Western University
Scholarship@Western

Digitized Theses Digitized Special Collections

2008

DESIGN OF EFFICIENT NANOELECTRONIC MEMORY AND CRYPTOGRAPHIC CIRCUITS

Mrinmoy Barua

Follow this and additional works at: https://ir.lib.uwo.ca/digitizedtheses

Recommended Citation
Barua, Mrinmoy, "DESIGN OF EFFICIENT NANOELECTRONIC MEMORY AND CRYPTOGRAPHIC CIRCUITS" (2008). *Digitized Theses*. 4619.
https://ir.lib.uwo.ca/digitizedtheses/4619

This Thesis is brought to you for free and open access by the Digitized Special Collections at Scholarship@Western. It has been accepted for inclusion in Digitized Theses by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

DESIGN OF EFFICIENT NANOELECTRONIC MEMORY AND CRYPTOGRAPHIC CIRCUITS

(Spine title: Efficient Nanoelectronic Circuits)

(Thesis format: Monograph)

by

Mrinmoy Barua

Graduate Program in Engineering Science
Department of Electrical and Computer Engineering

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Engineering Science

Faculty of Graduate Studies
The University of Western Ontario
London, Ontario, Canada

© Mrinmoy Barua 2008

Abstract

This thesis presents the design of Nanoelectronic Memory cell and arrays compatible with molecular switch (nanodevice) electrical characteristics. The proposed transmission gate based CMOL (hybrid CMOS / Molecular) memory cell surmounts the operational difficulties facing previous design. The Control circuitry with improved multiplexer design is introduced in this dissertation. Yield improvement through replacing the defective cell with a free cell can be achieved using a proposed algorithm. Moreover, the proposed memory cell has the same area as the existing CMOL inverter cells allowing easier implementation of both logic and memory circuits on the same chip. An efficient hardware implementation of the SBox from the Advanced Encryption Standard (AES) is presented in this dissertation. Modification of the design was achieved by adding Tri-state Inverter followed by an Inverter (TII). Simulation results show a reduction in the average power dissipation as well as the time delays. Reduction of supply voltage and using low V_{dd} in non critical path improved the performance by reducing the energy delay product. Different transistors models with dual threshold voltage (Vt) based on 65nm CMOS technology were applied to the design to achieve further improvement.

Keywords: CMOL, CMOL circuits, Nanoscale memory, Control Circuit, AES, SBox, Deep sub-micron CMOS technology, Tri-state Inverter, Low Power SBox.

Dedications

To
My Dear Father
G. B. Barua
[September 28, 1943 – September 26, 2007]

We know it by a sure induction that all we love must someday cease to be.
Things do not remain as we left them. But this fact that "all things die" does not itself die.
You did not "passed way", You did not "expired".
I think, you just stepped into eternity.
And I doubt if you looked back.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Professor Z. E. Abid for his generous help throughout the course of my Master's. Professor Abid is an admirable academic professional. He taught me not only his precious knowledge, but his exceptional professionalism. He impressed me very much by his responsibility and strict attitude in training students. He always provided timely and warm encouragement and support in difficult times. I especially thank him for his prompt reading and careful critique of my thesis.

I am grateful to the secretaries and technicians in the department of Electrical and Computer Engineering, for help and assisting me in many different ways.

It would have been difficult for me to continue the course of graduate school without an exceptional group of friends. We had serious research discussions and pleasant parties. I really want to send my thanks and best wishes to them: Abdallah, Mehran, Adem and Reza.

My deepest gratitude goes to my family for their unflagging love and support throughout my life. I am indebted to my father, G. B. Barua, for his care, encouragement and love. Although he is no longer with us, he is forever remembered. I am sure he shares our joy and happiness in the heaven. I am at a loss of words to express my gratitude to my mother and my brother for their continuous love and support. Their encouragement and support gave me the strength to bear the immense loss of my father passed way during my Master's. I feel fortunate to have them always standing by my side.

My loving thanks to my son Anindya, my wife Tanima Barua. They are the most important source of inspiration, encouragement, and happiness.

Most of all, I wish to thank and appreciate my wife for her immense love, sacrifice, and unforgettable and uncountable support and patience. She puts my dreams ahead of hers, offered her support during the hard time of research, and did everything to uphold my spirit whenever I was down.

Contents

Certificate of Examination ii
Abstract iii
Dedications iv
Acknowledgements v
Contents vi
List of Tables ix
List of Figures xi
List of Abbreviation xiv
Introduction 1

1.1 General Introduction 1
1.2 Motivation 2
1.3 Thesis Organization 4

Nanoelectronic Memory Design 5

2.1 Introduction 5
2.2 Review of CMOL Architecture 7
2.3 Nanodevice properties 10
2.4 Limitation of MOSFET when used as Pass transistor 12
2.5 Memory based on existing CMOL cell 14
2.6 Proposed Memory Cell 16
2.7 Design of control circuit 18
2.7.1 Proposed Multiplexer: 19
2.7.2 Memory Write Operation 22
2.8 Area Analysis of CMOL memory cells: 25
2.9 Crossbar memory architecture: 26
2.10 The Need o f improved nanodevice characteristics 28
2.11 Reconfi guration Algorithm for Defect tolerant Memory design 28
2.12 I mproved Reconfiguration Algorithm 30
2.13 Future Work 36
2.14 Conclusion 37

AES: Literature Review **38**

3.1 Introduction: 38
3.2 Information Security and Cryptography 39
3.3 Symmetric-key Cryptography and AES 41
3.4 The AES Cipher 43
3.5 Overall Structure of AES 44
3.6 Definitions 47
3.6.1 Symbols, and Functions 48
3.7 Mathematical Preliminaries 48
3.7.1 Finite Fields 49
3.7.2 Addition 49
3.7.3 Multiplication 50
3.8 SubBytes Transformation 51
3.9 InvSubBytes Transformation 53
3.10 ShiftRow s Transformation 53
3.11 Mix Columns Transformation 54
3.12 I nvMixColumns Transformation 55
3.13 AddRoundKe y Transformation 56
3.14 Ke yExpansion 57

SBox Circuit Implementation **60**

4.1 Introduction: 60
4.2 Mathematical Review: 61
4.2.1 $GF(2^8)$ as an Extension of $GF(2^4)$ 61
4.2.2 Inversion of Two-Term Polynomials 64
4.2.3 Transformation between $GF(2^8)$ and $GF(2^4)$ 64
4.3 SBox Building Blocks 65
4.4 SBox Design and Simulation 68
4.4.1 Design Modification 73
4.4.2 Minimum power and delay investigation: 78
4.4.3 Low Vdd for non-critical path circuits 81
4.4.4 Low V_{dd} for the all blocks: 83
4.5 Conclusion and Future Work 85

Power Reduction of SBox Circuit in Deep Sub-Micron CMOS Technology **86**

5.1 Introduction: 86
5.2 Sources of leakage currents in CMOS circuits 88
5.2.1 Gate oxide tunneling leakage (I_G) 88
5.2.2 Sub-threshold Leakage Current (I_{SUB}) 90
5.2.3 Reverse-bias source/drain junction leakages (I_{REV}) 90
5.2.4 Gate induced drain leakage (I_{GIDL}) 91

5.2.5 Gate current due to hot-carrier injection (I_H) 91
5.3 Leakage control techniques 92
5.3.1 Circuit level leakage control techniques 92
5.3.1.1 Dual threshold method: 93
5.3.1.2 Variable threshold method: 93
5.3.1.3 Power supply gating: 93
5.3.1.4 Input vector control 94
5.3.2 Power Challenges at 65 nm 95
5.4 SBox circuit simulation in 65nm CMOS technology 96
5.5 Summary 98

Conclusion and Future Work 99

6.1 Summary of Contributions 99
6.2 Future Work 101

Appendix 106
Vita 112

List of Tables

Table 2. 1: Nanodevice type and their operational voltages 12

Table 2. 2: Values of the components used in the circuit 15

Table 2. 3: Comparison between existing and proposed Memory cell 18

Table 2. 4: Comparison between the three multiplexers (transistors width = 10λ) 21

Table 2. 5: Simulation results using wider transistors Modified and Simplified Mux .. 211

Table 2. 6: Simulation result using Modified Mux (Wp= Wn= 10λ) 23

Table 2. 7: Simulation result using Optimized Control Circuit (Wp = Wn = 10λ) 234

Table 2. 8: Simulation result with wider transistors 234

Table 2. 9: Simulated operational voltages for future device 28

Table 3. 1: AES parameter 43

Table 4. 1: Inversion of Byte $\{xy\} \in GF(2^8)$ in hexadecimal notation 65

Table 4. 2: Simulation results of SBox for 4 input signals 71

Table 4. 3: Output signals delay and power estimation of the SBox 73

Table 4. 4: Simulated results with TII having input at every 3ns 78

Table 4. 5: Signals time delay measured at the input of the TII 79

Table 4. 6: Simulation result to check the minimum delay 80

Table 4. 7: Simulation results considering minimum time delay and power dissipation . 80

Table 4. 8: Simulation result with Low V_{dd} at particular units 83

Table 4. 9: Results with different V_{dd} 83

Table 4. 10: Performance analysis for the longer input signals 84

Table 4. 11: Simulation with β=0.5 (instead of β=1) 84

Table 5. 1: Power Predictions by ITRS 87

Table 5. 2: Comparison of different leakage reduction techniques 94
Table 5. 3 : Delay and average power of the SBox 96
Table 5. 4: Simulation result having input at every 1ns 97
Table 5. 5: Simulation result of SBox circuit with tri-state buffer and input signals with 1ns pulse width 97
Table 5. 6: Simulation result using dual Vt 98

List of Figures

Figure 2.1: a) CMOL inverter cell b) Addressing a selected nanodevice [3] 6

Figure 2.2: Low-level structure of the generic CMOL circuit [15] 7

Figure 2.3: CMOL cells pin configuration [14] 8

Figure 2.4: Connectivity of CMOL cells, grey marked cells are within the Connectivity Domain[14] 9

Figure 2.5: CMOL wired-NOR gate:(a) schematics and (b) one of (many possible) configurations. [14] 10

Figure 2.6: I-V Characteristics of a single molecule [15] 11

Figure 2.7: NMOS characteristics with Vg=1V, V_{in}=1V to -1V, Plotted V_{out} (bold) shows the V_t drop during its ON state 13

Figure 2.8: PMOS characteristics with V_g =0V, V_{in}=1V to -1V, Plotted V_{out} (bold) 13

Figure 2.9: (a) CMOL cell used as 1-bit memory (b) Equivalent circuit of current CMOL memory cell. 14

Figure 2.10: Proposed CMOL memory cell 16

Figure 2.11: Memory Structure with nanowire interface 17

Figure 2.12: Equivalent circuit of the proposed CMOL memory cell. 17

Figure 2.13: CMOL memory array with the control circuit 19

Figure 2. 14: Generalize TG based Multiplexer 20

Figure 2.15: (a) Modified Multiplexer design (b) Simplified Multiplexer design 20

Figure 2.16: Control circuit using Modified MUX 22

Figure 2.17: Optimized control circuit 23

Figure 2.18: Unwanted devices in operation due to multiple cells selection at a time..... 25

Figure 2.19: Layout of the proposed CMOL cell memory 26

Figure 2.20: Nanowire-molecule-nanowire crossbar memory structure 27
Figure 2.21: Operation on Crossbar molecular memory 27
Figure 2.22: Existing replacement algorithm [10] 30
Figure 2. 23: Fixing the least distance cell and checking the domain range intersection 33
Figure 2.24: After checking the input output domain range intersection, shows the new LD location. 33
Figure 2.25: Since the replacement was not possible, a new cell is chosen as 'LD' 34
Figure 2.26: Choose a new free cell as no candidate cell was available 34
Figure 2.27: Start the procedure again for the new target cell 35
Figure 2.28: Defective cell can be replaced with a free cell 35
Figure 2.29: Potential energy of the shuttle at different locations in the capsule. The solid line is when no electric field is applied. The dashed lines are the potential energy when the two-volt potential difference is applied. [24] 36
Figure 2.30: Write 0, 1 and read operation of a Buckyball-CNT based future memory cell. 37
Figure 3. 1: General structure of AES[25] 44
Figure 3. 2: AES Data Structures [25] 45
Figure 3. 3: AES Encryption round[25] 47
Figure 3. 4: The effect of the SubBytes transformation on the State [29]. 52
Figure 3. 5: S-box: substitution values for the byte xy (in hexadecimal format)[29] 52
Figure 3. 6: Inverse S-box: substitution values for the byte xy (in hexadecimal format)[29] 53
Figure 3. 7: Shift row transformation 53
Figure 3. 8: Example of the ShiftRow operation 54
Figure 3. 9: MixColumns operates on the State column-by-column. 55
Figure 3. 10: AddRoundKey() XORs each column of the State with a word from the key schedule 57
Figure 4. 1: Architecture of the AES-SBox and InvSBox [33] 67
Figure 4. 2: Schematic diagram of XOR gate 68
Figure 4. 3: Schematic diagram of "map" 69
Figure 4. 4: Schematic diagram of the Inversion ('Inv' in Fig 4.1) in GF(2^4) 69

Figure 4. 5: Schematic diagram of Multiplication (⊗) (q = a ⊗ b) 70
Figure 4. 6: Schematic diagram of Map_Inverse 70
Figure 4. 7: Schematic diagram of Affine Transformation 71
Figure 4. 8: (a) Measured signals at inverter-multiplier stages, Signals coming through the INV (bold lines) need more delay then others; (b) SBox output signals show unwanted switching during the first 1ns 72
Figure 4. 9: (a) Schematic diagram of the tri-state inverter [23] (b) Tri-state inverter followed by another inverter 74
Figure 4. 10: (a) Input and (b) output signals of the TII without any load. 75
Figure 4. 11: Output signal of the TII with load. 76
Figure 4. 12: Comparison of (a) input and (b) output signals of the wider TII with load 77
Figure 4. 13: Output signals with TII enabled after 2ns of the input signal 79
Figure 4. 14: Output signals of the simulation shown in table 4.6. 80
Figure 4. 15: Output signals of the simulation result shown in table 4.7 81
Figure 4. 16: Architecture of S-box with Low V_{dd} units 82
Figure 5. 1: Leakage current components in NMOS transistor [1] 88
Figure 5. 2: Gate tunneling leakage current in NMOS and PMOS transistors [41] 89
Figure 5. 3: Static and Dynamic Power vs. Process Nodes [51] 95

List of Abbreviation

λ	Half of the minimum transistor channel length
6T	6 Transistors
AES	Advanced Encryption Standard
Array	An enumerated collection of identical entities
ASIC	Application-specific integrated circuits
Bit	A binary digit having a value of 0 or 1
Block Round	Sequence of binary bits that comprise the input, output, state and key. Blocks are interpreted as arrays of bytes
BPDN-DT	Single bipyridyl-dinitro oligo-phenylene ethynylene dithiol
Byte Array	A group of eight bits that is treated either as a single entity or as an 8 individual bits
C1	Candidate cell
C_{60}	Molecule that consists of 60 carbon atoms
Cipher	Series of transformation that converts plaintext to ciphertext using the Cipher Key.
Cipher Key	Secret Cryptographic key that is used by the Key Expansion routine to generate a set of Round Keys.
CMOS	Complementary Metal Oxide Semiconductor
CMOL	CMOS-Molecule hybrid
De	Defective cell
DES	Data Encryption Standard
GF	Galois field
GHz	Giga Hertz
GP	General Purpose
Inv	Inversion
I_G	Gate oxide tunneling leakage current

I_{SUB}	Subthreshold leakage current
I_{REV}	Reverse-bias source/drain junction leakages current
I_{GIDL}	Gate Induced Drain Leakage current
I_H	Gate current due to hot-carrier injection current
ITRS	International Technology Roadmap for Semiconductors
Key Expansion	Routine used to generate a series of Round Keys from the Cipher Key
LD	Least distance cell
LP	Low power
Lvt	Low Vt
MOSFET	Metal-Oxide Semiconductor Field Effect Transistor
MUX	Multiplexer
NIST	National Institute of Standards and Technology
NMOS	n-channel MOSFET
PMOS	p-channel MOSFET
Round Key	Values derived from the Cipher Key using the Secret Cryptographic Key
State	Intermediate Cipher result that can be pictured as a rectangular array of bytes.
SBox	Substitution Box
SRAM	Static Random Access Memory
SQU	Square
Svt	Standard Vt
TII	Tri-state Inverter followed by another Inverter
TG	Transmission Gate
Vdd	Positive edge voltage of the power supply
Vt	Threshold Voltage
VLSI	Very Large Scale Integration
Wp	With of PMOS
Wn	Width of NMOS
W	Watt

Word	A group of 32 bits that is treated either as a single entity or as an array of 4 bytes.

Chapter 1

Introduction

1.1 General Introduction

The introduction, with continuous performance enhancement, of CMOS integrated circuit is a major milestone in the history of modern industry. It has driven a revolution in computing capability due to a long trend in increased performance, higher device density, and lower cost with scaling [1]. This is due to the continued scaling of technology and supply/threshold voltage. However; leakage power has become more significant in nanoscale CMOS circuits. Therefore, the reduction of the total leakage power is critical to achieve low power operation of digital circuits in deep sum-micron era.

ITRS 2007 predicted that during next decade the continuation of application of Moore's-Law to semiconductor electronics will require the transfer to hybrid VLSI circuit based on heterogeneous device technology, and not limited to CMOS [2]. Recently, a novel hybrid nanowire/molecular/semiconductor circuit, namely CMOL, was introduced [3],

consisting of nanowires on top of a CMOS stack. The basic idea of CMOL circuits is to combine the advantages of CMOS technology (with its flexibility and high fabrication yield) with those of molecular-scale nanodevices. Two-terminal nanodevices would be naturally incorporated into nanowire crossbar fabric, enabling very high density at acceptable fabrication costs. The most straightforward possible application of CMOL circuits is very high density resistive memories [4].

1.2 Motivation

In recent years, security has been more and more significant in network environment with the emergence of the internetworking technology and the important role of Cryptography in data transmission security. The inherent advantages of using VLSI chips for encryption are speed and physical security. Therefore, ASIC design and implementation of cryptosystems has been a motivational and challenging subject. In 1997, NIST (National Institute of Standards and Technology) decided that a new standard algorithm is needed because attacks like exhaustive key search exploiting the short key length of DES had been demonstrated. Through three Advanced Encryption Standard (AES) conferences, Rijndael [5] was selected as AES in October 2000. Its VLSI implementation was taken into consideration and it is currently integrated in various embedded applications like Web Servers, ATMs, Fiber Distributed Data Interfaces (FDDIs), smart cards, cellular phones etc. Since 2000, several architectures for efficient VLSI realization of AES algorithm have been proposed and their performance evaluated using ASIC libraries and FPGAs[6]. Further integration or speed-up of such circuits will not be easily possible in conventional manners. The ongoing feature size reduction of silicon based CMOS technology which has been a basis for explosive growth of the

semiconductor industry for the last three decades will run into severe physical and economic problems [6]. The scaling of MOSFETs is entering the deep-nanosized regime in which fundamental limits of CMOS technology along with economical challenges are encountered[6].

While traditional silicon based microelectronics is gradually approaching the end of its scaling, novel nanoelectronic solutions is needed to surmount the physical and economic barriers of current semiconductor technology. A feasible scenario is the integration of silicon with non-CMOS nanoelectronic, i.e. a mixed CMOS/nano system. Based on this concept, an AES design was proposed using CMOS/molecule hybrid (CMOL) technology [6]. Memory is one the impotent elements to design any electronic systems including the AES hardware implementation. Design of memory based on CMOL architecture is expectable improve the performance and enhance the application of CMOL. Considering this issue the design and simulation of a CMOL memory cell is integrated in this thesis.

Designing the AES in the deep submicron technology brings the high leakage power problem in front due to the aggressive scaling of the MOSFET devices. In fact any design in very deep-sub-micron process such as 90 nm or below, leakage power becomes significant and cannot be ignored [7]. For example, Intel Pentium IV processors running at 3GHz already have an almost equal amount of dynamic power and leakage power at a high operating temperature[7]. It is also established that the major power of the AES is consumed by the SBox [8]. Design of power optimized SBox using the deep sub-micron CMOS technology is now a demand of time to have an efficient ASIC design of AES.

1.3 Thesis Organization

This thesis is concerned with the design of a nanoelectronic memory cell based on CMOL and followed by an optimized design of SBox for Advanced Encryption Standard (AES) using deep submicron CMOS circuit technology. Chapter 2 begins by reviewing the architecture of CMOL, followed by a review of the characteristics of the nanodevice used as the memory element. Pass transistors are used in current CMOL architecture, their operational limitations are demonstrated through HSPICE simulations. Afterward, the analyses of the operational limitations of the existing CMOL architecture are presented followed by the proposed solution. A literature review on AES is described in chapter 3. In chapter 4, the design details of the SBox are presented along with a short mathematical review of the operations used in SBox. A design with improved performance of the SBox is also illustrated in this chapter. Chapter 5 described the leakage reduction techniques in the deep submicron technology along with their application for SBox design in 65nm CMOS technology. Last chapter, Chapter 6, presents a summary of the contributions and suggestions for further future work.

Chapter 2

Nanoelectronic Memory Design

2.1 Introduction

The performance of many electronic products, including all computing systems (from palmtops to supercomputers) and many portable consumer electronic devices (cell phones, portable digital video and music players, digital cameras, etc.) substantially depends on memory [9]. Advancement in silicon technology continues to the end of the Moore's Law, predicted with the end of CMOS scaling, only 10-15 years away. Consequently, there is a strong interest in new, molecular-scale devices that might complement the basic silicon platform by providing it with new capabilities or that might even replace Silicon technology and allow device scaling to continue to the atomic scale. Now the very success of some electronic systems may depend on the memory scaling prospects.

Among all the promising candidates for the next generation nanoelectronics technology, CMOL (hybrid CMOS/nanowire/molecule) is considered for the combination of the

advantages of CMOS technology (including flexibility and high fabrication yield) with the extremely high density of molecular-scale two terminal nanodevices. CMOL based memory cells offer high density, programmability and easy configuration superiority [10, 11]. The CMOL structure is based on the combination of a traditional CMOS stack, having a nanowire crossbar structure, with molecular electronic devices at the crossbar intersection points [3,12,13]. A single CMOL inverter cell and nanowires interfacing the nanodeive are shown in Figure 2.1 (a) and (b).

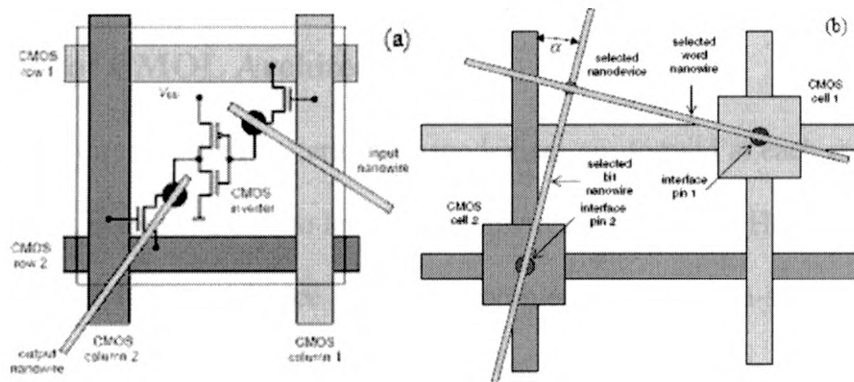


Figure 2.1: a) CMOL inverter cell b) Addressing a selected nanodevice [3]

Molecular nanodevices formed between the nanowires at every cross point, with charge storage properties, used as nanoscale memory devices. The nanodevice provides diode like I-V curves for logic circuit operations and allow circuit mapping on CMOL fabric [14]. The possibility to reconfigure the nanodevice after fabrication makes this approach even more valuable then the leading CMOS technology [4]. Based on CMOL concept, terabit scale memory was proposed [4] using traditional crossbar memory structure. The top-level nanowires stretch over the whole block, but the low-level nanowires are cut into segments of equal length. An analysis of the CMOL geometry shows that each nanowire

segment stretches over r CMOS cells and contacts r^2 crosspoint nanodevices, where r is a positive integer number [4]. Due to its different cell structure than the CMOL logic cell (like inverter cell), design and fabrication of any logic and memory circuit on a single chip will be complicated. Moreover, the horizontal single wire connected with r^2 nanodevices may affect some non selected devices that finally may change the stored information. To overcome all these complexity, Transmission gate based CMOL memory cell along with its control circuits and reconfiguration approach are proposed in this chapter.

2.2 Review of CMOL Architecture

In CMOL circuits, the two-terminal nanodevices are formed at each crosspoint of a "crossbar" array, consisting of two levels of nanowires (Fig 2.1(b)). However, in order to overcome the CMOS/nanodevice interface problems, a-few-nm-sharp-tip pins are distributed all over the circuit area, on the top of the CMOS stack (Fig 2.2) [15]. By activating two pairs of perpendicular CMOS lines, two pins (and two nanowires they contact) may be connected to CMOS data lines (Fig. 2.1(b)).

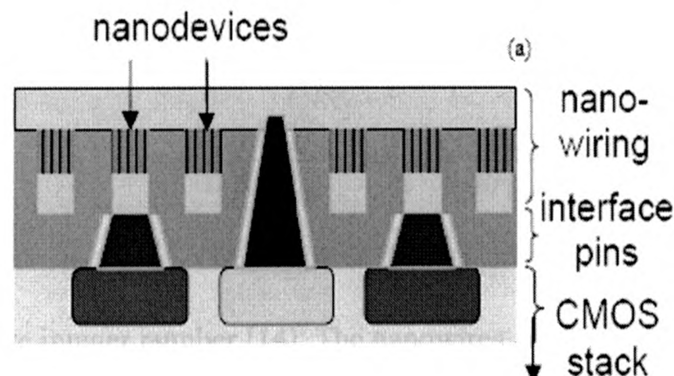


Figure 2.2: Low-level structure of the generic CMOL circuit [15]

Figure 2.3 shows pins of each type (reaching to the lower and upper nanowire level) are arranged into a square array with side $2\beta F_{\mathrm{CMOS}}$, where F_{CMOS} is the half-pitch of the CMOS subsystem, while β is a dimensionless factor larger than 1 depending on the CMOS cell complexity. The nanowire crossbar is turned by an angle $\alpha = \arcsin(F_{\mathrm{nano}}/\beta F_{\mathrm{CMOS}})$ relative to the CMOS pin array, where Fnano is the nanowiring half-pitch.

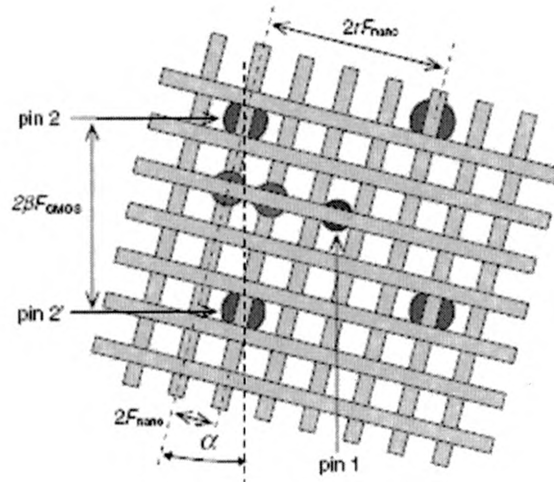


Figure 2.3: CMOL cells pin configuration [14]

More exactly, the requirements for the angle α and the dimensionless factor β that determines the CMOS cell area $A = (2\beta\, F_{\mathrm{CMOS}})^2$ now take the form:

$$\cos\alpha = \frac{rF_{nano}}{\beta F_{CMOS}}, \quad \sin\alpha = \frac{F_{nano}}{\beta F_{CMOS}}$$

where r is a positive integer number [14]. The nanowires are fabricated with small breaks repeated with period $L = 2\beta^2 F^2_{\mathrm{CMOS}}/F_{nano}$ [14]. With this arrangement, each nanowire segment is connected to one interface pin. As a result, each input or output of a CMOS cell can be connected through a pin–nanowire–nanodevice–nanowire–pin link to each of

$M = 2r\ (r - 1) - 1$ other cells located within a squareshaped 'connectivity domain' around the initial cell (Fig. 2.4). CMOS cells painted light-grey (in the shown case, $r = 3$, $M = 11$) form the 'connectivity domain' for the input pin of the cell painted dark-grey (Fig. 2.4). Note that there are r nanowires of one orientation and $(r - 1)$ of the perpendicular orientation per CMOS cell [14].

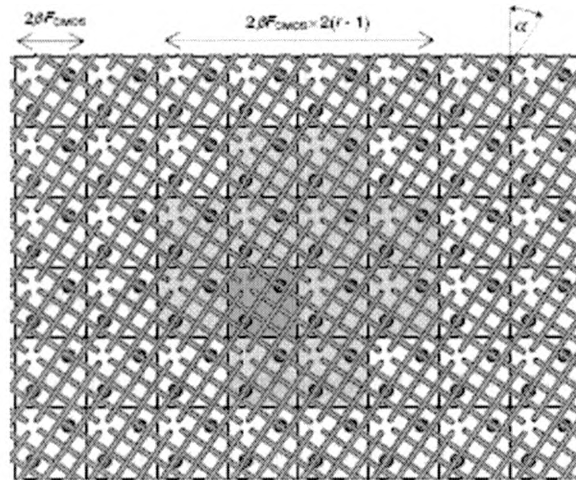


Figure 2.4: Connectivity of CMOL cells, grey marked cells are within the Connectivity Domain[14]

Implementation of any logic operation on CMOL is done in two stages. During the configuration stage, all inverters are disabled by an appropriate choice of global voltages V_{dd} and V_{gnd} (figure 2.1(a)). When the configuration stage has been completed, the pass transistors are used as pull-down resistors, while the nanodevices set into ON (low-resistive) state are used as pullup resistors. Together with CMOS inverters, these components may be used to form the basic 'wired-NOR' gates (figure 2.5).

