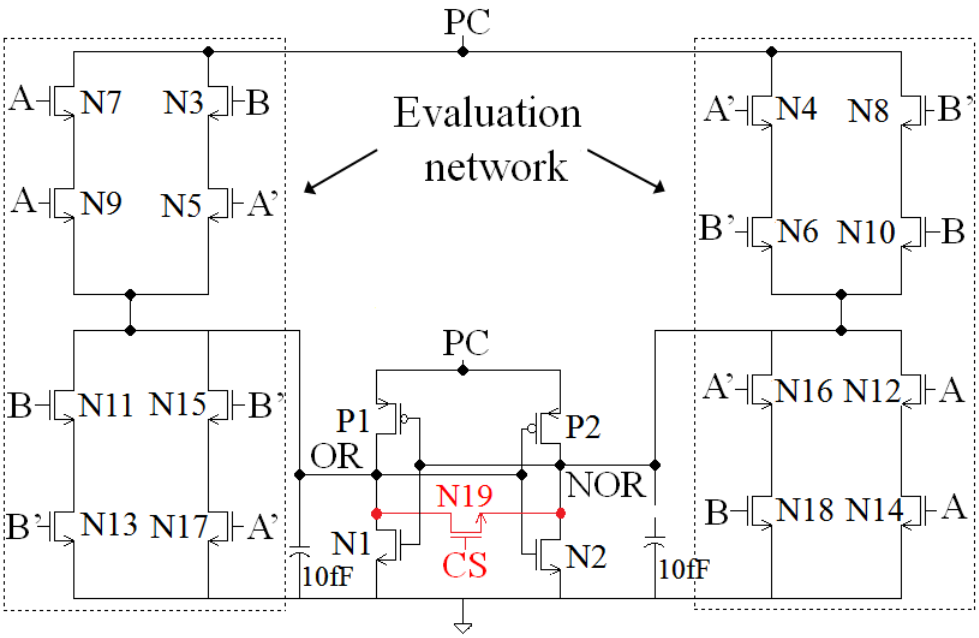
During the Evaluation phase (E), input A is logic '1' (A' is logic '0') and the PC ramps up. The Output node, 'Out' follows the PC through transistors, N3 and P1 from 0 to V_{DD}-V_{tn} and V_{tp} to V_{DD} respectively and thus, does not suffer from NAL.

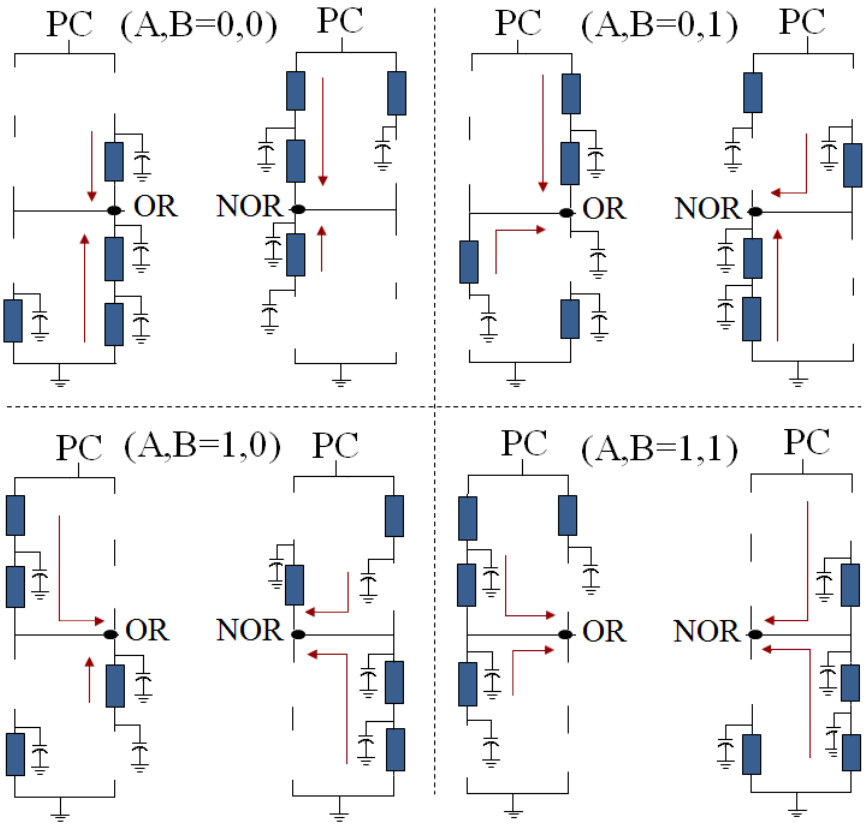
During the Hold phase (H), input A ramps down and the transistor N3 and N6 are switched OFF when the gate-to-source voltage falls below the threshold voltage, $V_{tn.}$ The output nodes, 'Out' and 'Outb' are held at their respective voltages due to the cross-coupled transistors (P1, P2, N1, and N2).

During the Recovery phase (R), the PC ramps down and the charge on the output node, 'Out' is recovered back to the PC through the transistor, P1. The charge is recovered till the PC falls below the threshold voltage, $|V_{tp}|$ of P1. At the time, T4', P1 is turned off and the node 'Out' stays at V_{tp}. The leftover charge will be discharged to ground in the idle phase when the charge sharing transistor is turned ON and the next input arrives and its gate voltage exceeds the threshold voltage (V_{tn}).

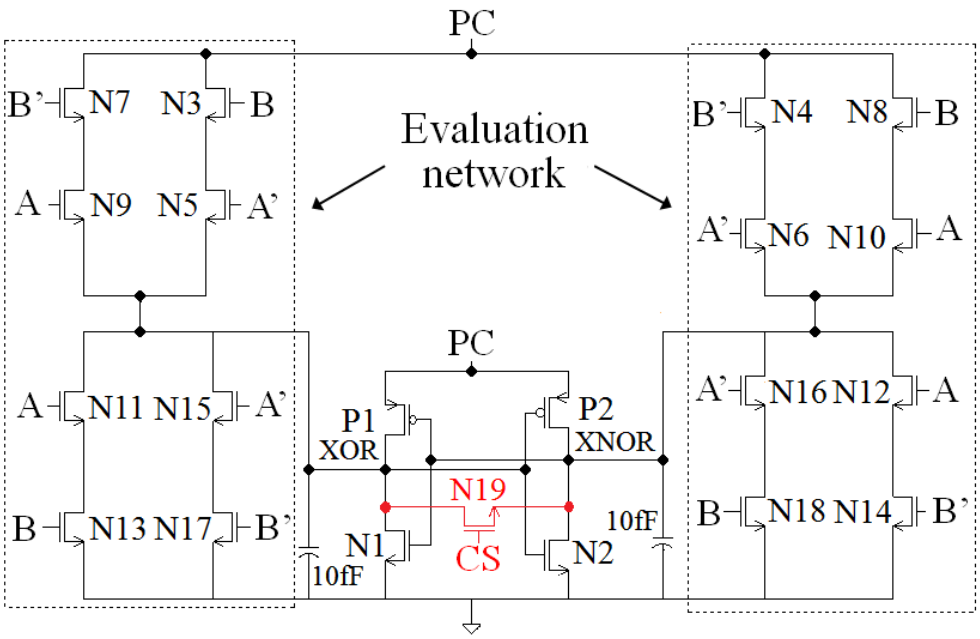
From Figure 7.6 (c) It can be seen that our proposed logic 2 exhibits nearly same current peaks for all the 4 input transitions in NOT/BUF gate.

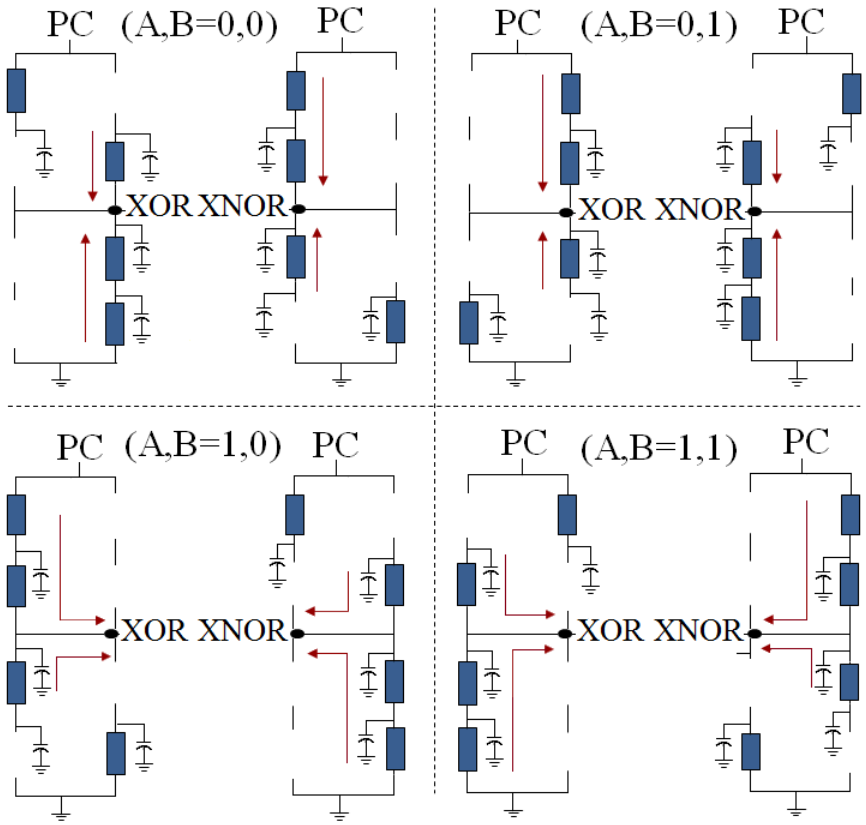
Figure 7.7 (a), (b) (c) and (d) show the schematics of OR/NOR, XOR/XNOR and AND/NAND, gates using the proposed logic 2 and their equivalent RC models of the internal nodes for 4 input combinations during the evaluation phase respectively. It can be seen that, in each gate, for each of the 4 input combinations, the capacitance at the two output nodes is same leading to symmetric structure and data-independent behaviour, unlike SPGAL and EE-SPFAL. All the 2-input logic gates using proposed logic 2 have the same structure and an equal number of transistors, except the position of the input signals.