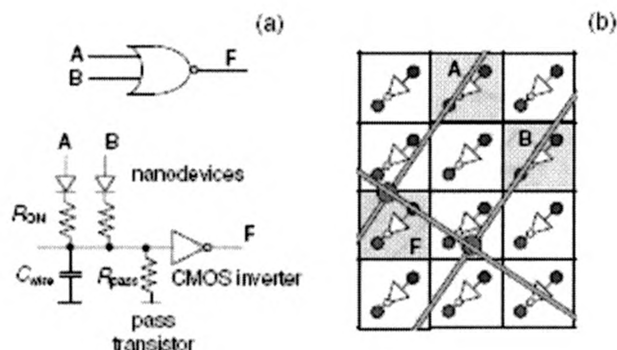


Figure 2.5: CMOL wired-NOR gate:(a) schematics and (b) one of (many possible) configurations. [14]

2.3 Nanodevice properties

A single molecule nanodevice is programmable to ON or OFF-state and consequently can be used as a storing element. Its characteristic is one of the major issues to overcome to design a memory cell whose operation is expected to be compatible with a specific CMOS nanotechnology. Different combinations of organic molecules, with varying electrical behaviors and molecular structures, have different operational voltages [16]. Physics of single-electron devices is based on the "Coulomb blockade" effect; it is the increased resistance at small bias voltages of an electronic device comprising at least one low capacitance tunnel junction [17].

Figure 2.6 shows the I-V characteristics of a typical latching switch that can be used as a memory storing element. In the OFF state the device passes almost no current until the applied voltage reaches a certain threshold value $V_t^{(+)}$. At this point, the device switches to ON-state (Write 1) with a finite current. The reciprocal switching takes place at voltage $V_t^{(-)}$ Write 0). A voltage of V_{READ} can be used to sense the state of the nanodevice (READ operation).

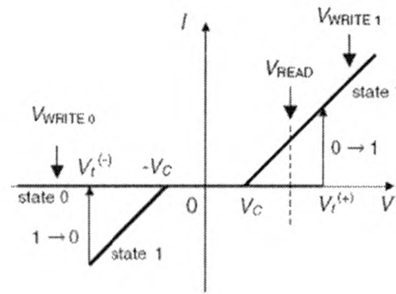


Figure 2.6: I-V Characteristics of a single molecule [15]

The Coulomb- blockade range is from $-V_C$ to $+V_C$. In the low-resistive state representing binary 1, the nanodevice is essentially a diode, so that the application of voltage $V_C<V_{READ}< V_t^{(+)}$ to top nanowire leading to the memory cell gives substantial current injection into the bottom wire. This current pulls up the output voltage which can be read out by a sense amplifier. In the OFF state (binary 0) the current through the molecular switch is very small, giving a nominally negligible contribution to output signals at readout. In order to switch it to "ON" state 1 (i.e., write binary 1 into the cell), the two nanowires leading to the device are fed by voltages $\pm V_{WRITE}$, with $V_{WRITE} < Vt(+) < 2V_{WRITE}$. (The left inequality ensures that this operation does not disturb the state of "semi-selected" devices contacting just one of the biased nanowires). The write 0 operation is performed similarly using the reciprocal switching with threshold $V_t(-)$ (Fig. 2.6). Table 2.1 shows the different operational voltages of various types of nanodevices.

Table 2. 1: Nanodevice type and their operational voltages

Device Type	Operational Voltages
Oligo (Phenylene ethynylene) molecule with a nitro sidegroup [18]	Write 1: More then 2.5v Write 0: Less then – 2.3v Read : 1V
Fe+ terpyridie molecules [19].	Write 1: 25v Write 0: –12.5v Read : 0.5v
monolayer of rotaxane [20]	Write 1: 3.5v to 7v Write 0: –3.5v to –7v Read : 0.5v
Redox active molecules [21]	Write 1: 10v Write 0: –10v Read : 0.2v
single bipyridyl-dinitro oligo-phenylene ethynylene dithiol (BPDN-DT) [16]	Write 1: more then 1.3V Write 0: less then –1.3v Read : 1v

Among the different operational voltages (Table 2.1), 1.6V and -1.6V were chosen as the ON (write 1) and OFF (write 0) states, and 1V for read operation using Single bipyridyl-dinitro oligo-phenylene ethynylene dithiol (BPDN-DT) as the nanodevice [16]. The Coulomb-blockade range [$-V_C$ to V_C, Fig. 2.6] was –1.3V to 1.3V.

2.4 Limitation of MOSFET when used as Pass transistor

NMOS is used as the interface between the data lines and the nanowires in the existing CMOL architecture (Fig. 2.1(a)). Functional limitations of the PMOS and NMOS need to be investigated before carrying out HSPICE simulation of CMOL equivalent circuits. HSPICE Simulation was performed, based on 45nm CMOS technology, to check the MOSFET used as Pass transistor behavior by passing a sinusoidal voltage of 1V amplitude. MOSFET models, based on Berkeley predictive

technology [22], were used in the circuit simulation, representing the future 45nm CMOS technology [2]. In the positive cycle a drop in the output voltage level, known as threshold voltage loss [23] of 0.3V due to the Vgs > Vt (Vgs is the voltage applied between the gate and source terminal of the transistor and Vt is the threshold voltage) to keep the NMOS transistor on, was found. In the negative cycle, the voltage drop was around –0.35V (Figure 2.7). PMOS can pass a high voltage level without any losses (strong logic 1); however, for low input voltages, below 0.3V, the transistor will be barely on, operating close to cut-off mode, by maintaining V_{sg} to at least +0.3V (Figure 2.8). 1V and 0V were the gate voltages (V_g) respectively for NMOS and PMOS in the simulation.

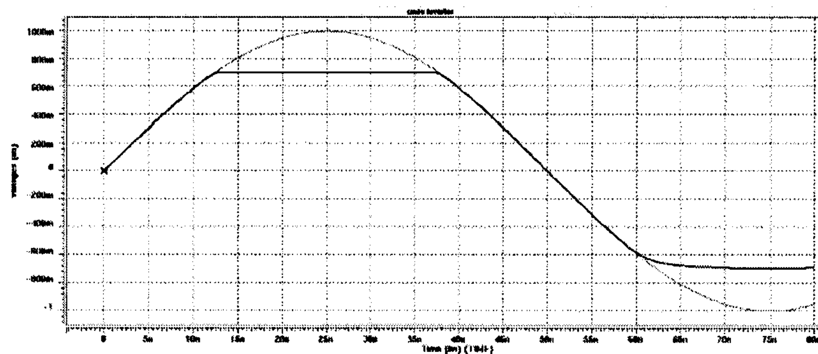


Figure 2.7: NMOS characteristics with Vg=1V, V_{in}=1V to -1V, Plotted V_{out} (bold) shows the V_t drop during its ON state

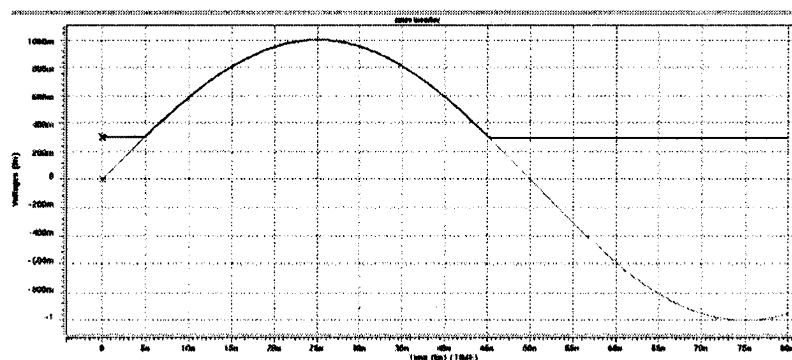


Figure 2.8: PMOS characteristics with V_g =0V, V_{in}=1V to -1V, Plotted V_{out} (bold)

Based on the simulation result, 1V and –0.6V were chosen as the two references voltages to have the +1.6V and -1.6V voltage differences across the molecular device to turn it ON and OFF respectively. In the next subsection the existing CMOL memory structure is describe followed by the need of an improved design and thereafter a new CMOL memory structure is proposed.

2.5 Memory based on existing CMOL cell

The CMOL cell can be used to store single bit information by using the two terminal bi-stable single electron nanodevices, known as the programmable molecular switch, which are formed at each crosspoint of nanowires in a CMOL cell (Fig. 2.9(a)). Implementation of nanoscale memory based on existing CMOL architecture, and its equivalent circuit, are shown in Fig. 2.9(a) and (b). The nanowire's equivalent resistance and capacitance are calculated by considering a single cell area, Acell = $(2\beta F_{CMOS})^2$ = 0.032 μm^2, Fcmos = half pitch of the CMOS subsystem =22.5nm and β(dimensionless factor depends on CMOS cell complexity) = 4 [10].

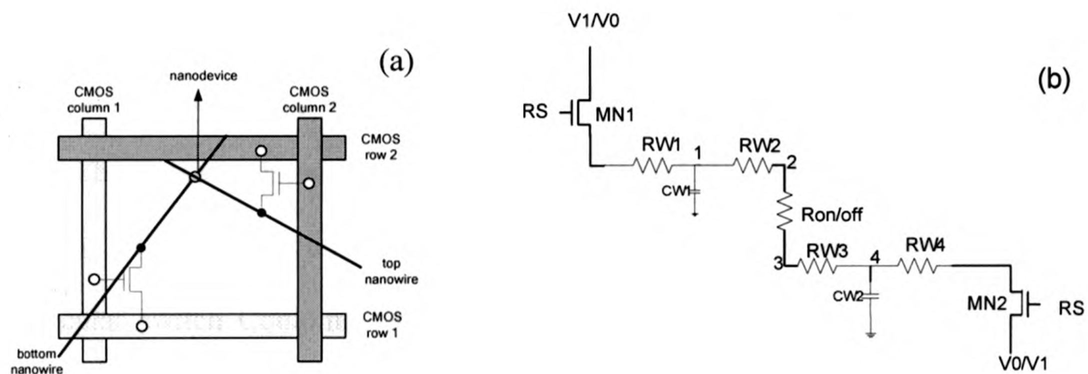


Figure 2.9: (a) CMOL cell used as 1-bit memory (b) Equivalent circuit of current CMOL memory cell.

A 1.8μm long naowire's equivalent resistance and capacitance are calculated considering the half pitch of the nanowire, Fnano =4.5nm, and wire capacitance of 0.2fF/μm [10]. Single molecule (BPDNDT) ON and OFF states have an equivalent resistance of 0.71 * 10^9 Ω and 18 * 10^9 Ω respectively [16]. For self-assembled monolayers, the footprint of a single molecular is 0.25nm^2 [14]. So the number of molecules per cross point is $F^2_{nano}/0.25 = 81$, and consequently 8.7 * 10^6 Ω and 0.22 * 10^9 Ω were used as the ON and OFF resistance in our HSPICE simulation. Table 2.2 shows the components values used in the equivalent circuit of a memory cell.

Table 2. 2: Values of the components used in the circuit

Names	Values
RW1, RW2, RW3, RW4	1kΩ
CW1, CW2	0.36 fF
Ron	8.7 * 10^6 Ω
Roff	0.22 * 10^9Ω

Analysis from the section 2.4, 1V and -0.6V were chosen as V1 and V0 to get the maximum voltage difference across the device (nodes 2 and 3, Fig 2.9(b)). HSPICE Simulation results show that, applying V1 and V0 will result in a voltage difference across the device during the ON and OFF of 1.15V and -1.05V only respectively. This is caused by pass transistors limitations since Vout =0.65V, with Vin =1V and Vg=1V (see section 2.4). Device current state can not be changed by these low voltages since molecular switch Coulomb blocked range is –1.3V to 1.3V [16]. Modification of the existing CMOL cell architecture is required to overcome this operational difficulty.

2.6 Proposed Memory Cell

MOSFET operation analysis with the voltage requirement shows that both the PMOS and NMOS are needed to pass the voltage signals during the ON and OFF operations. In the proposed CMOL memory cell, transmission gate (TG) based structure is proposed instead of pass transistors. Figure 2.10 shows the structure of the proposed CMOL cell.

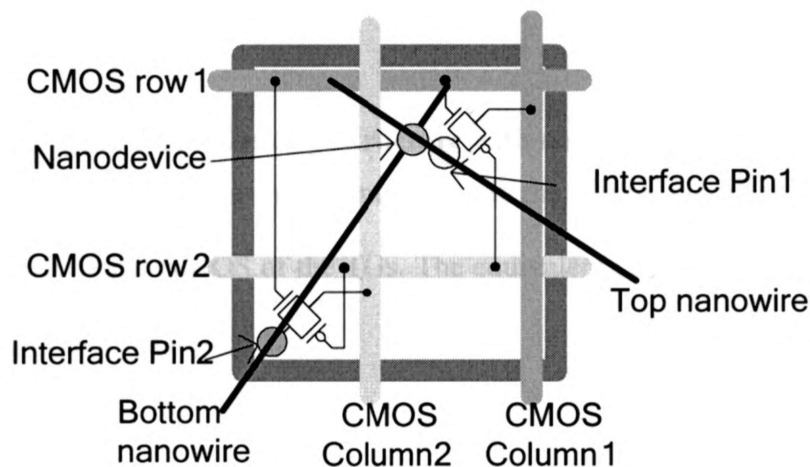


Figure 2.10: Proposed CMOL memory cell

Similar to the existing CMOL cell, activating two perpendicular CMOS lines (a row and a column), two pins (two nanowires they contact), a nanodevice is accessed to be programmed and consequently used as a single memory cell.

Figure 2.11 shows the memory structure along with the nanowires contacting the memory cells. To switch the device OFF to ON the reference voltages 1V and -0.6V need to be passed through top and bottom nanowires and to switch the device ON to OFF, the applied voltages are reversed.

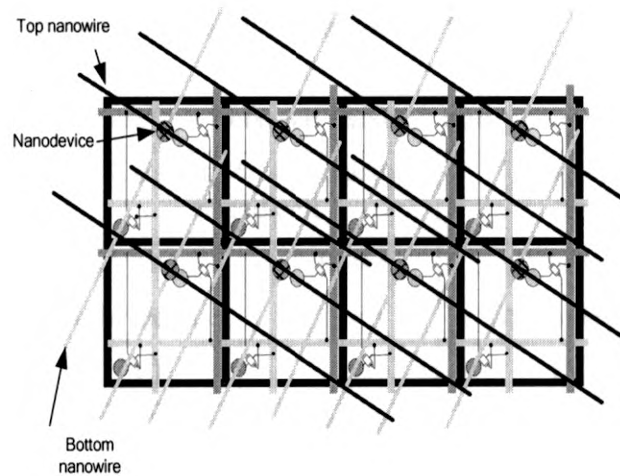


Figure 2.11: Memory Structure with nanowire interface

In an addition an external inverter is needed with the control circuit to pass the proper Vg to the PMOS and NMOS of the TGs. The equivalent circuit of the proposed single memory cell is shown in Fig. 2.12 .

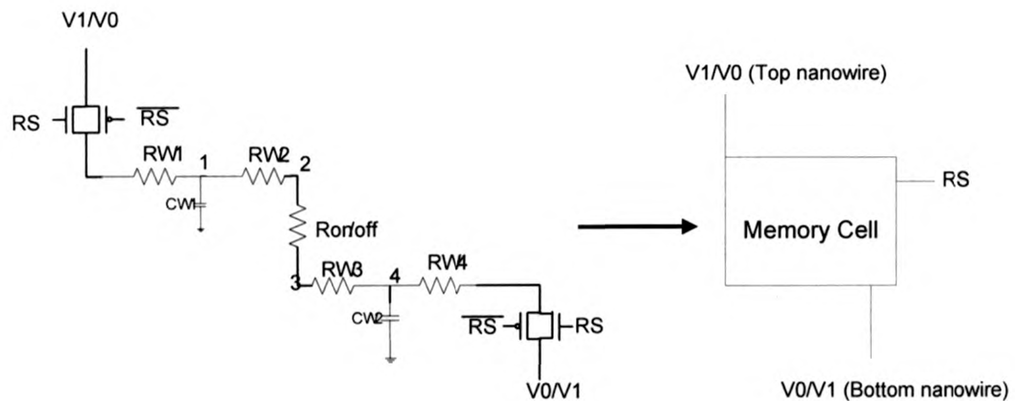


Figure 2.12: Equivalent circuit of the proposed CMOL memory cell and its symbol.

HSPICE Simulation results show that the proposed transmission gate based CMOL memory cell can pass voltages larger then the minimum required to turn the device ON and OFF (1.3V and –1.3V) while the average power is maintained the same as the existing memory cell. Table 2.3 shows the comparison between the existing and the

proposed CMOL memory cells operation. Here all the transistors have the same width of 10λ; where λ=22.5nm, half pitch of the 45nm technology.

Table 2. 3: Comparison between existing and proposed Memory cell

Operation	Existing Pass transistor based cell		Proposed Transmission gate based cell	
	Voltage at the device (V)	Power (W)	Voltage at the device (V)	Power (W)
OFF to ON	1.15V	$8.09*10^{-5}$	1.58	$8*10^{-5}$
ON to OFF	–1.05V	$8.32*10^{-5}$	–1.58	$8.34*10^{-5}$

2.7 Design of control circuit

Control circuits need to pass V1 and V0 to the device based on the specific operation; while a value of Vg = 1V causes Vgs < –Vt (if Vs > 1.3V) for PMOS. Similarly, Vg=0V causes Vgs > Vt (if Vs < –0.3V) for NMOS respectively (a threshold voltage |Vt|= 0.3V is assumed). Analyzing MOSFET operations (see section 2.4) and voltage requirements to make the device ON and OFF, 1.2V and –0.4V were chosen as the two reference voltages (V1 and V0 in Fig 2.13) needed to turn the device ON/OFF. As these voltages are within the range of Coulomb-blocked (–1.3Vto1.3V), and consequently the unselected devices will not be affected [16]. These two voltage levels (1.2V and –0.4V) are the best combination as more positive or negative voltages may violate V_{WRITE} < 1.3V and make the PMOS or NMOS ON, respectively.

Fig. 2.13 shows a CMOL memory array with the control circuit. The read sensor circuit is not shown, it will be connected with the CMOS column2 (Fig 2.10) and the stored bit can be read by sensing the on and off current.

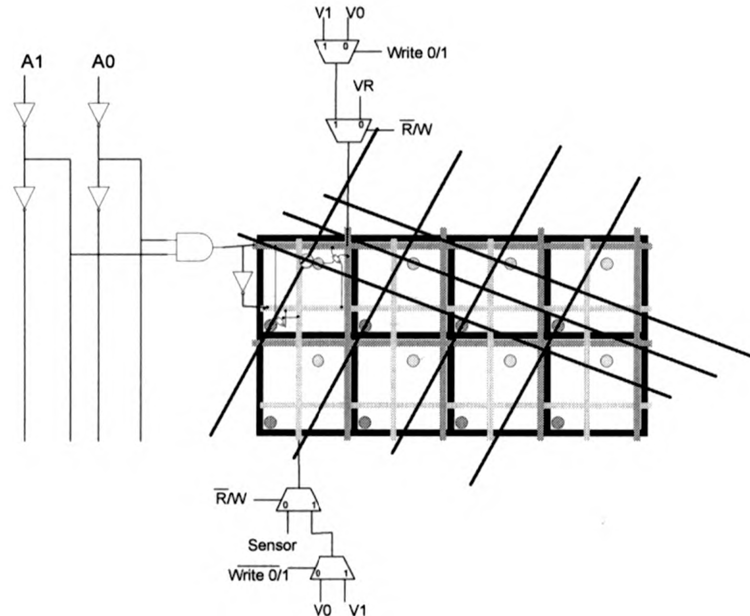


Figure 2.13: CMOL memory array with the control circuit

The required ON/OFF voltages, transmitted to the device through the cell's transmission gates, are selected by the control circuits connected with the top and bottom nanowires. Simulation result of a Transmission gate based multiplexer is presented in the following subsection. However, it does not operate properly and a modification of its design is proposed.

2.7.1 Proposed Multiplexer:

Transmission gate based multiplexer (Fig 2.14), to pass positive (V1) and negative (V0) references voltages based on the selection signal (S), is used in the control circuit. Figure 2.14 shows the general schematic diagram of the multiplexer. However, a voltage of V0 (−0.4V) turns the NMOS ON, instead of being OFF,

when S'= 0 (Vgs > Vt, Vt=0.3V) and V(V1) = 1.2V, causing a degraded output voltage (around 0.3V drop, see Table 2.4) due to the leakage current of the partially on NMOS.

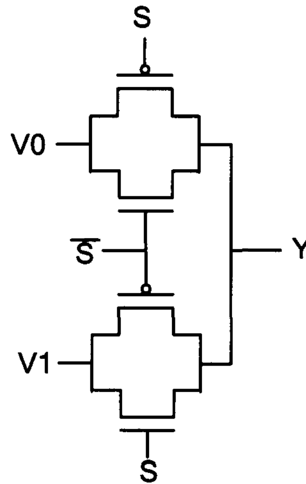
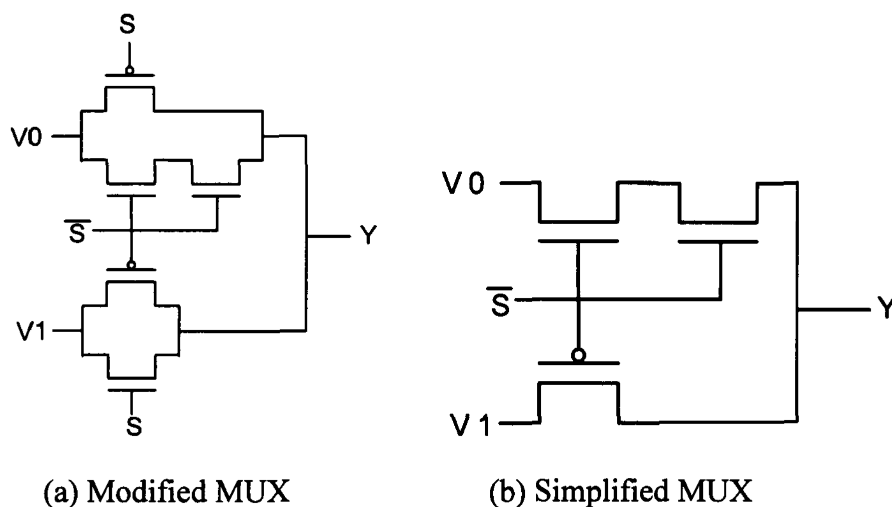


Figure 2. 14: Generalize TG based Multiplexer

A modified MUX is proposed to improve the performance of the multiplexer by adding a series NMOS (Fig. 2.15 (a)) to the transmission gates connected with V0; this forces the series NMOS into cutoff region when S'=0. Simplified MUX (Fig. 2.15 (b) having minimum number of transistors is presented for the specific memory operation.



(a) Modified MUX (b) Simplified MUX

Figure 2.15: (a) Modified Multiplexer design (b) Simplified Multiplexer design

Table 2.4 shows the comparison between the performances of the TG based, proposed Modified and Simplified Multiplexers.

Table 2. 4: Comparison between the three multiplexers (transistors with =10λ)

A 50MHz signal was connected as the selection signal S.

Type of Mux	Voltage at Y when S =1	Voltage at Y when S = 0	Power (W)	Time Delay
TG based MUX	0.956 V	− 0.357 V	$7.21 * 10^{-5}$	t_{LH}=10 Ps t_{HL}=06 Ps
Modified MUX	1.17 V	− 0.335 V	$3.16 * 10^{-5}$	t_{LH}=11 Ps t_{HL}=12 Ps
Simplified MUX	1.177 V	-0.394 V	$7.07 * 10^{-6}$	t_{LH}=11 Ps t_{HL}=07 Ps

Output voltage of the MUX can be improved by increasing the size of the PMOS(Wp) and series NMOS(Wn). Table 2.5 shows the result with Wp = 30λ and Wn (series) = 20λ.

Table 2. 5: Simulation results using wider transistors Modified and Simplified MUX

Type of Mux	Voltage at Y when S =1	Voltage at Y when S = 0	Power (W)	Time Delay
Modified MUX	1.188 V	- 0.34 V	$3.86 * 10^{-5}$	t_{LH}=08 Ps t_{HL}=18 Ps
Simplified MUX	1.192 V	- 0.386 V	$1.23 * 10^{-5}$	t_{LH}=08 Ps t_{HL}=14 Ps

The proposed Modified and Simplified MUXs reduce the average power dissipation by 46% and 83% compared to the TG based MUX with improvement in the output voltage levels.

2.7.2 Memory Write Operation

Turning the molecular device ON and OFF (writing 1 and 0) are more critical than the read operation, as applying voltage less than the coulomb blockade range is sufficient enough to sense the device state. Fig. 2.16 shows the full schematic diagram of the control circuit using four "Modified Multiplexers". Table 2.6 shows the simulation results with all transistors having equal width of 10λ. The average power was calculated considering the time period of voltage level changing of the Write 0/1 signal pulse, assuming that R/W signal arrives at the same time of the Write 0/1.

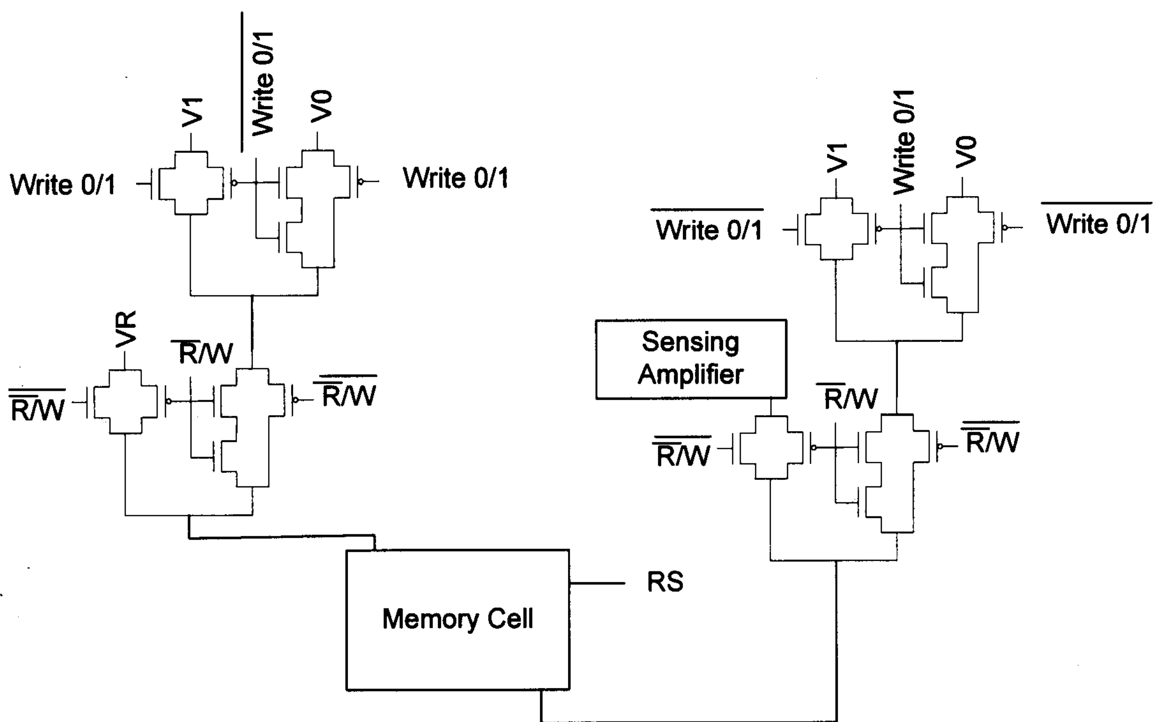


Figure 2.16: Control circuit using Modified MUX

Table 2. 6: Simulation result using Modified MUX (Wp = Wn = 10λ)

Operation	Delay1 (Ps)	Delay2 (Ps)	Volt. Drop (V)	Average Power (W)
ON to OFF	88	127	-1.466	7.191 * 10 ^ -5
OFF to ON	83	136	1.5	6.33 * 10 ^ -5

Delay 1 = 50% of the control signal 'Write 0/1' to 50% of the required voltage |0.8V|
Delay 2= 50% of the control signal 'Write 0/1' to the Coulomb threshold voltage |1.3V|

Another simulation was done using an optimized control circuit. Simplified MUX and optimized Modified MUX were used in the control circuits (Fig. 2.17). Table 2.7 shows the simulation results with all transistors having an equal width of 10λ.

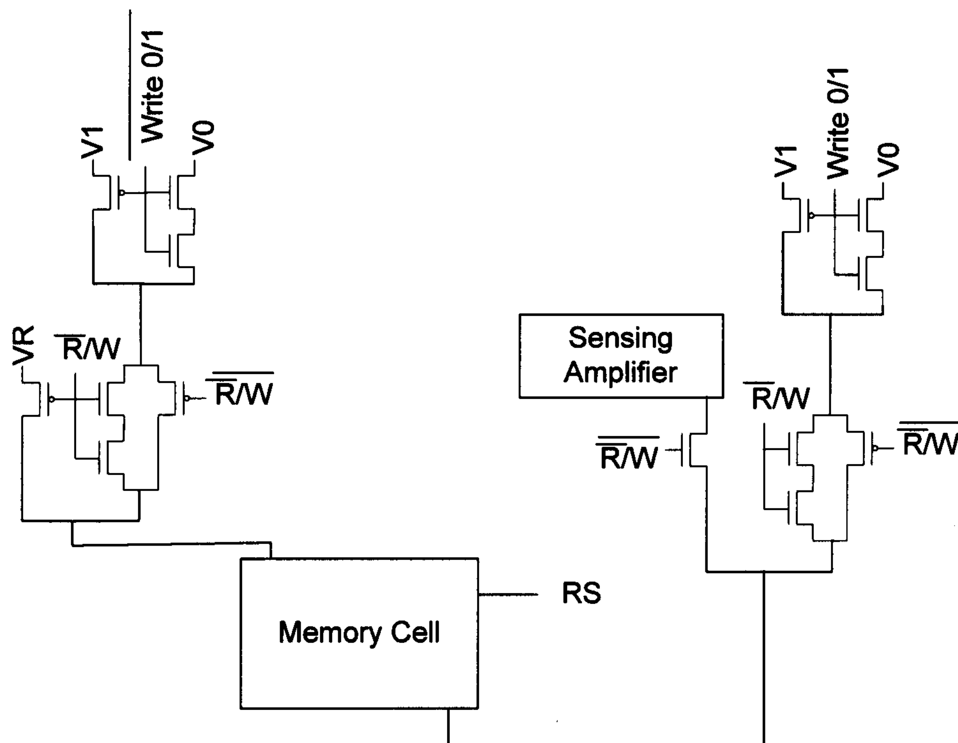


Figure 2.17: Optimized control circuit

Table 2. 7 Simulation result using Optimized control circuit (Wp= Wn= 10λ)

Operation	Delay1 Ps	Delay2 Ps	Volt. Drop (V)	Average Power (W)
ON to OFF	83	126	-1.566	1.454 * 10 ^ -5
OFF to ON	82	127	1.55	1.608 * 10 ^ -5

Circuits simulation, width chang ed from 10λ to 20λ for series NMOS and 30λ for PMOS, was carried out for the first type of control circuit (using Modified MUX) and the result showed some improvement was achieved by increasing the voltage across the nanodevice. Based on that, another simulation was performed with the optimized control circuit with Wp=30λ and Wn(series) =20λ , the results are shown in table 2.8.

Table 2. 8: Simulation Result with wider transistors

Operation	Control Circuit with Modified MUX (Fig. 2.16)				Optimized Control Circuit (Fig. 2.17)			
	Delay1 Ps	Delay2 Ps	Volt. Drop (V)	Average Power (μW)	Delay1 Ps	Delay2 Ps	Volt. Drop (V)	Average Power (μW)
ON to OFF	63	81	-1.495	89.12	62	84	-1.575	25.7
OFF to ON	65	90	1.523	78.12	64	87	1.564	27.02

Simulation results (Table 2.8) show that around ±1.57V (exceeding the Coulomb blocked ranges of ± 1.3V) can be achieved by using the optimized control circuit with 71% power saving and likely same delays.

Some unwanted devices may be turned ON or OFF due to selecting multiple series cells at a time. Fig. 2.18 shows this effect where grey circle in the dark grey cell is the chosen

nanodevice for write operation while the black circle devices are selected by the right and left side cells activated at the same time (r=3, grey marked cells defined domain connectivity region of the dark grey cell)). Making cells gap depends on the value of 'r' is a possible solution to do the parallel write operation.

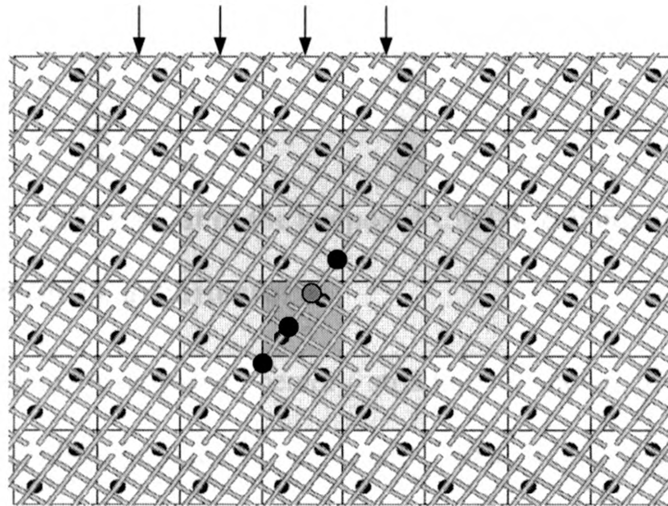


Figure 2.18: Unwanted devices in operation due to multiple cells selection at a time

2.8 Area Analysis of CMOL memory cells:

Proposed memory cell needs 4 transistors (2 transmission gates) similar to the existing CMOL inverter cell (Fig. 2.1(a)). So the proposed memory cell can be fabricated within the same area as CMOL inverter cell. This will simplify the fabrication process as well as implementation of some logic and memory integrated circuits. Following MOSIS design rules, the proposed memory cell needs 544 λ^2 (Fig. 2.19) area while a CMOS 6T SRAM needs 1170 λ^2 [23]. Based on the value of r, multiple bits can be stored in a CMOL memory cell that requires half of the area of a CMOS SRAM cell.

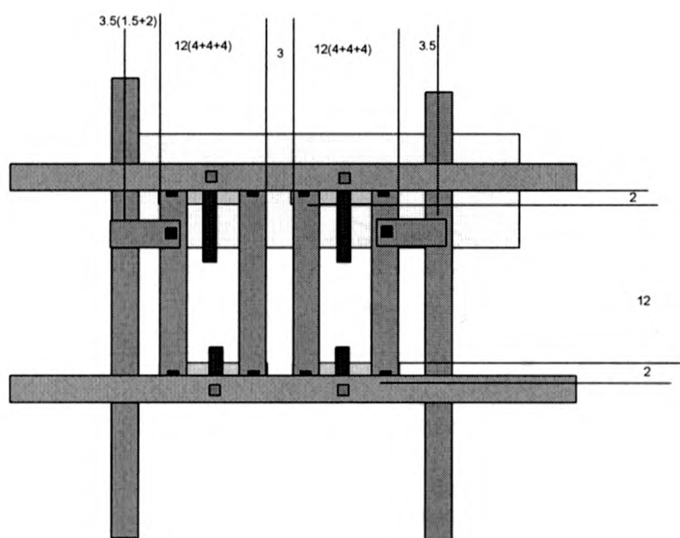


Figure 2.19: Layout of the proposed CMOL cell memory

2.9 Crossbar memory architecture:

A Molecule-nanowire crossbar memory structure is shown in Fig 2.20 for the generalize memory application with higher density. Write 1 and 0 operations may be done in a row at a time using single clock cycle. Control circuit shown in Fig 2.20 is able to pass the required 1.6V or -1.6V to the selected nanodevices to turn the devices ON or OFF. During the positive clock cycle 1.2V is passed through the top nanowire and 1.2V or -0.4V is passed through the bottom nanowire. The voltage across the nanodevices is either 1.6V (to turn the devices ON) or 0V (to leave the state of the devices unchanged). Turning OFF the nanodevice requires the application of voltages in reverse order (see Fig 2.20). Applying 0V to the unselected row's top nano-wires caused -1.2V or 0.4V voltage drop across the nanodevices. As these voltages are below the Coulomb blockade range, the unselected nanodevices will not be affected by the write operation.

Fig. 2.21 shows an example to write 11010011 in the first row of the memory along with the read operation.

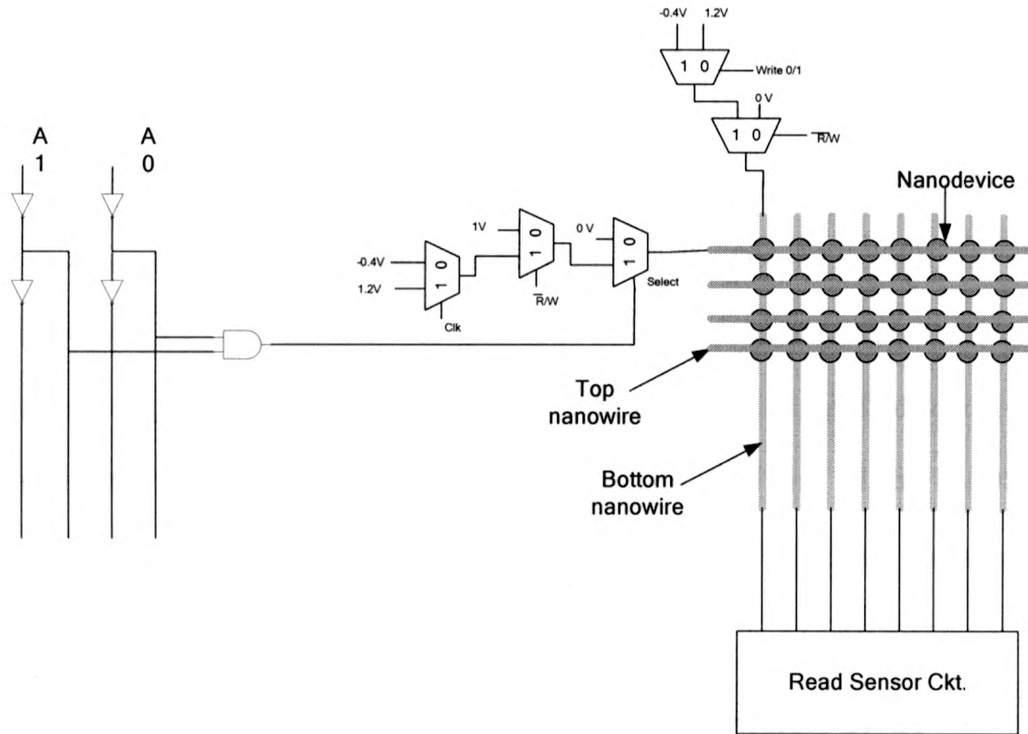


Figure 2.20: Nanowire-molecule-nanowire crossbar memory structure

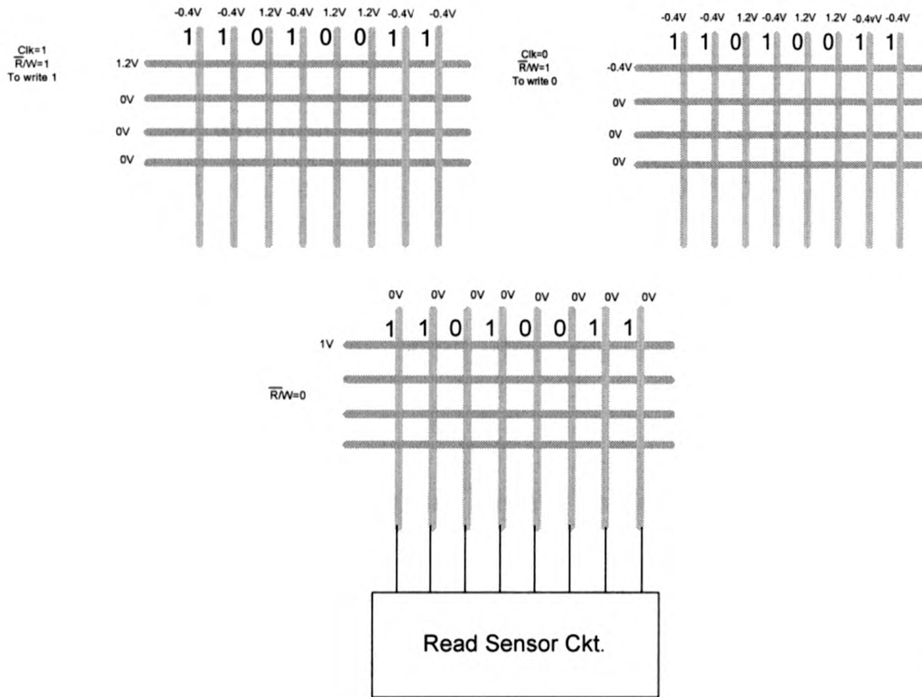


Figure 2.21: Operation on Crossbar molecular memory

2.10 The Need of improved nanodevice characteristics

Write and Read voltages of the nanodevices need to be reduced to the same levels predicted by the ITRS roadmap for sub 50nm technology. Assuming the future nanodevices coulomb blockade range will be -0.9V to 0.9V; ON, OFF and Read voltages were chosen as 1V, -1V and 0.5V so that CMOS 45nm technology can be used. Two references voltages V1 and V0 were chosen as 0.8V and -0.2V. Simplified control circuit with a single NMOS instead of two series NMOS was used and all the PMOS were chosen as 20λ width while NMOS width were 10λ. Table 2.8 shows the simulation results.

Table 2. 9: Simulated operational voltages for future device

Operation	Voltage Drop (V)	Power (W)
ON to OFF	-0.974	1.38*10^-6
OFF to ON	0.99	6.29*10^-7
Read with Ron	0.42	9.79*10^-7
Read with Roff	0.49	7.06*10^-7

Simulation results show the required voltage drop across the nanodevice. Proposed nanodevice reduce average power dissipation compare to the used BPDN-DT nanodevice.

2.11 Reconfiguration Algorithm for Defect tolerant Memory design

Using the same structure of the CMOL logic cell as memory, described reconfiguration algorithm in [14] is also applicable for the defect tolerant design of the CMOL memory. Generally, there may be many different algorithms to reconfigure the CMOL structure around known defects, including quasi-optimal, exhaustive-search

options which are impractical, the required resources for their implementation are exponential [14]. The key point of the linear reconfigurable algorithm in [14] is a sequential attempts to move each gate from a cell with bad input or/and output connections to a new cell, while keeping its input and output gates in fixed positions. At such a move, the gate may be swapped with another one, provided that all connections of the swapped gates can be realized with the CMOL fabric and are not defective. The algorithm calculate the 'repair region' of the gate, where it could be moved if there were no other cells around; this region is just the overlap of the connectivity domains (cell connected through a pin-nanowire-nanodevice-nanowire-pin link) of all its input and output cells. Fig. 2.22 shows the existing replacement algorithm. This algorithm can only be reconfigured in with a domain range. So it needs a large r to have more free cells to be used for repair. Large r always creates some new problems like increasing wire resistance, more chance to affect other non selected devices during write operation. Moreover reconfiguring the defective cells in a mostly used cells circuit needs more time as there is no tracking of the free cells locations. An improved reconfiguration algorithm is presented in the next section, keeping r small but successfully reconfigures the defective cell independently of its domain range.

```
INPUT:
  A) design_lst (list of gates mapped onto a perfect CMOL fabric, with entries holding positions of the
     initial cell and its input and output gates
  B) defect_pattern (location of defective bits; in our work simulated randomly)
  C) Width and geight of CMOL array, and parameter r

  START:
  1: Take the next cell (cur_gate) from design_list
  2: If cur_gate is not the end of the list {
  3: If some of the connections from cur_gate to its input and output gates are defective for
     defective_pattern {
  4:    Create a list (cur_candidate_list) through the following sequence of steps:
          (i) Based on location of inputs and output of the cur_gate, create the list of cells
             ("repair region") where cur_gate could be moved, if no other gates were involved
          (ii) In this list eliminate all the cells occupied by other gates which cannot be swapped with
          cur_cell
          (iii) Sort the cur_candidate_list by connection length penalty F
  5:   Take the next cell (candidate_cell) from cur_candidate_list
  6:    If candidate_cell is not the end of the list{
  7:       If all the connections from candidate_cell to cur_gate's input and output gates are
           defect-free for defec_pattern {
  8:           Move cur_gate into candidate_cell, exchanging it with the gate (if any) that
               occupied the cell
  9.          } Else {Go to step 7}
  10.        } Else {Exit without success}
  11.       }Else {Go to step 1}
  12.      } Else {Exit with success}
```

Figure 2.22: Existing replacement algorithm [10]

2.12 Improved Reconfiguration Algorithm

The existing algorithm is modified so that it may replace the defective with a free (unused) cell located outside of the domain region. Theoretical analysis shows that 100% yield can be achieved by the proposed replacement algorithm.

Step 1: Set priority =1; Find out the defective and free cells. Calculate the distance between the defective and free cells, then stored it in a table.

Step 2: Choose the free cells from the table based on priority. [priority =1, choose the least minimum distance from the defective to the free cell, priority =2, second minimum distance from the defective to free cell and so on.]

Step 3: Calculate the domain intersection region of the input and output cells of the defective cell; if the domain intersection region has any free cell go to step 4, otherwise go to step 5.

Step 4: Replace the defective cell with the free cell, decrease the total number of free cells by 1 and exit.

Step 5: Find a candidate cell (working cell) close to the free cell having less or equal distance than the previous distance between defective and free cell. The input and output cells of the candidate cell should have the free cell in their domain intersection region. If no candidate cell is available go to step 7.

Step 6: free cell = candidate cell, go to step 3

Step 7: If the circuit has unused cell, set priority = priority + 1, go to step 2. else print "No free cell is available to replace the defective cell" and exit.

Fig. 2.23 to Fig 2.28 describes this approach graphically. A total of 54 cells (2 horizontal cell= 1logic cell) are used to analysis the proposed algorithm. The defective cells marked as 'De' (Fig. 2.23) is the target cell to be replaced. Considering this cell as origin, the distance between the defective cell and all free cells are calculated. Cell marked as LD is least distance cell (priority 1) from the target cell.

Fig 2.23:

Since the 'LD' cell is not in the domain intersection region of the input-output cells of the target cells. So we need to first search another good cell which distance from the Target cell (De) is less than or equal to the previous distance (distance between 'De' and 'LD'). 'C1' and 'C' are the two probable candidate cells close to the target cell. Check which candidate cell is in the domain intersection region of the 'LD' cell.

Fig. 2.24:

Replace 'LD' by the candidate cell 'C1', as 'C1' is in the domain intersection region of the LD's input and output cells. Considering C1 as LD, check the possibility to replace the defective cell 'De' with the recent 'LD' cell.

Fig. 2.25:

New 'LD' cell is marked as replacement was not possible.

Fig 2.26:

As no candidate cell was available to mark as LD, choose a new free cell based on priority as LD.

Fig. 2.27 and 2.28:

Start all the steps again and replace the defective cell with a free cell.

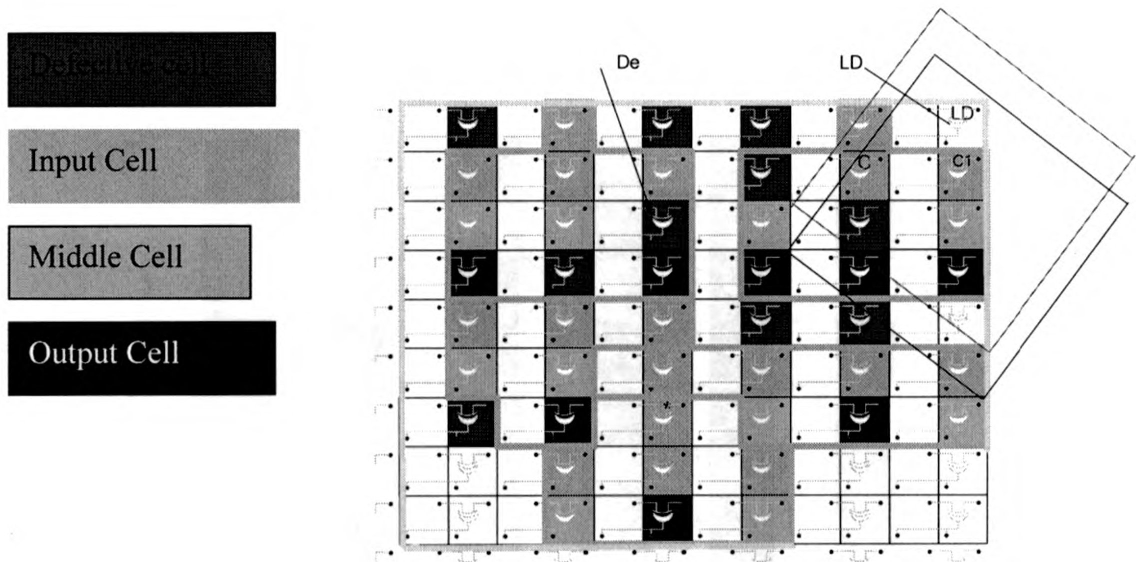


Figure 2. 23: Fixing the least distance cell and checking the domain range intersection

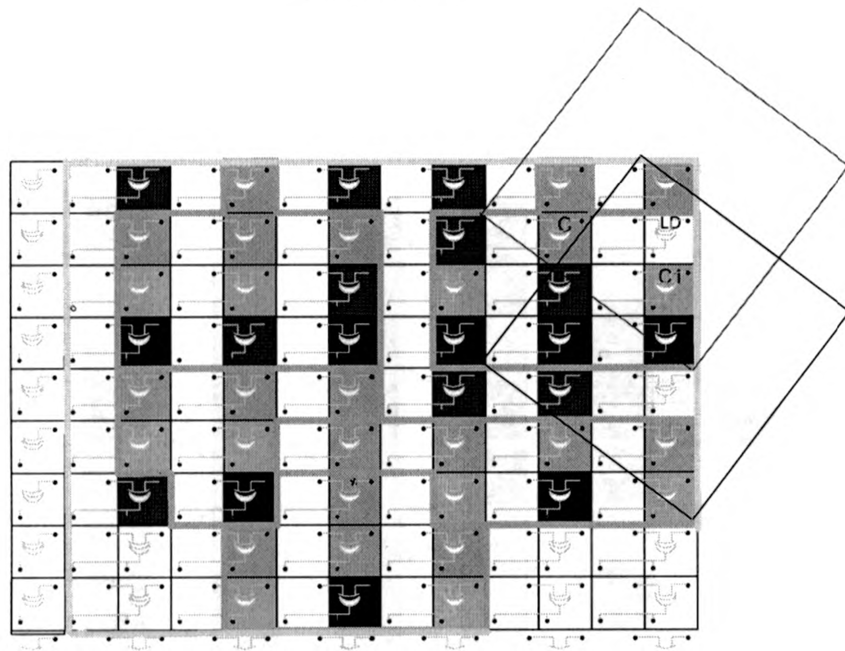


Figure 2.24: After checking the input output domain range intersection, shows the new LD location.

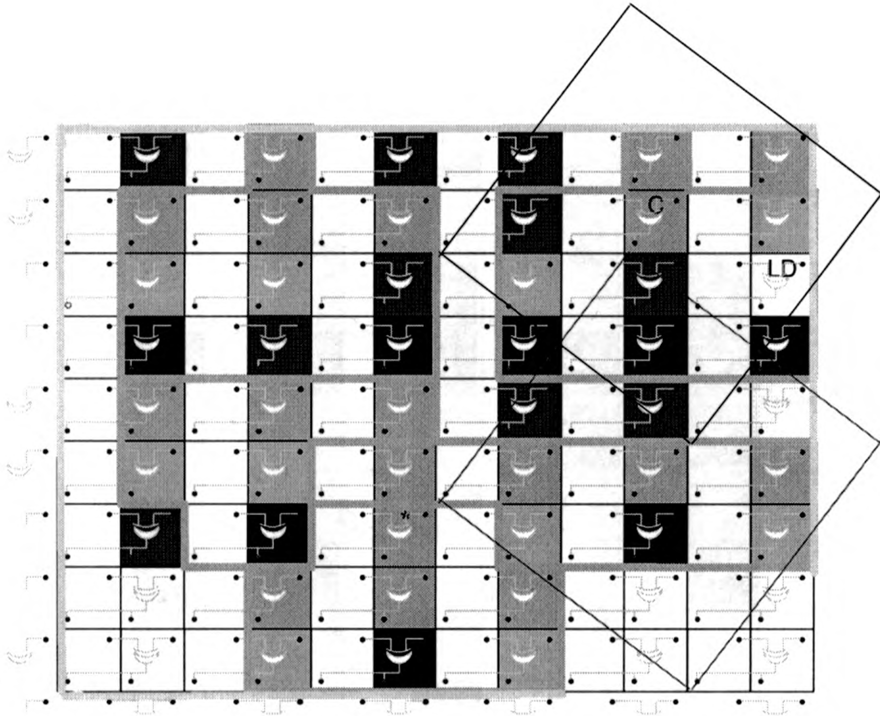


Figure 2.25: Since the replacement was not possible, a new cell is chosen as 'LD'

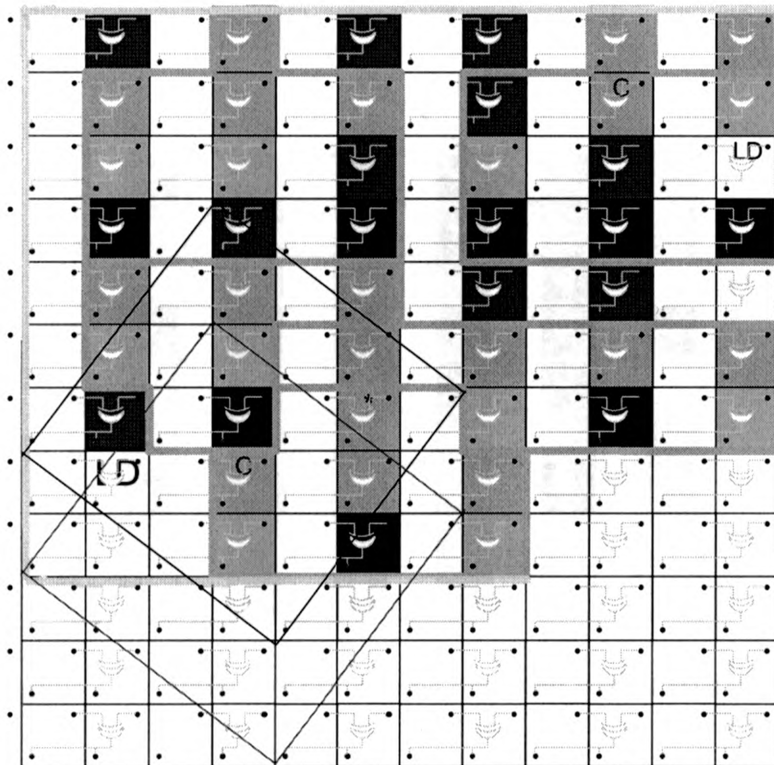


Figure 2.26: Choose a new free cell as no candidate cell was available

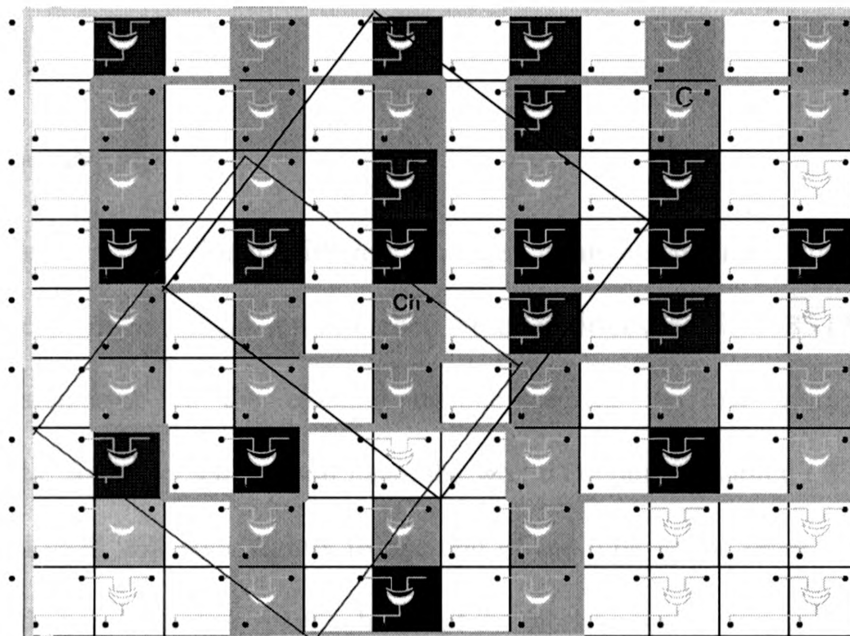


Figure 2.27: Start the procedure again for the new target cell

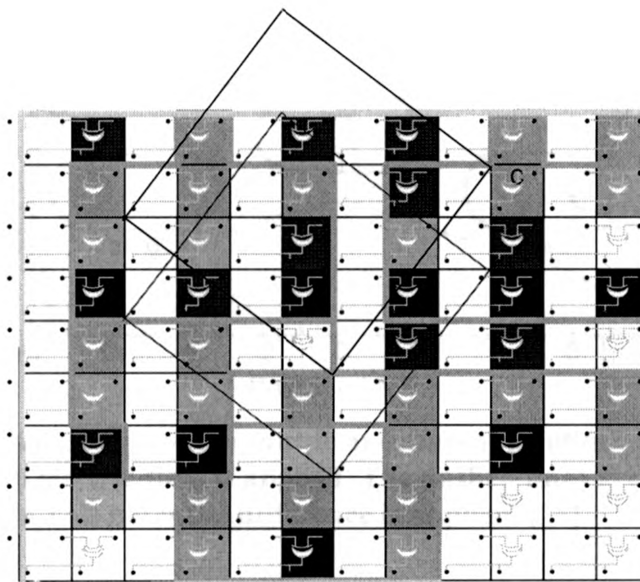


Figure 2.28: Defective cell can be replaced with a free cell

Above example shows that using the proposed algorithm a defective cell can be replaced by a free cell without depending on the domain region. All above examples are based on r=3.

2.13 Future Work:

Nanodevices operational with low voltages is an essential for future nanoscale memory. The Buckyball (C_{60}), a new carbon structure discovered in 1985 [24], may be a significant substitute of the existing nanodevice for its low voltage operation characteristics. A single-wall carbon nanotube would contain a charged (K+) buckyball and that buckyball will stick tightly to one end of the tube or the other. Assign the bit value of the device depending on which side of the tube the ball is (Fig. 2.29) [24].