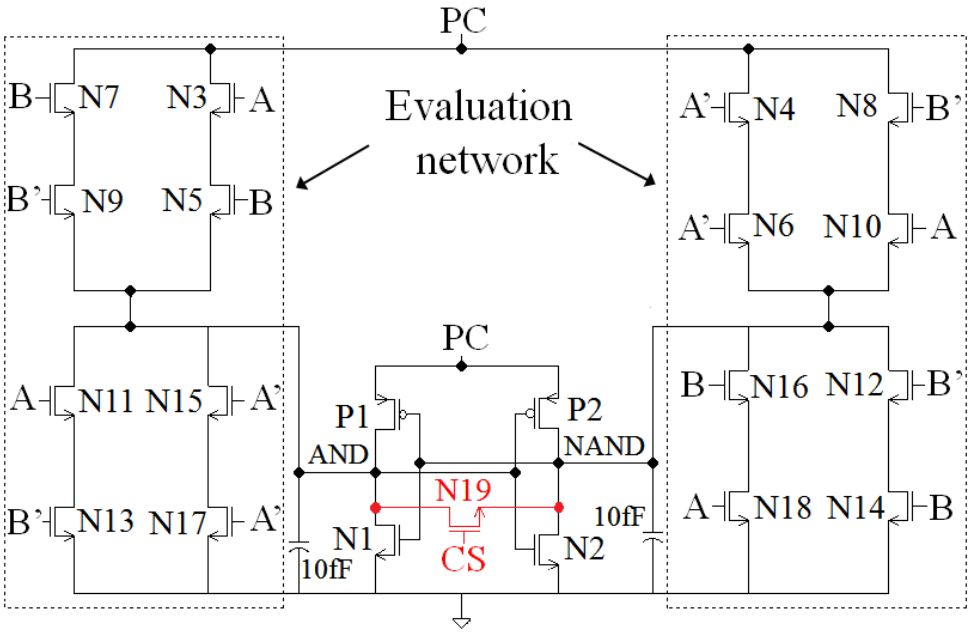

PC
(A,B=0,0)
PC
OR
NOR
PC
(A,B=0,1)
PC
OR
NOR
PC
(A,B=1,0)
PC
OR
NOR
PC
(A,B=1,1)
PC
OR
NOR

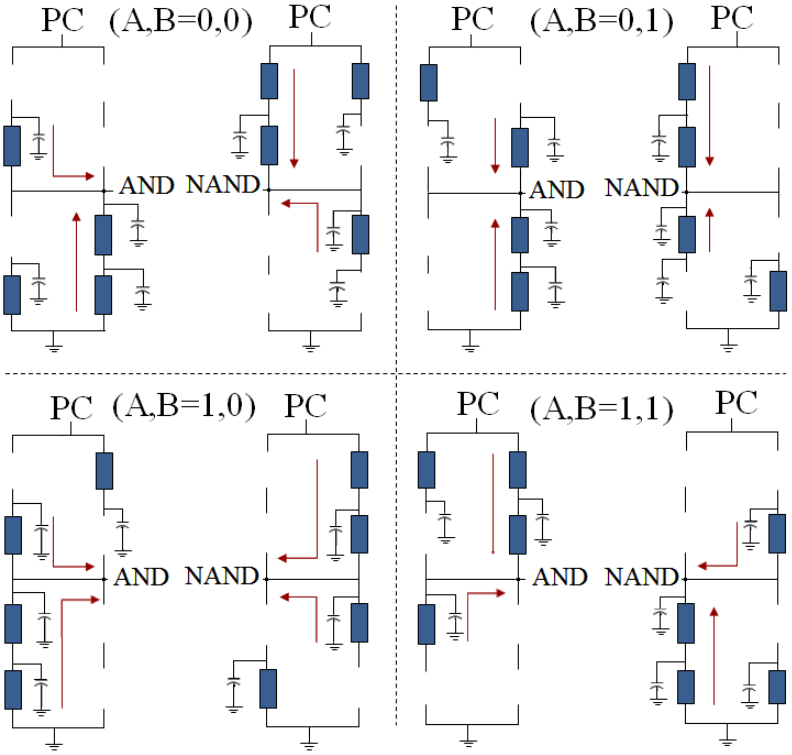
(a)


PC
B'
N7
N3
B
A
N9
N5
A'
Evaluation network
B'
N4
N8
B
A'
N6
N10
A
PC
A
N11
N15
A'
P1
XOR
P2
XNOR
N19
A'
N16
N12
A
B
N13
N17
B'
N1
10fF
CS
N2
10fF
B
N18
N14
B'


PC
(A,B=0,0)
PC
XOR XNOR
PC
(A,B=0,1)
PC
XOR XNOR
PC
(A,B=1,0)
PC
XOR XNOR
PC
(A,B=1,1)
PC
XOR XNOR

(b)


PC
Evaluation network
B
N7
N3
A
B'
N9
N5
B
A'
N4
N8
B'
A'
N6
N10
A
PC
A
N11
N15
A'
B'
N13
N17
A'
P1
AND
P2
NAND
N19
N1
CS
N2
10fF
10fF
B
N16
N12
B'
A
N18
N14
B



(c)

Figure 7.7: 2-input gates using proposed logic 2 and their equivalent RC models for evaluation phase (a) OR/NOR (b) XOR/XNOR. (c) AND/NAND.

7.3 Simulation Results

Simulations for all the secure adiabatic logic designs were performed with Spectre simulator using Cadence EDA tool in a 'typical-typical' (TT) process corner using TSMC 180nm CMOS process at 1.8V power supply. The transistor sizes for all the designs were set to the technology minimum ($W_{min}=W_n=W_p=220nm$, $L_{min}=L_n=L_p=180nm$) and the load capacitance was chosen as 10fF.

The simulations were performed at 1MHz, 10MHz and 100MHz frequencies. The energy dissipation was measured per cycle for 4 and 16 input transitions for NOT/BUF and 2-input gates using proposed logic 2, WCS-QuAL, SPGAL, and EE-SPFAL.

The maximum energy (E_{max}), minimum energy (E_{min}), the average energy (E_{av}), and the standard deviation (σ) for 4 and 16-input transitions of single and 2-input gates were measured. Normalized Energy Deviation (NED) and Normalised Standard Deviation (NSD) were obtained according to (6.1) and (6.2) and were adopted as the metric for measuring the

resistance of the proposed logic 2, WCS-QuAL, SPGAL and EE-SPFAL against PAA. Greater the difference between the maximum and minimum energy higher the %NED and %NSD and higher the cell's susceptibility to PAA.

Figure 7.8 shows the current peaks and output node voltages for 4 input transitions of NOT/BUF gate using the proposed logic 2 and WCS-QuAL. The structure of WCS-QuAL is similar to the structure of proposed logic 2 without the charge sharing transistor, N7. From Figure 7.8, WCS-QuAL exhibits variations in negative peak currents. The variations in the negative peak currents arise due to the unbalanced charge on the two output nodes during T4' to T5'.

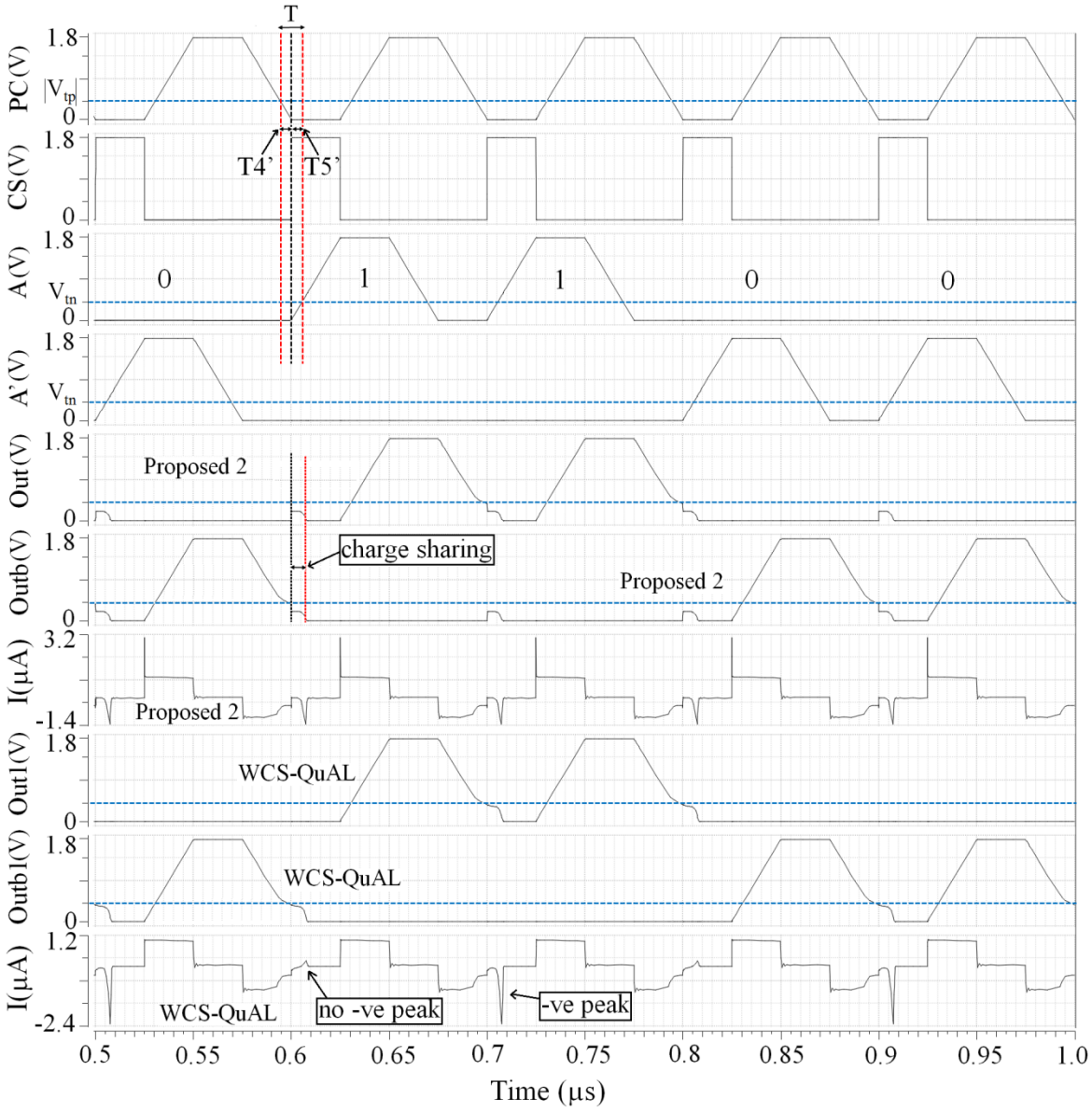


Figure 7.8: Current peaks and output node voltages for 4-input transitions of NOT/BUF gate using proposed logic 2 and WCS-QuAL.

In the proposed logic 2, the charge is shared during the time T5' causing the two output nodes to have equal charge, therefore, exhibiting nearly similar peak currents for each input transitions.

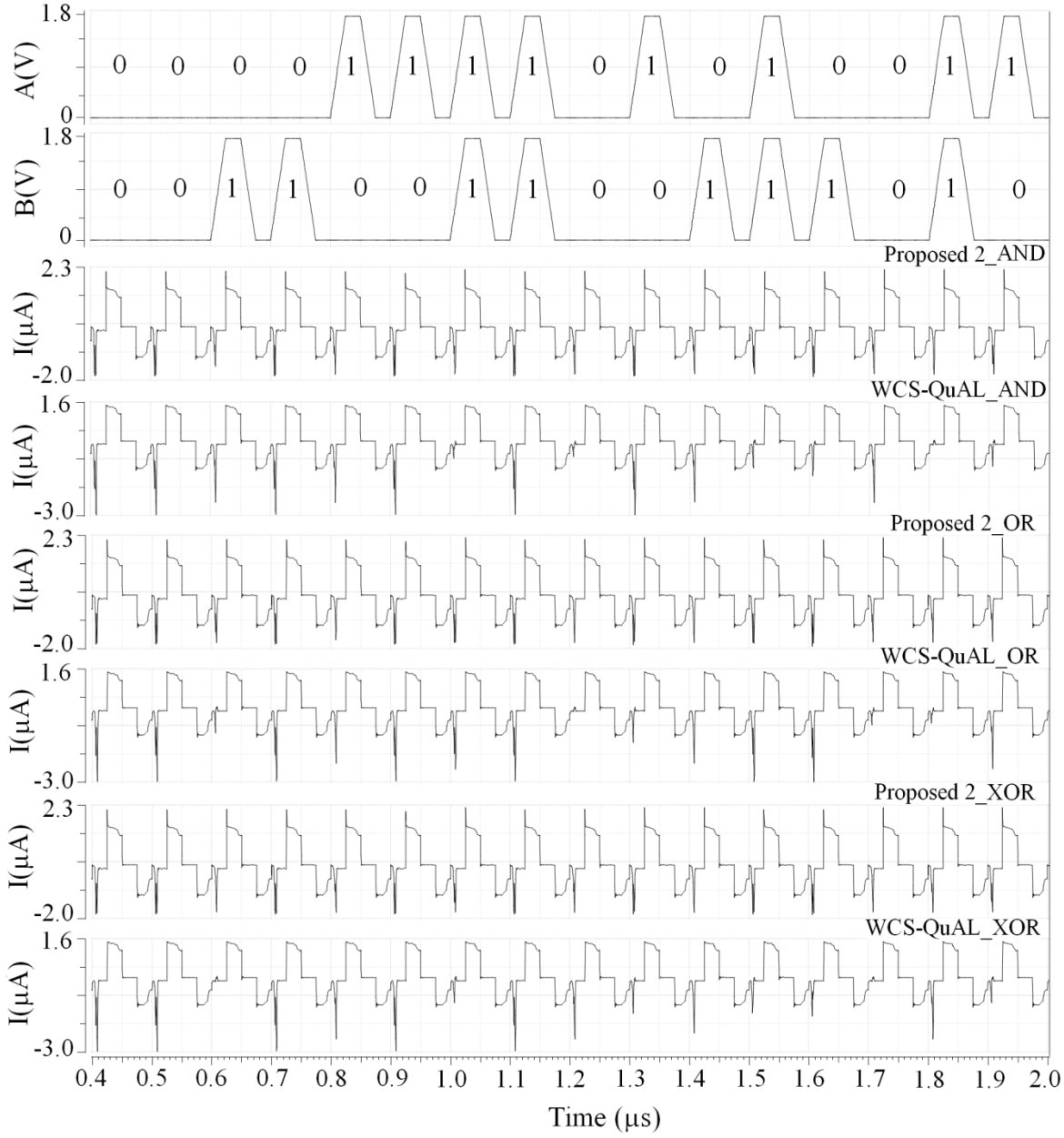


Figure 7.9: Current peaks for 16 input transitions of AND/NAND, OR/NOR and XOR/XNOR gates using proposed logic 2 and WCS-QuAL

Fig. 7.9 shows the current peaks for 16 input transitions of AND/NAND, OR/NOR and XOR/XNOR gates using the proposed logic-2 and WCS-QuAL. It can be seen that WCS-QuAL exhibits variations in negative peak currents. Proposed logic 2 on the other hand, exhibits nearly similar peak currents for all input transitions for all the 2-input gates.

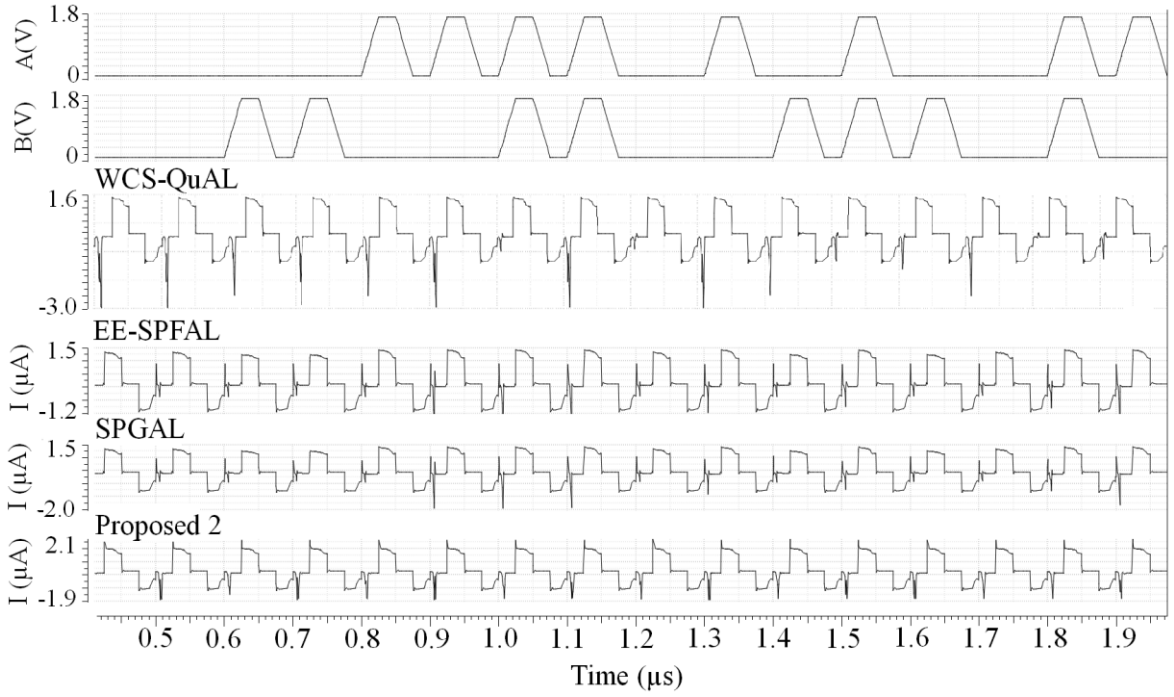


Figure 7.10: Current peaks for 16 input transitions of AND/NAND gate using WCS-QUAL, EE-SPFAL, SPGAL and proposed logic 2.

Figure 7.10 shows the current peaks for 16 input transitions of AND/NAND gate using WCS-QUAL, EE-SPFAL, SPGAL and proposed logic 2. It can be seen that proposed logic 2 exhibits nearly similar peak currents both in the negative and the positive directions for all the 16 input transitions. On the other hand, WCS-QuAL, SPGAL and EE-SPFAL exhibit different peak currents for all the input transitions.

7.3.1 Impact of Frequency Variations on NED and NSD

The simulation results of the evaluated gates using proposed logic 2, WCS-QuAL, SPGAL and EE-SPFAL at 1MHz, 10MHz and 100MHz at 10fF load capacitance are summarised in Table 7.1. Based on the %NED and %NSD, the performance of the proposed logic 2 is the best as it exhibits the least value of %NED and %NSD followed by WCS-QuAL, EE-SPFAL, and SPGAL at all simulated frequencies.