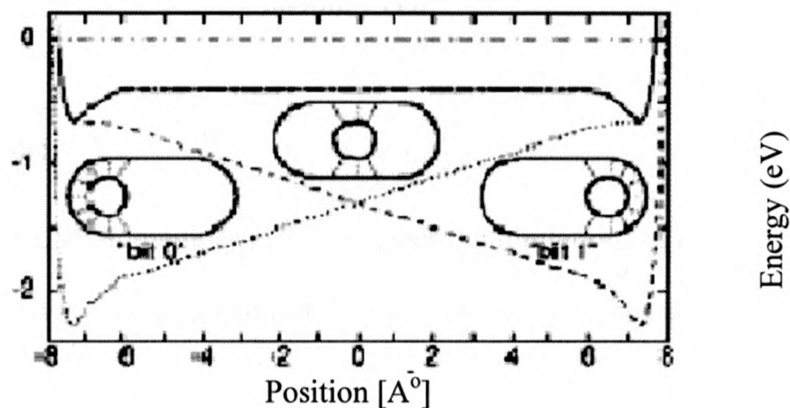


Figure 2.29: Potential energy of the shuttle at different locations in the capsule. The solid line is when no electric field is applied. The dashed lines are the potential energy when the two-volt potential difference is applied. [24]

In general the amount of voltage which needs to be applied depends upon the length of the capsule. A field of 0.1 volts/cm is sufficient to move the shuttle from one side of the tube to the other and the write speed is 20 picoseconds [24]. A new CMOL cell concept

structure with high speed access is shown in figure 2.30. Here the area is reduced by using 2 NMOS and 1 PMOS in a memory cell instead of 2 TG used in the proposed CMOL memory cell. Completion of this concept with proper simulation and applications are placed as a future works.

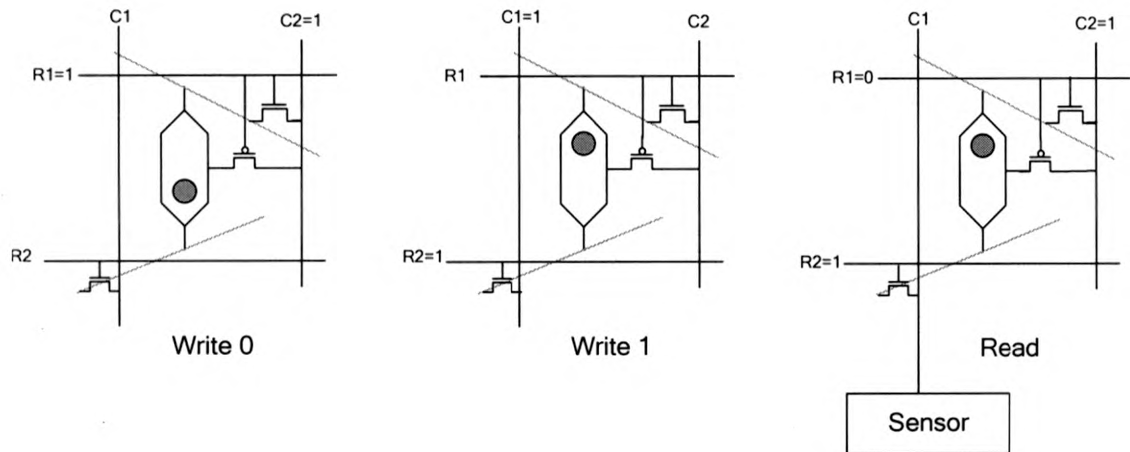


Figure 2.30: Write 0, 1 and read operation of a Buckyball-CNT based future memory cell.

2.14 Conclusion:

The proposed CMOL memory cell overcomes the limitations of current cell designs used for memory arrays. The optimized Control circuit can pass the proper voltages required across the nanodevice to turn it ON and OFF with 71% power saving. Area analysis shows that it is possible to fabricate CMOL logic and the proposed memory cells on a single chip. A suggested replacement algorithm increases the yield by replacing defective CMOL cells with working unused ones even if located outside the domain region. A high-speed, non-volatile bit of memory cell based on Buckyball (C_{60})-CNT is included as future works.

Chapter 3

AES: Literature Review

3.1 Introduction:

Cryptography (from Greek *kryptÃ³s*, "hidden", and *grÃ¡phein*, "to write") is generally understood to be the study of the principles and techniques by which information is converted into an encrypted version that is difficult (ideally impossible) for any unauthorized person to extract the original information, while still allowing the intended reader to do so[25]. In fact, cryptography covers rather more than merely encryption and decryption. It is, in practice, a specialized branch of information theory with substantial additions from other branches of mathematics. Cryptography is probably the most important aspect of communications security and is becoming increasingly important as a basic building block of computer security.

The increased use of computer and communications systems by the industry has increased the risk of theft of proprietary information. Although these threats may require

a variety of countermeasures, cryptography is a primary method of protecting valuable electronic information. In data and telecommunications, cryptography is necessary when communicating over any unsecured medium, which includes just about any network, particularly the Internet.

3.2 Information Security and Cryptography

Cryptography provides the mechanisms necessary to implement accountability, accuracy, and confidentiality in communications [26]. As demands for secure communication bandwidth grow, efficient cryptographic processing will become increasingly vital to good system performance. To introduce cryptography, an understanding of issues related to information security in general is necessary. Information security manifests itself in many ways according to the situation and requirement. Regardless of who is involved, to one degree or another, all parties to a transaction must have confidence that certain objectives associated with information security have been met. Over the centuries, an elaborate set of protocols and mechanisms has been created to deal with information security issues when the information is conveyed by physical documents. Often the objectives of information security cannot solely be achieved through mathematical algorithms and protocols alone, but require procedural techniques and abidance of laws to achieve the desired result. One of the fundamental tools used in information security is the signature. It is a building block for many other services such as non-repudiation, data origin authentication, identification, and witnessing, to mention a few. This signature is intended to be unique to the individual and serve as a means to identify, authorize, and validate. With electronic information the concept of a signature needs to be redressed; it cannot simply be

something unique to the signer and independent of the information signed. Electronic replication of it is so simple that appending a signature to a document not signed by the originator of the signature is almost a triviality.

Achieving information security in an electronic society requires a vast array of technical and legal skills. There is, however, no guarantee that all of the information security objectives deemed necessary can be adequately met. The technical means is provided through cryptography.

Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication [26]. Cryptography is not the only means of providing information security, but rather one set of techniques.

Cryptographic goals

The following four cryptographic goals form a framework from which other goals are derived:

Confidentiality is a service used to keep the content of information from all but those authorized to have it.

Data integrity is a service which addresses the unauthorized alteration of data.

Authentication is a service related to identification.

Non-repudiation is a service which prevents an entity from denying previous commitments or actions. When disputes arise due to an entity denying that certain actions were taken, a means to resolve the situation is necessary.

A fundamental goal of cryptography is to adequately address these four areas in both theory and practice. Cryptography is about the prevention and detection of cheating

and other malicious activities.

3.3 Symmetric-key Cryptography and AES

Symmetric-key cryptography, also called secret key cryptography, is the most intuitive kind of cryptography. It involves the use of a secret key known only to the participants of the secure communication. Symmetric-key cryptography can be used to transmit information over an insecure channel, but it has also other uses, such as secure storage on insecure media or strong mutual authentication. In symmetric-key cryptography, the key must be shared by both the sender and the receiver. The sender applies the encryption function using the key to the plaintext to produce the ciphertext. The ciphertext is sent to the receiver, who then applies the decryption function using the same shared key. Since the plaintext cannot be derived from the ciphertext without knowledge of the key, the ciphertext can be sent over public networks such as the Internet. Therefore, symmetric key cryptography is characterized by the use of a single key to perform both the encrypting and decrypting of data. Since the algorithms are public knowledge, security is determined by the level of protection afforded by the key. If the key is kept secret, both the secrecy and authentication services are provided. Secrecy is provided, because if the message is intercepted, the intruder cannot transform the ciphertext into its plaintext format. Assuming that only two users know the key, confidentiality is provided because only a user with the key can generate ciphertext that a recipient can transform into meaningful plaintext.

The United States standard for Symmetric-key cryptography, in which the same key is used for both encryption and decryption, is the Data Encryption Standard (DES) [26]. This is based upon a combination and permutation of shifts and exclusive OR operations

and so can be very fast when implemented directly on hardware (1GByte/s throughput or better) or on general purpose processors. DES uses a 56-bit key and maps a 64-bit input block of plaintext onto a 64-bit output block of ciphertext. The 56-bit key is a rather small key for today's computing power, the key size is indeed one of the most controversial aspects of this algorithm. The mainstream cryptographic community has long held that DES's 56-bit key is too short to withstand a brute-force attack from modern computers [27]. Computers have made it easier to attack ciphertext by using brute force methods rather than by attacking the mathematics [25]. With a brute force attack, the attacker merely generates every possible key and applies it to the ciphertext. Any resulting plaintext that makes sense offers a candidate for a legitimate key.

The current key size of 56 bits (plus 8 parity bits) of DES started to seem small, but the use of larger keys with triple DES (3DES) can generate much greater security. If security is the only consideration, then 3DES will be an appropriate choice for a standardized encryption algorithm for decades to come. However, the principle drawback of 3DES is that the algorithm is relatively sluggish in software. The original DES was designed for mid 1970s hardware implementation and does not produce efficient software code. 3DES, which has three times as many rounds as DES, is correspondingly slower. A secondary drawback is that both DES and 3DES use a 64- bit block size. For reasons of both efficiency and security, a larger block size is desirable. Because of these drawbacks, 3DES is not a reasonable candidate for long term use. As a replacement, NIST (National Institute of Standards and Technology) of USA in 1997 issued a call for proposals for a new Advanced Encryption Standard (AES), which would have security length equal to or better than 3DES and significantly, improved efficiency. In addition to these general

requirements, NIST specified that AES must be a symmetric block cipher with a block length of 128 bits and support key lengths of 128, 192, and 256 bits. The AES algorithm was selected in October 2001 after a multi-year evaluation process led by NIST with submissions and review by an international community of cryptography experts. The Rijndael algorithm [28], invented by Joan Daemen and Vincent Rijmen, was selected as the standard which was published in November 2002.

3.4 The AES Cipher

The Rijndael proposal for AES [29] defined a cipher in which the block length and the key length specified to be 128, 192, or 256 bits. The AES specification uses the same three key size alternatives but limits the block length to 128 bits. A number of AES parameters (Table 2.1) depend on the key length. Most of the implementation of AES uses the key length of 128 bits.

Table 3. 1: AES parameter

Key size (words/byte/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (word/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

Rijndael was designed to have the following characteristics:

- Resistance against all known attacks.
- Speed and code compactness on a wide range of platforms.
- Design simplicity.

3.5 Overall Structure of AES

The overall structure of AES is depicted in Figure 3.1. The input to the encryption and decryption algorithms is a single 128-bit block. This block of input is depicted as a square matrix of bytes. This block is copied into the state array, which is modified at each stage of encryption or decryption. After the final stage, state is copied to an output matrix. These operations are depicted in Figure 3.2. Similarly, the 128-bit key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words; each word is four bytes and total key schedule is 44 words for the 128-bit key. The ordering of bytes within a matrix is by column. So, for example, the first four bytes of a 128-bit plaintext input to the encryption cipher occupy the first column of the in matrix, the second four bytes occupy the second column, and so on. Similarly, the first four bytes of the expanded key, which form a word, occupy the first column of the 'w' matrix.

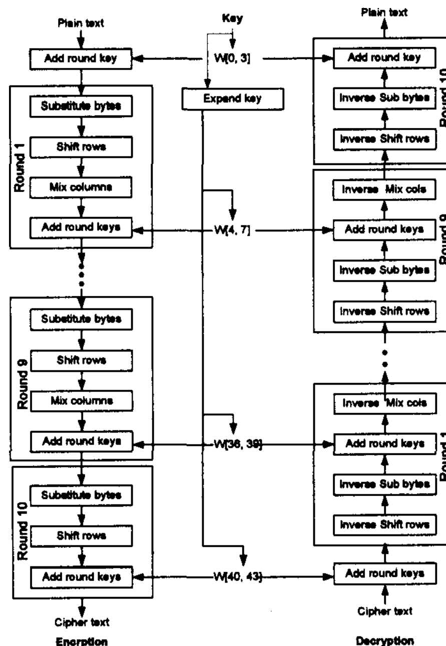


Figure 3. 1: General structure of AES [25]

In_0	In_4	In_8	In_{12}
In_1	In_5	In_9	In_{13}
In_2	In_6	In_{10}	In_{14}
In_3	In_7	In_{11}	In_{15}

→

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

→

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

→

Out_0	Out_4	Out_8	Out_{12}
Out_1	Out_5	Out_9	Out_{13}
Out_2	Out_6	Out_{10}	Out_{14}
Out_3	Out_7	Out_{11}	Out_{15}

a) Input, State Array, Output

k_0	k_4	k_8	k_{12}
k_1	k_5	k_9	k_{13}
k_2	k_6	k_{10}	k_{14}
k_3	k_7	k_{11}	k_{15}

→

w_0	w_1	w_2		w_{42}	w_{42}

b) key and expanded key

Figure 3. 2: AES Data Structures [25]

Several features of the overall AES structure [30]:

1. One noteworthy feature of this structure is that it is not a Feistel structure. In the classic feistel structure, half of the data block is used to modify the other half of the data block, and then the half are swapped. Rijndael does not use a Feistel structure but process the entire block in parallel during each round using substitutions and permutations.

2. The key that is provided as input is expanded into an array of forty-four 32-bit words. Four distinct words (128 bits) serve as a round key for each round; these are indicated in Figure 3.1.

3. Four different stages are used, one of permutation and three of substitution:

Substitute bytes: Uses an S-box to perform a byte-by-byte substitution of the block

Shift rows: A simple permutation

Mix Columns: A substitution that makes use of arithmetic operations over GF (2^8)

Add round Key: A simple bitwise XOR of the current block with a portion of the expanded key

4. The structure of AES is quite simple. For both encryption and decryption, the cipher begins with an Add Round Key stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages. Figure 3.3 depicts the structure of a full encryption round.

5. Only the Add Round Key stage makes use of the key. For this reason, the cipher begins and ends with an Add Round Key stage. Any other stage, applied at the beginning or end, is reversible without knowledge of the key and so would add no security.

6. The Add Round key stage is a simple bitwise XOR operations. The other three stages together provide confusion, diffusion, and nonlinearity, but by themselves would provide no security because they do not use the key. The cipher is an alternating operations of XOR encryption (Add Round Key) of a block, followed by scrambling of the block (the other three stages), and followed by XOR encryption, and so on. This scheme is both efficient and highly secure.

7. Each stage is easily reversible. For the Substitute Byte, Shift Row, and Mix Columns stages, an inverse function is used in the decryption algorithm. For the Add Round Key stage, the inverse is achieved by XORing the same round key to the block, using the result that

$A \oplus A \oplus B = B$.

8. As with most block ciphers, the decryption algorithm makes use of the expanded key in reverse order. However, the decryption algorithm is not identical to the encryption algorithm. This is a consequence of the particular structure of AES.

9. Once it is established that all four stages are reversible, it is easy to verify that decryption does recover the plaintext. Figure 3.1 lays out encryption and decryption

going in opposite vertical directions. At each horizontal point (e.g., the dashed line in the figure), **State** is the same for both encryption and decryption.

10. The final round of both encryption and decryption consists of only three stages. Again, this is a consequence of the particular structure of AES and it is required to make the cipher reversible.

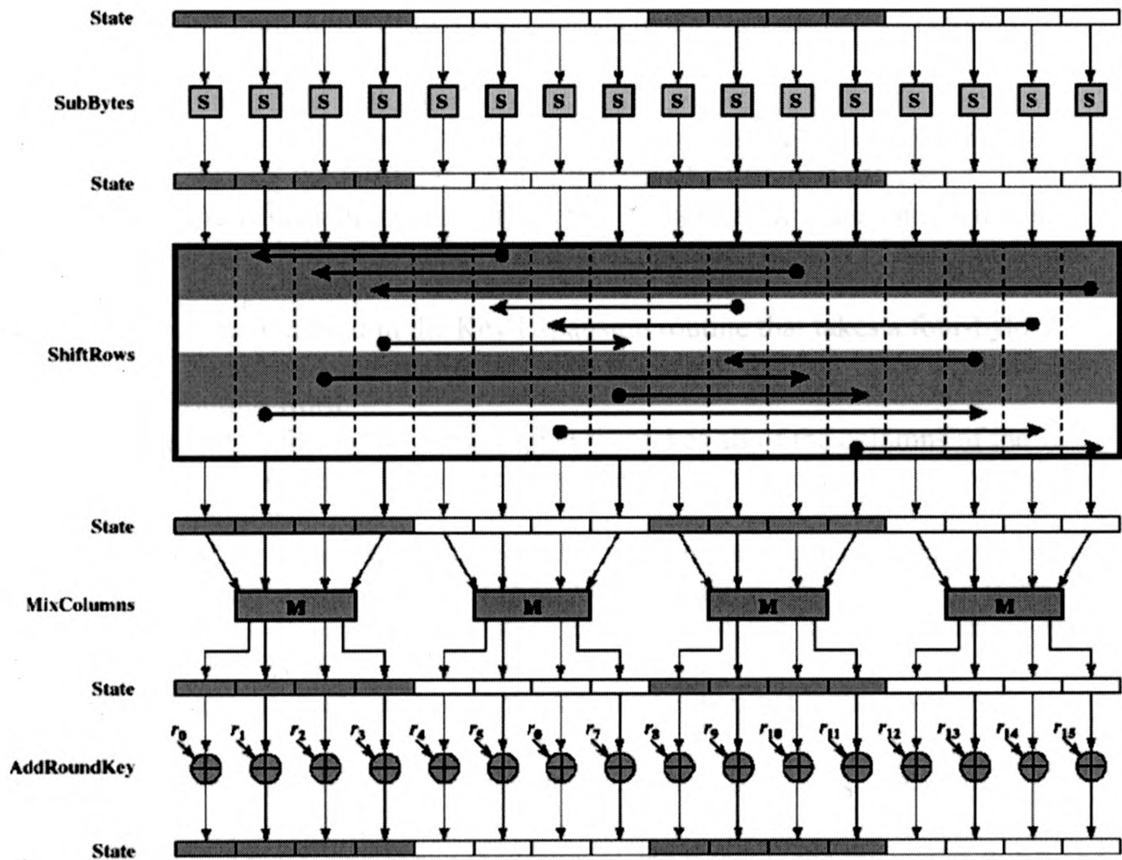


Figure 3. 3: AES Encryption round [25].

3.6 Definitions

Explanation of various used terms and symbols are descried in the next sub-sections.

3.6.1 Symbols, and Functions

The following symbols, and functions are used throughout this chapter:

Nb Number of columns (32-bit words) comprising the State. *Nb*=4

Nk Number of 32-bit words comprising the Cipher Key. For this standard, ***Nk*** = 4, 6, or 8.

Nr Number of rounds, which is a function of ***Nk*** and ***Nb*** (which is fixed). For this standard, ***Nr*** = 10, 12, or 14. (Also see Sec. 6.3.)

XOR Exclusive-OR operation.

$\oplus$ Exclusive-OR operation.

$\otimes$ Multiplication of two polynomials (each with degree < 4) modulo $x^4 + 1$.

• Finite field multiplication.

`ShiftRows` Transformation in the Cipher that processes the State by cyclically shifting the last three rows of the State by different offsets.

`SubBytes` Transformation in the Cipher that processes the State using a nonlinear byte substitution table (S-box) that operates on each of the State bytes independently.

`SubWord` Function used in the Key Expansion routine that takes a four-byte input word and applies an S-box to each of the four bytes to produce an output word.

`MixColumns` Transformation in the Cipher that takes all of the columns of the State and mixes their data (independently of one another) to produce new columns.

`AddRoundKey` Transformation in the Cipher and Inverse Cipher in which a Round Key is added to the State using an XOR operation. The length of a Round Key equals the size of the State (i.e., for *Nb* = 4, the Round Key length equals 128 bits/16 bytes).

`InvMixColumns` Transformation in the Inverse Cipher that is the inverse of **`MixColumns`**.

`InvShiftRows` Transformation in the Inverse Cipher that is the inverse of **`ShiftRows`**.

`InvSubBytes` Transformation in the Inverse Cipher that is the inverse of **`SubBytes`**.

3.7 Mathematical Preliminaries

All bytes in the AES algorithm are interpreted as finite field elements that can be added and multiplied, but these operations are different from those used for regular numbers. The following subsections introduce the basic mathematical concepts needed for the four stages that are used in AES.

3.7.1 Finite Fields

A field [31] is an algebraic object (collection of polynomial equations) with two operations: addition and multiplication, represented by " + " and " * ", although they will not necessarily be ordinary addition and multiplication. Using " + ", all the elements of the field must form a commutative group, with identity denoted by 0 and the inverse of each element is denoted by "-a". Using " * ", all the elements of the field except 0 must form another commutative group with identity denoted 1 and inverse element denoted by "a^{-1}". (The element 0 has no inverse under *). Finally, the distributive identity must hold: a*(b + c) = (a*b) + (a*c), for all field elements a, b, and c.

There are a number of different infinite fields, including the rational numbers (fractions), the real numbers (all decimal expansions), and the complex numbers. Cryptography focuses on finite fields. It turns out that for any prime integer p and any integer n greater than or equal to 1, there is a unique field with p^n elements in it, denoted $GF(p^n)$. [The "GF" stands for "Galois Field"]

In case n is equal to 1, the field is just the integers mod p, in which addition and multiplication are just the ordinary versions followed by taking the remainder on division by p. The only difficult part of this field is finding the multiplicative inverse of an element, that is, , finding a^{-1} of a non-zero element "a" in Z_p. This is the same as finding "a, b" such that a*b mod p = 1.

3.7.2 Addition

The addition of two elements in a finite field is achieved by "adding" the coefficients for the corresponding powers in the polynomials for the two elements. The addition is

performed with the XOR operation (denoted by ⊕) - i.e., modulo 2 - so that 1 ⊕ 1 = 0, 1 ⊕ 0 = 1, and 0 ⊕ 0 = 0.

Alternatively, addition of finite field elements can be described as the modulo 2 addition of corresponding bits in the byte. For two bytes $\{a_7a_6a_5a_4a_3a_2a_1a_0\}$ and $\{b_7b_6b_5b_4b_3b_2b_1b_0\}$, the sum is $\{c_7c_6c_5c_4c_3c_2c_1c_0\}$, where each $c_i = a_i \oplus b_i$ (i = 0,1,2.....7).

3.7.3 Multiplication

In the polynomial representation, multiplication in GF(2^8) (denoted by •) corresponds to the multiplication of polynomials modulo an irreducible polynomial of degree 8 [29]. This polynomial is irreducible if its only divisors are one and itself. For the AES algorithm, the mostly used irreducible polynomial is

$m(x) = x^8+x^4+x^3+x+1$ (3.1)

or {01}{1b} in hexadecimal notation.

For example,

{57} •{83} = {c1}, because

$(x^6+x^4+x^2+x+1)\ (x^7+x+1) = x^{13}+x^{11}+x^9+x^8+x^7+x^7+x^5+x^3+x^2+x+x^6+x^4+x^2+x+1$

$= x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1$

and

$x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1 \text{ modulo } (x^8+x^4+x^3+x+1) = x^7+x^6+1$

The modular reduction by $m(x)$ ensures that the result will be a binary polynomial of degree less than 8, and thus can be represented by a byte. Unlike addition, there is no simple operation at the byte level that corresponds to this multiplication.

The multiplication defined above is associative, and the element {01} is the multiplicative identity. For any non-zero binary polynomial $b(x)$ of degree less than 8, the

multiplicative inverse of $b(x)$, denoted $b^{-1}(x)$, can be found as follows: the extended Euclidean algorithm [32] is used to compute polynomials $a(x)$ and $c(x)$ such that

$$b(x)a(x) + m(x)c(x) = 1.$$

Hence, $a(x)\bullet b(x) \bmod m(x)=1$, which means

$b^{-1}(x)=a(x) \bmod m(x)$ (3.2)

Moreover, for any $a(x)$, $b(x)$ and $c(x)$ in the field, it holds that

$$a(x)\bullet(b(x)+c(x))= a(x)\bullet b(x)+a(x)\bullet c(x)$$

It follows that the set of 256 possible byte values, with XOR used as addition and the multiplication defined as above, has the structure of the finite field $GF(2^8)$.

3.8 SubBytes Transformation

The forward substitute byte transformation, called SubBytes is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box (Fig. 3.5), which is invertible, is constructed by composing two transformations:

1. Take the multiplicative inverse in the finite field $GF(2^8)$, the element {00} is mapped to itself.

2. Apply the following affine transformation (over GF(2)):

$$b_i^{'} = b_i \oplus b_{(i+4)\bmod 8} \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i \qquad (3.3)$$

for $0 \leq i < 8$, where b_i is the *i*-th bit of the byte, and c_i is the *i*-th bit of a byte *c* with the value {63} or {01100011}. Here and elsewhere, a prime on a variable (e.g., b') indicates that the variable is to be updated with the value on the right.

In matrix form, the affine transformation element of the S-box can be expressed as fig. 3.4:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Figure 3. 4: The effect of the SubBytes transformation on the State [29].

The S-box used in the **SubBytes** transformation is presented in hexadecimal form in Fig. 3.5. For example, if $s_{1,1}$={53}, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3' in Fig. 3.5. This would result in $s_{1,1}'$ having a value of {ed}.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 3. 5: S-box: substitution values for the byte xy (in hexadecimal format)[29]

3.9 InvSubBytes Transformation

InvSubBytes is the inverse of the byte substitution transformation, in which the inverse Sbox is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation followed by taking the multiplicative inverse in $GF(2^8)$.

The inverse S-box used in the InvSubBytes() transformation is presented in Fig. 3.6

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Figure 3. 6: Inverse S-box: substitution values for the byte xy (in hexadecimal format)[29]

3.10 ShiftRows Transformation

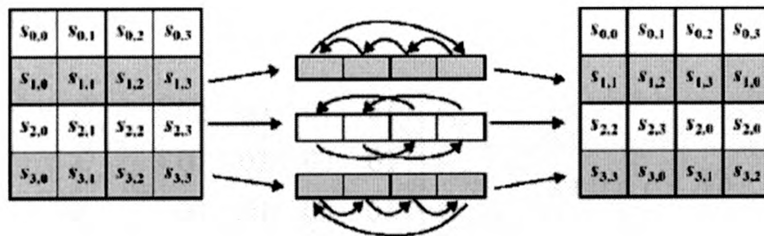


Figure 3. 7: Shift row transformation

The forward shift row transformation, called ShiftRows, is depicted in Figure 3.7. The first row of state is not altered. For the second row, a 1-byte circular left shift is

performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed. The following is an example of shiftRows:

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Figure 3. 8: Example of the ShiftRow operation

The inverse shift row transformation, called InvShiftRows, performs the circular shifts in the opposite direction for each of the last three rows, with a one-byte circular right shift for the second row, and so on.

3.11 MixColumns Transformation

The MixColumns transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ by a fixed polynomial $a(x)$, given by

$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$. (3.4)

this can be written as a matrix multiplication:

$s'(x) = a(x) \otimes s(x)$:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb.$$

The four bytes in a column are replaced by the following:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$
$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$
$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

Figure 3.9 illustrates the MixColumns transformation.

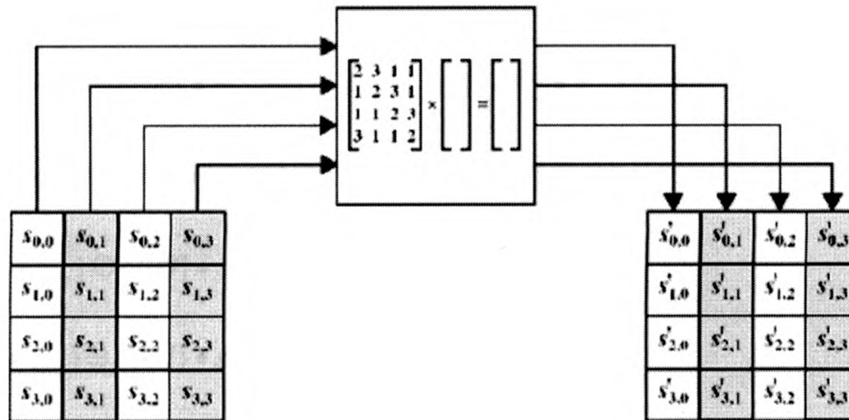


Figure 3. 9: MixColumns operates on the State column-by-column.

3.12 InvMixColumns Transformation

InvMixColumns is the inverse of the MixColumns transformation. InvMixColumns operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial

$a^{-1}(x)$, given by

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}. \quad (3.5)$$

this can be written as a matrix multiplication.

Let $s'(x) = a^{-1}(x) \otimes s(x)$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb.$$

As a result of this multiplication, the four bytes in a column are replaced by the following:

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$

$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$

$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

3.13 AddRoundKey Transformation

In the AddRoundKey transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of *Nb* words from the key schedule. Those ***Nb*** words are each added into the columns of the State, such that

$$[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}] = [S_{0,c}, S_{1,c}, S_{2,c}, S_{3,c}] \oplus [W_{round*Nb+c}] \text{ for } 0 \le c \le Nb$$

where $[w_i]$ are the key schedule words and *round* is a value in the range $0 \le \text{round} \le \text{Nr}$. In the Cipher, the initial Round Key addition occurs when *round* = 0, prior to the first application of the round function. The application of the AddRoundKey transformation to the *Nr* rounds of the Cipher occurs when $1 \le round \le Nr$. The action of this transformation is illustrated in Fig. 3.10, where $l = round * Nb$.