Table 7.1: Simulation results comparing the %NED and %NSD of NOT/BUF, AND/NAND, OR/NOR and XOR/XNOR gates.

Logic Gates	1 MHz				10 MHz				100MHz			
	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2
NOT/BUF												
E_{av} (fJ)	1.755	1.792	1.792	1.867	2.387	2.455	2.479	2.538	5.352	5.725	5.685	5.710
%NED	1.920	0.501	0.445	0.267	0.209	0.406	0.523	0.393	0.816	0.400	0.351	0.279
%NSD	0.725	0.255	0.257	0.104	0.114	0.147	0.281	0.176	0.365	0.174	0.176	0.122
AND/NAND												
E_{av} (fJ)	5.740	5.772	5.837	5.869	6.053	6.170	6.438	6.459	9.602	9.787	10.674	10.690
%NED	9.800	6.756	0.562	0.458	7.969	6.320	0.186	0.139	7.672	6.168	0.187	0.186
%NSD	2.355	2.290	0.167	0.111	1.992	2.460	0.047	0.033	1.843	3.163	0.076	0.093
OR/NOR												
E_{av} (fJ)	4.784	5.028	5.838	5.868	5.116	5.506	6.439	6.458	8.027	8.648	10.674	10.692
%NED	9.457	7.961	0.528	0.509	7.094	5.938	0.124	0.123	6.668	3.913	0.187	0.187
%NSD	4.722	3.233	0.165	0.119	3.705	2.647	0.034	0.061	1.698	1.099	0.076	0.060
XOR/XNOR												
E_{av} (fJ)	3.328	3.529	5.840	5.870	3.908	4.138	6.440	6.460	7.390	8.027	10.676	10.691
%NED	1.430	0.537	0.545	0.508	0.127	0.096	0.047	0.030	0.607	0.174	0.187	0.186
%NSD	0.310	0.146	0.183	0.137	0.057	0.024	0.019	0.007	0.148	0.062	0.068	0.050

Table 7.1 also shows that the energy dissipation of proposed logic 2 and WCS-QuAL for 2-input gates is greater than SPGAL and EE-SPFAL at all simulated frequencies. It is because they use dual duplicate evaluation network one connected between the power-clock and the output nodes and the other connected between the output nodes and ground thus have high internal node capacitance than SPGAL and EE-SPFAL. At lower values of load capacitances, the load at the output nodes of proposed logic 2 and WCS-QuAL will mainly be dominated by their internal load capacitance and thus dissipate more energy than SPGAL and EE-SPFAL. The energy dissipation of the proposed logic 2 and WCS-QuAL is comparable.

7.3.2 Intra-Operation Energy Variability

Table 7.2 shows the average energy dissipation for all possible input transitions of AND/NAND, OR/NOR and XOR/XNOR gates using SPGAL, EE-SPFAL, WCS-QuAL, and proposed logic 2. It also shows the standard deviation (σ) of average energy dissipated by AND/NAND, OR/NOR and XOR/XNOR at all the simulated frequencies. It can be seen that 2-input gates using proposed logic 2 and WCS-QuAL dissipate approximately same energy at all simulated frequencies and thus, show the least value of standard deviation in comparison to SPGAL and EE-SPFAL.

This offers an additional level of protection by ensuring, as far as possible, that all the gates use the same energy; thereby making it difficult to infer what logic operation is being performed at any one time. In other words, “gate-function-independence” as well as “data-independence” is achieved.

Table 7.2: Comparison of the standard deviation of average energy dissipated by 2-input gates.

Logic Gates	1 MHz				10MHz				100MHz			
	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2
AND/NAND E_{av} (fJ)	5.740	5.772	5.837	5.869	6.053	6.170	6.438	6.459	9.602	9.787	10.674	10.690
OR/NOR E_{av} (fJ)	4.275	5.028	5.838	5.868	5.116	5.506	6.439	6.458	8.027	8.648	10.674	10.692
XOR/XNOR E_{av} (fJ)	3.328	3.529	5.840	5.870	3.908	4.138	6.440	6.460	7.390	8.027	10.676	10.691
$E_{av,gates}$(fJ)	4.617	4.776	5.838	5.869	5.025	5.271	6.439	6.459	8.339	8.820	10.674	10.691
σ (fJ)	1.214	1.142	0.001	0.001	1.075	1.036	0.001	0.001	1.138	0.892	0.001	0.001

7.3.3 Impact of Load Variations on Energy Dissipation

Figure 7.11 shows the effect of loading on average energy consumption of AND/NAND gate using WCS-QuAL, SPGAL, EE-SPFAL and proposed logic 2 at 10MHz at load capacitance of 10fF, 100fF, 200fF, and 300fF. WCS-QuAL and proposed logic 2 use more number of transistors compared to SPGAL and EE-SPFAL due to the use of dual duplicate evaluation networks and thus have large internal load capacitance and dissipate more energy.

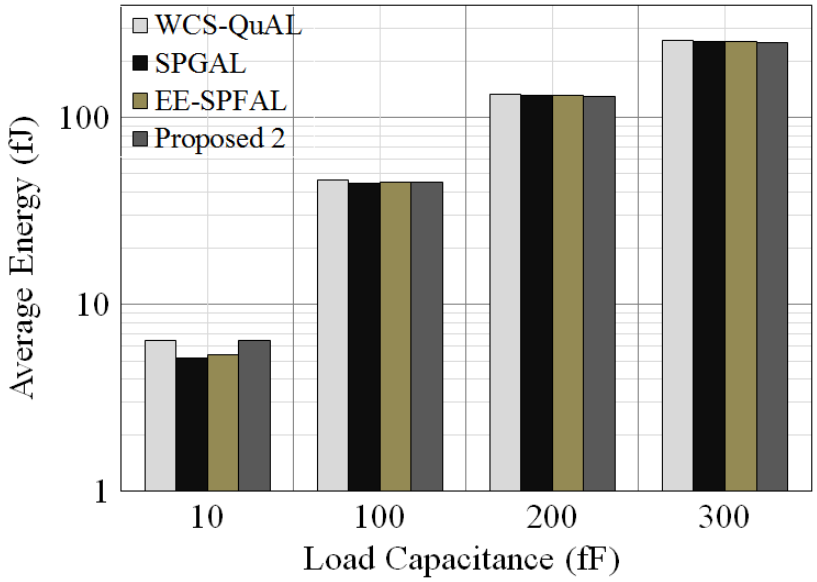


Figure 7.11: Average energy vs load capacitance for AND/NAND gate.

However, the energy dissipated by proposed logic 2 and WCS-QuAL approaches approximately to that of SPGAL and EE-SPFAL at load capacitance values higher than 100fF as can be seen from Figure 7.11. This is because, at lower values of load capacitances, the load at the output nodes of proposed logic 2 and WCS-QuAL will mainly be dominated by their internal load capacitance as they have more transistors than SPGAL and EE-SPFAL. Contrary to this, as the load capacitance value is increased, the effective load at the output nodes is dominated by the load capacitance rather than their internal load.

7.3.4 Case Study: 8-bit Montgomery Multiplier

To evaluate the performance of proposed logic 2 an 8-bit Montgomery multiplier was implemented. For comparison, WCS-QuAL, SPGAL, and EE-SPFAL logic versions were

also implemented. 8-bit Montgomery multipliers were implemented using Systolic Array Architecture 1 (SAA1) discussed in Chapter 5 of this thesis.

7.3.4.1 Impact of Frequency on NED, NSD, and Average Energy

Simulations for the Montgomery multiplier were performed at 1MHz, 13.56MHz and 100MHz frequencies. The energy dissipation was measured per cycle for 10 random input patterns. The simulation results are summarized in Table 7.3.

From Table 7.3, proposed logic 2 exhibits the least value of the %NED and %NSD for all the simulated frequencies followed by WCS-QuAL, EE-SPFAL, and SPGAL. Also, SPGAL failed to deliver the correct functionality at 1MHz due to the severe coupling effect which is caused by the absence of cross-coupled nMOS transistors. Because of the coupling effect, the output node which should remain at logic '0' gets coupled to the other output node following the power-clock. At low frequency (1MHz) the evaluating output node slowly follows the power-clock and therefore the coupled node gets enough time to follow the evaluating output node. As a result, the coupled output node reaches approximately 1.5V. This causes failure of functionality in SPGAL. At 13.56 MHz, due to the coupling effect, the coupled output node reaches approximately at 0.6V. At 100MHz, the coupled output node reaches approximately at 0.2V. This is the reason why 8-bit Montgomery multiplier using SPGAL dissipates more energy at 13.56 MHz in comparison to energy dissipated at 100MHz as can be seen from Table 7.3.

Table 7.3: Simulation results comparing the %NED and %NSD of 8-bit Montgomery Multiplier using proposed logic 2, WCS-QuAL, SPGAL, and EE-SPFAL.

		1 MHz				13.56 MHz				100 MHz			
		SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2	SPGAL	EE-SPFAL	WCS-QuAL	Proposed 2
Eav (pJ)			3.154	5.073	5.232	5.217	3.414	5.200	5.458	4.080	4.117	6.661	6.981
%NED	X		5.684	0.725	0.122	14.00	4.936	0.749	0.096	6.062	3.610	0.673	0.190
%NSD			2.416	0.205	0.060	7.768	2.400	0.254	0.039	3.051	1.878	0.189	0.091

7.3.4.2 Impact of Power Supply Scaling

Supply voltage is one of the dominant components of the energy dissipation in adiabatic logic. Energy can be reduced if the power supply is reduced. Therefore, it is important to

investigate the impact of power-clock scaling on the performance of the secure adiabatic logic designs.

The power-clock was scaled from 1.8V down to 0.6V. The simulation results of the power-supply/clock scaling at 13.56 MHz and 10fF load capacitance for 10 random inputs are summarized in Table 7.4. Since the simulation results for 1.8V power supply were included in Table 7.3, they are omitted in Table 7.4. It can be seen that the proposed logic 2 exhibits the lowest %NED and %NSD followed by WCS-QuAL, EE-SPFAL, and SPGAL at all power-clock values.

Table 7.4: Comparison of energy performance of Montgomery multiplier against power supply scaling

Logic Designs	**Power-clock scaling @ 13.56MHz**				
	V=0.6	**V=0.8**	**V=1.0**	**V=1.2**	**V=1.5**
SPGAL					
E_{av}(pJ)	0.864	1.007	1.297	1.770	2.807
%NED	1.303	3.449	4.260	5.531	8.625
%NSD	0.586	1.490	1.737	2.505	4.447
EE-SPFAL					
E_{av}(pJ)	0.997	1.049	1.303	1.680	2.475
%NED	0.689	1.819	2.976	3.612	4.182
%NSD	0.293	0.883	1.177	1.693	1.535
WCS-QuAL					
E_{av} (pJ)	1.297	1.617	2.016	2.637	3.681
%NED	0.643	0.678	0.691	0.793	0.622
%NSD	0.196	0.309	0.304	0.373	0.329
Proposed 2					
E_{av}(pJ)	1.568	1.746	2.131	2.723	3.953
%NED	0.114	0.042	0.093	0.058	0.156
%NSD	0.053	0.018	0.410	0.028	0.070

7.4 Chapter Summary

In this chapter, the problem of negative peak current variations in WCS-QuAL is discussed and the condition which triggers the variations is identified. As a solution, another novel PAA resilient adiabatic logic (proposed logic 2) is proposed which removes the condition of unbalanced output nodes during the Idle phase of the power-clock. The proposed logic 2

has symmetric structure and charge the same capacitance at the two output nodes for each input combinations. The proposed logic 2 is compared with WCS-QuAL, and two recently proposed secure adiabatic logic designs known as, SPGAL and EE-SPFAL. Simulations are performed to evaluate and compare the performance of the proposed logic 2, WCS-QuAL, SPGAL and EE-SPFAL. In particular, the impact of frequency variations on the %NED and %NSD of the single input and 2-input gates and impact of load on the average energy dissipated by the proposed logic 2, WCS-QuAL, SPGAL and EE-SPFAL are investigated. Simulation results show that proposed logic 2 and WCS-QuAL outperforms SPGAL and EE-SPFAL at all simulated frequencies. In addition, all the 2-input gates using proposed logic 2 and WCS-QuAL consume nearly equal energy thus, exhibiting data-independence and gate-function-independence. Also, proposed logic 2 exhibits the least variations in current peaks in comparison to WCS-QuAL, SPGAL, and EE-SPFAL.

To evaluate and compare the performance of proposed logic 2, WCS-QuAL, SPGAL and EE-SPFAL in a complex circuit, 8-bit Montgomery multiplier is implemented and the impact of frequency variations and power-clock scaling on the %NED and %NSD of the Montgomery multiplier is investigated. Simulation results show that the Montgomery multiplier using proposed logic 2 and WCS-QuAL exhibits the least (i.e. best) value of %NED and %NSD under frequency variations and power supply scaling.

8. Conclusion and Future work

This Chapter summarises the achievements of this research work and provides the author's recommendations for further work relating to the adiabatic approach.

8.1 Conclusions

The main motivation of this thesis was to exploit the energy efficient traits of the adiabatic logic technique for the ultra-low power operation in power-limited smartcards. The work reported in the thesis proposes several design rules for the energy efficient power-clock generation for adiabatic circuits.

It also proposes architectures suitable for energy efficient adiabatic implementation of cryptographic algorithms and a solution to reduce the area and energy overhead due to the synchronization buffers.

A problem of floating output nodes due to the application of power-clock gating in the cascade stages of adiabatic gates is identified. Due to the problem, the energy benefits that would otherwise be obtained by the application of power-clock gating were flattened. To remove this problem, a solution is proposed.

For the secure (PAA resilient) implementation of the cryptographic algorithms using adiabatic logic, shortcomings of the existing logic families were identified and as a solution, two novel PAA resilient adiabatic logic families were proposed.

The main contributions of the thesis on a chapter-by-chapter basis can be summarised as follows:

Chapter 1 is devoted to set the scene for the research problem together with novel contributions to the state-of-art by the author.

Chapter 2 gives the general background of adiabatic technique, its power-clocking schemes, and loss mechanism. It also reviews several existing adiabatic logic families followed by the discussion of the selected adiabatic logic families and the design challenges associated with the adiabatic logic.

In Chapter 3, stepwise charging strategies (2, 3, 4, 5, 6, 7, and 8-step) based on tank-capacitor circuits are compared in terms of their energy recovery properties and complexity. It is identified that the energy recovery achievable in step charging circuits depends on the tank-capacitor size and that can be reduced as the number of steps in a step charging circuit increases concluding that combined tank capacitance (CTT) versus load capacitance (C_L) ratio is the significant parameter. Tradeoffs can be made on the basis of total tank-capacitance to load capacitor (CTT/C_L) ratios. Suitable tradeoffs have been suggested -specifically that a CTT/C_L ratio of 10 with a 4-step charging circuit is appropriate, increasing either parameter yielding relatively little benefit.

Chapter 4 looks at the implementation of FSM controller for single channel and 4-phase PCG using n-step charging circuits. FSM controller for single phase PCG using 2, 3, 4, 5, 6, 7 and 8-step charging circuits and 4-phase PCG using 2, 3, and 4-step charging circuits are compared based on area, circuit complexity, and energy dissipation. The impact of step charging circuit switch sizes and supply voltage scaling on the energy dissipation of the FSM controller for single and 4-phase PCG using 2, 3, and the 4-step charging circuit is investigated. It is proposed that size of the switches in the step charging circuits should be just enough to provide sufficient time for charging and discharging of the load capacitor. It is also proposed that single and 4-phase PCG using 3 and 4-step charging circuit seems promising in comparison to the PCG with a larger number of steps due to the increased circuit complexity and energy dissipation of the FSM controller.

Chapter 5 looks at the adiabatic implementations of Montgomery multiplication algorithm. In particular, systolic array and iterative approach architectures are considered. A technique for the reduction of overhead due to synchronization buffers in the systolic array architecture is proposed. Due to the solution, the area (equivalent of), throughput, and energy are improved by 414 transistor counts, 3.5 power-clock cycles, 25.8% respectively. Three scalable, area and energy efficient iterative approach architectures using single, two and three adder stage in the datapath unit are proposed along with their methodology and the optimum number of adder stages in the datapath unit for 8-bit Montgomery multiplier

is investigated. Also, the Montgomery modular multiplication algorithm is modified. Amongst the three iterative approach architectures, MMM_IA3 is the most energy efficient.

In addition, a problem due to the application of power-clock gating in cascade stages of adiabatic gates is identified and a solution is proposed. Power-clock gating is applied in MMM_IA3 and a reduction of approximately, 24% is obtained in comparison to the energy dissipation of MMM_IA3. Using the proposed solution (MMM_IA3_PPG) a reduction of about 34% is obtained compared to the energy dissipation of MMM_IA3. In comparison to systolic array architectures, iterative approach architectures are more energy efficient. Systolic array architecture presents a large load to the step charging circuits (SCC) and therefore, the energy dissipation of the SCC dominates that of synchronous FSM controller for 2 & 3-step charging circuits. Contrary to this, in iterative approach architecture, the energy dissipation of the FSM controller dominates that of SCC in 3 and 4-step charging circuits. This suggests that in a large adiabatic system, the losses due to the FSM controller will be negligible compared to the overall losses.

Chapter 6 introduces the general background of Power-Analysis Attack (PAA) resilient logic designs, followed by the reviews of currently known secure adiabatic logic families with an emphasis on their shortcomings. As a solution to the shortcomings, a novel PAA resilient adiabatic logic family, WCS-QuAL is proposed. WCS-QuAL has a symmetric structure, removes NAL from the evaluation phase of the power-clock and obviate the need for the control signal for charge sharing transistors. It is shown that WCS-QuAL outperforms (based on NED and NSD) the existing secure adiabatic logic designs at all process corners at all simulated frequencies and shows the least sensitivity to the process corners. In addition, all the 2-input gates using WCS-QuAL consume nearly equal energy thus, exhibiting data-independence and gate-function independence. Also, the Montgomery multiplier using WCS-QuAL exhibits the least (i.e. best) value of NED and NSD against frequency variations and power supply scaling and dissipates the lowest energy at frequencies ranging from 20MHz to 100MHz.

Chapter 7 looks into the problem of variations in negative current peaks in the secure adiabatic logic, WCS-QuAL proposed in Chapter 6. The duration for which the two output nodes of WCS-QuAL remain unbalanced and thus exhibit variations in the negative peak currents is identified. As a solution, another novel PAA resilient adiabatic logic family

using single charge sharing transistor (proposed logic 2) is proposed which removes the condition of unbalanced output nodes during the idle phase of the power-clock. It also has the symmetric structure, It is shown that the proposed logic 2 outperforms (based on NED and NSD) the existing secure adiabatic logic designs at all simulated frequencies and exhibits least variations in the current peaks. In addition, all the 2-input gates using proposed logic 2 consume nearly equal energy thus, exhibiting data-independence and gate-function independence. Also, the Montgomery multiplier using proposed logic-2 exhibits the least (i.e. best) value of %NED and %NSD against frequency variations and power supply scaling in comparison to the existing logic families.