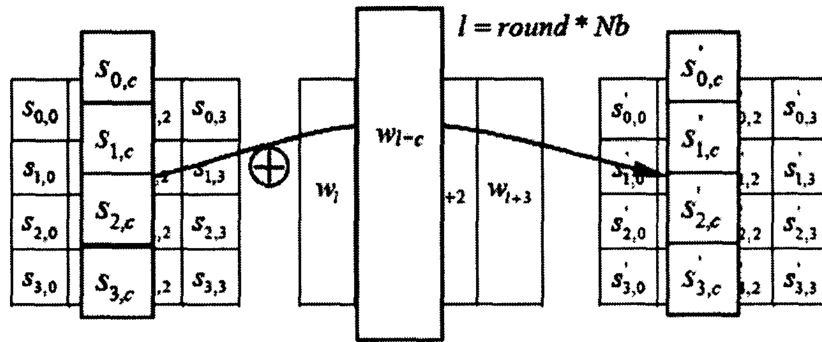


Figure 3. 10: AddRoundKey XORs each column of the State with a word from the key schedule.

The following is anther example of an AddRoundKey operation.

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$\oplus$

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D2

The first matrix is State, and the second matrix is the round key.

The inverse add round key transformation is identical to the forward add round key transformation, because the XOR operation is its own inverse. The add round key transformation is very simple and affects every bit of State. The complexity of the round key expansion, plus the complexity of the other stages of AES ensure security.

3.14 Key Expansion

The AES key expansion algorithm [29] takes as input a 4-word (16-byte) key and produces a linear array of 44 words (156 bytes). This is sufficient to provide a 4-word round key for the initial Add Round Key stage and each of the 10 rounds of the cipher. The following pseudocode describes the expansion:

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
        word temp
        i = 0
                while (i < Nk)
                 w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
                 i = i+1
                end while
        i = Nk
                while (i < Nb * (Nr+1)]
                  temp = w[i-1]
                     if (i mod Nk = 0)
                           temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
                        else if (Nk > 6 and i mod Nk = 4)
                           temp = SubWord(temp)
                     end if
                 w[i] = w[i-Nk] xor temp
                 i = i + 1
                end while
end
```

The Key Expansion generates a total of ***Nb*** (***Nr*** + 1) words: the algorithm requires an initial set of ***Nb*** words, and each of the ***Nr*** rounds requires ***Nb*** words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted [*wi*], with *i* in the range $0 \leq i < Nb(Nr + 1)$.

SubWord() is a function that takes a four-byte input word and applies the S-box to each of the four bytes to produce an output word. The function **RotWord()** takes a word

[$a0$,$a1$,$a2$,$a3$] as input, performs a cyclic permutation, and returns the word [a_1,a_2,a_3,a_0]. The round constant word array, **Rcon[i]**, contains the values given by [x^{i-1},{00},{00},{00}], with x^{i-1} being powers of x (x is denoted as {02}) in the field GF(28)(note that i starts at 1, not 0).

Chapter 4

SBox Circuit Implementation

4.1 Introduction:

The efficiency of an AES hardware implementation in terms of die-size, throughput, and power consumption is mainly determined by the implementation of the MixColumn-operation and the SBoxes. The remaining operations are trivial: ShiftRow is a simple cyclic shift, and AddRoundKey is an XOR-operation of the State and the RoundKey[33]. The exact number of SBoxes depends on the architecture's degree of parallelism and is determined by throughput requirements and the desired clock frequency. In case that the AES-module should also decrypt data, it has to be taken into account that the inverse SBoxes used for decryption have a different functionality. The number of SBoxes and their style of implementation have important influence on the size and the speed of AES hardware. For this reason, V. Rijmen (one of the AES inventors) suggested an alternative method for the computation of the AES-SBox [34]. It consists essentially of a replacement of the SBox lookup-table by an efficient combinational logic for the computation of the inverse elements in GF(2^8). Therefore, another representation of the

finite field GF(2^8) is used. This representation leads to an efficient implementation of the finite field arithmetic and was investigated in connection with the implementation of error correcting codes [35-38]. A polynomial representation of finite field elements is used in the design to have more flexible hardware architecture [33]. The main advantages of this architecture are:

- lower transistor count and die-size than a ROM-based approach,
- a short critical path to achieve a high operational frequency,
- easier implementation within a semi-custom design methodology since all computations can be done with standard-cells,
- flexibility for speed optimization: pipelining techniques can trade throughput for latency.
- suitability for a full-custom implementation: a few leaf-cells using an appropriate logic-style could increase speed or decrease power consumption.

4.2 Mathematical Review:

All mathematical operations required for the SBox are presented in this section.

4.2.1 GF (2^8) as an Extension of GF (2^4)

Usually, the field GF(2^8) is seen as a field extension of GF(2) and therefore its elements can be represented as Bytes. An isomorphic – but for hardware implementations far better suited – representation is to see the field GF(2^8) as a quadratic extension of the field GF(2^4). In this case, an element 'a' $\in$ GF(2^8) is represented as a linear polynomial with coefficients in GF(2^4),

$$a = a_h x + a_l, a \in GF(2^8), a_h, a_l \in GF(2^4) \quad (3.1) \quad [33]$$

and will be denoted by the pair $[a_h, a_l]$. Both coefficients of such a polynomial have four bits. All mathematical operations applied to elements of $GF(2^8)$ can also be computed in this representation which we call *two-term polynomials*. Two-term polynomials are added by addition of their corresponding coefficients

$$(a_h x + a_l) \oplus (b_h x + b_l) = (a_h \oplus b_h)x + (a_l \oplus b_l). \quad (3.2)$$

Multiplication and inversion of two-term polynomials require a modular reduction step to ensure that the result is a two-term polynomial too. The irreducible polynomial needed for the modular reduction is given by

$$n(x) = x^2 + \{1\}x + \{e\} \quad (3.3)$$

The coefficients of $n(x)$ are elements in $GF(2^4)$ and are written in hexadecimal notation. Their particular values are chosen to optimize the finite field arithmetic. The multiplication of two-term polynomials involves multiplication of elements in $GF(2^4)$ which requires an irreducible polynomial of degree 4 which is given by

$$m_4(x) = x^4 + x + 1. \quad (3.4) \quad [33]$$

Deriving formulas for multiplication in $GF(2^4)$ is similar to Byte-multiplication. Multiplication in $GF(2^4)$ is given by

$$q(x) = a(x) \otimes b(x) = a(x).b(x) \bmod m_4(x) \quad (3.5)$$

Where $a(x), b(x), q(x) \in GF(2^4), q(x) = q_3x^3 + q_2x^2 + q_1x^1 + q_0$

$$a(x) = a_3x^3 + a_2x^2 + a_1x^1 + a_0 \text{ and } b(x) = b_3x^3 + b_2x^2 + b_1x^1 + b_0$$

$$a_A = a_0 \oplus a_3, \quad a_B = a_2 \oplus a_3$$

$$q_0 = a_0b_0 \oplus a_3b_1 \oplus a_2b_2 \oplus a_1b_3 \qquad q_1 = a_1b_0 \oplus a_Ab_1 \oplus a_Bb_2 \oplus (a_1 \oplus a_2)b_3$$

$$q_2 = a_2b_0 \oplus a_1b_1 \oplus a_Ab_2 \oplus a_Bb_3 \qquad q_3 = a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_Ab_3$$

Squaring in GF(2^4) is a special case of multiplication and is given by

$$q(x) = a(x)^2 \bmod m_4(x) \qquad (3.6)$$

Where $a(x), q(x) \in GF(2^4)$

$$q(x) = q_3x^3 + q_2x^2 + q_1x^1 + q_0 \text{ and } a(x) = a_3x^3 + a_2x^2 + a_1x^1 + a_0$$

$$q_0 = a_0 \oplus a_2, \qquad q_1 = a_2, \qquad q_2 = a_1 \oplus a_3, \qquad q_3 = a_3,$$

The inverse a^{-1} of an element $a \in GF(2^4)$ can be derived by solving the equation $a(x) \cdot (a)^{-1}\ mod\ m_4(x) = 1$ as follows

$$q(x) = a(x)^{-1} \bmod m_4(x), \quad q(x), a(x) \in GF(2^4) \qquad (3.7)$$

Here $q(x) = q_3x^3 + q_2x^2 + q_1x^1 + q_0$ and $a(x) = a_3x^3 + a_2x^2 + a_1x^1 + a_0$

$$a_A = a_1 \oplus a_2 \oplus a_3 \oplus a_1a_2a_3$$

$$q_0 = a_A \oplus a_0 \oplus a_0a_2 \oplus a_1a_2 \oplus a_0a_1a_2$$

$$q_1 = a_0a_1 \oplus a_0a_2 \oplus a_1a_2 \oplus a_3 \oplus a_1a_3 \oplus a_0a_1a_3$$

$$q_2 = a_0a_1 \oplus a_2 \oplus a_0a_2 \oplus a_3 \oplus a_0a_3 \oplus a_0a_2a_3$$

$$q_3 = a_A \oplus a_0a_3 \oplus a_1a_3 \oplus a_2a_3$$

The concatenation of two bits a_ia_j in Equations (3.5) and (3.7) represents a binary multiplication which is an AND-operation. In contrast to inversion in GF(2^8), inversion in GF(2^4) is suitable for a hardware implementation using combinational logic since all Boolean equations depend only on four input bits.

4.2.2 Inversion of Two-Term Polynomials

Inversion of two-term polynomials is the equivalent operation to inversion in GF(2^8). A multiplication of a two-term polynomial with its inverse yields the 1-element of the field: $(a_h x + a_l) \otimes (a_h' x + a_l') = \{0\}x + \{1\}$, $a_h, a_l, a_h', a_l' \in GF(2^4)$

From this definition the formula for inversion can be derived:

$$(a_h x + a_l)^{-1} = a_h' x + a_l' = (a_h \otimes d)x + (a_h \oplus a_l) \otimes d \qquad (3.8)$$

$$d = ((a_h^2 \otimes \{e\}) \oplus (a_h \otimes a_l) \oplus a_l^2)^{-1} \qquad [33]$$

Inversion of two-term polynomials involves only operations in GF(2^4) which are suitable for a hardware implementation using combinational logic. Most of the functionality is used to calculate the term *d* which is used to calculate both coefficients of the inverted two-term polynomial.

4.2.3 Transformation between GF(2^8) and GF(2^4)

The finite field GF(2^8) is isomorphic to the finite field GF($(2^4)^2$) which means that for each element in GF(2^8) there exists exactly one element in GF($(2^4)^2$). The bijection from an element $a(x) \in GF(2^8)$ to a two-term polynomial $a_h x + a_l$ where $a_h, a_l \in GF(2^4)$ is given by the function *map*:

$$a_h x + a_l = map(a), \quad a_h, a_l \in GF(2^4), \quad a \in GF(2^8) \quad (3.9) \quad [33]$$

Here $a = a_7 x^7 + a_6 x^6 + a_5 x^5 + a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x^1 + a_0$,

$$a_h = a_{h3} x^3 + a_{h2} x^2 + a_{h1} x^1 + a_{h0}, a_l = a_{l3} x^3 + a_{l2} x^2 + a_{l1} x^1 + a_{l0}$$

$$a_A = a_1 \oplus a_7, \quad a_B = a_5 \oplus a_7, \quad a_C = a_4 \oplus a_6$$

$$a_{l0} = a_C \oplus a_0 \oplus a_5, \quad a_{l1} = a_1 \oplus a_2, \quad a_{l2} = a_A, \quad a_{l3} = a_2 \oplus a_4$$

$$a_{h0} = a_C \oplus a_5, \quad a_{h1} = a_A \oplus a_C, \quad a_{h2} = a_B \oplus a_2 \oplus a_3, \quad a_{h3} = a_B$$

The inverse transformation (map^{-1}) converts two-term polynomials $a_h x + a_l$, $a_h, a_l \in GF(2^4)$ back into elements a $\in$ $GF(2^8)$. It is given by

$$a = map^{-1}(a_h x + a_l), \quad a \in GF(2^8), \quad a_h, a_l \in GF(2^4) \quad (3.10) \quad [33]$$

$$a_A = a_{l1} \oplus a_{h3}, \quad a_B = a_{h0} \oplus a_{h1}$$

$$a_0 = a_{l0} \oplus a_{h0}, \quad a_1 = a_B \oplus a_{h3}$$

$$a_2 = a_A \oplus a_B, \quad a_3 = a_B \oplus a_{l1} \oplus a_{h2}$$

$$a_4 = a_A \oplus a_B \oplus a_{l3}, \quad a_5 = a_B \oplus a_{l2}$$

$$a_6 = a_A \oplus a_{l2} \oplus a_{l3} \oplus a_{h0}, \quad a_7 = a_B \oplus a_{l2} \oplus a_{h3}$$

4.3 SBox Building Blocks

The SubByte transformation operates independently on each Byte of the State using a substitution table (SBox). An AES SBox is composed of two transformations:

1. Calculate the multiplicative inverse in the finite field $GF(2^8)$. The element {00} is mapped to itself. Table 4.1 presents the inversion.

Table 4. 1: Inversion of Byte $\{xy\} \in GF(2^8)$ in hexadecimal notation

x\y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	00	01	8d	f6	cb	52	7b	d1	e8	4f	29	c0	b0	e1	e5	c7
1	74	b4	aa	4b	99	2b	60	5f	58	3f	fd	cc	ff	40	ee	b2
2	3a	6e	5a	f1	55	4d	a8	c9	c1	0a	98	15	30	44	a2	c2
3	2c	45	92	6c	f3	39	66	42	f2	35	20	6f	77	bb	59	19
4	1d	fe	37	67	2d	31	f5	69	a7	64	ab	13	54	25	e9	09
5	ed	5c	05	ca	4c	24	87	bf	18	3e	22	f0	51	ec	61	17
6	16	5e	af	d3	49	a6	36	43	f4	47	91	df	33	93	21	3b
7	79	b7	97	85	10	b5	ba	3c	b6	70	d0	06	a1	fa	81	82
8	83	7e	7f	80	96	73	be	56	9b	9e	95	d9	f7	02	b9	a4
9	de	6a	32	6d	d8	8a	84	72	2a	14	9f	88	f9	dc	89	9a
a	fb	7c	2e	c3	8f	b8	65	48	26	c8	12	4a	ce	e7	d2	62
b	0c	e0	1f	ef	11	75	78	71	a5	8e	76	3d	bd	bc	86	57
c	0b	28	2f	a3	da	d4	e4	0f	a9	27	53	04	1b	fc	ac	e6
d	7a	07	ae	63	c5	db	e2	ea	94	8b	c4	d5	9d	f8	90	6b
e	b1	0d	d6	eb	c6	0e	cf	ad	08	4e	d7	e3	5d	50	1e	b3
f	5b	23	38	34	68	46	03	8c	dd	9c	7d	a0	cd	1a	41	1c

2. Apply the affine transformation which is given by Equation 3.11.

$$q = \text{aff_trans}(a) \quad (3.11)$$

$a_A = a_0 \oplus a_1$, $a_B = a_2 \oplus a_3$, $a_C = a_4 \oplus a_5$, $a_D = a_6 \oplus a_7$

$q_0 = \bar{a}_0 \oplus a_C \oplus a_D$, $q_1 = \bar{a}_5 \oplus a_A \oplus a_D$, $q_2 = a_2 \oplus a_A \oplus a_D$, $q_3 = a_7 \oplus a_A \oplus a_B$

$q_4 = a_4 \oplus a_A \oplus a_B$, $q_5 = \bar{a}_1 \oplus a_B \oplus a_C$, $q_6 = \bar{a}_6 \oplus a_B \oplus a_C$, $q_7 = a_3 \oplus a_C \oplus a_D$

Decryption requires the inverse function of SubByte (*InvSubByte*) which reverses the SBox operation by applying the inverse affine transformation first (Equation 3.12). Then, the multiplicative inverse in the finite field $GF(2^8)$ is calculated.

$$q = \text{aff_trans}^{-1}(a) \quad (3.12)$$

$a_A = a_0 \oplus a_5$, $a_B = a_1 \oplus a_4$, $a_C = a_2 \oplus a_7$, $a_D = a_3 \oplus a_6$

$q_0 = \bar{a}_5 \oplus a_C$, $q_1 = a_0 \oplus a_D$, $q_2 = \bar{a}_7 \oplus a_B$, $q_3 = a_2 \oplus a_A$

$q_4 = a_1 \oplus a_D$, $q_5 = a_4 \oplus a_C$, $q_6 = a_3 \oplus a_A$, $q_7 = a_6 \oplus a_B$

Inversion in the finite field $GF(2^8)$ is needed to calculate the SubByte function as well as InvSubByte. It makes sense, to merge the SubByte with the InvSubByte in order to reuse the finite field inversion circuit (see Fig. 4.1).

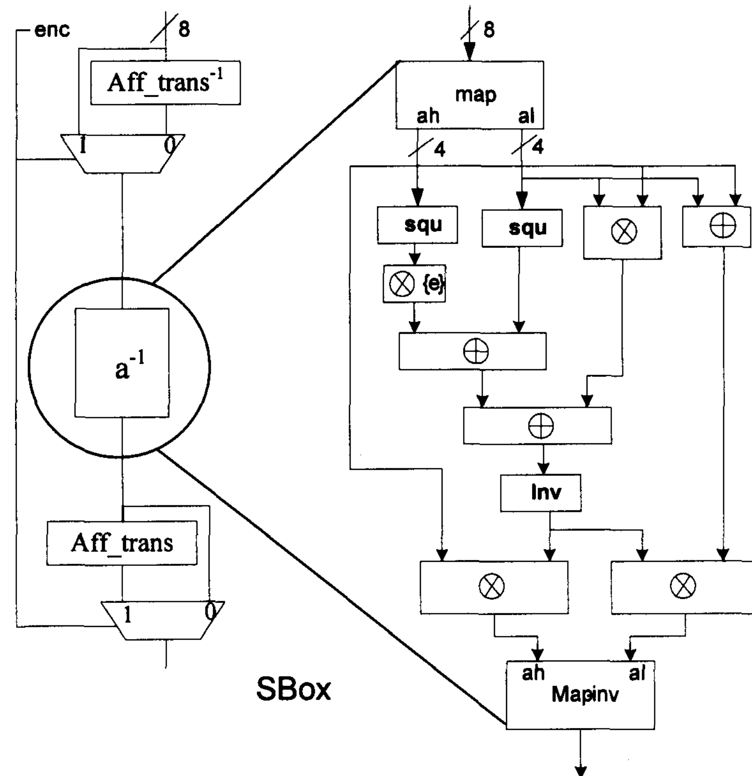


Figure 4. 1: Architecture of the AES SBox and InvSBox [33]

The control signal *enc* switches between encryption and decryption. If encryption is chosen (*enc* = 1), the inverse affine transformation (Aff_trans^{-1}) is bypassed and the input *a* is directly fed into the inversion circuit. The output of the inversion circuit is modified by the affine transformation block which calculates the result of the SubByte-function. During decryption (*enc* = 0), the inverse affine transformation is active and the affine transformation is bypassed to calculate InvSubByte. The delay for encryption and decryption is essentially the same because the circuit's complexity for the affine transformation and its inverse are equal (see Equation 3.11 and 3.12) .

The circuit for inversion of elements in $GF(2^8)$ covers most of the SBox functionality. In our approach the inversion is calculated with combinational logic and is based on Equation 3.8 which operates in $GF(2^4)$. The operations occurring in this equation

correspond to the function blocks shown in Fig. 4.1. Furthermore, this function block has to convert data from GF(2^8) to two-term polynomials and vice versa. The blocks *map* and *map*$^{-1}$ provide this functionality based on Equations (3.9) and (3.10) respectively. Addition of elements in GF(2^4) is accomplished by a bitwise XOR-operation. Squaring relies on (3.7). Multiplication of an element in GF(2^4) with the constant *{e}* is given by

$$q = q_3x^3 + q_2x^2 + q_1x^1 + q_0 = a \otimes \{e\} \qquad (3.13)$$

$$a_A = a_0 \oplus a_1, \quad a_B = a_2 \oplus a_3$$

$$q_0 = a_1 \oplus a_B, \qquad q_1 = a_A, \qquad q_2 = a_A \oplus a_2, \qquad a_3 = a_A \oplus a_B.$$

Multiplication and inversion in GF(2^4) are the most complex function blocks and rely on Equations 3.5 and 3.7.

4.4 SBox Design and Simulation

SBox implementation is based on the architecture described in Sect. 3.3. Most of the functionality can be implemented with XOR-gates. Fig. 4.2 shows the schematic diagram of the XOR gate that was used in the SBox.

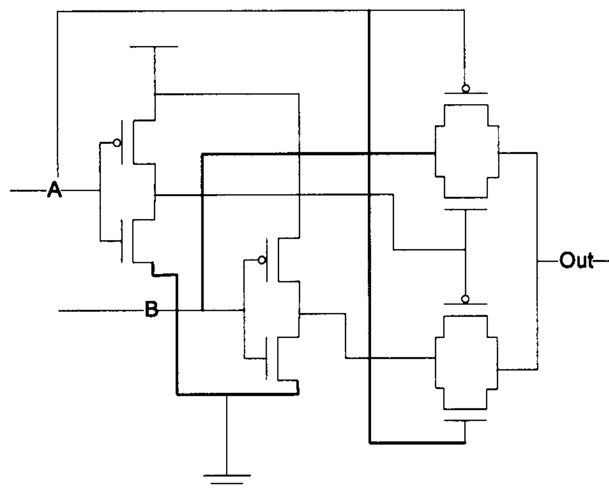


Figure 4. 2: Schematic diagram of an XOR gate

Schematic diagrams of the major function blocks needed for the AES-SBox are shown below:

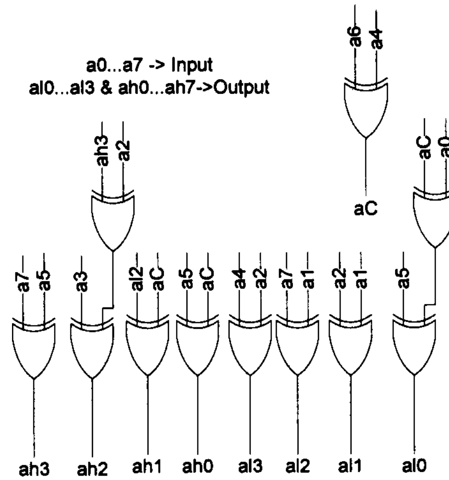


Figure 4. 3: Schematic diagram of "map" (Equation 3.9, critical path delay = 3 XOR)

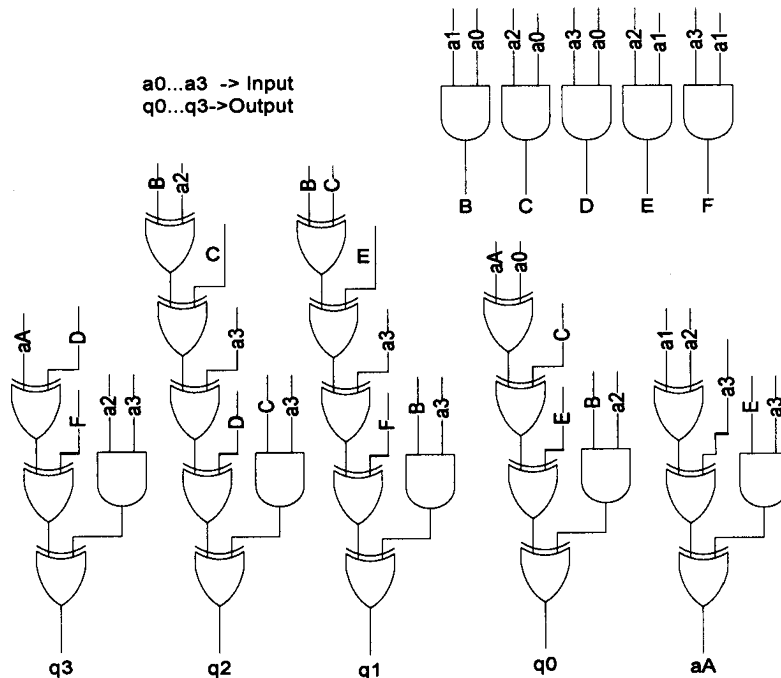


Figure 4. 4: Schematic diagram of the Inversion ('Inv' in Fig 4.1) in GF(2^4) (Equation 3.7, Critical Path Delay = 1 AND + 5 XOR, see Appendix Fig.1 for an improved design with 1 AND +4 XOR critical path delay)

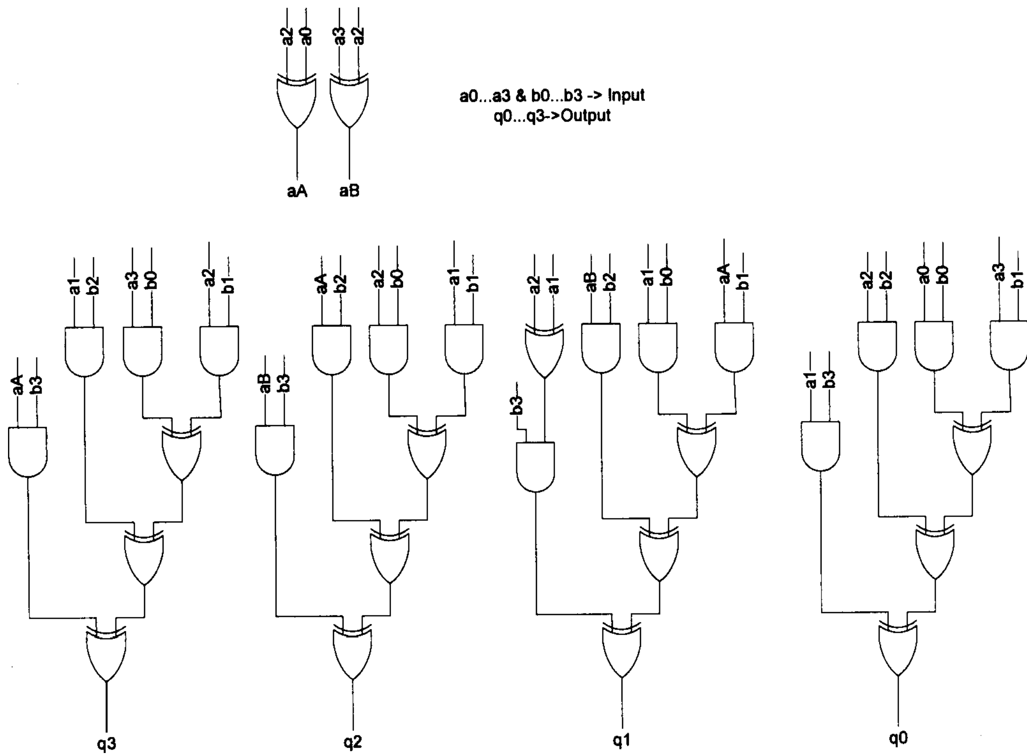


Figure 4. 5: Schematic diagram of Multiplication (⊗) (q = a ⊗ b)

(Equation 3.5, Critical Path Delay = 1 AND + 4 XOR, see Appendix Fig.2 for an improved design with 1 AND +3 XOR critical path delay)

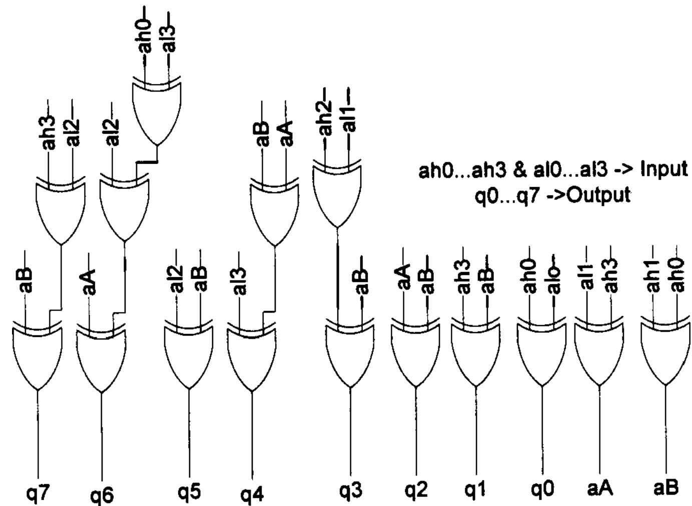


Figure 4. 6: Schematic diagram of Map_Inverse

(Equation 3.10, Critical Path Delay =3XOR)

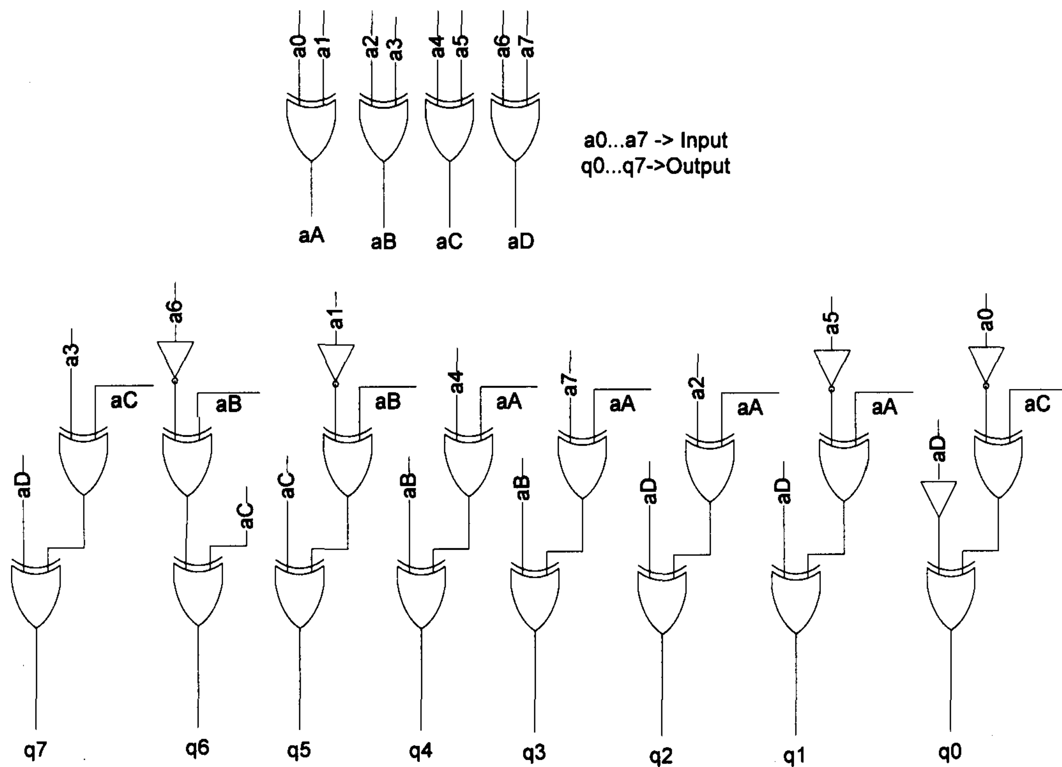


Figure 4. 7: Schematic diagram of Affine Transformation

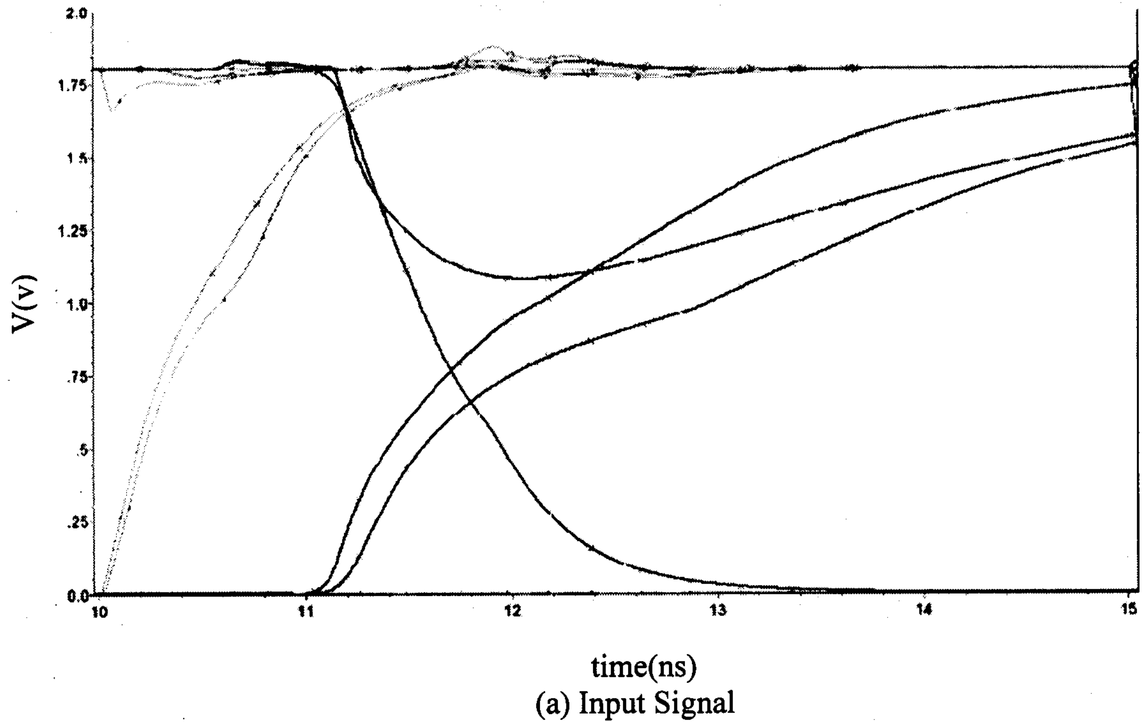
(Equation 3.11, Critical Path Delay=3XOR; See Appendix, Table 1 for details area and critical path delay measurements)

Before doing any kind of design optimization, four different input signals with the sequence of 8C, 62, 9D, 63 were chosen for testing the designed SBox circuit. Simulation was performed for a frequency of 200MHz. Table 4.2 shows the time delays and average power dissipation of the simulated SBox circuit implemented in 0.18μm CMOS technology.

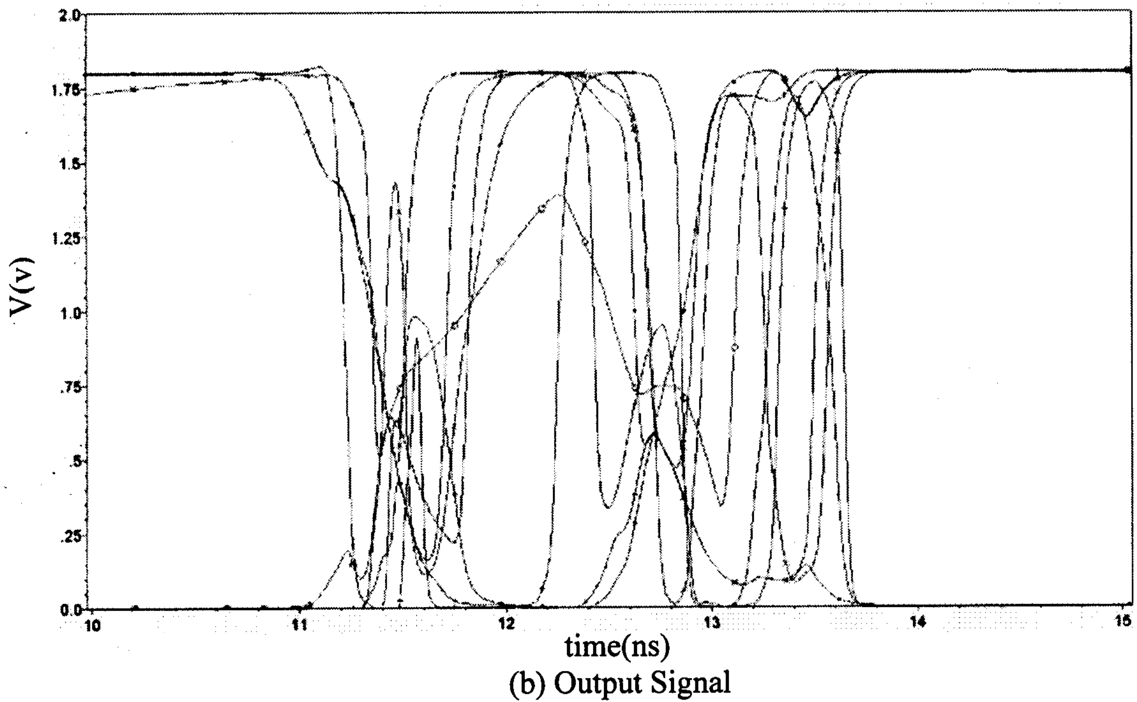
Table 4. 2: Simulation results of SBox for 4 input signals (Worst delay in bold)

Delay (ns)				Power
1st	2nd	3rd	4th	(mW)
2.88	3.403	**3.64**	3.54	1.568

Different number of gate level stages from the map to the multiplier (Fig. 4.1) have a significant impact on the output signals delay and the total average power.



(a) Input Signal



(b) Output Signal

Figure 4. 8: (a) Measured signals at inversion (INV)-multiplier(⊗) stages, Signals coming through the INV (bold lines) need more delay then others because of different gate levels; (b) SBox output signals show unwanted switching during the first 1ns (11ns – 12ns); It has a significant effect on the average power dissipation.

Figures 4.8 (a) and (b) show the signals at the output of the 'Inv' (see fig. 4.1) applied to the multiplier stages and the final output signals. Bold signals in Fig 4.8 (a) show output signals of the 'Inv' stage having more delay due to the longer data path. Another simulation was performed with eight input signals (85, E3, 94, F2, 0D, 6B, 1C, 7A) having a pulse width of 3ns to check the performance of the SBox circuit. These signals were chosen randomly having the property of most of the input bits change their values for each clock cycle. Table 4.3 shows the delays and average power of the simulation.

Table 4. 3: Output signals delay and power estimation of the SBox

Signals	1st	2nd	3rd	4th	5th	6th	7th	8th	Power (mW)
Delay ns	2.556	3.002*	2.836	2.64	2.336	**3.6 ***	3.3 *	2.8	2.477

['*' -> Delay is more than the input clock period]

Simulation results show the limitation of the designed SBox circuit by indicating that the operating can't exceed 0.3GHz.

One feasible solution is to add Tri-state Inverter followed by another inverter (TII) to transmit between the Inverter and Multiplier. This block will be enabled when all the signals coming from the different gate level stages are stable. By this addition, all signals can be transmitted to the multiplier stage at the same time eliminating glitches and consequently reducing power dissipation.

4.4.1 Design Modification

Fig. 4.9 (a) shows the schematic diagram of the tri-state inverter. When En=0 and En'=1, the output of the inverter is in a tri-state condition. When En=1 and En'=0, the Y output is equal to the complement of A. Fig. 4.9 (b) shows the tri-state inverter followed by

another inverter (TII) that is used for the design modification. The performance of the TII was checked to minimize the loading effect (In an average 3 logic gates (AND/XOR) are connect with an output signal) with the inverter size as NMOS width of 0.5μm and PMOS width of 1μm using 0.18μm CMOS technology.

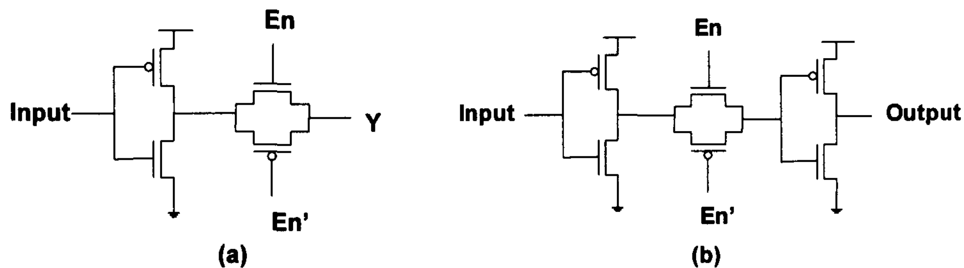
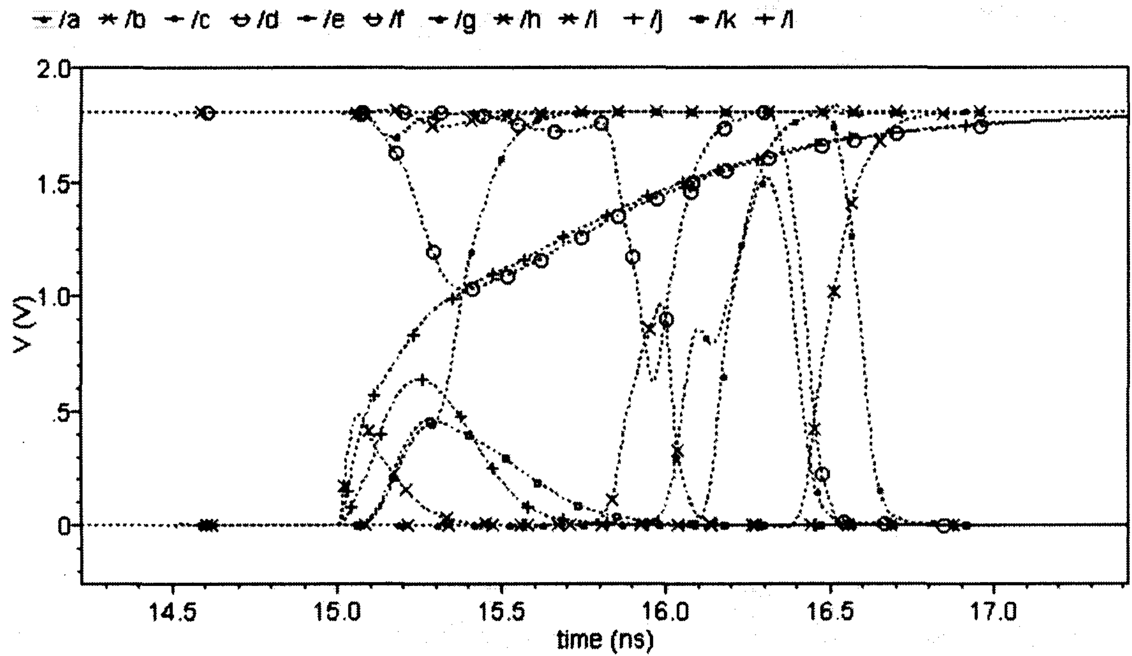
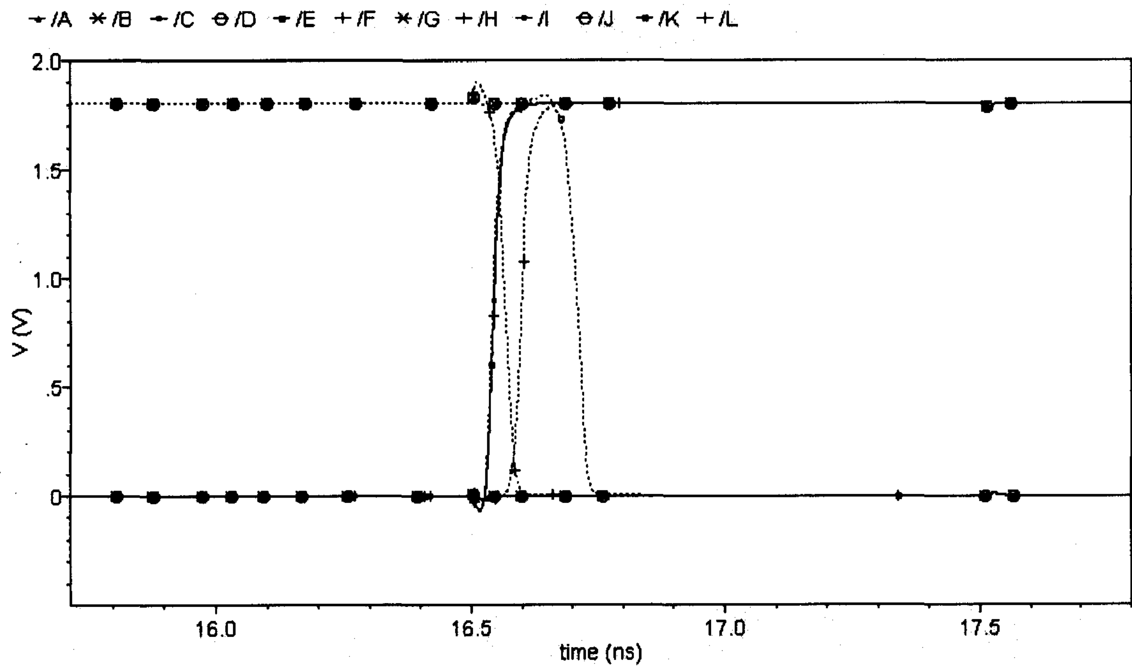


Figure 4. 9: (a) Schematic diagram of the tri-state inverter [23], (b) Tri-state inverter followed by another inverter (TII)

Fig 4.10 and Fig 4.11 show the input-output signals of the TIIs inserted between the 'Inv' and multiplier stages (Fig. 4.1), where (a) is the input signals and (b) is the output signals. Without any load, the modified design eliminated the false signals and reduced time delay (see Fig.4.10).



(a) Input Signal



(b) Output Signal

Figure 4. 10: (a)Input and (b)output signals of the TII without any load.

Whereas Fig. 4.11 shows that load affect on the output signals of the TIIs.

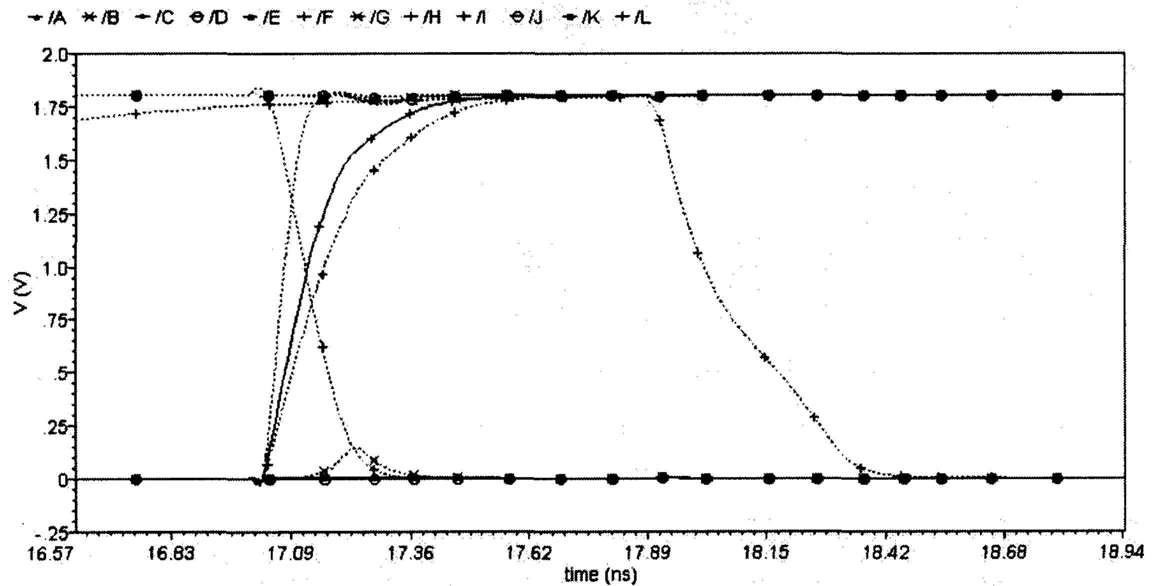


Figure 4. 11: Output signal of the TII with load.

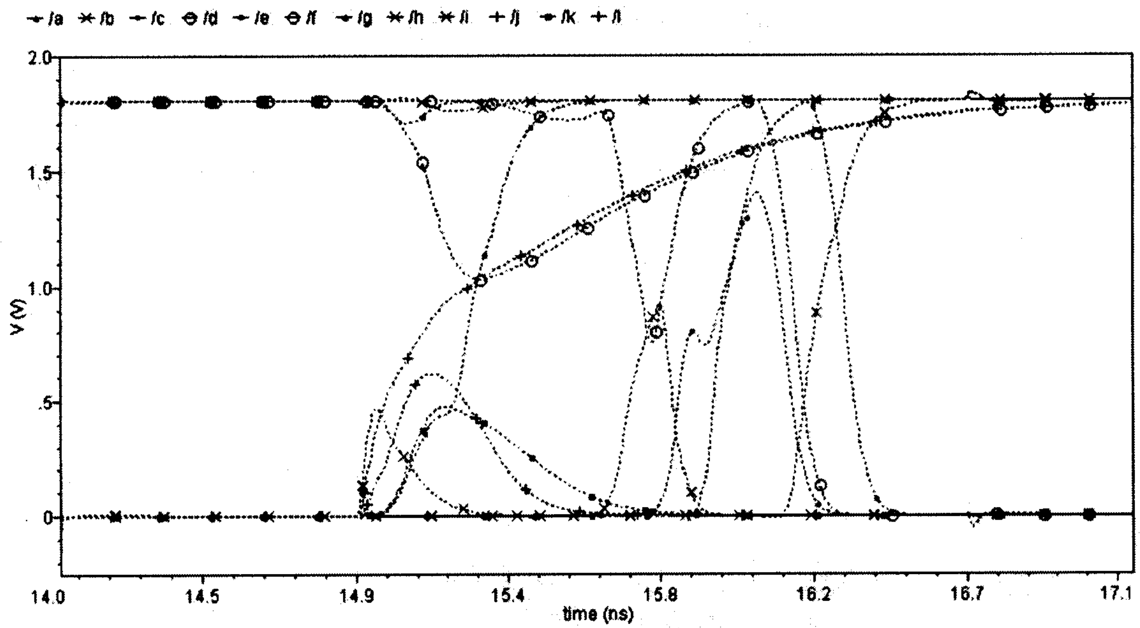
Transistors with larger width were used to minimize the loading effect. The following sizings were found suitable for an optimum design:

1st stage Inverter NMOS width is 0.7 μm, PMOS width is 1.5 μm

2nd stag Inverter NMOS width is 1.5 μm and PMOS width is 3μm

TG's NMOS width is 0.8 μm and PMOS width is 1μm

Changing the width of the transistors resulted in a significant reduction of the signals time delay (see Fig. 4.12).



(a) Input Signal

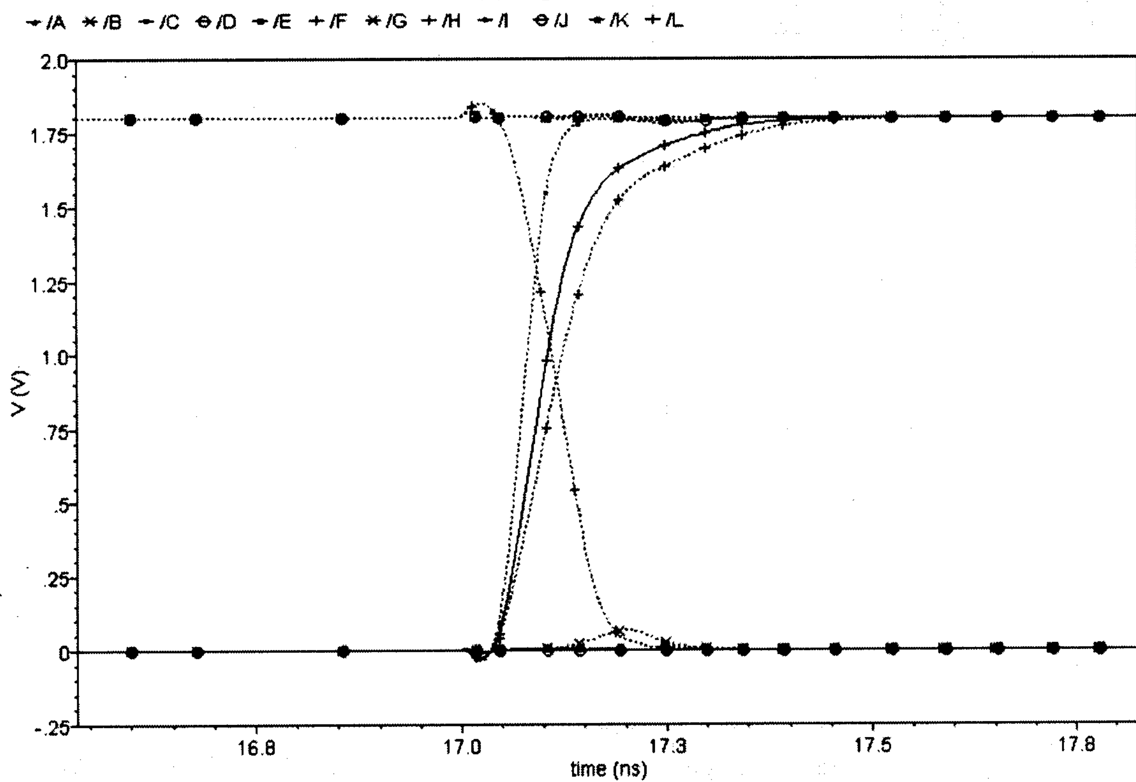


Figure 4. 12: Comparison of (a) input and (b) output signals of the wider TII with load.

4.4.2 Minimum power and delay investigation:

Eight input signals with the sequence of 85, E3, 94, F2, 0D, 6B, 1C, and 7A were chosen to investigate the minimum power and delay of the designed Sbox. All the input signals have 3ns pulse width and the simulation time period was 25ns using CMOS 0.18μm technology.

After applying the input signals to the SBox circuit the TII's enabled time and pulse width were varied to check the achievable minimum power. The TII's enabled time is more important then the pulse width to minimize the unwanted switching as well as total power dissipation. Simulation results (see Table 4.4) show that approximately 1.7 mW is a possible minimum average power dissipation. It was concluded that signals did arrive within 2ns period at the input of the TIIs stage. The third simulation (Table 4.4) was performed to find minimum power dissipation. Fig 4.13 shows the output signals of the first simulation (Table 4.4).

Table 4. 4: Simulated results with TII having input at every 3ns

TII Enabled	Delay (ns)								Power (mW)
	1st	2nd	3rd	4th	5th	6th	7th	8th	
2ns	3.176	**3.429**	2.906	3.11	3.16	3.04	3.09	3.19	1.702
2.5ns	Needs 0.5ns extra time than the 1st simulation								1.706
2.2ns	Needs 0.2ns extra time than the 1st Simulation								1.699

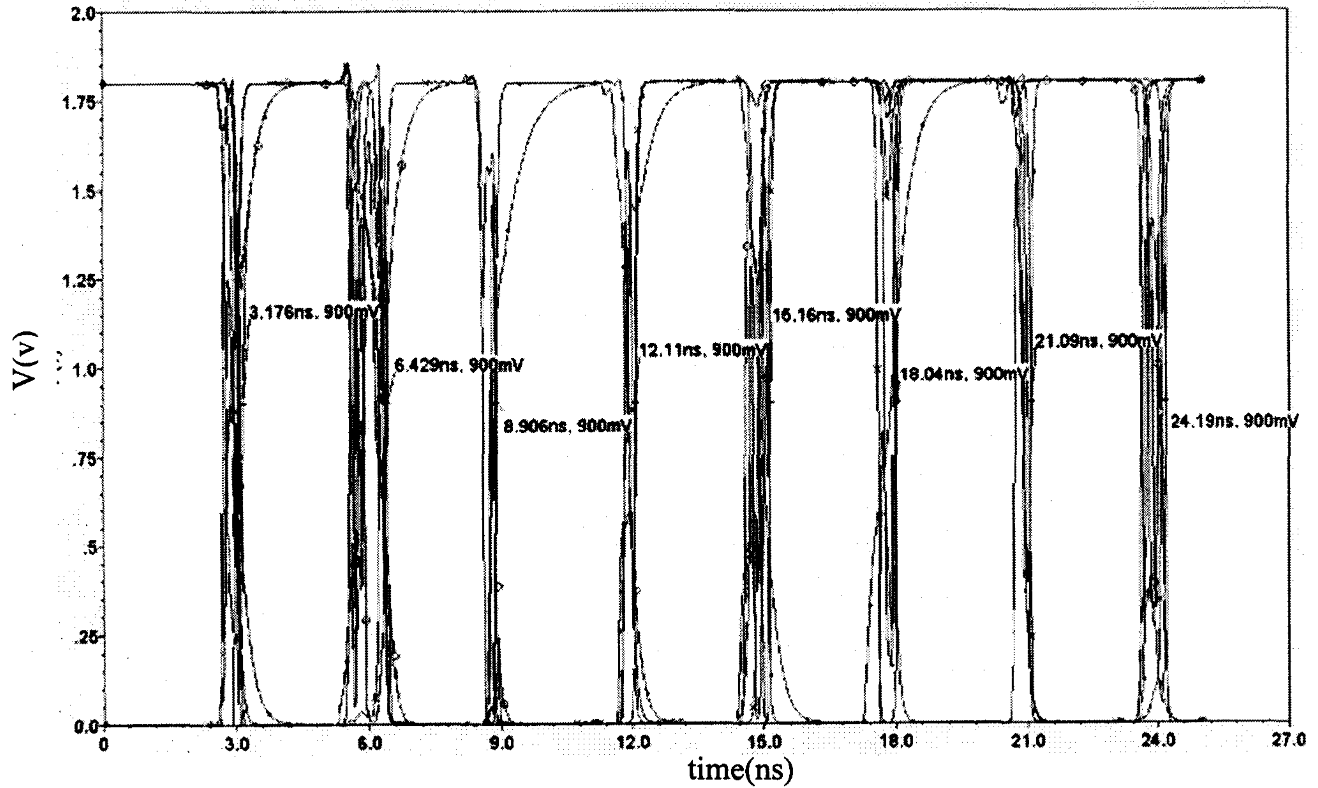


Figure 4. 13: Output signals with TII enabled after 2ns of the input signal

TIIs enabled time and clock pulse widths were chosen carefully by analyzing the signals arrival time at the TIIs stage. Table 4.5 shows the simulation result where 1.6ns is the maximum time delay for a signal to reach the TIIs stage.

Table 4. 5: Signals time delay measured at the input of the TII

TII Enabled	Clock pulse width	Delay (ns)							
		1st	2nd	3rd	4th	5th	6th	7th	8th
2ns	1ns	1	1.5	1.33	1.1	1.3	**1.6**	1.4	1

To check the minimum achievable time delay, another simulation was performed where TIIs were enabled after 1ns of the application of the input signals with the pulse width using 2ns. Table 4.6 shows the results.

Table 4. 6: Simulation result to check the minimum delay

TII Enabled	Clock pulse width	Delay (ns)								Power (mW)
		1st	2nd	3rd	4th	5th	6th	7th	8th	
1ns	2ns	2.22	2.64	2.15	2.19	2.63	2.63	**2.87**	2.49	2.12

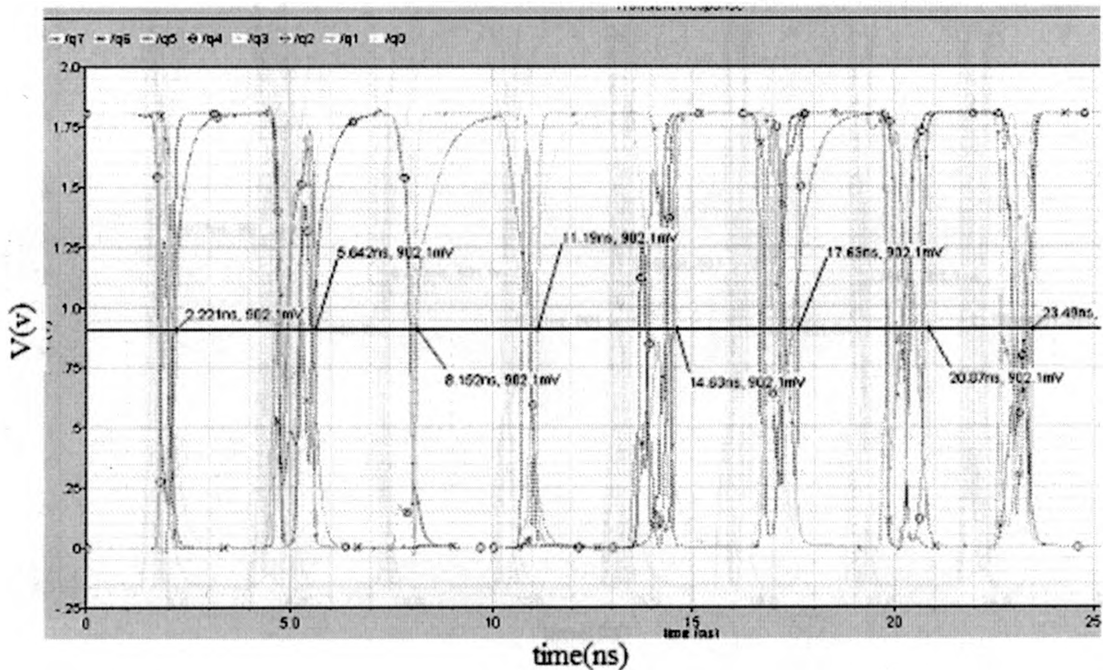


Figure 4. 14: Output signals of the simulation shown in table 4.6.

Fig. 4.14 shows the output signals of the table 4.5 simulation. To make the best combination of power and delay, another simulation was performed by enabling the TII after 1.5ns of the input signals and the clock pulse width was 1ns. Table 4.7 shows the result and Fig 4.15 shows the output signals.

Table 4. 7: Simulation results considering minimum time delay and power dissipation

TII Enabled	Clock pulse width	Delay (ns)								Power (mW)
		1st	2nd	3rd	4th	5th	6th	7th	8th	
1.5ns	1ns	2.677	**2.95**	2.413	2.61	2.66	2.65	2.76	2.78	1.808

With the above combinations, design modification reduced the average power dissipation from 2.477mW to 1.808mW (around 27%) and signals time delays were also minimized by 18% due to the use of TII.

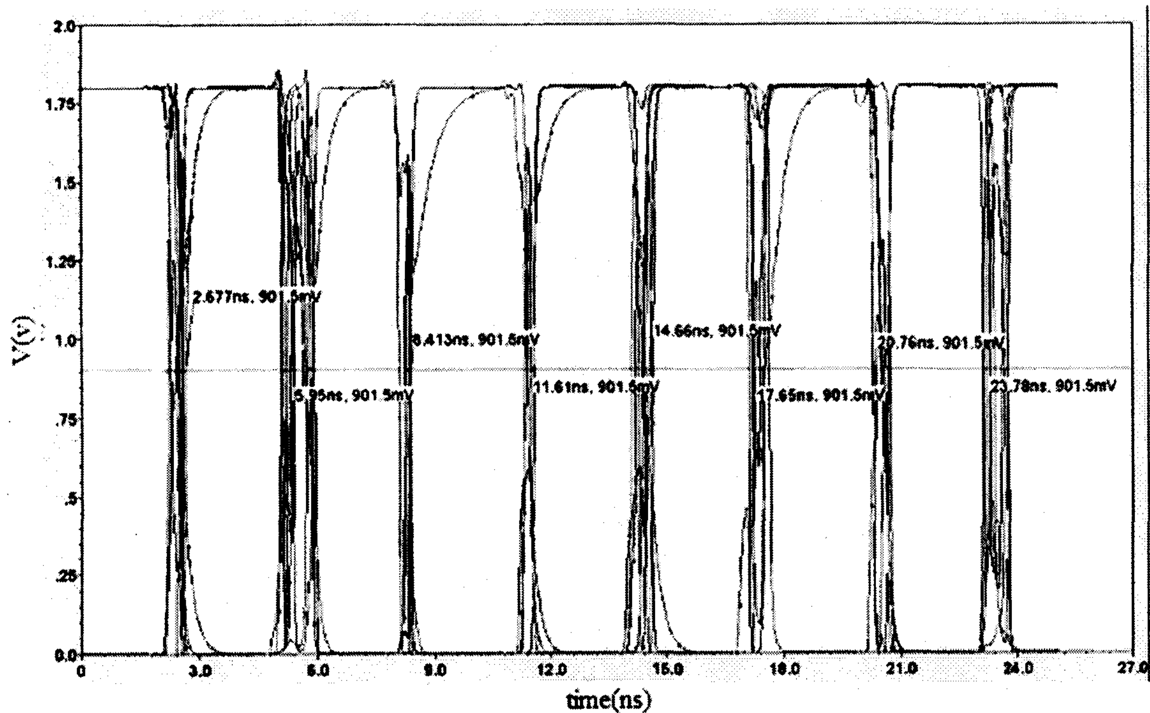


Figure 4. 15: Output signals of the simulation result shown in table 4.7

Among the different techniques used to reduce the power dissipation, lowering the supply voltages is one of the typically used methods. In the next subsection improvement on power dissipation by lowering the supply voltage is described.

4.4.3 Low Vdd for non-critical path circuits

Driving the circuit units of the non-critical path with a low Vdd is one of the ways to reduce the power dissipation without affecting the delays. To check the outcome of this

approach, one 4bit square ('Squ') and one 4bit XOR ('⊕') (see Fig 4.16) were operated at low V_{dd} in contrast to the other units.

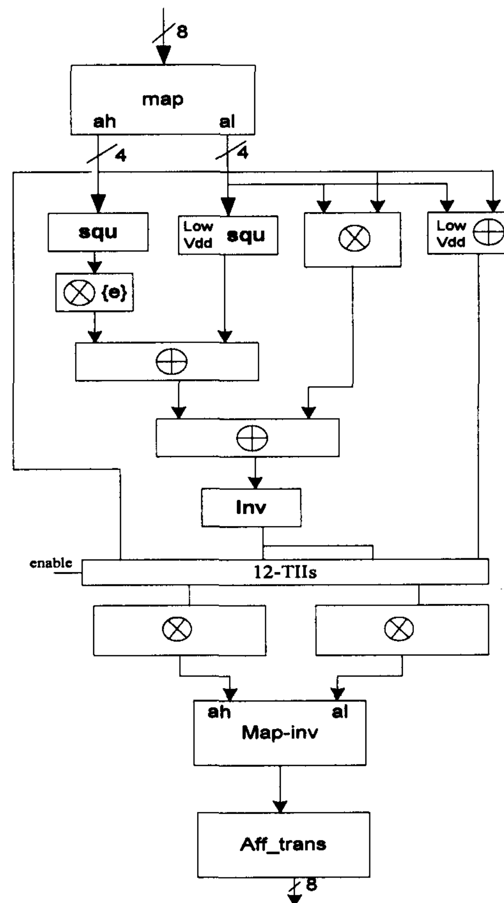


Figure 4. 16: Architecture of S-box with Low V_{dd} units

1.6V, 1.5V and 1.4V were chosen as Low V_{dd}. Table 4.8 shows the simulation results where energy * delay product shows the improvement by using low Vdd for the non-critical path. Among all results choosing 1.5V as the Vdd for non-critical path gave the best result with respect to Energy * Delay product.

Table 4. 8: Simulation result with Low V_{dd} at particular units

Voltage Source	Average Power Dissipation (W)	Total Power Dissipation (W)	Worst Delay	Energy * Delay
1.8 V	1.808 m	1.808 m	2.95 ns	15.73
1.8 V 1.7 V (low Vdd)	1.654 m 142.8 μ	1.796 m	2.955 ns	15.68
1.8 V 1.6 V (low Vdd)	1.658 m 120.9 μ	1.778 m	2.966 ns	15.64
1.8 V 1.5 V (low Vdd)	1.664 m 102.9 μ	1.766 m	2.97 ns	**15.57**
1.8 V 1.4 V (low Vdd)	1.669 m 96.5 μ	1.765 m	2.985 ns	15.72

4.4.4 Low V_{dd} for the all blocks:

It was proven that lowering the supply voltage decreases the total power dissipation at the expense of increasing the delay. Simulations were performed to check the minimum power that can be achieved by lowering the supply voltages. Table 4.9 shows the results of the simulations where 8 input signals were considered with a pulse width of 3ns:

Table 4. 9: Results with different V_{dd}

Vdd (V)	Power 1 (mW)	Power 2 (mW)	Power 1 * Delay	Energy * delay	Delay (ns)							
					1st	2nd	3rd	4th	5th	6th	7th	8th
1.8	1.808	1.751	5.335	15.74	2.677	**2.951**	2.413	2.61	2.66	2.65	2.76	2.78
1.7	1.563	1.51	4.889	**15.29**	2.767	**3.128**	2.485	2.8	2.73	2.93	2.97	2.95
1.6	1.369	1.322	4.606	15.49	2.88	**3.365**	2.596	2.98	2.96	3.26	3.25	3.11
1.5	1.219	1.176	4.485	16.50	3.024	3.419	2.668	3.06	3.28	**3.56**	3.68	3.45

The affect on the static power, by lowering the supply voltage, was also checked with the same sets of input signals having 7ns pulse width.

Table 4. 10: Performance analysis for the longer input signals

Vdd	Power 1 (μW)	Power 2 (μW)	Power * Delay	Energy * delay	Delay (ns)							
					1st	2nd	3rd	4th	5th	6th	7th	8th
1.8	834	785.6	2.461	7.262	2.677	**2.951**	2.41	2.6	2.66	2.65	2.76	2.78
1.7	733	677.4	2.294	**7.18**	2.767	**3.13**	2.48	2.79	2.73	2.94	2.99	2.95
1.6	642.9	591.4	2.166	7.299	2.88	**3.37**	2.6	2.97	2.96	3.27	3.27	3.1
1.5	568.1	525	2.079	7.606	3.024	3.42	2.67	3.02	3.28	3.56	**3.66**	3.48

Tables 4.9 and 4.10 show that lowering the voltage supply has a good impact on the power and the power-delay products at the expense of longer delays. Among all results choosing 1.7V as the supply voltage gave the best results with respect to Energy * Delay product.

However, the ratios of PMOS and NMOS (β) [β = Wn/Wp] of an inverter do influence the delay. So to have a lower switching threshold inverter in our XOR gates, another simulation was performed for β=0.5.

Table 4. 11: Simulation with β=0.5 (instead of β=1)

Vdd	Power 1 (mW)	Power 2 (mW)	Power * Delay	Delay (ns)							
				1st	2nd	3rd	4th	5th	6th	7th	8th
1.7	1.551	1.5	5.257	2.771	3.16	2.482	2.9	2.81	3.02	**3.39**	2.93
1.6	1.366	1.319	4.99	2.887	3.281	2.575	3.08	3.02	3.32	**3.66**	3.1
1.5	1.167	1.125		Wrong output							

Results from Table 4.11 show that due to the increased load capacitive charge/discharge time, reducing β from 1 to 0.5 is not an effective technique for our design.

4.5 Conclusion and Future Work

All the design components of the SBox are described briefly in this chapter. Simulations were performed to check the achievable minimum time delays and power dissipation. Modification of the design by adding a TII stages significantly reduced the average power by 27% and the delay by 18%. TIIs sizing were done after checking the loading effect. Optimum timing setup for the TIIs were checked to get the minimum delay and power dissipation. Lowering the supply voltage and applying dual voltage sources (high Vdd in the Critical path and low Vdd in the non-critical path) were performed to investigate the improvement on power dissipation. Choosing 1.7V as the Vdd gave the best compromise of lower power dissipation and higher time delays. More exhaustive testing with different set of inputs should be considered as a future work.

Chapter 5

Power reduction of SBox circuit in deep sub-micron CMOS technology

5.1 Introduction:

The evaluation of any system, in very deep sub-micron CMOS technology, depends on the level of power dissipation. Power consumption is directly related to heat dissipation and consequently the device temperature, which in turn have a significant impact on the system reliability. The bottom line is that lower power consumption generates direct benefits to the entire system in performance, cost, and quality.

ASIC implementation of the AES shows that 75% of the total power is consumed by the Subbyte (SBox) [8]. Reducing the SBox power dissipation is the key factor to design a low power AES circuit in deep submicron CMOS technologies. Power dissipation in CMOS circuits has both static and dynamic components. Due to the continuous scaling of CMOS technology towards nanometers dimensions and the reduction of threshold voltages, leakage power has become more and more significant. Table 5.1 shows the