8.2 Future Work

Taking the other unsolved problems into consideration, the author would like to make the following recommendations for the future work.

The work reported in this thesis leads to solving many associated problems of the design of adiabatic systems, but few others require further investigations.

Synchronous FSM controller for generating the control signals for switches of the step charging circuit dissipates high energy, degrading the overall energy efficiency of the adiabatic system. To reduce the energy dissipation, design of asynchronous FSM controller for 4-phase PCG using n-step charging circuits is worthy of investigation.

Energy dissipation of the power-clock generator can further be reduced by increasing the throughput of the adiabatic system. Therefore, architectural techniques which can offer high throughput in the adiabatic implementations are worth investigating.

From the literature review, adiabatic logic seems to be energy efficient countermeasure against PAA. The two secure adiabatic logic families proposed in Chapter 6 and 7 outperformed the existing secure adiabatic logic families based on %NED and %NSD. However, the practicality of the two proposed secure adiabatic logic families against PAA needs to be investigated by performing the DPA attacks. Also, the impact of the power-clock generator (step charging power-clock) on the robustness of the proposed logic families against PAA is worthy of investigation.

The next obvious step to follow from the exciting and promising results obtained through

the extensive simulation based studies carried out in this thesis, would be to create custom layouts and commit to Silicon some candidate circuits to undertake Silicon based measurements and compare the results to those obtained from the simulation studies, demonstrating the approach and its validity with real Silicon sample circuits, as well as possibly highlighting any possible problems that might have been overlooked or were not feasibly covered in the simulation campaigns.

References

[1] Claude E. Shannon, “A Mathematical Theory of Communication”, *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423,623–656,1948.

[2] R. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems”, *Communications of the ACM*, vol. 21, pp. 120–126, 1978.

[3] N. Koblitz, “Elliptic curve cryptosystems”, *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[4] P. L. Montgomery, “Modular multiplication without trial division”, *Mathematics of Computation*, vol. 44, no. 170, pp. 519–521, 1985.

[5] W. C. Athas, L. J. Svesson, J. G. Koller, N. Traztzanis and E. Y.-C. Chuo, "Low power digital system based on adiabatic-switching principles", *IEEE Transactions on VLSI Systems*, vol. 2, no. 4, pp. 398-406, 1994.

[6] S. G. Younis and T. F. Knight, “Practical Implementation of Charge Recovering Asymptotically Zero Power CMOS”, *Symposium on Research on Integrated Systems*, USA, 1993, pp. 234-250.

[7] P. C. Kocher, J. Jaffe, and B. Jun, ‘‘Differential Power Analysis’’, *19th Annual International Cryptology Conference Advances in Cryptology (CRYPTO 99)*, LNCS 1666, Springer-Verlag, 1999, pp. 388-397.

[8] A. G. Dickinson and J. S. Denker, “Adiabatic Dynamic Logic”, *IEEE Custom Integrated Circuits Conference (CICC’94)*, 1994, pp. 282-285.

[9] B. Kong, J. Choi, S. Lee and K. Lee, “Charge Recycling Differential Logic for Low-Power Application”, *43rd IEEE International Solid State Circuits Conference – Digest of Technical Papers (ISSCC’96)*, 1996, pp. 302-303.

[10] A. Kramer, J. S. Denker, S. C. Avery, A. G. Dickinson and T. R. Wik, "Adiabatic Computing with the 2n-2n2d Logic Family", *Symposium on VLSI Circuits – Digest of Technical Papers*, 1994, pp. 25-26.

[11] J. G. Koller and W. C. Athas, "Adiabatic Switching, Low Energy Computing, and the Physics of Storing and Erasing Information", *2nd Workshop on Physics and Computation*, 1992, pp. 267-270.

[12] R. T. Hinman and M. F. Schlecht, "Recovered Energy Logic – A Highly Efficient Alternative to Today's Logic Circuits", *24th Annual IEEE Power Electronics Specialists Conference (PESC'93)*, 1993, pp. 17-26.

[13] W. C. Athas, N. Tzartzanis, L. J. Svensson, L. Peterson, H. Li, P. Wang and W.-C. Liu, "AC-1: a clock-powered microprocessor", *International Symposium on Low Power Electronics and Design (ISLPED'97)*, 1997, pp. 328-333.

[14] V. K. De and J. D. Meindl, "A dynamic energy recycling logic family for ultra-low-power gigascale integration (GSI)", *International Symposium on Low Power Electronics and Design (ISLPED'96)*, 1996, pp. 371-375.

[15] S. G. Younis and T. F. Knight, "Asymptotically Zero Energy Computing Split-Level Charge Recovery Logic", *International Workshop on Low Power Design*, 1994, pp.177-182.

[16] S. G. Younis, "Asymptotically Zero Energy Computing Using Split-Level Charge Recovery Logic", Ph.D. Thesis, Massachusetts Institute of Technology, June 1994.

[17] A. Kramer, J. S. Denker, B. Flower and J. Moroney, "2nd Order adiabatic computation with 2n-2p and 2n-2n2p logic circuits", *IEEE Symposium Low Power Design (ISLPD'95)*, 1995, pp. 191-196.

[18] J. S. Denker, "A review of adiabatic computing", *IEEE Symposium on Low Power Electronics – Digest of Technical Papers (ISLPE'94)*, 1994, pp. 94-97.

[19] F. Liu and K. T. Lau, "Improved structure for efficient charge recovery logic", *Electronics Letters*, Vol. 34, no. 18, pp.1731-1732, 1998.

[20] D. Maksimović and V. G. Oklobdžija, "Integrated Power Clock Generators for Low Energy Logic", *26th Annual IEEE Power Electronics Specialists Conference (PESC'95)*, 1995, pp. 61-67.

[21] A. Vetuli, S. D. Pascoli, and L. M. Reyneri, “Positive feedback in adiabatic logic”, *Electronics Letters*, vol. 32, no. 20, pp. 1867-1869, 1996.

[22] F. Liu and K. T. Lau, “Pass-transistor adiabatic logic with nMOS pull-down configuration”, *Electronics Letters*, vol. 34, no. 8, pp. 739-741, 1998.

[23] C. C. Yeh, J. H. Lou, and J. B. Kuo, “1.5V CMOS full-swing energy efficient logic (EEL) circuit suitable for low-voltage and low-power VLSI applications”, *Electronics Letters*, vol. 33, no. 16, pp. 1375-1376,1997.

[24] V. G. Oklobdžija, D. Maksimović and F. Lin, “Pass-Transistor Adiabatic Logic Using Single Power-Clock Supply”, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 10, pp. 842-846, 1997.

[25] J. Lim, K. Kwon, and S. Chae, “Reversible energy recovery logic circuit without non-adiabatic energy loss”, *Electronics Letters*, vol. 34, no. 4, pp. 344-346, 1998.

[26] L. Varga, F. Kovács, and G. Hosszú, “An Efficient Adiabatic Charge-Recovery Logic”, *IEEE SouthEastCon*, 2001, pp. 17-20.

[27] H. Jianping, C. Lizhang, and L. Xiao, “A new type of low-power adiabatic circuit with complementary pass-transistor logic”, *5th International Conference on ASIC (ASIC’03)*, 2003, pp. 1235-1238.

[28] E. Amirante, A. Bargagli-Stoffi, J. Fischer, G. Iannaccone, D. Schmitt-Landsiedel,.: “Variations of the Power Dissipation in Adiabatic Logic Gates”, *11th International Workshop on Power And Timing Modeling, Optimization and Simulation, (PATMOS’01)*, 2001, pp. 9.1.1–9.1.10.

[29] C. Seitz, A. Frey, S. Mattison, S. Rabin, D. Speck, and J. Van De Snepscheut, “Hot-clock nMOS,” *Proceeding of Chapel Hill Conference VLSI*, 1985, pp. 1–17.

[30] W. C. Athas, L. Svensson, J. G. Koller, N. Tzartzanis, and E. Chou, “A framework for practically low-power digital CMOS systems using adiabatic switching principles”, *International Workshop on Low Power Design*, 1994, pp.1-6.

[31] Y. Moon and D. Jeong, “Efficient Charge Recovery Logic”, *Symposium on VLSI Circuits – Digest of Technical Papers*, 1995, pp. 129-130.

[32] Yong Moon and Deog-Kyoon Jeong, “An efficient charge recovery logic circuit”, *IEEE Journal of Solid-State Circuits*, vol. 31, no. 4, pp.514-522, 1996.

[33] S. Maheshwari, V.A.Bartlett and I. Kale "Adiabatic Flip-Flops and Sequential Circuit Designs using Novel Resettable Adiabatic Buffers", *23rd European Conference on Circuit Theory and Design*, 2017, pp. 4-6.

[34] M. C. Knapp, P. J. Kindlmann, and M. C. Papaefthymiou, "Implementing and evaluating adiabatic arithmetic units", *IEEE Custom Integrated Circuits Conference*, 1996, pp. 115–118.

[35] M. C. Knapp, P. J. Kindlmann, and M. C. Papaefthymiou, "Design and evaluation of adiabatic arithmetic units", *Analog Integrated Circuits Signal Process*, vol. 14, pp. 71–79, 1997.

[36] P. Teichmann, J. Fischer, F. Chouard, and D. Schmitt-Landsiedel "Design issues of arithmetic structures in adiabatic logic", *Advances in Radio Science*, vol. 5, pp. 291-295, 2007.

[37] P. Teichmann. Adiabatic logic: future trend and system level perspective, vol. 34. Springer Science & Business Media, 2011.

[38] P. Teichmann, J. Fischer, F. Chouard, and D. Schmitt-Landsiedel, "Design of ultra-low-power arithmetic structures in adiabatic logic", *International Symposium on Integrated Circuits 2007 (ISIC 2007)*, 2007, pp. 405–408.