components of the total power dissipation predicted by the ITRS for nanoscale CMOS technologies.

Table 5. 1: Power Predictions by ITRS[38]

Node	90nm	65nm	45nm
Dynamic Power per cm^2	1X	1.4X	2X
Static Power per cm^2	1X	2.5X	6.5X
Total Power per cm^2	1X	2X	4X

Static power caused by sub-threshold and gate currents is becoming more dominant than the dynamic power consumption (Table 5.1).

As transistor geometries are reduced, it is necessary to reduce the supply voltage to avoid electrical break down of gate oxide and to reduce the dynamic power dissipation. However, to retain or improve performance it is necessary to reduce the threshold voltage (V_t) as well, which result in exponential increase of sub-threshold leakage current. To control short channel effect and increase the transistor driving strength in deep sub-micron circuits, gate-oxide thickness also becomes thinner as technology scales down. The aggressive scaling in the gate-oxide results in tunneling current through the oxide, which is a strong exponential function of the oxide thickness and the voltage magnitude across the oxide [38].

Power optimized SBox circuit design in 65nm CMOS technology with an analysis of possible leakage current components and low power techniques are presented in the following sections.

5.2 Sources of leakage currents in CMOS circuits

There are five major sources of leakage currents (see Fig. 5.1) in CMOS transistors [1]

(A) Gate oxide tunneling leakage (I_G).

(B) Subthreshold leakage (I_{SUB}).

(C) Reverse-bias source/drain junction leakages (I_{REV}).

(D) Gate Induced Drain Leakage (I_{GIDL}).

(E) Gate current due to hot-carrier injection (I_H).

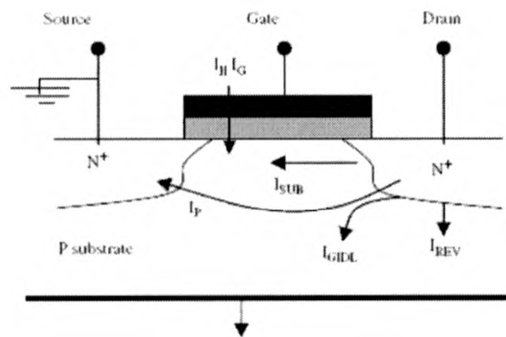


Figure 5. 1: Leakage current components in NMOS transistor [1].

In off-state, the main components of leakage current are sub-threshold leakage (I_{SUB}), gate induced drain leakage (I_{GIDL}) and gate tunneling leakage (I_G). In on-state, gate tunneling leakage(I_G) is the major component.

5.2.1 Gate oxide tunneling leakage (I_G)

The downscaling of the gate oxide thickness causes the electron tunneling from gate to substrate or from substrate to gate [39, 40]. The resulting current is called gate oxide tunneling current. With an increase in gate oxide thickness the tunneling drops exponentially, and it is given by

$$I_G = (A.C).(W.L)e^{-B\cdot\frac{T_{ox}}{V_{gs}}\cdot\alpha}$$

Where $A = q_3 / 8\pi h\phi_b$, $B = 8\pi\sqrt{2m_{ox}\phi_b^{3/2}} / 3hq$, $C = (V_{gs} / T_{ox})^2, \alpha$ is a parameter that ranges from 1 to 0.1 depending on the voltage drop across the oxide. H is Planck's constant, and ϕ_b is the barrier height of electrons/holes at the oxide/semiconductor interface [41].

The gate tunneling leakage current has been increasing by more than twice the subthreshold leakage current in nanometer CMOS technology [41]. Figure 5.2 shows the gate tunneling leakage currents in a nanoscale NMOS transistor.

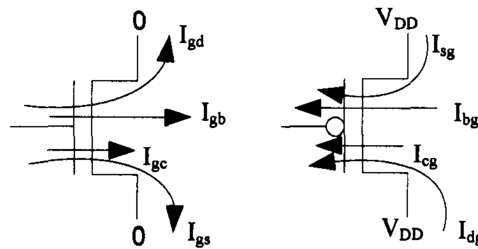


Figure 5. 2: Gate tunneling leakage current in NMOS and PMOS transistors [41]

The gate tunneling current consists of four components: gate-to-channel tunneling (I_{gc}), gate-to-drain edge tunneling (I_{gd}), gate-to-source edge tunneling (I_{gs}), and gate-to-body tunneling (I_{gb}). The magnitude of the gate tunneling depends on the applied voltage Vgs. For NMOS, four possible states exist depending on the voltages of its three CMOS terminals: drain/gate/source = 1/0/0,1/0/1,0/0/1, and 0/1/0. The leakage current under the 0/1/0 state is the highest due to the strong inversion. For PMOS, the current direction and the voltages are symmetric compared with NMOS (as shown in Figure 5.2). Since holes

must pass a higher barrier to tunnel, the PMOS tunneling current is less than the NMOS tunneling current [42,43].

5.2.2 Sub-threshold Leakage Current (I_{SUB})

The sub-threshold leakage is the drain-source current of a transistor during operation in weak inversion [44]. Unlike the strong inversion region in which the drift current dominates, the sub-threshold conduction is due to the diffusion current of the minority carriers in the channel. The magnitude of the sub-threshold current is a function of the temperature, supply voltage, device size, and the process parameters [7]. This relationship is presented by [45]:

$$I_{SUB} = \mu_o C_{ox} \frac{W}{L_{eff}} (\frac{kT}{q})^2 (e^{1.8} e^{\frac{V_{gs} - V_t - \gamma V_{sb} + \eta V_{ds})}{n.(kT/q)}})(1 - e^{-\frac{V_{ds}}{kT/q}})$$

Where V_t, V_{gs}, V_{ds}, V_{sb} are the threshold, gate-source, drain-source, source-bulk voltages, respectively, γ is the linearized body-effect coefficient, η is the DIBL coefficient, (kT/q) is the thermal voltages, n is the sub-threshold swing coefficient of the transistor, μ_0 is the carrier mobility, and W and L_{eff} are the effective gate width and length of the device. The sub-threshold leakage current increases exponentially with lower V_t, and higher temperature (T).

5.2.3 Reverse-bias source/drain junction leakages (I_{REV})

Though the p–n junctions between the source/drain and the substrate are reverse-biased, yet a small amount of current flows causing these junctions to leak [38]. This current is called reverse biased junction leakage current. The magnitude of this current depends on the area of the source/drain diffusion and the doping concentration. The highly doped

shallow junctions and halo doping necessary to control Short Channel Effect in the Ultra-deep-submicron devices has escalated this leakage current [39]. Under this situation, electrons tunnel across the p–n junction causing junction leakage. The tunnelling current density resulting from this leakage phenomenon is given by [39]:

$$J_b = A\frac{EV_{app}}{E_g^{0.5}}\exp(-\frac{BE_g^{3/2}}{E}), A = \frac{\sqrt{2m^*}q^3}{4\pi^3\hbar^2}; B = \frac{4\sqrt{2m^*}}{3q\hbar}$$

where m^* is the effective mass of electron, E_g is the energy bandgap; V_{app} is the applied reverse bias; E is the electric field at the junction; q is the electric charge; ħ is $1/2\pi$ times Planck's constant.

5.2.4 Gate induced drain leakage (I_{GIDL})

This leakage current is caused by high electric field effect in the drain junction of MOS transistors [40,46]. Over the years, transistor scaling has led to increasingly steep halo implants, where the substrate doping at the junction interfaces is increased, while the channel doping is low [44]. Its purpose is to control punch-through and drain induced barrier lowering with minimal impact on the mobility of the carrier in the channel. The steep doping profile that results at the drain edge increases the band-to-band tunneling currents there, especially as drain–bulk voltage (V_{db}) is increased. Thinner oxide and higher supply voltage increase GIDL current. Controlling the doping concentration in the drain of the transistor is the best way to control GIDL.

5.2.5 Gate current due to hot-carrier injection (I_H)

This leakage current is due to the drift, over time, of the threshold voltage in short channel devices [1]. The high electric field near the Si–SiO2 interface can cause electrons

or holes to gain sufficient energy to overcome the interface potential and enter into the oxide layer. In this phenomenon known as hot carrier effect, the electron injection is more likely to occur than the hole as electrons have both lower effective mass and smaller barrier height than holes [1]. These carriers trapped in the oxide layer change the threshold voltage of the device and consequently the subthreshold current. Proportionate scaling down of the supply voltage with the device dimension is one possible way of controlling this leakage.

5.3 Leakage control techniques

Among the different leakage currents in the deep-submicron devices, the sub-threshold and gate leakage are the most dominant. While the latter is mainly due to electron tunneling from the gate to the substrate, the former is caused by many other factors. As a result, the leakage control techniques to be discussed will focus more on subthreshold currents. Over the years, many techniques have been developed to reduce the subthreshold currents in both the active and standby modes of the CMOS circuits in order to minimize the total power consumption. Generally, reduction of leakage currents involves application of different device and circuit level techniques. At the device level, it involves controlling the doping profiles and physical dimensions of transistors while at the circuit level, it involves the manipulation of the threshold voltage V_t and source biasing of the transistor [38].

5.3.1 Circuit level leakage control techniques

Literature review of different circuit level leakage control techniques are described in this section.

5.3.1.1 Dual threshold method:

In logic circuits, leakage current can be reduced by assigning higher V_t to devices in non-critical paths, while maintaining performance with low V_t in the critical paths [38]. This technique is applicable to both standby and active mode leakage power dissipation control. It ensures that the circuit operates at a high speed and reduced power dissipation.

5.3.1.2 Variable threshold method:

This method is mainly used to reduce standby leakage currents by using a triple well process where the device V_t is dynamically adjusted by biasing the body terminal [39,45,46]. Through application of maximum reverse biasing during the standby mode, V_t is increased and the subthreshold leakage current minimized. In addition, this method could be applied in active mode operation to optimize circuit performance by dynamically tuning the V_t based on workload requirements [45]. Through this tuning capability, the circuit can operate at a minimal active leakage power.

5.3.1.3 Power supply gating:

The power supply gating tries to cut off the power from the circuit completely such that virtually there are no leakage paths from power to ground, by inserting an extra transistor switch. In [47], transistor switch with high threshold voltage was used to separate the virtual power and ground rail from the power supply lines. The switch will be turned on/off correspondingly for active/standby mode.

5.3.1.4 Input vector control

The input vector control techniques is applied to reduce the leakage current at the circuit level with little or no performance overhead [48,49]. It is based on the well-known transistor stack effect: a CMOS gates's subthreshold leakage current varies dramatically with the input vector applied to the gate [48]. As the operational state of the transistors that constitute a CMOS gate are determined by their input signal values, the goal can be expressed as finding the input pattern that maximizes the number of disabled ("off") transistors in all stacks across the circuit [50]. Other than the input vector control, the threshold voltage adjustment and the power supply gating require the modification of the logic gates structure and special fabrication process to support multiple threshold transistor fabrication. In addition, the input vector control does not have the speed penalty, while the threshold voltage adjustment and power supply gating do have a significant impact on the circuit delay. A comparison of these three techniques was done in [49] and the results are summarized in Table 5.2.

Table 5. 2: Comparison of different leakage reduction techniques

Techniques	Multiple V_t	Delay Penalty	Leakage Reduction (%)
Dual Threshold	Required	Small	64.1
Power supply gating	Required	Large	~100
Input vector control	Not required	No	75.8

Comparison studies show that the leakage reduction by controlling the input vector in the network is the best if we want to use the traditional fabrication process and do not want to have delay penalty.

5.3.2 Power Challenges at 65 nm

As semiconductor technology has been moving towards smaller geometries and higher system speeds, the rising in dynamic power have been manageable because the core voltage has been dropping with each node. This, along with smaller parasitic capacitances (associated with the smaller transistors) and shorter, less capacitive interconnects between logic, reduced the rate of increase of dynamic power. Moving to the 65-nm process node provides the benefits associated with smaller process geometries: lower cost, higher performance, and greater logic capacity. However, static power is growing exponentially due to increasing transistor leakage. Fig. 5.3 shows the cross-over point, where static power overtakes dynamic power, predicted at 65 nm [51].

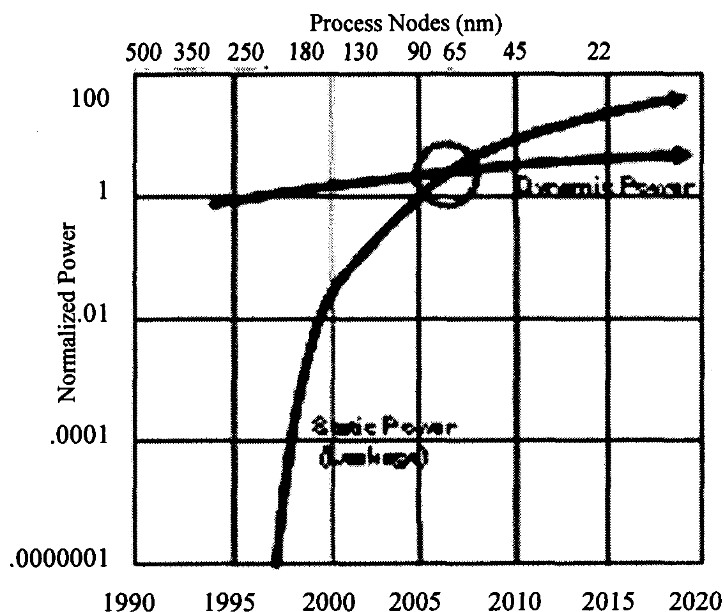


Figure 5. 3: Static and Dynamic Power vs. Process Nodes [51]

At each CMOS process node, since 0.13-μm, there is a specific technology family optimized for low-power applications. 65-nm low power (LP) technology family targets applications in the portable and consumer market such as digital video recorder (DVR),

handsets, and portable media players. To provide the lowest static and dynamic power consumption, the LP process has been fine tuned for low power including low leakage through the use of multiple threshold voltages, multiple I/O voltage transistors, and variable gate-length transistors. LP devices use a thicker gate oxide than general-purpose (GP) devices to decrease standby gate leakage exponentially while only trading off some performance. In addition, LP devices are also using a tightly integrated process and design technology method by providing libraries, IP, and design reference flow optimized for low power.

Analyses of different techniques to reduce the power dissipation with acceptable time delays of the SBox circuit, using 65nm CMOS technology, are presented in the following sub-sections.

5.4 SBox circuit simulation in 65nm CMOS technology

Different transistor models available in the 65nm technology were used to analyze the performance of the SBox. Table 5.3 shows the simulation results. Eight input signals with the sequence of 85, E3, 94, F2, 0D, 6B, 1C and 7A having a pulse width of 3ns were used in the simulation.

Table 5. 3 : Delay and average power of the SBox

Transistors Model	1st ns	2nd ns	3rd ns	4th ns	5th ns	6th Ns	7th ns	8th ns	Average Power (W)
Standard Vt (Svt) for Low Power (LP)	1.764	2.455	2.744	2.35	2.5	2.84	**2.84**	2.4	121 μ
Low Vt (Lvt) for Low Power (LP)	1.208	1.674	1.858	1.55	1.7	1.87	**2.01**	1.6	128.9 μ
Standard Vt (Svt) for General Purpose (GP)	0.555	0.703	0.872	0.669	0.75	0.8	**0.84**	0.76	191 μ

SBox design using general purpose, standard threshold (Svt-GP) model shows the minimum delays of less than 1ns compared to low power models. Table 5.4 shows the results of the simulations where eight input signals were considered with a pulse width of 1ns.

Table 5. 4: Simulation result having input at every 1ns

Transistors Model	1st ns	2nd ns	3rd ns	4th ns	5th ns	6th ns	7th ns	8th ns	Average Power (W)	Energy * Delay
Svt-GP	0.555	0.703	0.821	0.672	0.749	0.798	**0.847**	0.748	553.3 μ	396.9

Adding tri-state buffers between the Inverter and Multiplier stages eliminated the glitches and significantly reduced the power dissipation and the time delays (see section 4.4). Simulation of SBox with tri-state buffers was performed to check the improvement. Table 5.5 shows the time delays and average power dissipation for the same set of eight input signals.

Table 5. 5: Simulation result of SBox circuit with tri-state buffer and input signals with 1ns pulse width

TII Enabled	Delay (ns)								Average Power (W)	Energy * Delay
	1st	2nd	3rd	4th	5th	6th	7th	8th		
0.5 ns	0.825	**0.922**	0.742	0.735	0.819	0.789	0.795	0.844	366.3μ	311.3

A reduction of energy * delay product of about 21.5% reflects the improvement of the design due to the use of TIIs.

Simulation results with low power transistor models have a good impact on the average power dissipation at the expense of longer time delays (table 5.3). Another simulation was performed to check the minimum power that can be achieved by using LVt-Lp in the non-critical path [A square and 4bitXOR units (marked as LVdd in the Fig. 4.16)].

Table 5. 6: Simulation result using dual Vt

Design	1st ns	2nd ns	3rd ns	4th ns	5th ns	6th ns	7th ns	8th ns	Average Power (W)	Energy * Delay
Without TII	0.555	0.713	0.806	0.684	0.757	0.826	**0.912**	0.707	515.4 μ	428.6
With TII	0.826	**0.921**	0.742	0.735	0.819	0.789	0.811	0.844	354.5μ	300.7

The simulation results (Table 5.5 and 5.6) show about 3.6% improvement of the energy * delay product using the dual Vt approach with TII.

5.5 Summary

Performance of the SBox was tested using different types of transistors in CMOS 65nm technology. Among all the combinations, the design based on Svt-gp gave the best result. Modification of the design by adding TIIs considerably reduced the energy * delay product by 21.5% for input signals with a frequency of 1GHz. Without causing any effect to the time delays, simulation result using LVt-LP transistor model in the non-critical path improved 3.6% of the average power dissipation

Chapter 6

Conclusion and Future Work

6.1 Summary of Contributions

This chapter summarizes the contribution made in this thesis and the conclusions from the obtained results. It also discusses the scope for further research in the light of the presented work.

The objective of this thesis is to design nanoelectronic circuits for cryptography. Signals at different stages need to be stored to have a proficient cryptographic system. Hence, an efficint nanoelectronic memory with an improved control circuit is proposed in this work. A Power efficient design of SBox using deep sub-micron CMOS technology is presented in this thesis.

The main contributions made in this thesis are as follows:

In chapter 2, CMOL architecture is reviewed followed by the design of a nanoelectronic memory. Simulation of the memory cell, using the existing CMOL architecture, showed that the required voltages to turn the nano-device ON and OFF were not possible due to

the voltage drop at the pass transistors. The proposed CMOL memory cell overcomes the limitations of current cell designs used for memory arrays. A modified design of multiplexer is presented in this chapter. Around 46% reduction on power dissipation was possible by the modification of the control circuit. A suggested replacement algorithm increases the yield by replacing defective CMOL cells with working unused ones even if located outside the domain region.

Chapter 4 described all the design components of the SBox. Simulations were performed to check the achievable minimum time delays and power dissipation using 0.18μm CMOS technology. Modification of the design by adding a Tri-state buffer stages significantly reduced the average power by 27% and the delay by 18%. Proper Tri-state buffers sizing were achived taking into account the loading effect. Optimum timing setup for the TIIs was checked to get the minimum delay and power dissipation. Lowering the supply voltage and applying dual voltage sources (high Vdd in the Critical path and low Vdd in the non-critical path) were used to achieve low power dissipation. Choosing 1.5V as the Vdd for the non-critical path gave the best compromise of lower power dissipation and higher time delays.

Chapter 5 presented the deep sub-micron implementation of the SBox. Simulations using different types of transistors were performed in 65nm CMOS technology. Among all the combinations, the design using General-purpose Standard V_t (Svt-gp) gave the best result in terms of energy delay product. Modification of the design by adding TIIs considerably reduced the energy * delay product by 21.5% at an input frequency of 1 GHz. Energy *

delay product was improved by 3.6% by using General-purpose Standard V_t (Svt-Gp) and Low-power Low V_t (Lvt-Lp) combinations with Tri-state Buffers.

6.2 Future Work

The scope for further research, in the light of the work presented, is follows:

1) Replacing the nano-device by the Buckyball (C_{60}), a new carbon structure discovered in 1985 [23], might be a soluation for a low-voltage operational nano-electronic memory. The possible area of future work is to analysis and design of non-volatile, high speed nanowire-Buckyball-nanowire memory cell.
2) Implementation of the designed SBox using CMOL architecture
3) Investigating area analysis of the SBox design with physical layout design

Reference:

[1] Ndubuisi Ekekwe, Ralph Etienne-Cummings, "Power dissipation sources and possible control techniques in ultra deep submicron CMOS technologies", Microelectronics Journal 37 (2006) 851- 860

[2] International Technology Roadmap for Semiconductors (ITRS) 2006 Available online at http://public.itrs.net/

[3] K. Likharev and D. Strukov, "CMOL: devices, circuits, and architectures," Introducing MolecularElectronic, Berlin: Springer at press (preprint available online at http://rsfq1.physics.sunysb.edu/~likharev/nano/Springer04.pdf)

[4] Strukov, Dmitri B.; Likharev, Konstantin K, "Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories", Journal of Nanoscience and Nanotechnology, Volume 7, Number 1, pp. 151-167(17), January 2007

[5] Daemen, J, and Rijmen, V.: AES Proposal Rijndael, National Insttude of standards and Technology, july 2001, at www.daimi.au.dk/~ivan/rijndael.pdf

[6] Massoud Masoumi, Farshid Raissi, and Mahmoud Ahmadian, "NanoCMOS-Molecular Realization of Rijndael", CHES 2006, LNCS 4249, pp 285-297, 2006.

[7] Chi-ying Tsui_, Robert Yi-Ching Au, Ricky Yiu-kee Choi," Minimizing the dynamic and sub- threshold leakage power consumption using least leakage vector-assisted technology mapping", INTEGRATION, the VLSI journal 41 (2008) 76–86

[8] Morioka, S. and Satoh, A. 2003. An Optimized S-Box Circuit Architecture for Low Power AES Design. In *Revised Papers From the 4th international Workshop on Cryptographic Hardware and Embedded Systems* (August 13 - 15, 2002). B. S. Kaliski, Ç. K. Koç, and C. Paar, Eds. Lecture Notes In Computer Science, vol. 2523. Springer-Verlag, London, 172-186.

[9] Likharev, K.K., and Strukov, "Defect-Tolerant Architectures for Nanoelectronic Crossbar Memories", "Nanotechnology for Information Storage", August 2006

[10] George W. Hanson, "Fundamentals of Nanoelectronics", Pearson Prentice Happ, pp 181- 300, 2007

[11] K. Likharev: Electronics below 10 nm. In: Nano and Giga Challenges in Microelectronics (Elsevier, Amsterdam), pp. 27- 68, 2003

[12] D. B. Strukov and K. K. Likharev, "A Reconfigurable Architecture for Hybrid CMOS/Nanodevice Circuits," FPGA 06, Page(s): 131-140, Feb. 2006

[13] DeHon, A.; Likharev, K.K., "Hybrid CMOS/nanoelectronic digital circuits: devices, architectures, and design automation," Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on , vol., no., pp. 375-382, 6-10 Nov. 2005

[14] D. B. Strukov and K. K. Likharev, "CMOL FPGA: A Reconfigurable Architecture for Hybrid Digital Circuits with Two-terminal Nanodevices," Nanotechnology, vol. 16, pp. 888- 900, Apr. 2005.

[15] Strukov, D.B., and Likharev, K.K.: 'Prospects for terabit-scale nanoelectronic memories", Nanotechnology 16 (2006) pp. 137-148

[16] E. Lortscher, J. M. Tour, J. W. Ciszek and H. Riel, "A Single-Molecule Switch and Memory Element", 2007 J. Phys.: Conf. Ser. 61 987-991 doi:10.1088/1742 596/61/1/195

[17] "Coulomb blockade", from Wikipedia, the free encyclopedia at http:// en.wikipedia.org/wiki/Coulomb_blockade

[18] Folling, S.; Turel, O.; Likharev, K., "Single-electron latching switches as nanoscale synapses," Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on , vol.1, no., pp.216-221 vol.1, 2001

[19] Chao Li, Wendy Fan, Bo Lei, Daihua Zhang, "Multilevel memory based on molecular devices", Applied Physics lLetters, Vol. 84, Num. 11, 15 March 2004

[20] Chao Li, Bo Lei, Wendy Fan, Daihua Zhang, "Molecular Memory Based on Nanowire-Molecular Wire Heterostructures", Journal of Nanoscience and Nanotechnology, Vol. 7, 138-150, 2007

[21] E. L¨ortscher, J.W. Ciszek, J. M. Tour, and H. Riel. "Reversible and controllable switching of a single-molecule junction". Small, 2(7/8): 973–977, 2006.

[22] Berkeley predictive technology model files for future transistor and interconnect technologies. http://www.eas.asu.edu/~ptm/

[23] Weste H.E. Neil, Harris David, "CMOS VlSI Design: A circuit and systems perspective", 3/E, Pearson Education, pp 103, 713-720, 2005.

[24] Brehob, M, Enbody, R.J., "The Potential of Carbon-based Memory Systems," 1999 IEEE International Workshop on Memory Technology, Design, and Testing, August 1999, p.110-114.

[25] Sabbir Mahmud, "A Study on Parallel Implementation of Advanced Encryption Standard (AES)", MS thesis, Independent University, Bangladesh, May 27, 2004

[26] Menezes, A. and Vanstone, S. "Handbook of Applied Cryptography", CRC Press, Inc. 1996

[27] Coppersmith, D. "The Data Encryption Standard (DES) and Its Strength Against Attacks." IBM Journal of Research and Development, May 1994.

[28] Diffie, W. and Hellman, M. "Multiuser Cryptographic Techniques" proceedings of AFIPS National Computer Conference, 1976, 109-112

[29] Daemon, J. and Rijmen, V. "The Rijndael Block Cipher: AES Proposal", NIST, Version 2, March 1999.

[30] Stallings, W. "Cryptography and Network Security: Principles and Practices." Third Edition, Pearson Education, Inc. 2003.

[31] Lidl, R., and Niederreiter, H. "Introduction to finite fields and their applications" Cambridge University Press, 1986.

[32] Schneier, B. "Applied Cryptgraphy." New York: Wiley, 1996.

[33] Johannes Wolkerstorfer, Elisabeth Oswald and Mario Lamberger, "An ASIC Implmentation of the AES SBoxes", CT-RSA 2002, LNCS 2271, pp 67-78, 2002.

[34] V. Rijmen, Efficient Implementation of the Rijndael SBox, http://www.esat.kuleuven.ac.be/□rijmen/rijndael/ .

[35] C. Paar, Efficient VLSI Architectures for Bit Parallel Computation in Galois Fields, PhD thesis, Universit¨at Essen, 1994.

[36] E. Soljanin and R. Urbanke, "An efficient architecture for implementation of a multiplier and inverter in $GF(2^8)$," Bell Labs, Murray Hill, NJ, Tech. Rep. BL011217-960308-08TM, 1996.

[37] E. D. Mastrovito, VLSI Architectures for Computations in Galois Fields, PhD thesis, Link¨oping University, Link¨oping, Sweden, 1991.

[38] Keating, M., Flynn, D., Aitken, R., Gibbons, A., Shi, K., "Low Power Methodology Manual: For System-on-Chip Design (Series on Integrated Circuits and Systems)", 1st ed. 2007. Corr. 2nd printing, 2007, XVI, ISBN: 978-0-387-71818-7

[39] S. Mukhopadhyay, H. Mahmoodi-Meimand, C. Neau, K. Roy, "Leakage in Nanometer Scale CMOS Circuits, in: International Symposium on VLSI Technology, Systems, and Applications, 2003, pp. 307–312.

[40] K. Roy, S. Mukhopadhyay, H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits", Proc. IEEE 91 (2) (2003) 305–327.

[41] Kyung Ki Kim, Yong-Bin Kim, Minsu Choi, and Nohpill Park," Leakage Minimization Technique for Nanoscale CMOS VLSI Based On Macro-Cell Modeling", IEEE Design &Test of Computers, July-August, 2007, pp 322-330

[42] Shengqi Yang, Wayne Wolf, et al., "Accurate Stacking Macro-modeling of Leakage Power in Sub-100nm Circuits", IEEE VLSID 2005, Page(s) 165-170, 2005.

[43] Dongwoo Lee, et al., "Analysis and Minimization Techniques for Total Leakage Considering Gate Oxide Leakage", IEEE Design Automation Conference(DAC) 2003, Page(s) 175-180, 2003.

[44] F. Fallah, M. Pedram, Standby and active leakage current control and minimization in CMOS VLSI circuits, IEICE Trans. Electron. E88-C(4) (2005) 509–519.

[45] M. Walid, Elgharbawy, A. Magdy, Bayuomi, "Leakage Sources and Possible Solutions in Nanometer CMOS Technologies", IEEE Circ. and Systems Mag. 5 (4) (2005), pp 6-17.

[46] C. H. Kim, K. Roy, "Dynamic Vth Scaling Scheme for Active Leakage Power Reduction", DATE, 2002, pp. 163–167.

[47] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, J. Yamada, "1V high speed digital circuit technology with 0.5m multi-threshold CMOS", in: Proceedings of IEEE ASIC Conference and Exhibit, 1993, pp. 186–189.

[48] M.C. Johnson, D. Somasekhar, K. Roy, "Models and algorithms for bounds on leakage in CMOS circuits", IEEE Trans. CAD, 18 (1999) 714–725.

[49] D. Duarte, Y.-F. Tsai, N. Vijaykrishnan, M.J. Irwin, "Evaluation runtime techniques for leakage power reduction", in: Proceedings of IEEE/ACM Design Automation Conference, 2002, pp. 31– 38.

[50] Ye, Y., Borkar, S., and De, V., "A New Techniques for Standby Leakage Reduction in High Performance Circuits," Proc. Of Symposium on VLSI circuits, 1998, pp. 40-41

[51] Altera Corporation, "Achieving Low Power in 65-nm Cyclone III FPGAs" April 2007, ver. 1.1

Appendix

Berkely Predictive Technology Model

Website: http://eas.asu.edu/~ptm
For 45nm model file
Leff=17.5nm
Vdd = 1V
Tox =1.1nm
Rdsw =155ohm
Vth=0.3V