[39] P. Teichmann, J. Fischer, S. Henzler, E. Amirante, and D. Schmitt-Landsiedel, "Power-Clock gating in adiabatic logic circuits", *Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS'05)*, 2005, pp. 638-646.

[40] P. Teichmann, J. Fischer, S. Henzler, E. Amirante, and D. Schmitt-Landsiedel, "Power-Clock gating in adiabatic logic circuits", *Integrated Circuits and System Design, Lecture Notes in Computer Science, Springer*, 2005, pp.638-646.

[41] P. Teichmann, J. Fischer, S. Henzler, E. Amirante, and D. Schmitt-Landsiedel, "Power-Clock gating in adiabatic logic circuits", *Advances in Radio Science*, vol. 6, pp. 257-280, 2006.

[42] H. Jianping, D. Zhou, and L. Wang, "Power-gating adiabatic flip flops and sequential logic circuits", *IEEE International conference on Communications, Circuits and Systems ICCCAS'07*, 2007, pp. 1016-1020.

[43] Y. Moon, and D.K. Jeong, "A 32 x 32-bit adiabatic register file with supply clock generator", *IEEE Journal of Solid-state Circuits*, vol. 33, no. 5, pp. 696-701, 1998.

[44] L. J. Svensson, and J. G. Koller, “Driving a capacitive load without dissipating fCV2”, *IEEE Symposium on low power electronics*, 1994, pp. 100-101.

[45] A. Blotti, S. Borghese, and R. Saletti, “Single-inductor four-phase power-clock generator for positive-feedback adiabatic logic gates”, *International Conference on Electronics, Circuits,and System*, 2002, pp. 533-536.

[46] H. Mahmoodi-Meimand, and A. Afzali-Kusha, “Efficient power clock generation for adiabatic logic”, *IEEE International Symposium on Circuits and Systems*, 2001, pp. 642-645.

[47] M. Arsalan, and M. Shams, “Charge-recovery power clock generators for adiabatic logic circuits”, *International Conference on VLSI Design*, 2005, pp. 1 -174.

[48] S. Nakata, et al. “Adiabatic SRAM with a shared access port using a controlled ground line and step-voltage circuit”, *IEEE International Symposium on Circuits and Systems*, 2010, pp. 2474–2477.

[49] S. Nakata, S. Mutoh, H. Makino, M. Miyama, Y. Matsuda, “Stable adiabatic circuit using advanced series capacitors and time variation of energy dissipation”, *IEICE Electron. Express*, vol. 7, no. 9, pp. 640–646, 2010.

[50] S. Nakata, T. Kusumoto, M. Miyama., Y. Matsuda, “Adiabatic SRAM with a large margin of variation by controlling the cellpower-line and word-line voltage”, *IEEE International Symposium on Circuits and Systems*, 2009, pp. 393–396.

[51] S. Nakata, H. Makino, Y. Matsuda, “A new stepwise adiabatic charging circuit with a smaller capacitance in a regenerator than a load capacitance”, *IEEE 57th International Midwest Symposium on Circuits and Systems*, 2014, pp. 439-442.

[52] S. Nakata, et al., “General stability of stepwise waveform of an adiabatic charge recycling circuit with any circuit topology”, *IEEE Transactions on Circuits and Systems I*, vol. 59, no. 10, pp. 2301-2314, 2012.

[53] S. Nakata, R. Honda, H. Makino, H. Morimura, Y. Matsuda, “Energy dissipation reduction during adiabatic charging and discharging with controlled inductor current”, *IEEE 55th International Midwest Symposium on Circuits and Systems*, 2012, pp. 1068-1071.

[54] S. Nakata, H. Makino, S. Mutoh, M. Miyama, Y. Matsuda, “Energy Dissipation Decrease during Adiabatic Charging of a Capacitor by Changing the Duty Ratio”, *IEEE 54th International Midwest Symposium on Circuits and Systems*, 2011, pp. 1-4.

[55] W. Lin, J. Ye, and M. Shieh, “Scalable Montgomery modular multiplication architecture with low latency and low memory bandwidth requirement”, *IEEE Transactions on Computers*, vol. 63, no. 2, pp. 475-483, 2014.

[56] S. H. Wang, W.-C. Lin, J.-H. Ye, and M.-D. Shieh,“ Fast scalable radix-4 Montgomery modular multiplier”, *IEEE International Symposium on Circuits and Systems*, 2012, pp. 3049–3052.

[57] T. Wu, “Improving radix-4 feed-forward scalable Montgomery modular multiplier by pre-computation and double Booth-encodings”, *IEEE International Conference on Computer Science and Network Technology (ICCSNT)*, 2013, pp. 596–600.

[58] M. D. Shieh and W. C. Lin, “Word-based Montgomery modular multiplication algorithm for low-latency scalable architectures”, *IEEE Transactions on Computers*, vol. 59, no. 8, pp. 1145–1151, 2010.

[59] S. R. Kuang, J.-P. Wang, K. C. Chang, and H.-W. Hsu, “Energy-efficient high-throughput Montgomery modular multipliers for RSA cryptosystems”, *IEEE Transansactions on VLSI Systems*, vol. 21, no. 11, pp. 1999–2009, 2013.

[60] M. Huang, K. Gaj, and T. El-Ghazawi, "New hardware architectures for Montgomery modular multiplication algorithm", *IEEE Transactions on Computers*, vol. 60, no. 7, pp. 923-936, 2011.

[61] A. Tenca and C. Koc¸, “A scalable architecture for modular multiplication based on montgomery’s algorithm”, *IEEE Transactions On Computers*, vol. 52, no. 9, pp. 1215–1221, 2003.

[62] H-K Son and S-G Oh., “Design and implementation of scalable low-power Montgomery multiplier”, *IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD)*, 2004, pp. 524 – 531.

[63] A. Ibrahim, H. Elsimary, A. Nassar, “Design and implementation of scalable low power radix-4 Montgomery modular multiplier”, *International Conference on Computer Engineering & Systems (ICCES)*, 2007, pp. 395 – 400.

[64] A. Ibrahim, F. Gebali, H. Elsimary, A. Nassar, “New processor array architecture for scalable radix 8 montgomery modular multiplication algorithm”, *24th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2011, pp. 389 – 394.

[65] A. Ibrahim, F. Gebali, H.Elsimary, A. Nassar, “Processor Array Architectures for Scalable Radix 4 Montgomery Modular Multiplication Algorithm”, *IEEE Transactions on Parallel and Distributed Systems*, vol: 22, no. 7, pp. 1142 – 1149, 2011.

[66] D. M. Wang, Y. Y. Ding, J. G. Hu, H. Z. Tan, “Low power hardware design for Montgomery modular multiplication”, *IET International Conference on Information and Communications Technologies (IETICT)*, 2013, pp. 124–128.

[67] X. Wang, P. Noel, T. Kwasniewski, “Low power design techniques for a Montgomery modular multiplier”, *International Symposium on Intelligent Signal Processing and Communication Systems*, 2005, pp. 449 – 452.

[68] P. Teichmann, J. Fischer, D. Schmitt-Landsiedel, "A robust synchronized 2N2P LC oscillator with a shut-down mode for adiabatic logic circuits”, *International Symposium on Circuits and Systems (ISCAS)*, 2009, pp. 241-244.

[69] P. Kocher, “Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems”, *Advances in Cryptology: Proceedings of CRYPTO’96*, Lecture Notes in Computer Science, Springer-Verlag, 1996, pp. 104–113.

[70] K. Gandolfi, C. Mourtel, and F. Olivier, “Electromagnetic analysis: Concrete results”, 3rd *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 255–265.

[71] J.-J. Quisquater and D. Samyde, “Electromagnetic analysis (EMA): Measures and counter-measures for smard cards”, *International Conference on Research in Smart Cards: Smart Card Programming and Security (E-smart)*, Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 200–210.

[72] F. Mac´e, F.-X. Standaert, J.-J. Quisquater, “Information Theoretic Evaluation of Side-Channel Resistant Logic Styles”, *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, Lecture Notes in Computer Science, Springer-Verlag, 2007, pp. 427-442.

[73] T. Popp, S. Mangard, E. Oswald, “Power Analysis Attacks and Countermeasures”, *IEEE Design & Test of Computers*, vol. 2, no. 6, pp. 535–543, 2007.

[74] S. Mangard, E. Oswald, and T. Popp, Power Analysis Attacks, Revealing the Secrets of Smart Cards. Springer, 2007.

[75] T. S. Messerges, E. A. Dabbish, and R. H. Sloan “Investigations of power analysis attacks on smartcards”, *USENIX Workshop on Smartcard Technology*, 1999, pp. 151–161.

[76] K. Tiri, M. Akmal, and I. Verbauwhede, “A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards”, *28th European Solid-State Circuits Conference (ESSCIRC '02)*, 2002, pp. 403-406.

[77] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, “Examining smart-card security under the threat of power analysis attacks”, *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 541–552, 2002.

[78] H. Saputra, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, R. Brooks, S. Kim,and W. Zhang, “Masking the energy behavior of DES encryption”, *Design, Automation and Test in Europe Conference*, 2003, pp. 84-89.

[79] J. D. Golic and R. Menicocci, “Universal Masking on Logic Gate Level”, *Electronics Letters*, vol. 40, no. 9, pp. 526-528, 2004.

[80] K. Tiri and I. Verbauwhede, “A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation”, *Design, Automation and Test in Europe Conference and Exposition*, 2004, pp. 246-251.

[81] T. Popp and S. Mangard, “Masked Dual-Rail Pre-Charge Logic: DPA-Resistance without Routing Constraints”, *Workshop on Cryptographic Hardware and Embedded Systems (CHES ’05) Lecture Notes in Computer Science*, vol. 3659, pp. 172-186, 2005.

[82] A. Moradi, M. Kirschbaum; T. Eisenbarth, C. Paar, “Masked Dual-Rail Precharge Logic Encounters State-of-the-Art Power Analysis Methods”, *IEEE Transactions On VLSI Systems*, vol. 20, no. 9, pp.1578-1589, 2013.

[83] Z. Chen and Y. Zhou, "Dual-rail random switching logic: A countermeasure to reduce side channel leakage", *Lecture Notes in Computer Science*, Springer, vol. 4249, pp. 242-254, 2006.

[84] M. Bucci, L. Giancane, R.o Luzzi, and A. Trifiletti, "Three-Phase Dual-Rail Precharge Logic", *Cryptographic Hardware and Embedded Systems (CHES 2006), Lecture Notes in Computer Science*, vol. 4249, pp. 232-241, 2006.

[85] M. Bucci, L. Giancane, R. Luzzi, G. Scotti, and A. Triletti, "Delaybased dual-rail precharge logic", *IEEE Transaction on Very Large Scale (VLSI) System*, vol. 19, no. 7, pp. 1147–1153, 2011.

[86] M. A. Morrison, N. Ranganathan, J. Ligatti, "Design of Adiabatic Dynamic Differential Logic for DPA-Resistant Secure Integrated Circuits", *IEEE Transactions On VLSI Systems*, vol. 23, no. 8, pp.1381-1389, 2015.

[87] C. Monteiro, Y. Takahashi, and T. Sekine, "DPA Resistance of charge sharing symmetric adiabatic logic", *International Symposium on Circuits and Systems (ISCAS)*, 2013, pp. 2581–2584.

[88] C. Monteiro, Y. Takahashi, T .Sekine, "Resistance against power analysis attacks on adiabatic dynamic and adiabatic differential logic designs for smartcard", *IEEE Intelligent Signal Processing and Communication System (ISPACS'11)*, 2011, pp.1-5.

[89] C. Monteiro, Y. Takahashi, and T. Sekine, "Robust secure charge-sharing symmetric adiabatic logic against side-channel attacks", *36th International Conference on Telecommunications and Signal Processing (TSP)*, 2013, pp. 732–736.

[90] C. Monteiro, Y. Takahashi, and T. Sekine, "Low-power secure S-box circuit using charge-sharing symmetric adiabatic logic for advanced encryption standard hardware design", *IET Circuits, Devices & Systems*, vol 9, no. 5, pp. 362–369, 2015.

[91] B. D. Choi, K. E. Kim, K. S. Chung, and D. K. Kim, "Symmetric adiabatic logic circuits against differential power analysis", *ETRI Journal*, vol. 32, no. 1, pp. 166–168, 2010.

[92] M. Avital, H. Dagan, I. Levi, O. Keren, A. Fish, "DPA-Secure Quasi-Adiabatic Logic (SQAL) for Low-Power Passive RFID Tags Employing S-Boxes", *IEEE Transactions on Circuits and Systems*, vol. 62, no. 1, pp. 149 – 156, 2015.

[93] E. Tena-Sanchez, J. Castro, and A. J. Acosta, ”A Methodology for ´ Optimized Design of Secure Differential Logic Gates for DPA Resistant Circuits,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 4, no. 2, pp. 203-215, 2014.

[94] D. Suzuki and M. Saeki, “Security evaluation of DPA countermeasures using dual-rail pre-charge logic style,” *Workshop on Cryptographic Hardware and Embedded Systems*, 2006, pp. 255–269.

[95] K. Tiri, “Side-channel attack pitfalls,” *ACM/IEEE DAC*, 2007, pp. 15-20.

[96] S. Mangard, T. Popp, and B. Gammel, “Side-Channel Leakage of Masked CMOS Gates,” *The Cryptographers' Track at the RSA Conference, Lecture Notes in Computer Science, Springer*, 2005, pp. 351–365.

[97] K. Tiri and I. Verbauwhede, “Place and route for secure standard cell design”, *Smart Card Research and Advanced Application IFIP Conference (CARDIS '04)*, 2004, pp. 143-158.

[98] L. Lin, W. Burleson., “Analysis and mitigation of process variataion impacts on Power-Attack Tolerance”, *46th ACM/IEEE Design Automation Conference*, 2009, pp. 238-243.

[99] S. Guilley, P. Hoogvorst, Y. Mathieu, R. Pacalet, “The Backend Duplication Method: A Leakage-Proof Place-and-Route Strategy for ASICs”, *Workshop on Cryptographic Hardware and Embedded Systems*. LNCS, Springer, vol. 3659, pp. 383–397, 2005.

[100] J. Fisher, E. Amirante, F. Randazzo, G. Inannaccone, D. Schmitt-Landsiedel, “Reduction of the Energy Consumption in the Adiabatic Gates by Optimal Transistor Sizing”, *Workshop on Power And Timing Modeling, Optimization and Simulation*, 2003, pp. 309–318.

[101] S. D. Kumar, H. Thapliyal, A. Mohammad, and S. K. Perumalla, “Design exploration of a Symmetric Pass Gate Adiabatic Logic for Energy-Efficient and Secure Hardware”, *Integration, the VLSI Journal, Elsevier*, vol. 58, pp. 369-377, 2016.

[102] S. D. Kumar, H. Thapliyal, A. Mohammad, V. Singh, and S. K. Perumalla, "Energy-efficient and secure s-box circuit using Symmetric Pass Gate Adiabatic Logic", *IEEE computer society Annual Symposium on VLSI (ISVLSl)*, 2016, pp. 308-313.

[103] S. D. Kumar, H. Thapliyal, A. Mohammad, "EE-SPFAL: A Novel Energy-Efficient Secure Positive Feedback Adiabatic Logic for DPA Resistant RFID and Smart Card", *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1-12, 2016.

[104] Sachin Maheshwari, V.A.Bartlett and Izzet Kale, "Adiabatic Flip-Flops and Sequential Circuit Designs using Novel Resettable Adiabatic Buffers", *European Conf. on Circuit Theory and Design (ECCTD)*, pp. 1-4, 2017.

Appendix A

VHDL Code for Modified Montgomery Multiplier

This appendix gives the VHDL implementation of modified Montgomery Multiplier that is described in Chapter 5

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_1164.ALL;
--use IEEE.STD_LOGIC_UNSIGNED.ALL;
--use ieee.std_logic_arith.all;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

ENTITY M2 is
generic(
inwidth         :       integer := 8;
outwidth        :       integer := 10
);
port(
x       :       in              std_logic_vector(inwidth-1 downto 0);
y       :       in              std_logic_vector(inwidth-1 downto 0);
m       :       in              std_logic_vector(inwidth-1 downto 0);
```

```
        s       :       out     std_logic_vector(outwidth-1 downto 0)
        );
        end M2;

        architecture rtl of M2 is
         signal xint,yint,mint : std_logic_vector(outwidth-1 downto 0);
         constant n                      : integer       := inwidth;

        begin
         -- sign extension
         xint <= (outwidth-inwidth-1 downto 0=>'0') & x;
         yint <= (outwidth-inwidth-1 downto 0=>'0') & y;
         mint <= (outwidth-inwidth-1 downto 0=>'0') & m;

         stim:process(xint,yint,mint)
          variable sint : std_logic_vector(outwidth-1 downto 0);
          variable sint2: std_logic_vector(outwidth-1 downto 0);
          variable s0     : std_logic;
         begin
                 sint := (others=>'0');
                 for i in 0 to n-1 loop
                         s0      := sint(0);
                         sint2 := '0' & sint(outwidth-1 downto 1);
                         if ((xint(i) and yint(0)) or s0)='1' then
                                 sint    :=      std_logic_vector(unsigned(sint2)      +
        unsigned((outwidth-1 downto 1=>xint(i)) and yint(outwidth-1 downto 1)) + 1);
                         else
                                 sint    :=      std_logic_vector(unsigned(sint2)      +
        unsigned((outwidth-1 downto 1=>xint(i)) and yint(outwidth-1 downto 1)) + 0);
                         end if;

                         if ((xint(i) and yint(0)) xor s0)='1' then
                                 sint    :=      std_logic_vector(unsigned(sint)       +
        unsigned(mint(outwidth-1 downto 1)));
                         else
                                 sint := std_logic_vector(unsigned(sint) + 0);
                         end if;
                 end loop;
                 s <= sint;
         end process stim;
end rtl;
```

Test bench:

```
use ieee.std_logic_arith.all;

ENTITY M2_tb IS
END M2_tb;

ARCHITECTURE behavior OF M2_tb IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT M2
   PORT(
        x : IN  std_logic_vector(7 downto 0);
        y : IN  std_logic_vector(7 downto 0);
        m : IN  std_logic_vector(7 downto 0);
        s : OUT  std_logic_vector(9 downto 0)
       );
   END COMPONENT;


  --Inputs
  signal x : std_logic_vector(7 downto 0) := (others => '0');
  signal y : std_logic_vector(7 downto 0) := (others => '0');
  signal m : std_logic_vector(7 downto 0) := (others => '0');

          --Outputs
  signal s : std_logic_vector(9 downto 0);
  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

BEGIN

          -- Instantiate the Unit Under Test (UUT)
  uut: M2 PORT MAP (
         x => x,
         y => y,
         m => m,
         s => s
        );


  -- Stimulus process
  stim_proc: process
  begin
                    x <= (others=>'0');
```

```
                y <= (others=>'0');
                m <= (others=>'0');

            wait for 10 ns;
        --          x <= "01001111";
        --          y <= "10010101";
        --          m <= "11110011";

                    x <= conv_std_logic_vector(38,8);
                    y <= conv_std_logic_vector(12,8);
                    m <= conv_std_logic_vector(243,8);

                    wait for 10 ns;
            wait;
        end process stim_proc;
END;
```