```
* Customized PTM 45nm NMOS (Vt=0.3V)

.model nmos nmos level=54

+version=4.0 binunit=1 paramchk=1 mobmod=0
+capmod=2 igcmod=1 igbmod=1 geomod=1
+diomod=1 rdsmod=0 rbodymod=1 rgatemod=1
+permod=1 acnqsmod=0 trnqsmod=0

* parameters related to the technology node
+tnom=27 epsrox=3.9
+eta0=0.0049 nfactor=2.1 wint=5e-09
+cgso=1.1e-10 cgdo=1.1e-10 xl=-2e-08

* parameters customized by the user
+toxe=1.75e-09 toxp=1.1e-09 toxm=1.75e-09 toxref=1.75e-09
+dtox=6.5e-10 lint=3.75e-09
+vth0=0.559 k1=0.606 u0=0.0379 vsat=147390
+rdsw=155 ndep=4.32e+18 xj=1.4e-08

* secondary parameters
+ll=0 wl=0 lln=1 wln=1
+lw=0 ww=0 lwn=1 wwn=1
+lwl=0 wwl=0 xpart=0
+k2=0.01 k3=0
+k3b=0 w0=2.5e-006 dvt0=1 dvt1=2
+dvt2=-0.032 dvt0w=0 dvt1w=0 dvt2w=0
+dsub=0.1 minv=0.05 voffl=0 dvtp0=1.0e-009
+dvtp1=0.1 lpe0=0 lpeb=0
+ngate=2e+020 nsd=2e+020 phin=0
+cdsc=0.000 cdscb=0 cdscd=0 cit=0
+voff=-0.13 etab=0
+vfb=-0.55 ua=6e-010 ub=1.2e-018
+uc=0 a0=1.0 ags=1e-020
+a1=0 a2=1.0 b0=0 b1=0
```

```
+keta=0.04 dwg=0 dwb=0 pclm=0.04
+pdiblc1=0.001 pdiblc2=0.001 pdiblcb=-0.005 drout=0.5
+pvag=1e-020 delta=0.01 pscbe1=8.14e+008 pscbe2=1e-007
+fprout=0.2 pdits=0.08 pditsd=0.23 pditsl=2.3e+006
+rsh=5 rsw=85 rdw=85
+rdswmin=0 rdwmin=0 rswmin=0 prwg=0
+prwb=6.8e-011 wr=1 alpha0=0.074 alpha1=0.005
+beta0=30 agidl=0.0002 bgidl=2.1e+009 cgidl=0.0002
+egidl=0.8

+aigbacc=0.012 bigbacc=0.0028 cigbacc=0.002
+nigbacc=1 aigbinv=0.014 bigbinv=0.004 cigbinv=0.004
+eigbinv=1.1 nigbinv=3 aigc=0.012 bigc=0.0028
+cigc=0.002 aigsd=0.012 bigsd=0.0028 cigsd=0.002
+nigc=1 poxedge=1 pigcd=1 ntox=1

+xrcrg1=12 xrcrg2=5
+cgbo=2.56e-011 cgdl=2.653e-10
+cgsl=2.653e-10 ckappas=0.03 ckappad=0.03 acde=1
+moin=15 noff=0.9 voffcv=0.02

+kt1=-0.11 kt1l=0 kt2=0.022 ute=-1.5
+ua1=4.31e-009 ub1=7.61e-018 uc1=-5.6e-011 prt=0
+at=33000

+fnoimod=1 tnoimod=0

+jss=0.0001 jsws=1e-011 jswgs=1e-010 njs=1
+ijthsfwd=0.01 ijthsrev=0.001 bvs=10 xjbvs=1
+jsd=0.0001 jswd=1e-011 jswgd=1e-010 njd=1
+ijthdfwd=0.01 ijthdrev=0.001 bvd=10 xjbvd=1
+pbs=1 cjs=0.0005 mjs=0.5 pbsws=1
+cjsws=5e-010 mjsws=0.33 pbswgs=1 cjswgs=3e-010
+mjswgs=0.33 pbd=1 cjd=0.0005 mjd=0.5
+pbswd=1 cjswd=5e-010 mjswd=0.33 pbswgd=1
+cjswgd=5e-010 mjswgd=0.33 tpb=0.005 tcj=0.001
+tpbsw=0.005 tcjsw=0.001 tpbswg=0.005 tcjswg=0.001
+xtis=3 xtid=3

+dmcg=0e-006 dmci=0e-006 dmdg=0e-006 dmcgt=0e-007
+dwj=0.0e-008 xgw=0e-007 xgl=0e-008

+rshg=0.4 gbmin=1e-010 rbpb=5 rbpd=15
+rbps=15 rbdb=15 rbsb=15 ngcon=1
```

*** Customized PTM 45nm PMOS, Vt=-0.3V**

```
.model pmos pmos level=54

+version=4.0 binunit=1 paramchk=1 mobmod=0
+capmod=2 igcmod=1 igbmod=1 geomod=1
+diomod=1 rdsmod=0 rbodymod=1 rgatemod=1
+permod=1 acnqsmod=0 trnqsmod=0

* parameters related to the technology node
+tnom=27 epsrox=3.9
+eta0=0.0049 nfactor=2.1 wint=5e-09
+cgso=1.1e-10 cgdo=1.1e-10 xl=-2e-08

* parameters customized by the user
+toxe=1.85e-09 toxp=1.1e-09 toxm=1.85e-09 toxref=1.85e-09
+dtox=7.5e-10 lint=3.75e-09
+vth0=-0.492 k1=0.553 u0=0.00337 vsat=70000
+rdsw=155 ndep=3.21e+18 xj=1.4e-08

*secondary parameters
+ll=0 wl=0 lln=1 wln=1
+lw=0 ww=0 lwn=1 wwn=1
+lwl=0 wwl=0 xpart=0
+k2=-0.01 k3=0
+k3b=0 w0=2.5e-006 dvt0=1 dvt1=2
+dvt2=-0.032 dvt0w=0 dvt1w=0 dvt2w=0
+dsub=0.1 minv=0.05 voffl=0 dvtp0=1e-009
+dvtp1=0.05 lpe0=0 lpeb=0
+ngate=2e+020 nsd=2e+020 phin=0
+cdsc=0.000 cdscb=0 cdscd=0 cit=0
+voff=-0.126 etab=0
+vfb=0.55 ua=2.0e-009 ub=0.5e-018
+uc=0 a0=1.0 ags=1e-020
+a1=0 a2=1 b0=-1e-020 b1=0
+keta=-0.047 dwg=0 dwb=0 pclm=0.12
+pdiblc1=0.001 pdiblc2=0.001 pdiblcb=3.4e-008 drout=0.56
+pvag=1e-020 delta=0.01 pscbe1=8.14e+008 pscbe2=9.58e-007
+fprout=0.2 pdits=0.08 pditsd=0.23 pditsl=2.3e+006
+rsh=5 rsw=85 rdw=85
+rdswmin=0 rdwmin=0 rswmin=0 prwg=3.22e-008
+prwb=6.8e-011 wr=1 alpha0=0.074 alpha1=0.005
+beta0=30 agidl=0.0002 bgidl=2.1e+009 cgidl=0.0002
+egidl=0.8

+aigbacc=0.012 bigbacc=0.0028 cigbacc=0.002
+nigbacc=1 aigbinv=0.014 bigbinv=0.004 cigbinv=0.004
+eigbinv=1.1 nigbinv=3 aigc=0.69 bigc=0.0012
+cigc=0.0008 aigsd=0.0087 bigsd=0.0012 cigsd=0.0008
+nigc=1 poxedge=1 pigcd=1 ntox=1

+xrcrg1=12 xrcrg2=5
+cgbo=2.56e-011 cgdl=2.653e-10
+cgsl=2.653e-10 ckappas=0.03 ckappad=0.03 acde=1
+moin=15 noff=0.9 voffcv=0.02

+kt1=-0.11 kt1l=0 kt2=0.022 ute=-1.5
```

```
+ua1=4.31e-009 ub1=7.61e-018 uc1=-5.6e-011 prt=0
+at=33000

+fnoimod=1 tnoimod=0

+jss=0.0001 jsws=1e-011 jswgs=1e-010 njs=1
+ijthsfwd=0.01 ijthsrev=0.001 bvs=10 xjbvs=1
+jsd=0.0001 jswd=1e-011 jswgd=1e-010 njd=1
+ijthdfwd=0.01 ijthdrev=0.001 bvd=10 xjbvd=1
+pbs=1 cjs=0.0005 mjs=0.5 pbsws=1
+cjsws=5e-010 mjsws=0.33 pbswgs=1 cjswgs=3e-010
+mjswgs=0.33 pbd=1 cjd=0.0005 mjd=0.5
+pbswd=1 cjswd=5e-010 mjswd=0.33 pbswgd=1
+cjswgd=5e-010 mjswgd=0.33 tpb=0.005 tcj=0.001
+tpbsw=0.005 tcjsw=0.001 tpbswg=0.005 tcjswg=0.001
+xtis=3 xtid=3

+dmcg=0e-006 dmci=0e-006 dmdg=0e-006 dmcgt=0e-007
+dwj=0.0e-008 xgw=0e-007 xgl=0e-008

+rshg=0.4 gbmin=1e-010 rbpb=5 rbpd=15
+rbps=15 rbdb=15 rbsb=15 ngcon=1
```

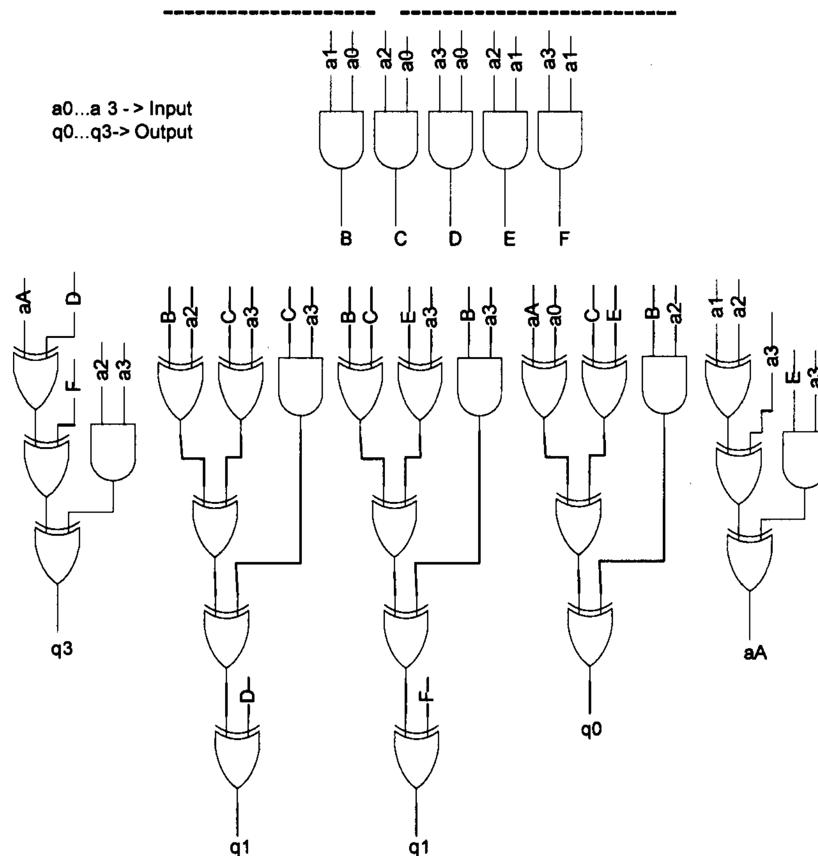


Figure 1: An Improved design of the Inversion (Ref. Fig. 4.4) with 1AND +4XOR critical path delay)

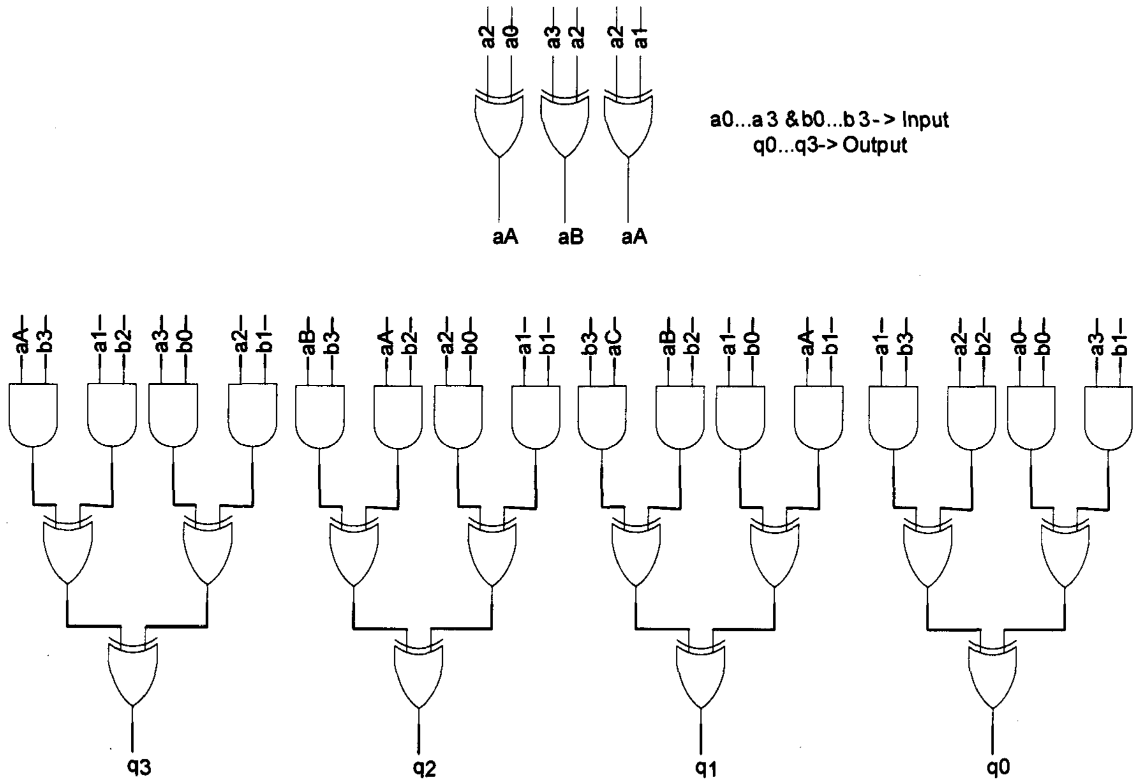


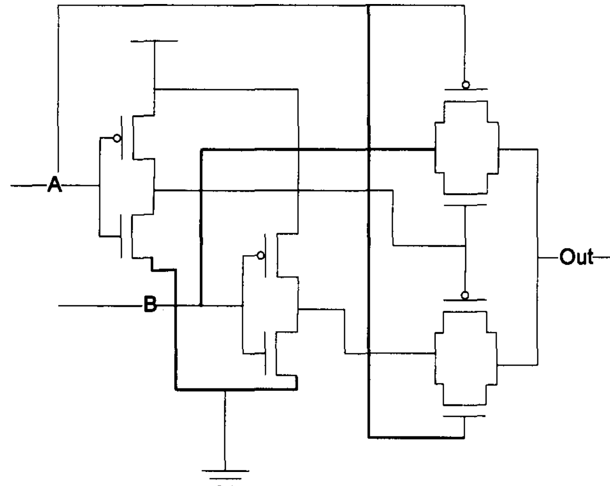
Figure 2: An Improved design of the Multiplication (Ref. Fig 4.5) (q = a ⊗ b) with 1AND +3XOR critical path delay

Table 1: AES SBox Area and Critical Path Measurements

Block	Meaning	Area	Delay
map	Isomorphic mapping	11 XOR	3 XOR
Squ	Squaring in GF (24)	2 XOR	1 XOR
⊗	Multiplier in GF (24)	15 XOR + 16 AND	1 AND + 3 XOR
Inv	Multiplicative inverse in GF (24)	20 XOR + 10 AND	3 XOR
Mapinv	Inverse isomorphic mapping	15 XOR	3 XOR
aff_trans	Affine transformation	20 XOR	3 XOR
Aff_trans-1	Inverse affine transformation	12 XOR	3 XOR

Transistors Sizing

1. XOR Gate



Width of all NMOS, Wn =4λ
Width of all PMOS, Wp=8λ

2. AND Gate

Pull up network, Wp = 13λ
Pull downnetwork, Wn=6λ
NMOS used in inverter = 5λ
PMOS used in inverter=12λ

3. Inverter

Pull up network, Wp=10λ
Pull down network, Wn =5λ